NATURAL HAZARDS RISKS AND RESILIENCY
RESEARCH CENTER (NHR3)

Development of Earthquake Ground-Motion Tools for End-Users of the NGA-Subduction and NGA-East Ground-Motion Models

Silvia Mazzoni and **Yousef Bozorgnia**

Natural Hazards Risk and Resiliency Research Center (NHR3), and

Department of Civil and Environmental Engineering

University of California, Los Angeles (UCLA)

A Final Report to the US Geological Survey (USGS)

On Research Grant Number G19AP00026

Principal Investigator: Yousef Bozorgnia

October 31, 2020

ABSTRACT

This report describes the development of earthquake ground-motion tools for end-users of the NGA-Subduction (NGA-Sub) and NGA-East ground-motion models (GMMs). We developed computer tools for NGA-East and NGA-Sub in multiple platforms: Excel coded in Visual Basic, “R”, MATLAB, and Python. For each of NGA-East and NGA-Sub, the results of these computer codes were compared to make sure there was no discrepancy in the results based on multiple codes. Such checks were carried out for various combinations of scenarios. This quality check can minimize the coding mistakes by end-users of the NGA-East and NGA-Sub GMMs. The report provides a summary, as well as details, of the codes in all platforms. Electronic Attachments include all computer codes and the necessary data to generate the GMMs results. The NGA-East and NGA-Sub computer tools have been posted for public use at the UCLA Natural Hazards Risk and Resiliency Research Center (NHR3) web site.

ACKNOWLEDGMENTS

This material is based upon work supported by the U.S. Geological Survey (USGS) under Grant No G19AP00026 with the University of California, Los Angeles (UCLA). The support of USGS is gratefully acknowledged. Core support for the NGA-Sub research project was provided by FM Global, the USGS, the California Department of Transportation, and the Pacific Gas & Electric Company. The support of these organizations is gratefully appreciated. The views, opinions, findings, conclusions, or recommendations expressed in this document are those of the authors and should not should not be interpreted as representing the opinions or policies of the U.S. Geological Survey, other sponsoring agencies, the Natural Hazards Risk and Resiliency Research Center (NHR3), UCLA, or the Regents of the University of California. Mention of trade names or commercial products does not constitute their endorsement by the U.S. Geological Survey.

Constructive comments provided by the NGA-Sub group, especially Drs. Nico Kuehn, Grace Parker, Jon Stewart and Nick Gregor are gratefully appreciated.

Table of Contents

ABSTRACT i

ACKNOWLEDGMENTS ii

Table of Contents iii

1. Introduction 1

2. Summary of the NGA-East Ground-Motion Characterization Tool 2

3. Summary of the NGA-Sub Ground-Motion Characterization Tool 2

4. Details of NGA-East GMM Tools 4

4.1 NGA-East Ground-Motion Characterization Tool – MS Excel 4

4.2 NGA-East Ground-Motion Characterization Tool – "R" 12

4.3 NGA-East Ground-Motion Characterization Tool – MATLAB 48

4.4 NGA-East Ground-Motion Characterization Tool – Python 84

5. Details of NGA-Sub GMM Tools 119

5.1 NGA-Sub Ground-Motion Characterization Tool – MS Excel 120

5.2 NGA-Sub Ground-Motion Characterization Tools – "R" 188

5.3 NGA-Sub Ground-Motion Characterization Tools – MATLAB 248

5.4 NGA-Sub Ground-Motion Characterization Tools – Python 313

6. References 377

Electronic Attachment A: Computer codes and data needed for NGA-East GMMs Tool 378

Electronic Attachment B: Computer codes and data needed for NGA-Sub GMMs Tool 378

1. Introduction

The NGA-East and NGA-Subduction (NGA-Sub) ground-motion models (GMMs) published to date have been implemented into a series of computer tools in different platforms and published on-line for public access. The platforms for the implementation consist of the following: Microsoft (MS) Excel, which has a Visual-Basic programmable backend, Matlab, Python, and R. All the ground-motion models compute the mean and aleatory variability of peak ground acceleration (PGA), peak ground velocity (PGV) and pseudo-spectral accelerations (PSA) at (at least) 21 oscillator periods ranging from 0.01 to 10 seconds. For an earthquake scenario selected by the end-users, these NGA "computer tools" provide the estimated intensity measures for individual models, as well as the weighted values of the three percentiles: median and median +/- N standard deviation, where N is defined by the user.

These tools may be downloaded via the Natural Hazards Risk and Resiliency Research Center (NHR3) Ground-Motion Tools portal: https://www.risksciences.ucla.edu/nhr3/gmtools. The on-line portal also includes an instructional video for each version. These videos have basic instructions and demonstrations.

The MS Excel version of each tool is intended to be the most versatile self-contained tool that has a friendly user-interface and does not require programming knowledge. Most users are familiar with the spreadsheet interface of the MS Excel and are able to navigate it with ease. All computations are performed in the background using Visual Basic (VBA) code; thus, the interface is very simple and intuitive. Function in Excel are also very simple and cell values are referenced by name, such that the user is able to reformat the Input/Output framework, if desired. The user may also add his or her own additional computations, formatting, and/or additional plots of the data. The MS Excel file is intended for users who wish to compute scenario spectra and visualize the results.

The other platforms of the tools, namely Matlab, Python, and R, on the other hand, are intended for developers who will integrate individual components of the program into their own program infrastructure. It is important to note that the current implementation loads and reads the data at each period point, making it easy to understand. The computation can speed up by optimizing input/output, but this process is dependent on the overall workflow of the larger program and left to the user. Each version of the tool has its own set of instructions in its first lines of comments, followed by script-input workflow.

Since the syntaxes of the platforms are different than each other, the flow of information also must be different from one platform to another. For example, when the input is run as a file, Matlab requires that the functions be defined after the main core of the program, while the other formats require that the functions be defined before they are used. In MS Excel, the functions are defined internally in the VBA interface and can be stored in separate modules. The Matlab, Python, and R implementations consist of a single script file in ASCII format, along with the necessary data in .csv files, if needed. The .csv files, which contain model coefficients, are independent of the tool format (Matlab, Python or R).

For the NGA-East GMMs are in tabulated format rather than equations; thus, the data tables of the NGA-East GMMs are included in the scripts. The MS Excel version, on the other hand, is self-contained Excel with separate worksheets and VBA modules within it. The data tables are also contained within the MS Excel tool in separate worksheets.

All versions of the tool follow the same basic format: modular functions/subroutines are used as much as possible so that a developer may extract different parts of the program to be integrated elsewhere. The tools are written for a single scenario (Magnitude, distance, etc.) and a list of periods, including PGA and PGV. In all cases, the input values are grouped in the same part of the script and clearly labeled for the user. The output is given in tables formatted with headers. When these tables are too large for screen output, they are exported to a .csv file whose path is given to the user in the screen output. All version, except for the python version, plot the computed response spectra.

The ground-motion tools provide minimal necessary documentation for the input/output parameters. They MS Excel tool provides some additional information that is helpful to the user, such as model limits and options. The MS Excel interface takes advantage of pull-down menus and conditional formatting to reduce the chances that the user input erroneous values. Information about the model documentation is provided in all tools. For each of NGA-East and NGA-Sub, the output of the tools at different platforms have been extensively checked to provide identical results given various scenario cases.

2. Summary of the NGA-East Ground-Motion Characterization Tool

The NGA-East GMMs have been documented in Goulet, et al. (2018). Since the NGA-East GMMs are in tabulated format rather than equations, the ground-motion tool computes the weighted average model for both the median and the sigma through table look-up and interpolation. Both ergodic and Single-Station Sigmas can be computed. In addition to the weighted average, the Excel tool computes the individual models: 17 models for the median (the results of the 17 cells in the NGA-East Sammon Map), and two models for the sigma. All versions of the tool compute the values for the reference site condition of Vs30=3000 m/sec, and for 5% damping.

The input for the NGA-East model consists of the following parameters. Ranges and options for these values are given in the MS Excel version of the Tool.

- Median Model: The user has the option to either obtain the results of one of the 17 models of the NGA-East model, or the Weighted Average of all 17 models.
- Magnitude: Event Magnitude
- Rrup (km): Rupture distance
- σ Type: Single-Station or ergodic Sigma
- σ Model: Weighted Average, Central, High, or Low
- N-sigma: Number of standard deviations to be computed for Median +/- N-sigma times Sigma.

The output of the NGA-East model consists of the predicted PSA: Median, σ, Median - 1*σ , Median + 1*σ. Details of the NGA-East GMMs Tools are provided in the next sections of this report. All computer codes and needed data for NGA-East are zipped into a single Zipped file as an **Electronic Attachment A**.

3. Summary of the NGA-Sub Ground-Motion Characterization Tool

The NGA-Sub GMMs have been documented in Kuehn, et al. (2020) and Parker, et al. (2020). The abbreviated names of these GMMs are KBCG20 and PSHAB20, respectively. The NGA-Sub Ground-Motion Characterization Tool includes the implementation of the two NGA-Sub GMMs in MS Excel, with VBA, MATLAB, Python, and R.

While the NGA-East tool implementation consists of a process of table lookups and interpolation, the NGA-Sub models are in form of equations. The tool computes for each model: median PSA and Sigma, ergodic and aleatory separately. It also computes the weighted average of the median and median +/- N times sigma of the two models, where N is defined by the end-user.

The NGA-Sub Ground-Motion Characterization Tool consists of two parts: the first part computes the predicted values for a single scenario, the second part performs a sensibility study on a combination of variations of the main input parameters. The sensibility study is included in the tool because it was used to validate the results of the different framework implementations and it gives the user an understanding of the behavior of the GMMs.

The MS Excel version is the most comprehensive version of the tool and was developed to be used on its own to perform scenario or sensitivity studies. The MATLAB, Python, and R versions are intended to be integrated into a workflow and do have instructions within each file as comment lines. The data tables, which are stored in separate .csv files, are read at each period point. This was done for simplicity in implementation. The functions are listed by their name and input parameters, but are not given in the table.

The NGA-Sub Ground-Motion Characterization Tools depend on a number of coefficient tables. In the MS Excel version of the tool, they are stored in individual sheets. The MATLAB, Python, and R versions, on the other hand, read these tables from external .csv files, which are included with the scripts. The same set of .csv files is shared by all. Details of the NGA-Sub GMMs Tools are provided in the next sections of this report.

All computer codes and needed data for NGA-Sub are zipped into a single Zipped file as an **Electronic Attachment B**.

4. Details of NGA-East GMM Tools

4.1 NGA-East Ground-Motion Characterization Tool – MS Excel

Below are screenshots of the main-page of the Excel version of the NGA-East GMMs, followed by the VB codes of the NGA-East.

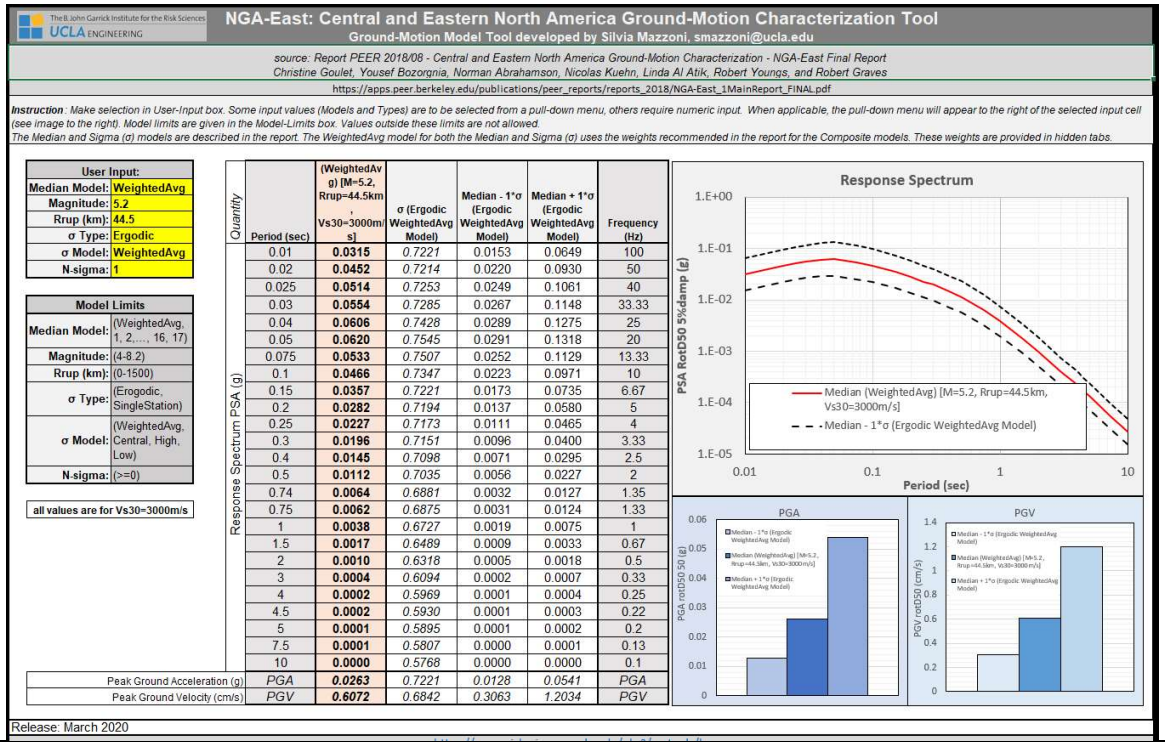
NGA-East_GMMtool_Excel_R200316_Protected.xlsm:

The Excel file has two worksheets.

The first worksheet is the main tab for the User Interface and it is automatically updated with the input values. The user input is highlighted in yellow:

The second worksheet has the Instructions in the first cell, and graphics showing pull-down menu options and the model limits.

Instruction: Make selection in User-Input box. Some input values (Models and Types) are to be selected from a pull-down menu, others require numeric input. When applicable, the pull-down menu will appear to the right of the selected input cell. Model limits are given in the Model-Limits box. Values outside these limits are not allowed.



NGA-East: Central and Eastern North America Ground-Motion Characterization Tool

Ground-Motion Model Tool developed by Silvia Mazzoni, smazzoni@ucla.edu

source: Report PEER 2018/08 - Central and Eastern North America Ground-Motion Characterization - NGA-East Final Report
Christine Goulet, Yousef Bozorgnia, Norman Abrahamson, Nicolas Kuehn, Linda Al Atik, Robert Youngs, and Robert Graves
https://apps.peer.berkeley.edu/publications/peer_reports/reports_2018/NGA-East_1MainReport_FINAL.pdf

Instruction: Make selection in User-Input box. Some input values (Models and Types) are to be selected from a pull-down menu, others require numeric input. When applicable, the pull-down menu will appear to the right of the selected input cell (see image to the right). Model limits are given in the Model-Limits box. Values outside these limits are not allowed.
The Median and Sigma (σ) models are described in the report. The WeightedAvg model for both the Median and Sigma (σ) uses the weights recommended in the report for the Composite models. These weights are provided in hidden tabs.

User Input:	
Median Model:	WeightedAvg
Magnitude:	5.2
Rrup (km):	44.5
σ Type:	Ergodic
σ Model:	WeightedAvg
N-sigma:	1

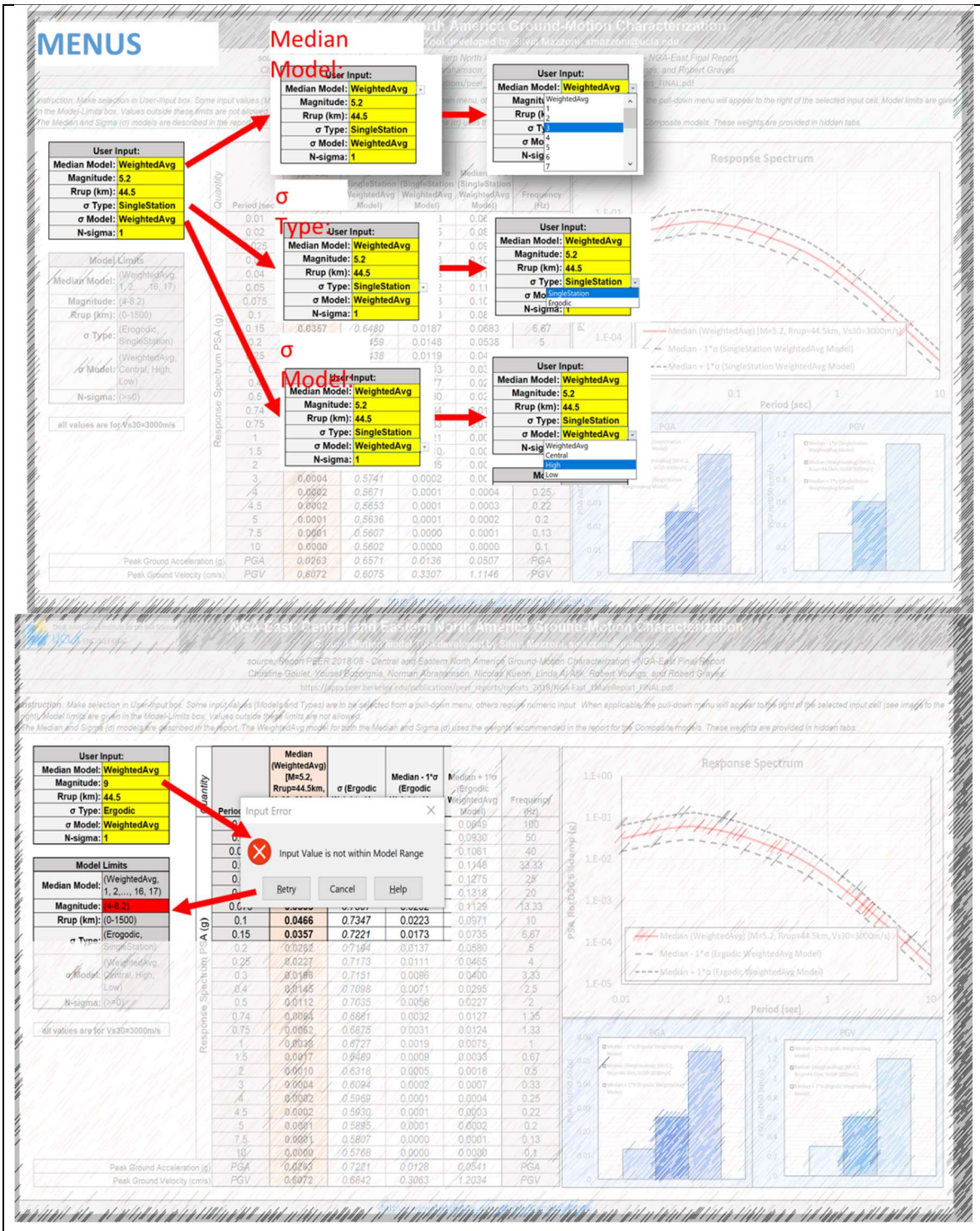
Model Limits	
Median Model:	(WeightedAvg, 1, 2,…., 16, 17)
Magnitude:	(4-8.2)
Rrup (km):	(0-1500)
σ Type:	(Erogodic, SingleStation)
σ Model:	(WeightedAvg, Central, High, Low)
N-sigma:	(>=0)

all values are for Vs30=3000m/s

Quantity: Response Spectrum PSA (g)

Period (sec)	(WeightedAvg) [M=5.2, Rrup=44.5km, Vs30=3000m/s]	σ (Ergodic WeightedAvg Model)	Median - 1*σ (Ergodic WeightedAvg Model)	Median + 1*σ (Ergodic WeightedAvg Model)	Frequency (Hz)
0.01	**0.0315**	*0.7221*	0.0153	0.0649	100
0.02	**0.0452**	*0.7214*	0.0220	0.0930	50
0.025	**0.0514**	*0.7253*	0.0249	0.1061	40
0.03	**0.0554**	*0.7285*	0.0267	0.1148	33.33
0.04	**0.0606**	*0.7428*	0.0289	0.1275	25
0.05	**0.0620**	*0.7545*	0.0291	0.1318	20
0.075	**0.0533**	*0.7507*	0.0252	0.1129	13.33
0.1	**0.0466**	*0.7347*	0.0223	0.0971	10
0.15	**0.0357**	*0.7221*	0.0173	0.0735	6.67
0.2	**0.0282**	*0.7194*	0.0137	0.0580	5
0.25	**0.0227**	*0.7173*	0.0111	0.0465	4
0.3	**0.0196**	*0.7151*	0.0096	0.0400	3.33
0.4	**0.0145**	*0.7098*	0.0071	0.0295	2.5
0.5	**0.0112**	*0.7035*	0.0056	0.0227	2
0.74	**0.0064**	*0.6881*	0.0032	0.0127	1.35
0.75	**0.0062**	*0.6875*	0.0031	0.0124	1.33
1	**0.0038**	*0.6727*	0.0019	0.0075	1
1.5	**0.0017**	*0.6489*	0.0009	0.0033	0.67
2	**0.0010**	*0.6318*	0.0005	0.0018	0.5
3	**0.0004**	*0.6094*	0.0002	0.0007	0.33
4	**0.0002**	*0.5969*	0.0001	0.0004	0.25
4.5	**0.0002**	*0.5930*	0.0001	0.0003	0.22
5	**0.0001**	*0.5895*	0.0001	0.0002	0.2
7.5	**0.0001**	*0.5807*	0.0000	0.0001	0.13
10	**0.0000**	*0.5768*	0.0000	0.0000	0.1
Peak Ground Acceleration (g)	*PGA*	***0.0263***	*0.7221*	*0.0128*	*0.0541*	*PGA*
Peak Ground Velocity (cm/s)	*PGV*	***0.6072***	*0.6842*	*0.3063*	*1.2034*	*PGV*

Release: March 2020


MENUS
Median Model:
σ Type:
σ Model:
User Input:
Median Model: WeightedAvg
Magnitude: 5.2
Rrup (km): 44.5
σ Type: SingleStation
σ Model: WeightedAvg
N-sigma: 1
Model Limits
Median Model: (WeightedAvg, 1, 2, ..., 16, 17)
Magnitude: (4-8.2)
Rrup (km): (0-1500)
σ Type: (Ergodic, SingleStation)
σ Model: (WeightedAvg, Central, High, Low)
N-sigma: (>=0)
all values are for Vs30=3000m/s
Response Spectrum
Input Error
Input Value is not within Model Range
Retry
Cancel
Help

```
Function NGAEast_GMM(thisT, thisM, thisRrup, thisInputLabel, thisModelLabel)
  ' Code written and developed by Silvia Mazzoni, smazzoni@ucla.edu, March 2020
  iLnY = -999
  Dim Frange() As Variant
  If LCase(thisInputLabel) = "ergodic" Then
    Frange = Range("FrequencyListCompErg")
  ElseIf LCase(thisInputLabel) = "singlestation" Then
    Frange = Range("FrequencyListCompSS")
  Else
    Frange = Range("medianLnYrefFrequencyValueColumn" & thisModelLabel)
  End If

  Flist = eliminateDuplicate(Frange)


  thisT = getTFvalue(thisT)

  For iiT = LBound(Flist) To UBound(Flist)
  ' debug.print "-" & iiT
    hereF = Flist(iiT)
    hereF = getTFvalue(hereF)
    ' debug.print "hereF " & hereF
    If hereF < 0 Then
      hereT = hereF
    Else
      hereT = 1 / hereF
    End If
    ' debug.print "hereT " & hereT
    If Abs(hereT - thisT) < 0.000001 Then
      If LCase(thisInputLabel) = LCase("Ergodic") Then
        iLnY = getSigma(hereF, thisM, thisRrup, "Ergodic", thisModelLabel)
      ElseIf LCase(thisInputLabel) = LCase("SingleStation") Then
        iLnY = getSigma(hereF, thisM, thisRrup, "SingleStation", thisModelLabel)
      Else
        iLnY = getLnY(hereF, thisM, thisRrup, thisModelLabel)
      End If
      iiT = UBound(Flist) + 1
    Else
      If iiT < UBound(Flist) Then
        nextF = Flist(iiT + 1)
        nextF = getTFvalue(nextF)
        nextT = 1 / nextF
        ' debug.print "nextF " & nextF
        ' debug.print "nextT " & nextT
        ' debug.print (hereT - thisT) * (nextT - thisT)
        If hereT > 0 And nextT > 0 And (hereT - thisT) * (nextT - thisT) < 0 Then
          ' debug.print "inn"
          If LCase(thisInputLabel) = LCase("Ergodic") Then
            hereLnY = getSigma(hereF, thisM, thisRrup, "Ergodic", thisModelLabel)
            nextLnY = getSigma(nextF, thisM, thisRrup, "Ergodic", thisModelLabel)
          ElseIf LCase(thisInputLabel) = LCase("SingleStation") Then
            hereLnY = getSigma(hereF, thisM, thisRrup, "SingleStation", thisModelLabel)
            nextLnY = getSigma(nextF, thisM, thisRrup, "SingleStation", thisModelLabel)
          Else
            hereLnY = getLnY(hereF, thisM, thisRrup, thisModelLabel)
            nextLnY = getLnY(nextF, thisM, thisRrup, thisModelLabel)
          End If
          thisLnT = Log(thisT)
```

```
            hereLnT = Log(hereT)
            nextLnT = Log(nextT)
            iLnY = hereLnY + (thisLnT - hereLnT) * (nextLnY - hereLnY) / (nextLnT - hereLnT)
            iiT = UBound(Flist) + 1
          End If
        End If
      End If
    Next iiT
    If LCase(thisInputLabel) = LCase("median") And iLnY > -100 Then
      NGAEast_GMM = Exp(iLnY)
    Else
      NGAEast_GMM = iLnY
    End If

End Function

Function getTFvalue(inValue)
    ' Code written and developed by Silvia Mazzoni, smazzoni@ucla.edu, March 2020

    If VarType(inValue) = vbString Then
      If LCase(inValue) = LCase("pga") Then
        getTFvalue = -1
      ElseIf LCase(inValue) = LCase("pgv") Then
        getTFvalue = -2
      End If
    Else
      getTFvalue = inValue
    End If
End Function



Function getLnY(thisFreq, thisMin, thisRrupin, thisMedianModelLabel)
    ' Code written and developed by Silvia Mazzoni, smazzoni@ucla.edu, March 2020
    thisM = thisMin
    thisRrup = thisRrupin

    ' this algorithm assumes thisFreq is one of the frequencies listed, if not, you need to write a wrapper that interpolates
    getLnY = -999
    ' define these intermally
    MagnitudeInterpolationType = "Linear"
    RrupInterpolationType = "Logarithmic"

    medianLnYrefMomentColumn = Range("medianLnYrefMomentColumn" & thisMedianModelLabel)
    MedianLnYrefDataTable = Range("MedianLnYrefDataTable" & thisMedianModelLabel)
    MedianLnYrefRrupRow = Range("MedianLnYrefRrupRow")
    medianLnYrefFrequencyColumn = Range("medianLnYrefFrequencyValueColumn" & thisMedianModelLabel)
    medianLnYrefModelLabelColumn = Range("medianLnYrefModelLabelColumn" & thisMedianModelLabel)

    thisFreq = getTFvalue(thisFreq)


    Mbelow = -888
    Mabove = 888
    For iRow = LBound(medianLnYrefFrequencyColumn, 1) To UBound(medianLnYrefFrequencyColumn, 1)
      hereModelLabel = medianLnYrefModelLabelColumn(iRow, 1)
      If hereModelLabel = thisMedianModelLabel Then

        hereFreq = medianLnYrefFrequencyColumn(iRow, 1)
```

```
    hereFreq = getTFvalue(hereFreq)

    If Abs(hereFreq - thisFreq) < 0.000001 Then
      hereM = medianLnYrefMomentColumn(iRow, 1)
      If Abs(hereM - thisM) < 0.001 Then

        For iColR = LBound(MedianLnYrefRrupRow, 2) To UBound(MedianLnYrefRrupRow, 2)
          hereRrup = MedianLnYrefRrupRow(1, iColR)
          If Abs(hereRrup - thisRrup) < 0.000001 Then
            getLnY = MedianLnYrefDataTable(iRow, iColR) ' nailed it!
            Exit Function
          Else
            If iColR = UBound(MedianLnYrefRrupRow, 2) Then
              getLnY = -888
              Exit Function
            End If


            nextRrup = MedianLnYrefRrupRow(1, iColR + 1)
            If (thisRrup - hereRrup) * (thisRrup - nextRrup) < 0 Then

              If LCase(Left(RrupInterpolationType, 3)) = LCase("log") Then
                If thisRrup = 0 Then
                  thisRrup = 0.0000001
                End If
                If hereRrup = 0 Then
                  hereRrup = 0.0000001
                End If
                If nextRrup = 0 Then
                  nextRrup = 0.0000001
                End If
                hereRrup = Log(hereRrup)
                nextRrup = Log(nextRrup)
                thisRrup = Log(thisRrup)
              End If
              Y0 = MedianLnYrefDataTable(iRow, iColR)
              Y1 = MedianLnYrefDataTable(iRow, iColR + 1)
              getLnY = Y0 + (Y1 - Y0) / (nextRrup - hereRrup) * (thisRrup - hereRrup)
              Exit Function
            End If
          End If
        Next iColR
      Else
        If hereM > Mbelow And hereM < thisM Then
          Mbelow = hereM
          Ibelow = iRow
        End If
        If hereM < Mabove And hereM > thisM Then
          Mabove = hereM
          Iabove = iRow
        End If
      End If
    End If
  End If
Next iRow


For iColR = LBound(MedianLnYrefRrupRow, 2) To UBound(MedianLnYrefRrupRow, 2)
```

```
      hereRrup = MedianLnYrefRrupRow(1, iColR)
      If Abs(hereRrup - thisRrup) < 0.000001 Then
         Z0 = MedianLnYrefDataTable(Ibelow, iColR) ' nailed it!
         Z1 = MedianLnYrefDataTable(Iabove, iColR) ' nailed it!
         iColR = UBound(MedianLnYrefRrupRow, 2) + 1
      Else
         If iColR < UBound(MedianLnYrefRrupRow, 2) Then
            nextRrup = MedianLnYrefRrupRow(1, iColR + 1)
            If (thisRrup - hereRrup) * (thisRrup - nextRrup) < 0 Then
               If LCase(Left(RrupInterpolationType, 3)) = "log" Then
                  If thisRrup = 0 Then
                     thisRrup = 0.1
                  End If
                  If hereRrup = 0 Then
                     hereRrup = 0.0000001
                  End If
                  If nextRrup = 0 Then
                     nextRrup = 0.0000001
                  End If
                  hereRrup = Log(hereRrup)
                  nextRrup = Log(nextRrup)
                  thisRrup = Log(thisRrup)
               End If
               iRow = Ibelow
                  Y0 = MedianLnYrefDataTable(iRow, iColR)
                  Y1 = MedianLnYrefDataTable(iRow, iColR + 1)
                  Z0 = Y0 + (thisRrup - hereRrup) * (Y1 - Y0) / (nextRrup - hereRrup)
               iRow = Iabove
                  Y0 = MedianLnYrefDataTable(iRow, iColR)
                  Y1 = MedianLnYrefDataTable(iRow, iColR + 1)
                  Z1 = Y0 + (thisRrup - hereRrup) * (Y1 - Y0) / (nextRrup - hereRrup)
            End If
         End If

      End If
   Next iColR
   ' interpolate on Moment
      If LCase(Left(MagnitudeInterpolationType, 3)) = "log" Then
         Mbelow = Log(Mbelow)
         Mabove = Log(Mabove)
         thisM = Log(thisM)
      End If
      getLnY = Z0 + (thisM - Mbelow) * (Z1 - Z0) / (Mabove - Mbelow)



End Function




Function eliminateDuplicate(poArr As Variant) As Variant
   ' Code written and developed by Silvia Mazzoni, smazzoni@ucla.edu, March 2020
   Dim poArrNoDup()

   dupArrIndex = -1
```

```
    For i = LBound(poArr, 1) To UBound(poArr, 1)
        dupBool = False

        For j = LBound(poArr, 1) To i
            If poArr(i, 1) = poArr(j, 1) And Not i = j Then
                dupBool = True
            End If
        Next j

        If dupBool = False Then
            dupArrIndex = dupArrIndex + 1
            ReDim Preserve poArrNoDup(dupArrIndex)
            poArrNoDup(dupArrIndex) = poArr(i, 1)
        End If
    Next i

    eliminateDuplicate = poArrNoDup
End Function
Function getSigma(thisFreq, thisM, thisRrup, thisSigmaType, thisSigmaModelLabel)

    ' Code written and developed by Silvia Mazzoni, smazzoni@ucla.edu, March 2020

    If LCase(thisSigmaType) = LCase("Ergodic") Then
        sigmaModelLabel = "CompErg"
    Else
        sigmaModelLabel = "CompSS"
    End If

    getSigma = 0
    totalWeightSum = 0
    'Central Model
        If thisSigmaModelLabel = "WeightedAvg" Or thisSigmaModelLabel = "Central" Then
            RangeModelLabel = "C"
                thisSigmaI = getSigmaModel(thisFreq, thisM, thisSigmaType, RangeModelLabel)
                thisWeight = Range("Weight_Sigma" & sigmaModelLabel & "_" & RangeModelLabel)
                totalWeightSum = totalWeightSum + thisWeight
                getSigma = getSigma + thisSigmaI * thisWeight
        End If
    'High Model
        If thisSigmaModelLabel = "WeightedAvg" Or thisSigmaModelLabel = "High" Then
            RangeModelLabel = "H"
                thisSigmaI = getSigmaModel(thisFreq, thisM, thisSigmaType, RangeModelLabel)
                thisWeight = Range("Weight_Sigma" & sigmaModelLabel & "_" & RangeModelLabel)
                totalWeightSum = totalWeightSum + thisWeight
                getSigma = getSigma + thisSigmaI * thisWeight
        End If
    'Low Model
        If thisSigmaModelLabel = "WeightedAvg" Or thisSigmaModelLabel = "Low" Then
            RangeModelLabel = "L"
                thisSigmaI = getSigmaModel(thisFreq, thisM, thisSigmaType, RangeModelLabel)
                thisWeight = Range("Weight_Sigma" & sigmaModelLabel & "_" & RangeModelLabel)
                totalWeightSum = totalWeightSum + thisWeight
                getSigma = getSigma + thisSigmaI * thisWeight
        End If


    getSigma = getSigma / totalWeightSum
```

```
End Function
Function getSigmaModel(thisFreq, thisM, thisSigmaType, RangeModelLabel)
   ' Code written and developed by Silvia Mazzoni, smazzoni@ucla.edu, March 2020

   If LCase(thisSigmaType) = LCase("Ergodic") Then
      sigmaModelLabel = "CompErg"
   Else
      sigmaModelLabel = "CompSS"
   End If

   thisFreq = getTFvalue(thisFreq)
   FrequencyList = Range("FrequencyList" & sigmaModelLabel)
   thisRowIndex = -999
   For iRowF = LBound(FrequencyList, 1) To UBound(FrequencyList, 1)
      hereFreq = FrequencyList(iRowF, 1)
      If Abs(hereFreq - thisFreq) < 0.000001 Then
         thisRowIndex = iRowF
      End If
   Next iRowF

         Sig1 = Range("Sig" & sigmaModelLabel & "1_" & RangeModelLabel)
         Sig2 = Range("Sig" & sigmaModelLabel & "2_" & RangeModelLabel)
         Sig3 = Range("Sig" & sigmaModelLabel & "3_" & RangeModelLabel)
         Sig4 = Range("Sig" & sigmaModelLabel & "4_" & RangeModelLabel)
         s1 = Sig1(thisRowIndex, 1)
         s2 = Sig2(thisRowIndex, 1)
         s3 = Sig3(thisRowIndex, 1)
         s4 = Sig4(thisRowIndex, 1)

   If thisM <= 4.5 Then
      getSigmaModel = s1
   ElseIf thisM <= 5# Then
      getSigmaModel = s1 + (s2 - s1) * (thisM - 4.5) / 0.5
   ElseIf thisM <= 5.5 Then
      getSigmaModel = s2 + (s3 - s2) * (thisM - 5#) / 0.5
   ElseIf thisM <= 6.5 Then
      getSigmaModel = s3 + (s4 - s3) * (thisM - 5.5) / 1#
   Else
      getSigmaModel = s4
   End If


End Function
```

4.2 NGA-East Ground-Motion Characterization Tool – "R"

NGA-East_GMMtool_R_R200316.r

```
### ################################################################
### ################################################################
### ----------------------------------------------------------------
### ----------------------------------------------------------------
### -- NGA-East: Central and Eastern North America
###              Ground-Motion Characterization Tool
###              R Version
### ----------------------------------------------------------------
###                        March 2020
### ----------------------------------------------------------------
### Ground-Motion Model Tool developed by
###          Silvia Mazzoni
###             smazzoni@ucla.edu
###          B. John Garrick Institute for the Risk Sciences
###             https://www.risksciences.ucla.edu/nhr3/gmtools/home
### ----------------------------------------------------------------
### Source:
###      Report PEER 2018/08 - Central and Eastern North America
###      Ground-Motion Characterization - NGA-East Final Report
###         Christine Goulet, Yousef Bozorgnia, Norman Abrahamson,
###         Nicolas Kuehn, Linda Al Atik, Robert Youngs, and Robert Graves
###         apps.peer.berkeley.edu/publications/peer_reports/reports_2018/NGA-East_1MainReport_FINAL.pdf


### ----------------------------------------------------------------
### Instruction:
###   1. Run full script
###   2. Wait while script runs with the initial values (this may take 30+ seconds)
###   3. A plot of the response spectra will open (you can copy and paste it elsewhere)
###   4. Result values are printed in Command Window (you can copy and paste them elsewhere)
###   5. Scroll to the bottom of the script
###   6. Edit the values of the user-input parameters.
###      Model-Range limits are given with each input parameters.
###   7. Select User-input and Run lines
##    8. Run selected lines
### ----------------------------------------------------------------
###   Notes:
###      The Median and Sigma models are described in the report.
###      The values are computed for Vs30=3000 (m/s)
###      Damping Ratio = 5#
###      This script only outputs the Weighted-Average Composite values for Median and Sigma.
###      The values for the individual components can be found in the Excel version of the NGA-East GMtool.
### ----------------------------------------------------------------
### Variables:
###   User-Input Parameters:
###      inMagnitude:    Event Magnitude. Range=4-8.2
###      inRrup:         Rupture Distance (km). Range=1-1500km
###      inSigmaType:    Type of Sigma: Ergodic or Single-Station
###      inNsigma:       Number of Standard Deviations (sigma)
###      inPeriodList:   Array of periods (sec) for computation. Range: 0.01-10, 'PGA','PGV', -1 (=PGA), -2 (=PGV)
###   Output Variables:
###     outDataTable:               Table with all in/out data matrix
###     outTlist:                 Array of Periods used for computuation (sec) -1 (=PGA), -2 (=PGV)
###     outMedianList:            Array of median PSA (g);
```

```
###    outSigmaList:              Array of Sigma;
###    outMedianMinusNSigmaList:  Array of Median - inNsigma*Sigma (g);
###    outMedianPlusNSigmaList:         Array of Median + inNsigma*Sigma (g);
### ------------------------------------------------------------------
### ##################################################################
### ##################################################################

# ##################################################################
# ##################################################################
#            SCROLL TO THE BOTTOM OF THE SCRIPT!!!!!!!!!!!
#            SCROLL TO THE BOTTOM OF THE SCRIPT!!!!!!!!!!!
#            SCROLL TO THE BOTTOM OF THE SCRIPT!!!!!!!!!!!
#            SCROLL TO THE BOTTOM OF THE SCRIPT!!!!!!!!!!!
#            SCROLL TO THE BOTTOM OF THE SCRIPT!!!!!!!!!!!
#            SCROLL TO THE BOTTOM OF THE SCRIPT!!!!!!!!!!!
# ##################################################################
# ##################################################################


# ##################################################################
# the following commands compute the output values at each period and
# generate a figure of the data
# The user does not need to edit the following
# ##################################################################

# initialize
# Clear all data in memory
rm(list=ls(all=TRUE))

#########################################################
################ PROGRAM DATA ###########################
#########################################################
getTvalue <-function(thisTin){
              thisTin
   if(is.character(thisTin)==1){
      if(tolower(thisTin) == tolower("PGA")){
         thisT=-1
      } else if(tolower(thisTin) == tolower("PGV")){
         thisT=-2
      } else {

         thisT = as.numeric(thisTin);
      }
   } else {
      thisT = thisTin;
   }
}


getNGAEastGMM <-function(thisTin,inMagnitude,inRrup,thisInputLabel){



   iNGAEastGMM = -999;
   iLnY = -999;

   if(tolower(thisInputLabel) == tolower("ErgodicSigma")){
      Frange = getFrequencyListCompErg();
```

```
} else if(tolower(thisInputLabel) == tolower("SingleStationSigma")){
   Frange = getFrequencyListCompSS();
} else {
   Frange = getmedianLnYrefFrequencyValueColumn();
}

Flist = unique(Frange);

thisT = getTvalue(thisTin);

for (iiT in 1:length(Flist)) {
   hereF = Flist[iiT];
   if(hereF < 0) {
      hereT = hereF;
   } else {
      hereT = 1 / hereF;
   }
   if(abs(hereT - thisT) < 0.000001 ){
      if(tolower(thisInputLabel) == tolower("ErgodicSigma")){
         iLnY = getSigmaCompErg(hereF, inMagnitude, inRrup);
      } else if(tolower(thisInputLabel) == tolower("SingleStationSigma")){
         iLnY = getSigmaCompSS(hereF, inMagnitude, inRrup);
      } else {
         iLnY = getLnY(hereF, inMagnitude, inRrup);
      }
      iiT = length(Flist) + 1;
   } else {
      if(iiT < length(Flist) ){
         nextF = Flist[iiT + 1];
         nextT = 1 / nextF;
         if(hereT > 0 && nextT > 0 && (hereT - thisT) * (nextT - thisT) < 0 ){

            if(tolower(thisInputLabel) == tolower("ErgodicSigma")){
               hereLnY = getSigmaCompErg(hereF, inMagnitude, inRrup);
               nextLnY = getSigmaCompErg(nextF, inMagnitude, inRrup);
            } else if(tolower(thisInputLabel) == tolower("SingleStationSigma")){
               hereLnY = getSigmaCompSS(hereF, inMagnitude, inRrup);
               nextLnY = getSigmaCompSS(nextF, inMagnitude, inRrup);
            } else {
               hereLnY = getLnY(hereF, inMagnitude, inRrup);
               nextLnY = getLnY(nextF, inMagnitude, inRrup);
            }
            thisLnT = log(thisT);
            hereLnT = log(hereT);
            nextLnT = log(nextT);
            iLnY = hereLnY + (thisLnT - hereLnT) * (nextLnY - hereLnY) / (nextLnT - hereLnT);
            iiT = length(Flist) + 1;
         }
      }
   }
}


if(tolower(thisInputLabel) == tolower("MedianPSA") && iLnY >-100 ){
   iNGAEastGMM = exp(iLnY);
} else {
   iNGAEastGMM = iLnY;
}
```

```
}



getmedianLnYrefFrequencyValueColumn <- function(){
   medianLnYrefFrequencyValueColumn =
c(0.1,0.1,0.1,0.1,0.1,0.1,0.1,0.1,0.1,0.1,0.1,0.133,0.133,0.133,0.133,0.133,0.133,0.133,0.133,0.133,0.133,0.133,0.2,0.2,0.2,0.
2,0.2,0.2,0.2,0.2,0.2,0.2,0.2,0.2,0.25,0.25,0.25,0.25,0.25,0.25,0.25,0.25,0.25,0.25,0.25,0.333,0.333,0.333,0.333,0.333,0.333,0.33
3,0.333,0.333,0.333,0.333,0.5,0.5,0.5,0.5,0.5,0.5,0.5,0.5,0.5,0.5,0.5,0.667,0.667,0.667,0.667,0.667,0.667,0.667,0.667,0.667,
0.667,0.667,1,1,1,1,1,1,1,1,1,1,1,1.333,1.333,1.333,1.333,1.333,1.333,1.333,1.333,1.333,1.333,1.333,2,2,2,2,2,2,2,2,2,2,2,2.
5,2.5,2.5,2.5,2.5,2.5,2.5,2.5,2.5,2.5,2.5,2.5,3.333,3.333,3.333,3.333,3.333,3.333,3.333,3.333,3.333,3.333,3.333,3.333,4,4,4,4,4,4,4,4,
4,4,5,5,5,5,5,5,5,5,5,5,5,6.667,6.667,6.667,6.667,6.667,6.667,6.667,6.667,6.667,6.667,6.667,10,10,10,10,10,10,10,10,10,
10,13.333,13.333,13.333,13.333,13.333,13.333,13.333,13.333,13.333,13.333,13.333,20,20,20,20,20,20,20,20,20,20,20,25,2
5,25,25,25,25,25,25,25,25,25,33.333,33.333,33.333,33.333,33.333,33.333,33.333,33.333,33.333,33.333,33.333,33.333,40,40,40,40,
40,40,40,40,40,40,40,50,50,50,50,50,50,50,50,50,50,50,100,100,100,100,100,100,100,100,100,100,100,-1,-1,-1,-1,-1,-1,-1,-
1,-1,-1,-1,-2,-2,-2,-2,-2,-2,-2,-2,-2,-2);
}
getmedianLnYrefFrequencyLabelColumn <- function(){
   medianLnYrefFrequencyLabelColumn =
c("F0.100","F0.100","F0.100","F0.100","F0.100","F0.100","F0.100","F0.100","F0.100","F0.100","F0.133","F0.133","
F0.133","F0.133","F0.133","F0.133","F0.133","F0.133","F0.133","F0.133","F0.200","F0.200","F0.200","F0.200","F0.
200","F0.200","F0.200","F0.200","F0.200","F0.200","F0.200","F0.250","F0.250","F0.250","F0.250","F0.250","F0.250","F0.25
0","F0.250","F0.250","F0.250","F0.250","F0.333","F0.333","F0.333","F0.333","F0.333","F0.333","F0.333","F0.333",
"F0.333","F0.333","F0.500","F0.500","F0.500","F0.500","F0.500","F0.500","F0.500","F0.500","F0.500","F0.500","F0
.667","F0.667","F0.667","F0.667","F0.667","F0.667","F0.667","F0.667","F0.667","F0.667","F0.667","F1.000","F1.000","F1.00
0","F1.000","F1.000","F1.000","F1.000","F1.000","F1.000","F1.000","F1.000","F1.333","F1.333","F1.333","F1.333","F1.333",
"F1.333","F1.333","F1.333","F1.333","F1.333","F1.333","F2.000","F2.000","F2.000","F2.000","F2.000","F2.000","F2.000","F2
.000","F2.000","F2.000","F2.000","F2.500","F2.500","F2.500","F2.500","F2.500","F2.500","F2.500","F2.500","F2.500","F2.50
0","F2.500","F3.333","F3.333","F3.333","F3.333","F3.333","F3.333","F3.333","F3.333","F3.333","F3.333","F3.333","F4.000",
"F4.000","F4.000","F4.000","F4.000","F4.000","F4.000","F4.000","F4.000","F4.000","F4.000","F5.000","F5.000","F5.000","F5
.000","F5.000","F5.000","F5.000","F5.000","F5.000","F5.000","F5.000","F6.667","F6.667","F6.667","F6.667","F6.667","F6.66
7","F6.667","F6.667","F6.667","F6.667","F6.667","F10.000","F10.000","F10.000","F10.000","F10.000","F10.000","F10.000","
F10.000","F10.000","F10.000","F10.000","F13.333","F13.333","F13.333","F13.333","F13.333","F13.333","F13.333","F13.333
","F13.333","F13.333","F13.333","F20.000","F20.000","F20.000","F20.000","F20.000","F20.000","F20.000","F20.000","F20.0
00","F20.000","F20.000","F25.000","F25.000","F25.000","F25.000","F25.000","F25.000","F25.000","F25.000","F25.000","F2
5.000","F25.000","F33.333","F33.333","F33.333","F33.333","F33.333","F33.333","F33.333","F33.333","F33.333","F33.333","
F33.333","F40.000","F40.000","F40.000","F40.000","F40.000","F40.000","F40.000","F40.000","F40.000","F40.000","F40.000
","F50.000","F50.000","F50.000","F50.000","F50.000","F50.000","F50.000","F50.000","F50.000","F50.000","F50.000","F100.
000","F100.000","F100.000","F100.000","F100.000","F100.000","F100.000","F100.000","F100.000","F100.000","F100.000","
PGA","PGA","PGA","PGA","PGA","PGA","PGA","PGA","PGA","PGA","PGA","PGV","PGV","PGV","PGV","PGV","PGV","PGV","PGV","P
GV","PGV","PGV","PGV");
}
getmedianLnYrefMomentColumn <- function(){
   medianLnYrefMomentColumn =
c(4,4.5,5,5.5,6,6.5,7,7.5,7.8,8,8.2,4,4.5,5,5.5,6,6.5,7,7.5,7.8,8,8.2,4,4.5,5,5.5,6,6.5,7,7.5,7.8,8,8.2,4,4.5,5,5.5,6,6.5,7,7.5,7.8,8,
8.2,4,4.5,5,5.5,6,6.5,7,7.5,7.8,8,8.2,4,4.5,5,5.5,6,6.5,7,7.5,7.8,8,8.2,4,4.5,5,5.5,6,6.5,7,7.5,7.8,8,8.2,4,4.5,5,5.5,6,6.5,7,7.5,7.8
,8,8.2,4,4.5,5,5.5,6,6.5,7,7.5,7.8,8,8.2,4,4.5,5,5.5,6,6.5,7,7.5,7.8,8,8.2,4,4.5,5,5.5,6,6.5,7,7.5,7.8,8,8.2,4,4.5,5,5.5,6,6.5,7,7.5,7
.8,8,8.2,4,4.5,5,5.5,6,6.5,7,7.5,7.8,8,8.2,4,4.5,5,5.5,6,6.5,7,7.5,7.8,8,8.2,4,4.5,5,5.5,6,6.5,7,7.5,7.8,8,8.2,4,4.5,5,5.5,6,6.5,7,7.
5,7.8,8,8.2,4,4.5,5,5.5,6,6.5,7,7.5,7.8,8,8.2,4,4.5,5,5.5,6,6.5,7,7.5,7.8,8,8.2,4,4.5,5,5.5,6,6.5,7,7.5,7.8,8,8.2,4,4.5,5,5.5,6,6.5,7,
7.5,7.8,8,8.2,4,4.5,5,5.5,6,6.5,7,7.5,7.8,8,8.2,4,4.5,5,5.5,6,6.5,7,7.5,7.8,8,8.2,4,4.5,5,5.5,6,6.5,7,7.5,7.8,8,8.2,4,4.5,5,5.5,6,6.5
,7,7.5,7.8,8,8.2,4,4.5,5,5.5,6,6.5,7,7.5,7.8,8,8.2);
}
getMedianLnYrefRrupRow <- function(){
```

```
   MedianLnYrefRrupRow =
c(0,1,5,10,15,20,25,30,40,50,60,70,80,90,100,110,120,130,140,150,175,200,250,300,350,400,450,500,600,700,800,1000,12
00,1500);
}
getMedianLnYrefDataTable <- function(){
                        MedianLnYrefRrupRow =getMedianLnYrefRrupRow ()
                        Ncol = length(MedianLnYrefRrupRow)
     MedianLnYrefDataTableList <- matrix(c(-9.605511837,-9.856291163,-11.00960426,-11.82957032,-12.37243119,-
12.77947739,-13.11741114,-13.38980611,-13.81525848,-14.14286981,-14.33636957,-14.46077937,-14.54637483,-
14.61907323,-14.68526388,-14.74619915,-14.80314538,-14.86447387,-14.92591871,-14.98726788,-15.13383919,-
15.2648154,-15.50453776,-15.71111627,-15.89439328,-16.06048595,-16.21874402,-16.36542119,-16.61035701,-
16.85344312,-17.07062066,-17.4924918,-17.87393458,-18.43088267,
     -8.632895422,-8.808995903,-9.741673551,-10.51024366,-11.02877457,-11.41642824,-11.73855713,-11.99988697,-
12.40649941,-12.7205702,-12.90340172,-13.01997491,-13.09870673,-13.16525683,-13.22570974,-13.28127306,-
13.33313003,-13.38924104,-13.44548257,-13.50189503,-13.63699049,-13.75774001,-13.97748607,-14.16657328,-
14.33428781,-14.48627938,-14.63131164,-14.76570622,-14.99235452,-15.21270808,-15.41772823,-15.80572415,-
16.15812022,-16.67464842,
     -7.628126741,-7.754042931,-8.470501031,-9.164651811,-9.650659581,-10.01663929,-10.31773552,-10.56548549,-
10.95379646,-11.2559549,-11.42998075,-11.54043378,-11.61361699,-11.67497706,-11.73064368,-11.78171109,-
11.82929366,-11.88086304,-11.93247565,-11.9844441,-12.10926094,-12.22088544,-12.42343116,-12.59744808,-
12.75244678,-12.89328261,-13.0279918,-13.15280403,-13.3613213,-13.56784311,-13.75631479,-14.1173078,-
14.44862775,-14.92937735,
     -6.634394627,-6.728558551,-7.239586836,-7.829807262,-8.271634746,-8.610992066,-8.890940953,-9.125179644,-
9.492758134,-9.779968814,-9.944571959,-10.04860333,-10.11610877,-10.17203308,-10.2227232,-10.26978155,-
10.31357445,-10.36131443,-10.40946022,-10.45818147,-10.57537956,-10.68012642,-10.86967543,-11.03221346,-
11.17567156,-11.30544243,-11.42885006,-11.54340737,-11.73694501,-11.9302876,-12.10562903,-12.44221981,-
12.74703027,-13.19477817,
     -5.685500404,-5.760246657,-6.120002438,-6.583382757,-6.96588858,-7.273349299,-7.529593949,-7.74824096,-
8.098394827,-8.374097089,-8.531679301,-8.630827331,-8.693808479,-8.745497831,-8.792300307,-8.836352187,-
8.877562813,-8.922175588,-8.967356245,-9.013205604,-9.124163229,-9.223936318,-9.402643941,-9.556046412,-
9.692161801,-9.815395627,-9.932163763,-10.04030563,-10.22386691,-10.4048243,-10.56802154,-10.88032707,-
11.16269267,-11.57727908,
     -4.800405449,-4.86070112,-5.123329674,-5.46830614,-5.785412651,-6.057208318,-6.292867269,-6.497538491,-
6.827830152,-7.089328065,-7.2399778,-7.334130278,-7.393014475,-7.441628015,-7.485504575,-7.52728048,-
7.566028691,-7.608293992,-7.651076235,-7.694656372,-7.800700833,-7.896617464,-8.068964995,-8.217012507,-
8.347518764,-8.465041457,-8.575885419,-8.6782273,-8.852031579,-9.020443496,-9.171549066,-9.461660097,-
9.723000624,-10.0981226,
     -3.988380738,-4.039620817,-4.246495375,-4.517990876,-4.777784017,-5.008947136,-5.212912818,-5.393401335,-
5.693280571,-5.929968766,-6.069708187,-6.158511616,-6.214828093,-6.261609781,-6.30410199,-6.34540248,-
6.383721986,-6.425122519,-6.466745331,-6.509138548,-6.611726057,-6.704344454,-6.871637014,-7.01510075,-
7.141229057,-7.254704516,-7.361242227,-7.459421836,-7.625349018,-7.784705046,-7.92793548,-8.201553061,-
8.444516759,-8.792133926,
     -3.295383805,-3.337669033,-3.506119623,-3.726210342,-3.935592116,-4.126148476,-4.299410316,-4.455342252,-
4.720158297,-4.926007881,-5.053726401,-5.134472259,-5.187351742,-5.231351473,-5.271402346,-5.311253123,-
5.348243333,-5.388766359,-5.429898758,-5.471648582,-5.572246295,-5.662870458,-5.822877092,-5.960119597,-
6.081574046,-6.191008105,-6.293356206,-6.387258822,-6.54795043,-6.70140648,-6.838945787,-7.098703411,-
7.32756554,-7.652964321,
     -2.947955959,-2.986949286,-3.139420371,-3.331206605,-3.515126577,-3.684383443,-3.840019759,-3.98127727,-
4.227794294,-4.414797434,-4.53588013,-4.611174698,-4.66171024,-4.704343183,-4.74325841,-4.782554002,-
4.819053724,-4.858942731,-4.899254131,-4.940125071,-5.038773023,-5.127347665,-5.284798174,-5.41992718,-
5.539656689,-5.647301307,-5.747702709,-5.83975969,-5.996632983,-6.145850152,-6.278515354,-6.529527144,-
6.752811794,-7.065091549,
     -2.742464853,-2.781731342,-2.924817321,-3.097354212,-3.2639971,-3.420054732,-3.564478853,-3.696092182,-
3.929902957,-4.102666099,-4.219155505,-4.290365389,-4.339194694,-4.380853712,-4.418964151,-4.457869461,-
4.494028897,-4.533488089,-4.573261644,-4.613498275,-4.710824899,-4.797997005,-4.953807389,-5.087555659,-
5.205729344,-5.312105359,-5.411298232,-5.502090051,-5.656076778,-5.802382996,-5.933492832,-6.179872946,-
6.396066756,-6.701456202,
```

-2.548821372,-2.591209906,-2.727436134,-2.881958851,-3.029866353,-3.172836014,-3.306113162,-3.428180029,-3.647927895,-3.80767519,-3.919688701,-3.986740634,-4.03383475,-4.074563771,-4.11180852,-4.150315822,-4.186071696,-4.225115013,-4.264337111,-4.303910382,-4.399864408,-4.485534807,-4.639545622,-4.771477809,-4.888165149,-4.993171014,-5.090848231,-5.180131736,-5.331702141,-5.473790551,-5.601495481,-5.842794097,-6.052468481,-6.34469623,

-8.897621841,-9.14758561,-10.31341195,-11.14141252,-11.68873225,-12.09876227,-12.43951131,-12.7140415,-13.14271447,-13.47283683,-13.6671022,-13.79315589,-13.87955592,-13.95271894,-14.01938979,-14.0808104,-14.13822585,-14.19983327,-14.26165159,-14.3235169,-14.47124447,-14.60326967,-14.84525722,-15.05358232,-15.23811889,-15.40503785,-15.56438964,-15.71160556,-15.95631049,-16.19751958,-16.411736,-16.82269953,-17.18915805,-17.71674244,

-7.931251855,-8.105834198,-9.046826291,-9.822915162,-10.34593868,-10.73675096,-11.06185351,-11.32553022,-11.73571261,-12.05268098,-12.23665483,-12.3551553,-12.43496345,-12.50225779,-12.56346343,-12.6197804,-12.67237152,-12.7290385,-12.78591865,-12.8431071,-12.97999621,-13.10242096,-13.3255822,-13.51753607,-13.68760995,-13.84151461,-13.9885201,-14.12436904,-14.35261216,-14.57325016,-14.77716015,-15.15849653,-15.50059273,-15.99514499,

-6.944350603,-7.068535156,-7.789627709,-8.490417639,-8.980838461,-9.350049699,-9.654042698,-9.904209963,-10.29636793,-10.60177304,-10.77727133,-10.8899062,-10.96441632,-11.02676229,-11.08341678,-11.13546767,-11.18401263,-11.23637616,-11.28885304,-11.34181614,-11.4689678,-11.5827964,-11.78972697,-11.96755455,-12.12586445,-12.2695625,-12.40691639,-12.53390437,-12.74558666,-12.95373429,-13.14288241,-13.50093605,-13.82576778,-14.29119922,

-5.983421346,-6.075679227,-6.587392012,-7.182468308,-7.628319359,-7.970894525,-8.253751454,-8.490541134,-8.862221481,-9.152958202,-9.319360493,-9.42582882,-9.494904612,-9.551990505,-9.603840687,-9.652019383,-9.696913839,-9.745606034,-9.794774646,-9.844642328,-9.964566891,-10.07194076,-10.2666683,-10.43369761,-10.58113158,-10.71443663,-10.8409662,-10.95824215,-11.1559747,-11.35226726,-11.5295351,-11.86612685,-12.16772075,-12.60627862,

-5.078344192,-5.151278175,-5.509402548,-5.975338635,-6.361127339,-6.671582631,-6.930597403,-7.151795892,-7.505988128,-7.785201741,-7.944650211,-8.046283077,-8.110888271,-8.163848363,-8.211917704,-8.257157195,-8.299365401,-8.345050668,-8.391392579,-8.438527153,-8.552522572,-8.655192879,-8.839533068,-8.997714034,-9.138167803,-9.265335715,-9.385634475,-9.496921106,-9.685666725,-9.870697345,-10.03720149,-10.35190099,-10.63387957,-11.04479903,

-4.244884535,-4.303478701,-4.563644731,-4.909291335,-5.22836513,-5.502203125,-5.740110772,-5.946994502,-6.281142556,-6.546092905,-6.698695874,-6.795371153,-6.855922859,-6.905558615,-6.950675218,-6.993752704,-7.033778921,-7.077195607,-7.121190177,-7.166106479,-7.275306501,-7.374240113,-7.552459929,-7.705393967,-7.840368088,-7.961979953,-8.076553162,-8.182283514,-8.361739784,-8.534992043,-8.690126642,-8.985078531,-9.248944501,-9.625232509,

-3.490658881,-3.540401805,-3.744679555,-4.01422142,-4.2744288,-4.50687013,-4.712509305,-4.894822076,-5.198201849,-5.438021075,-5.579711758,-5.670430773,-5.728339166,-5.776610434,-5.820560252,-5.863151109,-5.902737877,-5.945287767,-5.988112191,-6.031824866,-6.137524065,-6.233125915,-6.406262436,-6.554516073,-6.685060327,-6.802619761,-6.912936579,-7.014588043,-7.186405378,-7.351057654,-7.499059938,-7.779854819,-8.028001529,-8.381508605,

-2.860641988,-2.90076527,-3.065925102,-3.281585053,-3.489973512,-3.681041996,-3.855430168,-4.012773625,-4.280577851,-4.488590761,-4.617732081,-4.700960298,-4.755563256,-4.800976193,-4.84240872,-4.88343834,-4.921591091,-4.963198511,-5.00548302,-5.048494295,-5.152064971,-5.245543651,-5.411057034,-5.552774768,-5.678415328,-5.79176661,-5.897788982,-5.995242534,-6.162518518,-6.32200485,-6.465117235,-6.733601748,-6.969878205,-7.305617071,

-2.549595157,-2.586627207,-2.736048269,-2.924535974,-3.105150898,-3.273735943,-3.430169585,-3.572550911,-3.821551366,-4.009903409,-4.133048657,-4.210667313,-4.262838589,-4.306835571,-4.347081692,-4.387484332,-4.425077265,-4.465989157,-4.507390397,-4.549452364,-4.650910499,-4.742185213,-4.904907649,-5.044317291,-5.168079252,-5.279821001,-5.384154737,-5.479884154,-5.643370049,-5.79875414,-5.937180518,-6.197762557,-6.429817781,-6.754961394,

-2.367657402,-2.40521638,-2.545490625,-2.714859883,-2.87881912,-3.032778664,-3.176921283,-3.309459395,-3.545035525,-3.719374887,-3.837865557,-3.911273342,-3.961667341,-4.004651711,-4.044054152,-4.084012295,-4.121213645,-4.161651434,-4.202466059,-4.243842149,-4.343859824,-4.433616507,-4.594525127,-4.732413532,-4.854960842,-4.965442685,-5.068503607,-5.162927747,-5.323550159,-5.476095344,-5.613153624,-5.869786644,-6.095480175,-6.41559298,

-2.1967275,-2.237733268,-2.371370213,-2.523673169,-2.668295448,-2.809283173,-2.941396096,-3.062736819,-3.283592369,-3.445058337,-3.559004897,-3.628124832,-3.67671161,-3.718726005,-3.757222715,-3.796729017,-3.833470795,-3.87344503,-3.91365913,-3.954319112,-4.052838737,-4.140974352,-4.29989845,-4.436381475,-

4.557322003,-4.666342033,-4.767818005,-4.860685521,-5.01895304,-5.167338941,-5.301180856,-5.553349778,-5.773374172,-6.081966288,
-8.000428233,-8.254022172,-9.38231526,-10.18547739,-10.72014458,-11.12237695,-11.45575189,-11.72493489,-12.14608114,-12.47083991,-12.66561093,-12.7908555,-12.87710808,-12.9511945,-13.01863063,-13.0806837,-13.13865176,-13.20110015,-13.26352275,-13.32570766,-13.47411534,-13.60643216,-13.8469096,-14.05313183,-14.23512572,-14.39910192,-14.55443399,-14.69770445,-14.93629989,-15.16925903,-15.37509457,-15.77202969,-16.12277894,-16.61996932,
-7.009179878,-7.190128358,-8.107662033,-8.861953409,-9.373206038,-9.756378557,-10.07435467,-10.33275945,-10.73577201,-11.04759415,-11.23232661,-11.35047342,-11.43062212,-11.49929838,-11.56173133,-11.61913918,-11.67274526,-11.73066915,-11.7885862,-11.84652045,-11.98515038,-12.10891284,-12.33271759,-12.52460734,-12.69409376,-12.84696078,-12.99205672,-13.12594759,-13.35161846,-13.56748638,-13.76601409,-14.13944498,-14.47163168,-14.94485269,
-6.036491777,-6.167803138,-6.877672931,-7.560409911,-8.040071986,-8.402054719,-8.699804266,-8.945121251,-9.330511834,-9.630895946,-9.807322191,-9.919925797,-9.995127427,-10.05922203,-10.11747459,-10.17099235,-10.22092233,-10.2748798,-10.32875725,-10.38283805,-10.51264096,-10.62871416,-10.83812857,-11.01763017,-11.17696478,-11.32115146,-11.45844038,-11.58517282,-11.79741906,-12.00396775,-12.19065124,-12.54585718,-12.86544271,-13.31767881,
-5.114334845,-5.214077576,-5.727437842,-6.310915635,-6.747865433,-7.083779236,-7.360894311,-7.592932485,-7.958067556,-8.243895568,-8.411146297,-8.517632032,-8.58751325,-8.646615306,-8.700340084,-8.750360166,-8.796998692,-8.847584356,-8.898442491,-8.949720262,-9.073004634,-9.183259578,-9.381757608,-9.551919607,-9.701871599,-9.837144013,-9.96513863,-10.08358068,-10.28460584,-10.48150477,-10.65893362,-10.99705234,-11.29877829,-11.73100132,
-4.275999162,-4.355744198,-4.722811908,-5.184630697,-5.564153986,-5.869043648,-6.123191803,-6.340069163,-6.688289907,-6.963003756,-7.123445905,-7.225255955,-7.290861215,-7.345932105,-7.395979178,-7.443280116,-7.487719434,-7.535413072,-7.58353123,-7.632165085,-7.749817671,-7.855760401,-8.044963428,-8.207570625,-8.351646438,-8.481799572,-8.604607071,-8.718092189,-8.91171254,-9.099128305,-9.266937845,-9.586924823,-9.872824944,-10.28274254,
-3.53209544,-3.596805693,-3.868552942,-4.21598426,-4.532533459,-4.803187159,-5.037434784,-5.240767721,-5.569576323,-5.830416757,-5.983865609,-6.080626151,-6.142143229,-6.194264919,-6.24142729,-6.286482045,-6.328401007,-6.373785292,-6.419580944,-6.466017121,-6.578980333,-6.681321108,-6.864751994,-7.02271219,-7.161954985,-7.287221554,-7.404948564,-7.513535113,-7.699140281,-7.87600385,-8.034074579,-8.336367624,-8.606490289,-8.987340396,
-2.87835895,-2.933484945,-3.149853259,-3.426902469,-3.688498961,-3.919912863,-4.123469947,-4.303261283,-4.60203363,-4.838475656,-4.980806771,-5.072247143,-5.131056001,-5.180908135,-5.226374951,-5.270810488,-5.31216861,-5.356546726,-5.401075006,-5.446230013,-5.55553629,-5.654428608,-5.832549652,-5.9858029,-6.120662959,-6.241972805,-6.355582719,-6.460238236,-6.638674232,-6.807511657,-6.958836425,-7.247882553,-7.504046236,-7.866156562,
-2.34713038,-2.393304265,-2.571337583,-2.799133644,-3.012140429,-3.204230031,-3.377885859,-3.533674304,-3.797897739,-4.004403519,-4.134605406,-4.217386311,-4.272450741,-4.319249591,-4.36201977,-4.404814205,-4.44465096,-4.487889807,-4.531654608,-4.575906107,-4.682614397,-4.778978836,-4.949174831,-5.095696882,-5.225438945,-5.342358937,-5.451534501,-5.551733357,-5.724494397,-5.887480789,-6.033666669,-6.311053266,-6.556608882,-6.904217259,
-2.095892595,-2.138486208,-2.300098697,-2.499254343,-2.688152984,-2.859870439,-3.016405954,-3.15793844,-3.404214723,-3.592654851,-3.715756455,-3.792925303,-3.84552223,-3.890733301,-3.932151043,-3.974214432,-4.013387933,-4.055812984,-4.098594,-4.141813657,-4.246214851,-4.340165627,-4.507076013,-4.650870092,-4.77836114,-4.892950254,-4.999723504,-5.097770911,-5.266367779,-5.425136163,-5.566740678,-5.836562027,-6.078185756,-6.417149905,
-1.952569688,-1.994973541,-2.146706817,-2.326599289,-2.498275095,-2.657621053,-2.80360153,-2.935767999,-3.169754522,-3.344240653,-3.462640579,-3.5356056,-3.586416305,-3.630495733,-3.67095119,-3.712494964,-3.751199581,-3.793064968,-3.835188363,-3.877662626,-3.980470925,-4.07275487,-4.237486457,-4.37943769,-4.504828201,-4.617804663,-4.723124005,-4.819705992,-4.985170211,-5.140991434,-5.280736061,-5.546879693,-5.782791171,-6.117705512,
-1.819253853,-1.864098618,-2.008389773,-2.169871602,-2.323525642,-2.469989202,-2.605379088,-2.728869059,-2.949149936,-3.11097025,-3.224771989,-3.293452274,-3.342438979,-3.385416186,-3.42484322,-3.465853002,-3.50401999,-3.545331593,-3.586781462,-3.628474765,-3.729626245,-3.820140997,-3.982520872,-4.12199512,-4.245506172,-4.356828516,-4.460393681,-4.55527904,-4.717954309,-4.869683745,-5.006282953,-5.267992586,-5.498924108,-5.824367739,

-7.483444855,-7.736120033,-8.856411112,-9.657132495,-10.19158371,-10.59404486,-10.92750432,-11.19694377,-11.61877811,-11.94427141,-12.14046654,-12.26572315,-12.35221457,-12.42666519,-12.49445172,-12.55683726,-12.61512427,-12.67812627,-12.74109489,-12.80369445,-12.9530696,-13.08616952,-13.32759251,-13.53440997,-13.71671417,-13.88075357,-14.0351869,-14.1775882,-14.41531004,-14.64632696,-14.85036457,-15.24180365,-15.58635329,-16.07227505,

-6.485008511,-6.665764921,-7.578271672,-8.33057812,-8.841600875,-9.224965115,-9.54303616,-9.801701616,-10.20548637,-10.51813329,-10.70439758,-10.82275789,-10.90335157,-10.97259611,-11.03558212,-11.09352461,-11.14765207,-11.20632035,-11.26497755,-11.3235181,-11.46359274,-11.58861683,-11.81429418,-12.00767124,-12.17832921,-12.33209934,-12.47705016,-12.6108259,-12.83697584,-13.05248705,-13.25036775,-13.6212762,-13.95007544,-14.41665262,

-5.521745314,-5.653163738,-6.360389604,-7.041695546,-7.521177294,-7.883356539,-8.18130123,-8.426921677,-8.813132934,-9.114375909,-9.292388895,-9.405350742,-9.481148589,-9.545965351,-9.604926728,-9.659135162,-9.709742503,-9.764593344,-9.819366801,-9.874210173,-10.00584721,-10.12357763,-10.33561664,-10.51732834,-10.67853069,-10.82429859,-10.96204305,-11.08925106,-11.30316965,-11.51046165,-11.69772395,-12.05278446,-12.37128414,-12.82082357,

-4.623399694,-4.723360302,-5.236404913,-5.819387358,-6.256336656,-6.592464921,-6.869797947,-7.102112365,-7.468051658,-7.754733146,-7.923552308,-8.030467303,-8.10101256,-8.160727759,-8.215077768,-8.265736265,-8.313015173,-8.364459181,-8.416188632,-8.468182441,-8.593312742,-8.705428723,-8.907063368,-9.079465997,-9.231436664,-9.368518512,-9.498314732,-9.618507524,-9.822673586,-10.02164444,-10.20066889,-10.54063513,-10.84361507,-11.27663829,

-3.822303535,-3.902212905,-4.270464481,-4.732832064,-5.112685539,-5.417893162,-5.67234641,-5.889525431,-6.23814369,-6.513326166,-6.674987301,-6.776998131,-6.843102828,-6.89884488,-6.949574022,-6.997584149,-7.042474671,-7.09113728,-7.140261028,-7.189771813,-7.309574495,-7.41752196,-7.610246305,-7.775550746,-7.922070565,-8.054441329,-8.179304149,-8.294705705,-8.492472187,-8.68289989,-8.853770082,-9.177408042,-9.466423893,-9.880005299,

-3.125341102,-3.190225451,-3.463874424,-3.812713177,-4.12979201,-4.400544505,-4.634810584,-4.838172395,-5.167300337,-5.428604236,-5.583171646,-5.680106377,-5.742112562,-5.794653756,-5.842428815,-5.888277015,-5.930984562,-5.97735558,-6.024145795,-6.071450957,-6.186558848,-6.290918183,-6.477966725,-6.638826592,-6.780727926,-6.908439727,-7.028453564,-7.139211584,-7.32939647,-7.509867857,-7.67144039,-7.979250902,-8.254308678,-8.641789083,

-2.523974133,-2.57925535,-2.797647338,-3.075869195,-3.33811763,-3.569841562,-3.77354373,-3.953408497,-4.252389361,-4.489214765,-4.632533144,-4.723799424,-4.782994006,-4.833494994,-4.879622593,-4.924765644,-4.96683246,-5.012119818,-5.05757037,-5.103529399,-5.214834822,-5.315627021,-5.497150933,-5.653213716,-5.790677058,-5.914405771,-6.030308946,-6.137153301,-6.3202768,-6.492934869,-6.648072391,-6.943733888,-7.206271703,-7.577107298,

-2.045865442,-2.091974631,-2.271703626,-2.500526907,-2.714350894,-2.906955019,-3.080840333,-3.236731115,-3.501094632,-3.707835667,-3.838703916,-3.921590148,-3.977035417,-4.024360131,-4.06767379,-4.111073549,-4.15151936,-4.195556632,-4.240122529,-4.28506941,-4.393519088,-4.491555256,-4.664818625,-4.813934984,-4.946086594,-5.065267371,-5.17660704,-5.278948646,-5.4563864,-5.623327592,-5.773462647,-6.057890316,-6.310599245,-6.668949337,

-1.825145877,-1.867715421,-2.030970133,-2.231706662,-2.421031678,-2.59313698,-2.749974406,-2.891662347,-3.137988533,-3.326633772,-3.45055215,-3.527785945,-3.580701692,-3.626361188,-3.668247194,-3.71084846,-3.750566314,-3.793725472,-3.837242544,-3.881099964,-3.987103389,-4.082584969,-4.252284515,-4.398473225,-4.528201114,-4.644999363,-4.753928046,-4.854073741,-5.027166322,-5.189850755,-5.335471191,-5.612630789,-5.861645136,-6.212440042,

-1.701206163,-1.743538618,-1.896821042,-2.078396272,-2.250899979,-2.410433502,-2.55655042,-2.688912908,-2.922823405,-3.097756969,-3.2169065,-3.289912362,-3.341002025,-3.385475702,-3.426347372,-3.468382757,-3.507585797,-3.550141089,-3.592954728,-3.636027225,-3.740339853,-3.834055712,-4.001383307,-4.14558458,-4.273215438,-4.388315392,-4.49569223,-4.59429462,-4.764150071,-4.923844926,-5.067483609,-5.341039442,-5.584787314,-5.932028047,

-1.586386106,-1.631056157,-1.776787832,-1.940204197,-2.094833277,-2.241698729,-2.377161224,-2.500571819,-2.720671071,-2.883170918,-2.997656325,-3.066352696,-3.115577436,-3.158893185,-3.198683859,-3.24013748,-3.278758381,-3.320714183,-3.362808487,-3.405060861,-3.507618538,-3.59946947,-3.764252562,-3.905968724,-4.031587769,-4.144924394,-4.25045366,-4.347290899,-4.514197852,-4.66982571,-4.810341856,-5.079588461,-5.318753399,-5.657749222,

-6.804933471,-7.057199031,-8.230337515,-9.059175986,-9.606385763,-10.01644293,-10.3574822,-10.63234635,-11.06177429,-11.39285578,-11.58768681,-11.71752982,-11.80741501,-11.8837251,-11.95334689,-12.01753703,-12.07755265,-12.14102296,-12.20447371,-12.2680492,-12.41972843,-12.55545386,-12.80424606,-13.01791363,-

13.20663615,-13.37674672,-13.54150884,-13.69273152,-13.94465927,-14.18732892,-14.401459,-14.80717421,-15.1628702,-15.65970733,
-5.808277059,-5.98502469,-6.931591413,-7.70915796,-8.233174507,-8.624953629,-8.951227483,-9.216050905,-9.628467364,-9.947770298,-10.13361597,-10.25702847,-10.34141722,-10.41293292,-10.47813345,-10.53824569,-10.59444709,-10.65399454,-10.71349111,-10.77335325,-10.91654879,-11.04492754,-11.27930833,-11.48085621,-11.65917421,-11.82020362,-11.9766203,-12.12034383,-12.36273774,-12.59257836,-12.80241936,-13.19203264,-13.53651179,-14.02041001,
-4.857568512,-4.98415465,-5.709615178,-6.412249191,-6.904505476,-7.275474447,-7.581349723,-7.833340308,-8.229058232,-8.538010511,-8.716526804,-8.835016444,-8.915032297,-8.982495149,-9.044015274,-9.100716667,-9.153708608,-9.209797905,-9.265722523,-9.322173066,-9.457627055,-9.579312303,-9.801204412,-9.99221561,-10.16221536,-10.31635865,-10.46652199,-10.60462095,-10.83677521,-11.05953773,-11.26077619,-11.63845827,-11.97624821,-12.4482096,
-3.995665957,-4.090582392,-4.605380029,-5.202116735,-5.650307334,-5.995193602,-6.280543351,-6.519718007,-6.895973653,-7.191191072,-7.36151743,-7.47454005,-7.549859046,-7.613473059,-7.671505454,-7.725587203,-7.776112682,-7.829675916,-7.883375524,-7.937883056,-8.068535115,-8.185459545,-8.398091018,-8.582927277,-8.746228935,-8.893857409,-9.032730556,-9.160882885,-9.382088669,-9.596054223,-9.790206139,-10.15550765,-10.48014335,-10.9404269,
-3.240369634,-3.316049011,-3.676488685,-4.144339945,-4.532549882,-4.845461523,-5.107158582,-5.330956597,-5.691429919,-5.976741101,-6.141614554,-6.250965726,-6.322838078,-6.382589176,-6.437114414,-6.48855551,-6.537601281,-6.588307826,-6.639241079,-6.690984713,-6.816020613,-6.929125856,-7.132951198,-7.310188418,-7.467839798,-7.610713639,-7.745368602,-7.869906374,-8.083854963,-8.289430965,-8.473833687,-8.824253459,-9.136931891,-9.583006398,
-2.594598692,-2.655657237,-2.917524432,-3.265276016,-3.587352433,-3.864818757,-4.106321379,-4.316606509,-4.657129255,-4.92805346,-5.086157948,-5.190470429,-5.25823519,-5.315436788,-5.367205989,-5.416227943,-5.461970631,-5.510408611,-5.559129684,-5.608780392,-5.72936148,-5.83905544,-6.037248849,-6.209462385,-6.362000831,-6.49979547,-6.629332658,-6.749060886,-6.955081597,-7.151022388,-7.327176641,-7.661419983,-7.961110547,-8.385395358,
-2.048624951,-2.100602946,-2.306463624,-2.579722322,-2.843320995,-3.079137761,-3.288249552,-3.473964631,-3.783776626,-4.029607858,-4.17666386,-4.275617516,-4.340545823,-4.395096687,-4.445046624,-4.493457927,-4.538642601,-4.58610567,-4.63356574,-4.68192223,-4.798782118,-4.904909141,-5.097654065,-5.264652308,-5.41241018,-5.545960137,-5.671231783,-5.786948937,-5.985691725,-6.173986225,-6.344011295,-6.667036194,-6.955464777,-7.367059125,
-1.630654237,-1.673354104,-1.840713797,-2.060402936,-2.271960079,-2.466032466,-2.64360022,-2.804172068,-3.078225077,-3.292459074,-3.427132053,-3.516976957,-3.577831188,-3.62914837,-3.676226519,-3.722745644,-3.766175648,-3.812395209,-3.859049405,-3.90644299,-4.020528074,-4.123894893,-4.307934644,-4.467272038,-4.609191442,-4.737779158,-4.858182366,-4.96915508,-5.162392507,-5.345256905,-5.510972801,-5.824724461,-6.10579594,-6.508857808,
-1.442612547,-1.481870783,-1.633172734,-1.823788961,-2.008566576,-2.180477532,-2.33982044,-2.48517792,-2.740269327,-2.93469301,-3.062340998,-3.146212811,-3.204281341,-3.253853587,-3.299430431,-3.345019359,-3.38759992,-3.432839415,-3.47835485,-3.524541574,-3.635909226,-3.736568025,-3.916954547,-4.073117445,-4.212431813,-4.338539256,-4.45645501,-4.565236744,-4.754261677,-4.933025716,-5.094424867,-5.401895291,-5.680190158,-6.077131603,
-1.339071183,-1.378717614,-1.520759789,-1.691833135,-1.858759285,-2.01653331,-2.163998877,-2.29932694,-2.541080518,-2.720674206,-2.843477322,-2.922876796,-2.978917508,-3.027244051,-3.071754295,-3.116684393,-3.158666584,-3.203230448,-3.247974192,-3.293293041,-3.402787896,-3.501563949,-3.679511279,-3.833540582,-3.970787897,-4.095192845,-4.211540149,-4.31880054,-4.504722325,-4.68060072,-4.84061908,-5.144862446,-5.418360469,-5.813285581,
-1.243247002,-1.286220652,-1.421430505,-1.574506199,-1.721729453,-1.865867299,-2.001399337,-2.126210986,-2.353219038,-2.519216292,-2.637273535,-2.712115645,-2.766091964,-2.813213025,-2.856592903,-2.900853368,-2.942163923,-2.986059882,-3.030016885,-3.074433824,-3.18199157,-3.278775312,-3.454103326,-3.605777069,-3.740960683,-3.863574366,-3.978072007,-4.083577381,-4.266806163,-4.438784459,-4.596007341,-4.896466666,-5.166312277,-5.553882266,
-5.891045074,-6.141113033,-7.286879973,-8.105952327,-8.651362887,-9.061293275,-9.401695159,-9.67654538,-10.10665905,-10.43871796,-10.63716055,-10.76611577,-10.85540108,-10.93169193,-11.00133224,-11.0655561,-11.125626,-11.19001467,-11.25440639,-11.31866779,-11.47201708,-11.60904578,-11.85896765,-12.07343387,-12.26272849,-12.43323004,-12.59519243,-12.74439123,-12.99763931,-13.23913925,-13.45446376,-13.86051229,-14.2194283,-14.72399225,

-4.86751949,-5.044280053,-5.973706627,-6.743059218,-7.265062983,-7.656352733,-7.981817678,-8.24643088,-8.659413784,-8.9796052,-9.169043447,-9.291882527,-9.376017605,-9.447857888,-9.513420329,-9.573913571,-9.630525995,-9.691306759,-9.752074514,-9.812946361,-9.958633362,-10.08916596,-10.32631476,-10.53022936,-10.71063022,-10.87353218,-11.02828498,-11.17114984,-11.41659146,-11.648147,-11.8602226,-12.25485309,-12.6068325,-13.1053422,

-3.935495458,-4.062692212,-4.779009729,-5.475277304,-5.965534985,-6.335853034,-6.641103347,-6.892869627,-7.288987236,-7.598608825,-7.780573627,-7.898663737,-7.978642372,-8.04665886,-8.10877942,-8.166106382,-8.219756056,-8.277299993,-8.334742311,-8.392454645,-8.53103888,-8.655551484,-8.881457476,-9.076015701,-9.249215986,-9.406309535,-9.555551031,-9.693523523,-9.930293502,-10.15578404,-10.36108535,-10.7468837,-11.09457175,-11.58629762,

-3.126208015,-3.221802655,-3.735286465,-4.328915886,-4.775622081,-5.119774125,-5.40444312,-5.643153469,-6.019480684,-6.315077737,-6.488503091,-6.601090498,-6.676343612,-6.739737251,-6.797705754,-6.851882369,-6.902600349,-6.957166748,-7.011975699,-7.067246247,-7.200318251,-7.320165979,-7.537468293,-7.724516995,-7.890193476,-8.040281499,-8.182378007,-8.314275821,-8.543185282,-8.762620253,-8.962051878,-9.337751812,-9.675502793,-10.15968591,

-2.452950845,-2.528860512,-2.892925098,-3.36130329,-3.749119434,-4.061585954,-4.322838544,-4.54619034,-4.904875193,-5.188794712,-5.355150883,-5.462904213,-5.533805646,-5.593357148,-5.64783783,-5.699452268,-5.747670741,-5.799636638,-5.852042626,-5.905021323,-6.033236835,-6.149351605,-6.358383232,-6.538630446,-6.69933616,-6.845282545,-6.983135722,-7.111006112,-7.334582678,-7.54727918,-7.740660793,-8.103836442,-8.431746832,-8.90422019,

-1.90648546,-1.967660781,-2.235485521,-2.586795256,-2.908995294,-3.184948366,-3.42475028,-3.63342368,-3.971799226,-4.241283916,-4.40046414,-4.503009653,-4.56970006,-4.625551263,-4.676908024,-4.726403704,-4.772686458,-4.822394539,-4.872479737,-4.923266042,-5.04681752,-5.159369137,-5.36268128,-5.538194278,-5.694089411,-5.835281557,-5.968321358,-6.091740575,-6.307658041,-6.51158345,-6.696747106,-7.047480986,-7.365361758,-7.819521469,

-1.465377096,-1.517423008,-1.729677406,-2.005098778,-2.269142642,-2.504302086,-2.712253936,-2.896603996,-3.204030615,-3.448291853,-3.596056975,-3.691796056,-3.755310021,-3.809620895,-3.859475869,-3.908091995,-3.953568722,-4.002054034,-4.050652716,-4.099938566,-4.219300869,-4.327900853,-4.525029333,-4.695006729,-4.845868619,-4.982607103,-5.111238168,-5.230496698,-5.438895592,-5.635179352,-5.814456452,-6.155752059,-6.464477994,-6.907401845,

-1.147740744,-1.189588412,-1.36201791,-1.58302078,-1.795626934,-1.989926448,-2.166781928,-2.326234933,-2.597936775,-2.810214599,-2.944512363,-3.032553543,-3.092235814,-3.143023967,-3.189732936,-3.236234971,-3.279739649,-3.326711319,-3.374205322,-3.422255793,-3.538202432,-3.643482809,-3.831153376,-3.993097542,-4.137732415,-4.269157261,-4.392600661,-4.507081222,-4.710122532,-4.901281228,-5.076284651,-5.408007256,-5.709560892,-6.145352161,

-1.01389074,-1.052520586,-1.208816651,-1.403311172,-1.587344237,-1.75891603,-1.917914693,-2.062446089,-2.315003478,-2.507472482,-2.635681929,-2.717690337,-2.774510078,-2.823410237,-2.868477021,-2.913934794,-2.956478526,-3.002354185,-3.048594837,-3.095338525,-3.208325954,-3.310625594,-3.494055628,-3.652478719,-3.794190509,-3.923287574,-4.044519663,-4.156858232,-4.355172803,-4.54203329,-4.712710269,-5.038337841,-5.336545055,-5.766898712,

-0.943663827,-0.982559857,-1.129342374,-1.304839078,-1.47282682,-1.629328788,-1.775700889,-1.910423347,-2.149208883,-2.328040273,-2.451286499,-2.528795442,-2.583553493,-2.631118358,-2.675029806,-2.719758307,-2.761631289,-2.806761029,-2.852156976,-2.897973475,-3.008932128,-3.109173793,-3.289791175,-3.445844549,-3.585881578,-3.71318754,-3.832662367,-3.943321758,-4.138296521,-4.322118173,-4.490914785,-4.813129071,-5.107054955,-5.535339665,

-0.879107685,-0.92098147,-1.060569544,-1.219286383,-1.368155735,-1.51199853,-1.646133133,-1.769066121,-1.992793907,-2.159139798,-2.277531371,-2.350458853,-2.403122443,-2.449392315,-2.492089927,-2.536080514,-2.577219812,-2.621612184,-2.666155277,-2.711016823,-2.819895062,-2.917995794,-3.095645992,-3.249783653,-3.387521007,-3.512864593,-3.630334058,-3.739126114,-3.931013151,-4.111052832,-4.277064734,-4.595405323,-4.886282563,-5.309240359,

-5.249894183,-5.500613583,-6.658711798,-7.48292908,-8.030168805,-8.441022672,-8.78265766,-9.058391665,-9.48971138,-9.822744108,-10.02065359,-10.15185078,-10.24378135,-10.32236215,-10.39416696,-10.4604392,-10.5224461,-10.58815171,-10.65363049,-10.71905026,-10.87529298,-11.01530757,-11.27172527,-11.49221029,-11.68724804,-11.86329873,-12.03426507,-12.19164526,-12.46203111,-12.717363,-12.94703956,-13.38031188,-13.76763997,-14.31193754,

-4.221970301,-4.398704656,-5.336331391,-6.110423294,-6.634740094,-7.027513233,-7.354684493,-7.620620266,-8.035514477,-8.357307495,-8.546778171,-8.67226541,-8.759410394,-8.833891757,-8.901943139,-8.964800173,-9.023657688,-9.086074083,-9.148218064,-9.210526817,-9.359780463,-9.49391188,-9.738806813,-9.949941216,-

10.13718811,-10.30670483,-10.47182906,-10.62411985,-10.88852093,-11.13656038,-11.36397222,-11.78948551,-12.17277406,-12.71501075,
-3.320315139,-3.447837357,-4.168727571,-4.869074806,-5.361935138,-5.73412677,-6.041292568,-6.294644685,-6.693261958,-7.005028701,-7.187516834,-7.308565035,-7.391828534,-7.462730049,-7.527566983,-7.587474643,-7.643575314,-7.702970303,-7.761992571,-7.821335479,-7.963963334,-8.092496854,-8.327040076,-8.529676554,-8.710535128,-8.875009471,-9.035743404,-9.184097218,-9.44165746,-9.684589926,-9.906845677,-10.32593374,-10.70646773,-11.24416906,
-2.569416819,-2.665872336,-3.180097291,-3.77616067,-4.22538026,-4.571613912,-4.858439118,-5.099008733,-5.478273983,-5.776405169,-5.950781721,-6.066561143,-6.145356296,-6.212762449,-6.274454823,-6.332113243,-6.386108344,-6.443303293,-6.500406126,-6.558124311,-6.696709145,-6.821139936,-7.047722203,-7.245788641,-7.42154166,-7.581090402,-7.73139459,-7.870713171,-8.119256072,-8.354945867,-8.572347165,-8.982782902,-9.355124438,-9.886730207,
-1.968247072,-2.045633395,-2.408461838,-2.87778634,-3.267514415,-3.58180503,-3.845010391,-4.07014576,-4.433560374,-4.721726869,-4.890627622,-5.002741755,-5.078136315,-5.141648142,-5.199811354,-5.254866957,-5.307576249,-5.361923592,-5.416251357,-5.471184982,-5.604224437,-5.725014101,-5.943283174,-6.134243596,-6.304860775,-6.460188284,-6.606975801,-6.743454706,-6.985199483,-7.213885312,-7.422485262,-7.820911004,-8.183994596,-8.70535033,
-1.496899348,-1.559686596,-1.825368629,-2.176488104,-2.500819852,-2.77995893,-3.022864481,-3.234342469,-3.577393463,-3.85077744,-4.012500635,-4.11922683,-4.19022465,-4.25107858,-4.306284696,-4.358672228,-4.407698782,-4.459590618,-4.511570013,-4.564278345,-4.692603472,-4.809771611,-5.022020647,-5.207666276,-5.373011144,-5.523218587,-5.66493234,-5.796786097,-6.030360403,-6.250363595,-6.451906431,-6.835591339,-7.187234621,-7.692593682,
-1.125630104,-1.179158537,-1.388987822,-1.666581618,-1.932946779,-2.170562929,-2.381125734,-2.568016694,-2.879934986,-3.127914886,-3.278084915,-3.379267437,-3.447036892,-3.504606434,-3.557511115,-3.608969059,-3.657134466,-3.707753356,-3.758199892,-3.809358535,-3.93333559,-4.046333004,-4.252021354,-4.431493512,-4.591293539,-4.736625902,-4.873621245,-5.001047326,-5.226557486,-5.438687146,-5.633796203,-6.006459899,-6.348086342,-6.84288241,
-0.866097862,-0.910404778,-1.08170278,-1.306066096,-1.520658243,-1.716659043,-1.895583003,-2.057230916,-2.333090562,-2.549485292,-2.686903903,-2.778318085,-2.841497919,-2.895414021,-2.945053603,-2.994275832,-3.040357232,-3.089374078,-3.138663025,-3.188534886,-3.308914533,-3.418358027,-3.614118646,-3.78488619,-3.937895071,-4.077444747,-4.208855152,-4.330920725,-4.549562663,-4.755446031,-4.945708109,-5.30926304,-5.644309305,-6.132966493,
-0.762290494,-0.802991706,-0.957919051,-1.152538404,-1.340827597,-1.514896851,-1.675579036,-1.821976604,-2.078724363,-2.275393186,-2.40526491,-2.490533054,-2.550682445,-2.602584302,-2.650466153,-2.698525974,-2.743537826,-2.791350049,-2.839295531,-2.887769657,-3.004972175,-3.111305194,-3.302653507,-3.469587756,-3.619420865,-3.755900005,-3.884305001,-4.003876279,-4.217608533,-4.419179736,-4.605272992,-4.963207579,-5.294144972,-5.777246213,
-0.709527061,-0.750421201,-0.895844207,-1.070746696,-1.240914152,-1.401195691,-1.550266503,-1.686628464,-1.930084289,-2.111786615,-2.236642106,-2.317344383,-2.375322018,-2.425806723,-2.472459451,-2.519716061,-2.563987909,-2.610980495,-2.658025613,-2.705514762,-2.820553821,-2.924747462,-3.113187436,-3.277567658,-3.42485952,-3.559335923,-3.685960685,-3.803843371,-4.014209879,-4.212719119,-4.396924189,-4.75110465,-5.078025758,-5.55890076,
-0.660658285,-0.704567342,-0.842789932,-0.99911638,-1.149767484,-1.29634378,-1.433688876,-1.559963847,-1.788666895,-1.956745607,-2.076700915,-2.152752057,-2.20852328,-2.257639322,-2.303009876,-2.349457985,-2.392932148,-2.439123998,-2.48526405,-2.53174363,-2.64458011,-2.746556425,-2.931953316,-3.093484481,-3.238409374,-3.370938385,-3.495578638,-3.611619213,-3.81875031,-4.013793472,-4.195442886,-4.545605255,-4.870118258,-5.346927109,
-4.359877713,-4.610030428,-5.744234456,-6.557770756,-7.101730749,-7.511110406,-7.851049333,-8.125829321,-8.556191909,-8.888940176,-9.089398128,-9.221498415,-9.315071595,-9.39567485,-9.469409695,-9.537519186,-9.60127988,-9.669062285,-9.736483849,-9.803575267,-9.963941459,-10.10786593,-10.37122409,-10.59831516,-10.79988956,-10.98249199,-11.16017453,-11.3246398,-11.61734669,-11.88910845,-12.13923145,-12.61286731,-13.04664079,-13.66358744,
-3.343304382,-3.521635761,-4.444328605,-5.209306721,-5.730530637,-6.121751467,-6.447293502,-6.712252221,-7.126315455,-7.447887391,-7.639993927,-7.766695106,-7.85579986,-7.932611298,-8.002885189,-8.067873357,-8.128787178,-8.193548546,-8.257900554,-8.322132614,-8.476153315,-8.614835972,-8.867964475,-9.086934456,-9.281840018,-9.458997515,-9.631968245,-9.7924091,-10.0798373,-10.34747426,-10.59447455,-11.0634384,-11.49527533,-12.1127342,

-2.510730391,-2.640983301,-3.354085255,-4.047125366,-4.537169179,-4.907853704,-5.21373606,-5.466211617,-5.864019265,-6.175444769,-6.360477031,-6.482842221,-6.568197242,-6.641555118,-6.708743383,-6.770913547,-6.829200107,-6.891052433,-6.952422721,-7.013831348,-7.161594677,-7.295055571,-7.538566393,-7.749706642,-7.938808104,-8.111450173,-8.280666262,-8.437651313,-8.719268058,-8.981166555,-9.22422902,-9.687511598,-10.11556009,-10.72843021,

-1.857639586,-1.957452486,-2.470984689,-3.062689166,-3.509750643,-3.854521141,-4.140167492,-4.379725178,-4.757975644,-5.055543676,-5.232154316,-5.349116723,-5.429916024,-5.49980907,-5.563885926,-5.623900948,-5.680172244,-5.739877174,-5.799362568,-5.859190044,-6.003020792,-6.132451553,-6.368187081,-6.575185071,-6.759675551,-6.927919537,-7.086860721,-7.234974121,-7.507625547,-7.762011982,-8.000455429,-8.454859492,-8.875563399,-9.481279735,

-1.365547485,-1.446235798,-1.813138039,-2.281968712,-2.670552781,-2.983595509,-3.245797046,-3.469892031,-3.832100546,-4.119490301,-4.290348984,-4.403439432,-4.480704925,-4.546528082,-4.606913407,-4.664248563,-4.719304028,-4.776049427,-4.832629299,-4.889537782,-5.027590152,-5.153227237,-5.380584628,-5.580493942,-5.7598237,-5.923801578,-6.079298242,-6.224684946,-6.489611384,-6.737010765,-6.966281042,-7.408495191,-7.819273379,-8.413029993,

-0.998637863,-1.064532406,-1.336396533,-1.690339194,-2.01510965,-2.293787159,-2.535815998,-2.746242647,-3.087843914,-3.360186002,-3.523387622,-3.630699756,-3.70326159,-3.766322509,-3.823529211,-3.877932535,-3.928918745,-3.98299855,-4.037070517,-4.091595322,-4.224603457,-4.346329013,-4.567147167,-4.761361213,-4.935131023,-5.093775401,-5.244028236,-5.384734488,-5.640476649,-5.879453675,-6.10194068,-6.528740004,-6.927059044,-7.505775005,

-0.717328938,-0.773613246,-0.989918399,-1.272529233,-1.540794518,-1.77868249,-1.98888666,-2.175044858,-2.485419817,-2.732384319,-2.883663189,-2.98542024,-3.054432504,-3.113553316,-3.167981196,-3.221183478,-3.271056573,-3.323614768,-3.375933581,-3.428707794,-3.556878325,-3.673974495,-3.887340632,-4.074630985,-4.242241615,-4.395481517,-4.540612536,-4.676465037,-4.923178407,-5.15375503,-5.368958235,-5.783633031,-6.171325536,-6.737996686,

-0.527740772,-0.5749666,-0.752853186,-0.983644795,-1.201261942,-1.398302038,-1.577248176,-1.738446876,-2.013000239,-2.229094055,-2.367601437,-2.458970129,-2.522995827,-2.578189201,-2.629101419,-2.679880639,-2.727490235,-2.778218168,-2.829134301,-2.880407008,-3.004430549,-3.117453893,-3.320251851,-3.498297223,-3.658569421,-3.80552602,-3.944633615,-4.07468591,-4.313197705,-4.536601204,-4.746271512,-5.151051589,-5.531314659,-6.090936368,

-0.458423092,-0.501800833,-0.662893283,-0.863129231,-1.055734129,-1.231620148,-1.392626392,-1.538821198,-1.794520976,-1.99155637,-2.122006284,-2.20721757,-2.26812044,-2.321153281,-2.370168121,-2.419675952,-2.46611512,-2.515530073,-2.565004552,-2.614798733,-2.735443262,-2.845169577,-3.043150287,-3.21701794,-3.373802811,-3.517261727,-3.652962305,-3.780305757,-4.013416538,-4.232378127,-4.4379898,-4.837242924,-5.21157985,-5.764550591,

-0.425446944,-0.468675068,-0.619890481,-0.800280855,-0.97451101,-1.137477311,-1.287559227,-1.423902689,-1.666740809,-1.848905772,-1.974293198,-2.054959334,-2.113653661,-2.165183486,-2.212887544,-2.261531986,-2.307171368,-2.355704709,-2.404223159,-2.452990275,-2.571354407,-2.678836573,-2.873679412,-3.044783245,-3.198629735,-3.339914243,-3.473730592,-3.599312328,-3.828949983,-4.04473404,-4.247825054,-4.642747114,-5.01335429,-5.562598579,

-0.394946609,-0.440635852,-0.584059275,-0.745011696,-0.900271429,-1.049585891,-1.188524807,-1.315777877,-1.544275249,-1.712990043,-1.833442864,-1.909477994,-1.965931777,-2.016010765,-2.062362631,-2.11014335,-2.154938256,-2.202617484,-2.2501858,-2.297909797,-2.413979236,-2.519169802,-2.710779998,-2.878614629,-3.029974268,-3.169290215,-3.301088218,-3.424831505,-3.650860304,-3.863599035,-4.064284349,-4.454802398,-4.823426622,-5.370292908,

-3.721200854,-3.969255641,-5.099097187,-5.914454918,-6.461791343,-6.873985113,-7.216288263,-7.493073424,-7.926670397,-8.262152561,-8.465035761,-8.598745987,-8.693550791,-8.775361973,-8.850346705,-8.919712354,-8.984708485,-9.053898333,-9.122729567,-9.191315556,-9.355537696,-9.503363053,-9.774632181,-10.00958147,-10.21913899,-10.40987987,-10.59588578,-10.76910146,-11.08746968,-11.3791468,-11.65321996,-12.17164727,-12.65585256,-13.35197926,

-2.7367934,-2.913650407,-3.834172198,-4.601176646,-5.125603356,-5.519493341,-5.847395353,-6.114335937,-6.531657897,-6.855993684,-7.050562737,-7.179039874,-7.269529755,-7.347710807,-7.419369311,-7.485755315,-7.54806435,-7.614381407,-7.680273951,-7.746115274,-7.904288898,-8.04717197,-8.308796033,-8.53619607,-8.739576099,-8.925368456,-9.106858742,-9.276274063,-9.588557562,-9.877889602,-10.1467638,-10.66138859,-11.14369983,-11.84022045,

-1.973503244,-2.102613755,-2.814128092,-3.509128913,-4.002179769,-4.375425638,-4.68360611,-4.938013667,-5.339018038,-5.653133947,-5.8405688,-5.964748359,-6.051552848,-6.126329948,-6.194950761,-6.258567085,-6.318293262,-6.381745721,-6.444712393,-6.507783387,-6.659852215,-6.797662995,-7.049919213,-7.269698899,-

7.467449577,-7.648875513,-7.826439533,-7.992121174,-8.298246383,-8.580002243,-8.845043967,-9.352933535,-9.828838096,-10.51876108,

-1.401342813,-1.500130785,-2.012060159,-2.605347716,-3.055152397,-3.402277929,-3.690133917,-3.93148745,-4.312750408,-4.612850324,-4.791718576,-4.910434207,-4.992643792,-5.063701272,-5.128985922,-5.190258992,-5.247802986,-5.308947538,-5.369865745,-5.431166616,-5.579001219,-5.712705672,-5.957200405,-6.172353999,-6.365161853,-6.541981066,-6.709875373,-6.86734597,-7.164138514,-7.437883773,-7.697663092,-8.194969811,-8.662870287,-9.342858565,

-0.984964007,-1.06462842,-1.430742389,-1.901155285,-2.292104512,-2.607136007,-2.871290058,-3.096969145,-3.461164847,-3.750169108,-3.922588105,-4.036829364,-4.115037821,-4.181869167,-4.243316161,-4.301798793,-4.357608114,-4.41578644,-4.473834178,-4.532277223,-4.674412802,-4.8042049,-5.0401567,-5.247947948,-5.435380754,-5.607734191,-5.771906046,-5.926266367,-6.214612799,-6.481384841,-6.732494166,-7.216753686,-7.673249011,-8.338796029,

-0.679640062,-0.744545377,-1.016411527,-1.37234223,-1.698639639,-1.978071907,-2.220930705,-2.432045697,-2.774951719,-3.048478648,-3.212928612,-3.321081159,-3.39435924,-3.457715478,-3.515673016,-3.571244273,-3.623419099,-3.678872447,-3.734325533,-3.790297657,-3.927226964,-4.0529728,-4.282059321,-4.483826895,-4.665435002,-4.832256242,-4.991019544,-5.14065673,-5.41877767,-5.677372894,-5.921503035,-6.39066195,-6.83424306,-7.484786567,

-0.445696747,-0.501155623,-0.717723608,-0.999835113,-1.26851606,-1.506836608,-1.717630141,-1.904330289,-2.215645375,-2.463548317,-2.615850323,-2.71743373,-2.786885697,-2.846857817,-2.902196738,-2.956427661,-3.007366621,-3.061182188,-3.11477852,-3.168899897,-3.300717525,-3.421555344,-3.642656065,-3.837032555,-4.01211369,-4.173207034,-4.326646644,-4.471178258,-4.739528898,-4.989374205,-5.22580215,-5.682865978,-6.115762235,-6.752037901,

-0.293087653,-0.338705676,-0.515953869,-0.743741344,-0.960859224,-1.158113296,-1.337364061,-1.498928982,-1.774222339,-1.990640683,-2.129448481,-2.221491038,-2.286115851,-2.342018151,-2.393712536,-2.445405563,-2.493974053,-2.545849296,-2.597945572,-2.650472375,-2.777868996,-2.89433927,-3.104454434,-3.289301905,-3.456739949,-3.611279025,-3.758432176,-3.897128073,-4.157365044,-4.400172056,-4.630963693,-5.07737467,-5.50189502,-6.129519814,

-0.239628186,-0.281601362,-0.442182813,-0.640976233,-0.830705737,-1.005729788,-1.166920528,-1.313365164,-1.569414384,-1.766257808,-1.897813799,-1.983603015,-2.045034118,-2.098706549,-2.148438972,-2.198793169,-2.246130476,-2.296628154,-2.347230469,-2.398224357,-2.522109264,-2.635183559,-2.840291724,-3.020846459,-3.184692785,-3.335994428,-3.480082291,-3.616304393,-3.870813579,-4.109174103,-4.33598311,-4.777043221,-5.193609212,-5.812978184,

-0.215407864,-0.257374743,-0.408200682,-0.587461128,-0.759851194,-0.920599304,-1.069843013,-1.206374033,-1.448938655,-1.631544106,-1.757996042,-1.839179279,-1.898365457,-1.950498088,-1.998883387,-2.048338629,-2.094839333,-2.144420393,-2.194036621,-2.243977581,-2.365504576,-2.476272494,-2.678149856,-2.855871551,-3.017279649,-3.166493724,-3.308677564,-3.443131198,-3.694158856,-3.929243819,-4.153027513,-4.589031776,-5.001842854,-5.615851177,

-0.193206796,-0.237679233,-0.380620609,-0.541356262,-0.694546743,-0.842083597,-0.979455164,-1.105270822,-1.33294045,-1.502675395,-1.624157296,-1.700641018,-1.757550586,-1.808200013,-1.855207144,-1.90376493,-1.949393975,-1.99809112,-2.046736963,-2.095614191,-2.214784722,-2.323223976,-2.521797143,-2.69688601,-2.855755837,-3.003069904,-3.14327329,-3.275935193,-3.523044905,-3.755552808,-3.97708731,-4.408085904,-4.819273663,-5.432205991,

-2.890333243,-3.134750957,-4.222614709,-5.02405629,-5.570806507,-5.984289068,-6.326588827,-6.604164311,-7.039971326,-7.378086482,-7.586502019,-7.721545454,-7.819428748,-7.904485376,-7.982639085,-8.055079019,-8.123050838,-8.196120578,-8.268540385,-8.340268041,-8.512225759,-8.667591065,-8.952786803,-9.201612671,-9.425372913,-9.630716878,-9.829554516,-10.01696033,-10.37997889,-10.70598153,-11.02201504,-11.62101913,-12.19495649,-13.03036827,

-2.00066692,-2.177782058,-3.070877276,-3.826005654,-4.349339171,-4.743788792,-5.07134129,-5.338622852,-5.757681523,-6.084161972,-6.283815129,-6.413667006,-6.507331195,-6.588862323,-6.663769271,-6.733320606,-6.798724789,-6.869001834,-6.938558069,-7.007598061,-7.173692646,-7.324323508,-7.600377451,-7.842131181,-8.060050046,-8.260760514,-8.455214597,-8.638924731,-8.993525051,-9.319589178,-9.625778237,-10.22031034,-10.79015465,-11.62148858,

-1.359150208,-1.490188173,-2.186232995,-2.871613773,-3.363380991,-3.736813309,-4.044873716,-4.299470911,-4.701739269,-5.017359411,-5.209276222,-5.334670187,-5.424574466,-5.50262649,-5.57443288,-5.641163388,-5.703937279,-5.771287507,-5.837913667,-5.904183275,-6.064219506,-6.209850625,-6.47666531,-6.710859701,-6.923074883,-7.119271673,-7.309534428,-7.489039179,-7.836657716,-8.151597377,-8.453805181,-9.038599332,-9.596145949,-10.41488443,

-0.89442538,-0.99582399,-1.503847987,-2.091328841,-2.540166757,-2.887055382,-3.174686237,-3.415727864,-3.7974546,-4.098302201,-4.280784681,-4.400286072,-4.485242877,-4.558836052,-4.626650452,-4.690577341,-4.75075397,-4.815359967,-4.87949445,-4.94349586,-5.098452288,-5.239662058,-5.498525995,-5.726747057,-5.933119704,-6.12406801,-6.306937289,-6.480312369,-6.818602151,-7.125088054,-7.420861447,-7.99167116,-8.539236496,-9.340788147,

-0.56353832,-0.645466934,-1.015796321,-1.486556141,-1.877656645,-2.192365191,-2.456303444,-2.681387834,-3.044068486,-3.331934545,-3.506247084,-3.620044374,-3.700062785,-3.769148875,-3.832867201,-3.893919662,-3.95163878,-4.013300031,-4.074645651,-4.13594136,-4.285373113,-4.422397152,-4.672596133,-4.893533784,-5.09446908,-5.280798989,-5.459464135,-5.628977801,-5.957598139,-6.257043851,-6.545009019,-7.101087514,-7.633901115,-8.414448072,

-0.322292849,-0.389158617,-0.669057282,-1.03098343,-1.358787086,-1.637423985,-1.878862231,-2.088230058,-2.428732691,-2.700481944,-2.865907199,-2.973055752,-3.047717749,-3.112389375,-3.17212058,-3.230246209,-3.284964767,-3.343746741,-3.40232712,-3.460987079,-3.604872281,-3.737542324,-3.980305201,-4.194972237,-4.389904941,-4.570577858,-4.743736348,-4.908548935,-5.22512989,-5.516732483,-5.797155329,-6.337731858,-6.855489688,-7.618026191,

-0.135415715,-0.192539133,-0.417824088,-0.706300544,-0.977603887,-1.216202022,-1.426400727,-1.611946005,-1.920843943,-2.1670938,-2.319981131,-2.419646581,-2.489895635,-2.551355938,-2.608266058,-2.664678389,-2.717826545,-2.774634309,-2.831074855,-2.887631588,-3.025823629,-3.1530558,-3.386741633,-3.5933175,-3.781167845,-3.955606893,-4.123094288,-4.282344787,-4.587721738,-4.869934013,-5.141482955,-5.668519312,-6.173972573,-6.916426652,

-0.015659482,-0.062709081,-0.248057999,-0.482419409,-0.703224493,-0.901909286,-1.081113466,-1.242002664,-1.515422471,-1.730784793,-1.869719434,-1.960380682,-2.025558721,-2.082531301,-2.135424665,-2.189066936,-2.239627351,-2.294182943,-2.348792252,-2.403476384,-2.536545361,-2.658759241,-2.880806531,-3.07754231,-3.257289893,-3.424711223,-3.585479612,-3.738642434,-4.034984598,-4.309427491,-4.574502033,-5.088758899,-5.582826766,-6.311750998,

0.022684948,-0.020626707,-0.18884207,-0.395043281,-0.588508091,-0.76524763,-0.926826166,-1.072932458,-1.327231747,-1.523540356,-1.655405077,-1.73985015,-1.801653124,-1.856171718,-1.906891314,-1.959050465,-2.00824639,-2.061271919,-2.114260757,-2.167305565,-2.296613368,-2.41521686,-2.63176372,-2.823904029,-2.999759655,-3.163820433,-3.321508,-3.472191357,-3.762075154,-4.031859978,-4.292948174,-4.801310765,-5.283807642,-6.001084739,

0.038284462,-0.004672112,-0.162687822,-0.349653642,-0.526590187,-0.68929775,-0.839112188,-0.975565734,-1.216527663,-1.399328039,-1.526005403,-1.605912336,-1.665375018,-1.718215176,-1.767459175,-1.818633121,-1.86690844,-1.918926325,-1.970844547,-2.022787254,-2.149574037,-2.265737594,-2.478751747,-2.667846456,-2.841297313,-3.003172544,-3.158825741,-3.307646139,-3.593948591,-3.860212021,-4.117287354,-4.619283154,-5.097720835,-5.806379874,

0.051018244,0.006297417,-0.142919536,-0.311076097,-0.469609437,-0.619715886,-0.758054075,-0.883967805,-1.110293076,-1.280913199,-1.402553559,-1.477824826,-1.534915953,-1.586134686,-1.633884605,-1.684079265,-1.731413789,-1.782458174,-1.833342441,-1.884177768,-2.008469478,-2.122199117,-2.331648248,-2.518168884,-2.688910872,-2.848885413,-3.002537015,-3.149575164,-3.43134842,-3.69568253,-3.950611297,-4.446392586,-4.923083629,-5.6313148,

-2.460524787,-2.703133854,-3.784720046,-4.587313075,-5.137241313,-5.553439479,-5.897898911,-6.177378549,-6.616341871,-6.957342727,-7.168173301,-7.306012351,-7.406530085,-7.494080466,-7.574731015,-7.649639243,-7.720023659,-7.795440741,-7.870152942,-7.94424918,-8.122400909,-8.284089063,-8.582424835,-8.84438152,-9.081494813,-9.300469359,-9.515493844,-9.719151033,-10.12474808,-10.48403878,-10.83735963,-11.50562592,-12.15178947,-13.09119642,

-1.643410889,-1.819664997,-2.709899702,-3.466461548,-3.992933486,-4.39000041,-4.719760513,-4.988941851,-5.411211209,-5.740602093,-5.942643273,-6.075384238,-6.171756871,-6.255871321,-6.333327171,-6.405406553,-6.473300437,-6.545982877,-6.617869002,-6.689295711,-6.861654854,-7.01866666,-7.307955721,-7.562938958,-7.794200388,-8.008502505,-8.218757641,-8.418351519,-8.813075059,-9.17323876,-9.512806159,-10.17499241,-10.81396897,-11.74488009,

-1.069219365,-1.200314125,-1.894760921,-2.581740183,-3.076598658,-3.452627281,-3.762961145,-4.019467029,-4.425003021,-4.74351644,-4.937761353,-5.066053036,-5.158706885,-5.239355908,-5.313723603,-5.382994233,-5.448262539,-5.518028816,-5.587014849,-5.655696326,-5.822072211,-5.974139196,-6.254244546,-6.501645072,-6.727102676,-6.936718704,-7.142148168,-7.336819906,-7.723129623,-8.069293111,-8.40404568,-9.053291089,-9.675455363,-10.58878615,

-0.653652908,-0.755531864,-1.262179995,-1.850815355,-2.302588075,-2.651933771,-2.941894869,-3.184792383,-3.56973,-3.873414816,-4.058135735,-4.1804914,-4.268204476,-4.344568226,-4.415101026,-4.48173509,-4.544560815,-4.611724256,-4.678319595,-4.744849668,-4.906376379,-5.054108626,-5.326333323,-5.568245431,-5.788293637,-

5.993067171,-6.189769219,-6.377142469,-6.75207335,-7.087483547,-7.414505596,-8.046473403,-8.6562015,-9.546364759,

-0.353872347,-0.436264641,-0.806759873,-1.279356654,-1.673142695,-1.98998322,-2.256070146,-2.482846164,-2.848317361,-3.138667792,-3.315019059,-3.431439034,-3.514088212,-3.585739264,-3.651992919,-3.715627954,-3.77587638,-3.840015715,-3.903745586,-3.96746748,-4.123356126,-4.26689127,-4.530447769,-4.764663453,-4.978930441,-5.178744804,-5.371123146,-5.554593708,-5.917932155,-6.245621997,-6.563628492,-7.17837904,-7.7699429,-8.634413811,

-0.130299228,-0.197415769,-0.478516675,-0.843429293,-1.173498288,-1.453489104,-1.696224761,-1.906629704,-2.249139458,-2.522772149,-2.68981834,-2.799148719,-2.87613895,-2.942873977,-3.004861397,-3.065475612,-3.122654535,-3.183862019,-3.244806211,-3.305862806,-3.456186582,-3.59535161,-3.851314674,-4.078795349,-4.286690562,-4.480556308,-4.66717668,-4.845845255,-5.195619895,-5.515394514,-5.825577392,-6.422966385,-6.996373404,-7.84081938,

0.046242008,-0.011104342,-0.237914808,-0.52715102,-0.799668035,-1.039211276,-1.250475282,-1.43696826,-1.74752582,-1.995446954,-2.149752104,-2.250659059,-2.322864233,-2.386550059,-2.445687185,-2.504451582,-2.559942251,-2.619074916,-2.677798359,-2.736672068,-2.881079441,-3.014571865,-3.260968101,-3.479680789,-3.679948321,-3.86711633,-4.047737719,-4.220428283,-4.557928723,-4.867515387,-5.167689761,-5.749817154,-6.309395818,-7.128665764,

0.156787851,0.110700124,-0.07510666,-0.306905147,-0.527776835,-0.727174349,-0.907123579,-1.068813286,-1.343797772,-1.560292866,-1.699959817,-1.792497632,-1.859669889,-1.918681559,-1.973640836,-2.029499616,-2.082279001,-2.139034988,-2.19581473,-2.252716595,-2.391694119,-2.519835086,-2.754169421,-2.962556895,-3.154220026,-3.33388866,-3.507361413,-3.673731713,-4.001509632,-4.302969729,-4.595835831,-5.163216905,-5.708589244,-6.510692595,

0.192315868,0.149929407,-0.018705787,-0.223779987,-0.414359864,-0.590535656,-0.752725949,-0.899499994,-1.154930826,-1.35172518,-1.485175951,-1.57140621,-1.635094099,-1.691548691,-1.744235999,-1.798518646,-1.849848502,-1.904985729,-1.960076801,-2.015270139,-2.150305031,-2.274701551,-2.503284456,-2.706824254,-2.894357459,-3.070847822,-3.241482252,-3.405511296,-3.72616672,-4.022653144,-4.311510954,-4.872209577,-5.402731393,-6.190454368,

0.206432172,0.164328101,0.005845218,-0.180285061,-0.355382274,-0.515758974,-0.664934586,-0.801978588,-1.043321241,-1.227306992,-1.355547395,-1.437179535,-1.498461969,-1.553173198,-1.604323428,-1.65756172,-1.707914367,-1.761990759,-1.815964928,-1.870010936,-2.002404267,-2.124266001,-2.349154792,-2.54948218,-2.735103932,-2.909456672,-3.077947798,-3.240046723,-3.556885903,-3.849525835,-4.133452605,-4.685964186,-5.212269753,-5.987804342,

0.216709562,0.172882481,0.02347821,-0.144946771,-0.30116189,-0.449333718,-0.586024892,-0.710436959,-0.936456688,-1.108921543,-1.232099773,-1.309019908,-1.367861108,-1.420886151,-1.470489529,-1.522690272,-1.572049485,-1.625099612,-1.677996318,-1.730898867,-1.860685545,-1.980046864,-2.201215325,-2.399650257,-2.582475702,-2.754933379,-2.921388,-3.081685332,-3.393240393,-3.684233693,-3.965736275,-4.510719675,-5.034656774,-5.810462307,

-1.973193989,-2.217117582,-3.274220762,-4.065796124,-4.613704105,-5.029788176,-5.37357912,-5.653061335,-6.092711312,-6.435117049,-6.649783713,-6.791638125,-6.896232757,-6.988434534,-7.073629685,-7.152938793,-7.227550722,-7.307108996,-7.385681828,-7.463666111,-7.652059736,-7.823954646,-8.14190025,-8.423193896,-8.679757518,-8.918389864,-9.154129172,-9.379086485,-9.841627797,-10.24547013,-10.64863966,-11.40842316,-12.14837732,-13.21991188,

-1.261780194,-1.44028435,-2.316978342,-3.064293111,-3.588940402,-3.985613359,-4.314756301,-4.583783825,-5.006658822,-5.337249186,-5.542884393,-5.67966782,-5.780113922,-5.868898391,-5.950854147,-6.027315072,-6.099467481,-6.176293775,-6.252028732,-6.327296133,-6.509830414,-6.676974571,-6.985858296,-7.260116769,-7.510594244,-7.74427525,-7.974627727,-8.194881333,-8.642812752,-9.047780247,-9.431481299,-10.18192897,-10.90826037,-11.96346702,

-0.761860312,-0.895980918,-1.583374093,-2.26320806,-2.756545582,-3.132220827,-3.442331483,-3.698779922,-4.104889887,-4.424331661,-4.621867886,-4.754103991,-4.850789419,-4.936031103,-5.014834049,-5.08842813,-5.157889257,-5.231741223,-5.304568962,-5.377087844,-5.553638787,-5.715824053,-6.015474169,-6.281946712,-6.526258212,-6.754766463,-6.979420564,-7.193630221,-7.630841008,-8.018250032,-8.395080563,-9.127108159,-9.830605534,-10.85967574,

-0.389550052,-0.495034364,-1.000188733,-1.58410936,-2.035018571,-2.383962049,-2.673880692,-2.916517506,-3.301790682,-3.606128078,-3.793693362,-3.91974535,-4.011320245,-4.092085214,-4.166886529,-4.237782521,-4.304744336,-4.375914825,-4.446230929,-4.516465188,-4.687947277,-4.845654867,-5.137278669,-5.397963857,-5.636656706,-5.860164763,-6.07581522,-6.282475561,-6.706448847,-7.080117638,-7.447129327,-8.156931571,-8.843353093,-9.838814828,

-0.116276054,-0.201924401,-0.577122597,-1.049724553,-1.443552093,-1.759934308,-2.026000378,-2.252368664,-2.617112909,-2.90711918,-3.085474645,-3.204830768,-3.290832566,-3.366551444,-3.436776297,-3.504521535,-3.568547558,-3.636623319,-3.704022415,-3.771404661,-3.937210247,-4.09057338,-4.373510909,-4.626213615,-4.858791714,-5.07691581,-5.287837045,-5.490055401,-5.899868632,-6.264399063,-6.621524485,-7.310237187,-7.973129616,-8.934913321,

0.090598867,0.020724076,-0.268375267,-0.638237152,-0.969431369,-1.248680299,-1.490388701,-1.699498698,-2.040389869,-2.313024492,-2.48135513,-2.592908605,-2.672794957,-2.742818013,-2.808298641,-2.872873038,-2.933927028,-2.998930025,-3.063439481,-3.128037657,-3.288118572,-3.436993099,-3.711880805,-3.957355705,-4.183173038,-4.395050813,-4.599864245,-4.797056766,-5.191094793,-5.54728661,-5.895430866,-6.563909389,-7.203319882,-8.14116591,

0.255858628,0.196027574,-0.039613178,-0.333631596,-0.607821849,-0.847328677,-1.058206929,-1.243933042,-1.552915886,-1.800056939,-1.955269199,-2.057326814,-2.131879303,-2.198839381,-2.261224716,-2.323641182,-2.382731896,-2.445407966,-2.507498197,-2.569724226,-2.723380855,-2.866057218,-3.130316063,-3.366022711,-3.583400645,-3.787844395,-3.986049251,-4.176547438,-4.556597253,-4.901145525,-5.237266865,-5.886896128,-6.509542715,-7.413628271,

0.356861495,0.309126882,0.115512485,-0.119386632,-0.342359622,-0.542683318,-0.722660251,-0.884023927,-1.15807352,-1.374276584,-1.514234288,-1.608332911,-1.677725061,-1.739627644,-1.797496405,-1.856770672,-1.912931194,-1.972960736,-2.032847929,-2.092867395,-2.240409271,-2.377024549,-2.628402668,-2.853095169,-3.061083223,-3.257249928,-3.447563512,-3.631135168,-3.999662604,-4.335064401,-4.662173997,-5.293263737,-5.897177171,-6.779322867,

0.388445512,0.344527443,0.168523713,-0.040766436,-0.231932038,-0.408657599,-0.571205389,-0.717908489,-0.972422337,-1.169117168,-1.303435231,-1.391115979,-1.456874031,-1.515989181,-1.571365757,-1.628894512,-1.68344768,-1.741694235,-1.799771465,-1.857968734,-2.001260877,-2.133894508,-2.379018295,-2.59844223,-2.801890724,-2.994783883,-3.182224441,-3.363425464,-3.723688232,-4.05371881,-4.376764494,-5.000023201,-5.584102623,-6.448035054,

0.400149578,0.356746424,0.19124055,0.000592915,-0.176139525,-0.336272571,-0.485288133,-0.622447791,-0.862662996,-1.047536437,-1.176611594,-1.259634008,-1.322913782,-1.380140737,-1.433841217,-1.490218254,-1.543687473,-1.600768491,-1.657648986,-1.714628284,-1.855067418,-1.985001007,-2.226097336,-2.441999821,-2.643739231,-2.83440239,-3.019471826,-3.198564189,-3.554617258,-3.880264775,-4.197577387,-4.80895599,-5.388844491,-6.234833708,

0.406146066,0.361588848,0.20602386,0.032696795,-0.125358002,-0.273909506,-0.410128435,-0.533615198,-0.758395348,-0.932665266,-1.056648326,-1.134874457,-1.195632744,-1.2510295,-1.30305242,-1.35828707,-1.410665768,-1.466614038,-1.522345575,-1.578115788,-1.715746693,-1.843047445,-2.080104673,-2.294435975,-2.493144353,-2.681845281,-2.864752711,-3.041905512,-3.391459805,-3.715976423,-4.030484127,-4.631951891,-5.208275669,-6.05647346,

-1.676980126,-1.918494902,-2.979523965,-3.780383243,-4.335198998,-4.756290235,-5.104504724,-5.387647132,-5.832983188,-6.180173011,-6.397028728,-6.541170944,-6.648864184,-6.743761358,-6.831703786,-6.913754742,-6.991082621,-7.073243405,-7.154281043,-7.23477321,-7.43031259,-7.609722561,-7.943720321,-8.241339301,-8.51461246,-8.770364511,-9.027286735,-9.273342619,-9.78899196,-10.23466349,-10.68262159,-11.52283303,-12.3417063,-13.52140464,

-1.031232922,-1.207061743,-2.088315422,-2.843968412,-3.375299196,-3.776994417,-4.110718983,-4.383469198,-4.8121403,-5.147573152,-5.355361163,-5.494532017,-5.598100912,-5.689647715,-5.774353615,-5.853576037,-5.928471203,-6.007912375,-6.08611094,-6.163863358,-6.353519378,-6.528117614,-6.85301774,-7.143500737,-7.410445312,-7.660934672,-7.911864724,-8.152587363,-8.650732647,-9.0969322,-9.521028559,-10.34870115,-11.14860404,-12.30533093,

-0.570429637,-0.702841213,-1.392220865,-2.079528845,-2.579293262,-2.960015312,-3.274673711,-3.534829756,-3.946853218,-4.271190618,-4.470852527,-4.605535314,-4.705406749,-4.793427442,-4.874998801,-4.951371672,-5.023580453,-5.100063273,-5.175396978,-5.250434095,-5.434194977,-5.603879182,-5.919573844,-6.202179509,-6.462738726,-6.707704157,-6.952143923,-7.18590796,-7.670754633,-8.096883673,-8.511920496,-9.316092588,-10.0877312,-11.21183222,

-0.222536148,-0.327186165,-0.830454701,-1.419348348,-1.876120854,-2.229931795,-2.524444955,-2.770750811,-3.161939872,-3.471175371,-3.660867923,-3.789417997,-3.88427815,-3.96836417,-4.046418551,-4.120520906,-4.190625114,-4.264790671,-4.337913643,-4.411013189,-4.590334307,-4.755736789,-5.063502533,-5.341513591,-5.597379488,-5.838101546,-6.070842198,-6.294527527,-6.76305929,-7.171308686,-7.574722759,-8.351653662,-9.102462949,-10.18401598,

0.036597944,-0.048835427,-0.42288201,-0.899282616,-1.298180928,-1.618827761,-1.889081873,-2.118848123,-2.489561665,-2.784584696,-2.965384928,-3.087382301,-3.176876995,-3.255745921,-3.329080038,-3.399922831,-3.467378107,-3.538333558,-3.608383537,-3.678404179,-3.851821229,-4.012988361,-4.312282054,-4.581952256,-

4.831424234,-5.066455811,-5.294533001,-5.514033887,-5.965647999,-6.363382538,-6.753934101,-7.505675587,-8.227908074,-9.269123777,

0.237015777,0.167131819,-0.122014123,-0.495929545,-0.831443954,-1.114293848,-1.359426801,-1.571418742,-1.91729025,-2.194084952,-2.364451454,-2.47817254,-2.561258322,-2.634326327,-2.702758924,-2.770224834,-2.834160351,-2.90201031,-2.969211308,-3.03647115,-3.204345035,-3.361218892,-3.652560787,-3.914687227,-4.157165429,-4.38578157,-4.607587236,-4.821892086,-5.256090663,-5.644738701,-6.025901464,-6.753279265,-7.446260472,-8.459465959,

0.398469403,0.338559479,0.102655078,-0.193415512,-0.470230179,-0.712206975,-0.925789192,-1.114000372,-1.427233412,-1.677977021,-1.834967942,-1.938702705,-2.016115246,-2.085825657,-2.150979857,-2.21621715,-2.278133583,-2.343619569,-2.408368606,-2.473226936,-2.634536945,-2.78502005,-3.065298053,-3.316923215,-3.550390681,-3.771075993,-3.985817294,-4.192959518,-4.611764705,-4.987614228,-5.354800698,-6.059374533,-6.73256639,-7.705184399,

0.49549667,0.448557096,0.255546994,0.022296494,-0.201582145,-0.403545308,-0.585389606,-0.748745215,-1.026560987,-1.245621738,-1.386863376,-1.482797319,-1.554895382,-1.619383635,-1.679888621,-1.741862993,-1.800743604,-1.863489387,-1.925955928,-1.988551543,-2.143471493,-2.28755188,-2.554478417,-2.794468541,-3.017895875,-3.229694171,-3.43595731,-3.635647283,-4.041302662,-4.406672931,-4.762956663,-5.445810029,-6.09605904,-7.042421197,

0.527122012,0.484008886,0.308666893,0.100717812,-0.088753397,-0.26582157,-0.429803514,-0.57810218,-0.835769301,-1.034428674,-1.170466192,-1.259977822,-1.328306708,-1.389912143,-1.447838816,-1.507985064,-1.565186275,-1.62606804,-1.686664951,-1.747383346,-1.897888881,-2.03789408,-2.298423825,-2.532918266,-2.751589391,-2.960147674,-3.163588157,-3.360902399,-3.757304004,-4.116746252,-4.468823228,-5.142228612,-5.768953133,-6.694334766,

0.539310041,0.496588167,0.331664856,0.142365315,-0.032994157,-0.191897753,-0.341247661,-0.479750077,-0.72235914,-0.909327362,-1.040070346,-1.12493058,-1.190686629,-1.250336218,-1.306529086,-1.365463698,-1.421528041,-1.481188867,-1.540540738,-1.600001834,-1.74753225,-1.88475949,-2.141144077,-2.371962064,-2.589205703,-2.795578902,-2.996506017,-3.19161098,-3.583255872,-3.937840474,-4.282835703,-4.94180675,-5.564298353,-6.467258085,

0.544640037,0.500715224,0.346051269,0.173611573,0.017703159,-0.12961325,-0.265170884,-0.388228873,-0.614594253,-0.791164715,-0.91676088,-0.996799129,-1.05992582,-1.11767061,-1.172128676,-1.229859146,-1.284781559,-1.343249727,-1.401407021,-1.459624336,-1.604232165,-1.738764635,-1.991011422,-2.220721055,-2.434899758,-2.639296159,-2.838020455,-3.031119342,-3.414869738,-3.768532221,-4.110005206,-4.757249394,-5.374898459,-6.28175527,

-1.401245056,-1.632619248,-2.63323279,-3.413961926,-3.970148641,-4.394773668,-4.744157444,-5.029103566,-5.47830021,-5.829557132,-6.054678032,-6.202309405,-6.31575168,-6.416091985,-6.509259136,-6.596342291,-6.678553054,-6.767101363,-6.854097715,-6.939827648,-7.147469511,-7.338466812,-7.693706252,-8.011574333,-8.304662523,-8.580001132,-8.853437999,-9.116483727,-9.672409812,-10.15052505,-10.63257971,-11.53229896,-12.40404626,-13.65413783,

-0.81081115,-0.983725954,-1.820698156,-2.55959277,-3.091271582,-3.495090373,-3.829243066,-4.103045268,-4.534712478,-4.873426527,-5.088867332,-5.231435038,-5.340805063,-5.437873897,-5.527873956,-5.612198301,-5.692031398,-5.777849538,-5.861997678,-5.944975812,-6.146749397,-6.3329213,-6.679030316,-6.989700242,-7.276251984,-7.545987984,-7.813094803,-8.070351782,-8.606478265,-9.084821226,-9.539829477,-10.42268583,-11.27122606,-12.49056582,

-0.374558777,-0.507283183,-1.169027095,-1.84214418,-2.341713816,-2.723809869,-3.039149682,-3.300130408,-3.7145268,-4.041316,-4.24792752,-4.385810404,-4.491390024,-4.584855406,-4.671655692,-4.753078357,-4.830178731,-4.912946856,-4.994202589,-5.074440977,-5.270288488,-5.451506653,-5.788274586,-6.090868073,-6.370593611,-6.634240478,-6.894326377,-7.144065824,-7.664983464,-8.121056709,-8.565855659,-9.421152908,-10.23660809,-11.41722022,

-0.039770106,-0.146076406,-0.638809966,-1.218571129,-1.675302358,-2.029823588,-2.324930867,-2.571521746,-2.964228706,-3.275110617,-3.470855011,-3.602202931,-3.702467771,-3.791464371,-3.874275678,-3.953113081,-4.027827478,-4.107911081,-4.18655873,-4.264420863,-4.455049971,-4.631616347,-4.960082058,-5.256728902,-5.53081492,-5.789392795,-6.03964436,-6.280764456,-6.784441625,-7.221949724,-7.653646307,-8.476990376,-9.267570963,-10.39832838,

0.207661641,0.120579986,-0.253319122,-0.728563449,-1.128414666,-1.449533522,-1.720435401,-1.950208096,-2.320730728,-2.615726933,-2.800999833,-2.924811665,-3.018946162,-3.102501167,-3.180382765,-3.255890463,-3.327473892,-3.40426971,-3.4798129,-3.554625183,-3.739252705,-3.911123992,-4.230577299,-4.518641572,-4.785824016,-5.038090779,-5.282733114,-5.518508923,-6.003504817,-6.429590367,-6.847015356,-7.642370181,-8.40010267,-9.484435362,

0.395902125,0.32446213,0.030505377,-0.348992776,-0.687092152,-0.970380589,-1.215236422,-1.426333709,-1.771205232,-2.047303946,-2.221122283,-2.336165673,-2.42344917,-2.500568569,-2.573122327,-2.645104284,-2.713444649,-2.786850679,-2.859293779,-2.931108223,-3.109636845,-3.276669937,-3.587163877,-3.867020449,-4.126613315,-4.371866488,-4.609683103,-4.839717759,-5.305292463,-5.720669186,-6.127034843,-6.89496678,-7.620636659,-8.668575414,

0.546701865,0.48540831,0.243096051,-0.060705244,-0.34201059,-0.585670712,-0.799882378,-0.987922816,-1.300225803,-1.550498273,-1.710533372,-1.815212524,-1.896204105,-1.969623619,-2.03843244,-2.107717769,-2.173620724,-2.244190977,-2.313781586,-2.382840183,-2.554034755,-2.713999686,-3.012058451,-3.280523083,-3.530371104,-3.767038055,-3.997240575,-4.219487595,-4.667928036,-5.069808036,-5.460625033,-6.203462014,-6.906587624,-7.910179422,

0.63792701,0.588871044,0.388391131,0.145003422,-0.084843017,-0.289583399,-0.472457246,-0.636051052,-0.91343077,-1.133023273,-1.277152916,-1.37387057,-1.448971445,-1.516674986,-1.580387171,-1.646099049,-1.708673619,-1.776050105,-1.842881389,-1.909265519,-2.073148196,-2.225916535,-2.509836131,-2.76626762,-3.005543449,-3.232752803,-3.454053283,-3.668454489,-4.102681856,-4.492693007,-4.871813605,-5.590693254,-6.267538193,-7.239785263,

0.666221963,0.621117676,0.438430668,0.220398546,0.023214467,-0.157575332,-0.3229416,-0.471722285,-0.729101609,-0.929055424,-1.067387894,-1.157660441,-1.228697516,-1.293198232,-1.354033719,-1.417683369,-1.478360554,-1.5436089,-1.608338813,-1.672636962,-1.831662349,-1.979975319,-2.256762142,-2.507209276,-2.741260492,-2.964761446,-3.18286436,-3.394539972,-3.818892074,-4.202450185,-4.57657389,-5.284119364,-5.937458507,-6.887051501,

0.676457127,0.632224268,0.46021002,0.260544476,0.077381083,-0.086486589,-0.238107644,-0.377296288,-0.620079612,-0.808534597,-0.941389331,-1.027042954,-1.095322738,-1.157659553,-1.216569154,-1.278851658,-1.33825047,-1.402107804,-1.465424861,-1.528348238,-1.684089769,-1.829378911,-2.101532964,-2.347961747,-2.580148672,-2.801176791,-3.01655056,-3.225841874,-3.645236242,-4.023672654,-4.389978882,-5.084142666,-5.731625318,-6.659644202,

0.678712933,0.63414495,0.472935926,0.290861055,0.126000473,-0.02656381,-0.165321148,-0.2904172,-0.517445163,-0.695747378,-0.823286045,-0.904162226,-0.969635982,-1.029852043,-1.086845169,-1.147766875,-1.205888413,-1.268383086,-1.330361678,-1.391921463,-1.544453847,-1.686815192,-1.954370078,-2.199152027,-2.428017019,-2.646879187,-2.859851487,-3.066997995,-3.478115748,-3.855621908,-4.218093661,-4.899960941,-5.54213551,-6.469544308,

-1.089411062,-1.302445046,-2.236084847,-3.007515595,-3.575200737,-4.011127133,-4.368413217,-4.660755892,-5.122754237,-5.48550226,-5.724511953,-5.876858709,-5.996936528,-6.103409992,-6.202720163,-6.295890991,-6.384141769,-6.481292867,-6.576501288,-6.669947937,-6.895936932,-7.10481821,-7.494658851,-7.845590093,-8.170913622,-8.477829128,-8.78415743,-9.079184235,-9.707908045,-10.24044314,-10.77872186,-11.77210701,-12.72439208,-14.07763321,

-0.542758725,-0.706740698,-1.496875751,-2.228973756,-2.770439022,-3.183851819,-3.524957072,-3.80527216,-4.248662954,-4.597927251,-4.826391211,-4.973720804,-5.089855116,-5.193212058,-5.289452339,-5.379964012,-5.465932911,-5.560249868,-5.652491458,-5.743053953,-5.962884044,-6.166606552,-6.546530704,-6.889487643,-7.207281066,-7.507489729,-7.805950716,-8.09364304,-8.697738296,-9.229974741,-9.734944218,-10.70539145,-11.62967297,-12.94412794,

-0.121209608,-0.25068574,-0.882556842,-1.550462902,-2.058681176,-2.449282712,-2.77153738,-3.038581895,-3.463906092,-3.800284082,-4.019061678,-4.161603797,-4.273929576,-4.373663375,-4.46668341,-4.554277036,-4.637489808,-4.728593137,-4.817841272,-4.905561477,-5.119233005,-5.317721563,-5.687653035,-6.021785679,-6.331724652,-6.624678032,-6.914472264,-7.193013566,-7.777622102,-8.283842722,-8.776622013,-9.713565224,-10.59923668,-11.86825507,

0.202579726,0.097033876,-0.381045832,-0.958128864,-1.422802103,-1.784787272,-2.086562277,-2.338584644,-2.741190108,-3.060731158,-3.267733987,-3.403410341,-3.510226139,-3.605602181,-3.694730839,-3.779892378,-3.860861597,-3.949247258,-4.035779811,-4.121043243,-4.329273555,-4.522758125,-4.883631553,-5.211450691,-5.51553318,-5.803150705,-6.081225588,-6.349392812,-6.91246696,-7.396436463,-7.873592279,-8.772737063,-9.627400943,-10.83728939,

0.44186101,0.354890825,-0.015095795,-0.494455897,-0.902228473,-1.229886674,-1.507007642,-1.741558322,-2.120177064,-2.422190863,-2.617479619,-2.744831703,-2.844932634,-2.934379519,-3.018146401,-3.09968183,-3.177179372,-3.261928285,-3.345045506,-3.426938521,-3.628527843,-3.816790565,-4.167759744,-4.485750072,-4.781650036,-5.061702357,-5.333200373,-5.595158765,-6.135738706,-6.606875539,-7.065336183,-7.931463842,-8.746856389,-9.904434949,

0.624199441,0.552766337,0.257257103,-0.13229458,-0.478585942,-0.767152618,-1.016372363,-1.230677918,-1.581563066,-1.862952301,-2.045287036,-2.162924577,-2.255401184,-2.337463158,-2.415295957,-2.493008095,-2.567078532,-2.64808459,-2.727831093,-2.80647023,-3.001385399,-3.184211758,-3.525095226,-3.833457192,-

4.12059754,-4.392610556,-4.65638463,-4.911823978,-5.429960391,-5.887834438,-6.333643594,-7.166394756,-7.944760411,-9.058366488,

0.770354896,0.70906253,0.463173419,0.150919575,-0.138109827,-0.386875925,-0.605316614,-0.79658709,-1.113976879,-1.369008622,-1.536715173,-1.64251334,-1.72774483,-1.805925697,-1.879618487,-1.954149623,-2.025358923,-2.103001807,-2.179456269,-2.254943167,-2.441691975,-2.616726416,-2.943742258,-3.239304314,-3.515619836,-3.778116434,-4.033630244,-4.280509872,-4.779162961,-5.222006155,-5.650017273,-6.453477784,-7.206136689,-8.270167616,

0.85631285,0.808536873,0.604519089,0.355721767,0.119256656,-0.090483131,-0.276938991,-0.443425163,-0.725426194,-0.949214451,-1.099549794,-1.198024111,-1.277008791,-1.348950049,-1.417093693,-1.487700491,-1.555258679,-1.62917674,-1.702344404,-1.774696017,-1.953123573,-2.120083299,-2.432099416,-2.714901502,-2.979874808,-3.232209512,-3.478196999,-3.716876143,-4.199761497,-4.628999212,-5.043371694,-5.819451945,-6.540630657,-7.568358781,

0.884816329,0.840976249,0.654412864,0.428524771,0.228048214,0.043754414,-0.124949407,-0.276325988,-0.537314667,-0.740642362,-0.885650422,-0.977485815,-1.052042769,-1.120432471,-1.185380718,-1.253647485,-1.319059992,-1.390510323,-1.46128288,-1.531273068,-1.704267254,-1.866347978,-2.170498375,-2.446755052,-2.706009799,-2.954650469,-3.19754397,-3.433466573,-3.90517233,-4.326860166,-4.735265498,-5.497066802,-6.193651875,-7.19043747,

0.895834933,0.853094978,0.677006415,0.468504727,0.280334163,0.114900209,-0.038633837,-0.180283442,-0.425541041,-0.618276915,-0.757427207,-0.84454208,-0.916101127,-0.982089866,-1.0448932,-1.111605387,-1.175566867,-1.245403815,-1.314554612,-1.382990659,-1.552303272,-1.711071352,-2.010109809,-2.281984721,-2.53975896,-2.785819764,-3.02582074,-3.259138658,-3.725133004,-4.140947844,-4.540480465,-5.288361611,-5.975545333,-6.946218247,

0.897332988,0.854874382,0.68986495,0.496983389,0.326814505,0.171913549,0.032279586,-0.092883073,-0.321230035,-0.504907047,-0.638370338,-0.720579337,-0.789070532,-0.852675058,-0.913351135,-0.978492312,-1.041004835,-1.109252705,-1.176854897,-1.243761192,-1.40947566,-1.565066831,-1.859009387,-2.129806343,-2.383993726,-2.627801554,-2.865217783,-3.096208317,-3.552300229,-3.96663873,-4.361137729,-5.094347803,-5.768296367,-6.736668005,

-0.755077602,-0.961116609,-1.822426954,-2.571273798,-3.142699452,-3.585623232,-3.947192141,-4.244605941,-4.716567899,-5.089551887,-5.344410528,-5.507265782,-5.640053343,-5.759448946,-5.871315621,-5.976616776,-6.07646441,-6.186297624,-6.293429331,-6.398169835,-6.652036979,-6.888062732,-7.328243546,-7.726551596,-8.097107336,-8.446992423,-8.798390922,-9.135930169,-9.857579459,-10.45084506,-11.05252009,-12.13783792,-13.16181845,-14.60403289,

-0.244307691,-0.410147801,-1.152091679,-1.865624097,-2.41010567,-2.829143991,-3.174052885,-3.458650533,-3.91127409,-4.26979845,-4.513120463,-4.670871413,-4.799663321,-4.915855215,-5.024456117,-5.126926935,-5.224478453,-5.331308601,-5.435236672,-5.536751856,-5.783687041,-6.01382174,-6.442741821,-6.831448921,-7.192322085,-7.533333035,-7.874371723,-8.201971652,-8.889608721,-9.483367887,-10.0396892,-11.0926137,-12.08203845,-13.47359021,

0.158187815,0.020743196,-0.584966785,-1.238714038,-1.749771143,-2.14547445,-2.472177006,-2.74340849,-3.177333606,-3.521850412,-3.754419668,-3.906974389,-4.031746281,-4.144110446,-4.249270467,-4.348598265,-4.443152267,-4.546416043,-4.647086037,-4.745489557,-4.985689581,-5.209959666,-5.627556795,-6.00567327,-6.35654967,-6.687908475,-7.017795089,-7.333156902,-7.995217868,-8.555344637,-9.09610101,-10.1071291,-11.05248982,-12.38321534,

0.468035802,0.351348192,-0.122332768,-0.69183666,-1.159873342,-1.526134517,-1.832301254,-2.087638582,-2.497494258,-2.82382289,-3.043047469,-3.187765541,-3.306333156,-3.414602673,-3.516069002,-3.61339268,-3.70606601,-3.806755944,-3.904743421,-4.000800922,-4.235522763,-4.454075833,-4.860871517,-5.232722924,-5.577419828,-5.902810956,-6.215464961,-6.515590349,-7.147222886,-7.678235872,-8.197754582,-9.164748572,-10.0693159,-11.31967789,

0.6925287,0.593939395,0.212332529,-0.270473379,-0.68290403,-1.01428354,-1.295635596,-1.532952119,-1.917860586,-2.225749796,-2.432325185,-2.568060688,-2.679367106,-2.780564453,-2.875626565,-2.968701202,-3.058086052,-3.154352277,-3.24807586,-3.339847092,-3.566337854,-3.77875979,-4.174243663,-4.53413865,-4.868448569,-5.183909969,-5.488397376,-5.780809229,-6.381528705,-6.895074444,-7.389178734,-8.315016875,-9.170335624,-10.35892039,

0.864843765,0.782458179,0.467782586,0.065309548,-0.288297602,-0.581316646,-0.833838757,-1.050061191,-1.405038676,-1.690255506,-1.881763696,-2.006261541,-2.108601267,-2.201701098,-2.289902677,-2.378303074,-2.462724976,-2.55457589,-2.644476107,-2.732565404,-2.951530863,-3.157750048,-3.54141216,-3.889520398,-4.213068113,-4.518530244,-4.813301012,-5.097333176,-5.66873662,-6.165310691,-6.643367424,-7.524657056,-8.334020142,-9.460165831,

1.004110263,0.933184826,0.667334016,0.340268888,0.042465245,-0.211164048,-0.433233093,-0.62681138,-0.947417399,-1.205976005,-1.381344969,-1.493103818,-1.586893423,-1.674236243,-1.75690189,-1.841419499,-1.92236187,-2.010202013,-2.096272407,-2.180732004,-2.390148863,-2.587129411,-2.954142674,-3.286629905,-3.597006416,-3.890747661,-4.17544054,-4.44874201,-4.995577632,-5.473664961,-5.929523657,-6.774761808,-7.543498566,-8.600827444,

1.087417484,1.031497062,0.808419238,0.547020977,0.301771354,0.086677904,-0.103041777,-0.271785361,-0.55687247,-0.784862143,-0.941586749,-1.044940498,-1.131357441,-1.211537467,-1.287846,-1.367983153,-1.444860074,-1.528340657,-1.610487202,-1.691261434,-1.890773948,-2.078070671,-2.428306814,-2.746237477,-3.043393165,-3.325196601,-3.598463889,-3.861835813,-4.388991167,-4.849759478,-5.289688814,-6.096804475,-6.807472277,-7.832580152,

1.116378018,1.065180906,0.860567956,0.623002927,0.415251898,0.226070183,0.054145435,-0.099417487,-0.36299748,-0.570736163,-0.721617565,-0.818163789,-0.899714535,-0.975824647,-1.048469221,-1.125972718,-1.200464026,-1.281167719,-1.360662729,-1.438863144,-1.632467698,-1.814578914,-2.15631848,-2.467096503,-2.757998962,-3.035548815,-3.305175343,-3.565439558,-4.080271638,-4.532659303,-4.965081436,-5.745622746,-6.432863655,-7.441178111,

1.129033068,1.079790324,0.886202718,0.665864227,0.470142258,0.300601037,0.144268062,0.000446101,-0.246982125,-0.444837755,-0.589512374,-0.681259951,-0.759545103,-0.832896108,-0.90305891,-0.978820086,-1.051670634,-1.130535599,-1.208231916,-1.284748795,-1.474324005,-1.652895064,-1.989146163,-2.295072367,-2.584320162,-2.859043318,-3.125513965,-3.382815098,-3.891785851,-4.33706117,-4.76223412,-5.519506108,-6.200365009,-7.191359546,

1.126637698,1.079593581,0.89939954,0.695912644,0.517900273,0.358688633,0.216712041,0.090278511,-0.139657045,-0.329452833,-0.468079303,-0.5548162,-0.6297124,-0.700305768,-0.7680387,-0.842000007,-0.913220688,-0.990257608,-1.066231822,-1.14107506,-1.326670808,-1.501836373,-1.83250817,-2.137268843,-2.422355692,-2.694793462,-2.958407004,-3.212899847,-3.711037673,-4.154257202,-4.572114925,-5.3077134,-5.983878503,-6.974440013,

-0.552618032,-0.747592256,-1.563683135,-2.311060336,-2.895300449,-3.35088544,-3.721946447,-4.028283132,-4.515832764,-4.902770537,-5.172713968,-5.339047682,-5.477471581,-5.603206307,-5.721456412,-5.833057476,-5.939116225,-6.05794989,-6.173599295,-6.28639068,-6.560505781,-6.815987191,-7.291445399,-7.722408863,-8.123434874,-8.501764277,-8.883636511,-9.249227568,-10.02111557,-10.64965555,-11.28943327,-12.42940627,-13.47827041,-14.9480696,

-0.05785856,-0.217073354,-0.927286688,-1.639900456,-2.195379186,-2.625573677,-2.979106245,-3.271672387,-3.73865692,-4.109889752,-4.367055187,-4.528091673,-4.662267444,-4.784629107,-4.899314662,-5.007824995,-5.111379218,-5.226718899,-5.338691252,-5.447754111,-5.713754336,-5.962161863,-6.424204507,-6.843295452,-7.232249533,-7.599316537,-7.968179034,-8.321248654,-9.054278688,-9.682580997,-10.26981835,-11.36999294,-12.37977639,-13.77555917,

0.333084985,0.197222693,-0.38740401,-1.041306179,-1.562027523,-1.967765006,-2.303043166,-2.581709342,-3.028818065,-3.384643646,-3.629758913,-3.785276354,-3.915147957,-4.033394184,-4.1443812,-4.24950438,-4.349803472,-4.461075813,-4.569360374,-4.674894427,-4.933143441,-5.174664249,-5.623453418,-6.029926711,-6.406693112,-6.7618601,-7.117048966,-7.455461333,-8.157378772,-8.750418067,-9.318378909,-10.37155499,-11.3359342,-12.66835877,

0.631934235,0.513641742,0.049777977,-0.520901239,-0.997783981,-1.373002992,-1.687264774,-1.94917018,-2.370830201,-2.707153196,-2.937539761,-3.084680652,-3.207944131,-3.322334077,-3.429816281,-3.533132162,-3.631706908,-3.740286275,-3.845712218,-3.948797477,-4.201090806,-4.435957566,-4.8718855,-5.271390994,-5.641220101,-5.989510953,-6.32308369,-6.642219119,-7.308154304,-7.867035402,-8.410768839,-9.418278013,-10.32563837,-11.57278374,

0.846410801,0.744829534,0.360914942,-0.128620688,-0.54999431,-0.889487906,-1.178455675,-1.4217052,-1.817339263,-2.134206238,-2.351060792,-2.488878725,-2.604573239,-2.711152997,-2.811575684,-2.910129084,-3.00536761,-3.10878398,-3.209213625,-3.30724411,-3.549878106,-3.777723797,-4.200894852,-4.586508002,-4.944000935,-5.280409959,-5.604357468,-5.914521838,-6.54451347,-7.082798914,-7.596036519,-8.555943747,-9.400664956,-10.59154465,

1.012016688,0.926113598,0.602571776,0.188099242,-0.175085506,-0.475631224,-0.734442245,-0.955524266,-1.319128906,-1.611435631,-1.811568016,-1.937284451,-2.043315182,-2.141269838,-2.234300079,-2.327702961,-2.417136651,-2.51562784,-2.611836364,-2.70580457,-2.940175329,-3.161131156,-3.57117541,-3.943293959,-4.288495308,-4.613477516,-4.926232188,-5.226393893,-5.823798582,-6.341476758,-6.836588177,-7.741698738,-8.518142609,-9.652952553,

1.14479386,1.07069839,0.794527835,0.455550556,0.148864189,-0.111395991,-0.339120228,-0.537173217,-0.864972396,-1.129616843,-1.312310585,-1.424461033,-1.521255424,-1.612721606,-1.699647058,-1.788820481,-1.87446756,-1.968455495,-2.060417799,-2.150390619,-2.374294569,-2.585131193,-2.976866389,-3.331603235,-

3.662214417,-3.974157594,-4.275577005,-4.563782066,-5.13426473,-5.631483624,-6.102454889,-6.951871524,-7.690790361,-8.782262156,

1.222973994,1.165395035,0.932525747,0.662337152,0.409303567,0.188153845,-0.006153201,-0.178674777,-0.469908555,-0.703065889,-0.865521491,-0.969394707,-1.058558921,-1.142508243,-1.222787349,-1.307416323,-1.38885128,-1.478039745,-1.565625842,-1.651521172,-1.864519833,-2.064751462,-2.438729801,-2.777850713,-3.094178469,-3.393216673,-3.682177794,-3.9594517,-4.508174511,-4.986259983,-5.440070097,-6.230032267,-6.939659106,-8.013199888,

1.252543015,1.199849102,0.985299222,0.737652737,0.525179349,0.331492146,0.155470404,-0.001405642,-0.269951672,-0.48185059,-0.638508937,-0.735567946,-0.819754372,-0.899435969,-0.97588353,-1.057790494,-1.136777708,-1.222991499,-1.307749839,-1.390926534,-1.597706638,-1.792544678,-2.157662006,-2.489235783,-2.798932289,-3.093561591,-3.378638958,-3.65247366,-4.187915025,-4.657083028,-5.102388902,-5.864432703,-6.562395499,-7.619752366,

1.266700068,1.21634896,1.012750547,0.781541486,0.579882103,0.407790435,0.248797303,0.101941402,-0.149220508,-0.351698983,-0.50171696,-0.593994748,-0.674839038,-0.751613079,-0.825447806,-0.905545844,-0.982833282,-1.067065547,-1.149905055,-1.231293108,-1.433819915,-1.624988913,-1.984372787,-2.31084506,-2.619080868,-2.910744605,-3.192405424,-3.462918329,-3.991865413,-4.453492784,-4.889355074,-5.634724481,-6.329262839,-7.37050092,

1.263531147,1.216368644,1.026941877,0.811182021,0.627056694,0.464883556,0.321486737,0.1943684,-0.037699604,-0.233033144,-0.376548348,-0.463807975,-0.541146673,-0.614988371,-0.686260901,-0.764462733,-0.840061809,-0.922302732,-1.003276623,-1.082885874,-1.28118503,-1.468804825,-1.822323705,-2.147957214,-2.451756833,-2.740893748,-3.01938107,-3.286603706,-3.803550505,-4.262598508,-4.684022583,-5.41795803,-6.113188348,-7.152689059,

-0.336622989,-0.537044436,-1.334237661,-2.045940926,-2.630557561,-3.090652358,-3.463302974,-3.771009485,-4.260916442,-4.650929611,-4.931427798,-5.125597875,-5.288052698,-5.435914368,-5.574367443,-5.704617115,-5.827903824,-5.960367518,-6.088697361,-6.213441371,-6.514323079,-6.793682629,-7.306038706,-7.76506892,-8.187223218,-8.58080907,-8.968104488,-9.334881724,-10.07195758,-10.68368985,-11.28815986,-12.34473028,-13.27480624,-14.60757968,

0.113565922,-0.052585089,-0.737665017,-1.424971714,-1.980496897,-2.412659718,-2.766940864,-3.060004127,-3.52862339,-3.902248316,-4.169326453,-4.356588909,-4.513696524,-4.657018758,-4.790853605,-4.916967097,-5.036728071,-5.164939585,-5.28886011,-5.409118784,-5.700044601,-5.970382923,-6.465411744,-6.908676785,-7.31483773,-7.693196209,-8.063371694,-8.41372369,-9.107629377,-9.708327653,-10.26287339,-11.25746135,-12.13111002,-13.40781315,

0.471876379,0.328189287,-0.237075431,-0.870532701,-1.391301556,-1.798293954,-2.134928471,-2.414306854,-2.862676993,-3.220151062,-3.474304636,-3.654767049,-3.806491281,-3.944706256,-4.073851069,-4.195609792,-4.311177347,-4.434532955,-4.554022247,-4.670017747,-4.951363202,-5.212930195,-5.691134107,-6.118282612,-6.508641439,-6.871527332,-7.224231719,-7.556681682,-8.216506889,-8.779856736,-9.309712845,-10.24207648,-11.08013467,-12.29462214,

0.750789845,0.624366888,0.167237673,-0.388564161,-0.865757604,-1.241364772,-1.556924877,-1.819114035,-2.241502029,-2.578936326,-2.817095004,-2.987974556,-3.13200727,-3.264492307,-3.388357729,-3.506811784,-3.619218887,-3.738459352,-3.853689787,-3.965739547,-4.237924326,-4.490468494,-4.9520379,-5.366117474,-5.743921093,-6.094571922,-6.426610191,-6.740495497,-7.362753062,-7.892499944,-8.391273191,-9.268681649,-10.05399709,-11.18805251,

0.948138407,0.839263252,0.454518948,-0.026504197,-0.447117811,-0.786255008,-1.076489404,-1.320116724,-1.714500189,-2.030630451,-2.25281521,-2.412383121,-2.546983009,-2.67050199,-2.786217127,-2.898993381,-3.005999951,-3.119399023,-3.229129253,-3.335777752,-3.597130331,-3.840825491,-4.28646784,-4.683601132,-5.045913046,-5.381576292,-5.700270824,-6.001114159,-6.58674932,-7.091322205,-7.558276943,-8.383530999,-9.115208032,-10.19842093,

1.1025296,1.010339713,0.682583372,0.273113139,-0.08745326,-0.385553887,-0.643542892,-0.863622127,-1.225476016,-1.517127434,-1.721889968,-1.868813688,-1.993039631,-2.10591945,-2.213108943,-2.320515622,-2.422656553,-2.530620814,-2.635516289,-2.737528381,-2.989292109,-3.224924087,-3.654875618,-4.036226107,-4.383950926,-4.70584784,-5.011142444,-5.299734452,-5.851941967,-6.332616152,-6.771821784,-7.537618976,-8.225907879,-9.261230555,

1.234578074,1.154828116,0.872837983,0.541117249,0.23669271,-0.022201979,-0.249659031,-0.447459122,-0.774448327,-1.039753257,-1.226897048,-1.358341601,-1.472571677,-1.579749487,-1.680755793,-1.783364618,-1.881188193,-1.984204661,-2.084483963,-2.182136984,-2.422238256,-2.646433668,-3.055638242,-3.417859771,-3.749423662,-4.056825122,-4.349351304,-4.624812104,-5.149699599,-5.607333284,-6.019349769,-6.73737508,-7.405798946,-8.402071027,

1.311982221,1.249839428,1.011802442,0.749713968,0.498918819,0.278497859,0.084312339,-0.088356177,-0.379706579,-0.614332736,-0.780300854,-0.904551245,-1.010904062,-1.109815605,-1.203489806,-1.301168943,-1.394434346,-1.49235528,-1.587959821,-1.681251327,-1.909547446,-2.122207228,-2.512246297,-2.858011828,-3.174465335,-3.468094382,-3.747524993,-4.011793436,-4.517000641,-4.95563498,-5.343998006,-6.026546811,-6.671617148,-7.651732942,

1.342193639,1.284943221,1.065529371,0.824024294,0.61675233,0.424892057,0.248846541,0.09171591,-0.177051745,-0.390872788,-0.552422229,-0.669344512,-0.770270329,-0.864515655,-0.954022183,-1.048826995,-1.139517305,-1.234404574,-1.327123388,-1.417632123,-1.639593473,-1.846785648,-2.227927024,-2.566298841,-2.876226379,-3.166088278,-3.442408918,-3.703335837,-4.197020785,-4.627124014,-5.000622002,-5.670152099,-6.306960612,-7.273400632,

1.356293497,1.301624025,1.093217158,0.868213692,0.670582293,0.502047716,0.344347667,0.197262179,-0.05367355,-0.259921076,-0.414803905,-0.526569457,-0.623821082,-0.714884511,-0.801530075,-0.894413141,-0.983315459,-1.076164566,-1.166906134,-1.255591923,-1.473199002,-1.676693054,-2.052121408,-2.385507676,-2.694855458,-2.981927072,-3.254807801,-3.512443002,-4.000979119,-4.423371798,-4.794734121,-5.44892641,-6.082065918,-7.034818314,

1.351275704,1.300425043,1.106587972,0.896411093,0.717104791,0.55810148,0.416968034,0.29183843,0.059637517,-0.140333954,-0.288660502,-0.395081907,-0.488526503,-0.576398502,-0.660242365,-0.751119408,-0.838240309,-0.929018588,-1.017851918,-1.104688923,-1.317911246,-1.517788868,-1.887291695,-2.220740323,-2.525628029,-2.810010035,-3.07951008,-3.33382825,-3.813236575,-4.230427745,-4.588490329,-5.239764926,-5.873161534,-6.823008307,

-0.277831715,-0.478936514,-1.279763574,-2.000285978,-2.593922943,-3.061984419,-3.441253208,-3.754762848,-4.254355222,-4.652651087,-4.939762288,-5.137188447,-5.303661881,-5.455485557,-5.597653612,-5.731391287,-5.857997954,-5.994344295,-6.126245251,-6.254256423,-6.563138303,-6.849810068,-7.372999526,-7.840772049,-8.269807168,-8.668611158,-9.062061436,-9.433203284,-10.16401737,-10.77459721,-11.38049478,-12.42758069,-13.36207409,-14.72319032,

0.155543647,-0.012502657,-0.698728539,-1.393832909,-1.957356097,-2.39688586,-2.757441148,-3.055878452,-3.533584181,-3.914829751,-4.187767499,-4.377749989,-4.538372975,-4.685124184,-4.822172104,-4.951299988,-5.073919157,-5.205483913,-5.332502765,-5.455563103,-5.75335927,-6.029922368,-6.533821714,-6.983897347,-7.395111112,-7.776978408,-8.151609461,-8.504823725,-9.191867005,-9.791231912,-10.34292616,-11.32838076,-12.21462627,-13.51714005,

0.501597596,0.354093648,-0.212278767,-0.852824602,-1.380686665,-1.794533472,-2.137377069,-2.421896524,-2.878663026,-3.242971246,-3.502233889,-3.684987141,-3.839832606,-3.981150871,-4.113179908,-4.237633359,-4.355741773,-4.482019776,-4.604182968,-4.722569608,-5.009803865,-5.276636669,-5.762105388,-6.194541514,-6.588472577,-6.953464334,-7.309152442,-7.643261252,-8.2942307,-8.857252074,-9.38014214,-10.30572808,-11.16001888,-12.39828699,

0.773899619,0.642432921,0.18136957,-0.380492287,-0.863784306,-1.245588394,-1.566910687,-1.83375217,-2.263753125,-2.60726072,-2.849685932,-3.022397988,-3.169165209,-3.304654997,-3.43129167,-3.552381045,-3.667255455,-3.78921298,-3.906927715,-4.021225327,-4.29881067,-4.555921195,-5.023336881,-5.442034503,-5.822706165,-6.174738122,-6.507161214,-6.820321448,-7.431689608,-7.959398008,-8.448457705,-9.324624996,-10.12434475,-11.28167033,

0.9661582,0.851654112,0.4599051,-0.026465519,-0.452417787,-0.797283994,-1.093040148,-1.341169502,-1.743034285,-2.065103355,-2.291252293,-2.45257683,-2.589673147,-2.71564956,-2.833648924,-2.948650645,-3.057923656,-3.173510185,-3.285221486,-3.393612796,-3.659391437,-3.907060431,-4.357551874,-4.757982794,-5.121895778,-5.457747401,-5.775965632,-6.075450759,-6.649124122,-7.148889486,-7.604657639,-8.431283633,-9.176517843,-10.28298662,

1.118481607,1.020557736,0.684462453,0.270643578,-0.094350306,-0.397363379,-0.660203765,-0.884416978,-1.252977625,-1.549877852,-1.758072691,-1.906566045,-2.032964561,-2.147968466,-2.257162672,-2.366627621,-2.47067834,-2.580537942,-2.68713211,-2.790629132,-3.046305696,-3.285500931,-3.719577154,-4.103155327,-4.451451434,-4.772519996,-5.07638689,-5.362599456,-5.902587014,-6.374085049,-6.804868054,-7.573813784,-8.276679329,-9.334358342,

1.247886335,1.163115281,0.873624757,0.53928987,0.231898462,-0.030615075,-0.261775518,-0.462890913,-0.795458468,-1.065132802,-1.255035005,-1.387537416,-1.50370667,-1.612872722,-1.715743183,-1.820345029,-1.920023449,-2.024780668,-2.126617427,-2.225627859,-2.469142673,-2.6963429,-3.108656731,-3.472021238,-3.803221639,-4.108956326,-4.399252854,-4.671664657,-5.183969122,-5.631721294,-6.034104986,-6.762904939,-7.445508085,-8.463376697,

1.321962806,1.256833681,1.012950985,0.751234223,0.498959306,0.276098027,0.079373849,-0.095710924,-0.391304136,-0.628825343,-0.796472583,-0.922107542,-1.030365344,-1.131139387,-1.226580292,-1.326283514,-

1.421435152,-1.521003943,-1.618050427,-1.712593528,-1.943906797,-2.159142701,-2.55159073,-2.897704736,-3.21312264,-3.504485433,-3.781092259,-4.041876543,-4.534375222,-4.96272581,-5.34516684,-6.042423743,-6.700767618,-7.702758088,

1.35226774,1.292186293,1.067026785,0.825234614,0.6194265,0.426721439,0.248744046,0.089705246,-0.182385255,-0.398020824,-0.561641693,-0.679964744,-0.782765279,-0.878857296,-0.970140366,-1.067065883,-1.159747174,-1.256312266,-1.350477619,-1.442252839,-1.667241466,-1.876999702,-2.260512782,-2.599083432,-2.907818672,-3.195431055,-3.468935505,-3.726168765,-4.206805535,-4.626903329,-4.997923312,-5.68201662,-6.331972191,-7.320166869,

1.367054998,1.309702475,1.095520633,0.870105978,0.673289314,0.505848304,0.347685997,0.199064311,-0.054116878,-0.262349642,-0.419101898,-0.532282009,-0.631367883,-0.724254039,-0.812660041,-0.907727941,-0.998683525,-1.093221205,-1.185424375,-1.275372128,-1.49603214,-1.702110821,-2.079914883,-2.413446706,-2.721864871,-3.006650862,-3.276568372,-3.530360344,-4.006057924,-4.417886819,-4.790873405,-5.458388014,-6.104651812,-7.07981241,

1.362412874,1.309229088,1.109815647,0.897944944,0.720061726,0.56214001,0.421876012,0.297432075,0.064294059,-0.138001101,-0.2879679,-0.395821208,-0.491040427,-0.580702374,-0.666291018,-0.75939803,-0.848634003,-0.941104522,-1.031385433,-1.119484686,-1.335759601,-1.538235045,-1.910073996,-2.244045501,-2.547883528,-2.8297852,-3.096141414,-3.346412279,-3.81358762,-4.219511888,-4.58240128,-5.246983966,-5.893842116,-6.866626752,

-0.284387861,-0.479451148,-1.283930553,-2.018428788,-2.625844267,-3.106413937,-3.495936547,-3.818111423,-4.332086196,-4.74206493,-5.038871794,-5.23758354,-5.405402697,-5.558978595,-5.702691352,-5.837812713,-5.965741912,-6.105487416,-6.240498226,-6.371297733,-6.686199286,-6.977635386,-7.504943024,-7.974396962,-8.403034356,-8.799859744,-9.191181835,-9.55883949,-10.25757393,-10.85406311,-11.45453329,-12.49889933,-13.44784521,-14.8380346,

0.129650736,-0.032906749,-0.71607403,-1.422783681,-1.998012757,-2.448700918,-2.818439914,-3.124648275,-3.615357826,-4.007073758,-4.288602919,-4.47916086,-4.640469695,-4.788305272,-4.926285804,-5.056210882,-5.179563405,-5.31372042,-5.443151239,-5.568365414,-5.870637998,-6.150519743,-6.656109465,-7.105718865,-7.514824454,-7.893232593,-8.264534235,-8.613321969,-9.2710488,-9.85962172,-10.4042856,-11.39137216,-12.29673722,-13.63241059,

0.463623818,0.319410863,-0.241255129,-0.891104655,-1.428710815,-1.852409806,-2.203762501,-2.4953937,-2.963700563,-3.337172022,-3.603937369,-3.786793398,-3.941848141,-4.083880897,-4.216492775,-4.341413549,-4.459935454,-4.588190623,-4.712133238,-4.832079929,-5.1223622,-5.391215308,-5.876459647,-6.30690232,-6.697451939,-7.057956843,-7.40941619,-7.738702808,-8.360913967,-8.915694518,-9.431112774,-10.36353823,-11.24118072,-12.51143348,

0.729048236,0.599249001,0.141908737,-0.426544631,-0.917744191,-1.308322067,-1.637218117,-1.910503183,-2.350998776,-2.70272718,-2.951768448,-3.124370079,-3.271116816,-3.407165708,-3.534228294,-3.655548494,-3.770602255,-3.894037337,-4.013114369,-4.128606824,-4.408278489,-4.666419949,-5.131916237,-5.547497922,-5.923868477,-6.27061956,-6.597385975,-6.904384282,-7.487755231,-8.007251604,-8.490968154,-9.381683337,-10.20279361,-11.39454872,

0.917169741,0.802988005,0.412088944,-0.078964389,-0.511452348,-0.864343563,-1.167183388,-1.421469426,-1.833624992,-2.163732531,-2.396144844,-2.55763302,-2.694712049,-2.820977955,-2.939172961,-3.054059275,-3.163269384,-3.279691797,-3.392194063,-3.501241665,-3.767929696,-4.015791032,-4.463012998,-4.859057711,-5.217532267,-5.547173732,-5.859040757,-6.151937235,-6.699215784,-7.190782426,-7.642645528,-8.486272929,-9.255028925,-10.39780421,

1.067140612,0.968548935,0.631748671,0.215725166,-0.154267231,-0.464197822,-0.73363286,-0.963786138,-1.342026843,-1.646416155,-1.860553373,-2.009543894,-2.136020054,-2.251323524,-2.360714628,-2.470016217,-2.573850854,-2.684137596,-2.791084143,-2.894842989,-3.150552088,-3.389192358,-3.818937075,-4.19699983,-4.538859061,-4.852822052,-5.149578258,-5.428358003,-5.943924506,-6.407158333,-6.836015209,-7.626678941,-8.354405176,-9.450284067,

1.193489406,1.107994542,0.817406437,0.481285282,0.170195655,-0.097693732,-0.334043478,-0.539971459,-0.881037992,-1.157134801,-1.352319734,-1.485321466,-1.601755035,-1.71134333,-1.81452352,-1.918972787,-2.018434972,-2.123345195,-2.225285047,-2.32432639,-2.567266237,-2.793343535,-3.200420522,-3.557437638,-3.881478064,-4.179490392,-4.462035856,-4.726680918,-5.21567736,-5.656430058,-6.060093597,-6.814663355,-7.522742614,-8.579344411,

1.263768139,1.198743141,0.95346131,0.690464158,0.435479428,0.208719066,0.008021703,-0.170791243,-0.473144801,-0.715191436,-0.887145922,-1.013617556,-1.122390604,-1.223749034,-1.319647574,-1.419305426,-1.514333288,-1.613875497,-1.710835348,-1.80521322,-2.035506849,-2.249257056,-2.635643201,-2.974598271,-3.282284491,-3.565478792,-3.833947475,-4.086701402,-4.55726995,-4.980005754,-5.367883706,-6.093052832,-6.7757033,-7.818486819,

1.293396288,1.233172589,1.005624371,0.761175052,0.554255475,0.358955706,0.177736744,0.015554275,-0.262256554,-0.48114347,-0.649106637,-0.768307207,-0.871753766,-0.968554636,-1.060432773,-1.157444059,-1.250128017,-1.346637753,-1.440649778,-1.532208508,-1.756036502,-1.96411031,-2.341184171,-2.672171995,-2.972790528,-3.252007945,-3.517172247,-3.765992547,-4.225526792,-4.640478872,-5.018860683,-5.731058445,-6.405491614,-7.434400395,

1.307905892,1.250376207,1.03319445,0.804501225,0.605851239,0.437191558,0.276987018,0.125663036,-0.132194539,-0.343293817,-0.504115095,-0.618191695,-0.717990038,-0.811661649,-0.900741969,-0.995982981,-1.087029079,-1.181485905,-1.273511186,-1.363191838,-1.582628357,-1.786942659,-2.158063951,-2.483786492,-2.784022512,-3.060191523,-3.321525672,-3.566694237,-4.021548486,-4.429344088,-4.813847866,-5.506450303,-6.177236849,-7.192968329,

1.305194368,1.251710153,1.04809955,0.830508832,0.650738911,0.491622256,0.350412739,0.224892371,-0.011320241,-0.216486822,-0.370207448,-0.479000018,-0.574997175,-0.665524033,-0.751855508,-0.845216466,-0.934628897,-1.026990649,-1.117040275,-1.204825493,-1.419801805,-1.620399986,-1.985308231,-2.311384321,-2.606811805,-2.879658655,-3.137148137,-3.378540676,-3.825773624,-4.228910184,-4.60521906,-5.294080419,-5.966574828,-6.979140386,

-0.314546338,-0.507892016,-1.334756306,-2.077413696,-2.69046452,-3.175424848,-3.56880249,-3.893400971,-4.410332723,-4.822064597,-5.118649231,-5.32320124,-5.495531183,-5.652657066,-5.799350451,-5.937040779,-6.067237251,-6.207812016,-6.343615763,-6.475012332,-6.790877282,-7.082409704,-7.606757757,-8.071455558,-8.493861727,-8.883462386,-9.265963074,-9.624336213,-10.28951722,-10.87060362,-11.45298763,-12.47738362,-13.41434411,-14.79074111,

0.082806473,-0.076351386,-0.769928076,-1.48420976,-2.064395721,-2.518801209,-2.892024538,-3.200461542,-3.693889722,-4.087233211,-4.368372757,-4.564067422,-4.7293354,-4.880148936,-5.020634029,-5.152658483,-5.277763281,-5.412355846,-5.542255109,-5.667786693,-5.970301208,-6.249549155,-6.750904422,-7.194591401,-7.596582154,-7.966968679,-8.328790479,-8.667780435,-9.2939907,-9.86411108,-10.39452623,-11.36449502,-12.25859871,-13.58057458,

0.405411413,0.264646895,-0.297158766,-0.952288146,-1.49394773,-1.920888162,-2.275435934,-2.569231162,-3.040021051,-3.414990257,-3.681258507,-3.868806669,-4.027390613,-4.172079852,-4.306888631,-4.433622152,-4.553625264,-4.682086506,-4.80622594,-4.926229722,-5.216085797,-5.483679313,-5.963796687,-6.387645676,-6.770509827,-7.122547141,-7.464083674,-7.783482261,-8.375067861,-8.913430696,-9.414656248,-10.33171713,-11.19754178,-12.45386283,

0.663370646,0.536773722,0.083530744,-0.486250751,-0.980407968,-1.37384422,-1.705641935,-1.981108033,-2.424275834,-2.777695298,-3.026367237,-3.203512089,-3.353617374,-3.491887644,-3.620741866,-3.743387748,-3.859470749,-3.982725275,-4.101666358,-4.216867798,-4.495343053,-4.751686972,-5.211305376,-5.619033692,-5.986690922,-6.32411525,-6.641730283,-6.939486027,-7.494258219,-7.998896694,-8.469833831,-9.3468117,-10.15627702,-11.3358433,

0.846676899,0.735108471,0.350329218,-0.138580601,-0.572436362,-0.927740547,-1.23327174,-1.489859003,-1.904904329,-2.236923761,-2.469068248,-2.634907736,-2.775061889,-2.90342537,-3.023306962,-3.139261236,-3.24896467,-3.365069584,-3.477365914,-3.586098208,-3.851379592,-4.097113886,-4.537922964,-4.925726529,-5.275163761,-5.595278186,-5.897625263,-6.180953283,-6.701733268,-7.179224257,-7.620572162,-8.451993762,-9.21058428,-10.34120045,

0.993677702,0.897056475,0.56703286,0.156544077,-0.212802735,-0.524059764,-0.795760453,-1.028149035,-1.409550422,-1.71624101,-1.930710733,-2.084253874,-2.21386698,-2.331065993,-2.442099358,-2.552483274,-2.657112414,-2.76703694,-2.873639245,-2.976961643,-3.230987307,-3.467270492,-3.890210152,-4.259759319,-4.592335948,-4.896574974,-5.183679137,-5.45269684,-5.943577597,-6.393835535,-6.813003134,-7.594092758,-8.313437446,-9.397570253,

1.119592141,1.035686998,0.751273389,0.421496891,0.112103358,-0.156366078,-0.394192399,-0.601818985,-0.945909421,-1.224180449,-1.420024692,-1.557346652,-1.67700222,-1.788937469,-1.894017809,-1.999516112,-2.099726272,-2.204271734,-2.305854971,-2.404436152,-2.64561291,-2.86923875,-3.269331719,-3.617793244,-3.932519637,-4.220830845,-4.493631234,-4.748688098,-5.214405198,-5.64336114,-6.03995334,-6.785721612,-7.485729304,-8.530367278,

1.189176417,1.125621418,0.885983759,0.629040795,0.376217635,0.149593166,-0.051983983,-0.232007835,-0.536758365,-0.780133843,-0.95274458,-1.083909205,-1.196055236,-1.299853217,-1.397718199,-1.49839868,-1.594150067,-1.693379822,-1.790015087,-1.883945967,-2.112524713,-2.323886757,-2.703051198,-3.033323142,-3.331678367,-3.605196963,-3.864004571,-4.107333364,-4.555961641,-4.968213007,-5.350690645,-6.06759915,-6.74214028,-7.772844195,

1.218785444,1.159657843,0.936971263,0.697758242,0.49378755,0.299215967,0.117453634,-0.045641104,-0.32555188,-0.54522696,-0.714319889,-0.838134972,-0.944956044,-1.044274122,-1.138210538,-1.23628573,-1.329728404,-1.425991072,-1.519692445,-1.610816081,-1.832941819,-2.038577273,-2.408318519,-2.730521763,-

3.021715774,-3.291330992,-3.546935707,-3.786231659,-4.224625581,-4.628916196,-5.001974497,-5.706235694,-6.372215253,-7.388820612,

1.232839648,1.176170148,0.963394967,0.7399338,0.544055802,0.376779062,0.21664446,0.064649969,-0.194908375,-0.406711554,-0.568727067,-0.687358089,-0.790513292,-0.886761359,-0.977960074,-1.07428994,-1.166130623,-1.260381492,-1.352115816,-1.441361205,-1.659128445,-1.861000015,-2.224704079,-2.54160384,-2.832522525,-3.099040709,-3.350751119,-3.586349662,-4.020371477,-4.417657137,-4.796467511,-5.481565465,-6.143834417,-7.146505704,

1.231811503,1.178678015,0.978328522,0.764669102,0.588107226,0.430478173,0.289882181,0.164530256,-0.072931351,-0.278638771,-0.433597922,-0.546901061,-0.646254357,-0.739422871,-0.827917688,-0.922403322,-1.012644738,-1.104844555,-1.194619857,-1.281968559,-1.495312883,-1.69344371,-2.05088892,-2.368220527,-2.654333429,-2.917318082,-3.165076377,-3.396883074,-3.823937446,-4.216809018,-4.587951622,-5.268938149,-5.932584056,-6.932389949,

-0.405310348,-0.605462172,-1.466863045,-2.219651124,-2.829092503,-3.311206171,-3.703488911,-4.026737742,-4.541207077,-4.950741134,-5.242643821,-5.453033282,-5.631149328,-5.792659994,-5.942989598,-6.083758324,-6.216583238,-6.356536084,-6.491620321,-6.62201903,-6.935086103,-7.223302313,-7.738189146,-8.192689947,-8.604208712,-8.982577115,-9.353663127,-9.700511213,-10.33288364,-10.8981191,-11.4601573,-12.46427581,-13.39148775,-14.75727237,

-0.018752243,-0.183147645,-0.90101087,-1.622573216,-2.200242382,-2.652665986,-3.025264725,-3.33276389,-3.824228605,-4.215728264,-4.492343411,-4.693269728,-4.863706121,-5.018266687,-5.161857304,-5.296448168,-5.423640791,-5.557325039,-5.686293184,-5.810661302,-6.109937448,-6.385433507,-6.876645694,-7.309499169,-7.700203516,-8.05902071,-8.409513374,-8.737099936,-9.333239417,-9.884297354,-10.3999534,-11.35144473,-12.238163,-13.54701344,

0.294843937,0.149401882,-0.428316892,-1.088877353,-1.628404689,-2.053655988,-2.407692517,-2.700809976,-3.169832634,-3.543191128,-3.805115064,-3.997402696,-4.160662539,-4.308750242,-4.446335317,-4.575328415,-4.697138057,-4.824588472,-4.947622378,-5.066299377,-5.352496927,-5.615968156,-6.085622663,-6.498440531,-6.869903137,-7.210321594,-7.540827307,-7.849347234,-8.412343714,-8.933915315,-9.421399385,-10.3226274,-11.1780244,-12.42255694,

0.548640658,0.417565438,-0.046286244,-0.618604395,-1.110503665,-1.502344429,-1.833497639,-2.108466405,-2.550224654,-2.902332786,-3.146900217,-3.328420686,-3.48283605,-3.624250972,-3.755639492,-3.880302388,-3.997963306,-4.120196887,-4.238068003,-4.352002197,-4.626825648,-4.878927408,-5.327896296,-5.724734824,-6.081071228,-6.406959887,-6.713321674,-6.999899176,-7.527519349,-8.01646804,-8.47514741,-9.337550092,-10.13798657,-11.30438111,

0.72726355,0.610985564,0.219263562,-0.267639435,-0.698344984,-1.05222002,-1.357275608,-1.613767325,-2.028983817,-2.361114853,-2.590207454,-2.760494577,-2.904782491,-3.035900768,-3.157975518,-3.275560942,-3.386791747,-3.501687681,-3.612725477,-3.719996276,-3.981276754,-4.222671911,-4.652663702,-5.029547082,-5.367654907,-5.676287907,-5.967580368,-6.240101582,-6.735507095,-7.197481806,-7.627475436,-8.445652804,-9.195253098,-10.31101723,

0.870853011,0.769323649,0.43463004,0.03091267,-0.334579399,-0.645072415,-0.917338452,-1.15069076,-1.53326872,-1.840848189,-2.053157673,-2.211061715,-2.344520546,-2.464675594,-2.577788054,-2.689462817,-2.794995196,-2.903609574,-3.008812212,-3.110553374,-3.360368556,-3.592199696,-4.004212291,-4.362774722,-4.683910241,-4.976559576,-5.252600138,-5.510751566,-5.977814941,-6.413401025,-6.821863157,-7.590834201,-8.300264373,-9.368500127,

0.995824623,0.90750229,0.619674964,0.294395836,-0.011540029,-0.278885127,-0.516646285,-0.72470228,-1.069914959,-1.348923302,-1.542909869,-1.68501006,-1.80839204,-1.922555002,-2.029347129,-2.136018423,-2.237009478,-2.340325326,-2.4405592,-2.537593121,-2.774479259,-2.993485781,-3.382400248,-3.719720397,-4.022872493,-4.299540892,-4.561188242,-4.805531955,-5.248240873,-5.66402987,-6.050780646,-6.785002482,-7.473902619,-8.501653776,

1.066461605,0.998152051,0.755157473,0.500415523,0.250263699,0.025030315,-0.176274338,-0.356416578,-0.661711305,-0.905508611,-1.076907995,-1.211695186,-1.327033456,-1.432931625,-1.532380418,-1.634126675,-1.730563172,-1.8286871,-1.924069092,-2.0165244,-2.240869935,-2.447609488,-2.815200195,-3.133972426,-3.420615969,-3.682430342,-3.930146041,-4.162734617,-4.588725021,-4.988391152,-5.362430241,-6.068066884,-6.731746674,-7.74460911,

1.094460807,1.030655375,0.804676416,0.569767103,0.366495409,0.172472777,-0.008943359,-0.17209549,-0.45285321,-0.673118963,-0.840494827,-0.967802397,-1.077650533,-1.17902731,-1.2745304,-1.37366755,-1.467793817,-1.563051445,-1.655525241,-1.745206346,-1.963125389,-2.164066541,-2.522056271,-2.832536091,-3.111848645,-3.369377172,-3.613501222,-3.841747151,-4.258093661,-4.649488068,-5.01506732,-5.707984505,-6.362672824,-7.361350502,

1.106933347,1.045571442,0.829559202,0.611228698,0.417369045,0.249598146,0.089093438,-0.062867378,-0.32373298,-0.535340853,-0.695783406,-0.817843453,-0.923901877,-1.022192132,-1.114947475,-1.212343392,-1.304881298,-1.39818893,-1.488740094,-1.576554061,-1.790174505,-1.987332675,-2.339171631,-2.64423253,-2.922740252,-3.177011184,-3.417174025,-3.641724619,-4.054048884,-4.438368991,-4.808701962,-5.483898526,-6.134888233,-7.118983082,

1.105618378,1.047571824,0.843666027,0.636025175,0.461952378,0.30448354,0.163110557,0.036559001,-0.202681271,-0.407279116,-0.560791218,-0.677490519,-0.779632662,-0.874836141,-0.964863087,-1.060431822,-1.151387071,-1.24271708,-1.331344053,-1.417271791,-1.62654255,-1.819939571,-2.165441058,-2.470259983,-2.743949535,-2.994496474,-3.230661583,-3.451459739,-3.857455461,-4.237340784,-4.600856064,-5.271950262,-5.923627805,-6.905755479,

-0.839660915,-1.042290903,-1.903441818,-2.644963763,-3.247883029,-3.726394985,-4.115936922,-4.435780483,-4.944341506,-5.347802258,-5.637152491,-5.855368185,-6.035465593,-6.19842498,-6.348928975,-6.489013585,-6.620555598,-6.756376694,-6.887542825,-7.014107538,-7.314061744,-7.587410275,-8.065504758,-8.482463963,-8.856286928,-9.197637229,-9.527899821,-9.835934967,-10.37911718,-10.89086157,-11.39649477,-12.32137564,-13.18717842,-14.46578595,

-0.434321098,-0.602670401,-1.320995719,-2.033868208,-2.604922543,-3.052485272,-3.420888409,-3.724431867,-4.208980217,-4.593973645,-4.867698551,-5.075165945,-5.24651361,-5.401377059,-5.544303257,-5.677384594,-5.802400289,-5.93129909,-6.055850971,-6.176077554,-6.461316029,-6.721153567,-7.17489277,-7.570055738,-7.923652136,-8.246254197,-8.557702714,-8.848330451,-9.364695905,-9.860308315,-10.3301894,-11.20957801,-12.04060072,-13.26529383,

-0.105991869,-0.253831613,-0.835923818,-1.488584435,-2.021187374,-2.440467252,-2.789212708,-3.077942647,-3.539155045,-3.9058438,-4.16492919,-4.362970851,-4.526371423,-4.674275656,-4.810758186,-4.937859984,-5.057169546,-5.179601894,-5.297828303,-5.412017567,-5.68332399,-5.930506916,-6.362658657,-6.738345361,-7.073638091,-7.379027512,-7.672553111,-7.946529598,-8.435236918,-8.907874545,-9.352328834,-10.18811308,-10.98972642,-12.1589293,

0.162441393,0.030614579,-0.439663348,-1.007364448,-1.492254861,-1.877723968,-2.202592528,-2.47326385,-2.907327859,-3.25307792,-3.495428171,-3.682445715,-3.836621479,-3.976794875,-4.106177337,-4.227978828,-4.342287343,-4.458920614,-4.571637168,-4.680723048,-4.939899252,-5.175514539,-5.587484973,-5.94665571,-6.266738636,-6.557837532,-6.83135523,-7.086974125,-7.547721089,-7.993619661,-8.413873688,-9.214770893,-9.968076423,-11.06833024,

0.350613945,0.233718669,-0.157830812,-0.635466461,-1.058241762,-1.406097463,-1.70500142,-1.957498973,-2.366236743,-2.693293744,-2.921257969,-3.096749599,-3.240411389,-3.370334344,-3.49048696,-3.605004731,-3.712427111,-3.821797382,-3.927840388,-4.030496972,-4.27642697,-4.501484908,-4.89474872,-5.235320514,-5.538495333,-5.813815422,-6.073490889,-6.316283859,-6.752133506,-7.173969547,-7.570364211,-8.332893841,-9.039305343,-10.09258007,

0.497525929,0.394931131,0.06473234,-0.323346993,-0.680679191,-0.986301947,-1.25471322,-1.485770033,-1.863993174,-2.168474252,-2.381524667,-2.544909061,-2.677893925,-2.797222927,-2.908592359,-3.017139422,-3.119066234,-3.222041875,-3.321955587,-3.41880296,-3.652681692,-3.867739344,-4.242911664,-4.565715104,-4.852478624,-5.112437419,-5.357796547,-5.587069137,-5.999921007,-6.397620136,-6.775165946,-7.49422564,-8.164421353,-9.171528428,

0.627734697,0.538224257,0.255081472,-0.060745532,-0.362260555,-0.626573693,-0.860982545,-1.06644111,-1.408093453,-1.684519439,-1.879847311,-2.027908667,-2.151163007,-2.264317216,-2.369337849,-2.472695746,-2.569873171,-2.6674444,-2.762252823,-2.854214162,-3.074818404,-3.276816115,-3.628549105,-3.930345959,-4.199273097,-4.443494816,-4.674731858,-4.890915796,-5.28202471,-5.66238327,-6.019826772,-6.707252944,-7.358705717,-8.328260765,

0.702443074,0.631011773,0.39065401,0.135602128,-0.113775306,-0.337289978,-0.536819277,-0.714923468,-1.016608276,-1.25845205,-1.432256241,-1.571332404,-1.686275345,-1.791208424,-1.888846699,-1.987022615,-2.079380067,-2.171641127,-2.261488476,-2.348669755,-2.556378932,-2.745866981,-3.074682451,-3.356981045,-3.608958614,-3.838131867,-4.055577872,-4.260146548,-4.635276002,-5.000615073,-5.346212243,-6.00897609,-6.638635906,-7.591096045,

0.726005688,0.658710809,0.434804452,0.200557904,-0.00759559,-0.202715319,-0.383166643,-0.545066028,-0.823539313,-1.043552287,-1.212313412,-1.34337499,-1.452781309,-1.553206467,-1.646903054,-1.74232692,-1.832198196,-1.921484329,-2.00823501,-2.092431316,-2.293123662,-2.476004277,-2.793550337,-3.06623505,-3.309829075,-3.533495992,-3.746321863,-3.945764704,-4.311156295,-4.668263768,-5.006210009,-5.657279735,-6.278883101,-7.218220961,

0.732225831,0.667145265,0.453474187,0.237117245,0.039752562,-0.132414127,-0.294236538,-0.445393272,-0.705578336,-0.916304629,-1.078345405,-1.203783884,-1.309401231,-1.406787865,-1.497787401,-1.591380547,-1.679586489,-1.766853505,-1.851590047,-1.933773872,-2.129818882,-2.308421984,-2.618615888,-2.885000205,-

3.12623471,-3.345822639,-3.554280386,-3.74976925,-4.110982354,-4.461150314,-4.801655064,-5.439118952,-6.056948198,-6.984341371,

0.732033735,0.668882315,0.464935989,0.259554931,0.081129463,-0.080387233,-0.225001349,-0.354527639,-0.594975815,-0.797581091,-0.952861228,-1.072630629,-1.174378,-1.268772581,-1.357077695,-1.448783507,-1.535334416,-1.62060378,-1.70332159,-1.783482039,-1.974876678,-2.149209164,-2.451995631,-2.716049006,-2.951877602,-3.166932314,-3.371049931,-3.562638181,-3.917916476,-4.263589532,-4.59872748,-5.232924933,-5.850151576,-6.775461356,

-1.018147546,-1.227369088,-2.130805478,-2.893959126,-3.493462258,-3.967817429,-4.3552719,-4.672781543,-5.177126941,-5.573808146,-5.852253545,-6.058992526,-6.228330669,-6.378428733,-6.516286002,-6.644011184,-6.763563426,-6.887569664,-7.007366078,-7.122915237,-7.388517636,-7.625750617,-8.100396061,-8.514633737,-8.886270561,-9.225882734,-9.555981751,-9.864032739,-10.4016151,-10.91382643,-11.42478165,-12.35839441,-13.23444169,-14.52815802,

-0.61094234,-0.780922172,-1.52811317,-2.255493591,-2.823968055,-3.26913504,-3.636050162,-3.937967413,-4.419104899,-4.798452304,-5.06250723,-5.259751631,-5.421285637,-5.564464978,-5.695962276,-5.817839124,-5.931915331,-6.050020596,-6.164244084,-6.274521945,-6.528755252,-6.756038908,-7.206495828,-7.599082018,-7.950650452,-8.271675215,-8.583219228,-8.874096335,-9.386097543,-9.884347039,-10.35822343,-11.24662594,-12.08821637,-13.3290697,

-0.281172485,-0.428392933,-1.027420519,-1.6904921,-2.220360644,-2.637690582,-2.984825283,-3.27213385,-3.730248454,-4.092146279,-4.342612026,-4.531428366,-4.685815672,-4.823163083,-4.949291913,-5.066223898,-5.175585396,-5.288129669,-5.396843757,-5.501889176,-5.74484611,-5.962380695,-6.391346791,-6.764582249,-7.098005783,-7.401987567,-7.695860039,-7.970328561,-8.455624884,-8.93168109,-9.380254022,-10.22547885,-11.03914042,-12.22353369,

-0.007374024,-0.137610653,-0.61555555,-1.18932845,-1.671200469,-2.055317408,-2.378516826,-2.648220073,-3.079985725,-3.421910754,-3.656967512,-3.835964277,-3.982058522,-4.112767926,-4.232885235,-4.345508772,-4.450858718,-4.558587939,-4.66280404,-4.763761661,-4.9975604,-5.206454893,-5.615412951,-5.972906488,-6.29176799,-6.582014737,-6.854954071,-7.110190278,-7.567901088,-8.017063031,-8.441681155,-9.253859798,-10.01805385,-11.13322706,

0.185378482,0.069879901,-0.323485031,-0.800696155,-1.220503535,-1.567654399,-1.865223137,-2.117171963,-2.525107748,-2.84988648,-3.072195125,-3.240743665,-3.377181573,-3.498524292,-3.610255046,-3.716280191,-3.815658472,-3.916750877,-4.014878702,-4.109937733,-4.332335616,-4.532838717,-4.923170578,-5.262093183,-5.564085,-5.83861135,-6.097952708,-6.340736558,-6.774016814,-7.199151185,-7.599159627,-8.373068157,-9.090489235,-10.15911789,

0.333654825,0.231833859,-0.097291411,-0.479667393,-0.833934409,-1.139333194,-1.40754281,-1.638960575,-2.017331476,-2.320287401,-2.528694009,-2.685490903,-2.811497461,-2.922836472,-3.026137166,-3.126307129,-3.220064456,-3.31487739,-3.406990306,-3.496376,-3.707419469,-3.89887812,-4.271095588,-4.592153235,-4.877666137,-5.136800958,-5.381804337,-5.611048035,-6.022181405,-6.42317665,-6.804592869,-7.534752776,-8.215540579,-9.238487088,

0.4611546,0.372321749,0.091182183,-0.220969758,-0.519467962,-0.782900725,-1.01625059,-1.221055414,-1.562118158,-1.836074443,-2.026653617,-2.168338257,-2.284689376,-2.389447755,-2.486214998,-2.581076455,-2.669983451,-2.75927778,-2.846181253,-2.930594203,-3.128310275,-3.306916217,-3.655646562,-3.955577242,-4.223161001,-4.466503836,-4.697329287,-4.913468835,-5.303472363,-5.687099146,-6.048763031,-6.74698845,-7.40898039,-8.394619653,

0.532661562,0.461540904,0.223144381,-0.031482979,-0.278750412,-0.5008859,-0.69913488,-0.875955945,-1.175593665,-1.413441355,-1.582400873,-1.714095451,-1.821709755,-1.917928387,-2.006986231,-2.096249624,-2.179961796,-2.263591837,-2.345221657,-2.424558056,-2.608705452,-2.774408669,-3.100017806,-3.380224556,-3.630720077,-3.85893958,-4.075922186,-4.280360983,-4.654599345,-5.023153192,-5.373246977,-6.046947417,-6.686787619,-7.656200904,

0.551702332,0.484358863,0.261749424,0.028141677,-0.180071977,-0.374513108,-0.553545865,-0.714068897,-0.990268018,-1.205673488,-1.368732588,-1.491824563,-1.593432274,-1.68458793,-1.769173661,-1.855101751,-1.935776228,-2.015894618,-2.093918769,-2.169800978,-2.345645982,-2.503662203,-2.817775989,-3.088149679,-3.330094469,-3.552565605,-3.764703883,-3.963911132,-4.328673262,-4.689198913,-5.031720443,-5.693852434,-6.325778416,-7.282101622,

0.552994501,0.487676833,0.27541523,0.060284057,-0.136703381,-0.309348711,-0.470388814,-0.620106975,-0.878154408,-1.08302158,-1.239160654,-1.35623917,-1.453730287,-1.541463265,-1.62298888,-1.706662649,-1.785285972,-1.863003131,-1.938659242,-2.012189806,-2.182487647,-2.335484136,-2.642064589,-2.905991443,-3.145284769,-3.363568914,-3.571311868,-3.76655724,-4.12718746,-4.481019783,-4.827267593,-5.474806208,-6.103014493,-7.047611064,

0.553654696,0.489171896,0.284785154,0.07953678,-0.099040797,-0.260875601,-0.405287473,-0.534792144,-0.772805386,-0.968576043,-1.117756992,-1.228798263,-1.322116126,-1.406491982,-1.484949269,-1.56632478,-1.642909416,-1.718266362,-1.791551824,-1.862736468,-2.02754433,-2.175589042,-2.474584473,-2.735826109,-2.969630658,-3.18331605,-3.386709179,-3.578042879,-3.932977355,-4.282423996,-4.623320708,-5.267708127,-5.895942627,-6.837992602,

1.671335571,1.446330626,0.308517281,-0.497052049,-1.102733187,-1.571142871,-1.959028166,-2.267357596,-2.740256749,-3.101393974,-3.342535465,-3.523205246,-3.665540038,-3.786608821,-3.89749399,-4.000022111,-4.095927322,-4.196222128,-4.294007531,-4.388363909,-4.612609249,-4.8160213,-5.189409719,-5.51201364,-5.799177381,-6.060491704,-6.310554355,-6.542953793,-6.966856797,-7.349673069,-7.701416957,-8.337393376,-8.906983402,-9.727410557,

2.313901537,2.151399673,1.208325515,0.425976238,-0.151719023,-0.588315893,-0.954299012,-1.246462693,-1.693527679,-2.035134048,-2.261171145,-2.430661659,-2.564391895,-2.677527683,-2.781070354,-2.876759854,-2.966050389,-3.058676503,-3.148873862,-3.236036656,-3.443911532,-3.63241507,-3.979905917,-4.279915682,-4.546676755,-4.789073026,-5.021300263,-5.237037474,-5.630004537,-5.985172147,-6.311951861,-6.904082099,-7.436063681,-8.199011925,

2.864862945,2.746430942,2.009867662,1.287777682,0.746137387,0.337424164,-0.00655288,-0.282684316,-0.707348656,-1.032330021,-1.244891299,-1.405302169,-1.531146245,-1.637611598,-1.734909662,-1.824576163,-1.908427734,-1.994162481,-2.07781151,-2.158734403,-2.352121882,-2.527628095,-2.849576704,-3.127373768,-3.373803071,-3.597451852,-3.811746606,-4.010762522,-4.376067296,-4.706245718,-5.009977407,-5.55869765,-6.05252921,-6.760950969,

3.269327082,3.183357163,2.666285863,2.043607086,1.549897123,1.173223309,0.853469353,0.594169937,0.193430425,-0.114159537,-0.313408808,-0.46499779,-0.583298981,-0.683076198,-0.774215415,-0.858235623,-0.936422676,-1.015664499,-1.09303788,-1.168169884,-1.348027174,-1.510903242,-1.812316343,-2.073281682,-2.303836287,-2.512769433,-2.708943834,-2.89110093,-3.226979228,-3.53056048,-3.811058861,-4.317499687,-4.773725593,-5.426094399,

3.59333877,3.52847642,3.178025117,2.691574668,2.269512927,1.932977003,1.639253074,1.396395578,1.021621541,0.732505575,0.543614355,0.400279241,0.289529625,0.196353078,0.111306327,0.032887849,-0.040614085,-0.113414473,-0.184999909,-0.254446354,-0.42220805,-0.574883509,-0.851896926,-1.091505535,-1.304425648,-1.497987067,-1.680193442,-1.849585192,-2.159787384,-2.440600227,-2.698930624,-3.168341701,-3.591958117,-4.198175554,

3.888007793,3.835830929,3.592292491,3.24028693,2.898233016,2.607323968,2.347541602,2.126376432,1.779185192,1.508457606,1.330994316,1.196811349,1.094162181,1.007721529,0.928971167,0.856416439,0.788881639,0.722717962,0.657440299,0.593999872,0.439413139,0.298263399,0.041771958,-0.179781912,-0.37677474,-0.55587968,-0.724664192,-0.881633752,-1.169637422,-1.43048471,-1.670911007,-2.107737303,-2.502436502,-3.069100668,

4.217141562,4.169845441,3.982108477,3.716378569,3.443143775,3.201446829,2.985702519,2.796114963,2.486158098,2.242732695,2.082100416,1.95728211,1.860945912,1.779441306,1.70500663,1.636247343,1.572356228,1.511005944,1.450772065,1.39206796,1.249000714,1.118508971,0.881561711,0.677019801,0.495891068,0.331424844,0.176448726,0.032003535,-0.234394283,-0.476227461,-0.699487554,-1.106651157,-1.475110038,-2.003963656,

4.512456974,4.472284099,4.321881589,4.115185704,3.905178931,3.714489696,3.540778652,3.383732464,3.121516392,2.916198321,2.773820372,2.659107997,2.569795419,2.49509595,2.426585779,2.363369641,2.304482143,2.248954291,2.194241596,2.140833731,2.010556537,1.891727765,1.673221223,1.484386379,1.316668768,1.16392522,1.019965679,0.885632475,0.639800978,0.416114409,0.209351856,-0.16869689,-0.511547756,-1.003929383,

4.673324218,4.635135665,4.499033995,4.324725228,4.150160334,3.987646251,3.83620069,3.698975162,3.460942471,3.27803889,3.144353842,3.036000079,2.950403799,2.877332627,2.810359985,2.748315507,2.690817932,2.637174773,2.584671871,2.533483078,2.409387165,2.296828101,2.094231924,1.918703147,1.761887233,1.618351274,1.482803234,1.35620564,1.123571142,0.911311877,0.714675963,0.355044967,0.028067636,-0.443040463,

4.776206848,4.736186805,4.607417518,4.455244565,4.300509225,4.155547509,4.019437745,3.895091088,3.673860734,3.505974526,3.37875962,3.274941716,3.191854237,3.120006817,3.054036991,2.993036797,2.93616352,2.883893398,2.832931525,2.783230353,2.663255015,2.554896734,2.363023884,2.196394011,2.046358989,1.90895244,1.779160845,1.657896189,1.433977497,1.22926166,1.039421855,0.691629465,0.374978524,-0.082359227,

4.872608405,4.827713188,4.705442451,4.576585218,4.444146279,4.313722024,4.192622039,4.083045091,3.880024534,3

```
.72608906,3.605302107,3.506146912,3.425662456,3.354988212,3.289944081,3.230075798,3.173836199,3.1229214,3.073
456993,3.025457933,2.909546258,2.805399339,2.624249847,2.465827485,2.323031545,2.191872531,2.067849872,1.9519
65254,1.736850125,1.539832436,1.356739297,1.020882319,0.714731308,0.271309087),ncol=Ncol,byrow=TRUE)
}

getLnY <-function(thisFreq, thisMin, thisRrupin){
  inMagnitude = thisMin;
  inRrup = thisRrupin;

        # this algorithm assumes thisFreq is one of the frequencies listed,
        # if(not, you need to write a wrapper that interpolates
  LnY = -999;
  # define these internally
  MagnitudeInterpolationType = "Linear";
  RrupInterpolationType = "Logarithmic";

  medianLnYrefFrequencyLabelColumn = getmedianLnYrefFrequencyLabelColumn();
  medianLnYrefFrequencyValueColumn = getmedianLnYrefFrequencyValueColumn();
  medianLnYrefMomentColumn = getmedianLnYrefMomentColumn();
  MedianLnYrefRrupRow = getMedianLnYrefRrupRow();
  MedianLnYrefDataTable = getMedianLnYrefDataTable();

  if(is.character(thisFreq)){
    if(tolower(thisFreq) == tolower("PGA")){
      thisFreq=-1;
    } else if(tolower(thisFreq) == tolower("PGV")){
      thisFreq=-2;
    }
  }
  medianLnYrefFrequencyColumn = getmedianLnYrefFrequencyValueColumn();

  Mbelow = -888;
  Mabove = 888;
  Nrow = length(medianLnYrefMomentColumn);
  NcolR = length(MedianLnYrefRrupRow);

  for (iRow in 1 : Nrow) {
    Fcheck = -1;
    hereFreq = medianLnYrefFrequencyColumn[iRow];

    if(abs(hereFreq - thisFreq) < 0.000001 ){
      hereM = medianLnYrefMomentColumn[iRow];
      if(abs(hereM - inMagnitude) < 0.001){

        for (iColR in 1:NcolR){
          hereRrup = MedianLnYrefRrupRow[iColR];
          if(abs(hereRrup - inRrup) < 0.000001 ){
            LnY = MedianLnYrefDataTable[iRow, iColR] ; # ' nailed it!
            return(LnY)
          } else {
            if(iColR == NcolR){
              LnY = -888;
              return(LnY)
            }

            nextRrup = MedianLnYrefRrupRow[iColR + 1];
```

```
                if((inRrup - hereRrup) * (inRrup - nextRrup) < 0 ){
                  if(tolower(RrupInterpolationType) == tolower("Logarithmic")){
                    if(inRrup == 0 ){
                      inRrup = 0.0000001;
                    }
                    if(hereRrup == 0 ){
                      hereRrup = 0.0000001;
                    }
                    if(nextRrup == 0 ){
                      nextRrup = 0.0000001;
                    }
                    hereRrup = log(hereRrup);
                    nextRrup = log(nextRrup);
                    inRrup = log(inRrup);
                  }
                  Y0 = MedianLnYrefDataTable[iRow, iColR];
                  Y1 = MedianLnYrefDataTable[iRow, iColR + 1];
                  LnY = Y0 + (Y1 - Y0) / (nextRrup - hereRrup) * (inRrup - hereRrup);
                  return(LnY)
                }
              }
            }
        } else {
          if(hereM > Mbelow && hereM < inMagnitude ){
            Mbelow = hereM;
            Ibelow = iRow;
          }
          if(hereM < Mabove && hereM > inMagnitude ){
            Mabove = hereM;
            Iabove = iRow;
          }
        }

    }



 }; # for iRow



for (iColR in 1 : NcolR) {
    hereRrup = MedianLnYrefRrupRow[iColR];
    if(abs(hereRrup - inRrup) < 0.000001 ){
      Z0 = MedianLnYrefDataTable[Ibelow, iColR] ; # nailed it!
      Z1 = MedianLnYrefDataTable[Iabove, iColR] ; # nailed it!
      iColR = NcolR + 1;
    } else {
      if(iColR < NcolR ){
        nextRrup = MedianLnYrefRrupRow[iColR+1];
        if((inRrup - hereRrup) * (inRrup - nextRrup) < 0 ){
          if(tolower(RrupInterpolationType) == tolower("logarithmic")){
            if(inRrup == 0 ){
              inRrup = 0.1;
            }
            if(hereRrup == 0 ){
              hereRrup = 0.1;
```

```
                }
                if(nextRrup == 0 ){
                   nextRrup = 0.1;
                }
                hereRrup = log(hereRrup);
                nextRrup = log(nextRrup);
                inRrup = log(inRrup);
             }
             iRow = Ibelow;
                Y0 = MedianLnYrefDataTable[iRow, iColR];
                Y1 = MedianLnYrefDataTable[iRow, iColR + 1];
                Z0 = Y0 + (inRrup - hereRrup) * (Y1 - Y0) / (nextRrup - hereRrup);
             iRow = Iabove;
                Y0 = MedianLnYrefDataTable[iRow, iColR];
                Y1 = MedianLnYrefDataTable[iRow, iColR + 1];
                Z1 = Y0 + (inRrup - hereRrup) * (Y1 - Y0) / (nextRrup - hereRrup);
          }
                        }

     }
  }
   # interpolate on Moment
     if(tolower(MagnitudeInterpolationType) == tolower("logarithmic")){
        Mbelow = log(Mbelow);
        Mabove = log(Mabove);
        inMagnitude = log(inMagnitude);
     }
     LnY = Z0 + (inMagnitude - Mbelow) * (Z1 - Z0) / (Mabove - Mbelow);


}


getSigma <- function(thisFreq, inMagnitude, inRrup, sigmaModel){

   if(tolower(sigmaModel) == tolower("ErgodicSigma")){
      Sigma = getSigmaCompErg(thisFreq, inMagnitude, inRrup);
   } else {
      Sigma = getSigmaCompSS(thisFreq, inMagnitude, inRrup);
   }


}


getFrequencyListCompSS <- function(){
   FrequencyListCompSS =
c(100,50,40,33.333,25,20,13.333,10,6.667,5,4,3.333,2.5,2,1.333,1,0.667,0.5,0.333,0.25,0.2,0.133,0.1,-1,-2);
}

getSigmaCompSS <- function(thisFreq, inMagnitude, inRrup){
   WeightSigmaCompSSC=0.63;
   WeightSigmaCompSSH=0.185;
   WeightSigmaCompSSL=0.185;
   WeightSigmaCompSSSum=1;

   FrequencyListCompSS = getFrequencyListCompSS();
   thisRowIndex = -999;
```

```
  NrowFSS= length(FrequencyListCompSS);
  for (iRowF in 1 : NrowFSS) {
    hereFreq = FrequencyListCompSS[iRowF];
    if(abs(hereFreq - thisFreq) < 0.000001){
      thisRowIndex = iRowF;
    }
  }


  # Central Model
      SigCompSS1C =
c(0.7006,0.6998,0.6994,0.699,0.6981,0.6974,0.6955,0.6935,0.6898,0.6862,0.6827,0.6794,0.673,0.6671,0.6541,0.6432,0.626
2,0.6143,0.5998,0.5925,0.5889,0.5858,0.5853,0.7006,0.6303);
      SigCompSS2C =
c(0.6846,0.6838,0.6833,0.6829,0.682,0.6813,0.6793,0.6773,0.6735,0.6698,0.6662,0.6628,0.6563,0.6502,0.6369,0.6256,0.60
82,0.5959,0.5811,0.5736,0.5698,0.5666,0.5661,0.6846,0.6247);
      SigCompSS3C =
c(0.6127,0.6122,0.6119,0.6116,0.611,0.6105,0.6092,0.6079,0.6063,0.6066,0.6068,0.6072,0.6075,0.606,0.5999,0.5921,0.578
1,0.5674,0.5517,0.5438,0.5397,0.5364,0.5357,0.6127,0.5765);
      SigCompSS4C =
c(0.4973,0.4974,0.4974,0.4974,0.4975,0.4975,0.4977,0.4978,0.5007,0.5092,0.5171,0.5254,0.5399,0.5482,0.5577,0.5576,0.5
51,0.5436,0.5267,0.5181,0.5138,0.5102,0.5095,0.4973,0.4896);
    S1 = SigCompSS1C[thisRowIndex];
    S2 = SigCompSS2C[thisRowIndex];
    S3 = SigCompSS3C[thisRowIndex];
    S4 = SigCompSS4C[thisRowIndex];
    SigmaCompSSC = getSigmaCompSSi(inMagnitude, S1, S2, S3, S4);
  # High Model
      SigCompSS1H =
c(0.8193,0.8183,0.8178,0.8173,0.8162,0.8153,0.8129,0.8106,0.8061,0.8019,0.798,0.7943,0.7876,0.7816,0.7697,0.7611,0.75
02,0.7444,0.7391,0.7372,0.7364,0.7358,0.7357,0.8193,0.748);
      SigCompSS2H =
c(0.8075,0.8064,0.8059,0.8054,0.8043,0.8034,0.801,0.7986,0.7941,0.7899,0.7859,0.7822,0.7753,0.7693,0.7572,0.7484,0.73
73,0.7313,0.7258,0.7237,0.7228,0.7222,0.7221,0.8075,0.7431);
      SigCompSS3H =
c(0.7348,0.7343,0.734,0.7338,0.7333,0.7329,0.7317,0.7306,0.7292,0.7295,0.7297,0.73,0.7305,0.7295,0.7256,0.7209,0.7137
,0.7093,0.7033,0.7009,0.6999,0.699,0.6989,0.7348,0.696);
      SigCompSS4H =
c(0.6377,0.6379,0.6381,0.6382,0.6384,0.6385,0.639,0.6395,0.6413,0.6453,0.6492,0.6537,0.6623,0.6681,0.6762,0.6772,0.67
43,0.6738,0.6698,0.6687,0.6682,0.6679,0.6679,0.6377,0.6419);
    S1 = SigCompSS1H[thisRowIndex];
    S2 = SigCompSS2H[thisRowIndex];
    S3 = SigCompSS3H[thisRowIndex];
    S4 = SigCompSS4H[thisRowIndex];
    SigmaCompSSH = getSigmaCompSSi(inMagnitude, S1, S2, S3, S4);
  # Low Model
      SigCompSS1L =
c(0.5879,0.5873,0.587,0.5867,0.586,0.5855,0.584,0.5825,0.5795,0.5765,0.5736,0.5707,0.565,0.5595,0.5467,0.5354,0.5171,
0.5037,0.487,0.4784,0.474,0.4702,0.4695,0.5879,0.522);
      SigCompSS2L =
c(0.5682,0.5676,0.5672,0.5669,0.5662,0.5657,0.5641,0.5626,0.5595,0.5564,0.5533,0.5503,0.5444,0.5387,0.5253,0.5135,0.4
944,0.4803,0.4628,0.4538,0.4491,0.4451,0.4444,0.5682,0.5159);
      SigCompSS3L =
c(0.4981,0.4976,0.4973,0.497,0.4964,0.496,0.4946,0.4933,0.4917,0.4921,0.4924,0.4928,0.4931,0.4913,0.4837,0.4741,0.456
6,0.4433,0.4243,0.4148,0.4101,0.4061,0.4053,0.4981,0.4703);
      SigCompSS4L =
c(0.3901,0.3899,0.3898,0.3897,0.3895,0.3893,0.3887,0.3882,0.3904,0.3994,0.4079,0.4168,0.4326,0.4411,0.4504,0.4508,0.4
482,0.4416,0.4226,0.4124,0.4071,0.4025,0.4017,0.3901,0.3928);
```

```
      S1 = SigCompSS1L[thisRowIndex];
      S2 = SigCompSS2L[thisRowIndex];
      S3 = SigCompSS3L[thisRowIndex];
      S4 = SigCompSS4L[thisRowIndex];
    SigmaCompSSL = getSigmaCompSSi(inMagnitude, S1, S2, S3, S4);


  SigmaCompSS = SigmaCompSSC * WeightSigmaCompSSC / WeightSigmaCompSSSum + SigmaCompSSH *
WeightSigmaCompSSH / WeightSigmaCompSSSum + SigmaCompSSL * WeightSigmaCompSSL / WeightSigmaCompSSSum;


}



getSigmaCompSSi <- function(inMagnitude, S1, S2, S3, S4){

  if(inMagnitude <= 4.5 ){
    SigmaCompSSi = S1;
  } else if(inMagnitude <= 5.0 ){
    SigmaCompSSi = S1 + (S2 - S1) * (inMagnitude - 4.5) / 0.5;
  } else if(inMagnitude <= 5.5 ){
    SigmaCompSSi = S2 + (S3 - S2) * (inMagnitude - 5.0) / 0.5;
  } else if(inMagnitude <= 6.5 ){
    SigmaCompSSi = S3 + (S4 - S3) * (inMagnitude - 5.5) / 1.0;
  } else {
    SigmaCompSSi = S4;
  }

}


getFrequencyListCompErg <- function(){

FrequencyListCompErg=c(100,50,40,33.333,25,20,13.333,10,6.667,5,4,3.333,2.5,2,1.333,1,0.667,0.5,0.333,0.25,0.2,0.133,0.1
,-1,-2);
}

getSigmaCompErg <- function(thisFreq, inMagnitude, inRrup){
  WeightSigmaCompErgC=0.63;
  WeightSigmaCompErgH=0.185;
  WeightSigmaCompErgL=0.185;
  WeightSigmaCompErgSum=1;
  FrequencyListCompErg = getFrequencyListCompErg();


  thisRowIndex = -999;
  NrowFErg = length(FrequencyListCompErg);
  for (iRowF in 1 : NrowFErg) {
    hereFreq = FrequencyListCompErg[iRowF];
    if(abs(hereFreq - thisFreq) < 0.000001){
      thisRowIndex = iRowF;
    }
  }


  # Central Model
```

```
SigCompErg1C=c(0.7618,0.761,0.7646,0.7676,0.781,0.792,0.7881,0.7726,0.7597,0.7556,0.7523,0.7488,0.7412,0.7335,0.715
1,0.6992,0.6742,0.6562,0.6337,0.6212,0.6139,0.6053,0.6016,0.7618,0.7045);

SigCompErg2C=c(0.7471,0.7463,0.75,0.753,0.7667,0.7779,0.7739,0.7581,0.745,0.7408,0.7374,0.7338,0.7261,0.7182,0.6995,
0.6832,0.6576,0.6392,0.616,0.6032,0.5956,0.5867,0.5829,0.7471,0.6995);

SigCompErg3C=c(0.6819,0.6814,0.6856,0.6891,0.7044,0.7169,0.7133,0.6969,0.685,0.6844,0.6844,0.6842,0.6825,0.6786,0.6
661,0.6528,0.6299,0.6127,0.5885,0.5749,0.567,0.5576,0.5535,0.6819,0.6569);

SigCompErg4C=c(0.5809,0.5808,0.5861,0.5905,0.609,0.6239,0.6214,0.6039,0.5941,0.6001,0.6065,0.6129,0.6231,0.6274,0.6
283,0.6215,0.605,0.5907,0.5651,0.5509,0.5426,0.5327,0.5285,0.5809,0.5825);
    S1 = SigCompErg1C[thisRowIndex];
    S2 = SigCompErg2C[thisRowIndex];
    S3 = SigCompErg3C[thisRowIndex];
    S4 = SigCompErg4C[thisRowIndex];
    SigmaCompErgC = getSigmaCompErgi(inMagnitude, S1, S2, S3, S4);
  # High Model

SigCompErg1H=c(0.8747,0.8737,0.8767,0.8791,0.8909,0.9011,0.899,0.8858,0.8725,0.8668,0.8621,0.8575,0.8485,0.8402,0.8
228,0.8096,0.7916,0.7807,0.7687,0.7627,0.7591,0.7545,0.7519,0.8747,0.8134);

SigCompErg2H=c(0.8635,0.8625,0.8655,0.8679,0.8799,0.8902,0.888,0.8747,0.8613,0.8555,0.8508,0.8461,0.837,0.8286,0.81
1,0.7976,0.7792,0.7681,0.7558,0.7496,0.746,0.7412,0.7386,0.8635,0.809);

SigCompErg3H=c(0.7963,0.7958,0.7993,0.8022,0.8158,0.8275,0.8267,0.8137,0.8023,0.8004,0.7994,0.7983,0.7956,0.7918,0.
7814,0.7717,0.7568,0.7471,0.7343,0.7276,0.7238,0.7188,0.716,0.7963,0.7661);

SigCompErg4H=c(0.7076,0.7078,0.7121,0.7157,0.7315,0.7452,0.7457,0.7328,0.7233,0.7246,0.727,0.7295,0.734,0.736,0.736
1,0.7314,0.7203,0.7139,0.7025,0.6967,0.6932,0.6883,0.6855,0.7076,0.7163);
    S1 = SigCompErg1H[thisRowIndex];
    S2 = SigCompErg2H[thisRowIndex];
    S3 = SigCompErg3H[thisRowIndex];
    S4 = SigCompErg4H[thisRowIndex];
    SigmaCompErgH = getSigmaCompErgi(inMagnitude, S1, S2, S3, S4);
  # Low Model

SigCompErg1L=c(0.6539,0.6533,0.6575,0.661,0.6758,0.6874,0.682,0.6645,0.6521,0.6496,0.6476,0.6452,0.6392,0.6323,0.61
45,0.5977,0.5703,0.5497,0.5229,0.5076,0.4983,0.4871,0.4826,0.6539,0.6034);

SigCompErg2L=c(0.6361,0.6355,0.6398,0.6434,0.6586,0.6705,0.665,0.6471,0.6343,0.6317,0.6296,0.6271,0.621,0.6138,0.59
54,0.578,0.5496,0.5282,0.5004,0.4844,0.4747,0.463,0.4583,0.6361,0.598);

SigCompErg3L=c(0.5736,0.5731,0.5779,0.5819,0.5987,0.6118,0.606,0.5867,0.5743,0.575,0.5761,0.5767,0.576,0.5723,0.558
6,0.543,0.5156,0.4945,0.4648,0.448,0.4379,0.4256,0.4207,0.5736,0.5587);

SigCompErg4L=c(0.4807,0.4805,0.4864,0.4913,0.5113,0.5264,0.52,0.4982,0.4876,0.4958,0.504,0.5119,0.5244,0.5293,0.529
9,0.5227,0.508,0.4926,0.4624,0.4446,0.4337,0.4205,0.4153,0.4807,0.4936);
    S1 = SigCompErg1L[thisRowIndex];
    S2 = SigCompErg2L[thisRowIndex];
    S3 = SigCompErg3L[thisRowIndex];
    S4 = SigCompErg4L[thisRowIndex];
    SigmaCompErgL = getSigmaCompErgi(inMagnitude, S1, S2, S3, S4);


  SigmaCompErg = SigmaCompErgC * WeightSigmaCompErgC / WeightSigmaCompErgSum + SigmaCompErgH *
WeightSigmaCompErgH / WeightSigmaCompErgSum + SigmaCompErgL * WeightSigmaCompErgL /
WeightSigmaCompErgSum;
```

```
}

getSigmaCompErgi <- function(inMagnitude, S1, S2, S3, S4){

   if(inMagnitude <= 4.5 ){
      SigmaCompErgi = S1;
   } else if(inMagnitude <= 5.0 ){
      SigmaCompErgi = S1 + (S2 - S1) * (inMagnitude - 4.5) / 0.5;
   } else if(inMagnitude <= 5.5 ){
      SigmaCompErgi = S2 + (S3 - S2) * (inMagnitude - 5.0) / 0.5;
   } else if(inMagnitude <= 6.5 ){
      SigmaCompErgi = S3 + (S4 - S3) * (inMagnitude - 5.5) / 1.0;
   } else {
      SigmaCompErgi = S4;
   }

}

####
################################################################################



NGAEastGMM <- function(inPeriodList,inMagnitude,inRrup,inSigmaType,inNsigma){
            # initialize arrays:
                        outTlist<-0
                        outMedianList<-0
                        outSigmaList<-0
                        outMedianMinusNSigmaList<-0
                        outMedianMinusNSigmaList<-0
                        outMedianPlusNSigmaList<-0
                        outMagList<-0
                        outRrupList<-0
                        outNsigmaList<-0
                        outDataTable <- 0
                        outVs30List <- 0
                        Irow <-0

            for (iRow in 1:length(inPeriodList)){
               thisT = getTvalue(inPeriodList[iRow]);
               outTlist[iRow] = thisT;
               PSA = getNGAEastGMM(thisT,inMagnitude,inRrup,"MedianPSA");
               outMedianList[iRow] = PSA;
               Sigma = getNGAEastGMM(thisT,inMagnitude,inRrup,paste(inSigmaType,"Sigma", sep = ""));
               outSigmaList[iRow] = Sigma;
               outMedianMinusNSigmaList[iRow] = exp(log(PSA)-inNsigma*Sigma);
               outMedianPlusNSigmaList[iRow] = exp(log(PSA)+inNsigma*Sigma);
               outMagList[iRow] = inMagnitude;
               outRrupList[iRow] = inRrup;
               outVs30List[iRow] = 3000
            }
            outDataTable =
matrix(c(outTlist,outMagList,outRrupList,outVs30List,outMedianList,outSigmaList,outMedianMinusNSigmaList,outMedianPl
usNSigmaList),ncol=8,byrow=FALSE)
            outDataTable.df <- as.data.frame(outDataTable );
```

```
	Header <-
c('Period_(s)','Magnitude','Rrup_(km)','Vs30_(m/s)','PSA_median_(g)','Sigma_(ln_units)',paste('PSA_median-
',inNsigma,'*Sigma_(g)',sep=""),paste('PSA_median+',inNsigma,'*Sigma_(g)',sep=""));
	names(outDataTable.df) <- Header

	plot(outTlist,outMedianList,log="xy",type="l",col="black", lty = "solid",xlab="Period (s)",ylab="PSA RotD50
5%damp (g)")
	lines(outTlist,outMedianMinusNSigmaList,col="blue", lty = "dashed")
	lines(outTlist,outMedianPlusNSigmaList,col="blue", lty = "longdash")
	medianString = paste("Median: M=" ,inMagnitude ,", Rrup=" , inRrup , "km, Vs30=3000m/s", sep = "")
	minusString = paste("Median - ",inNsigma,"*Sigma (",inSigmaType,")", sep = "")
	plusString = paste("Median + ",inNsigma,"*Sigma (",inSigmaType,")", sep = "")
	legend("bottomleft",legend=c(medianString ,minusString ,plusString ))
	grid(col = "lightgray", lty = "dotted")

	print(outDataTable.df,row.names = FALSE)
}

#########################################################
############## END PROGRAM DATA #####################
#########################################################



# to recompute values for new input you only need to re-run the following lines:
#########################################################
################ USER INPUT #############################
#########################################################
### -------------------------------------------------------------------
### Variables:
###   User-Input Parameters:
###      inMagnitude:    Event Magnitude. Range=4-8.2
###      inRrup:        Rupture Distance (km). Range=1-1500km
###      inSigmaType:    Type of Sigma: Ergodic or Single-Station
###      inNsigma:      Number of Standard Deviations (sigma)
###      inPeriodList:   Array of periods (sec) for computation. Range: 0.01-10, "PGA","PGV", -1 (=PGA), -2 (=PGV)
### -------------------------------------------------------------------
   inMagnitude = 4.5;                                  # Range: 4-8.2
   inRrup = 17.3;                                # range: 0-1500km
   inSigmaType = "Ergodic";    # options: "Ergodic" "SingleStation"
   inNsigma = 1.2;
   inPeriodList =
c(0.01,0.015,0.02,0.025,0.03,0.04,0.0450.05,0.075,0.1,0.15,0.2,0.25,0.3,0.4,0.5,0.75,1,1.5,2,3,4,5,7.5,10,"PGA","PGV"); #
range:0.01-10sec
# RUN
    NGAEastGMM(inPeriodList,inMagnitude,inRrup,inSigmaType,inNsigma)
#########################################################
```

4.3 NGA-East Ground-Motion Characterization Tool – MATLAB

NGA-East_GMMtool_Matlab_R200316.m

```
%%% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%% ---------------------------------------------------------------------
%%% ---------------------------------------------------------------------
%%% -- NGA-East: Central and Eastern North America
%%%            Ground-Motion Characterization Tool
%%%            Matlab Version
%%% ---------------------------------------------------------------------
%%%                 March 2020
%%% ---------------------------------------------------------------------
%%% Ground-Motion Model Tool developed by
%%%        Silvia Mazzoni
%%%          smazzoni@ucla.edu
%%%        B. John Garrick Institute for the Risk Sciences
%%%          https://www.risksciences.ucla.edu/nhr3/gmtools/home
%%% ---------------------------------------------------------------------
%%% Source:
%%%      Report PEER 2018/08 - Central and Eastern North America
%%%      Ground-Motion Characterization - NGA-East Final Report
%%%        Christine Goulet, Yousef Bozorgnia, Norman Abrahamson,
%%%        Nicolas Kuehn, Linda Al Atik, Robert Youngs, and Robert Graves
%%%        apps.peer.berkeley.edu/publications/peer_reports/reports_2018/NGA-East_1MainReport_FINAL.pdf


%%% ---------------------------------------------------------------------
%%% Instruction:
%%%   1. Enter the values of the user-input parameters.
%%%      Model limits are given with each input parameters.
%%%   2. Run full script
%%%   3. Wait while script runs
%%%   4. A plot of the response spectra will open (you can copy and paste it elsewhere)
%%%   5. Result values are printed in Command Window (you can copy and paste them elsewhere)
%%% ---------------------------------------------------------------------
%%%   Notes:
%%%      The Median and Sigma models are described in the report.
%%%      The values are computed for Vs30=3000 (m/s)
%%%      Damping Ratio = 5%
%%%      This script only outputs the Weighted-Average Composite values for Median and Sigma.
%%%      The values for the individual components can be found in the Excel version of the NGA-East GMtool.
%%% ---------------------------------------------------------------------
%%% Variables:
%%%   User-Input Parameters:
%%%      inMagnitude:    Event Magnitude. Range=4-8.2
%%%      inRrup:         Rupture Distance (km). Range=1-1500km
%%%      inSigmaType:    Type of Sigma: Ergodic or Single-Station
%%%      inNsigma:       Number of Standard Deviations (sigma)
%%%      inPeriodList:   Array of periods (sec) for computation. Range: 0.01-10, 'PGA','PGV', -1 (=PGA), -2 (=PGV)
%%%   Output Variables:
%%%     outDataTable:          Matlab Table with all in/out data matrix
%%%     outTlist:              Array of Periods used for computuation (sec) -1 (=PGA), -2 (=PGV)
%%%     outMedianList:         Array of median PSA (g);
%%%     outSigmaList:          Array of Sigma;
%%%     outMedianMinusNSigmaList:   Array of Median - inNsigma*Sigma (g);
%%%     outMedianPlusNSigmaList:    Array of Median + inNsigma*Sigma (g);
%%% ---------------------------------------------------------------------
```

```
%%% ------------------------------------------------------------------
%%% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% initialize
clear all
close all

%% user input:
% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
   inMagnitude = 5; % Range: 4-8.2
   inRrup = 111.2; % range: 0-1500km
   inSigmaType = "SingleStation"; % options: "Ergodic" "SingleStation"
   inNsigma = 1.0;
   inPeriodList =
[0.01,0.015,0.02,0.025,0.03,0.04,0.05,0.075,0.1,0.15,0.2,0.25,0.3,0.4,0.5,0.75,1,1.5,2,3,4,5,7.5,10,"PGA","PGV"]; %
range:0.01-10sec
% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% the following commands compute the output values at each period and
% generate a figure of the data
% The user does not need to edit the following
% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Compute Values at each Period
for iRow = 1:length(inPeriodList)
   thisT = getTvalue(inPeriodList(iRow));
   outTlist(iRow) = thisT;
   PSA = NGAEastGMM(thisT,inMagnitude,inRrup,"MedianPSA");
   outMedianList(iRow) = PSA;
   Sigma = NGAEastGMM(thisT,inMagnitude,inRrup,inSigmaType + "Sigma");
   outSigmaList(iRow) = Sigma;
   outMedianMinusNSigmaList(iRow) = exp(log(PSA)-inNsigma*Sigma);
   outMedianPlusNSigmaList(iRow) = exp(log(PSA)+inNsigma*Sigma);
   MagList(iRow) = inMagnitude;
   RrupList(iRow) = inRrup;
   NsigmaList(iRow)=inNsigma;
   Vs30List(iRow)=3000;
end
% combine Data for output
Header = {'Period_(s)','Magnitude','Rrup_(km)','Vs30_(m/s)','Nsigma','PSA_median_(g)','Sigma','PSA_median-
N*Sigma_(g)','PSA_median+N*Sigma_(g)'};
outDataTable =
table(outTlist.',MagList.',RrupList.',Vs30List.',NsigmaList.',outMedianList.',outSigmaList.',outMedianMinusNSigmaList.',outM
edianPlusNSigmaList.');
outDataTable.Properties.VariableNames =Header;
% output Data:
disp(outDataTable)
% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% ------------------------------------------------------------------
figure()
loglog(outTlist(1:end-2),outMedianList(1:end-2),'k','LineWidth',2)
hold
loglog(outTlist(1:end-2),outMedianMinusNSigmaList(1:end-2),'b--','LineWidth',2)
loglog(outTlist(1:end-2),outMedianPlusNSigmaList(1:end-2),'b-.','LineWidth',2)
legend("Median: M=" +inMagnitude +",Rrup=" + inRrup + "km, Vs30=3000m/s","Median-"+inNsigma+"*Sigma (" +
inSigmaType + ")","Median+"+inNsigma+"*Sigma (" + inSigmaType + ")",'Location','southwest')
xlabel("Period (sec)")
```

```
ylabel("PSA RotD50 5%damp (g)")
grid on

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%
% program DO NOT edit lines below!!!
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%
function thisT = getTvalue(thisTin)
  % code developed and written by Silvia Mazzoni
  %     smazzoni@ucla.edu, March 2020
  if isstring(thisTin)==1
    if lower(thisTin) == lower("PGA")
      thisT=-1;
    elseif lower(thisTin) == lower("PGV")
      thisT=-2;
    else
      thisT = double(thisTin);
    end
  else
    thisT = thisTin;
  end
end

function iNGAEastGMM = NGAEastGMM(thisTin,inMagnitude,inRrup,thisInputLabel)
  % code developed and written by Silvia Mazzoni
  %     smazzoni@ucla.edu, March 2020

  iNGAEastGMM = -999;
  iLnY = -999;

  if lower(thisInputLabel) == lower("ErgodicSigma")
    Frange = getFrequencyListCompErg();
  elseif lower(thisInputLabel) == lower("SingleStationSigma")
    Frange = getFrequencyListCompSS();
  else
    Frange = getmedianLnYrefFrequencyValueColumn();
  end

  Flist = unique(Frange);

  thisT = getTvalue(thisTin);

  for iiT = 1:length(Flist)
    hereF = Flist(iiT);
    if hereF < 0
      hereT = hereF;
    else
      hereT = 1 / hereF;
    end
    if abs(hereT - thisT) < 0.000001
      if lower(thisInputLabel) == lower("ErgodicSigma")
        iLnY = getSigmaCompErg(hereF, inMagnitude, inRrup);
      elseif lower(thisInputLabel) == lower("SingleStationSigma")
```

```
            iLnY = getSigmaCompSS(hereF, inMagnitude, inRrup);
         else
            iLnY = getLnY(hereF, inMagnitude, inRrup);
         end
         iiT = length(Flist) + 1;
      else
         if iiT < length(Flist)
            nextF = Flist(iiT + 1);
            nextT = 1 / nextF;
            if hereT > 0 && nextT > 0 && (hereT - thisT) * (nextT - thisT) < 0

               if lower(thisInputLabel) == lower("ErgodicSigma")
                  hereLnY = getSigmaCompErg(hereF, inMagnitude, inRrup);
                  nextLnY = getSigmaCompErg(nextF, inMagnitude, inRrup);
               elseif lower(thisInputLabel) == lower("SingleStationSigma")
                  hereLnY = getSigmaCompSS(hereF, inMagnitude, inRrup);
                  nextLnY = getSigmaCompSS(nextF, inMagnitude, inRrup);
               else
                  hereLnY = getLnY(hereF, inMagnitude, inRrup);
                  nextLnY = getLnY(nextF, inMagnitude, inRrup);
               end
               thisLnT = log(thisT);
               hereLnT = log(hereT);
               nextLnT = log(nextT);
               iLnY = hereLnY + (thisLnT - hereLnT) * (nextLnY - hereLnY) / (nextLnT - hereLnT);
               iiT = length(Flist) + 1;
            end
         end
      end
   end


   if lower(thisInputLabel) == lower("MedianPSA") && iLnY > -100
      iNGAEastGMM = exp(iLnY);
   else
      iNGAEastGMM = iLnY;
   end



end




function LnY = getLnY(thisFreq, thisMin, thisRrupin)
   % code developed and written by Silvia Mazzoni
   %      smazzoni@ucla.edu, March 2020
   inMagnitude = thisMin;
   inRrup = thisRrupin;

            % this algorithm assumes thisFreq is one of the frequencies listed,
            % if not, you need to write a wrapper that interpolates
   LnY = -999;
   % define these internally
   MagnitudeInterpolationType = "Linear";
   RrupInterpolationType = "Logarithmic";
```

```
medianLnYrefFrequencyLabelColumn = getmedianLnYrefFrequencyLabelColumn;
medianLnYrefFrequencyValueColumn = getmedianLnYrefFrequencyValueColumn;
medianLnYrefMomentColumn = getmedianLnYrefMomentColumn;
MedianLnYrefRrupRow = getMedianLnYrefRrupRow;
MedianLnYrefDataTable = getMedianLnYrefDataTable;



if isstring(thisFreq)
   if lower(thisFreq) == lower("PGA")
      thisFreq=-1;
   elseif lower(thisFreq) == lower("PGV")
      thisFreq=-2;
   end
end
medianLnYrefFrequencyColumn = medianLnYrefFrequencyValueColumn;

Mbelow = -888;
Mabove = 888;
Nrow = length(medianLnYrefMomentColumn);
NcolR = length(MedianLnYrefRrupRow);

for iRow = 1 : Nrow
   Fcheck = -1;
   hereFreq = medianLnYrefFrequencyColumn(iRow, 1);

   if abs(hereFreq - thisFreq) < 0.000001
      hereM = medianLnYrefMomentColumn(iRow, 1);
      if abs(hereM - inMagnitude) < 0.001

         for iColR = 1:NcolR
            hereRrup = MedianLnYrefRrupRow(1, iColR);
            if abs(hereRrup - inRrup) < 0.000001
               LnY = MedianLnYrefDataTable(iRow, iColR) ; %' nailed it!
               return
            else
               if iColR == NcolR
                  LnY = -888;
                  return
               end


               nextRrup = MedianLnYrefRrupRow(1, iColR + 1);
               if (inRrup - hereRrup) * (inRrup - nextRrup) < 0
                  if lower(RrupInterpolationType) == lower("Logarithmic")
                     if inRrup == 0
                        inRrup = 0.0000001;
                     end
                     if hereRrup == 0
                        hereRrup = 0.0000001;
                     end
                     if nextRrup == 0
                        nextRrup = 0.0000001;
                     end
                     hereRrup = log(hereRrup);
                     nextRrup = log(nextRrup);
                     inRrup = log(inRrup);
```

```
                end
                Y0 = MedianLnYrefDataTable(iRow, iColR);
                Y1 = MedianLnYrefDataTable(iRow, iColR + 1);
                LnY = Y0 + (Y1 - Y0) / (nextRrup - hereRrup) * (inRrup - hereRrup);
                return
              end
            end
          end
        else
          if hereM > Mbelow && hereM < inMagnitude
            Mbelow = hereM;
            Ibelow = iRow;
          end
          if hereM < Mabove && hereM > inMagnitude
            Mabove = hereM;
            Iabove = iRow;
          end
        end

    end



 end; % end for iRow



for iColR = 1 : NcolR
    hereRrup = MedianLnYrefRrupRow(1, iColR);
    if abs(hereRrup - inRrup) < 0.000001
        Z0 = MedianLnYrefDataTable(Ibelow, iColR) ; % nailed it!
        Z1 = MedianLnYrefDataTable(Iabove, iColR) ; % nailed it!
        iColR = NcolR + 1;
    else
        if iColR < NcolR
          nextRrup = MedianLnYrefRrupRow(1, iColR + 1);
          if (inRrup - hereRrup) * (inRrup - nextRrup) < 0
            if lower(RrupInterpolationType) == lower("logarithmic")
              if inRrup == 0
                inRrup = 0.1;
              end
              if hereRrup == 0
                hereRrup = 0.1;
              end
              if nextRrup == 0
                nextRrup = 0.1;
              end
              hereRrup = log(hereRrup);
              nextRrup = log(nextRrup);
              inRrup = log(inRrup);
            end
            iRow = Ibelow;
              Y0 = MedianLnYrefDataTable(iRow, iColR);
              Y1 = MedianLnYrefDataTable(iRow, iColR + 1);
              Z0 = Y0 + (inRrup - hereRrup) * (Y1 - Y0) / (nextRrup - hereRrup);
            iRow = Iabove;
              Y0 = MedianLnYrefDataTable(iRow, iColR);
```

```
                Y1 = MedianLnYrefDataTable(iRow, iColR + 1);
                Z1 = Y0 + (inRrup - hereRrup) * (Y1 - Y0) / (nextRrup - hereRrup);
            end
          end

      end
  end
   % interpolate on Moment
      if lower(MagnitudeInterpolationType) == lower("logarithmic")
         Mbelow = log(Mbelow);
         Mabove = log(Mabove);
         inMagnitude = log(inMagnitude);
      end
      LnY = Z0 + (inMagnitude - Mbelow) * (Z1 - Z0) / (Mabove - Mbelow);









end


function medianLnYrefFrequencyValueColumn = getmedianLnYrefFrequencyValueColumn();
   % code developed and written by Silvia Mazzoni
   %      smazzoni@ucla.edu, March 2020
   medianLnYrefFrequencyValueColumn =
[0.1;0.1;0.1;0.1;0.1;0.1;0.1;0.1;0.1;0.1;0.1;0.133;0.133;0.133;0.133;0.133;0.133;0.133;0.133;0.133;0.133;0.2;0.2;0.2;0.
2;0.2;0.2;0.2;0.2;0.2;0.2;0.2;0.25;0.25;0.25;0.25;0.25;0.25;0.25;0.25;0.25;0.25;0.25;0.333;0.333;0.333;0.333;0.333;0.333;0.3
33;0.333;0.333;0.333;0.333;0.5;0.5;0.5;0.5;0.5;0.5;0.5;0.5;0.5;0.5;0.5;0.667;0.667;0.667;0.667;0.667;0.667;0.667;0.667;0.66
7;0.667;0.667;1;1;1;1;1;1;1;1;1;1;1.333;1.333;1.333;1.333;1.333;1.333;1.333;1.333;1.333;1.333;1.333;2;2;2;2;2;2;2;2;2;2;2
;2.5;2.5;2.5;2.5;2.5;2.5;2.5;2.5;2.5;2.5;2.5;3.333;3.333;3.333;3.333;3.333;3.333;3.333;3.333;3.333;3.333;3.333;4;4;4;4;4;4;
4;4;4;4;5;5;5;5;5;5;5;5;5;5;6.667;6.667;6.667;6.667;6.667;6.667;6.667;6.667;6.667;6.667;6.667;10;10;10;10;10;10;10;10;1
0;10;10;13.333;13.333;13.333;13.333;13.333;13.333;13.333;13.333;13.333;13.333;13.333;20;20;20;20;20;20;20;20;20;20;2
0;25;25;25;25;25;25;25;25;25;25;25;33.333;33.333;33.333;33.333;33.333;33.333;33.333;33.333;33.333;33.333;33.333;33.333;40;4
0;40;40;40;40;40;40;40;40;40;50;50;50;50;50;50;50;50;50;50;100;100;100;100;100;100;100;100;100;100;100;-1;-1;-1;-1;-
1;-1;-1;-1;-1;-1;-1;-2;-2;-2;-2;-2;-2;-2;-2;-2;-2;-2];
end
function medianLnYrefFrequencyLabelColumn = getmedianLnYrefFrequencyLabelColumn();
   % code developed and written by Silvia Mazzoni
   %      smazzoni@ucla.edu, March 2020
   medianLnYrefFrequencyLabelColumn =
["F0.100";"F0.100";"F0.100";"F0.100";"F0.100";"F0.100";"F0.100";"F0.100";"F0.100";"F0.100";"F0.133";"F0.133";"F
0.133";"F0.133";"F0.133";"F0.133";"F0.133";"F0.133";"F0.133";"F0.133";"F0.133";"F0.200";"F0.200";"F0.200";"F0.200";"F0.
200";"F0.200";"F0.200";"F0.200";"F0.200";"F0.200";"F0.250";"F0.250";"F0.250";"F0.250";"F0.250";"F0.250";"F0.25
0";"F0.250";"F0.250";"F0.250";"F0.250";"F0.333";"F0.333";"F0.333";"F0.333";"F0.333";"F0.333";"F0.333";"F0.333";"F0.333";
"F0.333";"F0.333";"F0.500";"F0.500";"F0.500";"F0.500";"F0.500";"F0.500";"F0.500";"F0.500";"F0.500";"F0.500";"F0.500";"F
0.667";"F0.667";"F0.667";"F0.667";"F0.667";"F0.667";"F0.667";"F0.667";"F0.667";"F0.667";"F0.667";"F1.000";"F1.000";"F1.
000";"F1.000";"F1.000";"F1.000";"F1.000";"F1.000";"F1.000";"F1.000";"F1.000";"F1.333";"F1.333";"F1.333";"F1.333";"F1.33
3";"F1.333";"F1.333";"F1.333";"F1.333";"F1.333";"F1.333";"F2.000";"F2.000";"F2.000";"F2.000";"F2.000";"F2.000";"F2.000";
"F2.000";"F2.000";"F2.000";"F2.000";"F2.500";"F2.500";"F2.500";"F2.500";"F2.500";"F2.500";"F2.500";"F2.500";"F2.500";"F
2.500";"F2.500";"F3.333";"F3.333";"F3.333";"F3.333";"F3.333";"F3.333";"F3.333";"F3.333";"F3.333";"F3.333";"F3.333";"F4.
000";"F4.000";"F4.000";"F4.000";"F4.000";"F4.000";"F4.000";"F4.000";"F4.000";"F4.000";"F5.000";"F5.000";"F5.00
0";"F5.000";"F5.000";"F5.000";"F5.000";"F5.000";"F5.000";"F5.000";"F6.667";"F6.667";"F6.667";"F6.667";"F6.667";
"F6.667";"F6.667";"F6.667";"F6.667";"F6.667";"F6.667";"F10.000";"F10.000";"F10.000";"F10.000";"F10.000";"F10.000";"F10
.000";"F10.000";"F10.000";"F10.000";"F13.333";"F13.333";"F13.333";"F13.333";"F13.333";"F13.333";"F13.333";"
F13.333";"F13.333";"F13.333";"F13.333";"F20.000";"F20.000";"F20.000";"F20.000";"F20.000";"F20.000";"F20.000
```

```
";"F20.000";"F20.000";"F20.000";"F25.000";"F25.000";"F25.000";"F25.000";"F25.000";"F25.000";"F25.000";"F25.
000";"F25.000";"F25.000";"F33.333";"F33.333";"F33.333";"F33.333";"F33.333";"F33.333";"F33.333";"F33.333";"F
33.333";"F33.333";"F40.000";"F40.000";"F40.000";"F40.000";"F40.000";"F40.000";"F40.000";"F40.000";"F40.000"
;"F40.000";"F50.000";"F50.000";"F50.000";"F50.000";"F50.000";"F50.000";"F50.000";"F50.000";"F50.000";"F50.0
00";"F100.000";"F100.000";"F100.000";"F100.000";"F100.000";"F100.000";"F100.000";"F100.000";"F100.000";"
F100.000";"PGA";"PGA";"PGA";"PGA";"PGA";"PGA";"PGA";"PGA";"PGA";"PGA";"PGA";"PGV";"PGV";"PGV";"PGV";"PGV";"PG
V";"PGV";"PGV";"PGV";"PGV";"PGV"];
end
function medianLnYrefMomentColumn = getmedianLnYrefMomentColumn();
  % code developed and written by Silvia Mazzoni
  %     smazzoni@ucla.edu, March 2020
  medianLnYrefMomentColumn =
[4;4.5;5;5.5;6;6.5;7;7.5;7.8;8;8.2;4;4.5;5;5.5;6;6.5;7;7.5;7.8;8;8.2;4;4.5;5;5.5;6;6.5;7;7.5;7.8;
8;8.2;4;4.5;5;5.5;6;6.5;7;7.5;7.8;8;8.2;4;4.5;5;5.5;6;6.5;7;7.5;7.8;8;8.2;4;4.5;5;5.5;6;6.5;7;7.5
;7.8;8;8.2;4;4.5;5;5.5;6;6.5;7;7.5;7.8;8;8.2;4;4.5;5;5.5;6;6.5;7;7.5;7.8;8;8.2;4;4.5;5;5.5;6;6.5;
7;7.5;7.8;8;8.2;4;4.5;5;5.5;6;6.5;7;7.5;7.8;8;8.2;4;4.5;5;5.5;6;6.5;7;7.5;7.8;8;8.2;4;4.5;5;5.5;6
;6.5;7;7.5;7.8;8;8.2;4;4.5;5;5.5;6;6.5;7;7.5;7.8;8;8.2;4;4.5;5;5.5;6;6.5;7;7.5;7.8;8;8.2;4;4.5;5;
5.5;6;6.5;7;7.5;7.8;8;8.2;4;4.5;5;5.5;6;6.5;7;7.5;7.8;8;8.2;4;4.5;5;5.5;6;6.5;7;7.5;7.8;8;8.2;4;4
.5;5;5.5;6;6.5;7;7.5;7.8;8;8.2;4;4.5;5;5.5;6;6.5;7;7.5;7.8;8;8.2];
end
function MedianLnYrefRrupRow = getMedianLnYrefRrupRow()
  % code developed and written by Silvia Mazzoni
  %     smazzoni@ucla.edu, March 2020
  MedianLnYrefRrupRow =
[0,1,5,10,15,20,25,30,40,50,60,70,80,90,100,110,120,130,140,150,175,200,250,300,350,400,450,500,600,700,800,1000,120
0,1500];
end
function MedianLnYrefDataTable = getMedianLnYrefDataTable()
  % code developed and written by Silvia Mazzoni
  %     smazzoni@ucla.edu, March 2020
    MedianLnYrefDataTable = [-9.605511837,-9.856291163,-11.00960426,-11.82957032,-12.37243119,-12.77947739,-
13.11741114,-13.38980611,-13.81525848,-14.14286981,-14.33636957,-14.46077937,-14.54637483,-14.61907323,-
14.68526388,-14.74619915,-14.80314538,-14.86447387,-14.92591871,-14.98726788,-15.13383919,-15.2648154,-
15.50453776,-15.71111627,-15.89439328,-16.06048595,-16.21874402,-16.36542119,-16.61035701,-16.85344312,-
17.07062066,-17.4924918,-17.87393458,-18.43088267;
    -8.632895422,-8.808995903,-9.741673551,-10.51024366,-11.02877457,-11.41642824,-11.73855713,-11.99988697,-
12.40649941,-12.7205702,-12.90340172,-13.01997491,-13.09870673,-13.16525683,-13.22570974,-13.28127306,-
13.33313003,-13.38924104,-13.44548257,-13.50189503,-13.63699049,-13.75774001,-13.97748607,-14.16657328,-
14.33428781,-14.48627938,-14.63131164,-14.76570622,-14.99235452,-15.21270808,-15.41772823,-15.80572415,-
16.15812022,-16.67464842;
    -7.628126741,-7.754042931,-8.470501031,-9.164651811,-9.650659581,-10.01663929,-10.31773552,-10.56548549,-
10.95379646,-11.2559549,-11.42998075,-11.54043378,-11.61361699,-11.67497706,-11.73064368,-11.78171109,-
11.82929366,-11.88086304,-11.93247565,-11.9844441,-12.10926094,-12.22088544,-12.42343116,-12.59744808,-
12.75244678,-12.89328261,-13.0279918,-13.15280403,-13.3613213,-13.56784311,-13.75631479,-14.1173078,-
14.44862775,-14.92937735;
    -6.634394627,-6.728558551,-7.239586836,-7.829807262,-8.271634746,-8.610992066,-8.890940953,-9.125179644,-
9.492758134,-9.779968814,-9.944571959,-10.04860333,-10.11610877,-10.17203308,-10.2227232,-10.26978155,-
10.31357445,-10.36131443,-10.40946022,-10.45818147,-10.57537956,-10.68012642,-10.86967543,-11.03221346,-
11.17567156,-11.30544243,-11.42885006,-11.54340737,-11.73694501,-11.9302876,-12.10562903,-12.44221981,-
12.74703027,-13.19477817;
    -5.685500404,-5.760246657,-6.120002438,-6.583382757,-6.96588858,-7.273349299,-7.529593949,-7.74824096,-
8.098394827,-8.374097089,-8.531679301,-8.630827331,-8.693808479,-8.745497831,-8.792300307,-8.836352187,-
8.877562813,-8.922175588,-8.967356245,-9.013205604,-9.124163229,-9.223936318,-9.402643941,-9.556046412,-
9.692161801,-9.815395627,-9.932163763,-10.04030563,-10.22386691,-10.4048243,-10.56802154,-10.88032707,-
11.16269267,-11.57727908;
    -4.800405449,-4.86070112,-5.123329674,-5.46830614,-5.785412651,-6.057208318,-6.292867269,-6.497538491,-
6.827830152,-7.089328065,-7.2399778,-7.334130278,-7.393014475,-7.441628015,-7.485504575,-7.52728048,-
7.566028691,-7.608293992,-7.651076235,-7.694656372,-7.800700833,-7.896617464,-8.068964995,-8.217012507,-
```

8.347518764,-8.465041457,-8.575885419,-8.6782273,-8.852031579,-9.020443496,-9.171549066,-9.461660097,-9.723000624,-10.0981226;

-3.988380738,-4.039620817,-4.246495375,-4.517990876,-4.777784017,-5.008947136,-5.212912818,-5.393401335,-5.693280571,-5.929968766,-6.069708187,-6.158511616,-6.214828093,-6.261609781,-6.30410199,-6.34540248,-6.383721986,-6.425122519,-6.466745331,-6.509138548,-6.611726057,-6.704344454,-6.871637014,-7.01510075,-7.141229057,-7.254704516,-7.361242227,-7.459421836,-7.625349018,-7.784705046,-7.92793548,-8.201553061,-8.444516759,-8.792133926;

-3.295383805,-3.337669033,-3.506119623,-3.726210342,-3.935592116,-4.126148476,-4.299410316,-4.455342252,-4.720158297,-4.926007881,-5.053726401,-5.134472259,-5.187351742,-5.231351473,-5.271402346,-5.311253123,-5.348243333,-5.388766359,-5.429898758,-5.471648582,-5.572246295,-5.662870458,-5.822877092,-5.960119597,-6.081574046,-6.191008105,-6.293356206,-6.387258822,-6.54795043,-6.70140648,-6.838945787,-7.098703411,-7.32756554,-7.652964321;

-2.947955959,-2.986949286,-3.139420371,-3.331206605,-3.515126577,-3.684383443,-3.840019759,-3.98127727,-4.227794294,-4.414797434,-4.53588013,-4.611174698,-4.66171024,-4.704343183,-4.74325841,-4.782554002,-4.819053724,-4.858942731,-4.899254131,-4.940125071,-5.038773023,-5.127347665,-5.284798174,-5.41992718,-5.539656689,-5.647301307,-5.747702709,-5.83975969,-5.996632983,-6.145850152,-6.278515354,-6.529527144,-6.752811794,-7.065091549;

-2.742464853,-2.781731342,-2.924817321,-3.097354212,-3.2639971,-3.420054732,-3.564478853,-3.696092182,-3.929902957,-4.102666099,-4.219155505,-4.290365389,-4.339194694,-4.380853712,-4.418964151,-4.457869461,-4.494028897,-4.533488089,-4.573261644,-4.613498275,-4.710824899,-4.797997005,-4.953807389,-5.087555659,-5.205729344,-5.312105359,-5.411298232,-5.502090051,-5.656076778,-5.802382996,-5.933492832,-6.179872946,-6.396066756,-6.701456202;

-2.548821372,-2.591209906,-2.727436134,-2.881958851,-3.029866353,-3.172836014,-3.306113162,-3.428180029,-3.647927895,-3.80767519,-3.919688701,-3.986740634,-4.03383475,-4.074563771,-4.11180852,-4.150315822,-4.186071696,-4.225115013,-4.264337111,-4.303910382,-4.399864408,-4.485534807,-4.639545622,-4.771477809,-4.888165149,-4.993171014,-5.090848231,-5.180131736,-5.331702141,-5.473790551,-5.601495481,-5.842794097,-6.052468481,-6.34469623;

-8.897621841,-9.14758561,-10.31341195,-11.14141252,-11.68873225,-12.09876227,-12.43951131,-12.7140415,-13.14271447,-13.47283683,-13.6671022,-13.79315589,-13.87955592,-13.95271894,-14.01938979,-14.0808104,-14.13822585,-14.19983327,-14.26165159,-14.3235169,-14.47124447,-14.60326967,-14.84525722,-15.05358232,-15.23811889,-15.40503785,-15.56438964,-15.71160556,-15.95631049,-16.19751958,-16.411736,-16.82269953,-17.18915805,-17.71674244;

-7.931251855,-8.105834198,-9.046826291,-9.822915162,-10.34593868,-10.73675096,-11.06185351,-11.32553022,-11.73571261,-12.05268098,-12.23665483,-12.3551553,-12.43496345,-12.50225779,-12.56346343,-12.6197804,-12.67237152,-12.7290385,-12.78591865,-12.8431071,-12.97999621,-13.10242096,-13.3255822,-13.51753607,-13.68760995,-13.84151461,-13.9885201,-14.12436904,-14.35261216,-14.57325016,-14.77716015,-15.15849653,-15.50059273,-15.99514499;

-6.944350603,-7.068535156,-7.789627709,-8.490417639,-8.980838461,-9.350049699,-9.654042698,-9.904209963,-10.29636793,-10.60177304,-10.77727133,-10.8899062,-10.96441632,-11.02676229,-11.08341678,-11.13546767,-11.18401263,-11.23637616,-11.28885304,-11.34181614,-11.4689678,-11.5827964,-11.78972697,-11.96755455,-12.12586445,-12.2695625,-12.40691639,-12.53390437,-12.74558666,-12.95373429,-13.14288241,-13.50093605,-13.82576778,-14.29119922;

-5.983421346,-6.075679227,-6.587392012,-7.182468308,-7.628319359,-7.970894525,-8.253751454,-8.490541134,-8.862221481,-9.152958202,-9.319360493,-9.42582882,-9.494904612,-9.551990505,-9.603840687,-9.652019383,-9.696913839,-9.745606034,-9.794774646,-9.844642328,-9.964566891,-10.07194076,-10.2666683,-10.43369761,-10.58113158,-10.71443663,-10.8409662,-10.95824215,-11.1559747,-11.35226726,-11.5295351,-11.86612685,-12.16772075,-12.60627862;

-5.078344192,-5.151278175,-5.509402548,-5.975338635,-6.361127339,-6.671582631,-6.930597403,-7.151795892,-7.505988128,-7.785201741,-7.944650211,-8.046283077,-8.110888271,-8.163848363,-8.211917704,-8.257157195,-8.299365401,-8.345050668,-8.391392579,-8.438527153,-8.552522572,-8.655192879,-8.839533068,-8.997714034,-9.138167803,-9.265335715,-9.385634475,-9.496921106,-9.685666725,-9.870697345,-10.03720149,-10.35190099,-10.63387957,-11.04479903;

-4.244884535,-4.303478701,-4.563644731,-4.909291335,-5.22836513,-5.502203125,-5.740110772,-5.946994502,-6.281142556,-6.546092905,-6.698695874,-6.795371153,-6.855922859,-6.905558615,-6.950675218,-6.993752704,-7.033778921,-7.077195607,-7.121190177,-7.166106479,-7.275306501,-7.374240113,-7.552459929,-7.705393967,-7.840368088,-7.961979953,-8.076553162,-8.182283514,-8.361739784,-8.534992043,-8.690126642,-8.985078531,-9.248944501,-9.625232509;

-3.490658881,-3.540401805,-3.744679555,-4.01422142,-4.2744288,-4.50687013,-4.712509305,-4.894822076,-5.198201849,-5.438021075,-5.579711758,-5.670430773,-5.728339166,-5.776610434,-5.820560252,-5.863151109,-5.902737877,-5.945287767,-5.988112191,-6.031824866,-6.137524065,-6.233125915,-6.406262436,-6.554516073,-6.685060327,-6.802619761,-6.912936579,-7.014588043,-7.186405378,-7.351057654,-7.499059938,-7.779854819,-8.028001529,-8.381508605;

-2.860641988,-2.90076527,-3.065925102,-3.281585053,-3.489973512,-3.681041996,-3.855430168,-4.012773625,-4.280577851,-4.488590761,-4.617732081,-4.700960298,-4.755563256,-4.800976193,-4.84240872,-4.88343834,-4.921591091,-4.963198511,-5.00548302,-5.048494295,-5.152064971,-5.245543651,-5.411057034,-5.552774768,-5.678415328,-5.79176661,-5.897788982,-5.995242534,-6.162518518,-6.32200485,-6.465117235,-6.733601748,-6.969878205,-7.305617071;

-2.549595157,-2.586627207,-2.736048269,-2.924535974,-3.105150898,-3.273735943,-3.430169585,-3.572550911,-3.821551366,-4.009903409,-4.133048657,-4.210667313,-4.262838589,-4.306835571,-4.347081692,-4.387484332,-4.425077265,-4.465989157,-4.507390397,-4.549452364,-4.650910499,-4.742185213,-4.904907649,-5.044317291,-5.168079252,-5.279821001,-5.384154737,-5.479884154,-5.643370049,-5.79875414,-5.937180518,-6.197762557,-6.429817781,-6.754961394;

-2.367657402,-2.40521638,-2.545490625,-2.714859883,-2.87881912,-3.032778664,-3.176921283,-3.309459395,-3.545035525,-3.719374887,-3.837865557,-3.911273342,-3.961667341,-4.004651711,-4.044054152,-4.084012295,-4.121213645,-4.161651434,-4.202466059,-4.243842149,-4.343859824,-4.433616507,-4.594525127,-4.732413532,-4.854960842,-4.965442685,-5.068503607,-5.162927747,-5.323550159,-5.476095344,-5.613153624,-5.869786644,-6.095480175,-6.41559298;

-2.1967275,-2.237733268,-2.371370213,-2.523673169,-2.668295448,-2.809283173,-2.941396096,-3.062736819,-3.283592369,-3.445058337,-3.559004897,-3.628124832,-3.67671161,-3.718726005,-3.757222715,-3.796729017,-3.833470795,-3.87344503,-3.91365913,-3.954319112,-4.052838737,-4.140974352,-4.29989845,-4.436381475,-4.557322003,-4.666342033,-4.767818005,-4.860685521,-5.01895304,-5.167338941,-5.301180856,-5.553349778,-5.773374172,-6.081966288;

-8.000428233,-8.254022172,-9.38231526,-10.18547739,-10.72014458,-11.12237695,-11.45575189,-11.72493489,-12.14608114,-12.47083991,-12.66561093,-12.7908555,-12.87710808,-12.9511945,-13.01863063,-13.0806837,-13.13865176,-13.20110015,-13.26352275,-13.32570766,-13.47411534,-13.60643216,-13.8469096,-14.05313183,-14.23512572,-14.39910192,-14.55443399,-14.69770445,-14.93629989,-15.16925903,-15.37509457,-15.77202969,-16.12277894,-16.61996932;

-7.009179878,-7.190128358,-8.107662033,-8.861953409,-9.373206038,-9.756378557,-10.07435467,-10.33275945,-10.73577201,-11.04759415,-11.23232661,-11.35047342,-11.43062212,-11.49929838,-11.56173133,-11.61913918,-11.67274526,-11.73066915,-11.7885862,-11.84652045,-11.98515038,-12.10891284,-12.33271759,-12.52460734,-12.69409376,-12.84696078,-12.99205672,-13.12594759,-13.35161846,-13.56748638,-13.76601409,-14.13944498,-14.47163168,-14.94485269;

-6.036491777,-6.167803138,-6.877672931,-7.560409911,-8.040071986,-8.402054719,-8.699804266,-8.945121251,-9.330511834,-9.630895946,-9.807322191,-9.919925797,-9.995127427,-10.05922203,-10.11747459,-10.17099235,-10.22092233,-10.2748798,-10.32875725,-10.38283805,-10.51264096,-10.62871416,-10.83812857,-11.01763017,-11.17696478,-11.32115146,-11.45844038,-11.58517282,-11.79741906,-12.00396775,-12.19065124,-12.54585718,-12.86544271,-13.31767881;

-5.114334845,-5.214077576,-5.727437842,-6.310915635,-6.747865433,-7.083779236,-7.360894311,-7.592932485,-7.958067556,-8.243895568,-8.411146297,-8.517632032,-8.58751325,-8.646615306,-8.700340084,-8.750360166,-8.796998692,-8.847584356,-8.898442491,-8.949720262,-9.073004634,-9.183259578,-9.381757608,-9.551919607,-9.701871599,-9.837144013,-9.96513863,-10.08358068,-10.28460584,-10.48150477,-10.65893362,-10.99705234,-11.29877829,-11.73100132;

-4.275999162,-4.355744198,-4.722811908,-5.184630697,-5.564153986,-5.869043648,-6.123191803,-6.340069163,-6.688289907,-6.963003756,-7.123445905,-7.225255955,-7.290861215,-7.345932105,-7.395979178,-7.443280116,-7.487719434,-7.535413072,-7.58353123,-7.632165085,-7.749817671,-7.855760401,-8.044963428,-8.207570625,-8.351646438,-8.481799572,-8.604607071,-8.718092189,-8.91171254,-9.099128305,-9.266937845,-9.586924823,-9.872824944,-10.28274254;

-3.53209544,-3.596805693,-3.868552942,-4.21598426,-4.532533459,-4.803187159,-5.037434784,-5.240767721,-5.569576323,-5.830416757,-5.983865609,-6.080626151,-6.142143229,-6.194264919,-6.24142729,-6.286482045,-6.328401007,-6.373785292,-6.419580944,-6.466017121,-6.578980333,-6.681321108,-6.864751994,-7.02271219,-7.161954985,-7.287221554,-7.404948564,-7.513535113,-7.699140281,-7.87600385,-8.034074579,-8.336367624,-8.606490289,-8.987340396;

-2.87835895,-2.933484945,-3.149853259,-3.426902469,-3.688498961,-3.919912863,-4.123469947,-4.303261283,-4.60203363,-4.838475656,-4.980806771,-5.072247143,-5.131056001,-5.180908135,-5.226374951,-5.270810488,-5.31216861,-5.356546726,-5.401075006,-5.446230013,-5.55553629,-5.654428608,-5.832549652,-5.9858029,-

6.120662959,-6.241972805,-6.355582719,-6.460238236,-6.638674232,-6.807511657,-6.958836425,-7.247882553,-7.504046236,-7.866156562;

-2.34713038,-2.393304265,-2.571337583,-2.799133644,-3.012140429,-3.204230031,-3.377885859,-3.533674304,-3.797897739,-4.004403519,-4.134605406,-4.217386311,-4.272450741,-4.319249591,-4.36201977,-4.404814205,-4.44465096,-4.487889807,-4.531654608,-4.575906107,-4.682614397,-4.778978836,-4.949174831,-5.095696882,-5.225438945,-5.342358937,-5.451534501,-5.551733357,-5.724494397,-5.887480789,-6.033666669,-6.311053266,-6.556608882,-6.904217259;

-2.095892595,-2.138486208,-2.300098697,-2.499254343,-2.688152984,-2.859870439,-3.016405954,-3.15793844,-3.404214723,-3.592654851,-3.715756455,-3.792925303,-3.84552223,-3.890733301,-3.932151043,-3.974214432,-4.013387933,-4.055812984,-4.098594,-4.141813657,-4.246214851,-4.340165627,-4.507076013,-4.650870092,-4.77836114,-4.892950254,-4.999723504,-5.097770911,-5.266367779,-5.425136163,-5.566740678,-5.836562027,-6.078185756,-6.417149905;

-1.952569688,-1.994973541,-2.146706817,-2.326599289,-2.498275095,-2.657621053,-2.80360153,-2.935767999,-3.169754522,-3.344240653,-3.462640579,-3.5356056,-3.586416305,-3.630495733,-3.67095119,-3.712494964,-3.751199581,-3.793064968,-3.835188363,-3.877662626,-3.980470925,-4.07275487,-4.237486457,-4.37943769,-4.504828201,-4.617804663,-4.723124005,-4.819705992,-4.985170211,-5.140991434,-5.280736061,-5.546879693,-5.782791171,-6.117705512;

-1.819253853,-1.864098618,-2.008389773,-2.169871602,-2.323525642,-2.469989202,-2.605379088,-2.728869059,-2.949149936,-3.11097025,-3.224771989,-3.293452274,-3.342438979,-3.385416186,-3.42484322,-3.465853002,-3.50401999,-3.545331593,-3.586781462,-3.628474765,-3.729626245,-3.820140997,-3.982520872,-4.12199512,-4.245506172,-4.356828516,-4.460393681,-4.55527904,-4.717954309,-4.869683745,-5.006282953,-5.267992586,-5.498924108,-5.824367739;

-7.483444855,-7.736120033,-8.856411112,-9.657132495,-10.19158371,-10.59404486,-10.92750432,-11.19694377,-11.61877811,-11.94427141,-12.14046654,-12.26572315,-12.35221457,-12.42666519,-12.49445172,-12.55683726,-12.61512427,-12.67812627,-12.74109489,-12.80369445,-12.9530696,-13.08616952,-13.32759251,-13.53440997,-13.71671417,-13.88075357,-14.0351869,-14.1775882,-14.41531004,-14.64632696,-14.85036457,-15.24180365,-15.58635329,-16.07227505;

-6.485008511,-6.665764921,-7.578271672,-8.33057812,-8.841600875,-9.224965115,-9.54303616,-9.801701616,-10.20548637,-10.51813329,-10.70439758,-10.82275789,-10.90335157,-10.97259611,-11.03558212,-11.09352461,-11.14765207,-11.20632035,-11.26497755,-11.3235181,-11.46359274,-11.58861683,-11.81429418,-12.00767124,-12.17832921,-12.33209934,-12.47705016,-12.6108259,-12.83697584,-13.05248705,-13.25036775,-13.6212762,-13.95007544,-14.41665262;

-5.521745314,-5.653163738,-6.360389604,-7.041695546,-7.521177294,-7.883356539,-8.18130123,-8.426921677,-8.813132934,-9.114375909,-9.292388895,-9.405350742,-9.481148589,-9.545965351,-9.604926728,-9.659135162,-9.709742503,-9.764593344,-9.819366801,-9.874210173,-10.00584721,-10.12357763,-10.33561664,-10.51732834,-10.67853069,-10.82429859,-10.96204305,-11.08925106,-11.30316965,-11.51046165,-11.69772395,-12.05278446,-12.37128414,-12.82082357;

-4.623399694,-4.723360302,-5.236404913,-5.819387358,-6.256336656,-6.592464921,-6.869797947,-7.102112365,-7.468051658,-7.754733146,-7.923552308,-8.030467303,-8.10101256,-8.160727759,-8.215077768,-8.265736265,-8.313015173,-8.364459181,-8.416188632,-8.468182441,-8.593312742,-8.705428723,-8.907063368,-9.079465997,-9.231436664,-9.368518512,-9.498314732,-9.618507524,-9.822673586,-10.02164444,-10.20066889,-10.54063513,-10.84361507,-11.27663829;

-3.822303535,-3.902212905,-4.270464481,-4.732832064,-5.112685539,-5.417893162,-5.67234641,-5.889525431,-6.23814369,-6.513326166,-6.674987301,-6.776998131,-6.843102828,-6.89884488,-6.949574022,-6.997584149,-7.042474671,-7.09113728,-7.140261028,-7.189771813,-7.309574495,-7.41752196,-7.610246305,-7.775550746,-7.922070565,-8.054441329,-8.179304149,-8.294705705,-8.492472187,-8.68289989,-8.853770082,-9.177408042,-9.466423893,-9.880005299;

-3.125341102,-3.190225451,-3.463874424,-3.812713177,-4.12979201,-4.400544505,-4.634810584,-4.838172395,-5.167300337,-5.428604236,-5.583171646,-5.680106377,-5.742112562,-5.794653756,-5.842428815,-5.888277015,-5.930984562,-5.97735558,-6.024145795,-6.071450957,-6.186558848,-6.290918183,-6.477966725,-6.638826592,-6.780727926,-6.908439727,-7.028453564,-7.139211584,-7.32939647,-7.509867857,-7.67144039,-7.979250902,-8.254308678,-8.641789083;

-2.523974133,-2.57925535,-2.797647338,-3.075869195,-3.33811763,-3.569841562,-3.77354373,-3.953408497,-4.252389361,-4.489214765,-4.632533144,-4.723799424,-4.782994006,-4.833494994,-4.879622593,-4.924765644,-4.96683246,-5.012119818,-5.05757037,-5.103529399,-5.214834822,-5.315627021,-5.497150933,-5.653213716,-5.790677058,-5.914405771,-6.030308946,-6.137153301,-6.3202768,-6.492934869,-6.648072391,-6.943733888,-7.206271703,-7.577107298;

-2.045865442,-2.091974631,-2.271703626,-2.500526907,-2.714350894,-2.906955019,-3.080840333,-3.236731115,-3.501094632,-3.707835667,-3.838703916,-3.921590148,-3.977035417,-4.024360131,-4.06767379,-4.111073549,-4.15151936,-4.195556632,-4.240122529,-4.28506941,-4.393519088,-4.491555256,-4.664818625,-4.813934984,-4.946086594,-5.065267371,-5.17660704,-5.278948646,-5.4563864,-5.623327592,-5.773462647,-6.057890316,-6.310599245,-6.668949337;

-1.825145877,-1.867715421,-2.030970133,-2.231706662,-2.421031678,-2.59313698,-2.749974406,-2.891662347,-3.137988533,-3.326633772,-3.45055215,-3.527785945,-3.580701692,-3.626361188,-3.668247194,-3.71084846,-3.750566314,-3.793725472,-3.837242544,-3.881099964,-3.987103389,-4.082584969,-4.252284515,-4.398473225,-4.528201114,-4.644999363,-4.753928046,-4.854073741,-5.027166322,-5.189850755,-5.335471191,-5.612630789,-5.861645136,-6.212440042;

-1.701206163,-1.743538618,-1.896821042,-2.078396272,-2.250899979,-2.410433502,-2.55655042,-2.688912908,-2.922823405,-3.097756969,-3.2169065,-3.289912362,-3.341002025,-3.385475702,-3.426347372,-3.468382757,-3.507585797,-3.550141089,-3.592954728,-3.636027225,-3.740339853,-3.834055712,-4.001383307,-4.14558458,-4.273215438,-4.388315392,-4.49569223,-4.59429462,-4.764150071,-4.923844926,-5.067483609,-5.341039442,-5.584787314,-5.932028047;

-1.586386106,-1.631056157,-1.776787832,-1.940204197,-2.094833277,-2.241698729,-2.377161224,-2.500571819,-2.720671071,-2.883170918,-2.997656325,-3.066352696,-3.115577436,-3.158893185,-3.198683859,-3.24013748,-3.278758381,-3.320714183,-3.362808487,-3.405060861,-3.507618538,-3.59946947,-3.764252562,-3.905968724,-4.031587769,-4.144924394,-4.25045366,-4.347290899,-4.514197852,-4.66982571,-4.810341856,-5.079588461,-5.318753399,-5.657749222;

-6.804933471,-7.057199031,-8.230337515,-9.059175986,-9.606385763,-10.01644293,-10.3574822,-10.63234635,-11.06177429,-11.39285578,-11.58768681,-11.71752982,-11.80741501,-11.8837251,-11.95334689,-12.01753703,-12.07755265,-12.14102296,-12.20447371,-12.2680492,-12.41972843,-12.55545386,-12.80424606,-13.01791363,-13.20663615,-13.37674672,-13.54150884,-13.69273152,-13.94465927,-14.18732892,-14.401459,-14.80717421,-15.1628702,-15.65970733;

-5.808277059,-5.98502469,-6.931591413,-7.70915796,-8.233174507,-8.624953629,-8.951227483,-9.216050905,-9.628467364,-9.947770298,-10.13361597,-10.25702847,-10.34141722,-10.41293292,-10.47813345,-10.53824569,-10.59444709,-10.65399454,-10.71349111,-10.77335325,-10.91654879,-11.04492754,-11.27930833,-11.48085621,-11.65917421,-11.82020362,-11.9766203,-12.12034383,-12.36273774,-12.59257836,-12.80241936,-13.19203264,-13.53651179,-14.02041001;

-4.857568512,-4.98415465,-5.709615178,-6.412249191,-6.904505476,-7.275474447,-7.581349723,-7.833340308,-8.229058232,-8.538010511,-8.716526804,-8.835016444,-8.915032297,-8.982495149,-9.044015274,-9.100716667,-9.153708608,-9.209797905,-9.265722523,-9.322173066,-9.457627055,-9.579312303,-9.801204412,-9.99221561,-10.16221536,-10.31635865,-10.46652199,-10.60462095,-10.83677521,-11.05953773,-11.26077619,-11.63845827,-11.97624821,-12.4482096;

-3.995665957,-4.090582392,-4.605380029,-5.202116735,-5.650307334,-5.995193602,-6.280543351,-6.519718007,-6.895973653,-7.191191072,-7.36151743,-7.47454005,-7.549859046,-7.613473059,-7.671505454,-7.725587203,-7.776112682,-7.829675916,-7.883375524,-7.937883056,-8.068535115,-8.185459545,-8.398091018,-8.582927277,-8.746228935,-8.893857409,-9.032730556,-9.160882885,-9.382088669,-9.596054223,-9.790206139,-10.15550765,-10.48014335,-10.9404269;

-3.240369634,-3.316049011,-3.676488685,-4.144339945,-4.532549882,-4.845461523,-5.107158582,-5.330956597,-5.691429919,-5.976741101,-6.141614554,-6.250965726,-6.322838078,-6.382589176,-6.437114414,-6.48855551,-6.537601281,-6.588307826,-6.639241079,-6.690984713,-6.816020613,-6.929125856,-7.132951198,-7.310188418,-7.467839798,-7.610713639,-7.745368602,-7.869906374,-8.083854963,-8.289430965,-8.473833687,-8.824253459,-9.136931891,-9.583006398;

-2.594598692,-2.655657237,-2.917524432,-3.265276016,-3.587352433,-3.864818757,-4.106321379,-4.316606509,-4.657129255,-4.92805346,-5.086157948,-5.190470429,-5.25823519,-5.315436788,-5.367205989,-5.416227943,-5.461970631,-5.510408611,-5.559129684,-5.608780392,-5.72936148,-5.83905544,-6.037248849,-6.209462385,-6.362000831,-6.49979547,-6.629332658,-6.749060886,-6.955081597,-7.151022388,-7.327176641,-7.661419983,-7.961110547,-8.385395358;

-2.048624951,-2.100602946,-2.306463624,-2.579722322,-2.843320995,-3.079137761,-3.288249552,-3.473964631,-3.783776626,-4.029607858,-4.17666386,-4.275617516,-4.340545823,-4.395096687,-4.445046624,-4.493457927,-4.538642601,-4.58610567,-4.63356574,-4.68192223,-4.798782118,-4.904909141,-5.097654065,-5.264652308,-5.41241018,-5.545960137,-5.671231783,-5.786948937,-5.985691725,-6.173986225,-6.344011295,-6.667036194,-6.955464777,-7.367059125;

-1.630654237,-1.673354104,-1.840713797,-2.060402936,-2.271960079,-2.466032466,-2.64360022,-2.804172068,-3.078225077,-3.292459074,-3.427132053,-3.516976957,-3.577831188,-3.62914837,-3.676226519,-3.722745644,-3.766175648,-3.812395209,-3.859049405,-3.90644299,-4.020528074,-4.123894893,-4.307934644,-4.467272038,-

4.609191442,-4.737779158,-4.858182366,-4.96915508,-5.162392507,-5.345256905,-5.510972801,-5.824724461,-6.10579594,-6.508857808;

-1.442612547,-1.481870783,-1.633172734,-1.823788961,-2.008566576,-2.180477532,-2.33982044,-2.48517792,-2.740269327,-2.93469301,-3.062340998,-3.146212811,-3.204281341,-3.253853587,-3.299430431,-3.345019359,-3.38759992,-3.432839415,-3.47835485,-3.524541574,-3.635909226,-3.736568025,-3.916954547,-4.073117445,-4.212431813,-4.338539256,-4.45645501,-4.565236744,-4.754261677,-4.933025716,-5.094424867,-5.401895291,-5.680190158,-6.077131603;

-1.339071183,-1.378717614,-1.520759789,-1.691833135,-1.858759285,-2.01653331,-2.163998877,-2.29932694,-2.541080518,-2.720674206,-2.843477322,-2.922876796,-2.978917508,-3.027244051,-3.071754295,-3.116684393,-3.158666584,-3.203230448,-3.247974192,-3.293293041,-3.402787896,-3.501563949,-3.679511279,-3.833540582,-3.970787897,-4.095192845,-4.211540149,-4.31880054,-4.504722325,-4.68060072,-4.84061908,-5.144862446,-5.418360469,-5.813285581;

-1.243247002,-1.286220652,-1.421430505,-1.574506199,-1.721729453,-1.865867299,-2.001399337,-2.126210986,-2.353219038,-2.519216292,-2.637273535,-2.712115645,-2.766091964,-2.813213025,-2.856592903,-2.900853368,-2.942163923,-2.986059882,-3.030016885,-3.074433824,-3.18199157,-3.278775312,-3.454103326,-3.605777069,-3.740960683,-3.863574366,-3.978072007,-4.083577381,-4.266806163,-4.438784459,-4.596007341,-4.896466666,-5.166312277,-5.553882266;

-5.891045074,-6.141113033,-7.286879973,-8.105952327,-8.651362887,-9.061293275,-9.401695159,-9.67654538,-10.10665905,-10.43871796,-10.63716055,-10.76611577,-10.85540108,-10.93169193,-11.00133224,-11.0655561,-11.125626,-11.19001467,-11.25440639,-11.31866779,-11.47201708,-11.60904578,-11.85896765,-12.07343387,-12.26272849,-12.43323004,-12.59519243,-12.74439123,-12.99763931,-13.23913925,-13.45446376,-13.86051229,-14.2194283,-14.72399225;

-4.86751949,-5.044280053,-5.973706627,-6.743059218,-7.265062983,-7.656352733,-7.981817678,-8.24643088,-8.659413784,-8.9796052,-9.169043447,-9.291882527,-9.376017605,-9.447857888,-9.513420329,-9.573913571,-9.630525995,-9.691306759,-9.752074514,-9.812946361,-9.958633362,-10.08916596,-10.32631476,-10.53022936,-10.71063022,-10.87353218,-11.02828498,-11.17114984,-11.41659146,-11.648147,-11.8602226,-12.25485309,-12.6068325,-13.1053422;

-3.935495458,-4.062692212,-4.779009729,-5.475277304,-5.965534985,-6.335853034,-6.641103347,-6.892869627,-7.288987236,-7.598608825,-7.780573627,-7.898663737,-7.978642372,-8.04665886,-8.10877942,-8.166106382,-8.219756056,-8.277299993,-8.334742311,-8.392454645,-8.53103888,-8.655551484,-8.881457476,-9.076015701,-9.249215986,-9.406309535,-9.555551031,-9.693523523,-9.930293502,-10.15578404,-10.36108535,-10.7468837,-11.09457175,-11.58629762;

-3.126208015,-3.221802655,-3.735286465,-4.328915886,-4.775622081,-5.119774125,-5.40444312,-5.643153469,-6.019480684,-6.315077737,-6.488503091,-6.601090498,-6.676343612,-6.739737251,-6.797705754,-6.851882369,-6.902600349,-6.957166748,-7.011975699,-7.067246247,-7.200318251,-7.320165979,-7.537468293,-7.724516995,-7.890193476,-8.040281499,-8.182378007,-8.314275821,-8.543185282,-8.762620253,-8.962051878,-9.337751812,-9.675502793,-10.15968591;

-2.452950845,-2.528860512,-2.892925098,-3.36130329,-3.749119434,-4.061585954,-4.322838544,-4.54619034,-4.904875193,-5.188794712,-5.355150883,-5.462904213,-5.533805646,-5.593357148,-5.64783783,-5.699452268,-5.747670741,-5.799636638,-5.852042626,-5.905021323,-6.033236835,-6.149351605,-6.358383232,-6.538630446,-6.69933616,-6.845282545,-6.983135722,-7.111006112,-7.334582678,-7.54727918,-7.740660793,-8.103836442,-8.431746832,-8.90422019;

-1.90648546,-1.967660781,-2.235485521,-2.586795256,-2.908995294,-3.184948366,-3.42475028,-3.63342368,-3.971799226,-4.241283916,-4.40046414,-4.503009653,-4.56970006,-4.625551263,-4.676908024,-4.726403704,-4.772686458,-4.822394539,-4.872479737,-4.923266042,-5.04681752,-5.159369137,-5.36268128,-5.538194278,-5.694089411,-5.835281557,-5.968321358,-6.091740575,-6.307658041,-6.51158345,-6.696747106,-7.047480986,-7.365361758,-7.819521469;

-1.465377096,-1.517423008,-1.729677406,-2.005098778,-2.269142642,-2.504302086,-2.712253936,-2.896603996,-3.204030615,-3.448291853,-3.596056975,-3.691796056,-3.755310021,-3.809620895,-3.859475869,-3.908091995,-3.953568722,-4.002054034,-4.050652716,-4.099938566,-4.219300869,-4.327900853,-4.525029333,-4.695006729,-4.845868619,-4.982607103,-5.111238168,-5.230496698,-5.438895592,-5.635179352,-5.814456452,-6.155752059,-6.464477994,-6.907401845;

-1.147740744,-1.189588412,-1.36201791,-1.58302078,-1.795626934,-1.989926448,-2.166781928,-2.326234933,-2.597936775,-2.810214599,-2.944512363,-3.032553543,-3.092235814,-3.143023967,-3.189732936,-3.236234971,-3.279739649,-3.326711319,-3.374205322,-3.422255793,-3.538202432,-3.643482809,-3.831153376,-3.993097542,-4.137732415,-4.269157261,-4.392600661,-4.507081222,-4.710122532,-4.901281228,-5.076284651,-5.408007256,-5.709560892,-6.145352161;

-1.01389074,-1.052520586,-1.208816651,-1.403311172,-1.587344237,-1.75891603,-1.917914693,-2.062446089,-2.315003478,-2.507472482,-2.635681929,-2.717690337,-2.774510078,-2.823410237,-2.868477021,-2.913934794,-2.956478526,-3.002354185,-3.048594837,-3.095338525,-3.208325954,-3.310625594,-3.494055628,-3.652478719,-3.794190509,-3.923287574,-4.044519663,-4.156858232,-4.355172803,-4.54203329,-4.712710269,-5.038337841,-5.336545055,-5.766898712;

-0.943663827,-0.982559857,-1.129342374,-1.304839078,-1.47282682,-1.629328788,-1.775700889,-1.910423347,-2.149208883,-2.328040273,-2.451286499,-2.528795442,-2.583553493,-2.631118358,-2.675029806,-2.719758307,-2.761631289,-2.806761029,-2.852156976,-2.897973475,-3.008932128,-3.109173793,-3.289791175,-3.445844549,-3.585881578,-3.71318754,-3.832662367,-3.943321758,-4.138296521,-4.322118173,-4.490914785,-4.813129071,-5.107054955,-5.535339665;

-0.879107685,-0.92098147,-1.060569544,-1.219286383,-1.368155735,-1.51199853,-1.646133133,-1.769066121,-1.992793907,-2.159139798,-2.277531371,-2.350458853,-2.403122443,-2.449392315,-2.492089927,-2.536080514,-2.577219812,-2.621612184,-2.666155277,-2.711016823,-2.819895062,-2.917995794,-3.095645992,-3.249783653,-3.387521007,-3.512864593,-3.630334058,-3.739126114,-3.931013151,-4.111052832,-4.277064734,-4.595405323,-4.886282563,-5.309240359;

-5.249894183,-5.500613583,-6.658711798,-7.48292908,-8.030168805,-8.441022672,-8.78265766,-9.058391665,-9.48971138,-9.822744108,-10.02065359,-10.15185078,-10.24378135,-10.32236215,-10.39416696,-10.4604392,-10.5224461,-10.58815171,-10.65363049,-10.71905026,-10.87529298,-11.01530757,-11.27172527,-11.49221029,-11.68724804,-11.86329873,-12.03426507,-12.19164526,-12.46203111,-12.717363,-12.94703956,-13.38031188,-13.76763997,-14.31193754;

-4.221970301,-4.398704656,-5.336331391,-6.110423294,-6.634740094,-7.027513233,-7.354684493,-7.620620266,-8.035514477,-8.357307495,-8.546778171,-8.67226541,-8.759410394,-8.833891757,-8.901943139,-8.964800173,-9.023657688,-9.086074083,-9.148218064,-9.210526817,-9.359780463,-9.49391188,-9.738806813,-9.949941216,-10.13718811,-10.30670483,-10.47182906,-10.62411985,-10.88852093,-11.13656038,-11.36397222,-11.78948551,-12.17277406,-12.71501075;

-3.320315139,-3.447837357,-4.168727571,-4.869074806,-5.361935138,-5.73412677,-6.041292568,-6.294644685,-6.693261958,-7.005028701,-7.187516834,-7.308565035,-7.391828534,-7.462730049,-7.527566983,-7.587474643,-7.643575314,-7.702970303,-7.761992571,-7.821335479,-7.963963334,-8.092496854,-8.327040076,-8.529676554,-8.710535128,-8.875009471,-9.035743404,-9.184097218,-9.44165746,-9.684589926,-9.906845677,-10.32593374,-10.70646773,-11.24416906;

-2.569416819,-2.665872336,-3.180097291,-3.77616067,-4.22538026,-4.571613912,-4.858439118,-5.099008733,-5.478273983,-5.776405169,-5.950781721,-6.066561143,-6.145356296,-6.212762449,-6.274454823,-6.332113243,-6.386108344,-6.443303293,-6.500406126,-6.558124311,-6.696709145,-6.821139936,-7.047722203,-7.245788641,-7.42154166,-7.581090402,-7.73139459,-7.870713171,-8.119256072,-8.354945867,-8.572347165,-8.982782902,-9.355124438,-9.886730207;

-1.968247072,-2.045633395,-2.408461838,-2.87778634,-3.267514415,-3.58180503,-3.845010391,-4.07014576,-4.433560374,-4.721726869,-4.890627622,-5.002741755,-5.078136315,-5.141648142,-5.199811354,-5.254866957,-5.307576249,-5.361923592,-5.416251357,-5.471184982,-5.604224437,-5.725014101,-5.943283174,-6.134243596,-6.304860775,-6.460188284,-6.606975801,-6.743454706,-6.985199483,-7.213885312,-7.422485262,-7.820911004,-8.183994596,-8.70535033;

-1.496899348,-1.559686596,-1.825368629,-2.176488104,-2.500819852,-2.77995893,-3.022864481,-3.234342469,-3.577393463,-3.85077744,-4.012500635,-4.11922683,-4.19022465,-4.25107858,-4.306284696,-4.358672228,-4.407698782,-4.459590618,-4.511570013,-4.564278345,-4.692603472,-4.809771611,-5.022020647,-5.207666276,-5.373011144,-5.523218587,-5.66493234,-5.796786097,-6.030360403,-6.250363595,-6.451906431,-6.835591339,-7.187234621,-7.692593682;

-1.125630104,-1.179158537,-1.388987822,-1.666581618,-1.932946779,-2.170562929,-2.381125734,-2.568016694,-2.879934986,-3.127914886,-3.278084915,-3.379267437,-3.447036892,-3.504606434,-3.557511115,-3.608969059,-3.657134466,-3.707753356,-3.758199892,-3.809358535,-3.93333559,-4.046333004,-4.252021354,-4.431493512,-4.591293539,-4.736625902,-4.873621245,-5.001047326,-5.226557486,-5.438687146,-5.633796203,-6.006459899,-6.348086342,-6.84288241;

-0.866097862,-0.910404778,-1.08170278,-1.306066096,-1.520658243,-1.716659043,-1.895583003,-2.057230916,-2.333090562,-2.549485292,-2.686903903,-2.778318085,-2.841497919,-2.895414021,-2.945053603,-2.994275832,-3.040357232,-3.089374078,-3.138663025,-3.188534886,-3.308914533,-3.418358027,-3.614118646,-3.78488619,-3.937895071,-4.077444747,-4.208855152,-4.330920725,-4.549562663,-4.755446031,-4.945708109,-5.30926304,-5.644309305,-6.132966493;

-0.762290494,-0.802991706,-0.957919051,-1.152538404,-1.340827597,-1.514896851,-1.675579036,-1.821976604,-2.078724363,-2.275393186,-2.40526491,-2.490533054,-2.550682445,-2.602584302,-2.650466153,-2.698525974,-2.743537826,-2.791350049,-2.839295531,-2.887769657,-3.004972175,-3.111305194,-3.302653507,-3.469587756,-

3.619420865,-3.755900005,-3.884305001,-4.003876279,-4.217608533,-4.419179736,-4.605272992,-4.963207579,-5.294144972,-5.777246213;

-0.709527061,-0.750421201,-0.895844207,-1.070746696,-1.240914152,-1.401195691,-1.550266503,-1.686628464,-1.930084289,-2.111786615,-2.236642106,-2.317344383,-2.375322018,-2.425806723,-2.472459451,-2.519716061,-2.563987909,-2.610980495,-2.658025613,-2.705514762,-2.820553821,-2.924747462,-3.113187436,-3.277567658,-3.42485952,-3.559335923,-3.685960685,-3.803843371,-4.014209879,-4.212719119,-4.396924189,-4.75110465,-5.078025758,-5.55890076;

-0.660658285,-0.704567342,-0.842789932,-0.99911638,-1.149767484,-1.29634378,-1.433688876,-1.559963847,-1.788666895,-1.956745607,-2.076700915,-2.152752057,-2.20852328,-2.257639322,-2.303009876,-2.349457985,-2.392932148,-2.439123998,-2.48526405,-2.53174363,-2.64458011,-2.746556425,-2.931953316,-3.093484481,-3.238409374,-3.370938385,-3.495578638,-3.611619213,-3.81875031,-4.013793472,-4.195442886,-4.545605255,-4.870118258,-5.346927109;

-4.359877713,-4.610030428,-5.744234456,-6.557770756,-7.101730749,-7.511110406,-7.851049333,-8.125829321,-8.556191909,-8.888940176,-9.089398128,-9.221498415,-9.315071595,-9.39567485,-9.469409695,-9.537519186,-9.60127988,-9.669062285,-9.736483849,-9.803575267,-9.963941459,-10.10786593,-10.37122409,-10.59831516,-10.79988956,-10.98249199,-11.16017453,-11.3246398,-11.61734669,-11.88910845,-12.13923145,-12.61286731,-13.04664079,-13.66358744;

-3.343304382,-3.521635761,-4.444328605,-5.209306721,-5.730530637,-6.121751467,-6.447293502,-6.712252221,-7.126315455,-7.447887391,-7.639993927,-7.766695106,-7.85579986,-7.932611298,-8.002885189,-8.067873357,-8.128787178,-8.193548546,-8.257900554,-8.322132614,-8.476153315,-8.614835972,-8.867964475,-9.086934456,-9.281840018,-9.458997515,-9.631968245,-9.7924091,-10.0798373,-10.34747426,-10.59447455,-11.0634384,-11.49527533,-12.1127342;

-2.510730391,-2.640983301,-3.354085255,-4.047125366,-4.537169179,-4.907853704,-5.21373606,-5.466211617,-5.864019265,-6.175444769,-6.360477031,-6.482842221,-6.568197242,-6.641555118,-6.708743383,-6.770913547,-6.829200107,-6.891052433,-6.952422721,-7.013831348,-7.161594677,-7.295055571,-7.538566393,-7.749706642,-7.938808104,-8.111450173,-8.280666262,-8.437651313,-8.719268058,-8.981166555,-9.22422902,-9.687511598,-10.11556009,-10.72843021;

-1.857639586,-1.957452486,-2.470984689,-3.062689166,-3.509750643,-3.854521141,-4.140167492,-4.379725178,-4.757975644,-5.055543676,-5.232154316,-5.349116723,-5.429916024,-5.49980907,-5.563885926,-5.623900948,-5.680172244,-5.739877174,-5.799362568,-5.859190044,-6.003020792,-6.132451553,-6.368187081,-6.575185071,-6.759675551,-6.927919537,-7.086860721,-7.234974121,-7.507625547,-7.762011982,-8.000455429,-8.454859492,-8.875563399,-9.481279735;

-1.365547485,-1.446235798,-1.813138039,-2.281968712,-2.670552781,-2.983595509,-3.245797046,-3.469892031,-3.832100546,-4.119490301,-4.290348984,-4.403439432,-4.480704925,-4.546528082,-4.606913407,-4.664248563,-4.719304028,-4.776049427,-4.832629299,-4.889537782,-5.027590152,-5.153227237,-5.380584628,-5.580493942,-5.7598237,-5.923801578,-6.079298242,-6.224684946,-6.489611384,-6.737010765,-6.966281042,-7.408495191,-7.819273379,-8.413029993;

-0.998637863,-1.064532406,-1.336396533,-1.690339194,-2.01510965,-2.293787159,-2.535815998,-2.746242647,-3.087843914,-3.360186002,-3.523387622,-3.630699756,-3.70326159,-3.766322509,-3.823529211,-3.877932535,-3.928918745,-3.98299855,-4.037070517,-4.091595322,-4.224603457,-4.346329013,-4.567147167,-4.761361213,-4.935131023,-5.093775401,-5.244028236,-5.384734488,-5.640476649,-5.879453675,-6.10194068,-6.528740004,-6.927059044,-7.505775005;

-0.717328938,-0.773613246,-0.989918399,-1.272529233,-1.540794518,-1.77868249,-1.98888666,-2.175044858,-2.485419817,-2.732384319,-2.883663189,-2.98542024,-3.054432504,-3.113553316,-3.167981196,-3.221183478,-3.271056573,-3.323614768,-3.375933581,-3.428707794,-3.556878325,-3.673974495,-3.887340632,-4.074630985,-4.242241615,-4.395481517,-4.540612536,-4.676465037,-4.923178407,-5.15375503,-5.368958235,-5.783633031,-6.171325536,-6.737996686;

-0.527740772,-0.5749666,-0.752853186,-0.983644795,-1.201261942,-1.398302038,-1.577248176,-1.738446876,-2.013000239,-2.229094055,-2.367601437,-2.458970129,-2.522995827,-2.578189201,-2.629101419,-2.679880639,-2.727490235,-2.778218168,-2.829134301,-2.880407008,-3.004430549,-3.117453893,-3.320251851,-3.498297223,-3.658569421,-3.80552602,-3.944633615,-4.07468591,-4.313197705,-4.536601204,-4.746271512,-5.151051589,-5.531314659,-6.090936368;

-0.458423092,-0.501800833,-0.662893283,-0.863129231,-1.055734129,-1.231620148,-1.392626392,-1.538821198,-1.794520976,-1.99155637,-2.122006284,-2.20721757,-2.26812044,-2.321153281,-2.370168121,-2.419675952,-2.46611512,-2.515530073,-2.565004552,-2.614798733,-2.735443262,-2.845169577,-3.043150287,-3.21701794,-3.373802811,-3.517261727,-3.652962305,-3.780305757,-4.013416538,-4.232378127,-4.4379898,-4.837242924,-5.21157985,-5.764550591;

-0.425446944,-0.468675068,-0.619890481,-0.800280855,-0.97451101,-1.137477311,-1.287559227,-1.423902689,-1.666740809,-1.848905772,-1.974293198,-2.054959334,-2.113653661,-2.165183486,-2.212887544,-2.261531986,-2.307171368,-2.355704709,-2.404223159,-2.452990275,-2.571354407,-2.678836573,-2.873679412,-3.044783245,-3.198629735,-3.339914243,-3.473730592,-3.599312328,-3.828949983,-4.04473404,-4.247825054,-4.642747114,-5.01335429,-5.562598579;

-0.394946609,-0.440635852,-0.584059275,-0.745011696,-0.900271429,-1.049585891,-1.188524807,-1.315777877,-1.544275249,-1.712990043,-1.833442864,-1.909477994,-1.965931777,-2.016010765,-2.062362631,-2.11014335,-2.154938256,-2.202617484,-2.2501858,-2.297909797,-2.413979236,-2.519169802,-2.710779998,-2.878614629,-3.029974268,-3.169290215,-3.301088218,-3.424831505,-3.650860304,-3.863599035,-4.064284349,-4.454802398,-4.823426622,-5.370292908;

-3.721200854,-3.969255641,-5.099097187,-5.914454918,-6.461791343,-6.873985113,-7.216288263,-7.493073424,-7.926670397,-8.262152561,-8.465035761,-8.598745987,-8.693550791,-8.775361973,-8.850346705,-8.919712354,-8.984708485,-9.053898333,-9.122729567,-9.191315556,-9.355537696,-9.503363053,-9.774632181,-10.00958147,-10.21913899,-10.40987987,-10.59588578,-10.76910146,-11.08746968,-11.3791468,-11.65321996,-12.17164727,-12.65585256,-13.35197926;

-2.7367934,-2.913650407,-3.834172198,-4.601176646,-5.125603356,-5.519493341,-5.847395353,-6.114335937,-6.531657897,-6.855993684,-7.050562737,-7.179039874,-7.269529755,-7.347710807,-7.419369311,-7.485755315,-7.54806435,-7.614381407,-7.680273951,-7.746115274,-7.904288898,-8.04717197,-8.308796033,-8.53619607,-8.739576099,-8.925368456,-9.106858742,-9.276274063,-9.588557562,-9.877889602,-10.1467638,-10.66138859,-11.14369983,-11.84022045;

-1.973503244,-2.102613755,-2.814128092,-3.509128913,-4.002179769,-4.375425638,-4.68360611,-4.938013667,-5.339018038,-5.653133947,-5.8405688,-5.964748359,-6.051552848,-6.126329948,-6.194950761,-6.258567085,-6.318293262,-6.381745721,-6.444712393,-6.507783387,-6.659852215,-6.797662995,-7.049919213,-7.269698899,-7.467449577,-7.648875513,-7.826439533,-7.992121174,-8.298246383,-8.580002243,-8.845043967,-9.352933535,-9.828838096,-10.51876108;

-1.401342813,-1.500130785,-2.012060159,-2.605347716,-3.055152397,-3.402277929,-3.690133917,-3.93148745,-4.312750408,-4.612850324,-4.791718576,-4.910434207,-4.992643792,-5.063701272,-5.128985922,-5.190258992,-5.247802986,-5.308947538,-5.369865745,-5.431166616,-5.579001219,-5.712705672,-5.957200405,-6.172353999,-6.365161853,-6.541981066,-6.709875373,-6.86734597,-7.164138514,-7.437883773,-7.697663092,-8.194969811,-8.662870287,-9.342858565;

-0.984964007,-1.06462842,-1.430742389,-1.901155285,-2.292104512,-2.607136007,-2.871290058,-3.096969145,-3.461164847,-3.750169108,-3.922588105,-4.036829364,-4.115037821,-4.181869167,-4.243316161,-4.301798793,-4.357608114,-4.41578644,-4.473834178,-4.532277223,-4.674412802,-4.8042049,-5.0401567,-5.247947948,-5.435380754,-5.607734191,-5.771906046,-5.926266367,-6.214612799,-6.481384841,-6.732494166,-7.216753686,-7.673249011,-8.338796029;

-0.679640062,-0.744545377,-1.016411527,-1.37234223,-1.698639639,-1.978071907,-2.220930705,-2.432045697,-2.774951719,-3.048478648,-3.212928612,-3.321081159,-3.39435924,-3.457715478,-3.515673016,-3.571244273,-3.623419099,-3.678872447,-3.734325533,-3.790297657,-3.927226964,-4.0529728,-4.282059321,-4.483826895,-4.665435002,-4.832256242,-4.991019544,-5.14065673,-5.41877767,-5.677372894,-5.921503035,-6.39066195,-6.83424306,-7.484786567;

-0.445696747,-0.501155623,-0.717723608,-0.999835113,-1.26851606,-1.506836608,-1.717630141,-1.904330289,-2.215645375,-2.463548317,-2.615850323,-2.71743373,-2.786885697,-2.846857817,-2.902196738,-2.956427661,-3.007366621,-3.061182188,-3.11477852,-3.168899897,-3.300717525,-3.421555344,-3.642656065,-3.837032555,-4.01211369,-4.173207034,-4.326646644,-4.471178258,-4.739528898,-4.989374205,-5.22580215,-5.682865978,-6.115762235,-6.752037901;

-0.293087653,-0.338705676,-0.515953869,-0.743741344,-0.960859224,-1.158113296,-1.337364061,-1.498928982,-1.774222339,-1.990640683,-2.129448481,-2.221491038,-2.286115851,-2.342018151,-2.393712536,-2.445405563,-2.493974053,-2.545849296,-2.597945572,-2.650472375,-2.777868996,-2.89433927,-3.104454434,-3.289301905,-3.456739949,-3.611279025,-3.758432176,-3.897128073,-4.157365044,-4.400172056,-4.630963693,-5.07737467,-5.50189502,-6.129519814;

-0.239628186,-0.281601362,-0.442182813,-0.640976233,-0.830705737,-1.005729788,-1.166920528,-1.313365164,-1.569414384,-1.766257808,-1.897813799,-1.983603015,-2.045034118,-2.098706549,-2.148438972,-2.198793169,-2.246130476,-2.296628154,-2.347230469,-2.398224357,-2.522109264,-2.635183559,-2.840291724,-3.020846459,-3.184692785,-3.335994428,-3.480082291,-3.616304393,-3.870813579,-4.109174103,-4.33598311,-4.777043221,-5.193609212,-5.812978184;

-0.215407864,-0.257374743,-0.408200682,-0.587461128,-0.759851194,-0.920599304,-1.069843013,-1.206374033,-1.448938655,-1.631544106,-1.757996042,-1.839179279,-1.898365457,-1.950498088,-1.998883387,-2.048338629,-2.094839333,-2.144420393,-2.194036621,-2.243977581,-2.365504576,-2.476272494,-2.678149856,-2.855871551,-

3.017279649,-3.166493724,-3.308677564,-3.443131198,-3.694158856,-3.929243819,-4.153027513,-4.589031776,-5.001842854,-5.615851177;

-0.193206796,-0.237679233,-0.380620609,-0.541356262,-0.694546743,-0.842083597,-0.979455164,-1.105270822,-1.33294045,-1.502675395,-1.624157296,-1.700641018,-1.757550586,-1.808200013,-1.855207144,-1.90376493,-1.949393975,-1.99809112,-2.046736963,-2.095614191,-2.214784722,-2.323223976,-2.521797143,-2.69688601,-2.855755837,-3.003069904,-3.14327329,-3.275935193,-3.523044905,-3.755552808,-3.97708731,-4.408085904,-4.819273663,-5.432205991;

-2.890333243,-3.134750957,-4.222614709,-5.02405629,-5.570806507,-5.984289068,-6.326588827,-6.604164311,-7.039971326,-7.378086482,-7.586502019,-7.721545454,-7.819428748,-7.904485376,-7.982639085,-8.055079019,-8.123050838,-8.196120578,-8.268540385,-8.340268041,-8.512225759,-8.667591065,-8.952786803,-9.201612671,-9.425372913,-9.630716878,-9.829554516,-10.01696033,-10.37997889,-10.70598153,-11.02201504,-11.62101913,-12.19495649,-13.03036827;

-2.00066692,-2.177782058,-3.070877276,-3.826005654,-4.349339171,-4.743788792,-5.07134129,-5.338622852,-5.757681523,-6.084161972,-6.283815129,-6.413667006,-6.507331195,-6.588862323,-6.663769271,-6.733320606,-6.798724789,-6.869001834,-6.938558069,-7.007598061,-7.173692646,-7.324323508,-7.600377451,-7.842131181,-8.060050046,-8.260760514,-8.455214597,-8.638924731,-8.993525051,-9.319589178,-9.625778237,-10.22031034,-10.79015465,-11.62148858;

-1.359150208,-1.490188173,-2.186232995,-2.871613773,-3.363380991,-3.736813309,-4.044873716,-4.299470911,-4.701739269,-5.017359411,-5.209276222,-5.334670187,-5.424574466,-5.50262649,-5.57443288,-5.641163388,-5.703937279,-5.771287507,-5.837913667,-5.904183275,-6.064219506,-6.209850625,-6.47666531,-6.710859701,-6.923074883,-7.119271673,-7.309534428,-7.489039179,-7.836657716,-8.151597377,-8.453805181,-9.038599332,-9.596145949,-10.41488443;

-0.89442538,-0.99582399,-1.503847987,-2.091328841,-2.540166757,-2.887055382,-3.174686237,-3.415727864,-3.7974546,-4.098302201,-4.280784681,-4.400286072,-4.485242877,-4.558836052,-4.626650452,-4.690577341,-4.75075397,-4.815359967,-4.87949445,-4.94349586,-5.098452288,-5.239662058,-5.498525995,-5.726747057,-5.933119704,-6.12406801,-6.306937289,-6.480312369,-6.818602151,-7.125088054,-7.420861447,-7.99167116,-8.539236496,-9.340788147;

-0.56353832,-0.645466934,-1.015796321,-1.486556141,-1.877656645,-2.192365191,-2.456303444,-2.681387834,-3.044068486,-3.331934545,-3.506247084,-3.620044374,-3.700062785,-3.769148875,-3.832867201,-3.893919662,-3.95163878,-4.013300031,-4.074645651,-4.13594136,-4.285373113,-4.422397152,-4.672596133,-4.893533784,-5.09446908,-5.280798989,-5.459464135,-5.628977801,-5.957598139,-6.257043851,-6.545009019,-7.101087514,-7.633901115,-8.414448072;

-0.322292849,-0.389158617,-0.669057282,-1.03098343,-1.358787086,-1.637423985,-1.878862231,-2.088230058,-2.428732691,-2.700481944,-2.865907199,-2.973055752,-3.047717749,-3.112389375,-3.17212058,-3.230246209,-3.284964767,-3.343746741,-3.40232712,-3.460987079,-3.604872281,-3.737542324,-3.980305201,-4.194972237,-4.389904941,-4.570577858,-4.743736348,-4.908548935,-5.22512989,-5.516732483,-5.797155329,-6.337731858,-6.855489688,-7.618026191;

-0.135415715,-0.192539133,-0.417824088,-0.706300544,-0.977603887,-1.216202022,-1.426400727,-1.611946005,-1.920843943,-2.1670938,-2.319981131,-2.419646581,-2.489895635,-2.551355938,-2.608266058,-2.664678389,-2.717826545,-2.774634309,-2.831074855,-2.887631588,-3.025823629,-3.1530558,-3.386741633,-3.5933175,-3.781167845,-3.955606893,-4.123094288,-4.282344787,-4.587721738,-4.869934013,-5.141482955,-5.668519312,-6.173972573,-6.916426652;

-0.015659482,-0.062709081,-0.248057999,-0.482419409,-0.703224493,-0.901909286,-1.081113466,-1.242002664,-1.515422471,-1.730784793,-1.869719434,-1.960380682,-2.025558721,-2.082531301,-2.135424665,-2.189066936,-2.239627351,-2.294182943,-2.348792252,-2.403476384,-2.536545361,-2.658759241,-2.880806531,-3.07754231,-3.257289893,-3.424711223,-3.585479612,-3.738642434,-4.034984598,-4.309427491,-4.574502033,-5.088758899,-5.582826766,-6.311750998;

0.022684948,-0.020626707,-0.18884207,-0.395043281,-0.588508091,-0.76524763,-0.926826166,-1.072932458,-1.327231747,-1.523540356,-1.655405077,-1.73985015,-1.801653124,-1.856171718,-1.906891314,-1.959050465,-2.00824639,-2.061271919,-2.114260757,-2.167305565,-2.296613368,-2.41521686,-2.63176372,-2.823904029,-2.999759655,-3.163820433,-3.321508,-3.472191357,-3.762075154,-4.031859978,-4.292948174,-4.801310765,-5.283807642,-6.001084739;

0.038284462,-0.004672112,-0.162687822,-0.349653642,-0.526590187,-0.68929775,-0.839112188,-0.975565734,-1.216527663,-1.399328039,-1.526005403,-1.605912336,-1.665375018,-1.718215176,-1.767459175,-1.818633121,-1.86690844,-1.918926325,-1.970844547,-2.022787254,-2.149574037,-2.265737594,-2.478751747,-2.667846456,-2.841297313,-3.003172544,-3.158825741,-3.307646139,-3.593948591,-3.860212021,-4.117287354,-4.619283154,-5.097720835,-5.806379874;

0.051018244,0.006297417,-0.142919536,-0.311076097,-0.469609437,-0.619715886,-0.758054075,-0.883967805,-1.110293076,-1.280913199,-1.402553559,-1.477824826,-1.534915953,-1.586134686,-1.633884605,-1.684079265,-1.731413789,-1.782458174,-1.833342441,-1.884177768,-2.008469478,-2.122199117,-2.331648248,-2.518168884,-2.688910872,-2.848885413,-3.002537015,-3.149575164,-3.43134842,-3.69568253,-3.950611297,-4.446392586,-4.923083629,-5.6313148;

-2.460524787,-2.703133854,-3.784720046,-4.587313075,-5.137241313,-5.553439479,-5.897898911,-6.177378549,-6.616341871,-6.957342727,-7.168173301,-7.306012351,-7.406530085,-7.494080466,-7.574731015,-7.649639243,-7.720023659,-7.795440741,-7.870152942,-7.94424918,-8.122400909,-8.284089063,-8.582424835,-8.84438152,-9.081494813,-9.300469359,-9.515493844,-9.719151033,-10.12474808,-10.48403878,-10.83735963,-11.50562592,-12.15178947,-13.09119642;

-1.643410889,-1.819664997,-2.709899702,-3.466461548,-3.992933486,-4.39000041,-4.719760513,-4.988941851,-5.411211209,-5.740602093,-5.942643273,-6.075384238,-6.171756871,-6.255871321,-6.333327171,-6.405406553,-6.473300437,-6.545982877,-6.617869002,-6.689295711,-6.861654854,-7.01866666,-7.307955721,-7.562938958,-7.794200388,-8.008502505,-8.218757641,-8.418351519,-8.813075059,-9.17323876,-9.512806159,-10.17499241,-10.81396897,-11.74488009;

-1.069219365,-1.200314125,-1.894760921,-2.581740183,-3.076598658,-3.452627281,-3.762961145,-4.019467029,-4.425003021,-4.74351644,-4.937761353,-5.066053036,-5.158706885,-5.239355908,-5.313723603,-5.382994233,-5.448262539,-5.518028816,-5.587014849,-5.655696326,-5.822072211,-5.974139196,-6.254244546,-6.501645072,-6.727102676,-6.936718704,-7.142148168,-7.336819906,-7.723129623,-8.069293111,-8.40404568,-9.053291089,-9.675455363,-10.58878615;

-0.653652908,-0.755531864,-1.262179995,-1.850815355,-2.302588075,-2.651933771,-2.941894869,-3.184792383,-3.56973,-3.873414816,-4.058135735,-4.1804914,-4.268204476,-4.344568226,-4.415101026,-4.48173509,-4.544560815,-4.611724256,-4.678319595,-4.744849668,-4.906376379,-5.054108626,-5.326333323,-5.568245431,-5.788293637,-5.993067171,-6.189769219,-6.377142469,-6.75207335,-7.087483547,-7.414505596,-8.046473403,-8.6562015,-9.546364759;

-0.353872347,-0.436264641,-0.806759873,-1.279356654,-1.673142695,-1.98998322,-2.256070146,-2.482846164,-2.848317361,-3.138667792,-3.315019059,-3.431439034,-3.514088212,-3.585739264,-3.651992919,-3.715627954,-3.77587638,-3.840015715,-3.903745586,-3.96746748,-4.123356126,-4.26689127,-4.530447769,-4.764663453,-4.978930441,-5.178744804,-5.371123146,-5.554593708,-5.917932155,-6.245621997,-6.563628492,-7.17837904,-7.7699429,-8.634413811;

-0.130299228,-0.197415769,-0.478516675,-0.843429293,-1.173498288,-1.453489104,-1.696224761,-1.906629704,-2.249139458,-2.522772149,-2.68981834,-2.799148719,-2.87613895,-2.942873977,-3.004861397,-3.065475612,-3.122654535,-3.183862019,-3.244806211,-3.305862806,-3.456186582,-3.59535161,-3.851314674,-4.078795349,-4.286690562,-4.480556308,-4.66717668,-4.845845255,-5.195619895,-5.515394514,-5.825577392,-6.422966385,-6.996373404,-7.84081938;

0.046242008,-0.011104342,-0.237914808,-0.52715102,-0.799668035,-1.039211276,-1.250475282,-1.43696826,-1.74752582,-1.995446954,-2.149752104,-2.250659059,-2.322864233,-2.386550059,-2.445687185,-2.504451582,-2.559942251,-2.619074916,-2.677798359,-2.736672068,-2.881079441,-3.014571865,-3.260968101,-3.479680789,-3.679948321,-3.86711633,-4.047737719,-4.220428283,-4.557928723,-4.867515387,-5.167689761,-5.749817154,-6.309395818,-7.128665764;

0.156787851,0.110700124,-0.07510666,-0.306905147,-0.527776835,-0.727174349,-0.907123579,-1.068813286,-1.343797772,-1.560292866,-1.699959817,-1.792497632,-1.859669889,-1.918681559,-1.973640836,-2.029499616,-2.082279001,-2.139034988,-2.19581473,-2.252716595,-2.391694119,-2.519835086,-2.754169421,-2.962556895,-3.154220026,-3.33388866,-3.507361413,-3.673731713,-4.001509632,-4.302969729,-4.595835831,-5.163216905,-5.708589244,-6.510692595;

0.192315868,0.149929407,-0.018705787,-0.223779987,-0.414359864,-0.590535656,-0.752725949,-0.899499994,-1.154930826,-1.35172518,-1.485175951,-1.57140621,-1.635094099,-1.691548691,-1.744235999,-1.798518646,-1.849848502,-1.904985729,-1.960076801,-2.015270139,-2.150305031,-2.274701551,-2.503284456,-2.706824254,-2.894357459,-3.070847822,-3.241482252,-3.405511296,-3.72616672,-4.022653144,-4.311510954,-4.872209577,-5.402731393,-6.190454368;

0.206432172,0.164328101,0.005845218,-0.180285061,-0.355382274,-0.515758974,-0.664934586,-0.801978588,-1.043321241,-1.227306992,-1.355547395,-1.437179535,-1.498461969,-1.553173198,-1.604323428,-1.65756172,-1.707914367,-1.761990759,-1.815964928,-1.870010936,-2.002404267,-2.124266001,-2.349154792,-2.54948218,-2.735103932,-2.909456672,-3.077947798,-3.240046723,-3.556885903,-3.849525835,-4.133452605,-4.685964186,-5.212269753,-5.987804342;

0.216709562,0.172882481,0.02347821,-0.144946771,-0.30116189,-0.449333718,-0.586024892,-0.710436959,-0.936456688,-1.108921543,-1.232099773,-1.309019908,-1.367861108,-1.420886151,-1.470489529,-1.522690272,-1.572049485,-1.625099612,-1.677996318,-1.730898867,-1.860685545,-1.980046864,-2.201215325,-2.399650257,-

2.582475702,-2.754933379,-2.921388,-3.081685332,-3.393240393,-3.684233693,-3.965736275,-4.510719675,-5.034656774,-5.810462307;

-1.973193989,-2.217117582,-3.274220762,-4.065796124,-4.613704105,-5.029788176,-5.37357912,-5.653061335,-6.092711312,-6.435117049,-6.649783713,-6.791638125,-6.896232757,-6.988434534,-7.073629685,-7.152938793,-7.227550722,-7.307108996,-7.385681828,-7.463666111,-7.652059736,-7.823954646,-8.14190025,-8.423193896,-8.679757518,-8.918389864,-9.154129172,-9.379086485,-9.841627797,-10.24547013,-10.64863966,-11.40842316,-12.14837732,-13.21991188;

-1.261780194,-1.44028435,-2.316978342,-3.064293111,-3.588940402,-3.985613359,-4.314756301,-4.583783825,-5.006658822,-5.337249186,-5.542884393,-5.67966782,-5.780113922,-5.868898391,-5.950854147,-6.027315072,-6.099467481,-6.176293775,-6.252028732,-6.327296133,-6.509830414,-6.676974571,-6.985858296,-7.260116769,-7.510594244,-7.74427525,-7.974627727,-8.194881333,-8.642812752,-9.047780247,-9.431481299,-10.18192897,-10.90826037,-11.96346702;

-0.761860312,-0.895980918,-1.583374093,-2.26320806,-2.756545582,-3.132220827,-3.442331483,-3.698779922,-4.104889887,-4.424331661,-4.621867886,-4.754103991,-4.850789419,-4.936031103,-5.014834049,-5.08842813,-5.157889257,-5.231741223,-5.304568962,-5.377087844,-5.553638787,-5.715824053,-6.015474169,-6.281946712,-6.526258212,-6.754766463,-6.979420564,-7.193630221,-7.630841008,-8.018250032,-8.395080563,-9.127108159,-9.830605534,-10.85967574;

-0.389550052,-0.495034364,-1.000188733,-1.58410936,-2.035018571,-2.383962049,-2.673880692,-2.916517506,-3.301790682,-3.606128078,-3.793693362,-3.91974535,-4.011320245,-4.092085214,-4.166886529,-4.237782521,-4.304744336,-4.375914825,-4.446230929,-4.516465188,-4.687947277,-4.845654867,-5.137278669,-5.397963857,-5.636656706,-5.860164763,-6.07581522,-6.282475561,-6.706448847,-7.080117638,-7.447129327,-8.156931571,-8.843353093,-9.838814828;

-0.116276054,-0.201924401,-0.577122597,-1.049724553,-1.443552093,-1.759934308,-2.026000378,-2.252368664,-2.617112909,-2.90711918,-3.085474645,-3.204830768,-3.290832566,-3.366551444,-3.436776297,-3.504521535,-3.568547558,-3.636623319,-3.704022415,-3.771404661,-3.937210247,-4.09057338,-4.373510909,-4.626213615,-4.858791714,-5.07691581,-5.287837045,-5.490055401,-5.899868632,-6.264399063,-6.621524485,-7.310237187,-7.973129616,-8.934913321;

0.090598867,0.020724076,-0.268375267,-0.638237152,-0.969431369,-1.248680299,-1.490388701,-1.699498698,-2.040389869,-2.313024492,-2.48135513,-2.592908605,-2.672794957,-2.742818013,-2.808298641,-2.872873038,-2.933927028,-2.998930025,-3.063439481,-3.128037657,-3.288118572,-3.436993099,-3.711880805,-3.957355705,-4.183173038,-4.395050813,-4.599864245,-4.797056766,-5.191094793,-5.54728661,-5.895430866,-6.563909389,-7.203319882,-8.14116591;

0.255858628,0.196027574,-0.039613178,-0.333631596,-0.607821849,-0.847328677,-1.058206929,-1.243933042,-1.552915886,-1.800056939,-1.955269199,-2.057326814,-2.131879303,-2.198839381,-2.261224716,-2.323641182,-2.382731896,-2.445407966,-2.507498197,-2.569724226,-2.723380855,-2.866057218,-3.130316063,-3.366022711,-3.583400645,-3.787844395,-3.986049251,-4.176547438,-4.556597253,-4.901145525,-5.237266865,-5.886896128,-6.509542715,-7.413628271;

0.356861495,0.309126882,0.115512485,-0.119386632,-0.342359622,-0.542683318,-0.722660251,-0.884023927,-1.15807352,-1.374276584,-1.514234288,-1.608332911,-1.677725061,-1.739627644,-1.797496405,-1.856770672,-1.912931194,-1.972960736,-2.032847929,-2.092867395,-2.240409271,-2.377024549,-2.628402668,-2.853095169,-3.061083223,-3.257249928,-3.447563512,-3.631135168,-3.999662604,-4.335064401,-4.662173997,-5.293263737,-5.897177171,-6.779322867;

0.388445512,0.344527443,0.168523713,-0.040766436,-0.231932038,-0.408657599,-0.571205389,-0.717908489,-0.972422337,-1.169117168,-1.303435231,-1.391115979,-1.456874031,-1.515989181,-1.571365757,-1.628894512,-1.68344768,-1.741694235,-1.799771465,-1.857968734,-2.001260877,-2.133894508,-2.379018295,-2.59844223,-2.801890724,-2.994783883,-3.182224441,-3.363425464,-3.723688232,-4.05371881,-4.376764494,-5.000023201,-5.584102623,-6.448035054;

0.400149578,0.356746424,0.19124055,0.000592915,-0.176139525,-0.336272571,-0.485288133,-0.622447791,-0.862662996,-1.047536437,-1.176611594,-1.259634008,-1.322913782,-1.380140737,-1.433841217,-1.490218254,-1.543687473,-1.600768491,-1.657648986,-1.714628284,-1.855067418,-1.985001007,-2.226097336,-2.441999821,-2.643739231,-2.83440239,-3.019471826,-3.198564189,-3.554617258,-3.880264775,-4.197577387,-4.80895599,-5.388844491,-6.234833708;

0.406146066,0.361588848,0.20602386,0.032696795,-0.125358002,-0.273909506,-0.410128435,-0.533615198,-0.758395348,-0.932665266,-1.056648326,-1.134874457,-1.195632744,-1.2510295,-1.30305242,-1.35828707,-1.410665768,-1.466614038,-1.522345575,-1.578115788,-1.715746693,-1.843047445,-2.080104673,-2.294435975,-2.493144353,-2.681845281,-2.864752711,-3.041905512,-3.391459805,-3.715976423,-4.030484127,-4.631951891,-5.208275669,-6.05647346;

-1.676980126,-1.918494902,-2.979523965,-3.780383243,-4.335198998,-4.756290235,-5.104504724,-5.387647132,-5.832983188,-6.180173011,-6.397028728,-6.541170944,-6.648864184,-6.743761358,-6.831703786,-6.913754742,-6.991082621,-7.073243405,-7.154281043,-7.23477321,-7.43031259,-7.609722561,-7.943720321,-8.241339301,-8.51461246,-8.770364511,-9.027286735,-9.273342619,-9.78899196,-10.23466349,-10.68262159,-11.52283303,-12.3417063,-13.52140464;

-1.031232922,-1.207061743,-2.088315422,-2.843968412,-3.375299196,-3.776994417,-4.110718983,-4.383469198,-4.8121403,-5.147573152,-5.355361163,-5.494532017,-5.598100912,-5.689647715,-5.774353615,-5.853576037,-5.928471203,-6.007912375,-6.08611094,-6.163863358,-6.353519378,-6.528117614,-6.85301774,-7.143500737,-7.410445312,-7.660934672,-7.911864724,-8.152587363,-8.650732647,-9.0969322,-9.521028559,-10.34870115,-11.14860404,-12.30533093;

-0.570429637,-0.702841213,-1.392220865,-2.079528845,-2.579293262,-2.960015312,-3.274673711,-3.534829756,-3.946853218,-4.271190618,-4.470852527,-4.605535314,-4.705406749,-4.793427442,-4.874998801,-4.951371672,-5.023580453,-5.100063273,-5.175396978,-5.250434095,-5.434194977,-5.603879182,-5.919573844,-6.202179509,-6.462738726,-6.707704157,-6.952143923,-7.18590796,-7.670754633,-8.096883673,-8.511920496,-9.316092588,-10.0877312,-11.21183222;

-0.222536148,-0.327186165,-0.830454701,-1.419348348,-1.876120854,-2.229931795,-2.524444955,-2.770750811,-3.161939872,-3.471175371,-3.660867923,-3.789417997,-3.88427815,-3.96836417,-4.046418551,-4.120520906,-4.190625114,-4.264790671,-4.337913643,-4.411013189,-4.590334307,-4.755736789,-5.063502533,-5.341513591,-5.597379488,-5.838101546,-6.070842198,-6.294527527,-6.76305929,-7.171308686,-7.574722759,-8.351653662,-9.102462949,-10.18401598;

0.036597944,-0.048835427,-0.42288201,-0.899282616,-1.298180928,-1.618827761,-1.889081873,-2.118848123,-2.489561665,-2.784584696,-2.965384928,-3.087382301,-3.176876995,-3.255745921,-3.329080038,-3.399922831,-3.467378107,-3.538333558,-3.608383537,-3.678404179,-3.851821229,-4.012988361,-4.312282054,-4.581952256,-4.831424234,-5.066455811,-5.294533001,-5.514033887,-5.965647999,-6.363382538,-6.753934101,-7.505675587,-8.227908074,-9.269123777;

0.237015777,0.167131819,-0.122014123,-0.495929545,-0.831443954,-1.114293848,-1.359426801,-1.571418742,-1.91729025,-2.194084952,-2.364451454,-2.47817254,-2.561258322,-2.634326327,-2.702758924,-2.770224834,-2.834160351,-2.90201031,-2.969211308,-3.03647115,-3.204345035,-3.361218892,-3.652560787,-3.914687227,-4.157165429,-4.38578157,-4.607587236,-4.821892086,-5.256090663,-5.644738701,-6.025901464,-6.753279265,-7.446260472,-8.459465959;

0.398469403,0.338559479,0.102655078,-0.193415512,-0.470230179,-0.712206975,-0.925789192,-1.114000372,-1.427233412,-1.677977021,-1.834967942,-1.938702705,-2.016115246,-2.085825657,-2.150979857,-2.21621715,-2.278133583,-2.343619569,-2.408368606,-2.473226936,-2.634536945,-2.78502005,-3.065298053,-3.316923215,-3.550390681,-3.771075993,-3.985817294,-4.192959518,-4.611764705,-4.987614228,-5.354800698,-6.059374533,-6.73256639,-7.705184399;

0.49549667,0.448557096,0.255546994,0.022296494,-0.201582145,-0.403545308,-0.585389606,-0.748745215,-1.026560987,-1.245621738,-1.386863376,-1.482797319,-1.554895382,-1.619383635,-1.679888621,-1.741862993,-1.800743604,-1.863489387,-1.925955928,-1.988551543,-2.143471493,-2.28755188,-2.554478417,-2.794468541,-3.017895875,-3.229694171,-3.43595731,-3.635647283,-4.041302662,-4.406672931,-4.762956663,-5.445810029,-6.09605904,-7.042421197;

0.527122012,0.484008886,0.308666893,0.100717812,-0.088753397,-0.26582157,-0.429803514,-0.57810218,-0.835769301,-1.034428674,-1.170466192,-1.259977822,-1.328306708,-1.389912143,-1.447838816,-1.507985064,-1.565186275,-1.62606804,-1.686664951,-1.747383346,-1.897888881,-2.03789408,-2.298423825,-2.532918266,-2.751589391,-2.960147674,-3.163588157,-3.360902399,-3.757304004,-4.116746252,-4.468823228,-5.142228612,-5.768953133,-6.694334766;

0.539310041,0.496588167,0.331664856,0.142365315,-0.032994157,-0.191897753,-0.341247661,-0.479750077,-0.72235914,-0.909327362,-1.040070346,-1.12493058,-1.190686629,-1.250336218,-1.306529086,-1.365463698,-1.421528041,-1.481188867,-1.540540738,-1.600001834,-1.74753225,-1.88475949,-2.141144077,-2.371962064,-2.589205703,-2.795578902,-2.996506017,-3.19161098,-3.583255872,-3.937840474,-4.282835703,-4.94180675,-5.564298353,-6.467258085;

0.544640037,0.500715224,0.346051269,0.173611573,0.017703159,-0.12961325,-0.265170884,-0.388228873,-0.614594253,-0.791164715,-0.91676088,-0.996799129,-1.05992582,-1.11767061,-1.172128676,-1.229859146,-1.284781559,-1.343249727,-1.401407021,-1.459624336,-1.604232165,-1.738764635,-1.991011422,-2.220721055,-2.434899758,-2.639296159,-2.838020455,-3.031119342,-3.414869738,-3.768532221,-4.110005206,-4.757249394,-5.374898459,-6.28175527;

-1.401245056,-1.632619248,-2.63323279,-3.413961926,-3.970148641,-4.394773668,-4.744157444,-5.029103566,-5.47830021,-5.829557132,-6.054678032,-6.202309405,-6.31575168,-6.416091985,-6.509259136,-6.596342291,-6.678553054,-6.767101363,-6.854097715,-6.939827648,-7.147469511,-7.338466812,-7.693706252,-8.011574333,-

8.304662523,-8.580001132,-8.853437999,-9.116483727,-9.672409812,-10.15052505,-10.63257971,-11.53229896,-12.40404626,-13.65413783;

-0.81081115,-0.983725954,-1.820698156,-2.55959277,-3.091271582,-3.495090373,-3.829243066,-4.103045268,-4.534712478,-4.873426527,-5.088867332,-5.231435038,-5.340805063,-5.437873897,-5.527873956,-5.612198301,-5.692031398,-5.777849538,-5.861997678,-5.944975812,-6.146749397,-6.3329213,-6.679030316,-6.989700242,-7.276251984,-7.545987984,-7.813094803,-8.070351782,-8.606478265,-9.084821226,-9.539829477,-10.42268583,-11.27122606,-12.49056582;

-0.374558777,-0.507283183,-1.169027095,-1.84214418,-2.341713816,-2.723809869,-3.039149682,-3.300130408,-3.7145268,-4.041316,-4.24792752,-4.385810404,-4.491390024,-4.584855406,-4.671655692,-4.753078357,-4.830178731,-4.912946856,-4.994202589,-5.074440977,-5.270288488,-5.451506653,-5.788274586,-6.090868073,-6.370593611,-6.634240478,-6.894326377,-7.144065824,-7.664983464,-8.121056709,-8.565855659,-9.421152908,-10.23660809,-11.41722022;

-0.039770106,-0.146076406,-0.638809966,-1.218571129,-1.675302358,-2.029823588,-2.324930867,-2.571521746,-2.964228706,-3.275110617,-3.470855011,-3.602202931,-3.702467771,-3.791464371,-3.874275678,-3.953113081,-4.027827478,-4.107911081,-4.18655873,-4.264420863,-4.455049971,-4.631616347,-4.960082058,-5.256728902,-5.53081492,-5.789392795,-6.03964436,-6.280764456,-6.784441625,-7.221949724,-7.653646307,-8.476990376,-9.267570963,-10.39832838;

0.207661641,0.120579986,-0.253319122,-0.728563449,-1.128414666,-1.449533522,-1.720435401,-1.950208096,-2.320730728,-2.615726933,-2.800999833,-2.924811665,-3.018946162,-3.102501167,-3.180382765,-3.255890463,-3.327473892,-3.40426971,-3.4798129,-3.554625183,-3.739252705,-3.911123992,-4.230577299,-4.518641572,-4.785824016,-5.038090779,-5.282733114,-5.518508923,-6.003504817,-6.429590367,-6.847015356,-7.642370181,-8.40010267,-9.484435362;

0.395902125,0.32446213,0.030505377,-0.348992776,-0.687092152,-0.970380589,-1.215236422,-1.426333709,-1.771205232,-2.047303946,-2.221122283,-2.336165673,-2.42344917,-2.500568569,-2.573122327,-2.645104284,-2.713444649,-2.786850679,-2.859293779,-2.931108223,-3.109636845,-3.276669937,-3.587163877,-3.867020449,-4.126613315,-4.371866488,-4.609683103,-4.839717759,-5.305292463,-5.720669186,-6.127034843,-6.89496678,-7.620636659,-8.668575414;

0.546701865,0.48540831,0.243096051,-0.060705244,-0.34201059,-0.585670712,-0.799882378,-0.987922816,-1.300225803,-1.550498273,-1.710533372,-1.815212524,-1.896204105,-1.969623619,-2.03843244,-2.107717769,-2.173620724,-2.244190977,-2.313781586,-2.382840183,-2.554034755,-2.713999686,-3.012058451,-3.280523083,-3.530371104,-3.767038055,-3.997240575,-4.219487595,-4.667928036,-5.069808036,-5.460625033,-6.203462014,-6.906587624,-7.910179422;

0.63792701,0.588871044,0.388391131,0.145003422,-0.084843017,-0.289583399,-0.472457246,-0.636051052,-0.91343077,-1.133023273,-1.277152916,-1.37387057,-1.448971445,-1.516674986,-1.580387171,-1.646099049,-1.708673619,-1.776050105,-1.842881389,-1.909265519,-2.073148196,-2.225916535,-2.509836131,-2.76626762,-3.005543449,-3.232752803,-3.454053283,-3.668454489,-4.102681856,-4.492693007,-4.871813605,-5.590693254,-6.267538193,-7.239785263;

0.666221963,0.621117676,0.438430668,0.220398546,0.023214467,-0.157575332,-0.3229416,-0.471722285,-0.729101609,-0.929055424,-1.067387894,-1.157660441,-1.228697516,-1.293198232,-1.354033719,-1.417683369,-1.478360554,-1.5436089,-1.608338813,-1.672636962,-1.831662349,-1.979975319,-2.256762142,-2.507209276,-2.741260492,-2.964761446,-3.18286436,-3.394539972,-3.818892074,-4.202450185,-4.57657389,-5.284119364,-5.937458507,-6.887051501;

0.676457127,0.632224268,0.46021002,0.260544476,0.077381083,-0.086486589,-0.238107644,-0.377296288,-0.620079612,-0.808534597,-0.941389331,-1.027042954,-1.095322738,-1.157659553,-1.216569154,-1.278851658,-1.33825047,-1.402107804,-1.465424861,-1.528348238,-1.684089769,-1.829378911,-2.101532964,-2.347961747,-2.580148672,-2.801176791,-3.01655056,-3.225841874,-3.645236242,-4.023672654,-4.389978882,-5.084142666,-5.731625318,-6.659644202;

0.678712933,0.63414495,0.472935926,0.290861055,0.126000473,-0.02656381,-0.165321148,-0.2904172,-0.517445163,-0.695747378,-0.823286045,-0.904162226,-0.969635982,-1.029852043,-1.086845169,-1.147766875,-1.205888413,-1.268383086,-1.330361678,-1.391921463,-1.544453847,-1.686815192,-1.954370078,-2.199152027,-2.428017019,-2.646879187,-2.859851487,-3.066997995,-3.478115748,-3.855621908,-4.218093661,-4.899960941,-5.54213551,-6.469544308;

-1.089411062,-1.302445046,-2.236084847,-3.007515595,-3.575200737,-4.011127133,-4.368413217,-4.660755892,-5.122754237,-5.48550226,-5.724511953,-5.876858709,-5.996936528,-6.103409992,-6.202720163,-6.295890991,-6.384141769,-6.481292867,-6.576501288,-6.669947937,-6.895936932,-7.10481821,-7.494658851,-7.845590093,-8.170913622,-8.477829128,-8.78415743,-9.079184235,-9.707908045,-10.24044314,-10.77872186,-11.77210701,-12.72439208,-14.07763321;

-0.542758725,-0.706740698,-1.496875751,-2.228973756,-2.770439022,-3.183851819,-3.524957072,-3.80527216,-
4.248662954,-4.597927251,-4.826391211,-4.973720804,-5.089855116,-5.193212058,-5.289452339,-5.379964012,-
5.465932911,-5.560249868,-5.652491458,-5.743053953,-5.962884044,-6.166606552,-6.546530704,-6.889487643,-
7.207281066,-7.507489729,-7.805950716,-8.09364304,-8.697738296,-9.229974741,-9.734944218,-10.70539145,-
11.62967297,-12.94412794;

-0.121209608,-0.25068574,-0.882556842,-1.550462902,-2.058681176,-2.449282712,-2.77153738,-3.038581895,-
3.463906092,-3.800284082,-4.019061678,-4.161603797,-4.273929576,-4.373663375,-4.46668341,-4.554277036,-
4.637489808,-4.728593137,-4.817841272,-4.905561477,-5.119233005,-5.317721563,-5.687653035,-6.021785679,-
6.331724652,-6.624678032,-6.914472264,-7.193013566,-7.777622102,-8.283842722,-8.776622013,-9.713565224,-
10.59923668,-11.86825507;

0.202579726,0.097033876,-0.381045832,-0.958128864,-1.422802103,-1.784787272,-2.086562277,-2.338584644,-
2.741190108,-3.060731158,-3.267733987,-3.403410341,-3.510226139,-3.605602181,-3.694730839,-3.779892378,-
3.860861597,-3.949247258,-4.035779811,-4.121043243,-4.329273555,-4.522758125,-4.883631553,-5.211450691,-
5.51553318,-5.803150705,-6.081225588,-6.349392812,-6.91246696,-7.396436463,-7.873592279,-8.772737063,-
9.627400943,-10.83728939;

0.44186101,0.354890825,-0.015095795,-0.494455897,-0.902228473,-1.229886674,-1.507007642,-1.741558322,-
2.120177064,-2.422190863,-2.617479619,-2.744831703,-2.844932634,-2.934379519,-3.018146401,-3.09968183,-
3.177179372,-3.261928285,-3.345045506,-3.426938521,-3.628527843,-3.816790565,-4.167759744,-4.485750072,-
4.781650036,-5.061702357,-5.333200373,-5.595158765,-6.135738706,-6.606875539,-7.065336183,-7.931463842,-
8.746856389,-9.904434949;

0.624199441,0.552766337,0.257257103,-0.13229458,-0.478585942,-0.767152618,-1.016372363,-1.230677918,-
1.581563066,-1.862952301,-2.045287036,-2.162924577,-2.255401184,-2.337463158,-2.415295957,-2.493008095,-
2.567078532,-2.64808459,-2.727831093,-2.80647023,-3.001385399,-3.184211758,-3.525095226,-3.833457192,-
4.12059754,-4.392610556,-4.65638463,-4.911823978,-5.429960391,-5.887834438,-6.333643594,-7.166394756,-
7.944760411,-9.058366488;

0.770354896,0.70906253,0.463173419,0.150919575,-0.138109827,-0.386875925,-0.605316614,-0.79658709,-
1.113976879,-1.369008622,-1.536715173,-1.64251334,-1.72774483,-1.805925697,-1.879618487,-1.954149623,-
2.025358923,-2.103001807,-2.179456269,-2.254943167,-2.441691975,-2.616726416,-2.943742258,-3.239304314,-
3.515619836,-3.778116434,-4.033630244,-4.280509872,-4.779162961,-5.222006155,-5.650017273,-6.453477784,-
7.206136689,-8.270167616;

0.85631285,0.808536873,0.604519089,0.355721767,0.119256656,-0.090483131,-0.276938991,-0.443425163,-
0.725426194,-0.949214451,-1.099549794,-1.198024111,-1.277008791,-1.348950049,-1.417093693,-1.487700491,-
1.555258679,-1.62917674,-1.702344404,-1.774696017,-1.953123573,-2.120083299,-2.432099416,-2.714901502,-
2.979874808,-3.232209512,-3.478196999,-3.716876143,-4.199761497,-4.628999212,-5.043371694,-5.819451945,-
6.540630657,-7.568358781;

0.884816329,0.840976249,0.654412864,0.428524771,0.228048214,0.043754414,-0.124949407,-0.276325988,-
0.537314667,-0.740642362,-0.885650422,-0.977485815,-1.052042769,-1.120432471,-1.185380718,-1.253647485,-
1.319059992,-1.390510323,-1.46128288,-1.531273068,-1.704267254,-1.866347978,-2.170498375,-2.446755052,-
2.706009799,-2.954650469,-3.19754397,-3.433466573,-3.90517233,-4.326860166,-4.735265498,-5.497066802,-
6.193651875,-7.19043747;

0.895834933,0.853094978,0.677006415,0.468504727,0.280334163,0.114900209,-0.038633837,-0.180283442,-
0.425541041,-0.618276915,-0.757427207,-0.84454208,-0.916101127,-0.982089866,-1.0448932,-1.111605387,-
1.175566867,-1.245403815,-1.314554612,-1.382990659,-1.552303272,-1.711071352,-2.010109809,-2.281984721,-
2.53975896,-2.785819764,-3.02582074,-3.259138658,-3.725133004,-4.140947844,-4.540480465,-5.288361611,-
5.975545333,-6.946218247;

0.897332988,0.854874382,0.68986495,0.496983389,0.326814505,0.171913549,0.032279586,-0.092883073,-
0.321230035,-0.504907047,-0.638370338,-0.720579337,-0.789070532,-0.852675058,-0.913351135,-0.978492312,-
1.041004835,-1.109252705,-1.176854897,-1.243761192,-1.40947566,-1.565066831,-1.859009387,-2.129806343,-
2.383993726,-2.627801554,-2.865217783,-3.096208317,-3.552300229,-3.96663873,-4.361137729,-5.094347803,-
5.768296367,-6.736668005;

-0.755077602,-0.961116609,-1.822426954,-2.571273798,-3.142699452,-3.585623232,-3.947192141,-4.244605941,-
4.716567899,-5.089551887,-5.344410528,-5.507265782,-5.640053343,-5.759448946,-5.871315621,-5.976616776,-
6.07646441,-6.186297624,-6.293429331,-6.398169835,-6.652036979,-6.888062732,-7.328243546,-7.726551596,-
8.097107336,-8.446992423,-8.798390922,-9.135930169,-9.857579459,-10.45084506,-11.05252009,-12.13783792,-
13.16181845,-14.60403289;

-0.244307691,-0.410147801,-1.152091679,-1.865624097,-2.41010567,-2.829143991,-3.174052885,-3.458650533,-
3.91127409,-4.26979845,-4.513120463,-4.670871413,-4.799663321,-4.915855215,-5.024456117,-5.126926935,-
5.224478453,-5.331308601,-5.435236672,-5.536751856,-5.783687041,-6.01382174,-6.442741821,-6.831448921,-

7.192322085,-7.533333035,-7.874371723,-8.201971652,-8.889608721,-9.483367887,-10.0396892,-11.0926137,-12.08203845,-13.47359021;

0.158187815,0.020743196,-0.584966785,-1.238714038,-1.749771143,-2.14547445,-2.472177006,-2.74340849,-3.177333606,-3.521850412,-3.754419668,-3.906974389,-4.031746281,-4.144110446,-4.249270467,-4.348598265,-4.443152267,-4.546416043,-4.647086037,-4.745489557,-4.985689581,-5.209959666,-5.627556795,-6.00567327,-6.35654967,-6.687908475,-7.017795089,-7.333156902,-7.995217868,-8.555344637,-9.09610101,-10.1071291,-11.05248982,-12.38321534;

0.468035802,0.351348192,-0.122332768,-0.69183666,-1.159873342,-1.526134517,-1.832301254,-2.087638582,-2.497494258,-2.82382289,-3.043047469,-3.187765541,-3.306333156,-3.414602673,-3.516069002,-3.61339268,-3.70606601,-3.806755944,-3.904743421,-4.000800922,-4.235522763,-4.454075833,-4.860871517,-5.232722924,-5.577419828,-5.902810956,-6.215464961,-6.515590349,-7.147222886,-7.678235872,-8.197754582,-9.164748572,-10.0693159,-11.31967789;

0.6925287,0.593939395,0.212332529,-0.270473379,-0.68290403,-1.01428354,-1.295635596,-1.532952119,-1.917860586,-2.225749796,-2.432325185,-2.568060688,-2.679367106,-2.780564453,-2.875626565,-2.968701202,-3.058086052,-3.154352277,-3.24807586,-3.339847092,-3.566337854,-3.77875979,-4.174243663,-4.53413865,-4.868448569,-5.183909969,-5.488397376,-5.780809229,-6.381528705,-6.895074444,-7.389178734,-8.315016875,-9.170335624,-10.35892039;

0.864843765,0.782458179,0.467782586,0.065309548,-0.288297602,-0.581316646,-0.833838757,-1.050061191,-1.405038676,-1.690255506,-1.881763696,-2.006261541,-2.108601267,-2.201701098,-2.289902677,-2.378303074,-2.462724976,-2.55457589,-2.644476107,-2.732565404,-2.951530863,-3.157750048,-3.54141216,-3.889520398,-4.213068113,-4.518530244,-4.813301012,-5.097333176,-5.66873662,-6.165310691,-6.643367424,-7.524657056,-8.334020142,-9.460165831;

1.004110263,0.933184826,0.667334016,0.340268888,0.042465245,-0.211164048,-0.433233093,-0.62681138,-0.947417399,-1.205976005,-1.381344969,-1.493103818,-1.586893423,-1.674236243,-1.75690189,-1.841419499,-1.92236187,-2.010202013,-2.096272407,-2.180732004,-2.390148863,-2.587129411,-2.954142674,-3.286629905,-3.597006416,-3.890747661,-4.17544054,-4.44874201,-4.995577632,-5.473664961,-5.929523657,-6.774761808,-7.543498566,-8.600827444;

1.087417484,1.031497062,0.808419238,0.547020977,0.301771354,0.086677904,-0.103041777,-0.271785361,-0.55687247,-0.784862143,-0.941586749,-1.044940498,-1.131357441,-1.211537467,-1.287846,-1.367983153,-1.444860074,-1.528340657,-1.610487202,-1.691261434,-1.890773948,-2.078070671,-2.428306814,-2.746237477,-3.043393165,-3.325196601,-3.598463889,-3.861835813,-4.388991167,-4.849759478,-5.289688814,-6.096804475,-6.807472277,-7.832580152;

1.116378018,1.065180906,0.860567956,0.623002927,0.415251898,0.226070183,0.054145435,-0.099417487,-0.36299748,-0.570736163,-0.721617565,-0.818163789,-0.899714535,-0.975824647,-1.048469221,-1.125972718,-1.200464026,-1.281167719,-1.360662729,-1.438863144,-1.632467698,-1.814578914,-2.15631848,-2.467096503,-2.757998962,-3.035548815,-3.305175343,-3.565439558,-4.080271638,-4.532659303,-4.965081436,-5.745622746,-6.432863655,-7.441178111;

1.129033068,1.079790324,0.886202718,0.665864227,0.470142258,0.300601037,0.144268062,0.000446101,-0.246982125,-0.444837755,-0.589512374,-0.681259951,-0.759545103,-0.832896108,-0.90305891,-0.978820086,-1.051670634,-1.130535599,-1.208231916,-1.284748795,-1.474324005,-1.652895064,-1.989146163,-2.295072367,-2.584320162,-2.859043318,-3.125513965,-3.382815098,-3.891785851,-4.33706117,-4.76223412,-5.519506108,-6.200365009,-7.191359546;

1.126637698,1.079593581,0.89939954,0.695912644,0.517900273,0.358688633,0.216712041,0.090278511,-0.139657045,-0.329452833,-0.468079303,-0.5548162,-0.6297124,-0.700305768,-0.7680387,-0.842000007,-0.913220688,-0.990257608,-1.066231822,-1.14107506,-1.326670808,-1.501836373,-1.83250817,-2.137268843,-2.422355692,-2.694793462,-2.958407004,-3.212899847,-3.711037673,-4.154257202,-4.572114925,-5.3077134,-5.983878503,-6.974440013;

-0.552618032,-0.747592256,-1.563683135,-2.311060336,-2.895300449,-3.35088544,-3.721946447,-4.028283132,-4.515832764,-4.902770537,-5.172713968,-5.339047682,-5.477471581,-5.603206307,-5.721456412,-5.833057476,-5.939116225,-6.05794989,-6.173599295,-6.28639068,-6.560505781,-6.815987191,-7.291445399,-7.722408863,-8.123434874,-8.501764277,-8.883636511,-9.249227568,-10.02111557,-10.64965555,-11.28943327,-12.42940627,-13.47827041,-14.9480696;

-0.05785856,-0.217073354,-0.927286688,-1.639900456,-2.195379186,-2.625573677,-2.979106245,-3.271672387,-3.73865692,-4.109889752,-4.367055187,-4.528091673,-4.662267444,-4.784629107,-4.899314662,-5.007824995,-5.111379218,-5.226718899,-5.338691252,-5.447754111,-5.713754336,-5.962161863,-6.424204507,-6.843295452,-7.232249533,-7.599316537,-7.968179034,-8.321248654,-9.054278688,-9.682580997,-10.26981835,-11.36999294,-12.37977639,-13.77555917;

0.333084985,0.197222693,-0.38740401,-1.041306179,-1.562027523,-1.967765006,-2.303043166,-2.581709342,-3.028818065,-3.384643646,-3.629758913,-3.785276354,-3.915147957,-4.033394184,-4.1443812,-4.24950438,-4.349803472,-4.461075813,-4.569360374,-4.674894427,-4.933143441,-5.174664249,-5.623453418,-6.029926711,-6.406693112,-6.7618601,-7.117048966,-7.455461333,-8.157378772,-8.750418067,-9.318378909,-10.37155499,-11.3359342,-12.66835877;

0.631934235,0.513641742,0.049777977,-0.520901239,-0.997783981,-1.373002992,-1.687264774,-1.94917018,-2.370830201,-2.707153196,-2.937539761,-3.084680652,-3.207944131,-3.322334077,-3.429816281,-3.533132162,-3.631706908,-3.740286275,-3.845712218,-3.948797477,-4.201090806,-4.435957566,-4.8718855,-5.271390994,-5.641220101,-5.989510953,-6.32308369,-6.642219119,-7.308154304,-7.867035402,-8.410768839,-9.418278013,-10.32563837,-11.57278374;

0.846410801,0.744829534,0.360914942,-0.128620688,-0.54999431,-0.889487906,-1.178455675,-1.4217052,-1.817339263,-2.134206238,-2.351060792,-2.488878725,-2.604573239,-2.711152997,-2.811575684,-2.910129084,-3.00536761,-3.10878398,-3.209213625,-3.30724411,-3.549878106,-3.777723797,-4.200894852,-4.586508002,-4.944000935,-5.280409959,-5.604357468,-5.914521838,-6.54451347,-7.082798914,-7.596036519,-8.555943747,-9.400664956,-10.59154465;

1.012016688,0.926113598,0.602571776,0.188099242,-0.175085506,-0.475631224,-0.734442245,-0.955524266,-1.319128906,-1.611435631,-1.811568016,-1.937284451,-2.043315182,-2.141269838,-2.234300079,-2.327702961,-2.417136651,-2.51562784,-2.611836364,-2.70580457,-2.940175329,-3.161131156,-3.57117541,-3.943293959,-4.288495308,-4.613477516,-4.926232188,-5.226393893,-5.823798582,-6.341476758,-6.836588177,-7.741698738,-8.518142609,-9.652952553;

1.14479386,1.07069839,0.794527835,0.455550556,0.148864189,-0.111395991,-0.339120228,-0.537173217,-0.864972396,-1.129616843,-1.312310585,-1.424461033,-1.521255424,-1.612721606,-1.699647058,-1.788820481,-1.87446756,-1.968455495,-2.060417799,-2.150390619,-2.374294569,-2.585131193,-2.976866389,-3.331603235,-3.662214417,-3.974157594,-4.275577005,-4.563782066,-5.13426473,-5.631483624,-6.102454889,-6.951871524,-7.690790361,-8.782262156;

1.222973994,1.165395035,0.932525747,0.662337152,0.409303567,0.188153845,-0.006153201,-0.178674777,-0.469908555,-0.703065889,-0.865521491,-0.969394707,-1.058558921,-1.142508243,-1.222787349,-1.307416323,-1.38885128,-1.478039745,-1.565625842,-1.651521172,-1.864519833,-2.064751462,-2.438729801,-2.777850713,-3.094178469,-3.393216673,-3.682177794,-3.9594517,-4.508174511,-4.986259983,-5.440070097,-6.230032267,-6.939659106,-8.013199888;

1.252543015,1.199849102,0.985299222,0.737652737,0.525179349,0.331492146,0.155470404,-0.001405642,-0.269951672,-0.48185059,-0.638508937,-0.735567946,-0.819754372,-0.899435969,-0.97588353,-1.057790494,-1.136777708,-1.222991499,-1.307749839,-1.390926534,-1.597706638,-1.792544678,-2.157662006,-2.489235783,-2.798932289,-3.093561591,-3.378638958,-3.65247366,-4.187915025,-4.657083028,-5.102388902,-5.864432703,-6.562395499,-7.619752366;

1.266700068,1.21634896,1.012750547,0.781541486,0.579882103,0.407790435,0.248797303,0.101941402,-0.149220508,-0.351698983,-0.50171696,-0.593994748,-0.674839038,-0.751613079,-0.825447806,-0.905545844,-0.982833282,-1.067065547,-1.149905055,-1.231293108,-1.433819915,-1.624988913,-1.984372787,-2.31084506,-2.619080868,-2.910744605,-3.192405424,-3.462918329,-3.991865413,-4.453492784,-4.889355074,-5.634724481,-6.329262839,-7.37050092;

1.263531147,1.216368644,1.026941877,0.811182021,0.627056694,0.464883556,0.321486737,0.1943684,-0.037699604,-0.233033144,-0.376548348,-0.463807975,-0.541146673,-0.614988371,-0.686260901,-0.764462733,-0.840061809,-0.922302732,-1.003276623,-1.082885874,-1.28118503,-1.468804825,-1.822323705,-2.147957214,-2.451756833,-2.740893748,-3.01938107,-3.286603706,-3.803550505,-4.262598508,-4.684022583,-5.41795803,-6.113188348,-7.152689059;

-0.336622989,-0.537044436,-1.334237661,-2.045940926,-2.630557561,-3.090652358,-3.463302974,-3.771009485,-4.260916442,-4.650929611,-4.931427798,-5.125597875,-5.288052698,-5.435914368,-5.574367443,-5.704617115,-5.827903824,-5.960367518,-6.088697361,-6.213441371,-6.514323079,-6.793682629,-7.306038706,-7.76506892,-8.187223218,-8.58080907,-8.968104488,-9.334881724,-10.07195758,-10.68368985,-11.28815986,-12.34473028,-13.27480624,-14.60757968;

0.113565922,-0.052585089,-0.737665017,-1.424971714,-1.980496897,-2.412659718,-2.766940864,-3.060004127,-3.52862339,-3.902248316,-4.169326453,-4.356588909,-4.513696524,-4.657018758,-4.790853605,-4.916967097,-5.036728071,-5.164939585,-5.28886011,-5.409118784,-5.700044601,-5.970382923,-6.465411744,-6.908676785,-7.31483773,-7.693196209,-8.063371694,-8.41372369,-9.107629377,-9.708327653,-10.26287339,-11.25746135,-12.13111002,-13.40781315;

0.471876379,0.328189287,-0.237075431,-0.870532701,-1.391301556,-1.798293954,-2.134928471,-2.414306854,-2.862676993,-3.220151062,-3.474304636,-3.654767049,-3.806491281,-3.944706256,-4.073851069,-4.195609792,-4.311177347,-4.434532955,-4.554022247,-4.670017747,-4.951363202,-5.212930195,-5.691134107,-6.118282612,-

6.508641439,-6.871527332,-7.224231719,-7.556681682,-8.216506889,-8.779856736,-9.309712845,-10.24207648,-11.08013467,-12.29462214;

0.750789845,0.624366888,0.167237673,-0.388564161,-0.865757604,-1.241364772,-1.556924877,-1.819114035,-2.241502029,-2.578936326,-2.817095004,-2.987974556,-3.13200727,-3.264492307,-3.388357729,-3.506811784,-3.619218887,-3.738459352,-3.853689787,-3.965739547,-4.237924326,-4.490468494,-4.9520379,-5.366117474,-5.743921093,-6.094571922,-6.426610191,-6.740495497,-7.362753062,-7.892499944,-8.391273191,-9.268681649,-10.05399709,-11.18805251;

0.948138407,0.839263252,0.454518948,-0.026504197,-0.447117811,-0.786255008,-1.076489404,-1.320116724,-1.714500189,-2.030630451,-2.25281521,-2.412383121,-2.546983009,-2.67050199,-2.786217127,-2.898993381,-3.005999951,-3.119399023,-3.229129253,-3.335777752,-3.597130331,-3.840825491,-4.28646784,-4.683601132,-5.045913046,-5.381576292,-5.700270824,-6.001114159,-6.58674932,-7.091322205,-7.558276943,-8.383530999,-9.115208032,-10.19842093;

1.1025296,1.010339713,0.682583372,0.273113139,-0.08745326,-0.385553887,-0.643542892,-0.863622127,-1.225476016,-1.517127434,-1.721889968,-1.868813688,-1.993039631,-2.10591945,-2.213108943,-2.320515622,-2.422656553,-2.530620814,-2.635516289,-2.737528381,-2.989292109,-3.224924087,-3.654875618,-4.036226107,-4.383950926,-4.70584784,-5.011142444,-5.299734452,-5.851941967,-6.332616152,-6.771821784,-7.537618976,-8.225907879,-9.261230555;

1.234578074,1.154828116,0.872837983,0.541117249,0.23669271,-0.022201979,-0.249659031,-0.447459122,-0.774448327,-1.039753257,-1.226897048,-1.358341601,-1.472571677,-1.579749487,-1.680755793,-1.783364618,-1.881188193,-1.984204661,-2.084483963,-2.182136984,-2.422238256,-2.646433668,-3.055638242,-3.417859771,-3.749423662,-4.056825122,-4.349351304,-4.624812104,-5.149699599,-5.607333284,-6.019349769,-6.73737508,-7.405798946,-8.402071027;

1.311982221,1.249839428,1.011802442,0.749713968,0.498918819,0.278497859,0.084312339,-0.088356177,-0.379706579,-0.614332736,-0.780300854,-0.904551245,-1.010904062,-1.109815605,-1.203489806,-1.301168943,-1.394434346,-1.49235528,-1.587959821,-1.681251327,-1.909547446,-2.122207228,-2.512246297,-2.858011828,-3.174465335,-3.468094382,-3.747524993,-4.011793436,-4.517000641,-4.95563498,-5.343998006,-6.026546811,-6.671617148,-7.651732942;

1.342193639,1.284943221,1.065529371,0.824024294,0.61675233,0.424892057,0.248846541,0.09171591,-0.177051745,-0.390872788,-0.552422229,-0.669344512,-0.770270329,-0.864515655,-0.954022183,-1.048826995,-1.139517305,-1.234404574,-1.327123388,-1.417632123,-1.639593473,-1.846785648,-2.227927024,-2.566298841,-2.876226379,-3.166088278,-3.442408918,-3.703335837,-4.197020785,-4.627124014,-5.000622002,-5.670152099,-6.306960612,-7.273400632;

1.356293497,1.301624025,1.093217158,0.868213692,0.670582293,0.502047716,0.344347667,0.197262179,-0.05367355,-0.259921076,-0.414803905,-0.526569457,-0.623821082,-0.714884511,-0.801530075,-0.894413141,-0.983315459,-1.076164566,-1.166906134,-1.255591923,-1.473199002,-1.676693054,-2.052121408,-2.385507676,-2.694855458,-2.981927072,-3.254807801,-3.512443002,-4.000979119,-4.423371798,-4.794734121,-5.44892641,-6.082065918,-7.034818314;

1.351275704,1.300425043,1.106587972,0.896411093,0.717104791,0.55810148,0.416968034,0.29183843,0.059637517,-0.140333954,-0.288660502,-0.395081907,-0.488526503,-0.576398502,-0.660242365,-0.751119408,-0.838240309,-0.929018588,-1.017851918,-1.104688923,-1.317911246,-1.517788868,-1.887291695,-2.220740323,-2.525628029,-2.810010035,-3.07951008,-3.33382825,-3.813236575,-4.230427745,-4.588490329,-5.239764926,-5.873161534,-6.823008307;

-0.277831715,-0.478936514,-1.279763574,-2.000285978,-2.593922943,-3.061984419,-3.441253208,-3.754762848,-4.254355222,-4.652651087,-4.939762288,-5.137188447,-5.303661881,-5.455485557,-5.597653612,-5.731391287,-5.857997954,-5.994344295,-6.126245251,-6.254256423,-6.563138303,-6.849810068,-7.372999526,-7.840772049,-8.269807168,-8.668611158,-9.062061436,-9.433203284,-10.16401737,-10.77459721,-11.38049478,-12.42758069,-13.36207409,-14.72319032;

0.155543647,-0.012502657,-0.698728539,-1.393832909,-1.957356097,-2.39688586,-2.757441148,-3.055878452,-3.533584181,-3.914829751,-4.187767499,-4.377749989,-4.538372975,-4.685124184,-4.822172104,-4.951299988,-5.073919157,-5.205483913,-5.332502765,-5.455563103,-5.75335927,-6.029922368,-6.533821714,-6.983897347,-7.395111112,-7.776978408,-8.151609461,-8.504823725,-9.191867005,-9.791231912,-10.34292616,-11.32838076,-12.21462627,-13.51714005;

0.501597596,0.354093648,-0.212278767,-0.852824602,-1.380686665,-1.794533472,-2.137377069,-2.421896524,-2.878663026,-3.242971246,-3.502233889,-3.684987141,-3.839832606,-3.981150871,-4.113179908,-4.237633359,-4.355741773,-4.482019776,-4.604182968,-4.722569608,-5.009803865,-5.276636669,-5.762105388,-6.194541514,-6.588472577,-6.953464334,-7.309152442,-7.643261252,-8.2942307,-8.857252074,-9.38014214,-10.30572808,-11.16001888,-12.39828699;

0.773899619,0.642432921,0.18136957,-0.380492287,-0.863784306,-1.245588394,-1.566910687,-1.83375217,-2.263753125,-2.60726072,-2.849685932,-3.022397988,-3.169165209,-3.304654997,-3.43129167,-3.552381045,-3.667255455,-3.78921298,-3.906927715,-4.021225327,-4.29881067,-4.555921195,-5.023336881,-5.442034503,-5.822706165,-6.174738122,-6.507161214,-6.820321448,-7.431689608,-7.959398008,-8.448457705,-9.324624996,-10.12434475,-11.28167033;

0.9661582,0.851654112,0.4599051,-0.026465519,-0.452417787,-0.797283994,-1.093040148,-1.341169502,-1.743034285,-2.065103355,-2.291252293,-2.45257683,-2.589673147,-2.71564956,-2.833648924,-2.948650645,-3.057923656,-3.173510185,-3.285221486,-3.393612796,-3.659391437,-3.907060431,-4.357551874,-4.757982794,-5.121895778,-5.457747401,-5.775965632,-6.075450759,-6.649124122,-7.148889486,-7.604657639,-8.431283633,-9.176517843,-10.28298662;

1.118481607,1.020557736,0.684462453,0.270643578,-0.094350306,-0.397363379,-0.660203765,-0.884416978,-1.252977625,-1.549877852,-1.758072691,-1.906566045,-2.032964561,-2.147968466,-2.257162672,-2.366627621,-2.47067834,-2.580537942,-2.68713211,-2.790629132,-3.046305696,-3.285500931,-3.719577154,-4.103155327,-4.451451434,-4.772519996,-5.07638689,-5.362599456,-5.902587014,-6.374085049,-6.804868054,-7.573813784,-8.276679329,-9.334358342;

1.247886335,1.163115281,0.873624757,0.53928987,0.231898462,-0.030615075,-0.261775518,-0.462890913,-0.795458468,-1.065132802,-1.255035005,-1.387537416,-1.50370667,-1.612872722,-1.715743183,-1.820345029,-1.920023449,-2.024780668,-2.126617427,-2.225627859,-2.469142673,-2.6963429,-3.108656731,-3.472021238,-3.803221639,-4.108956326,-4.399252854,-4.671664657,-5.183969122,-5.631721294,-6.034104986,-6.762904939,-7.445508085,-8.463376697;

1.321962806,1.256833681,1.012950985,0.751234223,0.498959306,0.276098027,0.079373849,-0.095710924,-0.391304136,-0.628825343,-0.796472583,-0.922107542,-1.030365344,-1.131139387,-1.226580292,-1.326283514,-1.421435152,-1.521003943,-1.618050427,-1.712593528,-1.943906797,-2.159142701,-2.55159073,-2.897704736,-3.21312264,-3.504485433,-3.781092259,-4.041876543,-4.534375222,-4.96272581,-5.34516684,-6.042423743,-6.700767618,-7.702758088;

1.35226774,1.292186293,1.067026785,0.825234614,0.6194265,0.426721439,0.248744046,0.089705246,-0.182385255,-0.398020824,-0.561641693,-0.679964744,-0.782765279,-0.878857296,-0.970140366,-1.067065883,-1.159747174,-1.256312266,-1.350477619,-1.442252839,-1.667241466,-1.876999702,-2.260512782,-2.599083432,-2.907818672,-3.195431055,-3.468935505,-3.726168765,-4.206805535,-4.626903329,-4.997923312,-5.68201662,-6.331972191,-7.320166869;

1.367054998,1.309702475,1.095520633,0.870105978,0.673289314,0.505848304,0.347685997,0.199064311,-0.054116878,-0.262349642,-0.419101898,-0.532282009,-0.631367883,-0.724254039,-0.812660041,-0.907727941,-0.998683525,-1.093221205,-1.185424375,-1.275372128,-1.49603214,-1.702110821,-2.079914883,-2.413446706,-2.721864871,-3.006650862,-3.276568372,-3.530360344,-4.006057924,-4.417886819,-4.790873405,-5.458388014,-6.104651812,-7.07981241;

1.362412874,1.309229088,1.109815647,0.897944944,0.720061726,0.56214001,0.421876012,0.297432075,0.064294059,-0.138001101,-0.2879679,-0.395821208,-0.491040427,-0.580702374,-0.666291018,-0.75939803,-0.848634003,-0.941104522,-1.031385433,-1.119484686,-1.335759601,-1.538235045,-1.910073996,-2.244045501,-2.547883528,-2.8297852,-3.096141414,-3.346412279,-3.81358762,-4.219511888,-4.58240128,-5.246983966,-5.893842116,-6.866626752;

-0.284387861,-0.479451148,-1.283930553,-2.018428788,-2.625844267,-3.106413937,-3.495936547,-3.818111423,-4.332086196,-4.74206493,-5.038871794,-5.23758354,-5.405402697,-5.558978595,-5.702691352,-5.837812713,-5.965741912,-6.105487416,-6.240498226,-6.371297733,-6.686199286,-6.977635386,-7.504943024,-7.974396962,-8.403034356,-8.799859744,-9.191181835,-9.55883949,-10.25757393,-10.85406311,-11.45453329,-12.49889933,-13.44784521,-14.8380346;

0.129650736,-0.032906749,-0.71607403,-1.422783681,-1.998012757,-2.448700918,-2.818439914,-3.124648275,-3.615357826,-4.007073758,-4.288602919,-4.47916086,-4.640469695,-4.788305272,-4.926285804,-5.056210882,-5.179563405,-5.31372042,-5.443151239,-5.568365414,-5.870637998,-6.150519743,-6.656109465,-7.105718865,-7.514824454,-7.893232593,-8.264534235,-8.613321969,-9.2710488,-9.85962172,-10.4042856,-11.39137216,-12.29673722,-13.63241059;

0.463623818,0.319410863,-0.241255129,-0.891104655,-1.428710815,-1.852409806,-2.203762501,-2.4953937,-2.963700563,-3.337172022,-3.603937369,-3.786793398,-3.941848141,-4.083880897,-4.216492775,-4.341413549,-4.459935454,-4.588190623,-4.712133238,-4.832079929,-5.1223622,-5.391215308,-5.876459647,-6.30690232,-6.697451939,-7.057956843,-7.40941619,-7.738702808,-8.360913967,-8.915694518,-9.431112774,-10.36353823,-11.24118072,-12.51143348;

0.729048236,0.599249001,0.141908737,-0.426544631,-0.917744191,-1.308322067,-1.637218117,-1.910503183,-2.350998776,-2.70272718,-2.951768448,-3.124370079,-3.271116816,-3.407165708,-3.534228294,-3.655548494,-3.770602255,-3.894037337,-4.013114369,-4.128606824,-4.408278489,-4.666419949,-5.131916237,-5.547497922,-

5.923868477,-6.27061956,-6.597385975,-6.904384282,-7.487755231,-8.007251604,-8.490968154,-9.381683337,-10.20279361,-11.39454872;

0.917169741,0.802988005,0.412088944,-0.078964389,-0.511452348,-0.864343563,-1.167183388,-1.421469426,-1.833624992,-2.163732531,-2.396144844,-2.55763302,-2.694712049,-2.820977955,-2.939172961,-3.054059275,-3.163269384,-3.279691797,-3.392194063,-3.501241665,-3.767929696,-4.015791032,-4.463012998,-4.859057711,-5.217532267,-5.547173732,-5.859040757,-6.151937235,-6.699215784,-7.190782426,-7.642645528,-8.486272929,-9.255028925,-10.39780421;

1.067140612,0.968548935,0.631748671,0.215725166,-0.154267231,-0.464197822,-0.73363286,-0.963786138,-1.342026843,-1.646416155,-1.860553373,-2.009543894,-2.136020054,-2.251323524,-2.360714628,-2.470016217,-2.573850854,-2.684137596,-2.791084143,-2.894842989,-3.150552088,-3.389192358,-3.818937075,-4.19699983,-4.538859061,-4.852822052,-5.149578258,-5.428358003,-5.943924506,-6.407158333,-6.836015209,-7.626678941,-8.354405176,-9.450284067;

1.193489406,1.107994542,0.817406437,0.481285282,0.170195655,-0.097693732,-0.334043478,-0.539971459,-0.881037992,-1.157134801,-1.352319734,-1.485321466,-1.601755035,-1.71134333,-1.81452352,-1.918972787,-2.018434972,-2.123345195,-2.225285047,-2.32432639,-2.567266237,-2.793343535,-3.200420522,-3.557437638,-3.881478064,-4.179490392,-4.462035856,-4.726680918,-5.21567736,-5.656430058,-6.060093597,-6.814663355,-7.522742614,-8.579344411;

1.263768139,1.198743141,0.95346131,0.690464158,0.435479428,0.208719066,0.008021703,-0.170791243,-0.473144801,-0.715191436,-0.887145922,-1.013617556,-1.122390604,-1.223749034,-1.319647574,-1.419305426,-1.514333288,-1.613875497,-1.710835348,-1.80521322,-2.035506849,-2.249257056,-2.635643201,-2.974598271,-3.282284491,-3.565478792,-3.833947475,-4.086701402,-4.55726995,-4.980005754,-5.367883706,-6.093052832,-6.7757033,-7.818486819;

1.293396288,1.233172589,1.005624371,0.761175052,0.554255475,0.358955706,0.177736744,0.015554275,-0.262256554,-0.48114347,-0.649106637,-0.768307207,-0.871753766,-0.968554636,-1.060432773,-1.157444059,-1.250128017,-1.346637753,-1.440649778,-1.532208508,-1.756036502,-1.96411031,-2.341184171,-2.672171995,-2.972790528,-3.252007945,-3.517172247,-3.765992547,-4.225526792,-4.640478872,-5.018860683,-5.731058445,-6.405491614,-7.434400395;

1.307905892,1.250376207,1.03319445,0.804501225,0.605851239,0.437191558,0.276987018,0.125663036,-0.132194539,-0.343293817,-0.504115095,-0.618191695,-0.717990038,-0.811661649,-0.900741969,-0.995982981,-1.087029079,-1.181485905,-1.273511186,-1.363191838,-1.582628357,-1.786942659,-2.158063951,-2.483786492,-2.784022512,-3.060191523,-3.321525672,-3.566694237,-4.021548486,-4.429344088,-4.813847866,-5.506450303,-6.177236849,-7.192968329;

1.305194368,1.251710153,1.04809955,0.830508832,0.650738911,0.491622256,0.350412739,0.224892371,-0.011320241,-0.216486822,-0.370207448,-0.479000018,-0.574997175,-0.665524033,-0.751855508,-0.845216466,-0.934628897,-1.026990649,-1.117040275,-1.204825493,-1.419801805,-1.620399986,-1.985308231,-2.311384321,-2.606811805,-2.879658655,-3.137148137,-3.378540676,-3.825773624,-4.228910184,-4.60521906,-5.294080419,-5.966574828,-6.979140386;

-0.314546338,-0.507892016,-1.334756306,-2.077413696,-2.69046452,-3.175424848,-3.56880249,-3.893400971,-4.410332723,-4.822064597,-5.118649231,-5.32320124,-5.495531183,-5.652657066,-5.799350451,-5.937040779,-6.067237251,-6.207812016,-6.343615763,-6.475012332,-6.790877282,-7.082409704,-7.606757757,-8.071455558,-8.493861727,-8.883462386,-9.265963074,-9.624336213,-10.28951722,-10.87060362,-11.45298763,-12.47738362,-13.41434411,-14.79074111;

0.082806473,-0.076351386,-0.769928076,-1.48420976,-2.064395721,-2.518801209,-2.892024538,-3.200461542,-3.693889722,-4.087233211,-4.368372757,-4.564067422,-4.7293354,-4.880148936,-5.020634029,-5.152658483,-5.277763281,-5.412355846,-5.542255109,-5.667786693,-5.970301208,-6.249549155,-6.750904422,-7.194591401,-7.596582154,-7.966968679,-8.328790479,-8.667780435,-9.2939907,-9.86411108,-10.39452623,-11.36449502,-12.25859871,-13.58057458;

0.405411413,0.264646895,-0.297158766,-0.952288146,-1.49394773,-1.920888162,-2.275435934,-2.569231162,-3.040021051,-3.414990257,-3.681258507,-3.868806669,-4.027390613,-4.172079852,-4.306888631,-4.433622152,-4.553625264,-4.682086506,-4.80622594,-4.926229722,-5.216085797,-5.483679313,-5.963796687,-6.387645676,-6.770509827,-7.122547141,-7.464083674,-7.783482261,-8.375067861,-8.913430696,-9.414656248,-10.33171713,-11.19754178,-12.45386283;

0.663370646,0.536773722,0.083530744,-0.486250751,-0.980407968,-1.37384422,-1.705641935,-1.981108033,-2.424275834,-2.777695298,-3.026367237,-3.203512089,-3.353617374,-3.491887644,-3.620741866,-3.743387748,-3.859470749,-3.982725275,-4.101666358,-4.216867798,-4.495343053,-4.751686972,-5.211305376,-5.619033692,-5.986690922,-6.32411525,-6.641730283,-6.939486027,-7.494258219,-7.998896694,-8.469833831,-9.3468117,-10.15627702,-11.3358433;

0.846676899,0.735108471,0.350329218,-0.138580601,-0.572436362,-0.927740547,-1.23327174,-1.489859003,-1.904904329,-2.236923761,-2.469068248,-2.634907736,-2.775061889,-2.90342537,-3.023306962,-3.139261236,-3.24896467,-3.365069584,-3.477365914,-3.586098208,-3.851379592,-4.097113886,-4.537922964,-4.925726529,-5.275163761,-5.595278186,-5.897625263,-6.180953283,-6.701733268,-7.179224257,-7.620572162,-8.451993762,-9.21058428,-10.34120045;

0.993677702,0.897056475,0.56703286,0.156544077,-0.212802735,-0.524059764,-0.795760453,-1.028149035,-1.409550422,-1.71624101,-1.930710733,-2.084253874,-2.21386698,-2.331065993,-2.442099358,-2.552483274,-2.657112414,-2.76703694,-2.873639245,-2.976961643,-3.230987307,-3.467270492,-3.890210152,-4.259759319,-4.592335948,-4.896574974,-5.183679137,-5.45269684,-5.943577597,-6.393835535,-6.813003134,-7.594092758,-8.313437446,-9.397570253;

1.119592141,1.035686998,0.751273389,0.421496891,0.112103358,-0.156366078,-0.394192399,-0.601818985,-0.945909421,-1.224180449,-1.420024692,-1.557346652,-1.67700222,-1.788937469,-1.894017809,-1.999516112,-2.099726272,-2.204271734,-2.305854971,-2.404436152,-2.64561291,-2.86923875,-3.269331719,-3.617793244,-3.932519637,-4.220830845,-4.493631234,-4.748688098,-5.214405198,-5.64336114,-6.03995334,-6.785721612,-7.485729304,-8.530367278;

1.189176417,1.125621418,0.885983759,0.629040795,0.376217635,0.149593166,-0.051983983,-0.232007835,-0.536758365,-0.780133843,-0.95274458,-1.083909205,-1.196055236,-1.299853217,-1.397718199,-1.49839868,-1.594150067,-1.693379822,-1.790015087,-1.883945967,-2.112524713,-2.323886757,-2.703051198,-3.033323142,-3.331678367,-3.605196963,-3.864004571,-4.107333364,-4.555961641,-4.968213007,-5.350690645,-6.06759915,-6.74214028,-7.772844195;

1.218785444,1.159657843,0.936971263,0.697758242,0.49378755,0.299215967,0.117453634,-0.045641104,-0.32555188,-0.54522696,-0.714319889,-0.838134972,-0.944956044,-1.044274122,-1.138210538,-1.23628573,-1.329728404,-1.425991072,-1.519692445,-1.610816081,-1.832941819,-2.038577273,-2.408318519,-2.730521763,-3.021715774,-3.291330992,-3.546935707,-3.786231659,-4.224625581,-4.628916196,-5.001974497,-5.706235694,-6.372215253,-7.388820612;

1.232839648,1.176170148,0.963394967,0.7399338,0.544055802,0.376779062,0.21664446,0.064649969,-0.194908375,-0.406711554,-0.568727067,-0.687358089,-0.790513292,-0.886761359,-0.977960074,-1.07428994,-1.166130623,-1.260381492,-1.352115816,-1.441361205,-1.659128445,-1.861000015,-2.224704079,-2.54160384,-2.832522525,-3.099040709,-3.350751119,-3.586349662,-4.020371477,-4.417657137,-4.796467511,-5.481565465,-6.143834417,-7.146505704;

1.231811503,1.178678015,0.978328522,0.764669102,0.588107226,0.430478173,0.289882181,0.164530256,-0.072931351,-0.278638771,-0.433597922,-0.546901061,-0.646254357,-0.739422871,-0.827917688,-0.922403322,-1.012644738,-1.104844555,-1.194619857,-1.281968559,-1.495312883,-1.69344371,-2.05088892,-2.368220527,-2.654333429,-2.917318082,-3.165076377,-3.396883074,-3.823937446,-4.216809018,-4.587951622,-5.268938149,-5.932584056,-6.932389949;

-0.405310348,-0.605462172,-1.466863045,-2.219651124,-2.829092503,-3.311206171,-3.703488911,-4.026737742,-4.541207077,-4.950741134,-5.242643821,-5.453033282,-5.631149328,-5.792659994,-5.942989598,-6.083758324,-6.216583238,-6.356536084,-6.491620321,-6.62201903,-6.935086103,-7.223302313,-7.738189146,-8.192689947,-8.604208712,-8.982577115,-9.353663127,-9.700511213,-10.33288364,-10.8981191,-11.4601573,-12.46427581,-13.39148775,-14.75727237;

-0.018752243,-0.183147645,-0.90101087,-1.622573216,-2.200242382,-2.652665986,-3.025264725,-3.33276389,-3.824228605,-4.215728264,-4.492343411,-4.693269728,-4.863706121,-5.018266687,-5.161857304,-5.296448168,-5.423640791,-5.557325039,-5.686293184,-5.810661302,-6.109937448,-6.385433507,-6.876645694,-7.309499169,-7.700203516,-8.05902071,-8.409513374,-8.737099936,-9.333239417,-9.884297354,-10.3999534,-11.35144473,-12.238163,-13.54701344;

0.294843937,0.149401882,-0.428316892,-1.088877353,-1.628404689,-2.053655988,-2.407692517,-2.700809976,-3.169832634,-3.543191128,-3.805115064,-3.997402696,-4.160662539,-4.308750242,-4.446335317,-4.575328415,-4.697138057,-4.824588472,-4.947622378,-5.066299377,-5.352496927,-5.615968156,-6.085622663,-6.498440531,-6.869903137,-7.210321594,-7.540827307,-7.849347234,-8.412343714,-8.933915315,-9.421399385,-10.3226274,-11.1780244,-12.42255694;

0.548640658,0.417565438,-0.046286244,-0.618604395,-1.110503665,-1.502344429,-1.833497639,-2.108466405,-2.550224654,-2.902332786,-3.146900217,-3.328420686,-3.48283605,-3.624250972,-3.755639492,-3.880302388,-3.997963306,-4.120196887,-4.238068003,-4.352002197,-4.626825648,-4.878927408,-5.327896296,-5.724734824,-6.081071228,-6.406959887,-6.713321674,-6.999899176,-7.527519349,-8.01646804,-8.47514741,-9.337550092,-10.13798657,-11.30438111;

0.72726355,0.610985564,0.219263562,-0.267639435,-0.698344984,-1.05222002,-1.357275608,-1.613767325,-2.028983817,-2.361114853,-2.590207454,-2.760494577,-2.904782491,-3.035900768,-3.157975518,-3.275560942,-3.386791747,-3.501687681,-3.612725477,-3.719996276,-3.981276754,-4.222671911,-4.652663702,-5.029547082,-

5.367654907,-5.676287907,-5.967580368,-6.240101582,-6.735507095,-7.197481806,-7.627475436,-8.445652804,-9.195253098,-10.31101723;

0.870853011,0.769323649,0.43463004,0.03091267,-0.334579399,-0.645072415,-0.917338452,-1.15069076,-1.53326872,-1.840848189,-2.053157673,-2.211061715,-2.344520546,-2.464675594,-2.577788054,-2.689462817,-2.794995196,-2.903609574,-3.008812212,-3.110553374,-3.360368556,-3.592199696,-4.004212291,-4.362774722,-4.683910241,-4.976559576,-5.252600138,-5.510751566,-5.977814941,-6.413401025,-6.821863157,-7.590834201,-8.300264373,-9.368500127;

0.995824623,0.90750229,0.619674964,0.294395836,-0.011540029,-0.278885127,-0.516646285,-0.72470228,-1.069914959,-1.348923302,-1.542909869,-1.68501006,-1.80839204,-1.922555002,-2.029347129,-2.136018423,-2.237009478,-2.340325326,-2.4405592,-2.537593121,-2.774479259,-2.993485781,-3.382400248,-3.719720397,-4.022872493,-4.299540892,-4.561188242,-4.805531955,-5.248240873,-5.66402987,-6.050780646,-6.785002482,-7.473902619,-8.501653776;

1.066461605,0.998152051,0.755157473,0.500415523,0.250263699,0.025030315,-0.176274338,-0.356416578,-0.661711305,-0.905508611,-1.076907995,-1.211695186,-1.327033456,-1.432931625,-1.532380418,-1.634126675,-1.730563172,-1.8286871,-1.924069092,-2.0165244,-2.240869935,-2.447609488,-2.815200195,-3.133972426,-3.420615969,-3.682430342,-3.930146041,-4.162734617,-4.588725021,-4.988391152,-5.362430241,-6.068066884,-6.731746674,-7.74460911;

1.094460807,1.030655375,0.804676416,0.569767103,0.366495409,0.172472777,-0.008943359,-0.17209549,-0.45285321,-0.673118963,-0.840494827,-0.967802397,-1.077650533,-1.17902731,-1.2745304,-1.37366755,-1.467793817,-1.563051445,-1.655525241,-1.745206346,-1.963125389,-2.164066541,-2.522056271,-2.832536091,-3.111848645,-3.369377172,-3.613501222,-3.841747151,-4.258093661,-4.649488068,-5.01506732,-5.707984505,-6.362672824,-7.361350502;

1.106933347,1.045571442,0.829559202,0.611228698,0.417369045,0.249598146,0.089093438,-0.062867378,-0.32373298,-0.535340853,-0.695783406,-0.817843453,-0.923901877,-1.022192132,-1.114947475,-1.212343392,-1.304881298,-1.39818893,-1.488740094,-1.576554061,-1.790174505,-1.987332675,-2.339171631,-2.64423253,-2.922740252,-3.177011184,-3.417174025,-3.641724619,-4.054048884,-4.438368991,-4.808701962,-5.483898526,-6.134888233,-7.118983082;

1.105618378,1.047571824,0.843666027,0.636025175,0.461952378,0.30448354,0.163110557,0.036559001,-0.202681271,-0.407279116,-0.560791218,-0.677490519,-0.779632662,-0.874836141,-0.964863087,-1.060431822,-1.151387071,-1.24271708,-1.331344053,-1.417271791,-1.62654255,-1.819939571,-2.165441058,-2.470259983,-2.743949535,-2.994496474,-3.230661583,-3.451459739,-3.857455461,-4.237340784,-4.600856064,-5.271950262,-5.923627805,-6.905755479;

-0.839660915,-1.042290903,-1.903441818,-2.644963763,-3.247883029,-3.726394985,-4.115936922,-4.435780483,-4.944341506,-5.347802258,-5.637152491,-5.855368185,-6.035465593,-6.19842498,-6.348928975,-6.489013585,-6.620555598,-6.756376694,-6.887542825,-7.014107538,-7.314061744,-7.587410275,-8.065504758,-8.482463963,-8.856286928,-9.197637229,-9.527899821,-9.835934967,-10.37911718,-10.89086157,-11.39649477,-12.32137564,-13.18717842,-14.46578595;

-0.434321098,-0.602670401,-1.320995719,-2.033868208,-2.604922543,-3.052485272,-3.420888409,-3.724431867,-4.208980217,-4.593973645,-4.867698551,-5.075165945,-5.24651361,-5.401377059,-5.544303257,-5.677384594,-5.802400289,-5.93129909,-6.055850971,-6.176077554,-6.461316029,-6.721153567,-7.17489277,-7.570055738,-7.923652136,-8.246254197,-8.557702714,-8.848330451,-9.364695905,-9.860308315,-10.3301894,-11.20957801,-12.04060072,-13.26529383;

-0.105991869,-0.253831613,-0.835923818,-1.488584435,-2.021187374,-2.440467252,-2.789212708,-3.077942647,-3.539155045,-3.9058438,-4.16492919,-4.362970851,-4.526371423,-4.674275656,-4.810758186,-4.937859984,-5.057169546,-5.179601894,-5.297828303,-5.412017567,-5.68332399,-5.930506916,-6.362658657,-6.738345361,-7.073638091,-7.379027512,-7.672553111,-7.946529598,-8.435236918,-8.907874545,-9.352328834,-10.18811308,-10.98972642,-12.1589293;

0.162441393,0.030614579,-0.439663348,-1.007364448,-1.492254861,-1.877723968,-2.202592528,-2.47326385,-2.907327859,-3.25307792,-3.495428171,-3.682445715,-3.836621479,-3.976794875,-4.106177337,-4.227978828,-4.342287343,-4.458920614,-4.571637168,-4.680723048,-4.939899252,-5.175514539,-5.587484973,-5.94665571,-6.266738636,-6.557837532,-6.83135523,-7.086974125,-7.547721089,-7.993619661,-8.413873688,-9.214770893,-9.968076423,-11.06833024;

0.350613945,0.233718669,-0.157830812,-0.635466461,-1.058241762,-1.406097463,-1.70500142,-1.957498973,-2.366236743,-2.693293744,-2.921257969,-3.096749599,-3.240411389,-3.370334344,-3.49048696,-3.605004731,-3.712427111,-3.821797382,-3.927840388,-4.030496972,-4.27642697,-4.501484908,-4.89474872,-5.235320514,-5.538495333,-5.813815422,-6.073490889,-6.316283859,-6.752133506,-7.173969547,-7.570364211,-8.332893841,-9.039305343,-10.09258007;

0.497525929,0.394931131,0.06473234,-0.323346993,-0.680679191,-0.986301947,-1.25471322,-1.485770033,-1.863993174,-2.168474252,-2.381524667,-2.544909061,-2.677893925,-2.797222927,-2.908592359,-3.017139422,-3.119066234,-3.222041875,-3.321955587,-3.41880296,-3.652681692,-3.867739344,-4.242911664,-4.565715104,-4.852478624,-5.112437419,-5.357796547,-5.587069137,-5.999921007,-6.397620136,-6.775165946,-7.49422564,-8.164421353,-9.171528428;

0.627734697,0.538224257,0.255081472,-0.060745532,-0.362260555,-0.626573693,-0.860982545,-1.06644111,-1.408093453,-1.684519439,-1.879847311,-2.027908667,-2.151163007,-2.264317216,-2.369337849,-2.472695746,-2.569873171,-2.6674444,-2.762252823,-2.854214162,-3.074818404,-3.276816115,-3.628549105,-3.930345959,-4.199273097,-4.443494816,-4.674731858,-4.890915796,-5.28202471,-5.66238327,-6.019826772,-6.707252944,-7.358705717,-8.328260765;

0.702443074,0.631011773,0.39065401,0.135602128,-0.113775306,-0.337289978,-0.536819277,-0.714923468,-1.016608276,-1.25845205,-1.432256241,-1.571332404,-1.686275345,-1.791208424,-1.888846699,-1.987022615,-2.079380067,-2.171641127,-2.261488476,-2.348669755,-2.556378932,-2.745866981,-3.074682451,-3.356981045,-3.608958614,-3.838131867,-4.055577872,-4.260146548,-4.635276002,-5.000615073,-5.346212243,-6.00897609,-6.638635906,-7.591096045;

0.726005688,0.658710809,0.434804452,0.200557904,-0.00759559,-0.202715319,-0.383166643,-0.545066028,-0.823539313,-1.043552287,-1.212313412,-1.34337499,-1.452781309,-1.553206467,-1.646903054,-1.74232692,-1.832198196,-1.921484329,-2.00823501,-2.092431316,-2.293123662,-2.476004277,-2.793550337,-3.06623505,-3.309829075,-3.533495992,-3.746321863,-3.945764704,-4.311156295,-4.668263768,-5.006210009,-5.657279735,-6.278883101,-7.218220961;

0.732225831,0.667145265,0.453474187,0.237117245,0.039752562,-0.132414127,-0.294236538,-0.445393272,-0.705578336,-0.916304629,-1.078345405,-1.203783884,-1.309401231,-1.406787865,-1.497787401,-1.591380547,-1.679586489,-1.766853505,-1.851590047,-1.933773872,-2.129818882,-2.308421984,-2.618615888,-2.885000205,-3.12623471,-3.345822639,-3.554280386,-3.74976925,-4.110982354,-4.461150314,-4.801655064,-5.439118952,-6.056948198,-6.984341371;

0.732033735,0.668882315,0.464935989,0.259554931,0.081129463,-0.080387233,-0.225001349,-0.354527639,-0.594975815,-0.797581091,-0.952861228,-1.072630629,-1.174378,-1.268772581,-1.357077695,-1.448783507,-1.535334416,-1.62060378,-1.70332159,-1.783482039,-1.974876678,-2.149209164,-2.451995631,-2.716049006,-2.951877602,-3.166932314,-3.371049931,-3.562638181,-3.917916476,-4.263589532,-4.59872748,-5.232924933,-5.850151576,-6.775461356;

-1.018147546,-1.227369088,-2.130805478,-2.893959126,-3.493462258,-3.967817429,-4.3552719,-4.672781543,-5.177126941,-5.573808146,-5.852253545,-6.058992526,-6.228330669,-6.378428733,-6.516286002,-6.644011184,-6.763563426,-6.887569664,-7.007366078,-7.122915237,-7.388517636,-7.625750617,-8.100396061,-8.514633737,-8.886270561,-9.225882734,-9.555981751,-9.864032739,-10.4016151,-10.91382643,-11.42478165,-12.35839441,-13.23444169,-14.52815802;

-0.61094234,-0.780922172,-1.52811317,-2.255493591,-2.823968055,-3.26913504,-3.636050162,-3.937967413,-4.419104899,-4.798452304,-5.06250723,-5.259751631,-5.421285637,-5.564464978,-5.695962276,-5.817839124,-5.931915331,-6.050020596,-6.164244084,-6.274521945,-6.528755252,-6.756038908,-7.206495828,-7.599082018,-7.950650452,-8.271675215,-8.583219228,-8.874096335,-9.386097543,-9.884347039,-10.35822343,-11.24662594,-12.08821637,-13.3290697;

-0.281172485,-0.428392933,-1.027420519,-1.6904921,-2.220360644,-2.637690582,-2.984825283,-3.27213385,-3.730248454,-4.092146279,-4.342612026,-4.531428366,-4.685815672,-4.823163083,-4.949291913,-5.066223898,-5.175585396,-5.288129669,-5.396843757,-5.501889176,-5.74484611,-5.962380695,-6.391346791,-6.764582249,-7.098005783,-7.401987567,-7.695860039,-7.970328561,-8.455624884,-8.93168109,-9.380254022,-10.22547885,-11.03914042,-12.22353369;

-0.007374024,-0.137610653,-0.61555555,-1.18932845,-1.671200469,-2.055317408,-2.378516826,-2.648220073,-3.079985725,-3.421910754,-3.656967512,-3.835964277,-3.982058522,-4.112767926,-4.232885235,-4.345508772,-4.450858718,-4.558587939,-4.66280404,-4.763761661,-4.9975604,-5.206454893,-5.615412951,-5.972906488,-6.29176799,-6.582014737,-6.854954071,-7.110190278,-7.567901088,-8.017063031,-8.441681155,-9.253859798,-10.01805385,-11.13322706;

0.185378482,0.069879901,-0.323485031,-0.800696155,-1.220503535,-1.567654399,-1.865223137,-2.117171963,-2.525107748,-2.84988648,-3.072195125,-3.240743665,-3.377181573,-3.498524292,-3.610255046,-3.716280191,-3.815658472,-3.916750877,-4.014878702,-4.109937733,-4.332335616,-4.532838717,-4.923170578,-5.262093183,-5.564085,-5.83861135,-6.097952708,-6.340736558,-6.774016814,-7.199151185,-7.599159627,-8.373068157,-9.090489235,-10.15911789;

0.333654825,0.231833859,-0.097291411,-0.479667393,-0.833934409,-1.139333194,-1.40754281,-1.638960575,-2.017331476,-2.320287401,-2.528694009,-2.685490903,-2.811497461,-2.922836472,-3.026137166,-3.126307129,-3.220064456,-3.31487739,-3.406990306,-3.496376,-3.707419469,-3.89887812,-4.271095588,-4.592153235,-4.877666137,-

5.136800958,-5.381804337,-5.611048035,-6.022181405,-6.42317665,-6.804592869,-7.534752776,-8.215540579,-9.238487088;

0.4611546,0.372321749,0.091182183,-0.220969758,-0.519467962,-0.782900725,-1.01625059,-1.221055414,-1.562118158,-1.836074443,-2.026653617,-2.168338257,-2.284689376,-2.389447755,-2.486214998,-2.581076455,-2.669983451,-2.75927778,-2.846181253,-2.930594203,-3.128310275,-3.306916217,-3.655646562,-3.955577242,-4.223161001,-4.466503836,-4.697329287,-4.913468835,-5.303472363,-5.687099146,-6.048763031,-6.74698845,-7.40898039,-8.394619653;

0.532661562,0.461540904,0.223144381,-0.031482979,-0.278750412,-0.5008859,-0.69913488,-0.875955945,-1.175593665,-1.413441355,-1.582400873,-1.714095451,-1.821709755,-1.917928387,-2.006986231,-2.096249624,-2.179961796,-2.263591837,-2.345221657,-2.424558056,-2.608705452,-2.774408669,-3.100017806,-3.380224556,-3.630720077,-3.85893958,-4.075922186,-4.280360983,-4.654599345,-5.023153192,-5.373246977,-6.046947417,-6.686787619,-7.656200904;

0.551702332,0.484358863,0.261749424,0.028141677,-0.180071977,-0.374513108,-0.553545865,-0.714068897,-0.990268018,-1.205673488,-1.368732588,-1.491824563,-1.593432274,-1.68458793,-1.769173661,-1.855101751,-1.935776228,-2.015894618,-2.093918769,-2.169800978,-2.345645982,-2.503662203,-2.817775989,-3.088149679,-3.330094469,-3.552565605,-3.764703883,-3.963911132,-4.328673262,-4.689198913,-5.031720443,-5.693852434,-6.325778416,-7.282101622;

0.552994501,0.487676833,0.27541523,0.060284057,-0.136703381,-0.309348711,-0.470388814,-0.620106975,-0.878154408,-1.08302158,-1.239160654,-1.35623917,-1.453730287,-1.541463265,-1.62298888,-1.706662649,-1.785285972,-1.863003131,-1.938659242,-2.012189806,-2.182487647,-2.335484136,-2.642064589,-2.905991443,-3.145284769,-3.363568914,-3.571311868,-3.76655724,-4.12718746,-4.481019783,-4.827267593,-5.474806208,-6.103014493,-7.047611064;

0.553654696,0.489171896,0.284785154,0.07953678,-0.099040797,-0.260875601,-0.405287473,-0.534792144,-0.772805386,-0.968576043,-1.117756992,-1.228798263,-1.322116126,-1.406491982,-1.484949269,-1.56632478,-1.642909416,-1.718266362,-1.791551824,-1.862736468,-2.02754433,-2.175589042,-2.474584473,-2.735826109,-2.969630658,-3.18331605,-3.386709179,-3.578042879,-3.932977355,-4.282423996,-4.623320708,-5.267708127,-5.895942627,-6.837992602;

1.671335571,1.446330626,0.308517281,-0.497052049,-1.102733187,-1.571142871,-1.959028166,-2.267357596,-2.740256749,-3.101393974,-3.342535465,-3.523205246,-3.665540038,-3.786608821,-3.89749399,-4.000022111,-4.095927322,-4.196222128,-4.294007531,-4.388363909,-4.612609249,-4.8160213,-5.189409719,-5.51201364,-5.799177381,-6.060491704,-6.310554355,-6.542953793,-6.966856797,-7.349673069,-7.701416957,-8.337393376,-8.906983402,-9.727410557;

2.313901537,2.151399673,1.208325515,0.425976238,-0.151719023,-0.588315893,-0.954299012,-1.246462693,-1.693527679,-2.035134048,-2.261171145,-2.430661659,-2.564391895,-2.677527683,-2.781070354,-2.876759854,-2.966050389,-3.058676503,-3.148873862,-3.236036656,-3.443911532,-3.63241507,-3.979905917,-4.279915682,-4.546676755,-4.789073026,-5.021300263,-5.237037474,-5.630004537,-5.985172147,-6.311951861,-6.904082099,-7.436063681,-8.199011925;

2.864862945,2.746430942,2.009867662,1.287777682,0.746137387,0.337424164,-0.00655288,-0.282684316,-0.707348656,-1.032330021,-1.244891299,-1.405302169,-1.531146245,-1.637611598,-1.734909662,-1.824576163,-1.908427734,-1.994162481,-2.07781151,-2.158734403,-2.352121882,-2.527628095,-2.849576704,-3.127373768,-3.373803071,-3.597451852,-3.811746606,-4.010762522,-4.376067296,-4.706245718,-5.009977407,-5.55869765,-6.05252921,-6.760950969;

3.269327082,3.183357163,2.666285863,2.043607086,1.549897123,1.173223309,0.853469353,0.594169937,0.193430425,-0.114159537,-0.313408808,-0.46499779,-0.583298981,-0.683076198,-0.774215415,-0.858235623,-0.936422676,-1.015664499,-1.09303788,-1.168169884,-1.348027174,-1.510903242,-1.812316343,-2.073281682,-2.303836287,-2.512769433,-2.708943834,-2.89110093,-3.226979228,-3.53056048,-3.811058861,-4.317499687,-4.773725593,-5.426094399;

3.59333877,3.52847642,3.178025117,2.691574668,2.269512927,1.932977003,1.639253074,1.396395578,1.021621541,0.732505575,0.543614355,0.400279241,0.289529625,0.196353078,0.111306327,0.032887849,-0.040614085,-0.113414473,-0.184999909,-0.254446354,-0.42220805,-0.574883509,-0.851896926,-1.091505535,-1.304425648,-1.497987067,-1.680193442,-1.849585192,-2.159787384,-2.440600227,-2.698930624,-3.168341701,-3.591958117,-4.198175554;

3.888007793,3.835830929,3.592292491,3.24028693,2.898233016,2.607323968,2.347541602,2.126376432,1.779185192,1.508457606,1.330994316,1.196811349,1.094162181,1.007721529,0.928971167,0.856416439,0.788881639,0.722717962,0.657440299,0.593999872,0.439413139,0.298263399,0.041771958,-0.179781912,-0.37677474,-0.55587968,-0.724664192,-0.881633752,-1.169637422,-1.43048471,-1.670911007,-2.107737303,-2.502436502,-3.069100668;

```
4.217141562,4.169845441,3.982108477,3.716378569,3.443143775,3.201446829,2.985702519,2.796114963,2.486158098,2
.242732695,2.082100416,1.95728211,1.860945912,1.779441306,1.70500663,1.636247343,1.572356228,1.511005944,1.45
0772065,1.39206796,1.249000714,1.118508971,0.881561711,0.677019801,0.495891068,0.331424844,0.176448726,0.0320
03535,-0.234394283,-0.476227461,-0.699487554,-1.106651157,-1.475110038,-2.003963656;

4.512456974,4.472284099,4.321881589,4.115185704,3.905178931,3.714489696,3.540778652,3.383732464,3.121516392,2
.916198321,2.773820372,2.659107997,2.569795419,2.49509595,2.426585779,2.363369641,2.304482143,2.248954291,2.1
94241596,2.140833731,2.010556537,1.891727765,1.673221223,1.484386379,1.316668768,1.16392522,1.019965679,0.885
632475,0.639800978,0.416114409,0.209351856,-0.16869689,-0.511547756,-1.003929383;

4.673324218,4.635135665,4.499033995,4.324725228,4.150160334,3.987646251,3.83620069,3.698975162,3.460942471,3.
27803889,3.144353842,3.036000079,2.950403799,2.877332627,2.810359985,2.748315507,2.690817932,2.637174773,2.58
4671871,2.533483078,2.409387165,2.296828101,2.094231924,1.918703147,1.761887233,1.618351274,1.482803234,1.356
20564,1.123571142,0.911311877,0.714675963,0.355044967,0.028067636,-0.443040463;

4.776206848,4.736186805,4.607417518,4.455244565,4.300509225,4.155547509,4.019437745,3.895091088,3.673860734,3
.505974526,3.37875962,3.274941716,3.191854237,3.120006817,3.054036991,2.993036797,2.93616352,2.883893398,2.83
2931525,2.783230353,2.663255015,2.554896734,2.363023884,2.196394011,2.046358989,1.90895244,1.779160845,1.6578
96189,1.433977497,1.22926166,1.039421855,0.691629465,0.374978524,-0.082359227;

4.872608405,4.827713188,4.705442451,4.576585218,4.444146279,4.313722024,4.192622039,4.083045091,3.880024534,3
.72608906,3.605302107,3.506146912,3.425662456,3.354988212,3.289944081,3.230075798,3.173836199,3.1229214,3.073
456993,3.025457933,2.909546258,2.805399339,2.624249847,2.465827485,2.323031545,2.191872531,2.067849872,1.9519
65254,1.736850125,1.539832436,1.356739297,1.020882319,0.714731308,0.271309087];
end



function Sigma = getSigma(thisFreq, inMagnitude, inRrup, inSigmaType)
   % code developed and written by Silvia Mazzoni
   %      smazzoni@ucla.edu, March 2020
   if lower(inSigmaType) == lower("ErgodicSigma")
      Sigma = getSigmaCompErg(thisFreq, inMagnitude, inRrup);
   else
      Sigma = getSigmaCompSS(thisFreq, inMagnitude, inRrup);
   end


end

function FrequencyListCompSS = getFrequencyListCompSS()
   % code developed and written by Silvia Mazzoni
   %      smazzoni@ucla.edu, March 2020
   FrequencyListCompSS =
[100;50;40;33.333;25;20;13.333;10;6.667;5;4;3.333;2.5;2;1.333;1;0.667;0.5;0.333;0.25;0.2;0.133;0.1;-1;-2];
end
function SigmaCompSS = getSigmaCompSS(thisFreq, inMagnitude, inRrup);
   % code developed and written by Silvia Mazzoni
   %      smazzoni@ucla.edu, March 2020
   WeightSigmaCompSSC=0.63;
   WeightSigmaCompSSH=0.185;
   WeightSigmaCompSSL=0.185;
   WeightSigmaCompSSSum=1;
   %PeriodListCompSS = [0.01;0.02;0.025;0.03;0.04;0.05;0.075;0.1;0.15;0.2;0.25;0.3;0.4;0.5;0.75;1;1.5;2;3;4;5;7.5;10;-1;-2];

   FrequencyListCompSS = getFrequencyListCompSS();
   thisRowIndex = -999;
```

```
  NrowFSS = length(FrequencyListCompSS);
  for iRowF = 1 : NrowFSS
    hereFreq = FrequencyListCompSS(iRowF, 1);
    if abs(hereFreq - thisFreq) < 0.000001
      thisRowIndex = iRowF;
    end
  end


  %Central Model
      SigCompSS1C =
[0.7006;0.6998;0.6994;0.699;0.6981;0.6974;0.6955;0.6935;0.6898;0.6862;0.6827;0.6794;0.673;0.6671;0.6541;0.6432;0.626
2;0.6143;0.5998;0.5925;0.5889;0.5858;0.5853;0.7006;0.6303];
      SigCompSS2C =
[0.6846;0.6838;0.6833;0.6829;0.682;0.6813;0.6793;0.6773;0.6735;0.6698;0.6662;0.6628;0.6563;0.6502;0.6369;0.6256;0.60
82;0.5959;0.5811;0.5736;0.5698;0.5666;0.5661;0.6846;0.6247];
      SigCompSS3C =
[0.6127;0.6122;0.6119;0.6116;0.611;0.6105;0.6092;0.6079;0.6063;0.6066;0.6068;0.6072;0.6075;0.606;0.5999;0.5921;0.578
1;0.5674;0.5517;0.5438;0.5397;0.5364;0.5357;0.6127;0.5765];
      SigCompSS4C =
[0.4973;0.4974;0.4974;0.4974;0.4975;0.4975;0.4977;0.4978;0.5007;0.5092;0.5171;0.5254;0.5399;0.5482;0.5577;0.5576;0.5
51;0.5436;0.5267;0.5181;0.5138;0.5102;0.5095;0.4973;0.4896];
    S1 = SigCompSS1C(thisRowIndex, 1);
    S2 = SigCompSS2C(thisRowIndex, 1);
    S3 = SigCompSS3C(thisRowIndex, 1);
    S4 = SigCompSS4C(thisRowIndex, 1);
    SigmaCompSSC = getSigmaCompSSi(inMagnitude, S1, S2, S3, S4);
  %High Model
      SigCompSS1H =
[0.8193;0.8183;0.8178;0.8173;0.8162;0.8153;0.8129;0.8106;0.8061;0.8019;0.798;0.7943;0.7876;0.7816;0.7697;0.7611;0.75
02;0.7444;0.7391;0.7372;0.7364;0.7358;0.7357;0.8193;0.748];
      SigCompSS2H =
[0.8075;0.8064;0.8059;0.8054;0.8043;0.8034;0.801;0.7986;0.7941;0.7899;0.7859;0.7822;0.7753;0.7693;0.7572;0.7484;0.73
73;0.7313;0.7258;0.7237;0.7228;0.7222;0.7221;0.8075;0.7431];
      SigCompSS3H =
[0.7348;0.7343;0.734;0.7338;0.7333;0.7329;0.7317;0.7306;0.7292;0.7295;0.7297;0.73;0.7305;0.7295;0.7256;0.7209;0.7137;
0.7093;0.7033;0.7009;0.6999;0.699;0.6989;0.7348;0.696];
      SigCompSS4H =
[0.6377;0.6379;0.6381;0.6382;0.6384;0.6385;0.639;0.6395;0.6413;0.6453;0.6492;0.6537;0.6623;0.6681;0.6762;0.6772;0.67
43;0.6738;0.6698;0.6687;0.6682;0.6679;0.6679;0.6377;0.6419];
    S1 = SigCompSS1H(thisRowIndex, 1);
    S2 = SigCompSS2H(thisRowIndex, 1);
    S3 = SigCompSS3H(thisRowIndex, 1);
    S4 = SigCompSS4H(thisRowIndex, 1);
    SigmaCompSSH = getSigmaCompSSi(inMagnitude, S1, S2, S3, S4);
  %Low Model
      SigCompSS1L =
[0.5879;0.5873;0.587;0.5867;0.586;0.5855;0.584;0.5825;0.5795;0.5765;0.5736;0.5707;0.565;0.5595;0.5467;0.5354;0.5171;0
.5037;0.487;0.4784;0.474;0.4702;0.4695;0.5879;0.522];
      SigCompSS2L =
[0.5682;0.5676;0.5672;0.5669;0.5662;0.5657;0.5641;0.5626;0.5595;0.5564;0.5533;0.5503;0.5444;0.5387;0.5253;0.5135;0.4
944;0.4803;0.4628;0.4538;0.4491;0.4451;0.4444;0.5682;0.5159];
      SigCompSS3L =
[0.4981;0.4976;0.4973;0.497;0.4964;0.496;0.4946;0.4933;0.4917;0.4921;0.4924;0.4928;0.4931;0.4913;0.4837;0.4741;0.456
6;0.4433;0.4243;0.4148;0.4101;0.4061;0.4053;0.4981;0.4703];
      SigCompSS4L =
[0.3901;0.3899;0.3898;0.3897;0.3895;0.3893;0.3887;0.3882;0.3904;0.3994;0.4079;0.4168;0.4326;0.4411;0.4504;0.4508;0.4
482;0.4416;0.4226;0.4124;0.4071;0.4025;0.4017;0.3901;0.3928];
```

```
      S1 = SigCompSS1L(thisRowIndex, 1);
      S2 = SigCompSS2L(thisRowIndex, 1);
      S3 = SigCompSS3L(thisRowIndex, 1);
      S4 = SigCompSS4L(thisRowIndex, 1);
    SigmaCompSSL = getSigmaCompSSi(inMagnitude, S1, S2, S3, S4);


  SigmaCompSS = SigmaCompSSC * WeightSigmaCompSSC / WeightSigmaCompSSSum + SigmaCompSSH *
WeightSigmaCompSSH / WeightSigmaCompSSSum + SigmaCompSSL * WeightSigmaCompSSL / WeightSigmaCompSSSum;


end


function SigmaCompSSi = getSigmaCompSSi(inMagnitude, S1, S2, S3, S4)
  % code developed and written by Silvia Mazzoni
  %      smazzoni@ucla.edu, March 2020
  if inMagnitude <= 4.5
     SigmaCompSSi = S1;
  elseif inMagnitude <= 5.0
     SigmaCompSSi = S1 + (S2 - S1) * (inMagnitude - 4.5) / 0.5;
  elseif inMagnitude <= 5.5
     SigmaCompSSi = S2 + (S3 - S2) * (inMagnitude - 5.0) / 0.5;
  elseif inMagnitude <= 6.5
     SigmaCompSSi = S3 + (S4 - S3) * (inMagnitude - 5.5) / 1.0;
  else
     SigmaCompSSi = S4;
  end

end


function FrequencyListCompErg = getFrequencyListCompErg()
  % code developed and written by Silvia Mazzoni
  %      smazzoni@ucla.edu, March 2020

FrequencyListCompErg=[100;50;40;33.333;25;20;13.333;10;6.667;5;4;3.333;2.5;2;1.333;1;0.667;0.5;0.333;0.25;0.2;0.133;0.1
;-1;-2];
end

function SigmaCompErg = getSigmaCompErg(thisFreq, inMagnitude, inRrup)
  % code developed and written by Silvia Mazzoni
  %      smazzoni@ucla.edu, March 2020
  WeightSigmaCompErgC=0.63;
  WeightSigmaCompErgH=0.185;
  WeightSigmaCompErgL=0.185;
  WeightSigmaCompErgSum=1;
 % PeriodListCompErg=[0.01;0.02;0.025;0.03;0.04;0.05;0.075;0.1;0.15;0.2;0.25;0.3;0.4;0.5;0.75;1;1.5;2;3;4;5;7.5;10;-1;-2];
  FrequencyListCompErg = getFrequencyListCompErg();


  thisRowIndex = -999;
  NrowFErg = length(FrequencyListCompErg);
  for iRowF = 1 : NrowFErg
     hereFreq = FrequencyListCompErg(iRowF, 1);
     if abs(hereFreq - thisFreq) < 0.000001
        thisRowIndex = iRowF;
     end
```

```
  end


  %Central Model

SigCompErg1C=[0.7618;0.761;0.7646;0.7676;0.781;0.792;0.7881;0.7726;0.7597;0.7556;0.7523;0.7488;0.7412;0.7335;0.7151
;0.6992;0.6742;0.6562;0.6337;0.6212;0.6139;0.6053;0.6016;0.7618;0.7045];

SigCompErg2C=[0.7471;0.7463;0.75;0.753;0.7667;0.7779;0.7739;0.7581;0.745;0.7408;0.7374;0.7338;0.7261;0.7182;0.6995;
0.6832;0.6576;0.6392;0.616;0.6032;0.5956;0.5867;0.5829;0.7471;0.6995];

SigCompErg3C=[0.6819;0.6814;0.6856;0.6891;0.7044;0.7169;0.7133;0.6969;0.685;0.6844;0.6844;0.6842;0.6825;0.6786;0.66
61;0.6528;0.6299;0.6127;0.5885;0.5749;0.567;0.5576;0.5535;0.6819;0.6569];

SigCompErg4C=[0.5809;0.5808;0.5861;0.5905;0.609;0.6239;0.6214;0.6039;0.5941;0.6001;0.6065;0.6129;0.6231;0.6274;0.62
83;0.6215;0.605;0.5907;0.5651;0.5509;0.5426;0.5327;0.5285;0.5809;0.5825];
    S1 = SigCompErg1C(thisRowIndex, 1);
    S2 = SigCompErg2C(thisRowIndex, 1);
    S3 = SigCompErg3C(thisRowIndex, 1);
    S4 = SigCompErg4C(thisRowIndex, 1);
    SigmaCompErgC = getSigmaCompErgi(inMagnitude, S1, S2, S3, S4);
  %High Model

SigCompErg1H=[0.8747;0.8737;0.8767;0.8791;0.8909;0.9011;0.899;0.8858;0.8725;0.8668;0.8621;0.8575;0.8485;0.8402;0.8
228;0.8096;0.7916;0.7807;0.7687;0.7627;0.7591;0.7545;0.7519;0.8747;0.8134];

SigCompErg2H=[0.8635;0.8625;0.8655;0.8679;0.8799;0.8902;0.888;0.8747;0.8613;0.8555;0.8508;0.8461;0.837;0.8286;0.81
1;0.7976;0.7792;0.7681;0.7558;0.7496;0.746;0.7412;0.7386;0.8635;0.809];

SigCompErg3H=[0.7963;0.7958;0.7993;0.8022;0.8158;0.8275;0.8267;0.8137;0.8023;0.8004;0.7994;0.7983;0.7956;0.7918;0.
7814;0.7717;0.7568;0.7471;0.7343;0.7276;0.7238;0.7188;0.716;0.7963;0.7661];

SigCompErg4H=[0.7076;0.7078;0.7121;0.7157;0.7315;0.7452;0.7457;0.7328;0.7233;0.7246;0.727;0.7295;0.734;0.736;0.736
1;0.7314;0.7203;0.7139;0.7025;0.6967;0.6932;0.6883;0.6855;0.7076;0.7163];
    S1 = SigCompErg1H(thisRowIndex, 1);
    S2 = SigCompErg2H(thisRowIndex, 1);
    S3 = SigCompErg3H(thisRowIndex, 1);
    S4 = SigCompErg4H(thisRowIndex, 1);
    SigmaCompErgH = getSigmaCompErgi(inMagnitude, S1, S2, S3, S4);
  %Low Model

SigCompErg1L=[0.6539;0.6533;0.6575;0.661;0.6758;0.6874;0.682;0.6645;0.6521;0.6496;0.6476;0.6452;0.6392;0.6323;0.614
5;0.5977;0.5703;0.5497;0.5229;0.5076;0.4983;0.4871;0.4826;0.6539;0.6034];

SigCompErg2L=[0.6361;0.6355;0.6398;0.6434;0.6586;0.6705;0.665;0.6471;0.6343;0.6317;0.6296;0.6271;0.621;0.6138;0.595
4;0.578;0.5496;0.5282;0.5004;0.4844;0.4747;0.463;0.4583;0.6361;0.598];

SigCompErg3L=[0.5736;0.5731;0.5779;0.5819;0.5987;0.6118;0.606;0.5867;0.5743;0.575;0.5761;0.5767;0.576;0.5723;0.5586
;0.543;0.5156;0.4945;0.4648;0.448;0.4379;0.4256;0.4207;0.5736;0.5587];

SigCompErg4L=[0.4807;0.4805;0.4864;0.4913;0.5113;0.5264;0.52;0.4982;0.4876;0.4958;0.504;0.5119;0.5244;0.5293;0.5299
;0.5227;0.508;0.4926;0.4624;0.4446;0.4337;0.4205;0.4153;0.4807;0.4936];
    S1 = SigCompErg1L(thisRowIndex, 1);
    S2 = SigCompErg2L(thisRowIndex, 1);
    S3 = SigCompErg3L(thisRowIndex, 1);
    S4 = SigCompErg4L(thisRowIndex, 1);
    SigmaCompErgL = getSigmaCompErgi(inMagnitude, S1, S2, S3, S4);
```

```
   SigmaCompErg = SigmaCompErgC * WeightSigmaCompErgC / WeightSigmaCompErgSum + SigmaCompErgH *
WeightSigmaCompErgH / WeightSigmaCompErgSum + SigmaCompErgL * WeightSigmaCompErgL /
WeightSigmaCompErgSum;


end

function SigmaCompErgi = getSigmaCompErgi(inMagnitude, S1, S2, S3, S4)
   % code developed and written by Silvia Mazzoni
   %      smazzoni@ucla.edu, March 2020
   if inMagnitude <= 4.5
      SigmaCompErgi = S1;
   elseif inMagnitude <= 5.0
      SigmaCompErgi = S1 + (S2 - S1) * (inMagnitude - 4.5) / 0.5;
   elseif inMagnitude <= 5.5
      SigmaCompErgi = S2 + (S3 - S2) * (inMagnitude - 5.0) / 0.5;
   elseif inMagnitude <= 6.5
      SigmaCompErgi = S3 + (S4 - S3) * (inMagnitude - 5.5) / 1.0;
   else
      SigmaCompErgi = S4;
   end

end
```

4.4 NGA-East Ground-Motion Characterization Tool – Python

NGA-East_GMMtool_Python_R200316.py

```
### ################################################################
### ################################################################
### ----------------------------------------------------------------
### ----------------------------------------------------------------
### -- NGA-East: Central and Eastern North America
###               Ground-Motion Characterization Tool
###               Python Version
### ----------------------------------------------------------------
###                         March 2020
### ----------------------------------------------------------------
### Ground-Motion Model Tool developed by
###           Silvia Mazzoni
###              smazzoni@ucla.edu
###           B. John Garrick Institute for the Risk Sciences
###              https://www.risksciences.ucla.edu/nhr3/gmtools/home
### ----------------------------------------------------------------
### Source:
###       Report PEER 2018/08 - Central and Eastern North America
###       Ground-Motion Characterization - NGA-East Final Report
###          Christine Goulet, Yousef Bozorgnia, Norman Abrahamson,
###          Nicolas Kuehn, Linda Al Atik, Robert Youngs, and Robert Graves
###          apps.peer.berkeley.edu/publications/peer_reports/reports_2018/NGA-East_1MainReport_FINAL.pdf


### ----------------------------------------------------------------
### Instruction:
###   1. Edit the values of the user-input parameters.
###       Model-Range limits are given with each input parameters.
###   2. Run Module
###   4. Result values are printed in Command Window (you can copy and paste them elsewhere)
### ----------------------------------------------------------------
###   Notes:
###       The Median and Sigma models are described in the report.
###       The values are computed for Vs30=3000 (m/s)
###       Damping Ratio = 5#
###       This script only outputs the Weighted-Average Composite values for Median and Sigma.
###       The values for the individual components can be found in the Excel version of the NGA-East GMtool.
### ----------------------------------------------------------------
### Variables:
###   User-Input Parameters:
###       inMagnitude:    Event Magnitude. Range=4-8.2
###       inRrup:         Rupture Distance (km). Range=1-1500km
###       inSigmaType:    Type of Sigma: Ergodic or Single-Station
###       inNsigma:       Number of Standard Deviations (sigma)
###       inPeriodList:   Array of periods (sec) for computation. Range: 0.01-10, 'PGA','PGV', -1 (=PGA), -2 (=PGV)
###   Output Variables:
###     outDataTable:               Table with all in/out data matrix
###     outTlist:                 Array of Periods used for computuation (sec) -1 (=PGA), -2 (=PGV)
###     outMedianList:              Array of median PSA (g);
###     outSigmaList:             Array of Sigma;
###     outMedianMinusNSigmaList:   Array of Median - inNsigma*Sigma (g);
###     outMedianPlusNSigmaList:          Array of Median + inNsigma*Sigma (g);
### ----------------------------------------------------------------
### ################################################################
```

```
### ############################################################


#########################################################
################ USER INPUT #############################
#########################################################
inMagnitude = 5.5;                      # Range: 4-8.2
inRrup = 77.3;                          # range: 0-1500km
inSigmaType = "SingleStation";          # options: "Ergodic" "SingleStation"
inNsigma = 1.0;
inPeriodList =
(0.01,0.015,0.02,0.025,0.03,0.04,0.05,0.075,0.1,0.15,0.2,0.25,0.3,0.4,0.5,0.75,1,1.5,2,3,4,5,7.5,10,"PGA","PGV"); #
range:0.01-10sec

#########################################################
############## end USER INPUT ###########################
#########################################################




#########################################################
############## RUN DATA    ##################
#########################################################
import math


def runNGAEastGMM(inPeriodList,inMagnitude,inRrup,inSigmaType,inNsigma):
    # initialize arrays:
    outTlist= []
    MedianList= []
    SigmaList= []
    MedianMinusNSigmaList= []
    MedianMinusNSigmaList= []
    MedianPlusNSigmaList= []
    MagList= []
    RrupList= []
    NsigmaList= []
    DataMatrix= []

    print ('Period_(s)          Magnitude          Rrup_(km)          Vs30_(m/s)          PSA_median_5pctDamp_(g)
        Sigma_'+inSigmaType+'_(ln_units)       PSA_median-'+str(inNsigma)+'*Sigma_(g)
        PSA_median+'+str(inNsigma)+'*Sigma_(g)');

    for iRow in range(len(inPeriodList)):
        hereT=inPeriodList[iRow]
        thisT = getTFvalue(hereT)
        medianPSA = NGAEastGMM(thisT,inMagnitude,inRrup,"MedianPSA")
        Sigma = NGAEastGMM(thisT,inMagnitude,inRrup,inSigmaType)
        outTlist.append(thisT)
        MedianMinusNSigma = math.exp(math.log(medianPSA)-inNsigma*Sigma)
        MedianPlusNSigma = math.exp(math.log(medianPSA)+inNsigma*Sigma)

print(hereT,inMagnitude,inRrup,3000,round(medianPSA,9),round(Sigma,9),round(MedianMinusNSigma,9),round(MedianPlu
sNSigma,9), sep='\t')
        DataMatrix.append([thisT,medianPSA,MedianMinusNSigma,MedianPlusNSigma])
        MedianList.append(medianPSA);
        SigmaList.append(Sigma);
```

```
            MedianMinusNSigmaList.append(MedianMinusNSigma);
            MedianPlusNSigmaList.append(MedianPlusNSigma);
            MagList.append(inMagnitude);
            RrupList.append(inRrup);
            NsigmaList.append(inNsigma);
      return DataMatrix



#########################################################
############## END RUN DATA ######################
#########################################################



#########################################################
################ PROGRAM DATA ##########################
#########################################################

def getTFvalue(thisTin):
      if type(thisTin)==str:
            if thisTin.lower() == "pga".lower():
                  thisT=-1
            elif thisTin.lower() == "pgv".lower():
                  thisT=-2
            else:
                  thisT = float(thisTin);
      else:
            thisT = thisTin;
      return thisT


# Python program to check if two
# to get unique values from list
# using set

# function to get unique values
def unique(list1):

   # insert the list to the set
   list_set = set(list1)
   # convert the set to the list
   unique_list = (list(list_set))
   unique_list.sort()
   return unique_list


def getmedianLnYrefFrequencyValueColumn():
   medianLnYrefFrequencyValueColumn =
(0.1,0.1,0.1,0.1,0.1,0.1,0.1,0.1,0.1,0.1,0.1,0.133,0.133,0.133,0.133,0.133,0.133,0.133,0.133,0.133,0.133,0.133,0.2,0.2,0.2,0.2
,0.2,0.2,0.2,0.2,0.2,0.2,0.2,0.2,0.25,0.25,0.25,0.25,0.25,0.25,0.25,0.25,0.25,0.25,0.25,0.333,0.333,0.333,0.333,0.333,0.333,0.333
,0.333,0.333,0.333,0.333,0.5,0.5,0.5,0.5,0.5,0.5,0.5,0.5,0.5,0.5,0.5,0.667,0.667,0.667,0.667,0.667,0.667,0.667,0.667,0.667,0.
667,0.667,1,1,1,1,1,1,1,1,1,1,1,1.333,1.333,1.333,1.333,1.333,1.333,1.333,1.333,1.333,1.333,1.333,2,2,2,2,2,2,2,2,2,2,2,2.5,
2.5,2.5,2.5,2.5,2.5,2.5,2.5,2.5,2.5,2.5,2.5,3.333,3.333,3.333,3.333,3.333,3.333,3.333,3.333,3.333,3.333,3.333,3.333,4,4,4,4,4,4,4,4,4,4,
4,5,5,5,5,5,5,5,5,5,5,5,5,6.667,6.667,6.667,6.667,6.667,6.667,6.667,6.667,6.667,6.667,6.667,10,10,10,10,10,10,10,10,10,10,10
,13.333,13.333,13.333,13.333,13.333,13.333,13.333,13.333,13.333,13.333,13.333,20,20,20,20,20,20,20,20,20,20,20,25,25,25
,25,25,25,25,25,25,25,25,33.333,33.333,33.333,33.333,33.333,33.333,33.333,33.333,33.333,33.333,33.333,33.333,40,40,40,40,40,
```

```
40,40,40,40,40,40,50,50,50,50,50,50,50,50,50,50,50,100,100,100,100,100,100,100,100,100,100,100,-1,-1,-1,-1,-1,-1,-1,-1,-
1,-1,-1,-2,-2,-2,-2,-2,-2,-2,-2,-2,-2,-2);
   return medianLnYrefFrequencyValueColumn

def getmedianLnYrefFrequencyLabelColumn():
   medianLnYrefFrequencyLabelColumn =
("F0.100","F0.100","F0.100","F0.100","F0.100","F0.100","F0.100","F0.100","F0.100","F0.100","F0.100","F0.133","F0.133","F
0.133","F0.133","F0.133","F0.133","F0.133","F0.133","F0.133","F0.133","F0.133","F0.200","F0.200","F0.200","F0.200","F0.2
00","F0.200","F0.200","F0.200","F0.200","F0.200","F0.200","F0.250","F0.250","F0.250","F0.250","F0.250","F0.250","F0.250"
,"F0.250","F0.250","F0.250","F0.250","F0.333","F0.333","F0.333","F0.333","F0.333","F0.333","F0.333","F0.333","F0.333","F
0.333","F0.333","F0.500","F0.500","F0.500","F0.500","F0.500","F0.500","F0.500","F0.500","F0.500","F0.500","F0.500","F0.6
67","F0.667","F0.667","F0.667","F0.667","F0.667","F0.667","F0.667","F0.667","F0.667","F0.667","F1.000","F1.000","F1.000"
,"F1.000","F1.000","F1.000","F1.000","F1.000","F1.000","F1.000","F1.000","F1.333","F1.333","F1.333","F1.333","F1.333","F
1.333","F1.333","F1.333","F1.333","F1.333","F1.333","F2.000","F2.000","F2.000","F2.000","F2.000","F2.000","F2.000","F2.0
00","F2.000","F2.000","F2.000","F2.500","F2.500","F2.500","F2.500","F2.500","F2.500","F2.500","F2.500","F2.500","F2.500"
,"F2.500","F3.333","F3.333","F3.333","F3.333","F3.333","F3.333","F3.333","F3.333","F3.333","F3.333","F3.333","F4.000","F
4.000","F4.000","F4.000","F4.000","F4.000","F4.000","F4.000","F4.000","F4.000","F4.000","F5.000","F5.000","F5.000","F5.0
00","F5.000","F5.000","F5.000","F5.000","F5.000","F5.000","F5.000","F6.667","F6.667","F6.667","F6.667","F6.667","F6.667"
,"F6.667","F6.667","F6.667","F6.667","F6.667","F10.000","F10.000","F10.000","F10.000","F10.000","F10.000","F10.000","F1
0.000","F10.000","F10.000","F10.000","F13.333","F13.333","F13.333","F13.333","F13.333","F13.333","F13.333","F13.333","
F13.333","F13.333","F13.333","F20.000","F20.000","F20.000","F20.000","F20.000","F20.000","F20.000","F20.000","F20.000
","F20.000","F20.000","F25.000","F25.000","F25.000","F25.000","F25.000","F25.000","F25.000","F25.000","F25.000","F25.0
00","F25.000","F33.333","F33.333","F33.333","F33.333","F33.333","F33.333","F33.333","F33.333","F33.333","F33.333","F3
3.333","F40.000","F40.000","F40.000","F40.000","F40.000","F40.000","F40.000","F40.000","F40.000","F40.000","F40.000","
F50.000","F50.000","F50.000","F50.000","F50.000","F50.000","F50.000","F50.000","F50.000","F50.000","F50.000","F100.00
0","F100.000","F100.000","F100.000","F100.000","F100.000","F100.000","F100.000","F100.000","F100.000","F100.000","P
GA","PGA","PGA","PGA","PGA","PGA","PGA","PGA","PGA","PGA","PGA","PGV","PGV","PGV","PGV","PGV","PGV","PGV","PG
V","PGV","PGV","PGV");
   return medianLnYrefFrequencyLabelColumn

def getmedianLnYrefMomentColumn():
   medianLnYrefMomentColumn =
(4,4.5,5,5.5,6,6.5,7,7.5,7.8,8,8.2,4,4.5,5,5.5,6,6.5,7,7.5,7.8,8,8.2,4,4.5,5,5.5,6,6.5,7,7.5,7.8,8,8.2,4,4.5,5,5.5,6,6.5,7,7.5,7.8,8,
8.2,4,4.5,5,5.5,6,6.5,7,7.5,7.8,8,8.2,4,4.5,5,5.5,6,6.5,7,7.5,7.8,8,8.2,4,4.5,5,5.5,6,6.5,7,7.5,7.8,8,8.2,4,4.5,5,5.5,6,6.5,7,7.5,7.8
,8,8.2,4,4.5,5,5.5,6,6.5,7,7.5,7.8,8,8.2,4,4.5,5,5.5,6,6.5,7,7.5,7.8,8,8.2,4,4.5,5,5.5,6,6.5,7,7.5,7.8,8,8.2,4,4.5,5,5.5,6,6.5,7,7.5,7
.8,8,8.2,4,4.5,5,5.5,6,6.5,7,7.5,7.8,8,8.2,4,4.5,5,5.5,6,6.5,7,7.5,7.8,8,8.2,4,4.5,5,5.5,6,6.5,7,7.5,7.8,8,8.2,4,4.5,5,5.5,6,6.5,7,7.
5,7.8,8,8.2,4,4.5,5,5.5,6,6.5,7,7.5,7.8,8,8.2,4,4.5,5,5.5,6,6.5,7,7.5,7.8,8,8.2,4,4.5,5,5.5,6,6.5,7,7.5,7.8,8,8.2,4,4.5,5,5.5,6,6.5,7,
7.5,7.8,8,8.2,4,4.5,5,5.5,6,6.5,7,7.5,7.8,8,8.2,4,4.5,5,5.5,6,6.5,7,7.5,7.8,8,8.2,4,4.5,5,5.5,6,6.5,7,7.5,7.8,8,8.2,4,4.5,5,5.5,6,6.5
,7,7.5,7.8,8,8.2,4,4.5,5,5.5,6,6.5,7,7.5,7.8,8,8.2);
   return medianLnYrefMomentColumn


def getMedianLnYrefRrupRow():
   MedianLnYrefRrupRow =
(0,1,5,10,15,20,25,30,40,50,60,70,80,90,100,110,120,130,140,150,175,200,250,300,350,400,450,500,600,700,800,1000,120
0,1500);
   return MedianLnYrefRrupRow


def getMedianLnYrefDataTable():
   MedianLnYrefDataTable = [[-9.605511837,-9.856291163,-11.00960426,-11.82957032,-12.37243119,-12.77947739,-
13.11741114,-13.38980611,-13.81525848,-14.14286981,-14.33636957,-14.46077937,-14.54637483,-14.61907323,-
14.68526388,-14.74619915,-14.80314538,-14.86447387,-14.92591871,-14.98726788,-15.13383919,-15.2648154,-
15.50453776,-15.71111627,-15.89439328,-16.06048595,-16.21874402,-16.36542119,-16.61035701,-16.85344312,-
17.07062066,-17.4924918,-17.87393458,-18.43088267],[
      -8.632895422,-8.808995903,-9.741673551,-10.51024366,-11.02877457,-11.41642824,-11.73855713,-11.99988697,-
12.40649941,-12.7205702,-12.90340172,-13.01997491,-13.09870673,-13.16525683,-13.22570974,-13.28127306,-
13.33313003,-13.38924104,-13.44548257,-13.50189503,-13.63699049,-13.75774001,-13.97748607,-14.16657328,-
```

14.33428781,-14.48627938,-14.63131164,-14.76570622,-14.99235452,-15.21270808,-15.41772823,-15.80572415,-16.15812022,-16.67464842],[
-7.628126741,-7.754042931,-8.470501031,-9.164651811,-9.650659581,-10.01663929,-10.31773552,-10.56548549,-10.95379646,-11.2559549,-11.42998075,-11.54043378,-11.61361699,-11.67497706,-11.73064368,-11.78171109,-11.82929366,-11.88086304,-11.93247565,-11.9844441,-12.10926094,-12.22088544,-12.42343116,-12.59744808,-12.75244678,-12.89328261,-13.0279918,-13.15280403,-13.3613213,-13.56784311,-13.75631479,-14.1173078,-14.44862775,-14.92937735],[
-6.634394627,-6.728558551,-7.239586836,-7.829807262,-8.271634746,-8.610992066,-8.890940953,-9.125179644,-9.492758134,-9.779968814,-9.944571959,-10.04860333,-10.11610877,-10.17203308,-10.2227232,-10.26978155,-10.31357445,-10.36131443,-10.40946022,-10.45818147,-10.57537956,-10.68012642,-10.86967543,-11.03221346,-11.17567156,-11.30544243,-11.42885006,-11.54340737,-11.73694501,-11.9302876,-12.10562903,-12.44221981,-12.74703027,-13.19477817],[
-5.685500404,-5.760246657,-6.120002438,-6.583382757,-6.96588858,-7.273349299,-7.529593949,-7.74824096,-8.098394827,-8.374097089,-8.531679301,-8.630827331,-8.693808479,-8.745497831,-8.792300307,-8.836352187,-8.877562813,-8.922175588,-8.967356245,-9.013205604,-9.124163229,-9.223936318,-9.402643941,-9.556046412,-9.692161801,-9.815395627,-9.932163763,-10.04030563,-10.22386691,-10.4048243,-10.56802154,-10.88032707,-11.16269267,-11.57727908],[
-4.800405449,-4.86070112,-5.123329674,-5.46830614,-5.785412651,-6.057208318,-6.292867269,-6.497538491,-6.827830152,-7.089328065,-7.2399778,-7.334130278,-7.393014475,-7.441628015,-7.485504575,-7.52728048,-7.566028691,-7.608293992,-7.651076235,-7.694656372,-7.800700833,-7.896617464,-8.068964995,-8.217012507,-8.347518764,-8.465041457,-8.575885419,-8.6782273,-8.852031579,-9.020443496,-9.171549066,-9.461660097,-9.723000624,-10.0981226],[
-3.988380738,-4.039620817,-4.246495375,-4.517990876,-4.777784017,-5.008947136,-5.212912818,-5.393401335,-5.693280571,-5.929968766,-6.069708187,-6.158511616,-6.214828093,-6.261609781,-6.30410199,-6.34540248,-6.383721986,-6.425122519,-6.466745331,-6.509138548,-6.611726057,-6.704344454,-6.871637014,-7.01510075,-7.141229057,-7.254704516,-7.361242227,-7.459421836,-7.625349018,-7.784705046,-7.92793548,-8.201553061,-8.444516759,-8.792133926],[
-3.295383805,-3.337669033,-3.506119623,-3.726210342,-3.935592116,-4.126148476,-4.299410316,-4.455342252,-4.720158297,-4.926007881,-5.053726401,-5.134472259,-5.187351742,-5.231351473,-5.271402346,-5.311253123,-5.348243333,-5.388766359,-5.429898758,-5.471648582,-5.572246295,-5.662870458,-5.822877092,-5.960119597,-6.081574046,-6.191008105,-6.293356206,-6.387258822,-6.54795043,-6.70140648,-6.838945787,-7.098703411,-7.32756554,-7.652964321],[
-2.947955959,-2.986949286,-3.139420371,-3.331206605,-3.515126577,-3.684383443,-3.840019759,-3.98127727,-4.227794294,-4.414797434,-4.53588013,-4.611174698,-4.66171024,-4.704343183,-4.74325841,-4.782554002,-4.819053724,-4.858942731,-4.899254131,-4.940125071,-5.038773023,-5.127347665,-5.284798174,-5.41992718,-5.539656689,-5.647301307,-5.747702709,-5.83975969,-5.996632983,-6.145850152,-6.278515354,-6.529527144,-6.752811794,-7.065091549],[
-2.742464853,-2.781731342,-2.924817321,-3.097354212,-3.2639971,-3.420054732,-3.564478853,-3.696092182,-3.929902957,-4.102666099,-4.219155505,-4.290365389,-4.339194694,-4.380853712,-4.418964151,-4.457869461,-4.494028897,-4.533488089,-4.573261644,-4.613498275,-4.710824899,-4.797997005,-4.953807389,-5.087555659,-5.205729344,-5.312105359,-5.411298232,-5.502090051,-5.656076778,-5.802382996,-5.933492832,-6.179872946,-6.396066756,-6.701456202],[
-2.548821372,-2.591209906,-2.727436134,-2.881958851,-3.029866353,-3.172836014,-3.306113162,-3.428180029,-3.647927895,-3.80767519,-3.919688701,-3.986740634,-4.03383475,-4.074563771,-4.11180852,-4.150315822,-4.186071696,-4.225115013,-4.264337111,-4.303910382,-4.399864408,-4.485534807,-4.639545622,-4.771477809,-4.888165149,-4.993171014,-5.090848231,-5.180131736,-5.331702141,-5.473790551,-5.601495481,-5.842794097,-6.052468481,-6.34469623],[
-8.897621841,-9.14758561,-10.31341195,-11.14141252,-11.68873225,-12.09876227,-12.43951131,-12.7140415,-13.14271447,-13.47283683,-13.6671022,-13.79315589,-13.87955592,-13.95271894,-14.01938979,-14.0808104,-14.13822585,-14.19983327,-14.26165159,-14.3235169,-14.47124447,-14.60326967,-14.84525722,-15.05358232,-15.23811889,-15.40503785,-15.56438964,-15.71160556,-15.95631049,-16.19751958,-16.411736,-16.82269953,-17.18915805,-17.71674244],[
-7.931251855,-8.105834198,-9.046826291,-9.822915162,-10.34593868,-10.73675096,-11.06185351,-11.32553022,-11.73571261,-12.05268098,-12.23665483,-12.3551553,-12.43496345,-12.50225779,-12.56346343,-12.6197804,-12.67237152,-12.7290385,-12.78591865,-12.8431071,-12.97999621,-13.10242096,-13.3255822,-13.51753607,-13.68760995,-13.84151461,-13.9885201,-14.12436904,-14.35261216,-14.57325016,-14.77716015,-15.15849653,-15.50059273,-15.99514499],[

-6.944350603,-7.068535156,-7.789627709,-8.490417639,-8.980838461,-9.350049699,-9.654042698,-9.904209963,-
10.29636793,-10.60177304,-10.77727133,-10.8899062,-10.96441632,-11.02676229,-11.08341678,-11.13546767,-
11.18401263,-11.23637616,-11.28885304,-11.34181614,-11.4689678,-11.5827964,-11.78972697,-11.96755455,-
12.12586445,-12.2695625,-12.40691639,-12.53390437,-12.74558666,-12.95373429,-13.14288241,-13.50093605,-
13.82576778,-14.29119922],[
-5.983421346,-6.075679227,-6.587392012,-7.182468308,-7.628319359,-7.970894525,-8.253751454,-8.490541134,-
8.862221481,-9.152958202,-9.319360493,-9.42582882,-9.494904612,-9.551990505,-9.603840687,-9.652019383,-
9.696913839,-9.745606034,-9.794774646,-9.844642328,-9.964566891,-10.07194076,-10.2666683,-10.43369761,-
10.58113158,-10.71443663,-10.8409662,-10.95824215,-11.1559747,-11.35226726,-11.5295351,-11.86612685,-
12.16772075,-12.60627862],[
-5.078344192,-5.151278175,-5.509402548,-5.975338635,-6.361127339,-6.671582631,-6.930597403,-7.151795892,-
7.505988128,-7.785201741,-7.944650211,-8.046283077,-8.110888271,-8.163848363,-8.211917704,-8.257157195,-
8.299365401,-8.345050668,-8.391392579,-8.438527153,-8.552522572,-8.655192879,-8.839533068,-8.997714034,-
9.138167803,-9.265335715,-9.385634475,-9.496921106,-9.685666725,-9.870697345,-10.03720149,-10.35190099,-
10.63387957,-11.04479903],[
-4.244884535,-4.303478701,-4.563644731,-4.909291335,-5.22836513,-5.502203125,-5.740110772,-5.946994502,-
6.281142556,-6.546092905,-6.698695874,-6.795371153,-6.855922859,-6.905558615,-6.950675218,-6.993752704,-
7.033778921,-7.077195607,-7.121190177,-7.166106479,-7.275306501,-7.374240113,-7.552459929,-7.705393967,-
7.840368088,-7.961979953,-8.076553162,-8.182283514,-8.361739784,-8.534992043,-8.690126642,-8.985078531,-
9.248944501,-9.625232509],[
-3.490658881,-3.540401805,-3.744679555,-4.01422142,-4.2744288,-4.50687013,-4.712509305,-4.894822076,-
5.198201849,-5.438021075,-5.579711758,-5.670430773,-5.728339166,-5.776610434,-5.820560252,-5.863151109,-
5.902737877,-5.945287767,-5.988112191,-6.031824866,-6.137524065,-6.233125915,-6.406262436,-6.554516073,-
6.685060327,-6.802619761,-6.912936579,-7.014588043,-7.186405378,-7.351057654,-7.499059938,-7.779854819,-
8.028001529,-8.381508605],[
-2.860641988,-2.90076527,-3.065925102,-3.281585053,-3.489973512,-3.681041996,-3.855430168,-4.012773625,-
4.280577851,-4.488590761,-4.617732081,-4.700960298,-4.755563256,-4.800976193,-4.84240872,-4.88343834,-
4.921591091,-4.963198511,-5.00548302,-5.048494295,-5.152064971,-5.245543651,-5.411057034,-5.552774768,-
5.678415328,-5.79176661,-5.897788982,-5.995242534,-6.162518518,-6.32200485,-6.465117235,-6.733601748,-
6.969878205,-7.305617071],[
-2.549595157,-2.586627207,-2.736048269,-2.924535974,-3.105150898,-3.273735943,-3.430169585,-3.572550911,-
3.821551366,-4.009903409,-4.133048657,-4.210667313,-4.262838589,-4.306835571,-4.347081692,-4.387484332,-
4.425077265,-4.465989157,-4.507390397,-4.549452364,-4.650910499,-4.742185213,-4.904907649,-5.044317291,-
5.168079252,-5.279821001,-5.384154737,-5.479884154,-5.643370049,-5.79875414,-5.937180518,-6.197762557,-
6.429817781,-6.754961394],[
-2.367657402,-2.40521638,-2.545490625,-2.714859883,-2.87881912,-3.032778664,-3.176921283,-3.309459395,-
3.545035525,-3.719374887,-3.837865557,-3.911273342,-3.961667341,-4.004651711,-4.044054152,-4.084012295,-
4.121213645,-4.161651434,-4.202466059,-4.243842149,-4.343859824,-4.433616507,-4.594525127,-4.732413532,-
4.854960842,-4.965442685,-5.068503607,-5.162927747,-5.323550159,-5.476095344,-5.613153624,-5.869786644,-
6.095480175,-6.41559298],[
-2.1967275,-2.237733268,-2.371370213,-2.523673169,-2.668295448,-2.809283173,-2.941396096,-3.062736819,-
3.283592369,-3.445058337,-3.559004897,-3.628124832,-3.67671161,-3.718726005,-3.757222715,-3.796729017,-
3.833470795,-3.87344503,-3.91365913,-3.954319112,-4.052838737,-4.140974352,-4.29989845,-4.436381475,-
4.557322003,-4.666342033,-4.767818005,-4.860685521,-5.01895304,-5.167338941,-5.301180856,-5.553349778,-
5.773374172,-6.081966288],[
-8.000428233,-8.254022172,-9.38231526,-10.18547739,-10.72014458,-11.12237695,-11.45575189,-11.72493489,-
12.14608114,-12.47083991,-12.66561093,-12.7908555,-12.87710808,-12.9511945,-13.01863063,-13.0806837,-
13.13865176,-13.20110015,-13.26352275,-13.32570766,-13.47411534,-13.60643216,-13.8469096,-14.05313183,-
14.23512572,-14.39910192,-14.55443399,-14.69770445,-14.93629989,-15.16925903,-15.37509457,-15.77202969,-
16.12277894,-16.61996932],[
-7.009179878,-7.190128358,-8.107662033,-8.861953409,-9.373206038,-9.756378557,-10.07435467,-10.33275945,-
10.73577201,-11.04759415,-11.23232661,-11.35047342,-11.43062212,-11.49929838,-11.56173133,-11.61913918,-
11.67274526,-11.73066915,-11.7885862,-11.84652045,-11.98515038,-12.10891284,-12.33271759,-12.52460734,-
12.69409376,-12.84696078,-12.99205672,-13.12594759,-13.35161846,-13.56748638,-13.76601409,-14.13944498,-
14.47163168,-14.94485269],[
-6.036491777,-6.167803138,-6.877672931,-7.560409911,-8.040071986,-8.402054719,-8.699804266,-8.945121251,-
9.330511834,-9.630895946,-9.807322191,-9.919925797,-9.995127427,-10.05922203,-10.11747459,-10.17099235,-
10.22092233,-10.2748798,-10.32875725,-10.38283805,-10.51264096,-10.62871416,-10.83812857,-11.01763017,-

11.17696478,-11.32115146,-11.45844038,-11.58517282,-11.79741906,-12.00396775,-12.19065124,-12.54585718,-12.86544271,-13.31767881],[
-5.114334845,-5.214077576,-5.727437842,-6.310915635,-6.747865433,-7.083779236,-7.360894311,-7.592932485,-7.958067556,-8.243895568,-8.411146297,-8.517632032,-8.58751325,-8.646615306,-8.700340084,-8.750360166,-8.796998692,-8.847584356,-8.898442491,-8.949720262,-9.073004634,-9.183259578,-9.381757608,-9.551919607,-9.701871599,-9.837144013,-9.96513863,-10.08358068,-10.28460584,-10.48150477,-10.65893362,-10.99705234,-11.29877829,-11.73100132],[
-4.275999162,-4.355744198,-4.722811908,-5.184630697,-5.564153986,-5.869043648,-6.123191803,-6.340069163,-6.688289907,-6.963003756,-7.123445905,-7.225255955,-7.290861215,-7.345932105,-7.395979178,-7.443280116,-7.487719434,-7.535413072,-7.58353123,-7.632165085,-7.749817671,-7.855760401,-8.044963428,-8.207570625,-8.351646438,-8.481799572,-8.604607071,-8.718092189,-8.91171254,-9.099128305,-9.266937845,-9.586924823,-9.872824944,-10.28274254],[
-3.53209544,-3.596805693,-3.868552942,-4.21598426,-4.532533459,-4.803187159,-5.037434784,-5.240767721,-5.569576323,-5.830416757,-5.983865609,-6.080626151,-6.142143229,-6.194264919,-6.24142729,-6.286482045,-6.328401007,-6.373785292,-6.419580944,-6.466017121,-6.578980333,-6.681321108,-6.864751994,-7.02271219,-7.161954985,-7.287221554,-7.404948564,-7.513535113,-7.699140281,-7.87600385,-8.034074579,-8.336367624,-8.606490289,-8.987340396],[
-2.87835895,-2.933484945,-3.149853259,-3.426902469,-3.688498961,-3.919912863,-4.123469947,-4.303261283,-4.60203363,-4.838475656,-4.980806771,-5.072247143,-5.131056001,-5.180908135,-5.226374951,-5.270810488,-5.31216861,-5.356546726,-5.401075006,-5.446230013,-5.55553629,-5.654428608,-5.832549652,-5.9858029,-6.120662959,-6.241972805,-6.355582719,-6.460238236,-6.638674232,-6.807511657,-6.958836425,-7.247882553,-7.504046236,-7.866156562],[
-2.34713038,-2.393304265,-2.571337583,-2.799133644,-3.012140429,-3.204230031,-3.377885859,-3.533674304,-3.797897739,-4.004403519,-4.134605406,-4.217386311,-4.272450741,-4.319249591,-4.36201977,-4.404814205,-4.44465096,-4.487889807,-4.531654608,-4.575906107,-4.682614397,-4.778978836,-4.949174831,-5.095696882,-5.225438945,-5.342358937,-5.451534501,-5.551733357,-5.724494397,-5.887480789,-6.033666669,-6.311053266,-6.556608882,-6.904217259],[
-2.095892595,-2.138486208,-2.300098697,-2.499254343,-2.688152984,-2.859870439,-3.016405954,-3.15793844,-3.404214723,-3.592654851,-3.715756455,-3.792925303,-3.84552223,-3.890733301,-3.932151043,-3.974214432,-4.013387933,-4.055812984,-4.098594,-4.141813657,-4.246214851,-4.340165627,-4.507076013,-4.650870092,-4.77836114,-4.892950254,-4.999723504,-5.097770911,-5.266367779,-5.425136163,-5.566740678,-5.836562027,-6.078185756,-6.417149905],[
-1.952569688,-1.994973541,-2.146706817,-2.326599289,-2.498275095,-2.657621053,-2.80360153,-2.935767999,-3.169754522,-3.344240653,-3.462640579,-3.5356056,-3.586416305,-3.630495733,-3.67095119,-3.712494964,-3.751199581,-3.793064968,-3.835188363,-3.877662626,-3.980470925,-4.07275487,-4.237486457,-4.37943769,-4.504828201,-4.617804663,-4.723124005,-4.819705992,-4.985170211,-5.140991434,-5.280736061,-5.546879693,-5.782791171,-6.117705512],[
-1.819253853,-1.864098618,-2.008389773,-2.169871602,-2.323525642,-2.469989202,-2.605379088,-2.728869059,-2.949149936,-3.11097025,-3.224771989,-3.293452274,-3.342438979,-3.385416186,-3.42484322,-3.465853002,-3.50401999,-3.545331593,-3.586781462,-3.628474765,-3.729626245,-3.820140997,-3.982520872,-4.12199512,-4.245506172,-4.356828516,-4.460393681,-4.55527904,-4.717954309,-4.869683745,-5.006282953,-5.267992586,-5.498924108,-5.824367739],[
-7.483444855,-7.736120033,-8.856411112,-9.657132495,-10.19158371,-10.59404486,-10.92750432,-11.19694377,-11.61877811,-11.94427141,-12.14046654,-12.26572315,-12.35221457,-12.42666519,-12.49445172,-12.55683726,-12.61512427,-12.67812627,-12.74109489,-12.80369445,-12.9530696,-13.08616952,-13.32759251,-13.53440997,-13.71671417,-13.88075357,-14.0351869,-14.1775882,-14.41531004,-14.64632696,-14.85036457,-15.24180365,-15.58635329,-16.07227505],[
-6.485008511,-6.665764921,-7.578271672,-8.33057812,-8.841600875,-9.224965115,-9.54303616,-9.801701616,-10.20548637,-10.51813329,-10.70439758,-10.82275789,-10.90335157,-10.97259611,-11.03558212,-11.09352461,-11.14765207,-11.20632035,-11.26497755,-11.3235181,-11.46359274,-11.58861683,-11.81429418,-12.00767124,-12.17832921,-12.33209934,-12.47705016,-12.6108259,-12.83697584,-13.05248705,-13.25036775,-13.6212762,-13.95007544,-14.41665262],[
-5.521745314,-5.653163738,-6.360389604,-7.041695546,-7.521177294,-7.883356539,-8.18130123,-8.426921677,-8.813132934,-9.114375909,-9.292388895,-9.405350742,-9.481148589,-9.545965351,-9.604926728,-9.659135162,-9.709742503,-9.764593344,-9.819366801,-9.874210173,-10.00584721,-10.12357763,-10.33561664,-10.51732834,-10.67853069,-10.82429859,-10.96204305,-11.08925106,-11.30316965,-11.51046165,-11.69772395,-12.05278446,-12.37128414,-12.82082357],[

-4.623399694,-4.723360302,-5.236404913,-5.819387358,-6.256336656,-6.592464921,-6.869797947,-7.102112365,-
7.468051658,-7.754733146,-7.923552308,-8.030467303,-8.10101256,-8.160727759,-8.215077768,-8.265736265,-
8.313015173,-8.364459181,-8.416188632,-8.468182441,-8.593312742,-8.705428723,-8.907063368,-9.079465997,-
9.231436664,-9.368518512,-9.498314732,-9.618507524,-9.822673586,-10.02164444,-10.20066889,-10.54063513,-
10.84361507,-11.27663829],[
-3.822303535,-3.902212905,-4.270464481,-4.732832064,-5.112685539,-5.417893162,-5.67234641,-5.889525431,-
6.23814369,-6.513326166,-6.674987301,-6.776998131,-6.843102828,-6.89884488,-6.949574022,-6.997584149,-
7.042474671,-7.09113728,-7.140261028,-7.189771813,-7.309574495,-7.41752196,-7.610246305,-7.775550746,-
7.922070565,-8.054441329,-8.179304149,-8.294705705,-8.492472187,-8.68289989,-8.853770082,-9.177408042,-
9.466423893,-9.880005299],[
-3.125341102,-3.190225451,-3.463874424,-3.812713177,-4.12979201,-4.400544505,-4.634810584,-4.838172395,-
5.167300337,-5.428604236,-5.583171646,-5.680106377,-5.742112562,-5.794653756,-5.842428815,-5.888277015,-
5.930984562,-5.97735558,-6.024145795,-6.071450957,-6.186558848,-6.290918183,-6.477966725,-6.638826592,-
6.780727926,-6.908439727,-7.028453564,-7.139211584,-7.32939647,-7.509867857,-7.67144039,-7.979250902,-
8.254308678,-8.641789083],[
-2.523974133,-2.57925535,-2.797647338,-3.075869195,-3.33811763,-3.569841562,-3.77354373,-3.953408497,-
4.252389361,-4.489214765,-4.632533144,-4.723799424,-4.782994006,-4.833494994,-4.879622593,-4.924765644,-
4.96683246,-5.012119818,-5.05757037,-5.103529399,-5.214834822,-5.315627021,-5.497150933,-5.653213716,-
5.790677058,-5.914405771,-6.030308946,-6.137153301,-6.3202768,-6.492934869,-6.648072391,-6.943733888,-
7.206271703,-7.577107298],[
-2.045865442,-2.091974631,-2.271703626,-2.500526907,-2.714350894,-2.906955019,-3.080840333,-3.236731115,-
3.501094632,-3.707835667,-3.838703916,-3.921590148,-3.977035417,-4.024360131,-4.06767379,-4.111073549,-
4.15151936,-4.195556632,-4.240122529,-4.28506941,-4.393519088,-4.491555256,-4.664818625,-4.813934984,-
4.946086594,-5.065267371,-5.17660704,-5.278948646,-5.4563864,-5.623327592,-5.773462647,-6.057890316,-
6.310599245,-6.668949337],[
-1.825145877,-1.867715421,-2.030970133,-2.231706662,-2.421031678,-2.59313698,-2.749974406,-2.891662347,-
3.137988533,-3.326633772,-3.45055215,-3.527785945,-3.580701692,-3.626361188,-3.668247194,-3.71084846,-
3.750566314,-3.793725472,-3.837242544,-3.881099964,-3.987103389,-4.082584969,-4.252284515,-4.398473225,-
4.528201114,-4.644999363,-4.753928046,-4.854073741,-5.027166322,-5.189850755,-5.335471191,-5.612630789,-
5.861645136,-6.212440042],[
-1.701206163,-1.743538618,-1.896821042,-2.078396272,-2.250899979,-2.410433502,-2.55655042,-2.688912908,-
2.922823405,-3.097756969,-3.2169065,-3.289912362,-3.341002025,-3.385475702,-3.426347372,-3.468382757,-
3.507585797,-3.550141089,-3.592954728,-3.636027225,-3.740339853,-3.834055712,-4.001383307,-4.14558458,-
4.273215438,-4.388315392,-4.49569223,-4.59429462,-4.764150071,-4.923844926,-5.067483609,-5.341039442,-
5.584787314,-5.932028047],[
-1.586386106,-1.631056157,-1.776787832,-1.940204197,-2.094833277,-2.241698729,-2.377161224,-2.500571819,-
2.720671071,-2.883170918,-2.997656325,-3.066352696,-3.115577436,-3.158893185,-3.198683859,-3.24013748,-
3.278758381,-3.320714183,-3.362808487,-3.405060861,-3.507618538,-3.59946947,-3.764252562,-3.905968724,-
4.031587769,-4.144924394,-4.25045366,-4.347290899,-4.514197852,-4.66982571,-4.810341856,-5.079588461,-
5.318753399,-5.657749222],[
-6.804933471,-7.057199031,-8.230337515,-9.059175986,-9.606385763,-10.01644293,-10.3574822,-10.63234635,-
11.06177429,-11.39285578,-11.58768681,-11.71752982,-11.80741501,-11.8837251,-11.95334689,-12.01753703,-
12.07755265,-12.14102296,-12.20447371,-12.2680492,-12.41972843,-12.55545386,-12.80424606,-13.01791363,-
13.20663615,-13.37674672,-13.54150884,-13.69273152,-13.94465927,-14.18732892,-14.401459,-14.80717421,-
15.1628702,-15.65970733],[
-5.808277059,-5.98502469,-6.931591413,-7.70915796,-8.233174507,-8.624953629,-8.951227483,-9.216050905,-
9.628467364,-9.947770298,-10.13361597,-10.25702847,-10.34141722,-10.41293292,-10.47813345,-10.53824569,-
10.59444709,-10.65399454,-10.71349111,-10.77335325,-10.91654879,-11.04492754,-11.27930833,-11.48085621,-
11.65917421,-11.82020362,-11.9766203,-12.12034383,-12.36273774,-12.59257836,-12.80241936,-13.19203264,-
13.53651179,-14.02041001],[
-4.857568512,-4.98415465,-5.709615178,-6.412249191,-6.904505476,-7.275474447,-7.581349723,-7.833340308,-
8.229058232,-8.538010511,-8.716526804,-8.835016444,-8.915032297,-8.982495149,-9.044015274,-9.100716667,-
9.153708608,-9.209797905,-9.265722523,-9.322173066,-9.457627055,-9.579312303,-9.801204412,-9.99221561,-
10.16221536,-10.31635865,-10.46652199,-10.60462095,-10.83677521,-11.05953773,-11.26077619,-11.63845827,-
11.97624821,-12.4482096],[
-3.995665957,-4.090582392,-4.605380029,-5.202116735,-5.650307334,-5.995193602,-6.280543351,-6.519718007,-
6.895973653,-7.191191072,-7.36151743,-7.47454005,-7.549859046,-7.613473059,-7.671505454,-7.725587203,-
7.776112682,-7.829675916,-7.883375524,-7.937883056,-8.068535115,-8.185459545,-8.398091018,-8.582927277,-

8.746228935,-8.893857409,-9.032730556,-9.160882885,-9.382088669,-9.596054223,-9.790206139,-10.15550765,-10.48014335,-10.9404269],[
-3.240369634,-3.316049011,-3.676488685,-4.144339945,-4.532549882,-4.845461523,-5.107158582,-5.330956597,-5.691429919,-5.976741101,-6.141614554,-6.250965726,-6.322838078,-6.382589176,-6.437114414,-6.48855551,-6.537601281,-6.588307826,-6.639241079,-6.690984713,-6.816020613,-6.929125856,-7.132951198,-7.310188418,-7.467839798,-7.610713639,-7.745368602,-7.869906374,-8.083854963,-8.289430965,-8.473833687,-8.824253459,-9.136931891,-9.583006398],[
-2.594598692,-2.655657237,-2.917524432,-3.265276016,-3.587352433,-3.864818757,-4.106321379,-4.316606509,-4.657129255,-4.92805346,-5.086157948,-5.190470429,-5.25823519,-5.315436788,-5.367205989,-5.416227943,-5.461970631,-5.510408611,-5.559129684,-5.608780392,-5.72936148,-5.83905544,-6.037248849,-6.209462385,-6.362000831,-6.49979547,-6.629332658,-6.749060886,-6.955081597,-7.151022388,-7.327176641,-7.661419983,-7.961110547,-8.385395358],[
-2.048624951,-2.100602946,-2.306463624,-2.579722322,-2.843320995,-3.079137761,-3.288249552,-3.473964631,-3.783776626,-4.029607858,-4.17666386,-4.275617516,-4.340545823,-4.395096687,-4.445046624,-4.493457927,-4.538642601,-4.58610567,-4.63356574,-4.68192223,-4.798782118,-4.904909141,-5.097654065,-5.264652308,-5.41241018,-5.545960137,-5.671231783,-5.786948937,-5.985691725,-6.173986225,-6.344011295,-6.667036194,-6.955464777,-7.367059125],[
-1.630654237,-1.673354104,-1.840713797,-2.060402936,-2.271960079,-2.466032466,-2.64360022,-2.804172068,-3.078225077,-3.292459074,-3.427132053,-3.516976957,-3.577831188,-3.62914837,-3.676226519,-3.722745644,-3.766175648,-3.812395209,-3.859049405,-3.90644299,-4.020528074,-4.123894893,-4.307934644,-4.467272038,-4.609191442,-4.737779158,-4.858182366,-4.96915508,-5.162392507,-5.345256905,-5.510972801,-5.824724461,-6.10579594,-6.508857808],[
-1.442612547,-1.481870783,-1.633172734,-1.823788961,-2.008566576,-2.180477532,-2.33982044,-2.48517792,-2.740269327,-2.93469301,-3.062340998,-3.146212811,-3.204281341,-3.253853587,-3.299430431,-3.345019359,-3.38759992,-3.432839415,-3.47835485,-3.524541574,-3.635909226,-3.736568025,-3.916954547,-4.073117445,-4.212431813,-4.338539256,-4.45645501,-4.565236744,-4.754261677,-4.933025716,-5.094424867,-5.401895291,-5.680190158,-6.077131603],[
-1.339071183,-1.378717614,-1.520759789,-1.691833135,-1.858759285,-2.01653331,-2.163998877,-2.29932694,-2.541080518,-2.720674206,-2.843477322,-2.922876796,-2.978917508,-3.027244051,-3.071754295,-3.116684393,-3.158666584,-3.203230448,-3.247974192,-3.293293041,-3.402787896,-3.501563949,-3.679511279,-3.833540582,-3.970787897,-4.095192845,-4.211540149,-4.31880054,-4.504722325,-4.68060072,-4.84061908,-5.144862446,-5.418360469,-5.813285581],[
-1.243247002,-1.286220652,-1.421430505,-1.574506199,-1.721729453,-1.865867299,-2.001399337,-2.126210986,-2.353219038,-2.519216292,-2.637273535,-2.712115645,-2.766091964,-2.813213025,-2.856592903,-2.900853368,-2.942163923,-2.986059882,-3.030016885,-3.074433824,-3.18199157,-3.278775312,-3.454103326,-3.605777069,-3.740960683,-3.863574366,-3.978072007,-4.083577381,-4.266806163,-4.438784459,-4.596007341,-4.896466666,-5.166312277,-5.553882266],[
-5.891045074,-6.141113033,-7.286879973,-8.105952327,-8.651362887,-9.061293275,-9.401695159,-9.67654538,-10.10665905,-10.43871796,-10.63716055,-10.76611577,-10.85540108,-10.93169193,-11.00133224,-11.0655561,-11.125626,-11.19001467,-11.25440639,-11.31866779,-11.47201708,-11.60904578,-11.85896765,-12.07343387,-12.26272849,-12.43323004,-12.59519243,-12.74439123,-12.99763931,-13.23913925,-13.45446376,-13.86051229,-14.2194283,-14.72399225],[
-4.86751949,-5.044280053,-5.973706627,-6.743059218,-7.265062983,-7.656352733,-7.981817678,-8.24643088,-8.659413784,-8.9796052,-9.169043447,-9.291882527,-9.376017605,-9.447857888,-9.513420329,-9.573913571,-9.630525995,-9.691306759,-9.752074514,-9.812946361,-9.958633362,-10.08916596,-10.32631476,-10.53022936,-10.71063022,-10.87353218,-11.02828498,-11.17114984,-11.41659146,-11.648147,-11.8602226,-12.25485309,-12.6068325,-13.1053422],[
-3.935495458,-4.062692212,-4.779009729,-5.475277304,-5.965534985,-6.335853034,-6.641103347,-6.892869627,-7.288987236,-7.598608825,-7.780573627,-7.898663737,-7.978642372,-8.04665886,-8.10877942,-8.166106382,-8.219756056,-8.277299993,-8.334742311,-8.392454645,-8.53103888,-8.655551484,-8.881457476,-9.076015701,-9.249215986,-9.406309535,-9.555551031,-9.693523523,-9.930293502,-10.15578404,-10.36108535,-10.7468837,-11.09457175,-11.58629762],[
-3.126208015,-3.221802655,-3.735286465,-4.328915886,-4.775622081,-5.119774125,-5.40444312,-5.643153469,-6.019480684,-6.315077737,-6.488503091,-6.601090498,-6.676343612,-6.739737251,-6.797705754,-6.851882369,-6.902600349,-6.957166748,-7.011975699,-7.067246247,-7.200318251,-7.320165979,-7.537468293,-7.724516995,-7.890193476,-8.040281499,-8.182378007,-8.314275821,-8.543185282,-8.762620253,-8.962051878,-9.337751812,-9.675502793,-10.15968591],[

-2.452950845,-2.528860512,-2.892925098,-3.36130329,-3.749119434,-4.061585954,-4.322838544,-4.54619034,-
4.904875193,-5.188794712,-5.355150883,-5.462904213,-5.533805646,-5.593357148,-5.64783783,-5.699452268,-
5.747670741,-5.799636638,-5.852042626,-5.905021323,-6.033236835,-6.149351605,-6.358383232,-6.538630446,-
6.69933616,-6.845282545,-6.983135722,-7.111006112,-7.334582678,-7.54727918,-7.740660793,-8.103836442,-
8.431746832,-8.90422019],[
-1.90648546,-1.967660781,-2.235485521,-2.586795256,-2.908995294,-3.184948366,-3.42475028,-3.63342368,-
3.971799226,-4.241283916,-4.40046414,-4.503009653,-4.56970006,-4.625551263,-4.676908024,-4.726403704,-
4.772686458,-4.822394539,-4.872479737,-4.923266042,-5.04681752,-5.159369137,-5.36268128,-5.538194278,-
5.694089411,-5.835281557,-5.968321358,-6.091740575,-6.307658041,-6.51158345,-6.696747106,-7.047480986,-
7.365361758,-7.819521469],[
-1.465377096,-1.517423008,-1.729677406,-2.005098778,-2.269142642,-2.504302086,-2.712253936,-2.896603996,-
3.204030615,-3.448291853,-3.596056975,-3.691796056,-3.755310021,-3.809620895,-3.859475869,-3.908091995,-
3.953568722,-4.002054034,-4.050652716,-4.099938566,-4.219300869,-4.327900853,-4.525029333,-4.695006729,-
4.845868619,-4.982607103,-5.111238168,-5.230496698,-5.438895592,-5.635179352,-5.814456452,-6.155752059,-
6.464477994,-6.907401845],[
-1.147740744,-1.189588412,-1.36201791,-1.58302078,-1.795626934,-1.989926448,-2.166781928,-2.326234933,-
2.597936775,-2.810214599,-2.944512363,-3.032553543,-3.092235814,-3.143023967,-3.189732936,-3.236234971,-
3.279739649,-3.326711319,-3.374205322,-3.422255793,-3.538202432,-3.643482809,-3.831153376,-3.993097542,-
4.137732415,-4.269157261,-4.392600661,-4.507081222,-4.710122532,-4.901281228,-5.076284651,-5.408007256,-
5.709560892,-6.145352161],[
-1.01389074,-1.052520586,-1.208816651,-1.403311172,-1.587344237,-1.75891603,-1.917914693,-2.062446089,-
2.315003478,-2.507472482,-2.635681929,-2.717690337,-2.774510078,-2.823410237,-2.868477021,-2.913934794,-
2.956478526,-3.002354185,-3.048594837,-3.095338525,-3.208325954,-3.310625594,-3.494055628,-3.652478719,-
3.794190509,-3.923287574,-4.044519663,-4.156858232,-4.355172803,-4.54203329,-4.712710269,-5.038337841,-
5.336545055,-5.766898712],[
-0.943663827,-0.982559857,-1.129342374,-1.304839078,-1.47282682,-1.629328788,-1.775700889,-1.910423347,-
2.149208883,-2.328040273,-2.451286499,-2.528795442,-2.583553493,-2.631118358,-2.675029806,-2.719758307,-
2.761631289,-2.806761029,-2.852156976,-2.897973475,-3.008932128,-3.109173793,-3.289791175,-3.445844549,-
3.585881578,-3.71318754,-3.832662367,-3.943321758,-4.138296521,-4.322118173,-4.490914785,-4.813129071,-
5.107054955,-5.535339665],[
-0.879107685,-0.92098147,-1.060569544,-1.219286383,-1.368155735,-1.51199853,-1.646133133,-1.769066121,-
1.992793907,-2.159139798,-2.277531371,-2.350458853,-2.403122443,-2.449392315,-2.492089927,-2.536080514,-
2.577219812,-2.621612184,-2.666155277,-2.711016823,-2.819895062,-2.917995794,-3.095645992,-3.249783653,-
3.387521007,-3.512864593,-3.630334058,-3.739126114,-3.931013151,-4.111052832,-4.277064734,-4.595405323,-
4.886282563,-5.309240359],[
-5.249894183,-5.500613583,-6.658711798,-7.48292908,-8.030168805,-8.441022672,-8.78265766,-9.058391665,-
9.48971138,-9.822744108,-10.02065359,-10.15185078,-10.24378135,-10.32236215,-10.39416696,-10.4604392,-
10.5224461,-10.58815171,-10.65363049,-10.71905026,-10.87529298,-11.01530757,-11.27172527,-11.49221029,-
11.68724804,-11.86329873,-12.03426507,-12.19164526,-12.46203111,-12.717363,-12.94703956,-13.38031188,-
13.76763997,-14.31193754],[
-4.221970301,-4.398704656,-5.336331391,-6.110423294,-6.634740094,-7.027513233,-7.354684493,-7.620620266,-
8.035514477,-8.357307495,-8.546778171,-8.67226541,-8.759410394,-8.833891757,-8.901943139,-8.964800173,-
9.023657688,-9.086074083,-9.148218064,-9.210526817,-9.359780463,-9.49391188,-9.738806813,-9.949941216,-
10.13718811,-10.30670483,-10.47182906,-10.62411985,-10.88852093,-11.13656038,-11.36397222,-11.78948551,-
12.17277406,-12.71501075],[
-3.320315139,-3.447837357,-4.168727571,-4.869074806,-5.361935138,-5.73412677,-6.041292568,-6.294644685,-
6.693261958,-7.005028701,-7.187516834,-7.308565035,-7.391828534,-7.462730049,-7.527566983,-7.587474643,-
7.643575314,-7.702970303,-7.761992571,-7.821335479,-7.963963334,-8.092496854,-8.327040076,-8.529676554,-
8.710535128,-8.875009471,-9.035743404,-9.184097218,-9.44165746,-9.684589926,-9.906845677,-10.32593374,-
10.70646773,-11.24416906],[
-2.569416819,-2.665872336,-3.180097291,-3.77616067,-4.22538026,-4.571613912,-4.858439118,-5.099008733,-
5.478273983,-5.776405169,-5.950781721,-6.066561143,-6.145356296,-6.212762449,-6.274454823,-6.332113243,-
6.386108344,-6.443303293,-6.500406126,-6.558124311,-6.696709145,-6.821139936,-7.047722203,-7.245788641,-
7.42154166,-7.581090402,-7.73139459,-7.870713171,-8.119256072,-8.354945867,-8.572347165,-8.982782902,-
9.355124438,-9.886730207],[
-1.968247072,-2.045633395,-2.408461838,-2.87778634,-3.267514415,-3.58180503,-3.845010391,-4.07014576,-
4.433560374,-4.721726869,-4.890627622,-5.002741755,-5.078136315,-5.141648142,-5.199811354,-5.254866957,-
5.307576249,-5.361923592,-5.416251357,-5.471184982,-5.604224437,-5.725014101,-5.943283174,-6.134243596,-

6.304860775,-6.460188284,-6.606975801,-6.743454706,-6.985199483,-7.213885312,-7.422485262,-7.820911004,-8.183994596,-8.70535033],[
-1.496899348,-1.559686596,-1.825368629,-2.176488104,-2.500819852,-2.77995893,-3.022864481,-3.234342469,-3.577393463,-3.85077744,-4.012500635,-4.11922683,-4.19022465,-4.25107858,-4.306284696,-4.358672228,-4.407698782,-4.459590618,-4.511570013,-4.564278345,-4.692603472,-4.809771611,-5.022020647,-5.207666276,-5.373011144,-5.523218587,-5.66493234,-5.796786097,-6.030360403,-6.250363595,-6.451906431,-6.835591339,-7.187234621,-7.692593682],[
-1.125630104,-1.179158537,-1.388987822,-1.666581618,-1.932946779,-2.170562929,-2.381125734,-2.568016694,-2.879934986,-3.127914886,-3.278084915,-3.379267437,-3.447036892,-3.504606434,-3.557511115,-3.608969059,-3.657134466,-3.707753356,-3.758199892,-3.809358535,-3.93333559,-4.046333004,-4.252021354,-4.431493512,-4.591293539,-4.736625902,-4.873621245,-5.001047326,-5.226557486,-5.438687146,-5.633796203,-6.006459899,-6.348086342,-6.84288241],[
-0.866097862,-0.910404778,-1.08170278,-1.306066096,-1.520658243,-1.716659043,-1.895583003,-2.057230916,-2.333090562,-2.549485292,-2.686903903,-2.778318085,-2.841497919,-2.895414021,-2.945053603,-2.994275832,-3.040357232,-3.089374078,-3.138663025,-3.188534886,-3.308914533,-3.418358027,-3.614118646,-3.78488619,-3.937895071,-4.077444747,-4.208855152,-4.330920725,-4.549562663,-4.755446031,-4.945708109,-5.30926304,-5.644309305,-6.132966493],[
-0.762290494,-0.802991706,-0.957919051,-1.152538404,-1.340827597,-1.514896851,-1.675579036,-1.821976604,-2.078724363,-2.275393186,-2.40526491,-2.490533054,-2.550682445,-2.602584302,-2.650466153,-2.698525974,-2.743537826,-2.791350049,-2.839295531,-2.887769657,-3.004972175,-3.111305194,-3.302653507,-3.469587756,-3.619420865,-3.755900005,-3.884305001,-4.003876279,-4.217608533,-4.419179736,-4.605272992,-4.963207579,-5.294144972,-5.777246213],[
-0.709527061,-0.750421201,-0.895844207,-1.070746696,-1.240914152,-1.401195691,-1.550266503,-1.686628464,-1.930084289,-2.111786615,-2.236642106,-2.317344383,-2.375322018,-2.425806723,-2.472459451,-2.519716061,-2.563987909,-2.610980495,-2.658025613,-2.705514762,-2.820553821,-2.924747462,-3.113187436,-3.277567658,-3.42485952,-3.559335923,-3.685960685,-3.803843371,-4.014209879,-4.212719119,-4.396924189,-4.75110465,-5.078025758,-5.55890076],[
-0.660658285,-0.704567342,-0.842789932,-0.99911638,-1.149767484,-1.29634378,-1.433688876,-1.559963847,-1.788666895,-1.956745607,-2.076700915,-2.152752057,-2.20852328,-2.257639322,-2.303009876,-2.349457985,-2.392932148,-2.439123998,-2.48526405,-2.53174363,-2.64458011,-2.746556425,-2.931953316,-3.093484481,-3.238409374,-3.370938385,-3.495578638,-3.611619213,-3.81875031,-4.013793472,-4.195442886,-4.545605255,-4.870118258,-5.346927109],[
-4.359877713,-4.610030428,-5.744234456,-6.557770756,-7.101730749,-7.511110406,-7.851049333,-8.125829321,-8.556191909,-8.888940176,-9.089398128,-9.221498415,-9.315071595,-9.39567485,-9.469409695,-9.537519186,-9.60127988,-9.669062285,-9.736483849,-9.803575267,-9.963941459,-10.10786593,-10.37122409,-10.59831516,-10.79988956,-10.98249199,-11.16017453,-11.3246398,-11.61734669,-11.88910845,-12.13923145,-12.61286731,-13.04664079,-13.66358744],[
-3.343304382,-3.521635761,-4.444328605,-5.209306721,-5.730530637,-6.121751467,-6.447293502,-6.712252221,-7.126315455,-7.447887391,-7.639993927,-7.766695106,-7.85579986,-7.932611298,-8.002885189,-8.067873357,-8.128787178,-8.193548546,-8.257900554,-8.322132614,-8.476153315,-8.614835972,-8.867964475,-9.086934456,-9.281840018,-9.458997515,-9.631968245,-9.7924091,-10.0798373,-10.34747426,-10.59447455,-11.0634384,-11.49527533,-12.1127342],[
-2.510730391,-2.640983301,-3.354085255,-4.047125366,-4.537169179,-4.907853704,-5.21373606,-5.466211617,-5.864019265,-6.175444769,-6.360477031,-6.482842221,-6.568197242,-6.641555118,-6.708743383,-6.770913547,-6.829200107,-6.891052433,-6.952422721,-7.013831348,-7.161594677,-7.295055571,-7.538566393,-7.749706642,-7.938808104,-8.111450173,-8.280666262,-8.437651313,-8.719268058,-8.981166555,-9.22422902,-9.687511598,-10.11556009,-10.72843021],[
-1.857639586,-1.957452486,-2.470984689,-3.062689166,-3.509750643,-3.854521141,-4.140167492,-4.379725178,-4.757975644,-5.055543676,-5.232154316,-5.349116723,-5.429916024,-5.49980907,-5.563885926,-5.623900948,-5.680172244,-5.739877174,-5.799362568,-5.859190044,-6.003020792,-6.132451553,-6.368187081,-6.575185071,-6.759675551,-6.927919537,-7.086860721,-7.234974121,-7.507625547,-7.762011982,-8.000455429,-8.454859492,-8.875563399,-9.481279735],[
-1.365547485,-1.446235798,-1.813138039,-2.281968712,-2.670552781,-2.983595509,-3.245797046,-3.469892031,-3.832100546,-4.119490301,-4.290348984,-4.403439432,-4.480704925,-4.546528082,-4.606913407,-4.664248563,-4.719304028,-4.776049427,-4.832629299,-4.889537782,-5.027590152,-5.153227237,-5.380584628,-5.580493942,-5.7598237,-5.923801578,-6.079298242,-6.224684946,-6.489611384,-6.737010765,-6.966281042,-7.408495191,-7.819273379,-8.413029993],[

-0.998637863,-1.064532406,-1.336396533,-1.690339194,-2.01510965,-2.293787159,-2.535815998,-2.746242647,-3.087843914,-3.360186002,-3.523387622,-3.630699756,-3.70326159,-3.766322509,-3.823529211,-3.877932535,-3.928918745,-3.98299855,-4.037070517,-4.091595322,-4.224603457,-4.346329013,-4.567147167,-4.761361213,-4.935131023,-5.093775401,-5.244028236,-5.384734488,-5.640476649,-5.879453675,-6.10194068,-6.528740004,-6.927059044,-7.505775005],[
-0.717328938,-0.773613246,-0.989918399,-1.272529233,-1.540794518,-1.77868249,-1.98888666,-2.175044858,-2.485419817,-2.732384319,-2.883663189,-2.98542024,-3.054432504,-3.113553316,-3.167981196,-3.221183478,-3.271056573,-3.323614768,-3.375933581,-3.428707794,-3.556878325,-3.673974495,-3.887340632,-4.074630985,-4.242241615,-4.395481517,-4.540612536,-4.676465037,-4.923178407,-5.15375503,-5.368958235,-5.783633031,-6.171325536,-6.737996686],[
-0.527740772,-0.5749666,-0.752853186,-0.983644795,-1.201261942,-1.398302038,-1.577248176,-1.738446876,-2.013000239,-2.229094055,-2.367601437,-2.458970129,-2.522995827,-2.578189201,-2.629101419,-2.679880639,-2.727490235,-2.778218168,-2.829134301,-2.880407008,-3.004430549,-3.117453893,-3.320251851,-3.498297223,-3.658569421,-3.80552602,-3.944633615,-4.07468591,-4.313197705,-4.536601204,-4.746271512,-5.151051589,-5.531314659,-6.090936368],[
-0.458423092,-0.501800833,-0.662893283,-0.863129231,-1.055734129,-1.231620148,-1.392626392,-1.538821198,-1.794520976,-1.99155637,-2.122006284,-2.20721757,-2.26812044,-2.321153281,-2.370168121,-2.419675952,-2.46611512,-2.515530073,-2.565004552,-2.614798733,-2.735443262,-2.845169577,-3.043150287,-3.21701794,-3.373802811,-3.517261727,-3.652962305,-3.780305757,-4.013416538,-4.232378127,-4.4379898,-4.837242924,-5.21157985,-5.764550591],[
-0.425446944,-0.468675068,-0.619890481,-0.800280855,-0.97451101,-1.137477311,-1.287559227,-1.423902689,-1.666740809,-1.848905772,-1.974293198,-2.054959334,-2.113653661,-2.165183486,-2.212887544,-2.261531986,-2.307171368,-2.355704709,-2.404223159,-2.452990275,-2.571354407,-2.678836573,-2.873679412,-3.044783245,-3.198629735,-3.339914243,-3.473730592,-3.599312328,-3.828949983,-4.04473404,-4.247825054,-4.642747114,-5.01335429,-5.562598579],[
-0.394946609,-0.440635852,-0.584059275,-0.745011696,-0.900271429,-1.049585891,-1.188524807,-1.315777877,-1.544275249,-1.712990043,-1.833442864,-1.909477994,-1.965931777,-2.016010765,-2.062362631,-2.11014335,-2.154938256,-2.202617484,-2.2501858,-2.297909797,-2.413979236,-2.519169802,-2.710779998,-2.878614629,-3.029974268,-3.169290215,-3.301088218,-3.424831505,-3.650860304,-3.863599035,-4.064284349,-4.454802398,-4.823426622,-5.370292908],[
-3.721200854,-3.969255641,-5.099097187,-5.914454918,-6.461791343,-6.873985113,-7.216288263,-7.493073424,-7.926670397,-8.262152561,-8.465035761,-8.598745987,-8.693550791,-8.775361973,-8.850346705,-8.919712354,-8.984708485,-9.053898333,-9.122729567,-9.191315556,-9.355537696,-9.503363053,-9.774632181,-10.00958147,-10.21913899,-10.40987987,-10.59588578,-10.76910146,-11.08746968,-11.3791468,-11.65321996,-12.17164727,-12.65585256,-13.35197926],[
-2.7367934,-2.913650407,-3.834172198,-4.601176646,-5.125603356,-5.519493341,-5.847395353,-6.114335937,-6.531657897,-6.855993684,-7.050562737,-7.179039874,-7.269529755,-7.347710807,-7.419369311,-7.485755315,-7.54806435,-7.614381407,-7.680273951,-7.746115274,-7.904288898,-8.04717197,-8.308796033,-8.53619607,-8.739576099,-8.925368456,-9.106858742,-9.276274063,-9.588557562,-9.877889602,-10.1467638,-10.66138859,-11.14369983,-11.84022045],[
-1.973503244,-2.102613755,-2.814128092,-3.509128913,-4.002179769,-4.375425638,-4.68360611,-4.938013667,-5.339018038,-5.653133947,-5.8405688,-5.964748359,-6.051552848,-6.126329948,-6.194950761,-6.258567085,-6.318293262,-6.381745721,-6.444712393,-6.507783387,-6.659852215,-6.797662995,-7.049919213,-7.269698899,-7.467449577,-7.648875513,-7.826439533,-7.992121174,-8.298246383,-8.580002243,-8.845043967,-9.352933535,-9.828838096,-10.51876108],[
-1.401342813,-1.500130785,-2.012060159,-2.605347716,-3.055152397,-3.402277929,-3.690133917,-3.93148745,-4.312750408,-4.612850324,-4.791718576,-4.910434207,-4.992643792,-5.063701272,-5.128985922,-5.190258992,-5.247802986,-5.308947538,-5.369865745,-5.431166616,-5.579001219,-5.712705672,-5.957200405,-6.172353999,-6.365161853,-6.541981066,-6.709875373,-6.86734597,-7.164138514,-7.437883773,-7.697663092,-8.194969811,-8.662870287,-9.342858565],[
-0.984964007,-1.06462842,-1.430742389,-1.901155285,-2.292104512,-2.607136007,-2.871290058,-3.096969145,-3.461164847,-3.750169108,-3.922588105,-4.036829364,-4.115037821,-4.181869167,-4.243316161,-4.301798793,-4.357608114,-4.41578644,-4.473834178,-4.532277223,-4.674412802,-4.8042049,-5.0401567,-5.247947948,-5.435380754,-5.607734191,-5.771906046,-5.926266367,-6.214612799,-6.481384841,-6.732494166,-7.216753686,-7.673249011,-8.338796029],[
-0.679640062,-0.744545377,-1.016411527,-1.37234223,-1.698639639,-1.978071907,-2.220930705,-2.432045697,-2.774951719,-3.048478648,-3.212928612,-3.321081159,-3.39435924,-3.457715478,-3.515673016,-3.571244273,-3.623419099,-3.678872447,-3.734325533,-3.790297657,-3.927226964,-4.0529728,-4.282059321,-4.483826895,-

4.665435002,-4.832256242,-4.991019544,-5.14065673,-5.41877767,-5.677372894,-5.921503035,-6.39066195,-6.83424306,-7.484786567],[
-0.445696747,-0.501155623,-0.717723608,-0.999835113,-1.26851606,-1.506836608,-1.717630141,-1.904330289,-2.215645375,-2.463548317,-2.615850323,-2.71743373,-2.786885697,-2.846857817,-2.902196738,-2.956427661,-3.007366621,-3.061182188,-3.11477852,-3.168899897,-3.300717525,-3.421555344,-3.642656065,-3.837032555,-4.01211369,-4.173207034,-4.326646644,-4.471178258,-4.739528898,-4.989374205,-5.22580215,-5.682865978,-6.115762235,-6.752037901],[
-0.293087653,-0.338705676,-0.515953869,-0.743741344,-0.960859224,-1.158113296,-1.337364061,-1.498928982,-1.774222339,-1.990640683,-2.129448481,-2.221491038,-2.286115851,-2.342018151,-2.393712536,-2.445405563,-2.493974053,-2.545849296,-2.597945572,-2.650472375,-2.777868996,-2.89433927,-3.104454434,-3.289301905,-3.456739949,-3.611279025,-3.758432176,-3.897128073,-4.157365044,-4.400172056,-4.630963693,-5.07737467,-5.50189502,-6.129519814],[
-0.239628186,-0.281601362,-0.442182813,-0.640976233,-0.830705737,-1.005729788,-1.166920528,-1.313365164,-1.569414384,-1.766257808,-1.897813799,-1.983603015,-2.045034118,-2.098706549,-2.148438972,-2.198793169,-2.246130476,-2.296628154,-2.347230469,-2.398224357,-2.522109264,-2.635183559,-2.840291724,-3.020846459,-3.184692785,-3.335994428,-3.480082291,-3.616304393,-3.870813579,-4.109174103,-4.33598311,-4.777043221,-5.193609212,-5.812978184],[
-0.215407864,-0.257374743,-0.408200682,-0.587461128,-0.759851194,-0.920599304,-1.069843013,-1.206374033,-1.448938655,-1.631544106,-1.757996042,-1.839179279,-1.898365457,-1.950498088,-1.998883387,-2.048338629,-2.094839333,-2.144420393,-2.194036621,-2.243977581,-2.365504576,-2.476272494,-2.678149856,-2.855871551,-3.017279649,-3.166493724,-3.308677564,-3.443131198,-3.694158856,-3.929243819,-4.153027513,-4.589031776,-5.001842854,-5.615851177],[
-0.193206796,-0.237679233,-0.380620609,-0.541356262,-0.694546743,-0.842083597,-0.979455164,-1.105270822,-1.33294045,-1.502675395,-1.624157296,-1.700641018,-1.757550586,-1.808200013,-1.855207144,-1.90376493,-1.949393975,-1.99809112,-2.046736963,-2.095614191,-2.214784722,-2.323223976,-2.521797143,-2.69688601,-2.855755837,-3.003069904,-3.14327329,-3.275935193,-3.523044905,-3.755552808,-3.97708731,-4.408085904,-4.819273663,-5.432205991],[
-2.890333243,-3.134750957,-4.222614709,-5.02405629,-5.570806507,-5.984289068,-6.326588827,-6.604164311,-7.039971326,-7.378086482,-7.586502019,-7.721545454,-7.819428748,-7.904485376,-7.982639085,-8.055079019,-8.123050838,-8.196120578,-8.268540385,-8.340268041,-8.512225759,-8.667591065,-8.952786803,-9.201612671,-9.425372913,-9.630716878,-9.829554516,-10.01696033,-10.37997889,-10.70598153,-11.02201504,-11.62101913,-12.19495649,-13.03036827],[
-2.00066692,-2.177782058,-3.070877276,-3.826005654,-4.349339171,-4.743788792,-5.07134129,-5.338622852,-5.757681523,-6.084161972,-6.283815129,-6.413667006,-6.507331195,-6.588862323,-6.663769271,-6.733320606,-6.798724789,-6.869001834,-6.938558069,-7.007598061,-7.173692646,-7.324323508,-7.600377451,-7.842131181,-8.060050046,-8.260760514,-8.455214597,-8.638924731,-8.993525051,-9.319589178,-9.625778237,-10.22031034,-10.79015465,-11.62148858],[
-1.359150208,-1.490188173,-2.186232995,-2.871613773,-3.363380991,-3.736813309,-4.044873716,-4.299470911,-4.701739269,-5.017359411,-5.209276222,-5.334670187,-5.424574466,-5.50262649,-5.57443288,-5.641163388,-5.703937279,-5.771287507,-5.837913667,-5.904183275,-6.064219506,-6.209850625,-6.47666531,-6.710859701,-6.923074883,-7.119271673,-7.309534428,-7.489039179,-7.836657716,-8.151597377,-8.453805181,-9.038599332,-9.596145949,-10.41488443],[
-0.89442538,-0.99582399,-1.503847987,-2.091328841,-2.540166757,-2.887055382,-3.174686237,-3.415727864,-3.7974546,-4.098302201,-4.280784681,-4.400286072,-4.485242877,-4.558836052,-4.626650452,-4.690577341,-4.75075397,-4.815359967,-4.87949445,-4.94349586,-5.098452288,-5.239662058,-5.498525995,-5.726747057,-5.933119704,-6.12406801,-6.306937289,-6.480312369,-6.818602151,-7.125088054,-7.420861447,-7.99167116,-8.539236496,-9.340788147],[
-0.56353832,-0.645466934,-1.015796321,-1.486556141,-1.877656645,-2.192365191,-2.456303444,-2.681387834,-3.044068486,-3.331934545,-3.506247084,-3.620044374,-3.700062785,-3.769148875,-3.832867201,-3.893919662,-3.95163878,-4.013300031,-4.074645651,-4.13594136,-4.285373113,-4.422397152,-4.672596133,-4.893533784,-5.09446908,-5.280798989,-5.459464135,-5.628977801,-5.957598139,-6.257043851,-6.545009019,-7.101087514,-7.633901115,-8.414448072],[
-0.322292849,-0.389158617,-0.669057282,-1.03098343,-1.358787086,-1.637423985,-1.878862231,-2.088230058,-2.428732691,-2.700481944,-2.865907199,-2.973055752,-3.047717749,-3.112389375,-3.17212058,-3.230246209,-3.284964767,-3.343746741,-3.40232712,-3.460987079,-3.604872281,-3.737542324,-3.980305201,-4.194972237,-4.389904941,-4.570577858,-4.743736348,-4.908548935,-5.22512989,-5.516732483,-5.797155329,-6.337731858,-6.855489688,-7.618026191],[

-0.135415715,-0.192539133,-0.417824088,-0.706300544,-0.977603887,-1.216202022,-1.426400727,-1.611946005,-1.920843943,-2.1670938,-2.319981131,-2.419646581,-2.489895635,-2.551355938,-2.608266058,-2.664678389,-2.717826545,-2.774634309,-2.831074855,-2.887631588,-3.025823629,-3.1530558,-3.386741633,-3.5933175,-3.781167845,-3.955606893,-4.123094288,-4.282344787,-4.587721738,-4.869934013,-5.141482955,-5.668519312,-6.173972573,-6.916426652],[

-0.015659482,-0.062709081,-0.248057999,-0.482419409,-0.703224493,-0.901909286,-1.081113466,-1.242002664,-1.515422471,-1.730784793,-1.869719434,-1.960380682,-2.025558721,-2.082531301,-2.135424665,-2.189066936,-2.239627351,-2.294182943,-2.348792252,-2.403476384,-2.536545361,-2.658759241,-2.880806531,-3.07754231,-3.257289893,-3.424711223,-3.585479612,-3.738642434,-4.034984598,-4.309427491,-4.574502033,-5.088758899,-5.582826766,-6.311750998],[

0.022684948,-0.020626707,-0.18884207,-0.395043281,-0.588508091,-0.76524763,-0.926826166,-1.072932458,-1.327231747,-1.523540356,-1.655405077,-1.73985015,-1.801653124,-1.856171718,-1.906891314,-1.959050465,-2.00824639,-2.061271919,-2.114260757,-2.167305565,-2.296613368,-2.41521686,-2.63176372,-2.823904029,-2.999759655,-3.163820433,-3.321508,-3.472191357,-3.762075154,-4.031859978,-4.292948174,-4.801310765,-5.283807642,-6.001084739],[

0.038284462,-0.004672112,-0.162687822,-0.349653642,-0.526590187,-0.68929775,-0.839112188,-0.975565734,-1.216527663,-1.399328039,-1.526005403,-1.605912336,-1.665375018,-1.718215176,-1.767459175,-1.818633121,-1.86690844,-1.918926325,-1.970844547,-2.022787254,-2.149574037,-2.265737594,-2.478751747,-2.667846456,-2.841297313,-3.003172544,-3.158825741,-3.307646139,-3.593948591,-3.860212021,-4.117287354,-4.619283154,-5.097720835,-5.806379874],[

0.051018244,0.006297417,-0.142919536,-0.311076097,-0.469609437,-0.619715886,-0.758054075,-0.883967805,-1.110293076,-1.280913199,-1.402553559,-1.477824826,-1.534915953,-1.586134686,-1.633884605,-1.684079265,-1.731413789,-1.782458174,-1.833342441,-1.884177768,-2.008469478,-2.122199117,-2.331648248,-2.518168884,-2.688910872,-2.848885413,-3.002537015,-3.149575164,-3.43134842,-3.69568253,-3.950611297,-4.446392586,-4.923083629,-5.6313148],[

-2.460524787,-2.703133854,-3.784720046,-4.587313075,-5.137241313,-5.553439479,-5.897898911,-6.177378549,-6.616341871,-6.957342727,-7.168173301,-7.306012351,-7.406530085,-7.494080466,-7.574731015,-7.649639243,-7.720023659,-7.795440741,-7.870152942,-7.94424918,-8.122400909,-8.284089063,-8.582424835,-8.84438152,-9.081494813,-9.300469359,-9.515493844,-9.719151033,-10.12474808,-10.48403878,-10.83735963,-11.50562592,-12.15178947,-13.09119642],[

-1.643410889,-1.819664997,-2.709899702,-3.466461548,-3.992933486,-4.39000041,-4.719760513,-4.988941851,-5.411211209,-5.740602093,-5.942643273,-6.075384238,-6.171756871,-6.255871321,-6.333327171,-6.405406553,-6.473300437,-6.545982877,-6.617869002,-6.689295711,-6.861654854,-7.01866666,-7.307955721,-7.562938958,-7.794200388,-8.008502505,-8.218757641,-8.418351519,-8.813075059,-9.17323876,-9.512806159,-10.17499241,-10.81396897,-11.74488009],[

-1.069219365,-1.200314125,-1.894760921,-2.581740183,-3.076598658,-3.452627281,-3.762961145,-4.019467029,-4.425003021,-4.74351644,-4.937761353,-5.066053036,-5.158706885,-5.239355908,-5.313723603,-5.382994233,-5.448262539,-5.518028816,-5.587014849,-5.655696326,-5.822072211,-5.974139196,-6.254244546,-6.501645072,-6.727102676,-6.936718704,-7.142148168,-7.336819906,-7.723129623,-8.069293111,-8.40404568,-9.053291089,-9.675455363,-10.58878615],[

-0.653652908,-0.755531864,-1.262179995,-1.850815355,-2.302588075,-2.651933771,-2.941894869,-3.184792383,-3.56973,-3.873414816,-4.058135735,-4.1804914,-4.268204476,-4.344568226,-4.415101026,-4.48173509,-4.544560815,-4.611724256,-4.678319595,-4.744849668,-4.906376379,-5.054108626,-5.326333323,-5.568245431,-5.788293637,-5.993067171,-6.189769219,-6.377142469,-6.75207335,-7.087483547,-7.414505596,-8.046473403,-8.6562015,-9.546364759],[

-0.353872347,-0.436264641,-0.806759873,-1.279356654,-1.673142695,-1.98998322,-2.256070146,-2.482846164,-2.848317361,-3.138667792,-3.315019059,-3.431439034,-3.514088212,-3.585739264,-3.651992919,-3.715627954,-3.77587638,-3.840015715,-3.903745586,-3.96746748,-4.123356126,-4.26689127,-4.530447769,-4.764663453,-4.978930441,-5.178744804,-5.371123146,-5.554593708,-5.917932155,-6.245621997,-6.563628492,-7.17837904,-7.7699429,-8.634413811],[

-0.130299228,-0.197415769,-0.478516675,-0.843429293,-1.173498288,-1.453489104,-1.696224761,-1.906629704,-2.249139458,-2.522772149,-2.68981834,-2.799148719,-2.87613895,-2.942873977,-3.004861397,-3.065475612,-3.122654535,-3.183862019,-3.244806211,-3.305862806,-3.456186582,-3.59535161,-3.851314674,-4.078795349,-4.286690562,-4.480556308,-4.66717668,-4.845845255,-5.195619895,-5.515394514,-5.825577392,-6.422966385,-6.996373404,-7.84081938],[

0.046242008,-0.011104342,-0.237914808,-0.52715102,-0.799668035,-1.039211276,-1.250475282,-1.43696826,-1.74752582,-1.995446954,-2.149752104,-2.250659059,-2.322864233,-2.386550059,-2.445687185,-2.504451582,-2.559942251,-2.619074916,-2.677798359,-2.736672068,-2.881079441,-3.014571865,-3.260968101,-3.479680789,-

3.679948321,-3.86711633,-4.047737719,-4.220428283,-4.557928723,-4.867515387,-5.167689761,-5.749817154,-6.309395818,-7.128665764],[
0.156787851,0.110700124,-0.07510666,-0.306905147,-0.527776835,-0.727174349,-0.907123579,-1.068813286,-1.343797772,-1.560292866,-1.699959817,-1.792497632,-1.859669889,-1.918681559,-1.973640836,-2.029499616,-2.082279001,-2.139034988,-2.19581473,-2.252716595,-2.391694119,-2.519835086,-2.754169421,-2.962556895,-3.154220026,-3.33388866,-3.507361413,-3.673731713,-4.001509632,-4.302969729,-4.595835831,-5.163216905,-5.708589244,-6.510692595],[
0.192315868,0.149929407,-0.018705787,-0.223779987,-0.414359864,-0.590535656,-0.752725949,-0.899499994,-1.154930826,-1.35172518,-1.485175951,-1.57140621,-1.635094099,-1.691548691,-1.744235999,-1.798518646,-1.849848502,-1.904985729,-1.960076801,-2.015270139,-2.150305031,-2.274701551,-2.503284456,-2.706824254,-2.894357459,-3.070847822,-3.241482252,-3.405511296,-3.72616672,-4.022653144,-4.311510954,-4.872209577,-5.402731393,-6.190454368],[
0.206432172,0.164328101,0.005845218,-0.180285061,-0.355382274,-0.515758974,-0.664934586,-0.801978588,-1.043321241,-1.227306992,-1.355547395,-1.437179535,-1.498461969,-1.553173198,-1.604323428,-1.65756172,-1.707914367,-1.761990759,-1.815964928,-1.870010936,-2.002404267,-2.124266001,-2.349154792,-2.54948218,-2.735103932,-2.909456672,-3.077947798,-3.240046723,-3.556885903,-3.849525835,-4.133452605,-4.685964186,-5.212269753,-5.987804342],[
0.216709562,0.172882481,0.02347821,-0.144946771,-0.30116189,-0.449333718,-0.586024892,-0.710436959,-0.936456688,-1.108921543,-1.232099773,-1.309019908,-1.367861108,-1.420886151,-1.470489529,-1.522690272,-1.572049485,-1.625099612,-1.677996318,-1.730898867,-1.860685545,-1.980046864,-2.201215325,-2.399650257,-2.582475702,-2.754933379,-2.921388,-3.081685332,-3.393240393,-3.684233693,-3.965736275,-4.510719675,-5.034656774,-5.810462307],[
-1.973193989,-2.217117582,-3.274220762,-4.065796124,-4.613704105,-5.029788176,-5.37357912,-5.653061335,-6.092711312,-6.435117049,-6.649783713,-6.791638125,-6.896232757,-6.988434534,-7.073629685,-7.152938793,-7.227550722,-7.307108996,-7.385681828,-7.463666111,-7.652059736,-7.823954646,-8.14190025,-8.423193896,-8.679757518,-8.918389864,-9.154129172,-9.379086485,-9.841627797,-10.24547013,-10.64863966,-11.40842316,-12.14837732,-13.21991188],[
-1.261780194,-1.44028435,-2.316978342,-3.064293111,-3.588940402,-3.985613359,-4.314756301,-4.583783825,-5.006658822,-5.337249186,-5.542884393,-5.67966782,-5.780113922,-5.868898391,-5.950854147,-6.027315072,-6.099467481,-6.176293775,-6.252028732,-6.327296133,-6.509830414,-6.676974571,-6.985858296,-7.260116769,-7.510594244,-7.74427525,-7.974627727,-8.194881333,-8.642812752,-9.047780247,-9.431481299,-10.18192897,-10.90826037,-11.96346702],[
-0.761860312,-0.895980918,-1.583374093,-2.26320806,-2.756545582,-3.132220827,-3.442331483,-3.698779922,-4.104889887,-4.424331661,-4.621867886,-4.754103991,-4.850789419,-4.936031103,-5.014834049,-5.08842813,-5.157889257,-5.231741223,-5.304568962,-5.377087844,-5.553638787,-5.715824053,-6.015474169,-6.281946712,-6.526258212,-6.754766463,-6.979420564,-7.193630221,-7.630841008,-8.018250032,-8.395080563,-9.127108159,-9.830605534,-10.85967574],[
-0.389550052,-0.495034364,-1.000188733,-1.58410936,-2.035018571,-2.383962049,-2.673880692,-2.916517506,-3.301790682,-3.606128078,-3.793693362,-3.91974535,-4.011320245,-4.092085214,-4.166886529,-4.237782521,-4.304744336,-4.375914825,-4.446230929,-4.516465188,-4.687947277,-4.845654867,-5.137278669,-5.397963857,-5.636656706,-5.860164763,-6.07581522,-6.282475561,-6.706448847,-7.080117638,-7.447129327,-8.156931571,-8.843353093,-9.838814828],[
-0.116276054,-0.201924401,-0.577122597,-1.049724553,-1.443552093,-1.759934308,-2.026000378,-2.252368664,-2.617112909,-2.90711918,-3.085474645,-3.204830768,-3.290832566,-3.366551444,-3.436776297,-3.504521535,-3.568547558,-3.636623319,-3.704022415,-3.771404661,-3.937210247,-4.09057338,-4.373510909,-4.626213615,-4.858791714,-5.07691581,-5.287837045,-5.490055401,-5.899868632,-6.264399063,-6.621524485,-7.310237187,-7.973129616,-8.934913321],[
0.090598867,0.020724076,-0.268375267,-0.638237152,-0.969431369,-1.248680299,-1.490388701,-1.699498698,-2.040389869,-2.313024492,-2.48135513,-2.592908605,-2.672794957,-2.742818013,-2.808298641,-2.872873038,-2.933927028,-2.998930025,-3.063439481,-3.128037657,-3.288118572,-3.436993099,-3.711880805,-3.957355705,-4.183173038,-4.395050813,-4.599864245,-4.797056766,-5.191094793,-5.54728661,-5.895430866,-6.563909389,-7.203319882,-8.14116591],[
0.255858628,0.196027574,-0.039613178,-0.333631596,-0.607821849,-0.847328677,-1.058206929,-1.243933042,-1.552915886,-1.800056939,-1.955269199,-2.057326814,-2.131879303,-2.198839381,-2.261224716,-2.323641182,-2.382731896,-2.445407966,-2.507498197,-2.569724226,-2.723380855,-2.866057218,-3.130316063,-3.366022711,-3.583400645,-3.787844395,-3.986049251,-4.176547438,-4.556597253,-4.901145525,-5.237266865,-5.886896128,-6.509542715,-7.413628271],[

0.356861495,0.309126882,0.115512485,-0.119386632,-0.342359622,-0.542683318,-0.722660251,-0.884023927,-1.15807352,-1.374276584,-1.514234288,-1.608332911,-1.677725061,-1.739627644,-1.797496405,-1.856770672,-1.912931194,-1.972960736,-2.032847929,-2.092867395,-2.240409271,-2.377024549,-2.628402668,-2.853095169,-3.061083223,-3.257249928,-3.447563512,-3.631135168,-3.999662604,-4.335064401,-4.662173997,-5.293263737,-5.897177171,-6.779322867],[

0.388445512,0.344527443,0.168523713,-0.040766436,-0.231932038,-0.408657599,-0.571205389,-0.717908489,-0.972422337,-1.169117168,-1.303435231,-1.391115979,-1.456874031,-1.515989181,-1.571365757,-1.628894512,-1.68344768,-1.741694235,-1.799771465,-1.857968734,-2.001260877,-2.133894508,-2.379018295,-2.59844223,-2.801890724,-2.994783883,-3.182224441,-3.363425464,-3.723688232,-4.05371881,-4.376764494,-5.000023201,-5.584102623,-6.448035054],[

0.400149578,0.356746424,0.19124055,0.000592915,-0.176139525,-0.336272571,-0.485288133,-0.622447791,-0.862662996,-1.047536437,-1.176611594,-1.259634008,-1.322913782,-1.380140737,-1.433841217,-1.490218254,-1.543687473,-1.600768491,-1.657648986,-1.714628284,-1.855067418,-1.985001007,-2.226097336,-2.441999821,-2.643739231,-2.83440239,-3.019471826,-3.198564189,-3.554617258,-3.880264775,-4.197577387,-4.80895599,-5.388844491,-6.234833708],[

0.406146066,0.361588848,0.20602386,0.032696795,-0.125358002,-0.273909506,-0.410128435,-0.533615198,-0.758395348,-0.932665266,-1.056648326,-1.134874457,-1.195632744,-1.2510295,-1.30305242,-1.35828707,-1.410665768,-1.466614038,-1.522345575,-1.578115788,-1.715746693,-1.843047445,-2.080104673,-2.294435975,-2.493144353,-2.681845281,-2.864752711,-3.041905512,-3.391459805,-3.715976423,-4.030484127,-4.631951891,-5.208275669,-6.05647346],[

-1.676980126,-1.918494902,-2.979523965,-3.780383243,-4.335198998,-4.756290235,-5.104504724,-5.387647132,-5.832983188,-6.180173011,-6.397028728,-6.541170944,-6.648864184,-6.743761358,-6.831703786,-6.913754742,-6.991082621,-7.073243405,-7.154281043,-7.23477321,-7.43031259,-7.609722561,-7.943720321,-8.241339301,-8.51461246,-8.770364511,-9.027286735,-9.273342619,-9.78899196,-10.23466349,-10.68262159,-11.52283303,-12.3417063,-13.52140464],[

-1.031232922,-1.207061743,-2.088315422,-2.843968412,-3.375299196,-3.776994417,-4.110718983,-4.383469198,-4.8121403,-5.147573152,-5.355361163,-5.494532017,-5.598100912,-5.689647715,-5.774353615,-5.853576037,-5.928471203,-6.007912375,-6.08611094,-6.163863358,-6.353519378,-6.528117614,-6.85301774,-7.143500737,-7.410445312,-7.660934672,-7.911864724,-8.152587363,-8.650732647,-9.0969322,-9.521028559,-10.34870115,-11.14860404,-12.30533093],[

-0.570429637,-0.702841213,-1.392220865,-2.079528845,-2.579293262,-2.960015312,-3.274673711,-3.534829756,-3.946853218,-4.271190618,-4.470852527,-4.605535314,-4.705406749,-4.793427442,-4.874998801,-4.951371672,-5.023580453,-5.100063273,-5.175396978,-5.250434095,-5.434194977,-5.603879182,-5.919573844,-6.202179509,-6.462738726,-6.707704157,-6.952143923,-7.18590796,-7.670754633,-8.096883673,-8.511920496,-9.316092588,-10.0877312,-11.21183222],[

-0.222536148,-0.327186165,-0.830454701,-1.419348348,-1.876120854,-2.229931795,-2.524444955,-2.770750811,-3.161939872,-3.471175371,-3.660867923,-3.789417997,-3.88427815,-3.96836417,-4.046418551,-4.120520906,-4.190625114,-4.264790671,-4.337913643,-4.411013189,-4.590334307,-4.755736789,-5.063502533,-5.341513591,-5.597379488,-5.838101546,-6.070842198,-6.294527527,-6.76305929,-7.171308686,-7.574722759,-8.351653662,-9.102462949,-10.18401598],[

0.036597944,-0.048835427,-0.42288201,-0.899282616,-1.298180928,-1.618827761,-1.889081873,-2.118848123,-2.489561665,-2.784584696,-2.965384928,-3.087382301,-3.176876995,-3.255745921,-3.329080038,-3.399922831,-3.467378107,-3.538333558,-3.608383537,-3.678404179,-3.851821229,-4.012988361,-4.312282054,-4.581952256,-4.831424234,-5.066455811,-5.294533001,-5.514033887,-5.965647999,-6.363382538,-6.753934101,-7.505675587,-8.227908074,-9.269123777],[

0.237015777,0.167131819,-0.122014123,-0.495929545,-0.831443954,-1.114293848,-1.359426801,-1.571418742,-1.91729025,-2.194084952,-2.364451454,-2.47817254,-2.561258322,-2.634326327,-2.702758924,-2.770224834,-2.834160351,-2.90201031,-2.969211308,-3.03647115,-3.204345035,-3.361218892,-3.652560787,-3.914687227,-4.157165429,-4.38578157,-4.607587236,-4.821892086,-5.256090663,-5.644738701,-6.025901464,-6.753279265,-7.446260472,-8.459465959],[

0.398469403,0.338559479,0.102655078,-0.193415512,-0.470230179,-0.712206975,-0.925789192,-1.114000372,-1.427233412,-1.677977021,-1.834967942,-1.938702705,-2.016115246,-2.085825657,-2.150979857,-2.21621715,-2.278133583,-2.343619569,-2.408368606,-2.473226936,-2.634536945,-2.78502005,-3.065298053,-3.316923215,-3.550390681,-3.771075993,-3.985817294,-4.192959518,-4.611764705,-4.987614228,-5.354800698,-6.059374533,-6.73256639,-7.705184399],[

0.49549667,0.448557096,0.255546994,0.022296494,-0.201582145,-0.403545308,-0.585389606,-0.748745215,-1.026560987,-1.245621738,-1.386863376,-1.482797319,-1.554895382,-1.619383635,-1.679888621,-1.741862993,-1.800743604,-1.863489387,-1.925955928,-1.988551543,-2.143471493,-2.28755188,-2.554478417,-2.794468541,-

3.017895875,-3.229694171,-3.43595731,-3.635647283,-4.041302662,-4.406672931,-4.762956663,-5.445810029,-6.09605904,-7.042421197],[
0.527122012,0.484008886,0.308666893,0.100717812,-0.088753397,-0.26582157,-0.429803514,-0.57810218,-0.835769301,-1.034428674,-1.170466192,-1.259977822,-1.328306708,-1.389912143,-1.447838816,-1.507985064,-1.565186275,-1.62606804,-1.686664951,-1.747383346,-1.897888881,-2.03789408,-2.298423825,-2.532918266,-2.751589391,-2.960147674,-3.163588157,-3.360902399,-3.757304004,-4.116746252,-4.468823228,-5.142228612,-5.768953133,-6.694334766],[
0.539310041,0.496588167,0.331664856,0.142365315,-0.032994157,-0.191897753,-0.341247661,-0.479750077,-0.72235914,-0.909327362,-1.040070346,-1.12493058,-1.190686629,-1.250336218,-1.306529086,-1.365463698,-1.421528041,-1.481188867,-1.540540738,-1.600001834,-1.74753225,-1.88475949,-2.141144077,-2.371962064,-2.589205703,-2.795578902,-2.996506017,-3.19161098,-3.583255872,-3.937840474,-4.282835703,-4.94180675,-5.564298353,-6.467258085],[
0.544640037,0.500715224,0.346051269,0.173611573,0.017703159,-0.12961325,-0.265170884,-0.388228873,-0.614594253,-0.791164715,-0.91676088,-0.996799129,-1.05992582,-1.11767061,-1.172128676,-1.229859146,-1.284781559,-1.343249727,-1.401407021,-1.459624336,-1.604232165,-1.738764635,-1.991011422,-2.220721055,-2.434899758,-2.639296159,-2.838020455,-3.031119342,-3.414869738,-3.768532221,-4.110005206,-4.757249394,-5.374898459,-6.28175527],[
-1.401245056,-1.632619248,-2.63323279,-3.413961926,-3.970148641,-4.394773668,-4.744157444,-5.029103566,-5.47830021,-5.829557132,-6.054678032,-6.202309405,-6.31575168,-6.416091985,-6.509259136,-6.596342291,-6.678553054,-6.767101363,-6.854097715,-6.939827648,-7.147469511,-7.338466812,-7.693706252,-8.011574333,-8.304662523,-8.580001132,-8.853437999,-9.116483727,-9.672409812,-10.15052505,-10.63257971,-11.53229896,-12.40404626,-13.65413783],[
-0.81081115,-0.983725954,-1.820698156,-2.55959277,-3.091271582,-3.495090373,-3.829243066,-4.103045268,-4.534712478,-4.873426527,-5.088867332,-5.231435038,-5.340805063,-5.437873897,-5.527873956,-5.612198301,-5.692031398,-5.777849538,-5.861997678,-5.944975812,-6.146749397,-6.3329213,-6.679030316,-6.989700242,-7.276251984,-7.545987984,-7.813094803,-8.070351782,-8.606478265,-9.084821226,-9.539829477,-10.42268583,-11.27122606,-12.49056582],[
-0.374558777,-0.507283183,-1.169027095,-1.84214418,-2.341713816,-2.723809869,-3.039149682,-3.300130408,-3.7145268,-4.041316,-4.24792752,-4.385810404,-4.491390024,-4.584855406,-4.671655692,-4.753078357,-4.830178731,-4.912946856,-4.994202589,-5.074440977,-5.270288488,-5.451506653,-5.788274586,-6.090868073,-6.370593611,-6.634240478,-6.894326377,-7.144065824,-7.664983464,-8.121056709,-8.565855659,-9.421152908,-10.23660809,-11.41722022],[
-0.039770106,-0.146076406,-0.638809966,-1.218571129,-1.675302358,-2.029823588,-2.324930867,-2.571521746,-2.964228706,-3.275110617,-3.470855011,-3.602202931,-3.702467771,-3.791464371,-3.874275678,-3.953113081,-4.027827478,-4.107911081,-4.18655873,-4.264420863,-4.455049971,-4.631616347,-4.960082058,-5.256728902,-5.53081492,-5.789392795,-6.03964436,-6.280764456,-6.784441625,-7.221949724,-7.653646307,-8.476990376,-9.267570963,-10.39832838],[
0.207661641,0.120579986,-0.253319122,-0.728563449,-1.128414666,-1.449533522,-1.720435401,-1.950208096,-2.320730728,-2.615726933,-2.800999833,-2.924811665,-3.018946162,-3.102501167,-3.180382765,-3.255890463,-3.327473892,-3.40426971,-3.4798129,-3.554625183,-3.739252705,-3.911123992,-4.230577299,-4.518641572,-4.785824016,-5.038090779,-5.282733114,-5.518508923,-6.003504817,-6.429590367,-6.847015356,-7.642370181,-8.40010267,-9.484435362],[
0.395902125,0.32446213,0.030505377,-0.348992776,-0.687092152,-0.970380589,-1.215236422,-1.426333709,-1.771205232,-2.047303946,-2.221122283,-2.336165673,-2.42344917,-2.500568569,-2.573122327,-2.645104284,-2.713444649,-2.786850679,-2.859293779,-2.931108223,-3.109636845,-3.276669937,-3.587163877,-3.867020449,-4.126613315,-4.371866488,-4.609683103,-4.839717759,-5.305292463,-5.720669186,-6.127034843,-6.89496678,-7.620636659,-8.668575414],[
0.546701865,0.48540831,0.243096051,-0.060705244,-0.34201059,-0.585670712,-0.799882378,-0.987922816,-1.300225803,-1.550498273,-1.710533372,-1.815212524,-1.896204105,-1.969623619,-2.03843244,-2.107717769,-2.173620724,-2.244190977,-2.313781586,-2.382840183,-2.554034755,-2.713999686,-3.012058451,-3.280523083,-3.530371104,-3.767038055,-3.997240575,-4.219487595,-4.667928036,-5.069808036,-5.460625033,-6.203462014,-6.906587624,-7.910179422],[
0.63792701,0.588871044,0.388391131,0.145003422,-0.084843017,-0.289583399,-0.472457246,-0.636051052,-0.91343077,-1.133023273,-1.277152916,-1.37387057,-1.448971445,-1.516674986,-1.580387171,-1.646099049,-1.708673619,-1.776050105,-1.842881389,-1.909265519,-2.073148196,-2.225916535,-2.509836131,-2.76626762,-3.005543449,-3.232752803,-3.454053283,-3.668454489,-4.102681856,-4.492693007,-4.871813605,-5.590693254,-6.267538193,-7.239785263],[

0.666221963,0.621117676,0.438430668,0.220398546,0.023214467,-0.157575332,-0.3229416,-0.471722285,-0.729101609,-0.929055424,-1.067387894,-1.157660441,-1.228697516,-1.293198232,-1.354033719,-1.417683369,-1.478360554,-1.5436089,-1.608338813,-1.672636962,-1.831662349,-1.979975319,-2.256762142,-2.507209276,-2.741260492,-2.964761446,-3.18286436,-3.394539972,-3.818892074,-4.202450185,-4.57657389,-5.284119364,-5.937458507,-6.887051501],[
0.676457127,0.632224268,0.46021002,0.260544476,0.077381083,-0.086486589,-0.238107644,-0.377296288,-0.620079612,-0.808534597,-0.941389331,-1.027042954,-1.095322738,-1.157659553,-1.216569154,-1.278851658,-1.33825047,-1.402107804,-1.465424861,-1.528348238,-1.684089769,-1.829378911,-2.101532964,-2.347961747,-2.580148672,-2.801176791,-3.01655056,-3.225841874,-3.645236242,-4.023672654,-4.389978882,-5.084142666,-5.731625318,-6.659644202],[
0.678712933,0.63414495,0.472935926,0.290861055,0.126000473,-0.02656381,-0.165321148,-0.2904172,-0.517445163,-0.695747378,-0.823286045,-0.904162226,-0.969635982,-1.029852043,-1.086845169,-1.147766875,-1.205888413,-1.268383086,-1.330361678,-1.391921463,-1.544453847,-1.686815192,-1.954370078,-2.199152027,-2.428017019,-2.646879187,-2.859851487,-3.066997995,-3.478115748,-3.855621908,-4.218093661,-4.899960941,-5.54213551,-6.469544308],[
-1.089411062,-1.302445046,-2.236084847,-3.007515595,-3.575200737,-4.011127133,-4.368413217,-4.660755892,-5.122754237,-5.48550226,-5.724511953,-5.876858709,-5.996936528,-6.103409992,-6.202720163,-6.295890991,-6.384141769,-6.481292867,-6.576501288,-6.669947937,-6.895936932,-7.10481821,-7.494658851,-7.845590093,-8.170913622,-8.477829128,-8.78415743,-9.079184235,-9.707908045,-10.24044314,-10.77872186,-11.77210701,-12.72439208,-14.07763321],[
-0.542758725,-0.706740698,-1.496875751,-2.228973756,-2.770439022,-3.183851819,-3.524957072,-3.80527216,-4.248662954,-4.597927251,-4.826391211,-4.973720804,-5.089855116,-5.193212058,-5.289452339,-5.379964012,-5.465932911,-5.560249868,-5.652491458,-5.743053953,-5.962884044,-6.166606552,-6.546530704,-6.889487643,-7.207281066,-7.507489729,-7.805950716,-8.09364304,-8.697738296,-9.229974741,-9.734944218,-10.70539145,-11.62967297,-12.94412794],[
-0.121209608,-0.25068574,-0.882556842,-1.550462902,-2.058681176,-2.449282712,-2.77153738,-3.038581895,-3.463906092,-3.800284082,-4.019061678,-4.161603797,-4.273929576,-4.373663375,-4.46668341,-4.554277036,-4.637489808,-4.728593137,-4.817841272,-4.905561477,-5.119233005,-5.317721563,-5.687653035,-6.021785679,-6.331724652,-6.624678032,-6.914472264,-7.193013566,-7.777622102,-8.283842722,-8.776622013,-9.713565224,-10.59923668,-11.86825507],[
0.202579726,0.097033876,-0.381045832,-0.958128864,-1.422802103,-1.784787272,-2.086562277,-2.338584644,-2.741190108,-3.060731158,-3.267733987,-3.403410341,-3.510226139,-3.605602181,-3.694730839,-3.779892378,-3.860861597,-3.949247258,-4.035779811,-4.121043243,-4.329273555,-4.522758125,-4.883631553,-5.211450691,-5.51553318,-5.803150705,-6.081225588,-6.349392812,-6.91246696,-7.396436463,-7.873592279,-8.772737063,-9.627400943,-10.83728939],[
0.44186101,0.354890825,-0.015095795,-0.494455897,-0.902228473,-1.229886674,-1.507007642,-1.741558322,-2.120177064,-2.422190863,-2.617479619,-2.744831703,-2.844932634,-2.934379519,-3.018146401,-3.09968183,-3.177179372,-3.261928285,-3.345045506,-3.426938521,-3.628527843,-3.816790565,-4.167759744,-4.485750072,-4.781650036,-5.061702357,-5.333200373,-5.595158765,-6.135738706,-6.606875539,-7.065336183,-7.931463842,-8.746856389,-9.904434949],[
0.624199441,0.552766337,0.257257103,-0.13229458,-0.478585942,-0.767152618,-1.016372363,-1.230677918,-1.581563066,-1.862952301,-2.045287036,-2.162924577,-2.255401184,-2.337463158,-2.415295957,-2.493008095,-2.567078532,-2.64808459,-2.727831093,-2.80647023,-3.001385399,-3.184211758,-3.525095226,-3.833457192,-4.12059754,-4.392610556,-4.65638463,-4.911823978,-5.429960391,-5.887834438,-6.333643594,-7.166394756,-7.944760411,-9.058366488],[
0.770354896,0.70906253,0.463173419,0.150919575,-0.138109827,-0.386875925,-0.605316614,-0.79658709,-1.113976879,-1.369008622,-1.536715173,-1.64251334,-1.72774483,-1.805925697,-1.879618487,-1.954149623,-2.025358923,-2.103001807,-2.179456269,-2.254943167,-2.441691975,-2.616726416,-2.943742258,-3.239304314,-3.515619836,-3.778116434,-4.033630244,-4.280509872,-4.779162961,-5.222006155,-5.650017273,-6.453477784,-7.206136689,-8.270167616],[
0.85631285,0.808536873,0.604519089,0.355721767,0.119256656,-0.090483131,-0.276938991,-0.443425163,-0.725426194,-0.949214451,-1.099549794,-1.198024111,-1.277008791,-1.348950049,-1.417093693,-1.487700491,-1.555258679,-1.62917674,-1.702344404,-1.774696017,-1.953123573,-2.120083299,-2.432099416,-2.714901502,-2.979874808,-3.232209512,-3.478196999,-3.716876143,-4.199761497,-4.628999212,-5.043371694,-5.819451945,-6.540630657,-7.568358781],[
0.884816329,0.840976249,0.654412864,0.428524771,0.228048214,0.043754414,-0.124949407,-0.276325988,-0.537314667,-0.740642362,-0.885650422,-0.977485815,-1.052042769,-1.120432471,-1.185380718,-1.253647485,-1.319059992,-1.390510323,-1.46128288,-1.531273068,-1.704267254,-1.866347978,-2.170498375,-2.446755052,-

2.706009799,-2.954650469,-3.19754397,-3.433466573,-3.90517233,-4.326860166,-4.735265498,-5.497066802,-6.193651875,-7.19043747],[
0.895834933,0.853094978,0.677006415,0.468504727,0.280334163,0.114900209,-0.038633837,-0.180283442,-0.425541041,-0.618276915,-0.757427207,-0.84454208,-0.916101127,-0.982089866,-1.0448932,-1.111605387,-1.175566867,-1.245403815,-1.314554612,-1.382990659,-1.552303272,-1.711071352,-2.010109809,-2.281984721,-2.53975896,-2.785819764,-3.02582074,-3.259138658,-3.725133004,-4.140947844,-4.540480465,-5.288361611,-5.975545333,-6.946218247],[
0.897332988,0.854874382,0.68986495,0.496983389,0.326814505,0.171913549,0.032279586,-0.092883073,-0.321230035,-0.504907047,-0.638370338,-0.720579337,-0.789070532,-0.852675058,-0.913351135,-0.978492312,-1.041004835,-1.109252705,-1.176854897,-1.243761192,-1.40947566,-1.565066831,-1.859009387,-2.129806343,-2.383993726,-2.627801554,-2.865217783,-3.096208317,-3.552300229,-3.96663873,-4.361137729,-5.094347803,-5.768296367,-6.736668005],[
-0.755077602,-0.961116609,-1.822426954,-2.571273798,-3.142699452,-3.585623232,-3.947192141,-4.244605941,-4.716567899,-5.089551887,-5.344410528,-5.507265782,-5.640053343,-5.759448946,-5.871315621,-5.976616776,-6.07646441,-6.186297624,-6.293429331,-6.398169835,-6.652036979,-6.888062732,-7.328243546,-7.726551596,-8.097107336,-8.446992423,-8.798390922,-9.135930169,-9.857579459,-10.45084506,-11.05252009,-12.13783792,-13.16181845,-14.60403289],[
-0.244307691,-0.410147801,-1.152091679,-1.865624097,-2.41010567,-2.829143991,-3.174052885,-3.458650533,-3.91127409,-4.26979845,-4.513120463,-4.670871413,-4.799663321,-4.915855215,-5.024456117,-5.126926935,-5.224478453,-5.331308601,-5.435236672,-5.536751856,-5.783687041,-6.01382174,-6.442741821,-6.831448921,-7.192322085,-7.533333035,-7.874371723,-8.201971652,-8.889608721,-9.483367887,-10.0396892,-11.0926137,-12.08203845,-13.47359021],[
0.158187815,0.020743196,-0.584966785,-1.238714038,-1.749771143,-2.14547445,-2.472177006,-2.74340849,-3.177333606,-3.521850412,-3.754419668,-3.906974389,-4.031746281,-4.144110446,-4.249270467,-4.348598265,-4.443152267,-4.546416043,-4.647086037,-4.745489557,-4.985689581,-5.209959666,-5.627556795,-6.00567327,-6.35654967,-6.687908475,-7.017795089,-7.333156902,-7.995217868,-8.555344637,-9.09610101,-10.1071291,-11.05248982,-12.38321534],[
0.468035802,0.351348192,-0.122332768,-0.69183666,-1.159873342,-1.526134517,-1.832301254,-2.087638582,-2.497494258,-2.82382289,-3.043047469,-3.187765541,-3.306333156,-3.414602673,-3.516069002,-3.61339268,-3.70606601,-3.806755944,-3.904743421,-4.000800922,-4.235522763,-4.454075833,-4.860871517,-5.232722924,-5.577419828,-5.902810956,-6.215464961,-6.515590349,-7.147222886,-7.678235872,-8.197754582,-9.164748572,-10.0693159,-11.31967789],[
0.6925287,0.593939395,0.212332529,-0.270473379,-0.68290403,-1.01428354,-1.295635596,-1.532952119,-1.917860586,-2.225749796,-2.432325185,-2.568060688,-2.679367106,-2.780564453,-2.875626565,-2.968701202,-3.058086052,-3.154352277,-3.24807586,-3.339847092,-3.566337854,-3.77875979,-4.174243663,-4.53413865,-4.868448569,-5.183909969,-5.488397376,-5.780809229,-6.381528705,-6.895074444,-7.389178734,-8.315016875,-9.170335624,-10.35892039],[
0.864843765,0.782458179,0.467782586,0.065309548,-0.288297602,-0.581316646,-0.833838757,-1.050061191,-1.405038676,-1.690255506,-1.881763696,-2.006261541,-2.108601267,-2.201701098,-2.289902677,-2.378303074,-2.462724976,-2.55457589,-2.644476107,-2.732565404,-2.951530863,-3.157750048,-3.54141216,-3.889520398,-4.213068113,-4.518530244,-4.813301012,-5.097333176,-5.66873662,-6.165310691,-6.643367424,-7.524657056,-8.334020142,-9.460165831],[
1.004110263,0.933184826,0.667334016,0.340268888,0.042465245,-0.211164048,-0.433233093,-0.62681138,-0.947417399,-1.205976005,-1.381344969,-1.493103818,-1.586893423,-1.674236243,-1.75690189,-1.841419499,-1.92236187,-2.010202013,-2.096272407,-2.180732004,-2.390148863,-2.587129411,-2.954142674,-3.286629905,-3.597006416,-3.890747661,-4.17544054,-4.44874201,-4.995577632,-5.473664961,-5.929523657,-6.774761808,-7.543498566,-8.600827444],[
1.087417484,1.031497062,0.808419238,0.547020977,0.301771354,0.086677904,-0.103041777,-0.271785361,-0.55687247,-0.784862143,-0.941586749,-1.044940498,-1.131357441,-1.211537467,-1.287846,-1.367983153,-1.444860074,-1.528340657,-1.610487202,-1.691261434,-1.890773948,-2.078070671,-2.428306814,-2.746237477,-3.043393165,-3.325196601,-3.598463889,-3.861835813,-4.388991167,-4.849759478,-5.289688814,-6.096804475,-6.807472277,-7.832580152],[
1.116378018,1.065180906,0.860567956,0.623002927,0.415251898,0.226070183,0.054145435,-0.099417487,-0.36299748,-0.570736163,-0.721617565,-0.818163789,-0.899714535,-0.975824647,-1.048469221,-1.125972718,-1.200464026,-1.281167719,-1.360662729,-1.438863144,-1.632467698,-1.814578914,-2.15631848,-2.467096503,-2.757998962,-3.035548815,-3.305175343,-3.565439558,-4.080271638,-4.532659303,-4.965081436,-5.745622746,-6.432863655,-7.441178111],[

1.129033068,1.079790324,0.886202718,0.665864227,0.470142258,0.300601037,0.144268062,0.000446101,-0.246982125,-0.444837755,-0.589512374,-0.681259951,-0.759545103,-0.832896108,-0.90305891,-0.978820086,-1.051670634,-1.130535599,-1.208231916,-1.284748795,-1.474324005,-1.652895064,-1.989146163,-2.295072367,-2.584320162,-2.859043318,-3.125513965,-3.382815098,-3.891785851,-4.33706117,-4.76223412,-5.519506108,-6.200365009,-7.191359546],[
1.126637698,1.079593581,0.89939954,0.695912644,0.517900273,0.358688633,0.216712041,0.090278511,-0.139657045,-0.329452833,-0.468079303,-0.5548162,-0.6297124,-0.700305768,-0.7680387,-0.842000007,-0.913220688,-0.990257608,-1.066231822,-1.14107506,-1.326670808,-1.501836373,-1.83250817,-2.137268843,-2.422355692,-2.694793462,-2.958407004,-3.212899847,-3.711037673,-4.154257202,-4.572114925,-5.3077134,-5.983878503,-6.974440013],[
-0.552618032,-0.747592256,-1.563683135,-2.311060336,-2.895300449,-3.35088544,-3.721946447,-4.028283132,-4.515832764,-4.902770537,-5.172713968,-5.339047682,-5.477471581,-5.603206307,-5.721456412,-5.833057476,-5.939116225,-6.05794989,-6.173599295,-6.28639068,-6.560505781,-6.815987191,-7.291445399,-7.722408863,-8.123434874,-8.501764277,-8.883636511,-9.249227568,-10.02111557,-10.64965555,-11.28943327,-12.42940627,-13.47827041,-14.9480696],[
-0.05785856,-0.217073354,-0.927286688,-1.639900456,-2.195379186,-2.625573677,-2.979106245,-3.271672387,-3.73865692,-4.109889752,-4.367055187,-4.528091673,-4.662267444,-4.784629107,-4.899314662,-5.007824995,-5.111379218,-5.226718899,-5.338691252,-5.447754111,-5.713754336,-5.962161863,-6.424204507,-6.843295452,-7.232249533,-7.599316537,-7.968179034,-8.321248654,-9.054278688,-9.682580997,-10.26981835,-11.36999294,-12.37977639,-13.77555917],[
0.333084985,0.197222693,-0.38740401,-1.041306179,-1.562027523,-1.967765006,-2.303043166,-2.581709342,-3.028818065,-3.384643646,-3.629758913,-3.785276354,-3.915147957,-4.033394184,-4.1443812,-4.24950438,-4.349803472,-4.461075813,-4.569360374,-4.674894427,-4.933143441,-5.174664249,-5.623453418,-6.029926711,-6.406693112,-6.7618601,-7.117048966,-7.455461333,-8.157378772,-8.750418067,-9.318378909,-10.37155499,-11.3359342,-12.66835877],[
0.631934235,0.513641742,0.049777977,-0.520901239,-0.997783981,-1.373002992,-1.687264774,-1.94917018,-2.370830201,-2.707153196,-2.937539761,-3.084680652,-3.207944131,-3.322334077,-3.429816281,-3.533132162,-3.631706908,-3.740286275,-3.845712218,-3.948797477,-4.201090806,-4.435957566,-4.8718855,-5.271390994,-5.641220101,-5.989510953,-6.32308369,-6.642219119,-7.308154304,-7.867035402,-8.410768839,-9.418278013,-10.32563837,-11.57278374],[
0.846410801,0.744829534,0.360914942,-0.128620688,-0.54999431,-0.889487906,-1.178455675,-1.4217052,-1.817339263,-2.134206238,-2.351060792,-2.488878725,-2.604573239,-2.711152997,-2.811575684,-2.910129084,-3.00536761,-3.10878398,-3.209213625,-3.30724411,-3.549878106,-3.777723797,-4.200894852,-4.586508002,-4.944000935,-5.280409959,-5.604357468,-5.914521838,-6.54451347,-7.082798914,-7.596036519,-8.555943747,-9.400664956,-10.59154465],[
1.012016688,0.926113598,0.602571776,0.188099242,-0.175085506,-0.475631224,-0.734442245,-0.955524266,-1.319128906,-1.611435631,-1.811568016,-1.937284451,-2.043315182,-2.141269838,-2.234300079,-2.327702961,-2.417136651,-2.51562784,-2.611836364,-2.70580457,-2.940175329,-3.161131156,-3.57117541,-3.943293959,-4.288495308,-4.613477516,-4.926232188,-5.226393893,-5.823798582,-6.341476758,-6.836588177,-7.741698738,-8.518142609,-9.652952553],[
1.14479386,1.07069839,0.794527835,0.455550556,0.148864189,-0.111395991,-0.339120228,-0.537173217,-0.864972396,-1.129616843,-1.312310585,-1.424461033,-1.521255424,-1.612721606,-1.699647058,-1.788820481,-1.87446756,-1.968455495,-2.060417799,-2.150390619,-2.374294569,-2.585131193,-2.976866389,-3.331603235,-3.662214417,-3.974157594,-4.275577005,-4.563782066,-5.13426473,-5.631483624,-6.102454889,-6.951871524,-7.690790361,-8.782262156],[
1.222973994,1.165395035,0.932525747,0.662337152,0.409303567,0.188153845,-0.006153201,-0.178674777,-0.469908555,-0.703065889,-0.865521491,-0.969394707,-1.058558921,-1.142508243,-1.222787349,-1.307416323,-1.38885128,-1.478039745,-1.565625842,-1.651521172,-1.864519833,-2.064751462,-2.438729801,-2.777850713,-3.094178469,-3.393216673,-3.682177794,-3.9594517,-4.508174511,-4.986259983,-5.440070097,-6.230032267,-6.939659106,-8.013199888],[
1.252543015,1.199849102,0.985299222,0.737652737,0.525179349,0.331492146,0.155470404,-0.001405642,-0.269951672,-0.48185059,-0.638508937,-0.735567946,-0.819754372,-0.899435969,-0.97588353,-1.057790494,-1.136777708,-1.222991499,-1.307749839,-1.390926534,-1.597706638,-1.792544678,-2.157662006,-2.489235783,-2.798932289,-3.093561591,-3.378638958,-3.65247366,-4.187915025,-4.657083028,-5.102388902,-5.864432703,-6.562395499,-7.619752366],[
1.266700068,1.21634896,1.012750547,0.781541486,0.579882103,0.407790435,0.248797303,0.101941402,-0.149220508,-0.351698983,-0.50171696,-0.593994748,-0.674839038,-0.751613079,-0.825447806,-0.905545844,-0.982833282,-1.067065547,-1.149905055,-1.231293108,-1.433819915,-1.624988913,-1.984372787,-2.31084506,-

2.619080868,-2.910744605,-3.192405424,-3.462918329,-3.991865413,-4.453492784,-4.889355074,-5.634724481,-6.329262839,-7.37050092],[
1.263531147,1.216368644,1.026941877,0.811182021,0.627056694,0.464883556,0.321486737,0.1943684,-0.037699604,-0.233033144,-0.376548348,-0.463807975,-0.541146673,-0.614988371,-0.686260901,-0.764462733,-0.840061809,-0.922302732,-1.003276623,-1.082885874,-1.28118503,-1.468804825,-1.822323705,-2.147957214,-2.451756833,-2.740893748,-3.01938107,-3.286603706,-3.803550505,-4.262598508,-4.684022583,-5.41795803,-6.113188348,-7.152689059],[
-0.336622989,-0.537044436,-1.334237661,-2.045940926,-2.630557561,-3.090652358,-3.463302974,-3.771009485,-4.260916442,-4.650929611,-4.931427798,-5.125597875,-5.288052698,-5.435914368,-5.574367443,-5.704617115,-5.827903824,-5.960367518,-6.088697361,-6.213441371,-6.514323079,-6.793682629,-7.306038706,-7.76506892,-8.187223218,-8.58080907,-8.968104488,-9.334881724,-10.07195758,-10.68368985,-11.28815986,-12.34473028,-13.27480624,-14.60757968],[
0.113565922,-0.052585089,-0.737665017,-1.424971714,-1.980496897,-2.412659718,-2.766940864,-3.060004127,-3.52862339,-3.902248316,-4.169326453,-4.356588909,-4.513696524,-4.657018758,-4.790853605,-4.916967097,-5.036728071,-5.164939585,-5.28886011,-5.409118784,-5.700044601,-5.970382923,-6.465411744,-6.908676785,-7.31483773,-7.693196209,-8.063371694,-8.41372369,-9.107629377,-9.708327653,-10.26287339,-11.25746135,-12.13111002,-13.40781315],[
0.471876379,0.328189287,-0.237075431,-0.870532701,-1.391301556,-1.798293954,-2.134928471,-2.414306854,-2.862676993,-3.220151062,-3.474304636,-3.654767049,-3.806491281,-3.944706256,-4.073851069,-4.195609792,-4.311177347,-4.434532955,-4.554022247,-4.670017747,-4.951363202,-5.212930195,-5.691134107,-6.118282612,-6.508641439,-6.871527332,-7.224231719,-7.556681682,-8.216506889,-8.779856736,-9.309712845,-10.24207648,-11.08013467,-12.29462214],[
0.750789845,0.624366888,0.167237673,-0.388564161,-0.865757604,-1.241364772,-1.556924877,-1.819114035,-2.241502029,-2.578936326,-2.817095004,-2.987974556,-3.13200727,-3.264492307,-3.388357729,-3.506811784,-3.619218887,-3.738459352,-3.853689787,-3.965739547,-4.237924326,-4.490468494,-4.9520379,-5.366117474,-5.743921093,-6.094571922,-6.426610191,-6.740495497,-7.362753062,-7.892499944,-8.391273191,-9.268681649,-10.05399709,-11.18805251],[
0.948138407,0.839263252,0.454518948,-0.026504197,-0.447117811,-0.786255008,-1.076489404,-1.320116724,-1.714500189,-2.030630451,-2.25281521,-2.412383121,-2.546983009,-2.67050199,-2.786217127,-2.898993381,-3.005999951,-3.119399023,-3.229129253,-3.335777752,-3.597130331,-3.840825491,-4.28646784,-4.683601132,-5.045913046,-5.381576292,-5.700270824,-6.001114159,-6.58674932,-7.091322205,-7.558276943,-8.383530999,-9.115208032,-10.19842093],[
1.1025296,1.010339713,0.682583372,0.273113139,-0.08745326,-0.385553887,-0.643542892,-0.863622127,-1.225476016,-1.517127434,-1.721889968,-1.868813688,-1.993039631,-2.10591945,-2.213108943,-2.320515622,-2.422656553,-2.530620814,-2.635516289,-2.737528381,-2.989292109,-3.224924087,-3.654875618,-4.036226107,-4.383950926,-4.70584784,-5.011142444,-5.299734452,-5.851941967,-6.332616152,-6.771821784,-7.537618976,-8.225907879,-9.261230555],[
1.234578074,1.154828116,0.872837983,0.541117249,0.23669271,-0.022201979,-0.249659031,-0.447459122,-0.774448327,-1.039753257,-1.226897048,-1.358341601,-1.472571677,-1.579749487,-1.680755793,-1.783364618,-1.881188193,-1.984204661,-2.084483963,-2.182136984,-2.422238256,-2.646433668,-3.055638242,-3.417859771,-3.749423662,-4.056825122,-4.349351304,-4.624812104,-5.149699599,-5.607333284,-6.019349769,-6.73737508,-7.405798946,-8.402071027],[
1.311982221,1.249839428,1.011802442,0.749713968,0.498918819,0.278497859,0.084312339,-0.088356177,-0.379706579,-0.614332736,-0.780300854,-0.904551245,-1.010904062,-1.109815605,-1.203489806,-1.301168943,-1.394434346,-1.49235528,-1.587959821,-1.681251327,-1.909547446,-2.122207228,-2.512246297,-2.858011828,-3.174465335,-3.468094382,-3.747524993,-4.011793436,-4.517000641,-4.95563498,-5.343998006,-6.026546811,-6.671617148,-7.651732942],[
1.342193639,1.284943221,1.065529371,0.824024294,0.61675233,0.424892057,0.248846541,0.09171591,-0.177051745,-0.390872788,-0.552422229,-0.669344512,-0.770270329,-0.864515655,-0.954022183,-1.048826995,-1.139517305,-1.234404574,-1.327123388,-1.417632123,-1.639593473,-1.846785648,-2.227927024,-2.566298841,-2.876226379,-3.166088278,-3.442408918,-3.703335837,-4.197020785,-4.627124014,-5.000622002,-5.670152099,-6.306960612,-7.273400632],[
1.356293497,1.301624025,1.093217158,0.868213692,0.670582293,0.502047716,0.344347667,0.197262179,-0.05367355,-0.259921076,-0.414803905,-0.526569457,-0.623821082,-0.714884511,-0.801530075,-0.894413141,-0.983315459,-1.076164566,-1.166906134,-1.255591923,-1.473199002,-1.676693054,-2.052121408,-2.385507676,-2.694855458,-2.981927072,-3.254807801,-3.512443002,-4.000979119,-4.423371798,-4.794734121,-5.44892641,-6.082065918,-7.034818314],[

1.351275704,1.300425043,1.106587972,0.896411093,0.717104791,0.55810148,0.416968034,0.29183843,0.059637517,-0.140333954,-0.288660502,-0.395081907,-0.488526503,-0.576398502,-0.660242365,-0.751119408,-0.838240309,-0.929018588,-1.017851918,-1.104688923,-1.317911246,-1.517788868,-1.887291695,-2.220740323,-2.525628029,-2.810010035,-3.07951008,-3.33382825,-3.813236575,-4.230427745,-4.588490329,-5.239764926,-5.873161534,-6.823008307],[
-0.277831715,-0.478936514,-1.279763574,-2.000285978,-2.593922943,-3.061984419,-3.441253208,-3.754762848,-4.254355222,-4.652651087,-4.939762288,-5.137188447,-5.303661881,-5.455485557,-5.597653612,-5.731391287,-5.857997954,-5.994344295,-6.126245251,-6.254256423,-6.563138303,-6.849810068,-7.372999526,-7.840772049,-8.269807168,-8.668611158,-9.062061436,-9.433203284,-10.16401737,-10.77459721,-11.38049478,-12.42758069,-13.36207409,-14.72319032],[
0.155543647,-0.012502657,-0.698728539,-1.393832909,-1.957356097,-2.39688586,-2.757441148,-3.055878452,-3.533584181,-3.914829751,-4.187767499,-4.377749989,-4.538372975,-4.685124184,-4.822172104,-4.951299988,-5.073919157,-5.205483913,-5.332502765,-5.455563103,-5.75335927,-6.029922368,-6.533821714,-6.983897347,-7.395111112,-7.776978408,-8.151609461,-8.504823725,-9.191867005,-9.791231912,-10.34292616,-11.32838076,-12.21462627,-13.51714005],[
0.501597596,0.354093648,-0.212278767,-0.852824602,-1.380686665,-1.794533472,-2.137377069,-2.421896524,-2.878663026,-3.242971246,-3.502233889,-3.684987141,-3.839832606,-3.981150871,-4.113179908,-4.237633359,-4.355741773,-4.482019776,-4.604182968,-4.722569608,-5.009803865,-5.276636669,-5.762105388,-6.194541514,-6.588472577,-6.953464334,-7.309152442,-7.643261252,-8.2942307,-8.857252074,-9.38014214,-10.30572808,-11.16001888,-12.39828699],[
0.773899619,0.642432921,0.18136957,-0.380492287,-0.863784306,-1.245588394,-1.566910687,-1.83375217,-2.263753125,-2.60726072,-2.849685932,-3.022397988,-3.169165209,-3.304654997,-3.43129167,-3.552381045,-3.667255455,-3.78921298,-3.906927715,-4.021225327,-4.29881067,-4.555921195,-5.023336881,-5.442034503,-5.822706165,-6.174738122,-6.507161214,-6.820321448,-7.431689608,-7.959398008,-8.448457705,-9.324624996,-10.12434475,-11.28167033],[
0.9661582,0.851654112,0.4599051,-0.026465519,-0.452417787,-0.797283994,-1.093040148,-1.341169502,-1.743034285,-2.065103355,-2.291252293,-2.45257683,-2.589673147,-2.71564956,-2.833648924,-2.948650645,-3.057923656,-3.173510185,-3.285221486,-3.393612796,-3.659391437,-3.907060431,-4.357551874,-4.757982794,-5.121895778,-5.457747401,-5.775965632,-6.075450759,-6.649124122,-7.148889486,-7.604657639,-8.431283633,-9.176517843,-10.28298662],[
1.118481607,1.020557736,0.684462453,0.270643578,-0.094350306,-0.397363379,-0.660203765,-0.884416978,-1.252977625,-1.549877852,-1.758072691,-1.906566045,-2.032964561,-2.147968466,-2.257162672,-2.366627621,-2.47067834,-2.580537942,-2.68713211,-2.790629132,-3.046305696,-3.285500931,-3.719577154,-4.103155327,-4.451451434,-4.772519996,-5.07638689,-5.362599456,-5.902587014,-6.374085049,-6.804868054,-7.573813784,-8.276679329,-9.334358342],[
1.247886335,1.163115281,0.873624757,0.53928987,0.231898462,-0.030615075,-0.261775518,-0.462890913,-0.795458468,-1.065132802,-1.255035005,-1.387537416,-1.50370667,-1.612872722,-1.715743183,-1.820345029,-1.920023449,-2.024780668,-2.126617427,-2.225627859,-2.469142673,-2.6963429,-3.108656731,-3.472021238,-3.803221639,-4.108956326,-4.399252854,-4.671664657,-5.183969122,-5.631721294,-6.034104986,-6.762904939,-7.445508085,-8.463376697],[
1.321962806,1.256833681,1.012950985,0.751234223,0.498959306,0.276098027,0.079373849,-0.095710924,-0.391304136,-0.628825343,-0.796472583,-0.922107542,-1.030365344,-1.131139387,-1.226580292,-1.326283514,-1.421435152,-1.521003943,-1.618050427,-1.712593528,-1.943906797,-2.159142701,-2.55159073,-2.897704736,-3.21312264,-3.504485433,-3.781092259,-4.041876543,-4.534375222,-4.96272581,-5.34516684,-6.042423743,-6.700767618,-7.702758088],[
1.35226774,1.292186293,1.067026785,0.825234614,0.6194265,0.426721439,0.248744046,0.089705246,-0.182385255,-0.398020824,-0.561641693,-0.679964744,-0.782765279,-0.878857296,-0.970140366,-1.067065883,-1.159747174,-1.256312266,-1.350477619,-1.442252839,-1.667241466,-1.876999702,-2.260512782,-2.599083432,-2.907818672,-3.195431055,-3.468935505,-3.726168765,-4.206805535,-4.626903329,-4.997923312,-5.68201662,-6.331972191,-7.320166869],[
1.367054998,1.309702475,1.095520633,0.870105978,0.673289314,0.505848304,0.347685997,0.199064311,-0.054116878,-0.262349642,-0.419101898,-0.532282009,-0.631367883,-0.724254039,-0.812660041,-0.907727941,-0.998683525,-1.093221205,-1.185424375,-1.275372128,-1.49603214,-1.702110821,-2.079914883,-2.413446706,-2.721864871,-3.006650862,-3.276568372,-3.530360344,-4.006057924,-4.417886819,-4.790873405,-5.458388014,-6.104651812,-7.07981241],[

1.362412874,1.309229088,1.109815647,0.897944944,0.720061726,0.56214001,0.421876012,0.297432075,0.064294059,-

0.138001101,-0.2879679,-0.395821208,-0.491040427,-0.580702374,-0.666291018,-0.75939803,-0.848634003,-
0.941104522,-1.031385433,-1.119484686,-1.335759601,-1.538235045,-1.910073996,-2.244045501,-2.547883528,-
2.8297852,-3.096141414,-3.346412279,-3.81358762,-4.219511888,-4.58240128,-5.246983966,-5.893842116,-
6.866626752],[
-0.284387861,-0.479451148,-1.283930553,-2.018428788,-2.625844267,-3.106413937,-3.495936547,-3.818111423,-
4.332086196,-4.74206493,-5.038871794,-5.23758354,-5.405402697,-5.558978595,-5.702691352,-5.837812713,-
5.965741912,-6.105487416,-6.240498226,-6.371297733,-6.686199286,-6.977635386,-7.504943024,-7.974396962,-
8.403034356,-8.799859744,-9.191181835,-9.55883949,-10.25757393,-10.85406311,-11.45453329,-12.49889933,-
13.44784521,-14.8380346],[
0.129650736,-0.032906749,-0.71607403,-1.422783681,-1.998012757,-2.448700918,-2.818439914,-3.124648275,-
3.615357826,-4.007073758,-4.288602919,-4.47916086,-4.640469695,-4.788305272,-4.926285804,-5.056210882,-
5.179563405,-5.31372042,-5.443151239,-5.568365414,-5.870637998,-6.150519743,-6.656109465,-7.105718865,-
7.514824454,-7.893232593,-8.264534235,-8.613321969,-9.2710488,-9.85962172,-10.4042856,-11.39137216,-
12.29673722,-13.63241059],[
0.463623818,0.319410863,-0.241255129,-0.891104655,-1.428710815,-1.852409806,-2.203762501,-2.4953937,-
2.963700563,-3.337172022,-3.603937369,-3.786793398,-3.941848141,-4.083880897,-4.216492775,-4.341413549,-
4.459935454,-4.588190623,-4.712133238,-4.832079929,-5.1223622,-5.391215308,-5.876459647,-6.30690232,-
6.697451939,-7.057956843,-7.40941619,-7.738702808,-8.360913967,-8.915694518,-9.431112774,-10.36353823,-
11.24118072,-12.51143348],[
0.729048236,0.599249001,0.141908737,-0.426544631,-0.917744191,-1.308322067,-1.637218117,-1.910503183,-
2.350998776,-2.70272718,-2.951768448,-3.124370079,-3.271116816,-3.407165708,-3.534228294,-3.655548494,-
3.770602255,-3.894037337,-4.013114369,-4.128606824,-4.408278489,-4.666419949,-5.131916237,-5.547497922,-
5.923868477,-6.27061956,-6.597385975,-6.904384282,-7.487755231,-8.007251604,-8.490968154,-9.381683337,-
10.20279361,-11.39454872],[
0.917169741,0.802988005,0.412088944,-0.078964389,-0.511452348,-0.864343563,-1.167183388,-1.421469426,-
1.833624992,-2.163732531,-2.396144844,-2.55763302,-2.694712049,-2.820977955,-2.939172961,-3.054059275,-
3.163269384,-3.279691797,-3.392194063,-3.501241665,-3.767929696,-4.015791032,-4.463012998,-4.859057711,-
5.217532267,-5.547173732,-5.859040757,-6.151937235,-6.699215784,-7.190782426,-7.642645528,-8.486272929,-
9.255028925,-10.39780421],[
1.067140612,0.968548935,0.631748671,0.215725166,-0.154267231,-0.464197822,-0.73363286,-0.963786138,-
1.342026843,-1.646416155,-1.860553373,-2.009543894,-2.136020054,-2.251323524,-2.360714628,-2.470016217,-
2.573850854,-2.684137596,-2.791084143,-2.894842989,-3.150552088,-3.389192358,-3.818937075,-4.19699983,-
4.538859061,-4.852822052,-5.149578258,-5.428358003,-5.943924506,-6.407158333,-6.836015209,-7.626678941,-
8.354405176,-9.450284067],[
1.193489406,1.107994542,0.817406437,0.481285282,0.170195655,-0.097693732,-0.334043478,-0.539971459,-
0.881037992,-1.157134801,-1.352319734,-1.485321466,-1.601755035,-1.71134333,-1.81452352,-1.918972787,-
2.018434972,-2.123345195,-2.225285047,-2.32432639,-2.567266237,-2.793343535,-3.200420522,-3.557437638,-
3.881478064,-4.179490392,-4.462035856,-4.726680918,-5.21567736,-5.656430058,-6.060093597,-6.814663355,-
7.522742614,-8.579344411],[
1.263768139,1.198743141,0.95346131,0.690464158,0.435479428,0.208719066,0.008021703,-0.170791243,-
0.473144801,-0.715191436,-0.887145922,-1.013617556,-1.122390604,-1.223749034,-1.319647574,-1.419305426,-
1.514333288,-1.613875497,-1.710835348,-1.80521322,-2.035506849,-2.249257056,-2.635643201,-2.974598271,-
3.282284491,-3.565478792,-3.833947475,-4.086701402,-4.55726995,-4.980005754,-5.367883706,-6.093052832,-
6.7757033,-7.818486819],[
1.293396288,1.233172589,1.005624371,0.761175052,0.554255475,0.358955706,0.177736744,0.015554275,-
0.262256554,-0.48114347,-0.649106637,-0.768307207,-0.871753766,-0.968554636,-1.060432773,-1.157444059,-
1.250128017,-1.346637753,-1.440649778,-1.532208508,-1.756036502,-1.96411031,-2.341184171,-2.672171995,-
2.972790528,-3.252007945,-3.517172247,-3.765992547,-4.225526792,-4.640478872,-5.018860683,-5.731058445,-
6.405491614,-7.434400395],[
1.307905892,1.250376207,1.03319445,0.804501225,0.605851239,0.437191558,0.276987018,0.125663036,-
0.132194539,-0.343293817,-0.504115095,-0.618191695,-0.717990038,-0.811661649,-0.900741969,-0.995982981,-
1.087029079,-1.181485905,-1.273511186,-1.363191838,-1.582628357,-1.786942659,-2.158063951,-2.483786492,-
2.784022512,-3.060191523,-3.321525672,-3.566694237,-4.021548486,-4.429344088,-4.813847866,-5.506450303,-
6.177236849,-7.192968329],[
1.305194368,1.251710153,1.04809955,0.830508832,0.650738911,0.491622256,0.350412739,0.224892371,-
0.011320241,-0.216486822,-0.370207448,-0.479000018,-0.574997175,-0.665524033,-0.751855508,-0.845216466,-
0.934628897,-1.026990649,-1.117040275,-1.204825493,-1.419801805,-1.620399986,-1.985308231,-2.311384321,-

2.606811805,-2.879658655,-3.137148137,-3.378540676,-3.825773624,-4.228910184,-4.60521906,-5.294080419,-5.966574828,-6.979140386],[
-0.314546338,-0.507892016,-1.334756306,-2.077413696,-2.69046452,-3.175424848,-3.56880249,-3.893400971,-4.410332723,-4.822064597,-5.118649231,-5.32320124,-5.495531183,-5.652657066,-5.799350451,-5.937040779,-6.067237251,-6.207812016,-6.343615763,-6.475012332,-6.790877282,-7.082409704,-7.606757757,-8.071455558,-8.493861727,-8.883462386,-9.265963074,-9.624336213,-10.28951722,-10.87060362,-11.45298763,-12.47738362,-13.41434411,-14.79074111],[
0.082806473,-0.076351386,-0.769928076,-1.48420976,-2.064395721,-2.518801209,-2.892024538,-3.200461542,-3.693889722,-4.087233211,-4.368372757,-4.564067422,-4.7293354,-4.880148936,-5.020634029,-5.152658483,-5.277763281,-5.412355846,-5.542255109,-5.667786693,-5.970301208,-6.249549155,-6.750904422,-7.194591401,-7.596582154,-7.966968679,-8.328790479,-8.667780435,-9.2939907,-9.86411108,-10.39452623,-11.36449502,-12.25859871,-13.58057458],[
0.405411413,0.264646895,-0.297158766,-0.952288146,-1.49394773,-1.920888162,-2.275435934,-2.569231162,-3.040021051,-3.414990257,-3.681258507,-3.868806669,-4.027390613,-4.172079852,-4.306888631,-4.433622152,-4.553625264,-4.682086506,-4.80622594,-4.926229722,-5.216085797,-5.483679313,-5.963796687,-6.387645676,-6.770509827,-7.122547141,-7.464083674,-7.783482261,-8.375067861,-8.913430696,-9.414656248,-10.33171713,-11.19754178,-12.45386283],[
0.663370646,0.536773722,0.083530744,-0.486250751,-0.980407968,-1.37384422,-1.705641935,-1.981108033,-2.424275834,-2.777695298,-3.026367237,-3.203512089,-3.353617374,-3.491887644,-3.620741866,-3.743387748,-3.859470749,-3.982725275,-4.101666358,-4.216867798,-4.495343053,-4.751686972,-5.211305376,-5.619033692,-5.986690922,-6.32411525,-6.641730283,-6.939486027,-7.494258219,-7.998896694,-8.469833831,-9.3468117,-10.15627702,-11.3358433],[
0.846676899,0.735108471,0.350329218,-0.138580601,-0.572436362,-0.927740547,-1.23327174,-1.489859003,-1.904904329,-2.236923761,-2.469068248,-2.634907736,-2.775061889,-2.90342537,-3.023306962,-3.139261236,-3.24896467,-3.365069584,-3.477365914,-3.586098208,-3.851379592,-4.097113886,-4.537922964,-4.925726529,-5.275163761,-5.595278186,-5.897625263,-6.180953283,-6.701733268,-7.179224257,-7.620572162,-8.451993762,-9.21058428,-10.34120045],[
0.993677702,0.897056475,0.56703286,0.156544077,-0.212802735,-0.524059764,-0.795760453,-1.028149035,-1.409550422,-1.71624101,-1.930710733,-2.084253874,-2.21386698,-2.331065993,-2.442099358,-2.552483274,-2.657112414,-2.76703694,-2.873639245,-2.976961643,-3.230987307,-3.467270492,-3.890210152,-4.259759319,-4.592335948,-4.896574974,-5.183679137,-5.45269684,-5.943577597,-6.393835535,-6.813003134,-7.594092758,-8.313437446,-9.397570253],[
1.119592141,1.035686998,0.751273389,0.421496891,0.112103358,-0.156366078,-0.394192399,-0.601818985,-0.945909421,-1.224180449,-1.420024692,-1.557346652,-1.67700222,-1.788937469,-1.894017809,-1.999516112,-2.099726272,-2.204271734,-2.305854971,-2.404436152,-2.64561291,-2.86923875,-3.269331719,-3.617793244,-3.932519637,-4.220830845,-4.493631234,-4.748688098,-5.214405198,-5.64336114,-6.03995334,-6.785721612,-7.485729304,-8.530367278],[
1.189176417,1.125621418,0.885983759,0.629040795,0.376217635,0.149593166,-0.051983983,-0.232007835,-0.536758365,-0.780133843,-0.95274458,-1.083909205,-1.196055236,-1.299853217,-1.397718199,-1.49839868,-1.594150067,-1.693379822,-1.790015087,-1.883945967,-2.112524713,-2.323886757,-2.703051198,-3.033323142,-3.331678367,-3.605196963,-3.864004571,-4.107333364,-4.555961641,-4.968213007,-5.350690645,-6.06759915,-6.74214028,-7.772844195],[
1.218785444,1.159657843,0.936971263,0.697758242,0.49378755,0.299215967,0.117453634,-0.045641104,-0.32555188,-0.54522696,-0.714319889,-0.838134972,-0.944956044,-1.044274122,-1.138210538,-1.23628573,-1.329728404,-1.425991072,-1.519692445,-1.610816081,-1.832941819,-2.038577273,-2.408318519,-2.730521763,-3.021715774,-3.291330992,-3.546935707,-3.786231659,-4.224625581,-4.628916196,-5.001974497,-5.706235694,-6.372215253,-7.388820612],[
1.232839648,1.176170148,0.963394967,0.7399338,0.544055802,0.376779062,0.21664446,0.064649969,-0.194908375,-0.406711554,-0.568727067,-0.687358089,-0.790513292,-0.886761359,-0.977960074,-1.07428994,-1.166130623,-1.260381492,-1.352115816,-1.441361205,-1.659128445,-1.861000015,-2.224704079,-2.54160384,-2.832522525,-3.099040709,-3.350751119,-3.586349662,-4.020371477,-4.417657137,-4.796467511,-5.481565465,-6.143834417,-7.146505704],[
1.231811503,1.178678015,0.978328522,0.764669102,0.588107226,0.430478173,0.289882181,0.164530256,-0.072931351,-0.278638771,-0.433597922,-0.546901061,-0.646254357,-0.739422871,-0.827917688,-0.922403322,-1.012644738,-1.104844555,-1.194619857,-1.281968559,-1.495312883,-1.69344371,-2.05088892,-2.368220527,-2.654333429,-2.917318082,-3.165076377,-3.396883074,-3.823937446,-4.216809018,-4.587951622,-5.268938149,-5.932584056,-6.932389949],[

-0.405310348,-0.605462172,-1.466863045,-2.219651124,-2.829092503,-3.311206171,-3.703488911,-4.026737742,-4.541207077,-4.950741134,-5.242643821,-5.453033282,-5.631149328,-5.792659994,-5.942989598,-6.083758324,-6.216583238,-6.356536084,-6.491620321,-6.62201903,-6.935086103,-7.223302313,-7.738189146,-8.192689947,-8.604208712,-8.982577115,-9.353663127,-9.700511213,-10.33288364,-10.8981191,-11.4601573,-12.46427581,-13.39148775,-14.75727237],[
-0.018752243,-0.183147645,-0.90101087,-1.622573216,-2.200242382,-2.652665986,-3.025264725,-3.33276389,-3.824228605,-4.215728264,-4.492343411,-4.693269728,-4.863706121,-5.018266687,-5.161857304,-5.296448168,-5.423640791,-5.557325039,-5.686293184,-5.810661302,-6.109937448,-6.385433507,-6.876645694,-7.309499169,-7.700203516,-8.05902071,-8.409513374,-8.737099936,-9.333239417,-9.884297354,-10.3999534,-11.35144473,-12.238163,-13.54701344],[
0.294843937,0.149401882,-0.428316892,-1.088877353,-1.628404689,-2.053655988,-2.407692517,-2.700809976,-3.169832634,-3.543191128,-3.805115064,-3.997402696,-4.160662539,-4.308750242,-4.446335317,-4.575328415,-4.697138057,-4.824588472,-4.947622378,-5.066299377,-5.352496927,-5.615968156,-6.085622663,-6.498440531,-6.869903137,-7.210321594,-7.540827307,-7.849347234,-8.412343714,-8.933915315,-9.421399385,-10.3226274,-11.1780244,-12.42255694],[
0.548640658,0.417565438,-0.046286244,-0.618604395,-1.110503665,-1.502344429,-1.833497639,-2.108466405,-2.550224654,-2.902332786,-3.146900217,-3.328420686,-3.48283605,-3.624250972,-3.755639492,-3.880302388,-3.997963306,-4.120196887,-4.238068003,-4.352002197,-4.626825648,-4.878927408,-5.327896296,-5.724734824,-6.081071228,-6.406959887,-6.713321674,-6.999899176,-7.527519349,-8.01646804,-8.47514741,-9.337550092,-10.13798657,-11.30438111],[
0.72726355,0.610985564,0.219263562,-0.267639435,-0.698344984,-1.05222002,-1.357275608,-1.613767325,-2.028983817,-2.361114853,-2.590207454,-2.760494577,-2.904782491,-3.035900768,-3.157975518,-3.275560942,-3.386791747,-3.501687681,-3.612725477,-3.719996276,-3.981276754,-4.222671911,-4.652663702,-5.029547082,-5.367654907,-5.676287907,-5.967580368,-6.240101582,-6.735507095,-7.197481806,-7.627475436,-8.445652804,-9.195253098,-10.31101723],[
0.870853011,0.769323649,0.43463004,0.03091267,-0.334579399,-0.645072415,-0.917338452,-1.15069076,-1.53326872,-1.840848189,-2.053157673,-2.211061715,-2.344520546,-2.464675594,-2.577788054,-2.689462817,-2.794995196,-2.903609574,-3.008812212,-3.110553374,-3.360368556,-3.592199696,-4.004212291,-4.362774722,-4.683910241,-4.976559576,-5.252600138,-5.510751566,-5.977814941,-6.413401025,-6.821863157,-7.590834201,-8.300264373,-9.368500127],[
0.995824623,0.90750229,0.619674964,0.294395836,-0.011540029,-0.278885127,-0.516646285,-0.72470228,-1.069914959,-1.348923302,-1.542909869,-1.68501006,-1.80839204,-1.922555002,-2.029347129,-2.136018423,-2.237009478,-2.340325326,-2.4405592,-2.537593121,-2.774479259,-2.993485781,-3.382400248,-3.719720397,-4.022872493,-4.299540892,-4.561188242,-4.805531955,-5.248240873,-5.66402987,-6.050780646,-6.785002482,-7.473902619,-8.501653776],[
1.066461605,0.998152051,0.755157473,0.500415523,0.250263699,0.025030315,-0.176274338,-0.356416578,-0.661711305,-0.905508611,-1.076907995,-1.211695186,-1.327033456,-1.432931625,-1.532380418,-1.634126675,-1.730563172,-1.8286871,-1.924069092,-2.0165244,-2.240869935,-2.447609488,-2.815200195,-3.133972426,-3.420615969,-3.682430342,-3.930146041,-4.162734617,-4.588725021,-4.988391152,-5.362430241,-6.068066884,-6.731746674,-7.74460911],[
1.094460807,1.030655375,0.804676416,0.569767103,0.366495409,0.172472777,-0.008943359,-0.17209549,-0.45285321,-0.673118963,-0.840494827,-0.967802397,-1.077650533,-1.17902731,-1.2745304,-1.37366755,-1.467793817,-1.563051445,-1.655525241,-1.745206346,-1.963125389,-2.164066541,-2.522056271,-2.832536091,-3.111848645,-3.369377172,-3.613501222,-3.841747151,-4.258093661,-4.649488068,-5.01506732,-5.707984505,-6.362672824,-7.361350502],[
1.106933347,1.045571442,0.829559202,0.611228698,0.417369045,0.249598146,0.089093438,-0.062867378,-0.32373298,-0.535340853,-0.695783406,-0.817843453,-0.923901877,-1.022192132,-1.114947475,-1.212343392,-1.304881298,-1.39818893,-1.488740094,-1.576554061,-1.790174505,-1.987332675,-2.339171631,-2.64423253,-2.922740252,-3.177011184,-3.417174025,-3.641724619,-4.054048884,-4.438368991,-4.808701962,-5.483898526,-6.134888233,-7.118983082],[
1.105618378,1.047571824,0.843666027,0.636025175,0.461952378,0.30448354,0.163110557,0.036559001,-0.202681271,-0.407279116,-0.560791218,-0.677490519,-0.779632662,-0.874836141,-0.964863087,-1.060431822,-1.151387071,-1.24271708,-1.331344053,-1.417271791,-1.62654255,-1.819939571,-2.165441058,-2.470259983,-2.743949535,-2.994496474,-3.230661583,-3.451459739,-3.857455461,-4.237340784,-4.600856064,-5.271950262,-5.923627805,-6.905755479],[
-0.839660915,-1.042290903,-1.903441818,-2.644963763,-3.247883029,-3.726394985,-4.115936922,-4.435780483,-4.944341506,-5.347802258,-5.637152491,-5.855368185,-6.035465593,-6.19842498,-6.348928975,-6.489013585,-6.620555598,-6.756376694,-6.887542825,-7.014107538,-7.314061744,-7.587410275,-8.065504758,-8.482463963,-

8.856286928,-9.197637229,-9.527899821,-9.835934967,-10.37911718,-10.89086157,-11.39649477,-12.32137564,-13.18717842,-14.46578595],[
-0.434321098,-0.602670401,-1.320995719,-2.033868208,-2.604922543,-3.052485272,-3.420888409,-3.724431867,-4.208980217,-4.593973645,-4.867698551,-5.075165945,-5.24651361,-5.401377059,-5.544303257,-5.677384594,-5.802400289,-5.93129909,-6.055850971,-6.176077554,-6.461316029,-6.721153567,-7.17489277,-7.570055738,-7.923652136,-8.246254197,-8.557702714,-8.848330451,-9.364695905,-9.860308315,-10.3301894,-11.20957801,-12.04060072,-13.26529383],[
-0.105991869,-0.253831613,-0.835923818,-1.488584435,-2.021187374,-2.440467252,-2.789212708,-3.077942647,-3.539155045,-3.9058438,-4.16492919,-4.362970851,-4.526371423,-4.674275656,-4.810758186,-4.937859984,-5.057169546,-5.179601894,-5.297828303,-5.412017567,-5.68332399,-5.930506916,-6.362658657,-6.738345361,-7.073638091,-7.379027512,-7.672553111,-7.946529598,-8.435236918,-8.907874545,-9.352328834,-10.18811308,-10.98972642,-12.1589293],[
0.162441393,0.030614579,-0.439663348,-1.007364448,-1.492254861,-1.877723968,-2.202592528,-2.47326385,-2.907327859,-3.25307792,-3.495428171,-3.682445715,-3.836621479,-3.976794875,-4.106177337,-4.227978828,-4.342287343,-4.458920614,-4.571637168,-4.680723048,-4.939899252,-5.175514539,-5.587484973,-5.94665571,-6.266738636,-6.557837532,-6.83135523,-7.086974125,-7.547721089,-7.993619661,-8.413873688,-9.214770893,-9.968076423,-11.06833024],[
0.350613945,0.233718669,-0.157830812,-0.635466461,-1.058241762,-1.406097463,-1.70500142,-1.957498973,-2.366236743,-2.693293744,-2.921257969,-3.096749599,-3.240411389,-3.370334344,-3.49048696,-3.605004731,-3.712427111,-3.821797382,-3.927840388,-4.030496972,-4.27642697,-4.501484908,-4.89474872,-5.235320514,-5.538495333,-5.813815422,-6.073490889,-6.316283859,-6.752133506,-7.173969547,-7.570364211,-8.332893841,-9.039305343,-10.09258007],[
0.497525929,0.394931131,0.06473234,-0.323346993,-0.680679191,-0.986301947,-1.25471322,-1.485770033,-1.863993174,-2.168474252,-2.381524667,-2.544909061,-2.677893925,-2.797222927,-2.908592359,-3.017139422,-3.119066234,-3.222041875,-3.321955587,-3.41880296,-3.652681692,-3.867739344,-4.242911664,-4.565715104,-4.852478624,-5.112437419,-5.357796547,-5.587069137,-5.999921007,-6.397620136,-6.775165946,-7.49422564,-8.164421353,-9.171528428],[
0.627734697,0.538224257,0.255081472,-0.060745532,-0.362260555,-0.626573693,-0.860982545,-1.06644111,-1.408093453,-1.684519439,-1.879847311,-2.027908667,-2.151163007,-2.264317216,-2.369337849,-2.472695746,-2.569873171,-2.6674444,-2.762252823,-2.854214162,-3.074818404,-3.276816115,-3.628549105,-3.930345959,-4.199273097,-4.443494816,-4.674731858,-4.890915796,-5.28202471,-5.66238327,-6.019826772,-6.707252944,-7.358705717,-8.328260765],[
0.702443074,0.631011773,0.39065401,0.135602128,-0.113775306,-0.337289978,-0.536819277,-0.714923468,-1.016608276,-1.25845205,-1.432256241,-1.571332404,-1.686275345,-1.791208424,-1.888846699,-1.987022615,-2.079380067,-2.171641127,-2.261488476,-2.348669755,-2.556378932,-2.745866981,-3.074682451,-3.356981045,-3.608958614,-3.838131867,-4.055577872,-4.260146548,-4.635276002,-5.000615073,-5.346212243,-6.00897609,-6.638635906,-7.591096045],[
0.726005688,0.658710809,0.434804452,0.200557904,-0.00759559,-0.202715319,-0.383166643,-0.545066028,-0.823539313,-1.043552287,-1.212313412,-1.34337499,-1.452781309,-1.553206467,-1.646903054,-1.74232692,-1.832198196,-1.921484329,-2.00823501,-2.092431316,-2.293123662,-2.476004277,-2.793550337,-3.06623505,-3.309829075,-3.533495992,-3.746321863,-3.945764704,-4.311156295,-4.668263768,-5.006210009,-5.657279735,-6.278883101,-7.218220961],[
0.732225831,0.667145265,0.453474187,0.237117245,0.039752562,-0.132414127,-0.294236538,-0.445393272,-0.705578336,-0.916304629,-1.078345405,-1.203783884,-1.309401231,-1.406787865,-1.497787401,-1.591380547,-1.679586489,-1.766853505,-1.851590047,-1.933773872,-2.129818882,-2.308421984,-2.618615888,-2.885000205,-3.12623471,-3.345822639,-3.554280386,-3.74976925,-4.110982354,-4.461150314,-4.801655064,-5.439118952,-6.056948198,-6.984341371],[
0.732033735,0.668882315,0.464935989,0.259554931,0.081129463,-0.080387233,-0.225001349,-0.354527639,-0.594975815,-0.797581091,-0.952861228,-1.072630629,-1.174378,-1.268772581,-1.357077695,-1.448783507,-1.535334416,-1.62060378,-1.70332159,-1.783482039,-1.974876678,-2.149209164,-2.451995631,-2.716049006,-2.951877602,-3.166932314,-3.371049931,-3.562638181,-3.917916476,-4.263589532,-4.59872748,-5.232924933,-5.850151576,-6.775461356],[
-1.018147546,-1.227369088,-2.130805478,-2.893959126,-3.493462258,-3.967817429,-4.3552719,-4.672781543,-5.177126941,-5.573808146,-5.852253545,-6.058992526,-6.228330669,-6.378428733,-6.516286002,-6.644011184,-6.763563426,-6.887569664,-7.007366078,-7.122915237,-7.388517636,-7.625750617,-8.100396061,-8.514633737,-8.886270561,-9.225882734,-9.555981751,-9.864032739,-10.4016151,-10.91382643,-11.42478165,-12.35839441,-13.23444169,-14.52815802],[

-0.61094234,-0.780922172,-1.52811317,-2.255493591,-2.823968055,-3.26913504,-3.636050162,-3.937967413,-
4.419104899,-4.798452304,-5.06250723,-5.259751631,-5.421285637,-5.564464978,-5.695962276,-5.817839124,-
5.931915331,-6.050020596,-6.164244084,-6.274521945,-6.528755252,-6.756038908,-7.206495828,-7.599082018,-
7.950650452,-8.271675215,-8.583219228,-8.874096335,-9.386097543,-9.884347039,-10.35822343,-11.24662594,-
12.08821637,-13.3290697],[
-0.281172485,-0.428392933,-1.027420519,-1.6904921,-2.220360644,-2.637690582,-2.984825283,-3.27213385,-
3.730248454,-4.092146279,-4.342612026,-4.531428366,-4.685815672,-4.823163083,-4.949291913,-5.066223898,-
5.175585396,-5.288129669,-5.396843757,-5.501889176,-5.74484611,-5.962380695,-6.391346791,-6.764582249,-
7.098005783,-7.401987567,-7.695860039,-7.970328561,-8.455624884,-8.93168109,-9.380254022,-10.22547885,-
11.03914042,-12.22353369],[
-0.007374024,-0.137610653,-0.61555555,-1.18932845,-1.671200469,-2.055317408,-2.378516826,-2.648220073,-
3.079985725,-3.421910754,-3.656967512,-3.835964277,-3.982058522,-4.112767926,-4.232885235,-4.345508772,-
4.450858718,-4.558587939,-4.66280404,-4.763761661,-4.9975604,-5.206454893,-5.615412951,-5.972906488,-
6.29176799,-6.582014737,-6.854954071,-7.110190278,-7.567901088,-8.017063031,-8.441681155,-9.253859798,-
10.01805385,-11.13322706],[
0.185378482,0.069879901,-0.323485031,-0.800696155,-1.220503535,-1.567654399,-1.865223137,-2.117171963,-
2.525107748,-2.84988648,-3.072195125,-3.240743665,-3.377181573,-3.498524292,-3.610255046,-3.716280191,-
3.815658472,-3.916750877,-4.014878702,-4.109937733,-4.332335616,-4.532838717,-4.923170578,-5.262093183,-
5.564085,-5.83861135,-6.097952708,-6.340736558,-6.774016814,-7.199151185,-7.599159627,-8.373068157,-
9.090489235,-10.15911789],[
0.333654825,0.231833859,-0.097291411,-0.479667393,-0.833934409,-1.139333194,-1.40754281,-1.638960575,-
2.017331476,-2.320287401,-2.528694009,-2.685490903,-2.811497461,-2.922836472,-3.026137166,-3.126307129,-
3.220064456,-3.31487739,-3.406990306,-3.496376,-3.707419469,-3.89887812,-4.271095588,-4.592153235,-4.877666137,-
5.136800958,-5.381804337,-5.611048035,-6.022181405,-6.42317665,-6.804592869,-7.534752776,-8.215540579,-
9.238487088],[
0.4611546,0.372321749,0.091182183,-0.220969758,-0.519467962,-0.782900725,-1.01625059,-1.221055414,-
1.562118158,-1.836074443,-2.026653617,-2.168338257,-2.284689376,-2.389447755,-2.486214998,-2.581076455,-
2.669983451,-2.75927778,-2.846181253,-2.930594203,-3.128310275,-3.306916217,-3.655646562,-3.955577242,-
4.223161001,-4.466503836,-4.697329287,-4.913468835,-5.303472363,-5.687099146,-6.048763031,-6.74698845,-
7.40898039,-8.394619653],[
0.532661562,0.461540904,0.223144381,-0.031482979,-0.278750412,-0.5008859,-0.69913488,-0.875955945,-
1.175593665,-1.413441355,-1.582400873,-1.714095451,-1.821709755,-1.917928387,-2.006986231,-2.096249624,-
2.179961796,-2.263591837,-2.345221657,-2.424558056,-2.608705452,-2.774408669,-3.100017806,-3.380224556,-
3.630720077,-3.85893958,-4.075922186,-4.280360983,-4.654599345,-5.023153192,-5.373246977,-6.046947417,-
6.686787619,-7.656200904],[
0.551702332,0.484358863,0.261749424,0.028141677,-0.180071977,-0.374513108,-0.553545865,-0.714068897,-
0.990268018,-1.205673488,-1.368732588,-1.491824563,-1.593432274,-1.68458793,-1.769173661,-1.855101751,-
1.935776228,-2.015894618,-2.093918769,-2.169800978,-2.345645982,-2.503662203,-2.817775989,-3.088149679,-
3.330094469,-3.552565605,-3.764703883,-3.963911132,-4.328673262,-4.689198913,-5.031720443,-5.693852434,-
6.325778416,-7.282101622],[
0.552994501,0.487676833,0.27541523,0.060284057,-0.136703381,-0.309348711,-0.470388814,-0.620106975,-
0.878154408,-1.08302158,-1.239160654,-1.35623917,-1.453730287,-1.541463265,-1.62298888,-1.706662649,-
1.785285972,-1.863003131,-1.938659242,-2.012189806,-2.182487647,-2.335484136,-2.642064589,-2.905991443,-
3.145284769,-3.363568914,-3.571311868,-3.76655724,-4.12718746,-4.481019783,-4.827267593,-5.474806208,-
6.103014493,-7.047611064],[
0.553654696,0.489171896,0.284785154,0.07953678,-0.099040797,-0.260875601,-0.405287473,-0.534792144,-
0.772805386,-0.968576043,-1.117756992,-1.228798263,-1.322116126,-1.406491982,-1.484949269,-1.56632478,-
1.642909416,-1.718266362,-1.791551824,-1.862736468,-2.02754433,-2.175589042,-2.474584473,-2.735826109,-
2.969630658,-3.18331605,-3.386709179,-3.578042879,-3.932977355,-4.282423996,-4.623320708,-5.267708127,-
5.895942627,-6.837992602],[
1.671335571,1.446330626,0.308517281,-0.497052049,-1.102733187,-1.571142871,-1.959028166,-2.267357596,-
2.740256749,-3.101393974,-3.342535465,-3.523205246,-3.665540038,-3.786608821,-3.89749399,-4.000022111,-
4.095927322,-4.196222128,-4.294007531,-4.388363909,-4.612609249,-4.8160213,-5.189409719,-5.51201364,-
5.799177381,-6.060491704,-6.310554355,-6.542953793,-6.966856797,-7.349673069,-7.701416957,-8.337393376,-
8.906983402,-9.727410557],[
2.313901537,2.151399673,1.208325515,0.425976238,-0.151719023,-0.588315893,-0.954299012,-1.246462693,-
1.693527679,-2.035134048,-2.261171145,-2.430661659,-2.564391895,-2.677527683,-2.781070354,-2.876759854,-
2.966050389,-3.058676503,-3.148873862,-3.236036656,-3.443911532,-3.63241507,-3.979905917,-4.279915682,-

```
4.546676755,-4.789073026,-5.021300263,-5.237037474,-5.630004537,-5.985172147,-6.311951861,-6.904082099,-
7.436063681,-8.199011925],[
      2.864862945,2.746430942,2.009867662,1.287777682,0.746137387,0.337424164,-0.00655288,-0.282684316,-
0.707348656,-1.032330021,-1.244891299,-1.405302169,-1.531146245,-1.637611598,-1.734909662,-1.824576163,-
1.908427734,-1.994162481,-2.07781151,-2.158734403,-2.352121882,-2.527628095,-2.849576704,-3.127373768,-
3.373803071,-3.597451852,-3.811746606,-4.010762522,-4.376067296,-4.706245718,-5.009977407,-5.55869765,-
6.05252921,-6.760950969],[

3.269327082,3.183357163,2.666285863,2.043607086,1.549897123,1.173223309,0.853469353,0.594169937,0.193430425,-
0.114159537,-0.313408808,-0.46499779,-0.583298981,-0.683076198,-0.774215415,-0.858235623,-0.936422676,-
1.015664499,-1.09303788,-1.168169884,-1.348027174,-1.510903242,-1.812316343,-2.073281682,-2.303836287,-
2.512769433,-2.708943834,-2.89110093,-3.226979228,-3.53056048,-3.811058861,-4.317499687,-4.773725593,-
5.426094399],[

3.59333877,3.52847642,3.178025117,2.691574668,2.269512927,1.932977003,1.639253074,1.396395578,1.021621541,0.7
32505575,0.543614355,0.400279241,0.289529625,0.196353078,0.111306327,0.032887849,-0.040614085,-0.113414473,-
0.184999909,-0.254446354,-0.42220805,-0.574883509,-0.851896926,-1.091505535,-1.304425648,-1.497987067,-
1.680193442,-1.849585192,-2.159787384,-2.440600227,-2.698930624,-3.168341701,-3.591958117,-4.198175554],[

3.888007793,3.835830929,3.592292491,3.24028693,2.898233016,2.607323968,2.347541602,2.126376432,1.779185192,1.
508457606,1.330994316,1.196811349,1.094162181,1.007721529,0.928971167,0.856416439,0.788881639,0.722717962,0.6
57440299,0.593999872,0.439413139,0.298263399,0.041771958,-0.179781912,-0.37677474,-0.55587968,-0.724664192,-
0.881633752,-1.169637422,-1.43048471,-1.670911007,-2.107737303,-2.502436502,-3.069100668],[

4.217141562,4.169845441,3.982108477,3.716378569,3.443143775,3.201446829,2.985702519,2.796114963,2.486158098,2
.242732695,2.082100416,1.95728211,1.860945912,1.779441306,1.70500663,1.636247343,1.572356228,1.511005944,1.45
0772065,1.39206796,1.249000714,1.118508971,0.881561711,0.677019801,0.495891068,0.331424844,0.176448726,0.0320
03535,-0.234394283,-0.476227461,-0.699487554,-1.106651157,-1.475110038,-2.003963656],[

4.512456974,4.472284099,4.321881589,4.115185704,3.905178931,3.714489696,3.540778652,3.383732464,3.121516392,2
.916198321,2.773820372,2.659107997,2.569795419,2.49509595,2.426585779,2.363369641,2.304482143,2.248954291,2.1
94241596,2.140833731,2.010556537,1.891727765,1.673221223,1.484386379,1.316668768,1.16392522,1.019965679,0.885
632475,0.639800978,0.416114409,0.209351856,-0.16869689,-0.511547756,-1.003929383],[

4.673324218,4.635135665,4.499033995,4.324725228,4.150160334,3.987646251,3.83620069,3.698975162,3.460942471,3.
27803889,3.144353842,3.036000079,2.950403799,2.877332627,2.810359985,2.748315507,2.690817932,2.637174773,2.58
4671871,2.533483078,2.409387165,2.296828101,2.094231924,1.918703147,1.761887233,1.618351274,1.482803234,1.356
20564,1.123571142,0.911311877,0.714675963,0.355044967,0.028067636,-0.443040463],[

4.776206848,4.736186805,4.607417518,4.455244565,4.300509225,4.155547509,4.019437745,3.895091088,3.673860734,3
.505974526,3.37875962,3.274941716,3.191854237,3.120006817,3.054036991,2.993036797,2.93616352,2.883893398,2.83
2931525,2.783230353,2.663255015,2.554896734,2.363023884,2.196394011,2.046358989,1.90895244,1.779160845,1.6578
96189,1.433977497,1.22926166,1.039421855,0.691629465,0.374978524,-0.082359227],[

4.872608405,4.827713188,4.705442451,4.576585218,4.444146279,4.313722024,4.192622039,4.083045091,3.880024534,3
.72608906,3.605302107,3.506146912,3.425662456,3.354988212,3.289944081,3.230075798,3.173836199,3.1229214,3.073
456993,3.025457933,2.909546258,2.805399339,2.624249847,2.465827485,2.323031545,2.191872531,2.067849872,1.9519
65254,1.736850125,1.539832436,1.356739297,1.020882319,0.714731308,0.271309087]]
   return MedianLnYrefDataTable

def getLnY (thisFreq, inMagnitudein, inRrupin):
   inMagnitude = inMagnitudein;
   inRrup = inRrupin;

         # this algorithm assumes thisFreq is one of the frequencies listed,
         # if not, you need to write a wrapper that interpolates
   LnY = -999;
   # define these internally
```

```
MagnitudeInterpolationType = "Linear";
RrupInterpolationType = "Logarithmic";

medianLnYrefFrequencyLabelColumn = getmedianLnYrefFrequencyLabelColumn();
medianLnYrefFrequencyValueColumn = getmedianLnYrefFrequencyValueColumn();
medianLnYrefMomentColumn = getmedianLnYrefMomentColumn();
MedianLnYrefRrupRow = getMedianLnYrefRrupRow();
MedianLnYrefDataTable = getMedianLnYrefDataTable();

thisFreq = getTFvalue(thisFreq)
medianLnYrefFrequencyColumn = getmedianLnYrefFrequencyValueColumn();

Mbelow = -888;
Mabove = 888;
Nrow = len(medianLnYrefMomentColumn);
NcolR = len(MedianLnYrefRrupRow);

for iRow in range(Nrow):
   Fcheck = -1;
   hereFreq = medianLnYrefFrequencyColumn[iRow];

   if abs(hereFreq - thisFreq) < 0.000001:
      hereM = medianLnYrefMomentColumn[iRow];
      if abs(hereM - inMagnitude) < 0.001:

         for iColR in range(NcolR):
            hereRrup = MedianLnYrefRrupRow[iColR];
            if abs(hereRrup - inRrup) < 0.000001:
               LnY = MedianLnYrefDataTable[iRow][iColR] ; # ' nailed it!
               return LnY
            else:
               if iColR == NcolR:
                  LnY = -888;
                  return LnY
               nextRrup = MedianLnYrefRrupRow[iColR + 1];
               if (inRrup - hereRrup) * (inRrup - nextRrup) < 0:
                  if RrupInterpolationType.lower() == "Logarithmic".lower():
                     if inRrup == 0:
                        inRrup = 0.0000001;
                     if hereRrup == 0:
                        hereRrup = 0.0000001;
                     if nextRrup == 0:
                        nextRrup = 0.0000001;
                     hereRrup = math.log(hereRrup);
                     nextRrup = math.log(nextRrup);
                     inRrup = math.log(inRrup);

                  Y0 = MedianLnYrefDataTable[iRow][iColR];
                  Y1 = MedianLnYrefDataTable[iRow][iColR+1];
                  LnY = Y0 + (Y1 - Y0) / (nextRrup - hereRrup) * (inRrup - hereRrup);
                  return LnY
      else:
         if hereM > Mbelow and hereM < inMagnitude:
            Mbelow = hereM;
            Ibelow = iRow;
         if hereM < Mabove and hereM > inMagnitude:
            Mabove = hereM;
            Iabove = iRow;
```

```
  for iColR in range(NcolR):
    hereRrup = MedianLnYrefRrupRow[iColR];
    if abs(hereRrup - inRrup) < 0.000001:
      Z0 = MedianLnYrefDataTable[Ibelow][iColR] ; # nailed it!
      Z1 = MedianLnYrefDataTable[Iabove][iColR] ; # nailed it!
      break
    else:
      if iColR < NcolR-1:
        nextRrup = MedianLnYrefRrupRow[iColR+1];
        if (inRrup - hereRrup) * (inRrup - nextRrup) < 0:
          if RrupInterpolationType.lower() == "logarithmic".lower():
            if inRrup == 0:
              inRrup = 0.1;
            if hereRrup == 0:
              hereRrup = 0.1;
            if nextRrup == 0:
              nextRrup = 0.1;
            hereRrup = math.log(hereRrup);
            nextRrup = math.log(nextRrup);
            inRrup = math.log(inRrup);

          iRow = Ibelow;
          Y0 = MedianLnYrefDataTable[iRow][iColR];
          Y1 = MedianLnYrefDataTable[iRow][iColR+1];
          Z0 = Y0 + (inRrup - hereRrup) * (Y1 - Y0) / (nextRrup - hereRrup);
          iRow = Iabove;
          Y0 = MedianLnYrefDataTable[iRow][iColR];
          Y1 = MedianLnYrefDataTable[iRow][iColR+1];
          Z1 = Y0 + (inRrup - hereRrup) * (Y1 - Y0) / (nextRrup - hereRrup);
          break


  # interpolate on Moment
  if MagnitudeInterpolationType.lower() == "logarithmic".lower():
    Mbelow = math.log(Mbelow);
    Mabove = math.log(Mabove);
    inMagnitude = math.log(inMagnitude);

  LnY = Z0 + (inMagnitude - Mbelow) * (Z1 - Z0) / (Mabove - Mbelow);
  return LnY


def NGAEastGMM (thisTin,inMagnitude,inRrup,thisInputLabel):

  iNGAEastGMM = -999;
  iLnY = -999;

  if thisInputLabel.lower() == "Ergodic".lower():
    Frange = getFrequencyListCompErg();
  elif thisInputLabel.lower() == ("SingleStation".lower):
    Frange = getFrequencyListCompSS();
  else:
    Frange = getmedianLnYrefFrequencyValueColumn();

  Flist = unique(Frange);
  NFreq = len(Flist);
```

```
thisT = getTFvalue(thisTin);
# print(Flist)
# print(thisT)

for iiT in range(NFreq):
   # print("")
   hereF = Flist[iiT];
   if hereF < 0 :
      hereT = hereF;
   else:
      hereT = 1 / hereF;
   # print("hereT")
   # print(hereT)
   if abs(hereT - thisT) < 0.000001:
      # print("same")
      if thisInputLabel.lower() == "Ergodic".lower():
         iLnY = getSigmaCompErg(hereF, inMagnitude, inRrup);
      elif thisInputLabel.lower() == "SingleStation".lower():
         iLnY = getSigmaCompSS(hereF, inMagnitude, inRrup);
      else:
         iLnY = getLnY(hereF, inMagnitude, inRrup);
      break
   else:
      # print("else")
      if iiT < NFreq-1:
         nextF = Flist[iiT + 1];
         nextT = 1 / nextF;
         if hereT > 0 and nextT > 0 and (hereT - thisT) * (nextT - thisT) < 0:
            # print("inn")
            # print(nextT)

            if thisInputLabel.lower() == "Ergodic".lower():
               hereLnY = getSigmaCompErg(hereF, inMagnitude, inRrup);
               nextLnY = getSigmaCompErg(nextF, inMagnitude, inRrup);
            elif thisInputLabel.lower() == "SingleStation".lower():
               hereLnY = getSigmaCompSS(hereF, inMagnitude, inRrup);
               nextLnY = getSigmaCompSS(nextF, inMagnitude, inRrup);
            else:
               hereLnY = getLnY(hereF, inMagnitude, inRrup);
               nextLnY = getLnY(nextF, inMagnitude, inRrup);

            thisLnT = math.log(thisT);
            hereLnT = math.log(hereT);
            nextLnT = math.log(nextT);
            iLnY = hereLnY + (thisLnT - hereLnT) * (nextLnY - hereLnY) / (nextLnT - hereLnT);
            break




if thisInputLabel.lower() == "MedianPSA".lower() and iLnY >-100:
   iNGAEastGMM = math.exp(iLnY);
else:
   iNGAEastGMM = iLnY;
return iNGAEastGMM
```

```
def getSigma(thisFreq, inMagnitude, inRrup, sigmaModel):

   if sigmaModel.lower() == "Ergodic".lower():
      Sigma = getSigmaCompErg(thisFreq, inMagnitude, inRrup);
   else:
      Sigma = getSigmaCompSS(thisFreq, inMagnitude, inRrup);
   return Sigma

def getFrequencyListCompSS():
   FrequencyListCompSS =
(100,50,40,33.333,25,20,13.333,10,6.667,5,4,3.333,2.5,2,1.333,1,0.667,0.5,0.333,0.25,0.2,0.133,0.1,-1,-2);
   return FrequencyListCompSS


def getSigmaCompSS(thisFreq, inMagnitude, inRrup):
   WeightSigmaCompSSC=0.63;
   WeightSigmaCompSSH=0.185;
   WeightSigmaCompSSL=0.185;
   WeightSigmaCompSSSum=1;

   FrequencyListCompSS = getFrequencyListCompSS();
   thisRowIndex = -999;
   NrowFSS= len(FrequencyListCompSS);
   for iRowF in range(NrowFSS):
      hereFreq = FrequencyListCompSS[iRowF]
      if abs(hereFreq - thisFreq) < 0.000001:
         thisRowIndex = iRowF;




   # Central Model
   SigCompSS1C =
(0.7006,0.6998,0.6994,0.699,0.6981,0.6974,0.6955,0.6935,0.6898,0.6862,0.6827,0.6794,0.673,0.6671,0.6541,0.6432,0.6262
,0.6143,0.5998,0.5925,0.5889,0.5858,0.5853,0.7006,0.6303);
   SigCompSS2C =
(0.6846,0.6838,0.6833,0.6829,0.682,0.6813,0.6793,0.6773,0.6735,0.6698,0.6662,0.6628,0.6563,0.6502,0.6369,0.6256,0.608
2,0.5959,0.5811,0.5736,0.5698,0.5666,0.5661,0.6846,0.6247);
   SigCompSS3C =
(0.6127,0.6122,0.6119,0.6116,0.611,0.6105,0.6092,0.6079,0.6063,0.6066,0.6068,0.6072,0.6075,0.606,0.5999,0.5921,0.5781
,0.5674,0.5517,0.5438,0.5397,0.5364,0.5357,0.6127,0.5765);
   SigCompSS4C =
(0.4973,0.4974,0.4974,0.4974,0.4975,0.4975,0.4977,0.4978,0.5007,0.5092,0.5171,0.5254,0.5399,0.5482,0.5577,0.5576,0.55
1,0.5436,0.5267,0.5181,0.5138,0.5102,0.5095,0.4973,0.4896);
   S1 = SigCompSS1C[thisRowIndex];
   S2 = SigCompSS2C[thisRowIndex];
   S3 = SigCompSS3C[thisRowIndex];
   S4 = SigCompSS4C[thisRowIndex];
   SigmaCompSSC = getSigmaCompSSi(inMagnitude, S1, S2, S3, S4);
   # High Model
   SigCompSS1H =
(0.8193,0.8183,0.8178,0.8173,0.8162,0.8153,0.8129,0.8106,0.8061,0.8019,0.798,0.7943,0.7876,0.7816,0.7697,0.7611,0.750
2,0.7444,0.7391,0.7372,0.7364,0.7358,0.7357,0.8193,0.748);
   SigCompSS2H =
(0.8075,0.8064,0.8059,0.8054,0.8043,0.8034,0.801,0.7986,0.7941,0.7899,0.7859,0.7822,0.7753,0.7693,0.7572,0.7484,0.737
3,0.7313,0.7258,0.7237,0.7228,0.7222,0.7221,0.8075,0.7431);
```

```
   SigCompSS3H =
(0.7348,0.7343,0.734,0.7338,0.7333,0.7329,0.7317,0.7306,0.7292,0.7295,0.7297,0.73,0.7305,0.7295,0.7256,0.7209,0.7137,
0.7093,0.7033,0.7009,0.6999,0.699,0.6989,0.7348,0.696);
   SigCompSS4H =
(0.6377,0.6379,0.6381,0.6382,0.6384,0.6385,0.639,0.6395,0.6413,0.6453,0.6492,0.6537,0.6623,0.6681,0.6762,0.6772,0.674
3,0.6738,0.6698,0.6687,0.6682,0.6679,0.6679,0.6377,0.6419);
   S1 = SigCompSS1H[thisRowIndex];
   S2 = SigCompSS2H[thisRowIndex];
   S3 = SigCompSS3H[thisRowIndex];
   S4 = SigCompSS4H[thisRowIndex];
   SigmaCompSSH = getSigmaCompSSi(inMagnitude, S1, S2, S3, S4);
   # Low Model
   SigCompSS1L =
(0.5879,0.5873,0.587,0.5867,0.586,0.5855,0.584,0.5825,0.5795,0.5765,0.5736,0.5707,0.565,0.5595,0.5467,0.5354,0.5171,0.
5037,0.487,0.4784,0.474,0.4702,0.4695,0.5879,0.522);
   SigCompSS2L =
(0.5682,0.5676,0.5672,0.5669,0.5662,0.5657,0.5641,0.5626,0.5595,0.5564,0.5533,0.5503,0.5444,0.5387,0.5253,0.5135,0.49
44,0.4803,0.4628,0.4538,0.4491,0.4451,0.4444,0.5682,0.5159);
   SigCompSS3L =
(0.4981,0.4976,0.4973,0.497,0.4964,0.496,0.4946,0.4933,0.4917,0.4921,0.4924,0.4928,0.4931,0.4913,0.4837,0.4741,0.4566
,0.4433,0.4243,0.4148,0.4101,0.4061,0.4053,0.4981,0.4703);
   SigCompSS4L =
(0.3901,0.3899,0.3898,0.3897,0.3895,0.3893,0.3887,0.3882,0.3904,0.3994,0.4079,0.4168,0.4326,0.4411,0.4504,0.4508,0.44
82,0.4416,0.4226,0.4124,0.4071,0.4025,0.4017,0.3901,0.3928);
   S1 = SigCompSS1L[thisRowIndex];
   S2 = SigCompSS2L[thisRowIndex];
   S3 = SigCompSS3L[thisRowIndex];
   S4 = SigCompSS4L[thisRowIndex];
   SigmaCompSSL = getSigmaCompSSi(inMagnitude, S1, S2, S3, S4);


   SigmaCompSS = SigmaCompSSC * WeightSigmaCompSSC / WeightSigmaCompSSSum + SigmaCompSSH *
WeightSigmaCompSSH / WeightSigmaCompSSSum + SigmaCompSSL * WeightSigmaCompSSL / WeightSigmaCompSSSum;
   return SigmaCompSS



def getSigmaCompSSi(inMagnitude, S1, S2, S3, S4):

   if inMagnitude <= 4.5:
      SigmaCompSSi = S1;
   elif inMagnitude <= 5.0:
      SigmaCompSSi = S1 + (S2 - S1) * (inMagnitude - 4.5) / 0.5;
   elif inMagnitude <= 5.5:
      SigmaCompSSi = S2 + (S3 - S2) * (inMagnitude - 5.0) / 0.5;
   elif inMagnitude <= 6.5:
      SigmaCompSSi = S3 + (S4 - S3) * (inMagnitude - 5.5) / 1.0;
   else:
      SigmaCompSSi = S4;
   return SigmaCompSSi







def getFrequencyListCompErg():
```

```
FrequencyListCompErg=(100,50,40,33.333,25,20,13.333,10,6.667,5,4,3.333,2.5,2,1.333,1,0.667,0.5,0.333,0.25,0.2,0.133,0.1,
-1,-2);
   return FrequencyListCompErg


def getSigmaCompErg(thisFreq, inMagnitude, inRrup):
   WeightSigmaCompErgC=0.63;
   WeightSigmaCompErgH=0.185;
   WeightSigmaCompErgL=0.185;
   WeightSigmaCompErgSum=1;
   FrequencyListCompErg = getFrequencyListCompErg();


   thisRowIndex = -999;
   NrowFErg = len(FrequencyListCompErg);
   for iRowF in range(NrowFErg):
      hereFreq = FrequencyListCompErg[iRowF];
      if abs(hereFreq - thisFreq) < 0.000001:
         thisRowIndex = iRowF;

   # Central Model

SigCompErg1C=(0.7618,0.761,0.7646,0.7676,0.781,0.792,0.7881,0.7726,0.7597,0.7556,0.7523,0.7488,0.7412,0.7335,0.7151,
0.6992,0.6742,0.6562,0.6337,0.6212,0.6139,0.6053,0.6016,0.7618,0.7045);

SigCompErg2C=(0.7471,0.7463,0.75,0.753,0.7667,0.7779,0.7739,0.7581,0.745,0.7408,0.7374,0.7338,0.7261,0.7182,0.6995,0
.6832,0.6576,0.6392,0.616,0.6032,0.5956,0.5867,0.5829,0.7471,0.6995);

SigCompErg3C=(0.6819,0.6814,0.6856,0.6891,0.7044,0.7169,0.7133,0.6969,0.685,0.6844,0.6844,0.6842,0.6825,0.6786,0.66
61,0.6528,0.6299,0.6127,0.5885,0.5749,0.567,0.5576,0.5535,0.6819,0.6569);

SigCompErg4C=(0.5809,0.5808,0.5861,0.5905,0.609,0.6239,0.6214,0.6039,0.5941,0.6001,0.6065,0.6129,0.6231,0.6274,0.62
83,0.6215,0.605,0.5907,0.5651,0.5509,0.5426,0.5327,0.5285,0.5809,0.5825);
   S1 = SigCompErg1C[thisRowIndex];
   S2 = SigCompErg2C[thisRowIndex];
   S3 = SigCompErg3C[thisRowIndex];
   S4 = SigCompErg4C[thisRowIndex];
   SigmaCompErgC = getSigmaCompErgi(inMagnitude, S1, S2, S3, S4);
   # High Model

SigCompErg1H=(0.8747,0.8737,0.8767,0.8791,0.8909,0.9011,0.899,0.8858,0.8725,0.8668,0.8621,0.8575,0.8485,0.8402,0.82
28,0.8096,0.7916,0.7807,0.7687,0.7627,0.7591,0.7545,0.7519,0.8747,0.8134);

SigCompErg2H=(0.8635,0.8625,0.8655,0.8679,0.8799,0.8902,0.888,0.8747,0.8613,0.8555,0.8508,0.8461,0.837,0.8286,0.811
,0.7976,0.7792,0.7681,0.7558,0.7496,0.746,0.7412,0.7386,0.8635,0.809);

SigCompErg3H=(0.7963,0.7958,0.7993,0.8022,0.8158,0.8275,0.8267,0.8137,0.8023,0.8004,0.7994,0.7983,0.7956,0.7918,0.7
814,0.7717,0.7568,0.7471,0.7343,0.7276,0.7238,0.7188,0.716,0.7963,0.7661);

SigCompErg4H=(0.7076,0.7078,0.7121,0.7157,0.7315,0.7452,0.7457,0.7328,0.7233,0.7246,0.727,0.7295,0.734,0.736,0.7361
,0.7314,0.7203,0.7139,0.7025,0.6967,0.6932,0.6883,0.6855,0.7076,0.7163);
   S1 = SigCompErg1H[thisRowIndex];
   S2 = SigCompErg2H[thisRowIndex];
   S3 = SigCompErg3H[thisRowIndex];
   S4 = SigCompErg4H[thisRowIndex];
   SigmaCompErgH = getSigmaCompErgi(inMagnitude, S1, S2, S3, S4);
   # Low Model
```

```
SigCompErg1L=(0.6539,0.6533,0.6575,0.661,0.6758,0.6874,0.682,0.6645,0.6521,0.6496,0.6476,0.6452,0.6392,0.6323,0.614
5,0.5977,0.5703,0.5497,0.5229,0.5076,0.4983,0.4871,0.4826,0.6539,0.6034);

SigCompErg2L=(0.6361,0.6355,0.6398,0.6434,0.6586,0.6705,0.665,0.6471,0.6343,0.6317,0.6296,0.6271,0.621,0.6138,0.595
4,0.578,0.5496,0.5282,0.5004,0.4844,0.4747,0.463,0.4583,0.6361,0.598);

SigCompErg3L=(0.5736,0.5731,0.5779,0.5819,0.5987,0.6118,0.606,0.5867,0.5743,0.575,0.5761,0.5767,0.576,0.5723,0.5586,
0.543,0.5156,0.4945,0.4648,0.448,0.4379,0.4256,0.4207,0.5736,0.5587);

SigCompErg4L=(0.4807,0.4805,0.4864,0.4913,0.5113,0.5264,0.52,0.4982,0.4876,0.4958,0.504,0.5119,0.5244,0.5293,0.5299,
0.5227,0.508,0.4926,0.4624,0.4446,0.4337,0.4205,0.4153,0.4807,0.4936);
   S1 = SigCompErg1L[thisRowIndex];
   S2 = SigCompErg2L[thisRowIndex];
   S3 = SigCompErg3L[thisRowIndex];
   S4 = SigCompErg4L[thisRowIndex];
   SigmaCompErgL = getSigmaCompErgi(inMagnitude, S1, S2, S3, S4);


   SigmaCompErg = SigmaCompErgC * WeightSigmaCompErgC / WeightSigmaCompErgSum + SigmaCompErgH *
WeightSigmaCompErgH / WeightSigmaCompErgSum + SigmaCompErgL * WeightSigmaCompErgL /
WeightSigmaCompErgSum;
   return SigmaCompErg


def getSigmaCompErgi(inMagnitude, S1, S2, S3, S4):
   if inMagnitude <= 4.5:
      SigmaCompErgi = S1;
   elif inMagnitude <= 5.0:
      SigmaCompErgi = S1 + (S2 - S1) * (inMagnitude - 4.5) / 0.5;
   elif inMagnitude <= 5.5:
      SigmaCompErgi = S2 + (S3 - S2) * (inMagnitude - 5.0) / 0.5;
   elif inMagnitude <= 6.5:
      SigmaCompErgi = S3 + (S4 - S3) * (inMagnitude - 5.5) / 1.0;
   else:
      SigmaCompErgi = S4;
   return SigmaCompErgi




########################################################
############## END PROGRAM DATA ########################
########################################################

runNGAEastGMM(inPeriodList,inMagnitude,inRrup,inSigmaType,inNsigma)
```

5. Details of NGA-Sub GMM Tools

5.1 NGA-Sub Ground-Motion Characterization Tool – MS Excel

The MS Excel version provides insight into the implementation of the NGA-Sub Ground-Motion Models and is presented here as a demonstration of the tool. The tool consists of two parts: (1) A single-scenario study, and (2) a sensitivity study. These two parts are contained within two separate worksheets in the Excel file.

The first worksheet, which contains the scenario study also has information about the user input and program output. Details of this worksheet are described in this section. The first part of the worksheet consists of title section, which includes the author, the release date, a reference to the PEER report, and simple instructions, as shown in **Figure *1***.

Figure 1: NGA-Sub Tool, MS Excel Version: Title table.

The instructions are given as: "*Instruction: Make selection in User-Input box. Some input values (Models and Types) are to be selected from a pull-down menu, others require numeric input. When applicable, the pull-down menu will appear to the right of the selected input cell (see image to the right). Model limits are given in the Model-Limits box. Values outside these limits are not allowed. Input parameters are "greyed out" when not needed for the selected input conditions. The Median and Sigma (σ) models are described in the report. NOTE: The program computes the geometric mean of the median ground-motion models, and the SRSS of the aleatory variability and epistemic uncertainty (see bottom of this page*)."

The User-Input table is shown in **Figure *2***, where the user input is highlighted in yellow. The cells that are in a lighter color have pull-down menus with options for the user. The values of some input menus depend on other user input, as shown in **Figure *3***. For example, the options for the Subducting Slab are dependent on the Region being selected. The columns to the right of the user-input column gives warnings to the user in case of errors in input.

User Input:		input okey
Region	2_Cascadia	
Subducting Slab (used to compute Mb)	Cascadia	
Moment Magnitude	7.5	
VS30 (m/sec)	800	
Rupture Distance (km)	56	
Fraction of Rrup in Backarc	1	
Fraction of Rrup in Nankai Region (Japan only)	0	
Z_tor (km) (KBCG20 only)	38	
Hypocentral Depth (km) (PSHAB20 only)	35	
Event Type	1_Intraslab	
Z1.0 input (m) (KBCG20 only)(can set = default)	31	
Z2.5 input (m) (can set = default)	57	
Mb (KBCG20 only, can set = default)	default	
PNW Basin Structure (Cascadia Only)	0_NoBasin	
Relative Weight -- KBCG20	1	
Relative Weight -- PSHAB20	0	
ne epistemic uncertainty (in median) with aleatory variability	1_Yes	
# Samples in Epistemic Uncertainty (0=none or 100-800) (slows computation)	100	
Number of Sigma above/below Median	1	

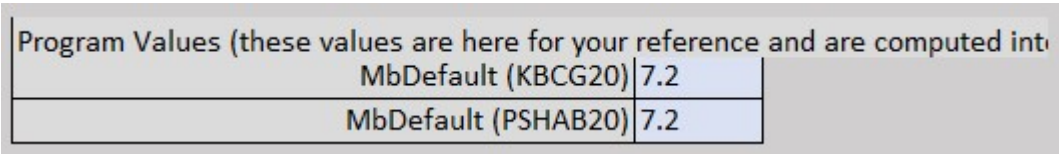
Figure 2: NGA-Sub Tool -- User-Input Table

User Input:		input okey
Region	2_Cascadia	
Subducting Slab (used to compute Mb)	0_global 1_Alaska 2_Cascadia 3_CentralAmerica&Mexico 4_Japan 5_NewZealand 6_SouthAmerica 7_Taiwan	
Moment Magnitude		
VS30 (m/sec)		
Rupture Distance (km)		
Fraction of Rrup in Backarc		
Fraction of Rrup in Nankai Region (Japan only)		
Z_tor (km) (KBCG20 only)	38	
Hypocentral Depth (km) (PSHAB20 only)	35	
Event Type	1_Intraslab	
Z1.0 input (m) (KBCG20 only)(can set = default)	31	
Z2.5 input (m) (can set = default)	57	
Mb (KBCG20 only, can set = default)	default	
PNW Basin Structure (Cascadia Only)	0_NoBasin	
Relative Weight -- KBCG20	1	
Relative Weight -- PSHAB20	0	
ine epistemic uncertainty (in median) with aleatory variability	1_Yes	
# Samples in Epistemic Uncertainty (0=none or 100-800) (slows computation)	100	
Number of Sigma above/below Median	1	

Figure 3: NGA-Sub Tool -- User-Input Table with Pull-Down Menu for Region

A feature unique to the NGA-Sub Ground-Motion Characterization tool is the new option of whether to include epistemic uncertainty along with aleatory variability in the Sigma term. The user can use the pull-down menu to make this choice. As presented in the PEER Report, the epistemic uncertainty model for the KBCG20 model is computed via a sampling method. The user can choose the number of samples to be used in the computation. Including epistemic uncertainty reflect the fact that the prediction of the median is uncertain.

The Excel worksheet also provides additional tables to assist the user. Based on the user-input, the default values for the Magnitude-Break value are given in the table. The user may use these default values or may enter a preferred value for the Mb in the input, as shown in **Figure *2***. **Figure *4*** presents the default Magnitude-Break Values for the current input.



Program Values (these values are here for your reference and are computed int	
MbDefault (KBCG20)	7.2
MbDefault (PSHAB20)	7.2

Figure 4: NGA-Sub Tool -- Default Magnitude-Break Values

The Excel worksheet also includes a table that provides insight into the input parameters, with a brief description, a marker to specify whether a GMM uses this parameter, and a recommended range. The user, however, is directed to the reference PEER reports for a more detailed and accurate discussion of these input parameters. This table is shown in **Figure *5***.

Input Parameter	KBCG20	PSHAB20	Description	Range
Region	x	x	Geographic region for model. These regions corresponds to the regions in the NGAsubudction database used to develop the models	0_global, 1_Alaska, 2_Cascadia, 3_CentralAmerica&Mexico, 4_Japan, 5_NewZealand (PSHAB20 uses global model), 6_SouthAmerica, 7_Taiwan
Subducting Slab		x	Saturation Region corresponds to regions defined by R. Archuleta and C. Ji. Used by PSHAB20 to compute Mb	Alaska,Aleutian,Cascadia,Central_America_N,Central_America_S,global,Japan_Pac,Japan_Phi, New_Zealand_N,New_Zealand_S,South_America_N,South_America_S,Taiwan_E,Taiwan_W
Magnitude	x	x	Earthquake magnitude	
VS30 (m/sec)	x	x	The average seismic shear-wave velocity from the surface to a depth of 30 meters	
Rupture Distance (km)	x	x	The shortest distance from a seismic station or any other site to the rupture surface of the seismic event	
Fraction of Rrup in Backarc	x		Fraction of Rrup in Backarc region (value <=1)	0-1
Fraction of Rrup in Nankai Region	x		Fraction of Rrup in Nankai Region (value <=1) (Japan only)	0-1
Z_tor (km)	x		Depth to top of rupture in km	
Hypocentral Depth (km)		x	Depth to Hypocenter in km	
Event Type	x	x	Event Type: Interface or Intraslab	0_Interface,1_Intraslab
Z1.0 input (m)	x (TW, NZ)	x (AK,NZ)	Depth to a shear wave horizon of 1000 m/s, in m. Used by PSHAB20 for regions Alaska and New Zealand. Used by KBCG for regions Taiwan and New Zealand. (Can set =default)	
Z2.5 input (m)	x (JP, CSC)	x (JP, CSC)	Depth to a shear wave horizon of 2500 m/s, in m. PSHAB used for regions Cascadia and Japan. KBCG used for regions Japan and Cascadia. (Can set =default)	
Mb	x		Magnitude scaling break point. Should be set regionally dependent based on Campbell (2020). KBCG20 has optional input, PSHAB20 computes it internally	
PNW Basin Structure		x (CSC)	Used if DatabaseRegion = "Cascadia". PSHAB20: 0 = having an estimate of Z2pt5 outside mapped basin, 1 = Seattle basin, and 0 = other mapped basin (Tacoma, Everett, Georgia, etc.), KBCG20: uses flat whether site is in Seattle Basin or not.	0_NoBasin,1_InSeattleBasin,2_InOtherPNWbasin
No. of Samples for Epistemic uncertainty	x		Number of scenarioes to be used to compute Epistemic uncertainty using KBCB20 model. A large number of samples may take some time to run	0 or 50-800 (recommend 100) (0: no Epistemic)

Figure 5: Input-Parameter Description Table

To assist the user, the MS Excel worksheet includes a table that lists the input regions, and how each model uses it, as shown in **Figure *6***. All the default Magnitude-Break values for each Region, and for Interface and Intraslab sources for each model, are provided in **Figure *7***.

Input Region:	KBCG20	PSHAB20
0_global	x	x
1_Alaska	x	x
2_Cascadia	x	x
3_CentralAmerica&Mexico	x	x
4_Japan	x	x
5_NewZealand	x	global
6_SouthAmerica	x	x
7_Taiwan	x	x

Figure 6: NGA-Sub Regions

Magnitude Break Point (Default Values)														
Region		0_global	1_Alaska	1_Alaska	2_Cascadia	3_CentralAmerica	3_CentralAmerica&M	4_Japan	4_Japan	5_NewZealand	5_NewZealand	6_SouthAmerica	6_SouthAmerica	7_Taiwan	7_Taiwan
Subducting	Slab	global	Aleutian	Alaska	Cascadia	Central_America_N	Central_America_S	Japan_Pac	Japan_Phi	New_Zealand_N	New_Zealand_S	South_America_N	South_America_S	Taiwan_W	Taiwan_E
0_Interface	KB	7.9	8	8.6	8	7.4	7.5	8.5	7.7	8.3	8	8.5	8.6	7.1	7.1
0_Interface	PSH	7.9	8	8.6	7.7	7.4	7.4	8.5	7.7	7.9	7.9	8.5	8.6	7.1	7.1
1_Intraslab	KB	7.6	8	7.2	7.2	7.4	7.6	7.6	7.6	7.6	7.6	7.3	7.2	7.7	7.7
1_Intraslab	PSH	7.6	7.98	7.2	7.2	7.4	7.6	7.65	7.55	7.6	7.6	7.3	7.25	7.7	7.7

Figure 7: NGA-Sub Magnitude-Break Points

The last table that is intended to assist the user found in the MS Excel tool is the Output Table, which provides insight into the definition of each output values, as shown in **Figure *8***.

OUTPUT	
Output	**Simplified Equation**
Weighted PSA +/- N*sigma (g)	= exp[(ln(PSA_KBCG20 * Weight_KBCG20 + ln(PSA_PSHAB20) * Weight_PSHAB20) / (Weight_KBCG20 + Weight_PSHAB20)]
PSA_KBCG20	= exp(ln(MedianPSA_KBCG20) +/- N*$\sigma_{total_}$KBCG20)
PSA_PSHAB20	= exp(ln(MedianPSA_PSHAB20) +/- N*$\sigma_{total_}$PSHAB20)
$\sigma_{total_}$KBCG20	= sqr($\sigma_{aleatory_}$KBCG20 * $\sigma_{aleatory_}$KBCG20 + EpistemicFlag*$\sigma_{epistemic_}$KBCG20 * $\sigma_{epistemic_}$KBCG20)
$\sigma_{total\_PSHAB20}$	= sqr($\sigma_{aleatory_}$PSHAB20 * $\sigma_{aleatory_}$PSHAB20 + EpistemicFlag*$\sigma_{epistemic_}$PSHAB20 * $\sigma_{epistemic_}$PSHAB20)
$\sigma_{aleatory_}$KBCG20	= sqr(Phi * Phi + Tau * Tau)
$\sigma_{aleatory_}$PSHAB20	= sqr(PhiTot * PhiTot + Tau * Tau)
$\phi_{total_}$KBCG20	= sqr(phiSquared + deltaVar)
$\sigma_{epistemic_}$KBCG20	= stdev(PSAposterior,)
$\sigma_{epistemic_}$PSHAB20	=σepistemic_PSHAB20

Figure 8: NGA-Sub Output Table

The scenario Weighted Spectra values are tabulated and plotted, as shown in **Figure *9***.

In addition to the weighted-average spectra, the worksheet also tabulates and plots all the components and values for each NGA-Sub GMM, as shown in **Figure *10*** and **Figure *11***. These tables are intended to allow the user to visualize the individual components as well as be able to use the individual internal functions to compute other combinations of the models. An example value for a cell is as follows: “KBCG20_medPSA($Y17, User_Magnitude,User_Rrup, User_AlphaBackarc, User_AlphaNankai, User_Ztor, User_EventType, User_Vs30,User_Z1pt0,User_Z2pt5, IF(LOWER(User_Mb)="default",Prog_MbDefault,User_Mb),User_Region,User_PNWbasinStrux)“.

This format is intuitive enough for a user to be able to use it.

Quantity	Period (sec)	Median PSA	Median - 1*σ	Median + 1*σ	Frequency (Hz)
Response Spectrum PSA (g)	0.010	0.1920	0.08328	0.44276	100.00
	0.020	0.1902	0.08401	0.43069	50.00
	0.030	0.2033	0.08642	0.47801	33.33
	0.050	0.2510	0.10442	0.60326	20.00
	0.075	0.3038	0.12247	0.75379	13.33
	0.100	0.3513	0.14223	0.8677	10.00
	0.150	0.4221	0.17188	1.03652	6.67
	0.200	0.4422	0.18257	1.07118	5.00
	0.250	0.4228	0.18029	0.99147	4.00
	0.300	0.3912	0.17049	0.89772	3.33
	0.400	0.3275	0.14797	0.72504	2.50
	0.500	0.2767	0.12604	0.60731	2.00
	0.750	0.1868	0.08566	0.40734	1.33
	1.000	0.1319	0.06038	0.28801	1.00
	1.500	0.0806	0.03665	0.17738	0.67
	2.000	0.0615	0.02781	0.1358	0.50
	3.000	0.0306	0.01392	0.06734	0.33
	4.000	0.0189	0.00876	0.04072	0.25
	5.000	0.0135	0.00636	0.02864	0.20
	7.500	0.0070	0.00346	0.01432	0.13
	10.000	0.0043	0.00222	0.0084	0.10
(g)	PGA	0.1883	0.08355	0.42438	PGA
(cm/s)	PGV	17.6730	8.78192	35.5656	PGV

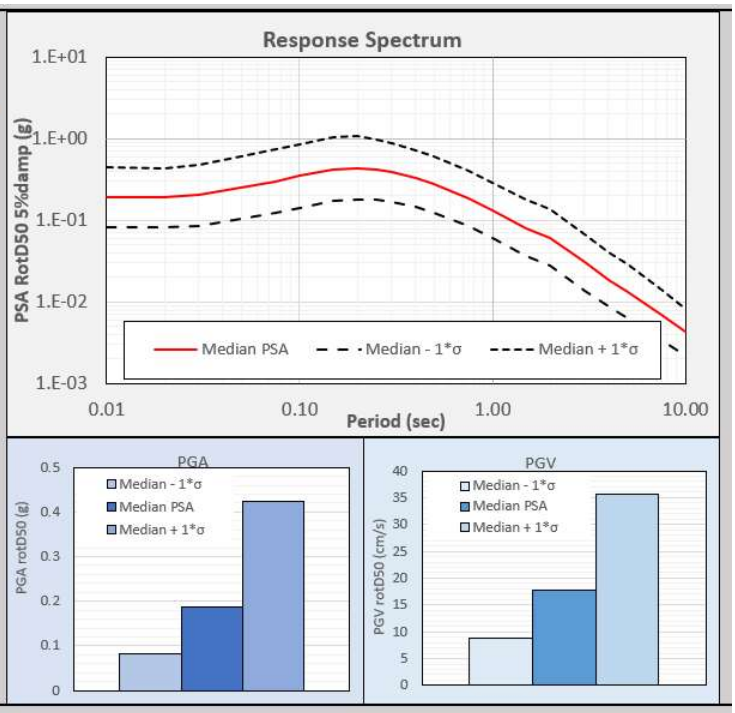


Figure 9: NGA-Sub Tool -- Scenario Spectra

Individual-model intermediate output. The input for this table is the same as that for the individual scenario, in column D.

	KBCG20												PSHAB20												
Tperiod (sec)	KBCG20 MedianPSA	KBCG20 Tau	KBCG20 Phi	KBCG20 SigmaAleatory	KBCG20 SigmaEpistemic	KBCG20 SigmaTotal	KBCG20 Median - 1*Sigma Total	KBCG20 Median + 1*Sigma Total	KBCG20 Median - 1*Sigma Aleatory	KBCG20 Median + 1*Sigma Aleatory	KBCG20 Median - 1*Sigma Epistemi	KBCG20 Median + 1*Sigma Epistemi	PSHAB20 MedianPSA	PSHAB20 Tau	PSHAB20 PhiTot	PSHAB20 PhiS2S	PSHAB20 PhiSS	PSHAB20 SigmaAleatory	PSHAB20 SigmaEpistemic	PSHAB20 SigmaTotal	PSHAB20 Median - 1*Sigma Total	PSHAB20 Median + 1*Sigma Total	PSHAB20 Median - 1*Sigma Aleatory	PSHAB20 Median + 1*Sigma Aleatory	PSHAB20 Median - 1*SigmaEpistemic	PSHAB20 Median + 1*SigmaEpistemic	Frequency (Hz)
0.01	0.19203	0.49423	0.5996	0.77703	0.30676	0.83539	0.08328	0.44276	0.08829	0.41766	0.1413	0.26097	0.2832602	0.476	0.63008	0.53427	0.4396	0.78967	0.35	0.86376	0.11942	0.67191	0.1286	0.62393	0.19961	0.401965	100
0.02	0.19021	0.50546	0.59242	0.77875	0.24788	0.81724	0.08401	0.43069	0.08731	0.41443	0.14845	0.24372	0.3068865	0.482	0.63325	0.54213	0.44217	0.79582	0.35	0.86938	0.12865	0.73206	0.13847	0.68014	0.216259	0.435493	50
0.03	0.20325	0.51167	0.60443	0.79192	0.32284	0.8552	0.08642	0.47801	0.09207	0.4487	0.14717	0.2807	0.3305853	0.5	0.64265	0.57567	0.44952	0.81425	0.35	0.88628	0.13626	0.80203	0.14644	0.74629	0.23296	0.469123	33.33
0.05	0.25099	0.51579	0.63802	0.82043	0.30974	0.87695	0.10442	0.60326	0.11049	0.57011	0.18413	0.34211	0.4745474	0.528	0.68775	0.64222	0.46483	0.86705	0.35	0.93503	0.18629	1.20881	0.1994	1.12937	0.334408	0.673415	20
0.075	0.30384	0.51487	0.66612	0.84191	0.34176	0.90863	0.12247	0.75379	0.13092	0.70514	0.21588	0.42762	0.732307	0.53	0.72732	0.6853	0.47344	0.89994	0.35	0.96561	0.27883	1.92332	0.29775	1.80108	0.516048	1.039193	13.33
0.1	0.3513	0.51197	0.67837	0.84988	0.3087	0.90421	0.14223	0.8677	0.15017	0.82181	0.25799	0.47834	0.8785165	0.524	0.71903	0.68183	0.46354	0.88971	0.35	0.95607	0.3377	2.28543	0.36087	2.13867	0.61908	1.246674	10
0.15	0.42209	0.50552	0.67774	0.8455	0.30375	0.89841	0.17188	1.03652	0.18122	0.98311	0.31152	0.5719	0.9406215	0.51	0.67602	0.6256	0.4609	0.84682	0.35	0.9163	0.37625	2.35157	0.40332	2.19373	0.662845	1.334805	6.67
0.2	0.44222	0.50008	0.66465	0.83177	0.30142	0.8847	0.18257	1.07118	0.19249	1.01596	0.32714	0.59779	0.8612618	0.501	0.65727	0.58915	0.45564	0.82644	0.35	0.8975	0.35104	2.11307	0.37689	1.96813	0.606921	1.222189	5
0.25	0.42279	0.49585	0.64953	0.81716	0.24224	0.85231	0.18029	0.99147	0.18674	0.95723	0.33184	0.53868	0.7307287	0.492	0.67082	0.57951	0.46698	0.8319	0.33434	0.89658	0.29811	1.79116	0.31803	1.67899	0.523061	1.020845	4
0.3	0.39122	0.49261	0.63579	0.80429	0.20736	0.83059	0.17049	0.89772	0.17503	0.87442	0.31796	0.48136	0.5678002	0.492	0.6603	0.55913	0.44768	0.82345	0.32155	0.884	0.23457	1.3744	0.24922	1.29364	0.411668	0.783148	3.33
0.4	0.32754	0.48821	0.61531	0.78547	0.12023	0.79462	0.14797	0.72504	0.14933	0.71844	0.29043	0.36938	0.468839	0.492	0.65803	0.50797	0.44585	0.82162	0.30137	0.87515	0.19541	1.12485	0.20616	1.06623	0.34685	0.633738	2.5
0.5	0.27667	0.48561	0.60326	0.77443	0.13561	0.78621	0.12604	0.60731	0.12754	0.60019	0.24158	0.31685	0.3687097	0.492	0.65422	0.49921	0.43363	0.81857	0.28571	0.867	0.15494	0.87744	0.16262	0.83596	0.277078	0.490645	2
0.75	0.1868	0.48286	0.59477	0.7661	0.14456	0.77962	0.08566	0.40734	0.08683	0.40188	0.16166	0.21585	0.234607	0.492	0.66933	0.49354	0.44787	0.8307	0.25726	0.86963	0.09833	0.55978	0.10223	0.53841	0.18139	0.303438	1.33
1	0.13187	0.48233	0.59897	0.76903	0.13727	0.78118	0.06038	0.28801	0.06112	0.28453	0.11496	0.15127	0.1631548	0.492	0.65574	0.48641	0.4717	0.8198	0.23708	0.85339	0.0695	0.38302	0.07187	0.37037	0.128717	0.206806	1
1.5	0.08063	0.48285	0.61155	0.77919	0.12003	0.78838	0.03665	0.17738	0.03699	0.17576	0.07151	0.09092	0.1069583	0.492	0.63718	0.46253	0.42691	0.80502	0.20863	0.83162	0.04656	0.24569	0.04782	0.23924	0.086817	0.131772	0.67
2	0.06145	0.48354	0.61735	0.78417	0.11783	0.79297	0.02781	0.1358	0.02805	0.13461	0.05462	0.06914	0.0653243	0.492	0.6269	0.45604	0.4159	0.79691	0.18845	0.81889	0.0288	0.14815	0.02944	0.14493	0.054104	0.078871	0.5
3	0.03062	0.48402	0.60929	0.77815	0.12444	0.78803	0.01392	0.06734	0.01406	0.06668	0.02704	0.03468	0.036894	0.492	0.60581	0.42379	0.41217	0.78043	0.16	0.79666	0.01663	0.08184	0.01691	0.08052	0.031439	0.043296	0.33
4	0.01889	0.4836	0.58779	0.76116	0.10232	0.76801	0.00876	0.04072	0.00882	0.04044	0.01705	0.02093	0.0215639	0.492	0.57446	0.40439	0.42377	0.75635	0.16	0.77309	0.00995	0.04672	0.01012	0.04594	0.018376	0.025305	0.25
5	0.0135	0.48273	0.5621	0.74094	0.1292	0.75212	0.00636	0.02864	0.00643	0.02832	0.01186	0.01536	0.0150546	0.492	0.54589	0.42928	0.37575	0.73489	0.16	0.75211	0.0071	0.03194	0.00722	0.03139	0.012829	0.017667	0.2
7.5	0.00704	0.48003	0.50092	0.69379	0.15191	0.71023	0.00346	0.01432	0.00352	0.01408	0.00605	0.00819	0.0070789	0.492	0.50398	0.4131	0.35121	0.70432	0.16	0.72226	0.00344	0.01458	0.0035	0.01432	0.006032	0.008307	0.13
10	0.00432	0.47748	0.45338	0.65844	0.09935	0.66589	0.00222	0.0084	0.00223	0.00834	0.00391	0.00477	0.0042704	0.492	0.48062	0.454	0.34738	0.6878	0.16	0.70616	0.00211	0.00865	0.00215	0.0085	0.003639	0.005011	0.1
0	0.1883	0.48874	0.59576	0.77058	0.2578	0.81256	0.08355	0.42438	0.08714	0.40693	0.14551	0.24368	0.2779233	0.48	0.62929	0.5294	0.43856	0.79145	0.35	0.86539	0.11697	0.66033	0.12595	0.61327	0.195849	0.394392	0
-1	17.673	0.45098	0.51149	0.68191	0.15516	0.69934	8.78192	35.5656	8.93632	34.9511	15.133	20.6392	16.40559	0.477	0.58992	0.41467	0.40001	0.75864	0.35	0.83548	7.11454	37.83	7.68281	35.0319	11.56082	23.28064	-1

Figure 10: NGA-Sub Individual-Model Components

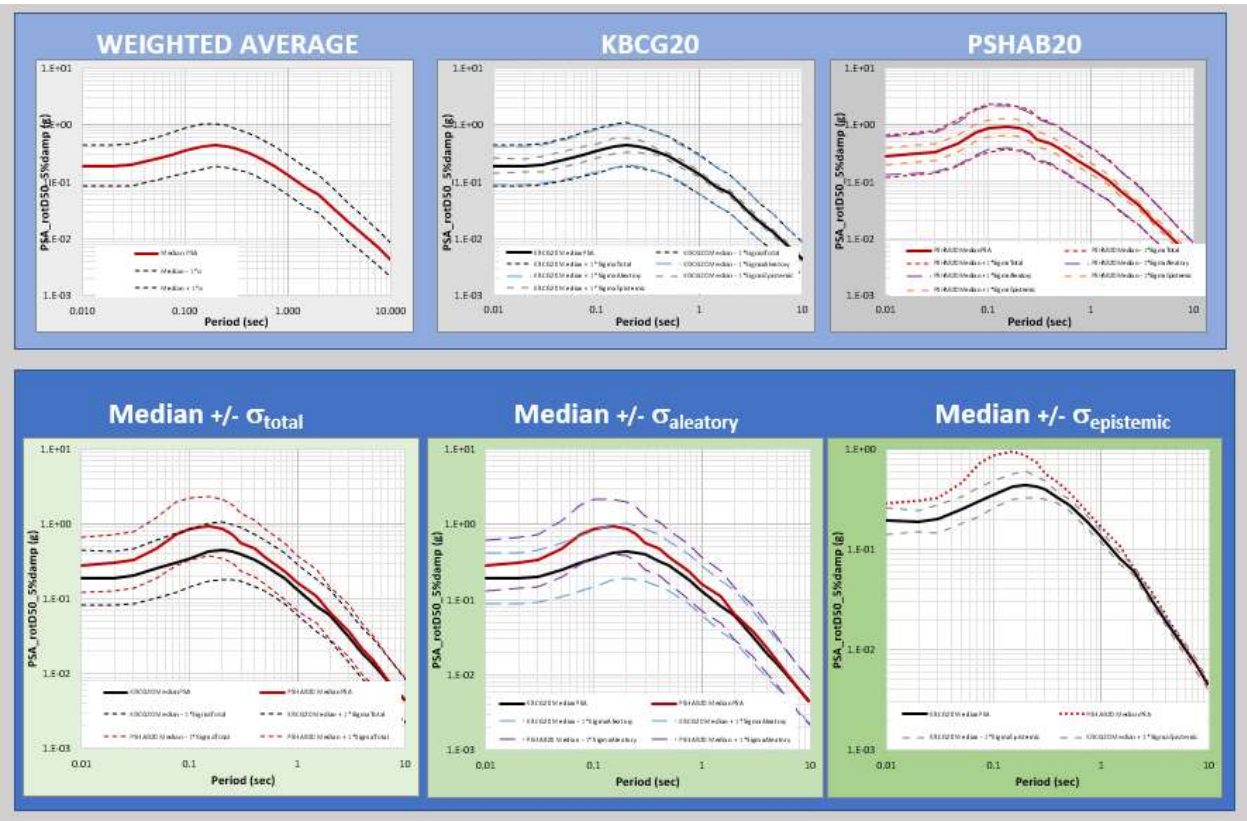
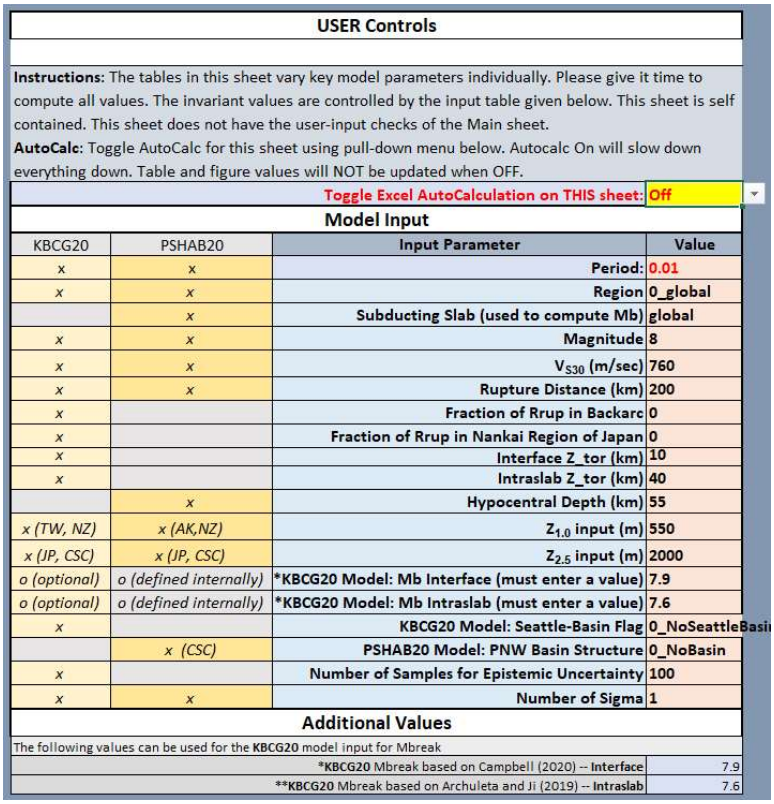


Figure 11: NGA-Sub Individual-Model Spectra

The second component of the NGA-Sub Ground-Motion Characterization tool is found in the SensitivityStudy worksheet. Even though this data is vast, it is self-explanatory and intuitive. All input values in this worksheet are independent of those of the Scenario-Study (main) worksheet. The user has an initial switch that enables or disables the computations in this worksheet. Since there are a lot of computations, it is recommended that this switch is set to "On" only when values are updated. This switch controls toggles the autocomputation feature of MS Excel. The User-Controls table for this section is shown in **Figure *12*** for the "Off" state and in **Figure *13*** for the "On" state. As shown in the figures, the user options are highlighted in yellow only in the On state.



USER Controls

Instructions: The tables in this sheet vary key model parameters individually. Please give it time to compute all values. The invariant values are controlled by the input table given below. This sheet is self contained. This sheet does not have the user-input checks of the Main sheet.
AutoCalc: Toggle AutoCalc for this sheet using pull-down menu below. Autocalc On will slow down everything down. Table and figure values will NOT be updated when OFF.

Toggle Excel AutoCalculation on THIS sheet:	Off

Model Input

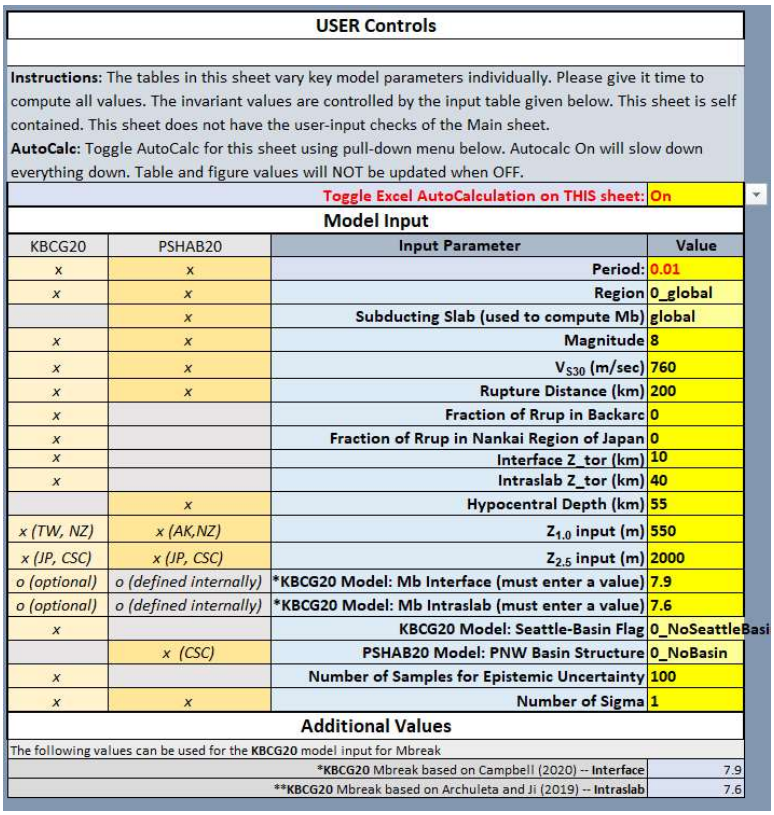
KBCG20	PSHAB20	Input Parameter	Value
x	x	Period:	0.01
x	x	Region	0_global
	x	Subducting Slab (used to compute Mb)	global
x	x	Magnitude	8
x	x	V_{S30} (m/sec)	760
x	x	Rupture Distance (km)	200
x		Fraction of Rrup in Backarc	0
x		Fraction of Rrup in Nankai Region of Japan	0
x		Interface Z_tor (km)	10
x		Intraslab Z_tor (km)	40
	x	Hypocentral Depth (km)	55
x (TW, NZ)	x (AK,NZ)	$Z_{1.0}$ input (m)	550
x (JP, CSC)	x (JP, CSC)	$Z_{2.5}$ input (m)	2000
o (optional)	o (defined internally)	*KBCG20 Model: Mb Interface (must enter a value)	7.9
o (optional)	o (defined internally)	*KBCG20 Model: Mb Intraslab (must enter a value)	7.6
x		KBCG20 Model: Seattle-Basin Flag	0_NoSeattleBasi
	x (CSC)	PSHAB20 Model: PNW Basin Structure	0_NoBasin
x		Number of Samples for Epistemic Uncertainty	100
x	x	Number of Sigma	1

Additional Values

The following values can be used for the **KBCG20** model input for Mbreak

*KBCG20 Mbreak based on Campbell (2020) -- Interface	7.9
**KBCG20 Mbreak based on Archuleta and Ji (2019) -- Intraslab	7.6

Figure 12: NGA-Sub Tool -- Sensitivity-Study Controls, Off Configuration



USER Controls

Instructions: The tables in this sheet vary key model parameters individually. Please give it time to compute all values. The invariant values are controlled by the input table given below. This sheet is self contained. This sheet does not have the user-input checks of the Main sheet.
AutoCalc: Toggle AutoCalc for this sheet using pull-down menu below. Autocalc On will slow down everything down. Table and figure values will NOT be updated when OFF.

		Toggle Excel AutoCalculation on THIS sheet:	On

Model Input

KBCG20	PSHAB20	Input Parameter	Value
x	x	Period:	0.01
x	x	Region	0_global
	x	Subducting Slab (used to compute Mb)	global
x	x	Magnitude	8
x	x	V_{S30} (m/sec)	760
x	x	Rupture Distance (km)	200
x		Fraction of Rrup in Backarc	0
x		Fraction of Rrup in Nankai Region of Japan	0
x		Interface Z_tor (km)	10
x		Intraslab Z_tor (km)	40
	x	Hypocentral Depth (km)	55
x (TW, NZ)	x (AK,NZ)	$Z_{1.0}$ input (m)	550
x (JP, CSC)	x (JP, CSC)	$Z_{2.5}$ input (m)	2000
o (optional)	o (defined internally)	*KBCG20 Model: Mb Interface (must enter a value)	7.9
o (optional)	o (defined internally)	*KBCG20 Model: Mb Intraslab (must enter a value)	7.6
x		KBCG20 Model: Seattle-Basin Flag	0_NoSeattleBasi
	x (CSC)	PSHAB20 Model: PNW Basin Structure	0_NoBasin
x		Number of Samples for Epistemic Uncertainty	100
x	x	Number of Sigma	1

Additional Values

The following values can be used for the **KBCG20** model input for Mbreak

*KBCG20 Mbreak based on Campbell (2020) -- Interface	7.9
**KBCG20 Mbreak based on Archuleta and Ji (2019) -- Intraslab	7.6

Figure 13: NGA-Sub Tool -- Sensitivity-Study Controls, On Configuration

The objective of the sensitivity study is to show the sensitivity of each GMM to variations in input. The input being studied is: Source type (Interface or Intraslab), Period, Magnitude, Rrup, Vs30, and Region. The Mean and Mean +/- N*sigma is tabulated in the worksheet. An excerpt of this feature is shown in **Figure *14***. All output values for each model are tabulated in rows. Interface and Intraslab events are handled separately. **Figure *15*** and **Figure *16***, together, show all the spectral plots for the sensitivity study.

KBCG20	PSHAB20	T=0.01sec	Mag=8,Vs30_mps=760,R 0_GLB_GLB_Mb7.9	Mag=8,Vs30_m 0_GLB_GLB_Mb	Mag=8,Vs30_mps= 0_GLB_GLB_Mb7.9	Mag=8,V 0_GLB_GL	Mag=8 0_GLB_
x	x	Magnitude	8	8	8	8	
x	x	VS30 (m/sec)	760	760	760	760	76
x	x	Region	0_global	0_global	0_global	0_global	0_glob
x	x	Rupture Distance (km)	200	200	200	200	20
x		Fraction of Rrup in Backarc	0	0	0	0	
x		Fraction of Rrup in Nankai Region of Japan	0	0	0	0	
x		**Interface Z_tor (km)**	**10**	10	10	10	1
x		KBCG20 Model: Seattle-Basin Flag	0_NoSeattleBasin	0_NoSeattleBas	0_NoSeattleBasin	0_NoSeat	0_NoS
x		Z1.0 input (m)	550	550	550	550	55
x	x (Japan, Cascadia)	Z2.5 input (m)	2000	2000	2000	2000	200
x		Number of Samples for Epistemic Uncertainty	100	100	100	100	10
x		*KBCG20 Model: Mb Interface (must enter a valu	7.9	7.9	7.9	7.9	7
	x	Hypocentral Depth (km)	55	55	55	55	5
	x	Subducting Slab (used to compute Mb)	global	global	global	global	global
	x (cascadia)	PSHAB20 Model: PNW Basin Structure	0_NoBasin	0_NoBasin	0_NoBasin	0_NoBasi	0_NoB
x	x	Number of Sigma	1	1	1	1	
x	x	**Type [0=Interface, 1=slab]**	**0_Interface**	0_Interface	0_Interface	0_Interfa	0_Inter
x	x	**Tperiod:**	0.01	0.02	0.03	0.05	0.07
x		KBCG20 MedianPSA	0.021505144	0.022013334	0.023223823	0.0265	0.029
x		KBCG20 Tau	0.494229731	0.505458398	0.511667679	0.5158	0.514
x		KBCG20 Phi	0.599595609	0.59241506	0.604430227	0.638	0.666
x		KBCG20 SigmaAleatory	0.77703148	0.778745014	0.791921532	0.8204	0.841
x		KBCG20 SigmaEpistemic	0.36643212	0.398598233	0.409045674	0.3344	0.418
x		KBCG20 SigmaTotal	0.859098609	0.874828182	0.891323777	0.886	0.940

Figure 14: NGA-Sub Tool: Sample of Sensitivity-Study Table.

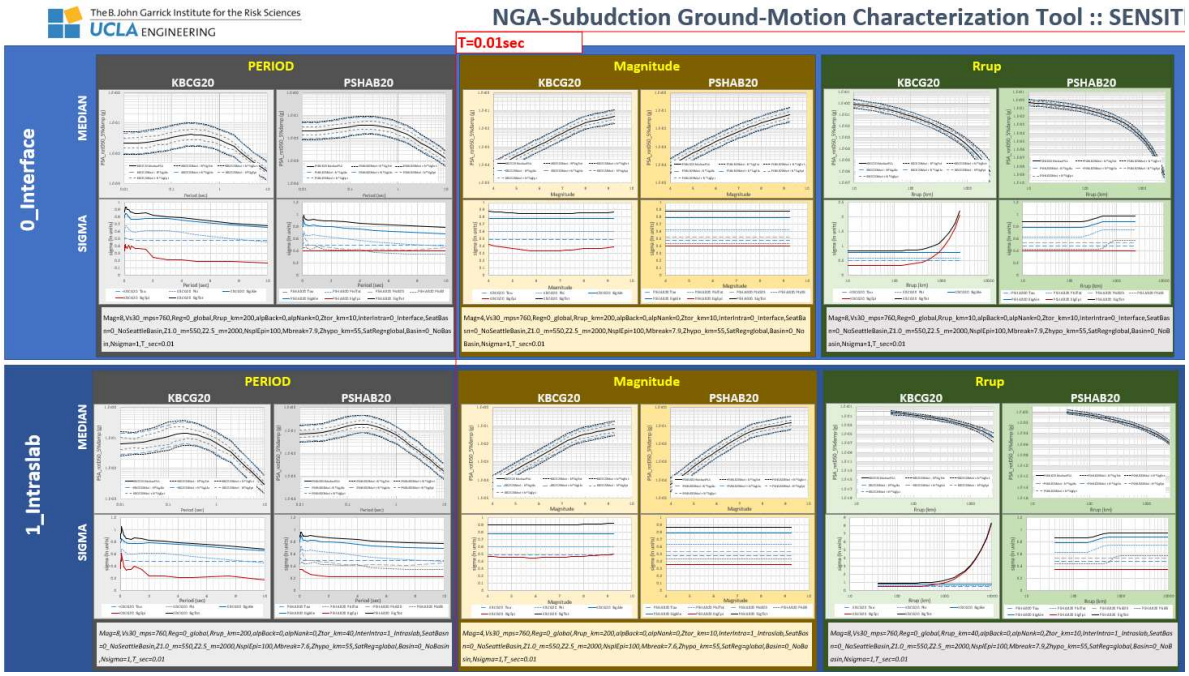


Figure 15: NGA-Sub Tool: Sensitivity-Study Spectra Plots, part A

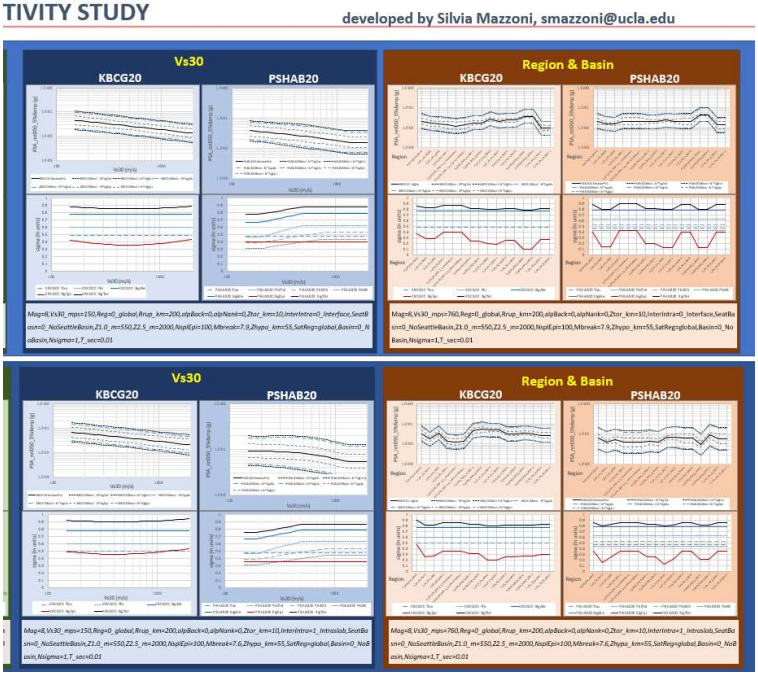


Figure 16: NGA-Sub Tool: Sensitivity-Study Spectra Plots, part B

5.1.1 NGA-Sub Ground-Motion Characterization Tools – MS Excel VB Computer Code

NGASubduction_GMMtool_Excel_r201031.xlsm

```
Option Base 1
Function NGAsubGMM_Weighted(UserPeriod, User_Nsigma, User_EpiInSigmaModels, User_NsampleEpi, UserRegion,
Magnitude, Vs30, Rrup, AlphaBackarc, AlphaNankai, Ztor, Zhypo, EventType, Z1pt0, Z2pt5, MbUser, SubductionSlab,
PNWbasinStrux, Weight_KBCG20, Weight_PSHAB20, Optional User_MeanType = "geometric")
   ' VBA CODE DEVELOPED/IMPLEMENTED BY
   '        Silvia Mazzoni, 2020
   '         smazzoni@ucla.edu
   NGAsubGMM_Weighted = 0

   User_AleInSigmaModels = "1_AllModels"   ' force this for the combined case

 '   Debug.Print "start" & Nsigma
   If LCase(MbUser) = LCase("default") Then
      MbKBCG20 = getMbDefault("KBCG20", EventType, UserRegion, SubductionSlab)
   Else
      MbKBCG20 = MbUser
   End If

   ' 201005: KBCG20 now uses the 3-value option for the basin terms in cascadia
'    If Left(PNWbasinStrux, 1) = 1 Then
'       Seattle_Basin = 1
'    Else
'       Seattle_Basin = 0
'    End If
   period = getTFvalue(UserPeriod)
 '   Debug.Print User_Nsigma
  '    Debug.Print period
   If User_Nsigma <> 0 Then
  '      Debug.Print "iff"
      If Left(User_EpiInSigmaModels, 1) = 1 Then ' yes
'       Debug.Print "here"
         SigmaEpistemicKBCG20 = KBCG20_SigmaEpistemic(period, Magnitude, Rrup, AlphaBackarc, AlphaNankai, Ztor,
EventType, Vs30, Z1pt0, Z2pt5, MbKBCG20, UserRegion, User_NsampleEpi, PNWbasinStrux)
         SigmaEpistemicPSHAB20 = PSHAB20_SigmaEpistemic(period, EventType, SubductionSlab)
      Else    ' none
         SigmaEpistemicKBCG20 = 0
         SigmaEpistemicPSHAB20 = 0
      End If

'         Debug.Print "User_AleInSigmaModels:" & User_AleInSigmaModels
'         Debug.Print "Left(User_AleInSigmaModels, 1):" & Left(User_AleInSigmaModels, 1)

      If Left(User_AleInSigmaModels, 1) = 1 Then  ' all
 '         Debug.Print "all"
         SigmaAleatoryKBCG20 = KBCG20_sigmaAleatory(period)
         SigmaAleatoryPSHAB20 = PSHAB20_SigmaAleatory(period, Rrup, Vs30)
      Else    ' none
'         Debug.Print "none"
         SigmaAleatoryKBCG20 = 0
         SigmaAleatoryPSHAB20 = 0
      End If
```

```
      KBCG20Sigma = Sqr(SigmaAleatoryKBCG20 * SigmaAleatoryKBCG20 + SigmaEpistemicKBCG20 * SigmaEpistemicKBCG20)
      PSHAB20Sigma = Sqr(SigmaAleatoryPSHAB20 * SigmaAleatoryPSHAB20 + SigmaEpistemicPSHAB20 *
SigmaEpistemicPSHAB20)
   Else
      KBCG20Sigma = 0
      PSHAB20Sigma = 0
   End If

'   Debug.Print User_Nsigma & "   " & SigmaAleatoryKBCG20 & "   " & SigmaAleatoryKBCG20
'   Debug.Print KBCG20Sigma
'   Debug.Print PSHAB20Sigma
   WeightedSum = 0
   WeightSum = 0
   If Weight_KBCG20 > 0 Then
      KBCG20Median = KBCG20_medPSA(period, Magnitude, Rrup, AlphaBackarc, AlphaNankai, Ztor, EventType, Vs30, Z1pt0,
Z2pt5, MbKBCG20, UserRegion, PNWbasinStrux)
      KBCG20 = Exp(Log(KBCG20Median) + User_Nsigma * KBCG20Sigma)
      WeightSum = WeightSum + Weight_KBCG20
      If LCase(User_MeanType) = "arithmetic" Then
         WeightedSum = WeightedSum + KBCG20 * Weight_KBCG20
      Else
         WeightedSum = WeightedSum + Log(KBCG20) * Weight_KBCG20
      End If
   End If
   If Weight_PSHAB20 > 0 Then
      PSHAB20Median = PSHAB20_Median(EventType, UserRegion, SubductionSlab, Rrup, Magnitude, Zhypo, period, Vs30,
Z2pt5, PNWbasinStrux)
      PSHAB20 = Exp(Log(PSHAB20Median) + User_Nsigma * PSHAB20Sigma)
      WeightSum = WeightSum + Weight_PSHAB20
      If LCase(User_MeanType) = "arithmetic" Then
         WeightedSum = WeightedSum + PSHAB20 * Weight_PSHAB20
      Else
         WeightedSum = WeightedSum + Log(PSHAB20) * Weight_PSHAB20
      End If
   End If
   If WeightSum > 0 Then
      If LCase(User_MeanType) = "arithmetic" Then
         NGAsubGMM_Weighted = WeightedSum / WeightSum
      Else
         NGAsubGMM_Weighted = Exp(WeightedSum / WeightSum)
      End If
   End If

End Function
' I removed NGAsubGMM_SigmaTotal
Option Base 1

'#' Function to calculate median prediction

'The following function calculates the median prediction of KBCG20, as described in chapter 4 of the PEER report.
'It takes as input the predictor variables, as well as a set of coefficients.
'Later, we define a function that acts as a wrapper around this function, and will take as input period, region index, and then
select the appropriate coefficients and pass them on.
'
'The inputs are

' `m`: moment magnitude
```

```
' `rlist`: a vector of length 3, which contains the distance in subregion 1,2,3 relative to volcanic arc; `rlist = c(R1,R2,R3)`.
Typically, `R1=R3=0` (corresponding to forearc).
' `ztor`: depth to top of rupture in km.
' `EventType`: flag for interface (`EventType = 0`) and intraslab (`EventType = 1`). Must be 0 or 1.
' `Vs30`: $V_{S30}$ n m/s.
' `fx`: arc crossing flag. Must be 0 or 1.
' `delta_ln_z`: difference between natural log of observed Z1.0/Z2.5 value and reference Z1.0/Z2.5 from $V_{S30}$.
' `coeffs`: vector containing coefficients needed to calculate median prediction.
' `coeffs_attn`: vector of length 6 to calcualte anelastic attenuation.
' `mbreak` and `zbreak`: magnitude and depth scaling break point
' `k1` and `k2`: parameters needed for site amplification
' `nft1` and `nft2`: coefficients needed for pseudo-depth term.
' `pgarock`: median pga prediction at $V_{S30} = 1100$

' function to calculate median prediction

Function KBCG20_med(M, Rrup, AlphaBackarc, AlphaNankai, Ztor, EventType, Vs30, delta_ln_z, coeffs, coeffs_attn, coeffs_z,
mbreak, zbreak, k1, k2, nft1, nft2, pgarock, region)
    ' vba CODE DEVELOPED/IMPLEMENTED BY
  '        Silvia Mazzoni, 2020
  '         smazzoni@ucla.edu



 If Left(region, 1) = 1 Then
    distR1 = 0
    distR3 = 0
 ElseIf Left(region, 1) = 2 Then
    distR1 = 0
    distR3 = 0
 ElseIf Left(region, 1) = 3 Then
    distR1 = AlphaBackarc * Rrup
    distR3 = 0
 ElseIf Left(region, 1) = 4 Then
    distR1 = AlphaBackarc * Rrup
    distR3 = AlphaNankai * Rrup
 ElseIf Left(region, 1) = 5 Then
    distR1 = 0
    distR3 = 0
 ElseIf Left(region, 1) = 6 Then
    distR1 = AlphaBackarc * Rrup
    distR3 = 0
 ElseIf Left(region, 1) = 7 Then
    distR1 = 0
    distR3 = 0
 Else  ' reg=0
    distR1 = AlphaBackarc * Rrup
    distR3 = 0
 End If

  distR2 = Rrup - distR1 - distR3



 theta10 = 0
 vsrock = 1100
 c = 1.88
 n = 1.18
```

```
  minmb = 6#
  delta = 0.1
  deltaz = 1
  refzif = 15
  refzslab = 50

  thisFs = val(Left(EventType, 1))

    'check if cross arc:
       fx = 1
       If distR1 = 0 And distR2 = 0 Then
          fx = 0
       End If
       If distR2 = 0 And distR3 = 0 Then
          fx = 0
       End If
       If distR1 = 0 And distR3 = 0 Then
          fx = 0
       End If

  fmag = (1 - thisFs) * loghinge(M, mbreak, coeffs(6) * (mbreak - minmb), coeffs(6), coeffs(8), delta) + thisFs * loghinge(M,
mbreak, coeffs(7) * (mbreak - minmb), coeffs(7), coeffs(8), delta)

  fgeom = (1 - thisFs) * (coeffs(3) + coeffs(5) * M) * Log(Rrup + 10 ^ (nft1 + nft2 * (M - 6)))
  fgeom_slab = thisFs * (coeffs(4) + coeffs(5) * M) * Log(Rrup + 10 ^ (nft1 + nft2 * (M - 6)))
  fdepth = (1 - thisFs) * loghinge(Ztor, zbreak, coeffs(12) * (zbreak - refzif), coeffs(12), theta10, deltaz) + thisFs *
loghinge(Ztor, zbreak, coeffs(13) * (zbreak - refzslab), coeffs(13), theta10, deltaz)


    DotProduct123 = distR1 * coeffs_attn(1) + distR2 * coeffs_attn(2) + distR3 * coeffs_attn(3)
    DotProduct456 = distR1 * coeffs_attn(4) + distR2 * coeffs_attn(5) + distR3 * coeffs_attn(6)

  fattn = fx * DotProduct123 + (1 - fx) * DotProduct456 + fx * coeffs(14)




  If Vs30 < k1 Then
    fsite = coeffs(11) * Log(Vs30 / k1) + k2 * (Log(pgarock + c * (Vs30 / k1) ^ n) - Log(pgarock + c))
  Else
    fsite = (coeffs(11) + k2 * n) * Log(Vs30 / k1)
  End If




  fbasin = coeffs_z(1) + coeffs_z(2) * delta_ln_z



  Median = (1 - thisFs) * coeffs(1) + thisFs * coeffs(2) + fmag + fgeom + fgeom_slab + fdepth + fattn + fsite + fbasin
  KBCG20_med = Median


End Function
```

```
Option Base 1

'#' Function to calculate median prediction for a given scenario and period
'This is a function that calculates median predictions of KBCG20 for a given scenario.
'I takes as input period, the predictor variables for the scenarios, selects the appropriate coefficients/parameters, and calls the function defined in the previous section.
'
'The arguments are similar to the function before.

' `m`: moment magnitude
' `rlist`: a vector of length 3, which contains the distances in subregion 1,2,3 relative to volcanic arc; `rlist = array(R1,R2,R3)`. For regions Alaska, Cascadia, New Zealand, and Taiwan, $R1 = R2 = 0$.
' `ztor`: depth to top of rupture in km.
' `EventType`: flag for interface (`EventType = 0`) and intraslab (`EventType = 1`). Must be 0 or 1.
' `Vs30`: $V_{S30}$ n m/s.
' `Z1pt0`: depth to a shear wave horizon of 1000 m/s, in m. Used for regions Alaska and New Zealand.
' `Z2pt5`: depth to a shear wave horizon of 2500 m/s, in m. Used for regions Cascadia and Japan.
' `fx`: arc crossing flag. Must be 0 or 1.
' `mb`: Magnitude scaling break point. Should be set regionally dependent based on Campbell (2020).

'The last input is a region index `region`, which is as follows:
'
'* 0: global
'* 1: Alaska
'* 2: Cascadia
'* 3: Central America & Mexico
'* 4: Japan
'* 5: New Zealand
'* 6: South America
'* 7: Taiwan
'
' this no longer holds true, we have switched to a 3-value switch
'The opional argumen `Seattle_Basin` is a flag that should be set to `TRUE` if the site is in the Seattle Basin, and `FALSE` otherwise.
'This flag determines which basin depth amplification model is used for Cascadia.
'It does not have an impact on any other region.
'



Function KBCG20_medPSA(period, Magnitude, Rrup, AlphaBackarc, AlphaNankai, Ztor, EventType, Vs30, Z1pt0, Z2pt5, Mb, UserRegion, PNWbasinStrux)
    ' vba CODE DEVELOPED/IMPLEMENTED BY
    '         Silvia Mazzoni, 2020
    '          smazzoni@ucla.edu

    thisOut = -999
    XinterpMin = 0
    XinterpMax = 10
    XinterpType = "log"
    YinterpType = "log"
    extrapolateType = "extrapolate"

    TvalueList = Range("TvalueList_KBCG19")
    InterpArray = interpolateFunction(period, TvalueList, XinterpType, extrapolateType, XinterpMin, XinterpMax)

    period0 = TvalueList(InterpArray(1), 1)
    y0 = KBCG20_medPSA_AtTlist(period0, Magnitude, Rrup, AlphaBackarc, AlphaNankai, Ztor, EventType, Vs30, Z1pt0, Z2pt5, Mb, UserRegion, PNWbasinStrux)
```

```
   If InterpArray(2) <= 0 Then
      thisOut = y0
   Else
      period1 = TvalueList(InterpArray(2), 1)
      y1 = KBCG20_medPSA_AtTlist(period1, Magnitude, Rrup, AlphaBackarc, AlphaNankai, Ztor, EventType, Vs30, Z1pt0,
Z2pt5, Mb, UserRegion, PNWbasinStrux)
      If LCase(Left(YinterpType, Len("log"))) = LCase("log") Then
         If y0 <= 0 Then
            y0 = 0.000000001
         End If
         If y1 <= 0 Then
            y1 = 0.000000001
         End If
         y0 = Log(y0)
         y1 = Log(y1)
      End If

      thisOut = y0 + (y1 - y0) * InterpArray(3)
      If LCase(Left(YinterpType, Len("log"))) = LCase("log") Then
         thisOut = Exp(thisOut)
      End If

   End If

   KBCG20_medPSA = thisOut

End Function

' function to calculate median prediction using mean coefficients
Function KBCG20_medPSA_AtTlist(period, M, Rrup, AlphaBackarc, AlphaNankai, Ztor, EventType, Vs30, Z1pt0, Z2pt5, Mb,
region, Optional PNWbasinStrux = 0)
   ' vba CODE DEVELOPED/IMPLEMENTED BY
   '        Silvia Mazzoni, 2020
   '         smazzoni@ucla.edu
'  ' ' ' Debug.print "KBCG20_medPSA_AtTlist"
   KBCG20_medPSA_AtTlist = -999
  ' need to add some checks for input (period, region)
  ' coefficients to calculate zref from vs30
  pars_z_ja = Array(7.6893685375, 2.30258509299405, 6.3091864, 0.7528670225, 1.2952369625)
  pars_z_casc = Array(8.29404964010203, 2.30258509299405, 6.39692965521615, 0.27081459, 1.7381352625)
  pars_z_nz = Array(6.859789675, 2.30258509299405, 5.745692775, 0.91563524375, 1.03531412375)
  pars_z_tw = Array(6.30560665, 2.30258509299405, 6.1104992125, 0.43671102, 0.7229702975)
 ' parameters =
read.csv('/Users/nico/GROUNDMOTION/PROJECTS/SUBDUCTION/CBGK/STAN_V16/RESULTS/parameters_allregca_attn3_co
rrreg_cs_dmb_smoothed_gprf.csv')
 ' parameters_zmod =
read.csv('/Users/nico/GROUNDMOTION/PROJECTS/SUBDUCTION/CBGK/STAN_V16/RESULTS/parameters_Z_ALL_allregca_at
tn3_corrreg_cs_dmb.csv')

  Parameters = Range("coefficients_KBCG19")
  parameters_zmod = Range("parameters_Z_ALL_allregca_attn3_corrreg_cs_dmb")

  ' calculate rock PGA
  period_used = 0#
  vsrock = 1100
  pars_period = getRangeRow(Parameters, period_used, "T")
  pars_period_zmod = getRangeRow(parameters_zmod, period_used, "T")  ' silviamazzoni: I added this
```

```
coeffs = getSubArrayRange(pars_period, 2, 15)


k1k2 = interp_k1k2(period_used)


delta_ln_z = 0
coeffs_z = Array(0, 0)
coeffs_z2 = Array(0, 0)


dmb = 0

If Left(region, 1) = 1 Then
  coeffs = updateArray(coeffs, Array(1, 2, 11), pars_period, Array(17, 24, 31))
  coeffs_attn = makeArray(pars_period, Array(38, 45, 52, 59, 66, 73))
 ElseIf Left(region, 1) = 2 Then
  coeffs = updateArray(coeffs, Array(1, 2, 11), pars_period, Array(18, 25, 32))
  coeffs_attn = makeArray(pars_period, Array(39, 46, 53, 60, 67, 74))
  ' coeffs_z2 = makeArray(pars_period_zmod, Array(2, 3))
  If val(Left(PNWbasinStrux, 1)) = 1 Then ' Seattle Basin
     coeff_seattle = getRangeValue(parameters_zmod, period_used, "T", "mean_residual_Seattle_basin")
     coeffs_z2 = Array(coeff_seattle, 0)
  ElseIf val(Left(PNWbasinStrux, 1)) = 2 Then ' Other Basin
     coeffs_z2 = makeArray(pars_period_zmod, Array(2, 3))
  End If
 ElseIf Left(region, 1) = 3 Then
  coeffs = updateArray(coeffs, Array(1, 2, 11), pars_period, Array(19, 26, 33))
  coeffs_attn = makeArray(pars_period, Array(40, 47, 54, 61, 68, 75))
 ElseIf Left(region, 1) = 4 Then
  dmb = interp_dmb(period_used)
  coeffs = updateArray(coeffs, Array(1, 2, 11), pars_period, Array(20, 27, 34))
  coeffs_attn = makeArray(pars_period, Array(41, 48, 55, 62, 69, 76))
  coeffs_z2 = makeArray(pars_period_zmod, Array(5, 6))
 ElseIf Left(region, 1) = 5 Then
  coeffs = updateArray(coeffs, Array(1, 2, 11), pars_period, Array(21, 28, 35))
  coeffs_attn = makeArray(pars_period, Array(42, 49, 56, 63, 70, 77))
  coeffs_z2 = makeArray(pars_period_zmod, Array(7, 8))
 ElseIf Left(region, 1) = 6 Then
  dmb = interp_dmb(period_used)
  coeffs = updateArray(coeffs, Array(1, 2, 11), pars_period, Array(22, 29, 36))
  coeffs_attn = makeArray(pars_period, Array(43, 50, 57, 64, 71, 78))
 ElseIf Left(region, 1) = 7 Then
  coeffs = updateArray(coeffs, Array(1, 2, 11), pars_period, Array(23, 30, 37))
  coeffs_attn = makeArray(pars_period, Array(44, 51, 58, 65, 72, 79))
  coeffs_z2 = makeArray(pars_period_zmod, Array(9, 10))
 Else  ' left(region,1)=0
  coeffs_attn = makeArray(pars_period, Array(11, 11, 11, 11, 10, 11))
  delta_ln_z = 0
  coeffs_z = Array(0, 0)
 End If
```

```
 delta_bz = makeArray(pars_period, Array(80, 81))
 coeffs_nft = makeArray(pars_period, Array(82, 83))

 thisFs = val(Left(EventType, 1))
 mbreak = (1 - thisFs) * (Mb + dmb) + thisFs * Mb
 zbreak = (1 - thisFs) * (30 + delta_bz(1)) + thisFs * (80 + delta_bz(2))

 mbreak_pga = mbreak
 zbreak_pga = zbreak
 k1k2_pga = k1k2
 coeffs_pga = coeffs   ' silviamazzoni: I added this so I alway have coeffs
 coeffs_attn_pga = coeffs_attn
 coeffs_z_pga = coeffs_z
 coeffs_z_pga2 = coeffs_z2
 coeffs_nft_pga = coeffs_nft
 pgarock = Exp(KBCG20_med(M, Rrup, AlphaBackarc, AlphaNankai, Ztor, EventType, vsrock, delta_ln_z, coeffs_pga,
coeffs_attn_pga, coeffs_z_pga, mbreak_pga, zbreak_pga, k1k2_pga(1), k1k2_pga(2), coeffs_nft_pga(1), coeffs_nft_pga(2), 0,
region))

 ' calculate PSA
 period_used = period
 pars_period = getRangeRow(Parameters, period_used, "T")
 pars_period_zmod = getRangeRow(parameters_zmod, period_used, "T") ' silviamazzoni: I added this
 coeffs = getSubArrayRange(pars_period, 2, 15)
 k1k2 = interp_k1k2(period_used)


 dmb = 0

  delta_ln_z = 0
  coeffs_z = Array(0, 0)
 If Left(region, 1) = 1 Then
  coeffs = updateArray(coeffs, Array(1, 2, 11), pars_period, Array(17, 24, 31))
  coeffs_attn = makeArray(pars_period, Array(38, 45, 52, 59, 66, 73))
 ElseIf Left(region, 1) = 2 Then
  coeffs = updateArray(coeffs, Array(1, 2, 11), pars_period, Array(18, 25, 32))
  coeffs_attn = makeArray(pars_period, Array(39, 46, 53, 60, 67, 74))
  If Z2pt5 = "default" Or Z2pt5 = "Default" Or Z2pt5 <= 0 Then
   delta_ln_z = 0
  Else
     delta_ln_z = Log(Z2pt5) - calc_z_from_Vs30(Vs30, pars_z_casc)
  End If
  ' coeffs_z = makeArray(pars_period_zmod, Array(2, 3))
  If val(Left(PNWbasinStrux, 1)) = 1 Then    ' Seattle Basin
     coeff_seattle = getRangeValue(parameters_zmod, period_used, "T", "mean_residual_Seattle_basin")
     coeffs_z = Array(coeff_seattle, 0)
  ElseIf val(Left(PNWbasinStrux, 1)) = 2 Then      ' Other Basin = 2
     coeffs_z = makeArray(pars_period_zmod, Array(2, 3))
  End If
 ElseIf Left(region, 1) = 3 Then
  coeffs = updateArray(coeffs, Array(1, 2, 11), pars_period, Array(19, 26, 33))
  coeffs_attn = makeArray(pars_period, Array(40, 47, 54, 61, 68, 75))
 ElseIf Left(region, 1) = 4 Then
  dmb = interp_dmb(period_used)
  coeffs = updateArray(coeffs, Array(1, 2, 11), pars_period, Array(20, 27, 34))
  coeffs_attn = makeArray(pars_period, Array(41, 48, 55, 62, 69, 76))
```

```
   If Z2pt5 = "default" Or Z2pt5 = "Default" Or Z2pt5 <= 0 Then
     delta_ln_z = 0
   Else
     delta_ln_z = Log(Z2pt5) - calc_z_from_Vs30(Vs30, pars_z_ja)
   End If
   coeffs_z = makeArray(pars_period_zmod, Array(5, 6))
  ElseIf Left(region, 1) = 5 Then
   coeffs = updateArray(coeffs, Array(1, 2, 11), pars_period, Array(21, 28, 35))
   coeffs_attn = makeArray(pars_period, Array(42, 49, 56, 63, 70, 77))
   If Z1pt0 = "default" Or Z1pt0 = "Default" Or Z1pt0 <= 0 Then
     delta_ln_z = 0
   Else
     delta_ln_z = Log(Z1pt0) - calc_z_from_Vs30(Vs30, pars_z_nz)
   End If
   coeffs_z = makeArray(pars_period_zmod, Array(7, 8))
  ElseIf Left(region, 1) = 6 Then
   dmb = interp_dmb(period_used)
   coeffs = updateArray(coeffs, Array(1, 2, 11), pars_period, Array(22, 29, 36))
   coeffs_attn = makeArray(pars_period, Array(43, 50, 57, 64, 71, 78))
  ElseIf Left(region, 1) = 7 Then
   coeffs = updateArray(coeffs, Array(1, 2, 11), pars_period, Array(23, 30, 37))
   coeffs_attn = makeArray(pars_period, Array(44, 51, 58, 65, 72, 79))
   If Z1pt0 = "default" Or Z1pt0 = "Default" Or Z1pt0 <= 0 Then
     delta_ln_z = 0
   Else
     delta_ln_z = Log(Z1pt0) - calc_z_from_Vs30(Vs30, pars_z_tw)
   End If
   coeffs_z = makeArray(pars_period_zmod, Array(9, 10))
  Else  ' reg=0
   coeffs_attn = makeArray(pars_period, Array(11, 11, 11, 11, 10, 11))
  End If

  delta_bz = makeArray(pars_period, Array(80, 81))
  coeffs_nft = makeArray(pars_period, Array(82, 83))

  thisFs = val(Left(EventType, 1))
  mbreak = (1 - thisFs) * (Mb + dmb) + thisFs * Mb
  zbreak = (1 - thisFs) * (30 + delta_bz(1)) + thisFs * (80 + delta_bz(2))


  Med = KBCG20_med(M, Rrup, AlphaBackarc, AlphaNankai, Ztor, EventType, Vs30, delta_ln_z, coeffs, coeffs_attn, coeffs_z,
mbreak, zbreak, k1k2(1), k1k2(2), coeffs_nft(1), coeffs_nft(2), pgarock, region)
  med_pga = KBCG20_med(M, Rrup, AlphaBackarc, AlphaNankai, Ztor, EventType, Vs30, delta_ln_z, coeffs_pga,
coeffs_attn_pga, coeffs_z_pga2, mbreak_pga, zbreak_pga, k1k2_pga(1), k1k2_pga(2), coeffs_nft_pga(1), coeffs_nft_pga(2),
pgarock, region)

  If Med < med_pga And period <= 0.1 Then
    Med = med_pga
  End If

  KBCG20_medPSA_AtTlist = Exp(Med)


End Function

Option Base 1

'#' Function to calculate distribution of median predictions (for epistemic uncertainty)
```

```
'
'This function is similar as the one in the previous section, but reads in posterior distribution, and loops over samples from posterior.
'The number of samples can be set, with 800 as default.
'Output is mean/median/5%,25%,75%,95% quntiles of distribution, and the median predictions themselves.
'
'Two functions are defined.
'The first calculates predictions for the global model using presampled adjustment coefficients for a new region, while the second one samples these internally
'The second function is implemented to demonstrate how to perform the sampling, (how to calculate the covariance matrix from the `output of the cholesky decomposition).
'To get reproducible results, one should use the function with presampled global coefficients.
'
'
' function to calculate median prediction using posterior distributions
' still need to add basin depth term

' function to calculate median prediction using mean coefficients

Function KBCG20_posterior(period, M, Rrup, AlphaBackarc, AlphaNankai, Ztor, EventType, Vs30, Z1pt0, Z2pt5, Mb, region, Optional num_samples = 100, Optional PNWbasinStrux = 0)
    ' vba CODE DEVELOPED/IMPLEMENTED BY
    '         Silvia Mazzoni, 2020
    '          smazzoni@ucla.edu

'    ' ' ' Debug.print "KBCG20_posterior"
    thisOut = -999
    XinterpMin = 0
    XinterpMax = 10
    XinterpType = "log"
    YinterpType = "linear"  ' KBCG20_posteriorAtTlist is already log
    extrapolateType = "extrapolate"

    TvalueList = Range("TvalueList_KBCG19")
    InterpArray = interpolateFunction(period, TvalueList, XinterpType, extrapolateType, XinterpMin, XinterpMax)

    period0 = TvalueList(InterpArray(1), 1)
    KBCG20_posteriorAtTlist0 = KBCG20_posteriorAtTlist(period0, M, Rrup, AlphaBackarc, AlphaNankai, Ztor, EventType, Vs30, Z1pt0, Z2pt5, Mb, region, num_samples, PNWbasinStrux)
    y0 = InterpArray(1)

    If InterpArray(2) <= 0 Then
        KBCG20_posterior = KBCG20_posteriorAtTlist0
        Exit Function
    Else
        period1 = TvalueList(InterpArray(2), 1)
        KBCG20_posteriorAtTlist1 = KBCG20_posteriorAtTlist(period1, M, Rrup, AlphaBackarc, AlphaNankai, Ztor, EventType, Vs30, Z1pt0, Z2pt5, Mb, region, num_samples, PNWbasinStrux)

        Dim med_predictions()
        ReDim med_predictions(UBound(KBCG20_posteriorAtTlist0))

        For iCase = LBound(KBCG20_posteriorAtTlist0) To UBound(KBCG20_posteriorAtTlist0)
            y0 = KBCG20_posteriorAtTlist0(iCase)
            y1 = KBCG20_posteriorAtTlist1(iCase)

            If LCase(Left(YinterpType, Len("log"))) = LCase("log") Then
                If y0 <= 0 Then
```

```
            y0 = 0.000000001
          End If
          If y1 <= 0 Then
            y1 = 0.000000001
          End If
          y0 = Log(y0)
          y1 = Log(y1)
        End If
        med_predictions(iCase) = y0 + (y1 - y0) * InterpArray(3)
        If LCase(Left(YinterpType, Len("log"))) = LCase("log") Then
          med_predictions(iCase) = Exp(med_predictions(iCase))
        End If
      Next iCase

   End If

   KBCG20_posterior = med_predictions

End Function



Function KBCG20_posteriorAtTlist(period, M, Rrup, AlphaBackarc, AlphaNankai, Ztor, EventType, Vs30, Z1pt0, Z2pt5, Mb,
region, Optional num_samples = 100, Optional PNWbasinStrux = 0)
   ' vba CODE DEVELOPED/IMPLEMENTED BY
   '        Silvia Mazzoni, 2020
   '        smazzoni@ucla.edu
  ' need to add some checks for input (period, region)
  ' ' ' ' Debug.print "KBCG20_posteriorAtTlist"

   KBCG20_posteriorAtTlist = -999
  ' coefficients to calculate zref from vs30
  pars_z_ja = Array(7.6893685375, 2.30258509299405, 6.3091864, 0.7528670225, 1.2952369625)
  pars_z_casc = Array(8.29404964010203, 2.30258509299405, 6.39692965521615, 0.27081459, 1.7381352625)
  pars_z_nz = Array(6.859789675, 2.30258509299405, 5.745692775, 0.91563524375, 1.03531412375)
  pars_z_tw = Array(6.30560665, 2.30258509299405, 6.1104992125, 0.43671102, 0.7229702975)

'  parameters =
read.csv('/Users/nico/GROUNDMOTION/PROJECTS/SUBDUCTION/CBGK/STAN_V16/RESULTS/parameters_allregca_attn3_co
rrreg_cs_dmb_smoothed_gprf.csv')
'  parameters_zmod =
read.csv('/Users/nico/GROUNDMOTION/PROJECTS/SUBDUCTION/CBGK/STAN_V16/RESULTS/parameters_Z_ALL_allregca_at
tn3_corrreg_cs_dmb.csv')

'SilviaMazzoni: what tables do i use??
  Parameters = Range("coefficients_KBCG19")
  parameters_zmod = Range("parameters_Z_ALL_allregca_attn3_corrreg_cs_dmb")


  ' calculate rock PGA
  thisPeriod = 0#
  vsrock = 1100
  pars_period = getRangeRow(Parameters, thisPeriod, "T")
  pars_period_zmod = getRangeRow(parameters_zmod, thisPeriod, "T")  ' silviamazzoni, I added this
  coeffs = getSubArrayRange(pars_period, 2, 15)
  k1k2 = interp_k1k2(thisPeriod)
  dmb = 0
```

```
   delta_ln_z = 0
   coeffs_z = Array(0, 0)
  If Left(region, 1) = 1 Then
   coeffs = updateArray(coeffs, Array(1, 2, 11), pars_period, Array(17, 24, 31))
   coeffs_attn = makeArray(pars_period, Array(38, 45, 52, 59, 66, 73))
  ElseIf Left(region, 1) = 2 Then
   coeffs = updateArray(coeffs, Array(1, 2, 11), pars_period, Array(18, 25, 32))
   coeffs_attn = makeArray(pars_period, Array(39, 46, 53, 60, 67, 74))
  ElseIf Left(region, 1) = 3 Then
   coeffs = updateArray(coeffs, Array(1, 2, 11), pars_period, Array(19, 26, 33))
   coeffs_attn = makeArray(pars_period, Array(40, 47, 54, 61, 68, 75))
  ElseIf Left(region, 1) = 4 Then
   dmb = interp_dmb(thisPeriod)
   coeffs = updateArray(coeffs, Array(1, 2, 11), pars_period, Array(20, 27, 34))
   coeffs_attn = makeArray(pars_period, Array(41, 48, 55, 62, 69, 76))
  ElseIf Left(region, 1) = 5 Then
   coeffs = updateArray(coeffs, Array(1, 2, 11), pars_period, Array(21, 28, 35))
   coeffs_attn = makeArray(pars_period, Array(42, 49, 56, 63, 70, 77))
  ElseIf Left(region, 1) = 6 Then
   dmb = interp_dmb(thisPeriod)
   coeffs = updateArray(coeffs, Array(1, 2, 11), pars_period, Array(22, 29, 36))
   coeffs_attn = makeArray(pars_period, Array(43, 50, 57, 64, 71, 78))
  ElseIf Left(region, 1) = 7 Then
   coeffs = updateArray(coeffs, Array(1, 2, 11), pars_period, Array(23, 30, 37))
   coeffs_attn = makeArray(pars_period, Array(44, 51, 58, 65, 72, 79))
  Else  ' reg=0
   coeffs_attn = makeArray(pars_period, Array(11, 11, 11, 11, 10, 11))
  End If


  delta_bz = makeArray(pars_period, Array(80, 81))
  coeffs_nft = makeArray(pars_period, Array(82, 83))
  thisFs = val(Left(EventType, 1))
  mbreak = (1 - thisFs) * (Mb + dmb) + thisFs * Mb
  zbreak = (1 - thisFs) * (30 + delta_bz(1)) + thisFs * (80 + delta_bz(2))
   pgarock = Exp(KBCG20_med(M, Rrup, AlphaBackarc, AlphaNankai, Ztor, EventType, vsrock, delta_ln_z, coeffs, coeffs_attn,
coeffs_z, mbreak, zbreak, k1k2(1), k1k2(2), coeffs_nft(1), coeffs_nft(2), 0, region))


  ' calculate PSA
  thisPeriod = period
  'parameters_posterior =
read.csv(sprintf('/Users/nico/GROUNDMOTION/PROJECTS/SUBDUCTION/CBGK/STAN_V16/RESULTS/CSV_allregca_attn3_co
rrreg_cs_dmb_smoothed_gprf/coefficients_KBCG20_global_smoothed_gprf_T%05.2f.csv',period))
   KBCG20_PeriodMapData = Range("KBCG19_PeriodMapData")
   Tstring = getRangeValue(KBCG20_PeriodMapData, thisPeriod, "Tnumeric", "Tstring")
   parameters_posterior = Range("posterior_coefficients_KBCG19_" & Tstring)

  pars_period_zmod = getRangeRow(parameters_zmod, thisPeriod, "T")  ' silviamazzoni, I added this



  coeffs = getSubArrayRange(pars_period, 2, 15)
  k1k2 = interp_k1k2(thisPeriod)
  dmb = 0
```

```
Dim med_predictions()
ReDim med_predictions(num_samples)

For k = 1 To num_samples

   pars_period = getRangeRowByIndex(parameters_posterior, k + 1) ' silviamazzoni: +1 because of header row
   coeffs = getSubArrayRange(pars_period, 2, 15)

    delta_ln_z = 0
    coeffs_z = Array(0, 0)
   If Left(region, 1) = 1 Then
    coeffs = updateArray(coeffs, Array(1, 2, 11), pars_period, Array(17, 24, 31))
    coeffs_attn = makeArray(pars_period, Array(38, 45, 52, 59, 66, 73))
   ElseIf Left(region, 1) = 2 Then
    coeffs = updateArray(coeffs, Array(1, 2, 11), pars_period, Array(18, 25, 32))
    coeffs_attn = makeArray(pars_period, Array(39, 46, 53, 60, 67, 74))
    If Z2pt5 = "default" Or Z2pt5 = "Default" Or Z2pt5 <= 0 Then
      delta_ln_z = 0
    Else
      delta_ln_z = Log(Z2pt5) - calc_z_from_Vs30(Vs30, pars_z_casc)
    End If
    ' coeffs_z = makeArray(pars_period_zmod, Array(2, 3))
    If val(Left(PNWbasinStrux, 1)) = 1 Then       ' Seattle Basin = 1
       coeff_seattle = getRangeValue(parameters_zmod, thisPeriod, "T", "mean_residual_Seattle_basin")
       coeffs_z = Array(coeff_seattle, 0)
    ElseIf val(Left(PNWbasinStrux, 1)) = 2 Then       ' Other Basin = 2
      coeffs_z = makeArray(pars_period_zmod, Array(2, 3))
    End If
   ElseIf Left(region, 1) = 3 Then
    coeffs = updateArray(coeffs, Array(1, 2, 11), pars_period, Array(19, 26, 33))
    coeffs_attn = makeArray(pars_period, Array(40, 47, 54, 61, 68, 75))
   ElseIf Left(region, 1) = 4 Then
    dmb = interp_dmb(thisPeriod)
    coeffs = updateArray(coeffs, Array(1, 2, 11), pars_period, Array(20, 27, 34))
    coeffs_attn = makeArray(pars_period, Array(41, 48, 55, 62, 69, 76))
    If Z2pt5 = "default" Or Z2pt5 = "Default" Or Z2pt5 <= 0 Then
      delta_ln_z = 0
    Else
      delta_ln_z = Log(Z2pt5) - calc_z_from_Vs30(Vs30, pars_z_ja)
    End If
    coeffs_z = makeArray(pars_period_zmod, Array(5, 6))
   ElseIf Left(region, 1) = 5 Then
    coeffs = updateArray(coeffs, Array(1, 2, 11), pars_period, Array(21, 28, 35))
    coeffs_attn = makeArray(pars_period, Array(42, 49, 56, 63, 70, 77))
    If Z1pt0 = "default" Or Z1pt0 = "Default" Or Z1pt0 <= 0 Then
      delta_ln_z = 0
    Else
      delta_ln_z = Log(Z1pt0) - calc_z_from_Vs30(Vs30, pars_z_nz)
    End If
    coeffs_z = makeArray(pars_period_zmod, Array(7, 8))
   ElseIf Left(region, 1) = 6 Then
    dmb = interp_dmb(thisPeriod)
    coeffs = updateArray(coeffs, Array(1, 2, 11), pars_period, Array(22, 29, 36))
    coeffs_attn = makeArray(pars_period, Array(43, 50, 57, 64, 71, 78))
   ElseIf Left(region, 1) = 7 Then
    coeffs = updateArray(coeffs, Array(1, 2, 11), pars_period, Array(23, 30, 37))
    coeffs_attn = makeArray(pars_period, Array(44, 51, 58, 65, 72, 79))
```

```
        If Z1pt0 = "default" Or Z1pt0 = "Default" Or Z1pt0 <= 0 Then
          delta_ln_z = 0
        Else
          delta_ln_z = Log(Z1pt0) - calc_z_from_Vs30(Vs30, pars_z_tw)
        End If
        coeffs_z = makeArray(pars_period_zmod, Array(9, 10))
       Else  ' reg=0
        coeffs = updateArray(coeffs, Array(1, 2, 11), pars_period, Array(172, 173, 174))
        coeffs_attn = makeArray(pars_period, Array(175, 176, 177, 178, 179, 180))
      End If



      delta_bz = makeArray(pars_period, Array(80, 81))
      coeffs_nft = makeArray(pars_period, Array(82, 83))
      thisFs = val(Left(EventType, 1))

      mbreak = (1 - thisFs) * (Mb + dmb) + thisFs * Mb
      zbreak = (1 - thisFs) * (30 + delta_bz(1)) + thisFs * (80 + delta_bz(2))

      Med = KBCG20_med(M, Rrup, AlphaBackarc, AlphaNankai, Ztor, EventType, Vs30, delta_ln_z, coeffs, coeffs_attn,
coeffs_z, mbreak, zbreak, k1k2(1), k1k2(2), coeffs_nft(1), coeffs_nft(2), pgarock, region)
      med_predictions(k) = Med
   Next k

   Dim KBCG20_posteriorAtTlist_All()
   this = UBound(med_predictions) + 3
   ReDim KBCG20_posteriorAtTlist_All(this)
   KBCG20_posteriorAtTlist_All(1) = getMeanOfArray(med_predictions)
   KBCG20_posteriorAtTlist_All(2) = getMedianOfArray(med_predictions)
   KBCG20_posteriorAtTlist_All(3) = getStdDev(med_predictions)
   For i = LBound(med_predictions) To UBound(med_predictions)
      KBCG20_posteriorAtTlist_All(3 + i + (1 - LBound(med_predictions))) = med_predictions(i)
   Next i
   KBCG20_posteriorAtTlist = KBCG20_posteriorAtTlist_All
End Function


' function to calculate aliatory sigma

Option Base 1

Function KBCG20_sigmaAleatory(period)
   ' VBA CODE DEVELOPED/IMPLEMENTED BY
   '        Silvia Mazzoni, 2020
   '         smazzoni@ucla.edu

   thisOut = -999
   XinterpMin = 0
   XinterpMax = 10
   XinterpType = "log"
   YinterpType = "linear"
   extrapolateType = "extrapolate"

   TvalueList = Range("TvalueList_KBCG19")
   InterpArray = interpolateFunction(period, TvalueList, XinterpType, extrapolateType, XinterpMin, XinterpMax)

   period0 = TvalueList(InterpArray(1), 1)
```

```
    y0 = KBCG20_sigmaAleatoryAtTlist(period0)

    If InterpArray(2) <= 0 Then
        thisOut = y0
    Else
        period1 = TvalueList(InterpArray(2), 1)
        y1 = KBCG20_sigmaAleatoryAtTlist(period1)
        If LCase(Left(YinterpType, Len("log"))) = LCase("log") Then
            If y0 <= 0 Then
                y0 = 0.000000001
            End If
            If y1 <= 0 Then
                y1 = 0.000000001
            End If
            y0 = Log(y0)
            y1 = Log(y1)
        End If
        thisOut = y0 + (y1 - y0) * InterpArray(3)
        If LCase(Left(YinterpType, Len("log"))) = LCase("log") Then
            thisOut = Exp(thisOut)
        End If

    End If

    KBCG20_sigmaAleatory = thisOut

End Function


Function KBCG20_sigmaAleatoryAtTlist(period)
    ' VBA CODE DEVELOPED/IMPLEMENTED BY
    '       Silvia Mazzoni, 2020
    '        smazzoni@ucla.edu

  Parameters = Range("coefficients_KBCG19")

  ' calculate PSA
  period_used = period
  pars_period = getRangeRow(Parameters, period_used, "T")

  Phi = pars_period(84)
  Tau = pars_period(85)
  KBCG20_sigmaAleatoryAtTlist = Sqr(Phi * Phi + Tau * Tau)

End Function

Function KBCG20_sigmaPhi(period)
    ' VBA CODE DEVELOPED/IMPLEMENTED BY
    '       Silvia Mazzoni, 2020
    '        smazzoni@ucla.edu

  Parameters = Range("coefficients_KBCG19")

  ' calculate PSA
  period_used = period
  pars_period = getRangeRow(Parameters, period_used, "T")

  Phi = pars_period(84)
```

```
  KBCG20_sigmaPhi = Phi

End Function

Function KBCG20_sigmaTau(period)
     ' VBA CODE DEVELOPED/IMPLEMENTED BY
   '         Silvia Mazzoni, 2020
   '          smazzoni@ucla.edu

  Parameters = Range("coefficients_KBCG19")

  ' calculate PSA
  period_used = period
  pars_period = getRangeRow(Parameters, period_used, "T")

  Tau = pars_period(85)
  KBCG20_sigmaTau = Tau

End Function



Function KBCG20_SigmaEpistemic(period, M, Rrup, AlphaBackarc, AlphaNankai, Ztor, EventType, Vs30, Z1pt0, Z2pt5, Mb,
region, Optional num_samples = 100, Optional PNWbasinStrux = 0)
   ' VBA CODE DEVELOPED/IMPLEMENTED BY
   '         Silvia Mazzoni, 2020
   '          smazzoni@ucla.edu

 '  ' ' ' Debug.print "KBCG20_SigmaEpistemic"
   If num_samples = 0 Then
      KBCG20_SigmaEpistemic = 0
   ElseIf num_samples < 50 Then
      KBCG20_SigmaEpistemic = "not enough samples, please select a value =0 or between 50 and 800"
   ElseIf num_samples <= 800 Then

      thisKBCG20_posterior = KBCG20_posterior(period, M, Rrup, AlphaBackarc, AlphaNankai, Ztor, EventType, Vs30, Z1pt0,
Z2pt5, Mb, region, num_samples, PNWbasinStrux)
      KBCG20_SigmaEpistemic = thisKBCG20_posterior(3)
   Else
      KBCG20_SigmaEpistemic = "Please enter a value =0 or between 50 and 800"
   End If


End Function

Function KBCG20_SigmaTotal(period, M, Rrup, AlphaBackarc, AlphaNankai, Ztor, EventType, Vs30, Z1pt0, Z2pt5, Mb, region,
Optional num_samples = 100, Optional PNWbasinStrux = 0)
   ' VBA CODE DEVELOPED/IMPLEMENTED BY
   '         Silvia Mazzoni, 2020
   '          smazzoni@ucla.edu

   SigmaAleatory = KBCG20_sigmaAleatory(period)
   If num_samples = 0 Then
      SigmaEpistemic = 0#
   Else
      SigmaEpistemic = KBCG20_SigmaEpistemic(period, M, Rrup, AlphaBackarc, AlphaNankai, Ztor, EventType, Vs30, Z1pt0,
Z2pt5, Mb, region, num_samples, PNWbasinStrux)
   End If
```

```
    KBCG20_SigmaTotal = Sqr(SigmaAleatory * SigmaAleatory + SigmaEpistemic * SigmaEpistemic)

End Function



Option Base 1
Function LoadData_KBCG20()
    ' vba CODE DEVELOPED/IMPLEMENTED BY
    '         Silvia Mazzoni, 2020
    '          smazzoni@ucla.edu


    Dim FileName As String
    Dim Path As String

    Dim FileNum As Integer
    Dim DataLine As String

    Dim FileList() As Variant

   OutWrkShLabel = "KBCG20_DataTables"
   Sheets(OutWrkShLabel).Cells.ClearContents
    thisFolder = Range("KBCG19_DataFolder")

    FileList = Range("KBCG19_DataFileList")

'   Dim FileList(1 To 500) As String
    Dim thisFilename As String


    ioutRowOffset = 0
    ' ****************************** process files
    For iFile = LBound(FileList) To UBound(FileList)
        FileLabel = FileList(iFile, 1)

        FileName = thisFolder & "\" & FileLabel & ".csv"
        Worksheets(OutWrkShLabel).Range("a1").Offset(ioutRowOffset, 0).Value = FileLabel
        Worksheets(OutWrkShLabel).Range("a1").Offset(ioutRowOffset, 1).Value = FileName




        ioutRowOffset = ioutRowOffset + 1
        FileNum = FreeFile()
        Open FileName For Input As #FileNum

        inLine = 1
        startRowOffset = ioutRowOffset + 1
        While Not EOF(FileNum)
            Line Input #FileNum, DataLine ' read in data 1 line at a time
            If Len(DataLine) > 0 Then
                dataLines = Split(DataLine, Chr(10))
                For iDataLine = LBound(dataLines) To UBound(dataLines)
                    thisDataLine = dataLines(iDataLine)
                    If Len(thisDataLine) > 0 Then
                        If inLine = 1 Then
                            thisDataLine = Replace$(thisDataLine, Chr(34) & "," & Chr(34), Chr(34) & vbTab & Chr(34))
```

```
                    thisDataLine = Replace$(thisDataLine, Chr(34), "")
                    LineData = Split(thisDataLine, vbTab)
                  Else
                    LineData = Split(thisDataLine, ",")
                  End If
                  For iCol = LBound(LineData) To UBound(LineData)
                    thisValue = LineData(iCol)
                    Worksheets(OutWrkShLabel).Range("a1").Offset(ioutRowOffset, iCol).Value = thisValue
                  Next iCol
                  Ncol = UBound(LineData)
                  ioutRowOffset = ioutRowOffset + 1
                  inLine = inLine + 1
                End If
              Next iDataLine
            End If
          Wend
          endRowOffset = ioutRowOffset
       ActiveWorkbook.Names.Add Name:=FileLabel, RefersToR1C1:= _
         "=KBCG20_DataTables!R" & startRowOffset & "C1:R" & endRowOffset & "C" & 1 + Ncol

     Next iFile




End Function



Option Base 1
' logistc hinge function
Function loghinge(x, x0, a, b0, b1, delta)
    ' vba CODE DEVELOPED/IMPLEMENTED BY
    '         Silvia Mazzoni, 2020
    '          smazzoni@ucla.edu
  loghinge = (a + b0 * (x - x0) + (b1 - b0) * delta * Log(1 + Exp((x - x0) / delta)))
End Function

' interpolation of adjustment to magnitude break point
Function interp_dmb(period)
    ' vba CODE DEVELOPED/IMPLEMENTED BY
    '         Silvia Mazzoni, 2020
    '          smazzoni@ucla.edu
 ' ap = approxfun(c(Log(0.01), Log(1), Log(3), Log(10)), c(0, 0, -0.4, -0.4), rule = 2)



  Aarray = Array(0.01, 1, 3, 10)
  Carray = Array(0, 0, -0.4, -0.4)
  If period = 0 Then
    interp_dmb = interpolateArray(0.01, Aarray, Carray, "log", "linear", "constant")
  ElseIf period = -1 Then
    interp_dmb = 0
  Else
    interp_dmb = interpolateArray(period, Aarray, Carray, "log", "linear", "constant")
  End If
End Function
```

```
' interpolation of k1/k2 (values taken from Campbell and Bozorgnia (2014)
Function interp_k1k2(period)
   ' vba CODE DEVELOPED/IMPLEMENTED BY
   '        Silvia Mazzoni, 2020
   '         smazzoni@ucla.edu
'  periods = [0.005, 0.01, 0.02, 0.03, 0.05, 0.075, 0.1, 0.15, 0.2, 0.25, 0.3, 0.4, 0.5, 0.75, 1., 1.5, 2., 3., 4., 5., 7.5, 10.]
'  k1 = [865., 865., 865., 908., 1054., 1086., 1032., 878., 748., 654., 587., 503., 457., 410.,400., 400., 400., 400., 400., 400.,
400., 400.]
'  k2 = [-1.186, -1.186, -1.219, -1.273, -1.346, -1.471, -1.624, -1.931, -2.188, -2.381, -2.518,-2.657, -2.669, -2.401, -1.955, -
1.025, -0.299, 0., 0., 0., 0., 0.]
   periods = Array(0.005, 0.01, 0.02, 0.03, 0.05, 0.075, 0.1, 0.15, 0.2, 0.25, 0.3, 0.4, 0.5, 0.75, 1#, 1.5, 2#, 3#, 4#, 5#, 7.5, 10#)
   k1 = Array(865#, 865#, 865#, 908#, 1054#, 1086#, 1032#, 878#, 748#, 654#, 587#, 503#, 457#, 410#, 400#, 400#, 400#,
400#, 400#, 400#, 400#, 400#)
   k2 = Array(-1.186, -1.186, -1.219, -1.273, -1.346, -1.471, -1.624, -1.931, -2.188, -2.381, -2.518, -2.657, -2.669, -2.401, -
1.955, -1.025, -0.299, 0#, 0#, 0#, 0#, 0#)

  If period = 0 Then
     ap_k1 = interpolateArray(0.005, periods, k1, "log", "linear", "constant")
     ap_k2 = interpolateArray(0.005, periods, k2, "log", "linear", "constant")
  ElseIf period = -1 Then
     ap_k1 = 400#
     ap_k2 = -1.995
  Else
    ap_k1 = interpolateArray(period, periods, k1, "log", "linear", "constant")
     ap_k2 = interpolateArray(period, periods, k2, "log", "linear", "constant")
'    return (c(ap_k1(log(period)),ap_k2(log(period))))
  End If
   interp_k1k2 = Array(ap_k1, ap_k2)
End Function


' funcation to calculate Z_1/Z2pt5 from Vs30
Function calc_z_from_Vs30(Vs30, coeffs)
   ' vba CODE DEVELOPED/IMPLEMENTED BY
   '        Silvia Mazzoni, 2020
   '         smazzoni@ucla.edu

  zref = coeffs(1) + (coeffs(2) - coeffs(1)) * Exp((Log(Vs30) - coeffs(3)) / coeffs(4)) / (1 + Exp((Log(Vs30) - coeffs(3)) / coeffs(4)))

  calc_z_from_Vs30 = zref

End Function


Option Base 1

Function puts(a)
   Debug.Print a
End Function

Function ConcatenateRange(inRange, separator)


   irow = 0
   For Each thisValue In inRange
      irow = irow + 1
      If irow = 1 Then
         If VarType(thisValue) = vbString Then
```

```
            thisString = """" & thisValue & """"
          Else
            thisString = thisValue
          End If
        Else
          If VarType(thisValue) = vbString Then
            thisString = thisString & ","""" & thisValue & """"
          Else
            thisString = thisString & "," & thisValue
          End If
        End If
    Next thisValue

    ConcatenateRange = thisString

End Function

Function replaceSpace(thisString)

    'Debug.Print thisString
    replaceSpace = Replace(thisString, " ", "_")

End Function

Option Base 1
' updated 6/1/2020 by Silvia Mazzoni
' GMM_at_760_IF_v4.R

'#Grace Parker
'#Modified February 26 to expand comments
'#Modified March 25, 2020 to call consolidated coefficient tables

'# Input Parameters ----------------------------------------------------------

'#Event type: 0 == interface, 1 == slab

'#region corresponds to options in the DatabaseRegion column of the flatfile, plus global. Must be a string. If no matches, default will be global model:
'# "global", "Alaska", "Cascadia", "CAM", "Japan", "SA" or "Taiwan"

'#Saturation Region corresponds to regions defined by R. Archuleta and C. Ji:
'# "global", "Aleutian","Alaska","Cascadia","CAM_S", "CAM_N", "Japan_Pac","Japan_Phi","SA_N","SA_S", "Taiwan_W","Taiwan_E"

'# Rrup is number in kilometers

'# period can be: (-1,0,0.01,0.02,0.025,0.03,0.04,0.05,0.075,0.1,0.15,0.2,0.25,0.3,0.4,0.5,0.75,1,1.5,2,2.5,3,4,5,7.5,10)
'# where -1 == PGV and 0 == PGA

'# Other pertinent information ----------------------------------------------
'# Coefficient files must be in the active working directory
'# This function has no site term. Can only estimate ground motion at the reference condition VS30 = 760m/s
'# The output is the desired median model prediction in LN units
'# Take the exponential to get PGA, PSA in g or the PGV in cm/s



Function PSHAB20_GMM_at_760_IF(EventType, UserRegion, saturation_region_user, Rrup, M, period)
```

```
    ' vba CODE DEVELOPED/IMPLEMENTED BY
    '         Silvia Mazzoni, 2020
    '          smazzoni@ucla.edu

  PSHAB20_GMM_at_760_IF = -999
  If val(Left(EventType, 1)) = 1 Or val(Left(EventType, 1)) = 5 Then
    Exit Function ' ("This function is only for IF")
  End If
  region = getPSHABregion(UserRegion)
  saturation_region = getPSHABSubductionRegion(saturation_region_user)

'# Import Master Coefficient Table ----------------------------------------
  ''import CoefficientsTable
  CoefficientsTable = Range("PSHAB_TableE1_Interface_Coefficients")

  ''Define Mb
    Mb = getMbDefault("PSHAB20", EventType, UserRegion, saturation_region)

'# Constant ---------------------------------------------------------------
  If region = "global" Then
    c0 = getRangeValue(CoefficientsTable, period, "Period (s)", "Global_c0")
  Else
    c0 = getRangeValue(CoefficientsTable, period, "Period (s)", saturation_region & "_c0")
  End If

'# Path Term --------------------------------------------------------------
  h = 10 ^ (-0.82 + 0.252 * M)
  Rref = Sqr(1 + h ^ 2)
  r = Sqr(Rrup ^ 2 + h ^ 2)
  LogR = Log(r)
  R_Rref = Log(r / Rref)

'  #Need  to isolate regional anelastic coefficient, a0
  If region = "global" Then
    a0 = getRangeValue(CoefficientsTable, period, "Period (s)", "Global_a0")
   Else
    a0 = getRangeValue(CoefficientsTable, period, "Period (s)", region & "_a0")
  End If

  If a0 = -999 Or a0 = "NA" Or IsNull(a0) Then
    a0 = getRangeValue(CoefficientsTable, period, "Period (s)", "Global_a0")
  End If

  c1 = getRangeValue(CoefficientsTable, period, "Period (s)", "c1")
  b4 = getRangeValue(CoefficientsTable, period, "Period (s)", "b4")
  Fp = val(c1 * LogR + (b4 * M) * R_Rref + a0 * r)




'# Magnitude Scaling ------------------------------------------------------
  c4 = getRangeValue(CoefficientsTable, period, "Period (s)", "c4")
  c5 = getRangeValue(CoefficientsTable, period, "Period (s)", "c5")
  c6 = getRangeValue(CoefficientsTable, period, "Period (s)", "c6")
  Fm = c4 * func1(M, Mb) + c6 * func2(M, Mb) + c5 * func3(M, Mb)

'# Add it all up! ---------------------------------------------------------
```

```
 mu = c0 + Fp + Fm

 PSHAB20_GMM_at_760_IF = mu


End Function

Function myprint(a)
   ' vba CODE DEVELOPED/IMPLEMENTED BY
   '         Silvia Mazzoni, 2020
   '          smazzoni@ucla.edu
   If IsArray(a) Then
      thisString = ""
      For i = LBound(a) To UBound(a)
       thisString = thisString & "    " & a(i)
      Next i
      Debug.Print thisString

   Else
     Debug.Print a
   End If


End Function



Option Base 1
' updated 6/1/2020 by Silvia Mazzoni

' GMM_at_760_slab_v4.R

'#Grace Parker
'#Modified February 26 to expand comments
'#Modified March 25, 2020 to call coefficients from master table

'# Input Parameters ----------------------------------------------------------

'#Event type: 0 == interface, 1 == slab

'#region corresponds to options in the DatabaseRegion column of the flatfile, plus global. Must be a string. If no matches,
default will be global model:
'# "global", "Alaska", "Cascadia", "CAM", "Japan", "SA" or "Taiwan"

'#Saturation Region corresponds to regions defined by R. Archuleta and C. Ji:
'# "global", "Aleutian","Alaska","Cascadia","Central_America_S", "Central_America_N",
"Japan_Pac","Japan_Phi","South_America_N","South_America_S", "Taiwan_W","Taiwan_E"

'# Rrup is number in kilometers

'#Hypocentral depth in km. To use Ztor value to estimate hypocentral depth, see Ch. 4.3.3 of Parker et al. PEER report

'# period can be: (-1,0,0.01,0.02,0.025,0.03,0.04,0.05,0.075,0.1,0.15,0.2,0.25,0.3,0.4,0.5,0.75,1,1.5,2,2.5,3,4,5,7.5,10)
'# where -1 == PGV and 0 == PGA

'# Other pertinent information ----------------------------------------------
'#Coefficient files must be in the active working directory
```

```
'# This function has no site term. Can only estimate ground motion at the reference condition VS30 = 760m/s
'# The output is the desired median model prediction in LN units
'# Take the exponential to get PGA, PSA in g or the PGV in cm/s

'#Function to compute GMM predictions at 760m/s for slab and interface

Function PSHAB20_GMM_at_760_Slab(EventType, UserRegion, saturation_region_user, Rrup, M, hypocentral_depth,
period)
     ' vba CODE DEVELOPED/IMPLEMENTED BY
    '         Silvia Mazzoni, 2020
    '          smazzoni@ucla.edu

  PSHAB20_GMM_at_760_Slab = -999
  If val(Left(EventType, 1)) = 0 Then
    Exit Function ' ("This function is only for slab")
  End If
  region = getPSHABregion(UserRegion)
  saturation_region = getPSHABSubductionRegion(saturation_region_user)

  ''import Coefficients

'# Import Master Coefficient Table ----------------------------------------
  CoefficientsTable = Range("PSHAB_TableE2_Slab_Coefficients")

   ''Define mb based on Archuleta and Ji (2019)
    Mb = getMbDefault("PSHAB20", EventType, UserRegion, saturation_region)

    ' ''myprint (Mb)
'# Constant ---------------------------------------------------------------
'  #Isolate constant
  If region = "global" Then
    c0 = getRangeValue(CoefficientsTable, period, "Period (s)", "Global_c0")
   ElseIf region = "Alaska" Or region = "SA" Then
    c0 = getRangeValue(CoefficientsTable, period, "Period (s)", saturation_region & "_c0")
   Else
    c0 = getRangeValue(CoefficientsTable, period, "Period (s)", region & "_c0")
  End If
    ' silviamazzoni, I added this:
    If c0 = -999 Or c0 = "NA" Or IsNull(c0) Then
       c0 = getRangeValue(CoefficientsTable, period, "Period (s)", "Global_c0")
    End If

'  # Path Term ------------------------------------------------------------

'  #near-source saturation
  If M <= Mb Then
    littleM = (Log10(35) - Log10(3.12)) / (Mb - 4)
    h = 10 ^ (littleM * (M - Mb) + Log10(35))
   Else
    h = 35
  End If
  Rref = Sqr(1 + h ^ 2)
  r = Sqr(Rrup ^ 2 + h ^ 2)
  LogR = Log(r)
  R_Rref = Log(r / Rref)


'  #Need  to isolate regional anelastic coefficient, a0
```

```
  If region = "global" Then
    a0 = getRangeValue(CoefficientsTable, period, "Period (s)", "Global_a0")
   Else
    a0 = getRangeValue(CoefficientsTable, period, "Period (s)", region & "_a0")
  End If
  If a0 = -999 Or a0 = "NA" Or IsNull(a0) Then
    a0 = getRangeValue(CoefficientsTable, period, "Period (s)", "Global_a0")
  End If


  c1 = getRangeValue(CoefficientsTable, period, "Period (s)", "c1")
  b4 = getRangeValue(CoefficientsTable, period, "Period (s)", "b4")
  Fp = val(c1 * LogR + (b4 * M) * R_Rref + a0 * r)


'  # Magnitude Scaling ----------------------------------------------------------
  c4 = getRangeValue(CoefficientsTable, period, "Period (s)", "c4")
  c5 = getRangeValue(CoefficientsTable, period, "Period (s)", "c5")
  c6 = getRangeValue(CoefficientsTable, period, "Period (s)", "c6")
  Fm = c4 * func1(M, Mb) + c6 * func2(M, Mb) + c5 * func3(M, Mb)




'  # Source Depth Scaling -------------------------------------------------------
  Db = getRangeValue(CoefficientsTable, period, "Period (s)", "db (km)")
  d = getRangeValue(CoefficientsTable, period, "Period (s)", "d")
  littleM = getRangeValue(CoefficientsTable, period, "Period (s)", "m")
  If hypocentral_depth >= Db Then
    Fd = d
   ElseIf hypocentral_depth <= 20 Then
    Fd = littleM * (20 - Db) + d
   Else
    Fd = littleM * (hypocentral_depth - Db) + d
  End If


    mu = c0 + Fp + Fm + Fd
    PSHAB20_GMM_at_760_Slab = mu

End Function

  Function func1(M, Mb)
    ' vba CODE DEVELOPED/IMPLEMENTED BY
    '         Silvia Mazzoni, 2020
    '          smazzoni@ucla.edu
    If M <= Mb Then
       func1 = M - Mb
    Else
       func1 = 0
    End If
  End Function
  Function func2(M, Mb)
    ' vba CODE DEVELOPED/IMPLEMENTED BY
    '         Silvia Mazzoni, 2020
    '          smazzoni@ucla.edu
    If M > Mb Then
       func2 = M - Mb
```

```
    Else
      func2 = 0
    End If
  End Function
  Function func3(M, Mb)
    ' vba CODE DEVELOPED/IMPLEMENTED BY
    '         Silvia Mazzoni, 2020
    '          smazzoni@ucla.edu
    If M <= Mb Then
      func3 = (M - Mb) ^ 2
    Else
      func3 = 0
    End If
  End Function



Option Base 1



Function PSHAB20_Median(EventType, UserRegion, SubductionSlab, Rrup, Magnitude, Zhypo, period, Vs30, Z2pt5,
PNWbasinStrux)
    ' vba CODE DEVELOPED/IMPLEMENTED BY
    '         Silvia Mazzoni, 2020
    '          smazzoni@ucla.edu

'    ' ' ' Debug.print "PSHAB20_Median"


    thisOut = -999
    XinterpMin = 0
    XinterpMax = 10
    XinterpType = "log"
    YinterpType = "log"
    extrapolateType = "extrapolate"

    TvalueList = Range("TvalueList_PSHAB19")
    InterpArray = interpolateFunction(period, TvalueList, XinterpType, extrapolateType, XinterpMin, XinterpMax)

    period0 = TvalueList(InterpArray(1), 1)
    y0 = PSHAB20_Median_AtTlist(EventType, UserRegion, SubductionSlab, Rrup, Magnitude, Zhypo, period0, Vs30, Z2pt5,
PNWbasinStrux)

    If InterpArray(2) <= 0 Then
      thisOut = y0
    Else
      period1 = TvalueList(InterpArray(2), 1)
      y1 = PSHAB20_Median_AtTlist(EventType, UserRegion, SubductionSlab, Rrup, Magnitude, Zhypo, period1, Vs30, Z2pt5,
PNWbasinStrux)
      If LCase(Left(YinterpType, Len("log"))) = LCase("log") Then
        If y0 <= 0 Then
          y0 = 0.000000001
        End If
        If y1 <= 0 Then
          y1 = 0.000000001
        End If
        y0 = Log(y0)
```

```
        y1 = Log(y1)
      End If
      thisOut = y0 + (y1 - y0) * InterpArray(3)
      If LCase(Left(YinterpType, Len("log"))) = LCase("log") Then
        thisOut = Exp(thisOut)
      End If

   End If

   PSHAB20_Median = thisOut

End Function

Function PSHAB20_Median_AtTlist(EventType, UserRegion, saturation_region, Rrup, M, hypocentral_depth, period, Vs30,
Z2pt5, basin)
   ' vba CODE DEVELOPED/IMPLEMENTED BY
   '        Silvia Mazzoni, 2020
   '         smazzoni@ucla.edu

   If val(Left(EventType, 1)) = 1 Or val(Left(EventType, 1)) = 5 Then
      PSHAB20_Median_AtTlist = Exp(PSHAB20_GMM_at_VS30_Slab(EventType, UserRegion, saturation_region, Rrup, M,
hypocentral_depth, period, Vs30, Z2pt5, basin))
   Else
      PSHAB20_Median_AtTlist = Exp(PSHAB20_GMM_at_VS30_IF(EventType, UserRegion, saturation_region, Rrup, M,
period, Vs30, Z2pt5, basin))
   End If

End Function



Option Base 1
' Updated by Silvia Mazzoni 6/1/2020

' GMM_at_Vs30_IF_v4.R

'#Grace Parker
'#Modified February 26, 2020, to expand comments
'#Modified March 25, 2020, to take coefficients from "Table_E1_Interface_Coefficients.csv"

'# Input Parameters ----------------------------------------------------------

'#Event type: 0 == interface, 1 == slab

'#region corresponds to options in the DatabaseRegion column of the flatfile, plus global. Must be a string. If no matches,
default will be global model:
  '# "global", "Alaska", "Cascadia", "CAM", "Japan", "SA" or "Taiwan"

'#Saturation Region corresponds to regions defined by R. Archuleta and C. Ji:
  '# "global", "Aleutian","Alaska","Cascadia","CAM_S", "CAM_N", "Japan_Pac","Japan_Phi","SA_N","SA_S",
"Taiwan_W","Taiwan_E"

'# Rrup is number in kilometers

'# period can be: (-1,0,0.01,0.02,0.025,0.03,0.04,0.05,0.075,0.1,0.15,0.2,0.25,0.3,0.4,0.5,0.75,1,1.5,2,2.5,3,4,5,7.5,10)
  '# where -1 == PGV and 0 == PGA

'#VS30 in units m/s
```

```
'#Z2.5 in units m. Only used if DatabaseRegion == "Japan" or "Cascadia". Can also specify "default" to get no basin term

'#basin is only used if DatabaseRegion == "Cascadia". Value can be 0, 1, or 2, where 0 == having an estimate of Z2.5 outside
mapped basin, 1 == Seattle basin, and 2 == other mapped basin (Tacoma, Everett, Georgia, etc.)

'# Other pertinent information ----------------------------------------------
'#Coefficient files must be in the active working directory
'# "GMM_at_VS30_IF_v3.R" calls function "GMM_at_760_IF_v2.R" to compute PGAr in the nonlinear site term. This function
must be in the R environment else an error will occur.
'# The output is the desired median model prediction in LN units
'# Take the exponential to get PGA, PSA in g or the PGV in cm/s


Function PSHAB20_GMM_at_VS30_IF(EventType, UserRegion, saturation_region_user, Rrup, M, period, Vs30, Z2pt5, basin)
   ' vba CODE DEVELOPED/IMPLEMENTED BY
   '         Silvia Mazzoni, 2020
   '          smazzoni@ucla.edu

  PSHAB20_GMM_at_VS30_IF = -999
  If val(Left(EventType, 1)) = 1 Or val(Left(EventType, 1)) = 5 Then
    Exit Function ' ("This function is only for IF")
  End If
  region = getPSHABregion(UserRegion)
  saturation_region = getPSHABSubductionRegion(saturation_region_user)
  basin = val(Left(basin, 1))

'# Import Master Coefficient Table ---------------------------------------
  CoefficientsTable = Range("PSHAB_TableE1_Interface_Coefficients")


  ''Define Mb
      Mb = getMbDefault("PSHAB20", EventType, UserRegion, saturation_region)

'# Constant ---------------------------------------------------------------
    If region = "global" Then
       c0 = getRangeValue(CoefficientsTable, period, "Period (s)", "Global_c0")
    Else
       c0 = getRangeValue(CoefficientsTable, period, "Period (s)", saturation_region & "_c0")
    End If
    If c0 = -999 Or c0 = "NA" Or IsNull(c0) Then
       c0 = getRangeValue(CoefficientsTable, period, "Period (s)", "Global_c0")
    End If

'# Path Term --------------------------------------------------------------
  h = 10 ^ (-0.82 + 0.252 * M)
  Rref = Sqr(1 + h ^ 2)
  r = Sqr(Rrup ^ 2 + h ^ 2)
  LogR = Log(r)
  R_Rref = Log(r / Rref)

'  #Need  to isolate regional anelastic coefficient, a0
  If region = "global" Then
    a0 = getRangeValue(CoefficientsTable, period, "Period (s)", "Global_a0")
   Else
    a0 = getRangeValue(CoefficientsTable, period, "Period (s)", region & "_a0")
  End If
```

```
  If a0 = -999 Or a0 = "NA" Or IsNull(a0) Then
    a0 = getRangeValue(CoefficientsTable, period, "Period (s)", "Global_a0")
  End If

  c1 = getRangeValue(CoefficientsTable, period, "Period (s)", "c1")
  b4 = getRangeValue(CoefficientsTable, period, "Period (s)", "b4")
  Fp = val(c1 * LogR + (b4 * M) * R_Rref + a0 * r)



'# Magnitude Scaling ----------------------------------------------------------

    c4 = getRangeValue(CoefficientsTable, period, "Period (s)", "c4")
    c5 = getRangeValue(CoefficientsTable, period, "Period (s)", "c5")
    c6 = getRangeValue(CoefficientsTable, period, "Period (s)", "c6")

  Fm = c4 * func1(M, Mb) + c6 * func2(M, Mb) + c5 * func3(M, Mb)


' # Linear Site Amplification ---------------------------------------------


    '#Site Coefficients
    V1 = getRangeValue(CoefficientsTable, period, "Period (s)", "V1 (m/s)")
    V2 = getRangeValue(CoefficientsTable, period, "Period (s)", "V2 (m/s)")
    Vref = getRangeValue(CoefficientsTable, period, "Period (s)", "Vref (m/s)")

  If region = "global" Or region = "CAM" Then
    s2 = getRangeValue(CoefficientsTable, period, "Period (s)", "Global_s2")
    s1 = s2
  ElseIf region = "Taiwan" Or region = "Japan" Then
    s2 = getRangeValue(CoefficientsTable, period, "Period (s)", region & "_s2")
    s1 = getRangeValue(CoefficientsTable, period, "Period (s)", region & "_s1")
  Else
    s2 = getRangeValue(CoefficientsTable, period, "Period (s)", region & "_s2")
    s1 = s2
  End If

  ''Compute linear site term
    If Vs30 <= V1 Then
      Flin = s1 * Log(Vs30 / V1) + s2 * Log(V1 / Vref)
    ElseIf Vs30 <= V2 Then
      Flin = s2 * Log(Vs30 / Vref)
    Else
      Flin = s2 * Log(V2 / Vref)
    End If


' # Nonlinear Site Term ---------------------------------------------------
  PGAr = Exp(PSHAB20_GMM_at_760_IF(EventType, UserRegion, saturation_region, Rrup, M, 0))
    f3 = 0.05
    Vb = 200
    Vref_Fnl = 760
  If period >= 3 Then
    Fnl = 0
  Else
    f4 = getRangeValue(CoefficientsTable, period, "Period (s)", "f4")
    f5 = getRangeValue(CoefficientsTable, period, "Period (s)", "f5")
```

```
    f2 = f4 * (Exp(f5 * (Min(Vs30, Vref_Fnl) - Vb)) - Exp(f5 * (Vref_Fnl - Vb)))
    Fnl = 0 + f2 * Log((PGAr + f3) / f3)


  End If


'# Basin Term ----------------------------------------------------------------
     If Z2pt5 = "default" Or Z2pt5 = "Default" Or Z2pt5 <= 0 Or (region <> "Japan" And region <> "Cascadia") Then
       Fb = 0
    Else
       If region = "Cascadia" Then
         theta0 = 3.94
         theta1 = -0.42
         vmu = 200
         vsig = 0.2
         e1 = getRangeValue(CoefficientsTable, period, "Period (s)", "C_e1")


         If basin = 0 Then
           C_e3 = getRangeValue(CoefficientsTable, period, "Period (s)", "C_e3")
           C_e2 = getRangeValue(CoefficientsTable, period, "Period (s)", "C_e2")
           del_None = getRangeValue(CoefficientsTable, period, "Period (s)", "del_None")
           e3 = C_e3 + del_None
           e2 = C_e2 + del_None
         ElseIf basin = 1 Then
           C_e3 = getRangeValue(CoefficientsTable, period, "Period (s)", "C_e3")
           C_e2 = getRangeValue(CoefficientsTable, period, "Period (s)", "C_e2")
           del_Seattle = getRangeValue(CoefficientsTable, period, "Period (s)", "del_Seattle")
           e3 = C_e3 + del_Seattle
           e2 = C_e2 + del_Seattle
         Else
           C_e3 = getRangeValue(CoefficientsTable, period, "Period (s)", "C_e3")
           C_e2 = getRangeValue(CoefficientsTable, period, "Period (s)", "C_e2")
           e3 = C_e3
           e2 = C_e2
         End If

       ElseIf region = "Japan" Then
         theta0 = 3.05
         theta1 = -0.8
         vmu = 500
         vsig = 0.33
         e3 = getRangeValue(CoefficientsTable, period, "Period (s)", "J_e3")
         e2 = getRangeValue(CoefficientsTable, period, "Period (s)", "J_e2")
         e1 = getRangeValue(CoefficientsTable, period, "Period (s)", "J_e1")
       End If

       Z2pt5_pred = 10 ^ (theta0 + theta1 * (1 + ErfR((Log10(Vs30) - Log10(vmu)) / (vsig * Sqr(2)))))
       delZ2pt5 = Log(Z2pt5) - Log(Z2pt5_pred)


       If delZ2pt5 <= (e1 / e3) Then
         Fb = e1
       ElseIf delZ2pt5 >= (e2 / e3) Then
         Fb = e2
       Else
         Fb = e3 * delZ2pt5
```

```
      End If
   End If


'  # Add it all up! ------------------------------------------------------------

   mu = c0 + Fp + Fnl + Fb + Flin + Fm

  PSHAB20_GMM_at_VS30_IF = mu
End Function




Option Base 1

' GMM_at_VS30_slab_v4.R

'#Grace Parker
'#Modified February 26 to expand comments
'#Modified March 25, 2020 to call consolidated coefficient table

'# Input Parameters ----------------------------------------------------------

'#Event type: 0 == interface, 1 == slab

'#region corresponds to options in the DatabaseRegion column of the flatfile, plus global. Must be a string. If no matches, default will be global model:
'# "global", "Alaska", "Cascadia", "CAM", "Japan", "SA" or "Taiwan"

'#Saturation Region corresponds to regions defined by R. Archuleta and C. Ji:
'# "global", "Aleutian","Alaska","Cascadia","Central_America_S", "Central_America_N",
"Japan_Pac","Japan_Phi","South_America_N","South_America_S", "Taiwan_W","Taiwan_E"

'# Rrup is number in kilometers

'#Hypocentral depth in km. To use Ztor value to estimate hypocentral depth, see Ch. XXX of Parker et al. PEER report

'# period can be: (-1,0,0.01,0.02,0.025,0.03,0.04,0.05,0.075,0.1,0.15,0.2,0.25,0.3,0.4,0.5,0.75,1,1.5,2,2.5,3,4,5,7.5,10)
'# where -1 == PGV and 0 == PGA

'#VS30 in units m/s

'#Z2.5 in units m. Only used if DatabaseRegion == "Japan" or "Cascadia". Can also specify "default" to get no basin term

'#basin is only used if DatabaseRegion == "Cascadia". Value can be 0, 1, or 2, where 0 == having an estimate of Z2.5 outside mapped basin, 1 == Seattle basin, and 0 == other mapped basin (Tacoma, Everett, Georgia, etc.)

'# Other pertinent information ---------------------------------------------
'#Coefficient files must be in the active working directory
'# "GMM_at_VS30_Slab_v2.R" calls function "GMM_at_760_Slab_v2.R" to compute PGAr in the nonlinear site term. This function must be in the R environment else an error will occur.
'# The output is the desired median model prediction in LN units
'# Take the exponential to get PGA, PSA in g or  PGV in cm/s


'#Function to compute GMM predictions at various VS30s for slab
```

```
Function PSHAB20_GMM_at_VS30_Slab(EventType, UserRegion, saturation_region_user, Rrup, M, hypocentral_depth,
period, Vs30, Z2pt5, basin)
   ' vba CODE DEVELOPED/IMPLEMENTED BY
   '        Silvia Mazzoni, 2020
   '         smazzoni@ucla.edu

'    ' ' ' Debug.print "PSHAB20_GMM_at_VS30_Slab"

 PSHAB20_GMM_at_VS30_Slab = -999
 If val(Left(EventType, 1)) = 0 Then
  Exit Function ' ("This function is only for slab")
 End If
 region = getPSHABregion(UserRegion)
 saturation_region = getPSHABSubductionRegion(saturation_region_user)

 basin = val(Left(basin, 1))
 ''import coefficients
 CoefficientsTable = Range("PSHAB_TableE2_Slab_Coefficients")


   'Define mb based on Archuleta and Ji (2019)
   Mb = getMbDefault("PSHAB20", EventType, UserRegion, saturation_region)

' # Constant ----------------------------------------------------------------

'  #Isolate constant
 If region = "global" Then
   c0 = getRangeValue(CoefficientsTable, period, "Period (s)", "Global_c0")
  ElseIf region = "Alaska" Or region = "SA" Then
   c0 = getRangeValue(CoefficientsTable, period, "Period (s)", saturation_region & "_c0")
  Else
   c0 = getRangeValue(CoefficientsTable, period, "Period (s)", region & "_c0")
 End If

 If c0 = -999 Or c0 = "NA" Or IsNull(c0) Then
    c0 = getRangeValue(CoefficientsTable, period, "Period (s)", "Global_c0")
 End If

'# Path Term ----------------------------------------------------------------

'  #near-source saturation
 If M <= Mb Then
   littleM = (Log10(35) - Log10(3.12)) / (Mb - 4)
   h = 10 ^ (littleM * (M - Mb) + Log10(35))
  Else
   h = 35
 End If


 Rref = Sqr(1 + h ^ 2)
 r = Sqr(Rrup ^ 2 + h ^ 2)
 LogR = Log(r)
 R_Rref = Log(r / Rref)


'  #Need  to isolate regional anelastic coefficient, a0
 If region = "global" Then
```

```
    a0 = getRangeValue(CoefficientsTable, period, "Period (s)", "Global_a0")
   Else
    a0 = getRangeValue(CoefficientsTable, period, "Period (s)", region & "_a0")
  End If
  If a0 = -999 Or a0 = "NA" Or IsNull(a0) Then
    a0 = getRangeValue(CoefficientsTable, period, "Period (s)", "Global_a0")
  End If

  c1 = getRangeValue(CoefficientsTable, period, "Period (s)", "c1")
  b4 = getRangeValue(CoefficientsTable, period, "Period (s)", "b4")
  Fp = val(c1 * LogR + (b4 * M) * R_Rref + a0 * r)


'# Magnitude Scaling ----------------------------------------------------------

    c4 = getRangeValue(CoefficientsTable, period, "Period (s)", "c4")
    c5 = getRangeValue(CoefficientsTable, period, "Period (s)", "c5")
    c6 = getRangeValue(CoefficientsTable, period, "Period (s)", "c6")

  Fm = c4 * func1(M, Mb) + c6 * func2(M, Mb) + c5 * func3(M, Mb)

'# Source Depth Scaling -------------------------------------------------------
    Db = getRangeValue(CoefficientsTable, period, "Period (s)", "db (km)")
    d = getRangeValue(CoefficientsTable, period, "Period (s)", "d")
    littleM = getRangeValue(CoefficientsTable, period, "Period (s)", "m")
  ''compute depth scaling term
  If hypocentral_depth >= Db Then
    Fd = d
  ElseIf hypocentral_depth <= 20 Then
    Fd = littleM * (20 - Db) + d
  Else
    Fd = littleM * (hypocentral_depth - Db) + d
  End If

'# Linear Site Amplification -------------------------------------------------

    '#Site Coefficients
    V1 = getRangeValue(CoefficientsTable, period, "Period (s)", "V1 (m/s)")
    V2 = getRangeValue(CoefficientsTable, period, "Period (s)", "V2 (m/s)")
    Vref = getRangeValue(CoefficientsTable, period, "Period (s)", "Vref (m/s)")

  If region = "global" Or region = "CAM" Then
    s2 = getRangeValue(CoefficientsTable, period, "Period (s)", "Global_s2")
    s1 = s2
  ElseIf region = "Taiwan" Or region = "Japan" Then
    s2 = getRangeValue(CoefficientsTable, period, "Period (s)", region & "_s2")
    s1 = getRangeValue(CoefficientsTable, period, "Period (s)", region & "_s1")
  Else
    s2 = getRangeValue(CoefficientsTable, period, "Period (s)", region & "_s2")
    s1 = s2
  End If

  ''Compute linear site term
    If Vs30 <= V1 Then
      Flin = s1 * Log(Vs30 / V1) + s2 * Log(V1 / Vref)
    ElseIf Vs30 <= V2 Then
      Flin = s2 * Log(Vs30 / Vref)
    Else
```

```
    Flin = s2 * Log(V2 / Vref)
  End If

' # Nonlinear Site Term ----------------------------------------------------
 PGAr = Exp(PSHAB20_GMM_at_760_Slab(EventType, UserRegion, saturation_region, Rrup, M, hypocentral_depth, 0))
  f3 = 0.05
  Vb = 200
  Vref_Fnl = 760

 If period >= 3 Then
  Fnl = 0
 Else
  f4 = getRangeValue(CoefficientsTable, period, "Period (s)", "f4")
  f5 = getRangeValue(CoefficientsTable, period, "Period (s)", "f5")
  f2 = f4 * (Exp(f5 * (Min(Vs30, Vref_Fnl) - Vb)) - Exp(f5 * (Vref_Fnl - Vb)))
  Fnl = 0 + f2 * Log((PGAr + f3) / f3)
 End If




'# Basin Term ----------------------------------------------------------
  If Z2pt5 = "default" Or Z2pt5 = "Default" Or Z2pt5 <= 0 Or (region <> "Japan" And region <> "Cascadia") Then
    Fb = 0
  Else
    If region = "Cascadia" Then
     theta0 = 3.94
     theta1 = -0.42
     vmu = 200
     vsig = 0.2
     e1 = getRangeValue(CoefficientsTable, period, "Period (s)", "C_e1")
      C_e3 = getRangeValue(CoefficientsTable, period, "Period (s)", "C_e3")
      C_e2 = getRangeValue(CoefficientsTable, period, "Period (s)", "C_e2")
     If basin = 0 Then
      del_None = getRangeValue(CoefficientsTable, period, "Period (s)", "del_None")
      e3 = C_e3 + del_None
      e2 = C_e2 + del_None
     ElseIf basin = 1 Then
      del_Seattle = getRangeValue(CoefficientsTable, period, "Period (s)", "del_Seattle")
      e3 = C_e3 + del_Seattle
      e2 = C_e2 + del_Seattle
     Else
      e3 = C_e3
      e2 = C_e2
     End If

    ElseIf region = "Japan" Then
     theta0 = 3.05
     theta1 = -0.8
     vmu = 500
     vsig = 0.33
     e3 = getRangeValue(CoefficientsTable, period, "Period (s)", "J_e3")
     e2 = getRangeValue(CoefficientsTable, period, "Period (s)", "J_e2")
     e1 = getRangeValue(CoefficientsTable, period, "Period (s)", "J_e1")
    End If

    Z2pt5_pred = 10 ^ (theta0 + theta1 * (1 + ErfR((Log10(Vs30) - Log10(vmu)) / (vsig * Sqr(2)))))
    delZ2pt5 = Log(Z2pt5) - Log(Z2pt5_pred)
```

```
      If delZ2pt5 <= (e1 / e3) Then
        Fb = e1
       ElseIf delZ2pt5 >= (e2 / e3) Then

        Fb = e2
       Else
        Fb = e3 * delZ2pt5
      End If
   End If


   mu = c0 + Fp + Fm + Fd + Fnl + Flin + Fb


   PSHAB20_GMM_at_VS30_Slab = mu

End Function


Option Base 1

Function testtt()



End Function


Function getMbDefault(NGAsubModelLabel, EventType, UserRegion, saturation_region_user)
   ' vba CODE DEVELOPED/IMPLEMENTED BY
   '        Silvia Mazzoni, 2020
   '         smazzoni@ucla.edu


   ' Silvia Mazzoni: I inserted the global model and replaced the NA values with those of the global model


   If val(Left(EventType, 1)) = 1 Or val(Left(EventType, 1)) = 5 Then  ' slab
      If NGAsubModelLabel = "KBCG20" Then
         'MbKBCG20Intraslab
         saturation_region_here = saturation_region_user
         saturation_regions_SBZ = Array("global", "Aleutian", "Alaska", "Cascadia", "Central_America_N",
"Central_America_S", "Japan_Pac", "Japan_Phi", "New_Zealand_N", "New_Zealand_S", "South_America_N",
"South_America_S", "Taiwan_W", "Taiwan_E")
         thisMbList = Array(7.6, 8, 7.2, 7.2, 7.4, 7.6, 7.6, 7.6, 7.6, 7.6, 7.3, 7.2, 7.7, 7.7)
         User_regions_SBZ = Array("0_global", "1_Alaska", "1_Alaska", "2_Cascadia", "3_CentralAmerica&Mexico",
"3_CentralAmerica&Mexico", "4_Japan", "4_Japan", "5_NewZealand", "5_NewZealand", "6_SouthAmerica",
"6_SouthAmerica", "7_Taiwan", "7_Taiwan")
      Else
         'MbPSHAB20Intraslab
         saturation_region_here = getPSHABSubductionRegion(saturation_region_user)
         Mb_global_Slab = 7.6
         saturation_regions_SBZ = Array("global", "Aleutian", "Alaska", "-999", "Cascadia", "CAM_S", "CAM_N", "Japan_Pac",
"Japan_Phi", "New_Zealand_N", "New_Zealand_S", "SA_N", "SA_S", "Taiwan_W", "Taiwan_E")
         thisMbList = Array(Mb_global_Slab, 7.98, 7.2, Mb_global_Slab, 7.2, 7.6, 7.4, 7.65, 7.55, Mb_global_Slab,
Mb_global_Slab, 7.3, 7.25, 7.7, 7.7)
```

```
        User_regions_SBZ = Array("0_global", "1_Alaska", "1_Alaska", "0_global", "2_Cascadia", "3_CentralAmerica&Mexico",
"3_CentralAmerica&Mexico", "4_Japan", "4_Japan", "5_NewZealand", "5_NewZealand", "6_SouthAmerica",
"6_SouthAmerica", "7_Taiwan", "7_Taiwan")
      End If
   Else
      If NGAsubModelLabel = "KBCG20" Then
         'MbKBCG20Interface
         saturation_region_here = saturation_region_user
         saturation_regions_SBZ = Array("global", "Aleutian", "Alaska", "Cascadia", "Central_America_N",
"Central_America_S", "Japan_Pac", "Japan_Phi", "New_Zealand_N", "New_Zealand_S", "South_America_N",
"South_America_S", "Taiwan_W", "Taiwan_E")
         thisMbList = Array(7.9, 8, 8.6, 8, 7.4, 7.5, 8.5, 7.7, 8.3, 8, 8.5, 8.6, 7.1, 7.1)
         User_regions_SBZ = Array("0_global", "1_Alaska", "1_Alaska", "2_Cascadia", "3_CentralAmerica&Mexico",
"3_CentralAmerica&Mexico", "4_Japan", "4_Japan", "5_NewZealand", "5_NewZealand", "6_SouthAmerica",
"6_SouthAmerica", "7_Taiwan", "7_Taiwan")
      Else
         'MbPSHAB20Interface =
         saturation_region_here = getPSHABSubductionRegion(saturation_region_user)
         Mb_global_IF = 7.9
         saturation_regions_SBZ = Array("global", "Aleutian", "Alaska", "-999", "Cascadia", "CAM_S", "CAM_N", "Japan_Pac",
"Japan_Phi", "New_Zealand_N", "New_Zealand_S", "SA_N", "SA_S", "Taiwan_W", "Taiwan_E")
         thisMbList = Array(Mb_global_IF, 8, 8.6, Mb_global_IF, 7.7, 7.4, 7.4, 8.5, 7.7, Mb_global_IF, Mb_global_IF, 8.5, 8.6,
7.1, 7.1)
         User_regions_SBZ = Array("0_global", "1_Alaska", "1_Alaska", "0_global", "2_Cascadia", "3_CentralAmerica&Mexico",
"3_CentralAmerica&Mexico", "4_Japan", "4_Japan", "5_NewZealand", "5_NewZealand", "6_SouthAmerica",
"6_SouthAmerica", "7_Taiwan", "7_Taiwan")
      End If
   End If

   thisSaturationRegionDbRegion = getArrayMap(saturation_region_here, saturation_regions_SBZ, User_regions_SBZ)
   If LCase(thisSaturationRegionDbRegion) <> LCase(UserRegion) Then
      saturation_region_here = "global"
   End If

   Mb = getArrayMap(saturation_region_here, saturation_regions_SBZ, thisMbList)

   If Mb = -999 Or Mb = "NA" Or IsNull(Mb) Then
     Mb = getArrayMap("global", saturation_regions_SBZ, thisMbList)
   End If

   getMbDefault = Mb



End Function


Function PSHAB20_SigmaTotal(period, Rrup, Vs30, EventType, saturation_region_user)
   ' VBA CODE DEVELOPED/IMPLEMENTED BY
   '        Silvia Mazzoni, 2020
   '         smazzoni@ucla.edu

   SigmaAleatory = PSHAB20_SigmaAleatory(period, Rrup, Vs30)
   SigmaEpistemic = PSHAB20_SigmaEpistemic(period, EventType, saturation_region_user)
   PSHAB20_SigmaTotal = Sqr(SigmaAleatory * SigmaAleatory + SigmaEpistemic * SigmaEpistemic)


End Function
```

```
Function PSHAB20_SigmaEpistemic(Ts, EventType, saturation_region_user)
   ' vba CODE DEVELOPED/IMPLEMENTED BY
   '        Silvia Mazzoni, 2020
   '         smazzoni@ucla.edu
   PSHAB20_SigmaEpistemic = -999
   If val(Left(EventType, 1)) = 1 Or val(Left(EventType, 1)) = 5 Then
      CoefficientsTable = Range("PSHAB20_SigmaEpistemicTable_Slab")
   Else
      CoefficientsTable = Range("PSHAB20_SigmaEpistemicTable_Interface")
   End If
   saturation_region = getPSHABSubductionRegion(saturation_region_user)
   If saturation_region = "Taiwan_W" Or saturation_region = "Taiwan_E" Then
      saturation_region = "Taiwan"
   End If
   If saturation_region = "New_Zealand_N" Or saturation_region = "New_Zealand_S" Then
      saturation_region = "global"
   End If
   SigEp1 = getRangeValue(CoefficientsTable, saturation_region, "Region", "SigEp1")
   If SigEp1 = -999 Then
      saturation_region = "global"
      SigEp1 = getRangeValue(CoefficientsTable, saturation_region, "Region", "SigEp1")
   End If
   SigEp2 = getRangeValue(CoefficientsTable, saturation_region, "Region", "SigEp2")
   T1 = getRangeValue(CoefficientsTable, saturation_region, "Region", "T1")
   T2 = getRangeValue(CoefficientsTable, saturation_region, "Region", "T2")
   If Ts < T1 Or Ts = 0 Then
      sigma_epi = SigEp1
   ElseIf Ts > T2 Then
      sigma_epi = SigEp2
   Else
      sigma_epi = SigEp1 - (SigEp1 - SigEp2) * (Log(Ts / T1) / Log(T2 / T1))
   End If

   PSHAB20_SigmaEpistemic = sigma_epi

End Function


Function PSHAB20_SigmaAleatory(period, Rrup, Vs30)
   ' vba CODE DEVELOPED/IMPLEMENTED BY
   '        Silvia Mazzoni, 2020
   '         smazzoni@ucla.edu

   thisOut = -999
   XinterpMin = 0
   XinterpMax = 10
   XinterpType = "log"
   YinterpType = "log"
   extrapolateType = "extrapolate"

   TvalueList = Range("TvalueList_PSHAB19")
   InterpArray = interpolateFunction(period, TvalueList, XinterpType, extrapolateType, XinterpMin, XinterpMax)

   period0 = TvalueList(InterpArray(1), 1)
   y0 = PSHAB20_SigmaAleatoryAtTlist(period0, Rrup, Vs30)

   If InterpArray(2) <= 0 Then
      thisOut = y0
```

```
    Else
        period1 = TvalueList(InterpArray(2), 1)
        y1 = PSHAB20_SigmaAleatoryAtTlist(period1, Rrup, Vs30)
        If LCase(Left(YinterpType, Len("log"))) = LCase("log") Then
            If y0 <= 0 Then
                y0 = 0.000000001
            End If
            If y1 <= 0 Then
                y1 = 0.000000001
            End If
            y0 = Log(y0)
            y1 = Log(y1)
        End If
        thisOut = y0 + (y1 - y0) * InterpArray(3)
        If LCase(Left(YinterpType, Len("log"))) = LCase("log") Then
            thisOut = Exp(thisOut)
        End If

    End If

    PSHAB20_SigmaAleatory = thisOut

End Function

Function PSHAB20_SigmaAleatoryAtTlist(period, Rrup, Vs30)
    ' vba CODE DEVELOPED/IMPLEMENTED BY
    '         Silvia Mazzoni, 2020
    '          smazzoni@ucla.edu

    PSHAB20_SigmaAleatoryAtTlist = -999

    PhiTot = PSHAB20_PhiTot(period, Rrup, Vs30)
    Tau = PSHAB20_Tau(period, Rrup, Vs30)


    PSHAB20_SigmaAleatoryAtTlist = Sqr(PhiTot * PhiTot + Tau * Tau)

End Function



Function PSHAB20_Tau(period, Rrup, Vs30)
    ' vba CODE DEVELOPED/IMPLEMENTED BY
    '         Silvia Mazzoni, 2020
    '          smazzoni@ucla.edu

    PSHAB20_Tau = -999
    SigmacoefficientsTable = Range("PSHAB_AleatoryCoefficients")
    Tau = getRangeValue(SigmacoefficientsTable, period, "Period (s)", "Tau")
    PSHAB20_Tau = Tau
End Function


Function PSHAB20_PhiTot(period, Rrup, Vs30)
    ' vba CODE DEVELOPED/IMPLEMENTED BY
    '         Silvia Mazzoni, 2020
    '          smazzoni@ucla.edu
```

```
    PSHAB20_PhiTot = -999
    SigmacoefficientsTable = Range("PSHAB_AleatoryCoefficients")

    PhiTot_Phi1squared = getRangeValue(SigmacoefficientsTable, period, "Period (s)", "PhiTot_Phi1squared")
    PhiTot_Phi2squared = getRangeValue(SigmacoefficientsTable, period, "Period (s)", "PhiTot_Phi2squared")
    PhiTot_PhiVsquared = getRangeValue(SigmacoefficientsTable, period, "Period (s)", "PhiTot_PhiVsquared")

    'Corner Distances:
    R1 = 200    ' km
    R2 = 500    ' km

    V1 = 200    ' m/s
    V2 = 500    ' m/s

    If Vs30 <= V1 Then
       Rprime = Max(R1, Min(R2, Rrup))
       deltaVar = PhiTot_PhiVsquared * (Log(R2 / Rprime)) / (Log(R2 / R1))
    ElseIf Vs30 < V2 Then
       Rprime = Max(R1, Min(R2, Rrup))
       deltaVar = PhiTot_PhiVsquared * (Log(R2 / Rprime)) / (Log(R2 / R1)) * (Log(V2 / Vs30)) / (Log(V2 / V1))
    Else
       deltaVar = 0
    End If

    If Rrup < R1 Then
       phiSquared = PhiTot_Phi1squared
    ElseIf Rrup < R2 Then
       phiSquared = (PhiTot_Phi2squared - PhiTot_Phi1squared) / (Log(R2 / R1)) * (Log(Rrup / R1)) + PhiTot_Phi1squared
    Else
       phiSquared = PhiTot_Phi2squared

    End If

    PSHAB20_PhiTot = Sqr(phiSquared + deltaVar)

End Function

Function PSHAB20_SigmaS2S(period, Rrup, Vs30)
    ' vba CODE DEVELOPED/IMPLEMENTED BY
    '         Silvia Mazzoni, 2020
    '          smazzoni@ucla.edu

    PSHAB20_SigmaS2S = -999
    SigmacoefficientsTable = Range("PSHAB_AleatoryCoefficients")

    PhiS2S_PhiS2S0squared = getRangeValue(SigmacoefficientsTable, period, "Period (s)", "PhiS2S_PhiS2S0squared")
    PhiS2S_a1 = getRangeValue(SigmacoefficientsTable, period, "Period (s)", "PhiS2S_a1")
    VM = getRangeValue(SigmacoefficientsTable, period, "Period (s)", "VM")

    'Corner Distances:
    R3 = 200    ' km
    R4 = 500    ' km

    V3 = 200    ' m/s
    V4 = 800    ' m/s

    If Vs30 <= V3 Then
       Rprime = Max(R3, Min(R4, Rrup))
```

```
    deltaVarS2S = PhiS2S_a1 * Log(V3 / VM) * (Log(R4 / Rprime)) / (Log(R4 / R3))
  ElseIf Vs30 < VM Then
    Rprime = Max(R3, Min(R4, Rrup))
    deltaVarS2S = PhiS2S_a1 * Log(Vs30 / VM) * (Log(R4 / Rprime)) / (Log(R4 / R3))
  ElseIf Vs30 < V4 Then
    deltaVarS2S = PhiS2S_a1 * Log(Vs30 / VM)
  Else
    deltaVarS2S = PhiS2S_a1 * Log(V4 / VM)
  End If

  PSHAB20_SigmaS2S = Sqr(PhiS2S_PhiS2S0squared + deltaVarS2S)

End Function


Function PSHAB20_SigmaSS(period, Rrup, Vs30)
  ' vba CODE DEVELOPED/IMPLEMENTED BY
  '        Silvia Mazzoni, 2020
  '         smazzoni@ucla.edu

  PSHAB20_SigmaSS = -999
  SigmacoefficientsTable = Range("PSHAB_AleatoryCoefficients")

  PhiSS_PhiSS1squared = getRangeValue(SigmacoefficientsTable, period, "Period (s)", "PhiSS_PhiSS1squared")
  PhiSS_PhiSS2squared = getRangeValue(SigmacoefficientsTable, period, "Period (s)", "PhiSS_PhiSS2squared")
  PhiSS_a2 = getRangeValue(SigmacoefficientsTable, period, "Period (s)", "PhiSS_a2")
  VM = getRangeValue(SigmacoefficientsTable, period, "Period (s)", "VM")

  'Corner Distances:
  R3 = 200   ' km
  R4 = 500   ' km
  R5 = 500   ' km
  R6 = 800   ' km

  V3 = 200   ' m/s
  V4 = 800   ' m/s

  If Rrup < R5 Then
    phiSquaredSS = PhiSS_PhiSS1squared
  ElseIf Rrup < R6 Then
    phiSquaredSS = (PhiSS_PhiSS2squared - PhiSS_PhiSS1squared) / (Log(R6 / R5)) * (Log(Rrup / R5)) + PhiSS_PhiSS1squared
  Else
    phiSquaredSS = PhiSS_PhiSS2squared

  End If

  If Vs30 <= V3 Then
    Rprime = Max(R3, Min(R4, Rrup))
    deltaVarSS = PhiSS_a2 * Log(V3 / VM) * (Log(R4 / Rprime)) / (Log(R4 / R3))
  ElseIf Vs30 < VM Then
    Rprime = Max(R3, Min(R4, Rrup))
    deltaVarSS = PhiSS_a2 * Log(Vs30 / VM) * (Log(R4 / Rprime)) / (Log(R4 / R3))
  ElseIf Vs30 < V4 Then
    deltaVarSS = PhiSS_a2 * Log(Vs30 / VM)
  Else
    deltaVarSS = PhiSS_a2 * Log(V4 / VM)
  End If
```

```
   PSHAB20_SigmaSS = Sqr(phiSquaredSS + deltaVarSS)

End Function



Option Base 1
Function getPSHABregion(UserRegion)
   ' vba CODE DEVELOPED/IMPLEMENTED BY
   '        Silvia Mazzoni, 2020
   '         smazzoni@ucla.edu

   PSHAB_RegionIDListMap = Range("PSHAB_RegionIDListMap")
   RegionIDList = Range("RegionIDList")

   getPSHABregion = getColumnRangeMap(UserRegion, RegionIDList, PSHAB_RegionIDListMap)

End Function



Function getPSHABSubductionRegion(UserSubductionRegion)
   ' vba CODE DEVELOPED/IMPLEMENTED BY
   '        Silvia Mazzoni, 2020
   '         smazzoni@ucla.edu

   PSHAB_SubductionRegionIDListMap = Range("SaturationRegionListPSHAB")
   SubductionRegionList = Range("SaturationRegionListAll")

   getPSHABSubductionRegion = getColumnRangeMap(UserSubductionRegion, SubductionRegionList,
PSHAB_SubductionRegionIDListMap)

End Function


Option Base 1
' I found this at:
'http://www.a1vbcode.com/snippet-5481.asp

'This function calculates erf (error function or gaussian aerror
'function) for complex and real numbers.
'
'   erf ( x ) = 2 / sqrt (pi) * integration (t=0,x) [  exp ( - t ^2 ) dt ]
'
'There is no Visual Basic or VBA (Excel) code to calculate erf
'function for complex numbers, useful calculation for math
'applcations.
'
'i 've based my code (see the source comments) in a code for
'MatLab, but it's more easy to do this in MatLab because
'MatLab does complex algebra.
'
'Gaussian Error Function is part of Cumulative Distribution
'Function (CDF) of a Normal Distribution:
'
'CDF = 1/2 * ( 1 +  erf [ (x - m ) / raiz(2 s^2 ) ]
'
```

```
'    Where m is the average, s is standard deviation and x is a
'aleatory variable.
'
'    If a normal distribution has mean 0 and variance (s^2) =
'0.5, the above expression becomes:
'
'CDF = 1/2 * (1 + erf(x) ) = 1/2 + 1/2 erf(x)
'
'    The probability from -infinite to 0 is 1/2 from a normal
'distribution with mean 0.  So erf(x) can be interpreted like
'twice the probability of a normal (average 0 and variance 0.5)
'to have a value between 0 and x.


'*********************************************************
'*  Paulo Buchsbaum - May,23,2012
'*
'* ERFZ Error function for complex (x,y). Return imaginary part.
'*     because for the most practical use in complex evaluation
'*     we use x=0 so the the real part is 0.
'*     The real part is returned on third parameter (RV)
'*
'*  The function -erfz(0,i) solves the expression i * erf ( bi),because
'*  erf(bi) = 0 + ti, portanto i * erf(bi) =-ti
'*
'*  .NET programmers - Includes Imports System.Math in you code
'*                Uses ATan and not Atn, and Sqrt and not Sqr
'*   (http://msdn.microsoft.com/en-us/library/thc0a116(v=vs.80).aspx)
'*
'*
'*  Based on MatLab Code (2001) written by
'*   Paul Godfrey <pgodfrey@intersil.com> and Peter J.Acklam <jacklam@math.uio.no>
'*   Ref: Abramowitz & Stegun section 7.1 equations 7.1.9,7.1.23, and 7.1.29
'* (http://www.mathworks.com/matlabcentral/fileexchange/3574-erfz)
'*********************************************


Function ErfZ(x As Double, y As Double, Optional ByRef Rv) As Double
   ' vba CODE DEVELOPED/IMPLEMENTED BY
   '         Silvia Mazzoni, 2020
   '          smazzoni@ucla.edu

   Const NMax = 193
   Const nn = 32
   Dim Re As Double
   Dim sqrtpi As Double
   Dim twopi As Double
   Dim az As Double
   Dim n As Integer
   Dim k1 As Double
   Dim k2r As Double, k2i As Double
   Dim k As Integer
   Dim xk As Double, yk As Double
   Dim Pi As Double
   Dim s1 As Double
   Dim s2r As Double, s2i As Double
   Dim s3 As Double
   Dim s4r As Double, s4i As Double
```

```
Dim s5r As Double, s5i As Double
Dim s6r As Double, s6i As Double
Dim sr As Double, si As Double
Dim yr As Double, yi As Double
Dim a As Double, b As Double

Dim Im As Double

' If real argument calculates with other function and exits

If y = 0 Then
  ErfZ = ErfR(x)
  Exit Function
End If

Pi = 4 * Atn(1)  ' Atn(1) = pi/4 (45 o.)
twopi = 2 * Pi

sqrtpi = Sqr(Pi) ' 1.772453850905516027298

az = Sqr(x ^ 2 + y ^ 2)

If az <= 8 Then
   k1 = 2 / Pi * Exp(-x * x)

   ' Euler's Formula: e(xi) = cos(x) + i sin(x)
   '
   ' k2 = Exp(-i * 2 * x * y)

   k2r = Cos(-2 * x * y)
   k2i = Sin(-2 * x * y)

   s1 = ErfR(x)

   If x <> 0 Then
      s2r = k1 / (4 * x) * (1 - k2r)
      s2i = k1 / (4 * x) * (1 - k2i)
   Else
      s2r = 0
      s2i = 1 / Pi * y
   End If

   Re = s1 + s2r
   Im = s2i

   xk = x
   yk = y

   s5r = 0
   s5i = 0
   For n = 1 To nn
      s3 = Exp(-n * n / 4) / (n * n + 4 * xk * xk)

      'Calculate s4 = 2*xk - k2 * (2*xk * cosh(n*yk) -i*n*sinh(n*yk))
      '
      ' (a+bi) * (c+di) =
      '    (ac-bd) + (ad + cb) i
      ' Where
```

```
      '   a = -k2r
      '   b = -k2i
      '   c = 2*xk * cosh(n*yk)
      '   d = - n * sinh(n*yj)

      s4r = 2 * xk - k2r * 2 * xk * Cosh(n * yk) - k2i * n * Sinh(n * yk)
      s4i = k2r * n * Sinh(n * yk) - k2i * 2 * xk * Cosh(n * yk)

      s5r = s5r + s3 * s4r
      s5i = s5i + s3 * s4i
    Next n

    s6r = k1 * s5r
    s6i = k1 * s5i

    Re = Re + s6r
    Im = Im + s6i

  Else
    If x <> 0 Then
      x = -x
      y = -y
    End If

    sr = 1
    si = 0

    ' Calculate y = 2 * z * z
    '
    ' Complex squared
    '
    '   ( a + bi ) * (a + bi ) =  (a^2-b^2) + 2ab i


    yr = 2 * x ^ 2 - 2 * y ^ 2
    yi = 2 * x * y



    For n = NMax To 1 Step -2

     ' Complex Quotient
     '   ( a+ bi ) / (c + di)
     '     =  (ac+bd)/(c2+d2) + (bc-ad)/(c2+d2)i
     '
     '  Calculate s = 1 - n * (s / y)
       az = (yr ^ 2 + yi ^ 2)
       sr = 1 - n * (sr * yr + si * yi) / az
       si = -n * (si * yr - sr * yi) / az

    Next

  ' z = x + yi
  ' - z * z =  - (x + yi) * (x+yi) = y^2 - x^2 - 2xy i
  '
  ' Euler's Formula:  e(xi) = cos(x) + i sin(x)
  '
  ' Where exp(-z * z) = exp(y^2 - x^2) * exp(-2xyi)
```

```
    '    =exp(y^2 - x^2) * ( cos(-2*x*y) + i sin (-2*x*y) )
    '
    ' Complex Quotient
    '   ( a+ bi ) / (c + di)
    '     =  (ac+bd)/(c2+d2) + (bc-ad)/(c2+d2)i
    '
    ' Calculate  f = 1 - s * Exp(-z * z) / (sqrtpi * z)
    '
    ' Before calculate  Exp(-z * z) / (sqrtpi * z)


      a = Exp(y ^ 2 - x ^ 2) * Cos(-2 * x * y) / sqrtpi
      b = Exp(y ^ 2 - x ^ 2) * Sin(-2 * x * y) / sqrtpi
      az = x ^ 2 + y ^ 2

      Re = (a * x + b * y) / az
      Im = (b * x - a * y) / az

    ' Calculate now  1 -s * (Re,Im)
    '
    '  Complex product: (a+bi) * (c+di) =  (ac-bd) + (ad + bc) i
    '


      Re = 1 - sr * Re + si * Im
      Im = -sr * Im - si * Re

      If x <> 0 Then
        Re = -Re
        Im = -Im
      End If

      If x = 0 Then
        Re = Re - 1
      End If

    End If

    Rv = Re
    ErfZ = Im

End Function
'*********************************************************
'*              ErfReal
'*   Gaussian Error function with real argument X
'*
'*  Based on
'*    Handbook of Mathematical Functions: with Formulas,Graphs, and Mathematical Tables
'*    Milton Abramowitz  e Irene A. Stegun
'*    Cited by John D. Cook that post a Python code in your blog
'*    ( http://www.johndcook.com/blog/2009/01/19/stand-alone-error-function-erf/ )
'*********************************************************
Function ErfReal(x As Double) As Double
    ' vba CODE DEVELOPED/IMPLEMENTED BY
    '         Silvia Mazzoni, 2020
    '          smazzoni@ucla.edu

' Constants
Const a1 = 0.254829592
```

```
Const a2 = -0.284496736
Const a3 = 1.421413741
Const a4 = -1.453152027
Const a5 = 1.061405429
Dim p As Double, t As Double, y As Double
Dim sign As Integer

p = 0.3275911

   ' x signal
sign = IIf(x < 0, -1, 1)
x = Abs(x)

t = 1 / (1 + p * x)
y = 1 - (((((a5 * t + a4) * t) + a3) * t + a2) * t + a1) * t * Exp(-x * x)

ErfReal = sign * y
End Function
'*********************************************************
'*              ErfR
'*   Gaussian Error function with real argument X
'*   using a standard Excel function
'*
'*     Paulo Buchsbaum - May/2012
'*********************************************************
Function ErfR(x As Double) As Double
   ' vba CODE DEVELOPED/IMPLEMENTED BY
   '        Silvia Mazzoni, 2020
   '         smazzoni@ucla.edu

' CDF of Normal Distribution = 1/2 * ( 1 +  erf [ (x -m ) / raiz(2*s^2 ) ]
' If Average = 0 and Standard Deviation = 1
'  CDF = 1/2 + 1/2 Erf (x)
'  Erf = ( CDF-1/2 )*2   (Double the probability from 0 to x )
'
On Error Resume Next
'  Works on Excel's VBA
ErfR = (WorksheetFunction.NormDist(x, 0, Sqr(1 / 2), True) - 0.5) * 2
If Err.Number <> 0 Then ErfR = ErfReal(x)

On Error GoTo 0
End Function


'*********************************************************
'*              Cosh
'*   Hyperbolic Cosine of x
'*********************************************************
Function Cosh(x As Double) As Double
   ' vba CODE DEVELOPED/IMPLEMENTED BY
   '        Silvia Mazzoni, 2020
   '         smazzoni@ucla.edu
' Modern VB Net has a native function math.cosh
' Excel users has worksheetfunction.cosh
Cosh = (Exp(x) + Exp(-x)) / 2
End Function
'*********************************************************
'*              Sinh
```

```
'*   Hyperbolic Syne of x
'*******************************************************
Function Sinh(x As Double) As Double
   ' vba CODE DEVELOPED/IMPLEMENTED BY
   '        Silvia Mazzoni, 2020
   '         smazzoni@ucla.edu
' Modern VB Net has a native function math.sinh
' Excel users has worksheetfunction.sinh
Sinh = (Exp(x) - Exp(-x)) / 2
End Function

Option Base 1
Option Compare Text

'https://stackoverflow.com/questions/11504418/excel-vba-quickest-way-to-sort-an-array-of-numbers-in-descending-order'
' Omit plngLeft & plngRight; they are used internally during recursion

Function BubbleSrt(ArrayIn, Ascending As Boolean)
   ' vba CODE DEVELOPED/IMPLEMENTED BY
   '        Silvia Mazzoni, 2020
   '         smazzoni@ucla.edu

   Dim SrtTemp As Variant
   Dim i As Long
   Dim j As Long

   If Ascending = True Then
      For i = LBound(ArrayIn) To UBound(ArrayIn)
         For j = i + 1 To UBound(ArrayIn)
            If ArrayIn(i) > ArrayIn(j) Then
               SrtTemp = ArrayIn(j)
               ArrayIn(j) = ArrayIn(i)
               ArrayIn(i) = SrtTemp
            End If
         Next j
      Next i
   Else
      For i = LBound(ArrayIn) To UBound(ArrayIn)
         For j = i + 1 To UBound(ArrayIn)
            If ArrayIn(i) < ArrayIn(j) Then
               SrtTemp = ArrayIn(j)
               ArrayIn(j) = ArrayIn(i)
               ArrayIn(i) = SrtTemp
            End If
         Next j
      Next i
   End If

   BubbleSrt = ArrayIn

End Function


Function sum(thisList)
   ' vba CODE DEVELOPED/IMPLEMENTED BY
   '        Silvia Mazzoni, 2020
   '         smazzoni@ucla.edu
```

```
   sum = 0
   For irow = LBound(thisList) To UBound(thisList)
      sum = sum + thisList(irow)
   Next irow

End Function
Function dotProduct(Avector, Bvector)
   ' vba CODE DEVELOPED/IMPLEMENTED BY
   '        Silvia Mazzoni, 2020
   '         smazzoni@ucla.edu

   dotProduct = 0
   For irow = LBound(Avector) To UBound(Avector)
      dotProduct = dotProduct + Avector(irow) * Bvector(irow)
   Next irow
End Function




Function getArrayCountIndex(Xvalue, Xarray)
   ' vba CODE DEVELOPED/IMPLEMENTED BY
   '        Silvia Mazzoni, 2020
   '         smazzoni@ucla.edu

   getArrayCountIndex = -999
   For irow = LBound(Xarray) To UBound(Xarray)
      If Xarray(irow) = Xvalue Then
         getArrayCountIndex = irow - LBound(Xarray) + 1
      End If
   Next irow
End Function


Function getArrayMap(Xvalue, Xarray, Yarray)
   ' vba CODE DEVELOPED/IMPLEMENTED BY
   '        Silvia Mazzoni, 2020
   '         smazzoni@ucla.edu
   getArrayMap = -999
   For irow = LBound(Xarray) To UBound(Xarray)
      If Xarray(irow) = Xvalue Then
         getArrayMap = Yarray(irow - LBound(Xarray) + LBound(Yarray))
      End If
   Next irow
End Function
Function getColumnRangeMap(Xvalue, Xarray, Yarray)
   ' vba CODE DEVELOPED/IMPLEMENTED BY
   '        Silvia Mazzoni, 2020
   '         smazzoni@ucla.edu

   getColumnRangeMap = -999
   For irow = LBound(Xarray, 1) To UBound(Xarray, 1)
      If Xarray(irow, 1) = Xvalue Then
         getColumnRangeMap = Yarray(irow - LBound(Xarray, 1) + LBound(Yarray, 1), 1)
         Exit Function
      End If
   Next irow
End Function
```

```
Function getRangeValue(thisRange, Xvalue, xHeader, yHeader)
   ' Debug.Print "in getRangeValue"
   ' vba CODE DEVELOPED/IMPLEMENTED BY
   '        Silvia Mazzoni, 2020
   '         smazzoni@ucla.edu
   getRangeValue = -999
   xColIndex = -1
   yColIndex = -1
   irow = LBound(thisRange, 1)
   For iCol = LBound(thisRange, 2) To UBound(thisRange, 2)
      thisHeaderValue = thisRange(irow, iCol)
      If LCase(thisHeaderValue) = LCase(xHeader) Then
         xColIndex = iCol
      End If
      If LCase(thisHeaderValue) = LCase(yHeader) Then
         yColIndex = iCol
      End If
   Next iCol
   If xColIndex > 0 And yColIndex > 0 Then
      For irow = irow + 1 To UBound(thisRange, 1)
         thisXvalue = thisRange(irow, xColIndex)
         If IsNumeric(thisXvalue) Then
            If Abs(thisXvalue - Xvalue) < 0.000000001 Then
               getRangeValue = thisRange(irow, yColIndex)
            End If
         Else
            If thisXvalue = Xvalue Then
               getRangeValue = thisRange(irow, yColIndex)
            End If
         End If
      Next irow
   End If
End Function

Function getRangeColumnIndex(thisRange, Xvalue, xHeader)
   ' vba CODE DEVELOPED/IMPLEMENTED BY
   '        Silvia Mazzoni, 2020
   '         smazzoni@ucla.edu
   getRangeColumnIndex = -999

   irow = LBound(thisRange, 1)
   For iCol = LBound(thisRange, 2) To UBound(thisRange, 2)
      thisHeaderValue = thisRange(irow, iCol)
      If LCase(thisHeaderValue) = LCase(xHeader) Then
         getRangeColumnIndex = iCol
      End If
   Next iCol
End Function

Function getRangeRowIndex(thisRange, Xvalue, xHeader)
   ' vba CODE DEVELOPED/IMPLEMENTED BY
   '        Silvia Mazzoni, 2020
   '         smazzoni@ucla.edu

   getRangeRowIndex = -999
   xColIndex = -1
```

```
    irow = LBound(thisRange, 1)
    For iCol = LBound(thisRange, 2) To UBound(thisRange, 2)
        thisHeaderValue = thisRange(irow, iCol)
        If LCase(thisHeaderValue) = LCase(xHeader) Then
            xColIndex = iCol
        End If
    Next iCol

    If xColIndex > 0 And yColIndex > 0 Then
        For irow = irow + 1 To UBound(thisRange, 1)
            thisXvalue = thisRange(irow, xColIndex)
            If Abs(thisXvalue - Xvalue) < 0.000000001 Then
                getRangeRowIndex = irow
            End If
        Next irow
    End If
End Function


Function testGetRangeRow()
    ' vba CODE DEVELOPED/IMPLEMENTED BY
    '         Silvia Mazzoni, 2020
    '         smazzoni@ucla.edu

     Parameters = Range("KBCG19_coefficients")

    tt = getRangeRow(Parameters, 0#, "T")

End Function
Function getRangeRow(thisRange, Xvalue, xHeader)
    ' vba CODE DEVELOPED/IMPLEMENTED BY
    '         Silvia Mazzoni, 2020
    '         smazzoni@ucla.edu
    getRangeRow = -999
    xColIndex = -1
    Dim temP()
    ReDim temP(UBound(thisRange, 2))

    startRow = LBound(thisRange, 1)
    For iCol = LBound(thisRange, 2) To UBound(thisRange, 2)
        thisHeaderValue = thisRange(startRow, iCol)
        If LCase(thisHeaderValue) = LCase(xHeader) Then
            xColIndex = iCol
        End If
    Next iCol

    If xColIndex > 0 Then
        For irow = startRow + 1 To UBound(thisRange, 1)
            thisXvalue = thisRange(irow, xColIndex)
            If Abs(thisXvalue - Xvalue) < 0.000000001 Then
                iIndex = 1
                For iCol = LBound(thisRange, 2) To UBound(thisRange, 2)
                    temP(iIndex) = thisRange(irow, iCol)
                    iIndex = iIndex + 1
                Next iCol
            End If
        Next irow
    End If
```

```
   getRangeRow = temP

End Function
Function getRangeRowByIndex(thisRange, rowIndex)
    ' vba CODE DEVELOPED/IMPLEMENTED BY
   '        Silvia Mazzoni, 2020
   '         smazzoni@ucla.edu
  getRangeRowByIndex = -999
   Dim temP()
   ReDim temP(UBound(thisRange, 2))
   iIndex = 1
   For iCol = LBound(thisRange, 2) To UBound(thisRange, 2)
      this = thisRange(rowIndex, iCol)
      temP(iIndex) = thisRange(rowIndex, iCol)
      iIndex = iIndex + 1
   Next iCol

   getRangeRowByIndex = temP

End Function
Function getSubArray(inArray, indicesArray)
   ' vba CODE DEVELOPED/IMPLEMENTED BY
   '        Silvia Mazzoni, 2020
   '         smazzoni@ucla.edu

   Dim temP()
   ReDim temP(UBound(indicesArray))

   irow = 1
   For ii = LBound(indicesArray) To UBound(indicesArray)
      thisIndex = indicesArray(ii)
      temP(irow) = inArray(thisIndex)
      irow = irow + 1
   Next ii
   getSubArray = temP

End Function

Function getSubArrayRange(inArray, startIndex, endIndex)
   ' vba CODE DEVELOPED/IMPLEMENTED BY
   '        Silvia Mazzoni, 2020
   '         smazzoni@ucla.edu

   Dim temP()
   ReDim temP(UBound(inArray))

   irow = 1
   For thisIndex = startIndex To endIndex
      temP(irow) = inArray(thisIndex)
      irow = irow + 1
   Next thisIndex
   getSubArrayRange = temP
End Function


Function updateArray(newArray, newArrayIndices, oldArray, oldArrayIndices)
   ' vba CODE DEVELOPED/IMPLEMENTED BY
```

```
    '        Silvia Mazzoni, 2020
    '        smazzoni@ucla.edu

    For irow = LBound(oldArrayIndices) To UBound(oldArrayIndices)
        oldIndex = oldArrayIndices(irow)
        newIndex = newArrayIndices(irow)
        newArray(newIndex) = oldArray(oldIndex)
    Next irow
    updateArray = newArray
End Function

Function makeArray(oldArray, oldArrayIndices)
    ' vba CODE DEVELOPED/IMPLEMENTED BY
    '        Silvia Mazzoni, 2020
    '        smazzoni@ucla.edu

    Dim newArray()
    ReDim newArray(UBound(oldArrayIndices))

    newIndex = 1
    For irow = LBound(oldArrayIndices) To UBound(oldArrayIndices)
        oldIndex = oldArrayIndices(irow)
        newArray(newIndex) = oldArray(oldIndex)
        newIndex = newIndex + 1
    Next irow
    makeArray = newArray
End Function

Function getMeanOfArray(inArray)
    ' vba CODE DEVELOPED/IMPLEMENTED BY
    '        Silvia Mazzoni, 2020
    '        smazzoni@ucla.edu

    getMeanOfArray = 0
    iCount = 0
    For ii = LBound(inArray) To UBound(inArray)
        getMeanOfArray = getMeanOfArray + inArray(ii)
        iCount = iCount + 1
    Next ii
    getMeanOfArray = getMeanOfArray / iCount

End Function

Function getMedianOfArray(inArray)
    ' vba CODE DEVELOPED/IMPLEMENTED BY
    '        Silvia Mazzoni, 2020
    '        smazzoni@ucla.edu

    inArray = BubbleSrt(inArray, True)
    num_samples = UBound(inArray)
    If Round(num_samples / 2, 0) = num_samples / 2 Then
        getMedianOfArray = 0.5 * (inArray(num_samples / 2) + inArray(num_samples / 2 + 1))
    Else
        getMedianOfArray = inArray((num_samples + 1) / 2)
    End If

End Function
```

```
'https://www.mrexcel.com/board/threads/standard-deviation-of-an-array.206375/
'*************************************************************************
'*            Standard Deviation  of a 1D array                 *
'*************************************************************************
Function getStdDev(Arr)
   ' vba CODE DEVELOPED/IMPLEMENTED BY
   '        Silvia Mazzoni, 2020
   '         smazzoni@ucla.edu
    Dim i As Integer
    Dim avg As Single, SumSq As Single
    Dim k1 As Long, k2 As Long

    Dim n As Long

    k1 = LBound(Arr)
    k2 = UBound(Arr)

    n = 0
    avg = getMeanOfArray(Arr)
    SumSq = 0
    For i = k1 To k2
      n = n + 1
       SumSq = SumSq + (Arr(i) - avg) ^ 2
    Next i

    getStdDev = Sqr(SumSq / (n - 1))

End Function

Function getAllowedSaturationList(UserRegion, iUser)
   ' vba CODE DEVELOPED/IMPLEMENTED BY
   '        Silvia Mazzoni, 2020
   '         smazzoni@ucla.edu

   If UserRegion = "0_global" Then
      If iUser <= 1 Then
         getAllowedSaturationList = "global"
      Else
         getAllowedSaturationList = ""
      End If
   Else
      SaturationRegionList = Range("SaturationRegionList")
      SaturationRegionListDbRegionWithTag = Range("SaturationRegionListDbRegionWithTag")

      Dim temP() As Variant

      ReDim temP(UBound(SaturationRegionList, 1))

      irow = 0
      For i = LBound(SaturationRegionList, 1) To UBound(SaturationRegionList, 1)
         thisRegion = SaturationRegionListDbRegionWithTag(i, 1)
         thisSatRegion = SaturationRegionList(i, 1)
         If thisRegion = UserRegion Then
            irow = irow + 1
            temP(irow) = thisSatRegion
         End If
      Next i
      If iUser <= irow Then
```

```
        getAllowedSaturationList = temP(iUser)
      Else
        getAllowedSaturationList = ""
      End If
   End If


End Function

Option Base 1


Function TransformToArray(A_Range)
   ' vba CODE DEVELOPED/IMPLEMENTED BY
   '        Silvia Mazzoni, 2020
   '         smazzoni@ucla.edu
' from: https://kurtverstegen.wordpress.com/2013/12/07/simulation/
' Function to fill Excel range values into an array (matrix)
   Dim r As Integer, c As Integer
   Dim temP()
   If A_Range.Columns.Count = 1 Then
    ReDim temP(A_Range.Rows.Count - 1)     '200317, silviamazzoni: i added the -1 since indices start at zero
    For r = 1 To A_Range.Rows.Count
          temP(r - 1) = A_Range(r, 1)      '200317, silviamazzoni: i added the -1 since indices start at zero
    Next r
   Else
    ReDim temP(A_Range.Rows.Count - 1, A_Range.Columns.Count - 1)    '200317, silviamazzoni: i added the -1 since indices
start at zero
    For r = 1 To A_Range.Rows.Count
     For c = 1 To A_Range.Columns.Count
       temP(r - 1, c - 1) = A_Range(r, c)      '200317, silviamazzoni: i added the -1 since indices start at zero
     Next c
    Next r
   End If
   TransformToArray = temP
End Function

Function interpolateRange(Xpoint, Xlist As Range, Ylist As Range, XinterpType, YinterpType, extrapolateType, Optional
XinterpMin = -1E+16, Optional XinterpMax = 1E+16)
   ' vba CODE DEVELOPED/IMPLEMENTED BY
   '        Silvia Mazzoni, 2020
   '         smazzoni@ucla.edu

   Dim Xnew As Variant
   Dim Ynew As Variant

   Xnew = TransformToArray(Xlist)
   Ynew = TransformToArray(Ylist)
   interpolateRange = interpolateArray(Xpoint, Xnew, Ynew, XinterpType, YinterpType, extrapolateType, XinterpMin,
XinterpMax)

End Function

Function interpolateArray(Xpoint, Xlist, Ylist, XinterpType, YinterpType, extrapolateType, Optional XinterpMin = -1E+16,
Optional XinterpMax = 1E+16)
   ' vba CODE DEVELOPED/IMPLEMENTED BY
   '        Silvia Mazzoni, 2020
   '         smazzoni@ucla.edu
```

```
    interpolateArray = -999
    irow = 0
    start0 = -1
    endrow0 = -1
    For irow = LBound(Xlist) To UBound(Xlist)
       this = Xlist(irow)
       If Xpoint = this Then
          interpolateArray = Ylist(irow)
          Exit Function
       End If
       If this >= XinterpMin And this <= XinterpMax Then
          If start0 < 0 Then
             start0 = irow
          End If
          end0 = irow
       End If
    Next irow
    iend = irow

    istart = start0
    Xstart = Xlist(start0)
    Xend = Xlist(end0)

    If (Xend - Xpoint) * (Xstart - Xpoint) <= 0 Then ' point is within limits
       For i = istart To iend - 1
          thisx = Xlist(i)
          nextX = Xlist(i + 1)
          If (Xpoint - thisx) * (Xpoint - nextX) <= 0 Then  ' point is between these two
             l1 = i
             l2 = i + 1
             GoTo Interp
          End If
       Next i
    Else
       If (Xend - Xstart) * (Xend - Xpoint) > 0 Then
          l1 = istart
          If Left(LCase(extrapolateType), Len("extra")) = LCase("extra") Then
             l2 = istart + 1
          Else
             l2 = istart
          End If
       Else
          l1 = iend
          If Left(LCase(extrapolateType), Len("extra")) = LCase("extra") Then
             l2 = iend - 1
          Else
             interpolateArray = Ylist(end0)
             Exit Function
          End If
       End If
    End If



    'debug. print"a"

Interp:
    x0 = Xlist(l1)
```

```
    x1 = Xlist(l2)
    y0 = Ylist(l1)
    y1 = Ylist(l2)
    hereX = Xpoint
    If LCase(Left(XinterpType, Len("log"))) = LCase("log") Then
        If x0 <= 0 Then
            x0 = 0.000000001
        End If
        If x1 <= 0 Then
            x1 = 0.000000002
        End If
        If hereX <= 0 Then
            hereX = 0.000000001
        End If
        x0 = Log(x0)
        x1 = Log(x1)
        hereX = Log(hereX)
    End If
    If LCase(Left(YinterpType, Len("log"))) = LCase("log") Then
        If y0 <= 0 Then
            y0 = 0.000000001
        End If
        If y1 <= 0 Then
            y1 = 0.000000001
        End If
        y0 = Log(y0)
        y1 = Log(y1)
    End If
    interpolateArray = y0 + (y1 - y0) * (hereX - x0) / (x1 - x0)


    If LCase(Left(YinterpType, Len("log"))) = LCase("log") Then
        interpolateArray = Exp(interpolateArray)
    End If

End Function




Function interpolateFunction(Xpoint, Xlist, XinterpType, extrapolateType, Optional XinterpMin = -1E+16, Optional
XinterpMax = 1E+16)
    ' vba CODE DEVELOPED/IMPLEMENTED BY
    '         Silvia Mazzoni, 2020
    '          smazzoni@ucla.edu
'   ''' Debug.print "interpolateFunction"
    interpolateFunction = -999
    irow = 0
    start0 = -1
    endrow0 = -1
    For irow = LBound(Xlist, 1) To UBound(Xlist, 1)
        this = Xlist(irow, 1)
        If Xpoint = this Then
            interpolateFunction = Array(irow, -1, 0)
            Exit Function
        End If
        If this >= XinterpMin And this <= XinterpMax Then
            If start0 < 0 Then
```

```
            start0 = irow
          End If
          end0 = irow
        End If
    Next irow
    iend = irow - 1

    istart = start0
    Xstart = Xlist(start0, 1)
    Xend = Xlist(end0, 1)

    If (Xend - Xpoint) * (Xstart - Xpoint) <= 0 Then ' point is within limits
        For i = istart To iend - 1
            thisx = Xlist(i, 1)
            nextX = Xlist(i + 1, 1)
            If (Xpoint - thisx) * (Xpoint - nextX) <= 0 Then  ' point is between these two
                l1 = i
                l2 = i + 1
                GoTo Interp
            End If
        Next i
    Else
        If (Xend - Xstart) * (Xend - Xpoint) > 0 Then
            l1 = istart
            If Left(LCase(extrapolateType), Len("extra")) = LCase("extra") Then
                l2 = istart + 1
            Else
                l2 = istart
            End If
        Else
            l1 = iend
            If Left(LCase(extrapolateType), Len("extra")) = LCase("extra") Then
                l2 = iend - 1
            Else
                interpolateFunction = Array(end0, -1, 0)
                Exit Function
            End If
        End If
    End If



    'debug. print"a"

Interp:
    x0 = Xlist(l1, 1)
    x1 = Xlist(l2, 1)
'    y0 = Ylist(l1)
'    y1 = Ylist(l2)
    hereX = Xpoint
    If LCase(Left(XinterpType, Len("log"))) = LCase("log") Then
        If x0 <= 0 Then
            x0 = 0.000000001
        End If
        If x1 <= 0 Then
            x1 = 0.000000002
        End If
        If hereX <= 0 Then
```

```
          hereX = 0.000000001
       End If
       x0 = Log(x0)
       x1 = Log(x1)
       hereX = Log(hereX)
    End If
'    If LCase(Left(YinterpType, Len("log"))) = LCase("log") Then
'       If y0 <= 0 Then
'          y0 = 0.000000001
'       End If
'       If y1 <= 0 Then
'          y1 = 0.000000001
'       End If
'       y0 = Log(y0)
'       y1 = Log(y1)
'    End If
'    interpolateFunction = y0 + (y1 - y0) * (hereX - x0) / (x1 - x0)
    interpolateFunction = Array(l1, l2, (hereX - x0) / (x1 - x0))


'    If LCase(Left(YinterpType, Len("log"))) = LCase("log") Then
'       interpolateFunction = Exp(interpolateFunction)
'    End If
End Function


Option Base 1
Function Log10(x)
    ' vba CODE DEVELOPED/IMPLEMENTED BY
    '         Silvia Mazzoni, 2020
    '          smazzoni@ucla.edu
    Log10 = Log(x) / Log(10#)
End Function
Function Min(a, b)
    ' vba CODE DEVELOPED/IMPLEMENTED BY
    '         Silvia Mazzoni, 2020
    '          smazzoni@ucla.edu
    Min = a
    If b < a Then
       Min = b
    End If
End Function

Function Max(a, b)
     ' vba CODE DEVELOPED/IMPLEMENTED BY
    '         Silvia Mazzoni, 2020
    '          smazzoni@ucla.edu
   Max = a
    If b > a Then
       Max = b
    End If
End Function


Function getTFvalue(inValue)
    ' vba CODE DEVELOPED/IMPLEMENTED BY
    '         Silvia Mazzoni, 2020
    '          smazzoni@ucla.edu
```

```
  ' Code Written and developed by Silvia Mazzoni, smazzoni@ucla.edu, March 2020
  If VarType(inValue) = vbString Then
    If LCase(inValue) = LCase("pga") Then
      getTFvalue = 0
    ElseIf LCase(inValue) = LCase("pgv") Then
      getTFvalue = -1
    End If
  Else
    getTFvalue = inValue
  End If
End Function
```

5.2 NGA-Sub Ground-Motion Characterization Tools – "R"

NGASubduction_GMMtool_R_ r201031.R

```
### ################################################################
### ################################################################
### ----------------------------------------------------------------
### ----------------------------------------------------------------
### -- NGA-Subduction:
###              Ground-Motion Characterization Tool
###              R Version
### ----------------------------------------------------------------
###                        October 2020
### ----------------------------------------------------------------
### Ground-Motion Model Tool developed by
###          Silvia Mazzoni
###             smazzoni@ucla.edu
###          B. John Garrick Institute for the Risk Sciences
###             https://www.risksciences.ucla.edu/nhr3/gmtools
### ----------------------------------------------------------------
### Sources:
###  KBCG20:
###    Kuehn, Bozorgnia, Campbell, and Gregor,
###        "Partially Nonergodic Ground-Motion Model for Subduction Regions using NGA-Subduction Database,"
###        Report 4/2020, Pacific Earthquake Engineering Research Center, UC Berkeley
###  PSHAB20:
###    Parker, G.A., Stewart, J.P., Hassani, B., Atkinson, G.M., and Boore, D.M. (2020).
###        "NGA-Subduction Global Ground Motion Models with Regional Adjustment Factors."
###        Report xx/2020, Pacific Earthquake Engineering Research Center, UC Berkeley
### ----------------------------------------------------------------
### Instructions:
###   1. Enter the values of the user-input parameters.
###      Input ranges are given with each input parameters.
###   2. Run full script
###   3. Wait while script runs
###   4. Result values are printed in Command Window (you may copy and paste them elsewhere)
### ----------------------------------------------------------------
###   Notes:
###      The Median and Sigma models are described in the reports.
###      Damping Ratio = 5#
###        You may speed up the program by managing the coefficients
###        tables more efficiently.
###      The program computes the geometric mean of the median ground-motion models,
###          and the SRSS of the aleatory variability and epistemic uncertainty.
###
###
### ################################################################
### ################################################################
# Clear all data in memory
rm(list=ls(all=TRUE))
library(MASS)

# ################################################################
# ################################################################
# ################################################################
############ USER INPUT :
# ################################################################
User_Region <- '0_global';   # Region (Options: 0_global, 1_Alaska, 2_Cascadia, 3_CentralAmerica&Mexico, 4_Japan,
5_NewZealand (PSHAB20 uses global model), 6_SouthAmerica, 7_Taiwan)
```

```
User_SubductionSlab <- 'global';   # Subducting Slab (used to compute Mb)
(Alaska,Aleutian,Cascadia,Central_America_N,Central_America_S,global,Japan_Pac,Japan_Phi,New_Zealand_N,New_Zealan
d_S,South_America_N,South_America_S,Taiwan_E,Taiwan_W)
User_Magnitude <- 8;   # Moment Magnitude
User_Vs30 <- 760;   # VS30 (m/sec)
User_Rrup <- 200;   # Rupture Distance (km)
User_AlphaBackarc <- 0;   # Fraction of Rrup in Backarc (Range: 0-1)
User_AlphaNankai <- 0;   # Fraction of Rrup in Nankai Region (Range: 0-(1-AlphaBackArc))(Japan only)
User_Ztor <- 10;   # Z_tor (km) (KBCG20 only)
User_Zhypo <- 55;   # Hypocentral Depth (km) (PSHAB20 only)
User_EventType <- '0_Interface';   # Event Type, (Options: 0_Interface, 1_Intraslab)
User_Z1pt0 <- 550;   # Z1.0 input (m) (KBCG20 only)
User_Z2pt5 <- 2000;   # Z2.5 input (m)
User_Mb <- 7.9;   # Mb (KBCG20 only, can set = default)
User_PNWbasinStrux <- '0_NoBasin';   # PNW Basin Structure (Options:
0_NoBasin,1_InSeattleBasin,2_InOtherPNWbasin)(Cascadia Only)
User_RelativeWeight_KBCG20 <- 1;   # Relative Weight -- KBCG20
User_RelativeWeight_PSHAB20 <- 0;   # Relative Weight -- PSHAB20
User_EpiInSigmaModels <- '1_Yes';   # Apply KBCG20 Epistemic to Sigma Models (Options: 0_None, 1_AllModels)
User_NsampleEpi <- 100;   # Number of Samples in Epistemic-Uncertainty Calculation (0=none, Range: 100-800)
User_Nsigma <- 1;   # Number of Sigma away from Median


User_PeriodList <- c(0.01,0.02,0.03,0.05,0.075,0.1,0.15,0.2,0.25,0.3,0.4,0.5,0.75,1,1.5,2,3,4,5,7.5,10,"PGA","PGV") #
range:0.01-10sec (PGA: T=0, PGV: T=-1)

# The following is an optional script that performs a large sensitivity study
# Running this sensitivity study make take a few minutes.
RunSensitivityStudyFlag <- 'yes'            # options: 'yes' 'no'




##############################################################################################
##############################################################################################
##############################################################################################
# program DO NOT edit lines below!!!
##############################################################################################
#########################################################
################ PROGRAM DATA ###########################
#########################################################




getTFvalue <-function(inValue) {
  # CODE DEVELOPED/IMPLEMENTED BY
  #         Silvia Mazzoni, 2020
  #          smazzoni@ucla.edu

  # Code Written and developed by Silvia Mazzoni, smazzoni@ucla.edu, April 2020
  if(is.character(inValue)) {
    if(tolower(inValue) == tolower("pga") ) {
      getTFvalueValue <- 0
    } else if(tolower(inValue) == tolower("pgv") ) {
      getTFvalueValue <- -1
            } else {
```

```
                    getTFvalueValue <- as.numeric(inValue)
                  }
   } else {
      getTFvalueValue <- inValue
   }
   getTFvalueValue <- as.numeric(getTFvalueValue)
   return(getTFvalueValue)
}









# 4.1 Helper Functions
# First, some functions that are needed to do some internal calculations.

# logistc hinge function
loghinge <- function(x, x0, a, b0, b1, delta) {
  return(a + b0 * (x - x0) + (b1 - b0) * delta * log(1 + exp((x - x0)/delta)))
}

# interpolation of adjustment to magnitude break point
interp_dmb <- function(period) {
  ap <- approxfun(c(log(0.01),log(1),log(3),log(10)), c(0,0,-0.4,-0.4), rule = 2)

  if(period == 0) {
    return (ap(log(0.01)))
  } else if(period == -1  ) {
    return(0)
  } else {
    return (ap(log(period)))
  }
}

# interpolation of k1/k2 (values taken from Campbell and Bozorgnia (2014)
interp_k1k2 <- function(period) {
  periods <- c(0.005, 0.01, 0.02, 0.03, 0.05, 0.075, 0.1, 0.15, 0.2, 0.25, 0.3, 0.4, 0.5, 0.75,
            1., 1.5, 2., 3., 4., 5., 7.5, 10.)
  k1 <- c(865., 865., 865., 908., 1054., 1086., 1032., 878., 748., 654., 587., 503., 457., 410.,
         400., 400., 400., 400., 400., 400., 400., 400.)
  k2 <- c(-1.186, -1.186, -1.219, -1.273, -1.346, -1.471, -1.624, -1.931, -2.188, -2.381, -2.518,
         -2.657, -2.669, -2.401, -1.955, -1.025, -0.299, 0., 0., 0., 0., 0.)

  ap_k1 <- approxfun(log(periods), k1, rule = 2)
  ap_k2 <- approxfun(log(periods), k2, rule = 2)

  if(period == 0) {
    return (c(ap_k1(log(0.005)),ap_k2(log(0.005))))
  } else if(period == -1  ) {
     ap_k1 <- 400.0
     ap_k2 <- -1.995
    return (c(ap_k1,ap_k2))
  }  else {
    return (c(ap_k1(log(period)),ap_k2(log(period))))
  }
}
```

```
# funcation to calculate Z1pt0/Z2pt5 from Vs30
calc_z_from_vs <- function(Vs30, coeffs) {
  zref <- coeffs[1] + (coeffs[2] - coeffs[1]) * exp((log(Vs30) - coeffs[3]) / coeffs[4]) /
    (1 + exp((log(Vs30) - coeffs[3]) / coeffs[4]))

  return (zref)
}



##          4.2 Function to calculate median prediction
##          The following function calculates the median prediction of KBCG20, as described in chapter 4 of the PEER report. It takes as input the predictor variables, as well as a set of coefficients. Later, we define a function that acts as a wrapper around this function, and will take as input period, region index, and then select the appropriate coefficients and pass them on.
##          The inputs are
##          M: moment magnitude
##          rlist: a vector of length 3, which contains the distance in subregion 1,2,3 relative to volcanic arc rlist <- c(R1,R2,R3). Typically, R1=R3=0 (corresponding to forearc).
##          Ztor: depth to top of rupture in km.
##          fs: flag for interface (fs <- 0) and intraslab (fs <- 1). Must be 0 or 1.
##          Vs: Vs30 n M/s.
##          fx: arc crossing flag. Must be 0 or 1.
##          delta_ln_z: difference between natural log of observed Z1.0/Z2pt5 value and reference Z1.0/Z2pt5 from Vs30.
##          coeffs: vector containing coefficients needed to calculate median prediction.
##          coeffs_attn: vector of length 6 to calcualte anelastic attenuation.
##          mbreak and zbreak: magnitude and depth scaling break point
##          k1 and k2: parameters needed for site amplification
##          nft1 and nft2: coefficients needed for pseudo-depth term.
##          pgarock: median pga prediction at Vs30=1100

KBCG20_med <-function(M, Rrup, AlphaBackarc, AlphaNankai, Ztor, EventType, Vs30, delta_ln_z, coeffs, coeffs_attn, coeffs_z, mbreak, zbreak, k1, k2, nft1, nft2, pgarock, UserRegion) {
    # CODE DEVELOPED/IMPLEMENTED BY
   #        Silvia Mazzoni, 2020
   #        smazzoni@ucla.edu
  reg <- getRegionKBCG(UserRegion)
  if(reg == 1 ) {
      distR1 <- 0
      distR3 <- 0
  } else if(reg == 2  ) {
      distR1 <- 0
      distR3 <- 0
  } else if(reg == 3  ) {
      distR1 <- AlphaBackarc * Rrup
      distR3 <- 0
  } else if(reg == 4  ) {
      distR1 <- AlphaBackarc * Rrup
      distR3 <- AlphaNankai * Rrup
  } else if(reg == 5  ) {
      distR1 <- 0
      distR3 <- 0
  } else if(reg == 6  ) {
      distR1 <- AlphaBackarc * Rrup
      distR3 <- 0
  } else if(reg == 7  ) {
      distR1 <- 0
```

```
    distR3 <- 0
 } else {  # reg=0
    distR1 <- AlphaBackarc * Rrup
    distR3 <- 0
 }

  distR2 <- Rrup - distR1 - distR3



theta10 <- 0
vsrock <- 1100
c <- 1.88
n <- 1.18
minmb <- 6.
delta <- 0.1
deltaz <- 1
refzif<- 15
refzslab <- 50
fs <- (LeftNumber(EventType, 1))

  #check if(cross arc:
    fx <- 1
    if(distR1 == 0 && distR2 == 0 ) {
       fx <- 0
    }
    if(distR2 == 0 && distR3 == 0  ) {
       fx <- 0
    }
    if(distR1 == 0 && distR3 == 0  ) {
       fx <- 0
    }

fmag <- (1 - fs) * loghinge(M, mbreak, coeffs[6] * (mbreak - minmb), coeffs[6], coeffs[8], delta) +
     fs * loghinge(M, mbreak, coeffs[7] * (mbreak - minmb), coeffs[7], coeffs[8], delta)

fgeom <- (1 - fs) * (coeffs[3] + coeffs[5] * M) * log(Rrup + 10^(nft1 + nft2 * (M - 6)))
fgeom_slab <- fs * (coeffs[4] + coeffs[5] * M) * log(Rrup + 10^(nft1 + nft2 * (M - 6)))

fdepth <- (1 - fs) * loghinge(Ztor, zbreak, coeffs[12] * (zbreak - refzif), coeffs[12], theta10, deltaz) +
       fs * loghinge(Ztor, zbreak, coeffs[13] * (zbreak - refzslab), coeffs[13], theta10, deltaz)

 DotProduct123 <- distR1 * coeffs_attn[1] + distR2 * coeffs_attn[2] + distR3 * coeffs_attn[3]
 DotProduct456 <- distR1 * coeffs_attn[4] + distR2 * coeffs_attn[5] + distR3 * coeffs_attn[6]
 fattn <- fx * DotProduct123 + (1 - fx) * DotProduct456 + fx * coeffs[14]

if(Vs30 < k1) {
  fsite <- coeffs[11] * log(Vs30 / k1) + k2 * (log(pgarock + c * (Vs30 / k1)^n) - log(pgarock + c))
} else {
  fsite <- (coeffs[11] + k2 * n) * log(Vs30/k1)
}

fbasin <- coeffs_z[1] + coeffs_z[2] * delta_ln_z


median <- (1 - fs) * coeffs[1] + fs * coeffs[2] + fmag + fgeom + fgeom_slab + fdepth + fattn + fsite + fbasin
return(median)
```

```
}


##          4.3 Function to calculate median prediction for a given scenario and period
##          This is a function that calculates median predictions of KBCG20 for a given scenario. I takes as input period, the predictor variables for the scenarios, selects the appropriate coefficients/parameters, and calls the function defined in the previous section.
##          The arguments are similar to the function before.
##          M: moment magnitude
##          rlist: a vector of length 3, which contains the distances in subregion 1,2,3 relative to volcanic arc rlist <- c(R1,R2,R3). For regions Alaska, Cascadia, New Zealand, and Taiwan, R1=R2=0R1=R2=0.
##          Ztor: depth to top of rupture in km.
##          fs: flag for interface (fs <- 0) and intraslab (fs <- 1). Must be 0 or 1.
##          Vs: Vs30 n M/s.
##          Z1pt0: depth to a shear wave horizon of 1000 M/s, in M. Used for regions Alaska and New Zealand.
##          Z2pt5: depth to a shear wave horizon of 2500 M/s, in M. Used for regions Cascadia and Japan.
##          fx: arc crossing flag. Must be 0 or 1.
##          Mb: Magnitude scaling break point. Should be set regionally dependent based on Campbell (2020).
##          The last input is a region index reg, which is as follows:
##          0: global
##          1: Alaska
##          2: Cascadia
##          3: Central America & Mexico
##          4: Japan
##          5: New Zealand
##          6: South America
##          7: Taiwan
##          The opional argumen Seattle_Basin is a flag that should be set to TRUE if(the site is in the Seattle Basin, and FALSE otherwise. This flag determines which basin depth amplification model is used for Cascadia. It does not have an impact on any other region.



KBCG20_medPSA <-function(period, Magnitude, Rrup, AlphaBackarc, AlphaNankai, Ztor, EventType, Vs30, Z1pt0, Z2pt5, Mb, UserRegion, PNWbasinStrux) {
  # CODE DEVELOPED/IMPLEMENTED BY
  #       Silvia Mazzoni, 2020
  #       smazzoni@ucla.edu

  XinterpMin <- 0
  XinterpMax <- 10
  XinterpType <- "log"
  YinterpType <- "log"
  extrapolateType <- "extrapolate"

  Parameters <- read.csv('NGAsubGMM_KBCG20_coefficients.csv')
  TvalueList <- Parameters$T
  InterpArray <- interpolateFunction(period, TvalueList, XinterpType, extrapolateType, XinterpMin, XinterpMax)

  period0 <- TvalueList[InterpArray[1]]
  y0 <- KBCG20_medPSA_AtTlist(period0, Magnitude, Rrup, AlphaBackarc, AlphaNankai, Ztor, EventType, Vs30, Z1pt0, Z2pt5, Mb, UserRegion, PNWbasinStrux)

  if(InterpArray[2] <= 0  ) {
    thisOut <- y0
  } else {
    period1 <- TvalueList[InterpArray[2]]
```

```
      y1 <- KBCG20_medPSA_AtTlist(period1, Magnitude, Rrup, AlphaBackarc, AlphaNankai, Ztor, EventType, Vs30, Z1pt0,
Z2pt5, Mb, UserRegion, PNWbasinStrux)
      if(tolower(LeftString(YinterpType, nchar("log"))) == tolower("log") ) {
         if(y0 <= 0  ) {
            y0 <- 0.000000001
         }
         if(y1 <= 0  ) {
            y1 <- 0.000000001
         }
         y0 <- log(y0)
         y1 <- log(y1)
      }
      thisOut <- y0 + (y1 - y0) * InterpArray[3]
      if(tolower(LeftString(YinterpType, nchar("log"))) == tolower("log")  ) {
         thisOut <- exp(thisOut)
      }

   }
   return(thisOut)

}


## function to calculate median prediction using mean coefficients
KBCG20_medPSA_AtTlist <-function(period, M, Rrup, AlphaBackarc, AlphaNankai, Ztor, EventType, Vs30, Z1pt0, Z2pt5, Mb,
UserRegion, PNWbasinStrux) {
   # CODE DEVELOPED/IMPLEMENTED BY
   #         Silvia Mazzoni, 2020
   #          smazzoni@ucla.edu
 # need to add some checks for input (period, UserRegion)

 # coefficients to calculate zref from Vs30
 pars_z_ja <- c(7.689368537500001, 2.302585092994046, 6.309186400000001, 0.7528670225000001, 1.2952369625)
 pars_z_casc <- c(8.294049640102028, 2.302585092994046, 6.396929655216146, 0.27081458999999997, 1.7381352625)
 pars_z_nz <- c(6.859789675000001, 2.302585092994046, 5.745692775, 0.91563524375, 1.03531412375)
 pars_z_tw <- c(6.30560665, 2.302585092994046, 6.1104992125, 0.43671101999999995, 0.7229702975000001)

 parameters <- read.csv('NGAsubGMM_KBCG20_coefficients.csv')
 parameters_zmod <- read.csv('NGAsubGMM_KBCG20_params_Z_ALL_allregca_attn3_corrreg_cs_dmb.csv')

 # calculate rock PGA
 period_used <- 0.
 vsrock <- 1100
 pars_period <- as.numeric(parameters[parameters$T == period_used,])

 coeffs_pga <- pars_period[seq(2,15)]
 k1k2 <- interp_k1k2(period_used)
 dmb <- 0


 delta_ln_z <- 0
 coeffs_z_pga <- c(0,0)
 coeffs_z_pga2 <- c(0,0)
 reg <- getRegionKBCG(UserRegion)

 if(reg == 1) {
  coeffs_pga[c(1,2,11)] <- pars_period[c(17, 24, 31)]
  coeffs_attn_pga <- pars_period[c(38, 45, 52, 59, 66, 73)]
```

```
  } else if(reg == 2) {
   coeffs_pga[c(1,2,11)] <- pars_period[c(18, 25, 32)]
   coeffs_attn_pga <- pars_period[c(39, 46, 53, 60, 67, 74)]
   if(LeftNumber(PNWbasinStrux, 1) == 1) {
    coeff_seattle <- parameters_zmod[parameters_zmod$T == period_used,]$mean_residual_Seattle_basin
    coeffs_z_pga2 <- c(coeff_seattle,0)
   } else if(LeftNumber(PNWbasinStrux, 1) == 2) {
    coeffs_z_pga2 <- as.numeric(parameters_zmod[parameters_zmod$T == period_used,c(2,3)])
   }
  } else if(reg == 3) {
   coeffs_pga[c(1,2,11)] <- pars_period[c(19, 26, 33)]
   coeffs_attn_pga <- pars_period[c(40, 47, 54, 61, 68, 75)]
  } else if(reg == 4) {
            dmb <- interp_dmb(period_used)
   coeffs_pga[c(1,2,11)] <- pars_period[c(20, 27, 34)]
   coeffs_attn_pga <- pars_period[c(41, 48, 55, 62, 69, 76)]
   coeffs_z_pga2 <- as.numeric(parameters_zmod[parameters_zmod$T == period_used,c(5,6)])
  } else if(reg == 5) {
   coeffs_pga[c(1,2,11)] <- pars_period[c(21, 28, 35)]
   coeffs_attn_pga <- pars_period[c(42, 49, 56, 63, 70, 77)]
   coeffs_z_pga2 <- as.numeric(parameters_zmod[parameters_zmod$T == period_used,c(7,8)])
  } else if(reg == 6) {
            dmb <- interp_dmb(period_used)
   coeffs_pga[c(1,2,11)] <- pars_period[c(22, 29, 36)]
   coeffs_attn_pga <- pars_period[c(43, 50, 57, 64, 71, 78)]
  } else if(reg == 7) {
   coeffs_pga[c(1,2,11)] <- pars_period[c(23, 30, 37)]
   coeffs_attn_pga <- pars_period[c(44, 51, 58, 65, 72, 79)]
   coeffs_z_pga2 <- as.numeric(parameters_zmod[parameters_zmod$T == period_used,c(9,10)])
  }
  else if(reg == 0) {
   coeffs_attn_pga <- pars_period[c(11, 11, 11, 11, 10, 11)]
  }


  delta_bz <- pars_period[c(80,81)]
  coeffs_nft_pga <- pars_period[c(82,83)]

  fs <- (LeftNumber(EventType, 1))

  mbreak_pga <- (1 - fs) * (Mb + dmb) + fs * Mb
  zbreak_pga <- (1 - fs) * (30 + delta_bz[1]) + fs * (80 + delta_bz[2])
  k1k2_pga <- k1k2
  pgarock <- exp(KBCG20_med(M, Rrup, AlphaBackarc, AlphaNankai, Ztor, EventType, vsrock, delta_ln_z, coeffs_pga,
coeffs_attn_pga, coeffs_z_pga, mbreak_pga, zbreak_pga, k1k2_pga[1], k1k2_pga[2], coeffs_nft_pga[1], coeffs_nft_pga[2], 0,
UserRegion))


  # calculate PSA
  period_used <- period
  pars_period <- as.numeric(parameters[parameters$T == period_used,])
  coeffs <- pars_period[seq(2,15)]
  k1k2 <- interp_k1k2(period_used)
  dmb <- 0

  delta_ln_z <- 0
  coeffs_z <- c(0,0)
  reg <- getRegionKBCG(UserRegion)
```

```
if(reg == 1) {
  coeffs[c(1,2,11)] <- pars_period[c(17, 24, 31)]
  coeffs_attn <- pars_period[c(38, 45, 52, 59, 66, 73)]
} else if(reg == 2) {
  coeffs[c(1,2,11)] <- pars_period[c(18, 25, 32)]
  coeffs_attn <- pars_period[c(39, 46, 53, 60, 67, 74)]
              if(tolower(Z2pt5) == 'default' || Z2pt5 <0) {
                delta_ln_z <- 0
              } else {
                delta_ln_z <- log(Z2pt5) - calc_z_from_vs(Vs30, pars_z_casc)
              }
  if(LeftNumber(PNWbasinStrux, 1) == 1) {
    coeff_seattle <- parameters_zmod[parameters_zmod$T == period_used,]$mean_residual_Seattle_basin
    coeffs_z <- c(coeff_seattle,0)
  } else if(LeftNumber(PNWbasinStrux, 1) == 2) {
    coeffs_z <- as.numeric(parameters_zmod[parameters_zmod$T == period_used,c(2,3)])
  }
} else if(reg == 3) {
  coeffs[c(1,2,11)] <- pars_period[c(19, 26, 33)]
  coeffs_attn <- pars_period[c(40, 47, 54, 61, 68, 75)]
} else if(reg == 4) {
  dmb <- interp_dmb(period_used)
  coeffs[c(1,2,11)] <- pars_period[c(20, 27, 34)]
  coeffs_attn <- pars_period[c(41, 48, 55, 62, 69, 76)]
              if(tolower(Z2pt5) == 'default' || Z2pt5 <0) {
                delta_ln_z <- 0
              } else {
             delta_ln_z <- log(Z2pt5) - calc_z_from_vs(Vs30, pars_z_ja)
              }
  coeffs_z <- as.numeric(parameters_zmod[parameters_zmod$T == period_used,c(5,6)])
} else if(reg == 5) {
  coeffs[c(1,2,11)] <- pars_period[c(21, 28, 35)]
  coeffs_attn <- pars_period[c(42, 49, 56, 63, 70, 77)]
              if(tolower(Z1pt0) == "default" || Z1pt0 <0) {
                delta_ln_z <- 0
              } else {
                delta_ln_z <- log(Z1pt0) - calc_z_from_vs(Vs30, pars_z_nz)
              }
  coeffs_z <- as.numeric(parameters_zmod[parameters_zmod$T == period_used,c(7,8)])
} else if(reg == 6) {
  dmb <- interp_dmb(period_used)
  coeffs[c(1,2,11)] <- pars_period[c(22, 29, 36)]
  coeffs_attn <- pars_period[c(43, 50, 57, 64, 71, 78)]
} else if(reg == 7) {
  coeffs[c(1,2,11)] <- pars_period[c(23, 30, 37)]
  coeffs_attn <- pars_period[c(44, 51, 58, 65, 72, 79)]
              if(tolower(Z1pt0) == "default" || Z1pt0 <0) {
                delta_ln_z <- 0
              } else {
                delta_ln_z <- log(Z1pt0) - calc_z_from_vs(Vs30, pars_z_tw)
              }
  coeffs_z <- as.numeric(parameters_zmod[parameters_zmod$T == period_used,c(9,10)])
}
else if(reg == 0) {
  coeffs_attn <- pars_period[c(11, 11, 11, 11, 10, 11)]
}
```

```
  delta_bz <- pars_period[c(80,81)]
  coeffs_nft <- pars_period[c(82,83)]

  mbreak <- (1 - fs) * (Mb + dmb) + fs * Mb
  zbreak <- (1 - fs) * (30 + delta_bz[1]) + fs * (80 + delta_bz[2])

  med <- KBCG20_med(M, Rrup, AlphaBackarc, AlphaNankai, Ztor, EventType, Vs30, delta_ln_z, coeffs, coeffs_attn, coeffs_z,
mbreak, zbreak, k1k2[1], k1k2[2], coeffs_nft[1], coeffs_nft[2], pgarock, UserRegion)
  med_pga <- KBCG20_med(M, Rrup, AlphaBackarc, AlphaNankai, Ztor, EventType, Vs30, delta_ln_z, coeffs_pga,
coeffs_attn_pga, coeffs_z_pga2, mbreak_pga, zbreak_pga, k1k2_pga[1], k1k2_pga[2], coeffs_nft_pga[1], coeffs_nft_pga[2],
pgarock, UserRegion)

  if(med < med_pga && period <= 0.1) {
    med <- med_pga
  }
  return(exp(med))

}

getRangeRowT <-function(thisRange, Xvalue) {
   # CODE DEVELOPED/IMPLEMENTED BY
   #         Silvia Mazzoni, 2020
   #          smazzoni@ucla.edu
              pars_period <- as.numeric(thisRange[thisRange$T == Xvalue,])
              return(pars_period)
}


getSubArrayRange <-function(inArray, startIndex, endIndex) {
   # CODE DEVELOPED/IMPLEMENTED BY
   #         Silvia Mazzoni, 2020
   #          smazzoni@ucla.edu


   irow <- 1
   for (thisIndex in startIndex : endIndex ){
      temP[irow] <- inArray(thisIndex)
      irow <- irow + 1
   }
   getSubArrayRangeValue <- temP
}

interpolateArray <-function(Xpoint, Xlist, Ylist, XinterpType, YinterpType, extrapolateType,  XinterpMin = -1e16 ,XinterpMax
= +1e16) {
   # CODE DEVELOPED/IMPLEMENTED BY
   #         Silvia Mazzoni, 2020
   #          smazzoni@ucla.edu

   Xlist <- simplify2array(Xlist)

   start0 <- -1
   nrows <- NROW(Xlist)
   ncols <- NCOL(Xlist)
      if(nrows == 1 && ncols>0 ) {
         t(Xlist)
      }
                  nrows <- NROW(Ylist)
                  ncols <- NCOL(Ylist)
```

```
    if(nrows == 1 && ncols>0 ) {
        t(Ylist)
    }
                nrows <- NROW(Xlist)
                ncols <- NCOL(Xlist)

  for (irow in 1 : nrows ){
    this <- Xlist[irow]
    if(Xpoint == this  ) {
      interpolateArrayValue <- Ylist[irow]
      return(interpolateArrayValue)
    }
    if(this >= XinterpMin && this <= XinterpMax  ) {
      if(start0 < 0  ) {
        start0 <- irow
      }
      end0 <- irow
    }
  }
 iend <- irow
  istart <- start0
  Xstart <- Xlist[start0]
  Xend <- Xlist[end0]
  if((Xend - Xpoint) * (Xstart - Xpoint) <= 0) {  # point is within limits
    for (i in istart : iend - 1 ){
      thisX <- Xlist[i]
      nextX <- Xlist[i + 1]
      if((Xpoint - thisX) * (Xpoint - nextX) <= 0 ){  # point is between these two
        l1 <- i
        l2 <- i + 1
        break
      }
    }
  } else {
    if((Xend - Xstart) * (Xend - Xpoint) > 0 ) {
      l1 <- istart
      if(LeftString(tolower(extrapolateType), nchar("extra")) == tolower("extra") ) {
        l2 <- istart + 1
      } else {
        l2 <- istart
      }
    } else {
      l1 <- iend
      if(LeftString(tolower(extrapolateType), nchar("extra")) == tolower("extra") ) {
        l2 <- iend - 1
      } else {
        interpolateArrayValue <- Ylist(end0)
        return(interpolateArrayValue)
      }
    }
  }

  x0 <- Xlist[l1]
  x1 <- Xlist[l2]
  y0 <- Ylist[l1]
  y1 <- Ylist[l2]
  hereX <- Xpoint
  if(tolower(LeftString(XinterpType, nchar("log"))) == tolower("log")) {
```

```
      if(x0 <= 0 ) {
         x0 <- 0.000000001
      }
      if(x1 <= 0 ) {
         x1 <- 0.000000002
      }
      if(hereX <= 0 ) {
         hereX <- 0.000000001
      }
      x0 <- log(x0)
      x1 <- log(x1)
      hereX <- log(hereX)
   }
   if(tolower(LeftString(YinterpType, nchar("log"))) == tolower("log") ) {
      if(y0 <= 0 ) {
         y0 <- 0.000000001
      }
      if(y1 <= 0 ) {
         y1 <- 0.000000001
      }
      y0 <- log(y0)
      y1 <- log(y1)
   }
   interpolateArrayValue <- y0 + (y1 - y0) * (hereX - x0) / (x1 - x0)


   if(tolower(LeftString(YinterpType, nchar("log"))) == tolower("log") ) {
      interpolateArrayValue <- exp(interpolateArrayValue)
   }
}

LeftString <-function(instring,ncars) {
   inn <-  as.character(instring)
   LeftStringValue <- substr(inn,1,ncars)
}
LeftNumber <-function(instring,ncars) {
   inn <-  as.character(instring)
   LeftNumberValue<-as.integer(substr(inn,1,ncars))
}


makeArray <-function(oldArray, oldArrayIndices) {
   # CODE DEVELOPED/IMPLEMENTED BY
   #         Silvia Mazzoni, 2020
   #          smazzoni@ucla.edu


   newIndex <- 1
                nrow <- NROW(oldArrayIndices)
                ncol <- NCOL(oldArrayIndices)
                nrowOldArray <- NROW(oldArrayIndices)
                ncolOldArray <- NCOL(oldArrayIndices)
                ############silvia

   nrowMax <- nrow
   if(ncol>nrow) {
      nrowMax <- ncol
   }
```

```
  for (irow in 1 : nrowMax ){
     if(nrow==1 && ncol> 1 ) {
        oldIndex <- oldArrayIndices(1,irow)
     } else {
        oldIndex <- oldArrayIndices(irow,1)
     }

     if(nrowOldArray==1 && ncolOldArray > 1 ) {
        newArray(newIndex) <- oldArray(1,oldIndex)
     } else {
        newArray(newIndex) <- oldArray(oldIndex,1)
     }
     newIndex <- newIndex + 1
  }
   makeArrayValue <- newArray
}

updateArray <-function(newArray, newArrayIndices, oldArray, oldArrayIndices) {
   # CODE DEVELOPED/IMPLEMENTED BY
   #         Silvia Mazzoni, 2020
   #          smazzoni@ucla.edu


                Nrow <- NROW(oldArrayIndices)
                Ncol <- NCOL(oldArrayIndices)
     if(Nrow==1 && Ncol>1 ){
#         oldArrayIndices <- oldArrayIndices.'
        t(oldArrayIndices)
     }
                NN <- NROW(oldArrayIndices)
                Ncol <- NCOL(oldArrayIndices)
   for (irow in 1 : NN ){
     oldIndex <- oldArrayIndices[irow]
     newIndex <- newArrayIndices[irow]

     newArray(newIndex) <- oldArray(oldIndex)
   }
   updateArrayValue <- newArray
}


interpolateFunction <-function(Xpoint, Xlist, XinterpType, extrapolateType, XinterpMin = -1e16 ,XinterpMax = +1e16 ) {
   # CODE DEVELOPED/IMPLEMENTED BY
   #         Silvia Mazzoni, 2020
   #          smazzoni@ucla.edu

   Xlist <- simplify2array(Xlist)

   interpolatefunctionValue <- -999
   irow <- 0
   start0 <- -1
   endrow0 <- -1
   NN <-NROW(Xlist)
   for (irow in 1 : NN ){
     this <- Xlist[irow]
     if(Xpoint == this ) {
        interpolatefunctionValue <- c(irow, -1, 0)
        return(interpolatefunctionValue)
```

```
      }
      if(this >= XinterpMin && this <= XinterpMax ) {
         if(start0 < 0 ) {
            start0 <- irow
         }
         end0 <- irow
      }
   }
   iend <- irow - 1
   istart <- start0
   Xstart <- Xlist[start0]
   Xend <- Xlist[end0]

   if((Xend - Xpoint) * (Xstart - Xpoint) <= 0 ) { # point is within limits
      for (i in istart : end0 - 1 ){
         thisX <- Xlist[i]

         nextX <- Xlist[i + 1]
         if((Xpoint - thisX) * (Xpoint - nextX) <= 0 ) {  # point is between these two
            l1 <- i
            l2 <- i + 1
            break
         }
      }
   } else {
      if((Xend - Xstart) * (Xend - Xpoint) > 0 ) {
         l1 <- istart
         if(LeftString(tolower(extrapolateType), nchar("extra")) == tolower("extra") ) {
            l2 <- istart + 1
         } else {
            l2 <- istart
         }
      } else {
         l1 <- iend
         if(LeftString(tolower(extrapolateType), nchar("extra")) == tolower("extra") ) {
            l2 <- iend - 1
         } else {
            interpolatefunctionValue <- c(end0, -1, 0)
            return(interpolatefunctionValue)
         }
      }
   }



#Interp:
   x0 <- Xlist[l1]
   x1 <- Xlist[l2]

   hereX <- Xpoint
   if(tolower(LeftString(XinterpType, nchar("log"))) == tolower("log") ) {
      if(x0 <= 0 ) {
         x0 <- 0.000000001
      }
      if(x1 <= 0 ) {
         x1 <- 0.000000002
      }
      if(hereX <= 0 ) {
```

```
        hereX <- 0.000000001
      }
      x0 <- log(x0)
      x1 <- log(x1)
      hereX <- log(hereX)
    }
    interpolatefunctionValue <- c(l1, l2, (hereX - x0) / (x1 - x0))
    return(interpolatefunctionValue)

}

##          4.4 Function to calculate distribution of median predictions (for epistemic uncertainty)
##          This function is similar as the one in the previous section, but reads in posterior distribution, and loops over samples from posterior. The number of samples can be set, with 800 as default. Output is mean/median/5#,25#,75#,95# quntiles of distribution, and the median predictions themselves.
##          Two functions are defined. The first calculates predictions for the global model using presampled adjustment coefficients for a new region, while the second one samples these internally The second function is implemented to demonstrate how to perform the sampling, (how to calculate the covariance matrix from the `output of the cholesky decomposition). To get reproducible results, one should use the function with presampled global coefficients.

# function to calculate median prediction using posterior distributions
KBCG20_posteriorAtTlist <-function(period, M, Rrup, AlphaBackarc, AlphaNankai, Ztor, EventType, Vs30, Z1pt0, Z2pt5, Mb, UserRegion, num_samples, PNWbasinStrux ) {
  # CODE DEVELOPED/IMPLEMENTED BY
  #        Silvia Mazzoni, 2020
  #         smazzoni@ucla.edu
 # need to add some checks for input (period, UserRegion)

 # coefficients to calculate zref from Vs30
 pars_z_ja <- c(7.689368537500001, 2.302585092994046, 6.309186400000001, 0.7528670225000001, 1.2952369625)
 pars_z_casc <- c(8.294049640102028, 2.302585092994046, 6.396929655216146, 0.27081458999999997, 1.7381352625)
 pars_z_nz <- c(6.859789675000001, 2.302585092994046, 5.745692775, 0.91563524375, 1.03531412375)
 pars_z_tw <- c(6.30560665, 2.302585092994046, 6.1104992125, 0.43671101999999995, 0.7229702975000001)

 parameters <- read.csv('NGAsubGMM_KBCG20_coefficients.csv')
 parameters_zmod <- read.csv('NGAsubGMM_KBCG20_params_Z_ALL_allregca_attn3_corrreg_cs_dmb.csv')

 # calculate rock PGA
 period_used <- 0.
 vsrock <- 1100
 pars_period <- as.numeric(parameters[parameters$T == period_used,])
 coeffs <- pars_period[seq(2,15)]
 k1k2 <- interp_k1k2(period_used)
 dmb <- 0

 delta_ln_z <- 0
 coeffs_z <- c(0,0)
 reg <- getRegionKBCG(UserRegion)

 if(reg == 1) {
  coeffs[c(1,2,11)] <- pars_period[c(17, 24, 31)]
  coeffs_attn <- pars_period[c(38, 45, 52, 59, 66, 73)]
 } else if(reg == 2) {
  coeffs[c(1,2,11)] <- pars_period[c(18, 25, 32)]
  coeffs_attn <- pars_period[c(39, 46, 53, 60, 67, 74)]
 } else if(reg == 3) {
  coeffs[c(1,2,11)] <- pars_period[c(19, 26, 33)]
  coeffs_attn <- pars_period[c(40, 47, 54, 61, 68, 75)]
```

```
  } else if(reg == 4) {
              dmb <- interp_dmb(period_used)
    coeffs[c(1,2,11)] <- pars_period[c(20, 27, 34)]
    coeffs_attn <- pars_period[c(41, 48, 55, 62, 69, 76)]
  } else if(reg == 5) {
    coeffs[c(1,2,11)] <- pars_period[c(21, 28, 35)]
    coeffs_attn <- pars_period[c(42, 49, 56, 63, 70, 77)]
  } else if(reg == 6) {
              dmb <- interp_dmb(period_used)
    coeffs[c(1,2,11)] <- pars_period[c(22, 29, 36)]
    coeffs_attn <- pars_period[c(43, 50, 57, 64, 71, 78)]
  } else if(reg == 7) {
    coeffs[c(1,2,11)] <- pars_period[c(23, 30, 37)]
    coeffs_attn <- pars_period[c(44, 51, 58, 65, 72, 79)]
  } else if(reg == 0) {
    coeffs_attn <- pars_period[c(11, 11, 11, 11, 10, 11)]
  }


  delta_bz <- pars_period[c(80,81)]
  coeffs_nft <- pars_period[c(82,83)]
  fs <- (LeftNumber(EventType, 1))

  mbreak <- (1 - fs) * (Mb + dmb) + fs * Mb
  zbreak <- (1 - fs) * (30 + delta_bz[1]) + fs * (80 + delta_bz[2])

  pgarock <- exp(KBCG20_med(M, Rrup, AlphaBackarc, AlphaNankai, Ztor, EventType, vsrock, delta_ln_z, coeffs, coeffs_attn,
coeffs_z, mbreak, zbreak, k1k2[1], k1k2[2], coeffs_nft[1], coeffs_nft[2], 0, UserRegion))


  # calculate PSA
  period_used <- period
  parameters_posterior <- read.csv(sprintf('posterior_coefficients_KBCG20_T%05.2f.csv',period_used))
  k1k2 <- interp_k1k2(period_used)
  dmb <- 0

  med_predictions <- matrix(nrow = num_samples, ncol = 1)

  for(k in 1:num_samples) {
    pars_period <- as.numeric(parameters_posterior[k,])
    coeffs <- pars_period[seq(2,15)]

    delta_ln_z <- 0
    coeffs_z <- c(0,0)

              reg <- getRegionKBCG(UserRegion)
    if(reg == 1) {
      coeffs[c(1,2,11)] <-pars_period[c(17, 24, 31)]
      coeffs_attn <-pars_period[c(38, 45, 52, 59, 66, 73)]
    } else if(reg == 2) {
      coeffs[c(1,2,11)] <-pars_period[c(18, 25, 32)]
      coeffs_attn <-pars_period[c(39, 46, 53, 60, 67, 74)]
                if(tolower(Z2pt5) == 'default' || Z2pt5 <0) {
                  delta_ln_z <- 0
                } else {
                delta_ln_z <- log(Z2pt5) - calc_z_from_vs(Vs30, pars_z_casc)
                }
                if(LeftNumber(PNWbasinStrux, 1) == 1) {
```

```
      coeff_seattle <- parameters_zmod[parameters_zmod$T == period_used,]$mean_residual_Seattle_basin
      coeffs_z <-c(coeff_seattle,0)
    } else if(LeftNumber(PNWbasinStrux, 1) == 2) {
      coeffs_z <- as.numeric(parameters_zmod[parameters_zmod$T == period_used,c(2,3)])
    }
  } else if(reg == 3) {
    coeffs[c(1,2,11)] <-pars_period[c(19, 26, 33)]
    coeffs_attn <-pars_period[c(40, 47, 54, 61, 68, 75)]
  } else if(reg == 4) {
              dmb <- interp_dmb(period_used)
    coeffs[c(1,2,11)] <-pars_period[c(20, 27, 34)]
    coeffs_attn <-pars_period[c(41, 48, 55, 62, 69, 76)]
              if(tolower(Z2pt5) == 'default' || Z2pt5 <0) {
                delta_ln_z <- 0
              } else {
             delta_ln_z <- log(Z2pt5) - calc_z_from_vs(Vs30, pars_z_ja)
              }
    coeffs_z <- as.numeric(parameters_zmod[parameters_zmod$T == period_used,c(5,6)])
  } else if(reg == 5) {
    coeffs[c(1,2,11)] <-pars_period[c(21, 28, 35)]
    coeffs_attn <-pars_period[c(42, 49, 56, 63, 70, 77)]
              if(tolower(Z1pt0) == "default" || Z1pt0 <0) {
                delta_ln_z <- 0
              } else {
                 delta_ln_z <- log(Z1pt0) - calc_z_from_vs(Vs30, pars_z_nz)
              }
    coeffs_z <- as.numeric(parameters_zmod[parameters_zmod$T == period_used,c(7,8)])
  } else if(reg == 6) {
              dmb <- interp_dmb(period_used)
    coeffs[c(1,2,11)] <-pars_period[c(22, 29, 36)]
    coeffs_attn <-pars_period[c(43, 50, 57, 64, 71, 78)]
  } else if(reg == 7) {
    coeffs[c(1,2,11)] <-pars_period[c(23, 30, 37)]
    coeffs_attn <-pars_period[c(44, 51, 58, 65, 72, 79)]
              if(tolower(Z1pt0) == "default" || Z1pt0 <0) {
                delta_ln_z <- 0
              } else {
                 delta_ln_z <- log(Z1pt0) - calc_z_from_vs(Vs30, pars_z_tw)
              }
    coeffs_z <- as.numeric(parameters_zmod[parameters_zmod$T == period_used,c(9,10)])
  } else if(reg == 0) {
    coeffs[c(1,2,11)] <-pars_period[c(172, 173, 174)]
    coeffs_attn <- pars_period[c(175, 176, 177, 178, 179, 180)]
  }


  delta_bz <- pars_period[c(80,81)]
  coeffs_nft <- pars_period[c(82,83)]

  mbreak <-(1 - fs) * (Mb + dmb) + fs * Mb
  zbreak <-(1 - fs) * (30 + delta_bz[1]) + fs * (80 + delta_bz[2])
  med <- KBCG20_med(M, Rrup, AlphaBackarc, AlphaNankai, Ztor, EventType, Vs30, delta_ln_z, coeffs, coeffs_attn,
coeffs_z, mbreak, zbreak, k1k2[1], k1k2[2], coeffs_nft[1], coeffs_nft[2], pgarock, UserRegion)
  med_predictions[k] <- med
 }

              KBCG20_posteriorAtTlist_All <- 0
  KBCG20_posteriorAtTlist_All[1] <- getMeanOfArray(med_predictions)
```

```
   KBCG20_posteriorAtTlist_All[2] <- getMedianOfArray(med_predictions)
   KBCG20_posteriorAtTlist_All[3] <- getStdDev(med_predictions)
   for (i in range(length(med_predictions)) ){
      KBCG20_posteriorAtTlist_All[3 + i  ] <- med_predictions[i]
   }
   return(KBCG20_posteriorAtTlist_All)
}


getMeanOfArray <-function(inArray) {
   # CODE DEVELOPED/IMPLEMENTED BY
   #         Silvia Mazzoni, 2020
   #          smazzoni@ucla.edu
   return(mean(inArray))

}

getMedianOfArray <-function(inArray) {
   # CODE DEVELOPED/IMPLEMENTED BY
   #         Silvia Mazzoni, 2020
   #          smazzoni@ucla.edu
   return(median(inArray))
}

getStdDev <-function(Arr) {
   # CODE DEVELOPED/IMPLEMENTED BY
   #         Silvia Mazzoni, 2020
   #          smazzoni@ucla.edu
#             return(sd(Arr))
# I used below to be consistent with other platforms
# the results are the same, off by 0.3#
    k1 <- 1
    k2 <- length(Arr)

    n <- 0
    avg <- getMeanOfArray(Arr)
    SumSq <- 0

              for (i in 1: length(Arr) ){
      n <- n + 1
      SumSq <- SumSq + (Arr[i] - avg) ** 2
    }

    getStdDevValue <- sqrt(SumSq / (n - 1))

    return(getStdDevValue)

}




KBCG20_posterior <-function(period, M, Rrup, AlphaBackarc, AlphaNankai, Ztor, EventType, Vs30, Z1pt0, Z2pt5, Mb,
UserRegion, num_samples = 100  , PNWbasinStrux = 0) {
   # CODE DEVELOPED/IMPLEMENTED BY
   #         Silvia Mazzoni, 2020
   #          smazzoni@ucla.edu
```

```
  thisOut <- -999
  XinterpMin <- 0
  XinterpMax <- 10
  XinterpType <- "log"
  YinterpType <- "linear" # KBCG20_posteriorAtTlist values are logs already
  extrapolateType <- "extrapolate"

  Parameters <- read.csv('NGAsubGMM_KBCG20_coefficients.csv')
  TvalueList <- Parameters$T
  InterpArray <- interpolateFunction(period, TvalueList, XinterpType, extrapolateType, XinterpMin, XinterpMax)

  med_predictions <- 0
  period0 <- TvalueList[InterpArray[1]]
  KBCG20_posteriorAtTlist0 <- KBCG20_posteriorAtTlist(period0, M, Rrup, AlphaBackarc, AlphaNankai, Ztor, EventType,
Vs30, Z1pt0, Z2pt5, Mb, UserRegion, num_samples, PNWbasinStrux)
  y0 <- InterpArray[1]
  if(InterpArray[2] <= 0 ) {
    KBCG20_posteriorValue <- KBCG20_posteriorAtTlist0
    return(KBCG20_posteriorValue)
  } else {
    period1 <- TvalueList[InterpArray[2]]
    KBCG20_posteriorAtTlist1 <- KBCG20_posteriorAtTlist(period1, M, Rrup, AlphaBackarc, AlphaNankai, Ztor, EventType,
Vs30, Z1pt0, Z2pt5, Mb, UserRegion, num_samples, PNWbasinStrux)
    for (iCase in 1: length(KBCG20_posteriorAtTlist0) ){
      y0 <- KBCG20_posteriorAtTlist0[iCase]
      y1 <- KBCG20_posteriorAtTlist1[iCase]

      if(tolower(LeftString(YinterpType, nchar("log"))) == tolower("log") ) {
        if(y0 <= 0 ) {
          y0 <- 0.000000001
        }
        if(y1 <= 0  ) {
          y1 <- 0.000000001
        }
        y0 <- log(y0)
        y1 <- log(y1)
      }
      y0
      y1
      InterpArray[3]
      y0 + (y1 - y0) * InterpArray[3]
      med_predictions[iCase] <- y0 + (y1 - y0) * InterpArray[3]
      if(tolower(LeftString(YinterpType, nchar("log"))) == tolower("log") ) {
        med_predictions[iCase] <- exp(med_predictions[iCase])
      }
    }

  }

  KBCG20_posteriorValue <- med_predictions

}




# function to calculate aliatory sigma
```

```
KBCG20_sigmaAleatory <-function(period) {
   # CODE DEVELOPED/IMPLEMENTED BY
   #         Silvia Mazzoni, 2020
   #          smazzoni@ucla.edu

   thisOut <- -999
   XinterpMin <- 0
   XinterpMax <- 10
   XinterpType <- "log"
   YinterpType <- "linear"
   extrapolateType <- "extrapolate"

    Parameters <- read.csv('NGAsubGMM_KBCG20_coefficients.csv')
   TvalueList <- Parameters$T
   InterpArray <- interpolateFunction(period, TvalueList, XinterpType, extrapolateType, XinterpMin, XinterpMax)

   period0 <- TvalueList[InterpArray[1]]
   y0 <- KBCG20_sigmaAleatoryAtTlist(period0)

   if(InterpArray[2] <= 0 ) {
      thisOut <- y0
   } else {
      period1 <- TvalueList[InterpArray[2]]
      y1 <- KBCG20_sigmaAleatoryAtTlist(period1)
      if(tolower(LeftString(YinterpType, nchar("log"))) == tolower("log") ) {
         if(y0 <= 0  ) {
            y0 <- 0.000000001
         }
         if(y1 <= 0  ) {
            y1 <- 0.000000001
         }
         y0 <- log(y0)
         y1 <- log(y1)
      }
      thisOut <- y0 + (y1 - y0) * InterpArray[3]
      if(tolower(LeftString(YinterpType, nchar("log"))) == tolower("log") ) {
         thisOut <- exp(thisOut)
      }

   }

   KBCG20_sigmaAleatoryValue <- thisOut

}


KBCG20_sigmaAleatoryAtTlist <-function(period) {
    # CODE DEVELOPED/IMPLEMENTED BY
   #         Silvia Mazzoni, 2020
   #          smazzoni@ucla.edu

  Parameters <- read.csv('NGAsubGMM_KBCG20_coefficients.csv')
  # calculate PSA
  period_used <- period
  pars_period <- getRangeRowT(Parameters, period_used)
```

```
  Phi <- pars_period[84]
  Tau <- pars_period[85]
  KBCG20_sigmaAleatoryAtTlistValue <- sqrt(Phi * Phi + Tau * Tau)

}

KBCG20_sigmaPhi <-function(period) {
    # CODE DEVELOPED/IMPLEMENTED BY
   #         Silvia Mazzoni, 2020
   #          smazzoni@ucla.edu

   Parameters <- read.csv('NGAsubGMM_KBCG20_coefficients.csv')

  # calculate PSA
  period_used <- period
  pars_period <- getRangeRowT(Parameters, period_used)

  Phi <- pars_period[84]
  KBCG20_sigmaPhiValue <- Phi

}

KBCG20_sigmaTau <-function(period) {
    # CODE DEVELOPED/IMPLEMENTED BY
   #         Silvia Mazzoni, 2020
   #          smazzoni@ucla.edu

   Parameters <- read.csv('NGAsubGMM_KBCG20_coefficients.csv')

  # calculate PSA
  period_used <- getTFvalue(period)
  pars_period <- getRangeRowT(Parameters, period_used)
  Tau <- pars_period[85]
  KBCG20_sigmaTauValue <- Tau

}



KBCG20_SigmaEpistemic <-function(period, M, Rrup, AlphaBackarc, AlphaNankai, Ztor, EventType, Vs30, Z1pt0, Z2pt5, Mb,
UserRegion,  num_samples = 100  , PNWbasinStrux = 0 ) {
   # CODE DEVELOPED/IMPLEMENTED BY
   #         Silvia Mazzoni, 2020
   #          smazzoni@ucla.edu


   if(num_samples == 0 ) {
      KBCG20_SigmaEpistemicValue <- 0
   } else if(num_samples < 50 ) {
      KBCG20_SigmaEpistemicValue <- "not enough samples, please select a value =0 || between 50 and 800"
   } else if(num_samples <= 800 ) {

      thisKBCG20_posterior <- KBCG20_posterior(period, M, Rrup, AlphaBackarc, AlphaNankai, Ztor, EventType, Vs30, Z1pt0,
Z2pt5, Mb, UserRegion, num_samples, PNWbasinStrux)
      KBCG20_SigmaEpistemicValue <- thisKBCG20_posterior[3]
   } else {
      KBCG20_SigmaEpistemicValue <- "Please enter a value =0 || between 50 and 800"
```

```
   }
   return(KBCG20_SigmaEpistemicValue)

}

KBCG20_SigmaTotal <-function(period, M, Rrup, AlphaBackarc, AlphaNankai, Ztor, EventType, Vs30, Z1pt0, Z2pt5, Mb,
UserRegion,  num_samples = 100  , PNWbasinStrux = 0 ) {
   # CODE DEVELOPED/IMPLEMENTED BY
   #         Silvia Mazzoni, 2020
   #          smazzoni@ucla.edu

   SigmaAleatory <- KBCG20_sigmaAleatory(period)
   if(num_samples == 0 ) {
      SigmaEpistemic <- 0.
   } else {
      SigmaEpistemic <- KBCG20_SigmaEpistemic(period, M, Rrup, AlphaBackarc, AlphaNankai, Ztor, EventType, Vs30, Z1pt0,
Z2pt5, Mb, UserRegion, num_samples, PNWbasinStrux)
   }
   KBCG20_SigmaTotalValue <- sqrt(SigmaAleatory * SigmaAleatory + SigmaEpistemic * SigmaEpistemic)

}




######################################################################
## if(you want the so-called 'error function'
 erf <- function(x) 2 * pnorm(x * sqrt(2)) - 1



getMbDefault <-function(NGAsubModelLabel, EventType, UserRegion, saturation_region_user) {
   # CODE DEVELOPED/IMPLEMENTED BY
   #         Silvia Mazzoni, 2020
   #          smazzoni@ucla.edu

   # Silvia Mazzoni: I inserted the global model and replaced the NA values with those of the global model

   if(LeftNumber(EventType, 1) == 1 || LeftNumber(EventType, 1) == 5) {
      if(NGAsubModelLabel == "KBCG20" ) {
         #MbKBCG20Intraslab
         saturation_region_here <- saturation_region_user
         saturation_regions_SBZ <- c("global", "Aleutian", "Alaska", "Cascadia", "Central_America_N", "Central_America_S",
"Japan_Pac", "Japan_Phi", "New_Zealand_N", "New_Zealand_S", "South_America_N", "South_America_S", "Taiwan_W",
"Taiwan_E")
         thisMbList <- c(7.6, 8, 7.2, 7.2, 7.4, 7.6, 7.6, 7.6, 7.6, 7.6, 7.3, 7.2, 7.7, 7.7)
         User_regions_SBZ <- c("0_global", "1_Alaska", "1_Alaska", "2_Cascadia", "3_CentralAmerica&Mexico",
"3_CentralAmerica&Mexico", "4_Japan", "4_Japan", "5_NewZealand", "5_NewZealand", "6_SouthAmerica",
"6_SouthAmerica", "7_Taiwan", "7_Taiwan")
      } else {
        #MbPSHAB20Intraslab
         saturation_region_here <- getPSHABSubductionRegion(saturation_region_user)
         Mb_global_Slab <- 7.6
         saturation_regions_SBZ <- c("global", "Aleutian", "Alaska", "-999", "Cascadia", "CAM_S", "CAM_N", "Japan_Pac",
"Japan_Phi", "New_Zealand_N", "New_Zealand_S", "SA_N", "SA_S", "Taiwan_W", "Taiwan_E")
         thisMbList <- c(Mb_global_Slab, 7.98, 7.2, Mb_global_Slab, 7.2, 7.6, 7.4, 7.65, 7.55, Mb_global_Slab, Mb_global_Slab,
7.3, 7.25, 7.7, 7.7)
```

```
        User_regions_SBZ <- c("0_global", "1_Alaska", "1_Alaska", "0_global", "2_Cascadia", "3_CentralAmerica&Mexico",
"3_CentralAmerica&Mexico", "4_Japan", "4_Japan", "5_NewZealand", "5_NewZealand", "6_SouthAmerica",
"6_SouthAmerica", "7_Taiwan", "7_Taiwan")
      }

   } else {
      if(NGAsubModelLabel == "KBCG20" ) {
         #MbKBCG20Interface
         saturation_region_here <- saturation_region_user
         saturation_regions_SBZ <- c("global", "Aleutian", "Alaska", "Cascadia", "Central_America_N", "Central_America_S",
"Japan_Pac", "Japan_Phi", "New_Zealand_N", "New_Zealand_S", "South_America_N", "South_America_S", "Taiwan_W",
"Taiwan_E")
         thisMbList <- c(7.9, 8, 8.6, 8, 7.4, 7.5, 8.5, 7.7, 8.3, 8, 8.5, 8.6, 7.1, 7.1)
         User_regions_SBZ <- c("0_global", "1_Alaska", "1_Alaska", "2_Cascadia", "3_CentralAmerica&Mexico",
"3_CentralAmerica&Mexico", "4_Japan", "4_Japan", "5_NewZealand", "5_NewZealand", "6_SouthAmerica",
"6_SouthAmerica", "7_Taiwan", "7_Taiwan")
      } else {
         #MbPSHAB20Interface
         saturation_region_here <- getPSHABSubductionRegion(saturation_region_user)
         Mb_global_IF <- 7.9
         saturation_regions_SBZ <- c("global", "Aleutian", "Alaska", "-999", "Cascadia", "CAM_S", "CAM_N", "Japan_Pac",
"Japan_Phi", "New_Zealand_N", "New_Zealand_S", "SA_N", "SA_S", "Taiwan_W", "Taiwan_E")
         thisMbList <- c(Mb_global_IF, 8, 8.6, Mb_global_IF, 7.7, 7.4, 7.4, 8.5, 7.7, Mb_global_IF, Mb_global_IF, 8.5, 8.6, 7.1,
7.1)
         User_regions_SBZ <- c("0_global", "1_Alaska", "1_Alaska", "0_global", "2_Cascadia", "3_CentralAmerica&Mexico",
"3_CentralAmerica&Mexico", "4_Japan", "4_Japan", "5_NewZealand", "5_NewZealand", "6_SouthAmerica",
"6_SouthAmerica", "7_Taiwan", "7_Taiwan")
      }
   }
   thisSaturationRegionDbRegion <- getArrayMap(saturation_region_here, saturation_regions_SBZ, User_regions_SBZ)
   if(tolower(thisSaturationRegionDbRegion) != tolower(UserRegion)) {
      saturation_region <- "global"
   }

   Mb <- getArrayMap(saturation_region_here, saturation_regions_SBZ, thisMbList)

   if(checkEmptyNA999(Mb)>0) {
     Mb <- getArrayMap("global", saturation_regions_SBZ, thisMbList)
   }
   return(Mb)


}


#####################################################################

# updated 6/1/2020 by Silvia Mazzoni

# GMM_at_760_slab_v4.R

##Grace Parker
##Modified February 26 to expand comments
##Modified March 25, 2020 to call coefficients from master table

## Input Parameters ----------------------------------------------------------

##Event type: 0 == interface, 1 == slab
```

```
##region corresponds to options in the DatabaseRegion column of the flatfile, plus global. Must be a string. If no matches,
default will be global model:
## "global", "Alaska", "Cascadia", "CAM", "Japan", "SA" or "Taiwan"

##Saturation Region corresponds to regions defined by R. Archuleta and C. Ji:
## "global", "Aleutian","Alaska","Cascadia","Central_America_S", "Central_America_N",
"Japan_Pac","Japan_Phi","South_America_N","South_America_S", "Taiwan_W","Taiwan_E"

## Rrup is number in kilometers

##Hypocentral depth in km. To use Ztor value to estimate hypocentral depth, see Ch. 4.3.3 of Parker et al. PEER report

## period can be: (-1,0,0.01,0.02,0.025,0.03,0.04,0.05,0.075,0.1,0.15,0.2,0.25,0.3,0.4,0.5,0.75,1,1.5,2,2.5,3,4,5,7.5,10)
## where -1 == PGV and 0 == PGA

## Other pertinent information ----------------------------------------------
##Coefficient files must be in the active working directory
## This function has no site term. Can only estimate ground motion at the reference condition VS30 = 760m/s
## The output is the desired median model prediction in LN units
## Take the exponential to get PGA, PSA in g or the PGV in cm/s

##Function to compute GMM predictions at 760m/s for slab and interface



PSHAB20_GMM_at_760_Slab <-function(EventType, UserRegion, saturation_region_user, Rrup, M, Zhypo, period) {
    # CODE DEVELOPED/IMPLEMENTED BY
   #         Silvia Mazzoni, 2020
   #          smazzoni@ucla.edu

 if (LeftNumber(EventType, 1) == 0 ) {
  stop("This function is only for slab")
 }

# Import Master Coefficient Table ----------------------------------------

 coefficients <- read.csv("PSHAB20_Table_E2_Slab_Coefficients_OneRowHeader.csv")

 #Isolate desired period
            coefficients.T <- coefficients[coefficients[1] == period,]

# Constant ----------------------------------------------------------------
            region <- getRegionPSHAB(UserRegion)
            saturation_region <- getPSHABSubductionRegion(saturation_region_user)


 #Define Mb based on Archuleta and Ji (2019)
 Mb <- getMbDefault("PSHAB20",EventType, UserRegion, saturation_region)

 #Isolate constant
 if(region == "global" ){
  c0 <- as.numeric(subset(coefficients.T,select = Global_c0))
 } else if(region == "Alaska" || region == "SA"){
  c0 <- as.numeric(subset(coefficients.T, select = c(eval(as.name(paste(saturation_region,"_c0",sep=""))))))
 } else {
  c0 <- as.numeric(subset(coefficients.T, select = c(eval(as.name(paste(region,"_c0",sep=""))))))
 }
```

```
# Path Term ------------------------------------------------------------------

 #near-source saturation
 if(M <= Mb){
  littleM <- (log10(35)-log10(3.12))/(Mb-4)
  h <- 10^(littleM*(M-Mb) + log10(35))
 } else {
  h <- 35
 }

 Rref <- sqrt(1 + h^2)
 R <- sqrt(Rrup^2 + h^2)
 LogR <- log(R)
 R_Rref <- log(R/Rref)
 #Need  to isolate regional anelastic coefficient, a0
 if(region == "global"){
  a0 <- coefficients.T$Global_a0
 } else {
  a0 <- as.numeric(subset(coefficients.T, select = c(eval(as.name(paste(region,"_a0",sep=""))))))
 }
 if(is.na(a0)){
  a0 <- coefficients.T$Global_a0
 }
 Fp <- as.numeric(coefficients.T$c1*LogR + (coefficients.T$b4*M)*R_Rref + a0*R)
 # Magnitude Scaling ----------------------------------------------------------

              func1 <- function(M,Mb){ifelse(M <= Mb,M-Mb,0)}
              func2 <- function(M,Mb){ifelse(M > Mb,M-Mb,0)}
              func3 <- function(M,Mb){ifelse(M <= Mb, (M-Mb)^2, 0)}

 Fm <- coefficients.T$c4*func1(M,Mb) + coefficients.T$c6*func2(M,Mb) + coefficients.T$c5*func3(M,Mb)


 # Source Depth Scaling ------------------------------------------------------

 if(Zhypo >= coefficients.T$db_km){
  Fd <-coefficients.T$d
 } else  if(Zhypo <= 20){
  Fd <- coefficients.T$m*(20-coefficients.T$db_km) + coefficients.T$d
 } else {
  Fd <- coefficients.T$m*(Zhypo - coefficients.T$db_km) + coefficients.T$d
 }

  mu <- c0 + Fp + Fm + Fd
 return(mu)

}



getRegionPSHAB <- function(UserRegion) {
   #' CODE DEVELOPED/IMPLEMENTED BY
   #'         Silvia Mazzoni, 2020
   #'          smazzoni@ucla.edu
```

```
   RegionIDList <-
c("0_global","1_Alaska","2_Cascadia","3_CentralAmerica&Mexico","4_Japan","5_NewZealand","6_SouthAmerica","7_Taiwa
n")
   PSHAB_RegionIDList <- c("global","Alaska","Cascadia","CAM","Japan","global","SA","Taiwan")

            thisIndex <- match(UserRegion,RegionIDList)
   getRegionPSHAB <- PSHAB_RegionIDList[thisIndex]
   return(getRegionPSHAB)
}

getPSHABSubductionRegion <- function(UserSubductionRegion) {
   #' CODE DEVELOPED/IMPLEMENTED BY
   #'         Silvia Mazzoni, 2020
   #'          smazzoni@ucla.edu
   SubdRegionIDList <-
c("Alaska","Aleutian","Cascadia","CAM_N","CAM_S","global","Japan_Pac","Japan_Phi","NZ_N","NZ_S","SA_N","SA_S","Taiw
an_E","Taiwan_W","Alaska","Aleutian","Cascadia","Central_America_N","Central_America_S","global","Japan_Pac","Japan_
Phi","New_Zealand_N","New_Zealand_S","South_America_N","South_America_S","Taiwan_E","Taiwan_W")
   PSHAB_SubdRegionIDList <-
c("Alaska","Aleutian","Cascadia","CAM_N","CAM_S","global","Japan_Pac","Japan_Phi","New_Zealand_N","New_Zealand_S"
,"SA_N","SA_S","Taiwan_E","Taiwan_W","Alaska","Aleutian","Cascadia","CAM_N","CAM_S","global","Japan_Pac","Japan_P
hi","New_Zealand_N","New_Zealand_S","SA_N","SA_S","Taiwan_E","Taiwan_W")

            thisIndex <- match(UserSubductionRegion,SubdRegionIDList)
   getsubRegionPSHAB <- PSHAB_SubdRegionIDList[thisIndex]
   return(getsubRegionPSHAB)
}

getRegionKBCG <- function(UserRegion) {
   #' CODE DEVELOPED/IMPLEMENTED BY
   #'         Silvia Mazzoni, 2020
   #'          smazzoni@ucla.edu

   out <- (LeftString(UserRegion,1))
   return(out)
}


getArrayMap <-function(Xvalue, Xarray, Yarray) {
   # CODE DEVELOPED/IMPLEMENTED BY
   #         Silvia Mazzoni, 2020
   #         smazzoni@ucla.edu
   thisIndex <- match(Xvalue,Xarray)
   getArrayMapValue <- Yarray[thisIndex]
            return(getArrayMapValue)
}



getRangeValue <-function(thisRange, Xvalue, xHeader, yHeader) {
   # CODE DEVELOPED/IMPLEMENTED BY
   #         Silvia Mazzoni, 2020
   #         smazzoni@ucla.edu
   # not sure how to make it work with xHeader as a variable
            #   getRangeValueValue <- table2array(thisRange(thisRange.xHeader==Xvalue,yHeader))
            coefficients.T <- subset(thisRange,xHeader == Xvalue)
   getRangeValueTValue <- as.numeric(subset(coefficients.T,select = yHeader))
   return(getRangeValueTValue)
```

```
}

getRangeValueRegion <-function(thisRange, Xvalue,  yHeader) {
   # CODE DEVELOPED/IMPLEMENTED BY
   #         Silvia Mazzoni, 2020
   #          smazzoni@ucla.edu
              coefficients.Region <- subset(thisRange,Region == Xvalue)
   getRangeValueTValue <- as.numeric(subset(coefficients.Region,select = yHeader))
   return(getRangeValueTValue)
}

getRangeValueT <-function(thisRange, Xvalue, yHeader) {
   # CODE DEVELOPED/IMPLEMENTED BY
   #         Silvia Mazzoni, 2020
   #          smazzoni@ucla.edu
              coefficients.T <- subset(thisRange,T == Xvalue)
   getRangeValueTValue <- as.numeric(subset(coefficients.T,select = yHeader))
   return(getRangeValueTValue)
}



 checkEmptyNA999 <-function(thisvalue) {
   checkEmptyNA999value <- -1
   if(is.null(thisvalue)) {
    checkEmptyNA999value <- 1
   } else if(as.character(thisvalue) == 'NA') {
      checkEmptyNA999value <- 1
   } else if(thisvalue == -999) {
      checkEmptyNA999value <- 1
   }
   return(checkEmptyNA999value)

}



# GMM_at_VS30_slab_v4.R

##Grace Parker
##Modified February 26 to expand comments
##Modified March 25, 2020 to call consolidated coefficient table

## Input Parameters ----------------------------------------------------------

##Event type: 0 == interface, 1 == slab

##region corresponds to options in the DatabaseRegion column of the flatfile, plus global. Must be a string. If no matches,
default will be global model:
## "global", "Alaska", "Cascadia", "CAM", "Japan", "SA" or "Taiwan"

##Saturation Region corresponds to regions defined by R. Archuleta and C. Ji:
## "global", "Aleutian","Alaska","Cascadia","Central_America_S", "Central_America_N",
"Japan_Pac","Japan_Phi","South_America_N","South_America_S", "Taiwan_W","Taiwan_E"

## Rrup is number in kilometers

##Hypocentral depth in km. To use Ztor value to estimate hypocentral depth, see Ch. XXX of Parker et al. PEER report
```

```
## period can be: (-1,0,0.01,0.02,0.025,0.03,0.04,0.05,0.075,0.1,0.15,0.2,0.25,0.3,0.4,0.5,0.75,1,1.5,2,2.5,3,4,5,7.5,10)
## where -1 == PGV and 0 == PGA

##VS30 in units m/s

##Z2.5 in units m. Only used if DatabaseRegion == "Japan" or "Cascadia". Can also specify "default" to get no basin term

##basin is only used if DatabaseRegion == "Cascadia". Value can be 0, 1, or 2, where 0 == having an estimate of Z2.5 outside mapped basin, 1 == Seattle basin, and 0 == other mapped basin (Tacoma, Everett, Georgia, etc.)

## Other pertinent information ----------------------------------------------
##Coefficient files must be in the active working directory
## "GMM_at_VS30_Slab_v2.R" calls function "GMM_at_760_Slab_v2.R" to compute PGAr in the nonlinear site term. This function must be in the R environment else an error will occur.
## The output is the desired median model prediction in LN units
## Take the exponential to get PGA, PSA in g or  PGV in cm/s


##Function to compute GMM predictions at various VS30s for slab


PSHAB20_GMM_at_Vs30_Slab <-function(EventType, UserRegion, saturation_region_user, Rrup, M, Zhypo, period, Vs30, Z2pt5, basin) {
   # CODE DEVELOPED/IMPLEMENTED BY
   #         Silvia Mazzoni, 2020
   #          smazzoni@ucla.edu
  if (LeftNumber(EventType, 1) == 0 ) {
   stop("This function is only for slab")
  }

# Import Master Coefficient Table ----------------------------------------

  coefficients <- read.csv("PSHAB20_Table_E2_Slab_Coefficients_OneRowHeader.csv")
  #Isolate desired period
              coefficients.T <- coefficients[coefficients[1] == period,]


# Constant ---------------------------------------------------------------
              region <- getRegionPSHAB(UserRegion)
              saturation_region <- getPSHABSubductionRegion(saturation_region_user)

  #Define Mb based on Archuleta and Ji (2019)
              Mb <- getMbDefault("PSHAB20",EventType, UserRegion, saturation_region)


  basin <- LeftNumber(basin, 1)
  #Isolate constant
  if(region == "global"){
    c0 <- as.numeric(subset(coefficients.T,select = Global_c0))
  } else  if(region == "Alaska" || region == "SA"){
    c0 <- as.numeric(subset(coefficients.T, select = c(eval(as.name(paste(saturation_region,"_c0",sep=""))))))
  } else {
    c0 <- as.numeric(subset(coefficients.T, select = c(eval(as.name(paste(region,"_c0",sep=""))))))
  }

# Path Term ---------------------------------------------------------------
```

```
  #near-source saturation
  if(M <= Mb){
    littleM <- (log10(35)-log10(3.12))/(Mb-4)
    h <- 10^(littleM*(M-Mb) + log10(35))
  } else {
    h <- 35
  }

  Rref <- sqrt(1 + h^2)
  R <- sqrt(Rrup^2 + h^2)
  LogR <- log(R)
  R_Rref <- log(R/Rref)

  #Need  to isolate regional anelastic coefficient, a0
  if(region == "global"){
    a0 <- coefficients.T$Global_a0
  } else {
    a0 <- as.numeric(subset(coefficients.T, select = c(eval(as.name(paste(region,"_a0",sep=""))))))
  }
  if(is.na(a0)){
    a0 <- coefficients.T$Global_a0
  }
  Fp <- as.numeric(coefficients.T$c1*LogR + (coefficients.T$b4*M)*R_Rref + a0*R)

# Magnitude Scaling ----------------------------------------------------------

              func1 <- function(M,Mb){ifelse(M <= Mb,M-Mb,0)}
              func2 <- function(M,Mb){ifelse(M > Mb,M-Mb,0)}
              func3 <- function(M,Mb){ifelse(M <= Mb, (M-Mb)^2, 0)}
  Fm <- coefficients.T$c4*func1(M,Mb) + coefficients.T$c6*func2(M,Mb) + coefficients.T$c5*func3(M,Mb)

# Source Depth Scaling -------------------------------------------------------


  if(Zhypo >= coefficients.T$db_km){
    Fd <-coefficients.T$d
  } else  if(Zhypo <= 20){
    Fd <- coefficients.T$m*(20-coefficients.T$db_km) + coefficients.T$d
  } else {
    Fd <- coefficients.T$m*(Zhypo - coefficients.T$db_km) + coefficients.T$d
  }
# Linear Site Amplification ------------------------------------------------

  #Site Coefficients
  V1 <- coefficients.T$V1_m_s
  V2 <- coefficients.T$V2_m_s
  Vref <- coefficients.T$Vref_m_s

  if(region == "global"| region == "CAM"){
    s2 <- coefficients.T$Global_s2
    s1 <- s2
  } else  if(region =="Taiwan" || region == "Japan"){
    s2 <- as.numeric(subset(coefficients.T, select = c(eval(as.name(paste(region,"_s2",sep=""))))))
    s1 <- as.numeric(subset(coefficients.T, select = c(eval(as.name(paste(region,"_s1",sep=""))))))
  } else {
    s2 <- as.numeric(subset(coefficients.T, select = c(eval(as.name(paste(region,"_s2",sep=""))))))
    s1 <- s2
  }
```

```
  #Compute linear site term
  if(Vs30 <= V1){
    Flin <- s1*log(Vs30/V1) + s2*log(V1/Vref)
  } else  if(Vs30 <= V2){
    Flin <- s2*log(Vs30/Vref)
  } else {
    Flin <- s2*log(V2/Vref)
  }
# Nonlinear Site Term ----------------------------------------------------

  PGAr <- exp(PSHAB20_GMM_at_760_Slab(EventType,UserRegion,saturation_region,Rrup,M,Zhypo,0))
  f3 <- 0.05
  Vb <- 200
  Vref.Fnl <- 760

  if(period >= 3){
    Fnl <- 0
  } else {
    f2 <- coefficients.T$f4*(exp(coefficients.T$f5*(min(Vs30,Vref.Fnl)-Vb))-exp(coefficients.T$f5*(Vref.Fnl-Vb)))
    Fnl <- 0 + f2*log((PGAr+f3)/f3)
  }
# Basin Term -------------------------------------------------------------

  #import coefficients

  if(tolower(Z2pt5) == 'default' || Z2pt5 <= 0 || (region != "Japan" & region != "Cascadia")){
    Fb <- 0
  } else {
    if(region == "Cascadia"){
      theta0 <- 3.94
      theta1 <- -0.42
      vmu <- 200
      vsig <- 0.2
      e1 <- coefficients.T$C_e1

      if(basin == 0){
        e3 <- coefficients.T$C_e3 + coefficients.T$del_None
        e2 <-  coefficients.T$C_e2 +  coefficients.T$del_None
      } else  if(basin == 1){
        e3 <- coefficients.T$C_e3 + coefficients.T$del_Seattle
        e2 <-  coefficients.T$C_e2 +  coefficients.T$del_Seattle
      } else {
        e3 <- coefficients.T$C_e3
        e2 <-  coefficients.T$C_e2
      }



    } else  if(region == "Japan"){
      theta0 <- 3.05
      theta1 <- -0.8
      vmu <- 500
      vsig <-0.33
      e3 <- coefficients.T$J_e3
      e2 <- coefficients.T$J_e2
      e1 <- coefficients.T$J_e1
    }
```

```
   Z2pt5.pred <- 10^(theta0 + theta1*(1 + erf((log10(Vs30) - log10(vmu))/(vsig*sqrt(2)))))
   delZ2pt5 <- log(Z2pt5) - log(Z2pt5.pred)

   if(delZ2pt5 <= (e1/e3)){
     Fb <- e1
   } else  if(delZ2pt5 >= (e2/e3)){
     Fb <- e2
   } else {
     Fb <- e3*delZ2pt5
   }
  }

   mu <- c0 + Fp + Fm + Fd + Fnl + Flin + Fb
  return(mu)

}

Log10 <-function(x) {
   # CODE DEVELOPED/IMPLEMENTED BY
   #         Silvia Mazzoni, 2020
   #          smazzoni@ucla.edu
   Log10Value <- log(x) / log(10.)
   return(Log10Value)
}
```

```
# updated 6/1/2020 by Silvia Mazzoni
# GMM_at_760_IF_v4.R

##Grace Parker
##Modified February 26 to expand comments
##Modified March 25, 2020 to call consolidated coefficient tables

## Input Parameters ----------------------------------------------------

##Event type: 0 == interface, 1 == slab

##region corresponds to options in the DatabaseRegion column of the flatfile, plus global. Must be a string. If no matches,
default will be global model:
## "global", "Alaska", "Cascadia", "CAM", "Japan", "SA" or "Taiwan"

##Saturation Region corresponds to regions defined by R. Archuleta and C. Ji:
## "global", "Aleutian","Alaska","Cascadia","CAM_S", "CAM_N", "Japan_Pac","Japan_Phi","SA_N","SA_S",
"Taiwan_W","Taiwan_E"

## Rrup is number in kilometers

## period can be: (-1,0,0.01,0.02,0.025,0.03,0.04,0.05,0.075,0.1,0.15,0.2,0.25,0.3,0.4,0.5,0.75,1,1.5,2,2.5,3,4,5,7.5,10)
## where -1 == PGV and 0 == PGA

## Other pertinent information ----------------------------------------------
## Coefficient files must be in the active working directory
```

```
## This function has no site term. Can only estimate ground motion at the reference condition VS30 = 760m/s
## The output is the desired median model prediction in LN units
## Take the exponential to get PGA, PSA in g or the PGV in cm/s



PSHAB20_GMM_at_760_IF <-function(EventType, UserRegion, saturation_region_user, Rrup, M, period) {
   # CODE DEVELOPED/IMPLEMENTED BY
   #         Silvia Mazzoni, 2020
   #         smazzoni@ucla.edu
  PSHAB20_GMM_at_760_IFValue <- -999
  if(LeftNumber(EventType, 1) == 1 || LeftNumber(EventType, 1) == 5 ) {
   return(-999) # ("This function is only for IF")
  }
              region <- getRegionPSHAB(UserRegion)
              saturation_region <- getPSHABSubductionRegion(saturation_region_user)

# Import Master Coefficient Table ----------------------------------------

  coefficients <- read.csv("PSHAB20_Table_E1_Interface_Coefficients_OneRowHeader.csv")

  #Isolate desired period
              coefficients.T <- coefficients[coefficients[1] == period,]

  #Define Mb
              Mb <- getMbDefault("PSHAB20",EventType, UserRegion, saturation_region)


# Constant ---------------------------------------------------------------

  if(region == "global" ){
    c0 <- as.numeric(subset(coefficients.T, select = Global_c0))
  } else {
    c0 <- as.numeric(subset(coefficients.T, select = c(eval(as.name(paste(saturation_region,"_c0",sep=""))))))
    }

# Path Term --------------------------------------------------------------
  h <- 10^(-0.82 + 0.252*M)
  Rref <- sqrt(1 + h^2)
  R <- sqrt(Rrup^2 + h^2)
  LogR <- log(R)
  R_Rref <- log(R/Rref)

  #Need  to isolate regional anelastic coefficient, a0
  if(region == "global" ){
    a0 <- as.numeric(subset(coefficients.T, select = c(Global_a0)))
  } else {
    a0 <- as.numeric(subset(coefficients.T, select = c(eval(as.name(paste(region,"_a0",sep=""))))))
  }

  if(is.na(a0)){
    a0 <- as.numeric(subset(coefficients.T, select = c(Global_a0)))
  }

  Fp <- as.numeric(coefficients.T$c1*LogR + (coefficients.T$b4*M)*R_Rref + a0*R)


# Magnitude Scaling ------------------------------------------------------
```

```
            func1 <- function(M,Mb){ifelse(M <= Mb,M-Mb,0)}
            func2 <- function(M,Mb){ifelse(M > Mb,M-Mb,0)}
            func3 <- function(M,Mb){ifelse(M <= Mb, (M-Mb)^2, 0)}
 #compute M-scaling term
 Fm <- coefficients.T$c4*func1(M,Mb) + coefficients.T$c6*func2(M,Mb) + coefficients.T$c5*func3(M,Mb)


# Add it all up! ------------------------------------------------------------
  mu <-   c0 + Fp + Fm
 return(mu)


}




# Updated by Silvia Mazzoni 6/1/2020

# GMM_at_Vs30_IF_v4.R

##Grace Parker
##Modified February 26, 2020, to expand comments
##Modified March 25, 2020, to take coefficients from "Table_E1_Interface_Coefficients.csv"

## Input Parameters ----------------------------------------------------------

##Event type: 0 == interface, 1 == slab

##region corresponds to options in the DatabaseRegion column of the flatfile, plus global. Must be a string. If no matches, default will be global model:
 ## "global", "Alaska", "Cascadia", "CAM", "Japan", "SA" or "Taiwan"

##Saturation Region corresponds to regions defined by R. Archuleta and C. Ji:
 ## "global", "Aleutian","Alaska","Cascadia","CAM_S", "CAM_N", "Japan_Pac","Japan_Phi","SA_N","SA_S", "Taiwan_W","Taiwan_E"

## Rrup is number in kilometers

## period can be: (-1,0,0.01,0.02,0.025,0.03,0.04,0.05,0.075,0.1,0.15,0.2,0.25,0.3,0.4,0.5,0.75,1,1.5,2,2.5,3,4,5,7.5,10)
 ## where -1 == PGV and 0 == PGA

##VS30 in units m/s

##Z2.5 in units m. Only used if DatabaseRegion == "Japan" or "Cascadia". Can also specify "default" to get no basin term

##basin is only used if DatabaseRegion == "Cascadia". Value can be 0, 1, or 2, where 0 == having an estimate of Z2.5 outside mapped basin, 1 == Seattle basin, and 0 == other mapped basin (Tacoma, Everett, Georgia, etc.)

## Other pertinent information ----------------------------------------------
##Coefficient files must be in the active working directory
## "GMM_at_VS30_IF_v3.R" calls function "GMM_at_760_IF_v2.R" to compute PGAr in the nonlinear site term. This function must be in the R environment else an error will occur.
## The output is the desired median model prediction in LN units
#
```

```
PSHAB20_GMM_at_Vs30_IF <-function(EventType, UserRegion, saturation_region_user, Rrup, M, period, Vs30, Z2pt5,
basin) {
   # CODE DEVELOPED/IMPLEMENTED BY
   #         Silvia Mazzoni, 2020
   #          smazzoni@ucla.edu

  PSHAB20_GMM_at_Vs30_IFValue <- -999
  if(LeftNumber(EventType, 1) == 1 || LeftNumber(EventType, 1) == 5 ) {
   return(-999) # ("This function is only for IF")
  }
              region <- getRegionPSHAB(UserRegion)
              saturation_region <- getPSHABSubductionRegion(saturation_region_user)
  basin <- LeftNumber(basin, 1)

# Import Master Coefficient Table ----------------------------------------

  coefficients <- read.csv("PSHAB20_Table_E1_Interface_Coefficients_OneRowHeader.csv")

  #Isolate desired period
              coefficients.T <- coefficients[coefficients[1] == period,]

  #Define Mb
              Mb <- getMbDefault("PSHAB20",EventType, UserRegion, saturation_region)


# Constant ---------------------------------------------------------------

  if(region == "global" ){
    c0 <- as.numeric(subset(coefficients.T, select = Global_c0))
  } else {
    c0 <- as.numeric(subset(coefficients.T, select = c(eval(as.name(paste(saturation_region,"_c0",sep=""))))))
    }

# Path Term --------------------------------------------------------------
  h <- 10^(-0.82 + 0.252*M)
  Rref <- sqrt(1 + h^2)
  R <- sqrt(Rrup^2 + h^2)
  LogR <- log(R)
  R_Rref <- log(R/Rref)

  #Need  to isolate regional anelastic coefficient, a0
  if(region == "global" ){
    a0 <- as.numeric(subset(coefficients.T, select = c(Global_a0)))
  } else {
    a0 <- as.numeric(subset(coefficients.T, select = c(eval(as.name(paste(region,"_a0",sep=""))))))
  }

  if(is.na(a0)){
    a0 <- as.numeric(subset(coefficients.T, select = c(Global_a0)))
  }

  Fp <- as.numeric(coefficients.T$c1*LogR + (coefficients.T$b4*M)*R_Rref + a0*R)


# Magnitude Scaling ------------------------------------------------------

              func1 <- function(M,Mb){ifelse(M <= Mb,M-Mb,0)}
              func2 <- function(M,Mb){ifelse(M > Mb,M-Mb,0)}
```

```
            func3 <- function(M,Mb){ifelse(M <= Mb, (M-Mb)^2, 0)}
 #compute M-scaling term
 Fm <- coefficients.T$c4*func1(M,Mb) + coefficients.T$c6*func2(M,Mb) + coefficients.T$c5*func3(M,Mb)

# Linear Site Amplification ----------------------------------------------

 #Site Coefficients
 V1 <- coefficients.T$V1_m_s
 V2 <- coefficients.T$V2_m_s
 Vref <- coefficients.T$Vref_m_s

 if(region == "global"| region == "CAM"){
  s2 <- coefficients.T$Global_s2
  s1 <- s2
 } else  if(region =="Taiwan" || region == "Japan"){
  s2 <- as.numeric(subset(coefficients.T, select = c(eval(as.name(paste(region,"_s2",sep=""))))))
  s1 <- as.numeric(subset(coefficients.T, select = c(eval(as.name(paste(region,"_s1",sep=""))))))
 } else {
  s2 <- as.numeric(subset(coefficients.T, select = c(eval(as.name(paste(region,"_s2",sep=""))))))
  s1 <- s2
 }

 #Compute linear site term
  if(Vs30 <= V1){
   Flin <- s1*log(Vs30/V1) + s2*log(V1/Vref)
  } else  if(Vs30 <= V2){
   Flin <- s2*log(Vs30/Vref)
  } else {
   Flin <- s2*log(V2/Vref)
  }

# Nonlinear Site Term ----------------------------------------------------

 PGAr <- exp(PSHAB20_GMM_at_760_IF(EventType,UserRegion,saturation_region,Rrup,M,0))
 f3 <- 0.05
 Vb <- 200
 Vref.Fnl <- 760

 if(period >= 3){
  Fnl <- 0
 } else {
  f2 <- coefficients.T$f4*(exp(coefficients.T$f5*(min(Vs30,Vref.Fnl)-Vb))-exp(coefficients.T$f5*(Vref.Fnl-Vb)))
  Fnl <- 0 + f2*log((PGAr+f3)/f3)

 }

# Basin Term -------------------------------------------------------------

 #import coefficients

  if(tolower(Z2pt5) == 'default' || Z2pt5 <= 0 || (region != "Japan" & region != "Cascadia")){
  Fb <- 0
  } else {
  if(region == "Cascadia"){
   theta0 <- 3.94
   theta1 <- -0.42
   vmu <- 200
   vsig <- 0.2
```

```
    e1 <- coefficients.T$C_e1

    if(basin == 0){
      e3 <- coefficients.T$C_e3 + coefficients.T$del_None
      e2 <-  coefficients.T$C_e2 +  coefficients.T$del_None
    } else  if(basin == 1){
      e3 <- coefficients.T$C_e3 + coefficients.T$del_Seattle
      e2 <-  coefficients.T$C_e2 +  coefficients.T$del_Seattle
    } else {
      e3 <- coefficients.T$C_e3
      e2 <-  coefficients.T$C_e2
    }

   } else  if(region == "Japan"){
    theta0 <- 3.05
    theta1 <- -0.8
    vmu <- 500
    vsig <-0.33
    e3 <- coefficients.T$J_e3
    e2 <- coefficients.T$J_e2
    e1 <- coefficients.T$J_e1
   }


   Z2pt5.pred <- 10^(theta0 + theta1*(1 + erf((log10(Vs30) - log10(vmu))/(vsig*sqrt(2)))))
   delZ2pt5 <- log(Z2pt5) - log(Z2pt5.pred)

   if(delZ2pt5 <= (e1/e3)){
    Fb <- e1
   } else  if(delZ2pt5 >= (e2/e3)){
    Fb <- e2
   } else {
    Fb <- e3*delZ2pt5
   }
  }



# Add it all up! ------------------------------------------------------------

   mu <-   c0 + Fp + Fm  + Fnl + Fb + Flin

  return(mu)
}






PSHAB20_Median <-function(EventType, UserRegion, SubductionSlab, Rrup, Magnitude, Zhypo, period, Vs30, Z2pt5,
PNWbasinStrux) {
   # CODE DEVELOPED/IMPLEMENTED BY
   #         Silvia Mazzoni, 2020
   #          smazzoni@ucla.edu

   thisOut <- -999
   XinterpMin <- 0
```

```
   XinterpMax <- 10
   XinterpType <- "log"
   YinterpType <- "log"
   extrapolateType <- "extrapolate"

   Parameters <- read.csv('PSHAB20_Table_E2_Slab_Coefficients_OneRowHeader.csv')
   TvalueList <- simplify2array(Parameters[1])
   InterpArray <- interpolateFunction(period, TvalueList, XinterpType, extrapolateType, XinterpMin, XinterpMax)
   period0 <- TvalueList[InterpArray[1]]
   y0 <- PSHAB20_Median_AtTlist(EventType, UserRegion, SubductionSlab, Rrup, Magnitude, Zhypo, period0, Vs30, Z2pt5,
PNWbasinStrux)

   if(InterpArray[2] <= 0) {
      thisOut <- y0
   } else {
      period1 <- TvalueList[InterpArray[2]]
      y1 <- PSHAB20_Median_AtTlist(EventType, UserRegion, SubductionSlab, Rrup, Magnitude, Zhypo, period1, Vs30, Z2pt5,
PNWbasinStrux)

      if(tolower(LeftString(YinterpType, nchar("log"))) == tolower("log") ) {
         if(y0 <= 0  ) {
            y0 <- 0.000000001
         }
         if(y1 <= 0  ) {
            y1 <- 0.000000001
         }
         y0 <- log(y0)
         y1 <- log(y1)
      }
      thisOut <- y0 + (y1 - y0) * InterpArray[3]
      if(tolower(LeftString(YinterpType, nchar("log"))) == tolower("log") ) {
         thisOut <- exp(thisOut)
      }

   }

   return(thisOut)

}

PSHAB20_Median_AtTlist <-function(EventType, UserRegion, saturation_region, Rrup, M, Zhypo, period, Vs30, Z2pt5, basin)
{
   # CODE DEVELOPED/IMPLEMENTED BY
   #         Silvia Mazzoni, 2020
   #          smazzoni@ucla.edu
   if(LeftNumber(EventType, 1) == 1 || LeftNumber(EventType, 1) == 5 ) {
      PSHAB20_Median_AtTlistValue <- exp(PSHAB20_GMM_at_Vs30_Slab(EventType, UserRegion, saturation_region, Rrup,
M, Zhypo, period, Vs30, Z2pt5, basin))
   } else {
      PSHAB20_Median_AtTlistValue <- exp(PSHAB20_GMM_at_Vs30_IF(EventType, UserRegion, saturation_region, Rrup, M,
period, Vs30, Z2pt5, basin))
   }

}


PSHAB20_SigmaEpistemic <-function (Ts, EventType, saturation_region_user) {
```

```
  # CODE DEVELOPED/IMPLEMENTED BY
  #         Silvia Mazzoni, 2020
  #          smazzoni@ucla.edu

  PSHAB20_SigmaEpistemicValue = -999;


 if (LeftNumber(EventType, 1) == 0) {
    CoefficientsTable = read.csv("PSHAB20_EpistemicModelCoefficients_Interface.csv");
 } else {
    CoefficientsTable = read.csv("PSHAB20_EpistemicModelCoefficients_Slab.csv");
 }

  saturation_region = getPSHABSubductionRegion(saturation_region_user);
 if (saturation_region == "Taiwan_W" || saturation_region == "Taiwan_E") {
    saturation_region = "Taiwan";
 } else {

 }


  if (saturation_region == "New_Zealand_N" || saturation_region == "New_Zealand_S") {
           saturation_region = "global";
  }
  SigEp1 = getRangeValueRegion(CoefficientsTable, saturation_region,  'SigEp1');
  if (SigEp1 == -999) {
    saturation_region = "global";
    SigEp1 = getRangeValueRegion(CoefficientsTable, saturation_region,  'SigEp1');
  }
  SigEp2 = getRangeValueRegion(CoefficientsTable, saturation_region, 'SigEp2');
  T1 = getRangeValueRegion(CoefficientsTable, saturation_region, 'T1');
  T2 = getRangeValueRegion(CoefficientsTable, saturation_region, 'T2');
  if (Ts < T1 || Ts == 0) {
    sigma_epi = SigEp1;
  } else if(Ts > T2) {
    sigma_epi = SigEp2;
  } else {
    sigma_epi = SigEp1 - (SigEp1 - SigEp2) * (log(Ts / T1) / log(T2 / T1));
  }
  PSHAB20_SigmaEpistemicValue = sigma_epi;

}

PSHAB20_SigmaAleatory <-function(period, Rrup, Vs30) {
  # CODE DEVELOPED/IMPLEMENTED BY
  #         Silvia Mazzoni, 2020
  #          smazzoni@ucla.edu

  thisOut <- -999
  XinterpMin <- 0
  XinterpMax <- 10
  XinterpType <- "log"
  YinterpType <- "linear"
  extrapolateType <- "extrapolate"

  Parameters <- read.csv('PSHAB20_Table_E2_Slab_Coefficients_OneRowHeader.csv')
  TvalueList <- simplify2array(Parameters[1])
  InterpArray <- interpolateFunction(period, TvalueList, XinterpType, extrapolateType, XinterpMin, XinterpMax)
```

```
  period0 <- TvalueList[InterpArray[1]]
  y0 <- PSHAB20_SigmaAleatoryAtTlist(period0, Rrup, Vs30)

  if(InterpArray[2] <= 0 ) {
    thisOut <- y0
  } else {
    period1 <- TvalueList[InterpArray[2]]
    y1 <- PSHAB20_SigmaAleatoryAtTlist(period1, Rrup, Vs30)
    if(tolower(LeftString(YinterpType, nchar("log"))) == tolower("log") ) {
      if(y0 <= 0  ) {
        y0 <- 0.000000001
      }
      if(y1 <= 0  ) {
        y1 <- 0.000000001
      }
      y0 <- log(y0)
      y1 <- log(y1)
    }
    thisOut <- y0 + (y1 - y0) * InterpArray[3]
    if(tolower(LeftString(YinterpType, nchar("log"))) == tolower("log") ) {
      thisOut <- exp(thisOut)
    }

  }

  return(thisOut)

}

PSHAB20_SigmaAleatoryAtTlist <-function(period, Rrup, Vs30) {
  # CODE DEVELOPED/IMPLEMENTED BY
  #         Silvia Mazzoni, 2020
  #          smazzoni@ucla.edu

  PhiTot <- PSHAB20_PhiTot(period, Rrup, Vs30)
  Tau <- PSHAB20_Tau(period, Rrup, Vs30)


  PSHAB20_SigmaAleatoryAtTlistValue <- sqrt(PhiTot * PhiTot + Tau * Tau)

}



PSHAB20_Tau <-function(period, Rrup, Vs30) {
  # CODE DEVELOPED/IMPLEMENTED BY
  #         Silvia Mazzoni, 2020
  #          smazzoni@ucla.edu

  SigmacoefficientsTable <- read.csv("PSHAB20_Table_E3_AleatoryCoefficients_OneRowHeader.csv")
  Tau <- getRangeValueT(SigmacoefficientsTable, period, "Tau")
  PSHAB20_TauValue <- Tau
}


PSHAB20_PhiTot <-function(period, Rrup, Vs30) {
  # CODE DEVELOPED/IMPLEMENTED BY
```

```
  #         Silvia Mazzoni, 2020
  #          smazzoni@ucla.edu

  PSHAB20_PhiTotValue <- -999
  SigmacoefficientsTable <- read.csv("PSHAB20_Table_E3_AleatoryCoefficients_OneRowHeader.csv")

  PhiTot_Phi1squared <- getRangeValueT(SigmacoefficientsTable, period, "PhiTot_Phi1squared")
  PhiTot_Phi2squared <- getRangeValueT(SigmacoefficientsTable, period, "PhiTot_Phi2squared")
  PhiTot_PhiVsquared <- getRangeValueT(SigmacoefficientsTable, period, "PhiTot_PhiVsquared")

  #Corner Distances:
  R1 <- 200    # km
  R2 <- 500    # km

  V1 <- 200    # M/s
  V2 <- 500    # M/s

  if(Vs30 <= V1 ) {
    Rprime <- max(R1, min(R2, Rrup))
    deltaVar <- PhiTot_PhiVsquared * (log(R2 / Rprime)) / (log(R2 / R1))
  } else if(Vs30 < V2 ) {
    Rprime <- max(R1, min(R2, Rrup))
    deltaVar <- PhiTot_PhiVsquared * (log(R2 / Rprime)) / (log(R2 / R1)) * (log(V2 / Vs30)) / (log(V2 / V1))
  } else {
    deltaVar <- 0
  }

  if(Rrup < R1 ) {
    phiSquared <- PhiTot_Phi1squared
  } else if(Rrup < R2 ) {
    phiSquared <- (PhiTot_Phi2squared - PhiTot_Phi1squared) / (log(R2 / R1)) * (log(Rrup / R1)) + PhiTot_Phi1squared
  } else {
    phiSquared <- PhiTot_Phi2squared

  }

  PSHAB20_PhiTotValue <- sqrt(phiSquared + deltaVar)

}

PSHAB20_SigmaS2S <-function(period, Rrup, Vs30) {
  # CODE DEVELOPED/IMPLEMENTED BY
  #         Silvia Mazzoni, 2020
  #          smazzoni@ucla.edu

  PSHAB20_SigmaS2SValue <- -999
  SigmacoefficientsTable <- read.csv("PSHAB20_Table_E3_AleatoryCoefficients_OneRowHeader.csv")

  PhiS2S_PhiS2S0squared <- getRangeValueT(SigmacoefficientsTable, period, "PhiS2S_PhiS2S0squared")
  PhiS2S_a1 <- getRangeValueT(SigmacoefficientsTable, period, "PhiS2S_a1")
  VM <- getRangeValueT(SigmacoefficientsTable, period, "VM")

  #Corner Distances:
  R3 <- 200    # km
  R4 <- 500    # km

  V3 <- 200    # M/s
  V4 <- 800    # M/s
```

```
  if(Vs30 <= V3 ) {
    Rprime <- max(R3, min(R4, Rrup))
    deltaVarS2S <- PhiS2S_a1 * log(V3 / VM) * (log(R4 / Rprime)) / (log(R4 / R3))
  } else if(Vs30 < VM ) {
    Rprime <- max(R3, min(R4, Rrup))
    deltaVarS2S <- PhiS2S_a1 * log(Vs30 / VM) * (log(R4 / Rprime)) / (log(R4 / R3))
  } else if(Vs30 < V4 ) {
    deltaVarS2S <- PhiS2S_a1 * log(Vs30 / VM)
  } else {
    deltaVarS2S <- PhiS2S_a1 * log(V4 / VM)
  }

  PSHAB20_SigmaS2SValue <- sqrt(PhiS2S_PhiS2S0squared + deltaVarS2S)

}


PSHAB20_SigmaSS <-function(period, Rrup, Vs30) {
  # CODE DEVELOPED/IMPLEMENTED BY
  #         Silvia Mazzoni, 2020
  #          smazzoni@ucla.edu

  PSHAB20_SigmaSSValue <- -999
  SigmacoefficientsTable <- read.csv("PSHAB20_Table_E3_AleatoryCoefficients_OneRowHeader.csv")

  PhiSS_PhiSS1squared <- getRangeValueT(SigmacoefficientsTable, period, "PhiSS_PhiSS1squared")
  PhiSS_PhiSS2squared <- getRangeValueT(SigmacoefficientsTable, period, "PhiSS_PhiSS2squared")
  PhiSS_a2 <- getRangeValueT(SigmacoefficientsTable, period, "PhiSS_a2")
  VM <- getRangeValueT(SigmacoefficientsTable, period, "VM")

  #Corner Distances:
  R3 <- 200    # km
  R4 <- 500    # km
  R5 <- 500    # km
  R6 <- 800    # km

  V3 <- 200    # M/s
  V4 <- 800    # M/s

  if(Rrup < R5 ) {
    phiSquaredSS <- PhiSS_PhiSS1squared
  } else if(Rrup < R6 ) {
    phiSquaredSS <- (PhiSS_PhiSS2squared - PhiSS_PhiSS1squared) / (log(R6 / R5)) * (log(Rrup / R5)) + PhiSS_PhiSS1squared
  } else {
    phiSquaredSS <- PhiSS_PhiSS2squared

  }

  if(Vs30 <= V3 ) {
    Rprime <- max(R3, min(R4, Rrup))
    deltaVarSS <- PhiSS_a2 * log(V3 / VM) * (log(R4 / Rprime)) / (log(R4 / R3))
  } else if(Vs30 < VM ) {
    Rprime <- max(R3, min(R4, Rrup))
    deltaVarSS <- PhiSS_a2 * log(Vs30 / VM) * (log(R4 / Rprime)) / (log(R4 / R3))
  } else if(Vs30 < V4 ) {
    deltaVarSS <- PhiSS_a2 * log(Vs30 / VM)
  } else {
```

```
    deltaVarSS <- PhiSS_a2 * log(V4 / VM)
  }


  PSHAB20_SigmaSSValue <- sqrt(phiSquaredSS + deltaVarSS)

}



PSHAB20_SigmaTotal <- function(period, Rrup,  Vs30, EventType,   SubductionSlab ) {
  # CODE DEVELOPED/IMPLEMENTED BY
  #         Silvia Mazzoni, 2020
  #          smazzoni@ucla.edu
  SigmaAleatory <- PSHAB20_SigmaAleatory(period, Rrup, Vs30);
  SigmaEpistemic <- PSHAB20_SigmaEpistemic(period, EventType, SubductionSlab);
  PSHAB20_SigmaTotalValue <- sqrt(SigmaAleatory * SigmaAleatory + SigmaEpistemic * SigmaEpistemic);

}



NGAsubGMM_MedianPlusMinusSigma <-function(UserPeriod, UserRegion, Magnitude, Vs30, Rrup, AlphaBackarc,
AlphaNankai, Ztor, Zhypo, EventType, Z1pt0, Z2pt5, MbUser, SubductionSlab, PNWbasinStrux, Weight_KBCG20,
Weight_PSHAB20, User_NSigma, User_EpiInSigmaModels, User_NsampleEpi ,User_MeanType = "geometric" ) {
  # CODE DEVELOPED/IMPLEMENTED BY
  #         Silvia Mazzoni, 2020
  #          smazzoni@ucla.edu

   NGAsubGMM_MedianValue <- 0
  if(tolower(MbUser) == tolower("default") ) {
    Mb <- getMbDefault("KBCG20",EventType, UserRegion, SubductionSlab)
  } else {
    Mb <- MbUser
  }


  period <- getTFvalue(UserPeriod)
  WeightedSum <- 0
  WeightSum <- 0

  if(Weight_KBCG20 > 0 ) {
    KBCG20median <- KBCG20_medPSA(period, Magnitude, Rrup, AlphaBackarc, AlphaNankai, Ztor, EventType, Vs30,
Z1pt0, Z2pt5, Mb, UserRegion, PNWbasinStrux)
    SigmaAleatoryKBCG20 <- KBCG20_sigmaAleatory(period)
    if (LeftNumber(User_EpiInSigmaModels, 1) == 1) {
      SigmaEpistemicKBCG20 <- KBCG20_SigmaEpistemic(period, Magnitude, Rrup, AlphaBackarc, AlphaNankai, Ztor,
EventType, Vs30, Z1pt0, Z2pt5, Mb, UserRegion, User_NsampleEpi, PNWbasinStrux)
    } else {
       SigmaEpistemicKBCG20 = 0;
    }
    KBCG20SigmaTotal = sqrt(SigmaAleatoryKBCG20 * SigmaAleatoryKBCG20 + SigmaEpistemicKBCG20 *
SigmaEpistemicKBCG20);
    KBCG20 = exp(log(KBCG20median)+User_NSigma*KBCG20SigmaTotal);
    WeightSum <- WeightSum + Weight_KBCG20
    if(tolower(User_MeanType) == "arithmetic" ) {
       WeightedSum <- WeightedSum + KBCG20 * Weight_KBCG20
    } else {
```

```
        WeightedSum <- WeightedSum + log(KBCG20) * Weight_KBCG20
      }
    }
    if(Weight_PSHAB20 > 0 ) {
      PSHAB20median <- PSHAB20_Median(EventType, UserRegion, SubductionSlab, Rrup, Magnitude, Zhypo, period, Vs30,
Z2pt5, PNWbasinStrux)
      SigmaAleatoryPSHAB20 <- PSHAB20_SigmaAleatory(period, Rrup, Vs30)
      if (LeftNumber(User_EpiInSigmaModels, 1) == 1)   {
       SigmaEpistemicPSHAB20 <- PSHAB20_SigmaEpistemic(period,EventType,SubductionSlab)
      } else {
       SigmaEpistemicPSHAB20 = 0;
      }
      PSHAB20SigmaTotal = sqrt(SigmaAleatoryPSHAB20 * SigmaAleatoryPSHAB20 + SigmaEpistemicPSHAB20 *
SigmaEpistemicPSHAB20);
      PSHAB20 = exp(log(PSHAB20median)+User_NSigma*PSHAB20SigmaTotal);
      WeightSum <- WeightSum + Weight_PSHAB20
      if(tolower(User_MeanType) == "arithmetic" ) {
        WeightedSum <- WeightedSum + PSHAB20 * Weight_PSHAB20
      } else {
        WeightedSum <- WeightedSum + log(PSHAB20) * Weight_PSHAB20
      }
    }
   if(WeightSum>0) {
      if(tolower(User_MeanType) == "arithmetic" ) {
        NGAsubGMM_MedianValue <- WeightedSum / WeightSum
      } else {
        NGAsubGMM_MedianValue <- exp(WeightedSum / WeightSum)
      }
    }
    return(NGAsubGMM_MedianValue)

}







# ################################################################
RunSensitivityStudy <-function() {
      print(' ### Performing Sensitivity Study')

                        outArrayAll <- 0
                        inArrayAll <-0

      for (iSet in 1:2 ){
        thisCase <- iSet
        if(iSet == 1) {    # interface

            CaseLabel <-
c("Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0
_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_s
ec=0.01","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,Seat
Basn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigm
a=1,T_sec=0.02","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interf
ace,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasi
n,Nsigma=1,T_sec=0.03","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=
```

0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.05","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.075","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.1","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.15","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.2","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.25","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.3","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.4","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.5","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.75","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=1","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=1.5","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=2","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=3","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=4","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=5","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=7.5","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=10","Mag=4,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=4.25,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=4.5,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=4.75,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=5,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=5.25,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=5.5,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=5.75,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=6,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=6.25,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=6.5,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_N

oBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=6.75,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=7,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=7.25,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=7.5,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=7.75,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8.25,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8.5,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8.75,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=9,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=9.25,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=10,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=13,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=16.9,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=21.97,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=28.561,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=37.1293,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=48.26809,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=62.74851699999999,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=81.5730720999999,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=106.04499373,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=137.858491849,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=179.2160394037,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=232.980851224809,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=302.875106592252,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=393.737638569928,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=511.858930140906,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma

=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=665.416609183177,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=865.04159193813,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=1124.55406951957,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=1461.92029037544,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=1900.49637748807,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=2470.64529073449,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=150,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=169.5,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=191.535,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=216.43455,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=244.5710415,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=276.365276895,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=312.29276289135,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=352.890822067225,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=398.766628935964,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=450.60629069764,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=509.185108488333,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=575.379172591816,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=650.178465028752,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=734.701665482489,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=830.212881995213,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=938.140556654591,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=1060.09882901969,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=1197.91167679225,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=1353.64019477524,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=1529.61342009602,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=1728.4631647085,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=

```
100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=1953.163376120
61,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,
Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30
_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_
m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag
=8,Vs30_mps=760,Reg=1_Alaska,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBa
sin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=8.6,Zhypo_km=55,SatReg=Alaska,Basin=0_NoBasin,Nsigma=1,T_sec=0.0
1","Mag=8,Vs30_mps=760,Reg=1_Alaska,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=
0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=8,Zhypo_km=55,SatReg=Aleutian,Basin=0_NoBasin,Nsigma=1,T
_sec=0.01","Mag=8,Vs30_mps=760,Reg=2_Cascadia,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface
,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.7,Zhypo_km=55,SatReg=Cascadia,Basin=0_NoBasin
,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=2_Cascadia,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra
=0_Interface,SeatBasn=1_InSeattleBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.7,Zhypo_km=55,SatReg=Cascadi
a,Basin=1_InSeattleBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=2_Cascadia,Rrup_km=200,alpBack=0,alpNank
=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=2_InOtherPNWbasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.7,Z
hypo_km=55,SatReg=Cascadia,Basin=2_InOtherPNWbasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=3_CentralAm
erica&Mexico,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=2_InOtherPNWbasin,Z1.0_
m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.4,Zhypo_km=55,SatReg=Central_America_N,Basin=0_NoBasin,Nsigma=1,T_sec
=0.01","Mag=8,Vs30_mps=760,Reg=3_CentralAmerica&Mexico,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=
0_Interface,SeatBasn=2_InOtherPNWbasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.4,Zhypo_km=55,SatReg=Cent
ral_America_S,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=4_Japan,Rrup_km=200,alpBack=0,alpN
ank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=2_InOtherPNWbasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=8
.5,Zhypo_km=55,SatReg=Japan_Pac,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=4_Japan,Rrup_k
m=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=2_InOtherPNWbasin,Z1.0_m=550,Z2.5_m=2000,
NsplEpi=100,Mbreak=7.7,Zhypo_km=55,SatReg=Japan_Phi,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760
,Reg=5_NewZealand,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=2_InOtherPNWbasin
,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=New_Zealand_N,Basin=0_NoBasin,Nsigma=1,T_
sec=0.01","Mag=8,Vs30_mps=760,Reg=5_NewZealand,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interf
ace,SeatBasn=2_InOtherPNWbasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=New_Zealan
d_S,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=6_SouthAmerica,Rrup_km=200,alpBack=0,alpNan
k=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=2_InOtherPNWbasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=8.5,
Zhypo_km=55,SatReg=South_America_N,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=6_SouthAm
erica,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=2_InOtherPNWbasin,Z1.0_m=550,Z
2.5_m=2000,NsplEpi=100,Mbreak=8.6,Zhypo_km=55,SatReg=South_America_S,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","M
ag=8,Vs30_mps=760,Reg=7_Taiwan,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=2_In
OtherPNWbasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.1,Zhypo_km=55,SatReg=Taiwan_E,Basin=0_NoBasin,Nsi
gma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=7_Taiwan,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_In
terface,SeatBasn=2_InOtherPNWbasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.1,Zhypo_km=55,SatReg=Taiwan_
W,Basin=0_NoBasin,Nsigma=1,T_sec=0.01");
	CaseLabelRegion <-
c("0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB
_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GL
B_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9"
,"0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_
Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB
_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","
0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_M
b7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_
GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0
_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb
7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_G
LB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_
GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7
.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GL
B_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_
GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","1_AK_AK_Mb8.6","1_AK_Aleu_Mb8","2_CSC_CSC_Mb7.7","2
_CSC_CSC_Mb7.7_1_InSeattleBasin","2_CSC_CSC_Mb7.7_2_InOtherPNWbasin","3_CAM_CAM_N_Mb7.4","3_CAM_CAM_S
```

```
_Mb7.4","4_JP_JP_Pac_Mb8.5","4_JP_JP_Phi_Mb7.7","5_NZ_NZ_N_Mb7.9","5_NZ_NZ_S_Mb7.9","6_SAM_SAM_N_Mb8.5"
,"6_SAM_SAM_S_Mb8.6","7_TW_TW_E_Mb7.1","7_TW_TW_W_Mb7.1");

        Mag <-
c(8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,4,4.25,4.5,4.75,5,5.25,5.5,5.75,6,6.25,6.5,6.75,7,7.25,7.5,7.75,8,8.25,8.5,8.75,9,9.25
,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8);
        Vs30mps <-
c(760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,
760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,76
0,760,760,760,760,760,760,760,760,150,169.5,191.535,216.43455,244.5710415,276.365276895,312.29276289135,352.890
822067225,398.766628935964,450.60629069764,509.185108488333,575.379172591816,650.178465028752,734.70166548
2489,830.212881995213,938.140556654591,1060.09882901969,1197.91167679225,1353.64019477524,1529.61342009602
,1728.4631647085,1953.16337612061,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760);
        Reg <-
c("0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_global",
"0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0
_global","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_g
lobal","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_glo
bal","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_globa
l","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_global",
"0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0
_global","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_global","1_Alaska","1_
Alaska","2_Cascadia","2_Cascadia","2_Cascadia","3_CentralAmerica&Mexico","3_CentralAmerica&Mexico","4_Japan","4_Ja
pan","5_NewZealand","5_NewZealand","6_SouthAmerica","6_SouthAmerica","7_Taiwan","7_Taiwan");
        Rrup_km <-
c(200,200,200,200,200,200,200,200,200,200,200,200,200,200,200,200,200,200,200,200,200,200,200,200,200,200,200,200,
200,200,200,200,200,200,200,200,200,200,200,200,200,200,200,10,13,16.9,21.97,28.561,37.1293,48.26809,62.7485169999
999,81.5730720999999,106.04499373,137.858491849,179.2160394037,232.980851224809,302.875106592252,393.737638
569928,511.858930140906,665.416609183177,865.04159193813,1124.55406951957,1461.92029037544,1900.4963774880
7,2470.64529073449,200,200,200,200,200,200,200,200,200,200,200,200,200,200,200,200,200,200,200,200,200,200,200,20
0,200,200,200,200,200,200,200,200,200,200,200,200,200,200);
        alpBack <-
c(0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0);
        alpNank <-
c(0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0);
        Ztor_km <-
c(10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,1
0,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,1
0,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10);
        SeatBasn <-
c("0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_
NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoB
asin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin"
,"0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_
NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoB
asin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin"
,"0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_
NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoB
asin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin"
,"0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","1_InSeattleBasin","2_InOther
PNWbasin","2_InOtherPNWbasin","2_InOtherPNWbasin","2_InOtherPNWbasin","2_InOtherPNWbasin","2_InOtherPNWbasin","2_InOtherPNWbasi
n","2_InOtherPNWbasin","2_InOtherPNWbasin","2_InOtherPNWbasin","2_InOtherPNWbasin","2_InOtherPNWbasin");
        Z1pt0m <-
c(550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,
550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,55
```

```
0,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,
550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550);
            Z2pt5m <-
c(2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000
,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,
2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2
000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,20
00,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000);
            NsplEpi <-
c(100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,
100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,10
0,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,
100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100);
            Mbreak <-
c(7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,
7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7
.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,8.6,8,7.7,7.7,7.7,7.4,7.4,8.5,7.7,7.9,7.9,8.5,
8.6,7.1,7.1);
            Zhypo_km <-
c(55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,5
5,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,5
5,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55);
            SatReg <-
c("global","global","global","global","global","global","global","global","global","global","global","global","global","global","
global","global","global","global","global","global","global","global","global","global","global","global","global","global","glo
bal","global","global","global","global","global","global","global","global","global","global","global","global","global","global
","global","global","global","global","global","global","global","global","global","global","global","global","global","global","g
lobal","global","global","global","global","global","global","global","global","global","global","global","global","global","glob
al","global","global","global","global","global","global","global","global","global","global","global","global","global","global",
"global","global","Alaska","Aleutian","Cascadia","Cascadia","Cascadia","Central_America_N","Central_America_S","Japan_P
ac","Japan_Phi","New_Zealand_N","New_Zealand_S","South_America_N","South_America_S","Taiwan_E","Taiwan_W");
            Basin <-
c("0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_
NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoB
asin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin"
,"0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_
NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoB
asin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin"
,"0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_
NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoB
asin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin"
,"0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","1_InSeattleBasin","2_InOther
PNWbasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_No
Basin","0_NoBasin");
            Nsigma <-
c(1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,
1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1);
            InterIntra <-
c("0_Interface","0_Interface","0_Interface","0_Interface","0_Interface","0_Interface","0_Interface","0_Interface","0_Interfa
ce","0_Interface","0_Interface","0_Interface","0_Interface","0_Interface","0_Interface","0_Interface","0_Interface","0_Inter
face","0_Interface","0_Interface","0_Interface","0_Interface","0_Interface","0_Interface","0_Interface","0_Interface","0_Int
erface","0_Interface","0_Interface","0_Interface","0_Interface","0_Interface","0_Interface","0_Interface","0_Interface","0_I
nterface","0_Interface","0_Interface","0_Interface","0_Interface","0_Interface","0_Interface","0_Interface","0_Interface","0
_Interface","0_Interface","0_Interface","0_Interface","0_Interface","0_Interface","0_Interface","0_Interface","0_Interface",
"0_Interface","0_Interface","0_Interface","0_Interface","0_Interface","0_Interface","0_Interface","0_Interface","0_Interfac
e","0_Interface","0_Interface","0_Interface","0_Interface","0_Interface","0_Interface","0_Interface","0_Interface","0_Interf
ace","0_Interface","0_Interface","0_Interface","0_Interface","0_Interface","0_Interface","0_Interface","0_Interface","0_Int
erface","0_Interface","0_Interface","0_Interface","0_Interface","0_Interface","0_Interface","0_Interface","0_Interface","0_I
```

```
nterface","0_Interface","0_Interface","0_Interface","0_Interface","0_Interface","0_Interface","0_Interface","0_Interface","0
_Interface","0_Interface","0_Interface","0_Interface","0_Interface","0_Interface");
            T_sec <-
c(0.01,0.02,0.03,0.05,0.075,0.1,0.15,0.2,0.25,0.3,0.4,0.5,0.75,1,1.5,2,3,4,5,7.5,10,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.0
1,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.0
1,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.0
1,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01);
        } else { # intraslab

        CaseLabel <-
c("Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=40,InterIntra=1_Intraslab,SeatBasn=0
_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_s
ec=0.01","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=40,InterIntra=1_Intraslab,Seat
Basn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigm
a=1,T_sec=0.02","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=40,InterIntra=1_Intrasl
ab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin
,Nsigma=1,T_sec=0.03","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=40,InterIntra=1
_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_
NoBasin,Nsigma=1,T_sec=0.05","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=40,Inte
rIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,B
asin=0_NoBasin,Nsigma=1,T_sec=0.075","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km
=40,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=
global,Basin=0_NoBasin,Nsigma=1,T_sec=0.1","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Zt
or_km=40,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,S
atReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.15","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNa
nk=0,Ztor_km=40,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_k
m=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.2","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0
,alpNank=0,Ztor_km=40,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zh
ypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.25","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=200,alp
Back=0,alpNank=0,Ztor_km=40,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak
=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.3","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=2
00,alpBack=0,alpNank=0,Ztor_km=40,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,M
break=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.4","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_
km=200,alpBack=0,alpNank=0,Ztor_km=40,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=
100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.5","Mag=8,Vs30_mps=760,Reg=0_global
,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=40,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,Ns
plEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.75","Mag=8,Vs30_mps=760,Reg=0
_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=40,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2
000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=1","Mag=8,Vs30_mps=760,Re
g=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=40,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_
m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=1.5","Mag=8,Vs30_mps=
760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=40,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550
,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=2","Mag=8,Vs30_
mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=40,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_
m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=3","Mag=8,
Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=40,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,
Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=4","M
ag=8,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=40,InterIntra=1_Intraslab,SeatBasn=0_NoB
asin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=5"
,"Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=40,InterIntra=1_Intraslab,SeatBasn=0_
NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_se
c=7.5","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=40,InterIntra=1_Intraslab,SeatBa
sn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=
1,T_sec=10","Mag=4,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,S
eatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsi
gma=1,T_sec=0.01","Mag=4.25,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_
Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_N
oBasin,Nsigma=1,T_sec=0.01","Mag=4.5,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,Inte
```

rIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=4.75,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=5,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=5.25,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=5.5,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=5.75,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=6,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=6.25,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=6.5,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=6.75,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=7,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=7.25,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=7.5,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=7.75,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8.25,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8.5,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8.75,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=9,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=9.25,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=40,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=52,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=67.6,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=87.88,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=114.244,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=148.5172,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=193.07236,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=250.994068,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=326.2922884,

alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=424.179974919999,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=551.433967395999,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=716.864157614799,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=931.923404899238,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=1211.50042636901,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=1574.95055427971,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=2047.43572056362,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=2661.66643673271,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=3460.16636775252,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=4498.21627807827,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=5847.68116150175,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=7601.98550995228,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=9882.58116293796,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=150,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=169.5,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=191.535,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=216.43455,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=244.5710415,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=276.365276895,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=312.292762891 35,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=352.890822067225,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=398.766628935964,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=450.60629069764,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=509.185108488333,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=575.379172591816,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=650.178465028752,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","

```
Mag=8,Vs30_mps=734.701665482489,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab
,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,N
sigma=1,T_sec=0.01","Mag=8,Vs30_mps=830.212881995213,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10
,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=glo
bal,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=938.140556654591,Reg=0_global,Rrup_km=200,alpBack=0,
alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zh
ypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=1060.09882901969,Reg=0_global,Rr
up_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplE
pi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=1197.911676
79225,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=5
50,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs
30_mps=1353.64019477524,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn
=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T
_sec=0.01","Mag=8,Vs30_mps=1529.61342009602,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra
=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=
0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=1728.4631647085,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,
Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55
,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=1953.16337612061,Reg=0_global,Rrup_km=200
,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbr
eak=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_k
m=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=1
00,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=1_Alaska
,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,Ns
plEpi=100,Mbreak=7.2,Zhypo_km=55,SatReg=Alaska,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=
1_Alaska,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m
=2000,NsplEpi=100,Mbreak=8,Zhypo_km=55,SatReg=Aleutian,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=
760,Reg=2_Cascadia,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=
550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.2,Zhypo_km=55,SatReg=Cascadia,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=
8,Vs30_mps=760,Reg=2_Cascadia,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=1_InSe
attleBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.2,Zhypo_km=55,SatReg=Cascadia,Basin=1_InSeattleBasin,Nsig
ma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=2_Cascadia,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_I
ntraslab,SeatBasn=2_InOtherPNWbasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.2,Zhypo_km=55,SatReg=Cascadi
a,Basin=2_InOtherPNWbasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=3_CentralAmerica&Mexico,Rrup_km=200,
alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=2_InOtherPNWbasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi
=100,Mbreak=7.4,Zhypo_km=55,SatReg=Central_America_N,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=7
60,Reg=3_CentralAmerica&Mexico,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=2_InO
therPNWbasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=Central_America_S,Basin=0_NoB
asin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=4_Japan,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntr
a=1_Intraslab,SeatBasn=2_InOtherPNWbasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=Ja
pan_Pac,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=4_Japan,Rrup_km=200,alpBack=0,alpNank=0
,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=2_InOtherPNWbasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhy
po_km=55,SatReg=Japan_Phi,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=5_NewZealand,Rrup_k
m=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=2_InOtherPNWbasin,Z1.0_m=550,Z2.5_m=2000,
NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=New_Zealand_N,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mp
s=760,Reg=5_NewZealand,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=2_InOtherPN
Wbasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=New_Zealand_S,Basin=0_NoBasin,Nsigm
a=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=6_SouthAmerica,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=
1_Intraslab,SeatBasn=2_InOtherPNWbasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.3,Zhypo_km=55,SatReg=Sout
h_America_N,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=6_SouthAmerica,Rrup_km=200,alpBack
=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=2_InOtherPNWbasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,M
break=7.2,Zhypo_km=55,SatReg=South_America_S,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=7_
Taiwan,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=2_InOtherPNWbasin,Z1.0_m=550,
Z2.5_m=2000,NsplEpi=100,Mbreak=7.7,Zhypo_km=55,SatReg=Taiwan_E,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,V
s30_mps=760,Reg=7_Taiwan,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=2_InOtherP
NWbasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.7,Zhypo_km=55,SatReg=Taiwan_W,Basin=0_NoBasin,Nsigma=1
,T_sec=0.01");
        CaseLabelRegion <-
c("0_GLB_GLB_Mb7.6","0_GLB_GLB_Mb7.6","0_GLB_GLB_Mb7.6","0_GLB_GLB_Mb7.6","0_GLB_GLB_Mb7.6","0_GLB_GLB
```

```
_Mb7.6","0_GLB_GLB_Mb7.6","0_GLB_GLB_Mb7.6","0_GLB_GLB_Mb7.6","0_GLB_GLB_Mb7.6","0_GLB_GLB_Mb7.6","0_GL
B_GLB_Mb7.6","0_GLB_GLB_Mb7.6","0_GLB_GLB_Mb7.6","0_GLB_GLB_Mb7.6","0_GLB_GLB_Mb7.6","0_GLB_GLB_Mb7.6"
,"0_GLB_GLB_Mb7.6","0_GLB_GLB_Mb7.6","0_GLB_GLB_Mb7.6","0_GLB_GLB_Mb7.6","0_GLB_GLB_Mb7.6","0_GLB_GLB_
Mb7.6","0_GLB_GLB_Mb7.6","0_GLB_GLB_Mb7.6","0_GLB_GLB_Mb7.6","0_GLB_GLB_Mb7.6","0_GLB_GLB_Mb7.6","0_GLB
_GLB_Mb7.6","0_GLB_GLB_Mb7.6","0_GLB_GLB_Mb7.6","0_GLB_GLB_Mb7.6","0_GLB_GLB_Mb7.6","0_GLB_GLB_Mb7.6","
0_GLB_GLB_Mb7.6","0_GLB_GLB_Mb7.6","0_GLB_GLB_Mb7.6","0_GLB_GLB_Mb7.6","0_GLB_GLB_Mb7.6","0_GLB_GLB_M
b7.6","0_GLB_GLB_Mb7.6","0_GLB_GLB_Mb7.6","0_GLB_GLB_Mb7.6","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_
GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0
_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb
7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_G
LB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.6","0_GLB_GLB_Mb7.6","0_GLB_GLB_Mb7.6","0_
GLB_GLB_Mb7.6","0_GLB_GLB_Mb7.6","0_GLB_GLB_Mb7.6","0_GLB_GLB_Mb7.6","0_GLB_GLB_Mb7.6","0_GLB_GLB_Mb7
.6","0_GLB_GLB_Mb7.6","0_GLB_GLB_Mb7.6","0_GLB_GLB_Mb7.6","0_GLB_GLB_Mb7.6","0_GLB_GLB_Mb7.6","0_GLB_GL
B_Mb7.6","0_GLB_GLB_Mb7.6","0_GLB_GLB_Mb7.6","0_GLB_GLB_Mb7.6","0_GLB_GLB_Mb7.6","0_GLB_GLB_Mb7.6","0_
GLB_GLB_Mb7.6","0_GLB_GLB_Mb7.6","0_GLB_GLB_Mb7.6","1_AK_AK_Mb7.2","1_AK_Aleu_Mb8","2_CSC_CSC_Mb7.2","2
_CSC_CSC_Mb7.2_1_InSeattleBasin","2_CSC_CSC_Mb7.2_2_InOtherPNWbasin","3_CAM_CAM_N_Mb7.4","3_CAM_CAM_S
_Mb7.6","4_JP_JP_Pac_Mb7.6","4_JP_JP_Phi_Mb7.6","5_NZ_NZ_N_Mb7.6","5_NZ_NZ_S_Mb7.6","6_SAM_SAM_N_Mb7.3"
,"6_SAM_SAM_S_Mb7.2","7_TW_TW_E_Mb7.7","7_TW_TW_W_Mb7.7");

        Mag <-
c(8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,4,4.25,4.5,4.75,5,5.25,5.5,5.75,6,6.25,6.5,6.75,7,7.25,7.5,7.75,8,8.25,8.5,8.75,9,9.25
,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8);
        Vs30mps <-
c(760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,
760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,76
0,760,760,760,760,760,760,760,760,150,169.5,191.535,216.43455,244.5710415,276.365276895,312.29276289135,352.890
822067225,398.766628935964,450.60629069764,509.185108488333,575.379172591816,650.178465028752,734.70166548
2489,830.212881995213,938.140556654591,1060.09882901969,1197.91167679225,1353.64019477524,1529.61342009602
,1728.4631647085,1953.16337612061,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760);
        Reg <-
c("0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_global",
"0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0
_global","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_g
lobal","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_glo
bal","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_globa
l","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_global",
"0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0
_global","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_global","1_Alaska","1_
Alaska","2_Cascadia","2_Cascadia","2_Cascadia","3_CentralAmerica&Mexico","3_CentralAmerica&Mexico","4_Japan","4_Ja
pan","5_NewZealand","5_NewZealand","6_SouthAmerica","6_SouthAmerica","7_Taiwan","7_Taiwan");
        Rrup_km <-
c(200,200,200,200,200,200,200,200,200,200,200,200,200,200,200,200,200,200,200,200,200,200,200,200,200,200,200,200,200,
200,200,200,200,200,200,200,200,200,200,200,200,200,200,40,52,67.6,87.88,114.244,148.5172,193.07236,250.994068
,326.2922884,424.179974919999,551.433967395999,716.864157614799,931.923404899238,1211.50042636901,1574.9505
5427971,2047.43572056362,2661.66643673271,3460.16636775252,4498.21627807827,5847.68116150175,7601.98550995
228,9882.58116293796,200,200,200,200,200,200,200,200,200,200,200,200,200,200,200,200,200,200,200,200,200,200,200,200,
200,200,200,200,200,200,200,200,200,200,200,200,200,200,200);
        alpBack <-
c(0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0);
        alpNank <-
c(0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0);
        Ztor_km <-
c(40,40,40,40,40,40,40,40,40,40,40,40,40,40,40,40,40,40,40,40,40,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,1
0,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,1
0,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10);
```

```
	SeatBasn <-
c("0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_
NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoB
asin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin"
,"0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_
NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoB
asin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin"
,"0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_
NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoB
asin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin"
,"0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","1_InSeattleBasin","2_InOther
PNWbasin","2_InOtherPNWbasin","2_InOtherPNWbasin","2_InOtherPNWbasin","2_InOtherPNWbasin","2_InOtherPNWbasi
n","2_InOtherPNWbasin","2_InOtherPNWbasin","2_InOtherPNWbasin","2_InOtherPNWbasin","2_InOtherPNWbasin");
	Z1pt0m <-
c(550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,
550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,55
0,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,
550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550);
	Z2pt5m <-
c(2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000
,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,
2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2
000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,20
00,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000);
	NsplEpi <-
c(100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,
100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,10
0,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,
100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100);
	Mbreak <-
c(7.6,7.6,7.6,7.6,7.6,7.6,7.6,7.6,7.6,7.6,7.6,7.6,7.6,7.6,7.6,7.6,7.6,7.6,7.6,7.6,7.6,7.6,7.6,7.6,7.6,7.6,7.6,7.6,7.6,7.6,7.6,7.6,
7.6,7.6,7.6,7.6,7.6,7.6,7.6,7.6,7.6,7.6,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.6,7
.6,7.6,7.6,7.6,7.6,7.6,7.6,7.6,7.6,7.6,7.6,7.6,7.6,7.6,7.6,7.6,7.6,7.6,7.6,7.6,7.6,7.2,8,7.2,7.2,7.2,7.4,7.6,7.6,7.6,7.6,7.6,7.3,
7.2,7.7,7.7);
	Zhypo_km <-
c(55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,5
5,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,5
5,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55);
	SatReg <-
c("global","global","global","global","global","global","global","global","global","global","global","global","global","global","
global","global","global","global","global","global","global","global","global","global","global","global","global","global","glo
bal","global","global","global","global","global","global","global","global","global","global","global","global","global","global
","global","global","global","global","global","global","global","global","global","global","global","global","global","global","g
lobal","global","global","global","global","global","global","global","global","global","global","global","global","global","glob
al","global","global","global","global","global","global","global","global","global","global","global","global","global","global",
"global","global","Alaska","Aleutian","Cascadia","Cascadia","Cascadia","Central_America_N","Central_America_S","Japan_P
ac","Japan_Phi","New_Zealand_N","New_Zealand_S","South_America_N","South_America_S","Taiwan_E","Taiwan_W");
	Basin <-
c("0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_
NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoB
asin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin"
,"0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_
NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoB
asin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin"
,"0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_
NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoB
asin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin"
,"0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","1_InSeattleBasin","2_InOther
```

```
PNWbasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_No
Basin","0_NoBasin");
        Nsigma <-
c(1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,
1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1);
        InterIntra <-
c("1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab
","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab
","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab
","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab
","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab
","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab
","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab
","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab
","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab
","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab
","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab
","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab
","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab");
        T_sec <-
c(0.01,0.02,0.03,0.05,0.075,0.1,0.15,0.2,0.25,0.3,0.4,0.5,0.75,1,1.5,2,3,4,5,7.5,10,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.0
1,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.0
1,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.0
1,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01);


        }
        NN <- NROW(T_sec)
        iRowStart <- 1
        iRowEnd <- NN
        for (irow in iRowStart:iRowEnd ){
          thisCaseLabel <- CaseLabel[irow]
          thisMagnitude <- Mag[irow]
          thisVs30 <- Vs30mps[irow]
          thisRegion <- Reg[irow]
          thisRrup <- Rrup_km[irow]
          thisAlphaBackarc <- alpBack[irow]
          thisAlphaNankai <- alpNank[irow]
          thisZtor <- Ztor_km[irow]
          thisSeatBasn <- SeatBasn[irow]
          thisZ1pt0 <- Z1pt0m[irow]
          thisZ2pt5 <- Z2pt5m[irow]
          thisNsampleEpi <- NsplEpi[irow]
          thisMb <- Mbreak[irow]
          thisZhypo <- Zhypo_km[irow]
          thisSubductionSlab <- SatReg[irow]
          thisPNWbasinStrux <- Basin[irow]
          thisNsigma <- Nsigma[irow]
          thisEventType <- InterIntra[irow]
          thisT <- T_sec[irow]
          hereKBCG20MedianPSA <- KBCG20_medPSA(thisT, thisMagnitude,thisRrup, thisAlphaBackarc, thisAlphaNankai,
thisZtor, thisEventType, thisVs30,thisZ1pt0,thisZ2pt5, thisMb,thisRegion,thisPNWbasinStrux)
          hereKBCG20Tau <- KBCG20_sigmaTau(thisT)
          hereKBCG20Phi <- KBCG20_sigmaPhi(thisT)
          hereKBCG20SigmaAleatory <- KBCG20_sigmaAleatory(thisT)
          hereKBCG20SigmaEpistemic <- KBCG20_SigmaEpistemic(thisT, thisMagnitude,thisRrup, thisAlphaBackarc,
thisAlphaNankai, thisZtor, thisEventType, thisVs30,thisZ1pt0,thisZ2pt5,
thisMb,thisRegion,thisNsampleEpi,thisPNWbasinStrux)
```

```
        hereKBCG20SigmaTotal <- KBCG20_SigmaTotal(thisT, thisMagnitude,thisRrup, thisAlphaBackarc, thisAlphaNankai,
thisZtor, thisEventType, thisVs30,thisZ1pt0,thisZ2pt5, thisMb,thisRegion,thisNsampleEpi,thisPNWbasinStrux)
        hereKBCG20MedianMinusNsigmaxSigmaTotal <- exp( log(hereKBCG20MedianPSA)-
thisNsigma*hereKBCG20SigmaTotal)
        hereKBCG20MedianPlusNsigmaxSigmaTotal <- exp(
log(hereKBCG20MedianPSA)+thisNsigma*hereKBCG20SigmaTotal)
        hereKBCG20MedianMinusNsigmaxSigmaAleatory <- exp( log(hereKBCG20MedianPSA)-
thisNsigma*hereKBCG20SigmaAleatory)
        hereKBCG20MedianPlusNsigmaxSigmaAleatory <- exp(
log(hereKBCG20MedianPSA)+thisNsigma*hereKBCG20SigmaAleatory)
        hereKBCG20MedianMinusNsigmaxSigmaEpistemic <- exp( log(hereKBCG20MedianPSA)-
thisNsigma*hereKBCG20SigmaEpistemic)
        hereKBCG20MedianPlusNsigmaxSigmaEpistemic <- exp(
log(hereKBCG20MedianPSA)+thisNsigma*hereKBCG20SigmaEpistemic)
        herePSHAB20MedianPSA <- PSHAB20_Median(thisEventType, thisRegion, thisSubductionSlab, thisRrup,
thisMagnitude, thisZhypo, thisT, thisVs30, thisZ2pt5, thisPNWbasinStrux)
        herePSHAB20Tau <- PSHAB20_Tau(thisT, thisRrup, thisVs30)
        herePSHAB20PhiTot <- PSHAB20_PhiTot(thisT, thisRrup, thisVs30)
        herePSHAB20PhiS2S <- PSHAB20_SigmaS2S(thisT, thisRrup, thisVs30)
        herePSHAB20PhiSS <- PSHAB20_SigmaSS(thisT, thisRrup, thisVs30)
        herePSHAB20SigmaAleatory <- PSHAB20_SigmaAleatory(thisT, thisRrup, thisVs30)
        herePSHAB20SigmaEpistemic <- PSHAB20_SigmaEpistemic(thisT, thisEventType, thisSubductionSlab);
        herePSHAB20SigmaTotal <- PSHAB20_SigmaTotal(thisT, thisRrup,  thisVs30, thisEventType, thisSubductionSlab);
        herePSHAB20MedianMinusNsigmaxSigmaTotal <- exp(log(herePSHAB20MedianPSA)-
thisNsigma*herePSHAB20SigmaTotal)
        herePSHAB20MedianPlusNsigmaxSigmaTotal <-
exp(log(herePSHAB20MedianPSA)+thisNsigma*herePSHAB20SigmaTotal)
        herePSHAB20MedianMinusNsigmaxSigmaAleatory <- exp(log(herePSHAB20MedianPSA)-
thisNsigma*herePSHAB20SigmaAleatory)
        herePSHAB20MedianPlusNsigmaxSigmaAleatory <-
exp(log(herePSHAB20MedianPSA)+thisNsigma*herePSHAB20SigmaAleatory)
        herePSHAB20MedianMinusNsigmaxSigmaEpistemic <- exp(log(herePSHAB20MedianPSA)-
thisNsigma*herePSHAB20SigmaEpistemic)
        herePSHAB20MedianPlusNsigmaxSigmaEpistemic <-
exp(log(herePSHAB20MedianPSA)+thisNsigma*herePSHAB20SigmaEpistemic)
                                        SensitivityRun <-
c(hereKBCG20MedianPSA,hereKBCG20Tau,hereKBCG20Phi,hereKBCG20SigmaAleatory,hereKBCG20SigmaEpistemic,hereKBC
G20SigmaTotal,hereKBCG20MedianMinusNsigmaxSigmaTotal,hereKBCG20MedianPlusNsigmaxSigmaTotal,hereKBCG20Medi
anMinusNsigmaxSigmaAleatory,hereKBCG20MedianPlusNsigmaxSigmaAleatory,hereKBCG20MedianMinusNsigmaxSigmaEpi
stemic,hereKBCG20MedianPlusNsigmaxSigmaEpistemic,herePSHAB20MedianPSA,herePSHAB20Tau,herePSHAB20PhiTot,her
ePSHAB20PhiS2S,herePSHAB20PhiSS,herePSHAB20SigmaAleatory,herePSHAB20SigmaEpistemic,herePSHAB20SigmaTotal,he
rePSHAB20MedianMinusNsigmaxSigmaTotal,herePSHAB20MedianPlusNsigmaxSigmaTotal,herePSHAB20MedianMinusNsig
maxSigmaAleatory,herePSHAB20MedianPlusNsigmaxSigmaAleatory,herePSHAB20MedianMinusNsigmaxSigmaEpistemic,her
ePSHAB20MedianPlusNsigmaxSigmaEpistemic)
                                        outArrayAll <- rbind(outArrayAll,SensitivityRun)
                                        inArray <-
c(thisMagnitude,thisVs30,thisRegion,thisRrup,thisAlphaBackarc,thisAlphaNankai,thisZtor,thisSeatBasn,thisZ1pt0,thisZ2pt5,t
hisNsampleEpi,thisMb,thisZhypo,thisSubductionSlab,thisPNWbasinStrux,thisNsigma,thisEventType,thisT)
                                        inArrayAll <- rbind(inArrayAll,inArray)

        }   # for irow
    }   # for iset
            inSensitivityTable <- as.data.frame(inArrayAll)
            inArrayHeader <-
c('Mag','Vs30mps','Reg','Rrup_km','alpBack','alpNank','Ztor_km','SeatBasn','Z1pt0m','Z2pt5m','NsplEpi','Mbreak','Zhypo_km'
,'SatReg','Basin','Nsigma','InterIntra','T_sec')
            colnames(inSensitivityTable) <- inArrayHeader
            #print('Sensitivity-Study Input:')
```

```
            #print(inSensitivityTable, row.names <- FALSE)

            outSensitivityTable <- as.data.frame(outArrayAll)
    outDataTableHeader <-
c('KBCG20MedianPSA','KBCG20Tau','KBCG20Phi','KBCG20SigmaAleatory','KBCG20SigmaEpistemic','KBCG20SigmaTotal','KBC
G20MedianMinusNsigmaxSigmaTotal','KBCG20MedianPlusNsigmaxSigmaTotal','KBCG20MedianMinusNsigmaxSigmaAleator
y','KBCG20MedianPlusNsigmaxSigmaAleatory','KBCG20MedianMinusNsigmaxSigmaEpistemic','KBCG20MedianPlusNsigmaxSi
gmaEpistemic','PSHAB20MedianPSA','PSHAB20Tau','PSHAB20PhiTot','PSHAB20PhiS2S','PSHAB20PhiSS','PSHAB20SigmaAleat
ory','PSHAB20SigmaEpistemic','PSHAB20SigmaTotal','PSHAB20MedianMinusNsigmaxSigmaTotal','PSHAB20MedianPlusNsig
maxSigmaTotal','PSHAB20MedianMinusNsigmaxSigmaAleatory','PSHAB20MedianPlusNsigmaxSigmaAleatory','PSHAB20Medi
anMinusNsigmaxSigmaEpistemicKBCG20','PSHAB20MedianPlusNsigmaxSigmaEpistemicKBCG20')
            colnames(outSensitivityTable) <- outDataTableHeader
            #print('Sensitivity-Study Output:')
            #print(outSensitivityTable, row.names <- FALSE)

    outArrayAllData <- rbind(t(inArrayAll),t(outArrayAll))
            outSensitivityAllDataTableT <- as.data.frame(outArrayAllData)
    outDataTableHeaderAll <- c(inArrayHeader,outDataTableHeader)
            row.names(outSensitivityAllDataTableT) <- outDataTableHeaderAll
            print('Sensitivity-Study Input/Output:')
            #print(outSensitivityAllDataTableT)
            outputFilename <- 'NGAsubduction_GMMtool_R_Out_SensitivityStudy.csv'
            write.csv(outSensitivityAllDataTableT,outputFilename, row.names = TRUE)
    print(paste("Sensitivity-Study Results have been saved to " , outputFilename))

}

# ################################################################
# ################################################################
# the following commands compute the output values at each period and generate a figure of the data
# The user may edit these lines, if(needed
# ################################################################
# Compute Values at each Period

ComputeScenario <-
function(User_Region,User_SubductionSlab,User_Magnitude,User_Vs30,User_Rrup,User_AlphaBackarc,User_AlphaNankai,
User_Ztor,User_Zhypo,User_EventType,User_Z1pt0,User_Z2pt5,User_Mb,User_PNWbasinStrux,User_RelativeWeight_KBCG
20,User_RelativeWeight_PSHAB20,User_EpiInSigmaModels,User_NsampleEpi,User_Nsigma,User_PeriodList) {
            outTlist<-0
            outDataScenarioMatrix <-0
            outIndivModelData <-0
            inputArrayData <-0
            InputMatrix <-0
  print( ' ### Computing Spectra for Single Scenario')
  for (iRow in 1:length(User_PeriodList) ){
    thisT <- getTFvalue(User_PeriodList[iRow])
    outTlist[iRow] <- thisT
    PSA <- NGAsubGMM_MedianPlusMinusSigma(thisT,
User_Region,User_Magnitude,User_Vs30,User_Rrup,User_AlphaBackarc,User_AlphaNankai,User_Ztor,User_Zhypo,User_Ev
entType,User_Z1pt0,User_Z2pt5,User_Mb,User_SubductionSlab,User_PNWbasinStrux,User_RelativeWeight_KBCG20,User_
RelativeWeight_PSHAB20, 0,0,0);
    outMedianMinusNSigma <- NGAsubGMM_MedianPlusMinusSigma(thisT,
User_Region,User_Magnitude,User_Vs30,User_Rrup,User_AlphaBackarc,User_AlphaNankai,User_Ztor,User_Zhypo,User_Ev
entType,User_Z1pt0,User_Z2pt5,User_Mb,User_SubductionSlab,User_PNWbasinStrux,User_RelativeWeight_KBCG20,User_
RelativeWeight_PSHAB20, -1*User_Nsigma,User_EpiInSigmaModels,User_NsampleEpi);
    outMedianPlusNSigma <- NGAsubGMM_MedianPlusMinusSigma(thisT,
User_Region,User_Magnitude,User_Vs30,User_Rrup,User_AlphaBackarc,User_AlphaNankai,User_Ztor,User_Zhypo,User_Ev
```

```
entType,User_Z1pt0,User_Z2pt5,User_Mb,User_SubductionSlab,User_PNWbasinStrux,User_RelativeWeight_KBCG20,User_
RelativeWeight_PSHAB20, +1*User_Nsigma,User_EpiInSigmaModels,User_NsampleEpi);
      outArray <- c(thisT,PSA,outMedianMinusNSigma,outMedianPlusNSigma)
            outDataScenarioMatrix <- rbind(outDataScenarioMatrix,outArray)

     inputArrayData <- c(thisT,
User_Region,User_Magnitude,User_Vs30,User_Rrup,User_AlphaBackarc,User_AlphaNankai,User_Ztor,User_Zhypo,User_Ev
entType,User_Z1pt0,User_Z2pt5,User_Mb,User_SubductionSlab,User_PNWbasinStrux,User_RelativeWeight_KBCG20,User_
RelativeWeight_PSHAB20,User_Nsigma)
     InputMatrix <- rbind(InputMatrix,inputArrayData)

      #### Individual-Model Data:
      Prog_MbDefault <- getMbDefault("KBCG20",User_EventType,User_Region,User_SubductionSlab)
      if(tolower(User_Mb) == "default") {
        inUser_Mb <- Prog_MbDefault
      } else {
        inUser_Mb <- User_Mb
      }

      outKBCG20_MedianPSA  <- KBCG20_medPSA(thisT, User_Magnitude,User_Rrup, User_AlphaBackarc,
User_AlphaNankai, User_Ztor, User_EventType, User_Vs30,User_Z1pt0,User_Z2pt5,
inUser_Mb,User_Region,User_PNWbasinStrux)
      outKBCG20_Tau            <- KBCG20_sigmaTau(thisT)
      outKBCG20_Phi            <- KBCG20_sigmaPhi(thisT)
      outKBCG20_SigmaAleatory          <- KBCG20_sigmaAleatory(thisT)
      outKBCG20_SigmaEpistemic         <- KBCG20_SigmaEpistemic(thisT, User_Magnitude,User_Rrup, User_AlphaBackarc,
User_AlphaNankai, User_Ztor, User_EventType, User_Vs30,User_Z1pt0,User_Z2pt5,
inUser_Mb,User_Region,User_NsampleEpi,User_PNWbasinStrux)
      outKBCG20_SigmaTotal  <- KBCG20_SigmaTotal(thisT, User_Magnitude,User_Rrup, User_AlphaBackarc,
User_AlphaNankai, User_Ztor, User_EventType, User_Vs30,User_Z1pt0,User_Z2pt5,
inUser_Mb,User_Region,User_NsampleEpi,User_PNWbasinStrux)
      outKBCG20_Median_minus_1_SigmaTotal    <- exp( log(outKBCG20_MedianPSA)-
User_Nsigma*outKBCG20_SigmaTotal)
      outKBCG20_Median_plus_1_SigmaTotal      <- exp(
log(outKBCG20_MedianPSA)+User_Nsigma*outKBCG20_SigmaTotal)
      outKBCG20_Median_minus_1_SigmaAleatory            <- exp( log(outKBCG20_MedianPSA)-
User_Nsigma*outKBCG20_SigmaAleatory)
      outKBCG20_Median_plus_1_SigmaAleatory  <- exp(
log(outKBCG20_MedianPSA)+User_Nsigma*outKBCG20_SigmaAleatory)
      outKBCG20_Median_minus_1_SigmaEpistemic          <- exp( log(outKBCG20_MedianPSA)-
User_Nsigma*outKBCG20_SigmaEpistemic)
      outKBCG20_Median_plus_1_SigmaEpistemic           <- exp(
log(outKBCG20_MedianPSA)+User_Nsigma*outKBCG20_SigmaEpistemic)
      outPSHAB20_MedianPSA             <- PSHAB20_Median(User_EventType, User_Region, User_SubductionSlab,
User_Rrup, User_Magnitude, User_Zhypo, thisT, User_Vs30, User_Z2pt5, User_PNWbasinStrux)
      outPSHAB20_Tau          <- PSHAB20_Tau(thisT, User_Rrup, User_Vs30)
      outPSHAB20_PhiTot       <- PSHAB20_PhiTot(thisT, User_Rrup, User_Vs30)
      outPSHAB20_PhiS2S       <- PSHAB20_SigmaS2S(thisT, User_Rrup, User_Vs30)
      outPSHAB20_PhiSS        <- PSHAB20_SigmaSS(thisT, User_Rrup, User_Vs30)
      outPSHAB20_SigmaAleatory         <- PSHAB20_SigmaAleatory(thisT, User_Rrup, User_Vs30)
      outPSHAB20_SigmaEpistemic        <- PSHAB20_SigmaEpistemic(thisT, User_EventType, User_SubductionSlab)
      outPSHAB20_SigmaTotal <- PSHAB20_SigmaTotal(thisT, User_Rrup,  User_Vs30, User_EventType,
User_SubductionSlab);
      outPSHAB20_Median_minus_1_SigmaTotal  <- exp(log(outPSHAB20_MedianPSA)-
User_Nsigma*outPSHAB20_SigmaTotal)
      outPSHAB20_Median_plus_1_SigmaTotal    <-
exp(log(outPSHAB20_MedianPSA)+User_Nsigma*outPSHAB20_SigmaTotal)
```

```
      outPSHAB20_Median_minus_1_SigmaAleatory          <- exp(log(outPSHAB20_MedianPSA)-
User_Nsigma*outPSHAB20_SigmaAleatory)
      outPSHAB20_Median_plus_1_SigmaAleatory           <-
exp(log(outPSHAB20_MedianPSA)+User_Nsigma*outPSHAB20_SigmaAleatory)
      outPSHAB20_Median_minus_1_SigmaEpistemic         <- exp(log(outPSHAB20_MedianPSA)-
User_Nsigma*outKBCG20_SigmaEpistemic)
      outPSHAB20_Median_plus_1_SigmaEpistemic          <-
exp(log(outPSHAB20_MedianPSA)+User_Nsigma*outKBCG20_SigmaEpistemic)
      outIndivModelArray <-
c(thisT,outKBCG20_MedianPSA,outKBCG20_Tau,outKBCG20_Phi,outKBCG20_SigmaAleatory,outKBCG20_SigmaEpistemic,ou
tKBCG20_SigmaTotal,outKBCG20_Median_minus_1_SigmaTotal,outKBCG20_Median_plus_1_SigmaTotal,outKBCG20_Media
n_minus_1_SigmaAleatory,outKBCG20_Median_plus_1_SigmaAleatory,outKBCG20_Median_minus_1_SigmaEpistemic,outK
BCG20_Median_plus_1_SigmaEpistemic,outPSHAB20_MedianPSA,outPSHAB20_Tau,outPSHAB20_PhiTot,outPSHAB20_PhiS
2S,outPSHAB20_PhiSS,outPSHAB20_SigmaAleatory,outPSHAB20_SigmaEpistemic,outPSHAB20_SigmaTotal,outPSHAB20_Me
dian_minus_1_SigmaTotal,outPSHAB20_Median_plus_1_SigmaTotal,outPSHAB20_Median_minus_1_SigmaAleatory,outPSH
AB20_Median_plus_1_SigmaAleatory,outPSHAB20_Median_minus_1_SigmaEpistemic,outPSHAB20_Median_plus_1_SigmaE
pistemic)
      outIndivModelData <- rbind(outIndivModelData,outIndivModelArray)
  }

   # combine Data for output
   InputTableNames <-
c('User_Tperiod','User_Region','User_Magnitude','User_Vs30','User_Rrup','User_AlphaBackarc','User_AlphaNankai','User_Zt
or','User_Zhypo','User_EventType','User_Z1pt0','User_Z2pt5','User_Mb','User_SubductionSlab','User_PNWbasinStrux','User
_RelativeWeight_KBCG20','User_RelativeWeight_PSHAB20','Nsigma')
   InputTable <- as.data.frame(InputMatrix)
   colnames(InputTable) <- InputTableNames
   print('Scenario Input Data:')
   print(InputTable, row.names = FALSE)
   # weighted-average data
   outDataTableScenarioNames <- c('Period_(s)','PSA_median_(g)','PSA_median-N*Sigma_(g)','PSA_median+N*Sigma_(g)')
   outScenarioTable <- as.data.frame(outDataScenarioMatrix)
   colnames(outScenarioTable) <- outDataTableScenarioNames
   print('Scenario Weighted-Average Data:')
   print(outScenarioTable, row.names = FALSE)
   # Individual-Model Data:
   outIndivModelDataNames <-
c('Period_s','KBCG20_MedianPSA','KBCG20_Tau','KBCG20_Phi','KBCG20_SigmaAleatory','KBCG20_SigmaEpistemic','KBCG20_
SigmaTotal','KBCG20_Median_minus_1*SigmaTotal','KBCG20_Median_plus_1*SigmaTotal','KBCG20_Median_minus_1*Sigm
aAleatory','KBCG20_Median_plus_1*SigmaAleatory','KBCG20_Median_minus_1*SigmaEpistemic','KBCG20_Median_plus_1*
SigmaEpistemic','PSHAB20_MedianPSA','PSHAB20_Tau','PSHAB20_PhiTot','PSHAB20_PhiS2S','PSHAB20_PhiSS','PSHAB20_Sig
maAleatory','PSHAB20_SigmaEpistemic','PSHAB20_SigmaTotal','PSHAB20_Median_minus_1*SigmaTotal','PSHAB20_Median
_plus_1*SigmaTotal','PSHAB20_Median_minus_1*SigmaAleatory','PSHAB20_Median_plus_1*SigmaAleatory','PSHAB20_Me
dian_minus_1*SigmaEpistemic','PSHAB20_Median_plus_1*SigmaEpistemic')
   outIndivModelDataTable <- as.data.frame(outIndivModelData)
   colnames(outIndivModelDataTable) <- outIndivModelDataNames
   print('Scenario Individual-Model Data:')
   print(outIndivModelDataTable, row.names = FALSE)
}

# RUN ------------
ComputeScenario(User_Region,User_SubductionSlab,User_Magnitude,User_Vs30,User_Rrup,User_AlphaBackarc,User_Alph
aNankai,User_Ztor,User_Zhypo,User_EventType,User_Z1pt0,User_Z2pt5,User_Mb,User_PNWbasinStrux,User_RelativeWeig
ht_KBCG20,User_RelativeWeight_PSHAB20,User_EpiInSigmaModels,User_NsampleEpi,User_Nsigma,User_PeriodList)
if(tolower(RunSensitivityStudyFlag) == tolower('yes') ) {
        RunSensitivityStudy()
}
```

5.3 NGA-Sub Ground-Motion Characterization Tools – MATLAB

NGASubduction_GMMtool_Matlab_ r201031.m

```
%%% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%% -----------------------------------------------------------------
%%% -----------------------------------------------------------------
%%% -- NGA-Subduction:
%%%            Ground-Motion Characterization Tool
%%%            Matlab Version
%%% -----------------------------------------------------------------
%%%                    October 2020
%%% -----------------------------------------------------------------
%%% Ground-Motion Model Tool developed by
%%%        Silvia Mazzoni
%%%           smazzoni@ucla.edu
%%%        B. John Garrick Institute for the Risk Sciences
%%%           https://www.risksciences.ucla.edu/nhr3/gmtools
%%% -----------------------------------------------------------------
%%% Sources:
%%%  KBCG20:
%%%   Kuehn, Bozorgnia, Campbell, and Gregor,
%%%       “Partially Nonergodic Ground-Motion Model for Subduction Regions using NGA-Subduction Database,”
%%%       Report 4/2020, Pacific Earthquake Engineering Research Center, UC Berkeley
%%%  PSHAB20:
%%%   Parker, G.A., Stewart, J.P., Hassani, B., Atkinson, G.M., and Boore, D.M. (2020).
%%%       "NGA-Subduction Global Ground Motion Models with Regional Adjustment Factors."
%%%       Report xx/2020, Pacific Earthquake Engineering Research Center, UC Berkeley
%%% -----------------------------------------------------------------
%%% Instruction:
%%%  1. Enter the values of the user-input parameters.
%%%     Input ranges are given with each input parameters.
%%%  2. Run full script
%%%  3. Wait while script runs
%%%  4. A plot of the response spectra will open (you can copy and paste it elsewhere)
%%%  5. Result values are printed in Command Window (you can copy and
%%%  paste them elsewhere) and exported to a file
%%% -----------------------------------------------------------------
%%%  Notes:
%%%     The Median and Sigma models are described in the reports.
%%%     Damping Ratio = 5%
%%%       You may speed up the program by managing the coefficients
%%%       tables more efficiently.
%%%     The program computes the geometric mean of the median ground-motion models,
%%%        and the SRSS of the aleatory variability and epistemic uncertainty.
%%%
%%% -----------------------------------------------------------------
%%% Variables:
%%%  User-Input Parameters: given below
%%%  Output Variables:
%%%   InputTable:                  Matlab Table with all input variables
%%%   outIndivModelDataTable:      Matlab Table with all individual-model output
%%%   outDataTableScenario:        Matlab Table with the weighted-average model output
%%%   outAllScenarioDataTable:     Matlab Table with all input/output values. It written prented to a file.
%%%
%%% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% initialize
initializeTool; % do not remove this command
```

```
% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% user input:
% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
User_Region = '0_global';   % Region (Options: 0_global, 1_Alaska, 2_Cascadia, 3_CentralAmerica&Mexico, 4_Japan,
5_NewZealand (PSHAB20 uses global model), 6_SouthAmerica, 7_Taiwan)
User_SubductionSlab = 'global';   % Subducting Slab (used to compute Mb)
(Alaska,Aleutian,Cascadia,Central_America_N,Central_America_S,global,Japan_Pac,Japan_Phi,New_Zealand_N,New_Zealan
d_S,South_America_N,South_America_S,Taiwan_E,Taiwan_W)
User_Magnitude = 8;   % Moment Magnitude
User_Vs30 = 760;   % VS30 (m/sec)
User_Rrup = 200;   % Rupture Distance (km)
User_AlphaBackarc = 0;   % Fraction of Rrup in Backarc (Range: 0-1)
User_AlphaNankai = 0;   % Fraction of Rrup in Nankai Region (Range: 0-(1-AlphaBackArc))(Japan only)
User_Ztor = 10;   % Z_tor (km) (KBCG20 only)
User_Zhypo = 55;   % Hypocentral Depth (km) (PSHAB20 only)
User_EventType = '0_Interface';   % Event Type, (Options: 0_Interface, 1_Intraslab)
User_Z1pt0 = 550;   % Z1.0 input (m) (KBCG20 only)
User_Z2pt5 = 2000;   % Z2.5 input (m)
User_Mb = 7.9;   % Mb (KBCG20 only, can set = default)
User_PNWbasinStrux = '0_NoBasin';   % PNW Basin Structure (Options:
0_NoBasin,1_InSeattleBasin,2_InOtherPNWbasin)(Cascadia Only)
User_RelativeWeight_KBCG20 = 1;   % Relative Weight -- KBCG20
User_RelativeWeight_PSHAB20 = 0;   % Relative Weight -- PSHAB20
User_EpiInSigmaModels = '1_Yes';   % Apply KBCG20 Epistemic to Sigma Models (Options: 0_None, 1_AllModels)
User_NsampleEpi = 100;   % Number of Samples in Epistemic-Uncertainty Calculation (0=none, Range: 100-800)
User_Nsigma = 1;   % Number of Sigma away from Median




User_PeriodList = [0.01,0.02,0.03,0.05,0.075,0.1,0.15,0.2,0.25,0.3,0.4,0.5,0.75,1,1.5,2,3,4,5,7.5,10,"PGA","PGV"]; %
range:0.01-10sec (PGA: T=0, PGV: T=-1)



% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% the following commands compute the output values at each period and generate a figure of the data
% The user may edit these lines, if needed
% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Compute Values at each Period


  disp( ' %%% Computing Spectra for Single Scenario')
  for iRow = 1:length(User_PeriodList)
    thisT = getTvalue(User_PeriodList(iRow));
    outTlist(iRow) = thisT;

    outMedianList(iRow) = NGAsubGMM_MedianPlusMinusSigma(thisT,
User_Region,User_Magnitude,User_Vs30,User_Rrup,User_AlphaBackarc,User_AlphaNankai,User_Ztor,User_Zhypo,User_Ev
```

```
entType,User_Z1pt0,User_Z2pt5,User_Mb,User_SubductionSlab,User_PNWbasinStrux,User_RelativeWeight_KBCG20,User_
RelativeWeight_PSHAB20, 0,0,0);
    outMedianMinusNSigmaList(iRow) = NGAsubGMM_MedianPlusMinusSigma(thisT,
User_Region,User_Magnitude,User_Vs30,User_Rrup,User_AlphaBackarc,User_AlphaNankai,User_Ztor,User_Zhypo,User_Ev
entType,User_Z1pt0,User_Z2pt5,User_Mb,User_SubductionSlab,User_PNWbasinStrux,User_RelativeWeight_KBCG20,User_
RelativeWeight_PSHAB20, -1*User_Nsigma,User_EpiInSigmaModels,User_NsampleEpi);
    outMedianPlusNSigmaList(iRow) = NGAsubGMM_MedianPlusMinusSigma(thisT,
User_Region,User_Magnitude,User_Vs30,User_Rrup,User_AlphaBackarc,User_AlphaNankai,User_Ztor,User_Zhypo,User_Ev
entType,User_Z1pt0,User_Z2pt5,User_Mb,User_SubductionSlab,User_PNWbasinStrux,User_RelativeWeight_KBCG20,User_
RelativeWeight_PSHAB20, +1*User_Nsigma,User_EpiInSigmaModels,User_NsampleEpi);

    inputArray(iRow,:) = {thisT,
User_Region,User_Magnitude,User_Vs30,User_Rrup,User_AlphaBackarc,User_AlphaNankai,User_Ztor,User_Zhypo,User_Ev
entType,User_Z1pt0,User_Z2pt5,User_Mb,User_SubductionSlab,User_PNWbasinStrux,User_RelativeWeight_KBCG20,User_
RelativeWeight_PSHAB20,User_Nsigma};

    %%%% Individual-Model Data:
    Prog_MbDefault = getMbDefault("KBCG20",User_EventType,User_Region,User_SubductionSlab);
    if User_Mb>0
      inUser_Mb = User_Mb;
    else
      inUser_Mb = Prog_MbDefault;
    end
    outKBCG20_MedianPSA  = KBCG20_medPSA(thisT, User_Magnitude,User_Rrup, User_AlphaBackarc, User_AlphaNankai,
User_Ztor, User_EventType, User_Vs30,User_Z1pt0,User_Z2pt5, inUser_Mb,User_Region,User_PNWbasinStrux);
    outKBCG20_Tau            = KBCG20_sigmaTau(thisT);
    outKBCG20_Phi            = KBCG20_sigmaPhi(thisT);
    outKBCG20_SigmaAleatory          = KBCG20_sigmaAleatory(thisT);
    outKBCG20_SigmaEpistemic         = KBCG20_SigmaEpistemic(thisT, User_Magnitude,User_Rrup, User_AlphaBackarc,
User_AlphaNankai, User_Ztor, User_EventType, User_Vs30,User_Z1pt0,User_Z2pt5,
inUser_Mb,User_Region,User_NsampleEpi,User_PNWbasinStrux);
    outKBCG20_SigmaTotal  = KBCG20_SigmaTotal(thisT, User_Magnitude,User_Rrup, User_AlphaBackarc,
User_AlphaNankai, User_Ztor, User_EventType, User_Vs30,User_Z1pt0,User_Z2pt5,
inUser_Mb,User_Region,User_NsampleEpi,User_PNWbasinStrux);
    outKBCG20_Median_minus_1_SigmaTotal    = exp( log(outKBCG20_MedianPSA)-User_Nsigma*outKBCG20_SigmaTotal);
    outKBCG20_Median_plus_1_SigmaTotal      = exp(
log(outKBCG20_MedianPSA)+User_Nsigma*outKBCG20_SigmaTotal);
    outKBCG20_Median_minus_1_SigmaAleatory           = exp( log(outKBCG20_MedianPSA)-
User_Nsigma*outKBCG20_SigmaAleatory);
    outKBCG20_Median_plus_1_SigmaAleatory  = exp(
log(outKBCG20_MedianPSA)+User_Nsigma*outKBCG20_SigmaAleatory);
    outKBCG20_Median_minus_1_SigmaEpistemic          = exp( log(outKBCG20_MedianPSA)-
User_Nsigma*outKBCG20_SigmaEpistemic);
    outKBCG20_Median_plus_1_SigmaEpistemic           = exp(
log(outKBCG20_MedianPSA)+User_Nsigma*outKBCG20_SigmaEpistemic);
    outPSHAB20_MedianPSA             = PSHAB20_Median(User_EventType, User_Region, User_SubductionSlab,
User_Rrup, User_Magnitude, User_Zhypo, thisT, User_Vs30, User_Z2pt5, User_PNWbasinStrux);
    outPSHAB20_Tau          = PSHAB20_Tau(thisT, User_Rrup, User_Vs30);
    outPSHAB20_PhiTot       = PSHAB20_PhiTot(thisT, User_Rrup, User_Vs30);
    outPSHAB20_PhiS2S       = PSHAB20_SigmaS2S(thisT, User_Rrup, User_Vs30);
    outPSHAB20_PhiSS        = PSHAB20_SigmaSS(thisT, User_Rrup, User_Vs30);
    outPSHAB20_SigmaAleatory         = PSHAB20_SigmaAleatory(thisT, User_Rrup, User_Vs30);
    outPSHAB20_SigmaEpistemic = PSHAB20_SigmaEpistemic(thisT, User_EventType, User_SubductionSlab);
    outPSHAB20_SigmaTotal=PSHAB20_SigmaTotal(thisT, User_Rrup,  User_Vs30, User_EventType, User_SubductionSlab );
    outPSHAB20_Median_minus_1_SigmaTotal  = exp(log(outPSHAB20_MedianPSA)-
User_Nsigma*outPSHAB20_SigmaTotal);
    outPSHAB20_Median_plus_1_SigmaTotal    =
exp(log(outPSHAB20_MedianPSA)+User_Nsigma*outPSHAB20_SigmaTotal);
```

```
    outPSHAB20_Median_minus_1_SigmaAleatory          = exp(log(outPSHAB20_MedianPSA)-
User_Nsigma*outPSHAB20_SigmaAleatory);
    outPSHAB20_Median_plus_1_SigmaAleatory           =
exp(log(outPSHAB20_MedianPSA)+User_Nsigma*outPSHAB20_SigmaAleatory);
    outPSHAB20_Median_minus_1_SigmaEpistemic         = exp(log(outPSHAB20_MedianPSA)-
User_Nsigma*outPSHAB20_SigmaEpistemic);
    outPSHAB20_Median_plus_1_SigmaEpistemic          =
exp(log(outPSHAB20_MedianPSA)+User_Nsigma*outPSHAB20_SigmaEpistemic);
    outIndivModelData(iRow,:) =
[thisT,outKBCG20_MedianPSA,outKBCG20_Tau,outKBCG20_Phi,outKBCG20_SigmaAleatory,outKBCG20_SigmaEpistemic,outKBCG20_SigmaTotal,outKBCG20_Median_minus_1_SigmaTotal,outKBCG20_Median_plus_1_SigmaTotal,outKBCG20_Median_minus_1_SigmaAleatory,outKBCG20_Median_plus_1_SigmaAleatory,outKBCG20_Median_minus_1_SigmaEpistemic,outKBCG20_Median_plus_1_SigmaEpistemic,outPSHAB20_MedianPSA,outPSHAB20_Tau,outPSHAB20_PhiTot,outPSHAB20_PhiS2S,outPSHAB20_PhiSS,outPSHAB20_SigmaAleatory,outPSHAB20_SigmaEpistemic,outPSHAB20_SigmaTotal,outPSHAB20_Median_minus_1_SigmaTotal,outPSHAB20_Median_plus_1_SigmaTotal,outPSHAB20_Median_minus_1_SigmaAleatory,outPSHAB20_Median_plus_1_SigmaAleatory,outPSHAB20_Median_minus_1_SigmaEpistemic,outPSHAB20_Median_plus_1_SigmaEpistemic];
  end

  % combine Data for output
  InputTable = cell2table(inputArray);
  InputTable.Properties.VariableNames =
{'User_Tperiod','User_Region','User_Magnitude','User_Vs30','User_Rrup','User_AlphaBackarc','User_AlphaNankai','User_Ztor','User_Zhypo','User_EventType','User_Z1pt0','User_Z2pt5','User_Mb','User_SubductionSlab','User_PNWbasinStrux','User_RelativeWeight_KBCG20','User_RelativeWeight_PSHAB20','Nsigma'}';
  % weighted-average data
  outDataTableScenario = table(outTlist.',outMedianList.',outMedianMinusNSigmaList.',outMedianPlusNSigmaList.');
  outDataTableScenario.Properties.VariableNames = {'Period_s','PSA_median_g','PSA_median-
N*Sigma_g','PSA_median+N*Sigma_g'};
  % Individual-Model Data:
  outIndivModelDataTable = array2table(outIndivModelData);
  outIndivModelDataTable.Properties.VariableNames =
{'Period__s','KBCG20_MedianPSA','KBCG20_Tau','KBCG20_Phi','KBCG20_SigmaAleatory','KBCG20_SigmaEpistemic','KBCG20_SigmaTotal','KBCG20_Median_minus_1*SigmaTotal','KBCG20_Median_plus_1*SigmaTotal','KBCG20_Median_minus_1*SigmaAleatory','KBCG20_Median_plus_1*SigmaAleatory','KBCG20_Median_minus_1*SigmaEpistemic','KBCG20_Median_plus_1*SigmaEpistemic','PSHAB20_MedianPSA','PSHAB20_Tau','PSHAB20_PhiTot','PSHAB20_PhiS2S','PSHAB20_PhiSS','PSHAB20_SigmaAleatory','outPSHAB20_SigmaEpistemic','PSHAB20_SigmaTotal','PSHAB20_Median_minus_1*SigmaTotal','PSHAB20_Median_plus_1*SigmaTotal','PSHAB20_Median_minus_1*SigmaAleatory','PSHAB20_Median_plus_1*SigmaAleatory','PSHAB20_Median_minus_1*SigmaEpistemic','PSHAB20_Median_plus_1*SigmaEpistemic'};
  % combine
  outAllScenarioDataTable = [InputTable outDataTableScenario outIndivModelDataTable];

  disp('Scenario Input Data:');
  disp(InputTable(1,:));
  disp('Scenario Individual-Model Data:');
  disp(outIndivModelDataTable);
  disp('Scenario Weighted-Average Data:');
  disp(outDataTableScenario);
  outputFilename = 'NGAsubduction_GMMtool_Matlab_Out_ScenarioData.csv';
  writetable(outAllScenarioDataTable,outputFilename,'QuoteStrings',1);
  disp("Scenario-Study Results have been saved to " + outputFilename);



  % %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
  % ------------------------------------------------------------------
  figure()
  loglog(outTlist(1:end-2),outMedianList(1:end-2),'k','LineWidth',2)
```

```
   evalc('hold');;
   loglog(outTlist(1:end-2),outMedianMinusNSigmaList(1:end-2),'b--','LineWidth',2)
   loglog(outTlist(1:end-2),outMedianPlusNSigmaList(1:end-2),'b-.','LineWidth',2)
   legend("Median: M=" +User_Magnitude +",Rrup=" + User_Rrup + "km, Vs30=" + User_Vs30 + "m/s","Median-
"+User_Nsigma+"*Sigma","Median+"+User_Nsigma+"*Sigma",'Location','southwest')
   xlabel("Period (sec)")
   ylabel("PSA RotD50 5%damp (g)")
   grid on


%%%%%
% The following script performs a large sensitivity study (see last function in this file)
% Running this sensitivity study make take a few minutes.
% The results from this study are output to a screen a to a csv file

 RunSensitivityStudy;  % uncomment to run this


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%
% program DO NOT edit lines below!!!
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%

function initializeTool()
   clc
   clear all
   close all
   disp(' %%% -- NGA-Subduction: ');
   disp(' %%%                Ground-Motion Characterization Tool');
   disp(' %%%                smazzoni@ucla.edu');
   disp(' %%%');
end

function thisT = getTvalue(thisTin)
   % code developed and written by Silvia Mazzoni
   %      smazzoni@ucla.edu, April 2020
   if isstring(thisTin)==1
      if lower(thisTin) == lower("PGA")
         thisT=0;
      elseif lower(thisTin) == lower("PGV")
         thisT=-1;
      else
         thisT = double(thisTin);
      end
   else
      thisT = thisTin;
   end
end
```

```
% logistc hinge function
function loghingeValue = loghinge(x, x0, a, b0, b1, delta)
   % CODE DEVELOPED/IMPLEMENTED BY
   %         Silvia Mazzoni, 2020
   %          smazzoni@ucla.edu
  loghingeValue = (a + b0 * (x - x0) + (b1 - b0) * delta * log(1 + exp((x - x0) / delta)));
end

% interpolation of adjustment to magnitude break point
function interp_dmbValue = interp_dmb(period)
   % CODE DEVELOPED/IMPLEMENTED BY
   %         Silvia Mazzoni, 2020
   %          smazzoni@ucla.edu

  Aarray = [0.01, 1, 3, 10];
  Carray = [0, 0, -0.4, -0.4];
  if period == 0
    interp_dmbValue = interpolateArray(0.01, Aarray, Carray, "log", "linear", "constant");
  elseif period == -1
    interp_dmbValue = 0;
  else
    interp_dmbValue = interpolateArray(period, Aarray, Carray, "log", "linear", "constant");
  end
end

% interpolation of k1/k2 (values taken from Campbell and Bozorgnia (2014)
function interp_k1k2Value = interp_k1k2(period)
   % CODE DEVELOPED/IMPLEMENTED BY
   %         Silvia Mazzoni, 2020
   %          smazzoni@ucla.edu
  periods = [0.005, 0.01, 0.02, 0.03, 0.05, 0.075, 0.1, 0.15, 0.2, 0.25, 0.3, 0.4, 0.5, 0.75, 1., 1.5, 2., 3., 4., 5., 7.5, 10.];
  k1 = [865., 865., 865., 908., 1054., 1086., 1032., 878., 748., 654., 587., 503., 457., 410.,400., 400., 400., 400., 400., 400.,
400., 400.];
  k2 = [-1.186, -1.186, -1.219, -1.273, -1.346, -1.471, -1.624, -1.931, -2.188, -2.381, -2.518,-2.657, -2.669, -2.401, -1.955, -
1.025, -0.299, 0., 0., 0., 0., 0.];

  if period == 0
     ap_k1 = interpolateArray(0.005, periods, k1, "log", "linear", "constant");
     ap_k2 = interpolateArray(0.005, periods, k2, "log", "linear", "constant");
  elseif period == -1
     ap_k1 = 400.0;
     ap_k2 = -1.995;
  else
    ap_k1 = interpolateArray(period, periods, k1, "log", "linear", "constant");
     ap_k2 = interpolateArray(period, periods, k2, "log", "linear", "constant");
  end
    interp_k1k2Value = [ap_k1, ap_k2];
end


% funcation to calculate Z_1/Z2pt5 from Vs30
function calc_z_from_Vs30Value = calc_z_from_Vs30(Vs30, coeffs)
   % CODE DEVELOPED/IMPLEMENTED BY
```

```
   %         Silvia Mazzoni, 2020
   %          smazzoni@ucla.edu

  zref = coeffs(1) + (coeffs(2) - coeffs(1)) * exp((log(Vs30) - coeffs(3)) / coeffs(4)) / (1 + exp((log(Vs30) - coeffs(3)) / coeffs(4)));

  calc_z_from_Vs30Value = zref;

end




%%% function to calculate median prediction

%The following function calculates the median prediction of KBCG20, as described in chapter 4 of the PEER report.
%It takes as input the predictor variables, as well as a set of coefficients.
%Later, we define a function that acts as a wrapper around this function, and will take as input period, region index, and
select the appropriate coefficients and pass them on.
%
%The inputs are

% `m`: moment magnitude
% `rlist`: a vector of length 3, which contains the distance in subregion 1,2,3 relative to volcanic arc; `rlist = c(R1,R2,R3)`.
Typically, `R1=R3=0` (corresponding to forearc).
% `ztor`: depth to top of rupture in km.
% `EventType`: flag for interface (`EventType = 0`) and intraslab (`EventType = 1`). Must be 0 || 1.
% `Vs30`: $V_{S30}$ n m/s.
% `fx`: arc crossing flag. Must be 0 || 1.
% `delta_ln_z`: difference between natural log of observed Z1.0/Z2.5 value and reference Z1.0/Z2.5 from $V_{S30}$.
% `coeffs`: vector containing coefficients needed to calculate median prediction.
% `coeffs_attn`: vector of length 6 to calcualte anelastic attenuation.
% `mbreak` and `zbreak`: magnitude and depth scaling break point
% `k1` and `k2`: parameters needed for site amplification
% `nft1` and `nft2`: coefficients needed for pseudo-depth term.
% `pgarock`: median pga prediction at $V_{S30} = 1100$

% function to calculate median prediction

function KBCG20_medValue = KBCG20_med(M, Rrup, AlphaBackarc, AlphaNankai, Ztor, EventType, Vs30, delta_ln_z, coeffs,
coeffs_attn, coeffs_z, mbreak, zbreak, k1, k2, nft1, nft2, pgarock, region)
    % CODE DEVELOPED/IMPLEMENTED BY
   %         Silvia Mazzoni, 2020
   %          smazzoni@ucla.edu


 if LeftNumber(region, 1) == 1
     distR1 = 0;
     distR3 = 0;
  elseif LeftNumber(region, 1) == 2
     distR1 = 0;
     distR3 = 0;
  elseif LeftNumber(region, 1) == 3
     distR1 = AlphaBackarc * Rrup;
     distR3 = 0;
  elseif LeftNumber(region, 1) == 4
     distR1 = AlphaBackarc * Rrup;
     distR3 = AlphaNankai * Rrup;
```

```
  elseif LeftNumber(region, 1) == 5
     distR1 = 0;
     distR3 = 0;
  elseif LeftNumber(region, 1) == 6
     distR1 = AlphaBackarc * Rrup;
     distR3 = 0;
  elseif LeftNumber(region, 1) == 7
     distR1 = 0;
     distR3 = 0;
  else  % reg=0
     distR1 = AlphaBackarc * Rrup;
     distR3 = 0;
  end

   distR2 = Rrup - distR1 - distR3;



  theta10 = 0;
  vsrock = 1100;
  c = 1.88;
  n = 1.18;
  minmb = 6.;
  delta = 0.1;
  deltaz = 1;
  refzif = 15;
  refzslab = 50;
  thisFs = (LeftNumber(EventType, 1));

   %check if cross arc:
     fx = 1;
     if distR1 == 0 && distR2 == 0
        fx = 0;
     end
     if distR2 == 0 && distR3 == 0
        fx = 0;
     end
     if distR1 == 0 && distR3 == 0
        fx = 0;
     end

  fmag = (1 - thisFs) * loghinge(M, mbreak, coeffs(6) * (mbreak - minmb), coeffs(6), coeffs(8), delta) + thisFs * loghinge(M,
mbreak, coeffs(7) * (mbreak - minmb), coeffs(7), coeffs(8), delta);
  fgeom = (1 - thisFs) * (coeffs(3) + coeffs(5) * M) * log(Rrup + 10 ^ (nft1 + nft2 * (M - 6)));
  fgeom_slab = thisFs * (coeffs(4) + coeffs(5) * M) * log(Rrup + 10 ^ (nft1 + nft2 * (M - 6)));
  fdepth = (1 - thisFs) * loghinge(Ztor, zbreak, coeffs(12) * (zbreak - refzif), coeffs(12), theta10, deltaz) + thisFs *
loghinge(Ztor, zbreak, coeffs(13) * (zbreak - refzslab), coeffs(13), theta10, deltaz);


   DotProduct123 = distR1 * coeffs_attn(1) + distR2 * coeffs_attn(2) + distR3 * coeffs_attn(3);
   DotProduct456 = distR1 * coeffs_attn(4) + distR2 * coeffs_attn(5) + distR3 * coeffs_attn(6);

  fattn = fx * DotProduct123 + (1 - fx) * DotProduct456 + fx * coeffs(14);


  if Vs30 < k1
   fsite = coeffs(11) * log(Vs30 / k1) + k2 * (log(pgarock + c * (Vs30 / k1) ^ n) - log(pgarock + c));
  else
```

```
   fsite = (coeffs(11) + k2 * n) * log(Vs30 / k1);
  end

  fbasin = coeffs_z(1) + coeffs_z(2) * delta_ln_z;

  Median = (1 - thisFs) * coeffs(1) + thisFs * coeffs(2) + fmag + fgeom + fgeom_slab + fdepth + fattn + fsite + fbasin;
  KBCG20_medValue = Median;

end



%%% function to calculate median prediction for a given scenario and period
%This is a function that calculates median predictions of KBCG20 for a given scenario.
%I takes as input period, the predictor variables for the scenarios, selects the appropriate coefficients/parameters, and calls
the function defined in the previous section.
%
%The arguments are similar to the function before.

% `m`: moment magnitude
% `rlist`: a vector of length 3, which contains the distances in subregion 1,2,3 relative to volcanic arc; `rlist = array(R1,R2,R3)`.
for regions Alaska, Cascadia, New Zealand, and Taiwan, $R1 = R2 = 0$.
% `ztor`: depth to top of rupture in km.
% `EventType`: flag for interface (`EventType = 0`) and intraslab (`EventType = 1`). Must be 0 || 1.
% `Vs30`: $V_{S30}$ n m/s.
% `Z1pt0`: depth to a shear wave horizon of 1000 m/s, in m. Used for regions Alaska and New Zealand.
% `Z2pt5`: depth to a shear wave horizon of 2500 m/s, in m. Used for regions Cascadia and Japan.
% `fx`: arc crossing flag. Must be 0 || 1.
% `mb`: Magnitude scaling break point. Should be set regionally dependent based on Campbell (2020).

%The last input is a region index `region`, which is as follows:
%
%* 0: global
%* 1: Alaska
%* 2: Cascadia
%* 3: Central America & Mexico
%* 4: Japan
%* 5: New Zealand
%* 6: South America
%* 7: Taiwan
%
%The opional argumen `Seattle_Basin` is a flag that should be set to `TRUE`
%if the site is in the Seattle Basin, and `FALSE` otherwise. % 10/28/2020:
%this has changed
%This flag determines which basin depth amplification model is used for Cascadia.
%It does not have an impact on any other region.
%


function KBCG20_medPSAValue = KBCG20_medPSA(period, Magnitude, Rrup, AlphaBackarc, AlphaNankai, Ztor, EventType,
Vs30, Z1pt0, Z2pt5, Mb, UserRegion, PNWbasinStrux)
   % CODE DEVELOPED/IMPLEMENTED BY
   %        Silvia Mazzoni, 2020
   %         smazzoni@ucla.edu
```

```
   thisOut = -999;
   XinterpMin = 0;
   XinterpMax = 10;
   XinterpType = "log";
   YinterpType = "log";
   extrapolateType = "extrapolate";

   Parameters = readtable('NGAsubGMM_KBCG20_coefficients.csv','PreserveVariableNames',true);
   TvalueList = Parameters.T;
   InterpArray = interpolateFunction(period, TvalueList, XinterpType, extrapolateType, XinterpMin, XinterpMax);

   period0 = TvalueList(InterpArray(1), 1);
   y0 = KBCG20_medPSA_AtTlist(period0, Magnitude, Rrup, AlphaBackarc, AlphaNankai, Ztor, EventType, Vs30, Z1pt0, Z2pt5,
Mb, UserRegion, PNWbasinStrux);

   if InterpArray(2) <= 0
      thisOut = y0;
   else
      period1 = TvalueList(InterpArray(2), 1);
      y1 = KBCG20_medPSA_AtTlist(period1, Magnitude, Rrup, AlphaBackarc, AlphaNankai, Ztor, EventType, Vs30, Z1pt0,
Z2pt5, Mb, UserRegion, PNWbasinStrux);
      if lower(LeftString(YinterpType, strlength("log"))) == lower("log")
         if y0 <= 0
            y0 = 0.000000001;
         end
         if y1 <= 0
            y1 = 0.000000001;
         end
         y0 = log(y0);
         y1 = log(y1);
      end
      thisOut = y0 + (y1 - y0) * InterpArray(3);
      if lower(LeftString(YinterpType, strlength("log"))) == lower("log")
         thisOut = exp(thisOut);
      end

   end

   KBCG20_medPSAValue = thisOut;

end


% function to calculate median prediction using mean coefficients
function KBCG20_medPSA_AtTlistValue = KBCG20_medPSA_AtTlist(period, M, Rrup, AlphaBackarc, AlphaNankai, Ztor,
EventType, Vs30, Z1pt0, Z2pt5, Mb, region, PNWbasinStrux)
   % CODE DEVELOPED/IMPLEMENTED BY
   %        Silvia Mazzoni, 2020
   %         smazzoni@ucla.edu

   KBCG20_medPSA_AtTlistValue = -999;
  % need to add some checks for input (period, region)
  % coefficients to calculate zref from vs30
  pars_z_ja = [7.6893685375, 2.30258509299405, 6.3091864, 0.7528670225, 1.2952369625];
  pars_z_casc = [8.29404964010203, 2.30258509299405, 6.39692965521615, 0.27081459, 1.7381352625];
  pars_z_nz = [6.859789675, 2.30258509299405, 5.745692775, 0.91563524375, 1.03531412375];
  pars_z_tw = [6.30560665, 2.30258509299405, 6.1104992125, 0.43671102, 0.7229702975];
```

```
 Parameters = readtable('NGAsubGMM_KBCG20_coefficients.csv','PreserveVariableNames',true);
 parameters_zmod =
readtable("NGAsubGMM_KBCG20_params_Z_ALL_allregca_attn3_corrreg_cs_dmb.csv",'PreserveVariableNames',true);

 % calculate rock PGA
 period_used = 0.;
 vsrock = 1100;
 pars_period = getRangeRowT(Parameters, period_used);
 pars_period_zmod = getRangeRowT(parameters_zmod, period_used) ; % silviamazzoni: I added this

 coeffs = getSubArrayRange(pars_period, 2, 15);
 k1k2 = interp_k1k2(period_used);
 dmb = 0;

 delta_ln_z = 0;
 coeffs_z = [0, 0];
 coeffs_z2 = [0, 0];

 if LeftNumber(region, 1) == 1
  coeffs = updateArray(coeffs, [1, 2, 11], pars_period, [17, 24, 31]);
  coeffs_attn = makeArray(pars_period, [38, 45, 52, 59, 66, 73]);
 elseif LeftNumber(region, 1) == 2
  coeffs = updateArray(coeffs, [1, 2, 11], pars_period, [18, 25, 32]);
  coeffs_attn = makeArray(pars_period, [39, 46, 53, 60, 67, 74]);
  if LeftNumber(PNWbasinStrux, 1) == 1
     coeff_seattle = getRangeValueT(parameters_zmod, period_used, "mean_residual_Seattle_basin");
     coeffs_z2 = [coeff_seattle, 0];
  elseif LeftNumber(PNWbasinStrux, 1) == 2
     coeffs_z2 = makeArray(pars_period_zmod, [2, 3]);
  end
 elseif LeftNumber(region, 1) == 3
  coeffs = updateArray(coeffs, [1, 2, 11], pars_period, [19, 26, 33]);
  coeffs_attn = makeArray(pars_period, [40, 47, 54, 61, 68, 75]);
 elseif LeftNumber(region, 1) == 4
            dmb = interp_dmb(period_used);
  coeffs = updateArray(coeffs, [1, 2, 11], pars_period, [20, 27, 34]);
  coeffs_attn = makeArray(pars_period, [41, 48, 55, 62, 69, 76]);
  coeffs_z2 = makeArray(pars_period_zmod, [5, 6]);
 elseif LeftNumber(region, 1) == 5
  coeffs = updateArray(coeffs, [1, 2, 11], pars_period, [21, 28, 35]);
  coeffs_attn = makeArray(pars_period, [42, 49, 56, 63, 70, 77]);
  coeffs_z2 = makeArray(pars_period_zmod, [7, 8]);
 elseif LeftNumber(region, 1) == 6
            dmb = interp_dmb(period_used);
  coeffs = updateArray(coeffs, [1, 2, 11], pars_period, [22, 29, 36]);
  coeffs_attn = makeArray(pars_period, [43, 50, 57, 64, 71, 78]);
 elseif LeftNumber(region, 1) == 7
  coeffs = updateArray(coeffs, [1, 2, 11], pars_period, [23, 30, 37]);
  coeffs_attn = makeArray(pars_period, [44, 51, 58, 65, 72, 79]);
  coeffs_z2 = makeArray(pars_period_zmod, [9, 10]);
 else  % LeftNumber(region,1)==0
  coeffs_attn = makeArray(pars_period, [11, 11, 11, 11, 10, 11]);
  delta_ln_z = 0;
  coeffs_z = [0, 0];
 end


 delta_bz = makeArray(pars_period, [80, 81]);
```

```
  coeffs_nft = makeArray(pars_period, [82, 83]);
  thisFs = (LeftNumber(EventType, 1));
  mbreak = (1 - thisFs) * (Mb + dmb) + thisFs * Mb;
  zbreak = (1 - thisFs) * (30 + delta_bz(1)) + thisFs * (80 + delta_bz(2));

  mbreak_pga = mbreak;
  zbreak_pga = zbreak;
  k1k2_pga = k1k2;
  coeffs_pga = coeffs;   % silviamazzoni: I added this so I alway have coeffs
  coeffs_attn_pga = coeffs_attn;
  coeffs_z_pga = coeffs_z;
  coeffs_z_pga2 = coeffs_z2;
  coeffs_nft_pga = coeffs_nft;
  pgarock = exp(KBCG20_med(M, Rrup, AlphaBackarc, AlphaNankai, Ztor, EventType, vsrock, delta_ln_z, coeffs_pga,
coeffs_attn_pga, coeffs_z_pga, mbreak_pga, zbreak_pga, k1k2_pga(1), k1k2_pga(2), coeffs_nft_pga(1), coeffs_nft_pga(2), 0,
region));

  % calculate PSA
  period_used = period;
  pars_period = getRangeRowT(Parameters, period_used);
  pars_period_zmod = getRangeRowT(parameters_zmod, period_used) ;% silviamazzoni: I added this
  coeffs = getSubArrayRange(pars_period, 2, 15);
  k1k2 = interp_k1k2(period_used);
  dmb = 0;


    delta_ln_z = 0;
    coeffs_z = [0, 0];
  if LeftNumber(region, 1) == 1
    coeffs = updateArray(coeffs, [1, 2, 11], pars_period, [17, 24, 31]);
    coeffs_attn = makeArray(pars_period, [38, 45, 52, 59, 66, 73]);
   elseif LeftNumber(region, 1) == 2
    coeffs = updateArray(coeffs, [1, 2, 11], pars_period, [18, 25, 32]);
    coeffs_attn = makeArray(pars_period, [39, 46, 53, 60, 67, 74]);
    if lower(Z2pt5) == 'default' | Z2pt5<0
              delta_ln_z = 0;
    else
              delta_ln_z = log(Z2pt5) - calc_z_from_Vs30(Vs30, pars_z_casc);
    end

    if LeftNumber(PNWbasinStrux, 1) == 1
       coeff_seattle = getRangeValueT(parameters_zmod, period_used, "mean_residual_Seattle_basin");
      coeffs_z = [coeff_seattle, 0];
    elseif LeftNumber(PNWbasinStrux, 1) == 2
       coeffs_z = makeArray(pars_period_zmod, [2, 3]);
    end
   elseif LeftNumber(region, 1) == 3
    coeffs = updateArray(coeffs, [1, 2, 11], pars_period, [19, 26, 33]);
    coeffs_attn = makeArray(pars_period, [40, 47, 54, 61, 68, 75]);
   elseif LeftNumber(region, 1) == 4
              dmb = interp_dmb(period_used);
    coeffs = updateArray(coeffs, [1, 2, 11], pars_period, [20, 27, 34]);
    coeffs_attn = makeArray(pars_period, [41, 48, 55, 62, 69, 76]);
    if lower(Z2pt5) == 'default' | Z2pt5<0
              delta_ln_z = 0;
    else
              delta_ln_z = log(Z2pt5) - calc_z_from_Vs30(Vs30, pars_z_ja);
    end
```

```
   coeffs_z = makeArray(pars_period_zmod, [5, 6]);
  elseif LeftNumber(region, 1) == 5
   coeffs = updateArray(coeffs, [1, 2, 11], pars_period, [21, 28, 35]);
   coeffs_attn = makeArray(pars_period, [42, 49, 56, 63, 70, 77]);
            if lower(Z1pt0) == 'default' | Z1pt0<0
                     delta_ln_z = 0;
            else
            delta_ln_z = log(Z1pt0) - calc_z_from_Vs30(Vs30, pars_z_nz);
            end
   coeffs_z = makeArray(pars_period_zmod, [7, 8]);
  elseif LeftNumber(region, 1) == 6
            dmb = interp_dmb(period_used);
   coeffs = updateArray(coeffs, [1, 2, 11], pars_period, [22, 29, 36]);
   coeffs_attn = makeArray(pars_period, [43, 50, 57, 64, 71, 78]);
  elseif LeftNumber(region, 1) == 7
   coeffs = updateArray(coeffs, [1, 2, 11], pars_period, [23, 30, 37]);
   coeffs_attn = makeArray(pars_period, [44, 51, 58, 65, 72, 79]);
            if lower(Z1pt0) == 'default' | Z1pt0<0
                     delta_ln_z = 0;
            else
            delta_ln_z = log(Z1pt0) - calc_z_from_Vs30(Vs30, pars_z_tw);
            end
   coeffs_z = makeArray(pars_period_zmod, [9, 10]);
  else  % reg=0
   coeffs_attn = makeArray(pars_period, [11, 11, 11, 11, 10, 11]);
  end

  delta_bz = makeArray(pars_period, [80, 81]);
  coeffs_nft = makeArray(pars_period, [82, 83]);

  thisFs = (LeftNumber(EventType, 1));
  mbreak = (1 - thisFs) * (Mb + dmb) + thisFs * Mb;
  zbreak = (1 - thisFs) * (30 + delta_bz(1)) + thisFs * (80 + delta_bz(2));
  Med = KBCG20_med(M, Rrup, AlphaBackarc, AlphaNankai, Ztor, EventType, Vs30, delta_ln_z, coeffs, coeffs_attn, coeffs_z,
mbreak, zbreak, k1k2(1), k1k2(2), coeffs_nft(1), coeffs_nft(2), pgarock, region);
  med_pga = KBCG20_med(M, Rrup, AlphaBackarc, AlphaNankai, Ztor, EventType, Vs30, delta_ln_z, coeffs_pga,
coeffs_attn_pga, coeffs_z_pga2, mbreak_pga, zbreak_pga, k1k2_pga(1), k1k2_pga(2), coeffs_nft_pga(1), coeffs_nft_pga(2),
pgarock, region);

  if Med < med_pga && period <= 0.1
    Med = med_pga;
  end

  KBCG20_medPSA_AtTlistValue = exp(Med);

end

function getRangeRowTValue = getRangeRowT(thisRange, Xvalue)
   % CODE DEVELOPED/IMPLEMENTED BY
   %         Silvia Mazzoni, 2020
   %          smazzoni@ucla.edu
  getRangeRowTValue = table2array(thisRange(thisRange.T==Xvalue,:));

end


function getSubArrayRangeValue = getSubArrayRange(inArray, startIndex, endIndex)
   % CODE DEVELOPED/IMPLEMENTED BY
```

```
   %         Silvia Mazzoni, 2020
   %          smazzoni@ucla.edu


   irow = 1;
   for thisIndex = startIndex : endIndex
      temP(irow) = inArray(thisIndex);
      irow = irow + 1;
   end
   getSubArrayRangeValue = temP;
end

function interpolateArrayValue = interpolateArray(Xpoint, Xlist, Ylist, XinterpType, YinterpType, extrapolateType,
XinterpMin , XinterpMax)
   % CODE DEVELOPED/IMPLEMENTED BY
   %         Silvia Mazzoni, 2020
   %          smazzoni@ucla.edu

   interpolateArrayValue = -999;
   if nargin<7
      XinterpMin = -1e16;
   end
   if nargin<8
      XinterpMax = +1e16;
   end

   irow = 0;
   start0 = -1;
   endrow0 = -1;
   [nrows,ncols] = size(Xlist);
      if nrows == 1 && ncols>0
         Xlist = Xlist.';
      end
      [nrows,ncols] = size(Ylist);
      if nrows == 1 && ncols>0
         Ylist = Ylist.';
      end
   [nrows,ncols] = size(Xlist);

   for irow = 1 : nrows
      this = Xlist(irow);
      if Xpoint == this
         interpolateArrayValue = Ylist(irow);
         return
      end
      if this >= XinterpMin && this <= XinterpMax
         if start0 < 0
            start0 = irow;
         end
         end0 = irow;
      end
   end
  iend = irow ;
   istart = start0;
   Xstart = Xlist(start0);
   Xend = Xlist(end0);
   if (Xend - Xpoint) * (Xstart - Xpoint) <= 0  % point is within limits
      for i = istart : iend - 1
```

```
      thisx = Xlist(i);
      nextX = Xlist(i + 1);
      if (Xpoint - thisx) * (Xpoint - nextX) <= 0   % point is between these two
         l1 = i;
         l2 = i + 1;
         break
      end
   end
else
   if (Xend - Xstart) * (Xend - Xpoint) > 0
      l1 = istart;
      if LeftString(lower(extrapolateType), strlength("extra")) == lower("extra")
         l2 = istart + 1;
      else
         l2 = istart;
      end
   else
      l1 = iend;
      if LeftString(lower(extrapolateType), strlength("extra")) == lower("extra")
         l2 = iend - 1;
      else
         interpolateArrayValue = Ylist(end0);
         return
      end
   end
end

x0 = Xlist(l1);
x1 = Xlist(l2);
y0 = Ylist(l1);
y1 = Ylist(l2);
hereX = Xpoint;
if lower(LeftString(XinterpType, strlength("log"))) == lower("log")
   if x0 <= 0
      x0 = 0.000000001;
   end
   if x1 <= 0
      x1 = 0.000000002;
   end
   if hereX <= 0
      hereX = 0.000000001;
   end
   x0 = log(x0);
   x1 = log(x1);
   hereX = log(hereX);
end
if lower(LeftString(YinterpType, strlength("log"))) == lower("log")
   if y0 <= 0
      y0 = 0.000000001;
   end
   if y1 <= 0
      y1 = 0.000000001;
   end
   y0 = log(y0);
   y1 = log(y1);
end
interpolateArrayValue = y0 + (y1 - y0) * (hereX - x0) / (x1 - x0);
```

```
   if lower(LeftString(YinterpType, strlength("log"))) == lower("log")
      interpolateArrayValue = exp(interpolateArrayValue);
   end
end

function LeftStringValue = LeftString(instring,ncars)
   inn =  char(instring);
   LeftStringValue = inn(1:ncars);
end
function LeftNumberValue = LeftNumber(instring,ncars)
   thisString = num2str(instring);
   inn =  char(thisString);
   LeftNumberValue=str2num(inn(1:ncars));
end


function makeArrayValue = makeArray(oldArray, oldArrayIndices)
   % CODE DEVELOPED/IMPLEMENTED BY
   %         Silvia Mazzoni, 2020
   %          smazzoni@ucla.edu


   newIndex = 1;
   [nrow,ncol] = size(oldArrayIndices);
   [nrowOldArray,ncolOldArray] = size(oldArrayIndices);

   nrowMax = nrow;
   if ncol>nrow
      nrowMax = ncol;
   end
  for irow = 1 : nrowMax
      if nrow==1 && ncol> 1
         oldIndex = oldArrayIndices(1,irow);
      else
         oldIndex = oldArrayIndices(irow,1);
      end

      if nrowOldArray==1 && ncolOldArray > 1
         newArray(newIndex) = oldArray(1,oldIndex);
      else
         newArray(newIndex) = oldArray(oldIndex,1);
      end
      newIndex = newIndex + 1;
  end
   makeArrayValue = newArray;
end

function updateArrayValue = updateArray(newArray, newArrayIndices, oldArray, oldArrayIndices)
   % CODE DEVELOPED/IMPLEMENTED BY
   %         Silvia Mazzoni, 2020
   %          smazzoni@ucla.edu


     [Nrow,Ncol]=size(oldArrayIndices);
     if Nrow==1 && Ncol>1
        oldArrayIndices = oldArrayIndices.';
     end
```

```
    [NN,Ncol]=size(oldArrayIndices);
  for irow = 1 : NN
    oldIndex = oldArrayIndices(irow);
    newIndex = newArrayIndices(irow);

    newArray(newIndex) = oldArray(oldIndex);
  end
  updateArrayValue = newArray;
end


function interpolatefunctionValue = interpolateFunction(Xpoint, Xlist, XinterpType, extrapolateType, XinterpMin
,XinterpMax )
  % CODE DEVELOPED/IMPLEMENTED BY
  %        Silvia Mazzoni, 2020
  %         smazzoni@ucla.edu
%   % % debug.print "interpolateFunction"
   if nargin<7
     XinterpMin = -1e16;
   end
   if nargin<8
     XinterpMax = +1e16;
  end
  interpolatefunctionValue = -999;
  irow = 0;
  start0 = -1;
  endrow0 = -1;
  [NN,CC] = size(Xlist);
  for irow = 1 : NN
    this = Xlist(irow, 1);
    if Xpoint == this
      interpolatefunctionValue = [irow, -1, 0];
      return
    end
    if this >= XinterpMin && this <= XinterpMax
      if start0 < 0
        start0 = irow;
      end
      end0 = irow;
    end
  end
  iend = irow - 1;

  istart = start0;
  Xstart = Xlist(start0, 1);
  Xend = Xlist(end0, 1);

  if (Xend - Xpoint) * (Xstart - Xpoint) <= 0  % point is within limits
    for i = istart : iend - 1
      thisx = Xlist(i, 1);
      nextX = Xlist(i + 1, 1);
      if (Xpoint - thisx) * (Xpoint - nextX) <= 0   % point is between these two
        l1 = i;
        l2 = i + 1;
        break
      end
    end
  else
```

```
      if (Xend - Xstart) * (Xend - Xpoint) > 0
         l1 = istart;
         if LeftString(lower(extrapolateType), strlength("extra")) == lower("extra")
            l2 = istart + 1;
         else
            l2 = istart;
         end
      else
         l1 = iend;
         if LeftString(lower(extrapolateType), strlength("extra")) == lower("extra")
            l2 = iend - 1;
         else
            interpolatefunctionValue = [end0, -1, 0];
            return
         end
      end
   end



%Interp:
   x0 = Xlist(l1, 1);
   x1 = Xlist(l2, 1);

   hereX = Xpoint;
   if lower(LeftString(XinterpType, strlength("log"))) == lower("log")
      if x0 <= 0
         x0 = 0.000000001;
      end
      if x1 <= 0
         x1 = 0.000000002;
      end
      if hereX <= 0
         hereX = 0.000000001;
      end
      x0 = log(x0);
      x1 = log(x1);
      hereX = log(hereX);
   end

   interpolatefunctionValue = [l1, l2, (hereX - x0) / (x1 - x0)];


end

function KBCG20_posteriorAtTlistValue =  KBCG20_posteriorAtTlist(period, M, Rrup, AlphaBackarc, AlphaNankai, Ztor,
EventType, Vs30, Z1pt0, Z2pt5, Mb, region, num_samples, PNWbasinStrux )
   % CODE DEVELOPED/IMPLEMENTED BY
   %         Silvia Mazzoni, 2020
   %          smazzoni@ucla.edu
 % need to add some checks for input (period, region)

   KBCG20_posteriorAtTlistValue = -999;
   if nargin<13
      num_samples = 100;
   end
   if nargin<14
```

```
    PNWbasinStrux = 0;
  end

 pars_z_ja = [7.6893685375, 2.30258509299405, 6.3091864, 0.7528670225, 1.2952369625];
 pars_z_casc = [8.29404964010203, 2.30258509299405, 6.39692965521615, 0.27081459, 1.7381352625];
 pars_z_nz = [6.859789675, 2.30258509299405, 5.745692775, 0.91563524375, 1.03531412375];
 pars_z_tw = [6.30560665, 2.30258509299405, 6.1104992125, 0.43671102, 0.7229702975];


 Parameters = readtable('NGAsubGMM_KBCG20_coefficients.csv','PreserveVariableNames',true);
parameters_zmod =
readtable("NGAsubGMM_KBCG20_params_Z_ALL_allregca_attn3_corrreg_cs_dmb.csv",'PreserveVariableNames',true);


 % calculate rock PGA
 thisPeriod = 0.;
 vsrock = 1100;
 pars_period = getRangeRowT(Parameters, thisPeriod);
 pars_period_zmod = getRangeRowT(parameters_zmod, thisPeriod) ; % silviamazzoni, I added this
 coeffs = getSubArrayRange(pars_period, 2, 15);
 k1k2 = interp_k1k2(thisPeriod);
 dmb = 0;


  delta_ln_z = 0;
  coeffs_z = [0, 0];
 if LeftNumber(region, 1) == 1
  coeffs = updateArray(coeffs, [1, 2, 11], pars_period, [17, 24, 31]);
  coeffs_attn = makeArray(pars_period, [38, 45, 52, 59, 66, 73]);
 elseif LeftNumber(region, 1) == 2
  coeffs = updateArray(coeffs, [1, 2, 11], pars_period, [18, 25, 32]);
  coeffs_attn = makeArray(pars_period, [39, 46, 53, 60, 67, 74]);
 elseif LeftNumber(region, 1) == 3
  coeffs = updateArray(coeffs, [1, 2, 11], pars_period, [19, 26, 33]);
  coeffs_attn = makeArray(pars_period, [40, 47, 54, 61, 68, 75]);
 elseif LeftNumber(region, 1) == 4
          dmb = interp_dmb(thisPeriod);
  coeffs = updateArray(coeffs, [1, 2, 11], pars_period, [20, 27, 34]);
  coeffs_attn = makeArray(pars_period, [41, 48, 55, 62, 69, 76]);
 elseif LeftNumber(region, 1) == 5
  coeffs = updateArray(coeffs, [1, 2, 11], pars_period, [21, 28, 35]);
  coeffs_attn = makeArray(pars_period, [42, 49, 56, 63, 70, 77]);
 elseif LeftNumber(region, 1) == 6
          dmb = interp_dmb(thisPeriod);
  coeffs = updateArray(coeffs, [1, 2, 11], pars_period, [22, 29, 36]);
  coeffs_attn = makeArray(pars_period, [43, 50, 57, 64, 71, 78]);
 elseif LeftNumber(region, 1) == 7
  coeffs = updateArray(coeffs, [1, 2, 11], pars_period, [23, 30, 37]);
  coeffs_attn = makeArray(pars_period, [44, 51, 58, 65, 72, 79]);
 else  % reg=0
  coeffs_attn = makeArray(pars_period, [11, 11, 11, 11, 10, 11]);
 end


 delta_bz = makeArray(pars_period, [80, 81]);
 coeffs_nft = makeArray(pars_period, [82, 83]);
 thisFs = (LeftNumber(EventType, 1));
 mbreak = (1 - thisFs) * (Mb + dmb) + thisFs * Mb;
```

```
  zbreak = (1 - thisFs) * (30 + delta_bz(1)) + thisFs * (80 + delta_bz(2));
   pgarock = exp(KBCG20_med(M, Rrup, AlphaBackarc, AlphaNankai, Ztor, EventType, vsrock, delta_ln_z, coeffs, coeffs_attn,
coeffs_z, mbreak, zbreak, k1k2(1), k1k2(2), coeffs_nft(1), coeffs_nft(2), 0, region));


  % calculate PSA
  thisPeriod = period;
     Tnumeric=[-1,0,0.01,0.02,0.03,0.05,0.075,0.1,0.15,0.2,0.25,0.3,0.4,0.5,0.75,1,1.5,2,3,4,5,7.5,10];
   Tstring=["T-
1.00","T00.00","T00.01","T00.02","T00.03","T00.05","T00.07","T00.10","T00.15","T00.20","T00.25","T00.30","T00.40","T00.
50","T00.75","T01.00","T01.50","T02.00","T03.00","T04.00","T05.00","T07.50","T10.00"];
   KBCG20_PeriodMapData = table(Tnumeric.',Tstring.');


   thisTstring = KBCG20_PeriodMapData(KBCG20_PeriodMapData.Var1==thisPeriod,:).Var2;
   parameters_posterior = readtable('posterior_coefficients_KBCG20_' + thisTstring + '.csv','PreserveVariableNames',true);

  pars_period_zmod = getRangeRowT(parameters_zmod, thisPeriod) ; % silviamazzoni, I added this
  Parameters = readtable('NGAsubGMM_KBCG20_coefficients.csv','PreserveVariableNames',true);


  coeffs = getSubArrayRange(pars_period, 2, 15);
  k1k2 = interp_k1k2(thisPeriod);
  dmb = 0;


   for k = 1 : num_samples
     pars_period = table2array(parameters_posterior(k,:));

     coeffs = getSubArrayRange(pars_period, 2, 15);

      delta_ln_z = 0;
      coeffs_z = [0, 0];
     if LeftNumber(region, 1) == 1
      coeffs = updateArray(coeffs, [1, 2, 11], pars_period, [17, 24, 31]);
      coeffs_attn = makeArray(pars_period, [38, 45, 52, 59, 66, 73]);
     elseif LeftNumber(region, 1) == 2
      coeffs = updateArray(coeffs, [1, 2, 11], pars_period, [18, 25, 32]);
      coeffs_attn = makeArray(pars_period, [39, 46, 53, 60, 67, 74]);
                        if lower(Z2pt5) == 'default' | Z2pt5<0
                                delta_ln_z = 0;
                        else
            delta_ln_z = log(Z2pt5) - calc_z_from_Vs30(Vs30, pars_z_casc);
                        end

      if LeftNumber(PNWbasinStrux, 1) == 1
        coeff_seattle = getRangeValueT(parameters_zmod, thisPeriod, "mean_residual_Seattle_basin");
        coeffs_z = [coeff_seattle, 0];
      elseif LeftNumber(PNWbasinStrux, 1) == 2
        coeffs_z = makeArray(pars_period_zmod, [2, 3]);
      end
     elseif LeftNumber(region, 1) == 3
      coeffs = updateArray(coeffs, [1, 2, 11], pars_period, [19, 26, 33]);
      coeffs_attn = makeArray(pars_period, [40, 47, 54, 61, 68, 75]);
     elseif LeftNumber(region, 1) == 4
                        dmb = interp_dmb(thisPeriod);
      coeffs = updateArray(coeffs, [1, 2, 11], pars_period, [20, 27, 34]);
      coeffs_attn = makeArray(pars_period, [41, 48, 55, 62, 69, 76]);
```

```
                        if lower(Z2pt5) == 'default' | Z2pt5<0
                                delta_ln_z = 0;
                        else
            delta_ln_z = log(Z2pt5) - calc_z_from_Vs30(Vs30, pars_z_ja);
                        end
       coeffs_z = makeArray(pars_period_zmod, [5, 6]);;
       elseif LeftNumber(region, 1) == 5
       coeffs = updateArray(coeffs, [1, 2, 11], pars_period, [21, 28, 35]);
       coeffs_attn = makeArray(pars_period, [42, 49, 56, 63, 70, 77]);
                        if lower(Z1pt0) == 'default' | Z1pt0<0
                                delta_ln_z = 0;
                        else
            delta_ln_z = log(Z1pt0) - calc_z_from_Vs30(Vs30, pars_z_nz);
                        end
       coeffs_z = makeArray(pars_period_zmod, [7, 8]);
       elseif LeftNumber(region, 1) == 6
                        dmb = interp_dmb(thisPeriod);
       coeffs = updateArray(coeffs, [1, 2, 11], pars_period, [22, 29, 36]);
       coeffs_attn = makeArray(pars_period, [43, 50, 57, 64, 71, 78]);
       elseif LeftNumber(region, 1) == 7
       coeffs = updateArray(coeffs, [1, 2, 11], pars_period, [23, 30, 37]);
       coeffs_attn = makeArray(pars_period, [44, 51, 58, 65, 72, 79]);
                        if lower(Z1pt0) == 'default' | Z1pt0<0
                                delta_ln_z = 0;
                        else
            delta_ln_z = log(Z1pt0) - calc_z_from_Vs30(Vs30, pars_z_tw);
                        end
       coeffs_z = makeArray(pars_period_zmod, [9, 10]);
       else  % reg=0
       coeffs = updateArray(coeffs, [1, 2, 11], pars_period, [172, 173, 174]);
       coeffs_attn = makeArray(pars_period, [175, 176, 177, 178, 179, 180]);
      end


      delta_bz = makeArray(pars_period, [80, 81]);
      coeffs_nft = makeArray(pars_period, [82, 83]);
      thisFs = (LeftNumber(EventType, 1));

      mbreak = (1 - thisFs) * (Mb + dmb) + thisFs * Mb;
      zbreak = (1 - thisFs) * (30 + delta_bz(1)) + thisFs * (80 + delta_bz(2));

      Med = KBCG20_med(M, Rrup, AlphaBackarc, AlphaNankai, Ztor, EventType, Vs30, delta_ln_z, coeffs, coeffs_attn,
coeffs_z, mbreak, zbreak, k1k2(1), k1k2(2), coeffs_nft(1), coeffs_nft(2), pgarock, region);
      med_predictions(k) = Med;
   end


   KBCG20_posteriorAtTlist_All(1) = getMeanOfArray(med_predictions);
   KBCG20_posteriorAtTlist_All(2) = getMedianOfArray(med_predictions);
   KBCG20_posteriorAtTlist_All(3) = getStdDev(med_predictions);
   for i = 1: length(med_predictions)
      KBCG20_posteriorAtTlist_All(3 + i  ) = med_predictions(i);
   end
   KBCG20_posteriorAtTlistValue = KBCG20_posteriorAtTlist_All;
end


function getMeanOfArrayValue = getMeanOfArray(inArray)
```

```
   % CODE DEVELOPED/IMPLEMENTED BY
   %         Silvia Mazzoni, 2020
   %          smazzoni@ucla.edu

   getMeanOfArrayValue = 0;
   iCount = 0;

   for ii = 1 : length(inArray)
      getMeanOfArrayValue = getMeanOfArrayValue + inArray(ii);
      iCount = iCount + 1;
   end
   getMeanOfArrayValue = getMeanOfArrayValue / iCount;

end

function getMedianOfArrayValue = getMedianOfArray(inArray)
   % CODE DEVELOPED/IMPLEMENTED BY
   %         Silvia Mazzoni, 2020
   %          smazzoni@ucla.edu

   inArray = BubbleSrt(inArray, "True");
   num_samples = length(inArray);
   if round(num_samples / 2, 0) == num_samples / 2
      getMedianOfArrayValue = 0.5 * (inArray(num_samples / 2) + inArray(num_samples / 2 + 1));
   else
      getMedianOfArrayValue = inArray((num_samples + 1) / 2);
   end

end

%https://www.mrexcel.com/board/threads/standard-deviation-of-an-array.206375/
%*************************************************************************
%*             Standard Deviation  of a 1D array                  *
%*************************************************************************
function getStdDevValue = getStdDev(Arr)
   % CODE DEVELOPED/IMPLEMENTED BY
   %         Silvia Mazzoni, 2020
   %          smazzoni@ucla.edu

    k1 = 1;
    k2 = length(Arr);

    n = 0;
    avg = getMeanOfArray(Arr);
    SumSq = 0;
    for i = k1 : k2
      n = n + 1;
       SumSq = SumSq + (Arr(i) - avg) ^ 2;
    end

    getStdDevValue = sqrt(SumSq / (n - 1));

end


%https://stackoverflow.com/questions/11504418/excel-vba-quickest-way-to-sort-an-array-of-numbers-in-descending-
order%
% Omit plngLeft & plngRight; they are used internally during recursion
```

```
function BubbleSrtValue = BubbleSrt(ArrayIn, Ascending)
   % CODE DEVELOPED/IMPLEMENTED BY
   %         Silvia Mazzoni, 2020
   %          smazzoni@ucla.edu



   if Ascending == "True"
      for i = 1 : length(ArrayIn)
         for j = i + 1 : length(ArrayIn)
            if ArrayIn(i) > ArrayIn(j)
               SrtTemp = ArrayIn(j);
               ArrayIn(j) = ArrayIn(i);
               ArrayIn(i) = SrtTemp;
            end
         end
       end
   else
      for i = 1 : length(ArrayIn)
         for j = i + 1 : length(ArrayIn)
            if ArrayIn(i) < ArrayIn(j)
               SrtTemp = ArrayIn(j);
               ArrayIn(j) = ArrayIn(i);
               ArrayIn(i) = SrtTemp;
            end
         end
       end
   end

   BubbleSrtValue = ArrayIn;

end

function KBCG20_posteriorValue = KBCG20_posterior(period, M, Rrup, AlphaBackarc, AlphaNankai, Ztor, EventType, Vs30,
Z1pt0, Z2pt5, Mb, region, num_samples  , PNWbasinStrux)
   % CODE DEVELOPED/IMPLEMENTED BY
   %         Silvia Mazzoni, 2020
   %          smazzoni@ucla.edu

   if nargin<13
      num_samples = 100;
   end
   if nargin<14
      PNWbasinStrux = 0;
   end
   thisOut = -999;
   XinterpMin = 0;
   XinterpMax = 10;
   XinterpType = "log";
   YinterpType = "linear"; % KBCG20_posteriorAtTlist values are logs already
   extrapolateType = "extrapolate";

   Parameters = readtable('NGAsubGMM_KBCG20_coefficients.csv','PreserveVariableNames',true);
   TvalueList = Parameters.T;
   InterpArray = interpolateFunction(period, TvalueList, XinterpType, extrapolateType, XinterpMin, XinterpMax);

   period0 = TvalueList(InterpArray(1), 1);
```

```
   KBCG20_posteriorAtTlist0 = KBCG20_posteriorAtTlist(period0, M, Rrup, AlphaBackarc, AlphaNankai, Ztor, EventType,
Vs30, Z1pt0, Z2pt5, Mb, region, num_samples, PNWbasinStrux);
   y0 = InterpArray(1);
   if InterpArray(2) <= 0
      KBCG20_posteriorValue = KBCG20_posteriorAtTlist0;
      return
   else
      period1 = TvalueList(InterpArray(2), 1);
      KBCG20_posteriorAtTlist1 = KBCG20_posteriorAtTlist(period1, M, Rrup, AlphaBackarc, AlphaNankai, Ztor, EventType,
Vs30, Z1pt0, Z2pt5, Mb, region, num_samples, PNWbasinStrux);
      for iCase = 1: length(KBCG20_posteriorAtTlist0)
         y0 = KBCG20_posteriorAtTlist0(iCase);
         y1 = KBCG20_posteriorAtTlist1(iCase);

         if lower(LeftString(YinterpType, strlength("log"))) == lower("log")
            if y0 <= 0
               y0 = 0.000000001;
            end
            if y1 <= 0
               y1 = 0.000000001;
            end
            y0 = log(y0);
            y1 = log(y1);
         end
         y0;
         y1;
         InterpArray(3);
         y0 + (y1 - y0) * InterpArray(3);
         med_predictions(iCase) = y0 + (y1 - y0) * InterpArray(3);
         if lower(LeftString(YinterpType, strlength("log"))) == lower("log")
            med_predictions(iCase) = exp(med_predictions(iCase));
         end
      end

   end

   KBCG20_posteriorValue = med_predictions;

end




% function to calculate aliatory sigma



function KBCG20_sigmaAleatoryValue = KBCG20_sigmaAleatory(period)
   % CODE DEVELOPED/IMPLEMENTED BY
   %        Silvia Mazzoni, 2020
   %         smazzoni@ucla.edu

   thisOut = -999;
   XinterpMin = 0;
   XinterpMax = 10;
   XinterpType = "log";
   YinterpType = "linear";
```

```
    extrapolateType = "extrapolate";

     Parameters = readtable('NGAsubGMM_KBCG20_coefficients.csv','PreserveVariableNames',true);
    TvalueList = Parameters.T;
    InterpArray = interpolateFunction(period, TvalueList, XinterpType, extrapolateType, XinterpMin, XinterpMax);

    period0 = TvalueList(InterpArray(1), 1);
    y0 = KBCG20_sigmaAleatoryAtTlist(period0);

    if InterpArray(2) <= 0
       thisOut = y0;
    else
       period1 = TvalueList(InterpArray(2), 1);
       y1 = KBCG20_sigmaAleatoryAtTlist(period1);
       if lower(LeftString(YinterpType, strlength("log"))) == lower("log")
          if y0 <= 0
             y0 = 0.000000001;
          end
          if y1 <= 0
             y1 = 0.000000001;
          end
          y0 = log(y0);
          y1 = log(y1);
       end
       thisOut = y0 + (y1 - y0) * InterpArray(3);
       if lower(LeftString(YinterpType, strlength("log"))) == lower("log")
          thisOut = exp(thisOut);
       end

    end

    KBCG20_sigmaAleatoryValue = thisOut;

end


function KBCG20_sigmaAleatoryAtTlistValue = KBCG20_sigmaAleatoryAtTlist(period)
     % CODE DEVELOPED/IMPLEMENTED BY
    %         Silvia Mazzoni, 2020
    %          smazzoni@ucla.edu

  Parameters = readtable('NGAsubGMM_KBCG20_coefficients.csv','PreserveVariableNames',true);
  % calculate PSA
  period_used = period;
  pars_period = getRangeRowT(Parameters, period_used);

  Phi = pars_period(84);
  Tau = pars_period(85);
  KBCG20_sigmaAleatoryAtTlistValue = sqrt(Phi * Phi + Tau * Tau);

end

function KBCG20_sigmaPhiValue = KBCG20_sigmaPhi(period)
     % CODE DEVELOPED/IMPLEMENTED BY
    %         Silvia Mazzoni, 2020
    %          smazzoni@ucla.edu

    Parameters = readtable('NGAsubGMM_KBCG20_coefficients.csv','PreserveVariableNames',true);
```

```
 % calculate PSA
 period_used = period;
 pars_period = getRangeRowT(Parameters, period_used);

 Phi = pars_period(84);
 KBCG20_sigmaPhiValue = Phi;

end

function KBCG20_sigmaTauValue = KBCG20_sigmaTau(period)
    % CODE DEVELOPED/IMPLEMENTED BY
  %         Silvia Mazzoni, 2020
  %          smazzoni@ucla.edu

  Parameters = readtable('NGAsubGMM_KBCG20_coefficients.csv','PreserveVariableNames',true);

 % calculate PSA
 period_used = getTFvalue(period);
 pars_period = getRangeRowT(Parameters, period_used);
 Tau = pars_period(85);
 KBCG20_sigmaTauValue = Tau;

end



function KBCG20_SigmaEpistemicValue = KBCG20_SigmaEpistemic(period, M, Rrup, AlphaBackarc, AlphaNankai, Ztor,
EventType, Vs30, Z1pt0, Z2pt5, Mb, region,  num_samples , PNWbasinStrux )
  % CODE DEVELOPED/IMPLEMENTED BY
  %         Silvia Mazzoni, 2020
  %          smazzoni@ucla.edu

  if nargin<13
     num_samples = 100;
  end
  if nargin<14
     PNWbasinStrux = 0;
  end
  %  % % debug.print "KBCG20_SigmaEpistemic"
  if num_samples == 0
     KBCG20_SigmaEpistemicValue = 0;
  elseif num_samples < 50
     KBCG20_SigmaEpistemicValue = "not enough samples, please select a value =0 || between 50 and 800";
  elseif num_samples <= 800

     thisKBCG20_posterior = KBCG20_posterior(period, M, Rrup, AlphaBackarc, AlphaNankai, Ztor, EventType, Vs30, Z1pt0,
Z2pt5, Mb, region, num_samples, PNWbasinStrux);
     KBCG20_SigmaEpistemicValue = thisKBCG20_posterior(3);
  else
     KBCG20_SigmaEpistemicValue = "Please enter a value =0 || between 50 and 800";
  end


end

function KBCG20_SigmaTotalValue = KBCG20_SigmaTotal(period, M, Rrup, AlphaBackarc, AlphaNankai, Ztor, EventType,
Vs30, Z1pt0, Z2pt5, Mb, region,  num_samples  ,  PNWbasinStrux )
```

```
   % CODE DEVELOPED/IMPLEMENTED BY
   %         Silvia Mazzoni, 2020
   %          smazzoni@ucla.edu
   if nargin<13
      num_samples = 100;
   end
   if nargin<14
      PNWbasinStrux = 0;
   end
   SigmaAleatory = KBCG20_sigmaAleatory(period);
   if num_samples == 0
      SigmaEpistemic = 0.;
   else
      SigmaEpistemic = KBCG20_SigmaEpistemic(period, M, Rrup, AlphaBackarc, AlphaNankai, Ztor, EventType, Vs30, Z1pt0,
Z2pt5, Mb, region, num_samples, PNWbasinStrux);
   end
   KBCG20_SigmaTotalValue = sqrt(SigmaAleatory * SigmaAleatory + SigmaEpistemic * SigmaEpistemic);

end

function PSHAB20_SigmaTotalValue = PSHAB20_SigmaTotal(period, Rrup,  Vs30, EventType,   SubductionSlab )
   % CODE DEVELOPED/IMPLEMENTED BY
   %         Silvia Mazzoni, 2020
   %          smazzoni@ucla.edu
   SigmaAleatory = PSHAB20_SigmaAleatory(period, Rrup, Vs30);
   SigmaEpistemic = PSHAB20_SigmaEpistemic(period, EventType, SubductionSlab);
   PSHAB20_SigmaTotalValue = sqrt(SigmaAleatory * SigmaAleatory + SigmaEpistemic * SigmaEpistemic);

end




%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% updated 6/1/2020 by Silvia Mazzoni

% GMM_at_760_slab_v4.R

%#Grace Parker
%#Modified February 26 to expand comments
%#Modified March 25, 2020 to call coefficients from master table

%# Input Parameters --------------------------------------------------------

%#Event type: 0 == interface, 1 == slab

%#region corresponds to options in the DatabaseRegion column of the flatfile, plus global. Must be a string. If no matches,
default will be global model:
%# "global", "Alaska", "Cascadia", "CAM", "Japan", "SA" or "Taiwan"

%#Saturation Region corresponds to regions defined by R. Archuleta and C. Ji:
%# "global", "Aleutian","Alaska","Cascadia","Central_America_S", "Central_America_N",
"Japan_Pac","Japan_Phi","South_America_N","South_America_S", "Taiwan_W","Taiwan_E"

%# Rrup is number in kilometers
```

```
%#Hypocentral depth in km. To use Ztor value to estimate hypocentral depth, see Ch. 4.3.3 of Parker et al. PEER report

%# period can be: (-1,0,0.01,0.02,0.025,0.03,0.04,0.05,0.075,0.1,0.15,0.2,0.25,0.3,0.4,0.5,0.75,1,1.5,2,2.5,3,4,5,7.5,10)
%# where -1 == PGV and 0 == PGA

%# Other pertinent information ----------------------------------------------
%#Coefficient files must be in the active working directory
%# This function has no site term. Can only estimate ground motion at the reference condition VS30 = 760m/s
%# The output is the desired median model prediction in LN units
%# Take the exponential to get PGA, PSA in g or the PGV in cm/s

%#Function to compute GMM predictions at 760m/s for slab and interface


function PSHAB20_GMM_at_760_SlabValue = PSHAB20_GMM_at_760_Slab(EventType, UserRegion,
saturation_region_user, Rrup, M, hypocentral_depth, period)
    % CODE DEVELOPED/IMPLEMENTED BY
   %        Silvia Mazzoni, 2020
   %         smazzoni@ucla.edu

%  % % debug.print "PSHAB20_GMM_at_760_Slab"
 PSHAB20_GMM_at_760_SlabValue = -999;
 if (LeftNumber(EventType, 1)) == 0
  return % ("This function is only for slab")
 end
 region = getPSHABregion(UserRegion);
 saturation_region = getPSHABSubductionRegion(saturation_region_user);

 %%import Coefficients

%%  Import Master Coefficient Table ----------------------------------------
   CoefficientsTable = readtable('PSHAB20_Table_E2_Slab_Coefficients_OneRowHeader.csv','PreserveVariableNames',true);

  %%Define mb based on Archuleta and Ji (2019)
   Mb = getMbDefault("PSHAB20",EventType, UserRegion, saturation_region);

%%  Constant ---------------------------------------------------------------
%  %Isolate constant
 if region == "global"
  c0 = getRangeValueT(CoefficientsTable, period, "Global_c0");
 elseif region == "Alaska" | region == "SA"
  c0 = getRangeValueT(CoefficientsTable, period, saturation_region + "_c0");
 else
  c0 = getRangeValueT(CoefficientsTable, period, region + "_c0");
 end
  % silviamazzoni, I added this:
  if checkEmptyNA999(c0)
     c0 = getRangeValueT(CoefficientsTable, period, "Global_c0");
  end
%  % Path Term ----------------------------------------------------------------

%  %near-source saturation
 if M <= Mb
  littleM = (Log10(35) - Log10(3.12)) / (Mb - 4);
  h = 10 ^ (littleM * (M - Mb) + Log10(35));
 else
  h = 35;
 end
```

```
  Rref = sqrt(1 + h ^ 2);
  r = sqrt(Rrup ^ 2 + h ^ 2);
  LogR = log(r);
  R_Rref = log(r / Rref);


%  %Need  to isolate regional anelastic coefficient, a0
  if region == "global"
    a0 = getRangeValueT(CoefficientsTable, period, "Global_a0");
   else
    a0 = getRangeValueT(CoefficientsTable, period, region + "_a0");
  end
  if checkEmptyNA999(a0)
    a0 = getRangeValueT(CoefficientsTable, period, "Global_a0");
  end



  c1 = getRangeValueT(CoefficientsTable, period, "c1");
  b4 = getRangeValueT(CoefficientsTable, period, "b4");
  Fp = (c1 * LogR + (b4 * M) * R_Rref + a0 * r);


%  % Magnitude Scaling ----------------------------------------------------------
  c4 = getRangeValueT(CoefficientsTable, period, "c4");
  c5 = getRangeValueT(CoefficientsTable, period, "c5");
  c6 = getRangeValueT(CoefficientsTable, period, "c6");
  Fm = c4 * func1(M, Mb) + c6 * func2(M, Mb) + c5 * func3(M, Mb);




%  % Source Depth Scaling -------------------------------------------------------
  Db = getRangeValueT(CoefficientsTable, period, "db_km");
  d = getRangeValueT(CoefficientsTable, period, "d");
  littleM = getRangeValueT(CoefficientsTable, period, 'm');
  if hypocentral_depth >= Db
    Fd = d;
   elseif hypocentral_depth <= 20
    Fd = littleM * (20 - Db) + d;
  else
    Fd = littleM * (hypocentral_depth - Db) + d;
  end


    mu = c0 + Fp + Fm + Fd;
    PSHAB20_GMM_at_760_SlabValue = mu;

end

  function func1value = func1(M, Mb)
    % CODE DEVELOPED/IMPLEMENTED BY
    %         Silvia Mazzoni, 2020
    %          smazzoni@ucla.edu
    if M <= Mb
       func1value = M - Mb;
    else
       func1value = 0;
```

```
    end
  end
  function func2value = func2(M, Mb)
    % CODE DEVELOPED/IMPLEMENTED BY
    %         Silvia Mazzoni, 2020
    %          smazzoni@ucla.edu
    if M > Mb
       func2value = M - Mb;
    else
       func2value = 0;
    end
  end
  function func3value = func3(M, Mb)
    % CODE DEVELOPED/IMPLEMENTED BY
    %         Silvia Mazzoni, 2020
    %          smazzoni@ucla.edu
    if M <= Mb
       func3value = (M - Mb) ^ 2;
    else
       func3value = 0;
    end
  end


function getPSHABregionValue = getPSHABregion(UserRegion)
    %' CODE DEVELOPED/IMPLEMENTED BY
    %'         Silvia Mazzoni, 2020
    %'          smazzoni@ucla.edu

   PSHAB_RegionIDList = ["global","Alaska","Cascadia","CAM","Japan","global","SA","Taiwan"];
   RegionIDList =
["0_global","1_Alaska","2_Cascadia","3_CentralAmerica&Mexico","4_Japan","5_NewZealand","6_SouthAmerica","7_Taiwan
"];
   PSHAB_RegionIDListMap = table(RegionIDList.',PSHAB_RegionIDList.');


    getPSHABregionValue = PSHAB_RegionIDListMap(PSHAB_RegionIDListMap.Var1==UserRegion,:).Var2;


    end


function getPSHABSubductionRegionValue = getPSHABSubductionRegion(UserSubductionRegion)
    %' CODE DEVELOPED/IMPLEMENTED BY
    %'         Silvia Mazzoni, 2020
    %'          smazzoni@ucla.edu
    PSHAB_SubdRegionIDList =
["Alaska","Aleutian","Cascadia","CAM_N","CAM_S","global","Japan_Pac","Japan_Phi","New_Zealand_N","New_Zealand_S",
"SA_N","SA_S","Taiwan_E","Taiwan_W","Alaska","Aleutian","Cascadia","CAM_N","CAM_S","global","Japan_Pac","Japan_Phi
","New_Zealand_N","New_Zealand_S","SA_N","SA_S","Taiwan_E","Taiwan_W"];
    SubdRegionIDList =
["Alaska","Aleutian","Cascadia","CAM_N","CAM_S","global","Japan_Pac","Japan_Phi","NZ_N","NZ_S","SA_N","SA_S","Taiwa
n_E","Taiwan_W","Alaska","Aleutian","Cascadia","Central_America_N","Central_America_S","global","Japan_Pac","Japan_P
hi","New_Zealand_N","New_Zealand_S","South_America_N","South_America_S","Taiwan_E","Taiwan_W"];
    PSHAB_SubductionRegionIDListMap = table(SubdRegionIDList.',PSHAB_SubdRegionIDList.');


    out = PSHAB_SubductionRegionIDListMap(PSHAB_SubductionRegionIDListMap.Var1==UserSubductionRegion,:).Var2;
```

```
   getPSHABSubductionRegionValue = out(1);

   end


function getMbDefaultValue = getMbDefault(NGAsubModelLabel,EventType, UserRegion, saturation_region_user)
   % CODE DEVELOPED/IMPLEMENTED BY
   %        Silvia Mazzoni, 2020
   %         smazzoni@ucla.edu
   % Silvia Mazzoni: I inserted the global model and replaced the NA values with those of the global model



   if (LeftNumber(EventType, 1)) == 1 | (LeftNumber(EventType, 1)) == 5
      if NGAsubModelLabel == "KBCG20"
         %MbKBCG20Intraslab
         saturation_region_here = saturation_region_user;
         saturation_regions_SBZ = ["global", "Aleutian", "Alaska", "Cascadia", "Central_America_N", "Central_America_S",
"Japan_Pac", "Japan_Phi", "New_Zealand_N", "New_Zealand_S", "South_America_N", "South_America_S", "Taiwan_W",
"Taiwan_E"];
         thisMbList = [7.6, 8, 7.2, 7.2, 7.4, 7.6, 7.6, 7.6, 7.6, 7.6, 7.3, 7.2, 7.7, 7.7];
         User_regions_SBZ = ["0_global", "1_Alaska", "1_Alaska", "2_Cascadia", "3_CentralAmerica&Mexico",
"3_CentralAmerica&Mexico", "4_Japan", "4_Japan", "5_NewZealand", "5_NewZealand", "6_SouthAmerica",
"6_SouthAmerica", "7_Taiwan", "7_Taiwan"];
      else
         %MbPSHAB20Intraslab
         saturation_region_here = getPSHABSubductionRegion(saturation_region_user);
         Mb_global_Slab = 7.6;
         saturation_regions_SBZ = ["global", "Aleutian", "Alaska", "-999", "Cascadia", "CAM_S", "CAM_N", "Japan_Pac",
"Japan_Phi", "New_Zealand_N", "New_Zealand_S", "SA_N", "SA_S", "Taiwan_W", "Taiwan_E"];
         thisMbList = [Mb_global_Slab, 7.98, 7.2, Mb_global_Slab, 7.2, 7.6, 7.4, 7.65, 7.55, Mb_global_Slab, Mb_global_Slab,
7.3, 7.25, 7.7, 7.7];
         User_regions_SBZ = ["0_global", "1_Alaska", "1_Alaska", "0_global", "2_Cascadia", "3_CentralAmerica&Mexico",
"3_CentralAmerica&Mexico", "4_Japan", "4_Japan", "5_NewZealand", "5_NewZealand", "6_SouthAmerica",
"6_SouthAmerica", "7_Taiwan", "7_Taiwan"];
      end
   else

      if NGAsubModelLabel == "KBCG20"
         %MbKBCG20Interface
         saturation_region_here = saturation_region_user;
         saturation_regions_SBZ = ["global", "Aleutian", "Alaska", "Cascadia", "Central_America_N", "Central_America_S",
"Japan_Pac", "Japan_Phi", "New_Zealand_N", "New_Zealand_S", "South_America_N", "South_America_S", "Taiwan_W",
"Taiwan_E"];
         thisMbList = [7.9, 8, 8.6, 8, 7.4, 7.5, 8.5, 7.7, 8.3, 8, 8.5, 8.6, 7.1, 7.1];
         User_regions_SBZ = ["0_global", "1_Alaska", "1_Alaska", "2_Cascadia", "3_CentralAmerica&Mexico",
"3_CentralAmerica&Mexico", "4_Japan", "4_Japan", "5_NewZealand", "5_NewZealand", "6_SouthAmerica",
"6_SouthAmerica", "7_Taiwan", "7_Taiwan"];
      else
         %MbPSHAB20Interface
         saturation_region_here = getPSHABSubductionRegion(saturation_region_user);
         Mb_global_IF = 7.9;
         saturation_regions_SBZ = ["global", "Aleutian", "Alaska", "-999", "Cascadia", "CAM_S", "CAM_N", "Japan_Pac",
"Japan_Phi", "New_Zealand_N", "New_Zealand_S", "SA_N", "SA_S", "Taiwan_W", "Taiwan_E"];
         thisMbList = [Mb_global_IF, 8, 8.6, Mb_global_IF, 7.7, 7.4, 7.4, 8.5, 7.7, Mb_global_IF, Mb_global_IF, 8.5, 8.6, 7.1,
7.1];
```

```
        User_regions_SBZ = ["0_global", "1_Alaska", "1_Alaska", "0_global", "2_Cascadia", "3_CentralAmerica&Mexico",
"3_CentralAmerica&Mexico", "4_Japan", "4_Japan", "5_NewZealand", "5_NewZealand", "6_SouthAmerica",
"6_SouthAmerica", "7_Taiwan", "7_Taiwan"];
      end

   end

   thisSaturationRegionDbRegion = getArrayMap(saturation_region_here, saturation_regions_SBZ, User_regions_SBZ);
   if lower(thisSaturationRegionDbRegion) ~= lower(UserRegion)
      saturation_region_here = "global";
   end

   Mb = getArrayMap(saturation_region_here, saturation_regions_SBZ, thisMbList);


   if checkEmptyNA999(Mb)
     Mb = getArrayMap("global", saturation_regions_SBZ, thisMbList);
   end

   getMbDefaultValue = Mb;


end




function getArrayMapValue = getArrayMap(Xvalue, Xarray, Yarray)
   % CODE DEVELOPED/IMPLEMENTED BY
   %         Silvia Mazzoni, 2020
   %          smazzoni@ucla.edu
             MapTable = table(Xarray.',Yarray.');
   getArrayMapValue = MapTable(MapTable.Var1==Xvalue,:).Var2;

end



function getRangeValueValue = getRangeValue(thisRange, Xvalue, xHeader, yHeader)
   % CODE DEVELOPED/IMPLEMENTED BY
   %         Silvia Mazzoni, 2020
   %          smazzoni@ucla.edu
   Xvalue = convertStringsToChars(Xvalue)
    getRangeValueValue = table2array(thisRange(thisRange.xHeader==Xvalue,yHeader))
end
function getRangeValueRegionValue = getRangeValueRegion(thisRange, Xvalue,  yHeader)
   % CODE DEVELOPED/IMPLEMENTED BY
   %         Silvia Mazzoni, 2020
   %          smazzoni@ucla.edu
  % Xvalue = convertStringsToChars(Xvalue)
  getRangeValueRegionValue = -999;
   [Nrow,Ncol] = size(thisRange);

   for i=1:Nrow
      thisRegionHere = table2array(thisRange(i,'Region'));
      thisRegionHere = string(thisRegionHere);
      if lower(thisRegionHere) == lower(Xvalue)
         %a = 'found'
```

```
        getRangeValueRegionValue = table2array(thisRange(i,yHeader));
      end
   end
end
function getRangeValueTValue = getRangeValueT(thisRange, Xvalue, yHeader)
   % CODE DEVELOPED/IMPLEMENTED BY
   %         Silvia Mazzoni, 2020
   %          smazzoni@ucla.edu
   getRangeValueTValue = table2array(thisRange(thisRange.T==Xvalue,yHeader));
   if iscell(getRangeValueTValue)
      getRangeValueTValue = char(getRangeValueTValue);
   end
end

function checkEmptyNA999value = checkEmptyNA999(thisvalue)
   checkEmptyNA999value = false;
   if ischar(thisvalue)
      if thisvalue == 'NA'
        checkEmptyNA999value = true;
      end
   elseif isempty(thisvalue)
      checkEmptyNA999value = true;
   elseif thisvalue == -999
      checkEmptyNA999value = true;
   end

end



% GMM_at_VS30_slab_v4.R

%#Grace Parker
%#Modified February 26 to expand comments
%#Modified March 25, 2020 to call consolidated coefficient table

%# Input Parameters ----------------------------------------------------

%#Event type: 0 == interface, 1 == slab

%#region corresponds to options in the DatabaseRegion column of the flatfile, plus global. Must be a string. If no matches,
default will be global model:
%# "global", "Alaska", "Cascadia", "CAM", "Japan", "SA" or "Taiwan"

%#Saturation Region corresponds to regions defined by R. Archuleta and C. Ji:
%# "global", "Aleutian","Alaska","Cascadia","Central_America_S", "Central_America_N",
"Japan_Pac","Japan_Phi","South_America_N","South_America_S", "Taiwan_W","Taiwan_E"

%# Rrup is number in kilometers

%#Hypocentral depth in km. To use Ztor value to estimate hypocentral depth, see Ch. XXX of Parker et al. PEER report

%# period can be: (-1,0,0.01,0.02,0.025,0.03,0.04,0.05,0.075,0.1,0.15,0.2,0.25,0.3,0.4,0.5,0.75,1,1.5,2,2.5,3,4,5,7.5,10)
%# where -1 == PGV and 0 == PGA

%#VS30 in units m/s

%#Z2.5 in units m. Only used if DatabaseRegion == "Japan" or "Cascadia". Can also specify "default" to get no basin term
```

```
%#basin is only used if DatabaseRegion == "Cascadia". Value can be 0, 1, or 2, where 0 == having an estimate of Z2.5 outside
mapped basin, 1 == Seattle basin, and 0 == other mapped basin (Tacoma, Everett, Georgia, etc.)

%# Other pertinent information ----------------------------------------------
%#Coefficient files must be in the active working directory
%# "GMM_at_VS30_Slab_v2.R" calls function "GMM_at_760_Slab_v2.R" to compute PGAr in the nonlinear site term. This
function must be in the R environment else an error will occur.
%# The output is the desired median model prediction in LN units
%# Take the exponential to get PGA, PSA in g or  PGV in cm/s


%#Function to compute GMM predictions at various VS30s for slab


function PSHAB20_GMM_at_VS30_SlabValue = PSHAB20_GMM_at_VS30_Slab(EventType, UserRegion,
saturation_region_user, Rrup, M, hypocentral_depth, period, Vs30, Z2pt5, basin)
   % CODE DEVELOPED/IMPLEMENTED BY
   %         Silvia Mazzoni, 2020
   %          smazzoni@ucla.edu

 %    % % debug.print "PSHAB20_GMM_at_VS30_SlabValue"

  PSHAB20_GMM_at_VS30_SlabValue = -999;
  if (LeftNumber(EventType, 1)) == 0
    return % ("This function is only for slab")
  end
  region = getPSHABregion(UserRegion);
  saturation_region = getPSHABSubductionRegion(saturation_region_user);

  basin = LeftNumber(basin, 1);
  %%import coefficients
    CoefficientsTable = readtable('PSHAB20_Table_E2_Slab_Coefficients_OneRowHeader.csv','PreserveVariableNames',true);

   %Define mb based on Archuleta and Ji (2019)
   Mb = getMbDefault("PSHAB20",EventType, UserRegion, saturation_region);

% % Constant ----------------------------------------------------------------
%  %Isolate constant
  if region == "global"
    c0 = getRangeValueT(CoefficientsTable, period,  "Global_c0");
  elseif region == "Alaska" | region == "SA"
    c0 = getRangeValueT(CoefficientsTable, period,  saturation_region + "_c0");
  else
    c0 = getRangeValueT(CoefficientsTable, period,  region + "_c0");
  end
  if checkEmptyNA999(c0)
     c0 = getRangeValueT(CoefficientsTable, period,  "Global_c0");
  end

%%  Path Term ---------------------------------------------------------------

%  %near-source saturation
  if M <= Mb
    littleM = (Log10(35) - Log10(3.12)) / (Mb - 4);
    h = 10 ^ (littleM * (M - Mb) + Log10(35));
  else
    h = 35;
```

```
 end


 Rref = sqrt(1 + h ^ 2);
 r = sqrt(Rrup ^ 2 + h ^ 2);
 LogR = log(r);
 R_Rref = log(r / Rref);


% %Need  to isolate regional anelastic coefficient, a0
 if region == "global"
  a0 = getRangeValueT(CoefficientsTable, period,  "Global_a0");
 else
  a0 = getRangeValueT(CoefficientsTable, period,  region + "_a0");
 end
 if checkEmptyNA999(a0)
  a0 = getRangeValueT(CoefficientsTable, period,  "Global_a0");
 end

 c1 = getRangeValueT(CoefficientsTable, period,  "c1");
 b4 = getRangeValueT(CoefficientsTable, period,  "b4");
 Fp = (c1 * LogR + (b4 * M) * R_Rref + a0 * r);


%%  Magnitude Scaling ----------------------------------------------------------

  c4 = getRangeValueT(CoefficientsTable, period,  "c4");
  c5 = getRangeValueT(CoefficientsTable, period,  "c5");
  c6 = getRangeValueT(CoefficientsTable, period,  "c6");

 Fm = c4 * func1(M, Mb) + c6 * func2(M, Mb) + c5 * func3(M, Mb);

%%  Source Depth Scaling -------------------------------------------------------
  Db = getRangeValueT(CoefficientsTable, period,  "db_km");
  d = getRangeValueT(CoefficientsTable, period,  "d");
  littleM = getRangeValueT(CoefficientsTable, period,  "m");
 %%compute depth scaling term
 if hypocentral_depth >= Db
  Fd = d;
 elseif hypocentral_depth <= 20
  Fd = littleM * (20 - Db) + d;
 else
  Fd = littleM * (hypocentral_depth - Db) + d;
 end

%%  Linear Site Amplification ------------------------------------------------

  %% Site Coefficients
  V1 = getRangeValueT(CoefficientsTable, period,  "V1_m_s");
  V2 = getRangeValueT(CoefficientsTable, period,  "V2_m_s");
  Vref = getRangeValueT(CoefficientsTable, period,  "Vref_m_s");

 if region == "global" | region == "CAM"
  s2 = getRangeValueT(CoefficientsTable, period,  "Global_s2");
  s1 = s2;
 elseif region == "Taiwan" | region == "Japan"
  s2 = getRangeValueT(CoefficientsTable, period,  region + "_s2");
  s1 = getRangeValueT(CoefficientsTable, period,  region + "_s1");
```

```
  else
    s2 = getRangeValueT(CoefficientsTable, period,  region + "_s2");
    s1 = s2;
  end

  %%Compute linear site term
    if Vs30 <= V1
      Flin = s1 * log(Vs30 / V1) + s2 * log(V1 / Vref);
     elseif Vs30 <= V2
      Flin = s2 * log(Vs30 / Vref);
     else
      Flin = s2 * log(V2 / Vref);
    end

% % Nonlinear Site Term ----------------------------------------------------------
  PGAr = exp(PSHAB20_GMM_at_760_Slab(EventType, UserRegion, saturation_region, Rrup, M, hypocentral_depth, 0));
    f3 = 0.05;
    Vb = 200;
    Vref_Fnl = 760;

  if period >= 3
    Fnl = 0;
   else
    f4 = getRangeValueT(CoefficientsTable, period,  "f4");
    f5 = getRangeValueT(CoefficientsTable, period,  "f5");
    f2 = f4 * (exp(f5 * (min(Vs30, Vref_Fnl) - Vb)) - exp(f5 * (Vref_Fnl - Vb)));
    Fnl = 0 + f2 * log((PGAr + f3) / f3);
  end




%%  Basin Term ------------------------------------------------------------------
    if isstring(Z2pt5)
       if Z2pt5== "default" | Z2pt5== "Default"
          Fb = 0;
       end
    elseif Z2pt5 <= 0 | (region ~= "Japan" && region ~= "Cascadia")
       Fb = 0;
    else
       if region == "Cascadia"
         theta0 = 3.94;
         theta1 = -0.42;
         vmu = 200;
         vsig = 0.2;
         e1 = getRangeValueT(CoefficientsTable, period,  "C_e1");


          C_e3 = getRangeValueT(CoefficientsTable, period,  "C_e3");
          C_e2 = getRangeValueT(CoefficientsTable, period,  "C_e2");

        if basin == 0
          del_None = getRangeValueT(CoefficientsTable, period,  "del_None");
          e3 = C_e3 + del_None;
          e2 = C_e2 + del_None;
         elseif basin == 1
          del_Seattle = getRangeValueT(CoefficientsTable, period,  "del_Seattle");
          e3 = C_e3 + del_Seattle;
```

```
        e2 = C_e2 + del_Seattle;
       else
        e3 = C_e3;
        e2 = C_e2;
       end

      elseif region == "Japan"

       theta0 = 3.05;
       theta1 = -0.8;
       vmu = 500;
       vsig = 0.33;
       e3 = getRangeValueT(CoefficientsTable, period,  "J_e3");
       e2 = getRangeValueT(CoefficientsTable, period,  "J_e2");
       e1 = getRangeValueT(CoefficientsTable, period,  "J_e1");

      end

      Z2pt5_pred = 10 ^ (theta0 + theta1 * (1 + erf((Log10(Vs30) - Log10(vmu)) / (vsig * sqrt(2)))));
      delZ2pt5 = log(Z2pt5) - log(Z2pt5_pred);

      if delZ2pt5 <= (e1 / e3)
       Fb = e1;
      elseif delZ2pt5 >= (e2 / e3)
       Fb = e2;
      else
       Fb = e3 * delZ2pt5;
      end
   end

   mu = c0 + Fp + Fm + Fd + Fnl + Flin + Fb;



   PSHAB20_GMM_at_VS30_SlabValue = mu;

end

function Log10Value = Log10(x)
   % CODE DEVELOPED/IMPLEMENTED BY
   %         Silvia Mazzoni, 2020
   %          smazzoni@ucla.edu
   Log10Value = log(x) / log(10.);
end

% updated 6/1/2020 by Silvia Mazzoni
% GMM_at_760_IF_v4.R

%#Grace Parker
%#Modified February 26 to expand comments
%#Modified March 25, 2020 to call consolidated coefficient tables

%# Input Parameters ----------------------------------------------------

%#Event type: 0 == interface, 1 == slab

%#region corresponds to options in the DatabaseRegion column of the flatfile, plus global. Must be a string. If no matches,
default will be global model:
```

```
%# "global", "Alaska", "Cascadia", "CAM", "Japan", "SA" or "Taiwan"

%#Saturation Region corresponds to regions defined by R. Archuleta and C. Ji:
%# "global", "Aleutian","Alaska","Cascadia","CAM_S", "CAM_N", "Japan_Pac","Japan_Phi","SA_N","SA_S",
"Taiwan_W","Taiwan_E"

%# Rrup is number in kilometers

%# period can be: (-1,0,0.01,0.02,0.025,0.03,0.04,0.05,0.075,0.1,0.15,0.2,0.25,0.3,0.4,0.5,0.75,1,1.5,2,2.5,3,4,5,7.5,10)
%# where -1 == PGV and 0 == PGA

%# Other pertinent information ----------------------------------------------
%# Coefficient files must be in the active working directory
%# This function has no site term. Can only estimate ground motion at the reference condition VS30 = 760m/s
%# The output is the desired median model prediction in LN units
%# Take the exponential to get PGA, PSA in g or the PGV in cm/s




function PSHAB20_GMM_at_760_IFValue = PSHAB20_GMM_at_760_IF(EventType, UserRegion, saturation_region_user,
Rrup, M, period)
   % CODE DEVELOPED/IMPLEMENTED BY
   %         Silvia Mazzoni, 2020
   %          smazzoni@ucla.edu
 % % % debug.print "PSHAB20_GMM_at_760_IF"
  PSHAB20_GMM_at_760_IFValue = -999;
  if (LeftNumber(EventType, 1)) == 1 | (LeftNumber(EventType, 1)) == 5
    return; % ("This function is only for IF")
  end
  region = getPSHABregion(UserRegion);
  saturation_region = getPSHABSubductionRegion(saturation_region_user);

%%  Import Master Coefficient Table ----------------------------------------
  %%import CoefficientsTable
  CoefficientsTable =
readtable('PSHAB20_Table_E1_Interface_Coefficients_OneRowHeader.csv','PreserveVariableNames',true);

  %%Define Mb
    Mb = getMbDefault("PSHAB20",EventType, UserRegion, saturation_region);

%%  Constant ---------------------------------------------------------------
  if region == "global"
    c0 = getRangeValueT(CoefficientsTable, period, "Global_c0");
  else
    c0 = getRangeValueT(CoefficientsTable, period, saturation_region + "_c0");
  end

%%  Path Term --------------------------------------------------------------
  h = 10 ^ (-0.82 + 0.252 * M);
  Rref = sqrt(1 + h ^ 2);
  r = sqrt(Rrup ^ 2 + h ^ 2);
  LogR = log(r);
  R_Rref = log(r / Rref);

%  %Need  to isolate regional anelastic coefficient, a0
  if region == "global"
```

```
   a0 = getRangeValueT(CoefficientsTable, period, "Global_a0");
  else
   a0 = getRangeValueT(CoefficientsTable, period, region + "_a0");
  end

  if checkEmptyNA999(a0)
   a0 = getRangeValueT(CoefficientsTable, period, "Global_a0");
  end

  c1 = getRangeValueT(CoefficientsTable, period, "c1");
  b4 = getRangeValueT(CoefficientsTable, period, "b4");
  Fp = (c1 * LogR + (b4 * M) * R_Rref + a0 * r);



%%  Magnitude Scaling ----------------------------------------------------------
  c4 = getRangeValueT(CoefficientsTable, period, "c4");
  c5 = getRangeValueT(CoefficientsTable, period, "c5");
  c6 = getRangeValueT(CoefficientsTable, period, "c6");
  Fm = c4 * func1(M, Mb) + c6 * func2(M, Mb) + c5 * func3(M, Mb);

%%  Add it all up! -------------------------------------------------------------


  mu = c0 + Fp + Fm;

  PSHAB20_GMM_at_760_IFValue = mu;


end


% Updated by Silvia Mazzoni 6/1/2020

% GMM_at_Vs30_IF_v4.R

%#Grace Parker
%#Modified February 26, 2020, to expand comments
%#Modified March 25, 2020, to take coefficients from "Table_E1_Interface_Coefficients.csv"

%# Input Parameters -----------------------------------------------------------

%#Event type: 0 == interface, 1 == slab

%#region corresponds to options in the DatabaseRegion column of the flatfile, plus global. Must be a string. If no matches,
default will be global model:
  %# "global", "Alaska", "Cascadia", "CAM", "Japan", "SA" or "Taiwan"

%#Saturation Region corresponds to regions defined by R. Archuleta and C. Ji:
  %# "global", "Aleutian","Alaska","Cascadia","CAM_S", "CAM_N", "Japan_Pac","Japan_Phi","SA_N","SA_S",
"Taiwan_W","Taiwan_E"

%# Rrup is number in kilometers

%# period can be: (-1,0,0.01,0.02,0.025,0.03,0.04,0.05,0.075,0.1,0.15,0.2,0.25,0.3,0.4,0.5,0.75,1,1.5,2,2.5,3,4,5,7.5,10)
  %# where -1 == PGV and 0 == PGA

%#VS30 in units m/s
```

```
%#Z2.5 in units m. Only used if DatabaseRegion == "Japan" or "Cascadia". Can also specify "default" to get no basin term

%#basin is only used if DatabaseRegion == "Cascadia". Value can be 0, 1, or 2, where 0 == having an estimate of Z2.5 outside
mapped basin, 1 == Seattle basin, and 0 == other mapped basin (Tacoma, Everett, Georgia, etc.)

%# Other pertinent information ----------------------------------------------
%#Coefficient files must be in the active working directory
%# "GMM_at_VS30_IF_v3.R" calls function "GMM_at_760_IF_v2.R" to compute PGAr in the nonlinear site term. This
function must be in the R environment else an error will occur.
%# The output is the desired median model prediction in LN units
%
function PSHAB20_GMM_at_VS30_IFValue = PSHAB20_GMM_at_VS30_IF(EventType, UserRegion, saturation_region_user,
Rrup, M, period, Vs30, Z2pt5, basin)
   % CODE DEVELOPED/IMPLEMENTED BY
   %         Silvia Mazzoni, 2020
   %          smazzoni@ucla.edu

%  % % debug.print "PSHAB20_GMM_at_VS30_IF"
  PSHAB20_GMM_at_VS30_IFValue = -999;
  if (LeftNumber(EventType, 1)) == 1 | (LeftNumber(EventType, 1)) == 5
    return % ("This function is only for IF")
  end
  region = getPSHABregion(UserRegion);
  saturation_region = getPSHABSubductionRegion(saturation_region_user);
  basin = LeftNumber(basin, 1);

%%  Import Master Coefficient Table ----------------------------------------
  CoefficientsTable =
readtable('PSHAB20_Table_E1_Interface_Coefficients_OneRowHeader.csv','PreserveVariableNames',true);


  %%Define Mb
      Mb = getMbDefault("PSHAB20",EventType, UserRegion, saturation_region);

%%  Constant ---------------------------------------------------------------
    if region == "global"
       c0 = getRangeValueT(CoefficientsTable, period, "Global_c0");
    else
       c0 = getRangeValueT(CoefficientsTable, period, saturation_region + "_c0");
    end
    if checkEmptyNA999(c0)
       c0 = getRangeValueT(CoefficientsTable, period, "Global_c0");
    end

%%  Path Term --------------------------------------------------------------
  h = 10 ^ (-0.82 + 0.252 * M);
  Rref = sqrt(1 + h ^ 2);
  r = sqrt(Rrup ^ 2 + h ^ 2);
  LogR = log(r);
  R_Rref = log(r / Rref);

%  %Need  to isolate regional anelastic coefficient, a0
  if region == "global"
    a0 = getRangeValueT(CoefficientsTable, period, "Global_a0");
   else
    a0 = getRangeValueT(CoefficientsTable, period, region + "_a0");
  end
```

```
 if checkEmptyNA999(a0)
  a0 = getRangeValueT(CoefficientsTable, period, "Global_a0");
 end

 c1 = getRangeValueT(CoefficientsTable, period, "c1");
 b4 = getRangeValueT(CoefficientsTable, period, "b4");
 Fp = (c1 * LogR + (b4 * M) * R_Rref + a0 * r);



%%  Magnitude Scaling ----------------------------------------------------------

   c4 = getRangeValueT(CoefficientsTable, period, "c4");
   c5 = getRangeValueT(CoefficientsTable, period, "c5");
   c6 = getRangeValueT(CoefficientsTable, period, "c6");

 Fm = c4 * func1(M, Mb) + c6 * func2(M, Mb) + c5 * func3(M, Mb);


% % Linear Site Amplification ---------------------------------------------


   %% Site Coefficients
   V1 = getRangeValueT(CoefficientsTable, period, "V1_m_s");
   V2 = getRangeValueT(CoefficientsTable, period, "V2_m_s");
   Vref = getRangeValueT(CoefficientsTable, period, "Vref_m_s");

 if region == "global" | region == "CAM"
   s2 = getRangeValueT(CoefficientsTable, period, "Global_s2");
   s1 = s2;
  elseif region == "Taiwan" | region == "Japan"
   s2 = getRangeValueT(CoefficientsTable, period, region + "_s2");
   s1 = getRangeValueT(CoefficientsTable, period, region + "_s1");
  else
   s2 = getRangeValueT(CoefficientsTable, period, region + "_s2");
   s1 = s2;
 end

 %%Compute linear site term
   if Vs30 <= V1
     Flin = s1 * log(Vs30 / V1) + s2 * log(V1 / Vref);
    elseif Vs30 <= V2
     Flin = s2 * log(Vs30 / Vref);
    else
     Flin = s2 * log(V2 / Vref);
   end


% % Nonlinear Site Term ---------------------------------------------------
 PGAr = exp(PSHAB20_GMM_at_760_IF(EventType, UserRegion, saturation_region, Rrup, M, 0));
 % % debug.print "PGAr: " & PGAr
   f3 = 0.05;
   Vb = 200;
   Vref_Fnl = 760;
 if period >= 3
   Fnl = 0;
  else
```

```
  f4 = getRangeValueT(CoefficientsTable, period, "f4");
  f5 = getRangeValueT(CoefficientsTable, period, "f5");
  f2 = f4 * (exp(f5 * (min(Vs30, Vref_Fnl) - Vb)) - exp(f5 * (Vref_Fnl - Vb)));
  Fnl = 0 + f2 * log((PGAr + f3) / f3);
 end


%%  Basin Term ----------------------------------------------------------------
  if isstring(Z2pt5)
     if Z2pt5== "default" | Z2pt5== "Default"
        Fb = 0;
     end
  elseif Z2pt5 <= 0 | (region ~= "Japan" && region ~= "Cascadia")
     Fb = 0;
  else
     if region == "Cascadia"
      theta0 = 3.94;
      theta1 = -0.42;
      vmu = 200;
      vsig = 0.2;
      e1 = getRangeValueT(CoefficientsTable, period, "C_e1");


      if basin == 0
       C_e3 = getRangeValueT(CoefficientsTable, period, "C_e3");
       C_e2 = getRangeValueT(CoefficientsTable, period, "C_e2");
       del_None = getRangeValueT(CoefficientsTable, period, "del_None");
       e3 = C_e3 + del_None;
       e2 = C_e2 + del_None;
      elseif basin == 1
       C_e3 = getRangeValueT(CoefficientsTable, period, "C_e3");
       C_e2 = getRangeValueT(CoefficientsTable, period, "C_e2");
       del_Seattle = getRangeValueT(CoefficientsTable, period, "del_Seattle");
       e3 = C_e3 + del_Seattle;
       e2 = C_e2 + del_Seattle;
      else
       C_e3 = getRangeValueT(CoefficientsTable, period, "C_e3");
       C_e2 = getRangeValueT(CoefficientsTable, period, "C_e2");
       e3 = C_e3;
       e2 = C_e2;
      end

     elseif region == "Japan"
      theta0 = 3.05;
      theta1 = -0.8;
      vmu = 500;
      vsig = 0.33;
      e3 = getRangeValueT(CoefficientsTable, period, "J_e3");
      e2 = getRangeValueT(CoefficientsTable, period, "J_e2");
      e1 = getRangeValueT(CoefficientsTable, period, "J_e1");
     end

     Z2pt5_pred = 10 ^ (theta0 + theta1 * (1 + erf((Log10(Vs30) - Log10(vmu)) / (vsig * sqrt(2)))));
     delZ2pt5 = log(Z2pt5) - log(Z2pt5_pred);


     if delZ2pt5 <= (e1 / e3)
      Fb = e1;
```

```
        elseif delZ2pt5 >= (e2 / e3)
         Fb = e2;
        else
         Fb = e3 * delZ2pt5;
       end
    end


%  % Add it all up! ------------------------------------------------------------

   mu = c0 + Fp  + Fnl + Fb + Flin + Fm;

  PSHAB20_GMM_at_VS30_IFValue = mu;
end









function PSHAB20_MedianValue = PSHAB20_Median(EventType, UserRegion, SubductionSlab, Rrup, Magnitude, Zhypo,
period, Vs30, Z2pt5, PNWbasinStrux)
    % CODE DEVELOPED/IMPLEMENTED BY
    %         Silvia Mazzoni, 2020
    %          smazzoni@ucla.edu

%    % % debug.print "PSHAB20_Median"
    thisOut = -999;
    XinterpMin = 0;
    XinterpMax = 10;
    XinterpType = "log";
    YinterpType = "log";
    extrapolateType = "extrapolate";

    Parameters = readtable('PSHAB20_Table_E2_Slab_Coefficients_OneRowHeader.csv','PreserveVariableNames',true);
    TvalueList = Parameters.T;

    InterpArray = interpolateFunction(period, TvalueList, XinterpType, extrapolateType, XinterpMin, XinterpMax);

    period0 = TvalueList(InterpArray(1), 1);
    y0 = PSHAB20_Median_AtTlist(EventType, UserRegion, SubductionSlab, Rrup, Magnitude, Zhypo, period0, Vs30, Z2pt5,
PNWbasinStrux);

    if InterpArray(2) <= 0
       thisOut = y0;
    else
       period1 = TvalueList(InterpArray(2), 1);
       y1 = PSHAB20_Median_AtTlist(EventType, UserRegion, SubductionSlab, Rrup, Magnitude, Zhypo, period1, Vs30, Z2pt5,
PNWbasinStrux);
       b = y1
       if lower(LeftString(YinterpType, strlength("log"))) == lower("log")
          if y0 <= 0
             y0 = 0.000000001;
          end
```

```
        if y1 <= 0
           y1 = 0.000000001;
        end
        y0 = log(y0);
        y1 = log(y1);
     end
     thisOut = y0 + (y1 - y0) * InterpArray(3);
     if lower(LeftString(YinterpType, strlength("log"))) == lower("log")
        thisOut = exp(thisOut);
     end

  end

  PSHAB20_MedianValue = thisOut;

end

function PSHAB20_Median_AtTlistValue = PSHAB20_Median_AtTlist(EventType, UserRegion, saturation_region, Rrup, M,
hypocentral_depth, period, Vs30, Z2pt5, basin)
  % CODE DEVELOPED/IMPLEMENTED BY
  %        Silvia Mazzoni, 2020
  %         smazzoni@ucla.edu
  if LeftNumber(EventType, 1) == 1 | LeftNumber(EventType, 1) == 5
     PSHAB20_Median_AtTlistValue = exp(PSHAB20_GMM_at_VS30_Slab(EventType, UserRegion, saturation_region, Rrup,
M, hypocentral_depth, period, Vs30, Z2pt5, basin));
  else
     PSHAB20_Median_AtTlistValue = exp(PSHAB20_GMM_at_VS30_IF(EventType, UserRegion, saturation_region, Rrup, M,
period, Vs30, Z2pt5, basin));
  end

end



function PSHAB20_SigmaEpistemicValue = PSHAB20_SigmaEpistemic(Ts, EventType, saturation_region_user)

  % CODE DEVELOPED/IMPLEMENTED BY
  %        Silvia Mazzoni, 2020
  %         smazzoni@ucla.edu

  PSHAB20_SigmaEpistemicValue = -999;

 if (LeftNumber(EventType, 1)) == 0
     CoefficientsTable = readtable("PSHAB20_EpistemicModelCoefficients_Interface.csv");
 else
     CoefficientsTable = readtable("PSHAB20_EpistemicModelCoefficients_Slab.csv");
 end


  saturation_region = getPSHABSubductionRegion(saturation_region_user);
  if saturation_region == "Taiwan_W" || saturation_region == "Taiwan_E"
           saturation_region = "Taiwan";
  end

  if saturation_region == "New_Zealand_N" || saturation_region == "New_Zealand_S"
           saturation_region = "global";
  end
```

```
   SigEp1 = getRangeValueRegion(CoefficientsTable, saturation_region,  'SigEp1');
   if SigEp1 == -999
      saturation_region = "global";
      SigEp1 = getRangeValueRegion(CoefficientsTable, saturation_region,  'SigEp1');
   end
   SigEp2 = getRangeValueRegion(CoefficientsTable, saturation_region, 'SigEp2');
   T1 = getRangeValueRegion(CoefficientsTable, saturation_region, 'T1');
   T2 = getRangeValueRegion(CoefficientsTable, saturation_region, 'T2');
   if Ts < T1 || Ts == 0
      sigma_epi = SigEp1;
   elseif Ts > T2
      sigma_epi = SigEp2;
   else
      sigma_epi = SigEp1 - (SigEp1 - SigEp2) * (log(Ts / T1) / log(T2 / T1));
   end
   PSHAB20_SigmaEpistemicValue = sigma_epi;

end




function PSHAB20_SigmaAleatoryValue = PSHAB20_SigmaAleatory(period, Rrup, Vs30)
   % CODE DEVELOPED/IMPLEMENTED BY
   %        Silvia Mazzoni, 2020
   %         smazzoni@ucla.edu

   thisOut = -999;
   XinterpMin = 0;
   XinterpMax = 10;
   XinterpType = "log";
   YinterpType = "linear";
   extrapolateType = "extrapolate";

   Parameters = readtable('PSHAB20_Table_E2_Slab_Coefficients_OneRowHeader.csv','PreserveVariableNames',true);
   TvalueList = Parameters.T;
   InterpArray = interpolateFunction(period, TvalueList, XinterpType, extrapolateType, XinterpMin, XinterpMax);

   period0 = TvalueList(InterpArray(1), 1);
   y0 = PSHAB20_SigmaAleatoryAtTlist(period0, Rrup, Vs30);

   if InterpArray(2) <= 0
      thisOut = y0;
   else
      period1 = TvalueList(InterpArray(2), 1);
      y1 = PSHAB20_SigmaAleatoryAtTlist(period1, Rrup, Vs30);
      if lower(LeftString(YinterpType, strlength("log"))) == lower("log")
         if y0 <= 0
            y0 = 0.000000001;
         end
         if y1 <= 0
            y1 = 0.000000001;
         end
         y0 = log(y0);
         y1 = log(y1);
      end
      thisOut = y0 + (y1 - y0) * InterpArray(3);
```

```
      if lower(LeftString(YinterpType, strlength("log"))) == lower("log")
         thisOut = exp(thisOut);
      end

   end

   PSHAB20_SigmaAleatoryValue = thisOut;

end

function PSHAB20_SigmaAleatoryAtTlistValue = PSHAB20_SigmaAleatoryAtTlist(period, Rrup, Vs30)
   % CODE DEVELOPED/IMPLEMENTED BY
   %        Silvia Mazzoni, 2020
   %         smazzoni@ucla.edu

   PSHAB20_SigmaAleatoryAtTlistValue = -999;

   PhiTot = PSHAB20_PhiTot(period, Rrup, Vs30);
   Tau = PSHAB20_Tau(period, Rrup, Vs30);


   PSHAB20_SigmaAleatoryAtTlistValue = sqrt(PhiTot * PhiTot + Tau * Tau);

end



function PSHAB20_TauValue = PSHAB20_Tau(period, Rrup, Vs30)
   % CODE DEVELOPED/IMPLEMENTED BY
   %        Silvia Mazzoni, 2020
   %         smazzoni@ucla.edu

   PSHAB20_TauValue = -999;
   SigmacoefficientsTable =
readtable("PSHAB20_Table_E3_AleatoryCoefficients_OneRowHeader.csv",'PreserveVariableNames',true);
   Tau = getRangeValueT(SigmacoefficientsTable, period, "Tau");
   PSHAB20_TauValue = Tau;
end


function PSHAB20_PhiTotValue = PSHAB20_PhiTot(period, Rrup, Vs30)
   % CODE DEVELOPED/IMPLEMENTED BY
   %        Silvia Mazzoni, 2020
   %         smazzoni@ucla.edu

   PSHAB20_PhiTotValue = -999;
   SigmacoefficientsTable =
readtable("PSHAB20_Table_E3_AleatoryCoefficients_OneRowHeader.csv",'PreserveVariableNames',true);

   PhiTot_Phi1squared = getRangeValueT(SigmacoefficientsTable, period, "PhiTot_Phi1squared");
   PhiTot_Phi2squared = getRangeValueT(SigmacoefficientsTable, period, "PhiTot_Phi2squared");
   PhiTot_PhiVsquared = getRangeValueT(SigmacoefficientsTable, period, "PhiTot_PhiVsquared");

   %Corner Distances:
   R1 = 200;    % km
   R2 = 500;    % km

   V1 = 200;    % m/s
```

```
   V2 = 500;    % m/s

   if Vs30 <= V1
      Rprime = max(R1, min(R2, Rrup));
      deltaVar = PhiTot_PhiVsquared * (log(R2 / Rprime)) / (log(R2 / R1));
   elseif Vs30 < V2
      Rprime = max(R1, min(R2, Rrup));
      deltaVar = PhiTot_PhiVsquared * (log(R2 / Rprime)) / (log(R2 / R1)) * (log(V2 / Vs30)) / (log(V2 / V1));
   else
      deltaVar = 0;
   end

   if Rrup < R1
      phiSquared = PhiTot_Phi1squared;
   elseif Rrup < R2
      phiSquared = (PhiTot_Phi2squared - PhiTot_Phi1squared) / (log(R2 / R1)) * (log(Rrup / R1)) + PhiTot_Phi1squared;
   else
      phiSquared = PhiTot_Phi2squared;

   end

   PSHAB20_PhiTotValue = sqrt(phiSquared + deltaVar);

end

function PSHAB20_SigmaS2SValue = PSHAB20_SigmaS2S(period, Rrup, Vs30)
   % CODE DEVELOPED/IMPLEMENTED BY
   %         Silvia Mazzoni, 2020
   %          smazzoni@ucla.edu

   PSHAB20_SigmaS2SValue = -999;
   SigmacoefficientsTable =
readtable("PSHAB20_Table_E3_AleatoryCoefficients_OneRowHeader.csv",'PreserveVariableNames',true);

   PhiS2S_PhiS2S0squared = getRangeValueT(SigmacoefficientsTable, period, "PhiS2S_PhiS2S0squared");
   PhiS2S_a1 = getRangeValueT(SigmacoefficientsTable, period, "PhiS2S_a1");
   VM = getRangeValueT(SigmacoefficientsTable, period, "VM");

   %Corner Distances:
   R3 = 200;    % km
   R4 = 500;    % km

   V3 = 200;    % m/s
   V4 = 800;    % m/s

   if Vs30 <= V3
      Rprime = max(R3, min(R4, Rrup));
      deltaVarS2S = PhiS2S_a1 * log(V3 / VM) * (log(R4 / Rprime)) / (log(R4 / R3));
   elseif Vs30 < VM
      Rprime = max(R3, min(R4, Rrup));
      deltaVarS2S = PhiS2S_a1 * log(Vs30 / VM) * (log(R4 / Rprime)) / (log(R4 / R3));
   elseif Vs30 < V4
      deltaVarS2S = PhiS2S_a1 * log(Vs30 / VM);
   else
      deltaVarS2S = PhiS2S_a1 * log(V4 / VM);
   end

   PSHAB20_SigmaS2SValue = sqrt(PhiS2S_PhiS2S0squared + deltaVarS2S);
```

```
end


function PSHAB20_SigmaSSValue = PSHAB20_SigmaSS(period, Rrup, Vs30)
   % CODE DEVELOPED/IMPLEMENTED BY
   %         Silvia Mazzoni, 2020
   %          smazzoni@ucla.edu

   PSHAB20_SigmaSSValue = -999;
   SigmacoefficientsTable =
readtable("PSHAB20_Table_E3_AleatoryCoefficients_OneRowHeader.csv",'PreserveVariableNames',true);

   PhiSS_PhiSS1squared = getRangeValueT(SigmacoefficientsTable, period, "PhiSS_PhiSS1squared");
    PhiSS_PhiSS2squared = getRangeValueT(SigmacoefficientsTable, period, "PhiSS_PhiSS2squared");
   PhiSS_a2 = getRangeValueT(SigmacoefficientsTable, period, "PhiSS_a2");
   VM = getRangeValueT(SigmacoefficientsTable, period, "VM");

   %Corner Distances:
   R3 = 200;    % km
   R4 = 500;    % km
   R5 = 500;    % km
   R6 = 800;    % km

   V3 = 200;    % m/s
   V4 = 800;    % m/s

   if Rrup < R5
      phiSquaredSS = PhiSS_PhiSS1squared;
   elseif Rrup < R6
      phiSquaredSS = (PhiSS_PhiSS2squared - PhiSS_PhiSS1squared) / (log(R6 / R5)) * (log(Rrup / R5)) + PhiSS_PhiSS1squared;
   else
      phiSquaredSS = PhiSS_PhiSS2squared;

   end

   if Vs30 <= V3
      Rprime = max(R3, min(R4, Rrup));
      deltaVarSS = PhiSS_a2 * log(V3 / VM) * (log(R4 / Rprime)) / (log(R4 / R3));
   elseif Vs30 < VM
      Rprime = max(R3, min(R4, Rrup));
      deltaVarSS = PhiSS_a2 * log(Vs30 / VM) * (log(R4 / Rprime)) / (log(R4 / R3));
   elseif Vs30 < V4
      deltaVarSS = PhiSS_a2 * log(Vs30 / VM);
   else
      deltaVarSS = PhiSS_a2 * log(V4 / VM);
   end


   PSHAB20_SigmaSSValue = sqrt(phiSquaredSS + deltaVarSS);

end
```

```
function NGAsubGMM_MedianPlusMinusSigmaValue = NGAsubGMM_MedianPlusMinusSigma(UserPeriod,  UserRegion,
Magnitude, Vs30, Rrup, AlphaBackarc, AlphaNankai, Ztor, Zhypo, EventType, Z1pt0, Z2pt5, MbUser, SubductionSlab,
PNWbasinStrux, Weight_KBCG20, Weight_PSHAB20,  User_NSigma, User_EpiInSigmaModels, User_NsampleEpi
,User_MeanType)
   % CODE DEVELOPED/IMPLEMENTED BY
   %         Silvia Mazzoni, 2020
   %          smazzoni@ucla.edu
    NGAsubGMM_MedianPlusMinusSigmaValue = 0;
    if nargin<21
       User_MeanType= "geometric";
    end

              if MbUser>0
                 Mb = MbUser;
              else
                 Mb =getMbDefault("KBCG20",EventType, UserRegion, SubductionSlab);
              end




   period = getTFvalue(UserPeriod);
   WeightedSum = 0;
   WeightSum = 0;

   if Weight_KBCG20 > 0
      KBCG20median = KBCG20_medPSA(period, Magnitude, Rrup, AlphaBackarc, AlphaNankai, Ztor, EventType, Vs30, Z1pt0,
Z2pt5, Mb, UserRegion, PNWbasinStrux);
      SigmaAleatoryKBCG20 = KBCG20_sigmaAleatory(period);
      if LeftNumber(User_EpiInSigmaModels, 1) == 1  ; % all models
         SigmaEpistemicKBCG20 = KBCG20_SigmaEpistemic(period, Magnitude, Rrup, AlphaBackarc, AlphaNankai, Ztor,
EventType, Vs30, Z1pt0, Z2pt5, Mb, UserRegion, User_NsampleEpi, PNWbasinStrux);
      else    % none
         SigmaEpistemicKBCG20 = 0;
      end
     KBCG20SigmaTotal = sqrt(SigmaAleatoryKBCG20 * SigmaAleatoryKBCG20 + SigmaEpistemicKBCG20 *
SigmaEpistemicKBCG20);
      KBCG20 = exp(log(KBCG20median)+User_NSigma*KBCG20SigmaTotal);

      WeightSum = WeightSum + Weight_KBCG20;
      if lower(User_MeanType) == "arithmetic"
         WeightedSum = WeightedSum + KBCG20 * Weight_KBCG20;
      else
         WeightedSum = WeightedSum + log(KBCG20) * Weight_KBCG20;
      end
   end
   if Weight_PSHAB20 > 0
     PSHAB20median = PSHAB20_Median(EventType, UserRegion, SubductionSlab, Rrup, Magnitude, Zhypo, period, Vs30,
Z2pt5, PNWbasinStrux);
      SigmaAleatoryPSHAB20 = PSHAB20_SigmaAleatory(period, Rrup, Vs30);
      if LeftNumber(User_EpiInSigmaModels, 1) == 1  ; % all models
         SigmaEpistemicPSHAB20 = PSHAB20_SigmaEpistemic(period,EventType,SubductionSlab);
      else    % none
         SigmaEpistemicPSHAB20 = 0;
      end
     PSHAB20SigmaTotal = sqrt(SigmaAleatoryPSHAB20 * SigmaAleatoryPSHAB20 + SigmaEpistemicPSHAB20 *
SigmaEpistemicPSHAB20);
```

```
      PSHAB20 = exp(log(PSHAB20median)+User_NSigma*PSHAB20SigmaTotal);


      WeightSum = WeightSum + Weight_PSHAB20;
      if lower(User_MeanType) == "arithmetic"
         WeightedSum = WeightedSum + PSHAB20 * Weight_PSHAB20;
      else
         WeightedSum = WeightedSum + log(PSHAB20) * Weight_PSHAB20;
      end
   end
   if WeightSum>0
      if lower(User_MeanType) == "arithmetic"
         NGAsubGMM_MedianPlusMinusSigmaValue = WeightedSum / WeightSum;
      else
         NGAsubGMM_MedianPlusMinusSigmaValue = exp(WeightedSum / WeightSum);
      end
   end

end





function getTFvalueValue = getTFvalue(inValue)
   % CODE DEVELOPED/IMPLEMENTED BY
   %         Silvia Mazzoni, 2020
   %          smazzoni@ucla.edu

   % Code Written and developed by Silvia Mazzoni, smazzoni@ucla.edu, April 2020
   if ischar(inValue)
      if lower(inValue) == lower("pga")
         getTFvalueValue = 0;
      elseif lower(inValue) == lower("pgv")
         getTFvalueValue = -1;
      end
   else
      getTFvalueValue = inValue;
   end
end



% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function RunSensitivityStudy()
   prompt = 'Are you sure you want to run the Sensitivity Study? I may take a few minutes! y/n [n]:';
   str = input(prompt,'s');
   if isempty(str)
      str = 'N';
   end
   if lower(str) == 'y'
      disp(' %%% Performing Sensitivity Study')
      for iSet = 1:2
         thisCase = iSet;
         if iSet == 1    ; % interface

              CaseLabel(iSet,:) =
["Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_
```

NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.02","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.03","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.05","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.075","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.1","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.15","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.2","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.25","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.3","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.4","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.5","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.75","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=1","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=1.5","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=2","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=3","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=4","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=5","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=7.5","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=10","Mag=4,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=4.25,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=4.5,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=4.75,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=5,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=5.25,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=5.5,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=5.75,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m

=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=6,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=6.25,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=6.5,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=6.75,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=7,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=7.25,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=7.5,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=7.75,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8.25,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8.5,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8.75,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=9,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=9.25,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=10,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=13,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=16.9,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=21.97,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=28.561,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=37.1293,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=48.26809,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=62.7485169999999,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=81.5730720999999,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=106.04499373,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=137.858491849,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=179.2160394037,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=232.980851224809,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=302.87

5106592252,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=393.737638569928,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=511.858930140906,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=665.416609183177,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=865.04159193813,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=1124.55406951957,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=1461.92029037544,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=1900.49637748807,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=2470.64529073449,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=150,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=169.5,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=191.535,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=216.43455,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=244.5710415,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=276.365276895,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=312.29276289135,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=352.890822067225,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=398.766628935964,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=450.60629069764,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=509.185108488333,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=575.379172591816,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=650.178465028752,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=734.701665482489,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=830.212881995213,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=938.140556654591,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=1060.09882901969,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=1197.91167679225,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsig

```
ma=1,T_sec=0.01","Mag=8,Vs30_mps=1353.64019477524,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,In
terIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global
,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=1529.61342009602,Reg=0_global,Rrup_km=200,alpBack=0,alp
Nank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhyp
o_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=1728.4631647085,Reg=0_global,Rrup_
km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=
100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=1953.163376120
61,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,
Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30
_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_
m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag
=8,Vs30_mps=760,Reg=1_Alaska,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBa
sin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=8.6,Zhypo_km=55,SatReg=Alaska,Basin=0_NoBasin,Nsigma=1,T_sec=0.0
1","Mag=8,Vs30_mps=760,Reg=1_Alaska,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=
0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=8,Zhypo_km=55,SatReg=Aleutian,Basin=0_NoBasin,Nsigma=1,T
_sec=0.01","Mag=8,Vs30_mps=760,Reg=2_Cascadia,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface
,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.7,Zhypo_km=55,SatReg=Cascadia,Basin=0_NoBasin
,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=2_Cascadia,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra
=0_Interface,SeatBasn=1_InSeattleBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.7,Zhypo_km=55,SatReg=Cascadi
a,Basin=1_InSeattleBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=2_Cascadia,Rrup_km=200,alpBack=0,alpNank
=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=2_InOtherPNWbasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.7,Z
hypo_km=55,SatReg=Cascadia,Basin=2_InOtherPNWbasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=3_CentralAm
erica&Mexico,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=2_InOtherPNWbasin,Z1.0_
m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.4,Zhypo_km=55,SatReg=Central_America_N,Basin=0_NoBasin,Nsigma=1,T_sec
=0.01","Mag=8,Vs30_mps=760,Reg=3_CentralAmerica&Mexico,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=
0_Interface,SeatBasn=2_InOtherPNWbasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.4,Zhypo_km=55,SatReg=Cent
ral_America_S,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=4_Japan,Rrup_km=200,alpBack=0,alpN
ank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=2_InOtherPNWbasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=8
.5,Zhypo_km=55,SatReg=Japan_Pac,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=4_Japan,Rrup_k
m=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=2_InOtherPNWbasin,Z1.0_m=550,Z2.5_m=2000,
NsplEpi=100,Mbreak=7.7,Zhypo_km=55,SatReg=Japan_Phi,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760
,Reg=5_NewZealand,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=2_InOtherPNWbasin
,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=New_Zealand_N,Basin=0_NoBasin,Nsigma=1,T_
sec=0.01","Mag=8,Vs30_mps=760,Reg=5_NewZealand,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interf
ace,SeatBasn=2_InOtherPNWbasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=New_Zealan
d_S,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=6_SouthAmerica,Rrup_km=200,alpBack=0,alpNan
k=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=2_InOtherPNWbasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=8.5,
Zhypo_km=55,SatReg=South_America_N,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=6_SouthAm
erica,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=2_InOtherPNWbasin,Z1.0_m=550,Z
2.5_m=2000,NsplEpi=100,Mbreak=8.6,Zhypo_km=55,SatReg=South_America_S,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","M
ag=8,Vs30_mps=760,Reg=7_Taiwan,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=2_In
OtherPNWbasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.1,Zhypo_km=55,SatReg=Taiwan_E,Basin=0_NoBasin,Nsi
gma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=7_Taiwan,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_In
terface,SeatBasn=2_InOtherPNWbasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.1,Zhypo_km=55,SatReg=Taiwan_
W,Basin=0_NoBasin,Nsigma=1,T_sec=0.01"];
            CaseLabelRegion(iSet,:) =
["0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_
Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB
_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","
0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_M
b7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_
GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0
_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb
7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_G
LB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_
GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7
.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GL
B_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_
```

```
GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7
.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GL
B_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_
GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","1_AK_AK_Mb8.6","1_AK_Aleu_Mb8","2_CSC_CSC_Mb7.7","2
_CSC_CSC_Mb7.7_1_InSeattleBasin","2_CSC_CSC_Mb7.7_2_InOtherPNWbasin","3_CAM_CAM_N_Mb7.4","3_CAM_CAM_S
_Mb7.4","4_JP_JP_Pac_Mb8.5","4_JP_JP_Phi_Mb7.7","5_NZ_NZ_N_Mb7.9","5_NZ_NZ_S_Mb7.9","6_SAM_SAM_N_Mb8.5"
,"6_SAM_SAM_S_Mb8.6","7_TW_TW_E_Mb7.1","7_TW_TW_W_Mb7.1"];

            Mag(iSet,:) =
[8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,4,4.25,4.5,4.75,5,5.25,5.5,5.75,6,6.25,6.5,6.75,7,7.25,7.5,7.75,8,8.25,8.5,8.75,9,9.25,
8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8];
            Vs30mps(iSet,:) =
[760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,7
60,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760
,760,760,760,760,760,760,760,760,150,169.5,191.535,216.43455,244.5710415,276.365276895,312.29276289135,352.8908
22067225,398.766628935964,450.60629069764,509.185108488333,575.379172591816,650.178465028752,734.701665482
489,830.212881995213,938.140556654591,1060.09882901969,1197.91167679225,1353.64019477524,1529.61342009602,
1728.4631647085,1953.16337612061,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760];
            Reg(iSet,:) =
["0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_global","
0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_
global","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_gl
obal","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_glo
bal","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_globa
l","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_global",
"0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0
_global","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_global","1_Alaska","1_
Alaska","2_Cascadia","2_Cascadia","2_Cascadia","3_CentralAmerica&Mexico","3_CentralAmerica&Mexico","4_Japan","4_Ja
pan","5_NewZealand","5_NewZealand","6_SouthAmerica","6_SouthAmerica","7_Taiwan","7_Taiwan"];
            Rrup_km(iSet,:) =
[200,200,200,200,200,200,200,200,200,200,200,200,200,200,200,200,200,200,200,200,200,200,200,200,200,200,200,200,2
00,200,200,200,200,200,200,200,200,200,200,200,200,200,200,10,13,16.9,21.97,28.561,37.1293,48.26809,62.74851699999
99,81.5730720999999,106.04499373,137.858491849,179.2160394037,232.980851224809,302.875106592252,393.7376385
69928,511.858930140906,665.416609183177,865.04159193813,1124.55406951957,1461.92029037544,1900.49637748807
,2470.64529073449,200,200,200,200,200,200,200,200,200,200,200,200,200,200,200,200,200,200,200,200,200,200,200,200
,200,200,200,200,200,200,200,200,200,200,200,200,200,200];
            alpBack(iSet,:) =
[0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0];
            alpNank(iSet,:) =
[0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0];
            Ztor_km(iSet,:) =
[10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10
,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10
,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10];
            SeatBasn(iSet,:) =
["0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_
NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoB
asin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin"
,"0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_
NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoB
asin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin"
,"0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_
NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoB
asin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin"
,"0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","1_InSeattleBasin","2_InOther
```

```
PNWbasin","2_InOtherPNWbasin","2_InOtherPNWbasin","2_InOtherPNWbasin","2_InOtherPNWbasin","2_InOtherPNWbasi
n","2_InOtherPNWbasin","2_InOtherPNWbasin","2_InOtherPNWbasin","2_InOtherPNWbasin","2_InOtherPNWbasin"];
            Z1pt0m(iSet,:) =
[550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,5
50,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550
,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,5
50,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550];
            Z2pt5m(iSet,:) =
[2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,
2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2
000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,20
00,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,200
0,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000];
            NsplEpi(iSet,:) =
[100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,1
00,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100
,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,1
00,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100];
            Mbreak(iSet,:) =
[7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,
7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7
.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,8.6,8,7.7,7.7,7.7,7.4,7.4,8.5,7.7,7.9,7.9,8.5,
8.6,7.1,7.1];
            Zhypo_km(iSet,:) =
[55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55
,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55
,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55];
            SatReg(iSet,:) =
["global","global","global","global","global","global","global","global","global","global","global","global","global","global","gl
obal","global","global","global","global","global","global","global","global","global","global","global","global","global","globa
l","global","global","global","global","global","global","global","global","global","global","global","global","global","global","
global","global","global","global","global","global","global","global","global","global","global","global","global","global","glo
bal","global","global","global","global","global","global","global","global","global","global","global","global","global","global
","global","global","global","global","global","global","global","global","global","global","global","global","global","global","g
lobal","global","Alaska","Aleutian","Cascadia","Cascadia","Cascadia","Central_America_N","Central_America_S","Japan_Pac
","Japan_Phi","New_Zealand_N","New_Zealand_S","South_America_N","South_America_S","Taiwan_E","Taiwan_W"];
            Basin(iSet,:) =
["0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_
NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoB
asin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin"
,"0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_
NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoB
asin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin"
,"0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_
NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoB
asin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin"
,"0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","1_InSeattleBasin","2_InOther
PNWbasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_No
Basin","0_NoBasin"];
            Nsigma(iSet,:) =
[1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1
,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1];
            InterIntra(iSet,:) =
["0_Interface","0_Interface","0_Interface","0_Interface","0_Interface","0_Interface","0_Interface","0_Interface","0_Interfac
e","0_Interface","0_Interface","0_Interface","0_Interface","0_Interface","0_Interface","0_Interface","0_Interface","0_Interf
ace","0_Interface","0_Interface","0_Interface","0_Interface","0_Interface","0_Interface","0_Interface","0_Interface","0_Int
erface","0_Interface","0_Interface","0_Interface","0_Interface","0_Interface","0_Interface","0_Interface","0_Interface","0_I
nterface","0_Interface","0_Interface","0_Interface","0_Interface","0_Interface","0_Interface","0_Interface","0_Interface","0
```

```
_Interface","0_Interface","0_Interface","0_Interface","0_Interface","0_Interface","0_Interface","0_Interface","0_Interface",
"0_Interface","0_Interface","0_Interface","0_Interface","0_Interface","0_Interface","0_Interface","0_Interface","0_Interfac
e","0_Interface","0_Interface","0_Interface","0_Interface","0_Interface","0_Interface","0_Interface","0_Interface","0_Interf
ace","0_Interface","0_Interface","0_Interface","0_Interface","0_Interface","0_Interface","0_Interface","0_Interface","0_Int
erface","0_Interface","0_Interface","0_Interface","0_Interface","0_Interface","0_Interface","0_Interface","0_Interface","0_I
nterface","0_Interface","0_Interface","0_Interface","0_Interface","0_Interface","0_Interface","0_Interface","0_Interface","0
_Interface","0_Interface","0_Interface","0_Interface","0_Interface","0_Interface"];
            T_sec(iSet,:) =
[0.01,0.02,0.03,0.05,0.075,0.1,0.15,0.2,0.25,0.3,0.4,0.5,0.75,1,1.5,2,3,4,5,7.5,10,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01
,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01
,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01
,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01];

        else; % intraslab

            CaseLabel(iSet,:) =
["Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=40,InterIntra=1_Intraslab,SeatBasn=0_
NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_se
c=0.01","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=40,InterIntra=1_Intraslab,SeatB
asn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma
=1,T_sec=0.02","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=40,InterIntra=1_Intrasla
b,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,
Nsigma=1,T_sec=0.03","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=40,InterIntra=1_
Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_N
oBasin,Nsigma=1,T_sec=0.05","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=40,Interl
ntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Ba
sin=0_NoBasin,Nsigma=1,T_sec=0.075","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km
=40,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=
global,Basin=0_NoBasin,Nsigma=1,T_sec=0.1","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Zt
or_km=40,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,S
atReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.15","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNa
nk=0,Ztor_km=40,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_k
m=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.2","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0
,alpNank=0,Ztor_km=40,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zh
ypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.25","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=200,alp
Back=0,alpNank=0,Ztor_km=40,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak
=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.3","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=2
00,alpBack=0,alpNank=0,Ztor_km=40,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,M
break=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.4","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_
km=200,alpBack=0,alpNank=0,Ztor_km=40,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=
100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.5","Mag=8,Vs30_mps=760,Reg=0_global
,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=40,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,Ns
plEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.75","Mag=8,Vs30_mps=760,Reg=0
_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=40,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2
000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=1","Mag=8,Vs30_mps=760,Re
g=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=40,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_
m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=1.5","Mag=8,Vs30_mps=
760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=40,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550
,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=2","Mag=8,Vs30_
mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=40,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_
m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=3","Mag=8,
Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=40,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,
Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=4","M
ag=8,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=40,InterIntra=1_Intraslab,SeatBasn=0_NoB
asin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=5"
,"Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=40,InterIntra=1_Intraslab,SeatBasn=0_
NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_se
c=7.5","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=40,InterIntra=1_Intraslab,SeatBa
```

sn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=10","Mag=4,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=4.25,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=4.5,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=4.75,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=5,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=5.25,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=5.5,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=5.75,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=6,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=6.25,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=6.5,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=6.75,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=7,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=7.25,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=7.5,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=7.75,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8.25,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8.5,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8.75,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=9,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=9.25,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=40,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=52,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=67.6,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=87.88,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=114.244,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=148.5172,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1

_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=193.07236,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=250.994068,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=326.2922884,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=424.179974919999,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=551.433967395999,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=716.864157614799,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=931.923404899238,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=1211.50042636901,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=1574.95055427971,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=2047.43572056362,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=2661.66643673271,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=3460.16636775252,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=4498.21627807827,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=5847.68116150175,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=7601.98550995228,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=9882.58116293796,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=150,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=169.5,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=191.535,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=216.43455,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=244.5710415,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=276.365276895,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=312.292762891 35,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=352.890822067225,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=398.766628935964,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=450.60629069764,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=509.185108488333,Reg=0_global,Rrup_km=200,a

lpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=575.379172591816,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=650.178465028752,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=734.701665482489,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=830.212881995213,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=938.140556654591,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=1060.09882901969,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=1197.91167679225,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=1353.64019477524,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=1529.61342009602,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=1728.4631647085,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=1953.16337612061,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=1_Alaska,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.2,Zhypo_km=55,SatReg=Alaska,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=1_Alaska,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=8,Zhypo_km=55,SatReg=Aleutian,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=2_Cascadia,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.2,Zhypo_km=55,SatReg=Cascadia,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=2_Cascadia,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=1_InSeattleBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.2,Zhypo_km=55,SatReg=Cascadia,Basin=1_InSeattleBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=2_Cascadia,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=2_InOtherPNWbasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.2,Zhypo_km=55,SatReg=Cascadia,Basin=2_InOtherPNWbasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=3_CentralAmerica&Mexico,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=2_InOtherPNWbasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.4,Zhypo_km=55,SatReg=Central_America_N,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=3_CentralAmerica&Mexico,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=2_InOtherPNWbasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=Central_America_S,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=4_Japan,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=2_InOtherPNWbasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=Japan_Pac,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=4_Japan,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=2_InOtherPNWbasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=Japan_Phi,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=5_NewZealand,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=2_InOtherPNWbasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=New_Zealand_N,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=5_NewZealand,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=2_InOtherPNWbasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=New_Zealand_S,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=6_SouthAmerica,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=2_InOtherPNWbasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.3,Zhypo_km=55,SatReg=South_America_N,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=6_SouthAmerica,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=2_InOtherPNWbasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.2,Zhypo_km=55,SatReg=South_America_S,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=7_Taiwan,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=2_InOtherPNWbasin,Z1.0_m=550,

```
Z2.5_m=2000,NsplEpi=100,Mbreak=7.7,Zhypo_km=55,SatReg=Taiwan_E,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,V
s30_mps=760,Reg=7_Taiwan,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=2_InOtherP
NWbasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.7,Zhypo_km=55,SatReg=Taiwan_W,Basin=0_NoBasin,Nsigma=1
,T_sec=0.01"];
            CaseLabelRegion(iSet,:) =
["0_GLB_GLB_Mb7.6","0_GLB_GLB_Mb7.6","0_GLB_GLB_Mb7.6","0_GLB_GLB_Mb7.6","0_GLB_GLB_Mb7.6","0_GLB_GLB_
Mb7.6","0_GLB_GLB_Mb7.6","0_GLB_GLB_Mb7.6","0_GLB_GLB_Mb7.6","0_GLB_GLB_Mb7.6","0_GLB_GLB_Mb7.6","0_GLB
_GLB_Mb7.6","0_GLB_GLB_Mb7.6","0_GLB_GLB_Mb7.6","0_GLB_GLB_Mb7.6","0_GLB_GLB_Mb7.6","0_GLB_GLB_Mb7.6","
0_GLB_GLB_Mb7.6","0_GLB_GLB_Mb7.6","0_GLB_GLB_Mb7.6","0_GLB_GLB_Mb7.6","0_GLB_GLB_Mb7.6","0_GLB_GLB_M
b7.6","0_GLB_GLB_Mb7.6","0_GLB_GLB_Mb7.6","0_GLB_GLB_Mb7.6","0_GLB_GLB_Mb7.6","0_GLB_GLB_Mb7.6","0_GLB_
GLB_Mb7.6","0_GLB_GLB_Mb7.6","0_GLB_GLB_Mb7.6","0_GLB_GLB_Mb7.6","0_GLB_GLB_Mb7.6","0_GLB_GLB_Mb7.6","0
_GLB_GLB_Mb7.6","0_GLB_GLB_Mb7.6","0_GLB_GLB_Mb7.6","0_GLB_GLB_Mb7.6","0_GLB_GLB_Mb7.6","0_GLB_GLB_Mb
7.6","0_GLB_GLB_Mb7.6","0_GLB_GLB_Mb7.6","0_GLB_GLB_Mb7.6","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_G
LB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_
GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7
.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GL
B_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.6","0_GLB_GLB_Mb7.6","0_GLB_GLB_Mb7.6","0_
GLB_GLB_Mb7.6","0_GLB_GLB_Mb7.6","0_GLB_GLB_Mb7.6","0_GLB_GLB_Mb7.6","0_GLB_GLB_Mb7.6","0_GLB_GLB_Mb7
.6","0_GLB_GLB_Mb7.6","0_GLB_GLB_Mb7.6","0_GLB_GLB_Mb7.6","0_GLB_GLB_Mb7.6","0_GLB_GLB_Mb7.6","0_GLB_GL
B_Mb7.6","0_GLB_GLB_Mb7.6","0_GLB_GLB_Mb7.6","0_GLB_GLB_Mb7.6","0_GLB_GLB_Mb7.6","0_GLB_GLB_Mb7.6","0_
GLB_GLB_Mb7.6","0_GLB_GLB_Mb7.6","0_GLB_GLB_Mb7.6","1_AK_AK_Mb7.2","1_AK_Aleu_Mb8","2_CSC_CSC_Mb7.2","2
_CSC_CSC_Mb7.2_1_InSeattleBasin","2_CSC_CSC_Mb7.2_2_InOtherPNWbasin","3_CAM_CAM_N_Mb7.4","3_CAM_CAM_S
_Mb7.6","4_JP_JP_Pac_Mb7.6","4_JP_JP_Phi_Mb7.6","5_NZ_NZ_N_Mb7.6","5_NZ_NZ_S_Mb7.6","6_SAM_SAM_N_Mb7.3"
,"6_SAM_SAM_S_Mb7.2","7_TW_TW_E_Mb7.7","7_TW_TW_W_Mb7.7"];

            Mag(iSet,:) =
[8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,4,4.25,4.5,4.75,5,5.25,5.5,5.75,6,6.25,6.5,6.75,7,7.25,7.5,7.75,8,8.25,8.5,8.75,9,9.25,
8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8];
            Vs30mps(iSet,:) =
[760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,7
60,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760
,760,760,760,760,760,760,760,760,150,169.5,191.535,216.43455,244.5710415,276.365276895,312.29276289135,352.8908
22067225,398.766628935964,450.60629069764,509.185108488333,575.379172591816,650.178465028752,734.701665482
489,830.212881995213,938.140556654591,1060.09882901969,1197.91167679225,1353.64019477524,1529.61342009602,
1728.4631647085,1953.16337612061,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760];
            Reg(iSet,:) =
["0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_global","
0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_
global","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_gl
obal","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_glo
bal","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_globa
l","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_global",
"0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0
_global","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_global","1_Alaska","1_
Alaska","2_Cascadia","2_Cascadia","2_Cascadia","3_CentralAmerica&Mexico","3_CentralAmerica&Mexico","4_Japan","4_Ja
pan","5_NewZealand","5_NewZealand","6_SouthAmerica","6_SouthAmerica","7_Taiwan","7_Taiwan"];
            Rrup_km(iSet,:) =
[200,200,200,200,200,200,200,200,200,200,200,200,200,200,200,200,200,200,200,200,200,200,200,200,200,200,200,200,200,200,2
00,200,200,200,200,200,200,200,200,200,200,200,200,200,200,200,200,40,52,67.6,87.88,114.244,148.5172,193.07236,250.994068,
326.2922884,424.179974919999,551.433967395999,716.864157614799,931.923404899238,1211.50042636901,1574.9505
5427971,2047.43572056362,2661.66643673271,3460.16636775252,4498.21627807827,5847.68116150175,7601.98550995
228,9882.58116293796,200,200,200,200,200,200,200,200,200,200,200,200,200,200,200,200,200,200,200,200,200,200,200,200,
200,200,200,200,200,200,200,200,200,200,200,200,200,200];
            alpBack(iSet,:) =
[0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0];
```

```
            alpNank(iSet,:) =
[0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0];
            Ztor_km(iSet,:) =
[40,40,40,40,40,40,40,40,40,40,40,40,40,40,40,40,40,40,40,40,40,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10
,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10
,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10];
            SeatBasn(iSet,:) =
["0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_
NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoB
asin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin"
,"0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_
NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoB
asin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin"
,"0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_
NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoB
asin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin"
,"0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","1_InSeattleBasin","2_InOther
PNWbasin","2_InOtherPNWbasin","2_InOtherPNWbasin","2_InOtherPNWbasin","2_InOtherPNWbasin","2_InOtherPNWbasi
n","2_InOtherPNWbasin","2_InOtherPNWbasin","2_InOtherPNWbasin","2_InOtherPNWbasin","2_InOtherPNWbasin"];
            Z1pt0m(iSet,:) =
[550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,5
50,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550
,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,5
50,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550];
            Z2pt5m(iSet,:) =
[2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,
2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2
000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,20
00,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,200
0,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000];
            NsplEpi(iSet,:) =
[100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,1
00,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100
,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,1
00,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100];
            Mbreak(iSet,:) =
[7.6,7.6,7.6,7.6,7.6,7.6,7.6,7.6,7.6,7.6,7.6,7.6,7.6,7.6,7.6,7.6,7.6,7.6,7.6,7.6,7.6,7.6,7.6,7.6,7.6,7.6,7.6,7.6,7.6,7.6,7.6,7.6,7.6,7.6,7.6,
7.6,7.6,7.6,7.6,7.6,7.6,7.6,7.6,7.6,7.6,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.6,7
.6,7.6,7.6,7.6,7.6,7.6,7.6,7.6,7.6,7.6,7.6,7.6,7.6,7.6,7.6,7.6,7.6,7.6,7.6,7.6,7.6,7.2,8,7.2,7.2,7.2,7.4,7.6,7.6,7.6,7.6,7.6,7.3,
7.2,7.7,7.7];
            Zhypo_km(iSet,:) =
[55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55
,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55
,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55];
            SatReg(iSet,:) =
["global","global","global","global","global","global","global","global","global","global","global","global","global","global","gl
obal","global","global","global","global","global","global","global","global","global","global","global","global","global","globa
l","global","global","global","global","global","global","global","global","global","global","global","global","global","global","
global","global","global","global","global","global","global","global","global","global","global","global","global","global","glo
bal","global","global","global","global","global","global","global","global","global","global","global","global","global","global
","global","global","global","global","global","global","global","global","global","global","global","global","global","global","g
lobal","global","Alaska","Aleutian","Cascadia","Cascadia","Cascadia","Central_America_N","Central_America_S","Japan_Pac
","Japan_Phi","New_Zealand_N","New_Zealand_S","South_America_N","South_America_S","Taiwan_E","Taiwan_W"];
            Basin(iSet,:) =
["0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_
NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoB
asin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin"
```

```
,"0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_
NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoB
asin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin"
,"0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_
NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoB
asin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin"
,"0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","1_InSeattleBasin","2_InOther
PNWbasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_No
Basin","0_NoBasin"];
            Nsigma(iSet,:) =
[1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1
,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1];
            InterIntra(iSet,:) =
["1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab"
,"1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab"
,"1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab"
,"1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab"
,"1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab"
,"1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab"
,"1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab"
,"1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab"
,"1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab"
,"1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab"
,"1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab"
,"1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab"];
            T_sec(iSet,:) =
[0.01,0.02,0.03,0.05,0.075,0.1,0.15,0.2,0.25,0.3,0.4,0.5,0.75,1,1.5,2,3,4,5,7.5,10,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01
,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01
,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01
,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01];

        end
        [junk,NN] = size(T_sec);
        iRowStart = 1;
        iRowEnd = NN;

        for irow = iRowStart:iRowEnd
           thisCaseLabel = CaseLabel(iSet,irow);
           thisMagnitude = Mag(iSet,irow);
           thisVs30 = Vs30mps(iSet,irow);
           thisRegion = Reg(iSet,irow);
           thisRrup = Rrup_km(iSet,irow);
           thisAlphaBackarc = alpBack(iSet,irow);
           thisAlphaNankai = alpNank(iSet,irow);
           thisZtor = Ztor_km(iSet,irow);
           thisSeatBasn = SeatBasn(iSet,irow);
           thisZ1pt0 = Z1pt0m(iSet,irow);
           thisZ2pt5 = Z2pt5m(iSet,irow);
           thisNsampleEpi = NsplEpi(iSet,irow);
           thisMb = Mbreak(iSet,irow);
           thisZhypo = Zhypo_km(iSet,irow);
           thisSubductionSlab = SatReg(iSet,irow);
           thisPNWbasinStrux = Basin(iSet,irow);
           thisNsigma = Nsigma(iSet,irow);
           thisEventType = InterIntra(iSet,irow);
           thisT = T_sec(iSet,irow);
```

```
        hereKBCG20MedianPSA = KBCG20_medPSA(thisT, thisMagnitude,thisRrup, thisAlphaBackarc, thisAlphaNankai,
thisZtor, thisEventType, thisVs30,thisZ1pt0,thisZ2pt5, thisMb,thisRegion,thisPNWbasinStrux);
        hereKBCG20Tau = KBCG20_sigmaTau(thisT);
        hereKBCG20Phi = KBCG20_sigmaPhi(thisT);
        hereKBCG20SigmaAleatory = KBCG20_sigmaAleatory(thisT);
        hereKBCG20SigmaEpistemic = KBCG20_SigmaEpistemic(thisT, thisMagnitude,thisRrup, thisAlphaBackarc,
thisAlphaNankai, thisZtor, thisEventType, thisVs30,thisZ1pt0,thisZ2pt5,
thisMb,thisRegion,thisNsampleEpi,thisPNWbasinStrux);
        hereKBCG20SigmaTotal = KBCG20_SigmaTotal(thisT, thisMagnitude,thisRrup, thisAlphaBackarc, thisAlphaNankai,
thisZtor, thisEventType, thisVs30,thisZ1pt0,thisZ2pt5, thisMb,thisRegion,thisNsampleEpi,thisPNWbasinStrux);
        hereKBCG20MedianMinusNsigmaxSigmaTotal = exp( log(hereKBCG20MedianPSA)-
thisNsigma*hereKBCG20SigmaTotal);
        hereKBCG20MedianPlusNsigmaxSigmaTotal = exp(
log(hereKBCG20MedianPSA)+thisNsigma*hereKBCG20SigmaTotal);
        hereKBCG20MedianMinusNsigmaxSigmaAleatory = exp( log(hereKBCG20MedianPSA)-
thisNsigma*hereKBCG20SigmaAleatory);
        hereKBCG20MedianPlusNsigmaxSigmaAleatory = exp(
log(hereKBCG20MedianPSA)+thisNsigma*hereKBCG20SigmaAleatory);
        hereKBCG20MedianMinusNsigmaxSigmaEpistemic = exp( log(hereKBCG20MedianPSA)-
thisNsigma*hereKBCG20SigmaEpistemic);
        hereKBCG20MedianPlusNsigmaxSigmaEpistemic = exp(
log(hereKBCG20MedianPSA)+thisNsigma*hereKBCG20SigmaEpistemic);
        herePSHAB20MedianPSA = PSHAB20_Median(thisEventType, thisRegion, thisSubductionSlab, thisRrup,
thisMagnitude, thisZhypo, thisT, thisVs30, thisZ2pt5, thisPNWbasinStrux);
        herePSHAB20Tau = PSHAB20_Tau(thisT, thisRrup, thisVs30);
        herePSHAB20PhiTot = PSHAB20_PhiTot(thisT, thisRrup, thisVs30);
        herePSHAB20PhiS2S = PSHAB20_SigmaS2S(thisT, thisRrup, thisVs30);
        herePSHAB20PhiSS = PSHAB20_SigmaSS(thisT, thisRrup, thisVs30);
        herePSHAB20SigmaAleatory = PSHAB20_SigmaAleatory(thisT, thisRrup, thisVs30);
        herePSHAB20SigmaEpistemic = PSHAB20_SigmaEpistemic(thisT, thisEventType, thisSubductionSlab);
        herePSHAB20SigmaTotal = PSHAB20_SigmaTotal(thisT, thisRrup,  thisVs30, thisEventType, thisSubductionSlab);
        herePSHAB20MedianMinusNsigmaxSigmaTotal = exp(log(herePSHAB20MedianPSA)-
thisNsigma*herePSHAB20SigmaTotal);
        herePSHAB20MedianPlusNsigmaxSigmaTotal =
exp(log(herePSHAB20MedianPSA)+thisNsigma*herePSHAB20SigmaTotal);
        herePSHAB20MedianMinusNsigmaxSigmaAleatory = exp(log(herePSHAB20MedianPSA)-
thisNsigma*herePSHAB20SigmaAleatory);
        herePSHAB20MedianPlusNsigmaxSigmaAleatory =
exp(log(herePSHAB20MedianPSA)+thisNsigma*herePSHAB20SigmaAleatory);
        herePSHAB20MedianMinusNsigmaxSigmaEpistemic = exp(log(herePSHAB20MedianPSA)-
thisNsigma*herePSHAB20SigmaEpistemic);
        herePSHAB20MedianPlusNsigmaxSigmaEpistemic =
exp(log(herePSHAB20MedianPSA)+thisNsigma*herePSHAB20SigmaEpistemic);
        outMatrixRows(irow + (iSet-1)*NN,:) =
[hereKBCG20MedianPSA,hereKBCG20Tau,hereKBCG20Phi,hereKBCG20SigmaAleatory,hereKBCG20SigmaEpistemic,hereKBC
G20SigmaTotal,hereKBCG20MedianMinusNsigmaxSigmaTotal,hereKBCG20MedianPlusNsigmaxSigmaTotal,hereKBCG20Medi
anMinusNsigmaxSigmaAleatory,hereKBCG20MedianPlusNsigmaxSigmaAleatory,hereKBCG20MedianMinusNsigmaxSigmaEpi
stemic,hereKBCG20MedianPlusNsigmaxSigmaEpistemic,herePSHAB20MedianPSA,herePSHAB20Tau,herePSHAB20PhiTot,her
ePSHAB20PhiS2S,herePSHAB20PhiSS,herePSHAB20SigmaAleatory,herePSHAB20SigmaEpistemic,herePSHAB20SigmaTotal,he
rePSHAB20MedianMinusNsigmaxSigmaTotal,herePSHAB20MedianPlusNsigmaxSigmaTotal,herePSHAB20MedianMinusNsig
maxSigmaAleatory,herePSHAB20MedianPlusNsigmaxSigmaAleatory,herePSHAB20MedianMinusNsigmaxSigmaEpistemic,her
ePSHAB20MedianPlusNsigmaxSigmaEpistemic];
      end;   % for irow
    end;   % for iset
    outDataTableStudy = array2table(outMatrixRows);
    outDataTableHeader =
{'KBCG20MedianPSA','KBCG20Tau','KBCG20Phi','KBCG20SigmaAleatory','KBCG20SigmaEpistemic','KBCG20SigmaTotal','KBCG
20MedianMinusNsigmaxSigmaTotal','KBCG20MedianPlusNsigmaxSigmaTotal','KBCG20MedianMinusNsigmaxSigmaAleatory',
```

```
'KBCG20MedianPlusNsigmaxSigmaAleatory','KBCG20MedianMinusNsigmaxSigmaEpistemic','KBCG20MedianPlusNsigmaxSigmaEpistemic','PSHAB20MedianPSA','PSHAB20Tau','PSHAB20PhiTot','PSHAB20PhiS2S','PSHAB20PhiSS','PSHAB20SigmaAleatory','PSHAB20SigmaEpistemic','PSHAB20SigmaTotal','PSHAB20MedianMinusNsigmaxSigmaTotal','PSHAB20MedianPlusNsigmaxSigmaTotal','PSHAB20MedianMinusNsigmaxSigmaAleatory','PSHAB20MedianPlusNsigmaxSigmaAleatory','PSHAB20MedianMinusNsigmaxSigmaEpistemic','PSHAB20MedianPlusNsigmaxSigmaEpistemic'};
      outDataTableStudy.Properties.VariableNames =outDataTableHeader;
      inDataTable =
table([CaseLabel(1,:),CaseLabel(2,:)].',[CaseLabelRegion(1,:),CaseLabelRegion(2,:)].',[Mag(1,:),Mag(2,:)].',[Vs30mps(1,:),Vs30mps(2,:)].',[Reg(1,:),Reg(2,:)].',[Rrup_km(1,:),Rrup_km(2,:)].',[alpBack(1,:),alpBack(2,:)].',[alpNank(1,:),alpNank(2,:)].',[Ztor_km(1,:),Ztor_km(2,:)].',[SeatBasn(1,:),SeatBasn(2,:)].',[Z1pt0m(1,:),Z1pt0m(2,:)].',[Z2pt5m(1,:),Z2pt5m(2,:)].',[NsplEpi(1,:),NsplEpi(2,:)].',[Mbreak(1,:),Mbreak(2,:)].',[Zhypo_km(1,:),Zhypo_km(2,:)].',[SatReg(1,:),SatReg(2,:)].',[Basin(1,:),Basin(2,:)].',[Nsigma(1,:),Nsigma(2,:)].',[InterIntra(1,:),InterIntra(2,:)].',[T_sec(1,:),T_sec(2,:)].');
      inDataTableHeader =
{'CaseLabel','CaseLabelRegion','Mag','Vs30mps','Reg','Rrup_km','alpBack','alpNank','Ztor_km','SeatBasn','Z1pt0m','Z2pt5m','NsplEpi','Mbreak','Zhypo_km','SatReg','Basin','Nsigma','InterIntra','T_sec'};
      inDataTable.Properties.VariableNames =inDataTableHeader;
      AllDataTableSensitivity = [inDataTable outDataTableStudy];
      disp(AllDataTableSensitivity)
      outputFilename = 'NGAsubduction_GMMtool_Matlab_Out_SensitivityStudy.csv';
      writetable(AllDataTableSensitivity,outputFilename,'Delimiter',',','QuoteStrings',true)
      disp("Sensitivity-Study Results have been saved to " + outputFilename);

   end
end
```

5.4 NGA-Sub Ground-Motion Characterization Tools – Python

NGASubduction_GMMtool_Python_ r201031.py

```
### ################################################################
### ################################################################
### ----------------------------------------------------------------
### ----------------------------------------------------------------
### -- NGA-Subduction:
###              Ground-Motion Characterization Tool
###              Python Version
### ----------------------------------------------------------------
###                        October 2020
### ----------------------------------------------------------------
### Ground-Motion Model Tool developed by
###          Silvia Mazzoni
###             smazzoni@ucla.edu
###          B. John Garrick Institute for the Risk Sciences
###             https://www.risksciences.ucla.edu/nhr3/gmtools
### ----------------------------------------------------------------
### SourceS:
###  KBCG20:
###    Kuehn, Bozorgnia, Campbell, and Gregor,
###        “Partially Nonergodic Ground-Motion Model for Subduction Regions using NGA-Subduction Database,”
###        Report 4/2020, Pacific Earthquake Engineering Research Center, UC Berkeley
###  PSHAB20:
###    Parker, G.A., Stewart, J.P., Hassani, B., Atkinson, G.M., and Boore, D.M. (2020).
###        "NGA-Subduction Global Ground Motion Models with Regional Adjustment Factors."
###        Report xx/2020, Pacific Earthquake Engineering Research Center, UC Berkeley
### ----------------------------------------------------------------
### Instruction:
###   1. Enter the values of the user-input parameters.
###      Input ranges are given with each input parameters.
###   2. Run full script
###   3. Wait while script runs
###   4. Result values are printed in Command Window (you may copy and
###   paste them elsewhere)
### ----------------------------------------------------------------
###   Notes:
###              This implementation was translated from an R script, where array indices start at 1
###         to minimize error, a place-holder was used at the zero index in this version
###      The Median and Sigma models are described in the reports.
###      Damping Ratio = 5#
###        You may speed up the program by managing the coefficients
###        tables more efficiently.
###      The program computes the geometric mean of the median ground-motion models,
###          and the SRSS of the aleatory variability and epistemic uncertainty.
###
### ----------------------------------------------------------------
###
### ################################################################
### ################################################################


# ################################################################
# ################################################################
# ################################################################
################ USER INPUT
# ################################################################
# ################################################################
```

```
# ############################################################


User_Region = '0_global';   # Region (Options: 0_global, 1_Alaska, 2_Cascadia, 3_CentralAmerica&Mexico, 4_Japan,
5_NewZealand (PSHAB20 uses global model), 6_SouthAmerica, 7_Taiwan)
User_SubductionSlab = 'global';   # Subducting Slab (used to compute Mb)
(Alaska,Aleutian,Cascadia,Central_America_N,Central_America_S,global,Japan_Pac,Japan_Phi,New_Zealand_N,New_Zealan
d_S,South_America_N,South_America_S,Taiwan_E,Taiwan_W)
User_Magnitude = 8;   # Moment Magnitude
User_Vs30 = 760;   # VS30 (m/sec)
User_Rrup = 200;   # Rupture Distance (km)
User_AlphaBackarc = 0;   # Fraction of Rrup in Backarc (Range: 0-1)
User_AlphaNankai = 0;   # Fraction of Rrup in Nankai Region (Range: 0-(1-AlphaBackArc))(Japan only)
User_Ztor = 10;   # Z_tor (km) (KBCG20 only)
User_Zhypo = 55;   # Hypocentral Depth (km) (PSHAB20 only)
User_EventType = '0_Interface';   # Event Type, (Options: 0_Interface, 1_Intraslab)
User_Z1pt0 = 550;   # Z1.0 input (m) (KBCG20 only)
User_Z2pt5 = 2000;   # Z2.5 input (m)
User_Mb = 7.9;   # Mb (KBCG20 only, can set = default)
User_PNWbasinStrux = '0_NoBasin';   # PNW Basin Structure (Options:
0_NoBasin,1_InSeattleBasin,2_InOtherPNWbasin)(Cascadia Only)
User_RelativeWeight_KBCG20 = 1;   # Relative Weight -- KBCG20
User_RelativeWeight_PSHAB20 = 0;   # Relative Weight -- PSHAB20
User_EpiInSigmaModels = '1_Yes';   # Apply KBCG20 Epistemic to Sigma Models (Options: 0_None, 1_AllModels)
User_NsampleEpi = 100;   # Number of Samples in Epistemic-Uncertainty Calculation (0=none, Range: 100-800)
User_Nsigma = 1;   # Number of Sigma away from Median


User_PeriodList = [0.01, 0.02, 0.03, 0.05, 0.075, 0.1, 0.15, 0.2, 0.25, 0.3, 0.4, 0.5, 0.75, 1, 1.5, 2, 3, 4, 5, 7.5,
            10, "PGA", "PGV"]  # range:0.01-10sec (PGA: T=0, PGV: T=-1)

# The following is an optional script that performs a large sensitivity study
# Running this sensitivity study make take a few minutes.
RunSensitivityStudyFlag = 'yes'  # options: 'yes' 'no'

#############################################################################################
#############################################################################################
#############################################################################################
# program
#  DO NOT edit lines below!!!
#  scroll to the bottom of the script
#############################################################################################
import math

import pandas as pd
import numpy as np
pd.set_option('display.max_rows', None)
#pd.set_option('display.max_columns', None)
desired_width=320
pd.set_option('display.width', desired_width)
pd.set_option('display.max_columns',60)


print(' ### -- NGA-Subduction: ')
print(' ###              Ground-Motion Characterization Tool')
print(' ###              smazzoni@ucla.edu')
print(' ###')
```

```
def getTFvalue(inValue):
   # CODE DEVELOPED/IMPLEMENTED BY
   #         Silvia Mazzoni, 2020
   #          smazzoni@ucla.edu

   # Code Written and developed by Silvia Mazzoni, smazzoni@ucla.edu, April 2020
   if type(inValue) == str:
      if inValue.lower() == "pga".lower():
         getTFvalueValue = 0
      elif inValue.lower() == "pgv".lower():
         getTFvalueValue = -1
   # end
   else:
      getTFvalueValue = inValue
   # end
   return getTFvalueValue


# end


# logistc hinge def
def loghinge(x, x0, a, b0, b1, delta):
   # CODE DEVELOPED/IMPLEMENTED BY
   #         Silvia Mazzoni, 2020
   #          smazzoni@ucla.edu
   loghingeValue = (a + b0 * (x - x0) + (b1 - b0) * delta * math.log(1 + math.exp((x - x0) / delta)))
   return loghingeValue


# end

# interpolation of adjustment to magnitude break point
def interp_dmb(period):
   # CODE DEVELOPED/IMPLEMENTED BY
   #         Silvia Mazzoni, 2020
   #          smazzoni@ucla.edu

   Aarray = [0.01, 1, 3, 10]
   Carray = [0, 0, -0.4, -0.4]
   if period == 0:
      interp_dmbValue = interpolateArray(0.01, Aarray, Carray, "log", "linear", "constant")
   elif period == -1:
      interp_dmbValue = 0
   else:
      interp_dmbValue = interpolateArray(period, Aarray, Carray, "log", "linear", "constant")
   # end
   return interp_dmbValue


# end


# interpolation of k1/k2 (values taken from Campbell and Bozorgnia (2014)
def interp_k1k2(period):
   # CODE DEVELOPED/IMPLEMENTED BY
   #         Silvia Mazzoni, 2020
```

```
    #         smazzoni@ucla.edu
    periods = [0.005, 0.01, 0.02, 0.03, 0.05, 0.075, 0.1, 0.15, 0.2, 0.25, 0.3, 0.4, 0.5, 0.75, 1., 1.5, 2., 3., 4., 5.,
               7.5, 10.]
    k1 = [865., 865., 865., 908., 1054., 1086., 1032., 878., 748., 654., 587., 503., 457., 410., 400., 400., 400., 400.,
          400., 400., 400., 400.]
    k2 = [-1.186, -1.186, -1.219, -1.273, -1.346, -1.471, -1.624, -1.931, -2.188, -2.381, -2.518, -2.657, -2.669,
          -2.401, -1.955, -1.025, -0.299, 0., 0., 0., 0., 0.]

    if period == 0:
        ap_k1 = interpolateArray(0.005, periods, k1, "log", "linear", "constant")
        ap_k2 = interpolateArray(0.005, periods, k2, "log", "linear", "constant")
    elif period == -1:
        ap_k1 = 400.0
        ap_k2 = -1.995
    else:
        ap_k1 = interpolateArray(period, periods, k1, "log", "linear", "constant")
        ap_k2 = interpolateArray(period, periods, k2, "log", "linear", "constant")
    # end
    interp_k1k2Value = [-999, ap_k1, ap_k2]  # inserted a spacer for zero-starting index
    return interp_k1k2Value


# end


# funcation to calculate Z_1/Z2pt5 from Vs30
def calc_z_from_Vs30(Vs30, coeffs):
    # CODE DEVELOPED/IMPLEMENTED BY
    #         Silvia Mazzoni, 2020
    #          smazzoni@ucla.edu

    zref = coeffs[1] + (coeffs[2] - coeffs[1]) * math.exp((math.log(Vs30) - coeffs[3]) / coeffs[4]) / (
            1 + math.exp((math.log(Vs30) - coeffs[3]) / coeffs[4]))

    calc_z_from_Vs30Value = zref
    return calc_z_from_Vs30Value


# end
def interpolateArray(Xpoint, Xlist, Ylist, XinterpType, YinterpType, extrapolateType, XinterpMin=-1e16,
                     XinterpMax=1e16):
    # CODE DEVELOPED/IMPLEMENTED BY
    #         Silvia Mazzoni, 2020
    #          smazzoni@ucla.edu

    newX = []
    interpolateArrayValue = -999

    irow = 0
    start0 = -1
    endrow0 = -1

    nrows = len(Xlist)

    for irow in range(0, nrows - 1 + 1):  # start-index 0
        this = Xlist[irow]
        if Xpoint == this:
            interpolateArrayValue = Ylist[irow]
```

```
        return interpolateArrayValue
    # end
    if this >= XinterpMin and this <= XinterpMax:
        if start0 < 0:
            start0 = irow
        # end
        end0 = irow
# end
# end
iend = irow
istart = start0
Xstart = Xlist[start0]
Xend = Xlist[end0]
if (Xend - Xpoint) * (Xstart - Xpoint) <= 0:
    for i in range(istart, iend - 1 + 1):
        thisx = Xlist[i]
        nextX = Xlist[i + 1]
        if (Xpoint - thisx) * (Xpoint - nextX) <= 0:
            l1 = i
            l2 = i + 1
            break
    # end
# end
else:
    if (Xend - Xstart) * (Xend - Xpoint) > 0:
        l1 = istart
        if LeftString(extrapolateType.lower(), len("extra")) == "extra".lower():
            l2 = istart + 1
        else:
            l2 = istart
    # end
    else:
        l1 = iend
        if LeftString(extrapolateType.lower(), len("extra")) == "extra".lower():
            l2 = iend - 1
        else:
            interpolateArrayValue = Ylist[end0]
            return interpolateArrayValue
    # end
# end
# end

x0 = Xlist[l1]
x1 = Xlist[l2]
y0 = Ylist[l1]
y1 = Ylist[l2]
hereX = Xpoint
if LeftString(XinterpType, len("log")).lower() == "log".lower():
    if x0 <= 0:
        x0 = 0.000000001
    # end
    if x1 <= 0:
        x1 = 0.000000002
    # end
    if hereX <= 0:
        hereX = 0.000000001
    # end
    x0 = math.log(x0)
```

```
      x1 = math.log(x1)
      hereX = math.log(hereX)
   # end
   if LeftString(YinterpType, len("log")).lower() == "log".lower():
      if y0 <= 0:
         y0 = 0.000000001
      # end
      if y1 <= 0:
         y1 = 0.000000001
      # end
      y0 = math.log(y0)
      y1 = math.log(y1)
   # end
   interpolateArrayValue = y0 + (y1 - y0) * (hereX - x0) / (x1 - x0)

   if LeftString(YinterpType, len("log")).lower() == "log".lower():
      interpolateArrayValue = math.exp(interpolateArrayValue)
   # end
   return interpolateArrayValue


# end


### def to calculate median prediction

# The following def calculates the median prediction of KBCG20, as described in chapter 4 of the PEER report.
# It takes as input the predictor variables, as well as a set of coefficients.
# Later, we define a def that acts as a wrapper around this def, and will take as input period, region index, and  select the
appropriate coefficients and pass them on.
#
# The inputs are

# `m`: moment magnitude
# `rlist`: a vector of length 3, which contains the distance in subregion 1,2,3 relative to volcanic arc `rlist = c(R1,R2,R3)`.
Typically, `R1=R3=0` (corresponding to forearc).
# `ztor`: depth to top of rupture in km.
# `EventType`: flag for interface (`EventType = 0`) and intraslab (`EventType = 1`). Must be 0 or 1.
# `Vs30`: $V_{S30}$ n m/s.
# `fx`: arc crossing flag. Must be 0 or 1.
# `delta_ln_z`: difference between natural log of observed Z1.0/Z2.5 value and reference Z1.0/Z2.5 from $V_{S30}$.
# `coeffs`: vector containing coefficients needed to calculate median prediction.
# `coeffs_attn`: vector of length 6 to calcualte anelastic attenuation.
# `mbreak` and `zbreak`: magnitude and depth scaling break point
# `k1` and `k2`: parameters needed for site amplification
# `nft1` and `nft2`: coefficients needed for pseudo-depth term.
# `pgarock`: median pga prediction at $V_{S30} = 1100$

# def to calculate median prediction

def KBCG20_med(M, Rrup, AlphaBackarc, AlphaNankai, Ztor, EventType, Vs30, delta_ln_z, coeffs, coeffs_attn, coeffs_z,
          mbreak, zbreak, k1, k2, nft1, nft2, pgarock, region):
   # CODE DEVELOPED/IMPLEMENTED BY
   #         Silvia Mazzoni, 2020
   #          smazzoni@ucla.edu
   #  x = "KBCG20_med"

   if LeftNumber(region, 1) == 1:
```

```
    distR1 = 0
    distR3 = 0
elif LeftNumber(region, 1) == 2:
    distR1 = 0
    distR3 = 0
elif LeftNumber(region, 1) == 3:
    distR1 = AlphaBackarc * Rrup
    distR3 = 0
elif LeftNumber(region, 1) == 4:
    distR1 = AlphaBackarc * Rrup
    distR3 = AlphaNankai * Rrup
elif LeftNumber(region, 1) == 5:
    distR1 = 0
    distR3 = 0
elif LeftNumber(region, 1) == 6:
    distR1 = AlphaBackarc * Rrup
    distR3 = 0
elif LeftNumber(region, 1) == 7:
    distR1 = 0
    distR3 = 0
else:  # reg=0
    distR1 = AlphaBackarc * Rrup
    distR3 = 0
# end

distR2 = Rrup - distR1 - distR3

theta10 = 0
vsrock = 1100
c = 1.88
n = 1.18
minmb = 6.
delta = 0.1
deltaz = 1
refzif = 15
refzslab = 50
thisFs = (LeftNumber(EventType, 1))

# check if cross arc:
fx = 1
if distR1 == 0 and distR2 == 0:
    fx = 0
# end
if distR2 == 0 and distR3 == 0:
    fx = 0
# end
if distR1 == 0 and distR3 == 0:
    fx = 0
# end

fmag = (1 - thisFs) * loghinge(M, mbreak, coeffs[6] * (mbreak - minmb), coeffs[6], coeffs[8],
                        delta) + thisFs * loghinge(M, mbreak, coeffs[7] * (mbreak - minmb), coeffs[7],
                                            coeffs[8], delta)
fgeom = (1 - thisFs) * (coeffs[3] + coeffs[5] * M) * math.log(Rrup + 10 ** (nft1 + nft2 * (M - 6)))
fgeom_slab = thisFs * (coeffs[4] + coeffs[5] * M) * math.log(Rrup + 10 ** (nft1 + nft2 * (M - 6)))
fdepth = (1 - thisFs) * loghinge(Ztor, zbreak, coeffs[12] * (zbreak - refzif), coeffs[12], theta10,
                          deltaz) + thisFs * loghinge(Ztor, zbreak, coeffs[13] * (zbreak - refzslab),
                                              coeffs[13], theta10, deltaz)
```

```
  DotProduct123 = distR1 * coeffs_attn[1] + distR2 * coeffs_attn[2] + distR3 * coeffs_attn[3]
  DotProduct456 = distR1 * coeffs_attn[4] + distR2 * coeffs_attn[5] + distR3 * coeffs_attn[6]
  fattn = fx * DotProduct123 + (1 - fx) * DotProduct456 + fx * coeffs[14]

  if Vs30 < k1:
    fsite = coeffs[11] * math.log(Vs30 / k1) + k2 * (
            math.log(pgarock + c * (Vs30 / k1) ** n) - math.log(pgarock + c))
  else:
    fsite = (coeffs[11] + k2 * n) * math.log(Vs30 / k1)
  # end

  fbasin = coeffs_z[1] + coeffs_z[2] * delta_ln_z

  Median = (1 - thisFs) * coeffs[1] + thisFs * coeffs[
    2] + fmag + fgeom + fgeom_slab + fdepth + fattn + fsite + fbasin
  return Median


# end


### def to calculate median prediction for a given scenario and period
# This is a def that calculates median predictions of KBCG20 for a given scenario.
# I takes as input period, the predictor variables for the scenarios, selects the appropriate coefficients/parameters, and calls the def defined in the previous section.
#
# The arguments are similar to the def before.

# `m`: moment magnitude
# `rlist`: a vector of length 3, which contains the distances in subregion 1,2,3 relative to volcanic arc `rlist = array(R1,R2,R3)`. for regions Alaska, Cascadia, New Zealand, and Taiwan, $R1 = R2 = 0$.
# `ztor`: depth to top of rupture in km.
# `EventType`: flag for interface (`EventType = 0`) and intraslab (`EventType = 1`). Must be 0 or 1.
# `Vs30`: $V_{S30}$ n m/s.
# `Z1pt0`: depth to a shear wave horizon of 1000 m/s, in m. Used for regions Alaska and New Zealand.
# `Z2pt5`: depth to a shear wave horizon of 2500 m/s, in m. Used for regions Cascadia and Japan.
# `fx`: arc crossing flag. Must be 0 or 1.
# `mb`: Magnitude scaling break point. Should be set regionally dependent based on Campbell (2020).

# The last input is a region index `region`, which is as follows:
#
# * 0: global
# * 1: Alaska
# * 2: Cascadia
# * 3: Central America & Mexico
# * 4: Japan
# * 5: New Zealand
# * 6: South America
# * 7: Taiwan
#
# The opional argumen `Seattle_Basin` is a flag that should be set to `TRUE` if the site is in the Seattle Basin, and `FALSE` otherwise.
# This flag determines which basin depth amplification model is used for Cascadia.
# It does not have an impact on any other region.
#
```

```
def KBCG20_medPSA(period, Magnitude, Rrup, AlphaBackarc, AlphaNankai, Ztor, EventType, Vs30, Z1pt0, Z2pt5, Mb,
            UserRegion, PNWbasinStrux):
   # CODE DEVELOPED/IMPLEMENTED BY
   #         Silvia Mazzoni, 2020
   #          smazzoni@ucla.edu

   thisOut = -999
   XinterpMin = 0
   XinterpMax = 10
   XinterpType = "log"
   YinterpType = "log"
   extrapolateType = "extrapolate"

   Parameters = pd.read_csv('NGAsubGMM_KBCG20_coefficients.csv')
   TvalueList = Parameters['T']
   InterpArray = interpolateFunction(period, TvalueList, XinterpType, extrapolateType, XinterpMin, XinterpMax)

   period0 = TvalueList[InterpArray[1]]
   y0 = KBCG20_medPSA_AtTlist(period0, Magnitude, Rrup, AlphaBackarc, AlphaNankai, Ztor, EventType, Vs30, Z1pt0, Z2pt5,
                     Mb, UserRegion, PNWbasinStrux)

   if InterpArray[2] <= 0:
      thisOut = y0
   else:
      period1 = TvalueList[InterpArray[2]]
      y1 = KBCG20_medPSA_AtTlist(period1, Magnitude, Rrup, AlphaBackarc, AlphaNankai, Ztor, EventType, Vs30, Z1pt0,
                        Z2pt5, Mb, UserRegion, PNWbasinStrux)
      if LeftString(YinterpType, len("log")).lower() == "log".lower():
         if y0 <= 0:
            y0 = 0.000000001
         # end
         if y1 <= 0:
            y1 = 0.000000001
         # end
         y0 = math.log(y0)
         y1 = math.log(y1)
      # end
      thisOut = y0 + (y1 - y0) * InterpArray[3]
      if LeftString(YinterpType, len("log")).lower() == "log".lower():
         thisOut = math.exp(thisOut)
   # end

   # end

   return thisOut


# end


# def to calculate median prediction using mean coefficients
def KBCG20_medPSA_AtTlist(period, M, Rrup, AlphaBackarc, AlphaNankai, Ztor, EventType, Vs30, Z1pt0, Z2pt5, Mb, region,
                  PNWbasinStrux):
   # CODE DEVELOPED/IMPLEMENTED BY
   #         Silvia Mazzoni, 2020
   #          smazzoni@ucla.edu
   KBCG20_medPSA_AtTlistValue = -999
   # need to add some checks for input (period, region)
```

```
# coefficients to calculate zref from vs30
pars_z_ja = [-999, 7.6893685375, 2.30258509299405, 6.3091864, 0.7528670225, 1.2952369625]
pars_z_casc = [-999, 8.29404964010203, 2.30258509299405, 6.39692965521615, 0.27081459, 1.7381352625]
pars_z_nz = [-999, 6.859789675, 2.30258509299405, 5.745692775, 0.91563524375, 1.03531412375]
pars_z_tw = [-999, 6.30560665, 2.30258509299405, 6.1104992125, 0.43671102, 0.7229702975]

Parameters = pd.read_csv('NGAsubGMM_KBCG20_coefficients.csv')
parameters_zmod = pd.read_csv('NGAsubGMM_KBCG20_params_Z_ALL_allregca_attn3_corrreg_cs_dmb.csv')

# calculate rock PGA
period_used = 0.
vsrock = 1100
pars_period = getRangeRowT(Parameters, period_used)
pars_period_zmod = getRangeRowT(parameters_zmod, period_used)  # silviamazzoni: I added this

coeffs = getSubArrayRange(pars_period, 2, 15)
k1k2 = interp_k1k2(period_used)
dmb = 0

delta_ln_z = 0
coeffs_z = [-999, 0, 0]
coeffs_z2 = [-999, 0, 0]

if LeftNumber(region, 1) == 1:
   coeffs = updateArray(coeffs, [1, 2, 11], pars_period, [17, 24, 31])
   coeffs_attn = makeArray(pars_period, [38, 45, 52, 59, 66, 73])
elif LeftNumber(region, 1) == 2:
   coeffs = updateArray(coeffs, [1, 2, 11], pars_period, [18, 25, 32])
   coeffs_attn = makeArray(pars_period, [39, 46, 53, 60, 67, 74])
   if LeftNumber(PNWbasinStrux, 1) == 1:
      coeff_seattle = getRangeValueT(parameters_zmod, period_used, "mean_residual_Seattle_basin")
      coeffs_z2 = [-999, coeff_seattle, 0]
   elif LeftNumber(PNWbasinStrux, 1) == 2:
      coeffs_z2 = makeArray(pars_period_zmod, [2, 3])
   # end
elif LeftNumber(region, 1) == 3:
   coeffs = updateArray(coeffs, [1, 2, 11], pars_period, [19, 26, 33])
   coeffs_attn = makeArray(pars_period, [40, 47, 54, 61, 68, 75])
elif LeftNumber(region, 1) == 4:
   dmb = interp_dmb(period_used)
   coeffs = updateArray(coeffs, [1, 2, 11], pars_period, [20, 27, 34])
   coeffs_attn = makeArray(pars_period, [41, 48, 55, 62, 69, 76])
   coeffs_z2 = makeArray(pars_period_zmod, [5, 6])
elif LeftNumber(region, 1) == 5:
   coeffs = updateArray(coeffs, [1, 2, 11], pars_period, [21, 28, 35])
   coeffs_attn = makeArray(pars_period, [42, 49, 56, 63, 70, 77])
   coeffs_z2 = makeArray(pars_period_zmod, [7, 8])
elif LeftNumber(region, 1) == 6:
   dmb = interp_dmb(period_used)
   coeffs = updateArray(coeffs, [1, 2, 11], pars_period, [22, 29, 36])
   coeffs_attn = makeArray(pars_period, [43, 50, 57, 64, 71, 78])
elif LeftNumber(region, 1) == 7:
   coeffs = updateArray(coeffs, [1, 2, 11], pars_period, [23, 30, 37])
   coeffs_attn = makeArray(pars_period, [44, 51, 58, 65, 72, 79])
   coeffs_z2 = makeArray(pars_period_zmod, [9, 10])
else:
   coeffs_attn = makeArray(pars_period, [11, 11, 11, 11, 10, 11])
   delta_ln_z = 0
```

```
   coeffs_z = [-999, 0, 0]
# end

delta_bz = makeArray(pars_period, [80, 81])
coeffs_nft = makeArray(pars_period, [82, 83])

thisFs = (LeftNumber(EventType, 1))

mbreak = (1 - thisFs) * (Mb + dmb) + thisFs * Mb
zbreak = (1 - thisFs) * (30 + delta_bz[1]) + thisFs * (80 + delta_bz[2])

mbreak_pga = mbreak
zbreak_pga = zbreak
k1k2_pga = k1k2
coeffs_pga = coeffs  # silviamazzoni: I added this so I alway have coeffs
coeffs_attn_pga = coeffs_attn
coeffs_z_pga = coeffs_z
coeffs_z_pga2 = coeffs_z2
coeffs_nft_pga = coeffs_nft
pgarock = math.exp(
   KBCG20_med(M, Rrup, AlphaBackarc, AlphaNankai, Ztor, EventType, vsrock, delta_ln_z, coeffs_pga, coeffs_attn_pga,
         coeffs_z_pga, mbreak_pga, zbreak_pga, k1k2_pga[1], k1k2_pga[2], coeffs_nft_pga[1], coeffs_nft_pga[2],
         0, region))

# calculate PSA
period_used = period
pars_period = getRangeRowT(Parameters, period_used)
pars_period_zmod = getRangeRowT(parameters_zmod, period_used)  # silviamazzoni: I added this
coeffs = getSubArrayRange(pars_period, 2, 15)
k1k2 = interp_k1k2(period_used)
dmb = 0

delta_ln_z = 0
coeffs_z = [-999, 0, 0]
if LeftNumber(region, 1) == 1:
   coeffs = updateArray(coeffs, [1, 2, 11], pars_period, [17, 24, 31])
   coeffs_attn = makeArray(pars_period, [38, 45, 52, 59, 66, 73])
elif LeftNumber(region, 1) == 2:
   coeffs = updateArray(coeffs, [1, 2, 11], pars_period, [18, 25, 32])
   coeffs_attn = makeArray(pars_period, [39, 46, 53, 60, 67, 74])
   if Z2pt5 > 0:
      delta_ln_z = math.log(Z2pt5) - calc_z_from_Vs30(Vs30, pars_z_casc)
   else:
      delta_ln_z = 0
   # end
   if LeftNumber(PNWbasinStrux, 1) == 1:
      coeff_seattle = getRangeValueT(parameters_zmod, period_used, "mean_residual_Seattle_basin")
      coeffs_z = [-999, coeff_seattle, 0]
   elif LeftNumber(PNWbasinStrux, 1) == 2:
      coeffs_z = makeArray(pars_period_zmod, [2, 3])
   # end
elif LeftNumber(region, 1) == 3:
   coeffs = updateArray(coeffs, [1, 2, 11], pars_period, [19, 26, 33])
   coeffs_attn = makeArray(pars_period, [40, 47, 54, 61, 68, 75])
elif LeftNumber(region, 1) == 4:
   dmb = interp_dmb(period_used)
   coeffs = updateArray(coeffs, [1, 2, 11], pars_period, [20, 27, 34])
   coeffs_attn = makeArray(pars_period, [41, 48, 55, 62, 69, 76])
```

```
      if Z2pt5 > 0:
         delta_ln_z = math.log(Z2pt5) - calc_z_from_Vs30(Vs30, pars_z_ja)
      else:
         delta_ln_z = 0
      # end
      coeffs_z = makeArray(pars_period_zmod, [5, 6])
   elif LeftNumber(region, 1) == 5:
      coeffs = updateArray(coeffs, [1, 2, 11], pars_period, [21, 28, 35])
      coeffs_attn = makeArray(pars_period, [42, 49, 56, 63, 70, 77])
      if Z1pt0 > 0:
         delta_ln_z = math.log(Z1pt0) - calc_z_from_Vs30(Vs30, pars_z_nz)
      else:
         delta_ln_z = 0
      # end
      coeffs_z = makeArray(pars_period_zmod, [7, 8])
   elif LeftNumber(region, 1) == 6:
      dmb = interp_dmb(period_used)
      coeffs = updateArray(coeffs, [1, 2, 11], pars_period, [22, 29, 36])
      coeffs_attn = makeArray(pars_period, [43, 50, 57, 64, 71, 78])
   elif LeftNumber(region, 1) == 7:
      coeffs = updateArray(coeffs, [1, 2, 11], pars_period, [23, 30, 37])
      coeffs_attn = makeArray(pars_period, [44, 51, 58, 65, 72, 79])
      if Z1pt0 > 0:
         delta_ln_z = math.log(Z1pt0) - calc_z_from_Vs30(Vs30, pars_z_tw)
      else:
         delta_ln_z = 0
      # end
      coeffs_z = makeArray(pars_period_zmod, [9, 10])
   else:
      coeffs_attn = makeArray(pars_period, [11, 11, 11, 11, 10, 11])
   # end

   delta_bz = makeArray(pars_period, [80, 81])
   coeffs_nft = makeArray(pars_period, [82, 83])

   thisFs = (LeftNumber(EventType, 1))
   mbreak = (1 - thisFs) * (Mb + dmb) + thisFs * Mb
   zbreak = (1 - thisFs) * (30 + delta_bz[1]) + thisFs * (80 + delta_bz[2])

   Med = KBCG20_med(M, Rrup, AlphaBackarc, AlphaNankai, Ztor, EventType, Vs30, delta_ln_z, coeffs, coeffs_attn,
               coeffs_z, mbreak, zbreak, k1k2[1], k1k2[2], coeffs_nft[1], coeffs_nft[2], pgarock, region)
   med_pga = KBCG20_med(M, Rrup, AlphaBackarc, AlphaNankai, Ztor, EventType, Vs30, delta_ln_z, coeffs_pga,
                  coeffs_attn_pga, coeffs_z_pga2, mbreak_pga, zbreak_pga, k1k2_pga[1], k1k2_pga[2],
                  coeffs_nft_pga[1], coeffs_nft_pga[2], pgarock, region)

   if Med < med_pga and period <= 0.1:
      Med = med_pga
   # end
   return math.exp(Med)


# end


def getRangeValue(thisRange, Xvalue, xHeader, yHeader):
   # CODE DEVELOPED/IMPLEMENTED BY
   #         Silvia Mazzoni, 2020
   #          smazzoni@ucla.edu
```

```
    # not sure how to make it work with xHeader as a variable
    getRangeRowValue = thisRange.loc[thisRange[xHeader] == Xvalue]
    getRangeValueValue = getRangeRowValue[yHeader]
    return getRangeValueValue.squeeze()


# end

def getRangeValueRegion(thisRange, Xvalue,  yHeader):
    # CODE DEVELOPED/IMPLEMENTED BY
    #         Silvia Mazzoni, 2020
    #          smazzoni@ucla.edu
    getRangeRowTValue = thisRange.loc[thisRange['Region'] == Xvalue]
    getRangeValueTValue = getRangeRowTValue[yHeader]
    return getRangeValueTValue.squeeze()
#end

def getRangeValueT(thisRange, Xvalue, yHeader):
    # CODE DEVELOPED/IMPLEMENTED BY
    #         Silvia Mazzoni, 2020
    #          smazzoni@ucla.edu
    getRangeRowTValue = thisRange.loc[thisRange['T'] == Xvalue]
    getRangeValueTValue = getRangeRowTValue[yHeader]
    return getRangeValueTValue.squeeze()


# end

def getRangeRowT(thisRange, Xvalue):
    # CODE DEVELOPED/IMPLEMENTED BY
    #         Silvia Mazzoni, 2020
    #          smazzoni@ucla.edu
    getRangeRowTValue = thisRange.loc[thisRange['T'] == Xvalue]
    this = getRangeRowTValue.values
    this = np.insert(this, 0, -999)
    out = this.flatten()
    return out


# end


def getRangeRowByIndex(thisRange, thisRowIndex):
    # CODE DEVELOPED/IMPLEMENTED BY
    #         Silvia Mazzoni, 2020
    #          smazzoni@ucla.edu
    getRangeRowTValue = thisRange.loc[thisRowIndex - 1]
    this = getRangeRowTValue.values
    this = np.insert(this, 0, -999)
    out = this.flatten()
    return out


# end

def getSubArrayRange(inArray, startIndex, endIndex):
    # CODE DEVELOPED/IMPLEMENTED BY
    #         Silvia Mazzoni, 2020
```

```
    #          smazzoni@ucla.edu
    temP = [-999]  # insert something at the zero start index spot
    for thisIndex in range(startIndex, endIndex + 1):
        # should offset
        temP.append(inArray[thisIndex])
    # end
    getSubArrayRangeValue = temP
    return getSubArrayRangeValue


# end


def LeftString(instring, ncars):
    LeftStringValue = instring[0:ncars]
    return LeftStringValue


# end
def LeftNumber(instring, ncars):
    instring = str(instring)
    LeftNumberValue = int(instring[0:ncars])
    return LeftNumberValue


# end


def makeArray(oldArray, oldArrayIndices):
    # CODE DEVELOPED/IMPLEMENTED BY
    #         Silvia Mazzoni, 2020
    #          smazzoni@ucla.edu

    nrow = len(oldArrayIndices)
    nrowOldArray = len(oldArrayIndices)
    newArray = [-999]  # insert spacer for zero start indexing
    nrowMax = nrow
    for irow in range(len(oldArrayIndices)):  # zero-start array
        oldIndex = oldArrayIndices[irow]
        newArray.append(oldArray[oldIndex])
    # end
    makeArrayValue = newArray
    return makeArrayValue


# end

def updateArray(newArray, newArrayIndices, oldArray, oldArrayIndices):
    # CODE DEVELOPED/IMPLEMENTED BY
    #         Silvia Mazzoni, 2020
    #          smazzoni@ucla.edu

    NN = len(oldArrayIndices)
    for irow in range(0, NN - 1 + 1):  # zero-start index
        oldIndex = oldArrayIndices[irow]
        newIndex = newArrayIndices[irow]

        newArray[newIndex] = oldArray[oldIndex]  # zero-start index not here as I have added spacers to the arrays
```

```
    # end
    updateArrayValue = newArray
    return updateArrayValue


# end


def interpolateFunction(Xpoint, Xlist, XinterpType, extrapolateType, XinterpMin=-1e16, XinterpMax=+1e16):
    # CODE DEVELOPED/IMPLEMENTED BY
    #         Silvia Mazzoni, 2020
    #          smazzoni@ucla.edu

    interpolateFunctionValue = -999
    irow = 0
    start0 = -1
    endrow0 = -1
    NN = len(Xlist)
    for irow in range(0, NN):
        this = Xlist[irow]
        if Xpoint == this:
            interpolateFunctionValue = [-999, irow, -1, 0]  # I have added a value at the zero-index location
            return interpolateFunctionValue
        # end
        if this >= XinterpMin and this <= XinterpMax:
            if start0 < 0:
                start0 = irow
            # end
            end0 = irow
    # end
    # end
    iend = irow - 1

    istart = start0
    Xstart = Xlist[start0]
    Xend = Xlist[end0]

    if (Xend - Xpoint) * (Xstart - Xpoint) <= 0:
        for i in range(istart, iend - 1 + 1):
            thisx = Xlist[i, 1]
            nextX = Xlist[i + 1, 1]
            if (Xpoint - thisx) * (Xpoint - nextX) <= 0:
                l1 = i
                l2 = i + 1
                break
        # end
    # end
    else:
        if (Xend - Xstart) * (Xend - Xpoint) > 0:
            l1 = istart
            if LeftString(extrapolateType.lower(), len("extra")).lower() == "extra".lower():
                l2 = istart + 1
            else:
                l2 = istart
        # end
        else:
            l1 = iend
            if LeftString(extrapolateType.lower(), len("extra")) == "extra".lower():
```

```
            l2 = iend - 1
         else:
            interpolateFunctionValue = [-999, end0, -1, 0]
            return interpolateFunctionValue
      # end
   # end
   # end

   # Interp:
   x0 = Xlist[l1]
   x1 = Xlist[l2]

   hereX = Xpoint
   if LeftString(XinterpType, len("log")).lower() == "log".lower():
      if x0 <= 0:
         x0 = 0.000000001
      # end
      if x1 <= 0:
         x1 = 0.000000002
      # end
      if hereX <= 0:
         hereX = 0.000000001
      # end
      x0 = math.log(x0)
      x1 = math.log(x1)
      hereX = math.log(hereX)
   # end

   interpolateFunctionValue = [-999, l1, l2, (hereX - x0) / (x1 - x0)]
   return interpolateFunctionValue


# end


def KBCG20_posteriorAtTlist(period, M, Rrup, AlphaBackarc, AlphaNankai, Ztor, EventType, Vs30, Z1pt0, Z2pt5, Mb, region,
                    num_samples=100, PNWbasinStrux=0):
   # CODE DEVELOPED/IMPLEMENTED BY
   #         Silvia Mazzoni, 2020
   #          smazzoni@ucla.edu
   # need to add some checks for input (period, region)

   KBCG20_posteriorAtTlistValue = -999
   # coefficients to calculate zref from vs30
   pars_z_ja = [-999, 7.6893685375, 2.30258509299405, 6.3091864, 0.7528670225, 1.2952369625]
   pars_z_casc = [-999, 8.29404964010203, 2.30258509299405, 6.39692965521615, 0.27081459, 1.7381352625]
   pars_z_nz = [-999, 6.859789675, 2.30258509299405, 5.745692775, 0.91563524375, 1.03531412375]
   pars_z_tw = [-999, 6.30560665, 2.30258509299405, 6.1104992125, 0.43671102, 0.7229702975]

   Parameters = pd.read_csv('NGAsubGMM_KBCG20_coefficients.csv')
   parameters_zmod = pd.read_csv("NGAsubGMM_KBCG20_params_Z_ALL_allregca_attn3_corrreg_cs_dmb.csv")

   # calculate rock PGA
   thisPeriod = 0.
   vsrock = 1100

   pars_period = getRangeRowT(Parameters, thisPeriod)
```

```
pars_period_zmod = getRangeRowT(parameters_zmod, thisPeriod)  # silviamazzoni, I added this
coeffs = getSubArrayRange(pars_period, 2, 15)
k1k2 = interp_k1k2(thisPeriod)
dmb = 0

delta_ln_z = 0
coeffs_z = [-999, 0, 0]
if LeftNumber(region, 1) == 1:
   coeffs = updateArray(coeffs, [1, 2, 11], pars_period, [17, 24, 31])
   coeffs_attn = makeArray(pars_period, [38, 45, 52, 59, 66, 73])
elif LeftNumber(region, 1) == 2:
   coeffs = updateArray(coeffs, [1, 2, 11], pars_period, [18, 25, 32])
   coeffs_attn = makeArray(pars_period, [39, 46, 53, 60, 67, 74])
elif LeftNumber(region, 1) == 3:
   coeffs = updateArray(coeffs, [1, 2, 11], pars_period, [19, 26, 33])
   coeffs_attn = makeArray(pars_period, [40, 47, 54, 61, 68, 75])
elif LeftNumber(region, 1) == 4:
   dmb = interp_dmb(thisPeriod)
   coeffs = updateArray(coeffs, [1, 2, 11], pars_period, [20, 27, 34])
   coeffs_attn = makeArray(pars_period, [41, 48, 55, 62, 69, 76])
elif LeftNumber(region, 1) == 5:
   coeffs = updateArray(coeffs, [1, 2, 11], pars_period, [21, 28, 35])
   coeffs_attn = makeArray(pars_period, [42, 49, 56, 63, 70, 77])
elif LeftNumber(region, 1) == 6:
   dmb = interp_dmb(thisPeriod)
   coeffs = updateArray(coeffs, [1, 2, 11], pars_period, [22, 29, 36])
   coeffs_attn = makeArray(pars_period, [43, 50, 57, 64, 71, 78])
elif LeftNumber(region, 1) == 7:
   coeffs = updateArray(coeffs, [1, 2, 11], pars_period, [23, 30, 37])
   coeffs_attn = makeArray(pars_period, [44, 51, 58, 65, 72, 79])
else:
   coeffs_attn = makeArray(pars_period, [11, 11, 11, 11, 10, 11])
# end

delta_bz = makeArray(pars_period, [80, 81])
coeffs_nft = makeArray(pars_period, [82, 83])
thisFs = (LeftNumber(EventType, 1))
mbreak = (1 - thisFs) * (Mb + dmb) + thisFs * Mb

zbreak = (1 - thisFs) * (30 + delta_bz[1]) + thisFs * (80 + delta_bz[2])

pgarock = math.exp(
   KBCG20_med(M, Rrup, AlphaBackarc, AlphaNankai, Ztor, EventType, vsrock, delta_ln_z, coeffs, coeffs_attn,
         coeffs_z, mbreak, zbreak, k1k2[1], k1k2[2], coeffs_nft[1], coeffs_nft[2], 0, region))

# calculate PSA
thisPeriod = period
Tnumeric = [-1, 0, 0.01, 0.02, 0.03, 0.05, 0.075, 0.1, 0.15, 0.2, 0.25, 0.3, 0.4, 0.5, 0.75, 1, 1.5, 2, 3, 4, 5,
        7.5, 10]
Tstring = ["T-1.00", "T00.00", "T00.01", "T00.02", "T00.03", "T00.05", "T00.07", "T00.10", "T00.15", "T00.20",
       "T00.25", "T00.30", "T00.40", "T00.50", "T00.75", "T01.00", "T01.50", "T02.00", "T03.00", "T04.00",
       "T05.00", "T07.50", "T10.00"]

thisIndex = Tnumeric.index(thisPeriod)
thisTstring = Tstring[thisIndex]

parameters_posterior = pd.read_csv('posterior_coefficients_KBCG20_' + thisTstring + '.csv')
```

```
pars_period_zmod = getRangeRowT(parameters_zmod, thisPeriod)  # silviamazzoni, I added this

coeffs = getSubArrayRange(pars_period, 2, 15)
k1k2 = interp_k1k2(thisPeriod)
dmb = 0
med_predictions = []
for k in range(1, num_samples + 1):
   pars_period = getRangeRowByIndex(parameters_posterior, k)

   coeffs = getSubArrayRange(pars_period, 2, 15)

   delta_ln_z = 0
   coeffs_z = [-999, 0, 0]
   if LeftNumber(region, 1) == 1:
      coeffs = updateArray(coeffs, [1, 2, 11], pars_period, [17, 24, 31])
      coeffs_attn = makeArray(pars_period, [38, 45, 52, 59, 66, 73])
   elif LeftNumber(region, 1) == 2:
      coeffs = updateArray(coeffs, [1, 2, 11], pars_period, [18, 25, 32])
      coeffs_attn = makeArray(pars_period, [39, 46, 53, 60, 67, 74])
      if Z2pt5 > 0:
         delta_ln_z = math.log(Z2pt5) - calc_z_from_Vs30(Vs30, pars_z_casc)
      else:
         delta_ln_z = 0
      # end
      if LeftNumber(PNWbasinStrux, 1) == 1:
         coeff_seattle = getRangeValueT(parameters_zmod, thisPeriod, "mean_residual_Seattle_basin")
         coeffs_z = [coeff_seattle, 0]
      elif LeftNumber(PNWbasinStrux, 1) == 2:
         coeffs_z = makeArray(pars_period_zmod, [2, 3])
      # end
   elif LeftNumber(region, 1) == 3:
      coeffs = updateArray(coeffs, [1, 2, 11], pars_period, [19, 26, 33])
      coeffs_attn = makeArray(pars_period, [40, 47, 54, 61, 68, 75])
   elif LeftNumber(region, 1) == 4:
      dmb = interp_dmb(thisPeriod)
      coeffs = updateArray(coeffs, [1, 2, 11], pars_period, [20, 27, 34])
      coeffs_attn = makeArray(pars_period, [41, 48, 55, 62, 69, 76])
      if Z2pt5 > 0:
         delta_ln_z = math.log(Z2pt5) - calc_z_from_Vs30(Vs30, pars_z_ja)
      else:
         delta_ln_z = 0
      # end
      coeffs_z = makeArray(pars_period_zmod, [5, 6])
   elif LeftNumber(region, 1) == 5:
      coeffs = updateArray(coeffs, [1, 2, 11], pars_period, [21, 28, 35])
      coeffs_attn = makeArray(pars_period, [42, 49, 56, 63, 70, 77])
      if Z1pt0 > 0:
         delta_ln_z = math.log(Z1pt0) - calc_z_from_Vs30(Vs30, pars_z_nz)
      else:
         delta_ln_z = 0
      # end
      coeffs_z = makeArray(pars_period_zmod, [7, 8])
   elif LeftNumber(region, 1) == 6:
      dmb = interp_dmb(thisPeriod)
      coeffs = updateArray(coeffs, [1, 2, 11], pars_period, [22, 29, 36])
      coeffs_attn = makeArray(pars_period, [43, 50, 57, 64, 71, 78])
   elif LeftNumber(region, 1) == 7:
      coeffs = updateArray(coeffs, [1, 2, 11], pars_period, [23, 30, 37])
```

```
        coeffs_attn = makeArray(pars_period, [44, 51, 58, 65, 72, 79])
        if Z1pt0 >0:
           delta_ln_z = math.log(Z1pt0) - calc_z_from_Vs30(Vs30, pars_z_tw)
        else:
           delta_ln_z = 0
        # end
        coeffs_z = makeArray(pars_period_zmod, [9, 10])
     else:  # reg=0
        coeffs = updateArray(coeffs, [1, 2, 11], pars_period, [172, 173, 174])
        coeffs_attn = makeArray(pars_period, [175, 176, 177, 178, 179, 180])
     # end

     delta_bz = makeArray(pars_period, [80, 81])
     coeffs_nft = makeArray(pars_period, [82, 83])
     thisFs = (LeftNumber(EventType, 1))

     mbreak = (1 - thisFs) * (Mb + dmb) + thisFs * Mb

     zbreak = (1 - thisFs) * (30 + delta_bz[1]) + thisFs * (80 + delta_bz[2])

     Med = KBCG20_med(M, Rrup, AlphaBackarc, AlphaNankai, Ztor, EventType, Vs30, delta_ln_z, coeffs, coeffs_attn,
                 coeffs_z, mbreak, zbreak, k1k2[1], k1k2[2], coeffs_nft[1], coeffs_nft[2], pgarock, region)
     med_predictions.append(Med)
  # end

  KBCG20_posteriorAtTlist_All = [-999]
  KBCG20_posteriorAtTlist_All.append(getMeanOfArray(med_predictions))
  KBCG20_posteriorAtTlist_All.append(getMedianOfArray(med_predictions))
  KBCG20_posteriorAtTlist_All.append(getStdDev(med_predictions))
  for i in range(1, len(med_predictions)):
     KBCG20_posteriorAtTlist_All.append(med_predictions[i])
  # end
  KBCG20_posteriorAtTlistValue = KBCG20_posteriorAtTlist_All
  return KBCG20_posteriorAtTlistValue


# end


def getMeanOfArray(inArray):
  # CODE DEVELOPED/IMPLEMENTED BY
  #         Silvia Mazzoni, 2020
  #          smazzoni@ucla.edu

  getMeanOfArrayValue = 0
  iCount = 0

  for ii in range(len(inArray)):
     getMeanOfArrayValue = getMeanOfArrayValue + inArray[ii]
     iCount = iCount + 1
  # end
  getMeanOfArrayValue = getMeanOfArrayValue / iCount
  return getMeanOfArrayValue


# end

def getMedianOfArray(inArray):
```

```
    # CODE DEVELOPED/IMPLEMENTED BY
    #         Silvia Mazzoni, 2020
    #          smazzoni@ucla.edu

    inArray = BubbleSrt(inArray, "True")
    num_samples = len(inArray)
    if round(num_samples / 2, 0) == num_samples / 2:
        getMedianOfArrayValue = 0.5 * (inArray[int(num_samples / 2)] + inArray[int(num_samples / 2 + 1)])
    else:
        getMedianOfArrayValue = inArray[int((num_samples + 1) / 2)]
    # end
    return getMedianOfArrayValue


# end

# https://www.mrexcel.com/board/threads/standard-deviation-of-an-array.206375/
# ************************************************************************
# *             Standard Deviation  of a 1D array                        *
# ************************************************************************
def getStdDev(Arr):
    # CODE DEVELOPED/IMPLEMENTED BY
    #         Silvia Mazzoni, 2020
    #          smazzoni@ucla.edu

    k1 = 1
    k2 = len(Arr)

    n = 0
    avg = getMeanOfArray(Arr)
    SumSq = 0
    for i in range(k2):
        n = n + 1
        SumSq = SumSq + (Arr[i] - avg) ** 2
    # end

    getStdDevValue = math.sqrt(SumSq / (n - 1))
    return getStdDevValue


# end


# https://stackoverflow.com/questions/11504418/excel-vba-quickest-way-to-sort-an-array-of-numbers-in-descending-order#
# Omit plngLeft & plngRight they are used internally during recursion

def BubbleSrt(ArrayIn, Ascending):
    # CODE DEVELOPED/IMPLEMENTED BY
    #         Silvia Mazzoni, 2020
    #          smazzoni@ucla.edu

    if Ascending == "True":
        for i in range(len(ArrayIn)):
            for j in range(i + 1, len(ArrayIn)):
                if ArrayIn[i] > ArrayIn[j]:
                    SrtTemp = ArrayIn[j]
                    ArrayIn[j] = ArrayIn[i]
```

```
                ArrayIn[i] = SrtTemp
            # end
        # end
    # end
    else:
        for i in range(len(ArrayIn)):
            for j in range(i + 1, len(ArrayIn)):
                if ArrayIn[i] < ArrayIn[j]:
                    SrtTemp = ArrayIn[j]
                    ArrayIn[j] = ArrayIn[i]
                    ArrayIn[i] = SrtTemp
                # end
            # end
    # end
    # end

    return ArrayIn


# end

def KBCG20_posterior(period, M, Rrup, AlphaBackarc, AlphaNankai, Ztor, EventType, Vs30, Z1pt0, Z2pt5, Mb, region,
                num_samples=100, PNWbasinStrux=0):
    # CODE DEVELOPED/IMPLEMENTED BY
    #         Silvia Mazzoni, 2020
    #          smazzoni@ucla.edu

    thisOut = -999
    XinterpMin = 0
    XinterpMax = 10
    XinterpType = "log"
    YinterpType = "linear"  # KBCG20_posteriorAtTlist values are logs already
    extrapolateType = "extrapolate"

    Parameters = pd.read_csv('NGAsubGMM_KBCG20_coefficients.csv')
    TvalueList = Parameters['T']
    InterpArray = interpolateFunction(period, TvalueList, XinterpType, extrapolateType, XinterpMin, XinterpMax)

    period0 = TvalueList[InterpArray[1]]
    KBCG20_posteriorAtTlist0 = KBCG20_posteriorAtTlist(period0, M, Rrup, AlphaBackarc, AlphaNankai, Ztor, EventType,
                                    Vs30, Z1pt0, Z2pt5, Mb, region, num_samples, PNWbasinStrux)

    med_predictions = []
    y0 = InterpArray[1]
    if InterpArray[2] <= 0:
        KBCG20_posteriorValue = KBCG20_posteriorAtTlist0
        return KBCG20_posteriorValue
    else:
        period1 = TvalueList[InterpArray[2]]
        KBCG20_posteriorAtTlist1 = KBCG20_posteriorAtTlist(period1, M, Rrup, AlphaBackarc, AlphaNankai, Ztor, EventType,
                                      Vs30, Z1pt0, Z2pt5, Mb, region, num_samples, PNWbasinStrux)
        for iCase in range(1, len(KBCG20_posteriorAtTlist0)):
            y0 = KBCG20_posteriorAtTlist0[iCase]
            y1 = KBCG20_posteriorAtTlist1[iCase]

            if LeftString(YinterpType, len("log")).lower() == "log".lower():
                if y0 <= 0:
                    y0 = 0.000000001
```

```
            # end
            if y1 <= 0:
               y1 = 0.000000001
            # end
            y0 = math.log(y0)
            y1 = math.log(y1)
         # end
         y0 + (y1 - y0) * InterpArray[3]
         med_predictions[iCase] = y0 + (y1 - y0) * InterpArray[3]
         if LeftString(YinterpType, len("log")).lower() == "log".lower():
            med_predictions[iCase] = math.exp(med_predictions[iCase])
      # end
   # end

   # end

   return med_predictions


# end


# def to calculate aliatory sigma


def KBCG20_sigmaAleatory(period):
   # CODE DEVELOPED/IMPLEMENTED BY
   #         Silvia Mazzoni, 2020
   #          smazzoni@ucla.edu

   thisOut = -999
   XinterpMin = 0
   XinterpMax = 10
   XinterpType = "log"
   YinterpType = "linear"
   extrapolateType = "extrapolate"

   Parameters = pd.read_csv('NGAsubGMM_KBCG20_coefficients.csv')
   TvalueList = Parameters['T']
   InterpArray = interpolateFunction(period, TvalueList, XinterpType, extrapolateType, XinterpMin, XinterpMax)

   period0 = TvalueList[InterpArray[1]]
   y0 = KBCG20_sigmaAleatoryAtTlist(period0)

   if InterpArray[2] <= 0:
      thisOut = y0
   else:
      period1 = TvalueList[InterpArray[2], 1]
      y1 = KBCG20_sigmaAleatoryAtTlist(period1)
      if LeftString(YinterpType, len("log")).lower() == "log".lower():
         if y0 <= 0:
            y0 = 0.000000001
         # end
         if y1 <= 0:
            y1 = 0.000000001
         # end
         y0 = math.log(y0)
         y1 = math.log(y1)
```

```
      # end
      thisOut = y0 + (y1 - y0) * InterpArray[3]
      if LeftString(YinterpType, len("log")).lower() == "log".lower():
         thisOut = math.exp(thisOut)
   # end

   # end

   return thisOut


# end


def KBCG20_sigmaAleatoryAtTlist(period):
   # CODE DEVELOPED/IMPLEMENTED BY
   #         Silvia Mazzoni, 2020
   #          smazzoni@ucla.edu

   Parameters = pd.read_csv('NGAsubGMM_KBCG20_coefficients.csv')
   # calculate PSA
   period_used = period
   pars_period = getRangeRowT(Parameters, period_used)

   Phi = pars_period[84]
   Tau = pars_period[85]
   KBCG20_sigmaAleatoryAtTlistValue = math.sqrt(Phi * Phi + Tau * Tau)
   return KBCG20_sigmaAleatoryAtTlistValue


# end

def KBCG20_sigmaPhi(period):
   # CODE DEVELOPED/IMPLEMENTED BY
   #         Silvia Mazzoni, 2020
   #          smazzoni@ucla.edu

   Parameters = pd.read_csv('NGAsubGMM_KBCG20_coefficients.csv')

   # calculate PSA
   period_used = period
   pars_period = getRangeRowT(Parameters, period_used)

   Phi = pars_period[84]
   KBCG20_sigmaPhiValue = Phi
   return KBCG20_sigmaPhiValue


# end

def KBCG20_sigmaTau(period):
   # CODE DEVELOPED/IMPLEMENTED BY
   #         Silvia Mazzoni, 2020
   #          smazzoni@ucla.edu

   Parameters = pd.read_csv('NGAsubGMM_KBCG20_coefficients.csv')

   # calculate PSA
```

```
    period_used = getTFvalue(period)
    pars_period = getRangeRowT(Parameters, period_used)
    Tau = pars_period[85]
    KBCG20_sigmaTauValue = Tau
    return KBCG20_sigmaTauValue


# end


def KBCG20_SigmaEpistemic(period, M, Rrup, AlphaBackarc, AlphaNankai, Ztor, EventType, Vs30, Z1pt0, Z2pt5, Mb, region,
                          num_samples=100, PNWbasinStrux=0):
    # CODE DEVELOPED/IMPLEMENTED BY
    #         Silvia Mazzoni, 2020
    #          smazzoni@ucla.edu

    if num_samples == 0:
        KBCG20_SigmaEpistemicValue = 0
    elif num_samples < 50:
        KBCG20_SigmaEpistemicValue = "not enough samples, please select a value =0 or between 50 and 800"
    elif num_samples <= 800:
        thisKBCG20_posterior = KBCG20_posterior(period, M, Rrup, AlphaBackarc, AlphaNankai, Ztor, EventType, Vs30,
                                    Z1pt0, Z2pt5, Mb, region, num_samples=100, PNWbasinStrux=0)
        KBCG20_SigmaEpistemicValue = thisKBCG20_posterior[3]
    else:
        KBCG20_SigmaEpistemicValue = "Please enter a value =0 or between 50 and 800"
    # end
    return KBCG20_SigmaEpistemicValue


# end

def KBCG20_SigmaTotal(period, M, Rrup, AlphaBackarc, AlphaNankai, Ztor, EventType, Vs30, Z1pt0, Z2pt5, Mb, region,
                      num_samples=100, PNWbasinStrux=0):
    # CODE DEVELOPED/IMPLEMENTED BY
    #         Silvia Mazzoni, 2020
    #          smazzoni@ucla.edu
    SigmaAleatory = KBCG20_sigmaAleatory(period)
    if num_samples == 0:
        SigmaEpistemic = 0.
    else:
        SigmaEpistemic = KBCG20_SigmaEpistemic(period, M, Rrup, AlphaBackarc, AlphaNankai, Ztor, EventType, Vs30, Z1pt0,
                                    Z2pt5, Mb, region, num_samples=100, PNWbasinStrux=0)
    # end
    SigmaTotalValue = math.sqrt(SigmaAleatory * SigmaAleatory + SigmaEpistemic * SigmaEpistemic)
    return SigmaTotalValue


# end


######################################################################

# updated 6/1/2020 by Silvia Mazzoni

# GMM_at_760_slab_v4.R

##Grace Parker
```

```
##Modified February 26 to expand comments
##Modified March 25, 2020 to call coefficients from master table

## Input Parameters ----------------------------------------------------------

##Event type: 0 == interface, 1 == slab

##region corresponds to options in the DatabaseRegion column of the flatfile, plus global. Must be a string. If no matches,
default will be global model:
## "global", "Alaska", "Cascadia", "CAM", "Japan", "SA" or "Taiwan"

##Saturation Region corresponds to regions defined by R. Archuleta and C. Ji:
## "global", "Aleutian","Alaska","Cascadia","Central_America_S", "Central_America_N",
"Japan_Pac","Japan_Phi","South_America_N","South_America_S", "Taiwan_W","Taiwan_E"

## Rrup is number in kilometers

##Hypocentral depth in km. To use Ztor value to estimate hypocentral depth, see Ch. 4.3.3 of Parker et al. PEER report

## period can be: (-1,0,0.01,0.02,0.025,0.03,0.04,0.05,0.075,0.1,0.15,0.2,0.25,0.3,0.4,0.5,0.75,1,1.5,2,2.5,3,4,5,7.5,10)
## where -1 == PGV and 0 == PGA

## Other pertinent information ----------------------------------------------
##Coefficient files must be in the active working directory
## This function has no site term. Can only estimate ground motion at the reference condition VS30 = 760m/s
## The output is the desired median model prediction in LN units
## Take the exponential to get PGA, PSA in g or the PGV in cm/s

##Function to compute GMM predictions at 760m/s for slab and interface

def PSHAB20_GMM_at_760_Slab(EventType, UserRegion, saturation_region_user, Rrup, M, hypocentral_depth, period):
   # CODE DEVELOPED/IMPLEMENTED BY
   #         Silvia Mazzoni, 2020
   #          smazzoni@ucla.edu

   PSHAB20_GMM_at_760_SlabValue = -999
   if (LeftNumber(EventType, 1)) == 0:
      return PSHAB20_GMM_at_760_SlabValue  # ("This def is only for slab")
   # end
   region = getPSHABregion(UserRegion)
   saturation_region = getPSHABSubductionRegion(saturation_region_user)

   ##import Coefficients

   ##  Import Master Coefficient Table ----------------------------------------
   #  CoefficientsTable =
readtable('PSHAB20_Table_E2_Slab_Coefficients_OneRowHeader.csv','PreserveVariableNames',True)
   CoefficientsTable = pd.read_csv('PSHAB20_Table_E2_Slab_Coefficients_OneRowHeader.csv')

   ##Define mb based on Archuleta and Ji (2019)
   Mb = getMbDefault("PSHAB20", EventType, UserRegion, saturation_region)

   ##  Constant ---------------------------------------------------------------
   #  #Isolate constant
   if region == "global":
      c0 = getRangeValueT(CoefficientsTable, period, "Global_c0")
   elif region == "Alaska" or region == "SA":
      c0 = getRangeValueT(CoefficientsTable, period, saturation_region + "_c0")
```

```
    else:
        c0 = getRangeValueT(CoefficientsTable, period, region + "_c0")
    # end
    # silviamazzoni, I added this:
    if checkEmptyNA999(c0):
        c0 = getRangeValueT(CoefficientsTable, period, "Global_c0")
    # end
    #  # Path Term ------------------------------------------------------------

    #  #near-source saturation
    if M <= Mb:
        littleM = (Log10(35) - Log10(3.12)) / (Mb - 4)
        h = 10 ** (littleM * (M - Mb) + Log10(35))
    else:
        h = 35
    # end
    Rref = math.sqrt(1 + h ** 2)
    r = math.sqrt(Rrup ** 2 + h ** 2)
    LogR = math.log(r)
    R_Rref = math.log(r / Rref)

    #  #Need  to isolate regional anelastic coefficient, a0
    if region == "global":
        a0 = getRangeValueT(CoefficientsTable, period, "Global_a0")
    else:
        a0 = getRangeValueT(CoefficientsTable, period, region + "_a0")
    # end
    if checkEmptyNA999(a0):
        a0 = getRangeValueT(CoefficientsTable, period, "Global_a0")
    # end

    c1 = getRangeValueT(CoefficientsTable, period, "c1")
    b4 = getRangeValueT(CoefficientsTable, period, "b4")
    Fp = (c1 * LogR + (b4 * M) * R_Rref + a0 * r)

    #  # Magnitude Scaling ----------------------------------------------------
    c4 = getRangeValueT(CoefficientsTable, period, "c4")
    c5 = getRangeValueT(CoefficientsTable, period, "c5")
    c6 = getRangeValueT(CoefficientsTable, period, "c6")
    Fm = c4 * func1(M, Mb) + c6 * func2(M, Mb) + c5 * func3(M, Mb)

    #  # Source Depth Scaling -------------------------------------------------
    Db = getRangeValueT(CoefficientsTable, period, "db_km")
    d = getRangeValueT(CoefficientsTable, period, "d")
    littleM = getRangeValueT(CoefficientsTable, period, 'm')
    if (hypocentral_depth >= Db):
        Fd = d
    elif (hypocentral_depth <= 20):
        Fd = littleM * (20 - Db) + d
    else:
        Fd = littleM * (hypocentral_depth - Db) + d
    # end

    mu = c0 + Fp + Fm + Fd
    return mu


# end
```

```
def func1(M, Mb):
   # CODE DEVELOPED/IMPLEMENTED BY
   #         Silvia Mazzoni, 2020
   #          smazzoni@ucla.edu
   if M <= Mb:
      func1value = M - Mb
   else:
      func1value = 0
   # end
   return func1value


# end
def func2(M, Mb):
   # CODE DEVELOPED/IMPLEMENTED BY
   #         Silvia Mazzoni, 2020
   #          smazzoni@ucla.edu
   if M > Mb:
      func2value = M - Mb
   else:
      func2value = 0
   # end
   return func2value


# end
def func3(M, Mb):
   # CODE DEVELOPED/IMPLEMENTED BY
   #         Silvia Mazzoni, 2020
   #          smazzoni@ucla.edu
   if M <= Mb:
      func3value = (M - Mb) ** 2
   else:
      func3value = 0
   # end
   return func3value


# end


def getPSHABregion(UserRegion):
   # ' CODE DEVELOPED/IMPLEMENTED BY
   # '         Silvia Mazzoni, 2020
   # '          smazzoni@ucla.edu

   RegionIDList = ["0_global", "1_Alaska", "2_Cascadia", "3_CentralAmerica&Mexico", "4_Japan", "5_NewZealand",
                   "6_SouthAmerica", "7_Taiwan"]
   PSHAB_RegionIDList = ["global", "Alaska", "Cascadia", "CAM", "Japan", "global", "SA", "Taiwan"]
   thisIndex = RegionIDList.index(UserRegion)
   getPSHABregionValue = PSHAB_RegionIDList[thisIndex]

   return getPSHABregionValue


# end
```

```
def getPSHABSubductionRegion(UserSubductionRegion):
   # ' CODE DEVELOPED/IMPLEMENTED BY
   # '         Silvia Mazzoni, 2020
   # '          smazzoni@ucla.edu

   SubdRegionIDList = ["Alaska", "Aleutian", "Cascadia", "CAM_N", "CAM_S", "global", "Japan_Pac", "Japan_Phi", "NZ_N",
                  "NZ_S", "SA_N", "SA_S", "Taiwan_E", "Taiwan_W", "Alaska", "Aleutian", "Cascadia",
                  "Central_America_N", "Central_America_S", "global", "Japan_Pac", "Japan_Phi", "New_Zealand_N",
                  "New_Zealand_S", "South_America_N", "South_America_S", "Taiwan_E", "Taiwan_W"]
   PSHAB_SubdRegionIDList = ["Alaska", "Aleutian", "Cascadia", "CAM_N", "CAM_S", "global", "Japan_Pac", "Japan_Phi",
                     "New_Zealand_N", "New_Zealand_S", "SA_N", "SA_S", "Taiwan_E", "Taiwan_W", "Alaska",
                     "Aleutian", "Cascadia", "CAM_N", "CAM_S", "global", "Japan_Pac", "Japan_Phi",
                     "New_Zealand_N", "New_Zealand_S", "SA_N", "SA_S", "Taiwan_E", "Taiwan_W"]
   thisIndex = SubdRegionIDList.index(UserSubductionRegion)
   getPSHABsubRegionValue = PSHAB_SubdRegionIDList[thisIndex]

   return getPSHABsubRegionValue


# end


def getMbDefault(NGAsubModelLabel, EventType, UserRegion, saturation_region_user):
   # CODE DEVELOPED/IMPLEMENTED BY
   #         Silvia Mazzoni, 2020
   #          smazzoni@ucla.edu

   # Silvia Mazzoni: I inserted the global model and replaced the NA values with those of the global model
   if (LeftNumber(EventType, 1)) == 1 or (LeftNumber(EventType, 1)) == 5:
      if NGAsubModelLabel == "KBCG20":
         # MbKBCG20Intraslab
         saturation_region_here = saturation_region_user
         saturation_regions_SBZ = ["global", "Aleutian", "Alaska", "Cascadia", "Central_America_N",
                           "Central_America_S", "Japan_Pac", "Japan_Phi", "New_Zealand_N", "New_Zealand_S",
                           "South_America_N", "South_America_S", "Taiwan_W", "Taiwan_E"]
         thisMbList = [7.6, 8, 7.2, 7.2, 7.4, 7.6, 7.6, 7.6, 7.6, 7.6, 7.3, 7.2, 7.7, 7.7]
         User_regions_SBZ = ["0_global", "1_Alaska", "1_Alaska", "2_Cascadia", "3_CentralAmerica&Mexico",
                        "3_CentralAmerica&Mexico", "4_Japan", "4_Japan", "5_NewZealand", "5_NewZealand",
                        "6_SouthAmerica", "6_SouthAmerica", "7_Taiwan", "7_Taiwan"]
      else:
         # MbPSHAB20Intraslab
         saturation_region_here = getPSHABSubductionRegion(saturation_region_user)
         Mb_global_Slab = 7.6
         saturation_regions_SBZ = ["global", "Aleutian", "Alaska", "-999", "Cascadia", "CAM_S", "CAM_N", "Japan_Pac",
                           "Japan_Phi", "New_Zealand_N", "New_Zealand_S", "SA_N", "SA_S", "Taiwan_W",
                           "Taiwan_E"]
         thisMbList = [Mb_global_Slab, 7.98, 7.2, Mb_global_Slab, 7.2, 7.6, 7.4, 7.65, 7.55, Mb_global_Slab,
                  Mb_global_Slab, 7.3, 7.25, 7.7, 7.7]
         User_regions_SBZ = ["0_global", "1_Alaska", "1_Alaska", "0_global", "2_Cascadia", "3_CentralAmerica&Mexico",
                        "3_CentralAmerica&Mexico", "4_Japan", "4_Japan", "5_NewZealand", "5_NewZealand",
                        "6_SouthAmerica", "6_SouthAmerica", "7_Taiwan", "7_Taiwan"]
   # end

   else:
      if NGAsubModelLabel == "KBCG20":
         # MbKBCG20Interface
         saturation_region_here = saturation_region_user
```

```
        saturation_regions_SBZ = ["global", "Aleutian", "Alaska", "Cascadia", "Central_America_N",
                          "Central_America_S", "Japan_Pac", "Japan_Phi", "New_Zealand_N", "New_Zealand_S",
                          "South_America_N", "South_America_S", "Taiwan_W", "Taiwan_E"]
        thisMbList = [7.9, 8, 8.6, 8, 7.4, 7.5, 8.5, 7.7, 8.3, 8, 8.5, 8.6, 7.1, 7.1]
        User_regions_SBZ = ["0_global", "1_Alaska", "1_Alaska", "2_Cascadia", "3_CentralAmerica&Mexico",
                    "3_CentralAmerica&Mexico", "4_Japan", "4_Japan", "5_NewZealand", "5_NewZealand",
                    "6_SouthAmerica", "6_SouthAmerica", "7_Taiwan", "7_Taiwan"]
      else:
        # MbPSHAB20Interface
        saturation_region_here = getPSHABSubductionRegion(saturation_region_user)
        Mb_global_IF = 7.9
        saturation_regions_SBZ = ["global", "Aleutian", "Alaska", "-999", "Cascadia", "CAM_S", "CAM_N", "Japan_Pac",
                          "Japan_Phi", "New_Zealand_N", "New_Zealand_S", "SA_N", "SA_S", "Taiwan_W",
                          "Taiwan_E"]
        thisMbList = [Mb_global_IF, 8, 8.6, Mb_global_IF, 7.7, 7.4, 7.4, 8.5, 7.7, Mb_global_IF, Mb_global_IF, 8.5,
                8.6, 7.1, 7.1]
        User_regions_SBZ = ["0_global", "1_Alaska", "1_Alaska", "0_global", "2_Cascadia", "3_CentralAmerica&Mexico",
                    "3_CentralAmerica&Mexico", "4_Japan", "4_Japan", "5_NewZealand", "5_NewZealand",
                    "6_SouthAmerica", "6_SouthAmerica", "7_Taiwan", "7_Taiwan"]
    # end

    # end

    thisSaturationRegionDbRegion = getArrayMap(saturation_region_here, saturation_regions_SBZ, User_regions_SBZ)
    if thisSaturationRegionDbRegion.lower() != UserRegion.lower():
      saturation_region = "global"
    # end

    Mb = getArrayMap(saturation_region_here, saturation_regions_SBZ, thisMbList)

    if checkEmptyNA999(Mb):
      Mb = getArrayMap("global", saturation_regions_SBZ, thisMbList)
    # end

    getMbDefaultValue = Mb

    return getMbDefaultValue


# end


def getArrayMap(Xvalue, Xarray, Yarray):
    # CODE DEVELOPED/IMPLEMENTED BY
    #         Silvia Mazzoni, 2020
    #          smazzoni@ucla.edu
    getArrayMapValue = -999
    for (thisX, thisY) in zip(Xarray, Yarray):
      if thisX == Xvalue:
        getArrayMapValue = thisY
    # end
    # end
    return getArrayMapValue


# end
```

```
def checkEmptyNA999(thisvalue):
   checkEmptyNA999value = False
   if str(thisvalue) == "NA":
      checkEmptyNA999value = True
   elif str(thisvalue) == "":
      checkEmptyNA999value = True
   elif str(thisvalue) == "nan":
      checkEmptyNA999value = True
   # end
   return checkEmptyNA999value


# end


# GMM_at_VS30_slab_v4.R

##Grace Parker
##Modified February 26 to expand comments
##Modified March 25, 2020 to call consolidated coefficient table

## Input Parameters ----------------------------------------------------------

##Event type: 0 == interface, 1 == slab

##region corresponds to options in the DatabaseRegion column of the flatfile, plus global. Must be a string. If no matches, default will be global model:
## "global", "Alaska", "Cascadia", "CAM", "Japan", "SA" or "Taiwan"

##Saturation Region corresponds to regions defined by R. Archuleta and C. Ji:
## "global", "Aleutian","Alaska","Cascadia","Central_America_S", "Central_America_N",
"Japan_Pac","Japan_Phi","South_America_N","South_America_S", "Taiwan_W","Taiwan_E"

## Rrup is number in kilometers

##Hypocentral depth in km. To use Ztor value to estimate hypocentral depth, see Ch. XXX of Parker et al. PEER report

## period can be: (-1,0,0.01,0.02,0.025,0.03,0.04,0.05,0.075,0.1,0.15,0.2,0.25,0.3,0.4,0.5,0.75,1,1.5,2,2.5,3,4,5,7.5,10)
## where -1 == PGV and 0 == PGA

##VS30 in units m/s

##Z2.5 in units m. Only used if DatabaseRegion == "Japan" or "Cascadia". Can also specify "default" to get no basin term

##basin is only used if DatabaseRegion == "Cascadia". Value can be 0, 1, or 2, where 0 == having an estimate of Z2.5 outside mapped basin, 1 == Seattle basin, and 0 == other mapped basin (Tacoma, Everett, Georgia, etc.)

## Other pertinent information ----------------------------------------------
##Coefficient files must be in the active working directory
## "GMM_at_VS30_Slab_v2.R" calls function "GMM_at_760_Slab_v2.R" to compute PGAr in the nonlinear site term. This function must be in the R environment else: an error will occur.
## The output is the desired median model prediction in LN units
## Take the exponential to get PGA, PSA in g or  PGV in cm/s


##Function to compute GMM predictions at various VS30s for slab
```

```
def PSHAB20_GMM_at_VS30_Slab(EventType, UserRegion, saturation_region_user, Rrup, M, hypocentral_depth, period,
Vs30,
                        Z2pt5, basin):
   # CODE DEVELOPED/IMPLEMENTED BY
   #         Silvia Mazzoni, 2020
   #          smazzoni@ucla.edu

   PSHAB20_GMM_at_VS30_SlabValue = -999
   if (LeftNumber(EventType, 1)) == 0:
      return PSHAB20_GMM_at_VS30_SlabValue  # ("This def is only for slab")
   # end
   region = getPSHABregion(UserRegion)
   saturation_region = getPSHABSubductionRegion(saturation_region_user)

   basin = LeftNumber(basin, 1)
   ##import coefficients
   #  CoefficientsTable =
readtable('PSHAB20_Table_E2_Slab_Coefficients_OneRowHeader.csv','PreserveVariableNames',True)
   CoefficientsTable = pd.read_csv('PSHAB20_Table_E2_Slab_Coefficients_OneRowHeader.csv')

   # Define mb based on Archuleta and Ji (2019)
   Mb = getMbDefault("PSHAB20", EventType, UserRegion, saturation_region)

   # # Constant ----------------------------------------------------------------
   #  #Isolate constant

   if region == "global":
      c0 = getRangeValueT(CoefficientsTable, period, "Global_c0")
   elif region == "Alaska" or region == "SA":
      c0 = getRangeValueT(CoefficientsTable, period, saturation_region + "_c0")
   else:
      c0 = getRangeValueT(CoefficientsTable, period, region + "_c0")
   # end
   if checkEmptyNA999(c0):
      c0 = getRangeValueT(CoefficientsTable, period, "Global_c0")
   # end

   ##  Path Term ---------------------------------------------------------------

   #  #near-source saturation
   if M <= Mb:
      littleM = (Log10(35) - Log10(3.12)) / (Mb - 4)
      h = 10 ** (littleM * (M - Mb) + Log10(35))
   else:
      h = 35
   # end

   Rref = math.sqrt(1 + h ** 2)
   r = math.sqrt(Rrup ** 2 + h ** 2)
   LogR = math.log(r)
   R_Rref = math.log(r / Rref)

   #  #Need  to isolate regional anelastic coefficient, a0
   if region == "global":
      a0 = getRangeValueT(CoefficientsTable, period, "Global_a0")
   else:
      a0 = getRangeValueT(CoefficientsTable, period, region + "_a0")
   # end
```

```
if checkEmptyNA999(a0):
   a0 = getRangeValueT(CoefficientsTable, period, "Global_a0")
# end

c1 = getRangeValueT(CoefficientsTable, period, "c1")
b4 = getRangeValueT(CoefficientsTable, period, "b4")
Fp = (c1 * LogR + (b4 * M) * R_Rref + a0 * r)

##  Magnitude Scaling -------------------------------------------------------

c4 = getRangeValueT(CoefficientsTable, period, "c4")
c5 = getRangeValueT(CoefficientsTable, period, "c5")
c6 = getRangeValueT(CoefficientsTable, period, "c6")

Fm = c4 * func1(M, Mb) + c6 * func2(M, Mb) + c5 * func3(M, Mb)

##  Source Depth Scaling ----------------------------------------------------
Db = getRangeValueT(CoefficientsTable, period, "db_km")
d = getRangeValueT(CoefficientsTable, period, "d")
littleM = getRangeValueT(CoefficientsTable, period, "m")
##compute depth scaling term

if (hypocentral_depth >= Db):
   Fd = d
elif (hypocentral_depth <= 20):
   Fd = littleM * (20 - Db) + d
else:
   Fd = littleM * (hypocentral_depth - Db) + d
# end

##  Linear Site Amplification -----------------------------------------------

## Site Coefficients
V1 = getRangeValueT(CoefficientsTable, period, "V1_m_s")
V2 = getRangeValueT(CoefficientsTable, period, "V2_m_s")
Vref = getRangeValueT(CoefficientsTable, period, "Vref_m_s")

if region == "global" or region == "CAM":
   s2 = getRangeValueT(CoefficientsTable, period, "Global_s2")
   s1 = s2
elif region == "Taiwan" or region == "Japan":
   s2 = getRangeValueT(CoefficientsTable, period, region + "_s2")
   s1 = getRangeValueT(CoefficientsTable, period, region + "_s1")
else:
   s2 = getRangeValueT(CoefficientsTable, period, region + "_s2")
   s1 = s2
# end

##Compute linear site term
if Vs30 <= V1:
   Flin = s1 * math.log(Vs30 / V1) + s2 * math.log(V1 / Vref)
elif Vs30 <= V2:
   Flin = s2 * math.log(Vs30 / Vref)
else:
   Flin = s2 * math.log(V2 / Vref)
# end

# # Nonlinear Site Term -----------------------------------------------------
```

```
   PGAr = math.exp(PSHAB20_GMM_at_760_Slab(EventType, UserRegion, saturation_region, Rrup, M, hypocentral_depth,
0))
   f3 = 0.05
   Vb = 200
   Vref_Fnl = 760

   if period >= 3:
      Fnl = 0
   else:
      f4 = getRangeValueT(CoefficientsTable, period, "f4")
      f5 = getRangeValueT(CoefficientsTable, period, "f5")
      f2 = f4 * (math.exp(f5 * (min(Vs30, Vref_Fnl) - Vb)) - math.exp(f5 * (Vref_Fnl - Vb)))
      Fnl = 0 + f2 * math.log((PGAr + f3) / f3)
   # end

   ##  Basin Term ----------------------------------------------------------------
   #    Z2pt5
   #    region
   if str(Z2pt5) == "default" or str(Z2pt5) == "Default":
      Fb = 0
   elif Z2pt5 <= 0 or (region != "Japan" and region != "Cascadia"):
      Fb = 0
   else:
      if region == "Cascadia":
         theta0 = 3.94
         theta1 = -0.42
         vmu = 200
         vsig = 0.2
         e1 = getRangeValueT(CoefficientsTable, period, "C_e1")

         C_e3 = getRangeValueT(CoefficientsTable, period, "C_e3")
         C_e2 = getRangeValueT(CoefficientsTable, period, "C_e2")

         if basin == 0:
            del_None = getRangeValueT(CoefficientsTable, period, "del_None")
            e3 = C_e3 + del_None
            e2 = C_e2 + del_None
         elif basin == 1:
            del_Seattle = getRangeValueT(CoefficientsTable, period, "del_Seattle")
            e3 = C_e3 + del_Seattle
            e2 = C_e2 + del_Seattle
         else:
            e3 = C_e3
            e2 = C_e2
         # end

      elif region == "Japan":

         theta0 = 3.05
         theta1 = -0.8
         vmu = 500
         vsig = 0.33
         e3 = getRangeValueT(CoefficientsTable, period, "J_e3")
         e2 = getRangeValueT(CoefficientsTable, period, "J_e2")
         e1 = getRangeValueT(CoefficientsTable, period, "J_e1")

      # end
```

```
      Z2pt5_pred = 10 ** (theta0 + theta1 * (1 + math.erf((Log10(Vs30) - Log10(vmu)) / (vsig * math.sqrt(2)))))
      delZ2pt5 = math.log(Z2pt5) - math.log(Z2pt5_pred)

      if delZ2pt5 <= (e1 / e3):
         Fb = e1
      elif delZ2pt5 >= (e2 / e3):
         Fb = e2
      else:
         Fb = e3 * delZ2pt5
   # end
   # end

   mu = c0 + Fp + Fm + Fd + Fnl + Flin + Fb
   return mu


# end

def Log10(x):
   # CODE DEVELOPED/IMPLEMENTED BY
   #         Silvia Mazzoni, 2020
   #          smazzoni@ucla.edu
   Log10Value = math.log(x) / math.log(10.)
   return Log10Value


# end


# updated 6/1/2020 by Silvia Mazzoni
# GMM_at_760_IF_v4.R

##Grace Parker
##Modified February 26 to expand comments
##Modified March 25, 2020 to call consolidated coefficient tables

## Input Parameters ----------------------------------------------------------

##Event type: 0 == interface, 1 == slab

##region corresponds to options in the DatabaseRegion column of the flatfile, plus global. Must be a string. If no matches,
default will be global model:
## "global", "Alaska", "Cascadia", "CAM", "Japan", "SA" or "Taiwan"

##Saturation Region corresponds to regions defined by R. Archuleta and C. Ji:
## "global", "Aleutian","Alaska","Cascadia","CAM_S", "CAM_N", "Japan_Pac","Japan_Phi","SA_N","SA_S",
"Taiwan_W","Taiwan_E"

## Rrup is number in kilometers

## period can be: (-1,0,0.01,0.02,0.025,0.03,0.04,0.05,0.075,0.1,0.15,0.2,0.25,0.3,0.4,0.5,0.75,1,1.5,2,2.5,3,4,5,7.5,10)
## where -1 == PGV and 0 == PGA

## Other pertinent information ----------------------------------------------
## Coefficient files must be in the active working directory
## This function has no site term. Can only estimate ground motion at the reference condition VS30 = 760m/s
## The output is the desired median model prediction in LN units
## Take the exponential to get PGA, PSA in g or the PGV in cm/s
```

```
def PSHAB20_GMM_at_760_IF(EventType, UserRegion, saturation_region_user, Rrup, M, period):
   # CODE DEVELOPED/IMPLEMENTED BY
   #         Silvia Mazzoni, 2020
   #          smazzoni@ucla.edu
   PSHAB20_GMM_at_760_IFValue = -999
   if (LeftNumber(EventType, 1)) == 1 or (LeftNumber(EventType, 1)) == 5:
      return PSHAB20_GMM_at_760_IFValue  # ("This def is only for IF")
   # end
   region = getPSHABregion(UserRegion)
   saturation_region = getPSHABSubductionRegion(saturation_region_user)

   ##  Import Master Coefficient Table ----------------------------------------
   ##import CoefficientsTable
   #  CoefficientsTable =
readtable('PSHAB20_Table_E1_Interface_Coefficients_OneRowHeader.csv','PreserveVariableNames',True)
   CoefficientsTable = pd.read_csv('PSHAB20_Table_E1_Interface_Coefficients_OneRowHeader.csv')

   ##Define Mb
   Mb = getMbDefault("PSHAB20", EventType, UserRegion, saturation_region)

   ##  Constant ---------------------------------------------------------------

   if region == "global":
      c0 = getRangeValueT(CoefficientsTable, period, "Global_c0")
   else:
      c0 = getRangeValueT(CoefficientsTable, period, saturation_region + "_c0")
   # end

   ##  Path Term --------------------------------------------------------------
   h = 10 ** (-0.82 + 0.252 * M)
   Rref = math.sqrt(1 + h ** 2)
   r = math.sqrt(Rrup ** 2 + h ** 2)
   LogR = math.log(r)
   R_Rref = math.log(r / Rref)

   #  #Need  to isolate regional anelastic coefficient, a0
   if region == "global":
      a0 = getRangeValueT(CoefficientsTable, period, "Global_a0")
   else:
      a0 = getRangeValueT(CoefficientsTable, period, region + "_a0")
   # end
   if checkEmptyNA999(a0):
      a0 = getRangeValueT(CoefficientsTable, period, "Global_a0")
   # end

   c1 = getRangeValueT(CoefficientsTable, period, "c1")
   b4 = getRangeValueT(CoefficientsTable, period, "b4")

   Fp = (c1 * LogR + (b4 * M) * R_Rref + a0 * r)

   ##  Magnitude Scaling ------------------------------------------------------
   c4 = getRangeValueT(CoefficientsTable, period, "c4")
   c5 = getRangeValueT(CoefficientsTable, period, "c5")
   c6 = getRangeValueT(CoefficientsTable, period, "c6")
   Fm = c4 * func1(M, Mb) + c6 * func2(M, Mb) + c5 * func3(M, Mb)
```

```
   ##  Add it all up! ------------------------------------------------------------

   mu = c0 + Fp + Fm

   PSHAB20_GMM_at_760_IFValue = mu
   return PSHAB20_GMM_at_760_IFValue


# end


# Updated by Silvia Mazzoni 6/1/2020

# GMM_at_Vs30_IF_v4.R

##Grace Parker
##Modified February 26, 2020, to expand comments
##Modified March 25, 2020, to take coefficients from "Table_E1_Interface_Coefficients.csv"

## Input Parameters ----------------------------------------------------------

##Event type: 0 == interface, 1 == slab

##region corresponds to options in the DatabaseRegion column of the flatfile, plus global. Must be a string. If no matches,
default will be global model:
## "global", "Alaska", "Cascadia", "CAM", "Japan", "SA" or "Taiwan"

##Saturation Region corresponds to regions defined by R. Archuleta and C. Ji:
## "global", "Aleutian","Alaska","Cascadia","CAM_S", "CAM_N", "Japan_Pac","Japan_Phi","SA_N","SA_S",
"Taiwan_W","Taiwan_E"

## Rrup is number in kilometers

## period can be: (-1,0,0.01,0.02,0.025,0.03,0.04,0.05,0.075,0.1,0.15,0.2,0.25,0.3,0.4,0.5,0.75,1,1.5,2,2.5,3,4,5,7.5,10)
## where -1 == PGV and 0 == PGA

##VS30 in units m/s

##Z2.5 in units m. Only used if DatabaseRegion == "Japan" or "Cascadia". Can also specify "default" to get no basin term

##basin is only used if DatabaseRegion == "Cascadia". Value can be 0, 1, or 2, where 0 == having an estimate of Z2.5 outside
mapped basin, 1 == Seattle basin, and 0 == other mapped basin (Tacoma, Everett, Georgia, etc.)

## Other pertinent information ----------------------------------------------
##Coefficient files must be in the active working directory
## "GMM_at_VS30_IF_v3.R" calls function "GMM_at_760_IF_v2.R" to compute PGAr in the nonlinear site term. This
function must be in the R environment else: an error will occur.
## The output is the desired median model prediction in LN units
#

def PSHAB20_GMM_at_VS30_IF(EventType, UserRegion, saturation_region_user, Rrup, M, period, Vs30, Z2pt5, basin):
   # CODE DEVELOPED/IMPLEMENTED BY
   #         Silvia Mazzoni, 2020
   #          smazzoni@ucla.edu

   PSHAB20_GMM_at_VS30_IFValue = -999
   if (LeftNumber(EventType, 1)) == 1 or (LeftNumber(EventType, 1)) == 5:
      return PSHAB20_GMM_at_VS30_IFValue  # ("This def is only for IF")
```

```
    # end
    region = getPSHABregion(UserRegion)
    saturation_region = getPSHABSubductionRegion(saturation_region_user)
    basin = LeftNumber(basin, 1)

    ##  Import Master Coefficient Table ----------------------------------------
    #  CoefficientsTable =
readtable('PSHAB20_Table_E1_Interface_Coefficients_OneRowHeader.csv','PreserveVariableNames',True)
    CoefficientsTable = pd.read_csv('PSHAB20_Table_E1_Interface_Coefficients_OneRowHeader.csv')

    ##Define Mb
    Mb = getMbDefault("PSHAB20", EventType, UserRegion, saturation_region)

    ##  Constant ---------------------------------------------------------------
    if region == "global":
       c0 = getRangeValueT(CoefficientsTable, period, "Global_c0")
    else:
       c0 = getRangeValueT(CoefficientsTable, period, saturation_region + "_c0")
    # end
    if checkEmptyNA999(c0):
       c0 = getRangeValueT(CoefficientsTable, period, "Global_c0")
    # end

    ##  Path Term --------------------------------------------------------------
    h = 10 ** (-0.82 + 0.252 * M)
    Rref = math.sqrt(1 + h ** 2)
    r = math.sqrt(Rrup ** 2 + h ** 2)
    LogR = math.log(r)
    R_Rref = math.log(r / Rref)

    #  #Need  to isolate regional anelastic coefficient, a0
    if region == "global":
       a0 = getRangeValueT(CoefficientsTable, period, "Global_a0")
    else:
       a0 = getRangeValueT(CoefficientsTable, period, region + "_a0")
    # end
    if checkEmptyNA999(a0):
       a0 = getRangeValueT(CoefficientsTable, period, "Global_a0")
    # end

    c1 = getRangeValueT(CoefficientsTable, period, "c1")
    b4 = getRangeValueT(CoefficientsTable, period, "b4")
    Fp = (c1 * LogR + (b4 * M) * R_Rref + a0 * r)

    ##  Magnitude Scaling ------------------------------------------------------

    c4 = getRangeValueT(CoefficientsTable, period, "c4")
    c5 = getRangeValueT(CoefficientsTable, period, "c5")
    c6 = getRangeValueT(CoefficientsTable, period, "c6")

    Fm = c4 * func1(M, Mb) + c6 * func2(M, Mb) + c5 * func3(M, Mb)

    # # Linear Site Amplification ---------------------------------------------

    ## Site Coefficients
    V1 = getRangeValueT(CoefficientsTable, period, "V1_m_s")
    V2 = getRangeValueT(CoefficientsTable, period, "V2_m_s")
    Vref = getRangeValueT(CoefficientsTable, period, "Vref_m_s")
```

```
if region == "global" or region == "CAM":
   s2 = getRangeValueT(CoefficientsTable, period, "Global_s2")
   s1 = s2
elif region == "Taiwan" or region == "Japan":
   s2 = getRangeValueT(CoefficientsTable, period, region + "_s2")
   s1 = getRangeValueT(CoefficientsTable, period, region + "_s1")
else:
   s2 = getRangeValueT(CoefficientsTable, period, region + "_s2")
   s1 = s2
# end

##Compute linear site term
if Vs30 <= V1:
   Flin = s1 * math.log(Vs30 / V1) + s2 * math.log(V1 / Vref)
elif Vs30 <= V2:
   Flin = s2 * math.log(Vs30 / Vref)
else:
   Flin = s2 * math.log(V2 / Vref)
# end

# # Nonlinear Site Term ----------------------------------------------------
PGAr = math.exp(PSHAB20_GMM_at_760_IF(EventType, UserRegion, saturation_region, Rrup, M, 0))

f3 = 0.05
Vb = 200
Vref_Fnl = 760
if period >= 3:
   Fnl = 0
else:
   f4 = getRangeValueT(CoefficientsTable, period, "f4")
   f5 = getRangeValueT(CoefficientsTable, period, "f5")
   f2 = f4 * (math.exp(f5 * (min(Vs30, Vref_Fnl) - Vb)) - math.exp(f5 * (Vref_Fnl - Vb)))
   Fnl = 0 + f2 * math.log((PGAr + f3) / f3)

# end

##  Basin Term ----------------------------------------------------------------
if str(Z2pt5) == "default" or str(Z2pt5) == "Default":
   Fb = 0
elif Z2pt5 <= 0 or (region != "Japan" and region != "Cascadia"):
   Fb = 0
else:
   if region == "Cascadia":
      theta0 = 3.94
      theta1 = -0.42
      vmu = 200
      vsig = 0.2
      e1 = getRangeValueT(CoefficientsTable, period, "C_e1")

      if basin == 0:
         C_e3 = getRangeValueT(CoefficientsTable, period, "C_e3")
         C_e2 = getRangeValueT(CoefficientsTable, period, "C_e2")
         del_None = getRangeValueT(CoefficientsTable, period, "del_None")
         e3 = C_e3 + del_None
         e2 = C_e2 + del_None
      elif basin == 1:
         C_e3 = getRangeValueT(CoefficientsTable, period, "C_e3")
```

```
            C_e2 = getRangeValueT(CoefficientsTable, period, "C_e2")
            del_Seattle = getRangeValueT(CoefficientsTable, period, "del_Seattle")
            e3 = C_e3 + del_Seattle
            e2 = C_e2 + del_Seattle
         else:
            C_e3 = getRangeValueT(CoefficientsTable, period, "C_e3")
            C_e2 = getRangeValueT(CoefficientsTable, period, "C_e2")
            e3 = C_e3
            e2 = C_e2
         # end

      elif region == "Japan":
         theta0 = 3.05
         theta1 = -0.8
         vmu = 500
         vsig = 0.33
         e3 = getRangeValueT(CoefficientsTable, period, "J_e3")
         e2 = getRangeValueT(CoefficientsTable, period, "J_e2")
         e1 = getRangeValueT(CoefficientsTable, period, "J_e1")
      # end

      Z2pt5_pred = 10 ** (theta0 + theta1 * (1 + math.erf((Log10(Vs30) - Log10(vmu)) / (vsig * math.sqrt(2)))))
      delZ2pt5 = math.log(Z2pt5) - math.log(Z2pt5_pred)

      if delZ2pt5 <= (e1 / e3):
         Fb = e1
      elif delZ2pt5 >= (e2 / e3):
         Fb = e2
      else:
         Fb = e3 * delZ2pt5
   # end
   # end

   #  # Add it all up! ------------------------------------------------------------
   mu = c0 + Fp + Fm + Fnl + Fb + Flin
   return mu


# end

def PSHAB20_Median(EventType, UserRegion, SubductionSlab, Rrup, Magnitude, Zhypo, period, Vs30, Z2pt5,
PNWbasinStrux):
   # CODE DEVELOPED/IMPLEMENTED BY
   #         Silvia Mazzoni, 2020
   #          smazzoni@ucla.edu

   thisOut = -999
   XinterpMin = 0
   XinterpMax = 10
   XinterpType = "log"
   YinterpType = "log"
   extrapolateType = "extrapolate"

   # Parameters = readtable('PSHAB20_Table_E2_Slab_Coefficients_OneRowHeader.csv','PreserveVariableNames',True)
   Parameters = pd.read_csv('PSHAB20_Table_E2_Slab_Coefficients_OneRowHeader.csv')
   TvalueList = Parameters['T']

   InterpArray = interpolateFunction(period, TvalueList, XinterpType, extrapolateType, XinterpMin, XinterpMax)
```

```
    period0 = TvalueList[InterpArray[1]]
    y0 = PSHAB20_Median_AtTlist(EventType, UserRegion, SubductionSlab, Rrup, Magnitude, Zhypo, period0, Vs30, Z2pt5,
                                PNWbasinStrux)

    if InterpArray[2] <= 0:
        thisOut = y0
    else:
        period1 = TvalueList[InterpArray[2]]
        y1 = PSHAB20_Median_AtTlist(EventType, UserRegion, SubductionSlab, Rrup, Magnitude, Zhypo, period1, Vs30, Z2pt5,
                                    PNWbasinStrux)
        if LeftString(YinterpType, len("log")).lower() == "log".lower():
            if y0 <= 0:
                y0 = 0.000000001
            # end
            if y1 <= 0:
                y1 = 0.000000001
            # end
            y0 = math.log(y0)
            y1 = math.log(y1)
        # end
        thisOut = y0 + (y1 - y0) * InterpArray[3]
        if LeftString(YinterpType, len("log")).lower() == "log".lower():
            thisOut = math.exp(thisOut)
    # end

    # end

    return thisOut


# end

def PSHAB20_Median_AtTlist(EventType, UserRegion, saturation_region, Rrup, M, hypocentral_depth, period, Vs30, Z2pt5,
                           basin):
    # CODE DEVELOPED/IMPLEMENTED BY
    #         Silvia Mazzoni, 2020
    #          smazzoni@ucla.edu

    if LeftNumber(EventType, 1) == 1 or LeftNumber(EventType, 1) == 5:
        this = PSHAB20_GMM_at_VS30_Slab(EventType, UserRegion, saturation_region, Rrup, M, hypocentral_depth, period,
                                        Vs30, Z2pt5, basin)
        PSHAB20_Median_AtTlistValue = math.exp(
            PSHAB20_GMM_at_VS30_Slab(EventType, UserRegion, saturation_region, Rrup, M, hypocentral_depth, period, Vs30,
                                     Z2pt5, basin))
    else:
        PSHAB20_Median_AtTlistValue = math.exp(
            PSHAB20_GMM_at_VS30_IF(EventType, UserRegion, saturation_region, Rrup, M, period, Vs30, Z2pt5, basin))
    # end
    return PSHAB20_Median_AtTlistValue


# end
def PSHAB20_SigmaEpistemic(Ts, EventType, saturation_region_user):
    # CODE DEVELOPED/IMPLEMENTED BY
    #         Silvia Mazzoni, 2020
    #          smazzoni@ucla.edu
```

```
    if (LeftNumber(EventType, 1)) == 0:
        CoefficientsTable = pd.read_csv("PSHAB20_EpistemicModelCoefficients_Interface.csv")
    else:
        CoefficientsTable = pd.read_csv("PSHAB20_EpistemicModelCoefficients_Slab.csv")
    #end

    saturation_region = getPSHABSubductionRegion(saturation_region_user)
    if saturation_region == "Taiwan_W" or saturation_region == "Taiwan_E":
        saturation_region = "Taiwan"
    #end

    if saturation_region == "New_Zealand_N" or saturation_region == "New_Zealand_S":
        saturation_region = "global"
    #end
    SigEp1 = getRangeValueRegion(CoefficientsTable, saturation_region, 'SigEp1')
    if SigEp1 == -999:
        saturation_region = "global"
        SigEp1 = getRangeValueRegion(CoefficientsTable, saturation_region, 'SigEp1')
    #end
    SigEp2 = getRangeValueRegion(CoefficientsTable, saturation_region, 'SigEp2')
    T1 = getRangeValueRegion(CoefficientsTable, saturation_region, 'T1')
    T2 = getRangeValueRegion(CoefficientsTable, saturation_region, 'T2')
    if Ts < T1 or Ts == 0:
        sigma_epi = SigEp1
    elif Ts > T2:
        sigma_epi = SigEp2
    else:
        sigma_epi = SigEp1 - (SigEp1 - SigEp2) * (math.log(Ts / T1) / math.log(T2 / T1))
    #end
    return sigma_epi

# end


def PSHAB20_SigmaAleatory(period, Rrup, Vs30):
    # CODE DEVELOPED/IMPLEMENTED BY
    #         Silvia Mazzoni, 2020
    #          smazzoni@ucla.edu

    thisOut = -999
    XinterpMin = 0
    XinterpMax = 10
    XinterpType = "log"
    YinterpType = "linear"
    extrapolateType = "extrapolate"

    #    Parameters = readtable('PSHAB20_Table_E2_Slab_Coefficients_OneRowHeader.csv','PreserveVariableNames',True)
    Parameters = pd.read_csv('PSHAB20_Table_E2_Slab_Coefficients_OneRowHeader.csv')

    TvalueList = Parameters['T']
    InterpArray = interpolateFunction(period, TvalueList, XinterpType, extrapolateType, XinterpMin, XinterpMax)

    period0 = TvalueList[InterpArray[1]]
    y0 = PSHAB20_SigmaAleatoryAtTlist(period0, Rrup, Vs30)

    if InterpArray[2] <= 0:
        thisOut = y0
    else:
```

```
      period1 = TvalueList[InterpArray[2]]
      y1 = PSHAB20_SigmaAleatoryAtTlist(period1, Rrup, Vs30)
      if LeftString(YinterpType, len("log")).lower() == "log".lower():
         if y0 <= 0:
            y0 = 0.000000001
         # end
         if y1 <= 0:
            y1 = 0.000000001
         # end
         y0 = math.log(y0)
         y1 = math.log(y1)
      # end
      thisOut = y0 + (y1 - y0) * InterpArray[3]
      if LeftString(YinterpType, len("log")).lower() == "log".lower():
         thisOut = math.exp(thisOut)
   # end

   # end

   return thisOut


# end

def PSHAB20_SigmaAleatoryAtTlist(period, Rrup, Vs30):
   # CODE DEVELOPED/IMPLEMENTED BY
   #         Silvia Mazzoni, 2020
   #          smazzoni@ucla.edu

   PSHAB20_SigmaAleatoryAtTlist = -999

   PhiTot = PSHAB20_PhiTot(period, Rrup, Vs30)
   Tau = PSHAB20_Tau(period, Rrup, Vs30)

   PSHAB20_SigmaAleatoryAtTlistValue = math.sqrt(PhiTot * PhiTot + Tau * Tau)

   return PSHAB20_SigmaAleatoryAtTlistValue


# end


def PSHAB20_Tau(period, Rrup, Vs30):
   # CODE DEVELOPED/IMPLEMENTED BY
   #         Silvia Mazzoni, 2020
   #          smazzoni@ucla.edu

   PSHAB20_TauValue = -999
   #    SigmacoefficientsTable =
readtable("PSHAB20_Table_E3_AleatoryCoefficients_OneRowHeader.csv",'PreserveVariableNames',True)
   SigmacoefficientsTable = pd.read_csv("PSHAB20_Table_E3_AleatoryCoefficients_OneRowHeader.csv")
   Tau = getRangeValueT(SigmacoefficientsTable, period, "Tau")
   return Tau


# end
```

```
def PSHAB20_PhiTot(period, Rrup, Vs30):
   # CODE DEVELOPED/IMPLEMENTED BY
   #         Silvia Mazzoni, 2020
   #          smazzoni@ucla.edu

   PSHAB20_PhiTotValue = -999
   #    SigmacoefficientsTable =
readtable("PSHAB20_Table_E3_AleatoryCoefficients_OneRowHeader.csv",'PreserveVariableNames',True)
   SigmacoefficientsTable = pd.read_csv("PSHAB20_Table_E3_AleatoryCoefficients_OneRowHeader.csv")

   PhiTot_Phi1squared = getRangeValueT(SigmacoefficientsTable, period, "PhiTot_Phi1squared")
   PhiTot_Phi2squared = getRangeValueT(SigmacoefficientsTable, period, "PhiTot_Phi2squared")
   PhiTot_PhiVsquared = getRangeValueT(SigmacoefficientsTable, period, "PhiTot_PhiVsquared")

   # Corner Distances:
   R1 = 200  # km
   R2 = 500  # km

   V1 = 200  # m/s
   V2 = 500  # m/s

   if Vs30 <= V1:
      Rprime = max(R1, min(R2, Rrup))
      deltaVar = PhiTot_PhiVsquared * (math.log(R2 / Rprime)) / (math.log(R2 / R1))
   elif Vs30 < V2:
      Rprime = max(R1, min(R2, Rrup))
      deltaVar = PhiTot_PhiVsquared * (math.log(R2 / Rprime)) / (math.log(R2 / R1)) * (math.log(V2 / Vs30)) / (
         math.log(V2 / V1))
   else:
      deltaVar = 0
   # end

   if Rrup < R1:
      phiSquared = PhiTot_Phi1squared
   elif Rrup < R2:
      phiSquared = (PhiTot_Phi2squared - PhiTot_Phi1squared) / (math.log(R2 / R1)) * (
         math.log(Rrup / R1)) + PhiTot_Phi1squared
   else:
      phiSquared = PhiTot_Phi2squared

   # end

   PSHAB20_PhiTotValue = math.sqrt(phiSquared + deltaVar)
   return PSHAB20_PhiTotValue


# end

def PSHAB20_SigmaS2S(period, Rrup, Vs30):
   # CODE DEVELOPED/IMPLEMENTED BY
   #         Silvia Mazzoni, 2020
   #          smazzoni@ucla.edu

   PSHAB20_SigmaS2SValue = -999
   #    SigmacoefficientsTable =
readtable("PSHAB20_Table_E3_AleatoryCoefficients_OneRowHeader.csv",'PreserveVariableNames',True)
   SigmacoefficientsTable = pd.read_csv("PSHAB20_Table_E3_AleatoryCoefficients_OneRowHeader.csv")
```

```
    PhiS2S_PhiS2S0squared = getRangeValueT(SigmacoefficientsTable, period, "PhiS2S_PhiS2S0squared")
    PhiS2S_a1 = getRangeValueT(SigmacoefficientsTable, period, "PhiS2S_a1")
    VM = getRangeValueT(SigmacoefficientsTable, period, "VM")

    # Corner Distances:
    R3 = 200  # km
    R4 = 500  # km

    V3 = 200  # m/s
    V4 = 800  # m/s

    if Vs30 <= V3:
        Rprime = max(R3, min(R4, Rrup))
        deltaVarS2S = PhiS2S_a1 * math.log(V3 / VM) * (math.log(R4 / Rprime)) / (math.log(R4 / R3))
    elif Vs30 < VM:
        Rprime = max(R3, min(R4, Rrup))
        deltaVarS2S = PhiS2S_a1 * math.log(Vs30 / VM) * (math.log(R4 / Rprime)) / (math.log(R4 / R3))
    elif Vs30 < V4:
        deltaVarS2S = PhiS2S_a1 * math.log(Vs30 / VM)
    else:
        deltaVarS2S = PhiS2S_a1 * math.log(V4 / VM)
    # end

    PSHAB20_SigmaS2SValue = math.sqrt(PhiS2S_PhiS2S0squared + deltaVarS2S)
    return PSHAB20_SigmaS2SValue


# end


def PSHAB20_SigmaSS(period, Rrup, Vs30):
    # CODE DEVELOPED/IMPLEMENTED BY
    #         Silvia Mazzoni, 2020
    #          smazzoni@ucla.edu

    PSHAB20_SigmaSSValue = -999
    #    SigmacoefficientsTable =
readtable("PSHAB20_Table_E3_AleatoryCoefficients_OneRowHeader.csv",'PreserveVariableNames',True)
    SigmacoefficientsTable = pd.read_csv("PSHAB20_Table_E3_AleatoryCoefficients_OneRowHeader.csv")

    PhiSS_PhiSS1squared = getRangeValueT(SigmacoefficientsTable, period, "PhiSS_PhiSS1squared")
    PhiSS_PhiSS2squared = getRangeValueT(SigmacoefficientsTable, period, "PhiSS_PhiSS2squared")
    PhiSS_a2 = getRangeValueT(SigmacoefficientsTable, period, "PhiSS_a2")
    VM = getRangeValueT(SigmacoefficientsTable, period, "VM")

    # Corner Distances:
    R3 = 200  # km
    R4 = 500  # km
    R5 = 500  # km
    R6 = 800  # km

    V3 = 200  # m/s
    V4 = 800  # m/s

    if Rrup < R5:
        phiSquaredSS = PhiSS_PhiSS1squared
    elif Rrup < R6:
        phiSquaredSS = (PhiSS_PhiSS2squared - PhiSS_PhiSS1squared) / (math.log(R6 / R5)) * (
```

```
        math.log(Rrup / R5)) + PhiSS_PhiSS1squared
    else:
        phiSquaredSS = PhiSS_PhiSS2squared

    # end

    if Vs30 <= V3:
        Rprime = max(R3, min(R4, Rrup))
        deltaVarSS = PhiSS_a2 * math.log(V3 / VM) * (math.log(R4 / Rprime)) / (math.log(R4 / R3))
    elif Vs30 < VM:
        Rprime = max(R3, min(R4, Rrup))
        deltaVarSS = PhiSS_a2 * math.log(Vs30 / VM) * (math.log(R4 / Rprime)) / (math.log(R4 / R3))
    elif Vs30 < V4:
        deltaVarSS = PhiSS_a2 * math.log(Vs30 / VM)
    else:
        deltaVarSS = PhiSS_a2 * math.log(V4 / VM)
    # end

    PSHAB20_SigmaSSValue = math.sqrt(phiSquaredSS + deltaVarSS)
    return PSHAB20_SigmaSSValue
# end


def PSHAB20_SigmaTotal(period, Rrup, Vs30, EventType, saturation_region_user):
    # CODE DEVELOPED/IMPLEMENTED BY
    #         Silvia Mazzoni, 2020
    #          smazzoni@ucla.edu
    SigmaAleatory = PSHAB20_SigmaAleatory(period, Rrup, Vs30)
    SigmaEpistemic = PSHAB20_SigmaEpistemic(period, EventType, saturation_region_user)
    SigmaTotalValue = math.sqrt(SigmaAleatory * SigmaAleatory + SigmaEpistemic * SigmaEpistemic)
    return SigmaTotalValue
# end


def NGAsubGMM_MedianPlusMinusSigma(UserPeriod, UserRegion, Magnitude, Vs30, Rrup, AlphaBackarc, AlphaNankai,
Ztor, Zhypo, EventType,
                Z1pt0, Z2pt5, MbUser, SubductionSlab, PNWbasinStrux, Weight_KBCG20, Weight_PSHAB20,  User_NSigma,
User_EpiInSigmaModels, User_NsampleEpi ,
                User_MeanType='geometric'):
    # CODE DEVELOPED/IMPLEMENTED BY
    #         Silvia Mazzoni, 2020
    #          smazzoni@ucla.edu

    if str(MbUser).lower() == "default".lower():
        Mb = getMbDefault("KBCG20", EventType, UserRegion, SubductionSlab)
    else:
        Mb = MbUser
    # end


    period = getTFvalue(UserPeriod)
    WeightedSum = 0
    WeightSum = 0

    if Weight_KBCG20 > 0:
        KBCG20median = KBCG20_medPSA(period, Magnitude, Rrup, AlphaBackarc, AlphaNankai, Ztor, EventType, Vs30, Z1pt0,
Z2pt5,
                        Mb, UserRegion, PNWbasinStrux)
```

```
      SigmaAleatoryKBCG20 = KBCG20_sigmaAleatory(period)
      SigmaEpistemicKBCG20 = KBCG20_SigmaEpistemic(period, Magnitude, Rrup, AlphaBackarc, AlphaNankai, Ztor,
                                  EventType, Vs30, Z1pt0, Z2pt5, Mb, UserRegion, User_NsampleEpi,
                                  PNWbasinStrux)
      KBCG20SigmaTotal = math.sqrt(SigmaAleatoryKBCG20 * SigmaAleatoryKBCG20 + SigmaEpistemicKBCG20 *
SigmaEpistemicKBCG20)
      KBCG20 = math.exp(math.log(KBCG20median)+User_NSigma*KBCG20SigmaTotal);
      WeightSum = WeightSum + Weight_KBCG20
      if User_MeanType.lower() == "arithmetic".lower():
        WeightedSum = WeightedSum + KBCG20 * Weight_KBCG20
      else:
        WeightedSum = WeightedSum + math.log(KBCG20) * Weight_KBCG20
    # end
    # end
    if Weight_PSHAB20 > 0:
      PSHAB20median = PSHAB20_Median(EventType, UserRegion, SubductionSlab, Rrup, Magnitude, Zhypo, period, Vs30,
Z2pt5,
                      PNWbasinStrux)
      SigmaAleatoryPSHAB20 = PSHAB20_SigmaAleatory(period, Rrup, Vs30)
      SigmaEpistemicPSHAB20 = PSHAB20_SigmaEpistemic(period,EventType,SubductionSlab)
      PSHAB20SigmaTotal = math.sqrt(SigmaAleatoryPSHAB20 * SigmaAleatoryPSHAB20 + SigmaEpistemicPSHAB20 *
SigmaEpistemicPSHAB20);
      PSHAB20 = math.exp(math.log(PSHAB20median) + User_NSigma * PSHAB20SigmaTotal);
      WeightSum = WeightSum + Weight_PSHAB20
      if User_MeanType.lower() == "arithmetic".lower():
        WeightedSum = WeightedSum + PSHAB20 * Weight_PSHAB20
      else:
        WeightedSum = WeightedSum + math.log(PSHAB20) * Weight_PSHAB20
    # end
    # end
    if WeightSum > 0:
      if User_MeanType.lower() == "arithmetic".lower():
        NGAsubGMM_MedianPlusMinusSigmaValue = WeightedSum / WeightSum
      else:
        NGAsubGMM_MedianPlusMinusSigmaValue = math.exp(WeightedSum / WeightSum)
    # end
    # end
    return NGAsubGMM_MedianPlusMinusSigmaValue


# end



# ################################################################
# The following script performs a large sensitivity study (see last def in this file)
# Running this sensitivity study make take a few minutes.
# The results from this study are output to a screen a to a csv file
#
def RunSensitivityStudy():
    CaseLabel = []
    CaseLabelRegion = []
    Mag = []
    Vs30mps = []
    Reg = []
    Rrup_km = []
    alpBack = []
    alpNank = []
```

```
    Ztor_km = []
    SeatBasn = []
    Z1pt0m = []
    Z2pt5m = []
    NsplEpi = []
    Mbreak = []
    Zhypo_km = []
    SatReg = []
    Basin = []
    Nsigma = []
    InterIntra = []
    T_sec = []
    outSensitivityData = []
    outSensitivityDataHeader = []

    print(' ### Performing Sensitivity Study')
    for iSet in range(2):
        thisCase = iSet
        if iSet == 0:  # interface
            CaseLabel=
["Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_
NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_se
c=0.01","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,Seat
Basn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigm
a=1,T_sec=0.02","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interf
ace,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasi
n,Nsigma=1,T_sec=0.03","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=
0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0
_NoBasin,Nsigma=1,T_sec=0.05","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,Inter
Intra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,
Basin=0_NoBasin,Nsigma=1,T_sec=0.075","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_k
m=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatRe
g=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.1","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,
Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55
,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.15","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alp
Nank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhyp
o_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.2","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBac
k=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.
9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.25","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=20
0,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,M
break=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.3","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_
km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=
100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.4","Mag=8,Vs30_mps=760,Reg=0_global
,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,Ns
plEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.5","Mag=8,Vs30_mps=760,Reg=0_
global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2
000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.75","Mag=8,Vs30_mps=760
,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2
.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=1","Mag=8,Vs30_mps
=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=5
50,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=1.5","Mag=8,Vs3
0_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0
_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=2","Mag=8
,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,
Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=3","M
ag=8,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_No
Basin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=
4","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=
```

0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=5","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=7.5","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=10","Mag=4,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=4.25,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=4.5,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=4.75,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=5,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=5.25,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=5.5,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=5.75,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=6,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=6.25,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=6.5,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=6.75,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=7,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=7.25,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=7.5,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=7.75,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8.25,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8.5,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8.75,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=9,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=9.25,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=10,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=13,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=16.9,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=21.97,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Inte

rface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=28.561,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=37.1293,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=48.26809,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=62.7485169999999,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=81.5730720999999,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=106.04499373,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=137.858491849,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=179.2160394037,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=232.980851224809,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=302.875106592252,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=393.737638569928,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=511.858930140906,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=665.416609183177,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=865.04159193813,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=1124.55406951957,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=1461.92029037544,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=1900.49637748807,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=2470.64529073449,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=150,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=169.5,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=191.535,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=216.43455,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=244.5710415,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=276.365276895,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=312.29276289135,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=352.890822067225,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=398.766628935964,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,

Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=450.60629069764,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=509.185108488333,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=575.379172591816,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=650.178465028752,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=734.701665482489,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=830.212881995213,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=938.140556654591,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=1060.09882901969,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=1197.91167679225,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=1353.64019477524,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=1529.61342009602,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=1728.4631647085,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=1953.16337612061,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=1_Alaska,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=8.6,Zhypo_km=55,SatReg=Alaska,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=1_Alaska,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=8,Zhypo_km=55,SatReg=Aleutian,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=2_Cascadia,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.7,Zhypo_km=55,SatReg=Cascadia,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=2_Cascadia,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=1_InSeattleBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.7,Zhypo_km=55,SatReg=Cascadia,Basin=1_InSeattleBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=2_Cascadia,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=2_InOtherPNWbasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.7,Zhypo_km=55,SatReg=Cascadia,Basin=2_InOtherPNWbasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=3_CentralAmerica&Mexico,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=2_InOtherPNWbasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.4,Zhypo_km=55,SatReg=Central_America_N,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=3_CentralAmerica&Mexico,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=2_InOtherPNWbasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.4,Zhypo_km=55,SatReg=Central_America_S,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=4_Japan,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=2_InOtherPNWbasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=8.5,Zhypo_km=55,SatReg=Japan_Pac,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=4_Japan,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=2_InOtherPNWbasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.7,Zhypo_km=55,SatReg=Japan_Phi,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=5_NewZealand,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=2_InOtherPNWbasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=New_Zealand_N,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=5_NewZealand,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=2_InOtherPNWbasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=New_Zealand_S,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=6_SouthAmerica,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=2_InOtherPNWbasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=8.5,

```
Zhypo_km=55,SatReg=South_America_N,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=6_SouthAm
erica,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=2_InOtherPNWbasin,Z1.0_m=550,Z
2.5_m=2000,NsplEpi=100,Mbreak=8.6,Zhypo_km=55,SatReg=South_America_S,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","M
ag=8,Vs30_mps=760,Reg=7_Taiwan,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_Interface,SeatBasn=2_In
OtherPNWbasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.1,Zhypo_km=55,SatReg=Taiwan_E,Basin=0_NoBasin,Nsi
gma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=7_Taiwan,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=0_In
terface,SeatBasn=2_InOtherPNWbasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.1,Zhypo_km=55,SatReg=Taiwan_
W,Basin=0_NoBasin,Nsigma=1,T_sec=0.01"];
        CaseLabelRegion=
["0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_
Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB
_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","
0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_M
b7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_
GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0
_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb
7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_G
LB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_
GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7
.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GL
B_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_
GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7
.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GL
B_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_
GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","1_AK_AK_Mb8.6","1_AK_Aleu_Mb8","2_CSC_CSC_Mb7.7","2
_CSC_CSC_Mb7.7_1_InSeattleBasin","2_CSC_CSC_Mb7.7_2_InOtherPNWbasin","3_CAM_CAM_N_Mb7.4","3_CAM_CAM_S
_Mb7.4","4_JP_JP_Pac_Mb8.5","4_JP_JP_Phi_Mb7.7","5_NZ_NZ_N_Mb7.9","5_NZ_NZ_S_Mb7.9","6_SAM_SAM_N_Mb8.5"
,"6_SAM_SAM_S_Mb8.6","7_TW_TW_E_Mb7.1","7_TW_TW_W_Mb7.1"];

        Mag=
[8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,4,4.25,4.5,4.75,5,5.25,5.5,5.75,6,6.25,6.5,6.75,7,7.25,7.5,7.75,8,8.25,8.5,8.75,9,9.25,
8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8];
        Vs30mps=
[760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,7
60,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760
,760,760,760,760,760,760,760,760,150,169.5,191.535,216.43455,244.5710415,276.365276895,312.29276289135,352.8908
22067225,398.766628935964,450.60629069764,509.185108488333,575.379172591816,650.178465028752,734.701665482
489,830.212881995213,938.140556654591,1060.09882901969,1197.91167679225,1353.64019477524,1529.61342009602,
1728.4631647085,1953.16337612061,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760];
        Reg=
["0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_global","
0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_
global","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_gl
obal","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_glo
bal","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_globa
l","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_global",
"0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0
_global","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_global","1_Alaska","1_
Alaska","2_Cascadia","2_Cascadia","2_Cascadia","3_CentralAmerica&Mexico","3_CentralAmerica&Mexico","4_Japan","4_Ja
pan","5_NewZealand","5_NewZealand","6_SouthAmerica","6_SouthAmerica","7_Taiwan","7_Taiwan"];
        Rrup_km=
[200,200,200,200,200,200,200,200,200,200,200,200,200,200,200,200,200,200,200,200,200,200,200,200,200,200,200,200,2
00,200,200,200,200,200,200,200,200,200,200,200,200,200,200,10,13,16.9,21.97,28.561,37.1293,48.26809,62.74851699999
99,81.5730720999999,106.04499373,137.858491849,179.2160394037,232.980851224809,302.875106592252,393.7376385
69928,511.858930140906,665.416609183177,865.04159193813,1124.55406951957,1461.92029037544,1900.49637748807
,2470.64529073449,200,200,200,200,200,200,200,200,200,200,200,200,200,200,200,200,200,200,200,200,200,200,200,200
,200,200,200,200,200,200,200,200,200,200,200,200,200,200];
```

```
	alpBack=
[0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0];
	alpNank=
[0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0];
	Ztor_km=
[10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10
,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10
,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10];
	SeatBasn=
["0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_
NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoB
asin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin"
,"0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_
NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoB
asin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin"
,"0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_
NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoB
asin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin"
,"0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","1_InSeattleBasin","2_InOther
PNWbasin","2_InOtherPNWbasin","2_InOtherPNWbasin","2_InOtherPNWbasin","2_InOtherPNWbasin","2_InOtherPNWbasi
n","2_InOtherPNWbasin","2_InOtherPNWbasin","2_InOtherPNWbasin","2_InOtherPNWbasin","2_InOtherPNWbasin"];
	Z1pt0m=
[550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,5
50,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550
,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,5
50,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550];
	Z2pt5m=
[2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,
2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2
000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,20
00,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,200
0,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000];
	NsplEpi=
[100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,1
00,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100
,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,1
00,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100];
	Mbreak=
[7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,
7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7
.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,8.6,8,7.7,7.7,7.7,7.4,7.4,8.5,7.7,7.9,7.9,8.5,
8.6,7.1,7.1];
	Zhypo_km=
[55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55
,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55
,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55];
	SatReg=
["global","global","global","global","global","global","global","global","global","global","global","global","global","global","gl
obal","global","global","global","global","global","global","global","global","global","global","global","global","global","globa
l","global","global","global","global","global","global","global","global","global","global","global","global","global","global","
global","global","global","global","global","global","global","global","global","global","global","global","global","global","glo
bal","global","global","global","global","global","global","global","global","global","global","global","global","global","global
","global","global","global","global","global","global","global","global","global","global","global","global","global","global","g
lobal","global","Alaska","Aleutian","Cascadia","Cascadia","Cascadia","Central_America_N","Central_America_S","Japan_Pac
","Japan_Phi","New_Zealand_N","New_Zealand_S","South_America_N","South_America_S","Taiwan_E","Taiwan_W"];
```

```
        Basin=
["0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_
NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoB
asin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin"
,"0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_
NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoB
asin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin"
,"0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_
NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoB
asin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin"
,"0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","1_InSeattleBasin","2_InOther
PNWbasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_No
Basin","0_NoBasin"];
        Nsigma=
[1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1
,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1];
        InterIntra=
["0_Interface","0_Interface","0_Interface","0_Interface","0_Interface","0_Interface","0_Interface","0_Interface","0_Interface
e","0_Interface","0_Interface","0_Interface","0_Interface","0_Interface","0_Interface","0_Interface","0_Interf
ace","0_Interface","0_Interface","0_Interface","0_Interface","0_Interface","0_Interface","0_Interface","0_Int
erface","0_Interface","0_Interface","0_Interface","0_Interface","0_Interface","0_Interface","0_Interface","0_I
nterface","0_Interface","0_Interface","0_Interface","0_Interface","0_Interface","0_Interface","0_Interface","0
_Interface","0_Interface","0_Interface","0_Interface","0_Interface","0_Interface","0_Interface","0_Interface",
"0_Interface","0_Interface","0_Interface","0_Interface","0_Interface","0_Interface","0_Interface","0_Interface","0_Interfac
e","0_Interface","0_Interface","0_Interface","0_Interface","0_Interface","0_Interface","0_Interface","0_Interf
ace","0_Interface","0_Interface","0_Interface","0_Interface","0_Interface","0_Interface","0_Interface","0_Int
erface","0_Interface","0_Interface","0_Interface","0_Interface","0_Interface","0_Interface","0_Interface","0_I
nterface","0_Interface","0_Interface","0_Interface","0_Interface","0_Interface","0_Interface","0_Interface","0
_Interface","0_Interface","0_Interface","0_Interface","0_Interface","0_Interface"];
        T_sec=
[0.01,0.02,0.03,0.05,0.075,0.1,0.15,0.2,0.25,0.3,0.4,0.5,0.75,1,1.5,2,3,4,5,7.5,10,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01
,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01
,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01
,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01];



    else:  # intraslab



        CaseLabel=
["Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=40,InterIntra=1_Intraslab,SeatBasn=0_
NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_se
c=0.01","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=40,InterIntra=1_Intraslab,SeatB
asn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma
=1,T_sec=0.02","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=40,InterIntra=1_Intrasla
b,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,
Nsigma=1,T_sec=0.03","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=40,InterIntra=1_
Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_N
oBasin,Nsigma=1,T_sec=0.05","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=40,Interl
ntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Ba
sin=0_NoBasin,Nsigma=1,T_sec=0.075","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km
=40,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=
global,Basin=0_NoBasin,Nsigma=1,T_sec=0.1","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Zt
or_km=40,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,S
atReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.15","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNa
nk=0,Ztor_km=40,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_k
m=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.2","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0
```

,alpNank=0,Ztor_km=40,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.25","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=40,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.3","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=40,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.4","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=40,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.5","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=40,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.75","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=40,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=1","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=40,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=1.5","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=40,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=2","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=40,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=3","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=40,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=4","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=40,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=5","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=40,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=7.5","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=40,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=10","Mag=4,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=4.25,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=4.5,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=4.75,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=5,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=5.25,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=5.5,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=5.75,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=6,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=6.25,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=6.5,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=6.75,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=7,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=7.25,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=7.5,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=7.75,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,

InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8.25,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8.5,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8.75,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=9,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=9.25,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=40,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=52,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=67.6,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=87.88,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=114.244,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=148.5172,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=193.07236,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=250.994068,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=326.2922884,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=424.179974919999,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=551.433967395999,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=716.864157614799,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=931.923404899238,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=1211.50042636901,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=1574.95055427971,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=2047.43572056362,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=2661.66643673271,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=3460.16636775252,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=4498.21627807827,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=5847.68116150175,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=7601.98550995228,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z

2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=9882.58116293796,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.9,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=150,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=169.5,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=191.535,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=216.43455,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=244.5710415,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=276.365276895,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=312.2927628913 5,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=352.890822067225,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=398.766628935964,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=450.60629069764,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=509.185108488333,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=575.379172591816,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=650.178465028752,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=734.701665482489,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=830.212881995213,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=938.140556654591,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=1060.09882901969,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=1197.911676 79225,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=1353.64019477524,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=1529.61342009602,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=1728.4631647085,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=1953.16337612061,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=0_global,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=global,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=1_Alaska,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.2,Zhypo_km=55,SatReg=Alaska,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=1_Alaska,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=8,Zhypo_km=55,SatReg=Aleutian,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=2_Cascadia,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=0_NoBasin,Z1.0_m=

```
550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.2,Zhypo_km=55,SatReg=Cascadia,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=
8,Vs30_mps=760,Reg=2_Cascadia,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=1_InSe
attleBasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.2,Zhypo_km=55,SatReg=Cascadia,Basin=1_InSeattleBasin,Nsig
ma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=2_Cascadia,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_I
ntraslab,SeatBasn=2_InOtherPNWbasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.2,Zhypo_km=55,SatReg=Cascadi
a,Basin=2_InOtherPNWbasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=3_CentralAmerica&Mexico,Rrup_km=200,
alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=2_InOtherPNWbasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi
=100,Mbreak=7.4,Zhypo_km=55,SatReg=Central_America_N,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=7
60,Reg=3_CentralAmerica&Mexico,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=2_InO
therPNWbasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=Central_America_S,Basin=0_NoB
asin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=4_Japan,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntr
a=1_Intraslab,SeatBasn=2_InOtherPNWbasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=Ja
pan_Pac,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=4_Japan,Rrup_km=200,alpBack=0,alpNank=0
,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=2_InOtherPNWbasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhy
po_km=55,SatReg=Japan_Phi,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=5_NewZealand,Rrup_k
m=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=2_InOtherPNWbasin,Z1.0_m=550,Z2.5_m=2000,
NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=New_Zealand_N,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mp
s=760,Reg=5_NewZealand,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=2_InOtherPN
Wbasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.6,Zhypo_km=55,SatReg=New_Zealand_S,Basin=0_NoBasin,Nsigm
a=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=6_SouthAmerica,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=
1_Intraslab,SeatBasn=2_InOtherPNWbasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.3,Zhypo_km=55,SatReg=Sout
h_America_N,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=6_SouthAmerica,Rrup_km=200,alpBack
=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=2_InOtherPNWbasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,M
break=7.2,Zhypo_km=55,SatReg=South_America_S,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,Vs30_mps=760,Reg=7_
Taiwan,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=2_InOtherPNWbasin,Z1.0_m=550,
Z2.5_m=2000,NsplEpi=100,Mbreak=7.7,Zhypo_km=55,SatReg=Taiwan_E,Basin=0_NoBasin,Nsigma=1,T_sec=0.01","Mag=8,V
s30_mps=760,Reg=7_Taiwan,Rrup_km=200,alpBack=0,alpNank=0,Ztor_km=10,InterIntra=1_Intraslab,SeatBasn=2_InOtherP
NWbasin,Z1.0_m=550,Z2.5_m=2000,NsplEpi=100,Mbreak=7.7,Zhypo_km=55,SatReg=Taiwan_W,Basin=0_NoBasin,Nsigma=1
,T_sec=0.01"];
        CaseLabelRegion=
["0_GLB_GLB_Mb7.6","0_GLB_GLB_Mb7.6","0_GLB_GLB_Mb7.6","0_GLB_GLB_Mb7.6","0_GLB_GLB_Mb7.6","0_GLB_GLB_
Mb7.6","0_GLB_GLB_Mb7.6","0_GLB_GLB_Mb7.6","0_GLB_GLB_Mb7.6","0_GLB_GLB_Mb7.6","0_GLB_GLB_Mb7.6","0_GLB
_GLB_Mb7.6","0_GLB_GLB_Mb7.6","0_GLB_GLB_Mb7.6","0_GLB_GLB_Mb7.6","0_GLB_GLB_Mb7.6","0_GLB_GLB_Mb7.6","
0_GLB_GLB_Mb7.6","0_GLB_GLB_Mb7.6","0_GLB_GLB_Mb7.6","0_GLB_GLB_Mb7.6","0_GLB_GLB_Mb7.6","0_GLB_GLB_M
b7.6","0_GLB_GLB_Mb7.6","0_GLB_GLB_Mb7.6","0_GLB_GLB_Mb7.6","0_GLB_GLB_Mb7.6","0_GLB_GLB_Mb7.6","0_GLB_
GLB_Mb7.6","0_GLB_GLB_Mb7.6","0_GLB_GLB_Mb7.6","0_GLB_GLB_Mb7.6","0_GLB_GLB_Mb7.6","0_GLB_GLB_Mb7.6","0
_GLB_GLB_Mb7.6","0_GLB_GLB_Mb7.6","0_GLB_GLB_Mb7.6","0_GLB_GLB_Mb7.6","0_GLB_GLB_Mb7.6","0_GLB_GLB_Mb
7.6","0_GLB_GLB_Mb7.6","0_GLB_GLB_Mb7.6","0_GLB_GLB_Mb7.6","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_G
LB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_
GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7
.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GL
B_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.9","0_GLB_GLB_Mb7.6","0_GLB_GLB_Mb7.6","0_GLB_GLB_Mb7.6","0_
GLB_GLB_Mb7.6","0_GLB_GLB_Mb7.6","0_GLB_GLB_Mb7.6","0_GLB_GLB_Mb7.6","0_GLB_GLB_Mb7.6","0_GLB_GLB_Mb7
.6","0_GLB_GLB_Mb7.6","0_GLB_GLB_Mb7.6","0_GLB_GLB_Mb7.6","0_GLB_GLB_Mb7.6","0_GLB_GLB_Mb7.6","0_GLB_GL
B_Mb7.6","0_GLB_GLB_Mb7.6","0_GLB_GLB_Mb7.6","0_GLB_GLB_Mb7.6","0_GLB_GLB_Mb7.6","0_GLB_GLB_Mb7.6","0_
GLB_GLB_Mb7.6","0_GLB_GLB_Mb7.6","0_GLB_GLB_Mb7.6","1_AK_AK_Mb7.2","1_AK_Aleu_Mb8","2_CSC_CSC_Mb7.2","2
_CSC_CSC_Mb7.2_1_InSeattleBasin","2_CSC_CSC_Mb7.2_2_InOtherPNWbasin","3_CAM_CAM_N_Mb7.4","3_CAM_CAM_S
_Mb7.6","4_JP_JP_Pac_Mb7.6","4_JP_JP_Phi_Mb7.6","5_NZ_NZ_N_Mb7.6","5_NZ_NZ_S_Mb7.6","6_SAM_SAM_N_Mb7.3"
,"6_SAM_SAM_S_Mb7.2","7_TW_TW_E_Mb7.7","7_TW_TW_W_Mb7.7"];

        Mag=
[8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,4,4.25,4.5,4.75,5,5.25,5.5,5.75,6,6.25,6.5,6.75,7,7.25,7.5,7.75,8,8.25,8.5,8.75,9,9.25,
8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8];
        Vs30mps=
[760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,7
60,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760
,760,760,760,760,760,760,760,760,150,169.5,191.535,216.43455,244.5710415,276.365276895,312.29276289135,352.8908
22067225,398.766628935964,450.60629069764,509.185108488333,575.379172591816,650.17846502875 2,734.701665482
```

```
489,830.212881995213,938.140556654591,1060.09882901969,1197.91167679225,1353.64019477524,1529.61342009602,
1728.4631647085,1953.16337612061,760,760,760,760,760,760,760,760,760,760,760,760,760,760,760];
        Reg=
["0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_global","
0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_
global","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_gl
obal","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_glo
bal","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_globa
l","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_global",
"0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0
_global","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_global","0_global","1_Alaska","1_
Alaska","2_Cascadia","2_Cascadia","2_Cascadia","3_CentralAmerica&Mexico","3_CentralAmerica&Mexico","4_Japan","4_Ja
pan","5_NewZealand","5_NewZealand","6_SouthAmerica","6_SouthAmerica","7_Taiwan","7_Taiwan"];
        Rrup_km=
[200,200,200,200,200,200,200,200,200,200,200,200,200,200,200,200,200,200,200,200,200,200,200,200,200,200,200,200,2
00,200,200,200,200,200,200,200,200,200,200,200,200,200,200,40,52,67.6,87.88,114.244,148.5172,193.07236,250.994068,
326.2922884,424.179974919999,551.433967395999,716.864157614799,931.923404899238,1211.50042636901,1574.9505
5427971,2047.43572056362,2661.66643673271,3460.16636775252,4498.21627807827,5847.68116150175,7601.98550995
228,9882.58116293796,200,200,200,200,200,200,200,200,200,200,200,200,200,200,200,200,200,200,200,200,200,200,200,
200,200,200,200,200,200,200,200,200,200,200,200,200,200,200];
        alpBack=
[0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0];
        alpNank=
[0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0];
        Ztor_km=
[40,40,40,40,40,40,40,40,40,40,40,40,40,40,40,40,40,40,40,40,40,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10
,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10
,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10];
        SeatBasn=
["0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_
NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoB
asin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin"
,"0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_
NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoB
asin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin"
,"0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_
NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoB
asin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin"
,"0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","1_InSeattleBasin","2_InOther
PNWbasin","2_InOtherPNWbasin","2_InOtherPNWbasin","2_InOtherPNWbasin","2_InOtherPNWbasin","2_InOtherPNWbasi
n","2_InOtherPNWbasin","2_InOtherPNWbasin","2_InOtherPNWbasin","2_InOtherPNWbasin","2_InOtherPNWbasin"];
        Z1pt0m=
[550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,5
50,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550
,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,5
50,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550,550];
        Z2pt5m=
[2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,
2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2
000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,20
00,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,200
0,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000,2000];
        NsplEpi=
[100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,1
00,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100
```

```
,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,1
00,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100,100];
        Mbreak=
[7.6,7.6,7.6,7.6,7.6,7.6,7.6,7.6,7.6,7.6,7.6,7.6,7.6,7.6,7.6,7.6,7.6,7.6,7.6,7.6,7.6,7.6,7.6,7.6,7.6,7.6,7.6,7.6,7.6,7.6,7.6,7.6,7.6,7.6,7.6,
7.6,7.6,7.6,7.6,7.6,7.6,7.6,7.6,7.6,7.6,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.9,7.6,7
.6,7.6,7.6,7.6,7.6,7.6,7.6,7.6,7.6,7.6,7.6,7.6,7.6,7.6,7.6,7.6,7.6,7.6,7.6,7.6,7.6,7.2,8,7.2,7.2,7.2,7.4,7.6,7.6,7.6,7.6,7.6,7.3,
7.2,7.7,7.7];
        Zhypo_km=
[55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55
,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55
,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55,55];
        SatReg=
["global","global","global","global","global","global","global","global","global","global","global","global","global","global","gl
obal","global","global","global","global","global","global","global","global","global","global","global","global","global","globa
l","global","global","global","global","global","global","global","global","global","global","global","global","global","global","
global","global","global","global","global","global","global","global","global","global","global","global","global","global","glo
bal","global","global","global","global","global","global","global","global","global","global","global","global","global","global
","global","global","global","global","global","global","global","global","global","global","global","global","global","global","g
lobal","global","Alaska","Aleutian","Cascadia","Cascadia","Cascadia","Central_America_N","Central_America_S","Japan_Pac
","Japan_Phi","New_Zealand_N","New_Zealand_S","South_America_N","South_America_S","Taiwan_E","Taiwan_W"];
        Basin=
["0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_
NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoB
asin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin"
,"0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_
NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoB
asin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin"
,"0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_
NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoB
asin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin"
,"0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","1_InSeattleBasin","2_InOther
PNWbasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_NoBasin","0_No
Basin","0_NoBasin"];
        Nsigma=
[1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1
,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1];
        InterIntra=
["1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab"
,"1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab"
,"1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab"
,"1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab"
,"1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab"
,"1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab"
,"1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab"
,"1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab"
,"1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab"
,"1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab"
,"1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab"
,"1_Intraslab","1_Intraslab","1_Intraslab","1_Intraslab"];
        T_sec=
[0.01,0.02,0.03,0.05,0.075,0.1,0.15,0.2,0.25,0.3,0.4,0.5,0.75,1,1.5,2,3,4,5,7.5,10,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01
,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01
,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01
,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01];

    # end

    for irow in range(len(T_sec)):
```

```
        thisCaseLabel = CaseLabel[irow]
        thisCaseLabelRegion = CaseLabelRegion[irow]
        thisMagnitude = Mag[irow]
        thisVs30 = Vs30mps[irow]
        thisRegion = Reg[irow]
        thisRrup = Rrup_km[irow]
        thisAlphaBackarc = alpBack[irow]
        thisAlphaNankai = alpNank[irow]
        thisZtor = Ztor_km[irow]
        thisSeatBasn = SeatBasn[irow]
        thisZ1pt0 = Z1pt0m[irow]
        thisZ2pt5 = Z2pt5m[irow]
        thisNsampleEpi = NsplEpi[irow]
        thisMb = Mbreak[irow]
        thisZhypo = Zhypo_km[irow]
        thisSubductionSlab = SatReg[irow]
        thisPNWbasinStrux = Basin[irow]
        thisNsigma = Nsigma[irow]
        thisEventType = InterIntra[irow]
        thisT = T_sec[irow]
        hereKBCG20MedianPSA = KBCG20_medPSA(thisT, thisMagnitude, thisRrup, thisAlphaBackarc, thisAlphaNankai,
                              thisZtor, thisEventType, thisVs30, thisZ1pt0, thisZ2pt5, thisMb,
                              thisRegion, thisPNWbasinStrux)
        hereKBCG20Tau = KBCG20_sigmaTau(thisT)
        hereKBCG20Phi = KBCG20_sigmaPhi(thisT)
        hereKBCG20SigmaAleatory = KBCG20_sigmaAleatory(thisT)
        hereKBCG20SigmaEpistemic = KBCG20_SigmaEpistemic(thisT, thisMagnitude, thisRrup, thisAlphaBackarc,
                                   thisAlphaNankai, thisZtor, thisEventType, thisVs30,
                                   thisZ1pt0, thisZ2pt5, thisMb, thisRegion, thisNsampleEpi,
                                   thisPNWbasinStrux)
        hereKBCG20SigmaTotal = KBCG20_SigmaTotal(thisT, thisMagnitude, thisRrup, thisAlphaBackarc, thisAlphaNankai,
                               thisZtor, thisEventType, thisVs30, thisZ1pt0, thisZ2pt5, thisMb,
                               thisRegion, thisNsampleEpi, thisPNWbasinStrux)
        hereKBCG20MedianMinusNsigmaxSigmaTotal = math.exp(
           math.log(hereKBCG20MedianPSA) - thisNsigma * hereKBCG20SigmaTotal)
        hereKBCG20MedianPlusNsigmaxSigmaTotal = math.exp(
           math.log(hereKBCG20MedianPSA) + thisNsigma * hereKBCG20SigmaTotal)
        hereKBCG20MedianMinusNsigmaxSigmaAleatory = math.exp(
           math.log(hereKBCG20MedianPSA) - thisNsigma * hereKBCG20SigmaAleatory)
        hereKBCG20MedianPlusNsigmaxSigmaAleatory = math.exp(
           math.log(hereKBCG20MedianPSA) + thisNsigma * hereKBCG20SigmaAleatory)
        hereKBCG20MedianMinusNsigmaxSigmaEpistemic = math.exp(
           math.log(hereKBCG20MedianPSA) - thisNsigma * hereKBCG20SigmaEpistemic)
        hereKBCG20MedianPlusNsigmaxSigmaEpistemic = math.exp(
           math.log(hereKBCG20MedianPSA) + thisNsigma * hereKBCG20SigmaEpistemic)
        herePSHAB20MedianPSA = PSHAB20_Median(thisEventType, thisRegion, thisSubductionSlab, thisRrup,
                               thisMagnitude, thisZhypo, thisT, thisVs30, thisZ2pt5,
                               thisPNWbasinStrux)
        herePSHAB20Tau = PSHAB20_Tau(thisT, thisRrup, thisVs30)
        herePSHAB20PhiTot = PSHAB20_PhiTot(thisT, thisRrup, thisVs30)
        herePSHAB20PhiS2S = PSHAB20_SigmaS2S(thisT, thisRrup, thisVs30)
        herePSHAB20PhiSS = PSHAB20_SigmaSS(thisT, thisRrup, thisVs30)
        herePSHAB20SigmaAleatory = PSHAB20_SigmaAleatory(thisT, thisRrup, thisVs30)
        herePSHAB20SigmaEpistemic = PSHAB20_SigmaEpistemic(thisT, thisEventType, thisSubductionSlab)
        herePSHAB20SigmaTotal = PSHAB20_SigmaTotal(thisT, thisRrup, thisVs30,thisEventType, thisSubductionSlab)
        herePSHAB20MedianMinusNsigmaxSigmaTotal = math.exp(
           math.log(herePSHAB20MedianPSA) - thisNsigma * herePSHAB20SigmaTotal)
        herePSHAB20MedianPlusNsigmaxSigmaTotal = math.exp(
```

```
            math.log(herePSHAB20MedianPSA) + thisNsigma * herePSHAB20SigmaTotal)
        herePSHAB20MedianMinusNsigmaxSigmaAleatory = math.exp(
            math.log(herePSHAB20MedianPSA) - thisNsigma * herePSHAB20SigmaAleatory)
        herePSHAB20MedianPlusNsigmaxSigmaAleatory = math.exp(
            math.log(herePSHAB20MedianPSA) + thisNsigma * herePSHAB20SigmaAleatory)
        herePSHAB20MedianMinusNsigmaxSigmaEpistemic = math.exp(
            math.log(herePSHAB20MedianPSA) - thisNsigma * herePSHAB20SigmaEpistemic)
        herePSHAB20MedianPlusNsigmaxSigmaEpistemic = math.exp(
            math.log(herePSHAB20MedianPSA) + thisNsigma * herePSHAB20SigmaEpistemic)
        outSensitivityData.append(
            [thisCaseLabel, thisCaseLabelRegion, thisMagnitude, thisVs30, thisRegion, thisRrup, thisAlphaBackarc,
             thisAlphaNankai, thisZtor, thisSeatBasn, thisZ1pt0, thisZ2pt5, thisNsampleEpi, thisMb, thisZhypo,
             thisSubductionSlab, thisPNWbasinStrux, thisNsigma, thisEventType, thisT, hereKBCG20MedianPSA,
             hereKBCG20Tau, hereKBCG20Phi, hereKBCG20SigmaAleatory, hereKBCG20SigmaEpistemic,
hereKBCG20SigmaTotal,
             hereKBCG20MedianMinusNsigmaxSigmaTotal, hereKBCG20MedianPlusNsigmaxSigmaTotal,
             hereKBCG20MedianMinusNsigmaxSigmaAleatory, hereKBCG20MedianPlusNsigmaxSigmaAleatory,
             hereKBCG20MedianMinusNsigmaxSigmaEpistemic, hereKBCG20MedianPlusNsigmaxSigmaEpistemic,
             herePSHAB20MedianPSA, herePSHAB20Tau, herePSHAB20PhiTot, herePSHAB20PhiS2S, herePSHAB20PhiSS,
             herePSHAB20SigmaAleatory, herePSHAB20SigmaEpistemic, herePSHAB20SigmaTotal,
herePSHAB20MedianMinusNsigmaxSigmaTotal,
             herePSHAB20MedianPlusNsigmaxSigmaTotal, herePSHAB20MedianMinusNsigmaxSigmaAleatory,
             herePSHAB20MedianPlusNsigmaxSigmaAleatory, herePSHAB20MedianMinusNsigmaxSigmaEpistemic,
             herePSHAB20MedianPlusNsigmaxSigmaEpistemic])

    # end    # for irow
    # end    # for iset
    print('Sensitivity-Study Data:')
    outSensitivityTable = pd.DataFrame(outSensitivityData)
    outSensitivityTable.columns = ['CaseLabel', 'CaseLabelRegion', 'Mag', 'Vs30mps', 'Reg', 'Rrup_km', 'alpBack',
                                   'alpNank', 'Ztor_km', 'SeatBasn', 'Z1pt0m', 'Z2pt5m', 'NsplEpi', 'Mbreak',
                                   'Zhypo_km', 'SatReg', 'Basin', 'Nsigma', 'InterIntra', 'Period_sec',
                                   'KBCG20MedianPSA', 'KBCG20Tau', 'KBCG20Phi', 'KBCG20SigmaAleatory',
                                   'KBCG20SigmaEpistemic', 'KBCG20SigmaTotal', 'KBCG20MedianMinusNsigmaxSigmaTotal',
                                   'KBCG20MedianPlusNsigmaxSigmaTotal', 'KBCG20MedianMinusNsigmaxSigmaAleatory',
                                   'KBCG20MedianPlusNsigmaxSigmaAleatory', 'KBCG20MedianMinusNsigmaxSigmaEpistemic',
                                   'KBCG20MedianPlusNsigmaxSigmaEpistemic', 'PSHAB20MedianPSA', 'PSHAB20Tau',
                                   'PSHAB20PhiTot', 'PSHAB20PhiS2S', 'PSHAB20PhiSS', 'PSHAB20SigmaAleatory', 'PSHAB20SigmaEpistemic',
                                   'PSHAB20SigmaTotal', 'PSHAB20MedianMinusNsigmaxSigmaTotal',
                                   'PSHAB20MedianPlusNsigmaxSigmaTotal', 'PSHAB20MedianMinusNsigmaxSigmaAleatory',
                                   'PSHAB20MedianPlusNsigmaxSigmaAleatory',
                                   'PSHAB20MedianMinusNsigmaxSigmaEpistemic',
                                   'PSHAB20MedianPlusNsigmaxSigmaEpistemic']
    blankIndex = [''] * len(outSensitivityTable)
    outSensitivityTable.index = blankIndex
    print(outSensitivityTable)
    print('done!')


# end


# ###############################################################
# the following commands compute the output values at each period and generate a figure of the data
# The user may edit these lines, if needed
# ###############################################################
# Compute Values at each Period
def runScenarioStudy(User_Region, User_SubductionSlab, User_Magnitude, User_Vs30, User_Rrup, User_AlphaBackarc,
```

```
            User_AlphaNankai, User_Ztor, User_Zhypo, User_EventType, User_Z1pt0, User_Z2pt5, User_Mb,
            User_PNWbasinStrux, User_RelativeWeight_KBCG20, User_RelativeWeight_PSHAB20,
            User_EpiInSigmaModels, User_NsampleEpi, User_Nsigma, User_PeriodList):
  UserInputArray = [User_Region, User_SubductionSlab, User_Magnitude, User_Vs30, User_Rrup, User_AlphaBackarc,
             User_AlphaNankai, User_Ztor, User_Zhypo, User_EventType, User_Z1pt0, User_Z2pt5, User_Mb,
             User_PNWbasinStrux, User_RelativeWeight_KBCG20, User_RelativeWeight_PSHAB20,
             User_EpiInSigmaModels, User_NsampleEpi, User_Nsigma]
  outTlist = []
  outMedianList = []
  outSigmaList = []
  outMedianMinusNSigmaList = []
  outMedianPlusNSigmaList = []
  outWeightedAvgData = []
  outIndivModelData = []
  print(' ### Computing Spectra for Single Scenario')
  # print('Period_(s)','PSA_median_(g)','Sigma','PSA_median-N*Sigma_(g)','PSA_median+N*Sigma_(g)')
  for irow in range(len(User_PeriodList)):
    thisT = getTFvalue(User_PeriodList[irow])
    outTlist.append(thisT)

    PSA = NGAsubGMM_MedianPlusMinusSigma(thisT, User_Region, User_Magnitude, User_Vs30, User_Rrup,
User_AlphaBackarc,
                  User_AlphaNankai, User_Ztor, User_Zhypo, User_EventType, User_Z1pt0, User_Z2pt5, User_Mb,
                  User_SubductionSlab, User_PNWbasinStrux, User_RelativeWeight_KBCG20,
                  User_RelativeWeight_PSHAB20, 0,0,0)
    outMedianMinusNSigma = NGAsubGMM_MedianPlusMinusSigma(thisT, User_Region, User_Magnitude, User_Vs30,
User_Rrup, User_AlphaBackarc,
                  User_AlphaNankai, User_Ztor, User_Zhypo, User_EventType, User_Z1pt0, User_Z2pt5, User_Mb,
                  User_SubductionSlab, User_PNWbasinStrux, User_RelativeWeight_KBCG20,
                  User_RelativeWeight_PSHAB20, -1*User_Nsigma,User_EpiInSigmaModels,User_NsampleEpi)
    outMedianPlusNSigma = NGAsubGMM_MedianPlusMinusSigma(thisT, User_Region, User_Magnitude, User_Vs30,
User_Rrup, User_AlphaBackarc,
                  User_AlphaNankai, User_Ztor, User_Zhypo, User_EventType, User_Z1pt0, User_Z2pt5, User_Mb,
                  User_SubductionSlab, User_PNWbasinStrux, User_RelativeWeight_KBCG20,
                  User_RelativeWeight_PSHAB20, +1*User_Nsigma,User_EpiInSigmaModels,User_NsampleEpi)
    outWeightedAvgData.append([thisT, PSA, outMedianMinusNSigma, outMedianPlusNSigma])
    # print(thisT,PSA,Sigma,outMedianMinusNSigma,outMedianPlusNSigma)

    #### Individual-Model Data:
    Prog_MbDefault = getMbDefault("KBCG20", User_EventType, User_Region, User_SubductionSlab)
    if str(User_Mb).lower() == "default".lower():
      inUser_Mb = Prog_MbDefault
    else:
      inUser_Mb = User_Mb
    # end
    outKBCG20_MedianPSA = KBCG20_medPSA(thisT, User_Magnitude, User_Rrup, User_AlphaBackarc, User_AlphaNankai,
                        User_Ztor, User_EventType, User_Vs30, User_Z1pt0, User_Z2pt5, inUser_Mb,
                        User_Region, User_PNWbasinStrux)
    outKBCG20_Tau = KBCG20_sigmaTau(thisT)
    outKBCG20_Phi = KBCG20_sigmaPhi(thisT)
    outKBCG20_SigmaAleatory = KBCG20_sigmaAleatory(thisT)
    outKBCG20_SigmaEpistemic = KBCG20_SigmaEpistemic(thisT, User_Magnitude, User_Rrup, User_AlphaBackarc,
                              User_AlphaNankai, User_Ztor, User_EventType, User_Vs30,
                              User_Z1pt0, User_Z2pt5, inUser_Mb, User_Region,
                              User_NsampleEpi, User_PNWbasinStrux)
    outKBCG20_SigmaTotal = KBCG20_SigmaTotal(thisT, User_Magnitude, User_Rrup, User_AlphaBackarc,
User_AlphaNankai,
                          User_Ztor, User_EventType, User_Vs30, User_Z1pt0, User_Z2pt5,
```

```
                                inUser_Mb, User_Region, User_NsampleEpi, User_PNWbasinStrux)
    outKBCG20_Median_minus_1_SigmaTotal = math.exp(
      math.log(outKBCG20_MedianPSA) - User_Nsigma * outKBCG20_SigmaTotal)
    outKBCG20_Median_plus_1_SigmaTotal = math.exp(
      math.log(outKBCG20_MedianPSA) + User_Nsigma * outKBCG20_SigmaTotal)
    outKBCG20_Median_minus_1_SigmaAleatory = math.exp(
      math.log(outKBCG20_MedianPSA) - User_Nsigma * outKBCG20_SigmaAleatory)
    outKBCG20_Median_plus_1_SigmaAleatory = math.exp(
      math.log(outKBCG20_MedianPSA) + User_Nsigma * outKBCG20_SigmaAleatory)
    outKBCG20_Median_minus_1_SigmaEpistemic = math.exp(
      math.log(outKBCG20_MedianPSA) - User_Nsigma * outKBCG20_SigmaEpistemic)
    outKBCG20_Median_plus_1_SigmaEpistemic = math.exp(
      math.log(outKBCG20_MedianPSA) + User_Nsigma * outKBCG20_SigmaEpistemic)
    outPSHAB20_MedianPSA = PSHAB20_Median(User_EventType, User_Region, User_SubductionSlab, User_Rrup,
                          User_Magnitude, User_Zhypo, thisT, User_Vs30, User_Z2pt5,
                          User_PNWbasinStrux)
    outPSHAB20_Tau = PSHAB20_Tau(thisT, User_Rrup, User_Vs30)
    outPSHAB20_PhiTot = PSHAB20_PhiTot(thisT, User_Rrup, User_Vs30)
    outPSHAB20_PhiS2S = PSHAB20_SigmaS2S(thisT, User_Rrup, User_Vs30)
    outPSHAB20_PhiSS = PSHAB20_SigmaSS(thisT, User_Rrup, User_Vs30)
    outPSHAB20_SigmaAleatory = PSHAB20_SigmaAleatory(thisT, User_Rrup, User_Vs30)
    outPSHAB20_SigmaEpistemic = PSHAB20_SigmaEpistemic(thisT, User_EventType, User_SubductionSlab)
    outPSHAB20_SigmaTotal = PSHAB20_SigmaTotal(thisT, User_Rrup,  User_Vs30, User_EventType, User_SubductionSlab )
    outPSHAB20_Median_minus_1_SigmaTotal = math.exp(
      math.log(outPSHAB20_MedianPSA) - User_Nsigma * outPSHAB20_SigmaTotal)
    outPSHAB20_Median_plus_1_SigmaTotal = math.exp(
      math.log(outPSHAB20_MedianPSA) + User_Nsigma * outPSHAB20_SigmaTotal)
    outPSHAB20_Median_minus_1_SigmaAleatory = math.exp(
      math.log(outPSHAB20_MedianPSA) - User_Nsigma * outPSHAB20_SigmaAleatory)
    outPSHAB20_Median_plus_1_SigmaAleatory = math.exp(
      math.log(outPSHAB20_MedianPSA) + User_Nsigma * outPSHAB20_SigmaAleatory)
    outPSHAB20_Median_minus_1_SigmaEpistemic = math.exp(
      math.log(outPSHAB20_MedianPSA) - User_Nsigma * outKBCG20_SigmaEpistemic)
    outPSHAB20_Median_plus_1_SigmaEpistemic = math.exp(
      math.log(outPSHAB20_MedianPSA) + User_Nsigma * outKBCG20_SigmaEpistemic)
    outIndivModelData.append([thisT, outKBCG20_MedianPSA, outKBCG20_Tau, outKBCG20_Phi,
outKBCG20_SigmaAleatory,
                      outKBCG20_SigmaEpistemic, outKBCG20_SigmaTotal, outKBCG20_Median_minus_1_SigmaTotal,
                      outKBCG20_Median_plus_1_SigmaTotal, outKBCG20_Median_minus_1_SigmaAleatory,
                      outKBCG20_Median_plus_1_SigmaAleatory, outKBCG20_Median_minus_1_SigmaEpistemic,
                      outKBCG20_Median_plus_1_SigmaEpistemic, outPSHAB20_MedianPSA, outPSHAB20_Tau,
                      outPSHAB20_PhiTot, outPSHAB20_PhiS2S, outPSHAB20_PhiSS, outPSHAB20_SigmaAleatory,
outPSHAB20_SigmaEpistemic,
                      outPSHAB20_SigmaTotal, outPSHAB20_Median_minus_1_SigmaTotal,
                      outPSHAB20_Median_plus_1_SigmaTotal, outPSHAB20_Median_minus_1_SigmaAleatory,
                      outPSHAB20_Median_plus_1_SigmaAleatory, outPSHAB20_Median_minus_1_SigmaEpistemic,
                      outPSHAB20_Median_plus_1_SigmaEpistemic])
  # end

  print('Scenario Input Data:')
  UserInputDataRows = ['Region', 'SubductionSlab', 'Magnitude', 'Vs30', 'Rrup', 'AlphaBackarc', 'AlphaNankai', 'Ztor',
              'Zhypo', 'EventType', 'Z1pt0', 'Z2pt5', 'Mb', 'PNWbasinStrux', 'RelativeWeight_KBCG20',
              'RelativeWeight_PSHAB20', 'EpiInSigmaModels', 'NsampleEpi', 'Nsigma']
  UserInputTable = pd.DataFrame(UserInputArray, index=UserInputDataRows)
  UserInputTable.columns = ['User Input']
  UserInputTable.index = UserInputDataRows
  print(UserInputTable)
  print('Scenario Weighted-Average Data:')
```

```
   outWeightedAvgTable = pd.DataFrame(outWeightedAvgData)
   outWeightedAvgTable.columns = ['Period_(s)', 'PSA_median_(g)', 'PSA_median-N*Sigma_(g)',
                          'PSA_median+N*Sigma_(g)']
   blankIndex = [''] * len(outWeightedAvgTable)
   outWeightedAvgTable.index = blankIndex
   print(outWeightedAvgTable)
   print('Scenario Individual-Model Data:')
   outIndivModelTable = pd.DataFrame(outIndivModelData)
   outIndivModelTable.columns = ['Period_s', 'KBCG20_MedianPSA', 'KBCG20_Tau', 'KBCG20_Phi', 'KBCG20_SigmaAleatory',
                         'KBCG20_SigmaEpistemic', 'KBCG20_SigmaTotal', 'KBCG20_Median_minus_1*SigmaTotal',
                         'KBCG20_Median_plus_1*SigmaTotal', 'KBCG20_Median_minus_1*SigmaAleatory',
                         'KBCG20_Median_plus_1*SigmaAleatory', 'KBCG20_Median_minus_1*SigmaEpistemic',
                         'KBCG20_Median_plus_1*SigmaEpistemic', 'PSHAB20_MedianPSA', 'PSHAB20_Tau',
                         'PSHAB20_PhiTot', 'PSHAB20_PhiS2S', 'PSHAB20_PhiSS', 'PSHAB20_SigmaAleatory',
'PSHAB20_SigmaEpistemic',
                         'PSHAB20_SigmaTotal', 'PSHAB20_Median_minus_1*SigmaTotal',
                         'PSHAB20_Median_plus_1*SigmaTotal', 'PSHAB20_Median_minus_1*SigmaAleatory',
                         'PSHAB20_Median_plus_1*SigmaAleatory', 'PSHAB20_Median_minus_1*SigmaEpistemic',
                         'PSHAB20_Median_plus_1*SigmaEpistemic']
   blankIndex = [''] * len(outIndivModelTable)
   outIndivModelTable.index = blankIndex
   print(outIndivModelTable)
   print('done!')


# end

###################### run user input
# The following optional command calls the Sensitivity Study.
# This study runs through a variety of scenarios and may take time to run

runScenarioStudy(User_Region, User_SubductionSlab, User_Magnitude, User_Vs30, User_Rrup, User_AlphaBackarc,
              User_AlphaNankai, User_Ztor, User_Zhypo, User_EventType, User_Z1pt0, User_Z2pt5, User_Mb,
              User_PNWbasinStrux, User_RelativeWeight_KBCG20, User_RelativeWeight_PSHAB20,
              User_EpiInSigmaModels, User_NsampleEpi, User_Nsigma, User_PeriodList)

if RunSensitivityStudyFlag.lower() == "yes".lower():
   RunSensitivityStudy()
# end
```

6. References

Goulet, C., Bozorgnia, Y., Abrahamson, N., Kuehn, N., Al Atik, L., Youngs, R., and Graves, R. (2018). *Central and Eastern North America Ground-Motion Characterization - NGA-East Final Report.* PEER Report 2019/08, Pacific Earthquake Engineering Research Center, University of California.

Kuehn, N., Bozorgnia, Y., Campbell, K., and Gregor, N. (2020), KBCG20. *Partially Nonergodic Ground-Motion Model for Subduction Regions using NGA-Subduction Database.* PEER Report 2020/04, Pacific Earthquake Engineering Research Center, University of California.

Parker, G., Stewart, J., Hassani, B., Atkinson, G., and Boore, D. (2020), PSHAB20. *NGA-Subduction Global Ground Motion Models with Regional Adjustment Factors*. PEER Report 2020/03, Pacific Earthquake Engineering Research Center, University of California.

Electronic Attachment A: Computer codes and data needed for NGA-East GMMs Tool

Electronic Attachment B: Computer codes and data needed for NGA-Sub GMMs Tool