“OpenQuake: Calculate, share, explore”

Ground Motion Prediction Equation (GMPE) and Strong Motion Modeller’s Tookit - User Guide

PUBLISHED BY GEM FOUNDATION

GLOBALQUAKEMODEL.ORG/OPENQUAKE

Disclaimer

The “Ground Motion Prediction Equation (GMPE) and Strong Motion Modeller’s Tookit - User Guide” is distributed in the hope that it will be useful, but without any warranty: without even the implied warranty of merchantability or fitness for a particular purpose. While every precaution has been taken in the preparation of this document, in no event shall the authors of the manual and the GEM Foundation be liable to any party for direct, indirect, special, incidental, or consequential damages, including lost profits, arising out of the use of information contained in this document or from the use of programs and source code that may accompany it, even if the authors and GEM Foundation have been advised of the possibility of such damage. The Book provided hereunder is on as "as is" basis, and the authors and GEM Foundation have no obligations to provide maintenance, support, updates, enhancements, or modifications.

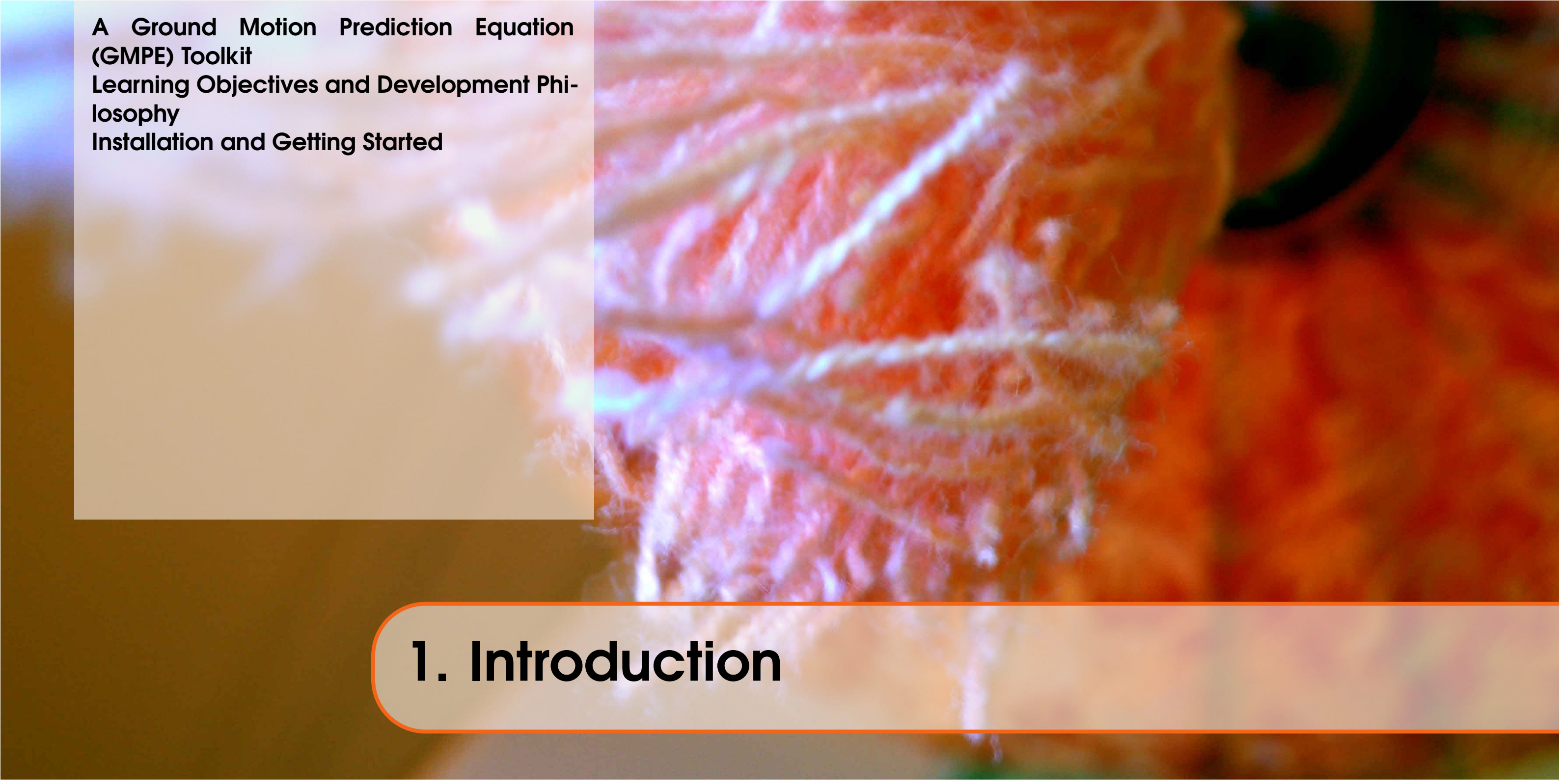
The current version of the book has been revised only by members of the GEM model facility and it must be considered a draft copy.

eeds to be up-
or removed

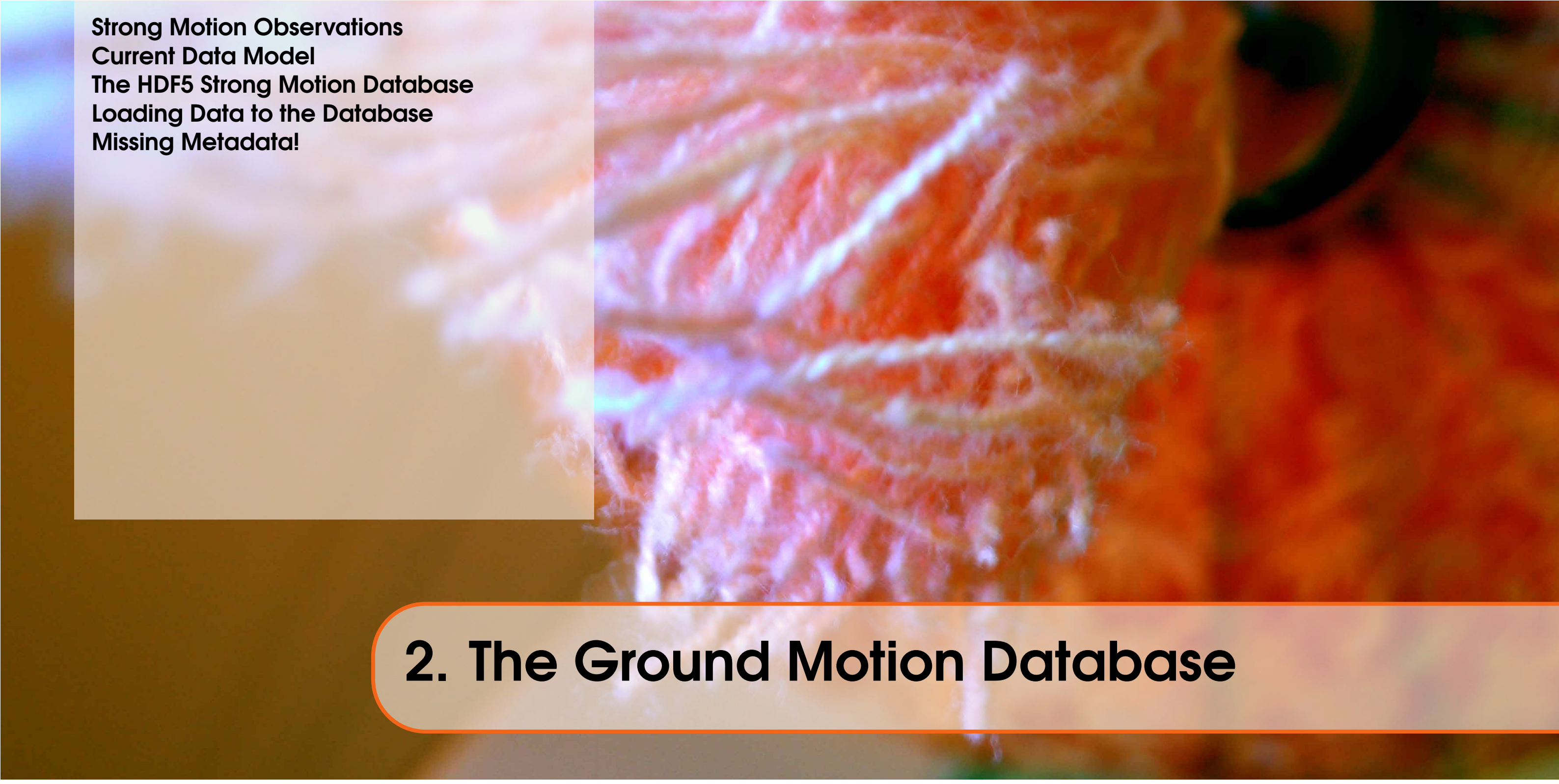
License

First printing, June 2014

Contents



1. Introduction



2. The Ground Motion Database

3. Response Spectra and Intensity Measures

3.1 Ground Motion Waveforms

Fundamental to the use of the GMPE-SMTK is the strong motion waveform. For the widest portability we adopt the simplest representation of the waveform in the toolkit, which requires only the acceleration and the time steps. As seen in the previous chapter, the strong motion database stores the acceleration, velocity and displacement trace.

The conventional units for acceleration, velocity and displacement waveforms in the GMPE-SMTK are "cm/s/s", "cm/s" and "cm" respectively.

The GMPE-SMTK contains two tools for extracting basic information and common ground motion intensity measures from single waveforms, or from a horizontal pair of waveforms: `intensity_measures` and `response_spectrum`. To use them in an application we can import them as follows:

```python
import smtk.intensity_measures as ims
import smtk.response_spectrum as rsp
```

This will load two objects into the workspace `ims` (the intensity measure tools) and `rsp` (the response spectrum tools)

In most applications we assume it is the acceleration time series that is available. In the following example we consider a pair of strong motion records, where each record is represented by a simple ascii (text) file:`sm_record_x.txt` and `sm_record_y.txt`. The timestep of each record is 0.002 s.

```python
# Import the numpy tools
import numpy as np
# Load in the records
accel_x = np.genfromtxt("sm_record_x.txt")
accel_y = np.genfromtxt("sm_record_y.txt")
time_step = 0.002
```

Once loaded the user can retrieve the velocity and displacement time-series, calculated using double integration of the acceleration time series. A simple tool to do this is found in the `ims` tools:

```python
vel_x, disp_x = ims.get_velocity_displacement(accel_x,
                                              time_step,
                                              units="cm/s/s")
```

where `units` is the units of the uploaded record.
The first steps one might wish to take when analysing a strong motion record is to look at the waveforms. To do this we make use of the simple tool inside the response spectrum package called `plot_time_series`. This can be called as follows for the x=component of the pair:

```
rsp.plot_time_series(accel_x,
                     time_step,
                     velocity=vel_x,
                     displacement=disp_x,
                     units="cm/s/s",
                     filename="path/to/output/image.eps",
                     filetype="eps",
                     dpi=300,
                     linewidth=1.5)
```

This command would produce a plot of the style shown in Figure 3.1.

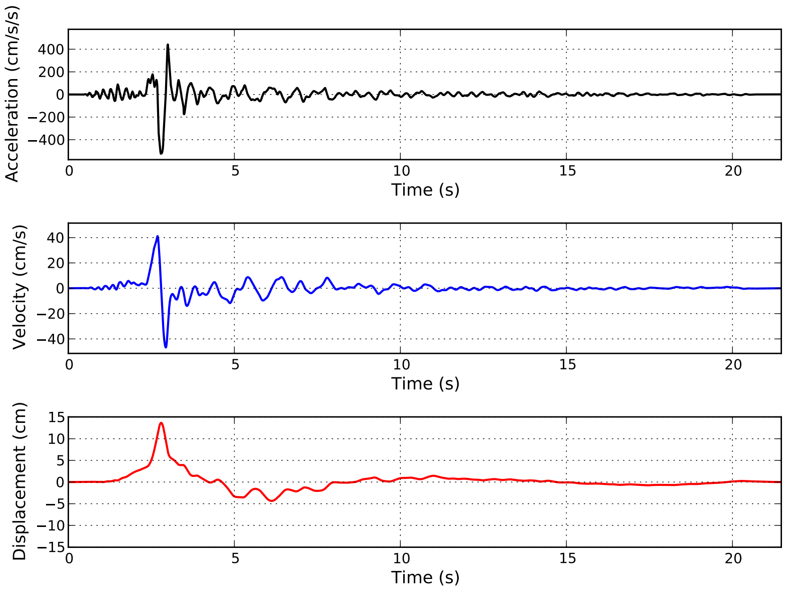


Figure 3.1 – *Example time series for a single record: acceleration (top), velocity (middle) and displacement (bottom)*

The function `plot_time_series` requires two essential arguments:

- `acceleration`: The acceleration time series
- `time_step`: The time step of the acceleration time series

If the user already has the velocity and displacement time series available these can be input with the following keyword arguments:

- `velocity`: The velocity time series (cm/s), which will be calculated if not provided.
- `displacement`: The time step of the displacement (cm) time series, which will be calculated if not provided.
- `units`: The units of the acceleration time series (defaults to “cm/s/s”)
- `figure_size`: The internal size of the figure (defaults to $(8,6)$)
- `filename`: The name of the file to which the figure will be saved (if desired, by default it is “None”)
- `filetype`: The format of the output file (default=“png”).
- `dpi`: Resolution (dots per inch) of the output figure (default = 300)
- `linewidth`: The thickness (point size) of the lines (default = 1.5)

3.2 Scalar Intensity Measures

3.2.1 Peak Measures

The following describes the simple "peak" measures, peak ground acceleration (PGA), peak ground velocity (PGV) and peak ground displacement (PGD), that can be extracted from the record:

$$\begin{aligned} PGA &= \max\left|a\left(t\right)\right| \\ PGV &= \max\left|v\left(t\right)\right| \\ PGD &= \max\left|d\left(t\right)\right| \end{aligned} \tag{3.1}$$

where $a\left(t\right)$, $v\left(t\right)$ and $d\left(t\right)$ are the acceleration, velocity and displacement time series respectively.

These values can all be extracted by one function in the `ims` tool named `get_peak_measures`. These values can be called as follows:

```
pga, pgv, pgd, vel_x, disp_x = get_peak_measures(time_step,
                                                 accel_x,
                                                 get_vel=True,
                                                 get_disp=True)
```

This function returns the three peak measures as well as the velocity and displacement (if required). The keywords `get_vel` and `get_disp` will, when set to `True`, calculate velocity and displacement.

3.2.2 Duration

3.2.3 Other Scalar Measures

3.3 Response Spectra

The response spectrum is one of the most important elements of a ground motion record used in earthquake engineering. The response spectrum provides the peak responses of a set a single-degree-of-freedom (SDOF) oscillators with different natural periods (T), and damping ratio ξ. The pseudospectral acceleration at a given period $Sa\left(T,\xi\right)$, and its corresponding pseudospectral velocity and displacements, $Sv\left(T,\xi\right)$ and $Sd\left(T,\xi\right)$ respectively, are the most widely used means to characterise the ground motion input for a given structure. In addition to the peak values however, it is also useful to have available the acceleration, velocity and displacement response time series of the SDOF oscillators for each natural period. The GMPE-SMTK response spectra tools are designed to return the most comprehensive set of information to describe the SDOF response for each record.

To calculate a response spectrum for a record, two response spectra calculators are currently available: "Newmark-β" and "Nigam & Jennings" (CITE NigamJennings1969).

These can be called following the process below:

```
# Define the periods for calculating spectral acceleration
periods = np.array([0.01, 0.02, 0.03, 0.04, 0.05, 0.075, 0.1,
                    0.11, 0.12, 0.13, 0.14, 0.15, 0.16, 0.17,
                    0.18, 0.19, 0.20, 0.22, 0.24, 0.26, 0.28,
                    0.30, 0.32, 0.34, 0.36, 0.38, 0.40, 0.42,
                    0.44, 0.46, 0.48, 0.50, 0.55, 0.6-, 0.65,
                    0.70, 0.75, 0.80, 0.85, 0.90, 0.95, 1.00,
                    1.10, 1.20, 1.30, 1.40, 1.50, 1.60, 1.70,
                    1.80, 1.90, 2.00, 2.20, 2.40, 2.60, 2.80,
                    3.00, 3.20, 3.40, 3.60, 3.80, 4.00, 4.20,
                    4.40, 4.60, 4.80, 5.00, 5.50, 6.00, 6.50,
                    7.00, 7.50, 8.00, 8.50, 9.00, 9.50, 10.0])
```

```python
# Call the Newmark-Beta methodology
newmark_beta = rsp.NewmarkBeta(acceleration,
                               time_step,
                               periods,
                               damping=0.05,
                               units="cm/s/s")

spectra, time_series, acc, vel, dis = newmark_beta.evaluate()

# Call the Nigam & Jennings (1969) method
nigam_jennings = rsp.NigamJennings(acceleration,
                                   time_step,
                                   periods,
                                   damping=0.05,
                                   units="cm/s/s")

spectra, time_series, acc, vel, dis = nigam_jennings.evaluate()
```

Each response spectrum method requires the following input:

- `acceleration`: The acceleration time-series
- `time_step`: The time step (s)
- `periods`: An array of natural periods
- `damping`: The fractional damping ratio of the oscillator (defaults to 0.05 if not specified)
- `units`: The units of the acceleration time-series (defaults to “cm/s/s”)

The calculators produce five outputs:

1. `spectra`: A dictionary with the following keys:
    - `Period`: The vector or spectral periods
    - `Acceleration`: The peak acceleration response at each period (cm/s/s)
    - `Velocity`: The peak velocity response at each period (cm/s)
    - `Displacement`: The peak displacement response at each period (cm)
    - `Pseudo-Acceleration`: The peak pseudo-acceleration response at each period (cm/s/s), where pseudo-acceleration is defined as:

      $$PSa\left(T,\xi\right) = \frac{4\pi^2}{T^2} Sd\left(T,\xi\right) \tag{3.2}$$

    - `Pseudo-Velocity`: The peak pseudo-velocity response at each period (cm/s/s), where pseudo-velocity is defined as:

      $$PSv\left(T,\xi\right) = \frac{2\pi}{T} Sd\left(T,\xi\right) \tag{3.3}$$

2. `time_series`: A dictionary with the following items:
    - `Time-Step`: The time step of the record
    - `Acceleration`: The acceleration time-series (cm/s/s) of the original record
    - `Velocity`: The velocity time-series (cm/s) of the original record
    - `Displacement`: The displacement time-series (cm) of the original record
    - `PGA`: The peak ground acceleration of the record (cm/s/s)
    - `PGV`: The peak ground velocity of the record (cm/s)
    - `PGD`: The peak ground displacement of the record (cm)
3. `acc`: A 2D array in which each column contains the acceleration time-series of the SDOF oscillator response to the record for each period.
4. `vel`: A 2D array in which each column contains the velocity time-series of the SDOF oscillator response to the record for each period.

5. dis: A 2D array in which each column contains the displacement time-series of the SDOF oscillator response to the record for each period.

If, alternatively, one does not wish to import the response spectra methods manually then the ims tools have a function to apply this process. The same results as those shown previously can be obtained from this tool as follows:

```
spectra, time_series, acc, vel, dis = ims.get_response_spectrum(
    accel_x,
    time_step,
    periods,
    damping=0.05,
    units="cm/s/s",
    method="Nigam-Jennings")
```

The inputs are the same as for the response spectrum calculators except for method, which indicates the preferred method for calculating the response spectrum. At present this is either Newmark-Beta or Nigam-Jennings, which the latter selected as the default.

To view the full set of resulting response spectra, the rsp tool has a method names plot_response_spectra, which is used as follows and will produce a plot similar to that of Figures 3.2 and 3.3:

```
rsp.plot_response_spectra(spectra,
                          axis_type="loglog")
```

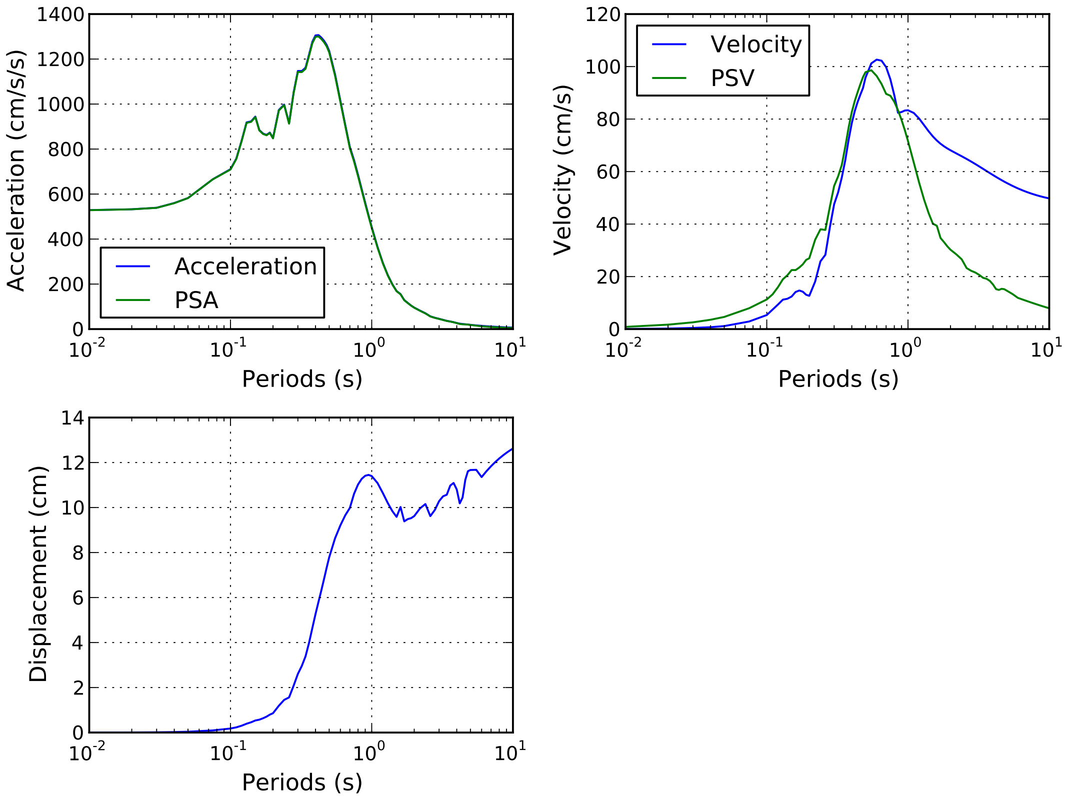


Figure 3.2 – *Full response spectra of the waveform shown in Figure 3.1, calculated using the Newmark-β method*

where spectra is the response spectra dictionary, axis_type defines the type of axes: “loglog” (double logarithmic - the default), “semilogx” (logarithmic period axis), “semilogy” (logarithmic

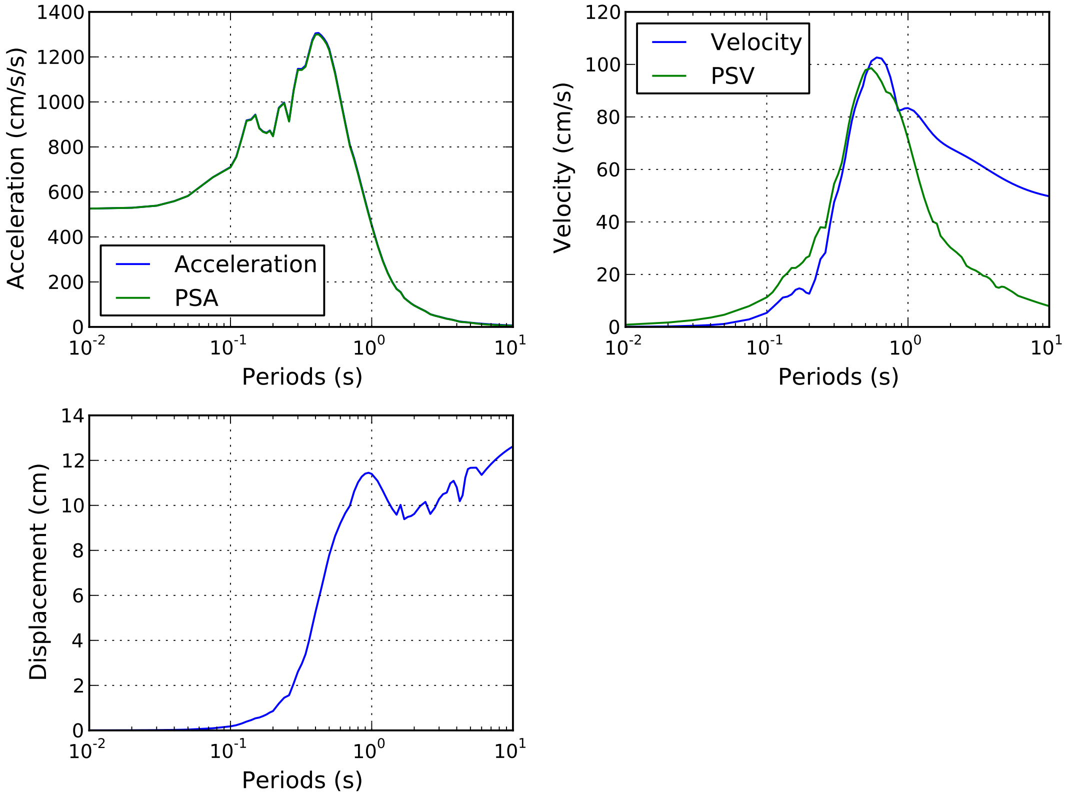


Figure 3.3 – *Full response spectra of the waveform shown in Figure 3.1, calculated using the Nigam & Jennings (1969) method*

spectral response axis), "linear" (both axes linear). The function also takes the same keywords as the `plot_time_series` method, i.e. `figure_size`, `filename`, `filetype`, `dpi`.

It can be seen from Figures 3.2 and 3.3 that both methods give very similar results.

3.4 Combining Horizontal Components of Motion

For a pair of horizontal records it is necessary to define a spectrum representing the "horizontal" component of ground motion. The manner in which the two horizontal components of motion are resolved can be treated in many different ways (cite: Douglas, 2003, ????; Beyer and Bommer etc.). The intensity measure tools contain various methods for combining horizontal response spectra.

The first step in the application of these methods may be to extract the respective response spectra for the two horizontal components. This can be done as follows:

```
spectra_x, spectra_y = ims.get_response_spectrum_pair(
    accel_x,
    time_step, # Time-step of x-component
    accel_y,
    time_step, # Time-step of y-component,
    periods,
    damping=0.05,
    units="cm/s/s",
    method="Nigam-Jennings")
```

In the case of the records demonstrated the time-step is the same for both horizontal components. If it is not the same, however, the response spectra can still be calculated.

Given the pair of records, spectra_x ($Sa_x(T,\xi)$)and spectra_y ($Sa_y(T,\xi)$), we can now obtain the following "resolved" horizontal spectra:

- **Geometric Mean**

$$Sa_{gm}(T,\xi) = \sqrt{Sa_x(T,\xi) \times Sa_y(T,\xi)} \tag{3.4}$$

Calculated using the following command:

```
sa_gm = ims.geometric_mean_spectrum(spectra_x, spectra_y)
```

- **Arithmetic Mean**

$$Sa_{am}(T,\xi) = \frac{1}{2}\left(Sa_x(T,\xi) + Sa_y(T,\xi)\right) \tag{3.5}$$

Calculated using the following command:

```
sa_am = ims.arithmetic_mean_spectrum(spectra_x, spectra_y)
```

- **Larger PGA** This simply returns the spectrum of the time series that gives the largest spectral acceleration. Calculated using the following command:

```
sa_larger = ims.larger_pga(spectra_x, spectra_y)
```

- **Envelope** This returns a spectrum representing the larger of the two components for each period such that:

$$Sa_{env}(T_i,\xi) = \max\left(Sa_x(T_i,\xi), Sa_y(T_i,\xi)\right) \quad \text{for} \quad i = 1,2,\ldots,N_{PERIODS} \tag{3.6}$$

```
sa_env = ims.envelope(spectra_x, spectra_y)
```

In each of these cases the output of the function is a dictionary containing the same keys as that of the spectra, but with the resolved horizontal values of each of the quantities.

For the two spectra considered here, the geometric mean spectra and the envelope spectra are shown in Figures 3.4 and 3.5 respectively.

3.5 Rotation of the Horizontal Records

In some applications it may be necessary to rotate a pair of horizontal time series, either to orient them into fault-normal and fault-parallel components, or to some direction that may represent the most adverse for a particular structure. For two time series of equal duration and time series, the records can be rotated through angle θ:

$$\begin{pmatrix} a_{x(\theta)}(t) \\ a_{y(\theta)}(t) \end{pmatrix} = \begin{bmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{bmatrix} \begin{pmatrix} a_x(t) \\ a_y(t) \end{pmatrix} \tag{3.7}$$

This can be achieved using the ims tools:

```
rot_hist_x, rot_hist_y = ims.rotate_horizontal(accel_x,
                                               accel_y,
                                               theta)
```

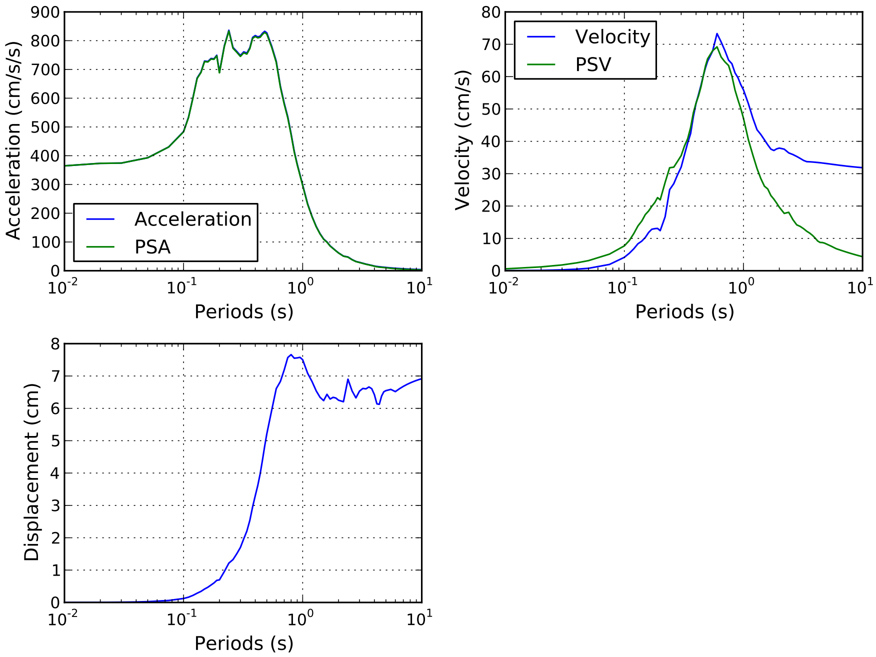


Figure 3.4 – *Geometric mean spectra of the two horizontal records*

The incorporation of rotational tools permits the toolkit to calculate the "orientation-dependent" and "orientation-independent" geometric mean of a horizontal pair of records, as described in detail in CITE Boore et al (2006). The two quantities, `GMRotDpp` and `GMRotIpp` respectively, cannot be calculated from the two horizontal response spectra directly as other horizontal measures are. As they require rotation of time-series through non redundant angles, it is necessary to determine the response spectra for each rotation angle.

To calculate `GMRotDpp` and `GMRotIpp` run:

```
gmrotd50 = ims.gmrotdpp(accel_x, time_step,
                        accel_y, time_step,
                        periods,
                        50.0, # Percentile
                        damping=0.05,
                        units="cm/s/s",
                        method="Nigam-Jennings")
gmroti50 = ims.gmrotipp(accel_x, time_step,
                        accel_y, time_step,
                        periods,
                        50.0, # Percentile
                        damping=0.05,
                        units="cm/s/s",
                        method="Nigam-Jennings")
```

3.6 Adding Horizontal Components to the Database

In the process of constructing the database of strong motion records it may often be prudent to compile the database in steps, of which the final one may often be the addition of the resolved horizontal components of the strong motion record. Depending on the data, and the required intensity measures, the calculation of horizontal spectra may be a (relatively) slow process with respect to the rest of the database construction. It is recommended that the database be

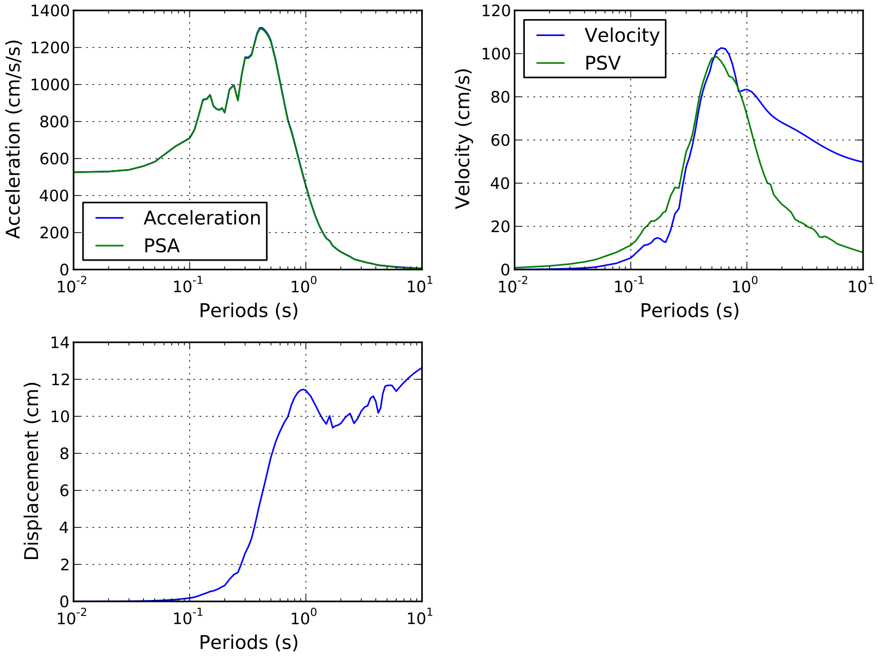
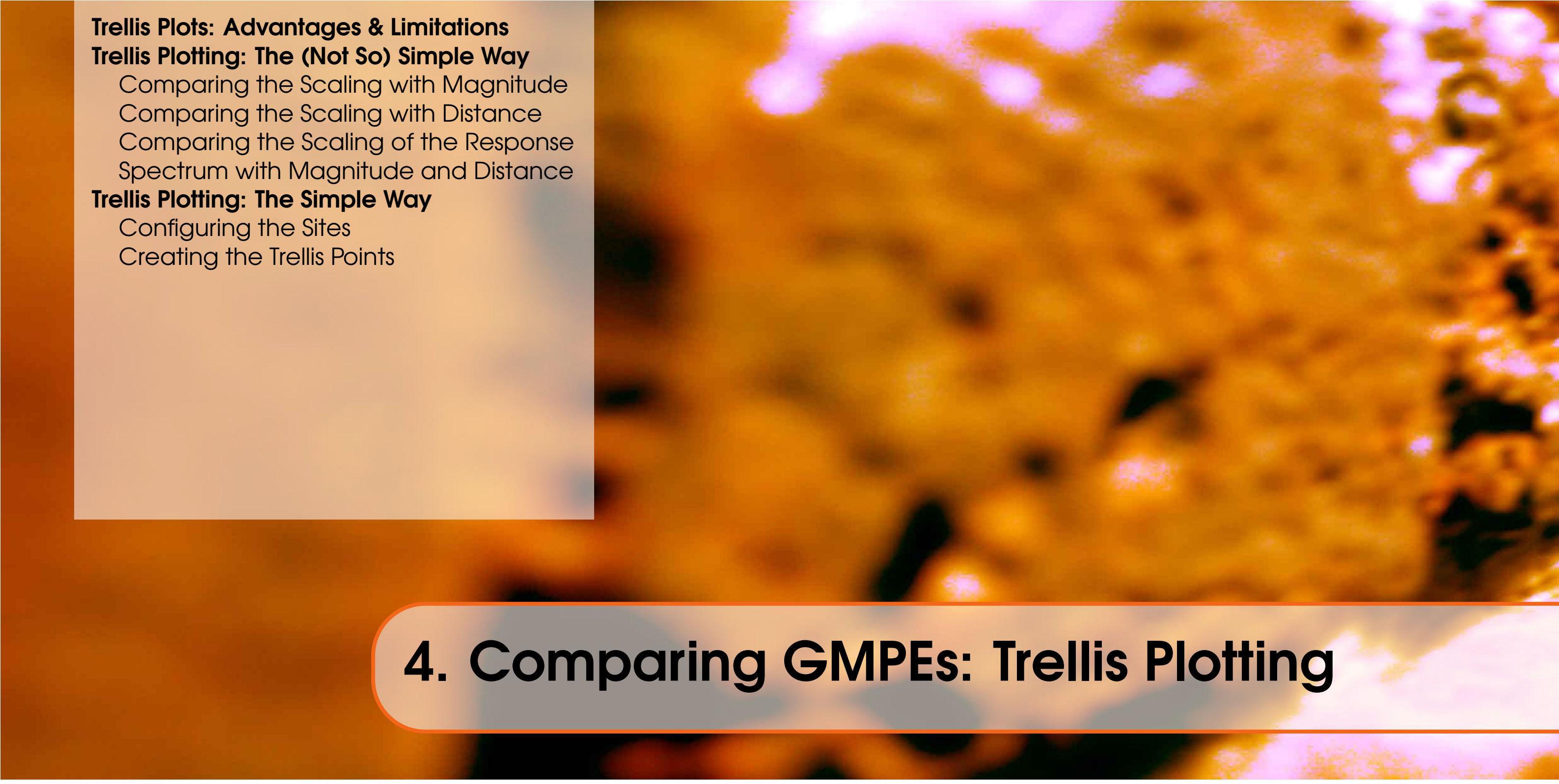


Figure 3.5 – *Envelope spectra of the two horizontal records*

constructed in the manner described in section 2.4, and then the resolved horizontal components added subsequently. To facilitate this a function is added to the database builder, which will add the desired horizontal ground motions to an existing database. An example is seen below:

```
# Import the function add_horizontal_im
from smtk.sm_database_buider import add_horizontal_im
# Load in the database metadata file
import cPickle
database = cPickle.load(open("path/to/metadata.pkl", "r"))
# Choose the desired horizontal intensity measures
# The following will add the horizontal PGA, PGV
# Geometric mean spectrum, envelope spectrum and GMRotI50
im_list = ["PGA", "PGV", "Geometric", "Envelope", "GMRotI50"]
add_horizontal_im(database,
                  im_list,
                  component="Geometric",
                  damping="05",
                  periods=[])
```

Depending on the size of the database this may take **many hours** to run (especially if rotational parameters are required). In this function the keyword `component` refers only to the horizontal component of the scalar values. For the spectra, the type of horizontal component should be given in the intensity measure list. Note also that unlike the `ims` and `rsp` tools in which the damping is input as a floating point value, here it is input as a string. If not input by the user, the periods will be taken from the values stored in the x-component of the record.



4. Comparing GMPEs: Trellis Plotting

4.1 Trellis Plots: Advantages & Limitations

The selection of GMPEs for use in probabilistic seismic hazard analysis should require an understanding of the manner in which each GMPE characterises the ground motion scaling with respect to the properties of the seismic source, and the attenuation of the motion with respect to distance. This information can help in the interpretation of the seismic hazard results, particularly those related to disaggregation, in order to best understand how the ground motion prediction equation can influence the seismic hazard at a site.

Trellis plots can be a useful tool in this process as they permit the hazard modeller to compare multiple GMPEs under a variety of conditions and understand how the GMPEs differ in terms of their fundamental source, path and site characteristics. The GMPE-SMTK provides a set of tools to provide the modeller the ability to compare OpenQuakes GMPE implementations under a variety of conditions.

One of the challenges in developing trellis plots is to compare, in a quantitatively meaningful sense, GMPEs that require different characteristics of the source, path and site. One of the most common areas of divergence can be in terms of the metric use to measure source to site distance. Many modern GMPEs for active shallow crustal regions adopt Joyner-Boore distance (shortest distance to the surface projection of the rupture) as the predictor variable for describing attenuation with distance, whilst others adopt rupture distance (the shortest distance from the site to the rupture surface), and several now adopt more complex forms that require characterisation of several types of source to site metric in order to capture the effects of directivity and hanging-wall scaling of the ground motion. This can be complicated further when we consider that the relation between source and distances metrics will vary depending on the scaling of the rupture with magnitude and the geometric orientation of the rupture. Typical trellis plots, whilst commonly used in studies of probabilistic seismic hazard, rarely seem to explicitly consider the differences in the relation between different distance metrics, except via the use of empirical relations, and their dependence on the physical characteristics of the source and site configuration.

The GMPE-SMTK attempts to address this issue by providing two contexts in which to compare GMPEs. The first is a "non-specific rupture" context in which the physical dimensions of the rupture are not explicitly considered. In this case, the user must specify the precise distances *a priori*, thus requiring them to determine the geometrical relationship between the metrics by other means. The second type is a "specific rupture" context, in which the user provides sufficient information in order to define a planar rupture with dimensions consistent with the magnitude of the earthquake. From this rupture the GMPE-SMTK uses OpenQuake's own geometry functions to calculate all the required source-to-site distances with respect to the rupture plane. This ensures that not only is the implicit relation between different attenuation metrics accurately defined, it also guarantees that the calculation is done consistently between the trellis plotting tools and the PSHA software that may be ultimately used to implement the GMPEs being compared.

The trellis plotting tools are all contained in the function `smtk.trellis.trellis_plots`, which we shall import as follows:

```
import smtk.trellis.trellis_plots as trpl
```

and which we will refer to as `trpl` hereafter.

If the Openquake-hazard library is installed in your computer, the full list of available GMPEs can be retrieved in the following manner:

```
# Import the get_available_gsims function from OpenQuake
from openquake.hazardlib.gsim import get_available_gsims
# Show list of gsims
get_available_gsims()
```

In the examples shown throughout this chapter we consider six GMPES: citeAkkarBommer2010, citeAkkarCagnan2010, site Akkar2014 (Joyner-Boore coefficients), citeBooreAtkinson2008, citeChiouYoungs2008, citeZhaoEtAl2006. We also compare using just four intensity measures: PGA, $Sa(0.2s)$, $Sa(1.0s)$ and $Sa(2.0s)$.

4.2 Trellis Plotting: The (Not So) Simple Way

The non-specific rupture context requires no additional tools besides the `trpl` function. However, it is necessary that the user specifies, manually, the distance metrics, and other rupture information that may be absent.

This could be done as follows:

```
gmpe_list = ["AkkarBommer2010",
             "AkkarCagnan2010",
             "AkkarEtAlRjb2014",
             "BooreAtkinson2008",
             "ChiouYoungs2008",
             "ZhaoEtAl2006Asc"]

imts = ["PGA", "SA(0.2)", "SA(1.0)", "SA(2.0)"]
params = {"ztor": 5.0,   # Top of rupture depth
          "hypo_depth": 10.0,   # Hypocentral depth
          "vs30": 800.0, # Vs30 for all sites
          "vs30measured": True, # Vs30 value is measured
          "z1pt0": 100.0, # Depth (m) to the 1.0 km/s Vs interface
          "dip": 90.0,  # Vertical Fault
          "rake": 0.0 # Strike-slip fault
          }
```

The GMPEs and intensity measures must be input as a list, whilst missing parameters a grouped together in a Python dictionary.

4.2.1 Comparing the Scaling with Magnitude

The first trellis tool simply shows how the ground motion intensity value scales with respect to the magnitude of the rupture. For the above GMPEs we consider magnitudes from M_W 4.5 to

M_W 8.0, every 0.1 magnitude units. We also assume that the ground motions correspond to a site located at a Joyner-Boore distance of 15 km away from the vertical dipping fault (with a top of rupture depth of 5 km). We assume that the epicentral distance is equal to 2 km. This configuration is specified like so:

```python
# Import the numerical python tool
import numpy as np
# Generate magnitudes from 4.5 to 8.0 every 0.1
magnitudes = np.arange(4.5, 8.1, 0.1)
distances = {"repi": 20.0,
             "rhypo": 22.5,
             "rjb": 15.0,
             "rrup": 16.0,
             "rx": 15.0}
```

Distances must be specified in a Python dictionary containing one or more of the following keys: `repi` (Epicentral Distance), `rhypo` (hypocentral distance), `rjb` (Joyner-Boore Distance), `rrup` (Rupture distance) and `rx` (Rx distance). Whilst not all distance metrics may be needed the tools with check the selected GMPEs and the distances dictionary provided. If any distance metric required by any of the GMPEs is not found in the distances input then an error will be raised.

The trellis plots can be generated using the following command, which will produce the plot shown in Figure 4.1

```python
trpl.MagnitudeIMTTrellis(magnitudes,
                         distances,
                         gmpe_list,
                         imts,
                         params,
                         figure_size=(7,5),
                         filename="path/to/plot",
                         filetype="png")
```

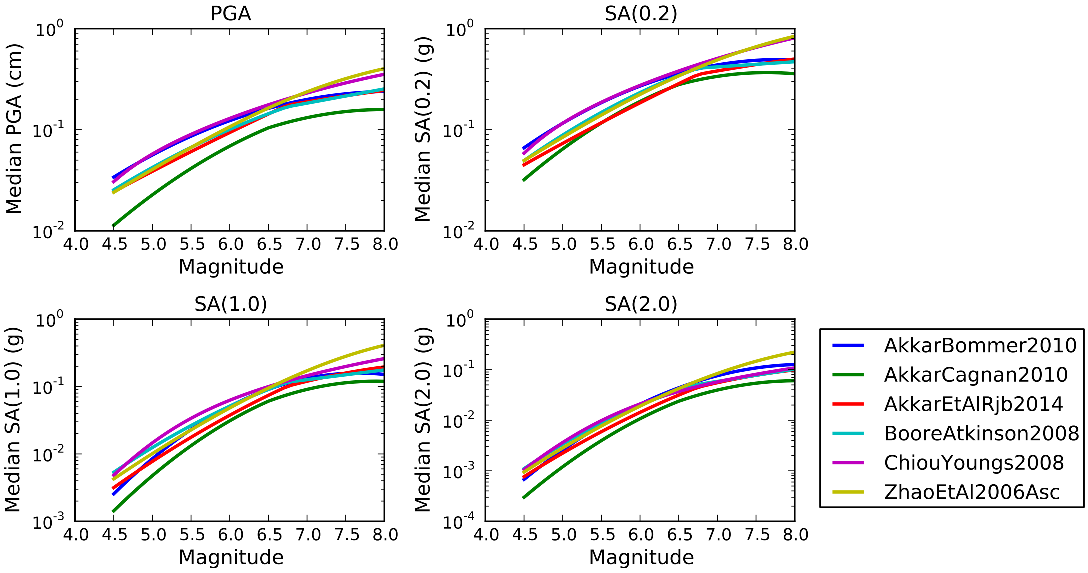


Figure 4.1 – *Scaling of GMPEs with respect to magnitude*

In addition to viewing the scaling of the expected ground motion from the GMPE, it is also possible to view the scaling of the standard deviation. This can be done with the following command:

```
trpl.MagnitudeSigmaIMTTrellis(magnitudes,
                              distances,
                              gmpe_list,
                              imts,
                              params,
                              stddevs="Total",
                              figure_size=(7,5),
                              filename="/path/to/plot",
                              filetype="png")
```

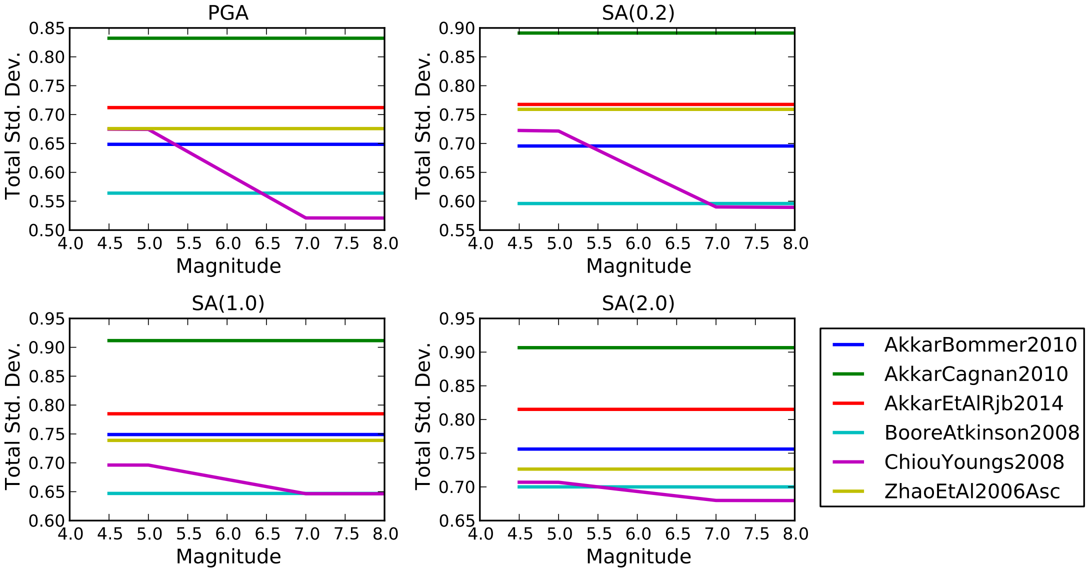


Figure 4.2 – *Scaling of total standard deviation of the GMPEs with respect to magnitude*

Figure 4.2 shows the scaling of the total standard deviation with magnitude. The type of standard deviation is configured using the `stddevs` option, which can take the values: "`Total`", "`Inter event`" or "`Intra event`". Examples of inter- and intra-event standard deviation are shown in Figure 4.3 and 4.4 respectively.

4.2.2 Comparing the Scaling with Distance

With this tool it is possible to compare how the GMPEs describe the attenuation of strong motion with distance for a given magnitude. As before, in order to utilise the tool in a "non-specific rupture" context, it is necessary to determine the distances relations manually. In this case we consider the same vertical rupture with a top of rupture depth of 5 km. For convenience we assume that the line of attenuation is perpendicular to the rupture strike, originating at the epicentre:

```
magnitude = 6.5
distances = {"repi": np.arange(0.0, 151.0, 1.0)}
distances["rhypo"] = np.sqrt(distances["repi"] ** 2.0 +
                             params["hypo_depth"] ** 2.0)
distances["rjb"] = distances["repi"]
distances["rrup"] = np.sqrt(distances["rjb"] ** 2.0 +
                            params["ztor"] ** 2.0)
distances["rx"] = distances["rjb"]
```

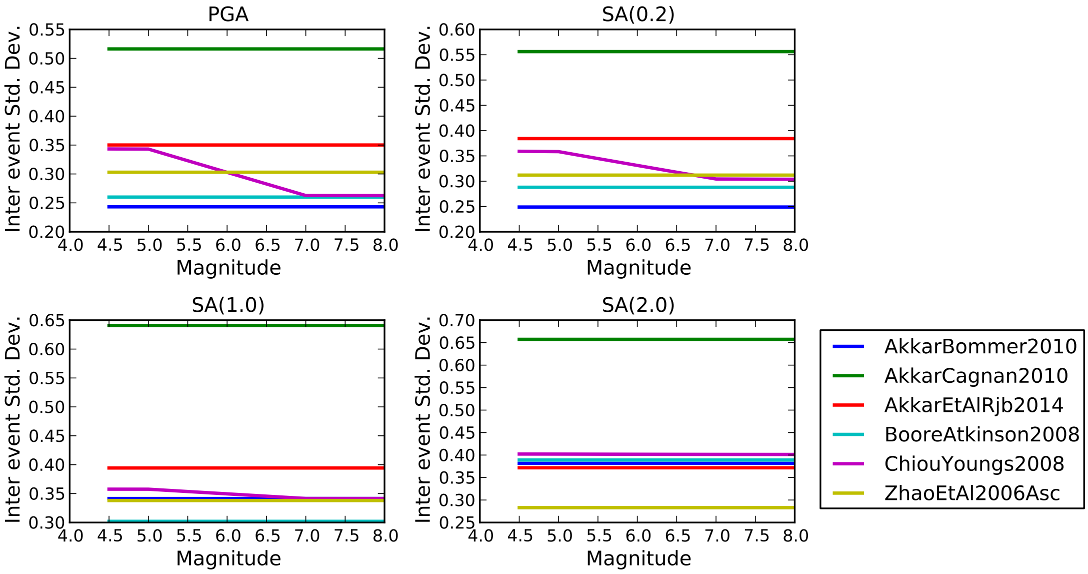


Figure 4.3 – *Scaling of inter-event deviation of the GMPEs with respect to magnitude*

The distance trellis plots, as shown in Figure 4.5 can be created with the following command:

```
trpl.DistanceIMTTrellis(magnitude,
                        distances,
                        gmpe_list,
                        imts,
                        params,
                        distance_type="rjb",
                        plot_type="loglog",
                        filename="path/to/image",
                        filetype="png")
```

The inputs are similar to that of the `trpl.MagnitudeIMTTrellis` tool, except that the user has the option of specifying which distance measure to use on the abscissa of the plot via the `distance_type` keyword. The use can also choose whether to plot the distance in logarithmic or linear axes via the `plot_type` keyword, which would take the value of “loglog” or “semilogy” respectively. The same GMPEs plotted in terms of linear distance axes are shown in Figure 4.6

As with the magnitude trellis plots. The standard deviations can also be plotted, as is the case for the total standard deviation shown in Figure 4.7:

```
trpl.DistanceSigmaIMTTrellis(magnitude,
                             distances,
                             gmpe_list,
                             imts,
                             params,
                             stddevs="Total"
                             distance_type="rjb",
                             plot_type="loglog",
                             filename="path/to/image",
                             filetype="png")
```

4.2.3 Comparing the Scaling of the Response Spectrum with Magnitude and Distance

Whilst understanding the scaling of a particular IMT can be useful, it can also be insightful to understand how a GMPE will scale the response spectrum with magnitude and distance. In the

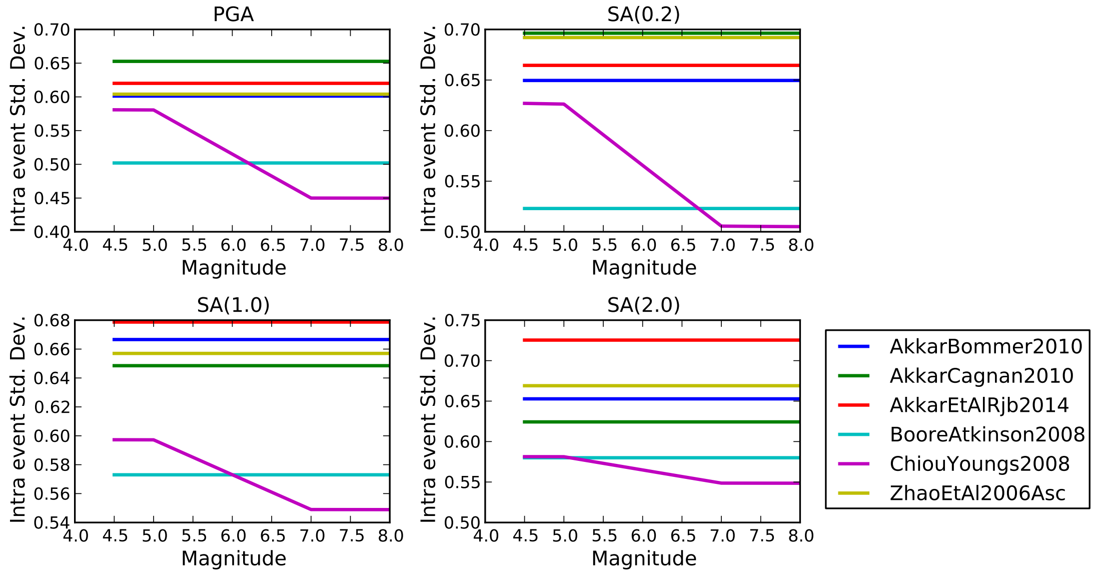


Figure 4.4 – *Scaling of intra-event deviation of the GMPEs with respect to magnitude*

following example we consider four magnitudes: M_W 5.0, M_W 6.0, M_W 7.0 and M_W 8.0; and four source-site distances R_{JB} 5.0, R_{JB} 20.0, R_{JB} 50.0, R_{JB} 100.0. Once again it is necessary to calculate the geometry in this simple case:

```
# Choose 4 magnitudes for comparison: 5, 6, 7, 8
magnitudes = np.array([5.0, 6.0, 7.0, 8.0])
# Choose 4 distances for comparison: 5., 20., 50., 100.
distances = {"repi": np.array([5.0, 20.0, 50.0, 100.0])}
distances["rhypo"] = np.sqrt(distances["repi"] ** 2.0 +
                             params["hypo_depth"] ** 2)
distances["rjb"] = distances["repi"]
distances["rrup"] = np.sqrt(distances["rjb"] ** 2.0 +
                            params["ztor"] ** 2)
distances["rx"] = distances["rjb"]
```

We need also to specify the spectral periods we wish to use for calculation. In this case, as citeAkkarCagnan2010 is limited to just 2 s period, we limit the period range for comparison so 0.05 s to 2 s.

```
periods = [0.05, 0.075, 0.1, 0.11, 0.12, 0.13, 0.14, 0.15,
           0.16, 0.17, 0.18, 0.19, 0.20, 0.22, 0.24, 0.26,
           0.28, 0.30, 0.32, 0.34, 0.36, 0.38, 0.40, 0.42,
           0.44, 0.46, 0.48, 0.50, 0.55, 0.60, 0.65, 0.70,
           0.75, 0.80, 0.85, 0.90, 0.95, 1.00, 1.10, 1.20,
           1.30, 1.40, 1.50, 1.60, 1.70, 1.80, 1.90, 2.00]
```

To produce the comparison shown in Figure 4.8 simply run:

```
trpl.MagnitudeDistanceSpectraTrellis(magnitudes,
                                     distances,
                                     gmpe_list,
                                     periods,
                                     params,
                                     plot_type="loglog",
                                     filename="path/to/image",
                                     filetype="png")
```

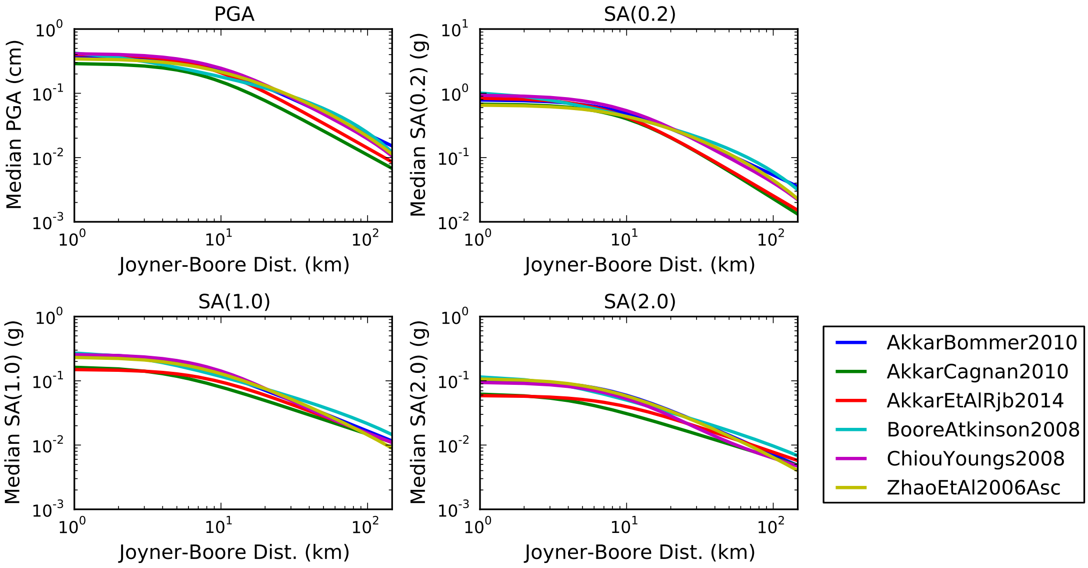


Figure 4.5 – *Scaling of the GMPEs with respect to distance*

Once again the corresponding standard deviations, shown in Figure 4.9, can also be plotted:

```
trpl.MagnitudeDistanceSpectraSigmaTrellis(magnitudes,
                                          distances,
                                          gmpe_list,
                                          periods,
                                          params,
                                          plot_type="semilogy",
                                          figure_size=(10,8),
                                          filename="path/to/image",
                                          filetype="png")
```

4.3 Trellis Plotting: The Simple Way

The examples seen section 4.2 are relatively feasible to implement in part because we are assuming a conveniently simple rupture configuration. To really understand how the GMPEs may compare in more physically consistent applications it is preferable to render the comparison with respect to a more realistic rupture alignment. To do this the GMPE-SMTK supplied a set of tools to help set up a rupture model that can be used to form the basis for the comparisons. These additional tools can be found in the `trellis/configure.py` module, which can be imported thus:

```
import smtk.trellis.configure as rcfg
```

A rupture is represented as an instance of the `rcfg.GSIMRupture` class, which can be created with the following essential information:

- `magnitude` The moment magnitude of the rupture
- `dip` The dip (in degrees) of the rupture
- `aspect` The along-strike to down-dip length ratio of the rupture

Further parameters can also be configured:

- `rake` Rake (degrees) of the rupture, default of 0.0
- `ztor` Top of rupture depth (km), default to 0 km
- `strike` Strike of the rupture (km, default to 0.0

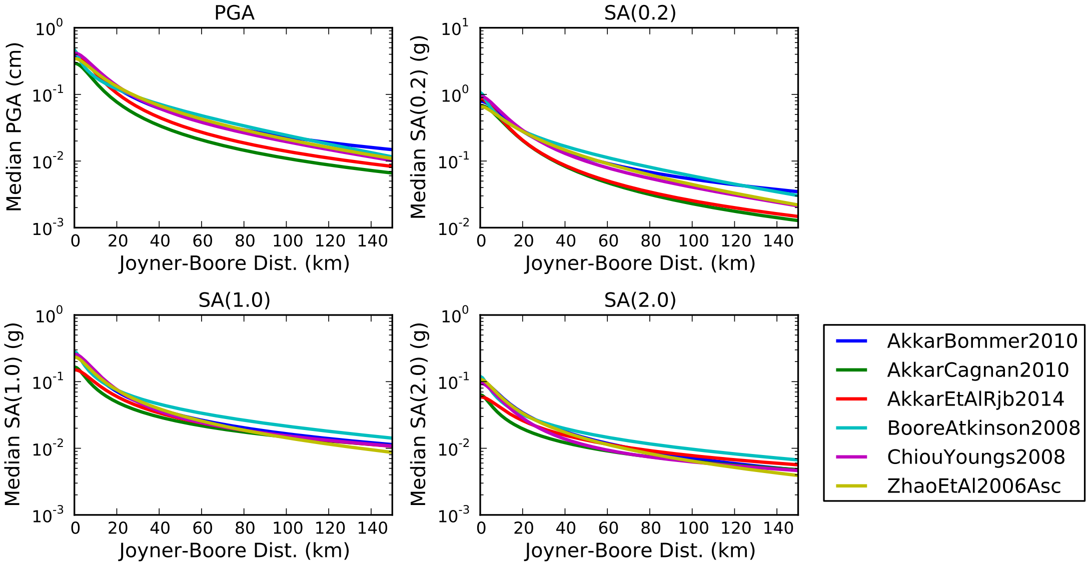


Figure 4.6 – *As Figure 4.5 with linear distance axes*

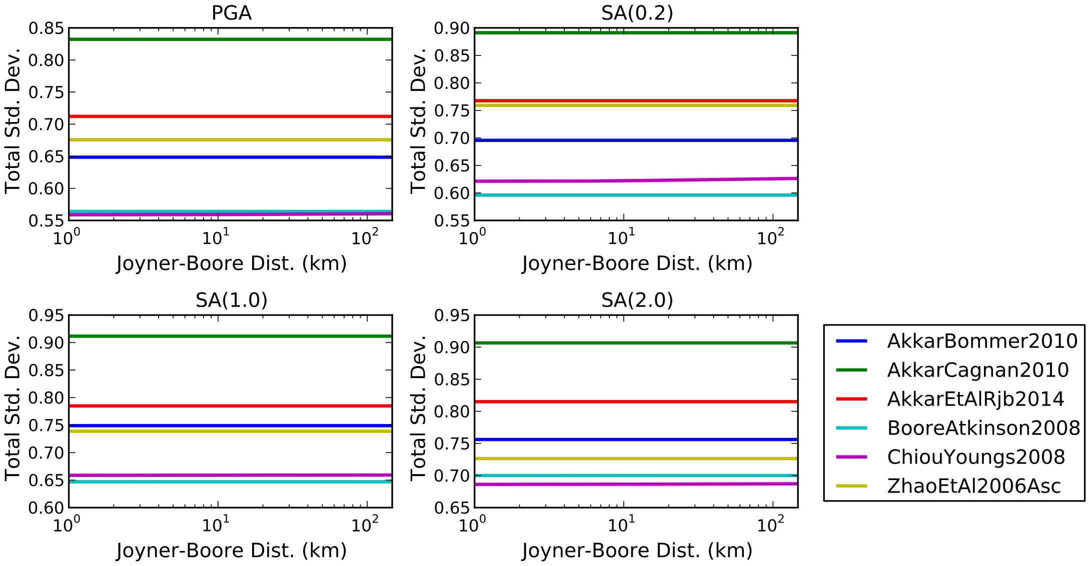


Figure 4.7 – *Scaling of the total standard deviation with distance.*

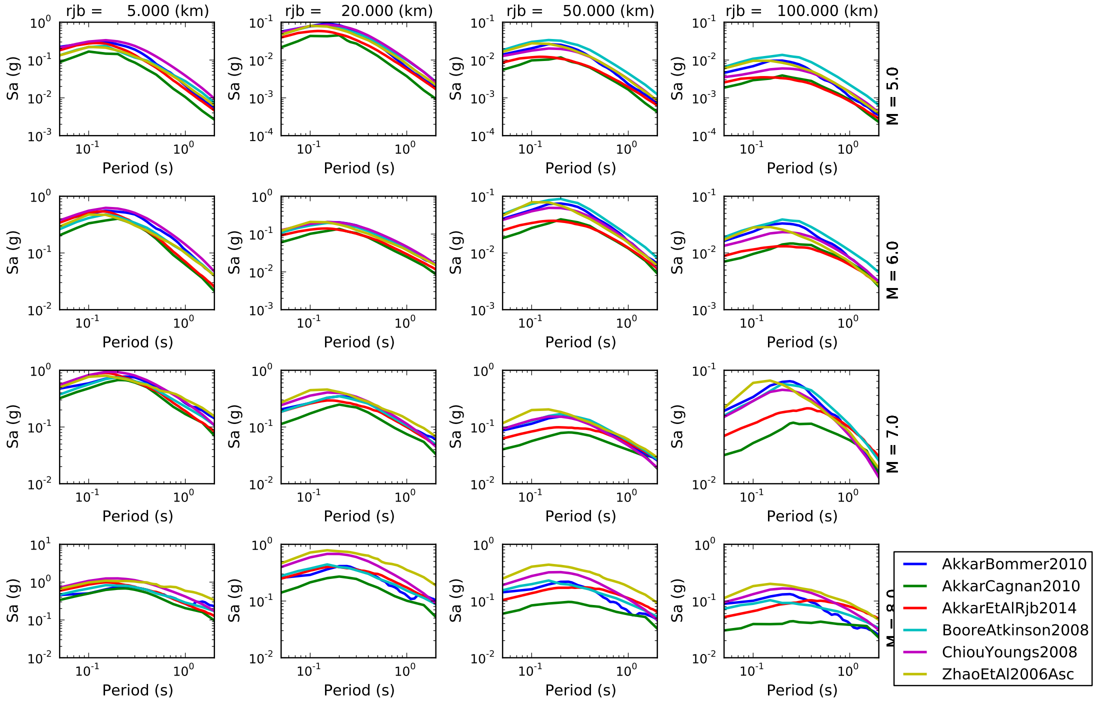


Figure 4.8 – *Scaling of the response spectrum for each GMPE for selected magnitude and source-site distance combinations*

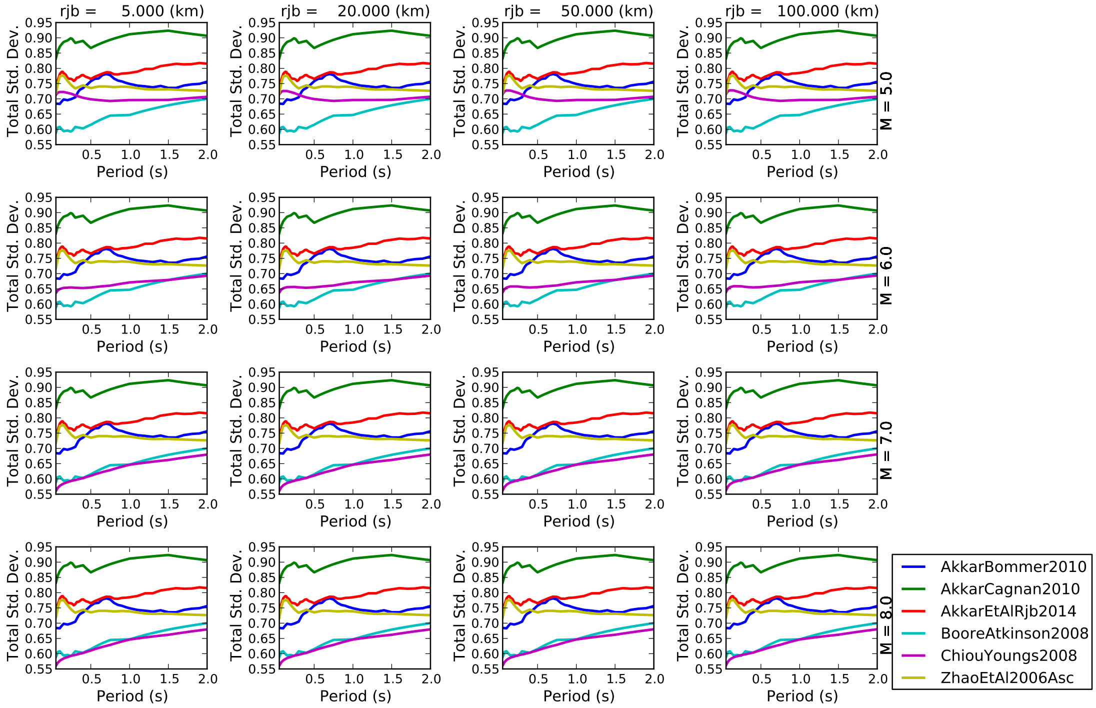


Figure 4.9 – *Scaling of the response spectrum total standard deviation for each GMPE for selected magnitude and source-site distance combinations*

- msr Magnitude scaling relation, default to **WellsCoppersmith1994**
- initial_point Location of the centroid on the Earth's surface as instance of openquake.hazardlib.geo.point.Point class
- hypocentre_location The location of the hypocentre within the rupture plane as a tuple of fraction of along-strike length and fraction of down-dip width (e.g. to place the hypocentre in the centroid of the rupture set this to(0.5, 0.5), the default value)

In the following example we consider a rupture plane from an earthquake of magnitude M_W 6.5, dip of 45°, aspect ratio of 1.5, rake of 90° (i.e. reverse fault), strike of 0° and hypocentre located in the centroid of the rupture:

```
rupt1 = rcfg.GSIMRupture(magnitude=6.5,
                         dip=45.0,
                         aspect=1.5,
                         rake=90.0,
                         ztor=0.0,
                         strike=0.0,
                         hypocentre_location=(0.5, 0.5))
```

4.3.1 Configuring the Sites

In order to generate the trellis plots we need to generate a site configuration that permits for the different distances to be rendered. There are three options by which this can be achieved, the selection of which may depend on the application.

Site as a Mesh

The first option, and the most computationally intensive, is to generate the target sites as an evenly spaced mesh of points centred on the rupture. This can be done using the method of the rcfg.GSIMRupture class named get_target_sites_mesh:

```
sites_mesh = rupt1.get_target_sites_mesh(maximum_distance=200.0,
                                         spacing=2.0,
                                         vs30=800.0,
                                         vs30measured=True,
                                         z1pt0=None,
                                         z2pt5=None)
```

Here the keyword maximum_distance specifies the maximum Joyner-Boore distance from the rupture to extend the mesh, spacing defines the mesh spacing (km), vs30 defines the 30-m averaged shear-wave velocity to assign to all the sites, vs30measured is a boolean parameter indicating whether the vs30 is measured (True, default) or inferred (False), z1pt0 and z2pt5 are the depth to the 1 km/s and 2.5 km/s shearwave velocity interfaces respectively. These are optional but if not specified they will be calculated from the vs30 value using the models of **AbrahamsonSilva2008** and **CampbellBozorgnia2008** respectively.

To visualise the rupture and site configuration as any time, simply use the plot_model_configuration method found in the rupture class:

```
rupt1.plot_model_configuration(figure_size=(7,5),
                               filename="/path/to/image",
                               filetype="png",
                               dpi=300)
```

The rupture-site mesh configuration for the above settings is shown in Figure 4.10

The rcfg.GSIMRupture class contains an additional method to allow the user to compare the distance metrics for the particular rupture and site configuration. For example, to compare the Joyner-Boore distance and rupture distance for the rupture mesh shown in Figure 4.10, the following will produce the plot shown in Figure 4.11

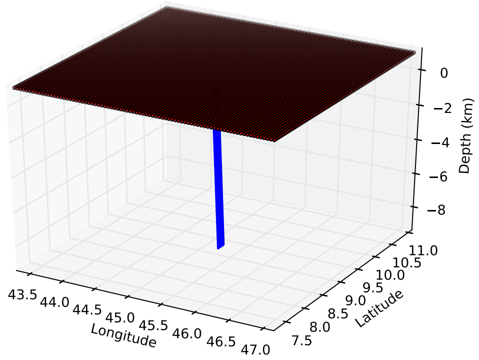


Figure 4.10 – *Rupture and site configuration for a regularly spaced mesh of points around the rupture*

```
rupt1.plot_distance_comparisons("rjb",
                                "rrup",
                                filename="path/to/image",
                                filetype="pdf")
```

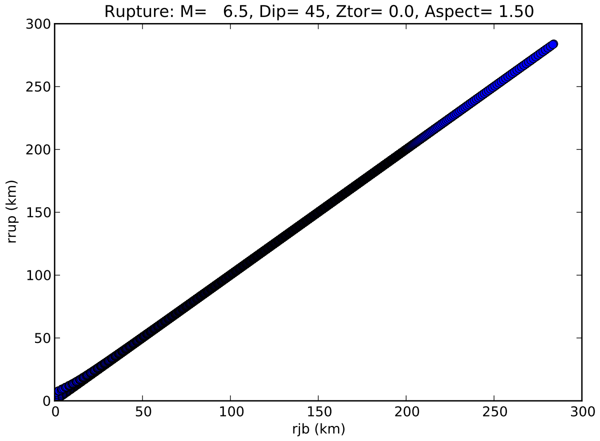


Figure 4.11 – *Comparison of Rupture distance with Joyner-Boore distance for the rupture and site configuration in Figure 4.10*

Sites as a Line

A more common configuration for exploring the attenuation properties of the GMPEs is to configure the sites in a single line propagating away from the rupture. However, it is important to recognise that the relation between the distance metrics will depend on the azimuth of the line with respect to the strike of the rupture. These properties can be configured using the command below, and the corresponding plot shown in Figure 4.12:

```
sites_line = rupt1.get_target_sites_mesh(maximum_distance=200.0,
                                         spacing=1.0,
                                         vs30=800.0,
                                         line_azimuth=90.
                                         vs30measured=True,
                                         z1pt0=None,
                                         z2pt5=None)
rupt1.plot_model_configuration()
```

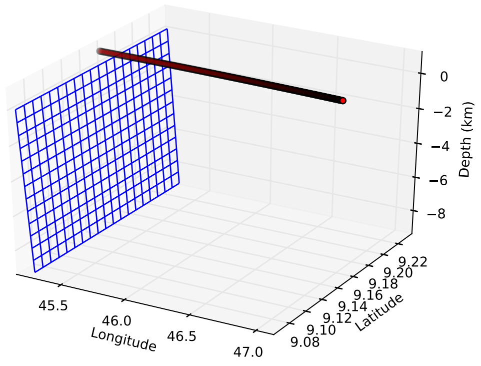


Figure 4.12 – *Rupture and site configuration for a regularly spaced line or points propagating at an angle of 90° from the rupture, along the hanging wall*

Site as a Point

When comparing GMPEs in terms of spectra it is preferable to consider the target site as a single point. In this case this both the distance, the distance type and the azimuth must be specified, in addition to the information required for the other site configurations. So to consider a site located at a Joyner-Boore distance of 20 km on a line perpendicular to the rupture, on the hanging wall, the configuration shown in Figure 4.13 can be constructed via:

```
sites_line = rupt1.get_target_sites_point(distance=20.0,
                                          distance_type="rjb",
                                          vs30=800.0,
                                          line_azimuth=90.
                                          vs30measured=True,
                                          z1pt0=None,
                                          z2pt5=None)
rupt1.plot_model_configuration()
```

4.3.2 Creating the Trellis Points

Each of the trellis plotting tools shown in section 4.2 can be implemented using their own built-in method called `from_rupture_model`. When using this method rather, than requiring the user manually specify the magnitude, distance and parameter inputs, it is possible to instead input only the rupture model.

For a rupture-specific configuration, distance trellis plots similar to those shown in Figure 4.12 can be created as follows:

```
trpl.DistanceIMTTrellis.from_rupture_model(
    rupt1,
    gmpe_list,
    imts,
    filename="path/to/image",
    filetype="pdf")
```

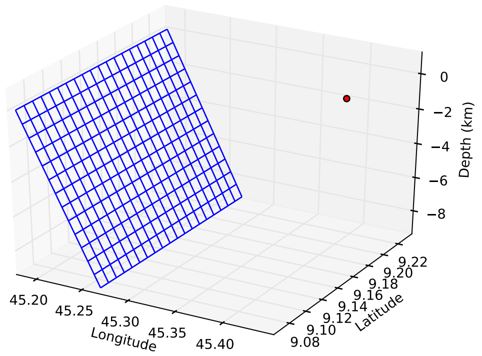


Figure 4.13 – *Rupture and site configuration for a single point located at 20 km (Joyner-Boore) from the rupture on a bearing of 90° on the hanging wall*

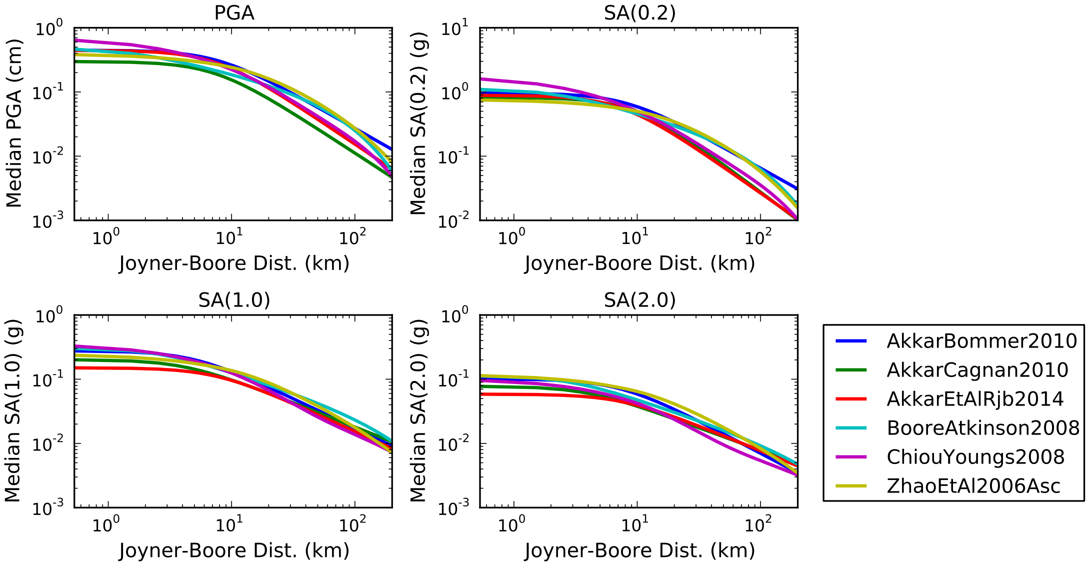


Figure 4.14 – *Scaling of the GMPEs with respect to distance for the rupture and site configuration in Figure 4.12*

Likewise, the standard deviation plots (e.g. Figure 4.15) can be created in a similar manner:

```
trpl.DistanceSigmaIMTTrellis.from_rupture_model(
    rupt1,
    gmpe_list,
    imts,
    filename="path/to/image",
    filetype="pdf")
```

The rupture-specific configuration allows for the same comparison to take place on the footwall of the rupture, for those GMPE's such as citeChiouYoungs2008 that have hanging wall coefficients. For example, rather than propagating the line at 90° from the rupture on the hanging wall, we will place the line at an azimuth of 230° from the rupture (i.e. on the footwall):

```
sites_line = rupt1.get_target_sites_mesh(maximum_distance=200.0,
                                         spacing=1.0,
```

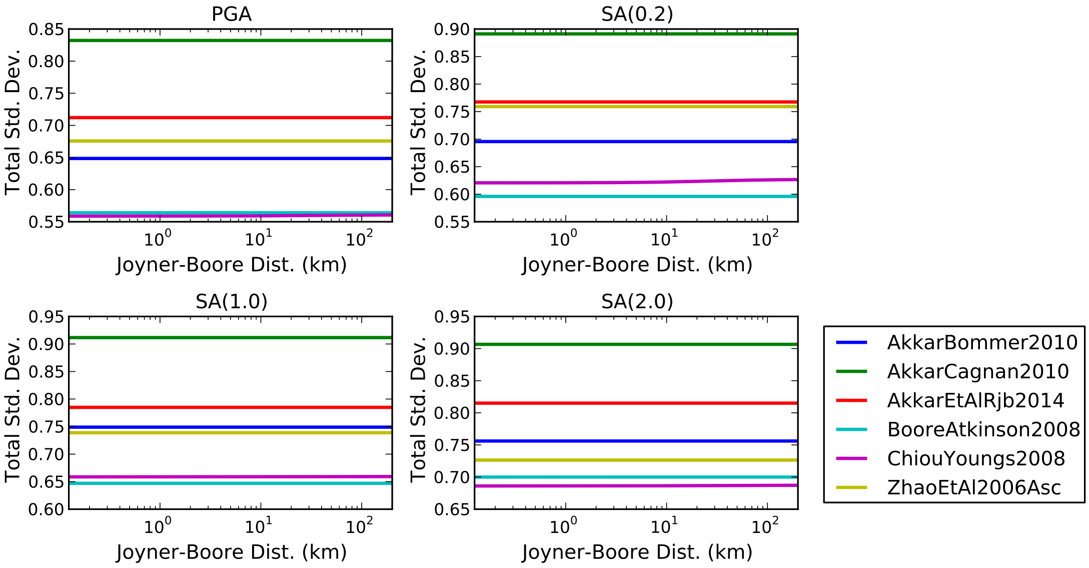


Figure 4.15 – *Scaling of the total standard deviation GMPEs with respect to distance for the rupture and site configuration in Figure 4.12*

```
                                         vs30=800.0,
                                         line_azimuth=230.
                                         vs30measured=True,
                                         z1pt0=None,
                                         z2pt5=None)
rupt1.plot_model_configuration()
```

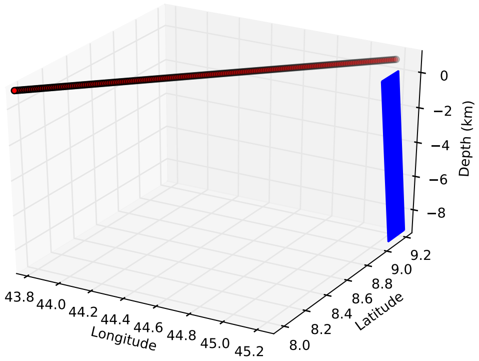


Figure 4.16 – *Rupture and site configuration for a regularly spaced line or points propagating at an angle of 90° from the rupture, along the footwall*

The corresponding trellis plots for the expected ground motion values is shown in Figure 4.17

This same comparison can be undertaken for the response spectra too. In this case we consider the rupture and site configuration shown in figure 4.13 (site at 20 km R_{JB} from the rupture along a bearing of 90°). To view the predicted response spectra at this site it is only necessary to do the following to produce the plots shown in Figures 4.18 and 4.19:

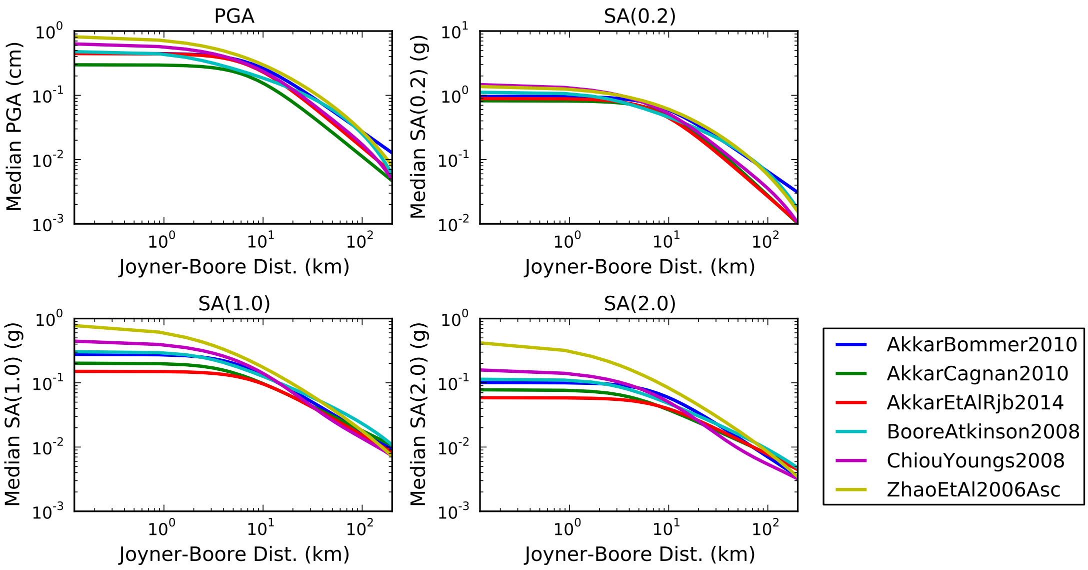


Figure 4.17 – *Scaling of the GMPEs along the footwall of a rupture, configured according to Figure 4.16*

```
sites_line = rupt1.get_target_sites_point(distance=20.0,
                                          distance_type="rjb",
                                          vs30=800.0,
                                          line_azimuth=90.
                                          vs30measured=True,
                                          z1pt0=None,
                                          z2pt5=None)
# Create the expected ground motion trellis plot
trpl.MagnitudeDistanceSpectraTrellis.from_rupture_model(
    rupt1,
    gmpe_list,
    periods,
    filename="path/to/image",
    filetype="png")
trpl.MagnitudeDistanceSpectraSigmaTrellis.from_rupture_model(
    rupt1,
    gmpe_list,
    periods,
    stddevs="Total",
    filename="path/to/image",
    filetype="png")
```

The same comparison can also be done on the footwall of the rupture. If the site is now placed at a Joyner-Boore distance of 20 km on the at a bearing of 220° from the rupture (i.e. on the footwall), the configuration in Figure 4.20 is created as shown:

```
sites_line = rupt1.get_target_sites_point(distance=20.0,
                                          distance_type="rjb",
                                          vs30=800.0,
                                          line_azimuth=220.
                                          vs30measured=True,
                                          z1pt0=None,
                                          z2pt5=None)
rupt1.plot_model_configuration()
```

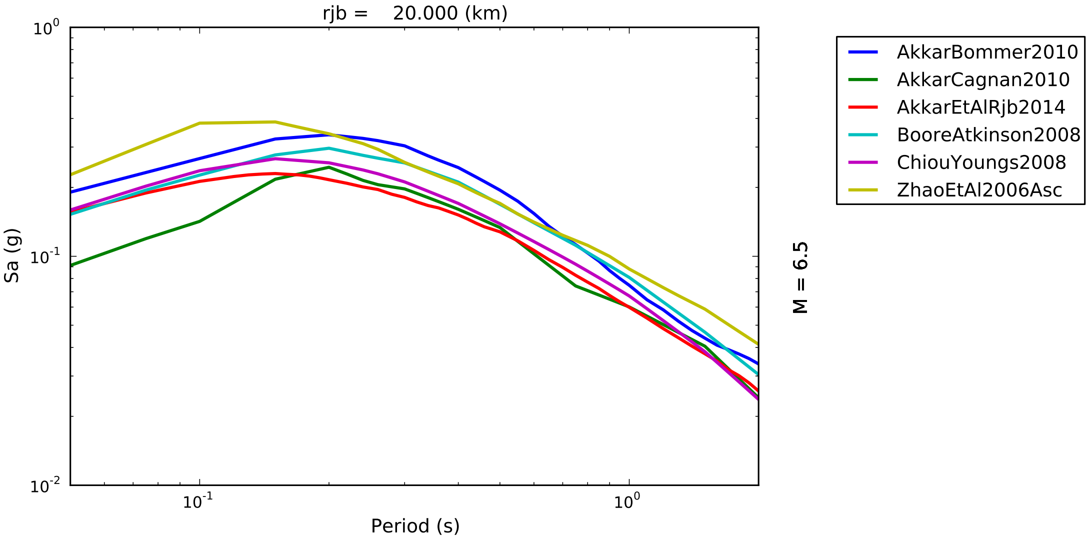


Figure 4.18 – *Comparison of response spectra from the GMPEs for the rupture and site configuration shown in Figure 4.13*

The corresponding GMPEs and their respective total standard deviations are shown in Figures 4.21 and 4.22 respectively.

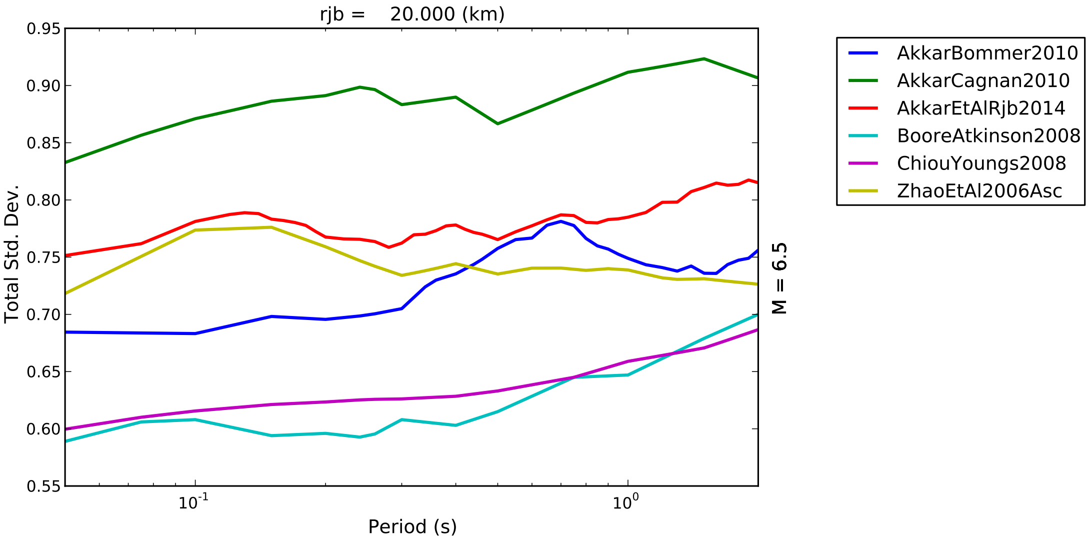


Figure 4.19 – *Comparison of total standard deviation of the response spectra from the GMPEs for the rupture and site configuration shown in Figure 4.13*

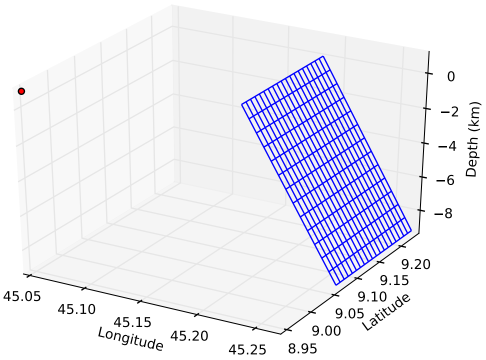


Figure 4.20 – *Rupture and site configuration for a single point located at 20 km (Joyner-Boore) from the rupture on a bearing of 220° on the hanging wall*

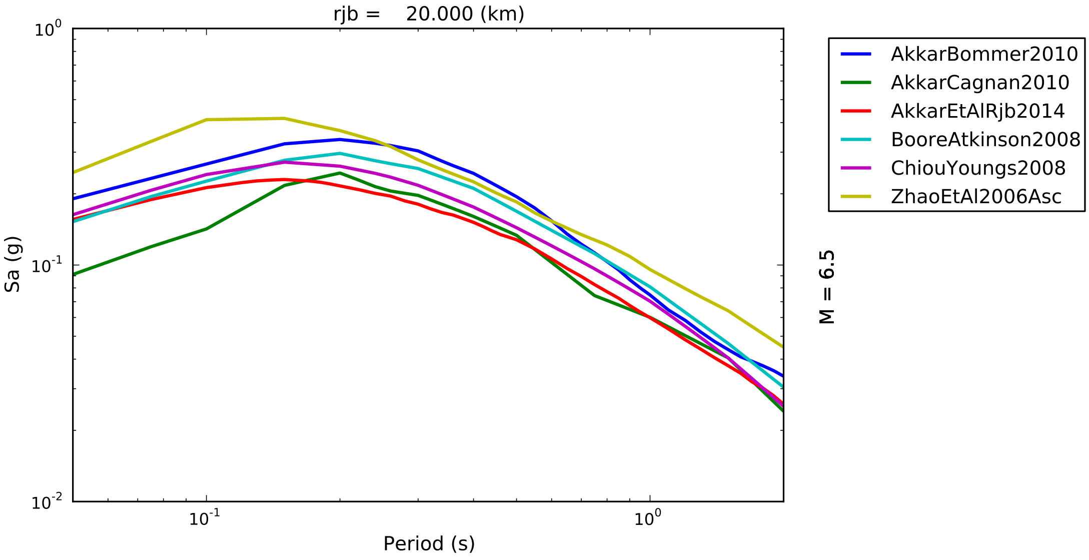


Figure 4.21 – *Comparison of response spectra from the GMPEs for the rupture and site configuration shown in Figure 4.20*

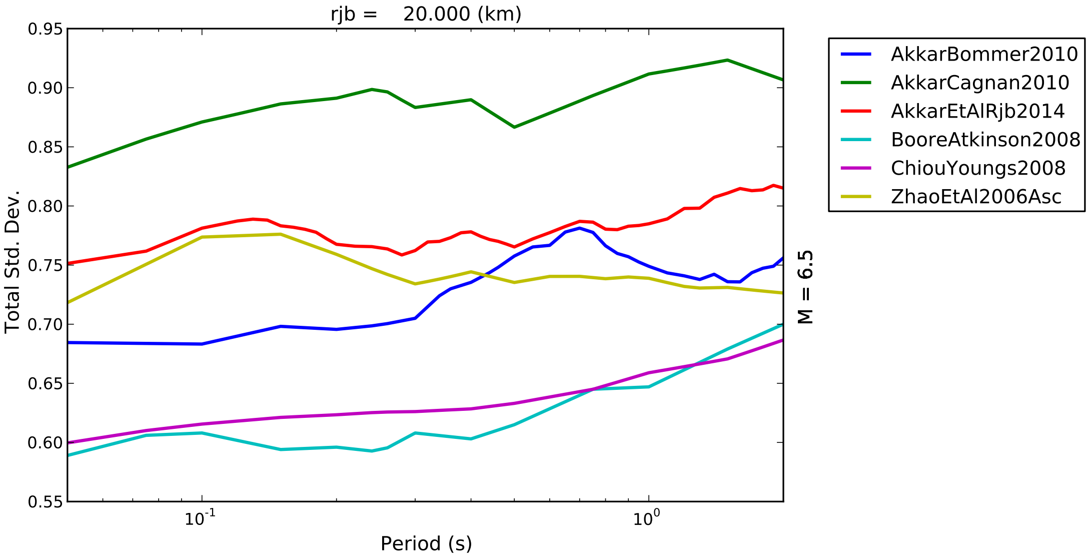
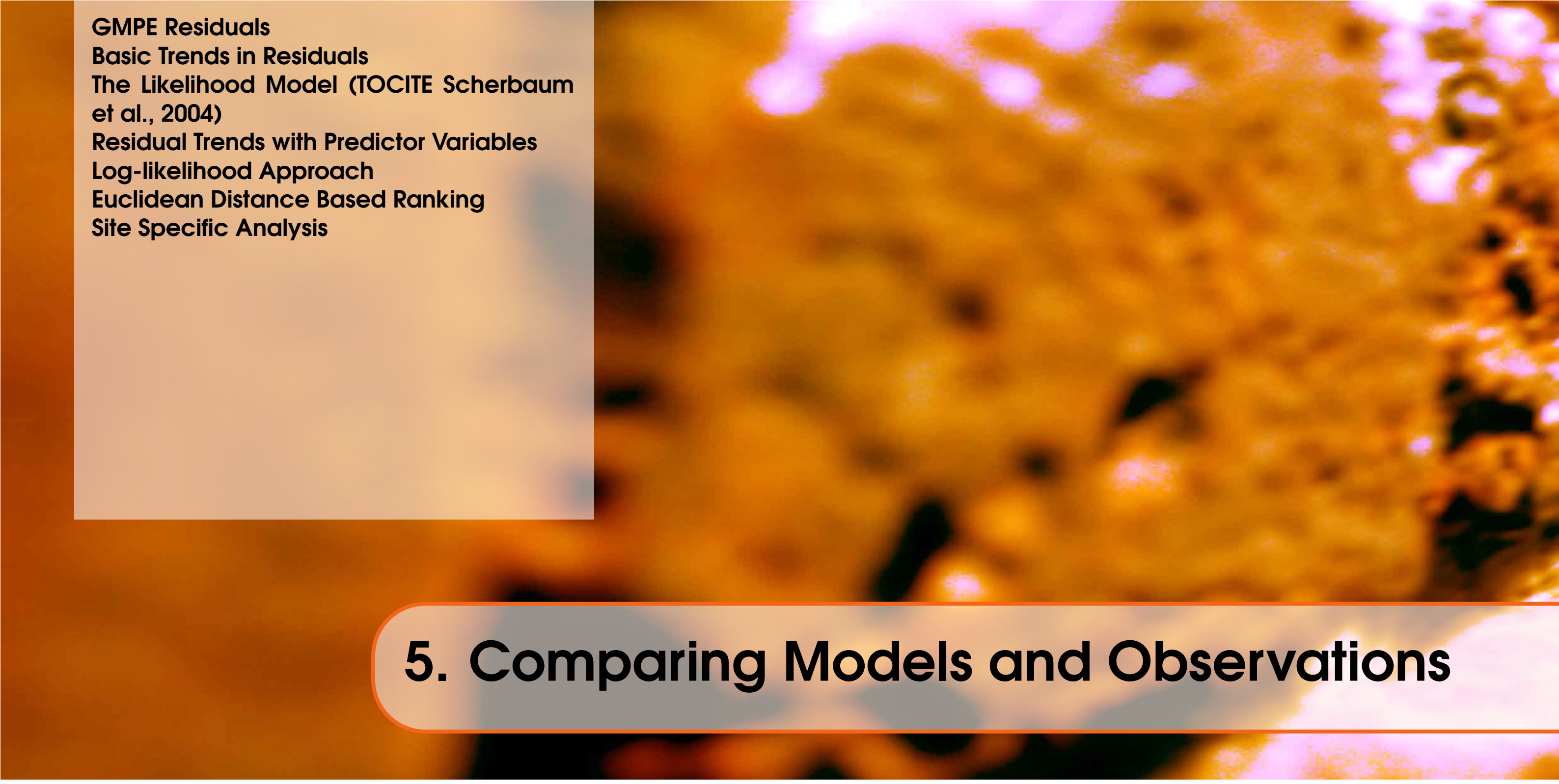
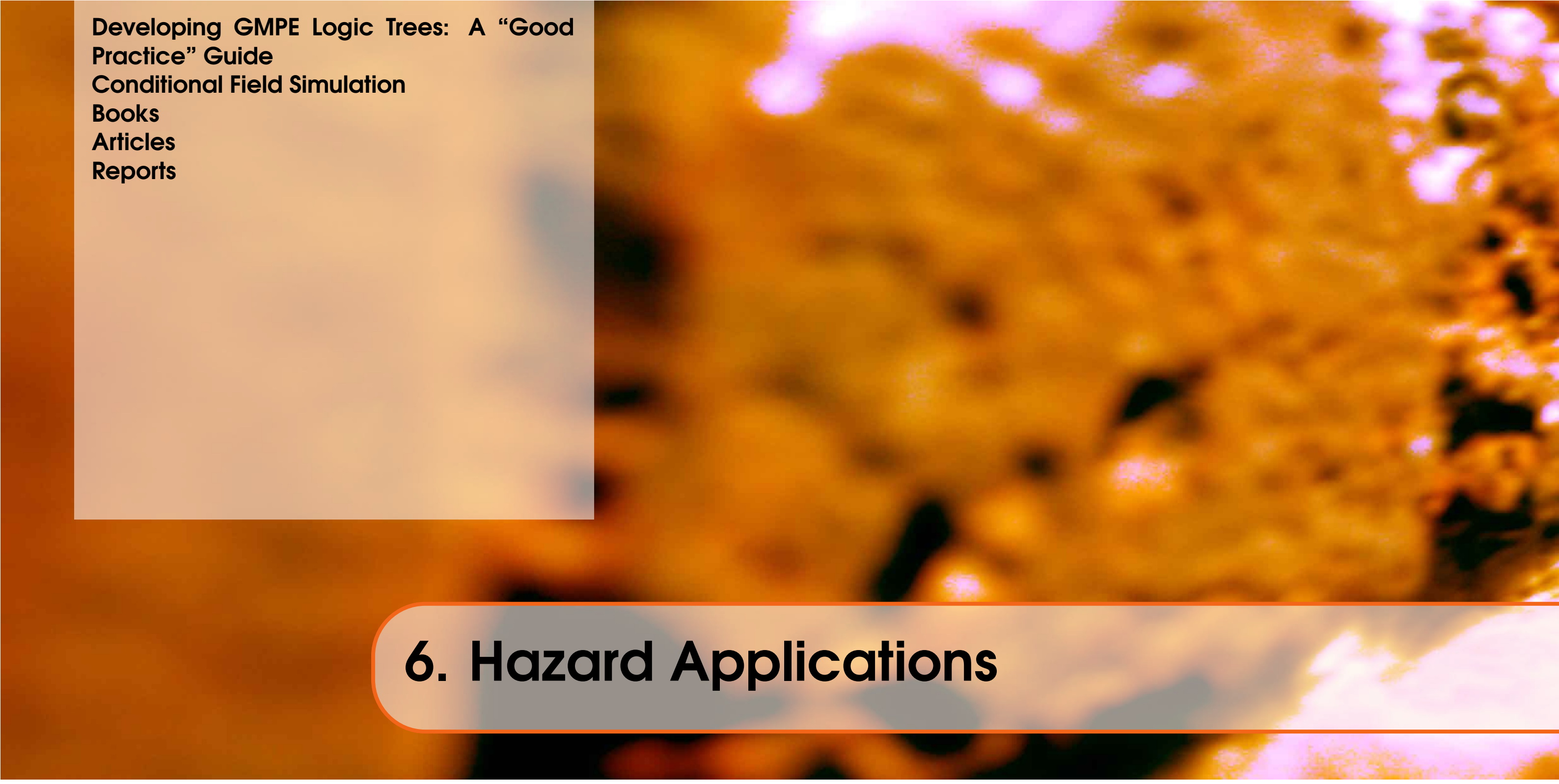


Figure 4.22 – *Comparison of total standard deviation of the response spectra from the GMPEs for the rupture and site configuration shown in Figure 4.20*



5. Comparing Models and Observations



6. Hazard Applications

6.1 Developing GMPE Logic Trees: A “Good Practice” Guide

6.2 Conditional Field Simulation

Bibliography

Books

Articles

Other Sources