OPENQUAKE
calculate share explore

RISK MODELLERS TOOLKIT
USER INSTRUCTION MANUAL

Version 1.0

Hands-on-instructions on the different functionalities of the Risk Modellers Toolkit.

GEM
GLOBAL EARTHQUAKE MODEL
working together to assess risk

OpenQuake: calculate, share, explore

Risk Modeller's Toolkit User Instruction Manual

github.com/gemsciencetools/rmtk

PUBLISHED BY GEM FOUNDATION

GLOBALQUAKEMODEL.ORG/OPENQUAKE

Citation

Please cite this document as:

Silva, V., Casotto, C., Rao, A., Villar, M., Crowley, H. and Vamvatsikos, D. (2015) OpenQuake Risk Modeller's Toolkit - User Guide. *Global Earthquake Model (GEM). Technical Report 2015-09.*

doi: 10.13117/GEM.OPENQUAKE.MAN.RMTK.1.0/02, 97 pages.

Disclaimer

The "Risk 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.

License

First printing, September 2015

Contents

Preface

The goal of this book is to provide a comprehensive and transparent description of the methodologies adopted during the implementation of the OpenQuake Risk Modeller's Toolkit (RMTK). The Risk Modeller's Toolkit (RMTK) is primarily a software suite for creating the input models required for running seismic risk calculations using the OpenQuake-engine. The RMTK implements several state-of-the-art methods for deriving robust analytical seismic fragility and vulnerability functions for single structures or building classes. The RMTK also provides interactive tools for post-processing and visualising different results from the OpenQuake-engine seismic risk calculations, such as loss exceedance curves, collapse maps, damage distributions, and loss maps.

The OpenQuake Risk Modeller's Toolkit is the result of an effort carried out jointly by the IT and Scientific teams working at the Global Earthquake Model (GEM) Secretariat. It is freely distributed under an Affero GPL license (more information available at this link http://www.gnu.org/licenses/agpl- 3.0.html).

1. Introduction

In recent years several free and open-source software packages for seismic hazard and risk assessment have been developed. The OpenQuake-engine developed by the Global Earthquake Model (GEM) Foundation (Pagani et al., 2014) is one such software which provides state-of-the-art scenario-based and probabilistic seismic hazard and risk calculations. The availability of such software has made it easier for hazard and risk modellers to run complex analyses without needing to code their own implementations of the scientific algorithms. However, the development of the input models for seismic risk analyses is often an equally challenging task, and the availability of tools to help modellers in this stage are very limited.

The main inputs required for a physical seismic risk analysis are a seismic hazard model, an exposure model, and a physical fragility or vulnerability model. A hazard model itself comprises three components: the seismogenic source model(s), the ground motion prediction equations and the logic tree to characterise the epistemic uncertainties in the model. The exposure model describes the locations and other physical characteristics of the buildings within the region of interest. Finally, the physical fragility and vulnerability models describe the probability of damage and loss, respectively, for different levels of ground shaking for the different building classes in the region of interest.

The lack of tools for model preparation may require modellers to use tools that were not specifically designed for the creation of seismic hazard or risk models. Alternatively, modellers justifiably create their own implementations of standard methodologies, leading to a possible lack of consistency between different implementations, even if the methods selected in the model preparation process were the same. Quality assurance and maintenance of code is another issue that modellers are required to deal with. There is clearly a strong motivation for extending the philosophy of open-source software development, review, and maintenance also to the process of model preparation.

Thus, in addition to the OpenQuake-engine, GEM is now in the process of developing a set of tools to aid hazard and risk modellers during the model preparation stage. These tools are currently made available in the form of three "Modeller's Toolkits":

1. **The Hazard Modeller's Toolkit:** a suite of open-source tools for the preparation of seismogenic source models for application in probabilistic seismic hazard assessment (Weatherill, 2014)
2. **The Ground Motion Toolkit:** a suite of open-source tools for analysis and interpretation of observed ground motions and ground motion prediction equations, for the purposes of GMPE selection in PSHA
3. **The Risk Modeller's Toolkit:** a suite of open-source tools for the preparation of physical fragility and vulnerability models for application in seismic risk assessment, and for the post-processing and visualising of results from OpenQuake risk analyses

This user manual describes the set of tools and other functionalities provided by the Risk Modeller's Toolkit (RMTK). In particular, the RMTK makes it possible for a risk modeller to select from several commonly used methods to derive seismic fragility or vulnerability functions for individual buildings or a class of buildings. The currently available functionalities of the toolkit are shown graphically in Figure 1.1 and described in more detail in Section 1.2.

As with the OpenQuake-engine, the RMTK is developed primarily using the Python programming language and released under the GNU Affero open-source license. The RMTK software and user manual are updated regularly by scientists and engineers working within the GEM Secretariat. The latest version of the software is available on an open GitHub repository: https://github.com/GEMScienceTools/rmtk. The user manual for the RMTK can be downloaded here: https://github.com/GEMScienceTools/rmtk_docs.

The RMTK is continuously evolving. It is already used extensively within the GEM Foundation, but we hope it will prove to be useful for other risk modellers, so please try it out! Feedback and contribution to the software is welcome and highly encouraged, and can be directed to the risk scientific staff of the GEM Secretariat (risk@globalquakemodel.org).

1.1 Getting started

The Risk Modeller's Toolkit makes extensive use of the Python programming language and the web-browser based interactive IPython notebook interface. As with the OpenQuake-engine, the preferred working environment is Ubuntu (12.04 or later) or Mac OS X. At present, the user must install the dependencies manually. An effort has been made to keep the number of additional dependencies to a minimum. More information regarding the current dependencies of the toolkit can be found at http://github.com/GEMScienceTools/rmtk.

The current dependencies are:

- numpy and scipy (included in the standard OpenQuake installation)
- matplotlib (http://matplotlib.org/)

The matplotlib library can be installed easily from the command line by:

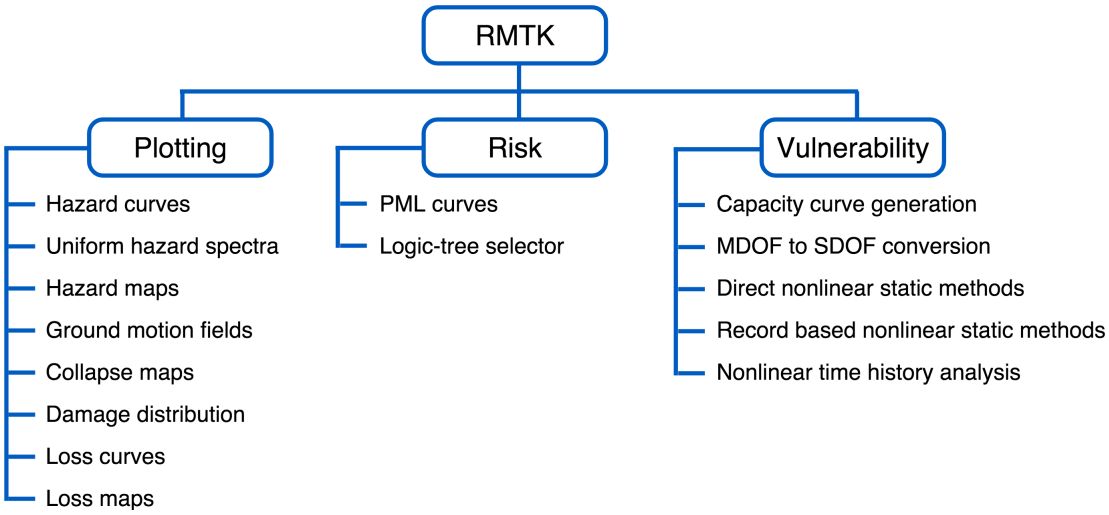

Figure 1.1 – *The modular structure of the Risk Modeller's Toolkit (RMTK), showing the currently available functionalities*

```
~$ sudo pip install matplotlib
```

The Risk Modeller's Toolkit itself requires no specific installation. In a Unix/Linux environment you can simply download the code as a zipped package from the website listed above, unzip the files, and move into the code directory. Alternatively you can download the code directly into any current repository with the command

```
~$ git clone https://github.com/GEMScienceTools/rmtk.git
```

To enable usage of the RMTK within any location in the operating system, OS X and Linux users should add the path to the RMTK folder to their profile file. This can be done as follows:

1. Using a command line text editor (e.g. VIM or Emacs), open the `~/.profile` folder as follows:

   ```
   ~$ vim ~/.profile
   ```

2. At the bottom of the profile file add the line:

   ```
   export PYTHONPATH=/path/to/rmtk/folder/:$PYTHONPATH
   ```

Where `/path/to/rmtk/folder/` is the system path to the location of the rmtk folder (use the command `pwd` from within the RMTK folder to view the full system path).

3. Reload the profile file using the command

```
~$ source ~/.profile
```

The IPython Notebook is a web browser-based notebook which provides support for interactive coding, text, mathematical expressions, inline plots and other rich media. Static notebooks can also be created for recording and distributing the results of the rich computations.

If you already have Python installed, you can get IPython along with the dependencies for the IPython notebook using pip:

```
~$ sudo pip install "ipython[notebook]"
```

A notebook session can be started via the command line:

```
~$ ipython notebook
```

This will print some information about the notebook server in your console, and open a web browser to the URL of the web application (by default, http://127.0.0.1:8888).

The landing page of the IPython notebook web application, the dashboard, shows the notebooks currently available in the notebook directory (by default, the directory from which the notebook server was started).

You can create new notebooks from the dashboard with the New Notebook button, or open existing ones by clicking on their name.

At present, the recommended approach for Windows users is to run Ubuntu Linux 14.04 within a Virtual Machine and install the RMTK following the instructions above. Up-to-date VirtualBox images containing the OpenQuake-engine and platform, and the Hazard and Risk Modeller's Toolkits are available here: http://www.globalquakemodel.org/openquake/start/download/

Knowledge of the Python programming language is not necessary in order to use the tools provided in the Risk Modeller's Toolkit. Nevertheless, a basic understanding of the data types and concepts of Python will come in handy if you are interested in modifying or enhancing the standard scripts provided in the toolkit. If you have never used Python before, the official Python tutorial is a good place to start. A Byte of Python is also a well-written guide to Python and a great reference for beginners. Appendix A at the end of this user manual also provides a quick-start guide to Python.

1.2 Current features

The Risk Modeller's Toolkit is currently divided into three modules:

Plotting tools: The plotting module of the RMTK provides scripts and notebooks for visualising all of the different hazard and risk outputs produced by calculations performed using the OpenQuake-engine. The visualisation tools currently support plotting of seismic hazard curves, uniform hazard spectra, hazard maps, loss exceedance curves, probabilistic loss maps, and damage distribution statistics, amongst others. This module also allow users to convert the different results from the standard OpenQuake XML format into other formats such as CSV.

Risk tools: This module provides useful scripts to further post-process the OpenQuake-engine hazard and risk results, and calculate additional risk metrics.

Vulnerability tools: This module implements several methodologies for estimating fragility and vulnerability functions for individual buildings or for a class of buildings. The provided scripts differ in level of complexity of the methodology and according to the type of input data available for the buildings under study. The guidelines for analytical vulnerability assessment provided by the GEM Global Vulnerability Consortium have been used as the primary reference for the development of this module.

A summary of the methods available in the present version is given in Table 1.1.

1.3 About this manual

This manual is designed to explain the various functions in the toolkit and to provide some illustrative examples showing how to implement them for particular contexts and applications. As previously indicated, the Risk Modeller's Toolkit itself is primarily a Python library comprising three modules containing plotting tools, risk tools, and vulnerability tools respectively. The modular nature of the toolkit means that all of the functions within the toolkit can be utilised by a risk modeller in different python applications. In addition to the Python scripts, the RMTK also includes several interactive IPython notebooks illustrating sample usage of the functions in the toolkit. Each IPython notebook is intended to be stand-alone and self-explanatory and includes the most relevant information about the methodologies demonstrated in that notebook. In addition to the information provided in the notebooks, this manual provides more details about the theory and implementation, the algorithms and equations, the input model formats, and references for the different methodologies included in the toolkit.

Chapter 1 describes the motivation behind the development of the RMTK, the installation and usage instructions, and the list of features that are currently implemented in the toolkit. Chapter 2 describes the tools and notebooks provided in the plotting module of the RMTK for visualisation of OpenQuake hazard and risk results. Chapter 3 provides a brief description of the risk module, which includes two tools: (1) for post-processing hazard curves from a classical PSHA calculation with non-trivial logic-trees and (2) for deriving probable maximum loss (PML) curves using loss tables from an event-based risk calculation. Chapter 4 describes the different methodologies implemented in the vulnerability module of the RMTK for

Table 1.1 – *Current features in the OpenQuake Risk Modeller's Toolkit*

Module	Feature / Methodology
Plotting Module	Hazard Curves Uniform Hazard Spectra Hazard Maps Ground Motion Fields Collapse Maps Scenario Loss Maps Probabilistic Loss Maps Loss Curves Damage Distribution
Risk Module	Probable Maximum Loss Calculator Logic-Tree Branch Selector
Vulnerability Module	**Capacity Curve Generation** DBELA SP-BELA Point Dispersion **MDOF→SDOF Conversion** Single Mode of Vibration Adaptive Approach **Direct Nonlinear Static Methods** SPO2IDA (Vamvatsikos and Cornell, 2005) Dolsek and Fajfar (2004) Ruiz-Garcia and Miranda (2007) **Record Based Nonlinear Static Methods** Vidic et al. (1994) Lin and Miranda (2008) Miranda (2000) for Firm Soils N2 (CEN, 2005) Capacity Spectrum Method (FEMA-440, 2005) DBELA (Silva et al., 2013) **Nonlinear Time History Analysis** for SDOF Oscillators

deriving seismic fragility and vulnerability functions for individual buildings or for a class of buildings. Several supplementary tools required in this process are also introduced and described in this chapter. Finally, Appendix A presents a brief introductory tutorial for the Python programming language.

2. Plotting

The OpenQuake-engine is capable of generating several seismic hazard and risk outputs, such as loss exceedance curves, seismic hazard curves, loss and hazard maps, damage statistics, amongst others. Most of these outputs are stored using the Natural hazards' Risk Markup Language (NRML), or simple comma separated value (CSV) files. The Plotting module of the Risk Modeller's Toolkit allows users to visualize the majority of the OpenQuake-engine results, as well as to convert them into other formats compatible with GIS software (e.g. QGIS). Despite the default styling of the maps and curves defined within the Risk Modeller's Toolkit, it is important to state that any user can adjust the features of each output by modifying the original scripts.

2.1 Plotting damage distribution

Using the Scenario Damage Calculator (Silva et al., 2014a) of the OpenQuake-engine, it is possible to assess the distribution of damage for a collection of assets considering a single seismic event. These results include damage distributions per building typology, total damage distribution, and distribution of collapsed buildings in the region of interest.

2.1.1 Plotting damage distributions

This feature of the Plotting module allows users to plot the distribution of damage across the various vulnerability classes, as well as the total damage distribution. For what concerns the former result, it is necessary to set the path to the output file using the parameter `tax_dmg_dist_file`. It is also possible to specify which vulnerability classes should be considered, using the parameter `taxonomy_list`. However, if a user wishes to consider all of the vulnerability classes, then this parameter should be left empty. It is also possible to specify if a 3D plot containing all of the vulnerability classes should be generated, or instead a 2D plot per vulnerability class. To follow the former option, the parameter `plot_3d` should

be set to `True`. It is important to understand that this option leads to a plot of damage fractions for each vulnerability class, instead of the number of assets in each damage state. An example of this output is illustrated in Figure 2.1.

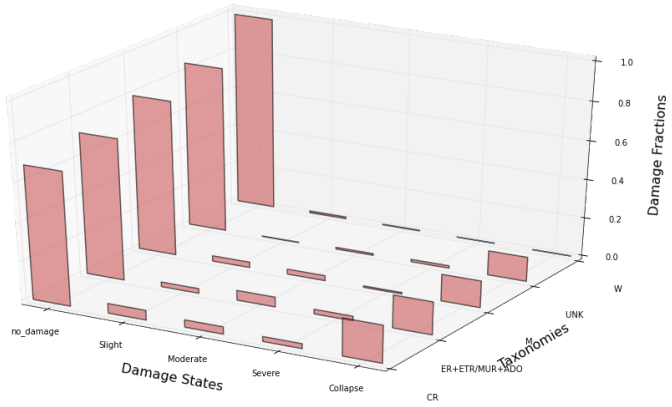

Figure 2.1 – *Damage distribution per vulnerability class.*

In order to plot the total damage distribution (considering the entire collection of assets), it is necessary to use the parameter `total_dmg_dist_file` to define the path to the respective output file. Figure 2.2 presents an example of this type of output.

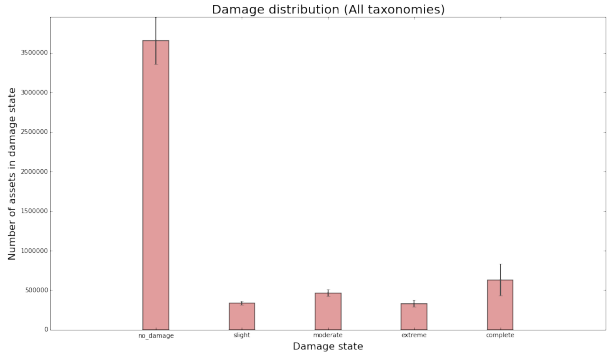

Figure 2.2 – *Total damage distribution.*

2.1.2 Plotting collapse maps

The OpenQuake-engine also generates an output defining the spatial distribution of the mean (and associated standard deviation) of assets in the last damage state (usually representing collapse or complete damage). The location of this output needs to be specified using the parameter `collapse_map`. Then, it is necessary to specify whether the user desires a map with the aggregated number of collapsed assets (i.e. at each location, the mean number of collapsed assets across all of the vulnerability classes are summed) or a map for each vulnerability class. Thus, the following options are permitted:

1. Aggregated collapse map only.

2. Collapse maps per vulnerability class only.
3. Both aggregated and vulnerability class-based.

The plotting option should be specified using the parameter `plotting_type`, and the location of the exposure model used to perform the calculations must be defined using the variable `exposure_model`. A number of other parameters can also be adjusted to modify the style of the resulting collapse map, as follows:

- `bounding_box`: If set to 0, the Plotting module will calculate the geographical distribution of the assets, and adjust the limits of the map accordingly. Alternatively, a user can also specify the minimum / maximum latitude and longitude that should be used in the creation of the map.
- `marker_size`: This attribute can be used to adjust the size of the markers in the map.
- `log_scale`: If set to `True`, it will apply a logarithmic scale on the colour scheme of the map, potentially allowing a better visualization of the variation of the numbers of collapsed assets in the region of interest.

An example of a collapse map is presented in Figure 2.3.

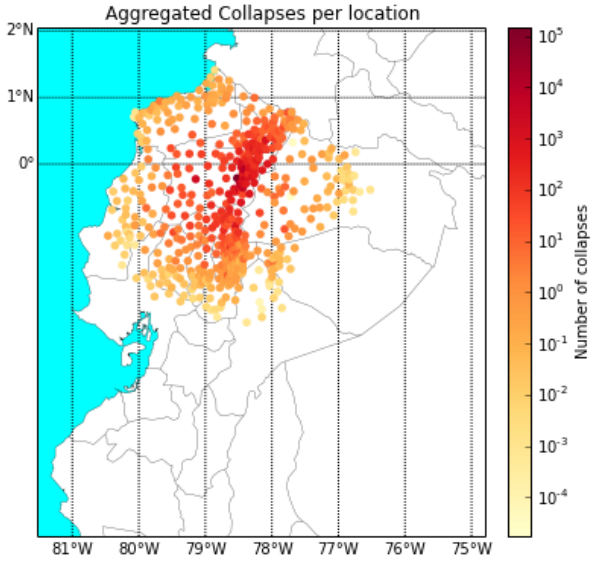

Figure 2.3 – *Spatial distribution of the mean number of collapsed assets.*

2.2 Plotting hazard and loss curves

Using the Classical PSHA-based or Probabilistic Event-based Calculators (Silva et al., 2014a, Pagani et al., 2014) of the OpenQuake-engine, it is possible to calculate seismic hazard curves for a number of locations, or loss exceedance curves considering a collection of spatially distributed assets.

2.2.1 Plotting hazard curves and uniform hazard spectra

A seismic hazard curve defines the probability of exceeding a number of intensity measure levels (e.g. peak ground acceleration or spectral acceleration) for a given interval of time (e.g. 50 years). In order to plot these curves, it is necessary to define the path to the output file in the parameter `hazard_curve_file`. Then, since each output file might contain a large number of hazard curves, it is necessary to establish the location for the hazard curve to be extracted. To visualize the list of locations comprised in the output file, the function `hazard_curves.loc_list` can be employed. Then, the chosen location must be provided to the plotting function (e.g. `hazard_curves.plot("81.213823|29.761172")`). An example of a seismic hazard curve is provided in Figure 2.4.

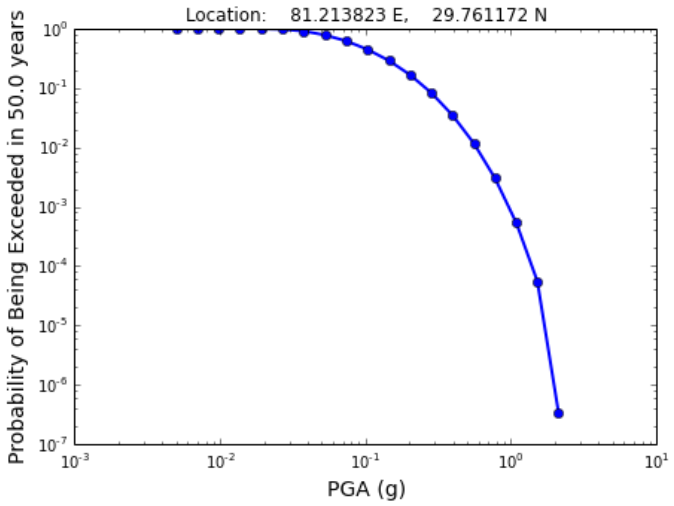

Figure 2.4 – *Seismic hazard curve for peak ground acceleration (PGA).*

To plot uniform hazard spectra (UHS), a similar approach should be followed. The output file containing the uniform hazard spectra should be defined using the parameter `uhs_file`, and then a location must be provided to the plotting function (e.g.`uhs.plot("81.213823|29.761172")`). An example of a uniform hazard spectrum is illustrated in Figure 2.5.

2.2.2 Plotting loss curves

A loss exceedance curve defines the relation between a set of loss levels and the corresponding probability of exceedance within a given time span (e.g. one year). In order to plot these curves, it is necessary to define the location of the output file using the parameter `loss_curves_file`. Since each output file may contain a large number of loss exceedance curves, it is necessary to define for which assets the loss curves will be extracted. The parameter `assets_list` should be employed to define all of the chosen asset ids. These ids can be visualized directly on the loss curve output file, or on the exposure model used for the risk calculations. It is also possible to define a logarithmic scale for the x and y axis using the

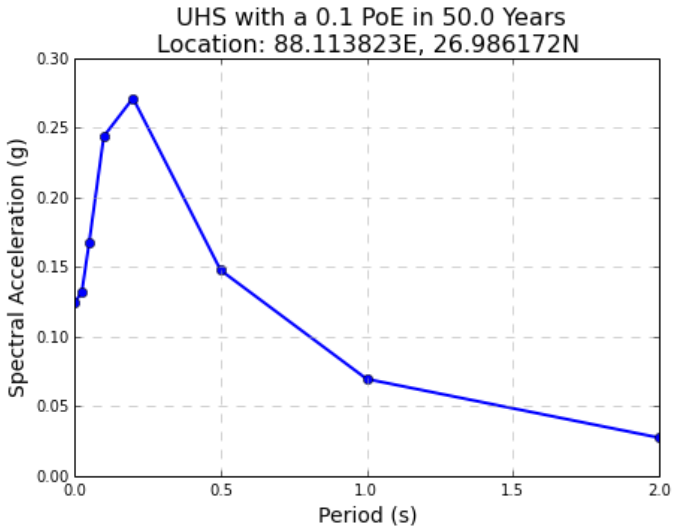

Figure 2.5 – *Uniform Hazard Spectrum for a probability of exceedance of 10% in 50 years.*

parameters `log_scale_x` and `log_scale_y`. A loss exceedance curve for a single asset is depicted in Figure 2.6.

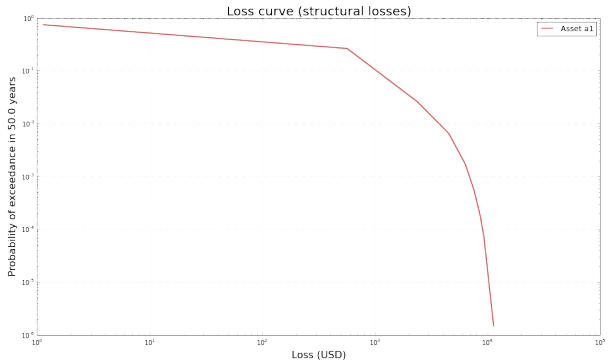

Figure 2.6 – *Loss exceedance curve.*

2.3 Plotting hazard and loss maps

The OpenQuake-engine offers the possibility of calculating seismic hazard and loss (or risk) maps. To do so, it utilizes the seismic hazard or loss exceedance curves, to estimate the corresponding hazard or loss for the pre-defined return period (or probability of exceedance within a given interval of time).

2.3.1 Plotting hazard maps

A seismic hazard map provides the expected ground motion (e.g. peak ground acceleration or spectral acceleration) at each location, for a certain return period (or probability of exceedance within a given interval of time). To plot this type of map, it is necessary to specify the location of the output file using the parameter `hazard_map_file`. An example hazard

map is displayed in Figure 2.4.

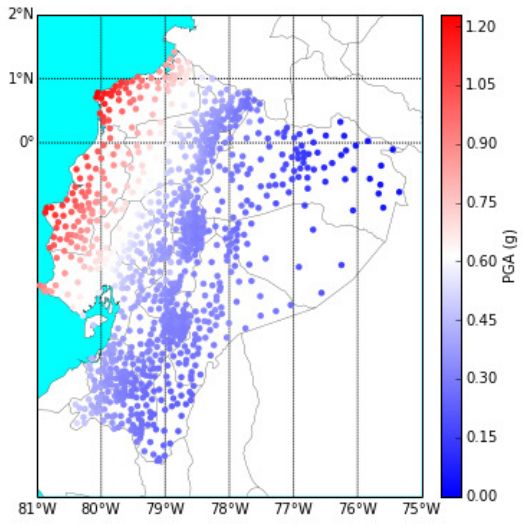

Figure 2.7 – *Seismic hazard map for a probability of exceedance of 10% in 50 years.*

2.3.2 Plotting loss maps

A loss map provides the estimated losses for a collection of assets, for a certain return period (or probability of exceedance within a given interval of time). It is important to understand that these maps are not providing the distribution of losses for a seismic event or level of ground motion with the chosen return period, nor can the losses shown on the map be summed to obtain the corresponding aggregate loss with the same return period. This type of maps is simply providing the expected loss for a specified frequency of occurrence (or return period), for each asset.

To use this feature, it is necessary to define the path of the output file using the parameter `loss_map_file`, as well as the exposure model used to perform the risk calculations through the parameter `exposure_model`. Then, similarly to the method explained in section 2.1.2 for collapse maps, it is possible to follow three approaches to generate the loss maps:

1. Aggregated loss map only.
2. Loss maps per vulnerability class only.
3. Both aggregated and vulnerability class-based.

Then, there are a number of options that can be used to modify the style of the maps. These include the size of the marker of the map (`marker_size`), the geographical limits of the map (`bounding_box`), and the employment of a logarithmic spacing for the colour scheme (`log_scale`). An example loss map for a single vulnerability class is presented in Figure 2.8.

As mentioned in the introductory section, it is also possible to convert any of the maps

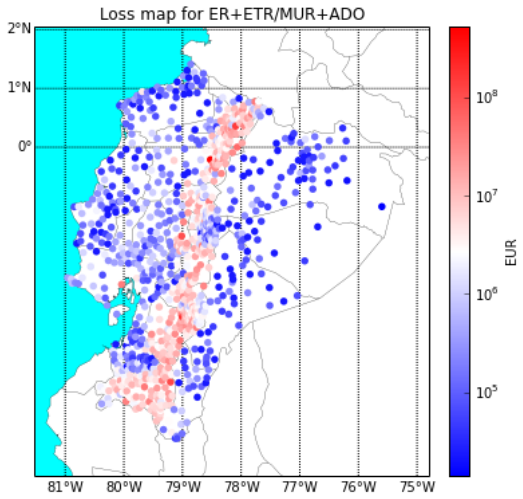

Figure 2.8 – *Loss (economic) map for a probability of exceedance of 10% in 50 years.*

into a format (.csv) that is easily readable by GIS software. To do so, it is necessary to set the parameter `export_map_to_csv` to `True`. As an example, a map containing the average annual losses for Ecuador has been converted to the `csv` format, and introduced into the QGIS software to produce the map presented in Figure 2.9.

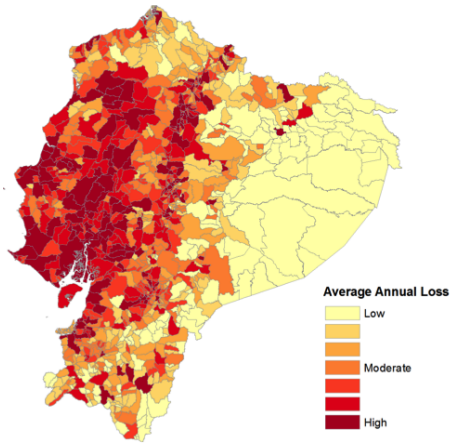

Figure 2.9 – *Average annual (economic) losses for Ecuador.*

3. Additional Risk Outputs

The OpenQuake-engine currently generates the most commonly used seismic hazard and risk results (e.g. hazard maps, loss curves, average annual losses). However, it is recognized that there are a number of other risk metrics that might not be of interest of the general GEM community, but fundamental for specific users. This module of the Risk Modeller's Toolkit aims to provide users with additional risk results and functionalities, based on the standard output of the OpenQuake-engine.

3.1 Deriving Probable Maximum Losses (PML)

The Probabilistic Event-based Risk calculator (Silva et al., 2014a) of the OpenQuake-engine is capable of calculating event loss tables, which contain a list of earthquake ruptures and associated losses. These losses may refer to specific assets, or the sum of the losses from the entire building portfolio (i.e. aggregated loss curves).

Using this module, it is possible to derive probable maximum loss (PML) curves (i.e. relation between a set of loss levels and corresponding return periods), as illustrated in Figure 3.1.

To use this feature, it is necessary to use the parameter `event_loss_table_folder` to specify the location of the folder that contains the set of event loss tables and stochastic event sets. Then, it is also necessary to provide the total economic value of the building portfolio (using the variable `total_cost`) and the list of return periods of interest (using the variable `return_periods`). This module also offers the possibility of saving all of the information in `csv` files, which can be used in other software packages (e.g. Microsoft Excel) for other post-processing purposes. To do so, the parameters `save_elt_csv` and `save_ses_csv` should be set to `True`.

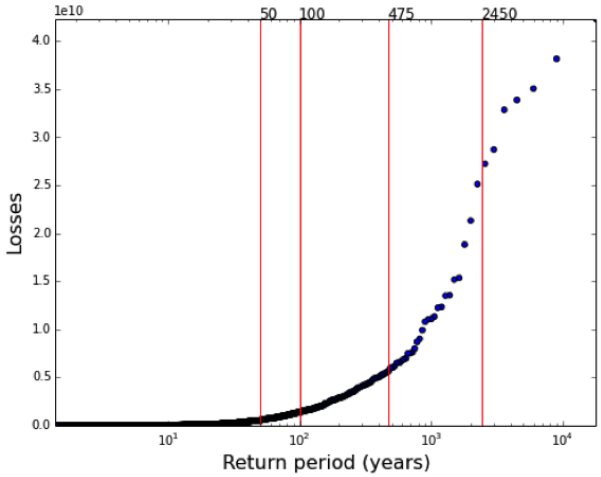

Figure 3.1 – *Probable Maximum Loss (PML) curve.*

3.2 Selecting a logic tree branch

When a non-trivial logic-tree is used to capture the epistemic uncertainty in the source model, or in the choice of ground motion prediction equations (GMPE) for each of the tectonic region types of the region considered, the OpenQuake-engine can calculate hazard curves for each end-branch of the logic-tree individually.

Should a risk modeller wish to just estimate the mean damage or losses of each asset in their exposure model, then they will only need the mean hazard curve. However, if they are interested in aggregating the losses from each asset in the portfolio, they should be using a Probabilistic Event-based Risk Calculator that makes use of spatially correlated ground motion fields per event, rather than hazard curves. For computational efficiency, it is useful to identify a branch of the logic tree that produces the aforementioned hazard outputs that are close to the mean, and this can be done by computing and comparing the hazard curves of each branch. Depending upon the distance of the hazard curve for a particular branch from the mean hazard curve, the risk modeller may can choose the branches for which the hazard curves are closest to the mean hazard curve. This Python script and corresponding IPython notebook allow the risk modeller to list the end-branches for the hazard calculation, sorted in increasing order of the distance of the branch hazard curve from the mean hazard curve. Currently, the distance metric used for performing the sorting is the root mean square distance.

4. Vulnerability

4.1 Introduction

Seismic fragility and vulnerability functions form an integral part of a seismic risk assessment project, along with the seismic hazard and exposure models. Fragility functions for a building or a class of buildings are typically associated with a set of discrete damage states. A fragility function defines the probabilities of exceedance for each of these damage states as a function of the intensity of ground motion. A vulnerability function for a building or a class of buildings defines the probability of exceedance of loss values as a function of the intensity of ground motion. A consequence model, sometimes also referred to as a damage-to-loss model - which describes the loss distribution for different damage states - can be used to derive the vulnerability function for a building or a class of buildings, from the corresponding fragility function.

Empirical methods are often preferred for the derivation of fragility and vulnerability functions when relevant data regarding the levels of physical damage and loss at various levels of ground shaking are available from past earthquakes. However, the major drawback of empirical methods is the highly limited quantity and quality of damage and repair cost data and availability of the corresponding ground shaking intensities from previous events.

The analytical approach to derive fragility and vulnerability functions for an individual structure relies on creating a numerical model of the structure and assessing the deformation behaviour of the modelled structure, by subjecting it to selected ground motion acceleration records or predetermined lateral load patterns. The deformation then needs to be related to physical damage to obtain the fragility functions. The fragility functions can be combined with the appropriate consequence model to derive a vulnerability function for the structure. Fragility and vulnerability functions for a class of buildings (a "building typology") can be obtained by considering a number of structures considered representative of that class. A combination of Monte Carlo sampling followed by regression analysis can be used to obtain

a single "representative" fragility or vulnerability function for the building typology.

The level of sophistication employed during the structural analysis stage is constrained both by the amount of time and the type of information regarding the structure that are available to the modeller. Although performing nonlinear dynamic analysis of a highly detailed model of the structure using several accelerograms is likely to yield a more representative picture of the dynamic deformation behaviour of the real structure during earthquakes, nonlinear static analysis is often preferred due to the lower modelling complexity and computational effort required by static methods. Different researchers have proposed different methodologies to derive fragility functions using pushover or capacity curves from nonlinear static analyses. Several of these methodologies have already been implemented in the RMTK and the following sections of this chapter describe some of these techniques in more detail.

4.2 Definition of input models

The following sections describe the parameters and file formats for the input models required for the various methodologies of the RMTK vulnerability module, including:

- Capacity curves
  - Base shear vs. roof displacement
  - Base shear vs. floor displacements
  - Spectral acceleration vs. spectral displacement
- Ground motion records
- Damage models
  - Strain-based damage criterion
  - Capacity curve-based damage criterion
  - Inter-storey drift-based damage criterion
- Consequence models

4.2.1 Definition of capacity curves

The derivation of fragility models requires the description of the characteristics of the system to be assessed. A full characterisation of a structure can be done with an analytical structural model, but for the use in some fragility methodologies its fundamental features can be adequately described using a pushover curve, which describes the nonlinear behaviour of each input structure subjected to a horizontal lateral load.

Different methodologies require the pushover curve to be expressed with different parameters and to be combined with additional building information (e.g. period of the structure, height of the structure). The following input models have thus been implemented in the Risk Modeller's Toolkit:

1. `Base Shear vs Roof Displacement`
2. `Base Shear vs Floor Displacements`

3. `Spectral acceleration vs Spectral displacement`

Within the description of each fragility methodology provided below, the required input model and the additional building information are specified. Moreover some methodologies give the user the chance to select the input model that fits better to the data at his or her disposal. Considering that different methodologies sharing the same input model may need different parameters, not all the information defined in the input file are necessarily used by each method. The various inputs are currently being stored in a `csv` file (tabular format), as illustrated in the following Sections for each input model.

Once the pushover curves have been defined in the input file and uploaded in the IPython notebook, they can be visualised with the following function:

```
utils.plot_capacity_curves(capacity_curves)
```

4.2.1.1 Base Shear vs Roof Displacement

Some methodologies require the pushover curve to be expressed in terms of Base Shear vs Roof Displacement (e.g. Dolsek and Fajfar 2004 in Section 4.5.2, SPO2IDA in Section 4.5.1). Additional building information is needed to convert the pushover curve (referring to a Multi Degree of Freedom, MDoF, system) to a capacity curve (Single Degree of Freedom, SDoF, system).

When the pushover curve is expressed in terms of Base Shear vs Roof Displacement, the user has to set the `Vb-droof` variable to `TRUE` in the input file, and define whether it is an idealised (e.g. bilinear) or a full pushover curve (i.e. with many pairs of base shear and roof displacement values), setting the variable `Idealised` to `TRUE` or `FALSE` respectively. Then the following information about the structures to be assessed is needed:

1. `Periods`, first period of vibration T_1.
2. `Ground height`, height of the ground floor.
3. `Regular height`, height of the regular floors.
4. `Gamma participation factors`, modal participation factor Γ_1 of the first mode of vibration, normalised with respect to the roof displacement.
5. `Effective modal masses`, effective modal masses M_1^* of the first mode of vibration, normalised with respect to the roof displacement (see Section 4.4.1 for description).
6. `Number storeys`, number of storeys.
7. `Weight`, weight assigned to each structure for the derivation of fragility models for many buildings.
8. `Vbn`, the base shear vector of the n^{th} structure.
9. `droofn`, the roof displacement vector of the n^{th} structure.

Only bilinear and quadrilinear idealisation shapes are currently supported to express the pushover curve in an idealised format, therefore the V_b and d_{roof} vectors should contain 3 or 5 V_b-d_{roof} pairs, respectively, as described in the following lists and illustrated in Figures 4.1 and 4.2.

Bilinear idealisation inputs:

- Displacement vector: displacement at Vb = 0 (d_0), yielding displacement (d_1), ultimate displacement (d_2).
- Base Shear vector: Vb = 0, base shear at yielding displacement (Vb_1), base shear at ultimate displacement ($Vb_2 = Vb_1$).

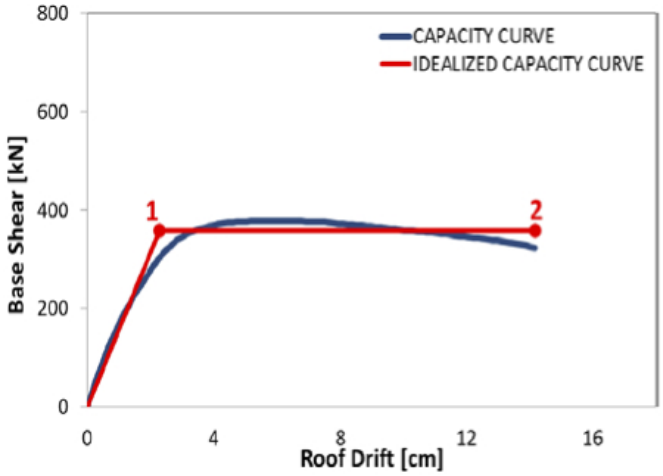

Figure 4.1 – *Inputs for bilinear idealisation of pushover curve.*

Quadrilinear idealisation inputs:

- Displacement vector: displacement at Vb = 0 (d_0), yielding displacement (d_1), displacement at maximum base shear (d_2), displacement at onset of residual force plateau (d_3), ultimate displacement (d_4).
- Base Shear vector: Vb = 0, base shear at yielding displacement (Vb_1), maximum base shear (Vb_2), residual force (Vb_3) and force at ultimate displacement (Vb_4).

An example of an input `csv` file for the derivation of a fragility model for a set of two structures, whose pushover curves are expressed with an idealised bilinear shape, is presented in Table 4.1.

Base shear vs Roof Displacement pushover curves can be idealised with bilinear and quadrilinear formats, according to (FEMA-440, 2005) and to the GEM Vulnerability guidelines ((D'Ayala, 2014)), respectively, using the following function:

```
idealised_capacity = utils.idealisation(idealised_type, capacity_curves)
```

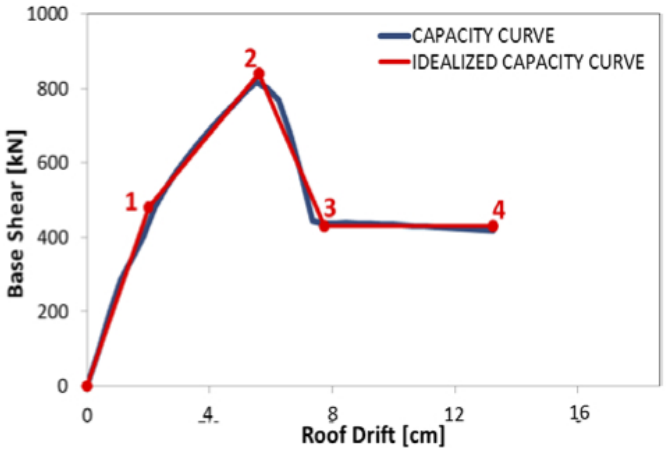

Figure 4.2 – *Inputs for quadrilinear idealisation of pushover curve.*

4.2.1.2 Base Shear vs Floor Displacements

A modification of the previous input model is the Base Shear vs Floor Displacements input type. In this case the conversion to a SDoF capacity curve is still based on the roof displacement, but a mapping scheme between the roof displacement and inter-storey drift at each floor level is also derived, so that the overall deformation state of the structure corresponding to a given roof displacement can be checked.

When the pushover curve is expressed in terms of Base Shear vs Floor Displacements, the user has to set the `Vb-floor` variable to `TRUE` in the input file. Only full pushover curves can be input, therefore the variable `Idealised` should be set to `FALSE`. Then the following information about the structures to be assessed is needed:

1. `Periods`, first period of vibration T_1.
2. `Ground height`, height of the ground floor.
3. `Regular height`, height of the regular floors.
4. `Gamma participation factors`, modal participation factor Γ_1 of the first mode of vibration, normalised with respect to the roof displacement.
5. `Effective modal masses`, effective modal masses M_1^* of the first mode of vibration, normalised with respect to the roof displacement (see Section 4.4.1 for description).
6. `Number storeys`, number of storeys.
7. `Weight`, weight assigned to each structure for the derivation of fragility models for many buildings.
8. `Vbn`, the base shear vector of the n^{th} structure.
9. `dfloorn-1`, the displacement vector of the 1^{st} floor of the n^{th} structure.
10. `dfloorn-2`, the displacement vector of the 2^{nd} floor of the n^{th} structure.
11. `dfloorn-k`, the displacement vector of the k^{th} floor of the n^{th} structure.

Table 4.1 – *Example of a Base Shear-Roof Displacement input model.*

Vb-droof	TRUE		
Vb-dfloor	FALSE		
Sd-Sa	FALSE		
Idealised	TRUE		
Periods [s]	1.61	1.5	
Ground heights [m]	7	6.5	
Regular heights [m]	2.7	3.0	
Gamma participation factors	1.29	1.4	
Effective modal masses	232	230	
Number storeys	6	6	
Weights	0.5	0.5	
Vb1 [kN]	0	2090	2090
droof1 [m]	0	0.1	0.6
Vb2 [kN]	0	1700	1700
droof2 [m]	0	0.08	0.5

An example of an input `csv` file for the derivation of a fragility model for a set of two structures, is presented in Table 4.2.

Base shear vs Floor Displacement pushover curves can be idealised with bilinear and quadrilinear formats, according to (FEMA-440, 2005) and to the GEM Vulnerability guidelines ((D'Ayala, 2014)), respectively, using the following function:

```
idealised_capacity = utils.idealisation(idealised_type, capacity_curves)
```

4.2.1.3 Spectral acceleration vs Spectral displacement

Some methodologies work directly with capacity-curves (Spectral acceleration vs Spectral displacement) of SDoF systems (e.g. N2 in Section 4.6.4, CSM in Section 4.6.5), so that the user can provide his or her own conversion of pushover curves, or can use the "Conversion from MDOF to SDOF" module in Section 4.4.

When the pushover curve is expressed in terms of Spectral acceleration vs Spectral displacement, the user has to set the `Sd-Sa` variable to `TRUE` in the input file. Then the following information about the structures to be assessed is needed:

1. `Periods`, first periods of vibration T_1.
2. `Heights`, heights of the structure.
3. `Gamma participation factors`, modal participation factors Γ_1 of the first mode of vibration.

Table 4.2 – *Example of a Base Shear-Floor Displacements input model.*

Vb-droof	TRUE					
Vb-dfloor	FALSE					
Sd-Sa	FALSE					
Idealised	FALSE					
Periods [s]	1.61	1.5				
Ground heights [m]	7	6.5				
Regular heights [m]	2.7	3.0				
Gamma participation factors	1.29	1.4				
Effective modal masses	232	230				
Number storeys	6	6				
Weights	0.5	0.5				
Vb1 [kN]	0	...	50	94	118	
dfloor1-1 [m]	0.002	...	0.05	0.09	0.11	
dfloor1-2 [m]	0.004	...	0.12	0.16	0.20	
Vb2 [kN]	0	...	79	100	105	150
dfloor2-1 [m]	0.006	...	0.018	0.023	0.05	0.1
dfloor2-2 [m]	0.008	...	0.023	0.030	0.038	0.15

4. `Effective modal masses`, effective modal masses M_1^* of the first mode of vibration, normalised with respect to the roof displacement (see Section 4.4.1 for description).
5. `Sdy`, the yielding spectral displacements.
6. `Say`, the yielding spectral accelerations.
7. `Sdn [m]`, the Sd vector of the n^{th} structure.
8. `San [g]`, the Sa vector of the n^{th} structure.

An example of an input `csv` file for the derivation of a fragility model for a set of three structures, is presented in Table 4.3.

Capacity curves can be idealised with bilinear and quadrilinear formats, according to (FEMA-440, 2005) and to the GEM Vulnerability guidelines ((D'Ayala, 2014)), respectively, using the following function:

```
idealised_capacity = utils.idealisation(idealised_type, capacity_curves)
```

4.2.2 Definition of ground motion records

The record-to-record variability is one of the most important sources of uncertainty in fragility assessment. In the Direct Nonlinear Static Procedures implemented in the Risk Modeller's

Table 4.3 – *Example of a Spectral acceleration-Spectral displacement input model.*

Vb-droof	FALSE		
Vb-dfloor	FALSE		
Sd-Sa	TRUE		
Periods [s]	1.52	1.63	1.25
Heights [m]	6	6	6
Gamma participation factors	1.24	1.22	1.27
Effective modal masses	232	230	240
Sdy [m]	0.0821	0.0972	0.0533
Say [g]	0.143	0.14723	0.13728
Sd1 [m]	0	0.0821	0.238
Sa1 [g]	0	0.143	0.143
Sd2 [m]	0	0.0972	0.264
Sa2 [g]	0	0.14723	0.14723
Sd3 [m]	0	0.0533	0.0964
Sa3 [g]	0	0.13728	0.13728

Toolkit (see Section 4.5), this source of uncertainty is directly introduce in the fragility estimates, based on previous nonlinear analyses. In the Record-based Nonlinear Static Procedures (Section 4.6) or in the Nonlinear Time History Analysis of Single Degree of Freedom Oscilators (Section 4.7), users have the possibility of introducing their own ground motion records. Each accelerogram needs to be stored in a `csv` file (tabular format), as depicted in Table 4.4. This file format uses two columns, in which the first one contains the time (in seconds) and the second the corresponding acceleration (in g).

In order to load a set of ground motion records into the Risk Modeller's Toolkit, it is necessary to import the module `utils`, and specify the location of the folder containing all of the ground motion records using the parameter `gmrs_folder`. Then, the collection of records can be loaded using the following command:

```
gmrs = utils.read_gmrs(gmrs_folder)
```

One the ground motion records have been loaded, it is possible to calculate and plot the acceleration and displacement response spectra. To do so, it is necessary to specify the minimum and maximum period of vibration to be used in the calculations, using the variables `minT` and `maxT`, respectively. Then, the following command must be used:

Table 4.4 – *Example of a ground motion record.*

0.01	0.0211
0.02	0.0292
0.03	0.0338
0.04	0.0274
0.05	0.0233
0.06	0.0286
0.07	0.0292
0.08	0.0337
0.09	0.0297
0.1	0.0286
...	...

```
minT = 0.1
maxT = 2
utils.plot_response_spectra(gmrs,minT,maxT)
```

This will generate three plots: 1) spectral acceleration (in g) versus period of vibration (in sec); 2) spectral displacement (in m) versus period of vibration (in sec); and spectral acceleration (in g) versus spectral displacement (in m). Figure 4.3 illustrates the first two plots for one hundred ground motion records.

4.2.3 Definition of damage model

The derivation of fragility models requires the definition of a criterion to allocate one (or multiple) structures into a set of damage states, according to their nonlinear structural response. These rules to relate structural response with physical damage can vary significantly across the literature. Displacement-based methodologies frequently adopt the strain of the concrete and steel (e.g. Borzi et al., 2008b; Silva et al., 2013). The vast majority of the methodologies that require equivalent linearisation methods or nonlinear time history analysis adopt inter-storey drifts (e.g. Vamvatsikos and Cornell, 2005; Rossetto and Elnashai, 2005), or spectral displacement calculated based on a pushover curve (e.g. Erberik, 2008; Silva et al., 2014c). The various rules dictated by the damage model are currently being stored in a `csv` file (tabular format), as described below for each type of model.

4.2.3.1 Strain-based damage criterion

Displacement-based (Crowley et al., 2004) or mechanics-based (Borzi et al., 2008b) methodologies use strain levels to define a number of limit states. Thus, for each limit state, a strain

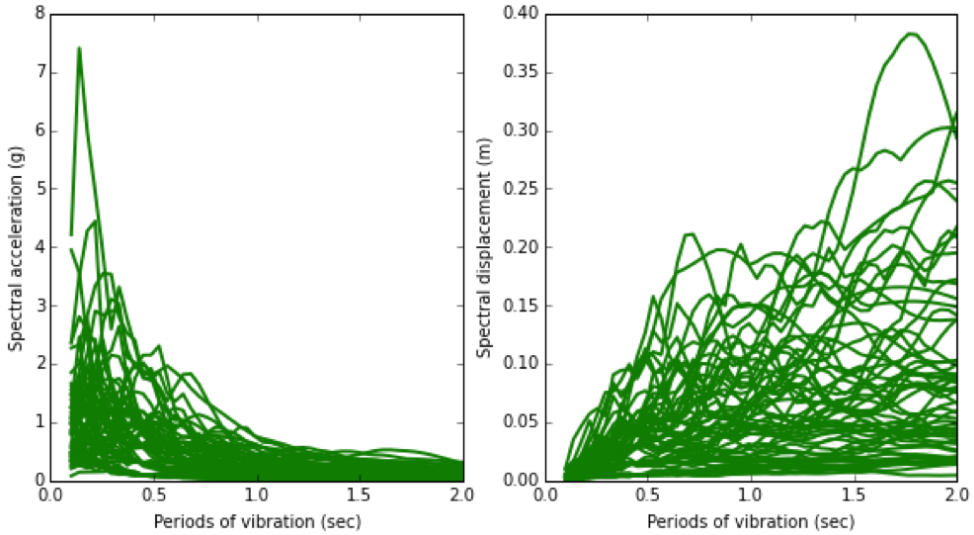

Figure 4.3 – *Response spectra in terms of spectral acceleration versus period of vibration (left) and spectral displacement versus period of vibration (right).*

for the conrete and steel should be provided. It is recognized that there is a large uncertainty in the allocation of a structure into a physical damage state based on its structural response. Thus, the Risk Modeller's Toolkit allows the representation of the damage criterion in a probabilistic manner. This way, the parameter that establishes the damage threshold can be defined by a mean, a coefficient of variation and a probabilistic distribution (normal, lognormal or gamma) (Silva et al., 2013). This approach is commonly used to at least assess the spectral displacement at the yielding point (`Sdy`) and for the ultimate capacity (`Sdu`). Other limit states can also be defined using other strain levels (e.g. Crowley et al., 2004), or a fraction of the yielding or ultimate displacement. For example, Borzi et al., 2008b defined light damage and collapse through the concrete and steel strains, and significant damage as $^3/_4$ of the ultimate displacement (`Sdu`).

To use this damage criteria, it is necessary to define the parameter `Type` as `strain dependent` within the damage model file. Then, each limit state needs to be defined by a name (e.g. light damage), type of criterion and the adopted probabilistic model. Using the damage criteria described above (by Borzi et al., 2008b), an example of a damage model is provided in Table 4.5. In this case, the threshold for light damage is defined at the yielding point, which in return is calculated based on the yielding strain of the steel. The limit state for collapse is computed based on the mean strain in the concrete and steel (0.0075 and 0.0225, respectively) and the a coefficient of variation (0.3 and 0.45, respectively). The remaining limit state (significant damage), is defined as fraction (0.75) of the ultimate displacement (collapse).

Table 4.5 – *Example of a strain dependent damage model*

Type	strain dependent			
Damage States	Criteria	distribution	mean	cov
light damage	Sdy	lognormal		0
significant damage	fraction Sdu	lognormal	0.75	0
collapse	strain	lognormal	0.0075 0.0225	0.30 0.45

4.2.3.2 Capacity curve-based damage criterion

Several existing studies (e.g. Erberik, 2008; Silva et al., 2014c; Casotto et al., 2015) have used capacity curves (spectral displacement versus spectral acceleration) or pushover curves (roof displacement versus base shear) to define a set of damage thresholds. In the vast majority of these studies, the various limit states are defined as a function of the displacement at the yielding point (`Sdy`), the maximum spectral acceleration (or base shear), and / or of the ultimate displacement capacity (`Sdu`). For this reason, the mechanism that has been implemented in the RMTK is considerably flexible, and allows users to define a set of limit states following the options below:

1. `fraction Sdy`: this limit state is defined as a fraction of the displacement at the yielding point (`Sdy`) (e.g. 0.75 of `Sdy`)
2. `Sdy` this limit state is equal to the displacement at the yielding point, usually marking the initiation of structural damage.
3. `max Sa` this limit state is defined at the displacement at the maximum spectral acceleration.
4. `mean Sdy Sdu` this limit state is equal to the mean between the displacement at the yielding point (`Sdy`) and ultimate displacement capacity (`Sdu`).
5. `X Sdy Y Sdu` this limit state is defined as the weighted mean between the displacement at the yielding point (`Sdy`) and ultimate displacement capacity (`Sdu`). X represents the weight associated with the former displacement, and Y corresponds to the weight of the latter (e.g. 1 Sdy 4 Sdu).
6. `fraction Sdu` this limit state is defined as a fraction of the ultimate displacement capacity (`Sdu`) (e.g. 0.75 of `Sdy`)
7. `Sdu` this limit state is equal to ultimate displacement capacity (`Sdu`), usually marking the point beyond which structural collapse is assumed to occur.

In order to create a damage model based on this criterion, it is necessary to define the parameter `Type` as `capacity curve dependent`. Then, each limit state needs to be defined by a name (e.g. slight damage), type of criterion (as defined in the aforementioned

list) and a potential probabilistic model (as described in the previous subsection). An example of a damage model considering all of the possible options described in the previous list is presented in Table 4.6, and illustrated in Figure 4.4. Despite the inclusion of all of the options, a damage model using this approach may use only a few of these criteria. Moreover, some of the options (namely the first, fifth and sixth) may by used multiple times.

Table 4.6 – *Example of a capacity curve dependent damage model.*

Type	capacity curve dependent			
Damage States	Criteria	distribution	Mean	Cov
LS1	fraction Sdy	lognormal	0.75	0.0
LS2	Sdy	normal		0.0
LS3	max Sda	normal		0.0
LS4	mean Sdy Sdu	normal		0.0
LS5	1 Sdy 2 Sdu	normal		0.0
LS6	fraction Sdu	normal	0.85	0.0
LS7	Sdu	normal		0.0

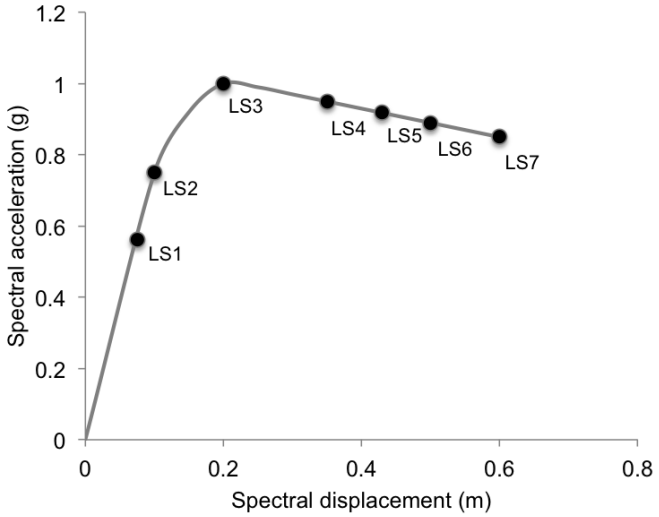

Figure 4.4 – *Representation of the possible options for the definition of the limit states using a capacity curve.*

4.2.3.3 Spectral displacement-based damage criterion

In many methodologies for the definition of the seismic vulnerability of structures an equivalent Single Degree Of Freedom (SDOF) system is subjected to multiple analyses instead of the complex Multi Degree Of Freedom (MDOF) system. The capacity of the structure is thus expressed in terms of spectral acceleration vs spectral displacement. The easiest way to

allocate the structure into a damage state is that of comparing spectral displacement demand with spectral displacement damage thresholds. A damage model has been implemented in the RMTK that allows to introduce directly spectral displacement damage thresholds, and an example of input file is provided in Table 4.7.

Table 4.7 – *Example of a spectral displacement based damage model*

Type	spectral displacement		
Damage States	distribution	Median	Dispersion
Slight	lognormal	0.01	0.0
Moderate	lognormal	0.05	0.1
Exyensive	lognormal	0.1	0.2
Collapse	lognormal	0.2	0.25

4.2.3.4 Inter-storey drift-based damage criterion

Maximum inter-storey drift is recognised by many researchers (e.g. Vamvatsikos and Cornell, 2005; Rossetto and Elnashai, 2005) as a good proxy of the damage level of a structure, because it can detect the storey by storey state of deformation as opposed to global displacement.

The use of this damage model is quite simple: the parameter Type in the csv file should be set to `interstorey drift` and inter-storey drift thresholds need to be defined for each damage state, in terms of median value and dispersion.

The probabilistic distribution of the damage thresholds implemented so far is lognormal. A different set of thresholds can be assigned to each structure, as in the example provided in Table 4.8, but also a single set can be defined for the entire building population to be assessed.

When a vulnerability assessment methodology uses an equivalent SDOF system instead of the complex MDOF system it is still possible to define an inter-storey drift-based damage model for the MDOF system and introduce a relationship to convert inter-storey drift to spectral displacement damage thresholds. The conversion file containing the relationship between the maximum inter-storey drift along the building height and the spectral displacement of the equivalent SDOF system can be obtained using the "Conversion from MDOF to SDOF" module (see Section 4.4). If this option wants to be enabled the variable `deformed shape path` should be set to TRUE in the csv input file, and the name of the conversion file should be specified in the next cell, as shown in Table 4.8. The conversion file should be placed in the same folder where the damage model is located. An example of conversion file is provided in Table 4.9 for the vulnerability assessment of a building population.

Table 4.8 – *Example of a inter-storey drift based damage model*

Type	interstorey drift				
deformed shape path	TRUE	ISD-Sd.csv			
Damage States	distribution	Median	Dispersion	Median	Dispersion
LS1	lognormal	0.001	0.0	0.001	0.0
LS2	lognormal	0.01	0.2	0.015	0.0
LS3	lognormal	0.02	0.2	0.032	0.0

Table 4.9 – *Example of file containing the relationship between maximum inter-storey drift and spectral displacement for each structure of the building population*

ISD	0.0004	0.0009	0.0013	0.0018	0.0023
Sd [m]	0.003	0.006	0.009	0.012	0.015
ISD	0.0003	0.0011	0.0014	0.002	0.0025
Sd [m]	0.002	0.005	0.011	0.013	0.014
ISD	...	...	...	...	...
Sd [m]	..	...	...	...	...

4.2.4 Consequence model

A consequence model (also known as damage-to-loss model), establishes the relation between physical damage and a measure of fraction of loss (i.e. the ratio between repair cost and replacement cost for each damage state). These models can be used to convert a fragility model (see Section 4.8.1) into a vulnerability function (see Section 4.8.1).

Several consequence models can be found in the literature for countries such as Greece (Kappos et al., 2006), Turkey (Bal et al., 2010), Italy (Di Pasquale and Goretti, 2001) or the United States (FEMA-443, 2003). The damage scales used by these models may vary considerably, and thus it is necessary to ensure compatibility with the fragility model. Consequence models are also one of the most important sources of variability, since the economical loss (or repair cost) of a group of structures within the same damage state (say moderate) can vary significantly. Thus, it is important to model this component in a probabilistic manner.

In the Risk Modeller's Toolkit, this model is being stored in a `csv` file (tabular format), as illustrated in Table 4.10. In the first column, the list of the damage states should be provided. Since the distribution of loss ratio per damage state can be modelled using a probabilistic model, the second column must be used to specify which statistical distribution should be

used. Currently, `normal`, `lognormal` and `gamma` distributions are supported. The mean and associated coefficient of variation (cov) for each damage state must be specified on the third and fourth columns, respectively. Finally, each distribution should be truncated, in order to ensure consistency during the sampling process (e.g. avoid negative loss ratios in case a `normal` distribution is used, or values above 1). This variability can also be neglected, by setting the coefficient of variation (cov) to zero.

Table 4.10 – *Example of a consequence model.*

Damage States	distribution	Mean	Cov	A	B
Slight	normal	0.1	0.2	0	0.2
Moderate	normal	0.3	0.1	0.2	0.4
Extensive	normal	0.6	0.1	0.4	0.8
Collapse	normal	1	0	0.8	1

4.3 Model generator

The methodologies currently implemented in the Risk Modeller's Toolkit require the definition of the capacity of the structure (or building class) using a capacity curve (or pushover curve). These curves can be derived using software for structural analysis (e.g. SeismoStruct, OpenSees); experimental tests in laboratories; observation of damage from previous earthquakes; and analytical methods. The Risk Modeller's Toolkit provides two simplified methodologies (DBELA - Silva et al., 2013; SP-BELA - Borzi et al., 2008b) to generate capacity curves, based on the geometrical and material properties of the building class (thus allowing the propagation of the building to building vartiability). Moreover, it also features a module to generate sets of capacity curves, based on the median curve (believed to be representative of the building class) and the expected variability at specific points of the reference capacity curve.

4.3.1 Generation of capacity curves using DBELA

The Displacement-based Earthquake Loss Assessment (DBELA) methodology permits the calculation of the displacement capacity of a collection of structures at a number of limit states (which could be structural or non-structural). These displacements are derived based on the capacity of an equivalent SDoF structure, following the principles of structural mechanics (Crowley et al., 2004; Bal et al., 2010; Silva et al., 2013).

The displacement at the height of the centre of seismic force of the original structure (H_{CSF}) can be estimated by multiplying the base rotation by the height of the equivalent SDoF structure (H_{SDOF}), which is obtained by multiplying the total height of the actual structure (H_T) by an effective height ratio (ef_h), as illustrated in Figure 4.5:

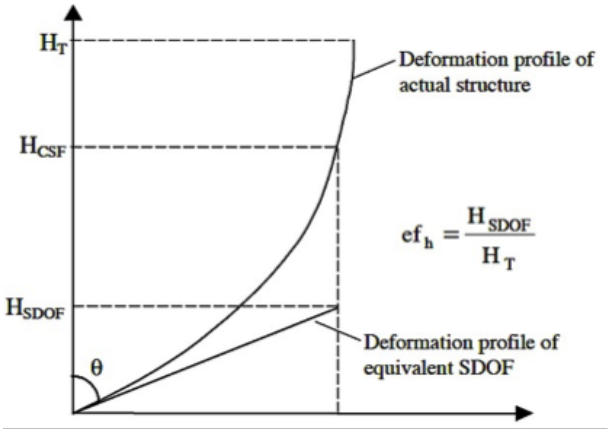

Figure 4.5 – *Definition of effective height coefficient Glaister and Pinho, 2003.*

Pinho et al., 2002 and Glaister and Pinho, 2003 proposed formulae for estimating the effective height coefficient for different response mechanisms. For what concerns the beam sway mechanism (or distributed plasticity mechanism, as shown in Figure 4.6), a ratio of 0.64 is proposed for structures with 4 or less storeys, and 0.44 for structures with 20 or more storeys. For any structures that might fall within these limits, linear interpolation should be employed. With regards to the column-sway mechanism (or concentrated plasticity mechanism, as shown in Figure 4.6), the deformed shapes vary from a linear profile (pre-yield) to a non-linear profile (post-yield). As described in Glaister and Pinho, 2003, a coefficient of 0.67 is assumed for the pre-yield response and the following simplified formula can be applied post-yield (to attempt to account for the ductility dependence of the effective height post-yield coefficient):

$$ef_h = 0.67 - 0.17\frac{\epsilon_{s(LS_i)} - \epsilon_y}{\epsilon_{s(LS_i)}} \tag{4.1}$$

The displacement capacity at different limit states (either at yield (δ_y) or post-yield ($\delta_{(LSi)}$) for bare frame or infilled reinforced concrete structures can be computed using simplified formulae, which are distinct if the structure is expected to exhibit a beam- or column-sway failure mechanism. These formulae can be found in Bal et al., 2010 or Silva et al., 2013, and their mathematical formulation is described in detail in Crowley et al., 2004.

In order to estimate whether a given frame will respond with a beam- or a column-sway mechanism it is necessary to evaluate the properties of the storey. A deformation-based index (R) has been proposed by Abo El Ezz, 2008 which reflects the relation between the stiffness of the beams and columns. This index can be computed using the following formula:

$$R = \frac{h_b/l_b}{h_c/l_c} \tag{4.2}$$

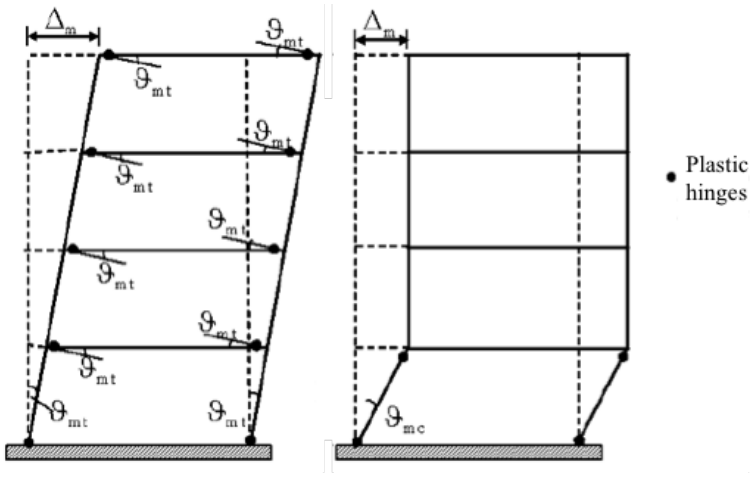

Figure 4.6 – *Deformed profiles for beam-sway (left) and column-sway (right) mechanismsPaulay and Priestley, 1992.*

Where l_c stands for the column length. Abo El Ezz, 2008 proposed some limits for this index applicable to bare and fully infilled frame structures, as described in Table 4.11.

Table 4.11 – *Limits for the deformation-based sway index proposed by Abo El Ezz, 2008*

Building Typology	Beam sway	Column sway
Bare frames	R≤1.0	R>1.5
Fully infilled frames	R≤1.0	R>1.0

The calculation of the corresponding spectral acceleration is performed by assuming a perfectly elasto-plastic behaviour. Thus, the spectral displacement for the yielding point is used to derive the associated acceleration through the following formula:

$$Sa_i = \frac{4\pi^2 Sd_i}{T_y^2} \tag{4.3}$$

Where T_y stands for the yielding period which can be calculated using simplified formulae (e.g. Crowley and Pinho, 2004; Crowley and Pinho, 2006), as further explained in Section 4.6.6. Due to the assumption of the elasto-plastic behaviour, the spectral acceleration for the remaining limit states (or spectral displacements) will be the same (see Figure 4.7).

In order to use this methodology it is necessary to define a building model, which specifies the probabilistic distribution of the geometrical and material properties. This information is currently stored in a *csv* file (tabular format), as presented in Table 4.12.

The `structure type` can be set to `bare frame` or `infilled frame`, while the parameter `ductility` can be equal to `ductile` or `non-ductile`. The variable `number of storeys`

Table 4.12 – *Example of a building model compatible with the DBELA method.*

Structure type	bare frame				
ductility	ductile				
number of storeys	3				
steel modulus	lognormal	210000	0.01	0	inf
steel yield strength	normal	371.33	0.24	0	inf
ground floor height	discrete	2.8 3.1 3.2	0.48 0.15 0.37	0	inf
regular floor height	lognormal	2.84	0.08	0	inf
column depth	lognormal	0.45	0.12	0.3	0.6
beam length	gamma	3.37	0.38	0	inf
beam depth	lognormal	0.6	0.16	0.4	0.8

must be equal to an integer defining the number of floors of the building class. The following parameters (`steel modulus` , `steel yield strength` , `ground floor height` , `regular floor height` , `column depth` , `beam length` , `beam depth`) represent the geometrical and material properties of the building class, and can defined in a probabilistic manner. Currently, three parametric statistical models have been implemented (`normal` , `lognormal` and `gamma`), as well as the discrete (i.e. probability mass function) model (`discrete`). The model that should be used must be specified on the second column. Then, for the former type of models (parametric), the mean and coefficient of variation should be provided in the third and fourth columns, respectively. For the discrete model, the central values of the bins and corresponding probabilities of occurrence should be defined in the third and fourth columns, respectively. The last two columns can be used to truncate the probabilistic distribution between a minimum (fifth column) and maximum (sixth column) values. In order to define a parameter in a deterministic manner (i.e. no variability), the coefficient of variation for the associated attribute can be set to zero, and the same value (mean) will be used repeatedly.

The location of the building model and the damage model (see Section 4.2.3) should be specified in the variables `building_model` and the `damage_model`, respectively. The number of capacity curves that should be generated must be defined using the parameter `no_assets`. Then, after importing the module `DBELA`, the set of capacity curves can be generated using the following commands:

```
building_class_model = DBELA.read_building_class_model(building_model)
assets = DBELA.generate_assets(building_class_model,no_assets)
capacity_curves = DBELA.generate_capacity_curves(assets,damage_model)
```

The first function (`read_building_class_model`) processes the information about the building class; the second function (`generate_assets`) uses a Monte Carlo sampling process to generate a set of synthetic structural models (each one with unique geometrical and material properties); and the final function (`generate_capacity_curves`) combines the generated assets with the damage model to calculate a capacity curve per structure. Figure 4.7 presents a collection of capacity curves generated using this methodology.

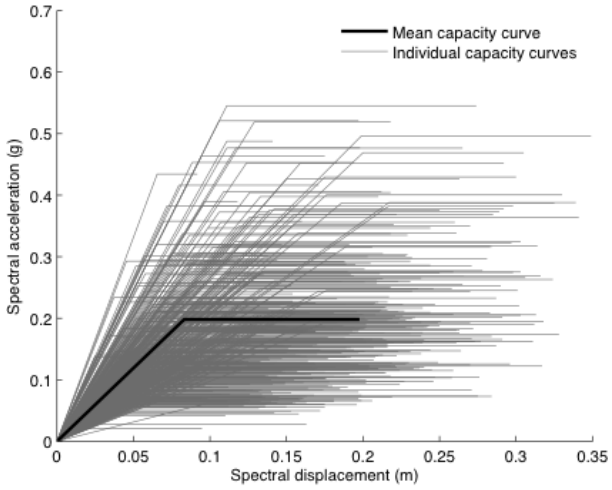

Figure 4.7 – *Capacity curves for reinforced concrete bare frames generated using the DBELA methodology.*

4.3.2 Generation of capacity curves using SP-BELA displacement equations

The Simplified Pushover-based Earthquake Loss Assessment methodology (Borzi et al., 2008b) allows the calculation of the displacement capacity (i.e. spectral displacement) and collapse multiplier (i.e. spectral acceleration) using a mechanics-based procedure, similar to what has been proposed by Cosenza et al. (2005). The methodology implemented in the Risk Modeller's Toolkit currently only uses the displacement capacity equations of SP-BELA for reinforced concrete frames (and does not yet estimate the collapse multiplier using the simplified pushover approach of SP-BELA). The full SP-BELA methodology will be implemented in the future, together with the similar process that has also been proposed for masonry structures (Borzi et al., 2008a).

The original methodology as proposed by Borzi et al. (2008b) considered three limit states (light damage corresponding to the yielding point, significant damage, and collapse). However, other limit states can be considered, granted that the user provides the information required to establish each limit state.

The spectral displacement at each limit state is calculated by firstly assessing the expected chord rotation at the associated damage threshold, and then multiplying this rotation by

the height of the equivalent single degree of freedom (SDoF) system. The rotation at the yielding point (Θ_y)can be calculated using the formula below, as proposed by Cosenza et al. (2005) and Panagiotakos and Fardis (2001):

$$\Theta_y = \phi_y \frac{L_V}{3} + 0.0013\left(1 + 1.5\frac{h}{L_V}\right) + 0.13\phi_y \frac{d_b f_y}{\sqrt{f_c}} \tag{4.4}$$

Where ϕ_y stands for the yield curvature of the section, L_V represents the shear span (which for columns can be assumed as half of the inter-storey height (Borzi et al., 2008b)), h stands for the section height, d_b is the longitudinal bar diameter, and f_y and f_c are the strength of the steel and concrete (in MPa), respectively. The yield curvature (ϕ_y) can be calculated using the equation proposed by Priestley et al., 2007:

$$\phi_y = 2.14\frac{\epsilon_y}{h} \tag{4.5}$$

Where ϵ_y represents the yield strain of the longitudinal rebars.

For what concerns the ultimate rotation capacity (Θ_u), Panagiotakos and Fardis, 2001 proposed the following formula:

$$\Theta_u = \frac{1}{\gamma_{el}}\left[\Theta_y + (\phi_u - \phi_y)L_{pl}\left(1 - \frac{0.5L_p l}{L_V}\right)\right] \tag{4.6}$$

Where γ_{el} is 1.5 for the primary structural elements and 1 for all others (Borzi et al., 2008b), ϕ_u stands for the ultimate curvature, and $L_p l$ is the plastic hinge length, which can be computed through the following equation:

$$\phi_u = \frac{\epsilon_{cu} + \epsilon_{su}}{h} \tag{4.7}$$

Where ϵ_{cu} and ϵ_{su} are the ultimate concrete and steel strain, respectively. These two parameters depend on the level of confinement of the reinforced concrete and expected ductility, and reasonable ranges can be found in Calvi, 1999, Crowley et al., 2004 or Bal et al., 2010.

Other rotation thresholds (corresponding to other limit states) can also be calculated, either through the definition of concrete and steel strains (e.g. Crowley et al., 2004), or as a fraction of the previously described rotations. For example, Borzi et al., 2008b defined the limit state for significant damage as $^3/_4$ of the ultimate rotation capacity (Θ_u).

As previously mentioned, the spectral displacement at each limit state can be calculated by multiplying the respective rotation by the height of the equivalent SDoF system. This

height is calculated by multiplying the total height of the structure (H_T) by an effective height ratio (ef_h). The calculation of this ratio depends on the expected failure mechanism (beam sway or column sway - see Figure 4.6), and its calculation has been explained in Section 4.3.1. For the yielding point, the spectral displacement can then be calculated as follows:

$$\Delta_y = \Theta_y ef_h H_T \tag{4.8}$$

The displacement capacity at the remaining limit states depends on the expected failure mechanisms (as defined in Figure 4.6), and can be calculated using the following formulae:

For beam sway:

$$\Delta_{LS_i} = \Delta_y + (\Theta_{LS_i} - \Theta_y) ef_h H_T \tag{4.9}$$

for column sway:

$$\Delta_{LS_i} = \Delta_y + (\Theta_{LS_i} - \Theta_y) h_p \tag{4.10}$$

Where h_p stands for the ground storey height.

The calculation of the spectral aceleration for each limit states follows the same procedure explained in the previous Section, in which an elasto-plastic behaviour is assumed, and the following formula is employed to calculate acceleration at the yielding point:

$$Sa_i = \frac{4\pi^2 Sd_i}{T_y^2} \tag{4.11}$$

Where T_y stands for the yielding period which is currently calculated using simplified formulae (e.g. Crowley and Pinho, 2004; Crowley and Pinho, 2006), as further explained in Section 4.6.6, rather than based on estimating the collapse multiplier as in the original SP-BELA methodology.

To use this methodology it is necessary to define a building model, which specifies the probabilistic distribution of the geometrical and material properties. This information is currently stored in a *csv* file (tabular format), as presented in Table 4.13.

The definition of each parameter follows the same approach described in the previous Section for the Displacement-based Earthquake Loss Assessment (DBELA) methodology.

The location of the building model and the damage model (see Section 4.2.3) should be specified in the variables `building_model` and the `damage_model`, respectively. The number of capacity curves that should be generated must be defined using the parameter `no_assets`. Then, after importing the module `SPBELA`, the set of capacity curves can be generated using the following commands:

Table 4.13 – *Example of a building model compatible with the SPBELA method*

Structure type	bare frame				
ductility	ductile				
number of storeys	3				
steel modulus	lognormal	210000	0.01	0	∞
concrete strength	normal	30	0.05	0	∞
steel bar diameter	discrete	0.01	1	0	∞
steel yield strength	normal	371.33	0.24	0	∞
ground floor height	discrete	2.8 3.1 3.2	0.48 0.15 0.37	0	∞
regular floor height	lognormal	2.84	0.08	0	∞
column depth	lognormal	0.45	0.12	0.3	0.6
beam length	gamma	3.37	0.38	0	∞
beam depth	lognormal	0.6	0.16	0.4	0.8

```
building_class_model = SPBELA.read_building_class_model(building_model)
assets = SPBELA.generate_assets(building_class_model,no_assets)
capacity_curves = SPBELA.generate_capacity_curves(assets,damage_model)
```

The first function (`read_building_class_model`) processes the information about the building class; the second function (`generate_assets`) uses a Monte Carlo sampling process to generate a set of synthetic structural models (each one with unique geometrical and material properties); and the final function (`generate_capacity_curves`) combines the generated assets with the damage model to calculate a capacity curve per structure. Figure 4.8 presents a collection of capacity curves generated using this methodology.

4.3.3 Generation of capacity curves using point dispersion

This module of the Risk Modeller's Toolkit allows the generation of a large number of capacity curves, based on a single median curve, and information regarding the expected dispersion at each point of the curve.

As an example, in a study carried out by Silva et al., 2014b, several capacity curves were derived for 100 reinforced concrete frames with 4 storeys without seismic provisions, as illustrated in Figure 4.9. In addition to the mean and median capacity curves, an additional capacity curve was also derived using the mean geometrical and material properties of the building class of interest. From this distribution of curves, it is possible to assess the probabilistic distribution of a number of specific points, such as the yielding or ultimate

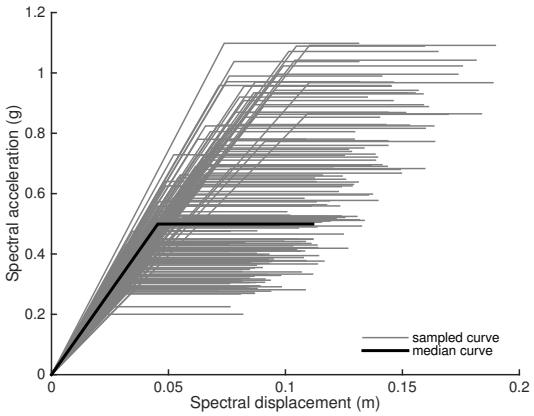

Figure 4.8 – *Capacity curves for reinforced concrete bare frames generated using the SPBELA methodology.*

capacities. Then, once adequate statistical models have been defined for these specific points, it is possible to reproduce the the distribution of capacity curves using a Monte Carlo approach. If one assumes that such distribution could be applicable to other building classes, then it is possible to produce sets of capacity curves (as a way to propagate the building-to-building variability), based on this dispersion and a capacity curve believed to be representative of the median capacity of the building class.

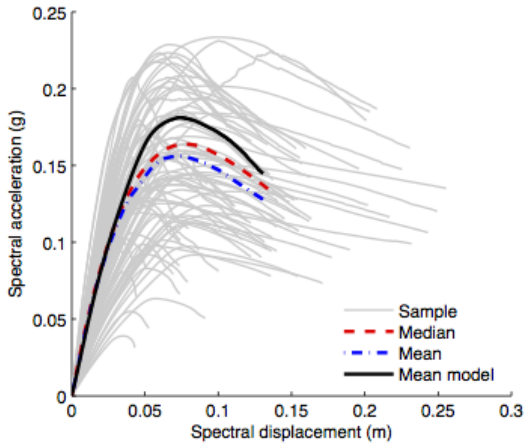

Figure 4.9 – *Capacity curves using a displacement-based adaptive pushover approach (Silva et al., 2014b).*

To use this methodology, it is necessary to import the module `point_dispersion`. Then, the spectral acceleration and displacement of the median capacity curve should be defined in the vectors `Sa` and `Sd`, respectively. For each point in these vectors, it is necessary to provide the corresponding variability (as a form of a coefficient of variation), in the variables `Sa_cov` and `Sd_cov`. The type of probabilistic distribution that should be used must be defined using the parameter `type_of_dist`. Currently this module allows `normal` and `lognormal`. An example of the definition of these variables is provided below. In this case,

the capacity curves are being defined using three points (the yielding capacity, the point of maximum force and the ultimate displacement).

```
Sa = [0.25, 0.45, 0.35]
Sa_cov = [0.25, 0.30, 0.30]
Sd =  [0.10, 0.20, 0.50]
Sd_cov = [0.20, 0.30, 0.45]
type_dist = 'normal'
```

Within this methodology, it is also possible to specify the correlation in the spectral acceleration (`Sa_corr`) and spectral displacement (`Sd_corr`). If these correlation factors are set to zero, then the sampling process is carried out independently. On the other hand, if these parameters are set to one, then a full correlation is assumed, which means, for example, if a higher displacement is sampled for the yielding point, then a large displacement will also be sampled for the remaining points. Values between zero and one will lead to intermediate situations. It is also possible to control the correlation between the spectral acceleration and displacement, using the variable `SaSd_corr`. The dispersion of the synthetic capacity curves can also be controlled through the allowed number of standard deviations above or below the median capacity curve. This aspect is controlled by the parameter `truncation`, which should be equal to the maximum number of allowed standard deviations. for example, a value equal to 1 signifies that all of the generated capacity curves will be within one standard deviation of the median capacity curve. Finally, the number of capacity curves that will be generated needs to be specified using the variable `no_cc`. Figure 4.10 illustrates 100 capacity curves generated using the data provided above.

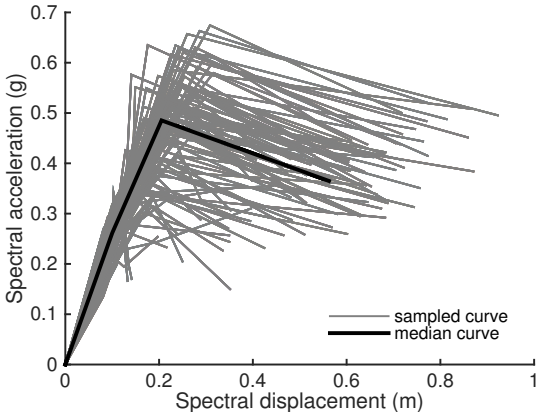

Figure 4.10 – *Capacity curves generated using the point dispersion approach.*

4.4 Conversion from MDOF to SDOF

Several structural analysis packages allow the user to perform a reliable pushover analysis on a nonlinear model of the MDoF structure. Often, these MDoF pushover cuves need to be converted to simplified SDoF models for use in nonlinear static analysis methods. This module allows the conversion of the MDoF results into 'equivalent SDoF' results, thus making them compatible with a wide range of non-linear static procedures. At present, two methods are provided in this module for obtaining equivalent SDoF capacity curves for an MDoF system.

4.4.1 Conversion based on one mode of vibration

This method allows the user to convert a pushover curve for a MDoF system into an equivalent SDoF capacity curve, considering the first mode of vibration only. The supplied pushover curve, which is in terms of base shear and roof displacement, is transformed into an equivalent SDoF capacity curve, which is in terms of spectral acceleration and spectral displacement. The properties of the equivalent SDoF model correspond to the properties of the first mode of vibration of the MDoF system. The roof displacement is converted to S_d by normalising by the participation factor of the first mode of vibration, whereas the base shear is normalised to give S_a using the the modal mass for the first mode. Details of this conversion method can be found in ATC-40 (1996) and FEMA-440 (2005).

The equivalent system acceleration $S_{a-capacity}$ and displacement $S_{d-capacity}$ are calculated as:

$$S_{a-capacity} = \frac{V_{b-pushover}}{M_1^* g} \tag{4.12}$$

$$S_{d-capacity} = \frac{\Delta_{roof}}{\Gamma_1 \phi_{1,roof}} \tag{4.13}$$

where Γ_1 and M_1^* are the modal participation factor and the effective modal mass of the first mode respectively, Δ_{roof} is the displacement of the roof, $\phi_{1,roof}$ is the roof displacement in the first mode, and $V_{b-pushover}$ is the total base shear from the pushover analysis. Γ_1 and M_1^* are obtained using the following equations:

$$\Gamma_1 = \frac{\sum_{i=1}^{N} m_i \phi_{i,1}}{\sum_{i=1}^{N} m_i \phi_{i,1}^2} \tag{4.14}$$

$$M_1^* = \sum_{i=1}^{N} m_i \phi_{i,1}^2 \tag{4.15}$$

where m_i is the mass of storey i, N is the number of storeys and $\phi_{i,1}$ is the displacement of storey i in the first mode. In the calculation of Γ_1 and M_1^*, we assume that the first mode

shape ϕ has been normalised to unit amplitude at the roof, i.e., $\phi_{1,roof} = 1$. Thus, we have the following:

$$S_{a-capacity} = \frac{V_{b-pushover}}{M_1^* g} \tag{4.16}$$

$$S_{d-capacity} = \frac{\Delta_{roof}}{\Gamma_1} \tag{4.17}$$

The user should thus provide the modal participation factor Γ_1 and M_1^* normalised with respect to the first mode eigenvector in the capacity curve input file (see Section 4.2.1).

4.4.2 Conversion using an adaptive approach

Adaptive pushover methods have the advantage of better accounting for stiffness degradation, influence of higher mode effects, and spectral amplifications because of ground motion frequency content. The method used for the determination of the 'equivalent SDoF adaptive capacity curve' in this module is the approach recommended in Casarotti and Pinho (2007). In this method, the equivalent SDoF capacity curve is calculated step by step based on the actual displacements, rather than a transformation of the capacity curve referred to the roof displacement. Instead of relying on an invariant elastic or inelastic modal shape, the equivalent system displacement and acceleration are calculated at each analysis step, using the actual displacements at that step.

The equivalent system modal participation factor $\Gamma_{sys,k}$ at step k is calculated as:

$$\Gamma_{sys,k} = \frac{\sum_i m_i \Delta_{i,k}}{\sum_i m_i \Delta_{i,k}^2} \tag{4.18}$$

The equivalent system displacement $\Delta_{sys,k}$ at step k is calculated as:

$$\Delta_{sys,k} = \frac{\sum_i m_i \Delta_{i,k}^2}{\sum_i m_i \Delta_{i,k}} \tag{4.19}$$

Note that $\Delta_{sys,k}$ is defined to be the inverse of the modal participation factor.

The equivalent system mass $M_{sys,k}$ at step k is defined as:

$$M_{sys,k} = \frac{\sum_i m_i \Delta_{i,k}}{\Delta_{sys,k}} \tag{4.20}$$

The equivalent system acceleration $S_{a-capacity}$ and displacement $S_{d-capacity}$ are calculated as:

$$S_{a-capacity} = \frac{V_{b-pushover}}{M_{sys} g} \tag{4.21}$$

$$S_{d-capacity} = \frac{1}{\Gamma_{sys}} \tag{4.22}$$

4.5 Direct nonlinear static procedures

The Ruiz-Garcia and Miranda (2007), Vamvatsikos and Cornell (2006) and Dolsek and Fajfar (2004) studies on the assessment of nonlinear structural response have been integrated into three nonlinear static procedures, which are based on the use of capacity curves, resulting from nonlinear static pushover analysis, to determine directly the median seismic intensity values $\hat{S}_a$ corresponding to the attainment of a certain damage state threshold (limit state) and the corresponding dispersion β_{S_a}. These parameters are used to represent a fragility curve as the probability of the limit state capacity C being exceeded by the demand D, both expressed in terms of intensity levels ($S_{a,ds}$ and S_a respectively), as shown in the following equation:

$$P_{LS}(S_a) = P(C < D|S_a) = \Phi(\frac{lnS_a - ln\hat{S}_a, ds}{\beta_{S_a}}) \tag{4.23}$$

The methodologies implemented allow to consider different shapes of the pushover curve, multilinear and bilinear, record-to-record dispersion and dispersion in the damage state thresholds in a systematic and harmonised way.

The intensity measure to be used is S_a and a mapping between any engineering demand parameter (EDP), assumed to describe the damage state thresholds, and the roof displacement should be available from the pushover analysis.

The methodologies were originally derived for single building fragility curves, however the fragility curves derived for single buildings can be combined in a unique fragility curve, which considers also the inter-building uncertainty, as described in the Section 4.5.1.

4.5.1 SPO2IDA (Vamvatsikos and Cornell 2006)

The tool SPO2IDA (Vamvatsikos and Cornell, 2005) is capable of converting static pushover curves into 16%, 50% and 84% IDA curves, as shown in Figure 4.11, using empirical relationships from a large database of incremental dynamic analysis results.

The SPO2IDA tool is applicable to any kind of multi-linear capacity curve. Making use of this tool, is is possible to estimate a single building fragility curve and fragility curves derived for single buildings can be combined in a unique fragility curve, which considers also the inter-building uncertainty.

Given an idealised capacity curve the SPO2IDA tool uses an implicit $R-\mu-T$ relation to correlate nonlinear displacement, expressed in terms of ductility μ to the corresponding median capacities in terms of the parameters R. R is the lateral strength ratio, defined as the ratio between the spectral acceleration S_a and the yielding capacity of the system S_{ay}. Each branch of the capacity curve, hardening, softening and residual plateau, is converted to a corresponding branch of the three IDA curves, using the $R-\mu-T$ relation, which is a function of the hardening stiffness, the softening stiffness and the residual force. These

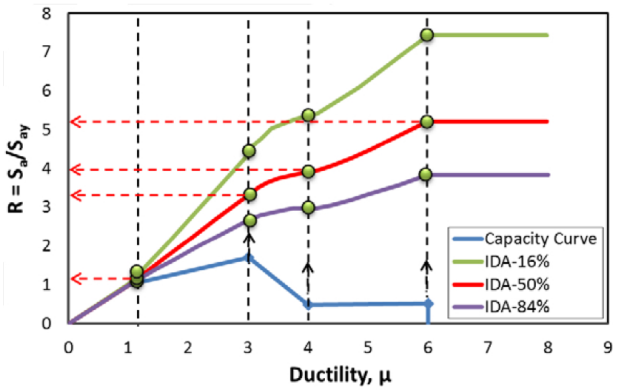

Figure 4.11 – *spo2ida tool: IDA curves derived from pushover curves*

parameters are derived from the idealised pushover capacity expressed in $\mu - R$ terms, as well as the ductility levels at the onset of each branch. If some of the branches of the pushover curve are missing because of the seismic behaviour of the system, SPO2IDA can equally work with bilinear, trilinear and quadrilinear idealisations.

The result of the SPO2IDA routine is thus a list of ductility levels and corresponding R values at 50%, 16% and 84% percentiles. For any inelastic displacement, and therefore any level of ductility μ the corresponding $R_{50\%}$, $R_{16\%}$, and $R_{84\%}$ values are found by interpolating the aforementioned IDA curves.

Median R and its dispersion at ductility levels corresponding to the damage thresholds ds can thus be determined, and converted into the median $S_{a,ds}$ and dispersion due to record-to-record variability $\beta_{S_{ad}}$ according to Equations 4.24 and 4.25.

$$\hat{S}_{a,ds} = R_{50\%}(\mu_{ds}) S_{ay} \tag{4.24}$$

$$\beta_{S_{ad}} = \beta_{R(\mu_{ds})} = \frac{\ln R(\mu_{ds})_{84\%} - \ln R(\mu_{ds})_{16\%}}{2} \tag{4.25}$$

If dispersion due to uncertainty in the limit state definition $\beta_{\theta c}$ is different from zero, a Monte Carlo sampling needs to be performed to combine it with the record-to-record dispersion. Different values of ductility limit state are sampled from the lognormal distribution with the median value of the ductility limit state, and dispersion the input $\beta_{\theta c}$. For each of these ductilities the corresponding median $R_{50\%}$ and $R_{16\%}$, $R_{84\%}$ are found and converted into $\hat{S}_{a,ds}$ and $\beta_{S_{ad}}$ according to equation 4.24 and 4.25. Monte Carlo random S_a

for each sampled ductility limit state are computed, and their median and the dispersion are estimated. These parameters constitute the median $\hat{S}_{a,ds}$ and the total dispersion β_{S_a} for the considered damage state. The procedure is repeated for each damage state.

4.5.1.1 Multiple-Building Fragility and Vulnerability functions

If multiple buildings have been input to derive fragility function for a class of buildings all $\hat{S}_{a,blg}$ and $\beta_{S_a,blg}$ are combined in a single lognormal curve. A minimum of 5 buildings should be considered to obtain reliable results for the class.

A new issue arises when multiple buildings are considered: the S_a at the fundamental period of each building should be converted to a common intensity measure, to be able to combine the different fragility functions. A common intensity measure is selected to be S_a at the period T_{av}, which is a weighted average of the individual building fundamental periods T_1. Then each individual fragility needs to be expressed in terms of the common $S_a(T_{av})$, using a spectrum. FEMA P-695 (ATC-63, 2007) far field set of 44 accelerograms (22 records for the two directions) was used to derive a mean spectrum, and the ratio between the S_a at different periods is used to scale the fragility functions. It can be noted that the actual values of the spectrum are not important, but just the spectral shape.

The median $\hat{S}_a$ is converted to the mean $\mu_{ln(S_a)}$ of the corresponding normal distribution ($\mu_{ln(S_a)} = ln(\hat{S}_a)$) and simply scaled to the common intensity measure as follows:

$$\mu_{ln(S_a),blg} = \mu_{ln(S_a),blg} S(T_{av})/S(T_{1,blg}) \tag{4.26}$$

$$\beta_{S_a,blg} = \beta_{S_a,blg} S(T_{av})/S(T_{1,blg}) \tag{4.27}$$

where $S(T_{av})/S(T_{1,blg}$ is defined as spectral ratio. Finally the parameters of the single lognormal curve for the class of buildings, mean and dispersion, can be computed as the weighted mean of the single means and the weighted SRSS of the inter-building and intra-building standard deviation, the standard deviation of the single means and the single dispersions respectively, as shown in the following equations:

$$\mu_{ln(S_a),tot} = \sum_{i=0}^{n.blg} w_{blg-i} \mu_{ln(S_a),blg-i} \tag{4.28}$$

$$\beta_{S_a,tot} = \sqrt{\sum_{i=0}^{n.blg} w_{blg-i}((\mu_{ln(S_a),blg-i} - \mu_{ln(S_a),tot})^2 + \beta^2_{S_a,blg-i})} \tag{4.29}$$

In order to use this methodology, it is necessary to load one or multiple capacity curves as described in Section 4.2.1. The pushover curve input type needs to be either Base Shear vs

Roof Displacement (Section 4.2.1.1), or Base Shear vs Floor Displacements (Section 4.2.1.2). It is also necessary to specify the type of shape the capacity curves want to be idealised with, using the parameter `idealised_type` (either `bilinear` or `quadrilinear`). If the user has already at disposal an idealised multilinear pushover curve for each building, the variable `Idealised` in the csv input file should be set to `TRUE`, and idealised curves should be provided according to what described in section 4.2.1. Then, it is necessary to specify a damage model using the parameter `damage_model`. The inter-storey drift based damage model described in 4.2.3.4 should be used for this method.

If dispersion due to uncertainty in the limit state definition is different from zero a Monte Carlo sampling needs to be performed to combine it with the record-to-record dispersion. The number of Monte Carlo samples should be defined in the variable `montecarlo_samples`. After importing the module `SPO2IDA_procedure`, it is possible to calculate the parameter of the fragility model, median and dispersion, using the following command:

```
fragility_model = SPO2IDA_procedure.calculate_fragility(capacity_curves,
... idealised_capacity, damage_model, montecarlo_samples, Sa_ratios,
... ida_plotflag)
```

where `Sa_ratios` is the spectral ratio variable needed to combine together fragility curves for many buildings, as described above, and `ida_plotflag` indicates whether ida plots should be displayed (`ida_plotflag` = 1) or not (`ida_plotflag` = 0).

4.5.2 Dolsek and Fajfar 2004

This procedure by (Dolsek and Fajfar, 2004) provides a simple relationship between inelastic displacement of a SDoF system and the corresponding median elastic spectral displacement value. The procedure presented herein is applicable to any kind of multi-linear capacity curve and it can be used to estimate single building fragility curves. Moreover the fragility curves derived for single buildings can be combined into a unique fragility curve, which considers also the inter-building uncertainty.

The relationship provided by (Dolsek and Fajfar, 2004) has been adapted for MDoF systems, relating the inelastic top displacement of a structure $\hat{d}_{roof}$ to the median elastic spectral acceleration value at its fundamental period of vibration $\hat{S}_a(T_1)$, as presented in the following equation:

$$\hat{S}_a(T_1) = \frac{4\pi^2}{\hat{C}_R T^2 \Gamma_1 \Phi_1} \hat{d}_{roof} \tag{4.30}$$

where $\Gamma_1 \Phi_1$ is the first mode participation factor estimated for the first-mode shape

normalised by the roof displacement. The value of C_R, the ratio between the inelastic and the elastic spectral displacement, is found from the following equation:

$$\hat{C}_R = \frac{\mu}{R(\mu)} \tag{4.31}$$

where μ and R are the median values of ductility level and the reduction factor for that level of ductility respectively. R is defined as the ratio between the spectral acceleration S_a and the yielding capacity of the system S_{ay}. According to the results of an extensive parametric study using three different sets of recorded and semi-artificial ground motions, (Dolsek and Fajfar, 2004) related the ductility demand μ and reduction factor R through the following formula:

$$\mu = \frac{1}{c}(R - R_0) + \mu_0 \tag{4.32}$$

In the proposed model μ is linearly dependent on R within two reduction factor intervals. The parameter c defines the slope of the R–μ relation, and it depends on the idealised pushover curve parameters (the initial period of the system T, the maximum to residual strength ratio r_u, the ductility at the onset of degradation μ_s) and the corner periods T_c and T_d. T_c and T_d are the corner periods between the constant acceleration and constant velocity part of the idealised elastic spectrum, and between the constant velocity and constant displacement part of the idealised elastic spectrum respectively. R_0 and μ_0 are the values of R and μ on the capacity curve corresponding to the onset of hardening or softening behaviour, according to the following relationship:

$$\mu_0 = 1 ... if\, R <= R(\mu_s); \mu_0 = \mu_s ... if\, R > R(\mu_s) \tag{4.33}$$

$$R_0 = 1 ... if\, R <= R(\mu_s); R_0 = R(\mu_s) ... if\, R > R(\mu_s) \tag{4.34}$$

Given the parameters of the multi-linear pushover curves (R_0, μ_0, r_u, μ_s) and T, the median R-μ curve can be constructed using the aforementioned relationship, as presented in the following Figure.

The relationship between the 16^{th} and 84^{th} fractiles of μ and R_{50} needs to be derived using the equations from (Ruiz-Garcia and Miranda, 2007) instead, given that (Dolsek and Fajfar, 2004) do not provide estimates of the dispersion of R. This is done by computing the value of record-to-record dispersion in terms of top displacement $\beta_{\theta d}$ for a number of R with Equation 4.41, and calculating the 16^{th} and 84^{th} fractiles of μ ($\mu_{16\%}$ and $\mu_{84\%}$), according to the Equations 4.35 and 4.36. The $\mu_{50\%} - R_{50\%}$, $\mu_{16\%} - R_{50\%}$ and $\mu_{84\%} - R_{50\%}$ curves can thus be drawn, as shown in Figure 4.13.

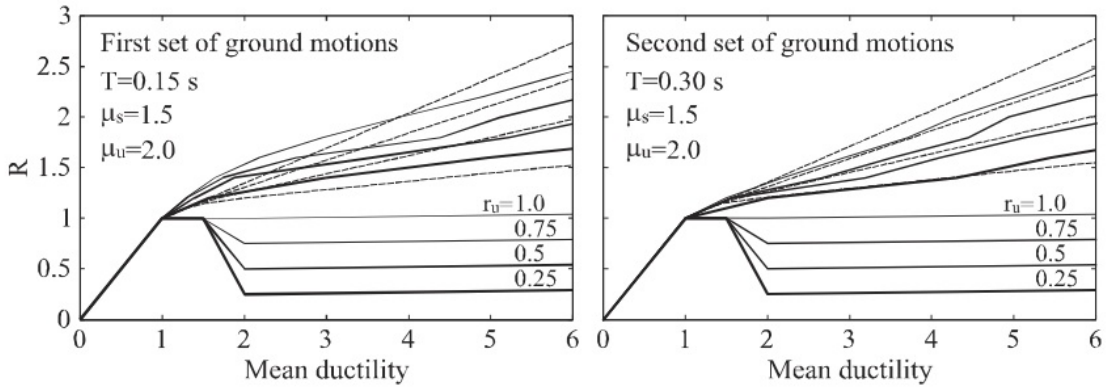

Figure 4.12 – *R-μ curves derived from Pushover curve.*

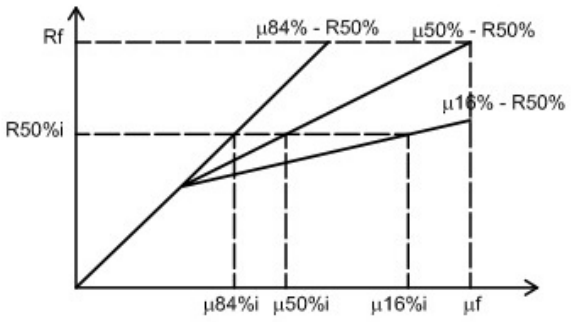

Figure 4.13 – $\mu_{50\%} - R_{50\%}$, $\mu_{16\%} - R_{50\%}$ *and* $\mu_{84\%} - R_{50\%}$ *curves*

$$\mu_{ds,16} = \hat{\mu}_{ds} e^{-\beta_{\theta d,ds}} \tag{4.35}$$

$$\mu_{ds,84} = \hat{\mu}_{ds} e^{\beta_{\theta d,ds}} \tag{4.36}$$

For any inelastic displacement, and therefore any level of ductility μ, the corresponding $R_{50\%}$, $R_{16\%}$, and $R_{84\%}$ values are found by interpolating the aforementioned curves. Median R and its dispersion at ductility levels corresponding to the damage thresholds ds can thus be determined, and converted into median $S_{a,ds}$ and its dispersion due to record-to-record variability $\beta_{S_{ad}}$ according to equations 4.24 and 4.25.

If dispersion in the damage state threshold is different from zero, different values of ductility limit state are sampled from the lognormal distribution with the median value of the ductility limit state, and dispersion of the input $\beta_{\theta c}$. For each of these ductilities the corresponding $R_{50\%}$, $R_{16\%}$, and $R_{84\%}$ values are found by interpolating the $\mu_{50\%} - R_{50\%}$, $\mu_{16\%} - R_5 0\%$ and $\mu_{84\%} - R_5 0\%$ curves, and converting into $\hat{S}_{a,ds}$ and $\beta_{S_{ad}}$ according to Equations 4.24 and 4.25. Monte Carlo random S_a for each of the sampled ductility limit states are computed using $\hat{S}_{a,ds}$ and $\beta_{S_{ad}}$, and their median and dispersion are estimated. These parameters constitute the median $\hat{S}_{a,ds}$ and the total dispersion β_{S_a} for the considered damage state. The procedure is repeated for each damage state.

If multiple buildings have been input to derive fragility function for a class of buildings, all $\hat{S}_{a,blg}$ and $\beta_{S_a,blg}$ are combined into a single lognormal curve as described in section 4.5.1.

In order to use this methodology, it is necessary to load one or multiple capacity curves as described in Section 4.2.1. The pushover curve input type needs to be either Base Shear vs Roof Displacement (Section 4.2.1.1), or Base Shear vs Floor Displacements (Section 4.2.1.2). The capacity curves are then idealised with a bilinear elasto-plastic shape. It is also necessary to specify the type of shape the capacity curves should be idealised with, using the parameter `idealised_type` (either `bilinear` or `quadrilinear`). If the user has already an idealised multilinear pushover curve for each building, the variable `Idealised` in the csv input file should be set to `TRUE`, and idealised curves should be provided according to section 4.2.1. Then, it is necessary to specify a damage model using the parameter `damage_model` (see Section 4.2.3). The inter-storey drift based damage model described in 4.2.3.4 should be used for this method.

If dispersion due to uncertainty in the limit state definition is different from zero, a Monte Carlo sampling needs to be performed to combine it with the record-to-record dispersion. The number of Monte Carlo samples should be defined in the variable `montecarlo_samples`.

After importing the module DF2004, it is possible to calculate the parameters of the fragility model, median and dispersion, using the following command:

```
fragility_model = DF2004.calculate_fragility(capacity_curves, ...
idealised_capacity, damage_model, montecarlo_samples, Sa_ratios, ...
corner_periods)
```

where `Sa_ratios` is a variable needed to combine together fragility curves for many buildings, as described in Section 4.5.1.

4.5.3 Ruiz Garcia and Miranda 2007

The research by (Ruiz-Garcia and Miranda, 2007) provides a simple relationship for SDoF systems between inelastic displacement and the corresponding median elastic spectral displacement value. The procedure presented herein is applicable to bilinear elasto-plastic capacity curve only and it can be used to estimate single building fragility curves. Moreover, the fragility curves derived for single buildings can be combined into a unique fragility curve, which considers also the inter-building uncertainty.

The relationship provided by (Ruiz-Garcia and Miranda, 2007) has been adapted for MDoF systems (Vamvatsikos, 2014), by relating the inelastic top displacement of a structure $\hat{d}_{roof}$ to the median elastic spectral displacement value at its fundamental period of vibration $\hat{S}_d(T)$, as presented in the following equation:

$$\hat{S}_d(T_1) = \frac{\hat{d}_{roof}}{\hat{C}_R \Gamma_1 \Phi_1} \tag{4.37}$$

where $\Gamma_1\Phi_1$ is the first mode participation factor estimated for the first-mode shape normalised by the roof displacement, and C_R is the inelastic displacement ratio (inelastic over elastic spectral displacement), computed by (Ruiz-Garcia and Miranda, 2007) for nonlinear SDoF systems, which is a function of the first-mode period of vibration and the relative lateral strength of the system R. Therefore the median spectral acceleration at the fundamental period of vibration $\hat{S}_a(T)$ turns out to be expressed as a function of top displacement according to the following equation:

$$\hat{S}_a(T) = \frac{4\pi^2}{\hat{C}_R T^2 \Gamma_1 \Phi_1} \hat{d_{roof}} \tag{4.38}$$

Estimates of $\hat{C}_R$ parameter are provided by (Ruiz-Garcia and Miranda, 2007), as result of nonlinear regression analysis of three different measures of central tendency computed from 240 ground motions:

$$\hat{C}_R = 1 + \frac{\hat{R} - 1}{79.12 T_1^{1.98}} \tag{4.39}$$

where $\hat{R}$ is given by the following equation:

$$\hat{R} = max(0.425(1 - c + \sqrt{c^2 + 2c(2\hat{\mu} - 1) + 1}), 1) \tag{4.40}$$

where $c = 79.12T^{1.98}$, and $\hat{\mu}$ is the median ductility level of interest.

Moreover (Ruiz-Garcia and Miranda, 2007) provide an estimate of the dispersion of C_R parameter due to record-to-record variability with Equation 4.41, that can be assumed equal to the dispersion of d_{roof}, since the two quantities are proportional.

$$\sigma_{\ln(C_R)} = \sigma_{\ln(d_{roof})} = \beta_{\theta d} = 1.975[\frac{1}{5.876} + \frac{1}{11.749(T + 0.1)}][1 - \exp(-0.739(R - 1))] \tag{4.41}$$

The median value of S_a corresponding to any level of ductility (d_{roof}/d_y) can be defined combining Equation from 4.38 to 4.40. The relationship between the 16^{th} and 84^{th} fractiles of μ and R can be drawn instead, computing $\beta_{\theta d}$ for a discretised number of R with eq. 4.41, and calculating the 16^{th} and 84^{th} fractiles of μ ($\mu_{16\%}$ and $\mu_{84\%}$), according to the Equations 4.35 and 4.36. $\mu_{16\%}$-$R_{50\%}$, $\mu_{50\%}$-$R_{50\%}$ obtained in such way are shown in Figure 4.14.

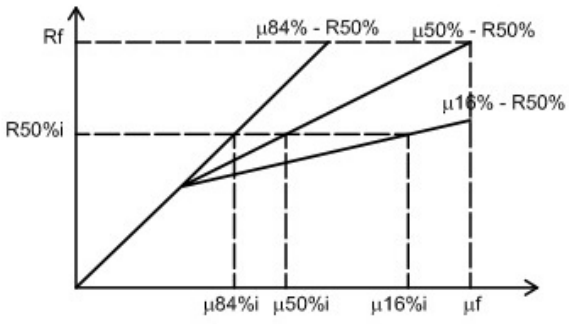

Figure 4.14 – *R-μ relationship.*

For any inelastic displacement, and therefore any level of ductility μ the corresponding $R_{50\%}$, $R_{16\%}$, and $R_{84\%}$ values are found interpolating the aforementioned curves. Median R and its dispersion at ductility levels corresponding to the damage thresholds ds can thus be determined, and converted into median $S_{a,ds}$ and its dispersion due to record-to-record variability $\beta_{S_{ad}}$, according to equations 4.24 and 4.25.

If dispersion in the damage state threshold is different from zero, different values of ductility limit state are sampled from the lognormal distribution with the median value of the ductility limit state, and dispersion $\beta_{\theta c}$. For each of these ductilities the corresponding $R_{50\%}$, $R_{16\%}$, and $R_{84\%}$ values are found interpolating the $\mu_{50\%} - R_{50\%}$, $\mu_{16\%} - R_5 0\%$ and $\mu_{84\%} - R_5 0\%$ curves, and converted into $\hat{S}_{a,ds}$ and $\beta_{S_{ad}}$ according to Equations 4.24 and 4.25. Monte Carlo random S_a for each of the sampled ductility limit states are computed using $\hat{S}_{a,ds}$ and $\beta_{S_{ad}}$, and their median and dispersion are estimated. These parameters constitute

the median $\hat{S}_{a,ds}$ and the total dispersion β_{S_a} for the considered damage state. The procedure is repeated for each damage state.

If multiple buildings have been input to derive fragility function for a class of buildings all $\hat{S}_{a,blg}$ and $\beta_{S_a,blg}$ are combined in a single lognormal curve as described in section 4.5.1.

In order to use this methodology, it is necessary to load one or multiple capacity curves as described in Section 4.2.1. The pushover curve input type needs to be either Base Shear vs Roof Displacement (Section 4.2.1.1), or Base Shear vs Floor Displacements (Section 4.2.1.2). The capacity curves are then idealised with a bilinear elasto-plastic shape. If the user has already at disposal an idealised multilinear pushover curve for each building, the variable `Idealised` in the csv input file should be set to `TRUE`, and idealised curves should be provided according to what described in section 4.2.1. Then, it is necessary to specify a damage model using the parameter `damage_model`. The inter-storey drift based damage model described in 4.2.3.4 should be used for this method.

If dispersion due to uncertainty in the limit state definition is different from zero a Monte Carlo sampling needs to be performed to combine it with the record-to-record dispersion. The number of Monte Carlo samples should be defined in the variable `montecarlo_samples`. After importing the module RGM2007, it is possible to calculate the parameter of the fragility model, median and dispersion, using the following command:

```
fragility_model = RGM2007.calculate_fragility(capacity_curves, ...
idealised_capacity, damage_model, montecarlo_samples, Sa_ratios)
```

where `Sa_ratios` is the spectral ratio variable, needed to combine together fragility curves for many buildings, as described in Section 4.5.1.

4.6 Record-based nonlinear static procedures

The nonlinear static procedures described in this section allow the calculation of the seismic response of a number of structures (in terms of maximum displacement of the equivalent single degree of freedom (SDoF) system), considering a set of ground motion records (see Section 4.2.2). The development of these methods involves numerical analysis of systems with particular structural and dynamic properties (e.g. periods of vibration, viscous damping, hysteretic behaviour, amongst others) and accelerograms selected for specific regions in the world (e.g. California, South Europe). For these reasons, their applicability to other types of structures and different ground motion records calls for due care. This section provides a brief description of each methodology, but users are advised to fully comprehend the chosen methodology by reading the original publications.

The main results of each of these methodologies is a probability damage matrix (i.e. fraction of assets per damage state for each ground motion record, represented by the

variable PDM), and the spectral displacement (i.e. expected maximum displacement of the equivalent SDoF system, represented by the variable Sds) per ground motion record. Using the probability damage matrix (PDM), it is possible to derive a fragility model (i.e. probability of exceedance of a number of damage states for a set of intensity measure levels - see Section 4.8.1), which can then be converted into a vulnerability function (i.e. distribution of loss ratio for a set of intensity measure levels - see Section 4.8.2), using a consequence model (see Section 4.2.4).

Table 4.14 comprises a probability damage matrix calculated considering 100 assets and 10 ground motion records. For the purposes of this example, an extra column has been added to this table in order to display the peak ground acceleration (PGA) of each accelerogram.

Table 4.14 – *Example of a probability damage matrix*

PGA	No damage	Slight damage	Moderate damage	Extensive damage	Collapse
0.015	1.00	0.00	0.00	0.00	0.00
0.045	0.85	0.12	0.03	0.00	0.00
0.057	0.72	0.20	0.08	0.00	0.00
0.090	0.31	0.35	0.33	0.01	0.00
0.126	0.12	0.34	0.53	0.01	0.00
0.122	0.07	0.18	0.73	0.02	0.00
0.435	0.00	0.00	0.53	0.32	0.15
0.720	0.00	0.00	0.26	0.45	0.29
0.822	0.00	0.00	0.16	0.48	0.36
0.995	0.00	0.00	0.02	0.48	0.50

4.6.1 Vidic, Fajfar and Fischinger 1994

This procedure aims to determine the displacements from an inelastic spectra for systems with a given ductility factor. The inelastic displacement spectra is determined by means of applying a ductility-based reduction factor (C), which depends on the natural period of the system, the given ductility factor, the hysteretic behaviour, the damping model, and the frequency content of the ground motion.

The procedure proposed by (Vidic et al., 1994) was validated by a comparison of the approximate spectra with the "exact" spectra obtained from non-linear dynamic time history analyses. Records from California and Montenegro were used as representative of "standard" ground motion, while the influence of input motion was analysed using other five groups of records (coming from different parts of the world) that represented different types of ground motions. The influence of the hysteretic models was taken into account by considering the bilinear model and the stiffness degrading Q-model. Finally, in order to analyse the effect

of damping, two models were considered: "mass-proportional" damping, which assumes a time-independent damping coefficient based on elastic properties, and "instantaneous stiffness-proportional" damping, which assumes a time-dependent damping coefficient based on tangent stiffness. For most cases, a damping ratio of 5% was assumed, although for some systems a value of 2% was adopted.

It is possible to derive approximate strength and displacement inelastic spectra from an elastic pseudo-acceleration spectrum using the proposed modified spectra. In the medium and long-period region, it was observed that the reduction factor is slightly dependent on the period T and is roughly equal to the prescribed ductility (μ). However, in the short-period region, the factor C strongly depends on both T and μ. The influence of hysteretic behaviour and damping can be observed for the whole range of periods. Based on this, a bilinear curve was proposed. Starting in $C = 1$, the value of C increases linearly along the short-period region up to a value approximately equal to the ductility factor. In the medium- and long-period range, the C-factor remains constant. This is mathematically expressed by the following relationships:

$$C_\mu = \begin{cases} c_1 (\mu - 1)^{c_R} \frac{T}{T_0} + 1, & T \leq T_0 \\ c_1 (\mu - 1)^{c_R} + 1 & T > T_0 \end{cases} \tag{4.42}$$

where:

$$T_0 = c_2 \mu^{c_T} T_c \tag{4.43}$$

And T_c stands for the characteristic spectral period and c_1, c_2, c_R, c_T are constants dependant on the hysteretic behaviour and damping model, as defined in Table 4.15.

Table 4.15 – *Parameretes for the estimation of the reduction factor C proposed by (Vidic et al., 1994)*

Hysteresis model	Damping model	c_1	c_2	c_R	c_T
Q	Mass	1.00	0.65	1.00	0.30
Q	Stiffness	0.75	0.65	1.00	0.30
Bilinear	Mass	1.35	0.75	0.95	0.20
Bilinear	Stiffness	1.10	0.75	0.95	0.20

In order to use this methodology, it is necessary to load one or multiple capacity curves as described in Section 4.2.1, as well as a set of ground motion records as explained in Section 4.2.2. Then, it is necessary to specify a damage model using the parameter `damage_model` (see Section 4.2.3), and a damping ratio using the parameter `damping`. It is also necessary to specify the type of hysteresis (`Q` or `bilinear`) and damping (`mass` or

`stiffness`) models as defined in Table 4.15, using the parameters `hysteresis_model` and `damping_model`, respectively. After importing the module `vidic_etal_1994`, it is possible to calculate the distribution of structures across the set of damage states for each ground motion record using the following command:

```
PDM, Sds = vidic_etal_1994.calculate_fragility(capacity_curves,gmrs,...
damage_model,damping)
```

Where `PDM` (i.e. probability damage matrix) represents a matrix with the number of structures in each damage state per ground motion record, and `Sds` (i.e. spectral displacements) represents a matrix with the maximum displacement (of the equivalent SDoF) of each structure per ground motion record. The variable PDM can then be used to calculate the mean fragility model as described in Section 4.8.1.

4.6.2 Lin and Miranda 2008

This methodology estimates the maximum inelastic displacement of an existing structure based on the maximum elastic displacement response of its equivalent linear system without the need for iterations, based on the strength ratio R (instead of the most commonly used ductility ratio).

In order to evaluate an existing structure, a pushover analysis should be conducted in order to obtain the capacity curve. This curve should be bilinearised in order to obtain the yield strength, fy, the post-yield stiffness ratio, α, and the strength ratio, R. With these parameters, along with the initial period of the system, it is possible to estimate the optimal period shift (i.e. the ratio between the period of the equivalent linear system and the initial period) and the equivalent viscous damping, ξeq, of the equivalent linear system, using the following relationships derived by (Lin and Miranda, 2008).

$$\frac{T_{eq}}{T_0} = 1 + \frac{m_1}{T_0^{m_2}}\left(R^{1.8} - 1\right) \tag{4.44}$$

$$\xi_{eq} = \xi_0 + \frac{n_1}{T_0^{n_2}}(R - 1) \tag{4.45}$$

Where the coefficients m_1, m_2, n_1, and n_2 depend on the post-yield stiffness ratio, as shown in the following Table 4.16.

Using ξeq and the damping modification factor, B (as defined in Table 15.6-1 of NEHRP-2003), it is possible to construct the reduced displacement spectrum, Sd(T, ξeq) from which the maximum displacement demand (i.e. the displacement corresponding to the equivalent system period) can be obtained, using the following equation:

In order to use this methodology, it is necessary to load one or multiple capacity curves as described in Section 4.2.1, as well as a set of ground motion records as explained in Section

Table 4.16 – *Paramereters for the estimation of the reduction factor C proposed by (Lin and Miranda, 2008)*

α	m_1	m_2	n_1	n_2
0%	0.026	0.87	0.016	0.84
5%	0.026	0.65	0.027	0.55
10%	0.027	0.51	0.031	0.39
20%	0.027	0.36	0.030	0.24

4.2.2. Then, it is necessary to specify a damage model using the parameter `damage_model` (see Section 4.2.3). After importing the module `lin_miranda_2008`, it is possible to calculate the distribution of structures across the set of damage state for each ground motion record using the following command:

```
PDM, Sds = lin_miranda_2008.calculate_fragility(capacity_curves,gmrs,...
damage_model,damping)
```

Where `PDM` (i.e. probability damage matrix) represents a matrix with the number of structures in each damage state per ground motion record, and `Sds` (i.e. spectral displacements) represents a matrix with the maximum displacement (of the equivalent SDoF) of each structure per ground motion record. The variable PDM can then be used to calculate the mean fragility model as described in Section 4.8.1.

4.6.3 Miranda (2000) for firm soils

This study by Miranda, 2000 aims to quantify the influence of soil conditions, earthquake magnitude, and epicentral distance on the inelastic displacement ratios, C_μ. For two systems with the same mass and period of vibration that have been subjected to the same earthquake ground motion. C_μ can be defined as the ratio of the maximum lateral inelastic displacement demand of one to the maximum lateral elastic displacement demand on the other, as shown in the following equation:

$$C_\mu = \frac{\Delta_{inelastic}}{\Delta_{elastic}} \tag{4.46}$$

In this study, 264 earthquake acceleration time histories recorded in California (USA) for 12 different events were used. In order to investigate the effect of the soil conditions, the records were classified into three groups: the first one consisted of ground motions recorded on stations located on rock (i.e. average shear-wave velocities >760 m/s). The second group included the records registered on stations on very dense soil or soft rock (i.e. average

shear-wave velocities between 360 m/s and 760 m/s). Finally, the third group consisted of ground motion records from stations located on stiff soil (i.e. average shear-wave velocities between 180 m/s and 360 m/s).
It was observed that for periods longer than about 1.0 s, the mean inelastic displacement ratios are approximately equal to 1, meaning that, on average, the maximum inelastic displacements are equal to the maximum inelastic displacements. On the other hand, for periods smaller than 1.0 s, the mean inelastic displacement ratios are larger than 1 and strongly depend on the period of vibration and on the level of inelastic deformation. The results of the investigation yielded that for the sites under consideration (i.e. average shear-wave velocities higher than 180 m/s) neither the soil conditions, nor the earthquake magnitude, nor the distance to rupture cause significant differences on the value of C_μ. However, if directivity effects are taken into consideration, the inelastic displacement ratios for periods between 0.1 s and 1.3 s can be larger than those estimated for systems not affected by directivity.

Based on the results of the mean inelastic displacement ratios, nonlinear regression analyses were conducted to estimate the following simplified expression for the inelastic displacement ratio of a system:

$$C_\mu = \left[1 + \left(\frac{1}{\mu} - 1\right) exp\left(-12T\mu^{-0.8}\right)\right]^{-1} \tag{4.47}$$

Where μ = displacement ductility ratio and T = period of vibration.
In order to use this methodology, it is necessary to load one or multiple capacity curves and a set of ground motion records, as explained in Section 4.2.1 and 4.2.2, respectively. Then, it is necessary to specify a damage model using the parameter `damage_model` (see Section 4.2.3), and a damping ratio using the parameter `damping`. After importing the module `miranda_2000_firm_soils`, it is possible to calculate the distribution of structures across the set of damage states for each ground motion record using the following command:

```
PDM, Sds = miranda_2000_firm_soils.calculate_fragility(capacity_curves,...
gmrs,damage_model,damping)
```

Where `PDM` (i.e. probability damage matrix) represents a matrix with the number of structures in each damage state per ground motion record, and `Sds` (i.e. spectral displacements) represents a matrix with the maximum displacement (of the equivalent SDoF) of each structure per ground motion record. The variable PDM can then be used to calculate the mean fragility model as described in Section 4.8.1.

4.6.4 N2 (EC8, CEN 2005)

This simplified nonlinear procedure was first proposed in Fajfar and Gaspersic (1996), and it is capable of estimating the seismic response of structures using capacity curves (for the

equivalent SDoF) and response spectra. It is somehow similar to the well-known Capacity Spectrum Method (see Section 4.6.5), but it does not require an iterative process and instead of elastic over-damped spectra, it uses inelastic response spectra. This method is part of recommendations of the Eurocode 8 (CEN, 2005) for the seismic design and assessment of structures, and the capacity curves are usually simplified by a elasto-perfectly plastic relationship.

To estimate the target displacement (δ_t) within this methodology, it is necessary to assess whether the SDoF structure is in the short-period or medium / long-period ranges. To do so, it is necessary to compare the fundamental period of vibration of the structure with the corner period of the ground motion record. If the structure is in the latter category, it is assumed that the target displacement is equal to the elastic spectral displacement for the fundamental period of the idealized SDoF. If on the other hand it is located in the short-period range, a procedure is carried out to evaluate whether the capacity of the SDoF at the yielding point (taken from the bilinear curve) is lower than the spectral acceleration response for the same period. If this is verified, then the structure is assumed to have an elastic response and once again, the target displacement will be equal to the elastic spectral displacement for the fundamental period. In case the capacity is lower than the response for the yielding point, the structure is assumed to have an inelastic response and the following formula is employed to determine the target displacement:

$$\delta_t = \frac{Sd(T_{el})}{q_u}\left(1+(q_u-1)\frac{T_c}{T_{el}}\right)Sd(T_{el}) \tag{4.48}$$

Where $Sd(T_{el})$ stands for the spectral displacement for the fundamental period of the idealized SDoF (T_{el}), T_c stands for the corner period and q_u represents the ratio between the spectral acceleration for T_{el} and the acceleration at the yielding point.

It is important to understand that this methodology has been developed originally to be combined with a design or code-based response spectrum, and not with a spectrum derived from real ground motion records. For this reason, its employment in the derivation of fragility functions calls for due care. For instance, the estimation of T_c (which is a fundamental parameter within this methodology) when considering accelerograms is not a trivial task, and various proposals can be found in the literature. For the sake of simplicity, a decision was made to adopt the formula recommended by the ASCE7-10, 2010 which defines:

$$T_c = \frac{Sa(T=1.0s)}{Sa(T=0.2s)} \tag{4.49}$$

Despite these caveats, it is worth mentioning that a recent study (Silva et al., 2014b) compared its performance in the derivation of vulnerability functions against nonlinear time

history analyses, and concluded that reasonable results can still be obtained.

In order to use this methodology, it is necessary to load one or multiple capacity curves and a set of ground motion records, as explained in Section 4.2.1 and 4.2.2, respectively. Then, it is necessary to specify a damage model using the parameter `damage_model` (see Section 4.2.3), and a damping ratio using the parameter `damping`. After importing the module `N2Method`, it is possible to calculate the distribution of structures across the set of damage states for each ground motion record using the following command:

```
PDM, Sds = N2Method.calculate_fragility(capacity_curves,gmrs,...
damage_model,damping)
```

Where `PDM` (i.e. probability damage matrix) represents a matrix with the number of structures in each damage state per ground motion record, and `Sds` (i.e. spectral displacements) represents a matrix with the maximum displacement (of the equivalent SDoF) of each structure per ground motion record. The variable PDM can then be used to calculate the mean fragility model as described in Section 4.8.1.

4.6.5 Capacity Spectrum Method (FEMA, 2005)

The capacity spectrum method (CSM) was initially proposed by (Freeman et al., 1975), and it represents a simplified methodology for many purposes such as the evaluation of a large inventory of buildings, structural assessment of new or existing buildings or to identify the correlation between damage states and levels of ground motion. ATC-40 (1996) proposes three different procedures (A, B and C) for the application of the Capacity Spectrum Method. However, procedure B adopts some simplifications that might not always be valid and procedure C has a very strong graphical component, making it difficult for systematic applications. Hence, procedure A, which is characterized by its intuitiveness and simplicity, has been implemented in the RMTK.

This procedure iteratively compares the capacity and the demand of a structure, using a capacity curve (for the equivalent SDoF) and a damped response spectrum, respectively. The ground motion spectrum is computed for a level of equivalent viscous damping that is estimated as a function of the displacement at which the response spectrum crosses the capacity curve, in order to take into account the inelastic behaviour of the structure. Iterations are needed until there is a match between the equivalent viscous damping of the structure and the damping applied to the spectrum. The final intersection of these two curves approximates the target displacement response of the structure. This result is presented in Figure 4.15 for a "weak" and a "strong" ground motion record.

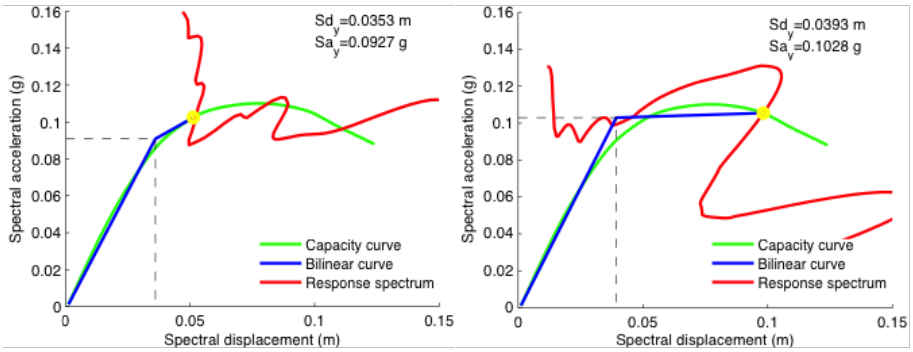

Figure 4.15 – *Assessment of the target displacement for "weak" (left) and "strong" (strong) ground motion record.*

The initial proposal of this method was heavily criticized due to its tendency to underestimate the deformation of the structures, which was mostly related with the model employed to calculate the equivalent viscous damping (e.g. Fajfar, 1999; Chopra and Goel, 2000). Thus, in FEMA-440, 2005, some modifications were proposed regarding the calculation of this component. Furthermore, several other models relating an equivalent viscous damping ratio (ξ_{eq}) with a ductility level (μ) have been proposed in the last decades, and implemented in the RMTK. The following list describes these models, and specifies the code that must be defined in the variable `damping_model` in order to follow the associated model in the vulnerability calculations.

- FEMA-440, 2005: This model assumes different expressions to calculate the equivalent viscous damping ratio depending on the ductility level, hysteretic model and post-elastic stiffness. However, for the sake of simplicity, approximate equations have been proposed to calculate ξ_{eq} with any capacity curve, that only depends on the level of ductility, as described below:

  For $1.0 < \mu < 4.0$:

  $$\xi_{eq} = \xi_0 + 0.049\,(\mu - 1)^2 - 0.01\,(\mu - 1)^3 \tag{4.50}$$

  For $4.0 \leq \mu \leq 6.5$:

  $$\xi_{eq} = \xi_0 + 0.14 + 0.0032\,(\mu - 1) \tag{4.51}$$

  For $\mu > 6.5$:

  $$\xi_{eq} = \xi_0 + 0.19\left[\frac{0.64\,(\mu - 1) - 1}{[0.64\,(\mu - 1)]^2}\right]\left(\frac{T_{eff}}{T_0}\right)^2 \tag{4.52}$$

  Where ξ_0 stands the initial elastic viscous damping ratio, and T_{eff} represents the effective period, which for ductility above 6.5 can be calculated using the following

expression:

$$T_{eff} = \left\{ 0.89 \left[\sqrt{\frac{(\mu - 1)}{1 + 0.05(\mu - 2)}} \right] + 1 \right\} T_0 \tag{4.53}$$

In order to use this model, the variable `damping_model` must be set to `FEMA_2005`.

- Kowalsky, 1994: This model establishes a relationship between the equivalent viscous damping ratio and a ductility level and a post-yield stiffness ratio α, as defined by the following equation:

$$\xi_{eq} = \xi_0 + \frac{1}{\pi} \left[1 - \frac{(1 - \alpha)}{\sqrt{\mu}} - \alpha \sqrt{\mu} \right] \tag{4.54}$$

In order to use this model, the variable `damping_model` must be set to `Kowalsky_1994`.

- (Iwan, 1980): This model was developed using a limited number of ground motion records and a single hysteretic model, leading to the following equation:

$$\xi_{eq} = \xi_0 + 0.0587 \, (\mu - 1)^{0} \, .371 \tag{4.55}$$

In order to use this model, the variable `damping_model` must be set to `Iwan_1980`.

- Gulkan and Sozen, 1974:This model was derived considering the Takeda hysteretic for elasto-plastic systems calibrated with experimental shaking-table results of a number of reinforced concrete frames. The equivalent viscous damping ratio is calculated using the following ductility-dependent formula:

$$\xi_{eq} = \xi_0 + 0.2 \left(1 - \frac{1}{\sqrt{\mu}} \right) \tag{4.56}$$

In order to use this model, the variable `damping_model` must be set to `Gulkan_Sozen_1974`.

- Priestley et al., 2007: These Authors proposed different models depending on the structure type. Currently, three models proposed by this study have been implemented, as described below:

  For reinforced concrete frame structures:

$$\xi_{eq} = 0.05 + 0.565 \left(\frac{\mu - 1}{\pi \mu} \right) \tag{4.57}$$

To use this model set the variable `damping_model` to `Priesley_et_al2007_frames`.

For reinforced concrete walls structures:

$$\xi_{eq} = 0.05 + 0.444\left(\frac{\mu - 1}{\pi\mu}\right) \tag{4.58}$$

To use this model set the variable `damping_model` to `Priesley_et_al2007_walls`.

For steel structures:

$$\xi_{eq} = 0.05 + 0.577\left(\frac{\mu - 1}{\pi\mu}\right) \tag{4.59}$$

To use this model set the variable `damping_model` to `Priesley_et_al2007_steel`.

- Calvi, 1999: This Author proposed a relationship between the equivalent viscous damping ratio and ductility following the expression below:

$$\xi_{eq} = \xi_0 + a\left(1 - \frac{1}{\mu^b}\right) \tag{4.60}$$

Where `a` and `b` are constants that vary between 20 and 30, and 0.5 and 1, respectively, depending on the hysteretic properties of the structure. Thus, this model can be employed for various structure types, by adjusting these two constants. Given the fact that most of the current damping models have been derived or calibrated for reinforced concrete structures, a decision was made to adjust these parameters for the assessment of masonry structures. The `a` and `b` constants have been set to 25 and 0.5, respectively, as proposed by Borzi et al., 2008a. Nonetheless, due to the open and transparent architecture of the RMTK, any user can modify these parameters.

In order to use this model, the variable `damping_model` must be set to `Calvi_1999`.

The performance point (or target displacement) calculated within this methodology is equivalent to what would be obtained by subjecting the equivalent single degree of freedom oscilator to a nonlinear time history analysis. Then, estimated target displacement can be used to allocate the structure in a damage state, based on a pre-established set of displacement thresholds. This process can be repeated several times considering other ground motion records, as well as structures (i.e. building class).

In order to use this methodology, it is necessary to load one or multiple capacity curves and a set of ground motion records, as explained in Section 4.2.1 and 4.2.2, respectively.

Then, it is necessary to specify a damage model using the parameter `damage_model` (see Section 4.2.3), and a damping ratio using the parameter `damping`. After importing the module `capacitySpectrumMethod`, it is possible to calculate the distribution of structures across the set of damage states for each ground motion record using the following command:

```
PDM, Sds = capacitySpectrumMethod.calculate_fragility(capacity_curves,...
gmrs,damage_model,damping)
```

Where `PDM` (i.e. probability damage matrix) represents a matrix with the number of structures in each damage state per ground motion record, and `Sds` (i.e. spectral displacements) represents a matrix with the maximum displacement (of the equivalent SDoF) of each structure per ground motion record. The variable PDM can then be used to calculate the mean fragility model as described in Section 4.8.1.

4.6.6 DBELA (Silva et al. 2013)

The Displacement-based Earthquake Loss Assessment (DBELA) methodology builds upon the urban assessment methodology proposed by Calvi, 1999, in which the principles of structural mechanics and seismic response of buildings are used to estimate the seismic vulnerability of classes of buildings. The current implementation of the RMTK is only compatible with bare or infilled frame reinforced concrete structures.

In this method, the displacement capacity and demand for a number of limit states needs to be calculated. Each limit state marks the threshold between the levels of damage that a building might withstand, usually described by a reduction in strength or by exceedance of certain displacement / drift levels. Once these parameters are obtained, the displacement capacity of the first limit state is compared with the respective demand. If the demand exceeds the capacity, the next limit states need to be checked successively, until the demand no longer exceeds the capacity and the building damage state can be defined. If the demand also exceeds the capacity of the last limit state, the building is assumed to have collapsed. This procedure is schematically depicted in Figure 4.16, in which the capacities for three limit states are represented by Δ_i and the associated demand by Sd_i. In this example, the demand exceeds the capacity in the first and second limit state but not in the third limit state, thus allocating the building to the third damage state.

The calculation of the displacement capacity at each limit state is explained in the Model Generation Section (4.3), as this methodology can be employed to generate large sets of capacity curves (Sa versus Sd), which can be combined with other methodologies besides DBELA to derive fragility functions. Instead, this section is focused on describing how the seismic demand is handled in this methodology.

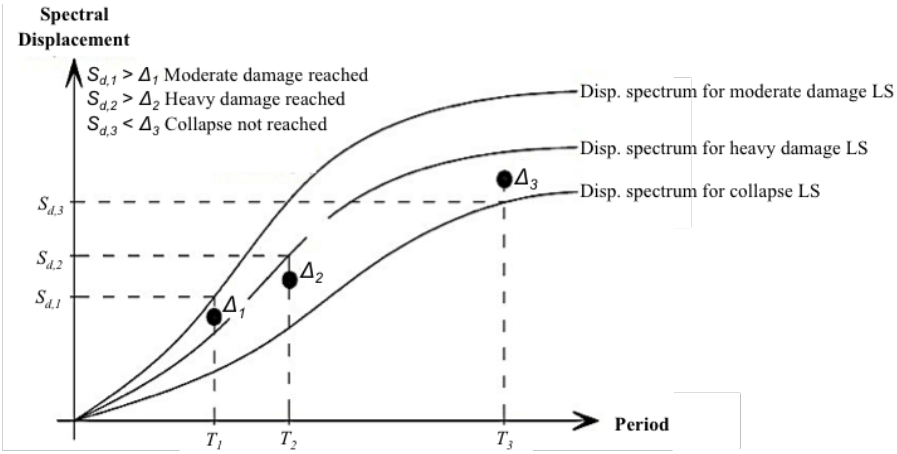

Figure 4.16 – *Comparison between limit state capacity and the associated demand (adapted from Bal et al., 2010).*

The demand is represented by a displacement spectrum which can be described as the expected displacement induced by an earthquake on a single-degree-of-freedom (SDoF) oscillator with a given period of vibration and viscous damping. This demand is initially calculated for a 5% viscous damping, and later modified for each limit state using a correction factor (η), representative of the equivalent viscous damping and ductility at the associated damage state. In the Eurocode 8 (CEN, 2005), the following equation is proposed for the calculation of the correction factor:

$$\eta_{LS_i} = \sqrt{\frac{10}{5 + \xi_{eq_i}}} \tag{4.61}$$

Where ξ_{eq_i} stands for the equivalent viscous damping at the limit state i. Although in theory there is a multitude of damping models in the literature that could be used to calculate this equivalent viscous damping (see Section 4.6.5 for a description of the damping models implemented within the Capacity Spectrum Method), this method has been tested following the proposals by Priestley et al., 2007 for reinforced concrete frames (e.g. Bal et al., 2010 Silva et al., 2013). This model uses the following equation:

$$\xi_{eq} = 0.05 + 0.565\left(\frac{\mu - 1}{\pi\mu}\right) \tag{4.62}$$

Where μ_i stands for the ductility at the limit state i (assumed as the ratio between Δ_i and Δ_y). More accurate approaches have recently been proposed to estimate the correction factors (η), considering additional parameters, such as the magnitude or source-to-site distance (Rezaeian et al., 2012).

With regards to the calculation of the yielding period (T_y) for bare frame structures, Crowley and Pinho, 2004 and Crowley et al., 2008 proposed a relationship between the

period and the total height (H_T) of $0.10H_T$ and $0.07H_T$ for structures without and with lateral load design, respectively. For infilled frames, a relation equal to $0.06H_T$ has been recommended by Crowley and Pinho, 2006 for structures without lateral load design. The elongated period of vibration for any of the limit states (T_{LS_i}) can be computed using the following formula:

$$T_{LS_i} = T_y \sqrt{\frac{\mu_i}{1 + \alpha\mu_i - \alpha}} \tag{4.63}$$

where α stands for the post-yield stiffness ratio. In cases where this ratio can be assumed as zero, the relation between T_{LS_i} and T_y will depend purely on the limit state ductility as follows:

$$T_{LS_i} = T_y \sqrt{\mu_i} \tag{4.64}$$

In order to use this methodology, it is necessary to first assess the capacity displacement of one or multiple assets, following the DBELA approach explained in Section 4.3.1 (Model Generator). Moreover, a set of ground motion records and a damage model should be provided, as explained in Sections 4.2.2 and 4.2.3, respectively. The type of structures that are being evaluated should be specified using the parameter `structure_type`. Currently this module of the RMTK accepts the options `bare frame` and `infilled frame`. After importing the module `DBELA`, it is possible to calculate the distribution of structures across the set of damage states for each ground motion record using the following command:

```
PDM, Sds = DBELA.calculate_fragility(capacity_curves,gmrs,...
damage_model,structure_type)
```

Where `PDM` (i.e. probability damage matrix) represents a matrix with the number of structures in each damage state per ground motion record, and `Sds` (i.e. spectral displacements) represents a matrix with the maximum displacement (of the equivalent SDoF) of each structure per ground motion record. The variable PDM can then be used to calculate the mean fragility model as described in Section 4.8.1.

4.7 Nonlinear time-history analysis in Single Degree of Freedom (SDOF) Oscillators

This methodology performs a series of non-linear time history analyses (NLTHA) over one or multiple Single Degree of Freedom (SDoF) systems. In order to determine the structural capacity of the Multi Degree of Freedom (MDoF) system(s) under analysis, it is necessary to identify the relationship between the base shear and roof displacement (i.e. pushover curve). This curve should be then converted to the capacity curve of an equivalent SDoF oscillator

(see Section 4.4). For low and midrise buildings it is typically assumed that the fundamental mode of vibration corresponds to the predominant response of the structure. Under this hypothesis, the SDoF oscillator represents the first mode of response of the structure. This is usually valid for buildings with fundamental periods of vibration up to approximately 1.0 s. Otherwise, higher modes should be taken into account. Along with the capacity curve, it is necessary to specify either the mass or the fundamental period of vibration of the SDoF system.

In this methodology the demand is represented by a set of ground motion records. The response of each structure is given by the solution of the equation of motion for an inelastic SDoF under earthquake excitation:

$$m\ddot{u}(t) + c\dot{u}(t) + ku(t) = p(t) \tag{4.65}$$

Where u, $\dot{u}$ and $\ddot{u}$ stand for the displacement, velocity and acceleration, respectively, over time (t), and p represents an external excitation. The nonlinear time history analysis are performed using the open-source software for structural analysis OpenSees (McKenna et al., 2000). It is important to understand the GEM Foundation does not have authorization to distribute this tool, and therefore in order to use this methodology, users are expected to download OpenSEES (http://opensees.berkeley.edu/), and allocate it in the appropriate folder (`vulnerability/derivation_fragility/NLTHA_on_SDOF`).

The user has to provide one (or multiple) capacity curve, idealized by five relevant Sd-Sa points, a value of damping ratio, the period of the structure and the degree of degradation in the cyclic rule. Of these five relevant points, four are needed to represent the hysteretic behaviour of the OpenSees material (see Figure 4.17); the fifth is the origin that should be also specified in the inputs for the generation of the SDOF system in the RMTK implementation. In summary, the first point corresponds to the origin (0, 0); the second couple of Sd-Sa values (ePd1, ePf1) corresponds to the yielding point of the structure, i.e. the point beyond which the structure no longer displays an elastic behaviour; the following two points, (ePd2, ePf2) and (ePd3, ePf3), are defined as any two intermediate points between the yield point and the ultimate point which can be used to represent particular structural properties, such as reduction of stiffness due to collapse of infill panels or softening behaviour due to P-delta effects; the last point (ePd4, ePf4) corresponds to the point of maximum displacement of the structure.

Pinching4 OpenSees material is employed to model the hysteretic behaviour of the SDOF oscillator. Figure 6.2 illustrates the input parameters for the Pinching4 one-dimensional model: a response envelope (black lines), an unload-reload path (grey lines), and three damage rules that control the evolution of these paths, as described by (Lowes et al., 2003).

It can be noted that the capacity curve can describe only the envelope response of the hysteretic behaviour. In order to keep the definition of the model as simple as possible, some

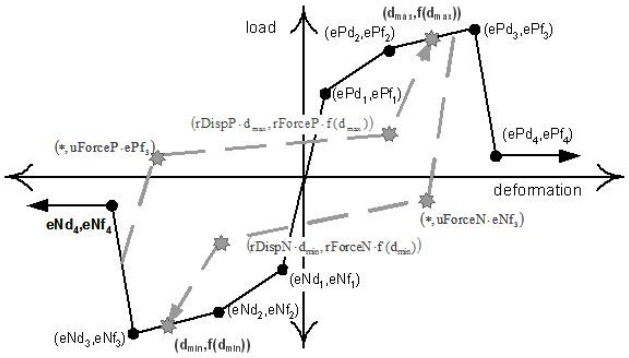

Figure 4.17 – *Representation of the capacity curve required to represent the structural capacity of the SDoF system.*

assumptions have been formulated on the unload-reload path and the three damage rules.

The first assumption regards the level of pinching of the cyclic behaviour. rDispP, rForceP and uForceP (see Figure 4.17) are the parameters that control the unload-reload paths of Pinching4 material; they are assigned default values of 0.5, 0.25 and 0.05, respectively to represent a medium level of pinching. Since the cyclic behaviour is considered symmetrical the parameters rDispN, rForceN and uForceN are set equal to rDispP, rForceP and uForceP, respectively. The second assumption regards the degrading behaviour of the model. The user has the option of introducing energy-based degradation during the cyclic hystory assigning non-zero values to the parameters gK2, gD2 and gF2, that control unloading, reloading stiffness and strength degradation, respectively. If degradation is selected gK2, gD2 and gF2 are assigned default values of 0.1, 0.1 and 0.4 respectively. More details about the degrading models and the meaning of the degrading parameters can be found in (Lowes et al., 2003) and in the OpenSees wiki (http://opensees.berkeley.edu/wiki/index.php/Pinching4_Material).

If the User agrees with these assumptions he should set the variable `sdof_hysteresis` to `"Default"` in the ipython notebook

```
sdof_hysteresis = "Default"
```

If the User wants instead to define each variable of the pinching4 material himself, he should assign the variable `sdof_hysteresis` the path to the file where these parameters are specified.

```
sdof_hysteresis = "../../../../../rmtk_data/pinching_parameters.csv"
```

The pinching parameters should be defined in a csv file (tabular format) as shown in Table

4.17. The variables of Table 4.17 are those characterising pinching4 material as explained in the OpenSees wiki (http://opensees.berkeley.edu/wiki/index.php/Pinching4_Material).

Table 4.17 – *Example of file containing Pinching4 material parameters*

rDisp	0.4				
fForce	0.6				
uForce	0				
gK1	0	0.6	0	0	1
gD1	0	0	0	0	0
gF1	0	0.07	0	0.07	0.9
gE	10				
dmgType	energy				

In order to use this methodology, it is necessary to load one or multiple capacity curves and a set of ground motion records, as explained in Sections 4.2.1 and 4.2.2, respectively. Then, it is necessary to specify a damage model using the parameter `damage_model`. Currently spectral displacement, capacity curve-based and inter-storey drift-based damage criterion can be used in the damage model (see Section 4.2.3). When the capacity curve-based criterion is selected, it may be necessary to identify the yielding spectral displacement and acceleration. The User can either input the yielding point in the input file, or he can use the algorithm described in Section 4.2.1 for the idealisation of the capacity curves, to identify the spectral displacement and acceleration at yielding on the capacity curves. When inter-storey drift-based damage criterion is selected it is necessary to define a file containing the relationship between maximum inter-storey drift and spectral displacement (see Section **??**). If this file is not defined a linear relationship between inter-storey drift and roof displacement is assumed and the spectral displacement is obtained dividing the roof displacement by the first modal participation factor Γ.

The damping ratio must be defined using the parameter `damping`, and if structural degradation should be considered in the analysis, it is necessary to set the parameter `degradation` to `True`. After importing the module `NLTHA_on_SDOF`, it is possible to calculate the distribution of structures across the set of damage states for each ground motion record using the following command:

```
PDM, Sds = NLTHA_on_SDOF.calculate_fragility(capacity_curves,gmrs,...
damage_model,damping)
```

Where `PDM` (i.e. probability damage matrix) represents a matrix with the number of structures in each damage state per ground motion record, and `Sds` (i.e. spectral displace-

ments) represents a matrix with the maximum displacement (of the equivalent SDoF) of each structure per ground motion record. The variable PDM can then be used to calculate the mean fragility model as described in Section 4.8.1.

4.8 Derivation of fragility and vulnerability functions

This section explains how a probability damage matrix (`PDM`) can be used to derive a fragility function (i.e. probability of exceedance of a number of damage states for a set of intensity measure levels), and then converted into a vulnerability function (i.e. distribution of loss ratio for a set of intensity measure levels), using a consequence model (see Section 4.2.4). Both of these models can be exported in the OpenQuake-engine format (`nrml`), or following a `csv` format. Additional intructions about the necessary input models and associated formats can be found on the OpenQuake-engine User Guide (GEM, 2015).

4.8.1 Derivation of fragility functions

These fragility functions can be used directly by the Scenario Damage or the Classiclal PSHA-based Damage calculators of the OpenQuake-engine (Silva et al., 2014a; Pagani et al., 2014).

In order to use the results of the nonlinear static procedures (see Section 4.6) for the derivation of a fragility model, there are a number of attributes that need to be specified. Each function must be related with a building class, which must be defined using the parameter `taxonomy`. Additional information about the GEM building taxonomy can be found in Brzev et al., 2013, and a tool to compile the GEM taxonomy can be found on the OpenQuake-platform (https://taxtweb.openquake.org/).

For what concerns the definition of the seismic input, it is necessary to establish the intensity measure type that should be considered using the variable `IMT`. Currently, the Risk Modeller's Toolkit supports `PGA`, `PGV` and `Sa`. If the latter intensity measure type is chosen, it is also necessary to establish the period of vibration (in seconds) using the variable `T`, and the elastic damping using the variable `damping`. Finally, the range of applicability of the fragility model should be defined using the `minIML` and `maxIML`.

The results in the probability damage matrix (`PDM`) are used to fit a lognormal cumulative function, with a logarithmic mean (μ) and logarithmic standard deviation (σ). These two parameters are calculated using one of the two currently implemented statistical methods: least squares or the maximum likelihood. The former approach estimates a solution (μ, σ) that minimizes the sum of the squares of the errors (i.e. difference between the prediction of the lognormal function and the data). The latter method leads to a solution that maximizes

the likelihood function. A comprehensive description of the strengths and limitations of these methodologies can be found in Lallemant et al., 2015. The method that should be followed must be specified by setting the parameter `regression_method` to `least squares` or to `maximum likelihood`.

The calculation of the fragility model also requires the set of ground motion records used in the analytical analyses (`gmrs`), and the damage model utilized to allocated each structure into a damage state (`damage_model`). A description of these two components have been provided in Section 4.2.2 and 4.2.3, respectively.

The function that calculates fragility models is contained in the module `utils`. An example of this process is depicted below.

```
IMT = 'Sa'
T = 0.3
regression_method = 'least squares'
taxonomy = 'RC'
minIML = 0.01
maxIML = 1
fragility_model = utils.calculate_mean_fragility(gmrs,PDM,T,damping,...
IMT,damage_model,regression_method)
```

Once the parameters (μ, σ) of the fragility model have been calculated, it is possible to save these results using the function `save_mean_fragility`. This feature can export the fragility model using the OpenQuake-engine format (`nrml`), or following a `csv` format. This indication should be defined using the variable `output_type`. It is also possible to create a plot of the resulting model, using the function `plot_fragility_model`. In order to use these functions, it is necessary to import the module `utils`. This process is demonstrated below.

```
output_type = 'nrml'
utils.save_mean_fragility(taxonomy,fragility_model,minIML,maxIML,...
output_type)
utils.plot_fragility_model(fragility_model,minIML,maxIML)
```

A detailed description of the `nrml` format can be found on the OpenQuake-engine manual (GEM, 2015). For what concerns the structure of the `csv` format, an example is provided in Table 4.18. The first row contains the building `taxonomy`, the intensity measure type (`IMT`), the minimum and maximum intensity measure levels (`minIML`, `maxIML`). The second row comprises the titles of the information stored in each column: damage states; logarithmic

mean; logarithmic standard deviation; mean; standard deviation; median and coefficient of variation. The remaining columns contain the results for each damage state.

Table 4.18 – *Example of a fragility model stored following a csv format.*

RC	Sa(0.3)	0.01	1.0			
Damage state	log mean	log stddev	mean	stddev	median	cov
Slight	-2.67	0.28	0.07	0.02	0.07	0.29
Moderate	-2.37	0.30	0.10	0.03	0.09	0.31
Extensive	-0.26	0.86	1.12	1.16	0.77	1.04
Collapse	0.42	0.96	2.42	2.99	1.52	1.24

Finally, a folder containing a set of fragility functions for buildings of different typologies derived using the RMTK and saved using the CSV format can be used to create a fragility model for use in OpenQuake risk analyses. In order to use the function `save_fragility_set_nrml`, it is necessary to import the module `utils`. The path to the folder containing the individual CSV fragility files, and the name of the destination XML file are the required inputs for this function. Usage of this function is shown below:

```
utils.save_fragility_set_nrml(folder, destination_file)
```

4.8.2 Derivation of vulnerability functions

These vulnerability functions can be used directly by the Scenario Risk, Classical PSHA-based Risk and Probabilistic Event-based Risk calculators of the OpenQuake-engine (Silva et al., 2014a; Pagani et al., 2014).

A vulnerability model can be derived directly from loss data (either analytically generated or based on past seismic events), or by combining a set of fragility functions with a consequence model (see Section 4.2.4). In this process, the fractions of buildings in each damage state are multiplied by the associated damage ratio (from the consequence model), in order to obtain a distribution of loss ratio for each intensity measure type. Currently only the latter approach is implemented in the Risk Modeller's Toolkit, though the former method will be included in a future release.

The location of the consequence model must be defined using the parameter `cons_model_file`, and loaded into the Risk Modeller's Toolkit using the function `read_consequence_model`. The intensity measure levels for which the distribution of loss ratio will be calculated must be defined using the variable `imls`.

The Risk Modeller's Toolkit allows the propagation of the uncertainty in the consequence model to the vulnerability function. Thus, instead of just providing a single loss ratio per intensity measure type, it is possible to define a probabilistic model (following a `lognormal` or `beta` functions) or a non-parametric model (i.e. probability mass function - `PMF`). This model must be defined using the variable `distribution_type`.

The derivation of the vulnerability function also requires the previously computed `fragility_model`. The function that calculates this result is contained in the module `utils`. An example of this process is depicted below.

```
cons_model_file = '../../../../../../rmtk_data/cons_model.csv'
cons_model = utils.read_consequence_model(cons_model_file)
imls = [0.1,0.2,0.3,0.4,0.5,0.6,0.8,0.9,1.0]
distribution_type = 'PMF'
vul_model = utils.convert_fragility_vulnerability(fragility_model,...
cons_model,imls,type_distribution)
```

The resulting vulnerability function can be saved using the function `save_vulnerability`. This feature can export the vulnerability function using the OpenQuake-engine format (`nrml`), or following a `csv` format. Similarly to what was described for the fragility models, this indication should be provided using the variable `output_type`. It is also possible to plot vulnerability functions, using the function `plot_vulnerability_model`. In order to use these functions, it is necessary to import the module `utils`. This process is demonstrated below.

```
output_type = 'nrml'
utils.save_vulnerability(taxonomy,vulnerability_model,output_type)
utils.plot_vulnerability_model(vulnerability_model)
```

A detailed description of the `nrml` format for vulnerability functions can be found on the OpenQuake-engine manual (GEM, 2015). For what concerns the structure of the `csv` file, this format varies depending on how the uncertainty is being defined: parametric (lognormal or beta) or non-parametric (probability mass function). For the former case, an example is provided in Table 4.19. The first row contains the building `taxonomy`, the intensity measure type (`IMT`), and the type of probabilistic model used to represent the uncertainty. The second row comprises the list of the intensity measure levels, and the means and associated coefficients of variation are provided in the third and fourth row, respectively.

For what concerns the `csv` format for vulnerability functions using the non-parametric approach, an example can be found in Table 4.20. The first two rows are similar to the

Table 4.19 – *Example of a vulnerability model with a parametric uncertainty modelling.*

RC	Sa(0.3)	lognormal							
imls	0.10	0.20	0.30	0.40	0.50	0.60	0.80	0.90	1.00
mean	0.00	0.02	0.05	0.08	0.12	0.17	0.25	0.29	0.33
cov	0.36	0.23	0.18	0.13	0.07	0.04	0.02	0.02	0.00

Table 4.20 – *Example of a vulnerability model with a non-parametric uncertainty modelling.*

RC	Sa(0.3)	PMF							
imls	0.10	0.20	0.30	0.40	0.50	0.60	0.80	0.90	1.00
loss ratio	probabilities								
0.00	0.80	0.15	0.00	0.00	0.00	0.00	0.00	0.00	0.00
0.11	0.20	0.60	0.30	0.00	0.00	0.00	0.00	0.00	0.00
0.22	0.00	0.25	0.60	0.40	0.00	0.00	0.00	0.00	0.00
0.33	0.00	0.00	0.10	0.50	0.20	0.00	0.00	0.00	0.00
0.44	0.00	0.00	0.00	0.10	0.70	0.10	0.00	0.00	0.00
0.56	0.00	0.00	0.00	0.00	0.10	0.50	0.10	0.00	0.00
0.67	0.00	0.00	0.00	0.00	0.00	0.35	0.30	0.00	0.00
0.78	0.00	0.00	0.00	0.00	0.00	0.05	0.50	0.10	0.00
0.89	0.00	0.00	0.00	0.00	0.00	0.00	0.10	0.70	0.00
1.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.20	1.00

previous case, and the remaining columns contain the probability of having a given loss ratio, conditional on an intensity measure level.

Finally, a folder containing a set of vulnerability functions for buildings of different typologies derived using the RMTK and saved using the CSV format can be used to create a vulnerability model for use in OpenQuake risk analyses. In order to use the function `save_vulnerability_set_nrml`, it is necessary to import the module `utils`. The path to the folder containing the individual CSV vulnerability files, and the name of the destination XML file are the required inputs for this function. Usage of this function is shown below:

```
utils.save_vulnerability_set_nrml(folder, destination_file)
```

Appendices

A. The 10 Minute Guide to Python

The RMTK is intended to be used by scientists and engineers without the necessity of having an existing knowledge of Python. It is hoped that the examples contained in this manual should provide enough context to allow the user to understand how to use the tools for their own needs. In spite of this, however, an understanding of the fundamentals of the Python programming language can greatly enhance the user experience and permit the user to join together the tools in a workflow that best matches their needs.

The aim of this appendix is therefore to introduce some fundamentals of the Python programming language in order to help understand how, and why, the RMTK can be used in a specific manner. If the reader wishes to develop their knowledge of the Python programming language beyond the examples shown here, there is a considerable body of literature on the topic from both a scientific and developer perspective. The bulk of this Python guide has been adapted from the Hazard Modeller's Toolkit - User Guide (Weatherill, 2014).

A.1 Basic Data Types

Fundamental to the use of the RMTK is an understanding of the basic data types Python recognises:

A.1.1 Scalar Parameters

- **float** A floating point (decimal) number. If the user wishes to enter in a floating point value then a decimal point must be included, even if the number is rounded to an integer.

```
1 >> a = 3.5
2 >> print a, type(a)
3 3.5 <type 'float'>
```

- **integer** An integer number. If the decimal point is omitted for a floating point number the number will be considered an integer

```
>> b = 3
>> print b, type(b)
3 <type 'int'>
```

The functions `float()` and `int()` can convert an integer to a float and vice-versa. Note that taking `int()` of a fraction will round the fraction down to the nearest integer

```
>> float(b)
3
>> int(a)
3
```

- **string** A text string (technically a "list" of text characters). The string is indicated by the quotation marks "something" or 'something else'

```
>> c = "apples"
>> print c, type(c)
apples <type 'str'>
```

- **bool** For logical operations python can recognise a variable with a boolean data type (`True` / `False`).

```
>> d = True
>> if d:
       print "y"
   else:
       print "n"
y
>> d = False
>> if d:
       print "y"
   else:
       print "n"
n
```

Care should be taken in Python as the value 0 and 0.0 are both recognised as False if applied to a logical operation. Similarly, booleans can be used in arithmetic where True and False take the values 1 and 0 respectively

```
>> d = 1.0
>> if d:
       print "y"
   else:
       print "n"
y
>> d = 0.0
>> if d:
       print "y"
   else:
```

```
11          print "n"
12  n
```

A.1.1.1 Scalar Arithmetic

Scalars support basic mathematical operations (# indicates a comment):

```
1  >> a = 3.0
2  >> b = 4.0
3  >> a + b # Addition
4  7.0
5  >> a * b # Multiplication
6  12.0
7  >> a - b # Subtraction
8  -1.0
9  >> a / b # Division
10 0.75
11 >> a ** b  # Exponentiation
12 81.0
13 # But integer behaviour can be different!
14 >> a = 3; b = 4
15 >> a / b
16 0
17 >> b / a
18 1
```

A.1.2 Iterables

Python can also define variables as lists, tuples and sets. These data types can form the basis for iterable operations. It should be noted that unlike other languages, such as Matlab or Fortran, Python iterable locations are zero-ordered (i.e. the first location in a list has an index value of 0, rather than 1).

- **List** A simple list of objects, which have the same or different data types. Data in lists can be re-assigned or replaced

```
1  >> a_list = [3.0, 4.0, 5.0]
2  >> print a_list
3  [3.0, 4.0, 5.0]
4  >> another_list = [3.0, "apples", False]
5  >> print another_list
6  [3.0, 'apples', False]
7  >> a_list[2] = -1.0
8  a_list = [3.0, 4.0, -1.0]
```

- **Tuples** Collections of objects that can be iterated upon. As with lists, they can support mixed data types. However, objects in a tuple cannot be re-assigned or replaced.

```
1  >> a_tuple = (3.0, "apples", False)
2  >> print a_tuple
3  (3.0, 'apples', False)
```

```
# Try re-assigning a value in a tuple
>> a_tuple[2] = -1.0
TypeError                 Traceback (most recent call last)
<ipython-input-43-644687cfd23c> in <module>()
----> 1 a_tuple[2] = -1.0

TypeError: 'tuple' object does not support item assignment
```

- **Range** A range is a convenient function to generate arithmetic progressions. They are called with a `start`, a `stop` and (optionally) a `step` (which defaults to 1 if not specified)

```
>> a = range(0, 5)
>> print a
[0, 1, 2, 3, 4]  # Note that the stop number is not
                 # included in the set!
>> b = range(0, 6, 2)
>> print b
[0, 2, 4]
```

- **Sets** A set is a special case of an iterable in which the elements are unordered, but contains more enhanced mathematical set operations (such as intersection, union, difference, etc.)

```
>> from sets import Set
>> x = Set([3.0, 4.0, 5.0, 8.0])
>> y = Set([4.0, 7.0])
>> x.union(y)
Set([3.0, 4.0, 5.0, 7.0, 8.0])
>> x.intersection(y)
Set([4.0])
>> x.difference(y)
Set([8.0, 3.0, 5.0]) # Notice the results are not ordered!
```

A.1.2.1 Indexing

For some iterables (including lists, sets and strings) Python allows for subsets of the iterable to be selected and returned as a new iterable. The selection of elements within the set is done according to the `index` of the set.

```
>> x = range(0, 10)  # Create an iterable
>> print x
[0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
>> print x[0] # Select the first element in the set
0             # recall that iterables are zero-ordered!
>> print x[-1] # Select the last element in the set
9
>> y = x[:] # Select all the elements in the set
>> print y
[0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
>> y = x[:4]  # Select the first four element of the set
```

```
>> print y
[0, 1, 2, 3]
>> y = x[-3:] # Select the last three elements of the set
>> print y
[7, 8, 9]
>> y = x[4:7] # Select the 4th, 5th and 6th elements
>> print y
[4, 5, 6]
```

A.1.3 Dictionaries

Python is capable of storing multiple data types associated with a map of variable names inside a single object. This is called a "Dictionary", and works in a similar manner to a "data structure" in languages such as Matlab. Dictionaries are used frequently in the RMTK as ways of structuring inputs to functions that share a common behaviour but may take different numbers and types of parameters on input.

```
>> earthquake = {"Name": "Parkfield",
                 "Year": 2004,
                 "Magnitude": 6.1,
                 "Recording Agencies" = ["USGS", "ISC"]}
# To call or view a particular element in a dictionary
>> print earthquake["Name"], earthquake["Magnitude"]
Parkfield 6.1
```

A.1.4 Loops and Logicals

Python's syntax for undertaking logical operations and iterable operations is relatively straightforward.

A.1.4.1 Logical

A simple logical branching structure can be defined as follows:

```
>> a = 3.5
>> if a <= 1.0:
       b = a + 2.0
   elif a > 2.0:
       b = a - 1.0
   else:
       b = a ** 2.0
>> print b
2.5
```

Boolean operations can are simply rendered as `and`, `or` and `not`.

```
>> a = 3.5
>> if (a <= 1.0) or (a > 3.0):
       b = a - 1.0
   else:
```

```
5         b = a ** 2.0
6 >> print b
7 2.5
```

A.1.4.2 Looping

There are several ways to apply looping in Python. For simple mathematical operations, the simplest way is to make use of the **range** function:

```
1 >> for i in range(0, 5):
2         print i, i ** 2
3 0   0
4 1   1
5 2   4
6 3   9
7 4   16
```

The same could be achieved using the `while` function:

```
1 >> i = 0
2 >> while i < 5:
3         print i, i ** 2
4         i += 1
5 0   0
6 1   1
7 2   4
8 3   9
9 4   16
```

A `for` loop can be applied to any iterable:

```
1 >> fruit_data = ["apples", "oranges", "bananas", "lemons",
2                  "cherries"]
3 >> i = 0
4 >> for fruit in fruit_data:
5         print i, fruit
6         i += 1
7 0   apples
8 1   oranges
9 2   bananas
10 3   lemons
11 4   cherries
```

The same results can be generated, arguably more cleanly, by making use of the `enumerate` function:

```
1 >> fruit_data = ["apples", "oranges", "bananas", "lemons",
2                  "cherries"]
3 >> for i, fruit in enumerate(fruit_data):
4         print i, fruit
5 0   apples
6 1   oranges
```

```
2  bananas
3  lemons
4  cherries
```

As with many other programming languages, Python contains the statements `break` to break out of a loop, and `continue` to pass to the next iteration.

```
>> i = 0
>> while i < 10:
        if i == 3:
            i += 1
            continue
        elif i == 5:
            break
        else:
            print i, i ** 2
        i += 1
0  0
1  1
2  4
4  16
```

A.2 Functions

Python easily supports the definition of functions. A simple example is shown below. *Pay careful attention to indentation and syntax!*

```
>> def a_simple_multiplier(a, b):
        """
        Documentation string - tells the reader the function
        will multiply two numbers, and return the result and
        the square of the result
        """
        c = a * b
        return c, c ** 2.0

>> x = a_simple_multiplier(3.0, 4.0)
>> print x
(12.0, 144.0)
```

In the above example the function returns two outputs. If only one output is assigned then that output will take the form of a tuple, where the elements correspond to each of the two outputs. To assign directly, simply do the following:

```
>> x, y = a_simple_multiplier(3.0, 4.0)
>> print x
12.0
>> print y
144.0
```

A.3 Classes and Inheritance

Python is one of many languages that is fully object-oriented, and the use (and terminology) of objects is prevalent throughout the RMTK and this manual. A full treatise on the topic of object oriented programming in Python is beyond the scope of this manual and the reader is referred to one of the many textbooks on Python for more examples

A.3.1 Simple Classes

A class is an object that can hold both attributes and methods. For example, imagine we wish to convert an earthquake magnitude from one scale to another; however, if the earthquake occurred after a user-defined year we wish to use a different formula. This could be done by a method, but we can also use a class:

```
>> class MagnitudeConverter(object):
       """
       Class to convert magnitudes from one scale to another
       """
       def __init__(self, converter_year):
           """
           """
           self.converter_year = converter_year

       def convert(self, magnitude, year):
           """
           Converts the magnitude from one scale to another
           """
           if year < self.converter_year:
               converted_magnitude = -0.3 + 1.2 * magnitude
           else:
               converted_magnitude = 0.1 + 0.94 * magnitude
           return converted_magnitude

>> converter1 = MagnitudeConverter(1990)
>> mag_1 = converter1.convert(5.0, 1987)
>> print mag_1
5.7
>> mag_2 = converter1.convert(5.0, 1994)
>> print mag_2
4.8
# Now change the conversion year
>> converter2 = MagnitudeConverter(1995)
>> mag_1 = converter2.convert(5.0, 1987)
>> print mag_1
5.7
>> mag_2 = converter2.convert(5.0, 1994)
>> print mag_2
5.7
```

In this example the class holds both the attribute `converter_year` and the method to convert the magnitude. The class is created (or "instantiated") with only the information regarding the cut-off year to use the different conversion formulae. Then the class has a method to convert a specific magnitude depending on its year.

A.4 Numpy/Scipy

Python has two powerful libraries for undertaking mathematical and scientific calculation, which are essential for the vast majority of scientific applications of Python: Numpy (for multi-dimensional array calculations) and Scipy (an extensive library of applications for maths, science and engineering). Both libraries are critical to both OpenQuake and the RMTK. Each package is so extensive that a comprehensive description requires a book in itself. Fortunately there is abundant documentation via the online help for Numpy www.numpy.org and Scipy www.scipy.org, so we do not need to go into detail here.

The particular facet we focus upon is the way in which Numpy operates with respect to vector arithmatic. Users familiar with Matlab will recognise many similarities in the way the Numpy package undertakes array-based calculations. Likewise, as with Matlab, code that is well vectorised is signficantly faster and more efficient than the pure Python equivalent.

The following shows how to undertake basic array arithmetic operations using the Numpy library

```
>> import numpy as np
# Create two vectors of data, of equal length
>> x = np.array([3.0, 6.0, 12.0, 20.0])
>> y = np.array([1.0, 2.0, 3.0, 4.0])
# Basic arithmetic
>> x + y   # Addition (element-wise)
np.array([4.0, 8.0, 15.0, 24.0])
>> x + 2   # Addition of scalar
np.array([5.0, 8.0, 14.0, 22.0])
>> x * y   # Multiplication (element-wise)
np.array([3.0, 12.0, 36.0, 80.0])
>> x * 3.0   # Multiplication by scalar
np.array([9.0, 18.0, 36.0, 60.0])
>> x - y   # Subtraction (element-wise)
np.array([2.0, 4.0, 9.0, 16.0])
>> x - 1.0   # Subtraction of scalar
np.array([2.0, 5.0, 11.0, 19.0])
>> x / y   # Division (element-wise)
np.array([3.0, 3.0, 4.0, 5.0])
>> x / 2.0   # Division over scalar
np.array([1.5, 3.0, 6.0, 10.0])
>> x ** y    # Exponentiation (element-wise)
np.array([3.0, 36.0, 1728.0, 160000.0])
>> x ** 2.0   # Exponentiation (by scalar)
np.array([9.0, 36.0, 144.0, 400.0])
```

Numpy contains a vast set of mathematical functions that can be operated on a vector (e.g.):

```
>> x = np.array([3.0, 6.0, 12.0, 20.0])
>> np.exp(x)
np.array([2.00855369e+01, 4.03428793e+02, 1.62754791e+05,
          4.85165195e+08])
# Trigonometry
>> theta = np.array([0., np.pi / 2.0, np.pi, 1.5 * np.pi])
>> np.sin(theta)
np.array([0.0000, 1.0000, 0.0000, -1.0000])
>> np.cos(theta)
np.array([1.0000, 0.0000, -1.0000, 0.0000])
```

Some of the most powerful functions of Numpy, however, come from its logical indexing:

```
>> x = np.array([3.0, 5.0, 12.0, 21.0, 43.0])
>> idx = x >= 10.0   # Perform a logical operation
>> print idx
np.array([False, False, True, True, True])
>> x[idx]   # Return an array consisting of elements
            # for which the logical operation returned True
np.array([12.0, 21.0, 43.0])
```

Create, index and slice n-dimensional arrays:

```
>> x = np.array([[3.0,  5.0, 12.0, 21.0, 43.0],
                 [2.0,  1.0,  4.0, 12.0, 30.0],
                 [1.0, -4.0, -2.1,  0.0, 92.0]])
>> np.shape(x)
(3, 5)
>> x[:, 0]
np.array([3.0, 2.0, 1.0])
>> x[1, :]
np.array([2.0, 1.0, 4.0, 12.0, 30.0])
>> x[:, [1, 4]]
np.array([[ 5.0, 43.0],
          [ 1.0, 30.0],
          [-4.0, 92.0]])
```

The reader is referred to the official Python documentation for the full set of functions! The official Python tutorial is a good resource to learn more about Python programming. A Byte of Python is also a well-written guide to Python and a great reference for beginners.

Bibliography

Books

Paulay, T. and M. J. N. Priestley (1992). *Seismic Design of Reinforced Concrete and Masonry Buildings*. John Wiley and Sons, Inc., New York, USA (cited on page 45).

Articles

Borzi, B., H. Crowley, and R. Pinho (2008a). "Simplified Pushover-Based Earthquake Loss Assessment (SP-BELA) Method for Masonry Buildings". In: *International Journal of Architectural Heritage* 2.4, pages 353–376 (cited on pages 47, 74).

Borzi, B., R. Pinho, and H. Crowley (2008b). "Simplified pushover-based vulnerability analysis for large-scale assessment of RC buildings". In: *Engineering Structures* 30.3, pages 804–820 (cited on pages 37, 38, 43, 47, 48).

Calvi, G. M. (1999). "A Displacement-Based Approach for Vulnerability Evaluation of Classes of Buildings". In: *Journal of Earthquake Engineering* 3.3, pages 411–438 (cited on pages 48, 74, 75).

Casarotti, C. and R. Pinho (2007). "An adaptive capacity spectrum method for assessment of bridges subjected to earthquake action". In: *Bulletin of Earthquake Engineering* 5.3, pages 377–390. ISSN: 1570761X (cited on page 54).

Casotto, C., V Silva, H Crowley, R. Nascimbene, and R. Pinho (2015). "Seismic fragility of Italian RC precast industrial structures". In: *Engineering Structures* (cited on page 39).

Chopra, A. and R. Goel (2000). "Evaluation of NSP to estimate seismic deformation: SDF systems". In: *Journal of Structural Engineering* 126.4, pages 482–490 (cited on page 72).

Cosenza, E., G. Manfredi, M. Polese, and G. Verderame (2005). "A multi-level approach to the capacity assessment of existing RC buildings". In: *Journal of Earthquake Engineering* 9.1, pages 1–22 (cited on pages 47, 48).

Crowley, H and R Pinho (2004). "Period-height relationship for existing European reinforced concrete buildings". In: *Journal of Earthquake Engineering* 8, pages 93–119 (cited on pages 45, 49, 76).

Crowley, H, B Borzi, R Pinho, M. Colombi, and M. Onida (2008). "Comparison of two mechanics-based methods for simplified structural analysis in vulnerability assessment". In: *Advances in Civil Engineering* 19 (cited on page 76).

Crowley, H., R. Pinho, and J. Bommer (2004). "A Probabilistic Displacement-based Vulnerability Assessment Procedure for Earthquake Loss Estimation". In: *Bulletin of Earthquake Engineering* 2, pages 173–219 (cited on pages 37, 38, 43, 44, 48).

Dolsek, M. and P. Fajfar (2004). “Inelastic spectra for infilled reinforced concrete frames”. In: *Earthquake Engineering and Structural Dynamics* 33.25, pages 1395–1416 (cited on pages 16, 58, 59).

Erberik, M. A. (2008). “Fragility-based assessment of typical mid-rise and low-rise RC buildings in Turkey”. In: *Engineering Structures* 30 (cited on pages 37, 39).

Fajfar, P. (1999). “Capacity spectrum method based on inelastic demand spectra.” In: *Earthquake Engineering and Structural Dynamics* 28.9, pages 979–993 (cited on page 72).

Fajfar, P. and P. Gaspersic (1996). “The N2 method for the seismic damage analysis of RC buildings”. In: *Earthquake Engineering & Structural Dynamics* 25.1, pages 31–46 (cited on page 69).

Glaister, S. and R. Pinho (2003). “Development of a Simplified Deformation-Based Method for Seismic Vulnerability Assessment”. In: *Journal of Earthquake Engineering* 7, pages 107–140 (cited on page 44).

Gulkan, P. and M. A. Sozen (1974). “Inelastic Responses of Reinforced Concrete Structure to Earthquake Motions”. In: *ACI Journal Proceedings* 71.12, pages 604–610 (cited on page 73).

Iwan, W (1980). “Estimating inelastic response spectra from elastic spectra”. In: *Earthquake Engineering and Structural Dynamics* (cited on page 73).

Kappos, A., G. Panagopoulos, C. Panagiotopoulos, and G. Penelis (2006). “A hybrid method for the vulnerability assessment of R/C and URM buildings”. In: *Bulletin of Earthquake Engineering* 4.4, pages 391–413 (cited on page 42).

Lallemant, D., A. Kiremidjian, and H Burton (2015). “Statistical procedures for developing earthquake damage fragility curves”. In: *Earthquake Engineering and Structural Dynamics* 44, pages 1373–1389 (cited on page 82).

Lin, Y.-Y. and E. Miranda (2008). “Noniterative equivalent linear method for evaluation of existing structures”. In: *Journal of structural engineering* 134.11, pages 1685–1695. (Visited on 06/18/2015) (cited on pages 16, 67, 68).

Miranda, E. (2000). “Inelastic Displacement Ratios for Structures on Firm Sites”. In: *Journal of AAPOSStructural Engineering* 126.10, pages 1150–1159 (cited on pages 16, 68).

Pagani, M., D. Monelli, G. Weatherill, L. Danciu, H. Crowley, V. Silva, P. Henshaw, L. Butler, M. Nastasi, L. Panzeri, M. Simionato, and D. Vigano (2014). “OpenQuake Engine: An open hazard (and risk) software for the Global Earthquake Model”. In: *Seismological Research Letters* (cited on pages 11, 21, 81, 83).

Panagiotakos, T. and M. Fardis (2001). “Deformation of R.C. members at yielding and ultimate”. In: *ACI Structural Journal* 98, pages 135–48 (cited on page 48).

Rossetto, T. and A. Elnashai (2005). “A new analytical procedure for the derivation of displacement-based vulnerability curves for populations of RC structures”. In: *Engineering Structures* 27, pages 397–409 (cited on pages 37, 41).

Ruiz-Garcia, J. and E. Miranda (2007). “Probabilistic estimation of maximum inelastic displacement demands for performance-based design”. In: *Earthquake Engineering and Structural Dynamics* 36, pages 1235–1254 (cited on pages 16, 59, 62, 63).

Silva, V, H Crowley, R Pinho, and H Varum (2013). “Extending Displacement-Based Earthquake Loss Assessment (DBELA) for the Computation of Fragility Curves.” In: *Engineering Structures* 56, pages 343–356 (cited on pages 16, 37, 38, 43, 44, 76).

Silva, V., H. Crowley, M. Pagani, D. Monelli, and R. Pinho (2014a). “Development of the OpenQuake engine, the Global Earthquake Model open-source software for seismic risk assessment”. en. In: *Natural Hazards* 72.3, pages 1409–1427. ISSN: 0921-030X, 1573-0840. (Visited on 06/04/2015) (cited on pages 19, 21, 27, 81, 83).

Silva, V., H. Crowley, R. Pinho, and H. Varum (2014b). “Evaluation of Analytical Methodologies Used to Derive Vulnerability Functions”. In: *Earthquake Engineering and Structural Dynamics* 43.2, pages 181–204 (cited on pages 50, 51, 70).

— (2014c). “Investigation of the characteristics of Portuguese regular moment-frame RC buildings and development of a vulnerability model”. In: *Bulletin of Earthquake Engineering* (cited on pages 37, 39).

Vamvatsikos, D. and A. Cornell (2005). “Direct Estimation of the Seismic Demand and Capacity of Oscillators with Multi-Linear Static Pushovers through IDA”. In: *Earthquake Engineering and Structural Dynamics* 0, pages 1–20 (cited on pages 16, 37, 41, 55).

Vidic, T., P. Fajfar, and F. M. (1994). “Consistent inelastic design spectra: Strength and displacement.” In: *Earthquake Engineering and Structural Dynamics* 23.5, pages 507–521 (cited on pages 16, 65, 66).

Other Sources

Abo El Ezz, A. (2008). “Deformation and strength based assessment of seismic failure mechanisms for existing RC frame buildings”. Master’s thesis. ROSE School, Pavia, Italy (cited on pages 44, 45).

ASCE7-10 (2010). *Minimum Design Loads for Buildings and Other Structures*. Technical report. American Society of Civil Engineers, Reston, VA (cited on page 70).

ATC-40 (1996). *Seismic Evaluation and Retrofit of Concrete Buildings*. Technical report. Applied Technology Council, Redwood City, CA (cited on pages 53, 71).

ATC-63 (2007). *Recommended Methodology for Quantification of Building System Performance and Response Parameters*. Technical report. Applied Technology Council, Redwood City, CA (cited on page 57).

Bal, I., H. Crowley, and R Pinho (2010). *Displacement-based earthquake loss assessment: method development and application to Turkish building stock*. Research Report ROSE 2010-02. IUSS Press, Pavia, Italy (cited on pages 42–44, 48, 76).

Brzev, S., C. Scawthorn, A. Charleson, L. Allen, M. Greene, K. Jaiswal, and V. Silva (2013). *GEM Building Taxonomy Version 2.0*. Technical report. Global Earthquake Model Foundation (cited on page 81).

CEN (2005). *Eurocode 8: Design of Structures for Earthquake Resistance - Part 1: General rules, seismic actions and rules for buildings.* Technical report EN 1998-2. Comite Europeen de Normalisation, Brussels, Belgium (cited on pages 16, 70, 76).

Crowley, H and R Pinho (2006). "Simplified equations for estimating the period of vibration of existing buildings". In: *Proceedings of the 1st European Conference on Earthquake Engineering and Seismology, Geneva, Switzerland*. 1122 (cited on pages 45, 49, 77).

D'Ayala D., M. A. V. D. P. K. R. T. C. H. S. V. (2014). *Guidelines for Analytical Vulnerability Assessment of low/mid-rise Buildings*. Technical report. GEM foundation (cited on pages 32, 34, 35).

Di Pasquale, G. and A. Goretti (2001). "Vulnerabilita funzionale ed economica degli edifici residenziali colpiti dai recenti eventi sismici italiani". In: *Proceedings of the 10th national conference Lingegneria Sismica in Italia, Potenza-Matera, Italy.* (Cited on page 42).

FEMA-440 (2005). *Improvement of Nonlinear Static Seismic Analysis Procedures*. Technical report. California, USA (cited on pages 16, 32, 34, 35, 53, 72).

FEMA-443 (2003). *HAZUS-MH Technical Manual*. Technical report. Federal Emergency Management Agency, Washington DC, USA. (cited on page 42).

Freeman, S., J. Nicoletti, and J. Tyrell (1975). "Evaluation of Existing Buildings for seismic risk - A case study of Puget Sound Naval Shipyard, Bremerton". In: *Proceedings of the 1st U.S. National Conference on Earthquake Engineering, Berkley, USA* (cited on page 71).

GEM (2015). *The OpenQuake-engine User Manual.* Technical report. Global Earthquake Model Foundation (cited on pages 81, 82, 84).

Kowalsky, M. J. (1994). "Displacement-based design - a methodology for seismic design applied to RC bridge columns." Master's thesis. University of California, San Diego (cited on page 73).

Lowes, L., N. Mitra, and A. Altoontash (2003). *A Beam-Column Joint Model for Simulating the Earthquake Response of Reinforced Concrete Frames*. PEER Report 2003/10. University of California, Berkeley, CA (cited on pages 78, 79).

McKenna, F., G. Fenves, M. Scott, and B. Jeremic (2000). *Open System for Earthquake Engineering Simulation (OpenSees). Pacific Earthquake Engineering Research Center*. Technical report. University of California, Berkeley, CA (cited on page 78).

Pinho, R., J. J. Bommer, and S. Glaister (2002). "A simplified approach to displacement-based earthquake loss estimation analysis". In: *Proceedings of the 12th European Conference on Earthquake Engineering, London, England* (cited on page 44).

Priestley, M. J. N., G. M. Calvi, and M. J. Kowalsky (2007). *Displacement-based Seismic Design of Structures*. Technical report. IUSS Press, Pavia, Italy (cited on pages 48, 73, 76).

Rezaeian, S., Y. Bozorgnia, I. Idriss, K. Campbell, N. Abrahamson, and W. Silva (2012). *Spectral Damping Scaling Factors for Shallow Crustal Earthquakes in Active Tectonic Regions*. PEER

Report 2012/01. Pacific Earthquake Engineering Research Center, California, USA (cited on page 76).

Vamvatsikos, D. (2014). *A static pushover methodology for extracting fragility curves with record-to-record dispersion information*. Technical report. GEM foundation (cited on page 62).

Weatherill, G. (2014). *Hazard Modeller's Toolkit - User Guide*. Technical report. Global Earthquake Model (GEM) (cited on pages 12, 89).

Glossary