Hazard Modeller's Toolkit
Documentation & Tutorial

July 2013

GEM Foundation, Pavia

Contents

Contents

Introduction to the Hazard Modeller's Toolkit

The objectives of this chapter are:

- Outline the purpose and creation of the Hazard Modeller's Toolkit

- Setup and Installation of the Software

The Hazard Modeller's Toolkit (or "hmtk") is a Python 2.7 library of functions written by scientists at the GEM Model Facility, which is intended to provide scientists and engineers with the tools to help create the seismogenic input models that go into the OpenQuake hazard engine. The process of developing building the hazard model is a complex and often challenging one, and whilst many aspect of the practice are relatively common, the choice of certain methods or tools for undertaking each step can be a matter of judgement. The intention of this software is to provide scientists and engineers with the means to apply many of the most commonly used algorithms for preparing seismogenic source models using seismicity and geological data. It is still in an early stage of development and the current versions contain only a preliminary set of tools for undertaking the necessary workflows. In forthcoming versions will hope to make available more tools for the current processes indicated here, and to integrate new functionalities for i) merging and homogenisation of earthquake catalogues, ii) calculation of activity rates from geological and geodetic data, iii) testing and interpretation of Ground Motion Prediction Equations, and iv) integration of seismological and geological data and treatment of uncertainty in the construction of seismogenic source zones.

1.1. The Development Process

The Hazard Modeller's Toolkit is an activity that has been under development in GEM and has followed different stages. The present decision to make the modelling tools available as a library reflects the general trend in the OpenQuake development process toward having a modular software framework. This means that the modelling - hazard - risk process is separated into libraries (e.g. oq-hazardlib, oq-risklib) that can be utilised as standalone tools, in addition to being integrated within the OpenQuake engine and platform. This is designed to allow for flexibility in the process, to allow the user to begin to utilise (possibly in other contexts) functions and classes that are intended to address particular stages of the calculation. Such an approach ensure that each sub-component of the toolkit is fully tested, with a minimal degree of duplication in the testing process. In the HMTK this is taken a step further as we are aiming to provide the hazard modeller as much control over the modelling process as possible, whilst retaining as complete a level of code testing as is practical to implement given the development resources available.

The change in the HMTK development process that leads towards the current version is designed to address particular objectives:

Portability Reduction in the number of python dependencies to allow for a greater degree of cross-platform deployment than is currently feasible with the main Open-Quake engine

Adaptability Cleaner separation of methods into self-contained components that can be implemented and tested within requiring adaption of the remainder of the code.

Abstraction This concept is often a critical component object-oriented development. It describes the specification of a core behaviour of a method, which implementations (by means of the subclass) must follow. For example, a declustering algorithm must follow the common behaviour path, in this instance i) reading and earthquake catalogue and some configurable parameters, ii) identifying the clusters of events, iii) identifying the mainshocks from within each cluster,iv) returning this information to the user. The details of the implementation are then dependent on the algorithm, providing that the core flow is met. This is designed to allow the algorithms to be *interchangeable* in the sense that different methods for particular task could be selected with no (or at least minimal) modification to the rest of the code.

Usability The creation of a library which could itself be embedded within larger applications (e.g. as part of a graphical user interface).

1.2. Getting Started and Running the Software

The Modeller's Toolkit and associated software are designed for execution from the command line. As with OpenQuake, the preferred environment is Ubuntu Linux (11.04 or later). A careful effort has been made to keep the number of additional dependencies to a minimum. No packaged version of the software has been released at the time of writing, so the user must install the dependencies manually. More information regarding the current dependencies of the toolkit can be found at
http://github.com/GEMScienceTools/hmtk.

The current dependencies are:

- Numpy and Scipy (included in the standard OpenQuake installation)

- Shapely (included in the standard OpenQuake installation)

- Openquake nrmllib (included in the standard OpenQuake installation)
  (http:/github.com/gem/oq-nrmllib)

- Openquake hazardlib (included in the standard OpenQuake installation)
  (http:/github.com/gem/oq-hazardlib)

- Matplotlib (http://matplotlib.org/)

- PyYaml

- Python Decorator

If the OpenQuake Hazard Library (oq-hazardlib) and the OpenQuake Nrml library are already installed on your system (as would be the case if a full installation of the OpenQuake-engine has already been successfully installed) then both dependencies are already available. If this is not the case, however, you will need to install the packages manually. For Linux and OSX users, the recommended approach is to run:

```
~$ sudo pip install -e https://github.com/gem/oq-hazardlib.git
~$ sudo pip install -e https://github.com/gem/oq-nrmllib.git
```

As of April 2013, the packages have not been tested in a Windows environment, so the recommended approach for windows users would be to use Ubuntu Linux 12.04 within a Virtual Machine.

The Matplotlib, Pyyaml and Decorator dependencies are installed in the library for the demos, but can be installed easy from the command line by:

```
~$ sudo pip install matplotlib
~$ sudo pip install pyyaml
~$ sudo pip install decorator
```

To enable usage of the hmtk within any location in the operating system, OSX and Linux users should add the folder location manually to the command line profile file. This can be done as follows:

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

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

2. At the bottom of the profile file (if one does not exist it will be created) add the line:

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

Where `/path/to/hmtk/folder/ is the system path to the location of the hmtk folder` from within the hmtk folder to view the full system path).

3. Re-source the bash shell via the command

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

1.2.1. Windows Installation

Although this installation has been primarily tested in a Linux/Unix environment it may be possible to install natively in Windows using the following process. This assumes that no other version of Python is installed in your windows environment.

The easiest way to install the OpenQuake hazardlib and nrmllib is by virtue of the PythonXY program http://code.google.com/p/pythonxy/. This is a free and open python user interface, which will bring in almost all of the dependencies the openquake libraries need. The an installer for the latest version of PythonXY can be downloaded from here: http://code.google.com/p/pythonxy/wiki/Downloads?tm=2.

Click on the executable and follow the instructions (the installation may take up to half an hour or more, depending on the system). It is strongly recommended that the use opt for the "**FULL**" installation, which should bring in almost all of the dependencies needed for installing the OpenQuake hazard library.

For the oq-nrmllib it is necessary to install the Lxml library (http://lxml.de). This is not officially supported for Windows, so it is recommended (by Python developers themselves!) to install the unofficial Lxml binding from here: http://www.lfd.uci.edu/ gohlke/python-libs/. Download and then run the 32-bit package named "lxml-#.#.#.win32-py2.7.exe".

Next you will need to install the op-hazardlib and oq-nrmllib. From the web repositories listed previously click the button "Download Zip", then extract contents to the folders `C:/oq-hazardlib` and `C:/oq-nrmllib` respectively.

Now open up an enhanced IPython console. Go to Start $->$ Python(xy) $->$ Enhanced Consoles $->$ Ipython (sh). This will open up an Ipython console terminal. To install the oq-hazardlib, at the console prompt type:

```
~$: cd C:/oq-hazardlib/
~$: python setup.py install build --compiler=mingw32
```

This will install the oq-hazardlib with the full C-extensions, which speed up some of the geometry calculations. Then do the same with the oq-nrmllib.

```
~$: cd C:/oq-hazardlib/
~$: python setup.py install
```

Then close the console by typing `exit`.

Finally, download the zipped folder of the hmtk from the github repository

1.2.2. Current Features

The Hazard Modeller's Toolkit is currently divided into three sections, which can be used interactively:

1. **Earthquake Catalogue and Seismicity Analysis** These functions are intended to address the needs of defining seismic activity rate from an earthquake catalogue. They algorithms for identification of Non-Poissonian events (declustering), analysis of catalogue completeness, calculation of activity rate and b-value and, finally, estimation of maximum magnitude using statistical analyses of the earthquake catalogue. Also included in these tools is an initial implementation of a smoothed seismicity algorithm using the Frankel [1995] approach.

2. **Active Faults Source Models from Geological Data**

   These functions are intended to address the Modeller needs for defining earthquake activity rates on fault sources from the geological slip rate, including support for some epistemic uncertainty analysis on critical parameters in the process.

3. **Seismic Source Models from Geodetic Data**

   These functions are intended to address the use of geodetic data to derive seismic activity rates from a strain rate model for a region, implementing the Seismic Hazard Inferred from Tectonics (SHIFT) methodology developed by Bird and Liu [2007] and applied on a global scale by Bird et al. [2010].

Feature	Algorithm
Seismicity	
Declustering	Gardner and Knopoff [1974]
	AFTERAN (Musson [1999b])
Completeness	Stepp [1971]
Recurrence	Maximum Likelihood Aki [1965]
	Time-dependent MLE
	Weichert [1980]
Smoothed Seismicity	Frankel [1995]
Geology	
Recurrence	Anderson and Luco [1983] "Arbitrary"
	Anderson and Luco [1983] "Area M_{MAX}"
	Characteristic (Truncated Gaussian)
	Youngs and Coppersmith [1985] Exponential
	Youngs and Coppersmith [1985] Characteristic
Geodetic Strain	
Recurrence	Seismic Hazard Inferred from Tectonics (SHIFT)
	Bird and Liu [2007]; Bird et al. [2010]

A summary of the algorithms available in the present version (V0.1) is given in Table

1.2.3. About this Tutorial

As previously indicated, the Modeller's Toolkit itself is a Python library. This means that its functions can be utilised in many different python applications. It is not, at present, a stand-alone software, and requires some investment of time from the user to understand the functionalities and learn how to link the various tools together into a workflow that will be suitable for the modelling problem at hand.

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. The tutorial itself does not specifically require a working knowledge of Python. However, an understanding of the basic python data types, and ideally some familiarity with the use of Python objects, is highly desirable. A good overview of the features of the python programming language can be found in the standard python documentation (http://docs.python.org/2/tutorial/). Where necessary particular Python programming concepts will be explained in further detail.

The code snippets (indicated by verbatim text) can be executed from within an "Interactive Python" environment, or may form the basis for usage of the hmtk in other python scripts that the user may wish to run construct themselves. If not already installed on your system, ipython can be installed from the python package repository by entering:

```
~$ sudo pip install ipython
```

1. Starting with a single magnitude-homogenised earthquake catalogue apply a declustering algorithm to identify and remove non-Poissonian events

2. For the declustered catalogue, use a statistical methodology to determine the time variation in completeness. Currently only the methodology of Stepp [1971] is implemented.

3. With the completeness periods of the catalogue known calculate a Gutenberg and Richter [1944] recurrence model (a- and b-value with uncertainties) for either the catalogue as a whole, or for earthquakes selected within certain area zones.

4. Implementation of a statistical estimator of maximum magnitude from the earthquake catalogue

An "interactive" session can then be opened by typing `ipython` at the command prompt. If matplotlib is installed and you want to make use of one or two of the plotting functionalities then you should open ipython with `ipython --pylab`. To exit an ipython session at any time simply type `exit`.

The 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 shown previously, then unzip 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/hmtk.git
```

Seismicity Tools

The objectives of this chapter are:

- Describe features of the seismicity tools of the Hazard Modeller's Toolkit

- Run simple calculations using the seismicity tools

2.1. The Seismicity Tools

The seismicity tools are intended for use in deriving activity rates from an observed earthquake catalogue, which may include both instrumental and historical seismicity. The tools are broken down into five separate libraries: i) Declustering, ii) Completeness, iii) Calculation of Gutenberg-Richter a- and b-value, iv) Statistical estimators of maximum magnitude from seismicity) and v) Smoothed Seismicity. In a common use case it is likely that many of the above methods, particularly recurrence and maximum magnitude estimation, may need to be applied to a selected sub-catalogue (e.g. earthquakes within a particular polygon). The toolkit allows for the creation of a source model containing one or more of the supported OpenQuake seismogenic source typologies, which can be used as a reference for selection, e.g. events within an area source (polygon), events within a distance of a fault etc. The supported input formats for both the catalogue and the source model are described below.

2.1.1. The Catalogue Format and Class

The input catalogue must be formatted as a comma-separated value file (.csv), with the following attributes in the header line (attributes with an * indicate essential attributes), although the order of the columns need not be fixed:

To load the catalogue using the IPython environment, in an open IPython session type:

13

Attribute	Description
eventID*	A unique identifier (integer) for each earthquake in the catalogue
Agency	The code (string) of the recording agency for the event solution
year*	Year of event (integer) in the range -10000 to present (events before common era (BCE) should have a negative value)
month*	Month of event (integer)
day*	Day of event (integer)
hour*	Hour of event (integer) - if unknown then set to 0
minute*	Minute of event (integer) - if unknown then set to 0
second*	Second of event (float) - if unknown set to 0.0
timeError	Error in event time (float)
longitude*	Longitude of event, in decimal degrees (float)
latitude*	Latitude of event, in decimal degrees (float)
SemiMajor90	Length (km) of the semi-major axis of the 90 % confidence ellipsoid for location error (float)
SemiMinor90	Length (km) of the semi-minor axis of the 90 % confidence ellipsoid for location error (float)
ErrorStrike	Azimuth (in degrees) of the 90 % confidence ellipsoid for location error (float)
depth*	Depth (km) of earthquake (float)
depthError	Uncertainty (as standard deviation) in earthquake depth (km) (float)
magnitude*	Homogenised magnitude of the event (float) - typically Mw
sigmaMagnitude*	Uncertainty on the homogenised magnitude (float) typically Mw

Table 2.1 – *List of Attributes in the Earthquake Catalogue File (* Indicates Essential)*

```
[1] from hmtk.parsers.catalogue import CsvCatalogueParser

[2] catalogue_filename = 'path/to/catalogue_file.csv'

[3] parser = CsvCatalogueParser(catalogue_filename)

[4] catalogue = parser.read_file()
```

N.B. the csv file can contain additional attributes of the catalogue too and will be parsed correctly; however, if the attribute is not one that is specifically recognised by the catalogue class then a message will be displayed indicating:

```
Catalogue Attribute ... is not a recognised catalogue key
```

This is expected behaviour and simply indicates that although this data is given in the input file, it is not retained in the data dictionary.

The variable `catalogue` is an instance of the class hmtk.seismicity.catalogue.Catalogue, which now contains the catalogue itself (as `catalogue.data`) and some methods that can be applied to the catalogue. The first attribute (`catalogue.data`), is a dictionary where each attribute of the catalogue is either a 1-D numpy vector (for float and integer values) or a python list (for string values). For example, to return a vector containing all the magnitudes in the `magnitude` column of the catalogue simply type:

```
In [5]: catalogue.data['magnitude']
Out [e.g.] [5]: array([ 6.5,  6.5,  6. , ...,  4.8,  5.2,  4.1])
```

The catalogue class contains several helpful methods (called via `catalogue. ...`):

- `catalogue.get_number_events()` Returns the number of events currently in the catalogue (integer)

- `catalogue.load_to_array(keys)` Returns a numpy array of floating data, with the columns ordered according to the list of keys. If the key corresponds to a string item (e.g. Agency) then an error will be raised.

```
In [6]: catalogue.load_to_array(['year', 'longitude', 'latitude', 'depth', 'magnitude'])

Out[6]:
array([[ 1910.   ,    26.941,    38.507,    13.2 ,    6.5 ],
       [ 1910.   ,    22.19 ,    37.72 ,    20.4 ,    6.5 ],
       [ 1910.   ,    28.881,    33.274,    25.  ,    6.  ],
       ...,
       [ 2009.   ,    20.054,    39.854,    20.2 ,    4.8 ],
       [ 2009.   ,    23.481,    38.05 ,    15.2 ,    5.2 ],
       [ 2009.   ,    28.959,    34.664,    18.4 ,    4.1 ]])
```

- `catalogue.load_from_array(keys, data_array)` Creates the catalogue data dictionary from an array, given header as an ordered list of dictionary keys. This can be used in the case where the earthquake catalogue is loaded in a simple ascii format. For example, if the user wishes to load in a catalogue from the Zmap format, which gives the columns as:

  ```
  longitude, latitude, year, month, day, magnitude, depth, hour,
  minute, second
  ```

  This file type could be parsed into a catalogue without the need of a specific parser, as follows:

  ```
  In [1]: import numpy
  # Assuming no headers in the file (set skip_header=1 if headers are found)
  In [2]: data = numpy.genfromtxt('PATH/TO/ZMAP_FILE.txt', skip_header=0)

  In [3]: headers = ['longitude', 'latitude', 'year', 'month', 'day',
                     'magnitude', 'depth', 'hour', 'minute', 'second']

  # Create instance of a catalogue class
  In [4]: from hmtk.seismicity.catalogue import Catalogue
  In [5]: catalogue = Catalogue()

  # Load the data array into the catalogue
  In [6]: catalogue.load_from_array(data, headers)
  ```

- `catalogue.get_decimal_time()` Returns the time of the earthquake in a decimal format

- `catalogue.hypocentres_as_mesh()` Returns the hypocentres of an earthquake as an instance of the openquake.hazardlib.geo.mesh.Mesh class (useful for geospatial functions)

- `catalogue.hypocentres_to_cartesian()` Returns the hypocentres in a 3D cartesian framework

- `catalogue.purge_catalogue(flag_vector)` Purges the catalogue of all `False` events in the boolean vector. Thus is used for removing foreshocks and aftershocks from a catalogue after the application of a declustering algorithm.

- `catalogue.sort_catalogue_chronologically()` Sorts an input into chronological order (N.B. some methods will implicitly assume that the catalogue is in chronological order, so it is recommended to run this function if you believe that there may be events out of order)

- `catalogue.select_catalogue_events(IDX)` Orders the catalogue according to the event order specified in IDX. Behaves the same as `catalogue.purge_catalogue(IDX)` if IDX is a boolean vector

- `catalogue.get_depth_distribution(depth_bins, normalisation\False, boot-`

strapN̄one)= Returns a depth histogram for the catalogue using bins specified by `depth_bins`. If `normalisation`\True= then the function will return the histogram as a probability mass function, otherwise the original count will be returned. If uncertainties are reported on depth such that one or more values in `catalogue.data['depthError']` are greater than 0., the function will perform a bootstrap analysis, taking into account the depth error, with the number of bootstraps given by the keyword `bootstrap`. To generate a simple histogram plot of hypocentral depth, the process below can be followed to produce a depth histogram similar to the one shown in Figure :

```
# Import numpy and matplotlib
In [1]: import numpy as np
In [2]: import matplotlib.pyplot as plt

# Define depth bins for (e.g) 0. - 150 km in intervals of 10 km
In [3]: depth_bins = np.arange(0., 160., 10.)

# Get normalised histograms (without bootstrapping)
In [4]: depth_hist = catalogue.get_depth_distribution(depth_bins, normalisation=True)

# Create a simple histogram plot
In [5]: plt.bar(depth_bins[:-1], depth_hist, width=10.)

# Add some labels
In [6]: plt.xlabel('Depth (km)')
In [7]: plt.ylabel('Density')
```

- `catalogue.get_magnitude_depth_distribution(magnitude_bins, depth_bins, normalisation`\Fa
  bootstrapN̄one)= Returns a two-dimensional histogram of magnitude and hypocentral depth, with the corresponding bins defined by the vectors `magnitude_bins` and `depth_bins`. The options `normalisation` and `bootstrap` are the same as for the one dimensional histogram.

2.1.2. The Source Model Format

The seismic source model formats currently required by the hmtk are the nrml The source model is both input and output in GEM's NRML (`Natural Risk Markup Language`) format (although support for shapefile input definitions are expected in future releases). However, unlike the OpenQuake engine, for which each source typology must contain all of the necessary attributes, it is recognised that it may be desirable to use the seismic source model with a partially defined source (one for which only the ID, name and geometry are known) in order to make use of the modelling tools. Therefore, the validation checks have been relaxed to allow for data such as the recurrence model, the hypocentral depth distribution and the faulting mechanism to be specified at a later stage. However, if using this minimal format it will not be possible to use the resulting output file in OpenQuake until the remaining information is filled in.

A full description of the complete nrml seismogenic source model format is found in the OpenQuake Version 1.0 manual Crowley et al. [2010]. An example of a minimal format is shown below for:

2.1.2.1. Point Source

```
<?xml version='1.0' encoding='utf-8'?>
<nrml xmlns:gml="http://www.opengis.net/gml" xmlns="http://openquake.org/xmlns/nrml/0.4">
    <sourceModel name="Some Source Model">
        <pointSource id="2" name="point" tectonicRegion="">

        <pointGeometry>
            <gml:Point>
                <gml:pos>-122.0 38.0</gml:pos>
            </gml:Point>

            <upperSeismoDepth>0.0</upperSeismoDepth>
            <lowerSeismoDepth>10.0</lowerSeismoDepth>
        </pointGeometry>

        <magScaleRel></magScaleRel>
        <ruptAspectRatio></ruptAspectRatio>

        <truncGutenbergRichterMFD aValue="" bValue="" minMag="" maxMag="" />

        <nodalPlaneDist>
            <nodalPlane probability="" strike="" dip="" rake="" />
            <nodalPlane probability="" strike="" dip="" rake="" />
        </nodalPlaneDist>

        <hypoDepthDist>
            <hypoDepth probability="" depth="" />
            <hypoDepth probability="" depth="" />
        </hypoDepthDist>

    </pointSource>
    </sourceModel>
</nrml>
```

2.1.2.2. Area Source

```xml
<?xml version='1.0' encoding='utf-8'?>
<nrml xmlns:gml="http://www.opengis.net/gml" xmlns="http://openquake.org/xmlns/nrml/0.4">

<sourceModel name="Some Source Model">
    <!-- Note: Area sources are identical to point sources, except for the geometry. -->
    <areaSource id="1" name="Quito" tectonicRegion="">
        <areaGeometry>
            <gml:Polygon>
                <gml:exterior>
                    <gml:LinearRing>
                        <gml:posList>
                          -122.5 38.0
                          -122.0 38.5
                          -121.5 38.0
                          -122.0 37.5
                        </gml:posList>
                    </gml:LinearRing>
                </gml:exterior>
            </gml:Polygon>

            <upperSeismoDepth>0.0</upperSeismoDepth>
            <lowerSeismoDepth>10.0</lowerSeismoDepth>
        </areaGeometry>

        <magScaleRel></magScaleRel>

        <ruptAspectRatio></ruptAspectRatio>

        <incrementalMFD minMag="" binWidth="">
            <occurRates></occurRates>
        </incrementalMFD>

        <nodalPlaneDist>
            <nodalPlane probability="" strike="" dip="" rake="" />
            <nodalPlane probability="" strike="" dip="" rake="" />
        </nodalPlaneDist>

        <hypoDepthDist>
            <hypoDepth probability="" depth="" />
            <hypoDepth probability="" depth="" />
        </hypoDepthDist>

    </areaSource>
    </sourceModel>
</nrml>
```

2.1.2.3. Simple Fault Source

```
<?xml version='1.0' encoding='utf-8'?>
<nrml xmlns:gml="http://www.opengis.net/gml" xmlns="http://openquake.org/xmlns/nrml/0.4">
    <sourceModel name="Some Source Model">
        <simpleFaultSource id="3" name="Mount Diablo Thrust" tectonicRegion="">

            <simpleFaultGeometry>
                <gml:LineString>
                    <gml:posList>
                        -121.82290 37.73010
                        -122.03880 37.87710
                    </gml:posList>
                </gml:LineString>

                <dip>45.0</dip>
                <upperSeismoDepth>10.0</upperSeismoDepth>
                <lowerSeismoDepth>20.0</lowerSeismoDepth>
            </simpleFaultGeometry>

            <magScaleRel></magScaleRel>

            <ruptAspectRatio></ruptAspectRatio>

            <incrementalMFD minMag="" binWidth="">
                <occurRates></occurRates>
            </incrementalMFD>

            <rake></rake>
        </simpleFaultSource>
    </sourceModel>
</nrml>
```

2.1.2.4. Complex Fault Source

```
<?xml version='1.0' encoding='utf-8'?>
<nrml xmlns:gml="http://www.opengis.net/gml" xmlns="http://openquake.org/xmlns/nrml/0.4">
    <sourceModel name="Some Source Model">
    <complexFaultSource id="4" name="Cascadia Megathrust" tectonicRegion="">

        <complexFaultGeometry>
            <faultTopEdge>
                <gml:LineString>
                    <gml:posList>
                        -124.704 40.363 0.5493260E+01
                        -124.977 41.214 0.4988560E+01
                        -125.140 42.096 0.4897340E+01
                    </gml:posList>
                </gml:LineString>
            </faultTopEdge>

            <intermediateEdge>
                <gml:LineString>
                    <gml:posList>
                        -124.704 40.363 0.5593260E+01
                        -124.977 41.214 0.5088560E+01
                        -125.140 42.096 0.4997340E+01
                    </gml:posList>
                </gml:LineString>
            </intermediateEdge>

            <intermediateEdge>
                <gml:LineString>
                    <gml:posList>
                        -124.704 40.363 0.5693260E+01
                        -124.977 41.214 0.5188560E+01
                        -125.140 42.096 0.5097340E+01
                    </gml:posList>
                </gml:LineString>
            </intermediateEdge>

            <faultBottomEdge>
                <gml:LineString>
                    <gml:posList>
                        -123.829 40.347 0.2038490E+02
                        -124.137 41.218 0.1741390E+02
                        -124.252 42.115 0.1752740E+02
                    </gml:posList>
                </gml:LineString>
            </faultBottomEdge>
        </complexFaultGeometry>

        <magScaleRel></magScaleRel>

        <ruptAspectRatio></ruptAspectRatio>

        <truncGutenbergRichterMFD aValue="" bValue="" minMag="" maxMag="" />

        <rake></rake>
    </complexFaultSource>
    </sourceModel>
</nrml>
```

To load in a source model such as those shown above, in an IPython environment simply execute the following:

```
In: [1] from hmtk.parsers.source_model.nrml04_parser import nrmlSourceModelParser

In: [2] model_filename = 'path/to/source_model_file.xml'

In: [3] model_parser = nrmlSourceModelParser(model_filename)

In: [4] model = model_parser.read_file()
Area source - ID: 1, name: Quito
Point Source - ID: 2, name: point
Simple Fault source - ID: 3, name: Mount Diablo Thrust
Complex Fault Source - ID: 4, name: Cascadia Megathrust
```

If loaded successfully a list of the source typology, ID and source name for each source will be returned to the screen as shown above. The variable `model` contains the whole source model, and can support multiple typologies (i.e. point, area, simple fault and complex fault).

2.2. Declustering

To identify Poissonian rate of seismicity, it is necessary to remove foreshocks/aftershocks/swarms from the catalogue. The Modeller's Toolkit contains, at present, two algorithms to undertake this task, with more under development.

2.2.1. Gardner and Knopoff [1974]

The most widely applied simple windowing algorithm is that of Gardner and Knopoff [1974]. Originally conceived for Southern California, the method simply identifies aftershocks by virtue of fixed time-distance windows proportional to the magnitude of the main shock. Whilst this premise is relatively simple, the manner in which the windows are applied can be ambiguous. Four different possibilities can be considered (Luen and Stark [2012]):

1. Search events in magnitude-descending order. Remove events if it is in the window of the largest event

2. Remove every event that is inside the window of a previous event, including larger events

3. An event is in a cluster if, and only if, it is in the window of at least one other event in the cluster. In every cluster remove all events except the largest

4. In chronological order, if the ith event is in the window of a preceding larger shock that has not already been deleted, remove it. If a larger shock is in the window of the ith event, delete the ith event. Otherwise retain the ith event.

It is the first of the four options that is implemented in the current toolkit, whilst others may be considered in future. The algorithm is capable if identifying foreshocks and aftershocks, simply by applying the windows forward and backward in time from the mainshock. No distinction is made between primary aftershocks (those resulting from the mainshock) and secondary or tertiary aftershocks (those originating due to the previous aftershocks); however, it is assumed all would occur within the window.

Several modifications to the time and distance windows have been suggested, which are summarised in van Stiphout et al. [2012]. The windows originally suggested by Gardner and Knopoff [1974] are approximated by:

$$\text{distance (km)} = 10^{0.1238M+0.983}$$
$$\text{time (decimal years)} = \begin{cases} 10^{0.032M+2.7389} & \text{if } M \geq 6.5 \\ 10^{0.5409M-0.547} & \text{otherwise} \end{cases} \qquad (2.1)$$

An alternative formulation is proposed by Grünthal (as reported in van Stiphout et al. [2012]):

$$\text{distance (km)} = e^{1.77+(0.037+1.02M)^2}$$
$$\text{time (decimal years)} = \begin{cases} |e^{-3.95+(0.62+17.32M)^2}| & \text{if } M \geq 6.5 \\ 10^{2.8+0.024M} & \text{otherwise} \end{cases} \qquad (2.2)$$

A further alternative is suggested by Uhrhammer [1986]

$$\text{distance (km)} = e^{-1.024+0.804M} \quad \text{time (decimal years)} = e^{-2.87+1.235M} \qquad (2.3)$$

A comparison of the expected window sizes with magnitude are shown for distance and time (Figure 2.1).

The Gardner and Knopoff [1974] algorithm and its derivatives represent are most computationally straightforward approach to declustering. The `time_dist_windows` attribute indicates the choice of the time and distance window scaling model from the three listed. As the current version of this algorithm considers the events in a descending-magnitude order, the parameter `foreshock_time_window` defines the size of the time window used for searching for foreshocks, as a fractional proportion of the size of the aftershock window (the distance windows are always equal for both fore- and aftershocks). So for an evenly sized time window for foreshocks and aftershocks, `foreshock_time_window`

(a)

(b)
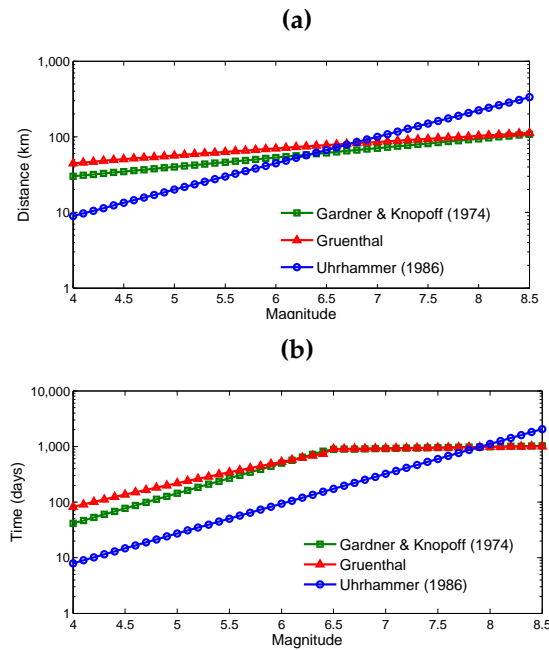
Figure 2.1 – *Scaling of declustering time and distance windows with magnitude*

should equal 1. For shorter or longer foreshock time windows this parameter can be reduced or increased respectively.

To run a declustering analysis on the earthquake catalogue it is necessary to set-up the configuration using a python dictionary.[1] A config file for the Gardner and Knopoff [1974] algorithm, using for example the Uhrhammer [1986] time-distance windows with equal sized time window for aftershocks and foreshocks, would be created as shown:

```
In [1]: from hmtk.seismicity.declusterer.distance_time_windows import UhrhammerWindow

In [2]: declust_config = {'time_distance_window': UhrhammerWindow(),
   ....:                  'fs_time_prop': 1.0}
```

To run the declustering algorithm simply import and run the algorithm as shown:

---

[1]A dictionary is Python's way of storing a mapping of data (including different types) to a set of keys (similar to a data structure in Matlab).

```
In [1]: from hmtk.seismicity.declusterer.distance_time_windows import UhrhammerWindow

In [2]: declust_config = {'time_distance_window': UhrhammerWindow(),
   ....:                   'fs_time_prop': 1.0}
```

```
In [3]: from hmtk.seismicity.declusterer.dec_gardner_knopoff import GardnerKnopoffType1

In [4]: declustering = GardnerKnopoffType1()

In [5]: cluster_index, cluster_flag = declustering.decluster(catalogue,
                                                              declust_config)
```

There are two outputs of a declustering algorithm: `cluster_index` and `cluster_flag`. Both are numpy vectors, of the same length as the catalogue, containing information about the clusters in the catalogue. `cluster_index` indicates the cluster to which each event is assigned (0 if not assigned to a cluster). `cluster_flag` indicates whether an event is a non-Poissonian event, in which case the value is assigned to 1, or a mainshock, the value is assigned as 0. This output definition is the same for all declustering algorithms.

At this point the user may wish to either retain the catalogue in its current format, in which case they may wish to add on the clustering information into another attribute of the catalogue.data dictionary, or they may wish to purge the catalogue of non-Poissonian events.

To simply add the clustering information to the data dictionary simply type:

```
In [6]: catalogue.data['Cluster_Index'] = cluster_index
In [7]: catalogue.data['Cluster_Flag'] = cluster_flag
```

Alternatively, to purge the catalogue of non-Poissonian events:

```
In [6]: mainshock_flag = cluster_flag == 0
In [7]: catalogue.purge_catalogue(mainshock_flag)
```

2.2.2. AFTERAN (Musson [1999a])

A particular development of the standard windowing approach is introduced in the program AFTERAN Musson [1999a]. This is a modification of the Gardner and Knopoff [1974] algorithm, using a moving time window rather than a fixed time window. In AF-

TERAN, considering each earthquake in order of descending magnitude, events within a fixed distance window are identified (the distance window being those suggested previously). These events are searched using a moving time window of T days. For a given mainshock, non Poissonian events are identified if they occur both within the distance window and the initial time window. The time window is then moved, beginning at the last flagged event, and the process repeated. For a given mainshock, all non-Poissonian events are identified when the algorithm finds a continuous period of T days in which no aftershock or foreshock is identified.

The theory of the AFTERAN algorithm is broadly consistent with that of Gardner and Knopoff [1974]. This algorithm, whilst a little more computationally complex, and therefore slower, than the Gardner and Knopoff [1974] windowing approach, remains simple to implement.

As with the Gardner and Knopoff [1974] function, the `time_dist_window` attribute indicates the choice of the time and distance window scaling model. The parameter `time_window` indicates the size (in days) of the moving time window used to identify fore- and aftershocks. The following example will show how to run the AFTERAN algorithm, using the Gardner and Knopoff [1974] definition of the distance windows, and a fixed-width moving time window of 100 days.

```
In [1] from hmtk.seismicity.declusterer.dec_afteran import Afteran

In [2]: from hmtk.seismicity.declusterer.distance_time_windows import GardnerKnopoffWindow

In [3]: declust_config = {'time_distance_window': GardnerKnopoffWindow(),
                          'time_window': 100.0}

In [4]: declustering = Afteran()

In [5]: cluster_index, cluster_flag = declustering.decluster(catalogue, declust_config)
```

2.3. Completeness

In the earliest stages of processing an instrumental seismic catalogue to derive inputs for seismic hazard analysis, it is necessary to determine the magnitude completeness threshold of the catalogue. To outline the meaning of the term "magnitude completeness" and the requirements for its analysis as an input to PSHA, the terminology of Mignan and Wöessner [2012] is adopted. This defines the magnitude of completeness as the "lowest magnitude at which 100 % of the events in a space-time volume are detected (Rydelek and Sacks [1989]; Woessner and Wiemer [2005])". Incompleteness of an earthquake catalogue will produce bias when determining models of earthquake recurrence, which may have a significant impact on the estimation of hazard at a site. Identification of the completeness magnitude of an earthquake catalogue is therefore a

clear requirement for the processing of input data for seismic hazard analysis.

It should be noted that this summary of methodologies for estimating completeness is directed toward techniques that can be applied to a "typical" instrumental seismic catalogue. We therefore make the assumption that the input data will contain basic information for each earthquake such as time, location, magnitude. We do not make the assumption that network-specific or station-specific properties (e.g., configuration, phase picks, attenuation factors) are known a priori. This limits the selection of methodologies to those classed as estimators of "sample completeness", which defines completeness on the basis of the statistical properties of the earthquake catalogue, rather than "probability-based completeness", which defines the probability of detection given knowledge of the properties of the seismic network (Schorlemmer and Woessner [2008]). This therefore excludes the methodology of (Schorlemmer and Woessner [2008]), and similar approaches such as that of Felzer [2008]

The current workflows assume that completeness will be applied to the whole catalogue, ideally returning a table of time-varying completeness. The option to explore spatial variation in completeness is not explicitly supported, but could be accommodated by an appropriate configuration of the toolkit.

In the current version (July 2013) of the Modeller's Toolkit only the Stepp [1971] methodology for analysis of catalogue completeness is implemented. Further methods are in development, and will be input in future releases.

2.3.1. Stepp [1971]

This is one of the earliest analytical approaches to estimation of completeness magnitude. It is based on estimators of the mean rate of recurrence of earthquakes within given magnitude and time ranges, identifying the completeness magnitude when the observed rate of earthquakes above MC begins to deviate from the expected rate. If a time interval (T_i) is taken, and the earthquake sequence assumed Poissonian, then the unbiased estimate of the mean rate of events per unit time interval of a given sample is:

$$\lambda = \frac{1}{n} \sum_{i=1}^{n} T_i \tag{2.4}$$

with variance $\sigma_\lambda^2 = \lambda/n$. Taking the unit time interval to be 1 year, the standard deviation of the estimate of the mean is:

$$\sigma_\lambda = \sqrt{\lambda}/\sqrt{T} \tag{2.5}$$

where T is the sample length. As the Poisson assumption implies a stationary process, σ_λ behaves as $1/\sqrt{T}$ in the sub-interval of the sample in which the mean rate of

occurrence of a magnitude class is constant. Time variation of M_C can usually be inferred graphically from the analysis, as is illustrated in Figure 2.2. In this example, the deviation from the $1/\sqrt{T}$ line for each magnitude class occurs at around 40 years for $4.5 < M < 5$, 100 years for $5.0 < M < 6.0$, approximately 150 years for $6.0 < M < 6.5$ and 300 years for $M > 6.5$. Knowledge of the sources of earthquake information for a given catalogue may usually be reconciled with the completeness time intervals.
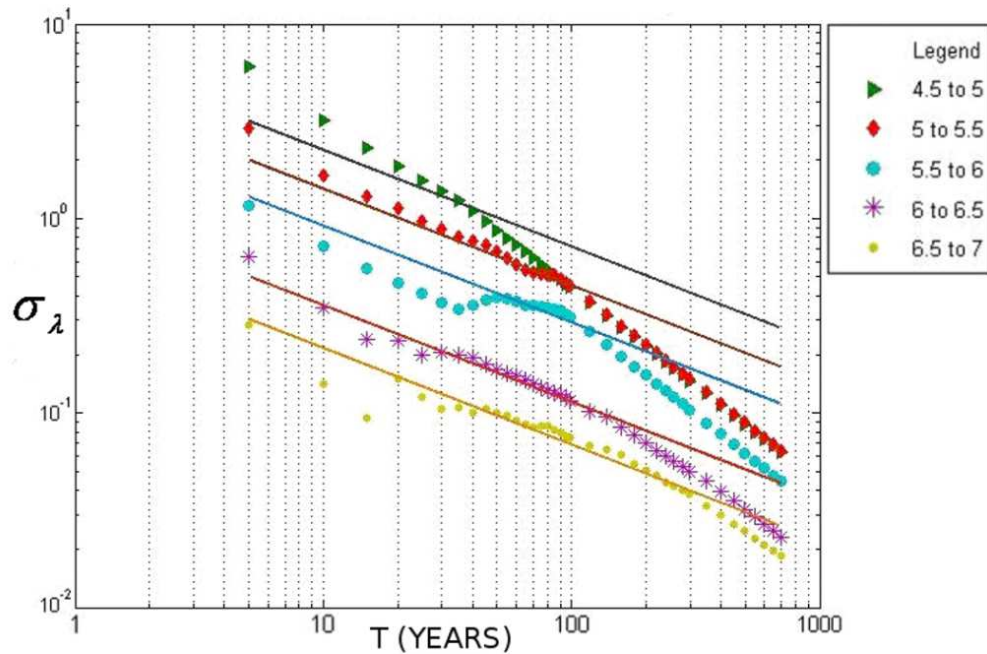
Figure 2.2 – *Example of Completeness Estimation by the Stepp [1971] methodology*

The analysis of Stepp [1971] is a coarse, but relatively robust, approach to estimating the temporal variation in completeness of a catalogue. It has been widely applied since its development. The accuracy of the completeness magnitude depends on the magnitude and time intervals considered, and a degree of judgement is often needed to determine the time at which the rate deviates from the expected values. It has tended to be applied to catalogues on a large scale, and for relatively higher completeness magnitudes.

To translate the methodology from a largely graphical methods into a computational method the completeness period needs to be identified by automatically identifying the point at which the gradient of the observed values decreases with respect to that expected from a Poisson process (see 2.2). In the implementation found within the current toolkit, the divergence point is identified by fitting a two-segment piecewise linear function to the observed data. Although a two-segment piecewise linear function is normally fit with four parameters (intercept, $slope_1$, $slope_2$ and crossover point), by

virtue of the assumption that for the complete catalogue the rate is assumed to be stationary such that $\sigma_\lambda = \frac{1}{\sqrt{T}}$ the slope of the first segment can be fixed as -0.5, and the second slope should be constrained such that $slope_2 \leq -0.5$, whilst the crossover point (x_c) is subject to the constraint ($x_c \geq 0.0$). Thus it is possible to fit the two-segment linear function using constrained optimisation with only three free parameters. For this purpose the toolkit minimises the residual sum-of-squares of the model fit using numerical optimisation.

To run the Stepp [1971] algorithm the configuration parameters should be entered in the form of a dictionary, such as the example shown below:

```
In [1]: comp_config = {'magnitude_bin': 0.5,
                       'time_bin': 5.,
                       'increment_lock': True}
```

The algorithm has three configurable options. The `time_bin` parameter describes the size of the time window in years, the `magnitude_bin` parameter describes the size of the magnitude bin, sensitivity is as described previously. The final option (`increment_lock`) is an option that is used to ensure consistency in the results to avoid the completeness magnitude increasing for the latest intervals in the catalogue simply due to the variability associated with the short duration. If `increment_lock` is set to `True`, the program will ensure that the completeness magnitude for shorter, more recent windows is less than or equal to that of older, longer windows. This is often a condition for some recurrence analysis tools, so it may be advisable to set this option to true in certain workflows. Otherwise it should be set to `False to show the apparent variability`. Some degree of judgement is necessary here. In particular it is expected that the user may be aware of circumstances particular to their catalogue for which a recent increase in completeness magnitude is expected (for example, a certain recording network no longer operational).

The process of running the algorithm is shown below:

```
In [2]: from hmtk.seismicity.completeness.comp_stepp_1971 import Stepp1971

In [3]: completeness_algorithm = Stepp1971()

In [4]: completeness_table = completeness_algorithm.completeness(catalogue, comp_config)

In [5]: completeness_table
Out[5]:
array([[ 1990.  ,      4.25],
       [ 1962.  ,      4.75],
       [ 1959.  ,      5.25],
       [ 1906.  ,      5.75],
       [ 1906.  ,      6.25],
       [ 1904.  ,      6.75],
       [ 1904.  ,      7.25]])
```

As shown in the resulting `completeness_table`, the completeness algorithm will output the time variation in completeness (in this example with the `increment_lock` set) in the form of a two-column table with column 1 indicating the completeness year for the magnitude bin centred on the magnitude value found in column 2.

At present, it may be the case that the user wishes to enter a time-varying completeness results for use in subsequent functions, based on alternative methods or on judgement. This can be entered in the `completeness_table` setting, as in the example shown here (take note of the requirements for the square brackets):

```
completeness_table: [[1990., 4.0],
                     [1960., 5.0],
                     [1930., 6.0],
                     [1900., 6.5]]
```

If a `completeness_table` is input then this will override the selection of the completeness algorithm, and the calculation will take the values in `completeness_table` directly.

2.4. Recurrence Models

The current sets of tools are intended to determine the parameters of the Gutenberg and Richter [1944] recurrence model, namely the a- and b-value. It is expected that in the most common use case the catalogue that is input to these algorithms will be declustered, with a time-varying completeness defined according to a `completeness_table` of the kind shown previously. If no `completeness_table` is input the algorithm will assume the input catalogue is complete above the minimum magnitude for its full duration.

2.4.1. Aki [1965]

The classical maximum likelihood estimator for a simple unbounded Gutenberg and Richter [1944] model is that of Aki [1965], adapted for binned magnitude data by Bender [1983]. It assumes a fixed completeness magnitude (M_C) for the catalogue, and a simple power law recurrence model. It does not explicitly take into account magnitude uncertainty.

$$b = \frac{\log_{10}(e)}{\bar{m} - m_0 + \left(\frac{\Delta M}{2}\right)} \tag{2.6}$$

where $\bar{m}$ is the mean magnitude, m_0 the minimum magnitude and ΔM the discretisation interval of magnitude within a given sample.

2.4.2. Maximum Likelihood

This method adjusts the Aki [1965] and Bender [1983] method to incorporate for time variation in completeness. The catalogue is divided into into S sub-catalogues, where each sub-catalogue corresponds to a period with a corresponding M_C. An average a- and b-value (with uncertainty) is returned by taking the mean of the a- and b-value of each sub-catalogue, weighted by the number of events in each sub-catalogue.

$$\hat{b} = \frac{1}{S} \sum_{i=1}^{S} w_i b_i \tag{2.7}$$

```
In [1]: mle_config = {'magnitude_interval': 0.1,
                      'Average Type': 'Weighted',
                      'reference_magnitude': None}

In [2]: from hmtk.seismicity.occurrence.b_maximum_likelihood import BMaxLikelihood

In [3]: recurrence = BMaxLikelihood()

In [4]: bval, sigmab, aval, sigmaa = recurrence.calculate(catalogue, mle_config,
   completeness=completeness_table)
```

Where `magnitude_window` indicates the size of the magnitude bin, `recurrence_algorithm` and `reference_magnitude` the magnitude for which the output calculates that rate greater than or equal to (set to 0 for 10^a).

2.4.3. **Kijko and Smit** [2012]

A recent adaption of the Aki [1965] estimator of b-value for a catalogue containing different completeness periods has been proposed by Kijko and Smit [2012]. Dividing the earthquake catalogue into s subcatalogues of n_i events with corresponding completeness magnitudes m_{c_i} for $i = 1, 2, ..., s$, the likelihood function of β where $\beta = b \ln(10.0)$ is given as:

$$\mathbf{L} = \prod_{i=1}^{s} \prod_{j=1}^{n_i} \beta \exp([-\beta(m_j^i - m_{min}^i)]) \tag{2.8}$$

which gives a maximum likelihood estimator of β:

$$\beta = \left(\frac{r_1}{\beta_1} + \frac{r_2}{\beta_2} + \cdots + \frac{r_s}{\beta_s} \right)^{-1} \tag{2.9}$$

where $r_i = n_i/n$ and $n = \sum_{i=1}^{s} n_i$ above the level of completeness m_i.

```
In [1]: kijko_smit_config = {'magnitude_interval': 0.1,
                             'reference_magnitude': None}
```

2.4.4. **Weichert** [1980]

Recognising the typical conditions of an earthquake catalogue, Weichert [1980] developed a maximum likelihood estimator of b for grouped magnitudes and unequal periods of observation. The likelihood formulation for this approach is:

$$\mathbf{L}(\beta|n_i, m_i, t_i) = \frac{N!}{\prod_i n_i!} \prod_i p_i^{n_i} \tag{2.10}$$

where $\mathbf{L}$ is the likelihood estimator of β, n the number of earthquakes in magnitude bin m with observation period t. The parameter p is defined as:

$$p_i = \frac{t_i \exp(-\beta m_i)}{\sum_j t_j \exp(-\beta m_j)} \tag{2.11}$$

The extremum of $\ln(\mathbf{L})$ is found at:

$$\frac{\sum_i t_i m_i \exp(-\beta m_i)}{\sum_j t_j \exp(-\beta m_j)} \tag{2.12}$$

The computational implementation of this method is given as an appendix to Weichert [1980]. This formulation of the maximum likelihood estimator for b-value, and consequently seismicity rate, is in widespread use, with applications in many national seismic hazard analysis [Frankel et al., 1996, 2002, e.g.]. The algorithm has been demonstrated to be efficient and unbiased for most applications. It is recognised by Felzer [2008] that an implicit assumption is made regarding the stationarity of the seismicity for all the time periods.

To implement the Weichert [1980] recurrence estimator, the configuration properties are defined as:

```
In [1]: weichert_config = {'magnitude_interval': 0.1,
                           'reference_magnitude': None,
                           # The remaining parameters are optional
                           'bvalue': 1.0,
                           'itstab': 1E-5,
                           'maxiter': 1000}
```

As the Weichert [1980] algorithm is reaches the MLE estimation by iteration then three additional optional parameters can control the iteration process: `bvalue` is the initial guess for the b-value, `itstab` the difference in b-value in order to reach convergence, and `maxiter` the maximum number of iterations. [2]

2.5. Maximum Magnitude

The estimation of the maximum magnitude for use in seismic hazard analysis is a complex, and often controversial, process that should be guided by information from geology and the seismotectonics of a seismic source. Estimation of maximim magnitude from the observed (instrumental and historical) seismicity can be undertaken using methods assuming a truncated Gutenberg and Richter [1944] model, or via non-parametric methods that are independent any assumed functional form.

2.5.1. Kijko [2004]

Three different estimators of maximum magnitude are given by Kijko [2004], each depending on a different set of assumptions:

1. "Fixed b-value": Assumes a single b-value with no uncertainty

---

[2]The iterative nature of the Weichert [1980] algorithm can result in very slow convergence and unstable behaviour when the magnitudes infer b-values that are very small, or even negative. This can occur when very few events are in the resulting catalogue, or when the magnitudes converge within a narrow range.

2. "Uncertain b-value": Assumes and uncertain b-value defined by an expected b and the standard deviation

3. "Non-Parametric Gaussian": Assumes no functional form (can be applied to seismicity observed to follow a more characteristic distribution)

Each of these estimators assumes the general form:

$$m_{max} = m_{max}^{obs} + \Delta \tag{2.13}$$

where Δ is an increment that is dependent on the estimator used.

The uncertainty on m_{max} is also defined according to:

$$\sigma_{m_{max}} = \sqrt{\sigma_{m_{max}^{obs}}^2 + \Delta^2} \tag{2.14}$$

In the three estimators some lower bound magnitude constraint must be defined. For those estimators that assume an exponential recurrence model the lower bound magnitude must be specified by the users. For the non-Parametric Gaussian method and explicit lower bound magnitude does not have to be specified; however, the estimation is conditioned upon the largest N magnitudes, where N must be specified by the user.

If the user wishes to input a maximum magnitude that is larger than that observed in the catalogue (e.g. a known historical magnitude), this can be specified in the config file using `input_mmax` with the corresponding uncertainty defined by `input_mmax_uncertainty`. If these are not defined (i.e. set to `None`) then the maximum magnitude will be taken from the catalogue.

All three estimators require an iterative solution, therefore additional parameters can be specified in the configuration file that control the iteration process: `tolerance` difference in M_Max estimate for the algorithm to be considered converged, and `maximum_iterations` the maximum number of iterations for stability.

2.5.1.1. "Fixed b-value"

For a catalogue of n earthquakes, whose magnitudes are distributed by a Gutenberg and Richter [1944] distribiution with a fixed "b" value, the increment of maximum magnitude is determined via:

$$\Delta = \int_{m_{min}}^{m_{max}} \left[\frac{1 - \exp\left[-\beta\left(m - m_{min}\right)\right]}{1 - \exp\left[-\beta\left(m_{max}^{obs} - m_{min}\right)\right]} \right]^n dm \tag{2.15}$$

The execution of the Kijko [2004] "fixed-b" algorithm is as follows:

```
In [1]: mmax_config = {'input_mmax': 7.6,
                       'input_mmax_uncertainty': 0.22,
                       'b-value': 1.0,
                       'input_mmin': 5.0,
                       'tolerance': 1.0E-5,  # Defaults to this if not specified
                       'maximum_iterations': 1000} # Defaults to this if not specified

In [2]: from hmtk.seismicity.max_magnitude.kijko_sellevol_fixed_b import KijkoSellevolFixedb

In [3]: mmax_estimator = KijkoSellevolFixedb()

In [4]: mmax, mmax_uncertainty = mmax_estimator.get_mmax(catalogue, mmax_config)
```

2.5.1.2. "Uncertain b-value"

For a catalogue of n earthquakes, whose magnitudes are distributed by a Gutenberg and Richter [1944] distribiution with an uncertain "b" value, characterised by and expected term (b) and a corresponding undertainty (σ_b), the increment of maximum magnitude is determined via:

$$\Delta = (C_\beta)^n \int_{m_min}^{m_max} \left[1 - \left(\frac{p}{p + m - m_{min}} \right)^q \right]^n dm \tag{2.16}$$

where $\beta = b \ln{(10.0)}$, $p = \beta / (\sigma_\beta)^2$, $q = (\beta / \sigma_\beta)^2$ and C_β is a normalising coefficient determined via:

$$C_\beta = \frac{1}{1 - [p / (p + m_{max} - m_{min})]^q} \tag{2.17}$$

In both the fixed and uncertain "b" case a minimum magnitude will need to be input into the calculation. If this value is lower than the minimum magnitude observed in the catalogue the iterator may not stabilise to a satisfactory value, so it is recommended to use a minimum magnitude that is greater than the minimum found in the observed catalogue.

The execution of the "uncertain b-value" estimator is undertaken in a very similar to that of the fixed b-value, the only additional parameter being the `sigma-b` term:

```
In [1]: mmax_config = {'input_mmax': 7.6,
                       'input_mmax_uncertainty': 0.22,
                       'b-value': 1.0,
                       'sigma-b': 0.15
                       'input_mmin': 5.0,
```

```
                     'tolerance': 1.0E-5, # Defaults to this if not specified
                     'maximum_iterations': 1000} # Defaults to this if not specified

In [2]: from hmtk.seismicity.max-magnitude.kijko_sellevol_bayes import KijkoSellevolBayes

In [3]: mmax_estimator = KijkoSellevolBayes()

In [4]: mmax, mmax_uncertainty = mmax_estimator.get_mmax(catalogue, mmax_config)
```

2.5.1.3. Non-Parametric Gaussian

The non-parametric Gaussian estimator for maximum magnitude m_{max} is defined as:

$$\Delta = \int\limits_{m_{min}}^{m_{max}} \left[\frac{\sum_{i=1}^{n} \left[\Phi\left(\frac{m-m_i}{h}\right) - \Phi\left(\frac{m_{min}-m_i}{h}\right) \right]}{\sum_{i=1}^{n} \left[\Phi\left(\frac{m_{max}-m_i}{h}\right) - \Phi\left(\frac{m_{min}-m_i}{h}\right) \right]} \right]^n dm \tag{2.18}$$

where m_{min} and m_{max} are the minimum and maximum magnitudes from a set of n events, Φ is the standard normal cumulative distribution function. h a kernel smoothing factor:

$$h = 0.9 \times min\left(\sigma, IQR/1.34\right) \times n^{-1/5} \tag{2.19}$$

with σ the standard deviation of a set of n earthquakes with magnitude m_i where $i = 1, 2, ...n$, and IQR the inter-quartile range.

Therefore the uncertainty on m_{max} is conditioned primarily on the uncertainty of the largest observed magnitude. As in many catalogues the largest observed magnitude may be an earlier historical event, which will be associated with a large uncertainty, this estimator tends towards large uncertainties on m_{max}.

Due to the need to define some additional parameters the configuration file is slightly different. No b-value or minimum magnitude needs to be specified; however, the algorithm will consider only the largest `number_earthquakes` magnitudes (or all magnitudes if the number of observations is smaller). The algorithm also numerically approximates the integral of the Gaussian pdf, so `number_samples` is the number of samples of the distribution. The rest of the execution remains the same as for the exponential recurrence estimators of M_{max}:

```
In [1]: mmax_config = {'input_mmax': 7.6,
                       'input_mmax_uncertainty': 0.22,
                       'number_samples': 51, # Default value
                       'number_earthquakes': 100 # Default value
                       'tolerance': 1.0E-5, # Defaults to this if not specified
                       'maximum_iterations': 1000} # Defaults to this if not specified

In [2]: from hmtk.seismicity.max-magnitude.kijko_nonparametric_gausian import KijkoNonParametricGaussian
```

```
In [3]: mmax_estimator = KijkoNonParametricGaussian()

In [4]: mmax, mmax_uncertainty = mmax_estimator.get_mmax(catalogue, mmax_config)
```

2.5.2. Cumulative Moment[Makropoulos and Burton, 1983]

The cumulative moment release method is an adaptation of the cumulative strain energy release method for estimating m_{max} originally proposed by Makropoulos and Burton [1983]. Another method based on a pseudo-graphical formulation, an estimator of maximum magnitude can be derived from a plot of cumulative seismic moment release with time. The average slope of this plot indicates the mean moment release for the input catalogue in question. Two further straight lines are defined with gradients equal to that of the slope of mean cumulative moment release, both enveloping the cumulative plot. The vertical distance between these two lines indicates the total amount of moment that may be released in the region, if no earthquakes were to occur in the corresponding time (i.e. the distance between the upper and lower bounding lines on the time axis). This concept is illustrated in Figure 2.3.
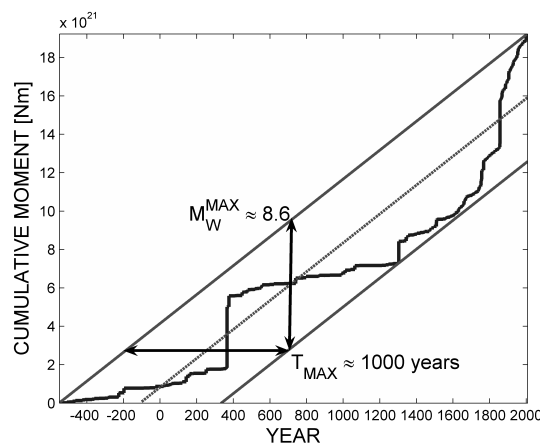

Figure 2.3 – *Illustration of Cumulative Moment Release Concept*

The cumulative moment estimator of m_{max}, whilst simple in concept, has several key advantages. As a non-parametric method it is independent of any assumed probability distribution and cannot estimate m_{max} lower than the observed m_{max}. It is also principally controlled by the largest events in the catalogue, this making it relative insensitive to uncertainties in completeness or lower bound threshold. In practice, this estimator, and to some extent that of Kijko [2004] are dependent on having a sufficiently long record of events relative to the strain cycle for the region in question, such that the estimate of average moment release is stable. This will obviously depend on the length of the catalogue, and for some regions, particularly those in low strain intraplate envi-

ronments, it is often the case that m_{max} will be close to the observed m_{max}. Therefore it may be the case that it is most appropriate to use these techniques on a larger scale, either considering multiple sources or an appropriate tectonic proxy.

For the cumulative moment estimator it is possible to take into account the uncertainty on m_{max} by applying bootstrap sampling to the observed magnitudes and their respective uncertainties. This has the advantage that $\sigma_{m_{max}}$ is not controlled by the uncertainty on the observed m_{max}, as it is for the Kijko [2004] algorithm. Instead it takes into account the uncertainty on all the magnitudes in the catalogue. The cost of this, however, is that this method is more computationally intensive, and therefore slower, than Kijko [2004], depending on the number of bootstrap samples the user chooses.

The algorithm is slightly simpler to run than the Kijko [2004] methods; however, due to the bootstrapping process it is slightly slower. It is run as per the following example:

```
In [1]: mmax_config = 'number_bootstraps': 1000

In [2]: from hmtk.seismicity.max_magnitude.cumulative_moment_release import CumulativeMoment

In [3]: mmax_estimator = CumulativeMoment()

In [4]: mmax, mmax_uncertainty = mmax_estimator.get_mmax(catalogue, mmax_config)
```

For the cumulative moment algorithm the only user configurable parameter is the `number_bootstraps`, which is the number of samples used during the bootstrapping process.

2.6. Smoothed Seismicity

The use of smoothed seismicity in seismic hazard analysis has generally become a common way of characterising distributed seismicity, for which the seismogenic source are defined exclusively from the uncertain locations of observed seismicity. There are many different methods for smoothing the catalogue, adopting different smoothing kernels or making different correction factors to compensate for spatial and/or temporal completeness.

2.6.1. Frankel [1995]

A smoothed seismicity method that has one of the clearest precedents for use in seismic hazard analysis is that of Frankel [1995], originally derived to characterise the seismicity of the Central and Eastern United States as part of the 1996 National Seismic Hazard Maps of the United States. The method applies a simple isotropic Gaussian smoothing

kernel to derive the expected rate of events at each cell $\tilde{n}_i$ from the observed rate n_j of seismicity in a grid of j cells. This kernel takes the form:

$$\tilde{n}_i = \frac{\sum_j n_j e^{d_{ij}^2/c^2}}{\sum_j e^{d_{ij}^2/c^2}} \tag{2.20}$$

In the implementation of the algorithm, two steps are taken that we prefer to make configurable options here. The first step is that the time-varying completeness is accounted for using a correction factor (t_f) based on the Weichert [1980] method:

$$t_f = \frac{\sum_i e^{-\beta m_{c_i}}}{\sum_i T_i e^{-\beta m_{c_i}}} \tag{2.21}$$

where m_{c_i} the completeness magnitude corresponding to the mid-point of each completeness interval, and T_i the duration of the completeness interval. The completeness magnitude bins must be evenly-spaced; hence, within the application of the progress a function is executed to render the input completeness table to one in which the magnitudes are evenly spaced with a width of 0.1 magnitude units.

2.6.2. Implementing the Smoothed Seismicity Analysis

The smoothed seismicity separates out the core implementation (i.e. the gridding, counting and execution of the code) and the choice of kernel. An example of the execution process is as follows:

The first stage is to upload the catalogue into an instance of the catalogue class

```
In [1]: input_file = 'path/to/input_file.csv'

In [2]: from hmtk.parsers.catalogue.csv_catalogue_parser import CsvCatalogueParser

In [3]: parser = CsvCatalogueParser(input_file)

In [4]: catalogue = parser.read_file()
```

Next setup the smoothing algorithm using and the corresponding kernel:

```
# Imports the smoothed seismicity algorithm
In [5]: from hmtk.seismicity.smoothing.smoothed_seismicity import SmoothedSeismicity

# Imports the Kernel function
In [6]: from hmtk.seismicity.smoothing.kernels.isotropic_gaussian import IsotropicGaussian

# Grid limits should be set up as
```

```
# [min_long, max_long, spc_long, min_lat max_lat, spc_lat, min_depth, max_depth,
# spc_depth]
In [7]: grid_limits = [0., 10., 0.1, 0., 10., 0.1, 0., 60., 30.]
# Assuming a b-value of 1.0
In [8]: smooth_seis = SmoothedSeismicity(grid_limits, use_3d=True, bvalue=1.0)
```

The smoothed seismicity function needs to be set up with three variables: i) the extent (and spacing) of the grid, ii) the choice to use 3D smoothing (i.e. distances are taken as hypocentral rather than epicentral) and iii) the input b-value. The extent of the grid can also be defined from the catalogue. If preferred the user need only specify the spacing of the longitude-latitude grid (as a single floating point value), then the grid will be defined by taking the bounding box of the earthquake catalogue and extended by the total smoothing length (i.e. the bandwidth (in km) multiplied by the maximum number of bandwidths).

To run the smoothed seismicity analysis, the configurable parameters are: `BandWidth` the bandwidth of the Gaussian kernel (in km), `Length_Limit` the number of bandwidths considered as a maximum smoothing length, and `increment` chooses whether to output the incremental a-value (for consistency with the original Frankel [1995] methodology) or the cumulative a-value (corresponding to the a-value of the Gutenberg-Richter model).

The algorithm requires two essential inputs (the earthquake catalogue and the config file), and three optional inputs:

- `completeness_table` A table of completeness magnitudes and their corresponding completeness years (as output from the completeness algorithms)

- `smoothing_kernel` An instance of the required smoothing kernel class (currently only Isotropic Gaussian is supported - and will be used if not specified)

- `end_year` The final year of the catalogue. This will be taken as the last year found in the catalogue, if not specified by the user

The analysis is then run via:

```
# Set up config (e.g. 50 km band width, up to 3 bandwidths)
In [9]: config = 'Length_Limit': 3., 'BandWidth': 50., 'increment': True

# Run the analysis!
In [10]: output_data = smooth_seis.run_analysis(catalogue, config, completeness_table, smoothing_kernel=IsotropicGa

# To write the resulting data to a csv file
In [11] = smooth_seis.write_to_csv('path/to/output_file.csv')
```

The resulting output will be a csv file with the following columns:

```
Longitude, Latitude, Depth, Observed Count, Smoothed Rate, b-value
```

where `Observed Count` is the observed number of earthquakes in each cell, and `Smoothed Rate` is the smoothed seismicity rate.

Geological Tools

The objectives of this chapter are:

- Describe features of the geological tools Hazard Modeller's Toolkit

- Run simple calculations using the geological tools

3.1. Fault Recurrence from Geology

DISCLAIMER: The model in this section has been constructed as a "real-world"-style example. The assumptions and parameters used in its creation are, however, merely used to illustrate the functionality of the tools and MUST NOT be adopted in subsequent models!

The second set of tools is designed to support a workflow in which the modeller has sufficient information to define both the geometry of the active fault surface and the slip rate, from which they then wish to calculate the activity rate on the fault according to a particular magnitude frequency distribution. It is recognised that in practice this is a complex and challenging process as the physical parameters of many faults may be highly uncertain, and the propagation of this uncertainty is critical in defining the epistemic uncertainty on such source models [Peruzza et al., 2010]. The current implementation of the tools focusses on the time-independent workflow entirely, aiming to allow the user to constrain the activity rate from the geological slip for an assumed single section. It is hoped that in future this will evolve to consider more complex conditions, such as those in which the observations of displacement at points along the segment and interactions between segments can be taken into consideration. The manner in which these features take shape will become clearer as more data is input into the global fault database created by the GEM Faulted Earth project.

The core of the time-independent workflow originates from the simple moment balance

in which the total moment release rate (in Nm) on the fault $\dot{M}_o$ is derived from the slip rate $\dot{s}$[Anderson and Luco, 1983; Bungum, 2007]:

$$\dot{M}_o = c\mu A\dot{s} \tag{3.1}$$

where A is the area of the fault surface (in km^2), μ is the shear modulus (characterised in the toolkit in terms of GigaPascals, GPa) and c the coefficient of seismogenic coupling. Slip rates must be input in $mm\ yr^{-1}$; lengths and area in km or km^2 respectively. The magnitude frequency distribution calculators differ primarily in the manner in which this moment rate is distributed in order to constrain the activity rate. The different calculators are described below.

3.1.1. Epistemic Uncertainties in the Fault Modelling

As previously mentioned, the manner in which epistemic uncertainties are incorporated into the fault recurrence calculation would appear to vary somewhat in practice. This is in no small part due to the manner in which the uncertainty on the contributing parameters is represented in a quantitative sense. For the present implementation, and driven in part by the need for consistency with the OpenQuake hazardlib, a purely decision based epistemic uncertainty analysis is supported. This requires that, for each parameter upon which epistemic uncertainty is supported, the user must specify the alternative values and the corresponding weightings. Currently, we support epistemic uncertainty on six different parts of the model:

1. Slip Rate ($mm\ yr^{-1}$)

2. Magnitude Scaling Relation

3. Shear Modulus (GPa)

4. Displacement to Length Ratio

5. Number of standard deviations on the Magnitude Scaling Relation

6. Configuration of the Magnitude Frequency Distribution

As input to the function, these epistemic uncertainties must be represented as a list of tuples, with a corresponding value and weight. For example, if one wished to characterise the slip on a fault by three values (e.g. 3.0, 5.0, 7.0) with corresponding weightings (e.g. 0.25, 0.5, 0.25), the slip should be characterised as shown below:

```
In [1]: slip = [(3.0, 0.25), (5.0, 0.5), (7.0, 0.25)]
```

In characterising uncertainty in this manner the user is essentially creating a logic tree for each source, in which the total number of activity rate models (i.e. the end branches

of the logic tree) is the product of the number of alternative values input for each supported parameter. The user can make a choice as to how they wish this uncertainty to be represented in the resulting hazard model:

3.1.1.1. Complete Enumeration

This will essentially reproduce the fault in the source model N times, where N is the total number of end branches, where on each branch the resulting activity rate is multiplied by the end weight the branch. For example, consider this simple two parameter case in which three values of slip and two magnitude scaling relations are considered (all other parameters are uniquely defined):

3.1.1.2. Collapsed Branches

In some cases it may become to costly to reproduce faults separately for each end branch and the user may simply wish to collapse the logic tree into a single activity rate. This rate, represented by an incremental magnitude frequency distribution, is the sum of the weighted activity rates on all the branches. To calculate this the program will determine the minimum and maximum magnitude of all the branches, then using a user specified bin width will calculate the weighted sum of the occurrence rates in each bin.

N.B. When collapsing the branches it the original magnitude scaling relations used on the branches and the scaling relation associated to the source in the resulting Open-Quake source model are not necessarily the same! The user will need to specify the scaling relation that will be assigned to the fault source when the branches are collapsed.

3.1.1.3. Magnitude Scaling Relations

To ensure compatibility with the OpenQuake engine, the scaling relations are taken directly from the OpenQuake library. Therefore the only scaling relations available are those that can be currently found in the oq-hazardlib (Wells and Coppersmith [1994] and Thomas et al. [2010] at the time of writing). To implement new magnitude scaling relations the reader is referred to the documentation and source code for the OpenQuake hazard library (http://docs.openquake.org/oq-hazardlib)

3.1.2. Tectonic Regionalisation

Recognising once again that certain parameters may not be possible to constrain on a fault by fault basis, a tectonic regionalisation can be invoked in order to define param-

eters, or a distribution of values, that can be assigned to all faults sharing that tectonic regionalisation classification. At present, the regionalisation can be used to define default parameters/distributions for:

1. Magnitude Scaling Relation

2. Shear Modulus

3. Displacement to Length Ratio

If defining a tectonic regionalisation the inputs must be specified as a set of tuples, in the same fashion as described for the epistemic uncertainties. In the present version there is no direct geographical link between the fault and the tectonic region (i.e. the regionalisation is a data holder and does not have geographical attributes), although it is anticipated that this may change in the future. At present it will be necessary to indicate for each fault the tectonic regionalisation class to which it belongs.

3.1.3. Definition of the Fault Input

3.1.3.1. The "YAML" Format

The establishment of a standard xml for representing input faults in OpenQuake remains to be undertaken, and will be done so following the completion of the GEM Faulted Earth project. The current version supports a less verbose, and more human-readable, characterisation using the "Yet Another Markup Language (YAML)" format. The Yaml format is both case and spacing sensitive, so care must be paid to the spacing and the punctuation characters below. For development, the primary advantage of the Yaml format is that the data are largely being defined in a manner that is consistent with the corresponding python objects, lists and dictionaries. This makes the reading of the file a simpler process.

A template example (for a single simple fault) is broken down into steps below.

The first component of the Yaml file is the tectonic regionalisation:

```
#*****************************************************************************
#FAULT FILE IN YAML (Yet Another Markup Language) FORMAT
#*****************************************************************************
#
tectonic_regionalisation:
    - Name: Active Shallow Crust
      Code: 001
      # Magnitude scaling relation (see http://docs.openquake.org/oq-hazardlib)
      #for currently available choices!
      Magnitude_Scaling_Relation: {
          Value: [WC1994],
          Weight: [1.0]}
      # Shear Modulus (in gigapascals, GPa)
      Shear_Modulus: {
```

```
        Value: [30.0],
        Weight: [1.0]}
    # Fault displacement to length ratio
    Displacement_Length_Ratio: {
        Value: [1.25E-5],
        Weight: [1.0]}
```

Here the tectonic regionalisation represents a list of categories (albeit only one is shown above). The lists elements are demarcated by the (-) symbol and all indentation is with respect to that symbol. So in the above example, a single region class has been defined with the name "Active Shallow Crust" and a unique identifier code "001". The default values are now provided for the three data attributes: magnitude scaling relation, shear modulus and displacement to length ratio. Each attribute is associated with a dictionary containing two keys: "Value" and "Weight", which then define the lists of the values and the corresponding weights, respectively.

N.B. If, for any of the above attributes, the number of weights is not the same as the number of values, or the weights do not sum to 1.0 then an error will be raised!

A fault model must be defined with both a model identifier key ("001") and a model name, "001" and "Template Simple Fault" in the example below. From then on the each fault is then defined as an element in the list.

```
Fault_Model_ID: 001
Fault_Model_Name: Template Simple Fault
Fault_Model:
    - ID: 001
      Tectonic_Region: Active Shallow Crust
      Fault_Name: A Simple Fault
      Fault_Geometry: {
          Fault_Typology: Simple,
          # For simple typology, defines the trace in terms of Long., Lat.
          Fault_Trace: [30.0, 30.0,
                        30.0, 31.5],

          # Upper Seismogenic Depth (km)
          Upper_Depth: 0.0,
          # Lower Seismogenic Depth (km)
          Lower_Depth: 20.0,
          Strike: ,
          # Dip (degrees)
          Dip: 60.}
      Rake: -90
      Slip_Type: Thrust
      Slip_Completeness_Factor: 1
      # slip [value_1, value_2, ... value_n]
      # [weight_1, weight_2, ... weight_n]
      Slip: {
          Value: [18., 20.0, 23.],
          Weight: [0.3, 0.5, 0.2]}
      #Aseismic Slip Factor
      Aseismic: 0.0
      MFD_Model:
          # Example of constructor for characteristic earthquake
        - Model_Type: Characteristic
          # Spacing (magnitude units) of the magnitude frequency distribution
```

```
      MFD_spacing: 0.1
      # Weight of the model
      Model_Weight: 0.2
      # Magnitude of the Characteristic Earthquake
      Maximum_Magnitude:
      # Uncertainty on Characteristic Magnitude (in magnitude units)
      Sigma: 0.12
      # Lower bound truncation (in number of standard deviations)
      Lower_Bound: -3.0
      # Upper bound truncation (in number of standard deviations)
      Upper_Bound: 3.0
      ##################################################
  - Model_Name: AndersonLucoArbitrary
      # Example constructor of the Anderson & Luco (1983) - Arbitrary Exponential
      # Type - chooses between type 1 ('First'), type 2 ('Second') or type 3 ('Third')
      Type: First
      MFD_spacing: 0.1
      Model_Weight: 0.1
      # Maximum Magnitude of the exponential distribution
      Maximum_Magnitude:
      Maximum_Magnitude_Uncertainty:
      Minimum_Magnitude: 4.5
      # b-value of the exponential distribution as [expected, uncertainty]
      b_value: [0.8, 0.05]
      ##################################################
  - Model_Name: AndersonLucoAreaMmax
      # Example constructor of the Anderson & Luco (1983) - Area-Mmax Exponential
      # Type - chooses between type 1 ('First'), type 2 ('Second') or type 3 ('Third')
      Model_Type: Second
      MFD_spacing: 0.1
      Model_Weight: 0.1
      Maximum_Magnitude:
      Maximum_Magnitude_Uncertainty:
      Minimum_Magnitude: 4.5
      b_value: [0.8, 0.05]
      ##################################################
  - Model_Name: YoungsCoppersmithExponential
      # Example constructor of the Youngs & Coppersmith (1985) Exponential model
      MFD_spacing: 0.1
      Model_Weight: 0.3
      Maximum_Magnitude:
      Maximum_Magnitude_Uncertainty:
      Minimum_Magnitude: 5.0
      b_value: [0.8, 0.05]
      ##################################################
      # Example constructor of the Youngs & Coppersmith (1985) Characteristic model
  - Model_Name: YoungsCoppersmithCharacteristic
      Model_Weight: 0.3
      Maximum_Magnitude:
      Maximum_Magnitude_Uncertainty:
      Minimum_Magnitude: 5.0
      MFD_spacing: 0.1
      b_value: [0.8, None]
Shear_Modulus: {
    Value: [30., 35.0],
    Weight: [0.8, 0.2]}
Magnitude_Scaling_Relation: {
    Value: [WC1994],
    Weight: [1.0]}
Scaling_Relation_Sigma: {
    Value: [-1.5, 0.0, 1.5],
    Weight: [0.15, 0.7, 0.15]}
```

```
Aspect_Ratio: 1.5
Displacement_Length_Ratio: {
    Value: [1.25E-5, 1.5E-5],
    Weight:[0.5, 0.5]}
```

The details of the magnitude frequency distribution configurations `MFD_Model` will be expanded upon in the respective sections. The critical attributes for a fault are then:

- `ID` The unique identifier for the fault

- `Name` The fault name

- `Tectonic Region` The tectonic region class to which the fault is assigned. Note that if the region class if not defined in the tectonic region header than an error will be raised.

- `Fault Geometry` A dictionary of the geometrical properties of the fault (described in further detail in due course

- `Rake` The rake of the fault (as defined using the Aki and Richards [2002] convention)

- `Slip` A dictionary with the slip rate of the fault in mm yr^{-1} and their corresponding uncertainties

- `Aseismic` A coefficient describing the pfraction of slip released aseismically (effectively $1 - c$ where c is the coupling coefficient)

- `MFD_Model` A list of models describing the corresponding magnitude frequency distribution properties

- `Aspect_Ratio` The rupture aspect ratio

- `Scaling_Relation_Sigma` The number of standard deviations of the magnitude scaling relation and their corresponding weights.

For shear modulus, magnitude scaling relation and displacement to length ratio the fault input is the same as for the tectonic regionalisation. These attributes can be for a single fault, but if they are not provided for the fault they can be defined within the tectonic regionalisation. If they are not defined within the tectonic regionalisation then the default values are assumed: 30 GPa for the shear modulus, Wells and Coppersmith [1994] for the scaling relation and 1.25×10^{-5} for the displacement to length ratio.

The remaining attributes are not essential and are not used in the calculation at present, but are included for completeness:

- `Slip_Type` Description of the type of slip (i.e. normal, thrust, etc.)

- `Slip_Completeness_Factor` The completeness factor (or quality factor) of the slip as an integer from 1 (Well constrained) to 4 (Unconstrained)

3.1.4. Fault Recurrence Models

The current implementation of the hmtk supports nine different models for deriving recurrence models from geological parameters. Six different models are provided for exponential distributions, which originate from the study of Anderson and Luco [1983], one for a "simple characteristic" earthquake (used here), a further exponential model (Youngs and Coppersmith [1985]) and the hybrid characteristic-exponential model also presented by Youngs and Coppersmith [1985]). Models describing exponential recurrence requiring the definition of a "b-value" for each source. The various nuances and assumptions behind each model are described in the sources cited here, and the reader is strongly encouraged to refer to the source material for further detail.

Common to many of these models is the moment magnitude definition of Hanks and Kanamori [1979]:

$$M_o\left(M_W\right) = 10.0^{16.05+1.5M_W} = M_o\left(M_{MAX}\right)e^{-\bar{d}\Delta M} \qquad (3.2)$$

where $\bar{d} = 1.5\log_e\left(10.0\right)$.

3.1.4.1. Anderson and Luco [1983] "Arbitrary"

The study of Anderson and Luco [1983] defines three categories of models for defining a magnitude recurrence distribution from the geological slip. The first model refers to the case when the recurrence is related to the entire fault, which we call the "Arbitrary" model here. The second model refers the recurrence to the rupture area of the maximum earthquake, the "Area Mmax" model here. The third category relates the recurrence to a specific site on the fault, and is not yet implemented in to tools. From within each of the three categories there are three different subcategories, which allow for different forms of tapering at the upper edge of the model. The reader is referred to the original paper of Anderson and Luco [1983] for further details and a complete derivation of the models. The different forms of the recurrence model are referred to here as types 1, 2 and 3, which correspond to equations 4, 3 and 2 in the original paper of Anderson and Luco [1983].

1. The 'first' type of Anderson and Luco [1983] arbitrary model is defined as:

$$N\left(M_W \geq M\right) = \frac{\bar{d}-\bar{b}}{\bar{d}}\frac{\mu A\dot{s}}{M_o\left(M_{MAX}\right)}\exp\left(\bar{b}\left[M_{MAX}-M\right]\right) \qquad (3.3)$$

   where $\bar{b} = b\log_e\left(10.0\right)$, $M_o\left(M_{MAX}\right)$ is the moment of the maximum magnitude, and A and $\dot{s}$ are the are of the fault and the slip rate, as defined previously.

2. The 'second' type of model is defined as:

$$N\left(M_W \geq M\right) = \frac{\bar{d} - \bar{b}}{\bar{b}} \frac{\mu A \dot{s}}{M_o\left(M_{MAX}\right)} \exp\left(-\bar{b}\left[M_{MAX} - M\right] - 1\right) \qquad (3.4)$$

3. The 'third' type of model is defined as:

$$\begin{aligned} N\left(M_W \geq M\right) =& \frac{\bar{d}\left(\bar{d} - \bar{b}\right)}{\bar{b}} \frac{\mu A \dot{s}}{M_o\left(M_{MAX}\right)} \times \ldots \\ & \left(\frac{1}{\bar{b}}\left(\exp\left(\bar{b}\left[M_{MAX} - M\right]\right) - 1\right) - \left[M_{MAX} - M\right]\right) \end{aligned} \qquad (3.5)$$

The configuration of the `MFD_Model` for the Anderson and Luco [1983] "Arbitrary" calculators shown as follows:

```
- Model_Name: AndersonLucoArbitrary
  # Example constructor of the Anderson & Luco (1983) - Arbitrary Exponential
  # Type - chooses between type 1 ('First'), type 2 ('Second') or type 3 ('Third')
  Type: First
  MFD_spacing: 0.1
  Model_Weight: 0.1
  # Maximum Magnitude of the exponential distribution
  Maximum_Magnitude:
  Maximum_Magnitude_Uncertainty:
  Minimum_Magnitude: 4.5
  # b-value of the exponential distribution as [expected, uncertainty]
  b_value: [0.8, 0.05]
```

The `Model_Name` and the `Type` are self explanatory. `Model_Weight` is the weighting of the particular MFD within the epistemic uncertainty analysis. `MFD_Spacing` is the spacing of the evenly discretized magnitude frequency distribution that will be output. The three parameters `Minimum_Magnitude`, `Maximum_Magnitude` and `Maximum_Magnitude_Uncertainty` define the bounding limits of the MFD and the standard deviation of M_{MAX} in M_W units. `Minimum_Magnitude` is an essential attribute, whereas `Maximum_Magnitude` and `Maximum_Magnitude_Uncertainty` are optional. If not specified the code will calculate `Maximum_Magnitude` and `Maximum_Magnitude_Uncertainty` from the magnitude scaling relationship. Finally, as these are exponential models the b-value must be specified. Here it is preferred that b-value is specified as a tuple of $[b - value, b - value\ error]$, although at present the epistemic uncertainty on b-value does not propagate (this may change in future!).

3.1.4.2. Anderson and Luco [1983] "Area Mmax"

The second set of models from Anderson and Luco [1983] consider the case when the the recurrence model is referred to the rupture area of the maximum earthquake specified on the fault. As the area is not extracted directly from the geometry, additional

information must be provided, namely the aspect ratio of ruptures on the fault and the displacement to length ratio (α) of the fault Bungum [2007]. This information is used to derived an additional parameter (β):

$$\beta = \sqrt{\frac{\alpha M_o(0)}{\mu W}} \qquad (3.6)$$

The three types of Anderson and Luco [1983] "Area Mmax" model are then calculated via:

1. Type 1 ("First")

$$N(M_W \geq M) = \frac{\bar{d} - \bar{b}}{\bar{d}} \frac{\dot{s}}{\beta} \exp\left(-\frac{\bar{d}}{2}M_{MAX}\right) \exp\left(\bar{b}[M_{MAX} - M]\right) \qquad (3.7)$$

2. Type 2 ("Second")

$$N(M_W \geq M) = \frac{\bar{d} - \bar{b}}{\bar{b}} \frac{\dot{s}}{\beta} \exp\left(-\frac{\bar{d}}{2}M_{MAX}\right) \exp\left(-\bar{b}[M_{MAX} - M] - 1\right) \qquad (3.8)$$

3. Type 3 ("Third")

$$\begin{aligned} N(M_W \geq M) = &\frac{\bar{d}(\bar{d} - \bar{b})}{\bar{b}} \frac{\dot{s}}{\beta} \exp\left(-\frac{\bar{d}}{2}M_{MAX}\right) \times \ldots \\ &\left(\frac{1}{\bar{b}}\left(\exp\left(\bar{b}[M_{MAX} - M]\right) - 1\right) - [M_{MAX} - M]\right) \end{aligned} \qquad (3.9)$$

As the rupture aspect ratio and displacement to length ratio are attributes of the fault and not of the MFD, then the MFD configuration is the same as that of the Anderson and Luco [1983] "Arbitrary" calculator, albeit that `Model_Name` must now be specified as `AndersonLucoAreaMmax`. As before, the maximum magnitude and their uncertainty are optional, and will taken from the magnitude scaling relation if not specified in the configuration. This is permitted simply to ensure flexibility of the algorithm, although given the context of the "Area MMax" algorithm it is understood that the maximum magnitude should be interpreted by the modeller. If this is not the case, and the maximum magnitude is intended to be constrained using the geometry of the rupture, the "Arbitrary" model may be preferable.

3.1.4.3. Characteristic

Although the term "Characteristic" may take on certain different meanings in the literature, in the present calculator it is referring to the circumstance when the fault is assumed to rupture with magnitudes distributed in a narrow range around the single characteristic magnitude. The model is therefore a truncated Gaussian distribution, in which the following must be specified: mean characteristic magnitude, the uncertainty (in magnitude units) and the number of standard deviations above and below the mean to be used as truncation limits.

```
                # Example of constructor for characteristic earthquake
    - Model_Type: Characteristic
        # Spacing (magnitude units) of the magnitude frequency distribution
        MFD_spacing: 0.1
        # Weight of the model
        Model_Weight: 0.2
        # Magnitude of the Characteristic Earthquake
        Maximum_Magnitude:
        # Uncertainty on Characteristic Magnitude (in magnitude units)
        Sigma: 0.12
        # Lower bound truncation (in number of standard deviations)
        Lower_Bound: -3.0
        # Upper bound truncation (in number of standard deviations)
        Upper_Bound: 3.0
```

The parameters `Model_Type`, `MFD_Spacing`, `Model_Weight`, and `Maximum_Magnitude` are as described for the previous calculators. `Sigma` is the uncertainty of the characteristic magnitude (in magnitude units), and `Lower_Bound` and `Upper_Bound` are the lower and upper truncation limits of the Gaussian distribution respectively. Note that setting `Sigma` to 0.0 or `Lower_Bound` and `Upper_Bound` to zero will simply result in the characteristic magnitude being evaluated as a single Dirac function.

3.1.4.4. Youngs and Coppersmith [1985] "Exponential"

This model is another form of "Exponential" model and is noted as being similar in construct to the Anderson and Luco [1983] Type 2 models. It is included here mostly for completeness. The model is given as:

$$N\left(M_W \geq M\right) = \frac{\mu A \dot{s}\left(1.5 - b\right)\left(1 - \exp\left(-\beta\left(M_{MAX} - M\right)\right)\right)}{b M_0^{MAX} \exp\left(-\beta\left(M_{MAX} - M\right)\right)} \tag{3.10}$$

where M_0^{MAX} is the moment corresponding to the maximum magnitude. The inputs for the model are defined in a similar manner as for the Anderson and Luco [1983] models:

```
    - Model_Name: YoungsCoppersmithExponential
        # Example constructor of the Youngs & Coppersmith (1985) Exponential model
        MFD_spacing: 0.1
```

```
Model_Weight: 0.3
Maximum_Magnitude:
Maximum_Magnitude_Uncertainty:
Minimum_Magnitude: 5.0
b_value: [0.8, 0.05]
```

Note that all of the exponential models described here contain the term $d - b$, or some variant thereof, where d is equal to 1.5. This introduces the condition that $b \leq 1.5$.

3.1.4.5. Youngs and Coppersmith [1985] "Characteristic"

The Youngs and Coppersmith [1985] model is a hybrid model, comprising an exponential distribution for lower magnitudes and a fixed recurrence rate for the characteristic magnitude M_C. The exponential component of the model is described via:

$$N(M) - N(M_C) = \frac{\mu A \dot{s} \, e^{(-\beta(M_{MAX} - M - 0.5))} M_0^{MAX}}{1 - e^{(-\beta(M_{MAX} - M - 0.5))}} \left[\frac{b10^{-c/2}}{c - b} + \frac{be^{\beta} \left(1 - 10^{-c/2}\right)}{c} \right]$$

(3.11)

where $\beta = b\ln(10)$, $c = 1.5$ and all other parameters are described as for the Youngs and Coppersmith [1985] "Exponential" and Anderson and Luco [1983] models. The rate for the characteristic magnitude is then given by:

$$N(M_C) = \frac{\beta \left(N(M) - N(M_C)\right) e^{-\beta(M_{MAX} - M - 1.5)}}{2 \left(1 - e^{-\beta(M_{MAX} - M - 1.5)}\right)}$$

(3.12)

As described in Youngs and Coppersmith [1985], this model assumes that:

1. The characteristic magnitude bin width ΔM_C is 0.5 M_W

2. Magnitudes are exponentially distributed up to a value M', where $M' = M_{MAX} - \Delta M_C$

3. The absolute rate of characteristic earthquakes $\dot{M}(M_C)$ is approximately equal to $\dot{M}(M' - 1)$

The current calculator adopts the implementation found in the OpenQuake hazardlib. At present, these three model assumptions are hard-coded, meaning that the distribution need only be defined from the moment rate and the characteristic magnitude. For [implementation] simplicity the input definition for the characteristic earthquake model is actually the same as for the exponential model. However, here the attribute `Maximum_Magnitude` is actually referring to the characteristic magnitude and not to the absolute maximum magnitude, which will be 0.25 larger.

3.1.5. Running a Recurrence Calculation from Geology

In this particular workflow is largely established in the input definition, and not in the configuration file. Execution of the whole process is then relatively simple, so it is broken down here into two steps. The first step simply imports an appropriate parser and loads the fault source. The parser contains a method called `read_file(x)` which takes as an input the desired mesh spacing (in km) used to create the mesh of the fault surface. In the example below this is set to 1.0 km, but for large complex faults (such as large subduction zones) it may be desirable to select a larger spacing to avoid storing a large mesh in RAM.

```
In [1]: from hmtk.parsers.faults.fault_yaml_parser import FaultYmltoSource

In [2]: input_file = 'path/to/fault_model_file.yml'

In [3]: parser = FaultYmltoSource(input_file)

# Spacing of the fault mesh (km) must be specified at input (here 1.0 km)
In [4]: fault_model, tect_reg = parser.read_file(1.0)
```

In the second step, we simply execute the method to calculate the recurrence on the fault, and the write the resulting source model to an xml file.

```
# Builds the fault model
In [5]: fault_model.build_fault_model()
# Specify the xml file for writing the output model
In [6]: output_file  = 'path/to/output_source_model_file.yml'

In [7]: fault_model.source_model.serialise_to_nrml(outfile, use_defaults=True)
```

The serialiser takes as an optional input the choice to accept default values for certain attributes that may be missing from the fault source definition. These values are as follows:

Attribute	Default Value
Aspect Ratio	1.0
Magnitude Scaling Relation	Wells and Coppersmith [1994] ('WC1994')
Nodal Plane Distribution	Strike = 0., Dip = 90, Rake = 0., probability= 1.0
Hypocentral Depth Distribution	Depth = 10.0 km, Probability = 1.0

3.1.5.1. Epistemic Uncertainties

The following example compares the case when epistemic uncertainties are incorporated into the analysis. A demonstration file (`tests/parsers/faults/yaml_examples/`

`simple_fault_example_4branch.yml`) is included, which considers two epistemic uncertainties for a specific fault: slip rate and magnitude frequency distribution type. The fault has two estimates of slip rate (5 $mm\ yr^{-1}$ and 7 $mm\ yr^{-1}$), each assigned a weighting of 0.5. Two magnitude frequency distribution types (Characteristic and Anderson and Luco [1983] "Arbitrary") are assigned, with weights of 0.7 and 0.3 respectively. The analysis is run, in the manner described previously. Firstly we consider the case when the different options are enumerated (the default option:

```
In [1]: from hmtk.parsers.faults.fault_yaml_parser import FaultYmltoSource
In [2]: input_file = 'tests/parsers/faults/yaml_examples/simple_fault_example_4branch.yml'
In [3]: parser = FaultYmltoSource(input_file)
In [4]: fault_model, tect_reg = parser.read_file(1.0)
In [5]: fault_model.build_fault_model()
In [6]: output_file  = 'path/to/output_source_model_file.yml'
In [7]: fault_model.source_model.serialise_to_nrml(output_file, use_defaults=True)
```

As four different activity rates are produced, the source is duplicated four time, each with an activity rate that corresponds to the rate calculated for the specific branch, multiplied by the weight of the branch. The output is a nrml file with four sources, as illustrated below:

```
<?xml version='1.0' encoding='UTF-8'?>
<nrml xmlns:gml="http://www.opengis.net/gml" xmlns="http://openquake.org/xmlns/nrml/0.4">
  <sourceModel name="Template Simple Fault">
    <simpleFaultSource id="1_1" name="A Simple Fault" tectonicRegion="Active Shallow Crust">
      <simpleFaultGeometry>
        <gml:LineString>
          <gml:posList>30.0 30.0 30.0 31.0</gml:posList>
        </gml:LineString>
        <dip>30.0</dip>
        <upperSeismoDepth>0.0</upperSeismoDepth>
        <lowerSeismoDepth>20.0</lowerSeismoDepth>
      </simpleFaultGeometry>
      <magScaleRel>WC1994</magScaleRel>
      <ruptAspectRatio>1.5</ruptAspectRatio>
      <incrementalMFD minMag="6.64" binWidth="0.1">
        <occurRates>1.66984888376e-05 0.000165760464747 0.000658012622916
                    0.00135343006814 0.00144508466725 0.00080109356265
                    0.000230216031359 3.05523435745e-05 0.0</occurRates>
      </incrementalMFD>
      <rake>-90.0</rake>
    </simpleFaultSource>
    <simpleFaultSource id="1_2" name="A Simple Fault" tectonicRegion="Active Shallow Crust">
      <simpleFaultGeometry>
        <gml:LineString>
          <gml:posList>30.0 30.0 30.0 31.0</gml:posList>
        </gml:LineString>
        <dip>30.0</dip>
        <upperSeismoDepth>0.0</upperSeismoDepth>
        <lowerSeismoDepth>20.0</lowerSeismoDepth>
      </simpleFaultGeometry>
      <magScaleRel>WC1994</magScaleRel>
      <ruptAspectRatio>1.5</ruptAspectRatio>
      <incrementalMFD minMag="4.5" binWidth="0.1">
        <occurRates>0.0404671423911 0.033659102961 0.0279964224108
                    0.0232864098818 0.0193687920987 0.0161102595577
```

```
                    0.0133999302432 0.0111455765116 0.00927048675038
                    0.00771085501945 0.00641360984941 0.00533460831472
                    0.00443713392921 0.00369064724985 0.00306974667434
                    0.00255330407018 0.00212374582219 0.00176645483392
                    0.00146927313415 0.00122208816284 0.00101648865894
                    0.000845478440245 0.000703238335844 0.000584928170206
                    0.000486522060675 0.000404671423911</occurRates>
      </incrementalMFD>
      <rake>-90.0</rake>
    </simpleFaultSource>
    <simpleFaultSource id="1_3" name="A Simple Fault" tectonicRegion="Active Shallow Crust">
      <simpleFaultGeometry>
        <gml:LineString>
          <gml:posList>30.0 30.0 30.0 31.0</gml:posList>
        </gml:LineString>
        <dip>30.0</dip>
        <upperSeismoDepth>0.0</upperSeismoDepth>
        <lowerSeismoDepth>20.0</lowerSeismoDepth>
      </simpleFaultGeometry>
      <magScaleRel>WC1994</magScaleRel>
      <ruptAspectRatio>1.5</ruptAspectRatio>
      <incrementalMFD minMag="6.64" binWidth="0.1">
        <occurRates>2.33778843726e-05 0.000232064650645 0.000921217672083
                    0.0018948020954 0.00202311853414 0.00112153098771
                    0.000322302443902 4.27732810043e-05 0.0</occurRates>
      </incrementalMFD>
      <rake>-90.0</rake>
    </simpleFaultSource>
    <simpleFaultSource id="1_4" name="A Simple Fault" tectonicRegion="Active Shallow Crust">
      <simpleFaultGeometry>
        <gml:LineString>
          <gml:posList>30.0 30.0 30.0 31.0</gml:posList>
        </gml:LineString>
        <dip>30.0</dip>
        <upperSeismoDepth>0.0</upperSeismoDepth>
        <lowerSeismoDepth>20.0</lowerSeismoDepth>
      </simpleFaultGeometry>
      <magScaleRel>WC1994</magScaleRel>
      <ruptAspectRatio>1.5</ruptAspectRatio>
      <incrementalMFD minMag="4.5" binWidth="0.1">
        <occurRates>0.0566539993476 0.0471227441454 0.0391949913751
                    0.0326009738345 0.0271163089382 0.0225543633808
                    0.0187599023404 0.0156038071162 0.0129786814505
                    0.0107951970272 0.00897905378917 0.00746845164061
                    0.00621198750089 0.00516690614979 0.00429764534408
                    0.00357462569825 0.00297324415106 0.00247303676749
                    0.00205698238781 0.00171092342797 0.00142308412252
                    0.00118366981634 0.000984533670182 0.000818899438288
                    0.000681130884944 0.000566539993476</occurRates>
      </incrementalMFD>
      <rake>-90.0</rake>
    </simpleFaultSource>
  </sourceModel>
</nrml>
```

If, alternatively, the user wishes to collapse the logic tree branches to give a single source as an output, this option can be selected as follows:

```
...
In [8]: from openquake.hazardlib.scalerel.wc1994 import WC1994
```

```
In [9]: fault_model.build_fault_model(collapse=True, rendered_msr=WC1994())
...
```

To collapse the branches it is simple necessary to specify as an input to the function `collapse True`. The second option requires further explanation. A seismogenic source requires the definition of a corresponding magnitude scaling relation, even if multiple scaling relations have been used as part of the epistemic uncertainty analysis. As collapsing the branches means that the activity rate is no longer associated with a specified scaling relation, but is an aggregate of many, the user must select the scaling relation to be associated with the output activity rate, for use in the OpenQuake hazard calculation. Therefore the input `rendered_msr` must be given an instance of one of the supported magnitude scaling relationships.

The output file should resemble the following:

```
<?xml version='1.0' encoding='UTF-8'?>
<nrml xmlns:gml="http://www.opengis.net/gml" xmlns="http://openquake.org/xmlns/nrml/0.4">
  <sourceModel name="Template Simple Fault">
    <simpleFaultSource id="1_1" name="A Simple Fault" tectonicRegion="Active Shallow Crust">
      <simpleFaultGeometry>
        <gml:LineString>
          <gml:posList>30.0 30.0 30.0 31.0</gml:posList>
        </gml:LineString>
        <dip>30.0</dip>
        <upperSeismoDepth>0.0</upperSeismoDepth>
        <lowerSeismoDepth>20.0</lowerSeismoDepth>
      </simpleFaultGeometry>
      <magScaleRel>WC1994</magScaleRel>
      <ruptAspectRatio>1.5</ruptAspectRatio>
      <incrementalMFD minMag="4.5" binWidth="0.1">
        <occurRates>0.0971211417387 0.0807818471064 0.0671914137858
                    0.0558873837162 0.0464851010369 0.0386646229385
                    0.0321598325836 0.0267493836278 0.0222491682009
                    0.0185060520467 0.0153926636386 0.0128030599553
                    0.0106491214301 0.00885755339963 0.00736739201842
                    0.00612792976843 0.00509698997324 0.00423949160142
                    0.00352625552195 0.00293301159081 0.00243957278146
                    0.00202914825659 0.00184661595792 0.00231362389313
                    0.00360190992617 0.00434969292261 0.00243432469089
                    0.000909841780938 0.000164472079868 0.0</occurRates>
      </incrementalMFD>
      <rake>-90.0</rake>
    </simpleFaultSource>
  </sourceModel>
</nrml>
```

Geodetic Tools

The objectives of this chapter are:

- Describe features of the geodetic tools Hazard Modeller's Toolkit

- Run simple calculations using the geodetic tools

4.1. Recurrence from Geodetic Strain

The final third workflow currently supported by the Hazard Modeller's toolkit is somewhat more experimental than the seismicity or geological workflows. The role that geodesy plays in directly constraining earthquake recurrence rates for active seismological structures and regions is becoming widely recognised. Whilst s standard methodology has not yet emerged, the tools constructed here are built around the "Seismic Hazard Inferred from Tectonics (SHIFT)" methodology originally proposed by Bird and Liu [2007] and applied on a global scale by Bird et al. [2010]. The focus is placed on this model, in part, because it is fed by regional and/or global scale models of strain on a continuum. The initial global application, which underpins part of the present implementation, uses the first version of the Global Strain Rate Model [Kreemer et al., 2003]. The second version of the Global Strain Rate Model has been produced as part of the Global Earthquake Model, and it is anticipated that this will form a basis for many future uses of the present geodetic tools.

4.1.1. The Recurrence Calculation

The point of convergence between the geological and geodetic methodologies for estimating earthquake recurrence is in the definition of the total moment rate for an active

seismogenic structure:

$$\dot{M}_o = c\mu A\dot{s} \tag{4.1}$$

where A is the area of the coupled surface, $\dot{s}$ is the slip rate, μ the shear modulus and c the coefficient describing the fraction of seismogenic coupling. Initially, the moment-rate tensor may be related to the strain rate tensor (ϵ_{ij}) using the formula of Kim and Sanderson [1974]:

$$\dot{M}_{oij} = 2\mu H A\varepsilon_{ij} \tag{4.2}$$

where H is the seismogenic thickness and A the area of the seismogenic source. Adopting the definition for a deforming continuum given by Bird and Liu [2007] the moment rate is equal to:

$$\dot{M}_o = A\langle cz\rangle\mu \begin{cases} 2\dot{\varepsilon}_3; & \text{if } \dot{\varepsilon}_2 < 0 \\ -2\dot{\varepsilon}_1; & \text{if } \dot{\varepsilon}_2 \geq 0 \end{cases} \tag{4.3}$$

where assuming $\dot{\varepsilon}_1 \leq \dot{\varepsilon}_2 \leq \dot{\varepsilon}_3$ and $\dot{\varepsilon}_1 + \dot{\varepsilon}_2 + \dot{\varepsilon}_3 = 0$ (i.e. no volumetric changes are observed). The coupled seismogenic thickness ($\langle cz\rangle$) is a characteristic of each tectonic zone, and for the current purposes corresponds to the values (and regionalisation) proposed by Bird and Kagan [2004].

4.1.1.1. The Bird and Liu [2007] Approach

As the regionalisation of Bird and Kagan [2004] underpins the Bird and Liu [2007] methodology, the following approach is used to derive the shallow sesimicity rate. The geodetic moment rate is first divided by the "model" moment rate ($\dot{M}_o^{CMT}$), which is the integral of the tapered Gutenberg-Richter distribution fit to the subset of the Global CMT catalogue for each tectonic zone, then multiplied by the number of events in the sub-catalogue (N^{CMT}) above the threshold (completeness) magnitude for the sub-catalogue (m_T):

$$\dot{N}\left(m > m_T^{CMT}\right) = \left(\frac{\dot{M}_o}{\dot{M}_o^{CMT}}\right)\dot{N}^{CMT} \tag{4.4}$$

The forecast rate of seismicity greater than m_T ($\dot{N}\left(m > m_T\right)$) for a particular zone (or cell) is described using the tapered Gutenberg-Richter distribution:

$$\begin{aligned}
\dot{N}\left(m > m_T\right) = &\dot{N}\left(m > m_T^{CMT}\right)\left(\frac{M_o\left(m_T\right)}{M_o\left(m_T^{CMT}\right)}\right)^{-\beta} \times \dots \\
&\exp\left(\frac{M_o\left(m_T^{CMT}\right) - M_o\left(m_T\right)}{M_o\left(m_c\right)}\right)
\end{aligned} \tag{4.5}$$

4.1.2. Running a Strain Calculation

4.1.2.1. Strain File Format

As the current process considers only continuum models, a strain model can be input as a simple comma separated value (.csv) file. This is a basic text file with the following headers:

```
longitude, latitude, exx, eyy, exy, region
45.0, -45., 0.0, 0.0, 0.0, IPL
.
.
.
177.0,-38.2,19.3,-37.0,-49.7,C
```

In the present format, the values `exx, eyy, exy` describe the horizontal components of the strain tensor (in the example above in terms of nanostrain, 10^{-9}). The `region` term corresponds to the region (in this instance the Kreemer et al. [2003] class) to which the cell belongs: Intra-plate [IPL], Subduction [S], Continental [C], Oceanic[O] and Ridge[R]. If the user does not have a previously defined regionalisation, can be determined using the $0.6° \times 0.5°$ global regionalisation cells.

The simplest strain workflow is to implement the model as defined by Bird et al. [2010]. This process is illustrated in the following steps:

1. To simply load in the csv data the `ReadStrainCsv` tool is used:

```
In [1]: from hmtk.parsers.strain.strain_csv_parser import ReadStrainCsv, WriteStrainCsv

# Load the file
In [2]: reader = ReadStrainCsv(''path/to/strain/file.csv'')

In [3]: strain_model = reader.read_data(scaling_factor=1E-9)
```

2. If the regionalisation is not supplied within the input file then it can be assigned initially from the in-built Kreemer et al. [2003] regionalisation. This is done as follows (and may take time to execute depending on the size of the data):

```
# (Optional) To assign regionalisation from Kreemer et al. 2003
In [4]: from hmtk.strain.regionalisation.kreemer_regionalisation import  KreemerRegionalisation
# (Optional)
In [5]: regionalisation = KreemerRegionalisation()
# (Optional)
In [6]: strain_model = regionalisation.get_regionalisation(strain_model)
```

3. The next step is to implement the Shift calculations. The Shift module must first be imported and the magnitudes for which the activity rates are to be calculated must be defined as a list (or array). The strain data is input into the calculation

along with two other configurable options: `cumulative` decides whether it is the cumulative rate of events above each magnitude (`True`), or the incremental activity rate for the bin $M(i) : M(i+1)$ (`False`), `in_seconds` decides whether to return the rates per second for consistency with Bird et al. [2010] (`True`) or as annual rates (`False`).

```
In [7]: from hmtk.strain.shift import Shift

# In this example, calculate cumulative rates for M > 5., 6., 7., 8.
In [8]: magnitudes = [5., 6., 7., 8.]

In [9]: model = Shift(magnitudes)

In [10]: model.calculate_activity_rate(strain_model, cumulative=False, in_seconds=False)
```

4. Finally the resulting model can be written to a csv file. This will be in the same format as the input file, now with additional attributes and activity rates calculated.

```
# Export the resulting rates to a csv file
In [11]: writer = WriteStrainCsv(''path/to/output/file.csv'')

In [12]: writer.write_file(model.strain, scaling_factor=1E-9)
```

Additional support for writing a continuum model to a nrml Point Source model is envisaged, although further work is needed to determine the optimum approach for defining the seismogenic coupling depths, hypocentral depths and focal mechanisms.

Bibliography

Aki, K. (1965). Maximum likelihood estimate of b in the formula log N = a - b M and its Confidence Limits. *Bulletin of the Earthquake Research Institute*, 43:237 – 239.

Aki, K. and Richards, P. G. (2002). *Quantitative Seismology*. University Science Books, Sausalito, California.

Anderson, J. G. and Luco, J. E. (1983). Consequences of slip rate constraints on earthquake occurrence relations. *Bull. Seism. Soc. Am.*, 73:471–496.

Bender, B. (1983). Maximum likelihood estimatation of b values for magnitude grouped data. *Bulletin of the Seismological Society of America*, 73(3):831 – 851.

Bird, P. and Kagan, Y. Y. (2004). Plate-Tectonic analysis of shallow seismicity: Apparent boundary width, beta, corner magnitude, coupled lithosphere thickness, and couping in seven tectonic settings. *Bulletin of the Seismological Society of America*, 94(6):2380 – 2399.

Bird, P., Kreemer, C., and Holt, W. . E. (2010). A long-term forecast of shallow seismicity based on the Global Strain Rate Map. *Seismological Research Letters*, 81(2):184 – 194.

Bird, P. and Liu, Z. (2007). Seismic Hazard Inferred from Tectonics: California. *Seismological Research Letters*, 78(1):37 – 48.

Bungum, H. (2007). Numerical modelling of fault activities. *Computers & Geosciences*, 33:808 – 820.

Crowley, H., Colombi, M., Crempien, J., Erduran, E., Lopez, M., Liu, H., Mayfield, M., and Milanesi, M. (2010). GEM1 Seismic Risk Report: Part 1. GEM Technical Report 2010-5, GEM Foundation, Pavia, Italy.

Felzer, K. R. (2008). The uniform California earthquake rupture forecast, version 2 (UCERF 2) appendix i: Calculating California seismicity rates. Technical report, U.S. Geological Survey Open File Report 2007-01347I.

Frankel, A. (1995). Mapping Seismic Hazard in the Central and Eastern United States. *Seismological Research Letters*, 66(4):8–21.

Bibliography

Frankel, A., Mueller, C., Barnhard, T., Perkins, D., Leyendecker, E. V., Dickman, N., Hanson, S., and Hopper, M. (1996). National seismic hazard maps: Documentation june 1996. Technical report, United States Geological Survey Open-File Report 96-532.

Frankel, A., Petersen, M. D., Mueller, C. S., Haller, K. M., Wheeler, R. L., Leyendecker, E. V., Wesson, R. L., Harmson, S. C., Cramer, C. H., Perkins, D. M., and Rukstales, K. S. (2002). Documentation for the 2002 updata to the national seismic hazard maps. Technical report, United States Geological Survey Open-File Report 02-420.

Gardner, J. K. and Knopoff, L. (1974). Is the sequence of earthquakes in Southern California, with aftershocks removed, poissonian? *Bulletin of the Seismological Society of America*, 64(5):1363 – 1367.

Gutenberg, B. and Richter, C. F. (1944). Frequency of earthquakes in California. *Bulletin of the Seismological Society of America*, 34:185 – 188.

Hanks, T. and Kanamori, H. (1979). A moment magnitude scale. *Journal of Geophysical Research*, 84:2348 – 2350.

Kijko, A. (2004). Estimation of the maximum earthquake magnitude, m_{MAX}. *Pure and Applied Geophysics*, 161:1655 – 1681.

Kijko, A. and Smit, A. (2012). Extension of the aki-utsu b-value estimator for incomplete catalogs. *Bulletin of the Seismological Society of America*, 3:1283 – 1287.

Kim, Y.-S. and Sanderson, D. J. (1974). Seismic moment and energy of earthquakes and seismic flow of rock. *Izv. Acad. Sci. USSR Phys. Solid Earth*, 1:13–21 (English Transl.).

Kreemer, C., Holt, W. . E., and Haines, A. J. (2003). An integrated global model of present-day plate motions and plate boundary deformation. *Geophysical Journal International*, 154:8 – 34.

Luen, B. and Stark, P. B. (2012). Poisson tests of declustered catalogues. *Geophys. J. Int.*, 189:691 –700.

Makropoulos, K. C. and Burton, P. W. (1983). Seismic risk of circum-pacific earthquakes i. strain energy release. *Pure and Applied Geophysics*, 121(2):247 – 266.

Mignan, A. and Wöessner, J. (2012). Theme IV - Understanding Seismicity Catalogs and their Problems. Technical Report doi: 10.5078/corssa-00180805, Community Online Resource for Statistical Seismicity Analysis.

Musson, R. M. W. (1999a). Probabilistic seismic hazard maps for the north balkan region. *Annali di Geofisica*, 42(2):1109 – 1124.

Musson, R. M. W. (1999b). Probabilistic seismic hazard maps for the north balkan region. *Annali di Geofisica*, 42(2):1109 – 1124.

Peruzza, L., Pace, B., and Cavallini, F. (2010). Error propagation in time-dependent probability of occurrence for characterisrtic earthquakes in italy. *Journal of Seismology*, 14:119 – 141.

Rydelek, P. A. and Sacks, I. S. (1989). Testing the completeness of earthquake catalogues and the hypothesis of self-similarity. *Nature*, 337(19):251 – 253.

Schorlemmer, D. and Woessner, J. (2008). Probability of detecting an earthquake. *Bulletin of the Seismological Society of America*, 98(5):2103 – 2117.

Stepp, J. C. (1971). *An investigation of earthquake risk in the Puget Sound area by the use of the type I distribution of largest extreme*. PhD thesis, Pennsylvania State University.

Thomas, P., Wong, I., and Abrahamson, N. A. (2010). Verification of probabilistic seismic hazard analysis computer programs. PEER Report 2010/106, Pacific Earthquake Engineering Research Center, College of Engineering, University of California, Berkeley.

Uhrhammer, R. (1986). Characteristics of Northern and Central California seismicity. *Earthquake Notes*, 57(1):21.

van Stiphout, T., Zhuang, J., and Marsan, D. (2012). Theme V -models and techniques for analysing seismicity. Technical report, Community Online Resource for Statistical Seismicity Analysis.

Weichert, D. H. (1980). Estimation of the earthquake recurrance parameters for unequal observation periods for different magnitudes. *Bulletin of the Seismological Society of America*, 70(4):1337 – 1346.

Wells, D. L. and Coppersmith, K. J. (1994). New empirical relationships among magnitude, rupture length, rupture width, rupture area, and surface displacement. *Bull. Seism. Soc. Am.*, 84(4):974–1002.

Woessner, J. and Wiemer, S. (2005). Assessing the quality of earthquake catalogues: estimating the magnitude of completeness and its uncertainty. *Bulletin of the Seismological Society of America*, 95(2):684 – 698.

Youngs, R. R. and Coppersmith, K. J. (1985). Implications of fault slip rates and earthquake recurrence models to probabilistic seismic hazard estimates. *Bull. Seism. Soc. Am.*, 75:939 – 964.