“OpenQuake: Calculate, share, explore”

Testing procedures adopted in the development of the risk component of the OpenQuake-engine

Authors
Anirudh Rao[1]

[1] GEM Model Facility
via Ferrata, 1
20133 Pavia
Italy

Email address (for all the authors):
<name.surname>@globalquakemodel.org

Citation
Please cite this document as:
Rao, Anirudh (2015). Testing procedures adopted in the development of the risk component of the OpenQuake-engine *Global Earthquake Model (GEM) Technical Report 2015-04.*
doi: 10.13117/GEM.OPENQUAKE.TR2015.03, 27 pages.

Third printing, March 2015

Contents

Part I

Introduction

1. Software Testing

The current document describes the testing procedures adopted in the development of the hazard component of the OpenQuake-engine (OQ-engine), the open source hazard and risk software developed by the Global Earthquake Model initiative.

Nowadays seismic hazard analysis serves different needs coming from a variety of users and applications.

These may encompass engineering design, assessment of earthquake risk to portfolios of assets within the insurance and reinsurance sectors, engineering seismological research, and effective mitigation via public policy in the form of urban zoning and building design code formulation.

Decisions based on seismic risk results may have impacts on population, properties and capitals, possibly with important repercussions on our day-to-day life. For these reasons, it is recommendable that the generation of hazard models and their calculation is based on well-recognized, state-of-the-art and tested techniques, requirements that must be reconciled with the need to regularly incorporate recent advances given the progress carried out within the scientific community.

The features described below contribute to fulfill these requirements:

- Software should have a modular and flexible structure capable of incorporating new features and - as a consequence - offer users the most recent and advanced techniques. In very general terms, modularity is the level to which a component of a system can be moved, replaced or reused. In software design, modularity means the separation of the software into smaller independent components that can be implemented, maintained and tested easily and efficiently.
- Software should have and extensive test coverage which captures possible errors and avoids regressions (i.e. unexpected behaviors introduced by new features). Software testing (Myers et al., 2012) is an important, complex and vast discipline which helps in developing methods and processes aimed at certifying the extent to which a computer code behaves according to the original design intent and user specifications.

The OQ-engine includes different levels of modularity. The first is the one separating the engine itself into a number of libraries (see Figure 1.1), each one containing well identified knowledge, objects and methods (e.g. the OQ-hazardlib includes objects and methods needed to compute probabilistic seismic hazard and the OQ-risklib contains methods to compute scenario

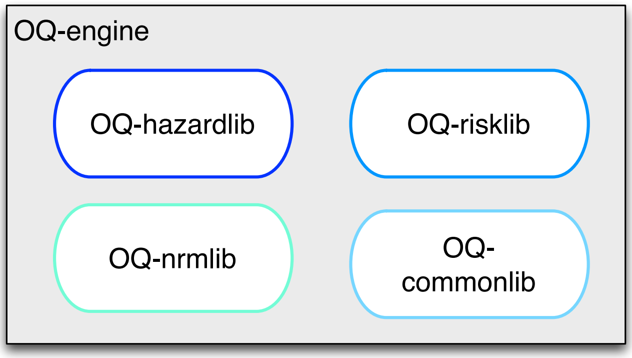

Figure 1.1 – *A schematic describing the main components of the OpenQuake-engine software.*

and probabilistic seismic risk).

The second one pertains to the data model adopted in the development of each library as a result of the abstraction process.

According to Berkes (2012) scientific software must be:

- Error proof
- Flexible and able to accommodate different methods
- Reproducible and re-usable

1.1 Testing and Quality Assurance

Despite the distinction between software testing (in some cases also called Quality Control) and Software Quality Assurance (SQA) being somewhat vague and partly open to personal judgment, it's clear that SQA is a more comprehensive and overarching process than software testing. SQA aims at the definition of the best processes that should be used to provide guarantees that user expectations will be met. Software testing focuses instead on detecting software faults by inspecting and testing the product at different stages of development.

1.1.1 Software testing

Software testing can be implemented at different stages of the development process, with varying strategies to approach the problem. The OQ-engine and the associated libraries are developed following an agile paradigm. This development strategy is organized in a way that the creation of the real code is completed in parallel and fully integrated with the software testing process.

The software engineering community provides a wide range of testing levels and typologies. In the current document we consider just a portion of them with the specific intent to illustrate the standards used in the development of the OQ-engine and particularly of its risk calculation component.

1.1.2 Quality assurance

From the IEEE "Standard for Software Quality Assurance Processes": *Software quality assurance is a set of activities that define and assess the adequacy of software processes to provide evidence that establishes confidence that the software processes are appropriate for and produce software products of suitable quality for their intended purposes. A key attribute of SQA is the objectivity of the SQA function with respect to the project. The SQA function may also be organizationally independent of the project; that is, free from technical, managerial, and financial pressures from* *the project.* In this document we are not covering topics related to SQA since this would go beyond its scope.

1.2 Organization of Report

This document is organized into eight chapters.

The current chapter provides a very brief and general introduction to software testing with a focus on the testing of scientific software.

The second chapter describes the module, or unit testing procedures adopted in the development of the OQ-engine and we discuss some examples. The continuous integration mechanism used for development is also discussed.

The third chapter describes the general framework for the acceptance tests for the OpenQuake risk calculators. A brief overview of the theoretical background for the different calculators is also provided in this chapter.

The fourth chapter describes the different test cases, input models, and results for the acceptance tests implemented for the OpenQuake scenario risk, classical risk, and event-based risk calculators.

In the fifth chapter, we compare the loss curves computed using the event-based calculator with the corresponding loss curves computed using the classical-PSHA based calculator.

In the sixth chapter, we illustrate tests comparing the results computed with the OQ-engine against the ones computed using different probabilistic seismic risk analysis software.

Chapter seven describes the OpenQuake risk demos and the average

The final chapter describes the set of

Part II

Unit Tests

2. Unit Testing in the OpenQuake-engine

This chapter provides an introduction to the module (unit) testing procedures (Myers et al., 2012) and describes the estensive series of tests implemented in the OQ-engine.

2.1 Overview of Unit-Testing

2.2 Continuous Integration

2.3 Unit-Tests in the OpenQuake Risk Library

2.4 Summary

Part III

Acceptance Tests

3. Framework for Acceptance Testing

3.1 Verification Framework

The main purpose of the acceptance tests is to ensure that the risk calculators work according to the design specifications and to verify that the calculators produce correct results for a variety of input cases. Correctness of the test case results are verified by comparing with hand calculations for the simple test cases or with alternate implementaions in Julia for the complex cases.

3.2 Theoretical Background

3.2.1 Scenario risk

3.2.2 Classical PSHA-based risk

3.2.3 Event-based risk

4. Test Cases and Results

4.1 Scenario Risk Calculator

The tests for the scenario risk calculator assume the correct computation of the ground motion fields at the locations of the assets in the exposure model. Thus, the risk tests implicitly rely on the acceptance tests for the scenario hazard calculator.

The rupture model used for the tests comprises a magnitude $M_W 6.7$ rupture on a vertical strike-slip fault.

Details of the rupture are given below:

Fault type: Strike slip
Fault dip: 90°
Fault plane depths: 0–20 km
Fault coordinates:
South end: $38.0000°N$, $122.0000°W$
North end: $38.2248°N$, $122.0000°W$
Rupture magnitude: 6.7
Rupture hypocenter: $38.1124°N$, $122.0000°W$
Hypocenter depth: 10 km

The complete collection of input models and job configuration files used in these test cases can be accessed here: https://github.com/gem/oq-risklib/tree/master/openquake/qa_tests_data/scenario_risk

4.1.1 Single asset tests

Site	Taxonomy	Latitude	Longitude	Comment
1	Wood	38.113	-122.000	On fault midpoint, along strike

Table 4.1 – *Asset location and taxonomy for the single-asset test cases*

The single asset test cases are designed to test the basic elements of the scenario risk calculator, such as:

- basic loss field computation
- calculation of mean and standard deviation of scenario loss

The location and taxonomy of the single asset in the exposure model used for the single-asset test cases for the scenario risk calculator are given in Table 4.1.

4.1.1.1 Case 1a

Test case 1a uses a set of 10,000 identical ground motion values. There is no uncertainty in the vulnerability function used for this case. The coefficient of variation of the loss ratio is zero at all intensity measure levels.

The purpose of this case is to test the accurate computation of the mean and standard deviation of the loss, given zero variability in both the ground motion values and in the vulnerability function.

GMF	Site	IMT	GMV
1	1	PGA	0.5000
2	1	PGA	0.5000
3	1	PGA	0.5000
4	1	PGA	0.5000
⋮	⋮	⋮	⋮
10000	1	PGA	0.5000

Table 4.2 – *Ground motion fields for the test cases 1a and 1b*

Table 4.2 lists five of the ten thousand ground motion values used in this test case.

PGA	0.05	0.20	0.40	0.60	0.80	1.00	1.20	1.40	1.60	1.80	2.00
Mean LR	0.01	0.04	0.10	0.20	0.33	0.50	0.67	0.80	0.90	0.96	0.99
CoV LR	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0

Table 4.3 – *Vulnerability function for scenario risk test case 1a*

Table 4.3 shows the mean loss ratios and corresponding coefficients of variation in the vulnerability function used in this test case.

Since all ground motion values are identical and there is no variability in the loss ratio, calculation of the loss ratios is straightforward in this case. The ground motion value at the location of the single asset is $PGA = 0.5g$. The vulnerability function for this case provides mean loss ratio values at intensity measure levels $0.4g$ and $0.6g$, but none at $0.5g$. The mean loss ratios at $0.4g$ and $0.6g$ are 0.10 and 0.20 respectively.

The mean loss ratio at $0.5g$ is obtained by interpolating between these two values. Linear interpolation gives a mean loss ratio of 0.15 for $PGA = 0.5g$. Since there is no variability in the ground motion, the mean loss ratio is also 0.15, and the standard deviation of the loss ratio is 0.0.

These numbers are multiplied by the asset value of $10,000$ to give the mean and standard deviation of loss for the scenario as $1,500$ and 0 respectively.

Table 4.4 shows the comparison of the OpenQuake result with the expected result.

Result	Expected	OpenQuake	Difference
Mean loss	1,500	1,500	0%
Std. loss	0	0	0%

Table 4.4 – *Results for scenario risk test case 1a*

4.1.1.2 Case 1b

Test case 1b uses the same set of identical ground motion values as case 1a, and described in Table 4.2. However, in contrast to case 1a, variability in the loss ratio *is* considered in the vulnerability function for this case.

PGA	0.05	0.20	0.40	0.60	0.80	1.00	1.20	1.40	1.60	1.80	2.00
Mean LR	0.01	0.04	0.10	0.20	0.33	0.50	0.67	0.80	0.90	0.96	0.99
CoV LR	0.03	0.12	0.24	0.32	0.38	0.40	0.38	0.32	0.24	0.12	0.03

Table 4.5 – *Vulnerability function for scenario risk test case 1b*

Table 4.5 shows the mean loss ratios and corresponding coefficients of variation in the vulnerability function used in this test case.

Similar to case 1a described above, linear interpolation gives a mean loss ratio of 0.15 for $PGA = 0.5g$. The vulnerability function for this case provides coefficients of variation for the loss ratio at intensity measure levels $0.4g$ and $0.6g$, but none at $0.5g$. The CoVs of the loss ratio at $0.4g$ and $0.6g$ are 0.24 and 0.32 respectively. The coefficient of variation of the loss ratio for $PGA = 0.5g$ is thus obtained by linear interpolation as 0.28.

The loss ratio at $PGA = 0.5g$ follows a lognormal distribution with a mean of 0.15 and a standard deviation of $0.28 \times 0.15 = 0.042$.

Since there is no variability in the ground motion, the mean loss ratio for the scenario is also 0.15, and the standard deviation of the loss ratio is 0.042.

These numbers are multiplied by the asset value of $10,000$ to give the mean and standard deviation of loss for the scenario as $1,500$ and 420 respectively.

Result	Expected	OpenQuake	Difference
Mean loss	1,500	1491.15	0.59%
Std. loss	420	413.36	1.58%

Table 4.6 – *Results for scenario risk test case 1b*

Table 4.6 shows the comparison of the OpenQuake result with the expected result.

4.1.1.3 Case 1c

Variability in the ground motion is considered in all cases starting from case 1c. Ten thousand ground motion fields are generated for the given rupture, taking into consideration both the inter-event and intra-event variability in the ground motion. The ground motion prediction equation used is Boore and Atkinson (2008).

The purpose of this case is to test the computation of the mean and standard deviation of the loss, given variability in both the ground motion values and no variability in the vulnerability function.

Table 4.7 lists five of the ten thousand ground motion values generated by OpenQuake.

Since the mean loss ratios in the vulnerability function are not a linear function of the intensity measure levels, an analytical solution for the mean and standard deviation of loss

GMF	Site	IMT	GMV
1	1	PGA	1.3495
2	1	PGA	0.5393
3	1	PGA	0.5240
4	1	PGA	1.0385
⋮	⋮	⋮	⋮
10000	1	PGA	0.1327

Table 4.7 – *Ground motion fields for the single asset tests*

for the scenario cannot be found. Thus, in order to check the OpenQuake results, an alternate implementation of the calculator algorithm in the programming language Julia is used.

The mean and standard deviation of the logarithm of the ground motion calculated at the location of the asset as obtained by using the Boore and Atkinson (2008) equation are -0.648 and 0.564 respectively. Assuming a lognormal distribution for the variability in the ground motion, one million ground motion values are generated using Julia with these logarithmic mean and standard deviation values.

The variability in the loss ratios is zero for the vulnerability function used in this test case. Thus, the loss ratio for each simulated ground motion value is obtained through interpolation on the mean loss ratios provided by the vulnerability function. The mean and standard deviation of loss ratio for the scenario are estimated simply as the mean and standard deviation of the million simulated loss ratios.

Result	Expected	OpenQuake	Difference
Mean loss	2,406.35	2,381.29	0.62%
Std. loss	2,181.39	2,169.72	0.53%

Table 4.8 – *Results for scenario risk test case 1c*

Table 4.8 shows the comparison of the OpenQuake result with the expected result.

4.1.1.4 Case 1d

The purpose of this case is to test the computation of the mean and standard deviation of the loss, given variability in both the ground motion values and in the vulnerability function.

Similar to case 1c, one million ground motion values are generated using Julia with the logarithmic mean and standard deviation of the ground motion value at the location of the asset as obtained by using the Boore and Atkinson (2008).

The mean loss ratio and standard deviation of loss ratio for each simulated ground motion value are obtained through interpolation on the mean loss ratios and corresponding coefficients of variation provided by the vulnerability function. Using the interpolated mean and standard deviation of loss ratios, one loss ratio is sampled for each ground motion value, assuming a lognormal distibution.

The mean and standard deviation of loss ratio for the scenario are estimated simply as the mean and standard deviation of the million simulated loss ratios.

Result	Expected	OpenQuake	Difference
Mean loss	2,383.1	2,370.7	0.52%
Std. loss	2,419.3	2,401.7	0.72%

Table 4.9 – *Results for scenario risk test case 1d*

Table 4.9 shows the comparison of the OpenQuake result with the expected result.

4.1.2 Multiple asset tests

Site	Taxonomy	Latitude	Longitude	Comment
1	Wood	38.113	-122.000	On fault midpoint, along strike
2	Wood	38.113	-122.114	10 km west of fault, at midpoint
3	RC	38.113	-122.570	50 km west of fault, at midpoint
4	RC	38.000	-122.000	South end of fault
5	Steel	37.910	-122.000	10 km south of fault, along strike
6	Wood	38.225	-122.000	North end of fault
7	Steel	38.113	-121.886	10 km east of fault, at midpoint

Table 4.10 – *Asset sites and taxonomies for the multiple-asset test cases*

The multiple asset test cases are designed to test the loss aggregation functions of the scenario risk calculator, such as:

- portfolio loss computation for a given ground motion field
- calculation of mean and standard deviation of portfolio scenario loss

The list of assets in the exposure model used for the multiple-asset test cases for the scenario risk calculator is given in Table 4.13.

Ten thousand ground motion fields are generated for the given rupture, taking into consideration both the inter-event and intra-event variability in the ground motion. The ground motion prediction equation used is Boore and Atkinson (2008), and the Jayaram and Baker (2009) model for spatial correlation of ground motion values is applied.

GMF	Site	IMT	GMV
1	1	PGA	1.125
1	2	PGA	0.261
1	3	PGA	0.073
1	4	PGA	0.756
1	5	PGA	0.121
1	6	PGA	0.671
1	7	PGA	0.274
2	1	PGA	0.386
2	2	PGA	0.198
2	3	PGA	0.058
2	4	PGA	0.660
2	5	PGA	0.396
2	6	PGA	0.529
2	7	PGA	0.297
⋮	⋮	⋮	⋮
10,000	1	PGA	0.265
10,000	2	PGA	0.106
10,000	3	PGA	0.114
10,000	4	PGA	0.213
10,000	5	PGA	0.293
10,000	6	PGA	0.639
10,000	7	PGA	0.093

Table 4.11 – *Ground motion fields for the multiple asset tests*

Table 4.11 lists three of the ten thousand ground motion fields generated.

4.1.2.1 Case 5a

The purpose of this case is to test the computation of the mean and standard deviation of the loss for multiple assets, given variability in both the ground motion values and in the vulnerability function. The computation of the mean and standard deviation of the portfolio loss is also tested in this case. The asset vulnerability correlation factor is zero for this test case.

Asset	Result	Expected	OpenQuake	Difference
a1	Mean loss	2,383.1	2,370.7	0.52%
	Std. loss	2,419.3	2,401.7	0.72%
a2	Mean loss			%
	Std. loss			%
a3	Mean loss			%
	Std. loss			%
a4	Mean loss			%
	Std. loss			%
a5	Mean loss			%
	Std. loss			%
a6	Mean loss			%
	Std. loss			%
a7	Mean loss			%
	Std. loss			%
Total	Mean loss			%
	Std. loss			%

Table 4.12 – *Results for scenario risk test case 5a*

Table 4.12 shows the comparison of the OpenQuake result with the expected result.

4.2 Classical Risk Calculator

The tests for the classical PSHA-based risk calculator assume the correct computation of the hazard curves at the locations of the assets in the exposure model. Thus, the risk tests implicitly rely on the acceptance tests for the classical PSHA-based hazard calculator.

The source model used for the tests comprises a single vertical strike-slip fault with a Gutenberg-Richter b-value equal to 0.9 and a slip rate of 2 mm/yr. The MFD is a Gutenberg-Richter distribution truncated between magnitudes 5.0 and 6.5, while the Ground Motion Prediction Equation (GMPE) used is Sadigh et al. (1997), with sigma set to zero.

Details of the fault geometry are given below:

Fault type: Strike slip
Fault dip: 90°
Fault plane depths: 0–12 km
Fault coordinates:
South end: 38.0000°N, 122.0000°W
North end: 38.2248°N, 122.0000°W

The complete collection of input models and job configuration files used in these test cases can be accessed here: https://github.com/gem/oq-risklib/tree/master/openquake/qa_tests_data/classical_risk

4.2.1 Single asset tests

4.2.2 Multiple asset tests

4.2.3 Calculation with logic-trees

4.3 Event-Based Risk Calculator

The tests for the event-based risk calculator assume the correct computation of the ground motion fields at the locations of the assets in the exposure model. Thus, the risk tests implicitly rely on the acceptance tests for the event-based hazard calculator.

The source model used for the tests comprises a single vertical strike-slip fault with a Gutenberg-Richter b-value equal to 0.9 and a slip rate of 2 mm/yr. The MFD is a Gutenberg-Richter distribution truncated between magnitudes 5.0 and 6.5, while the Ground Motion Prediction Equation (GMPE) used is Sadigh et al. (1997), with sigma set to zero.

Details of the fault geometry are given below:

Fault type: Strike slip
Fault dip: 90°
Fault plane depths: 0–12 km
Fault coordinates:
South end: 38.0000°N, 122.0000°W
North end: 38.2248°N, 122.0000°W

The list of assets in the exposure model used for the multiple-asset test cases for the event-based risk calculator is given in Table 4.13. The complete collection of input models and job configuration files used in these test cases can be accessed here: https://github.com/gem/oq-risklib/tree/master/openquake/qa_tests_data/event_based_risk

Site	Taxonomy	Latitude	Longitude	Comment
1	Wood	38.113	-122.000	On fault midpoint, along strike
2	Wood	38.113	-122.114	10 km west of fault, at midpoint
3	RC	38.113	-122.570	50 km west of fault, at midpoint
4	RC	38.000	-122.000	South end of fault
5	Steel	37.910	-122.000	10 km south of fault, along strike
6	Wood	38.225	-122.000	North end of fault
7	Steel	38.113	-121.886	10 km east of fault, at midpoint

Table 4.13 – *Asset sites and taxonomies for the multiple-asset test cases*

4.3.1 Single asset tests

4.3.2 Multiple asset tests

4.3.3 Calculation with logic-trees

Part IV

Benchmark Tests

5. Comparing the Classical and Event-Basec

6. Comparison with Other Softwares

Part V

Performance Tests

7. Demos

8. Stress Tests

Bibliography

Books

Myers, G. J., C. Sandler, and T. Badgett (2012). *The art of software testing*. Wiley and Sons, Inc. (cited on pages 9, 15).

Articles

Other Sources

Berkes, P. (2012). *Writing robust scientific code with testing (and Python)*. Euroscipy. URL: http://archive.euroscipy.org/talk/6634 (cited on page 10).