scxxlmag Documentation

Release 1.0.RC1

Javier Quinteros and Joachim Saul

September 30, 2014

CONTENTS

CHAPTER

ONE

FUNCTIONALITY

The *scxxlmag* is a SeisComP3 module developed by Javier Quinteros and Joachim Saul in the context of the NERA project. Its main purpose is to be able to calculate the m_{Bc} magnitude for large earthquakes in real-time. The present revision of the software includes the following functionality:

- Receive information about an event from a SeisComP3 server.
- Request all the waveforms needed to perform the calculation.
- Calculate the rupture duration while the information is being received.
- Calculate the magnitude while the information is being received.
- Send updates of the (preliminary) magnitude value and rupture duration at regular intervals
- Generate graphs showing the temporal evolution of the magnitude and rupture duration for a particular event

CHAPTER

TWO

SETUP

2.1 Installation

The latest version of *scxxlmag* can be downloaded from a GitHub respository under the official repository of SeisComP3.

2.1.1 Dependencies

- *Python* >= 2.6 (2.7 would be better)
- SeisComP3 release *Seattle* or newer (e.g. *Jakarta*)
- *seispy* tools from Joachim Saul (https://github.com/jsaul) is needed and should be in the Python path.

CHAPTER

THREE

RUNNING THE APPLICATION

3.1 Setting the environment

The SeisComP3 variables should be already in the session environment. You can load the necessary variables by executing:

```
user@hostname ~/scxxlmag $ ~/seiscomp3/bin/seiscomp print env
```

and then copy-paste the output of the previous command in the console. For instance,:

```
user@hostname ~/scxxlmag $ export SEISCOMP_ROOT=/home/user/seiscomp3
user@hostname ~/scxxlmag $ export PATH=/home/user/seiscomp3/bin:$PATH
user@hostname ~/scxxlmag $ export LD_LIBRARY_PATH=/home/user/seiscomp3/lib:LD_LIBRARY_PATH
user@hostname ~/scxxlmag $ export PYTHONPATH=/home/user/seiscomp3/lib/python:$PYTHONPATH
user@hostname ~/scxxlmag $ export MANPATH=/home/user/seiscomp3/share/man:$MANPATH
user@hostname ~/scxxlmag $ export LC_ALL=C
```

3.2 Calling the application

The application can be run by executing:

```
user@hostname ~/scxxlmag $ ./scxxlmag-compute.py
```

A file called `scxxlmag.sh` is provided as an example of how the application can be called with some of the most common options.

1. The SeisComP3 messaging system is contacted by the application.
2. All the information from the event is requested.
3. The full inventory is read and the streams are checked against the black list (see parameter `--blacklist`) and the operational time in relation to the event.
4. Inventory is filtered based on the location of the stations w.r.t. the event location. Only stations between 5° and 100° of radial distance to the event are used.
5. Timewindows are calculated for every stream based on the distance to the event origin. The timewindow is encompassed between the theoretical P arrival and the minimum expected S arrival. The theoretical S arrival is calculated based on the distance and the maximum expected S wave velocity. Two extra buffers are added before the theoretical P arrival (see below for more details).
6. Waveforms are requested to the recordStream specified.
7. For every record which is received, check that:

- the record is in chronological order. Otherwise, the stream and all the records coming from it are discarded.
- the record is inside the requested timewindow for this particular stream.
- there are no gaps between records. A gap is defined as a fix amount of time and can be changed to fit the needs of the user. Otherwise, the stream and all the records coming from it are discarded.

8. During the *first 20 seconds* (can be set by the user), the RMS of the signal is calculated. This will be used to center the signal around the 0 value. Another *extra 30 seconds* are included before the theoretical P arrival to have some error margin in case of an unexpected early arrival.
9. The system calculates the magnitude using the raw signal, but the rupture duration is calculated after filtering the signal with a *Butterworth filter around 2 Hz (1-3 Hz)*.
10. For every piece of the signal (raw and filtered) with the same sign the maximum value is detected and saved. However, even if we identify a maximum in the segment, there is an aditional condition that it should be fulfiled as a maximum (local or global). Namely, **the value must be greater than 60% of the global maximum** [1].
11. At the beginning of the programm a timer is defined, that at regular intervals will call a function to *update* the result of the magnitude calculation and the rupture duration. This values will be output and should be taken as a preliminary result (see *Examples of the real-time evolution of the results*).
12. In order for a stream to be taken into account for the preliminary results, one of the two conditions must be fulfiled:
    - the calculation is already complete in this stream.
    - the rupture duration is at least 90% of the current (preliminary) value. This needs to be done as a trade-off between early results and accuracy. We try to include all available data from the stations as soon as possible, even from the ones located far away from the event, but at the same time it is very important not to have a distorted value from stations that have only a few seconds of information, because due to this the duration would be initially very, very short. If the duration was overestimated from the first streams, the streams that are still being processed could take a little bit more of time to be included, but in the worst case, they will be there when the calculation of the stream is already done.
13. The list with rupture durations calculated from every stream is sorted and the one located at 3/4 of the list is selected as the *rupture duration of the event*. All the maximum values detected in the raw signal that lie inside the general rupture duration (the same for all streams) are used to calculate the m_{Bc} magnitude by means of the following formula:

$$m_{Bc} = Q(\Delta, z) + \log \frac{\frac{1}{2}\sum_i V_i}{2\pi}$$

being V_i the maximum values detected on the raw signal.

3.2.1 Main parameters

`-E [eventID]` specifies the event that we would like to work with.

`--inventory-db [invSource]` provides the location of the complete inventory that the application could use to select the most suitable streams. There are two main options that here one can use:

1. A SeisComP3 database: for instance, `mysql://yourUser:yourPassword@myservername/seiscomp3`
2. A file name: The inventory should have been download for instance from a SeisComP3 instance by means of the following command:

[1] Bormann, Peter and Saul, Joachim (2009) "A Fast, Non-saturating Magnitude Estimator for Great Earthquakes", Seismological Research Letters, 80: 808-816, doi:10.1785/gssrl.80.5.808

:: scxmldump -I -o inventory.xml -d [SC3db]

where `SC3db` has the same format as specified in point 1.

`-I [recordstream]` specifies the data source where the waveforms should be requested from. The source could have one of the following formats specified in the SeisComP3 documentation page. The *combined* and *file* mode can be seen as examples in `scxxlmag.sh`.

`-d [SC3db]` specifies the database connection string, format: `service://user:pwd@host/database`.

`--blacklist [filename]` provides a list of streams that should be avoided. For example, because the quality of those streams is known to be bad.

`--dump-waveforms` is a debug option to save the requested and processed waveforms in the disk with an ASCII format. Five types of files are saved, which can be seen in the following table.

Filename	Data type	Description
`mBc-NSLC.dat`	*Raw waveform*	The signal is saved exactly as it is received. Only the division by gain is applied.
`mBc-NSLC-f.dat`	*Filtered signal*	Signal which will be used to calculate the rupture duration.
`mBc-NSLC-p.dat`	*Recognized peaks for magnitude*	Peaks that will be included in the magnitude calculation.
`mBc-NSLC-p2.dat`	*Recognized peaks for rupture duration*	Peaks that will be included in the rupture duration estimation.
`mBc-NSLC-m.dat`	*Final magnitude*	The final magnitude calculated based on the information from **one stream**

In every case, the expresion `NSLC` in the format means that `N` is the network, `S` the station, `L` the location and `C` the channel.

3.2.2 Complete list of parameters

A complete list of the parameters available can be seen with:

```
user@hostname ~/scxxlmag $ ./scxxlmag-compute.py -h
```

Generic:

`-h [ --help ]` produce help message

`-V [ --version ]` show version information

`--config-file arg` Use alternative configuration file

`--plugins arg` Load given plugins

Verbose:

`--verbosity arg` verbosity level [0..4]

`-v [ --v ]` increase verbosity level (may be repeated, eg. -vv)

`-q [ --quiet ]` quiet mode: no logging output

`--print-component arg` print the log component (default: file:1, stdout:0)

`--print-context arg (=0)` print source file and line number

`--component arg` limits the logging to a certain component. this option can be given more than once

-s [--syslog] use syslog

-l [--lockfile] arg path to lock file

--console arg (=1) send log output to stdout

--debug debug mode: –verbosity=4 –console=1

--trace trace mode: `--verbosity=4 --console=1 --print-component=1 --print-context=1`

--log-file arg Use alternative log file

Database:

--db-driver-list list all supported database drivers

-d [--database] arg the database connection string, format: `service://user:pwd@host/database`

--config-module arg (=trunk) the configmodule to use

--inventory-db arg load the inventory from the given database or file, format: `[service://]location`

--config-db arg load the configuration from the given database or file, format: `[service://]location`

Records:

--record-driver-list list all supported record stream drivers

-I [--record-url] arg the recordstream source URL, format: `[service://]location[#type]`

--record-file arg specify a file as recordsource

--record-type arg specify a type for the records being read

Processing:

-E [--event] arg ID of event to process

Input:

-f [--format] arg input format to use (xml [default], zxml (zipped xml), binary)

-i [--input] arg input file, default: stdin

Control:

-b [--blacklist] arg stream blacklist

-w [--dump-waveforms] Save the requested waveforms in ASCII format

CHAPTER

FOUR

EXAMPLES OF THE REAL-TIME EVOLUTION OF THE RESULTS

CHAPTER

FIVE

CONTACTS

- Javier Quinteros <javier@gfz-potsdam.de>
- Joachim Saul <saul@gfz-potsdam.de>