ObsPy: A Python Toolbox for Seismology and Seismological Observatories

Lion Krischer[1], Tobias Megies[1], Robert Barsch[2], Moritz Beyreuther, Joachim Wassermann[1], and The ObsPy Development Team
[1] Department of Earth and Environmental Sciences, Ludwig-Maximilians-Universität München
[2] EGU Office
Contact: *devs@obspy.org* **http://www.obspy.org**

Summary

Python combines the power of a **full-blown programming language** with the flexibility and accessibility of an **interactive scripting language**. Its extensive standard library and large variety of freely available high quality scientific modules cover most needs in developing **scientific processing workflows**.
ObsPy extends Python's capabilities to fit the specific needs that arise when working with seismological data. It a) comes with a continuously growing **signal processing toolbox** that covers most tasks common in seismological analysis, b) provides **read and write support for many common waveform, station and event metadata formats** and c) enables **access to various data centers, webservices and databases** to retrieve waveform data and station/event metadata.
In combination with mature and free Python packages like **NumPy**, **SciPy**, **Matplotlib**, **IPython**, **Pandas**, **lxml**, and **PyQt**, ObsPy makes it possible to develop complete workflows in Python, ranging from reading locally stored data or requesting data from one or more different data centers via signal analysis and data processing to visualization in GUI and web applications, output of modified/derived data and the creation of publication-quality figures.
All functionality is **extensively documented** and the **ObsPy Tutorial** and Gallery give a good impression of the wide range of possible use cases. ObsPy is tested and **running on Linux, OS X and Windows** and comes with installation routines for these systems. ObsPy is developed in a test-driven approach and is available under the **LGPLv3 open source licence**.
Users are welcome to request help, report bugs, propose enhancements or contribute code via either the user mailing list or the **project page on GitHub**.

Why Python?

- ★ Easy to learn ⇒ Learning curve similar to Matlab
- ★ Free and Open Source
- ★ Cross-platform ⇒ Runs everywhere from Raspberry Pis to large supercomputers
- ★ General purpose language (in contrast to many other tools used in science)
- ★ No need to compile and interactive shell available
- ★ "Batteries included"
- ★ Mature third party libraries
- ★ Makes it easy to interact with existing C and Fortran code
- ★ One of the most used programming languages
- ★ Huge community outside of science ⇒ Tools and support widely available
- ★ Used a lot in the web community ⇒ Will become more and more important
- ★ Big support from the data analysis community
- ★ Interesting new developments: PyPy, Blaze, numba, IPython, pandas, …

What Can ObsPy Do?

- ★ **Read and write waveform data in various formats** (MiniSEED, SAC, GSE, SEG Y, …) with a unified interface.
- ★ **Database and webservice access clients** for NERIES, IRIS DMC, ArcLink, SeisHub and Earthworm (experimental).
- ★ **Many seismological signal processing routines** like filters, trigger, instrument correction, array analysis, beamforming, …
- ★ **Support for inventory data** (SEED, XSEED, RESP and planned StationXML support)
- ★ **Event data handling** (Supports the latest QuakeML 1.2 version)
- ★ **Unified time handling** Uses UTC ⇒ No ambiguities

\+

The full power and flexibility of Python.

Example

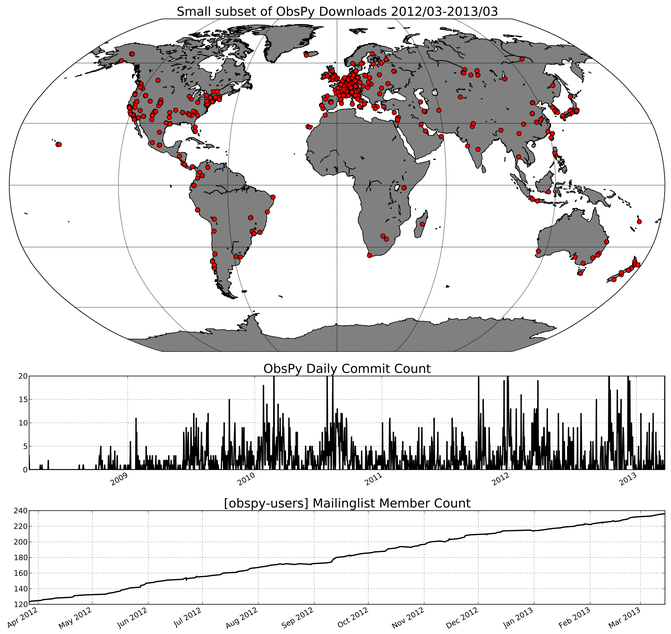
```
>>> from obspy import read
>>> st = read("waveform.mseed")
>>> print st
1 Trace(s) in Stream:
BW.FURT..EHZ | 2010-01-04 ... | 200.0 Hz, 7204234 samples
>>> st[0].data
array([-426, -400, ... , -489, -339], dtype=int32)
>>> st.trim(endtime=st[0].stats.starttime + 3600)
>>> st.decimate(factor=2)
>>> st.filter("lowpass", freq=1.0)
>>> print st BW.FURT..EHZ | 2010-01-05T ... | 100.0 Hz, 360001 samples
>>> st.plot()
```

What's next?

- ★ **Getting more developers and external contributions**
  ⇒ We strive to be as open as possible and very much encourage people to add to and modify our code basis
- ★ Support for Python 3 (partially done)
- ★ Support for StationXML (partially done)
- ★ More powerful instrument correction module
- ★ Suggestions? Let us know!

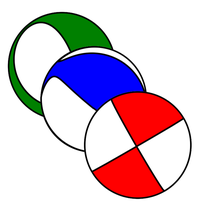
Impact and Future-Proofness

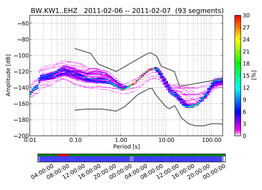
- ★ Around 25 people contributed code so far
- ★ Estimated active user base of **more than 5000** people
- ★ Code hosted on Github ⇒ Anyone can contribute
- ★ Goal: Built enough momentum for ObsPy to be self-sustainable

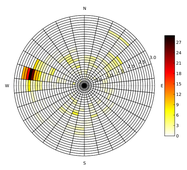


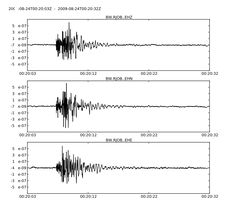
http://www.obspy.org

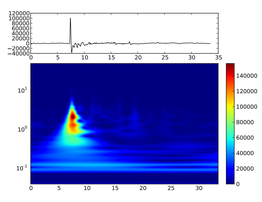
- ★ Source Code, Bug Tracker, and Wiki
- ★ Extensive documentation and example gallery
- ★ Tutorial on how to get started
- ★ **[obspy-users]** mailing list
- ★ Installation Instructions for various platforms
- ★ Use cases

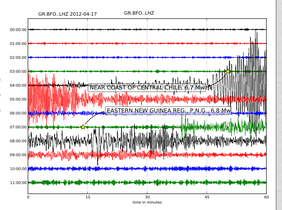












References

Beyreuther, M., R. Barsch, L. Krischer, T. Megies, Y. Behr and J. Wassermann (2010)
ObsPy: A Python Toolbox for Seismology
Seismological Research Letters, 81(3):530-533, doi:10.1785/ gssrl.81.3.530

Megies, T., M. Beyreuther, R. Barsch, L. Krischer and J. Wassermann (2011)
ObsPy – What can it do for data centers and observatories?
Annals Of Geophysics, 54(1), 47-58, doi:10.4401/ag-4838