ObsPy: A Python Toolbox for Seismology

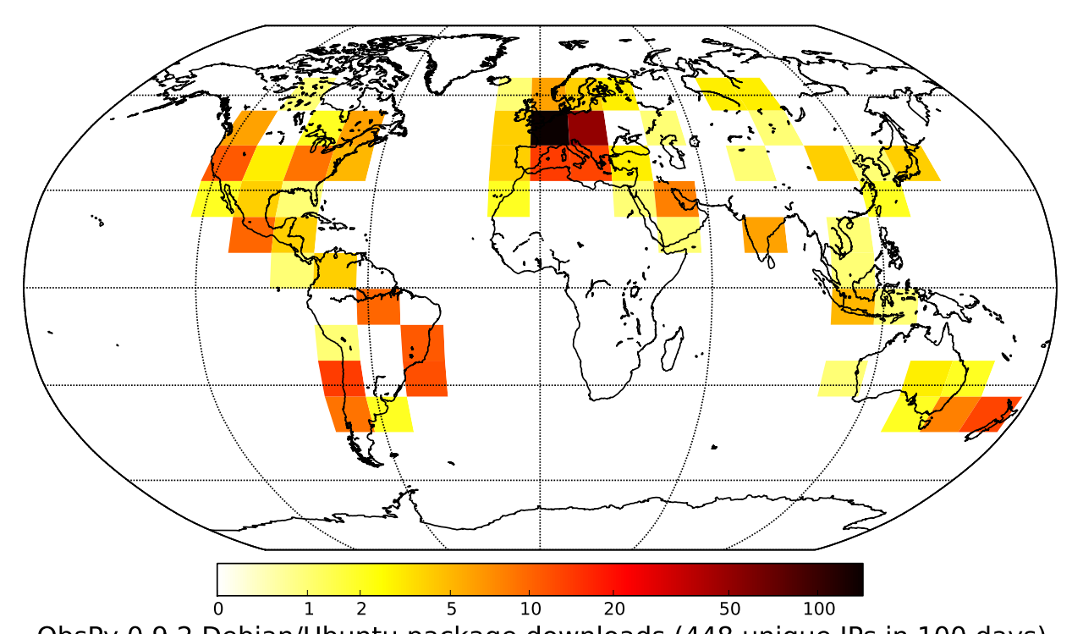
Tobias Megies, Lion Krischer, Robert Barsch, Moritz Beyreuther, Joachim Wassermann
Department of Earth and Environmental Sciences, Ludwig-Maximilians-University Munich, Germany
Contact: *devs@obspy.org* **http://www.obspy.org**

Summary

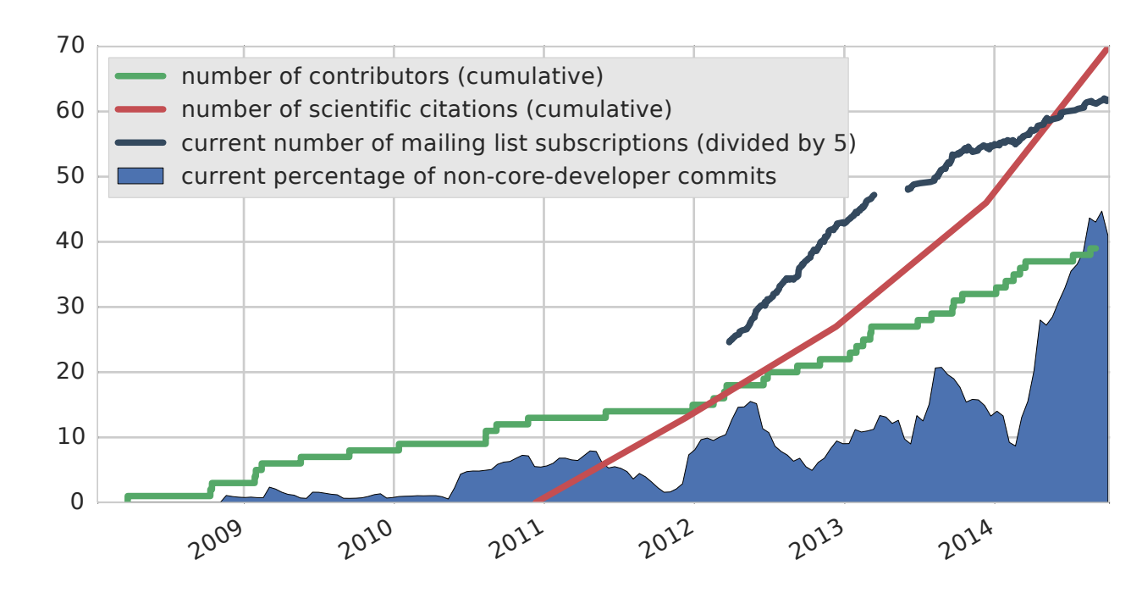
Python combines the power of a **full-blown programming language** with the flexibility and fast code development of an **interactive scripting language**. Its extensive standard library and large variety of freely available high quality scientific modules cover most needs in **developing scientific processing workflows**.
ObsPy extends Python's capabilities to fit the specific needs that arise when working with seismological data. It *a)* provides **read and write support for all of the most important waveform, station and event metadata formats** *b)* enables **direct access to all important data centers, web services and databases** to easily retrieve waveform data and station/event metadata and *c)* comes with a continuously growing **powerful signal processing toolbox** that covers all every-day tasks in seismological analysis.
In combination with mature and free Python packages like NumPy, SciPy, Matplotlib, IPython, Pandas and PyQt, **ObsPy makes it possible to develop complete seismological processing workflows**, 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 online **ObsPy Tutorial and Gallery** give a good impression of the wide range of possible use cases. ObsPy is tested and **running on Linux, MacOS 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**.

Impact

Six years after the beginnings of the project, ObsPy is used by seismologists all around the world. With **more than 500 downloads for Debian/Ubuntu Linux alone**, we estimate – including Mac, Windows and other Linux/Unix users – an **active user base of around 1000-2000 people**.
Since ObsPy's start by a **core developer team of 3-5 people** at LMU Munich, ObsPy has evolved into a community effort with **contributions to the code base by 40 individuals**. The **percentage of contributions from outside the core developer team is constantly on the rise and is currently at around 40%**.
The user mailing list currently has over **300 subscribers** and serves as a place for discussions and asking for help from more experienced users.
The **impact of ObsPy and the appreciation within the seismological community** finds expression in the **rapidly increasing number of scientific citations, which stands at over 70 as of October 2014**.

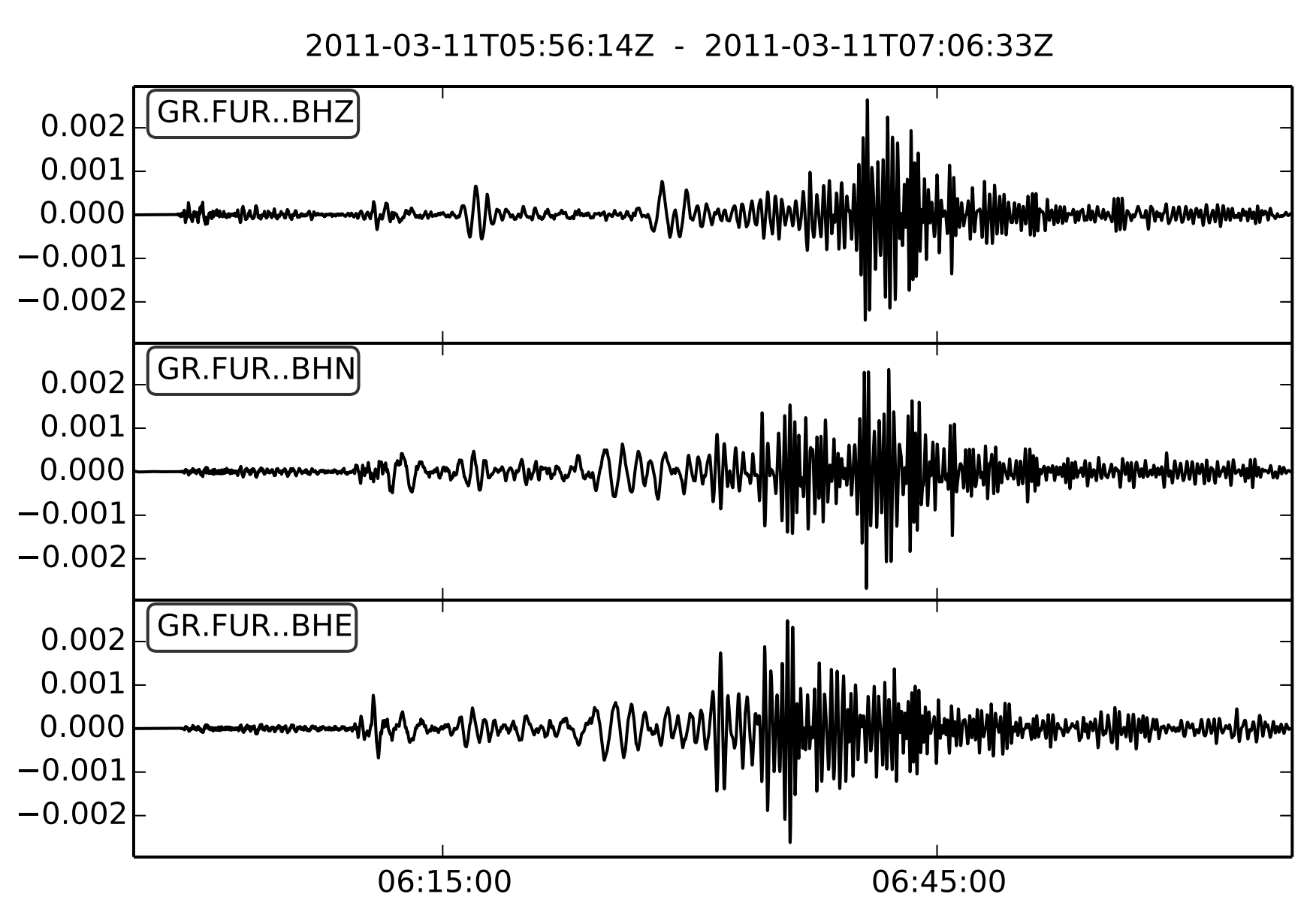


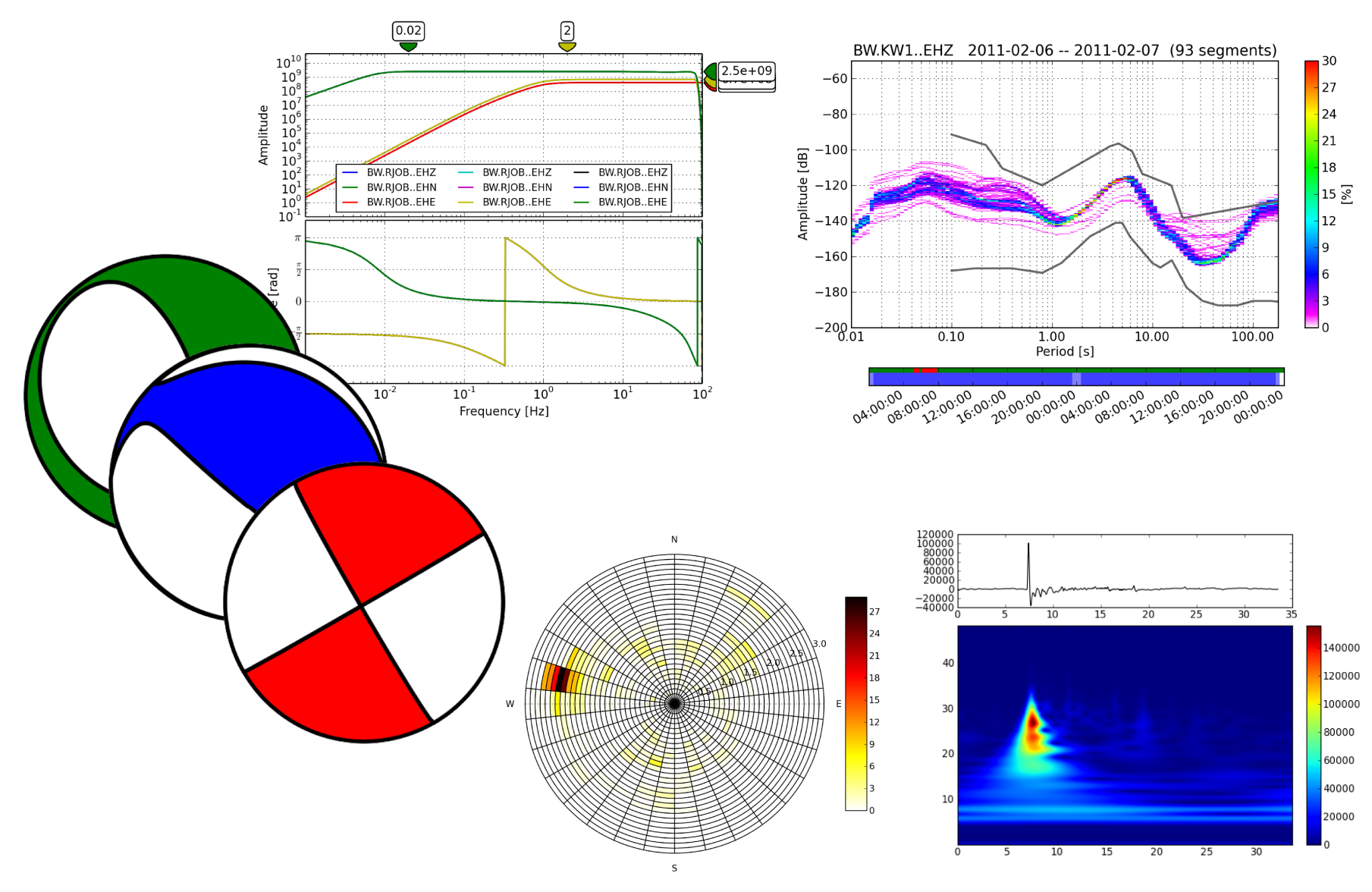
ObsPy 0.9.2 Debian/Ubuntu package downloads (448 unique IPs in 100 days)
This represents only a small subset of all downloads.



Basic Example

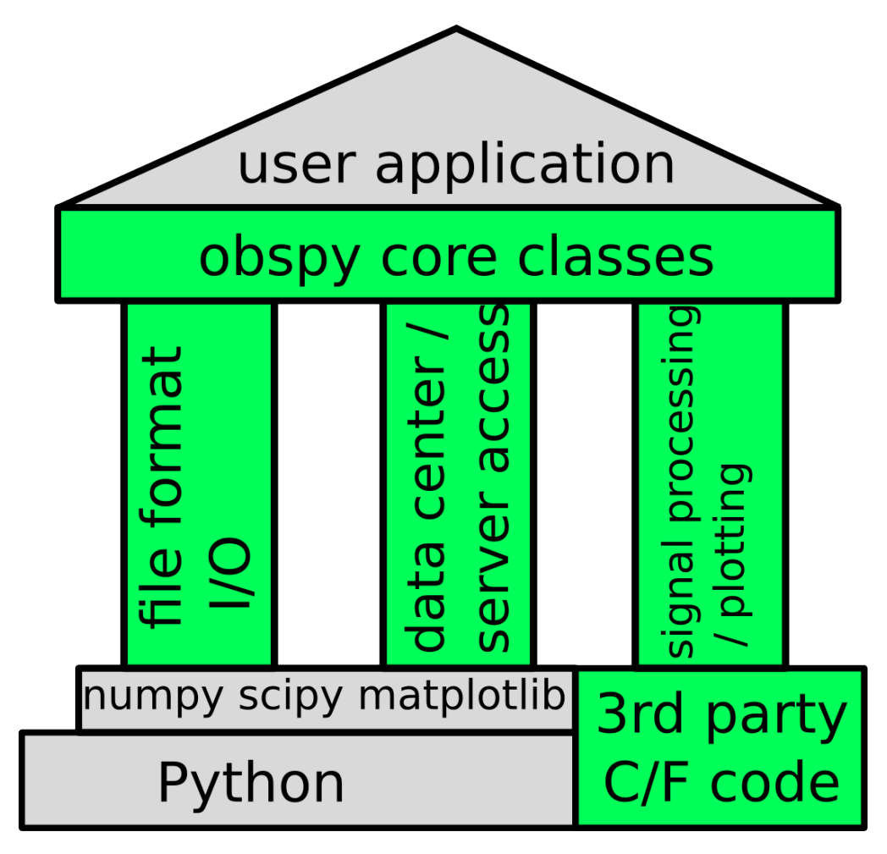
```
from obspy import UTCDateTime
from obspy.fdsn import Client

# connect to LMU observatory web service
client = Client("http://erde.geophysik.uni-muenchen.de:8080")

# use origin time of devastating Japan earthquake
start = UTCDateTime("2011-03-11 05:46:23") + 10 * 60
end = start + 70 * 60

# download waveform and station metadata of station FUR
stream = client.get_waveforms(
network="GR", station="FUR", location="", channel="BH*",
starttime=start, endtime=end, attach_response=True)

# do basic signal processing and plot the data! ---->
stream.remove_response()
stream.filter("bandpass", freqmin=0.01, freqmax=1)
stream.plot()
```

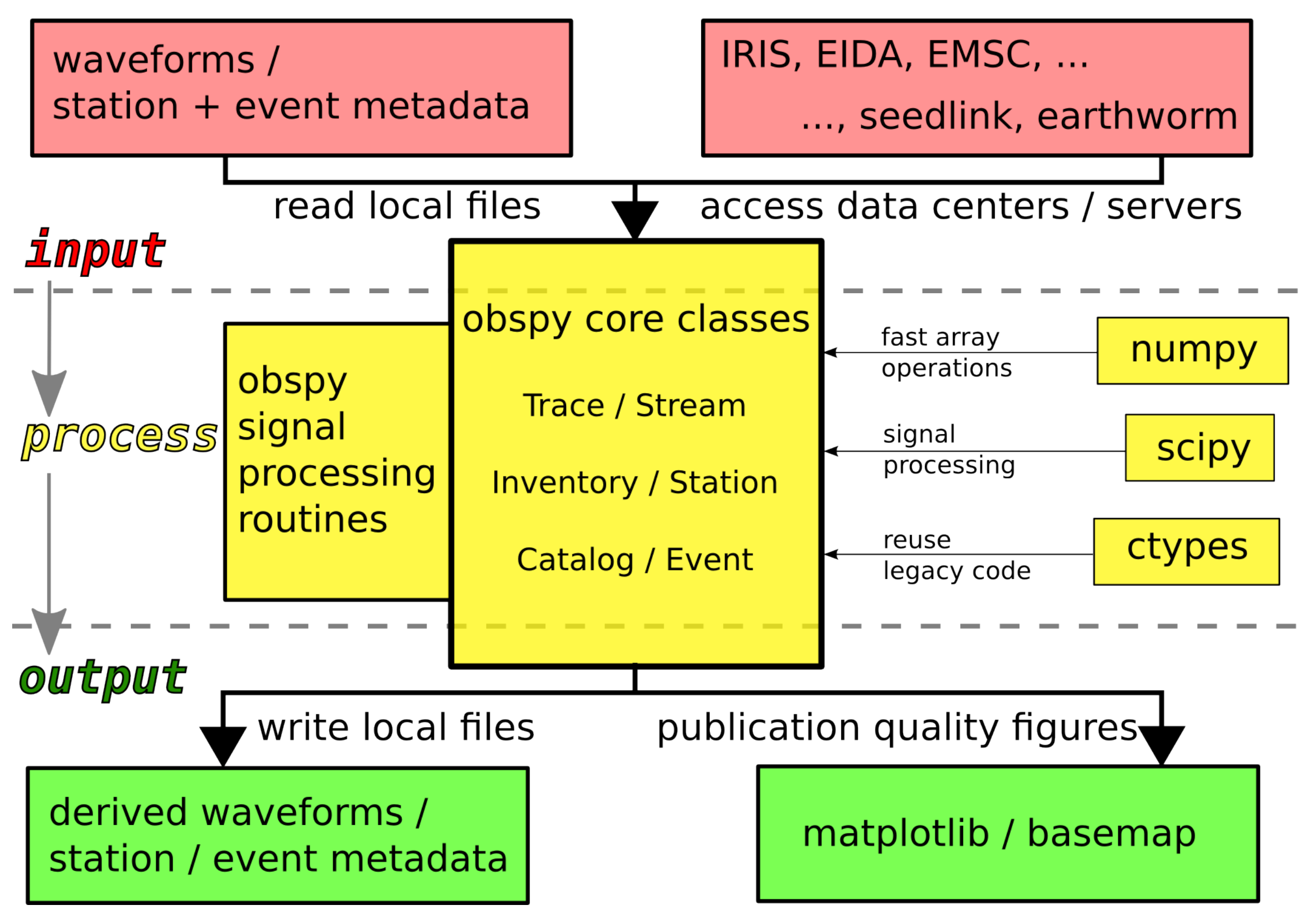




What Can I Do with ObsPy?

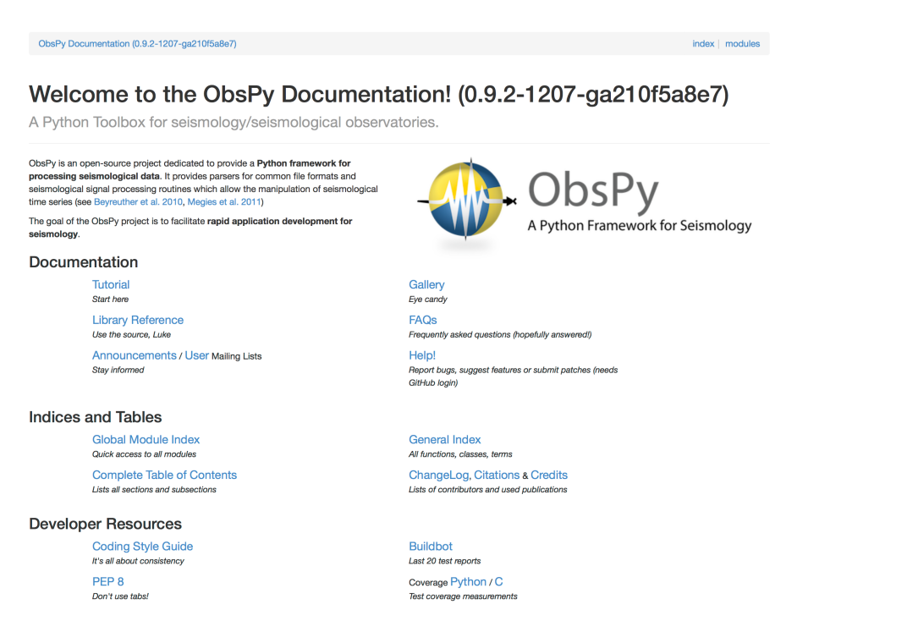
Easy to use helper functions to access local files and online data centers give **quick access to all data neccessary for seismological data analysis**.
All acquired information is exposed to the user in **ObsPy's core classes that handle waveform data, station and event metadata in a unified, consistent fashion, regardless of the data source**. This makes it easy to combine data from different sources in unified workflows, both interactively and automated.
ObsPy's core classes have many convenience routines for signal processing directly attached for **quick, reproducible and well tested execution of common processing tasks**.




waveforms / station + event metadata
IRIS, EIDA, EMSC, ..., seedlink, earthworm
read local files
access data centers / servers
input
process
output
obspy core classes
Trace / Stream
Inventory / Station
Catalog / Event
obspy signal processing routines
fast array operations
numpy
signal processing
scipy
reuse legacy code
ctypes
write local files
publication quality figures
derived waveforms / station / event metadata
matplotlib / basemap

ObsPy's Online Resources

- 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 and user application showcases


Welcome to the ObsPy Documentation! (0.9.2-1207-ga210f5a8e7)

Why Use Python?

- Easy to learn ⇒ Learning curve similar to Matlab
- Free and Open Source
- Cross-platform ⇒ Runs everywhere from RaspberryPi to large supercomputers
- General purpose language
  (in contrast to many other tools used in science)
- No need to compile and interactive shell available
- Mature third party libraries
- Easy to interact with legacy C and Fortran code
- One of the most used programming languages
- Huge community outside of science
  ⇒ Tools and support widely available
- Interesting new developments: PyPy, Blaze, numba, IPython, pandas, ...

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

Krischer, L., T. Megies, R. Barsch, M. Beyreuther, T. Lecocq, C. Caudron and J. Wassermann (2014)
ObsPy: A Bridge for Seismology into the Scientific Python Ecosystem
Computational Science & Discovery, accepted

The Scientific Python Ecosystem

Over the last decade, Python grew a vast and rich ecosystem of scientific third party libraries. **ObsPy serves as a bridge** and enables its users to effortlessly tie into this system giving access to large amounts of tools designed to process and analyze data. This box quickly introduces the core packages needed for scientific analysis.

IPython: Enhanced Interactive Console

- Powerful interactive shells (terminal and Qt-based)
- A browser-based notebook with support for code, text, mathematical expressions, inline plots and other rich media
- Easy to use, high performance tools for parallel computing

NumPy: *Modern Array Processing*

NumPy is the fundamental package needed for scientific computing with Python. Besides its obvious scientific uses, NumPy can also be used as an efficient multi-dimensional container of generic data. Arbitrary data-types can be defined. This allows NumPy to seamlessly and speedily integrate with a wide variety of databases.

SciPy: *Fundamental library for scientific computing*

SciPy is open-source software for mathematics, science, and engineering. It is also the name of a very popular conference on scientific programming with Python. The SciPy library depends on NumPy, which provides convenient and fast N-dimensional array manipulation. The SciPy library is built to work with NumPy arrays, and provides many user-friendly and efficient numerical routines such as routines for numerical integration and optimization.

Matplotlib: *Comprehensive, high quality 2D Plotting*

2D plotting library for Python that produces high quality figures that can be used in various hardcopy and interactive environments.

SymPy: *Symbolic mathematics*

SymPy is a Python library for symbolic mathematics. It aims to become a full-featured computer algebra system (CAS) while keeping the code as simple as possible in order to be comprehensible and easily extensible. SymPy is written entirely in Python and does not require any external libraries.

pandas: *Data structure & analysis*

pandas is a Python package providing fast, flexible, and expressive data structures designed to make working with "relational" or "labeled" data both easy and intuitive. It aims to be the fundamental high-level building block for doing practical, real world data analysis in Python. Additionally, it has the broader goal of becoming the most powerful and flexible open source data analysis / manipulation tool available in any language.

Mayavi: *3D Scientific Data Visualization and Plotting*

- A simple and clean scripting interface in Python, including one-liners, or an object-oriented programming interface. Mayavi integrates seamlessly with numpy and scipy for 3D plotting and can even be used in IPython interactively, similarly to Matplotlib.
- The power of the VTK toolkit, harnessed through these interfaces, without forcing you to learn it.