Installation

Installations methods include:

- Distributions
- pip
- Package Manager
- Source
- Binaries

Methods differ in ease of use, coverage, maintenance of old versions, system-wide versus local environment use, and control. With pip or Anaconda's conda, you can control the package versions for a specific project to prevent conflicts. Conda also controls non-Python packages, like MKL or HDF5. System package managers, like `apt-get`, install across the entire computer, often have older versions, and don't have as many available versions. Source compilation is much more difficult but is necessary for debugging and development. If you don't know which installation method you need or prefer, we recommend the Scientific Python Distribution Anaconda .

Scientific Python Distributions (recommended)

Python distributions provide the language itself, along with the most commonly used packages and tools. These downloadable files require little configuration, work on almost all setups, and provide all the commonly used scientific python tools.

Anaconda works on Windows, Mac, and Linux, provides over 1,500 Python/R packages, and is used by over 15 million people. Anaconda is best suited to beginning users; it provides a large collection of libraries all in one.

For more advanced users who will need to install or upgrade regularly, Miniconda is a more suitable way to install the *conda* package manager.

Other options include:

- WinPython: Another free distribution including scientific packages and the Spyder IDE; Windows only, but more actively maintained and supports the latest Python 3 versions.
- Pyzo: A free distribution based on Anaconda and the IEP interactive development environment; Supports Linux, Windows, and Mac.

Installing via pip

Python comes with an inbuilt package management system, pip. Pip can install, update, or delete any official package.

You can install packages via the command line by entering:

```
python -m pip install --user numpy scipy matplotlib ipython jupyter pandas sympy nose
```

We recommend using an *user* install, sending the `--user` flag to pip. `pip` installs packages for the local user and does not write to the system directories. Preferably, do not use `sudo pip`, as this combination can cause problems.

Pip accesses the Python Package Index, PyPI , which stores almost 200,000 projects and all previous releases of said projects. Because the repository keeps previous versions, you can pin to a version and not worry about updates causing conflicts. Pip can also install packages in local *virtualenv*, or virtual environment.

Install system-wide via a package manager

System package managers can install the most common Python packages. They install packages for the entire computer, often use older versions, and don't have as many available versions.

Ubuntu and Debian

using apt-get:

```
sudo apt-get install python-numpy python-scipy python-matplotlib ipython ipython-notebook python-pandas py
thon-sympy python-nose
```

Fedora 22 and later

using dnf:

```
sudo dnf install numpy scipy python-matplotlib ipython python-pandas sympy python-nose atlas-devel
```

Mac

Mac doesn't have a preinstalled package manager, but there are a couple of popular package managers you can install.

For Python 3.5 with Macports , execute this command in a terminal:

```
sudo port install py35-numpy py35-scipy py35-matplotlib py35-ipython +notebook py35-pandas py35-sympy py35
-nose
```

Homebrew has an incomplete coverage of the SciPy ecosystem, but does install these packages:

```
brew install numpy scipy ipython jupyter
```

Source packages

You can build any of the packages from source. Those involved in development may take this route to get developmental versions or alter source code. Refer to individual projects for more details.

Binaries

Binary files can directly install the packages. These can either come from the direct source, like GitHub or PyPI , or third-party repositories. Linux operating systems, like Ubuntu , have package repositories where you can search for and download individual binaries. For Windows, Christoph Gohlke provides pre-built Windows installers for many packages.