Building and installing SciPy

+++++++++++++++++++++++++++++

See https://www.scipy.org/install.html

.. Contents::

INTRODUCTION

============

It is \*strongly\* recommended that you use either a complete scientific Python

distribution or binary packages on your platform if they are available, in

particular on Windows and Mac OS X. You should not attempt to build SciPy if

you are not familiar with compiling software from sources.

Recommended distributions are:

- Enthought Canopy (https://www.enthought.com/products/canopy/)

- Anaconda (https://www.anaconda.com)

- Python(x,y) (https://python-xy.github.io/)

- WinPython (https://winpython.github.io/)

The rest of this install documentation summarizes how to build Scipy. Note

that more extensive (and possibly more up-to-date) build instructions are

maintained at https://scipy.github.io/devdocs/building/

PREREQUISITES

=============

SciPy requires the following software installed for your platform:

1) Python\_\_ >= 3.5

\_\_ https://www.python.org

2) NumPy\_\_ >= 1.13.3

\_\_ https://www.numpy.org/

If building from source, SciPy also requires:

3) setuptools\_\_

\_\_ https://github.com/pypa/setuptools

4) pybind11\_\_ >= 2.4.0

\_\_ https://github.com/pybind/pybind11

5) If you want to build the documentation: Sphinx\_\_ >= 1.2.1

\_\_ http://www.sphinx-doc.org/

6) If you want to build SciPy master or other unreleased version from source

(Cython-generated C sources are included in official releases):

Cython\_\_ >= 0.28.5

\_\_ http://cython.org/

Windows

-------

Compilers

~~~~~~~~~

There are two ways to build SciPy on Windows:

1. Use Intel MKL, and Intel compilers or ifort + MSVC. This is what Anaconda

and Enthought Canopy use.

2. Use MSVC + GFortran with OpenBLAS. This is how the SciPy Windows wheels are

built.

Mac OS X

--------

It is recommended to use GCC or Clang, both work fine. Gcc is available for

free when installing Xcode, the developer toolsuite on Mac OS X. You also

need a Fortran compiler, which is not included with Xcode: you should use a

recent GFortran from an OS X package manager (like Homebrew).

Please do NOT use GFortran from `hpc.sourceforge.net <http://hpc.sourceforge.net>`\_,

it is known to generate buggy SciPy binaries.

You should also use a BLAS/LAPACK library from an OS X package manager.

ATLAS, OpenBLAS, and MKL all work.

As of SciPy version 1.2.0, we do not support compiling against the system

Accelerate library for BLAS and LAPACK. It does not support a sufficiently

recent LAPACK interface.

Linux

-----

Most common distributions include all the dependencies. You will need to

install a BLAS/LAPACK (all of ATLAS, OpenBLAS, MKL work fine) including

development headers, as well as development headers for Python itself. Those

are typically packaged as python-dev.

INSTALLING SCIPY

================

For the latest information, see the website:

https://www.scipy.org

Development version from Git

----------------------------

Use the command::

git clone https://github.com/scipy/scipy.git

cd scipy

git clean -xdf

python setup.py install --user

Documentation

-------------

Type::

cd scipy/doc

make html

From tarballs

-------------

Unpack ``SciPy-<version>.tar.gz``, change to the ``SciPy-<version>/``

directory, and run::

pip install . -v --user

This may take several minutes to half an hour depending on the speed of your

computer.

TESTING

=======

To test SciPy after installation (highly recommended), execute in Python::

>>> import scipy

>>> scipy.test()

To run the full test suite use::

>>> scipy.test('full')

If you are upgrading from an older SciPy release, please test your code for any

deprecation warnings before and after upgrading to avoid surprises:

$ python -Wd -c my\_code\_that\_shouldnt\_break.py

Please note that you must have version 1.0 or later of the Pytest test

framework installed in order to run the tests. More information about Pytest is

available on the website\_\_.

\_\_ https://pytest.org/

COMPILER NOTES

==============

You can specify which Fortran compiler to use by using the following

install command::

python setup.py config\_fc --fcompiler=<Vendor> install

To see a valid list of <Vendor> names, run::

python setup.py config\_fc --help-fcompiler

IMPORTANT: It is highly recommended that all libraries that SciPy uses (e.g.

BLAS and ATLAS libraries) are built with the same Fortran compiler. In most

cases, if you mix compilers, you will not be able to import SciPy at best, and will have

crashes and random results at worst.

UNINSTALLING

============

When installing with ``python setup.py install`` or a variation on that, you do

not get proper uninstall behavior for an older already installed SciPy version.

In many cases that's not a problem, but if it turns out to be an issue, you

need to manually uninstall it first (remove from e.g. in

``/usr/lib/python3.4/site-packages/scipy`` or

``$HOME/lib/python3.4/site-packages/scipy``).

Alternatively, you can use ``pip install . --user`` instead of ``python

setup.py install --user`` in order to get reliable uninstall behavior.

The downside is that ``pip`` doesn't show you a build log and doesn't support

incremental rebuilds (it copies the whole source tree to a tempdir).

TROUBLESHOOTING

===============

If you experience problems when building/installing/testing SciPy, you

can ask help from scipy-user@python.org or scipy-dev@python.org mailing

lists. Please include the following information in your message:

NOTE: You can generate some of the following information (items 1-5,7)

in one command::

python -c 'from numpy.f2py.diagnose import run; run()'

1) Platform information::

python -c 'import os, sys; print(os.name, sys.platform)'

uname -a

OS, its distribution name and version information

etc.

2) Information about C, C++, Fortran compilers/linkers as reported by

the compilers when requesting their version information, e.g.,

the output of

::

gcc -v

g77 --version

3) Python version::

python -c 'import sys; print(sys.version)'

4) NumPy version::

python -c 'import numpy; print(numpy.\_\_version\_\_)'

5) ATLAS version, the locations of atlas and lapack libraries, building

information if any. If you have ATLAS version 3.3.6 or newer, then

give the output of the last command in

::

cd scipy/Lib/linalg

python setup\_atlas\_version.py build\_ext --inplace --force

python -c 'import atlas\_version'

7) The output of the following commands

::

python INSTALLDIR/numpy/distutils/system\_info.py

where INSTALLDIR is, for example, /usr/lib/python3.4/site-packages/.

8) Feel free to add any other relevant information.

For example, the full output (both stdout and stderr) of the SciPy

installation command can be very helpful. Since this output can be

rather large, ask before sending it into the mailing list (or

better yet, to one of the developers, if asked).

9) In case of failing to import extension modules, the output of

::

ldd /path/to/ext\_module.so

can be useful.