



Python for Astronomers

scipy & from 2 to 3

scipy

A collection of modules for
scientific applications.

scipy

- `scipy.cluster`
- `scipy.fftpack`
- `scipy.integrate`
- `scipy.interpolate`
- `scipy.linalg`
- `scipy.misc`
- `scipy.ndimage`
- `scipy.odr`
- `scipy.optimize`
- `scipy.signal`
- `scipy.sparse`
- `scipy.stats`
- `scipy.io`
- `scipy.lib`
- `scipy.sandbox`
- `scipy.special`
- `scipy.weave`

scipy.linalg

```
$ ipython -pylab
>>> import numpy
>>> import scipy.linalg
>>>
>>> A = numpy.array( [[1,1,1],
...                  [4,4,3],
...                  [7,8,5]], numpy.float)
>>>
>>> print numpy.linalg.inv(A)
>>> print scipy.linalg.inv(A)
```

scipy.linalg

```
>>> # calculate the right eigenvectors
>>> eval, evec = numpy.linalg.eig(A)
>>>
>>> print eval
>>> print evec[:,0] # first right eigenvector
>>>
>>> eval, levec, revec = scipy.linalg.eig(A,
...     left=True, right=True)
>>>
```

scipy.fftpack

```
>>> import scipy.fftpack
>>>
>>> x = numpy.linspace(0, 2*numpy.pi, 100)
>>> # perform complex 1D FFT:
>>> X1 = numpy.fft.fft(x)
>>> X2 = scipy.fftpack.fft(x)
>>>
>>> y = scipy.fftpack.ifft(X2)
>>>
>>> # also fft2 and fftn
```

scipy.integrate

```
>>> import scipy.integrate
>>>
>>> def f(x): return x**2
>>> x = numpy.linspace(1,3,10)
>>> y = f(x)
>>> # numerical integration of functions
>>> scipy.integrate.quad(f, 1, 3)
>>>
>>> scipy.integrate.trapz(y, x)
>>> scipy.integrate.simps(y, x)
```

scipy.interpolate

```
>>> import scipy.interpolate
>>> x = numpy.linspace(0, 10, 7)
>>> y = numpy.exp(-x / 3.)
>>> f = scipy.interpolate.interp1d(x, y)
>>>
>>> X = numpy.linspace(0, 10, 100)
>>> plot(x, y, 'bo')
>>> plot(X, f(X), 'g-')
>>> # only possible in range [0,10]
```

scipy.interpolate

```
>>> spl = \
    scipy.interpolate.UnivariateSpline(x,y)

>>> plot(X, spl(X), 'r--')

>>>

>>> Z = numpy.linspace(-1, 11, 100)

>>> plot(Z, spl(Z), 'm:')
```

scipy.optimize

- find minima of functions using different algorithms (e.g. downhill simplex)
- least square fitting
see `scipydemo_lsq.py`
- root finding of functions

scipy

- `scipy.special`: e.g. Bessel functions, Statistical functions, gamma, error function, Legendre function, ...
- `scipy.signal`: e.g. filtering and convolution functions
- `scipy.stats`: a large number of pdfs (normal, alpha, anglit, arcsine, and about 75 more ...)
- coming soon: `scipy 0.7`

Python 3.0

Python 3.0

From 2 to 3

Python 3.0 is the “*first ever intentionally backwards incompatible Python release*”.

From 2 to 3

- New syntax
- Changes syntax
- Removed syntax
- Library changes
- Some changes already present in 2.6

From 2 to 3

Old: `print "Hey there", 3*2`

New: `print("Hey there", 3*2)`

Old: `print "this is old",`

New: `print("this is new", end=" ")`

New: `print("The mean is <", 3, "> !", sep="")`

`# will output: "The mean is <3> !"`

From 2 to 3

```
Old: l = range(10)
```

```
Old: print l
```

```
[0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
```

```
New: l = list(range(10))
```

```
Old: for i in xrange(10): print i
```

```
New: for i in range(10): print I
```

```
Also changes for dict.keys(), dict.items(),  
map(), filter() and zip()
```

From 2 to 3

Old: `print 5/2`

`"2"`

New: `print(5/2)`

`"2.5"`

Old & New: `print 5//2` `print(5//2)`

`"2"`

From 2 to 3

Old: Text is 8-bit string or Unicode (u"Text")
& can represent binary data

New: Text is Unicode
datatypes **bytes** (immutable, b"0000") and
bytearray (mutable)

`str.encode(): str -> bytes`

`bytes.decode(): bytes -> str`

From 2 to 3

Old: No difference between text and binary files

New: support for binary mode

Old: String formatting `'%.2f' % math.pi`

New: `'%2.f' % math.pi`

`{0:.2f}.format(math.pi)`

Python 3 supports old string formatting and new, more powerful string formatting method

From 2 to 3

New: reserved keywords

`True, False, None, as, with`

Old: `except Exception, var: pass`

New: `except Exception as var: pass`

The way from 2 to 3

- Make your code work with “python2.6”
- Make your code work with “python2.6 -3”
- Run the “2to3” source-to-source translator

<http://docs.python.org/3.0/whatsnew/3.0.html>

<http://docs.python.org/3.0/library/2to3.html#to3-reference>

Guido van Rossum

Over six years ago, in December 1989, I was looking for a "hobby" programming project that would keep me occupied during the week around Christmas. My office ... would be closed, but I had a home computer, and not much else on my hands. I decided to write an interpreter for the new scripting language I had been thinking about lately: a descendant of ABC that would appeal to Unix/C hackers. I chose Python as a working title for the project, being in a slightly irreverent mood (and a big fan of Monty Python's Flying Circus).

