Scipy for Statistical Computing

Berian James

24 Sep 2012

Scipy contains a large number of statistics, data analysis, and signal processing routines

- dozens of discrete & continuous probability distributions, descriptive statistics, statistical tests, linear regression
- efficient nearest neighbor search & Delaunay tesselation
- sophisticated signal processing: FFT, filtering, convolution, wavelets, ...
- smoothing and interpolation
- optimization and model-fitting
- statistical clustering routines: K-means, hierarchical clustering

Of course, other modules provide even more sophisticated statistical tools (e.g., scikits) & seamless integration with R (via rpy2)

We will talk about these modules next week!

Subpackage	Description
cluster	Clustering algorithms
constants	Physical and mathematical constants
fftpack	Fast Fourier Transform routines
integrate	Integration and ordinary differential equation solvers
interpolate	Interpolation and smoothing splines
io	Input and Output
linalg	Linear algebra
maxentropy	Maximum entropy methods
ndimage	N-dimensional image processing
odr	Orthogonal distance regression
optimize	Optimization and root-finding routines
signal	Signal processing
sparse	Sparse matrices and associated routines
spatial	Spatial data structures and algorithms
special	Special functions
stats	Statistical distributions and functions
weave	C/C++ integration

import scipy.stats as stats

```
>>> import scipy.stats as stats
>>> stats?

.. module:: scipy.stats

This module contains a large number of probability distributions as
well as a growing library of statistical functions.
```

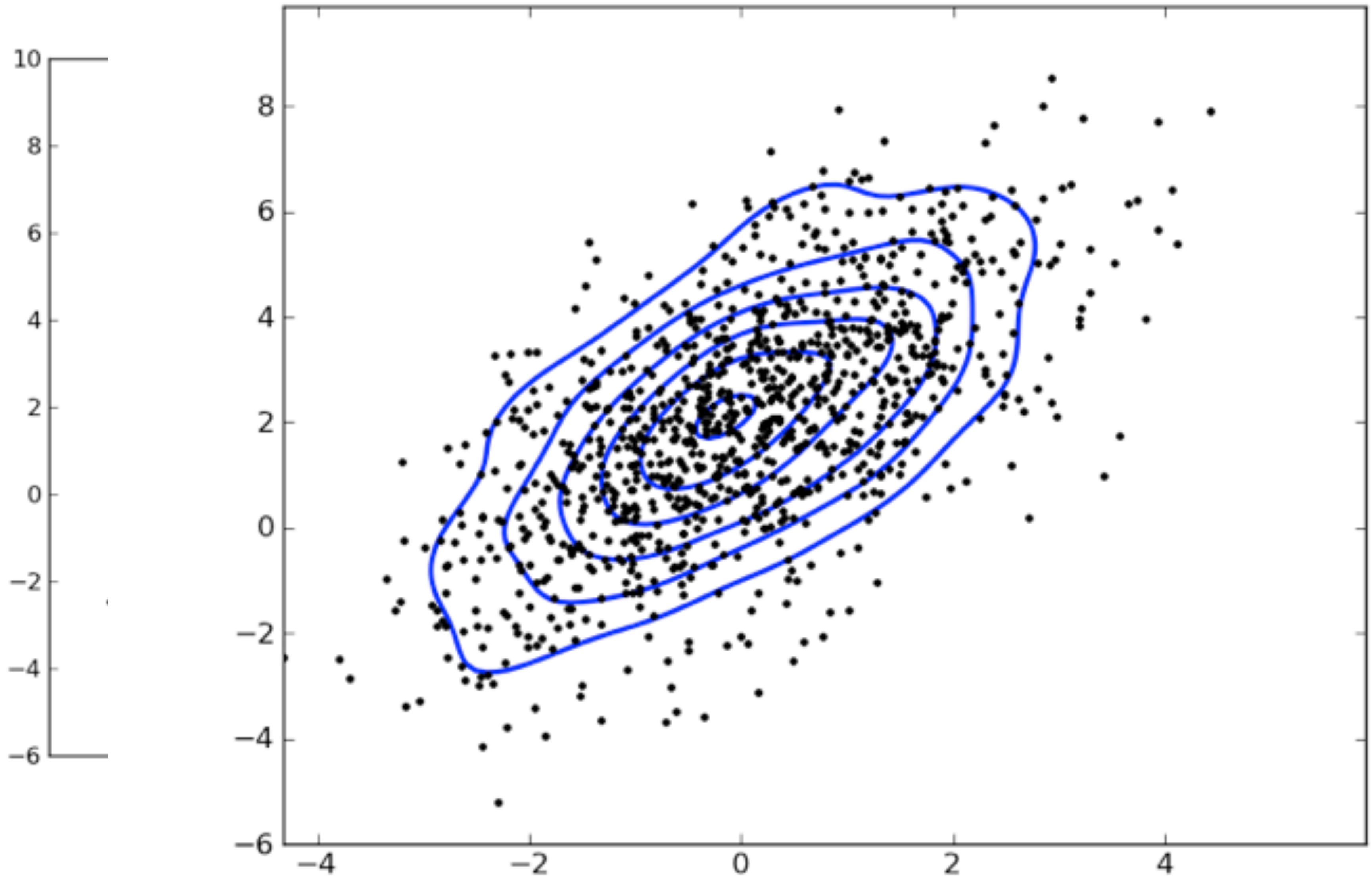
```
Continuous distributions
========================
   norm                 -- Normal
(Gaussian)
   alpha                -- Alpha
   anglit               -- Anglit
   arcsine              -- Arcsine
   beta                 -- Beta
   betaprime            -- Beta Prime
   bradford             -- Bradford
   burr                 -- Burr
   cauchy               -- Cauchy
etc., etc.
```

```
Discrete distributions
======================
    binom                 -- Binomial
    bernoulli             -- Bernoulli
    nbinom                -- Negative Bin.
    geom                  -- Geometric
    hypergeom             -- Hypergeom.
    logser                -- Logarithmic
    poisson               -- Poisson
    planck                -- Planck
    boltzmann             -- Boltzmann
    randint               -- Discrete Unif.
etc., etc.
```

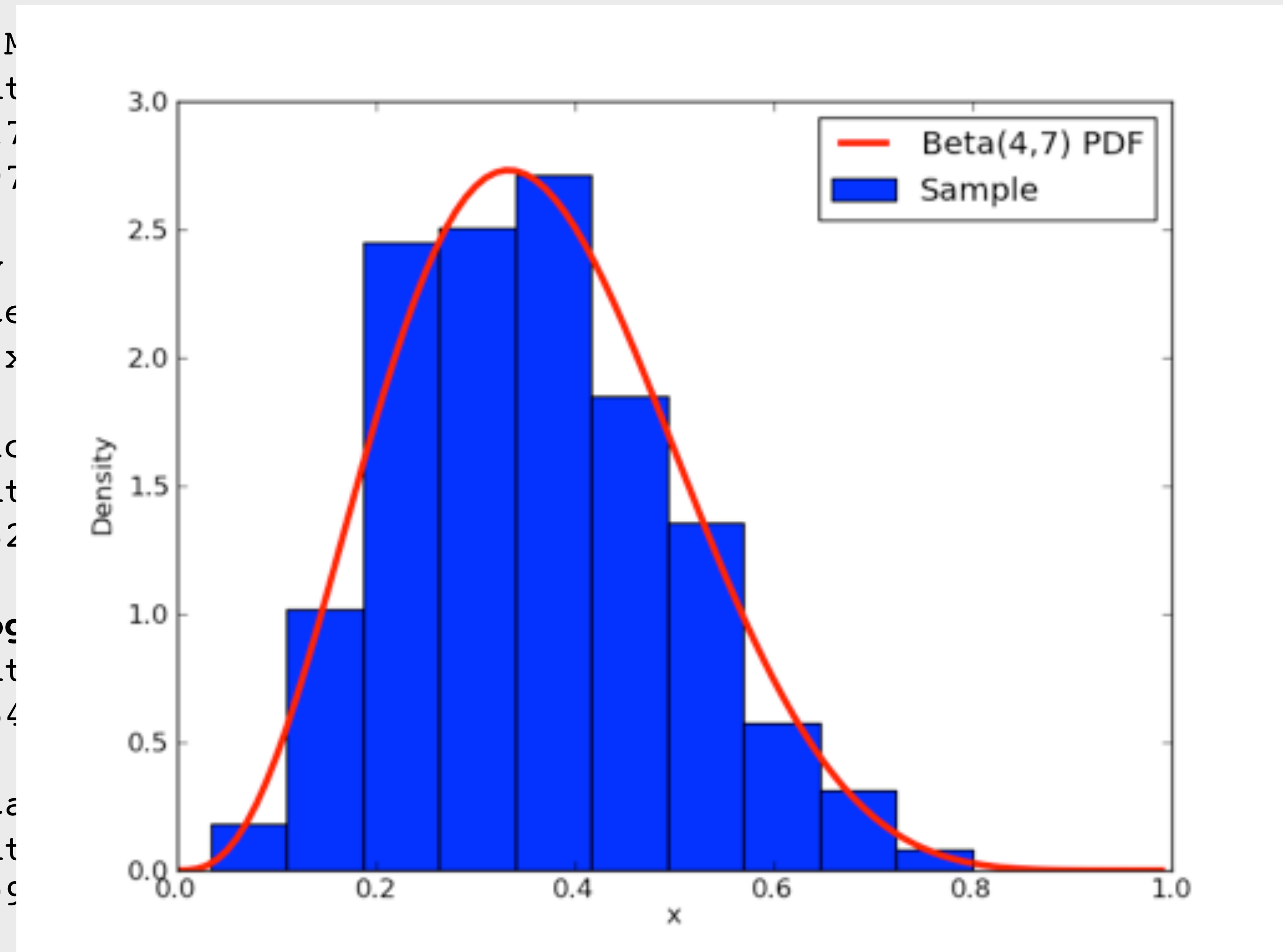
Working with distributions

```
>>> rv = stats.norm()
>>> rv.mean()
0.0
>>> rv.std()
1.0
>>> rv.pdf([-2,-1,0,1,2])
array([ 0.05399097,  0.24197072,  0.39894228,  0.24197072,  0.05399097])
>>> rv.cdf([-2,-1,0,1,2])
array([ 0.02275013,  0.15865525,  0.5       ,  0.84134475,  0.97724987])
>>> 1 - rv.sf([-2,-1,0,1,2]) # 1 - survival function = CDF
array([ 0.02275013,  0.15865525,  0.5       ,  0.84134475,  0.97724987])
>>> rv.ppf([.0227,.159,.5, 0.841, 0.977]) # inverse CDF
array([-2.00092939, -0.99857627,  0.        ,  0.99857627,  1.99539331])
>>> rv.rvs(10) # take a random sample from N(0,1)
array([ 1.68222907, -0.6600351 , -0.5766894 , -2.18279668,
0.37017356,-1.47523043, -1.75281044, -0.38918405,  0.09937893,
0.755736  ])
>>> rv.moment(3) # non-central moments
0.0
>>> rv.moment(4)
3.0
>>> rv.interval(alpha=0.95)
(-1.959963984540054, 1.959963984540054)
```

```
# sampling a multivariate normal distribution
>>> mu = [0,2]
>>> sig = [[2,2],[2,5]]
>>> sample = numpy.random.multivariate_normal(mu,sig,1000)
>>> plot(s
```



```
# maximum likelihood estimation
# take a sample from Beta(4,7)
>>> betasamp = stats.beta.rvs(4,7,size=1000)
# plot histogram of sample plus Beta(4,7) PDF
>>> xvec = arange(0,1,0.01)
>>> plot(xvec, map(lambda x: stats.beta.pdf(x,4,7),xvec), 'r-',
linewidth=3)
>>> hist(betasamp,normed=True)
>>> legend(("Beta(4,7) PDF", "Sample"))

# find M
>>> stat
(2.75917
0.770497

# every                                                        o
paramete                                                       n like
this: (x

# fix lo
>>> stat
(3.91832

# Kolmog
>>> stat
(0.02154

>>> beta
>>> stat
(0.35299
```



Let’s analyze some real data…

We can do more than just linear model fitting

- Polynomial Regression

```
numpy.polyfit(x,y,deg=2)
```

- Smoothing Splines

```
scipy.interpolate.splrep(x, y, k=3, s=0.1)
```

- Non-linear Regression

```
scipy.optimize.curve_fit(f, xdata, ydata)
```

- Non-parametric Regression on Basis Functions

```
numpy.fft; scipy.signal.wavelets; scipy.signal.cheby1
scipy.signal.lombscargle; scipy.optimize.leastsq
```

Other non-parametric regression models such as loess, local-linear smoothing and trees don’t live within scipy

What other data analysis tools are in scipy?

- Statistical clustering: K-means and hierarchical clustering in scipy.cluster
- Kernel Density Estimation: scipy.stats.kde
- Simple image manipulation tools: scipy.ndimage
- Image filtering, denoising: scipy.signal
- Convolutions and deconvolutions: scipy.signal.fftconvolve, scipy.signal.deconvolve
- Sparse 2D matrix operations: scipy.sparse
- Optimized FFT implementation: scipy.fftpack
- Interpolation & smoothing: scipy.interpolate
- Optimization tools: scipy.optimize
- And much more.....

scikits.statsmodels - a brief introduction

```
>>> import statsmodels.api as sm
```

http://scikits.appspot.com/statsmodels

Statistical models and computations for SciPy

- statsmodels is a statistical modelling and computation toolbox for numpy/scipy, aimed at complementing scipy.stats with 'frequentist' modelling tools; cf. pymc, which is a 'bayesian' toolbox.

- It is built on numpy, i.e., numpy arrays are the most practical data type; they are generic, efficient and straight-forward to handle. Some of the time-series analysis is designed for use with Pandas (more on this later).

- statsmodels is available through PyPI, easy_install, github, ...
  http://pypi.python.org/pypi/scikits.statsmodels

- statsmodels is already compatible with Python 3 and is almost wholly pure python, with a handful of cython wrappings

scikits.statsmodels resources

- http://scikits.appspot.com/statsmodels
  statsmodels homepage, download, installation

- http://statsmodels.sourceforge.net/
  statsmodels documentation, API reference, examples; not complete

- http://www.github.com/statsmodels/statsmodels
  http://pypi.python.org/pypi/scikits.statsmodels/
  statsmodels repositories

- http://conference.scipy.org/scipy2010/slides/skipper_seabold_statsmodels.pdf
  http://conference.scipy.org/scipy2011/slides/mckinney_time_series.pdf
  SciPy 2010/2011 by Skipper Seabold and Wes McKinney

Inbuilt data sets and statsmodels io

- statsmodels contains a number of inbuilt data sets (sm.datasets)
  e.g. >>> data = sm.datasets.scotland.load()

- Variables are cast as either ‘endogenous’ or ‘exogenous’

- Particularly with the time series analysis module, the pandas TimeSeries data structure is available for use

- Ultimately, statsmodels is targetted at (in the words of its creator) “Statistical, Financial Econometric, and Econometric models”

Regression in statsmodels

- Implementation of least-squares routines: ordinary least squares, weight least squares and general least squares.

- Discuss notebook for examples

- Extensions of these methods: generalised linear models and robust linear models, which will not be covered here.

- There are also time-series specific regression methods.

Time-series analysis and regression

- statsmodels provides fundamental time-series analysis methods, including:
  - auto- and cross-correlation and -covariance
    `sm.tsa.acovf, sm.tsa.acf, sm.tsa.ccf, sm.tsa.ccovf`
  - periodogram for regularly-spaced data, i.e. $|\mathcal{F}(\mathbf{x})|^2/N$
    `sm.tsa.periodogram`
- Many of these are also available through numpy/scipy, so that the power of sm.tsa lies in its estimation methods, for univariate and vector autoregressive processes (AR, VAR) and auto-regressive moving-average processes (ARMA)
- Discuss example in notebook

Later in the course we will cover more sophisticated statistical computing tools available through scikits, rpy2, pandas, etc.

At the end of the scipy.stats docstring:

For many more stat related functions install the software R and the interface package rpy.