Random Numbers

Probability, Statistics and Discrete Mathematics, Spring 2017

Helsinki Metropolia University of Applied Sciences

Importing Modules

```
In [1]: %pylab inline
```

```
Populating the interactive namespace from numpy and matplotlib
```

Is same as writing all these commands:

```
import numpy
import matplotlib
from matplotlib import pylab, mlab, pyplot
np = numpy
plt = pyplot

from IPython.display import display
from IPython.core.pylabtools import figsize, getfigs

from pylab import *
from numpy import *
```

Basic Statistics

```
In [6]: x = np.array([3.5, 1.1, 3.2, 2.8, 6.7, 4.4, 0.9, 2.2])
In [7]: np.mean(x)
Out[7]: 3.10
In [8]: np.median(x)
Out[8]: 3.0
In [9]: x.min(), x.max()
Out[9]: (0.90, 6.70)
In [10]: x.var()
Out[10]: 3.07
In [11]: x.std()
Out[11]: 1.7521415467935233
```

Random Number generators

```
In [14]: np.random.rand()
Out[14]: 0.532833024789759
In [15]: np.random.randn()
Out[15]: 0.876834210149254 1

In [16]: np.random.rand(5)
Out[16]: array([ 0.71356403,  0.25699895,  0.75269361,  0.88387918,  0.15489908])
In [17]: np.random.randn(2, 4)
Out[17]: array([[ 3.13325952,  1.15727052,  1.37591514,  0.94302846],
                [ 0.8478706 ,  0.52969142, -0.56940469,  0.83180456]])
```

From np.random import rand, randn

Random Integers

```
In [18]: np.random.randint(10, size=10)
Out[18]: array([0, 3, 8, 3, 9, 0, 6, 9, 2, 7])
In [19]: np.random.randint(low=10, high=20, size=(2, 10))
Out[19]: array([[12, 18, 18, 17, 14, 12, 14, 10, 16, 19],
                [15, 13, 15, 18, 11, 17, 17, 10, 13, 17]])
```

Random sampling (numpy.random)

Simple random data

rand(d0, d1, ..., dn)	Random values in a given shape.
randn(d0, d1, ..., dn)	Return a sample (or samples) from the "standard normal" distribution.
randint(low[, high, size, dtype])	Return random integers from *low* (inclusive) to *high* (exclusive).
random_integers(low[, high, size])	Random integers of type np.int between *low* and *high*, inclusive.
random_sample([size])	Return random floats in the half-open interval [0.0, 1.0).
random([size])	Return random floats in the half-open interval [0.0, 1.0).
ranf([size])	Return random floats in the half-open interval [0.0, 1.0).
sample([size])	Return random floats in the half-open interval [0.0, 1.0).
choice(a[, size, replace, p])	Generates a random sample from a given 1-D array
bytes(length)	Return random bytes.

https://docs.scipy.org/doc/numpy/reference/routines.random.html

Random generator settings

Random generator

RandomState	Container for the Mersenne Twister pseudo-random number generator.
seed([seed])	Seed the generator.
get_state()	Return a tuple representing the internal state of the generator.
set_state(state)	Set the internal state of the generator from a tuple.