1_3_An_introductory_exercise_12345

May 13, 2015

QuantEcon 1.3 Exercises
Exercise 1

In [2]:
```
"""
Exercise 1: the function 'factorial(n)' calculates n!
"""

def factorial(n):

    fval = 1; # initial value of n

    for i in range(n):     # i in [0,...,n]
        fval = fval*(n-i) # iteration of n*(n-1)*...*1
    return fval
```

Exercise 2

In [21]:
```
"""
Exercise 2: the function 'binomial_rv(n, p)' calculates a random variable from B(n,p)
recall function 'factorial', defined in Exercise 1.
"""

def binomial_rv(n, p):

    from random import uniform

    U = uniform(0,1)

    nf = factorial(n)

    F = 0

    for x in range(n+1):

        xf  = factorial(x)    # x!
        nxf = factorial(n-x)  # (n-x)!
        F   = F + nf/(xf*nxf)*(p**x)*((1-p)**(n-x)) # sum nCx p^x(1-p)^(n-x)

        if U<=F: # check U <= F(x).
            break

    return x
```

Exercise 3

```
In [20]: """
         Exercise 3: Approximation to pi by Monte-Carlo
         """

         from random import uniform

         n = 10000 #
         k = 0     #

         for i in range(n):

             Ux = uniform(0, 1) # pseudo radom intval [0,1]
             Uy = uniform(0, 1)

             if Ux**2+Uy**2<1:
                 k +=  1 # count U lies in subset B

         k*1.0/n*4 # approximation to pi

Out[20]: 3.1676
```

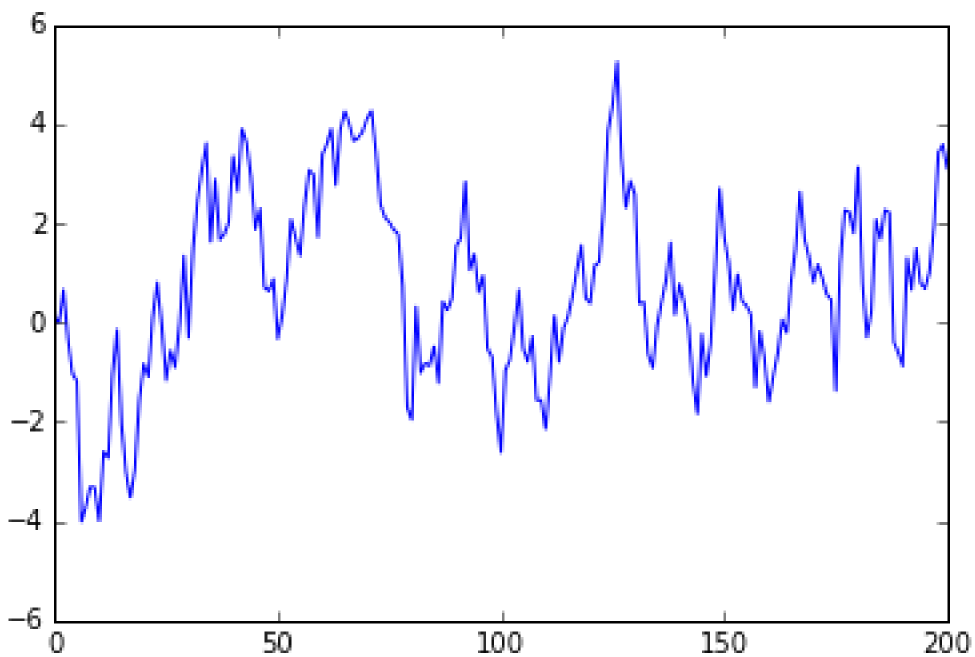
Exercise 4

```
In [19]: """
         Exercise 4: coin toss program
         If 3 consecutive heads occur one or more times, pay one dollar.
         """

         from random import uniform

         k = 0 # # of consecutive heads occur within 3 times
         t = 0 # # of 3 consecutive heads occur

         for i in range(10):

             if uniform(0,1) > 0.5: # if head occurs
                 k += 1
                 if k == 3:
                     t += 1
             else:
                 k = 0 # if tale occures, reset the consecutive-head record

         pay = 1 if t > 0 else 0

         pay

Out[19]: 1
```

Exercise 5

```
In [73]: """
         Exercise 5: Plot AR1 process
         """

         import matplotlib.pyplot as plt
```

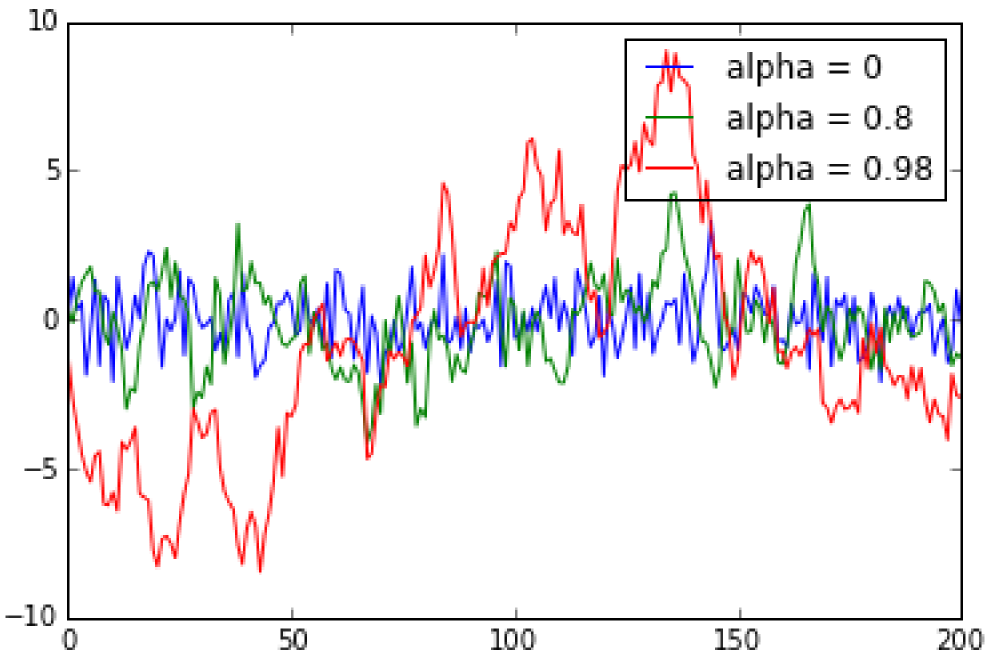
```python
from random import normalvariate as randn

# params

T = 200         # total periods
alpha = 0.9     # persistence
x_series = [] # prepare box for timeseries
x = 0           # initual of x

for t in range(T+1):
    x = alpha*x+randn(0,1)
    x_series.append(x)

%matplotlib inline
plt.plot(x_series, 'b-')
plt.show()
```



Exercise 6

In [41]:
```python
"""
Exercise 6: Plot 3 AR1 process in one figure
"""

import matplotlib.pyplot as plt
from random import normalvariate
%matplotlib inline

alphas = [0, 0.8, 0.98] # persistence

# params
```

```
T = 200        # total periods

for alpha in alphas:
    x_series = [] # reset and prepare box for timeseries
    x = 0         # reset inital of x
    for t in range(T+1):
        x = alpha*x+randn(0,1)
        x_series.append(x)
    plt.plot(x_series, label = 'alpha = ' + str(alpha))
plt.legend()
plt.show()
```



In [ ]:

In [ ]:

In [ ]: