Introduction to Python

Lecture 5

Arul Lakshminarayan, 27/10/17

Random Numbers, Distributions, Random walks, diffusion.

```
import numpy as np
import pylab as plt

#import random

print(np.random.random())
print(np.random.randint(2,10))
print(np.random.randn()) #random does not have randn, numpy random does.
print(np.random.randn(10))
print(np.random.standard_normal())
```

```
0.44350529372513126
6
0.7539763503165181
[ 0.78235046 -1.93029173  0.82831209  0.98688588  0.22914279  0.10015548
 -1.77098236  0.58417319 -0.32755198 -0.74392304]
0.3754486459368379
```

Distributions

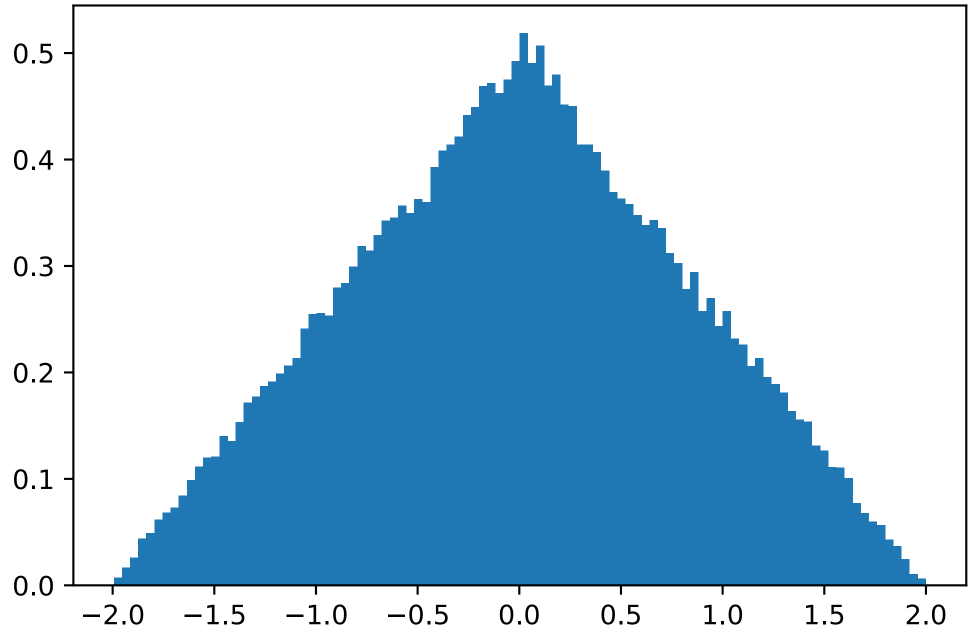
```
print(np.random.normal(10,1,10)) #second argument is "scale" which is SD
#not variance.
print(np.random.normal(10,5,10))
print(np.random.triangular(-1,0,1,10))
```

```
[  9.92541085  11.15491144   9.37169858   8.49200941  11.16353265
  10.41560171  10.14376784  10.61781619   9.16879106  11.6186145 ]
[  9.9524564    9.62457032   9.68267751  21.9433258   12.14252911
  14.87027148  15.33858677  13.03726948   8.56657642  13.39298587]
[ 0.43599409  0.29512933  0.42096111 -0.13285522  0.2574846   0.07986563
  0.66169105  0.23906404 -0.50036016 -0.00808208]
```

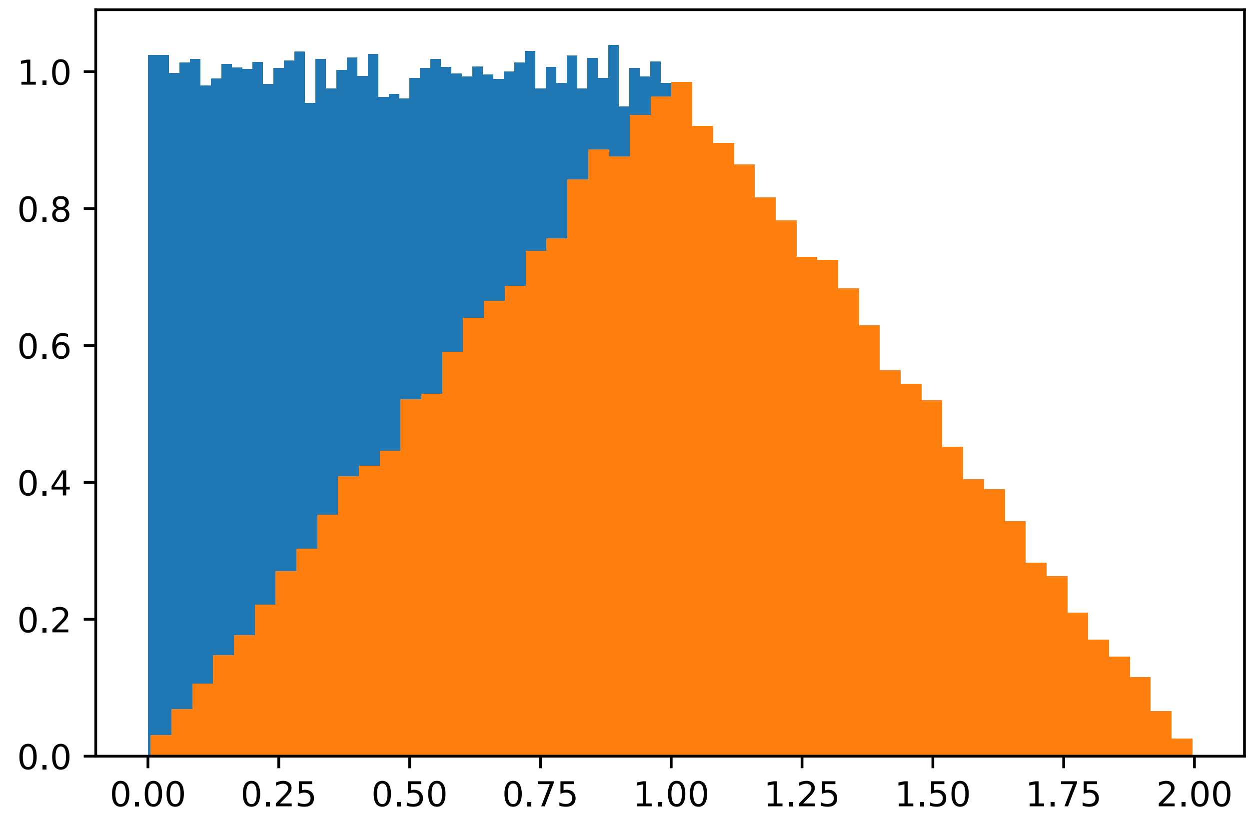
Plotting histogram

```
stnorm=np.random.randn(100000)
plt.hist(stnorm)
#plt.hist(stnorm,bins=50)
#plt.hist(stnorm,bins=50,normed='True')
plt.show()
```

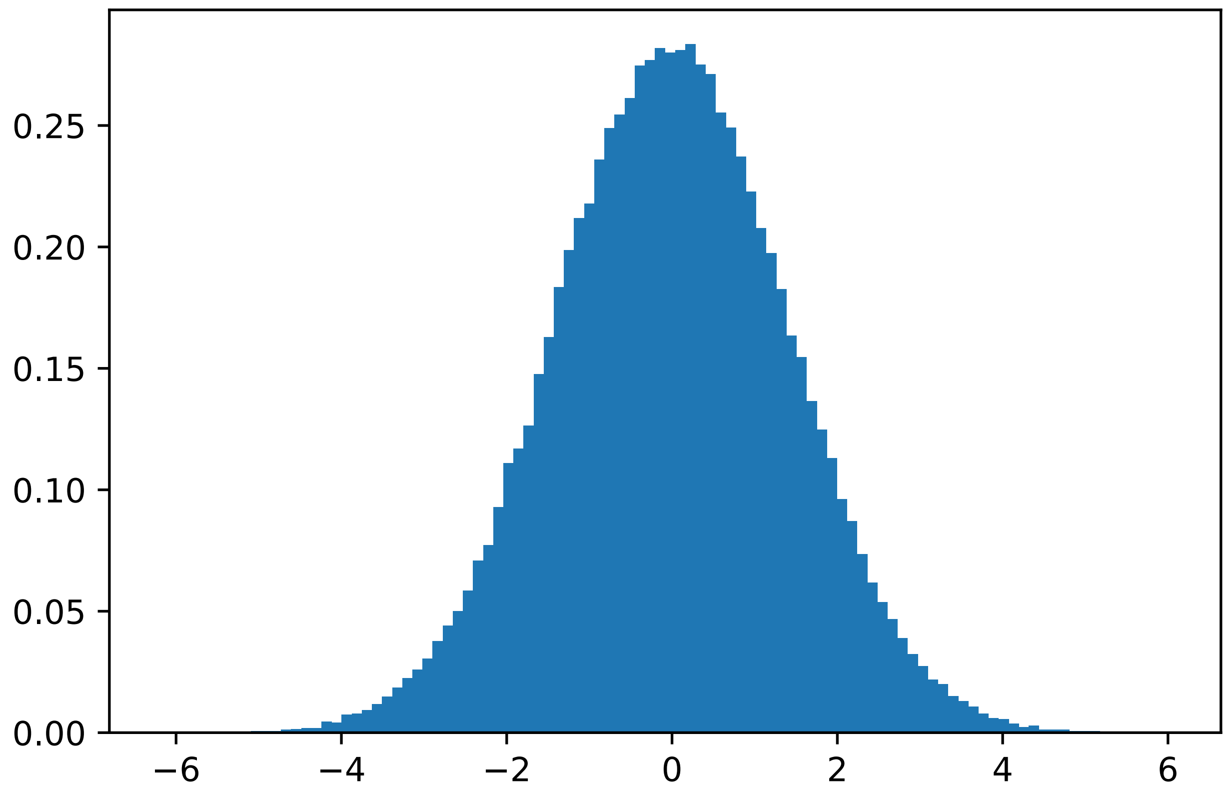
```
triran=np.random.triangular(-2,0,2,100000)
#plt.hist(stnorm)
#plt.hist(stnorm,bins=50)
plt.hist(triran,bins=100,normed='True')
plt.show()
```



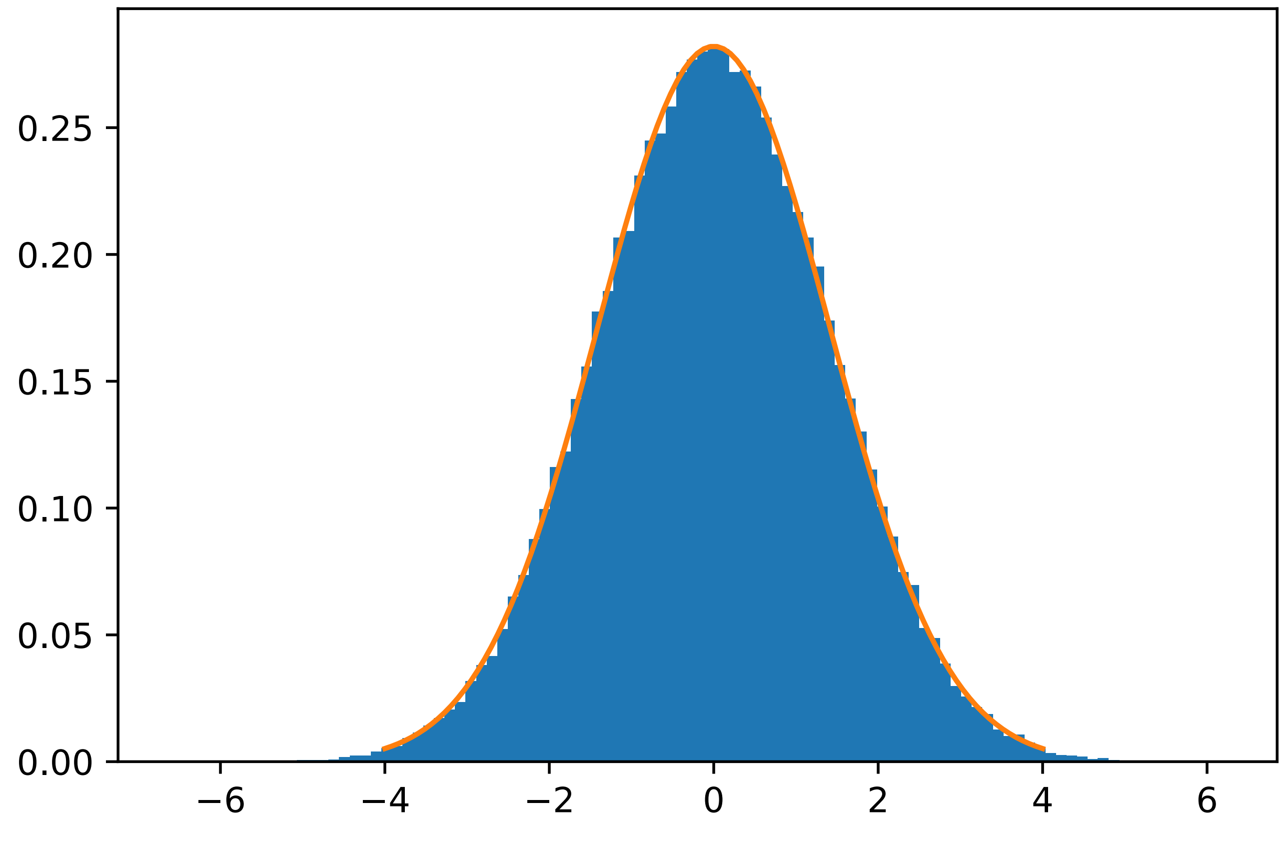
```
xran=np.random.random(100000)
plt.hist(xran,bins=50,normed='True')
#plt.show()

xran=np.random.random(100000)+np.random.random(100000)
plt.hist(xran,bins=50,normed='True')
#plt.show()
plt.savefig('sumof2unif.pdf')
```



```
xran=np.random.randn(100000)+np.random.randn(100000)
plt.hist(xran,bins=100,normed='True')
#plt.show()

#xran=np.random.randn(100000)+np.random.randn(100000)
#plt.hist(xran/2,bins=100,normed='True')
#plt.show()
plt.savefig('sumof2normal.pdf')
```

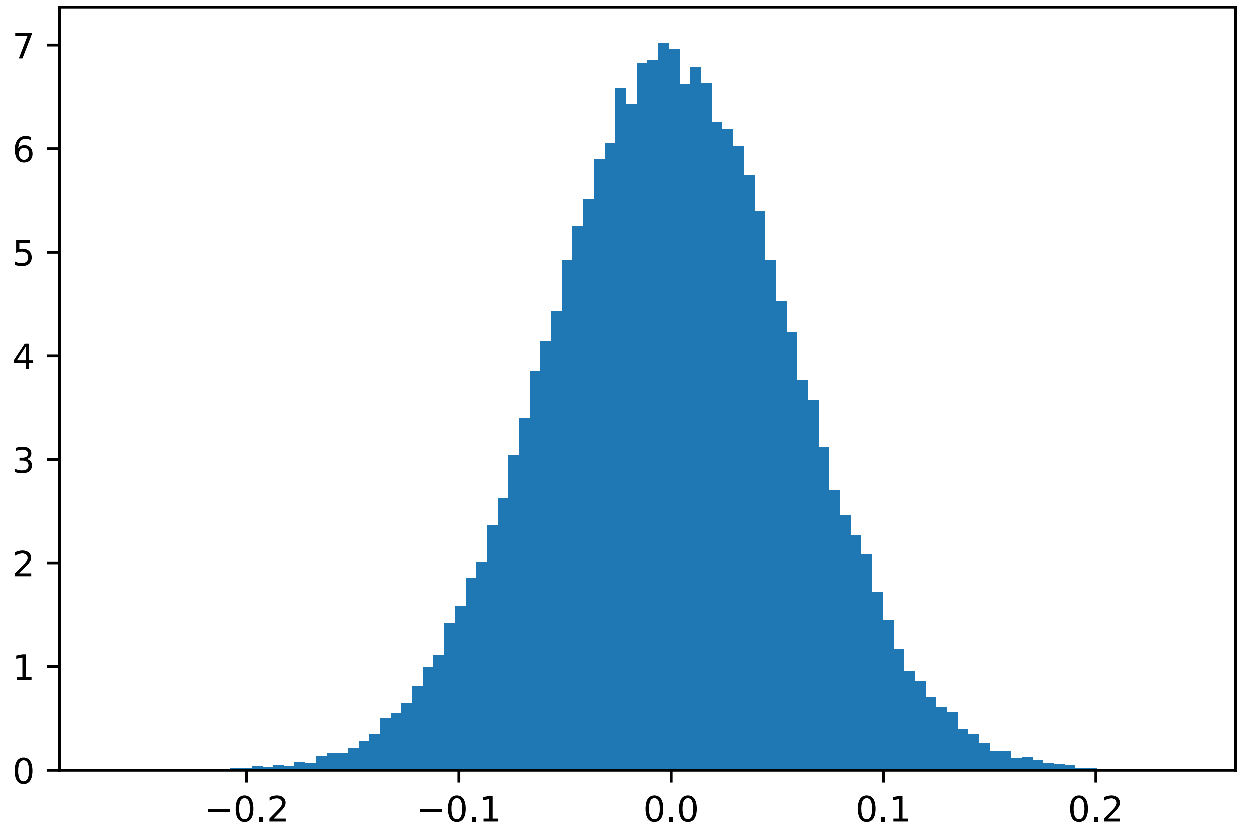


```
xran=np.random.randn(100000)+np.random.randn(100000)
plt.hist(xran,bins=100,normed='True')

x=np.linspace(-4,4,100)
y=np.exp(-x*x/4)/(2*np.pi*2)**.5
plt.plot(x,y)
plt.savefig('sumof2normal_curve.pdf')
```



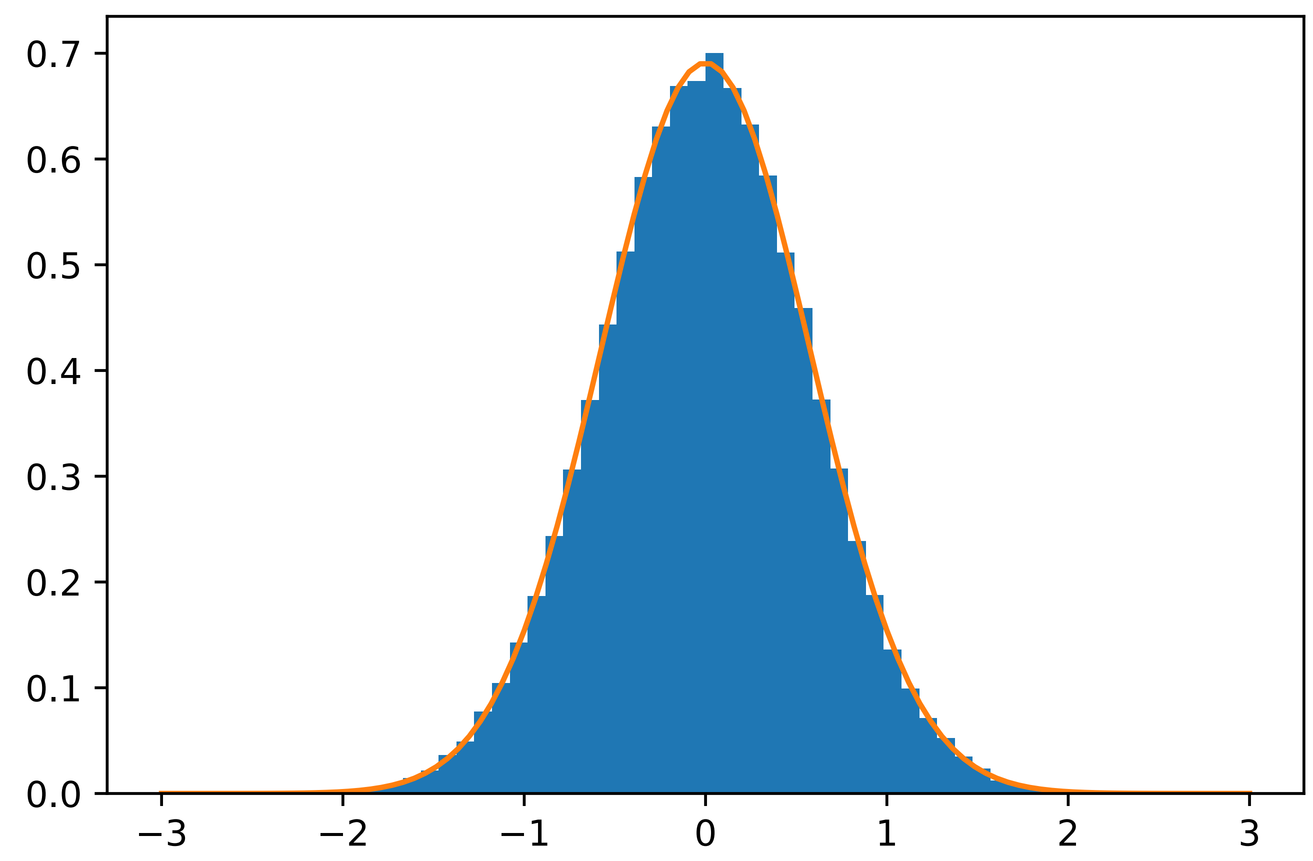
Central limit theorem: illustration

```
##The following 6 lines illustrates the Central Limit Theorem
x=np.zeros(100000)
for i in range(100):
    x+=np.random.uniform(-1,1,100000)
x=x/100
plt.hist(x,bins=100,normed='True')
#plt.show()
plt.savefig('avg_of100unif.pdf')
```



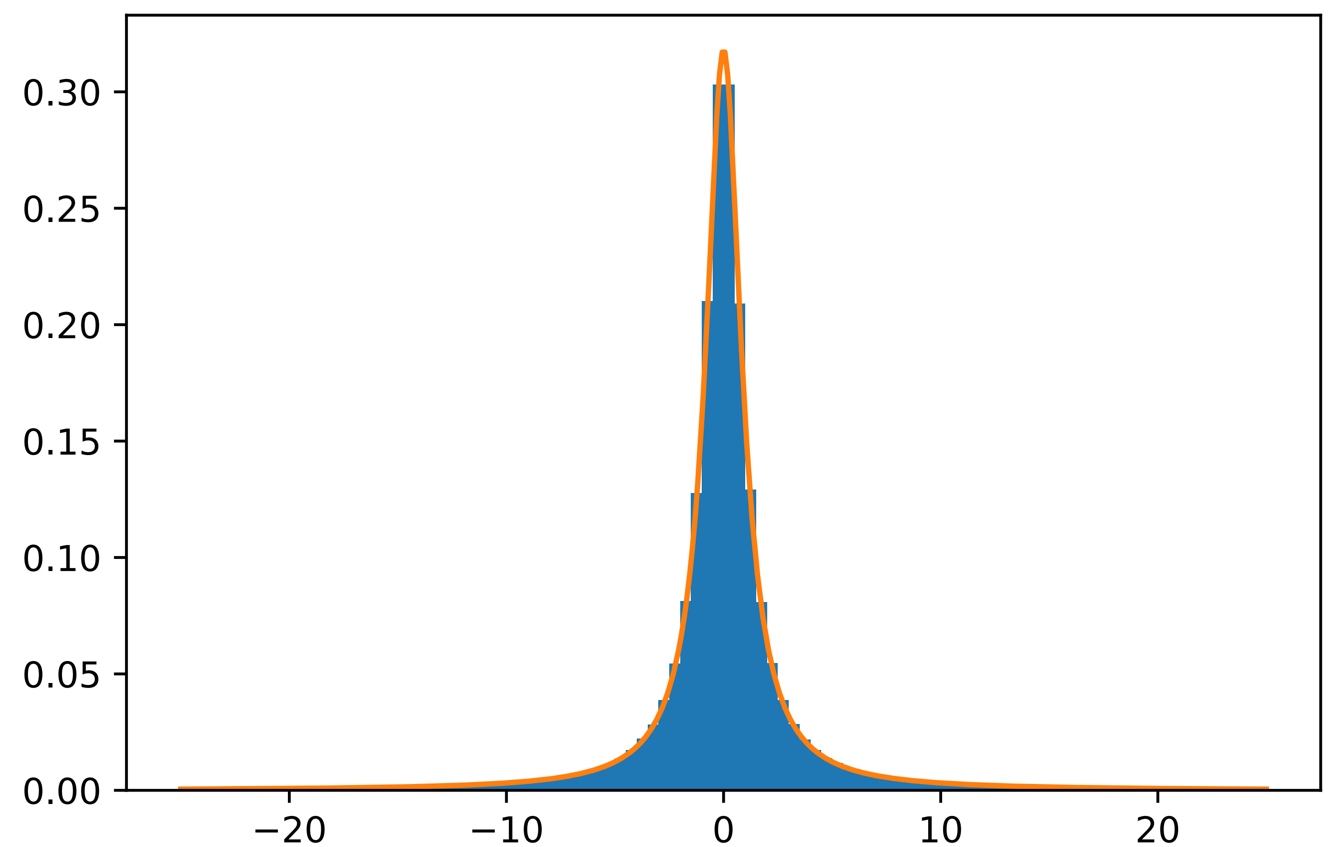
Central limit theorem: illustration

```
x=np.zeros(100000)
for i in range(100):
    x+=np.random.uniform(-1,1,100000)
x=x/10
plt.hist(x,bins=50,normed='True')
x=np.linspace(-3,3,100)
sig=(1/3)**.5 #\sigma of uniform [-1,1]
s=sig
y=np.exp(-x*x/(2*s**2))/(2*np.pi*s**2)**.5
plt.plot(x,y)
#plt.show()
plt.savefig('avg_of100unif.pdf')
```

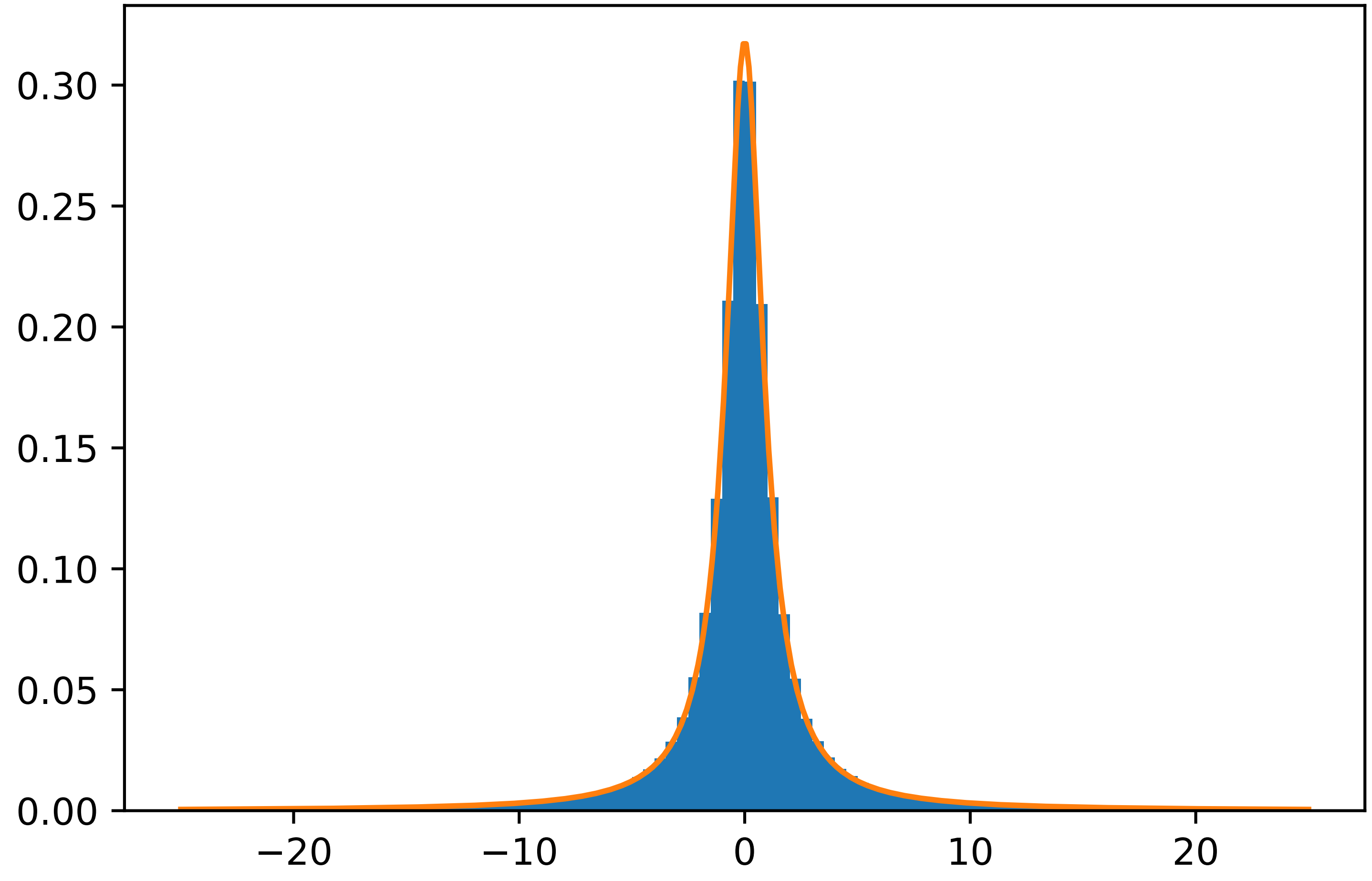


When CLT Fails: Distributions without finite moments. “fat tail”, “heavy tail”. Example of Cauchy

```
#Standard Cauchy:1/pi*(1+x**2)
#
xc=np.random.standard_cauchy(1000000)
xc = xc[(xc>-25) & (xc<25)]
plt.hist(xc,bins=100,normed=1)
x=np.linspace(-25,25,400)
y=1/(np.pi*(1+x*x))
plt.plot(x,y)
#plt.show()
plt.savefig('Cauchy1.pdf')
```

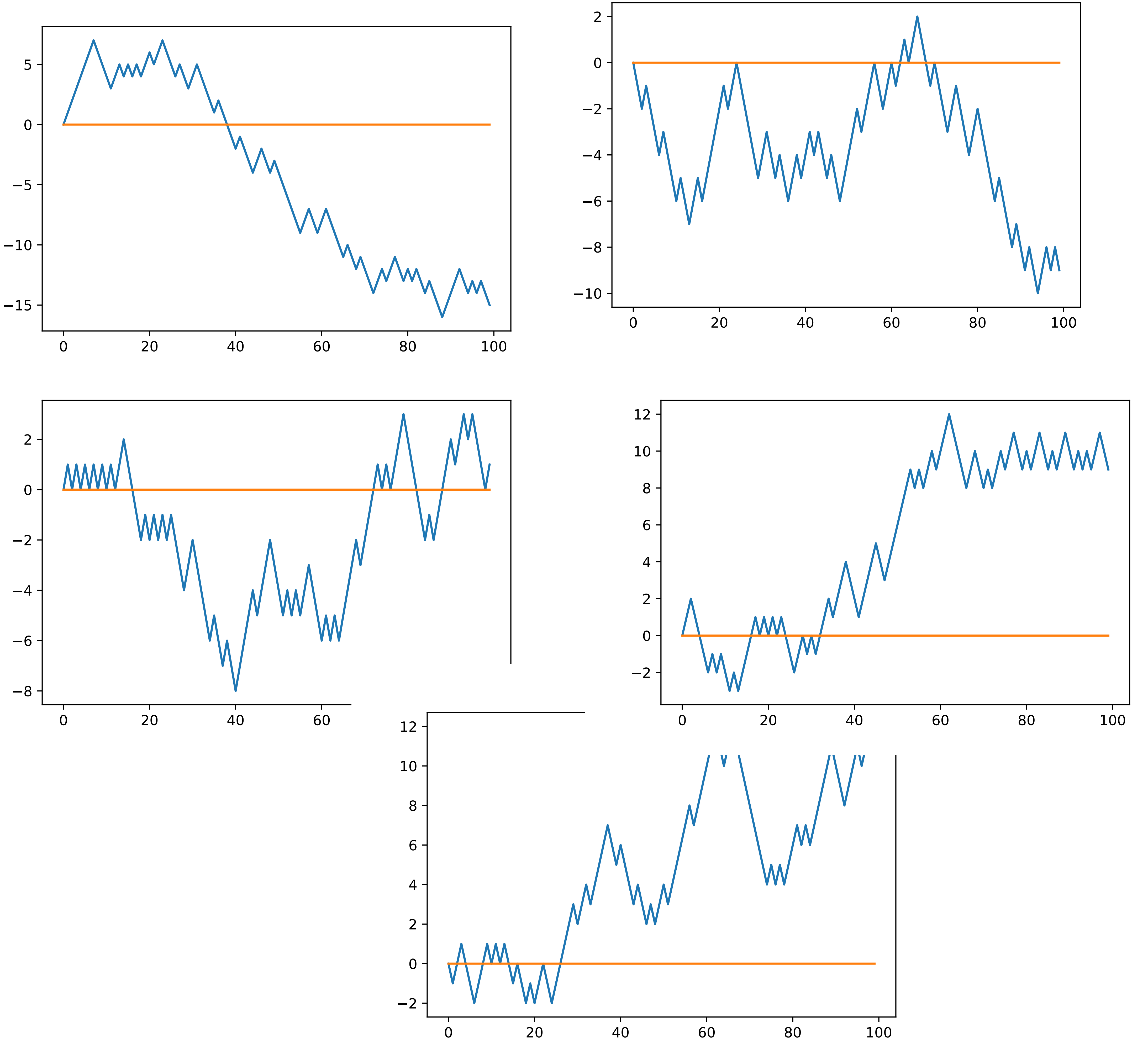


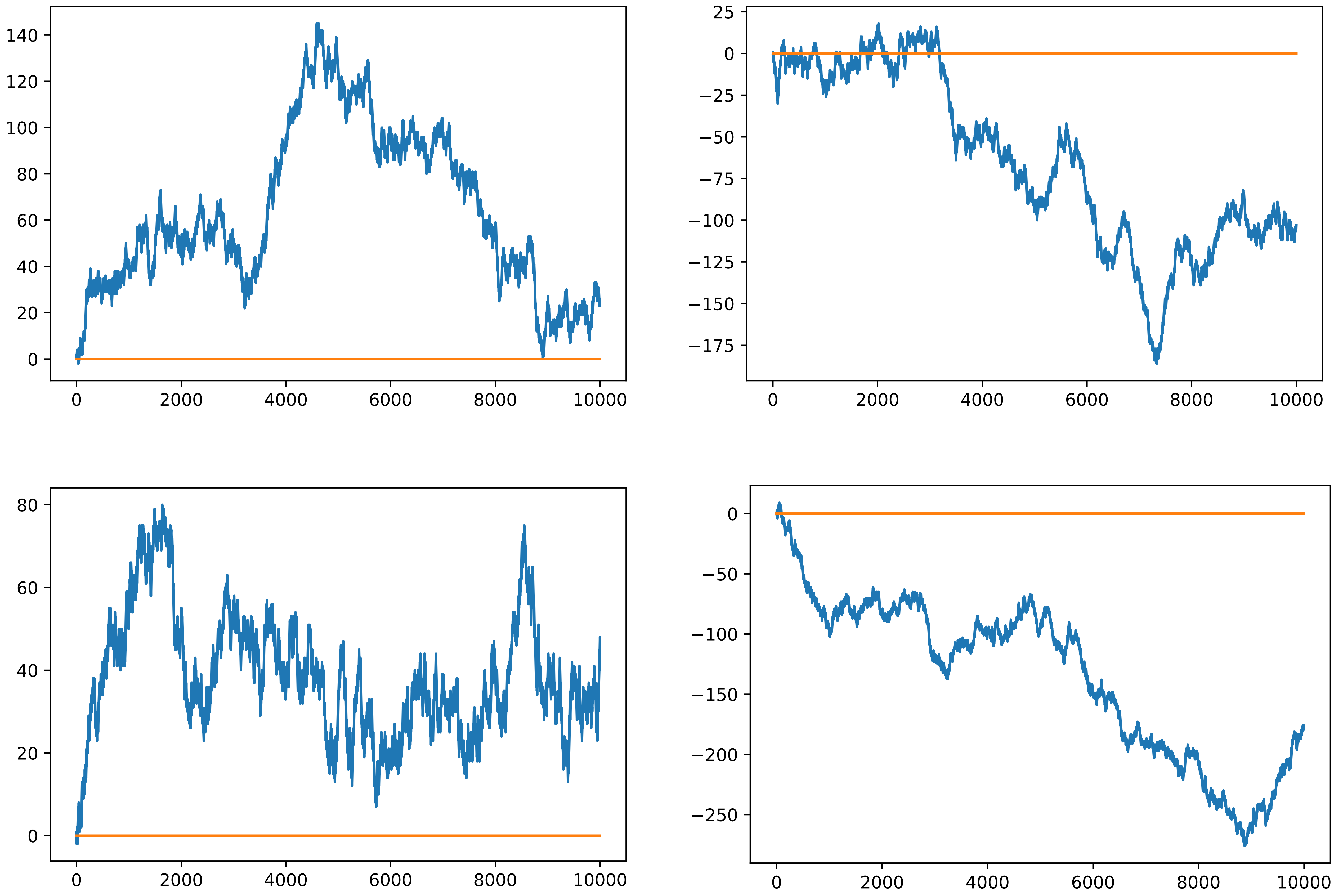
```
#Standard Cauchy is a Levy Stable Distribution of index 1.
##\sum(x_i, 1=1,...,N)/N is just as "wide" as original distribution,
# in fact it is exactly standard Cauchy again!

xc=np.random.standard_cauchy(1000000)+np.random.standard_cauchy(1000000)
xc=xc/2
xc = xc[(xc>-25) & (xc<25)]
plt.hist(xc,bins=100,normed=1)
x=np.linspace(-25,25,400)
y=1/(np.pi*(1+x*x))
plt.plot(x,y)
#plt.show()
plt.savefig('CauchyLevy.pdf')
```



Diffusion, Random Walk

```
import numpy as np
import pylab as plt

def d1rw(n): #of steps
    x=np.zeros(n) #initial array set as 0
    for i in range(n-1):
        if np.random.randint(0,2)==0:
            x[i+1]=x[i]+1
        else:
            x[i+1]=x[i]-1
    print(x)
    y=np.zeros(n)
    plt.plot(x)
    plt.plot(y)
    plt.show()
```

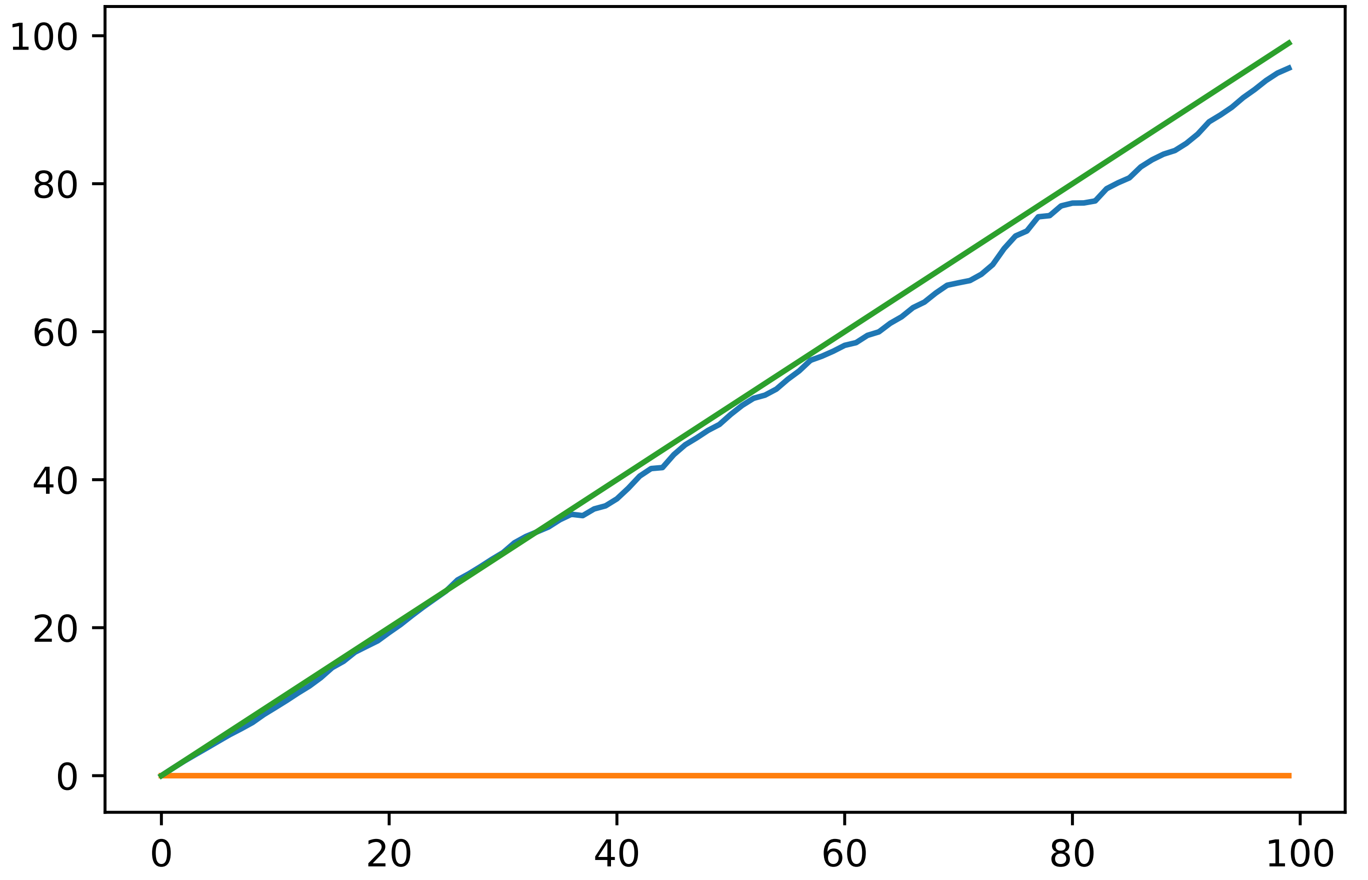

5
0
−5
−10
−15
0 20 40 60 80 100
2
0
−2
−4
−6
−8
−10
0 20 40 60 80 100
2
0
−2
−4
−6
−8
0 20 40 60
12
10
8
6
4
2
0
−2
0 20 40 60 80 100
12
10
8
6
4
2
0
−2
0 20 40 60 80 100

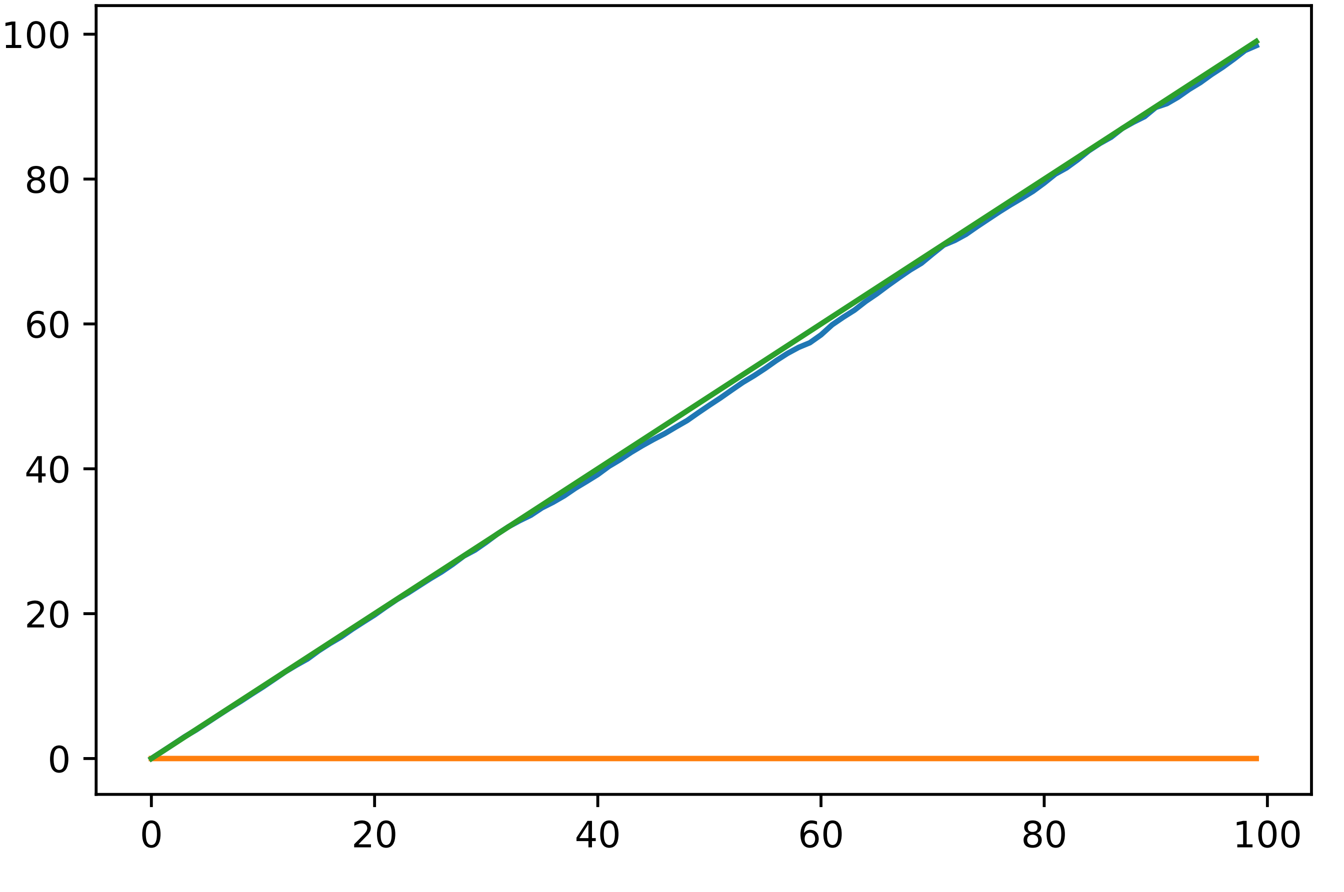


140
120
100
80
60
40
20
0
0
2000
4000
6000
8000
10000
25
0
−25
−50
−75
−100
−125
−150
−175
0
2000
4000
6000
8000
10000
80
60
40
20
0
0
2000
4000
6000
8000
10000
0
−50
−100
−150
−200
−250
0
2000
4000
6000
8000
10000

Diffusion law

```
import numpy as np
import pylab as plt

def d1rwens(n,nw): #of steps and walkers
    y=np.zeros(n) #array to which walkes displacement is added.
    for i in range(nw):
        x=np.zeros(n) #initial array set as 0
        for i in range(n-1):
            if np.random.randint(0,2)==0:
                x[i+1]=x[i]+1
            else:
                x[i+1]=x[i]-1
        y+=x**2
    y=y/nw
    #print(y)
    z=np.zeros(n)
    plt.plot(y)
    plt.plot(z)
    z1=np.arange(n)
    plt.plot(z1,z1) #straight line
    plt.show()
    #plt.savefig('1drwdiff_100_10000.pdf')
```


0
20
40
60
80
100
0
20
40
60
80
100


0
20
40
60
80
100
0
20
40
60
80
100

Exercise: Show that the distribution of the position of the walkers at any given instance tends for large time to the Normal distribution that is zero centered and with the appropriate variance which you will state.