Assignment 10: On Brownian Motion

PH1050 Computational Physics

By Aarya.G (EP23B025)

29th Nov 2023

Introduction

Brownian Motion is the random motion of suspended particals in a fluid. We simulate a simpler model using random walk. In random walk we take steps determined by a normally distributed random number. This idea of random walk gives rise to stochastic calculus which has huge applications in statistical physics and finance.

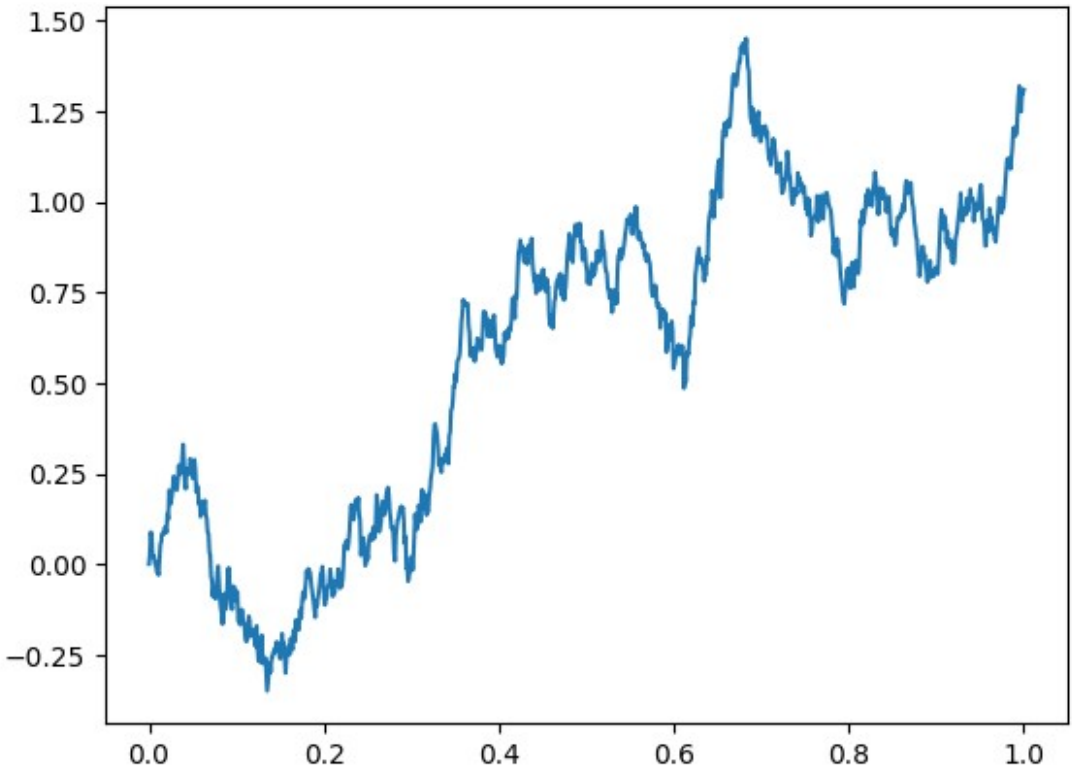
Aim

1. simulate random walk for one sample
2. simulate random walk for many number of samples
3. simulate random motion of partical in 2-D
4. simulate random motion of many particals in 2-D

simulate random walk for one sample

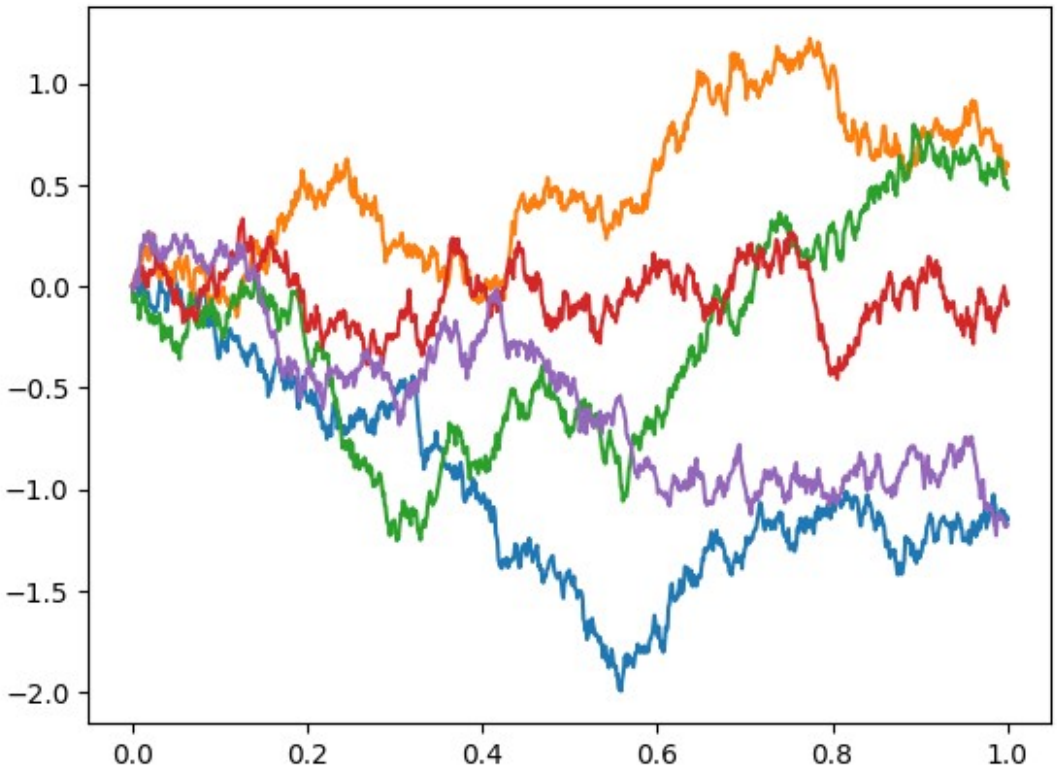
```
import numpy as np
import matplotlib.pyplot as plt
num_steps = 1000
final_T = 1
times = np.linspace(0, final_T, num_steps)
dt = times[1] - times[0]
dX = np.sqrt(dt) * np.random.normal(size=(num_steps-1,))
X0 = np.array(np.zeros(shape=(1)))
X = np.concatenate((X0, np.cumsum(dX)))
plt.plot(times,X)
```

```
[<matplotlib.lines.Line2D at 0x1dfef2c5a90>]
```



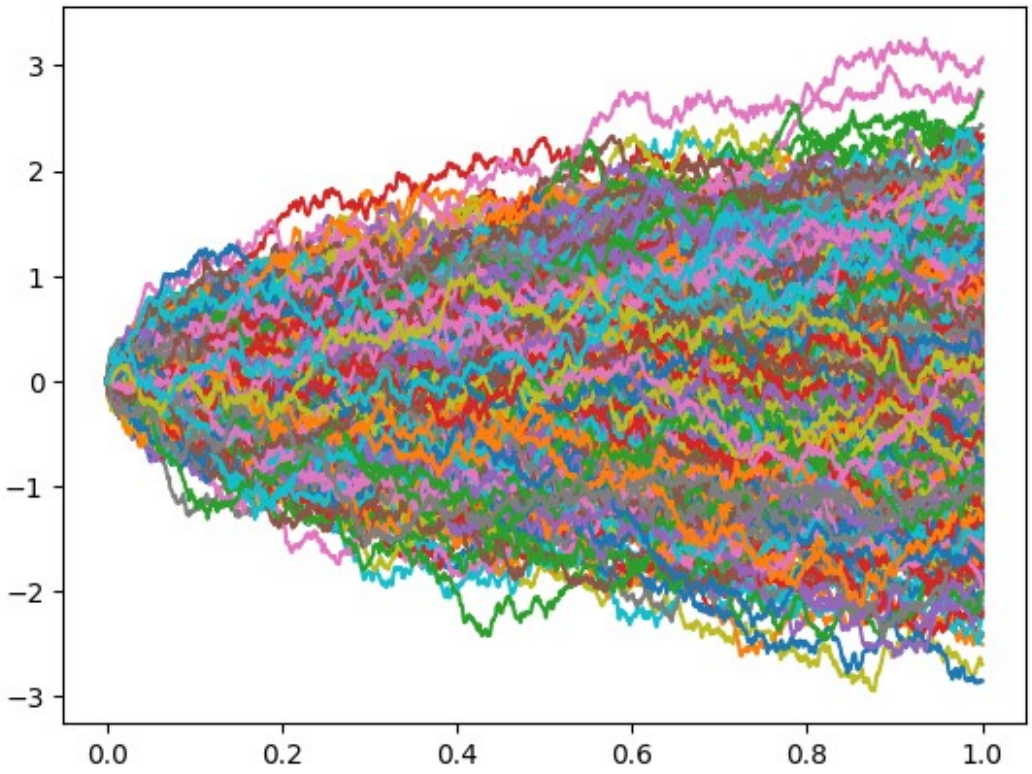
simulate random walk for many number of samples

```
n = 5
dX = np.sqrt(dt) * np.random.normal(size=(num_steps-1,n))
X0 = np.array(np.zeros(shape=(1,n)))
X = np.concatenate((X0, np.cumsum(dX, axis=0)) , axis=0)
plt.plot(times,X)
plt.show()
```



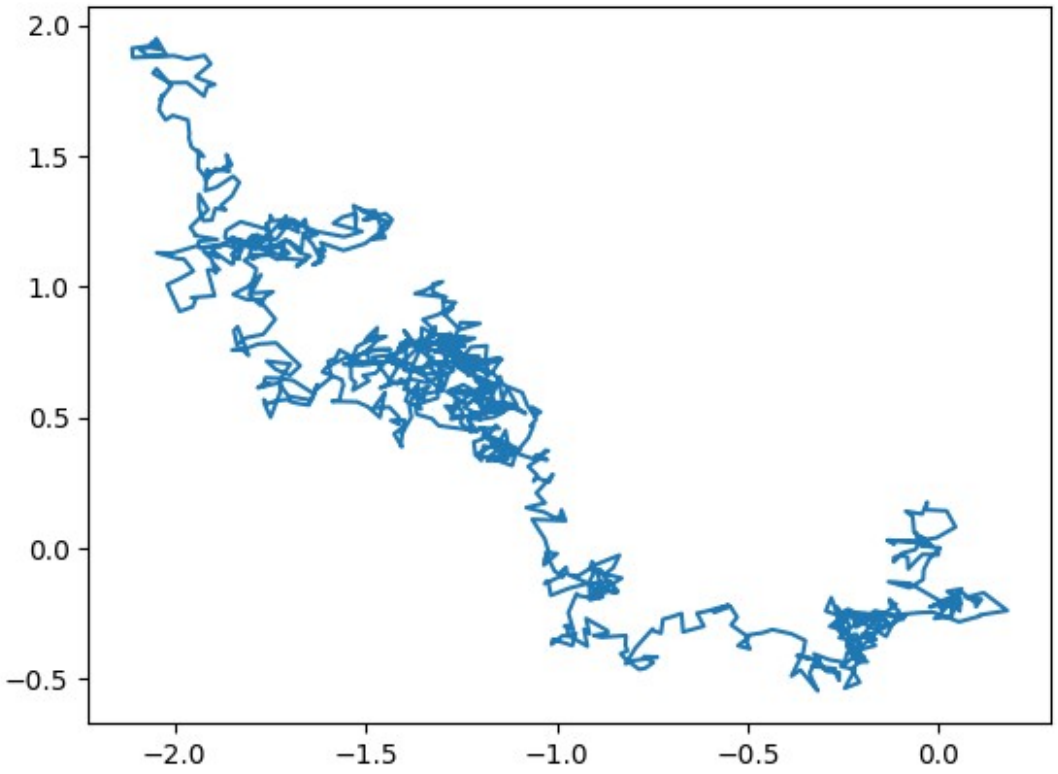
For Very large number of samples

```
n = 1000
dX = np.sqrt(dt) * np.random.normal(size=(num_steps-1,n))
X0 = np.array(np.zeros(shape=(1,n)))
X = np.concatenate((X0, np.cumsum(dX, axis=0)) , axis=0)
plt.plot(times,X)
plt.show()
```



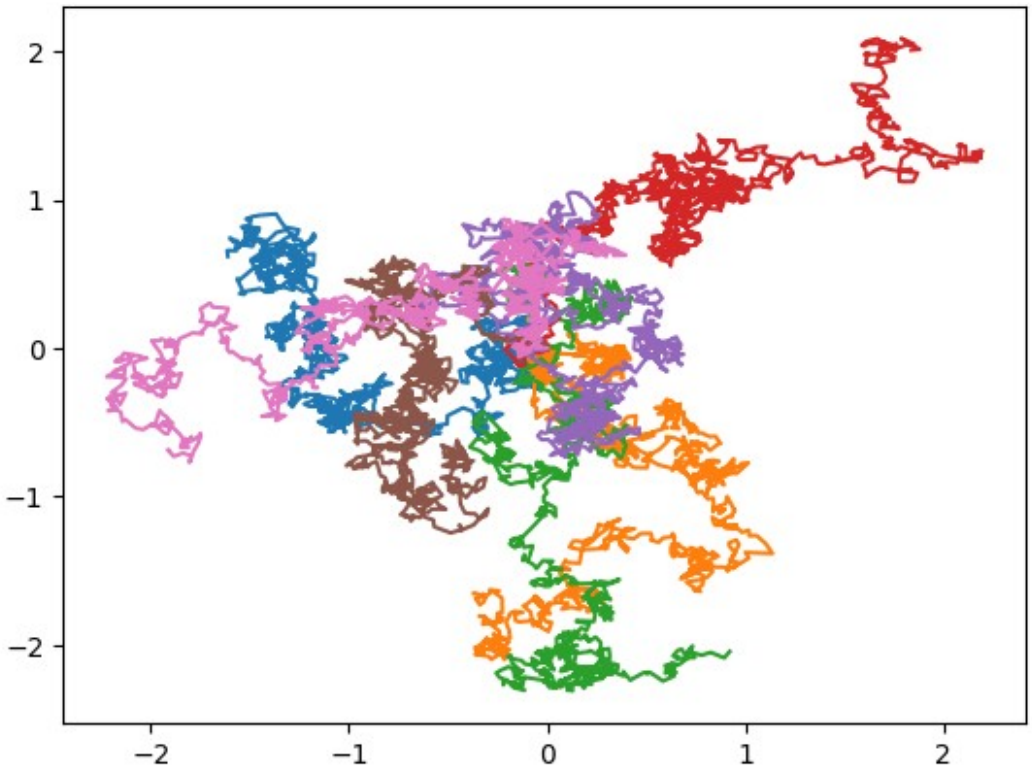
simulate random motion of partical in 2-D

```
d= 2
dX = np.sqrt(dt) * np.random.normal(size=(d,num_steps-1,))
X0 = np.array(np.zeros(shape=(d,1,)))
X = np.concatenate((X0, np.cumsum(dX, axis=1)) , axis=1)
plt.plot(X[0],X[1])
plt.show()
```



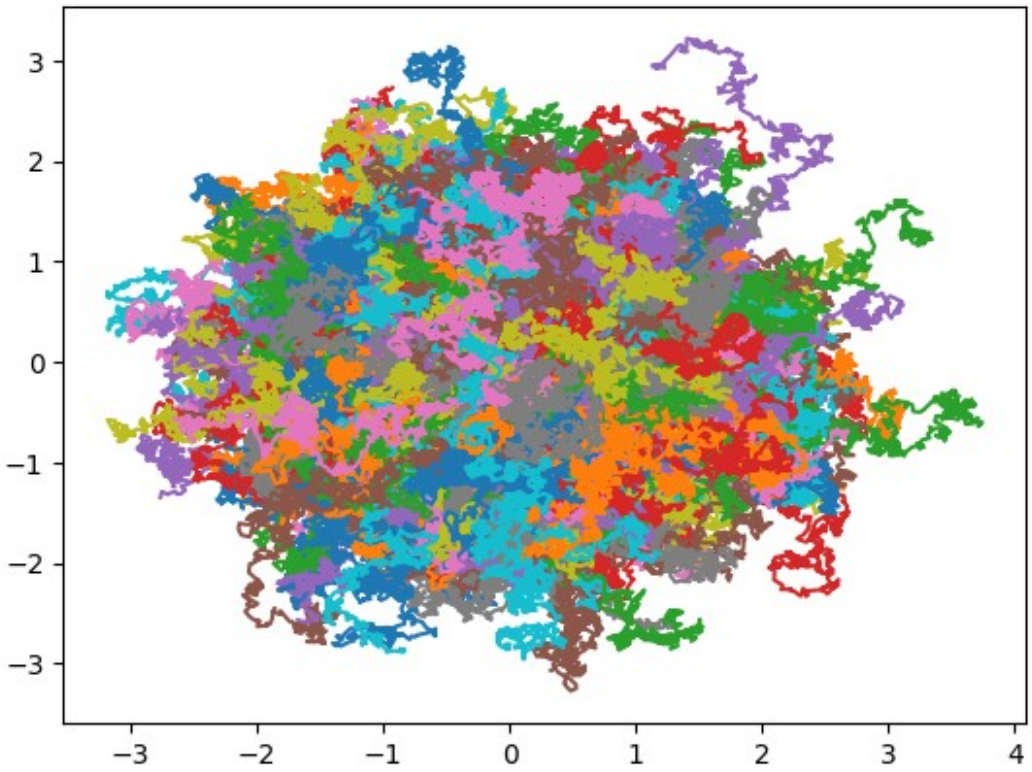
simulate random motion of many particals in 2-D

```
d= 2
n = 7
dX = np.sqrt(dt) * np.random.normal(size=(d,num_steps-1,n))
X0 = np.array(np.zeros(shape=(d,1,n)))
X = np.concatenate((X0, np.cumsum(dX, axis=1)) , axis=1)
plt.plot(X[0],X[1])
plt.show()
```



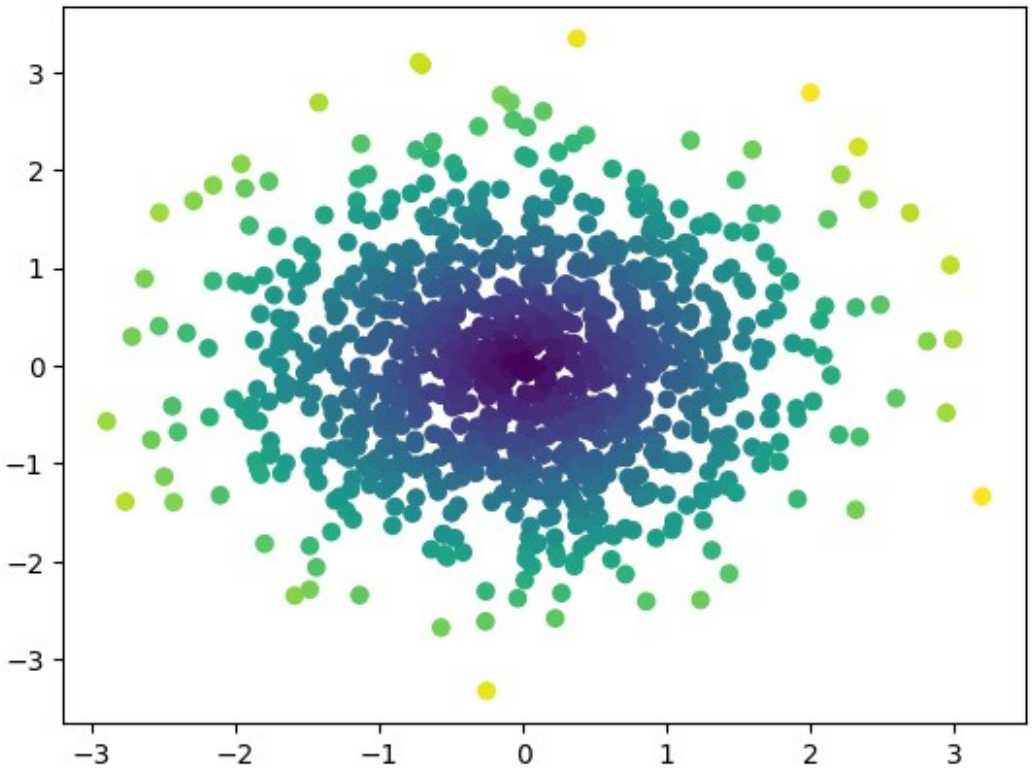
For a very large number of sample

```
d= 2
n = 1000
dX = np.sqrt(dt) * np.random.normal(size=(d,num_steps-1,n))
X0 = np.array(np.zeros(shape=(d,1,n)))
X = np.concatenate((X0, np.cumsum(dX, axis=1)) , axis=1)
plt.plot(X[0],X[1])
plt.show()
```



Plotting the final position of all the molecules starting from 0,0

```
d= 2
n = 1000
dX = np.sqrt(dt) * np.random.normal(size=(d,num_steps-1,n))
X0 = np.array(np.zeros(shape=(d,1,n)))
X = np.concatenate((X0, np.cumsum(dX, axis=1)) , axis=1)
plt.scatter(X[0][-1],X[1][-1], c = np.sqrt(np.power(X[0][-1],2) +
np.power(X[1][-1] , 2)))
plt.show()
```



Conclusion

1. For many partical system, the range of the walk tends to be under a certain curve with a few exceptions.
2. The Gas in 2-D tends to go in all direction without any bias towards a particular direction.