
Monte Carlo Simulation with Python

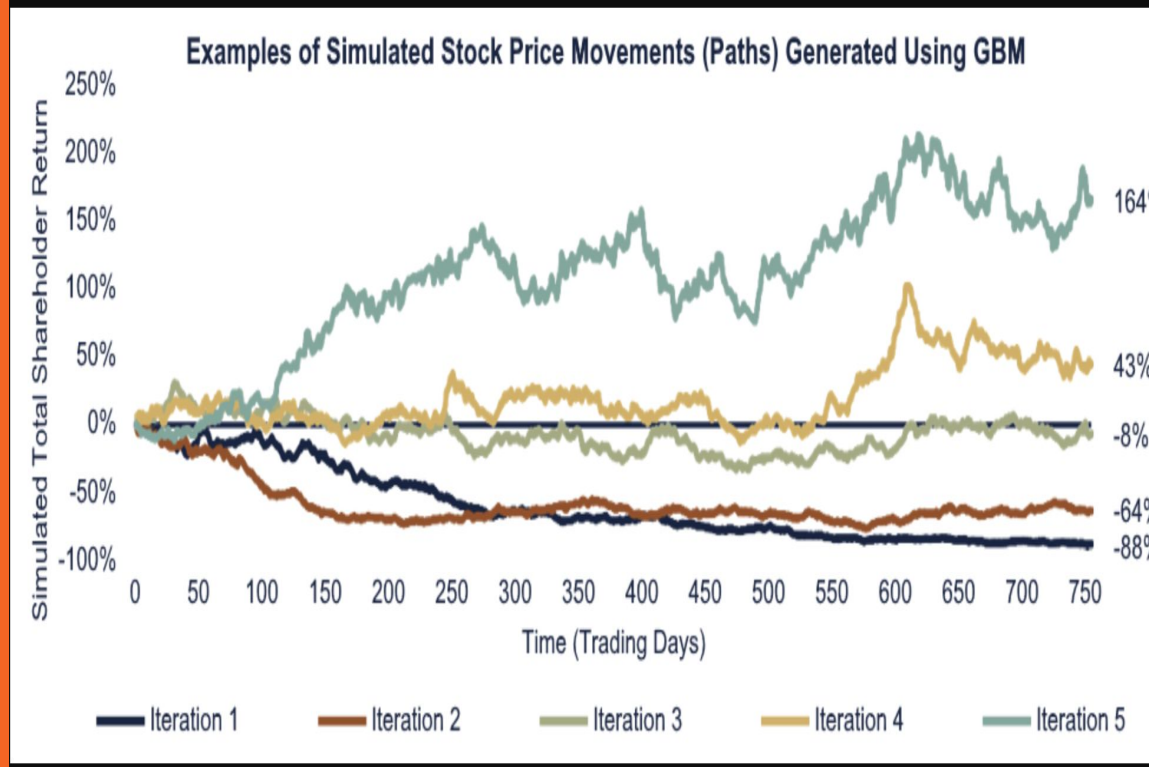
Tokhirjon Malikov

Professor : Dr. Ariyeh Maller



Context

- Some problems can not be expressed in analytical form and are difficult to define in a deterministic manner
- Monte Carlo allows you to run numerical experiments to see what happens on average over a large number of runs
- It is also called stochastic simulation. (Stochastic is a synonym for probabilistic.)





Monte Carlo simulations gets its name from the gambling that is totally depends on random outcomes. This process is called Random Number Generation that cannot be reasonably predicted better than by a random chance.

Monte Carlo Simulations: What are they?

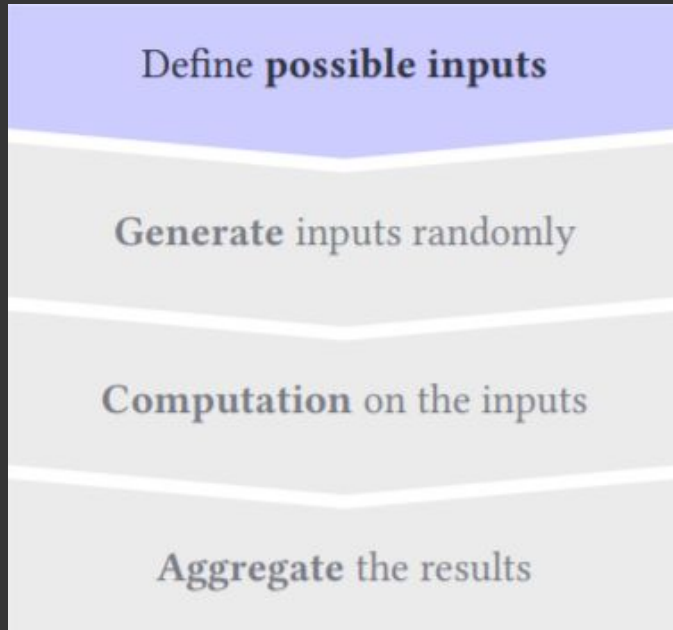
- Monte Carlo method: computational method using repeated random sampling to obtain numerical results.
- It is named after a Monte Carlo Casino located in administrative area in Monaco.
- Technique was invented by Polish American Physicist Stanislaw Ulam during the Manhattan Project (US nuclear bomb development)
- It is widely used in Science, Finance and Business Projects
However it used the most in Statistical Mechanics.

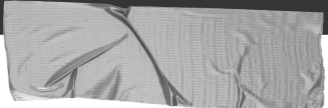


Importance Sampling and Statistical Mechanics

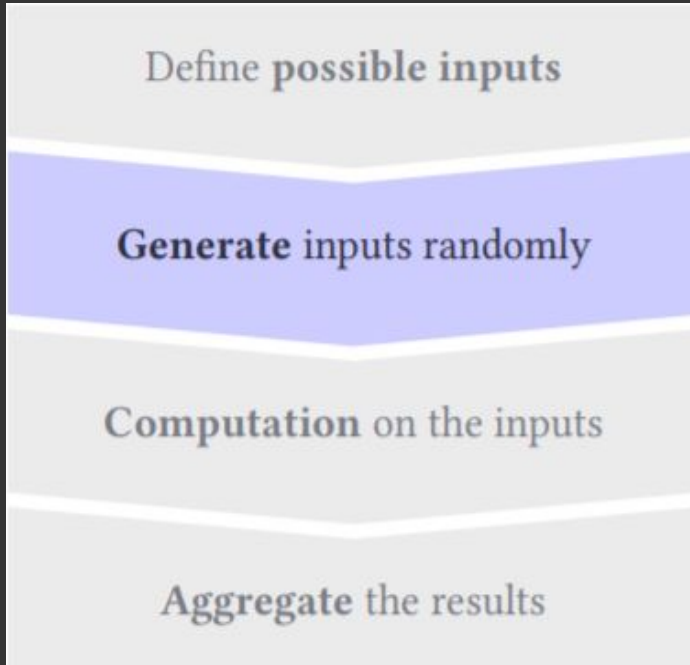
- Fundamental problem of statistical mechanics is to calculate the expected value of a quantity of interest in a physical system in thermal equilibrium
 - Normally we cannot simply evaluate the sum over states directly because the number of states is far too large.
-

Monte Carlo Simulation - Steps



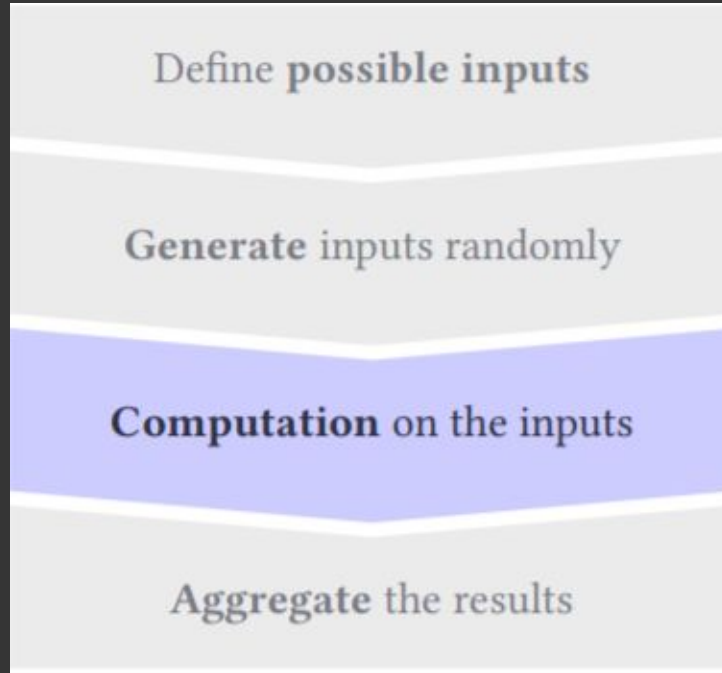
- 
- Define the domain of possible inputs.
 - Simulated problem should be similar to the real world problem whose behavior we wish to investigate.

Monte Carlo Simulation - Steps



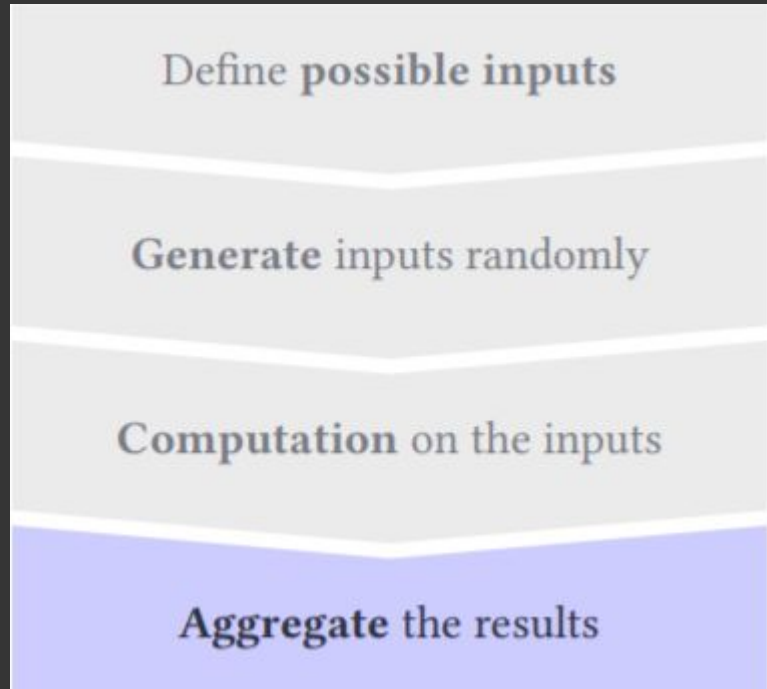
- Generate inputs randomly from a probability distribution over the domain
- Inputs should be generated so that their characteristics are similar to the real world problem we are trying to simulate.

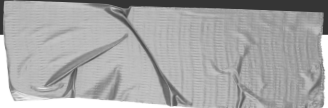
Monte Carlo Simulation - Steps



- The computation should be deterministic over all.

— Monte Carlo Simulation - Steps



- 
- Collect the results to obtain the output of the interest.
 - Typical outputs:
 - Histogram
 - Confidence interval
 - Summary statistics (mean, Sd and etc...)

```

@author: tmali
"""
from scipy import random
import numpy as np

a = 0
b = np.pi
N = 1000

ar = np.zeros(N)

for i in range(len(ar)):
    ar[i] = random.uniform(a,b)

integral = 0.0

def f(x):
    return np.sin(x)

for i in ar:
    integral += f(i)

ans = (b-a)/float(N)*integral

print ("The value calculated by monte carlo integration is {}".format(ans))

```

$$\int_0^{\pi} \sin(x) dx = 2$$

```

In [15]: runfile('C:/Users/tmali/Desktop/Physics 4150 Lab/untitled1.py', wdir='C:/Users/tmali/
Desktop/Physics 4150 Lab')
The value calculated by monte carlo integration is 1.9772250822507889.

```

```

In [16]: runfile('C:/Users/tmali/Desktop/Physics 4150 Lab/untitled1.py', wdir='C:/Users/tmali/
Desktop/Physics 4150 Lab')
The value calculated by monte carlo integration is 1.983714069026911.

```

```

In [17]: runfile('C:/Users/tmali/Desktop/Physics 4150 Lab/untitled1.py', wdir='C:/Users/tmali/
Desktop/Physics 4150 Lab')
The value calculated by monte carlo integration is 1.9970588209178433.

```

```

In [18]:

```

Geometric Brownian Motion

$$S(t) = s \exp \left(\left(\mu - \frac{1}{2} \sigma^2 \right) t + \sigma W(t) \right)$$

$$\log S(t) - \log s = \left(\mu - \frac{1}{2} \sigma^2 \right) t + \sigma W(t)$$

To find the expected asset price. A geometric Brownian Motion has been used.

Here $S(t)$ is Current day Price

S is the previous day price

μ is the expected return

$W(t)$ is a random value from normal distribution

For the drift we use the expected rate of return. In other words we use the rate that we expect the price to change each day. The average return rate based on volatility the average return rate based on volatility. the standard rate for return erosion is $1/2$ of the variance over time.

simulation_df - DataFrame

Index	990	991	992	993	994	995	996	997	998	999
234	185.957	24.1595	71.8116	779.068	61.7695	238.046	352.555	296.771	86.0618	158.895
235	187.776	25.485	75.4463	836.085	63.9297	236.193	340.494	283.897	85.4161	158.59
236	190.028	24.302	72.4944	836.025	63.0274	221.974	352.722	294.988	86.0787	154.254
237	198.719	25.4466	69.389	791.191	56.1072	216.939	321.309	297.962	86.6664	159.756
238	205.417	24.1544	68.5124	889.45	57.9078	211.423	336.503	293.241	85.9498	162.868
239	192.952	24.7801	67.0999	875.301	55.5742	221.269	340.812	274.629	89.833	163.952
240	192.326	25.3063	71.057	908.855	57.2297	224.122	360.053	275.846	94.3925	161.15
241	199.483	26.2584	70.3647	934.927	59.9999	233.251	325.29	275.262	89.8169	161.605
242	188.765	24.3472	64.3684	935.133	63.6161	215.925	314.389	271.584	99.9902	160.059
243	193.198	22.6116	66.1393	1006.71	63.9885	192.277	289.623	279.618	93.9593	161.337
244	183.931	22.7298	68.1097	961.983	68.225	202.091	276.642	295.621	90.5563	158.703
245	167.89	23.4407	72.4851	971.001	64.1295	188.214	290.725	288.326	82.4857	145.515
246	155.496	24.6145	73.5806	921.705	68.4982	190.778	298.737	305.546	82.2792	143.479
247	158.95	24.3221	65.819	936.059	68.0283	192.153	317.346	327.139	81.9701	144.397
248	160.365	25.6752	68.4518	944.268	61.9093	191.848	321.172	325.4	75.2711	134.277
249	171.689	23.7184	75.2552	920.621	59.7976	186.1	351.375	330.369	75.4996	125.11
250	155.207	24.8641	81.5845	874.276	63.1056	182.508	348.946	306.045	77.3721	119.297
251	163.046	24.7208	83.8471	873.196	67.8473	176.162	367.576	305.238	72.2431	115.606

Format

Resize



Background color



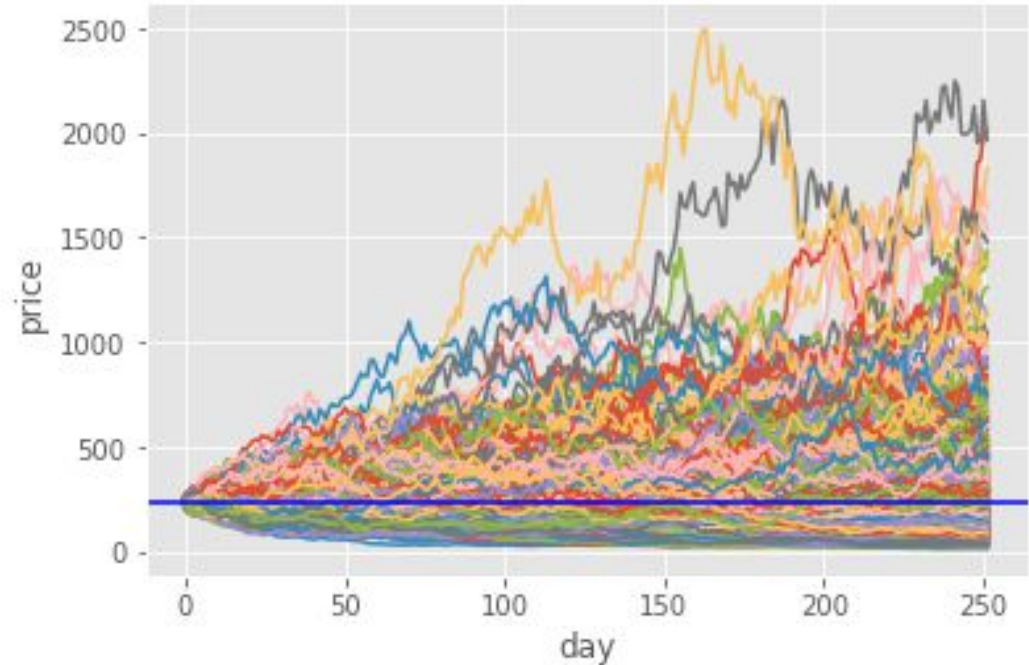
Column min/max

Save and Close

Close

Plot

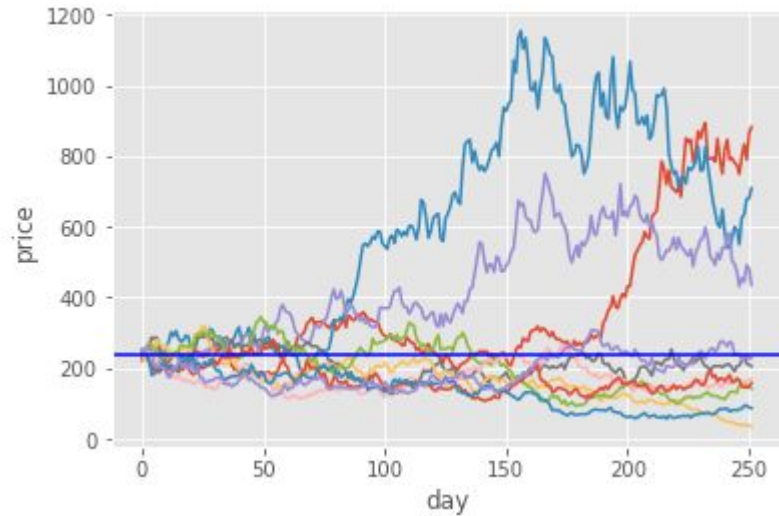
Monte Carlo Simulation: BA



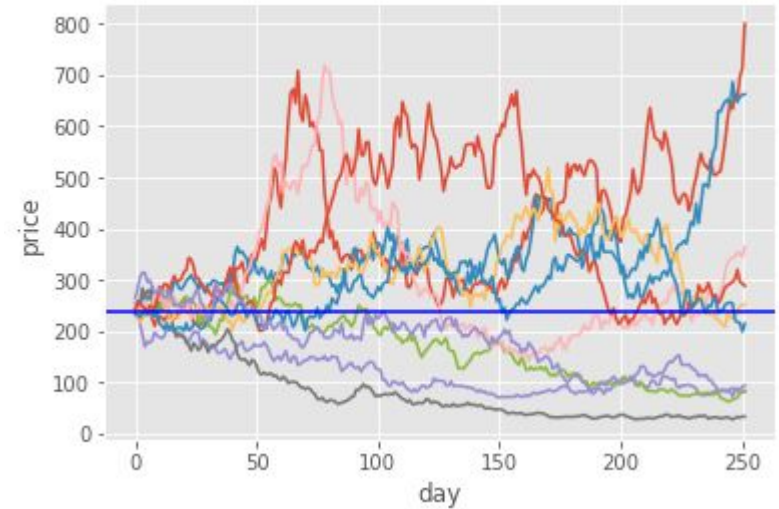
The blue line represents what should be by our model the current price of stock for each single trial.

Continuation but lowering number of trials from 1000 to 20, 10.

Monte Carlo Simulation: BA



Monte Carlo Simulation: BA



Closing

The law of large numbers describes what happens when performing the same experiment many times

After many trials, the average of the results should be close to the expected value.

This means for Monte Carlo simulation we can learn properties of a random variable (mean variance, etc) simply by simulating it over many trials



References

- <https://towardsdatascience.com/brownian-motion-with-python-9083ebc46ff0>
- <https://www.ibm.com/cloud/learn/monte-carlo-simulation#:~:text=The%20Monte%20Carlo%20Method%20was,to%20a%20game%20of%20roulette>.
- Computational Physics by Mark Newman, 2012 edition
- <https://pbpython.com/monte-carlo.html>
- <https://www.youtube.com/watch?v=7ESK5SaP-bc>
- <https://datascienceplus.com/how-to-apply-monte-carlo-simulation-to-forecast-stock-prices-using-python/>
- https://matplotlib.org/stable/gallery/style_sheets/style_sheets_reference.html