Monte Carlo Simulation

Rupesh Poudel

Seminar Paper:
Numerical Introductory Course

Lecturer: Prof. Dr. Brenda López Cabrera

February 25, 2022

Abstract

Monte Carlo Simulation uses random numbers to perform repeated sampling of a complex process. This paper generates uniformly distributed sequence of random numbers using Linear Congruential Generator and Lagged Fibonacci Generator, and converts the uniformly distributed random sequence of numbers into the standard normally distributed random sequence of numbers by Box-Müller Method, Inversion method and the Marsaglia method. These random sequence of numbers, along with the default random numbers generated from Python's Numpy library will be used to carry out the Monte Carlo Simulation to predict the price of the stock of the Apple Inc. 100 days into the future. Sensitivity of the simulation with respect to the change in random number generator and the number of trials are reported along with the computational time of the simulation. Mean Squared Error and Confidence Intervals are also reported. The Kernel Density Estimators is drawn for each variation.

Contents

1 Motivation

In a financial market, the information about the value a stock price of a particular company prevailing in the future is highly sought after. Investors can buy shares, even with high leverage, if the shares are expected to rise in the future, and offload their position in addition with short selling if the shares are expected to fall. Complex financial derivatives like futures and options, whose payoffs are based on the future price of the underlying, can allow investors to gain the profits with relatively less initial investment. Different private and institutional investors have their own future price prediction model, which they use to pick stocks to buy. The prediction power of the model determines the success of an investors. It is therefore important to find a suitable model that outperforms the market and hopefully the competitors. This paper will discuss about Monte Carlo Simulation, a statistical method of repeated sampling and aggregation, as an initial estimate.

Monte Carlo Simulation is a tool that can be used to solve problems which is too complex to be solved analytically. It can solve higher dimensional problems whose numerical evaluation is computationally engaging and tedious. Furthermore, when the underlying input for the estimation, forecasting, or some decision process is stochastic in nature, Monte Carlo Simulation can provide with a range of choice for that input.(Surname 2009)

Scientists working for The Manhattan Project, namely Stanislaw Ulam, John von Neumann, and Nicholas Metropolis, is credited for the invention of Monte Carlo Simulation. It all started with Ulam's inability to compute the probability of him winning Solitaire, a popular card game. Ulam hypothesised whether playing, or equivalently simulating the game for sufficiently large number of trials and counting the number of wins would be a reasonable alternative, when combinatorics involved in the game is too complex. The same approach could be replicated in numerous other sectors, including nuclear physics, where they had to "solve the problem of neutron diffusion in fissionable material." Neumann, after being suggested the approach, wrote a detailed, computer execution-able set of instructions for neutron diffusion be executed in ENIAC, a first generation computer. Metropolis suggested the name "Monte Carlo", referencing to Ulam's uncle and his gambling tendencies in casinos of Monte Carlo, Monaco. Metropolis (1987) outlines the aforementioned origin story and formalises the method of Monte Carlo Simulation.

Uniformly distributed (pseudo) random numbers are generated using an algorithm, in our previous story, the middle square digits. These random numbers are then converted to a suitable distribution that best matches the problem under consideration. The inverse of the cumulative distribution function of our desired distribution is used for the conversion. These transformed random numbers are used to carry out instances of deterministic computation. Appendix 1 shows an example of a coin toss experiment which can serve as a primer for the way in which Monte Carlo Simulation works.

2 Relation to existing literature or methods

Monte Carlo Simulation draws upon the Bernouli's law of large numbers. When an experiment is conducted multiple times, as the number of times the experiment is performed increases, the mean of the results from those experiments will converge to the true (theoretical) mean. A fair coin tossed countably infinite times will see the probability of each of its outcome, Heads or Tails, approach 0.5. A fair six-sided dice will witness the probability of each of its side to face upwards after a spin converge to 1/6.

3 Application for Finance and Statistics

3.1 Application for Statistics

Statistics is among the many fields that the use cases of the Monte Carlo Simulation is plentiful. Monte Carlo Simulation can be used to provide an efficient random estimate for the Hessian matrix. Yu (2016) uses automatic differentiation under monte carlo setting to compute estimate of the Hessian Matrix. The Hessian matrix can be used in Fisher Scoring, and subsequently the Fisher Information matrix. Das, Spall, Ghanem (2010) show the use of resampling algorithm, a Monte Carlo technique, for computing Fisher Information Matrix. The analytical determination of the Fisher Information Matrix in non linear models is difficult because of the intractable higher dimension integration and modelling requirement. Similarly, Bayesian Statistics is making use of Monte Carlo Simulation. Chen, Shao, Ibrahim (2012) develop advanced monte carlo methods by using posterior distribution's sample to compute posterior qualities, namely: marginal posterior densities, marginal likelihood, Bayesian Credible Intervals and so on. Monte Carlo Simulation is used in statistical interference as well. Hypothesis testing, comparison of variance of two estimators, evaluating higher dimension integrals and parametric bootstraps are some uses identified in Gentle (2009, Chapter 11).

3.2 Application for Finance

Hull (20xx) suggests Monte Carlo Simulation for option pricing. Path dependent options such as Asian options , and options with many stochastic variables can use Monte Carlo Simulation. Hull(20xx, pg 511) also suggests Monte Carlo Simulation to calculate the dollar change in portfolio in a day, or VaR. The value of derivative such as options can be calculated by sampling the random path of an underlying in a risk neutral world, calculating the payoff go the option, and run sufficiently high number of trials so that average of the sample payoff can be calculated to get the expected payoff in the risk neutral world. The option is now discounted expected payoff at the risk free rate. (Pg. 469). The performance of stop loss hedging can be seen by using Monte Carlo Simulation. In addition, an investor can analyse the default risk of a company or credit. The cash flow analysis of an uncertain project can be carried out. Instead of a point estimate, a confidence interval can be computed which guides the investors to make informed decisions. Furthermore, the evolution of a stock price, and consequently the investment portfolio of an investor can be modelled. This paper will carry out an analysis on the evaluation of the Apple share price in the future.

4 Formalization of the method

4.1 Random Numbers

Random numbers, or more accurately, a random sequence of numbers are essential for Monte Carlo Simulation. Without sufficiently randomized sequence of numbers, the estimates of the inputs will be biased. Biased inputs will lead to less trustworthy outputs. To get unbiased estimates, different attempts have been made to construct algorithms that produce uniformly distributed and independent random numbers. Random numbers must adhere to some conditions for it to be usable. Each numbers in the range must be equally likely, the random numbers must be continuous and not discrete. The mean and variance must neither be too high or too low. Similarly, there should not be significant auto-correlation between numbers.There must be no seeming pattern between numbers. A period of a sequence of random number reflects the number of terms in the sequence before the sequence repeats. An appropriate PRNG consists of higher period.
In this paper, once I generate the uniformly distributed random sequence of numbers, I test for their randomness. I plot the scatterplot of two consecutive random numbers to check whether there is a pattern. The points in scatterplot which is all over the graph shows that there is no apparent pattern. I also compute the variance covariance matrix of the random numbers to check whether there is an association in the random sequence of numbers generated from different algorithm/transformation. A kernel density estimator is also plotted to show the pdf of the estimated random sequence of numbers, be it uniform or standard normal.

4.2 Pros and Cons of the Method

Pros: Deterministically complex problems can be solved numerically For non-invertible functions, Monte Carlo can use reverse search to plot the graph of the inverse when possible. Monte Carlo method is a powerful tool for solving any problem with uncertainty involved.
Cons: Considerable computational time and resources are consumed. GIGO: if our input or parameters are garbage, our output will be garbage too. This includes the domain of input and the underlying probability distribution. Monte Carlo method does not factor in irrationality and behaviour aspects of say, finance and economics very well. Monte Carlo methods do not accurately predict tail events.

5 Data Description, Simulation Settings, Testing Results

The data used for the prediction of the share price of the Apple stock is extracted from "yfinance", a Python module for Yahoo Finance Database. The adjusted close price of Apple from December 10 2017 to February 6, 2022 is downloaded from Yahoo Finance. The plot of the aforementioned time series is given in Figure 1. From the time series, the natural logarithm of stock returns from day

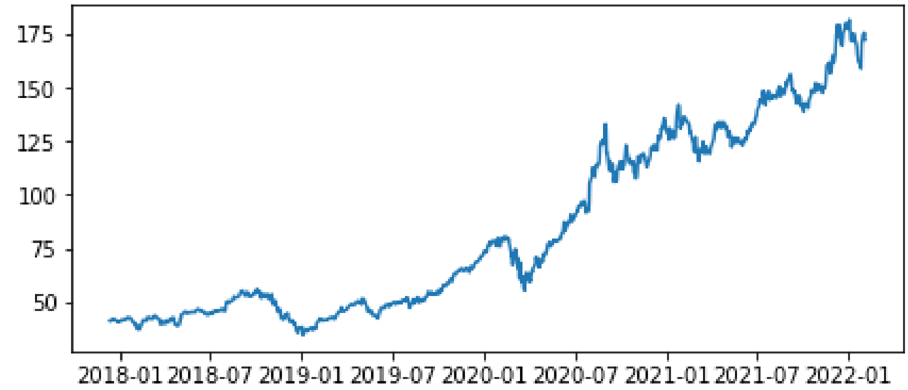

Figure 1: Apple Share Price

$t - 1$ to t is given calculated. The paper assumes that the natural logarithm of the share follows normal distribution with given mean and standard deviation. Linear Congruential Generator, and Lagged Fibonacci Generator, in addition to Numpy's default pseudo uniform random number generator generates uniformly distributed random sequence of numbers. These sequence is thereafter transformed into standard normal distribution using the Box-Müller Method, Inversion method, and Marsaglia method, details of which is found in Franke, Härdle, Hafner (2019, Chapter 6). I then calculated the mean and standard deviation of the log-returns calculated above, and use the estimates to convert the standard normally distributed random sequence of numbers into normally distributed random sequence of numbers.
Once an array of 10 million normally distributed random sequence of numbers, with the desired mean and dispersion, are generated, the array is then reshaped into a matrix of size [**100000,100**]. As we wish to predict the price path for 100 days ahead and run the Monte Carlo Simulation 100,000 times, our matrix has that specific dimension.For comparability and results reproduction, the matrix is stored under a variable so that the same random sequence of numbers can be used to check the sensibility of the final results on a reduced number of simulations. Different set of matrices representing different pseudo random number generator and consequently different conversion methods will be generated in order to check the sensitivity of the final results on the underlying pseudo random number generator.

6 Results of Empirical Analysis

Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua (Baddeley & Hitch 1974, p. 48). At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet. Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum.

Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet.

7 Conclusion

Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet. Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet.

8 Appendix

8.1 Appendix 1: The Coin Toss Experiment

Let there be two players: A B. A and B will participate in a coin toss game where a fair coin will be tossed 100 times. The outcome of each tosses will be recorded. Any "Heads" will entitle A a balance transfer of 1 from B and "Tails" will endow B with 1 from the bank balance of A. Alternatively, the winnings can be recorded and a single transfer to offset the payables can be carried out after 100 tosses. This is a setup of an ordinary random walk with two nodes. Franke, Härdle, Hafner (2019, Chapter 4) outlines the ordinary random walk in the following way:

$$X_t = X_0 + \sum_{k=1}^{t} Z_t, \quad t = 1, 2, 3...$$

where $X_0, Z_1, Z_2....$ are independent. $P(Z_k = 1) = p$, and $P(Z_k = -1) = 1 - p$ for all k . There is a source of uncertainty in this problem. The value of at the end of each coin toss will not be known beforehand. Monte Carlo will make use of random number it generates for each trial to suggest the value for Z_t. As the heads and tails are equally likely, any symmetric distribution is a suitable distribution, provided that the experimental design is unbiased towards either outcome. Using the uniformly generated random numbers for example, one can fairly allocate the random numbers if one allocates even numbers for head and odd numbers for tail when choosing for integers between 0 and 100. One can achieve the same results by allocating the numbers greater or equal to 0.50 as heads and the remaining half as tails when choosing the numbers in range [0,1). Once the heads and tails are determined by the use of uniformly distributed random numbers, can be computed. The coin toss experiment was settled without tossing a single coin. Such is the use of Monte Carlo Simulation.

8.2 Appendix 2: Middle Square Digits Algorithm

John von Neumann, in his first Monte Carlo code ever written, proposed the middle square digits algorithm for random number generation. In this algorithm, an arbitrary number of N digits is chosen. A 4 digit number 5555 for

illustration purposes will show the workings. The initial number from which a random number generator starts to generate subsequent random numbers is called the seed of the algorithm. Starting with the same seed will help replicate the sequence of pseudo random numbers for future use. The square of 5555 is 30858025. This is an eight digit number. The middle 4 digits, 8580 is the new random number and the process is repeated for the generation of subsequent random numbers.
edftests
anderson darling kolmogotov smirnov

References

Baddeley & Hitch 1974

BADDELEY, Alan D.; HITCH, Graham: Working memory. In: *Psychology of learning and motivation* (1974), pages 47–89

Surname 2009

SURNAME, First N.: *This is a test entry of type @ONLINE*. `http://today.slac.stanford.edu/images/2009/colloquium-web-collide.jpg`. Version: Juni 2009. – Last accessed 16 July 2013

References Baumgärtner, A., Binder, K., Hansen, J. P., Kalos, M. H., Kehr, K., Landau, D. P., ... Weis, J. J. (2013). Applications of the Monte Carlo method in statistical physics (Vol. 36). Springer Science Business Media.

Chen, M. H., Shao, Q. M., Ibrahim, J. G. (2012). Monte Carlo methods in Bayesian computation. Springer Science Business Media.

Das, S., Spall, J. C., Ghanem, R. (2010). Efficient Monte Carlo computation of Fisher information matrix using prior information. Computational Statistics Data Analysis, 54(2), 272-289.

Eckhardt, R., Ulam, S., Von Neumann, J. (1987). the Monte Carlo method. Los Alamos Science, 15, 131.

Franke, J., Härdle, W.K., Hafner, C.M. (2019). Statistics of Financial Markets: An Introduction. (5th edition). Springer.

Gentle, J. E. (2009). Monte carlo methods for statistical inference. In Computational Statistics (pp. 417-433). Springer, New York, NY.

Harrison, R. L. (2010, January). Introduction to monte carlo simulation. In AIP conference proceedings (Vol. 1204, No. 1, pp. 17-21). American Institute of Physics.

Matsumoto, M., Nishimura, T. (1998). Mersenne twister: a 623-dimensionally equidistributed uniform pseudo-random number generator. ACM Transactions on Modeling and Computer Simulation (TOMACS), 8(1), 3-30.

Metropolis, N. (1987). The beginning of the Monte Carlo method. Los Alamos Science, 15(584), 125-130.

Riffenburgh, R. H. (2006). Methods You Might Meet, But Not Every Day. Statistics in Medicine, 521-529.

Van Groenendaal, W. J., Kleijnen, J. P. (1997). On the assessment of economic risk: factorial design versus Monte Carlo methods. Reliability Engineering System Safety, 57(1), 91-102.

Yu, X. (2016). Efficient Computation of Hessian Matrices in a Monte Carlo Setting using Automatic Differentiation. [Master's Thesis, University of Waterloo]. https://uwaterloo.ca/computational-mathematics/sites/ca.computational-mathematics/files/uploads/files/x