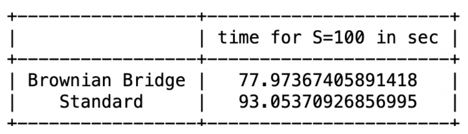
AMS 514 Project 1

Zhiwei Lin

October 2023

Introduction

The Longstaff-Schwartz method provides a flexible way to value American options by incorporating the dynamic decision-making aspect of when to exercise the option. It combines Monte Carlo simulation with polynomial regression to handle the high-dimensional problem of valuing American options. In Gustafsson's paper, he improved the algorithm by incorporating Laguerre polynomials as basis functions in the least squares polynomial regression. Also, he introduced the Brownian bridge, a method of generating of Geometric Brownian Motion (GBM) from time T to time 0. This approach improve the computation efficiency in terms of memory usage. In comparing to the standard Least Squares Monte Carlo (LSM) method, it reduces memory usage by orders of magnitude.



```
+-----------------+------------------------+
|                 | time for S=100 in sec |
+-----------------+------------------------+
| Brownian Bridge |   77.97367405891418   |
|    Standard     |   93.05370926856995   |
+-----------------+------------------------+
```

Figure 1: S0 = 100 K = 100 T = 1 r = 0.03 sigma = 0.15 N = 200 M = 10**6 k = 5 dividend = 0 option_type=put

Q1

I agree with Gustafsson's choice of using Laguerre polynomials as basis functions in least squares polynomial regression. Reason is that Laguerre has some properties that are very useful in computation - orthogonality and convergence. The orthogonality simplifies regression computations and reduces basis function correlation, contributing to numerical stability. Convergence property offer stability and accuracy in regression results.

I also agree with Gustafsson's choice of using Brownian Bridge to simulate stock paths. It is computational efficiency, which you don't have to store the realized paths in memory while maintains the necessary level of detail for accurate modeling.

However, I don't think least square Monte Carlo can perfectly replicate the real American option price movement. First, it is based on geometric Brownian motion, this is common assumption for stock movement; however, in real world, many stock movement do not follow GBM. In addition, LSMC may face challenges in accurately capturing the nonlinear nature of American options.

Q2

I implemented my code in Python instead of MATLAB. It's known that Python is slower than MATLAB. Therefore, I did some optimization to my code. First of all, Python defaults to using SVD for solving least squares, this method would solve the system of the equations in a slow computation. To address this, I replaced SVD with QR decomposition on my input matrix X, which speed up the code. Additionally, I used NumPy to further optimize execution speed and minimize redundant calculations.

Furthermore, I extended the functionality by introducing two additional arguments - option_type and dividend, to complement the function. The original code only accommodated put options, and there is no consideration when dividend is exist.

$$S_{t+1} = S_t \exp\left((\mu - \delta - \frac{\sigma^2}{2})\Delta t + \sigma\sqrt{\Delta t} Z_{t+1} \right) \tag{1}$$

Where δ is dividend rate. I incorporated this to my code to ensure my code work for both put and call and stocks with dividend

Q3

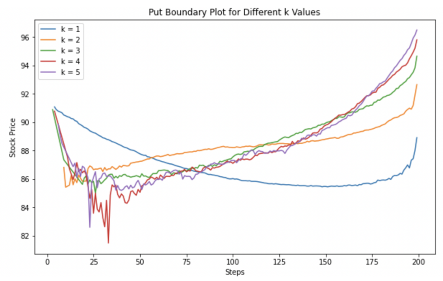


Figure 2: S0 = 100, K = 100, T = 1, r = 0.03, sigma = 0.15, N = 200, M = 10**6, dividend=0, option_type=put

This is my put option decision boundary. As can be seen, increasing K will make decision boundary close to "true" decision boundary in Longstaff-Schwartz method. However, this "true" decision boundary is not likely to be real decision boundary in real market because it's based on some assumptions such as stock price is geometric Brownian motion and linear relationship.

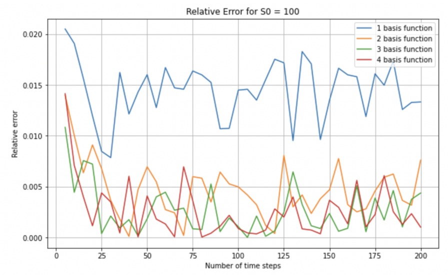


Figure 3: S0 = 100, K = 100, T = 1, r = 0.03, sigma = 0.15, N = 200, M = 10**5, dividend=0, option_type=put

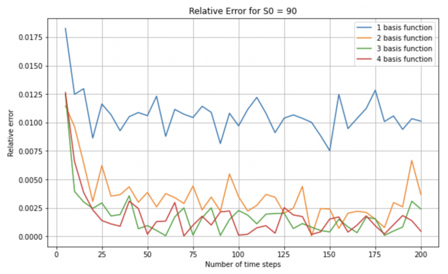


Figure 4: S0 = 90, K = 100, T = 1, r = 0.03, sigma = 0.15, N = 200, M = 10^5, dividend=0, option_type=put

These are the relative errors for put options with $S_0 = 100$ and $S_0 = 90$. As observed, $K = 1$ performs the worst compared to others. Larger values of K are expected to yield better results. Unfortunately, my plots cannot differentiate the performance for $k > 1$ due to the limited sample size ($M = 10^5$). If M is increased, larger values of k should have better performance.

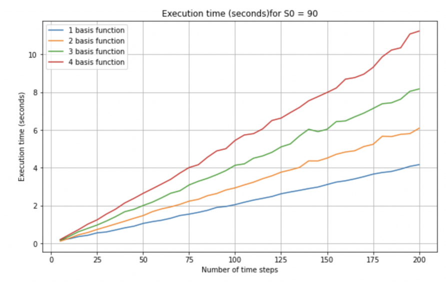


Figure 5: S0 = 90, K = 100, T = 1, r = 0.03, sigma = 0.15, N = 200, M = 10**5, dividend=0, option_type=put

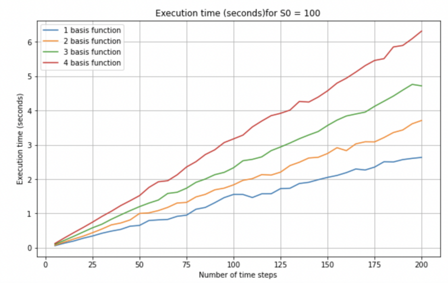


Figure 6: S0 = 100, K = 100, T = 1, r = 0.03, sigma = 0.15, N = 200, M = 10**5, dividend=0, option_type=put

These are the execution speed for put options with $S_0 = 100$ and $S_0 = 90$. As observed, larger K take more time to execute; however, they would have better result.

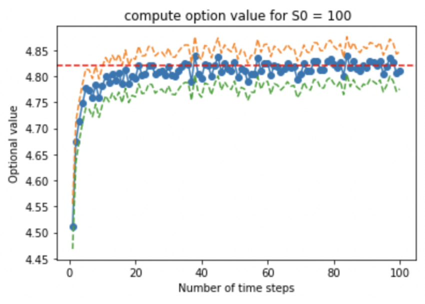


Figure 7: S0 = 90 K = 100 T = 1 r = 0.03 sigma = 0.15 M = 10**5 option_type=put k = 4

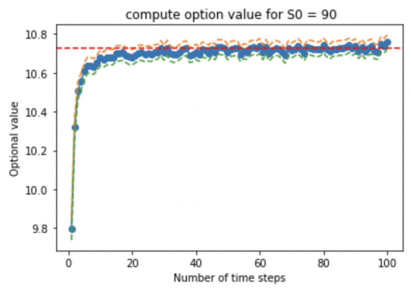


Figure 8: S0 = 100 K = 100 T = 1 r = 0.03 sigma = 0.15 M = 10**5 option_type=put k = 4

These my computed option value for S=100 and S=90 with confidence interval 0.95. They are pretty close to the true value.

1 Q4

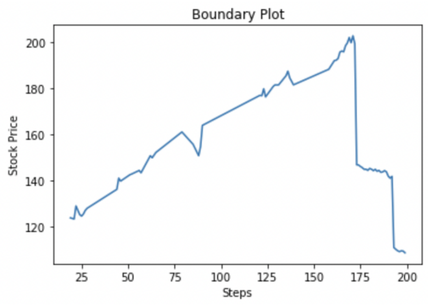


Figure 9: S0 = 100 K = 100 T = 1 r = 0.03 sigma = 0.15 N = 200 M = 10**6 k = 5 option_type=call dividend = 0

This is my decision boundary for call options. As can be seen, the algorithm mistakenly exercises some call options at early time. This shows a limitation of the Longstaff-Schwartz method. In theory, no call option should be exercised early, and thus, the decision boundary for call options should ideally be a vertical line at maturity.

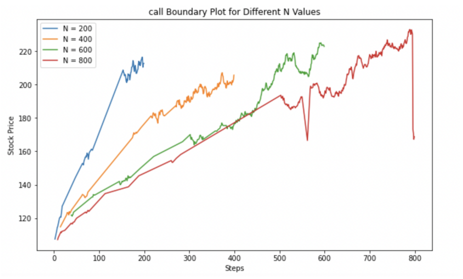
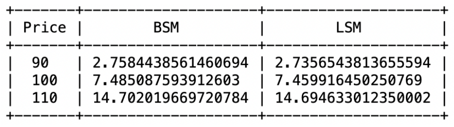


Figure 10: S0 = 100 K = 100 T = 1 r = 0.03 sigma = 0.15 N = [range(200,1000,200)] M = 10**6 k = 5 option$_{t}ype = calldividend = 0$

This shows the convergence of the call option as it runs with different values of N. As N becomes larger, the plot becomes a smooth linear line. However, at the end of the line, there is a straight drop-down, forming a vertical line



```
+-------+--------------------+--------------------+
| Price |        BSM         |        LSM         |
+-------+--------------------+--------------------+
|  90   | 2.7584438561460694 | 2.7356543813655594 |
|  100  | 7.485087593912603  | 7.459916450250769  |
|  110  | 14.702019669720784 | 14.694633012350002 |
+-------+--------------------+--------------------+
```

Figure 11: American Call option value with parameters K = 100 T = 1 r = 0.03 sigma = 0.15 N = 200 M = 10**6 k = 5 option_type=call dividend = 0

As can be seen, LSM agrees with BSM, demonstrating that even though LSM exercises some call options early, it can still price the American call option similarly to the Black-Scholes model