Advanced Model in Financial Engineering Project Report

SONG Zihao 1155154519 SEEM 15/05/2021

1. Introduction

An American option grants the holder the right to select the time at which to exercise the option. It makes American options more difficult to price than European as it requires the solving of an optimal stopping problem. American options can be found in all major financial markets including the equity, commodity, credit, swap and foreign exchange markets. In this project, we will price American options using the Longstaff-Schwartz algorithm, the Regression method, a powerful approach developed in 2001 and based on the use of least-squares to estimate the conditional expected payoff to the option holder from continuation. Inside we consider two stock assets modeled by correlated geometric Brownian motions contained two-dimensional stochastic parts, which would be simulated via Monte Carlo method. Also, we will try as much as possible in obtaining the neural network-based algorithm results for high-dimension case, introduced by Sebastian Becker, Patrick Cheridito and Arnulf Jentzen (2019), inside we consider 50 stock assets modeled by correlated geometric B.M. contained 50 stochastic parts, and use the Cholesky factorization to compute the correlation matrix in Monte Carlo simulation. Eventually, followed the alternative High-Biased estimate (duality method) developed by Rogers (2002), and Haugh and Kogan (2004), we may check the upper bound of the simulated option price and show our results are in a good quality, for neural network-based method, we need to do more.

We then present the Monte Carlo simulation, Longstaff-Schwartz algorithm, duality methods and neural network-based algorithm in section 2, then provide the simulated Bermudan max-call American option results, the lower and upper bounds for each pricing algorithm in section 3. In the last section, we discuss the numerical results and give the conclusion and references.

2. Methods presentation

2.1 The whole question formulation and Monte Carlo

We consider optimal stopping problems of the form $sup_{\tau}\mathbb{E}[h(\tau, X_{\tau})]$, where $h(t, X_t, Y_t) = e^{-r*t} * \max((\max(X_t, Y_t) - K), 0)$.

Thus, it is quite important to simulate each kind of path driven by stochastic diffusion dX_t, and in this project, we choose the normal correlated geometric Brownian motions, which diffusion is shown as:

$$dS_t^i = rS_t^i dt + \sigma S_t^i dW_t^{\mathbb{Q},i};\ S_0^i = S_0.$$

where the $\mathbb{Q}$ represents the risk-neutral probability measure and each of $W_t^{\mathbb{Q},i}$ has correlation ρ. Specifically, there are two versions in this project:

①For the Longstaff Schwartz Algorithm with Regression Method, we obtain two assets in two dimensional, which is:

$$S_0^1 = S_0^2 = S_0;\ dS_t^1 = rS_t^1 dt + \sigma S_t^1 dW_t^{\mathbb{Q},1};$$

$$dS_t^2 = rS_t^2 dt + \sigma S_t^2 dW_t^{\mathbb{Q},2} = rS_t^2 dt + \sigma S_t^2 \left(\rho dW_t^{\mathbb{Q},1} + \sqrt{1-\rho^2} dZ_t\right).$$

where Z_t is a random variable with standard normal distribution.

②For the neural network-based approximation method, we use directly 50 assets in 50 dimensions, each of $W_t^{\mathbb{Q},i}, i = 1,2,\dots,50$ has correlation matrix as the form:

$$\begin{matrix} 1 & 0.1 & \cdots & 0.1 & 0.1 \\ 0.1 & 1 & & 0.1 & 0.1 \\ & & \cdots & & \\ 0.1 & 0.1 & \cdots & 1 & 0.1 \\ 0.1 & 0.1 & & 0.1 & 1 \end{matrix}$$

So, the Monte Carlo simulation algorithm for each situation follows below

Algorithm (Monte Carlo for 2-dimensional):

```
1. Set $S_0^1 = S_0^2 = S_0$ as the initial asset price;
2. Generate $X_{1,N}^1, Z_{1,N}^1$ i.i.d. from $N(0,1)$ and form $W_{1,N}^1 = \sqrt{dt} * X_{1,N}^1$ and
   $W_{1,N}^2 = \sqrt{dt} * (\rho * X_{1,N}^1 + \sqrt{1-\rho^2} * Z_{1,N}^1)$;
3. Generate each path each time step by each time step:
   $S_{t_j}^i = S_0^i * \exp((r - 0.5 * \sigma * \sigma) * dt + \sigma * W_{t_{j-1},t_j}^i)$.
```

Algorithm (Monte Carlo for high-dimensional):

```
1. Set $S_0^i = S_0$ as the initial asset price;
2. Generate the Cholesky factorization via $L = chol('correlation\ matrix','lower')$ function and each $W_{1,N}^i = \sqrt{dt} * L * randn(i,N)$;
3. Generate each path each time step by each time step:
   $S_{t_j}^i = S_0^i * \exp((r - 0.5 * \sigma * \sigma) * dt + \sigma * W_{t_{j-1},t_j}^i)$.
```

4. Generate K times and gather the corresponding paths for ith asset (so we have simulated M assets with K paths and N time periods each).

2.2 Longstaff Schwartz Algorithm with Regression method for 2-D

Working with a finite-exercise Markovian formulation of American option pricing lends itself to a characterization of its value through dynamic programming. Let $V_i(x)$ denote the value of the option at t_i given the underlying asset price $S_{t_i}^1 = x$ and $S_{t_i}^2 = y$, assuming the option has not previously been exercised, then, with a fixed set of exercise opportunities $0 = t_0 < t_1 < t_2 < \cdots < t_m = T$, we obtain:

$$V_m(x, y) = h_m(x, y);$$

$$V_{i-1}(x, y) = \max\{\, h_{i-1}(x, y), \mathbb{E}[V_i(S_{t_i}^1, S_{t_i}^2) | S_{t_{i-1}}^1 = x, S_{t_{i-1}}^2 = y]\,\}$$

So, we need to compute the simulation of continuation value of an American option as the value of holding rather than exercising the option, i.e., the expectation part.

Introduced by Longstaff, F. A., and Schwartz, E. S. in 2001, regression-based method posits an expression for the continuation value of the form:

$$\mathbb{E}[V_i(S_{t_i}^1, S_{t_i}^2) | S_{t_{i-1}}^1 = x, S_{t_{i-1}}^2 = y] = \sum_{r=1}^{M} \beta_{ir} \psi_r(x, y)$$

for some basic functions ψ_r and constants $\beta_{ir}, r = 1, \dots, M$. When we have observations $(S_{t_i,j}^k, V_{i+1}(S_{t_{i+1},j}^1, S_{t_{i+1},j}^2)), k = 1,2;\ j = 1, \dots, b$, where b is the number of paths we simulate in Monte Carlo part, the standard least square arguments lead to an estimate for $\widehat{\beta_i} := \hat{B}_{\psi}^{-1} \hat{B}_{\psi V}$, where $\hat{B}_{\psi}^{-1}$ is the inverse of a $M \times M$ matrix with (q,r) element $\frac{1}{b}\sum_{j=1}^{b}\psi_q(S_{t_i,j}^1, S_{t_i,j}^2)\psi_r(S_{t_i,j}^1, S_{t_i,j}^2)$, and $\hat{B}_{\psi V}$ is the M-vector with r-th entry $\frac{1}{b}\sum_{j=1}^{b}\psi_r(S_{t_i,j}^1, S_{t_i,j}^2) * V_{i+1}$.

Thus, the choice of basic functions is a significant part of this algorithm, during my project, I will choose two different types of basic functions:

① ***the power function for 2-D***:

$\psi_r(x,y) = 1, r = 1; x, r = 2; y, r = 3; x^2, r = 4; y^2, r = 5; x^3, r = 6; xy, r = 7; x^2y, r = 8; xy^2, r = 9; x^2y^2, r = 10.$

② ***the (weighted) Laguerre polynomials for 2-D:***

$$\psi_r(x,y) = 1, r = 1; -xy + 1, r = 2;\ 0.5(x^2y^2 - 4xy + 2), r = 3; \frac{1}{6}(-x^3y^3 + 9x^2y^2 - 18xy + 6), r = 4$$

Then, the Longstaff Schwartz Algorithm I implement is below:

Algorithm (Longstaff Schwartz Algorithm for 2-D):

---

1. Simulate b independent paths via Monte Carlo method;
2. At terminal nodes, set $\hat{V}_m(x,y) = h_m(S_{t_m,j}^1, S_{t_m,j}^2), j = 1, \dots, b;$
3. Apply backward induction for $i = m-1, \dots, 1$, and calculate $\hat{\beta}_i$ and the continuation value $\hat{\beta}_i{}'\psi(S_{t_i,j}^1, S_{t_i,j}^2)$, and compute the $\hat{V}_{i,j} = \begin{cases} h_i(S_{t_i,j}^1, S_{t_i,j}^2), & if\ \ \hat{\beta}_i{}'\psi(S_{t_i,j}^1, S_{t_i,j}^2) < h_i(S_{t_i,j}^1, S_{t_i,j}^2) \\ \hat{V}_{i+1,j}, & otherwise \end{cases}$.
4. $\hat{V} = \frac{1}{b} * \sum_{j=1}^{b}\hat{V}_1$ is the final price.

This project chooses $r = 5\%;\ \sigma = 0.2;\ \rho = 0.1;\ T = 1;\ K = 11;\ S_0^1 = 10;\ S_0^2 = 10;\ b = 100;\ m = 365$. And I decide to test the algorithm for $G = 10$ times and eventually choose the mean of all results as the price.

Also, we change the K, S_0^i, b, m, G and different basic functions to test the accuracy and efficiency of this algorithms and give the table in next section.

2.3 Neural Network-based approximation

Algorithm (Neural Network-based approximation): ***<u>(in my own words)</u>***

1. Generate M assets with K paths and N time periods by Monte Carlo method in high-dimensional;
2. For one path of each asset, from $n = 1, \dots, N$, set $Y_n = (X_n^1, \dots, X_n^M, h(n, X_n^1, \dots, X_n^M))$ be a $M + 1$-dimensional vector as the input training element into the neural network (choose $I = 2$; $q_1 = q_2 = M + 40$): $f^{\theta_n} = \mathbb{I}_{[0,+\infty)}(\text{"}\frac{1}{(1+e^{-x})}\ for\ F^{\theta_n}\ below\text{"}) \circ a_2^{\theta_n} \circ \phi_{q_1}^{\theta_n} \circ a_1^{\theta_n}$ where $a_i^{\theta_n}$ is the affine function $A_i^{\theta_n} x + b_i^{\theta_n}$, and $\phi_{q_1}^{\theta_n}$ is the component-wise ReLU activation function.
3. After training, we implement the mini-batch gradient ascent with Xavier initialization backward from $n = N - 1, \dots, 1$, on
$$r_n^{\theta_n} = h(n, X_n^1, \dots, X_n^M) F^{\theta_n} + h(l_{n+1}, X_{l_{n+1}}^1, \dots, X_{l_{n+1}}^M)(1 - F^{\theta_n})$$
where $l_{n+1} = \begin{cases} N, & if\ n = N-1 \\ l_{n+1}(X_{n+1}, .., X_{N-1}) : \mathbb{R}^{N-n-1} \to \{n+1, \dots, N-1\}, & else \end{cases}$
to get the optimized each $\{\theta_n, n = 1, \dots, N-1\}$ maximizing $r_n^{\theta_n}$;
4. Plug $\{\theta_n, n = 1, \dots, N-1\}$ inside the optimal stopping time formula: $\tau = \sum_{n=1}^{N-1} n f^{\theta_n}(Y_n) \prod_{j=1}^{n} (1 - f^{\theta_j}(Y_j))$ and obtain the option value.

I choose the same stock parameters as above and $M = 50, N = 365, K = 10$ to do the neural network approximation.

2.4Duality method checking simulation quality

①***For Longstaff Schwartz Algorithm***, we follow Haugh and Kogan stated in 2004, via the Doob-Meyer decomposition of a super-martingale, the duality approach (i.e., add a martingale in the price formula after Jensen's inequality as the penalty function) to find the upper bound of our simulation of price.

Algorithm (Duality Method for 2-D Longstaff Schwartz Algorithm):

```
1. Generate $b$ paths of the stock prices at each timestep $\{S^n_{t_{i,j}}\}, i = 1, \dots, m; j = 1, \dots, b; n = 1,2$, and implement Longstaff Schwartz Algorithm to obtain each $\hat{V}_{i,j}$;
2. For 2 assets, each path $j = 1, \dots, b$, from $i = 1, \dots, m$, generate $P$ mini-paths for each timestep $t_i$, and for each $(t_i, j)$ element pairs, use Longstaff Schwartz Algorithm to obtain each $\hat{V}^P{}_{(t_i,j)}$;
3. Set $\hat{M}^j = \sum_{i=1}^{m}(\hat{V}_{i,j} - \hat{V}^P{}_{(t_i,j)})$ for each path $j = 1, \dots, b$, and calculate $max_i\ (h_i(S^1_{t_{i,j}}, S^2_{t_{i,j}}) - \hat{M}^j)$, thus, upper bound is the mean over $b$ paths.
```

In my project, similarly as in lecture, the lower bound is the $(min\hat{V})$ we simulated in Longstaff Schwartz Algorithm, and I decide to experience the upper bound under $r = 5\%;\ \sigma = 0.2;\ \rho = 0.1;\ T = 1;\ K = 11;\ S^1_0 = 10;\ S^2_0 = 10;\ b = 10;\ P = 3;\ m = 30;\ m_0 = 3,$ where m_0 is the number of Monte Carlo samples in suboptimal question (the mini-path samples in each node), and we repeat this work for $W = 50$ times and inside each repeat we do the duality algorithm for $E = 10$ times to obtain all the corresponding final option value and the upper bound and take mean of both as the final result. Also, we change the $K, S_0^i, b, m, P, m_0, E, W$ to test the algorithms and give the table in next section.

②***For N N-based approximation***, it is still need to be computed right now.

3. Results

3.1 2-D Longstaff Schwartz Algorithm option price and bounds

Table 1 Option Price and Lower Bound under different parameters

$basis$	S_0	K	m	b	G	$option$	$Lower$
①	10	11	365	100	10	1.1485	1.0792
①	11	11	365	100	10	2.0448	2.0012
①	10	9	365	100	10	2.6141	2.5237
①	10	11	100	100	10	1.3269	1.2045
①	10	11	500	100	10	1.2624	1.1932
①	10	11	365	500	10	1.1112	1.0377
①	10	11	365	50	10	1.4856	1.4293
①	10	11	365	100	5	1.1951	1.1549
①	10	11	365	100	50	1.1069	0.9952
②	10	11	365	100	10	0.9905	0.9517

Table 2 Option Price and upper Bound under different parameters

$basis$	S_0	K	m	b	P	m_0	W	E	$option$	$Upper$
①	10	11	30	10	3	3	50	10	1.6232	1.6901
①	10	10	30	10	3	3	20	10	2.3130	3.8027
①	10	11	30	10	5	10	50	10	1.5003	18.5157
②	10	11	30	10	5	10	50	10	1.5263	15.1267
①	10	11	40	20	3	3	50	10	1.2301	8.0892
①	10	11	10	3	3	3	20	10	1.2313	3.1527
①	10	11	30	10	3	3	100	10	1.4970	2.9503
①	10	11	30	10	3	3	50	50	1.4711	3.5328
②	10	11	30	10	3	3	50	10	1.0035	11.4052

Remark: in MATLAB files, Test.m is for the Longstaff Schwartz Algorithm using in pricing and finding lower bound; Testvalue.m is for it using in finding upper.

3.2 Neural Network-based approximation option price and bounds

I have explained each steps in MATLAB code: Maincode for HD neural network option value.m, and due to the complexity of mini-batch gradient ascent with Xavier initialization, there is still many work to obtain the final results.

4. Analysis

1. *What factors will affect the accuracy and efficiency of your algorithms? How do you improve their performance?*

①For ***L-S algorithm in 2-D***, b, m, G and different basic functions would always affect the accuracy and efficiency of pricing option:

For more paths and time periods in doing Monte Carlo and more repeat times, the pricing is more precise, you can see in table 1, it converges to $1.0 \sim 1.1$ when we take $b: 100 \rightarrow 500; m: 365 \rightarrow 500;\ G: 10 \rightarrow 50$, the value is closer to $1.0 \sim 1.1$, but the time used becomes slightly larger; and for change basic functions to type 2, it becomes amazedly fast to get the value near 1.0.

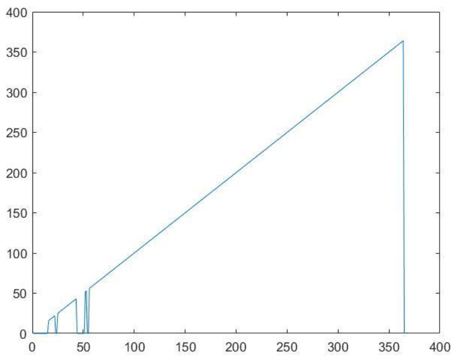
②For ***Duality Method in 2-D***, the reason for why choose much lower b, m in Duality Method than the L-S Algorithm is there are too many repeats inside each node so I need to decline the numbers to get a quick result. We try to change b, m, P, m_0, E, W and different type basic functions, you can see within lots of numerical test:

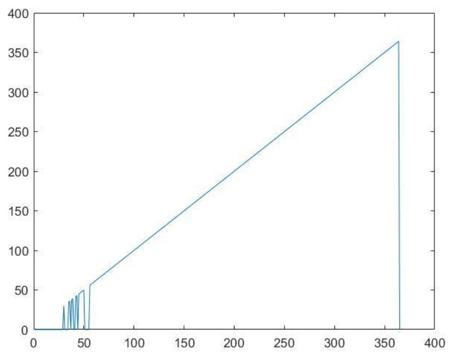
For speed, clearly if you decrease b, m, P, m_0, E, W, the time used becomes shorter, and also the same as above, if you change the basic function to type 2, the time used becomes much shorter than type 1; But for the accuracy, it is not as simple as above, if $b: 10 \rightarrow 20, m: 30 \rightarrow 40, P: 3 \rightarrow 5, m_0: 3 \rightarrow 10$, you would observe that the upper bound is not as close to the option value as the smaller situation, and also if you $b: 10 \rightarrow 3, m: 30 \rightarrow 10$, you can also see the upper bound is not as close as the larger situation; the same is for E, W, it is not more accuracy if you change them larger, so all shows that you need to try lots of times changing variables to find the proper parameters to make the results more accuracy (I find choose $b: 10; m: 30; P: 3; m_0: 3; E = 10; W = 50$ may be the proper parameters for my situation)

Remark: there would be several warnings that the matrix we generate in $\hat{B}_{\psi}$ would near a singular matrix, so the simulations need more check.

2. *What kind of features can you observe about the optimal exercising rule for Bermuda max-call options? How can you explain?*

A Bermuda option can be exercised early before maturity time T, but only on a set of specific dates before its expiration, that is, like in my project, $T = 1, m = 365$, I have below graphs for exercise time (2 tries from 100 repeats):





As for reason why it is shown like that, it’s may due to, after some period of time, the continuation value is always less than the payoff function, in other words, after several dates (in graphs, 50 days), the probability of underlying stock price tends to extreme situation (via Monte Carlo) is much larger than before, so it is better to exercise our option as early as possible.

Reference

[1] Sebastian Becker, Patrick Cheridito, Arnulf Jentzen. Deep Optimal Stopping, *Journal of Machine Learning Research*, 20, (2019), 1-25.
[2] Prof. Nan Chen. Lecture 7: Numerical Solutions to American Option Pricing, *Lecture Note from CUHK SEEM5680*, 2021.
[3] Andersen, L., and Broadie, M. 2004. Primal-dual simulation algorithm for pricing multidimensional American options. *Management Sciences*, 50, pp. 1222-1234.
[4] Haugh, M. B., and Kogan, L. 2004. Pricing American options: a duality approach. *Operations Research*, 52, pp. 258-270.
[5] Longstaff, F. A., and Schwartz, E. S. 2001. Valuing American options by simulation: a simple least-squares approach. *Review of Financial Studies*, 14, pp. 113-147.
[6] Rogers, L. C. G.. 2002. Monte Carlo valuation of American options. *Mathematical Finance*, 12, pp. 271-286.
[7] P. Glasserman. Monte-Carlo Methods in Financial Engineering, *Springer*, 2004.
[8] Prof. Mike Giles, Monte Carlo Methods, *Lecture Note from Oxford University*.
[9] Karl Sigman. Simulating normal (Gaussian) rvs with applications to simulating geometric Brownian motion in one and two dimensions, *Lecture Note*, 2007.