Computational Finance Project 2

Kai Chen, Rutgers University 02/12/2018

Introduction

This paper reports the main work of Project2 for Computational Finance (Math 16:642:623). In this project, some classical numerical methods are implemented by Monte Carlo simulation to approximate the price of a European-style up-and-out barrier call option. Numerical solutions are then compared with the closed-form solution obtained from a revised version of Black–Scholes–Merton formula and a benchmark given by an online option pricing calculator. Besides benchmarking, this paper also presents the convergence analysis and error estimation to verify the models.

Preliminary

In this project, we assume that the stock price is geometric Brownian motion and all the needed variables are constant:

- Volatility $\sigma = 0.3$
- Initial asset price $S(0) = 100$
- Risk-free interest rate $r = 0.05$
- Dividend yield $d = 0.02$
- Strike $K = 110$
- Maturity $T = 1$ year
- Barrier $U = 120$

Closed-form Formula

According to Shreve, the price of an up-and-out call satisfies a Black-Scholes-Merton equation that has been modified to account for the barrier. Besides, we also need to include the dividend yield into that formula.

Eventually, the arbitrage-free barrier call option price implied by Black-Scholes-Merton model is:

$$
\begin{aligned}
V(0) &= S(0)e^{-dT}\left[N\left(\delta_+\left(T, \frac{S(0)}{K}\right)\right) - N\left(\delta_+\left(T, \frac{S(0)}{B}\right)\right)\right] \\
&\quad - e^{-rT}K\left[N\left(\delta_-\left(T, \frac{S(0)}{K}\right)\right) - N\left(\delta_-\left(T, \frac{S(0)}{B}\right)\right)\right] \\
&\quad - S(0)e^{-dT}\left(\frac{S(0)}{B}\right)^{-\frac{2(r-d)}{\sigma^2}-1}\left[N\left(\delta_+\left(T, \frac{B^2}{KS(0)}\right)\right) - N\left(\delta_+\left(T, \frac{B}{S(0)}\right)\right)\right] \\
&\quad + e^{-rT}K\left(\frac{S(0)}{B}\right)^{-\frac{2(r-d)}{\sigma^2}+1}\left[N\left(\delta_-\left(T, \frac{B^2}{KS(0)}\right)\right) - N\left(\delta_-\left(T, \frac{B}{S(0)}\right)\right)\right] \\
&= S(0)e^{-dT}I_1 - KI_2 - S(0)e^{-dT}I_3 + KI_4
\end{aligned}
$$

where,

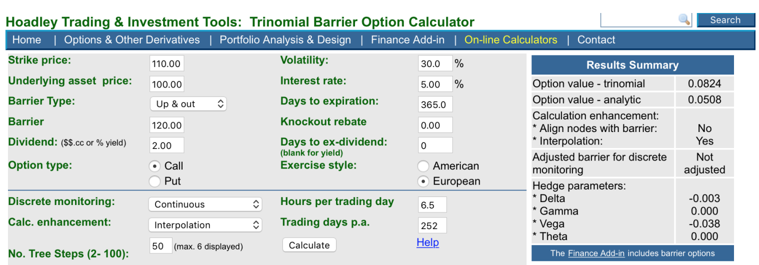
$$\delta_\pm(\tau, s) = \frac{1}{\sigma\sqrt{\tau}}\left[\log s + (r - d \pm \frac{1}{2}\sigma^2)\tau\right]$$

The prices of up-and-out barrier call option calculated by our program is:

$$c = 0.0507592$$

Benchmark

This result is identical to the benchmark that we use, as is shown in the following screen-shots:



Numerical Methods

1. Euler Solution of SDE for Spot

First, we use Euler method to generate stock price process and calculate the payoff for each iteration. The SDE of stock price is given as:

$$dS(t) = S(t)((r - d)dt + \sigma dW(t))$$

We take 252 time steps per year to simulate the process. Euler method uses the relation:

$$\hat{X}(t_{i+1}) = \hat{X}(t_i) + a(t_i, \hat{X}(t_i))(t_{i+1} - t_i) + b(t_i, \hat{X}(t_i))(W(t_{i+1}) - W(t_i))$$

with initial condition $\hat{X}(0) = X(0)$, where the Brownian motion increments may be generated using:

$$W(t_{i+1}) - W(t_i) = \sqrt{t_{i+1} - t_i} Z_{i+1}$$

Besides the stock process, we also need to generate a maximum process $M(t) = \max\{S(u) : 0 \leq u \leq t\}$ to determine the payoff of the barrier option. At time $t = T$, the payoff is given as:

$$(S(T) - K)^+ \mathbb{1}_{\{M(T) \leq U\}}$$

where $\mathbb{1}_A$ is an indicator function which has value one if the condition is true and zero otherwise.

We use in total $J = 10,000$ paths to do the simulation and the estimator of option price is given as the mean of each discounted payoff:

$$\hat{V}_n = \frac{1}{n} \sum_{j=1}^{n} V^j$$

The result calculated by our program is:

$$c_{Euler} = 0.0678352$$

Then, in order to see how good is the estimator, we do error analysis for Euler method. The standard error (SE) of a parameter is the standard deviation of its sampling distribution or an estimate of the standard deviation. The option price is estimated by the mean of all the simulations. Thus, the *standard error* can be calculated as:

$$\varepsilon = \frac{s_Y}{\sqrt{n}}$$

where, s_Y is the sample standard deviation:

$$s_Y = \sqrt{\frac{1}{n-1} \sum_{i=1}^{n} (Y_i - \hat{Y})^2}$$

Here, use Y_i represents an i.i.d random variable (in our case, it is one of the estimated prices generated by sample paths, $Y_i = V^i$). Then $\hat{Y}(= \hat{V}_n)$ is an unbiased estimator.

The standard error for Euler method in our program is:

$$\varepsilon_{Euler} = 0.00590545$$

which is small enough, showing that this estimator is reliable.

2. Antithetic Variates Method to Estimate Barrier Option

Consider pairs of random variables $(Y_i, \tilde{Y}_i), i = 1, ..., n$. We say that the pair $(Y, \tilde{Y})$ is *antithetic* if $Y = g(Z)$ and $\tilde{Y} = g(-Z)$, where Z is a standard normal random variable and $g(z)$ deterministic.

Based on the definition of antithetic variates, one can see that if $S(t)$ is geometric Brownian motion, then $(S(T), \tilde{S(T)})$ forms an antithetic pair. Thus, the stochastic differential equations of this pair are given as:

$$S(t_i + h) = S(t_i)(1 + rh + \sigma\sqrt{h}Z_{i+1})$$

$$\tilde{S}(t_i + h) = \tilde{S}(t_i)(1 + rh + \sigma\sqrt{h}Z_{i+1})$$

Finally, the *antithetic variates estimator* is defined by:

$$\hat{Y}_{AV} = \frac{1}{n}\sum_{i=1}^{n}(\frac{Y_i + \tilde{Y}_i}{2})$$

where $(Y_i, \tilde{Y}_i)$ are the discounted payoff of the up-and-out option.

The call option's price calculated by this antithetic variates method is:

$$c_{AV} = 0.0772661$$

Similarly, by treating $\hat{Y}_{AV}$ as the estimator of the price, the standard error of this estimator is:

$$\varepsilon_{AV} = 0.0044526$$

which is smaller than the standard error generated by classical Euler method, showing that we obtain a variance reduction by using AV method.

3. Control Variates Method to Estimate Barrier Option

In control variates method, we consider a pair of random variables $(X_i, Y_i), i = 1, ..., n$, where Y_i is the discounted payoff that we want to calculate (in our case, Y_i is the discounted payoff of up-and-out option). We can use a known (or easily estimable) mean $E[X]$ to give a *control variate estimator* for $E[Y]$ as:

$$E[Y] = \bar{Y}(b) = \frac{1}{n}\sum_{i=1}^{n}(Y_i - b(X_i - E[X]))$$

Here, we can use vanilla call option's price with the same strike as X variable, since its closed-form price can be easily calculated from Black-Scholes-Merton formula.

The optimal coefficient b for this method is given as:

$$b^* = \frac{Cov[X, Y]}{Var[X]}$$

In practice, we can first estimate the optimal coefficient by a set of n_1 simulations. The estimator is give by:

$$\hat{b}_{n_1} = \frac{\sum_{i=1}^{n_1}(X_i - \bar{X})(Y_i - \bar{Y})}{(X_i - \bar{X})^2}$$

Then using another set of n_2 simulations, we can obtain $\bar{Y}(b)$, which is the estimator for $E[Y]$.

The call option's price calculated by this control variates method is:

$$c_{CV} = 0.0675786$$

And the standard error is:

$$\varepsilon_{CV} = 0.00990597$$

In this experiment, we pick n_1 and n_2 as:

$$n_1 = n_2 = \frac{2}{3}J$$

where J is the total number of simulations. Notice that we choose $n_1 + n_2 << J$, however, the estimator is much closer to the closed-form solution than the results given by Antithetic variateds and standard Euler methods. On the other hand, the standard error increases compared with previous two numerical methods since we use fewer simulations in this Control method.

Question

- To answer this question, we print out the b^* and ρ when calling the Control method function with different inputs. The results are lised in the following table.

Barrier(U)	b^*	ρ	run time(s)	Option Price
120	-0.00103741	-0.033213	0.34795	0.0675786
200	0.6444	0.78421	0.340795	7.69047
1000	1	1	0.34901	9.05705

  Notice that when barrier U is getting larger, b^* and ρ are tending to 1 and the option price is tending to a vanilla call. Intuitively thinking, this is because when barrier is getting larger, the stock price is less likely to knock out that line and the barrier option is tending to a vanilla option. This is as what we expect. Thus, our program is verified.

- For this question, we use asset price ($S(t)$) as the control variate X, and vanilla discounted payoff (which is the original X variable) as the estimated variable. Then implement the Control method with different strikes. The results are listed in the following table. (see the top of next page)

  Notice that when $K = 20$, the option is deep in the money. Thus, b^* and ρ are both equal to one. Then when strike is getting larger, vanilla option is transferring into an out of money option, in which case b^* and ρ are getting smaller.

- From the results in Numerical Solution section, we find that using Antithetic method can reduce the standard error compared with standard Euler method. However, the run time of Antithetic method is longer than Euler. Thus, there is a trade-off between higher speed and less deviation.

Strike(K)	b^*	ρ	run time(s)	Vanilla Option Price
20	1	1	0.336168	78.4156
80	0.872208	0.979331	0.348753	24.3617
100	0.662269	0.920891	0.365409	13.066
120	0.414011	0.802451	0.364320	5.56197
200	0.0237219	0.314527	0.355004	0.219089

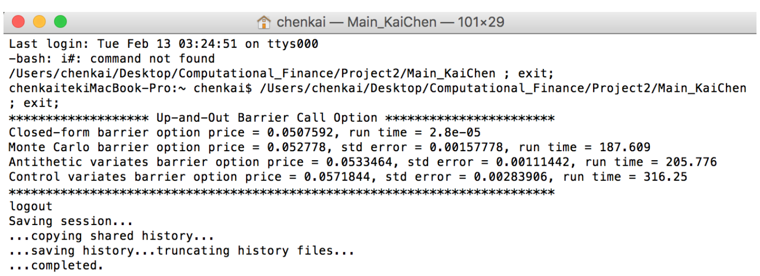
Convergence Analysis

In this part of analysis, we choose different numbers of Monte Carlo paths and different time steps to check whether the solution is converging. The results are listed below.

Num of Paths	Euler for spot	Antithetic	Control
10000	0.0704369	0.0750107	0.0661539
20000	0.0738386	0.0724388	0.0701409
50000	0.0753205	0.0745698	0.0612347
100000	0.0733213	0.0709229	0.0672835

Time Steps	Euler for spot	Antithetic	Control
252	0.0704369	0.0750107	0.0661539
500	0.0731855	0.0650737	0.0620137
1000	0.0620851	0.0628557	0.0581217
2520	0.0610172	0.0581572	0.0573027

As we can see from the above two tables, the numerical solutions are converging to the exact closed-form price. One might think the convergence is not significant enough, we think one main reason is the number of steps and time steps are still not big enough. Thus, we set big numbers for both of them and get a much better result as is shown in the following figure.


```
chenkai — Main_KaiChen — 101×29
Last login: Tue Feb 13 03:24:51 on ttys000
-bash: i#: command not found
/Users/chenkai/Desktop/Computational_Finance/Project2/Main_KaiChen ; exit;
chenkaitekiMacBook-Pro:~ chenkai$ /Users/chenkai/Desktop/Computational_Finance/Project2/Main_KaiChen
; exit;
******************* Up-and-Out Barrier Call Option ***********************
Closed-form barrier option price = 0.0507592, run time = 2.8e-05
Monte Carlo barrier option price = 0.052778, std error = 0.00157778, run time = 187.609
Antithetic variates barrier option price = 0.0533464, std error = 0.00111442, run time = 205.776
Control variates barrier option price = 0.0571844, std error = 0.00283906, run time = 316.25
**************************************************************************
logout
Saving session...
...copying shared history...
...saving history...truncating history files...
...completed.
```