CS676 Assignment 1

Author: Ziqiao Lin (John)
Student ID:20849038

Question 1

Option 1.

Q1.

$$\begin{bmatrix}1\\1\end{bmatrix}\eta_0^1+\begin{bmatrix}18u\\18d\end{bmatrix}\delta_0^1=\begin{bmatrix}\max(18-18u,0)\\ \max(18-18d,0)\end{bmatrix}$$

$$V_1^u=0 \quad V_1^d=18-18d$$

$$V_t=e^{-r_0 t}E[V_{t+1}]$$

$$=e^0\cdot(q_d^*(18-18d)+q_u^*\cdot 0)=q_d^*(18-18d)=4$$

$$\begin{bmatrix}1\\1\end{bmatrix}\eta_0^2+\begin{bmatrix}18u\\18d\end{bmatrix}=\begin{bmatrix}\max(14-18u,0)\\ \max(14-18d,0)\end{bmatrix}$$

$$V_1^u=0 \quad V_1^d=14-18d$$

$$V_t=e^{-r_0 t}E[V_{t+1}]$$

$$=1\cdot(q_d^*(14-18d)+q_u^*\cdot 0)=q_d^*(14-18d)=\frac{4}{3}$$

$$\begin{cases}q_d^*(18-18d)=4\\ q_d^*(14-18d)=\frac{4}{3}\end{cases}\Rightarrow\frac{18-18d}{14-18d}=3$$

$$18-18d=3(14-18d)$$

$$18-18d=42-54d$$

$$36d=24$$

$$d=\frac{2}{3}$$

$$q_d^*=\frac{2}{3}\quad q_u^*=\frac{1}{3}$$

$$q_u^*=\frac{1-d}{u-d}=\frac{1}{3}$$

$$u-d=3-3d$$

$$u=3-2d=3-\frac{4}{3}=\frac{5}{3}.$$

(a)

$$V_t=e^{-r_0 t}E_Q[V_{t+1}]$$

$$V_{t+1}=\begin{bmatrix}\max(18u-15,0)\\ \max(18d-15,0)\end{bmatrix}=\begin{bmatrix}15\\0\end{bmatrix}$$

$$V_t=1\cdot(15\times q_u^*+0\times q_d^*)$$

$$=15\times\frac{1}{3}+0\times q_d^*=\$5.$$

(b)

$$\begin{bmatrix}1\\1\end{bmatrix}\eta_0+\begin{bmatrix}uS_0\\dS_0\end{bmatrix}\delta_0=\begin{bmatrix}V_1^u\\V_1^d\end{bmatrix}$$

$$\begin{bmatrix}1\\1\end{bmatrix}\eta_0+\begin{bmatrix}\frac{5}{3}\times 18\\ \frac{2}{3}\times 18\end{bmatrix}\delta_0=\begin{bmatrix}15\\0\end{bmatrix}$$

$$\begin{cases}\eta_0+30\delta_0=15\\ \eta_0+12\delta_0=0\end{cases}\Rightarrow 18\delta_0=15\quad\begin{cases}\delta_0=\frac{5}{6}\\ \eta_0=-10\end{cases}$$

Figure 1. Question 1 (a)& (b)

c) $E^P[V_{t+1}] = P_u \times V_{t+1}^u + P_d \times V_{t+1}^d$

$= 0.5 \times 15 + 0.5 \times 0 = \$7.5$

By using the risk-neutral probability means there exists replicate portfolio. which is exactly the same price. Therefore, it causes the market is complete and free of arbitrage. However, if we use the real-world probability to price this option, there is an abitrage and people can use it to get money.

We can construct a replicate portfolio by \$5 and sell the option by \$7.5 (calculated by real-world probability).

Figure 2. Question 1 (c)

Question 2

Q2. $S_{t_{n+1}} = S(t_n)\, e^{(\mu - \frac{1}{2}\sigma^2)\Delta t + \sigma\phi\sqrt{\Delta t}}$ $\quad \phi \sim N(0,1)$.

$\ln\left(\frac{S_{t_{n+1}}}{S_{t_n}}\right) = (\mu - \frac{1}{2}\sigma^2)\Delta t + \sigma\sqrt{\Delta t}\,\phi$.

$$P\left(\ln\left(\frac{S_{t_{n+1}}}{S_{t_n}}\right) \le y\right) = P\left((\mu - \tfrac{1}{2}\sigma^2)\Delta t + \sigma\sqrt{\Delta t}\, Z \le y\right)$$
$$= P\left(Z \le \frac{y - (\mu - \frac{1}{2}\sigma^2)\Delta t}{\sigma\sqrt{\Delta t}}\right)$$
$$= \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{\frac{y - (\mu - \frac{1}{2}\sigma^2)\Delta t}{\sigma\sqrt{\Delta t}}} e^{-\frac{z^2}{2}}\, dz.$$

$$P(S_{t_{n+1}} \le x) = P\left(\ln\left(\frac{S_{t_{n+1}}}{S_{t_n}}\right) \le \ln\left(\frac{x}{S_{t_n}}\right)\right)$$
$$= \frac{1}{\sqrt{2\pi}} \int^{\frac{\ln(\frac{x}{S_{t_n}}) - (\mu - \frac{1}{2}\sigma^2)\Delta t}{\sigma\sqrt{\Delta t}}} e^{-\frac{z^2}{2}}\, dz$$

$$\text{pdf}(x) = \frac{dP(S_{t_{n+1}} \le x)}{dx} = \frac{1}{\sqrt{2\pi}\,\sigma x\sqrt{\Delta t}}\, e^{-\left[\frac{\ln(\frac{x}{S_{t_n}}) - (\mu - \frac{\sigma^2}{2}\Delta t)}{\sigma\sqrt{\Delta t}}\right]^2 / 2}$$

$$\sim \text{lognormal}\left((\mu - \tfrac{\sigma^2}{2})\Delta t,\ \sigma^2\Delta t\right)$$

$$E\left[\frac{S_{t_{n+1}}}{S_{t_n}}\right] = e^{\left((\mu - \frac{\sigma^2}{2})\Delta t + \frac{\sigma^2}{2}\Delta t\right)}$$

Since stock price S_n and t_n are given

$$E[S_{t_{n+1}}] = S_{t_n} e^{\mu\Delta t}$$

$$Var\left(\frac{S_{t_{n+1}}}{S_{t_n}}\right) = (e^{\sigma^2\Delta t} - 1)\, e^{2(\mu - \frac{\sigma^2}{2})\Delta t + \sigma^2\Delta t} = (e^{\sigma^2\Delta t} - 1)\, e^{2\mu\Delta t}$$
$$Var(S_{t_{n+1}}) = S_{t_n}^2 (e^{\sigma^2\Delta t} - 1)\, e^{2\mu\Delta t}$$

$$E[S_{t_{n+1}}^2] = Var[S_{t_{n+1}}] + E[S_{t_{n+1}}]^2 = S_{t_n}^2 (e^{\sigma^2\Delta t} - 1)\, e^{2\mu\Delta t} + S_{t_n}^2 e^{2\mu\Delta t}$$
$$= S_{t_n}^2 e^{(\sigma^2 + 2\mu)\Delta t}$$

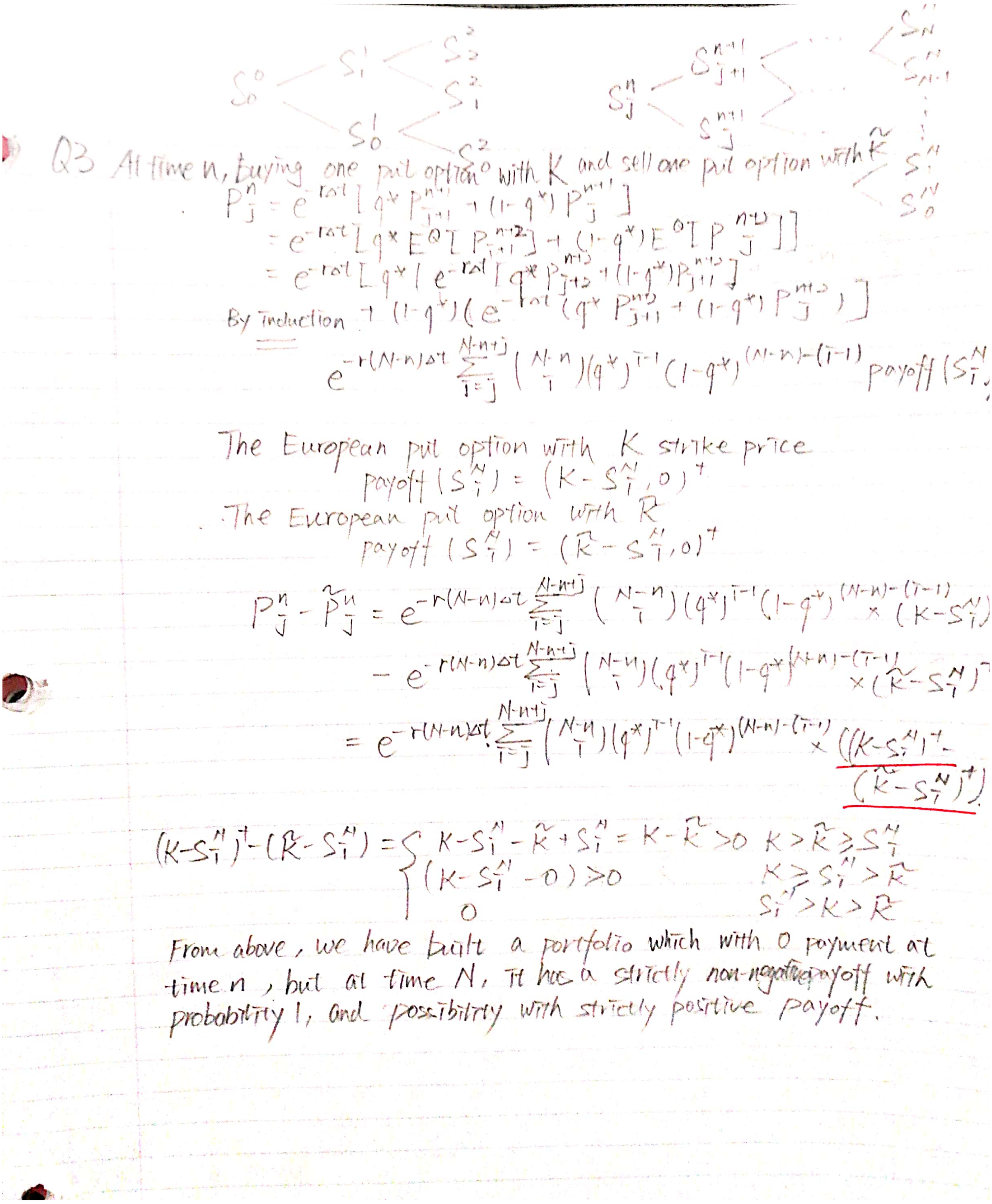
Next, we will show the $E[S_{t_{n+1}}^2]$ and $E[S_{t_{n+1}}]$ from (3).

$$S_{t_n}(pu + (1-p)d) = S_{t_n} e^{\mu\Delta t}$$
$$\Rightarrow \quad E[S_{t_{n+1}}] = S_{t_n} e^{\mu\Delta t}$$
$$S_{t_n}^2 (pu^2 + (1-p)d^2) = S_{t_n}^2 e^{(2\mu + \sigma^2)\Delta t}$$
$$E[S_{t_{n+1}}^2] = S_{t_n}^2 e^{(2\mu + \sigma^2)\Delta t}$$

Therefore, we can conclude that a binomial model satisfying (3) converges to the lognormal Black-Scholes model (1).

Figure 3. Question 2

Question 3



S_0^0, S_1^1, S_0^1, S_2^2, S_1^2, S_0^2, S_j^n, S_{j+1}^{n+1}, S_j^{n+1}, S_N^N, S_{N-1}^N, S_1^N, S_0^N

Q3 At time n, buying one put option with K and sell one put option with $\tilde{K}$

$$P_j^n = e^{-r\Delta t}[q^* P_{j+1}^{n+1} + (1-q^*)P_j^{n+1}]$$
$$= e^{-r\Delta t}[q^* E^Q[P_{j+1}^{n+2}] + (1-q^*)E^Q[P_j^{n+2}]]$$
$$= e^{-r\Delta t}[q^*(e^{-r\Delta t}[q^* P_{j+2}^{n+2} + (1-q^*)P_{j+1}^{n+2}]$$
$$+ (1-q^*)(e^{-r\Delta t}(q^* P_{j+1}^{n+2} + (1-q^*)P_j^{n+2}))]$$

By Induction

$$e^{-r(N-n)\Delta t}\sum_{i=j}^{N-n+j}\binom{N-n}{i}(q^*)^{i-1}(1-q^*)^{(N-n)-(i-1)}\,\text{payoff}(S_i^N)$$

The European put option with K strike price

$$\text{payoff}(S_i^N) = (K - S_i^N, 0)^+$$

The European put option with $\tilde{K}$

$$\text{payoff}(S_i^N) = (\tilde{K} - S_i^N, 0)^+$$

$$P_j^n - \tilde{P}_j^n = e^{-r(N-n)\Delta t}\sum_{i=j}^{N-n+j}\binom{N-n}{i}(q^*)^{i-1}(1-q^*)^{(N-n)-(i-1)} \times (K-S_i^N)$$
$$- e^{-r(N-n)\Delta t}\sum_{i=j}^{N-n+j}\binom{N-n}{i}(q^*)^{i-1}(1-q^*)^{(N-n)-(i-1)} \times (\tilde{K}-S_i^N)^+$$
$$= e^{-r(N-n)\Delta t}\sum_{i=j}^{N-n+j}\binom{N-n}{i}(q^*)^{i-1}(1-q^*)^{(N-n)-(i-1)} \times \underline{((K-S_i^N)^+ - (\tilde{K}-S_i^N)^+)}$$

$$(K-S_i^N)^+ - (\tilde{K}-S_i^N) = \begin{cases} K - S_i^N - \tilde{K} + S_i^N = K - \tilde{K} > 0 & K > \tilde{K} \geq S_i^N \\ (K - S_i^N - 0) > 0 & K \geq S_i^N > \tilde{K} \\ 0 & S_i^N > K > \tilde{K} \end{cases}$$

From above, we have built a portfolio which with 0 payment at time n, but at time N, it has a strictly non-negative payoff with probability 1, and possibility with strictly positive payoff.

Figure 4. Question 3

Question 4

Question 5

Q5

$$Z = f(S_t, t) = S^2. \quad \frac{\partial f}{\partial S} = 2S, \quad \frac{\partial f}{\partial t} = 0 \quad \frac{\partial^2 f}{\partial S^2} = 2.$$

$$dZ = \cancel{\frac{\partial f}{\partial t} dt} + \frac{\partial f}{\partial S} dS + \frac{1}{2} \frac{\partial^2 f}{\partial S^2} dS^2$$

$$dZ = 2S\,dS + \frac{1}{2} \cdot 2 (u\,dt + \sigma\, dZ_t)^2$$

$$2S\,dS = dZ - (\underbrace{u^2 dt^2}_{\to 0} + \underbrace{2u\sigma\, dt\, dZ_t}_{\to 0} + \underbrace{\sigma^2 dZ_t^2}_{= \sigma^2 dt})$$

$$2\int_0^T S(t)\,dS(t) = \int_0^T dS(t)^2 - \sigma^2 \int_0^T dt$$

$$\int_0^T S(t)\,dS(t) = \frac{S(T)^2 - S(0)^2}{2} - \left(\frac{T}{2}\right)\sigma^2.$$

Figure 6. Question 5

Question 6 (a)

The followings are the code for Question 6 part (a)

```
%
% Compute Black-Scholes iption value using a binomial tree
% Eurpoean
% vectorized code
n = 10;          % running n times
S0 = 100;        % S0  -   current stock price
K = 100;         % K - strike
T = ones(1,n);   % T- expiry time
r = 0.025;       % r - interest rate
sigma = 0.25;    % sigma - volatility
optype = 0;      % optype  -  0 for call , 1 for a put
Nsteps = 10*2.^([1:n]');  %Nsteps - number of timesteps

delt = T./Nsteps;  % delt_time

% tree parameters

   u  = exp ((sigma*sqrt(delt)) + (r-sigma^2/2)*delt);
   d =  exp (-(sigma*sqrt(delt)) + (r-sigma^2/2)*delt);
   a = exp(r * delt);
   p = (a - d)./(u-d);

%
% payoff at T
%
for j = 1:n

   W = S0*d(j).^([Nsteps(j):-1:0]').*u(j).^([0:Nsteps(j)]');
          % W is column vector of size Nsteps+1 X 1
   if(optype == 1)
        W = max(W - K, 0);
   else
      W = max(K - W, 0 );
   end

   %backward recursion

   for i = Nsteps(j):-1:1
      W = exp(-r*delt(j))*(p(j)*W(2:i+1) + (1-p(j))*W(1:i));
   end
```

```
  value(j) = W(1);

  disp(sprintf('Tree Value: %.9g \n',value));
end
value = value';

[Call,Put] = blsprice(100,100,0.025,1,0.25)
```

Firstly, we set the optype = 1 to simulate the put option price. Then we conclude the results in the following table:

Table 1 : Convergence Test for Put Option without dividend				
Δt	Value	Change	Ratio	
0.05	8.567114			blsprice
0.025	8.612735	0.045621		8.6392
0.0125	8.632355	0.019619	2.325304	
0.00625	8.639855	0.0075	2.615857	
0.003125	8.641962	0.002107	3.559206	
0.001563	8.641853	-0.00011	-19.2478	
0.000781	8.640975	-0.00088	0.124802	
0.000391	8.639955	-0.00102	0.859937	
0.000195	8.639034	-0.00092	1.107403	
9.77E-05	8.639289	0.000254	-3.62038	

Table 1: Convergence Test for Put Option without dividend

Then, we can change the optype = 0 and use this algorithm again and again while changing the value of Δt to get the results of call option in the table as follow:

Table 1 : Convergence Test for Call Option without dividend				
Δt	Value	Change	Ratio	
0.05	11.03612			blsprice
0.025	11.08174	0.045621		11.1082
0.0125	11.10136	0.019619	2.325304	
0.00625	11.10886	0.0075	2.615857	
0.003125	11.11097	0.002107	3.559206	
0.001563	11.11086	-0.00011	-19.2478	
0.000781	11.10998	-0.00088	0.124802	
0.000391	11.10896	-0.00102	0.859937	
0.000195	11.10804	-0.00092	1.107403	
9.77E-05	11.1083	0.000254	-3.62038	

Table 2: Convergence Test for Call Option without dividend

As we can see from the 2 tables above, both the put price and the call price converge to the put and call option price done by BS solutions blsprice. However, the ratio seems to have the trend to 4 at the beginning, but it suddenly drops to negative. The reason is because of the drift term in u and d. If we have to conclude, I would say that the $\lim_{\Delta t \to 0} \frac{V(\frac{\Delta t}{2}) - V(\Delta t)}{V(\frac{\Delta t}{4}) - V(\frac{\Delta t}{2})}$ is closing to 4 rather than 2, which means it is a quadratic convergence rate model instead of the linear.

Question 6 part (b)

```
% Compute Black-Scholes option value using a binomial tree
% Eurpoean case
% Vcetorized code
S0 = 100;           %S0 - current stock price
K = 100;            %K  - strike price
T = 1.0;            %T  - expiry time
r = 0.025;          %r  - interest rate
sigma = 0.25;       %Sigma - volitality
optype = 0;         %Option type
Nsteps = 100;       %Number of timesteps

Ndsteps = 50;          %number of timesteps to calculate the dividend
delt = T/Nsteps;       %Time Period
phi = [0,0.04,0.08];    %Dividend Rate


% tree parameters

   u  = exp ((sigma*sqrt(delt)) + (r-sigma^2/2)*delt);
   d =  exp (-(sigma*sqrt(delt)) + (r-sigma^2/2)*delt);
   a = exp(r * delt);
   p = (a - d)/(u-d);
%
%
%
%Stock value at dividend date.
Sd = S0*d.^([Ndsteps:-1:0]').*u.^([0:Ndsteps]');
%
% payoff at T
%

 W = S0*d.^([Nsteps:-1:0]').*u.^([0:Nsteps]');
 % W is column vector of size Nsteps+1 X 1
   if(optype == 0)
       W = max(W - K, 0);  % call option
```

```
else
   W = max(K - W, 0 ); % put option
end

%backward recursion

for i = Nsteps:-1:Ndsteps+1
   W = exp(-r*delt)*(p*W(2:i+1) + (1-p)*W(1:i));
end
value = W(1:Ndsteps+1,1);
value2 = value;
W_out = dividend( value, Sd, phi.*Sd);  % calculate the dividend
W_out_2 = W_out;
for i = Ndsteps:-1:1
    W_out =  exp(-r*delt)*(p*W_out(2:i+1,:) + (1-p)*W_out(1:i,:));
end
value = W_out;

disp(sprintf('Tree Value: %.9g \n',value));
```

Table 3: Price of at the money Put Option with dividend yield

ρ	0	0.04	0.08
V0	8.63566791	10.4349779	12.4822949

Table 4: Price of at the money Call Option with dividend yield

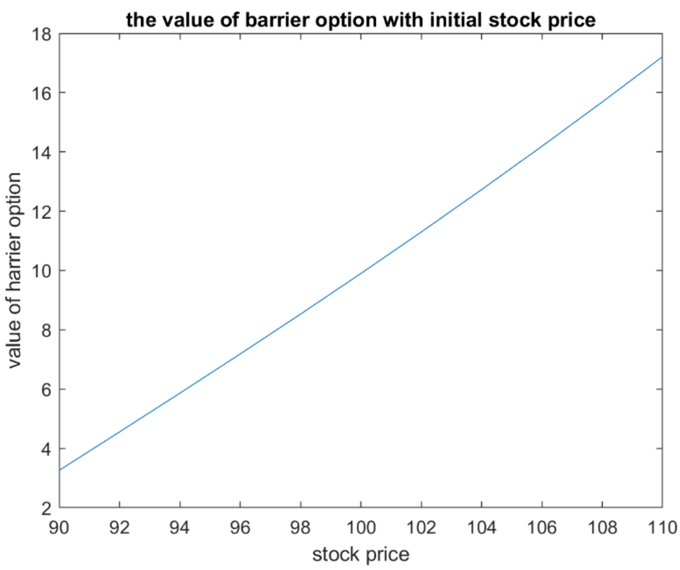
ρ	0	0.04	0.08
V0	11.1046767	8.90398666	6.95130368

As I have shown above, we can see that with the ρ increases, the value of put option increases as well, but the call option decreases. This is reasonable since that once a stock gives dividend, the value of the stock decreases, and payoff for put **max(K – S, 0)** increases, so the put option price is about to increase. However, for call option **max(S – K, 0)** decreases, so the call option price decreases.

Question 7(a)

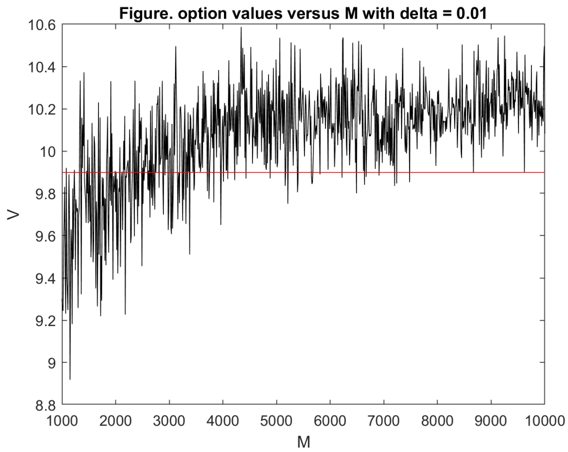
```
randn('state',100);
%
T = 1.00;      %expiry time
```

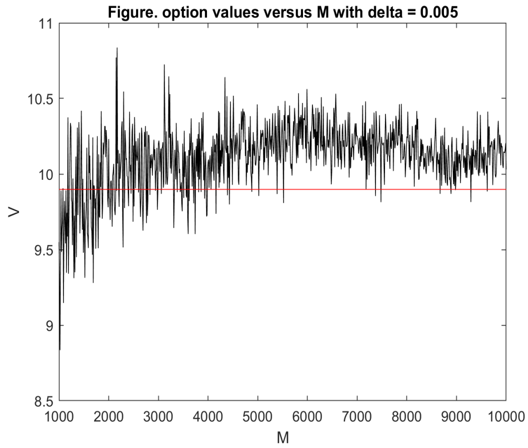
```
sigma = 0.25;   %volatility
mu = 0.01;      %P measure drift
S = [90:2:110];   %initial value
N_sim = 10000;  %   number of simulations
N = 100;        %   number of timesteps
delt = T/N;     %   time stpe
B = 85;         % Barrier value
S_init = 100;   %initial value
t = 0; % initial time
K = S_init;     % Strike price
r = 0.025;      % risk-free interest rate

d1 = (log( S/K ) + ( r + 1/2 * sigma^2)*( T - t))/( sigma*sqrt(T-t));
d2 = (log( S/K ) + ( r - 1/2 * sigma^2)*( T - t))/( sigma*sqrt(T-t));
d3 = (log( S/B ) + ( r + 1/2 * sigma^2)*( T - t))/( sigma*sqrt(T-t));
d4 = (log( S/B ) + ( r - 1/2 * sigma^2)*( T - t))/( sigma*sqrt(T-t));
d5 = (log( S/B ) - ( r - 1/2 * sigma^2)*( T - t))/( sigma*sqrt(T-t));
d6=  (log( S/B )  -( r + 1/2 * sigma^2)*( T - t))/( sigma*sqrt(T-t));
d7 = (log( S*K/B^2 ) - ( r- 1/2 * sigma^2)*( T-t ))/( sigma*sqrt(T-t));
d8 = (log( S*K/B^2 ) - ( r + 1/2 * sigma^2)*( T-t ))/( sigma*sqrt(T-t));

V_1 =  S.*(normcdf(d1) - ((B./S).^(1+2*r/sigma^2)).*(1-normcdf(d8)))-...
       K*exp((-r*(T-t)))*(normcdf(d2) - ((B./S).^(-1+2*r/sigma^2)).*(1-normcdf(d7)));

plot(S,V_1);
```

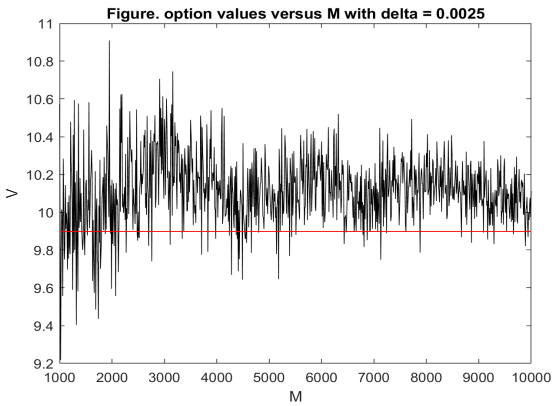


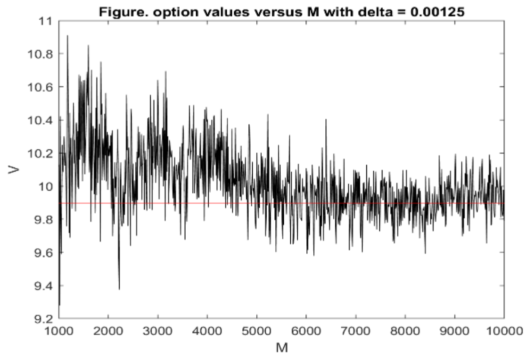
90	3.257551
92	4.554787
94	5.861533
96	7.184398
98	8.528717
100	9.898665
102	11.29738
104	12.72708
106	14.18918
108	15.68443
110	17.21298

(b) and (c) Yes. The $V(\widetilde{S(0)},0)$ depends on the time discretization. Since when we use Eqn(12) to simulate the option value, we actually discretize the time into many small pieces. We can only know in certain discrete time point that the stock drops below the bound, but we cannot know the exact time that the stock hits the boundary. The larger the timesteps we split, the delt_t is smaller, we know the more accurate time that hits the bound. Therefore, we can select the time interval get close to 0, the discrete time process will get close to a continuous process.





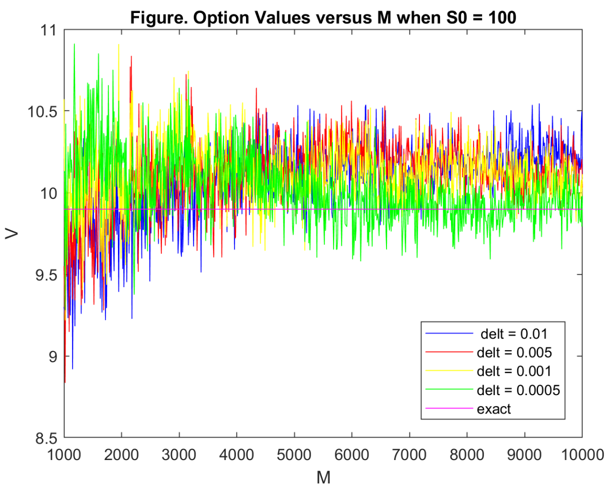




As shown in the 4 figures, with the increase in delta_t, the simulated option value is closer to the continuous option value, and the volatility is smaller.

```
for M = 1000:10:10000
randn('state',3)
K=100;   %strike price
r=0.025;   % interest free rate
sigma=0.25;     % volitality
T=1;           % 1 year = 250 trading days
S0=100;          % initial price
n=100;           % number of simulations
%M=8000;
B = 85;
delt = T/n;     %dleta_t, starting from 5 days
%Simulataneous Creation of the Wiener-Process for M Path
S_old = zeros(M,1);
S_new = zeros(M,1);

S_old(1:M,1) = S0;
```

```
drift = (r - sigma^2/2).*delt;
sigma_sqrt_delt = sigma.*sqrt(delt);

for i = 1:n     %timestep loop
   % now, for each timestep, generate info for
   % all simulations

   S_new(:,1) = ...
          S_old(:,1).*exp((drift + sigma_sqrt_delt.*randn(M,1)));

   %S_new(:,1) = max(0.0,S_new(:,1));
          % check to make sure that S_new cannot be <0
   S_new = S_new .*(S_new(:,1) > B);
   S_old(:,1) = S_new(:,1);
          %
          % end of generation of all data for alll simulations
          % for this timestep
end % timestep loop

S_N = S_new;
%Simultaneous Calculation of the Payoff
%S_new(S_new < B) = 0;
%S_new(S_new > B) = max(S_new(S_new > B)- K, 0 );
S_new = max(S_new - K, 0);
payoff= S_new;
%Simultaneous Calculationof the Estimator and the Option Prices
V1((M-1000)/10 + 1)=exp(-r*T)*(mean(payoff));
disp(sprintf('Price: %.5g\n',V1((M-1000)/10 + 1)));
end

X = 1000 : 10 :10000;
plot(X, V1)
```



Not only from those 4 figures above, from this combined figure, we can see that as the delt_t gets smaller and smaller, the simulated price will converge to its continuous value. The easiest way to think about this is as the time interval gets smaller to 0, the discrete process gets close to the continuous process. Besides, we can see that as the delt_t becomes smaller, the price converges faster, and the overall volatility is smaller as well. This is all because the central limit theorem.

Question 8

As we have analysed from last question, the main problem from last simulation is the discretization error, which means we cannot have a exact time that hits the barrier, but only a discrete point that we know the stock price hits the barrier in between two discrete time. To make it easier to identify the time point that hits the barrier, the author uses uniformly distributed random variables and an exit probability to obustly estimate the first time that stock price hits the barrier. The following code is the implementation of MMC method.

```
randn('state',3)
   K=100;   %strike price
   r=0.025;  % interest free rate
   sigma=0.25;     % volitality
   T=1;          % 1 year = 250 trading days
   S0=100;         % initial price
   n=[10,50,100,200];          % number of simulations
   M=100000;
   B = 85;
for j = 1:4
   delt = T/n(j);     %dleta_t, starting from 5 days
   %Simulataneous Creation of the Wiener-Process for M Path
   S_old = zeros(M,1);
   S_new = zeros(M,1);
   P = zeros(M,1);
   U = zeros(M,1);
   S_old(1:M,1) = S0;
   drift = (r - sigma^2/2)*delt;
   sigma_sqrt_delt = sigma*sqrt(delt);

   for i = 1:n(j)     %timestep loop
      % now, for each timestep, generate info for
      % all simulations

      S_new(:,1) = ...
             S_old(:,1).*exp((drift + sigma_sqrt_delt.*randn(M,1)));
      P(:,1) = exp(-2.*(B-S_new).*(B-S_old)./(delt * sigma^2 .* S_old.^2));
      U(:,1) = unifrnd(0,1,M,1);
      %S_new(:,1) = max(0.0,S_new(:,1));
```

```
            % check to make sure that S_new cannot be <0
        %flag(:,1) = (U(:,1) > P(:,1)) && (S_new(:,1) > B);
        S_new = S_new .*((U(:,1) > P(:,1)));
        S_new = S_new .*((S_new(:,1) > B));
        S_old(:,1) = S_new(:,1);
            % S_new(:,1) > B &&
            % end of generation of all data for alll simulations
            % for this timestep
    end % timestep loop

    S_N = S_new;
    %Simultaneous Calculation of the Payoff
    %S_new(S_new < B) = 0;
    %S_new(S_new > B) = max(S_new(S_new > B)- K, 0 );
    S_new = max(S_new - K, 0);
    payoff= S_new;
    %Simultaneous Calculationof the Estimator and the Option Prices

    V_MMC(j)=mean(exp(-r*T)*payoff);
    std_MMC(j) = std(payoff);
    MMC_error(j) = V_MMC(j)-9.8987;

end
disp(sprintf('Price: %.5g\n',V_MMC));



% real value 9.8987

%Grafical Output
%Vexakt=call(S,0,K,r,sigma,T);
%plot (abs(bsxfun(@minus,V,Vexakt))./Vexakt)
```

MM	n	V_MMC	V_MC	MC_error	MMC_error
1e+05	10	9.8226	10.571	0.67243	-0.076077
1e+05	50	9.7623	10.14	0.24153	-0.13639
1e+05	100	9.8677	10.155	0.25668	-0.031024
1e+05	200	9.9048	10.104	0.20495	0.0060772

Table: Comparison of the exact and the MC, MMC approximated values for down-and-out call option.

The modified model can calculate the price more accurate than the non-modified model. And the reason for this is that, after the modification, we analyze the times between two time points, therefore the modified simulated price is more accurate.