Question1

(a)

Stepping.m

This matlab file is used to determine the option value by implicit method, CN method and CN-R method

function [result1,result2,result3] = stepping(S,N)

alpha\_parameter =0.8; % constant related to alpha

r = .02; % risk free rate

T = 1; % time to expiry

K = 10; % strike price

S0 = 10; % initial stock price

delt = T/N; % time interval

m = length(S); %number of grids in row

alpha\_central = zeros(m,1);

beta\_central = zeros(m,1);

alpha\_forward = zeros(m,1);

beta\_forward = zeros(m,1);

alpha\_backward = zeros(m,1);

beta\_backward = zeros(m,1);

alpha = zeros(m,1);

beta = zeros(m,1);

for i = 2: m - 1

%%central alpha and beta formula

alpha\_central(i) = alpha\_parameter^2\*S(i)/((S(i) - S(i-1))\*(S(i+1) - S(i-1)))...

-r\*S(i)/(S(i+1) - S(i-1));

beta\_central(i) = alpha\_parameter^2\*S(i)/((S(i+1) - S(i))\*(S(i+1) - S(i-1)))...

+r\*S(i)/(S(i+1) - S(i-1));

%%forward alpha and beta formula

alpha\_forward(i) = alpha\_parameter^2\*S(i)/((S(i) - S(i-1))\*(S(i+1) - S(i-1)));

beta\_forward(i) = alpha\_parameter^2\*S(i)/((S(i+1) - S(i))\*(S(i+1) - S(i-1)))...

+ r\*S(i)/(S(i+1) - S(i)); % same as beta\_central

%%backward alpha and beta formula

alpha\_backward(i) = alpha\_parameter^2\*S(i) / ((S(i) - S(i-1))\*(S(i+1) - S(i-1))) ...

- r \* S(i) / (S(i+1) - S(i));

beta\_backward(i) = alpha\_parameter^2\*S(i)/((S(i+1) - S(i))\*(S(i+1) - S(i-1)));

end

%% choosing parameter

for i = 2:m-1

if(alpha\_central(i) >=0 && beta\_central(i) >=0)

alpha(i) = alpha\_central(i);

beta(i) = beta\_central(i);

elseif (alpha\_forward(i) >=0 && beta\_forward(i) >=0)

alpha(i) = alpha\_forward(i);

beta(i) = beta\_forward(i);

else

alpha(i) = alpha\_backward(i);

beta(i) = beta\_backward(i);

end

end

M\_matrix = [delt.\*-alpha,delt.\*(alpha + beta + r) , delt.\*-beta];

M = spdiags(M\_matrix, [-1,0,1], m-1, m);

M = full([M;zeros(1,m)]);

I = eye(m);

A1 = sparse(I + M); % create space matrix

[L1,U1,P1,Q1] = lu(A1); % lu factorization

V = zeros(m, N+1);

V\_init = max(K - S.^2, S.^2 - K)';

V(:,1) = V\_init;

V\_old = V\_init;

V\_new = zeros(m,1);

for i = 1: N

V\_new = Q1 \* ((L1\*U1) \ P1) \* V\_old;

V\_old = V\_new;

V(:,i+1) = V\_new;

end

X1 = sprintf('The option value for fully implict in N = %d, S= %d, is: %s',N,length(S),V\_new(S == S0));

disp(X1)

result1 = V\_new(S == S0);

%% for CN time stepping

M\_matrix\_CN = [delt.\*-alpha/2, delt.\*(alpha + beta + r)/2, delt.\*-beta/2];

M\_CN = spdiags(M\_matrix\_CN, [-1,0,1], m-1, m);

M\_CN = full([M\_CN;zeros(1,m)]);

A2 = sparse(spdiags(ones(m,1),0,m,m)+M\_CN); % create space matrix

[L2,U2,P2,Q2] = lu(A2); % lu factorization

V\_2 = zeros(m, N+1);

V\_init\_2 = max(K - S.^2, S.^2 - K)';

V\_2(:,1) = V\_init\_2;

V\_old\_2 = V\_init\_2;

V\_new\_2 = zeros(m,1);

for i = 1: N

Vt = (I - M\_CN)\*V\_old\_2;

V\_new\_2 = Q2 \* ((L2\*U2) \ (P2\*Vt)) ;

V\_old\_2 = V\_new\_2;

V\_2(:,i+1) = V\_new\_2;

end

X2 = sprintf('The option value for NC in N = %d, S= %d, is: %s',N,length(S),V\_new\_2(S == S0));

disp(X2)

result2 = V\_new\_2(S == S0);

%% CN-Rannacher time stepping

V\_3 = zeros(m, N+1);

V\_init\_3 = max(K - S.^2, S.^2 - K)';

V\_3(:,1) = V\_init\_3;

V\_old\_3 = V\_init\_3;

V\_new\_3 = zeros(m,1);

%first two use fully implicit method

for i = 1: 2

V\_new\_3 = Q1 \* ((L1\*U1) \ (P1 \* V\_old\_3));

V\_old\_3 = V\_new\_3;

V\_3(:,i+1) = V\_new\_3;

end

for i = 3: N

Vt\_R = (I - M\_CN)\*V\_old\_3;

V\_new\_3 = Q2 \* ((L2\*U2) \ (P2 \* Vt\_R));

V\_old\_3 = V\_new\_3;

V\_3(:,i+1) = V\_new\_3;

end

X3 = sprintf('The option value for NC-R in N = %d, S= %d, is: %s',N,length(S),V\_new\_3(S == S0));

disp(X3)

result3 = V\_new\_3(S == S0);

end

K = 10;

N = 25;

S = [0:0.1\*K:0.4\*K,... %input S value

0.45\*K:0.05\*K:0.8\*K,...

0.82\*K:0.02\*K:0.9\*K,...

0.91\*K:0.01\*K:1.1\*K,...

1.12\*K:0.02\*K:1.2\*K,...

1.25\*K:.05\*K:1.6\*K,...

1.7\*K:0.1\*K:2\*K,...

2.2\*K, 2.4\*K, 2.8\*K,...

3.6\*K, 5\*K, 7.5\*K, 10\*K];

% for node 62 and timestep 25

L = length(S);

[V,V\_CN,V\_CNR] = stepping(S,N);

% for node 123 and timestep 50

N1 = 50;

S1 = movmean(S,2);

S1 = [S,S1];

S1 = sort(S1);

S1 = S1(2:end);

L1 = length(S1);

[V1,V\_CN1,V\_CNR1] = stepping(S1,N1);

% for node 245 and timestep 100

N2 = 100;

S2 = movmean(S1,2);

S2 = [S1,S2];

S2 = sort(S2);

S2 = S2(2:end);

L2 = length(S2);

[V2,V\_CN2,V\_CNR2] = stepping(S2,N2);

% for node 489 and timestep 200

N3 = 200;

S3 = movmean(S2,2);

S3 = [S2,S3];

S3 = sort(S3);

S3 = S3(2:end);

L3 = length(S3);

[V3,V\_CN3,V\_CNR3] = stepping(S3,N3);

% for node 489 and timestep 200

N4 = 400;

S4 = movmean(S3,2);

S4 = [S3,S4];

S4 = sort(S4);

S4 = S4(2:end);

L4 = length(S4);

[V4,V\_CN4,V\_CNR4] = stepping(S4,N4);

T\_1 =table([N;N1;N2;N3;N4],...

[L;L1;L2;L3;L4],...

[V;V1;V2;V3;V4],...

[NaN;V1-V;V2-V1;V3-V2;V4-V3],...

[NaN;NaN;(V1-V)/(V2-V1);(V2-V1)/(V3-V2);(V3-V2)/(V4-V3)]);

T\_1.Properties.VariableNames ={'Timesteps','Node','Value','Change','Ratio'};

T\_2 =table([N;N1;N2;N3;N4],...

[L;L1;L2;L3;L4],...

[V\_CN;V\_CN1;V\_CN2;V\_CN3;V\_CN4],...

[NaN;V\_CN1-V\_CN;V\_CN2-V\_CN1;V\_CN3-V\_CN2;V\_CN4-V\_CN3],...

[NaN;NaN;(V\_CN1-V\_CN)/(V\_CN2-V\_CN1);(V\_CN2-V\_CN1)/(V\_CN3-V\_CN2);(V\_CN3-V\_CN2)/(V\_CN4-V\_CN3)]);

T\_2.Properties.VariableNames ={'Timesteps','Node','Value','Change','Ratio'};

T\_3 =table([N;N1;N2;N3;N4],...

[L;L1;L2;L3;L4],...

[V\_CNR;V\_CNR1;V\_CNR2;V\_CNR3;V\_CNR4],...

[NaN;V\_CNR1-V\_CNR;V\_CNR2-V\_CNR1;V\_CNR3-V\_CNR2;V\_CNR4-V\_CNR3],...

[NaN;NaN;(V\_CNR1-V\_CNR)/(V\_CNR2-V\_CNR1);(V\_CNR2-V\_CNR1)/(V\_CNR3-V\_CNR2);(V\_CNR3-V\_CNR2)/(V\_CNR4-V\_CNR3)]);

T\_3.Properties.VariableNames ={'Timesteps','Node','Value','Change','Ratio'};

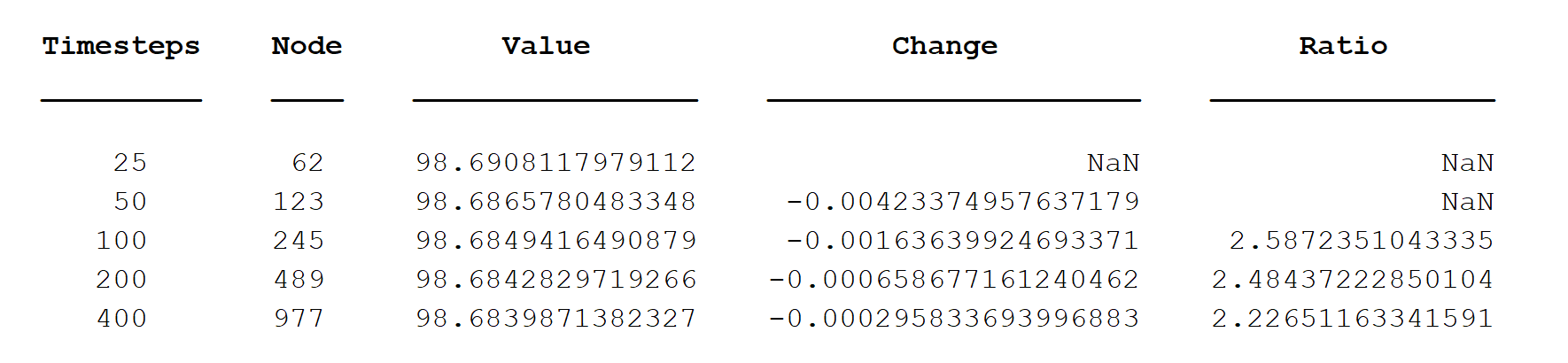


Table 1: The option value determined by the implicit method in different timesteps and nodes

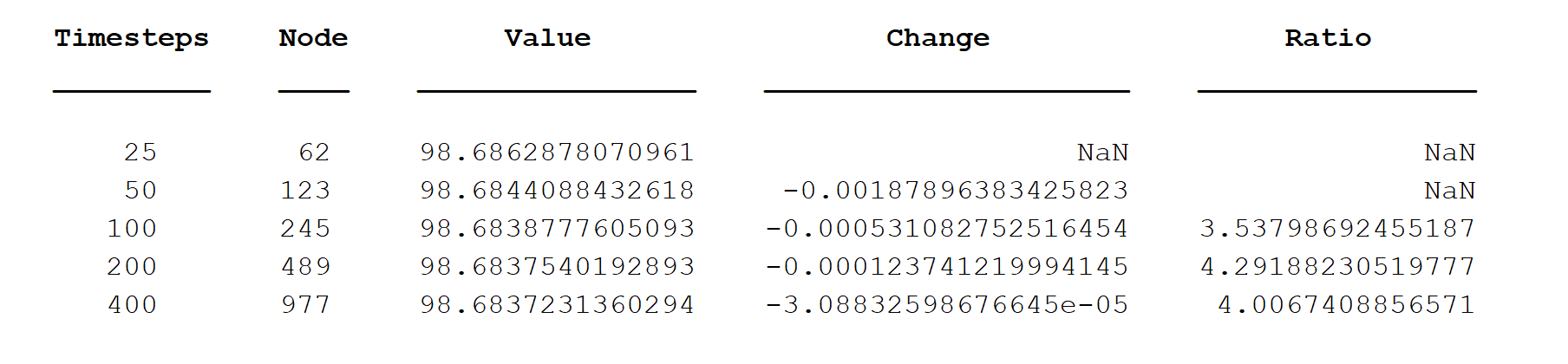


Table 2: The option value determined by the CN method in different timesteps and nodes

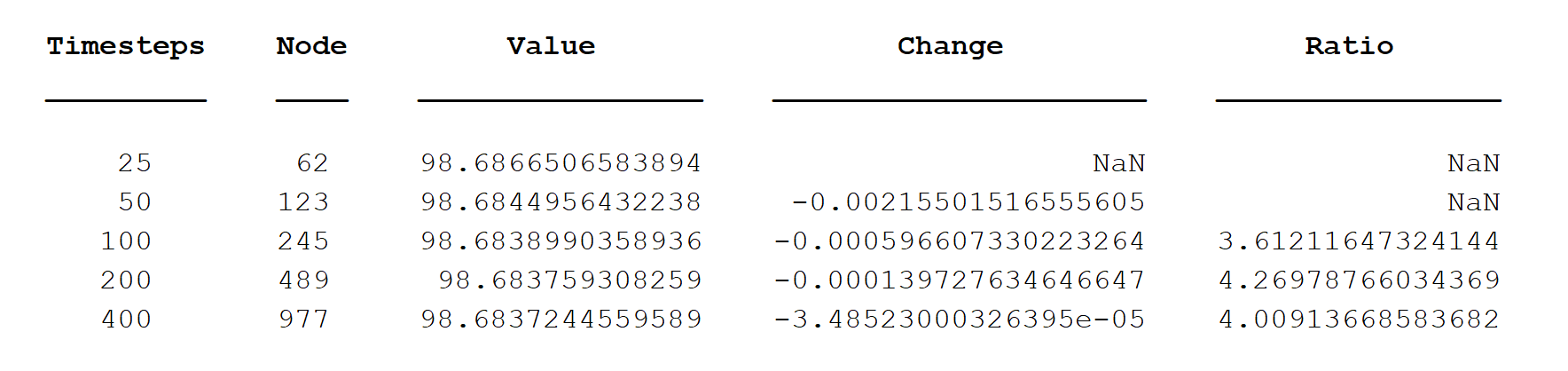


Table 2: The option value determined by the CN-R method in different timesteps and nodes

From the convergence table, we can see that the ratio of fully implicit method is around 2, but for CN method and CN-R method, there ratio is all about 4. It implies that the fully implicit is linear convergent, while CN and CN-R are quadratic convergent, but CN-R has less fluctuate than CN method.

(b)

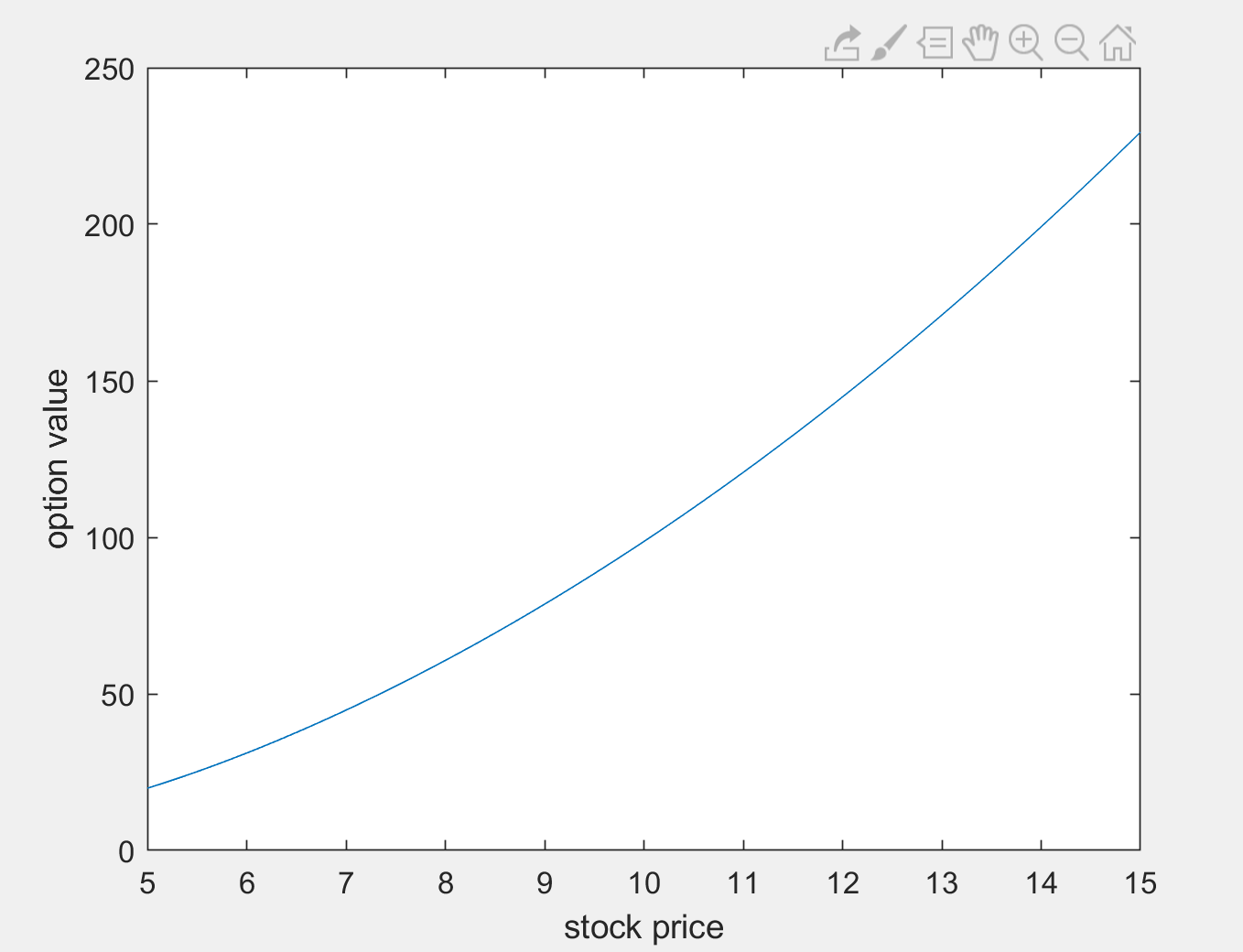


Figure1 . the plot option value varied with stock price

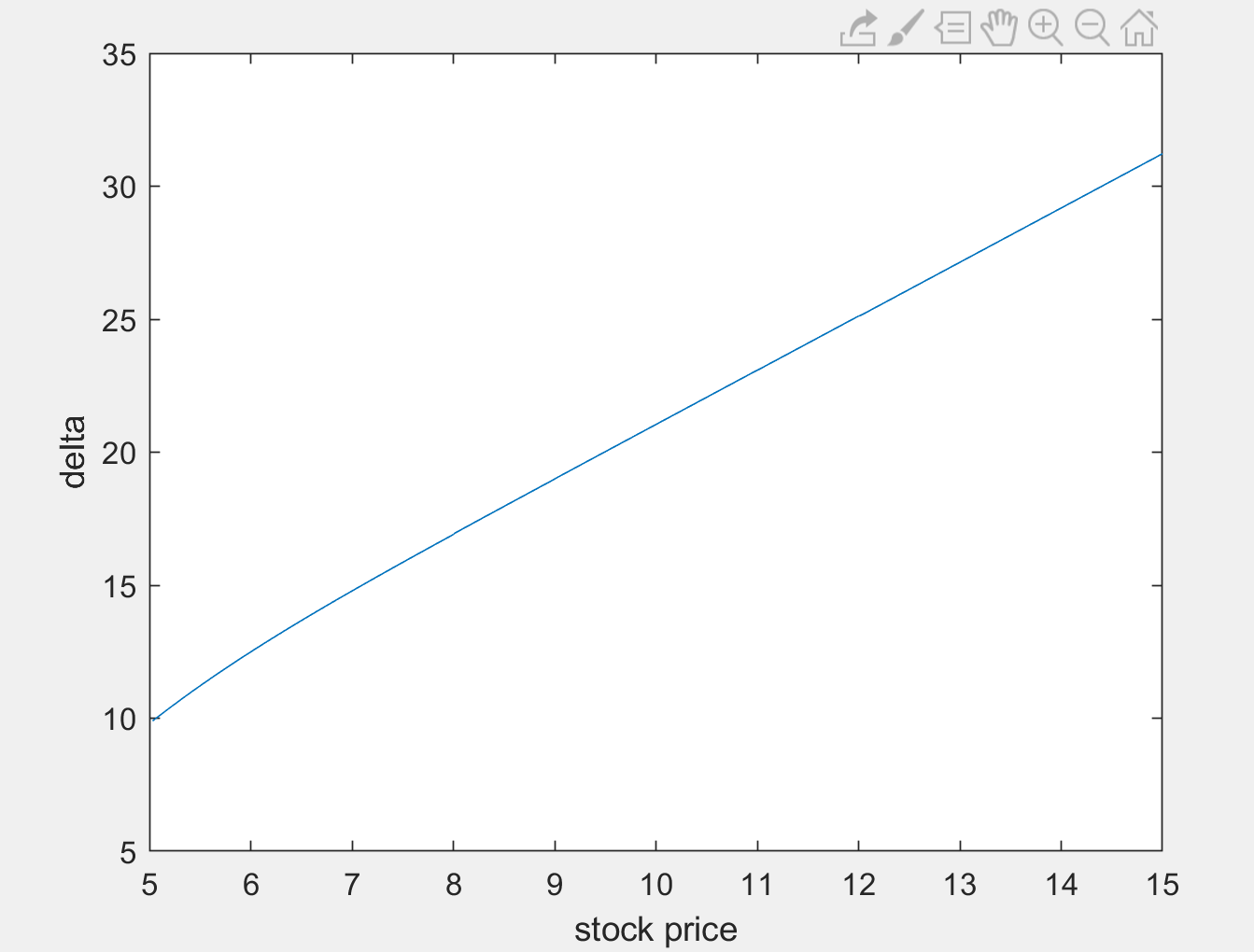


Figure2 . the plot delta varied with stock price

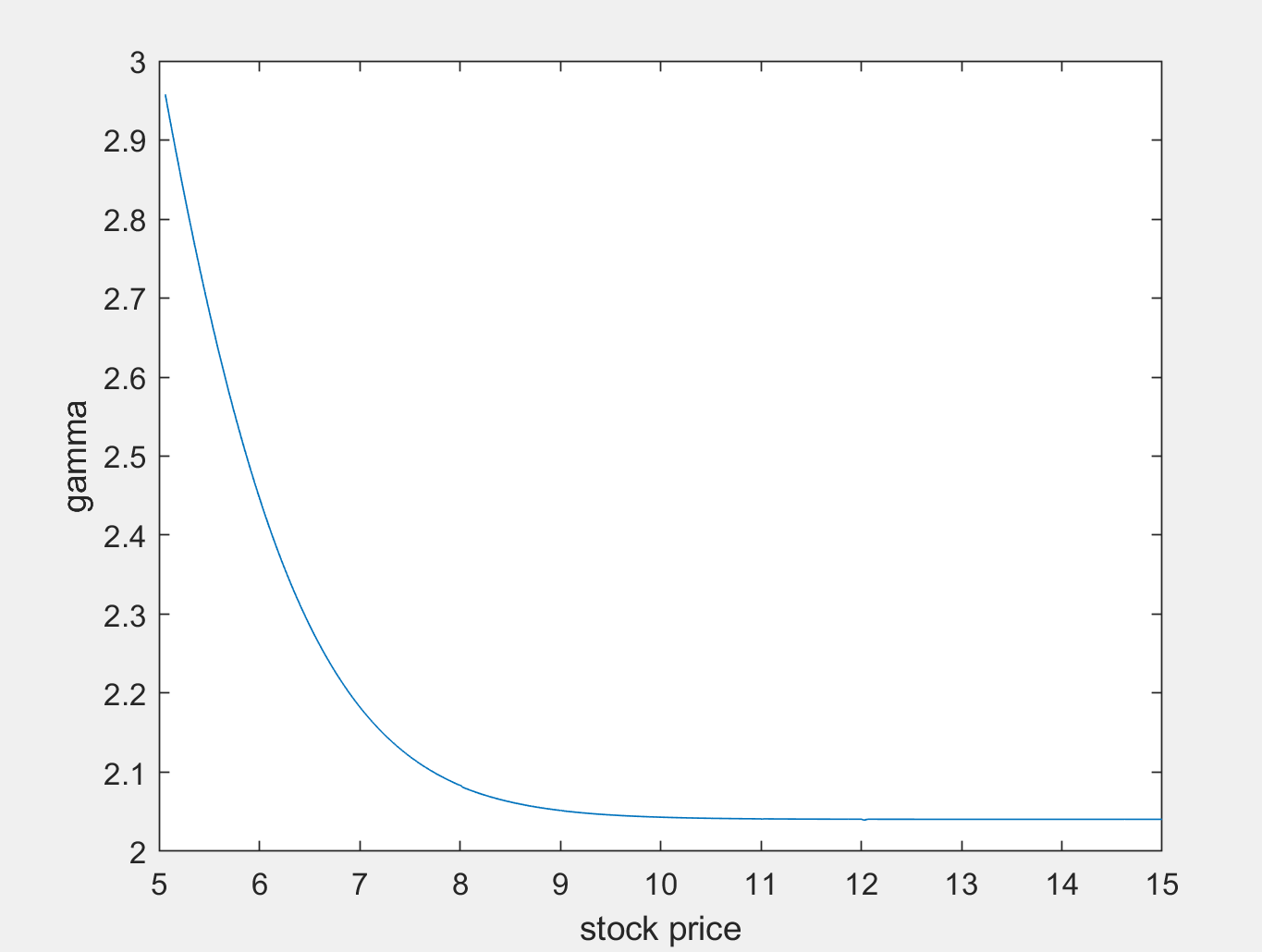


Figure2 . the plot gamma varied with stock price

From those plots, we can see that by using CN-Rannacher timestepping, I don’t see any no oscillation in delta and gamma. And the option price plot quadratically increases, although the delt plot is almost linear, but not exactly linear. As for the gamma, the plot decreases suddenly and converge to about 2.