A1.3)

i)

cfl.m

%cfl(r,q,s,sigma,t)

%if option is initiated at t=0, set smin=s

%else input the min of s

function cfl(r,q,s,smin,sigma,t)

a1=(log(s/smin)+t\*(r-q+0.5\*sigma^2))/(sigma\*sqrt(t));

a2=a1-sigma\*sqrt(t);

a3=(log(s/smin)+(-r+q+0.5\*sigma^2)\*t)/(sigma\*sqrt(t));

y1=-(log(s/smin)\*2\*(r-q-0.5\*sigma^2))/sigma^2;

part1=s\*exp(-q\*t)\*normcdf(a1);

part2=s\*exp(-q\*t)\*normcdf(-a1)\*(sigma^2)/(2\*(r-q));

part3=s\*exp(-r\*t)\*(normcdf(a2)-exp(y1)\*normcdf(-a3)\*(sigma^2)/(2\*(r-q)));

cfl=part1-part2-part3

return

ii)

cfl\_bm.m

function cfl\_bm(r,q,s,sigma,t,N)

dt=t/N;

u=exp(sigma\*sqrt(dt));

d=1/u;

disc = exp(-r\*dt);

p = (exp((r-q)\*dt)-d)/(u-d);

n=N:-1:0;

W=ones(1,N+1)- d'.^n; %Value of option at terminal

for i=N-1:-1:0

k=1:1:i;

temp=disc\*(p\*u\*W(i(1)+1)+(1-p)\*d\*W(i(1)+2));

W=disc\*(p\*u\*W(k)+(1-p)\*d\*W(k+2));

W(i+1)=temp;

end

cfl\_bm=W(1)\*s

return

iii)

BM

>> cfl\_bm(0.02,0.01,5,0.35,0.5,50)

cfl\_bm =

**0.8515**

>> cfl\_bm(0.02,0.01,5,0.35,0.5,500)

cfl\_bm =

**0.8968**

>> cfl\_bm(0.02,0.01,5,0.35,0.5,5000)

cfl\_bm =

**0.9119**

>> cfl\_bm(0.02,0.01,5,0.35,0.5,10000)

cfl\_bm =

**0.9139**

Actual

>> cfl(0.02,0.01,5,0.35,0.5)

cfl =

**0.9189**

Comment

As the number of iterations increases the value obtained from the binomial method converges towards the value produced by the closed formula in i). However, the convergence rate decreases as the number of iterations increases.

iv)

clf\_bm3.m

% cfl\_bm2(r,q,s,smin,sigma,t,N)

%N is number of time periods

function cfl\_bm3(r,q,s,smin,sigma,t,N)

x=min(s,smin)/s;

dt=t/N;

u=exp(sigma\*sqrt(dt));

d=1/u;

k=abs(ceil(log(x)/log(u)));

disc = exp(-r\*dt);

p = (exp((r-q)\*dt)-d)/(u-d);

j=(N+k+1):-1:max(0,k-N);%Set indices for Terminal value

setx=d.^j;

W=ones(1,length(j))- setx; %Value of option at terminal

for i=N-1:-1:0

if k-i<=0 %Check for terminal lower bound

temp = disc\*(p\*u\*W(length(W)-1)+(1-p)\*d\*W(length(W))); %Calculate lower bound if it exists

end

j=1:1:length(W)-2;

W=disc\*(p\*u\*W(j)+(1-p)\*d\*W(j+2));

if k-i<=0

W(length(W)+1)=temp; %Input the lower bound value

end

end

z=length(setx);

cfl\_bm3=(x-setx(z-k-1))/(setx(z-k)-setx(z-k-1))\*(s\*W(2))+(setx(z-k)-x)/(setx(z-k)-setx(z-k-1))\*(s\*W(1))

return

>> cfl\_bm3(0.02,0.01,5,4.75,0.35,0.5,50)

cfl\_bm3 =

0.8838

>> cfl\_bm3(0.02,0.01,5,4.75,0.35,0.5,500)

cfl\_bm3 =

0.9189

>> cfl\_bm3(0.02,0.01,5,4.75,0.35,0.5,5000)

cfl\_bm3 =

0.9311

>> cfl\_bm3(0.02,0.01,5,4.75,0.35,0.5,10000)

cfl\_bm3 =

0.9328

Actual

>> cfl(0.02,0.01,5,4.75,0.35,0.5)

cfl =

0.9369

Comment

The option price obtained from the binomial method is quite close to the price obtained from the closed formula. The price could possibly be closer if a quadratic interpolation is used instead. Also, note the slowing rate of increment as the number of iterations increases.

v)

cfl\_bm\_am.m

>> cfl\_bm\_am(0.02,0.01,5,4.75,0.35,0.5,50)

cfl\_bm\_am =

0.8838

>> cfl\_bm\_am(0.02,0.01,5,4.75,0.35,0.5,500)

cfl\_bm\_am =

0.9189

>> cfl\_bm\_am(0.02,0.01,5,4.75,0.35,0.5,5000)

cfl\_bm\_am =

0.9311

>> cfl\_bm\_am(0.02,0.01,5,4.75,0.35,0.5,10000)

cfl\_bm\_am =

0.9328

Comments

The values of the American option are very close to the European ones. This could be because that there is not much additional value to exercise the option early as the payoff is: max(S-Smin,0).

A 1.4)

i)

aafsput.m

>> aafsput(0.03,0.02,7.5,0.45,0.25,5)

aafput =

0.3776

Elapsed time is 0.001077 seconds.

>> aafsput(0.03,0.02,7.5,0.45,0.25,10)

aafput =

0.3781

Elapsed time is 0.000706 seconds.

>> aafsput(0.03,0.02,7.5,0.45,0.25,15)

aafput =

0.3790

Elapsed time is 0.004672 seconds.

ii)

>> fsg\_aafsput(0.03,0.02,7.5,0.45,0.25,5,0.5)

ans =

0.3793

Elapsed time is 0.002566 seconds.

>> fsg\_aafsput(0.03,0.02,7.5,0.45,0.25,10,0.5)

ans =

0.3828

Elapsed time is 0.001521 seconds.

>> fsg\_aafsput(0.03,0.02,7.5,0.45,0.25,15,0.5)

ans =

0.3839

Elapsed time is 0.003173 seconds.

>> fsg\_aafsput(0.03,0.02,7.5,0.45,0.25,100,0.5)

ans =

0.3840

Elapsed time is 0.257100 seconds.

>> fsg\_aafsput(0.03,0.02,7.5,0.45,0.25,200,0.5)

ans =

0.3834

Elapsed time is 1.711905 seconds.

Comment

The Forward Shooting grid method gives a much more accurate result as compared to the standard binomial tree. Further, although the computation time for 15 iterations and below is ionger as compared to the binomial tree, the computation time for 100 iterations and above is much shorter than that of the binomial tree.