Comsats Institute for Information Technology
Maths Department
April-May 2012
Stochastic Calculus and Finance

LASTNAME : Date : .
FIRSTNAME : Group : .

Answer-sheet n°2
Limiting price in a n time-steps CRR model when n tends to infinity and the Black-Scholes price

Please answer as clearly as possible in the provided space. The sheets will be collected at the end of the session.

Let us first enter the CRR model as a function of `n` :

```
//The CRR random walk as a function of n //
clear;
T=1; Nmax=500; S0=140; sigma=0.2; K=S0*0.9;
function delta_t=delta_t(n);
    delta_t=T./n ;//mind the "./"
endfunction;
function up=up(n);
    up=exp(sigma*sqrt(delta_t(n)));
endfunction;
function down=down(n)
    down=exp(-sigma*sqrt(delta_t(n)));
endfunction;
function p=p(n); //risk-neutral probability
    p=(1-down(n))./(up(n)-down(n));// mind the "./"
endfunction;
function s=S(n,i,j);
    s=S0*up(n)^j.*down(n)^(i-j);
endfunction;
// Call option
function c=phi(S);
    c=max(S-K,0);
endfunction;
```

What is the biggest value of `S` for `n=Nmax`? and what is the smallest.

Here the code that allows to compute the Call price by backward induction :

```
// backward induction computation.
CC=zeros(Nmax,Nmax+1,Nmax+1);
for n=1 :Nmax
    CC(n,n+1,1 :n+1)=phi(S(n,n,0 :n)); //price wellknown at the end.
    for i=n-1 :-1 :0 //downward induction
        for j=0 :i;
            CC(n,i+1,j+1)=p(n)*CC(n,i+2,j+1+1)+(1-p(n))*CC(n,i+2,j+1);
        end ;
    end ;
end;
```

```
function c=CallFromMatrix(n,i,j);//allows CallFromMatrix(n,0,0)
    c=CC(n,i+1,j+1)
endfunction;
// disp(CallFromMatrix(Nmax,0,0));
```

What is the price of the Call option for `n=2`, `n=5`, `n=50`, `n=Nmax` ? Does it seems to converge ?

Ask `Scilab` for the value of `binomial(0.4,10)` . Possibly using `plot(0 :10,binomial(0.4,10)` guess what is the purpose of function `binomial(p,n)` .

What is the difference between `binomial(0.4,10)'` and `binomial(0.4,10)` ?

Here the code for computing the same Call price, but using the binomial expectation formula

```
// the binomial expectation computation
function c=Call(n)
    c=binomial(p(n),n)*phi(S(n,n,0 :n))'
endfunction;
//price at time=0
C=zeros(1,Nmax);
for n=1 :Nmax;
    C(1,n)=Call(n);
end;
```

Please comment the formula used :

Please enter `plot(20 :Nmax,C(1,20 :Nmax)` . Does it seem to converge ? Comment.

Recall that the Black-Scholes price is given by

$$C = S\mathcal{N}(d_1) - Ke^{-rT}\mathcal{N}(d_2),$$

where $\mathcal{N}(d) := \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{d} e^{-\frac{x^2}{2}} dx$ is the Gaussian function, and with

$$d_1 = \frac{1}{\sigma\sqrt{T}} \left[\ln \frac{S_0}{K} + T \left(r + \frac{\sigma^2}{2} \right) \right] \text{ and } d_2 = d_1 - \sigma\sqrt{T}.$$

Unfortunately, `Scilab` does not provide the Gaussian function, but the so-called *error function* `erf`. Look at the online help for its definition and show that $\mathcal{N}(x) = \frac{1}{2}\left(\mathtt{erf}\left(\frac{x}{\sqrt{2}}\right) + 1\right)$.

Enter the following code for computing the Black-Scholes price and plot it in red on top of the values obtained with the CRR model for various n. Can you figure out what is the convergence rate[1] α (a descrepency of order $\frac{1}{n^\alpha}$) ?

```
// Normal distribution and erf (error function) //
function y=N(x);
    y=(erf(x/sqrt(2))+1)/2;
endfunction;
function BS=BlackScholes(S,K,r,T,sigma);
    d1=(log(S/K)+T*(r+(sigma^2)/2))/(sigma*sqrt(T));
    d2=d1-sigma*sqrt(T);
    BS=S*N(d1)-K*exp(-r*T)*N(d2);
endfunction;
CallBS=BlackScholes(S0,K,0,T,sigma);
plot(1 :Nmax,CallBS,'-r');
```

```
Keep a souvenir of this experience with
xs2eps(gcf(),'BSversusCRR.eps'); :-)
```

[1] for more on this, see *Francine Diener, Marc Diener,* Asymptotics of the price oscillations of a European call option in a tree model, *Mathematical Finance,* Vol. 14-3, pp 271-293 (2004).