MATLAB HW11

1.AssetPaths

```
function SPaths=AssetPaths(S0,mu,sigma,T,NSteps,NRepl)

SPaths = zeros(NRepl, 1+NSteps);
SPaths(:,1) = S0;
dt = T/NSteps;
nudt = (mu-0.5*sigma^2)*dt;
sidt = sigma*sqrt(dt);

for i=1:NRepl
    for j=1:NSteps
        SPaths(i,j+1)=SPaths(i,j)*exp(nudt + sidt*randn);
    end
end
```

2. AssetPaths1

```
function SPaths=AssetPaths1(S0,mu,sigma,T,NSteps,NRepl)
dt = T/NSteps;
nudt = (mu-0.5*sigma^2)*dt;
sidt = sigma*sqrt(dt);

Increments = nudt + sidt*randn(NRepl, NSteps);
LogPaths = cumsum([log(S0)*ones(NRepl,1) , Increments] , 2);
SPaths = exp(LogPaths);
```

3.DOPut

```
function P = DOPut(S0,X,r,T,sigma,Sb)

a = (Sb/S0)^(-1 + (2*r / sigma^2));
b = (Sb/S0)^(1 + (2*r / sigma^2));
d1 = (log(S0/X) + (r+sigma^2 / 2)* T) / (sigma*sqrt(T));
d2 = (log(S0/X) + (r-sigma^2 / 2)* T) / (sigma*sqrt(T));
d3 = (log(S0/Sb) + (r+sigma^2 / 2)* T) / (sigma*sqrt(T));
d4 = (log(S0/Sb) + (r-sigma^2 / 2)* T) / (sigma*sqrt(T));
d5 = (log(S0/Sb) - (r-sigma^2 / 2)* T) / (sigma*sqrt(T));
d6 = (log(S0/Sb) - (r+sigma^2 / 2)* T) / (sigma*sqrt(T));
d7 = (log(S0*X/Sb^2) - (r-sigma^2 / 2)* T) / (sigma*sqrt(T));
```

```
d8 = (log(S0*X/Sb^2) - (r+sigma^2 / 2)* T) / (sigma*sqrt(T));

P = X*exp(-r*T)*(normcdf(d4)-normcdf(d2) - ...
    a*(normcdf(d7)-normcdf(d5))) ...
    - S0*(normcdf(d3)-normcdf(d1) - ...
    b*(normcdf(d8)-normcdf(d6)));
```

4.CVDOPutMC

```
function [P,std,CI] = CVDOPutMC(S0,Sb,X,r,T,sigma,NSteps,NRepl,NPilot)
Payoff = zeros(NPilot,1);
VanillaPayoff = zeros(NPilot,1);
[aux,muVanilla] = blsprice(S0,X,r,T,sigma);
for i=1:NPilot
    Path = AssetPaths(S0,r,sigma,T,NSteps,1);
    VanillaPayoff(i) = max(0,X-Path(NSteps+1));
    crossed = any(Path<=Sb);
    if crossed == 0
        Payoff(i) = max(0,X-Path(NSteps+1));
    end
end
    VanillaPayoff = exp(-r*T)*VanillaPayoff;
    Payoff = exp(-r*T)*Payoff;

    covMat = cov(VanillaPayoff,Payoff);
    varVanilla = var(VanillaPayoff);
    c = -covMat(1,2)/varVanilla;

    newPayoff = zeros(NRepl,1);
    newVanillaPayoff = zeros(NRepl,1);
    for i = 1:NRepl
        Path = AssetPaths(S0,r,sigma,T,NSteps,1);
        newVanillaPayoff(i) = max(0,X-Path(NSteps+1));
        crossed = any(Path<=Sb);
        if crossed == 0
            newPayoff(i) = max(0,X-Path(NSteps+1));
        end
    end
    newVanillaPayoff = exp(-r*T)*newVanillaPayoff;
```

```
    newPayoff = exp(-r*T)*newPayoff;
    CVpayoff = newPayoff + c*(newVanillaPayoff - muVanilla);
    [P,std,CI] = normfit(CVpayoff);
```

5.CompCVDOPutMC

```
S0=50;
X=50;
r=0.1;
T=5/12;
sigma=0.4;
Sb1=40;
Sb2=20;
Sb3=1;
NSteps=365;
NPilot=5000;
NRepl=1;

[C,P] = blsprice(S0,X,r,T,sigma);

CVDOPutMC40 = CVDOPutMC(S0,Sb1,X,r,T,sigma,NSteps,NRepl,NPilot);
CVDOPutMC30 = CVDOPutMC(S0,Sb2,X,r,T,sigma,NSteps,NRepl,NPilot);
CVDOPutMC1 = CVDOPutMC(S0,Sb3,X,r,T,sigma,NSteps,NRepl,NPilot);

[C,P]
CVDOPutMC40
CVDOPutMC30
CVDOPutMC1
```

結果

```
ans =
    6.1165    4.0760

CVDOPutMC30 =
    4.0594

CVDOPutMC1 =
    4.0760
```

6.DOPutHalton

```
function Price = DOPutHalton(S0,X,r,T,sigma,NPoints,Base1,Base2,Sb)
nuT=(r-0.5*sigma^2)*T;
siT=sigma*sqrt(T);

%use box muller to generate standard normals
H1=GetHalton(ceil(NPoints/2),Base1);
H2=GetHalton(ceil(NPoints/2),Base2);
VLog=sqrt(-2*log(H1));
Norm1 = VLog.*cos(2*pi*H2);
Norm2 = VLog.*sin(2*pi*H2);
Norm = [Norm1;Norm2];

%generate asset paths
Payoff=zeros(NPoints,1);
NCrossed = 0;
Path=S0*exp(nuT+siT*Norm)
for i=1:NPoints
    crossed=any(Path<=Sb);
    if crossed == 0
        Payoff(i)=max(0,X-Path(NPoints));
    else
        Payoff(i)=0;
        NCrossed = NCrossed+1;
    end
end
Price = mean(exp(-r*T)*Payoff);
```

7.GetHalton

```
function Seq = GetHalton(HowMany,Base)
Seq = zeros(Howany,1);
%use ceil to find the approximate real solution
NumBits = 1+ceil(log(HowMany)/log(Base));
VetBase = Base.^(-(1:NumBits));
WorkVet = zeros(1,NumBits);
for i = 1:HowMany
    %increment last bit and carry over if necessary
    j=1;
    ok = 0;
```

```
        while ok == 0;
            WorkVet(j)=WorkVet(j)+1;
            if WorkVet(j)<Base
                ok = 1;
            else
                WorkVet(j)=0;
                j=j+1;
            end
        end
            Seq(i)=dot(WorkVet,VetBase);
end
```

8.ComputeDOPutHalton

```
S0=50;
X=50;
r=0.1;
T=2/12;
sigma=0.4;
Sb=30;
NPoints=10;
Base1=2;
Base2=3;
Price=DOPutHalton(S0,X,r,T,sigma,NPoints,Base1,Base2,Sb)
```

結果

```
Path =
    45.5687
    43.7898
    55.1603
    36.6868
    51.5653
    59.2560
    39.6414
    54.3246
    56.2191
    42.9242

Price1 =
     6.9588
```

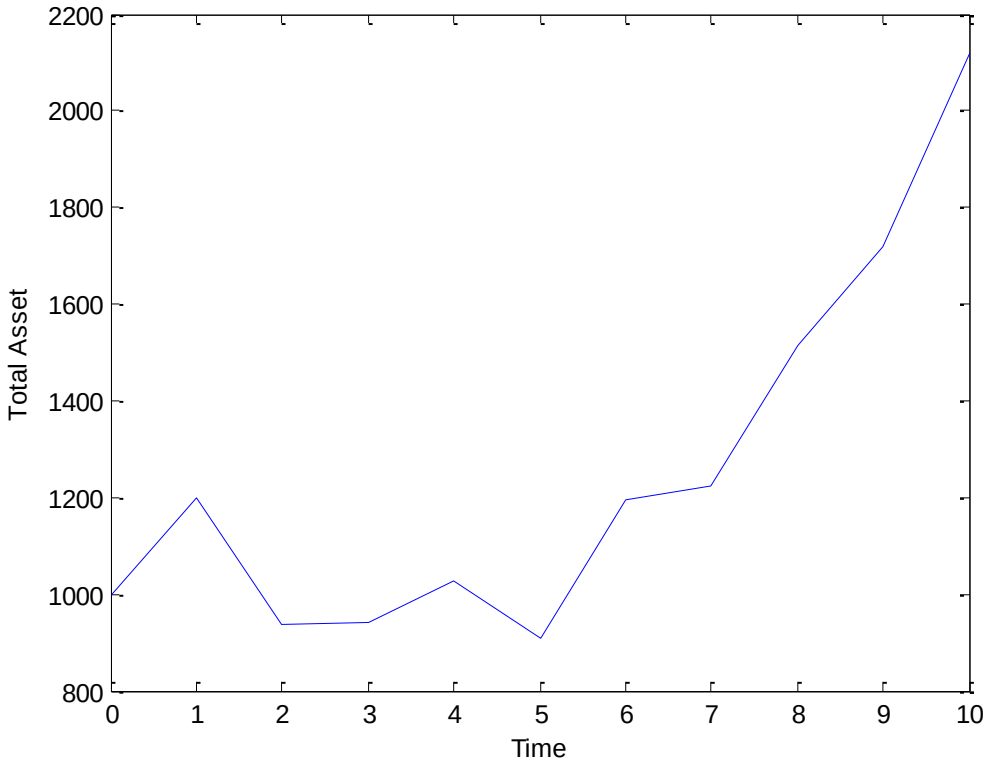
9.lockin

```
function [AssetR,Asset]=lockin(S0,r,T,sigma,risklv,riskv)
randn('seed',0);
Asset=zeros(1,T+1);
Asset(1)=riskv+risklv;
nuT=(r-0.5*sigma^2)*1;
siT=sigma*sqrt(1);
S=S0*exp(nuT+siT*randn(T,1));
n=riskv/S0;
for i=1:T

    risklv=risklv*((1+r)^1);

    riskv=S(i)*n;

    Asset(i+1)=riskv+risklv;

    if Asset(i+1)>1000
        temp=Asset(i+1)/((1+r)^(T-i));
        riskv=riskv-(temp-risklv);
        risklv=temp;
    end
end
AssetR=Asset(T+1)-max(Asset(1:T));
plot(0:T,Asset);
xlabel('Time');
ylabel('Total Asset');
```

10.InterestLockin

```
%interestlockin
S0=30;
T=10;
r=0.1;
sigma=0.25;
riskla=700;
riska=300;
[AssetR,Asset]=lockin(S0,r,T,sigma,riskla,riska)
```

%1.0e+003 * = 1000


Total Asset
Time
2200
2000
1800
1600
1400
1200
1000
800
0
1
2
3
4
5
6
7
8
9
10