Homework # 5

Pittsburgh
Jingyi Guo

1. Stratification

(a) standard Monte Carlo Simulation
Code:

```
S0=100;
sig=0.2;
T=1;
r=0.05;
K=100;
n=1000;

% a
U=rand(n,2);
Z=norminv(U);
S1=S0*exp((r-1/2*sig^2)*T+sig*sqrt(T).*Z(:,1));
S2=S0*exp((r-1/2*sig^2)*T+sig*sqrt(T).*Z(:,2));
C_a=exp(-r*T)*max(1/2*(S1+S2)-K,0);
C_a_bar=mean(C_a)
C_a_stderr=std(C_a)/sqrt(n)
```

Result:

```
C_a_bar =

    8.3984

C_a_stderr =

    0.3282
```

So the estimate is 8.3984, its standard error is 0.3282.

(b) bivariate stratification
Code:

```
% generate 10 points within each bin
C_b=[];
sqrerr_b=[];
for i=1:10
    for j=1:10
        % in bin (i-1)/10~i/10 , (j-1)/10~j/10)
```

```
            u=rand(10,2);
            u(:,1)=(i-1)/10+1/10.*u(:,1);
            u(:,2)=(j-1)/10+1/10.*u(:,2);
            z=norminv(u);
            s1=S0*exp((r-1/2*sig^2)*T+sig*sqrt(T).*z(:,1));
            s2=S0*exp((r-1/2*sig^2)*T+sig*sqrt(T).*z(:,2));
            C_temp=exp(-r*T)*max(1/2*(s1+s2)-K,0);
            C_b=[C_b;C_temp];
            % append square error for each bin
            sqrerr_b=[sqrerr_b;(std(C_temp))^2];
        end
    end
    C_b_bar=mean(C_b)
    C_b_stderr=sqrt(1/100*sum(sqrerr_b))/sqrt(1000)
```

Result:

C_b_bar =

8.4608

C_b_stderr =

0.1014

So the estimate is 8.4608, its standard error is 0.1014.

(c) stratification of a projection

Code:

```
    C_c=[];
    sqrerr_c=[];
    for i=1:250
        % each bin: (i-1)/250~i/250
        nu=[1/sqrt(2);1/sqrt(2)];
        for k=1:4 % sample size n=4 in each bin
            u=(i-1)/250+1/250.*rand(1);
            x=norminv(u);
            X=[randn(1),x]; % pair (Z,X=x)
            Mu=x*nu;
            Sigma=eye(2)-nu*nu';
            z=mvnrnd(Mu,Sigma);
            s1=S0*exp((r-1/2*sig^2)*T+sig*sqrt(T)*z(1));
            s2=S0*exp((r-1/2*sig^2)*T+sig*sqrt(T)*z(2));
            C_temp=exp(-r*T)*max(1/2*(s1+s2)-K,0);
            C_c=[C_c;C_temp];
```

```
    end
    % append square error for each bin
    sqrerr_c=[sqrerr_c;(std(C_c((i-1)*4+1:(i-1)*4+4)))^2];
end
C_c_bar=mean(C_c)
C_c_stderr=sqrt(1/250*sum(sqrerr_c))/sqrt(1000)
```

Result:

C_c_bar =

8.2953

C_c_stderr =

0.0390

Compare the results in (a)(b)(c), I found that stratification of a projection has the least standard error, while standard Monte Carlo simulation has the largest standard error of the three methods.

2. Brownian bridge method

(a) Examine Table1

Code:

```
T=0.25;
N=30;
n=1000;
S0=50;
mu=0.15;
sig=0.25;
r=0.1;
K=S0;

% standard Monte Carlo
delta=T/N;
S_a1=zeros(N+1,n);
S_a1(1,:)=S0*ones(1,n);

for j=2:(N+1)
    z=randn(1,n);
    S_a1(j,:)=S_a1(j-1,:)+r*delta*S_a1(j-
1,:)+sqrt(delta)*sig*S_a1(j-1,:).*z;
end

C_a1=exp(-r*T)*max(max(S_a1)-S0,0);
```

```
Cbar_a1=mean(C_a1)
Cstderr_a1=std(C_a1)/sqrt(n)

M=zeros(N,n);
% brownian bridge
for j=1:N
    % consider time period j*delta,(j+1)*delta
    b=(S_a1(j+1,:)-S_a1(j,:))./(sig.*S_a1(j,:)); % B_end
    u=rand(1,n);
    maxB=(b+sqrt(b.^2-2*delta*log(1-u)))/2;
    M(j,:)=S_a1(j,:)+sig*S_a1(j,:).*maxB;
end

C_a2=exp(-r*T)*max(max(M)-S0,0);
Cbar_a2=mean(C_a2)
Cstderr_a2=std(C_a2)/sqrt(n)
```

Result:

	Standard Simulation	Brownian bridge
Estimate	4.9387	5.5952
Standard Error	0.1280	0.1298

(b) $K=S_T$ scenario

Code:

```
T=0.25;
N=30;
n=1000;
S0=50;
mu=0.15;
sig=0.25;
r=0.1;

% standard Monte Carlo
delta=T/N;
S_b1=zeros(N+1,n);
S_b1(1,:)=S0*ones(1,n);

for j=2:(N+1)
    z=randn(1,n);
    S_b1(j,:)=S_b1(j-1,:)+r*delta*S_b1(j-
1,:)+sqrt(delta)*sig*S_b1(j-1,:).*z;
end
```

```
C_b1=exp(-r*T)*max(max(S_b1)-S_b1(N+1,:),0);
Cbar_b1=mean(C_b1)
Cstderr_b1=std(C_b1)/sqrt(n)

M=zeros(N,n);
% brownian bridge
for j=1:N
    % consider time period j*delta,(j+1)*delta
    b=(S_b1(j+1,:)-S_b1(j,:))./(sig.*S_b1(j,:)); % B_end
    u=rand(1,n);
    maxB=(b+sqrt(b.^2-2*delta*log(1-u)))/2;
    M(j,:)=S_b1(j,:)+sig*S_b1(j,:).*maxB;
end

C_b2=exp(-r*T)*max(max(M)-S_b1(N+1,:),0);
Cbar_b2=mean(C_b2)
Cstderr_b2=std(C_b2)/sqrt(n)
```

Result:

	Standard Simulation	Brownian bridge
Estimate	3.6710	4.3202
Standard Error	0.1042	0.1047

(c) Knock out option

Code:

```
T=0.25;
N=30;
n=100000;
S0=50;
mu=0.15;
sig=0.5;
r=0.1;
K=50;
H=45;

% standard Monte Carlo
delta=T/N;
S_c1=zeros(N+1,n);
S_c1(1,:)=S0*ones(1,n);

for j=2:(N+1)
    z=randn(1,n);
```

```
    S_c1(j,:)=S_c1(j-1,:)+r*delta*S_c1(j-
1,:)+sqrt(delta)*sig*S_c1(j-1,:).*z;
end

C_c1=zeros(1,n);
minS=min(S_c1);

for i=1:n
    if (minS(i)>H)
        C_c1(i)=exp(-r*T)*max(S_c1(N+1,i)-K,0);
    end
end
Cbar_c1=mean(C_c1)
Cstderr_c1=std(C_c1)/sqrt(n)

Mm=zeros(N,n);
% brownian bridge
for j=1:N
    % consider time period j*delta,(j+1)*delta
    b=(S_c1(j+1,:)-S_c1(j,:))./(sig.*S_c1(j,:)); % B_end
    u=rand(1,n);
    minB=(b-sqrt(b.^2-2*delta*log(1-u)))/2;
    Mm(j,:)=S_c1(j,:)+sig*S_c1(j,:).*minB;
end

C_c2=zeros(1,n);
minMm=min(Mm);
for i=1:n
    if (minMm(i)>H)
        C_c2(i)=exp(-r*T)*max(S_c1(N+1,i)-K,0);
    end
end
Cbar_c2=mean(C_c2)
Cstderr_c2=std(C_c2)/sqrt(n)
```

Result:

	Standard Simulation	Brownian bridge
Estimate	4.5579	4.0416
Standard Error	0.0274	0.0266

3. Two-Asset Down-and-Out Call Option Pricing

(a) standard Monte Carlo simulation

Code:

```
S10=100;
S20=100;
K=100;
r=0.1;
sig1=0.3;
sig2=0.3;
rho=0.5;
T=0.2;
H=95;
n=10000;
N=50;

% a
S1store=zeros(N+1,n);
S2store=zeros(N+1,n);
S1store(1,:)=S10*ones(1,n); %initial value
S2store(1,:)=S20*ones(1,n); %initial value
delta=T/N;

for j=2:(N+1)
    z1=randn(1,n);
    z2=rho*z1+sqrt(1-rho^2)*randn(1,n);
    S1store(j,:)=S1store(j-1,:).*exp((r-
1/2*sig1^2)*delta+sig1*sqrt(delta).*z1);
    S2store(j,:)=S2store(j-1,:).*exp((r-
1/2*sig2^2)*delta+sig2*sqrt(delta).*z2);
end

minS2=min(S2store);
C1=zeros(1,n);
for i=1:n
    if (minS2(i)>95)
        C1(i)=exp(-r*T)*max(S1store(N+1,i)-K,0);
    end
end

C1_bar=mean(C1)
C1_stderr=std(C1)/sqrt(n)
```

Result:

Estimation	3.5887
Std. Err	0.0795

(b) Brownian bridge

Code:

```
Mm=zeros(N,n);
for j=1:N
    % consider time period j*delta,(j+1)*delta
    b=(S2store(j+1,:)-S2store(j,:))./(sig2.*S2store(j,:)); %
B_end
    u=rand(1,n);
    minB=(b-sqrt(b.^2-2*delta*log(1-u)))/2;
    Mm(j,:)=S2store(j,:)+sig2*S2store(j,:).*minB;
end

minMm=min(Mm);

C2=zeros(1,n);
```

Result:

Estimation	3.0899
Std. Err	0.0753

4. Credit Derivatives and Copulas

Code:

```
clc
clear
N=5;
T=5;
r=0.04;
s=0.01;
R=0.35;
lam=s/(1-R);
corr=0;   % or 0.2, 0.4, 0.6, 0.8
itd=5;  % or 2, 3, 4, 5
n=100000;

%Gaussian Copula
Sigma=corr*ones(5)+(1-corr)*eye(5);
A=chol(Sigma,'lower');
z=randn(5,n);
```

```
y=A*z; % or y=z if rho=1

U=normcdf(y);
% exp(lam) cdf: 1-exp(-lam*x)
X=-1/lam*log(1-U);

count=zeros(1,n);
C=zeros(1,n);
for i=1:n
    count(i)=sum(X(:,i)<=5);
    if count(i)>=itd
        C(i)=exp(-r*T)*(1-R);
    end
end

C_bar=mean(C)
C_stderr=std(C)/sqrt(n)


%% if rho=1
% comment all above
N=5;
T=5;
r=0.04;
s=0.01;
R=0.35;
lam=s/(1-R);
corr=1;
itd=1;  % or 2, 3, 4, 5
n=100000;

%Gaussian Copula
z=randn(1,n);
y=[z;z;z;z;z];

U=normcdf(y);
% exp(lam) cdf: 1-exp(-lam*x)
X=-1/lam*log(1-U);

count=zeros(1,n);
C=zeros(1,n);
for i=1:n
```

```
        count(i)=sum(X(:,i)<=5);
        if count(i)>=itd
            C(i)=exp(-r*T)*(1-R);
        end
    end

    C_bar=mean(C)
    C_stderr=std(C)/sqrt(n)
```

Result:

- Case $\rho = 0$:

	FtD	2tD	3tD	4tD	5tD
Estimate	0.1700	0.0253	0.0021	6.3861e-05	0
Std. Err	7.8468e-04	3.5821e-04	1.0516e-04	1.8434e-05	0

- Case $\rho = 0.2$:

	FtD	2tD	3tD	4tD	5tD
Estimate	0.1515	0.0371	0.0074	0.0010	9.5791e-05
Std. Err	7.5941e-04	4.2836e-04	1.9647e-04	7.4242e-05	2.2576e-05

- Case $\rho = 0.4$:

	FtD	2tD	3tD	4tD	5tD
Estimate	0.1330	0.0438	0.0149	0.0042	6.8118e-04
Std. Err	7.2863e-04	4.6272e-04	2.7768e-04	1.4871e-04	6.0170e-05

- Case $\rho = 0.6$:

	FtD	2tD	3tD	4tD	5tD
Estimate	0.1114	0.0497	0.0240	0.0102	0.0032
Std. Err	6.8464e-04	4.8973e-04	3.4898e-04	2.3106e-04	1.2943e-04

- Case $\rho = 0.8$:

	FtD	2tD	3tD	4tD	5tD
Estimate	0.0866	0.0505	0.0315	0.0195	0.0096
Std. Err	6.2133e-04	4.9298e-04	3.9725e-04	3.1651e-04	2.2405e-04

- Case $\rho = 1$:

	FtD	2tD	3tD	4tD	5tD
Estimate	0.0395	0.0392	0.0394	0.0396	0.0394
Std. Err	4.4108e-04	4.3957e-04	4.4081e-04	4.4171e-04	4.4072e-04