MATH 565 Monte Carlo Methods in Finance

Fred J. Hickernell **Fall 2005**

Final Exam **Due 12:30 PM, Thursday, December 15**

Instructions:

i. This exam consists of SIX questions. Answer all of them.

ii. This exam is open book and notes, but you must not consult with any other person verbally, by email, or by any other means.

iii. Show all your work to justify your answers. Answers without adequate justification will not receive credit. Include listings of your programs and output for the computational parts.

1. (5 marks)
Let X be a random variable defined on $(-\infty, \infty)$ with probability density function $\rho(x)$, and cumulative distribution function $R(x) = \int_{-\infty}^{x} \rho(t)\,dt$. Let $Y = f(X)$ for some known function f. Let E denote expectation. What is $E(Y)$ in terms of f, ρ, and/or R?

Answer:

$$E(Y) = \int_{-\infty}^{\infty} f(x)\rho(x)\,dx$$

2. (20 marks)
Consider again the situation in Problem 1. For given $X_1, \ldots, X_n$, let $Y_i = f(X_i)$ and

$$\bar{Y} := \frac{1}{n}\sum_{i=1}^{n} Y_i = \frac{1}{n}\sum_{i=1}^{n} f(X_i)$$

be the sample mean. The sample mean is an estimator of $E(Y)$. For the each of the following choices of X_i:

a. the X_i are i.i.d. with common distribution function R,

b. the X_i are i.i.d. with common distribution function $\tilde{R} \neq R$,

c. $X_i = R^{-1}((i - 1/2)/n)$, and

d. $X_i = R^{-1}((i - U_i)/n)$, where the U_i are i.i.d. uniform on $[0, 1]$,

determine

i) whether or not $\bar{Y}$ is an unbiased estimator of $E(Y)$, and

ii) the variance of $\bar{Y}$.

Which is more important — having small or zero bias, or having small variance? Explain why?

Answer: For X_i i.i.d. with common distribution function R, it follows that

$$E[\bar{Y}] = \frac{1}{n}\sum_{i=1}^{n} E[f(X_i)] = E[f(X_1)] = \int_{-\infty}^{\infty} f(x)\rho(x)\,dx = E(Y),$$

so $\bar{Y}$ is unbiased. Furthermore, because the X_i are i.i.d., it follows that

$$\operatorname{var}(\bar{Y}) = \frac{1}{n^2}\sum_{i=1}^{n}\operatorname{var}(f(X_i)) = \frac{1}{n}\operatorname{var}(f(X_1)) = \frac{1}{n}\operatorname{var}(Y),$$

where

$$\operatorname{var}(Y) = \int_{-\infty}^{\infty}[f(x)]^2\rho(x)\,dx - \left[\int_{-\infty}^{\infty} f(x)\rho(x)\,dx\right]^2$$

For X_i i.i.d. with common distribution function $\tilde{R} \neq R$, it follows that

$$E[\bar{Y}] = \frac{1}{n}\sum_{i=1}^{n}E[f(X_i)] = E[f(X_1)] = \int_{-\infty}^{\infty} f(x)\tilde{\rho}(x)\,dx \neq E(Y),$$

where $\tilde{\rho}$ is the density corresponding to $\tilde{R}$. Thus, $\bar{Y}$ is biased. Furthermore, because the X_i are i.i.d., it follows as in the preceding case that

$$\operatorname{var}(\bar{Y}) = \frac{1}{n}\left\{\int_{-\infty}^{\infty}[f(x)]^2\tilde{\rho}(x)\,dx - \left[\int_{-\infty}^{\infty} f(x)\tilde{\rho}(x)\,dx\right]^2\right\},$$

The points $X_i = R^{-1}((i-1/2)/n)$ are deterministic, so for this case, $\operatorname{var}(\bar{Y}) = 0$, and it follows that

$$E[\bar{Y}] = \bar{Y} = \frac{1}{n}\sum_{i=1}^{n} f(R^{-1}((i-1/2)/n)),$$

which is not equal to $E(Y)$ in general. Thus, this estimator is biased.

For $X_i = R^{-1}((i-U_i)/n)$, the estimator is again unbiased, since applying a change of variable,

$$\begin{aligned} E[\bar{Y}] &= \frac{1}{n}\sum_{i=1}^{n}E[f(X_i)] = \sum_{i=1}^{n}\frac{1}{n}\int_0^1 f(R^{-1}((i-u)/n))\,du \\ &= \sum_{i=1}^{n}\int_{R^{-1}((i-1)/n)}^{R^{-1}(i/n)} f(x)\rho(x)\,dx = \int_{-\infty}^{\infty} f(x)\rho(x)\,dx = E(Y). \end{aligned}$$

Although the $f(X_i)$ are not identically distributed, they are independent. Thus, the variance of this estimator is

$$\begin{aligned} \operatorname{var}[\bar{Y}] &= \frac{1}{n^2}\sum_{i=1}^{n}\operatorname{var}(f(X_i)) = \frac{1}{n^2}\sum_{i=1}^{n}\left\{E[(f(X_i))^2] - [E(f(X_i))]^2\right\} \\ &= \frac{1}{n^2}\sum_{i=1}^{n}\left\{\int_0^1[f(R^{-1}((i-u)/n))]^2\,du - \left[\int_0^1 f(R^{-1}((i-u)/n))\,du\right]^2\right\} \\ &= \frac{1}{n}\sum_{i=1}^{n}\left\{\int_{R^{-1}((i-1)/n)}^{R^{-1}(i/n)}[f(x)]^2\rho(x)\,dx - n\left[\int_{R^{-1}((i-1)/n)}^{R^{-1}(i/n)} f(x)\rho(x)\,dx\right]^2\right\}. \end{aligned}$$

The mean square error of the estimator $\bar{Y}$ is

$$E[E(Y) - \bar{Y}]^2 = [E(Y) - E(\bar{Y})]^2 + \mathrm{var}(\bar{Y}).$$

Therefore, it is desirable to have both small bias, $\left|E(Y) - E(\bar{Y})\right|$, and small variance, $\mathrm{var}(\bar{Y})$. Good random estimators, such as 2d, typically have zero bias and small variance. Good determistic estimators, such as 2c, have zero variance, but small bias.

3. (15 marks)
Again consider the situation in Problems 1 and 2. Suppose that X is a standard Gaussian random variable, i.e.,

$$\rho(x) = \frac{1}{\sqrt{2\pi}} e^{-x^2/2}.$$

Furthermore, suppose that f belongs to some Hilbert space with reproducing kernel

$$K(x, z) = 1 + \max(x, 0) + \max(z, 0) - \max(x, z).$$

Evaluate numerically the largest bias possible for $\bar{Y}$ assuming $\|f\|_K \leq 1$ and for the X_i defined in 2c. above.

Answer: Following the same procedeure as notes in class for the case of integration over the unit cube, one may derive an upper bound for the square bias:

$$\sup_{\|f\|_K \leq 1} \left|E(Y) - \bar{Y}\right|^2 = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} K(x,z)\rho(x)\rho(z)\,dx\,dz - \frac{2}{n}\sum_{i=1}^{n}\int_{-\infty}^{\infty} K(X_i,z)\rho(z)\,dz + \frac{1}{n^2}\sum_{i,k=1}^{n} K(X_i, X_k)$$

Noting that

$$\int_{-\infty}^{\infty} \max(x,z)\rho(z)\,dz = \int_{-\infty}^{x} x\rho(z)\,dz + \int_{x}^{\infty} z\rho(z)\,dz = xR(x) + \rho(x),$$

where $R(x)$ is the standard normal cumulative distribution function, it follows that

$$\begin{aligned}\int_{-\infty}^{\infty} K(x,z)\rho(z)\,dz &= 1 + \max(x,0) + \rho(0) - xR(x) - \rho(x)\\ &= 1 + \frac{1}{\sqrt{2\pi}} + \max(x,0) - xR(x) - \rho(x),\end{aligned}$$

$$\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} K(x,z)\rho(x)\rho(z)\,dx\,dz = 1 + \frac{2-\sqrt{2}}{\sqrt{2\pi}}.$$

For an arbitrary choice of X_i it follows that

$$\sup_{\|f\|_K \leq 1} \left|E(Y) - \bar{Y}\right|^2 = \frac{-1}{\sqrt{\pi}} + \frac{2}{n}\sum_{i=1}^{n}\left[X_i R(X_i) + \rho(X_i)\right] - \frac{1}{n^2}\sum_{i,k=1}^{n}\max(X_i, X_k)$$

Using the program below one obtains

$$\sup_{\|f\|_K \le 1} \left| E(Y) - \bar{Y} \right| = 0.0007879.$$

```
%MATH 565 Monte Carlo Methods in Finance
%Final exam Question 3
clear all, format long e, format compact
normcdf=inline('(1+erf(x/sqrt(2)))/2','x'); %normal cdf
invnorm=inline('sqrt(2)*erfinv(2*x-1)','x'); %inverse normal cdf

c=-1/sqrt(pi);
nvec=10.^[0:3];
disc=0;
for i=1:length(nvec);
    n=nvec(i);
    x=invnorm(([1:n]'-1/2)/n);
    intkvec=x.*normcdf(x) + exp(-x.*x/2)/(sqrt(2*pi));
    xmat=repmat(x,1,n);
    kmat=max(xmat,xmat');
    disc(i)=c+2*mean(intkvec)-mean(mean(kmat));
end
disc=sqrt(disc)
```

4. (20 marks)
Consider a principal of \$10,000 placed in a risk-free interest bearing account. The balance after 5 years is

$$B = 10^4 \exp\left(\int_0^5 r(t)\, dt \right),$$

where $r(t)$ is a random, time dependent interest rate. Suppose that $r(t)$ is a pure jump process, i.e., it is constant on the random intervals, $[0, T_1), [T_1, T_2), \ldots$. Specifically, $r(t) = r_j$ for $t \in [T_{j-1}, T_j)$ and the r_j are i.i.d. uniformly distributed on $[2\%, 4\%]$. Furthermore $T_1, T_2 - T_1, T_3 - T_2, \ldots$ are i.i.d. exponential variables with common mean 0.3 years.

   a. Construct a simple Monte Carlo algorithm to evaluate $E(B)$ to a relative accuracy of 1 cent on 10 dollars. What $E(B)$? How many samples are needed?
   b. Use antithetic variates for both r_j and $T_j - T_{j-1}$. Is there any improvement in the accuracy of your algorithm? Now how many samples are needed to achieve the same accuracy?

Answer: Note that one may write

$$T_j = \Delta_1 + \cdots + \Delta_j,$$
$$B = 10^4 \exp\left(r_1 \Delta_1 + \cdots + r_K \min(\Delta_K, 5 - T_K) \right)$$

where the Δ_j are i.i.d. exponential random variables, and K is the random variable satisfying $T_{K-1} < 5 \le T_K$. To generate the Δ_j and r_j, one uses the inverse transform method, i.e., $\Delta_j = -0.3 \log(U_j)$ and $r_j = 0.2 + 0.2 V_j$, where the U_j and V_j are i.i.d. uniform random

variables on $(0,1)$*. For simple Monte Carlo it takes about* $n = 400$ *runs to estimate the balance with* 0.1% *error.*

For the antithetic variates one has 4 potential estimators:

$$B_{\pm\pm} = 10^4 \exp\left(r_{\pm 1}\Delta_{\pm 1} + \cdots + r_{\pm K}\min(\Delta_{\pm K}, 5 - T_{\pm K})\right),$$
$$\Delta_{+j} = -0.3\log(U_j), \quad \Delta_{-j} = -0.3\log(1-U_j), \quad r_{+j} = 0.2 + 0.2V_j, \quad r_{-j} = 0.4 - 0.2V_j,$$

where the first $\pm$ *in the subscript of* B *goes with the interest rate and the second* $\pm$ *goes with the time. One may take any one or more of these four estimators and combine them. Suppose one chooses* k *of them, and let construct a Monte Carlo estimator based on the average of the sample means, each sample mean containing* n *points:*

$$\bar{B} = \frac{1}{k}\left(\bar{B}_1 + \cdots + \bar{B}_k\right), \qquad \mathrm{var}(\bar{B}) = \frac{1}{nk^2}\sum_{l,m}\mathrm{cov}(B_l, B_m) \approx \frac{1}{nk^2}\sum_{l,m}\widehat{\mathrm{cov}}(B_l, B_m),$$

where $\widehat{\mathrm{cov}}$ *denotes the smaple covariance. When trying the combinations of antithetic variates,*

$$(B_{++}, B_{--}), (B_{++}, B_{-+}), (B_{++}, B_{+-}), (B_{++}, B_{+-}, B_{-+}, B_{--}),$$

the second choice seems to give the best improvement on simple Monte Carlo. That is, taking an antithetic variable for interest helps a lot, but for time does not help much. This can be seen by looking at the sample covariance matrix and finding the large correlation between B_{++} *and* B_{-+}*, and also between* B_{+-} *and* B_{--}*, but not say, between* B_{++} *and* B_{+-}*. For this choice of antithetic variates it seems that only a handful of samples are necessary to get a relative error of less than* 0.1%*. The expected balance is about \$11,618.*

```
%MATH 565 Monte Carlo Methods in Finance
%Final exam Question 4
clear all, format long e, format compact

princip=1e4; %principal
Tmax=5; %time to maturity
rmin=0.02; rmax=0.04; %upper and lower bounds on interest rate

%Simple Monte Carlo
n=400; %number of scenarios
notyet=1:n; %indices for which sum of Delta < Tmax
nny=length(notyet);
totT=zeros(n,1); Delta=totT; r=totT; d=0; %initialize
while nny>0;
    d=d+1; %increase dimension (number of time steps
    tempDel=-0.3*log(rand(nny,1)); %generate exponential random variable with mean 0.3
    maxDel=Tmax-totT;
    tempDel=min(tempDel,maxDel); %cut off the last Delta
    Delta(notyet,d)=tempDel;
    r(notyet,d)=rmin+rand(nny,1)*(rmax-rmin); %generate interest rate
    totT=totT+tempDel; %update total time
    wh=find(totT<Tmax); %find total times not yet to Tmax
    totT=totT(wh); notyet=notyet(wh); nny=length(notyet); %update
end
B=princip*exp(sum(r.*Delta,2)); %balance for n scenarios
meanB=mean(B); %sample expected balance
esterr=1.96*std(B)/sqrt(n); %estimated error
estrelerr=esterr/meanB; %estimated relative error
```

```
disp(['Using ' num2str(n) ' runs of simple Monte Carlo the mean balance is $' num2str(meanB) ...
    ' + or - $' num2str(esterr)])
disp(['     for an error of ' num2str(100*estrelerr) ' cents on the dollar.'])
disp(' ');

%Antithetic variates
n=5;
notyet=1:n; %indices for which sum of Delta < Tmax
nny=length(notyet);
totT=zeros(n,1); Delta=totT; r=totT; d=0; %initialize
totTanti=totT; Deltaanti=totT; ranti=totT; %initialize antithetic counterparts
while nny>0;
    d=d+1; %increase dimension (number of time steps
    u=rand(nny,1); %get uniform random numbers first
    tempDel=-0.3*log(u); %generate exponential random variable with mean 0.3
    tempDelanti=-0.3*log(1-u); %and its antithetic counterpart
    maxDel=Tmax-totT; maxDelanti=Tmax-totTanti;
    tempDel=min(tempDel,maxDel); tempDelanti=min(tempDelanti,maxDelanti); %cut off the last Delta
    Delta(notyet,d)=tempDel; Deltaanti(notyet,d)=tempDelanti;
    u=rand(nny,1); %get uniform random numbers first
    r(notyet,d)=rmin+u*(rmax-rmin); %generate interest rates
    ranti(notyet,d)=rmin+(1-u)*(rmax-rmin); %and the antithetic counterparts
    totT=totT+tempDel; totTanti=totTanti+tempDelanti; %update total time
    wh=find(min(totT,totTanti)<Tmax);  %find total times not yet to Tmax
    totT=totT(wh); totTanti=totTanti(wh); notyet=notyet(wh); nny=length(notyet); %update
end
%Ball=princip*exp([sum(r.*Delta,2) sum(ranti.*Deltaanti,2)]);
Ball=princip*exp([sum(r.*Delta,2) sum(ranti.*Delta,2)]);
%Ball=princip*exp([sum(r.*Delta,2) sum(r.*Deltaanti,2)]);
%Ball=princip*exp([sum(r.*Delta,2) sum(ranti.*Delta,2) sum(r.*Deltaanti,2) sum(ranti.*Deltaanti,2)]);
[ans,nanti]=size(Ball); %number of antithetic combinations
ntot=nanti*n; %total number of scenarios
meanBall=mean(Ball,2); %average over different antithetics
meanBanti=mean(meanBall); %estimated balance
esterr=1.96*std(meanBall)/sqrt(n);
estrelerr=esterr/meanBanti;
disp(['Using ' num2str(ntot) ' total runs of antithetic variates the mean balance is $' num2str(meanBanti) ...
    ' + or - $' num2str(esterr)])
disp(['     for an error of ' num2str(100*estrelerr) ' cents on the dollar.'])
disp(' ');
```

5. (20 marks)
Consider a stock whose price follows a geometric Brownian motion with constant interest rate 3%, volatility 30% and initial price \$300. Let $S(i)$ denote the stock price after i weeks. Consider an option with a payoff

$$\text{payoff} = \max\left(\left[\frac{1}{10}\sum_{i=1}^{10}\min(\max(S(i),280),320)\right] - 300, 0\right).$$

   a. Construct a simple Monte Carlo algorithm to evaluate the fair price of this option to a relative accuracy of 1 cent on the dollar. What is the fair price? How many samples are needed?

   b. Identify a good control variate for this option and construct a Monte Carlo method using this control variate. Now how many samples are needed to achieve the same accuracy?

Answer: This problem is similar to the Asian arithmetic mean put option that we studied in class. We perform the simulation as for an Asian arithmetic mean call option and use the

geometric mean call for the control variate. Don't forget to discount the payoff. The price of the option is about \$5.60. It takes about 70,000 runs to obtain the desired accuracy using simple Monte Carlo but only about 10000 runs with the control variate.

```
%MATH 565 Monte Carlo Methods in Finance
%Final exam Question 5
close all, clear all, format long e, format compact,
set(0,'defaultaxesfontsize',18,'defaulttextfontsize',18)

s0=300; %initial price
strike=300; %strike price
maxcut=320; %cap
mincut=280; %floor
r=0.03; %interest rate
sig=0.3; %volatility
T=10/52; %time until expiry
d=10; %number of trading periods
delt=T/d; %difference in time

%Arithmetic Mean option by simple Monte Carlo
n=70000; %number of stock paths
x=randn(n,d); %normal random numbers
smat=cumprod([s0*ones(n,1) exp((r-sig.*sig/2)*delt + x*sig*sqrt(delt))],2);
Sbar=mean(min(max(smat(:,2:d+1),mincut),maxcut),2); %average price with caps and floors
payoff=max(Sbar-strike,0)*exp(-r*T);
Price=mean(payoff); %fair price of option estimated by MC
esterr=1.96*std(payoff)/sqrt(n);
estrelerr=esterr/Price;
disp(['Using ' int2str(n) ' sample paths the fair price of the call option is $' ...
    num2str(Price) ' + or - $' num2str(esterr)])
disp(['     for an error of ' num2str(100*estrelerr) ' cents on the dollar.'])
disp(' ');

%Using Geometric mean call Asian option as Control Variate
Tbar=(T+delt)/2;
factor=(2*d+1)/(3*d);
sigbar=sig*sqrt(factor);
rbar=r-(sig*sig/2)*(1-factor);
priceratio=strike*exp(-rbar*Tbar)/s0;
xbig=log(priceratio)/(sigbar*sqrt(Tbar))+sigbar*sqrt(Tbar)/2;
xsmall=log(priceratio)/(sigbar*sqrt(Tbar))-sigbar*sqrt(Tbar)/2;
normcdf=inline('(1+erf(x/sqrt(2)))/2','x');
AsianCallPrice=s0*(normcdf(-xsmall) - priceratio*normcdf(-xbig)); %exact fair price of call

n=10000; %number of stock paths
x=randn(n,d); %normal random numbers
smat=cumprod([s0*ones(n,1) exp((r-sig.*sig/2)*delt + x*sig*sqrt(delt))],2);
Sbar=mean(min(max(smat(:,2:d+1),mincut),maxcut),2); %average price with caps and floors
payoff=max(Sbar-strike,0)*exp(-r*T);
Price=mean(payoff); %fair price of option estimated by MC
geoSbar=exp(mean(log(smat(:,2:d+1)),2)); %geometric mean asset price
geopayoff=max(geoSbar-strike,0)*exp(-r*T); %payoff for geometric mean option
geoprice=mean(geopayoff);
beta=(geopayoff - geoprice)\(payoff - Price);
rho=corrcoef(geopayoff,payoff); rho=rho(1,2);
PriceCV=Price-beta*(geoprice-AsianCallPrice);
estCVerr=1.96*std(payoff-beta*(geopayoff-AsianCallPrice))/sqrt(n);
estrelCVerr=estCVerr/PriceCV;
disp(['Using the geometric mean call option as a control variate, beta = ' ...
    num2str(beta) ', rho = ' num2str(rho)])
disp(['Using ' int2str(n) ...
    ' sample paths the fair price of the Arithmetic Mean Put option is $' ...
    num2str(PriceCV) ' + or - $' num2str(estCVerr)])
disp(['     for an error of ' num2str(100*estrelCVerr) ' cents on the dollar.'])
disp(' ');
```

```
figure;
h=plot(geopayoff,payoff,'bs'); set(h,'linewidth',6)
print -depsc callgeopay.eps
```

6. (20 marks)
Consider a simplified power generation problem. The power demand for the jth day in the coming week is a Gaussian random variable X_j, and $X_1, \ldots, X_7$ are i.i.d. with mean 100 MW and standard deviation 20 MW. The variable cost of power generation is \$240 per MW per day. There is also a fixed cost of \$1000 per generator per day. Each generator can generate at most 25 MW. So, generating 100 MW of power for one day costs the power company $100 \times \$240 + 4 \times \$1000 = \$28,000$. Generators may be turned on instantaneously, but once turned on, they must stay on at least 2 days. The income to the power company is \$350 per MW per day.

   a. Construct a simple Monte Carlo algorithm to compute the expected weekly profit for the power company to a relative accuracy of of 1 cent on 10 dollars. What is the expected weekly profit? How many samples are needed?

   b. Use a quasi-Monte Carlo (low discepancy sequence) method, e.g., a Faure sequence or a lattice rule using generator based on the discrepancy from your last homework. Now how many samples are needed to achieve the same accuracy?

Answer: For a sample of power demands, $X_1, \ldots, X_7$, let G_j denote the number of generators needed per day. Note that $\max(G_j - G_{j-1}, 0)$ *is the number of generators turned on on day j. Thus, G_j may be defined recursively as*

$$G_1 = \lceil X_1/25 \rceil, \qquad G_2 = \max\left(\lceil X_2/25 \rceil, G_1\right),$$
$$G_j = \max\left(\lceil X_j/25 \rceil, G_{j-1} - G_{j-2}\right), \quad j = 3, \ldots, 7,$$

where $\lceil\rceil$ is the ceiling function, since the number of generators turned on for only one day is $\max(G_{j-1} - G_{j-2}, 0)$. *The profit to the power company is*

$$P := \sum_{j=1}^{7} [350X_j - (240X_j + 1000G_j)] = \sum_{j=1}^{7} [110X_j - 1000G_j].$$

The expected profit is approximately \$4500 and it requires 30,000 samples for simple Monte Carlo to get a relative error under 0.1%. *Using a lattice rule chosen by the same discrepancy as in the last homework problem,* 50 *lattice points, with generator* $(1, 13, 13^2, \ldots, 13^6)$, *and* 50 *random shifts for a total of 2500 samples one obtains the same accuracy.*

```
%MATH 565 Monte Carlo Methods in Finance
%Final exam Question 6
clear all, format long e, format compact
invnorm=inline('sqrt(2)*erfinv(2*x-1)','x'); %inverse normal cdf

d=7; %number of days
meandem=100; %mean demand
stddem=20; %standard deviation of demand
powpg=25; %maximum power per generator
costpmw=240; %variable cost per MW
```

```
costpg=1000; %fixed cost per generator used
incpmw=350; %income per MW

%Simple Monte Carlo
n=30000; %number of scenarios
x=meandem+stddem*randn(n,d); %demand
ngenmin=ceil(x/powpg); %minimum number of generators required
ngen=ngenmin(:,1); %number of generators day 1
ngen(:,2)=max(ngenmin(:,2),ngen(:,1)); %number of generators day 2
for j=3:d
    ngen(:,j)=max(ngenmin(:,j),ngen(:,j-1)-ngen(:,j-2)); %number of generators required
end
profit=sum(x*(incpmw-costpmw)-ngen*costpg,2); %profit to the power company
meanprofit=mean(profit); %mean profit
esterr=1.96*std(profit)/sqrt(n); %estimated error
estrelerr=esterr/meanprofit; %estimated relative error
disp(['Using ' num2str(n) ' samples the average profit is $' ...
    num2str(meanprofit) ' + or - $' num2str(esterr)])
disp(['     for an error of ' num2str(100*estrelerr) ' cents on the dollar.'])
disp(' ');

%Using a lattice rule
n=2500;
m=50; %number of reps
k=n/m; %number of lattice points

%Find good lattice generator
h2=[1:k-1];
ngen=length(h2);
yj=repmat([0:k-1]',1,ngen);
kvec=ones(k,ngen);
for j=1:d;
    xj=yj/k;
    kvec=kvec.*(1+(1/6 + xj.*(-1 + xj)));
    yj=mod(yj.*repmat(h2,k,1),k);
end
disc2=-1+mean(kvec,1);
[best,wh]=min(disc2);
h2best=h2(wh);
disp(['Best lattice generator for ' num2str(k) ' lattice points is (1,' ...
    num2str(h2best) ',' num2str(h2best) '^2, ...)'])

hvec=1; for j=2:d; hvec(j)=mod(hvec(j-1)*h2best,k); end %construct the lattice generator
u0=mod([0:k-1]'*hvec/k,1); %unshifted points
for i=1:m %m replications
u=mod(u0+repmat(rand(1,d),k,1),1); %random shift
x=meandem+stddem*invnorm(u); %demand
ngenmin=ceil(x/powpg); %minimum number of generators required
ngen=ngenmin(:,1); %number of generators day 1
ngen(:,2)=max(ngenmin(:,2),ngen(:,1)); %number of generators day 2
for j=3:d
    ngen(:,j)=max(ngenmin(:,j),ngen(:,j-1)-ngen(:,j-2)); %number of generators required
end
profitk(i)=mean(sum(x*(incpmw-costpmw)-ngen*costpg,2)); %profit for replication i
end
meanprofitlat=mean(profitk); %sample average profit
esterr=1.96*std(profitk)/sqrt(m);  %estimated error
estrelerr=esterr/meanprofit; %estimated relative error
disp(['Using ' int2str(m) ' replications of ' int2str(k) ' lattice points for '...
    int2str(n) ' samples ']);
disp(['     the average profit is $' num2str(meanprofitlat) ' + or - $' num2str(esterr)])
disp(['     for an error of ' num2str(100*estrelerr) ' cents on the dollar.'])
disp(' ');
```