Monte Carlo

Monte Carlo

Goal

Compute or estimate $\mathbb{E}g(X)$

Monte Carlo

[Step 1] Generate n iid samples X_i from original PDF $f(x)$.

[Step 2] Approximate $\mathbb{E}g(X)$ by

$$\mathbb{E}g(X) = \int g(x)f(x)dx \quad \approx \quad \frac{1}{n}\sum_{i=1}^{n} g(X_i)$$

Monte Carlo simulation to estimate π

Draw n random points X_i from $[-1,1]^2$ and record R_i whether the point is inside of the unit circle.

$$R_i = \begin{cases} 1 & \text{if } X_i \text{ is inside of the unit circle} \\ 0 & \text{otherwise} \end{cases}$$

Then,

$$R_i \text{ is iid } B(p),\ p = \frac{\pi}{4}.$$

Therefore, by the weak or strong law of large numbers we have for large n

$$\frac{\sum_{i=1}^n R_i}{n} \approx \frac{\pi}{4} \qquad \Rightarrow \qquad \pi \approx \frac{4\sum_{i=1}^n R_i}{n}$$

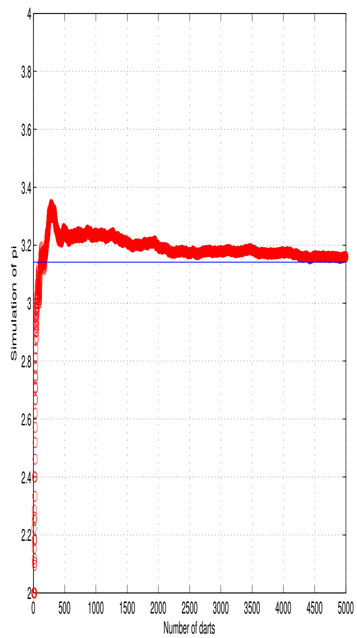

Figure 1: Monte Carlo simulation to estimate π. On the top we draw 5000 random darts to $[-1,1]^2$. We color red on the darts inside the unit circle and blue on the darts out side the unit circle. On the bottom left for each i we estimate π using the first i random draw. As we note, the estimate are getting better as we have more samples. On the bottom right we draw 100 random darts and we estimate π. We do this 100 times and make a histogram.

```
clear all; close all; clc; rng('default');

n=5000; % Number of darts for each estimate
x=2*rand(2,n)-1; % Uniform random samples from [-1,1]^2
r2=sum(x.^2); % Squre distance from the origin
Ncircle=sum(r2<=1); % Number of random samples inside unit circle
estimated_pi=4*Ncircle/n % Estimate pi

indicator=zeros(1,n);
indicator(r2<=1)=1;

subplot(2,2,1:2)
plot(x(1,indicator==1),x(2,indicator==1),'or'); hold on
plot(x(1,indicator==0),x(2,indicator==0),'o');

subplot(2,2,3)
plot(1:n,pi*ones(1,n),'-',1:n,4*cumsum(indicator)./(1:n),'or'); grid on;
axis([0 n 2 4])
xlabel('Number of darts'); ylabel('Simulation of \pi')

subplot(2,2,4)
n=100; % Number of darts for each estimate
m=100; % Number of estimates computed using n dart
x=2*rand(2,n,m)-1; % Uniform random samples from [-1,1]^2
r2=sum(x.^2); % Squre distance from the origin
Ncircle=sum(r2<=1); % Number of random samples inside unit circle
estimated_pi=4*Ncircle/n; % Estimate pi
estimated_pi=estimated_pi(:);
hist(estimated_pi); xlabel('Estimate of \pi');
```

Buffon's needle

On a paper we draw parallel lines 1 units apart. We drop a needle of length 1 onto the paper n times and record R_i whether the needle intersect the line.

$$R_i = \begin{cases} 1 & \text{if the needle intersect the line at the } i\text{-th drop} \\ 0 & \text{otherwise} \end{cases}$$

Then,

$$R_i \text{ is iid } B(p),\ p = \frac{2}{\pi}.$$

Therefore, by the weak or strong law of large numbers we have for large n

$$\frac{\sum_{i=1}^n R_i}{n} \approx \frac{2}{\pi} \qquad \Rightarrow \qquad \pi \approx \frac{2n}{\sum_{i=1}^n R_i}$$

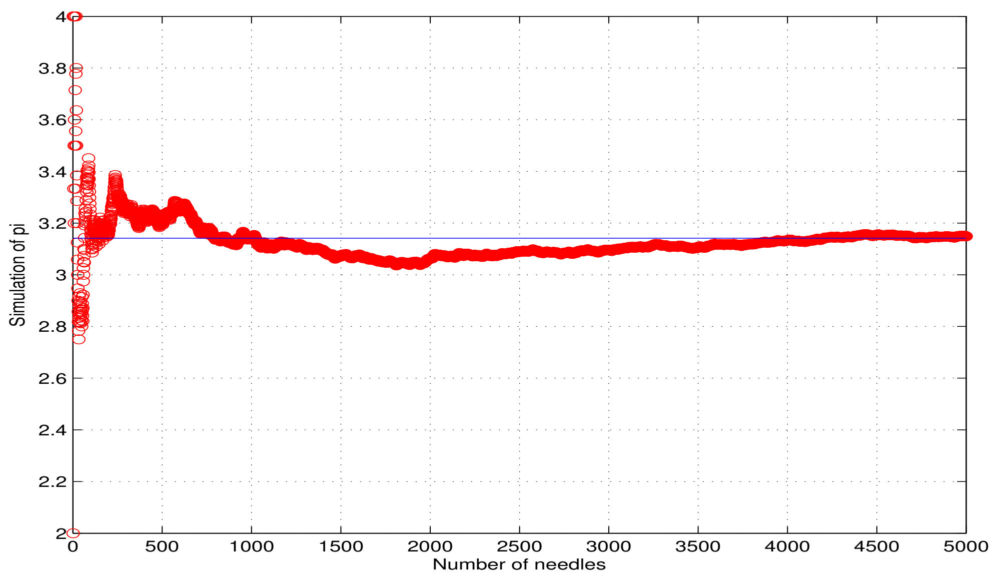

Figure 2: Buffon's needle, a simulation to estimate π On the topt we draw 5000 random bars such that the lower end of the bar are between 0 and 1. We color red on the bars that cross $y = 1$ and blue on the bars that don't cross $y = 1$. On the bottom left for each i we estimate π using the first i random draw. As we note, the estimate are getting better as we have more samples. On the bottom right we draw 100 random bars and we estimate π. We do this 100 times and make a histogram.

```
clear all; close all; clc; rng('default');

n=5000; % Number of random samples generated
x=rand(2,n); % First row = Height of lower end; Second row = Angle/pi;
h=x(1,:)+sin(pi*x(2,:)); % Height of higher end
Nbar=sum(h>=1); % Number of random samples hit the upper bar at y=1
estimated_pi=2*n/Nbar % Estimate pi

indicator=zeros(1,n);
indicator(h>=1)=1;

subplot(2,2,1:2)
for i=1:n

    temp=randn(1,1);
    plot_x=[temp temp+cos(pi*x(2,i))];
    plot_y=[x(1,i) x(1,i)+sin(pi*x(2,i))];

    if (indicator(i)==1),
        plot(plot_x,plot_y,'-r'); hold on
    else
        plot(plot_x,plot_y,'-b'); hold on;
    end

end

subplot(2,2,3)
plot(1:n,pi*ones(1,n),'-',1:n,2*(1:n)./cumsum(indicator),'or'); grid on;
axis([0 n 2 4])
xlabel('Number of bars'); ylabel('Simulation of pi')

subplot(2,2,4)
n=100; % Number of darts for each estimate
m=100; % Number of estimates computed using n dart
x=rand(2,n,m); % First row = Height of lower end; Second row = Angle/pi;
h=x(1,:,:)+sin(pi*x(2,:,:)); % Height of higher end
Nbar=sum(h>=1); % Number of random samples hit the upper bar at y=1
estimated_pi=2*n/Nbar; % Estimate pi
estimated_pi=estimated_pi(:);
hist(estimated_pi)
```

Monte Carlo - Option valuation

[Step 1] Simulate many stock price paths using risk-neutral measure $\mathbb{Q}$.

$$S_T = S * e^{(r-0.5*v^2)*T+v*\sqrt{T}*\texttt{randn(1)}}$$

[Step 2] Compute option values for each stock price path and take the average.

```
clear all; close all; clc; rng('default');

S = 258.07; K = 250.5; T = 0.1205; r = 0.027; v = 0.126;

% Option pricing by Monte Carlo
N = 10000; % Number of simulation
S_T = S*exp((r-0.5*v^2)*T+v*sqrt(T)*randn(N,1));
Call_MC = exp(-r*T)*mean(max(S_T-K,0));
Put_MC = exp(-r*T)*mean(max(K-S_T,0));

% Option pricing by Black-Scholes formula
Call_BS = Call(S,K,T,r,v);
Put_BS = Put(S,K,T,r,v);
% [Call_BS,Put_BS] = blsprice(S,K,r,T,v,0);

fprintf('                              Black-Scholes     Monte Carlo\n')
fprintf('Call option price    %g            %g\n',Call_BS,Call_MC)
fprintf('Put option price     %g            %g\n',Put_BS,Put_MC)

%% Output

                     Black-Scholes     Monte Carlo
Call option price    9.82709           9.8045
Put option price     1.44341           1.40617
```

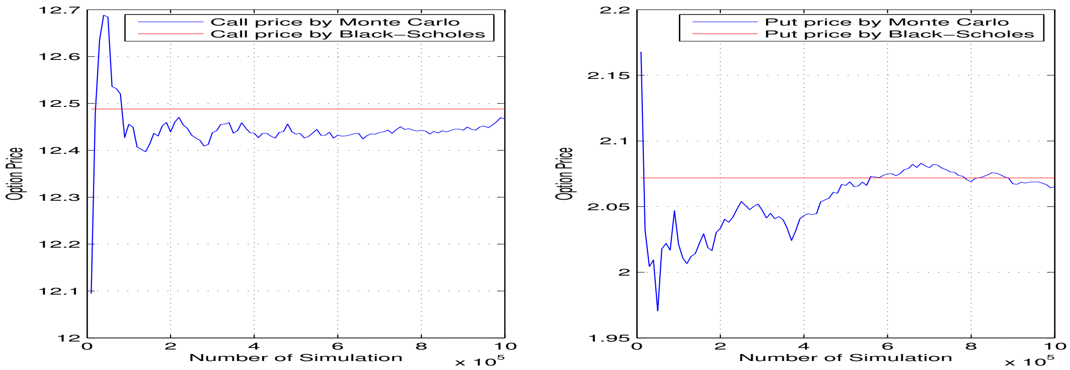

Figure 3: Monte Carlo - Speed of convergence

```
clear all; close all; clc; rng('default');

S=258.07; K=250.5; T=0.1205; r=0.027; v=0.126;

% Option pricing by Monte Carlo
N_Simu=1000000; % Number of simulation
S_T=S*exp((r-0.5*v^2)*T+v*sqrt(T)*randn(N_Simu,1));
N=length(10000:10000:N_Simu);
C_MC=zeros(1,N);
P_MC=zeros(1,N);

for i=1:N
    C_MC(i)=exp(-r*T)*mean(max(S_T(1:1000*i)-K,0));
    P_MC(i)=exp(-r*T)*mean(max(K-S_T(1:1000*i),0));
end

% Option pricing by Black-Scholes formula
C=Call(S,K,T,r,v);
P=Put(S,K,T,r,v);

subplot(1,2,1)
plot(10000:10000:N_Simu,C_MC); grid on; hold on;
plot(10000:10000:N_Simu,C*ones(1,N),'-r');
legend('Call price by Monte Carlo','Call price by Black-Scholes')
xlabel('Number of Simulation'); ylabel('Option Price')

subplot(1,2,2)
plot(10000:10000:N_Simu,P_MC); grid on; hold on;
plot(10000:10000:N_Simu,P*ones(1,N),'-r');
legend('Put price by Monte Carlo','Put price by Black-Scholes')
xlabel('Number of Simulation'); ylabel('Option Price')
```

Importance sampling

Goal

Compute or estimate $\mathbb{E}g(X)$

Importance sampling

[**Step 1**] Generate n iid samples X_i not from $f(x)$, but from a new PDF $q(x)$.

[**Step 2**] Approximate $\mathbb{E}g(X)$ by

$$\mathbb{E}g(X) = \int g(x)f(x)dx = \int \frac{g(x)f(x)}{q(x)}q(x)dx \quad \approx \quad \frac{1}{n}\sum_{i=1}^{n} \omega(X_i)g(X_i)$$

where

$$\omega(X_i) = \frac{f(X_i)}{q(X_i)} \qquad \text{(Importance weight)}$$

Importance sampling without normalization constants

Goal

$$\text{Compute or estimate } \mathbb{E}g(X)$$

Without normalization constants

$$\begin{aligned}
&(1) \qquad \text{Know } \tilde{f}(x) \text{ not } f(x), \text{ where } f(x) = \frac{\tilde{f}(x)}{Z_f} \\
&(2) \qquad \text{Know } \tilde{q}(x) \text{ not } q(x), \text{ where } q(x) = \frac{\tilde{q}(x)}{Z_q} \\
&(3) \qquad \text{Can generate } n \text{ iid samples } X_i \text{ from } q(x)
\end{aligned}$$

Importance sampling without normalization constants

[**Step 1**] Generate n iid samples X_i not from $f(x)$, but from a new PDF $q(x)$.

[**Step 2**] Approximate $\mathbb{E}g(X)$ by

$$\begin{aligned}
\mathbb{E}g(X) &= \int g(x)f(x)dx = \int \frac{g(x)f(x)}{q(x)}q(x)dx = \frac{Z_q}{Z_f}\int \frac{g(x)\tilde{f}(x)}{\tilde{q}(x)}q(x)dx \\
&\approx \frac{1}{Z_f/Z_q}\left[\frac{1}{n}\sum_{i=1}^{n}\omega_0(X_i)g(X_i)\right]
\end{aligned}$$

where $\omega_0(X_i) = \frac{\tilde{f}(X_i)}{\tilde{q}(X_i)}$. Here, Z_f/Z_q is not known and it is approximated further by

$$\begin{aligned}
\frac{Z_f}{Z_q} &= \frac{1}{Z_q}\int \tilde{f}(x)dx = \int \tilde{f}(x)\frac{1}{Z_q}dx = \int \tilde{f}(x)\frac{q(x)}{\tilde{q}(x)}dx = \int \frac{\tilde{f}(x)}{\tilde{q}(x)}q(x)dx \\
&\approx \frac{1}{n}\sum_{i=1}^{n}\omega_0(X_i)
\end{aligned}$$

Combining these two approximations, approximate $\mathbb{E}g(X)$ by

$$\mathbb{E}g(X) \approx \frac{1}{n}\sum_{i=1}^{n}\tilde{\omega}(X_i)g(X_i)$$

where

$$\tilde{\omega}(X_i) = \frac{\omega_0(X_i)}{\frac{1}{N}\sum_{j=1}^{N}\omega_0(X_j)}$$

Pros and cons of importance sampling

Pros

- Easier to sample from q
- Can reduce the variance of the estimator
- Works well in low dimension up to 6

Cons

- Does not work well in high dimension $\Rightarrow$ MCMC
- Need modification in sequential setting $\Rightarrow$ sequential importance sampling

How to choose q

$$\text{argmin}_q \int \frac{g^2 f^2}{q^2} q - \left(\int \frac{gf}{q} q \right)^2 = \text{argmin}_q \int \frac{g^2 f^2}{q^2} q - \left(\int gf \right)^2 = \text{argmin}_q \int \frac{g^2 f^2}{q^2} q$$

Minimize the variance of the estimator - Lagrangian

$$\text{argmin}_{q_i > 0,\ \sum_i q_i = 1} \quad \sum_i \frac{g_i^2 f_i^2}{q_i^2} q_i$$

$$\mathcal{L} = \sum_i \frac{g_i^2 f_i^2}{q_i^2} q_i - \beta \left(\sum_i q_i - 1 \right) = \sum_i \frac{g_i^2 f_i^2}{q_i} - \beta \left(\sum_i q_i - 1 \right)$$

$$\frac{\partial}{\partial q_j} \mathcal{L} = -\frac{g_i^2 f_i^2}{q_i^2} - \beta = 0 \quad \Rightarrow \quad q_i^2 \propto g_i^2 f_i^2 \quad \Rightarrow \quad q_i \propto |g_i| f_i \quad \Rightarrow \quad q_i = \frac{|g_i| f_i}{Z}$$

Minimize the variance of the estimator - Jensen's inequality

$$RHS = \int \frac{g^2 f^2}{q^2} q \geq \left(\int \frac{|g| f}{q} q \right)^2 = \left(\int |g| f \right)^2 = LHS$$

With

$$q \propto |g| f \quad \Rightarrow \quad q = \frac{|g| f}{Z}$$

$$RHS = \int \frac{g^2 f^2}{q^2} q = \int \frac{g^2 f^2}{g^2 f^2 / Z^2} \frac{|g| f}{Z} = Z \int |g| f = Z^2 = \left(\int |g| f \right)^2 = LHS$$

How to choose q

1st choice	$q \propto \lvert g \rvert f \quad \Rightarrow \quad q = \frac{\lvert g \rvert f}{Z}$
2nd choice	Choose q large when $\lvert g \rvert f$ is large
Other choice	Minimize the variance of the posterior
Other choice	Minimize the variance of the MCMC
Other choice	Use MLE or MAP
Other choice	Study the nature of the problem
Other choice	Cross validation