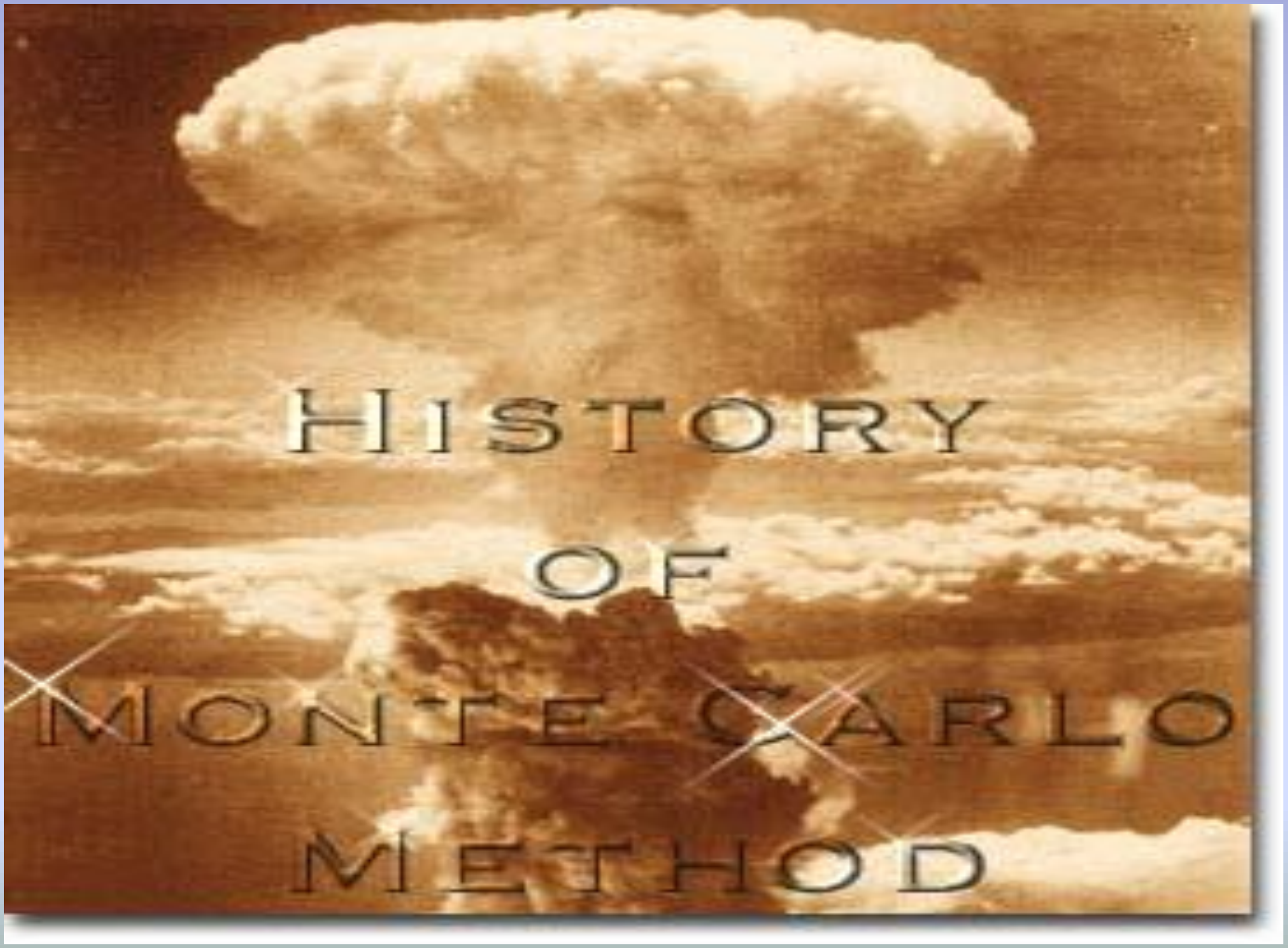
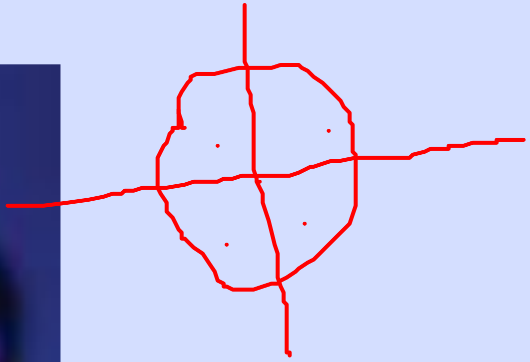
Monte Carlo Simulation

History



What is Monte Carlo (MC) method ?

The Monte Carlo method :is a numerical method for statistical simulation which utilizes sequences of random numbers to perform the simulation



What the meaning of MC simulation?

- MC simulation is a versatile tool to analyse and evaluate complex measurements
- Constructing a model of a system.
- Experimenting with the model to draw inferences of the system's behavior

A simulation model

Inputs

Decision and uncontrollable variables

→ ***Simulation model*** →

outputs

Measures of performance or behaviour of the system

A simulation model cont..

- Model inputs capture the environment of the problem
- The simulation model
  - Conceptual model: set of assumptions that define the system
  - Computer code: the implementation of the conceptual model
- Outputs describe the aspects of system behaviour that we are interested in

Random numbers

- Uniform Random numbers or pseudo-random numbers (PRN) are essentially independent random variables uniformly Distributed over the unit interval (0,1).

- The PRNs are good if they are uniformly distributed, statistically independent and reproducible.

Linear congruential generator

- *Generating a random sequence of numbers* $\{X_1, X_2, \ldots\ldots, X_k\}$ *of length M over the interval [0,M-1]*

$$X_i = \mathrm{mod}(AX_{i-1} + C, M)$$

$$R = X_i / M$$

♠♠♠ $\mathrm{mod}(b, M) = b - \mathrm{int}(b/M) * M$

- *Starting value* X_0 *is called "seed"*
- *M,A and C are nonnegative integers known*

Modulus, multiplier and increment, respectively

- *M is must be prime number*$(2^{31}-1, 2^7-1, \ldots\ldots)$

Fortran program

```
 program random_num
 implicit none
 integer,parameter::m=2**21,a=17,c=43
 integer::i,n,s
 real(8)::R,Xo
 read*,n
 !n is the number of random points
 Xo=27.0d0
!------------------------
 do i=1,n
 s=A*Xo+c
 Xo=mod(s,m)
 R=Xo/m
  write(*,*) R
 end do
!------------------------
 end program random_num
```

Classic Example

Find the value of π *?*

Use the reject and accept method

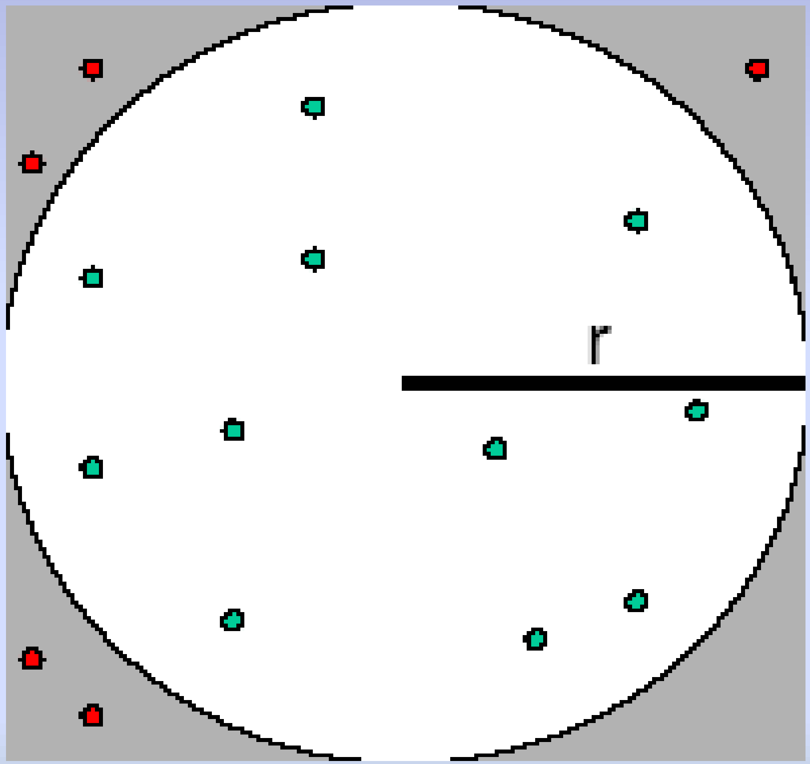
Or hit and miss method

The area of square=$(2r)^2$

The area of circle = πr^2

$$\frac{area \cdot of \cdot square}{area \cdot of \cdot circle} = \frac{4r^2}{\pi r^2} = \frac{4}{\pi}$$

$$\pi = 4 * \frac{area \cdot of \cdot circle}{area \cdot of \cdot square}$$

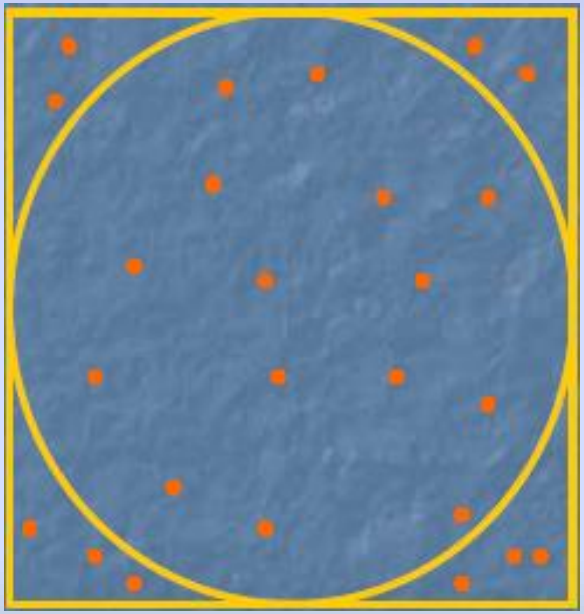


Cont....

$$\frac{area.of\,.circle}{area.of\,.square} = \frac{\#.of\,.dots.inside.circle}{total.number.of\,.dots}$$

Hit and miss algorithm

- Generate two sequences of N of PRN :: R_i, R_j
- $X_i = -1 + 2R_i$
- $Y_j = -1 + 2R_j$
- Start from s=zero
- If $(X^2 + Y^2 < 1)$ $s = s + 1$
- \# of dots inside circle=s
- total number of dots=N



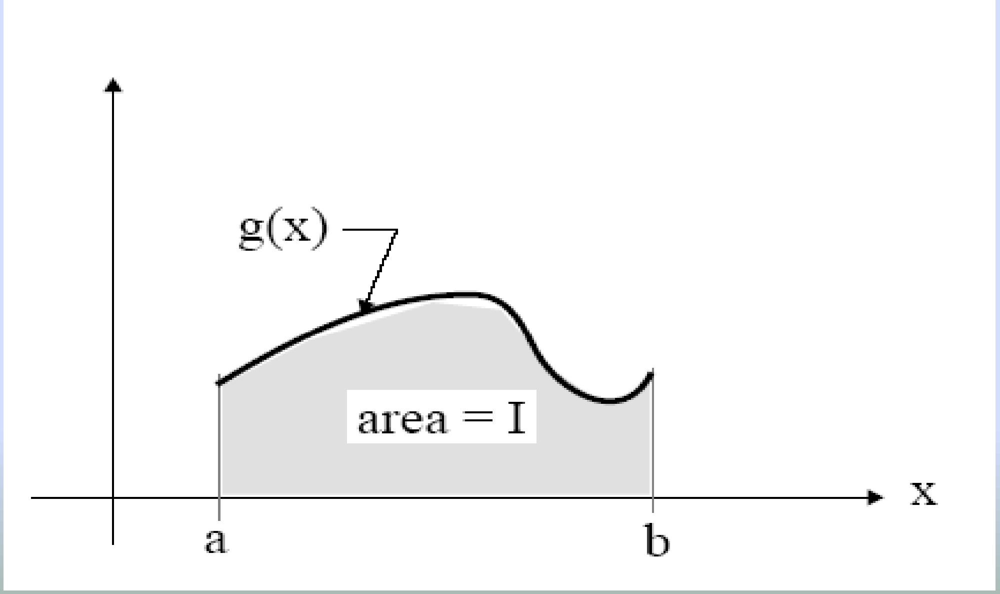
$$\pi = 4 * S / N$$

Fortran program

```
 PROGRAM Pi
 implicit none
 integer,parameter::mm=(2**17)-1,aa=(2**7)-1
REAL(8)::X, Y, Pi
Real(8),dimension(:),allocatable::R
 INTEGER::I,N
 Pi = 0.0d0
READ*, N
allocate(R(n))
DO I = 1,N
CALL RANDOM_NUMBER(R(n))
CALL RANDOM_NUMBER(R(n+1))
 X =-1.0d0+2.0d0*R(n)
 Y=-1.0d0+2.0d0*R(N+1)
IF (X*X+Y*Y<1.0d0) Pi=Pi+1.0d0
END DO
Pi=4*Pi/N
PRINT*,Pi
END program pi
```

Monte Carlo Integration

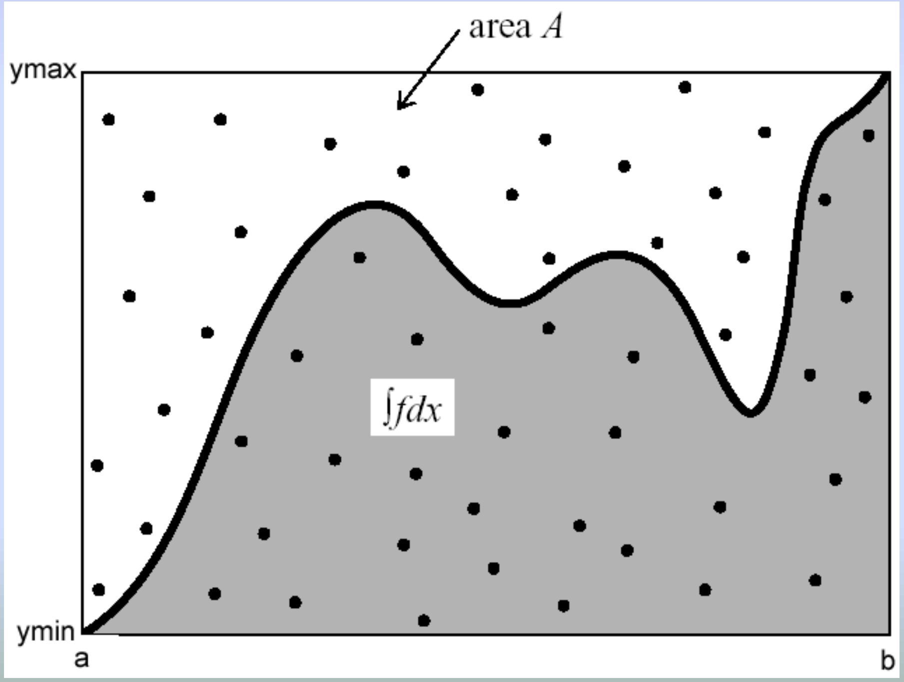
- ♥ *Hit and miss method*
- ♥ *Sample mean method*
- ♥ *importance sampled method*



Hit and Miss method

$$I = \int_a^b f(x)dx \qquad a,b \in R$$

- Generate two sequence of N of PRN (R_i, R_j) $i\&j=1,2,\ldots,N$
- $0 \le f(x) \le Y_{max}$, for $X \in (a,b)$
- $X_i = a + R_i(b-a)$
- $Y_i = Y_{max} R_j$
- start from s=0
- if $Y_j < f(x)$ $s=s+1$
- $I = Y_{max}(b-a)\,S/N$

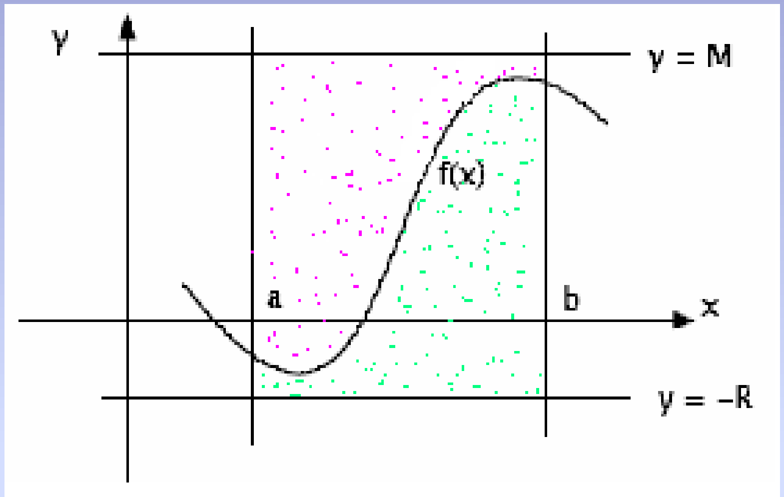


```
program Hit_miss
implicit none
real(8),dimension(:),allocatable::R
real(8)::X,Y,m,a,b,integ
integer::i,N,s
read*,a !the lower value of the interval
read*,b !the upper value of the interval
M=f(b)
if (dabs(f(a))>dabs(f(b))) M=f(a)
read*,N       !the number of random numbers
allocate(R(n))
s=0
do i=1,N
call random_number(R(n))
call random_number(R(n+1))
X=a+R(n)*(b-a)
Y=M*R(n+1)
if (y<f(x)) s=s+1
end do
INTEG=M*(b-a)*s/N
print*,integ
contains
real(8) function F(X)
real(8),intent(in)::X
F=2.0*X+1.0
end function F
end program Hit_miss
```

Cont….

If there is a part of the function under X-axis

$$\frac{s}{N} = \frac{area.under.curve}{Total.area} = \frac{\int_a^b f(x)dx}{(M+R)(b-a)}$$

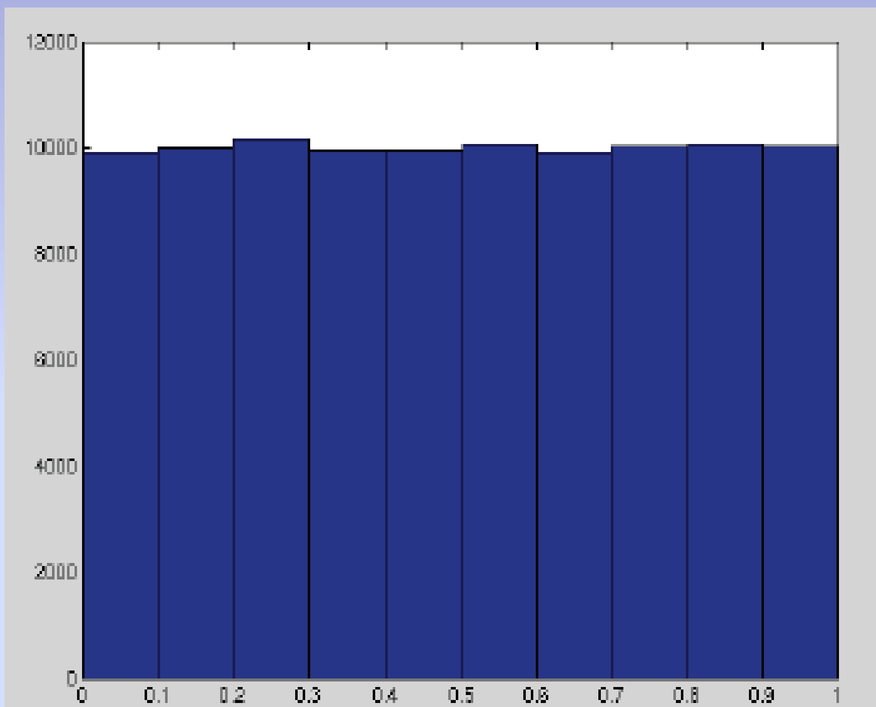


$$\int_a^b f(x)dx = (M+R)(b-a)\frac{s}{N}$$

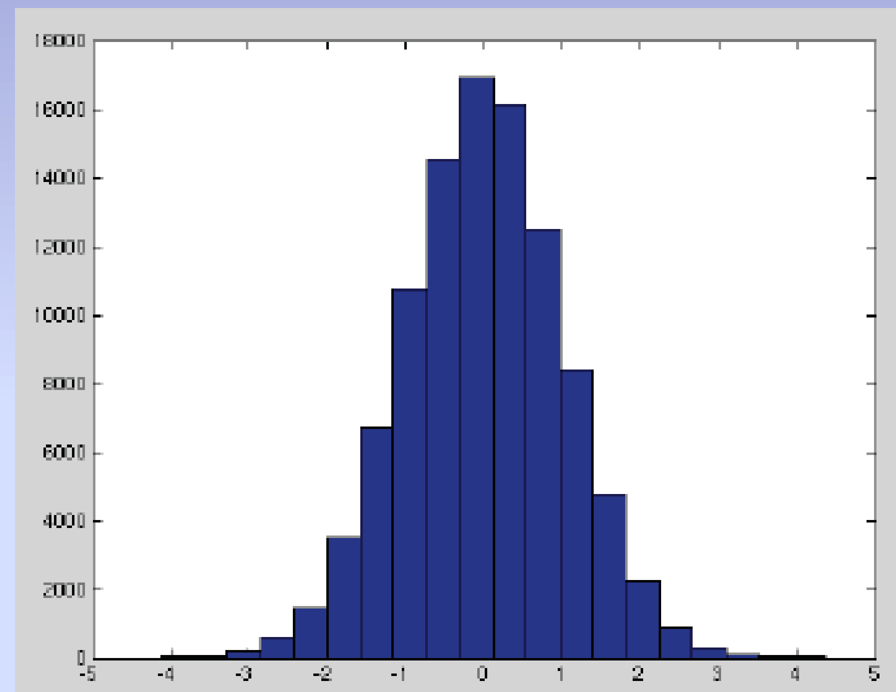
This must now be corrected for the fact that the line y = -R has been used as the baseline for the integral instead of the line y = 0. This is accomplished by subtracting the rectangular area R(b-a).

then $$\int_a^b f(x)dx = (M+R)(b-a)\frac{s}{N} - R(b-a)$$

Types of distribution



Uniform distribution



Gaussian or normal distribution

Sample Mean method

Rewrite $I = \int_a^b f(x)dx$ *By* $I = \int_a^a h(x)\phi(x)dx$

Where ϕ *is p.d.f*

$\phi(x) \geq 0 \qquad \int_a^b \phi(x)dx = 1$

$h(x) = f(x) / \phi(x)$

Theorem….

If $x_1, x_2, x_3, \ldots\ldots, x_N$ *are i,i,d uniformly distributed on [a,b],then*

$I = \int_a^b f(x)dx \approx (b-a)\langle f \rangle \quad , \quad \langle f \rangle = \frac{1}{N}\sum_{i=1}^{N} f(x_i)$

Cont…

From the theorem choose $\phi(x) = \frac{1}{b-a}$ *and* $h(x) = (b-a)\,f(x)$

Then an estimate of I is

$$\hat{I} = \frac{(b-a)}{N}\sum_{i=1}^{N} f(x_i)$$

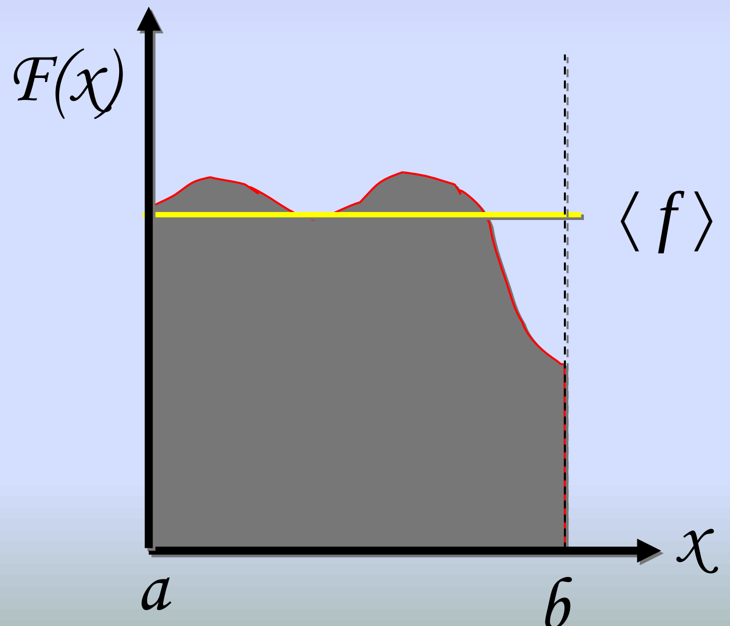
You can calculate the value of error from the variance

$$error = \sqrt{\mathrm{var}(\hat{I})}$$

$$\mathrm{var}(\hat{I}) = \frac{(b-a)^2}{N}\,\mathrm{var}(f)$$

$$\mathrm{var}(f) = \langle f^2 \rangle - \langle f \rangle^2$$

$$\approx \frac{1}{N}\sum_{i=1}^{N} f^2(x_i) - [\frac{1}{N}\sum_{i=1}^{N} f(x_i)]^2$$



Sample Mean MC algorithm

- ♠ Generate sequence of N of PRN : R_i
- ♠ Compute $X_i = a + R_i(b-a)$
- ♠ compute $f(X_i)$, $i=1,2,3,\ldots,N$
- ♠ use $$\hat{I} = \frac{(b-a)}{N}\sum_{i=1}^{N} f(x_i)$$

♠♠♠ note:: if $f(x)$ is not square integrable ,then the MC Estimate $\hat{I}$ will still converge to the true value, but The error estimate becomes unreliable.

```
program Sampled_Mean
implicit none
real(8),dimension(:),allocatable::R
real(8)::a,b,sum,integ,X
integer::i,N
read*,a   ! Lower value
read*,b   ! upper value
read*,n   ! number of random points
allocate(R(n))
sum=0.0d0
do i=1,n
call random_number(R(n))
call random_number(R(n+1))
X=a+R(n)*(b-a)
sum=sum+f(x)
int=((b-a)/N)*sum
end do
write(*,*) "integ=",integ
contains
real(8) function F(X)
real(8),intent(in)::X
F=2*X+1.0d0
end function F
end program Sampled_Mean
```

Generalization to multidimensional cases

Rewrite d-dimension integral

$$I = \int_{\Gamma} f(x)dx \quad \text{by} \quad I = \int_{\Gamma} h(x)\phi(x)dx$$

Because the uniform distribution ,choose

$$\phi(x) = \frac{1}{\prod_{i=1}^{d}(b_i - a_i)} = \frac{1}{V}$$

$$h(x) = [\prod_{i=1}^{d}(b_i - a_i)] f(x_i) = Vf(x_i)$$

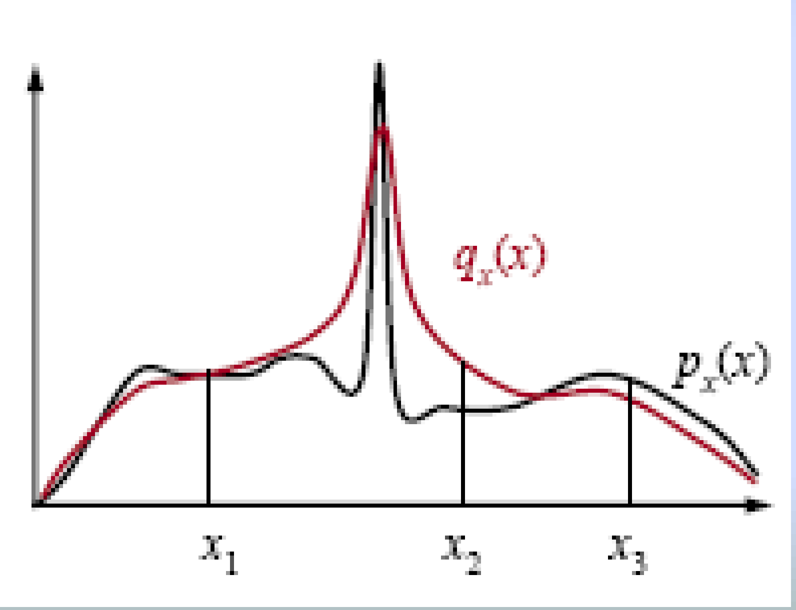
$$\phi(x) \geq 0 \quad \text{and} \quad \int_{\Gamma} \phi(x)dx = 1$$

The estimate of I is

$$\hat{I} = \frac{1}{N}\sum_{i=1}^{N} h(x_i) = \frac{V}{N}\sum_{i=1}^{N} f(X_i)$$

Importance Sampled method

- It is quite obvious that most of the integral comes from the region of the peak. But if we generate points evenly in the interval [a, b], most points won't be in the peak area ,and their contribution to the total will be relatively small….
- The idea behind importance sampling is to transform f(x) into another, flatter function which is then MC integrated of course there has to be a back-transformation to give the original integral we really want.



Important properties of continuous and discrete pdf's

Property	Continuous: $f(x)$	Discrete: $\{p_i\}$
Positivity	$f(x) \geq 0$, all x	$p_i > 0$, all i
Normalization	$\int_{-\infty}^{\infty} f(x')\,dx' = 1$	$\sum_{j=1}^{N} p_j = 1$
Interpretation	$f(x)\,dx$ $\mathrm{prob}(x \leq x' \leq x + dx)$	$p_i = \mathrm{prob}(i) =$ $\mathrm{prob}(x_j = x_i)$
Mean	$\bar{x} = \int_{-\infty}^{\infty} x f(x)\,dx$	$\bar{x} = \sum_{j=1}^{N} x_j p_j$
Variance	$\sigma^2 = \int_{-\infty}^{\infty} (x-\bar{x})^2 f(x)\,dx$	$\sigma^2 = \sum_{j=1}^{N} (x_j - \bar{x})^2 p_j$

Steps of Importance Sampled method

rewrite $I = \int_a^b f(x)dx$ *by* $I = \int_a^b h(x)g(x)dx$

where $h(x) = \dfrac{f(x)}{g(x)}$

$I = \int_a^b h(x)dG(x)$ *where* $G(x) = \int_0^x g(x)dx$

Now make a variable change

$u = G(x)$ *and* $du = dG(x)$

$x = G^{-1}(u)$ *Then*

$$I = \int_{G(a)}^{G(b)} \frac{f(G^{-1}(u))}{g(G^{-1}(u))} du$$

Cont….

now the value of the integration equal to the average value

$$I = \frac{1}{N}\sum_{i=1}^{N}\frac{f(G^{-1}(u_i))}{g(G^{-1}(u_i))}$$

The error calculated by variance method

$$error = \sqrt{\mathrm{var}(I)} \qquad \mathrm{var}(I) = \frac{1}{N}\mathrm{var}(\frac{f}{g})$$

$$\mathrm{var}(\frac{f}{g}) = \langle \frac{f^2}{g^2} \rangle - \langle \frac{f}{g} \rangle^2 \qquad \mathrm{var}(I) = \frac{1}{N}[\langle \frac{f^2}{g^2} \rangle - \langle \frac{f}{g} \rangle^2]$$

♠♠♠ *note that, the function* $G(x)$ *must be invertible*

Example (Importance-Sampled method)

find $$I = \int_0^1 e^{-x^2}\,dx$$

In this region, the function decreases from 1 to 1/e. The simple exponential function e^{-x} *does the same, so let's use that for g(x). We first have to normalize g, so we calculate*

$$\int_0^1 e^{-x}\,dx = \frac{-1}{e} + 1 = \frac{e-1}{e}$$

Then our normalized weighting function is

$$g(x) = \frac{e^{-x}e}{e-1}$$

Cont....

$$G(x)=\int_0^x g(x)dx=\int_0^x \frac{e^{1-x}}{e-1}dx$$

$$G(x)=\frac{(1-e^{-x})e}{e-1}$$

let $u=G(x)$

$$x=G^{-1}(u)=-\log_e(1-u\frac{e-1}{e})$$

Then

$$G^{-1}(u_i)=-\log_e(1-u_i\frac{e-1}{e})$$

Cont…

♣♣♣ *note:: from the last result we redistribute the PRN according to the pdf (g(x)), then the new values (i.e.,* $G^{-1}(u_i)$ *) are uniform random numbers used To find the value of*

$$I = \frac{1}{N}\sum_{i=1}^{N}\frac{f(G^{-1}(u_i))}{g(G^{-1}(u_i))}$$

on the interval of integration ,where the average value is the estimator value for the original integral.

```
program important_sample
implicit none
real(8)::sum,u,R,integ
integer::i,N
read*,N
sum=0.0
do i=1,N
call random_number(R)
u=G_inv(R)
sum=sum+f(u)/g(u)
end do
integ=sum/N
write(*,*) integ
contains
real(8) function F(x)
real(8),intent(in)::x
F=dexp(-x**2)
end function F
real(8) function g(x)
real(8),intent(in)::x
g=dexp(-x)
end function g
real(8) function G_inv(x)
real(8),intent(in)::x
G_inv=-dlog(1-x*1.718d0/2.718d0)
end function G_inv
end program important_sample
```

Why monte carlo ?

Method	*Conv.rate in one dim*	*Conv.rate in d-dim*
Basic MC	$N^{-1/2}$	$N^{-1/2}$
Trapezoidal rule	N^{-2}	$N^{-2/d}$
Simpson's rule	N^{-4}	$N^{-4/d}$

MC is from the best methods to find the partition function numerically , then you can solve the stochastic processes.

Suppose we want to solve the following integral Using any other numerical method ,(i.e. Trapezoidal or Simpson)

$$I = \int_{\Gamma} f(x_i)\,dx_i \qquad \textit{Where} \qquad i=1,2,3,\ldots\ldots,30$$

So we generate a grid in the domain of integration ,suppose a grid with 10 nodes on each coordinate axis in 30 dimension cube $[0,1]^{30}$ be chosen .in this case, we have 10^{30} abscissas.
Suppose a time of 10^{-7} second is necessary for calculating One value of the function . Therefore , a time of 10^{15} years will be necessary for evaluating the integral.