Monte Carlo and PRNG

- Monte Carlo methods
- Random number generators
- Generating non-uniform random numbers

Geant4 uses mainly the following MC methods

- Composition + (Inverse Transform) Method
- Acceptance-Rejection Method

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

Named after the famous Casino City

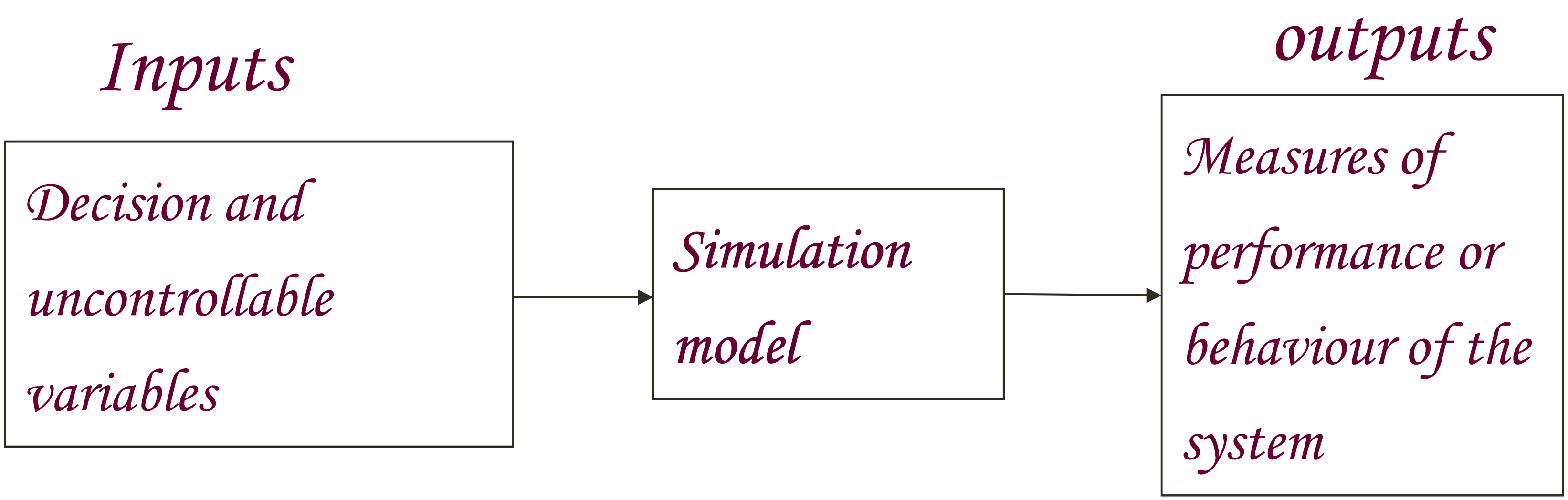
What the meaning of MC simulation?

- 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



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

Components of Monte Carlo simulation

- *Probability distribution functions (pdf's)* - the physical (or mathematical) system must be described by a set of pdf's.
- *Random number generator* - a source of random numbers uniformly distributed on the unit interval must be available.
- *Sampling rule* - a prescription for sampling from the specified pdf's, assuming the availability of random numbers on the unit interval, must be given.
- *Scoring (or tallying)* - the outcomes must be accumulated into overall tallies or scores for the quantities of interest.

Components of Monte Carlo simulation (cont.)

- *Error estimation* - an estimate of the statistical error (variance) as a function of the number of trials and other quantities must be determined.
- *Variance reduction techniques* - methods for reducing the variance in the estimated solution to reduce the computational time for Monte Carlo simulation
- *Parallelization and vectorization* - algorithms to allow Monte Carlo methods to be implemented efficiently on advanced computer architectures.

What MC Needs

- MC methods might needs different RNG.
  - For example, when simulating outgoing direction for a launched particle and interactions of the particle with the medium, the following would be the desirable properties:
- The attribute of each particle should be independent from each other.
- The attribute of all the particles should span across the entire attribute space. I.e., as we approach infinite numbers of particles, the particles launched into a space should cover the space completely.

Next slide will states the properties of the RNG needed.

What MC Needs (cont.)

- Any subsequence of random numbers should not be correlated with any other subsequence of random numbers. For example, when simulating the launched particles, we should not generate geometrical patterns.
- Random number repetition should occur only after a very large generation of random numbers.
- The random numbers generated should be uniform. This point and the first one are loosely related. To achieve more uniformity, some correlations between random numbers must be established.
- The RNG should be efficient. It should be vectorizable with low overhead. The processors in parallel systems, should not be required to talk between each other.

Probability Density Function

- A **probability density function** (or **probability distribution function**) is a function f defined on an interval (a, b) and having the following properties:

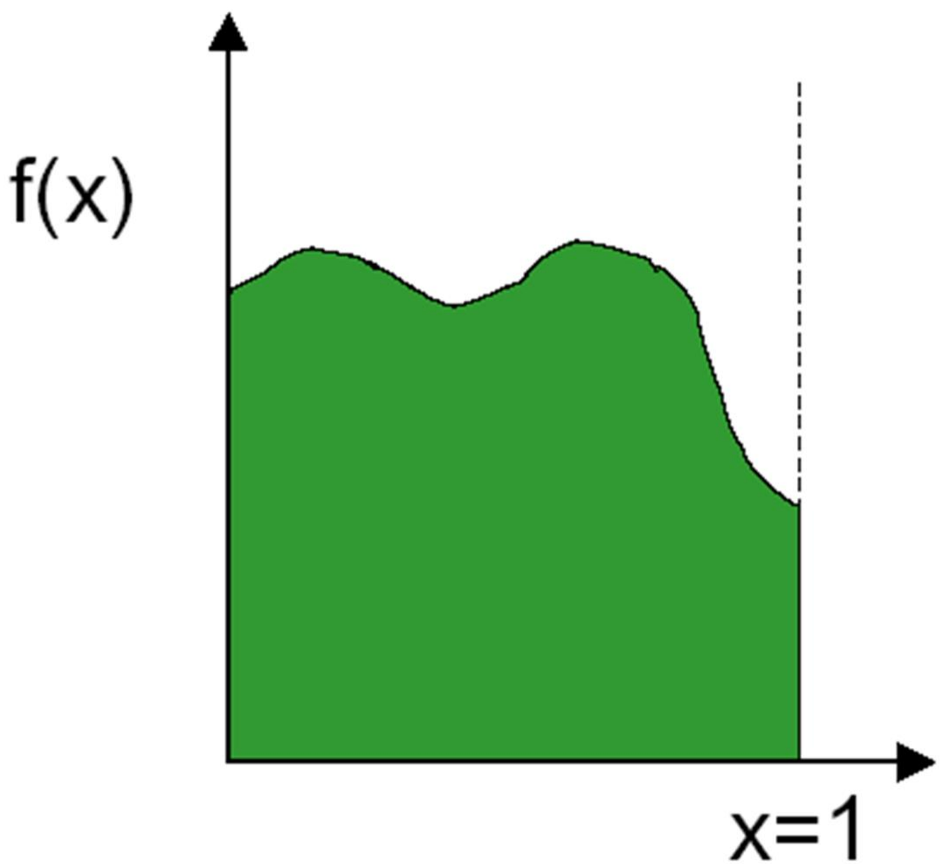
1) $f(x) \geq 0 \qquad \forall x \in [a,b]$

2) $\int_a^b f(x)\,dx = 1$

3) $P(x_1 \leq x < x_2) = \int_{x_1}^{x_2} f(x)\,dx$

Solving Integration Problems via Statistical Sampling: Monte Carlo Approximation

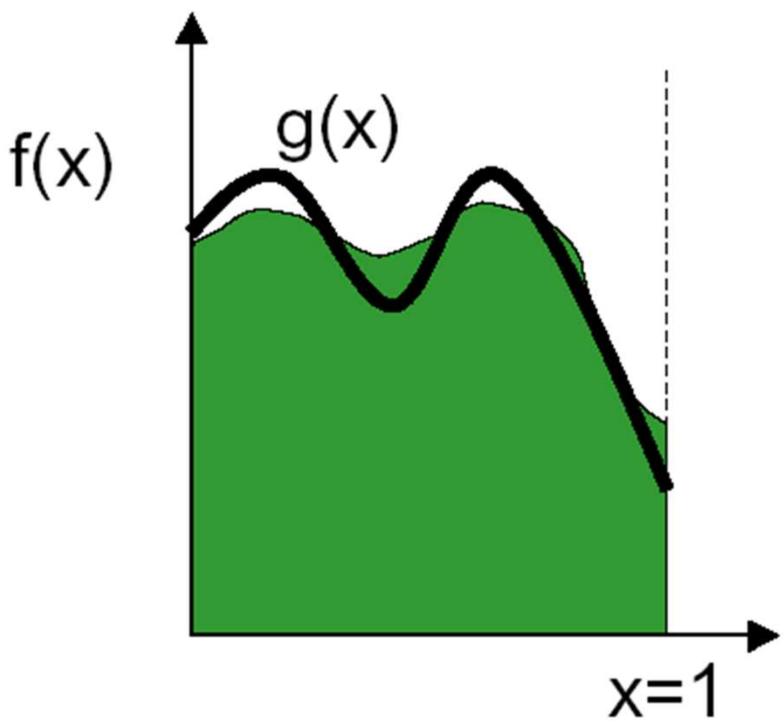
- How to evaluate integral of f(x)?



$$\int_0^1 f(x)dx = ?$$

Integration Approximation

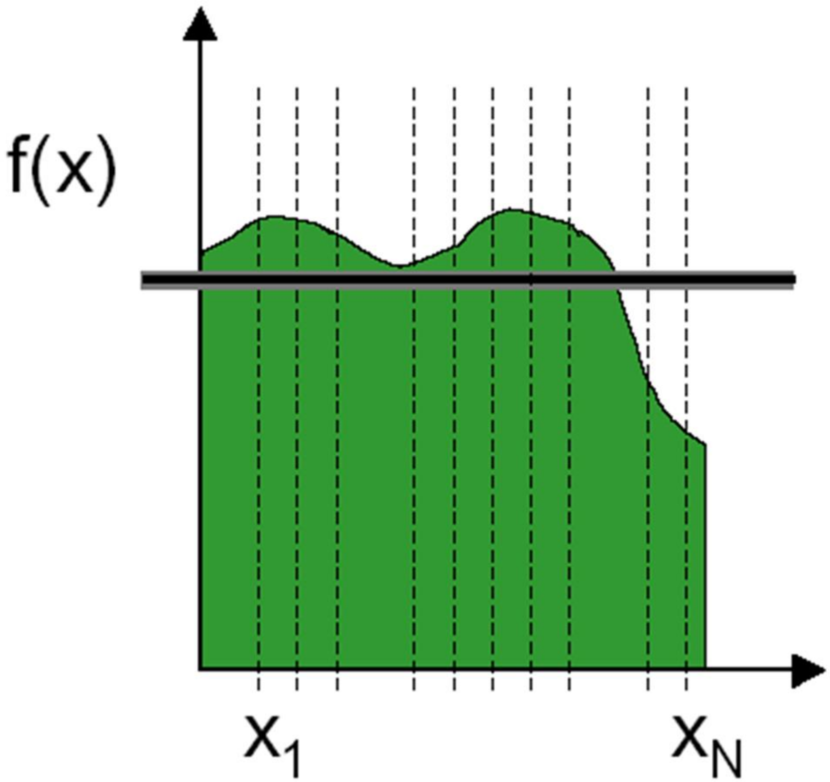
- Can approximate using another function g(x)



$$\int_0^1 f(x)dx = \int_0^1 g(x)dx$$

Integration Approximation

- Estimate the average by taking N samples



$$< f >$$

$$\int_0^1 f(x)dx = \frac{1}{N}\sum_{i=1}^{N} f(x_i)$$

$$< f > = \frac{1}{N}\sum_{i=1}^{N} f(x_i)$$

Monte Carlo Integration

$$\mathbf{I} = \int_{a}^{b} f(x)dx$$

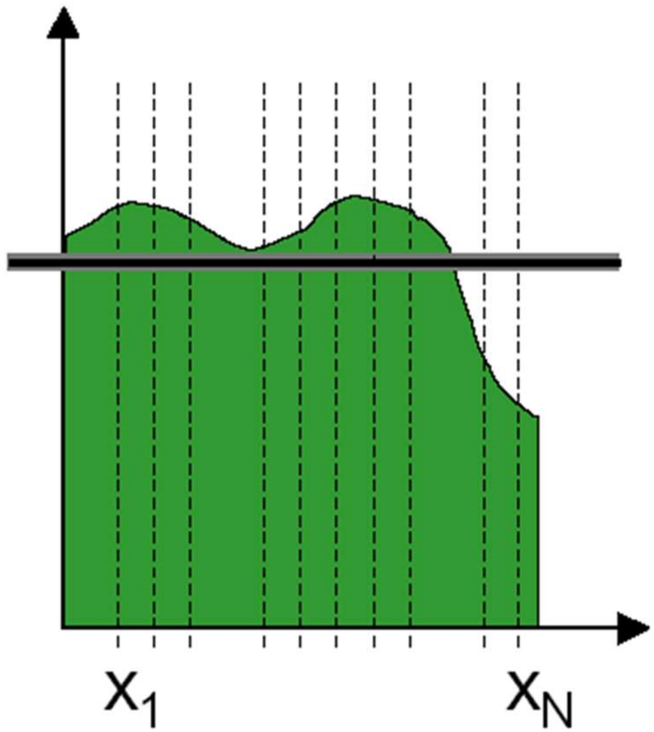
$$\mathbf{I}_m = (b-a)\frac{1}{N}\sum_{i=1}^{N} f(x_i)$$

- I_m = Monte Carlo estimate
- N = number of samples
- x_1, x_2, …, x_N are uniformly distributed random numbers between a and b

$$\lim_{N \to \infty} \mathbf{I}_m = \mathbf{I}$$

Variance

- The variance describes how much the sampled values *vary* from each other.



$$< f > = \frac{1}{N} \sum_{i=1}^{N} f(x_i)$$

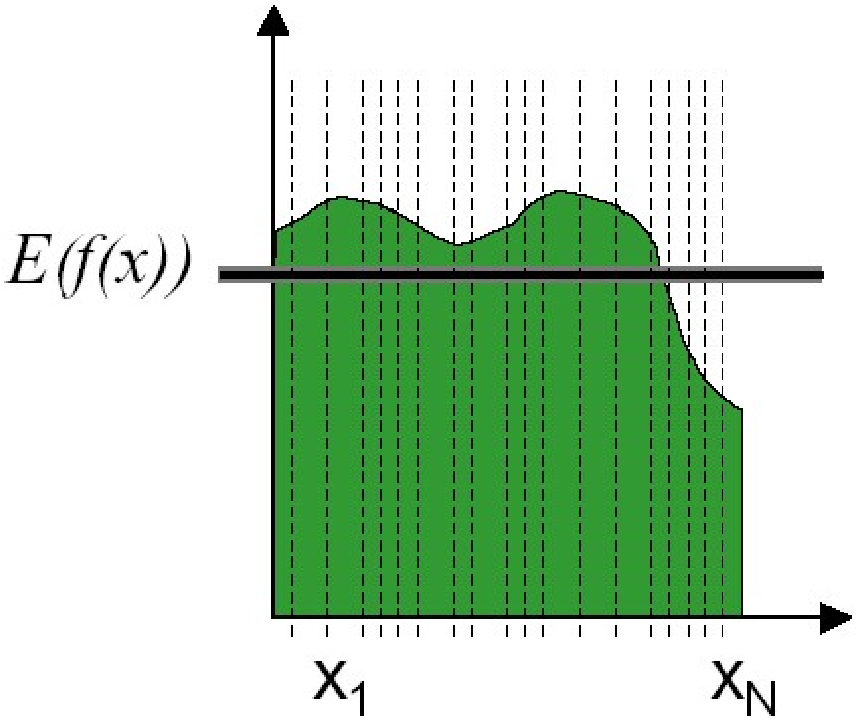
$$< f^2 > = \frac{1}{N} \sum_{i=1}^{N} f^2(x_i)$$

$$Var(I_m) = < f^2 > - < f >^2$$

- Variance proportional to 1/N

Variance

- Standard Deviation is just the square root of the variance
- Standard Deviation proportional to $1/\sqrt{N}$

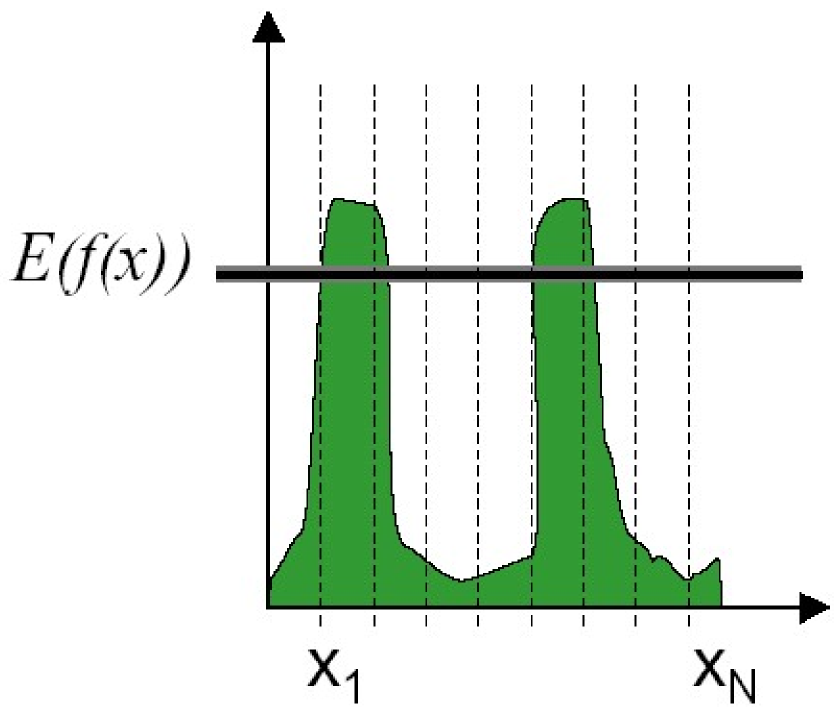


$$\mathrm{I} = I_m \pm \sqrt{Var(I_m)}$$

- **Need 4X samples to halve the error**

Variance

- Problem:
  - Variance (noise) decreases slowly
  - Using more samples only removes a small amount of noise

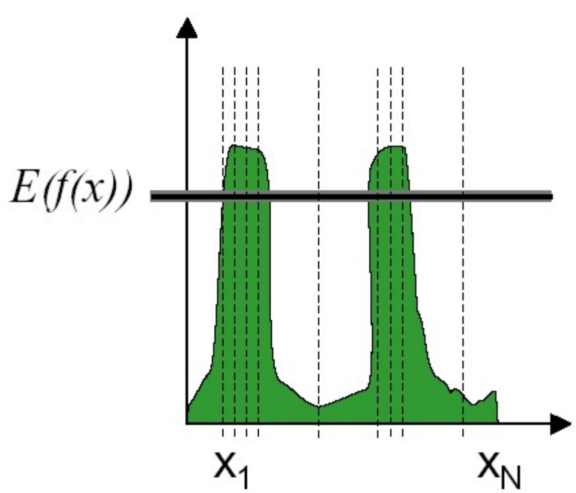


Variance Reduction

- There are several ways to reduce the variance
  - Importance Sampling
  - Stratified Sampling

Importance Sampling

- Idea: use more samples in important regions of the function
- If function is high in small areas, use more samples there



PSEUDO RANDOM NUMBERS

Random versus Pseudo-random

- Virtually all computers have “random number” generators
- Their operation is deterministic
- Sequences are predictable
- More accurately called “pseudo-random number” generators
- In this chapter “random” is shorthand for “pseudo-random”
- “RNG” means “random number generator”

Properties of an Ideal RNG

- Uniformly distributed
- Uncorrelated
- Never cycles
- Satisfies any statistical test for randomness
- Reproducible
- Machine-independent
- Changing “seed” value changes sequence
- Easily split into independent subsequences
- Fast
- Limited memory requirements

No RNG Is Ideal

- Finite precision arithmetic $\Rightarrow$ finite number of states $\Rightarrow$ cycles
  - Period = length of cycle
  - If period > number of values needed, effectively acyclic
- Reproducible $\Rightarrow$ correlations
- Often speed versus quality trade-offs

Linear Congruential RNGs

$$X_i = (a \times X_{i-1} + c) \bmod M$$

Modulus

Additive constant

Multiplier

Sequence depends on choice of seed, X_0

Period of Linear Congruential RNG

- Maximum period is M
- For 32-bit integers maximum period is 2^{32}, or about 4 billion
- This is too small for modern computers
- Use a generator with at least 48 bits of precision

Producing Floating-Point Numbers

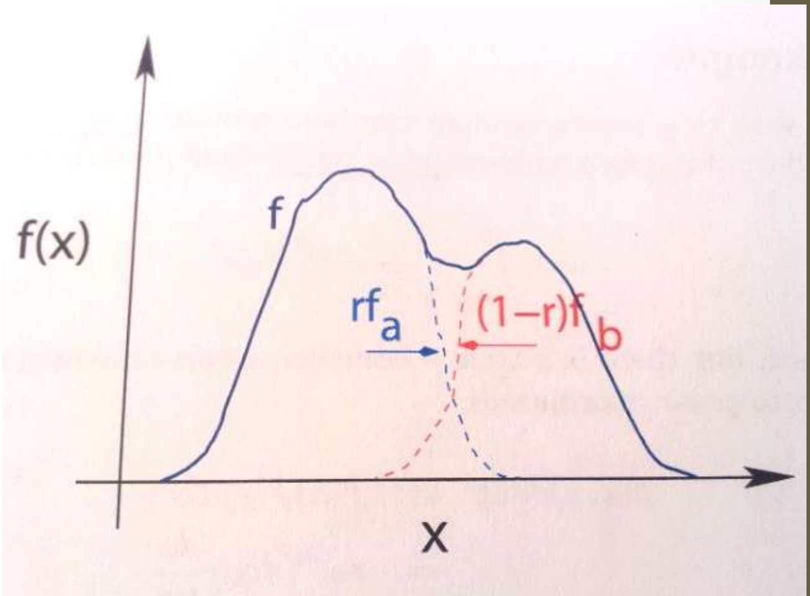
- X_i, a, c, and M are all integers
- X_is range in value from 0 to M-1
- To produce floating-point numbers in range [0, 1), divide X_i by M

Defects of Linear Congruential RNGs

- Least significant bits correlated
  - Especially when M is a power of 2
- k-tuples of random numbers form a lattice
  - Points tend to lie on hyperplanes
  - Especially pronounced when k is large

Composition Method

- The desired pdf is in the form of a sum of terms (eg 1 + $\cos^2\theta$)
- We can break it up into pieces
- $f(x) = rf_a(x) + (1-r)f_b(x)$
- f_a *and* f_b *normalized pdf*
- $0 \leq r \leq 1$



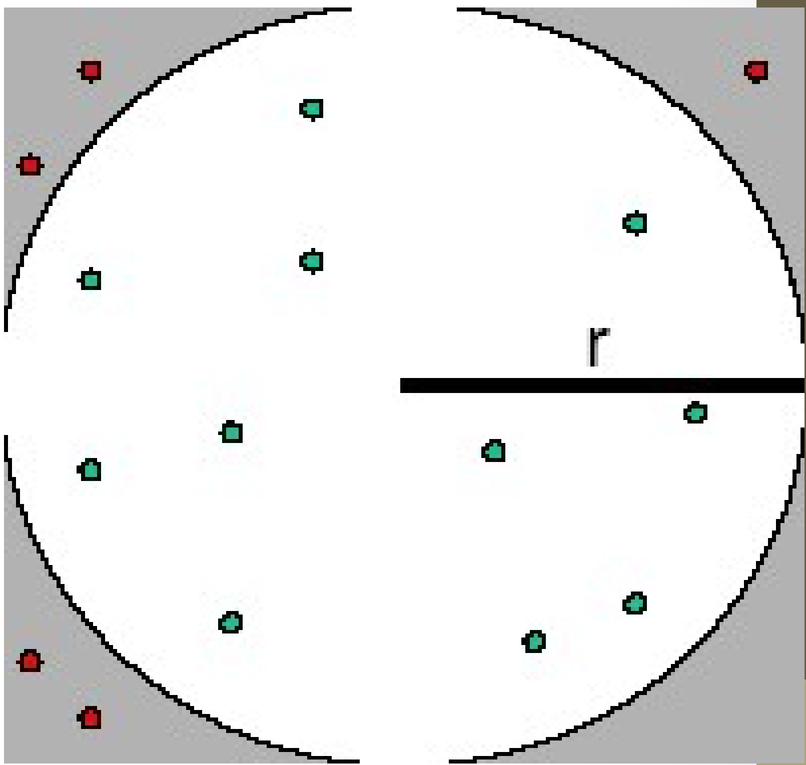
1. Generate two random numbers (uniform) u_1 and u_2
2. If $u_1 < r$ *let* $x = F_a^{-1}(u_2)$
3. If $u_1 \geq r$ *let* $x = F_b^{-1}(u_2)$

EXAMPLES

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

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

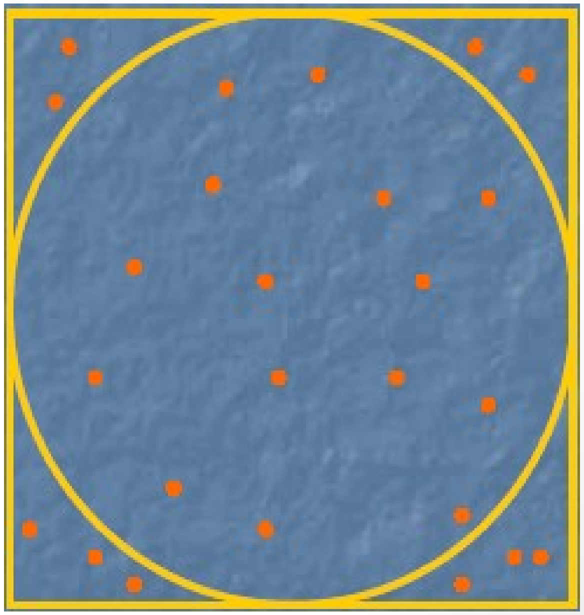
$$\pi = 4 * \frac{area \cdot of \cdot circle}{area \cdot of \cdot square}$$

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,,Rj
- ♣ $X_i=-1+2R_i$
- ♣ $Y_j=-1+2R_j$
- ♣ Start from s=zero
- ♣ If $(X^2+Y^2<1)$ s=s+1
- ♣ # of dots inside circle=s
- ♣ total number of dots=N

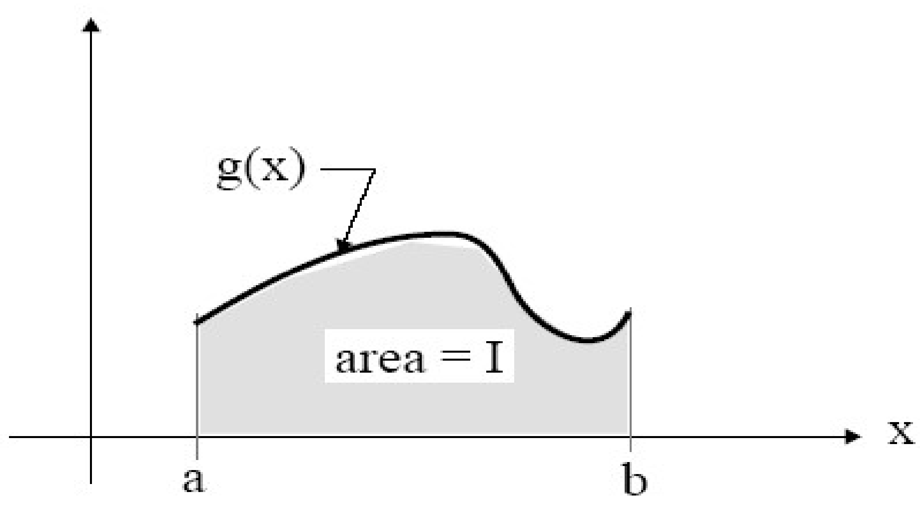


$$\pi = 4 * S / N$$

Monte Carlo Integration

- *Hit and miss method*
- *Sample mean method*

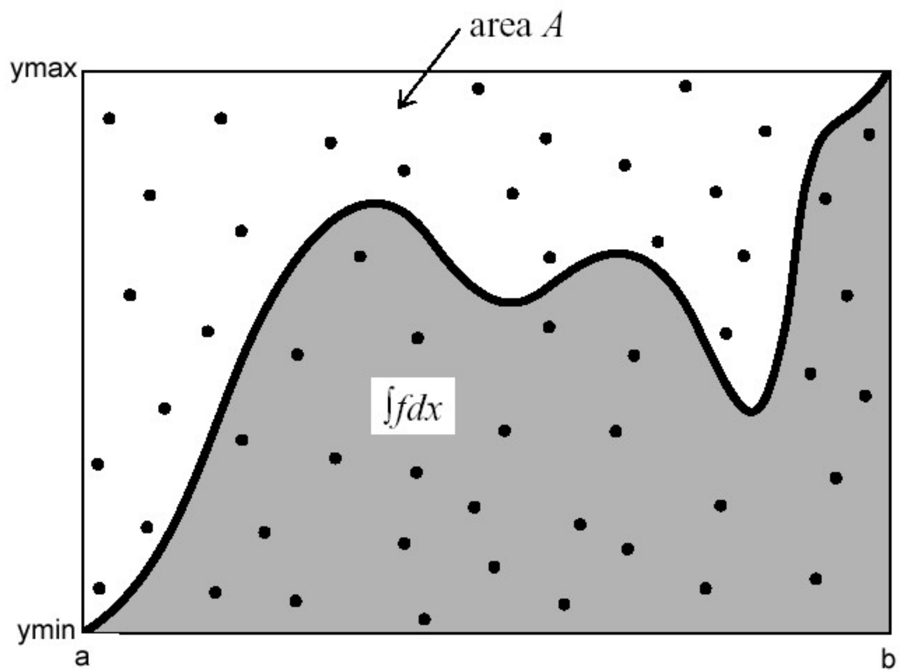
- *importance sampled method*



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,….,N
- 0 ≤f(x) ≤Y_{max} ,for X Є (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



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 $x_1, x_2, x_3, \ldots, x_N$ *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

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

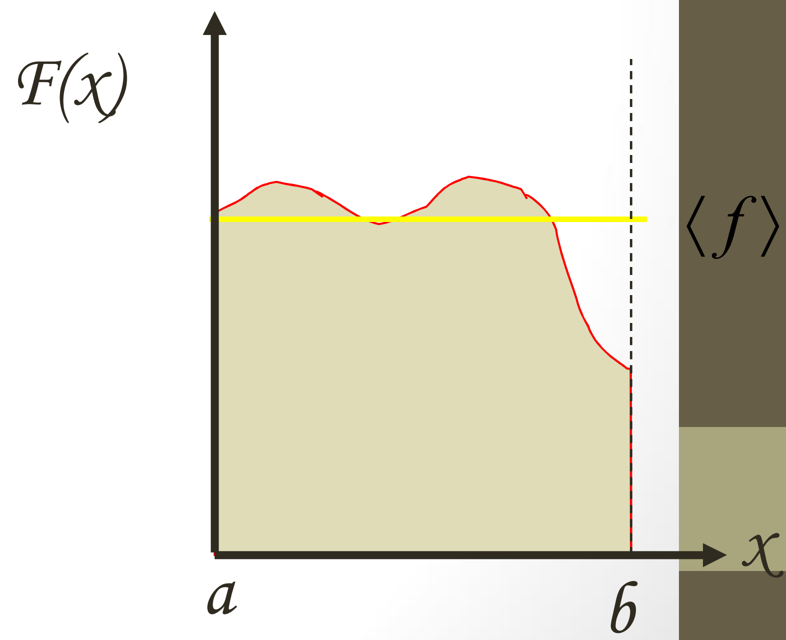
You can calculate the value of error from the variance

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

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

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

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



Sample Mean MC algorithm

♠ Generate sequence of N of PRN : R_i

♠Compute $X_i=a+R_i(b-a)$

♠ compute f(Xi) , i=1,2,3,….,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 Î will still converge to the true value, but The error estimate becomes unreliable.

An Interesting History

- In 1738, Swiss physicist and mathematician Daniel Bernoulli published *Hydrodynamica* which laid the basis for the kinetic theory of gases: great numbers of molecules moving in all directions, that their impact on a surface causes the gas pressure that we feel, and that what we experience as heat is simply the kinetic energy of their motion.

- In 1859, Scottish physicist James Clerk Maxwell formulated the distribution of molecular velocities, which gave the proportion of molecules having a certain velocity in a specific range. This was the first-ever statistical law in physics. Maxwell used a simple thought experiment: particles must move independent of any chosen coordinates, hence the only possible distribution of velocities must be normal in each coordinate.

- In 1864, Ludwig Boltzmann, a young student in Vienna, came across Maxwell's paper and was so inspired by it that he spent much of his long, distinguished, and tortured life developing the subject further.

History of Monte Carlo Method

- Credit for inventing the Monte Carlo method is shared by Stanislaw Ulam, John von Neuman and Nicholas Metropolis.
- Ulam, a Polish born mathematician, worked for John von Neumann on the Manhattan Project. Ulam is known for designing the hydrogen bomb with Edward Teller in 1951. In a thought experiment he conceived of the MC method in 1946 while pondering the probabilities of winning a card game of solitaire.
- Ulam, von Neuman, and Metropolis developed algorithms for computer implementations, as well as exploring means of transforming non-random problems into random forms that would facilitate their solution via statistical sampling. This work transformed statistical sampling from a mathematical curiosity to a formal methodology applicable to a wide variety of problems. It was Metropolis who named the new methodology after the casinos of Monte Carlo. Ulam and Metropolis published a paper called “The Monte Carlo Method” in *Journal of the American Statistical Association*, 44 (247), 335-341, in 1949.