Introduction to Monte Carlo

G. Battistoni
INFN Milano

Monte Carlo and Medical Physics

- Radiotherapy
  - External/internal sources and dosimetry
  - Phantom simulation
  - Treatment planning
  - Quality assurance
- Nuclear medicine
  - Detector simulation
  - Imaging correction
  - Absorbed dose evaluation
- Radiation protection
  - Shielding design of facilities
  - Evaluation of ambent dose

Use of MC is spreading out in medical physics since the '80s

Overview:

General concepts:

- Monte Carlo foundations
- Simulation vs. integration

Random Sampling

- Random numbers

Sampling techniques from distribution

- by inversion (discrete, continue)

The Monte Carlo method

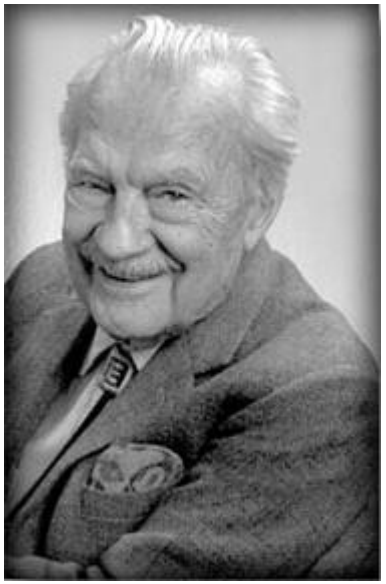
- Numerical method based on random sampling of probability distributions
- Strong mathematical foundations

Early historical ideas:

- G. Leclerc, Comte de Buffon (1777): idea of tossing a needle to calculate π
- Laplace (1886): random points in a square enclosing a circle to calculate π

The Monte Carlo method in the modern age

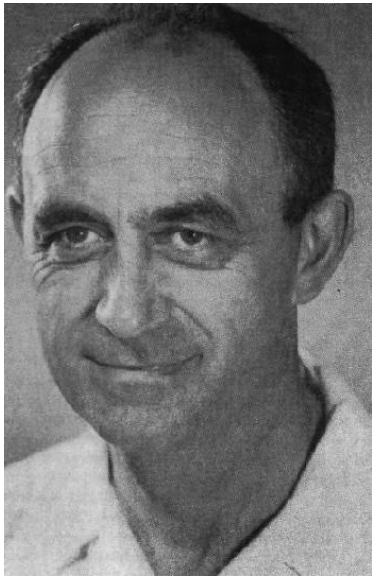
Invented by John von Neumann, Stanislaw Ulam and Nicholas Metropolis (who gave it its name), and independently by Enrico Fermi



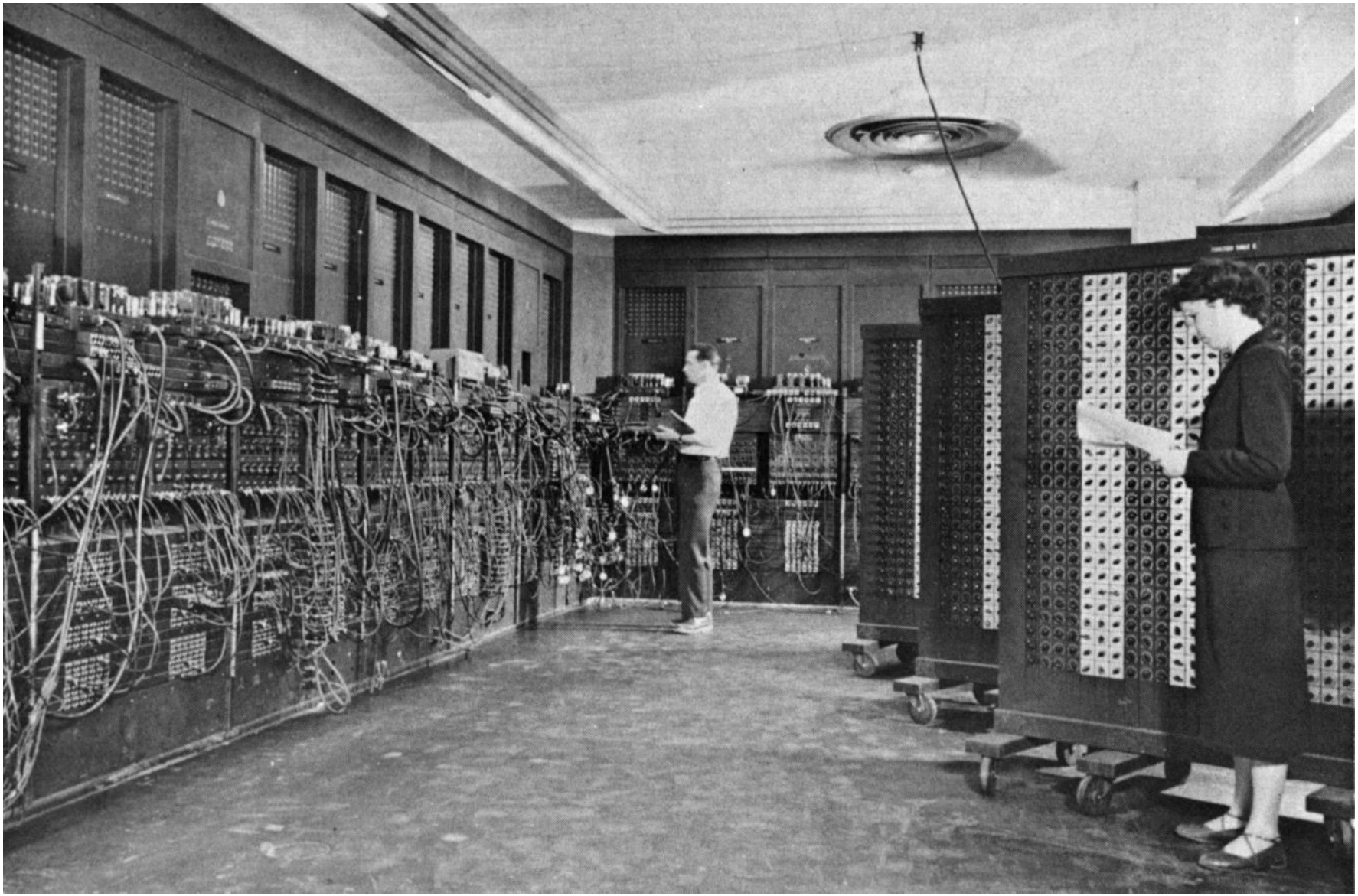
N. Metropolis

S. Ulam

J. von Neumann



E. Fermi



The ENIAC

Electronic Numerical Integrator And Computer

Integration? Or simulation?

Why, then, is MC often considered a simulation technique?

- Originally, the Monte Carlo method was not a simulation method, but a device to solve a multidimensional integro-differential equation by building a stochastic process such that some parameters of the resulting distributions would satisfy that equation
- The equation itself did not necessarily refer to a physical process, and if it did, that process was not necessarily stochastic

MC Mathematical foundation

The Central Limit Theorem is the mathematical foundation of the Monte Carlo method. In words:

> Given any observable A, that can be expressed as the result of a convolution of random processes, the average value of A can be obtained by sampling many values of A according to the probability distributions of the random processes.

MC is indeed an integration method that allows to solve multi-dimensional integrals by sampling from a suitable stochastic distribution.

The accuracy of MC estimator depends on the number of samples:

$$\sigma \propto \frac{1}{\sqrt{N}}$$

Central Limit theorem

Central limit theorem:

$$\lim_{N\to\infty} P(S_N) = \frac{1}{\sqrt{\frac{2\pi}{N}}\sigma_A} e^{-\frac{(S_N-\overline{A})^2}{2\sigma_A^2/N}}$$

For large values of N, the distribution of averages (normalized sums S_N) of N independent random variables identically distributed (according to any distribution with mean and variance ≠ ∞) tends to a normal distribution with mean $\overline{A}$ and variance σ_A^2/N

$$\lim_{N\to\infty} S_N = \lim_{N\to\infty} \frac{\sum_1^N A(x, y, z,...) f'(x)\, g'(y)\, h'(z) \ldots}{N} = \overline{A}$$

Mean of a distribution (1)

- In one dimension:

  Given a variable *x*, distributed according to a function *f(x),* the mean or average of another function of the same variable *A(x)* over an interval [a,b] is given by:

$$\overline{A} = \frac{\int_a^b A(x) f(x)\, dx}{\int_a^b f(x) dx}$$

Or, introducing the normalized distribution f' :

$$f'(x) = \frac{f(x)}{\int_a^b f(x) dx}$$

$$\overline{A} = \int_a^b A(x) f'(x) dx$$

Mean of a distribution (2)

- In several dimensions:

Given *n* variables *x,y,z,…* distributed according to the (normalized) functions *f′(x), g′(y), h′(z)…,* the mean or average of a function of those variables *A(x,y,z)* over an *n*-dimensional domain D is given by:

$$\overline{A} = \int_x \int_y \int_z \ldots \int_n A(x, y, z, \ldots)\, f'(x)\, g'(x)\, h'(x) \ldots dx\, dy\, dz \ldots$$

Often impossible to calculate with traditional methods, but we can sample N values of A with probability f′·g′·h′… and divide the sum of the sampled A_i values (i=1,2,…N) by N:

$$S_N = \frac{\sum_1^N A_i(x, y, z, \ldots)}{N}$$

Each term of the sum is distributed like A (Analog Monte Carlo)

In this case the integration is also a simulation!

Analog Monte Carlo

In an analog Monte Carlo calculation, not only the mean of the contributions converges to the mean of the actual distribution, but also the variance and all moments of higher order:

$$\lim_{N\to\infty}\left[\frac{\sum_1^N \left(x-\bar{x}\right)^n}{N}\right]^{\frac{1}{n}} = \sigma_n$$

Then, partial distributions, fluctuations and correlations are all faithfully reproduced: in this case (and in this case only!) we have a real simulation

Simulation: in special cases

- It was soon realized, however, that when the method was applied to an equation describing a physical stochastic process, such as neutron diffusion, the model (in this case a random walk) could be identified with the process itself

- In these cases the method (analog Monte Carlo) has become known as a simulation technique, since every step of the model corresponds to an identical step in the simulated physical process

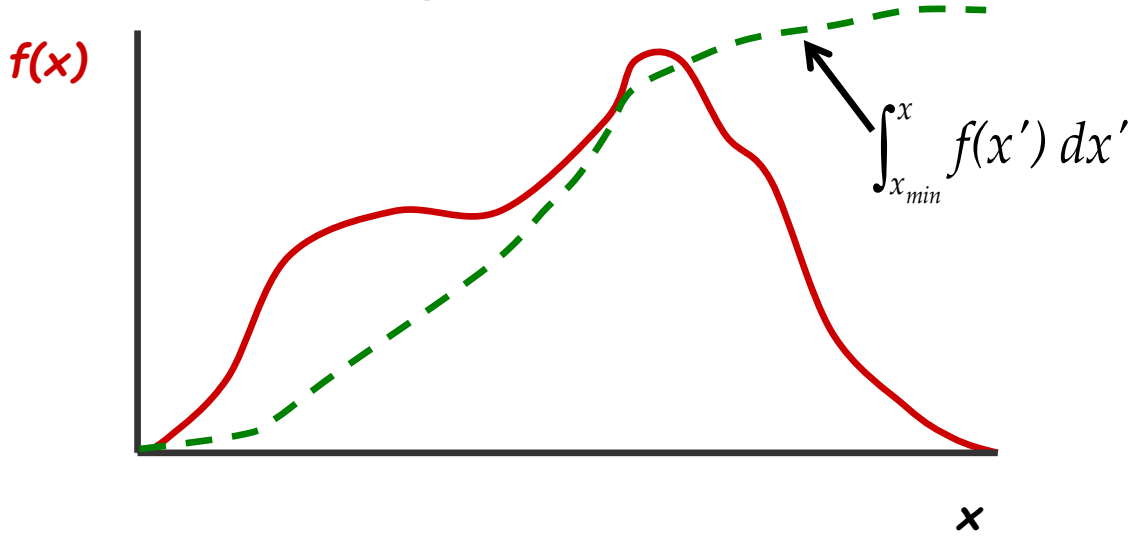
Integration without simulation

- In many cases, it is more efficient to replace the actual process by a different one resulting in the same average values but built by sampling from modified distributions
- Such a *biased process*, if based on mathematically correct variance reduction techniques, converges to the same expectation values as the unbiased one
- But it cannot provide information about the higher moments of statistical distributions (fluctuations and correlations)
- In addition, the faster convergence in some user-privileged regions of phase space is compensated by a slower convergence elsewhere

Random sampling: the key to Monte Carlo

The central problem of the Monte Carlo method:

Given a Probability Density Function (pdf), ***f(x)***, generate a sample of ***x***'s distributed according to ***f(x)*** (***x*** can be multidimensional)



The use of random sampling techniques is the distinctive feature of Monte Carlo

Solving the integral Boltzmann transport equation by Monte Carlo consists of:

- Geometry and material description of the problem
- Random sampling from probability distributions of the outcome of physical events

(Pseudo)random numbers

- The basis of all Monte Carlo integrations are random numbers, i.e. random values of a variable distributed according to a pdf
- In real world: the random outcomes of physical processes
- In computer world: pseudo-random numbers
- The basic pdf is the uniform distribution: $f(\xi)=1 \quad 0 \leq \xi < 1$

- Pseudo-random numbers (PRN) are sequences that reproduce the uniform distribution, constructed from mathematical algorithms (PRN generators).
- A PRN sequence looks random but it is not: it can be successfully tested for statistical randomness although it is generated deterministically
- A pseudo-random process is easier to produce than a really random one, and has the advantage that it can be reproduced exactly
- PRN generators have a period, after which the sequence is identically repeated. However, a repeated number does not imply that the end of the period has been reached. Some available generators have periods $>10^{61}$

Generatori di numeri pseudo-casuali

Il primo generatore di numeri casuali e' stato il *generatore di von Neumann*. La generazione avviene nel modo seguente:

a) si parte da un numero intero di $2m$ cifre n_1 e se ne considerano le m cifre centrali, ottenendo cosi' il numero k_1;

b) si quadra poi k_1 e si ottiene cosi' l'intero di $2m$ cifre n_2; si considerano le m cifre centrali e si ottiene k_2.

c) Proseguendo in questo modo si ottiene una sequenza di interi k_1, k_2,.... e, dividendo ciascuno di essi per 10^m, si ottiene una sequenza di numeri reali (razionali) $k_1, k_2, \ldots$ nell'intervallo $(0; 1)$.

Questo generatore ha mostrato diversi problemi e riveste piu' che altro importanza storica.

tutte le librerie matematiche dei vari linguaggi di programmazione hanno già implementato un generatore di numeri pseudo-random in [0,1)

Esempi:

1) Root:

```
TRandom2 *rnd = new TRandom2();
....
// Estrazione uniforme in [0,1)
Dpuble_t x = rnd->Uniform(0,1);
```

2) Python

```
import random as rnd
....
#Estrazione uniforme in [0,1)
x = rnd.random()
```

Sampling from a distribution

Sampling from a discrete distribution:

Suppose we have a *discrete* random variable x, that can assume values $x_1, x_2, \ldots, x_n, \ldots$ with probability $p_1, p_2, \ldots, p_n, \ldots$

- Assume $\Sigma_i p_i = 1$, or normalize it
- Divide the interval [0,1) in n subintervals, with n+1 limits

$y_0 = 0,\ y_1 = p_1,\ y_2 = p_1+p_2,\ y_3 = p_1+p_2+p_3, \ldots$

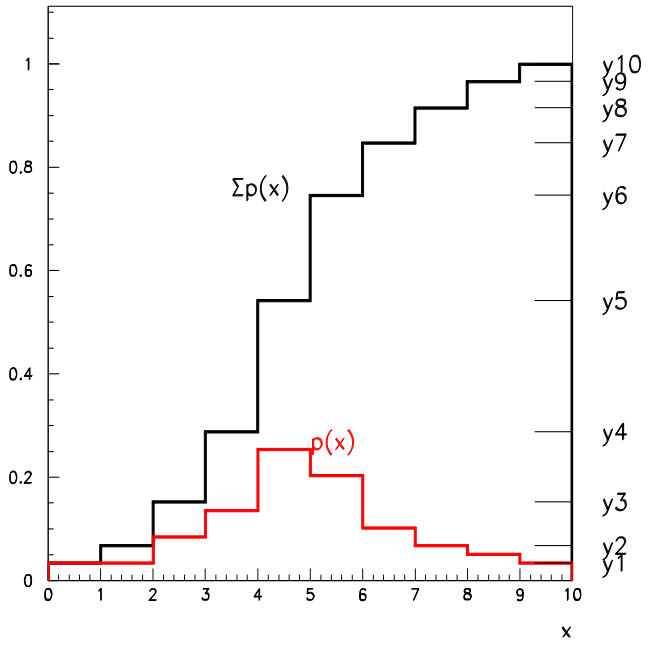
Note the use of the cumulative probability!

- Generate a uniform pseudo-random number $\xi \in [0,1)$
- Find the i^{th} y-interval such that

$$y_i - 1 < \xi < y_i$$

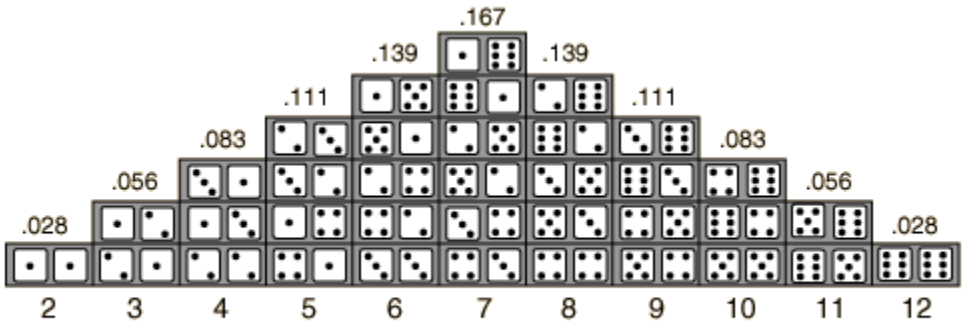
- Select $X = x_i$ as the sampled value
- Since ξ is uniformly random:

$$P(x_i) = P(y_{i-1} \leq \xi < y_i) = y_i - y_{i-1} = p_i$$



Sampling from a distribution

Example: simulate a throw of pair of dices:



Total number of states: 36

11 possible results

$x_1 = 2, x_2 = 3, x_3 = 4, \ldots, x_{10} = 11, x_{11} = 12$

$p_1 = 1, p_2 = 2, p_3 = 3, \ldots, p_{11} = 1$

[$p_1 = 0.028, p_2 = 0.056, p_3 = 0.083, \ldots, p_{11} = 0.028$ (normalized)]

Cumulative values:

$y_0 = 0, y_1 = 1, y_2 = 1+2 = 3, y_3 = 1+2+3 = 6, \ldots, y_{11} = 36$

Normalized:

$y_0 = 0, y_1 = 1/36 = 0.028, y_2 = 3/36 = 0.083, y_3 = 0.166, \ldots, y_{11} = 1$

Get a pseudorandom number ξ , e.g.: 0.125

ξ is found to be between $y_2 = 0.083$ and $y_3 = 0.167$

$P(y_3 - y_2) = y_3 - y_2 = 0.166 - 0.083 = 0.083 = p_3$

So, our sampled dice throw is $x_3 = 4$

Sampling from a continuous distribution

Sampling from a generic continuous distribution:

- Integrate the distribution function, *f(x)*, analytically or numerically, and normalize to 1 to obtain the normalized cumulative distribution:

$$F(\xi) = \frac{\int_{x_{\min}}^{\xi} f(x)\,dx}{\int_{x_{\min}}^{x_{\max}} f(x)\,dx}$$

Again, we use the cumulative probability: remember, MC is integration!

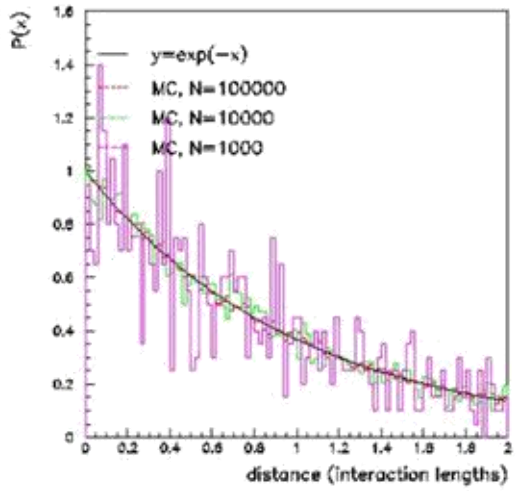
- Generate a uniform pseudo-random number ξ
- Get a sample of *f(x)* by finding the inverse value $X = F^{-1}(\xi)$, analytically or most often numerically by interpolation (table look-up)
- Since ξ is uniformly random:

$$P(a \le x < b) = P[F(a) \le \xi < F(b)] = F(b) - F(a) = \int_a^b f(x)dx$$

Sampling from a distribution

Example: sampling from an exponential distribution (this is frequently needed in particle transport, to find the point of next interaction or the distance to decay)

$$f(x) = e^{-x/\lambda}\ ,\ x \in [0,\infty)$$



- Cumulative distribution: $F(t)=\int_0^t e^{-\frac{x}{\lambda}}dx=\lambda(1-e^{-\frac{t}{\lambda}})$
- Normalized: $F'(t)=\int_0^t \frac{e^{-\frac{x}{\lambda}}}{\lambda}dx=1-e^{-\frac{t}{\lambda}}$
- Sample a uniform **ξ** ϵ [0,1), e.g.: 0.745 $\xi = F'(t) = 1 - e^{-\frac{t}{\lambda}} = 0.745$
- Sample **t** by inverting: $t = -\lambda \ln(1-\xi)$
- But ξ is distributed like 1 – ξ. Therefore our sampled value is:

$$t = -\lambda \ln \xi = -\lambda \ln 0.745 = 0.294\ \lambda$$

- If we are sampling the next interaction point, we will make a step of 0.294 mfp