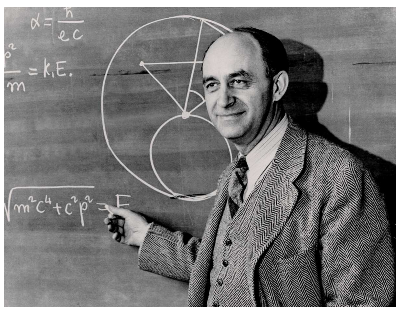
Montecarlo Methods

Alessandro Torcini

LPTM - Université de Cergy-Pontoise

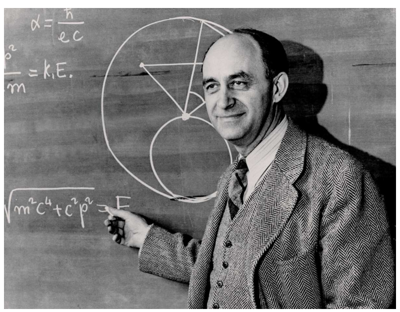


The course

1. CM Tuesday : 9.00-12.00 (E428) all
2. TP Tuesday : 15.30-18.30 (E428) (29p) (French)
3. TP Wendsday : 14.00-17.00 (E428) (30p) (English)

The term Monte Carlo Method refers to any numerical method employing random numbers to solve a problem in a probabilistic manner.
These methods are largely used in science :

1. to simulate stochastic and deterministic processes ;
2. to perform approximate numerical estimation (integrals etc.)
3. to simulate the response of experimental apparatus
4. to find a minimum or a maximum of a function (simulated annealing)



Brief History

1. **1930s** First significant scientific application of MC : Enrico Fermi used it for neutron transport in fissile material. Segre : “Fermi took great delight in astonishing his Roman colleagues with his ”too-good-to-believe” predictions of experimental results.”
2. **1940s** Monte Carlo named by Nicholas Metropolis and Stanislaw Ulam
3. **1953** Algorithm for sampling any probability density Metropolis, Rosenbluth, and Teller (generalized by Hastings in 1970)
4. **1962,1974** First QMC calculations, Kalos, Levesque, Verlet.

Program of the Course

1. Integral Estimation with Monte Carlo Methods
2. Statistical Physics and Equilibrium Thermodynamics (Canonical Ensemble)
3. The Montecarlo Method
4. Applications
   (a) Ising model
   (b) Simulated annealing
   (c) etc

Integration of a functon in 1d

One of the most employed deterministic methods consists in dividing the interval of definition of a function $f(x)$ in small sub-intervals :

Simple approximation

The most simple estimation of the integral of a function f over the interval $[a, b]$ can be obtained by estimating the surface of a rectangle of sides $(b-a)$ and $f(a)$ (the inferior limit of the function)

$$\int_a^b \mathrm{d}x\, f(x) \approx (b-a)\, f(a) =: F_1\,.$$

In order to estimate the error done in such an estimation of the real integral, we uses of the Taylor expansion of f around a

$$f(x) = f(a) + (x-a)\, f'(a) + \ldots\,,$$

therefore

$$\int_a^b \mathrm{d}x\, f(x) = \int_a^b \mathrm{d}x\, \left[f(a) + (x-a)\, f'(a)\right] + \ldots = F_1 + \frac{(b-a)^2}{2}\, f'(a) + \ldots\,.$$

The error made with the approximation $F1$ is proportional to $(b-a)^2$.

Method of the median point

To obtain a better estimation it is sufficient to consider the median point of the interval $[a, b]$ to estimate the integral of f :

$$\int_a^b \mathrm{d}x\, f(x) \approx (b-a)\, f\left(\frac{a+b}{2}\right) =: F_2\,.$$

By considering the Taylor expansion around the point $x_m = \frac{a+b}{2}$

$$f(x) = f\left(x_m\right) + \left(x - x_m\right)\, f'\left(x_m\right) + \frac{\left(x-x_m\right)^2}{2}\, f''\left(x_m\right) + \ldots$$

one gets

$$\begin{aligned}\int_a^b \mathrm{d}x\, f(x) &= \int_a^b \mathrm{d}x \left(f\left(x_m\right) + \left(x - x_m\right)\, f'\left(x_m\right) + \frac{\left(x-x_m\right)^2}{2}\, f''\left(x_m\right)\right) + \ldots \\ &= F_2 + \frac{(b-a)^3}{24}\, f''(x_m) + \ldots\,.\end{aligned}$$

Now the eror of the approximation F2 is proportional to $(b-a)^3$. and if $b-a$ is small, the approximation F2 is better than the approximation given by F1.

Simpson Method

By employing the lower and upper bound of the interval, as well the median point one can obtain an even better approximation of the integral :

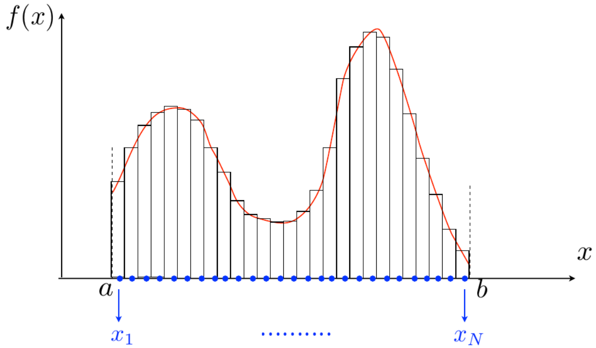
$$\int_a^b \mathrm{d}x\, f(x) \approx \frac{b-a}{6} \left(f(a) + 4\, f\left(\frac{a+b}{2}\right) + f(b) \right) =: F_3 \,.$$

One can use once more the Taylor expansion to estimate the error done with such approximation, after very long calculus . . . one gets

$$\int_a^b \mathrm{d}x\, f(x) = F_3 + \frac{(b-a)^5}{2880}\, f^{(iv)}(\xi)$$

with $\xi \in [a, b]$.

Equidistant values



The approximations are good for small $b-a$, therefore one should divide the interval $[a,b]$ in small sub-intervals and estimate the integral on each of that. By taking N intervals of the same length,

$$\Delta x = \frac{b-a}{N}$$

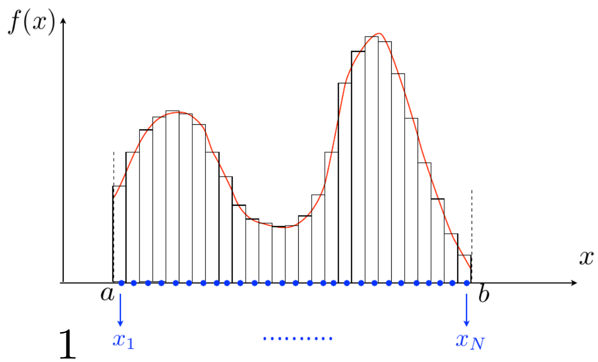
with extrema $a_i := a + i\,\Delta x\,,\qquad b_i := a + (i+1)\,\Delta x$ for $i = 0, \ldots, N-1$. Obviously, $a_0 = a$, $b_{N-1} = b$ et $a_{i+1} = b_i$. Finally, one can utilize the expressions for the approximations F1, F2 and F3, for the sub-intervals $[a_i, b_i]$.

In the case of the method of the median point one gets :

$$\int_a^b \mathrm{d}x\, f(x) \approx \Delta x \sum_{i=0}^{N-1} f\left(a + \left(i + \frac{1}{2}\right)\Delta x\right) =: F_2(N)\,.$$

In this case, for every sub-interval the error is proportional to Δx^3. Since we have N intervals, the total error is proportional to $N\,\Delta x^3 = (b-a)^3/N^2 \propto 1/N^2$.

Integration errors



1. Simple approximation

$$\left| \int_a^b \mathrm{d}x\, f(x) - F_1(N) \right| \propto \frac{1}{N}$$

2. Method of the median point

$$\left| \int_a^b \mathrm{d}x\, f(x) - F_2(N) \right| \propto \frac{1}{N^2}$$

3. Simpson Method

$$\left| \int_a^b \mathrm{d}x\, f(x) - F_3(N) \right| \propto \frac{1}{N^4}$$

Higher dimensions

Let us consider an hypercube $Q = [a_1, b_1] \times \ldots \times [a_d, b_d]$ in d dimensions, in this case the d-dimensonal integral can be written as

$$\int\limits_Q \mathrm{d}^d x\, f(\vec{x}) = \int\limits_{a_d}^{b_d} \mathrm{d}x_d \ldots \int\limits_{a_1}^{b_1} \mathrm{d}x_1\, f(x_1, \ldots, x_d)\,.$$

One can use one of the three methods descibed so-far, if we discretize each dimension with N points the total number of points we have is $N_T = N^d$ or otherwise $N = \sqrt[d]{N_T} = (N_T)^{1/d}$.

The error for the medina point is therefore

$$\left| \int\limits_Q \mathrm{d}^d x\, f(\vec{x}) - F_2(N) \right| \propto N_T^{-2/d}\,.$$

The error for the simple approximation is proportonal to $N_T^{-1/d}$ and for the Simpson method is proportional to $N_T^{-4/d}$

Higher dimensions

In higher dimensions one need many more N_T points to get a precise result :

1. Let us assume that $N = 100$ for each dimension and we are in $d = 10$ dimensions, therefore we must estimate the sum over $N_T = 100^{10} = 10^{20}$ points ;
2. if our computer needs 10^{-10} secs to estimate each element of the sum
3. the total CPU time required by the computer is $T \simeq 10^{10} secs \simeq 1000$ years

We need another method

Montecarlo Method

How to estimate the area of a surface

We want to estimate the area A_F of a surface $F \subset \mathbb{R}^2$ in 2 dimensions. Le us use the characteristic funtion χ_F of F,

$$\chi_F(\vec{x}) = \begin{cases} 1 & \text{si } \vec{x} \in F \\ 0 & \text{si } \vec{x} \notin F \end{cases}$$

We should therefore estimate a two dimensional integral. We can approximate this integral with a so-called Riemann sum, namely

$$A_F = \int_{\mathbb{R}^2} \mathrm{d}^2 x \, \chi_F(\vec{x}) \approx \sum_i \chi_F(\vec{x}_i) \, A_i \, .$$

where $\vec{x}_i$ are points in the plane, each one surrounded by a surface A_i.
One would think to use a regular distribution of the points $\vec{x}_i$, but this is not needed. We can use a completely random distribution

The Monte-Carlo method

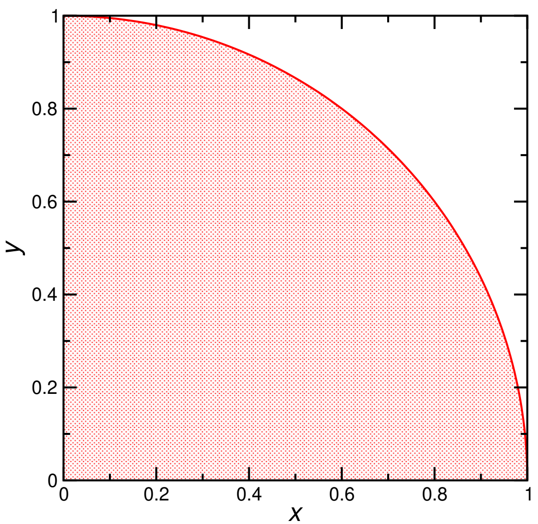
the Hit and Miss method

In practice, one should include the surface F p.-ex. within a rectangle R.

The area A_R of the rectangle is known and if we consder N points inside R, their average area is $A_i = A_R/N$.

Therefore for $F \subset R$ the area can be approximated as

$$A_F = \int_R \mathrm{d}^2 x \, \chi_F(\vec{x}) \approx \frac{A_R}{N} \sum_{i=1}^{N} \chi_F(\vec{x}_i) \,.$$



Exercise (*Estimate π with the Hit and Miss Method*) **:** Consider $N = 100\ 000$ couples (x_i, y_i) with $x_i, y_i, \in [0, 1)$. Now you should count the number of cases for which $x_i^2 + y_i^2 < 1$ and estimate the probabilty of having the point (x_i, y_i) inside the circle see the Figure. Estimate the error for such evaluation and compare it with $\pi/4$.

Please employ `np.random.random_sample()` . Random generator within $[0 : 1)$ with uniform distribution.

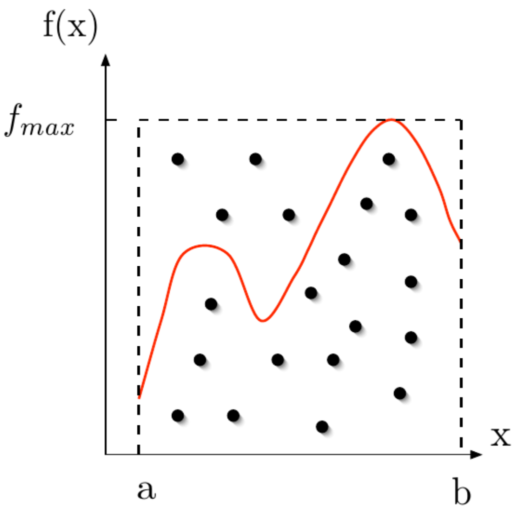
Hit and Miss method

```
# pi and the Monte-Carlo Method
import numpy as np            # importer le module comme "np"
N=1000                        # number of points points
cnt = 0
var = 0

for i in range(0,N):
  x = np.random.random_sample() # generate the couple (x,y)
  y = np.random.random_sample()
  if x*x + y*y < 1:             # Is the couple within the circle ?
    cnt += 1                    # If yes count them
    var += 1**2

moy = cnt/float(N)              # the average
var=  var/float(N) -moy*moy     #  the variance
ecart = np.sqrt(var/N)          # standard deviation of the average

print "compte =", moy, "+/-", ecart
print "pi/4   =", np.pi/4
print "(error =", np.abs(moy-np.pi/4), ")"
```

Hit and Miss method : Function

The Hit and Miss Method for a function $f(x)$

We want to estimate $A = \int_a^b f(x)dx$



$$A_{estimate} = \frac{N_a}{N} f_{max}(b-a)$$

1. N points uniformely chosen within the square
2. N_a : number of points that “fall”” under $f(x)$
3. The probabilty to fall below the curve is $p_a \simeq \frac{N_a}{N}$

The following is true

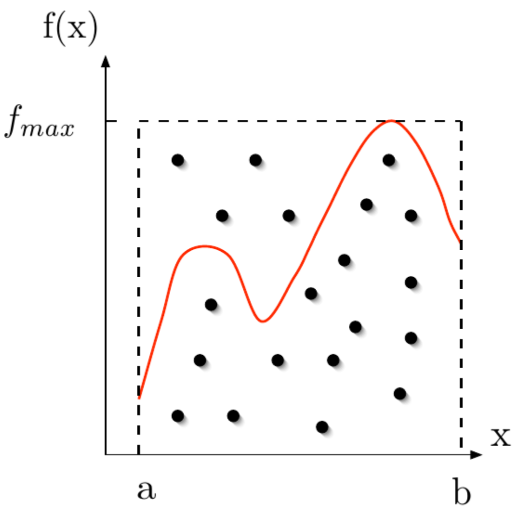
$$\lim_{N\to\infty} A_{estimate} = A \qquad \lim_{N\to\infty} p_a = p$$

but the convergence is very slow , you need a lot of points N

Hit and Miss Method

The blind archer

An archer is throwing uniformly and randomly arrows in points (x_i, y_i) on the plane



1. if $y_i \leq f(x_i) \rightarrow$ YES (0)
2. if $y_i > f(x_i) \rightarrow$ NO (1)
3. This is a Bernoulli process with probability p
4. We perform N times the random experiment to throw arrows on the plane

   $(x_1, y_1), (x_2, y_2), (x_3, y_3) \ldots (x_N, y_N)$

   and N_a times we will be below the curve $f(x)$.
5. N_a is random and follows a binomial distribution

   (a) average pN

   (b) standard deviation $\sqrt{Np(1-p)}$

Hit and Miss Method

The blind archer

The estimate of the area below the function $f(x)$ is also a random variable

$$A_{estimate} = \frac{N_a}{N} f_{max}(b-a)$$

1. Average $\langle A_{estimate} \rangle = p f_{max}(b-a)$
2. Standard Deviation $\sigma_A = \frac{\sqrt{Np(1-p)}}{N} f_{max}(b-a) = \sqrt{\frac{p(1-p)}{N}} f_{max}(b-a)$

The error is decreasing as $1/\sqrt{N}$ with the number of trials N, as expected from the Central Limit Theorem and the $A_{estimate}$ will be distributed for different realizations of the random sequence of the points as a Gaussian distribution with average $\bar{A}_{estimate}$ and standard deviation σ_A.

Blind Archer $\rightarrow$ Random generator

Hit and Miss method

```
# Integration of a function with Hit and Miss Method
import numpy as np                    # import the library as "np"

def HM(f, a, b, fmax):
  #
  somme = 0.
  i = 0
  while i<HM.N:
    x = np.random.random_sample() # the blind archer
    y = np.random.random_sample() # the blind archer
    xi = (b-a)*x+a                # random number in [a,b]
    fi = f(xi)                    # value of the function in xi
    if fi > fmax*y :
      somme += 1
    i += 1
  somme /= float(HM.N)                   # probability p
  area  =  somme*fmax*(b-a)              # average area
  HM.erreur = np.sqrt(somme*(1-somme)/float(HM.N))*fmax*(b-a)
  #standard deviation on the average
  return area
```

Hit and Miss method

```
def f1(x):          # the function
    return x**4

# we can estimate the area over many realizations N
for N in [1e2, 1e3, 1e4]:
  print "N =", N
  HM.N = N
  print ("I1 = ", HM(f1, 0, 1,1), "+/-", HM.erreur, "(exact : 1/5)")
```

Idle HitandMiss.py

Uniform Sampling

We consider again the integral

$$I = \int_a^b f(x)dx$$

but we rewrite it as

$$I = \int_a^b G(x)p(x)dx = \langle G \rangle_p \qquad \text{with} \quad p(x) = \frac{1}{(b-a)} \quad G(x) = (b-a)f(x)$$

where $p(x)$ is the uniform probability distribution function on the interval $[a, b]$, please notice that $\int_a^b p(x)dx = 1$
As already previously explained $I = \langle G \rangle_p$ is the average of the function $G(x)$ with respect to the random variable x uniformly distributed in $[a, b]$:

$$x = a + (b-a) * r \quad r \in [0, 1]$$

r can be generated with `np.random.random_sample()` in $[0, 1]$.

Uniform Sampling

Pratical Implementation

We employ the random generator and we produce a sequence of N random numbers $(x_1, x_2, \ldots, x_N)$ from these we can compute the mean the variance of G

$$\langle G \rangle = \frac{1}{N} \sum_{i=1}^{N} G(x_i)$$

$$\sigma_N^2(G) = \frac{1}{N} \sum_{i=1}^{N} G^2(x_i) - \left(\frac{1}{N} \sum_{i=1}^{N} G(x_i) \right)^2$$

and from these the integral with the erron on the average

$$I = \langle G \rangle \pm \frac{\sigma_N(G)}{\sqrt{N}}$$

in terms of the original function f this becomes finally

$$\boxed{I = (b-a) \left[\langle f \rangle \pm \frac{\sigma_N(f)}{\sqrt{N}} \right]}$$

the error on the integral decreases as $1/\sqrt{N}$

Generalization to higher dimensions

The uniform sampling Montecarlo method can be easily generalized to estimate integrals in dimension larger then one.

In general, if R is a region in d dimensions with a volume V_R, l'integral is given by

$$\int_R \mathrm{d}^d x \, f(\vec{x}) \approx \frac{V_R}{N} \sum_{i=1}^{N} f(\vec{x}_i) \, .$$

where $\vec{x}_i = (x_i^{(1)}, x_i^{(2)}, \ldots, x_i^{(d)}) \in R$ is a random vector, where each component is uniformly distributed.

In two dimensons ($d = 2$) the region $R = [a, b]x[c, d]$ is a rectangle with « volume » $V_R = (b - a) \times (d - c)$. Therefor ein 2d one has

$$\int_a^b \int_c^d \mathrm{d}x \, \mathrm{d}y \, f(x, y) \approx \frac{(b-a) \times (d-c)}{N} \sum_{i=1}^{N} f(x_i, y_i) \, ,$$

where (x_i, y_i) are random points uniformly distributed in R.

Comparison Monte Carlo and deterministc methods

The error for the MC method for any dimension remains $\propto \frac{1}{\sqrt{N}}$

Instead for the deterministic methods (quadrature) the error increases with the dimension d.

For the method of the median point the error was growing as $1/N^{2/d}$, therefofre the Monte Carlo method becomes more efficient for

$$\frac{1}{2} > \frac{2}{d},$$

when

$$d > 4$$

For the Simpson method the MC method becomes more efficient for

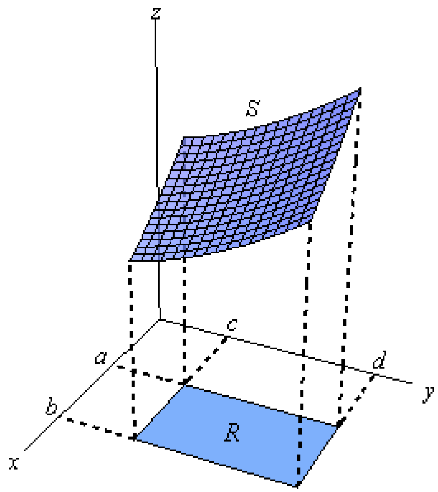
$$\frac{1}{2} > \frac{4}{d},$$

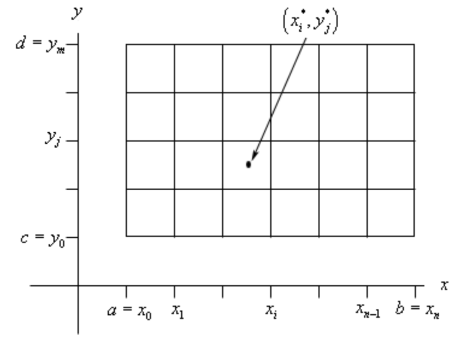
when

$$d > 8$$

and for the simple approximation, already for $d > 2$.

Method of the median point for 2 dimensions





We would liek to integrate the two dimensional function $f(x, y)$ over the rectangle $R = [a, b] \times [c, d]$ with the method of the median point. As a first step we will divide the interval $[a, b]$ in N equidistant sub-intervals and $[c, d]$ in M equidistant sub-intervals, each of length

$$\Delta x = \frac{b-a}{N} \qquad \Delta y = \frac{d-c}{M}$$

and with extrema

$$a_i := a + i\,\Delta x\,, \qquad b_i := a + (i+1)\,\Delta x \qquad c_j := c + j\,\Delta y\,, \qquad d_j := c + (j+1)\,\Delta x$$

for $i = 0, \ldots, N-1$ et $j = 0.\ldots.M-1$. Obviously,, $a_0 = a$, $b_{N-1} = b$ and $a_{i+1} = b_i$; $c_0 = c$ and $d_{M-1} = d$.

Method of the median point for 2 dimensions

The method of the median point in 2d is given by :

$$\int_a^b \mathrm{d}x \int_c^d \mathrm{d}y\, f(x,y) \approx \Delta x \times \Delta y \sum_{i=0}^{N-1} \sum_{j=0}^{M-1} f(x_i^*, y_j^*) =: F_2^{2d}(N,M)\,.$$

where $x_i^* = a + \left(i + \frac{1}{2}\right) \Delta x$ and $y_j^* = c + \left(j + \frac{1}{2}\right) \Delta y$

For the SImpon method, the expression in 2d is really complicated, see
http ://mathfaculty.fullerton.edu/mathews/n2003/SimpsonsRule2DMod.html

Exercise : Please estimate the following integral with the method of the median point and with the Monte Carlo method :

$$I_{2d} = \int_3^5 dx \int_2^9 dy (x^2 + 10y)$$

Method of the median point for 2 dimensions

$$I_{2d} = \int_3^5 dx \int_2^9 dy(x^2 + 10y) = \int_2^9 dy \left[\frac{x^3}{3} + 10xy\right]_{x=3}^{x=5} =$$

$$= \int_2^9 dy \frac{98}{3} + 20y = \left[\frac{98}{3}y + 10y^2\right]_{y=2}^{y=9} = \frac{686}{3} + 770 \simeq 998.666$$

Summary

In summary the Monte Carlo method with uniform sampling is :

1. a very simple method
2. that converges for any dimension
3. but the method is not very efficient, because the points x_i [or $(x_i, y_i), \ldots$] are not selected with regards to their weight (that is, their “importance”...) in the integral !

We need a better sampling !