Computational Methods in Nonlinear Physics
2. Monte Carlo integration

P.I. Hurtado

Departamento de Electromagnetismo y Física de la Materia, and Instituto Carlos I de Física Teórica y Computacional. Universidad de Granada. E-18071 Granada. Spain

E-mail: phurtado@onsager.ugr.es

Abstract. These notes correspond to the first part (20 hours) of a course on Computational Methods in Nonlinear Physics within the Master on Physics and Mathematics (*FisyMat*) of University of Granada. In this second chapter we describe the use of random numbers and Monte Carlo methods to perform integrals, and the adecuacy of this method for high-dimensional problems.

Keywords: Computational physics, probability and statistics, Monte Carlo methods, stochastic differential equations, Langevin equation, Fokker-Planck equation, molecular dynamics.

References and sources

[1] R. Toral and P. Collet, *Stochastic Numerical Methods*, Wiley (2014).

Contents

In this chapter, we review the basic algorithms for the calculation of integrals using random variables and define the general strategy based on the replacement of an integral by a sample mean.

1. Hit and miss

- The hit-and-miss method is the simplest of the integration methods that use ideas from probability theory. Although it is not very competitive in practical applications, it is very convenient in order to explain in a simple and comprehensive manner some of the ideas of more general methods.

- Let $y = g(x)$ be a real function which takes only bounded *positive* values in a finite interval $[a, b]$, that is, $0 \leq g(x) \leq c$. In Riemann's sense, the integral

$$I = \int_a^b g(x)\, dx \tag{1}$$

is nothing but the area of the region Ω between the x-axis and the curve given by $g(x)$.

- In the rectangle $S = \{(x, y), a \leq x \leq b, 0 \leq y \leq c\}$, we consider a pair of independent random variables $(\hat{\mathbf{x}}, \hat{\mathbf{y}})$ which are uniformly distributed in S, see Fig. 1. The joint probability density function of $\hat{\mathbf{x}}$ and $\hat{\mathbf{y}}$ is

$$f_{\hat{\mathbf{x}}\hat{\mathbf{y}}}(x, y) = \begin{cases} \dfrac{1}{c(b-a)} & (x, y) \in S\,, \\ \\ 0 & (x, y) \notin S\,. \end{cases} \tag{2}$$

- The probability that a point (x, y) distributed according to $f_{\hat{\mathbf{x}}\hat{\mathbf{y}}}(x, y)$ lies within Ω is,

$$p = \int_\Omega f_{\hat{\mathbf{x}}\hat{\mathbf{y}}}(x, y)\, dx\, dy = \frac{1}{c(b-a)} \int_\Omega dx\, dy = \frac{I}{c(b-a)}\,, \tag{3}$$

as $\int_\Omega dx\, dy$ is the area of Ω, or equivalently the integral I.

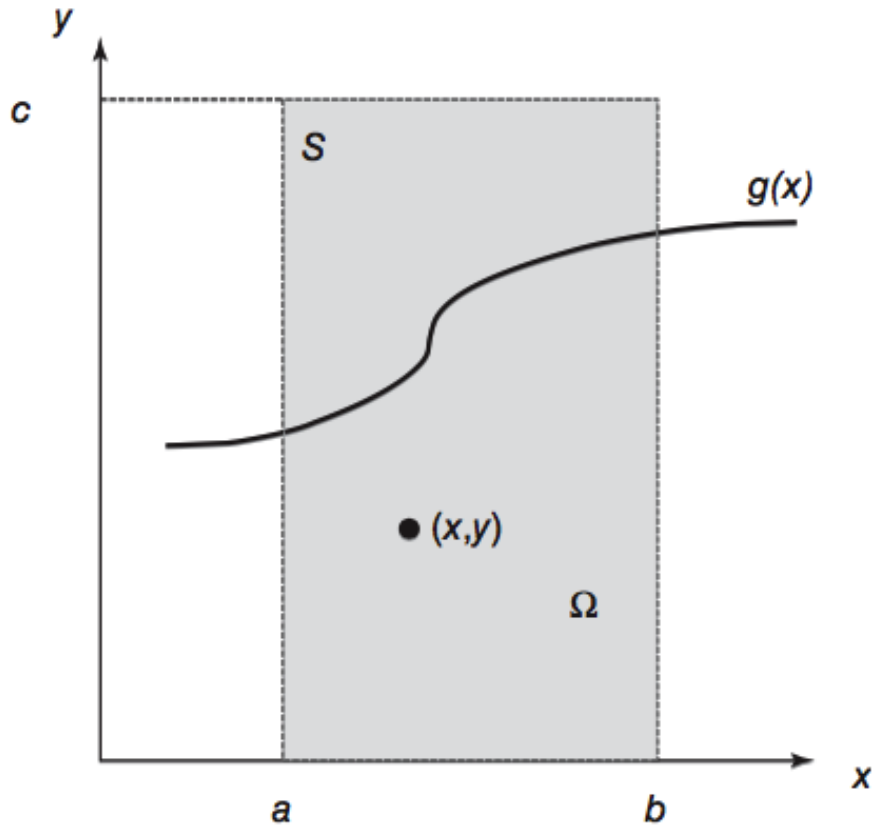

Figure 1. Schematics and definitions of the hit-and-miss method.

- The idea of the Monte Carlo integration is to read this equation as $I = c(b-a)p$. So, if we know the probability p, then we know the integral I
- But how can we obtain the probability p that a point distributed according to a uniform distribution lies in region Ω? Simple: perform an experiment whose outcome is the pair (x, y), distributed according to Eq. (2), and compute from that experiment the probability p.

- It might help to imagine an archer who is sending arrows uniformly to region S. If he sends 100 arrows, and 78 of them are in region Ω, then an estimation of the probability is $p = 0.78$. The hit-and-miss method is a sophisticated manner of replacing the archer, giving us, hopefully, a precise value for p as well as the error in the calculation of the integral I.

- Let us imagine that we have an *ideal archer* who is able to send arrows that fall uniformly (and this is the key word) in points (x, y) of S. The point (x, y) can or cannot fall within Ω and, hence, the event *"the point* (x, y) *is in* Ω*"* can take the values *"yes"* or *"no"*, a binary variable that follows a Bernoulli distribution.

- We make M independent repetitions of this Bernoulli experiment, generate the points (x_1, y_1), (x_2, y_2), ..., (x_M, y_M) uniformly distributed in S, and define the random variable $\hat{\mathbf{N}}_B$ as the number of points that belong to Ω, that is, those satisfying $y_i \leq g(x_i)$.

- We introduce a random variable $\hat{\mathbf{I}}_1 = c(b-a)\hat{\mathbf{p}} = c(b-a)\hat{\mathbf{N}}_B/M$, where we have defined $\hat{\mathbf{p}} = \hat{\mathbf{N}}_B/M$. As $\hat{\mathbf{N}}_B$ clearly follows a binomial distribution with mean value pM, then $\langle \hat{\mathbf{p}} \rangle = p$ and it results that $\langle \hat{\mathbf{I}}_1 \rangle = I$. Such a random variable whose average value is equal to I is called an unbiased estimator of the integral.

- Using the results of the binomial distribution, we can compute also the standard deviation of this estimator as

$$\sigma[\hat{\mathbf{I}}_1] = \frac{c(b-a)}{M}\sigma[\hat{\mathbf{N}}_B] = \frac{c(b-a)}{M}\sqrt{Mp(1-p)} = c(b-a)\sqrt{\frac{p(1-p)}{M}}\,. \tag{4}$$

- Therefore, using the short-hand notation we derived from Chebyshev's theorem, we can write

$$I = \hat{\mathbf{I}}_1 \pm \hat{\sigma}[\hat{\mathbf{I}}_1] = c(b-a)\hat{\mathbf{p}} \pm c(b-a)\sqrt{\frac{\hat{\mathbf{p}}(1-\hat{\mathbf{p}})}{M}} \tag{5}$$

  with the usual confidence intervals for the error.

- If we make a large number of repetitions M, we can approximate the binomial distribution by a Gaussian distribution, and the confidence levels indicate that there is a 68% probability that the integral is indeed in the interval $(\hat{\mathbf{I}}_1 - \hat{\sigma}[\hat{\mathbf{I}}_1], \hat{\mathbf{I}}_1 + \hat{\sigma}[\hat{\mathbf{I}}_1])$, 95% that it is in the interval $(\hat{\mathbf{I}}_1 - 2\hat{\sigma}[\hat{\mathbf{I}}_1], \hat{\mathbf{I}}_1 + 2\hat{\sigma}[\hat{\mathbf{I}}_1])$, and so on. Consequently, we associate $\hat{\sigma}[\hat{\mathbf{I}}_1]$ to the error (in the statistical sense) of our estimator.

- The relative error is

$$\frac{\hat{\sigma}[\hat{\mathbf{I}}_1]}{\langle \hat{\mathbf{I}}_1 \rangle} \approx \sqrt{\frac{(1-p)}{pM}}\,, \tag{6}$$

  From this expression, we see (i) the relative error decreases as the inverse square root $M^{-1/2}$ of the number of repetitions M, and (ii) the error decreases with increasing p. Therefore, it is convenient to choose the, otherwise arbitrary, rectangle S as small as possible, which is equivalent to taking $c = \max[g(x)]$ in the interval $x \in [a, b]$.

- In a numerical algorithm, to replace our archer we need to generate random points (x, y) from a two-dimensional uniform distribution. This can be obtained by generating one random variable $\hat{\mathbf{x}}$ uniformly in the interval $[a, b]$ and another independent random variable $\hat{\mathbf{y}}$ uniformly in $(0, c)$. In practice, we use two independent random variables $\hat{\mathbf{u}}$, $\hat{\mathbf{v}}$ uniformly distributed in the interval $[0, 1]$ and the linear transformations $\hat{\mathbf{x}} = a + (b - a)\hat{\mathbf{u}}$, $\hat{\mathbf{y}} = c\hat{\mathbf{v}}$.

- Numerical implementation of the hit and miss method: We now implement the hit-and-miss method as a subroutine. The subroutine returns the estimator of the integral of an external function $g(x)$ in the variable r and the error in s.

```
! Subroutine for the hit and miss method
subroutine mc1(g,a,b,c,M,r,s)
implicit none
double precision, intent (in) :: a,b,c
integer, intent (in) :: M
double precision, intent (out) :: r,s
double precision :: g,p,u,v,ran_u
integer :: nb,i
! External function whose integral we want to compute
external g

nb=0                ! Counter for the number of hits
do i=1,M
   u=ran_u()        ! Homogeneous random number in (0,1]
   v=ran_u()        ! Homogeneous random number in (0,1]
   if (g(a+(b-a)*u) > c*v) nb=nb+1
enddo
p=dble(nb)/M
r=(b-a)*c*p
s=sqrt(p*(1.d0-p)/M)*c*(b-a)

end subroutine mc1
```

- For the sake of clarity, we now include an example of a full program for calculating the area of the function $g(x) = x^2$ in the interval $[0, 1]$ (run this simple program and check compatibility with the exact value $I = 1/3$)

```
program area
implicit none
interface
   double precision function g(x)
      double precision, intent (in) :: x
   end function g
end interface
double precision :: a,b,c,r,s
integer :: M
a=0.d0
b=1.d0
c=1.d0
M=100000
call mc1(g,a,b,c,M,r,s)
write (*,*) "The estimated value of the integral is", r
write (*,*) "The estimated error is", s
end program area

double precision function g(x)
implicit none
double precision, intent (in) :: x
g=x*x
end function g
```

2. Uniform sampling

- Let's explain now a second more sophisticated method to perform integrals using random numbers and ideas from probability theory.

- We consider again the integral

$$I = \int_a^b g(x)\, dx \tag{7}$$

but now $g(x)$ does not need to be necessarily limited to take nonnegative values.

- We write this integral as

$$I = \int_{-\infty}^{\infty} G(x) f_{\hat{\mathbf{x}}}(x)\, dx \tag{8}$$

with $G(x) = (b-a)g(x)$ and

$$f_{\hat{\mathbf{x}}}(x) = \begin{cases} \dfrac{1}{b-a} & a \leq x \leq b\,, \\ \\ 0 & \text{otherwise}\,. \end{cases} \tag{9}$$

- Note that this integral can be understood as the average value of the function $G(x)$ with respect to a random variable $\hat{\mathbf{x}}$ whose probability density function is $f_{\hat{\mathbf{x}}}(x)$, that is, a uniform random variable in the interval (a, b).

- If we now devise an experiment whose outcome is the random variable $\hat{\mathbf{x}}$ and repeat that experiment M times to obtain the values $\hat{\mathbf{x}}_1, \ldots, \hat{\mathbf{x}}_M$, we can compute from those values the sample mean and variance as

$$\hat{\boldsymbol{\mu}}_M[G] = \frac{1}{M} \sum_{i=1}^{M} G(x_i)\,, \tag{10}$$

$$\hat{\boldsymbol{\sigma}}_M^2[G] = \frac{1}{M} \sum_{i=1}^{M} G(x_i)^2 - \left(\frac{1}{M} \sum_{i=1}^{M} G(x_i) \right)^2 . \tag{11}$$

- Note that we have not included the term $M/(M-1)$ in the definition of $\hat{\boldsymbol{\sigma}}^2_M[G]$ as we will use these formulas for very large values of M, where this quotient tends to 1.

- The integral can be hence estimated as

$$I = \hat{\boldsymbol{\mu}}_M[G] \pm \frac{\hat{\boldsymbol{\sigma}}_M[G]}{\sqrt{M}} = (b-a)\hat{\boldsymbol{\mu}}_M[g] \pm (b-a)\frac{\hat{\boldsymbol{\sigma}}_M[g]}{\sqrt{M}}\,, \tag{12}$$

  where we have used that $G(x) = (b-a)g(x)$, $\hat{\boldsymbol{\mu}}_M[g]$ and $\hat{\boldsymbol{\sigma}}^2_M[g]$ are the sample mean and variance of the function $g(x)$, and where the error has to be understood in the statistical sense of Chebyshev's theorem as explained in the previous chapter.

- This method can be easily implemented now because we already know how to generate values of a random variable $\hat{\mathbf{x}}$ which is uniformly distributed in an interval (a,b). Let us give a simple computer program as follows

```
! Implementation of the uniform sampling method
subroutine mc2(g,a,b,M,r,s)
implicit none
double precision, intent (in) :: a,b
integer, intent (in) :: M
double precision, intent (out) :: r,s
double precision :: g,g0,ran_u
integer :: i
external g
```

```
r=0.d0
s=0.d0
do i=1,M
   g0=g(a+(b-a)*ran_u())
   r=r+g0
   s=s+g0*g0
enddo
r=r/M
s=sqrt((s/M-r*r)/M)
r=(b-a)*r
s=(b-a)*s

end subroutine mc2
```

- If we think of the Riemann integral as the limiting sum of the function $g(x)$ over an infinite number of points in the interval (a, b), the uniform sampling method replaces that infinite sum by a finite sum $(b-a)\hat{\boldsymbol{\mu}}_M[g]$ over points randomly distributed in the same interval.

3. General sampling methods

- Similar ideas can be used to compute the integral

$$I = \int g(x)\,dx = \int G(x) f_{\hat{\mathbf{x}}}(x)\,dx \tag{13}$$

- There are no restrictions about the function $G(x)$, but we demand that $f_{\hat{\mathbf{x}}}(x)$ has the required properties of a probability density function: that is, nonnegative and normalized.

- The key point is to understand the integral as the average value of the function $G(x)$ with respect to a random variable $\hat{\mathbf{x}}$ whose probability density function is $f_{\hat{\mathbf{x}}}(x)$, i.e. $I = E[G]$.

- If we devise now an experiment whose outcome is the random variable $\hat{\mathbf{x}}$ with probability density function $f_{\hat{\mathbf{x}}}(x)$ and repeat that experiment M times to obtain the values $x_1, \ldots, x_M$, we can compute from those values the sample mean and variance, and from them the integral can be approximated as $I = \hat{\boldsymbol{\mu}}_M[G] \pm \hat{\boldsymbol{\sigma}}_M[G]/\sqrt{M}$, as before.

- The main difference with respect to the uniform sampling is the substitution of the random variables drawn from the uniform distribution by values distributed according to a distribution $f_{\hat{\mathbf{x}}}(x)$.

- It is straightforward to write a computer program to implement this algorithm

```
! Implementation of the general sampling method
subroutine mc3(g,ran_f,M,r,s)
implicit none
integer, intent (in) :: M
double precision, intent (out) :: r,s
double precision :: g,ran_f,g0
```

```
integer :: i
external g,ran_f
r=0.d0
s=0.d0
do i=1,M
   g0=g(ran_f())
   r=r+g0
   s=s+g0*g0
enddo
r=r/M
s=sqrt((s/M-r*r)/M)
end subroutine mc3
```

- How do we obtain random numbers with some arbitrary pdf $f_{\hat{\mathbf{x}}}(x)$? We will briefly describe this issue in the next section

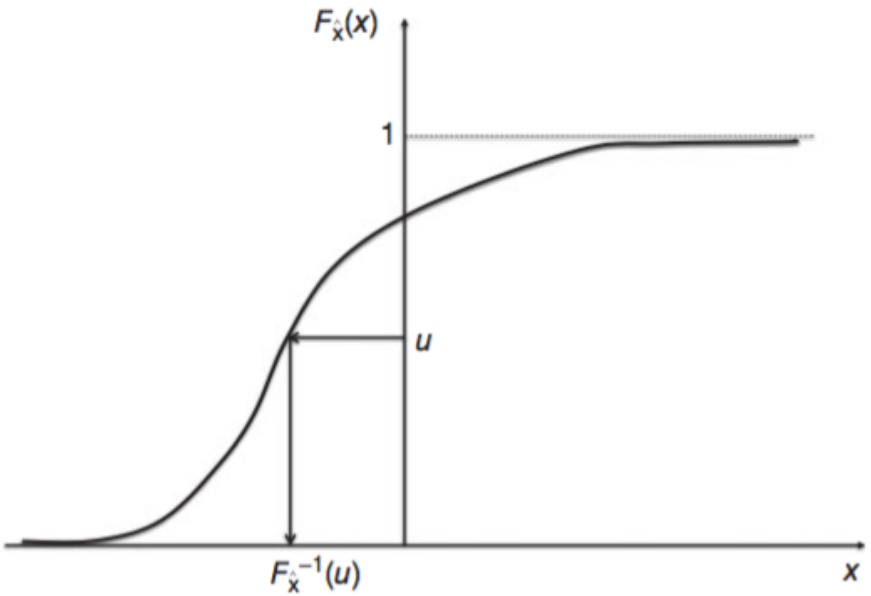

Figure 2. Inversion of the cdf $F_{\hat{\mathbf{x}}}(x)$. If u is the value of a random variable $\hat{\mathbf{u}}$ uniformly distributed in the $(0, 1)$ interval, then $x = F_{\hat{\mathbf{x}}}^{-1}(u)$ follows the corresponding pdf $f_{\hat{\mathbf{x}}}(x) = dF_{\hat{\mathbf{x}}}(x)/dx$.

4. Generation of nonuniform random numbers: Basic concepts

- The basic method to generate values of a random variable $\hat{\mathbf{x}}$ distributed according to the probability distribution function $f_{\hat{\mathbf{x}}}(x)$ uses a theorem that we proved in the previous chapter on probability.
- Theorem: If $\hat{\mathbf{x}}$ is a random variable whose cumulative probability function is $F_{\hat{\mathbf{x}}}(x)$, then the change of variables $\hat{\mathbf{u}} = F_{\hat{\mathbf{x}}}(\hat{\mathbf{x}})$ yields a random variable $\hat{\mathbf{u}}$ uniformly distributed in the interval $(0, 1)$, i.e. a $\hat{\mathbf{U}}(0, 1)$ variable.
- We now read this theorem backward: if $\hat{\mathbf{u}}$ is a $\hat{\mathbf{U}}(0, 1)$ random variable, then the probability distribution function of $\hat{\mathbf{x}} = F_{\hat{\mathbf{x}}}^{-1}(\hat{\mathbf{u}})$ is $f_{\hat{\mathbf{x}}}(x)$, see Figure 2.

- Example: We want to generate random numbers distributed according to an exponential distribution

$$f_{\hat{\mathbf{x}}}(x) = \begin{cases} a\mathrm{e}^{-ax} & x \geq 0\,, \\ 0 & x < 0\,. \end{cases} \tag{14}$$

We first compute the cumulative distribution

$$F_{\hat{\mathbf{x}}}(x) = \int_{-\infty}^{x} f_{\hat{\mathbf{x}}}(y)\, dy = 1 - \mathrm{e}^{-ax} \qquad x \geq 0\,, \tag{15}$$

and then invert $x = F_{\hat{\mathbf{x}}}^{-1}(u)$ by solving $u = F_{\hat{\mathbf{x}}}(x)$ to find $F_{\hat{\mathbf{x}}}^{-1}(u) = -\frac{1}{a}\log(1-u)$. This means that, to generate random numbers x distributed according to an exponential distribution, we need to generate numbers u uniformly distributed in the $(0,1)$ and then apply $x = -\frac{1}{a}\log(1-u)$. As $1 - \hat{\mathbf{u}}$ is obviously also a $\hat{\mathbf{U}}(0,1)$ variable, we can simply use the formula

$$x = -\frac{1}{a}\log(u) \tag{16}$$

- The interested reader can find in the book by Toral and Colet a number of different examples with details on how to obtain random numbers distributed according to some non-uniform pdfs, both continuous and discrete.

- In general, the main problem of this method is the lack of a good algorithm for the calculation of the inverse cumulative distribution function and, sometimes, it is a bottleneck for the implementation of the method described in this section.

5. Importance sampling

- Let us consider again the general integral

$$I = \int g(x)\,dx = \int G(x) f_{\hat{\mathbf{x}}}(x)\,dx \tag{17}$$

with $G(x) = g(x)/f_{\hat{\mathbf{x}}}(x)$ and $f_{\hat{\mathbf{x}}}(x)$ has to be a probability density function (nonnegative and normalized). In the sampling method, we consider this to be equal to the average $E[G]$ and use the approximation used by the sample mean.

- There are infinite ways in which the integral I can be decomposed as an average over some $f_{\hat{\mathbf{x}}}(x)$, although some might be more *natural* than others.

- For instance, take the integral

$$I = \int_0^\infty \cos(x) x^2 \mathrm{e}^{-x}\,dx = -\frac{1}{2}\,. \tag{18}$$

If we would like to use a sampling method for its calculation, possible *natural* choices would be

$$G^{(1)}(x) = \cos(x)x^2\,, \quad x \geq 0\,, \tag{19}$$

$$f^{(1)}_{\hat{\mathbf{x}}}(x) = \begin{cases} \mathrm{e}^{-x} & x \geq 0\,, \\ 0\,, & x < 0\,. \end{cases} \tag{20}$$

or

$$G^{(2)}(x) = \cos(x)x\,, \quad x \geq 0\,, \tag{21}$$

$$f^{(2)}_{\hat{\mathbf{x}}}(x) = \begin{cases} x\,\mathrm{e}^{-x} & x \geq 0\,, \\ 0\,, & x < 0\,. \end{cases} \tag{22}$$

or

$$G^{(3)}(x) = 2\cos(x)\,, \quad x \geq 0\,, \tag{23}$$

$$f_{\hat{\mathbf{x}}}^{(3)}(x) = \begin{cases} \frac{1}{2}x^2\,\mathrm{e}^{-x} & x \geq 0\,, \\ 0\,, & x < 0\,. \end{cases} \tag{24}$$

but we could have used, for example, a not-so-obvious splitting

$$G^{(4)}(x) = \begin{cases} \pi(1+x^2)\cos(x)x^2\mathrm{e}^{-x} & x \geq 0\,, \\ 0\,, & x < 0\,. \end{cases} \tag{25}$$

$$f_{\hat{\mathbf{x}}}^{(4)}(x) = \frac{1}{\pi}\frac{1}{1+x^2} \quad \forall x\,. \tag{26}$$

- Which is the best way to split $g(x) = G(x)/f_{\hat{\mathbf{x}}}(x)$? Of course, a possible criterion is that the numerical generation of random numbers according to$f_{\hat{\mathbf{x}}}(x)$ turns out to be easy from a practical point of view. But leaving aside this otherwise very reasonable requirement, a sensible condition to choose the splitting is to obtain an algorithm with the minimum statistical error.

- The smallest error is obtained when the sample variance $\hat{\boldsymbol{\sigma}}_M^2[G]$ is minimum, or, equivalently, when the variance as given by

$$\sigma^2[G] = \int G(x)^2 f_{\hat{\mathbf{x}}}(x)dx - \left(\int G(x) f_{\hat{\mathbf{x}}}(x)dx\right)^2 = \int \frac{g(x)^2}{f_{\hat{\mathbf{x}}}(x)}dx - I^2 \tag{27}$$

  is a minimum.

- As I is independent of $f_{\hat{\mathbf{x}}}(x)$, the minimization of $\sigma^2[G]$ is equivalent to the minimization of $\int \frac{g(x)^2}{f_{\hat{\mathbf{x}}}(x)}dx$. The minimization has to be achieved in the space of functions $f_{\hat{\mathbf{x}}}(x)$, which can be considered as probability density

functions, that is, those functions satisfying

$$f_{\hat{\mathbf{x}}}(x) \geq 0\,, \tag{28}$$
$$\int f_{\hat{\mathbf{x}}}(x)\,dx = 1\,. \tag{29}$$

- The optimal solution $f_{\hat{\mathbf{x}}}^{\rm opt}(x)$ can be found by using the method of the Lagrange multipliers, which help us in enforcing the previous constraints. Let us introduce the functional $\mathcal{L}[f_{\hat{\mathbf{x}}}]$ defined as

$$\mathcal{L}[f_{\hat{\mathbf{x}}}] = \int \frac{g(x)^2}{f_{\hat{\mathbf{x}}}(x)} dx + \lambda \int f_{\hat{\mathbf{x}}}(x)\,dx \tag{30}$$

with λ being the Lagrange multiplier needed to take into account the normalization condition above. The minimization of $\mathcal{L}[f_{\hat{\mathbf{x}}}]$ via functional differentiation leads to

$$\left.\frac{\delta \mathcal{L}[f_{\hat{\mathbf{x}}}]}{\delta f_{\hat{\mathbf{x}}}}\right|_{f_{\hat{\mathbf{x}}}^{\rm opt}(x)} = 0 \Rightarrow -\left(\frac{g(x)}{f_{\hat{\mathbf{x}}}^{\rm opt}(x)}\right)^2 + \lambda = 0 \tag{31}$$

or equivalently

$$f_{\hat{\mathbf{x}}}^{\rm opt}(x) = +\lambda^{-1/2}|g(x)| \geq 0\,. \tag{32}$$

The Lagrange multiplier now follows from the normalization condition, so we arrive at

$$f_{\hat{\mathbf{x}}}^{\rm opt}(x) = \frac{|g(x)|}{\int |g(x)| dx}\,. \tag{33}$$

- The corresponding optimal function $G^{\rm opt}(x)$ is

$$G^{\rm opt}(x) = \frac{g(x)}{f_{\hat{\mathbf{x}}}^{\rm opt}(x)} = {\rm sgn}[g(x)] \int |g(y)| dy \tag{34}$$

and the associated minimum variance is

$$\sigma^2[G^{\rm opt}] = \left(\int |g(x)|dx\right)^2 - I^2 = \lambda - I^2\,. \tag{35}$$

- Interestingly, this result indicates that, if $g(x)$ is a nonnegative function, such that $|g(x)| = g(x)$, then the variance of the optimal estimator is 0. In other words, the statistical error is 0 and the sample average is exact (independent of the number of points M used). Where is the trick? If we look at the optimal choice, in the denominator of Eq. (33) appears precisely the integral I. But knowing the value of the integral was the initial goal. If we know it, why should we need a numerical algorithm whose goal is the calculation of the integral?

- However, the idea of importance sampling is to replace the optimal value $f_{\hat{\mathbf{x}}}^{\rm opt}(x)$ by some other function $f_{\hat{\mathbf{x}}}(x)$ close to it (while still keeping the generation of the random variable easy enough).

- For example, let us look at the calculation of the integral in Eq. (17). The optimal choice for this example would be the splitting

$$G^{\rm opt}(x) = \lambda^{1/2}\frac{\cos(x)}{|\cos(x)|} \quad x \geq 0\,, \tag{36}$$

$$f_{\hat{\mathbf{x}}}^{\rm opt}(x) = \begin{cases} \lambda^{-1/2}|\cos(x)|x^2{\rm e}^{-x} & x \geq 0\,, \\ 0\,, \quad x < 0\,, \end{cases} \tag{37}$$

with $\lambda^{1/2} = \int_0^\infty |\cos(x)|x^2{\rm e}^{-x}\,dx$ being the normalization constant. The minimal variance of the optimal choice is $\sigma^2[G^{\rm opt}] = \lambda - I^2 \approx 1.190009$. However, there is no simple way to solve and invert $\int_0^x f_{\hat{\mathbf{x}}}^{\rm opt}(y)dy = u$ and the optimal choice is useless. We could use instead, for example, any of the four splittings given above. Which one would be the most efficient? The one that uses a probability density function $f_{\hat{\mathbf{x}}}(x)$ closer to $f_{\hat{\mathbf{x}}}^{\rm opt}(x)$. Intuitively, it is option number 3, as all it does is to replace $|\cos(x)|$ by an average value 1/2. We can check this out by

computing (analytically) the variance in the four cases using the integrals of Eq. (27), with the result

$$\sigma^2[G^{(1)}] = \frac{148843}{12500} \approx 11.907\,, \tag{38}$$

$$\sigma^2[G^{(2)}] = \frac{6791}{2500} \approx 2.716\,, \tag{39}$$

$$\sigma^2[G^{(3)}] = \frac{787}{500} \approx 1.574\,, \tag{40}$$

$$\sigma^2[G^{(4)}] = -\frac{1}{4} + \frac{27\pi}{16} \approx 5.051\,. \tag{41}$$

Hence, the splitting number 3 is indeed the most efficient, at least from the point of view of the associated variance.

- Summing up, the basic idea of the importance sampling method is to split the integrand as $g(x) = G(x) f_{\hat{\mathbf{x}}}(x)$ using a suitable random variable $\hat{\mathbf{x}}$ with pdf $f_{\hat{\mathbf{x}}}(x)$. It is desirable to choose $f_{\hat{\mathbf{x}}}(x)$ in such a way that (i) it is reasonably simple to generate, and (ii) it gives a small variance. The optimal $f_{\hat{\mathbf{x}}}(x)$ most often does not satisfy condition (i), but we might then look for functions $f_{\hat{\mathbf{x}}}(x)$ that are close enough to the optimal one.

6. Advantages of Monte Carlo integration

- Although there might be examples of one-dimensional integrals in which the Monte Carlo integration could be competitive compared to more traditional methods (like Simpson integration), the truth is that the real power of Monte Carlo integration lies in the calculation of N-dimensional integrals.

- Let us consider, for example, the integral

$$I(N) = \int_{-\infty}^{\infty} dx_1 \ldots \int_{-\infty}^{\infty} dx_N \mathrm{e}^{-(x_1+\ldots+x_N)} J_0(x_1^2 + \ldots + x_N^2)\,, \tag{42}$$

  where $J_0(x)$ is the Bessel function.

- For large N, it would be difficult to write an efficient code implementing, for instance, a Simpson-like algorithm. Let us consider it from the point of view of importance sampling. We split the integrand $g(x_1, \ldots, x_N) = G(x_1, \ldots, x_N) f_{\hat{\mathbf{x}}_1,\ldots,\hat{\mathbf{x}}_N}(x_1, \ldots, x_N)$ with

$$G(x_1, \ldots, x_N) = J_0(x_1^2 + \ldots + x_N^2)\,, \tag{43}$$
$$f_{\hat{\mathbf{x}}_1,\ldots,\hat{\mathbf{x}}_N}(x_1, \ldots, x_N) = \mathrm{e}^{-(x_1+\ldots+x_N)}\,. \tag{44}$$

- The key point is that $f_{\hat{\mathbf{x}}_1,\ldots,\hat{\mathbf{x}}_N}(x_1, \ldots, x_N)$ can be considered as the probability density function of an N-dimensional random variable $\hat{\mathbf{x}} = (\hat{\mathbf{x}}_1, \ldots, \hat{\mathbf{x}}_N)$.

- In fact, as it can be factorized as $f_{\hat{\mathbf{x}}_1,\ldots,\hat{\mathbf{x}}_N}(x_1, \ldots, x_N) = f_{\hat{\mathbf{x}}_1}(x_1) \ldots f_{\hat{\mathbf{x}}_N}(x_N)$ with $f_{\hat{\mathbf{x}}}(x) = \mathrm{e}^{-x}$, the N random variables $\hat{\mathbf{x}}_1, \ldots, \hat{\mathbf{x}}_N$ are independent of each other and follow the same exponential distribution.

- It is very easy to use the method of importance sampling to compute the integral $I(N)$ above: generate M values $\mathbf{x}_1, \ldots, \mathbf{x}_M$ of the N-dimensional vector $\mathbf{x} = (x_1, \ldots, x_N)$; use them to compute the sample mean and

variance of $G(\mathbf{x})$, using the known formulas

$$\hat{\boldsymbol{\mu}}_M[G] = \frac{1}{M}\sum_{i=1}^{M} G(\mathbf{x}_i)\,, \tag{45}$$

$$\hat{\boldsymbol{\sigma}}^2_M[G] = \frac{1}{M}\sum_{i=1}^{M} G(\mathbf{x}_i)^2 - \left(\frac{1}{M}\sum_{i=1}^{M} G(\mathbf{x}_i)\right)^2\,. \tag{46}$$

and approximate the integral by

$$I(N) = \hat{\boldsymbol{\mu}}_M[G] \pm \frac{\hat{\boldsymbol{\sigma}}_M[G]}{\sqrt{M}}\,, \tag{47}$$

- Let us present a simple computer program:

```
program n_dimensional_integral

implicit none
interface
   double precision function ran_exp(a)
      double precision, intent (in) :: a
   end function ran_exp
end interface
double precision :: r,s,x,dbesj0,sx,s2,g0
integer, parameter :: N=4
integer ::M,i,k

M=1000000
r=0.d0
s=0.d0
```

```
do k=1,M
   s2=0.d0
   do i=1,N
      x=ran_exp(1.d0)
      s2=s2+x*x
   enddo
   g0=dbesj0(s2)
   r=r+g0
   s=s+g0*g0
enddo
r=r/M
s=sqrt((s/M-r*r)/M)

write (*,*) "The estimated value of the integral is", r
write (*,*) "The estimated error is", s

end program n_dimensional_integral
```

Here, *dbesj0* is the Bessel function $J_0(x)$ as implemented in *gfortran* and also in the Intel fortran compiler *ifort*. Otherwise, a routine that returns the value of the Bessel function $J_0(x)$ must be provided.

- As an example, we have run this program with $M = 10^6$ samples and it took us a fraction of a second on a desktop computer to obtain the values $I(2) = 0.38596 \pm 0.00049$, $I(3) = 0.20028 \pm 0.00044$, and $I(4) = 0.08920 \pm 0.00036$.

- The complexity of the Monte Carlo algorithm does not increase with N, the number of integration variables. The program runs perfectly for $N = 10$ giving, for $M = 10^8$, $I(10) = 0.002728 \pm 0.000016$ in less than 1 min, whereas it would be extremely costly to get the same accuracy with a deterministic integration algorithm.

7. Remarks on dimensionality

- We have argued above that numerical integration based on the sampling algorithm

$$\int G(\mathbf{x}) f_{\hat{\mathbf{x}}}(\mathbf{x})\, dx = \hat{\boldsymbol{\mu}}_M[G] \pm \frac{\hat{\boldsymbol{\sigma}}_M[G]}{\sqrt{M}} \tag{48}$$

  with

$$\hat{\boldsymbol{\mu}}_M[G] = \frac{1}{M}\sum_{i=1}^{M} G(\mathbf{x}_i)\,, \tag{49}$$

$$\hat{\boldsymbol{\sigma}}^2_M[G] = \frac{1}{M}\sum_{i=1}^{M} G(\mathbf{x}_i)^2 - \left(\frac{1}{M}\sum_{i=1}^{M} G(\mathbf{x}_i)\right)^2 . \tag{50}$$

  where $\mathbf{x}_i$, $i = 1, \ldots, M$ are the values of the random variable $\hat{\mathbf{x}}$ whose pdf is $f_{\hat{\mathbf{x}}}(\mathbf{x})$, can be very competitive if $\mathbf{x} = (x_1, ..., x_N)$ is high dimensional.

- How high is high? How large must N be for this method to be competitive? The exact answer might depend on the specific details of the function $G(\mathbf{x})$ and the difficulty in the generation of the random variables.

- The answer might be $N > 10$ or $N > 5$, but one thing is sure: when N is very large, on the order of thousands or millions of variables involved in the integral, then *there is no alternative to the Monte Carlo sampling*.

- There are many problems that need of the calculation of such high-dimensional integrals, but the typical applications are in the field of statistical mechanics, where ideally one would like to deal with N on the order of the Avogadro number, $N \sim 10^{23}$. This methos is also essential in quantum field theory.

8. Rejection methods

- We have already described a method that allows, in principle, the generation of values of a random variable distributed according to a given probability distribution function $f_{\hat{\mathbf{x}}}(x)$ based on the relation $\hat{\mathbf{x}} = F^{-1}_{\hat{\mathbf{x}}}(\hat{\mathbf{u}})$, with $\hat{\mathbf{u}}$ being a $\hat{\mathbf{U}}(0,1)$ random variable uniformly distributed in the interval $(0,1)$.
- The main problem with this general method is the technical difficulty in finding a good implementation of the inverse cumulative distribution function $F^{-1}_{\hat{\mathbf{x}}}$. A possibility is to solve numerically the equation $F_{\hat{\mathbf{x}}}(x) - u = 0$ using e.g. the Newton-Raphson method, but this method can be extremely slow for mildly complex problems, and moreover there is no guarantee that the algorithm converges to the right solution.
- On the other hand, the so-called *rejection methods* are best suited to the generation of high (and not so high) dimensional sets of random variables.

8.1. A simple example

- Rejection methods are based on a simple idea: propose a value of the random variable $\hat{\mathbf{x}}$ we want to sample, and then decide whether we keep this proposed value or not.
- For the sake of clarity, we analyze first an example with only one variable. Let us consider a random variable $\hat{\mathbf{x}}$ whose pdf is

$$f_{\hat{\mathbf{x}}}(x) = \begin{cases} C\, \exp(-\frac{x^2}{2}) & -1 \leq x \leq 1\,, \\ 0 \quad x \notin [-1,1]\,. \end{cases} \tag{51}$$

  which is a cut-off Gaussian distribution limited to the interval $(-1,1)$. It is possible to obtain the normalization constant C from

$$1 = C \int_{-1}^{1} dx \exp(-\frac{x^2}{2}) = C\sqrt{2\pi}\mathrm{erf}\left(\frac{1}{\sqrt{2}}\right) \Rightarrow C = \frac{1}{\sqrt{2\pi}\mathrm{erf}\left(\frac{1}{\sqrt{2}}\right)} = 0.584369\ldots \tag{52}$$

However, we do not really need to know the value of the normalization constant C. As we will see, this is one of the nice features that make rejection methods so useful.

- Recall now what a pdf means: $f_{\hat{\mathbf{x}}}(x)$ is a measure of the probability of finding a value of the random variable around x. Hence, if, for example, $f_{\hat{\mathbf{x}}}(x_1) = 2f_{\hat{\mathbf{x}}}(x_2)$, then x_1 is twice as probable to appear as x_2. What we will do is to generate values uniformly in the interval $(-1, 1)$ and accept the proposed value x with a probability proportional to its pdf $f_{\hat{\mathbf{x}}}(x)$. It seems obvious that those values that make $f_{\hat{\mathbf{x}}}(x)$ larger will be accepted more often than those that make $f_{\hat{\mathbf{x}}}(x)$ small, which is, at least qualitatively, what we want to achieve.

- Summing up, we propose the following steps:

  1 Propose a value x sampled from the $\hat{\mathbf{U}}(-1, 1)$, uniform distribution, that is, $x = 2u - 1$ with u sampled from $\hat{\mathbf{U}}(0, 1)$.

  2 Accept that value with a probability $h(x)$, proportional to $f_{\hat{\mathbf{x}}}(x) \propto \exp(-x^2/2)$.

  In step 2, it is important to stress that the acceptance probability $h(x)$ must be a number between 0 and . Recall that the probability *density* function $f_{\hat{\mathbf{x}}}(x)$ is nonnegative and normalized, but otherwise it may take any value between 0 and ∞. The acceptance probability is $\alpha f_{\hat{\mathbf{x}}}(x)$, with α a real number carefully chosen to ensure that the resulting probability $h(x)$ indeed does not exceed the value 1. Otherwise, α is arbitrary.

- In the example we are considering, the resulting acceptance probability is $\alpha C \exp(-x^2/2)$. As $\exp(-x^2/2) \leq 1$ $\forall x$, all we need is that $\alpha C \leq 1$ and we take the simplest (and, as we will see, also the more convenient) choice $\alpha = 1/C$ so the acceptance probability of a proposed value x is $h(x) = \exp(-x^2/2)$.

- How is the acceptance process performed? This is a Bernoulli process: either we accept the proposed value with probability $\exp(-x^2/2)$, or we do not. This decision is made by comparing a new uniform $u = \hat{\mathbf{U}}(0, 1)$ random number with the acceptance probability. If $u < \exp(-x^2/2)$, the proposed value is accepted. Otherwise, if $u \geq \exp(-x^2/2)$, it is rejected. What do we do if we reject the value? Answer: propose a new one.

- This algorithm can be programmed as follows:

```
! Loop for the simple example on the rejection method
do
    x=2.0d0*ran_u()-1.0d0
    if(ran_u() < exp(-0.5d0*x*x)) exit
enddo
```

We see in this simple example the power of the rejection method. The algorithm is really simple and requires only two lines of code. Try to do the same using the method based on the inversion of the cumulative function or $x = F_{\hat{\mathbf{x}}}^{-1}(u)$.

- The only problem with the rejection method could be that the rejection step is taken too often, requiring a large number of trial proposals before a number is actually generated. This will be a real issue for high-dimensional problems.

8.2. General rejection method

- Let us now define what we mean by a general rejection method. It is based on the two following steps:

  1 Propose a value x for the random variable distributed according to a given probability density function $g(x)$.

  2 Accept that value x with a probability $h(x)$. Recall that this probability must satisfy $0 \leq h(x) \leq 1$, $\forall x$.

- For the acceptance step, we consider a Bernoulli random variable $\hat{\mathbf{B}}$ which takes the value $\hat{\mathbf{B}} = 1$ (acceptance) with probability $h(x)$ and the value $\hat{\mathbf{B}} = 0$ (rejection) with probability $1 - h(x)$.

- During the proposal and acceptance/rejection process, we generate a two-dimensional random variable $(\hat{\mathbf{x}}, \hat{\mathbf{B}})$ with joint distribution $f_{\hat{\mathbf{x}},\hat{\mathbf{B}}}(x, n)$.

- A proposal x drawn from the pdf $g(x)$ can give rise to two values of $\hat{\mathbf{B}}$, which implies

$$g(x) = f_{\hat{\mathbf{x}},\hat{\mathbf{B}}}(x, \hat{\mathbf{B}} = 0) + f_{\hat{\mathbf{x}},\hat{\mathbf{B}}}(x, \hat{\mathbf{B}} = 1)\,. \tag{53}$$

- We are interested in those values of x that result after acceptance. They are distributed according to $f_{\hat{\mathbf{x}}}(x|\hat{\mathbf{B}} = 1)$, which we want to be identical to the given $f_{\hat{\mathbf{x}}}(x)$ we wish to sample. According to the definition of conditional pdf, this distribution is given by

$$f_{\hat{\mathbf{x}}}(x|\hat{\mathbf{B}} = 1) = \frac{f_{\hat{\mathbf{x}},\hat{\mathbf{B}}}(x, \hat{\mathbf{B}} = 1)}{\mathrm{Prob}(\hat{\mathbf{B}} = 1)} = \frac{f_{\hat{\mathbf{x}},\hat{\mathbf{B}}}(x, \hat{\mathbf{B}} = 1)}{\int_{-\infty}^{\infty} f_{\hat{\mathbf{x}},\hat{\mathbf{B}}}(x, \hat{\mathbf{B}} = 1)\, dx}\,. \tag{54}$$

- Using the rules of conditional probabilities, we have now for the acceptance probability $h(x)$

$$h(x) = \mathrm{Prob}(\hat{\mathbf{B}} = 1|x) = \frac{f_{\hat{\mathbf{x}},\hat{\mathbf{B}}}(x, \hat{\mathbf{B}} = 1)}{f_{\hat{\mathbf{x}},\hat{\mathbf{B}}}(x, \hat{\mathbf{B}} = 0) + f_{\hat{\mathbf{x}},\hat{\mathbf{B}}}(x, \hat{\mathbf{B}} = 1)} = \frac{f_{\hat{\mathbf{x}},\hat{\mathbf{B}}}(x, \hat{\mathbf{B}} = 1)}{g(x)}\,. \tag{55}$$

  and therefore $f_{\hat{\mathbf{x}},\hat{\mathbf{B}}}(x, \hat{\mathbf{B}} = 1) = h(x)g(x)$.

- Using this in Eq. (54) we arrive at

$$f_{\hat{\mathbf{x}}}(x) = f_{\hat{\mathbf{x}}}(x|\hat{\mathbf{B}} = 1) = \frac{h(x)g(x)}{\int_{-\infty}^{\infty} h(x)g(x)\, dx}\,. \tag{56}$$

  as the pdf resulting of the double proposal/acceptance process. As it should be, this pdf is properly normalized.

- Summing up, to sample some $f_{\hat{\mathbf{x}}}(x)$ we need to find two functions $g(x)$ (sampling pdf) and $h(x)$ (acceptance probability) and such that $f_{\hat{\mathbf{x}}}(x)$ can be split as $f_{\hat{\mathbf{x}}}(x) = Ag(x)h(x)$, where A is a constant. Note that this implies

$$g(x)h(x)f_{\hat{\mathbf{x}}}(y) = g(y)h(y)f_{\hat{\mathbf{x}}}(x)\,, \qquad \forall x, y \tag{57}$$

  which is the so-called detailed balance condition[1].

8.3. Efficiency of the rejection method

- The efficiency of a rejection method depends on the average probability of acceptance ϵ. If ϵ is very small, then we need a large average number of trials to generate a single value of $\hat{\mathbf{x}}$.

- The average acceptance probability can be computed using

$$\epsilon = \int_{-\infty}^{\infty} P(\hat{\mathbf{B}} = 1|x)g(x)\,dx = \int_{-\infty}^{\infty} h(x)g(x)\,dx = \langle h \rangle\,. \tag{58}$$

  This implies that, given a fixed $g(x)$, the efficiency increases with increasing $h(x)$ and, hence, it is convenient to take $h(x)$ as large as possible, always keeping the condition $0 \leq h(x) \leq 1$.

- The average acceptance probability is related to the normalization constant C. If we know C, we can easily obtain ϵ. If we did not know C, we could derive it from a numerical estimation of ϵ.

- Example: We want to generate values of a random variable $\hat{\mathbf{x}}$ whose pdf is

$$f_{\hat{\mathbf{x}}}(x) = C \exp\left(-\frac{x^2}{2} - x^4\right)\,, \qquad x \in (-\infty, \infty) \tag{59}$$

† The name is not by accident. As the generation of the different values of the random variable is done *independently of each other*, it can be thought, formally, as a homogeneous Markov chain in which the proposal pdf $f(x|y) = h(x)g(x)/\int_{-\infty}^{\infty} h(x)g(x)\,dx$ is independent of y. The stationarity condition $f_{\hat{\mathbf{x}}}^{\rm st}(x) = \int_{-\infty}^{\infty} f(x|y)\, f_{\hat{\mathbf{x}}}^{\rm st}(y)\,dy$ of chapter 1 then leads trivially to the detailed balance condition.

where C is the normalization constant given by $C = 2\sqrt{2}e^{-1/32}K_{1/4}(1/32) = 0.643162\ldots$ We need to split $f_{\hat{\mathbf{x}}}(x) \propto g(x)h(x)$, $g(x)$ being a pdf we know how to sample numerically and $0 \leq h(x) \leq 1$. The obvious choice is

$$g(x) = \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{x^2}{2}\right), \qquad x \in (-\infty, \infty), \tag{60}$$

$$h(x) = \exp\left(-x^4\right). \tag{61}$$

So we propose a value x drawn from a $\hat{\mathbf{G}}(0,1)$ Gaussian distribution and accept it with probability $\exp(-x^4)$. The algorithm can be programmed as follows:

```
! Sampling for exp(-x**2/2 - x**4)
do
   x=ran_g()
   if (ran_u() < exp(-x**4) ) exit
enddo
```

The average acceptance probability is

$$\epsilon = \int_{-\infty}^{\infty} dx\, g(x)h(x) = \frac{1}{C\sqrt{2\pi}} \approx 0.620283\ldots \tag{62}$$

- Example: Let us now consider an n-dimensional problem. Imagine we need to compute the integral

$$I(n) = \frac{\int_{-\infty}^{\infty} dx_1 \ldots \int_{-\infty}^{\infty} dx_n\, \mathrm{e}^{-(x_1^2+\ldots+x_n^2)/2-(x_1\cdots x_n)^4} \cos(x_1\cdots x_n)}{\int_{-\infty}^{\infty} dx_1 \ldots \int_{-\infty}^{\infty} dx_n\, \mathrm{e}^{-(x_1^2+\ldots+x_n^2)/2-(x_1\cdots x_n)^4}} \tag{63}$$

Obviously, we interpret this integral as the average value $\langle G(x_1,\ldots,x_n)\rangle$ with respect to the pdf $f_{\hat{\mathbf{x}}_1,\ldots,\hat{\mathbf{x}}_n}(x_1,\ldots,x_n)$, with

$$G(x_1,\ldots,x_n) = \cos(x_1\cdots x_n)\,, \tag{64}$$

$$f_{\hat{\mathbf{x}}_1,\ldots,\hat{\mathbf{x}}_n}(x_1,\ldots,x_n) = C\,\mathrm{e}^{-(x_1^2+\ldots+x_n^2)/2-(x_1\cdots x_n)^4}\,, \tag{65}$$

where C is the normalization constant of the pdf. To generate values of the n-dimensional random variable $(\hat{\mathbf{x}}_1,\ldots,\hat{\mathbf{x}}_n)$, we split the pdf as

$$f_{\hat{\mathbf{x}}_1,\ldots,\hat{\mathbf{x}}_n}(x_1,\ldots,x_n) \propto g(x_1,\ldots,x_n)\,h(x_1,\ldots,x_n)\,, \tag{66}$$

$$g(x_1,\ldots,x_n) = \prod_{i=1}^{n}\left[\frac{1}{\sqrt{2\pi}}\mathrm{e}^{-x_i^2/2}\right] \tag{67}$$

$$h(x_1,\ldots,x_n) = \mathrm{e}^{-(x_1\cdots x_n)^4}\,. \tag{68}$$

Note that, indeed, $0 \le h(x_1,\ldots,x_n) \le 1$. The proposal $g(x_1,\ldots,x_n)$ corresponds to n independent Gaussian $\hat{\mathbf{G}}(0,1)$ distributions. The program to implement the rejection algorithm to compute this integral is as follows:

```
! Rejection method for multidimensional integral
program rejection

implicit none
double precision r,s,b,p,g,ran_g,ran_u
double precision, parameter :: pi=3.1415926535897 93d0
integer, parameter :: n=2
integer M,i,j,ij,na
```

```
M=10000000
r=0.d0
s=0.d0
na=0
do ij=1,M
   do
      b=1.0d0
      do i=1,n
         b=b*dran_g()  ! Product of n random Gaussian variables
      enddo
      na=na+1
      if (dran_u() < exp(-b**4)) exit
   enddo
   g=cos(b)
   r=r+g
   s=s+g*g
enddo

p=dble(M)/na
r=r/M
s=sqrt((s/M-r*r)/M)
write(6,*) n,r,s,p

end program rejection
```

- Note that, besides the value of the integral and its error, we also write the average acceptance probability. This program runs without problems for any arbitrary value of n. Some results are as follows: $I(n=2) = 0.922467 \pm 0.000037$, $I(n=10) = 0.993885 \pm 0.000011$, $I(n=40) = 0.99999666 \pm 0.00000024$, which strongly suggest $\lim_{n\to\infty} I(n) = 1$, a result that can be confirmed analytically. The average acceptance probability increases with n as $\epsilon(n=2) = 0.748$, $\epsilon(n=10) = 0.977$, and $\epsilon(n=40) = 0.999987$. This increase of ϵ with n, unfortunately, is not a typical feature of rejection methods. On the contrary, usually the average acceptance probability decreases with n and becomes very small for the values of n of interest.

- Example: Consider the n-dimensional random variable $(\hat{\mathbf{x}}_1, \ldots, \hat{\mathbf{x}}_n)$ uniformly distributed inside an az$n$-dimensional sphere of radius 1 whose pdf is

$$f_{\hat{\mathbf{x}}_1,\ldots,\hat{\mathbf{x}}_n}(x_1,\ldots,x_n) = \begin{cases} 1/V_n & x_1^2 + \ldots + x_n^2 \leq 1\,, \\ 0 & \text{otherwise}\,, \end{cases} \tag{69}$$

  where $V_n = \pi^{\frac{n}{2}}/\Gamma(\frac{n}{2}+1)$ is the volume of the n-hypersphere of radius 1. If we use a simple rejection method where we propose a value for the i-th coordinate x_i taken independently from a uniform distribution in $(-1,1)$, and accept the proposed vector $(x_1,\ldots,x_n)$ only if it satisfies $x_1^2 + \ldots + x_n^2 \leq 1$, then the average acceptance probability is $\epsilon(n) = V_n/2^n$, as 2^n is the volume of the n-dimensional hypercube $[-1,1]^n$. This takes values $\epsilon(n=2) = 0.7854\ldots$, $\epsilon(n=10) = 2.49 \times 10^{-3}$, and $\epsilon(n=100) = 1.87 \times 10^{-70}$. We see that, for $n = 100$, we need to make, on average, 10^{70} proposals to accept one! This is clearly not useful. The method of rejection needs to be modified in order to be able to deal with distributions like this one.

9. Rejection with repetition

- Simple rejection methods for high-dimensional probability distributions have, in general, a low acceptance probability – so low, in fact, that it renders them unfeasible to sample most distributions of interest.
- In this section we develop a variant of the rejection method that allows us, finally, to sample almost any probability distribution using simple rules. The method is that of rejection with repetition.
- Of course, there is no free lunch, and there is an important price to pay when using these methods: as they produce correlated (as opposed to independent) values of the random variables, it turns out that the error of the estimate of the integral or sum increases with N, the number of variables.
- We will see, though, that the increase of the error with N is moderate as compared to the decrease of the average acceptance probability if a simple rejection method is used.

9.1. A simple case of rejection with repetition

- Our goal here is, again, to devise methods to generate values x_k that can be used in a sampling method in order to compute a given integral

$$I = \int dx\, g(x) = \int dx\, f_{\hat{\mathbf{x}}}(x) G(x)\,, \tag{70}$$

using the estimation $I = \hat{\boldsymbol{\mu}}_M[G] \pm \dfrac{\hat{\boldsymbol{\sigma}}_M[G]}{\sqrt{M}}$, with the sample mean and variance defined as

$$\hat{\boldsymbol{\mu}}_M[G] = \frac{1}{M}\sum_{i=1}^{M} G(x_i)\,, \tag{71}$$

$$\hat{\boldsymbol{\sigma}}^2_M[G] = \frac{1}{M}\sum_{i=1}^{M} G(x_i)^2 - \left(\frac{1}{M}\sum_{i=1}^{M} G(x_i)\right)^2\,. \tag{72}$$

For that purpose, we need to generate values x_k of a random variable distributed according to the given pdf $f_{\hat{\mathbf{x}}}(x)$.

- We propose now a modification of the rejection method, which we name *rejection with repetition*. The modification is such that every time a proposal x_k is rejected, instead of proposing a new value, we simply return the value that had been generated in the previous step, $x_k = x_{k-1}$.

- Note that, just in case the trial at the first initial step fails, the value of x_0 should have been initialized following some distribution $f_{\hat{\mathbf{x}}_0}(x)$ (e.g. a Gaussian distribution), or simply $x_0 = a$ with a some initial value. This corresponds to taking $f_{\hat{\mathbf{x}}_0}(x) = \delta(x - a)$.

- Obviously, the output of this process will be a sequence of correlated (i.e. not independent) numbers: in fact, some of them are equal along the sequence! The question is: can we still use this list of numbers in the estimation of the integral above?. Put in another way, is it true that the series of numbers x_k are still distributed according to $f_{\hat{\mathbf{x}}}(x)$?

$$f_{\hat{\mathbf{x}}}(x) = f_{\hat{\mathbf{x}}}(x|\hat{\mathbf{B}} = 1) = \frac{h(x)g(x)}{\int_{-\infty}^{\infty} h(x)g(x)\,dx}\,. \tag{73}$$

- As one value x_{k+1} might be equal to the previous value x_k, we need to find the probability distribution that governs the outcome of the k^{th} step of this process.

- Let $\hat{\mathbf{x}}_k$ be the random variable obtained during the k^{th} step. Its precise form will depend on the outcome of the acceptance process. As there are two options, namely accept or reject, we consider the acceptance step to be a Bernoulli variable $\hat{\mathbf{y}}$ which can take two possible values $\hat{\mathbf{y}} = 1$, accept, and $\hat{\mathbf{y}} = 0$, reject. Recalling that

$$f_{\hat{\mathbf{x}}}(x) = \int_{-\infty}^{\infty} f_{\hat{\mathbf{x}}}(x|\hat{\mathbf{y}} = y)\, f_{\hat{\mathbf{y}}}(y)\, dy = \sum_i f_{\hat{\mathbf{x}}}(x|\hat{\mathbf{y}} = y_i)\, P(\hat{\mathbf{y}} = y_i)\,, \tag{74}$$

where we have used the discrete character of the random variable $\hat{\mathbf{y}}$ in the last equality, we thus find

$$f_{\hat{\mathbf{x}}_k}(x) = f_{\hat{\mathbf{x}}_k}(x|\text{accept})P(\text{accept}) + f_{\hat{\mathbf{x}}_k}(x|\text{reject})P(\text{reject})\,, \tag{75}$$

with $P(\text{reject}) = 1 - P(\text{accept})$ being the probabilities of acceptance and rejection, respectively, at step k.

- Now we can use Bayes theorem $f_{\hat{\mathbf{x}}}(x|\hat{\mathbf{y}} = y)f_{\hat{\mathbf{y}}}(y) = f_{\hat{\mathbf{y}}}(y|\hat{\mathbf{x}} = x)f_{\hat{\mathbf{x}}}(x)$ in its discrete form to modify the first term of this sum above

$$f_{\hat{\mathbf{x}}_k}(x) = P(\text{accept}|x)g(x) + f_{\hat{\mathbf{x}}_k}(x|\text{reject})P(\text{reject})\,, \tag{76}$$

where we have used that $g(x)$ is the pdf of proposing the value x at step k. The probability of accepting a given value x is $P(\text{accept}|x) = h(x)$.

- If we reject, the pdf at step k is the same as the valid one at step $k-1$, that is, $f_{\hat{\mathbf{x}}_k}(x|\text{reject}) = f_{\hat{\mathbf{x}}_{k-1}}(x)$. Finally, the acceptance probability is given by $\epsilon = \int_{-\infty}^{\infty} h(x)g(x)\,dx$, so $P(\text{reject}) = 1 - \epsilon$. This leads to

$$f_{\hat{\mathbf{x}}_k}(x) = h(x)g(x) + f_{\hat{\mathbf{x}}_{k-1}}(x)\,(1-\epsilon) \tag{77}$$

This recurrence gives $f_{\hat{\mathbf{x}}_k}(x)$ in terms of $f_{\hat{\mathbf{x}}_{k-1}}(x)$.

- To solve this recurrence in terms of the initial distribution $f_{\hat{\mathbf{x}}_0}(x)$, we iterate to see

$$\begin{aligned} f_{\hat{\mathbf{x}}_k}(x) &= h(x)g(x) + (1-\epsilon)\left[h(x)g(x) + (1-\epsilon)\,f_{\hat{\mathbf{x}}_{k-2}}(x)\right] = h(x)g(x)\sum_{i=0}^{k-1}(1-\epsilon)^i + (1-\epsilon)^k f_{\hat{\mathbf{x}}_0}(x) \\ &= h(x)g(x)\frac{1-(1-\epsilon)^k}{\epsilon} + (1-\epsilon)^k f_{\hat{\mathbf{x}}_0}(x) = (1-\epsilon)^k\left[f_{\hat{\mathbf{x}}_0}(x) - \frac{1}{\epsilon}h(x)g(x)\right] + \frac{1}{\epsilon}h(x)g(x) \end{aligned} \tag{78}$$

or equivalently

$$f_{\hat{\mathbf{x}}_k}(x) = (1-\epsilon)^k\left[f_{\hat{\mathbf{x}}_0}(x) - \frac{h(x)g(x)}{\int_{-\infty}^{\infty}h(z)g(z)\,dz}\right] + \frac{h(x)g(x)}{\int_{-\infty}^{\infty}h(z)g(z)\,dz} \tag{79}$$

- Finally, recalling that the pdf $f_{\hat{\mathbf{x}}}(x)$ we want to sample can be written as

$$f_{\hat{\mathbf{x}}}(x) = \frac{h(x)g(x)}{\int_{-\infty}^{\infty} h(x)g(x)\,dx}\,, \tag{80}$$

we arrive at

$$f_{\hat{\mathbf{x}}_k}(x) = (1-\epsilon)^k \Big[f_{\hat{\mathbf{x}}_0}(x) - f_{\hat{\mathbf{x}}}(x) \Big] + f_{\hat{\mathbf{x}}}(x)\,. \tag{81}$$

- Therefore, the answer to the question: are all the x_k values distributed according to $f_{\hat{\mathbf{x}}}(x)$? is, generally, **NO**, as we cannot conclude from this formula that $f_{\hat{\mathbf{x}}_k}(x) = f_{\hat{\mathbf{x}}}(x)$, $\forall k$.

- However, if the initial numbers are already distributed according to $f_{\hat{\mathbf{x}}}(x)$, that is, if $f_{\hat{\mathbf{x}}_0}(x) = f_{\hat{\mathbf{x}}}(x)$, then we find $f_{\hat{\mathbf{x}}_k}(x) = f_{\hat{\mathbf{x}}}(x)$ $\forall k$, and the numbers we obtain by this procedure are indeed distributed according to the desired distribution.

- If, on the other hand, the initial pdf $f_{\hat{\mathbf{x}}_0}(x)$ is not equal to the one we want to sample, the situation is not so bad either. According to Eq. (81), the difference between the actual distribution at step k, $f_{\hat{\mathbf{x}}_k}(x)$, and the one we want to sample, $f_{\hat{\mathbf{x}}}(x)$, monotonically decreases with k. In fact, as $0 < \epsilon \leq 1$, we have the exact result, independent of the initial distribution $f_{\hat{\mathbf{x}}_0}(x)$

$$\lim_{k\to\infty} f_{\hat{\mathbf{x}}_k}(x) = f_{\hat{\mathbf{x}}}(x)\,. \tag{82}$$

- Imagine $\epsilon = 0.5$. After $k = 10$ steps, the factor $(1-\epsilon)^k \approx 10^{-3}$. If $k = 100$, $(1-\epsilon)^k \approx 10^{-30}$, an absolutely negligible contribution in most cases. Even for a low acceptance probability $\epsilon = 10^{-2}$, it takes about $k = 2300$ steps to have $(1-\epsilon)^k \approx 10^{-10}$.

- Therefore, in order to ensure that the produced x_k values satisfy the desired distribution, we need to discard some steps at the beginning. This is the process called *thermalization*. How many steps M_0 we should discard depends on the distance between the initial and the desired distributions and the average acceptance probability ϵ.

- Note that the process of generation of the numbers x_k can be understood in terms of a Markov chain, since the result at step k depends only on the distribution at step $k-1$.

- It is easy to write the above recurrence relation in the form of the standard recurrence equation for a homogeneous Markov chain

$$f_{\hat{\mathbf{x}}_k}(x) = \int_{-\infty}^{\infty} f(x|y)\, f_{\hat{\mathbf{x}}_{k-1}}(y)\, dy\,, \tag{83}$$

with a transition probability density function

$$f(x|y) = h(x)g(x) + \left[1 - \int_{-\infty}^{\infty} h(z)g(z)\, dz\right]\delta(x-y) \tag{84}$$

The two terms of the sum on the right-hand side correspond to the two possibilities, namely acceptance of a proposed value or its rejection.

9.2. Statistical errors

- Remember the approximation consisting of replacing an average value by the sample mean

$$I = \int_{-\infty}^{\infty} G(x) f_{\hat{\mathbf{x}}}(x)\, dx = \hat{\boldsymbol{\mu}}_M[G] \pm \hat{\boldsymbol{\sigma}}_M[\hat{\boldsymbol{\mu}}_M] \tag{85}$$

  with the sample average

$$\hat{\boldsymbol{\mu}}_M[G] = \frac{1}{M} \sum_{k=1}^{M} G(x_k) \tag{86}$$

- As usual, as dictated by Chebyshev's theorem, the error of the sample mean is measured by its standard deviation. However, we cannot use now the relation

$$\sigma^2[\hat{\boldsymbol{\mu}}_M] = \frac{\sigma^2[G]}{M} \approx \frac{\hat{\boldsymbol{\sigma}}_M^2[G]}{M} \tag{87}$$

  because the derivation of this formula assumes that all contributions to the sum in $\hat{\boldsymbol{\mu}}_M[G]$ are independent of each other and we know this is not the case for the rejection with repetition method.

- Let us get now a useful expression for the error in the case of correlated values. We assume that we have thermalized conveniently the algorithm by discarding the necessary steps at the beginning and the Markov chain is in the steady state and, consequently, all variables x_k are distributed according to the same pdf $f_{\hat{\mathbf{x}}}(x)$.

- Using the definition of $\hat{\boldsymbol{\mu}}_M[G]$ above and writing G_k instead of $G(x_k)$ for brevity in notation, the variance of the sample mean can be written as

$$\sigma^2[\hat{\boldsymbol{\mu}}_M] = \langle (\hat{\boldsymbol{\mu}}_M - I)^2 \rangle = \left\langle \left[\frac{1}{M} \sum_{k=1}^{M} (G_k - I) \right]^2 \right\rangle = \frac{1}{M^2} \sum_{k=1}^{M} \sum_{j=1}^{M} \langle (G_k - I)(G_j - I) \rangle\,. \tag{88}$$

- In this double sum, we consider separately the terms with $k = j$, $k > j$, and $j > k$. Due to the symmetry between k and j, the last two contributions are equal and we can write

$$\sigma^2[\hat{\boldsymbol{\mu}}_M] = \langle(\hat{\boldsymbol{\mu}}_M - I)^2\rangle = \frac{1}{M^2}\sum_{k=1}^{M}\langle(G_k - I)^2\rangle + \frac{2}{M^2}\sum_{k=1}^{M-1}\sum_{j=k+1}^{M}\langle(G_k - I)(G_j - I)\rangle\,. \tag{89}$$

Expanding the product in the second sum, and using that $\langle G_i\rangle = I\ \forall i$, we get

$$\begin{aligned}\sigma^2[\hat{\boldsymbol{\mu}}_M] = \langle(\hat{\boldsymbol{\mu}}_M - I)^2\rangle &= \frac{1}{M^2}\sum_{k=1}^{M}\left[\langle G_k^2\rangle - I^2\right] + \frac{2}{M^2}\sum_{k=1}^{M-1}\sum_{j=k+1}^{M}\left[\langle G_k G_j\rangle - I^2\right] \\ &= \frac{\sigma^2[G]}{M}\left[1 + \frac{2}{M}\sum_{k=1}^{M-1}\sum_{j=k+1}^{M}\rho_G(k,j)\right],\end{aligned} \tag{90}$$

where we have used that all G_k yield the same variance $\sigma^2[G] = \langle G_k\rangle^2 - I^2$, and the definition of the normalized correlation between G_k and G_j

$$\rho_G(k,j) = \frac{\langle G_k G_j\rangle - I^2}{\sigma^2[G]}\,. \tag{91}$$

- As we are in the steady state of a homogeneous Markov chain, the correlation function depends only on the difference $j - k \equiv i$, so $G(k,j) = G(j-k) = G(i)$, and we can write

$$\sigma^2[\hat{\boldsymbol{\mu}}_M] = \frac{\sigma^2[G]}{M}\left[1 + \frac{2}{M}\sum_{k=1}^{M-1}\sum_{i=1}^{M-k}\rho_G(i)\right], \tag{92}$$

Exchanging the summation order, we get

$$\sum_{k=1}^{M-1}\sum_{i=1}^{M-k}\rho_G(i) = \sum_{i=1}^{M-1}\sum_{k=1}^{M-i}\rho_G(i) = \sum_{i=1}^{M-1}(M-i)\rho_G(i) \tag{93}$$

since the sum over k could be performed as $G(i)$ does not depend on k. We have now the final result

$$\sigma^2[\hat{\boldsymbol{\mu}}_M] = \frac{\sigma^2[G]}{M}\left[2\sum_{i=1}^{M-1}\left(1-\frac{i}{M}\right)\rho_G(i)+1\right]. \tag{94}$$

- Let us now define the autocorrelation time τ_G as

$$\tau_G = \sum_{i=1}^{M-1}\left(1-\frac{i}{M}\right)\rho_G(i)\,, \tag{95}$$

We hence obtain the final expression for the variance of the sample mean, again replacing the true variance $\sigma^2[G]$ by the sample variance $\hat{\boldsymbol{\sigma}}^2_M[G]$

$$\sigma^2[\hat{\boldsymbol{\mu}}_M] = \frac{\hat{\boldsymbol{\sigma}}^2_M}{M}(2\tau_G+1)\,. \tag{96}$$

- Let us stress that the only assumption in this definition is that the Markov chain is in the steady state where variables at each step k have the same distribution and the correlations between steps k and j depend only on the difference $j-k$.

- Note that if all variables $\hat{\mathbf{x}}_k$ were independent, we would have $\rho_G(i) = \delta_{i,0}$, the autocorrelation time would be 0, and the variance would simplify to the known result $\sigma^2[\hat{\boldsymbol{\mu}}_M] = \sigma^2[G]/M$.

- In general, the autocorrelation function $\rho_G(i)$ decays with i and tends to 0 as $i \to \infty$ (usually, but not always, the decay is exponential).

- The autocorrelation time τ_G is a measure of the decay of the correlation function. It can be defined, loosely, as the number of steps that are needed for the correlation to decay significantly from the initial value $\rho_G(0) = 1$.

A more precise definition is simply Eq. (95). For instance, if the decay were truly exponential $\rho_G(i) = [\rho_G(1)]^i$, with $\rho_G(1) < 1$, then according to the definition (95) the correlation time is

$$\tau_G = \sum_{i=1}^{M-1} \left(1 - \frac{i}{M}\right) [\rho_G(1)]^i = \frac{\rho_G(1)}{1-\rho_G(1)} - \frac{\rho_G(1)[1-\rho_G(1)^M]}{[1-\rho_G(1)]^2 M} \tag{97}$$

which can be reduced in the limit of large M to

$$\tau_G = \frac{\rho_G(1)}{1-\rho_G(1)} \tag{98}$$

Indeed, in most cases, the correlation function decays rapidly and, in the limit of large M, we can neglect the factor i/M in the definition of τ_G and write $\tau_G = \Sigma_{i=1}^{M-1} \rho_G(i)$.

- For the rejection with repetition method explained above, it is possible to compute the correlation function. If the acceptance probability is ϵ, then either $G_{i+k} = G_k$ with probability $(1-\epsilon)^i$ (all i steps in going from k to $i+k$ have been rejections) or G_{i+k} is independent of G_k with probability $1-(1-\epsilon)^i$. This gives

$$\langle G_k G_{i+k} \rangle = (1-\epsilon)^i \langle G_k^2 \rangle + [1-(1-\epsilon)^i] \langle G_k \rangle \langle G_{i+k} \rangle \tag{99}$$

which, replaced in the definition of $\rho_G(i)$ above, leads to $\rho_G(i) = (1-\epsilon)^i$, which is an exponential decay.

- The autocorrelation time is independent of the function G and is given by $\tau_G = \frac{1-\epsilon}{\epsilon}$, and then $2\tau_G + 1 = \frac{2-\epsilon}{\epsilon}$ and the correct expression for the estimator and its error is

$$I = \hat{\boldsymbol{\mu}}_M[G] \pm \frac{\hat{\boldsymbol{\sigma}}_M[G]}{\sqrt{M}} \sqrt{\frac{2-\epsilon}{\epsilon}} \,. \tag{100}$$

- Note that this is equivalent to using an effective number $M_{\text{eff}} = M\frac{\epsilon}{2-\epsilon} \leq M$ of *independent* contributions to the estimator. Recall that if we wanted to have the same error using a rejection method *without repetition* (i.e., trying again if the proposed value is rejected), then we would need an average number M_{eff}/ϵ of proposals. As $\frac{2-\epsilon}{\epsilon} \geq \epsilon$, it turns out that the method of rejection with repetition needs to generate more random numbers in order to yield the same statistical error. However, we will see below that this method is the one that can be generalized to high-dimensional distributions while keeping a reasonable acceptance probability.

10. Dynamical methods

- Rejection methods, irrespective of whether we use repetition or not, can have a very low acceptance probability. This is especially true if the proposal probability $g(x)$ is very different from $f_{\hat{\mathbf{x}}}(x)$ and does not reflect the strong variations in value that $f_{\hat{\mathbf{x}}}(x)$ might have, that is, the ratio $f_{\hat{\mathbf{x}}}(x_{\max})/f_{\hat{\mathbf{x}}}(x_{\min})$. This usually occurs when we have no clue on how the pdf $f_{\hat{\mathbf{x}}}(x)$ we want to sample looks like, a problem particularly severe and common for distributions taking values in a high N-dimensional space, $\mathbf{x} = (x_1, \ldots, x_N)$.

- Example: Consider the distribution

$$f_{\hat{\mathbf{x}}}(x_1, \ldots, x_N) = \begin{cases} C \, \exp\left(-\Sigma_{i=1,j=i}^{N} x_i x_j\right) , & \text{if } x_i \in (-1,1) \, , \forall i \, , \\ 0 \, , & \text{if } x_i \notin (-1,1) \, . \end{cases} \tag{101}$$

  One might be tempted to use a rejection method in which the proposal are independent values $x = (x_1, \ldots, x_N)$, each x_i drawn from $\hat{\mathbf{U}}(-1,1)$ or uniformly distributed in the interval $(-1,1)$. This proposal would be accepted with a probability $h(x_1, \ldots, x_N)$ proportional to $f_{\hat{\mathbf{x}}}(x_1, \ldots, x_N)$. We note that the maximum value of the distribution $f_{\hat{\mathbf{x}}}(x_1, \ldots, x_N)$ occurs for $x_i = 0$, $\forall i$, and there are two equivalent minima for $x_i = 1$, $\forall i$, and $x_i = -1$, $\forall i$. As there are a total of $N(N+1)/2$ terms in the sums of the exponential defining $f_{\hat{\mathbf{x}}}(x)$, the ratio between these two extreme values is $f_{\hat{\mathbf{x}}}(x_{\max})/f_{\hat{\mathbf{x}}}(x_{\min}) = \mathrm{e}^{N(N+1)/2}$. The optimal acceptance probability would be $h(x) = \exp\left(-\Sigma_{i=1,j=i}^{N} x_i x_j\right)$, whose minimum and maximum values are $\mathrm{e}^{-N(N+1)/2}$ and 1, respectively. When N is large, it turns out that most of the proposed values will belong to a region in which the acceptance probability is very small. Hence, the average acceptance probability is very small.

- Indeed, it can be shown that the average acceptance probability decreases exponentially with N: for $N = 100$ we have $\epsilon \approx 3 \times 10^{-8}$, which means that we have to propose, on average, around 30 million numbers, in order to accept one, which is a very inefficient procedure!!.

- How can we avoid proposing values that have a very low acceptance probability? The idea of the so-called dynamical methods is to use the information gathered at previous proposals to make a new proposal. In this way, when by repeated trials we reach the region in which the pdf is high, our proposals will not tend to abandon this region.

- Let us explain these ideas using again a simpler example than the N-dimensional distribution above. Consider the cutoff Gaussian probability distribution

$$f_{\hat{\mathbf{x}}}(x) = C \exp(-x^2/2\sigma^2)\,, \quad x \in (-1,1) \tag{102}$$

  for small values of the parameter σ. If we propose *blindly* values uniformly distributed according to a uniform distribution $g(x)$ in the interval $(-1,1)$ and accept this value with a probability $h(x) = \exp(-x^2/2\sigma^2)$, most of the times we are proposing values that have a very small acceptance probability. However, when we reach a value near $x = 0$ and note that there is a high acceptance probability, what we should do is that the new proposed value is again close to 0, as we know now (and did not know before) that $x = 0$ is the region of high probability.

- If x_k is the value obtained at the kth step, then the new proposal at step $k+1$ could be, for example, a number x_{k+1} chosen uniformly in the interval $x_{k+1} \in (x_k - \delta, x_k + \delta)$, with δ a number comparable to σ, so we do not leave the region of large probability. This is the basic principle behind dynamical methods.

- Summing up, in the static methods explained in previous sections, the new value for the random variable was chosen each time independently of previous values, whereas in the dynamical methods the proposal (and the acceptance probability as we will see) depends on the previous values of the random variable.

- In other words, in the static methods, if we denote by x_n the value of the random variable generated at the nth step, then both the proposal $g(x)$ and the acceptance $h(x)$ are independent of the previous values $(x_1, \ldots, x_{n-1})$.

As a consequence, the pdf $f_{\hat{\mathbf{x}}_n}(x)$ at step n is independent of the previous steps. More precisely, in the static methods

$$f_{\hat{\mathbf{x}}_n}(x_n|x_0, x_1, \ldots, x_{n-1}) = f_{\hat{\mathbf{x}}_n}(x_n)\,, \quad n = 0, 1, 2, \ldots \tag{103}$$

and all random variables x_n are independently distributed according to the same pdf $f_{\hat{\mathbf{x}}}(x)$.

- In a dynamical method, both the proposal and the acceptance probabilities depend on the previous results. As a consequence, the pdf of the random variable x_n obtained in the nth proposal/acceptance step depends on the previous values of the random variables $x_0, x_1, \ldots, x_{n-1}$. The simplest way to implement this dependence is via the latest result, that is, take

$$f_{\hat{\mathbf{x}}_n}(x_n|x_0, x_1, \ldots, x_{n-1}) = f_{\hat{\mathbf{x}}_n}(x_n|x_{n-1})\,, \tag{104}$$

  defining our series of proposal/acceptance steps as a Markov chain. Again, the simplest implementation uses a *homogeneous* Markov chain where the conditional probabilities are independent of the step, or $f_{\hat{\mathbf{x}}_n}(x_n|x_{n-1}) = f(x_n|x_{n-1})$.

- Let us now discuss the details of a dynamical method. We perform a realization of the Markov chain generating values $x_0, x_1, \ldots, x_n, \ldots$ of the random variables. The value x_n is sampled from a distribution $f(x_n|x_{n-1})$. Ideally, we would like all probability distributions $f_{\hat{\mathbf{x}}_n}(x)$ to be equal to the desired pdf $f_{\hat{\mathbf{x}}}(x)$. However, our discussion of the simple dynamical method of rejection with repetition tells us that it might be more reasonable to demand only that the asymptotic distribution reproduces the desired pdf, $\lim_{n\to\infty} f_{\hat{\mathbf{x}}_n}(x) = f_{\hat{\mathbf{x}}}(x)$. Our discussion on Markov chains in the first chapter tells us that a sufficient condition for this to happen is the *detailed balance condition*

$$f(y|x)f_{\hat{\mathbf{x}}}(x) = f(x|y)f_{\hat{\mathbf{x}}}(y)\,. \tag{105}$$

- Usually (but not always), the generation of random variables distributed according to $f(x|y)$ uses a rejection method. This means that we propose a value x from a pdf $g(x|y)$, and then accept it with a probability $h(x|y)$.

As explained, both proposal and acceptance of a number x at step n depend on the value y obtained in the previous step.

- We now obtain the transition probability $f(x|y)$ that corresponds to this proposal/acceptance algorithm in the case of rejection with repetition, that is, if the value x is not accepted, then we keep the previous value y. It will be clear in a moment why, of all possible rejection methods, we adopt this one.

- Note that the derivation of the transition pdf is similar to the explained above, but now we have to take into account that both the proposal pdf $g(x|y)$ and the acceptance probabilities $h(x|y)$ depend on y.

- The transition pdf $f(x|y)$ has two contributions corresponding to the acceptance or not of the proposed value. Given a value y, the overall rejection probability is

$$P(\text{rejection}|y) = 1 - P(\text{acceptance}|y) = 1 - \int h(z|y)\, g(z|y)\, dz \equiv 1 - \epsilon(y)\,, \tag{106}$$

where we have used the definition of the average acceptance probability with the only modification that both the proposal and the acceptance probabilities now depend on the value y. If the proposal x is not accepted, then the transition probability is the Dirac-delta $\delta(x-y)$, as we keep the old value y. This leads to a transition probability

$$f(x|y) = h(x|y)\, g(x|y) + \delta(x-y)\,[1 - \epsilon(y)]\ . \tag{107}$$

- Similarly, we have

$$f(y|x) = h(y|x)\, g(y|x) + \delta(x-y)\,[1 - \epsilon(x)]\ . \tag{108}$$

- Detailed balance: Which functions $h(x|y)$ and $g(x|y)$ can be used? The only requirement is that the stationary pdf of this Markov chain is the distribution $f_{\hat{\mathbf{x}}}(x)$. We enforce this by imposing the sufficient condition of detailed

balance. Replacing $f(x|y)$ and $f(y|x)$ in the detailed balance condition (105), the two terms with delta-functions are identical as they force $x = y$, leading to

$$h(y|x)\, g(y|x) f_{\hat{\mathbf{x}}}(x) = h(x|y)\, g(x|y) f_{\hat{\mathbf{x}}}(y) \tag{109}$$

as the equation to be satisfied by the proposal $g(x|y)$ and the acceptance $h(x|y)$ to ensure that the stationary distribution of the Markov chain is indeed $f_{\hat{\mathbf{x}}}(x)$.

- It is essential to return the value $x_n = x_{n-1}$ if the proposal x is not accepted instead of keeping on proposing new values until they get accepted as we used to do in a standard rejection method. Otherwise the detailed balance conditions leads to an integral equation difficult to solve for $g(x|y)$ and $h(x|y)$.

- We write now in full the recurrence relation of the Markov chain:

$$f_{\hat{\mathbf{x}}_{n+1}}(x) = \int f(x|y)\, f_{\hat{\mathbf{x}}_n}(y)\, dy = \int h(x|y)\, g(x|y)\, f_{\hat{\mathbf{x}}_n}(y)\, dy + f_{\hat{\mathbf{x}}_n}(x)\, [1 - \epsilon(x)] \tag{110}$$

where

$$\epsilon(x) = \int h(z|x)\, g(z|x)\, dz \tag{111}$$

is the acceptance probability given x. The overall acceptance probability is

$$\epsilon = \langle \epsilon(x) \rangle = \int dx \epsilon(x) f_{\hat{\mathbf{x}}}(x)\,. \tag{112}$$

- There are a number of different solutions to the detailed balance condition in its form Eq. (105). We will focus here on one of the most useful and widespread solutions: the Metropolis algorithm.

11. Metropolis algorithm

- In the Metropolis algorithm, we fix the proposal distribution $g(x|y)$ and solve for the acceptance probability. The proposal $g(x|y)$ is arbitrary insofar the resulting algorithm is *ergodic*. In particular, it is required that all possible values for the random variable $\hat{\mathbf{x}}$ have the chance to be reached eventually by the Markov chain.

- Given $g(x|y)$ we must find then a function $h(x|y)$ satisfying

$$0 \leq h(x|y) \leq 1\,, \tag{113}$$

$$\frac{h(x|y)}{h(y|x)} = q(x|y) \tag{114}$$

  where we have introduced

$$q(x|y) = \frac{g(y|x) f_{\hat{\mathbf{x}}}(x)}{g(x|y) f_{\hat{\mathbf{x}}}(y)} \tag{115}$$

  which satisfies the condition

$$q(y|x) = \frac{1}{q(x|y)}\,. \tag{116}$$

- To find all solutions of Eq. (114) we introduce the function $\omega(x,y)$ by means of

$$h(x|y) = \sqrt{q(x|y)}\,\omega(x,y) \Leftrightarrow \omega(x,y) \equiv \frac{h(x|y)}{\sqrt{q(x|y)}} \tag{117}$$

  which, when replaced in Eq. (114), requires the symmetry condition $\omega(x,y) = \omega(y,x)$.

- Condition (113) is more delicate. A possibility is to demand that $\omega(x,y)$ depends on x and y through the function $q(x|y)$, i.e. $\omega(x,y) = \omega[q(x|y)]$, with the symmetry condition $\omega(x,y) = \omega(y,x)$ being equivalent

to $\omega(q) = \omega(1/q)$. The requirement (113) then leads to $0 \leq \sqrt{q}\,\omega(q) \leq 1$ or $\omega(q) \leq 1/\sqrt{q}$, from where $\omega(q) = \omega(1/q) \leq 1/\sqrt{1/q}$. We are looking then for functions $\omega(q)$ satisfying

$$\omega(q) = \omega(1/q)\,, \tag{118}$$

$$\omega(q) \leq \min\left(\sqrt{q}, \frac{1}{\sqrt{q}}\right)\,. \tag{119}$$

- The first solution is the one given by Metropolis *et al.*

$$\omega(q) = \min\left(\sqrt{q}, \frac{1}{\sqrt{q}}\right)\,. \tag{120}$$

It satisfies trivially $\omega(q) = \omega(1/q)$. For the other condition, we take separately the cases $q \leq 1$ and $q > 1$. If $q \leq 1$, the $\omega(q) = \sqrt{q}$. For $q > 1$ we have $\omega(q) = 1/\sqrt{q} \leq \sqrt{q}$.

- The previous choice hence leads to one of the most widely used solutions of the detailed balance condition. Recalling that $h(x|y) = \sqrt{q(x|y)}\omega[q(x|y)]$, we find

$$h(x|y) = \min\left[1, q(x|y)\right]\,. \tag{121}$$

- The second most widely used solution is that of Glauber

$$\omega(q) = \frac{1}{\sqrt{q} + \sqrt{1/q}} \tag{122}$$

which trivially satisfies the conditions above. In terms of the transition probability, we have $h(x|y) = \frac{q(x|y)}{1+q(x|y)}$.

- Another solution was used later by van Beijeren and Schulman. It is simply $\omega(q) = C$, a constant. The constant has to be chosen such that $C \leq \max_{x,y} \sqrt{q(y|x)}$, in order to fulfill the condition $h(x|y) \in [0, 1]$.

- A simple example: Gaussian distribution. We consider the following random variable $\hat{\mathbf{x}}$ whose pdf is Gaussian:

$$f_{\hat{\mathbf{x}}}(x) = C\, \exp(-\frac{x^2}{2})\,. \tag{123}$$

  The normalization constant C is irrelevant for the Metropolis algorithm.

- We take the following proposal probability:

$$g(x|y) = \begin{cases} \frac{1}{2\Delta}\,, & |x-y| < \Delta\,, \\ \\ 0\,, & \text{otherwise}\,. \end{cases} \tag{124}$$

  In other words, x is drawn from a uniform distribution in the interval $(y-\Delta, y+\Delta)$. The constant Δ is arbitrary for now, but later we will determine which is the best choice for it.

- Obviously, the proposal satisfies the symmetry condition $g(x|y) = g(y|x)$, so the function $q(x|y)$ is

$$q(x|y) = \frac{g(y|x) f_{\hat{\mathbf{x}}}(x)}{g(x|y) f_{\hat{\mathbf{x}}}(y)} = \exp\left(\frac{y^2 - x^2}{2}\right)\,. \tag{125}$$

- We now need an acceptance probability $h(x|y)$. We will use Metropolis choice $h(x|y) = \min[1, q(x|y)]$. In our example:

$$h(x|y) = \begin{cases} 1\,, & |x| < |y|\,, \\ \\ \exp\left(\frac{y^2-x^2}{2}\right)\,, & |x| > |y|\,. \end{cases} \tag{126}$$

- As usual, the acceptance step is a Bernuilli process with probability q. We draw a $\hat{\mathbf{U}}(0,1)$ random number u and compare it with q. If $u \leq q$, we accept the proposal. Note that if $q = 1$, the proposal is always accepted and there is no need to spend time drawing a random number and comparing it to 1. The algorithm can be summarized as follows

  1 Given y, propose x uniformly from $(y - \Delta, y + \Delta)$.

  2 If $|x| \leq |y|$, accept x.

  3 If $|x| > |y|$, accept x with probability $q = \exp\left(\frac{y^2 - x^2}{2}\right)$. If x is rejected, return y again.

- The program could look like this:

```
! Metropolis algorithm to sample the Gaussian distribution
double precision function ran_gauss_mc(y,delta)
implicit none
double precision:: x,y,delta,ran_u
x=y+delta*(2.0d0*ran_u()-1.0d0)
if(abs(x) > abs(y)) then
   if (ran_u() > exp(0.5d0*(y-x)*(y+x))) then
   ! Reject. Do not accept the proposed value. Keep old one.
      ran_gauss_mc=y
      return
   endif
endif
y=x
ran_gauss_mc=y
end function ran_gauss_mc
```