Exercises NMSM 2024-2025

Name Surname
matr. 0123456

November 16, 2024

Lecture 1: Crude Monte Carlo and inversion sampling

Sampling random points within D-dimensional domains by hit and miss We want to evaluate the volume of an ellipsoid of given semi-axes $a = 3,\ b = 2,\ c = 2$. In order to reduce the error on our estimate from the get go, we make the following consideration. We can approach the problem in 2 ways : we can consider an integration box containing the whole ellipsoid and proceed with our calculation, or use to our advantage the problem's symmetry, and use as integration box the one containing one octant of the ellipsoid, and then multiply our estimate by 8.
The latter choice has the effect to reduce the variance in our sampling, and by consequence the error on our estimate.

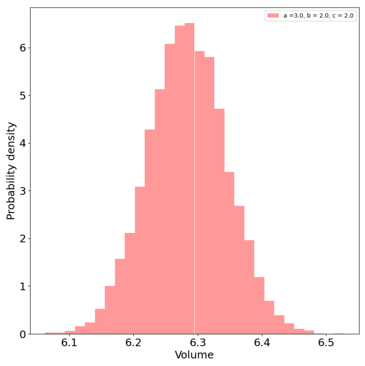


Figure 1.1: Volume distribution of the Monte Carlo estimate by hit and miss

This happens because , given that each hit or miss procedure is in fact a Bernoulli trial, the error on the volume estimate is evaluated by :

$$\sigma_V = \sqrt{\frac{p(1-p)}{N}} V_{\text{integration box}}$$

Where p is our success probability, i.e. the probability to hit inside the integration box, and N is the number of trials, i.e. the number of iterations.
So, as we seen $\sigma_V \propto V_{\text{integration box}}$, so the smaller it is, better is our estimate.
Given this consideration we report in figure 1.1 the result obtained using the procedure with the ellipsoid octant for an ellipsoid of given semi-axes $a = 3,\ b = 1,\ c = 1$:

$$\langle V \rangle_1 \simeq 6.283 \pm 0.060$$

Now we propose to evaluate the volume of an ellipsoid of given semi-axes $a = 3,\ b = 1,\ c = 1$. From the previous considerations we'll expect to have on this evaluation a smaller error, specifically, considering a situation in which p and N are the same, we can expect an error that is $\frac{V_{\text{box1}}}{V_{\text{box2}}} = 4$ times smaller, even before running any simulation.
We obtain for this ellipsoid:

$$\langle V \rangle_2 \simeq 1.571 \pm 0.015$$

Which confirms our prediction on the error. One can see also graphically this by compating the 2 volume distributions and see in figure 1.2 that the one with the smaller integration box is narrower.
Moreover, given that the same volume can be evaluate analitically, we can study the deviation of the estimate from the analytical value.
In figure 1.3 we report for each of the 2 simulations the deviation from the analytical value, i.e $\Delta f = \mid f_{th} - \langle f \rangle \mid$ as a function of the number of iterations.

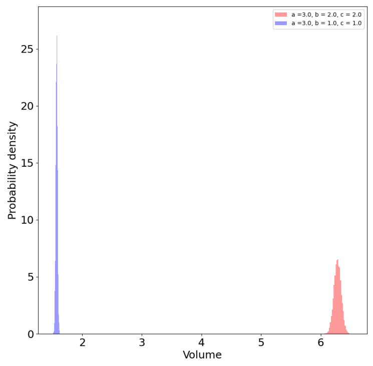


Figure 1.2: Volume distribution comparison between the 2 ellipsoids

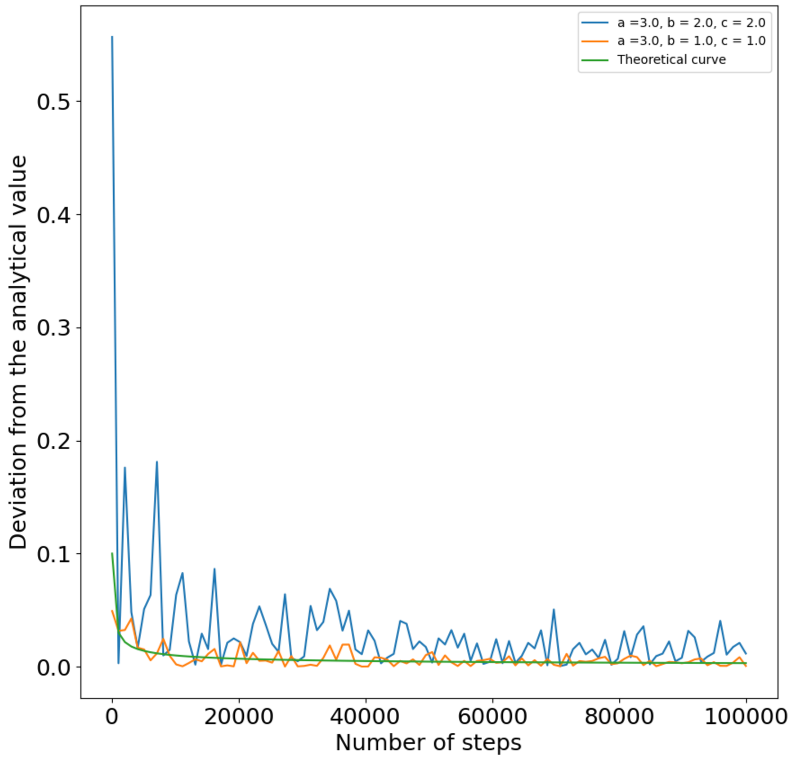


Figure 1.3: Comparison between the relative deviations for the 2 ellipsoids and the expected theroretical trend

Notice that, as discussed before, even if having a bigger number of iterations has a net effect of diminishing the error, the bigger variance in the case of the first ellipsoid makes the deviations bigger respect to the second ellipsoid.
We plotted for reference the general trend $\frac{1}{\sqrt{N}}$ which is the one to be expected in this kind of curves.

Sampling random numbers from a given distribution: inversion method We start from the following PDF's:

$$\rho_1(x) = cxe^{-x^2} \;, x \in \mathbb{R}^+ \qquad \rho_2(x) = bx^4 \;, x \in [0,3]$$

We first proceed by doing the normalization to find the coefficients c and b:

$$\int_{\mathbb{R}^+} \rho_1(x)dx = 1 \iff c = 2 \qquad \int_0^3 \rho_2(x)dx = 1 \iff b = \frac{5}{243}$$

Then one can evaluate the CDF's by definition:

$$F_1(t) = \int_0^t \rho_1(x)dx = 1 - e^{-t^2},\; t \in \mathbb{R}^+ \qquad F_2(t) = \int_0^t \rho_2(x)dx = \frac{t^5}{243},\; t \in [0,3]$$

Finally by inversion:

$$F_1^{-1}(y) = \sqrt{\ln\left(\frac{1}{1-y}\right)} \qquad F_2^{-1}(y) = \sqrt[5]{243y}$$

For both $y \in (0,1]$.
We report in the following the obtained graphs:

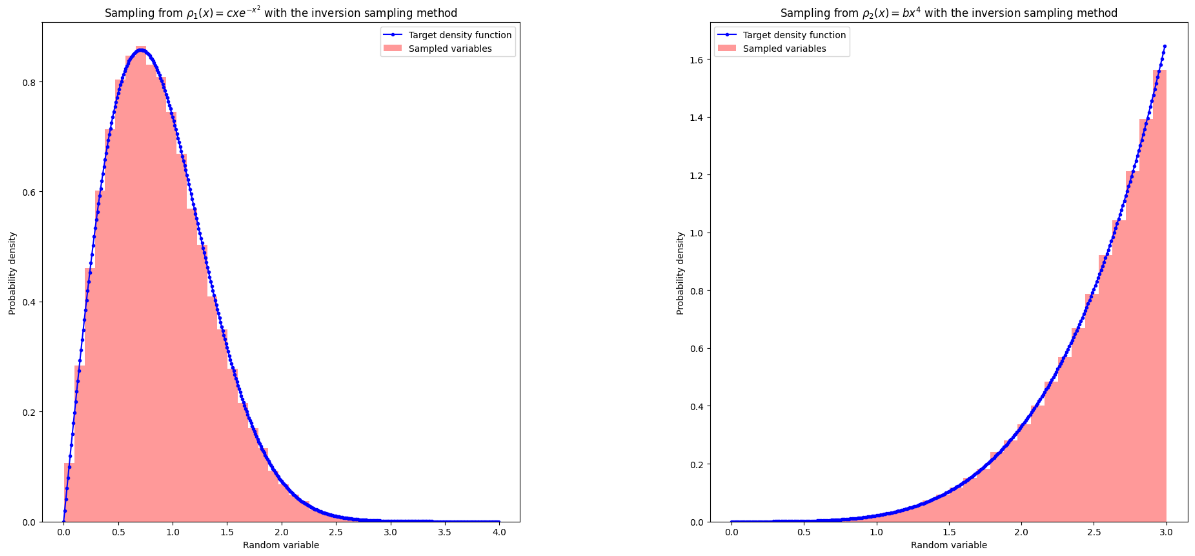


Figure 1.4: Sampling from distribution ρ_1 and ρ_2 using the inversion method.

Lecture 2: Rejection method

The rejection sampling method is useful when our pdf is not easily invertible, and so the inversion method cannot be applied.
Suppose to have as pdf:

$$\rho(x) = A_f \exp\left(-8\left(\frac{x^2}{2} + \frac{x^4}{4}\right)\right)$$

Imposing the normalization condition we obtain:

$$\int_{\mathbb{R}} \rho(x)dx = 1 \qquad \Longleftrightarrow \qquad \frac{eA_f}{\sqrt{2}} K_{1/4}(1) = 1$$

Where we denote with $K_{1/4}(x)$ the Bessel function of the second kind. We find that $A_f \simeq 1.21$. Now we want to find an invertible pdf $g(x)$ (that we'll call candidate density) with the property:

$$\rho(x) \leq cg(x) \qquad \forall x \in S$$

Where S is the sample space of $\rho(x)$. The closer the function $cg(x)$ is to $\rho(x)$ the more efficient the sampling is.

There can be different ways to choose the candidate density. We explore different possibilities in 1D and higher dimensions.
First we must start with a guess for the candidate density and the c parameter. Then we see if this guess respects the proposed inequality: if it does we are done, if not we do another guess and try again.
In 1D the easiest way to to test our guess is the graphical one: we plot $\rho(x)$ and $cg(x)$ in the same graph and evaluate if the inequality holds. Another way can be to evaluate point-wise the inequality. In dimensions where no graphical rapresentation is available, the second method can be easily extended and in order to reduce the computational complexity instead of starting with a proper point-wise evaluation, we can roughly grid the hyper-domain to check if our guess of the candidate density is sensible. If it is, we refine the grid and iterate the process until we are satisfied.

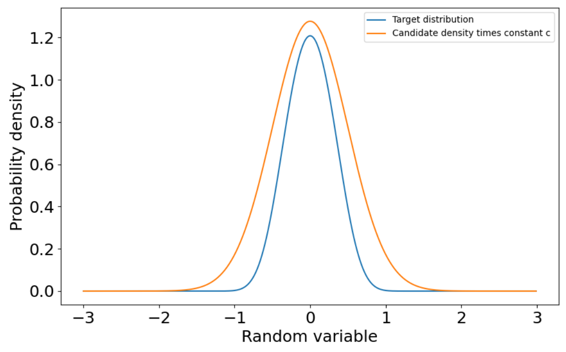


Figure 1.5: Graphical method for evaluating the inequality $\rho(x) < cg(x)$ in the rejection method

For our exercise, we can choose as candidate density $\mathcal{N}(0, \frac{1}{2})$ with c = 1.6, as shown in the picture 1.5

Then the sampling is done with algorithm 2. One can show graphically that the sample is good by plotted the properly binned random variables obtained by the process together with the target density, as showed in figure 1.6

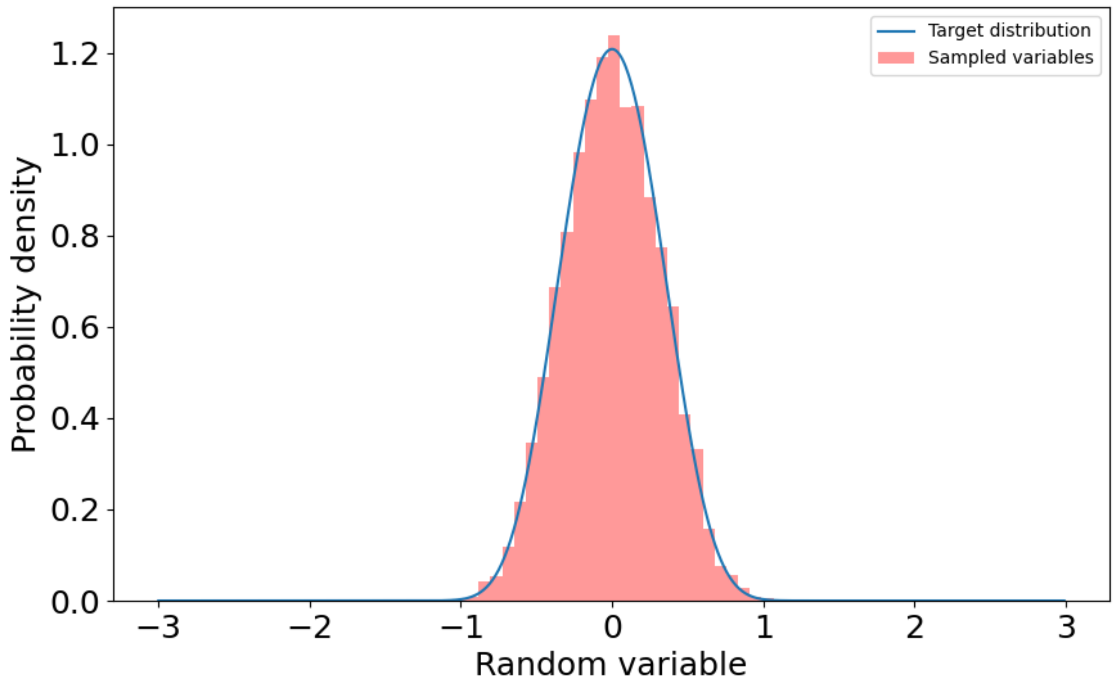


Figure 1.6: Sampled variables with target density

Lecture 3: Importance sampling

Part 1 We want to estimate the following integral:

$$\int_0^{\pi/2} \sin(x)dx$$

First we observe that the integral is easily evaluable analitically and is equal to 1.
We then proceed in doing a crude Monte Carlo estimate, obtaining the result in figure 1.7, leading to the evaluation:

$$\langle f \rangle_{\text{Crude MC}} \simeq 1.0010 \pm 0.0153$$

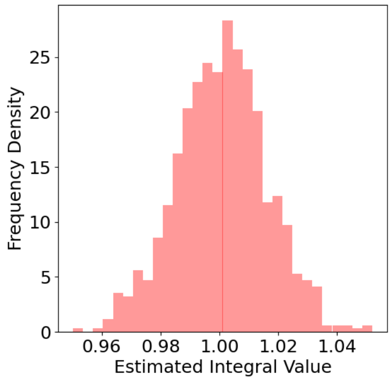


Figure 1.7: Crude Monte Carlo distribution (1000 iterations per point)

In order to reduce the error on the Monte Carlo evaluation one common approach is to use the importance sampling method.
The main algorithm is described in 3.
We propose as the new sampling function the family of functions defined such that:

$$g_{a,b}(x) = a + bx^2 \qquad a, b \in \mathbb{R}$$

We first make some consideration on the family of functions $g_{a,b}(x)$. First we impose, as they must be PDF's, the normalization condition on $[0, \pi/2]$, so that we obtain the constraint:

$$b = \frac{24}{\pi^3} - \frac{12a}{\pi^2}$$

In this way $g_{a,b}(x)$ collapses to a one parameter family:

$$g_a(x) = a + \left(\frac{24}{\pi^3} - \frac{12a}{\pi^2}\right)x^2$$

Moreover if we impose the condition $g_a(x) > 0$ we obtain for a the restriction $0 < a < 1$

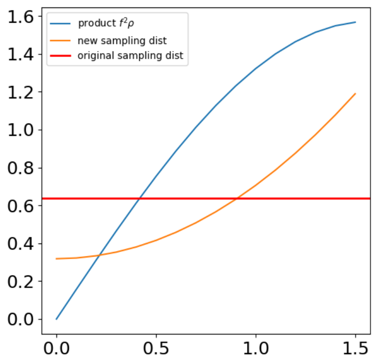


Figure 1.8: Graphical criterion for choosing a in the $g_a(x)$ family

There are now many recipes for choosing the function $g_a(x)$. We prescribe to the one that choose $g(x)$ if satisfies the condition $\rho(x) < g(x)$ when the product $f^2(x)\rho(x)$ is "large" and $\rho(x) > g(x)$ when the product $f^2(x)\rho(x)$ is "small" ,

where $\rho(x)$ in our case is the original sampling distribution, which is $U(0, \frac{\pi}{2})$.
We show in fig 1.8 that for $a = \frac{1}{\pi}$ these conditions are satisfied.

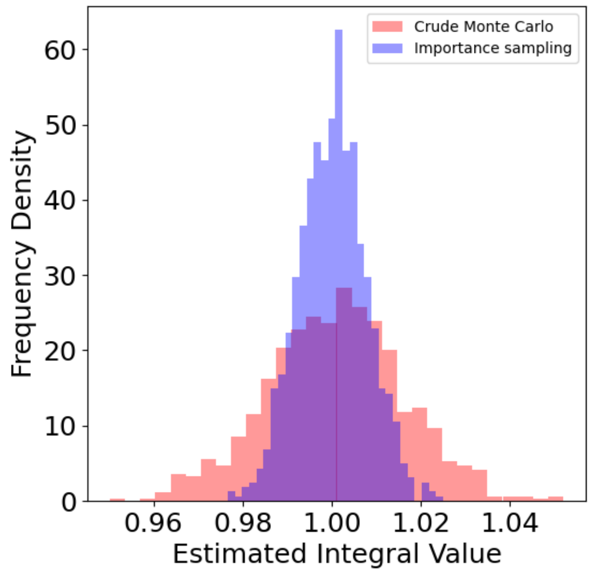


Figure 1.9: Distribution comparison between crude Monte Carlo and importance sampling with the same number of iterations

To sample from $g(x)$ we will use a rejection method algorithm using as a candidate distribution the truncated normal $\mathcal{N}(\frac{\pi}{2}, 1)_{|[0,\frac{\pi}{2}]}$. Note that this specific sampling presents some interesting technical details in Python discussed in the relative algorithm section
Now we perform the importance sampling for the same number of iterations as the crude Monte Carlo, and compare the results, seen in figure 1.9.
As expected the variance is reduced. Specifically we have for this Monte Carlo evaluation:

$$\langle f \rangle_{\text{Importance sampling}} \simeq 1.0001 \pm 0.0078$$

Which is an error reduction of $\sim 51\%$.

Part 2 We want now to evaluate the integral

$$\int_{\mathbb{R}} \left[e^{-(x-3)^2/2} + e^{-(x-6)^2/2} \right] \mathcal{N}(0,1) dx = \int_{\mathbb{R}} f(x) \mathcal{N}(0,1)$$

Where $\mathcal{N}(0,1)$ is the normal distribution with mean 0 and variance 1. One can prove (to prove) that the integral has the analytical value:

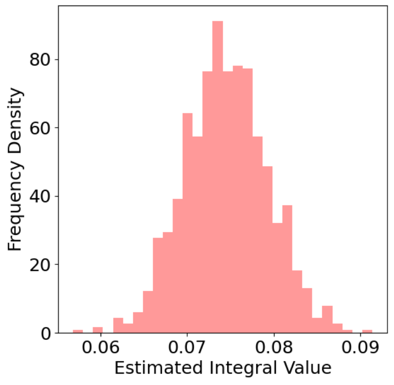
$$\int_{\mathbb{R}} \left[e^{-(x-3)^2/2} + e^{-(x-6)^2/2} \right] \mathcal{N}(0,1) dx = \frac{1 + e^{27/4}}{\sqrt{2} e^9} \simeq 0.075$$

Instead of going through the analytical route, one can think to evaluate it by using a crude Monte Carlo method, by sampling respect the normal distribution, i.e. by evaluating $\langle f(x) \rangle_{\mathcal{N}(0,1)}$.
We obtain:

$$\langle f(x) \rangle_{\mathcal{N}(0,1)} = 0.0743 \pm 0.0048$$

As also shown in the picture 1.10.

If, as proposed by the exercise, we now try to evaluate the same integral by the importance sampling method using as sampling function the uniform distribution $U(-8,-1)$ for 1000 iterations, we will see nothing.
It's easy to see way if we show the involved functions together in a graph 1.11
The uniform distribution samples in that interval values for which the integrand function assumes values between 10^{-41} to 10^{-5}, so is not possible with this iteration number to reach convergence.



distribution for the eval-

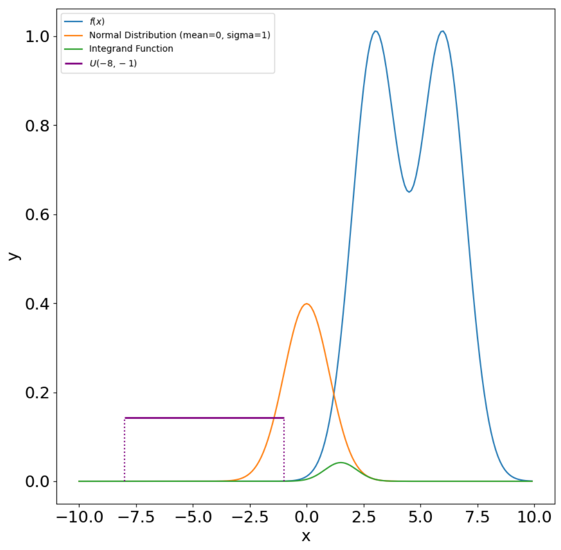


Figure 1.11: Graphical analysis for the bad choice of $U(-8,-1)$ as a sampling function for the integral $\langle f(x) \rangle_{\mathcal{N}(0,1)}$

With this number of iterations one can do way better if proposes for the importance sampling the distribution $U(-1, 4.5)$, in this case one obtain the better estimate:

$$\langle f(x)\mathcal{N}(0,1) \rangle_{U(-1,4.5)} = 0.074518 \pm 0.000003$$

This result is further presented in figure 1.12.

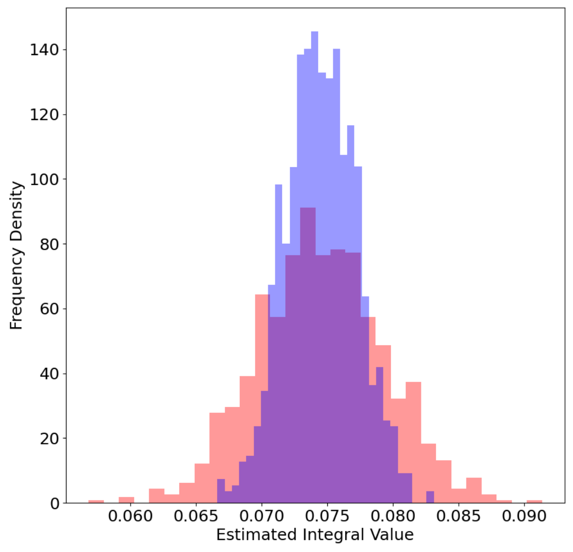
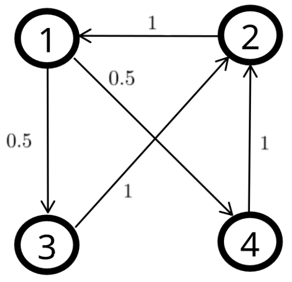


Figure 1.12: Comparison between the integral estimate distributions

Lecture 4: Markov chains

Part 1 We report the stochastic matrixes with the relative digraphs in figure 1.13.
One can prove from the digraph representation that chain A is irreducible, while chain B is not (for example in chain B from state 2 is not possible to reach any other state).

$$(A) \quad \boldsymbol{\mathcal{P}} = \begin{pmatrix} 0 & 0 & 0.5 & 0.5 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{pmatrix}$$



$$(B) \quad \boldsymbol{\mathcal{P}} = \begin{pmatrix} 0.3 & 0.4 & 0 & 0 & 0.3 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0.6 & 0.4 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 \end{pmatrix}$$

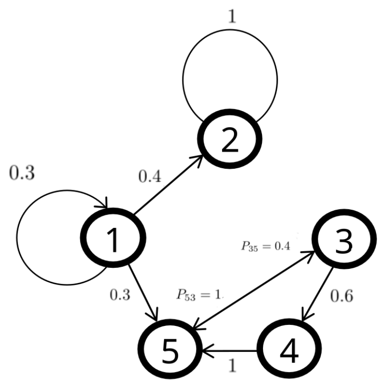


Figure 1.13: Stochastic matrixes with relative digraph representation

Part 2 Suppose to have the Markov chain with the following stochastic matrix:

$$P = \left\| \begin{matrix} \frac{1}{2} & \frac{1}{3} & \frac{1}{6} \\ \frac{3}{4} & 0 & \frac{1}{4} \\ 0 & 1 & 0 \end{matrix} \right\|$$

One can show with the corresponding digraph that is irreducible and aperiodic (spend some extra word). Now we evaluate the stochastic matrix in the limit $n \to +\infty$ in 2 ways, first by using the analytical approach and the by computation trying to estimate it. First we evaluate the invariant distribution:

$$\bar{\pi} P = \bar{\pi} \iff P^t \bar{\pi}^t = \bar{\pi}^t$$

In the end one as to solve this linear system with the normalization constraint $\sum_i \pi_i = 1$. We then obtain the following linear system:

$$\begin{cases} \frac{1}{2}\pi_1 + \frac{3}{4}\pi_2 = \pi_1 \\ \frac{1}{3}\pi_1 + \pi_3 = \pi_2 \\ \frac{1}{6}\pi_1 + \frac{1}{4}\pi_2 = \pi_3 \\ \pi_1 + \pi_2 + \pi_3 = 1 \end{cases} \iff \bar{\pi} = \left(\frac{1}{2}, \frac{1}{3}, \frac{1}{6}\right)$$

The limiting stochastic matrix will have for each row the invariant distribution:

$$P^\infty = \begin{Vmatrix} \frac{1}{2} & \frac{1}{3} & \frac{1}{6} \\ \frac{1}{2} & \frac{1}{3} & \frac{1}{6} \\ \frac{1}{2} & \frac{1}{3} & \frac{1}{6} \end{Vmatrix}$$

Now we should obtain the same result after many steps:

Balls and boxes We can solve this exercise using a Markov chain approach. Called a_i the state in which there are i red balls in box A there can be just 3 possible states. Our chain is then represent by the following digraph:

——— INSERT DIGRAPH ———-

So we need to evaluate the transition probabilities. In general, for a given state i:

- $a_i \to a_{i+1} =$ $\mathbb{P}$(red extraction in B) and $\mathbb{P}$(white extraction in A) - $a_i \to a_{i-1} =$ $\mathbb{P}$(red extraction in A) and $\mathbb{P}$(white extraction in B) - $a_i \to a_i =$ ($\mathbb{P}$(red extraction in B) and $\mathbb{P}$(red extraction in A)) or ($\mathbb{P}$(white extraction in B) and $\mathbb{P}$(white extraction in A))

Now we can proceed to evaluate these probabilities and then evaluate the transition matrix elements. In general one has, for a given state i:

$$\mathbb{P}(\text{red extraction in A at state i}) = \frac{i}{2} \qquad \mathbb{P}(\text{white extraction in A at state i}) = 1 - \frac{i}{2} = \frac{2-i}{2}$$

$$\mathbb{P}(\text{red extraction in B at state i}) = 1 - \frac{i}{3} = \frac{3-i}{3} \qquad \mathbb{P}(\text{white extraction in B at state i}) = \frac{i}{3}$$

We obtain the stochastic matrix

$$P = \begin{Vmatrix} 0 & 1 & 1 \\ \frac{1}{6} & \frac{1}{2} & \frac{1}{3} \\ 0 & \frac{2}{3} & \frac{1}{3} \end{Vmatrix}$$

Which is a stochastic matric because all rows sums up to 1.

Lecture 5: Ising 2D

The Ising model is, by earsay, the undisputed king of the Monte Carlo simulations. Its simplicity, combined with the richness of its behavior, makes it an ideal playground for testing and developing important computational techniques as the Metropolis algorithm, that we will see in the following.
In this exercise we'll simulate the 2D Ising model on a square lattice, using the Glauber dynamics (local spin flip) and the Metropolis algorithm for the configuration evolution. We'll then evaluate typical system observables at equilibrium for different temperatures and different system sizes.

We recall briefly some fundamental theoretical facts about the Ising model that we'll use through the simulation. We begin considering a generic lattice where we place N spins of the type $\sigma_i = \pm 1$. If one fixes an origin on the lattice, and consider from this the set of positions $\{\bar{r}_0, \bar{r}_1, \dots \bar{r}_{N-1}\}$ of the other lattice points, then is possible to define a configuration $\mathcal{C} = \{\sigma_{\bar{r}_0}, \dots \sigma_{\bar{r}_{N-1}}\}$ as the set of spin values assumed on each lattice point.
We define the energy of configuration $\mathcal{C}$ as:

$$\mathcal{H}(C) = -J \sum_{\langle \bar{r_i} \bar{r_j} \rangle} \sigma_{\bar{r_i}} \sigma_{\bar{r_i}}$$

Where the symbol $\sum_{\langle \bar{r_i} \bar{r_j} \rangle}$ implies that the sum is done only over the nearest neighbors. In our simulation we'll consider for simplicity $J = 1$.

In the case of the square lattice, this model can be solved exactly (Onsager).
The result for the critical temperature and the expected magnetisation in function of the temperature are reported in the following. We have that the magnetization has the following function:

$$M = \begin{cases} (1 - \sinh^{-4}(2\beta J))^{1/8} & T < T_c \\ 0 & T \geq T_c \end{cases}$$

Where holds for the critical temperature the relation:

$$\frac{k_B T_c}{J} = \frac{2}{\ln\left(1 + \sqrt{2}\right)} \simeq 2.269$$

Again, to simplify our simulation, we will consider $k_B = 1$ (natural units).

Now we can begin discussing the simulation details .
We begin by generating a random configuration at the start of the simulation such that the probability to have a spin up or a spin down is $\frac{1}{2}$. With this choice, if the system evolves with $T < T_C$ can either reach an equilibrium configuration where all spins are up or down , according to the phase diagram.
Notice that if we, for whatever reason, would like to reach an equilibrium configuration respect to the other we can do that by skewing the picking probability in favor of what we desire.
Then we go from this configuration $\mathcal{C}$ to a new configuration $\mathcal{C}'$ performing N spin local moves, where the algorithm to perform a single spin flip and decide to accept it or not, is described in the following:

Algorithm 1 Metropolis Spin-Flip Dynamics

```
 1: $(i, j) \sim \mathrm{U}(0, \mathrm{length})$
 2: neighbor_sum ← $\sum$(old_configuration[neighbor] for each neighbor in neighbors_list[$(i, j)$])
 3: $\Delta E \leftarrow 2J \times$ old_configuration[$i, j$] $\times$ neighbor_sum
 4: new_configuration ← old_configuration
 5: new_configuration[$i, j$] ← −old_configuration[$i, j$]          ▷ Flip the spin at $(i, j)$
 6: if $\Delta E \leq 0$ then
 7:     Return new_configuration
 8: else
 9:     $u \sim U(0, 1)$
10:     if $u \leq e^{-\beta \Delta E}$ then
11:         Return new_configuration
12:     else
13:         Return old_configuration
14:     end if
15: end if
```

Notice that in order to reduce the computation time a neighbor list for each atom, considering the periodic boundary condition, is evaluated using an hash table at the start of the program, so that the algorithm can use it as input. This can be implemented easily in Python using a dictionary.
In our simulation we'll define 1 Monte Carlo timestep as the number of steps in which we perform N local spin flip moves.

Thermalization Given that we're starting from a random configuration we'll need some time before reaching the thermal equilibrium, i.e. the system we'll need to evolve for a certain number of Monte Carlo timesteps that we'll call τ_{eq} before we start to evaluate any mean observable.
This process is called thermalization and is a fundamental part in any simulation of this kind.
There are different recipes to find τ_{eq} for a given system, either qualitative or quantitative. One possible way can be to plot the observables of interest against time and see qualitatively when they tend to stabilize. One can, on top of this, introduce a quantitative criterion, say that the variance per observable must be less than a predetermined value α. This approach, even if simple, has a clear caveat : it establish a lower bound for the observables time series but not an upper bound.
Another approach is to evaluate at this stage the autocorrelation time for each observable. If the process is stationary or weakly stationary this number is well defined through the time series ($C_O(s, s+t) = C_O = (0, t)\ \forall s$ for these kind of processes). Then we can use this number as a discard criterion (the common recipe is $\tau_{eq} \simeq 20\tau$, see [1]) but moreover we can use it in establishing an upper bound for the simulation.
We'll discuss in detail in the autocorrelation paragraph that errors for correlated measures are of the order $\sim (\frac{\tau}{N})^{1/2}$, so in order to achieve 1% accuracy for our measurements or better, $N > 10000\tau$.
We will use the second approach.

A point of interest is that, technically, there is no need to discard any data. The initial data (biased estimation) leads to a systematic error $\sim \frac{\tau}{N}$, while the statistical errors will be of the order $\sim (\frac{\tau}{N})^{1/2}$, which is larger. In practice, given that the systematic error can be big, and we can let it be 0 with no cost, we discard the initial transient.

We then start the thermalization process with a warmup run where the maximum number of iter-

ations is not the final one, and plot for each temperature and for each square lattice the 2 observables of interest (energy and magnetisation per spin) against the simulation time. If an observable stabilize to a constant value (apart from the statistical noise) we assume that from there on the process is stationary. Suppose instead that the 2 observables did not stabilized, for example this happens near criticality for the magnetisation due to the critical slowing down phenomenon, then if the other is stable, we can hypothesize that the magnetisation is weakly stationary.
In order to prove this (approximatively) we numerically see if has constant mean, constant variance, and constant autocorrelation time in a certain tolerance range. Then from this we have our autocorrelation time and proceed as specified before.
A technical note here: the implementation of this procedure, while straightforward, is extremely susceptible to numerical instability, so in the code a proof of concept version is presented and used, beacuse its optimization is behind the scope of this analysis.
We attach in the report an example for just 2 temperature values for the square lattice of side 50 in figure 1.14, in order to avoid cluttering, the rest of the images can be found in the folder attached to the report.

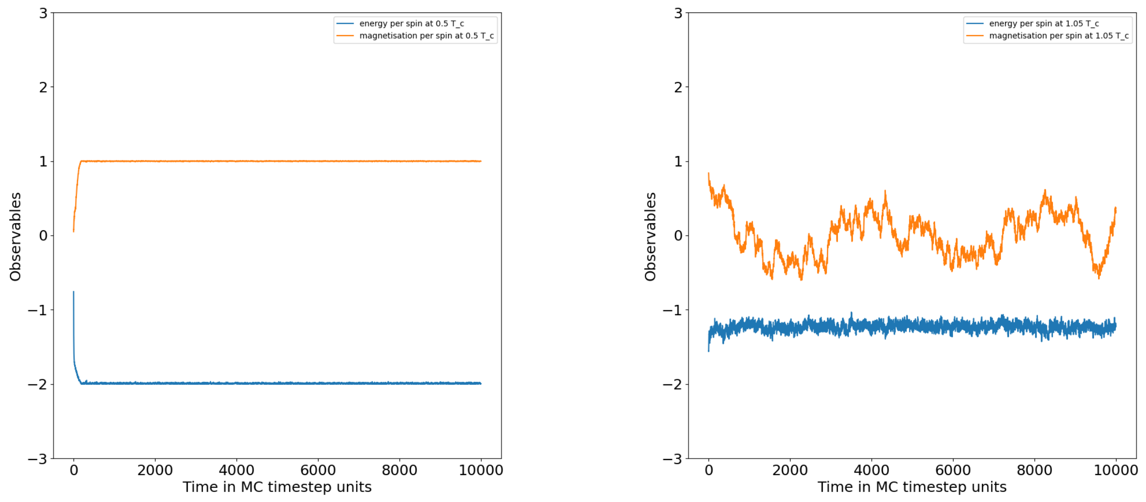


Figure 1.14: Warmup run for $T = 0.5\ T_c$ and $T = 1.05\ T_c$ respectively for the square lattice of side 50. In the first figure we are away from criticality and both observables stabilize fast. In this setting we can see that both observables stabilize to a constant value, so they can be considered stationary. In this situation

Autocorrelation In Monte Carlo simulations, successive samples for each observable are often correlated due to the nature of the sampling algorithm.
How much they are correlated is described by the autocorrelation function which can be showed that for large t goes to 0 as an exponential:

$$C_O(T) \sim e^{-t/\tau_{int}^O}$$

Where τ_{int}^O is called integrated correlation time.

Lecture 6: Polymer MMC

Lecture 7: Brussellator

The Brussellator is a model that describes an autocatalytic and oscillatory chemical reaction. In literature is given as follows:

$$\begin{cases} A \xrightarrow{K_1} X, & \text{(1a)} \\ B + X \xrightarrow{K_2} Y + D, & \text{(2a)} \\ 2X + Y \xrightarrow{K_3} 3X, & \text{(3a)} \\ X \xrightarrow{K_4} E. & \text{(4a)} \end{cases} \tag{1.1}$$

Notice that for the reaction dynamics, only the third equation is important, i.e. the autocatalysis reaction $2X + Y \rightarrow 3X$, while the other equations describe ,in order, the X production (1a), the $X \rightarrow Y$ conversion (2a), the Y depletion (4a).
So, in a context in which we are interest in simulating just the autocatalysis dynamics and not the system details, we can reduce the set of reactions to the following one:

$$\begin{cases} \emptyset \xrightarrow{k_1} A_X \\ A_X \xrightarrow{k_2} \emptyset \\ 2A_X + A_Y \xrightarrow{k_3} 3A_X \\ A_X \xrightarrow{k_4} A_Y \end{cases}$$

Where with A_X, A_Y we denote molecules of two different species.
The only hypothesis one has to make is that the molecules A and B are in high concentration so that the reactions (1a) and (2a) can happen with no problem.
This implies automatically that our system will perform limit cycles and the stationary state will be unstable and oscillatory.
This because, from the stability analysis of 1.1, we have that the system is unstable if the following condition is satisfied:

$$K_2[B] - K_4 - K_3[X_{eq}]^2 > 0$$

Which is satisfied for an high B concentration.

Instead of going through the deterministic route, we can consider the stochastic version of these reactions, transforming in this way the deterministic dynamics into a Markov process.

We denote from now with $\mathcal{C} = (X, Y)$ a state of the system where X and Y represents the number molecules of the two species A_X and A_Y respectively. The transition rates for a state $\mathcal{C} = (X, Y)$ in this process are given by the equations:

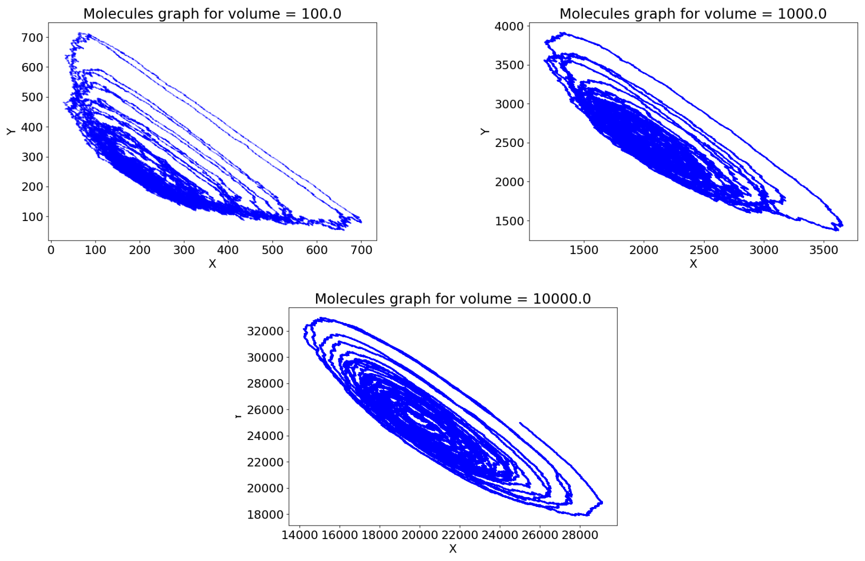


Figure 1.15: Comparison between the molecules graphs for different volumes

$$\begin{cases} w_1 = a\Omega & \text{for} \quad (X,Y) \rightarrow (X+1,Y) \\ w_2 = X & \text{for} \quad (X,Y) \rightarrow (X-1,Y) \\ w_3 = \frac{1}{\Omega^2}X(X-1)Y & \text{for} \quad (X,Y) \rightarrow (X+1,Y-1) \\ w_4 = bX & \text{for} \quad (X,Y) \rightarrow (X-1,Y+1) \end{cases}$$

Notice that these equations introduces the finite size of the system via a volume Ω. We study in the following the effects on evolution for different volumes ($\Omega = 10^2, 10^3, 10^4$) , given $a = 2$, $b = 5$. We report in figure 1.15 the molecules orbits for the different volumes.

As expected we can see we have a limit cycle for each volume, and moreover we expect an oscillatory, described by picture 1.16.

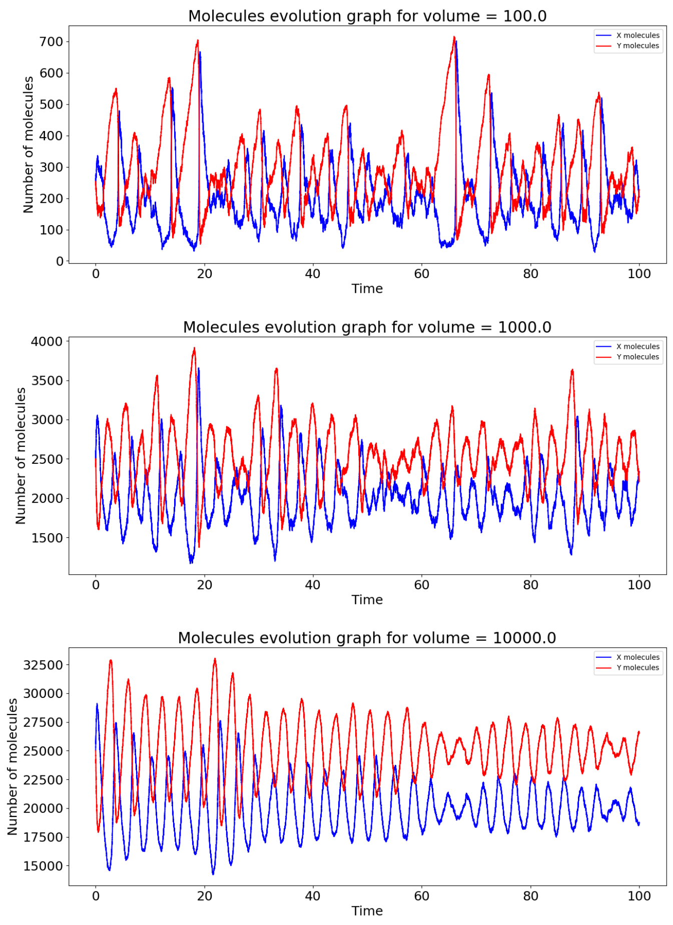


Figure 1.16: Comparison between the molecules graphs for different volumes

Notice that the volume has a regularizing effect on the oscillations.

Appendix : algorithms

Algorithm 2 Rejection Sampling Algorithm

```
1: $S \leftarrow \emptyset$
2: count $\leftarrow 0$
3: while count $<$ N do
4:     Sample $X \sim g(x)$                      ▷ Generate sample from candidate distribution
5:     Sample $U \sim \text{Uniform}(0,1)$
6:     $A \leftarrow \frac{f(X)}{c \cdot g(X)}$                      ▷ Calculate acceptance threshold
7:     if $U \leq A$ then
8:         Append $X$ to $S$
9:         count $\leftarrow$ count $+ 1$
10:    end if
11: end while
12: Return $S$
```

Implementation detail : Note that to obtain exactly N samples a while loop must be used in the algorithm.
Moreover is interesting to discuss some technical details regarding the implementation of this algorithm in Python when the candidate distribution is a truncated distribution, or in general a distribution that cannot be generated easily but needs the `scipy.stats` package to be generated.
In this case a naive implementation of algorithm 2 can hinder the perfomance greatly, because even if one has in mind to speed up the function with a `numba` decorator, this will not work, due to known incompatibility issues within the 2 cited packages.
The easiest solution to this problem is to vectorize our algorithm using the `numpy` library, overproducing samples and then cutting what is not necessary. This approach has a clear caveat, we risk an `int` overflow, so we must put attention in not choosing an high value of N for this solution. If a big number of samples is required, this approach must be revised.

Algorithm 3 Importance Sampling Algorithm

```
1: $\mu \leftarrow 0$
2: count $\leftarrow 0$
3: while count $<$ N do
4:     Sample $X \sim g(x)$                 ▷ Generate sample from proposal distribution $g(x)$
5:     $\mu \leftarrow \mu + \frac{f(X)}{g(X)}\rho(X)$
6:     count $\leftarrow$ count $+ 1$
7: end while
8: Return $\mu/N$
```

Bibliography

[1] A. Sokal. *Monte Carlo Methods in Statistical Mechanics: Foundations and New Algorithms*, pages 131–192. Springer US, Boston, MA, 1997.