BOSE-EINSTEIN CONDENSATION WITH VARIATIONAL MONTE CARLO

Author

Joseph P.Vera & Luis Jimenez

Computational Physics II: Quantum Mechanical Systems

University of Oslo

April 9, 2024

Contents

1 Introduction

The spectacular demonstration of Bose-Einstein condensation (BEC) in gases of alkali atoms ^{87}Rb, ^{23}Na, ^{7}Li confined in magnetic traps has led to an explosion of interest in confined Bose systems. Of interest is the fraction of condensed atoms, the nature of the condensate, the excitations above the condensate, the atomic density in the trap as a function of Temperature and the critical temperature of BEC, T_c.

A key feature of the trapped alkali and atomic hydrogen systems is that they are dilute. The characteristic dimensions of a typical trap for ^{87}Rb is $a_{ho} = (\hbar/m\omega_{\perp})^{\frac{1}{2}} = 1 - 2 \times 10^4$ Å . The interaction between ^{87}Rb atoms can be well represented by its s-wave scattering length, a_{Rb}. This scattering length lies in the range $85a_0 < a_{Rb} < 140a_0$ where $a_0 = 0.5292$ Å is the Bohr radius. The definite value $a_{Rb} = 100a_0$ is usually selected and for calculations the definite ratio of atom size to trap size $a_{Rb}/a_{ho} = 4.33 \times 10^{-3}$ is usually chosen. A typical ^{87}Rb atom density in the trap is $n \simeq 10^{12} - 10^{14}$ atoms per cubic cm, giving an inter-atom spacing $\ell \simeq 10^4$ Å. Thus the effective atom size is small compared to both the trap size and the inter-atom spacing, the condition for diluteness ($na_{Rb}^3 \simeq 10^{-6}$ where $n = N/V$ is the number density).

Many theoretical studies of Bose-Einstein condensates (BEC) in gases of alkali atoms confined in magnetic or optical traps have been conducted in the framework of the Gross-Pitaevskii (GP) equation. The key point for the validity of this description is the dilute condition of these systems, that is, the average distance between the atoms is much larger than the range of the inter-atomic interaction. In this situation the physics is dominated by two-body collisions, well described in terms of the s-wave scattering length a. The crucial parameter defining the condition for diluteness is the gas parameter $x(\mathbf{r}) = n(\mathbf{r})a^3$, where $n(\mathbf{r})$ is the local density of the system. For low values of the average gas parameter $x_{av} \leq 10^{-3}$, the mean field Gross-Pitaevskii equation does an excellent job. However, in recent experiments, the local gas parameter may well exceed this value due to the possibility of tuning the scattering length in the presence of a so-called Feshbach resonance.

Thus, improved many-body methods like Monte Carlo calculations may be needed.

The aim of this project is to use the Variational Monte Carlo (VMC) method and evaluate the ground state energy of a trapped, hard sphere Bose gas for different numbers of particles with a specific trial wave function.

This trial wave function is used to study the sensitivity of condensate and non-condensate properties to the hard sphere radius and the number of particles. The trap we will use is a spherical (S) or an elliptical (E) harmonic trap in one, two and finally three dimensions, with the latter given by

$$V_{ext}(\mathbf{r}) = \begin{cases} \frac{1}{2}m\omega_{ho}^2 r^2 & (S) \\ \frac{1}{2}m[\omega_{ho}^2(x^2+y^2)+\omega_z^2 z^2] & (E) \end{cases} \tag{1}$$

where (S) stands for spherical and

$$H = \sum_i^N \left(\frac{-\hbar^2}{2m}\bigtriangledown_i^2 + V_{ext}(\mathbf{r}_i) \right) + \sum_{i<j}^N V_{int}(\mathbf{r}_i, \mathbf{r}_j), \tag{2}$$

as the two-body Hamiltonian of the system. Here ω_{ho}^2 defines the trap potential strength. In the case of the elliptical trap, $V_{ext}(x,y,z)$, $\omega_{ho} = \omega_\perp$ is the trap frequency in the perpendicular or xy plane and ω_z the frequency in the z direction. The mean square vibrational amplitude of a single boson at $T = 0K$ in the trap (1) is $\langle x^2 \rangle = (\hbar/2m\omega_{ho})$ so that $a_{ho} \equiv (\hbar/m\omega_{ho})^{\frac{1}{2}}$ defines the characteristic length of the trap. The ratio of the frequencies is denoted $\lambda = \omega_z/\omega_\perp$ leading to a ratio of the trap lengths $(a_\perp/a_z) = (\omega_z/\omega_\perp)^{\frac{1}{2}} = \sqrt{\lambda}$. Note that we use the shorthand notation

$$\sum_{i<j}^N V_{ij} \equiv \sum_{i=1}^N \sum_{j=i+1}^N V_{ij}, \tag{3}$$

that is, the notation $i < j$ under the summation sign signifies a double sum running over all pairwise interactions once.

We will represent the inter-boson interaction by a pairwise, repulsive potential

$$V_{int}(|\mathbf{r}_i - \mathbf{r}_j|) = \begin{cases} \infty & |\mathbf{r}_i - \mathbf{r}_j| \le a \\ 0 & |\mathbf{r}_i - \mathbf{r}_j| > a \end{cases} \tag{4}$$

where a is the so-called hard-core diameter of the bosons. Clearly, $V_{int}(|\mathbf{r}_i - \mathbf{r}_j|)$ is zero if the bosons are separated by a distance $|\mathbf{r}_i - \mathbf{r}_j|$ greater than a but infinite if they attempt to come within a distance $|\mathbf{r}_i - \mathbf{r}_j| \le a$.

Our trial wave function for the ground state with N atoms is given by

$$\Psi_T(\mathbf{r}) = \Psi_T(\mathbf{r}_1, \mathbf{r}_2, \ldots \mathbf{r}_N, \alpha, \beta) = \left[\prod_i g(\alpha, \beta, \mathbf{r}_i) \right] \left[\prod_{j<k} f(a, |\mathbf{r}_j - \mathbf{r}_k|) \right], \tag{5}$$

where α and β are variational parameters. The single-particle wave function is proportional to the harmonic oscillator function for the ground state, i.e.,

$$g(\alpha,\beta,\mathbf{r}_i) = \exp\left[-\alpha(x_i^2+y_i^2+\beta z_i^2)\right]. \tag{6}$$

For spherical traps we have $\beta = 1$ and for non-interacting bosons ($a = 0$) we have $\alpha = 1/2a_{ho}^2$. The correlation wave function is

$$f(a,|\mathbf{r}_i-\mathbf{r}_j|) = \begin{cases} 0 & |\mathbf{r}_i-\mathbf{r}_j| \leq a \\ (1-\frac{a}{|\mathbf{r}_i-\mathbf{r}_j|}) & |\mathbf{r}_i-\mathbf{r}_j| > a. \end{cases} \tag{7}$$

2 Monte Carlo Method

The Monte Carlo method (MCM) is a stochastic (random sampling of inputs) method to solve a statistical problem [1]. MCM algorithm allows us to find numerical approximations of multi-dimensional integrals, these became very important when we are dealing with many-body system. and one of the other advantages it is a very sample method does not require a lot of mathematical background.

2.1 Monte Carlo Method: The basic intuition idea

One simple Monte Carlo experiment considers rain which falls uniformly at random the position of any raindrop may be interpreted as a realization of a uniformly distributed random variable that give us the following idea. Suppose that any raindrop can draw a dot in a certain a square surface them someone draw a circle inscribed and thinking in someone that like the math and the person, want to know what it is the probability (P) that a raindrops fall inside the circle ?; that question it is not very difficult to answer the probability is given

$$P = \frac{A_{Circle}}{A_{squart}} = \frac{\pi r^2}{(2r)^2} = \frac{\pi}{4} \tag{8}$$

The person thinks if I can count the number of raindrop inside the surface and also the number inside of circle, then the probability to be a uniform distribution(random) can be calculated as

$$P = \frac{\textbf{\# raindrops inside circle}}{\textbf{\# raindrops inside surface}} \tag{9}$$

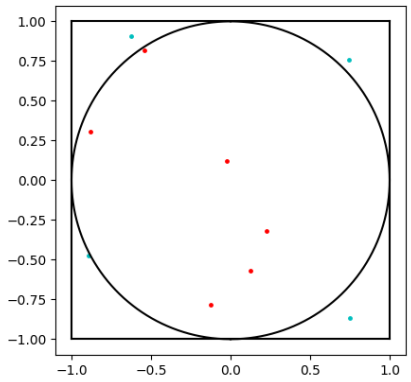
but the person already knew that the probability it $\frac{\pi}{4}$, then solving π and also A_{Circle} finds that,

$$\pi = 4P = 4\frac{\textbf{\# raindrops inside circle}}{\textbf{\# raindrops inside surface}} \tag{10}$$

$$A = A_{squart}P = A_{squart}\frac{\textbf{\# raindrops inside circle}}{\textbf{\# raindrops inside surface}} \tag{11}$$

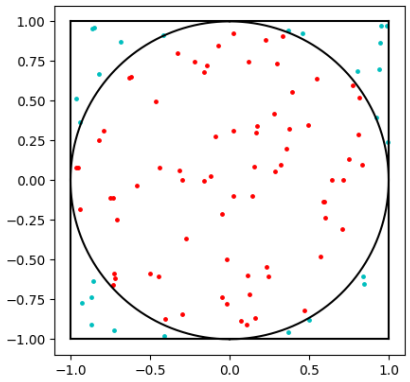
the **Eqs.** (10) and (11) gives a way to calculate the pi and Areas values the stockists way (random inputs). Finally, the person makes many repetition experiments (sampling) and after some of them realizing that the better approximation come when the number of the raindrop it is bigger.

Now suppose we want to replicate the experimental to prove it is true everything the person did, for simplicity we will use a circle the radio 1 and then the area of square is 2 in unit of $Area^2$, but we do not much time free to wait for the raining days, so we are going to do the experimenter with the computer using so called quasi random number with the help of python. first we start with "raindrops" numbers equal to 10, 10^2, 10^3, 10^4, 10^5 and 10^6 the **Fig.** 1 shows result of the experiment look like,



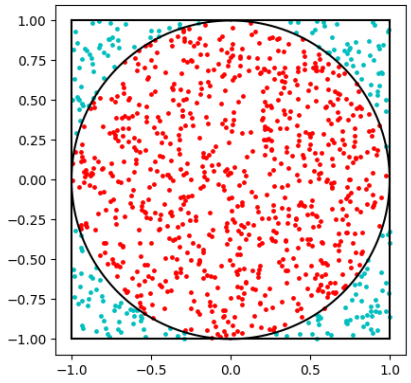
(a) Number of “raindrops” equal 10.

$Area = \pi \approx 3,2$



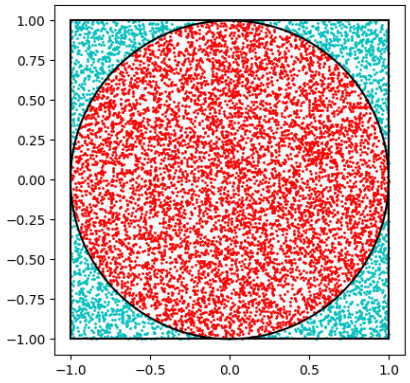
(b) Number of “raindrops” equal 10^2.

$Area = \pi \approx 3,24$



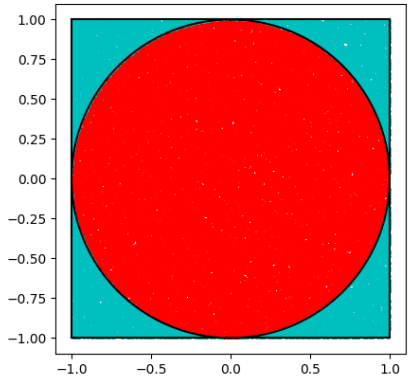
(c) Number of “raindrops” equal 10^3.

$Area = \pi \approx 3,272$



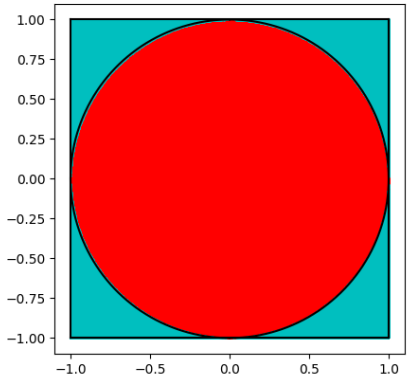
(d) Number of “raindrops” equal 10^4.

$Area = \pi \approx 3,1468$



(e) Number of “raindrops” equal 10^5.

$Area = \pi \approx 3.1446$



(f) Number of “raindrops” equal 10^6.

$Area = \pi \approx 3.141336$

Figure 1: The figures show the different values of π and Areas dependent of the number “raindrops”

We can see **Fig.** 1 easily the values of π and Area improve when increasing the number of "raindrops". the next step is to think a better way to calculate these values, it is calculating the average of many sampling with an efficient number of "raindrops" or in more formal words number of Monte Carlos cycles (MCc). In the **Fig.** 2 we have the calculation for number sampling equal to 10000 with 50000 MCc each one.

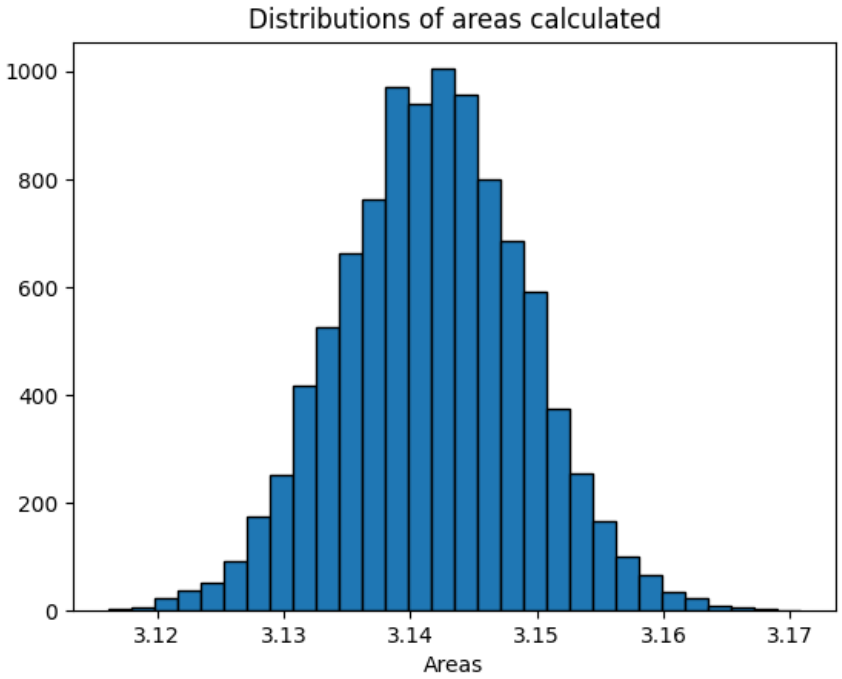
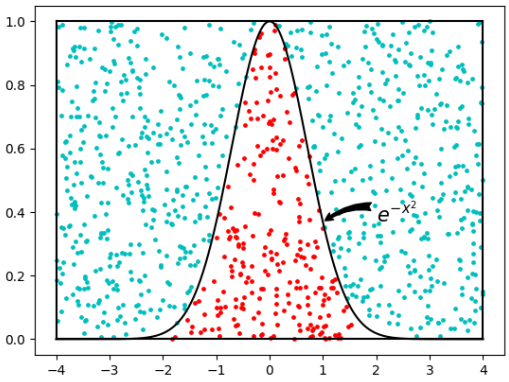


Figure 2: Distribution of the areas using 10000 sampling with 50000 MCc each one

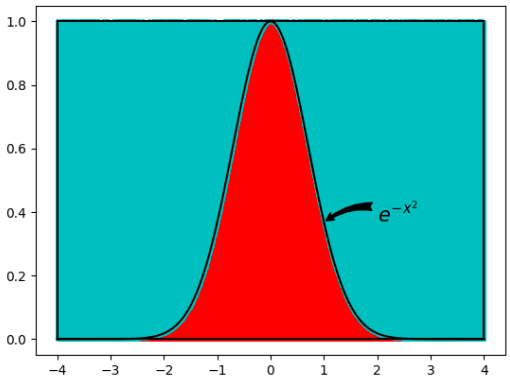
the distribution for Pi has to be equal to this, one thing we can see is the probability distribution going to follow a Gaussian that mean the average is giving by the maximum point of the Gaussian function and this value is $Area = \pi \approx 3.141723$. We know that it is not the true value of Pi, but if you want to improve you are going to need a reduction of the variance and that mean you have to increase the MCc.

This method you can implement to calculate any type function, and it is possible to generalize for many dimensions as we will see later. But to make more clear the idea we are going to calculate the "integral" (Area) of the Gaussian function in one dimension but now as you will see we implement other equivalent approach to MCM as well. First we start with the previous process, we calculate the area for 10^2 and 10^6 MCc look at **Fig.** 3,



(a) Number of MCc equal 10^3.

$Area \approx 1.584$



(b) Number of MCc equal 10^6.

$Area \approx 1.775064$

Figure 3: The figures show areas for 10^3 and 10^6 MCc.

the "exact value" calculated using Gauss quadrature integration is 1.772453, we can see MCM is not the best approximation and the question came ours, why no just uses method like Gauss quadrature?; the answer is given when we need to find multidimensional-integral because the regular methods became to have problems like so computation cost and the error increase with the dimensional and instead MCM this does not happen [2].

The other equivalent approach we can explain the following form, suppose we have any function for simplicity one dimension we want to find its area, one thing we can make it is approximate as

$$\int_a^b f(x)dx = \sum_i^N f(x_i)\Delta x_i = \frac{(b-a)}{N}\sum_i^N f(x_i) \tag{12}$$

where we are using the simplest case rectangular method, now the way to proceed instead to calculate the area for each rectangle and then add all them (rectangle method), we can just calculate the random areas between integration interval that means for just one calculation (one MCc) $Area = (b-a)f(x_i = x_{random})$ obviously this it is not a good approximation but if we do this many times and calculating an average like as we did in the last part we can find the stochastic way or in other words using MCM the area under the curve $f(x)$. We can show now the idea graphically for Gaussian integral **Fig.** 4 with $N = 9$,

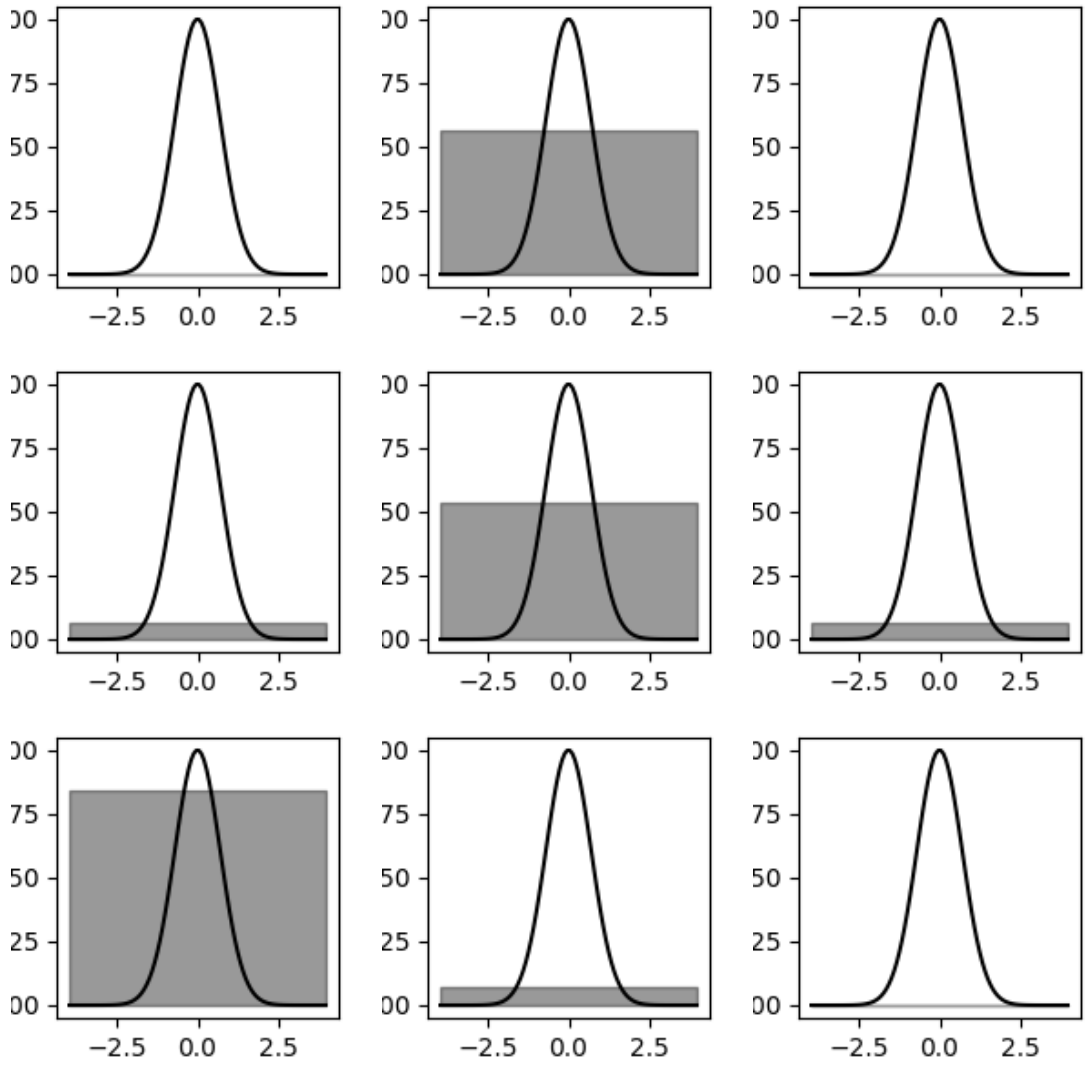


Figure 4: The Gaussian integral using MCM with MCc = 9, the value of the integral is 1.900273

It is not a good result too, but if we do for MCc $= 10^7$ the integral is 1.773115 that it is better result so we keep creasing the number of MCc the result should be converged to the exact value of the integral. The last implementation is something called brute force Monte Carlo method (BFMCM). We are going to see that BFMCM can be improved, just mentioning one example Metropolis–Hastings algorithm [2, 3]

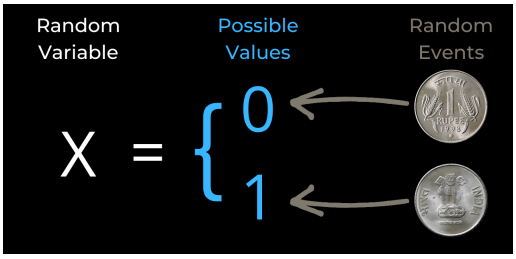
2.2 Monte Carlo formalism

The Monte Carlo methods can be described as statistical simulation methods, where statistical simulation is defined in quite general terms to be any method that utilizes sequences of random numbers to perform the simulation, so to understand the formalism of Monte Carlo Method we

need some import quantities,

1. Random variables,
2. Probability distribution functions (PDF)
3. Moments of a PDF
4. Variance σ^2

Random variables is a variable whose value is unknown, in other words we can not be possible to know the exact value of it. the more classic example it uses dices or coins to explain what does mean random, if we have for example two coins, and we roll the coins many times the output in each roll changed randomly this defines us random variable. To understand the idea suppose someone tells us if we predict the result will win a gift, then we need to chose, the problem it is obviously that we do not have the variable result it is random, but we have some information called probability, so the possible outputs are $\{$(**Heads-Heads**),(**Heads-Tails or Tails-Heads**), (**Tails,Tails**)$\}$ then we can find probability $\{\frac{1}{4}, \frac{2}{4}, \frac{1}{4}\}$, the highest probability is given when we chose coins fall one in heads and other one in tails.



https://yousefhosni.medium.com/15-important-probability-concepts-to-review-before-data-science-interview-part-1-bc1742ea7e3e

The figure above shows the idea behind a random variable where we are representing outputs **Heads** = 0 and **Tails** = 1 .

Probability distribution functions (PDF) is a statistical function that describes all the possible values and probabilities for a random variable within a given range. following the before example, PDF is given for the different outcomes probabilities.

Moments of a PDF it is a representative number that gives us information about the experiment when we talk about moment zero is becalming the Average.

Variance is the degree of spread of a data set with respect to a central value (Average) and it

calculated by taking the average of squared deviations from the average value.

Now we are going to start with some import definition, for random variables (stochastic variable) it is going to be using capital latter $\{X_1, X_2 ...\}$ and for the domain (possibles values could be taken) it going to be use lowercase letters $\{x_1, x_2 ...\}$, in the examples that means X is the event to produce by roll a coin and x is the possibles result can be 0 or 1. We know the PDF is a function $p(x)$ on the domain which, in the discrete case, gives us the probability or relative frequency with which these values of X occur $p(x) = \textbf{Prob}(X = x)$ referring to the example the probability of the answer we choose is $p(x) = \textbf{Prob}(X = 0,1 \textbf{ or } 1,0) = \frac{2}{4}$.

If we consider a continuous distribution, the PDF does not exactly actual probability, so we define the probability for the stochastic variable to assume any value on an infinitesimal interval between $x + dx$ to be $p(x)dx$, then the probability is defined

$$\textbf{Prob}(a \leq X \leq b) = \int_a^b p(x)dx \tag{13}$$

from this definition we can also define an accumulative distribution of probability (CDF) as

$$\textbf{Prob}(X \leq x) = \int_{-\infty}^{x} p(x')dx' \tag{14}$$

this says us what is the probability below a certain value x, the import of this definition it is that everyone CDF has a corresponding PDF, then we can define a density of probability

$$p(x) = \frac{d}{dx}P(x) \tag{15}$$

the density of probability has two import properties, first it has to be normalized,

$$\sum_{x_i} p(x_i) = 1 \quad \Rightarrow \quad \textbf{Discreet case} \tag{16}$$

$$\int_{x \in D} p(x)dx = 1 \quad \Rightarrow \quad \textbf{Continuos case} \tag{17}$$

and the other properties is if density probability is normalized then

$$0 \leq p(x) \leq 1 \tag{18}$$

Equ. (18) says when the probability is 1 we will sure the event has to be happened and it is 0 never will happened.

Now let $f(x)$ be an arbitrary functional then the expectation value it is defined for $X = x \in D$ as

$$\langle f(x)\rangle = \int_{x\in D} f(x)p(x)dx \tag{19}$$

and then we define the moment of the PDF

$$\langle x^n\rangle = \int_{x\in D} x^n p(x)dx \tag{20}$$

and with $n = 1$ we find the something so called mean or just avenge of p and this it is often denoted by μ,

$$\langle x\rangle = \mu = \int_{x\in D} xp(x)dx \tag{21}$$

other import quantity it is the variance and this it is defined as

$$\sigma^2(x) = \langle (x - \langle x\rangle)^2\rangle = \int_{x\in D} \langle (x - \langle x\rangle)^2\rangle p(x)dx = \langle x^2\rangle - \langle x\rangle^2 = \langle x^2\rangle - \mu^2 \tag{22}$$

the square root of the variance $\sigma(x) = \sqrt{\langle x^2\rangle - \langle x\rangle^2}$ is called standard deviation, the interpretation is quite similar to variance it is the "spread" of p around its mean.

Everything so far can be an easy way generalized to N dimension system, so we have n stochastic variables $\{X_1, ..., X_n\}$ then it is possibly generalized all quantities before as

$$\langle H(x_1, ..., x_n)\rangle = \int_{x_1\in D(X_1)} ... \int_{x_n\in D(X_n)} H(x_1, ..., x_n)p(x_1, ..., x_n)dx_1...dx_n \tag{23}$$

the moments

$$\langle r(x_1, .., x_n)^n\rangle = \int_{x_1\in D(X_1)} ... \int_{x_n\in D(X_n)} r(x_1, .., x_n)^n p(x_1, ..., x_n)dx_1...dx_n \tag{24}$$

the central value

$$\langle r(x_1, .., x_n)\rangle = \mu = \int_{x_1\in D(X_1)} ... \int_{x_n\in D(X_n)} r(x_1, ..x_n)p(x_1, ..., x_n)dx_1...dx_n \tag{25}$$

and variance

$$\sigma^2(r(x_1, ..x_n)) = \langle r(x_1, ..x_n)^2\rangle - \mu^2 \tag{26}$$

with this "tools" we can start to make a more formal introduction to MCM, we start trying to find some type of integral as did before,

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

the simplest form to solves it is using rectangles method last said before

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

so far nothing new but now if we try to calculate the expectation value of $f(x)$ where x is a random variable and supposing that we can now the probability distribution,

$$\langle f(x)\rangle = \int_a^b f(x)p(x)dx \approx \frac{(b-a)}{N}\sum_{i}^{N} f(x_i)p(x_i)$$

let just says we are dealing with uniform probability where p(x) = 1 in the MCM that meaning we are always accepted the values x_i to calculate the average,

$$\langle f(x_i)\rangle = I \approx \frac{(b-a)}{N}\sum_{i}^{N} f(x_i) \tag{27}$$

this result gives us a way to say why the integral can be calculated as we did in the previous part **Section** 2.1, because we have a stochastic variable X then it has associated a probability distribution and then this has a density of probability $p(x)$. This process mention became relevant if we want to improve the MCM because if we can do something that gives us what random values $p(x_{random})$ is good (accepted) and which other is not good (not accepted) that leaves us the metropolis algorithms. The final comment about the process is the "only" changes we will have to do it is just finding more optimal form to chose points with the helps of PDF, so this translates to "optimization" or improved to MCM.

The last thing we can mention it the error it is reacted with standard deviation, for the one sampling or simulation (1-MCs) the variance is given,

$$\sigma^2_{1-MCs} = \int_a^b (f(x) - \langle f(x)\rangle)^2 p(x)dx = \frac{(b-a)}{N}\sum_{i}^{N}(f(x_i) - \langle f(x_i)\rangle)^2 p(x_i)$$

let assume that we are dealing with uniform distribution

$$\sigma^2_{1-MCs} = \frac{(b-a)}{N}\sum_{i}^{N}(f(x_i) - \langle f(x_i)\rangle)^2 = \frac{(b-a)}{N}\sum_{i}^{N}\left[f(x_i)^2 - \left(\frac{(b-a)}{N}\sum_{i}^{N} f(x_i)\right)^2\right]$$

or just rewriting as

$$\sigma^2_{1-MCs} = \langle f^2\rangle - \langle f\rangle^2 \tag{28}$$

supposes we make M simulation to improve the result of the integral as was shown in the last and each one is independent (uncorrelated) then we can calculate for each sampling an average

that leaves the result like **Equ.** (27) for $i-nth$ sampling just called $\langle f\rangle_i$, then the overage after M Monte Carlos sampling $M-MCs$ is given

$$\langle I\rangle_{M-MCs} = \frac{1}{M}\sum_{i}^{M}\langle f\rangle_i$$

and if we assume in **Equ.** (28) N is enough large to approximate to the true, the variance can be written

$$\sigma^2_{M-MCs} \approx \frac{1}{M}(\langle f^2\rangle - \langle f\rangle^2) = \frac{\sigma^2_{1-MCs}}{M} \tag{29}$$

that gives us a standard deviation that it is proportional to

$$\sigma^2_{M-MCs} \sim \frac{1}{\sqrt{M}} \tag{30}$$

therefore to minimize the error we need to do **Equ.** (30) smaller that as possible and this it is done when M larger that as possible, so to make a good calculation in MCM we are going to need many MCs like we "proof" in the **Section** 2.1 using "experiment with raindrops".

2.3 Variational Monte Carlo Method (VMCM)

The VMCM is a way to find the ground state of any system, we are going to see in us case will be the ground state of the wave function of many components system can be a solid, molecules or just a many interaction electrons as will be seen when we solve the **project 1** what it is the idea of everything we have been done, and we will do in this writing. The main idea of VMCM is in the construction of the so called trial wave function $\Psi_T(R,\alpha)$, where R and α can be en general a vector and this last it knows as variational parameter, which should be optimized to find the ground state of the system [4, 5].

The basic steps to start with VMCM

- Find the trial wave $\Psi_T(R,\alpha)$
- Define so called local energy $E_L(R,\alpha) = \frac{1}{\Psi_T(R,\alpha)}\hat{H}\Psi_T(R,\alpha)$
- Define PDF us case of the trial wave function $\Psi_T(R,\alpha)^2$
- Calculate the expectation value of the energy $\langle\hat{H}\rangle$ but rewriting as a function of $\Psi_T(R,\alpha)^2$ and $E_L(R,\alpha)$
- Find the optimal α through minimization of $\langle E[\alpha]\rangle$

- Find the standard deviation $\sigma(E) = \sqrt{\langle \hat{H}^2 \rangle - \langle \hat{H} \rangle^2}$ to see how good it is the result

following the steps

$$\langle E \rangle = \frac{\int_a^b \Psi_T(R,\alpha)^\dagger \hat{H} \Psi_T(R,\alpha) dR}{\int_a^b \Psi_T(R,\alpha)^2 dR} \tag{31}$$

multiplying and dividing in the numerator by $\Psi_T(R,\alpha)$ and assuming the wave function is normalized we can rewrite the expectation value as,

$$\langle E \rangle = \int_a^b \Psi_T(R,\alpha)^2 E_L(R,\alpha) dR \tag{32}$$

we know that norm of the wave function can be interpreted as PDF, then we want to calculate this integral, but it is a difficult thing to do, although the density probability looks pretty good. The best way to find the integral when the dimension growth it is using MCM, so with this idea the first approach we can do just say that us PDF is uniform,

$$\langle E \rangle \approx \frac{1}{N} \sum_i^N E_L(R_i,\alpha) \tag{33}$$

where for simplicity we have used $b-a=1$ that meaning we have calculating the integral between 0 to 1, also take into account R_i and α can be a vector or just a scalar as we mentioned.

Then the step is to find the best α parameter through minimization, we know that α can be a vector o just a scalar, so in general we havethen

$$\frac{d}{d\alpha}\langle E[\alpha] \rangle = \bar{E}_\alpha = 2\left(\langle \frac{\bar{\psi}_\alpha}{\psi[\alpha]} E_L[\alpha] \rangle - \langle \frac{\bar{\psi}_\alpha}{\psi[\alpha]} \rangle \langle E_L[\alpha] \rangle \right). \tag{34}$$

where $\bar{\psi}_\alpha = \frac{d\langle \psi[\alpha] \rangle}{d\alpha}$. Then final step is just find the variance, it should be near to zero if we have a good convergence of MCM and α.

2.3.1 Markov chains and Metropolis algorithm

The Markov chains (MC) is a special type of stochastic process in which the probability of an event occurring depends only on the immediately preceding event, this especial process is an ally to find the so called Metropolis algorithm this helps us find a better way to calculate the integral using MCM.

To find a Monte Carlos Metropolis algorithm (MCMa) we need defining PDF but also a transition probability distribution function (TPDF), using the MC we know the transition probability

only has to be depended on the probability immediately preceding event if we start to assume a discrete probability how show figure below

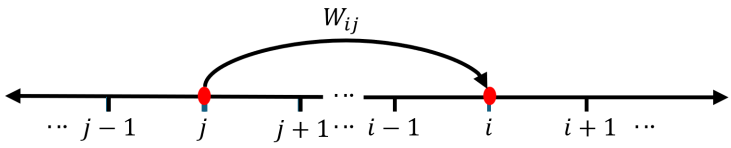


Figure 5: Illustration of the Markov chains process

In the **Fig.** 5 we have that $i \longrightarrow w_i(t)$ is the PDF in us case is a square wave function, and it is mean to find the system in the state i at the time t, then defining the TPDF $W_{ij}(t)$

$$w_i(t) = \sum_j W_{ij}(t) w_j(t) = \sum_j W_{(j \longleftarrow i)}(t) w_j(t)$$

it gives us the TPDF to pass the state $w_j(t)$ to $w_i(t)$.

The problem comes that we do not have the TPDF, but we can do some subjection about that because has to satisfy some properties since

- $0 \leq w_i(t) \leq 1$
- $\sum_i w_i = 1$
- $0 \leq W_i j(t) \leq 1$
- the sum of some columns or rows $\sum_i W_{ij}(t) = \sum_j W_{ij}(t)$
- The eigenvalues $\lambda(W_i j) \leq 1$ because TPDF must be a stochastic matrix, the $\lambda_{max} = 1$

for simplicity we can do other assumption that $W_{ij}(t)$ time independent the transition or jumping the state i to j are instantaneously. We can find the probability after the time t,

$$w_i(t_0 + \varepsilon) = \sum_j W_{(j \longrightarrow i)} w_j(0) = W w_j(0)$$

$$w_i(t_0 + 2\varepsilon) = \sum_j W_{(j \longrightarrow i)} w_j(\varepsilon) = W w_j(\varepsilon) = W^2 w_j(0)$$

$$w_i(t_0 + 3\varepsilon) = \sum_j W_{(j \longrightarrow i)} w_j(2\varepsilon) = W w_j(2\varepsilon) = W^3 w_j(0)$$

if we follow with this idea after n steps

$$w_i(t_0 + n\varepsilon) = \sum_j W_{(j \longrightarrow i)} w_j((n-1)\varepsilon) = W^n w_j(0)$$

let $t_0 = 0$ and $n\varepsilon = t$, we found the probability at the time t

$$w_i(t) = W^{t/\varepsilon} w_j(0)$$

then we consider W has eigenvalues $\{\lambda_0, \lambda_1, ..., \lambda_k ... \lambda_F\}$ where the largest value is $\{\lambda_0 = 1 \leq \lambda_1 \leq ... \leq \lambda_k ... \leq \lambda_F\}$ and eigenvectors $\{v_0, v_1, ..., v_k, ... v_F\}$, this way we can expand $w_j(0)$ as a lineal combination of v_k

$$W w_j(0) = W \sum_k^F \alpha_i v_k = \sum_k^F \alpha_i \lambda_k v_k$$

introducing a variables called dumping time $\tau_k = -\frac{1}{\ln \lambda_k}$, then we can write the PDF for $w_i(t)$ as

$$w_i(t) = \sum_k^F \alpha_k v_k \exp(-t/\tau_k) = \alpha_0 v_0 + \sum_{k=1}^F \alpha_k v_k \exp(-t/\tau_k) \tag{35}$$

after the so large time $t \longrightarrow \infty$ the **Equ.** (35) becomes

$$\lim_{t\longrightarrow\infty} w_i(t) = \alpha_0 v_0 + \lim_{t\longrightarrow\infty} \sum_{k=1}^F \alpha_k v_k \exp(-t/\tau_k)$$

$$\lim_{t\longrightarrow\infty} w_i(t) = w_i = \alpha_0 v_0 \tag{36}$$

Equ. (36) says for long time PDF becomes to be constant, this state is called the more likely state or stationary state, the next step it is constructing the Metropolis algorithm and this will be useful.

We do not TPDF, therefor we need to make a model is there when Metropolis algorithm helps ours. We are going to start with the TPDF can be written as product of $W_{(j\Longrightarrow i)} = T_{(j\Longrightarrow i)} A_{(j\Longrightarrow i)}$ where $T_{(j\Longrightarrow i)}$ and $A_{(j\Longrightarrow i)}$ gives us the probability to do a transition (jumping) and the probability to accept the transition (jumping) from j to i respectively

$$w_i(t) = \sum_j W_{(j\longrightarrow i)} w_j(t-\varepsilon) = \sum_j \left\{ w_j(t-\varepsilon) T_{(j\longrightarrow i)} A_{(j\longrightarrow i)} + w_i(t-\varepsilon) T_{(i\longrightarrow j)} \left(1 - A_{(i\longrightarrow j)}\right) \right\} \tag{37}$$

we have two terms in the right hand, the first is easy to interpret because is the probabilities jumping from j to j and the second term maybe little more difficult to interpret, but it is the probability just staying $w_i(t)$ if we already are in i not jumping, for this reason appear $(1 - A_{(i\longrightarrow j)})$ we neglected the probability to jumping.

we need to assume $T_{(i\longrightarrow j)}$ and $A_{(i\longrightarrow j)}$ have to be normalized then we can rewrite **Equ.** (37) as

$$w_i(t) = w_i(t-\varepsilon) + \sum_j \left\{ w_j(t-\varepsilon) T_{(j\longrightarrow i)} A_{(j\longrightarrow i)} + w_i(t-\varepsilon) T_{(i\longrightarrow j)} A_{(i\longrightarrow j)} \right\} \tag{38}$$

now if we take the limit when time a large quantity says infinite the PDF becomes to be constant, therefor **Equ.** (38) can be written as

$$0 = \sum_j \left\{ w_j T_{(j\longrightarrow i)} A_{(j\longrightarrow i)} + w_i T_{(i\longrightarrow j)} A_{(i\longrightarrow j)} \right\}$$

$$w_j T_{(j\longrightarrow i)} A_{(j\longrightarrow i)} + w_i T_{(i\longrightarrow j)} A_{(i\longrightarrow j)} = 0$$

or

$$\frac{w_i}{w_j} = \frac{T_{(j\longrightarrow i)} A_{(j\longrightarrow i)}}{T_{(i\longrightarrow j)} A_{(i\longrightarrow j)}} \tag{39}$$

the problem still do not finish because we do not what it exactly $T_{(\ldots)}$ and $A_{(\ldots)}$, so we can do some restriction like that the transition probability has a reversal symmetric property that meaning $T_{(j\longrightarrow i)} = T_{(i\longrightarrow j)}$

$$\frac{w_i}{w_j} = \frac{A_{(j\longrightarrow i)}}{A_{(i\longrightarrow j)}} \quad \Rightarrow \quad \textbf{Metropolis algorithm} \tag{40}$$

this algorithm has an advantage in the discussions VMCM we assume that wave functions are normalized but to do this we need to find integral of norm and obviously is very time-consuming, but the **Equ.** (40) we do not need because this factor is canceled.

The next steps it is illustrated how works **Equ.** (40), the answer is in the fact we always want to maximize the probability the better way to see it thinking in a Gaussian distribution **Fig.** 6 the value central is which maximized the probability, in the other hand also we need to choose the value of $A_{(j\longrightarrow i)}$ it is actually computational Metropolis algorithm did.

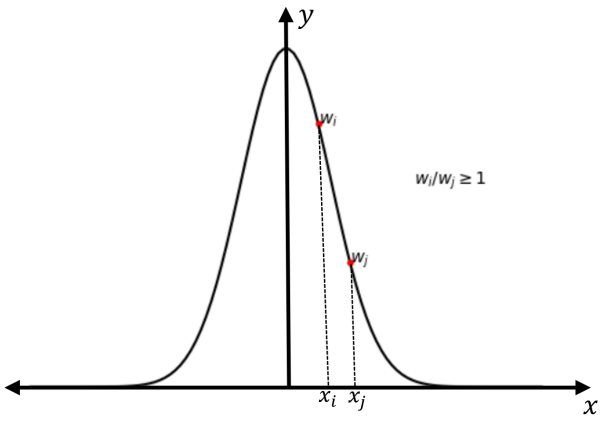


Figure 6: Illustration of Metropolis algorithm

normally this values it is chosen $A_{(j\longrightarrow i)} = 1$ and then we are going to have $\frac{A_{(j\longrightarrow i)}}{A_{(i\longrightarrow j)}} \geq 1$ and $\frac{w_i}{w_j} \geq 1$. It gives us the recipe to maximize the PDF.

2.3.2 Importance Sampling

The important sampling is a way to improve the MCMa, We need to know more about the transition probability that was mention because this allow us to find a better selection (see **Equ.** (39)) in jumping from w_j to w_i. This approach is based on the Fokker-Planck equation and the Langevin equation for generating a trajectory in coordinate space.

For a diffusion process characterized by a time-dependent probability (transition probability) density $P(x,t)$ in one dimension the Fokker-Planck equation for one particle or walker is given

$$\frac{\partial P}{\partial t} = D\frac{\partial}{\partial x}\left(\frac{\partial}{\partial x} - F\right)P(x,t), \tag{41}$$

where is F a drift quantum force term and contained derivative of the trial wave function $\mathbf{F} = 2\frac{1}{\Psi_T}\nabla\Psi_T$, and D is the diffusion coefficient.

The new positions in coordinate space are given as the solutions of the Langevin equation using Euler's method, and form the Langevin equation we can show

$$\frac{\partial x(t)}{\partial t} = DF(x(t)) + \eta,$$

where η is a random variable, then this give us a new position

$$y = x + DF(x)\Delta t + \xi\sqrt{\Delta t}, \tag{42}$$

where ξ is Gaussian random variable and Δt is a chosen time step. the quantity D can be found trough the so called fluctuation-dissipation theorem and it is given $D = 1/2$ and other important aspect we have to be into account the time here it just a parameter in the best option to choose for a good convergence it a values between $\Delta t \in [0.001, 0.01]$.

The solution the Fokker-Planck equation give us a transition probability which has as a solution the Green's function

$$G(y,x,\Delta t) = \frac{1}{(4\pi D\Delta t)^{3N/2}}\exp\left(-(y-x-D\Delta t F(x))^2/4D\Delta t\right) \tag{43}$$

which means we have found a transition probability ($T(...)$) for the Metropolis algorithm, so now we have

$$q(y,x) = \frac{G(x,y,\Delta t)|\Psi_T(y)|^2}{G(y,x,\Delta t)|\Psi_T(x)|^2} \tag{44}$$

where q can be interpret like a reason between probability distribution to be in x and y respectively. look back in the discreet case we have

$$q(y,x) \equiv \frac{T_{(i\longrightarrow j)}}{T_{(j\longrightarrow i)}}\frac{w_i}{w_j} = \frac{A_{(j\longrightarrow i)}}{A_{(i\longrightarrow j)}} \tag{45}$$

and

$$T_{(k\longrightarrow l)} \equiv G(x_k, x_l, \Delta t) = \frac{1}{(4\pi D\Delta t)^{3N/2}} \exp\left(-(x_k - x_l - D\Delta t F(x))^2/4D\Delta t\right)$$

we have found a Importance sampling algorithm and then we can use this new condition in ordered to find a Monte Carlo Importance Sampling algorithm (MCISa). On the other hand it is not difficult to see the improvement is coming for the fact that has walker has a restriction where can jumping (see **Equ.** (42)) and obviously for the knowledge transition probability.

3 Developing the project

3.1 Part a): Local energy

Find the analytic expressions for the local energy

$$E_L(\mathbf{r}) = \frac{1}{\Psi_T(\mathbf{r})} H\Psi_T(\mathbf{r}), \tag{46}$$

for the above trial wave function of **Equ.** (5) and defined by the terms in Eqs. (6) and (7).
Find first the local energy the case with only the harmonic oscillator potential, that is we set $a = 0$ and discard totally the two-body potential.

Use first that $\beta = 1$ and find the relevant local energies in one, two and three dimensions for one and N particles with the same mass. Compute also the analytic expression for the drift force to be used in importance sampling

$$F = \frac{2\nabla\Psi_T}{\Psi_T}. \tag{47}$$

Find first the equivalent expressions for the just the harmonic oscillator part in one, two and three dimensions with $\beta = 1$.

Our next step involves the calculation of local energy for the full problem in three dimensions. The tricky part is to find an analytic expressions for the derivative of the trial wave function

$$\frac{1}{\Psi_T(\mathbf{r})}\sum_i^N \nabla_i^2 \Psi_T(\mathbf{r}),$$

with the above trial wave function of Eq. (5). We rewrite

$$\Psi_T(\mathbf{r}) = \Psi_T(\mathbf{r}_1,\mathbf{r}_2,\dots\mathbf{r}_N,\alpha,\beta) = \left[\prod_i g(\alpha,\beta,\mathbf{r}_i)\right]\left[\prod_{j<k} f(a,|\mathbf{r}_j-\mathbf{r}_k|)\right],$$

as

$$\Psi_T(\mathbf{r}) = \left[\prod_i g(\alpha,\beta,\mathbf{r}_i)\right]\exp\left(\sum_{j<k} u(r_{jk})\right)$$

where we have defined $r_{ij} = |\mathbf{r}_i - \mathbf{r}_j|$ and

$$f(r_{ij}) = \exp\big(u(r_{ij})\big),$$

with $u(r_{ij}) = \ln f(r_{ij})$. We have also

$$g(\alpha,\beta,\mathbf{r}_i) = \exp\left[-\alpha(x_i^2+y_i^2+\beta z_i^2)\right] = \phi(\mathbf{r}_i).$$

Show that the first derivative for particle k is

$$\nabla_k\Psi_T(\mathbf{r}) = \nabla_k\phi(\mathbf{r}_k)\left[\prod_{i\neq k}\phi(\mathbf{r}_i)\right]\exp\left(\sum_{j<m} u(r_{jm})\right) + \left[\prod_i \phi(\mathbf{r}_i)\right]\exp\left(\sum_{j<m} u(r_{jm})\right)\sum_{l\neq k}\nabla_k u(r_{kl}),$$

and find the final expression for our specific trial function. The expression for the second derivative is (show this)

$$\frac{1}{\Psi_T(\mathbf{r})}\nabla_k^2\Psi_T(\mathbf{r}) = \frac{\nabla_k^2\phi(\mathbf{r}_k)}{\phi(\mathbf{r}_k)} + 2\frac{\nabla_k\phi(\mathbf{r}_k)}{\phi(\mathbf{r}_k)}\left(\sum_{j\neq k}\frac{(\mathbf{r}_k-\mathbf{r}_j)}{r_{kj}}u'(r_{kj})\right) + \sum_{i\neq k}\sum_{j\neq k}\frac{(\mathbf{r}_k-\mathbf{r}_i)(\mathbf{r}_k-\mathbf{r}_j)}{r_{ki}r_{kj}}u'(r_{ki})u'(r_{kj}) + \sum_{j\neq k}\left(u''(r_{kj})+\frac{2}{r_{kj}}u'(r_{kj})\right).$$

Use this expression to find the final second derivative entering the definition of the local energy. You need to get the analytic expression for this expression using the harmonic oscillator wave functions and the correlation term defined in the project.

Note: In parts 1b, 1c, 1d, 1e and 1f you will develop all computational ingredients needed by studying only the non-interacting case. We add the repulsive interaction in the final two parts, 1g and 1h. The reason for doing so is that we can develop all programming ingredients and compare our results against exact analytical results.

Using

$$E_L = \frac{1}{\Psi_T(r)} H\Psi_T(r)$$

with

$$H = \sum_i^N \left(-\frac{\hbar^2}{2m}\nabla_i^2 + V_{ext}(r_i) \right)$$

Spherical traps $\beta = 1$

$$g(\alpha,\beta,r_i) = exp\{-\alpha(x_i^2 + y_i^2 + z_i^2)\}$$

$$\Psi_T(r) = \prod_i g(\alpha,\beta,r_i)$$

$$H = \sum_i^N \left(-\frac{\hbar^2}{2m}\nabla_i^2 + \frac{1}{2}m\omega_{h_0}^2 r^2 \right)$$

3.1.1 Calculating the local energy

A. 1D for one particle

$$\Psi_T(r) = g(\alpha,\beta,x) = e^{-\alpha x^2}$$

$$H = -\frac{\hbar^2}{2m}\nabla_i^2 + \frac{1}{2}m\omega_{h_0}^2 x^2$$

then

$$E_L = e^{\alpha x^2}\left\{ -\frac{\hbar^2}{2m}\nabla^2 e^{-\alpha x^2} + \frac{1}{2}m\omega_{h_0}^2 x^2 e^{-\alpha x^2} \right\}$$

Solving:

$$\nabla^2 e^{-\alpha x^2} = (4\alpha^2 x^2 - 2\alpha)e^{-\alpha x^2}$$

Replacing

$$E_L = -\frac{\hbar^2}{m}(2\alpha^2 x^2 - \alpha) + \frac{1}{2}m\omega_{h_0}^2 x^2$$

or in natural units

$$E_L = \left(\frac{1}{2} - 2\alpha^2\right)x^2 + \alpha \tag{48}$$

B. 2D for one particle

$$\Psi_T(r) = g(\alpha,\beta,x) = e^{-\alpha(x^2+y^2)}$$

$$H = -\frac{\hbar^2}{2m}\nabla_i^2 + \frac{1}{2}m\omega_{h_0}^2(x^2+y^2)$$

then

$$E_L = e^{\alpha(x^2+y^2)}\left\{-\frac{\hbar^2}{2m}\nabla^2 e^{-\alpha(x^2+y^2)} + \frac{1}{2}m\omega_{h_0}^2(x^2+y^2)e^{-\alpha x^2}\right\}$$

Solving:

$$\nabla^2 e^{-\alpha(x^2+y^2)} = \left[4\alpha^2(x^2+y^2) - 4\alpha\right] e^{-\alpha(x^2+y^2)}$$

Replacing

$$E_L = -\frac{2\hbar^2}{m}\left[\alpha^2(x^2+y^2) - \alpha\right] + \frac{1}{2}m\omega_{h_0}^2(x^2+y^2)$$

or in natural units

$$E_L = \left(\frac{1}{2} - 2\alpha^2\right)(x^2+y^2) + 2\alpha \tag{49}$$

C. 3D for one particle Following the above calculations

$$\Psi_T(r) = g(\alpha,\beta,x) = e^{-\alpha(x^2+y^2+z^2)}$$

$$H = -\frac{\hbar^2}{2m}\nabla_i^2 + \frac{1}{2}m\omega_{h_0}^2(x^2+y^2+z^2)$$

Solving:

$$\nabla^2 e^{-\alpha(x^2+y^2+z^2)} = \left[4\alpha^2(x^2+y^2+z^2) - 6\alpha\right] e^{-\alpha(x^2+y^2+z^2)}$$

Replacing

$$E_L = -\frac{\hbar^2}{m}\left[2\alpha^2(x^2+y^2+z^2) - 3\alpha\right] + \frac{1}{2}m\omega_{h_0}^2(x^2+y^2+z^2)$$

or in natural units

$$E_L = \left(\frac{1}{2} - 2\alpha^2\right)(x^2+y^2+z^2) + 3\alpha \tag{50}$$

Using **Eqs.** (48), (49), and (50), it is possible to write the generalized equation:

$$E_L = \left(\frac{1}{2} - 2\alpha^2\right)\sum_i^N r_i^2 + \alpha N D \tag{51}$$

where N is number of particles and D is the dimension (D = 1, 2 or 3), r_i^2 depend of the dimension:

$$r_i^2 = x_i^2 \longleftrightarrow 1D$$

$$r_i^2 = x_i^2 + y^2 \longleftrightarrow 2D$$

$$r_i^2 = x_i^2 + y^2 + z^2 \longleftrightarrow 3D$$

D. 1D for N particles

$$\Psi_T(r) = \prod_i^N e^{-\alpha x_i^2} = exp\left(\sum_i^N (-\alpha x_i^2)\right)$$

$$E_L = \sum_i^N e^{\alpha x_i^2}\left\{-\frac{\hbar^2}{2m}\nabla_i^2 e^{-\alpha x_i^2} + \frac{1}{2}m\omega_{h_0}^2 x_i^2 e^{-\alpha x_i^2}\right\}$$

$$E_L = \sum_i^N \left\{-\frac{\hbar^2}{m}(2\alpha^2 x_i^2 - \alpha) + \frac{1}{2}m\omega_{h_0}^2 x_i^2\right\}$$

Using natural units

$$E_L = \sum_i^N \left\{\left(\frac{1}{2} - 2\alpha^2\right)x_i^2 + \alpha\right\} \tag{52}$$

E. 2D for N particles

$$E_L = \sum_i^N e^{\alpha(x_i^2+y_i^2)}\left\{-\frac{\hbar^2}{2m}\nabla_i^2 e^{-\alpha(x_i^2+y_i^2)} + \frac{1}{2}m\omega_{h_0}^2(x_i^2+y_i^2)e^{-\alpha(x_i^2+y_i^2)}\right\}$$

$$E_L = \sum_i^N \left\{-\frac{2\hbar^2}{m}\left[\alpha^2(x_i^2+y_i^2) - \alpha\right] + \frac{1}{2}m\omega_{h_0}^2(x_i^2+y_i^2)\right\}$$

in natural units

$$E_L = \sum_i^N \left\{\left(\frac{1}{2} - 2\alpha^2\right)(x_i^2+y_i^2) + 2\alpha\right\} \tag{53}$$

F. 3D for N particles

$$E_L = \sum_i^N \left\{-\frac{\hbar^2}{m}\left[2\alpha^2(x_i^2+y_i^2+z_i^2) - 3\alpha\right] + \frac{1}{2}m\omega_{h_0}^2(x_i^2+y_i^2+z_i^2)\right\}$$

in natural units

$$E_L = \sum_i^N \left\{\left(\frac{1}{2} - 2\alpha^2\right)(x_i^2+y_i^2+z_i^2) + 3\alpha\right\} \tag{54}$$

3.1.2 Drift force

$$F = \frac{2\nabla\Psi_T}{\Psi_T}$$

A. 1D for one particle

$$F = \frac{2\nabla e^{-\alpha x^2}}{e^{-\alpha x^2}}$$

$$\nabla e^{-\alpha x^2} = -2\alpha x e^{-\alpha x^2}\hat{i}$$

$$F = \frac{2(-2\alpha x)e^{-\alpha x^2}}{e^{-\alpha x^2}}\hat{i} = -4\alpha x\hat{i} \tag{55}$$

B. 2D for one particle

$$\nabla e^{-\alpha(x^2+y^2)} = -2\alpha(x^2\hat{i}+y^2\hat{j})e^{-\alpha(x^2+y^2)}$$

$$F = -4\alpha(x^2\hat{i}+y^2\hat{j}) \tag{56}$$

C. 3D for one particle

$$\nabla e^{-\alpha(x^2+y^2+z^2)} = -2\alpha(x^2\hat{i}+y^2\hat{j}+z^2\hat{k})e^{-\alpha(x^2+y^2+z^2)}$$

$$F = -4\alpha(x^2\hat{i}+y^2\hat{j}+z^2\hat{k}) \tag{57}$$

D. 1D for jth particle

$$F_j = \frac{2\nabla_j\Psi_T}{\Psi_T}$$

$$\Psi_T(r) = \prod_i^N e^{-\alpha x_i^2} = e^{-\alpha x_1^2}\dots e^{-\alpha x_j^2}\dots e^{-\alpha x_N^2}$$

$$\nabla_j\Psi_T = e^{-\alpha x_1^2}\dots\frac{\partial}{\partial x_j}e^{-\alpha x_j^2}\dots e^{-\alpha x_N^2} = -2\alpha x_j e^{-\alpha x_1^2}\dots e^{-\alpha x_j^2}\dots e^{-\alpha x_N^2}\hat{i} = -2\alpha x_j\Psi_T\hat{i}$$

$$F_j = -4\alpha x_j\hat{i} \tag{58}$$

E. 2D for jth particle

$$\nabla_j\Psi_T = e^{-\alpha(x_1^2+y_1^2)}\dots\frac{\partial}{\partial x_j}e^{-\alpha(x_j^2+y_j^2)}le^{-\alpha(x_N^2+y_N^2)} = -2\alpha(x_j^2+y_j^2)e^{-\alpha(x_1^2+y_1^2)}\dots e^{-\alpha(x_j^2+y_j^2)}\dots e^{-\alpha(x_N^2+y_N^2)}\hat{i}$$

$$\nabla_j \Psi_T = -2\alpha(x_j + y_j)\Psi_T \hat{i}$$

$$F_j = -4\alpha(x_j\hat{i} + y_j\hat{j}) \tag{59}$$

F. 3D for jth particle

$$F_j = -4\alpha(x_j\hat{i} + y_j\hat{j} + z_j\hat{k}) \tag{60}$$

for all particles is just to do sum over all particles, for N particles an 3D

$$F_j = \sum_i^N -4\alpha(x_j\hat{i} + y_j\hat{j} + z_j\hat{k}) \tag{61}$$

3.1.3 Show that the first derivative for particle k is

$$\Psi_T(r) = \Psi_T(r_1, \ldots, r_N, \alpha, \beta) = \left[\prod_i g(\alpha, \beta, r_i)\right]\left[\prod_{j<k} f(a, r_j - r_k)\right]$$

$$\Psi_T(r) = \left[\prod_i \phi(r_i)\right]\left[exp\left\{\sum_{i<j} u(r_{ij})\right\}\right]$$

$$\nabla_k \Psi_T(r) = \nabla_k \left[\prod_i \phi(r_i)\right]\left[exp\left\{\sum_{i<j} u(r_{ij})\right\}\right] \tag{62}$$

$$\nabla_k \Psi_T(r) = \underbrace{\left[\nabla_k \prod_i \phi(r_i)\right]}_{(i)}\left[exp\left\{\sum_{i<j} u(r_{ij})\right\}\right] + \left[\prod_i \phi(r_i)\right]\underbrace{\left[\nabla_k exp\left\{\sum_{i<j} u(r_{ij})\right\}\right]}_{(ii)} \tag{63}$$

(i)

$$\nabla_k \prod_i \phi(r_i) = \nabla_k \left[\phi(r_1)\ldots\phi(r_k)\ldots\phi(r_N)\right] \tag{64}$$

$$= \nabla_k \phi(r_k)\left[\prod_{i\neq k}^N \phi(r_i)\right] \tag{65}$$

(ii)

$$\nabla_k exp\left\{\sum_{j<k} u(r_{jk})\right\} = \nabla_k exp\left\{u(r_{12})+u(r_{13})+\ldots+u(r_{23})+u(r_{24})+\ldots+u(r_{kj})+\ldots+u(r_{kN})+\ldots+u(r_N\right.$$

$$= \nabla_k e^{u(r_{12})} e^{u(r_{13})} \ldots e^{u(r_{23})} \ldots \underbrace{e^{u(r_{kj})} \ldots e^{u(r_{kN})}} \ldots e^{u(r_{N-1,N})}$$

$$= \nabla_k e^{u(r_{kj})} \ldots e^{u(r_{kN})} \left\{e^{u(r_{12})} e^{u(r_{13})} \ldots e^{u(r_{23})} \ldots e^{u(r_{N-1,N})}\right\}$$

$$= \left\{\nabla_k u(r_{kj}) + \ldots + \nabla_k u(r_{kN})\right\} \left\{e^{u(r_{12})} e^{u(r_{13})} \ldots e^{u(r_{23})} \ldots e^{u(r_{kj})} \ldots e^{u(r_{kN})} \ldots e^{u(r_{N-1,N})}\right\}$$

$$= \left\{\nabla_k u(r_{kj}) + \ldots + \nabla_k u(r_{kN})\right\} exp\left\{\sum_{i<j} u(r_{ij})\right\}$$

$$\nabla_k exp\left\{\sum_{j<k} u(r_{jk})\right\} = exp\left\{\sum_{i<j} u(r_{ij})\right\} \sum_{j\neq k}^{N} \nabla_k u(r_{kj}) \tag{66}$$

therefore, replacing in **Equ.** (63) equation

$$\nabla_k \Psi_T(r) = \nabla_k \phi(r_k) \left[\prod_{i\neq k}^{N} \phi(r_i)\right] \left[exp\left\{\sum_{i<j} u(r_{ij})\right\}\right] + \left[\prod_i \phi(r_i)\right] exp\left\{\sum_{i<j} u(r_{ij})\right\} \sum_{j\neq k}^{N} \nabla_k u(r_{kj}) \tag{67}$$

second derivate

$$\frac{1}{\Psi_T(r)} \nabla_k^2 \Psi_T(r) \qquad \longrightarrow \qquad \frac{1}{\Psi_T(r)} \nabla_k \left(\nabla_k \Psi_T(r)\right)$$

$$\frac{1}{\Psi_T(r)} \nabla_k \left(\nabla_k \Psi_T(r)\right) = \frac{1}{\Psi_T(r)} \left\{ \underbrace{\left[\nabla_k \left(\nabla_k \phi(r_k) \left[\prod_{i\neq k}^{N} \phi(r_i)\right]\right)\right] \left[exp\left\{\sum_{i<j} u(r_{ij})\right\}\right]}_{(i)} + \right.$$

$$+ \underbrace{\nabla_k \phi(r_k) \left[\prod_{i\neq k}^{N} \phi(r_i)\right] \left[\nabla_k exp\left\{\sum_{i<j} u(r_{ij})\right\}\right]}_{(ii)} + \underbrace{\left[\nabla_k \prod_i \phi(r_i)\right] exp\left\{\sum_{i<j} u(r_{ij})\right\} \sum_{j\neq k}^{N} \nabla_k u(r_{kj})}_{(iii)} +$$

$$\left. + \underbrace{\left[\prod_i \phi(r_i)\right] \left[\nabla_k exp\left\{\sum_{i<j} u(r_{ij})\right\}\right] \sum_{j\neq k}^{N} \nabla_k u(r_{kj})}_{(iv)} + \underbrace{\left[\prod_i \phi(r_i)\right] exp\left\{\sum_{i<j} u(r_{ij})\right\} \left[\nabla_k \sum_{j\neq k}^{N} \nabla_k u(r_{kj})\right]}_{(v)} \right\}$$

(i)

$$\frac{\nabla_k^2 \phi(r_k) \left[\prod_{i\neq k}^{N} \phi(r_i)\right] \left[exp\left\{\sum_{i<j} u(r_{ij})\right\}\right]}{\left[\prod_i \phi(r_i)\right] \left[exp\left\{\sum_{i<j} u(r_{ij})\right\}\right]} = \frac{\nabla_k^2 \phi(r_k) \left[\prod_{i\neq k}^{N} \phi(r_i)\right]}{\left[\prod_i \phi(r_i)\right]} = \frac{\nabla_k^2 \phi(r_k) \left[\prod_{i\neq k}^{N} \phi(r_i)\right]}{\phi(r_k) \left[\prod_{i\neq k}^{N} \phi(r_i)\right]}$$

$$= \frac{\nabla_k^2 \phi(r_k)}{\phi(r_k)} \tag{68}$$

(ii)

$$\frac{\nabla_k \phi(r_k) \left[\prod_{i\neq k}^{N} \phi(r_i)\right] \left[\nabla_k exp\left\{\sum_{i<j} u(r_{ij})\right\}\right]}{\left[\prod_i \phi(r_i)\right] \left[exp\left\{\sum_{i<j} u(r_{ij})\right\}\right]} = \frac{\nabla_k \phi(r_k)}{\phi(r_k)} \frac{exp\left\{\sum_{i<j} u(r_{ij})\right\} \sum_{j\neq k}^{N} \nabla_k u(r_{kj})}{exp\left\{\sum_{i<j} u(r_{ij})\right\}}$$

$$= \frac{\nabla_k \phi(r_k)}{\phi(r_k)} \sum_{j\neq k}^{N} \nabla_k u(r_{kj}) \tag{69}$$

(iii)

$$\frac{\left[\nabla_k \prod_i \phi(r_i)\right] exp\left\{\sum_{i<j} u(r_{ij})\right\} \sum_{j\neq k}^{N} \nabla_k u(r_{kj})}{\left[\prod_i \phi(r_i)\right] \left[exp\left\{\sum_{i<j} u(r_{ij})\right\}\right]} = \frac{\nabla_k \phi(r_k) \left[\prod_{i\neq k}^{N} \phi(r_i)\right]}{\prod_i \phi(r_i)} \sum_{j\neq k}^{N} \nabla_k u(r_{kj})$$

$$= \frac{\nabla_k \phi(r_k)}{\phi(r_k)} \sum_{j\neq k}^{N} \nabla_k u(r_{kj}) \tag{70}$$

(iv)

$$\frac{\left[\prod_i \phi(r_i)\right] \left[\nabla_k exp\left\{\sum_{i<j} u(r_{ij})\right\}\right] \sum_{j\neq k}^{N} \nabla_k u(r_{kj})}{\left[\prod_i \phi(r_i)\right] \left[exp\left\{\sum_{i<j} u(r_{ij})\right\}\right]} = \sum_{i\neq k}^{N} \nabla_k u(r_{ki}) \sum_{j\neq k}^{N} \nabla_k u(r_{kj}) \tag{71}$$

(v)

$$\frac{\left[\prod_i \phi(r_i)\right] exp\left\{\sum_{i<j} u(r_{ij})\right\} \left[\nabla_k \sum_{j\neq k}^{N} \nabla_k u(r_{kj})\right]}{\left[\prod_i \phi(r_i)\right] \left[exp\left\{\sum_{i<j} u(r_{ij})\right\}\right]} = \sum_{j\neq k}^{N} \nabla_k^2 u(r_{kj}) \tag{72}$$

replacing

$$\frac{1}{\Psi_T(r)} \nabla_k \left(\nabla_k \Psi_T(r)\right) = \frac{\nabla_k^2 \phi(r_k)}{\phi(r_k)} + 2\frac{\nabla_k \phi(r_k)}{\phi(r_k)} \sum_{j\neq k}^{N} \underbrace{\nabla_k u(r_{kj})}_{} + \sum_{i\neq k}^{N} \nabla_k u(r_{ki}) \sum_{j\neq k}^{N} \nabla_k u(r_{kj}) + \sum_{j\neq k}^{N} \underbrace{\nabla_k^2 u(r_{kj})}_{} \tag{73}$$

Solving $\nabla_k u(r_{kj})$ and $\nabla_k^2 u(r_{kj})$

$$r_{kj} = \sqrt{|x_k - x_j|^2 + |y_k - y_j|^2 + |z_k - z_j|^2}$$

Applying the chain rule

$$\nabla_k u(r_{kj}) = \left(\frac{\partial}{\partial x_k}\hat{i} + \frac{\partial}{\partial y_k}\hat{j} + \frac{\partial}{\partial z_k}\hat{k}\right) u(r_{kj})$$

$$\frac{\partial}{\partial x_k}u(r_{kj})\hat{i} = \frac{\partial u(r_{kj})}{\partial r_{kj}}\frac{\partial r_{kj}}{\partial x_k}\hat{i} = \frac{u'(r_{kj})|x_k - x_j|}{r_{kj}}\hat{i}$$
$$\frac{\partial}{\partial x_k}u(r_{kj})\hat{j} = \frac{\partial u(r_{kj})}{\partial r_{kj}}\frac{\partial r_{kj}}{\partial y_k}\hat{j} = \frac{u'(r_{kj})|y_k - y_j|}{r_{kj}}\hat{j}$$
$$\frac{\partial}{\partial x_k}u(r_{kj})\hat{k} = \frac{\partial u(r_{kj})}{\partial r_{kj}}\frac{\partial r_{kj}}{\partial z_k}\hat{i} = \frac{u'(r_{kj})|z_k - z_j|}{r_{kj}}\hat{k}$$

therefore

$$\nabla_k u(r_{kj}) = \frac{u'(r_{kj})}{r_{kj}}\left\{|x_k - x_j|r_{kj}\hat{i} + |y_k - y_j|r_{kj}\hat{j} + |z_k - z_j|r_{kj}\hat{k}\right\} = \frac{u'(r_{kj})}{r_{kj}}\left(\vec{r}_k - \vec{r}_j\right) \qquad (74)$$

second derivate (∇^2)

$$\nabla_k^2 u(r_{kj}) \qquad \longrightarrow \qquad \nabla_k^2 = \frac{\partial^2}{\partial x_k^2} + \frac{\partial^2}{\partial y_k^2} + \frac{\partial^2}{\partial z_k^2}$$

$$\Rightarrow \frac{\partial^2}{\partial x_k^2}u(r_{kj}) = \frac{\partial^2 u(r_{kj})}{\partial r_{kj}^2}\left(\frac{\partial r_{kj}}{\partial x_k}\right)^2 + \frac{\partial u(r_{kj})}{\partial r_{kj}}\frac{\partial^2 r_{kj}}{\partial x_k^2} = u''(r_{kj})\frac{(x_k - x_j)^2}{r_{kj}^2} + u'(r_{kj})\frac{\partial^2 r_{kj}}{\partial x_k^2}$$
$$= u''(r_{kj})\frac{(x_k - x_j)^2}{r_{kj}^2} + u'(r_{kj})\left\{\frac{1}{r_{kj}} - \frac{(x_k - x_j)^2}{r_{kj}^3}\right\}$$

$$\Rightarrow \frac{\partial^2}{\partial y_k^2}u(r_{kj}) = u''(r_{kj})\frac{(y_k - y_j)^2}{r_{kj}^2} + u'(r_{kj})\left\{\frac{1}{r_{kj}} - \frac{(y_k - y_j)^2}{r_{kj}^3}\right\}$$

$$\Rightarrow \frac{\partial^2}{\partial z_k^2}u(r_{kj}) = u''(r_{kj})\frac{(z_k - z_j)^2}{r_{kj}^2} + u'(r_{kj})\left\{\frac{1}{r_{kj}} - \frac{(z_k - z_j)^2}{r_{kj}^3}\right\}$$

then

$$\nabla_k^2 u(r_{kj}) = \frac{u''(r_{kj})}{r_{kj}^2}\underbrace{\left\{(x_k - x_j)^2 + (y_k - y_j)^2 + (z_k - z_j)^2\right\}}_{r_{kj}^2} + u'(r_{kj})\left\{\frac{3}{r_{kj}} - \frac{(x_k - x_j)^2 + (y_k - y_j)^2 + (z_k - z_j)^2}{r_{kj}^3}\right\}$$

$$\nabla_k^2 u(r_{kj}) = u''(r_{kj}) + 2\frac{u'(r_{kj})}{r_{kj}} \qquad (75)$$

therefore

$$\frac{1}{\Phi_T(r)}\nabla_k^2\Phi_T(r) = \frac{\nabla_k^2\phi(r_k)}{\phi(r_k)} + 2\frac{\nabla_k\phi(r_k)}{\phi(r_k)}\sum_{j\neq k}^{N}\frac{u'(r_{kj})}{r_{kj}}\left(\vec{r}_k-\vec{r}_j\right) + \sum_{i\neq k}^{N}\sum_{j\neq k}^{N}\frac{(\vec{r}_k-\vec{r}_i)}{r_{ki}}\frac{\left(\vec{r}_k-\vec{r}_j\right)}{r_{kj}}u'(r_{ri})u'(r_{rj}) + \sum_{j\neq k}^{N}\left\{u''(r_{kj})+2\frac{u'(r_{kj})}{r_{kj}}\right\}$$

analytical solution

$$\Rightarrow \frac{\nabla_k^2\phi(r_k)}{\phi(r_k)} \quad \longrightarrow \quad \frac{\partial}{\partial x_k}\left(\frac{\partial}{\partial x_k}\phi(r_k)\right) + \frac{\partial}{\partial y_k}\left(\frac{\partial}{\partial y_k}\phi(r_k)\right) + \frac{\partial}{\partial z_k}\left(\frac{\partial}{\partial z_k}\phi(r_k)\right)$$

$$\frac{\partial}{\partial x_k}\left(\frac{\partial}{\partial x_k}\phi(r_k)\right) = \frac{\partial}{\partial x_k}\left(-2\alpha x_k\phi(r_k)\right) = -2\alpha\left\{\phi(r_k)+x_k\frac{\partial}{\partial x_k}\phi(r_k)\right\} = (4\alpha^2x_k^2-2\alpha)\phi(r_k)$$
$$\frac{\partial}{\partial y_k}\left(\frac{\partial}{\partial y_k}\phi(r_k)\right) = (4\alpha^2y_k^2-2\alpha)\phi(r_k)$$
$$\frac{\partial}{\partial z_k}\left(\frac{\partial}{\partial z_k}\phi(r_k)\right) = (4\alpha^2\beta^2z_k^2-2\alpha\beta)\phi(r_k)$$

$$\frac{\nabla_k^2\phi(r_k)}{\phi(r_k)} = 4\alpha^2(x_k^2+y_k^2+\beta^2z_k^2)-4\alpha-2\alpha\beta \tag{76}$$

$$\Rightarrow \frac{\nabla_k\phi(r_k)}{\phi(r_k)} \quad \longrightarrow \quad \left(\frac{\partial}{\partial x_k}\hat{i}+\frac{\partial}{\partial y_k}\hat{j}+\frac{\partial}{\partial z_k}\hat{k}\right)e^{-\alpha(x_k^2+y_k^2+z_k^2)}$$

$$\frac{\nabla_k\phi(r_k)}{\phi(r_k)} = -2\alpha(x_k\hat{i}+y_k\hat{j}+\beta z_k\hat{k}) \tag{77}$$

$$\Rightarrow u'(r_{kj}) = \frac{\partial u(r_{kj})}{\partial r_{kj}}$$

using

$$u(r_{kj}) = Lnf(r_{kj}) = Ln\left(1-\frac{a}{r_k-r_j}\right)$$

$$\frac{\partial}{\partial r_{kj}}Lnf(r_{kj}) = \frac{\partial}{\partial r_{kj}}Ln\left(1-\frac{a}{\underbrace{r_k-r_j}_{r_{kj}}}\right) = \frac{a}{r_{kj}^2-ar_{kj}} \tag{78}$$

$$\Rightarrow u''(r_{kj}) = \frac{\partial}{\partial r_{kj}}\left(\frac{a}{r_{kj}^2 - a r_{kj}}\right) = -\frac{a}{(r_{kj}^2 - a r_{kj})^2}(2r_{kj} - a) = \frac{a(a-2r_{kj})}{r_{kj}^2(r_{kj}-a)^2} \tag{79}$$

LOCAL ENERGY

$$E_L = \sum_i^N \left\{ -\frac{\hbar^2}{2m}\frac{1}{\Psi_T(r)}\nabla_i^2 \Psi_T(r) + \frac{1}{2}m\omega_{h_o}^2 r_i^2 \right\} \tag{80}$$

We can generalize these equations for N particles and different dimensions (D):

Local energy

$$E_L = \left(\frac{1}{2} - 2\alpha^2\right)\sum_i^N r_i^2 + \alpha N D \tag{81}$$

Exact Energies

$$\bar{E} = \langle E_L \rangle = \left(\frac{1}{2} - 2\alpha^2\right)\frac{DN}{4\alpha} + \alpha N D \tag{82}$$

expectation value

$$\langle E_L^2 \rangle = N\left(\frac{1}{2} - 2\alpha^2\right)\cdot 2\frac{D(D+2)}{16\alpha^2} + \frac{ND^2}{2}\left(\frac{1}{2} - 2\alpha^2\right) + N(\alpha D)^2 \tag{83}$$

Variance

$$\sigma = \langle E_L^2 \rangle - \langle E_L \rangle^2 \tag{84}$$

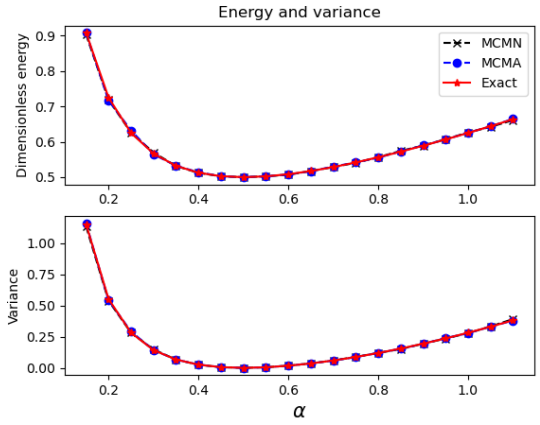
3.2 Part b): Developing the code

Write a Variational Monte Carlo program which uses standard Metropolis sampling and compute the ground state energy of a spherical harmonic oscillator ($\beta = 1$) with no interaction and one dimension. Use natural units and make an analysis of your calculations using both the analytic expression for the local energy and a numerical calculation of the kinetic energy using numerical derivation. Compare the CPU time difference. The only variational parameter is α. Perform these calculations for $N = 1$, $N = 10$, 100 and 500 atoms. Compare your results with the exact answer. Extend then your results to two and three dimensions and compare with the analytical results.

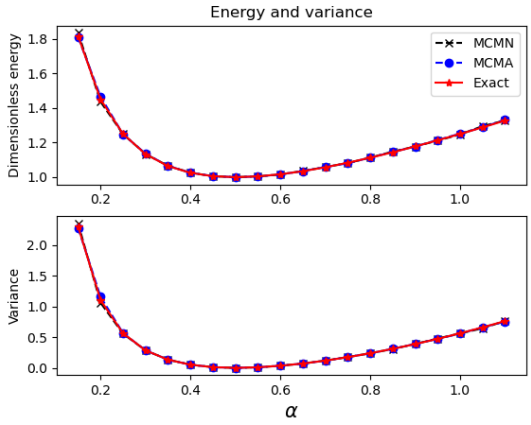
For this section we have took as model the code from the class. Taking advantage of the definition of wave function and local energy, we have introduced the generalize equations for each

case, **Eqs.** (46) and (81), as well as the generalize equations for the exact energy and variance from the **Eqs.** (82) and (84). You can check the code in Github - Part b.

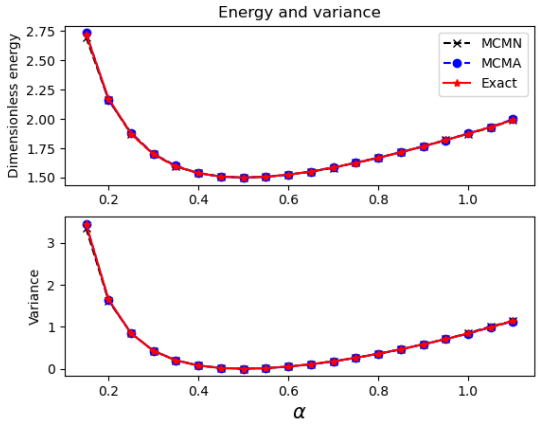
Here we are going to show the **Figs.** 7 and 8, which are the comparison between the exact solution, analytic and numeric solution. Where exact solution as its name implies the true solution, for the analytic solution we going to refer as solution where the local energy is calculated analytic, and finally the numeric solution is when the local energy is calculated numeric that meaning the kinetic energy is calculated numerically. As it was mentioned here we are using a MCMa.



(a) N = 1 and D = 1



(b) N = 1 and D = 2

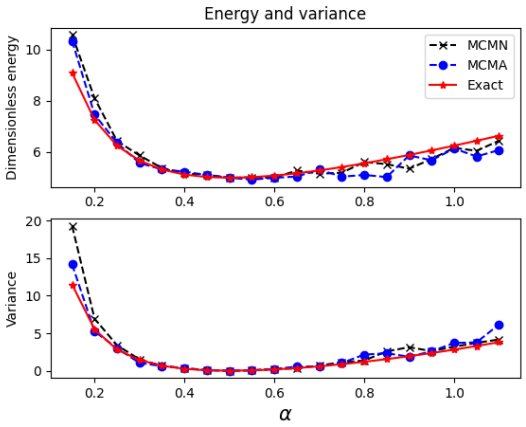


(c) N = 1 and D = 3

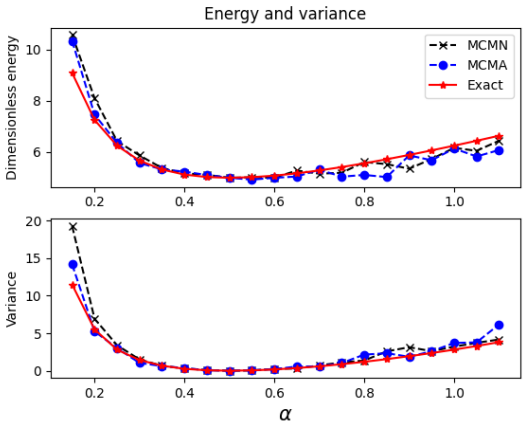
Figure 7: Comparison between the Monte Carlo using Metropolis algorithm for N=1 and D=1,2,3. Considering the exact (Exact), analytic (MCMA) and numerical calculation (MCMN).

The result below was calculate using 10^6 Monte Carlos cycles (MCc), but for the next calcu-

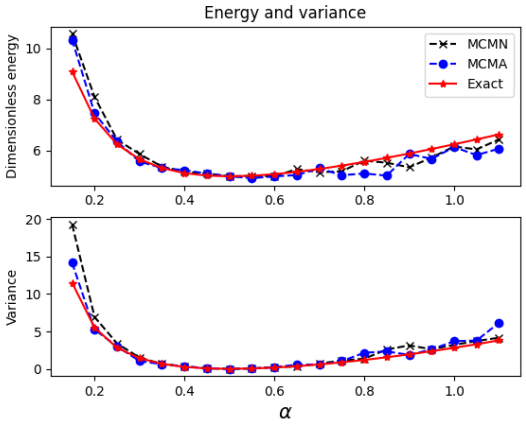
lations we have used a smaller MCc 10^3 because otherwise it becomes computationally very expensive even with process paralysis.



(a) N = 10 and D = 1



(b) N = 10 and D = 2

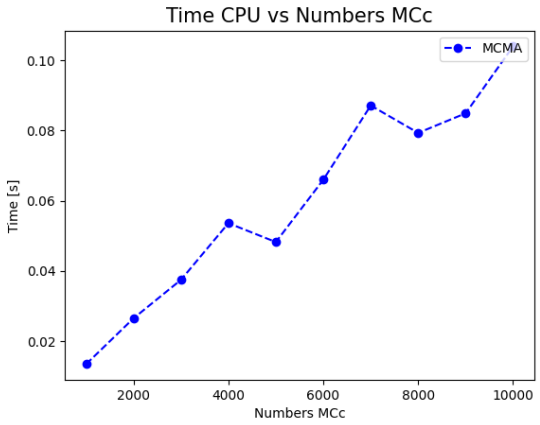


(c) N = 10 and D = 3

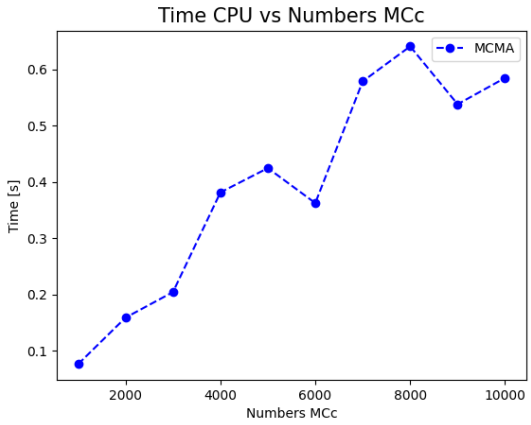
Figure 8: Comparison between the Monte Carlo using Metropolis algorithm for N=10 and D=1,2,3. Considering the exact (Exact), analytic (MCMA) and numerical (MCMN) calculation.

The **Figs.** 7 and 8 show us that results are good enough and as expected the more MCc we use better approximation will have.

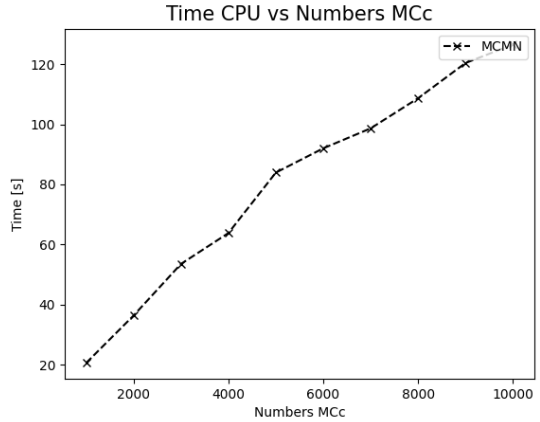
We can also study the time consumption as function of number MCc for N = 1, 10 and D = 1, the **Fig.** 9 show the result.



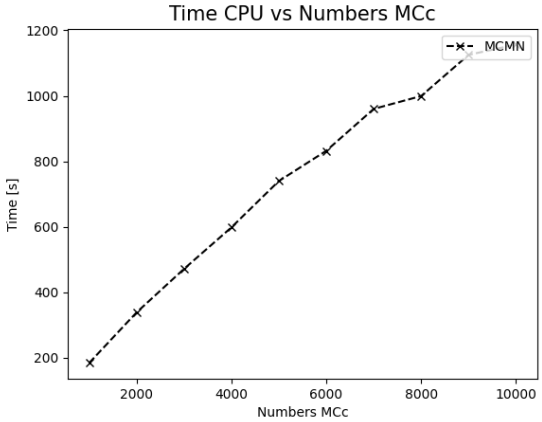
(a) N = 1 and D = 1



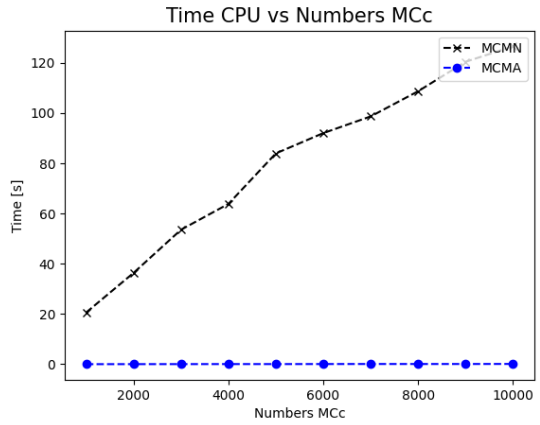
(b) N = 10 and D = 1



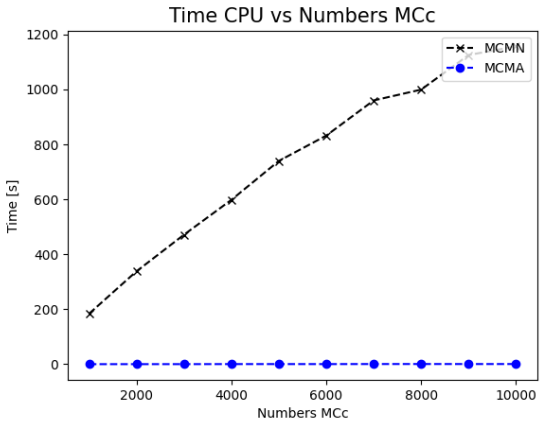
(c) N = 1 and D = 1



(d) N = 10 and D = 1



(e) Comparison for N = 1 and D = 1



(f) Comparison for N = 10 and D = 1

Figure 9: The figure shows the time consuming vs number MCc, for N = 1, 10 and D =1 using analytic (MCMA) (9a) and (9b) and numerical (MCMN) (9c) and (9d) calculation .

From **Fig.** 9, we can see that the numeric solution takes more time and as well if the number

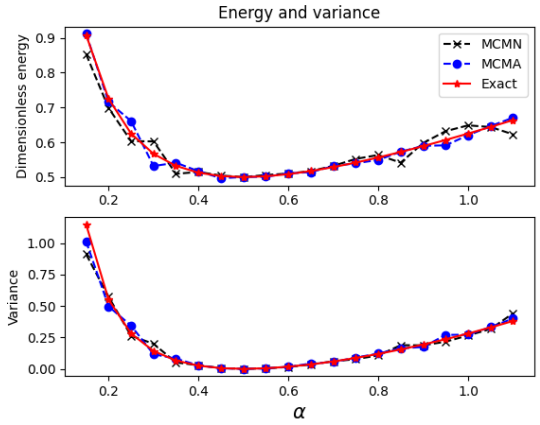
particles increase the time consuming increase very fast (see **Figs.** 9c and 9d). That mean, it is possible to get results from the numerical calculations for greater number particles or dimension but will take a lot of time.

For this part, we also have information without including numerical calculations. You can see it in github, section of results, for either energy or variance. There are included the calculations for number of particles 1 and 10 in different dimensions with MCc equal to 10^6.

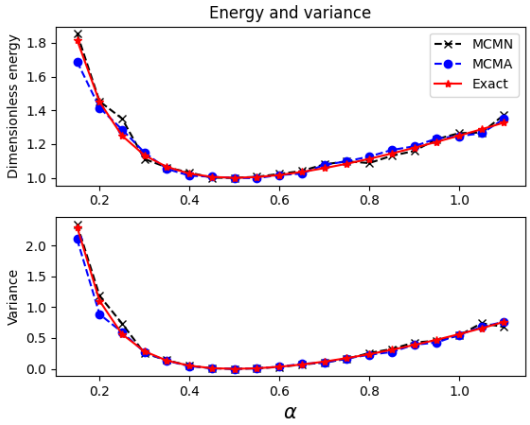
3.3 Part c): Adding Importance Sampling

We repeat part b), but now we replace the brute force Metropolis algorithm with importance sampling based on the Fokker-Planck and the Langevin equations. Discuss your results and comment on eventual differences between importance sampling and brute force sampling. Run the calculations for the one, two and three-dimensional systems only and without the repulsive potential. Study the dependence of the results as a function of the time step δt. Compare the results with those obtained under b) and comment eventual differences.
For this section we have added the importance sampling with the Fokker-Planck and Langevin equations. That mean, introduce the generalize drift force (see **Equ.** (61)). You can check the code in Github - Part c.

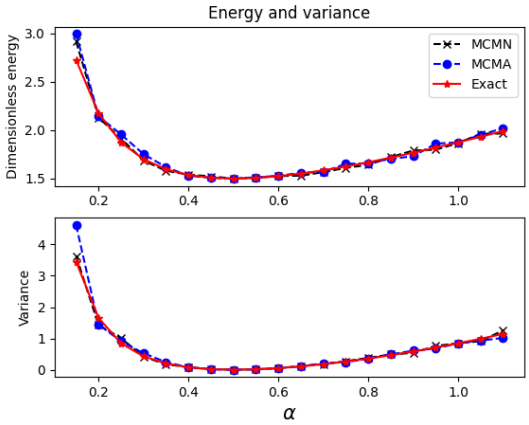
Adding import sampling as we mention **Section** 2.3.2 this allows us to chose a better way what point are accepted or neglected, in the **Fig.** 10 for number MCc equal to 10^4.



(a) N = 1 and D = 1



(b) N = 1 and D = 2

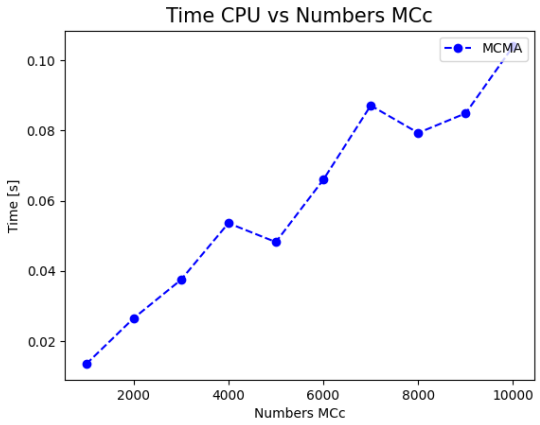


(c) N = 1 and D = 3

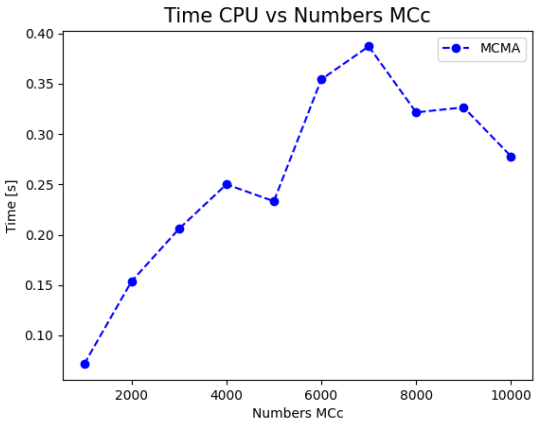
Figure 10: Comparison between the Monte Carlo using MCISa for N=1 and D=1,2,3. Considering the exact (Exact), analytic (MCMA) and numerical calculation (MCMN).

The **Fig.** 10 show again a good enough result, but the import point now it is that the number MCc is 10^2 times smaller that MCMa, but they could still be comparable (see **Figs.** 7 and 10). That meaning we will need less MCc to get the same result.

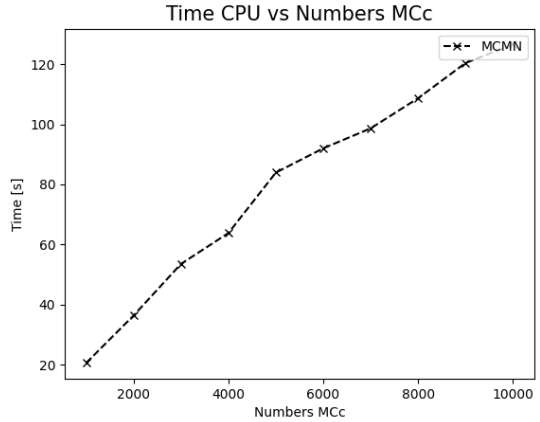
Other import aspect is the time consuming of course, in the **Fig.** 11 show the comparison between time consuming for MCMa and MCISa with N=1 and D=1 with the same number the Monte Carlos cycles MCc= 10^3.



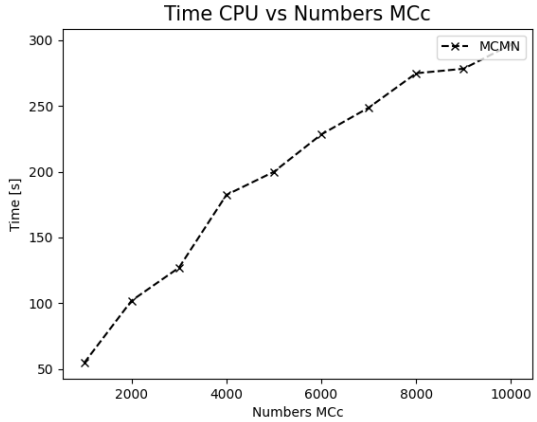
(a) N = 1 and D = 1



(b) N = 1 and D = 1



(c) N = 1 and D = 1



(d) N = 1 and D = 1

Figure 11: The figure shows the comparison between MCMa (11a) and (11c) and MCISa (11b) and (11d) for time consuming versus the number MCc, for N = 1 and D = 1 using analytic (MCMA) and numerical (MCMN) calculation .

As we expect the time consumption for MCISa is greater that MCMa because in MCISa is necessary to calculate the quantum force and this cause the time different. A final comment is that although MCISa need less MCc take more time, so we need to try for us system which method could be better.

Other import point in MCISa is the time step Δt (see **Equ.** (43)), the **Fig.** 12 show the energy, variance, time consuming behavior as time step for N = 1, D = 1 and MCc = 10^3.

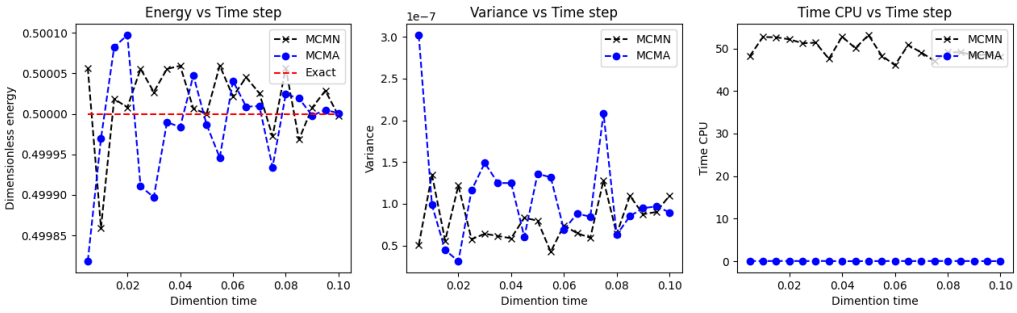


Figure 12: Energy, variance and time consuming vs time step for N = 1, D = 1 and MCc = 10^3.

In the figure **Fig.** 12 the behavior of energy tells us may be values which there are a better convergence, if we look carefully for values around $[0.04, 0.06]$ the energy is closer to the exact answer. For the variance shows us the result are good enough since its value is small for any time step and finally the time consuming It is something that was expected the numeric solution always will be spend more time, but as well we will not have much change when changing the time step.

Similar to the above part, We also have information without including numerical calculations. You can see it in github, section of results, for either energy or variance. There are included the calculations for number of particles 1 and 10 in different dimensions with MCc equal to 10^6.

3.4 Part d): Finding the best parameter(s)

When we performed the calculations in parts 1b) and 1c), we simply plotted the expectation value of the energy as a function of the parameter α. For large systems, this means that we end up with spending equally many Monte Carlo cycles for values of the energy away from the minimum. We can improve upon this by using various optimization algorithms. The aim of this part, still using only the non-interacting case, is to add to our code either a steepest descent algorithm or a stochastic gradient optimization algorithm in order to obtain the best possible parameter α which minimized the expectation value of the energy. You can check the code in Github - Part d.

The Steepest descent method as was mention it gives us a way to find the minimum value of a function, the basic idea of gradient descent is that a function $F(x)$ where $x = (x_1, x_2, ...x_n)$ decreases fastest if one goes from x in the direction of the negative gradient that mean $-\nabla F(x)$,

with this idea it can be is possible to show that

$$x_{k+1} = x_k - \gamma \nabla F(x_k) \tag{85}$$

and if $\gamma \geq 0$ and small enough, then $F(x_{k+1}) \leq F(x_k)$ that meaning we will find the minimum value of the function $F(x)$ if we have enough iterations. As a comment the parameter γ is often referred to as the step length or the learning rate.

Herein, the optimizer play a very important role for reduce the consumption time and find the best parameter, we have considered $\beta = 1$. This optimal parameter (α), we allow us find the best expectation value. For this case, we have used the gradient descent method, for which we need to use the derivative wave function with respect to α.

The elements of the gradient of the local energy (from **Equ.** (34)) is given by

$$\bar{E}_\alpha = 2\left(\langle \frac{\bar{\psi}_\alpha}{\psi[\alpha]} E_L[\alpha]\rangle - \langle \frac{\bar{\psi}_\alpha}{\psi[\alpha]}\rangle\langle E_L[\alpha]\rangle\right) \tag{86}$$

That means, compute the expectation values of

$$\langle \frac{\bar{\psi}_\alpha}{\psi[\alpha]} E_L[\alpha]\rangle \tag{87}$$

and

$$\langle \frac{\bar{\psi}_\alpha}{\psi[\alpha]}\rangle\langle E_L[\alpha]\rangle \tag{88}$$

Once we have the gradient of the local energy, it just uses the gradient descent method **Equ.** (85) in order to find the minimum.

The results are given in the table below,

α	Result	Time CPU [s]
Analytic	0.5000025256	1.5659
Numeric	0.4999998300	796.8570

Table 1: Result of the optimization part for analytic and numeric solution.

these result are pretty good since the exact solution we know and the value is $\alpha = 0.5$.

3.5 Part e): A better statistical analysis

In performing the Monte Carlo analysis we will use the Blocking or Bootstrap techniques to make the final statistical analysis of the numerical data. Present your results with a proper evaluation of the statistical errors. Repeat the calculations from part d) (or c) and include a proper error analysis. Limit yourself to the three-dimensional case only. A useful strategy here is to write your expectation values to file and then have a Python code which does the final statistical analysis. Alternatively, you can obviously write addition functions to be used by your main program and perform the final statistical analysis within the same code.

In this part we are going to dealing with a statistical analysis for that is normally used resampling techniques, they involve repeatedly selecting samples from a training set and then fitting a model of interest to each sample. This iterative process offers additional information about the fitted model, as well allows us to gain insights that would otherwise be unavailable from a single fitting using the original training sample.

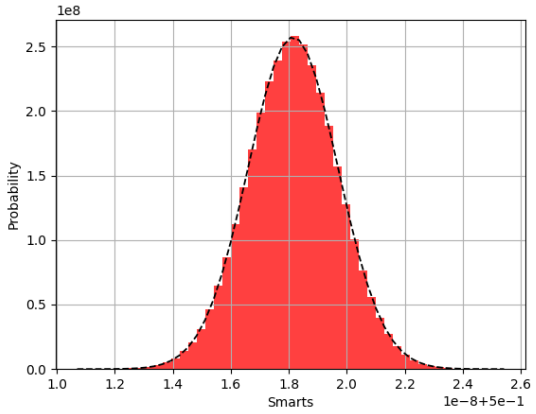
For us approach we are going to use Bootstrap technique, the basic idea comes when we do not have a enough large set the data for this reason we are not able to do a good statistic analysis, so the Bootstrap is a way to dealing with that. The solution is basically to do many resampling over us data base but with some condition, the resampling has to be chosen random and independent of each other.

The result are shown in the **Tab.** 2 and also the histogram of the result data after the Bootstrap technique **Fig.** 13, it is figured using the number of MCc= 10^3 doing 10^5 samplings for our base data and 10^6 resampling.

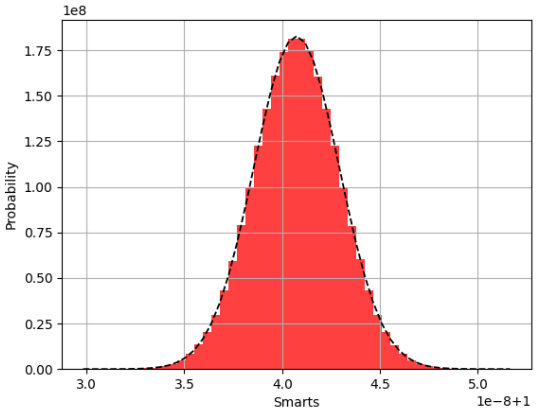
Dimension	Using	Mean value	Variance	Error
1	Original data	0.5	4.9021410^{-7}	2.2140810^{-6}
	Bootstrap	0.5	1.5505710^{-9}	$3.93773\ 10^{-8}$
2	Original data	1.0	$6.90436x10^{-7}$	$2.62761x10^{-6}$
	Bootstrap	1.0	$2.18362x10^{-9}$	$4.67292x10^{-8}$
3	Original data	1.5	$8.49689x10^{-7}$	$2.91494\ x10^{-6}$
	Bootstrap	1.5	$2.68573x10^{-9}$	$5.1824\ x10^{-8}$

Table 2: Comparison the result of data base with Bootstrap technique.

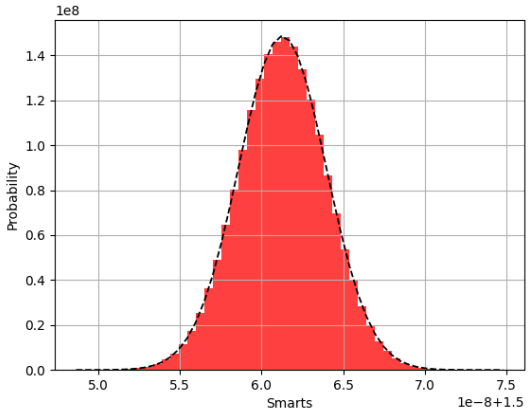
the result shows us that using the resampling technique we will be able to find a more appropriate (exact) information about my data base, when we do not have enough data. the mean value found it is the exact solution that mean our result are good, we can see the histogram distribution **Fig.** 13 the media it is basically the one that is shown in the **Tab.** 2.



(a) N = 1 and D = 1



(b) N = 1 and D = 2



(c) N = 1 and D = 3

Figure 13: The histogram distribution for the output after doing Bootstrap technique for N= 1 and D $= 1, 2, 3$.

You can check the code for this section in Github - Part e.

3.6 Part f): Parallelizing your code

Before we add the two-body interaction, our final computational ingredient is to parallelize our code. With this last ingredient we have obtained a code framework which contains the essential elements used in a Variational Monte Carlo approach to a many-body problem. Dealing with a non-interacting case only till now allows us to continuously check our results against exact solutions.

The idea behind the parallelized comes when we have tasks are executed sequentially, so the program should wait for one task to complete before moving on to the next. This can lead

to a waste of valuable processing time, especially if some tasks are independent and don't need to wait for others to complete, last part it becomes important to us because in the MCM we are dealing with a independent processes. the more easy way we can parallelized MCM is doing many samplings in parallel, that meaning when before only we was running one task (one sampling) now we can running several tasks (sampling) and this it is translated in a optimization time. Obviously it will be dependent our computer but always will improve execution time. The idea is shown in the figure below

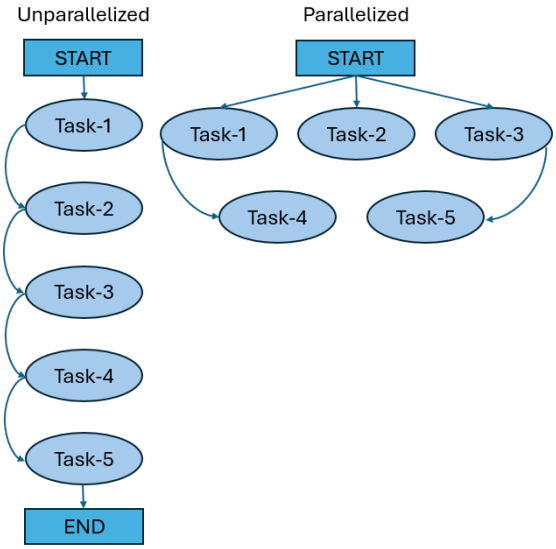


Figure 14: Scheme for the different approach unparalleled and parallelized processes

the big different we can see in the parallelized we can do multitask while that unparalleled processes just one task at the time. For the particular scheme **Fig.** 14 we are doing three task at the same time. You will be able to see that idea was apply in different part of the code to try to optimize the time consuming CPU. You can check the code in Github - Part f. The file Samplings.py is a good example.

3.7 Part g): The repulsive interaction

We are now ready to include the repulsive two-body interaction.

We turn to the elliptic trap with a repulsive interaction. We fix, as in Refs. [1,2] below, $a/a_{ho} = 0.0043$. We introduce lengths in units of a_{ho}, $r \to r/a_{ho}$ and energy in units of $\hbar\omega_{ho}$. Show then that the original Hamiltonian can be rewritten as

$$H = \sum_{i=1}^{N} \frac{1}{2}\left(-\nabla_i^2 + x_i^2 + y_i^2 + \gamma^2 z_i^2\right) + \sum_{i<j} V_{int}(|\mathbf{r}_i - \mathbf{r}_j|).$$

What is the expression for γ? Choose the initial value for $\beta = \gamma = 2.82843$ and compute ground state energy using the trial wave function of Eq. (5) using only α as variational parameter. Vary again the parameter α in order to find a minimum. Perform the calculations for $N = 10, 50$ and $N = 100$ and compare your results to those from the ideal case in the previous exercises. Benchmark your results with those of Refs. [1,2].

Herein the expression γ is the atomic thermal wavelength.

3.8 Part h): Onebody densities

With the optimal parameters for the ground state wave function, compute again the onebody density with and without the Jastrow factor. How important are the correlations induced by the Jastrow factor?
One-body density is related to find a particle at a distance from the center of a system.

$$\rho(r, r') = \int dr' |\Psi_T(R)|^2 \tag{89}$$

The Jastrow factor introduce a kind of interaction between bosons (repel), we can see it in the **Eqs.** (7). Therefore if we neglected this factor, the probability of finding some boson in the center of the system is higher. Unlike if we include this factor, the probability of finding the bosons will be lower, that means that the bosons will be far away from the center.

All the codes have also been sort in a more professional way, sort by scripts. You can check it in Github - Scripts. There we include different modules for each section.

References

[1] Harrison RL. Introduction To Monte Carlo Simulation. *AIP Conf Proc.*, 2010 January 5; 1204: 17–21. doi:10.1063/1.3295638.

[2] Morten Hjorth-Jensen. Computational physics lecture note fall 2015.

[3] W. K. Hastings. Monte Carlo sampling methods using Markov chains and their applications. *Biometrika*, 57(1):97–109, 04 1970.

[4] J. L. DuBois and H. R. Glyde. Bose-einstein condensation in trapped bosons: A variational monte carlo analysis. *Phys. Rev. A*, 63:023602, Jan 2001.

[5] J. K. Nilsen, J. Mur-Petit, M. Guilleumas, M. Hjorth-Jensen, and A. Polls. Vortices in atomic bose-einstein condensates in the large-gas-parameter region. *Phys. Rev. A*, 71:053610, May 2005.