Project 3.
FYS3150. Computational Physics.

Ana Planes Hernandez.
Nikolai Sigersvold.
Faria Ghaffar.

github.com/fastfirefly

October 21, 2019

1 ABSTRACT

The aim of this project is to study different methods for solving a six dimensional integral. We will make use of Gaussian quadrature, whose results using both Legendre and Laguerre polynomials will be compared to Monte Carlo integration for the same problem. We will prove how the latter works in a more efficient way, giving a much better result in a very small amount of time in comparison with the other.

2 INTRODUCTION

The correlation energy between two or more electrons is a measurement to determine how affected one electron is by the presence of the rest in a particular system due to electron repulsion. In other words, when the repulsion energy of two electrons is calculated between an electron and the average electron density for the other electron, it does not take into account the fact that the electron will push away the other electrons as it moves around, which will diminish the repulsion energy. The electron correlation accounts for this.

We aspire to determine the ground state correlation energy between the two electrons in the helium atom. It is a highly intriguing problem, as it has not been possible to find a closed-form solution to the Schrödinger equation of the helium yet, and it appears frequently in quantum mechanics calcullations.

In order to find this energy, the resolution of a six dimensional integral is needed, as we will present in the next section. It will be solved using different methods for both low dimensional and multidimensional integrals. To start, the problem will be approached using Gauss-Legendre quadrature. Next, it will be improved by using a change of variable to spherical coordinates with Laguerre polinomials and, finally, Monte Carlo integration will be implemented, and boosted using parallelization. In this report, all of the methods are expounded along with a brief guide of how to implement the code to resolve this problem. Results are critically discussed and a conclusion is presented at the end where we offer a summary of the outcomes.

3 THEORY

An atom of atomic number Z consists of a heavy nucleus with electric charge Ze surrounded by Z electrons of mass m and charge $-e$. The Hamiltonian of the system is

$$H = \sum_{i=1}^{Z} \left(-\frac{\hbar^2}{2m} \nabla_i^2 - \left(\frac{1}{4\pi\epsilon_0} \right) \frac{Ze^2}{r_i} \right) + \frac{1}{2} \left(\frac{1}{4\pi\epsilon_0} \right) \sum_{i \neq j}^{Z} \frac{e^2}{|r_i - r_j|}$$

The term in curly brackets represents the kinetic plus potential energy of the ith electron, in the electric field of the nucleus. The second sum is the potential energy associated with the mutual repulsion of the electrons. The problem is to solve the Schrödinger's equation for the wave function.

The wave function of an electron i in a helium atom in the 1s state is given by

$$\psi_{1s}(r_i) = e^{-\alpha r_i}$$

where α is a parameter and the coordinates are

$$r_i = \sqrt{x_i^2 + y_i^2 + z_i^2}$$

In our case, for the helium atom, we will take $\alpha = 2$, which corresponds to its charge Z.
Then, for two electrons, the wave function is simply the product of the two sepparate wave functions of each electron.

$$\psi(r_1, r_2) = e^{-\alpha(r_1 + r_2)}$$

Our prime objective is to establish the ground state correlation energy between these two electrons in the helium atom.
Consequently, the integral we need to solve is the quantum mechanical expectation value of this correlation, which is

$$\langle \frac{1}{|r_1 - r_2|} \rangle = \int_{-\infty}^{\infty} dr_1 dr_2 e^{-2\alpha(r_1 + r_2)} \frac{1}{|r_1 - r_2|}$$

The analitical value of this integral is known to be $5\pi^2/16^2 \simeq 0.19276571$

Using the different methods which will be presented in the following section, the numerical results attained will be compared to this one.

4 METHODS

4.1 Gauss Quadrature

The Gauss Quadrature is used to obtain the best approximation of an integral by using orthogonal polynomials in a determined interval. It yields exact values of integrals for polynomials of degree up to $2N - 1$.

$$\int f(x)dx \approx \int P_{2N-1}(x)dx = \sum P_{2N-1}(x_i)\omega_i$$

The integration points are given by the zeros of the chosen orthogonal polynomials of order N, and the weights ω_i determined from the inverse of a matrix. In the case of **Legendre Polynomials**, we work in the interval [-1,1]. The polynomial $P_{2N-1}(x)$ may be approximated by

$$P_{2N-1}(x) = L_N(x)P_{N-1}(x) + Q_{N-1}(x)$$

where $L_N(x)$ are the orthogonal Legendre polynomials, defined in the interval. Therefore, the integral can be written as

$$\int\limits_{-1}^{1} P_{2N-1}(x)dx = \int\limits_{-1}^{1} L_N(x)P_{2N-1}(x)dx + \int\limits_{-1}^{1} Q_{N-1}(x)dx = \int\limits_{-1}^{1} Q_{N-1}(x)dx$$

where we have used the orthogonality property in the last step. At the points x_k where L_N is 0

$$P_{2N-1}(x_k) = Q_{N-1}(x_k) \ k = 0, 1, ...N-1$$

Now, we can expand $Q_{N-1}(x)$ in terms of the Legendre polynomials as

$$Q_{N-1}(x) = \sum_{i=0}^{N-1} \alpha_i L_i(x)$$

where α_i are constants.
Making use of the orthogonality condition again, we have that

$$\int\limits_{-1}^{1} Q_{N-1}(x)dx = \sum_{i=0}^{N-1} \alpha_i \int\limits_{-1}^{1} L_0(x)L_i(x)dx = 2\alpha_0$$

where we have chosen $L_0(x) = 1$
To get the weights, we simply need to find the coefficient α_0. We can rewrite

$$Q_{N-1}(x) = \sum_{i=0}^{N-1} \alpha_i L_i(x_k) = \sum_{i=0}^{N-1} \alpha_i L_{ik}$$

the L_{ik} being the elements of the matrix L

$$L = \begin{bmatrix} L_0(x_0) & L_1(x_0) & ... & L_{N-1}(x_0) \\ L_0(x_1) & L_1(x_1) & ... & ... \\ ... & ... & ... & ... \\ L_0(x_{N-1}) & ... & ... & L_{N-1}(x_{N-1}) \end{bmatrix}$$

which has the property $LL^{-1} = I$, due to the fact that these polynomials are all linearly independent of each other.

Then, we obtain the equations:

$$\begin{gathered} Q_{N-1}(x_0) = \alpha_0 L_0(x_0) + \alpha_1 L_1(x_0) + ... + \alpha_{N-1} L_{N-1}(x_0) \\ Q_{N-1}(x_1) = \alpha_0 L_0(x_1) + \alpha_1 L_1(x_1) + ... + \alpha_{N-1} L_{N-1}(x_1) \\ ... \\ Q_{N-1}(x_{N-1}) = \alpha_0 L_0(x_{N-1}) + ... + \alpha_{N-1} L_{N-1}(x_{N-1}) \end{gathered}$$

It is straightforward to extract the coefficients α_k, which are

$$\alpha_k = \sum_{i=0}^{N-1} (L^{-1})_{ki} Q_{N-1}(x_i)$$

and, in particular,

$$\alpha_0 = \sum_{i=0}^{N-1} (L^{-1})_{0i} Q_{N-1}(x_i)$$

that was the coefficient we needed.

Consequently, we attain

$$\int_{-1}^{1} P_{2N-1}(x) = \int_{-1}^{1} Q_{N-1}(x)dx = 2\alpha_0 = 2\sum_{i=0}^{N-1} (L^{-1})_{0i} P_{2N-1}(x_i)$$

where $\omega_i = (L^{-1})_{0i}$ are the integration weights.

After all this procedure, we may conclude that the integral could be approximated by

$$\int_{-1}^{1} f(x)dx \approx \sum \omega_i P_{2N-1}(x_i)$$

In the event that we are able to approximate the original integral with a weight function $x^{\alpha}e^{-x}$ with integration limits [0,∞) it is wise to use **Laguerre polynomials** instead.

$$\int_{0}^{\infty} f(x)dx = \int_{0}^{\infty} x^{\alpha}e^{-x}g(x)dx \approx \sum \omega_i f(x_i)$$

4.2 Monte Carlo Integration

Monte Carlo Integration is a method for computing the value of multidimensional integrals using probabilistic techniques, in order to obtain the expected value of the integral by choosing an appropiate probability distribution function.

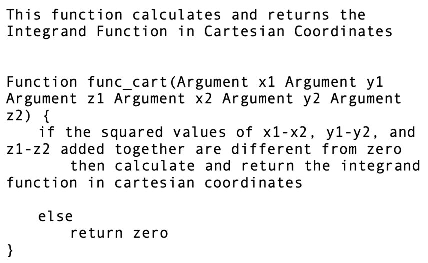
The expectation value of a function $f(x)$ is

$$E[f] =< f >= \int_{a}^{b} f(x)p(x)dx \simeq \frac{1}{N}\sum_{i=0}^{N-1} f(x_i)$$

where $p(x)$ is the probability distribution, and we assume that each event on the domain is equally likely.
Often, we do not know the nature of the distribution, but we can use sample points according to the way the function looks like, which is known as importance sampling, and we will carry out in this project.

4.3 Implementation

First, we use **Gauss-Lagendre** quadrature in a brute force way to resolve the integral, using cartesian coordinates. We approximate the limits $\mp\infty$ by certain values of $\mp\lambda$, and make a plot to check suitable the value we have chosen for λ is.
To account for the problems that might arise when the denominator $|r_1 - r_2| = 0$, we include a statement in the function defining the cartesian coordinates.



```
This function calculates and returns the
Integrand Function in Cartesian Coordinates


Function func_cart(Argument x1 Argument y1
Argument z1 Argument x2 Argument y2 Argument
z2) {
    if the squared values of x1-x2, y1-y2, and
z1-z2 added together are different from zero
        then calculate and return the integrand
function in cartesian coordinates

    else
        return zero
}
```

Figure 1: Pseudocode describing the function used for the cartesian coordinates

Next, we approach the same problem using spherical coordinates instead. We can rewrite the integral so that

$$dr_1 dr_2 = r_1^2 dr_1 r_2^2 dr_2 dcos(\theta_1) dcos(\theta_2) d\phi_1 d\phi_2$$

The function, $f(r_1, r_2, \phi_1, \phi_2, \theta_1, \theta_2)$ turning into

$$\frac{1}{r_{12}} = \frac{1}{\sqrt{r_1^2 + r_2^2 - 2r_1 r_2 cos(\beta)}}$$

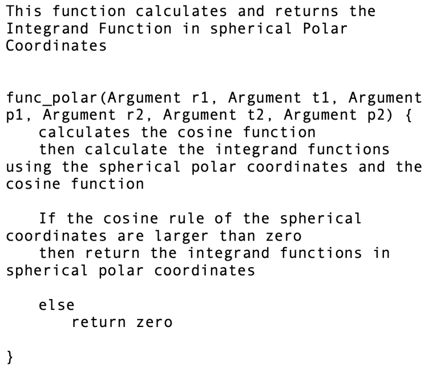
where $cos(\beta) = cos(\theta_1 cos(\theta_2) + sin(\theta_1) sin(\theta_2) cos(\phi_1 - \phi_2))$

The angles are $\theta \in [0, \pi]$ and $\phi \in [0, 2\pi]$, and $r \in [0, \infty)$

Then, the integral we need to resolve becomes

$$\int_0^{2\pi} d\phi_1 \int_0^{2\pi} d\phi_2 \int_0^{\pi} d\theta_1 sin\theta_1 \int_0^{\pi} d\theta_2 sin\theta_2 \int_0^{\infty} dr_1 r_1^2 \int_0^{\infty} dr_2 r_2^2 e^{-\alpha(r_1 r_2)} f(r_1, r_2, \phi_1, \phi_2, \theta_1, \theta_2)$$

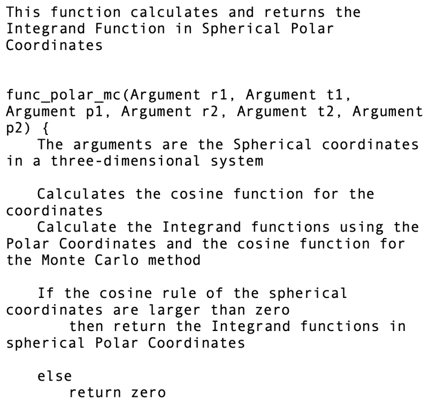
For the integrals corresponding to the angles ϕ, θ, we make use of Legendre polynomials again, but for the part of r, we can use Laguerre polynomials.



```
This function calculates and returns the
Integrand Function in spherical Polar
Coordinates


func_polar(Argument r1, Argument t1, Argument
p1, Argument r2, Argument t2, Argument p2) {
    calculates the cosine function
    then calculate the integrand functions
using the spherical polar coordinates and the
cosine function

    If the cosine rule of the spherical
coordinates are larger than zero
    then return the integrand functions in
spherical polar coordinates

    else
        return zero

}
```

Figure 2: Pseudocode describing the function used for the spherical coordinates

To implement the Monte Carlo method, we will do it by brute force and using importance sampling.
When using brute force, we approximate again the limits $\mp\infty$ by $\mp\lambda$ and, we make use of the uniform distribution only to get the value of the integral.
Once we use importance sampling, we rewrite the integral in spherical coordinates and choose the exponential distribution for the part of the integral corresponding to $r_i \in [0, \infty]$. For the angles $\theta \in [0, \pi]$ and $\phi \in [0, 2\pi]$, we use the uniform distribution to generate random numbers in the domain.



```
This function calculates and returns the
Integrand Function in Spherical Polar
Coordinates


func_polar_mc(Argument r1, Argument t1,
Argument p1, Argument r2, Argument t2, Argument
p2) {
    The arguments are the Spherical coordinates
in a three-dimensional system

    Calculates the cosine function for the
coordinates
    Calculate the Integrand functions using the
Polar Coordinates and the cosine function for
the Monte Carlo method

    If the cosine rule of the spherical
coordinates are larger than zero
        then return the Integrand functions in
spherical Polar Coordinates

    else
        return zero
```

Figure 3: Pseudocode describing the function in spherical coordinates for the Monte Carlo method.

5 RESULTS

5.1 Gauss Methods

When we implement the Gauss-Legendre algorithm in a brute force way, approximating the integration limits with ∓ 3 and choosing 30 mesh points, the result we get for the integral is **0.1772829565**, which we can see differs considerably from the analitical result.

In the following plot, we have represented the function $e^{-\alpha r_i}$ with respect to r_i. We can see how, when $r_i > 3$, the function goes to 0, which means the limits we have chosen are appropriate.

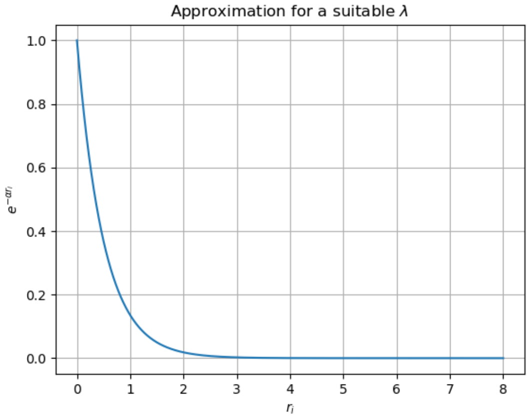


Figure 4: Plot representing e^{r_i} with respect to r_i to check the best values for λ

Once the integral is rewritten in terms of spherical coordinates and we use both Legendre and Laguerre polynomials to approach the problem, the result obtained for 30 integration points, which is **0.1947787611**, improves significantly. This is, due to the fact that when we used Gauss-Legendre only, approximating the limits $\mp\infty$ with finite numbers caused a bigger error.

The following plot shows the values obtained for different mesh points N. We can see how, for small values of N, the values differ greatly from the exact one but, as we increase N, they get closer. In particular, for the Gauss-Laguerre method, the best results are when N equals 15.

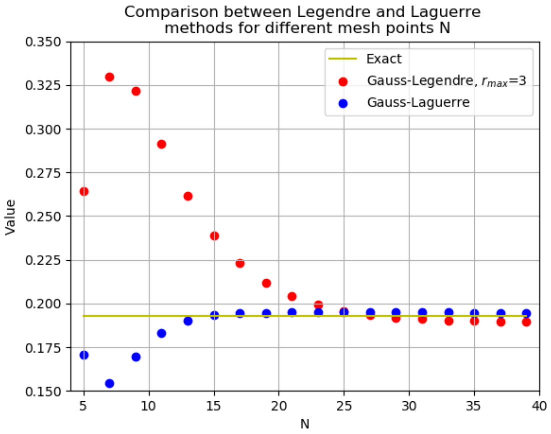


Figure 5: Plot to compare both methods for different N values

A table is shown next where we have compared the time each method takes for different integration points using both methods.

Table 1: Time comparison for different N values with Gauss methods

Method	N=10	N=20	N=30	N=40
Gauss-Legendre	0.0625	5.2969	60.1719	351.8750
Gauss-Laguerre	0.2969	18.0781	225.4063	1293.0781

It can be seen from the table above that using Legendre in a brute force way takes less time than using Gauss-Laguerre. However, the results that we attain using the latter are substantially better and, therefore, worth the time.

5.2 Monte Carlo Integration

When we compute Monte Carlo Integration in a brute force way, using the uniform distribution with 10^8 integration points, we obtain a result for the integral of **0.1914321401**, which is fairly close to the known analytical result, but could be better.

However, once we use importance sampling in our program, transforming the integral to spherical coordinates and using two different probability functions as we explained before, the result gets better, giving a value of **0.1928820193** for the same number of mesh points.
A table is shown next where we compare the results obtained along with their variance and standard deviation.

Table 2: Results using Monte Carlo Integration

Method	Result	Standard deviation σ	Variance σ^2
Brute Force	0.1914321401	0.000013359945	0.00000000002
Improved MC	0.1928820193	0.000021023050	0.00000000004

We can see that both the variance does not decrease when we use the improved Monte Carlo Integration. However, this could also happen due to the fact that, because of the randomness of the Monte Carlo method, we obtain varying values each time we run it. To be able to appreciate it in a more graphical way, two histograms are shown, where we can see thee different values we get when we run the program an specific amount of times.

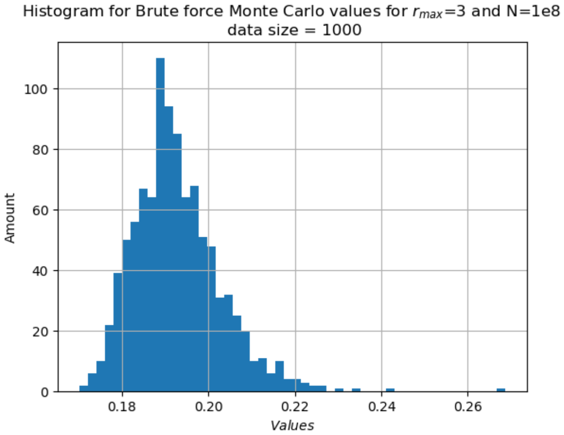


Figure 6: Histogram showing the different values obtained for brute force

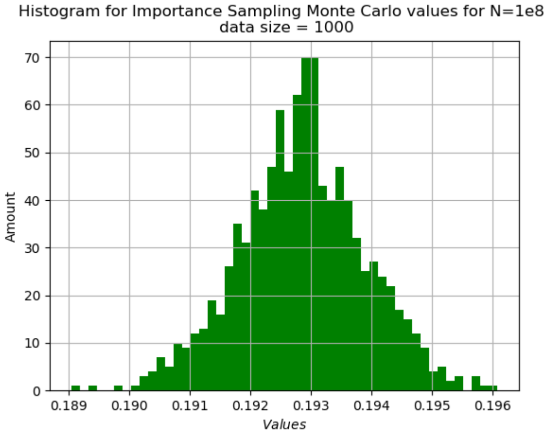


Figure 7: Histogram showing the different values obtained for importance sampling

Next, we attach a table showing the time each method uses to get the value of the integral, for N= 10^8.

Table 3: Time comparison for the different methods used

Method	Brute MC	Improved MC	Parallization
Time (s)	16.296875	35.093750	14.380651

It is straightforward to see that, using parallelization, the result obtained will be practically the same as in the case of importance sampling but the time it will take to obtain it will be much less, which is something to take into account when running calculations like these.

6 CONCLUSION

Applying Gauss Quadrature to get the expectation value of the ground state correlation energy between two electrons in the helium atom gave us fairly good results. However, these low dimensional methods begin getting inaccurate when considering integrals of larger dimension. Monte Carlo Integration, based on probabilistic techniques, gave us better results with a very small error and worked in a much more efficient way, being able to take in a very large number of integration points and little time. Moreover, including parallelization in the program made it speed up even more. Therefore, we can conclude that Monte Carlo integration is an exceptionally powerful method that it is convenient to use when we bump into integrals of high dimension like the one we were concerned about in this project.

7 REFERENCES

(1) Hjorth-Jensen, M. *Computational Physics. Lecture Notes 2015.* University of Oslo, 2015.
(2) William H. Press, Saul A. Teukolsky, William T. Vetterling, Brian P. Flannery, *Numerical Recipes* Cambridge University Press, 2007.
(3) Griffiths, David J. *Introduction to Quantum Mechanics.* Cambridge University Press, 2017.