Measuring the relative efficiency and accuracy of Gaussian quadrature and Monte Carlo integration methods

René Ask, Kaspara Skovli Gåsvær & Maria Linea Horgen

(Dated: October 21, 2019)

The Coulomb-repulsion term in the Hamiltonian for a helium atom makes it impossible to obtain a closed-form expression of the energy spectrum. The standard strategy to compute the ground state energy of Helium E_{GS} is to apply the variational principle. This requires us to solve $I = \langle 1/r_{12} \rangle$, where $r_{12} \equiv |\mathbf{r}_1 - \mathbf{r}_2|$. We attack this problem with deterministic approach with Gaussian quadrature, using Legendre- and Laguerre polynomials, and a non-deterministic approach with Monte Carlo integration. We measure the relative efficiency and accuracy of the various methods. With Gauss-Legendre quadrature we determined the integral I to four leading digits, within a reasonably time frame. We observed that by using Laguerre polynomials we obtained results that exceeded the Legendre approach, and with an optimized choice of coordinate system we reached an average speedup of a factor $2.77 \cdot 10^5$. We implement two Monte Carlo methods, one using uniform PDFs known as brute force (BF) and one using specialized PDFs to reduce the variance, a technique known as importance sampling (IS). The best approximation with the Monte Carlo methods was obtained using IS and gave a relative error $\sim 10^{-4}$. We achieve a reduced variance by IS with a ratio of 17.3, on average. Similarly, with the parallelized code we acquired a mean reduced variance ratio of ~ 9.3. Through parallelization with MPI, we achieved an average speed up of $\bar{S} \simeq 3.4$ for the BF implementation and $\bar{S} \simeq 3.0$ for IS. As a final test of the implementation, we applied the code to compute $\langle H \rangle$ and achieved a relative error of $\sim 5\%$ with respect to the experimental value of E_{GS}. The results obey $\langle H \rangle \geq E_{\text{GS}}$, as expected.

I. INTRODUCTION

The Schrödinger equation for multi-particle systems that includes the Coulomb interaction permits no closed-form solutions. For the Helium atom, the Hamiltonian is

$$H = -\frac{\hbar^2}{2m}\left(\nabla_1^2 + \nabla_2^2\right) - \frac{e^2}{4\pi\epsilon_0}\left(\frac{2}{r_1} + \frac{2}{r_2} - \frac{1}{|\mathbf{r}_1 - \mathbf{r}_2|}\right) \tag{1}$$

which can be rewritten into

$$H = H_0 + \frac{e^2}{4\pi\epsilon_0}\frac{1}{|\mathbf{r}_1 - \mathbf{r}_2|}, \tag{2}$$

where H_0 is the Hamiltonian of the Hydrogen-like Helium atom and $\|\mathbf{r}_i\| \equiv r_i$. Quantum mechanics plays an important role in modern physics, and being able to test its predictions on realistic systems is important to confirm its validity. Although an analytical expression for the energy spectrum of this Hamiltonian isn't permissible, one can approximate its ground state using what is known as the *variational principle*. This principle states that the expectation value of the Hamiltonian $\langle H \rangle$ yields an upper-bound on the ground state energy E_{GS}. Specifically, we can approximate the ground state energy of Helium by

$$E_{\text{GS}} \approx \langle H \rangle = \langle H_0 \rangle + \frac{e^2}{4\pi\epsilon_0}\left\langle \frac{1}{|\mathbf{r}_1 - \mathbf{r}_2|} \right\rangle. \tag{3}$$

For this upper-bound to be a good approximation to the ground state energy, the choice of a trial wave function to compute $\langle H \rangle$ is important, and a wise choice here is to use the ground state of the Hamiltonian H_0, which is

$$\Psi(\mathbf{r}_1, \mathbf{r}_2) = Ae^{-\alpha(r_1 + r_2)}, \tag{4}$$

where A is the appropriate normalization constant and $\alpha = Z/a$, where Z is the proton number and a is the Bohr radius. For the Helium atom, $Z = 2$. Now, for the ground state $\langle H_0 \rangle = 8E_1$, where E_1 is the ground state energy of the Hydrogen atom. To estimate the ground state energy, it is then instructive to compute

$$\left\langle \frac{1}{|\mathbf{r}_1 - \mathbf{r}_2|} \right\rangle = \int d^3\mathbf{r}_1 d^3\mathbf{r}_2 |\Psi(\mathbf{r}_1, \mathbf{r}_2)|^2 \frac{1}{|\mathbf{r}_1 - \mathbf{r}_2|}, \tag{5}$$

which with $\alpha = 2$ has the closed-form solution

$$\left\langle \frac{1}{|\mathbf{r}_1 - \mathbf{r}_2|} \right\rangle = \frac{5\pi^2}{256}, \tag{6}$$

which we derive in the appendix, see section VII B. In this article we present several methods for numerical integration to tackle eq. (5). The two first are versions of Gaussian Quadrature, where the basic idea is to approximate eq.(5) by using the quantitative nature of Legendre- and Laguerre-polynomials. The last two methods are Monte Carlo methods for integration. One known as a *brute force* (abbreviated BF) method based on uniform distributions only. The other utilizes probability distributions (PDFs) specialized for the specific integrand at hand that reduces variance, which is a technique known as *importance sampling* (abbreviated IS).

In this article, we benchmark and analyse the efficiency of the various methods, with a focus on time-consumption and accuracy. In particular, we'll parallelize the codes for Monte Carlo integration using MPI and study the acquired speedup, or lack thereof. As a final test of our implementation, we'll apply the variational principle to approximate the ground state energy of the helium atom.

For codes and their documentation, see [1].

II. FORMALISM

A. Gaussian Quadrature

The main underlying idea of Gaussian quadrature is that

$$\int_a^b W(x) f(x) dx \approx \sum_{i=0}^{N-1} w_i f(x_i), \tag{7}$$

where $W(x)$ is a weight function, w_i are the weights and x_i the mesh points. $W(x)$ is dependent on which set of orthogonal functions, and which set we use is dependent upon the interval $[a, b]$. The derivation in this section and the following sections on Gaussian quadrature methods heavily lends inspiration from [7].

B. Gauss-Legendre method

The Gauss-Legendre method is based on the weight function $W(x) = 1$ and a complete set of polynomials called Legendre polynomials for which we shall denote the Nth Legendre polynomial as $L_N(x)$. Furthermore, the interval of interest is $[a, b] = [-1, 1]$, so the specialized version of eq. (7) is

$$\int_{-1}^{1} f(x) dx \approx \sum_{i=0}^{N-1} w_i f(x_i). \tag{8}$$

The rest of this section is devoted to deriving the mesh points x_i and the weights w_i. We first note that we can approximate

$$\int_{-1}^{1} f(x) dx \approx \int_{-1}^{1} P_{2N-1}(x) dx = \sum_{i=0}^{N-1} w_i P_{2N-1}(x_i), \tag{9}$$

where $P_{2N-1}(x)$ is a polynomial with $\deg P_{2N-1} = 2N - 1$, x_i are the zeros of the Nth Legendre polynomial $L_N(x)$ and w_i are the corresponding weights. The polynomial $P_{2N-1}(x)$ can be represented as

$$P_{2N-1}(x) = L_N(x) P(x) + Q(x), \tag{10}$$

where $\deg P < N$ and $\deg Q < N$. The Legendre polynomials obey the following orthogonality relation.

$$\int_{-1}^{1} L_i(x) L_j(x) dx = \frac{2}{2i+1} \delta_{ij}, \tag{11}$$

which implies

$$\int_{-1}^{1} P_{2N-1}(x)dx = \int_{-1}^{1} \left[L_N(x)P(x) + Q(x)\right] dx = \int_{-1}^{1} Q(x)dx. \tag{12}$$

Suppose further that x_k for $k = 0, 1, ..., N-1$ are the zeros of $L_N(x)$ such that $L_N(x_k) = 0$. Then $P_{2N-1}(x)$ can be completely specified by

$$P_{2N-1}(x_k) = Q(x_k). \tag{13}$$

The Legendre polynomials $\{L_i(x)\}$ forms a complete function set on the interval $[-1, 1]$, thus $Q(x)$ can be written as linear combination of the form

$$Q(x) = \sum_{i=0}^{N-1} \alpha_i L_i(x). \tag{14}$$

Using the orthogonality relation together with $L_0(x) = 1$, we obtain

$$\int_{-1}^{1} Q(x)dx = \sum_{i=0}^{N-1} \alpha_i \int_{-1}^{1} dx L_0(x)L_i(x) = \sum_{i=0}^{N-1} \alpha_i \frac{2}{2i+1}\delta_{i0} = 2\alpha_0 \tag{15}$$

and assuming that we know $Q_k \equiv Q(x_k)$ for $k = 0, 1, ..., N-1$, we can rewrite this as

$$Q_k = \sum_{i=0}^{N-1} \alpha_i L_i(x_k) = \sum_{i=0}^{N-1} \alpha_i L_{ik}, \tag{16}$$

where $L_{ik} \equiv L_i(x_k)$. Since the the Legendre Polynomials form a complete set on $[-1, 1]$, the column space of L_{ij} forms a basis for $\mathbb{R}^n$, so it follows that there exists an inverse. It then follows that

$$\alpha_k = \sum_{i=0}^{N-1} L_{ki}^{-1} Q_i, \tag{17}$$

and since we're interested in α_0, we obtain the explicit formula

$$\alpha_0 = \sum_{i=0}^{N-1} L_{0i}^{-1} Q_i \tag{18}$$

Now, since

$$\int_{-1}^{1} P_{2N-1}(x)dx = \int_{-1}^{1} Q(x)dx = 2\alpha_0, \tag{19}$$

it follows that

$$\int_{-1}^{1} P_{2N-1}(x)dx = \sum_{i=0}^{N-1} \omega_i P_{2N-1}(x_i) = \sum_{i=0}^{N-1} 2L_{0i}^{-1} P_{2N-1}(x_i), \tag{20}$$

where we used the fact that $P_{2N-1}(x_i) = Q(x_i)$ where x_i are the zeros of $L_N(x)$. It thus follows that the weights are $w_i = 2L_{0i}^{-1}$. Reverting back to the general formalism then, we get

$$\int_{-1}^{1} f(x)dx \approx \sum_{i=0}^{N-1} 2L_{0i}^{-1} f(x_i). \tag{21}$$

The general formalism for d-dimensions is evident (albeit, if the notation looks unfamiliar, consult appendix VII A for an explanation of the notation used here and the next couple sections we develop multi-dimensional formalism for the various integration methods):

$$I = \int_{-1}^{1} \cdots \int_{-1}^{1} f(\mathbf{x}) d^d\mathbf{x} \approx \sum_{i=0}^{N-1} \cdots \sum_{l=0}^{N-1} w_i^{(1)} \cdots w_l^{(d)} f(x_i^{(1)}, ..., x_l^{(d)}). \tag{22}$$

In our particular case, we have a six-dimensional integral that in Cartesian coordinates can be written as

$$\int_{-\infty}^{\infty}\cdots\int_{-\infty}^{\infty} dx_1dy_1dz_1dx_2dy_2dz_2 \frac{e^{-2\alpha\left(\sqrt{x_1^2+y_1^2+z_1^2}+\sqrt{x_2^2+y_2^2+z_2^2}\right)}}{\sqrt{(x_1-x_2)^2+(y_1-y_2)^2+(z_1-z_2)^2}}. \tag{23}$$

Now, we can obviously not integrate on the intervals $(-\infty,\infty)$ on a computer. We can, however, remedy this problem by finding some fitting constant λ and integrate over the interval $[-\lambda,\lambda]$ in all six dimensions. This is a viable solutions since $e^{-\alpha r_i}\approx 0$ for some value appropriate value of r_i. The number of mesh points needed to reach a satisfactory result, say three leading digits, also has to be decided. This is done through a trial and error process, where we want to find the N that requires the least CPU time to generate a result with a small error. Since the Legendre-polynomials has zeros that lies symmetrically around $x=0$, number of mesh points has to be an odd number.

The determination of λ is done by minimizing an error function $\xi(\lambda)$ numerically,

$$\xi(\lambda)\equiv\left|\frac{I-I'}{I}\right|, \tag{24}$$

where I is the exact value of eq. (5) and I' is the computed value, which will depend on the limits of the integral. As general knowledge the exponential function, e^{-Cx} where C is a constant, decreases rapidly. We therefore compute $\xi(\lambda)$ for $\lambda\in[1,3]$, with the optimal N mentioned above.

Another problem that may arise is when $|\mathbf{r}_1-\mathbf{r}_2|=0$. We solve this by defining a lower boundary for the difference, and if it goes below this limit we automatically set it to zero. In our calculations this limit is set to 10^{-8}.

C. Gauss-Laguerre method

In this section we'll take a different approach to solving the integral, by now using Laguerre polynomials. The Nth Laguerre polynomial is hereafter denoted as $\mathcal{L}_N(x)$. The polynomials are used to approximate integrals of the type

$$\int_0^\infty x^\beta e^{-x}f(x)dx, \tag{25}$$

where $x^\beta e^{-x}$ corresponds to the weight function $W(x)$ in eq. (7). Deriving the mesh points x_i and weights w_i, are done in a similar manner as for the Gauss-Legendre method. We note that the orthogonality relation for Laguerre polynomials is different from Legendre:

$$\int_0^\infty \mathcal{L}_i(x)\mathcal{L}_j dx=\frac{(i+\beta)!}{i!}\delta_{ij}, \tag{26}$$

and the interval of interest is now $[a,b]=[0,\infty)$. By substituting this in eq. (15) the rest of the derivation will be identical. In our special case $\beta=0$, so we obtain the following approximation:

$$\int_0^\infty f(x)dx\approx\sum_{i=0}^{N-1}\mathcal{L}_{0i}^{-1}f(x_i). \tag{27}$$

The weights are now $w_i=\mathcal{L}_{0i}^{-1}$.

We take a moment to generalize this to d-dimensions. Suppose $f(\mathbf{x})=x_1^{\beta_1}\cdots x_d^{\beta_d}e^{-(x_1+\cdots+x_d)}g(\mathbf{x})$, then

$$\begin{aligned} I&=\int_0^\infty\cdots\int_0^\infty f(\mathbf{x})d^d\mathbf{x}\\ &=\int_0^\infty\cdots\int_0^\infty x_1^{\beta_1}\cdots x_d^{\beta_d}e^{-(x_1+\cdots+x_d)}g(\mathbf{x})d^d\mathbf{x}\\ &\approx\sum_{i=0}^{N-1}\cdots\sum_{l=0}^{N-1}w_i^{(1)}\cdots w_l^{(d)}g\left(x_i^{(1)},...,x_l^{(d)}\right), \end{aligned} \tag{28}$$

where $w_i^{(p)}$ and $x_i^{(p)}$ denotes the weights and abscissas of $\mathcal{L}_N(x^{(p)})$.

Since the limits of our integral has changed, we rewrite our problem in eq. (23) to spherical coordinates.

$$\int_0^\infty\int_0^\infty\int_0^{2\pi}\int_0^{2\pi}\int_0^{\pi}\int_0^{\pi} d\theta_1 d\theta_2 d\phi_1 d\phi_2 r_1^2 dr_1 r_2^2 dr_2 \frac{e^{-2\alpha(r_1+r_2)}\sin\theta_1\sin\theta_2}{\sqrt{r_1^2+r_2^2-2r_1r_2\cos\gamma}}, \tag{29}$$

where

$$\cos\gamma \equiv \cos\theta_1\cos\theta_2 + \sin\theta_1\sin\theta_2\cos(\phi_1-\phi_2), \tag{30}$$

defined here [2]. For solving the angular parts of the integral we use the Gauss-Legendre method, through a change of variable. We introduce

$$t(x) = \frac{b-a}{2}x + \frac{b+a}{2}, \tag{31}$$

so an arbitrary integral can be rewritten as

$$\int_a^b f(t)dt = \frac{b-a}{2}\int_{-1}^{1} f\left(t(x)\right)dx. \tag{32}$$

1. *Optimized choice of coordinate system*

Eq. (29) is a six dimensional integral, by changing coordinate system we can reduce it to a three dimensional integral. If we orient the $\mathbf{r}_2$ coordinate system so that the polar axis lies along $\mathbf{r}_1$, we get

$$|\mathbf{r}_1-\mathbf{r}_2| = \sqrt{r_1^2+r_2^2-2r_1r_2\cos\theta_2}, \tag{33}$$

since $\mathbf{r}_1$ now is fixed, (see FIG. 8 in appendix VII B). By introducing the variable changes $u_1 = 2\alpha r_1$ and $u_2 = 2\alpha r_2$, and setting $\beta = 2$, we approximate

$$I \approx \frac{\pi^2}{128}\sum_{i,j,k=0}^{N} w_i^{(1)}w_j^{(2)}w_k^{(3)}\frac{\sin\theta_{2k}}{\sqrt{u_{1i}^2+u_{2j}^2-2u_{1i}u_{2j}\cos\theta_{2k}}}, \tag{34}$$

where the notation u_{1i} and u_{2j} denotes the i-th and j-th zero of the Laguerre-polynomial $\mathcal{L}_N(x)$, respectively. Similarly, θ_{2k} denotes the k-th zero of the Legendre-polynomial $L_N(x)$. By choosing $\beta = 2$, we incorporate u_1^2 and u_2^2, which are given from the Jacobi determinant, in the weight functions w_1 and w_2. In our numerical calculations we have solved the integral for both three and six dimensions, and compared them with one another, and also with the Gauss-Legendre method.

D. Monte-Carlo Integration

1. *Preliminaries*

Suppose X is a stochastic variable that can assume real values $x \in [a,b]$. The expectation value of a function $f(x)$ for some probability distribution $p(x) : [a,b] \to [0,1]$ is

$$\langle f\rangle = \int_a^b p(x)f(x)dx \approx \frac{1}{N}\sum_{i=1}^{N} f(x_i)p(x_i) \equiv \langle f\rangle_N, \tag{35}$$

where x_i for $i = 1,2,...,N$ is a sample set of X. Suppose we draw N such samples and approximate $\langle f(x)\rangle$ for all N samples. The law of large numbers states that

$$\langle f\rangle = \lim_{N\to\infty}\langle f\rangle_N \tag{36}$$

Suppose $p(x)$ is a uniform distribution defined on the interval $[0,1]$ such that $\int p(x)dx = 1$. Since it is uniform $p(x) = A$ for $x \in [0,1]$ where A is some appropriate normalization constant, which we determine

$$\int_{-\infty}^{\infty} p(x)dx = \int_0^1 p(x)dx = A = 1 \tag{37}$$

meaning $p(x) = 1$. The expectation value of $f(x)$ is therefore

$$\langle f \rangle = \int_0^1 f(x)dx. \tag{38}$$

For our purposes it is convenient to use a uniform distribution $p(x) : [0,1] \to [0,1]$. But often times, we're dealing with some arbitrary probability distribution $p(\mu) : [a,b] \to [0,1]$, and so a change of variable is necessary for it to fit into our formalism. Through the change of variable, probability must be conserved, implying that $p(x)dx = p(\mu)d\mu$. And since $p(x) = 1$, we get

$$dx = p(\mu)d\mu. \tag{39}$$

Making the substitution $\mu = \mu(x)$ where $x \in [0,1]$ yields the general expression

$$x(\mu) = \int_0^{x(\mu)} dx = \int_a^{\mu} p(\mu')d\mu', \tag{40}$$

which implies that

$$\int_a^b f(\mu)d\mu = \int_a^b \frac{f(\mu)}{p(\mu)} \underbrace{p(\mu)d\mu}_{dx} = \int_0^1 \frac{f(\mu(x))}{p(\mu(x))}dx. \tag{41}$$

This is as far as we can go until the probability distribution $p(\mu) : [a,b] \to [0,1]$ is specified.

2. *One-dimensional Monte Carlo Integration*

Brute-force Monte Carlo integration involves a uniform probability distribution $p(x) : [0,1] \to [0,1]$. Suppose we want to solve the integral

$$I = \int_a^b f(\mu)d\mu. \tag{42}$$

Let $p(\mu) : [a,b] \to [0,1]$ be a uniform distribution. For $\int p(\mu)d\mu = 1$, it's clear that $p(\mu) = 1/(b-a)$. Then by eq. (40) we get

$$x(\mu) = \int_a^{\mu} p(\mu')d\mu' = \frac{1}{b-a}\int_a^{\mu} d\mu' = \frac{\mu - a}{b-a}, \tag{43}$$

in other words, the appropriate substitution is $\mu(x) = a + (b-a)x$. This can be applied by

$$I = \int_a^b f(\mu)d\mu = \int_a^b \frac{f(\mu)}{p(\mu)} \underbrace{p(\mu)d\mu}_{dx} = \int_0^1 \frac{f(\mu(x))}{p(\mu(x))}dx = (b-a)\int_0^1 f(\mu(x))dx \tag{44}$$

which is then approximated as

$$I = \int_a^b f(\mu)d\mu \approx \frac{b-a}{N}\sum_{i=1}^{N} f(\mu(x_i)) \tag{45}$$

where N is the number of sample sets taken from $p(x) = 1$.

3. Multi-dimensional Monte Carlo Integration

The formalism developed so far is easily generalized to n-dimensions, (again, just a reminder: the notation is explained in appendix VII A). Suppose $\Omega = [a_1, b_1] \times \cdots \times [a_n, b_n] \subseteq \mathbb{R}^n$ and $\Gamma = [0,1]^n \subset \mathbb{R}^n$. Suppose further that $p(\boldsymbol{\mu}) : \Omega \to \Gamma$ is a smooth probability distribution. Then

$$\int_\Omega f(\boldsymbol{\mu}) d^n\boldsymbol{\mu} = \int_\Omega \frac{f(\boldsymbol{\mu})}{p(\boldsymbol{\mu})} p(\boldsymbol{\mu}) d^n\boldsymbol{\mu} = \int_\Gamma \frac{f(\boldsymbol{\mu}(\mathbf{x}))}{p(\boldsymbol{\mu}(\mathbf{x}))} d^n\mathbf{x} \equiv \int_\Gamma g(\boldsymbol{\mu}(\mathbf{x})) d^n\mathbf{x}, \tag{46}$$

which in turn can be approximated as [8]:

$$\int_\Gamma g(\boldsymbol{\mu}(\mathbf{x})) d^n\mathbf{x} \approx V_\Gamma \langle g \rangle \pm V_\Gamma \sqrt{\frac{\langle g^2 \rangle - \langle g \rangle^2}{N}}, \tag{47}$$

where

$$\langle g^p \rangle \equiv \frac{1}{N} \sum_{i=1}^{N} \left[g(\boldsymbol{\mu}(\mathbf{x}_i)) \right]^p. \tag{48}$$

and the standard deviation of the mean σ is defined as

$$\sigma \equiv \sqrt{\frac{\langle g^2 \rangle - \langle g \rangle^2}{N}}. \tag{49}$$

We will interchangeably refer to σ as the variance or standard deviation of the mean. Note especially that $V_\Gamma = 1$ since Γ is the unit-cube defined on $\mathbb{R}^n$.

In the special case where the multivariate probability distribution $p(\boldsymbol{\mu}) : \Omega \to \Gamma$ is uniform, we obtain

$$\int_\Omega f(\boldsymbol{\mu}) d^n\boldsymbol{\mu} = \prod_{i=1}^{n} (b_i - a_i) \int_\Gamma f(\boldsymbol{\mu}(\mathbf{x})) d^n\mathbf{x} \approx \prod_{i=1}^{n} (b_i - a_i) \left[\langle f \rangle \pm \sqrt{\frac{\langle f^2 \rangle - \langle f \rangle^2}{N}} \right]. \tag{50}$$

In the case of importance sampling, we revert back to the general relationship found in eq. (47). In general, finding the inverse $\boldsymbol{\mu}(\mathbf{x})$ can be a tricky business, so for our purposes we'll restrict ourselves to the case where $p(\boldsymbol{\mu}) = \prod_{i=1}^{n} p_i(\mu_i)$. In this case, the one-dimensional formalism carries over naturally such that

$$x_i(\mu_i) = \int_{a_i}^{\mu_i} p_i(\mu_i') d\mu_i', \tag{51}$$

with the inverse $\mu_i(x_i)$. Then eq. (47) can be simplified as

$$\int_\Gamma g(\boldsymbol{\mu}(\mathbf{x})) d^n\mathbf{x} = \int_\Gamma g(\mu_1(x_1), ..., \mu_n(x_n)) dx_1 \cdots dx_n. \tag{52}$$

4. Specializing the theory

In our particular case, the integral in spherical coordinates can be expressed as

$$\begin{aligned} I &= \int dr_1 d\theta_1 d\phi_1 dr_2 d\theta_2 d\phi_2 r_1^2 r_2^2 \sin\theta_1 \sin\theta_2 \frac{e^{-2\alpha(r_1+r_2)}}{\sqrt{r_1^2 + r_2^2 + -2r_1 r_2 \cos\theta_2}} \\ &= 8\pi^2 \int_0^\infty \int_0^\infty \int_0^\pi dr_1 dr_2 d\theta_2 \frac{e^{-2\alpha(r_1+r_2)} \sin\theta_2}{\sqrt{r_1^2 + r_2^2 - 2r_1 r_2 \cos\theta_2}} r_1^2 r_2^2. \end{aligned} \tag{53}$$

To apply importance sampling, we introduce the probability distributions $p_i(r_i) = Ae^{-2\alpha r_i}$, where A is an appropriate normalization constant. We'll shortly show how this simplifies the problem, but first we'll find the normalization constant:

$$\int_0^\infty P_i(r_i) dr_i = A \int_0^\infty e^{-2\alpha r_i} dr_i = \frac{A}{2\alpha} = 1, \tag{54}$$

evidently, $A = 2\alpha$. Denote the integrand as $f(r_1, r_2, \theta_2)$. Then

$$f(r_1, r_2, \theta_2) = \frac{1}{4\alpha^2} p_1(r_1) p_2(r_2) g(r_1, r_2, \theta_2), \tag{55}$$

where

$$g(r_1, r_2, \theta_2) \equiv \frac{r_1^2 r_2^2 \sin\theta_2}{\sqrt{r_1^2 + r_2^2 - 2r_1 r_2 \cos\theta_2}}. \tag{56}$$

Requiring that probabilities are conserved, we get:

$$x_i(r_i) = \int_0^{r_i} p_i(r_i') dr_i' = 2\alpha \int_0^{r_i} e^{-2\alpha r_i'} dr_i' = 1 - e^{-2\alpha r_i}, \tag{57}$$

which easily leads to the inverse

$$r_i(x_i) = -\frac{\ln(1 - x_i)}{2\alpha}, \qquad i = 1, 2. \tag{58}$$

Introducing the uniform probability distribution $p_3(\theta_2) = 1/\pi$, the same argument leads to $\theta_2(x_3) = \pi x_3$, where $x_1, x_2, x_3 \in [0, 1]$. Hence our integral can be rewritten as

$$\begin{aligned} I &= 8\pi^2 \int_0^\infty \int_0^\infty \int_0^\pi f(r_1, r_2, \theta_2) dr_1 dr_2 d\theta_2 \\ &= \frac{2\pi^2}{\alpha^2} \int_0^\infty \int_0^\infty \int_0^\pi dr_1 dr_2 d\theta_2 \frac{g(r_1, r_2, \theta_2)}{p(\theta_2)} p(r_1) p(r_2) p(\theta_2) dr_1 dr_2 d\theta_2 \\ &= \frac{2\pi^3}{\alpha^2} \int_0^1 \int_0^1 \int_0^1 g(r_1(x_1), r_2(x_2), \theta_2(x_3)) dx_1 dx_2 dx_3, \end{aligned} \tag{59}$$

which is then approximated with eq. (47).

E. Estimating the ground state energy of the helium atom

As explained in the introduction, the integral studied in this article shows up in quantum mechanics when studying the helium atom. The variational principle states that we can approximate the ground state energy of the system E_{GS} by computing the expectation value of the Hamiltonian $\langle H \rangle \equiv \langle \Psi | H | \Psi \rangle$. More precisely, it states that the expectation value of H yields an upper-bound [4]:

$$E_{\text{GS}} \leq \langle H \rangle . \tag{60}$$

To this end, we must let $\alpha \to Z/a$ in the trial wave function, where $Z = 2$ is the proton number of Helium and a is the Bohr radius. Furthermore, we need a normalization constant A such that the wavefunction indeed obeys the Born rule. In the formalism developed so far, we've set $a = 1$. But to compute $\langle H \rangle$, we must solve

$$\langle H \rangle = \int d^3\mathbf{r}_1 d^3\mathbf{r}_2 \left| A e^{-Z(r_1 + r_2)/a} \right|^2 \frac{1}{|\mathbf{r}_1 - \mathbf{r}_2|}, \tag{61}$$

where $a = 5.29177 \cdot 10^{-2}$ nm and $A = 8/(\pi a^3)$, as found in [5]. To solve eq. (61) we introduce the variable change $r_i' = r_i/a$, which yields

$$I' = |A|^2 a^2 \int d^3\mathbf{r}_1' d^3\mathbf{r}_2' \frac{\sin\theta_1 \sin\theta_2 e^{-2Z(r_1' + r_2')} a^4 r_1'^2 r_2'^2}{a\sqrt{r_1'^2 + r_2'^2 - 2r_1' r_2' \cos\theta_2}}. \tag{62}$$

We see that $I' = |A|^2 a^5 I$. So the energy is given by

$$E_{\text{GS}} \approx \langle H \rangle = \langle H_0 \rangle + \frac{e^2}{4\pi\epsilon_0} \cdot |A|^2 a^5 I = 8E_1 + \frac{e^2 a^5}{4\pi\epsilon_0} \left(\frac{8}{\pi a^3} \right)^2 I, \tag{63}$$

where $E_1 = -13.6$ eV is the ground state energy of the hydrogen atom. The experimental value is $E_{\text{GS}} = -79.0051538$ eV [3].

F. Relative error

In the analysis of the results generated from the different integration methods, we will look at the relative error between the computed and analytical result. We use the following definition for relative error

$$\epsilon = \left| \frac{C - A}{A} \right|, \tag{64}$$

where C is the computed result and A is the analytical answer.

G. Speedup

As a measure of the relative efficiency of our codes, we define the *speedup*

$$S = \frac{T}{T'}, \tag{65}$$

where S is the speedup, T and T' are the time used by two different implementations. As a convention, we'll typically assign T to the implementation we expect to be slower, and T' to the one we expect to be fastest. We also define the *average speedup* to be

$$\bar{S} = \frac{1}{N} \sum_{i=1}^{N} \frac{T_i}{T_i'}, \tag{66}$$

which tells us on average how much larger T is relative to T' over a dataset of N tuples (T_i, T_i').

III. ALGORITHMS

The algorithm for Gaussian Quadrature with a set of orthogonal polynomials $\{P_N(x)\}_{N=0}^{\infty}$ can be summarized in the following way:

Algorithm 1 Gaussian Quadrature

Compute weights $w^{(1)}, ..., w^{(d)}$ corresponding to the set of polynomials $\{P_N(x^{(i)})\}_{i=1}^{d}$
Compute zeros $x^{(1)}, ..., x^{(d)}$ of the set of polynomials $\{P_N(x^{(i)})\}_{i=1}^{d}$
$I = 0.0$;
for $i = 1, 2, ..., N$ **do**
    ⋮
    for $l = 1, 2, ..., N$ **do**
        $I = I + w_i^{(1)} \cdots w_l^{(d)} f(x_i^{(1)}, ..., x_l^{(d)})$; ▷ Adds the contribution to the integral

The algorithm for Monte Carlo integration can be summarized as follows:

Algorithm 2 Monte Carlo Integration

$I = 0.0$;
$\langle f^2 \rangle = 0$;
$J = |\mathbf{J}|$; ▷ Jacobi determinant
for $i = 1, 2, ..., N$ **do**; ▷ Iterates over each Monte Carlo sample
    $\mathbf{x} = (x_1, ..., x_d) \in [0, 1]^d$; ▷ Sampled from a uniform distribution $P(\mathbf{x}) : [0, 1]^d \to [0, 1]^d$
    $\boldsymbol{\mu}(\mathbf{x}) = (\mu_1(x_1), ..., \mu_d(x_d))$; ▷ Computes the mapping $\boldsymbol{\mu}(\mathbf{x})$ that pertains to $P(\boldsymbol{\mu}) : \Omega \to [0, 1]^d$
    $I = I + f(\boldsymbol{\mu}(\mathbf{x}))$; ▷ Adds the contribution from each Monte Carlo sample to the integral
    $\langle f^2 \rangle = \langle f^2 \rangle + [f(\boldsymbol{\mu}(\mathbf{x}))]^2$; ▷ Adds the contribution from each Monte Carlo sample to $\langle f^2 \rangle$
$I = JI/N$; ▷ The final value of the integral
$\langle f^2 \rangle = \langle f^2 \rangle / N$; ▷ Final value of $\langle f^2 \rangle$
$\langle f \rangle^2 = I^2$; ▷ Square of the expectation value of the integral
$\mathrm{Var}(f) = \langle f^2 \rangle - \langle f \rangle^2$; ▷ Variance of the integral.
$\sigma_m = J\sqrt{\mathrm{Var}(f)/N}$; ▷ Standard deviation of the mean

IV. RESULTS

A. Gauss-Legendre Quadrature

Through trial and error we ended up comparing two mesh points numbers, $N = 25, 29$, to see which generated the lowest relative error. In other words, we minimized $\xi(\lambda)$ for $\lambda_{N=25} \in [1, 3]$ and $\lambda_{N=29} \in [2.9, 3.9]$. FIG. 1 displays the results for the two values of N.

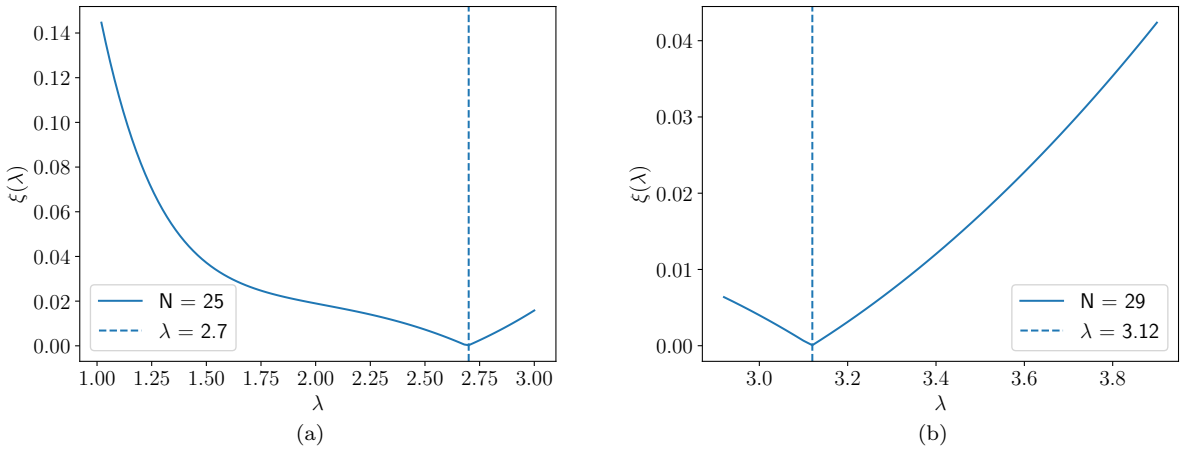

FIG. 1. Relative error as a function of the upper integral limit λ for $N = 25, 29$, for the Gauss-Legendre quadrature.

From the figure it can be seen that for $N = 25$ the optimal integration limits are $[-2.7, 2.7]$, for $N = 29$ the same is true for $[-3.12, 3.12]$. Calculating the integral between these values produces the results in table I.

N	I	ϵ_{rel}	Time used [s]	λ
25	0.192818	0.000274	5.57653	2.70
29	0.192779	$6.649 \cdot 10^{-5}$	14.8638	3.12

TABLE I. The table displays the value of the integral, relative error and time used when calculating eq. (5) for Gauss-Legendre with $N = 25$ on the interval $[-2.7, 2.7]$ and $N = 29$ on the interval $[-3.12, 3.12]$.

B. Gauss-Laguerre Quadrature

FIG. 2 displays the comparison of integral value and the CPU time used by the Gauss-Laguerre Quadrature for three and six dimensions.

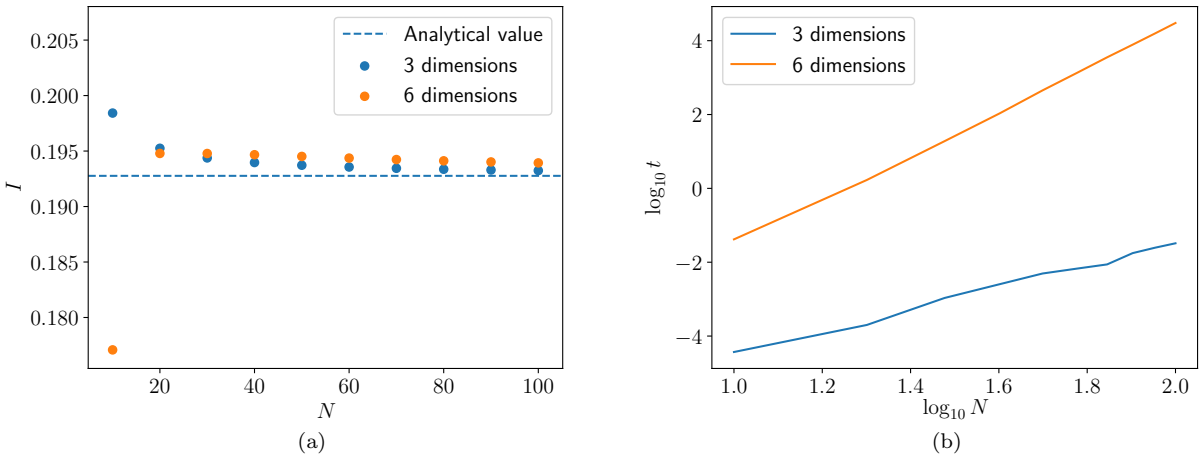

FIG. 2. (a): Comparison on numerical integral value I for three and six dimensions using Gauss-Laguerre quadrature. (b): CPU time t, in seconds, for different values of integration points N using Gauss-Lagurre quadrature.

Calculating the average speedup for the CPU data from FIG. 2, with eq. (66), we found that the three dimensional case on average is $2.77 \cdot 10^5$ times faster than the six dimensional integral.

The relative error for both integrals are graphically presented in FIG. 3.

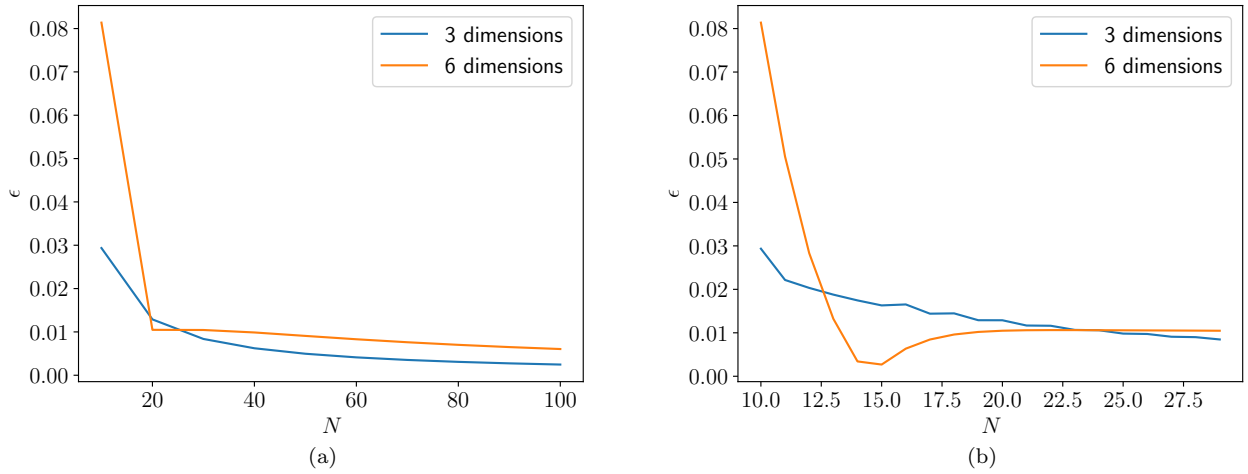

FIG. 3. Relative error for eq. (5) calculated with Gauss-Laguerre in three and six dimensions. Plot (b) is zoomed in around $N = 15$, to examine the drop in the error for $N = 20$ in (a).

The approximation in eq. (34) is clearly superior to the six dimensional case. Based on the results above we chose to only compare the three dimensional case with the Gauss-Legendre quadrature on the integration intervals $[-3.12, 3.12]^6$ for $N = 29$. The benchmarking was done graphically for the value of eq. (5), relative error and CPU time, and can be seen in FIG. 4.

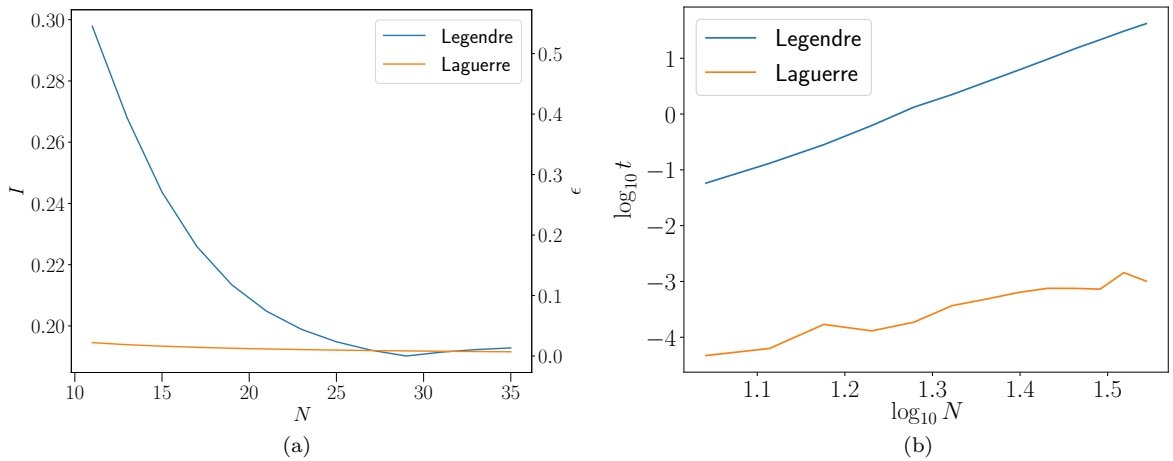

FIG. 4. (a): Integral value I and relative error ϵ for Gaussian quadrature for Legendre- and Laguerre-polynomials. (b): CPU time t for both methods.

C. Monte Carlo Integration

In this section, the results presented are the arithmetic mean of 1000 simulations for each sample size N. In FIG. 5, we represent the computed integrals and the corresponding standard deviations of the mean. Table II shows the relative error from the analytical value of the integral for varying sample size N for the different Monte Carlo methods. We observe that we get a relative error as low as 0.02% for the Monte Carlo method with importance sampling using parallelization.

N	ϵ_{BF} (MPI)	ϵ_{IS} (MPI)	ϵ_{BF}	ϵ_{IS}
100	0.913911	0.124510	0.951535	0.244956
1000	0.946413	0.294029	0.715102	0.000146
10000	0.643998	0.010075	0.544347	0.022455
100000	0.218807	0.012498	0.139551	0.007917
1000000	0.083045	0.006434	0.011047	0.003492
10000000	0.022140	0.000256	0.040594	0.001004

TABLE II. The table displays the relative error computed with the different Monte Carlo methods. The results are the arithmetic mean relative error taken over 10000 simulations of each Monte Carlo sample size N.

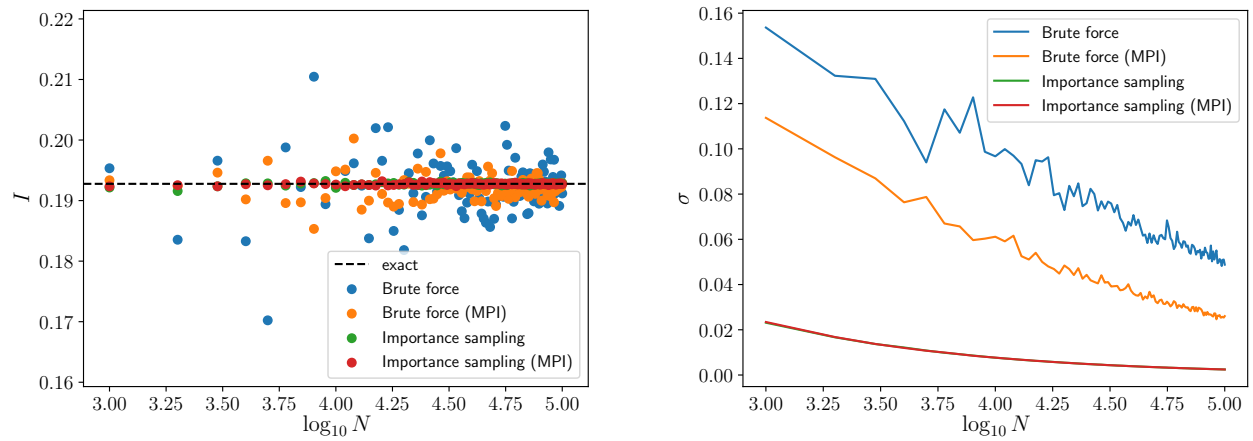

FIG. 5. The figure on the left shows the computed integrals I for each Monte Carlo method. The figure on the right shows the standard deviation σ of the mean for each Monte Carlo method. The results were computed by taking the arithmetic mean over 1000 simulations for each Monte Carlo sample size N.

In FIG. 6, we show the time used by each method and the acquired speedup of the parallelized code relative to the unparallelized code for two Monte Carlo methods respectively.

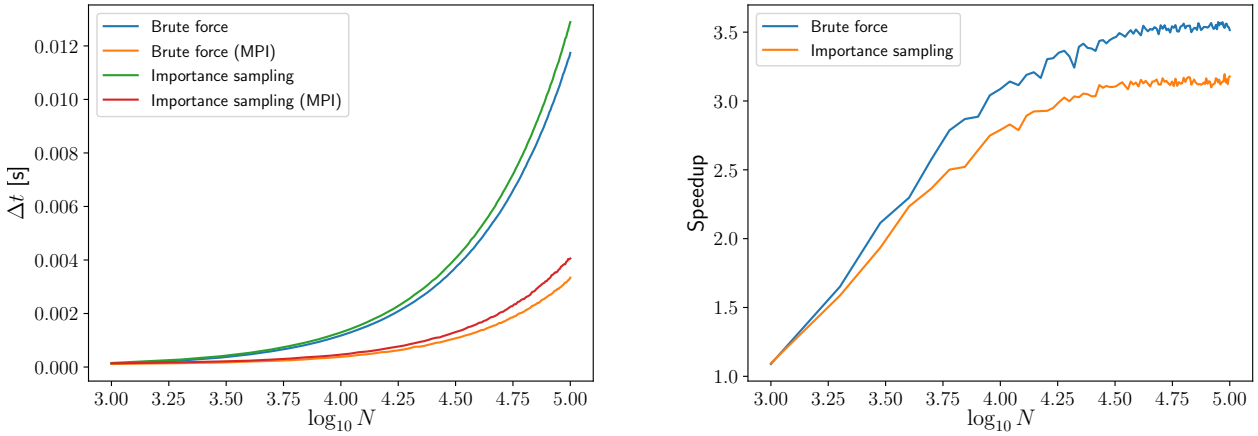

FIG. 6. The figure on the left shows the time used by each Monte Carlo method as a function of $\log_{10} N$, where N is the Monte Carlo sample size. The figure on the right shows the speedup acquired by the parallelized code relative to the unparallelized code with $p = 4$ processes run in parallel as a function of $\log_{10} N$. The results are the arithmetic mean of 1000 simulations for each sample size.

$\bar{S}$ (Brute Force)	$\bar{S}$ (Importance sampling)
3.4	3.0

TABLE III. The table shows the average speedup $\bar{S}$ obtained with the code parallelized with MPI. The average is computed taking the arithmetic mean of the data shown in FIG. 6.

In FIG. 7, we show the ratio $\sigma_{\mathrm{BF}}/\sigma_{\mathrm{IS}}$ as function of Monte Carlo sample size.

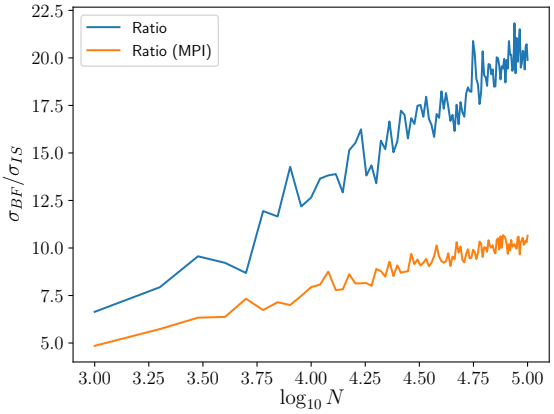

FIG. 7. The figure shows the ratio $\sigma_{\mathrm{BF}}/\sigma_{\mathrm{IS}}$ as a function of $\log_{10} N$, where N is the Monte Carlo sample size. The ratios are the arithmetic mean of 1000 simulations.

In table IV, we show the computed mean ratio of $\sigma_{\mathrm{BF}}/\sigma_{\mathrm{IS}}$ using the data depicted in FIG. 7.

$\sigma_{\mathrm{BF}}/\sigma_{\mathrm{IS}}$	$\sigma_{\mathrm{BF}}/\sigma_{\mathrm{IS}}$ (MPI)
17.3	9.3

TABLE IV. The table shows the mean ratio of the standard deviations of the mean with and without MPI. These were computed using the dataset shown in FIG. 7. σ_{BF} and σ_{IS} refers to the standard deviation of the mean of the BF and IS method, respectively.

D. The ground state energy of a Helium atom

We made a slight modification to the code with importance sampling (without parallelization) to test the variational principle. The results are shown in table V.

N	$\langle H \rangle$ [eV]	ϵ
100	-77.1170	0.023899
1000	-77.7398	0.016016
10000	-74.5396	0.056522
100000	-74.4078	0.058191
1000000	-74.8000	0.053227
10000000	-74.8294	0.052854

TABLE V. The table shows the computed values of $\langle H \rangle$ for several Monte Carlo samples N. ϵ is here the relative error between $\langle H \rangle$ and the experimental value of $E_{\mathrm{GS}} = -79.005151042(40)$ eV. Besides missing the mark by roughly 5%, we can note that the tabulated values obey the variational principle: $\langle H \rangle \geq E_{\mathrm{GS}}$.

V. DISCUSSION

A. Approximating infinity in Gauss-Legendre Quadrature

In the matter of deciding the N-values we were going to minimize the error function $\xi(\lambda)$ for, both accuracy and CPU time were important factors. Too small values lacks the accuracy, and too high values lacks the efficiency. As a compromise, the values $N = 25, 29$ were chosen. From the theory we would expect a more accurate result for a higher value of N, and this is exactly what is depicted in table I. We note that for $N = 29$, the integral in eq. (5) can be approximated with four leading digits, versus three for $N = 25$. The time needed to produce this level of accuracy is roughly three times longer than for the three digit accuracy. So upon approximating infinity, you have to

consider what is more important for your specific problem: time efficiency or accuracy. Ideally we could have run the simulation for bigger samples of Ns and λs, and estimated a more general result. This turns out to be relatively time consuming, so this would be a topic for further investigation on its own.

Running the Gauss-Legendre quadrature for a selected number of Ns, with the knowledge that the obtained values are a function of $\lambda(N)$, would therefore be pointless in our present case.

B. Optimized coordinate system for Gauss-Laguerre quadrature

From figures 2 and 3 we see that the results for the Gauss-Laguerre method for a three dimensional integral exceeds the six dimensional case by far. The obvious reason for this is related to the total number of iterations. By reducing the integral from six to three dimensions we run N^3 fewer iterations, where N corresponds to the number of zeros of $\mathcal{L}_N(x)$. This is clearly visible in the measured CPU time, where the average speedup was $2.77 \cdot 10^5$. For the relative error, both methods converge to a final result with three leading digits, but the error is generally larger for the six dimensional integral. The drop in relative error for the six dimensional case around $N = 15$ is due to the fact that for low values of N, the integral is ≈ 0. This can be observed in FIG. 2 (a). The methods calculate the same integral, so in theory the answers should be identical. Since it is calculated numerically, our results are as expected, since the computer round off error comes in to play. A larger number of iterations yields a larger round off error, and our results mirror this feature.

By redefining the coordinate system we are able to run the calculations with a substantially larger N, which again require a fraction of the CPU time needed compared to the six dimensional case. We are therefore able to obtain a more exact result in a substantially shorter time.

C. Gauss-Legendre- vs. Gauss-Laguerre Quadrature

From FIG. 4 it is evident that the Gauss-Laguerre quadrature is the preferable method when choosing between the two. The Laguerre method is faster and has a lower relative error, except in the case where N is closing in on $N = 29$. As mentioned in the preceding section V A, the Legendre method yields a relatively exact result around the N-value used to determine the integration limits $[-\lambda, \lambda]$ for which the integral is calculated with, before it rises again. From FIG. 2 (a) it can be seen that the Laguerre method converges to the analytical value for, relatively, high values of N, so it is fair to assume that the Laguerre method is better for high Ns, even though we only tested for an upper limit of $N = 35$. The shorter usage in CPU time is due to the fact that for the Gauss-Laguerre quadrature the integral is three dimensional, versus six dimensional as it is for the Gauss-Legendre quadrature. The difference in number of dimensions contributes to a higher error when calculating eq. (5) with Legendre polynomials.

The remaining results can easily be explained by the area of validity of the two orthogonal polynomials. The Legendre polynomials are defined over the interval $[-1, 1]$, so by approximating infinity with λ we implicitly take this into account in the Cartesian case, since the transformation $[a, b] \to [-1, 1]$ is only valid for a finite interval. Hence, we have already made an approximation prior to solving the integral, which hinders us in obtaining the analytical value. The function in eq. (5) would now be a linear combination of N Legendre polynomials, so even if $N \to \infty$, we still wouldn't be able to acquire the true value for the integral. By changing coordinate system and polynomials, we can utilize the entire interval for which the Laguerre polynomials are defined over for the radial parts in eq. (29). Furthermore are the intervals for the angular parts finite, so there are no need for approximations, and when $N \to \infty$ the integral in eq. (5) can be solved exact in theory.

D. The Monte Carlo methods

From FIG. 5, it's clear that IS yields faster convergence than applying BF. This is a sensible results, since IS is expected to reduce the variance and thus speed up convergence. Furthermore, we can infer that the standard deviations of the mean is reduced significantly when using IS as opposed to applying the BF method, consistent with the former argument. To reduce the variance is, naturally, the sole purpose of applying IS, and it's nice to see that it checks out in practice. Note especially the erratic behaviour of the computed value for the integral using the BF method. All the other methods seems to converge quickly while the the former method displays an unpredictable behaviour. This can possibly be because we haven't specified a different seed to the random number generator for each simulation and computing the arithmetic mean is not sufficient to remedy this problem. However, the other methods does not appear to suffer this problem to the same degree, so it could be that the problem resides with the method itself and not the implementation per se.

Studying FIG. 6, we see that the speedup gained from parallelization is roughly the same for low N, but for larger N the speedup is greatest for the brute force method. Nevertheless, we achieved an average speedup $\bar{S} \simeq 3$ with the BF code and $\bar{S} \simeq 3.4$ with the IS code. From the same figure, we see that the fastest code for a given N is the BF method run in parallel. However, it's only marginally faster than the IS method run in parallel. Furthermore, an analysis of the time used alone is of limited value seeing as we already found that the computed integral value of the latter is more accurate than that of the former for a given N. Therefore, IS becomes the better choice because you can run the code with fewer Monte Carlo samples than what you have to do with the BF code and thus reduce the time used to achieve a desired accuracy. To explain why the BF method indeed is faster, we have to specifically inspect the codes we've implemented. While BF relies on a simple linear mapping $\mu(x) = a + (b - a)x$, the code applying IS requires a call to the natural logarithm on every iteration, implying a slower code for a given N. Our point still stands, though: the code using IS reach convergence for smaller N and will generally be faster to reach a desired accuracy.

E. The approximation to the ground state energy of helium

If we inspect table V, we can deduce that $\langle H \rangle \geq E_{\text{GS}}$ for all N we present, which is just another confirmation of our implementation, since it obeys the variational principle. For large sample sizes N, the relative error stabilizes at roughly 5%, which is likely to be close to the best approximation we can achieve with the parameters used in the trial wave function. Inspecting the table for low values of N, we see that for $N = 1000$, we achieve the best approximation to the ground state energy. This does not, however, imply that in order to estimate the ground state energy, we ought to choose low values of N. Quite on the contrary, because in general there's no (as far as we can tell) way to know whether it will overshoot or undershoot the true value of the expectation value we want to approximate. In this case, it undershoots and gives us a slightly lower value than what is predicted with the parameters used and hence we achieve a lower relative error with respect to the ground state energy. We already know from the preceding sections that the relative error to the true value of the integral we solve actually is lower for larger N. In order to approximate the ground state energy better, we could for instance have treated the proton number Z as a parameter and minimized the expectation value of H with respect to this parameter. This is obviously not a groundbreaking idea, and in fact it is done here [6]. An even better approximation can be achieved if we introduce a larger set of parameters $\{\lambda_1, ..., \lambda_n\}$ and minimize $\langle H \rangle$ with respect to all of them. This, of course, requires a more complicated trial wave function and doing so in this experiment would be too time-consuming and we suggest that this is an excellent problem to research further as a project on its own.

VI. CONCLUSION

In this article we studied the Coulomb-repulsion term in the Hamiltonian for a helium atom, with two different approaches: Gaussian quadrature with Legendre- and Laguerre polynomials and Monte Carlo integration with brute force and importance sampling. In our comparison of the two Gaussian methods we found that an approximation of eq. (5) with Laguerre polynomials was superior to that of the Legendre polynomials. This result is partially attributed to the different domains of the applied polynomials act on, and to the fact that the Laguerre method runs N^3 fewer iterations due to the coordinate transformation performed to obtain a simpler integrand. The transformation from six to three dimensions is responsible for an average speedup of a factor $2.77 \cdot 10^5$. For both methods we obtained a satisfactory result of four leading digits, but the time consumption in attaining them were in favor of the Gass-Laguerre quadrature.

The main result of interest obtained by the Monte Carlo codes are the relative error $\epsilon \simeq 10^{-4}$ using importance sampling. Parallelization with MPI yielded an average speedup of $\bar{S} \simeq 3.4$ with the BF method and $\bar{S} \simeq 3.0$ with the IS code. The specialized use of PDFs for our integration reduced the variance on average by a factor ~ 17.3 in the unparallelized code and ~ 9.3 in the parallelized code, meaning importance sampling is a far better method with respect to accuracy. Finally, by application of the variational principle, we obtained a reliable relative error of $\sim 5\%$ with respect to the experimental value of ground state energy of the helium. A future problem of interest to research further, is to modify our implementation using Monte Carlo integration to handle a more complicated trial wave function with a large set of parameters, which can then be used minimize $\langle H \rangle$ in order to approximate the ground state more accurately.

VII. APPENDIX

A. Notation

We take some time to explain the notation used in the the development of the multi-dimensional formalism for the various integration methods study in this article. We define the hypercube $\Omega = [a_1, b_1] \times \cdots \times [a_n, b_n] \subseteq \mathbb{R}^n$ and the unit-cube $\Gamma = [0,1] \times \cdots \times [0,1] \equiv [0,1]^n \subset \mathbb{R}^n$. Then the we denote a multi-dimensional integral on Ω as

$$\int_\Omega f(\boldsymbol{\mu}) d^n \boldsymbol{\mu} \equiv \int_{a_1}^{b_1} \cdots \int_{a_n}^{b_n} f(\mu_1, ..., \mu_n) d\mu_1 \cdots d\mu_n. \tag{67}$$

Furthermore a vector-valued function $\boldsymbol{\mu}(\mathbf{x})$ is to be interpreted as

$$\boldsymbol{\mu}(\mathbf{x}) \equiv (\mu_1(\mathbf{x}), ..., \mu_n(\mathbf{x})) = (\mu_1(x_1, ..., x_n), ..., \mu_n(x_1, ..., x_n)). \tag{68}$$

B. Derivation of the closed-form solution of the expectation value of the relative particle distance

In this section, we'll derive the closed-form solution of the integral studied in this article. The integral is

$$\left\langle \frac{1}{|\mathbf{r}_1 - \mathbf{r}_2|} \right\rangle = \int d^3\mathbf{r}_1 d^3\mathbf{r}_2 \frac{|\Psi(\mathbf{r}_1, \mathbf{r}_2)|^2}{|\mathbf{r}_1 - \mathbf{r}_2|}, \tag{69}$$

with the unnormalized wave function

$$\Psi(\mathbf{r}_1, \mathbf{r}_2) = e^{-\alpha(r_1 + r_2)}. \tag{70}$$

We'll use spherical coordinates, so that $r_i \in [0, \infty)$, $\theta_i \in [0, \pi)$ and $\phi_i \in [0, 2\pi)$ for $i = 1, 2$. Aligning the polar axis along $\mathbf{r}_1$ (we'll integrate $\boldsymbol{r}_2$ first, so $\boldsymbol{r}_1$ is held fixed allowing this simple coordinate system, see FIG. 8), it follows that

$$|\mathbf{r}_1 - \mathbf{r}_2| = \sqrt{(\mathbf{r}_1 - \mathbf{r}_2) \cdot (\mathbf{r}_1 - \mathbf{r}_2)} = \sqrt{r_1^2 + r_2^2 - 2r_1 r_2 \cos\theta_2}, \tag{71}$$

meaning that our integral becomes

$$\left\langle \frac{1}{|\mathbf{r}_1 - \mathbf{r}_2|} \right\rangle = \int d^3\mathbf{r}_1 d^3\mathbf{r}_2 \frac{e^{-2\alpha(r_1 + r_2)}}{\sqrt{r_1^2 + r_2^2 - 2r_1 r_2 \cos\theta_2}}. \tag{72}$$

Explicitly, $d^3\mathbf{r}_j = dr_j d\theta_j d\phi_j r_j^2 \sin\theta_j$, meaning

$$\left\langle \frac{1}{|\mathbf{r}_1 - \mathbf{r}_2|} \right\rangle = \int dr_1 dr_2 d\theta_1 d\theta_2 d\phi_1 d\phi_2 r_1^2 r_2^2 \sin\theta_1 \sin\theta_2 \frac{e^{-2\alpha(r_1 + r_2)}}{\sqrt{r_1^2 + r_2^2 - 2r_1 r_2 \cos\theta_2}}. \tag{73}$$

There's no dependence on ϕ_i for $i = 1, 2$, nor on θ_1, so it follows that

$$\begin{aligned}
\left\langle \frac{1}{|\mathbf{r}_1 - \mathbf{r}_2|} \right\rangle &= 8\pi^2 \int dr_1 dr_2 d\theta_2 r_1^2 r_2^2 \frac{e^{-2\alpha(r_1 + r_2)}}{\sqrt{r_1^2 + r_2^2 - 2r_1 r_2 \cos\theta_2}} \sin\theta_2 \\
&= 8\pi^2 \int dr_1 dr_2 r_1^2 r_2^2 e^{-2\alpha(r_1 + r_2)} \underbrace{\int d\theta_2 \frac{\sin\theta_2}{\sqrt{r_1^2 + r_2^2 - 2r_1 r_2 \cos\theta_2}}}_{I_{\theta_2}(r_1, r_2)} \\
&= 8\pi^2 \int dr_1 dr_2 r_1^2 r_2^2 e^{-2\alpha(r_1 + r_2)} I_{\theta_2}(r_1, r_2).
\end{aligned} \tag{74}$$

It is instructive, then, to first solve $I_{\theta_2}(r_1, r_2)$:

$$I_{\theta_2}(r_1, r_2) = \int_0^\pi d\theta_2 \frac{\sin\theta_2}{\sqrt{r_1^2 + r_2^2 - 2r_1 r_2 \cos\theta_2}}. \tag{75}$$

We'll use the following substitution:

$$u(\theta_2) = r_1^2 + r_2^2 - 2r_1 r_2 \cos\theta_2 \tag{76}$$
$$u(0) = r_1^2 + r_2^2 - 2r_1 r_2 = (r_1 - r_2)^2 \tag{77}$$
$$u(\pi) = r_1^2 + r_2^2 + 2r_1 r_2 = (r_1 + r_2)^2 \tag{78}$$
$$du = 2r_1 r_2 \sin\theta_2 d\theta_2, \tag{79}$$

leading to

$$\begin{aligned} I_{\theta_2}(r_1, r_2) &= \frac{1}{2r_1 r_2}\int_{(r_1-r_2)^2}^{(r_1+r_2)^2} \frac{du}{\sqrt{u}} = \frac{1}{r_1 r_2}\sqrt{u}\Big|_{(r_1-r_2)^2}^{(r_1+r_2)^2} \\ &= \frac{1}{r_1 r_2}\left[\sqrt{(r_1+r_2)^2} - \sqrt{(r_1-r_2)^2}\right] \\ &= \frac{1}{r_1 r_2}\left[r_1 + r_2 - |r_1 - r_2|\right] = \begin{cases} \frac{2}{r_2}, & r_2 > r_1, \\ \\ \frac{2}{r_1}, & r_2 < r_1. \end{cases} \end{aligned} \tag{80}$$

Now back to the rest of the problem. Let $\lambda \equiv 2\alpha$.

$$\begin{aligned} \left\langle \frac{1}{|\mathbf{r}_1 - \mathbf{r}_2|} \right\rangle &= 8\pi^2 \int dr_1 dr_2 r_1^2 r_2^2 e^{-\lambda(r_1+r_2)} I_{\theta_2}(r_1, r_2) \\ &= 8\pi^2 \int dr_1 r_1^2 e^{-\lambda r_1} \int dr_2 r_2^2 e^{-\lambda r_2} I_{\theta_2}(r_1, r_2) \\ &= 16\pi^2 \int dr_1 r_1 e^{-\lambda r_1} \underbrace{\int_0^{r_1} dr_2 r_2^2 e^{-\lambda r_2}}_{I_1(r_1)} + 16\pi^2 \int dr_1 r_1^2 e^{-\lambda r_1} \underbrace{\int_{r_1}^{\infty} dr_2 r_2 e^{-\lambda r_2}}_{I_2(r_1)} \\ &= 16\pi^2 \int dr_1 r_1 e^{-\lambda r_1} I_1(r_1) + 16\pi^2 \int dr_1 r_1^2 e^{-\lambda r_1} I_2(r_2). \end{aligned} \tag{81}$$

We commence by computing $I_1(r_1)$:

$$\begin{aligned} I_1(r_1) &= \int_0^{r_1} dr_2 r_2^2 e^{-\lambda r_2} = \int_0^{r_1} dr_2 \frac{\partial^2}{\partial\lambda^2} e^{-\lambda r_2} = \frac{\mathrm{d}^2}{\mathrm{d}\lambda^2}\int_0^{r_1} dr_2 e^{-\lambda r_2} \\ &= -\frac{\mathrm{d}^2}{\mathrm{d}\lambda^2}\frac{1}{\lambda} e^{-\lambda r_2}\Big|_0^{r_1} = \frac{\mathrm{d}^2}{\mathrm{d}\lambda^2}\frac{1}{\lambda}\left[1 - e^{-\lambda r_1}\right] \\ &= \frac{e^{-\lambda r_1}\left(-r_1^2\lambda^2 - 2r_1\lambda + 2e^{\lambda r_1} - 2\right)}{\lambda^3} \end{aligned} \tag{82}$$

Similarly, we solve $I_2(r_1)$:

$$\begin{aligned} I_2(r_1) &= \int_{r_1}^{\infty} dr_2 r_2 e^{-\lambda r_2} = -\frac{\mathrm{d}}{\mathrm{d}\lambda}\int_{r_1}^{\infty} dr_2 e^{-\lambda r_2} = \frac{\mathrm{d}}{\mathrm{d}\lambda}\frac{1}{\lambda} e^{-\lambda r_2}\Big|_{r_1}^{\infty} = -\frac{\mathrm{d}}{\mathrm{d}\lambda}\frac{1}{\lambda} e^{-\lambda r_1} \\ &= \frac{e^{-\lambda r_1}(r_1\lambda + 1)}{\lambda^2}. \end{aligned} \tag{83}$$

Combining the results, we get

$$\begin{aligned}\left\langle \frac{1}{|\mathbf{r}_1-\mathbf{r}_2|}\right\rangle &= 16\pi^2\int_0^\infty dr_1 r_1 e^{-\lambda r_1} I_1(r_1) + 16\pi^2\int_0^\infty dr_1 r_1^2 e^{-\lambda r_1} I_2(r_1)\\ &= 16\pi^2\int_0^\infty dr_1 r_1 e^{-\lambda r_1}\frac{e^{-\lambda r_1}\left(-r_1^2\lambda^2 - 2r_1\lambda + 2e^{\lambda r_1} - 2\right)}{\lambda^3}\\ &+ 16\pi^2\int_0^\infty dr_1 r_1^2 e^{-\lambda r_1}\frac{e^{-\lambda r_1}(r_1\lambda+1)}{\lambda^2}\\ &= \frac{16\pi^2}{\lambda^3}\left[2\underbrace{\int_0^\infty dr_1 r_1 e^{-\lambda r_1}}_{\mathcal{I}_1} - \lambda\underbrace{\int_0^\infty dr_1 r_1^2 e^{-2\lambda r_1}}_{\mathcal{I}_2} - 2\underbrace{\int_0^\infty dr_1 r_1 e^{-2\lambda r_1}}_{\mathcal{I}_3}\right].\end{aligned} \tag{84}$$

We proceed by solving $\mathcal{I}_1$, $\mathcal{I}_2$ and $\mathcal{I}_3$ separately by differentiation under the integral sign. For convenience, we define $\gamma \equiv 2\lambda$

$$\mathcal{I}_1 = \int_0^\infty dr_1 r_1 e^{-\lambda r_1} = -\frac{\mathrm{d}}{\mathrm{d}\lambda}\int_0^\infty e^{-\lambda r_1} = \frac{\mathrm{d}}{\mathrm{d}\lambda}\frac{1}{\lambda}e^{-\lambda r_1}\Big|_0^\infty = -\frac{\mathrm{d}}{\mathrm{d}\lambda}\frac{1}{\lambda} = \frac{1}{\lambda^2}, \tag{85}$$

and

$$\mathcal{I}_2 = \int_0^\infty dr_1 r_1^2 e^{-\gamma r_1} = \frac{\mathrm{d}^2}{\mathrm{d}\gamma^2}\int_0^\infty dr_1 e^{-\gamma r_1} = \frac{\mathrm{d}^2}{\mathrm{d}\gamma^2}\frac{1}{\gamma} = \frac{2}{\gamma^3}, \tag{86}$$

and $\mathcal{I}_3$ is tantamount to $\mathcal{I}_1$ with $\lambda \to \gamma$, so $\mathcal{I}_3 = 1/\gamma^2$. Recalling that $\gamma = 2\lambda$ and $\lambda = 2\alpha$, and doing some simple algebraic manipulations, we obtain

$$\left\langle \frac{1}{|\mathbf{r}_1-\mathbf{r}_2|}\right\rangle = \frac{5\pi^2}{8\alpha^5}\Big|_{\alpha=2} = \frac{5\pi^2}{256}. \tag{87}$$

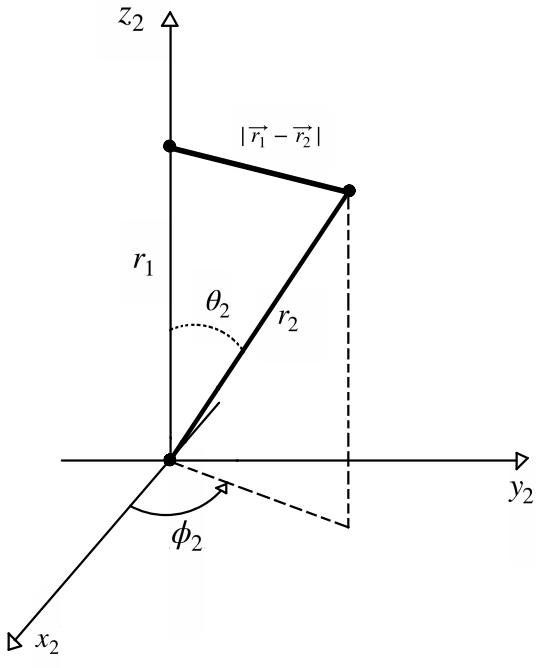

FIG. 8. Aligning the polar axis, in the $\mathbf{r}_2$ system, along $\mathbf{r}_1$ yields the coordinate system pictured above. This reduces the integral in eq. (29) to a three dimensional problem.

[1] https://github.com/reneaas/ComputationalPhysics/tree/master/projects/project3.
[2] https://github.com/CompPhysics/ComputationalPhysics/blob/master/doc/Projects/2019/Project3/pdf/Project3.pdf.
[3] https://en.wikipedia.org/wiki/Helium_atom.
[4] Darrel F. Schroeter David J. Griffiths. *Introduction to Quantum Mechanics, 3rd ed.*, chapter 8.1, page 327. 2018.
[5] Darrel F. Schroeter David J. Griffiths. *Introduction to Quantum Mechanics, 3rd ed.*, chapter 8.2, page 333. 2018.
[6] Darrel F. Schroeter David J. Griffiths. *Introduction to Quantum Mechanics, 3rd ed.*, chapter 8.2, pages 335–336. 2018.
[7] Morten Hjorth-Jensen. *Computational Physics, Lecture Notes Fall 2015*, chapter 5.3. 2015.
[8] William T. Vetterling Brian P. Flannery William H. Press, Saul A. Teukolsky. *Numerical Recipes. The Art of Scientific Computing, 3rd edition*, chapter 7.7. Cambridge University Press, 2007.