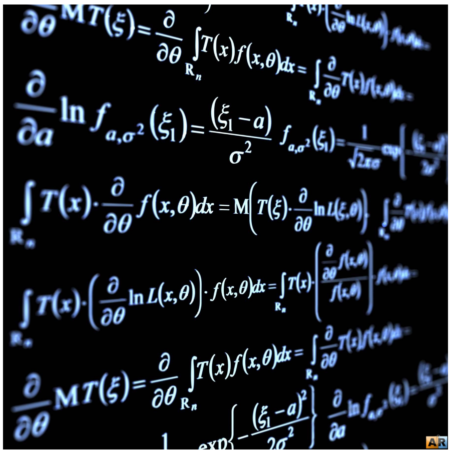

Project 3

FYS3150 - Computational physics

Quantum dots

Author:
Vidar Skogvoll

November 24, 2014

Abstract

Here is a short summary of the project.

Contents

1 Introduction

Quantum mechanics is an exciting field.

2 Theory

Here is all the theory needed to understand the project.

2.1 The physical system

This is the section explaining the physics of the system. Throughout the project, *natural units* are used ($\hbar = 1$,$c = 1$,$e = 1$,$m_e = 1$) and all energies are in so-called *atomix units* a.u.

2.1.1 The quantum mechanics and the variational principle

2.1.1.1 The quantum mechanics

In this project we will look at a system of N electrons in a so-called *quantum dot.* That is, a two dimensional harmonic oscillator with potential

$$V(\vec{r}) = \frac{1}{2}\omega^2 r^2 \tag{2.1.1}$$

This potential gives rise to a multi-particle Hamiltonian $\hat{H}$ given as the sum of an ordinary Hamiltonian and an electron repulsive part

$$\hat{H} = \sum_{i=1}^{N}\left(-\frac{1}{2}\nabla_i^2 + \frac{1}{2}\omega^2 r_i^2\right) + \sum_{i<j}\frac{1}{r_{ij}} \tag{2.1.2}$$

Where $r_{ij} = |\vec{r}_i - \vec{r}_j|$ is the distance between the electrons i and j and $r_i = |\vec{r}_i| = \sqrt{x_i^2 + y_i^2}$ when $\vec{r}_i = \begin{pmatrix} x_i \\ y_i \end{pmatrix}$. Our goal in this project is to find the ground eigenstate and energy of this multi-particle Hamiltonian numerically.

2.1.1.2 The variational principle

We will approach this by constructing a real test function $\Psi_T(\vec{r}_0, \vec{r}_1, ..., \vec{r}_{N-1}, \alpha, \beta)$ dependent on two parameters α and β and calculate the expextation value of the hamilton operator $\langle\hat{H}\rangle$. As we know, the orthonormal eigenstates Ψ_i of the Hamiltonian forms a complete basis, so any state, including our test state Ψ_T, can be written as a linear combination of the eigenstates

$$\Psi_T = \sum_i c_i \Psi_i \tag{2.1.3}$$

Inserting this expression into the equation for the expectation value of $\hat{H}$ gives

$$\langle\hat{H}\rangle = \frac{\int \Psi_T \hat{H} \Psi_T d\vec{r}}{\int \Psi_T \Psi_T d\vec{r}} = \frac{\int \left(\sum_i c_i^* \Psi_i^*\right) \hat{H} \left(\sum_i c_i \Psi_i\right) d\vec{r}}{\int \left(\sum_i c_i^* \Psi_i^*\right) \left(\sum_i c_i \Psi_i\right) d\vec{r}} = \frac{\int \left(\sum_i c_i^* \Psi_i^*\right) \left(\sum_i c_i E_i \Psi_i\right) d\vec{r}}{\int \left(\sum_i c_i^* \Psi_i^*\right) \left(\sum_i c_i \Psi_i\right) d\vec{r}}$$
$$= \frac{\sum_i |c_i|^2 E_i}{\sum_i |c_i|} \tag{2.1.4}$$

The energy of the ground state E_0 is smaller than all other E_i's so

$$\frac{\sum_i |c_i|^2 E_i}{\sum_i |c_i|} \geq \frac{\sum_i |c_i|^2 E_0}{\sum_i |c_i|} = E_0 \frac{\sum_i |c_i|^2}{\sum_i |c_i|} = E_0 \tag{2.1.5}$$
$$\langle H \rangle \geq E_0 \tag{2.1.6}$$

This simple observation is called *the variational principle* and is what we will use to narrow our search for the optimal parameters α and β. We will look for the parameters α and β that gives us the smallest value of $\langle\hat{H}\rangle$ and this will be our estimate for the ground state energy.

2.1.1.3 Finding the expectation value of $\hat{H}$

We have

$$\langle\hat{H}\rangle = \frac{\int \Psi_T \hat{H} \Psi_T d\vec{r}}{\int \Psi_T \Psi_T d\vec{r}} = \int \frac{\Psi_T \Psi_T}{\int \Psi_T \Psi_T d\vec{r}} \frac{1}{\Psi_T} \hat{H} \Psi_T d\vec{r} \tag{2.1.7}$$

If we rename probability density function of the particles

$$\frac{\Psi_T \Psi_T}{\int \Psi_T \Psi_T d\vec{r}} = P(\vec{r}) \tag{2.1.8}$$

And introduce the local energy

$$E_L(\vec{r}) = \frac{1}{\Psi_T} \hat{H} \Psi_T \tag{2.1.9}$$

The integral becomes

$$\langle\hat{H}\rangle = \int P(\vec{r}) E_L(\vec{r}) d\vec{r} = \langle E_L \rangle \tag{2.1.10}$$

Thus, to calculate the expectation value of $\hat{H}$ we can just calculate the expectation value of the local energy.

2.1.1.4 Verifying that we have found an eigenstate

We could very well find a minimum of $\langle\hat{H}\rangle$ that is not an eigen energy of the system, i.e. still larger than E_0. To address this problem, let's look at the variance V_{E_L} of $\langle E_L \rangle$.

$$V_{E_L} = \langle E_L^2 \rangle - \langle E_L \rangle^2 = \int P(\vec{r}) \left(\frac{1}{\Psi_T} \hat{H} \Psi_T \right)^2 d\vec{r} - \left(\int P(\vec{r}) \frac{1}{\Psi_T} \hat{H} \Psi_T d\vec{r} \right)^2 \tag{2.1.11}$$

Since

$$\langle (E_L - \langle E_L\rangle)^2\rangle = \langle E_L^2 - 2E_L\langle E_L\rangle + \langle E_L\rangle^2\rangle = \langle E_L^2\rangle - 2\langle E_L\rangle\langle E_L\rangle + \langle E_L\rangle^2$$
$$= \langle E_L^2\rangle - \langle E_L\rangle^2 = V_{E_L} \tag{2.1.12}$$

We have that if $V_{E_L} = 0$, then

$$\langle (E_L - \langle E_L\rangle)^2\rangle = 0 \tag{2.1.13}$$
$$0 = \int \left(P(\vec{r})E_L(\vec{r}) - \int P(\vec{r})E_L(\vec{r})d\vec{r}\right)^2 d\vec{r} = \int \left(\frac{1}{\Psi_T}\hat{H}\Psi_T - \int \Psi_T\hat{H}\Psi_T d\vec{r}\right)^2 d\vec{r} \tag{2.1.14}$$

When the Hamiltonian acts on a real function, it gives a real function. And since Ψ_T is real $\frac{1}{\Psi_T}\hat{H}\Psi_T - \int \Psi_T\hat{H}\Psi_T d\vec{r}$ is real. By consequence

$$\left(\frac{1}{\Psi_T}\hat{H}\Psi_T - \int \Psi_T\hat{H}\Psi_T d\vec{r}\right)^2 > 0 \tag{2.1.15}$$

Which means that for the integral in equation 2.1.14 to be zero, the following must be true for all $\vec{r}$

$$\frac{1}{\Psi_T}\hat{H}\Psi_T - \int \Psi_T\hat{H}\Psi_T d\vec{r} = 0 \tag{2.1.16}$$
$$\Psi_T\hat{H}\Psi_T = \int \Psi_T\hat{H}\Psi_T d\vec{r} \tag{2.1.17}$$

The right hand side of this equation is just a real number. Naming this number E gives

$$\frac{1}{\Psi_T}\hat{H}\Psi_T = E \tag{2.1.18}$$
$$\hat{H}\Psi_T = E\Psi_T \tag{2.1.19}$$

Which is nothing but the eigenvalue equation stating that Ψ_T is an eigenstate of $\hat{H}$. This will serve as a test to see if the state we have found when minimizing the expectation value of E_L is an eigenstate of the Hamilton operator. It is just as easily (perhaps easier) shown that if Ψ_T *is* and eigenstate of $\hat{H}$, then the variance of the local energy is 0. This means that if the variance of our test function is *not* 0, then it is *not* an eigenfunction of $\hat{H}$. We can summarize this discussion as follows

$$\boxed{V_{E_L} = 0 \quad \text{if and only if} \quad \Psi_T \text{ is an eigenstate of } \hat{H}} \tag{2.1.20}$$

2.1.2 The test function Ψ_T

We will in this project use the trial wavefunctions of $\vec{r}_i = \begin{pmatrix} x_i \\ y_i \end{pmatrix}$ given by

$$\Psi_T(\vec{r}_0, ..., \vec{r}_{N-1}) = \mathrm{Det}_M(\phi_0, ..., \phi_{N-1}) \cdot \mathrm{J}(\vec{r}_0, ..., \vec{r}_{N-1}) \tag{2.1.21}$$

2.1.2.1 The modified Slater determinant $\mathrm{Det}_M(\phi_0, ..., \phi_{N-1})$

$\mathrm{Det}_M(\phi_0, ..., \phi_{N-1})$ is a modified *Slater determinant* defined as

$$\mathrm{Det}_M(\phi_0, ..., \phi_{N-1}) = |U| \cdot |D|$$
$$\begin{vmatrix} \phi_0(\vec{r}_0) & \phi_2(\vec{r}_0) & ... & \phi_{N-2}(\vec{r}_0) \\ \phi_0(\vec{r}_2) & \phi_2(\vec{r}_2) & ... & \phi_{N-2}(\vec{r}_2) \\ ... & ... & ... & ... \\ \phi_0(\vec{r}_{N-2}) & \phi_2(\vec{r}_{N-2}) & ... & \phi_{N-2}(\vec{r}_{N-2}) \end{vmatrix} \cdot \begin{vmatrix} \phi_1(\vec{r}_1) & \phi_3(\vec{r}_1) & ... & \phi_{N-1}(\vec{r}_1) \\ \phi_1(\vec{r}_3) & \phi_3(\vec{r}_3) & ... & \phi_{N-1}(\vec{r}_3) \\ ... & ... & ... & ... \\ \phi_1(\vec{r}_{N-1}) & \phi_3(\vec{r}_{N-1}) & ... & \phi_{N-1}(\vec{r}_{N-1}) \end{vmatrix} \tag{2.1.22}$$

Where $\phi_i(\vec{r}_i)$ is a wavefunction resembling one of the eigenfunctions of the Hamilton operator for *one* particle in a two dimensional harmonic oscillator, but parameterized by α in the following way:

$$\phi_i(\vec{r}_j) = H_{n_x}(\sqrt{\alpha\omega}x_j)H_{n_y}(\sqrt{\alpha\omega}y_j)\exp(-\alpha\omega(x^2 + y^2)/2) \tag{2.1.23}$$

The reason for this modified version of the Slater determinant (whose real form can be explored elsewhere[1]) is that the spin parts of the wavefunctions are not incorporated in the expressions of $\phi_i(\vec{r}_j)$. The result is that if we were to insert these into a regular Slater determinant we would get 0, everytime. The modified Slater determinant Det_M avoids this issue while still conserving some of the most important properties of the Slater determinant.

$n_x(i)$ and $n_y(i)$ corresponds to the quantum numbers needed to "fill up" the system from the lowest energy levels twice (one for each spin configuration). For $i < 12$, the explicit depence of n_x, n_y on i is given in table 2.1.1.

$i =$	0	1	2	3	4	5	6	7	8	9	10	11
$n_x =$	0	0	1	1	0	0	2	2	1	1	0	0
$n_y =$	0	0	0	0	1	1	0	0	1	1	2	2

Table 2.1.1: The explicit dependence of n_x and n_y on i in the construction of the trial wavefunctions.

2.1.2.2 The Jastrow factor $\mathrm{J}(\vec{r}_0, ..., \vec{r}_{N-1})$

$\mathrm{J}(\vec{r}_0, ..., \vec{r}_{N-1})$ is a so-called *Jastrow factor*, which represents the electron repulsion part of the wavefunction, defined as

$$\boxed{\begin{gathered} \mathrm{J}(\vec{r}_0, ...\vec{r}_{N-1}) = \textstyle\prod_{i<j}^{N} \exp\left(\dfrac{a_{ij}r_{ij}}{1+\beta r_{ij}}\right) \\ \text{where} \quad r_{ij} = |\vec{r}_i - \vec{r}_j| \quad \text{and} \quad a_{ij} = \begin{cases} 1/3 & \text{if spin(i) and spin(j) are parallell} \\ 1 & \text{if spin(i) and spin(j) are anti-parallell} \end{cases} \end{gathered}} \tag{2.1.24}$$

[1] http://en.wikipedia.org/wiki/Slater_determinant

From the expression of the Jastrow factor, we see that it is zero whenever $r_{ij} = 0$ for any pair $i \neq j$. This is what we want from such a factor, namely that the wavefunction is zero whenever there is no distance between two particles (i.e. $r_{ij} = 0$).

2.1.2.3 Motivation

The motivation of such a trial wavefunction is that it is partly made of the unperturbed harmonic oscillator ground states. See figure 2.1.1.

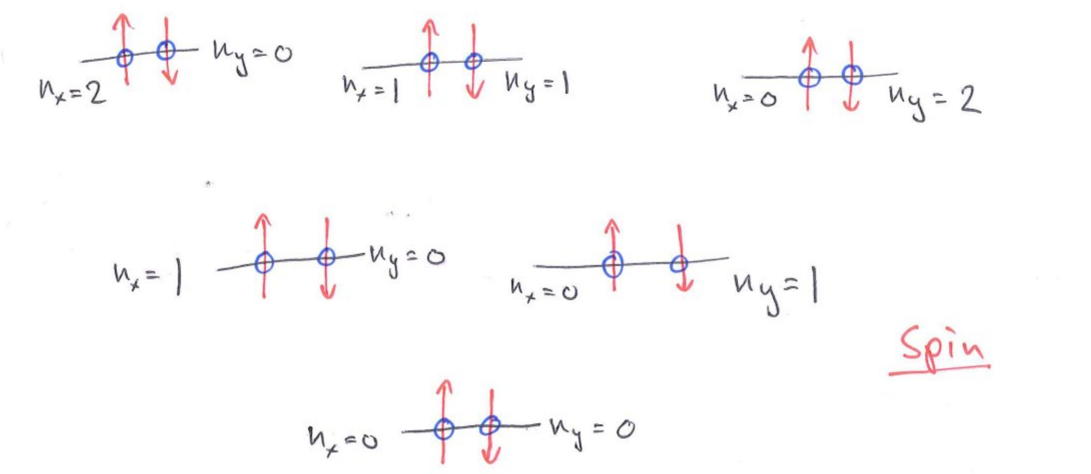

Figure 2.1.1: The wave function levels. The blue dots represent electrons filling up the states from $n_x = n_y = 0$ to the third lowest states. 2 electrons fill up the lowest level, 6 electrons the two lowest levels and 12 electrons all three first energy levels. The net spin of each level must be 0 due to the Pauli Exclusion principle.

The energies of these levels are given by the well known 2D-harmonic oscillator energy formula

$$E_{n_x,n_y} = \hbar\omega(1 + n_x + n_y) \tag{2.1.25}$$

So if we have two electrons in the lowest state ($n_x = n_y = 0$), we would expect the energy of this state to be

$$2 \cdot \hbar\omega(1 + 0 + 0) = 2\hbar\omega = 2\omega \tag{2.1.26}$$

When using natural units.

To have two electrons in the same state, their spin must be opposite due to the Pauli exclusion principle. This means that the total spin of the $n_x = n_y = 0$ state is 0. It can be shown that the wavefunction given by equation 2.1.21 when $N = 2$ (i.e. two electrons) is given by the following equation

$$\Psi_T(\vec{r}_0, \vec{r}_1) = \exp(-\alpha\omega(r_0^2 + r_1^2)/2) \cdot \mathrm{J} \tag{2.1.27}$$

If we do not include the repulsion part of the system, there is no need to include the Jastrow Factor (for details, see section 2.1.2.2). It can be shown that this state is an eigenstate of the unperturbed harmonic oscillator with energy 2ω when $\alpha = 1$. This analogy can be extended to the $N = 6$ and $N = 12$ electron case as well. When $N = 6$ the unperturbed energy should be the sum of the energies in the first level (i.e. 2ω) and the collective energies of the electrons in the second level states ($n_x = 1,\ n_y = 0 \vee n_x = 0,\ n_y = 1$) which is $4 \cdot 2\omega = 8\omega$, resulting in a total energy of 10ω. Equivalently, the energy for the $N = 12$ electron case should be 28ω. These energies will all serve as benchmarks and we should get the exact results when there is no repulsion, $\alpha = 1$ and the jastrow factor is omitted.

2.1.2.4 Closed form expression of the local energy

To evaluate the local energy

$$E_L(\vec{r}) = \frac{1}{\Psi_T}\hat{H}\Psi_T = \frac{1}{\Psi_T}\left(\sum_i -\frac{1}{2}\nabla_i^2 + \sum_i V(\vec{r}_i)\right)\Psi_T = \sum_i V(\vec{r}_i) - \frac{1}{2}\sum_i \frac{1}{\Psi_T}\nabla_i^2\Psi_T \quad (2.1.28)$$

We need a lot of computational power. This is mainly due to the fact that we need to compute the sum of the laplacian operators on each particle. This is an easy task to do "brute force", but if we were able to find an analytical expression for the local energy, it would possibly simplify calculations by alot. Let's look at one of the terms in the laplacian sum, naming it LSP (Laplacian sum part)

$$LSP = \frac{1}{\Psi_T}\nabla_i^2\Psi_T \quad (2.1.29)$$

Inserting the trial wavefunction expression (equation 2.1.21) into the latter gives

$$\frac{1}{\Psi_T}\nabla_i^2\Psi_T = \frac{1}{\mathrm{Det}_M\mathrm{J}}\nabla_i^2(\mathrm{Det}_M\mathrm{J}) \quad (2.1.30)$$

The function Det_M is a product of two matrice-determinants $|U|$ and $|D|$ where $|U|$ handles all the particles assigned spin up and $|D|$ the ones assigned spin down. Particle i has either spin up or down, so let $|S_i|$ denote the matrix determinant which handles particle i and $|S_{j\neq i}|$ denote the one that doesn't, then

$$\frac{1}{\Psi_T}\nabla_i^2\Psi_T = \frac{1}{|S_i||S_{j\neq i}|\mathrm{J}}\nabla_i^2(|S_i||S_{j\neq i}|\mathrm{J}) = \frac{1}{|S_i||S_{j\neq i}|\mathrm{J}}|S_{j\neq i}|\nabla_i^2(|S_i|\mathrm{J}) = \frac{1}{|S_i|\mathrm{J}}\nabla_i^2(|S_i|\mathrm{J}) \quad (2.1.31)$$

Using the product rule of the laplacian operator gives

$$\boxed{\frac{1}{\Psi_T}\nabla_i^2\Psi_T = \frac{\nabla_i^2|S_i|}{|S_i|} + \frac{\nabla_i^2\mathrm{J}}{\mathrm{J}} + 2\frac{\nabla_i\mathrm{J}}{\mathrm{J}} \cdot \frac{\nabla_i|S_i|}{|S_i|}} \quad (2.1.32)$$

It is possible to find analytical expression for all these terms, and that has been already been done in a master thesis written by Jørgen Høgberget [3]. The arguments will not be repeated but the results are as follows (with names added for code reference)

$$\boxed{\begin{aligned}
NSS &= \frac{\nabla_i |S_i|}{|S_i|} = \sum_{k=0}^{N/2} \left[(S^{-1})_{ki} \cdot \nabla_i \phi_{2k}(\vec{r}_i)\right] && (2.1.33a)\\
N2SS &= \frac{\nabla_i^2 |S_i|}{|S_i|} = \sum_{k=0}^{N/2} \left[(S^{-1})_{ki} \cdot \nabla_i^2 \phi_{2k}(\vec{r}_i)\right] && (2.1.33b)\\
NJJ &= \frac{\nabla_i \mathrm{J}}{\mathrm{J}} = \sum_{k\neq i} \frac{a_{ik}}{r_{ik}} \frac{\vec{r}_i - \vec{r}_k}{(1+\beta r_{ik})^2} && (2.1.33c)\\
N2JJ &= \frac{\nabla_i^2 \mathrm{J}}{\mathrm{J}} = \left|\frac{\nabla_i J}{J}\right|^2 + \sum_{k\neq i} \frac{a_{ik}}{r_{ik}} \frac{1-\beta r_{ik}}{(1+\beta r_{ik})^3} && (2.1.33d)
\end{aligned}}$$

$\nabla_i \phi_k(\vec{r}_i)$ and $\nabla_i^2 \phi_k(\vec{r}_i)$ scan be found simply by inserting and taking the derivative of the Hermite polynomials. The explicit formulas for ϕ_k for k in the range 0 to 11 is given in table 2.1.2 and are also taken from the master thesis by Jørgen Høgberget.

k	(n_x, n_y)	$\phi_k(\vec{r})$	$\nabla_i \phi_k(\vec{r}_i) = [\nabla_x, \nabla_y]$	$\nabla_i^2 \phi_k(\vec{r}_i)$
0	(0,0)	1	$-\left[l^2 x, l^2 y\right]$	$l^2(l^2 r^2 - 2)$
2	(1,0)	$2lx$	$-2l\left[(lx-1)(lx+1), l^2 xy\right]$	$2l^3 x(l^2 r^2 - 4)$
4	(0,1)	$2ly$	$-2l\left[l^2 xy, (ly-1)(ly+1)\right]$	$2l^3 y(l^2 r^2 - 4)$
6	(2,0)	$4l^2 x^2 - 2$	$-2\left[l^2 x(2l^2 x^2 - 5), l^2 y(2l^2 x^2 - 1)\right]$	$2l^2(l^2 r^2 - 6)(2l^2 x^2 - 1)$
8	(1,1)	$4l^2 xy$	$-4l^2\left[y(lx-1)(lx+1), x(ly-1)(ly+1)\right]$	$4l^4 xy(l^2 r^2 - 6)$
10	(0,2)	$4l^2 y^2 - 1$	$-2\left[l^2 x(2l^2 y^2 - 1), l^2 y(2l^2 y^2 - 5)\right]$	$2l^2(l^2 r^2 - 6)(2l^2 y^2 - 1)$

Table 2.1.2: Table of derivatives of $\phi_k(\vec{r}_i)$ where $l = \sqrt{\alpha\omega}$. The factor $e^{-\frac{1}{2}l^2 r^2}$ is ommitted from all expressions. The formula for $k+1$ is the same as for k if k is even (e.g. $\phi_0 = \phi_1$).

2.1.3 The virial theorem

2.2 The numerical foundation

This is the section explaining the numerical theory upon which the project is built.

2.2.1 Monte Carlo simulations

A Monte Carlo simulation is a way of solving a mathematical or physical problem by generating a random (or pseudorandom [2]) sequence of numbers and evaluating some quantity on the assumption that our the random sequence of numbers is representative of the domain from which the

[2] No electronic random number generator of today is truly random. The sequence of numbers generated will repeat itself after a long period. These periods however, are increadibly long and we will for this report consider the random number generators to be truly random.

quantity is evaluated. An example is evaluating the area of the unit circle by randomly placing points in a $[-1,1] \times [-1,1]$ grid and find the fraction points whose distance to the origin is ≤ 1 and multiply this fraction by the area of the grid (i.e. 4). Such a simple Monte Carlo simulation can give the result as shown in figure 2.2.1.

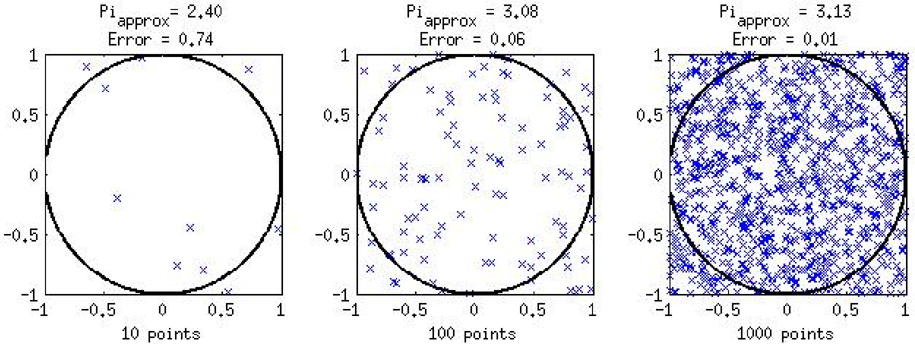

Figure 2.2.1: The results from a very simple Monte Carlo simulation of estimating the circle constant π. The precision increases with the number of points.

However, the method is not confined to this sort of problem, but can be applied to a variety of mathematical and physical problems. In this report, the method, through the Metropolis algorithm (see section 2.2.2) has been applied to a quantum mechanical system.

2.2.2 The Metropolis algorithm

The Metropolis algorithm is a method which cleverly employs a stoichastic approach in order to quickly estimate certain mathematical objects. The method is explained at lengths elsewhere[2], but in this section we will look at an example which captures the main idea of the method.

Suppose we have a PDF[3] $P(x)$ in a domain $[a,b]$ for which we want to calculate the expectation value $\langle g \rangle$ of some function $g(x)$. The integral we need to solve is then

$$\langle g \rangle = \int_a^b P(x)g(x)dx \tag{2.2.1}$$

This integral can be approximated as follows

$$\int_a^b P(x)g(x)dx \approx \frac{b-a}{N} \sum_i P(x_i)g(x_i) \equiv I \tag{2.2.2}$$

Where x_i are some uniformly chosen values in the interval $[a,b]$. Now, imagine instead of picking values x_i uniformly and weighing them by multiplying $g(x)$ with $P(x)$ instead chose the values of $\tilde{x}_i$ from the PDF $P(x)$ and calculated the quantity $\tilde{I}$ given by

$$\tilde{I} = \frac{1}{N} \sum_i g(\tilde{x}_i) \tag{2.2.3}$$

[3] Probability Distribution Function

It can be shown mathematically that for large enough N, these two quantities I and $\tilde{I}$ approach the same value. The problem with such an approach is that we need the precise expression for the PDF $P(x)$ and a robust algorithm for choosing random values from it. With the Metropolis algorithm however, we can use this approach *without* knowing the precise expression of the PDF and the relevant values from the domain come naturally.

The algorithm requires that we are able to calculate $\tilde{P}(x)$, an unnormalized version of $P(x)$ (i.e. some function $aP(x)$ proportional to $P(x)$). This may seem like a very strong requirement, but in many applications, as in this project, this is a much easier task than to calculate the precise PDF. The algorithm goes as follows. Starting with a position x choose a new trial position x_p by

$$x_p = x + \Delta x \tag{2.2.4}$$

Where Δx is a random step according to some rule (see subsections). Then generate a probability criteria $s \in [0, 1]$.

$$\frac{P(x_p)}{P(x)} = \frac{aP(x_p)}{aP(x)} = \frac{\tilde{P}(x_p)}{\tilde{P}(x)} \equiv w \geq s \tag{2.2.5}$$

We accept the trial position as our new x and if not we reject it. If we choose new values of x_i in this manner, the collection of x_i's will in fact reflect the PDF $P(x)$, which was what we needed in order to use equation 2.2.3. Note how equation 2.2.5 doesn't require us to have the exact form of the probability distribution function, only a function $\tilde{P}(x)$ proportional to it.

The intuition behind the algorithm is that for each new position x_i we generate is drawn towards the part of the domain where $P(x)$ is bigger. To see this, we note that if $P(x_p) > P(x)$ then $\frac{P(x_p)}{P(x)} > 1$ which is always bigger than $s \in [0, 1]$ and the new move is always accepted. Whereas if $P(x_p) < P(x)$, the move might be rejected. This allows new values of x_i to be chosen from where $P(x)$ is big, but at the same time allows values with lower values of $P(x)$ to be chosen. Which is what we expect from a PDF. The fact that for a large number M of such steps, the values x_i picked actually reflects the PDF requires some more mathematics, and once again we refer to the lecture notes of the course [2].

2.2.2.1 Brute force Metropolis

If we have no information about the physical nature of the system a reasonable way to model Δx is the following

$$\Delta x = r \Delta x_0 \tag{2.2.6}$$

Where r is a random number between 0 and 1 and Δx_0 is a predefined step length. The step length Δx_0 is affecting the effectiveness of the algorithm in two contradicting ways. A small step length increases the probability of each suggested move x_p being accepted, but weakens the ergodicity[4] of the method.

Since we have no physical understanding of the system when using the brute force Metropolis algorithm, we model our probability criteria s as a uniform random number between 0 and 1.

[4] The way in which the walker is able to reach all positions within a finite number of steps.

Increasing the acceptance probability reduces the amount of times we need to evaluate the probability ratio w, but also increses the amount of Monte Carlo simulations needed in order to get a representative collection of x_i's. It can be argued that a good balance between these two aspects is to achieve an acceptance ratio (i.e. the ratio between accepted and rejected moves) of around 0.5.

This is called the "Brute Force Metropolis algorithm".

2.2.2.2 Importance sampling

If our current position x is in a region where the probability distribution is important, i.e. has a large value, a small step Δx would be favorable. This is because we want to sample many points in this region, which is what a small step allows. In contrast, if the current position x is in an unimportant region, we want a large step Δx since we don't mind moving a bit farther from the region we're in. The brute force approach produced a step independent of the PDF value in each point, which resulted in an optimal acceptance ratio of around 0.5. If we could introduce some sort of rule which adjusts the step Δx according to the value of the PDF in the point we currently are, this could allow us to achieve an acceptance rate of around 0.9 with the same ergodicity.

To make such a rule, we need to use our physical understanding of the system. One way of doing so is to consider the points to move as a random walker would where the resulting probability is equal to the PDF we're treating. Doing this, and invoking the Fokker-Planck and Langevin equations [5] , it can be shown that the choice of Δx is as follows

$$\Delta x = DF(x)\delta t + \eta \tag{2.2.7}$$

Where D is the diffusion term, $F(x)$ is a drift term which is responsible for pulling the particle towards regions where the PDF is important and η is a gaussian random number.

The way the metropolis algorithm relies as we have seen on accepting and rejecting proposed moves in a domain. Where the probability criteria in the brute force method was just a uniform random number between 0 and 1, the physics behind the system now applies to this criteria as well. It affects the criteria in the following way, the probability criteria s is now given by

$$s = \frac{G(x_p, x, \delta t)}{G(x, x_p, \delta t)} s_0 \tag{2.2.8}$$

Where s_0 is a random unifor number between 0 and 1 and $G(a, b, \delta t)$ is the so-called greenfunction given by

$$G(a, b, \delta t) = \frac{1}{(4\pi D\delta t)^{3N/2}} \exp\left(-(a - b - D\delta t F(b))^2 / 4D\delta t\right) \tag{2.2.9}$$

Using this approach is what we will call the "Metropolis algorithm with importance sampling".

2.2.2.3 The metropolis algorithm for our wavefunction

As discussed in section 2.1.1 we need to solve the integral

[5] Once again we refer to the lecture notes [2] for a more detailed explanation.

$$\langle E_L \rangle = \int P(\vec{r}) E_L(\vec{r}) d\vec{r} \tag{2.2.10}$$

Where we have a trial function

$$\Psi_T(\vec{r}_0, ..., \vec{r}_{N-1}, \alpha, \beta) \tag{2.2.11}$$

dependent on 2 trial parameters α and β where $\vec{r}_i = \begin{pmatrix} x_i \\ y_i \end{pmatrix}$. This is exactly the kind of problem the Metropolis algorithm can solve and the explicit algorithm for calculating $\langle E_L \rangle$ and $\langle E_L^2 \rangle$ is given in algorithm 2.1.

```
Data:
An initial position matrix $\mathbf{r} = (\vec{r}_0, \vec{r}_1 ... \vec{r}_{N-1}) = \begin{pmatrix} x_0 & x_1 & ... & x_{N-1} \\ y_0 & y_1 & ... & y_{N-1} \end{pmatrix}$
A method of chosing the step    $\text{Method}(\mathbf{r}) = \Delta\vec{r} = \begin{pmatrix} \Delta x \\ \Delta y \end{pmatrix}$
Result:
The expectation value of the local energy: $\langle E_L \rangle$
The expectation value of the local energy squared: $\langle E_L^2 \rangle$
begin
    $cumulative\_local\_energy = E_L(\mathbf{r}, \alpha, \beta)$    // Initialization
    $cumulative\_local\_energy\_squared = 0$
    $counter = 0$
    while $counter < M$ do
        $i = \text{randint}(0, 1, ..., N-1)$    // Choose random element index
        $\Delta\vec{r} = \text{Method}(\mathbf{r})$    // Create a random two-dimensional step
        $\mathbf{r_p} = \left(\begin{matrix} x_0 & x_1 \\ y_0 & y_1 \end{matrix} \dots \begin{matrix} x_i \\ y_i \end{matrix} + \Delta\vec{r} \dots \begin{matrix} x_{N-1} \\ y_{N-1} \end{matrix}\right)$    // Create a trial position matrix
        $s = \text{prob\_criteria(0,1)}$    // Generate a probability criteria
        $w = |\psi(\alpha, \beta, \mathbf{r_p})|^2 / |\psi(\alpha, \beta, \mathbf{r})|^2$    // Calulate the probability ratio
        if $w \geq s$ then
            $\vec{r} = \vec{r}_p$
            $E_L(\mathbf{r}, \alpha, \beta) = \frac{1}{\Psi_T(\mathbf{r}, \alpha, \beta)} \hat{H} \psi_T(\mathbf{r}, \alpha, \beta)$    // Calculate the local energy
        end
        $cumulative\_local\_energy \overset{+}{=} E_L(\mathbf{r}, \alpha, \beta)$    // Update cumulative_local_energy
        $cumulative\_local\_energy\_squared \overset{+}{=} E_L(\mathbf{r}, \alpha, \beta)^2$
            // Update cumulative_local_energy_squared
        $counter \overset{+}{=} 1$    // Update counter
    end
    Calculate $\langle E_L \rangle = \frac{cumulative\_local\_energy}{M}$
    Calculate $\langle E_L^2 \rangle = \frac{cumulative\_local\_energy\_squared}{M}$
end
```

Algorithm 2.1: The metropolis algorithm used for finding the expecation value of the local energy and the expecation value of the local energy squared. "Method" refers to either the brute force approach or importance sampling.

If we move only one particle at a time, each new trial position will be given by

$$\boxed{\mathbf{r_p} = \begin{pmatrix} x_0 & x_1 & \dots & x_i & + \Delta\vec{r} & \dots & x_{N-1} \\ y_0 & y_1 & & y_i & & & y_{N-1} \end{pmatrix}} \tag{2.2.12}$$

Where $\mathbf{r} = \begin{pmatrix} x_0 & x_1 & \dots & x_{N-1} \\ y_0 & y_1 & \dots & y_{N-1} \end{pmatrix}$. If we are using the brute force approach, then $\Delta\vec{r}$ is simply given by

$$\boxed{\Delta\vec{r} = \Delta r \cdot \vec{\text{rand}}} \tag{2.2.13}$$

Where Δr is a predefined step length and $\vec{\text{rand}}$ is a random 2-vector with elements between -1 and 1.

If we want to implement importance sampling however, we need expressions for the terms in equation 2.2.7. These terms can be shown [1] to be

$$D = \frac{1}{2} \tag{2.2.14}$$

Which stems from the fact that the drift is caused by kinetic energy in front of which is a factor $\frac{1}{2}$ and

$$F = 2\frac{1}{\Psi_T}\nabla_i\Psi_T \tag{2.2.15}$$

The formula for using importance sampling when choosing the trial position $\mathbf{r_p}$ for our wavefunction is thus

$$\boxed{\Delta\vec{r} = \left(\frac{1}{\Psi_T}\nabla_i\Psi_T\right)\delta t + \eta} \tag{2.2.16}$$

We can rewrite this, as in section 2.1.2.4

$$\frac{1}{\Psi_T}\nabla_i\Psi_T = \frac{1}{|S_i|\mathrm{J}}\nabla_i(|S_i|\mathrm{J}) = \frac{1}{|S_i|\mathrm{J}}\left(|S_i|\nabla_i\mathrm{J} + \mathrm{J}\nabla_i|S_i|\right) \tag{2.2.17}$$

$$\boxed{\frac{1}{\Psi_T}\nabla_i\Psi_T = \frac{\nabla_i|S_i|}{|S_i|} + \frac{\nabla_i\mathrm{J}}{\mathrm{J}}} \tag{2.2.18}$$

Expressions we can find both from numerical differenciation and the close form expressions in equations 2.1.33. In this report, we will use both these approaches and compare the CPU time needed.

2.3 Parallelization

Parallelization is a way of exploiting the multiple CPU's you find in modern computers. The idea behind is that tasks that are not dependent on each other and can be done in any order may be divided into so-called threads, and each thread can its own task, see figure 2.3.1.

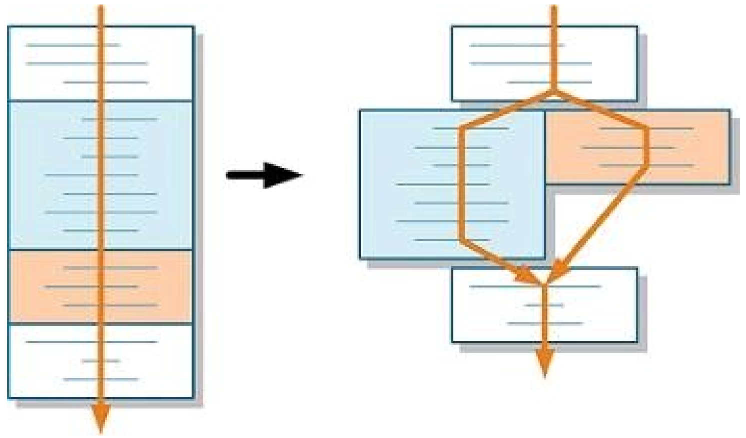

Figure 2.3.1: Flow chart of a parallelization process. The red and blue section are codes which do not depend on each other and can be parallelized. The image is downloaded from http://www.embedded.com/.../parallelization, 22.11.14.

Using a library such as OpenMP (used in this project) makes use of all four CPU's if told to split the work into four different threads. Such an approach is easy to implement to this problem, because we can calculate $\langle E_L \rangle$ for different trial parameters at the same time.

3 Experimental

3.1 Benchmarks and verification

3.1.1 Benchmarks for the brute force approach, no repulsion or jastrow factor

As discussed in section 2.1.2.3, when $\omega = 1$, the trial wavefunction should be able to reproduce the exact solution $E = 2$ (in atomic units) when we disregard the electron repulsion part of the Hamiltonian and don't include the Jastrow factor. This benchmark was tested [6] with the brute force metropolis method by varying α from 0 to 1.5 with steps of 0.05 using numerical diffrentiation of the wavefunction in the expression of the local energy. 10^7 Monte Carlo simulations were performed for each α with a step length Δr suited to each case to get an acceptance rate of around 0.5 (which is implemented in the code before any monte carlo simulation is begun).

Then then the benchmarks for the $N = 6$ and $N = 12$ electron case was tested[7] , still with the brute force approach and 10^7 monte carlo simulations for the $N = 6$ case and 10^6 for the $N = 12$ case, but this time with a smaller interval around $\alpha = 1$, ranging from 0.9 to 1.1 with steps of 0.05. In order to also verify the correct implementation of the oscillator frequency ω, this was set to 1.5, so the energies to reproduce are $10\omega = 15$ a.u. and $28\omega = 42$ a.u.

[6] /Logs/N2_norep_bruteforce_num/test_investigate.cpp, 21.11.14. See appendix, section B.

[7] /Logs/N12_norep_bruteforce_num/test_investigate.cpp and /Logs/N6_norep_bruteforce_num/test_investigate.cpp, 21.11.14.

3.1.2 Benchmark for the brute force approach, with repulsion and jastrow factor

The exact energy of the two electron state *with* repulsion has been shown [4] to be 3ω. To test this result, first a fast investigation of $\langle E_L \rangle$ was performed as function of α and β to find the region in which the lowest energy is. Then, a more detailed search[8] was made with $\alpha \in [0.9, 1.1]$ and $\beta \in [0.35, 0.45]$, both in steps of 0.01 and 10^6 MC simulations at each step. The brute force approach with numerical evaluation of the local energy was used and ω was set to 1. If the exact wavefunction were within our trial parameters, then we would thus expect to get the exact answer $3 \cdot 1 = 3$ a.u., but since this may not be the case we expect the lowest energy to be larger than this, according to section 2.1.1.2.

3.1.3 Comparison of different methods

As described in the theory section, a variety of different methods for solving the VMC problem has been explained. Firstly, there is a choice whether to use brute force (BF) or importance sampling (IS) when picking new trial positions in the metropolis algorithm. Secondly there is the the possibility of using numerical methods (NLE) or the analytical expressions (ALE) when evaluating the local energy. In addition, if we're using importance sampling in the metropolis algorithm, there is a choice to whether or not we should use numerical (NQF) or analytical (AQF) expressions for the quantum force. All these methods should output the same result for the expectation value of the local energy, and to verify this an investigation[9] of $\langle E_L \rangle$ with the different methods were performed for three different, semi-random[10] , combination of problem and trial function parameters; (Number of electrons N, α, Jastrow Factor on (Jn) or off (Jf), β, ω, Electron repulsion on (En) or off (Ef)) with 10^6 monte carlo simulations.

3.2 Optimizations and differences

In this section, the different optimization methods, such as the jastrow factor, analytical expressions and importance sampling, was investigated. The investigation of these issues was based upon twelve test cases with two electrons for which the optimal parameters were found, the procedure is explained in the following section. Since we have verified the validity of the different methods in section 3.1, we could use whatever method we wanted to run sample the energies. For this section, if nothing else is written, the standard method is the brute force approach with analytical expressions for the local energy.

3.2.1 Test cases

The twelve test cases for this section was $\omega = 0.01, 0.1, 0.28, 0.5, 0.75, 1.0$, with repulsion but with and without Jastrow factor. For each test case, the optimal parameters of α and β was found by calculating the energies for α and β in range $[0, 1.2]$ with a resolution of 0.01 with 10^5 MC simulations. Then the parameters yielding the smallest energy was logged[11].

[8] /Logs/N2_rep_jast_num/test_investigate.cpp, 21.11.14.

[9] /Logs/compare_methods/first_example.cpp, second_example.cpp, third_example.cpp, 21.11.14.

[10] Chosen randomly by me, that is.

[11] /Logs/find_parameters2/, 22.11.14

3.2.2 Jastrow factor

When the repulsion part is present, we expect the jastrow factor to improve the correlation (interaction) between the electrons, giving us a better (i.e. lower) estimate of the energy. We can measure the correlation by seeing how the lowest energy estimate changes with the jastrow factor relative to without the Jastrow factor.

3.2.3 Importance sampling

Using importance sampling introduces the depence on δt when evaluating $\langle E_L \rangle$, as discussed in section 2.2.2.2. The dependence of the importance sampling method on δt was investigated[12] for $10^3 - 10^6$ monte carlo simulations and $\delta_t \in [10^{-6}, 10^2]$. Trial wave function parameters were chosen to be $\alpha = 0.72$ and $\beta = 0.24$[13]. For every time step δt, a brute force evaluation of the energy was also performed to compare the different methods.

3.2.4 Timely differences between methods

3.3 Applications

3.3.1 Properties of the approximated wavefunctions

3.3.2 The virial theorem

4 Results and discussion

4.1 Benchmarks and verification

4.1.1 Benchmarks for the brute force approach, no repulsion or jastrow factor

Table 4.1.1 shows the results from the brute force simulation of the two electron - no repulsion case. A plot of the results is shown in figure 4.1.1.

The figure shows exactly what we would expect from the discussion of section 2.1.2.3. The energy is always larger than 2 a.u. and takes this value only when $\alpha = 1$. We also see a huge drop in the variance just as we reach $\alpha = 1$ which indicates that this is indeed an eigenstate of the system. What little is rest of the variance at $\alpha = 1$ can be due to numerical errors in the calculation of the laplacians. This has in retropspect been verified to be true by using the analytical expression for the local energy.

Table 4.1.2 shows the result from the $N = 6$ and $N = 12$ electrons case with no repulsion using the brute force approach with numerical evaluation of the local energy.

The table shows that we are able to produce the results we anticipated. What little there is of variance at $\alpha = 1$ is probably due the numerical evaluation of the local energy, which has been confirmed in retrospect by using the analytical expression for the local energy. It also shows that the code has implemented the oscillator frequency ω correctly.

[12] Logs/importance_sampling/, 24.11.14

[13] Which is not close to an eigenstate of the system for which we found optimal parameters in section 4.1.2. This is important because if the parameters are such that we are close to an eigenstate, E_L will not vary much from point to point, which reduces the importance of good sampling.

α	0.0	0.05	0.1	0.15	0.2	0.25	0.3	0.35	0.40	0.45
E(a.u)	1.1e5	19.96	10.03	6.77	5.17	4.23	3.61	3.19	2.89	2.66
Variance	7.4e9	2.0e2	4.8e1	2.0e1	1.1e1	7.0e0	4.5e0	3.1e0	2.2e0	1.5e0
α	0.5	0.55	0.6	0.65	0.7	0.75	0.8	0.85	0.9	0.95
E(a.u.)	2.49	2.36	2.26	2.18	2.12	2.08	2.047	2.024	2.0096	2.002
Variance	1.1e0	7.9e-1	5.6e-1	3.9e-1	2.6e-1	1.7e-1	1.0e-1	5.3e-2	2.6e-2	5.2e-3
α	1.0	1.05	1.1	1.15	1.2	1.25	1.3	1.35	1.4	1.45
E(a.u)	2	2.0029	2.010	2.021	2.035	2.05	2.07	2.09	2.12	2.14
Variance	3.3e-13	4.7e-3	1.8e-2	3.9e-2	6.7e-2	1.0e-1	1.4e-1	1.8e-1	2.3e-1	2.9e-1

Table 4.1.1: Table showing the results from the brute force VMC simulation of the expectation value of the local energy $\langle E_L \rangle$ for α between 0 and 1.45 in the 2-electron case with no repulsion or Jastrow factor. The results show a minimum for the energy at $\alpha = 1$, as expected, and the variance at this point is so small that we expect it to be an eigenstate of the system.

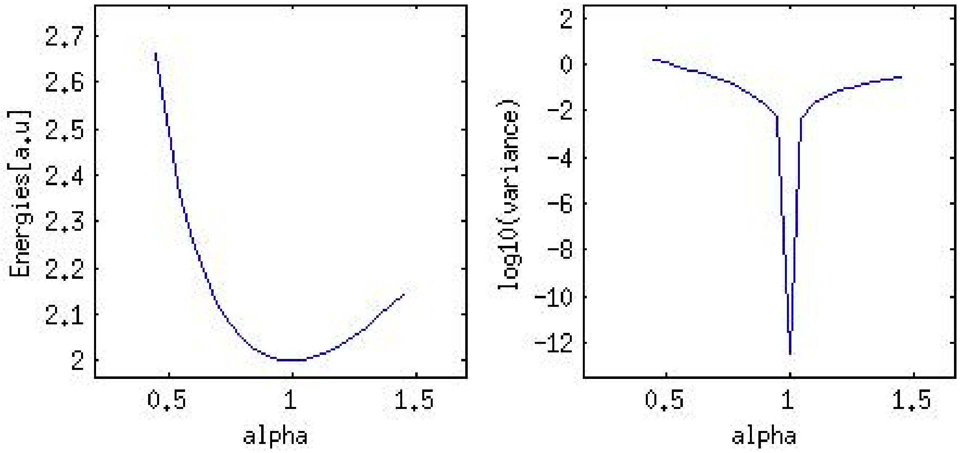

Figure 4.1.1: Plot of the values given in table 4.1.1, the brute force VMC simulation of the two-electron no repulsion or Jastrow factor case. Some of the first values of α has been omitted to make the graph more informative.

4.1.2 Benchmark for the brute force approach, with repulsion and jastrow factor

A plot of the energies and log_{10} of the variances after the investigation is shown in figure 4.1.2.

It is not easy to see from the figure what is the lowest energy, but manipulating the figure in matlab revealed the smallest test state energy, which is given in table 4.1.3.

The lowest energy was also found in the region where the variance was lowest. It seems thus that the value obtained in this VMC calculation was very close to the real lowest eigenstate. Very close, however, does not mean that we have found *the* eigenstate. As we saw in the previous section, when we hit the real eigenstate, the variance dropped rapidly to sizes in order of magnitude $10^{-12} - 10^{-13}$ which was not the case here.

	α	0.9	0.95	1	0.105	0.11
N=6	E(a.u)	15.0803	15.0850	15	15.0184	15.0708
	Variance	2.49e-1	5.89e-2	2.56e-13	2.36e-2	2.05e-1
	α	0.9	0.95	1	0.105	0.11
N=12	E(a.u)	42.2114	42.0497	42	42.0563	42.1937
	Variance	6.92e-1	1.66e-1	1.72e-10	1.48e-1	5.69e-1

Table 4.1.2: The results from calculating the expectation value of the local energy and its variance. We see that the code works for the $N = 6$ and $N = 12$ case because we are producing the expected results, $E_6 = 15$ and $E_{12} = 42$ ($\omega = 1.5$). The variance at $\alpha = 1$ is so small that we have probably the exact wavefunction.

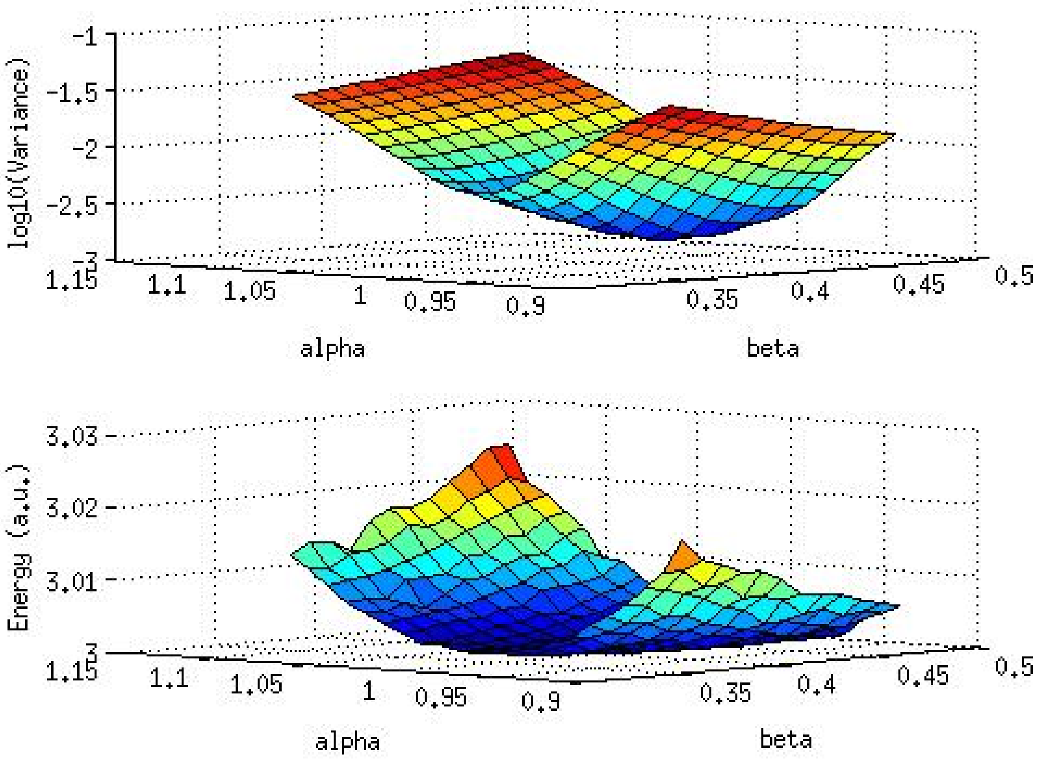

Figure 4.1.2: Plot of the expectations of the local energy and log_{10} of variances after the VMC simulations with two electrons, with repulsion and jastrow factor. Each point was based on 10^6 simulations. The figure gives an overview of how the energies vary as function of α and β. The lowest energy $E = 3.00022$ where $\alpha = 0.98$ and $\beta = 0.42$.

4.1.3 Comparison of different methods

The results from calculation $\langle E\rangle$ for the different method combinations and problem/wave function parameters are shown in table 4.1.4.

The lowest energy $E = 3.00022$ and Variance $= 0.001568$ at $\alpha = 0.98$ and $\beta = 0.42$.

Table 4.1.3: The lowest energy found from the VMC study of the two electron case with repulsion and Jastrow factor.

	Problem and Wavefunction parameters		
Method parameters	(2, 1, Jf, 0, 1, Ef)	(12, 0.5, Jf, 0, 1.5, En)	(6, 0.82, Jo, 0.22, 3, En)
(BF, NLE)	(2.000, 6e-13)	(92.10, 67.54)	(54.74, 28.11)
(BF, ALE)	(2.000, 0)	(92.18, 161.4)	(54.72, 28.19)
(IS, NLE,NQF)	(2.000, 6e-14)	(92.15, 66.43)	(54.72, 27.33)
(IS, NLE, AQF)	(2.000, 1e-13)	(92.06, 54.49)	(54.72, 27.32)
(IS, ALE, NQF)	(2.000, 0)	(92.13, 143.2)	(54.76, 27.35)
(IS, ALE, AQF)	(2.000, 0)	(92.12, 76.24)	(54.74, 27.44)

Table 4.1.4: A table of the expectation value and variance (Energy, Variance) of the local energy obatined with different Problem/Wavefunction parameters (N,α, Jn/Jf, β, ω, En/Ef). For explanation of abbreviations, see section 3.1.3. For all trials, 10^6 VMC calculations were performed and for the importance sampling methods, a timestep of $\delta t = 0.1$ was used.

These are very strong results. Firstly, we have verified the nummerical brute force method against known benchmarks and seen that they are correct, thus, it seems that the other methods are working properly! Secondly, the different methods used very different ways to obtain the results. That the results are almost the same is a strong indication that all the methods are doing the same thing and working correctly.

4.2 Optimizations and differences

4.2.1 Test cases

The results from finding the optimal parameters for α and β for the different test cases are given in table 4.2.1.

The result for $\omega = 1$ with jastrow factor is almost in perfect agreement with what was found in section 4.1.2, even though the method of finding the optimal parameters was different. Since only 10^5 MC simulations were used here, I think this high correspondance is a lucky coincidence, but it does show that the results obtained are reliable, and for the purposes of this section, they are probably good enough.

Another approach to get these parameters would be to do a simular approach as in section 4.1.2 for each case. This proved to be too time-consuming, and an automated approach such as the one used here could be run overnight, which was done here. For a further discussion on code efficiency and time constraints, see section 4.2.5.

4.2.2 Jastrow factor

The relative change in energy (i.e. correlation) due to the Jastrow factor for the test cases is given in table 4.2.2.

Test cases: Optimal parameters	$\omega = 0.01$ No Jast.	 Jast.	$\omega = 0.10$ No Jast.	 Jast	$\omega = 0.28$ No Jast.	 Jast.
α	0.30	0.91	0.56	0.93	0.60	0.96
β	-	0.07	-	0.18	-	0.26
Energy (a.u.)	1.03e-1	7.40e-2	5.25e-1	4.41e-1	1.14	1.02
Variance	1.12e-2	1.48e-5	1.53e-1	2.76e-4	9.43e-1	7.05e-4
Test cases: Optimal parameters	$\omega = 0.50$ No Jast.	 Jast.	$\omega = 0.75$ No Jast.	 Jast.	$\omega = 1$ No Jast.	 Jast.
α	0.70	0.97	0.74	0.98	0.72	0.97
β	-	0.32	-	0.38	-	0.42
Energy (a.u.)	1.80	1.66	2.50	2.34	3.15	3.00
Variance	1.25	1.01e-3	1.69	1.35e-3	2.03	2.7e-3

Table 4.2.1: The optimal parameters with and without Jastrow factor (Jast.). Results were produced by calculating the energy for every α and β in the interval $[0, 1.2]$ with a step length of 0.01 and 10^5 MC simulation. The estimates with the Jastrow factor seem to be better than those without, though the importance seem to decrease with increasing oscillator frequency ω (see section 4.2.2).

ω	0.01	0.10	0.28	0.50	0.75	1
Correlation $C = \frac{E_{NJ} - E_J}{E_{NJ}}$	0.282	0.160	0.105	0.078	0.064	0.048

Table 4.2.2: The correlation introduced by the Jastrow factor for the optimal parameters α and β as a function of ω. We see that the correlation factor is less important for high oscillator potentials. "NJ" means no Jastrow-factor and "J" means with Jastrow factor.

A plot of the values in table 4.2.2 is given in figure 4.2.1.

Our first observation is that the Jastrow factor does indeed better our estimation of the ground state energy. We already knew from section 4.1.2 that the including the jastrow factor gave us a very good estimate of the ground state energy when $\omega = 1$, and from table 4.2.1 we see that the ratio between variance and energy estimate is almost constant ($\sim 5 \times 10^{-4}$) in the case with the Jastrow factor as opposed to the case without the Jastrow factor.

Our second observation from both the table and the figure is that the correlations get smaller as the oscillator frequency increases.

From the left hand plot in the figure, one might suspect that the correlation decreases exponentially as a function of ω. To test this, a plot of $\ln(C)$ against ω has been included in the right hand plot. If this hypothesis was true, the result should be a straight line, but alas, it is not. It is therefore probably some other rule governing the relation between C and ω.

4.2.3 Importance sampling

The results from evaluation $\langle E_L \rangle$ as a function of Monte Carlo simulations and time steps δt are given in table 4.2.3 and figure 4.2.2.

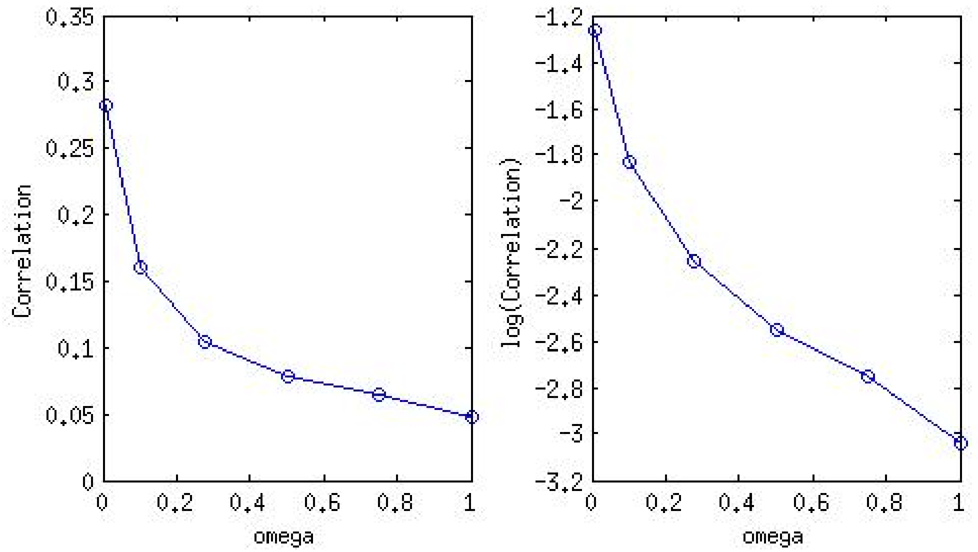

Figure 4.2.1: Plot of the correlation introduced by the Jastrow factor for the optimal parameters α and β as a function of ω. From the left figure one may suspect a curve of the shape $C = C_0 \exp(-k\omega)$ but the right hand plot of $\ln(C)$ against ω does not look very linear which weakens this hypothesis.

Energies obtained with the brute force method									
MCS \ $\delta t =$	10^{-6}	10^{-5}	10^{-4}	10^{-3}	10^{-2}	10^{-1}	1	10	10^{2}
10^3	3.26	3.21	3.16	3.24	3.35	3.29	3.36	3.11	3.25
10^4	3.21	3.22	3.22	3.27	3.25	3.26	3.24	3.22	3.26
10^5	3.23	3.24	3.21	3.22	3.24	3.23	3.24	3.24	3.23
10^6	3.23	3.23	3.23	3.23	3.24	3.24	3.24	3.24	3.24
Energies obtained with importance sampling									
MCS \ $\delta t =$	10^{-6}	10^{-5}	10^{-4}	10^{-3}	10^{-2}	10^{-1}	1	10	10^{2}
10^3	7.78	2.13	4.49	4.01	3.25	3.21	3.28	3.23	2.86
10^4	3.95	2.74	2.42	3.19	3.33	3.29	3.21	3.18	4.02
10^5	2.46	3.58	3.22	3.26	3.21	3.23	3.20	3.07	5.40
10^6	3.99	3.41	3.13	3.22	3.24	3.22	3.20	4.60	2.60

Table 4.2.3: Energies obtained with the brute force method and the importance sampling method as function of number of Monte Carlo simulations (MCS) and time step δt. Brute force energy estimates do naturally not vary with δt since the latter do not play a role in the brute force method. The different energies obtained with the brute force method are thus only repetitions of the same simulation. Importance sampling seem to be stable around $\delta t = 10^{-2}$.

We see from the results that the reliability of using importance sampling depends greatly on the time step chosen. The results seem to coincide with the brute force method when $\delta t \approx 10^-2 - 10^-2$. This is natural because if δt is too small, the steps in the metropolis algorithm does also become very small. It then takes a large number of Monte Carlo simulations for the

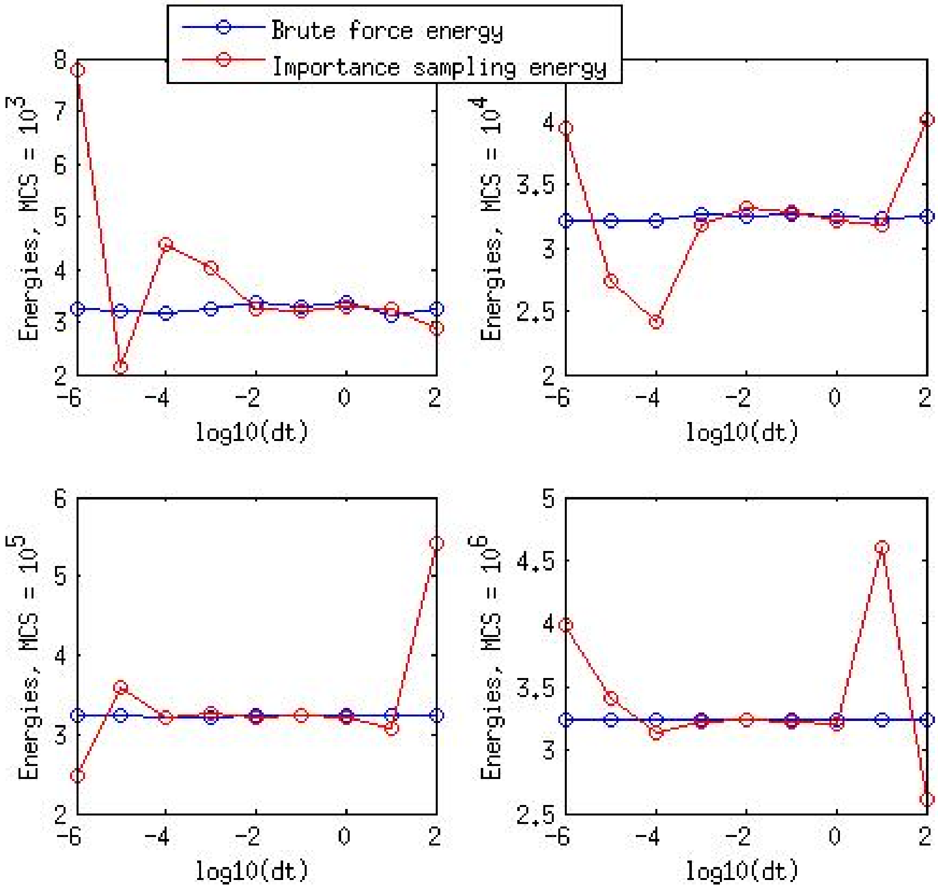

Figure 4.2.2: Plot of the energies evaluated by the brute force method and the metropolis method as a function of number of Monte Carlo simulations (MCS) and time step δt.

metropolis walker to move across a representative selection of points. This is illustrated by the figure by noticing that as the number of monte carlo simulations gets larger, a greater range of δt's converges to the proper solution. When δt gets too large, the step length gets too large and we have little control over how the energy estimates will behave.

The energy estimates by the brute force method does not vary with δt. This is natural because δt plays no role in the evaluation of $\langle E_L \rangle$ with the brute force method. The different energies obtained for different δt's are thus just repetitions of the same measurement, and serve thus only as a illustration of the uncertainty in the evaluation of $\langle E_L \rangle$. For the purposes of this project, it seems that the brute force method serves as a stable and reliable method for calculating $\langle E_L \rangle$ with good precision only with a 10^3 Monte Carlo simulations. For more complex problems, it may be that importance sampling, with its foundation in the physical system, attributes to greater precision in producing results, but for the problems discussed here, it seems that it only introduces a new uncertainty in whether the time step δt has been chosen correctly.

4.2.4 Timely differences between methods

4.2.5 Discussion: code efficiency and time constraints

The code upon which this project is built has a lot of potential for improvement. Lots of the methods developed in this project has been programmed to the point at which the worked, and time constraints have cut short further improvements with regards to code efficacy. Examples of things that could have improved general code efficacy are things as

- **Reducing the amount of times a variable is calculated**: Although this aspect have been in the back of my mind while programming, I'm sure that a thorough review of the code would reveal a lot of potential.
- **Avoiding too many divisions**: Divisions in a computer is normally more slow than multiplication[14]. Replacing recurrent divisions by clever multiplication might have improved code efficiency.
- **Reduce usage of complicated functions such as** sin**,** exp **etc**: Although I have put some thought into this while designing the code, there are probably some places where a lot of CPU time could be saved.

There are also things in this specific project that could have improved code efficiency. Examples are

- **More efficient ratio calcuations**: In the Metropolis algorithm used in this project, a ratio of the wavefunctions squared was evaluated. But since we only move *one* particle at a time, a lot of the so-called *co-factors* in the Slater determinant cancel[15] and reduces the number of expressions we need to calculate.
- **Reusage of some expressions**: In the calculation of the analytical local energy, the terms needed for the analytical quantum force was also calculated. Implementing this into the code so that if the analytical expression for the local energy was evaluated *and* importance sampling was used with analytical quantum force, the expressions would reused and not calulated twice. There might also be other such examples of expressions which are evaluated multiple times in this code.
- **Better usage of OpenMP parallelization**: An interesting question is how much time it takes for OpenMP to communicate between the different threads. If this would prove itself a significant issue, then a better implementation of parallelization could have saved some time.

Finally there is the matter of method for finding the optimal parameters α and β. In this project, I used a very brute force method for which I could only spare 10^5 MC simulations for each energy in order for the process not to take days. A better algorithm for finding the optimal α and β was not in the scope of this project, but I believe it could seriously improve the efficacy of the process.

But all things considered, I am happy to have paid the price of a slow running code in exchange for having been able to implement lots of different methods for solving the problems discussed

[14] Source: http://streamcomputing.eu/

[15] Page 81, [3].

in this project. Even though the weak efficiency is probably affecting the precision of the result in the next section in a negative manner, the results are still correct within a certain margin of error. Also, the code is not going anywhere, so if I later want to improve upon it and implement better algorithms for finding α and β I am free to do so, even I don't have time to do so before the deadline of this project.

4.3 Applications

4.3.1 Properties of the approximated wavefunctions

4.3.2 The virial theorem

5 Conclusion

References

[1] Morten Hjorth-Jensen. Additional slides for monte carlo project. http://www.uio.no/.../montecarloaddition.pdf, 2014.

[2] Morten Hjorth-Jensen. *Computational Physics - Lecture Notes Fall 2014*. August 2014.

[3] Jørgen Høgberget. Quantum monte-carlo studies of generalized many-body systems, June 2013.

[4] M. Taut. Two electrons in an external oscillator potential: Particular analytic solutions of a coulomb correlation problem. *Phys. Rev. A*, 48:3561–3566, Nov 1993.

A Reference to the questions posed in the project instructions

Since the format of this report does not correspond to the structure of the project instructions, a reference list over the posed questions and where to find the answer is given below.

Section	Instruction	Reference
a)	Convince yourself that ... is simply 2ω.	Section 2.1.2.3
	What is the total spin ...? Find arguments for why...	Section 2.1.2.3
b)	Perform a Variational Monte Carlo calculation...	Section 4.1.2
c)		
d)	How important are ... the Jastrow factor?	Section 4.2.2
e)	Compute... for N = 6 electrons ... and $\omega = 1.0$.	Section 4.3.1
	Reproduce the unperturbed ... is switched off.	Section 4.1.1
	Convince yourself ... for N = 6 is 10ω	Section 2.1.2.3
	What is the expected total spin of the ground states?	Section 2.1.2.3
f)		
g)	Find closed-form expressions ...	Section 2.1.2.4
	compare the results ... for both N = 2 and N = 6	Section 4.2.4

Table A.0.1: Reference list over the posed questions and where to find the answer.

B Codes

All codes used in this exercise can be found at GitHub:
https://github.com/vidarsko/Project3.

The class structure of the code is given in the bullet points below

- TrialWavefunction
- QuantumDots
- Investigate

Througout this report, reference to different codes are made as footnotes together with the date at which they were run.