FYS4150 - Project 2

Simen Nyhus Bastnes & Eirik Ramsli Hauge

3. October 2016

Abstract

The aim of this project is to solve Schrödinger's equation for two electrons with and without a repulsive Coulomb interaction in a three-dimensional harmonic oscillator well. We start by reformulating the equation in a discretized form as an eigenvalue equation, which can be solved with Jacobi's method. The program will include unit tests and time measurement comparing Jacobi's method to another method. We found that Jacobi's method became more accurate if we started out with a bigger matrix, but it also became very slow with increasing n. The probability of finding the electrons over a bigger ρ interval decreases as ω_r increases, but if we add the Coulomb interaction the interval will be larger than for its non-Coulomb counterpart.

1 Introduction

We started by first studying one then two electrons in a harmonic oscillator well and by solving the Schrödinger's equation by reformulating it in a discretized form as an eigenvalue equation and solving that equation with Jacobi's method.
Once the Schrödinger's equation was on a form we can solve with Jacobi's method, we will solve it for nxn matrices of different size for one electron and compare our results. Then we will see if the number of rotations needed to complete Jacobi's method can be apporximated by a polynomal and compare the time usage of our "home made" Jacobi's method with the time usage of Armadillos solver for eigenvalues.

Afterwards we will increase the number of electrons to two. We change our harmonic oscillator potential to make it dependent on ω_r, reflecting the strength of the potential. The Coulomb interaction will also be implemented in this part of the program, so that we can compare the probability $|\Psi(\rho)|^2$ for different ω_r with and without Coulomb interaction.

The program will also include three unit tests checking if we get the right eigenvalues, find the largest non-diagonal element and if orthogonality is preserved during Jacobi's method.

2 Theory

2.1 Schrödinger's equation

In this project, we will assume that the electrons move in a three-dimensional harmonic oscillator potential, and repel each other via the static Coulomb interaction. By assuming spherical symmetry, the solution for the radial part of Schrödinger's equation for one electron reads

$$-\frac{\hbar^2}{2m}\left(\frac{1}{r^2}\frac{d}{dr}r^2\frac{d}{dr}-\frac{l(l+1)}{r^2}\right)R(r)+V(r)R(r)=ER(r) \tag{1}$$

For the rest of the project, we set the orbital momentum quantum number l to zero. In our non-interacting case, the harmonic oscillator potential is $V(r)=(1/2)kr^2$ with $k=m\omega^2$.

We perform a substitution for $R(r)=(1/r)u(r)$, introduce a dimensionless variable $\rho=(1/\alpha)r$, where α is a constant with dimension length $\alpha=(\hbar^2/mk)^{1/4}$. Inserting this into the Scrödinger equation (1), we get

$$-\frac{d^2}{d\rho^2}u(\rho)+\rho^2u(\rho)=\lambda u(\rho) \tag{2}$$

where $\lambda=(2m\alpha^2/\hbar^2)E$. Equation (2) is the first equation we want to solve numerically, and we will later use that for $l=0$, the first eigenvalues λ are $\lambda_0=3$, $\lambda_1=7$ and $\lambda_2=11$.

Starting our journey to write equation (2) as an eigenvalue equation, we use the by now standard expression for the second derivative

$$u''=\frac{u(\rho+h)-2u(\rho)+u(\rho-h)}{h^2}+\mathcal{O}(h^2)$$

where h is our step length. We define minimum and maximum values for ρ, $\rho_{\min}=0$ and $\rho_{\max}$, as we cannot set $\rho_{\max}=\infty$. When setting $\rho_{\max}$, we need to make sure that it is set high enough so that the square of the wavefunction u is small near $\rho_{\max}$ for accuracy of the solution. Next, we split this interval into N number of mesh points, so that we can define the step length as

$$h=\frac{\rho_N-\rho_0}{N}$$

where $\rho_0=\rho_{\min}$ and $\rho_{\max}=\rho_N$. This gives us an expression for ρ at point i

$$\rho_i=\rho_0+ih \qquad i=1,2,\dots,N$$

Now we can rewrite the second derivative as

$$u_i''=\frac{u_{i+1}-2u_i+u_{i-1}}{h^2}$$

Using this, we can rewrite the dimensionless Schrödinger equation (2) as a discretized equation.

$$-\frac{u_{i+1}-2u_i+u_{i-1}}{h^2}+\rho_i^2u_i=-\frac{u_{i+1}-2u_i+u_{i-1}}{h^2}+V_iu_i=\lambda u_i \tag{3}$$

where $V_i=\rho_i^2$ is the harmonic oscillator potential. From the previous project, we recall that we can represent equation (3) in terms of a tridiagonal matrix, where the diagonal elements are given by

$$d_i=\frac{2}{h^2}+V_i=\frac{2}{h^2}+\rho_i^2$$

and the non-diagonal matrix elements

$$e_i = -\frac{1}{h^2}$$

We notice that in this case, the non-diagonal matrix elements are constant. Using this, we can now write the Schrödinger equation as

$$d_i u_i + e_{i-1}u_{i-1} + e_{i+1}u_{i+1} = \lambda u_i$$

So that we now have the following eigenvalue equation

$$\begin{pmatrix} \frac{2}{h^2}+V_1 & -\frac{1}{h^2} & 0 & \dots & 0 \\ -\frac{1}{h^2} & \frac{2}{h^2}+V_2 & -\frac{1}{h^2} & 0 & \dots \\ \dots & \dots & \dots & \dots & 0 \\ \dots & \dots & -\frac{1}{h^2} & \frac{2}{h^2}+V_{N-2} & -\frac{1}{h^2} \\ 0 & \dots & 0 & -\frac{1}{h^2} & \frac{2}{h^2}+V_{N-1} \end{pmatrix} \begin{pmatrix} u_1 \\ u_2 \\ \dots \\ u_{N-2} \\ u_{N-1} \end{pmatrix} = \lambda \begin{pmatrix} u_1 \\ u_2 \\ \dots \\ u_{N-2} \\ u_{N-1} \end{pmatrix} \tag{4}$$

which is what we want to solve by using Jacobi's method.

Interacting case

The eigenvalue equation (4) was derived by assuming one electron in a harmonic oscillator potential. For two electrons with no repulsive Coulomb interaction, we have the following Schrödinger equation

$$\left(-\frac{\hbar^2}{2m}\frac{d^2}{dr_1^2} - \frac{\hbar^2}{2m}\frac{d^2}{dr_2^2} + \frac{1}{2}kr_1^2 + \frac{1}{2}kr_2^2\right)u(r_1, r_2) = E^{(2)}u(r_1, r_2)$$

With no interaction, this can be written out as the product of two single-electron wave functions, like we did earlier. Introducing relative coordinates $\mathbf{r} = \mathbf{r_1} - \mathbf{r_2}$ and center-of-mass coordinate $\mathbf{R} = 1/2(\mathbf{r_1} + \mathbf{r_2})$, and adding the repulsive Coulomb interaction between the two electrons to our potential

$$V(r_1, r_2) = \frac{\beta e^2}{|\mathbf{r_1} - \mathbf{r_2}|} = \frac{\beta e^2}{r}$$

So that we get the following radial Schrödinger equation (omitting the center-of-mass motion)

$$\left(-\frac{\hbar^2}{m}\frac{d^2}{dr^2} + \frac{1}{4}kr^2 + \frac{\beta e^2}{r}\right)\psi(r) = E_r\psi(r)$$

This is quite similar to the equation (1) we found earlier, and we can likewise introduce a dimensionless variable $\rho = r/\alpha$, and rewrite Schrödinger's equation as

$$-\frac{d^2}{d\rho^2}\psi(\rho) + \omega_r^2\rho^2\psi(\rho) + \frac{1}{\rho}\psi(\rho) = \lambda\psi(\rho)$$

So we see that the only thing we have to modify when adding the repulsive coloumb interaction is to change the potential from ρ^2 to $\omega_r\rho^2 + 1/\rho$, where we treat ω_r as a parameter reflecting the strength of the harmonic oscillator potential. We will later look at how the wave function behaves for different values of ω_r.

2.2 Jacobi's method

The aim of Jacobi's method is to use similarity transformation to make a $n \times n$ matrix into a diagonal $n \times n$ matrix with its eigenvalues as the diagonal. We do this by starting with a matrix $\mathbf{A}$ which is diagonally dominant [1]. Within this matrix we find the largest element that is not on the diagonal, calling its indexes k and l. The algorithm goes as follows:

Algorithm 1 Maximum non-diagonal element

```
1: $max = 0.0$
2: for $i = 0, n$ do
3:     for $j = i + 1, n$ do
4:         if $|A[i][j]| > max$ then
5:             $|A[i][j]| = max$
6:             k = i
7:             l = j
8:         end if
9:     end for
10: end for
```

Now we want to rotate matrix $\mathbf{A}$ such that the larges off-diagonal element becomes zero. This is done by using equation:

$$\mathbf{B} = \mathbf{S^T A S} \tag{5}$$

Where $\mathbf{S}$ is given as:

$$\begin{pmatrix}
1 & 0 & \cdots & 0 & 0 & \cdots & 0 & 0 \\
0 & 1 & \cdots & 0 & 0 & \cdots & 0 & 0 \\
\vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \vdots & \vdots \\
0 & 0 & \cdots & \cos\theta & 0 & \cdots & 0 & \sin\theta \\
0 & 0 & \cdots & 0 & 1 & \cdots & 0 & 0 \\
\vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\
0 & 0 & \cdots & 0 & 0 & \cdots & 1 & 0 \\
0 & 0 & \cdots & -\sin\theta & 0 & \cdots & 0 & \cos\theta
\end{pmatrix}$$

With properties $\mathbf{S^T} = \mathbf{S}^{-1}$. The positioning of $\cos\theta$ and $\sin\theta$ within the matrix is defined by the index of the largest off-digaonal element like this:

$$s_{kk} = s_{ll} = \cos\theta, \quad s_{kl} = -s_{lk} = \sin\theta, \quad s_{ii} = -s_{i i} = 1 \quad i \neq k,\, i \neq l$$

This is the same as performing a plane rotation on A around an angle θ in the Euclidean n-dimensional space.

For **B** this gives us:

$$\begin{aligned}
b_{ii} &= a_{ii},\ i \neq k,\ i \neq l \\
b_{ik} &= a_{ik}\cos\theta - a_{il}\sin\theta,\ i \neq k,\ i \neq l \\
b_{il} &= a_{il}\cos\theta + a_{ik}\sin\theta,\ i \neq k,\ i \neq l \\
b_{kk} &= a_{kk}\cos^2\theta - 2a_{kl}\cos\theta\sin\theta + a_{ll}\sin^2\theta \\
b_{ll} &= a_{ll}\cos^2\theta + 2a_{kl}\cos\theta\sin\theta + a_{kk}\sin^2\theta \\
b_{kl} &= (a_{kk} - a_{ll})\cos\theta\sin\theta + a_{kl}(\cos^2\theta - \sin^2\theta)
\end{aligned}$$

Where θ is set as the angle that makes $b_{kl} = 0$. Each transformation can be proven to bring the matrix closer to the diagonal form. [3]

When all this is done, the old maxmium off-diagonal value will have become zero and a new one will have risen, but it will not be as large as the old one. We repeat our whole process until the maxmium value of all off-diagonal elements are below a certain threshold value. Ideally we would want them all to be zero, but very, very small values is good enough.
By looking at our expression for $b_kl = 0$ and defining

$$\begin{aligned}
t &= \tan\theta = \frac{\sin\theta}{\cos\theta} \\
\tau &= \cot 2\theta = \frac{a_{ll} - a_{kk}}{2a_{kl}}
\end{aligned}$$

We get:

$$\begin{aligned}
(a_{kk} - a_{ll})\cos\theta\sin\theta + a_{kl}(\cos^2\theta - \sin^2\theta) &= 0 \\
(a_{kk} - a_{ll})\cos^2\theta t + a_{kl}\cos^2\theta - a_{kl}\sin^2\theta &= 0 \\
(a_{kk} - a_{ll})t + a_{kl} - a_{kl}\frac{\sin^2\theta}{\cos^2\theta} &= 0 \\
(a_{kk} - a_{ll})t + a_{kl}(1 - t^2) &= 0 \\
\frac{a_{kk} - a_{ll}}{a_{kl}}t &= t^2 - 1 \\
-\tau t &= \frac{t^2 - 1}{2} \\
t^2 + 2\tau t - 1 = 0, \quad \Rightarrow \quad t &= -\tau \pm \sqrt{1 + \tau^2}
\end{aligned}$$

Cosine and sinus are easily found to be

$$\cos\theta = \frac{1}{\sqrt{1 + t^2}}, \quad \sin\theta = \cos\theta \cdot t$$

As we can see we can choose two different values for t. The one we want is the one that is the smallest. This is evident if we look at b_{ik} and b_{il}. Hopefully these values will quickly reach values near zero and we want to keep them there. To do this we want $\cos\theta$ to go to 1 and $\sin\theta$ to go to 0 simultaniously. By always choosing the lowest t-value we choose the lowest possible angle and therefore we will minimize the changes to the non-diagonal elements in the matrix. Except for b_{kl} which is always set to zero.

To fully grasp the idea of this method we have imagined it as this:
Let us pretend that you have ten bars in front of you to affect one big bar placed in the middle of the ten. These ten bars can be pushed down, but as you push down one, the others will be affected also. We want to push all the bars (except the big one in the middle) as far down as possible, and if you push down the tallest bar, the others can't reach its height, but they are free to go up and down. Then all you have to do is push down one bar (find the maximum non-diagonal element and rotate the matrix) see how the other bars react (view the matrix after rotation), locate the bar that is now the highest and push that one down. As you do this, gradually all the bars will come under a certain height, even though they may not be completely pushed down. When you are satisfied with the shortness of the tallest bar, your job is done and we have found our diagonal matrix with eigenvalues.

2.3 Unit tests

We implemented three unit tests. The first checks if Jacobi's method returns the right eigenvalues for a known matrix with known eigenvalues. The second test checks if our function for finding the largest non-diagonal element does so for a known symmetric matrix And the last one checks if the eigenvectors of our rotated matrix are still orthogonal. For the last check we use the fact that for a vector $\mathbf{v}$ that has orthogonal properties:

$$\mathbf{v}_j^T \mathbf{v}_i = \delta_{ij}$$

The vector $\mathbf{w_i} = \mathbf{U}\mathbf{v}_i$ will also be orthogonal after a unitary or orthogonal transformation. This can be showed by the following:

$$\mathbf{w}_j^T \mathbf{w}_i = \mathbf{v}_j^T \mathbf{U}^T \mathbf{U} \mathbf{v}_i = \mathbf{v}_j^T \mathbf{v}_i = \delta_{ij}$$

This was used by implementing the third method as a part of each step in Jacobi's method when called upon.

3 Experimental

The programs used in this project can be found in the GitHub repository [2], in the `/src/` folder. When running the program, it takes 3 command line arguments, N, $\rho_{\max}$ and `mode`. The program has four different runmodes, which we will discuss in some more detail.

3.1 One electron, non-interacting case

Setting `mode` to `one` runs the program for one electron and no Coulomb interacted, with specified dimension N and maximum value $\rho_{\max}$, and stores the 5 lowest eigenvalues (calculated both by Jacobi's method and Armadillo's `eig_sym()` function) in a file `eigenvalues_noint_n<N>.dat`. The execution time for both methods are appended to the file `timing.txt`.

3.2 Two electrons, interacting case

Setting `mode` to `two-int` runs the program for 4 different values of $\omega_r = (0.01, 0.5, 1.0, 5.0)$ with the Coulomb interaction added.
The eigenvector for the ground state is written to the file `eigenvectors_two-int_n<N>.dat`

3.3 Two electrons, non-interacting case

Setting `mode` to `two-noint` runs the program for the same values of ω_r as the interacting case, with no Coulomb interaction.
The ground state eigenvector is written to the file `eigenvectors_two-noint_n<N>.dat`.

3.4 Unit tests

Setting `mode` to `unit-tests` runs three unit tests on the different modules in our program, and the result from the tests are written to the file `unittest.txt`.

4 Results

The results created by the program are stored in the `/benchmarks/` folder, while plots generated from these text-files are in `/figures/`. For Jacobi's method, the number of rotations is printed to terminal.

4.1 One Electron in Harmonic Oscillator

From the part of our program that looks at the non-interactive case for one electron we found our lowest three eigenvalues. The result is presented in table 1 where we see the eigevalues as a function of matrix size n.

Table 1: The three lowest eigenvalues for the non interactive case for different n-values. $\rho_{max} = 10$.

n	10	50	100	200	300	400	500
λ_1	2.6867	2.9874	2.9968	2.9992	2.9996	2.9998	2.9998
λ_2	6.1130	6.9369	6.9843	6.9960	6.9982	6.9990	6.9993
λ_3	11.057	10.8452	10.9617	10.9904	10.9957	10.9976	10.9984

But n is not the only variant, we can also change ρ_{max}. In table 2 we see how the eigenvalues change as a function of ρ_{max}. Here we have used $n = 101$.

Table 2: The difference in eigenvalues as a function of ρ_{max} at matrix size $n = 101$.

ρ_{max}	1	5	10	15
λ_1	9.963	2.999	2.997	2.993
λ_2	39.023	6.996	6.985	6.965
λ_3	87.365	10.990	10.962	10.915

Jaobi's method uses a certain number of iterations to reach the point where all non-diagonal numbers are as good as zero. For our program, number of iterations as a function of n is plotted in 1 where we have used the limit $\epsilon = 1.0 \cdot 10^{-8}$ as a threshold.

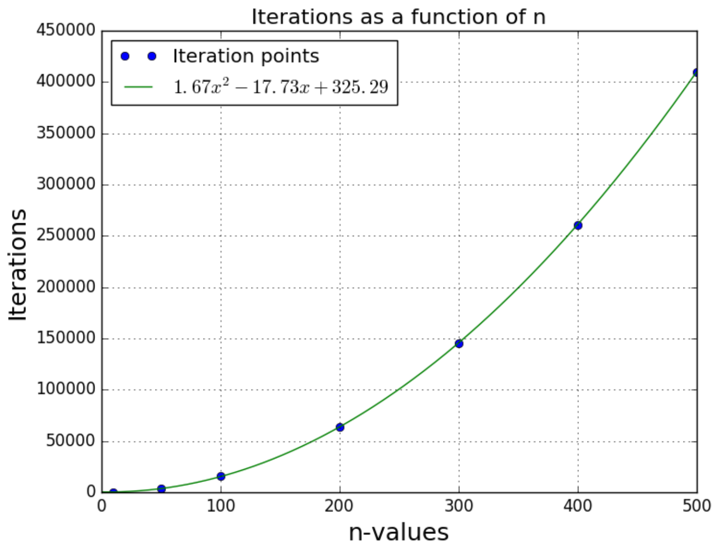

Figure 1: Number of iterations as a function of matrix size n.

In our program we have used two different methods for finding eigenvalues: Jacobi's method and Armadillo. The difference in time for each method is presented in table 3

Table 3: The time used by each method with matrix size n and $\rho_{max} = 10$.

n	Jacobi [ms]	Armadillo [ms]
10	0.27	0.65
50	24.48	1.34
100	339.93	4.79
200	4609.32	34.13
300	27044.07	108.31
400	73842.08	230.47
500	190636.43	436.97

4.2 Two electrons in a harmonic oscillator

We run our program without the repulsive Coulomb interaction. The eigenvectors for the ground state is shown in figure 2.

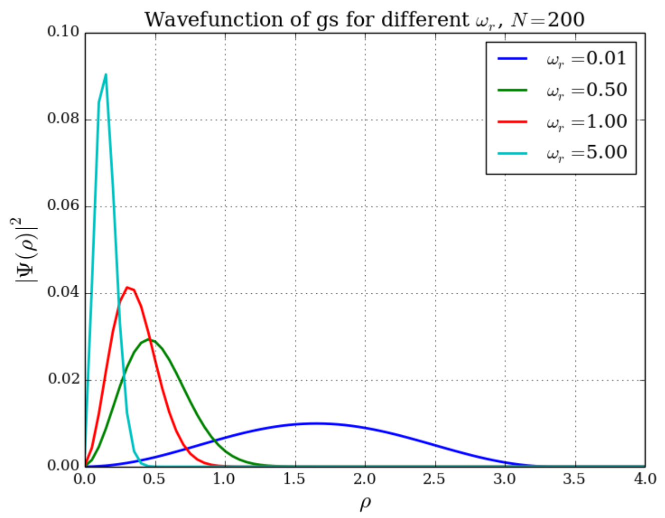

Figure 2: Plot of the ground state eigenvector for different ω_r values, without Coulomb interaction, and matrix dimension $N = 200$. $\rho_{\max} = 10$, but plotted from $\rho = 0$ to $\rho = 4$ for readability.

Running the program with the Coulomb interaction gives us figure 3.

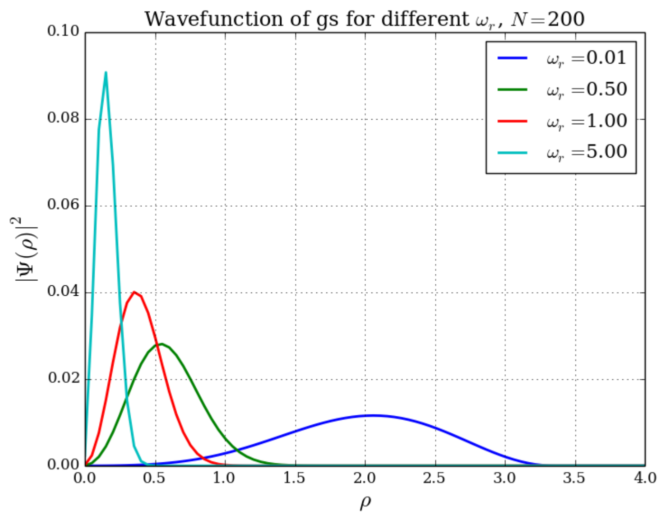

Figure 3: Plot of the ground state eigenvector for different ω_r values, with repulsive Coulomb interaction, and matrix dimension $N = 200$. $\rho_{\max} = 10$, but plotted from $\rho = 0$ to $\rho = 4$ for readability.

We look a bit closer at figure 2 and 3, and plot some of the eigenvectors in the same plot.

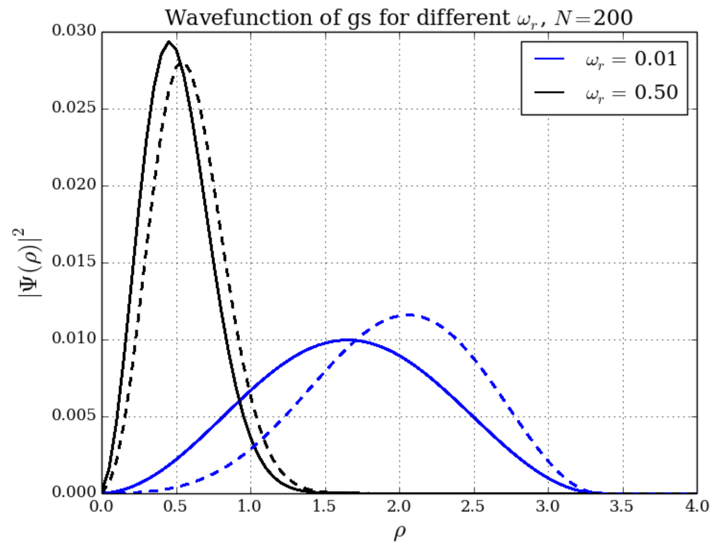

Figure 4: Plot of the ground state eigenvectors for $\omega_r = 0.01$ and $\omega_r = 0.5$ and no Coulomb interaction. Overplotted in dashed lines is the corresponding eigenvector with a repulsive Coulomb interaction added.

5 Discussion

5.1 One Electron in Harmonic Oscillator

As we can see from table 1 the larger matrix we have, the better eigenvalues we get. Our aim was to get four leading digits after the decimal point. But we only achieved this for our lowest eigenvalue. It is clear that for a matrix of size 400x400 the lowest eigenvalue will have four leading digits after the decimal point. The other two are getting nearer four digits, but since we have no repetition of the last digit or the two last digits in respectively the case of λ_2 and λ_3 we can not say anything certain. However, the fact that λ_1 needed $n = 400$ and that λ_2 seems to be very close at $n = 500$ should gives us a hint of how big a matrix we would need. If it was very important for us to have all four digits in the case of λ_3 we would take our time and run the program for higher and higher n-values. Since this is not the case, we are satisfied by stopping here. It is also apparent from table 2 that the eigenvalues are also dependent on ρ_{max} which is logical.

From figure 1 we can see that the number of iterations needed can be approximated as a second degree polynomial as a function of the matrix size n. This is logical as the "area" of the matrix will increase as a function of n^2 giving the program more to work with. This is not a perfect fit as we have too few iteration points to make a good formula, but it works. Again, if a part of a bigger project was dependent on this part of the program being as accurate as possible, I would run the program for higher values of n.

Table 3 clearly shows us that our method is timewise inferior compared to the method Armadillo uses. While we have to wait more than 3 minutes for $n = 500$ with Jacobi, it is done in 0.4 seconds with Armadillo. This is because Jacobi is a very brute force way of finding the eigenvectors. While it works, the problem is that when we have found the largest non-diagonal element of the matrix and set it to zero, we are never sure that it won't become the largest again (even though it will never be as large as it was). This means that we have to look over all the non-diagonal elements for every iteration and that we have to work on the same element multiple times. This again leads to a slow method, even though it gives the results we want. Therefore it is important to think of the accuracy of the eigenvalues compared to the time usage. With high n you can get good accuracy, but it will also take more time. Even though we are pretty sure Armadillo always will be correct we can never know. If this program were to be used to find the eigenvalues of many different matrices, we would have to test out many known cases and see if both methods deliver each time.

5.2 Two electrons in a harmonic oscillator

For our case with two electrons in the harmonic oscillator potential without Coulomb interaction, we look at figure 2 to see how the wave function (squared) behaves when varying ω_r. We see that when increasing ω_r, the peak becomes increasingly higher, while the width decreases. From quantum physics, we recall that when plotting $|\psi|^2$, this corresponds to the probability distribution. We can therefore infer from the figure that as ω_r increases, we are more likely to find the particle closer to $\rho = 0$. This makes a lot of sense, since we earlier said that we treat ω_r as a parameter reflecting the strength of the harmonic oscillator potential, and increasing it causes the potential well to get stronger.

Figure 3 shows the same plot as figure 2, but with a repulsive Coulomb interaction added. It might not be easy to see that much difference between the figures, but if we study them closely, we see that the wave function curves seems to be shifted slightly to the right as compared to without interaction. This would make a lot of sense, as a repulsive interaction between the two electrons should push the electrons apart the closer they are to each other. In doing so, it should counteract (to some degree) the effect of the harmonic oscillator potential.

To more clearly see this, we can look at figure 4, where the eigenvectors for the two lowest ω_r values, with and without interaction is overplotted. Looking at this, we see clearly that when adding the Coulomb interaction, the wave function gets shifted towards the right. We also see that when increasing ω_r, this shift gets smaller, as the harmonic oscillator potential starts to dominate strength wise.

6 Conclusion

From table 1 we can clearly see that the accuracy of the eigenvalues given to us by Jacobi's method is a function of the matrix size n. However, from table 3 we can also see that the time used by the program greatly increases with the increasing of n. Although we have mostly used Jacobi's method here, Armadillo's eigenvalue finder seems to be the better choice in the future.

In the two electron case, we see from figure 4 that when adding the repulsive Coulomb interaction, the probability distributions get shifted away from the center, where the harmonic oscillator potential is the strongest, but as strength of potential ω_r increases, this shift gets smaller.

References

[1] Diagonally dominant matrix. https://en.wikipedia.org/wiki/Diagonally_dominant_matrix.

[2] Github. https://github.com/simennb/fys4150-project-2.

[3] Morten Hjorth-Jensen. Lecture notes. https://github.com/CompPhysics/ComputationalPhysics/blob/master/doc/Lectures/lectures2015.pdf, 2015.