Project 2 FYS4150

Kjetil Karlsen and Vilde Mari Reinertsen

October 2, 2017

Abstract

In this report we used Jacobi's method to solve the eigenvalue problems of finding the energy of a single electron in a harmonic potential and two electrons in a harmonic potential that are interacting by a Coulomb potential. The results were the energies of the ground state of the single electron and the energies of the ground state for the two electrons for different oscillator frequencies. How the eigenvalues converge with Jacobi's method for increasing mesh points was investigated and the CPU time of Jacobi's method was compared to armadillo's solver for eigenvalues, to be able to analyse the method.

The program used in this project can be found at Github.

Contents

1 Introduction

Many problems in physics are eigenvalue problems [referanse. er dette sant egentlig? Ifølge wikipedia ja...]. Especially when we are at the quantum level. The Schrödinger equation with different potentials are an important example and that is the eigenvalue problem this report will attend to.

In this report we will use Jacobi's method to solve an eigenvalue problem. The eigenvalue problem is the Schrödinger equation for both a single electron in a harmonic potential and two electrons in a harmonic potential that are interacting by a Coulomb potential.

The report starts with some theory on how the equations are scaled and made dimensionless and how the problem ends up being a eigenvalue problem with a tridiagonal matrix. On this form, the problem can be solved numerically with Jacobi's method. After that, the algorithm with the unit tests are described in the method. Then, the results are presented in different tables. In the end a discussion of the results are made. At last the report is summarized in the conclusion.

2 Theory

In this report we are looking at how to solve an eigenvalue problem using Jacobi's method. The eigenvalue problem we will solve is the Schrödinger Equation for the ground state and the harmonic oscillator potential. This section is based on the lecture notes written by Jensen [1].

2.1 Dimensionless and scaled Schrödinger Equation

We then start out with the one particle radial Schrödinger Equation (Equation 1 for the ground state which means that the quantum number $l = 0$. Here R is the radial part of the wave function, r is the distance from the origin to the electron, m is the mass of the particle, $\hbar$ is Planck's constant and E is the energy. The harmonic oscillator potential is shown in Equation 2. Here k is the wave number and ω is the oscillator frequency.

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

$$V(r) = \frac{1}{2}kr^2 \quad \text{where } k = m\omega^2 \tag{2}$$

After substituting $R(r) = \frac{1}{r}u(r) = \frac{\alpha}{\rho}u(\rho)$ where $\rho = \frac{r}{\alpha}$, a dimensionless variable, and α is subsequently a constant with dimension length, the radial Schrödinger equation with the harmonic oscillator potential looks like Equation 3.

$$-\frac{\hbar^2}{2m\alpha^2}\frac{d^2}{d\rho^2}u(\rho) + \frac{1}{2}k\alpha^2\rho^2u(\rho) = Eu(\rho) \tag{3}$$

To make the equation a pure eigenvalue problem, we need to scale the equation properly, that we can to with the help of the inserted α. We start by multiplying Equation 3 with $\frac{2m\alpha^2}{\hbar^2}$:

$$-\frac{d^2}{d\rho^2}u(\rho) + \frac{m\alpha^4}{\hbar^2}k\rho^2u(\rho) = \frac{2m\alpha^2}{\hbar^2}Eu(\rho)$$

If we set $\frac{m\alpha^4}{\hbar^2}k = 1$ then $\alpha = \left(\frac{\hbar^2}{mk}\right)^{\frac{1}{4}}$ and the equation is then written as Equation 4.

$$-\frac{d^2}{d\rho^2}u(\rho) + \rho^2u(\rho) = \lambda u(\rho) \quad \text{where } \lambda = \frac{2m\alpha^2}{\hbar^2}E \tag{4}$$

This problem has an analytical solution and the three lowest eigenvalues for the ground state are $\lambda_1 = 3$, $\lambda_2 = 7$ and $\lambda_3 = 11$.

2.2 An eigenvalue problem

We have the scaled and dimensionless Schrödinger Equation (Equation 4) and the next move is to make it into a numerical eigenvalue problem. We start by making the continuous function $u(\rho)$ a discrete number of values $u_i = u(\rho_i)$. Here $\rho_i = \rho_0 + ih$ where h is the step length $h = \frac{\rho_{max}}{N}$, N is the number of mesh points and $\rho_0 = 0$. ρ_{max} should be infinitely big which is not possible, since $r \in [0, \infty)$, but we will come back to that in the method part of the report. Using the expression for the second derivative we get:

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

from Equation 4.

This equation can be rewritten into a matrix eigenvalue equation shown in Equation 5

$$\begin{bmatrix} d_1 & e_1 & 0 & \cdots & 0 & 0 \\ e_1 & d_2 & e_2 & \cdots & 0 & 0 \\ 0 & e_2 & d_3 & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & d_{N-1} & e_{N-1} \\ 0 & 0 & 0 & \cdots & e_{N-1} & d_N \end{bmatrix}\begin{bmatrix} u_1 \\ u_2 \\ u_3 \\ \vdots \\ u_{N-1} \\ u_N \end{bmatrix} = \lambda\begin{bmatrix} u_1 \\ u_2 \\ u_3 \\ \vdots \\ u_{N-1} \\ u_N \end{bmatrix} \tag{5}$$

where $d_i = \frac{2}{h^2} + \rho_i^2$ and $e_i = -\frac{1}{h^2}$.

2.3 Interacting case

In this report we will also study the eigenvalue problem that is the Schrödinger equation of two particles interacting with each other by a Coulomb potential. With relative and center-off-mass coordinates and discarding the center-off-mass energy, a new scaling and insertion of the Coulomb interaction the two particle Schrödinger Equation is shown in Equation 6.

$$-\frac{d^2}{d\rho^2}\psi(\rho) + \omega_r^2\rho^2\psi(\rho) + \frac{1}{\rho}\psi(\rho) = \lambda\psi(\rho) \tag{6}$$

Here $\rho = r/\alpha$ as before, $\omega = \frac{1}{4}\frac{mk}{\hbar^2}\alpha^4$ and reflects the strength of the oscillator, $\alpha = \frac{\hbar^2}{m\beta e^2}$ to get the right scaling, β is Coulomb's constant, $\lambda = \frac{m\alpha^2}{\hbar^2}E$ and E is the energy.

Because of good scaling, this case is almost the same eigenvalue problem as before. The only change is the potential, V_i, which is now $V_i = \omega_r^2\rho_i^2 + \frac{1}{\rho_i}$ instead of only $V_i = \rho_i$. That means that the diagonal of the matrix changes to $d_i = \frac{2}{h^2} + \omega_r^2\rho_i^2 + \frac{1}{\rho_i}$.

This equation has analytical solutions for some frequencies and the resulting energies are listed in 2.1.

Table 2.1: This table lists the eneries of the lowest states in the ground state for two electrons in the harmonic potential and Coulomb potential [?].

Frequency []	Energies $\left[\frac{\hbar^2}{m\alpha^2}\right]$
ω_r	λ_1
0.005	0.35
0.25	1.25

2.4 Jacobi's method

We are using Jacobi's method to solve the eigenvalue problem from Equation 5 in the previous section. We know that for a symmetric matrix $\mathbf{A} \in \mathbb{R}^{n\times n}$ there exists a real orthogonal matrix $\mathbf{S}$ so that $\mathbf{S}^T AS = D$ where D is a diagonal matrix with the eigenvalues on the diagonal.

The idea of eigenvalue problem solving is to do a series of similarity transformations of the matrix $\mathbf{A}$ so that eventually the matrix A is reduced to the matrix D and we have the eigenvalues. $\mathbf{B}$ is a similarity transformation of $\mathbf{A}$ if $\mathbf{B} = \mathbf{S}^T\mathbf{A}\mathbf{S}$.

In Jacobi's method the matrix $\mathbf{S}$ used in the similarity transformations is the orthogonal transformation matrix on the form:

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & \cos\theta & 0 & 0 & \sin\theta \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & -\sin\theta & 0 & 0 & \cos\theta \end{bmatrix}$$

Let s_{ij} be an element in the matrix $\mathbf{S}$ and then $s_{ii} = 1$, $s_{kk} = s_{ll} = \cos\theta$ and $s_{kl} = -s_{kl} = -\sin\theta$ where $i \neq k$ and $i \neq l$. Here k, l are the number of a row or a column and i is the index of the columns and row.

The similarity transformation of a matrix will then change the elements in the matrix. The clue of the method is to choose the angle, θ, so that the elements that are not on the diagonal become zero. Because of the nature of the similarity transformation the eigenvalues stay the same:

$$\begin{aligned} \mathbf{A}\mathbf{x} &= \lambda\mathbf{x} \\ \mathbf{S}^T\mathbf{A}\mathbf{x} &= \mathbf{S}^T\lambda\mathbf{x} \\ \mathbf{S}^T\mathbf{A}\mathbf{S}^T\mathbf{S}\mathbf{x} &= \lambda\mathbf{S}^T\mathbf{x} \\ (\mathbf{S}^T\mathbf{A}\mathbf{S})(\mathbf{S}^T\mathbf{x}) &= \lambda(\mathbf{S}^T\mathbf{x}) \\ \mathbf{B}(\mathbf{S}^T\mathbf{x}) &= \lambda(\mathbf{S}^T\mathbf{x}) \end{aligned}$$

The eigenfunctions change though, but their orthogonality remains because of $\mathbf{S}$ is an orthogonal matrix and then $\mathbf{S}^T\mathbf{x} = \mathbf{U}\mathbf{x}$, a unitary transformation. It can be shown like this:

$$\begin{aligned} \mathbf{w}_i &= \mathbf{U}\mathbf{v}_i \\ \mathbf{w}_i^T\mathbf{w}_j &= (\mathbf{U}\mathbf{v}_i)^T\mathbf{U}\mathbf{v}_j \\ &= \mathbf{v}_i^T\mathbf{U}^T\mathbf{U}\mathbf{v}_j \\ &= \mathbf{v}_i^T\mathbf{v}_j = \delta_{ij} \end{aligned}$$

After a similarity transformation on the matrix $\mathbf{A}$ the elements of the similarity transformation $\mathbf{B}$ becomes:

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

Jacobi's method is to reduce the norm of the non-diagonal elements of the matrix with similarity transformations. We change define $c = \cos\theta$ and $s = \sin\theta$. When we require the non-diagonal elements to be zero that implies that:

$$b_{kl} = (a_{kk} - a_{ll})cs + a_{kl}(s^2 - c^2) = 0$$

We can rearrange some variables and get:

$$(a_{ll} - a_{kk})cs = a_{kl}(s^2 - c^2) \implies \frac{(a_{ll} - a_{kk})}{2a_{kl}} = \frac{1}{2}\left(\frac{s^2}{sc} - \frac{c^2}{sc}\right)$$

We define $\tan\theta = t = s/c$ and $\cot 2\theta = \tau = \frac{a_{ll}-a_{kk}}{2a_{kl}}$ and insert that to the equation:

$$\frac{(a_{ll} - a_{kk})}{2a_{kl}} = \frac{1}{2}\left(\frac{s}{c} - \frac{c}{s}\right) \implies \cot 2\theta = \frac{1}{2}(\tan\theta - \cot\theta)$$

$$\cot\theta = \tan\theta - 2\cot 2\theta = (\tau - 2t)$$

$$\tau = \frac{1}{2}(t - (\tau - 2t)) \implies t^2 + 2\tau t - 1 = 0$$

Which then comes down to:

$$t = -\tau \pm \sqrt{1 + \tau^2} \tag{7}$$

$$c = \cos\theta = \frac{1}{\sqrt{1 + \tan^2\theta}} = \frac{1}{\sqrt{1 + t^2}} \tag{8}$$

$$s = \sin\theta = \tan\theta\cos\theta = tc \tag{9}$$

$$\tag{10}$$

3 Method

3.1 Jacobi's method

$$\begin{aligned} t &= \tau \pm \sqrt{1 + \tau^2} \\ c &= \cos\theta = \frac{1}{\sqrt{1 + \tan^2\theta}} = \frac{1}{\sqrt{1 + t^2}} \\ s &= \sin\theta = \tan\theta\cos\theta = tc \end{aligned}$$

3.2 The Jacobi algorithm

The essence of the algorithm is to continue to do similarity transforms on the matrix A until it is more or less diagonalized. Because the off-diagonal elements will approach 0, it is sufficient to count until the largest off-diagonal element is $< 10^{-10}$.

Our matrix $\mathbf{A}$ is symmetric, which means that $\mathbf{A}^T = \mathbf{A}$. From this it follows that the product $\mathbf{S}^T AS$ also is symmetric, as is shown under:

$$\mathbf{S}^T\mathbf{A}\mathbf{S} = \mathbf{S}^T\mathbf{A}^T(\mathbf{S}^T)^T \tag{11}$$
$$= \mathbf{S}^T\mathbf{A}\mathbf{S} \tag{12}$$

Thus it is sufficient to search only the elements above the diagonal for the highest valued element in the matrix, instead of the entire matrix. This greatly speeds up the searching-algorithm, as you half the number of elements to read in order to find the greatest value.

Depending on the sign of the ± in equation 7 one either gets a relatively high value for t or a small one. We chose to only use the smallest solution of that equation, see snippet below. This means that we choose the c which will be closest to unity and the value of s which will be the smallest, see equations 8 and 9. This against means we each time chose the similarity transform that rotates the matrix A the most.

Listing 1: Determining t in the program

```
if(tau>= 0)
        t = 1.0/(tau + sqrt(1+tau*tau));

else
        t = -1.0/(- tau + sqrt(1+tau*tau));
```

A computer has a finite number of digits to describe a number. For high values of τ equation 7 will turn to $t = \tau \pm \tau$, as $\sqrt{1+\tau^2} \simeq \tau$ for large τ. To minimize this effect we rewrote the equation:

$$t = \frac{(-\tau \pm \sqrt{1+\tau^2})(-\tau \mp \sqrt{1+\tau^2})}{-\tau \mp \sqrt{1+\tau^2}} \tag{13}$$
$$= \frac{-1}{-\tau \mp \sqrt{1+\tau^2}} \tag{14}$$

We chose to stop the algorithm when all the off-diagonal elements in A where smaller than $\epsilon = 10^{-10}$. For greater accuracy this number could be smaller, but this gives a decent payoff between time and accuracy.

The strength of the oscillator potential is reflected in ω_r. If ω_r is small, the wave function will be less confined and will thus need a higher ρ_{max} than for strongly confined wave functions. This leads to choosing $\rho_{max} = 50$ for $\omega < 0.1$ and $\rho_{max} = 7.9$ for the other values of ω_r.

3.3 Unit tests

In this project we used two unit tests to ensure that the program performs as expected and delivers accurate enough results.

One way to be sure that our algorithm gives correct answers is to task it to find the eigenvalues of a matrix with known eigenvalues. These values had been pre calculated by Matlab.

The other unit test we utilized was to check that our algorithm to find the largest off-diagonal elements actually found the largest off-diagonal elements. This was done by setting up a known matrix and tasking our algorithm to find the largest off-diagonal element and checking it against the manually found largest element.

4 Result

Table 4.1: Convergence of three smallest eigenvalues for the non-interacting case using the Jacobi method.

N	λ_1	λ_2	λ_3
10	2.9263	6.61967	10.0351
100	2.99928	6.99642	10.9919
200	2.99982	6.99911	10.9986
400	2.99996	6.99979	11.0003

The eigenvalues converges towards the excact solutions $\lambda_1 = 3, \lambda_2 = 7$ and $\lambda_3 = 11$ for each increase in mesh size $(N \times N)$ in table 4.1. The lower the eigenvalue, the quicker it converges.

The energies corresponding to the dimensionless energies $\lambda_1 = 3$, $\lambda_2 = 7$ and $\lambda_3 = 11$ are 81.6 eV, 190.4 eV and 299.3 eV respectively.

Table 4.2: Comparisson of the time used between the Jacobi-algorithm and the Armadillo function eig_sym as a function of the mesh size N of matrix A. "Transforms" refers to the amount of similarity transforms the Jacobi method uses.

N	Transforms	time Jacobi (s)	time Armadillo (s)
10	144	3.400e-05	4.400e-05
100	17681	2.026e-01	1.567e-03
200	71500	3.360e+00	8.803e-03
400	287490	6.467e+01	2.516e-02

As expected, the brute force method of the Jacobi algorithm is much slower than the algorithm used by armadillo and LAPACK. LAPACK is up to approximately 2 500 times faster. The higher calculation costs of the Jacobi method is also reflected in the amount of similarity transforms required. In figure 4.1 the development of the amount of similarity transforms as a function of N was approximated with the function $y = 1.806 * (x - 1.039)^2 + b$. This shows that for our tridiagonal matrix and choices of ϵ, the Jacobi method converges after approximately $1.81n^2$ similarity transforms.

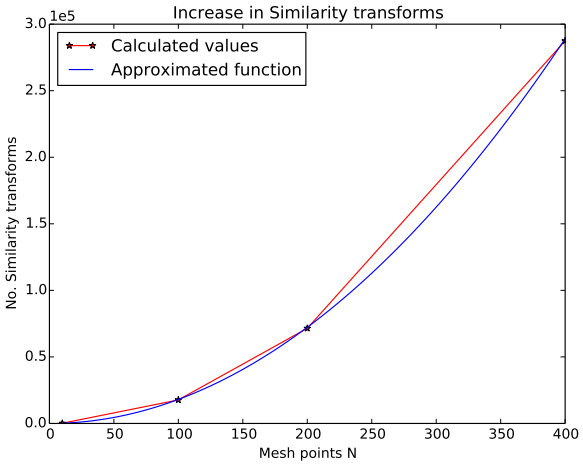

Figure 4.1: Number of similarity transforms as a function of N approximated with a non-linear least squares fit.

Table 4.3: Eigenvalues for the interacting case. All the calculations are done with a mesh size of N=400. .

ω_r	λ_1	λ_2	λ_3
0.01	0.106	0.142	0.178
0.05	0.350	0.532	0.719
0.25	1.250	2.193	3.192
0.5	2.230	4.134	6.074
1	4.058	7.909	11.818
5	17.445	37.057	56.817

Table 4.3 shows that higher values of ω_r yields higher energies, which is as expected from section 3.2. The values of ω_r = 0.05 and ω_r = 0.25 corresponds to exactly double that of what is found in the article by M. Taut, [2].

5 Discussion

5.1 The boundaries - ρ_{max}

Because we used the radial Schrödinger Equation our boundary conditions are given by $r \in [0, \infty)$ and since $\rho = r/\alpha$ the same boundary conditions are still relevant. That means that ρ_{max} should be ∞ that is not possible however on a computer.

Therefore an approximation of the boundary conditions had to be made. We chose ρ_{max} =?? for all our calculations except when ω_r = 0.01 then we changed it to be ρ_{max} = 50. The small frequency made the wave function smear out, and we needed a bigger interval to make sure we had the whole function in our calculations.

We also noticed that the result was very dependent on ρ_{max} especially if we chose a too small value. We used our knowledge of the analytical result to find a good ρ_{max}, but without this knowledge it would have been difficult. It is important to make sure that the boundary conditions are reflecting the reality in a good way to get the right results.

5.2 Efficiency of method

As we can see in Table 4.2 the CPU time of our Jacobi's method is big compared to armadillo's eigenvalue solver function. We needed 400 mesh points to get a good accuracy for the eigenvalues and if we would have needed more it would have been a very time consuming job.

The number of similarity transformations was big and it give a picture of why the CPU time is big. The similarity transformations increase a lot with the number of mash point, so if the method converged with a lower number of mesh points, it could decrease the CPU time.

No matter how we changed the number of mesh points and ρ_{max}, we could not get a better accuracy then three decimal points, with at least a reasonable CPU time. We did not try for mesh points resulting in CPU times over four minutes. In our mind, that is to long.

This might indicate that we should have chosen another method to solve this eigenvalue problem, if it was not a project to learn about programming.

A interesting point we noticed was that our number of similarity transformations was lower then the one presented in the lectures for Jacobi's method. Here it was presented that a general, real and symmetric matrix requires $3n^2 - 5n^2$ similarity transformations. In this specific case of a tridiagonal matrix, figure 4.1 shows that it converges after $1.81n^2$ similarity transforms.

This might be because we have a tridiagonal matrix, our matrix is full of zeros already, and because we exploit the fact that our matrix is symmetric when searching for the largest element, like mentioned in method. The number of similarity transformations are also dependent on the tolerance of when a number is so small that we reckon it to be zero. Increasing the tolerance would give fewer similarity transformations.

5.3 Unit test. Kjetil

Our unit tests checked that the entire jacobi algorithm gave correct answers when presented to a matrix with known eigenvalues and that our searching algorithm found the largest off-diagonal element. The strength of the first test is that it checks the entire system, but will not indicate what is wrong. This could be solved by writing more unit test, like checking that the norm of the matrix was conserved through the transform, which would fail if there where for instance round-off-errors.

If the algorithm were to be used on a different matrix it would be useful to check that this matrix is symmetric. In addition the column-vectors of the transformed matrix A must be orthogonal if the transformation is to be unitary. A simple for loop checking the inner product between with different column-vectors would suffice.

6 Conclusion

We used Jacobi's method to solve the eigenvalue problems of finding the energy of a single electron in a har-

monic potential and two electrons in a harmonic potential that are interacting by a Coulomb potential. First we scaled the different Schrödinger equations and made them dimensionless. We then rewrote the equation to a matrix equation. Then the problem ended up being an eigenvalue problem with a tridiagonal matrix. The scaling made it easy to adapt the algorithm for both cases with just changing the matrix involved, in this case the term for the potential. Two unit tests made sure the program was doing what it was supposed to do and the results were the energies of the ground state of the single electron and the energies of the ground state for the two electrons for different oscillator frequencies. The results were compared with the analytical values and the accuracy was down to the third decimal point with 400 mesh point and a step length of ??. The CPU time of Jacobi's method was compared to armadillo's solver for eigenvalues and our algorithm using Jacobi's method was very slow compared to armadillo's.

As mentioned in the discussion Jacobi's method was not the best method to use with this problem. It was slow and it did not give us better accuracy for the eigenvalues then three decimal points. The program could therefore have been better is we chose another method. We could choose a method that exploits the fact that our matrix is a tridiagonal matrix. Jacobi's method could also have been optimized by checking were the program used the most time and also specialising the steps, h.

References

[1] Morten Hjorth-Jensen. Computational physics: Lecture notes fall 2015. Department of Physics, University of Oslo, 8 2015. Chapter 2 and 6.

[2] 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.