Solving eigenvalue problems using similarity transformations

Anders Bråte
Institue for Physics
University of Oslo
Norway
October 1, 2019

In modern numerical science one often express physical situations through the use of matrices. Hence being able to accurately calculate eigenvectors and eigenvalues is often crucial. We have looked closely at the Jacobi algorithm for solving eigenvalue problems on real and symmetrical matrices and shown how this method works mathematically, as well as compared our implemented Jacobi algorithm with the numerical library 'armadillo's eigenvalue solver. We do this by looking at a buckling beam, or oscillating spring example, as well as two quantum mechanical harmonic oscillator cases. We found that for the buckling beam case we could get very accurate results, yet not as fast as armadillos algorithm. We also found our algorithm lacking when it came to the more complex cases for the harmonic oscillator. In general armadillos pre-made module was both easier to implement, and faster.

I. INTRODUCTION

In modern science one often finds oneself required to work with large matrices that describe a certain system, and so finding the attributes and traits of how this matrix behaves is often paramount. As many have experienced first hand, working analytically with a large, often messy matrix is not very time efficient so using numerical methods comes naturally, though not without its consequences. All numerical calculations involve truncation and approximation, as only a limited amount of decimal points is possible for each variable. This means that we need to be aware of how this affects our results. Due to this we choose systems which we know has simple analytical solutions, so that it is possible to compare and contrast the results.

Initially we will describe the general algorithm known as the Jacobi method, and the mathematics that lay the groundwork. We will then apply the algorithm to a 'buckling beam' problem, which is a relatively simple fixed boundary problem, and compare and contrast our results. Then we will apply our algorithms to a quantum mechanic infinite well for one and two electrons. Though far more complex than the first example, surprisingly little needs to be change as we shall see later.

Throughout the mathematical calculations of the different problems we will also include scaling of equations, since numerical calculations need values for each variable a certain calculation will not be valid for many different scenarios. For a dimensionless mathematical model, no physical variables or constants needs to be defined, which in large part simplifies and generalises our results and algorithms.

Naturally there are alternatives to the Jacobi method in form of pre-made modules which perform similar calculations. We will also compare certain results from our algorithm to those of the module 'armadillo' which also computes eigenvalues for matrices.

A. Reproducibility

All code is present on GitHub [3] and follows a fairly simple layout. All algorithms are in 'main.cpp', which generates data in the form of text files, which is then read and plotted by 'readfile.py'.

II. THEORY

In general the eigenvalues of a matrix A is given by the matrix equation

$$Ax = \lambda x \tag{1}$$

where λ and x denotes the eigenvalues and eigenvectors of the matrix A respectively.
The Jacobi method applies a series of similarity transformations on a matrix, until the initial matrix is transformed into a diagonal matrix, with the eigenvalues as diagonal elements. The thing that makes this possible is that similarity transformations change the

eigenvectors but not the eigenvalues.

Given a real and symmetric matrix A, we say that B is a similarity transformed matrix of A if

$$B = S^T AS, \tag{2}$$

and

$$S^T S = S^{-1} S = I \tag{3}$$

Here I is the identity matrix. The way we can prove that the eigenvalues stay the same is by looking at equation 2 and 3. We can multiply equation 2 with s^T on the left side, and insert $I = S^T S$ which in matrix operations is like multiplying with one, and we get the expression

$$S^T ASS^T x = \lambda S^T x$$

Looking back at equation 3 we see that what we have on the left side of the equation is equal to $B(S^T x)$ which gives us the equation

$$B(S^T x) = \lambda(S^T x).$$

This shows us that λ is also an eigenvalue for B but with a eigenvector $S^T x$ instead of just x.

Now that we know we can find the eigenvalues of a matrix we need to find how to describe our system with a matrix.

The buckling beam problem can be described by a differential equation that reads

$$\gamma \frac{d^2 u(x)}{dx^2} = -Fu(x). \tag{4}$$

What we're after in this equation is the displacement of the beam given by $u(x)$. F is a force applied to the beam somewhere along its length L, and γ is a material constant which is dependent on many of the structural properties of the given beam. We know that the beam is fastened at both ends, so we can apply the following boundary conditions $u(0) = u(x) = 0$.

We can assume that all our parameters γ, L and F are known, meaning we can focus on scaling of the variables. We start with the length by defining

$$\rho = \frac{x}{L}, \tag{5}$$

meaning $\rho \in [0, 1]$.

If we rewrite equation 4 we can get the expression

$$\frac{d^2 u(\rho)}{d\rho^2} = -\frac{FL^2}{\gamma} u(\rho) \tag{6}$$

where we can define the constant $\lambda = \frac{FL^2}{\gamma}$ to get the expression

$$\frac{d^2 u(\rho)}{d\rho^2} = -\lambda u(\rho). \tag{7}$$

This is a typical differential equation which in order to solve numerically we must discretize. If we look at the definition of the double derivative we have the expression

$$u''(x) = \lim_{h \to 0} \frac{u(x+h) - 2u(x) + u(x-h)}{h^2}, \tag{8}$$

which we can use when describing our equation above. h is in practice the step-length and is defined as $h = L/n$ where n is the number of points, and therefore also the size of our matrix.

$$\lim_{h \to 0} \frac{u(\rho+h) - 2u(\rho) + u(\rho-h)}{h^2} = -\lambda u(\rho). \tag{9}$$

This is where numerical inaccuracies arise, since we can't in practice get to where h reaches zero we instead have to have it very small and live with the inaccuracies that arise due to this.

We now have an equation that lends itself nicely to discretization. We can rewrite the equation above to get the expression

$$u(\rho+h) - 2u(\rho) + u(\rho-h) = -h^2 \lambda u(\rho)$$

Now the actual discretization comes when we define $u_i = u(\rho)$, $u_{i-1} = u(\rho - h)$ and $u_{i+1} = u(\rho + h)$ which means we can write the equation above like this

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

Since we discretized for n steps, we get n of the above equation,

$$\begin{aligned}
i = 1: &\qquad -1/h^2(u_2 + u_0 - 2u_1) = \lambda u(\rho_1), \\
i = 2: &\qquad -1/h^2(u_3 + u_1 - 2u_2) = \lambda u(\rho_2), \\
&\qquad \vdots \\
i = n: &\qquad -1/h^2(u_{n+1} + u_{n-1} - 2u_n) = \lambda u(\rho_n),
\end{aligned}$$

which can be written as the following matrix and vectors.

$$\mathbf{A} = \begin{bmatrix} \frac{2}{h^2} & -\frac{1}{h^2} & 0 & 0 & 0 \\ -\frac{1}{h^2} & \frac{2}{h^2} & -\frac{1}{h^2} & 0 & 0 \\ 0 & \ddots & \ddots & \ddots & 0 \\ 0 & 0 & \ddots & \ddots & -\frac{1}{h^2} \\ 0 & 0 & 0 & -\frac{1}{h^2} & \frac{2}{h^2} \end{bmatrix} \cdot \begin{bmatrix} u_1 \\ u_2 \\ \vdots \\ \vdots \\ u_n \end{bmatrix} = \begin{bmatrix} u(\rho_1) \\ u(\rho_2) \\ \vdots \\ \vdots \\ u(\rho_n) \end{bmatrix}.$$

We now have our system expressed as a Toplitz matrix, and we know that applying similarity transformations to the matrix will not change the eigenvalues. With this we can choose our matrix such that the elements on the off diagonal become zero. This is the crux of the Jacobi method. By applying a rotational matrix to A we can set an element in the matrix to be zero.

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

The index of the cosines and sines are such that $s_{kk} = s_{ll} = \cos\theta$ and $s_{kl} = -s_{lk} = -\sin\theta$

This matrix rotates an element in euclidean space θ degrees, and we can choose which element it rotates by changing the index of the cosine and sine. If we choose the correct angle we can set an element of our choosing to be zero. One thing to note is that rotating one element will change the value of other elements, however over many iterations the matrix will be closer to the diagonal matrix. This can be shown by looking at the Frobenius norm of the transformation (see lecture notes chap. 7.4 page 217 [1]).

The set of equations which results when applying the rotational matrix to our Toplitz matrix $B = S^T AS$ is as follows

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

What we then seek to achieve is for all the off-diagonal elements b_{kl} to be zero, or at least as close as we wish for them to be.

For simplicity we define cos as c, sin as s and tan $= s/c$ as t. We then define a new variable $\tau = \frac{a_{ll}-a_{kk}}{2a_{kl}}$. From this we obtain the quadratic equation $t^2 + 2\tau t - 1 = 0$ which results in

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

From this we can use $t = s/c$ and find

$$c = \frac{1}{\sqrt{1+t^2}}$$

with $s = tc$. It is important to choose t to be the smaller of the two roots, since we then ensure that $|\theta| \leq \pi/4$, which in turn minimises the difference between the two matrices after the transformation.

For finding and comparing the analytical eigenvalues we can use the following expression

$$\lambda_j = d + 2a\cos\left(\frac{j\pi}{N+1}\right) \tag{10}$$

A. Three-dimensional harmonic oscillator, one electron

In order to show the versatility of the eigenvalue solver we wish to look at a quantum mechanical harmonic oscillator with first one, then two electrons which interact through electromagnetic Coulomb forces. We start off with the radial Schroedinger's equation

$$-\frac{\hbar^2}{2m}\left(\frac{1}{r^2}\frac{d}{dr}r^2\frac{d}{dr} - \frac{l(l+1)}{r^2}\right)R(r) + V(r)R(r) = ER(r) \tag{11}$$

In our case the potential V is $(\frac{1}{2})kr^2$ with $k = m\omega^2$ and E is the energy given by the quantum numbers n, which is the energy level, and l which denotes the orbital momentum.

$$E_{nl} = \hbar\omega\big(2n + l + \frac{3}{2}\big)$$

For simplicity we will only look at the case for $l = 0$.

Since we are looking at particles in radial coordinates it means that the distance from the origin r is in the interval $[0, \infty]$. However, infinity is not very easy to use when calculating numerically which means that this also needs to be approximated. Our boundary conditions in this case is $u(0) = u(\infty) = 0$. Like earlier we introduce a dimensionless variable for position given by $\rho = \frac{r}{\alpha}$, where we define α a bit later.

Inserting the potential, and $l = 0$ we get the following

$$-\frac{\hbar^2}{2m\alpha^2}\frac{d^2}{d\rho^2}u(\rho) + \frac{k\alpha^2\rho^2}{2}u(\rho) = Eu(\rho). \tag{12}$$

rewriting this a bit by multiplying the constant $\frac{\hbar^2}{2m\alpha^2}$ over to the right side we get

$$-\frac{d^2}{d\rho^2}u(\rho) + \frac{mk}{\hbar^2}\alpha^4\rho^2 u(\rho) = \frac{2m\alpha^2}{\hbar^2}Eu(\rho) \tag{13}$$

From here we can define our constant α so that the expression $\frac{mk}{\hbar^2}\alpha^4 = 1$. this means α must be the following

$$\alpha = \left(\frac{\hbar^2}{mk}\right)^{1/4}. \tag{14}$$

For simplicity we also wish to define a new variable λ to be the following

$$\lambda = \frac{2m\alpha^2}{\hbar^2}E. \tag{15}$$

With these defined we can rewrite the equation above 13 to the following

$$-\frac{d^2}{d\rho^2}u(\rho) + \rho^2 u(\rho) = \lambda u(\rho) \tag{16}$$

This equation we again discretize as we did with the buckling beam example. And again we need to define our step size. This will be more important in this example, since we in theory look at the wave function as it propagates til infinity. Dividing infinity in n pieces in order to calculate the wave function numerically is hardly practical, so we need to look at how accurate the results are for different step sizes n and our approximation of infinity.

We define the step length to be $h = \frac{\rho_n}{n}$. Here ρ_n is the maximum distance, or rho_i at $i = n$.

Using the same logic as previous we use the definition of the derivative to rewrite equation 16

$$-\frac{u_{i+1} - 2u_i + u_{i-1}}{h^2} + \rho_i^2 u_i = -\frac{u_{i+1} - 2u_i + u_{i-1}}{h^2} V_i u_i = \lambda u_i \tag{17}$$

the matrix we then end up with looks as follows

$$\begin{bmatrix} \frac{2}{h^2} + V_1 & -\frac{1}{h^2} & 0 & \dots & 0 & 0 \\ -\frac{1}{h^2} & \frac{2}{h^2} + V_2 & -\frac{1}{h^2} & \dots & 0 & 0 \\ & \ddots & \ddots & \dots & \frac{2}{h^2} + V_{n-1} & -\frac{1}{h^2} \\ 0 & 0 & 0 & 0 \dots & -\frac{1}{h^2} & \frac{2}{h^2} + V_n \end{bmatrix}$$

B. Harmonic oscillator with two electrons

The schroedinger equation for two interacting electrons in a similar setting as the previous example is given by the following lengthy expression

$$\left(-\frac{\hbar^2}{2m}\frac{d^2}{dr_1^2} - \frac{\hbar^2}{2m}\frac{d^2}{dr_2^2} + \frac{1}{2}kr_1^2 + \frac{1}{2}kr_2^2\right)u(r_1, r_2) = E^{(2)}u(r_1, r_2) \tag{18}$$

Here $E^{(2)}$ is the two-electron energy, which can be expressed as the sum of a relative energy, and a centre of mass energy $E^{(2)} = E_r + E_R$. We can do this due to the wave function ansatz which states $u(r, R) = \psi(r)\phi(R)$. Here we have defined $\vec{R} = \frac{1}{2}(r_1 + r_2)$ and $\vec{r} = r_1 - r_2$.

The potential given by the Coulumb forces acting upon the particles is given by the expression

$$V(r_1, r_2) = \frac{\beta e^2}{\vec{r}}, \tag{19}$$

where $\vec{r} = |\vec{r_1} - \vec{r_2}|$, $\beta e^2 = 1.44$ eVnm and $\beta = \frac{1}{4\pi\epsilon_0}$. Using the potential above, and focusing only on $\psi(r)$ we get the expression

$$\left(-\frac{\hbar^2}{m}\frac{d^2}{dr^2} + \frac{1}{4}kr^2 + \frac{\beta e^2}{r}\right)\psi(r) = E_r\psi(r) \tag{20}$$

Similarly to the previous example we introduce a dimensionless variable for distance from the origin given by $\rho = \frac{r}{\alpha}$. And like last time, we can rewrite the equation to the following

$$-\frac{d^2}{d\rho^2}\psi(\rho) + \frac{mk}{4\hbar^2}\alpha^4\rho^2\psi(\rho) + \frac{m\alpha\beta e^2}{\rho\hbar^2}\psi(\rho) = \frac{m\alpha^2}{\hbar^2}E_r\psi(\rho) \tag{21}$$

Again we wish to define a series of constants and variables which makes this expression a whole lot prettier. beginning with a frequency

$$\omega_r^2 = \frac{mk}{4\hbar^2}\alpha^4. \tag{22}$$

Just like last time we now define our previously undefined constant α by requiring that it makes the following constants equal one

$$\frac{m\alpha\beta e^2}{\hbar^2} = 1 \tag{23}$$

and therefore

$$\alpha = \frac{\hbar^2}{m\beta e^2}. \tag{24}$$

We also define our another constant λ for simplicity

$$\lambda = \frac{m\alpha^2}{\hbar^2}E. \tag{25}$$

With all these combined we can finally arrive at our final equation which reads

$$-\frac{d^2}{d\rho^2}\psi(\rho) + \omega_r^2\rho^2\psi(\rho) + \frac{1}{\rho} = \lambda\psi(\rho). \tag{26}$$

Following the same steps as earlier, with regards to discretizing we end up with a matrix that looks as follows

$$\begin{bmatrix} \omega_r^2\rho^2 + \frac{1}{\rho} & -\frac{1}{h^2} & 0 & \dots & 0 & 0 \\ -\frac{1}{h^2} & \omega_r^2\rho^2 + \frac{1}{\rho} & -\frac{1}{h^2} & \dots & 0 & 0 \\ & \ddots & \ddots & \dots & \frac{2}{h^2} + V_{n-1} & -\frac{1}{h^2} \\ 0 & 0 & 0 & 0 \dots & -\frac{1}{h^2} & \omega_r^2\rho^2 + \frac{1}{\rho} \end{bmatrix}$$

In our case we will only conscider cases for $l = 0$, and for a few varying values of ω_r.

III. RESULTS

A. Buckling beam

The eigenvectors for $n = 10$ is listed below. The number of iterations needed to reach an accuracy of $10 \cdot 10^{-10}$ was 160. This took less than 0.001 seconds.

For $n = 100$ it took 18443 to reach the same accuracy. The time used for this was 1.764 seconds. for comparison it took the armadillo algorithm 0.016 seconds to find the same eigenvalues.

analytical eigenvalue	Jacobi	armadillo
8.1014	8.1014	8.1014
3.1749E1	3.1749E1	3.1749E1
6.9028E1	6.9028E1	6.9028E1
1.1692E2	1.1692E2	1.1692E2
1.7154E2	1.7154E2	1.7154E2
2.2846E2	2.2846E2	2.2846E2
2.8308E2	2.8308E2	2.8308E2
3.3097E2	3.3097E2	3.3097E2
3.6825E2	3.6825E2	3.6825E2
3.9190E2	3.9190E2	3.9190E2

Table I. The analyical and numerical eigenvalues for the buckling beam problem with $n = 10$. Both jacobi and armadillo methods used less than 0.001 seconds to compute

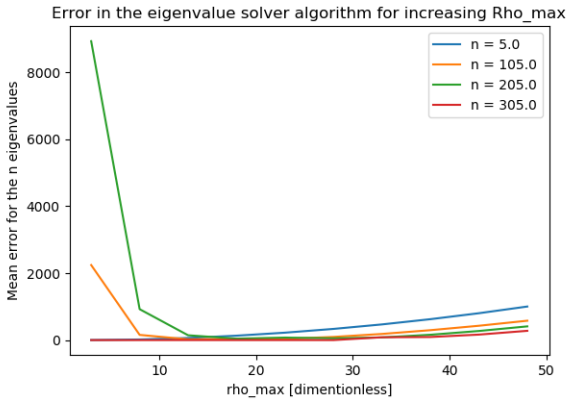

Figure 1. the mean relative error over varying rho for different values of n to visualise the best selection of ρ_{max} and n

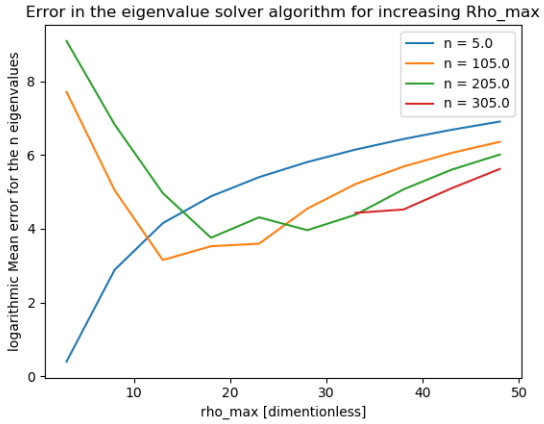

Figure 2. The men relative error over varying rho plotted logarithmicly for different values of n. $n = 305$ could not be plotted due to the RAM limitations on the hardware used.

B. Harmonic oscillator with one electron

For a relativly small n the errors are relatively large, however for $n = 100$ the larges difference was only 69.9

analytical eigenvalue	jacobi & armadillo	difference
3	3.1097	1.0970E-1
7	9.9005	2.9005
11	2.1144E1	1.0144E1
15	3.6892E1	2.21892E1
19	5.7141E1	3.8141E1
23	8.1890E1	5.8890E1
27	1.1114E2	8.4140E1
31	1.4489E2	1.1389E2
35	1.8314E2	1.4814E2
39	2.2589E2	1.8689E2

Table II. The analytical and numerical values from both jacobi and armadillo for the harmonic oscillator with one electron. All the values from both of these algorithms were identical. The difference between the analytical and numerical values increase as n increases.

with a ρ_{max} of 15.

C. Harmonic oscillator with two electrons

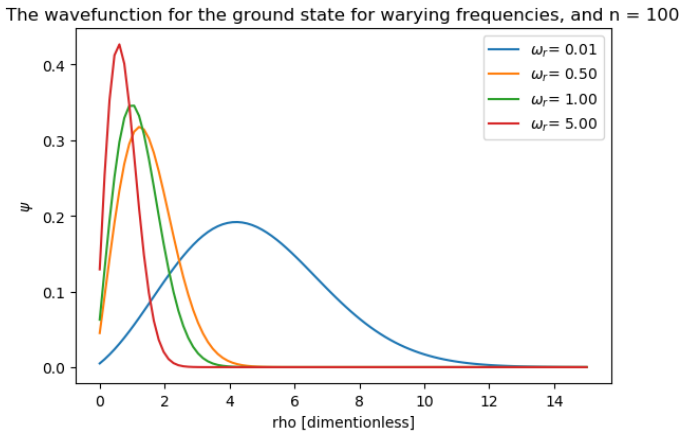

Figure 3. The wavefunction of the groundstate for the harmonic oscillator with two electrons and varying frequencies.

$\omega = 0.01$	$\omega = 0.5$	$\omega = 1$	$\omega = 5$
-0.2966	1.7276	2.8860	1.1911E1
0.0481	4.8616	9.3463	4.5336E1
0.3766	1.0358E1	2.0477E1	1.0147E2
0.6815	1.8172E1	3.6169E1	1.8017E2
0.9562	2.8262E1	5.6385E1	2.8138E2
1.1696	4.0614E1	8.1112E1	4.0511E2
1.3354	5.5222E1	1.1035E2	5.5135E2
1.6306	7.2085E1	1.4408E2	7.2008E2
2.0497	9.1200E1	1.8232E2	9.1132E2
2.6640	1.1258E2	2.2507E2	1.1251E3

Table III. The numerical eigenvalues for $n = 10$ and $\rho_{max} = 15$. Both the jacobi method and armadillos algorithm calculated the exact same result.

IV. DISCUSSION

Our use of scaling can be said to have saved a lot of time in coding, as we can see in the little difference there is between these three very different cases. If one looks at the equations we start with for the buckling beam case 4 and the two electron case 18, it is quite remarkable that the only difference in the numerical mathematics is the addition of a simple Coulomb potential and a step length. This shows really the practicality of scaling in numerical calculations. -jacobi using 1.827s and arma using 0.016s

Looking at the one electron harmonic oscillator example we see that as n increases, so does the accuracy. As shown in figure 2, there is a significant difference in the mean error for the different values of n. However independently from how large n is, the error still does accumulate for the larger values of n.
It is interesting to note that as ρ_{max} increased, so did the relative error. It would be logical to think that a large number would be a better approximation to intifity, however the larger ρ is, the larger each stepsize is. Looking at figure 1 it would be logical to conclude that a relativly large n, and a ρ_{max} of around 15 would provide almost the best accuracy of most other values. With this said, there is a tendency for the accuracy to get better and better for larger n, so if hardware limitations isn't an issue, then there would be no reason to not increase n at the cost of run time.

The fact that the values generated by both our jacobi algorithm, and armadillos 'eig_sym' function might indicate that something is amiss in our algorithm. However the behaviour of the algorithm is as expected apart from this. The accuracy of the calculations increased with larger n, and the ratio between n and ρ is really what made the bigger difference in the one electron harmonic oscillator case.

Another aspect to consider is the time used by the two algorithms. For the buckling beam case with $n = 100$ our jacobi algorithm made 18443 iterations which took 1.827 seconds. The armadillo algorithm on the same matrix used 0.03 seconds. We have not looked into what algorithms and methods armadillo applies, however the difference is significant.
It is likely that armadillo uses a different algorithm, since the efficiency of the jacobi algorithm is rather poor. We have seen that a $n = 100$ matrix needs 18443 rotations, each of which require $4 \cdot n$ operations. The inefficiency on this method is due to the fact that one one element in the matrix is reduced to zero, other elements that might have already been reduced changes value.

For the two electron case we used $\rho_{max} = 15$ and $n = 100$, since this proved relatively accurate in the one electron case, without being too demanding on the hardware. One can consider doing similar test as with the one electron case to see if the difference in the mathematical expression would have made a major difference, however we concluded that since there is so little difference in the mathematics due to our use of scaling, it would not be neccesary to achieve the desired accuracy.
The error for the higher energy states became increasingly large, and we did not seem to be able to bring them down to managable levels. The highest value of 186.8 can hardly be accepted as an approximation at all. Wether this is due to a poor choice of ρ_{max} and n, or if it simply is a weakness in the algorithm will need further research.

V. CONCLUSION

We have shown how mathematically the jacobi method is a fairly general method for diagonalizing matrices numerically, and that the use of scaling can further generalise the algorithm. For the simple case of the buckling beam our algorithm with $n = 10$ it took our algorithm 160 iterations to reduce the matrix down with an accuracy of 10^{-10}. Both the Jacobi algorithm and the armadillo module used less than 0.001 seconds, and obtained the same result. The approximation to the analytical result was very good, and we were able to achieve the wanted accuracy III A.

For the more advanced case of the harmonic oscillator the case was further complicated due to the necessity to approximate infinity. We therefore looked at the mean relative error of the numerical and analytical eigenvalues and plotted these for different values of n and ρ_{max} 1. From the plot it would seem that a relatively small rho_{max} of 15 and a n of around 100 would provide the best results with the hardware available. Despite this the error in our algorithms is very large. As displayed in table III A the difference, especially for the higher energy states is very large, so much so that they can barely be considered results. The ground state however is accurate.
For the two particle case we got corresponding eigenvalues for $n = 10$ displayed in table III C. Whether these are accurate is not known. The eigenvectors corresponding to the ground state for varying frequencies is shown in fig 3.
We can clearly conclude that further work is needed with this algorithm. Our results with armadillo and Jacobi can indicate that our algorithm is not perfect and that improvements need to be made in order to achieve accurate results for more complex cases.

[1] Jensen, M. H., Lecture in FYS3150 29/8-19.

[2] Weisstein, Eric W. "Frobenius Norm." From MathWorld–A Wolfram Web Resource.

http://mathworld.wolfram.com/FrobeniusNorm.html
[3] https://github.com/andebraa/FYS3150-prosjekt2