Solving eigenvalue problems using numerical algorithms

FYS3150 - Project 2

Philip Hoel

September 30, 2020

UiO : Universitetet i Oslo

Contents

1 Abstract

A project testing the jacobi rotation algorithm on three different eigenvalue problems. Through the use of C++, we test the accuracy and the speed of the algorithm. A short look on discretizing equations and rewriting to matrix equations. We also have a short look on the value of unit testing.

2 Introduction

In this project, we will be focusing on eigenvalues. Solving for the eigenvalues of tridiagonal matrices through the Jacobi method. This will be done by numerical programming, using the programming language C++. In the coding part, we will also be focusing on unit testing, making sure the individual functions in the program is accurately calculating the problems.

In the first part of this project, we will be looking at the buckling beam problem. And in the second part of the project, we will be finding the energy levels of two electrons in a three-dimensional harmonic oscillator well using Schrödinger's equation. These will be found as eigenvalues of the matrix for the eigenstates of the electron. This report will be focusing mainly on the mathematics part of the problem and not the physical interpretation.

The report will first explain some mathematical theory of the problem and the algorithm. Second a quick walkthrough of the implementation of the code. Then some results found by the program and lastly a discussion/conclusion of the project and results as a whole.

3 Theory

3.1 Jacobi's rotation algorithm

Jacobi's rotation algorithm or Jacobi's method is an algorithm for diagonalizing a matrix. It was named after the mathematician Carl Gustav Jacob Jacobi. In short, the algorithm goes like this:

- find the max value of the non-diagonal entries
- store the indexes of that value
- perform similarity transform (rotate) on the matrix around the max value (indexes stored)
- repeat until the max value of non-diagonal entries are (approximately) zero

In this project, we have specialized the algorithm to work for symmetric matrices. A rotation thus becomes a unitary transformation. This means that the transformation preserves the orthogonality of the vectors. Let say we have a basis of vectors $\mathbf{v}_i$, that is orthogonal.

$$\mathbf{v}_j^T \mathbf{v}_i = \delta_{ij} \tag{1}$$

and we have

$$\mathbf{w}_i = \mathbf{U}\mathbf{v}_i \tag{2}$$

since $\mathbf{U}$ is a unitary transformation, by the definition of a unitary transformation $\mathbf{U}^T = \mathbf{U}^{-1}$, we can rewrite it as

$$\begin{aligned} \mathbf{w}_j^T \mathbf{w}_i &= (\mathbf{U}\mathbf{v}_j)^T (\mathbf{U}\mathbf{v}_i) \\ (\mathbf{v}_j \mathbf{U}^{-1})(\mathbf{U}\mathbf{v}_i) &= \mathbf{v}_j^T \mathbf{v}_i \end{aligned} \tag{3}$$

and we can then see that the orthogonality is preserved. To see full algorithm Computational Physics lecture slides pg.5 - 10.

3.2 Buckling beam

First, we will look at the buckling beam problem. Consider a horizontal beam on the x-axis. The beam is placed on the axis from $x = 0$ to $x = L$. We then have a force F applied at (L,0) towards the origin. This problem can be model as a classical wave function in one-dimension, with the following ordinary differential equation.

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

here, $u(x)$ is the vertical displacement of the beam in the y direction. The constant γ is defined by the rigidity of the beam. We assume that we know

F and L, thus the eigenvalue problem will allow us to find γ. To make the equation dimensionless, we define a new variable

$$\rho = \frac{x}{L}$$

this gives us $\rho \in [0, 1]$. Now we get the equation

$$\frac{d^2u(\rho)}{d\rho^2} = -\frac{FL^2}{\gamma}u(\rho) = -\lambda u(\rho) \tag{5}$$

With this equation, we get an eigenvalue problem, when discretized. Now we set up the equation for u

$$\ddot{u} = \frac{u(\rho + h) - 2u(\rho) + u(\rho - h)}{h^2} + O(h^2) \tag{6}$$

which we then discretize and get

$$\begin{aligned} -\frac{u(\rho_i + h) - 2u(\rho_i) + u(\rho_i - h)}{h^2} &= \lambda u(\rho_i) \\ -\frac{u_{i+1} - 2u_i + u_{i-1}}{h^2} &= \lambda u_i \end{aligned} \tag{7}$$

Now that we have discretized the equation, we can rewrite it as a matrix equation.

$$\begin{bmatrix} d & a & 0 & 0 & ... & 0 & 0 \\ a & d & a & 0 & ... & 0 & 0 \\ 0 & a & d & a & 0 & ... & 0 \\ ... & ... & ... & ... & ... & ... & ... \\ 0 & ... & ... & ... & a & d & a \\ 0 & ... & ... & ... & ... & a & d \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \\ u_3 \\ ... \\ u_{N-2} \\ u_{N-1} \end{bmatrix} = \lambda \begin{bmatrix} u_1 \\ u_2 \\ u_3 \\ ... \\ u_{N-2} \\ u_{N-1} \end{bmatrix} \tag{8}$$

Here $d = \frac{2}{h^2}$ and $a = -\frac{1}{h^2}$ and we will call our matrix $\mathbf{A}$. And the analytical eigenvalues for this problem can be found from

$$\lambda_j = d + 2cos\Big(\frac{j\pi}{N}\Big), \qquad j = 1, 2, ..., n \tag{9}$$

3.3 Quantum dots in 3 dimensions

We now go over to the quantum mechanical eigenvalue problem.

3.3.1 One electron

In this problem, we look at an electron in a harmonic oscillator potential. We start with the equation

$$-\frac{\hbar^2}{2m}\Big(\frac{1}{r^2}\frac{d^2}{dr}r^2\frac{d}{dr}-\frac{l(l+1)}{r^2}\Big)R(r)+V(r)R(r)=ER(r) \tag{10}$$

Just like in the buckling beam problem, we will make the equation dimensionless. We do this by introducing again a $/rho$. For full calculation see Computational Physics Lecture Slides, pg 12 - 15
We end up with a similar equation as earlier

$$-\frac{d^2u(\rho)}{d\rho^2}+\rho^2u(\rho)=\lambda u(\rho) \tag{11}$$

then discretize as earlier

$$-\frac{u_{i-1}-2u_i+u_{i+1}}{h^2}+\rho_i^2u_i=\lambda u_i \tag{12}$$

and defining

$$\begin{aligned} d_i &= \frac{2}{h^2}+\rho_i^2 \\ e_i &= -\frac{1}{h^2} \end{aligned} \tag{13}$$

Now we can solve the eigenvalue problem using the Jacobi method. The analytical eigenvalues can be found by the equation

$$\frac{2m\alpha^2}{\hbar^2}E \tag{14}$$

and we have the analytical eigenvalues

$$\lambda = 3, 7, 11, 15, ...$$

3.3.2 Two electrons

At last, we look at two electrons. Here we also include the repulsive Coulomb interactions. Again, we make the equation dimensionless. To see full calculation Computational Physics Lecture Slides pg 15 - 17
After making the equation dimensionless, we get

$$-\frac{d^2}{d\rho^2}\psi(\rho)+\omega_r^2\rho^2\psi(\rho)+\frac{1}{\rho}\psi(\rho)=\lambda\psi(\rho) \tag{15}$$

where

$$\lambda = \frac{m\alpha^2}{\hbar^2} E_r \tag{16}$$

and we define

$$\begin{aligned} d_i &= \frac{2}{h^2} + \omega_r^2 \rho_i^2 + \frac{1}{\rho_i} \\ e_i &= -\frac{1}{h^2} \end{aligned} \tag{17}$$

3.4 Implementation

As a quick note on implementation. All programs can be found at:
philhoel/Computational-Physics

For readability and reusability, the code was written in object-oriented design. It also has a separate file for running unit tests. These unit-tests for this project tests if the function *offdiag()* finds the biggest element or not and if the eigenvalues stay the same before and after the jacobi method.

3.4.1 What is object-oriented design?

Object-oriented programming (OOP) is about making objects for easier and safer interaction between different parts of the code. The variables can be stored as class variables, where they will be locked to a specific object. This also means that the value is gotten from within the class and we don't need to be afraid of accidentally making a copy of our variable.

3.4.2 What is a Unit test?

Unit testing is simply making a test for a specific function. The point with unit testing is to test functions individually to make sure that specific function always does it's calculation correctly. This makes it also much easier to debug the code, since you can always be sure that the function that is being tested is working properly as long as the test does not fail.

4 Result

4.1 Eigenvalues for buckling beam

As we can see from the result of the program, the jacobi method is an accurate algorithm. With the same results as armadillo library. That doesn't mean it is always a good algorithm. It is also very slow. For $N = 200$, as was used in the Table 1, the jacobi method ran a total of 75000 iterations. This was run with a tolerance $\epsilon = 10e - 10$

Analytic	Jacobi	eig_sym
9.96834	9.8694	9.8694
39.8709	39.4752	39.4752
89.7003	88.8102	88.8102
159.444	157.862	157.862

Table 1: Comparison of eigenvalues. Analytic, Jacobi method and armadillo's eig_sym, respectively. Here $N = 200$

As we explained in 3.2, the eigenvector of the smallest eigenvalue is supposed to show the shape after the deformation of the buckling beam. This is indeed what we see in Figure 1.

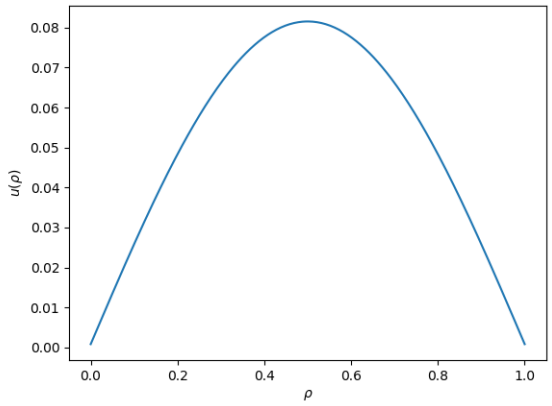

Figure 1: Plot of the eigenvector corresponding to the lowest eigenvalue for the buckling beam

4.2 Eignevalues for one electron

We can see in Table 2 below that the best ρ is 4. Then we can get as close as 2.99991 with $N = 200$, which is a lot faster than the others. Now in this test, we have only varied ρ with 0.5 for each and jumped 100 for N each time. A better pair might be between these values.

	N = 100	N = 200	N = 300
$\rho = 4$	2.99954	2.99991	2.99997
$\rho = 4.5$	2.99938	2.99984	2.99993
$\rho = 5$	2.99923	2.99981	2.99991
$\rho = 5.5$	2.99907	2.99977	2.9999
$\rho = 6$	2.9989	2.99972	2.99988

Table 2: Comparing which pair of ρ and N gives best results.

Table 3: CPU time for Jacobi method

Time table		
N = 100	N = 200	N = 300
0.752873s	11.1564s	57.5929s

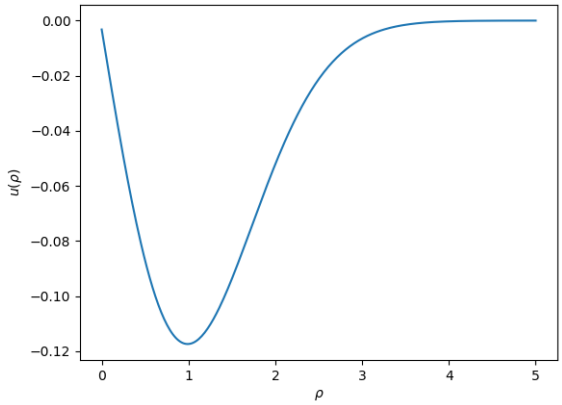

Figure 2: Plot of the eigenvector corresponding to the lowest eigenvalue for the quantum dots problem for one electron

4.3 Eigenvectors for two electrons

In this last one, we have added an ω to the potential. As we can see in Figure 3, the higher value of ω, the steeper the curve becomes.

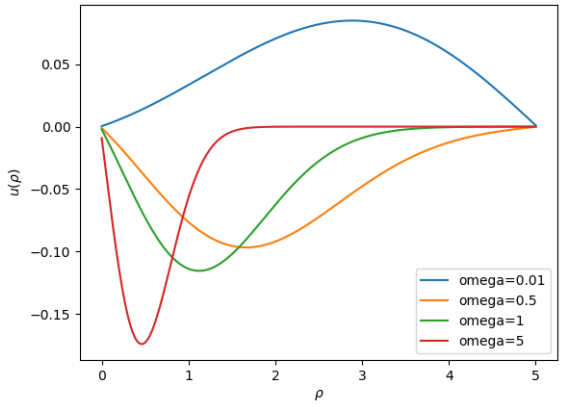

Figure 3: Plot of the eigenvector corresponding to the lowest eigenvalue for the quantum dots problem for two electrons. Each with its own omega value.

5 Conclusion

In this project, we have looked at eigenvalue and eigenvector problems. We have seen how we discretize the equation and rewrite it as an eigenvalue problem. We have also seen how the jacobi rotation algorithm works. We have tested it on three different eigenvalue problems. The buckling beam problem and harmonic oscillator for both one and two electrons and plotted some of the eigenvectors. We have looked at the accuracy of the algorithm, through comparing our results with Armadillo/LAPACK library, as well as the speed of the algorithm. From the results of this project, we can conclude that the jacobi algorithm is an accurate algorithm, but rather ineffective on larger matrices.

6 References

Morten Hjorth-Jensen - Computational Physics: Lecture Notes
David C. Lays, Steven R. Lay, Judi J. McDonald - Linear Algebra and it's Applications