UiO : Department of Physics
University of Oslo

Project 2

FYS3150 - Computational Physics

Anders Johansson

Contents

1   Abstract

All files for this project are available at GitHub[1].

2   Introduction

3   Physical theory

This project deals with the famous Schödinger equation for various systems. A detailed analytical solution of the problems can be found in [1].

---

[1] https://github.com/anjohan/Offentlig/tree/master/FYS3150/Oblig2

3.1   One particle in a harmonic oscillator

The time independent form of the Schrödinger equation for one particle is

$$-\frac{\hbar^2}{2m}\nabla^2\psi(\vec{r}) + V(\vec{r})\psi(\vec{r}) = E\psi(\vec{r}) \tag{1}$$

where V is a potential and ψ is the wave function of the particle. According to Max Born's interpretation, the absolute square of the wave function is a probability distribution, meaning that the probability of finding the particle in a spesific area is equal to the integral of the absolute square of the wave function over that area. One consequence is that if the integral is taken over all of space, the result must be 1.

If the potential is centrosymmetric, one can assume that the wave function is the product of one sentrosymmetric and one angle-dependent function. This leads to a separation of variables, where the angle-dependent equations have the spherical harmonics as solutions. The radial equation is

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

If we look at the states where the orbital quantum number ℓ is 0 and introduce the function $u(r) = rR(r)$, this can be simplified to

$$-\frac{\hbar^2}{2m}\frac{d^2u(r)}{dr^2} + V(r)u(r) = Eu(r)$$

A harmonic oscillator means that the potential is given by $V(r) = \frac{1}{2}kr^2$. Introducing the variable $\rho = r/\alpha$ with $\alpha = \left(\hbar^2/mk\right)^{1/4}$ and defining $\lambda = 2m\alpha^2E/\hbar^2$, we arrive at the final equation

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

It is known[2] that this equation has $\lambda = 3, 7, 11, \ldots$.

3.2   Two particles

For two particles in a harmonic oscillator potential, the Schrödinger equation can be written as

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

Using a similar approach as in the case with one particle, this can be rewritten as

$$-\frac{d^2\psi(\rho)}{d\rho^2} + \omega_r^2\rho^2\psi(\rho) = \lambda\psi(\rho) \tag{3}$$

where ρ is the distance between the particles. There is also an equation for the position of the centre of mass for the system consisting of the two particles, which will not be studied here. If we include the Coulomb interaction between the electrons, the equation instead takes the form

$$-\frac{d^2\psi(\rho)}{d\rho^2} + \omega_r^2\rho^2\psi(\rho) + \frac{1}{\rho} = \lambda\psi(\rho) \tag{4}$$

3.3   Transformation to an eigenvalue problem

Equation (2), equation (3) and equation (4) can all be written on the generic form

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

where $V(\rho)$ is either ρ^2, $\omega_r^2\rho^2$ or $\omega_r^2\rho^2 + 1/\rho$ depending on the situation.

The wave function must be normalized, which requires that the integral of the absolute square of the function over all of space must be 1. Because ρ is a distance and therefore positive, all of space translates to $[0,\infty)$. The particles can not be on top of each other, so we must have $\psi_0 = 0$, while normalizability requires that $\psi_{n+1} = \psi(\infty) = 0$. This gives the boundary conditions of the problem.

As discussed in [3], this equation can be discretized using the three point approximation of the second derivative:

$$\psi_i'' = \frac{\psi_{i+1} + \psi_{i-1} - 2\psi_i}{h^2} + O(h^2)$$

Inserting this into the equation above, we get

$$-\frac{\psi_{i+1} + \psi_{i-1} - 2\psi_i}{h^2} + V_i\psi_i = \lambda\psi_i$$

where $V_i = V(\rho_i)$. We know from the boundary conditions that $\psi_0 = \psi_{n+1} = 0$, so the set of equations for the remaining n points can be written on a matrix form as

$$\begin{bmatrix} \frac{2}{h^2}+V_1 & -\frac{1}{h^2} & 0 & 0 & 0 & \cdots & 0 & 0 \\ -\frac{1}{h^2} & \frac{2}{h^2}+V_2 & -\frac{1}{h^2} & 0 & 0 & \cdots & 0 & 0 \\ 0 & -\frac{1}{h^2} & \frac{2}{h^2}+V_3 & -\frac{1}{h^2} & 0 & \cdots & 0 & 0 \\ \vdots & & & \ddots & & & & \vdots \\ 0 & 0 & \cdots & 0 & -\frac{1}{h^2} & \frac{2}{h^2}+V_{n-2} & -\frac{1}{h^2} & 0 \\ 0 & 0 & \cdots & 0 & 0 & -\frac{1}{h^2} & \frac{2}{h^2}+V_{n-1} & -\frac{1}{h^2} \\ 0 & 0 & \cdots & 0 & 0 & 0 & -\frac{1}{h^2} & \frac{2}{h^2}+V_n \end{bmatrix} \begin{bmatrix} \psi_1 \\ \psi_2 \\ \psi_3 \\ \vdots \\ \psi_{n-2} \\ \psi_{n-1} \\ \psi_n \end{bmatrix} = \lambda \begin{bmatrix} \psi_1 \\ \psi_2 \\ \psi_3 \\ \vdots \\ \psi_{n-2} \\ \psi_{n-1} \\ \psi_n \end{bmatrix}$$

If we define A as the matrix above and $\vec{\psi}$ as $(\psi_1, \psi_2, \ldots, \psi_n)$, we have an eigenvalue problem

$$A\vec{\psi} = \lambda\vec{\psi}$$

4   Mathematical theory

4.1   Orthogonal transformations of vectors

Jacobi's method relies heavily on orthogonal transformations and their properties. Orthogonal transformations are, in $\mathbb{R}^n$, functions on the form

$$T : \vec{x} \mapsto U\vec{x}$$

where $\vec{x}$ is a vector in $\mathbb{R}^n$ and U is a real, orthogonal $n \times n$ matrix.

If U is orthogonal, $U^T U = I$. This implies:

$$T(\vec{v}_i) \cdot T(\vec{v}_j) = (U\vec{v}_i) \cdot (U\vec{v}_j) = (U\vec{v}_i)^T (U\vec{v}_j) = \vec{v}_i^T \overbrace{U^T U}^{I} \vec{v}_j = \vec{v}_i^T \vec{v}_j = \vec{v}_i \cdot \vec{v}_j$$

Orthogonal transformations hence conserve the inner product, and therefore also the orthogonality.

4.2   Orthogonal similarity transformations of matrices

If we have a matrix A with an eigenvector $\vec{x}$ with corresponding eigenvalue λ, then

$$A\vec{x} = \lambda\vec{x}$$

Multiplying with the transpose of an orthogonal matrix S, we get

$$S^T A\vec{x} = \lambda S^T \vec{x}$$

As S is orthogonal, $SS^T = I$, so we can sneak in an identity matrix disguised as SS^T between A and $\vec{x}$ on the left hand side:

$$S^T A S S^T \vec{x} = \lambda S^T \vec{x}$$
$$\left(S^T A S\right)\left(S^T \vec{x}\right) = \lambda \left(S^T \vec{x}\right) \tag{5}$$

This proves that if A is transformed to $S^T A S$, the eigenvalues remain the same, while the new eigenvector is $S^T \vec{x}$.

It is easy to prove that symmetry is conserved by the similarity transformation:

$$\left(S^T A S\right)^T = S^T A^T \left(S^T\right)^T = S^T A^T S = S^T A S$$

if and only if $A^T = A$, i.e. when A is symmetric.

It can also be shown that the Frobenius norm, the square root of the sum of the squares of the matrix elements, is also conserved by the transformation[2].

4.3   Strategy for finding eigenvalues and eigenvectors, using orthogonal similarity transformations

If the transformation matrix S is chosen in a special way such that the largest non-diagonal element of A is transformed to 0, the sum of the squares of the non-diagonal elements decreases from the transformations. This means that if the transformation is applied repeatedly, the matrix is converging towards a diagonal matrix D.

If $D = \text{diag}\{d_1, d_2, \ldots, d_n\}$, the characteristic polynomial is given by

$$\det(\lambda I - D) = (\lambda - d_1) \cdot (\lambda - d_2) \cdot \ldots \cdot (\lambda - d_n)$$

and hence the eigenvalues of D are simply the diagonal elements. The eigenvectors can be chosen as the elements in the standard basis for $\mathbb{R}^n$, i.e. the columns of the identity matrix.

When A is transformed to D, the matrix of eigenvectors, R, is therefore transformed to I. On the other hand, we know from equation (5) on the preceding page that if $\vec{x}$ is an eigenvector of A, then $S^T \vec{x}$ is an eigenvector of the transformed matrix. If A is transformed into D after n transformations with the transformation matrices $S_1, S_2, \ldots, S_n$, then each eigenvector $\vec{x}$ is transformed to $S_n^T S_{n-1}^T \ldots S_1^T \vec{x}$. We must therefore have

$$I = S_n^T S_{n-1}^T \ldots S_1^T R \implies R = S_1 S_2 \ldots S_n I = I S_1 S_2 \ldots S_n \tag{6}$$

The conclusion is that if a suitable transformation matrix can be chosen, we have a method for finding both the eigenvalues and the eigenvectors of any symmetric matrix.

4.4   Choosing the transformation matrix

Jacobi discovered that one could choose a matrix S which is identical to the identity apart from the following elements:

$$s_{kk} = s_{ll} = \cos(\theta), \quad s_{kl} = \sin(\theta), \quad s_{lk} = \sin(\theta)$$

where (k, l) are the indices of the largest non-diagonal element of the matrix to be transformed, and θ is to be determined in such a way that the largest non-diagonal elements are transformed to 0.

With $B = S^T A S$, all elements apart from those in row or column k and l are left unchanged. The other elements are transformed as follows:

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

Additionally, we know from equation (6) that the new eigenvector matrix is transformed by multiplying by S from the right hand side, which yields the matrix $\tilde{R}$. All elements are left unchanged, apart from

$$
\begin{aligned}
\tilde{r}_{ik} &= r_{ik}\cos(\theta) - r_{il}\sin(\theta) \\
\tilde{r}_{il} &= r_{ik}\sin(\theta) + r_{il}\cos(\theta)
\end{aligned}
$$

We must now choose θ such that a_{kl}, the largest non-diagonal element, is transformed to 0. This means solving the equation $b_{kl} = 0$:

$$0 = b_{kl} = (a_{kk} - a_{ll})\cos(\theta)\sin(\theta) + a_{kl}\left(\cos^2(\theta) - \sin^2(\theta)\right)$$

This leads to

$$(a_{ll} - a_{kk})\cos(\theta)\sin(\theta) = a_{kl}\left(\cos^2(\theta) - \sin^2(\theta)\right)$$
$$\frac{a_{ll} - a_{kk}}{a_{kl}} = \frac{\cos^2(\theta) - \sin^2(\theta)}{\cos(\theta)\sin(\theta)}$$

Defining the left hand side as 2τ and dividing by $\cos^2(\theta)$ in the numerator and denominator on the right hand side, we get

$$2\tau = \frac{1 - \tan^2(\theta)}{\tan(\theta)}$$
$$0 = \tan^2(\theta) - 2\tau\tan(\theta) - 1$$
$$\tan(\theta) = -\tau \pm \sqrt{\tau^2 + 1}$$

Either sign in front of the square root can in principle be chosen. Numerically, however, there is a difference. The point of the algorithm is to get the non-diagonal elements, of which a_{kl} is the largest, as small as possible, which means τ will grow larger for every iteration. When τ is large, $\sqrt{\tau^2 + 1} \approx |\tau|$.

If τ is positive and the + sign is chosen (or vice versa), a large, negative number and a large, positive number are added and give approximately 0. If for example the computer is able to store 8 digits in the mantissa and the result is 6 orders of magnitudes smaller, the result will only have 2 accurate digits.

The conclusion is that if τ is negative, the + sign should be chosen, while if τ is positive, the − sign should be chosen.

$\cos(\theta)$ and $\sin(\theta)$ can then be calculated from the formulas

$$\cos(\theta) = \frac{1}{\sqrt{1 + \tan^2(\theta)}} \qquad \sin(\theta) = \tan(\theta)\cos(\theta)$$

4.5 Implementation of the algorithm

The complete function is available on GitHub[2].

Code snippet 1: Implementation of Jacobi's method in C++.

```cpp
    max = largest_nondiagonal_symmetric(A,n,kl);
    while(iterations <= max_iterations && max > tolerance) {
        k = kl[0]; l = kl[1];
        a_kk = A[k][k]; a_ll = A[l][l]; a_kl = A[k][l];
        tau = (a_ll-a_kk)/(2*a_kl);
        tangens = -tau - (tau>0 ? -1 : 1)*sqrt(1 + tau*tau);
        cosinus = 1/sqrt(1 + tangens*tangens);
        sinus = tangens*cosinus;

        A[k][k] = a_kk*cosinus*cosinus - 2*a_kl*cosinus*sinus\
                + a_ll*sinus*sinus;
        A[l][l] = a_ll*cosinus*cosinus + 2*a_kl*cosinus*sinus\
                + a_kk*sinus*sinus;
        A[k][l] = 0;
        A[l][k] = 0;
        for(i=0; i<n; i++){
            if(i!=k && i!=l){
                a_ik = A[i][k]; a_il = A[i][l];
                A[i][k] = a_ik*cosinus - a_il*sinus;
                A[i][l] = a_il*cosinus + a_ik*sinus;
                A[k][i] = A[i][k]; A[l][i] = A[i][l]; //Symmetric
            }
            r_ik = R[i][k]; r_il = R[i][l];
            R[i][k] = r_ik*cosinus - r_il*sinus;
            R[i][l] = r_ik*sinus + r_il*cosinus;
        }

        iterations++;
        max = largest_nondiagonal_symmetric(A,n,kl);
    }
```

---

[2]https://github.com/anjohan/Offentlig/blob/master/FYS3150/Oblig2/jacobi.cpp

5   Results and discussion

5.1   Simulation for one particle
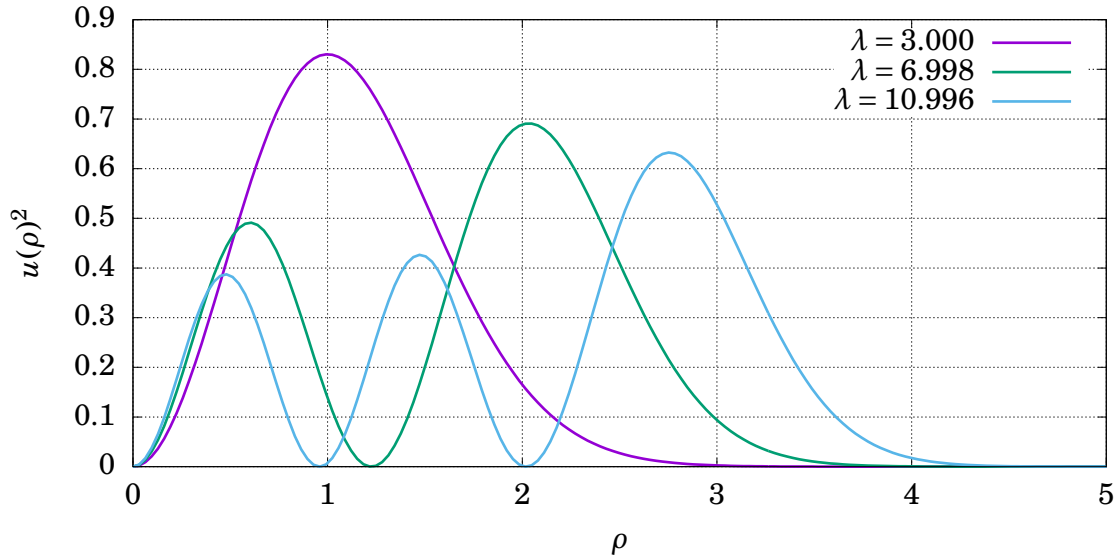
Figure 1: Simulation of one particle in a harmonic oscillator potential for $n = 250$ with $\rho_{\max} = 8$.

n	λ_0	λ_1	λ_2	Number of iterations
50	2.992	6.960	10.902	2863
100	2.998	6.990	10.976	12307
150	2.999	6.996	10.989	28353
200	2.999	6.997	10.994	51095
250	3.000	6.998	10.996	80474

Table 1: The three lowest eigenvalues found by the algorithm for the different number of mesh points n. The analytical values are $\lambda_0 = 3$, $\lambda_1 = 7$ and $\lambda_2 = 11$.

The number of iterations is approximately quadrupled when the number of mesh points is doubled, indicating that the numbers of iterations runs as n^2. This means that the number of iterations is proportional to the number of elements in the matrix. However, the number of non-diagonal elements in the tridiagonal elements handled in this project is proportional to n, so this trend is unlikely to hold for dense matrices.
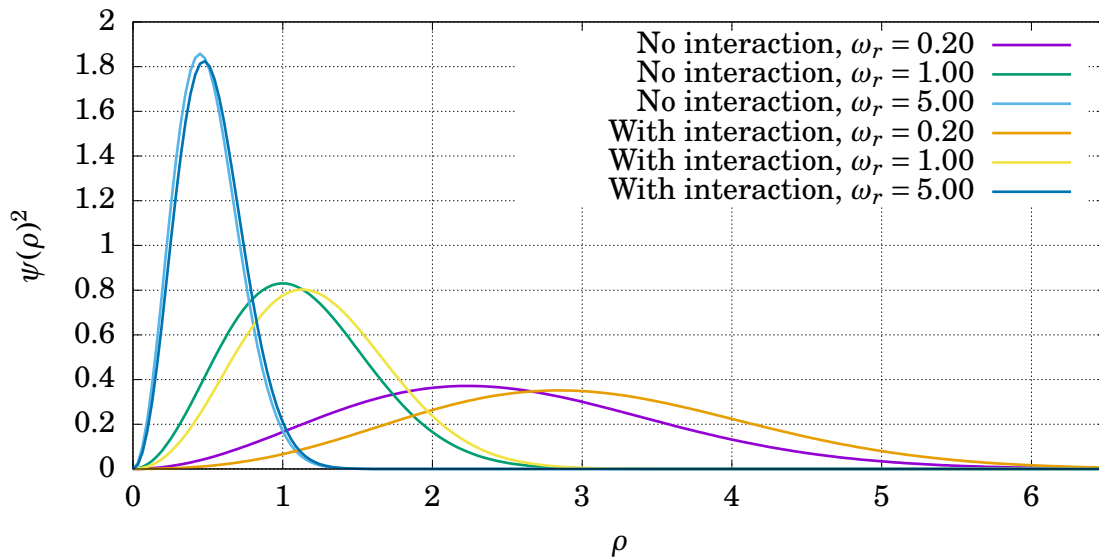
5.2   Simulation for two particles

Figure 2: Simulation of two particles, with and without Coulomb interaction, for two different values of ω_r. A larger ω_r corresponds to a stronger harmonic oscillator.

Without interaction and with $\omega_r = 1$, equation (2) and equation (3), become identical, and we see from figure 1 on the preceding page and figure 2 that the algorithm finds the same eigenstates.

The strength of the harmonic oscillator is represented by ω_r. When ω_r is large, it is therefore expected that the particles are pulled closer to origo so that ρ decreases, and that their position is more sharply determined. This fits very well with the figure.

Because the interaction between two electrons is repulsive, the particles are further apart when the interaction is taken into account. The effect of this is greater when ω_r is smaller and the particles are more free to move around, as the Coulomb interaction is a far reaching force.

6   Tests

The code is found at GitHub[3].

---

[3]https://github.com/anjohan/Offentlig/blob/master/FYS3150/Oblig2/test.cpp

6.1 Ability to find eigenvalues and eigenvectors

The matrix

$$A = \begin{bmatrix} 7 & -2 & 0 \\ -2 & 6 & -2 \\ 0 & -2 & 5 \end{bmatrix}$$

has the pretty eigenvalues 3, 6 and 9, with corresponding eigenvectors

$$\begin{bmatrix} 1 \\ 2 \\ 2 \end{bmatrix}, \quad \begin{bmatrix} -2 \\ 1 \\ 2 \end{bmatrix}, \quad \begin{bmatrix} 2 \\ -2 \\ 1 \end{bmatrix}$$

The implementation of Jacobi's method can be tested on this matrix using

Code snippet 2: Test of eigenvalues.

```
A[0][0] = 7; A[0][1] = -2; A[0][2] = 0;
A[1][0] = -2; A[1][1] = 6; A[1][2] = -2;
A[2][0] = 0; A[2][1] = -2; A[2][2] = 5;

iterasjoner = jacobi(A,R,n, 1E-12);
```

where A and R are 3×3 pointer matrices and n is 3. Formatted printing of the resulting A and R yields

```
 9.000   0.000   0.000
 0.000   3.000   0.000
 0.000   0.000   6.000
------------------------
 0.667   0.333  -0.667
-0.667   0.667  -0.333
 0.333   0.667   0.667
```

This is the expected result, except that the program returns normalized eigenvectors. The second printed matrix has columns of eigenvectors corresponding to different eigenvalues for a diagonal matrix, so the columns should be orthogonal. Simple arithmetic shows that this is correct.

6.2 Ability to find largest non-diagonal elements

A simple test is

Code snippet 3: Test of search for largest non-diagonal element.

```
A[0][0] = 2;
A[1][0] = 4; A[1][1] = 12;
A[2][0] = -2; A[2][1] = -5; A[2][2] = 1;
A[3][0] = 1; A[3][1] = 0; A[3][2] = 4; A[3][3] = 2;
A[4][0] = 1; A[4][1] = 0; A[4][2] = 4; A[4][3] = 2; A[4][4] = -7;
```

```
    largest_nondiagonal_symmetric(A,5,kl);
    fprintf(file,"k=%d, l=%d",kl[0],kl[1]);
```

which yields

```
k=2, l=1
```

This is correct, as the function searches for the element with the largest absolute value. It is not necessary to specify the upper triangle of the matrix, as the function is specialized for symmetric matrices and therefore only searches the lower triangle.

References

[1] David Griffiths. *Introduction to quantum mechanics*. Upper Saddle River, NJ: Pearson Prentice Hall, 2005. ISBN: 978-0131118928.

[2] Morten Hjorth-Jensen. *Computational Physics*. Lecture notes. 2015. URL: https://github.com/CompPhysics/ComputationalPhysics/blob/master/doc/Lectures/lectures2015.pdf.

[3] Anders Johansson. "Project 1". In: *FYS3150, Computational Physics* (Sept. 2016), pp. 4–6. URL: https://github.com/anjohan/Offentlig/blob/master/FYS3150/Oblig1/Johansson_Anders_FYS3150_Oblig1.pdf.