Cool Title

Project 2, FYS-3150

Ina K. B. Kullmann

October 28, 2015

All source codes can be found at: `https://github.com/inakbk/Project_2`.

Contents

Abstract

bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla blav bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla blav bla bla bla bla blav bla bla bla bla bla

bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla blav bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla bla blav bla bla bla bla blav bla bla bla bla bla

1 Motivation and purpose

The aim of this project is to solve Schrödinger's equation for two electrons in a three-dimensional harmonic oscillator well with and without a repulsive Coulomb interaction. Your task is to solve this equation by reformulating it in a discretized form as an eigenvalue equation to be solved with Jacobi's method. −>code which implements Jacobi's method. −> plot the wave function for two electrons. first look at one electron, then two without coul and then with coul.

Electrons confined in small areas in semiconductors, so-called quantum dots, form a hot research area in modern solid-state physics, with applications spanning from such diverse fields as quantum nano-medicine to the contruction of quantum gates. You can read about quantum dots at `http://en.wikipedia.org/wiki/Quantum_dot`, which also contains links to several scientific articles. A recent article of interest is the review by Semonin *et al* in Materials Today, volume 15, page 508 (2012).

In this article we will let two electrons move in a three-dimensional harmonic oscillator potential that repel each other via the Coulomb interaction. Throughout this paper we will assume spherically symmetry and let the angular momentum be $l = 0$. Before solving the problem for two electrons we will look at one electron in a harmonic oscillator potential.

To solve the Schrödinger's equation for one electron(and two??) we will have to transformed the equations into a matrix eigenvalue problem. When we have rewritten the problem we will use Jacobi's rotation algorithm to find the solutions.

Together with linear equations and least squares, the third major problem in matrix computations deals with the algebraic eigenvalue problem. Here we limit our attention to the symmetric case. We focus in particular similarity transformations, −> Jacobi [1]

[1] sitert fra lectures2015 p. 225

Part I
Theory

2 Numerical method

2.1 Solving eigenvalueproblems using similarity transformations

Let us assume that we have the eigenvalue problem

$$\mathbf{A}\mathbf{x}^{(\mathbf{v})} = \lambda \mathbf{x}^{(\mathbf{v})}$$

where $\lambda^{(v)}$ are the eigenvalues and $\mathbf{x}^{(\mathbf{v})}$ the corresponding eigenvectors.[2] Assuming that the matrix $\mathbf{A}$ is real and symetric we can use the Jacobi rotation algoritm to solve the eigenvalue problem. First we will describe the use of similarity tansformations in general and then the Jacobi rotation algorithm in detail.

Let $\mathbf{D}$ be the diagonal matrix with the eigenvalues of $\mathbf{A}$ on the diagonal:

$$\mathbf{D} = \begin{pmatrix} \lambda_1 & 0 & 0 & 0 & \dots & 0 & 0 \\ 0 & \lambda_2 & 0 & 0 & \dots & 0 & 0 \\ 0 & 0 & \lambda_3 & 0 & 0 & \dots & 0 \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ 0 & \dots & \dots & \dots & \dots & \lambda_{n-1} & 0 \\ 0 & \dots & \dots & \dots & \dots & 0 & \lambda_n \end{pmatrix} \tag{1}$$

We say that a matrix $\mathbf{B}$ is a similarity transform of $\mathbf{A}$ if

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

Where $\mathbf{S}$ is a unitary matrix so $\mathbf{S}^{\mathbf{T}}\mathbf{S} = \mathbf{S}^{-\mathbf{1}}\mathbf{S} = \mathbf{I}$. The point of this is that the matrix $\mathbf{B}$ has the same eigenvalues as $\mathbf{A}$. Since $\mathbf{A}$ is real and symetric there exists a real orthonogal matrix $\mathbf{S}'$ such that:

$$\mathbf{S}'^{\mathbf{T}}\mathbf{A}\mathbf{S}' = \mathbf{D}$$

So the strategy is then to perform a series of similarity transformations on the original matrix $\mathbf{A}$ so that the matrix reduces to the diagonal matrix $\mathbf{D}$ with the eigenvalues on the diagonal:

$$\mathbf{S}_{\mathbf{N}}^{\mathbf{T}} \dots \mathbf{S}_{\mathbf{1}}^{\mathbf{T}}\mathbf{A}\mathbf{S}_{\mathbf{1}} \dots \mathbf{S}_{\mathbf{N}} = \mathbf{D}$$

where $\mathbf{S_1} \dots \mathbf{S_N} = \mathbf{S}'$. This must be done on both sides of the equation. Only one transformation is given by:

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

using that $\mathbf{S}$ is unitary. We see that the eigenvalue of $\mathbf{B}$ is the same as for $\mathbf{A}$, but the eigenvector is changed to $\mathbf{S}^{\mathbf{T}}\mathbf{x}$.

[2] The introduction to this section is based on the lecturenotes"lectures2015.pdf"

2.2 Jacobi's rotation algorithm

We will now see how we can find the eigenvalues of a real and symetric matrix $\mathbf{A}$ by using the Jacobi's rotation algoritm, or Jacobi's method.

The Jacobi's method introduces the $(n \times n)$ unitary orthogonal transformation matrix

$$\mathbf{S} = \begin{pmatrix} 1 & 0 & 0 & 0 & \dots & 0 & 0 \\ 0 & 1 & 0 & 0 & \dots & 0 & 0 \\ 0 & 0 & \cos\theta & 0 & 0 & \dots & \sin\theta \\ 0 & 0 & 0 & 1 & 0 & \dots & 0 \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ 0 & \dots & \dots & \dots & \dots & 1 & 0 \\ 0 & \dots & -\sin\theta & \dots & \dots & 0 & \cos\theta \end{pmatrix} \tag{2}$$

that performs a plane rotation around an angle θ in the Euclidean n-dimensional space. For simpler notation we define the quantities $\tan\theta = t = s/c$, with $s = \sin\theta$ and $c = \cos\theta$. The elements of the matrix $\mathbf{S}$ that differ from zero is given by the indexes k, l:

$$s_{kk} = s_{ll} = c, \quad s_{kl} = -s_{lk} = -s, \quad s_{ii} = 1$$

where $i \neq k$, $i \neq l$ and i, j are the indexes of the matrix. Then the similarity transformation

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

can be written on component form as:

$$\begin{aligned} b_{ii} &= a_{ii} \\ b_{ik} &= ca_{ik} - sa_{il} \\ b_{il} &= ca_{il} - sa_{ik} \\ b_{kk} &= c^2 a_{kk} - scsa_{kl} + s^2 a_{ll} \\ b_{ll} &= c^2 a_{ll} + 2csa_{kl} + s^2 a_{kk} \\ b_{kl} &= cs(a_{kk} - a_{ll}) + (c^2 - s^2)a_{kl} \end{aligned}$$

We want to choose the angle θ so that all the non-diagonal matric elements becomes zero, that is $b_{kl} = 0$. Introducing a new variable τ dependent on k, l:

$$\begin{aligned} b_{kl} &= cs(a_{kk} - a_{ll}) + (c^2 - s^2)a_{kl} = 0 \\ &\Rightarrow \frac{a_{ll} - a_{kk}}{a_{kl}} = \frac{c^2 - s^2}{cs} = \frac{2\cos 2\theta}{\sin 2\theta} = 2\cot 2\theta \\ &\Rightarrow \tau = \cot 2\theta = \frac{a_{ll} - a_{kk}}{2a_{kl}} \end{aligned}$$

Then $b_{kl} = 0$ can be written as a second order equation

$$\begin{aligned} 2\tau = \frac{c^2 - s^2}{cs} &= \frac{c^2}{cs} - \frac{s^2}{cs} = \frac{1}{t} - t \\ &\Rightarrow 2\tau t = 1 - t^2 \\ &\Rightarrow t^2 + 2\tau t - 1 = 0 \end{aligned}$$

with the solution

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

We obtain c ans s from

$$\begin{aligned} c^2 + s^2 &= 1 \\ \Rightarrow c &= \frac{1}{\sqrt{1+t^2}}, \end{aligned}$$

and using that $s = tc$.

But which of the roots of t should we choose? To make shure that the the similarity transformation does not make bigger changes to the other elements of $\mathbf{A}$ while making another element zero we minimize the difference between the matrices $\mathbf{B}$ and $\mathbf{A}$ by letting $|\theta| \leq \pi/4$. This also makes shure that the iterations goes faster toward the solution, else the iterations might not converge at all after a reasonable number of iterations.

Separating the inequality gives $\theta \leq \pi/4$ and $\theta \geq -\pi/4$. Starting with the first inequality:

$$\begin{aligned} \theta = \arctan t &\leq \pi/4 \\ t &\leq \tan(\pi/4) = 1 \end{aligned}$$

and

$$\begin{aligned} \theta = \arctan t &\geq -\pi/4 \\ t &\geq \tan(-\pi/4) = -1 \end{aligned}$$

so we see that $|t| \leq 1 \Rightarrow -\tau \pm \sqrt{1+\tau^2} \leq 1$. If we look at the case where $\tau > 0$ and try the positive root of t we see that:

$$\begin{aligned} -\tau + \sqrt{1+\tau^2} &\leq -|\tau| + 1 + |\tau| = 1 \\ &\Rightarrow t \leq 1 \Rightarrow \theta \leq \pi/4 \end{aligned}$$

must be satisfied.

If we then try the case when $\tau < 0$ and the negative root of t we see that:

$$\begin{aligned} -\tau - \sqrt{1+\tau^2} = |\tau| - \sqrt{1+\tau^2} &\geq |\tau| - (1 + |\tau|) = -1 \\ &\Rightarrow t \geq -1 \Rightarrow \theta \geq -\pi/4 \end{aligned}$$

must be satisfied. So the choice of the root of t is dependent on τ. When $\tau > 0$ we choose the positive root, and when $\tau < 0$ we choose the negative root to make shure that $|\theta| \leq \pi/4$ is satisfied.

The Jacobi algorithm can then be described as follows:

1. Find the indexes k, l of the maximum element of the matrix $\mathbf{A}$.
2. Obtain c and s, the matrix elements of $\mathbf{S}$, given by k, l.
3. Calculate the similarity transformation $\mathbf{B} = \mathbf{S^T A S}$.

4. Start on [1.] again, setting $\mathbf{A} = \mathbf{B}$, untill all off-diagonal elements are esentially zero (less than a given threshold). When the iterations stop the eigenvalues are given by the matrix $\mathbf{D} = \mathbf{B}$.

We see that all information needed to perform a Jacobi rotation is given by the indexes k, l of the maximum elements of $\mathbf{A}$.

Retriving the eigenvectors

In this particular problem we also want to find the eigenvectors $\mathbf{x}^{(\mathbf{v})}$ of the matrix $\mathbf{A}$. The Jacobi's rotation method described in section 2.2 does not return the eigenvectors. For every similarity transformation in the Jacobi's method the eigenvector is changed from $\mathbf{x}$ to $\mathbf{S^T x}$. So it is possible to add a routine to the Jacobi's algorithm which ceeps track of the changes to the eigenvectors so that the eigenvectors can be returned when the iterations stop. This is a bit tricy and the eigenvectors also need to be normalized.

We will use the existing *tqli* function found at(??). Here the eigenvectors are obtained from the matrix $z[i][j]$, where the index j refers to eigenvalue j. The index i points to the value of the wave function in position ρ_j. That is, $u^{(\lambda_j)}(\rho_i) = z[i][j]$.

The eigenvectors are normalized.

Wil also compare with (?) Finally, the armadillo function *eig_sys* can be used to find eigenvalues and eigenvectors.

2.3 Implementation and testing of the Jacobi algorithm

The Jacobi rotation algorithm is structured into separate files which can be run from `main.cpp`. All files refered to in this section can be found in the folder `project_2_test_jacobi` at:
`https://github.com/inakbk/Project_2/tree/master/project_2_test_jacobi`

The `jacobi.h` file contains all the routines for one rotation. The code finds the maximum element of the matrix, obtains the transformation matrix and does one rotation. The `jacobisolver.h` file does all the rotations for a given matrix $\mathbf{B}$ by calling the `jacobi.h` code and returns the eigenvalues and the first eigenvector. The parameter `tolerance` sets the limit of how small all the off diagonal elements in the matrix $\mathbf{B}$ needs to be before the rotations stop and the eigenvalues and the first eigenvector are returned. If the number of rotations (`numberOfIterations`) exceeds the maximum number of rotations (`maxNumberOfIterations`) before the off diagonal elements are smaller than the tolerance, the opreations are aborted with a warning message that the routine did not converge. The code also computes the execution time of all iterations for one matrix.

We then see that the code in `jacobisolver.h` does the whole job of solving the eigenvalue problem with the Jacobi method and need only be given the matrix $\mathbf{B}$ with its dimension and the maximum number of iterations. Then the `main.cpp` program only needs to constructs the matrix $\mathbf{B}$ and call the `jacobisolver.h` code. With the help of `writetofile.h` the main program will write all parameters and the eigenvalues and eigenvectors to a file with filenames that distinguish the parameters that where run. These datafiles will then later be read by a python script and plot the data.

To test the implementation of the Jacobi algorithm the matrix $\mathbf{B}$ is initialized to a random symetric matrix. This is done by taking the matrix $\mathbf{C}$ as a random matrix and let $\mathbf{B} = \mathbf{CC^T}$. Then $\mathbf{B}$ is a symetric matrix:

$$\mathbf{B^T} = (\mathbf{CC^T})^\mathbf{T} = (\mathbf{C^T})^\mathbf{T}\mathbf{C^T} = \mathbf{CC^T} = \mathbf{B}$$

using that $(\mathbf{ab})^{\mathbf{T}} = \mathbf{b}^{\mathbf{T}}\mathbf{a}$[3].

To check that the implementation of the Jacobi rotation algorithm is correct the solution of the symetric matrix $\mathbf{B}$ is compared to the solutions with the `eig_sym` function in the armadillo library.

3 A simple system: One electron in a three-dimensional harmonic oscillator potential.

here we will look eqiation (?)

A system with known solutions, good for testing

3.1 Rewriting the Schrödinger's equation to a dimensionless form

We look at the Schödinger's equation for one electron at a radius $r \in [0, \infty)$ in a harmonic oscillator potential given by:

$$V(r) = (1/2)kr^2$$

where $k = m\omega^2$ and ω is the oscillator frequence. The energy E of the harmonic oscillator in three dimensions is given by:

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

with $n = 0, 1, 2, \ldots$ and $l = 0, 1, 2, \ldots$ is the orbital momentum of the electron.

We are only interested in the solution of the radial part of the Schrödinger equation given by

$$-\frac{\hbar^2}{2m}\left(\frac{1}{r^2}\frac{d}{dr}r^2\frac{d}{dr} - \frac{l(l+1)}{r^2}\right)R(r) + V(r)R(r) = ER(r). \tag{3}$$

We then use the substitution $R(r) = (1/r)u(r)$ and obtain

$$-\frac{\hbar^2}{2m}\frac{d^2}{dr^2}u(r) + \left(V(r) + \frac{l(l+1)}{r^2}\frac{\hbar^2}{2m}\right)u(r) = Eu(r).$$

The boundary conditions are $u(0) = 0$ and $u(\infty) = 0$.

We will modify equation 3 further by introducing a dimensionless variable $\rho = (1/\alpha)r$ where α is a constant with dimension length and get

$$-\frac{\hbar^2}{2m\alpha^2}\frac{d^2}{d\rho^2}u(\rho) + \left(V(\rho) + \frac{l(l+1)}{\rho^2}\frac{\hbar^2}{2m\alpha^2}\right)u(\rho) = Eu(\rho).$$

Now inserting $l = 0$ and the rewritten potential $V(\rho) = (1/2)k\alpha^2\rho^2$ we obtain:

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

and then multiply by $2m\alpha^2/\hbar^2$ on both sides

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

[3] `https://en.wikipedia.org/wiki/Transpose` (I dont have the mat1120 book here so wikipedia refrences it is then..)

We fix the constant α so that

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

and then define

$$\lambda = \frac{2m\alpha^2}{\hbar^2}E,$$

Finally equation 3 can be written as

$$-\frac{d^2}{d\rho^2}u(\rho) + \rho^2 u(\rho) = \lambda u(\rho). \tag{4}$$

which is the equation we want to solve numerically. We know[4] that this equation has the eigenvalues $\lambda_0 = 3, \lambda_1 = 7, \lambda_2 = 11, \ldots$. for $l = 0$. These analytical values will be used to test the precition of the numerical method.

3.2 Discretizing the Schrödingers equation and solving the equation numerically.

We will now rewrite the Schrödingers equation on a discretized form to be able to solve it as an matrix eigenvalue problem.

We use the expression for the second derivative of the function $u(\rho)$

$$u'' = \frac{u(\rho + h) - 2u(\rho) + u(\rho - h)}{h^2} + O(h^2), \tag{5}$$

where h is our step. We define the minimum and maximum values for the variable ρ, $\rho_{\min} = 0$ and $\rho_{\max}$ so that:

$$h = \frac{\rho_{\max} - \rho_{\min}}{n_{\text{step}}}.$$

where n_{step} is a given number of steps. Since $\rho \propto r$ and $r \in [0, \infty)$ the maximum value of ρ should be $\rho_{\max} = \infty$. But we cannot set a infinite value when computing the solution numerically. We will therefore have to choose a significally large enough ρ.

We can then define an arbitrary value of ρ as

$$\rho_i = \rho_{\min} + ih \qquad i = 0, 1, 2, \ldots, n_{\text{step}}$$

so that the Schrödinger equation for ρ_i reads

$$-\frac{u(\rho_i + h) - 2u(\rho_i) + u(\rho_i - h)}{h^2} + \rho_i^2 u(\rho_i) = \lambda u(\rho_i),$$

or in a more compact way with the harmonic oscillator potential $V_i = \rho_i^2$:

$$-\frac{u_{i+1} - 2u_i + u_{i-1}}{h^2} + V_i u_i = \lambda u_i, \tag{6}$$

Then we can define first the diagonal matrix element

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

[4]This is given in the assignment text for Project 2, FYS3150..

and the non-diagonal matrix element:

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

We observe that in this case all the non-diagonal matrix element is given by a constant, so we can denote them all as $e = -\frac{1}{h^2}$ instead. (insert e og ikke ei hele veien??)

With these definitions the Schrödinger equation (4) takes the following form

$$d_i u_i + e_{i-1} u_{i-1} + e_{i+1} u_{i+1} = \lambda u_i, \tag{7}$$

where u_i is unknown. This can be written as a matrix eigenvalue problem

$$\begin{pmatrix} d_1 & e_1 & 0 & 0 & \dots & 0 & 0 \\ e_1 & d_2 & e_2 & 0 & \dots & 0 & 0 \\ 0 & e_2 & d_3 & e_3 & 0 & \dots & 0 \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ 0 & \dots & \dots & \dots & \dots & d_{n_{\text{step}}-2} & e_{n_{\text{step}}-1} \\ 0 & \dots & \dots & \dots & \dots & e_{n_{\text{step}}-1} & d_{n_{\text{step}}-1} \end{pmatrix} \begin{pmatrix} u_1 \\ u_2 \\ \dots \\ \dots \\ \dots \\ u_{n_{\text{step}}-1} \end{pmatrix} = \lambda \begin{pmatrix} u_1 \\ u_2 \\ \dots \\ \dots \\ \dots \\ u_{n_{\text{step}}-1} \end{pmatrix} \tag{8}$$

We will use the analytical eigenvalues to test the implementation of the problem for this one electron system, but we are also interested in the eigenvector of the ground state so that we can plot the probability distrubution.

3.2.1 Using the Jacobi method and finding reasonable values for $p_{\max}$ and n_{step}

We can solve equation 8 numerically using the Jacobi rotation algorithm as described in section 2.3 by using the files `jacobisolver.h` and `jacobi.h` and `writetofile.h` to save the results. But in `main.cpp` the matrix would have to be initialized to the matrix in equation 8:

$$\mathbf{B} = \begin{pmatrix} d_1 & e & 0 & 0 & \dots & 0 & 0 \\ e & d_2 & e & 0 & \dots & 0 & 0 \\ 0 & e & d_3 & e & 0 & \dots & 0 \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ 0 & \dots & \dots & \dots & \dots & d_{n_{\text{step}}-2} & e \\ 0 & \dots & \dots & \dots & \dots & e & d_{n_{\text{step}}-1} \end{pmatrix}$$

This matrix is dependent on the elements d_i and e which is only dependent on p_i, or $p_{\max}$, and the dimension of the matrix n_{step}. So we need to test the algorithm so that we know which $p_{\max}$ that gives stable results and which values of n_{step} it is reasonable to use.

Results? Eller annen seksjon?

How many points n_{step} do you need in order to get the lowest three eigenvalues with four leading digits? Remember to check the eigenvalues for the dependency on the choice of $\rho_{\max}$.

How many similarity transformations are needed before you reach a result where all non-diagonal matrix elements are essentially zero? Try to estimate the number of transformations and extract a behavior as function of the dimensionality of the matrix.

You can check your results against the code based on Householder's algorithm, *tqli* in the file lib.cpp. Alternatively, you can use the Armadillo function for solving eigenvalue problems.

Comment your results (here you could for example compute the time needed for both algorithms for a given dimensionality of the matrix).

---

4 A complex system: Two electrons in a three-dimensional harmonic oscillator potential with Coulomb interaction.

We will now study two electrons in a harmonic oscillator well which also interact via a repulsive Coulomb interaction.

We will now rewrite the single-electron equation as

$$-\frac{\hbar^2}{2m}\frac{d^2}{dr^2}u(r) + \frac{1}{2}kr^2u(r) = E^{(1)}u(r),$$

where $E^{(1)}$ stands for the energy with one electron only.

look at with and without Coulom again we only interested in the ground state with $l = 0$.

4.1 Without Coulomb interaction

The Schrödingers equation for two electrons in a harmonic oscillator potential with no Coulomb interaction can be written as

$$\left(-\frac{\hbar^2}{2m}\frac{d^2}{dr_1^2} - \frac{\hbar^2}{2m}\frac{d^2}{dr_2^2} + \frac{1}{2}kr_1^2 + \frac{1}{2}kr_2^2\right)u(r_1, r_2) = E^{(2)}u(r_1, r_2) \tag{9}$$

with a two-electron wave function $u(r_1, r_2)$ and the two-electron energy $E^{(2)}$.

When there is no electron interaction the equation 9 can be written as the product of two single-electron wave functions, the equation is separabel. We introduce the relative coordinate $\mathbf{r} = \mathbf{r}_1 - \mathbf{r}_2$ and the center-of-mass coordinate $\mathbf{R} = 1/2(\mathbf{r}_1 + \mathbf{r}_2)$. With these new coordinates, the equation 9 reads

$$\left(-\frac{\hbar^2}{m}\frac{d^2}{dr^2} - \frac{\hbar^2}{4m}\frac{d^2}{dR^2} + \frac{1}{4}kr^2 + kR^2\right)u(r, R) = E^{(2)}u(r, R).$$

The equations for r and R can be separated as the wave function $u(r, R) = \psi(r)\phi(R)$ and the energy is given by the sum of the relative energy E_r and the center-of-mass energy E_R, that is

$$E^{(2)} = E_r + E_R.$$

For this article we will omit the center-of-mass energy in the calculations.

With no repulsive Coulomb interaction you should get a result which corresponds to the relative energy of a non-interacting system. Make sure your results are stable as functions of $\rho_{\max}$ and the number of steps.

4.2 With Coulomb interaction

We will now include the repulsive Coulomb interaction between two electrons. Introducing the term

$$V(r_1, r_2) = \frac{\beta e^2}{|\mathbf{r}_1 - \mathbf{r}_2|} = \frac{\beta e^2}{r},$$

with $\beta e^2 = 1.44$ eVnm. With this extra term representing the interaction the r-dependent Schrödinger equation becomes

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

We want to manipulate this equation further to make it as similar as possible to the one electron equation (3) we obtained earlier. Again we introduce the dimensionless variable $\rho = r/\alpha$ and repeating the same steps as in section 3.1, we arrive at

$$-\frac{d^2}{d\rho^2}\psi(\rho) + \frac{1}{4}\frac{mk}{\hbar^2}\alpha^4\rho^2\psi(\rho) + \frac{m\alpha\beta e^2}{\rho\hbar^2}\psi(\rho) = \frac{m\alpha^2}{\hbar^2}E_r\psi(\rho).$$

We manipulate this equation further by defining a 'frequency' which reflects the strength of the oscillator potential

$$\omega_r^2 = \frac{1}{4}\frac{mk}{\hbar^2}\alpha^4,$$

and fix the constant α by requiring

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

or

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

Then defining

$$\lambda = \frac{m\alpha^2}{\hbar^2}E,$$

so we finally can write the Schrödinger's equation as

$$-\frac{d^2}{d\rho^2}\psi(\rho) + \omega_r^2\rho^2\psi(\rho) + \frac{1}{\rho} = \lambda\psi(\rho). \tag{10}$$

For specific oscillator frequencies, the above equation has answers in an analytical form found by M. Taut[5]. We will study the cases $\omega_r = 0.01$, $\omega_r = 0.5$, $\omega_r = 1$, and $\omega_r = 5$ for the ground state only ($l = 0$).

By following the same approach as in section 3.2 equation 10 can written on a discrete form which can be used to solve the problem numerically. (write explixitly?)

We can use equation 6 if we change the potential from ρ^2 to $V_i = \omega_r^2\rho^2 + 1/\rho$.

(write solve=plot prob?)

Comment the results for the lowest state (ground state) as function of varying strengths of ω_r.

---

d) In this exercise we want to plot the wave function for two electrons as functions of the relative coordinate r and different values of ω_r. With no Coulomb interaction you should have a result which corresponds to the non-interacting case. Plot either the function itself or the probability distribution (the function squared) with and without the repulsion between the two electrons. Varying ω_r, the shape of the wave function will change.

We are only interested in the wave function for the ground state with $l = 0$ and the two first excited states with the same symmetry and omit again the center-of-mass motion.

[5]M. Taut, Phys. Rev. A 48, 3561 - 3566 (1993).The article can be retrieved from the following web address `http://prola.aps.org/abstract/PRA/v48/i5/p3561_1`

e) Here you are asked to implement unit tests in your C++ program. For C++ users, please follow the guidelines on how to install unit tests with c++ on the webpage. These issues will be discussed in more detail at the lab and lectures during week 38.