Computational Physics Project 2

Marius B. Pran,[1] Espen Hodne,[1] and Marc K. Pestana[1]
[1] *Institute of Theoretical Astrophysics, University of Oslo*

We use numerical methods to find the energy levels of an atom with two electrons. This is done by converting it into an eigenvalue problem, and using Jacobi's method to find the eigenvalues. The eigenvalues are equivalent to the relative energy between the two electrons. To verify, we compare our results to those from Armadillo's "eigsys"-function, and to the original article written by M. Taut. We test for the non-interacting case, and get the potential energies expected. Then we test for the ground state of two electrons interacting with Coloumb force, with potential $V = \omega_r^2 + \rho^2 - \frac{1}{\rho}$. Our most accurate methods give (for varying ω_rs) scaled energies of $E_0 = 0.31$ ($\omega_r = 0.01$), $E_0 = 0.2.23$ ($\omega_r = 0.5$), $E_0 = 4.06$ ($\omega_r = 1$) and $E_0 = 17.37$ ($\omega_r = 1$). This is a report written by the authors for Project 2, Computation Physics 2017. A copy of this report, associated output files and plots, as well as C++ programming code, and other related files can be found at the github repository found at: Project files

INTRODUCTION

The two-electron problem was for a long time one of the great, unsolved problems of physics. While the single-electron atom is easy enough to solve analytically, the two-electron problem (with interaction between the electrons) is nigh impossible. Numerically, however, this problem can be solved.

Jacobi's method is a so-called 'direct eigenvalue solver'. This means it finds the eigenvalues of a matrix by working directly on the matrix itself. This means it has an inherent problem with calculating large matrices, i.e. when we get to matrix sizes of $\approx 10^{10}$.

Matrix operations are ideal for solving this problem numerically. After performing the Jacobi rotations (detailed in the *Methods and Algorithms* section), the eigenvalues of the resulting matrix will give us the energy levels of the electrons in different states.

The energies we are trying to get are specifically the scaled (detailed, again, in the *Methods and Algorithms* section) energy from the relative forces between two interacting electrons in the ground state of an atom.

METHODS AND ALGORITHMS

The aim of this project is to develop a program that uses Jacobi's (also called Givens') method for finding the eigenvalues of a symmetric matrix. For this purpose, we developed a C++ application using the Qt development environment. We developed our C++ application to model the simple case of a quantum mechanical particle moving in a potential with infinite walls, and then moved on to model the Schroedinger equation for two electrons in a three-dimensional harmonic oscillator well, with and without a repulsive Coulomb interaction. Another important aim of this project was to demonstrate the scaling of equations to simpler forms which are more aminniable to being discretized. Once the scaling is performed, care must be taken in choosing values for the free parameters otherwise the computer generated solution can fail to properly represent the intended physical systems. We show below how improper scaling can result in poor performance of our model in simulating physical phenomema under consideration. Once in a discretized form as an eigenvalue equation to be solved with Jacobi's method, we implemented this model into our application and then used our application to model the one body and then two body problems. Our sources for the scaled and descritized equations were found in the Project 2 documentation and lecture material for FYS4150. Using these equations we successfully completed the Project 2 requirements.

What follows is an abbreviated description of the eigenvalue problem, with a specialization to the case of electrons confined to move in a potential well of the type of a harmonic oscillator. Here we will assume that these electrons move in a three-dimensional harmonic oscillator potential (they are confined by for example quadrupole fields) and repel each other via the static Coulomb interaction. Also, we assume spherical symmetry. Since the Hamiltonian and has units of energy, and in this formulation has no explicit time dependence the its solution must satisfy boundary conditions so that the energy values are discretized into energy levels.

Here we will assume that the electrons move in a three-dimensional harmonic oscillator potential (they are confined by for example quadrupole fields) and repel each other via the static Coulomb interaction. We begin with the solution of the radial part of Schroedinger's equation for one electron. This equation reads

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

In our case $V(r)$ is the harmonic oscillator potential $(1/2)kr^2$ with $k = m\omega^2$ and E is the energy of the harmonic oscillator in three dimensions. $R(r)$ is the wave function, which is the unknown. The oscillator frequency is ω and the energies are

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

with $n = 0, 1, 2, \ldots$ and $l = 0, 1, 2, \ldots$.

Since we have made a transformation to spherical coordinates it means that $r \in [0, \infty)$. The quantum number l is the orbital momentum of the electron. Then we substitute $R(r) = \frac{1}{r}u(r)$ 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 introduce a dimensionless variable $\rho = \frac{1}{\alpha}r$ where α is a constant with dimension length and get

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

We will set in this project $l = 0$. Inserting $V(\rho) = \frac{1}{2}k\alpha^2\rho^2$ results in

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

Multiply thereafter with $2m\alpha^2/\hbar^2$ on both sides and obtain

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

The constant α can now be fixed so that

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

or

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

Defining

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

we can rewrite Schroedinger's equation as

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

This is the first equation to solve numerically. In three dimensions the eigenvalues for $l = 0$ are $\lambda_0 = 3, \lambda_1 = 7, \lambda_2 = 11, \ldots$.

We used the by now standard expression for the second derivative of a function u

$$u'' = \frac{u(\rho + h) - 2u(\rho) + u(\rho - h)}{h^2} + O(h^2), \quad (1)$$

where h is our step. Next we define minimum and maximum values for the variable ρ, $\rho_{\min} = 0$ and $\rho_{\max}$, respectively. We check for $\rho_{\max} = 20$ and $\rho_{\max} = 10$, of which the results can be found in tables II and III.

With a given number of mesh points, N, we define the step length h as, with $\rho_{\min} = \rho_0$ and $\rho_{\max} = \rho_N$,

$$h = \frac{\rho_N - \rho_0}{N}.$$

The value of ρ at a point i is then

$$\rho_i = \rho_0 + ih \qquad i = 1, 2, \ldots, N.$$

The Schroedinger equation for a value ρ_i can be rewritten as

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

or in a more compact way

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

where $V_i = \rho_i^2$ is the harmonic oscillator potential. This potential will have a different form in the two body problem, but as will be show, it really doesn't change the form to the wave equation dramatically, and more importantly the code for the algorithm. We simply used a different form the of the potential.

We define first the diagonal matrix element

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

and the non-diagonal matrix element

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

In this case the non-diagonal matrix elements are given by the constant e_i. Thus, *All non-diagonal matrix elements are equal*. With these definitions the Schroedinger equation takes the following form

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

where u_i is unknown. We can write the latter equation as a matrix eigenvalue problem thus

$$\begin{bmatrix} d_0 & e_0 & 0 & 0 & \ldots & 0 & 0 \\ e_1 & d_1 & e_1 & 0 & \ldots & 0 & 0 \\ 0 & e_2 & d_2 & e_2 & 0 & \ldots & 0 \\ \ldots & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots \\ 0 & \ldots & \ldots & \ldots & \ldots e_{N-1} & d_{N-1} & e_{N-1} \\ 0 & \ldots & \ldots & \ldots & \ldots & e_N & d_N \end{bmatrix} \begin{bmatrix} u_0 \\ u_1 \\ \ldots \\ \ldots \\ \ldots \\ u_N \end{bmatrix} = \lambda \begin{bmatrix} u_0 \\ u_1 \\ \ldots \\ \ldots \\ \ldots \\ u_N \end{bmatrix}. \quad (2)$$

Since the values of u at the two endpoints are known via the boundary conditions, we can skip the rows and columns that involve these values. Inserting the values for d_i and e_i we have the following matrix

$$\begin{bmatrix} \frac{2}{h^2}+V_1 & -\frac{1}{h^2} & 0 & 0 & \dots & 0 & 0 \\ -\frac{1}{h^2} & \frac{2}{h^2}+V_2 & -\frac{1}{h^2} & 0 & \dots & 0 & 0 \\ 0 & -\frac{1}{h^2} & \frac{2}{h^2}+V_3 & -\frac{1}{h^2} & 0 & \dots & 0 \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ 0 & \dots & \dots & \dots & -\frac{1}{h^2} & \frac{2}{h^2}+V_{N-2} & -\frac{1}{h^2} \\ 0 & \dots & \dots & \dots & \dots & -\frac{1}{h^2} & \frac{2}{h^2}+V_{N-1} \end{bmatrix} \tag{3}$$

Project 2 a): Mathematical intermezzo. An central notion in Physics in the idea of the invariance of physical quantities under coordinate transformations. If a scalar or vector quantities have real physical significance, then their physically significant values shouldn't depend on the coordinated system used to represent them. Examples of such coordinate transformations are the Orthogonal and Unitary Transformations. Orthogonal transformations apply to $\mathbb{R}^n$ and Unitary Transformations operate on complex vector spaces. Here we showed that orthonormal transformations preserve two fundamental properties of basis vectors, i.e. orthonormality and the dot product between vectors. We focused on Orthogonal transformations since the algebra is more direct. The unitary case involve accounting for complex valued coordinates which don't change the final results. First, we proved that an orthogonal transformation, hereafter denoted by $\mathbf{U}$, preserves the orthonormality of an orthonormal basis of vectors. To see this consider first a basis vectors $\mathbf{v}_i$ assumed to be in $\mathbb{R}^n$,

$$\mathbf{v}_i = \begin{bmatrix} v_{i1} \\ \dots \\ \dots \\ v_{in} \end{bmatrix}$$

We assume that the basis is orthogonal, that is

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

We showed under these circumstances that $\mathbf{U}$ preserves the dot product and orthonormality meaning that if $\mathbf{v}_i$ and $\mathbf{v}_j$ are orthnormal then so are $\mathbf{U}\mathbf{v}_i$ and $\mathbf{U}\mathbf{v}_i$. Also, with respect to the dot product, $\mathbf{v}_i \cdot \mathbf{v}_j = \mathbf{U}\mathbf{v}_i \cdot \mathbf{U}\mathbf{v}_i$ which is our dot product preservation identity.
First we showed that an orthogonal transformation preserves orthonormality of the orthonormal basis

$$\mathbf{w}_i = \mathbf{U}\mathbf{v}_i,$$

Proof/

Let $\mathbf{U}$ be a unitary transformation so that $\mathbf{U}$ satisfies:

$$\mathbf{U}^T = \mathbf{U}^{-1},$$

In particular,

$$\mathbf{U}^{-1}\mathbf{U} = \mathbf{U}\mathbf{U}^{-1} = \mathbf{I},$$

Furthermore the following simple property of the matrix transpose also holds,

$$(\mathbf{U}\mathbf{v}_i)^T = \mathbf{v}_i^T \mathbf{U}^T,$$

Using these equations, and expanding the following expression, we obtained:

$$\mathbf{w}_i^T \mathbf{w}_j = (\mathbf{U}\mathbf{v}_i)^T (\mathbf{U}\mathbf{v}_j),$$

using the property of the tranpose, we get

$$(\mathbf{U}\mathbf{v}_i)^T (\mathbf{U}\mathbf{v}_j) = \mathbf{v}_i^T \mathbf{U}^T (\mathbf{U}\mathbf{v}_j),$$

Matrix multiplication is associative so,

$$\mathbf{v}_i^T \mathbf{U}^T (\mathbf{U}\mathbf{v}_j) = \mathbf{v}^T (\mathbf{U}^T \mathbf{U}) \mathbf{v}_j,$$

from our first fun fact, we get

$$\mathbf{v}_i^T (\mathbf{U}^T \mathbf{U}) \mathbf{v} = \mathbf{v}^T (\mathbf{I}) \mathbf{v}_j,$$

and from another application of associativity and our assumption about $\mathbf{v}$,

$$\mathbf{v}_i^T (\mathbf{U}^T \mathbf{U}) \mathbf{v} = \mathbf{v}^T (\mathbf{I}\mathbf{v}_j),$$

without further ado yet summarizing our results, we obtain

$$\mathbf{w}_j^T \mathbf{w}_i = \mathbf{v}_j^T \mathbf{v}_i = \delta_{ij},$$

which show that $\mathbf{U}$ conserves orthonormality.
To show that the dot product is perserved under Orthogonal Transformations, we'll will need another set of facts. First the equations,

$$\mathbf{U}^{-1}\mathbf{U} = \mathbf{U}\mathbf{U}^{-1} = \mathbf{I},$$

contain more than meets the eye. Recall that $\mathbf{U}^{-1} = \mathbf{U}^T$ means that the rows of $\mathbf{U}^{-1}$ equal the rows of $\mathbf{U}$, therefore by the rules or matrix multiplications $\mathbf{U}^{-1}\mathbf{U}$ gives the dot product of every row of $\mathbf{U}$ with every row of itself. The fact that the dot product of every row with itself equals 1, means the rows are unit vectors. In addition, the dot product of every row with a different row is 0 implies that they are orthogonal. In summary, the rows of $\mathbf{U}$ are unit vectors that are mutually orthogonal (similarly $\mathbf{U}\mathbf{U}^{-1} = \mathbf{I}$ implies similar facts hold for the colunms of $\mathbf{U}$). And since the dimension of $\mathbf{U}$ is $\mathbf{n}$, the fact that the rows (or columns) are orthonormal, linearly independent and a maximal set, they must constitute an orthonormal basis. Interestingly, though not crucial for this proof, since $\mathbf{U}$ has linearly independent rows (and therefore columns) $\mathbf{U}$ is an automorphism on $\mathbb{R}^n$. So if

we let $\mathbf{u}^i$ be the i^{th} column of U, and $\mathbf{u}^j$ be the j^{th} column of U, where $i, j = 1...n$ then we have the following relations,

$$\mathbf{u}^i \cdot \mathbf{u}^j = \delta_{ij},$$

furthermore from the definition of matrix multiplication, we have

$$\mathbf{U}\mathbf{v} = \sum_{i=1}^{n} \mathbf{u}^i v_i$$

and finally we have that the dot product is bilinear in the context of vectors of $\mathbb{R}^n$, so we can distribute and re-arrange dot products as if we're doing arithmatic on real numbers. So the following dot product between $\mathbf{v}$ and $\mathbf{w}$, where $\mathbf{v}$, $\mathbf{w}$ are now arbirary vectors in $\mathbb{R}^n$, becomes

$$\mathbf{U}\mathbf{v} \cdot \mathbf{U}\mathbf{w} = (\sum_{i=1}^{n} \mathbf{u}^i v_i) \cdot (\sum_{j=1}^{n} \mathbf{u}^j w_j)$$

Since the dot product is bilinear, we can re-arrange the terms into the double summation as follows

$$(\sum_{i=1}^{n} \mathbf{u}^i v_i) \cdot (\sum_{j=1}^{n} \mathbf{u}^j w_j) = \sum_{i=1}^{n}\sum_{j=1}^{n} v_i w_j (\mathbf{u}^i \cdot \mathbf{u}^j)$$

Using orthonormality of the columns of U, we have

$$\sum_{i=1}^{n}\sum_{j=1}^{n} v_i w_j (\mathbf{u}^i \cdot \mathbf{u}^j) = \sum_{i=1}^{n}\sum_{j=1}^{n} v_i w_j \delta_{ij},$$

Using the definition of δ_{ij},

$$\mathbf{U}\mathbf{v} \cdot \mathbf{U}\mathbf{w} = \sum_{i=1}^{n}\sum_{j=1}^{n} v_i w_j \delta_{ij} = \sum_{i=1}^{n} v_i w_i = \mathbf{u} \cdot \mathbf{w}$$

so the dot product is preserved under orthogonal transformations!

Project 2 b): Setting up a code for the non-interacting case. For this part of Project 2 we wrote a function which implements Jacobi's rotation algorithm (see Lecture notes chapter 7).

In order to solve Eq. (2). We define the quantities $\tan\theta = t = s/c$, with $s = \sin\theta$ and $c = \cos\theta$ and

$$\cot 2\theta = \tau = \frac{a_{ll} - a_{kk}}{2a_{kl}}.$$

We can then define the angle θ so that the non-diagonal matrix elements of the transformed matrix a_{kl} become zero and we obtain the quadratic equation (using $\cot 2\theta = 1/2(\cot\theta - \tan\theta)$

$$t^2 + 2\tau t - 1 = 0,$$

resulting in

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

and c and s are easily obtained via

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

and $s = tc$. Using our C++ program that implements the Jacobi method defined above, we were able to perform the following analysis

- Checked how many mesh points N you need in order to get the lowest three eigenvalues with approximately four leading digits after the decimal point. Remember to check the eigenvalues for the dependency on the choice of $\rho_{\max} = \rho_N$.
- Estimate the number of transformations and extract a behavior as function of the dimensionality of the matrix.
- Checked our results against the Armadillo function for solving eigenvalue problems. The armadillo function *eigsys* can be used to find eigenvalues and eigenvectors.

Project 2 c): Implementing tests in your code. In this project we will implemented so-called **unit** tests. Our unit tests are mainly meant to test mathematical properties of our algorithm. During the development phase of a program it is useful to devise tests that your program should pass. We successfully used unit tests to perform the following test of our code:

- Tested for the condition that for a given simple test matrix (say a 5×5 matrix) our algorithm for searching for the largest non-diagonal element always returns the correct answer.
- Furthermore, for a known simple matrix, irrespective of changes made, we verified that we got the same and correct eigenvalues.
- We also verified that the orthogonality shown in exercise (a) is preserved.

Project 2 d): The interacting case. We then modeled the two electrons in a harmonic oscillator well which also interact via a repulsive Coulomb interaction. Following the derivation presented in the documentation for Project 2, we end up with the r-dependent Schroedinger equation becomes

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

This equation is similar to the one we had previously in (b) and we introduce again a dimensionless variable $\rho = r/\alpha$. Repeating the same steps as above, 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 wanted to manipulate this equation further to make it as similar to that in (a) as possible. As in the notes for Project 2 we defined a new 'frequency'

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

and after scaling we used Schroedinger's equation as

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

We treat ω_r as a parameter which reflects the strength of the oscillator potential.

Here we studied the cases $\omega_r = 0.01$, $\omega_r = 0.5$, $\omega_r = 1$, and $\omega_r = 5$ for the ground state only, that is the lowest-lying state.

With no repulsive Coulomb interaction we found a result which corresponds to the relative energy of a non-interacting system. Further, we found our results are stable as functions of $\rho_{\max}$ and the number of steps.

We were only interested in the ground state with $l = 0$. We omitted the center-of-mass energy.

We reused the code written for part (b), but used a changed potential from ρ^2 to $\omega_r^2\rho^2 + 1/\rho$.

For specific oscillator frequencies, the above equation has answers in an analytical form, see the article by M. Taut, Phys. Rev. A 48, 3561 (1993). We compared our results with those found in this paper. See the results section for the outcome of our comparisonl.

RESULTS

TABLE I: Scaled solution to the eigenvalues of an electron in a harmonic oscillator. The numerical values are calculated using $\rho_{max} = 10$, except for one column which is calculated using $N = 50$, $\rho_{max} = 20$. The exact values are calculated using $E_n = \hbar\omega(2n + 3/2)$ and then scaled the same way as our equations.

	N = 50	$\rho_{max} = 20$	N = 100	N=150	Exact
E_0	2.98793	2.95197	2.99693	2.99863	3
E_1	6.93939	6.75043	6.98465	6.99314	7
E_2	10.8514	10.3769	10.9625	10.9833	11

TABLE II: Scaled solutions to the ground-state eigenvalies E_0 of two electrons interacting in an atom. The results are calculated using $\rho_{max} = 20$.

ω_r	N = 50	N = 100
0.01	0.137699	0.137716
0.5	2.21761	2.22696
1.0	4.0062	4.04496
5.0	15.9763	17.1182

TABLE III: Scaled solutions to the ground-state eigenvalies E_0 of two electrons interacting in an atom. The results are calculated using $\rho_{max} = 10$.

ω_r	N = 50	N = 100
0.01	0.31157	0.31162
0.5	2.22702	2.22933
1.0	4.04321	4.05466
5.0	17.1248	17.3680

First, we look at the non-interacting case. In Table I we look at the three lowest energy levels, and compare them to the analytical answer for a harmonic oscillator. The numbers we get using our program is consistently the same as what we get if we were to use the built-in Armadillo function "eigsys".

Secondly, we take a look at the results of the example using interacting electrons in an atom. Table II and III we see the results we get for calculating the ground state energies using different values for ω_r. As we can see, the results are quite different from the non-interactive example. The potential now looks like $V = \omega_r 2 + \rho 2 - \frac{1}{\rho}$.

CONCLUSIONS AND DISCUSSION

For the non-interacting case, the result is about exactly as expected. The larger our matrices were, the more accurate the eigenvalues became. We never reached the exact values, but we got pretty close.

The interesting part to look at here is what happens at ρ_{MAX} varies. If we increase the maximum limit, we are able to 'catch' the higher eigenvalues more accurately. However, the result becomes more inaccurate for the lower values. Just as an example, the results gets visibly farther from the exact solution when we increase ρ_{MAX} from 10 to 20. This makes sense - after all, we are looking at the lowest eigenvalues.

Next, we run the program for two interacting electrons in the ground state. Things here get more interesting. Sadly, we cannot compare to the exact solution, as it is nigh impossible to solve analytically. However, the same principles of balancing between matrix size n and ρ_{MAX} should still apply.

We run the program for both $n = 50$ and $n = 100$, with $\omega_r = 0.01, 0.5, 1, 5$ and $\rho_{MAX} = 10, 20$. We also

ran the program for $n = 150$, $\omega_r = 0.01$ and $\rho_{MAX} = 20$. However, the result we got, $E_0 = 0.137729$, had a difference of 0.000003 compared the equivalent $n = 100$ result. Therefore we decided it was not worth the considerately higher computation time it would take to run it for $n = 150$ in the other cases.

We will now discuss the $n = 100$ case, as that has the highest precision.

The most noteworthy thing to notice in tables II and III is how changing ρ_{MAX} affects the result. We can see that for $\omega_r = 0.5$ the energy is affected in the 3rd decimal, and for $\omega_r = 1$ the result is affected in the 2nd decimal. However, in the "edge cases", $\omega_r = 0.01$ and $\omega_r = 5$, the energy changes in the first decimal. This is due to how the wave function scales: with $e^{-\omega_r^2\rho^2}$. Since it is exponential, the largest changes comes from the largest and smallest ω_r.

All in all, we have now solved the 2-electron problem, and found the energies for different potentials. It should now be possible to extend the program to more electrons and whatever potentials we want. This shows how immensely useful numerical methods can be as a tool to solve impossible analytical problems.

EXTRA MATERIAL

You can find the code we used to calculate these results at:

[1] M. Taut, Phys. Rev. A 48, 3561 (1993)