Project 2: Solving Schrödinger’s equation for two electrons in a 3D harmonic oscillator well using Jacobi’s rotation algorithm

Adam Powers

8 May 2016

Abstract

We derive expressions for the Hamiltonian of one and two electrons trapped in a harmonic-oscillator potential. Jacobi’s algorithm used to diagonalize a system of equations is used and implemented in a C++ program in order to find the first few eigenvalues and eigenfunctions of the system. The results are compared to a function from the Armadillo linear algebra library[1] and to an analytic solution[2]. All of the data from these sources are in agreement. The performance of Jacobi’s algorithm is then compared to the Armadillo function and is found to be slower than the Armadillo algorithm.

1 Introduction

The electron in a potential well is a commonly studied system. It is considered in the study of quantum dots, quantum computing and many other areas of solid state physics.[3] For this paper, we will only be worried about the case of one or two electrons confined by a harmonic oscillator potential. The one-electron case is simple enough, dealing only with the interactions between the electron and the potential. But once a second electron is added into the system, it becomes more complex, since one also has to account for the Coulomb repulsion force between the two electrons. Even with these complexities there are analytic solutions that can be solved for these systems. The solution to the one-electron problem is solved in many introductory quantum mechanics texts (eg. Shankar[4] p. 351], but an exact solution for the two-electron system was only more recently found by Taut[2].

Instead of analytic solutions, it is often easier to solve for a numeric solution. One simple way to do this is using Jacobi’s rotation algorithm. This produces comparable results to the analytic solutions and is easy to implement. It’s downfall is that is is quite inefficient as will be explored in a late section.

2 Theory

2.1 One electron case

Assuming spherical symmetry, we can write the radial part of Schrödinger's equation for a trapped electron in three dimensions as

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

Where R(r) is the radial wave function of the electron, l is the angular momentum, V(r) is the confining potential, and E is the energy. If we take the potential to be the three-dimensional harmonic oscillator potential,

$$V(r)=\frac{1}{2}m\omega^2r^2, \tag{2}$$

then the energies are known to be

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

with principal quantum number n = 0,1,2,… and angular momentum number l = 0,1,2,…[4]

Next, make the substitution R(r) = (1/r)u(r) in (1), we can also set l = 0 and neglect the centrifugal barrier term, we can write our equation as

$$-\frac{\hbar^2}{2m}\frac{d^2}{dr^2}u(r)+\frac{1}{2}m\omega^2r^2u(r)=Eu(r)$$

where V(r) has been replaced with the expression for the harmonic oscillator potential introduced above. This can be simplified by introducing the dimensionless variable $\rho=\frac{r}{\alpha}$ where α is an arbitrary constant with dimension of length. Rearranging our previous equation with this is mind, we get

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

which suggests that we define $\alpha=\sqrt{\hbar/m\omega}$ and $\lambda=2E/\hbar\omega$ to get

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

2.2 Two-electron case

The theory of the two-electron case is similar to what is presented above, we just need to also consider the Coulomb repulsion force between the two electrons as well as the effect of the potential one the two electrons.

We begin again with the radial part of Schrödinger's equation for the two electrons in a potential V:

$$(-\frac{h^2}{2m}\frac{\delta^2}{\delta r_1^2}-\frac{h^2}{2m}\frac{\delta^2}{\delta r_2^2}+V(r_1,r_2))u(r_1,r_2)=Eu(r_1,r_2). \quad (5)$$

The potential V must include the interaction of each electron with the well as well as the Coulomb repulsion between the two electrons. This can be represented as

$$V(r_1,r_2)=\frac{1}{2}m\omega^2 r_1^2+\frac{1}{2}m\omega^2 r_2^2+\frac{ke^2}{|r_1-r_2|}$$

where $k=1/4\pi\epsilon_0$.

Equation 5 can be simplified by introducing the electron separation distance $r=|r_1-r_2|$ and the center-of-mass coordinate $R=(1/2)(r_1+r_2)$. Using this substitution, neglecting the center of mass motion, inserting a unitless variable $\rho=\frac{r}{\alpha}$ as before and rearranging, we get

$$(-\frac{d^2}{d\rho^2}+\frac{m^2\omega^2\alpha^2}{4\hbar^2}\rho^2+\frac{ke^2m\alpha}{\hbar^2\rho})u(\rho)=\frac{m\alpha^2}{\hbar^2}E_r u(\rho)$$

Define $\alpha=\frac{\hbar^2}{mke^2}$, $\omega_r^2=\frac{m^2\omega^2\alpha^2}{4\hbar^2}$, and $\lambda=m\alpha^2E_r/\hbar^2$ to find

$$-\frac{d^2u}{d\rho^2}+\omega_r^2\rho^2u(\rho)+\frac{1}{\rho}u(\rho)=\lambda u(\rho) \quad (6)$$

Equation 6 is nearly identical to (4). The only difference is the replacement of the potential $V(\rho)=\rho^2$ with the new potential $V(\rho)=\omega_r^2\rho^2+1/\rho$. This implies that we can develop a numerical solution to this simplified equation

$$-\frac{d^2u}{d\rho^2}+V(\rho)u(\rho)=\lambda u(\rho) \quad (7)$$

and then plug in the two potentials to solve each of the two cases.

3 Numerical Solution

To solve (7) numerically we need to create a finite difference equation to express the second derivative

$$u''(x) = \frac{u(\rho+h)+u(\rho-h)-2u(\rho)}{h^2} + O(h^2) \tag{8}$$

with step size h. To discretize, define:

$$u_i = u(\rho)$$

$$u_{i\pm1} = u(\rho \pm h)$$

$$\rho_i = ih$$

$$V_i = V(\rho_i).$$

Using these between $\rho = 0$ and some arbitrary ρ_{max}:

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

for $i = 0,1, \dots , N$ and $h = \ \rho_{max}/N$. Rearranging this gives

$$-\frac{1}{h^2}u_{i+1} - \frac{1}{h^2}u_{i-1} + \left(\frac{2}{h^2} + V_i\right)u_i = \ \lambda u_i$$

which suggests treating the problem as a system of linear equations

$$\boldsymbol{Au} = \ \lambda\boldsymbol{u} \tag{9}$$

where the tridiagonal matrix **A** is defined as

$$\begin{pmatrix} \frac{2}{h^2}+V_1 & -\frac{1}{h^2} & & & \\ -\frac{1}{h^2} & \frac{2}{h^2}+V_2 & -\frac{1}{h^2} & & \\ & -\frac{1}{h^2} & \ddots & \ddots & \\ & & \ddots & \ddots & -\frac{1}{h^2} \\ & & & -\frac{1}{h^2} & \frac{2}{h^2}+V_{N-1} \end{pmatrix} \tag{10}$$

and the vector $\boldsymbol{u} = (u_1\ u_2 \dots u_{N-1}\ )^T$.

3.1 Jacobi's rotation algorithm

Our approach to solving the above system of linear equations will be to using Jacobi's rotation algorithm. This algorithm will iteratively apply similarity transformations to the matrix **A** until it becomes diagonal.[3] Tso, for orthogonal matrices $\mathbf{S_i}$, the matric is transformed as

$$\boldsymbol{S_m^T S_{m-1}^T \dots S_1^T A S_1 S_2 \dots S_m = D}$$

where the final matrix **D** is diagonal. Naturally, identical transformations must be applied to the eigenvector to maintain equality, but since the constant eigenvalue can be factored out of all of the matrix multiplications, these similarity transformations do not change the eigenvalues of the matrix.[3]

Jacobi's rotation algorithm, in particular, chooses these orthogonal matrices to be

$$\mathbf{S} = \begin{pmatrix} 1 & 0 & \dots & 0 & \dots & 0 & \dots & 0 \\ 0 & 1 & \dots & 0 & \dots & 0 & \dots & 0 \\ \vdots & \vdots & & \vdots & & \vdots & & \vdots \\ 0 & 0 & \dots & \cos\theta & \dots & \sin\theta & \dots & 0 \\ \vdots & \vdots & & \vdots & & \vdots & & \vdots \\ 0 & 0 & \dots & -\sin\theta & \dots & \cos\theta & \dots & 0 \\ \vdots & \vdots & & \vdots & & \vdots & & \vdots \\ 0 & 0 & \dots & 0 & \dots & 0 & \dots & 1 \end{pmatrix}, \tag{11}$$

a generalized matrix with elements

$$S_{ij} = \begin{cases} 1 & i = j, \quad i, j \notin \{p, q\} \\ \cos\theta & (ij = pp) \cup (ij = qq) \\ \sin\theta & ij = pq \\ -\sin\theta & ij = qp \\ 0 & \text{elsewhere.} \end{cases}$$

Note that the cos (θ) elements are not necessarily along the diagonal of the matrix.

The values of cos (θ) and $\sin(\theta)$ can be found by looking at the elements affected by the transformation:

$$\begin{pmatrix} b_{pp} & b_{pq} \\ b_{qp} & b_{qq} \end{pmatrix} = \begin{pmatrix} c & -s \\ s & c \end{pmatrix} \begin{pmatrix} a_{pp} & a_{pq} \\ a_{qp} & a_{qq} \end{pmatrix} \begin{pmatrix} c & s \\ -s & c \end{pmatrix}$$

We have defined c ≡ cos (θ) and s ≡ $\sin(\theta)$. To make the matrix **A** diagonal, we want to make $b_{pq} = b_{qp} = 0$. Therefore we carry out the matrix multiplication and find

$$b_{qp} = a_{pq}(c^2 - c^2) + (a_{pp} - a_{qq})cs = 0 \tag{12}$$

If we define

$$\tau = \frac{a_{qq} - a_{pp}}{2a_{pq}} \tag{13}$$

$$t = \tan(\theta) = \frac{s}{c} \tag{14}$$

then (12) can be rearranged and written as

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

which has roots

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

To help prevent t from diverging due to loss of numerical precision, when $|\tau| << 1$, we choose the smaller of the two roots. Thus,

$$t = \begin{cases} \dfrac{1}{\tau + \sqrt{\tau^2 + 1}} & \tau \geq 0 \\ \dfrac{1}{\tau - \sqrt{\tau^2 + 1}} = \dfrac{-1}{-\tau + \sqrt{\tau^2 + 1}} & \tau < 0 \end{cases} \tag{15}$$

Knowing t then gives values for s and c via trigonometric identities

$$c = \frac{1}{\sqrt{1+t^2}} \quad \text{and} \quad s = tc \tag{16}$$

and thereby defines the elements of the rotation matrix **S**. Therefore the elements of the transformed matrix **B** = $\mathbf{S}^T$**AS** are

$$b_{ip} = ca_{ip} - sa_{iq}, \quad i \notin \{p, q\} \tag{17}$$
$$b_{iq} = ca_{iq} + sa_{ip}, \quad i \notin \{p, q\} \tag{18}$$
$$b_{pp} = c^2 a_{pp} - 2csa_{pq} + s^2 a_{qq} \tag{19}$$
$$b_{qq} = c^2 a_{qq} + 2csa_{pq} + s^2 a_{pp} \tag{20}$$
$$b_{pq} = a_{pq}(c^2 - s^2) + (a_{pp} - a_{qq})cs = 0 \tag{21}$$

with equivalent transformations for the elements opposite the diagonal to maintain orthogonality. These equations are the basis of the Jacobi rotation algorithm.

The algorithm itself consists of the following steps:

1. Find the largest off-diagonal element in **A**. The indices of this element will be defined p and q.
2. Calculate τ using (13), $\tan(\theta)$ using (15), and cos (θ) and sin (θ) using (16).
3. Calculate the new matrix elements b_{ij} using (17-21).
4. Check to see if the norm of the off-diagonal elements is less than some tolerance ϵ.
    - If not, repeat the process until it is

4 Results

We used the algorithm explained above in Section 3 in a program written using C++ to solve both the one- and two-electron problems.

4.1 One electron case

We ran our algorithm with the V(r^2) potential. The data converged to eigenvalues of 3.3463, 6.1362, and 11.057 with percent differences of 11.5%, 12.34%, and 0.52% respectively when compared to the expected values of 3, 7, and 11. Our algorithm was definitely not as accurate as it could be. After approximately 200 iterations at a matrix size equaling [375 375], our values did not get any closer to the expected values.

4.2 Two electron case

Our two-electron solver was good at all.

ω_r	Exact	Calculated	Difference
0.25	1.250	1.749	.499
0.05	0.3500	2.593	2.243
0.01827	0.1644	2.942	2.7776

Table 1: Comparison of calculated eigenvalues of the two-electron system to an analytic solution for three values of the parameter ω_r. The exact values are from Taut[2].

After going back through our program, we thought we had forgotten the 1/ ρ factor, but we had not. I am not sure why our program gave us such bad numbers and they seemed to get worse as we increased ω_r. Our guess is the problem probably lies in the initialization of the potential energy matrix, but the problem was not found by the writing of this paper. The issue would need to be flushed out before further study could be done on this problem.

5 Conclusion

The Jacobi rotation algorithm is a straightforward method to solving eigenvalues and eigenvectors of matrices. We were not able to implement it well for two different situations and therefore were not able to compare its performance to other types of solvers, but in reading through different literatures, it seems as though it being straightforward does not make it efficient. Armadillo's linear algebra library seems to have a more efficient solver for eigenvalues and eigenvectors than a Jacobi rotation algorithm could ever be.

[1] C. Sanderson, NICTA, Tech. Rep. (2010).

[2] M. Taut, Physical Review A **48**, 3561 (1993).

[3] M Hjorth-Jensen, "Computational Physics MSU," https://github.com/CompPhysics/ComputationalPhysicsMSU (2016).

[4] R. Shankar, *Principles of Quantum Mechanics*, 2nd ed. (Springer US, Boston, MA, 1994).

[5] https://arma.sourceforge.com