Project 2

Benedicte Allum Pedersen, Emil Heland Broll
Fredrik Oftedal Forr

1 Abstract

Main summary of our work - in this project we have ... Eigenvalue problems, from the equations of a buckling beam to Schroedinger's equation for two electrons in a three-dimensional harmonic oscillator well

2 Introduction

Aims and rationale of the physics case, what you have done - end with brief summary of structure of report In this project we will solve eigenpair-problems using numerical calculations. We will also implement unit testing in our code to avoid mathematical and programming errors.

We begin by explaining the theoretical models and algorithms we use to ompute the eigenvalues, along with technicalities related to the programming of these models.

We will then present and discuss our results, before finally reaching a conclusion regarding the methods we use in terms of efficiency and precision.

3 Methods

Theoretical models and technicalities

3.1 Mathematics

We will start off by proving that orhogonal or unitary transformations preserves the orthogonality/dot product of the vectors.

Starting with an orthogonal basis of vectors, $\mathbf{v}_i$, we know that:

$$\mathbf{v}_i = \begin{bmatrix} v_{i1} \\ \vdots \\ v_{in} \end{bmatrix}, \qquad \mathbf{v}_j^T \mathbf{v}_i = \delta_{ij}$$

An orthogonal transformation gives us the following:

$$\mathbf{w}_i = U\mathbf{v}_i, \text{where } U^T U = UU^T = \mathbf{I}$$

We want to prove that orthogonality is preserved:

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

Now that we have proved that orthogonal transformations preserve orthogonality, we can move on to performing our Jacobi iterations. To do so, we use an orthogonal transformation matrix, S:

$$S = \begin{bmatrix} 1 & 0 & \cdots & & & & \\ 0 & 1 & 0 & & & & \\ \vdots & 0 & \ddots & & & & \\ & & & \cos\theta & 0 & \cdots & \sin\theta \\ & & & 0 & 1 & \cdots & 0 \\ & & & \vdots & \vdots & \ddots & \vdots \\ & & & -\sin\theta & 0 & \cdots & \cos\theta \end{bmatrix}, \qquad S^T = S^{-1}$$

We simpltfy by assigning:

$$c = \cos\theta, s = \sin\theta, t = \tan\theta = \frac{s}{c}$$

For our positive definite matrix A, we perform the transformation $S^TAS = B$, giving us the general expression:

$$\begin{aligned} & b_{ii} = a_{ii}, i \neq k, i \neq l \\ & b_{ik} = a_{ik}c - a_{il}s, i \neq k, i \neq l \\ & b_{il} = a_{il}c + a_{ik}s, i \neq k, i \neq l \\ & b_{kk} = a_{kk}c^2 - 2a_{kl}cs + a_{ll}s^2 \\ & b_{ll} = a_{ll}c^2 - 2a_{kl}cs + a_{kk}s^2 \\ & b_{kl} = (a_{kk} - a_{ll})cs + a_{kl}(c^2 - s^2) \end{aligned}$$

By choosing the largest non-diagonal element in the original matrix A, we can fix θ by choosing to set the largest non-diagonal element to zero. From this, we can deduce the required values of c and s:

$$c = \frac{1}{\sqrt{(1+t^2)}}, \qquad s = tc$$

$$t = -\tau \pm \sqrt{1+\tau^2}, \qquad \tau = \frac{a_{ll} - a_{kk}}{2a_{kl}}$$

This will result in a new matrix, B. We can then find the largest non-diagonal element of the new matrix and repeat the algorithm until this value is less than a given tolerance.

$$B_2 = S_2^T B S_2$$

When this is achieved, we will have a diagonal matrix, where all non-diagonal matrix elements are approaching 0.

$$D = S_n^T S_{n-1}^T \cdots S_1^T A S_1 \cdots S_{n-1} S_n$$

Since we have only performed orthogonal transformations, the eigenvalues for D and A will be the same. Since diagonal matrices have their eigenvalues on the diagonal, we can easily read off the eigenvalues.

3.1.1 Quanton dots in three dimensions, one electron

We will now consider a physical application of our algorithm by adapting it in order to solve the three dimensional Schrödinger Equation. The general radial form of this equation is:

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

We can now do a series of subtitutions and insert some boundary conditions. We assume $l = 0$.

$$R(r)=\frac{1}{r}u(r), \qquad u(0)=u(\infty)=0$$
$$\rho=\frac{1}{\alpha}r$$
$$V(\rho)=\frac{1}{2}k\alpha^2\rho^2$$
$$\frac{mk}{\hbar^2}\alpha^4=1\Rightarrow\alpha=\left(\frac{\hbar^2}{mk}\right)^{1/4}$$
$$\lambda=\frac{2m\alpha^2}{\hbar^2}E$$

We can then rewrite the Schröedinger Equation like this:

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

We can now discretise this equation by defining the step length h and $\rho_{min} = \rho_0$. We also need to approximate ∞, as the computer cannot represent infinity: $p_{max} = \rho_N$.

$$h=\frac{\rho_N-\rho_0}{N}$$
$$\rho_i=\rho_0+ih, \qquad i=1,\dots,N$$

The Schröedinger equation now becomes:

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

Where $u_i = u(\rho_i)$ and $u_{i\pm1} = u(\rho_i) \pm h$ and $V_i = \rho_i^2$. We can then see that the Schröedinger equation can be simplified even more:

$$
\begin{aligned}
d_i u_i + e_i u_{i-1} + e_i u_{i+1} = \lambda u_i \\
d_i = \frac{2}{h^2} + V_i \\
e_i = -\frac{1}{h^2} = e
\end{aligned}
$$

Where d_i are the diagonal elements of our tridiagonal matrix, and e_i are the non-diagonal elements, which are all equal. This lets us rewrite the Schröedinger equation as a linear problem:

$$
\begin{bmatrix}
d_1 & e & 0 & 0 & \dots & 0 & 0 \\
e & d_2 & e & 0 & \dots & 0 & 0 \\
0 & e & d_3 & e & 0 & \dots & 0 \\
\dots & \dots & \dots & \dots & \dots & \dots & \dots \\
0 & \dots & \dots & \dots & \dots e & d_{N-2} & e \\
0 & \dots & \dots & \dots & \dots & e & d_{N-1}
\end{bmatrix}
\begin{bmatrix}
u_1 \\ u_2 \\ \dots \\ \dots \\ \dots \\ u_{N-1}
\end{bmatrix}
= \lambda
\begin{bmatrix}
u_1 \\ u_2 \\ \dots \\ \dots \\ \dots \\ u_{N-1}
\end{bmatrix}
$$

We can now see that we can use our algorithm to solve our radial Schröedinger equation. The resulting value λ is a unitless number, which can be tranformed back into energies by using the value α.
We can then run our algorithms with different number of integration points, N, and the approximation to ∞, ρ_{max}.

3.1.2 Quantum dots in three dimensions, two electrons

If we now consider a system with two electrons, the electron repulsion between them will come into play in the Schröedinger equation. By doing a lot of the equivalent substitutions and rewriting the equation, we get to a point where we end up with a linear problem, with a tridiagonal matrix with diagonal elements:

$$d_i = \frac{2}{h^2} + \omega_r^2 \rho^2 + 1/\rho$$

Here, ω_r represents the strength of the oscillator potential. We will study the cases $\omega_r = 0.01, 0.5, 1, 5$, for the ground state ($l = 0$), and compare these to known analytical solutions of the problem.

3.2 Programming

To avoid errors in our code, we implement unit tests at the following points:
* test * Test

4 Results, discussion

For our eigenproblem-solver, we tested and found a reasonable tolerance, 10^{-8}, so that the algorithm converges without reaching the maximum number of iterations, n^3. We test this using a tridiagonal matrix:

$$A = \begin{bmatrix} 2 & -1 & 0 & \dots & 0 & 0 \\ -1 & 2 & -1 & \dots & 0 & 0 \\ 0 & -1 & 2 & \dots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \dots & 2 & -1 \\ 0 & 0 & 0 & \dots & -1 & 2 \end{bmatrix}$$

Tabel 1 shows the number of similarity transformations were needed to reach a point where all non-diagonal matrix elements approach 0, for matrix dimensionalities.

Dimensionality	# of transformations
4	6
10	141
20	551
50	3529

We can see that it looks like $i = 1.49n^2 - 1.63n - 4.23$, where i is number of iterations and n is dimensinality of the matix.

Table 2 shows a comparison of eigenvalues found with Armadillo's eigensystem-solver, on our tridiagonal matrix A.

λ_{Jacobi}	$\lambda_{\text{Armadillo}}$	Diff
0.2679	0.2679	0.0
1.0000	1.0000	0.0
2.0000	2.0000	0.0
3.0000	3.0000	0.0
3.7321	3.7321	0.0

Table 3 shows a comparison of runtimes of the Armadillo eigensystem-solver and our diagonalisation, for a 50x50-matrix.

Jacobi runtime	Armadillo runtime	Diff
0.090051s	0.000405s	0.089646s

5 Conclusions, perspectives

6 Appendix, extra material

7 Bibliography