FYS4150 - Computational Physics - Project 2

Eimund Smestad
Email: eimundsm@fys.uio.no
16th September, 2014

Abstract

This report looks at different algorithms to solve the one-dimensional Poisson's equation with Dirichlet boundary condition. The algorithms are compared by speed and floating point error in the solution. The results show how different ways of writing code effect speed and floating point error, and discusses how the machines architecture and functionally are the reason for the performance difference.

1 Theory

Side 24

$$j\hbar\frac{\partial\Psi}{\partial t} = -\frac{\hbar^2}{2m}\nabla^2\Psi + V\Psi$$

$$\Psi\left(\vec{x},t\right) = \psi\left(\vec{x}\right)\varphi\left(t\right)$$

$$-\frac{\hbar^2}{2m}\frac{\mathrm{d}^2\psi}{\mathrm{d}x^2} + V\psi = E\psi$$

Side 133

$$\nabla^2 = \frac{1}{r^2}\frac{\partial}{\partial r}\left(r^2\frac{\partial}{\partial r}\right) + \frac{1}{r^2\sin\theta}\frac{\partial}{\partial\theta}\left(\sin\theta\frac{\partial}{\partial\theta}\right) + \frac{1}{r^2\sin^2\theta}\frac{\partial^2}{\partial\phi^2}$$

$$-\frac{\hbar^2}{2m}\left[\frac{1}{r^2}\frac{\partial}{\partial r}\left(r^2\frac{\partial\psi}{\partial r}\right) + \frac{1}{r^2\sin\theta}\frac{\partial}{\partial\theta}\left(\sin\theta\frac{\partial\psi}{\partial\theta}\right) + \frac{1}{r^2\sin^2\theta}\frac{\partial^2\psi}{\partial\phi}\right] + V\psi = E\psi$$

$$\psi\left(r,\theta,\phi\right) = R\left(r\right)Y\left(\theta,\phi\right) + C$$

$$\frac{1}{R}\frac{\mathrm{d}}{\mathrm{d}r}\left(r^2\frac{\mathrm{d}R}{\mathrm{d}r}\right) - \frac{2mr^2}{\hbar^2}\left(V-E\right) + \frac{1}{Y}\left(\frac{1}{\sin\theta}\frac{\partial}{\partial\theta}\left(\sin\theta\frac{\partial Y}{\partial\theta}\right) + \frac{1}{\sin^2\theta}\frac{\partial^2 Y}{\partial\phi^2}\right) = \frac{2mr^2C}{RY\hbar^2}\left(V-E\right)$$

2 Eigenvalue algorithms

2.1 Diagonalize by matrix similarity

If we have a eigenvalue problem for a matrix $\mathbb{A}$ with eigenvector $\mathbf{x}$ and eigenvalue λ

$$\mathbf{A}\mathbf{x} = \lambda\mathbf{x}\,,$$

we can transform it by a basis matrix $\mathbf{S}$ to a similar matrix $\mathbf{B} = \mathbf{S}^{-1}\mathbf{A}\mathbf{S}$ with the same eigenvalue λ but with a different eigenvector $\mathbf{S}^{-1}\mathbf{x}$;

$$\lambda\mathbf{S}^{-1}\mathbf{x} = \mathbf{S}^{-1}\mathbf{A}\mathbf{x} = \mathbf{S}^{-1}\mathbf{A}\mathbf{S}\mathbf{S}^{-1}\mathbf{x} = \mathbf{B}\mathbf{S}^{-1}\mathbf{x}\,.$$

If we are able to diagonalize out matrix $\mathbf{A}$ the eigenvalue problem becomes trivial to solve. However to directly diagonalize matrix $\mathbf{A}$ of size $n \times n$ requires to solve a n-th polynomial problem, which is a very hard task. So we choose a easier way by iteratively zero out more and more non-diagonal elements of matrix $\mathbf{A}$, which will eventually give us the diagonal matrix

$$\mathbf{D} = \prod_i \mathbf{S}_i^{-1}\mathbf{A}\prod_i \mathbf{S}_i\,.$$

To make the problem simple to solve we define a basis matrix for transformation as such

$$\mathbf{S} = \begin{bmatrix} 1 & 0 & \cdots & 0 & \cdots & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 & \cdots & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & s_{ii} & \cdots & s_{ij} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & s_{ji} & \cdots & s_{jj} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 & \cdots & 0 & \cdots & 1 \end{bmatrix}, \tag{1}$$

where the strategy is for this basis matrix $\mathbf{S}$ will make $b_{ij} = b_{ji} = 0$ when we do a matrix similarity transformation of $\mathbf{A}$ to $\mathbf{B}$. The matrix elements of $\mathbf{S}^{-1}$ is then given by

$$s^{-}_{k\ell} = \begin{cases} \frac{s_{jj}}{s_{ii}s_{jj}-s_{ij}s_{ji}} & \text{for } k = \ell = i \\ -\frac{s_{ij}}{s_{ii}s_{jj}-s_{ij}s_{ji}} & \text{for } k = i \text{ and } \ell = j \\ -\frac{s_{ji}}{s_{ii}s_{jj}-s_{ij}s_{ji}} & \text{for } k = j \text{ and } \ell = i \\ \frac{s_{ii}}{s_{ii}s_{jj}-s_{ij}s_{ji}} & \text{for } k = \ell = j \\ 1 & \text{for } k = \ell \in \mathbb{N}_1^n / \left\{i, j\right\} \\ 0 & \text{elsewhere.} \end{cases}$$

To see how this matrix similarity transformation behaves we must do the cumbersome matrix multiplication of the transformation;

$$\mathbf{B} = \mathbf{S}^{-1}\mathbf{A}\mathbf{S}$$

$$= \begin{bmatrix} 1 & 0 & \cdots & 0 & \cdots & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 & \cdots & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & s_{ii}^{-} & \cdots & s_{ij}^{-} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & s_{ji}^{-} & \cdots & s_{jj}^{-} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 & \cdots & 0 & \cdots & 1 \end{bmatrix} \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1i} & \cdots & a_{1j} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2i} & \cdots & a_{2j} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots & \ddots & \vdots \\ a_{i1} & a_{i2} & \cdots & a_{ii} & \cdots & a_{ij} & \cdots & a_{in} \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots & \ddots & \vdots \\ a_{j1} & a_{j2} & \cdots & a_{ji} & \cdots & a_{jj} & \cdots & a_{jn} \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{ni} & \cdots & a_{nj} & \cdots & a_{nn} \end{bmatrix} \mathbf{S}$$

$$= \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1i} & \cdots & a_{1j} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2i} & \cdots & a_{2j} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots & \ddots & \vdots \\ s_{ii}^{-}a_{i1} + s_{ij}^{-}a_{j1} & s_{ii}^{-}a_{i2} + s_{ij}^{-}a_{j2} & \cdots & s_{ii}^{-}a_{ii} + s_{ij}^{-}a_{ji} & \cdots & s_{ii}^{-}a_{ij} + s_{ij}^{-}a_{jj} & \cdots & s_{ii}^{-}a_{in} + s_{ij}^{-}a_{jn} \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots & \ddots & \vdots \\ s_{ji}^{-}a_{i1} + s_{jj}^{-}a_{j1} & s_{ji}^{-}a_{i2} + s_{jj}^{-}a_{j2} & \cdots & s_{ji}^{-}a_{ii} + s_{jj}^{-}a_{ji} & \cdots & s_{ji}^{-}a_{ij} + s_{jj}^{-}a_{jj} & \cdots & s_{ji}^{-}a_{in} + s_{jj}^{-}a_{jn} \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{ni} & \cdots & a_{nj} & \cdots & a_{nn} \end{bmatrix} \mathbf{S}$$

$$= \begin{bmatrix} a_{11}^{-} & a_{12}^{-} & \cdots & a_{1i}^{-} & \cdots & a_{1j}^{-} & \cdots & a_{1n}^{-} \\ a_{21}^{-} & a_{22}^{-} & \cdots & a_{2i}^{-} & \cdots & a_{2j}^{-} & \cdots & a_{2n}^{-} \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots & \ddots & \vdots \\ a_{i1}^{-} & a_{i2}^{-} & \cdots & a_{ii}^{-} & \cdots & a_{ij}^{-} & \cdots & a_{in}^{-} \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots & \ddots & \vdots \\ a_{j1}^{-} & a_{j2}^{-} & \cdots & a_{ji}^{-} & \cdots & a_{jj}^{-} & \cdots & a_{jn}^{-} \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots & \ddots & \vdots \\ a_{n1}^{-} & a_{n2}^{-} & \cdots & a_{ni}^{-} & \cdots & a_{nj}^{-} & \cdots & a_{nn} \end{bmatrix} \begin{bmatrix} 1 & 0 & \cdots & 0 & \cdots & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 & \cdots & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & s_{ii} & \cdots & s_{ij} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & s_{ji} & \cdots & s_{jj} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 & \cdots & 0 & \cdots & 1 \end{bmatrix}$$

$$= \begin{bmatrix} a_{11}^{-} & a_{12}^{-} & \cdots & a_{1i}^{-}s_{ii} + a_{1j}^{-}s_{ji} & \cdots & a_{1i}^{-}s_{ij} + a_{1j}^{-}s_{jj} & \cdots & a_{1n}^{-} \\ a_{21}^{-} & a_{22}^{-} & \cdots & a_{2i}^{-}s_{ii} + a_{2j}^{-}s_{ji} & \cdots & a_{2i}^{-}s_{ij} + a_{2j}^{-}s_{jj} & \cdots & a_{2n}^{-} \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots & \ddots & \vdots \\ a_{i1}^{-} & a_{i2}^{-} & \cdots & a_{ii}^{-}s_{ii} + a_{ij}^{-}s_{ji} & \cdots & a_{ii}^{-}s_{ij} + a_{ij}^{-}s_{jj} & \cdots & a_{in}^{-} \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots & \ddots & \vdots \\ a_{j1}^{-} & a_{j2}^{-} & \cdots & a_{ji}^{-}s_{ii} + a_{jj}^{-}s_{ji} & \cdots & a_{ji}^{-}s_{ij} + a_{jj}^{-}s_{jj} & \cdots & a_{jn}^{-} \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots & \ddots & \vdots \\ a_{n1}^{-} & a_{n2}^{-} & \cdots & a_{ni}^{-}s_{ii} + a_{nj}^{-}s_{ji} & \cdots & a_{ni}^{-}s_{ij} + a_{nj}^{-}s_{jj} & \cdots & a_{nn}^{-} \end{bmatrix},$$

which gives us the matrix elements of the similarity matrix $\mathbf{B}$

$$b_{k\ell} = \begin{cases} \bar{s_{ii}} a_{ii} s_{ii} + \bar{s_{ij}} a_{ji} s_{ii} + \bar{s_{ii}} a_{ij} s_{ji} + \bar{s_{ij}} a_{jj} s_{ji} & \text{for } k = \ell = i \\ \bar{s_{ii}} a_{ii} s_{ij} + \bar{s_{ij}} a_{ji} s_{ij} + \bar{s_{ii}} a_{ij} s_{jj} + \bar{s_{ij}} a_{jj} s_{jj} & \text{for } k = i \text{ and } \ell = j \\ \bar{s_{ji}} a_{ii} s_{ii} + \bar{s_{jj}} a_{ji} s_{ii} + \bar{s_{ji}} a_{ij} s_{ji} + \bar{s_{jj}} a_{jj} s_{ji} & \text{for } k = j \text{ and } \ell = i \\ \bar{s_{ji}} a_{ii} s_{ij} + \bar{s_{jj}} a_{ji} s_{ij} + \bar{s_{ji}} a_{ij} s_{jj} + \bar{s_{jj}} a_{jj} s_{jj} & \text{for } k = \ell = j \\ \bar{s_{ki}} a_{i\ell} + \bar{s_{kj}} a_{j\ell} & \text{for } k \in \{i, j\} \text{ and } \ell \in \mathbb{N}_1^n / \{i, j\} \\ a_{ki} s_{i\ell} + a_{kj} s_{j\ell} & \text{for } \ell \in \{i, j\} \text{ and } k \in \mathbb{N}_1^n / \{i, j\} \\ a_{k\ell} & \text{elsewhere.} \end{cases} \tag{2}$$

What we want our matrix similarity transformation to do is to set $b_{ij} = b_{ji} = 0$, but we also need to ensure that the basis matrix $\mathbf{S}$ for the transformation is invertible. Hence we have the following three equation we want to solve

$$s_{ii} s_{jj} - s_{ij} s_{ji} = 1 \tag{3}$$

$$a_{ji} {s_{ij}}^2 - \left(a_{ii} - a_{jj}\right) s_{jj} s_{ij} - a_{ij} {s_{jj}}^2 = 0 \tag{4}$$

$$a_{ij} {s_{ji}}^2 - \left(a_{jj} - a_{ii}\right) s_{ii} s_{ji} - a_{ji} {s_{ii}}^2 = 0\,. \tag{5}$$

If the matrix $\mathbf{A}$ is indeed diagonalizable then the above three equations above solvable. We further observer that we have four unknowns in the three equations, which means that we can choose one of them as we want, I suggest $s_{ii} = 1$. I will not show the solution here, but it involves in principle in two second order equations $ax^2 + bx + c = 0$ which has a well known analytical solution $x = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$. Be aware of that the coefficient a may be zero which would make it a first order equations to solve instead. If the coefficient $b = 0$ in addition to $a = 0$, then the matrix $\mathbf{A}$ is not diagonalizable.

There are on last thing we need to consider regarding the diagonalizing algorithm, and that is that the matrix elements b_{ix}, b_{jx}, b_{xi} and b_{xj} for all $x \in \mathbb{N}_1^n / \{i, j\}$ may change from zero to a value if we are not careful. Which may undermine the work we try to do when we diagonalize the matrix. What we see in (2) is if we lock k to i and let ℓ go through all possibilities $\ell \in \mathbb{N}_1^n / \{k\}$ exactly once, we ensure that the values we have zeroed out for $k = i$ does not change when we do successive transformation with the same $k = i$;

$$\begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nn} \end{bmatrix} \sim \begin{bmatrix} b_{11}^{(1)} & b_{12}^{(1)} & \cdots & 0 \\ b_{21}^{(1)} & b_{22}^{(1)} & \cdots & b_{2n}^{(1)} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & b_{n2}^{(1)} & \cdots & b_{nn}^{(1)} \end{bmatrix} \sim \begin{bmatrix} b_{11}^{(n-1)} & 0 & \cdots & 0 \\ 0 & b_{22}^{(n-1)} & \cdots & b_{2n}^{(n-1)} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & b_{n2}^{(n-1)} & \cdots & b_{nn}^{(n-1)} \end{bmatrix} \to \begin{bmatrix} a_{11} & 0 & \cdots & 0 \\ 0 & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & a_{n2} & \cdots & a_{nn} \end{bmatrix}$$

Note that we have now transformed such that $a_{ki} = a_{ii}\delta_{ik} = a_{ki}$, where δ_{ik} is the Kronecker delta function. Now we change indices to $i' \neq i$ and let $j \in \mathbb{N}_1^n / \{i, i'\}$, because we are done with the row and column i. What we now realize in (2) is that the elements of matrix $\mathbb{B}$ in the row or column i is given by $b_{i\ell} = a_{ii'} s_{i'\ell} + a_{ij} s_{j\ell} = 0$ because $k \notin \{i', j\}$ and $i' \neq i \neq j$ which means that $a_{ii'} = 0$ and $a_{ij} = 0$, and $b_{ki} = \bar{s_{ki'}} a_{i'i} + \bar{s_{kj}} a_{ji} = 0$ because $\ell \notin \{i', j\}$ and $i' \neq i \neq j$ which means that $a_{i'i} = 0$ and $a_{ji} = 0$. In other words the matrix elements in row or column i that was zeroed out does not change when we continue to zero out matrix elements on row and column i', which means that we can continue the iteration until the matrix $\mathbf{A}$ is completely diagonalized;

$$\begin{bmatrix} a_{11} & 0 & 0 & \cdots & 0 \\ 0 & a_{22} & a_{23} & \cdots & a_{2n} \\ 0 & a_{32} & a_{33} & \cdots & a_{3n} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & a_{n2} & a_{n3} & \cdots & a_{nn} \end{bmatrix} \sim \begin{bmatrix} b_{11}^{(1)} & 0 & 0 & \cdots & 0 \\ 0 & b_{22}^{(1)} & b_{23}^{(1)} & \cdots & 0 \\ 0 & b_{32}^{(1)} & b_{33}^{(1)} & \cdots & b_{3n}^{(1)} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & b_{n3}^{(1)} & \cdots & b_{nn}^{(1)} \end{bmatrix} \sim \begin{bmatrix} b_{11}^{(n-2)} & 0 & 0 & \cdots & 0 \\ 0 & b_{22}^{(n-2)} & 0 & \cdots & 0 \\ 0 & 0 & b_{33}^{(n-2)} & \cdots & b_{3n}^{(n-2)} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & b_{n3}^{(n-2)} & \cdots & b_{nn}^{(n-2)} \end{bmatrix}$$

$$\sim \begin{bmatrix} \lambda_1 & 0 & 0 & \cdots & 0 \\ 0 & \lambda_2 & 0 & \cdots & 0 \\ 0 & 0 & \lambda_3 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & \lambda_n \end{bmatrix}.$$

2.2 Jacobi's eigenvalue algorithm

The Jacobi's eigenvalue algorithm is a special case of diagonalize by matrix similarity algorithm discussed in section 2.1, where we use a two dimensional rotation matrix in n dimensional space

$$\mathbf{R}_{i,j,\theta} = \begin{bmatrix} 1 & 0 & \cdots & 0 & \cdots & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 & \cdots & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & r_{ii} = \cos\theta & \cdots & r_{ij} = -\sin\theta & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & r_{ji} = \sin\theta & \cdots & r_{jj} = \cos\theta & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 & \cdots & 0 & \cdots & 1 \end{bmatrix}, \tag{6}$$

which satisfies the basis matrix $\mathbf{S}$ in (1). The rotation matrix can easily be shown to be a orthogonal matrix by realizing $\mathbf{R}_{-\theta} = \mathbf{R}_{\theta}{}^{-1} = \mathbf{R}_{\theta}{}^{\mathrm{T}}$, because when we rotate an angle θ, we need to rotate an angle $-\theta$ to get back (or you could realize this by a more cumbersome way by actually do the matrix inversion).

By doing the matrix similarity transformation $\mathbf{B} = \mathbf{R}_{i,j,-\theta}\mathbf{A}\mathbf{R}_{i,j,\theta}$, we get the following similar matrix elements according to (2)

$$b_{k\ell} = \begin{cases} a_{ii}\cos^2\theta + a_{jj}\sin^2\theta + \left(a_{ij} + a_{ji}\right)\sin\theta\cos\theta & \text{for } k = \ell = i \\ a_{ij}\cos^2\theta - a_{ji}\sin^2\theta + \left(a_{jj} - a_{ii}\right)\sin\theta\cos\theta & \text{for } k = i \text{ and } \ell = j \\ a_{ji}\cos^2\theta - a_{ij}\sin^2\theta + \left(a_{jj} - a_{ii}\right)\sin\theta\cos\theta & \text{for } k = j \text{ and } \ell = i \\ a_{jj}\cos^2\theta + a_{ii}\sin^2\theta - \left(a_{ij} + a_{ji}\right)\sin\theta\cos\theta & \text{for } k = \ell = j \\ a_{i\ell}\cos\theta + a_{j\ell}\sin\theta & \text{for } k = i \text{ and } \ell \in \mathbb{N}_1^n / \left\{i, j\right\} \\ a_{j\ell}\cos\theta - a_{i\ell}\sin\theta & \text{for } k = j \text{ and } \ell \in \mathbb{N}_1^n / \left\{i, j\right\} \\ a_{ki}\cos\theta + a_{kj}\sin\theta & \text{for } \ell = i \text{ and } k \in \mathbb{N}_1^n / \left\{i, j\right\} \\ a_{kj}\cos\theta - a_{ki}\sin\theta & \text{for } \ell = j \text{ and } k \in \mathbb{N}_1^n / \left\{i, j\right\} \\ a_{k\ell} & \text{elsewhere.} \end{cases}$$

$$b_{k\ell} = \begin{cases} a_{ii}\cos^2\theta + a_{jj}\sin^2\theta + \left(a_{ij} + a_{ji}\right)\sin\theta\cos\theta & \text{for } k = \ell = i \\ a_{ij}\cos^2\theta - a_{ji}\sin^2\theta + \left(a_{jj} - a_{ii}\right)\sin\theta\cos\theta & \text{for } k = i \text{ and } \ell = j \\ a_{ji}\cos^2\theta - a_{ij}\sin^2\theta + \left(a_{jj} - a_{ii}\right)\sin\theta\cos\theta & \text{for } k = j \text{ and } \ell = i \\ a_{jj}\cos^2\theta + a_{ii}\sin^2\theta - \left(a_{ij} + a_{ji}\right)\sin\theta\cos\theta & \text{for } k = \ell = j \\ a_{i\ell}\cos\theta + a_{j\ell}\sin\theta & \text{for } k = i \text{ and } \ell \in \mathbb{N}_1^n / \left\{i, j\right\} \\ a_{j\ell}\cos\theta - a_{i\ell}\sin\theta & \text{for } k = j \text{ and } \ell \in \mathbb{N}_1^n / \left\{i, j\right\} \\ a_{ki}\cos\theta + a_{kj}\sin\theta & \text{for } \ell = i \text{ and } k \in \mathbb{N}_1^n / \left\{i, j\right\} \\ a_{kj}\cos\theta - a_{ki}\sin\theta & \text{for } \ell = j \text{ and } k \in \mathbb{N}_1^n / \left\{i, j\right\} \\ a_{k\ell} & \text{elsewhere.} \end{cases}$$

The strategy is to set b_{ij} and b_{ji} to zero, which enables us by iteration to zero out every element in the matrix except the diagonal elements (as described in section 2.1). Unfortunately we have only one variable, namely θ, which makes it impossible to diagonalize a general matrix $\mathbf{A}$ by similarity. However when $\mathbf{A}$ is a symmetric matrix, $a_{ij} = a_{ji}$, which gives us only one equation to solve to set both b_{ij} and b_{ji} to zero. For a symmetric matrix $\mathbf{A}$ we have the following elements of the similarity matrix $\mathbf{B}$

$$b_{k\ell} = \begin{cases} a_{ii}\cos^2\theta + a_{jj}\sin^2\theta + 2a_{ij}\sin\theta\cos\theta & \text{for } k = \ell = i \\ a_{ij}\left(\cos^2\theta - \sin^2\theta\right) + \left(a_{jj} - a_{ii}\right)\sin\theta\cos\theta & \text{for } k = i \text{ and } \ell = j, \text{ or } k = j \text{ and } \ell = i \\ a_{jj}\cos^2\theta + a_{ii}\sin^2\theta - 2a_{ij}\sin\theta\cos\theta & \text{for } k = \ell = j \\ a_{i\ell}\cos\theta + a_{j\ell}\sin\theta & \text{for } k = i \text{ and } \ell \in \mathbb{N}_1^n / \left\{i, j\right\} \\ a_{j\ell}\cos\theta - a_{i\ell}\sin\theta & \text{for } k = j \text{ and } \ell \in \mathbb{N}_1^n / \left\{i, j\right\} \\ a_{ki}\cos\theta + a_{kj}\sin\theta & \text{for } \ell = i \text{ and } k \in \mathbb{N}_1^n / \left\{i, j\right\} \\ a_{kj}\cos\theta - a_{ki}\sin\theta & \text{for } \ell = j \text{ and } k \in \mathbb{N}_1^n / \left\{i, j\right\} \\ a_{k\ell} & \text{elsewhere.} \end{cases} \tag{7}$$

Now we are able to solve the following equation

$$b_{ij} = b_{ji} = a_{ij}\left(\cos^2\theta - \sin^2\theta\right) + \left(a_{jj} - a_{ii}\right)\sin\theta\cos\theta = 0\,,$$

and by using the trigonometric relation $\cos^2\theta + \sin^2\theta = 1$ we can rewrite to

$$\cos^4\theta - \cos^2\theta + \frac{{a_{ij}}^2}{\left(a_{ii} - a_{jj}\right)^2 + 4{a_{ij}}^2} = 1\,.$$

By solving this as a second order equation with regard to $\cos^2\theta$ and using the trigonometric relation $\cos^2\theta + \sin^2\theta = 1$ again, we get

$$\theta = \arccos\sqrt{\frac{1}{2}\left(1 + \sqrt{\frac{\left(a_{ii} - a_{jj}\right)^2}{\left(a_{ii} - a_{jj}\right)^2 + 4{a_{ij}}^2}}\right)} = \arcsin\sqrt{\frac{1}{2}\left(1 - \sqrt{\frac{\left(a_{ii} - a_{jj}\right)^2}{\left(a_{ii} - a_{jj}\right)^2 + 4{a_{ij}}^2}}\right)}. \tag{8}$$

We can diagonalize the matrix $\mathbf{A}$ by doing matrix similarity transformation iteratively as described in section 2.1. However we can optimize Jacobi's eigenvalue algorithm by using the Frobenius norm.

3 Numerical implementation

We can optimize the calculation of diagonal elements in (7) by rewriting them by using the trigonometrical relation $\cos^2\theta + \sin^2\theta = 1$;

$$\begin{aligned} b_{ii} &= a_{ii}\cos^2\theta + a_{jj}\sin^2\theta + 2a_{ij}\sin\theta\cos\theta = a_{ii}\left(1-\sin^2\theta\right) + a_{jj}\sin^2\theta + 2a_{ij}\sin\theta\cos\theta \\ &= a_{ii} + \left[\left(a_{jj}-a_{ii}\right)\sin^2\theta + 2a_{ij}\sin\theta\cos\theta\right] \end{aligned} \tag{9}$$

$$\begin{aligned} b_{jj} &= a_{jj}\cos^2\theta + a_{ii}\sin^2\theta - 2a_{ij}\sin\theta\cos\theta = a_{jj}\left(1-\sin^2\theta\right) + a_{ii}\sin^2\theta - 2a_{ij}\sin\theta\cos\theta \\ &= a_{jj} - \left[\left(a_{jj}-a_{ii}\right)\sin^2\theta + 2a_{ij}\sin\theta\cos\theta\right] \end{aligned} \tag{10}$$

Note that it's better to use $\sin^2\theta$ instead of $\cos^2\theta$, because when we look at

$$\begin{aligned} b_{ii} &= a_{jj} + \left[\left(a_{ii}-a_{jj}\right)\cos^2\theta + 2a_{ij}\cos\theta\sin\theta\right] \\ b_{jj} &= a_{ii} - \left[\left(a_{ii}-a_{jj}\right)\cos^2\theta + 2a_{ij}\cos\theta\sin\theta\right], \end{aligned}$$

we see that a_{jj} corresponds to b_{ii} and a_{ii} corresponds to b_{jj}, which means that we need to backup either a_{ii} or a_{jj} before we do the calculation, so we don't overwrite the value before we have used in the calculation. However for the $\sin^2\theta$ expressions we have that a_{ii} corresponds to b_{ii} and a_{jj} corresponds to b_{jj}, which means that we don't need to backup a_{ii} or a_{jj}, and we can use the program operator += directly on the diagonal elements.

4 Attachments

The files produced in working with this project can be found at
https://github.com/Eimund/UiO/tree/master/FYS4150/Project%202

The source files developed are

5 Resources

1. QT Creator 5.3.1 with C11
2. Armadillo
3. Eclipse Standard/SDK - Version: Luna Release (4.4.0) with PyDev for Python
4. Ubuntu 14.04.1 LTS
5. ThinkPad W540 P/N: 20BG0042MN with 32 GB RAM

References

[1] Morten Hjorth-Jensen, *FYS4130 - Project 2 - Schrödinger's equation for two electrons in a three-dimensional harmonic oscillator well*, University of Oslo, 2014

[2] Morten Hjorth-Jensen, *Computational Physics - Lecture Notes Fall 2014*, University of Oslo, 2014

[3] http://en.wikipedia.org/wiki/Eigenvalues_and_eigenvectors

[4] http://en.wikipedia.org/wiki/Jacobi_eigenvalue_algorithm

[5] http://en.wikipedia.org/wiki/Matrix_similarity

[6] http://en.wikipedia.org/wiki/Rotation_matrix

[7] http://en.wikipedia.org/wiki/Orthogonal_matrix

[8] http://en.wikipedia.org/wiki/Matrix_norm#Frobenius_norm