Project 2 - Computational physics - FYS3150

Department of Physics, University of Oslo, Norway.
Philip Niane og Rohullah Akbari
Link til githubmappen:
https://github.com/philipkarim/Philip-and-Rohullah-ComFys

Abstrakt

I denne rapporten er forskjellige egenverdiproblemer løst numerisk ved bruk av jacobis metode og armadillo (som viste seg å være lettere å bruke.). Det er tatt utgangspunkt i Schroedingers likning for et elektron uten potensial, et elektron med potensial og to elektroner (både potensial og frastøtningskrefter). Ut ifra resultatene vil frastøtningskreftene dominere når frekvensen er liten, og potensialet dominere når frekvensen er stor.

Contents

1 Introduksjon

Målet med dette prosjektet er å lære å utvikle programmeringskoder som løser egenverdiproblemer. Metoden som skal utvikles er Jacobis metode. Deretter er målet å finne egenverdiene ved bruk av armadillo. Deretter følger naturligvis en sammenlikning mellom de to løsningsmetodene. Matrisen som skal brukes er samme matrise som fra prosjekt 1, en tridiagonal Toeplitz matrise. Etter disse er kodet og klare, skal begge løsningsmetodene brukes til å løse egenverdiproblemer for schrödingers likning med hensyn til ett elektron(kun potensial) og to elektroner(potensial og frastøtningskrefter). Prosjektet gir også trening i å utføre "unixtests" for å sjekke at alt er greit med algoritmene underveis i arbeidsøkten.

2 Teori

2.1 Jacobis metode

Algoritmen som er et av hovedverktøyene våre dette prosjektet er som sagt Jacobis metode. Jacobis metode er en metode som kan brukes på relle symmetriske matriser. Metoden går ut på å diagonalisere en matrise ved å multiplisere matrisen og utføre en rekke med såkalte "similarity transformations" helt til elementene utenfor diagonalen blir lik 0. Siden matrisen vår blir diagonalisert vil elementene våre langs diagonalen være lik de forskjellige egenverdiene. Se appendiks for bevis for at indreproduktet og ortogonaliteten er bevart når vi utfører similarity transformations.[1][2]

2.2 Schrodinger likning

Likningen som skal tas i bruk er som sagt Scrodingers likning. Scrodingers likning kan blandt annet brukes til å regne ut energier til elektroner. Grunnen til at denne kan brukes, er fordi som vist i project 1 kan likninger på denne formen skrives om til matriseproblemer.
Schrodingers likning ser hovedsaklig slik ut:

$$\hat{H}\Psi = E\Psi \tag{1}$$

Denne kan videre skrives som en radial likning [3];

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

Potensialet V(r) tilsvarer det harmoniske oscillator potensialet $(1/2)kr^2$ der $k = m\omega^2$, og ω er $2\pi\nu$, altså et uttrykk for frekvensen. Energien kan skrives som følgende:

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

Der vi vet fra kvantefysikk at kvantetallene n og l varierer i intervallet 0,1,2...

Vi bytter ut $R(r)$ med $(1/r)u(r)$ og får følgende uttrykk.

$$-\frac{\hbar^2}{2m}\frac{d^2}{dr^2}u(r) + \left(V(r) + \frac{l(l+1)}{r^2}\frac{\hbar^2}{2m}\right)u(r) = Eu(r). \tag{4}$$

Med grenseverdier u(0)=u(∞)=0.
Scrodingers likning kan deretter utledes til følgende. (Utledningen kan ses i appendiks)

$$-\frac{d^2}{d\rho^2}u(\rho) + \rho^2 u(\rho) = \lambda u(\rho). \tag{5}$$

Hvor en passende maksimums ρ må prøves fram til med tanke på at $\rho = \infty$ blir vanskelig å bruke.

Som det ble vist i prosjekt 1, kan en differensiallikning på denne formen skrives som et matrise problem [4]. Schroedingers likning kan altså skrives på følgende form:

$$\begin{bmatrix} d_1 & a_1 & 0 & 0 & \dots & 0 & 0 \\ a_1 & d_2 & a_2 & 0 & \dots & 0 & 0 \\ 0 & a_2 & a_3 & a_3 & 0 & \dots & 0 \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ 0 & \dots & \dots & \dots & \dots a_{N-3} & d_{N-2} & a_{N-2} \\ 0 & \dots & \dots & \dots & \dots & a_{N-2} & d_{N-1} \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \\ \dots \\ \dots \\ \dots \\ u_{N-1} \end{bmatrix} = \lambda \begin{bmatrix} u_1 \\ u_2 \\ \dots \\ \dots \\ \dots \\ u_{N-1} \end{bmatrix}. \tag{6}$$

Hvor d og a er som følger:

$$d_i = \frac{2}{h^2} + V_i \tag{7}$$

og

$$a_i = -\frac{1}{h^2}. \tag{8}$$

Der steglengden $h = \frac{\rho_N - \rho_0}{N}$. Siden $d_i = d_{i+1} = d_{i+2}$ og $a_i = a_{i+1} = a_{i+2}$ i denne problemstillingen jobbes det videre med følgende likning i prosjektet:

$$d_i u_i + e_i u_{i-1} + e_i u_{i+1} = \lambda u_i \tag{9}$$

3 Metode

3.1 Jacobis algoritme

Vi har brukt Jacobis algoritme, til å løse problemstillingne vår, som er storsett inpspirert fra Lecture Notes. Bruker funksjonen $maxoffdiag()$ først til å finne

den største ikke-diagonale matrise elementet. Deretter kjøres denne while-løkken

$$while(fabs(max_offdiag) > epsilon(double) iterations < max_number_iterations)$$

Altså sjekker den om det elementet er større en toleranse, ϵ, og om antall iterasjoner er mindre enn den største interasjonen. Dersom disse kravene er oppfylt så utfører algoritmen $rotate()$-funksjonen, altså en rotasjon på den største ikke-diagonale matrise og eliminerer det. Samtidig så blir $iterations$ økt med 1. Deretter så finner algoritmen ett nytt størst ikke-diagonal element og utfører rotasjon på det. Når programmet har kjørt gjennom alle elementene så vil den største ikke-diagonale elementet vil være mindre enn ϵ og algoritmen vil stoppe.

3.1.1 Ett elektron-harmonisk potensiale

Her anvender vi algoritmen ovenfor til å finne energiene til et elektron i ett harmonisk potensiale. Vi starter med å fylle matrisen ved funksjonen $fylleMatriseApotensial()$. På diagonalen skal det stå d-ene, uttrykk (7), der vi setter inn potensialet for ett elektron for V_i. Og på ikke diagonalen skal det stå, uttrykk (8), a-ene.

3.1.2 To elektroner-harmonisk potensiale og frastøtningskrefter

I denne delen utvider vi algoritmen fra 3.1.1, og i V_i setter vi inn uttrykket for potensial og frastøtningskrefter.

3.2 Unit tests

For å teste at kodene og algoritmene fungerer som de skal er det utført såkalte unit tests underveis i arbeidet. Målet med unit tests er å forsikre seg om at funksjonene man skriver faktisk fungerer som de skal, ved å teste på matriser vi vet svaret på.

Første test var en test som sjekker at funksjonen for å finne maksimumselementet i en matrise utenfor diagonalen. For å teste dette ble en symmetrisk toeplitz matrise (5x5) valgt. Matrisens høyeste element utenfor diagonalen var 5. Funksjonen ble kjørt på matrisen og maksimumsverdien 5 spratt ut. Test 1: bestått!

Neste test gikk ut på å teste at de riktige egenverdiene ble funnet. Det ble klart fra test 1 at maksimumsfunksjonen fungerte som den skal. Altså er det noe feil med rotasjons algoritmen om feil skulle oppstå.
Den samme matrisen fra test 1 ble brukt videre. De korrekte egenverdiene ble lagret i en array i stigende rekkefølge. Videre ble programmet kjørt der egenverdiene ble lagret langs diagonalen i matrisen. en for-loop ble brukt til å fylle en array med egenverdiene. Etter egenverdiene ble fylt i arrayen ble to for-loops brukt til å sortere arrayen i stigende rekkefølge. Til slutt ble egenverdiene sjekket ved å ha en sette absoluttverdien mellom de tilsvarende egenverdiene mindre enn epislon=1e-8. Programmet kjører og dunker ut "Test 2 bestått!".

4 Resultater

4.1 Algoritmetid

Det finnes naturligvis flere måter å regne ut egenverdier på. Derfor ble tiden det tok å kjøre jacobialgortimen tatt for forskjellige verdier av N. Det ble også gjort for armadillos egenverdifunksjon. Dette ble gjort for N=50,100,150 og 200. Resultatet kan ses i figur 1.

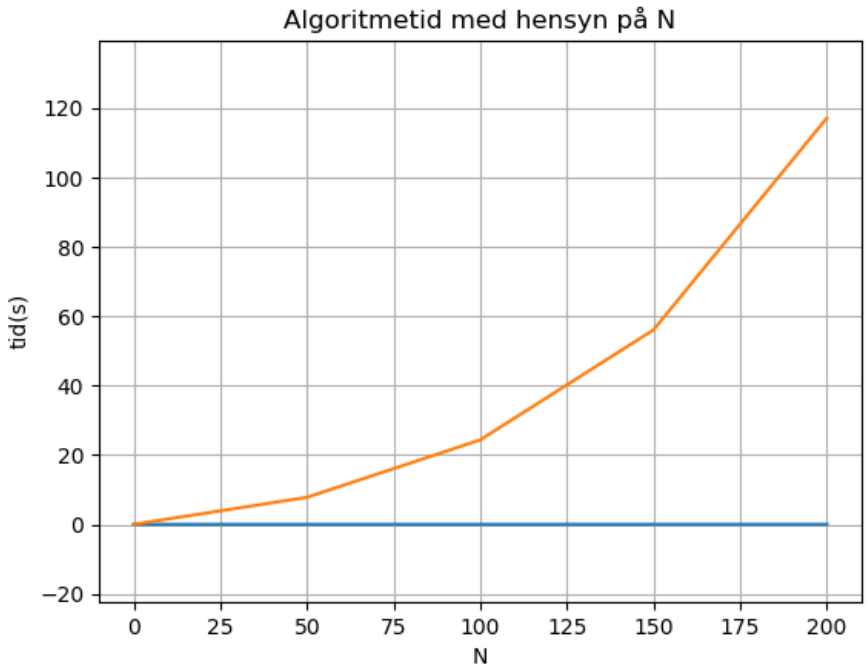

Figure 1: Algoritmetid med hensyn på N. Blå linje tilsvarer tiden det tar å utføre utregningene ved bruk av armadillo, mens den oransje linjen tilsvarer tiden det tar å utføre utregningene for Jacobis algortime

4.2 Feil med hensyn N og ρ_{max}

Med utgangspunkt for ett elektron i et harmonisk potensiale ble noen plott fremstilt der den relative feilen for de første 4 egenverdiene ble plottet mot forskjellige verdier av ρ. Det ble produsert tre plott der N varierer for å se hvordan N varierer. Plottene kan ses i figur 2-4.

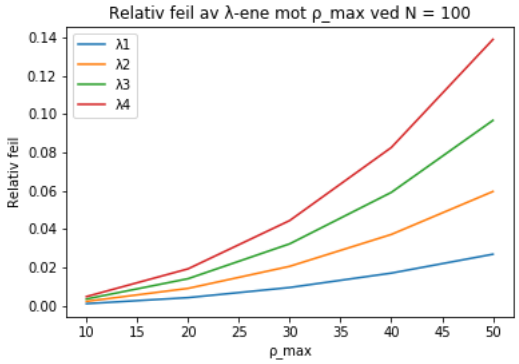

Figure 2: N=100. Den relative feilen for de fire første egenverdiene plottet mot forskjellige verdier av ρ_{max} der N=100

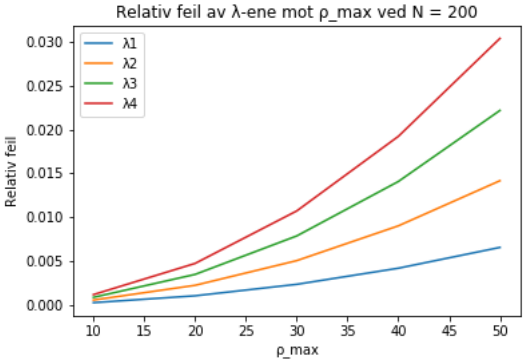

Figure 3: N=200. Den relative feilen for de fire første egenverdiene plottet mot forskjellige verdier av ρ_{max} der N=200

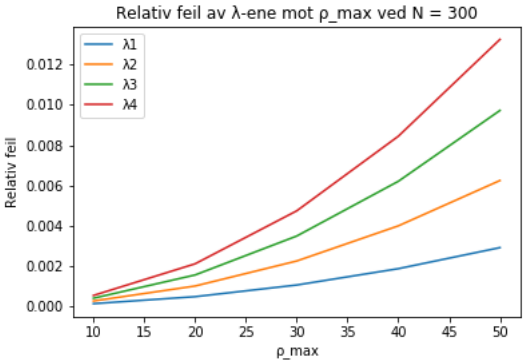

Figure 4: N=300. Den relative feilen for de fire første egenverdiene plottet mot forskjellige verdier av ρ_{max} der N=300.

4.3 To elektroner i et harmonisk potensiale

I denne delen ble analysert to elektroner i et harmonisk potensialet. Vi så på sannsynlighetstettheten for disse elektronene over distane ρ. Her ble det antatt at det er ingen interaksjon mellom elektronene.

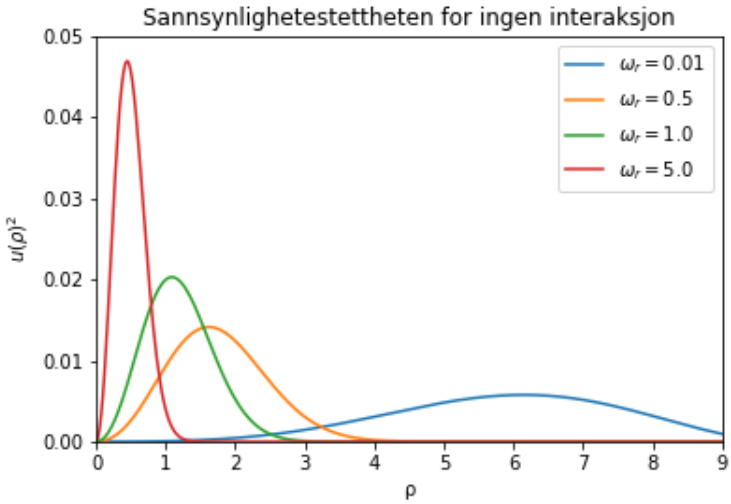

Figure 5: Sannsynlighetstettheten for ingen interaksjon mellom to elektroner i et harmonisk potensial.

5 Diskusjon

Diagonalisering av en matrise ved bruk av Jacobi metoden er avhengig av hva slags matrise det er. Jacobi metoden vil ikke fungere for alle typer matriser, spesielt for matriser med mange nullere i seg. Etter å ha kjørt denne algoritmen ett par ganger så vil det oppstå forskjellige verdier på elementene der det var null i starten.

Ved økende dimensjonen så øker tiden det tar å utføre algortimene. Ut ifra figur 1(plottet over algoritmetid) er det lett å se at Jacobis algoritme er svært tidskrevende iforhold til armadillos utregning. Jacobis metode øker eksponensielt med tanke på tid, mens armadillos metode nesten ikke øker i tid.

Det kan observeres, i figurene 2-4, at ved høyere ρ_{max} får vi større relativ feil. Dette skyldes at ved økende ρ_{max} får vi mindre verdier for diagonalen. Fra ligning 7 har vi

$$\begin{aligned} d_i &= \frac{2}{h^2} + V_i \\ &= \frac{2n^2}{\rho_{max}^2} + V_i \end{aligned}$$

siden $\rho_{min} = 0$. Her ser at ved økende ρ_{max} vil vi få mindre og mindre bidrag fra venstresiden og større bidraget fra potensialet. Dette fører til presisjonstap. Derfor får vi større relativ feil iforhold til de eksakte verdiene.

En annen parameteren som er med på å påvirke feilen er dimensjonen N. Ut ifra figurene 2-4 kan det observeres at økende N-verdier gir oss mindre feil. Ved økende N-verdier vil vi flere punkter i matrisen vår og alle løkkene våre vil kjøre ganger. Det gjør at vi får større presisjon. Problemet med større matriser er at dette krever mer datakraft og tar tid å kjøre, noe som ble vist i figur 1.

Fra figur 5 ovenfor kan se sannsynlighetstettheten for elektronene med ingen ineraksjon mellom dem. Vi ser på figuren at ved økende ω_r-verdier har vi høyere toppunktet for grafen. Altså når elektronene kommer nærmere hverandre så får vi smallere og smallere på breddren til grafene.

6 Konklusjon

I dette projektet analyserte og anvendte vi Jacobi metoden. Vi implementerte denne metoden for en helt generell tilfellet. Videre brukte det til å løse Shrodringer likning for to tilfeller. I det ene tilfellet så vi på elektron i et harmonisk tilstand. Og det andre tilfellet ble denne metoden anvendt på på to elektroner. Vi kom fram til at Jacobi metoden kan brukes på noen spesielle matrisen. Ulempen med denne metoden er at det er ikke alltid den diagonaliserer helt perfekt. Det vil være tilfeller der vi tall på de ikke-diagonale plassene i matrisen vår.

I den kvantemekaniske analysen så på hvilke faktorer som kunne påvirke den relative energien til elektronet i grunntilstanden. Vi kom fram til at ved mindre ρ_{max} og større N verdier vil vi mer presisjon.

7 Appendiks

7.1 Bevis: Indreprodukt og ortogonaliteten er bavart

En enhetlig transformasjon bevarer ortogonaliteten til de oppnådde egenvektorene. Vi ser på en basis $\vec{v}_i$,

$$\vec{v}_i = [v_{i1}, , , v_{in}] \tag{10}$$

Vi antar at basisen er ortogonal, altså

$$\vec{v}_j^T \vec{v}_i = \delta_{ij}. \tag{11}$$

Vi kan dermed bevise at en ortogonal eller en enhetlig transformasjon bevarer prikkproduktet og ortogonaliteten. Starter med

$$\vec{w}_i = \vec{U}\vec{v}_i \tag{12}$$

Antar at U er en symmerisk matrise slik at $\vec{U}^T = \vec{U}^{-1}$. Da har vi at:

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

$$= \vec{v}_j^T \vec{U}^T \vec{U}\vec{v}_i = \vec{v}_j^T \vec{U}^{-1}\vec{U}\vec{v}_i = \vec{v}_j^T I_n \vec{v}_i$$

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

Altså vi har at $\vec{w}_j^T \vec{w}_i = \delta_{ij}$ og vi har vist at en enhetlig transformasjon bevarer prikkproduktet og ortogonaliteten

7.2 Videre utledning av Shroedingers likning[3]

Det tas utgangspunkt i følgende versjon av schroedingers likning:

$$-\frac{\hbar^2}{2m}\frac{d^2}{dr^2}u(r) + \left(V(r) + \frac{l(l+1)}{r^2}\frac{\hbar^2}{2m}\right)u(r) = Eu(r). \tag{13}$$

Det første som gjøres er å introdusere variabelen $\rho = \frac{r}{\alpha}$ som naturligvis gir r=$\rho\alpha$. Radiusen r i liknignen fyller inn og gir følgende:

$$-\frac{\hbar^2}{2m\alpha^2}\frac{d^2}{d\rho^2}u(\rho) + \left(V(\rho) + \frac{l(l+1)}{\rho^2}\frac{\hbar^2}{2m\alpha^2}\right)u(\rho) = Eu(\rho). \tag{14}$$

Videre fylles det harmoniske oscillator potensialet $V(\rho) = (1/2)k\alpha^2\rho^2$ på i likningen samtidig som det antas at kvantetallet l=0 og gir følgende:

$$-\frac{\hbar^2}{2m\alpha^2}\frac{d^2}{d\rho^2}u(\rho)+\frac{k}{2}\alpha^2\rho^2u(\rho)=Eu(\rho). \tag{15}$$

Nå er det bare å multiplisere likningen med $2m\alpha^2/\hbar^2$. Deretter kan α bestemmes ved å sette faktoren foran som ikke er avhengig av ρ i det andre leddet lik 1. Dette gir $\alpha=\left(\frac{\hbar^2}{mk}\right)^{1/4}$. Delen foran energien settes lik egenverdien på følgende måte:

$$\lambda=\frac{2m\alpha^2}{\hbar^2}E \tag{16}$$

Som videre gir schroedinger likningen vi skulle utlede:

$$-\frac{d^2}{d\rho^2}u(\rho)+\rho^2u(\rho)=\lambda u(\rho). \tag{17}$$

8 Bibliografi

[1] Morten Hjorth-Jensen, august 2015, Computational Physics Lecture Notes, 6.4 Linear systems, (https://github.com/CompPhysics/ComputationalPhysics/blob/master/doc/Lectures/lectures2015.pdf)

[2] Jim Lambers, Jacobi Methods, Spring Quarter 2010-11, (https://web.stanford.edu/class/cme335/lecture7.pdf)

[3] Morten Hjorth-Jensen, september 2019 Project 2, https://github.com/CompPhysics/ComputationalPhysics/blob/master/doc/Projects/2019/Project2/pdf/Project2.pdf.

[4] Philip Niane og Rohullah Akbari, 2019 Project 1, https://github.com/philipkarim/Philip-and-Rohullah-ComFys/blob/master/Project1/Ferdigrapport.pdf

Link til githubmappen: https://github.com/philipkarim/Philip-and-Rohullah-ComFys