

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 egenverdi-problemer løst numerisk ved bruk av Jacobis metode og armadillo (som viste seg å være mye raskere enn Jacobis metode). Det er tatt utgangspunkt i Schrödingers likning for ett elektron uten harmonisk potensial, ett elektron med harmonisk potensial og to elektroner (både harmonisk potensial og frastøtningskrefter). Ut ifra resultatene vil for store ρ_{max} føre til større feil. Itillegg vil større frekvenser i et to-elektron problem (uten interaksjon) føre til at elektronene er å finne innenfor et kortere intervall.

Contents

1	Introduksjon	2
2	Teori	2
2.1	Jacobis metode	2
2.2	Schrödingers likning	2
3	Metode	4
3.1	Jacobis algoritme	4
3.2	Unit tests	4
4	Resultater	5
4.1	Algoritmetid	5
4.2	Feil med hensyn N og ρ_{max}	6
4.3	To elektroner i et harmonisk potensial	7
5	Diskusjon	8
6	Konklusjon	8
7	Appendiks	9
7.1	Bevis: Indreprodukt og ortogonaliteten er bevart	9
7.2	Videre utledning av Shroedingers likning[3]	9

1 Introduksjon

Målet med dette prosjektet er å lære å utvikle programmeringskoder som løser egenverdi-problemer. 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 egenverdi-problemer for Schrödingers likning med hensyn til ett elektron (kun harmonisk potensial) og to elektroner (harmonisk potensial og frastøttingskrefter). Prosjektet gir også trening i å utføre "unixtests" for å sjekke at alt er greit med algoritmene underveis i prosjektarbeidet.

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 reelle symmetriske matriser. Metoden går ut på å diagonalisere en matrise ved å multiplisere matrisen og utføre en rekke med "similarity transformations" helt til elementene utenfor diagonalen blir lik 0. Siden matrisen blir diagonalisert vil elementene langs diagonalen være lik de forskjellige egenverdiene. Se appendiks for bevis for at indreproduktet og ortogonaliteten er bevart når det utføres similarity transformations.[1][2]

2.2 Schrödingers likning

Likningen som skal tas i bruk er som sagt Schrödingers likning. Schrödingers 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.

Schrödingers likning ser hovedsaklig slik ut:

$$\hat{H}\Psi = E\Psi \quad (1)$$

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

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

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:

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

Der det er kjent fra kvantefysikk at kvantetallene n og l varierer i intervallet $0, 1, 2, \dots$

$R(r)$ byttes ut med $(1/r)u(r)$ og gir følgende uttrykk.

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

Med grenseverdier $u(0)=u(\infty)=0$.

Schrödingers likning kan deretter utledes til følgende: (Utleddningen kan ses i appendiks)

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

Hvor en passende maksimums ρ må prøves fram, 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]. Schrödingers likning kan altså skrives på følgende form:

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

Hvor d og a er som følger:

$$d_i = \frac{2}{h^2} + V_i \quad (7)$$

og

$$a_i = -\frac{1}{h^2}. \quad (8)$$

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:

$$d_i u_i + e_i u_{i-1} + e_i u_{i+1} = \lambda u_i \quad (9)$$

3 Metode

3.1 Jacobis algoritme

Jacobis algoritme er brukt til å løse problemstillingen, der koden stort sett er inspirert fra undervisningsnotater[1].

Når det gjelder psudokode for koden går denne ut på at funksjonen *maxoffdiag()* finner det største ikke-diagonale matrise elementet. Deretter kjøres en while-løkke som sjekker at elementet som ble funnet er større enn en toleranse, samtidig som det sjekkes at antall iterasjoner er mindre enn det største antallet iterasjoner. Dersom kravene er oppfylt fører dette videre til at funksjonen *rotate()* kjører. Denne funksjonen utfører en rotasjon på elementet som ble funnet i matrisen og eliminerer det. Samtidig øker iterationer med 1. Deretter finner algoritmen et nytt størst ikke-diagonal element og utfører rotasjonen igjen. Programmet kjører videre gjennom elementene utenfor diagonalen til det største elementet blir mindre enn toleransefaktoren. Da stopper programmet.

3.1.1 Ett elektron-harmonisk potensiale

Her anvendes algoritmen ovenfor til å finne energiene til ett elektron i et harmonisk potensiale. Matrisen fylles ved bruk av funksjonen *fyllMatriseApotensial()*. Denne gjør at matrisen som skal gjennom Jacobialgoritmen fylles med uttrykk (7) langs diagonalen, mens langs utenom diagonalen fylles det på med uttrykk (8).

3.1.2 To elektroner-harmonisk potensiale og frastøtningskrefter

I denne delen utvides algoritmen fra 3.1.1, og for V_i fylles uttrykket for et harmonisk 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 $\epsilon=1e-8$. 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 jacobialgoritmen 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.

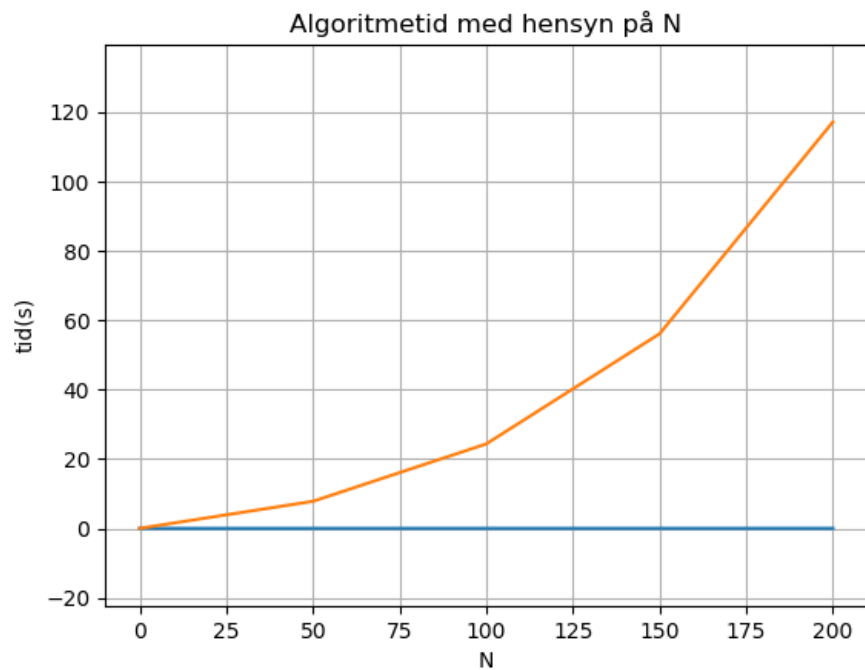


Figure 1: Algoritmetid med hensyn på N . Blå linje tilsvare tiden det tar å utføre utregningene ved bruk av armadillo, mens den oransje linjen tilsvare tiden det tar å utføre utregningene for Jacobis algoritme

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.

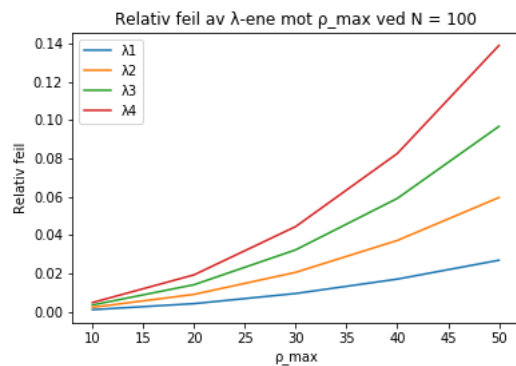


Figure 2: N=100. Den relative feilen for de fire første egenverdiene plottet mot forskjellige verdier av ρ_{max} der N=100

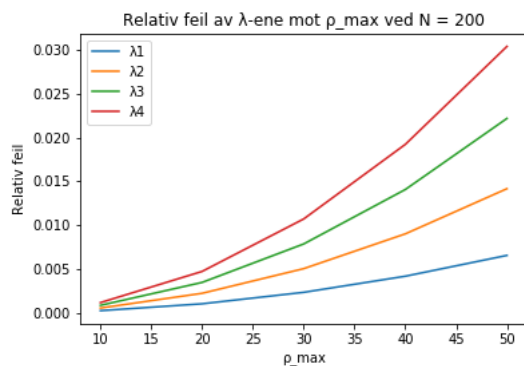


Figure 3: N=200. Den relative feilen for de fire første egenverdiene plottet mot forskjellige verdier av ρ_{max} der N=200

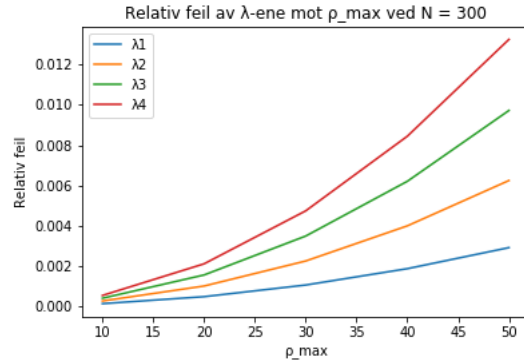


Figure 4: $N=300$. Den relative feilen for de fire første egenverdiene plottet mot forskjellige verdier av ρ_{\max} der $N=300$.

4.3 To elektroner i et harmonisk potensial

I denne delen ble to elektroner analysert i et harmonisk potensial. Det ble sett på sannsynlighetstettheten for elektronene over distansen ρ , plottet kan ses i figur 5. Her ble det antatt at elektronene ikke intererer med hverandre.

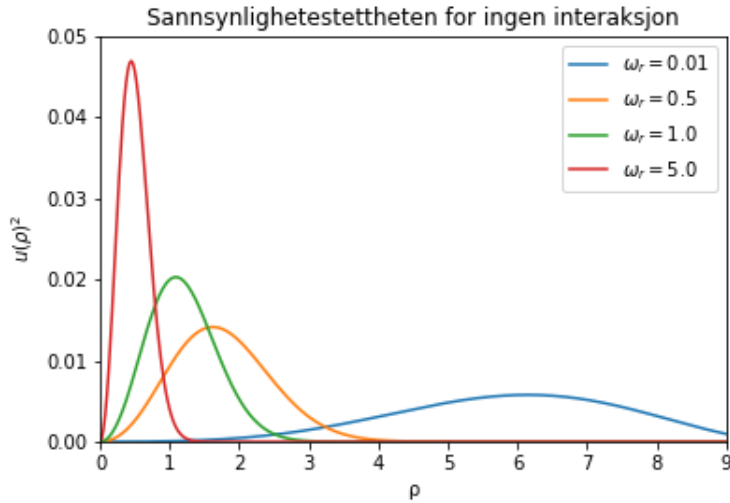


Figure 5: Sannsynlighetstettheten uten interaksjon mellom to elektroner i et harmonisk potensial.

5 Diskusjon

Diagonalisering av en matrise ved bruk av Jacobi metoden er avhengig av at den kjøres på en symmetrisk Toeplitz matrise. Denne metoden vil miste presisjon ved bruk på matriser som inneholder mange null-elementer. Etter rotasjonsfunksjonen blir kjørt N ganger så vil det oppstå forskjellige små verdier på null-element plassene.

Ved økende dimensjonen øker tiden det tar å utføre algoritmene. 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} gis det større relativ feil. Dette skyldes at ved økende ρ_{max} gis mindre verdier for diagonalen ut ifra likning 7:

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

siden $\rho_{min} = 0$. Her er det mulig å se at ved økende ρ_{max} fås et mindre og mindre bidrag fra venstre leddet og større bidrag fra potensialet. Dette fører til presisjonstap. Dermed fører en økende verdi for ρ_{max} til en større relativ feil iforhold til de eksakte verdiene.

En annen parameter 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 få flere punkter i matrisen og alle løkkene vil kjøre ganger. Dette gir en større presisjon. Problemet med større matriser er at dette krever mer datakraft(RAM) og tar lenger tid å kjøre, noe som ble vist i figur 1.

Fra figur 5 vises sannsynlighetstettheten for elektronene uten interaksjon mellom dem. Figuren viser også at for økende frekvenser befinner elektronene seg innenfor et kortere intervall, altså nærmere hverandre.

6 Konklusjon

I dette projektet ble Jacobis metode anvendt og analysert. Metoden ble implementert for en generell symmetrisk toeplitz matrise. Videre ble dette brukt til å løse Schrödingers likning for både ett og to elektroner. Alt i alt er Jacobis metode en fungerende metode som gir liten error så lenge dimensjonen N er stor nok. Problemet med dette er at for stor N vil føre til for stor bruk av RAM og tid. Altså vil nok armadillo være den mest brukervennlige metoden.

I den kvantemekaniske analysen ble det sett på hvilke faktorer som kunne påvirke den relative energien til elektronet i grunntilstanden. Her ble resultatet at ved en relativ liten ρ_{max} og større N gir bedre presisjon. For videre fordypning i prosjektet er det mulig å gå dypere inn på hvilke ρ_{max} som gir den miste relative feilen.

7 Appendiks

7.1 Bevis: Indreprodukt og ortogonaliteten er bevart

En enhetlig transformasjon bevarer ortogonaliteten til de oppnådde egenvektorene. Det blir sett på en basis \vec{v}_i ,

$$\vec{v}_i = [v_{i1}, \dots, v_{in}] \quad (10)$$

Det antas at basisen er ortogonal, altså

$$\vec{v}_j^T \vec{v}_i = \delta_{ij}. \quad (11)$$

Dermed er det mulig å bevise at en ortogonal eller en enhetlig transformasjon bevarer prikkproduktet og ortogonaliteten. Starter med

$$\vec{w}_i = \vec{U} \vec{v}_i \quad (12)$$

Antar at U er en symmetrisk matrise slik at $\vec{U}^T = \vec{U}^{-1}$. Dermed fås det at:

$$\begin{aligned} \vec{w}_j^T \vec{w}_i &= (\vec{U} \vec{v}_j)^T \vec{U} \vec{v}_i \\ &= \vec{v}_j^T \vec{U}^T \vec{U} \vec{v}_i = \vec{v}_j^T \vec{U}^{-1} \vec{U} \vec{v}_i = \vec{v}_j^T I_n \vec{v}_i \\ &= \vec{v}_j^T \vec{v}_i = \delta_{ij}. \end{aligned}$$

Altså er $\vec{w}_j^T \vec{w}_i = \delta_{ij}$ som betyr 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:

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

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

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

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:

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

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:

$$\lambda = \frac{2m\alpha^2}{\hbar^2} E \quad (16)$$

Som videre gir schroedinger likningen vi skulle utlede:

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

8 Bibliografi

[1] Morten Hjorth-Jensen, august 2015, Computational Physics Lecture Notes, 7 Eigensystems, (<https://github.com/CompPhysics/ComputationalPhysics/blob/master/doc/Lectures/lectures20>)

[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>