Multigrid methode toegepast op de Schrödinger vergelijking

Andy Martinez

27 January 2016

Inhoudsopgave

1 Inleiding

1.1 Discretisatie van model problemen

Stel we hebben een differentiaal vergelijking zoals de diffusie vergelijking:

$$\frac{\partial^2 u}{\partial x^2} = f(x) \tag{1}$$

Deze vergelijking stelt de steady-state (tijdsonafhankelijke) 1D diffusie vergelijking voor. Een toepassing van deze vergelijking is bijvoorbeeld de warmte diffusie in een 1D staaf als **u** de temperatuur voorstelt. We gebruiken Dirichlet randvoorwaarden waarbij we $u(0) = u(L) = 0$ stellen met L de lengte van de beschouwde 1D ruimte. Deze Dirichlet randvoorwaarden kunnen in het geval van de temperatuur worden voorgesteld als de staaf die met zijn uiteinden in ijswater wordt gehouden.

Deze vergelijking kan analytisch worden opgelost, maar we kunnen doen alsof dit niet gaat en dat we dit numeriek moeten oplossen. Als we deze vergelijking numeriek willen oplossen dan hebben we het probleem dat de vergelijking een continue vergelijking is. Allereerst zullen we dus een discretisatie moeten invoeren zodat de vergelijking in de continue ruimte nu een vergelijking wordt in de discrete ruimte. Als we een eindig aantal elementen hebben waarmee we deze vergelijking kunnen voorstellen dan kan dit dus ook worden voorgesteld in een computer en is het probleem geschikt om numeriek op te lossen.

De discretisatie die zal worden ingevoerd zal de eindige differentie methode zijn. Hierbij benaderen we de functie door zijn Taylor expansie af te kappen na een bepaald aantal termen. De Taylor expansie van een functie wordt geschreven als:

$$f(x_0 + h) = f(x_0) + hf'(x_0) + h^2 \frac{f''(x_0)}{2!} + O(h^3) \tag{2}$$

uit deze Taylor expansie kan een benadering voor de eerste afgeleide worden gevonden. Als we de reeks tot eerste orde schrijven dan krijgen we:

$$f(x_0 + h) \approx f(x_0) + hf'(x_0) \tag{3}$$

waardoor de eerste afgeleide in een punt $x_0 = a$ wordt:

$$f'(a) \approx \frac{f(a+h) - f(a)}{h} \tag{4}$$

Voor de tweede afgeleide kunnen we hetzelfde doen. Als we volgende twee expansies opschrijven tot 2e orde:

$$f(a + h) \approx f(a) + hf'(a) + h^2 \frac{f''(a)}{2!} \tag{5}$$

$$f(a - h) \approx f(a) - hf'(a) + h^2 \frac{f''(a)}{2!} \tag{6}$$

dan kunnen we zien dat als we de som nemen, we geen benadering meer moeten invoeren voor de eerste afgeleide omdat deze wegvalt:

$$f(a + h) + f(a - h) \approx 2f(a) + h^2 f''(a) \tag{7}$$

hieruit halen we de benadering voor de tweede afgeleide in een punt a:

$$f''(a) \approx \frac{f(a+h) - 2f(a) + f(a-h)}{h^2} \tag{8}$$

Met deze twee benaderingen kunnen we de differentiaal vergelijking (1) benaderen met zijn discrete vorm:

$$\frac{v_{i-1} - 2v_i + v_{i+1}}{h^2} = f_i \tag{9}$$

waarbij we twee notaties hebben ingevoerd. Allereerst hebben we v geschreven in plaats van u waarbij we aanduiden dat dit een benadering is voor u. Een tweede additie aan de notatie zijn de subscripts zoals v_i. Hiermee bedoelen we de waarde van v in gridpunt i. Herinner dat we de ruimte nu gediscretiseerd hebben in een eindig aantal punten en het zijn in deze punten dat we v berekenen. Dit is samen te vatten in:

$$v_{i-1} \approx u(x_i - h) \tag{10}$$
$$v_i \approx u(x_i) \tag{11}$$
$$v_{i+1} \approx u(x_i + h) \tag{12}$$
$$\tag{13}$$

Nu dat we de vergelijking in component vorm hebben gezet kunnen we opmerken dat dit ook in matrix notatie kan gezet worden:

$$\frac{1}{h^2}\begin{bmatrix} -2 & 1 & & & & & \\ 1 & -2 & 1 & & & & \\ & \cdot & \cdot & \cdot & & & \\ & & \cdot & \cdot & \cdot & & \\ & & & \cdot & \cdot & \cdot & \\ & & & & \cdot & \cdot & 1 \\ & & & & & 1 & -2 \end{bmatrix} \begin{pmatrix} v_1 \\ v_2 \\ \cdot \\ \cdot \\ \cdot \\ \cdot \\ v_n \end{pmatrix} = \begin{pmatrix} f_1 \\ f_2 \\ \cdot \\ \cdot \\ \cdot \\ \cdot \\ f_n \end{pmatrix} \tag{14}$$

of dus $\mathrm{A}\mathbf{v} = \mathbf{f}$.

In dit hoofdstuk werd het continue probleem gediscretiseerd waardoor we nu een stelsel van lineaire vergelijkingen moeten oplossen om een benadering te krijgen op de oplossing van het gezochte continue probleem. Er zijn tal van manieren om zo'n stelsel op te lossen en we zullen hier in het volgende hoofdstuk wat meer aandacht aan schenken.

1.2 Iteratieve methodes

Stel dat $\mathrm{A}\mathbf{u} = \mathbf{f}$ het exacte probleem is en dat $\mathrm{A}\mathbf{v} = \mathbf{f}$ het benaderde probleem is. Nu willen we graag een maat voor de goedheid van de benadering $\mathbf{v}$. Een voor de hand liggende maat is de error:

$$\mathbf{e} = \mathbf{u} - \mathbf{v} \tag{15}$$

dit resultaat is op zich ook een vector en omdat we graag 1 getal hebben dat de goedheid van de benadering voorstelt kunnen we hierop een norm toepassen. De meest gebruikte normen hiervoor zijn de max-norm en de Euclidische 2-norm:

$$||\mathbf{e}||_\infty = \max(|e_i|) \quad 1 \leq i \leq n \tag{16}$$
$$||\mathbf{e}||_2 = \left(\sum_{i=1}^{n} e_i^2\right)^{\frac{1}{2}} \tag{17}$$

waarbij n het aantal segmenten zijn waarin we onze ruimte hebben opgedeeld.

Wat natuurlijk meteen opvalt in de vergelijking voor de error is dat hiervoor de exacte oplossing nodig is wat nu juist datgene is dat wordt gezocht.

Een andere maat voor de goedheid van de gevonden benadering is het residu:

$$\mathbf{r} = \mathbf{f} - A\mathbf{v} \tag{18}$$

Het residu geeft een maat voor hoeveel de benaderde oplossing naast de oplossing van het exacte probleem $A\mathbf{u} = \mathbf{f}$ zit. Het residu is ook een vector waardoor we weer moeten grijpen naar een van de vooraf gedefinieerde normen.

Door de uniekheid van de oplossing van $A\mathbf{u} = \mathbf{f}$ zal $\mathbf{r} = 0$ zijn als en slechts als $\mathbf{e} = 0$. Maar het is niet zo dat een klein residu een kleine error voorstelt dus hier moet voor worden opgepast. Omdat dit een gemeen addertje onder het gras is zal dit worden geillustreerd met een voorbeeld:

$$\begin{pmatrix} 1 & -1 \\ 21 & -20 \end{pmatrix} \begin{pmatrix} u_1 \\ u_2 \end{pmatrix} = \begin{pmatrix} -1 \\ -19 \end{pmatrix} \tag{19}$$

$$\begin{pmatrix} 1 & -1 \\ 3 & -1 \end{pmatrix} \begin{pmatrix} u_1 \\ u_2 \end{pmatrix} = \begin{pmatrix} -1 \\ 1 \end{pmatrix} \tag{20}$$

Beide stelsels hebben de exacte oplossing $\mathbf{u} = (1, 2)^T$, maar stel nu dat we de benadering $\mathbf{v} = (1.95, 3)^T$ hebben berekend dan is de error voor deze benadering:

$$\mathbf{e} = \mathbf{u} - \mathbf{v} = (-0.95, -1)^T \tag{21}$$

waarvoor de Euclidische 2-norm $||\mathbf{e}||_2 = 1.379$. Nu kunnen ook de normen van de residus $\mathbf{r} = \mathbf{f} - A\mathbf{v}$ berekend worden. Voor het eerste stelsel krijgen we $||\mathbf{r_1}||_2 = 0.071$ en voor het tweede stelsel krijgen we $||\mathbf{r_2}||_2 = 1.851$. We zien duidelijk dat het kleine residu voor het eerste stelsel zich niet weerspiegeld in de vrij grote error. Dit maar om de vorig vernoemde valkuil te illustreren.

Als we hebben dat A$\mathbf{u} = \mathbf{f}$, $\mathbf{r} = \mathbf{f}$ - A$\mathbf{v}$ en $\mathbf{e} = \mathbf{u}$ - $\mathbf{v}$ kunnen we de volgende afleiding maken:

$$\begin{aligned} A\mathbf{u} &= \mathbf{f} \\ A\mathbf{u} &= \mathbf{r} + A\mathbf{v} \\ A\mathbf{u} - A\mathbf{v} &= \mathbf{r} \\ A\mathbf{e} &= \mathbf{r} \end{aligned} \tag{22}$$

Dit is de residu vergelijking. Deze zegt dat de error aan dezelfde set vergelijkingen moet voldoen als de oorspronkelijke $\mathbf{u}$ en $\mathbf{v}$ als we $\mathbf{f}$ vervangen met het residu.

Dit is een belangrijke vergelijking in de multigrid methode en zal later verenigd worden met de andere elementen die nodig zijn voor multigrid.

Gebruikmakende van de residu vergelijking kunnen we de volgende correctie toepassen op een benadering $\mathbf{v}$. Stel we hebben een benadering $\mathbf{v}$, hiermee kunnen we het residu $\mathbf{r} = \mathbf{f}$ - A$\mathbf{v}$ berekenen. Dit kan gebruikt worden om de residu vergelijking op te lossen zodat we een benadering voor $\mathbf{e}$ hebben. Met $\mathbf{e}$ kan dan een nieuwe benadering voor de oplossing berekend worden uit:

$$\mathbf{u} = \mathbf{v} + \mathbf{e} \tag{23}$$

Laten we nu terugkeren naar het diffusie probleem:

$$u_{i-1} - 2u_i + u_{i+1} = h^2 f_i \tag{24}$$

Er zijn tal van methodes om dit probleem op te lossen. In dit werkje zal de Jacobi methode worden

besproken waarna we een kleine maar belangrijke aanpassing zullen maken waardoor we de weighted Jacobi methode verkrijgen. Hierna zal Gauss-Seidel in het kort besproken worden en het grootste verschil met de Jacobi methode worden aangehaald. Er zijn ook aanpassing op Gauss-Seidel zoals Gauss-Seidel rood-zwart maar deze zal niet worden besproken. Weet gewoon dat de methodes die in dit werk worden besproken maar het topje van de ijsberg zijn.

De Jacobi methode is een van de simpelste manieren om het gediscretiseerde probleem op te lossen. Bij de Jacobi methode lossen we de i-de vergelijking op voor de i-de component waarbij we gebruik maken van de huidige benaderingen voor de $i+1$-de en $i-1$-de component. Dit geeft in component vorm:

$$\frac{1}{2}\left(v_{i+1}^{(0)} + v_{i-1}^{(0)} - h^2 f_i\right) = v_i^{(1)} \quad 1 \leq i \leq N-1 \tag{25}$$

Hier hebben we een nieuwe notatie ingevoerd waarbij we de n-de iteratie als superscript hebben genoteerd waarbij $n = 0$ de begingok is.

We kunnen de Jacobi methode ook in matrix vorm zetten. We splitsen de matrix A in de vorm:

$$A = D - L - U \tag{26}$$

waarbij D de diagonaal is en -L en -U respectievelijk de benedendriehoeks en bovendriehoeks componenten zijn. Dit geeft dan voor het oorspronkelijke probleem:

$$\begin{aligned} (D-L-U)\mathbf{u} &= \mathbf{f} \\ D\mathbf{u} &= \mathbf{f} + (L+U)\,\mathbf{u} \\ \mathbf{u} &= D^{-1}\mathbf{f} + D^{-1}\,(L+U)\,\mathbf{u} \end{aligned} \tag{27}$$

Het vermenigvuldigen met D^{-1} komt overeen met het oplossen van de i-de vergelijking voor u_i. Als we de Jacobi iteratie matrix definieren als:

$$R_J = D^{-1}\,(L+U) \tag{28}$$

dan krijgen we de Jacobi methode in matrix vorm:

$$\mathbf{v}^{(1)} = R_J \mathbf{v}^{(0)} + D^{-1}\mathbf{f} \tag{29}$$

Hierop kan een belangrijke aanpassing worden gedaan. We kunnen nu net als hiervoor de volgende iteratie berekenen (in component vorm):

$$v_i^\star = \frac{1}{2}\left(v_{i+1}^{(0)} + v_{i-1}^{(0)} - h^2 f_i\right) \tag{30}$$

maar deze keer gebruiken we dit als een tussenuitkomst. De echte nieuwe iteratie zal worden berekend uit een gewogen gemiddelde:

$$v_i^{(1)} = (1-\omega)v_i^{(0)} + \omega v_i^\star \tag{31}$$

$$= v_i^{(0)} + \omega\left(v_i^\star - v_i^{(0)}\right) \tag{32}$$

waarbij ω de gewichtsfactor is. Merk op dat als we $\omega = 1$ kiezen we de gewone Jacobi methode terugkrijgen. Deze vergelijking geeft ons een hele nieuwe familie van iteratieve methodes gebaseerd op de Jacobi methode. Deze methode wordt de weighted Jacobi methode genoemd. In matrix vorm wordt deze gegeven door:

$$\mathbf{v}^{(1)} = \left[(1-\omega)\, I + \omega R_J\right] \mathbf{v}^{(0)} + \omega D^{-1}\mathbf{f} \tag{33}$$

als we de weighted Jacobi matrix definieren als:

$$R_W = (1-\omega)\, I + \omega R_J \tag{34}$$

dan krijgen we:

$$\mathbf{v}^{(1)} = R_W \mathbf{v}^{(0)} + \omega D^{-1}\mathbf{f} \tag{35}$$

Wat moet worden opgemerkt is dat de Jacobi methode alle componenten berekend vooraleer het een van de nieuw berekende componenten gebruikt. Hierdoor zal de opslagruimte die nodig is om deze methode te gebruiken gelijk zijn aan 2N. Het betekent ook dat nieuwe informatie niet meteen wordt gebruikt vanaf dat het beschikbaar is.

De Gauss-Seidel methode implementeerd een simpele aanpassing. De componenten van de nieuwe benadering worden meteen gebruikt vanaf ze beschikbaar zijn. Deze aanpassing verminderd de opslagruimte voor de benaderingsvectoren naar N. In matrix notatie is deze methode gegeven als:

$$\mathbf{v} = R_G \mathbf{v} + (D-L)^{-1}\,\mathbf{f} \tag{36}$$

waarbij R_G gegeven wordt door:

$$R_G = (D-L)^{-1}\, U \tag{37}$$

Er zijn nog tal van andere methodes om stelsels op te lossen waarbij de computers waar ze op worden opgelost van belang zijn voor de keuze van de iteratieve methode. Nu werden de Jacobi methode en Gauss-Seidel methode aangehaald omdat dit de methodes zijn die in de loop van dit werk zullen worden gebruiken.

1.3 Invloed van de Iteratieve Methodes

Het oorspronkelijke probleem hebben we gediscretiseerd en we hebben oplossings methodes besproken. Nu rest er nog een belangrijke vraag namelijk: Wat is de invloed van zo'n oplossings methode op mijn oplossing ? Om de invloed te bepalen van een iteratieve methode zullen we het probleem een beetje veranderen zodat de invloed makkelijk zichtbaar wordt. We zullen het homogene probleem

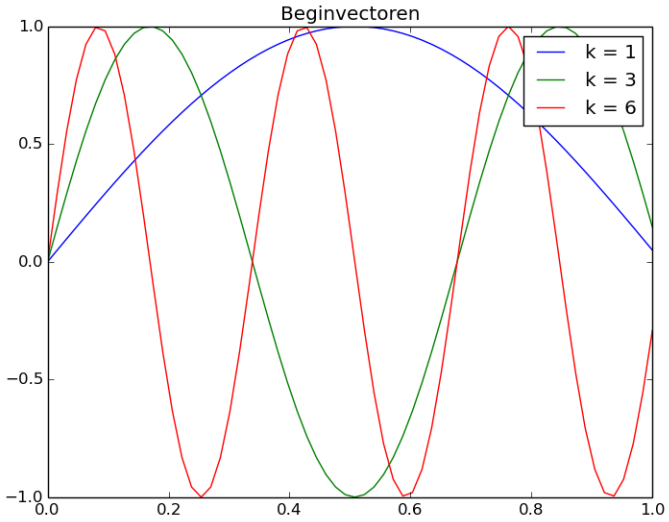
$$A\mathbf{u} = \mathbf{0} \tag{38}$$

beschouwen. Dit heeft als reden dat de oplossing dan $\mathbf{u} = \mathbf{0}$ is waardoor dat de error $\mathbf{e} = \mathbf{u} - \mathbf{v} = -\mathbf{v}$ is.

We kunnen nu zien wat er gebeurt met onze oplossing als we verschillende begingokken vergelijken. De begingokken die we gaan vergelijken zullen van de vorm:

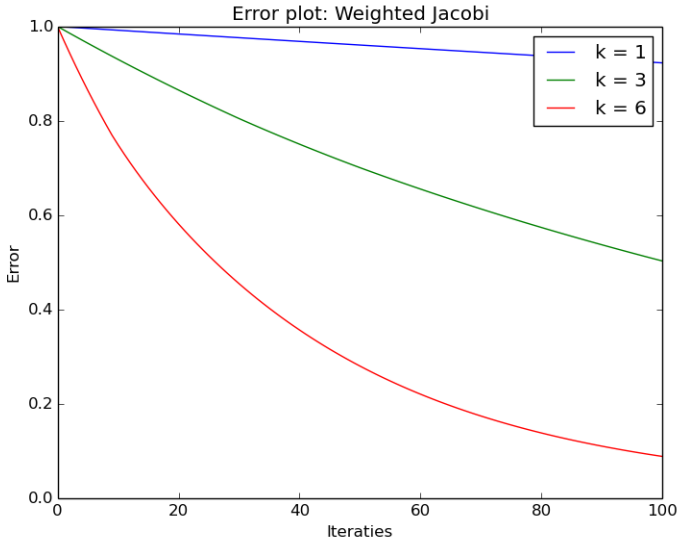
$$v_j = \sin\left(\frac{jk\pi}{n}\right) \tag{39}$$

waarbij j het gridpunt is en k het golfgetal (of frequentie op een constante na). Stel we nemen k = 1, 3, 6 dit worden dan vectoren die eruitzien als volgt:



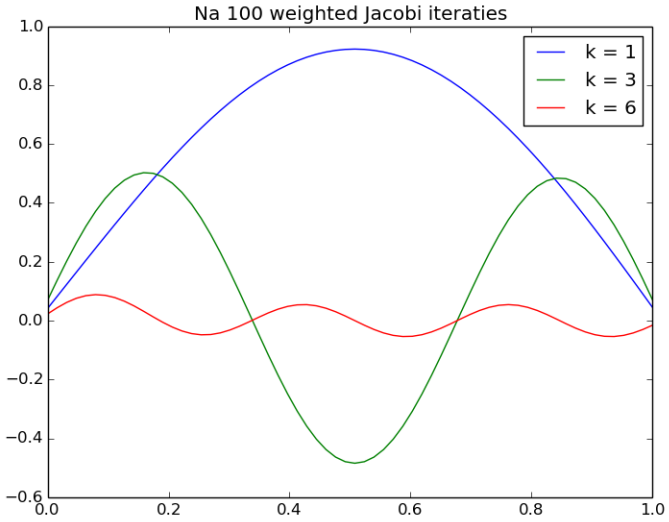
Figuur 1: Voorbeeld van beginvectoren

Als we hierop de weighted Jacobi methode op toepassen met een $\omega = \frac{2}{3}, n = 64$ en 100 iteraties dan zien we het volgende gedrag in de max-norm van de error:



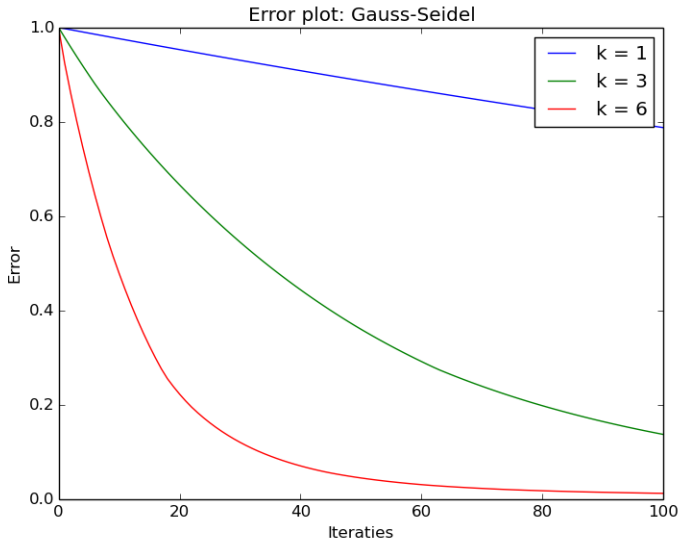
Figuur 2: De max norm van de error na weighted Jacobi iteraties

De beginvectoren zien er na 100 weighted Jacobi iteraties als volgt uit:



Figuur 3: De beginvectoren na 100 Weighted Jacobi iteraties

Voor de Gauss-Seidel methode zien deze er gelijkaardig uit:



Figuur 4: De max norm van de error na Gauss-Seidel iteraties

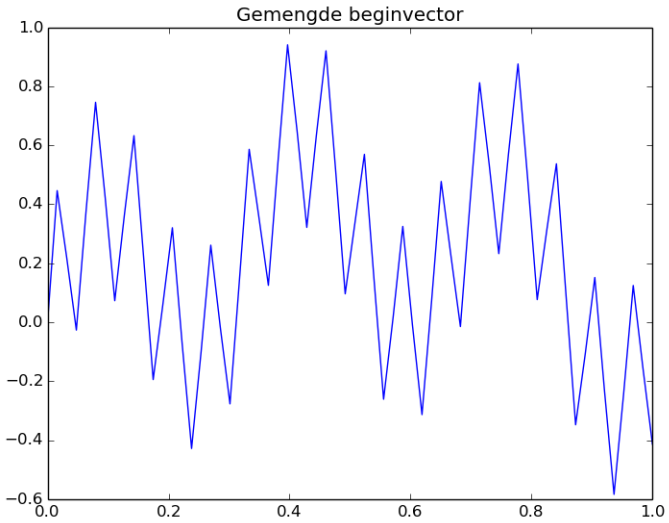
Zoals we zien daalt de error wel met meerdere iteraties, maar de snelheid waarmee de error vermindert hangt af van de frequentie. We zien dat grotere frequenties veel sneller in error verminderen dan de lage frequenties.

Uiteraard is het niet realistisch dat er maar 1 mode in de rechterhand of begingok zit dus we zullen nu een meer realistisch geval beschouwen. Stel we hebben een begingok die de volgende vorm heeft:

$$v_i = \frac{1}{3}\left(\sin\left(\frac{i\pi}{n}\right) + \sin\left(\frac{6i\pi}{n}\right) + \sin\left(\frac{32i\pi}{n}\right)\right) \tag{40}$$

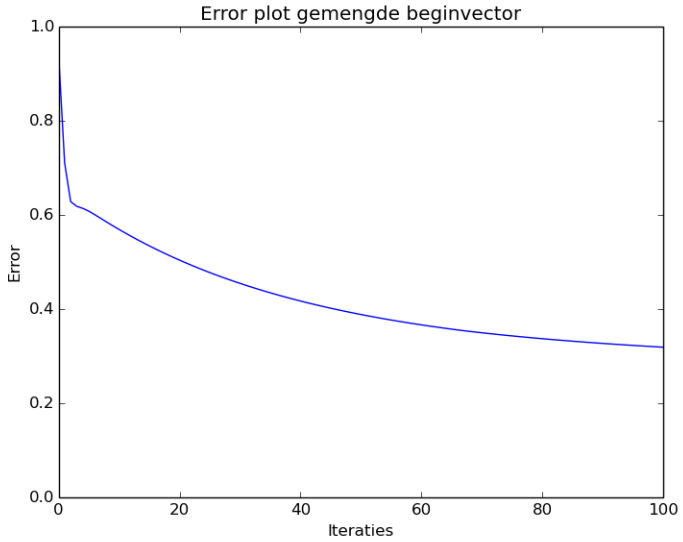
hier hebben we een k= 1, 6 en 32 component.

Deze beginvector ziet er als volgt uit:



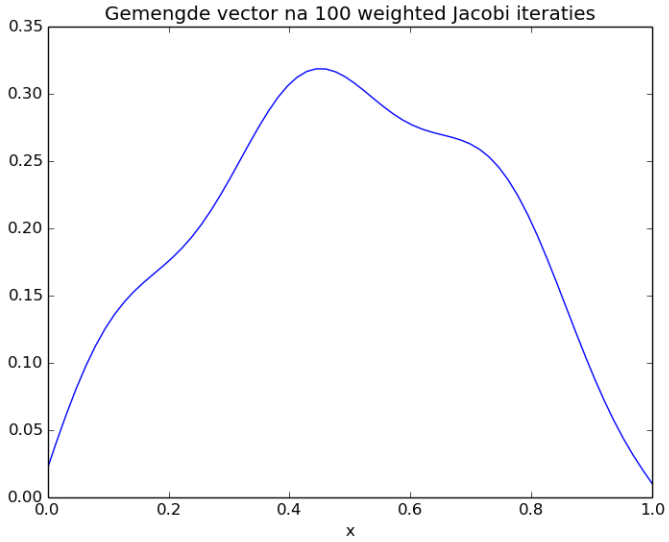
Figuur 5: De beginvector met verschillende k componenten

De max-norm van de error van de oplossing ziet er met de vorig gekozen weighted Jacobi methode uit als volgt:



Figuur 6: De max norm error van de gemengde vector na 100 weighted Jacobi iteraties

De beginvector ziet er na 100 iteraties als volgt uit:



Figuur 7: De gemengde vector na 100 weighted Jacobi iteraties

De snelle vermindering van de error in het begin komt door de hoge frequentie component. Hierna zijn het enkel de lage frequentie componenten die overblijven waardoor dat de snelheid van de afname van de error sterk afneemt. We zien dus dat de standaard iteratieve methodes goed zijn om hoge frequentie componenten uit te schakelen maar niet efficient zijn in het uitschakelen van de lage frequentie componenten.

Dit resultaat is redelijk experimenteel maar kan ook analytisch worden aangetoond, maar omdat dit niet nodig is voor de multigrid methode te verstaan zal dit niet worden gedaan in dit werk.

De beperking dat de beschouwde relaxtie methodes niet effectief zijn in het uitschakelen van de lage frequentie componenten zal worden opgelost in het volgende deel en deze oplossing zal ons leiden tot de multigrid methode.

1.4 Het gebruik van groffere grids

Wat ons in het vorige hoofdstuk is opgevallen was dat de standaard relaxatie methodes niet effectief zijn in het uitschakelen van de lage frequentie componenten, maar zeer effectief zijn in het uitschakelen van de hoge frequentie componenten. Een eerste manier om de relaxatie methode beter te laten werken is om een goede begingok te hebben van de oplossing. Een veelgebruikte methode om dit te verwezenlijken is om de relaxatie methode eerst toe te passen op een groffer grid omdat hier minder onbekenden op zijn en dus minder duur is. Deze oplossing zal dan op een of andere manier gebruikt worden als begingok op het fijne grid.

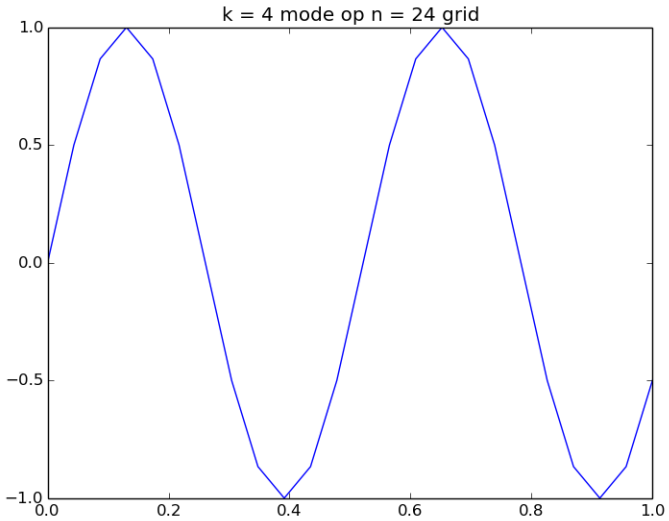
Omdat we nu met het idee spelen om groffere grids te gebruiken zullen we extra notatie invoeren. Een grid met gridafstand h zal worden genoteerd als Ω^h. Als we werken met een groffer grid dan zal dit genoteerd worden als Ω^{2h}.

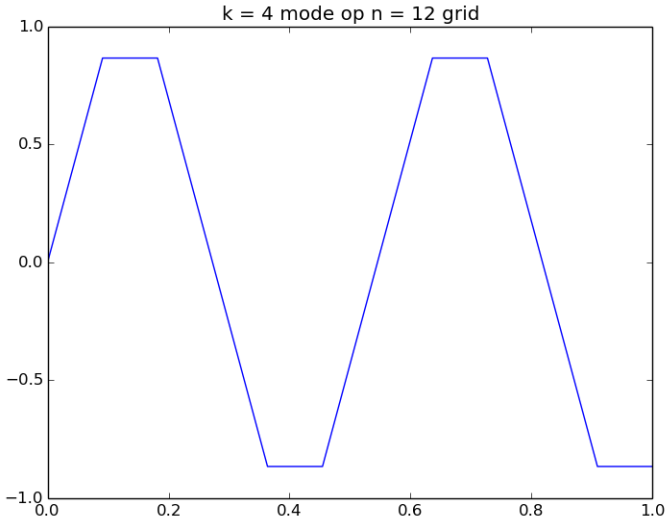
Wat als we nu de relaxatie methodes hebben toegepast op het fijne grid Ω^h totdat er enkel maar gladde componenten over zijn. Hoe zou dit er dan uitzien als we naar het groffe grid Ω^{2h} overgaan ?

Stel we hebben een oplossing:

$$v_i = \sin\left(\frac{ik\pi}{n}\right) \tag{41}$$

met k = 4 op een grid Ω^h met n = 24. Als we dit projecteren naar een grid Ω^{2h} met n = 12 dan zien we dat de oplossing er meer oscillatorisch uitziet:





Merk op dat de de oplossing op het Ω^{2h} grid simpelweg al de even punten zijn van de oplossing van het Ω^h grid. Om dit beter te illustreren beschouw dan de even punten van de k-de mode op het fijne grid:

$$\omega^h_{k,2j} = \sin\left(\frac{2jk\pi}{n}\right) = \sin\left(\frac{jk\pi}{n/2}\right) = \omega^{2h}_{k,j} \tag{42}$$

waarbij k nu begrensd is door $1 \leq k < \frac{n}{2}$, dit moet nu gebeuren omdat we anders te maken hebben met aliasing wat later kort zal worden aangehaald.

We hebben nu superscripts toegevoegd aan ω om aan te tonen bij welk grid de oplossing hoort. Zoals we zien is de k-de mode op het Ω^h grid de k-de mode op het Ω^{2h} grid. Dit zorgt er voor dat we dus een meer oscillerende functie krijgen op het groffe grid.

Als $k > \frac{n}{2}$ dan hebben we het probleem van aliasing dit betekent dat de functie zo'n grote frequentie heeft dat er niet genoeg punten zijn op het groffe grid om dezelfde k-component voor te stellen. Dit zorgt ervoor dat hoge frequentie componenten op Ω^h weergegeven worden als gladde componenten op Ω^{2h}. De k-de modes voor $k > \frac{n}{2}$ op Ω^h worden voorgesteld als de (n-k)-de mode op Ω^{2h}.

Het belangrijkste om mee te nemen uit deze uitleg is dat gladde modes op Ω^h oscillerende modes worden op Ω^{2h}. Dit kan nu worden gebruikt om het probleem van de traag varierende error op te lossen. Als we de relaxatie methodes toepassen op Ω^h totdat de error niet significant meer veranderd dan weten we dat er enkel nog maar gladde componenten over zijn. Als deze oplossing dan wordt geprojecteerd naar het Ω^{2h} grid dan kunnen de relaxatie methodes weer effectief zijn omdat deze gladde modes nu oscillerende modes zijn geworden.

De belangrijjkste vraag wordt nu: Hoe transformeren we het probleem naar een grof grid en wat betekent het om te relaxeren op dit grof grid ?

Ales wat we nu nodig hebben om een multigrid methode voor te stellen lligt op tafel. Herinner dat we een vergelijking hebben afgeleid voor de error:

$$A\mathbf{e} = \mathbf{r} = \mathbf{f} - A\mathbf{v} \tag{43}$$

welke zegt dat we rechtstreeks op de error kunnen relaxeren door gebruik te maken van de residu vergelijking. Er is nog een reden waarom we de residu vergelijking willen gebruiken en dat is dat relaxatie op $\mathrm{A}\mathbf{u} = \mathbf{f}$ met willekeurige gok equivalent is aan relaxatie op $\mathrm{A}\mathbf{e} = \mathbf{r}$ met begingok $\mathbf{e} = \mathbf{0}$.

We kunnen nu ook motiveren dat het gebruik van groffe grids tijdens de berekeningen ervoor kan zorgen dat we efficient relaxatie kunnen uitvoeren op alle componenten van de error.

De allereerste verbetering dat we hadden voorgesteld op de relaxatie was om een betere begingok te halen uit een groffer grid. Dit idee kan ook worden toegepast op het groffer grid waarbij we een begingok halen bij een nog groffer grid en ook op dit nog groffer grid kan dezelfde redenering worden toegepast en ga zo maar verder. Dit wordt weergegeven door volgende stappen:

- Relaxeer op $\mathrm{A}\mathbf{u} = \mathbf{f}$ op een zeer grof grid om een initieele gok te verkrijgen voor het volgende fijner grid
- .... etc.
- Relaxeer op $\mathrm{A}\mathbf{u} = \mathbf{f}$ op Ω^{4h} om een gok te verkrijgen voor Ω^{2h}
- Relaxeer op $\mathrm{A}\mathbf{u} = \mathbf{f}$ op Ω^{2h} om een gok te verkrijgen voor Ω^h
- Relaxeer op $\mathrm{A}\mathbf{u} = \mathbf{f}$ op Ω^h om een laatste approximatie te verkrijgen voor de oplossing

Het idee om een groffer grid te gebruiken om een verbeterde begin gok te genereren is de basis van wat "genestelde iteratie"wordt genoemd. Dit is een aantrekkelijk plan, maar we blijven met hetzelfde probleem kampen: Hoe transformeren we het probleem tussen grids ?

We kunnen nu ook een strategie voorstellen die gebruik maakt van de residu vergelijking om te relaxeren op de error. Dit kan worden gedaan door de volgende procedure te volgen:

- Relaxeer op $\mathrm{A}\mathbf{u} = \mathbf{f}$ op Ω^h om een benadering te verkrijgen voor $\mathbf{v}^h$
- Bereken het residu $\mathbf{r} = \mathbf{f}$ - $\mathrm{A}\mathbf{v}^h$

- Relaxeer op de residu vergelijking $\mathbf{Ae} = \mathbf{r}$ op Ω^{2h} om een benadering te verkrijgen op de error $\mathbf{e}^{2h}$
- Corrigeer de benadering die verkregen werd op Ω^h met de benadering voor de error verkregen op Ω^{2h} als $\mathbf{v}^h \leftarrow \mathbf{v}^h + \mathbf{e}^{2h}$

Deze procedure is de basis van wat het "correctie schema"wordt genoemd. We relaxeren op het fijn grid tot de error niet significant meer verminderd waarna we relaxeren op de residu vergelijking op een groffer grid om een benadering te krijgen op de error die we dan gebruiken om de oplossing van het fijne grid te corrigeren. Ook hier zijn er weer vragen zoals: Wat betekent het om te relaxeren op de residu vergelijking op een grof grid ?

1.5 Interpolatie en Restrictie

Om deze vragen te kunnen beantwoorden moeten we eerst weten hoe we informatie tussen verschillende grids kunnen uitwisselen. Als we tussen grids informatie uitwisselen dan beschouwen we enkel maar paren van grids waarbij het groffere grid een gridsafstand heeft van 2h waarbij h de gridafstand is van het fijnere grid. Het is nooit nodig om andere veelvouden te beschouwen. Als we van een grof grid naar een fijn grid moeten transformeren dan wordt dit interpolatie genoemd. Het omgekeerde noemt men restrictie. Er zijn veel interpolatie methodes, maar gelukkig voor ons is de lineaire interpolatie methode in de meeste gevallen effectief genoeg.

De lineaire interpolatie operator wordt genoteerd als I_{2h}^h. Deze neemt een vector op het groffe grid en maakt er een vector van voor het fijne grid op de volgende manier:

$$I_{2h}^h \mathbf{v}^{2h} = \mathbf{v}^h \tag{44}$$

waarbij de interpolatie als volgt gebeurd:

$$\begin{aligned} v_{2j}^h &= v_j^{2h} \\ v_{2j+1}^h &= \frac{1}{2}\left(v_j^{2h} + v_{j+1}^{2h}\right) \quad 0 \leq j \leq \frac{n}{2} - 1 \end{aligned} \tag{45}$$

de even punten worden direct overgenomen en voor de andere punten wordt er een gemiddelde genomen van de naaste punten op het groffe grid.

Stel nu dat de echte error (dat onbekend is) op het fijne grid glad is en dat we een benadering hebben op het groffe grid dan zal de interpolatie van de error op het groffe grid een glad resultaat zijn en dus zal het een vrij goede benadering zijn voor de error op het fijne grid.

In het omgekeerde geval, als de echte error oscillerend is dan kan zelfs een goede benadering op het groffe grid een slechte interpolatie geven voor het fijne grid.

De interpolatie is dus het effectiefste als de error glad is. Merk op dat we deze interpolatie procedure nodig hebben voor zowel de genestelde iteratie en het correctie schema, hieruit kunnen we concluderen dat deze twee methodes het beste werken als de error glad is. Het goed werken van de interpolatie procedure bij gladde errors kan goed van pas komen als we herinneren dat de relaxatie methodes het beste werken als de error oscillerend is.

Naast de interpolatie operators die ons van het grof grid naar het fijn grid brengen, hebben we ook restrictie operators nodig die ons van het fijn grid naar het grof grid nemen. Deze operators worden genoteerd als I_h^{2h} en werken in op een fijne grid vector als volgt $I_h^{2h}\mathbf{v}^h = \mathbf{v}^{2h}$. De restrictie gebeurd als volgt:

$$v_j^{2h} = \frac{1}{4}\left(v_{2j-1}^h + 2v_{2j}^h + v_{2j+1}^h\right) \tag{46}$$

Dit wordt de full weighting genoemd. Een andere simpelere manier van restrictie die we ook eerder hebben aangehaald is om de even punten van het fijne grid te gebruiken als de punten van het groffe grid.

Wat de beste methode is om restrictie of interpolatie te doen is een belangrijk onderdeel in de theorie van multigrid, maar verdere discussie hierover valt buiten het bereik van dit werk. Alle resultaten in dit werk zijn bekomen met de full weighting restrictie operator.

Wat handig is aan de full weighting restrictie operator is de volgende relatie met de interpolatie operator:

$$I_{2h}^{h} = c\left(I_{h}^{2h}\right)^{T} \tag{47}$$

waarbij c een reele constante is.

Omdat we niet enkel in 1D zulen werken moeten deze operators uitgebreid worden naar 2D en dat wordt op de volgende manier gedaan:

voor de interpolatie operator:

$$\begin{aligned} v_{2i,2j}^{h} &= v_{ij}^{2h} \\ v_{2i+1,2j}^{h} &= \frac{1}{2}\left(v_{ij}^{2h} + v_{i+1,j}^{2h}\right) \\ v_{2i,2j+1}^{h} &= \frac{1}{2}\left(v_{ij}^{2h} + v_{i,j+1}^{2h}\right) \\ v_{2i+1,2j+1}^{h} &= \frac{1}{4}\left(v_{ij}^{2h} + v_{i+1,j}^{2h} + v_{i,j+1}^{2h} + v_{i+1,j+1}^{2h}\right) \end{aligned} \tag{48}$$

voor de restrictie operator:

$$\begin{aligned} v_{ij}^{2h} = \frac{1}{16}[&v_{2i-1,2j}^{h} + v_{2i-1,2j+1}^{h} + v_{2i+1,2j-1}^{h} + v_{2i+1,2j+1}^{h} \\ &+ 2\left(v_{2i,2j}^{h} + v_{2i,2j+1}^{h} + v_{2i-1,2j}^{h} + v_{2i+1,2j}^{h}\right) \\ &+ 4v_{2i,2j}^{h}] \end{aligned} \tag{49}$$

Als de 1D interpolatie operator is geimplementeerd dan kan de 2D interpolatie operator bekomen worden door de volgende vergelijking:

$$P_{2h}^{h} = I_{2h}^{h} \otimes I_{2h}^{h} \tag{50}$$

waarbij P_{2h}^{h} de 2D interpolatie operator is. Gebruikmakende van de relatie (47) kan de 2D restrictie operator bekomen worden waarbij dat $c = 4$.

1.6 Multigrid methode

Nu dat we een manier hebben gedefinieerd waarop we informatie tussen grids kunnen uitwisselen kunnen we nog eens terug kijken naar het correctie schema. Het correctie schema kan nu precieser worden opgeschreven met behulp van de intergrid operators:

- Relaxeer ν_1 keer op $A^h\mathbf{u}^h = \mathbf{f}^h$ op Ω^h met initieele gok $\mathbf{v}^h$
- Bereken het fijne grid residu $\mathbf{r}^h = \mathbf{f}^h - A^h\mathbf{v}^h$ en transformeer het naar het groffe grid door gebruik te maken van $\mathbf{r}^{2h} = I_h^{2h} r^h$
- Los de residu vergelijking $A^{2h}\mathbf{e}^{2h} = \mathbf{r}^{2h}$ op het groffe grid Ω^{2h} op
- Interpoleer de error die werd berekend op het groffe grid naar het fijne grid door gebruik te maken van $\mathbf{e}^h = I_{2h}^h \mathbf{e}^{2h}$

- Corrigeer de benadering op het fijne grid met de geinterpoleerde error van het groffe grid $\mathbf{v}^h = \mathbf{v}^h + \mathbf{e}^h$
- Relaxeer ν_2 keer op $A^h\mathbf{u}^h = \mathbf{f}^h$ op het fijne grid Ω^h met de gecorrigeerde benadering $\mathbf{v}^h$

Deze procedure is het correctie schema dat we eerder hadden bedacht, maar nu concreter gemaakt doordat we de intergrid operatoren hebben ingevoerd. In de eerste stap relaxeren we op het fijne grid totdat het niet meer nuttig is. Zoals we eerder hebben gezien worden de oscillerende componenten vrij snel weggewerkt dus in de praktijk is ν_1 =1, 2 of 3.

Er is nog wel een probleem dat niet behandeld is geweest. Alle operatoren, vectoren en grids in het correctie schema zijn goed gedefinieerd buiten de operator A^{2h}. We kunnen dit beschouwen als de operator die we krijgen als we het probleem discretiseren op Ω^{2h}. Deze wordt berekend door gebruik te maken van de intergrid operators:

$$A^{2h} = I_h^{2h} A^h I_{2h}^h \tag{51}$$

We moeten nu wel de goede samenwerking van elk onderdeel benadrukken. Relaxatie op het fijne grid elimineert de oscillerende componenten van de error op een efficiente manier, waardoor we een gladde error krijgen. Als we veronderstellen dat de residu vergelijking op het groffe grid nauwkeurig kan worden opgelost, is het nog steeds van belang dat de error op een correcte manier wordt getransformeerd naar een error op het fijne grid. Doordat de error glad is en omdat we weten dat de interpolatie operatoren het beste werken op gladde componenten kunnen we er vanuit gaan dat de error op een correcte manier wordt geinterpoleerd naar het fijne grid gebruikmakende van de vooraf gedefinieerde interpolatie operator. Hierdoor zal de correctie op het fijne grid effectief blijken.

In een van de stappen van het twee-grid correctie schema moet de oplossing gezocht worden voor: $A^{2h}\mathbf{e}^{2h} = \mathbf{r}^{2h}$ op Ω^{2h}. Hoe kan dit nu het beste worden gedaan ? Als we de hele lijn van denken van het twee-grid correctie schema nu toepassen met Ω^{2h} als het 'fijne' grid dan kunnen we hiermee de oplossing vinden op het voorop gestelde probleem waarbij we naar het Ω^{4h} grid moeten gaan. Ook op het Ω^{4h} grid kan dezelfde vraag worden gesteld en ook hierop kunnen we het twee-grid schema toepassen waarbij we naar het Ω^{6h} grid moeten gaan en zo kunnen we voort blijven gaan. Deze recursie blijft gebeuren tot het niveau waarop een directe oplossingsmethode kan worden toegepast om het probleem op te lossen.

Dit recursief gebruik van het twee-grid correctie schema wordt de V-cycle genoemd en kan in zijn recursieve vorm geschreven worden als volgt:

$$\mathbf{v}^h = V^h\left(\mathbf{v}^h, \mathbf{f}^h\right) \tag{52}$$

1. Relaxeer ν_1 keer op $A^h\mathbf{u}^h = \mathbf{f}^h$ met initieele gok $\mathbf{v}^h$
2. Als Ω^h het grofste grid is ga dan naar stap 4
   anders
   $$\mathbf{f}^{2h} = I_h^{2h}(\mathbf{f}^h - A^h\mathbf{v}^h)$$
   $$\mathbf{v}^{2h} = \mathbf{0}$$
   $$\mathbf{v}^{2h} = V^{2h}(\mathbf{v}^{2h}, \mathbf{f}^{2h})$$
3. Corrigeer $\mathbf{v}^h = \mathbf{v}^h + I_{2h}^h\mathbf{v}^{2h}$
4. Relaxeer ν_2 keer op $A^h\mathbf{u}^h = \mathbf{f}^h$ met gecorrigeerde gok $\mathbf{v}^h$

De V-cycle is maar een voorbeeld van een hele familie van multigrid cycling schemas. Dit is ook het multigrid schema dat zal worden gebruikt voor alle toepassingen die volgen in dit werk.

Er moet nu wel een belangrijke opmerking worden gemaakt. Oorspronkelijk hadden we 2 schemas uitgedacht waarbij er een het correctie schema was en de andere het genestelde iteratie schema. Uit het correctie schema werd de V-cycle multigrid schema gehaald, maar we hebben het genestelde iteratie

schema nog niet gebruikt. Herinner dat het genestelde iteratie schema werd gebruikt om uit een grof grid Ω^{2h} een initieele gok te halen voor het fijn grid Ω^h hiermee kunnen we dus ook een initieele gok voor het Ω^{2h} grid halen uit het Ω^{4h} grid en we zien weer een recursie optreden.

Het algoritme dat zowel genestelde iteratie als de V-cycle bij elkaar brengt wordt het volledige multigrid V-cycle genoemd (FMG). In zijn recursieve vorm ziet deze er als volgt uit:

$$\mathbf{v}^h = FMG^h(\mathbf{f}^h) \tag{53}$$

1. Als Ω^h het grofste grid is neem $\mathbf{v}^h = \mathbf{0}$ en ga naar puntje 3
   anders
   $$\mathbf{f}^{2h} = I_h^{2h}(\mathbf{f}^h)$$
   $$\mathbf{v}^{2h} = FMG^{2h}(\mathbf{f}^{2h})$$
2. Corrigeer $\mathbf{v}^h = I_{2h}^h \mathbf{v}^{2h}$
3. $\mathbf{v}^h = V^h(\mathbf{v}^h, \mathbf{f}^{2h})$ ν_0 keer

Wat wel belangrijk is om te weten is dat in dit werkje enkel maar de V-cycle werd gebruikt. Herinner dat de genestelde iteratie nodig was om een initieele gok te verkrijgen door het probleem op Ω^{2h} op te lossen en dan recursief verder te gaan. In de implementatie voor dit werkje werd de initieele gok gehaald door het probleem op een grof grid Ω^{nh} (met n > 1) op te lossen met een Krylov methode, de impliciet herstarte Lanczos methode. Deze Krylov methode is de ingebouwde methode in zowel Matlab als Numpy/Scipy om de eigenwaarden en eigenvectoren van een sparse matrix te vinden. Deze initieele gok werd dan geinterpoleerd naar het fijne grid gebruikmakende van de interpolatie operator die eerder werd gedefinieerd. Hierdoor werd er enkel maar gebruik gemaakt van de V-cycle en niet de volledige multigrid V-cycle.

1.7 De Schrödinger vergelijking

Deze lange afleiding van de multigrid methode was niet zonder doel. In dit werkje zal getracht worden om de afgeleide multigrid methode toe te passen op een zeer belangrijke vergelijking in de fysica: de Schrödinger vergelijking.

De tijdsonafhankelijke Schrödinger vergelijking ziet eruit als volgt:

$$H\Psi = E\Psi \tag{54}$$

waarbij H de hamiltoniaan is van het beschouwde systeem en E de energie waarden van het systeem. Zoals te zien uit de vorm van de vergelijking is dit een eigenwaarde probleem.

Voor een vrij deeltje ziet de hamiltoniaan H van het systeem er als volgt uit:

$$\frac{\hat{p}^2}{2m}\Psi = E\Psi \tag{55}$$

waarbij we de kwantumechanische operator voor de impuls p hebben genoteerd als $\hat{p}$. De eerste stap in het oplossen van het diffusie probleem waarmee we de multigrid afleiding zijn gestart was het discretiseren van de vergelijking. Als we herinneren dat de kwantummechanische operator $\hat{p}$ geschreven wordt als:

$$\hat{p} \rightarrow i\hbar\nabla \tag{56}$$

dan kunnen we de Schrödinger vergelijking voor een vrij deeltje in 1D schrijven als:

$$-\frac{\hbar^2}{2m}\frac{\partial^2\Psi}{\partial x^2} = E\Psi \tag{57}$$

We hebben in een vorig hoofdstuk afgeleid hoe we de tweede afgeleide kunnen discretiseren gebruikmakende van de Taylor reeks van een functie. De gediscretiseerde vorm van deze vergelijking wordt dan:

$$-\frac{\hbar^2}{2m}\frac{\Psi_{i+1}-2\Psi_i+\Psi_{i-1}}{h^2}=E\Psi_i \tag{58}$$

Dit kan nu ook in matrix notatie geschreven worden waarbij de operator H nu de gediscretiseerde vorm is van de Hamiltoniaan:

$$-\frac{\hbar^2}{2m}\begin{bmatrix}-2 & 1 & & & & & \\ 1 & -2 & 1 & & & & \\ & \cdot & \cdot & \cdot & & & \\ & & \cdot & \cdot & \cdot & & \\ & & & \cdot & \cdot & \cdot & \\ & & & & \cdot & \cdot & 1 \\ & & & & & 1 & -2\end{bmatrix}\begin{pmatrix}\Psi_1\\ \Psi_2\\ \cdot\\ \cdot\\ \cdot\\ \cdot\\ \Psi_n\end{pmatrix}=\begin{pmatrix}E_1\\ E_2\\ \cdot\\ \cdot\\ \cdot\\ \cdot\\ E_n\end{pmatrix}\begin{pmatrix}\Psi_1\\ \Psi_2\\ \cdot\\ \cdot\\ \cdot\\ \cdot\\ \Psi_n\end{pmatrix} \tag{59}$$

Merk op dat we hier twee onbekenden hebben: De energie E en de eigenvector Ψ. Als we Ψ exact kennen dan kunnen we de energie berekenen uit:

$$E=\frac{\Psi^T H\Psi}{\Psi^T\Psi} \tag{60}$$

Dit wordt het Rayleigh quotient genoemd in de wiskunde of de verwachte waarde van de energie in de fysica. In de fysica wordt er meestal gebruik gemaakt van de braket notatie waarbij dat de verwachte waarde van de energie wordt geschreven als:

$$E=\frac{\langle\Psi|\,H\,|\Psi\rangle}{\langle\Psi|\Psi\rangle} \tag{61}$$

Er zijn een aantal iteratieve methodes om eigenwaarden en eigenvectoren te vinden die in de loop van dit werkje werden uitgeprobeerd. De eerste methode waar er aandacht aan werd geschonken was de power methode. Met deze methode kan de grootste eigenwaarde (en dus grootste energie van het systeem) gevonden worden op een iteratieve manier.

Het algoritme voor de power methode werkt als volgt:

1. $v^{(0)}=\frac{v}{||v||}$
2. for k = 1, 2, ... do
3. $\quad w=Av^{(k-1)}$
4. $\quad v^{(k)}=\frac{w}{||w||}$
5. $\quad \lambda_k=(v^{(k)})^T Av^{(k)}$

Er zijn hier twee minpunten. Allereerst zijn we meestal niet geinteresseerd in de grootste energiewaarde van een systeem, maar eerder in de kleinste, de grondtoestand dus. Dit kan opgelost worden door de power methode aan te passen waarbij er dan gewerkt moet worden met de inverse operator A waarbij dat de grootste eigenwaarde van A^{-1} overeenstemd met de kleinste eigenwaarde van A.

Het algoritme voor de inverse power methode werkt als volgt:

1. $v^{(0)}=\frac{v}{||v||}$
2. for k = 1, 2, ... do

3. $Aw = v^{(k-1)}$

4. $v^{(k)} = \frac{w}{||w||}$

5. $\lambda_k = (v^{(k)})^T A v^{(k)}$

Het tweede probleem is dat er enkel maar de grootste of de kleinste eigenwaarde gevonden kan worden. Er kan geen eigenwaarde tussenin worden gevonden met de power methode. Dit is uiteraard een probleem als we bijvoorbeeld de bandenstructuur van een halfgeleider willen bepalen waarvoor we dus meerdere eigenwaarden nodig hebben. Dit probleem kan in zekere zin opgelost worden door gebruik te maken van de shift methode.

Het grote verschil tussen de power methode en de shift methode is dat we voor de shift methode zelf moeten weten waar de eigenwaarde ongeveer ligt. Het principe van de shift methode is hetzelfde als deze van de geinverteerde power methode. We vullen een gok in voor de energie E waardoor dat het oplossen van het probleem ons een goede benadering geeft voor de eigenvector Ψ waarna we dan het Rayleigh quotient kunnen gebruiken om de gok voor de energie E te verbeteren.

Stel dat λ en een niet 0 vector V een eigenpaar zijn van de matrix A. Als α een constante is dan zal $\lambda - \alpha$ en V een eigenpaar zijn van de matrix $(A - \alpha I)$. Hieruit halen we ook dat als λ en V een eigenpaar zijn van A en λ is niet gelijk aan α dan is $\frac{1}{\lambda - \alpha}$ en V een eigenpaar van $(A - \alpha I)^{-1}$. Dit is de basis van de shift methode waarbij er gebruik wordt gemaakt van de geinverteerde power methode.

Stel dat een N $\times$ N matrix de eigenwaarden $\lambda_1, \lambda_2, \cdots, \lambda_N$ heeft. Beschouw dan de eigenwaarde λ_j. Dan kunnen we nu een constante α kiezen zodat $\sigma_j = \frac{1}{\lambda_j - \alpha}$ de dominante eigenwaarde is van $(A - \alpha I)^{-1}$ dan zal de reeks (X_k) gedefinieerd door:

$$Y_k = (A - \alpha I)^{-1} X_k \tag{62}$$

$$X_{k+1} = \frac{Y_k}{||Y_k||} \tag{63}$$

en de reeks (c_k) gedefinieerd door:

$$c_k = \frac{Y_k^T X_k}{X_k^T X_k} \tag{64}$$

convergeren naar het eigenpaar σ_j, V_j van de matrix $(A - \alpha I)^{-1}$ waarbij dat de corresponderende eigenwaarde van A gehaald kan worden uit:

$$\lambda_j = \frac{1}{\sigma_j} + \alpha \tag{65}$$

of uit het Rayleigh Quotient gebruikmakende van eigenvector V_j.

Het algoritme van deze shift methode ziet er als volgt uit:

1. $v^{(0)} = \frac{v}{||v||}$

2. for k = 1, 2, ... do

3. $(A - \mu I)w = v^{(k-1)}$

4. $v^{(k)} = \frac{w}{||w||}$

5. $\lambda_k = (v^{(k)})^T A v^{(k)}$

Gebruikmakende van deze shift methode werd het volgende iteratie schema gemaakt voor het oplossen van het eigenwaarde probleem:

- Zoek de eigenwaarden en eigenvectoren op een zeer grof grid van de gediscretiseerde hamiltoniaan H gebruikmakende van de Impliciet Herstarte Lanczos methode
- Interpoleer de eigenvector van het zeer groffe grid naar het fijne grid waarop we het eigenpaar willen vinden
- Shift de hamiltoniaan met de eigenwaarde gevonden op het zeer groffe grid
- Pas de V-cycle multigrid methode toe op het oplossen van het inverse probleem van de vorm $A\mathbf{x} = \mathbf{f}$ waarbij dat A nu de inverse geshifte hamiltoniaan is
- Als het resultaat nog niet genoeg is geconvergeerd pas dan opnieuw een V-cycle toe
- Als het resultaat is geconvergeerd of de error divergeert stop dan
- Gebruik de verkregen eigenvector om de bijhorende eigenwaarde te vinden gebruikmakende van het Rayleigh quotient

Zoals we zien in het schema is het gebruik van de Impliciet Herstarte Lanczos methode belangrijk om een goede gok te verkrijgen voor de eigenwaarden en eigenvectoren, maar omdat dit een Krylov methode is welke niet wordt behandeld in mijn opleiding is dit een methode waar ik verder geen info over kan geven. Dit zal in een volgend hoofdstuk aangehaald worden als een voorstel om het gebruikte schema te verbeteren.

2 1D Problemen

2.1 De oneindige potentiaalput

Als eerste onderzochte probleem gaan we naar het typische modelprobleem in de kwantummechanica: Het deeltje in een oneindige potentiaalput. De Hamiltoniaan voor dit systeem ziet eruit als volgt:

$$\hat{H} = \frac{\hat{p}^2}{2m} + V(x) \tag{66}$$

waarbij:

$$V(x) = \begin{cases} 0, & 0 < x < L \\ \infty, & elders \end{cases}$$

waarbij L de lengte is van de put.

We beginnen met dit probleem omdat dit exact oplosbaar is. Hierdoor kunnen we zien hoe goed ons ontwikkelde algoritme werkt omdat we toegang hebben tot de error.

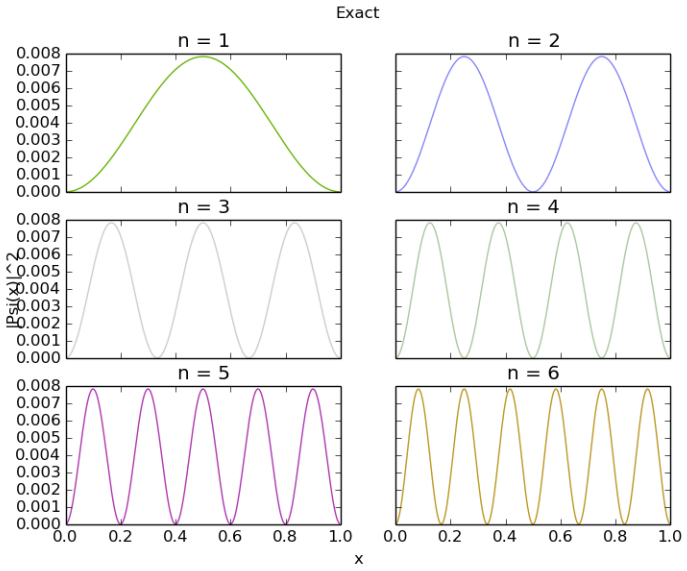
Door de impuls operator in te vullen en het probleem te discretiseren zoals werd getoond in een vorig hoofdstuk kan dit numeriek worden opgelost.

2.1.1 Exacte resultaat

De eigenvectoren van het deeltje in de oneindige potentiaal put zijn voor de n-de toestand van de vorm:

$$\Psi_n(x) = \sqrt{\frac{2}{L}} \sin\left(\frac{n\pi x}{L}\right) \tag{67}$$

De waarschijnlijkheidsdichtheden $\rho = \Psi^T \Psi$ van deze eigentoestanden zien er voor de eerste 6 toestanden als volgt uit:



Figuur 8: De waarschijnlijkheidsdichtheid voor de eerste 6 eigentoestanden

De energieen zijn:

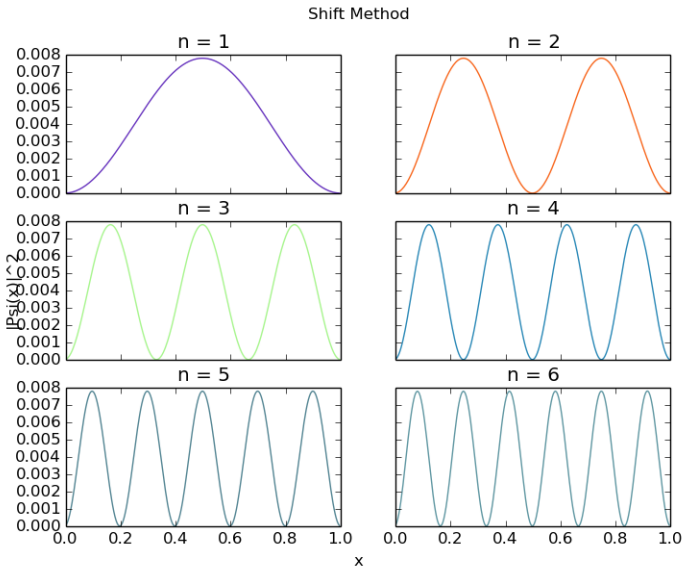
$$E_n = E_0 n^2 = \frac{\hbar^2 \pi^2}{2mL^2} n^2 \tag{68}$$

In eenheden van E_0 zijn de energieen van deze 6 toestanden simpelweg n^2 en dus: [1, 4, 9, 16, 25, 36].

2.1.2 Iteratief bekomen resultaat

Het fijnste grid waar we het resultaat op willen weten heeft n = 256 punten. We zullen de begingok zoeken op het groffe grid met n = 128 punten.

De waarschijnlijkheidsdichtheden zien eruit als volgt:



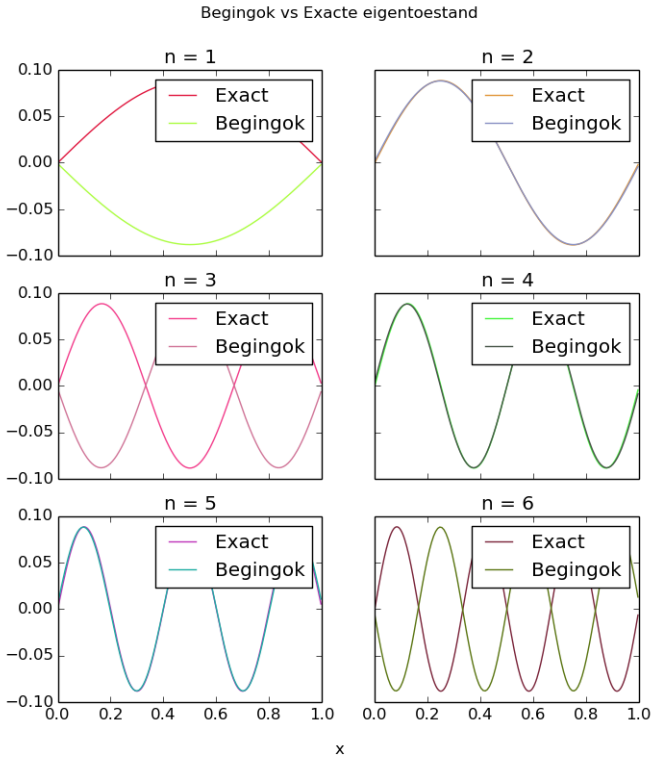
Figuur 9: De waarschijnlijkheidsdichtheid van de eerste 6 eigentoestanden

De corresponderende eigenwaarden (energieen) zijn: [0.9922207, 3.96873506, 8.92910211, 15.87258998, 24.79815738, 35.70429256]

De resultaten zijn in goede overeenstemming met de exacte resultaten, maar om een beter inzicht te krijgen in hoe goed onze methode werkt zullen we een betere analyse doen.

2.1.3 De faseverschuiving op de begingok

Het allereerste dat er gebeurt in de voorgestelde methode is een begingok zoeken voor zowel de eigentoestanden als de eigenwaarden. Dit gebeurt aan de hand van de ingebouwde Matlab en Numpy methodes die gebruik maken van de impliciet herstarte Lanczos methode. Als we de beginvectoren die we hebben gebruikt om het goede resultaat te verkrijgen met de exacte eigentoestanden vergelijken dan krijgen we het volgende te zien:



Figuur 10: De begingokken vergeleken met de exacte eigentoestanden

Er zijn al meteen twee zaken die opvallen. Allereerst lijkt het alsof sommige subplots maar uit 1 curve bestaan. Dit is simpelweg omdat de begingok zo goed is dat ze bijna recht op de exacte eigentoestand ligt. Als we zo meteen naar de error kijken zullen we een meer kwantitatief resultaat hiervan hebben. Het tweede dat opvalt is dat sommige curves een faseverschuiving hebben ten opzichte van elkaar. Dit is iets waar zeker en vast meer aandacht aan moet worden geschonken.

In kwantummechanica maakt een globale fasefactor niet uit aangezien alle fysisch relevante resultaten worden gehaald uit het modulus kwadraat van de eigentoestand. Dit wilt concreet zeggen dat als we een eigentoestand $\Psi' = \Psi e^{i\omega x}$ hebben dat het modulus kwadraat ons het volgende geeft:

$$\begin{aligned} |\Psi'|^2 &= \Psi^\dagger e^{-i\omega x} \Psi e^{i\omega x} \\ &= \Psi^\dagger \Psi \\ &= |\Psi|^2 \end{aligned} \tag{69}$$

Wat ons dus toont dat een globale faseverschuiving van de eigentoestand geen enkele invloed heeft op de fysica. Deze faseverschuiving in de begingok en mogelijks in opeenvolgende iteraties zal wel een fout beeld schetsen in de error van de eigentoestanden.

Als we de max norm errors $||\mathbf{e}_n||_\infty$ bekijken met n de toestand dan verkrijgen we:

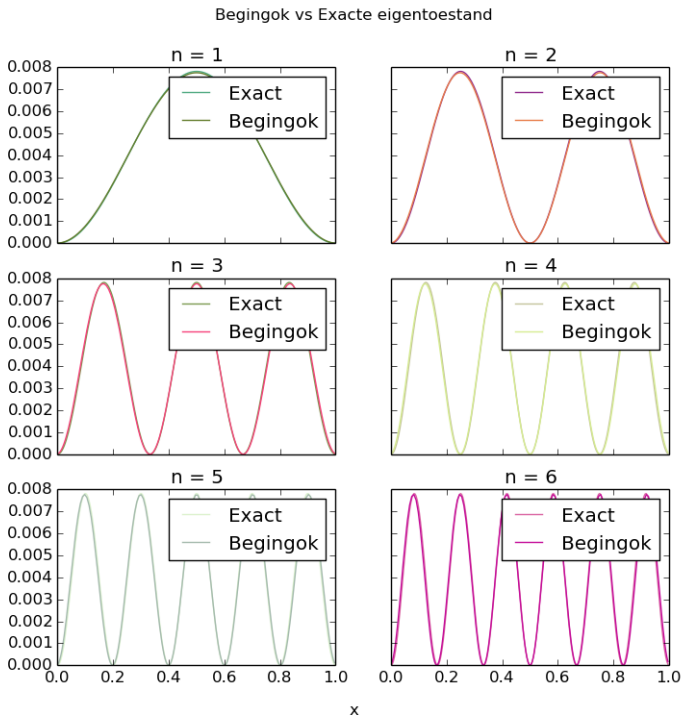
$$||\mathbf{e}_1||_\infty = 0.176430225068 \tag{70}$$
$$||\mathbf{e}_2||_\infty = 0.0021436751088 \tag{71}$$
$$||\mathbf{e}_3||_\infty = 0.176404508568 \tag{72}$$
$$||\mathbf{e}_4||_\infty = 0.00428419989893 \tag{73}$$
$$||\mathbf{e}_5||_\infty = 0.00535229600832 \tag{74}$$
$$||\mathbf{e}_6||_\infty = 0.176467779166 \tag{75}$$

We zien meteen dat de begingokken die praktisch op de exacte eigentoestand liggen een error hebben die 100 keer kleiner is dan die van de faseverschoven curves. Zoals eerder verteld schetst dit een enorm fout beeld van hoe goed of slecht de gevonden eigentoestand is. Een betere maat voor de error zou zijn om naar de errors in waarschijnlijkheidsdichtheden te zien omdat dan de faseverschuivingen weggewerkt worden.

De max norm errors voor de waarschijnlijkheidsdichtehden $\rho = \Psi^T\Psi$ worden gegeven door:

$$||\mathbf{e}_1||_\infty = 0.0000611278204893 \tag{76}$$
$$||\mathbf{e}_2||_\infty = 0.000124737677469 \tag{77}$$
$$||\mathbf{e}_3||_\infty = 0.00021663874061 \tag{78}$$
$$||\mathbf{e}_4||_\infty = 0.000310730330702 \tag{79}$$
$$||\mathbf{e}_5||_\infty = 0.000408165266869 \tag{80}$$
$$||\mathbf{e}_6||_\infty = 0.0005043732686 \tag{81}$$
$$\tag{82}$$

De curves zien er dan als volgt uit:



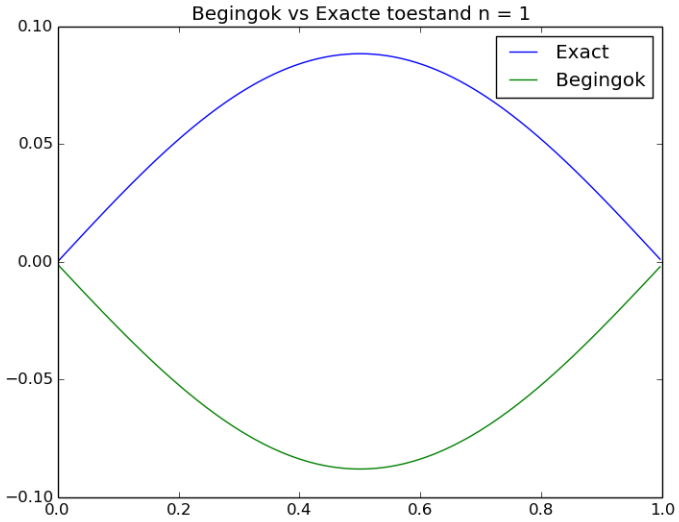
Figuur 11: De waarschijnlijkheidsdichtheden van de exacte eigentoestanden vergelijken met de begingokken

Zoals te zien uit deze resultaten zijn de errors in de waarschijnlijkheidsdichtheden zeer klein en dus een betere maat voor de goedheid van de gevonden eigenvectoren. Nogmaals, de waarschijnlijkheidsdichtheden zijn de enige relevante resultaten. Voor we verder gaan met het analyseren van elk onderdeel in de voorgestelde methode, moeten we eerst de vraag stellen: Wat is het gedrag van de iteratieve methodes wanneer een faseverschuiving plaatsvind ? Deze vraag is zeer belangrijk als we iteratieve methoden willen toepassen op kwantum mechanica.

2.1.4 Invloed van faseverschuiving op iteratieve methodes

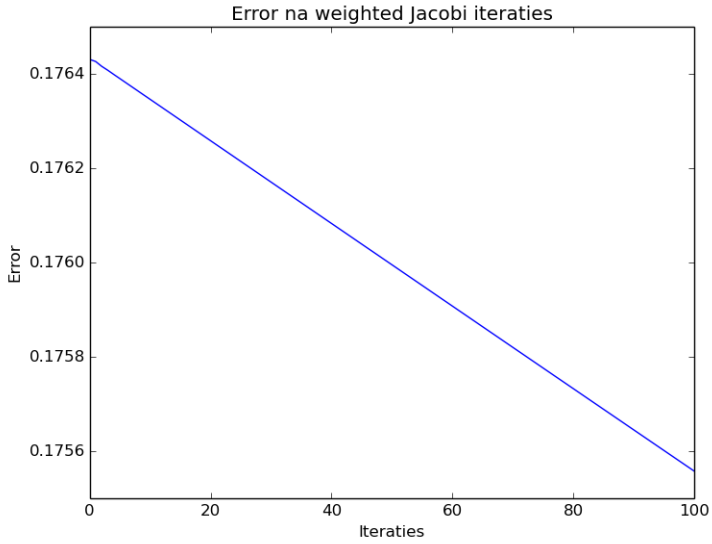
We kunnen onderzoeken wat het effect is van de faseverschuiving door de grondtoestand van het deeltje in een put probleem te onderwerpen aan een aantal weighted Jacobi iteraties. We doen hiermee exact dezelfde analyse als we in de inleiding hebben gedaan om de invloed van de iteratieve methodes na te gaan.

Voor de weighted Jacobi iteraties ziet ons systeem er as volgt uit:



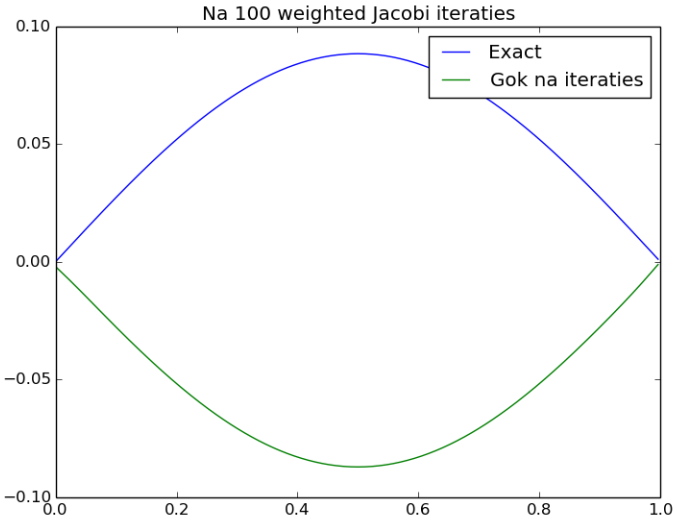
Figuur 12: De begingok en exacte toestand voor de grondtoestand van het deeltje in een put probleem

Zoals eerder is opgemerkt zit er een fase verschuiving in de begingok. Fysisch gezien zijn deze twee toestanden dezelfde. Als we hierop 100 weighted Jacobi iteraties op toepassen met $\omega = \frac{2}{3}$ en hiervan de max norm error plotten dan krijgen we het volgende te zien:



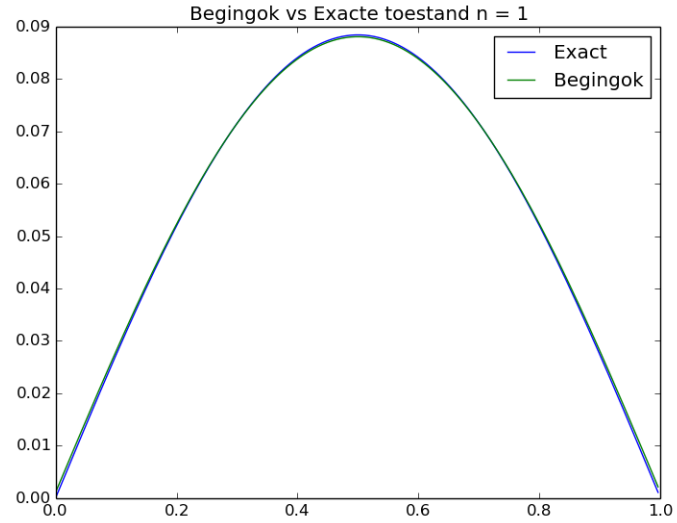
Figuur 13: De max norm error voor weighted Jacobi iteraties

Het resultaat na 100 weighted Jacobi iteraties is vrij teleurstellend de volgende:



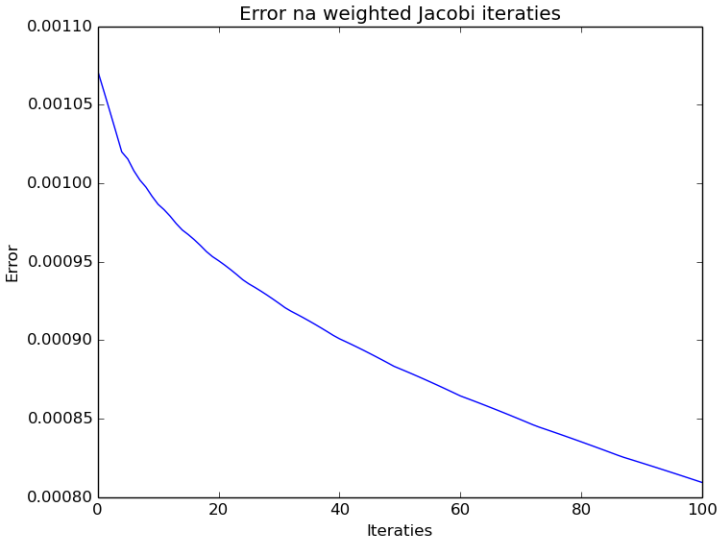
Figuur 14: De begingok na 100 weighted Jacobi iteraties

Nu kunnen we heel simpel deze fase verschuiving wegwerken door op te merken dat de faseverschuiving telkens π bedraagt, wat dus wilt zeggen dat we enkel het tegengestelde van de begingok moeten nemen om de faseverschuiving weg te werken. Dit resulteer in volgende begintoestand:



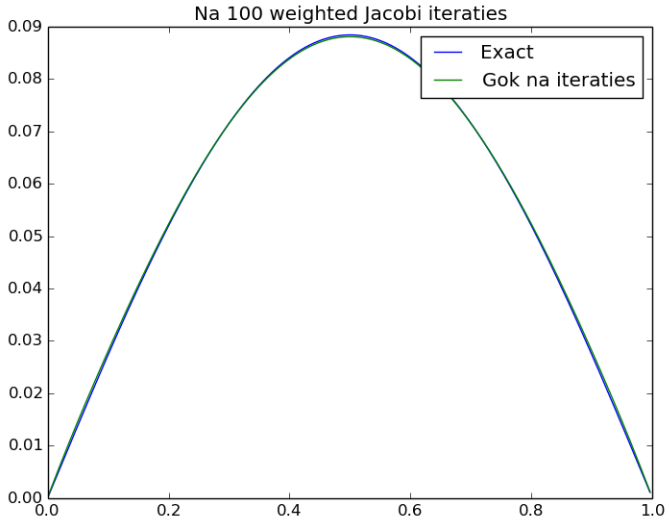
Figuur 15: De fase verschoven begingok en exacte toestand voor de grondtoestand van het deeltje in een put probleem

Als we nu 100 weighted Jacobi iteraties uitvoeren op deze geflipte begingok dan krijgen we het volgende resultaat:



Figuur 16: De max norm error voor weighted Jacobi iteraties op de geflipte begingok

Het resultaat na deze iteraties is dan :



Figuur 17: De geflipte begingok na 100 weighted Jacobi iteraties

Dit is toch een verschillend resultaat als voor de faseverschoven begingok. Let er ook op dat de y-as voor de error plots in beide situaties verschillende waarden tonen. Dit verschil in gedrag van de error zal het dus moeilijk maken om een goede convergentie te krijgen.

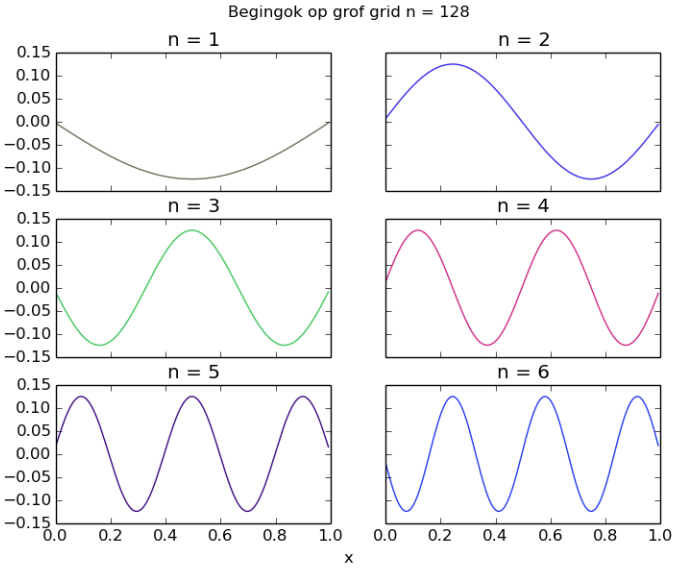
Ik wil er nogmaals op duiden dat dit een puur wiskundig probleem is aangezien fysisch gezien er geen enkel verschil is tussen deze twee toestanden. Dit zal er dus ook voor zorgen dat zowel de eigentoestand als de eigenwaarde goed kunnen overeenkomen met de realiteit ondanks dat de convergentie slecht lijkt te zijn. Dit is uiteraard veraderlijk aangezien we enkel kunnen vaststellen of de eigenwaarde/eigentoestand van een niet convergerend probleem goed is als we de exacte oplossingen hebben.

Zoals eerder al vermeld: als er een convergentiecriterium moet worden opgesteld dan is het beter om

te werken met de errors op de waarschijnlijkheidsdichtheden dan met de errors op de eigentoestanden zelf.

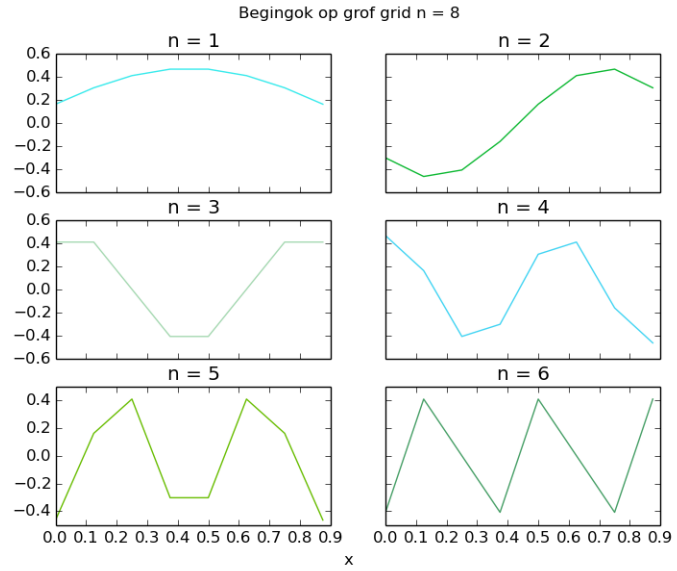
2.1.5 Convergentie naar een verkeerde toestand

We kunnen nu verder gaan met elk element van de voorgestelde methode te analyseren. Na de begingok op het groffe grid wordt deze begingok geinterpoleerd naar het fijne grid. Dit gebeurt met de eerder gedefinieerde interpolatie operator. Voor dit probleem zagen de groffe grid toestanden er als volgt uit:



Figuur 18: De begingokken op het groffe rooster met n = 128

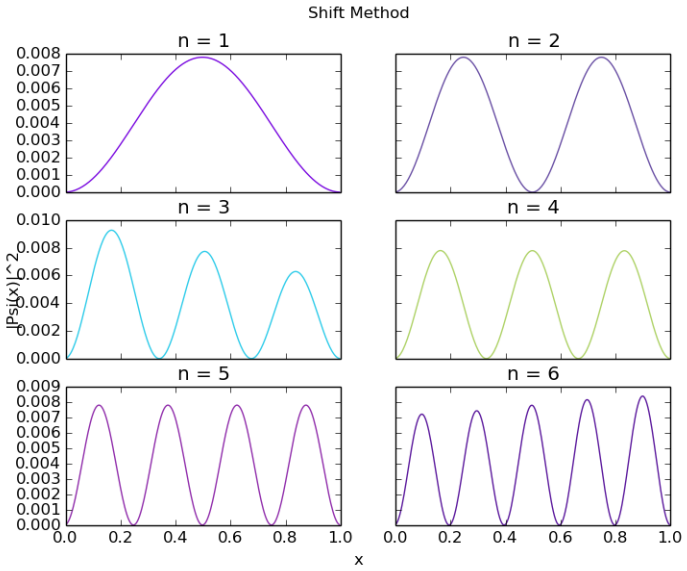
Omdat het gebruikte groffe grid n = 128 punten heeft zien deze begingokken er al glad uit. Wat we uiteraard willen is dat het bepalen van begingokken gebruikmakende van groffe roosters zo goedkoop mogelijk is. Als we een groffer grid kiezen zoals n = 8 dan krijgen we de volgende begingokken:



Figuur 19: De begingokken op het groffe rooster met n = 8

De begingokken voor de eigenwaarden zijn hiervoor: [0.78213313, 3.0341957, 6.48455575, 10.71704893, 15.22117409, 19.45366726]

Het resultaat op het fijne grid na het doorlopen van de multigrid methode is:



Figuur 20: Het resultaat na het multigrid schema te hebben doorgelopen

Een eerste manier om de eigentoestanden van het deeltje in een oneindige put te verifieren is om te kijken naar het aantal toppen dat de eigentoestand heeft. We zien dat de n = 1 toestand en de n = 2 toestand respectievelijk 1 en 2 toppen hebben. De n = 3 en n = 4 toestanden hebben allebei 3 toppen, de n = 5 toestand heeft er 4 en de n = 6 toestand heeft er 5. We weten al meteen dat dit niet klopt omdat het aantal toppen gelijk moet zijn aan n.

Laten we voor een beter zicht te krijgen op het probleem eens de begingokken voor de eigenwaarden vergelijken met de exacte eigenwaarden:

$$\text{Begingok} = [0.78213313,\ 3.0341957,\ 6.48455575,\ 10.71704893,\ 15.22117409,\ 19.45366726] \tag{83}$$

$$\text{Exact} = [1,\ 4,\ 9,\ 16,\ 25,\ 36] \tag{84}$$

Als we zien naar de 4e begingok 10.7 dan merken we op dat deze dichter bij de exacte eigenwaarde 9 van eigentoestand n = 3 ligt dan bij de eigenwaarde 16 die overeenstemt met eigentoestand n = 4. Dit zorgt ervoor dat als we de hamiltoniaan shiften met deze eigenwaarde en hierop de inverse power methode op toepassen dat het stelsel zal convergeren naar de n = 3 eigentoestand omdat dit verschil in eigenwaarden het kleinste is.

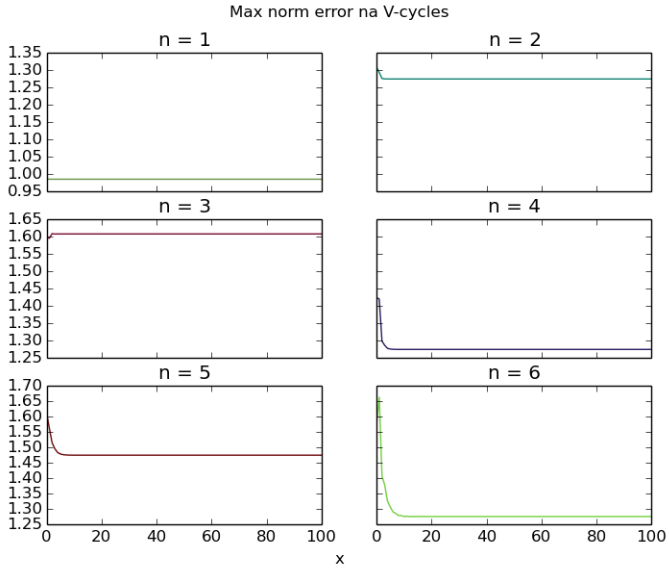
Uiteraard hebben we dit probleem zelf uitgelokt door het groffe grid maar 8 gridpunten te geven. De eigenwaarden van het deeltje in een oneindige put probleem liggen voor alle praktische toepassingen ver genoeg uit elkaar dat dit niet zou moeten gebeuren. Dit is natuurlijk wel een groot probleem als we systemen onderzoeken waarvan er eigenwaarden dicht bij elkaar liggen. Dit kan ervoor zorgen dat onze multigrid methode naar de foute eigentoestand zal convergeren.

2.1.6 Invloed van V-cycles op de error

We kunnen nu ook eens kijken hoe de error zich gedraagt na opeenvolgende V-cycles. In de V-cycle maken we gebruik van een smoother. In dit werkje is dat de weighted Jacobi methode, maar het kan

evengoed een van de andere relaxatie methodes zijn die in het begin zijn besproken. Er wordt een aantal keer de weighted Jacobi methode doorlopen in het begin (ν_1) van de V-cycle en op het einde (ν_2) van de V-cycle. Deze waarden staan voor volgende error analyse op: $\nu_1 = 4$ en $\nu_2 = 4$. Deze waarden zijn niet willekeurig gekozen, we hebben in de inleiding gezien dat de weighted Jacobi zeer effectief werkt op oscillatorische componenten, maar hierna weinig invloed heeft op de gladde componenten. Dit zorgt ervoor dat we niet al te veel keren de weighted Jacobi iteratie moeten doorlopen omdat dit weinig zin heeft.

Omdat we hebben gezien dat voor de begingok n = 8 te weinig gridpunten zijn, zullen we werken met n = 16 gridpunten waarbij we wel naar de juiste eigentoestanden convergeren op het fijne grid van n = 256 gridpunten. Als we kijken naar het gedrag van de max norm van de error op de waarschijnlijkheidsdichtheid voor 100 V-cycles voor elke mode dan krijgen we het volgende resultaat te zien:



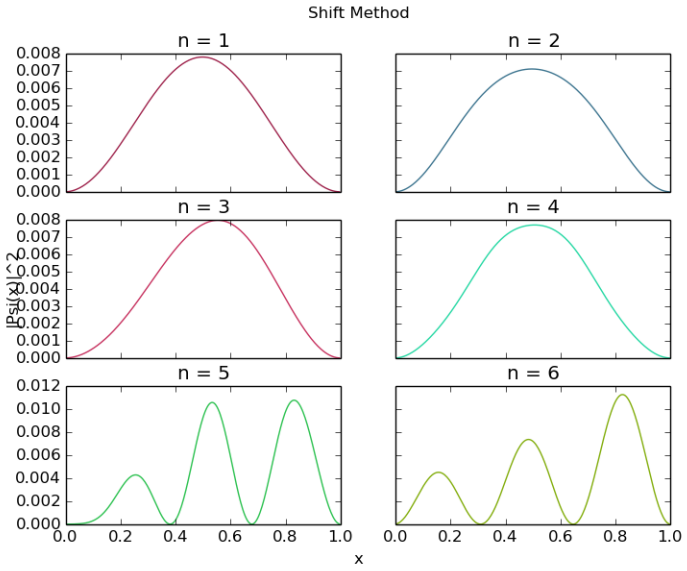
Figuur 21: De error op de waarschijnlijkheidsdichtheid voor opeenvolgende V-cycles

We zien iets soortgelijks als bij de error analyse op de weighted Jacobi methode. Voor de lagere toestanden hebben we een gladdere begingok waardoor dat opeenvolgende V-cycles vrij weinig effect hebben op de error, zelfs in het begin. Als we zien naar de n = 5 en n = 6 toestanden, die meer oscillatorisch zijn, dan observeren we dat er initeel een sterke daling in error is, maar ook hierna hebben opeenvolgende V-cycles weinig effect op de error. Dit is een jammer resultaat omdat we net multigrid hebben ontwikkeld om ervoor te zorgen dat de error altijd blijft verbeteren door telkens naar groffere grids te gaan.

2.1.7 Het grofste grid in de V-cycle

Er is nog 1 element van de multigrid methode die nog niet is besproken geweest en dat is het grofste grid waarop we een directe methode toepassen om de oplossing van het stelsel vergelijkingen te vinden.

Al de resultaten die we tot nu toe hebben verkregen maakten gebruik van een n = 16 grid als laagste niveau in de V-cycle. Het standaard laagste niveau voor de V-cycle is een n = 2 grid. Als we dit gebruiken dan krijgen we als resultaten:



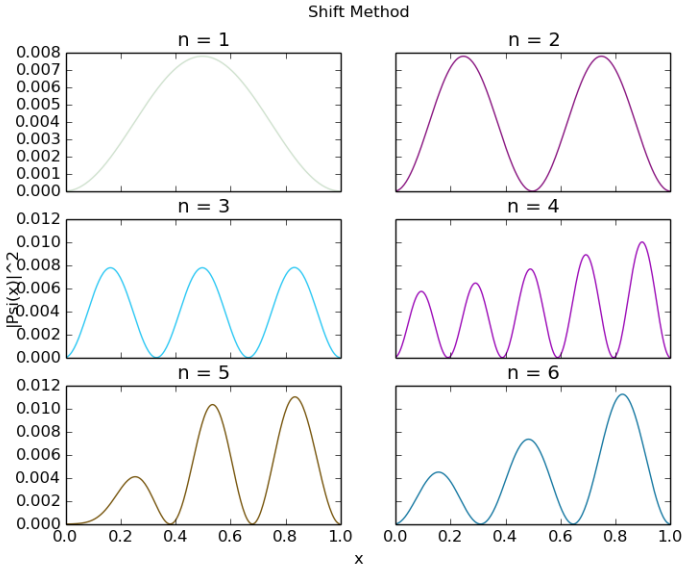
Figuur 22: Bekomen eigentoestanden met n = 2 als grofste grid in V-cycle

en voor de eigenwaarden: [0.99222154, 1.00760007, 1.01481353, 0.99472865, 10.41611622, 8.66668108]

Het is duidelijk dat deze resultaten zeer slecht zijn. De eerste eigentoestand is nog ok, maar de volgende zijn zeer slecht.

Dit kan waarschijnlijk worden verklaard doordat de grote oscillaties niet weergegeven kunnen worden op zo'n grof grid. Dit is het probleem van aliasing dat in het begin kort werd vermeld. De grotere modes kunnen niet goed genoeg worden weergegeven op een n = 2 grid.

Als we als grofste grid een n = 8 grid kiezen dan krijgen we het volgende te zien:



Figuur 23: Bekomen eigentoestanden met n = 8 als grofste grid in V-cycle

en voor de eigenwaarden: [0.99222076, 3.96873779, 8.92909231, 24.60848801, 10.33116699, 8.66669132]

We zien dat nu ook de 2e en de 3e eigentoestand correct zijn, maar de hogere modes zijn nog steeds niet in orde. Dit resultaat toont ons dat het inderdaad een aliasing probleem is.

Doordat het laagste niveau van de V-cycle dus niet probleem invariant is, zorgt dit voor moeilijkheden om een algemene multigrid methode te kunnen opstellen voor de Schrödinger vergelijking. Er is kennis nodig van de vorm van de eigentoestanden van het probleem vooraleer de multigrid methode hierop kan worden toegepast. Deze kennis is in vele gevallen niet verkrijgbaar.

2.2 Het k-dot-p model voor het berekenen van de elektonische bandenstructuur van GaAs

2.2.1 Fysische schets

In het vorige hoofdstuk hebben we het deeltje in een oneindige put besproken. Dit geeft ons de oplossingen voor een deeltje die opgesloten zit in een potentiaal put waarvan de wanden oneindig hoog zijn. De oneindigheid van de potentiaal is op zich al niet fysisch, maar het probleem beschouwde ook geen spin of spin-baan koppeling wat uiteraard relevant is.

Wat we nu gaan beschouwen is een deeltje in GaAs die opgesloten zit langs beide kanten door een andere halfgeleider met een grotere bandgap dan GaAs. In ons geval zal dit gebeuren door GaAs te sandwichen tussen $Al_xGa_{1-x}As$. Dit zorgt ervoor dat in de richting waarin de opsluiting gebeurd het probleem zich herleid naar een deeltje in een put, maar deze keer zijn de wanden van de put van eindige hoogte.

Het deeltje in GaAs zal niet enkel de eindige potentiaalput aan weerszijden van het kristal voelen, maar zal ook een periodische potentiaal van in het kristal zelf voelen.

Doordat de potentiaal in het kristal periodisch is dan zal volgens het Bloch theorema de golffunctie er als volgt uitzien:

$$\Psi_{\boldsymbol{k}}(\boldsymbol{r}) = e^{i\boldsymbol{k}\cdot\boldsymbol{r}} u_{\boldsymbol{k}}(\boldsymbol{r}) \tag{85}$$

waarbij dat $u_{\boldsymbol{k}}(\boldsymbol{r})$ dezelfde periodiciteit heeft als de potentiaal/het rooster.

De afleiding van de Hamiltoniaan zal in dit werk niet gebeuren.

De hamiltoniaan heeft de volgende vorm:

$$H = H_0 + \frac{\hbar^2 k^2}{2m_0} + \frac{\hbar}{4m_0^2c^2}\nabla V \times \boldsymbol{p}\cdot\boldsymbol{\sigma} + H' \tag{86}$$

$$H_0 = \frac{p^2}{2m_0} + V(\boldsymbol{r}) \tag{87}$$

$$H' = \frac{\hbar}{m_0}\boldsymbol{k}\cdot\boldsymbol{\Pi} \tag{88}$$

$$\boldsymbol{\Pi} = \boldsymbol{p} + \frac{\hbar}{4m_0c^2}\boldsymbol{\sigma}\times\nabla V \tag{89}$$

waarbij dat de termen met $\boldsymbol{\sigma}$ de spin-baan interacties voorstellen en $\boldsymbol{\sigma}$ de Pauli matrices zijn.

In dit model zitten 3 valentie banden: De Heavy Hole, Light Hole en Split Off. De conductieband wordt niet in rekening gebracht. Doordat de spin in dit model zit verwerkt en omdat de beschouwde deeltjes spin-$\frac{1}{2}$ deeltjes zijn zal elke band tweevoudig ontaard zijn waardoor we dus 6 banden in totaal beschouwen.

Het 6 banden model kan dan als een matrix worden uitgeschreven die de Luttinger-Kohn hamiltoniaan wordt genoemd. Deze ziet er als volgt uit:

$$\bar{\bar{\boldsymbol{H}}}^{\boldsymbol{LK}} = \begin{bmatrix} P+Q & -S & R & 0 & \frac{-S}{\sqrt{2}} & \sqrt{2}R \\ -S^\dagger & P-Q & 0 & R & -\sqrt{2}Q & \sqrt{\frac{3}{2}}S \\ R^\dagger & 0 & P-Q & S & \sqrt{\frac{3}{2}}S^\dagger & \sqrt{2}Q \\ 0 & R^\dagger & S^\dagger & P+Q & -\sqrt{2}R^\dagger & \frac{-S^\dagger}{\sqrt{2}} \\ \frac{-S^\dagger}{\sqrt{2}} & -\sqrt{2}Q^\dagger & \sqrt{\frac{3}{2}}S & -\sqrt{2}R & P+\Delta & 0 \\ \sqrt{2}R^\dagger & \sqrt{\frac{3}{2}}S^\dagger & \sqrt{2}Q^\dagger & \frac{-S}{\sqrt{2}} & 0 & P+\Delta \end{bmatrix} \tag{90}$$

Als we de split-off band niet mee beschouwen dan krijgen we het 4 banden model die er als volgt uitziet:

$$\bar{\bar{\boldsymbol{H}}}^{\boldsymbol{LK}} = \begin{bmatrix} P+Q & -S & R & 0 \\ -S^\dagger & P-Q & 0 & R \\ R^\dagger & 0 & P-Q & S \\ 0 & R^\dagger & S^\dagger & P+Q \end{bmatrix} \tag{91}$$

met:

$$\begin{aligned} P &= \frac{\hbar^2\gamma_1}{2m_0}(k_x^2+k_y^2+k_z^2) \\ Q &= \frac{\hbar^2\gamma_2}{2m_0}(k_x^2+k_y^2-2k_z^2) \\ R &= \frac{\hbar^2}{2m_0}\left[-\sqrt{3}\gamma_2(k_x^2-k_y^2)+i2\sqrt{3}\gamma_3 k_x k_y\right] \\ S &= \frac{\hbar^2\gamma_3}{m_0}\sqrt{3}(k_x-ik_y)k_z \end{aligned}$$

Omdat we het probleem numeriek willen oplossen moet dit uiteraard gediscretiseerd worden, maar herinner dat we de volgende substitutie kunnen invoeren:

$$k_j \to -i\frac{\partial}{\partial x_j} \tag{92}$$

Als we kiezen om te werken in de energie eenheid:

$$E_0 = \frac{\hbar^2\pi^2}{2m_0L^2} \tag{93}$$

dan kunnen de P, Q, R en S termen in de Hamiltoniaan in eenheidsloze vorm geschreven worden als:

$$\begin{aligned} P &= \frac{\gamma_1}{\pi^2}\left(k_y^2+k_z^2-\frac{d^2}{dx^2}\right) \\ Q &= \frac{\gamma_2}{\pi^2}\left(k_y^2-2k_z^2-\frac{d^2}{dx^2}\right) \\ R &= \frac{1}{\pi^2}\left[-\sqrt{3}\gamma_2(-\frac{d^2}{dx^2}-k_y^2)+2\sqrt{3}\gamma_3\frac{d}{dx}k_y\right] \\ S &= \frac{2\gamma_3}{\pi^2}\sqrt{3}(-i\frac{d}{dx}-ik_y)k_z \end{aligned}$$

waarbij er is gekozen voor een opsluiting in de x-richting waardoor enkel k_x geschreven wordt als $-i\frac{\partial}{\partial x}$.

Als dit nu gediscretiseerd wordt gebruikmakende van de eindige differentie methode dan krijgen we volgend gediscretiseerd probleem:

$$P\psi = \frac{\gamma_1}{\pi^2}\left(k_y^2 + k_z^2 + \frac{2}{h^2}\right)\psi_i - \frac{\gamma_1}{\pi^2 h^2}\psi_{i+1} - \frac{\gamma_1}{\pi^2 h^2}\psi_{i-1} \tag{94}$$

$$Q\psi = \frac{\gamma_2}{\pi^2}\left(k_y^2 - 2k_z^2 + \frac{2}{h^2}\right)\psi_i - \frac{\gamma_2}{\pi^2 h^2}\psi_{i+1} - \frac{\gamma_2}{\pi^2 h^2}\psi_{i-1} \tag{95}$$

$$R\psi = \frac{\sqrt{3}\gamma_2}{\pi^2}\left(-\frac{2}{h^2} + k_y^2\right)\psi_i + \frac{\sqrt{3}}{\pi^2}\left(\frac{\gamma_2}{h^2} + \frac{k_y\gamma_3}{h}\right)\psi_{i+1} + \frac{\sqrt{3}}{\pi^2}\left(\frac{\gamma_2}{h^2} - \frac{k_y\gamma_3}{h}\right)\psi_{i-1} \tag{96}$$

$$S\psi = -\frac{2\sqrt{3}\gamma_3}{\pi^2} ik_z k_y \psi_i - \frac{\sqrt{3}\gamma_3}{\pi^2 h} ik_z \psi_{i+1} + \frac{\sqrt{3}\gamma_3}{\pi^2 h} ik_z \psi_{i-1} \tag{97}$$

Dit bied nu de mogelijkheid om het probleem numeriek op te lossen. Merk op dat het k-dot-p model een perturbatief model is, waardoor er geen exacte oplossing aanwezig is. Dit zal zich uiten in het feit dat we de error nu niet kunnen gebruiken als maat voor de goedheid van de multigrid methode.

Dit werd in mijn bachelorthesis al uitgewerkt in verschillende Python klasses om deze matrix op te stellen. Deze code kon zonder enige aanpassing meteen worden gebruikt om multigrid op toe te passen.

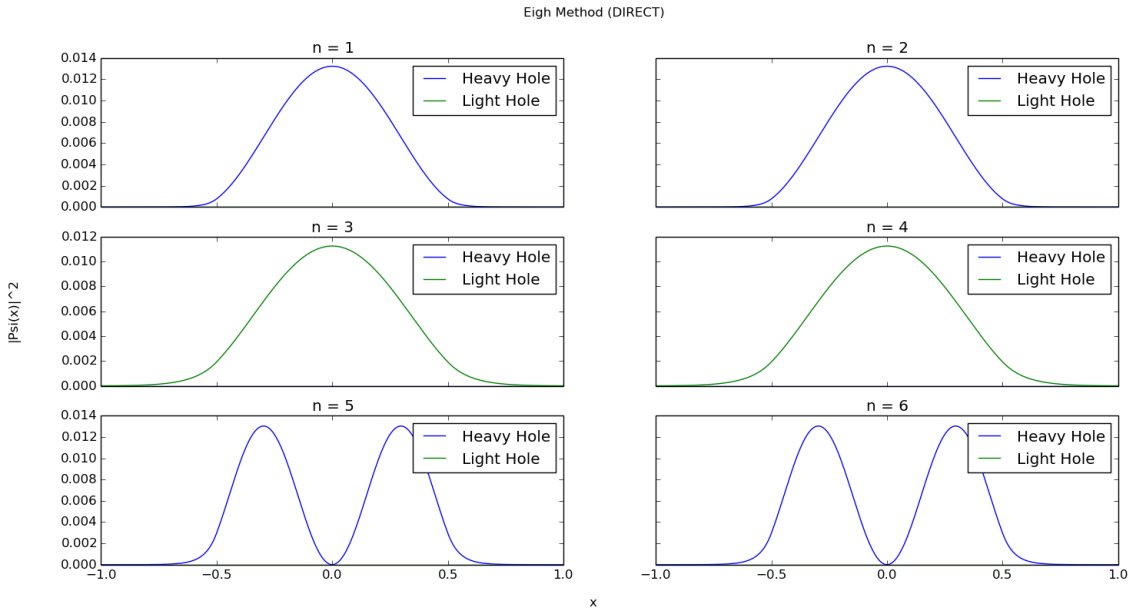
Omdat het voor dit werkje niet uitmaakt of we het 4-banden of 6-banden model beschouwen kiezen we om met het 4-banden model te werken.

2.2.2 Directe methode in Matlab en Numpy/Scipy

Wanneer dit probleem werd opgelost voor mijn bachelorthesis werd er enkel maar gebruik gemaakt van de reeds geimplementeerde Eigsh functie in Matlab en Numpy/Scipy die een eindig aantal eigenwaarden en eigenvectoren van sparse hermitische matrices bepaald gebruikmakende van de impliciet herstarte Lanczos methode.

Omdat die methode op zich al een iteratieve methode is, en omdat ik deze methode zelf gebruik in het begin van de multigrid methode leek het me raar om dit te gebruiken. Er werd gekozen om een dense hamiltoniaan te maken en deze met een directe methode op te lossen.

De eerste 6 eigentoestanden die we hiermee vinden zijn:



Figuur 24: De eerste 6 eigentoestanden van het 4-banden model

met de eigenwaarden (in meV): [7.1880383898490017, 7.1880383898490017, 22.252915198535629, 22.252915198535629, 28.48381132649094, 28.48381132649094]

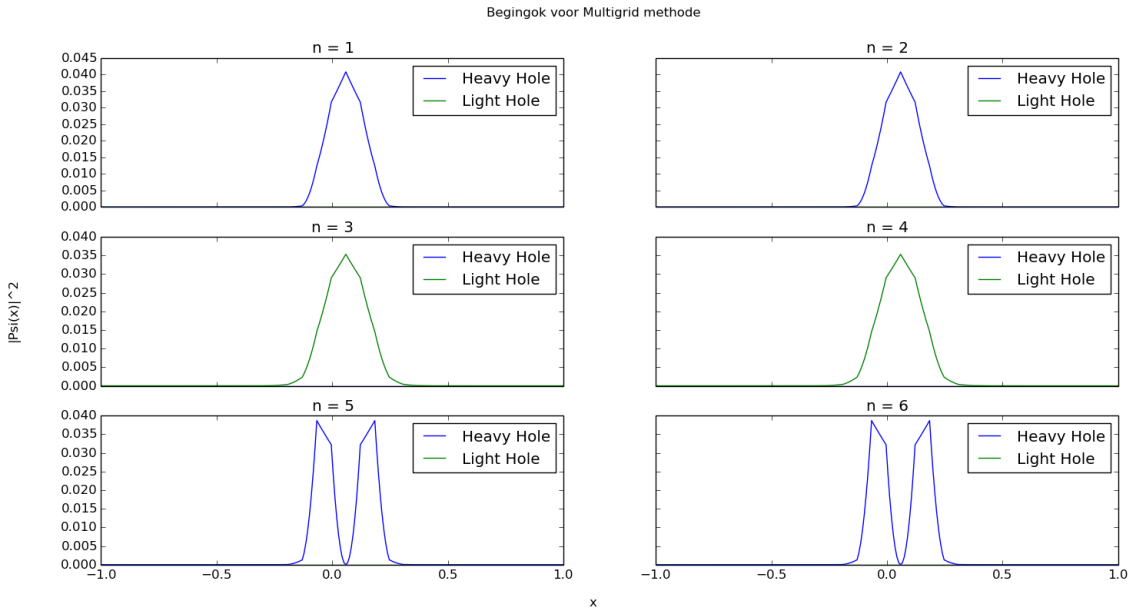
We zien al meteen de twee-voudige ontaarding van elke toestand die eerder werd aangehaald.

We zien dat de resultaten sterk lijken op die van een deeltje in een oneindige put en dit is ook niet zo verbazingwekkend aangezien we nu ook een deeltje hebben opgesloten in een potentiaalput, maar deze is nu eindig waardoor de waarschijnlijkheidsdichtheid niet naar 0 wordt geforceerd op de rand, maar exponentieel daalt vanaf het de wand raakt.

2.2.3 Iteratief bekomen resultaat

Zoals we hebben gezien bij het deeltje in een oneindige put, zijn er nu 3 grids waarmee gespeeld kan worden: het fijnste grid waarop we de oplossing willen weten, het groffe grid waarop de begingok wordt gevonden en het grofste grid in de V-cycle. We zullen in de analyse spelen met elk niveau. De volgende beste resultaten zijn bekomen met het fijnste grid op n = 256 en zowel het groffe grid voor de begingok als het grofste niveau in de V-cycle op n = 32.

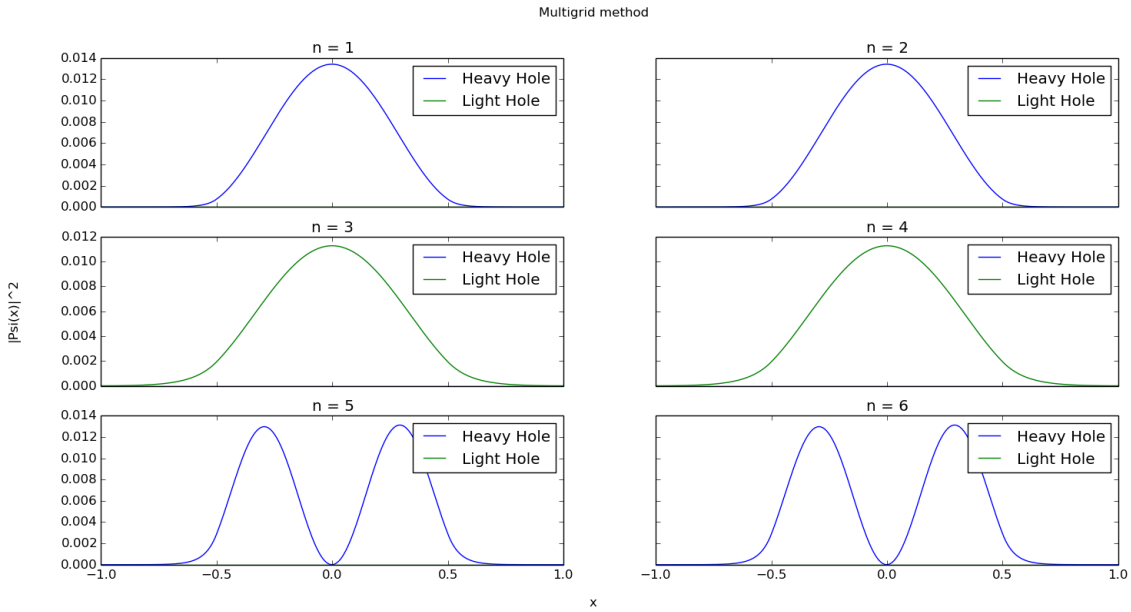
De geinterpoleerde begingokken zagen er als volgt uit:



Figuur 25: De geinterpoleerde begingokken afkomstig van een n = 32 grid

met de eigenwaarden: [6.86373362, 6.86373362, 22.52575977, 22.52575977, 25.99002308, 25.99002308]

Als de multigrid methode hierop werd toegepast werd het volgende resultaat bekomen:



Figuur 26: De eerste 6 eigentoestanden bekomen met de multigrid methode met fijn grid n = 256

met de eigenwaarden: [7.19194015, 7.19193938, 22.2564022, 22.25651056, 28.49233372, 28.49233555]

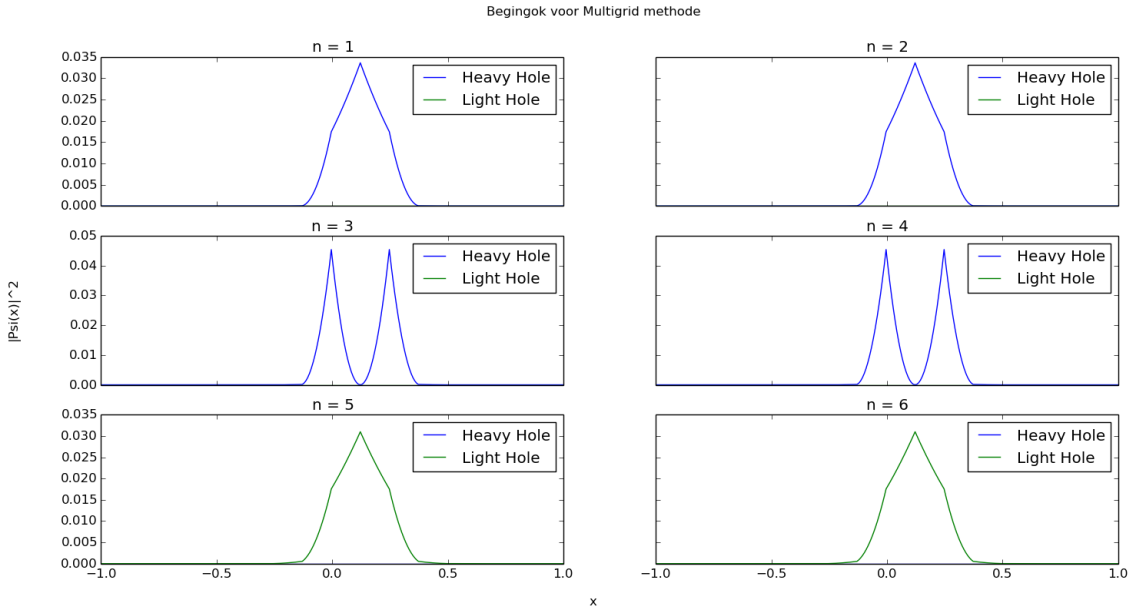
We zien dat dit resultaat zeer goed overeenkomt met de resultaten bekomen uit de directe methode.

2.2.4 Aanpassing van de 3 verschillende niveaus

Zoals eerder vermeld zijn er 3 verschillende grids die we kunnen aanpassen en die een effect zullen hebben op de accuraatheid van de resultaten.

Het eerste grid dat we zullen aanpassen is het groffe grid waaruit we de begingokken halen. We zullen dit grid verkleinen naar n = 16 gridpunten en het effect hiervan bekijken.

De geinterpoleerde begingokken zien er als volgt uit:

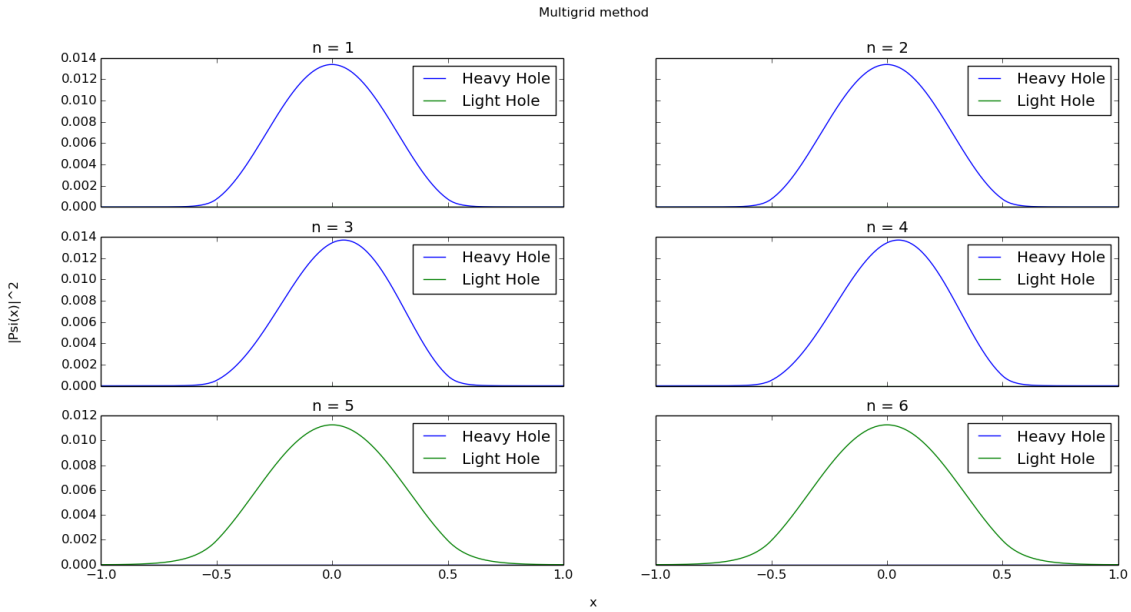


Figuur 27: Geinterpoleerde begingok afkomstig van het n = 16 grid

met de eigenwaarden: [4.02526561, 4.02526561, 13.98158257, 13.98158257, 14.83297303, 14.83297303]

Als we deze begingokken voor de eigenwaarden bekijken en vergelijken met de juiste eigenwaarden dan kunnen we al verwachten wat er zal gebeuren met de resultaten als we ons de resultaten van het deeltje in een oneindige put herinneren waarbij dat het groffe begingok grid te grof werd gemaakt.

De multigrid resultaten zien er als volgt uit:



Figuur 28: Eigentoestanden na multigrid methode

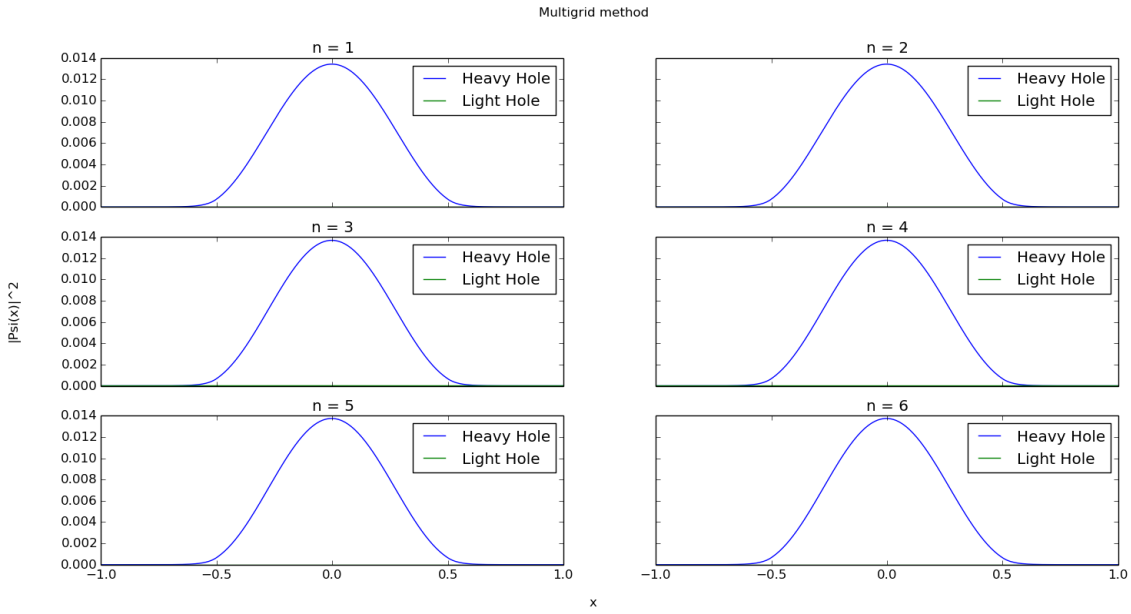
met eigenwaarden: [7.19103234, 7.19103314, 7.31073014, 7.31072926, 22.25715549, 22.25719659]

Zoals verwacht uit ervaring met het deeltje in een oneindige put, zijn de eerste 4 toestanden allemaal naar dezelfde toestand geconvergeerd doordat de gok van de eigenwaarde dichter bij de eerste toestand lag dan bij de andere.

Wat wel moet worden opgmerkt is dat de gokken voor de n = 3 en n = 4 toestanden eigenlijk de n = 5 en n = 6 toestanden voorstellen. Waarom deze op het groffe grid opeens van plaats veranderen weet ik niet.

Het volgende grid dat we zullen aanpassen is het grofste niveau van de V-cycle. We hebben al gezien dat dit bij de oneindige put voor problemen kan zorgen.

We kiezen nu het grofste niveau van de V-cycle op n = 16 gridpunten. Dit geeft als resultaat:



Figuur 29: Multigrid resultaten voor een grof V-cycle niveau van n = 16

met de eigenwaarden: [7.19225534, 7.1922458, 7.222635, 7.21558327, 7.2178878, 7.21787106]

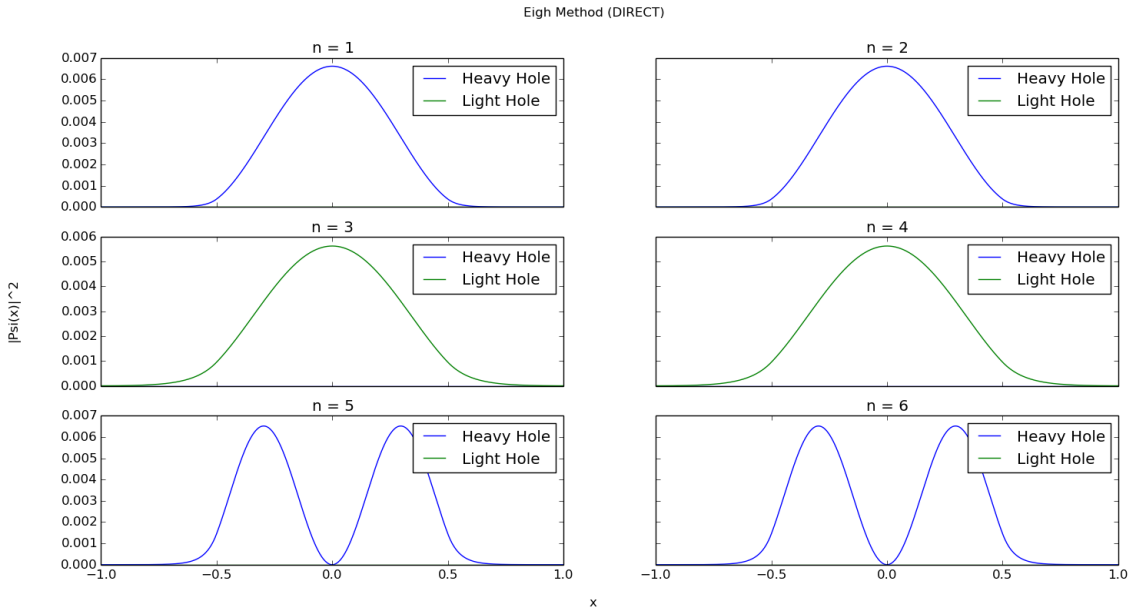
We zien meteen dat dit resultaat echt slecht is. Elke toestand is geconvergeerd naar de eerste toestand. Bij het oneindige put probleem leek dit op een aliasing probleem doordat de lage modes wel juist convergeerde, maar de hogere modes niet.

Deze uitleg is moeilijk te verbinden met de huidige resultaten. De andere eigentoestanden zijn niet zo oscillatorisch en er zouden genoeg gridpunten zijn om deze oscillaties te kunnen weergeven. Het lastige is ook dat we dit niet beter kunnen bestuderen door het grid een klein beetje fijner te maken omdat we dan al meteen op het goede resultaat voor alle modes komen.

Bij de oneindige put hadden we het groffe niveau een beetje fijner gemaakt en we zagen dat er telkens een paar extra modes juist convergeerden.

Ondanks dat dit slechte resultaat lastig te bestuderen is, blijft dezelfde conclusie geldig als bij de oneindige put: het grofste niveau van de V-cycle is niet probleem onafhankelijk, waardoor dit voor moeilijkheden kan zorgen bij het trachten oplossen van problemen waar weinig informatie a priori aanwezig is.

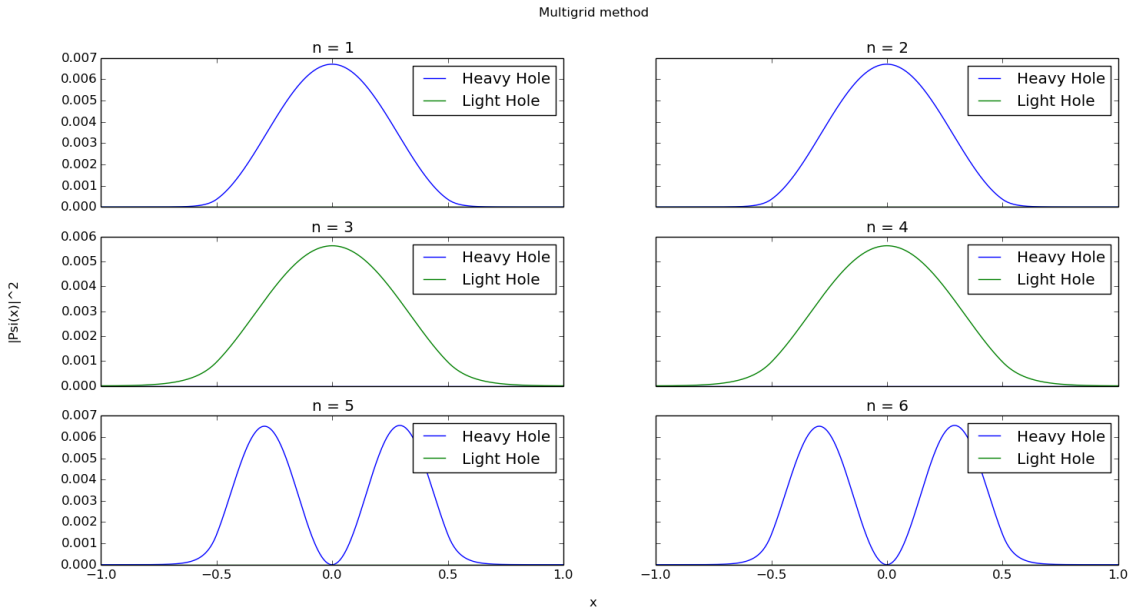
Het laatste niveau dat we kunnen aanpassen is het fijne grid. Als we het fijnste niveau nu n = 512 gridpunten geven dan krijgen we voor de directe methode volgende resultaten:



Figuur 30: Eigentoestanden voor n = 512 gebruikmakende van de directe methode

met de eigenwaarden: [7.1897399053934503, 7.1897399053934503, 22.257595259706761, 22.257595259706761, 28.491830894100818, 28.491830894100818]

De resultaten voor de multigrid methode zijn dan:



Figuur 31: Eigentoestanden voor een fijn grid met n = 512 gebruikmakende van multigrid

met de eigenwaarden: [7.1935961, 7.19359567, 22.26109673, 22.26115752, 28.50091123, 28.5009122]

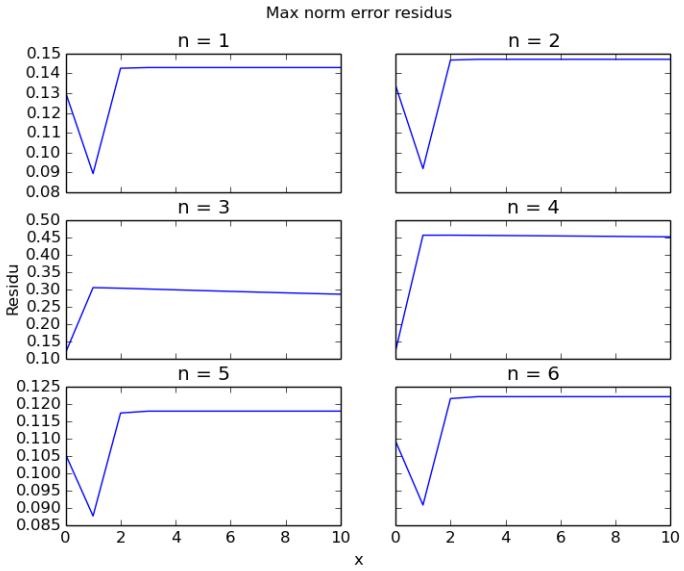
Er is geen significant verschil op te merken door een fijner grid te kiezen als fijnste niveau. Eens dat het grid waarop we de oplossing willen weten fijn genoeg is, zal een fijner grid geen verschil brengen in de resultaten.

2.2.5 Convergentie analyse

In tegenstelling tot het onderzochte oneindige put probleem is er voor dit probleem geen exacte oplossing voorhande. Dit betekent dat er geen toegang is tot de error. We hebben in de inleiding gezien dat als de error niet toegankelijk is we het residu kunnen gebruiken als convergentie maat. Uiteraard moet voor het residu wel rekening gehouden worden dat een klein residu niet noodzakelijk een kleine error betekend.

Als we het residu gebruiken mogen we dus niet zien naar de numerieke waarde (tenzij deze 0 is want dan is de error ook 0), maar enkel naar het gedrag van het residu.

Wat we telkens laten zien is de max norm error van de genormeerde residu na elke V-cycle. Dit ziet er als volgt uit:

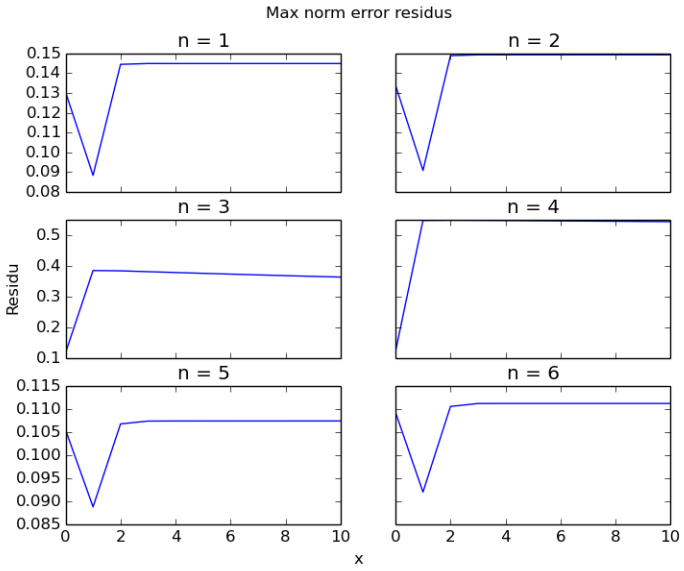


Figuur 32: De max norm residuen voor de laagste 6 toestanden

Het gedrag van het residu is zeer vreemd. Bij 4 van de 6 zien we na de 1e V-cycle een daling in residu waarna het residu terug stijgt en dan zo goed als niet meer veranderd. Bij de andere 2 stijgt het residu na de 1e V-cycle en vertoont dan hetzelfde quasi constante gedrag.

Dit constante gedrag is iets dat we ook hebben gezien bij het deeltje in een oneindige put na een aantal V-cycles.

Het is lastig om de resultaten te kunnen verbeteren aangezien de 3 grids waarmee we kunnen spelen al moeten worden vastgezet om toch nog het juiste resultaat te kunnen verkrijgen. We kunnen nog wel met een andere smoother werken. Als we in plaats van de Weighted Jacobi smoother de Gauss-Seidel smoother gebruiken dan vinden we:



Figuur 33: De max norm residuen voor de laagste 6 toestanden gebruikmakende van de Gauss-Seidel smoother

Zoals we zien is het resultaat praktisch hetzelfde. Wat ons dus geen extra informatie oplevert over waarom het residu dit gedrag heeft.

3 2D problemen

3.1 De oneindige potentiaalput

Het oneindige potentiaalput probleem dat we in 1D hebben opgelost kan uitgebreid worden naar 2D waarbij dat het nog steeds exact oplosbaar is. De hamiltoniaan ziet er gelijkaardig uit aan het 1D probleem en ziet er als volgt uit:

$$H = \frac{p_x^2}{2m} + \frac{p_y^2}{2m} + V(x,y) \tag{98}$$

waarbij:

$$V(x,y) = \begin{cases} 0, & 0 < x < L_x \quad \text{en} \quad 0 < y < L_y \\ \infty, & elders \end{cases}$$

3.1.1 Exacte resultaat

De exacte eigentoestanden worden geschreven als:

$$\Psi_{n_x,n_y} = \sqrt{\frac{4}{L_x L_y}} \sin\left(\frac{n_x \pi x}{L_x}\right) \sin\left(\frac{n_y \pi y}{L_y}\right) \tag{99}$$

waarbij dat de eigenwaarden worden gegeven door:

$$E_{n_x,n_y} = \frac{\hbar^2}{2m}\left[\left(\frac{n_x \pi}{L_x}\right)^2 + \left(\frac{n_y \pi}{L_y}\right)^2\right] \tag{100}$$

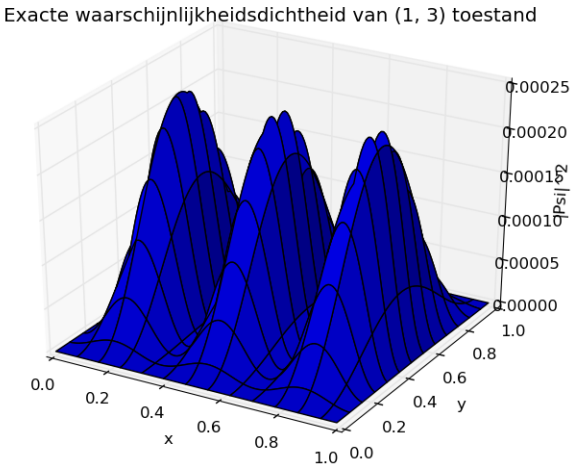
waarbij we het probleem in eenheidsloze dimensies zullen oplossen door volgende energie eenheid in te voeren:

$$E_0 = \frac{\hbar^2 \pi^2}{2mL^2} \tag{101}$$

waarbij we hebben verondersteld dat $L_x = L_y$. Dit geeft voor de energie dan:

$$E_{n_x,n_y} = E_0 \left[n_x^2 + n_y^2\right] \tag{102}$$

De toestand met $n_x = 1$ en $n_y = 3$ ziet er als volgt uit:



Figuur 34: Exact resultaat voor (1,3) toestand

welke de dimensieloze eigenwaarde van 10 heeft.

3.1.2 Iteratief bekomen resultaat

We zien dat de Hamiltoniaan nu 2-dimensionaal is wat ons de volgende Schrödinger vergelijking oplevert:

$$-\frac{\hbar^2}{2m}\left[\frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2}\right]\Psi(x,y) = E_{n_x,n_y}\Psi(x,y) \tag{103}$$

Als we de eindige differentie methode gebruiken om deze vegelijking te discretiseren dan bekomen we:

$$-\frac{\hbar^2}{2m}\frac{1}{h^2}\left(\Psi_{i+1,j} - 4\Psi_{i,j} + \Psi i - 1, j + \Psi i, j + 1 + \Psi i, j - 1\right) = E\Psi_{i,j} \tag{104}$$

Om na te gaan of de 2D Laplace operator correct werd geimplementeerd werden de eigenwaarden gecontroleerd die we krijgen als we de hamiltoniaan onderwerpen aan de reeds geimplementeerde methode om eigenwaarden te berekenen voor sparse hermitische matrices (Impliciet Herstarte Lanczos methode). De kleinste 6 eigenwaarden op een grid met n = 128 punten zijn: [1.96901511, 4.92195391, 4.92195391, 7.87489271, 9.84157268, 9.84157268]

De exacte eigenwaarden zijn: [2, 5, 5, 8, 10, 10]

Waaruit we kunnen afleiden dat de 2D Laplace operator correct is geimplementeerd.

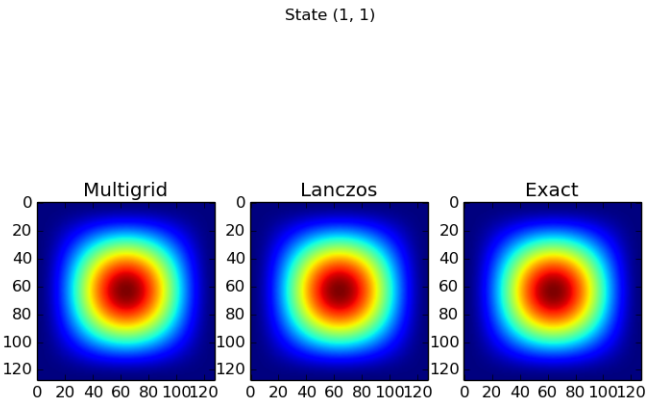
Een makkelijkere manier om de juiste 2D Laplace operator te bekomen is om te beginnen van de 1D Laplace operator en gebruik te maken van het volgende:

$$\hat{\Delta}_{2D} = \hat{\Delta}_{1D} \oplus \hat{\Delta}_{1D} \tag{105}$$

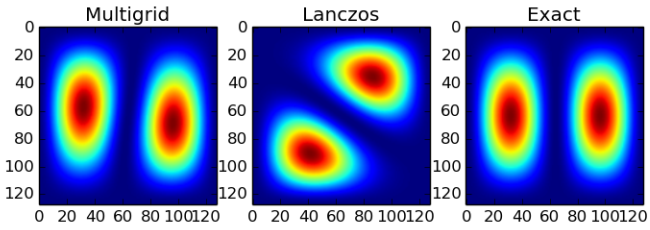
De parameters die werden gebruikt waren: het fijne grid had n = 128 (n = 256 geeft al geheugen errors), het groffe grid had n = 16, het laagste niveau van de vcycle had n = 8. De begingokken die werden verkregen op het groffe grid waren voor de energie : [1.76659015, 4.38639582, 4.38639582, 7.00620149, 8.65349775, 8.65349775].

De uiteindelijke energien gevonden met de multigrid methoden waren: [1.96901685, 4.92197738, 4.92197786, 7.87499665, 9.84169385, 9.84166931] vergeleken met de exacte waarden: [2, 5, 5, 8, 10, 10] komen deze vrij goed overeen. De energieen gevonden met enkel de Impliciet herstarte Lanczos methode zijn: [1.96901511, 4.92195391, 4.92195391, 7.87489271, 9.84157268, 9.84157268] wat goed overeenkomt met de waarden die we zijn bekomen met multigrid.

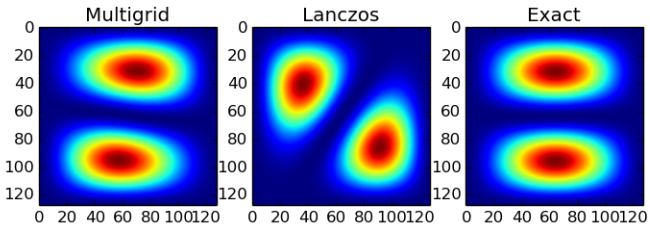
De eerste 6 toestanden bekomen doormiddel van multigrid werden naast de exacte oplossing en de oplossing van de Lanczos methode gezet om ze te kunnen vergelijken:



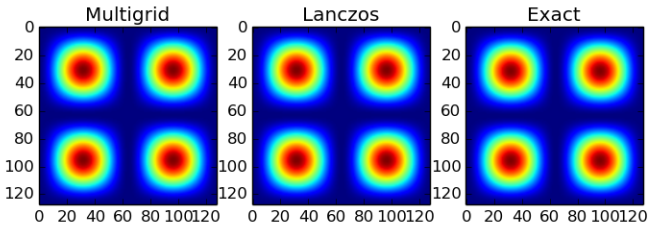
State (1, 2)



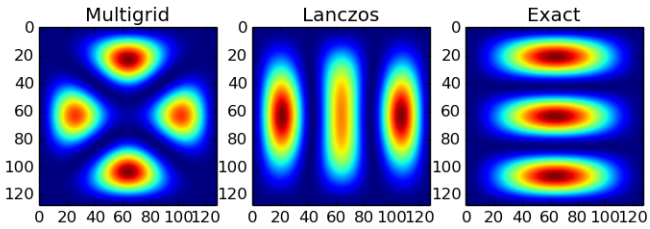
State (2, 1)

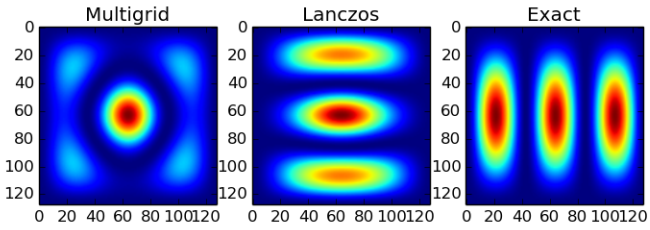


State (2, 2)



State (1, 3)





Wat bij deze figuren moet worden opgemerkt is dat bij de ontaarde toestanden ((1,2) en (2,1) en (1,3) en (3,1)) de eigentoestanden soms niet overeenkomen met de exacte toestand die ernaast staat geplot. Soms convergeert de Lanczos methode naar de andere toestand wat geen fout is in het berekenen, maar dit zorgt ervoor dat de plots soms fout zijn.

We zien dat de eigentoestanden (herinner dat we de dichtheden plotten en niet de eigenvectoren zelf) afkomstig uit de multigrid methode tot aan de (1,3) en (3,1) toestand redelijk goed overeenkomen met de exacte oplossing. Ze komen niet slechter overeen met de exacte dan de eigentoestanden die afkomstig zijn uit de Lanczos methode.

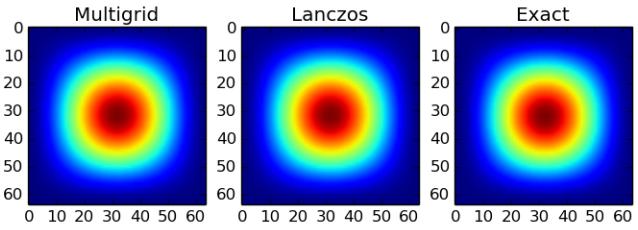
Voor de laatste twee toestanden is het duidelijk dat de Lanczos methode nog de beste oplossing geeft. Later zullen we zien wat er gebeurt als we het aantal gridpunten halveren.

De eigentoestanden van de laatste twee toestanden afkomstig van de multigrid methode zijn niet echt te vergelijken met de exacte toestand. Merk wel op dat de eigenwaarden van de laatste twee toestanden berekend met de multigrid methode praktisch hetzelfde zijn als deze berekend met de Lanczos methode. Waarom de dictheden dan zo verschillend zijn is niet gekend en misschien zouden meer gridpunten hier verbetering in brengen (iets wat niet mogelijk is aangezien een verdubbeling ons geheugenfouten geeft).

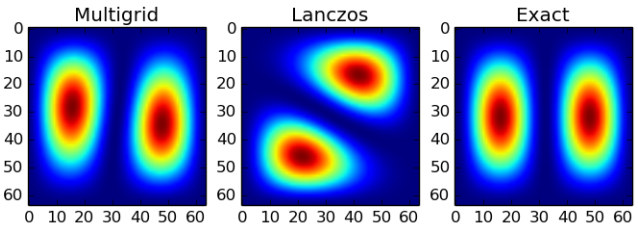
3.1.3 Het veranderen van het fijnste niveau

Om wat meer inzicht te krijgen in de vergelijking tussen de Lanczos en de multigrid toestanden zullen we het fijnste grid van n = 128 naar n = 64 verlagen. Dit geeft de volgende resultaten:

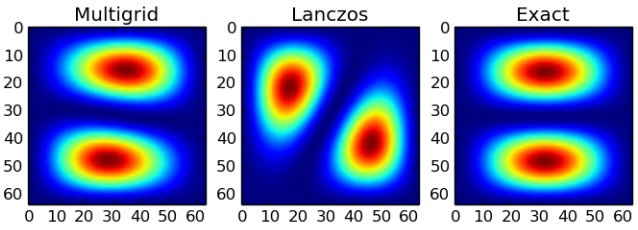
State (1, 1)



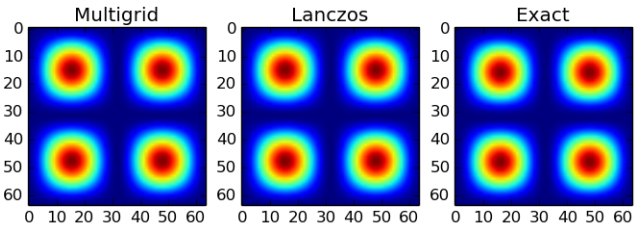
State (1, 2)



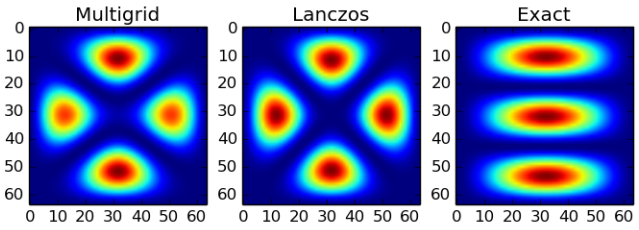
State (2, 1)



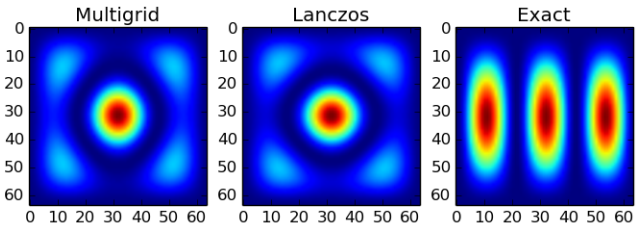
State (2, 2)



State (1, 3)



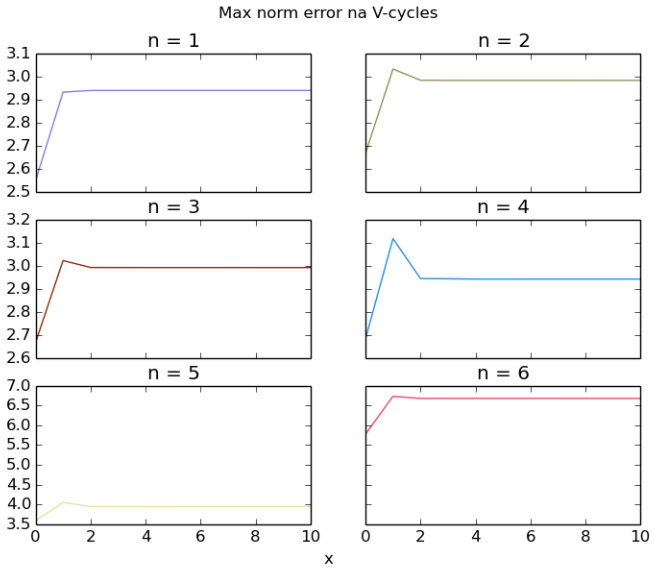
State (3, 1)



Wat ons opvalt is dat als we nu kijken naar de laatste twee toestanden we voor de Lanczos toestanden als voor de Multigrid toestanden ongeveer hetzelfde gedrag zien. Het lijkt erop dat indien we meer gridpunten hebben dat ook de multigrid toestanden zullen convergeren naar de juiste toestanden. Dit resultaat toont ons ook dat de Lanczos methode sterker wordt beinvloed door een verdubbeling in het aantal gridpunten.

3.1.4 Error analyse

Als we 10 vcycles doen en telkens de max-norm van de error op de dichtheden berekenen dan krijgen we voor de eerste 6 toestanden het volgende te zien:



Wat al meteen opgemerkt wordt is dat de error stijgt na de eerste vcycle, wat zeer vreemd is. Hierna heeft bijna elke toestand hetzelfde gedrag, eerst een maximum waarna het dan daalt en constant blijft. Hieruit kunnen we besluiten dat veel vcyles doen geen zin heeft, maar ook dat de error niet verkleint met meerdere vcycles wat een vreemd resultaat is. Dit vreemde constante error gedrag zagen we ook bij de beschouwde 1D problemen.

4 Volgende stappen

In de huidige implementatie van de multigrid methode wordt de begingok gehaald uit de impliciet herstarte Lanczos methode welke een Krylov methode is. Het zou interessant zijn om de Krylov methode volledig uit de multigrid methode te halen zodat we beide methodes kunnen vergelijken met elkaar. Dit brengt het probleem met zich mee dat we ergens anders begingokken moeten halen voor onze multigrid methode. Een mogelijkheid hiervoor kan bij padintegralen worden gevonden.

Een andere aanpassing aan de multigridmethode is om de discretizatie van de laplaciaan te veranderen. In elk kwantummechanisch probleem is een laplaciaan nodig. Op dit moment wordt de laplaciaan door een simpele eindige differentie methode gemaakt. Er zijn 9-punts stencils voor de laplaciaan die misschien betere resultaten kan geven. Ook word er in andere papers gebruik gemaakt van de Mehrstellen operator wat ook meer lokale informatie gebruikt voor de laplaciaan dan de huidige implementatie.

Er zijn niet veel artikels te vinden over het gebruik van multigrid methoden om bandenstructuren te berekenen, wat uiteindelijk een doel is van mij. De artikels die dit wel behandelen bekomens telkens wel goede resultaten.

Het oorspronkelijke idee om multigrid toe te passen op de Schrödinger vergelijking kwam van een paper [2] die multigrid methodes toepaste op gecondenseerde materie problemen. De onderzoekers waren in staat om op een efficiente manier problemen in supercellen op te lossen. Dit waren problemen waarbij dat interacties tussen een 100-tal atomen aanbod kwamen. Ze gebruikten hierbij een grid in de reele ruimte als basis wat computationeel zeer intensief is.

Met de kennis vergaard uit dit vak en met het maken van dit werk hoop ik om dit uitgebreid artikel beter te kunnen verstaan en misschien de kennis van multigrid te kunnen meenemen naar de gecondenseerde materie fysica onderzoeksgroep waardat ze al vaak op problemen zijn gestoten doordat hun supercell niet groot genoeg kon gemaakt worden.

5 Besluit

Doordat er 3 verschillende grid niveaus zijn die in deze multigrid methode moeten worden aangepast, is het lastig om een algemene methode op te stellen voor het oplossen van de Schrödinger vergelijking. Voor elk probleem moeten er andere fijnheden van grids worden gekozen. Dit wordt uiteraard heel lastig voor problemen waar er a priori geen voldoende kennis over is. Hierdoor kan er niet gezegd worden welke combinaties van grids het juiste resultaat geven.

Wanneer de juiste combinaties zijn gevonden dan convergeert de multigrid methode naar de juiste oplossingen. Maar als we naar de error plots zien, dan zien we niet het gewenste gedrag. In al de beschouwde gevallen veranderd de error niet meer na een aantal V-cycles, terwijl we hoopten dat de error als maar kleiner en kleiner zou worden.

Doordat de implementatie altijd getest werd op model problemen die zowel in het boek [1] als in de les werden aangehaald lijken deze problemen niet afkomstig van de implementatie maar zijn ze misschien afkomstig van het eigenwaarde probleem.

Referenties

[1] W. L. Briggs, V. E. Henson en S. F. McCormick, *A Multigrid Tutorial*, 2nd edition, 2000.

[2] E. L. Briggs, J. Bernholc, *Multigrid methods in Electronic Structure Calculations*, `https://www.researchgate.net/publication/2143586_Multigrid_Methods_in_Electronic_Structure_Calculations`.