Inhoudsopgave

Hoofdstuk 1

Inleiding

1.1 Doel van de opdracht

De oliemaatschappij SHELL bevat een aantal afdelingen die ter ondersteuning dienen van de winning en exploitatie van olie en gas. Een van deze onderdelen is de geologische afdeling die zich bezighoudt met het modelleren van geologische processen. Aan de hand van de geologische geschiedenis probeert men de kans op de aanwezigheid van winbare hoeveelheden gas en olie te voorspellen. Daarnaast is men sterk geinteresseerd in de drukken van de in de bodem aanwezige vloeistoffen en gassen. Een belangrijk aspect van de druk is dat hij invloed heeft op de akoestische eigenschappen van de materie. Kennis van de overdruk is dus nodig om seïsmische gegevens beter te kunnen interpreteren. Daarnaast kan de heersende druk het wegsijpelen van olie door de bodemlagen heen veroorzaken. Het gevolg is dat er feitelijk geen olie meer te winnen is uit een gebied waar men olie had verwacht. Het doel van deze afstudeeropdracht is een mathematisch model te construeren teneinde de druk, als functie van de tijd en plaats, in de bodemlagen te kunnen voorspellen.

1.2 Achtergrond van het probleem

Een mogelijkheid om meer kennis over de heersende drukken te vergaren is het voorspellen van de druk als functie van tijd en plaats, door middel van een mathematisch model. Een eenvoudig doch veel gebruikt model is dat van de diffusievergelijking, gebaseerd op massabehoud en de wet van Darcy [1]. Deze instationaire vergelijking moet in de driedimensionale ruimte worden opgelost in geologische tijdschalen, dat wil zeggen over miljoenen jaren. Vanwege het driedimensionale karakter zijn de modellen altijd erg groot en moet men worstelen met de aanwezige computer resources.

Het oplossen van de diffusie vergelijking door middel van een discretiseringsmethode leidt tot zeer grote stelsels lineaire vergelijkingen welke slechts door middel van iteratieve technieken kunnen worden opgelost. Een groot probleem dat zich hierbij voordoet, is dat de bodem veelal gelaagd is. De doorlatendheid van deze lagen kan onderling zeer sterk verschillen. Waarbij relatieve factoren van de orde 10^8 geen uitzondering zijn. De figuur op bladzijde 5 geeft een schematische schets van zo'n bodem. Tabel 1.1 geeft een indicatie van de orde van grootte van permeabiliteit en porositeit voor verschillende soorten grond ([1] pagina's 12 en 68).

soort ondergrond	porositeit (in %)	permeabiliteit (in Darcy)
zand(sand)	37-50	$0.2 \cdot 10^2 - 1.8 \cdot 10^2$
zandsteen(sandstone)	4-50	$5 \cdot 10^{-4} - 30 \cdot 10^{-1}$
baksteen	12-34	$4.8 \cdot 10^{-3} - 2 \cdot 10^{-1}$
lei(shale)	0-80	$10^{-11} - 10^{-6}$
kalksteen(limestone)	4-10	$2 \cdot 10^{-11} - 5 \cdot 10^{-10}$
conglomerate	10-50	$10^{-5} - 0.1$

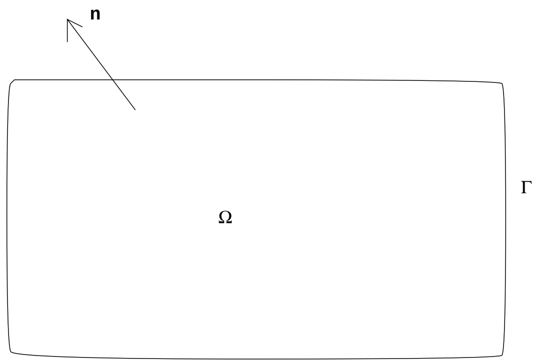
Tabel 1.1: Permeabiliteit en porositeit in verschillende lagen.

De eenheid van de permeabiliteit is Darcy, waarbij 1 Darcy= $9.8710^{-9}\mathrm{cm}^2$.
Deze grote verschillen zorgen voor een slechte conditie van het op te lossen stelsel vergelijkingen. Als gevolg daarvan zullen iteratieve methoden niet of zeer slecht convergeren. Men moet dus maatregelen nemen om deze problematiek op een efficiënte manier op te lossen. Eén van de mogelijke remedies is het zoeken naar goede preconditioners, die rekening houden met de gelaagdheid, zonder dat de opslagkosten extreem toenemen. In dit verslag zullen we een onderzoek doen naar zulke preconditioners.

1.3 Het wiskundig model

De diffusievergelijking waarvan we gebruik zullen maken om de druk te bepalen, leiden we af uit de wet van massabehoud en de wet van Darcy. De porositeit ϕ definieert men als het volume ingenomen door de vloeistof ten opzichte van het totale volume van het gebied (figuur 1.1), dat uit vaste (s) en vloeibare (f) stoffen bestaat. In formule notatie is dit gelijk aan:

$$\phi = \frac{V_f}{V_{tot}},$$



Figuur 1.1: Het poreuze medium Ω

waarin V_f en V_{tot} respectievelijk het vloeistofvolume en het totale volume zijn. het volume V_s dat door de vaste stof wordt ingenomen is: $V_{tot} - V_f$.

Wet van massabehoud

Behoud van massa impliceert, dat de verandering van de massa in het gebied gelijk is aan de netto massaflux door zijn oppervlak A, als we er tenminste vanuit gaan dat er geen bronnen in Ω aanwezig zijn ([2] hoofdstuk 2). De vergelijking van de massabehoudswet luidt als volgt

$$\frac{d}{dt}(\phi M) = \frac{d}{dt}\int_{\Omega}(\phi\rho)dV = \int_{\Omega}\frac{\partial(\phi\rho)}{\partial t}dV = -\int_{\Gamma}(\phi\rho\boldsymbol{u}).\boldsymbol{n}dA. \tag{1.1}$$

Met
M : massa van de vloeistof (in kg),

ϕ : de poreusheid van het medium (afhankelijk van de plaats x en de tijd t),

ρ : dichtheid van de vloeistof (in kgm^{-3} afhankelijk van x en t),

$\boldsymbol{u}$: de snelheid van de vloeistof (in ms^{-1}).

n is de normale vektor die naar buiten gericht positief wordt genomen.

Met behulp van stelling van Gauss wordt vergelijking 1.1 verder ontwikkeld tot

$$\frac{\partial(\phi\rho)}{\partial t} + \nabla\cdot(\phi\rho\boldsymbol{u}) = 0. \tag{1.2}$$

Voor de overblijvende vaste stof s in het medium met volume fractie ϕ_s ($\phi_s = 1 - \phi$) [3], geldt ook de massabehoudwet:

$$\frac{\partial(\phi_s\rho_s)}{\partial t} + \nabla\cdot(\phi_s\rho_s\boldsymbol{u_s}) = 0, \tag{1.3}$$

waarbij ρ_s en $\mathbf{u_s}$ respectievelijk de dichtheid en de snelheid van rots s zijn.

Hierbij wordt dus aangenomen dat ook de rots in de loop van de tijd van positie verandert. Nemen we verder aan dat de dichtheid ρ van een rotsdeeltje niet met de tijd verandert, dan geldt

$$\frac{D\rho_s}{Dt} = \frac{\partial \rho_s}{\partial t} + \boldsymbol{u_s}.\nabla\rho_s = 0. \tag{1.4}$$

Vergelijkingen 1.1, 1.2, 1.3 en 1.4 samen geven na enige rekenwerk (voor meer details zie [4]) de volgende vergelijking:

$$\nabla \cdot \boldsymbol{q} = -\frac{1}{1-\phi}\frac{D\phi}{Dt}. \tag{1.5}$$

$\boldsymbol{q}$ is de relatieve grondwatersnelheid [5], gegeven door: $\boldsymbol{q} = \phi(\boldsymbol{u} - \boldsymbol{u}_s)$.

De wet van Darcy

De vloeistof in het gebied Ω ondervindt twee soorten krachten:
de volumekracht $\boldsymbol{F}_{vol}{=}\int_\Omega(\rho\boldsymbol{g})dV$,
met g wordt hier de zwaartekrachtversnelling bedoeld en
de oppervlaktekracht $\boldsymbol{F}_{op}{=}-\int_\Gamma(p_f\boldsymbol{n})dA$,
waarbij p_f de vloeistofdruk is. Deze twee krachten tezamen vormen de totale, op het fluidum per volume eenheid werkende, kracht $\boldsymbol{F}$ $(=\rho\boldsymbol{g}-\nabla p_f)$. Darcy heeft aan de hand van een aantal experimenten, met stromend water in zand, geconstateerd dat de verandering van de vloeistofsnelheid evenredig is met de verandering van de druk:

$$\boldsymbol{q} = \tfrac{k}{\mu}\boldsymbol{F}.$$

In deze vergelijking is k de permeabiliteit van het medium (afhankelijk van de ϕ) en μ is de viscositeit van het fluidum. In onze problemen mag μ wel als een constante gekozen worden, omdat de invloed van de permeabiliteit van het medium veel belangrijker is dan de invloed van de viscositeit. Voor de afleiding van de diffusievergelijking (1.6), verwijzen we nogmaals naar [4], hoofdstukken 2 en3.

$$\frac{1}{\mu}\nabla \cdot (k\nabla p) = (\frac{\alpha}{1-\phi})(\frac{\partial\sigma}{\partial t} - \frac{\partial p}{\partial t}). \tag{1.6}$$

De parameters α en σ moeten nog bepaald worden. Voor de oplossing van dit probleem zijn begin-en randvoorwaarden nodig. Gebaseerd op de fysica, kiezen we als randvoorwaarden het volgende:

- de bovenste rand is een Dirichlet-rand: $p = 1$ en
- de overige randen zijn Neumann-randen: $\frac{\partial p}{\partial n} = 0$.

Voor de beginvoorwaarde kiezen $p(0) = c$, waarbij c een konstante is.
Vergelijking (1.6) is niet lineair want zowel de permeabiliteit k als de porositeit ϕ zijn beiden functies van de druk p. Dus (1.6) is in het algemeen analytisch niet oplosbaar. Teneinde de oplossing p te bepalen, moeten we dus een numeriek methode gebruiken. Voor de ruimtediscretisatie zullen we gebruik maken van de eindige elementen methode

(e.e.m). Voor de tijdsintegratie is hier voor de Euler impliciet methode gekozen. Als elementen worden driehoekjes (tetraëders in $\mathbb{R}^3$) gekozen. Eerst wordt de vergelijking lineair gemaakt met behulp van Picard-iteratie:
stel dat p^t de druk op tijdstip t is, dan wordt (1.6) lineair door voor het tijdstip $t + \Delta t$, de coefficiënten k en ϕ op het niveau t te nemen. Voor de volledige details over de discretisatie, zie [4] hoofdstuk 4.

Het resulterende stelsel vergelijkingen

Het stelsel vergelijkingen dat we uiteindelijk krijgen is van de vorm:

$$\boldsymbol{A}^t \boldsymbol{p}^{t+\Delta t} = \boldsymbol{b}^t. \tag{1.7}$$

Dit stelsel moet voor iedere tijdstap t opgelost worden in 3D. Door het lokale karakter van de eindige elementen methode is de matrix $\boldsymbol{A}$ een zeer grote ijle-matrix. Dus om het stelsel (1.7) op te lossen, ligt het voor de hand om iteratieve technieken toe te passen.

1.4 Iteratieve methoden

Voor een simpele notatie vervangen we het stelsel (1.7) door

$$\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}. \tag{1.8}$$

Voor de oplossing van dit stelsel zijn er twee aanpakken mogelijk:
- directe methoden: Gauss-eliminatie en Cholesky decompositie of
- iteratieve methoden: Jacobi, Gauss-Seidel, successieve overrelaxatie, geconjugeerde gradiënten (CG) methode, enz

Daar $\boldsymbol{A}$ zeer ijl is, en in het 3D geval erg groot is een directe methode inefficient. De reden hiervoor is dat door fill in de matrices $\boldsymbol{L}$ en $\boldsymbol{U}$ waarvoor geldt

$$\boldsymbol{A} = \boldsymbol{L}\boldsymbol{U},$$

veel minder ijl zijn dan $\boldsymbol{A}$. In het 3D geval leidt dit tot een zeer groot geheugengebruik. Daarom kiezen we voor een iteratieve oplosmethode. Omdat het probleem tijdsafhankelijk is, is er vaak een goede start oplossing ($\boldsymbol{x}_0$) beschikbaar. De matrix $\boldsymbol{A}$ is symmetrisch, daarom ligt het voor de hand om de CG-methode toe te gaan passen. Deze CG-methode behoort tot de klasse van Krylov deelruimte methoden. De Krylov deelruimte wordt gegeven door:

$$K_i(\boldsymbol{A}, \boldsymbol{r}_0) = \text{span}\{\boldsymbol{r}_0, \boldsymbol{A}\boldsymbol{r}_0, ..., \boldsymbol{A}^{i-1}\boldsymbol{r}_0\},$$

hierbij geldt:
i is de dimensie van de ruimte $K_i(\boldsymbol{A}, \boldsymbol{r}_0)$ en $\boldsymbol{r}_0$ is het residu gedefinieerd door:

$$\boldsymbol{r}_0 = \boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}_0.$$

Er geldt dat iedere iteraat $\boldsymbol{x}_i$ in die ruimte zit, dat wil zeggen

$\boldsymbol{x}_i \in \boldsymbol{x}_0 + \ K_i(\boldsymbol{A}, \boldsymbol{r}_0)$.

De methode probeert de afstand tussen de echte oplossing $\boldsymbol{x}$ en de benadering $\boldsymbol{x}_i$ gemeten in de $\boldsymbol{A}$-norm, zo klein mogelijk te maken. het CG-algoritme ziet als volgt uit:

k = 0; $\boldsymbol{x}_0 = 0$; $\boldsymbol{r}_0 = \boldsymbol{b}$;
while $\boldsymbol{r}_k \neq 0$ **do**
  k = k + 1;
  if k = 1 **then**
    $\boldsymbol{p}_1 = \boldsymbol{r}_0$
  else
    $\beta_k = \frac{(\boldsymbol{r}_{k-1}, \boldsymbol{r}_{k-1})}{(\boldsymbol{r}_{k-2}, \boldsymbol{r}_{k-2})}$
    $\boldsymbol{p}_k = \boldsymbol{r}_{k-1} + \beta_k \boldsymbol{p}_{k-1}$
  end if
  $\alpha_k = \frac{(\boldsymbol{r}_{k-1}, \boldsymbol{r}_{k-1})}{(\boldsymbol{p}_k, \boldsymbol{A}\boldsymbol{p}_k)}$
  $\boldsymbol{x}_k = \boldsymbol{x}_{k-1} + \alpha_k \boldsymbol{p}_k$
  $\boldsymbol{r}_k = \boldsymbol{r}_{k-1} - \alpha_k \boldsymbol{A}\boldsymbol{p}_k$
end while

Hierbij is k de iteratie-index van het algoritme en $\boldsymbol{p}_k$ is de zoekrichting.

Hier dient opgemerkt te worden dat in plaats van het stopkriterium (while $\boldsymbol{r}_0 \neq 0$), men een ander stopkriterium kan gebruiken.

Bij elke iteratieve methode speelt de convergentie van het proces een hoofdrol. Bekend is dat de convergentie van de CG-methode bepaald wordt door de eigenwaardenverdeling van de matrix $\boldsymbol{A}$, zie bijvoorbeeld [6]. Het conditiegetal $K_2(\boldsymbol{A})$ van $\boldsymbol{A}$ is gegeven door:

$$K_2(\boldsymbol{A}) = \frac{\lambda_n}{\lambda_1},$$

waarbij λ_n en λ_1 de grootste resp. de kleinste eigenwaarde van $\boldsymbol{A}$ zijn. In het algemeen geldt: hoe kleiner K_2 is hoe beter is de convergentie van CG.

Door de grote verschillen in de coefficiënten van $\boldsymbol{A}$ onderling, verwachten we dat het CG-proces toegepast op 1.8 slecht zal convergeren. Deze grote verschillen veroorzaken een groot conditiegetal van de matrix $\boldsymbol{A}$ en dus een zeer trage convergentie. Om dit probleem aan te pakken gaan we gebruik maken van preconditioners.

1.5 Doel van een preconditionering

We gebruiken een preconditionering om de conditie van de matrix $\boldsymbol{A}$ te verbeteren, d.w.z. het conditiegetal van $\boldsymbol{A}$ kleinerte maken. Hierdoor wordt de convergentie van het iteratie-proces sneller. Met andere woorden we moeten een preconditioneringsmatrix $\boldsymbol{P}$ gebruiken zodanig dat:

$$K_2(\boldsymbol{P}^{-1}\boldsymbol{A}) < K_2(\boldsymbol{A}).$$

Een veel gebruikte preconditionering is die van de incomplete LU decompositie [7]. Een zeer speciale preconditionering die we aanvankelijk wilden gebruiken is gebaseerd op een benaderende oplossingstechniek die door Kraaijevanger werd gehanteerd [8]. We geven hierna in het kort weer wat deze methode inhoudt.

1.5.1 Beschrijving van de Kraaijevanger-methode

Kraaijevanger heeft het drukprobleem in het verleden al opgelost door een groot aantal simplificaties aan te brengen. Om tot een vereenvoudigd probleem te komen heeft Kraaijevanger een tweetal aannames gedaan:

1. De overdruk (druk) in een hoog permeabele laag (bijvoorbeeld een zandlaag) is konstant ($p = p_k$).

2. In laag permeabele lagen (bijvoorbeeld lei) is de vloeistofstroming vertikaal.

We beschouwen nu het permeabiliteitsmodel dat Kraaijevanger behandeld heeft:
- in een zandlaag geldt dat de permeabiliteit k oneindig hoog is en dat de druk slechts van de tijd t afhankelijk is. De druk is onafhankelijk van x, y en z: $\frac{Dp}{Dt} = \frac{dp}{dt}$. Dit betekent dat de wet van Darcy ($\boldsymbol{q} = \frac{k}{\mu}(-\nabla p)$) niet van toepassing is.

- in een leilaag heeft Kraaijevanger geëist dat de vloeistof slechts in de vertikale richting kan bewegen.

De twee aannames, die Kraaijevanger heeft beschouwd, stelden hem in staat om het gebied Ω in een soort kolommen te verdelen. Het probleem is nog extra vereenvoudigd door te veronderstellen dat de lagen per kolom horizontaal lopen. Vervolgens wordt aangenomen dat in iedere kolom alle fysische componenten konstant in x en y zijn, voor gegeven diepte z en tijd t. Met het mathematische model dat hij heeft beschouwd is een stelsel vergelijkingen gekregen van de vorm:

$$\boldsymbol{Bp} = \boldsymbol{c}.$$

De bedoeling was dat we de verkregen matrix $\boldsymbol{B}$, als preconditionering voor onze oorspronkelijk matrix $\boldsymbol{A}$ in (1.8) gaan gebruiken. Een uitgebreide afleiding van de Kraaijevanger-methode staat ook in het scriptieverslag [4].

1.5.2 De gepreconditioneerde CG methode

Omdat we nu een aantal preconditioneringen zullen testen, gaan wij met name het gepreconditioneerde CG-algoritme gebruiken. En dat ziet er als volgt uit [9]:

k = 0; $\boldsymbol{x}_0 = 0$; $\boldsymbol{r}_0 = \boldsymbol{b}$;
while $\boldsymbol{r}_k \neq 0$ **do**
  los op: $\boldsymbol{P}\boldsymbol{z}_k = \boldsymbol{r}_k$
  k = k+1;
  if k = 1 **then**
    $\boldsymbol{p}_1 = \boldsymbol{z}_0$
  else
    $\beta_k = \frac{(\boldsymbol{r}_{k-1}, \boldsymbol{z}_{k-1})}{(\boldsymbol{r}_{k-2}, \boldsymbol{z}_{k-2})}$
    $\boldsymbol{p}_k = \boldsymbol{z}_{k-1} + \beta_k \boldsymbol{p}_{k-1}$
  end if
  $\alpha_k = \frac{(\boldsymbol{r}_{k-1}, \boldsymbol{z}_{k-1})}{(\boldsymbol{p}_k, \boldsymbol{A}\boldsymbol{p}_k)}$
  $\boldsymbol{x}_k = \boldsymbol{x}_{k-1} + \alpha_k \boldsymbol{p}_k$
  $\boldsymbol{r}_k = \boldsymbol{r}_{k-1} - \alpha_k \boldsymbol{A}\boldsymbol{p}_k$.
end while

k is de iteratie-index van het algoritme, $\boldsymbol{r}$ het residu, $\boldsymbol{p}_k$ de zoekrichting en $\boldsymbol{P}$ de preconditioneringsmatrix die symmetrisch en positief definiet moet zijn. Per iteratie is een matrix-vektor produkt vereist. Merk op dat het gebruikte stopcriterium aanzienlijk verbeterd kan worden.
We zullen nu een aantal eigenschappen van deze methode bespreken. De eerste beshrijving van het CG-algoritme dateert van 1952 [10].
De vektoren die in het CG-algoritme voorkomen hebben de volgende eigenschappen:

1. de zoekrichtingsvektoren $\boldsymbol{p}_i$ voldoen aan

$$span\{\boldsymbol{p}_1, ..., \boldsymbol{p}_k\} = span\{\boldsymbol{r}_0, ..., \boldsymbol{r}_{k-1}\} = K^k(\boldsymbol{A}; \boldsymbol{r}_0),$$

dit betekent dat de zogenaamde Krylov-ruimte $K^i(\boldsymbol{A}; \boldsymbol{r}_0)$ opgespannen wordt door de vektoren $\boldsymbol{r}_i$. Iedere iteraat $\boldsymbol{x}_k$ ligt dus in deze ruimte en CG probeert de afstand tussen deze $\boldsymbol{x}_k$'s en de echte oplossing $\boldsymbol{x}$ zo klein mogelijk te maken, dit wil zeggen

$$\min_{\boldsymbol{y} \in K^i(\boldsymbol{A}; \boldsymbol{r}_0)} \left\| \boldsymbol{y} - \boldsymbol{x} \right\|.$$

Omdat we de echte oplossing niet kennen maar wel $\boldsymbol{A}\boldsymbol{x}$, ligt het voor de hand om in de zogenaamde $\boldsymbol{A}$-norm te minimaliseren. De $\boldsymbol{A}$-norm is gedefiniëerd als

$$\left\| \boldsymbol{v} \right\|_{\boldsymbol{A}} = (\boldsymbol{v}, \boldsymbol{A}\boldsymbol{v})^{\frac{1}{2}}$$

We minimaliseren dan: $\left\| \boldsymbol{x} - \boldsymbol{x}_i \right\|_{\boldsymbol{A}} = \min_{\boldsymbol{y} \in K^i(\boldsymbol{A}; \boldsymbol{r}_0)} \left\| \boldsymbol{y} - \boldsymbol{x} \right\|_{\boldsymbol{A}}$.

2. de residuen $\boldsymbol{r}_i$ zijn orthogonaal: $\boldsymbol{r}_i^T \boldsymbol{r}_j = 0, j \leq i - 1$,

3. $\boldsymbol{r}_i^T \boldsymbol{p}_j = 0$ voor $j = 1, ..., i$ en

4. $\boldsymbol{p}_i^T \boldsymbol{A}\boldsymbol{p}_j = 0 j = 1, ..., i - 1$.

In theorie is de CG een eindige methode, dit betekent dat na n iteraties de Krylov deelruimte is gelijk aan $\mathbb{R}^n$. In dit geval is $\boldsymbol{x}_i = \boldsymbol{x}$, want

$\left\|\boldsymbol{y} - \boldsymbol{x}\right\|_{\boldsymbol{A}}$ is over $K^n(\boldsymbol{A}; \boldsymbol{r}_0) = \mathbb{R}^n$ geminimaliseerd.
Desondanks wordt dit in de prakrijk niet gebruikt, wegens het nadeel dat men de convergentie na n iteraties bereikt, waarbij n een onpraktisch grote waarde kan aannemen. Daarnaast is het ook mogelijk dat de resultaten, zodanig verstoord kunnen worden door afrondfouten, dat de eigenschappen (1,2,3 en 4) van CG niet meer voor de berekende vektoren gelden.

Voor het bewijs van al deze eigenschappen zie [9]. Merk op dat de vektoren $\boldsymbol{r}_i$ een orthogonale basis vormen voor de Krylov-ruimte. De $\boldsymbol{A}$-norm van de fout is monotoon dalend

$$\left\|\boldsymbol{x} - \boldsymbol{x}_{i+1}\right\|_{\boldsymbol{A}} \leq \left\|\boldsymbol{x} - \boldsymbol{x}_i\right\|_{\boldsymbol{A}}.$$

Omdat in de praktijk de norm $\left\|\boldsymbol{x} - \boldsymbol{x}_i\right\|_{\boldsymbol{A}}$ onmogelijk is uit te rekenen, kijkt men naar het residu

$$\left\|\boldsymbol{r}_i\right\|_2 = \left\|\boldsymbol{x} - \boldsymbol{x}_i\right\|_{\boldsymbol{A}^T\boldsymbol{A}}.$$

Merk op dat deze rij $\left\|\boldsymbol{r}_i\right\|_2$ niet monotoon dalend hoeft te zijn. De ongelijkheid

$$\left\|\boldsymbol{r}_i\right\|_2 = \left\|\boldsymbol{A}\boldsymbol{x}_i - \boldsymbol{b}\right\|_2 \leq \left\|\boldsymbol{A}\right\|_2 \left\|\boldsymbol{x} - \boldsymbol{x}_i\right\|_{\boldsymbol{A}},$$

zorgt ervoor dat de norm van het residu na enige iteraties weer dalend wordt.

Hoofdstuk 2

Testproblemen en standaard preconditioneringen

Inleiding

Het stelsel vergelijkingen dat we gaan oplossen is zeer groot. Omdat onze matrix $\boldsymbol{A}$ symmetrisch is wordt hier CG als de iteratieve methode gebruikt, in combinatie met verschillende preconditioneringen om de convergentie te versnellen. De bedoeling is dat we nu een paar van de veel gebruikte preconditioneringen op ons probleem gaan uittesten en de hieruit volgende resultaten met elkaar vergelijken. De vergelijkingen die wij zullen doen, worden gebaseerd op de convergentiesnelheid van het CG-proces of het aantal benodigde iteraties om convergentie te bereiken. Daarnaast zal ook de conditie van de gepreconditioneerde matrix ter sprake komen.

2.1 Beschrijving van de testproblemen

In onze testproblemen beschouwen we verschillende strukturen van het gebied Ω:

- een gebied bestaand uit verschillende lagen (afwisselend zand- en leilagen). Het aantal lagen is achtereenvolgens: 3,5 en 11,

- een gebied waarbij een zandlaag wordt ingesloten door lei,

- een gebied waarbij een leilaag ingesloten wordt door zand.

De druk in de verschillende gebieden, wordt door de volgende diffusievergelijking beschreven:

$$\frac{\partial p}{\partial t} + \nabla \cdot (k \nabla p) = 0, \qquad (2.1)$$

waarin k een parameter is, die de permeabiliteit van het medium aangeeft. In ons probleem, kiezen we de parameter k als volgt: $k = 1$ in een zandlaag en $k = 10^{-7}$ in een leilaag.

We kiezen als randvoorwaarden voor het probleem:

- aan de bovenkant van Ω een Dirichlet randvoorwaarde,

- voor de resterende randen kiezen we een homogene Neumann randvoorwaarde.

In eerste instantie, beschouwen we het gebied uit Figuur 2.3 en we bekijken voorlopig het stationaire probleem. Voor de oplossing van dit probleem worden de volgende stappen genomen:

1) Verdeling van het gebied in eindige elementen (omdat we in de tweedimensionale ruimte werken, wordt voor driehoekjes als elementen gekozen),

2) discretisatie van de vergelijking aan de hand van de eindige elementen methode, wat resulteert in een groot stelsel vergelijkingen van de volgende vorm:

$$\boldsymbol{A}\boldsymbol{p} = \boldsymbol{b}, \tag{2.2}$$

3) gebruik van iteratieve methoden om $\mathbf{p}$ op te lossen uit (2.2).

Omdat $\boldsymbol{A}$ symmetrisch en positieve definiet is, zullen we CG als iteratieve methode gebruiken. Bij iteratieve methoden speelt de convergentie een essentiële rol. Daarom zullen wij trachten de convergentie van deze methode, met behulp van een aantal preconditioneringen, te versnellen.
De preconditioneringen die we zullen gebruiken zijn onder andere:

$*$ diagonaal preconditionering,
$*$ ILU preconditionering,
$*$ Eisenstat preconditionering en
$*$ mod_Eisenstat preconditionering.

Daarnaast bekijken we de CG methode zonder preconditionering.

Bij iedere iteratieve methode hoort een start oplossing (start-vektor). In ons geval, maken we gebruik van twee soorten start-vektoren namelijk: de nul vektor en een random vektor. Met random wordt hier een vektor bedoeld, waarvan de componenten in het interval (-1,1) liggen en iedere component door een random generator wordt bepaald.
De reden voor deze keuze is dat we willen uitsluiten dat door toeval een bepaalde belangrijke eigenvektor geen componenten in de start vektor heeft.

2.2 Een eerste experiment

Allereerst hebben we het probleem met één leilaag opgelost. Hierbij deed zich een vreemd verschijnsel voor. Indien de nauwkeurigheid van de solver met ILU preconditionering op een gebruikelijke waarde van $O(10^{-3})$ werd gezet, waren er weinig iteraties nodig. Echter de verkregen resultaten waren slechts tot op zekere hoogte acceptabel. Dat wil zeggen het stopcriterium was relatief snel bereikt en het residu had ook aannemelijke waarden. Echter na verhoging van de nauwkeurigheid ontdekten we een opmerkelijk gedrag van het residu, namelijk dat in een bepaalde iteratie, het residu opeens aanzienlijk groter werd. Er was een duidelijke sprong waarneembaar. Een mogelijk verklaring voor dit verschijnsel is dat, de bijdrage van de bij de kleinste eigenwaarde behorende eigenvector in de oplossing in eerste instantie erg klein is. Pas na verloop van een groot aantal iteraties wordt deze bijdrage zichtbaar. In feite is CG gebaseerd op het Lanczos proces dat kleine eigenwaarden pas na verloop van tijd "ontdekt". Zolang de kleinste eigenwaarde niet zichtbaar is, verloopt het iteratie proces redelijk snel, de kleinste eigenwaarde vertraagt het proces aanzienlijk en krikt kennelijk het residu aanmerkelijk op. Dit gedrag wordt nader geanalyseerd in de volgende paragraaf.

2.3 Analyse van de resultaten

Voordat we met de analyse beginnen, zullen we eerst iets vertellen over de gebruikte standaard stopcriterium. Beschouw het stelsel

$$\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}. \tag{2.3}$$

We willen het CG proces stoppen als:

$$\boldsymbol{e}_k = \frac{\left\|\boldsymbol{x}-\boldsymbol{x}_k\right\|}{\left\|\boldsymbol{x}\right\|} \leq \varepsilon,$$

met $\boldsymbol{e}_k$ de relatieve fout en ε de nauwkeurigheid. Stel dat λ_1 de kleinste eigenwaarde van de matrix $\boldsymbol{A}$ is en dat $\tilde{\lambda_1}$ zijn approximatie (bekend onder de naam Ritzwaarde) is. Omdat $\boldsymbol{x}$ niet bekend is, gebruiken we het volgende stopcriterium:

stop CG als

$$\frac{\left\|\boldsymbol{r}_k\right\|}{\left\|\boldsymbol{x}_k\right\|} = \frac{\left\|\boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}_k\right\|}{\left\|\boldsymbol{x}_k\right\|} \leq \tilde{\lambda_1}\varepsilon. \tag{2.4}$$

Stel verder dat $\tilde{\lambda_1} \simeq \lambda_1$ en $\boldsymbol{x}_k \simeq \boldsymbol{x}$, dan geldt

$$
\begin{aligned}
\frac{\left\|\boldsymbol{x}-\boldsymbol{x}_k\right\|}{\left\|\boldsymbol{x}\right\|} &\simeq \frac{\left\|\boldsymbol{A}^{-1}(\boldsymbol{b}-\boldsymbol{A}\boldsymbol{x}_k)\right\|}{\left\|\boldsymbol{x}_k\right\|} \\
&\leq \frac{\left\|\boldsymbol{A}^{-1}\right\|\left\|\boldsymbol{r}_k\right\|}{\left\|\boldsymbol{x}_k\right\|} \\
&\simeq \frac{\left\|\boldsymbol{r}_k\right\|}{\tilde{\lambda}_1\left\|\boldsymbol{x}_k\right\|} \leq \varepsilon \qquad (\text{omdat}\left\|\boldsymbol{A}^{-1}\right\| \simeq \frac{1}{\tilde{\lambda}_1}). \qquad (2.5)
\end{aligned}
$$

Hiermee laten we zien dat de gewenste nauwkeurigheid ε gehaald is, indien $\tilde{\lambda}_1$ een goede benadering is van de kleinste eigenwaarde λ_1 van $\boldsymbol{A}$.

Een mogelijke verklaring van de grote sprong van het residu is het volgende:

neem aan dat: $\boldsymbol{x}-\boldsymbol{x}_0 = \sum_{j=1}^{n} c_j \boldsymbol{v}_j$,
met $\boldsymbol{v}_j$ de eigenvektoren van de matrix met lengte 1 en

$$\boldsymbol{A}\boldsymbol{v}_j = \lambda_j \boldsymbol{v}_j,$$

met λ_j de eigenwaarden van $\boldsymbol{A}$ geordend volgens $\lambda_1 < \lambda_2 < ... < \lambda_n$. We weten bovendien dat CG de iteranden $\boldsymbol{x}_i$ bepaalt zodanig dat

$\left\|\boldsymbol{x}-\boldsymbol{x}_i\right\|_{\boldsymbol{A}}$ minimaal is. Hierbij is $\left\|\boldsymbol{x}\right\|_{\boldsymbol{A}} = \sqrt{\boldsymbol{x}^T\boldsymbol{A}\boldsymbol{x}}$. Met andere woorden in de $\boldsymbol{A}$-norm moet de fout dalen.
De jump die wij hebben geconstateerd is vermoedelijk afkomstig van het feite dat de hierboven berekende $\boldsymbol{A}$-norm wel een optimale waarde bereikt terwijl dit voor de 2-norm van het residu

$$\left\|\boldsymbol{r}_i\right\|_2 = \left\|\boldsymbol{A}(\boldsymbol{x}-\boldsymbol{x}_i)\right\|_2$$

niet het geval is. Als bijvoorbeeld $\lambda_1 \simeq 10^{-7}\cdot\lambda_2$, dan zou dit kunnen betekenen dat $\boldsymbol{x}_i - \boldsymbol{x}_0$ een grote component in de richting van de eigenvektor $\boldsymbol{v}_1$ heeft, terwijl $\left\|\boldsymbol{A}(\boldsymbol{x}_i-\boldsymbol{x}_0)\right\|_2$ heel klein is.
We zullen nu proberen te begrijpen waarom de approximatie $\tilde{\lambda}_1$ niet in de buurt van de kleinste eigenwaarde λ_1 ligt en hoe het kan dat het residu een grote sprong vertoont. Daarvoor bekijken we het volgende voorbeeld:

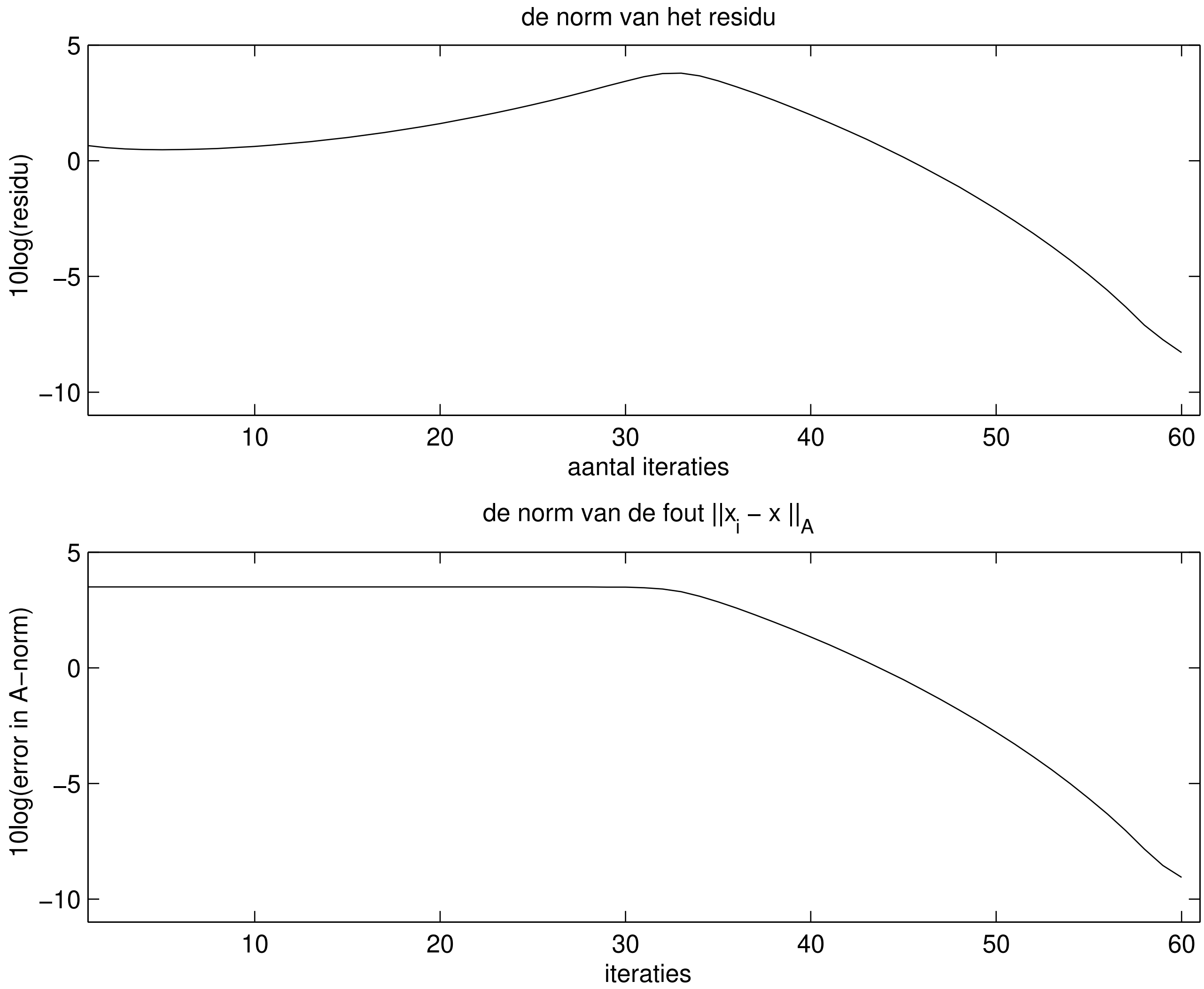
$$\boldsymbol{A} = \begin{pmatrix} 10^{-7} & & & & \\ & 1 & & \mathbf{0} & \\ & & 2 & & \\ & \mathbf{0} & & \ddots & \\ & & & & 60 \end{pmatrix},$$

de matrix $\boldsymbol{A}$ is 61x61 en $\boldsymbol{b} = (1...1)^T$.

Merk op dat dit voorbeeld slechts een illustratie is van de grote contrasten die bij (2.1) kunnen voorkomen. Met behulp van Matlab zijn de norm van het residu en de fout in de $\boldsymbol{A}$-norm berekend als functie van het aantal iteraties. In Figuur 2.1 zijn de resultaten geplot. Duidelijk is dat het residu eerst aanzienlijk stijgt alvorens te dalen, terwijl de $\boldsymbol{A}$-norm monotoon dalend is. Dit gedrag komt dus overeen met onze eerdere waarnemingen. Het feit dat $\tilde{\lambda}_1$ in eerste instantie niet in de buurt van λ_1 ligt kunnen we als volgt beredeneren. Stel dat een Ritzvektor een goede benadering is van de eigenvektor $\boldsymbol{v}_1$, dan impliceert dit dat de vektor $\boldsymbol{v}_1$ nagenoeg in de Krylov-ruimte $K^i(\boldsymbol{A}; \boldsymbol{r}_0)$ ligt. Het is bekend dat $\boldsymbol{x}_i - \boldsymbol{x}$ loodrecht staat op $K^i(\boldsymbol{A}; \boldsymbol{r}_0)$ in het $\boldsymbol{A}$ inprodukt. Dit betekent dat $\boldsymbol{x}_i - \boldsymbol{x}$ ongeveer $\boldsymbol{A}$ loodrecht staat op $\boldsymbol{v}_1$:

$$(\boldsymbol{x}_i - \boldsymbol{x}, \boldsymbol{v}_1)_{\boldsymbol{A}} = (\boldsymbol{x}_i - \boldsymbol{x}, \boldsymbol{A}\boldsymbol{v}_1) \cong 0.$$

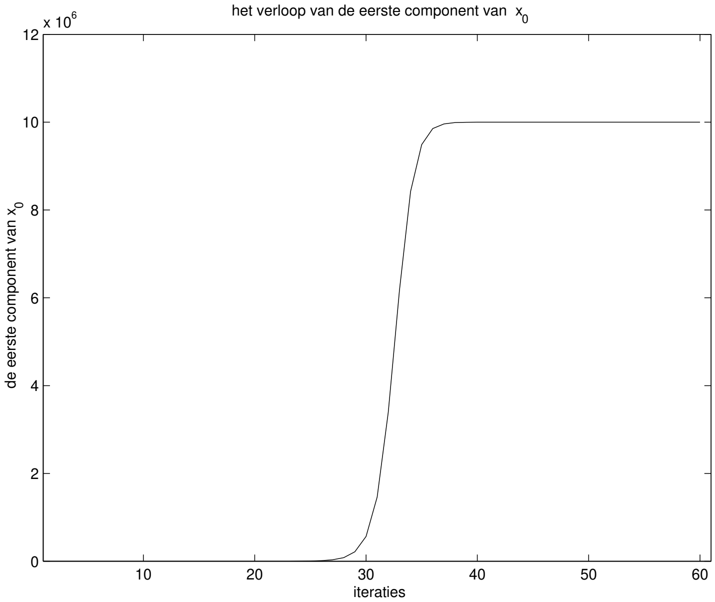
Omdat $\boldsymbol{A}\boldsymbol{v}_1 = \lambda_1 \boldsymbol{v}_1$, volgt hieruit dat $(\boldsymbol{x}_i, \boldsymbol{v}_1) \cong (\boldsymbol{x}, \boldsymbol{v}_1)$,
zodat de component van $\boldsymbol{x}_i$ in de richting van $\boldsymbol{v}_1$ overeenkomt met de component van $\boldsymbol{x}$ in de zelfde richting. Echter in ons voorbeeld zou dit betekenen dat de eerste component van $\boldsymbol{x}_k$ op de eerste component van $\boldsymbol{x}(= 10^7)$ zal lijken.



Figuur 2.1: Schets van het residu en de $\boldsymbol{A}$-norm van de fout

Hoe de eerste component van de vektor $\boldsymbol{x}_i$ verloopt als functie van de iteraties, zien we duidelijk in Figuur 2.2. We zien dus dat deze component tot ongeveer de 28^{ste} iteratie klein blijft, daarna neemt hij snel toe tot de "exacte" waarde.
Uit deze constatering volgt dat $\boldsymbol{v}_1$ niet in de ruimte $K^i(\boldsymbol{A}; \boldsymbol{r}_0)$ ligt met $i < 28$. Als $\boldsymbol{v}_1$ in $K^i(\boldsymbol{A}; \boldsymbol{r}_0)$ zou zitten, dan had de eerste component van $\boldsymbol{x}_i$ ongeveer gelijk aan 10^7 moeten zijn. We concluderen dat de kleinste Ritzwaarde, althans in het begin, inderdaad niet op de kleinste eigenwaarde λ_1 zal lijken.
Uit het voorgaande volgt dat het pas zinnig is om het stopcriterium (2.5) toe te passen indien λ_1 voldoende goed benaderd is. Op grond van de experimenten weten we dat $\lambda_1 < 10^{-7}$ is en dus hebben we het stopcriterium in de CG methode ad-hoc zodanig aangepast dat CG pas uitgeconvergeerd is indien:

- $\lambda_{min} < 10^{-7}$ en



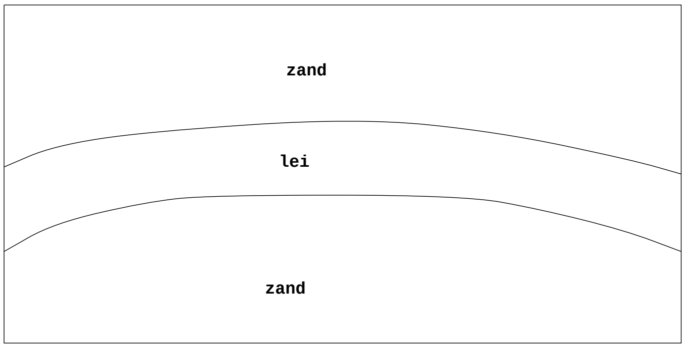
Figuur 2.2: Het verloop van de eerste component van $\boldsymbol{x}_i$

- $\left\|\boldsymbol{r}_k\right\| \leq \lambda_{min}\varepsilon$,

met ε de gewenste nauwkeurigheid. Merk op dat dit alleen werkt als λ_{min} daadwerkelijk kleiner dan 10^{-7} is. In de volgende paragraaf zullen we een aantal experimenten behandelen die ons inzicht geven over het gedrag van verschillende preconditioneringen.

2.4 Experimenten met verschillende preconditioneringen

Het gebied dat wij zullen bekijken wordt gegeven in Figuur 2.3. De lengte en de hoogte van Ω zijn zodanig geschaald dat de verhouding tussen beide 4 is. Vervolgens hebben we een



Figuur 2.3: Gebied Ω bestaand uit twee zandlagen en een leilaag

aantal experimenten gedaan voor het gebied uit Figuur 2.3. Hierbij is met name gekeken naar:

- het aantal uitgevoerde iteraties,
- de benadering van de kleinste eigenwaarde en
- het aantal iteraties nodig om een goede approximatie van de kleinste eigenwaarde te vinden (in de tabellen aangeduid met jump).

Het niet gepreconditioneerde probleem

Tabel 2.1 en Tabel 2.2 geven de resultaten met respectievelijk de nul- en random startvektor weer.

gridsize	approx. λ	aantal iter.	jump	2-norm res
15x9	1.63E-08	320	26	1.89E-09
30x9	8.42E-09	469	43	1.29E-09
30x18	5.36E-09	1495	81	9.46E-10

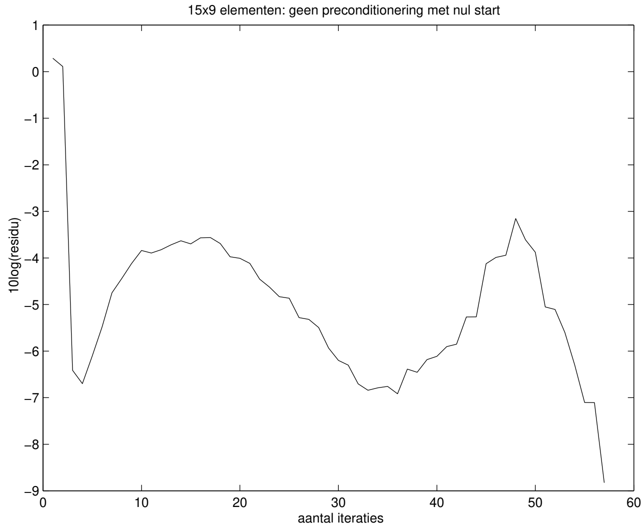
Tabel 2.1: Geen preconditionering met nul start vektor.

gridsize	approx. λ	aantal iter.	jump	2-norm res
15x9	1.63E-08	447	80	1.23E-09
30x9	8.44E-09	692	90	1.19E-09
30x18	7.56E-08	1578	160	1.43E-08

Tabel 2.2: Geen preconditionering met random start vektor.

In de eerste kolom van iedere tabel staat het aantal gebruikte gridpunten in de x- en de y-richting. In de tweede kolom staat de approximatie van λ, in de derde kolom het aantal uitgevoerde iteraties, in de vierde kolom de iteratie waarop de jump plaatsvindt en de laatste kolom bevat de 2-norm van het residu (vermenigvuldigd met de preconditioneringsmatrix).
Vergelijken we de resultaten, dan zien we dat de nulvektor betere resultaten geeft dan de random startvektor. De gridafhankelijk is voor beide gevallen gelijk. De kleinste eigenwaarde is in alle gevallen van de orde 10^{-8}, waarbij de gridafhankelijkheid zichtbaar is. Het aantal benodigde iteraties om de kleinste eigenwaarde te bepalen, blijkt ook gridafhankelijk te zijn. Hier dient ook opgemerkt te worden dat het aantal benodigde iteraties, zoals in de tabellen staat, veel groter is dan het aantal onbekenden. Dit verschijnsel kan optreden als de matrix slecht geconditioneerd is. In dat geval kan het dat, door doorwerking van afrondfouten (ruis), het CG proces na n iteraties (met n het aantal onbekenden), nog lang

niet klaar is. Dit is in tegenspraak met de theorie die uitgaat van exact rekenen. Een standaard voorbeeld waarbij dit verschijsel zich voordoet is dat van de buigende balk. Voor meer details zie [14]. In de hoofdstukken 3 en 4 hebben we een aantal experimenten uitgevoerd met Matlab voor rechte interfaces. Tot onze verwondering vertoonde Matlab deze eigenschap niet. Echter nadat we de start vektor hadden vervangen door een randomstart bleek dat gedrag zichtbaar. De resultaten van Matlab en Sepran waren nagenoeg identiek indien alle driehoekjes dezelfde diagonaal richting hadden gekregen. Figuur 2.4 laat het gedrag van het residu zien voor een nulstart, Figuur 2.5 voor een randomstart. Kennelijk gaat in het laatste geval de ruis een rol spelen. Opvallend zijn de "wiggles" in het residu bij een random start, hoewel een gemiddelde dalende tendens zichtbaar is.



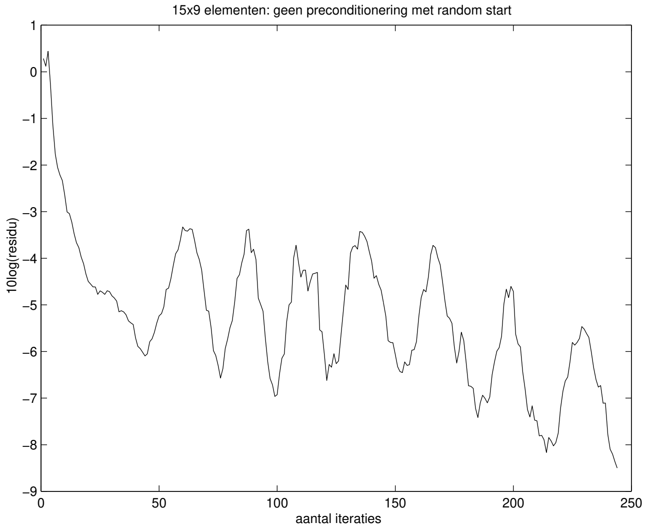
Figuur 2.4: Gedrag van het residu in het geval van nul startvektor

Diagonaal preconditionering

De resultaten van de diagonaal schaling worden gedemonstreerd in de tabellen 2.3 en 2.4.

gridsize	approx. λ	aantal iter.	jump	2-norm res
15x9	5.54E-09	55	35	7.91E-10
30x9	3.36E-09	68	39	5.53E-10
30x18	1.39E-09	118	78	5.57E-10

Tabel 2.3: Diagonaal preconditionering met nul start vektor.



Figuur 2.5: Gedrag van het residu in het geval van random startvektor

gridsize	approx. λ	aantal iter.	jump	2-norm res
15x9	5.54E-09	98	69	8.93E-10
30x9	3.38E-09	114	82	5.44E-10
30x18	1.38E-09	196	136	4.51E-10

Tabel 2.4: Diagonaal preconditionering met een random start vektor.

Ook hier zien we hetzelfde gedrag met betrekking tot random start en nul start. Opvallend is dat het aantal iteraties dat nodig is om de kleinste eigenwaarde te schatten van dezelfde orde van grootte is als zonder preconditionering. Niettemin is het aantal iteraties nodig om de convergentie te bereiken, in dit geval veel lager. Vergeleken met het aantal onbekenden is het echter nog steeds erg hoog.

Andere types preconditioneringen

De tabellen 2.5 en 2.6 geven de resultaten voor ILU preconditionering ([7]), tabellen 2.7 en 2.8 voor de zogenaamde Eisenstat implementatie ([13]), welke een aanzienlijke besparing aan geheugenruimte geeft en tenslotte de tabellen 2.9 en 2.10 de zogenaamde modified Eisenstat preconditionering, waarbij de matrix volgens het recept van Gustafson ([12]) is aangepast. De twee tabellen hieronder geven een overzicht van de verkregen resultaten met ILU als preconditionering.

gridsize	approx. λ	aantal iter.	jump	2-norm res
15x9	8.23E-08	20	13	4.57E-09
30x9	3.28E-08	29	19	2.76E-09
30x18	2.03E-08	35	24	1.42E-09

Tabel 2.5: ILU preconditionering met nul start vektor.

gridsize	approx. λ	aantal iter.	jump	2-norm res
15x9	8.23E-08	23	17	1.27E-08
30x9	3.28E-08	32	24	1.25E-09
30x18	2.03E-08	39	29	5.29E-09

Tabel 2.6: ILU preconditionering met random start vektor.

Het is duidelijk dat het aantal iteraties bij toepassing van deze preconditionering, veel minder is dan bij de eerder verkregen resultaten. Dit betekent dus dat ILU een geschiktere preconditionering is voor ons probleem dan de andere. Een bevestiging van deze conclusie zijn de resultaten in de tabellen 2.7 t/m 2.12, behorend respectievelijk bij Eisenstat en mod_Eisenstat met respectievelijk modificatie factoren $f = 0.4$ en $f = 0.975$. Het ad-hoc stopcriterium bleek voor $f = 0.975$ in eerste instantie niet goed te werken. De reden is dat $\tilde{\lambda}$ niet kleiner word dan 10^{-6}. Daarna hebben we dit aangepast tot

- $\lambda_{min} < 10^{-5}$ en
- $\left\| r_k \right\| \leq \lambda_{min} \cdot \varepsilon$.

gridsize	approx. λ	aantal iter.	jump	2-norm res
15x9	4.98E-08	25	14	2.22E-09
30x9	2.25E-08	34	18	1.13E-09
30x18	1.25E-08	44	24	2.83E-09

Tabel 2.7: Eisenstat met nul start vektor.

gridsize	approx. λ	aantal iter.	jump	2-norm res
15x9	4.99E-08	28	21	8.65E-09
30x9	2.24E-08	38	28	9.44E-10
30x18	1.26E-08	49	37	1.57E-09

Tabel 2.8: Eisenstat met random start vektor.

gridsize	approx. λ	aantal iter.	jump	2-norm res
15x9	7.61E-08	24	14	1.28E-08
30x9	3.39E-08	32	18	4.69E-09
30x18	1.92E-08	42	23	1.74E-09

Tabel 2.9: mod_Eisenstat met nul start vektor ($f = 0.4$).

gridsize	approx. λ	aantal iter.	jump	2-norm res
15x9	7.62E-08	29	21	4.31E-09
30x9	3.38E-08	36	26	1.54E-09
30x18	1.92E-08	46	35	4.94E-09

Tabel 2.10: mod_Eisenstat met random start vektor ($f = 0.4$).

gridsize	approx. λ	aantal iter.	jump	2-norm res
15x9	1.51E-06	24	14	2.03E-07
30x9	6.12E-07	29	16	1.01E-07
30x18	3.55E-07	38	21	4.63E-08

Tabel 2.11: mod_eisenstat met nul start vektor ($f = 0.975$).

gridsize	approx. λ	aantal iter.	jump	2-norm res
15x9	1.50E-06	32	22	3.26E-08
30x9	6.14E-07	35	25	5.58E-08
30x18	3.53E-07	47	32	5.92E-08

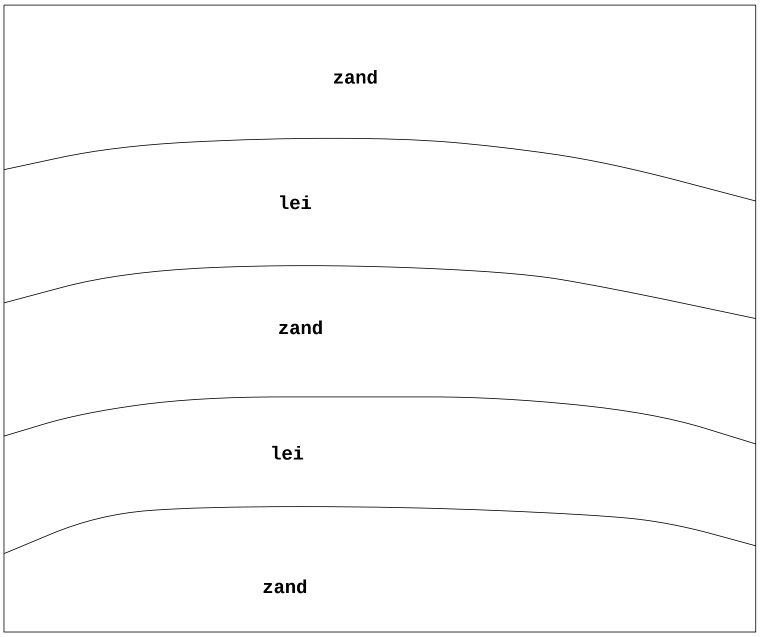
Tabel 2.12: mod_eisenstat met een random start ($f = 0.975$).

Van deze vier gebruikte preconditioneringen , is gebleken dat de ILU preconditionering de beste resultaten voor dit probleem oplevert. Het totaal benodigde iteraties in dat geval blijkt echter nog altijd vergelijkbaar te zijn met het aantal elementen in de horizontale richting. We zien dat de conclusies met betrekking tot de nul start en de random start hier onverminderd opgaan en zullen in het vervolg ons dus uitsluitend beperken tot de nul start. Verder laten de laatste tabellen een aanzienlijke verbetering ten opzichte van diagonaal schaling zien. ILU is iets beter dan Eisenstat (los van de opslagruimte). Modified Eisenstat is met $f = 0.975$ iets beter dan Eisenstat en vergelijkbaar met ILU. We gaan nu andere gebieden bekijken en daarbij zullen we alleen gebruik maken van ILU- en diagonaal-preconditioneringen.

2.5 Modificatie van het gebied Ω

2.5.1 Gebied bestaande uit 5 respectievelijk 11 lagen

Teneinde de invloed van het aantal lagen te onderzoeken, bekijken we allereerst het gebied van 5 lagen. Dit gebied is in Figuur 2.6 geschetst. De zandlaag staat aan de boven- en de onderkant van het gebied. We bekijken het totaal aantal uitgevoerde iteraties in het geval van ILU- en diagonaal-preconditionering.



Figuur 2.6: een gebied Ω met 5 lagen

De resultaten die bij een gebied met twee leilagen horen, zijn in de volgende tabellen samengevat.

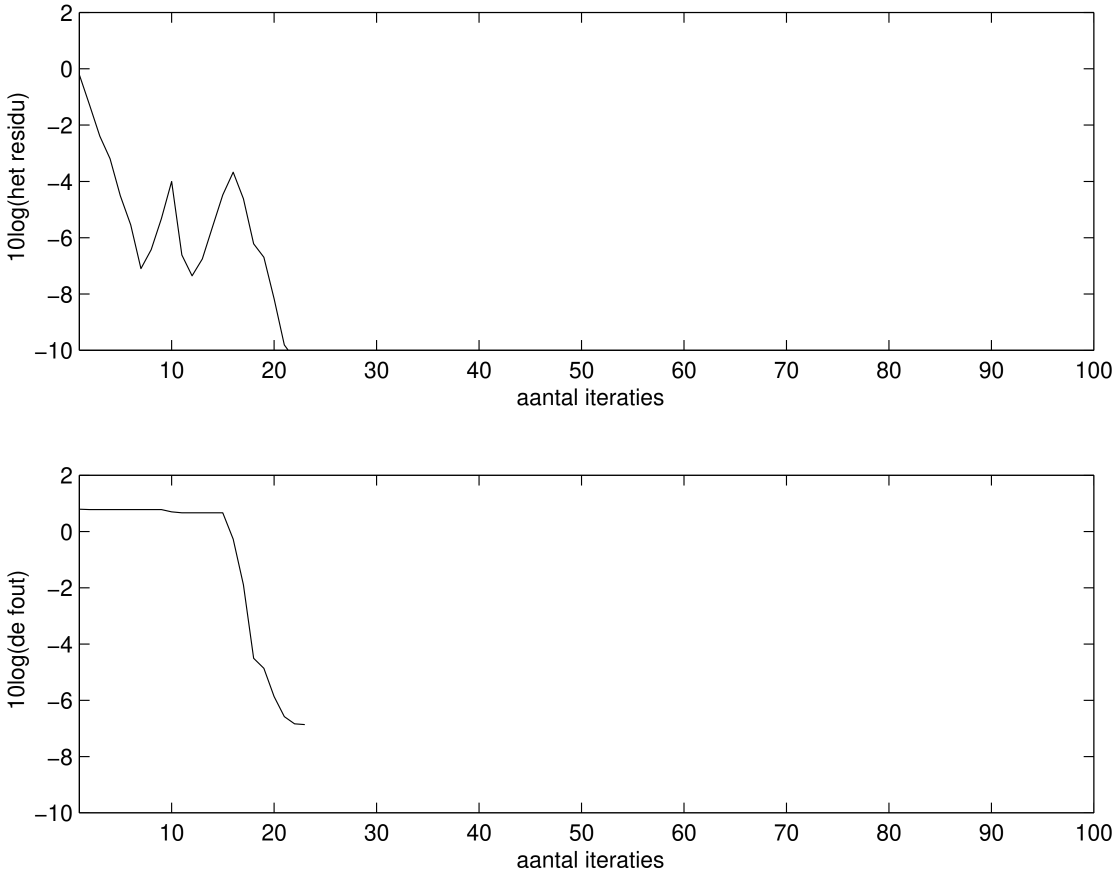
gridsize	approx. λ	aantal iter.	jump	2-norm res
22x9	9.55E-08	28	10	9.20E-09
30x9	6.79E-08	34	12	1.65E-09
30x18	3.31E-08	43	17	6.35E-09

Tabel 2.13: ILU preconditionering met nul start vektor.

gridsize	approx. λ	aantal iter.	jump	2-norm res
22x9	8.44E-09	62	16	1.13E-09
30x9	5.52E-09	75	18	7.86E-10
30x18	3.14E-09	120	33	7.09E-10

Tabel 2.14: Diagonaal preconditionering met nul start vektor.

Eens te meer blijkt dat ILU, in veel opzichten, veel betere resultaten oplevert dan diagonaal preconditionering. Met behulp van het Matlab programma hebben wij naast dit ook naar het gedrag van het residu $\|\boldsymbol{r}\|$ en de fout $\|\boldsymbol{e}\|$ gekeken. De resultaten die hierbij horen zijn in figuur 2.7 gegeven. We zien dat de fout tijdens de eerste 9 iteraties vrijwel konstant blijft. Pas op de tiende iteratie (het moment waarin de eerste kleine eigenwaarde ontdekt

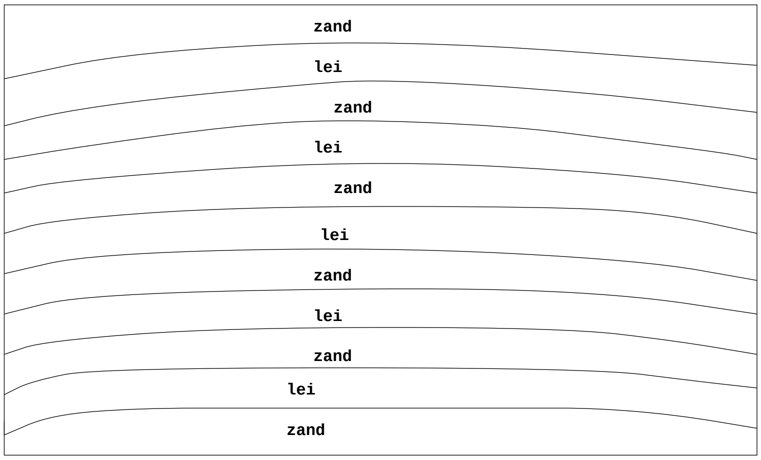


Figuur 2.7: Het residu en de berekende fout in het geval van ILU preconditionering

wordt) is de fout wat kleiner geworden en deze wordt nog echt klein als de tweede kleine eigenwaarde gevonden is. Dit resultaat illustreert het verband tussen de fout in de oplossing $\boldsymbol{x}$ ($\boldsymbol{e}_k = \boldsymbol{x}_k - \boldsymbol{x}$) en het vinden van de kleinste eigenwaarden. Het residu ($\boldsymbol{r}_k = \boldsymbol{A}\boldsymbol{x}_k - \boldsymbol{b}$) daarentegen geeft geen bruikbare informatie, immers $\left\|\boldsymbol{r}_8\right\|$ is klein terwijl $\left\|\boldsymbol{x}_8 - \boldsymbol{x}\right\|$ groot is.
Uitbreiding van het voorbeeld naar een gebied met 11 afwisselende lagen (figuur 2.8) laat dezelfde tendens zien. De resultaten staan in de tabellen 2.15 en 2.16.

gridsize	approx. λ	aantal iter.	jump	2-norm res
30x11	3.54E-08	77	13	3.36E-09
30x21	3.53E-08	78	13	1.48E-09
40x21	2.34E-08	92	15	4.53E-09

Tabel 2.15: ILU preconditionering met nul start.



Figuur 2.8: Gebied Ω met 11 lagen

gridsize	approx. λ	aantal iter.	jump	2-norm res
30x11	2.63E-09	231	11	2.01E-10
30x21	2.64E-09	245	14	4.04E-10
40x21	1.99E-09	274	16	3.13E-10

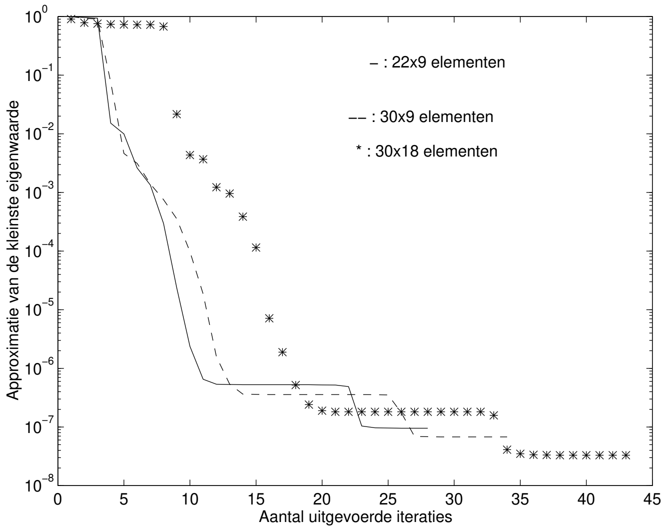
Tabel 2.16: Diagonaal preconditionering met nul start.

Merk hierbij op dat wegens de grootte van het gebied, het aantal gridpunten in dit geval verhoogd is.
Deze tabellen geven slechts een bevestiging van wat al eerder gevonden is. In de volgende tabel maken we een overzicht van het aantal benodigde iteraties en de iteratie waarbij de jump optreedt, behorende bij de fijnste roosters in het geval van ILU preconditionering.

aantal lagen	aantal iteraties	jump
3	35	24
5	43	17
11	92	15

Tabel 2.17: Resultaten behorende bij de fijnste roosters voor ILU

Het blijkt dat de approximatie van de kleinste eigenwaarde λ van belang is. Teneinde het gedrag van λ te kunnen volgen, wordt λ (althans zijn approximatie) tegen het aantal benodigde iteraties uitgezet. Ter illustratie beperken we ons hier tot slechts twee grafieken behorende bij het probleem van het 5 lagen gebied. Figuur 2.9 correspondeert met de convergentie van de kleinste eigenwaarde als functie van het aantal iteraties, bij het gebruik van ILU preconditionering. Figuur 2.10 geeft de convergentie van λ weer, voor het geval dat we diagonaal preconditionering toepassen. Merk op: in iedere figuur zijn drie grafieken getekend die met de drie verschillende roosters corresponderen. In het vervolg analyseren wij alleen de resultaten van het fijnste rooster (de grafiek aangegeven met *), voor de twee andere gevallen geldt precies hetzelfde.

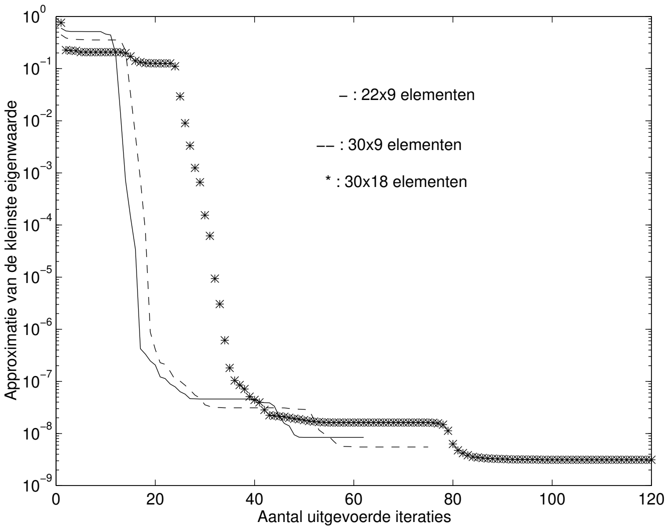


Figuur 2.9: ILU preconditionering voor een 5 lagen probleem.

De grafieken vertonen het volgende gedrag:

- om convergentie te bereiken, in het geval van ILU preconditionering zijn veel minder iteraties nodig dan bij diagonaal preconditionering.

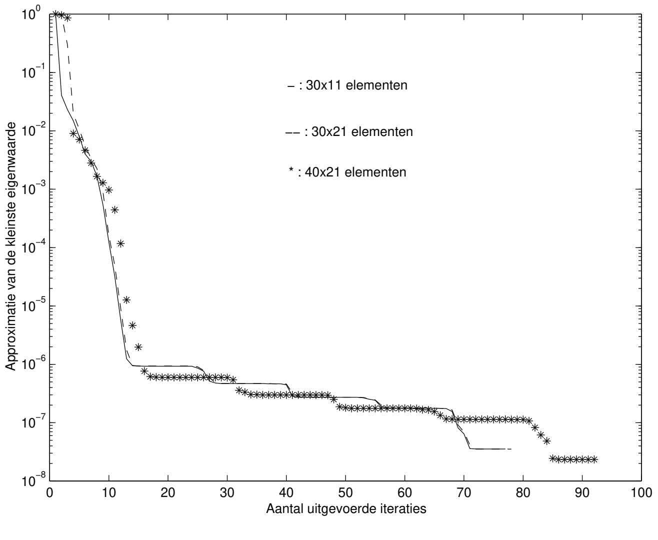
- in beide figuren, vinden twee belangrijk sprongen (jumps) plaats, de eerste sprong gebeurt in de eerste fase (tijdens de eerste 10 tot 25 iteraties voor respektievelijk ILU- en diagonaal-preconditionering). De tweede sprong is vlak voor het einde van het proces te zien. Om meer inzicht over de gevonden resultaten te krijgen, hebben we dezelfde experimenten herhaald, voor het geval van één leilaag (gebied met drie lagen) en ook voor het gebied met 5 leilagen (11 lagen in totaal). Opmerkelijk daarbij was dat in het geval van 1 leilaag, we maar één grote sprong zagen, terwijl in het geval van 5 leilagen, er 5 sprongen zichtbaar zijn (zie Figuur 2.11). Om vervolgens na te gaan of de sprong van het residu ongeveer op dezelfde iteratieslag plaats vindt als wanneer de kleinste eigenwaarden worden "ontdekt", kijken we naar Figuur 2.12. We zien dat het maximum in het residu precies op hetzelfde moment plaats vindt als een kleine eigenwaarde wordt "ontdekt"(vergelijk Figuur 2.11 en 2.12). Als er een conclusie is die uit deze experimenten getrokken mag worden, dan is dat dat het aantal geobserveerde sprongen het aantal gevonden kleine eigenwaarden aanduidt. Sterker nog de sprong blijkt alleen bij de overgang lei-zand (en niet andersom) plaats te vinden. Hierbij wordt gekeken vanuit de Dirichlet rand naar beneden. Dit resultaat blijkt uit een experiment met een gebied van slechts twee lagen. De resultaten hierbij duiden op het feit dat bij zand-lei overgang geen kleine eigenwaarde te observeren is, in tegenstelling tot een lei-zand overgang waar wel één kleine eigenwaarde te constateren is.
De vraag nu is of dit alleen opgaat voor een lagen struktuur of ook voor het fysisch interessante geval van insluitingen. Dat geval zullen we in de volgende paragraaf bekijken.



Figuur 2.10: diagonaal preconditionering voor een 5 lagen probleem.

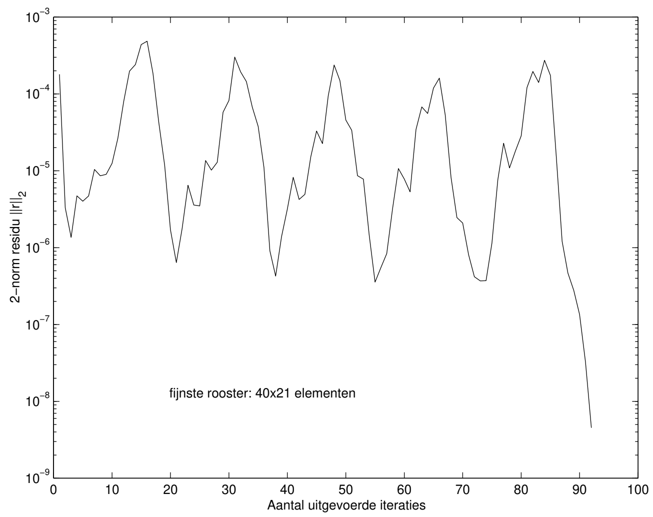
2.5.2 Gebied Ω met insluitingen

Teneinde de invloed van een insluiting te onderzoeken, bekijken we twee gevallen:
- eerst beschouwen we het geval van zandlaag ingesloten door lei (zie Figuur 2.13(I)),
- daarna het omgekeerde geval (leilaag ingesloten door zand, Figuur 2.13(II)). In de ta-

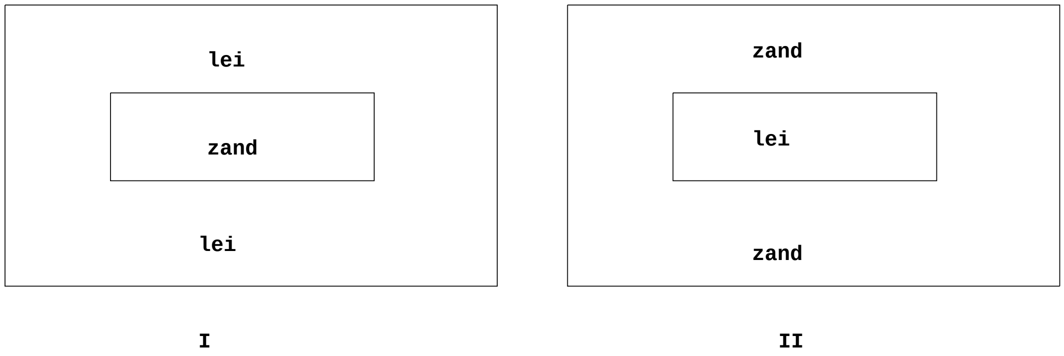


Figuur 2.11: ILU preconditionering voor een 11 lagen probleem.

bellen 2.18 en 2.19 zien we de resultaten van de experimenten behorende bij het eerste geval.



Figuur 2.12: De sprongen die het residu maakt gedurend het iteratie proces.



Figuur 2.13: een zandlaag ingesloten door de lei (I) resp. een leilaag ingesloten door zand (II)

gridsize	approx. λ	aantal iter.	jump	2-norm res
15x9	2.89E-08	31	20	3.17E-09
30x9	2.06E-08	33	20	1.49E-09
30x18	1.17E-08	49	26	2.07E-09

Tabel 2.18: ILU preconditionering voor een zandlaag in lei.

gridsize	approx. λ	aantal iter.	jump	2-norm res
15x9	2.47E-09	64	41	3.03E-10
30x9	2.21E-09	76	47	2.81E-10
30x18	9.93E-10	128	79	2.04E-10

Tabel 2.19: Diagonaal preconditionering voor een zandlaag in lei.

Het lijkt erop dat het gedrag van deze insluiting volledig analoog is aan die van een doorlopende zandlaag.
Tenslotte hebben wij gekeken naar het geval dat een leilaag ingesloten wordt door zand (zie Figuur 2.13(II)). De resultaten van de bijbehorende experimenten vatten we in tabellen 2.20 en 2.21 samen.

gridsize	approximatie van λ	aantal iteraties
15x9	8.22E-02	30
30x9	7.37E-02	31
30x18	1.72E-02	50

Tabel 2.20: ILU preconditionering voor een leilaag in zand.

gridsize	approximatie van λ	aantal iteraties
15x9	8.28E-03	64
30x9	4.83E-03	79
30x18	1.70E-03	129

Tabel 2.21: Diagonaal preconditionering voor een leilaag in zand.

We constateren dat er geen jump in beide tabellen voorkomt, omdat de kleinste eigenwaarde van de orde 10^{-3} is. Dit is een heel opvallend resultaat. Naast dit resultaat, blijkt ILU ook hier een veel betere preconditionering te zijn voor ons probleem.
Kennelijk is de kleinste eigenwaarde van groot belang voor het iteratieproces. Om het gedrag van de iteratieprocessen beter te kunnen verklaren, gaan we in het volgende hoofdstuk een nader onderzoek doen naar de eigenwaarden van deze problemen.

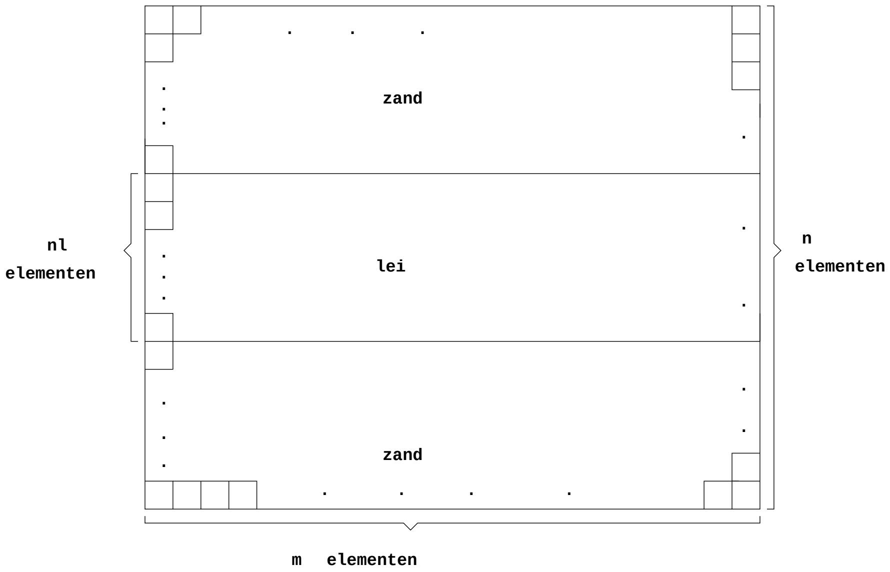
Hoofdstuk 3

Effect van de preconditioneringen op de kleinste eigenwaarden

Uit de experimenten beschreven in hoofdstuk 2 is gebleken dat, behalve in het geval van een leilaag ingesloten door een zandlaag, de kleinste eigenwaarde een hele kleine waarde aanneemt.
Tevens hebben we gezien dat preconditionering een aanzienlijke verbetering van de convergentiesnelheid geeft. Een probleem dat hier vermeld moet worden, is dat het enige tijd duurt voordat de kleinste eigenwaarde gevonden is. Daarom wordt in dit hoofdstuk een nader onderzoek gedaan naar dit probleem.
Om het gedrag van de eigenwaarden te onderzoeken, is een vereenvoudigd model geconstrueerd waarin het gebied uit rechte lagen bestaat (Figuur 3.1). Om de matrix op te bouwen is hier voor de eenvoud een eindige differentie methode gebruikt. Deze is op een zodanige wijze aangepast dat hij identiek is aan de gebruikte eindige elementen methode indien alle driehoekjes op dezelfde wijze geordend zijn. Dat wil zeggen de diagonalen in de vierhoeken moeten alle dezelfde kant uitwijzen.
Met behulp van het Matlabpakket, zijn alle eigenwaarden van de discretisatiematrix berekend. Het Matlab programma is toegevoegd in appendix 1. Eerst bekijken we het gebied Ω zoals gegeven in Figuur 3.1.

3.1 Zonder preconditionering

Als we geen preconditionering toepassen, dan krijgen we voor het aantal kleinste eigenwaarden (van de orde 10^{-7}) van de matrix, de resultaten in Tabel 3.1.



Figuur 3.1: Gebied Ω bestaand uit drie rechte lagen

gridsize: nxm	aantal kleine eigenwaarden	aantal elementen (nl) in de leilaag
7x1	2	2
7x2	3	2
7x3	4	2
7x7	8	2
10x1	3	3
10x2	5	3
10x3	7	3
10x10	21	3
13x13	40	4
19x19	96	6
25x25	176	8

Tabel 3.1: Het aantal verkregen kleine eigenwaarden in het niet gepreconditoneerde geval.

Uit deze Tabel kunnen we concluderen dat het aantal kleine eigenwaarden dat verkregen wordt, gelijk is aan: $(nl-1)m+1$, waarbij m het aantal elementen in de horizontale en n in de verticale richting is. Het aantal gekozen elementen in de verticale richting van de leilaag is gelijk aan nl (zie Figuur 3.1). Bekijken we de struktuur van de discretisatie matrix $\boldsymbol{A}$, dan ziet deze er bij benadering als volgt uit:

$$\boldsymbol{A} = \begin{pmatrix} \boldsymbol{H} & \boldsymbol{K} & & & \\ \boldsymbol{K}^T & \boldsymbol{V}(\epsilon) & \epsilon\boldsymbol{K} & & \\ & \epsilon\boldsymbol{K}^T & \epsilon\boldsymbol{H} & \epsilon\boldsymbol{K} & \\ & & \epsilon\boldsymbol{K}^T & \boldsymbol{V}(\epsilon) & \boldsymbol{K} \\ & & & \boldsymbol{K}^T & \boldsymbol{H}_n \end{pmatrix} \tag{3.1}$$

De blokmatrix $\boldsymbol{H}$ (3.2) correspondeert met de matrix behorende bij de zandlaag (Laplace matrix), de matrix $\epsilon\boldsymbol{H}$ behoort bij de leilaag, $\boldsymbol{K}$ is de zogenaamde koppelingsmatrix (3.3), $\boldsymbol{V}(\epsilon)$ is een matrix waarvan de componenten coefficiënten zijn van de punten aan de overgang (3.4) en $\boldsymbol{H}_n$ is de matrix behorende bij het Neumann probleem (3.5). Om zo'n struktuur van de matrix te krijgen is er voor een horizontale nummering gekozen.

$$\boldsymbol{H} = \begin{pmatrix} 2 & -1 & & & & \\ -1 & 4 & -1 & \mathbf{0} & & \\ \mathbf{0} & -1 & 4 & -1 & & \\ & & \ddots & \ddots & \ddots & \\ & & & -1 & 4 & -1 \\ & & & & -1 & 2 \end{pmatrix}, \tag{3.2}$$

$$\boldsymbol{K} = \begin{pmatrix} -0.5 & & & & \\ & -1 & & & \\ & & -1 & & \\ & & & \ddots & \\ & & & & -0.5 \end{pmatrix}, \tag{3.3}$$

$$\boldsymbol{V}(\epsilon) = \begin{pmatrix} 1+\epsilon & -(1+\epsilon)/2 & & & \\ -(1+\epsilon)/2 & 2(1+\epsilon) & -(1+\epsilon)/2 & & \\ & \ddots & \ddots & \ddots & \\ & & -(1+\epsilon)/2 & 2(1+\epsilon) & -(1+\epsilon)/2 \\ & & & -(1+\epsilon)/2 & 1+\epsilon \end{pmatrix}, \tag{3.4}$$

en

$$\boldsymbol{H}_n = \begin{pmatrix} 2 & -1 & & & \\ -1 & 4 & -1 & & \\ & \ddots & \ddots & \ddots & \\ & & -0.5 & 2 & -0.5 \\ & & & -0.5 & 1 \end{pmatrix}, \tag{3.5}$$

Hier wordt aangenomen dat de permeabiliteit k in zand gelijk is aan 1, terwijl in de leilaag k gelijk is aan een kleine waarde ϵ ($\approx 10^{-7}$). De dimensie van $\epsilon\boldsymbol{H}$ is $(nl-1)m$. We laten nu zien dat er minstens $(nl-1)\cdot m$ kleine eigenwaarden zijn (als we geen preconditionering

toepassen). We gaan een orthogonalisatie principe toepassen om de matrix $\boldsymbol{A}$ op een diagonaal vorm te brengen.
Stel dat er een $\boldsymbol{Q} \in I\!R^{N1 \times N1}$ is zodat

$$\boldsymbol{Q}^T \cdot \begin{pmatrix} \boldsymbol{H} & \boldsymbol{K} \\ \boldsymbol{K}^T & \boldsymbol{V}(\epsilon) \end{pmatrix} \cdot \boldsymbol{Q} = \begin{pmatrix} d_1 & & \boldsymbol{0} \\ & \ddots & \\ \boldsymbol{0} & & d_{N1} \end{pmatrix}, \tag{3.6}$$

waarbij de coefficiënten $d_i \in [ch^2, 4]$, met h de stapgrootte en c een konstante. Verder nemen we aan dat er een matrix $\boldsymbol{R} \in I\!R^{N1 \times N1}$ is zodanig dat

$$\boldsymbol{R}^T \cdot \begin{pmatrix} \boldsymbol{V}(\epsilon) & \boldsymbol{K} \\ \boldsymbol{K}^T & \boldsymbol{H}_n \end{pmatrix} \cdot \boldsymbol{R} = \begin{pmatrix} \tilde{d}_1 & & \boldsymbol{0} \\ & \ddots & \\ \boldsymbol{0} & & \tilde{d}_{N1} \end{pmatrix}. \tag{3.7}$$

Als $\epsilon = 0$ dan correspondeert $\begin{pmatrix} \boldsymbol{V}(\epsilon) & \boldsymbol{K} \\ \boldsymbol{K}^T & \boldsymbol{H} \end{pmatrix}$ met een Neumann probleem. Hiervoor geldt dat er één eigenwaarde 0 is en de rest van de eigenwaarden ligt in het interval $[ch^2, 1]$. Als $\epsilon < ch^2$ dan geldt dat er maar één $\tilde{d}_i$ is van de orde ϵ, dit is een gevolg van de tweede Gershgorin cirkel stelling. Deze stelling luidt:

Stelling(2^e Gershgorin cirkel stelling)

Indien n Gershgorin-cirkels van $\boldsymbol{A}$ geisoleerd liggen van de overige, dan bevatten deze n cirkels tezamen precies n eigenwaarden van $\boldsymbol{A}$, elke eigenwaarde geteld naar zijn multipliciteit.

Merk op dat bij de eerste overgang van zand naar lei geen kleine (O(ϵ)) eigenwaarde is. De reden hiervoor is dat aan de bovenrand een Dirichlet randvoorwaarde opgelegd is. Bij de overgang van lei-zand is wel 1 kleine eigenwaarde te constateren.
We gaan nu de totale matrix $\boldsymbol{A}$ beschouwen en voeren daarvoor de symmetrische matrix $\boldsymbol{B}$ in:

$$\begin{aligned} \boldsymbol{B} &= \begin{pmatrix} \boldsymbol{Q}^T & & \boldsymbol{0} \\ & \boldsymbol{I} & \\ \boldsymbol{0} & & \boldsymbol{R}^T \end{pmatrix} \cdot \boldsymbol{A} \cdot \begin{pmatrix} \boldsymbol{Q} & & \boldsymbol{0} \\ & \boldsymbol{I} & \\ \boldsymbol{0} & & \boldsymbol{R} \end{pmatrix} \\ &= \begin{pmatrix} d_1 & & & & & & \\ & \ddots & & \epsilon\boldsymbol{Q}^T\boldsymbol{K} & & & \\ & & d_{N1} & & & & \\ & & \epsilon\boldsymbol{K}^T\boldsymbol{Q} & \epsilon\boldsymbol{H} & \epsilon\boldsymbol{R}^T\boldsymbol{K} & & \\ & & & \epsilon\boldsymbol{K}^T\boldsymbol{R} & \tilde{d}_1 & & \\ & & & & & \ddots & \\ & & & & & & \tilde{d}_{N1} \end{pmatrix}. \end{aligned} \tag{3.8}$$

Nu geldt dat de eigenwaarden van de matrix $\boldsymbol{B}$ gelijk zijn aan die van de matrix $\boldsymbol{A}$. Gebruiken we de stelling van Gershgorin, dan geldt dat er $(nl-1)m+1$ cirkels zijn met middelpunt ϵ en straal $\mathrm{O}(\epsilon)$. Voor de overige cirkels geldt dat het middelpunt bij benadering $> ch^2$. Is $\epsilon << ch^2$ dan liggen er $(nl-1)m+1$ cirkels gescheiden van de rest en dit betekent dat er precies zoveel kleine eigenwaarden zijn.

Voorbeeld

We bekijken het ééndimensionaal geval. We kiezen een 11×11 matrix $\boldsymbol{A}$ die op dezelfde gedaante wordt gebracht als in 3.1. De blokmatrices zoals in 3.6 en 3.7 is gegeven, zijn achtereenvolgens:

$$\boldsymbol{B}1 = \begin{pmatrix} 2 & -1 & & \\ -1 & 2 & -1 & \\ & -1 & 2 & -1 \\ & & -1 & 1+\epsilon \end{pmatrix}$$

met $\epsilon = 10^{-7}$ en

$$\boldsymbol{B}2 = \begin{pmatrix} 1+\epsilon & -1 & & \\ -1 & 2 & -1 & \\ & -1 & 2 & -1 \\ & & -1 & 1 \end{pmatrix}.$$

We voeren nu het matrixproduct 3.6 en 3.7 uit, waarbij $\boldsymbol{Q}$ en $\boldsymbol{R}$ de eigenvektoren van $\boldsymbol{B}1$ respectievelijk $\boldsymbol{B}2$ als kolommen hebben. De diagonaal matrices die we uit deze producten krijgen zijn respectievelijk

$$\boldsymbol{D}1 = \begin{pmatrix} 1.00000033333344 & & & \\ & 2.34729653896767 & & \\ & & 3.53208893822810 & \\ & & & 0.12061518947078 \end{pmatrix}$$

en

$$\boldsymbol{D}2 = \begin{pmatrix} 0.58578686440370 & & & \\ & 0.00000024999978 & & \\ & & 2.00000025000009 & \\ & & & 3.41421363559643 \end{pmatrix}.$$

We zien nu duidelijk dat de extra kleine eigenwaarde komt uit het product 3.7, dat wil zeggen uit de overgang lei-zand. Dit bevestigt onze eerdere waarneming dat de extra kleine eigenwaarde uit de overgang lei-zand komt. Bekijken we nu het product 3.8, dan zien we dat er in totaal 4 heel kleine eigenwaarden zijn te onderscheiden van de rest:

$$\boldsymbol{D} = \begin{pmatrix} \tilde{\boldsymbol{D}}1 & & \\ & \boldsymbol{D}_{lei} & \\ & & \tilde{\boldsymbol{D}}2 \end{pmatrix},$$

waarbij $\boldsymbol{D}_{lei}$ is de blokmatrix die in de leilaag geldt, met

$$\boldsymbol{D}_{lei} = \begin{pmatrix} 0.00000343453450 & & \\ & 0.00000206857074 & \\ & & 0.00000069635756 \end{pmatrix},$$

$\tilde{\boldsymbol{D}}1$ en $\tilde{\boldsymbol{D}}2$ zijn de volgende matrices

$$\tilde{\boldsymbol{D}}1 = \begin{pmatrix} 1.00000033333378 & & & \\ & 2.34729653896775 & & \\ & & 3.53208893822812 & \\ & & & 0.12061518947436 \end{pmatrix}$$

en

$$\tilde{\boldsymbol{D}}2 = \begin{pmatrix} 0.00000005053210 & & & \\ & 0.58578686440443 & & \\ & & 2.00000025000022 & \\ & & & 3.41421363559645 \end{pmatrix}.$$

In dit voorbeeld is het aantal kleine eigenwaarden gelijk aan 4 en dit klopt ook met de eerdere gegeven formule $((nl-1).m+1)$. We hebben in dit ééndimensionale probleem $(m = 1)$, 4 elementen in de verticale richting van de leilaag gekozen, en dus er geldt $(4-1).m+1 = 4$.

3.2 Diagonaal preconditionering

Een zeer verrassend resultaat wordt verkregen na toepassing van diagonaal preconditionering, namelijk dat het aantal kleine eigenwaarden zich reduceert tot slechts het aantal lei-zand overgangen. Dit resultaat doet ons denken aan het feite dat bij diagonaal schaling het aantal $(nl-1)\cdot m$, dat we in vorige paragraaf introduceerden, helemaal wegvalt. Dit is een aanzienlijk verbetering en met behulp van het bovenstaande voorbeeld zullen wij trachten dit te verifiëren. De diagonaal geschaalde matrix $\tilde{\boldsymbol{A}}$ ziet er nu als volgt uit

$$\tilde{\boldsymbol{A}} = \begin{pmatrix} \tilde{\boldsymbol{A}}_1 & & & & \\ \boldsymbol{c}_1 & c_2 & \boldsymbol{c}_3 & & \\ & \boldsymbol{c}_3^T & \tilde{\boldsymbol{A}}_2 & \boldsymbol{c}_4^T & \\ & & \boldsymbol{c}_4 & c_5 & \boldsymbol{c}_6 \\ & & & \boldsymbol{c}_6^T & \tilde{\boldsymbol{A}}_3 \end{pmatrix}$$

$\boldsymbol{c}_1, c_2, \boldsymbol{c}_3$ en $\boldsymbol{c}_4, c_5, \boldsymbol{c}_6$ zijn de koppelingsmatrices aan de overgangen (zand-lei en lei-zand) gegeven door

$\boldsymbol{c}_1 = (0, 0, -0.70710642763342)$
$c_2 = 1.0$
$\boldsymbol{c}_3 = (-0.00070710642763, 0, ..., 0)$

$\boldsymbol{c}_4 = (0, 0, -0.70710642763342)$
$c_5 = 1.0$ en
$\boldsymbol{c}_6 = (-0.70710642763342, 0, 0)$.
De bloktridiagonaalmatrices $\tilde{\boldsymbol{A}}_1$, $\tilde{\boldsymbol{A}}_2$ en $\tilde{\boldsymbol{A}}_3$ zijn gelijk aan

$$\tilde{\boldsymbol{A}}_1 = \begin{pmatrix} 1.0 & -0.5 & 0 \\ -0.5 & 1.0 & -0.5 \\ 0 & -0.5 & 1.0 \end{pmatrix}$$

$$\tilde{\boldsymbol{A}}_2 = \begin{pmatrix} 1.0 & -0.5 & 0 \\ -0.5 & 1.0 & -0.5 \\ 0 & -0.5 & 1.0 \end{pmatrix}$$

en

$$\tilde{\boldsymbol{A}}_3 = \begin{pmatrix} 1.0 & -0.5 & 0 \\ -0.5 & 1.0 & -0.70710678118655 \\ 0 & -0.70710678118655 & 1.0 \end{pmatrix}.$$

Voor de verificatie, hebben we ook naar de eigenwaarden van de matrix $\tilde{\boldsymbol{A}}$ gekeken:
0.50000016666571
0.61731656763547
0.07612046748875
0.29289304203722
1.38268343236453
1.00000000000000
1.92387953251125
1.70710695796278
1.49999983333429
0.00000004166660
1.99999995833340
duidelijk is nu te zien dat er maar één kleine eigenwaarde is.

3.3 Effect van ILU preconditionering

Uit een aantal experimenten die gedaan werden met het Matlab programma, is gebleken dat men bij ILU preconditionering, op één uitzondering na, dezelfde resultaten krijgt als bij de diagonaal schaling. Echter in het geval van ILU preconditionering en als we $m = 1$ kiezen, dan blijkt dat er geen enkele kleine eigenwaarde te zien is (dat is logisch omdat ILU in het ééndimensionaal geval, de volledige inverse van de matrix is).
Nogmaals dient hier opgemerkt te worden dat het aantal kleine eigenwaarden dat men in het gepreconditioneerde geval krijgt gelijk is aan het aantal lei-zand overgangen in het gebied (hoofdstuk 2).

Hoofdstuk 4

Aanpassing van het CG-proces

Inleiding

Het stelsel dat wij moeten oplossen is van de vorm:

$$\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}. \tag{4.1}$$

Stel dat λ_i de eigenwaarden zijn van de matrix $\boldsymbol{A}$, dan geldt:

$$\boldsymbol{A}\boldsymbol{v}_i = \lambda_i \boldsymbol{v}_i. \tag{4.2}$$

De eigenwaarden worden zo gerangschikt dat: $\lambda_1 \leq \lambda_2 \leq ... \leq \lambda_n$.
$\boldsymbol{v}_i$ zijn de bijbehorende eigenvektoren. De ruimte $\mathbb{R}^n$ is nu opgespannen door :

$$\mathbb{R}^n = \text{span}\{\boldsymbol{v}_i\}.$$

$\boldsymbol{v}_i$ zijn de orthonormale eigenvektoren van $\boldsymbol{A}$, dit betekent dat

$$< \boldsymbol{v}_i, \boldsymbol{v}_j >= \begin{cases} 1, & \text{als} \quad i=j \\ 0, & \text{als} \quad i \neq j \end{cases},$$

voor $i = 1, ..., n$.
Het systeem convergeert snel indien het conditiegetal $(\frac{\lambda_n}{\lambda_1})$ van de matrix $\boldsymbol{A}$ klein is. Uit het vorige hoofdstuk is gebleken dat de matrix $\boldsymbol{A}$ minstens 1 hele kleine eigenwaarde heeft. In het geval dat we k kleine eigenwaarden hebben met $k \geq 1$, geldt dan: $\lambda_1 \leq ... \leq \lambda_k << \lambda_{k+1} \leq ... \leq \lambda_n$.
Deze kleine eigenwaarden veroorzaken een slechte conditie van de matrix $\boldsymbol{A}$ en we weten dat het conditiegetal de convergentie van het CG-proces bepaalt ([9] pagina 225)

$$\|\boldsymbol{x}_i - \boldsymbol{x}\|_{\boldsymbol{A}} \leq 2\|\boldsymbol{x}_0 - \boldsymbol{x}\|_{\boldsymbol{A}} \left(\frac{\sqrt{K_2(\boldsymbol{A})} - 1}{\sqrt{K_2(\boldsymbol{A})} + 1} \right)^i. \tag{4.3}$$

Met $K_2(\boldsymbol{A})$ wordt het conditiegetal bedoeld en $\boldsymbol{x}_i$ is de i^{de} approximatie van $\boldsymbol{x}$. Teneinde de invloed van de zeer kleine eigenwaarde te elimineren, zullen wij trachten een geschikte aanpassing van de CG-methode te verzinnen. Een van de mogelijkheden daarvoor is dat we een projectie op de fou $\boldsymbol{x} - \boldsymbol{x}_0$ toepassen zodanig dat de bijdrage van de eigenvektoren behorende bij de kleinste eigenwaarden geélimineerd wordt uit de beginfout.

4.1 Projectie op een nieuwe oplossingsruimte

De projectie wordt op twee verschillende oplossingsruimten uitgevoerd. In paragraaf 4.1.1 bekijken we de ruimte opgespannen door de eigenvektoren behorende bij de kleinste eigenwaarden. Daarna in paragraaf 4.1.2 wordt de projectie op de benaderende vektoren behandeld. We beschouwen een projectie op een ruimte die opgespannen wordt door de kolommen van $\boldsymbol{V}$.
De projectiematrix $\boldsymbol{P}$ waarvan we gebruik zullen maken, is dan van de volgende vorm:

$$\boldsymbol{P} = \boldsymbol{I} - \boldsymbol{V}\boldsymbol{E}^{-1}(\boldsymbol{A}\boldsymbol{V})^T. \tag{4.4}$$

Waarbij $\boldsymbol{E} = (\boldsymbol{A}\boldsymbol{V})^T\boldsymbol{V}$. Merk op dat $\boldsymbol{V}$ geen vierkante matrix is.
Er geldt ook $\boldsymbol{P}^2 = \boldsymbol{P}$:

$$\begin{aligned}
\boldsymbol{P}^2 &= (\boldsymbol{I} - \boldsymbol{V}\boldsymbol{E}^{-1}(\boldsymbol{A}\boldsymbol{V})^T)\cdot(\boldsymbol{I} - \boldsymbol{V}\boldsymbol{E}^{-1}(\boldsymbol{A}\boldsymbol{V})^T) \\
&= \boldsymbol{I} - 2\boldsymbol{V}\boldsymbol{E}^{-1}(\boldsymbol{A}\boldsymbol{V})^T + (\boldsymbol{V}\boldsymbol{E}^{-1}(\boldsymbol{A}\boldsymbol{V})^T)^2 \\
&= \boldsymbol{I} - 2\boldsymbol{V}\boldsymbol{E}^{-1}(\boldsymbol{A}\boldsymbol{V})^T + (\boldsymbol{V}((\boldsymbol{A}\boldsymbol{V})^T\boldsymbol{V})^{-1}((\boldsymbol{A}\boldsymbol{V})^T\boldsymbol{V})((\boldsymbol{A}\boldsymbol{V})^T\boldsymbol{V})^{-1}(\boldsymbol{A}\boldsymbol{V})^T) \\
&= \boldsymbol{I} - 2\boldsymbol{V}\boldsymbol{E}^{-1}(\boldsymbol{A}\boldsymbol{V})^T + \boldsymbol{V}\boldsymbol{E}^{-1}(\boldsymbol{A}\boldsymbol{V})^T \\
&= \boldsymbol{I} - \boldsymbol{V}\boldsymbol{E}^{-1}(\boldsymbol{A}\boldsymbol{V})^T \\
&= \boldsymbol{P}.
\end{aligned}$$

Één van voordelen van de toepassing van deze projectie is het verbeteren van het conditiegetal van de matrix.

Verbetering van het conditiegetal van de gepreconditioneerde matrix

We passen een projectie toe zodanig dat we in plaats van het residu

$$\boldsymbol{r}_0 = \sum_{i=1}^{n} \alpha_i \boldsymbol{v}_i, \tag{4.5}$$

een andere residu krijgen, namelijk

$$\tilde{\boldsymbol{r}}_0 = \sum_{i=k+1}^{n} \alpha_i \boldsymbol{v}_i. \tag{4.6}$$

Dit heeft tot gevolg dat het effectieve conditiegetal van een matrix $\boldsymbol{A}$ aanzienlijk verbeterd wordt. Het effectieve conditiegetal (K^{eff}) van een matrix $\boldsymbol{A}$ behorende bij een startvektor gegeven door

$$\boldsymbol{x}_0 = \sum_{i=1}^{n} \alpha_i \boldsymbol{v}_i, \tag{4.7}$$

is als volgt gedefiniëerd:

$$K^{eff}(\boldsymbol{A}, \boldsymbol{x}_0) = \frac{\beta}{\gamma}, \tag{4.8}$$

waarbij $\gamma = \min\{\lambda_j | \alpha_j \neq 0\}$ en
$\beta = \max\{\lambda_j | \alpha_j \neq 0\}$.
In ons geval geldt dus , met $\boldsymbol{A}$ de gepreconditioneerde matrix, dat in plaats van
$K_2(\boldsymbol{A}) = \frac{\lambda_n}{\lambda_1}$,
wordt nu K^{eff} gebruikt en die is gelijk aan: $K_2^{eff}(\boldsymbol{A}) = \frac{\lambda_n}{\lambda_{k+1}}$.
De eigenwaarde λ_{k+1} wordt nu de kleinste effectieve eigenwaarde van $\boldsymbol{A}$. dit natuurlijk betekent dat de convergentie van het CG-proces aanzienlijk versneld wordt.

4.1.1 Projectie op de eigenvektoren behorende bij de kleinste eigenwaarden

Om de matrices $\boldsymbol{V}$, $\boldsymbol{E}$ en $\boldsymbol{A}$ in 4.4 te kunnen definiëren, is het noodzakelijk de onderscheid tussen de symmetrisch gepreconditioneerde matrix $\hat{\boldsymbol{A}}$ (met $\hat{\boldsymbol{A}} = \boldsymbol{L}^{-1}\boldsymbol{A}\boldsymbol{L}^{-T}$) en de niet symmetrisch gepreconditioneerde $\bar{\boldsymbol{A}} = (\boldsymbol{L}\boldsymbol{L}^T)^{-1}\boldsymbol{A}$ te maken.

De projectie-matrix in het symmetrische geval

De kolommen $\boldsymbol{v}_i$ van de matrix $\boldsymbol{V} \in \mathbb{R}^{n \times k}$ corresponderen in dit geval met de eigenvektoren behorende bij de kleinste eigenwaarden van de matrix $\hat{\boldsymbol{A}}$. Er geldt dus
$\hat{\boldsymbol{A}}\boldsymbol{v}_i = \lambda_i \boldsymbol{v}_i$, voor $i = 1, 2, ..., k$.
De lengte van iedere kolomvektor $\boldsymbol{v}_i$ is gelijk aan 1,voor $i = 1, ..., k$. De matrix $\boldsymbol{E}$ is gedefiniëerd als

$$\boldsymbol{E} = (\hat{\boldsymbol{A}}\boldsymbol{V})^T\boldsymbol{V}. \tag{4.9}$$

Merk op dat we het $\boldsymbol{A}$-inprodukt gebruiken omdat CG ook in $\boldsymbol{A}$-norm werkt. De matrix $\boldsymbol{A}$ in 4.4 wordt in dit symmetrische geval vervangen door $\hat{\boldsymbol{A}}$.
Passen we deze projectie toe op de matrix $\hat{\boldsymbol{A}}$, dan krijgen we:

$$\tilde{\boldsymbol{A}} = \boldsymbol{P}\hat{\boldsymbol{A}} = \hat{\boldsymbol{A}} - \boldsymbol{V}\boldsymbol{E}^{-1}(\hat{\boldsymbol{A}}\boldsymbol{V})^T\hat{\boldsymbol{A}}. \tag{4.10}$$

Er geldt nu dat:

$$\tilde{\boldsymbol{A}}\boldsymbol{V} = \boldsymbol{0}. \tag{4.11}$$

Bewijs:

$$\begin{aligned}
\tilde{\boldsymbol{A}}\boldsymbol{V} &= \boldsymbol{P}\hat{\boldsymbol{A}}\boldsymbol{V} \\
&= \hat{\boldsymbol{A}}\boldsymbol{V} - \boldsymbol{V}\boldsymbol{E}^{-1}(\hat{\boldsymbol{A}}\boldsymbol{V})^T\hat{\boldsymbol{A}}\boldsymbol{V} \\
&= \boldsymbol{V}\boldsymbol{\Lambda} - \boldsymbol{V}\boldsymbol{E}^{-1}(\hat{\boldsymbol{A}}\boldsymbol{V})^T\boldsymbol{V}\boldsymbol{\Lambda} \\
&= \boldsymbol{V}\boldsymbol{\Lambda} - \boldsymbol{V}\boldsymbol{\Lambda} \\
&= \boldsymbol{0},
\end{aligned}$$

waarbij $\boldsymbol{\Lambda}$ een diagonaal matrix is met als diagonaal elementen de eigenwaarden van $\hat{\boldsymbol{A}}$. We kunnen nu concluderen dat de matrix $\tilde{\boldsymbol{A}}$ in (4.11) een k-voudige eigenwaarde 0 heeft, de resterende eigenwaarden zijn $\lambda_{k+1}, ..., \lambda_n$. Omdat dit bewijs geldt voor de eigenvektoren $\boldsymbol{v}_1, \boldsymbol{v}_2, ..., \boldsymbol{v}_k$ geldt hij ook voor iedere lineaire combinatie van deze eigenvektoren.

De projectie-matrix in het niet symmetrische geval

We zullen de projectie-matrix voor het niet symmetrische geval afleiden uit de projectie $\boldsymbol{P}$ die in het symmetrische geval wordt gebruikt. Dat gaat als volgt:
Het stelsel dat voor het symmetrische geval wordt opgelost is van de vorm

$$\boldsymbol{P}\hat{\boldsymbol{A}}\boldsymbol{y} = \boldsymbol{P}\boldsymbol{L}^{-1}\boldsymbol{b}, \tag{4.12}$$

ofwel

$$\boldsymbol{P}\boldsymbol{L}^{-1}\boldsymbol{A}\boldsymbol{x} = \boldsymbol{P}\boldsymbol{L}^{-1}\boldsymbol{b}. \tag{4.13}$$

vermenigvuldigen we nu (4.13) links met de matrix $\boldsymbol{L}^{-T}$, dan geldt

$$\boldsymbol{L}^{-T}\boldsymbol{P}\boldsymbol{L}^{T}\boldsymbol{L}^{-T}\boldsymbol{L}^{-1}\boldsymbol{A}\boldsymbol{x} = \boldsymbol{L}^{-T}\boldsymbol{P}\boldsymbol{L}^{T}\boldsymbol{L}^{-T}\boldsymbol{L}^{-1}\boldsymbol{b}. \tag{4.14}$$

Dit is gelijk aan:

$$\boldsymbol{L}^{-T}\boldsymbol{P}\boldsymbol{L}^{T}\bar{\boldsymbol{A}}\boldsymbol{x} = \boldsymbol{L}^{-T}\boldsymbol{P}\boldsymbol{L}^{T}\boldsymbol{L}^{-T}\boldsymbol{L}^{-1}\boldsymbol{b}, \tag{4.15}$$

dit betekent dus dat de projectie-matrix ($\tilde{\boldsymbol{P}}$) in dit geval is gelijk aan:

$\tilde{\boldsymbol{P}} = \boldsymbol{L}^{-T}\boldsymbol{P}\boldsymbol{L}^{T}$. Met

$$\begin{aligned}
\tilde{\boldsymbol{P}} &= \boldsymbol{I} - \boldsymbol{L}^{-T}\boldsymbol{V}\boldsymbol{E}^{-1}(\boldsymbol{L}^{-1}\boldsymbol{A}\boldsymbol{L}^{-T}\boldsymbol{V})^{T}\boldsymbol{L}^{T} \\
&= \boldsymbol{I} - \boldsymbol{L}^{-T}\boldsymbol{V}\boldsymbol{E}^{-1}(\boldsymbol{A}\boldsymbol{L}^{-T}\boldsymbol{V})^{T}\boldsymbol{L}^{-T}\boldsymbol{L}^{T} \\
&= \boldsymbol{I} - \tilde{\boldsymbol{V}}\boldsymbol{E}^{-1}(\boldsymbol{A}\tilde{\boldsymbol{V}})^{T},
\end{aligned} \tag{4.16}$$

waarbij $\tilde{\boldsymbol{V}} = \boldsymbol{L}^{-T}\boldsymbol{V}$. Eenvoudig is in te zien dat als $\boldsymbol{V}$ bestaat uit eigenvektoren van $\hat{\boldsymbol{A}}$, $\tilde{\boldsymbol{V}}$ eigenvektoren van $\bar{\boldsymbol{A}}$ bevat.
De matrix $\boldsymbol{E}$ voldoet aan:

$$\begin{aligned}
\boldsymbol{E} &= (\hat{\boldsymbol{A}}\boldsymbol{V})^{T}\boldsymbol{V} \\
&= (\boldsymbol{L}^{-1}\boldsymbol{A}\boldsymbol{L}^{-T}\boldsymbol{V})^{T}\boldsymbol{V} \\
&= (\boldsymbol{A}\boldsymbol{L}^{-T}\boldsymbol{V})^{T}\boldsymbol{L}^{-T}\boldsymbol{V} \\
&= (\boldsymbol{A}\tilde{\boldsymbol{V}})^{T}\tilde{\boldsymbol{V}}.
\end{aligned} \tag{4.17}$$

Het produkt $\tilde{\boldsymbol{P}}\bar{\boldsymbol{A}}\tilde{\boldsymbol{V}}$ levert ook een 0 op:

$$\begin{aligned}
\tilde{\boldsymbol{A}}\tilde{\boldsymbol{V}} &= \tilde{\boldsymbol{P}}\bar{\boldsymbol{A}}\tilde{\boldsymbol{V}} \\
&= \bar{\boldsymbol{A}}\tilde{\boldsymbol{V}} - \tilde{\boldsymbol{V}}\boldsymbol{E}^{-1}(\boldsymbol{A}\tilde{\boldsymbol{V}})^{T}\bar{\boldsymbol{A}}\tilde{\boldsymbol{V}} \\
&= \tilde{\boldsymbol{V}}\boldsymbol{\Lambda} - \tilde{\boldsymbol{V}}\boldsymbol{E}^{-1}(\boldsymbol{A}\tilde{\boldsymbol{V}})^{T}\tilde{\boldsymbol{V}}\boldsymbol{\Lambda} \\
&= \boldsymbol{0},
\end{aligned}$$

ook hier geldt dat de diagonaal elementen van de diagonaal matrix $\mathbf{\Lambda}$ eigenwaarden van $\bar{\boldsymbol{A}}$ zijn.
Dit betekent dus dat de matrix $\tilde{\boldsymbol{P}}\bar{\boldsymbol{A}}$ ook een k-voudige 0 eigenwaarde heeft.

Opmerking

De projecties die hierboven worden gebruikt zijn gebaseerd op de eigenvektoren van de gepreconditioneerde matrices ($\hat{\boldsymbol{A}}$ en $\bar{\boldsymbol{A}}$ met de respectievelijke eigenvektoren $\boldsymbol{v}_i$ en $\tilde{\boldsymbol{v}}_i$). Wij zullen een benadering gebruiken van de eigenvektoren behorende bij de kleinste eigenwaarden. Om deze eigenvektoren te bepalen zou men de methode van Lanczos kunnen gebruiken. Een belangrijke vraag hierbij is: hoe groot is de invloed van de nauwkeurigheid van de benaderingen op de convergentie van het proces?. Om op deze vraag een antwoord te kunnen geven, zullen we in hoofdstuk 5 een onderzoek doen met behulp van een Matlab programma.

4.1.2 Projectie op de benaderende eigenruimte

Een andere mogelijkheid, om de (eigen)vektoren te bepalen, is gebaseerd op wat in de fysica gebeurt.

Een op de fysica gebaseerde methode

De bepaling van de eigenwaarden en eigenvektoren is duur en bij eenmalig oplossen van het stelsel vergelijkingen zeker niet zinvol. Uit de fysica kunnen we ook proberen inzicht te krijgen over het gedrag van de eigenvektoren (dat komt pas in het volgende hoofdstuk). We gaan dan niet meer de eigenwaarden en eigenvektoren berekenen, doch slechts een benadering van de eigenvektoren ($\boldsymbol{v}_i$, $i = 1, ..., k$) behorende bij de kleinste eigenwaarden op grond van fysische argumenten schatten. Dus in plaats van de oorspronkelijke basis bestaande uit deze eigenvektoren $\boldsymbol{v}_i$, gaan we een nieuwe basis $\boldsymbol{w}_i$ voor $i = 1, ..., k$ gebruiken.
We gaan nu projecteren op de ruimte opgespannen door de vektoren $\boldsymbol{w}_1, ..., \boldsymbol{w}_k$, in plaats van de eigenvektoren $\boldsymbol{v}_i$. Daarmee zullen we de kleinste eigenwaarden niet nodig hebben. De karakteristieken waaraan deze vektoren $\boldsymbol{w}_i$, voor $i = 1, ..., k$, moeten voldoen zullen we later zien. Als we de projectie, gebaseerd op deze vektoren, gaan toepassen dan geldt voor het matrix-vektor produkt het volgende

$$\tilde{\boldsymbol{A}}\boldsymbol{y} = \boldsymbol{P}\boldsymbol{A}\boldsymbol{y} = (\boldsymbol{I} - \boldsymbol{W}\boldsymbol{F}^{-1}(\boldsymbol{A}\boldsymbol{W})^T)\boldsymbol{A}\boldsymbol{y}, \tag{4.18}$$

waarbij $\boldsymbol{W}$ een matrix is waarvan de kolommen $\boldsymbol{w}_i$, voor $i = 1, ..., k$, een basis vormen voor de deelruimte span$\{\boldsymbol{v}_1, ..., \boldsymbol{v}_k\}$, het zijn dus een soort "benaderende" vektoren voor $\boldsymbol{v}_i$. Er geldt met andere woorden span$\{\boldsymbol{v}_1, ..., \boldsymbol{v}_k\} \approx$ span$\{\boldsymbol{w}_1, ..., \boldsymbol{w}_k\}$.
Bovendien is $\boldsymbol{I} - \boldsymbol{W}\boldsymbol{F}^{-1}(\boldsymbol{A}\boldsymbol{W})^T$ de nieuwe projectie-matrix met $\boldsymbol{F}$ gedefiniëerd als (vergelijk 4.9):

$\boldsymbol{F} = (\boldsymbol{AW})^T\boldsymbol{W}$.

Bij het gebruik van projectie moet men de startoplossing van het iteratieproces aanpassen. In de volgende paragraaf zullen we dit uitvoerig bespreken, zowel voor $\hat{\boldsymbol{A}}$ als $\bar{\boldsymbol{A}}$.

4.2 Aanpassing van de startvektor $\boldsymbol{x}_0$

De startvektor mag geen bijdrage hebben in de ruimte opgespannen door de vektoren $\boldsymbol{v}_i$, voor $i = 1, 2, ..., k$. De vektoren $\boldsymbol{v}_i$ stellen, of de eigenvektoren behorende bij de kleinste eigenwaarden dan wel de benaderende vektoren, voor. We definiëren de nieuwe startvektor $\tilde{\boldsymbol{x}}_0$ als:

$$\tilde{\boldsymbol{x}}_0 = \boldsymbol{x}_0 + \sum_{i=1}^{k} a_i \boldsymbol{v}_i. \tag{4.19}$$

De coefficiënten a_i worden nu zo bepaald dat het verschil $\boldsymbol{x} - \tilde{\boldsymbol{x}}_0$ loodrecht staat op iedere vektor $\boldsymbol{v}_j$, voor $j = 1, ..., k$. Dus:

$$(\boldsymbol{x} - \tilde{\boldsymbol{x}}_0) \perp \boldsymbol{v}_j, \text{ voor } j = 1, ..., k.$$

Ofwel:

$$(\boldsymbol{x} - \tilde{\boldsymbol{x}}_0)^T \boldsymbol{v}_j = 0, \text{ voor } j = 1, ..., k.$$

Gebruikmakend van (4.19) geeft dit:

$$(\boldsymbol{x} - \boldsymbol{x}_0, \boldsymbol{v}_j) = \sum_{i=1}^{k} a_i (\boldsymbol{v}_i, \boldsymbol{v}_j). \tag{4.20}$$

De keuze van het inwendig produkt staat nog open. Omdat de projectie $\boldsymbol{P}$ in het $\boldsymbol{A}$-inprodukt is gekozen, ligt een dergelijke keuze voor het inprodukt in (4.20) ook voor de hand. Een andere belangrijk reden voor deze keuze is dat $\boldsymbol{x}$ onbekend is, maar $\boldsymbol{Ax}$ niet. Merk op dat de startoplossing voor beide varianten (de symmetrische en de niet symmetrische) apart moet worden bekeken. Dat doen we door het inwendige produkt voor beide gevallen te beschouwen.

De startvektor $\hat{\boldsymbol{y}}_0$ en het inprodukt bij de symmetrisch matrix $\hat{\boldsymbol{A}}$

We lossen het stelsel

$$\boldsymbol{P}\hat{\boldsymbol{A}}\boldsymbol{y} = \boldsymbol{P}\boldsymbol{L}^{-1}\boldsymbol{b}, \tag{4.21}$$

op met $\hat{\boldsymbol{A}} = \boldsymbol{L}^{-1}\boldsymbol{A}\boldsymbol{L}^{-T}$ en $\boldsymbol{y} = \boldsymbol{L}^T\boldsymbol{x}$ ($\boldsymbol{x}$ is de exacte oplossing van (4.1)).
De oplossing $\boldsymbol{y}$ kunnen we schrijven als

$$\boldsymbol{y} = \boldsymbol{y}_0 + \sum_{i=1}^{k} a_i \boldsymbol{v}_i + \sum_{i=k+1}^{n} a_i \boldsymbol{v}_i, \tag{4.22}$$

waarbij $\boldsymbol{y}_0 = \boldsymbol{L}^T \boldsymbol{x}_0$. We nemen aan dat de vektoren $\boldsymbol{v}_i$ alle lineair onafhankelijk zijn en we kiezen de vektoren $\boldsymbol{v}_{k+1}, ..., \boldsymbol{v}_n$ zodat ze $\hat{\boldsymbol{A}}$-loodrecht staan op de ruimte opgespannen door $\boldsymbol{v}_1, ..., \boldsymbol{v}_k$.
De iteratie wordt gestart met een nader te kiezen startvektor $\hat{\boldsymbol{y}}_0$. Voor de iterand $\boldsymbol{y}_m$ geldt het volgende:

$\boldsymbol{y}_m = \hat{\boldsymbol{y}}_0 + \boldsymbol{e}_m$, waarbij $\boldsymbol{e}_m \in K_m(\tilde{\boldsymbol{A}}, \boldsymbol{P}\boldsymbol{L}^{-1}\hat{\boldsymbol{r}}_0)$.
Omdat $\tilde{\boldsymbol{A}}\boldsymbol{v}_i = 0$, voor $i = 1, 2, ..., k$ (zie (4.11)), geldt dat

$\boldsymbol{e}_m = \sum_{i=k+1}^{n} a_i^m \boldsymbol{v}_i$.

Om de vektor $\boldsymbol{y}_m$ naar $\boldsymbol{y}$ te laten convergeren ($\boldsymbol{y}_m \longrightarrow \boldsymbol{y}$), moeten we $\hat{\boldsymbol{y}}_0$ zo kiezen dat:

$$\hat{\boldsymbol{y}}_0 = \boldsymbol{y}_0 + \sum_{i=1}^{k} a_i \boldsymbol{v}_i. \tag{4.23}$$

Voor $j \in \{1, ..., k\}$ en $i \in \{k+1, ..., n\}$, geldt nu

$(\hat{\boldsymbol{A}}\boldsymbol{v}_j)^T \boldsymbol{v}_i = \boldsymbol{v}_j^T \hat{\boldsymbol{A}} \boldsymbol{v}_i = 0$

Als we nu (4.22) met $\hat{\boldsymbol{A}}\boldsymbol{v}_j$ (voor $j = 1, ..., k$) gaan vermenigvuldigen, dan geldt

$$\begin{aligned} (\hat{\boldsymbol{A}}\boldsymbol{v}_j)^T(\boldsymbol{y} - \boldsymbol{y}_0) &= (\hat{\boldsymbol{A}}\boldsymbol{v}_j)^T(\sum_{i=1}^{k} a_i \boldsymbol{v}_i + \sum_{j=k+1}^{n} a_i \boldsymbol{v}_i) \\ &= \sum_{i=1}^{k} a_i (\hat{\boldsymbol{A}}\boldsymbol{v}_j)^T \boldsymbol{v}_i, \end{aligned} \tag{4.24}$$

We kunnen dus in dit geval het inprodukt (4.20) vervangen door het $\hat{\boldsymbol{A}}$-inprodukt. Dat levert:

$$(\boldsymbol{y} - \boldsymbol{y}_0, \boldsymbol{v}_j)_{\hat{\boldsymbol{A}}} = \sum_{i=1}^{k} a_i (\boldsymbol{v}_i, \boldsymbol{v}_j)_{\hat{\boldsymbol{A}}}. \tag{4.25}$$

De coëfficiënten a_i van de vektor $\boldsymbol{a}$ voldoen nu aan:

$$\boldsymbol{a} = \boldsymbol{E}^{-1} \cdot \boldsymbol{d}, \tag{4.26}$$

waarbij

$$\boldsymbol{a} = \begin{pmatrix} a_1 \\ \vdots \\ a_k \end{pmatrix} \text{ en } \boldsymbol{d} = \begin{pmatrix} (\boldsymbol{y} - \boldsymbol{y}_0, \boldsymbol{v}_1)_{\hat{\boldsymbol{A}}} \\ \vdots \\ (\boldsymbol{y} - \boldsymbol{y}_0, \boldsymbol{v}_k)_{\hat{\boldsymbol{A}}} \end{pmatrix} = \begin{pmatrix} (\boldsymbol{L}^{-1}\boldsymbol{b} - \hat{\boldsymbol{A}}\boldsymbol{y}_0, \boldsymbol{v}_1) \\ \vdots \\ (\boldsymbol{L}^{-1}\boldsymbol{b} - \hat{\boldsymbol{A}}\boldsymbol{y}_0, \boldsymbol{v}_k) \end{pmatrix}.$$

De overgang van het gewone inprodukt (4.20) naar het $\hat{\boldsymbol{A}}$-inprodukt is hier van groot belang omdat het inprodukt in (4.20) niet bepaald kan worden, immers de oplossing $\boldsymbol{x}$ is onbekend. Echter het inprodukt

$$\begin{aligned}
(\boldsymbol{y}-\boldsymbol{y}_0,\boldsymbol{v}_j)_{\hat{\boldsymbol{A}}} &= (\hat{\boldsymbol{A}}\boldsymbol{y}-\hat{\boldsymbol{A}}\boldsymbol{y}_0,\boldsymbol{v}_j) \\
&= (\boldsymbol{L}^{-1}\boldsymbol{b}-\hat{\boldsymbol{A}}\boldsymbol{y}_0,\boldsymbol{v}_j), \qquad (4.27)
\end{aligned}$$

kan wel uitgerekend worden voor $j = 1, 2, ..., k$.

De startvektor $\hat{\boldsymbol{x}}_0$ en het inprodukt in het geval van de niet symmetrische matrix $\tilde{\boldsymbol{A}}$

Ook in het niet symmetrische geval moeten we de startvektor op de juiste manier gaan aanpassen. In de symmetrische variant hadden we gezien dat de startoplossing $\hat{\boldsymbol{y}}_0$ gelijk is aan $\boldsymbol{L}^T\boldsymbol{x}_0$. Dit betekent dat de startoplossing in het niet symmetrische geval, gebruikmakend van (4.23), geschreven kan worden als:

$$\begin{aligned}
\hat{\boldsymbol{x}}_0 &= \boldsymbol{L}^{-T}\hat{\boldsymbol{y}}_0 \\
&= \boldsymbol{x}_0 + \sum_{i=1}^{k} a_i \boldsymbol{L}^{-T}\boldsymbol{v}_i \\
&= \boldsymbol{x}_0 + \sum_{i=1}^{k} a_i \tilde{\boldsymbol{v}}_i. \qquad (4.28)
\end{aligned}$$

De coefficiënten a_i van de vektor $\boldsymbol{a}$ voldoen in dit geval aan:

$$\boldsymbol{a} = \boldsymbol{E}^{-1}\begin{pmatrix} \boldsymbol{v}_1^T \\ \vdots \\ \boldsymbol{v}_k^T \end{pmatrix}\boldsymbol{L}^{-1}\boldsymbol{r}_0$$

$$= \boldsymbol{E}^{-1}\begin{pmatrix} (\boldsymbol{L}^{-T}\boldsymbol{v}_1)^T \\ \vdots \\ (\boldsymbol{L}^{-T}\boldsymbol{v}_k)^T \end{pmatrix}\boldsymbol{r}_0 = \boldsymbol{E}^{-1}\begin{pmatrix} \tilde{\boldsymbol{v}}_1^T \\ \vdots \\ \tilde{\boldsymbol{v}}_{\boldsymbol{k}}^T \end{pmatrix}\boldsymbol{r}_0.$$

4.2.1 Aanpassing van het residu

Bij het nieuwe stelsel ($\tilde{\boldsymbol{A}}\tilde{\boldsymbol{x}} = \tilde{\boldsymbol{b}}$) behoort nu het nieuwe residu $\tilde{\boldsymbol{r}}_0$. Gebruik makend van (4.19) moeten we nogmaals onderscheid maken tussen twee residuen $\tilde{\boldsymbol{r}}_0$, namelijk :

Het residu behorende bij $\hat{\boldsymbol{A}}$

Het residu $\tilde{\boldsymbol{r}}_0$ wordt in dit geval gelijk aan:

$$\begin{aligned}
\tilde{\boldsymbol{r}}_0 &= \tilde{\boldsymbol{b}} - \tilde{\boldsymbol{A}}\tilde{\boldsymbol{x}}_0 \\
&= \boldsymbol{P}\boldsymbol{L}^{-1}\boldsymbol{b} - \boldsymbol{P}(\hat{\boldsymbol{A}}\tilde{\boldsymbol{x}}_0) \\
&= \boldsymbol{P}\cdot(\boldsymbol{L}^{-1}\boldsymbol{b} - \hat{\boldsymbol{A}}\boldsymbol{x}_0) - \boldsymbol{P}\cdot\hat{\boldsymbol{A}}\cdot(\sum_{i=1}^{k} a_i\boldsymbol{v}_i) \\
&= (\boldsymbol{I} - \boldsymbol{V}\boldsymbol{E}^{-1}(\hat{\boldsymbol{A}}\boldsymbol{V})^T)\cdot\boldsymbol{r}_0. \qquad (4.29)
\end{aligned}$$

De passage van de voorlaatste naar de laatste formule geldt omdat, zoals we al eerder hadden gezien, $\tilde{\boldsymbol{A}}\boldsymbol{v}_i = \boldsymbol{P}\hat{\boldsymbol{A}}\boldsymbol{v}_i = 0$ (4.11). We zien dat dit residu onafhankelijk is van de keuze van de coefficiënten a_i. Vergelijking (4.29) laat zien dat het residu nu geen component meer in de richting van de vektoren $\boldsymbol{v}_i$ heeft, voor $i = 1, 2, ..., k$.

Het residu behorende bij de matrix $\bar{\boldsymbol{A}}$

We definiëren het residu $\tilde{\boldsymbol{r}}_0$, in dit geval, door de volgende vergelijking:

$$\begin{aligned}
\tilde{\boldsymbol{r}}_0 &= \tilde{\boldsymbol{b}} - \tilde{\boldsymbol{A}}\tilde{\boldsymbol{x}}_0 \\
&= \tilde{\boldsymbol{P}}(\boldsymbol{L}\boldsymbol{L}^T)^{-1}\boldsymbol{b} - \tilde{\boldsymbol{P}}(\bar{\boldsymbol{A}}\tilde{\boldsymbol{x}}_0) \\
&= \tilde{\boldsymbol{P}}(\boldsymbol{L}\boldsymbol{L}^T)^{-1}\cdot(\boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}_0) - \tilde{\boldsymbol{P}}(\boldsymbol{L}\boldsymbol{L}^T)^{-1}\cdot\boldsymbol{A}\cdot(\sum_{i=1}^{k} a_i\boldsymbol{v}_i) \\
&= (\boldsymbol{I} - \tilde{\boldsymbol{V}}\boldsymbol{E}^{-1}(\boldsymbol{A}\tilde{\boldsymbol{V}})^T)(\boldsymbol{L}\boldsymbol{L}^T)^{-1}\cdot\boldsymbol{r}_0. \qquad (4.30)
\end{aligned}$$

Merk op dat bij de overstap van de voorlaatste naar de laatste vergelijking in (4.30) gebruik is gemaakt van het feit dat
$\tilde{\boldsymbol{P}}(\boldsymbol{L}\boldsymbol{L}^T)^{-1}\boldsymbol{A}(\sum_{i=1}^{k} a_i\boldsymbol{v}_i) = (\boldsymbol{I} - \boldsymbol{V}\boldsymbol{E}^{-1}(\boldsymbol{A}\boldsymbol{V})^T)(\boldsymbol{L}\boldsymbol{L}^T)^{-1}\cdot\boldsymbol{A}\cdot(\sum_{i=1}^{k} a_i\boldsymbol{v}_i) = 0$. Voordat we nu met de Matlab experimenten beginnen, zullen we eerst het gepreconditioneerde CG-algoritme en de te gebruiken projectie weergeven.

4.3 Aanpassing van de gepreconditioneerde CG-algoritme bij het gebruik van projecties

Een belangrijke punt voorafgaand aan het iteratie-proces, zoals we in de vorige paragraaf hebben gezien, is dat we de startoplossing $\hat{\boldsymbol{x}}_{\boldsymbol{0}}$ zodanig moeten kiezen dat $\boldsymbol{x} - \hat{\boldsymbol{x}}_{\boldsymbol{0}}$ geen component meer in de richting van de vektoren $\boldsymbol{v}_i$ (dit zijn de eigenvektoren en/of de benaderende vektoren).
Het Matlab programma waarin dit allemaal wordt geïmplementeerd staat in Appendix 1. Vervolgens beschouwen we twee varianten van de gepreconditioneerde matrix $\boldsymbol{A}$, namelijk de symmetrische $\hat{\boldsymbol{A}}$ (=$\boldsymbol{L}^{-1}\boldsymbol{A}\boldsymbol{L}^{-T}$) en de niet symetrische $\bar{\boldsymbol{A}}$ (=$(\boldsymbol{L}\boldsymbol{L}^T)^{-1}\boldsymbol{A}$). De matrix $(\boldsymbol{L}\boldsymbol{L}^T)$ is de zogenaamde incomplete Choleski-ontbinding van $\boldsymbol{A}$.

i = 0; $\boldsymbol{x}_0 = 0$; $\boldsymbol{r}_0 = \boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}$; $\boldsymbol{z}_0 = \boldsymbol{P}\boldsymbol{r}_0$ en $\boldsymbol{zr}_0 = \boldsymbol{z}_0$;
while $\left\|\boldsymbol{r}/\boldsymbol{b}\right\| > \epsilon$ en $i < 200$ **do**
  i = i + 1;
  $\boldsymbol{r}_{old} = (\boldsymbol{r}_{i-1}, \boldsymbol{z}_{i-1})$;
  $\alpha_i = \frac{\boldsymbol{r}_{old}}{(\boldsymbol{zr}_i, \boldsymbol{A}\boldsymbol{zr}_i)}$;
  $\boldsymbol{x}_i = \boldsymbol{x}_{i-1} + \alpha_i \boldsymbol{zr}_{i-1}$;
  $\boldsymbol{r}_i = \boldsymbol{r}_{i-1} - \alpha_i \boldsymbol{A}\boldsymbol{zr}_{i-1}$;
  $\boldsymbol{z}_i = \boldsymbol{P}\boldsymbol{r}_i$;
  $\beta_i = (\boldsymbol{r}_i, \boldsymbol{z}_i)/\boldsymbol{r}_{old}$;
  $\boldsymbol{zr}_i = \boldsymbol{z}_i + \beta_i \boldsymbol{zr}_{i-1}$.
end while

Hierbij geldt dat ϵ de voor dit algoritme gebruikte nauwkeurigheid, in ons Matlab programma is dat gelijk aan 10^{-11}, i geeft de iteratiestap aan, $\boldsymbol{r}$ het residu, $\boldsymbol{zr}$ de zoekrichting en de matrix $\boldsymbol{P}$ is voor iedere van de boven genoemde varianten anders.
We bekijken in het vervolg ieder variant appart om de matrix $\boldsymbol{P}$ op de juiste manier weer te geven.

De matrix $\boldsymbol{P}$ behorende bij $\hat{\boldsymbol{A}}$

In het symmetrisch geval ($\hat{\boldsymbol{A}} = \boldsymbol{L}^{-1}\boldsymbol{A}\boldsymbol{L}^{-T}$) beschouwen we het volgende stelsel:

$$(\hat{\boldsymbol{A}})(\hat{\boldsymbol{x}}) = (\boldsymbol{L}^{-1}\boldsymbol{A}\boldsymbol{L}^{-T})(\boldsymbol{L}^T\boldsymbol{x}) = \boldsymbol{L}^{-1}\boldsymbol{b}, \tag{4.31}$$

de projectie-matrix $\boldsymbol{P}$ is in dit geval:

$$\boldsymbol{P} = \boldsymbol{I} - \boldsymbol{Y}\boldsymbol{Q}^{-1}(\hat{\boldsymbol{A}}\boldsymbol{Y})^T.$$

Waarbij de matrix $\boldsymbol{Y}$ heeft als kolommen $\boldsymbol{y}_i$ de eigenvektoren behorende bij de kleinste eigenwaarden van $\hat{\boldsymbol{A}}$, of de benaderende vektoren. We lossen hierbij het stelsel

$$\boldsymbol{P}\hat{\boldsymbol{A}}\hat{\boldsymbol{x}} = \boldsymbol{P}\hat{\boldsymbol{b}} \tag{4.32}$$

op. De matrix $\boldsymbol{Q}$ voldoet aan:

$$\boldsymbol{Q}_{ij} = (\hat{\boldsymbol{A}}\boldsymbol{y}_i)^T\boldsymbol{y}_j.$$

Merk hierbij op dat we in dit geval de beide vektoren $\boldsymbol{y}_j$ en $\hat{\boldsymbol{A}}\boldsymbol{y}_i$, in het geheugen, op moeten slaan. De matrix $\boldsymbol{P}$ is dus in dit symmetrische geval een pure projectie.

De matrix $\boldsymbol{P}$ behorende bij $\bar{\boldsymbol{A}}$

Beschouwen we nu de niet symmetrische matrix $\bar{\boldsymbol{A}}$, dan voldoet de projectie-matrix $\boldsymbol{P}$ in het algoritme aan:

$$\boldsymbol{P} = (\boldsymbol{I} - \bar{\boldsymbol{Y}}\bar{\boldsymbol{Q}}^{-1}(\boldsymbol{A}\bar{\boldsymbol{Y}})^T) \cdot (\boldsymbol{L}\boldsymbol{L}^T)^{-1},$$

merk op dat dit een combinatie is van een een projectie $(\boldsymbol{I} - \bar{\boldsymbol{Y}}\bar{\boldsymbol{Q}}^{-1}(\boldsymbol{A}\bar{\boldsymbol{Y}})^T)$ en een preconditionering $(\boldsymbol{L}\boldsymbol{L}^T)^{-1}$. Er geldt bovendien dat $\bar{\boldsymbol{Y}}$ een matrix is waarvan de kolommen $(\bar{\boldsymbol{y}}_{\boldsymbol{i}})$ de eigenvektoren (of de benaderingen daarvan) van $(\boldsymbol{L}\boldsymbol{L}^T)^{-1}\boldsymbol{A}$ en de matrix $\bar{\boldsymbol{Q}}$ is gegeven door:

$$\bar{\boldsymbol{Q}}_{ij} = (\boldsymbol{A}\bar{\boldsymbol{y}}_{\boldsymbol{i}})^T \bar{\boldsymbol{y}}_{\boldsymbol{j}}.$$

Het stelsel dat we in dit geval moeten oplossen is:

$$\boldsymbol{P}\boldsymbol{A}\boldsymbol{x} = \boldsymbol{P}\boldsymbol{b}. \tag{4.33}$$

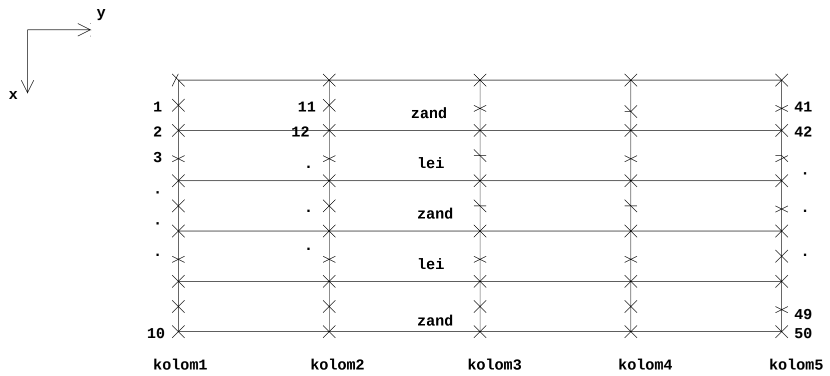
In dit geval is de projectie gebaseerd op het $\boldsymbol{A}$-inprodukt in plaats van $\bar{\boldsymbol{A}}$-inprodukt. Hierin worden ook de bijbehorende vektoren $\bar{\boldsymbol{y}}_{\boldsymbol{i}}$ en $\boldsymbol{A}\bar{\boldsymbol{y}}_{\boldsymbol{i}}$ opgeslagen.

Hoofdstuk 5

MATLAB experimenten

We geven allereerst de nodige informatie over het gebied dat we gaan bestuderen, namelijk

a) het gebied bestaat uit 5 lagen (3 zandlagen en 2 leilagen), zie figuur 5.1

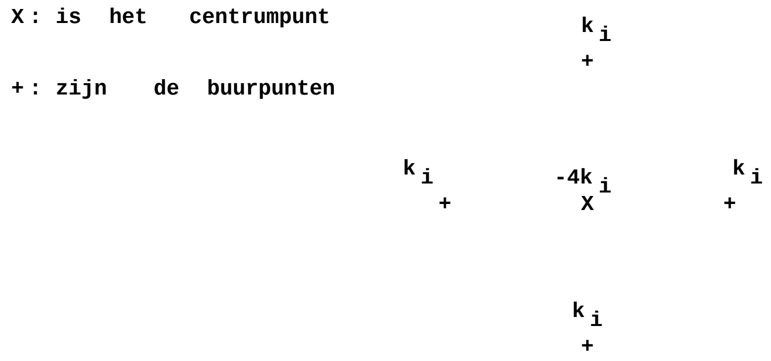


Figuur 5.1: Het beschouwde gebied met de roosterpunten

b) de diepte wordt aangegeven met de x-as (de verticale richting),
c) de lengte wordt met de y-as aangegeven,
d) we kiezen in de verticale richting 10 elementen (rechthoekjes) en 4 elementen in de horizontale richting en
e) de nummering wordt verticaal gekozen, in de positieve x-richting, van boven naar beneden zoals in figuur 5.1 wordt aangegeven.
Let op: de bovenste rij punten correspondeert met de Dirichlet-rand.
In figuur 5.2 is het discretisatie-molecuul gegeven, dat geheel in zand en/of in lei ligt.

5.1 Projectie op de eigenvektoren behorende bij de kleinste eigenwaarden

We zullen in deze paragraaf een aantal resultaten bespreken, die met behulp van het Matlabprogramma (zie Appendix 1) verkregen zijn. We beperken ons hier tot ILU preconditi-

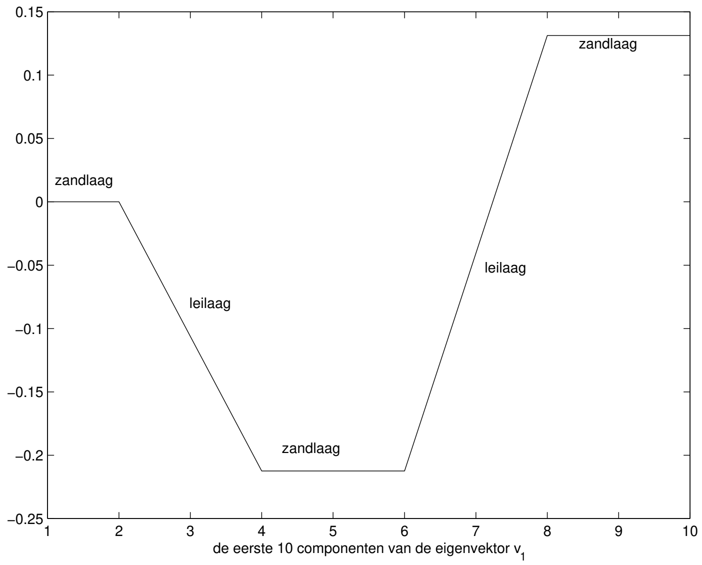


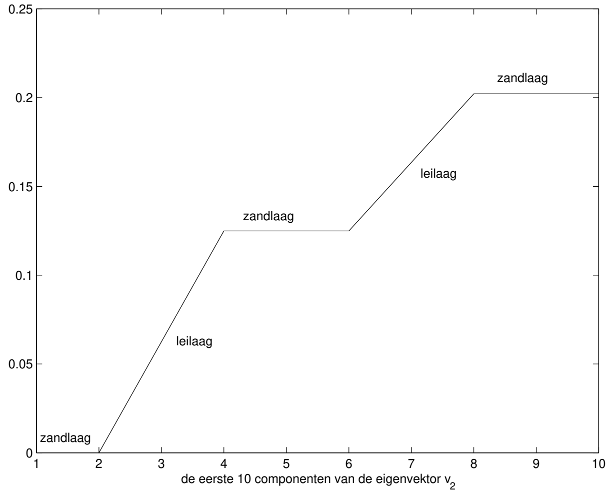
Figuur 5.2: Een roosterpunt x geheel in zand (i=z) en lei (i=l)

onering. In voorgaande hoofdstukken hebben we gezien dat het aantal kleine eigenwaarden in het gepreconditioneerde geval gelijk is aan het aantal lei-zand overgangen in het gebied. Voor het gebied 5.1 betekent dit dat we 2 kleine eigenwaarden krijgen: λ_1 en λ_2 met de bijbehorende eigenvektoren $\boldsymbol{v}_1$ en $\boldsymbol{v}_2$. De vektoren $\boldsymbol{v}_1$ en $\boldsymbol{v}_2$ zijn in figuur 5.3 gegeven en ze behoren tot de niet symmetrische gepreconditioneerde matrix (zie hoofdstuk 4). Als we naar kolom1 van het gebied kijken, dan zien we dat de componenten in de zandlaag konstant zijn terwijl in de leilaag de bijbehorende componenten een lineair verloop vertonen.

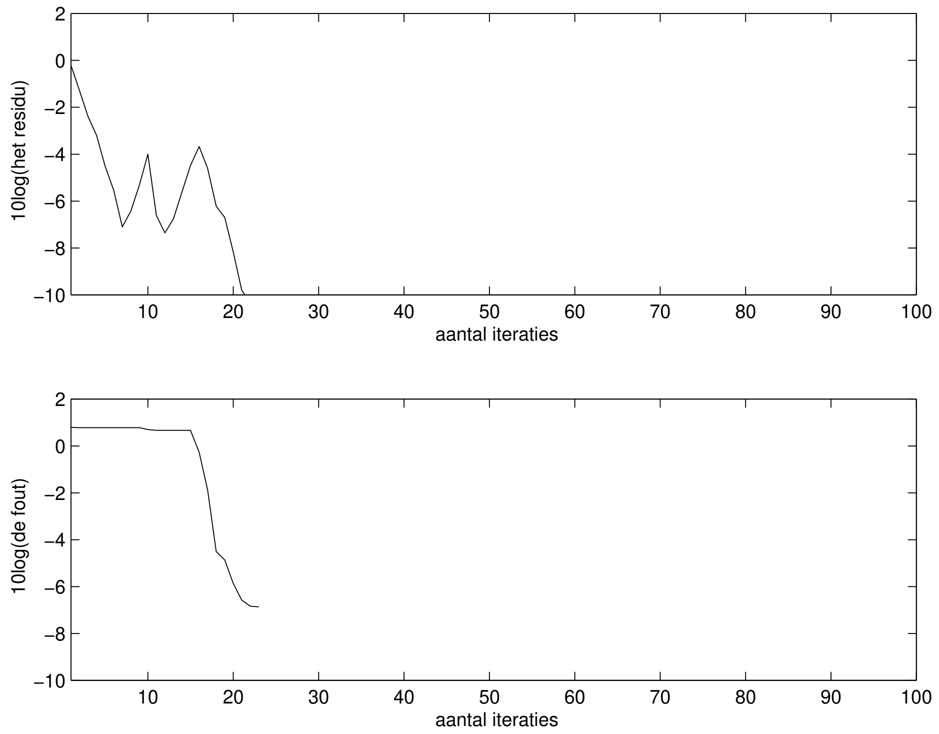
Belangrijk: de struktuur geschetst in figuur 5.3 geldt voor iedere kolom van het geroosterde gebied (figuur 5.1).

Teneinde de convergentie van het CG-proces te versnellen, moeten we dus van die vektoren $\boldsymbol{v}_1$ en $\boldsymbol{v}_2$ zien af te komen. Daartoe zullen we de projecties, die uitvoerig in hoofdstuk 4 zijn behandeld, gaan toepassen. Het effect van de projecties op de convergentie kunnen we duidelijk zien aan het residu en de fout. We hebben eerst het residu en de fout bekeken zonder toepassing van de projectie, de resultaten staan in figuur 5.4. Deze figuur toont 2 "bultjes/topjes" in het residu aan (2 gevonden kleine eigenwaarden). De figuren laten zien dat de convergentie heel slecht is. De fout blijft in eerste instantie konstant totdat de eerste kleine eigenwaarde ontdekt wordt. Op dat moment wordt de fout wat kleiner en pas als de tweede kleine eigenwaarde gevonden is, zakt de fout tot een kleine waarde. Dit gedrag van de fout, ten opzicht van het vinden van de kleinste eigenwaarde, hadden we al eerder in hoofdstuk 2 (zie figuur 2.1) gezien. Merk op dat de fout aan het einde een rare "knik" vertoont. Dit verschijnsel kan door verschillende factoren tot stand komen (bijvoorbeeld de afrondfouten en/of de slechte conditie van de matrix,...etc). Om dit gedrag (in het residu en de fout) kwijt te raken gaan we de projecties, op de eigenvektoren die bij deze eigenwaarden behoren, toepassen.





Figuur 5.3: De struktuur van de eerste 10 componenten van de eigenvektoren $\boldsymbol{v}_1$ en $\boldsymbol{v}_2$



Figuur 5.4: Het residu en de fout voor ILU zonder projectie

De eigenvektoren in figuur 5.3 zijn afkomstig van de matrix $(\boldsymbol{L}\boldsymbol{L}^T)^{-1}\boldsymbol{A}$. Er geldt dus: $(\boldsymbol{L}\boldsymbol{L}^T)^{-1}\boldsymbol{A}\boldsymbol{v}_i = \lambda_i \boldsymbol{v}_i$, voor $i = 1, 2$.
Beschouwen we nu de eigenvektoren $\hat{\boldsymbol{v}}_i$ behorende bij de symmetrische gepreconditioneerde matrix ($\hat{\boldsymbol{A}} = \boldsymbol{L}^{-1}\boldsymbol{A}\boldsymbol{L}^{-T}$), met: $\boldsymbol{L}^{-1}\boldsymbol{A}\boldsymbol{L}^{-T}\boldsymbol{v}_i = \lambda_i \boldsymbol{v}_i$, dan geldt dat

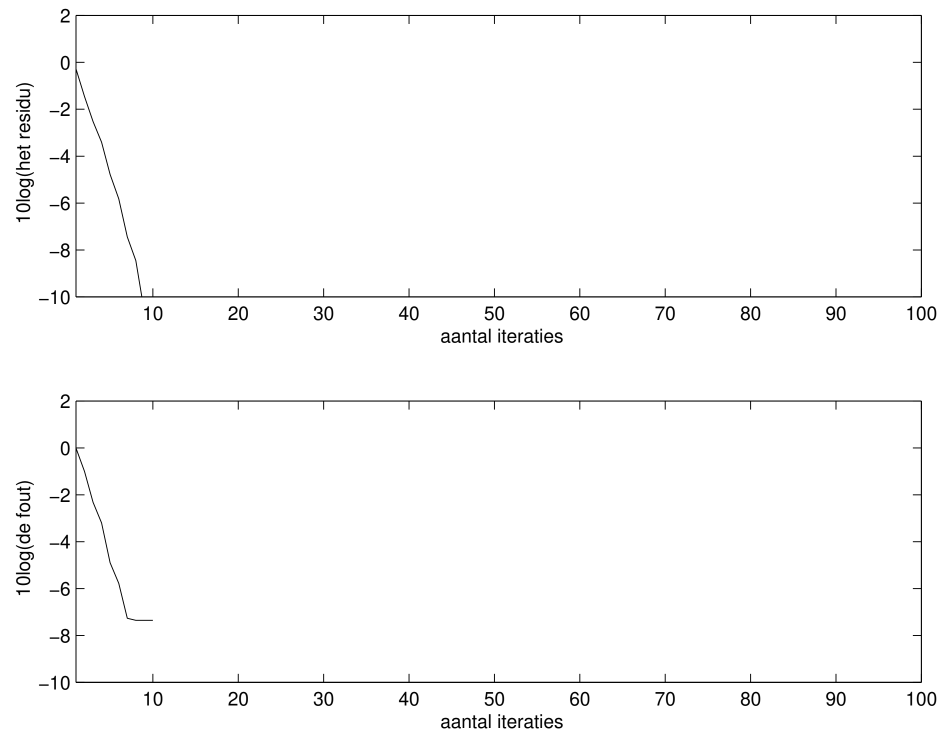
$$\begin{aligned} (\boldsymbol{L}\boldsymbol{L}^T)^{-1}\boldsymbol{A}\boldsymbol{v}_i &= \lambda_i \boldsymbol{v}_i \\ \boldsymbol{L}^{-T}\boldsymbol{L}^{-1}\boldsymbol{A}\boldsymbol{v}_i &= \lambda_i \boldsymbol{v}_i \\ \boldsymbol{L}^{-1}\boldsymbol{A}\boldsymbol{L}^{-T}\boldsymbol{L}^T\boldsymbol{v}_i &= \lambda_i \boldsymbol{L}^T\boldsymbol{v}_i, \end{aligned} \tag{5.1}$$

voor $i = 1, 2$.
Uit de laatste regel van 5.1 concluderen we dat de eigenvektoren van $\boldsymbol{L}^{-1}\boldsymbol{A}\boldsymbol{L}^{-T}$ gelijk zijn aan die van de matrix $(\boldsymbol{L}\boldsymbol{L}^T)^{-1}\boldsymbol{A}$ linksvermenigvuldigd met $\boldsymbol{L}^T$. De struktuur van deze eigenvektoren wordt getoond in figuur 7.1 (zie Appendix 2).

5.1.1 Projectie gebruik makend van de symmetrisch matrix $\boldsymbol{L}^{-1}\boldsymbol{A}\boldsymbol{L}^{-T}$

In hoofdstuk 4 hebben we twee verschillende varianten van de geschaalde matrix $\boldsymbol{A}$ behandeld:
a) $\boldsymbol{L}^{-1}\boldsymbol{A}\boldsymbol{L}^{-T}$ (de symmetrische) en
b) $(\boldsymbol{L}\boldsymbol{L}^T)^{-1}\boldsymbol{A}$ (de niet symmetrische).

In deze paragraaf beschouwen we het eerste geval (a). We gaan dus projecteren op de ruimte opgespannen door de eigenvektoren behorende bij de kleinste eigenwaarden van de matrix $\boldsymbol{L}^{-1}\boldsymbol{A}\boldsymbol{L}^{-T}$. De resultaten die we hebben gekregen na het gebruik van de projectie staan in figuur 5.5. We zien dus dat na projectie het residu strikt dalend blijft totdat hij

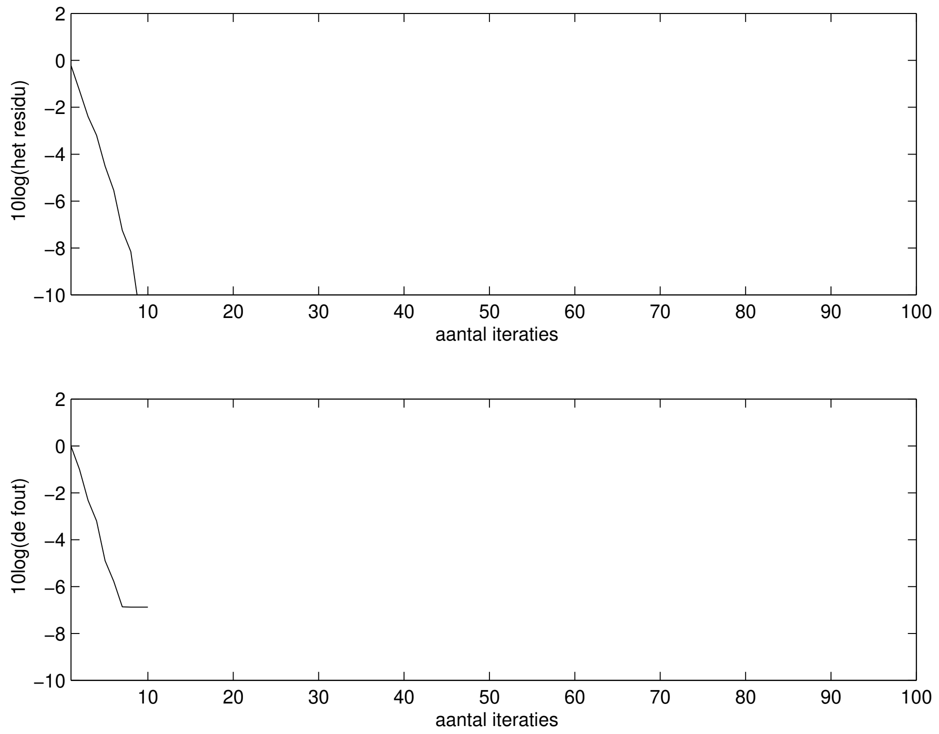


Figuur 5.5: Het residu en de fout behorend bij de symmetrische variant

een minimale waarde ($O(10^{-10})$) heeft bereikt. De fout blijft ook dalend maar zij komt na afloop van het proces niet verder dan een waarde van de orde 10^{-7}. We zien dat de fout aan het einde van de iteratieslag ook zo'n “knik” vertoont als in figuur 5.4. Naast het feit dat het residu en de fout klein worden, zien we ook dat het aantal benodigde iteraties om dit resultaat te bereiken, aanzienlijk verminderd wordt. Vergelijk dit met figuur 5.4 waar geen projectie gebruikt is. In de praktijk is men al tevreden met een fout van de orde 10^{-2}. Het aantal iteraties nodig om deze nauwkeurigheid te bereiken is 18-19 (zie figuur 5.4), zonder projectie en slechts 2-3 iteraties indien we projectie toepassen (figuur 5.5). Verder blijkt dat het residu een betrouwbare graadmeter is voor de fout. Hierna hebben wij ook het niet symmetrisch geval (b) bekeken.

5.1.2 Projectie behorende bij de niet symmetrisch matrix $(\boldsymbol{L}\boldsymbol{L}^T)^{-1}\boldsymbol{A}$

Na het gebruik van de bijbehorende projectie (zie hoofdstuk 4), zijn de in figuur 5.6 geschetste resultaten verkregen. De resultaten zijn vrijwel identiek aan die verkregen in het symmetrische geval (de fout ligt in dit geval tussen 10^{-6} en 10^{-7}). Hij is iets groter dan in

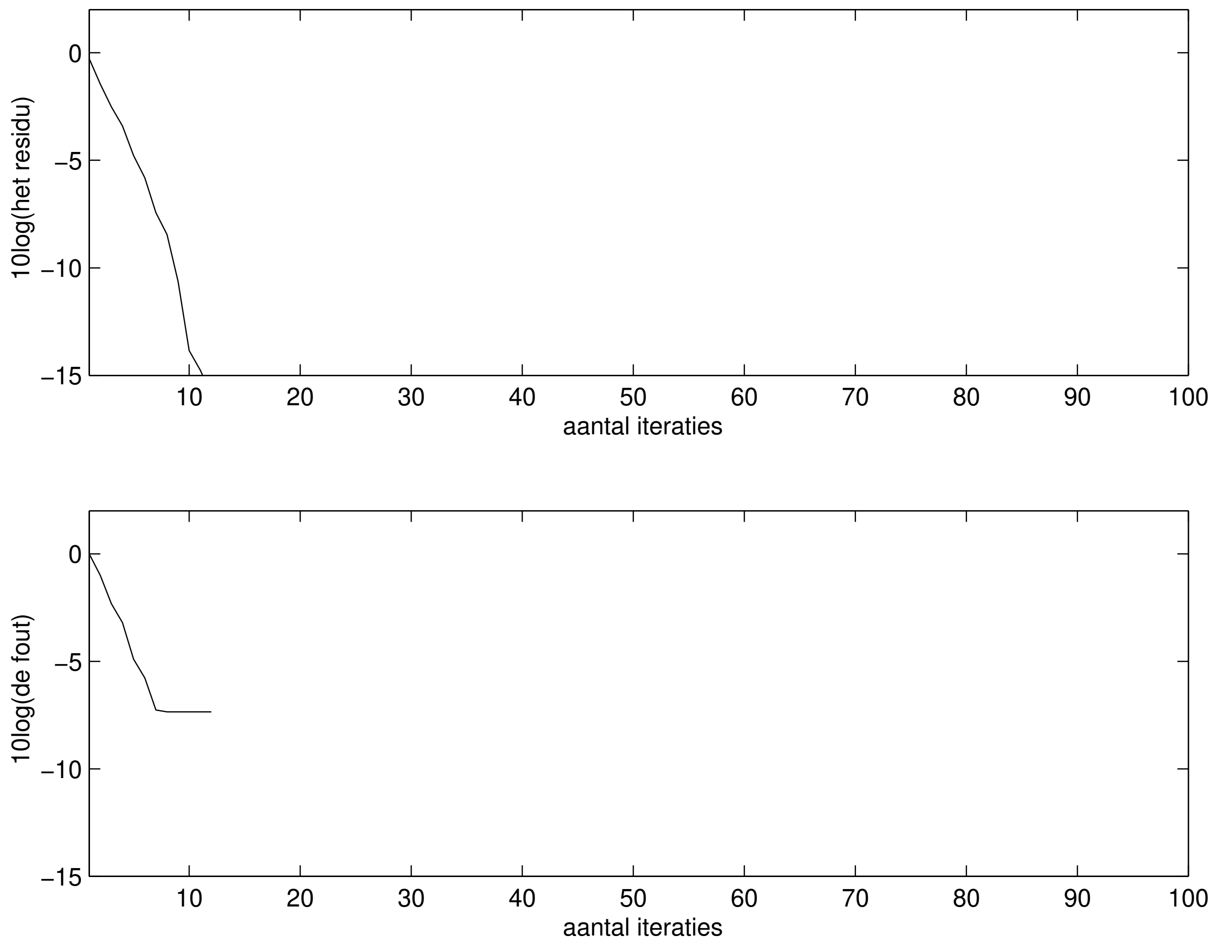


Figuur 5.6: Het residu en de fout behorend bij de niet symmetrische variant

het symmetrische geval. Voor meer inzicht op het probleem, hebben we de nauwkeurigheid in het CG-algoritme verhoogd van 10^{-11} naar 10^{-15}.

5.1.3 Het residu en de fout na verhoging van de nauwkeurigheid

We bekijken in deze paragraaf de beide gevallen (de symmetrische en de niet symmetrische geschaalde matrix $\boldsymbol{A}$). De resultaten die hieruit volgden staan in de figuren 5.7 en 5.8. In deze twee figuren zien we dat het gedrag van het residu niet beïnvloedt wordt door deze verhoging van de nauwkeurigheid. Bij de fout daarentegen is wel een verandering zichtbaar, namelijk in de laatste iteraties heeft de fout een constant gedrag aangenomen maar de orde van grootte is nog steeds heel klein ($\approx 10^{-7}$). De verklaring voor dit verschijnsel is niet echt duidelijk. Mogelijk zorgt de slechte conditie van de matrix $\boldsymbol{A}$ ervoor dat we geen grotere nauwkeurigheid kunnen bereiken. Voor de praktijk zijn we in een dergelijke grote nauwkeurigheid niet geïnteresseerd.
In de volgende paragraaf beschouwen we de situatie waarbij de eigenvektoren worden benaderd door op de fysica gebaseerde schattingen.



Figuur 5.7: Het residu en de fout in het symmetrische geval bij een nauwkeurigheid 10^{-15}

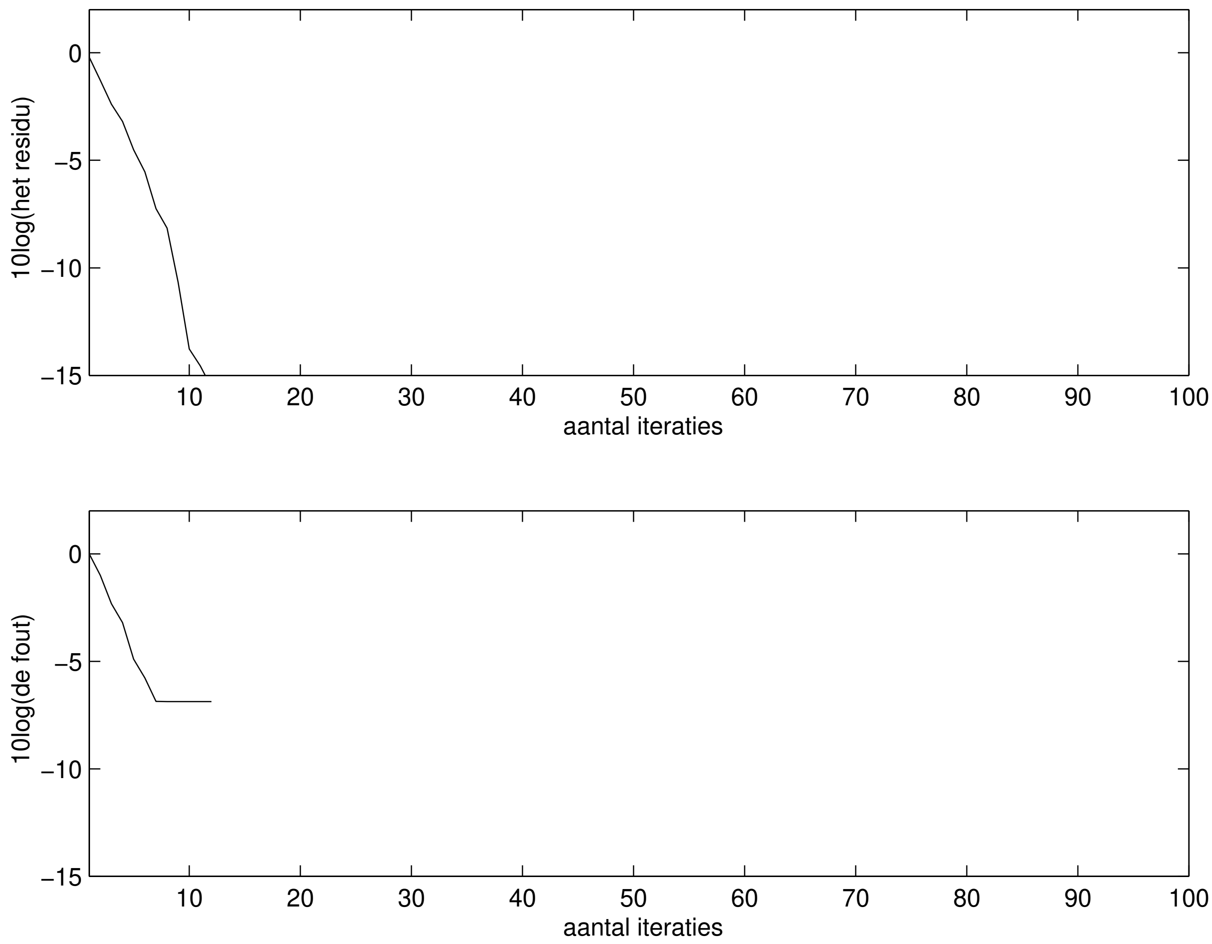
5.2 Projectie op de ruimte opgespannen door de benaderende vektoren $\boldsymbol{w}_1$ en $\boldsymbol{w}_2$

De eigenvektoren die in paragraaf 5.1 gevonden zijn vertonen per kolom het volgende gedrag

- ze zijn constant in een zandlaag
- ze verlopen lineair per leilaag.

Dit geeft ons het idee dat we benaderende eigenvektoren kunnen opbouwen door laagsgewijs te kijken.

- per zandlaag nemen we de vektor constant,
- per leilaag beschouwen we de vektor als de oplossing van een Dirichlet probleem met

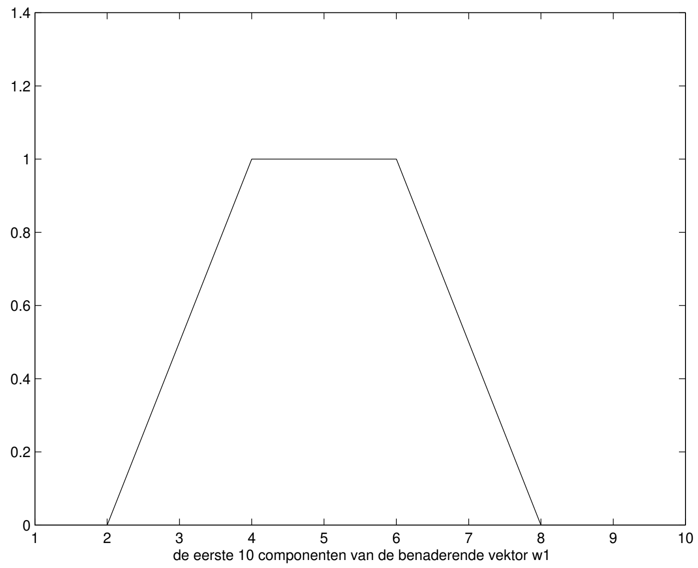


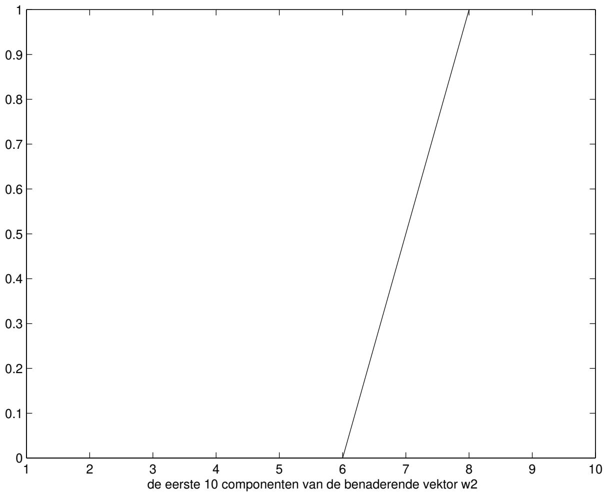
Figuur 5.8: Het residu en de fout in het niet symmetrische geval bij een nauwkeurigheid 10^{-15}

bijvoorbeeld een constante 1 aan de bovenkant en 0 aan de onderkant,
- per eigenvektor kiezen we slechts 2 leilagen waar het Dirichlet probleem wordt opgelost.
Toepassing van bovenstaande leidt tot de benaderende eigenvektoren zoals geschetst in figuur 5.9. Toevallig blijken deze exact de echte eigenruimte, behorende bij de kleine eigenwaarden, op te spannen. Dat is het gevolg van de wel heel eenvoudige configuratie waaraan wij onze experimenten doen.
Het voordeel van de benaderende vektoren t.o.v. de echte is dat ze ijl zijn (ze bestaan uit veel nullen), plus het feit dat we in dit geval de eigenwaarden niet hoeven te kennen (hoofdstuk 4). Dit maakt het werk een stuk gemakkelijker. Gebruiken we de projecties op dezelfde wijze als in het vorige paragraaf, dan krijgen we in het geval van symmetrische preconditionering de resultaten in figuur 5.10. De resultaten geschetst in figuur 5.11 horen bij de niet symmetrische preconditionering van de matrix $\boldsymbol{A}$ (d.w.z $(\boldsymbol{L}\boldsymbol{L}^T)^{-1}\boldsymbol{A}$).
Zoals te verwachten tonen deze figuren dezelfde resultaten als in de figuren 5.5 en 5.6, dat

wil zeggen het residu bereikt zijn minimum waarde met behulp van de projectie na weinig iteraties en ook de fout wordt klein na afloop van het iteratie-proces.

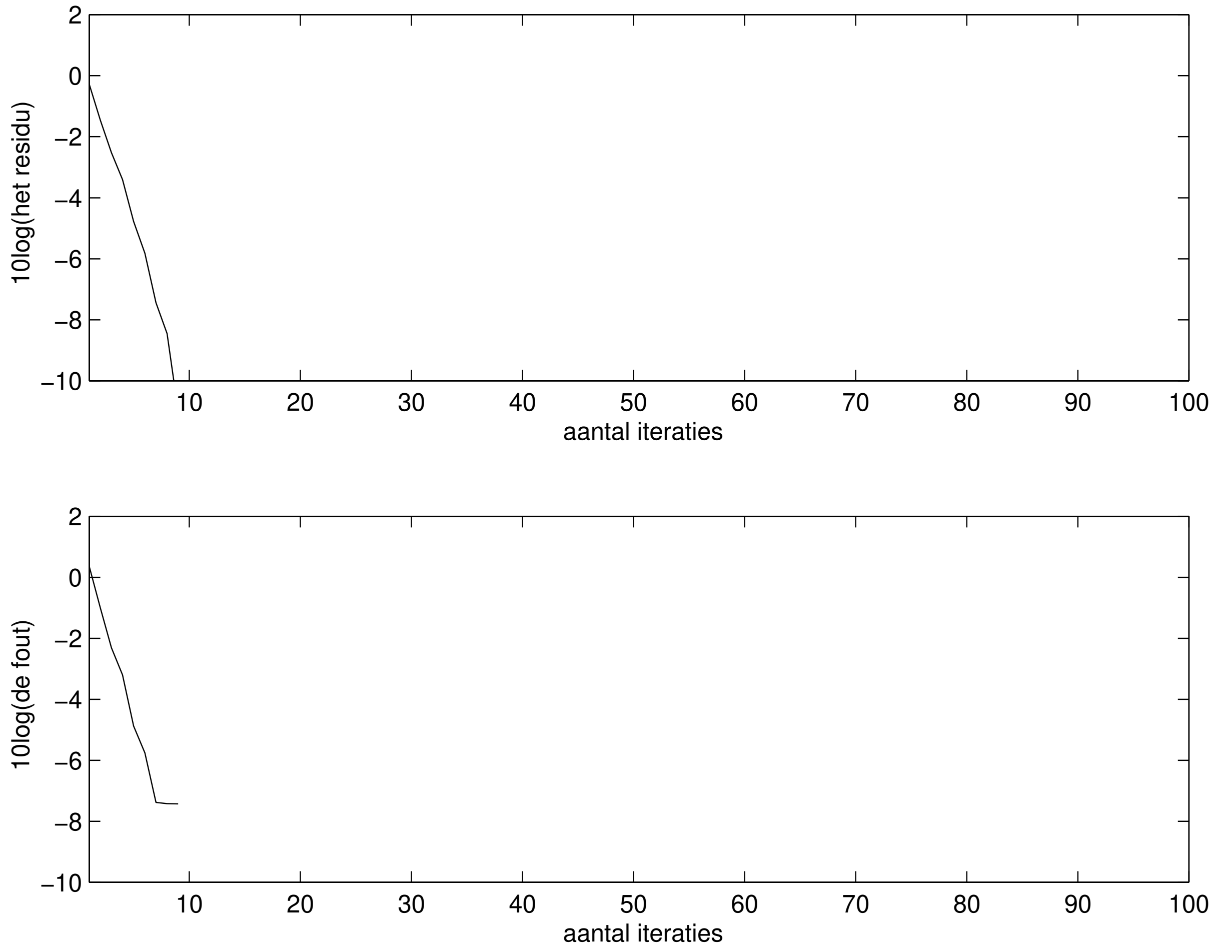
Opmerking:
De resultaten die we hebben gekregen met ILU preconditionering gelden ook in het geval van diagonaal schaling. De benaderende vektoren in figuur 5.9 zijn gebaseerd op





Figuur 5.9: De nieuwe basisvektoren $\boldsymbol{w}_1$ en $\boldsymbol{w}_2$ van de geprojecteerde ruimte

de struktuur van de eigenvektoren behorend bij de niet symmetrisch geschaalde matrix $(\boldsymbol{L}\boldsymbol{L}^T)^{-1}\boldsymbol{A}$. In dit geval moeten we de vektoren $\boldsymbol{w}_i$ zelf en de vektoren $\boldsymbol{A}\boldsymbol{w}_i$, voor $i = 1, 2$,

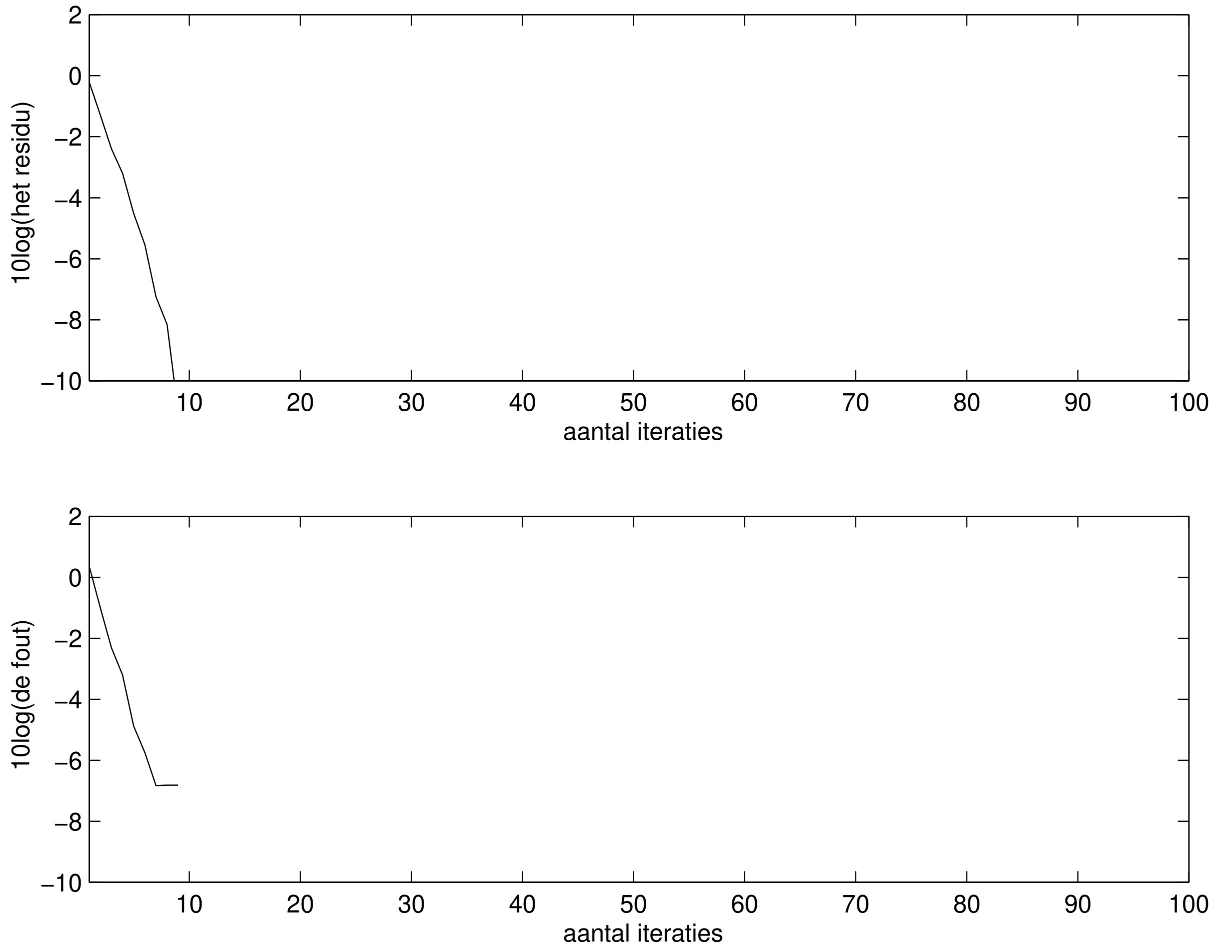


Figuur 5.10: Het residu en de fout na projectie op de basisvektoren $\boldsymbol{w}_1$ en $\boldsymbol{w}_2$ in het symmetrische geval

opslaan. De struktuur van de vektoren $\boldsymbol{A}\boldsymbol{w}_i$ staat in figuur 5.12. Merk op dat de vektoren $\boldsymbol{A}\boldsymbol{w}_i$ minder ijl zijn dan de vektoren $\boldsymbol{w}_i$, voor $i = 1, 2$. Echter de vektoren $\boldsymbol{A}\boldsymbol{w}_i$ lopen ook niet vol.

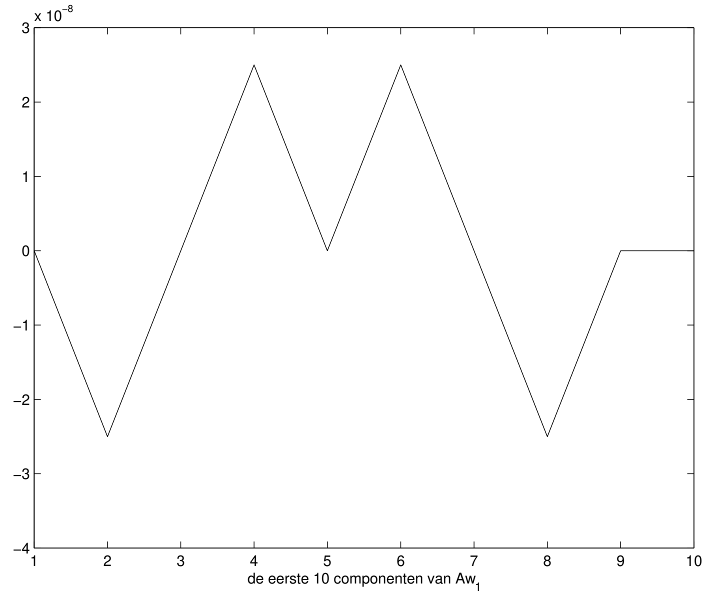
Beschouwen we nu de symmetrisch geschaalde matrix ($\boldsymbol{L}^{-1}\boldsymbol{A}\boldsymbol{L}^{-T}$), dan voldoen de benaderende vektoren ($\boldsymbol{L}^T\boldsymbol{w}_i$, $i = 1, 2$) aan de in figuur 7.2 gegeven strukturen (zie Appendix 2). We constateren dat de vektoren $\boldsymbol{L}^T\boldsymbol{w}_i$ ook ijl zijn in vergelijking met de vektoren $\boldsymbol{w}_i$. Het is nu bovendien al bekend dat in het symmetrische geval zowel de vektoren $\boldsymbol{L}^T\boldsymbol{w}_i$ als $\boldsymbol{L}^{-1}\boldsymbol{A}\boldsymbol{L}^{-T}(\boldsymbol{L}^T\boldsymbol{w}_i)$ (hoofdstuk 4) opgeslagen moeten worden. Daarom is het van belang om de struktuur van de vektoren $\boldsymbol{L}^{-1}\boldsymbol{A}\boldsymbol{L}^{-T}(\boldsymbol{L}^T\boldsymbol{w}_i)$ te weten. Voor de struktuur van deze vektoren, zie figuur 7.3 in Appendix 2. Het is duidelijk in te zien dat de vektoren $\boldsymbol{L}^{-1}\boldsymbol{A}\boldsymbol{L}^{-T}(\boldsymbol{L}^T\boldsymbol{w}_i)$ kostbaarder zijn dan de vektoren $\boldsymbol{L}^T\boldsymbol{w}_i$ zelf.

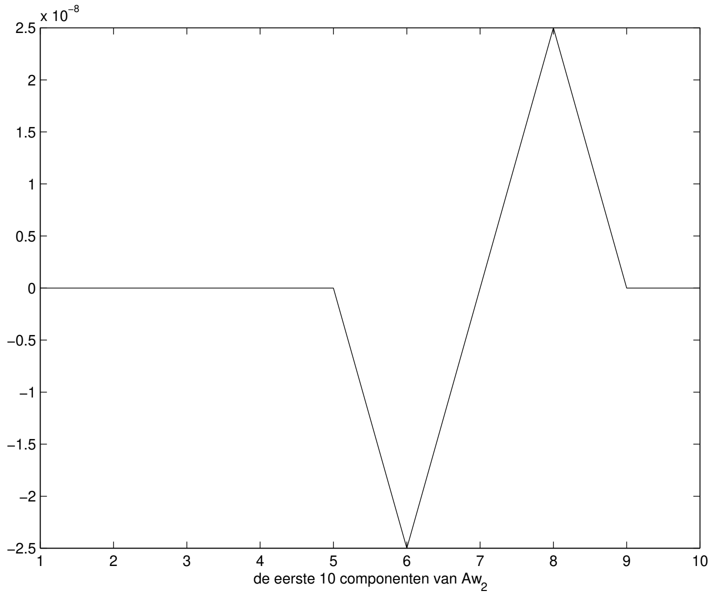
In hoofdstuk 6 zullen we een soortgelijk onderzoek (experimenten) doen, maar dan met behulp van het Sepran-pakket. Hierbij zullen we uittesten of de projecties die in hoofdstuk



Figuur 5.11: Het residu en de fout na projectie in het niet symmetrische geval

4 zijn gedefinëerd ook goede resultaten kunnen opleveren als de struktuur van de lagen van het beschouwde gebied gemodificeerd zijn (kromme lagen en/of lagen met variabele hoogte).





Figuur 5.12: De struktuur van de vektoren $\boldsymbol{Aw}_1$ en $\boldsymbol{Aw}_2$

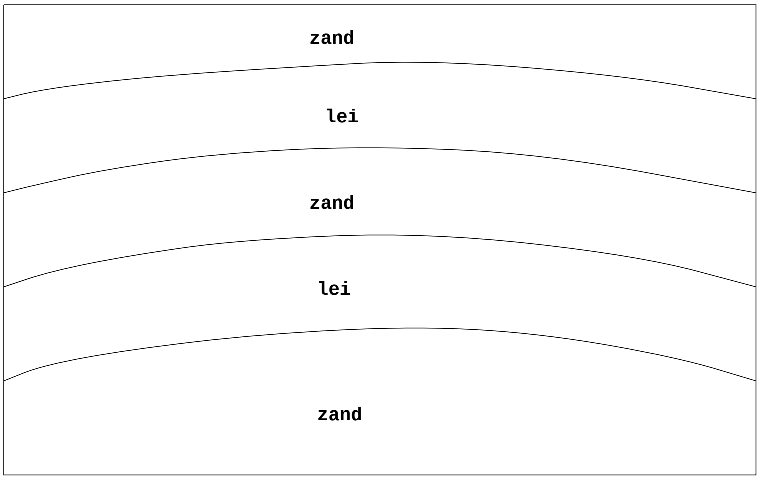
Hoofdstuk 6

SEPRAN experimenten

6.1 Inleiding

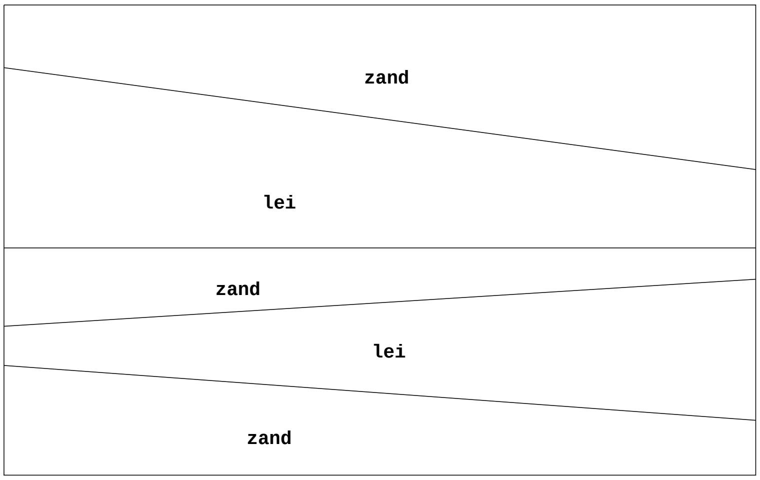
Met behulp van het Sepran programma zullen we testen of de prachtige resultaten die wij in hoofdstuk 5 gekregen hebben, voor gebieden met rechte lagen, ook voor een willekeurige lagenstruktuur (zoals kromme lagen en lagen met een variërende dikte) gelden. We gaan er hierbij vanuit dat de eigenvektoren behorende bij de kleinste eigenwaarden van de matrix, de struktuur behouden die wij in het vorige hoofdstuk geïntroduceerd hebben:
a) konstante in zand en
b) lineair in lei.
Vandaar dat we bij de Sepran-experimenten slechts met de benaderende vektoren zullen werken. De struktuur van deze benaderende vektoren is gedetaileerd besproken in hoofdstuk 5.
Bij de experimenten zullen we ons beperken tot 4 verschillende gebieden:

- een krom lagengebied met 2 leilagen (dus 5 lagen in totaal), zie figuur 6.1



Figuur 6.1: Een gebied met 5 kromme lagen

- een gebied van 5 lagen met variabele dikte, figuur 6.2



Figuur 6.2: Een gebied van 5 lagen met variabel hoogte

- een krom lagengebied met 3 leilagen (7 lagen in totaal), figuur 6.3

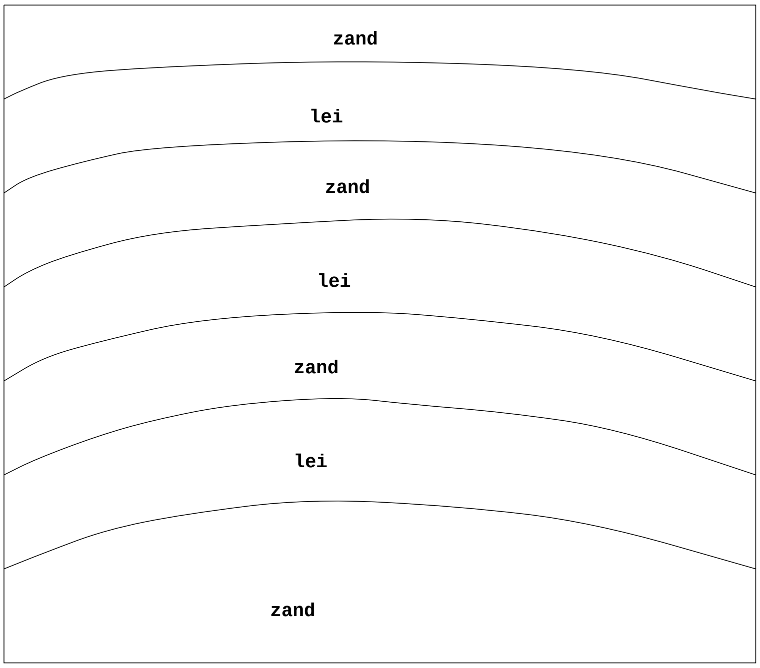
- een 7 lagengebied met variabele hoogten (dikte) figuur 6.4. In de volgende paragraaf zullen we de experimenten en bijbehorende resultaten voor ieder van de hierboven weergegeven gebieden bespreken.

6.2 Bespreking van de resultaten voor een gebied met 5 lagen

6.2.1 Resultaten behorende bij de kromme lagen

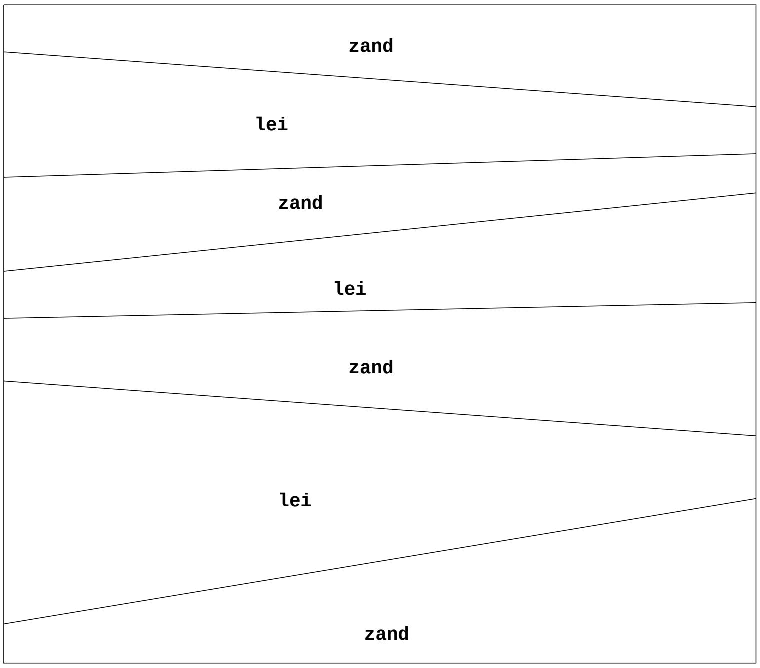
Het gebied dat we hierbij beschouwen is in figuur 6.1 aangegeven. De roosterpunten worden genummerd van boven naar beneden (verticale nummering). Voor de nauwkeurigheid ϵ die wij in het PCG-algoritme (zie hoofdstuk 4) zullen gebruiken kiezen we 10^{-12}. In onze experimenten beperken we ons tot de IC (incomplete Choleski) preconditionering.
Op dezelfde wijze als in het vorige hoofdstuk, bekijken we hier ook twee aspekten van het probleem:

1) resultaten volgend uit de experimenten zonder toepassing van projecties,
2) de resultaten na het gebruik maken van projecties.
De resultaten die bij geval 1 horen zijn in figuur 6.5 gegeven. Dit resultaat is vergelijkbaar met wat we al eerder in hoofdstuk 5 (Matlab experimenten) hebben gezien (zie figuur 5.4). het residu toont ook hier twee bulltjes (of sprongen) aan. Deze twee sprongen geven aan dat er twee kleine eigenwaarden gevonden zijn. Echter in het begin van het iteratie-proces daalt het residu heel snel en pas als de twee kleine eigenwaarden gevonden zijn, neemt het



Figuur 6.3: Een gebied van 7 kromme lagen

residu zijn minimaal waarde aan ($\approx 10^{-12}$). Dit verschijnsel in het residu vertaalt zich in de fout op een zodanig manier dat deze constant blijft (≈ 1) gedurende de eerste 18 iteraties (d.w.z voordat de laatste kleine eigenwaarde gevonden wordt), daarna neemt de fout sterk af tot een waarde van de orde 10^{-8}. Vanwege de slechte conditie van de matrix (of beter gezegd van de gepreconditioneerde matrix), blijft de fout deze waarde houden ook als we de nauwkeurigheid verhogen.
Om dit rare gedrag in zowel het residu als de fout weg te nemen, hebben we de techniek van projecties toegepast. De projecties waarvan we gebruik zullen maken zijn gebaseerd op de zogenaamd benaderende vektoren. Deze benaderende vektoren hebben het gedrag van de "echte" eigenvektoren, behorende bij de kleinste eigenwaarden, van de matrix. Dat wil zeggen ze zijn constant in de zandlagen en zij verlopen lineair in de leilagen (hoofdstuk 4 en 5). We gebruiken bovendien de projectie die bij het niet symmetrische variant hoort (hoofdstuk 4). De resultaten die met behulp van de projectie (geval 2 dus) zijn verkregen, staan in figuur 6.6.

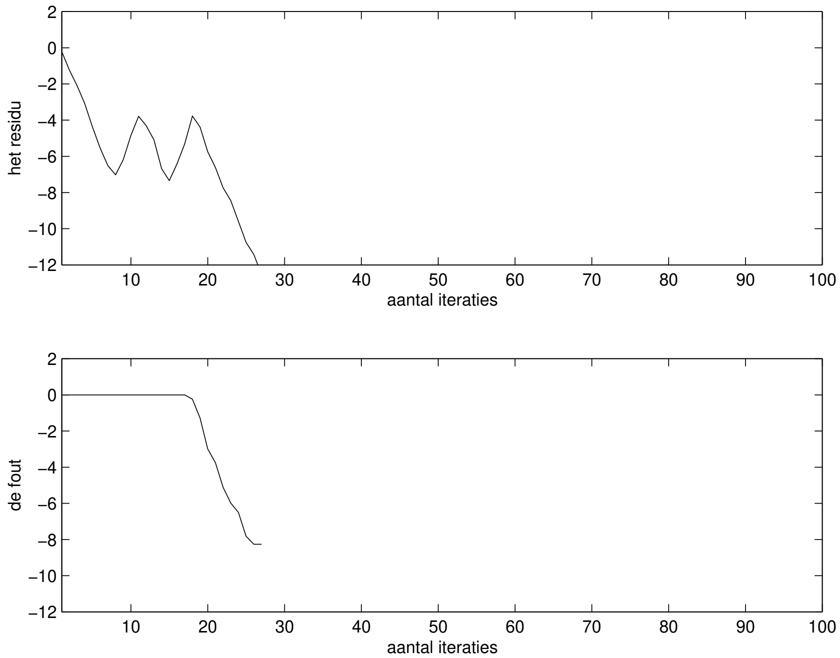


Figuur 6.4: Een gebied van 7 lagen met variabel hoogte

In dit geval wordt dus in het PCG-proces een combinatie van IC en de projectie-matrix gebruikt. Uit figuur 6.6 is nu duidelijk te zien dat zowel het residu als de fout een hele kleine waarde, na afloop van het iteratie-proces, hebben aangenomen ($\approx 10^{-12}$ voor het residu respectievelijk $O(10^{-8})$ voor de fout). Dit resultaat komt dus precies overeen met datgene wat we met het Matlab programma hebben gekregen (zie hoofdstuk 5).
Dezelfde experimenten hebben we voor een gebied dat uit lagen met variabele diktes bestaat (figuur 6.2) herhaald. In de volgende paragraaf geven we de resultaten die hierbij horen weer.

6.2.2 Resultaten behorende bij de lagen met variabele hoogten

We geven eerst de resultaten die bij PCG, zonder projectie, behoren (zie figuur 6.7). De resultaten in dit geval lijken sprekend op die behorende bij een krom lagengebied, het residu vertoont twee sprongen en de fout blijft gedurende de eerste iteraties op $O(1)$ hangen. Aan het einde neemt zowel het residu als de fout een zeer kleine waarde aan (resp. $\approx 10^{-12}$ en 10^{-8}). Dit gedrag in het residu en de fout zullen we ook proberen kwijt te raken door een geschikte projectie toe te passen. De resultaten die hieruit worden verkregen zijn in figuur 6.8 geïllustreerd. We zien duidelijk dat ook in dit geval het residu en de fout keurig naar hun minimale waarden ($O(10^{-12})$ resp. $O(10^{-8})$) naderen. Vervolgens hebben wij deze experimenten voor grotere gebieden uitgevoerd. We bekijken hier alleen het gebied dat in totaal uit 7 lagen (4 zand- en 3 leilagen) bestaat.



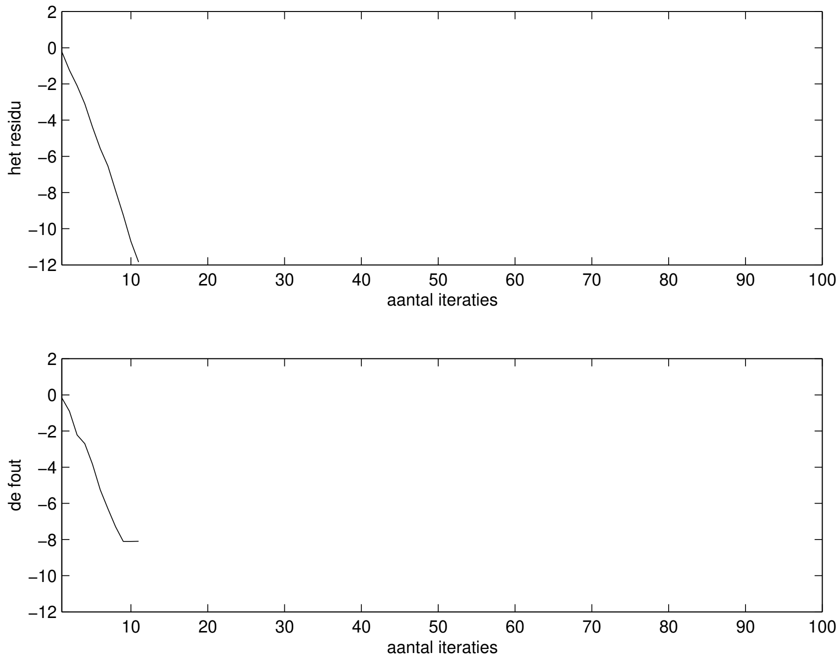
Figuur 6.5: het residu en de fout behorend bij een 5 kromme lagengebied vóór de projectie

6.3 Resultaten behorende bij een 7 lagengebied

Op dezelfde wijze als in voorgaand paragraaf, bekijken we eerst een gebied met kromme lagen figuur 6.3.

6.3.1 Kromme lagen

Indien we geen projectie in het PCG-proces invoeren, dan krijgen we de resultaten geschetst in figuur 6.9. Een opmerkelijk verschil dat we aan figuur 6.9 zien is dat het aantal uitgevoerde iteraties groot is in vergelijking met een 5 lagen gebied. Daarnaast zijn er ook 3 sprongen zichtbaar, deze corresponderen met het vinden van de kleine eigenwaarden van de matrix. Ter herinnering: iedere sprong die we hier zien duidt op een lei-zand overgang (hoofdstuk 2). De fout blijft in eerst instantie konstant totdat de derde en laatste kleine eigenwaarde gevonden is, op dat moment daalt de fout aanzienlijk. De waarden die het residu en de fout uiteindelijk hebben bereikt zijn identiek aan de waarden die voor het 5 lagen gebied gelden ($O(10^{-12})$ voor het residu en $O(10^{-8})$ voor de fout). De fout blijft vervolgens constant op $O(10^{-8})$. Een mogelijke oorzaak hiervan is waarschijnlijk de slechte conditie van de matrix. Hierna hebben wij het geval met projectie bekeken, en de resultaten die daaruit volgen zijn in figuur 6.10 geschetst. Allereerst zien we dat na toepassing van de bijbehorende projectie, het aantal uitgevoerd iteraties aanzienlijk verminderd is ten opzicht van het geval zonder projectie. Het residu neemt geleidelijk af naar zijn minimaal waarde ($\approx 10^{-12}$). De fout daalt ook heel snel tot $O(10^{-8})$ waar hij vervolgens een constant

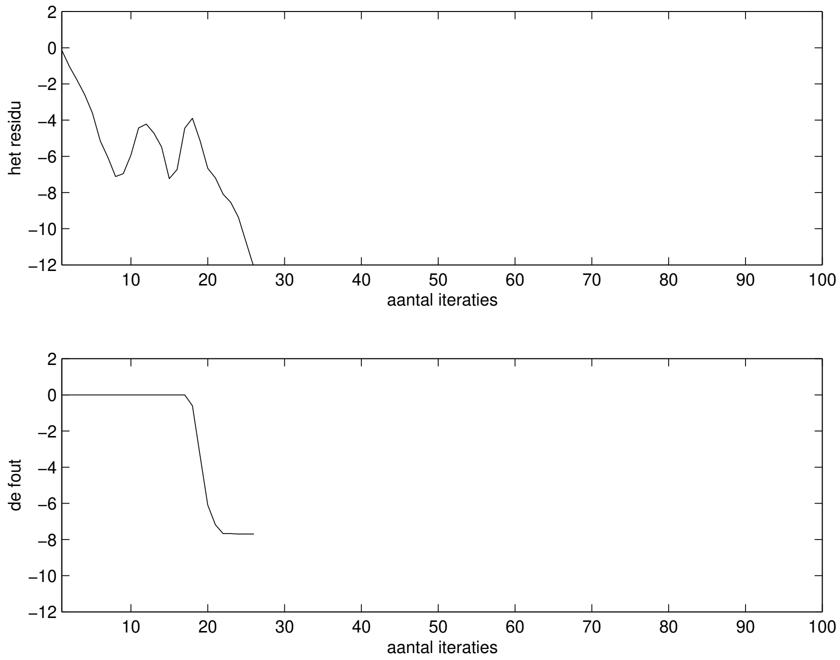


Figuur 6.6: het residu en de fout behorend bij een 5 krom lagengebied na projectie

gedrag aanneemt.
Als laatste hebben wij voor het gebied van 7 lagen waarvan de dikte (of de hoogte) variabel is (zie figuur 6.4), de oplossing bepaald.

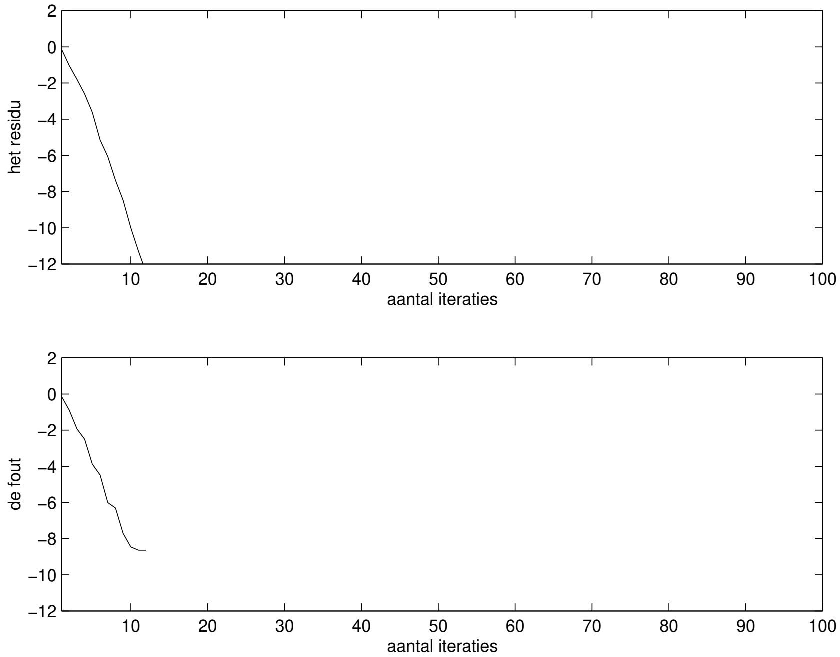
6.3.2 Gebied van 7 lagen met variabele hoogten

Voor het geval dat we geen projectie toepassen, krijgen we de resultaten in figuur 6.11. Ook hier geldt dat het aantal iteraties nodig om een nauwkeurigheid van 10^{-12} te bereiken groot is (ongeveer 75 iteraties). Het residu vertoont hierbij drie sprongen, dus er zijn drie kleine eigenwaarden ontdekt. De fout blijft in eerste instantie groot en voordat de laatste kleine eigenwaarde van de matrix ondekt wordt, konstant. De fout blijft ook na afloop konstant op de waarde 10^{-8}. De bijdrage van deze kleine eigenwaarden speelt een direkte rol bij dit constant gedrag in de fout, door de conditie van de matrix te verslechteren. Met behulp van projectie kunnen we dit gedrag in het residu en de fout kwijt raken. De resultaten die we krijgen na projectie zijn in figuur 6.12 geplot. Het effect van de projectie is duidelijk aan figuur 6.12 te zien. Echter het residu toont een strikt monotoon dalend gedrag, ook de fout tijdens de eerste 11 à 12 iteraties is strikt dalend gebleven. Daarna neemt hij een konstante gedrag aan bij zijn minimaal waarde $O(10^{-8})$. Het aantal iteraties wordt in dit geval ook aanzienlijk minder (ongeveer 19 iteraties).
Teneinde de resultaten van Matlab en SEPRAN met elkaar te vergelijken, hebben we nog een keer het gebied bestaand uit 5 rechte lagen beschouwd. Daarin hebben we hetzelfde aantal punten genomen over het gebied en natuurlijk ook dezelfde nummering gebruikt.

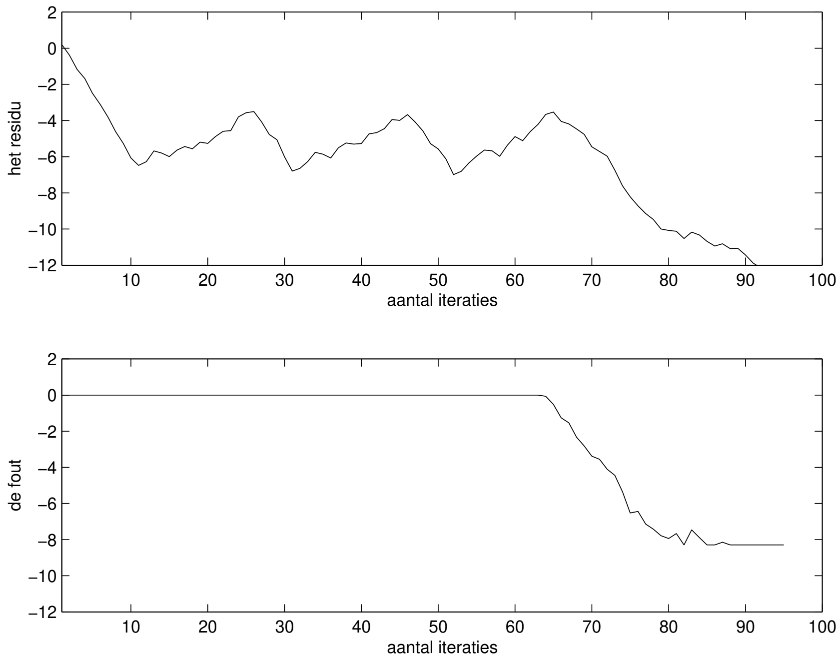


Figuur 6.7: het residu en de fout behorend bij een 5 lagengebied met variabele diktes, vóór de projectie

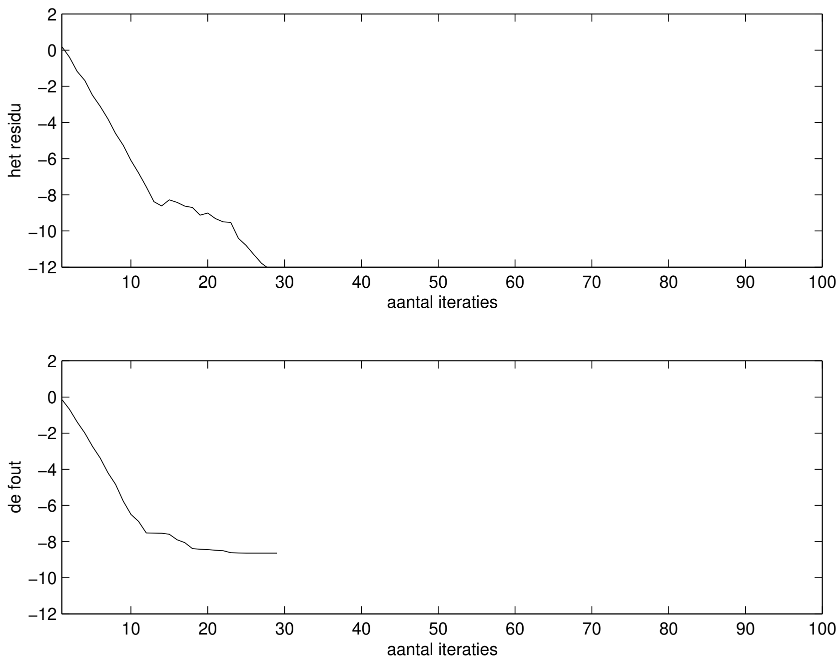
Voor de nauwkeurigheid kiezen we, net als bij figuur 5.8 in hoofdstuk 5, $\epsilon = 10^{-15}$. Het resultaat dat wij hieruit hebben gekregen is in figuur 6.13 gegeven. We zien dus dat het residu en de fout in figuur 6.13 zich precies als in figuur 5.8 gedragen.



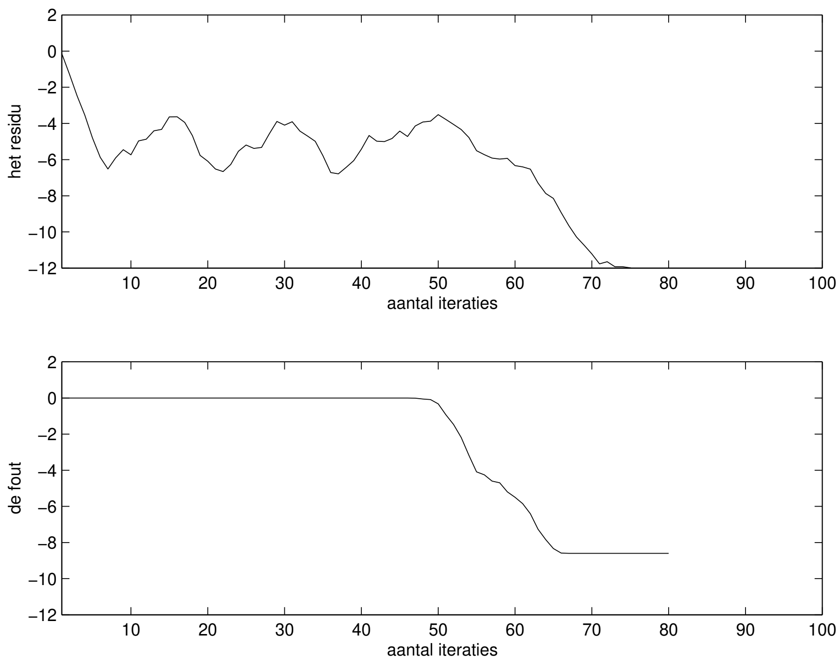
Figuur 6.8: het residu en de fout behorend bij een 5 lagengebied met variabele diktes, na projectie



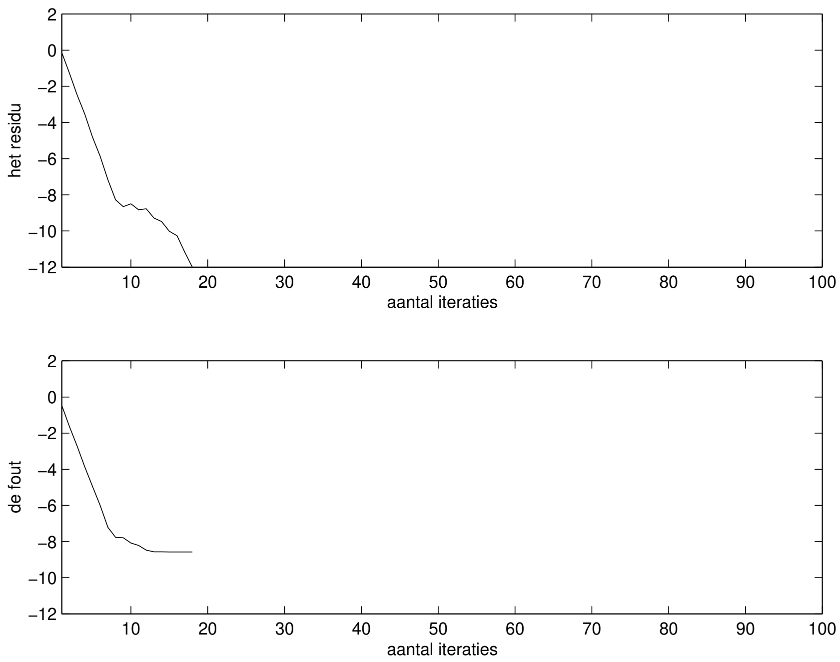
Figuur 6.9: het residu en de fout vóór projectie voor een 7 kromme lagengebied



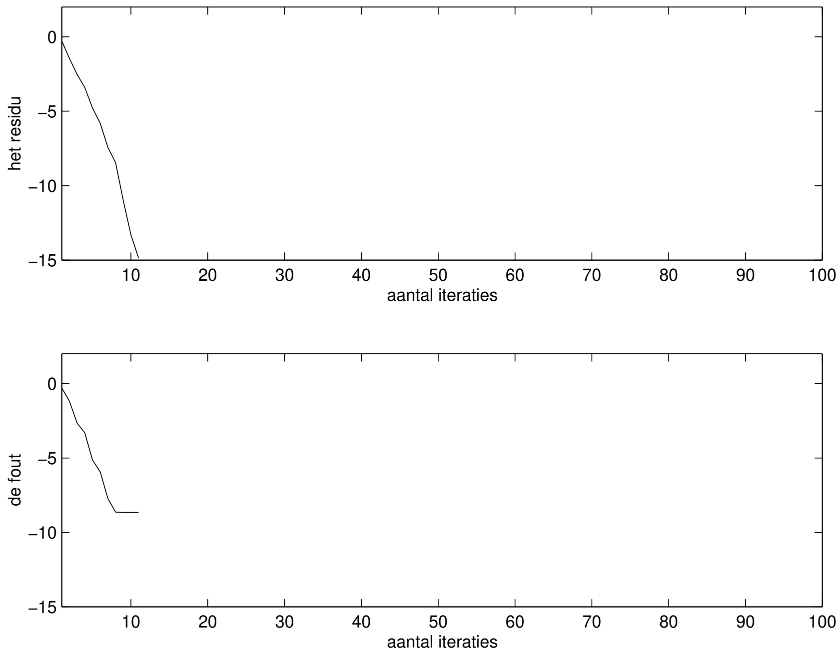
Figuur 6.10: het residu en de fout na projectie voor een 7 kromme lagengebied



Figuur 6.11: het residu en de fout vóór projectie voor een 7 lagengebied



Figuur 6.12: het residu en de fout na projectie voor een 7 lagengebied



Figuur 6.13: het residu en de fout bij een nauwkeurigheid 10^{-15}

Hoofdstuk 7

Conclusies en aanbevelingen voor verder ondrzoek

7.1 Conclusies

In dit afstudeerwerk is onderzoek gedaan naar het iteratief oplossen van stelsels lineaire vergelijkingen zoals die verkregen worden uit de discretisatie van de drukvergelijking in een sterk gelaagde bodem. Doordat de permeabiliteiten van de verschillende bodemlagen vele grootte ordes verschillen, blijkt de matrix zeer slecht geconditioneerd te zijn. Gevolg daarvan is dat zelfs een gepreconditioneerde CG-methode zeer slecht convergeert.
De eerste opvallende conclusie uit dit onderzoek was dat het residu gedurende de eerste iteraties een normale tendens vertoonde. Hierdoor wordt de schijn van nauwkeurigheid gewekt, temeer daar de schatting van de kleinste eigenwaarde een veel te grote waarde gaf. Analyse van de fout toonde echter aan dat de nauwkeurigheid totaal niet toenam. Verder heeft het analyse aangetoond dat de kleinste eigenwaarden van de matrix een groot rol spelen in dit probleem. In het gepreconditioneerde geval kwam het aantal kleine eigenwaarden precies overeen met het aantal overgangen van laag permeabel naar hoog permeabel lagen. Pas als alle kleine eigenwaarden "ontdekt" zijn door het iteratieproces, wordt het resultaat betrouwbaar.
Op grond van fysische argumenten is de struktuur van de eigenvektoren behorend bij deze kleine eigenwaarden gevonden. Deze struktuur bleek relatief simpel te zijn. Met behulp van schattingen van eigenvektoren behorend bij die kleine eigenwaarden is een projectietechniek ontwikkeld welke de slechte eigenschappen van het oorspronkelijk iteratieproces te niet doet, omdat met deze techniek worden de bijdrage van die eigenvektoren helemaal geanullerde.
Hierdoor is een snelle, betrouwbare oplosmethode voor deze specifieke matrices ontwikkeld. In eerste instantie zijn een aantal experimenten gedaan met behulp van eenvoudige Matlab-programma's. Daarna zijn we met een SEPRAN programma wat complexer gevallen gaan analyseren en verifieren. De algemene conclusie uit deze analyse is dat de door ons voorgestelde projectietechniek volledig aan zijn doel beantwoord:

- het aantal uitgevoerd iteraties reduceert zich aanzienlijk en

- het residu bleek een goede graadmeter voor de fout te zijn.

7.2 Aanbevelingen en suggesties voor een verder onderzoek

De volgende punten dienen in eerste instantie nader onderzocht te worden:

- hoe nauwkeurig moeten we de projectie vektoren bepalen,

- hoe belangrijk is de struktuur van de projectie vektor mat name in de laag permeabele lagen,

- kan door gebruik te maken van de eenvoudige struktuur van de projectie vektoren de projectie efficienter worden opgezet qua geheugen gebruik en rekentijd.

Tenslotte kan onderzocht worden of de methode eenvoudig uitgebreid kan worden naar praktische 3D problemen. Tevens kan nagegaan worden of het toevoegen van tijdsafhankelijkheid nog enige invloed heeft op het probleem als geheel.

Appendix 1

Het programma die wij in de Matlab-experimenten hebben gebruikt bestaat uit 4 files:

- de file Matrix-vektor: hier worden de matrix $\boldsymbol{A}$, de preconditioneringsmatrices ($(LL^T)^{-1}$ en de diagonaal schaling $\boldsymbol{D}$) en het rechterlid $\boldsymbol{b}$ geconstrueerd,

- Projectie_2: deze file is gebruikt in het geval van projectie op 2 basisvektoren (de "echte" $\boldsymbol{v}_1$ en $\boldsymbol{v}_2$ of de benaderende $\boldsymbol{w}_1$ en $\boldsymbol{w}_2$, voor een 5 lagengebied),

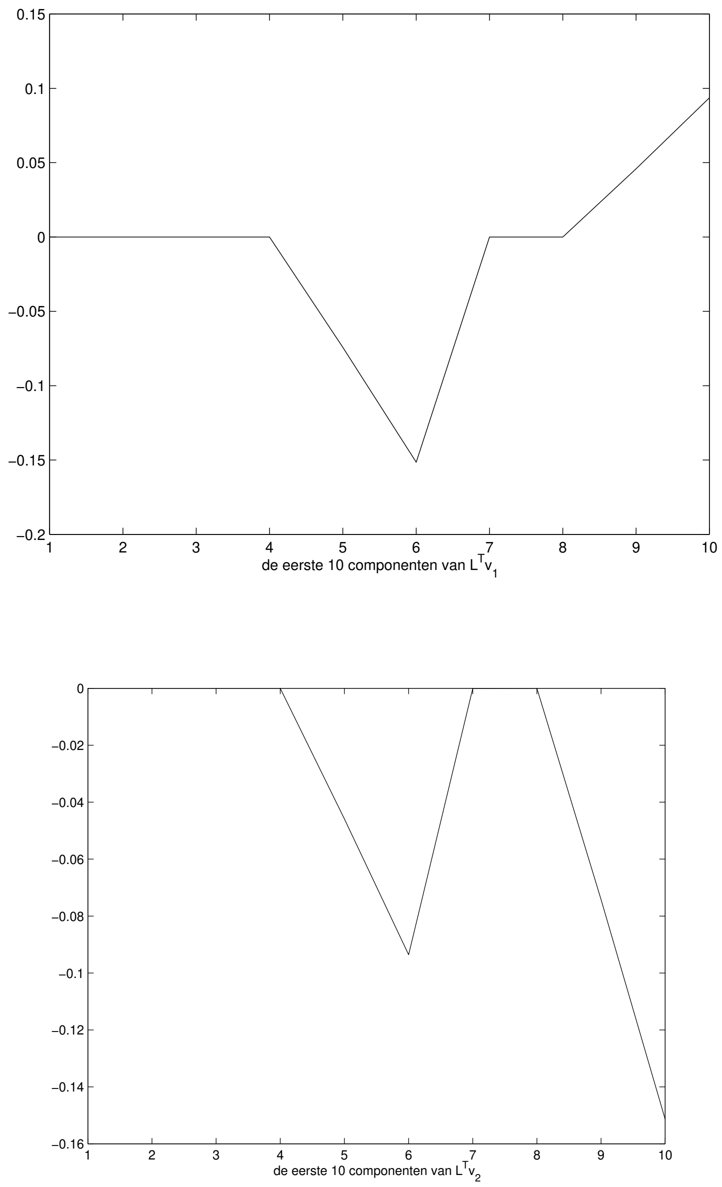
- de file PCG-algoritme: in deze file wordt de gepreconditioneerde CG-algoritme geïmplementeerd en

- Programma: met deze file wordt het volledige programma gedraaid.

Appendix 2

A2.1 De struktuur van de eigenvektoren van $\boldsymbol{L}^{-1}\boldsymbol{A}\boldsymbol{L}^{-T}$

Let op: om de struktuur van de eigenvektoren in figuur 7.1 goed te kunnen volgen, hebben



Figuur 7.1: De struktuur van de eigenvektoren van de matrix $\boldsymbol{L}^{-1}\boldsymbol{A}\boldsymbol{L}^{-T}$

we de roosterpunten in de zandlaag en die in de leilaag, met de bijbehorende punten aan de randen, op de volgende wijze gekozen (zie figuur 5.1):
[1 2]: correspondeert met de bovenste zandlaag

[2 4]: geldt in de bovenste leilaag
[4 6]: de tweede zandlaag
[6 8]: de tweede leilaag en
[8 10] is de onderste zandlaag.
De eerste 10 coefficiënten van de vektor $\boldsymbol{L}^T \boldsymbol{v}_1$ zijn gegeven door

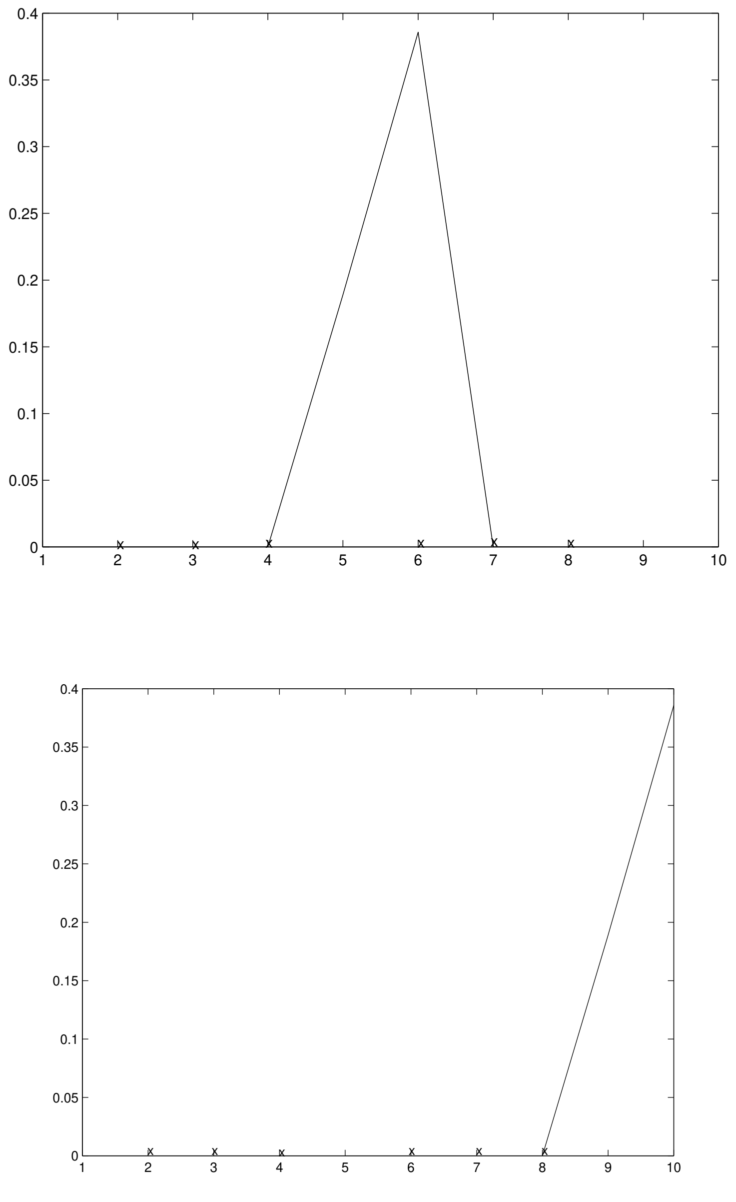
0.00000000000000
-0.00000000524524
-0.00000000000313
-0.00000000490648
-0.07417893069098
-0.15141709676342
-0.00004388484343
-0.00000000678058
0.04584509682965
0.09358092252897

we zien dus dat de coefficiënten in de eerste leilaag ([2 4]) niet echt 0 zijn. Hetzelfde geldt op het interval [7 8].
Dezelfde opmerking geldt ook voor de vektor $\boldsymbol{L}^T \boldsymbol{v}_2$, waarvan de eerste 10 componenten worden gegeven door

0.00000000000000
-0.00000000324174
-0.00000000000194
-0.00000000303237
-0.04584509805855
-0.09358090906419
-0.00002712232517
-0.00000001097121
-0.07417892923496
-0.15141709772472

A2.2 De struktuur van de vektoren $\boldsymbol{L}^T\boldsymbol{w}_1$ en $\boldsymbol{L}^T\boldsymbol{w}_2$

de punten die in de leilagen gekozen zijn, worden in figuur 7.2 weergegeven met $\times$. de



Figuur 7.2: De struktuur van de vektoren $\boldsymbol{L}^T\boldsymbol{w}_1$ en $\boldsymbol{L}^T\boldsymbol{w}_2$

eerste 10 componenten van de vektor $\boldsymbol{L}^T \boldsymbol{w}_1$ zijn:

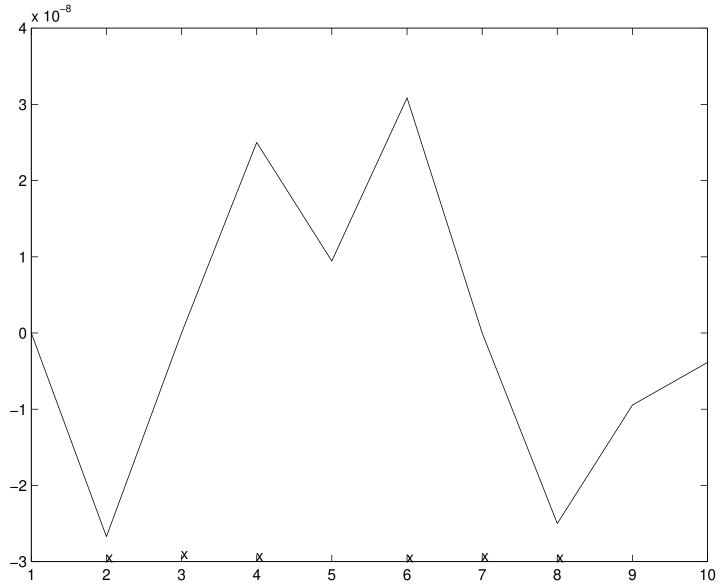
0.00000000000000
-0.00000002672612
-0.00000000000319
0.00000003750000
0.18898225185942
0.38575838093270
0.00011180339643
0
0
0

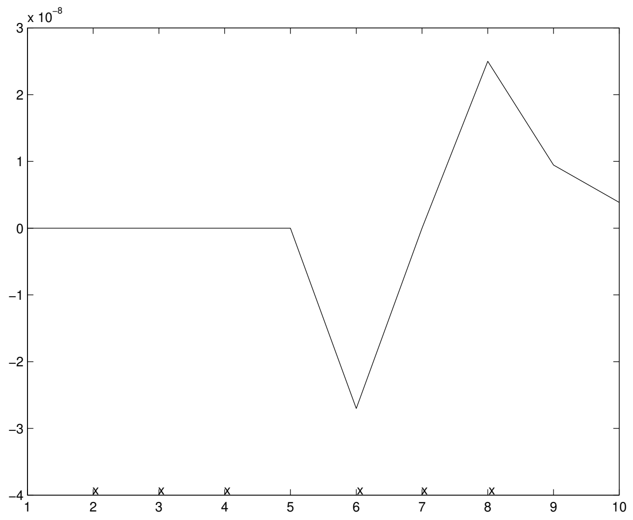
de 10 eerste componenten van $\boldsymbol{L}^T \boldsymbol{w}_2$ zijn

0.00000000000000
0.00000000000000
0.00000000000000
0.00000000000000
0.00000000000000
-0.00000002700308
-0.00000000000326
0.00000003750000
0.18898225185942
0.38575837643219

A2.3 De struktuur van de vektoren $\hat{\boldsymbol{A}}\boldsymbol{L}^T\boldsymbol{w}_i$

De eerste 10 componenten van ieder van de vektoren $\boldsymbol{L}^{-1}\boldsymbol{A}\boldsymbol{L}^{-T}(\boldsymbol{L}^T\boldsymbol{w}_i)$ $(i = 1, 2)$





Figuur 7.3: De struktuur van de vektoren $\boldsymbol{L}^{-1}\boldsymbol{A}\boldsymbol{L}^{-T}(\boldsymbol{L}^T\boldsymbol{w}_1)$ en $\boldsymbol{L}^{-1}\boldsymbol{A}\boldsymbol{L}^{-T}(\boldsymbol{L}^T\boldsymbol{w}_2)$

worden gegeven door:

voor de vektor $\boldsymbol{L}^{-1}\boldsymbol{A}\boldsymbol{L}^{-T}(\boldsymbol{L}^T\boldsymbol{w}_1)$ geldt:

0.00000000000000
-0.26726122664035.$10^{-}07$
-0.00003194382479.$10^{-}07$
0.24999998539497.$10^{-}07$
0.09449110814885.$10^{-}07$
0.30860667750201.$10^{-}07$
0.00003726779469.$10^{-}07$
-0.24999998489583.$10^{-}07$
-0.09449110781891.$10^{-}07$
-0.03857583294983.$10^{-}07$

voor de vektor $\boldsymbol{L}^{-1}\boldsymbol{A}\boldsymbol{L}^{-T}(\boldsymbol{L}^T\boldsymbol{w}_2)$ geldt:

0.00000000000000
0.00000000000000
0.00000000000000
0.00000000000000
0.00000000000000
-0.27003084640058.$10^{-}07$
-0.00003260932103.$10^{-}07$
0.24999998528114.$10^{-}07$
0.09449110736914.$10^{-}07$
0.03857583488753.$10^{-}07$

Bibliografie

[1] Adrian E. Scheidegger: The physics of flow through porous media, PHD thesis University of Toronto, Toronto, 1958.

[2] F.T.M Nieuwstadt: Notities en oefenvraagstukken bij het college voortgezete stromingsleer b56A, (1992).

[3] E. Fjaer/ R.M. Holt/ P. Horsrud: Petroleum related rock mechanics, Elsevier,Amsterdam, 1992.

[4] M. Mokaddam (Verslag): Gedrag van de overdruk ten aanzien van de porositeit- en permeabiliteitsverschillen in sedimentlagen, TU Delft 1997.

[5] J. Bear: Dynamics of fluids in porous media, American Elsevier, New York, (1972).

[6] J. van Kan/ A. Segal: Numerieke methoden voor partiële differentiaalvergelijkingen, DUM Delft 1993.

[7] A. Meijerink/ H.A.v.d. Vorst: An iterative solution method for linear systems of which the coefficient matrix is a symmetric M-matrix, Math. Comp, 31, 148-162, 1977.

[8] Durtmond (Student report): An efficient numerical approximation method for fluid pressures in 3D sedimentary basins, Rijksuniversiteit Leiden, Department of mathematics, August 1995.

[9] G.H. Golub and C.F. Van Loan. *Matrix Computations* (*second edition*). The Johns Hopkins University Press, Baltimore, 1989.

[10] M.R. Hestenes and E. Stiefel. Methods of Conjugate Gradients for Solving Linear Systems. *J.Res.Nat.Bur.Stand.*, 49:409-436, 1952.

[11] H.A. van der Vorst/ C. Vuik: Voortgezette numerieke lineaire algebra a228, (1993).

[12] I.A. Gustafsson: A class of first order factorisation methodes, Journal: BIT, pages (142-156), (1978).

[13] S.C. Eisenstat: Efficient implementation of a class of preconditioned conjugate gradient methodes, Journal: Siam J.sci.stat.compt., pages (1-4), (1981).

[14] T. Ginsburg. The conjugate gradient method. In J.H. Wilkinson and C. Reinsch, editors, *Handbook for automatic computation 2 LinearAlgebra*, pages 57-69, Berlin, 1971.
Springer Verlag.