TNO-rapport
FSP-RPT-980039

Versnellen van SIMPLE(R) voor WISH3D met Krylov methoden

TNO Technisch Physische Dienst
TU Delft

Stieltjesweg 1
Postbus 155
2600 AD Delft

Telefoon 015 - 2692000
Fax 015 - 2692111

Datum
25 juni 1998

Auteur(s)
A. Saghir

Opdrachtnummer
015.783

Aan
Prof.dr.ir. P. Wesseling (TU Delft)
Dr.ir. C. Vuik (TU Delft)
Dr.ir. G.P. Boerstoel (TNO-TPD)

Inhoud

Voorwoord

Dit afstudeeronderzoek heeft plaats gevonden bij TNO-TPD (Technische Physische Dienst) gevestigd te Delft, in samenwerking met de TU Delft. Dit onderzoek heeft als doel het versnellen van de SIMPLE (Semi-Implicit Method voor Pressure-Linked Equations) methode met behulp van Krylov methoden.

Mijn dank aan Prof.dr.ir. P. Wesseling, Dr.ir C. Vuik en Dr.ir G.P. Boerstoel voor hun uitstekende begeleiding om dit afstudeerwerk goed te laten verlopen.

Ahmed Saghir

1 Inleiding

TNO (Nederlandse Organisatie voor Toegepast Natuurwetenschappelijk Onderzoek) is actief in diverse domeinen van de industrie en werkt ook samen met de universiteiten, waaronder de TU Delft. Eén van de 15 instituten van TNO is de TPD (Technisch Physische Dienst) te Delft die uit meerdere divisies bestaat. De divisie Fysische Systemen en Processen bestaat uit 5 subdivisies, waaronder Procesfysica. Procesfysica doet onderzoek aan processen waarin warmteoverdracht, stoftransport en stroming een rol spelen. Hiertoe wordt veelal gebruik gemaakt van geavanceerde rekenmodellen. Ook de ontwikkeling van thermofysische instrumentatie is belangrijk. Het modelleren van dynamische stromingen is een belangrijk vakgebied binnen Procesfysica. Om stromingen, warmteoverdracht en stoftransport te simuleren beschikt TNO-TPD over het CFD (Computational Fluid Dynamics) model TNO-WISH3D.

1.1 Het doel

Fysische problemen, veelal turbulente processen met of zonder verbranding, worden door een mathematische model beschreven. Er ontstaat een aantal niet-lineaire en gekoppelde partiële differentiaalvergelijkingen (PDV). Deze vergelijkingen worden met behulp van de eindige volume methode (EVM) gediscretiseerd. Verder worden de gediscretiseerde vergelijkingen opgelost met behulp van de iteratieve methode SIMPLE (Semi-Implicit Method for Pressure-Linked Equations), die in hoofdstuk 4 besproken wordt, of de variant daarvan SIMPLER (SIMPLE Revised). In de praktijk is gebleken dat de SIMPLE(R) methode vaak veel tijd in beslag neemt. Dit heeft aanleiding gegeven om dit onderzoek te starten. Dit onderzoek is een voortzetting van het onderzoek van H. Tazammourti [6] en heeft als doel het versnellen van de iteratieve methode SIMPLE(R). Bekend is dat, een iteratieve Krylov versnelling goedkoop is, vaak beter convergeert en dat langzaam convergerende componenten na een aantal iteraties geen invloed meer hebben op de convergentie (super lineaire convergentie). In dit onderzoek wordt de SIMPLE(R) methode versneld met behulp van een Krylov methode.

1.2 Algemene opzet

In hoofdstuk 2 van dit verslag wordt het mathematische model beschreven. Er wordt aandacht besteed aan de incompressibele Navier-Stokes vergelijkingen en het $k - \epsilon$ turbulentie model.
De genoemde vergelijkingen worden in hoofdstuk 3 gediscretiseerd met behulp van de eindige volume methode. Dit levert een stelsel vergelijkingen van de vorm $\mathbf{A}\underline{x} = \underline{b}$. Waarin $\mathbf{A}$ een ijle matrix is, en de vector $\underline{x}$ bevat de onbekende snelheidscomponenten u, v en de druk p. Als het $k - \epsilon$ model gebruikt wordt bevat $\underline{x}$ bovendien de turbulente kinetische energie k en de dissipatie ervan ϵ. In hoofdstuk 4 wordt de SIMPLE methode afgeleid gebruikmakend van de distributieve iteratie methode. Hierin wordt de matrix $\mathbf{A}$ gepostconditioneerd door een nog te definieren matrix $\mathbf{B}$. De meeste aandacht wordt besteed aan het versnellen van de SIMPLE methode met behulp van Krylov methoden. Gebruikmakend van de GCR (Generalized Conjugate Residuals) en WISH3D solvers (TDMA-solvers) wordt het GCR-SIMPLE algoritme ontwikkeld. Hoofdstuk 5 bevat een inleidende beschrijving over het toepassen van het GCR-SIMPLE algoritme. In hoofdstuk 6, 7 en 8 wordt het GCR-SIMPLE algoritme respectievelijk toegepast op Stokes, Navier-Stokes en Navier-Stokes problemen gecombineerd met een turbulentie model ($k-\epsilon$) model. De stroming tussen twee vlakke platen en de stroming in een backward facing step worden als testproblemen gebruikt. Voor beide testproblemen worden de resultaten, de convergentie en de executietijd met elkaar vergeleken.

2 Incompressibele stromingen

2.1 Inleiding

Op basis van de wetten van behoud van massa, impuls, energie en chemische stoffen beschrijft men de stroming van vloeibare of gasvormige media. Uitgaande van deze wetten worden de vergelijkingen geformuleerd (zie [4]).
We introduceren de sommatieconventie. Dit betekent dat in het geval van een herhaalde index (bijvoorbeeld $u_i u_i$) over alle coördinaatrichtingen wordt gesommeerd:

$$u_i u_i = \sum_{i=1}^{3} u_i u_i.$$

We beschouwen een vloeistofelement met een volume δV die zich zal bewegen door het medium met een snelheid u_j. Veranderingen langs de baan van het vloeistofelement als functie van de tijd worden aangegeven met de zogenaamde materiële afgeleide: $\frac{D}{Dt}$. In het vaste assenstelsel is deze afgeleide gedefinieerd als:

$$\frac{D}{Dt} = \frac{\partial}{\partial t} + u_j \frac{\partial}{\partial x_j}. \tag{2.1}$$

De term $\frac{\partial}{\partial t}$ staat bekend als de locale tijdsafgeleide en $u_j \frac{\partial}{\partial x_j}$ wordt de advectieterm genoemd. Deze advectieterm beschrijft de bijdrage aan de materiële afgeleide tengevolge van het transport in een ruimtelijk variërend veld.

2.2 Bewegingsvergelijkingen

2.2.1 Continuïteitsvergelijking

Behoud van massa voor een incompressibele stroming kunnen we als volgt schrijven

$$\frac{\partial \rho}{\partial t} + \frac{\partial (\rho u_i)}{\partial x_i} = 0, \tag{2.2}$$

waarbij ρ de dichtheid van het vloeistof is. Merk op dat de dichtheid kan variëren door temperatuurverschillen. Deze vergelijking wordt ook de continuïteitsvergelijking genoemd.

2.2.2 Navier-Stokes vergelijkingen

Behoud van impuls voor een incompressibele stroming wordt beschreven door de volgende vergelijkingen, die bekend staan als Navier-Stokes vergelijkingen.

$$\frac{\partial}{\partial t}(\rho u_i) + \frac{\partial}{\partial x_j}(\rho u_i u_j) = -\frac{\partial p}{\partial x_i} - \frac{\partial}{\partial x_j}(\tau_{ij}) + \rho g_i, \tag{2.3}$$

waarbij p en g_i respectievelijk de druk en de zwaartekrachtversnelling in de x_i-richting voorstellen. De term τ_{ij} is de spanningtensor. Voor een Newtonse vloeistof ziet τ_{ij} er als volgt uit

$$\tau_{ij} = -\mu(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i}) + \frac{2}{3}\mu\frac{\partial u_k}{\partial x_k}\delta_{ij}, \tag{2.4}$$

waarin δ_{ij} de Kronecker-delta voorstelt en μ is de dynamische viscositeit.

2.2.3 Stokes-stromingen

We beschouwen de stromingen waarin de traagheidskrachten verwaarloosbaar zijn ($Re \ll 1$, waarin Re het Reynoldsgetal voorstelt). Dit zijn stromingen waarvoor de niet-lineaire termen $\frac{\partial}{\partial x_j}(\rho u_j u_i)$ in (2.3) verwaarloosbaar zijn en wel Stokes-stromingen genoemd. De vergelijking (2.3) reduceert dan tot

$$\frac{\partial}{\partial t}(\rho u_i) = -\frac{\partial p}{\partial x_i} - \frac{\partial}{\partial x_j}(\tau_{ij}) + \rho g_i. \tag{2.5}$$

2.2.4 Behoud van energie

Behoud van energie in een vloeistof of gas wordt gegeven door

$$\begin{aligned}\frac{\partial}{\partial t}(\rho U + \frac{1}{2}\rho u_i u_i) + \frac{\partial}{\partial x_j}\left(\rho u_j(U + \frac{1}{2}u_i u_i)\right) = \\ -\frac{\partial}{\partial x_j}(q_j) - \frac{\partial}{\partial x_j}(p u_j) + \rho u_i g_i - \frac{\partial}{\partial x_j}(\tau_{ij} u_i) + S_{rad},\end{aligned} \tag{2.6}$$

waarbij U de inwendige energie per massaeenheid is, $\frac{1}{2}u_i u_i$ de kinetische energie, q_j de j-component van de energieflux en S_{rad} de stralingsbronterm.

Als we de energie vergelijking in termen van enthalpie H (met $H = U + \frac{p}{\rho}$) uitdrukken dan volgt

$$\frac{\partial}{\partial t}(\rho H + \frac{1}{2}\rho u_i u_i - p) + \frac{\partial}{\partial x_j}\left(\rho u_j (H + \frac{1}{2}u_i u_i)\right) = - \frac{\partial}{\partial x_j}(q_j) + \rho u_i g_i - \frac{\partial}{\partial x_j}(\tau_{ij} u_i) + S_{rad}. \tag{2.7}$$

In ons geval is de enthalpie de som van de voelbare enthalpie H_{sens} en de latente (chemische) enthalpie H_{lat}. Voor een ideaal gas geldt

$$dH_{sens} = c_p dT, \tag{2.8}$$

waarbij T de temperatuur voorstelt en c_p de specifieke warmte voor constante druk is. Voor de energieflux geldt bovendien het volgende

$$q_j = -\frac{\lambda}{c_p}\frac{\partial H}{\partial x_j}, \tag{2.9}$$

waarbij λ de thermische conductiviteit voorstelt. Voor meer details zie [3]. De kinetische energie $\frac{1}{2}u_i u_i$ is verwaarloosbaar ten opzichte van de enthalpie H zolang het Mach getal $\left(\text{Mach getal} = \frac{\text{snelheid van de stroming}}{\text{geluidssnelheid}}\right)$ klein is. Vervolgens worden de visceuse dissipatieterm en de de gravitatieterm in de energie vergelijking verwaarloosd. De vergelijking (2.7) reduceert tot

$$\frac{\partial}{\partial t}(\rho H - p) + \frac{\partial}{\partial x_j}(\rho u_j H) = -\frac{\partial}{\partial x_j}(\frac{\lambda}{c_p}\frac{\partial H}{\partial x_j}) + S_{rad}. \tag{2.10}$$

2.2.5 Behoud van chemische stoffen

Na enkele vereenvoudigingen (zie [3]) wordt de vergelijking voor behoud van chemische stoffen als volgt geformuleerd

$$\frac{\partial}{\partial t}(\rho M_a) + \frac{\partial}{\partial x_j}(\rho u_j M_a) = \frac{\partial}{\partial x_j}(\Gamma_a \frac{\partial M_a}{\partial x_j}) + S_a, \tag{2.11}$$

waarbij M_a de massafractie van de chemische stof a is, Γ_a is het product van de dichtheid van de vloeistof en de diffusiecoëfficient. S_a is de netto productie van de stoffen door chemische reacties.
De massfractie M_a voldoet aan

$$\sum_a M_a = 1. \tag{2.12}$$

Gebruikmakend van vergelijkingen (2.11), (2.12) en (2.2) volgt

$$\sum_a S_a = 0. \tag{2.13}$$

2.2.6 Toestandsvergelijking

Beschouwen we een vloeistof als mengsel van ideale gassen, dan luidt de toestandsvergelijking

$$Wp = \rho RT, \tag{2.14}$$

waarin W de moleculaire massa van het gemengde gas voorstelt en R de universele gasconstante.

2.3 Turbulente stromingen

Een turbulente stroming is een fluctuerend verschijnsel. In veel toepassingen is men niet geïnteresseerd in de instantane waarden van de stromingsvariabelen zoals, de snelheid en de druk, maar in de tijdsgemiddelde van deze waarden. Iedere instantane variabele ϕ kan in een gemiddelde en een fluctuatie gesplitst worden.

$$\phi(\underline{x}, t) = \overline{\phi}(\underline{x}, t) + \phi'(\underline{x}, t), \tag{2.15}$$

waarin het Reynoldsgemiddelde $\overline{\phi}(\underline{x})$ gegeven wordt door

$$\overline{\phi}(\underline{x}, t) = \frac{1}{\tau}\int_{-\frac{\tau}{2}}^{\frac{\tau}{2}} \phi(\underline{x}, t+s)ds, \tag{2.16}$$

en de uitdrukking (2.15) wordt de Reynoldsdecompositie genoemd. De middelingstijd τ wordt groot genoeg gekozen ten opzichte van dezelfde tijdschaal van de turbulentie maar steeds klein vergeleken met andere tijdsafhankelijke verschijnselen. Er geldt bovendien

$$\overline{\phi'} = 0. \tag{2.17}$$

Om in de volgende paragrafen met deze middelingsoperator te kunnen manipuleren in de bewegingsvergelijkingen eisen we dat deze operator voldoet aan de volgende voorwaarden zie [4], die bekend staan als de Reynoldsvoorwaarden

$$\begin{aligned}
&\overline{f+g} = \overline{f} + \overline{g}, \\
&\overline{\alpha f} = \alpha \overline{f}, \\
&\overline{\frac{\partial f}{\partial s}} = \frac{\partial \overline{f}}{\partial s}, \\
&\overline{\overline{f} g} = \overline{f}\,\overline{g},
\end{aligned}$$

hierin zijn f en g fluctuerende grootheden en α is een constante. Op basis van deze Reynoldsvoorwaarden kunnen we bijvoorbeeld de volgende relaties afleiden

$$\overline{\overline{f}} = \overline{f}, \quad \overline{\overline{f}\,\overline{g}} = \overline{f}\,\overline{g}. \tag{2.18}$$

2.3.1 Continuïteitsvergelijking

Als we de Reynoldsdecompositie $\rho = \overline{\rho} + \rho'$, $u = \overline{u} + u'$ in de continuïteitsvergelijking voor incompressibele stroming (2.2) invullen dan volgt

$$\frac{\partial(\overline{\rho} + \rho')}{\partial t} + \frac{\partial}{\partial x_j}[(\overline{\rho} + \rho')(\overline{u}_j + u'_j)] = 0. \tag{2.19}$$

Wanneer de termen waarin de fluctuatie van de dichtheid ρ' voorkomt worden verwaarloosd, reduceert (2.19) tot

$$\frac{\partial \overline{\rho}}{\partial t} + \frac{\partial}{\partial x_j}(\overline{\rho}\ \overline{u}_j) + \frac{\partial}{\partial x_j}(\overline{\rho} u'_j) = 0. \tag{2.20}$$

Vervolgens wordt het Reynoldsgemiddeld van (2.20) genomen en dit levert een vergelijking voor de gemiddelde stroming

$$\frac{\partial \overline{\rho}}{\partial t} + \frac{\partial}{\partial x_j}(\overline{\rho}\ \overline{u}_j) = 0. \tag{2.21}$$

2.3.2 Navier-Stokes vergelijkingen

Behoud van impuls voor een turbulente stroming wordt als volgt geformuleerd (naast de termen waarin de fluctuatie van de dichtheid voorkomt worden bovendien de termen met derde orde correlaties ook verwaarloosd)

$$\frac{\partial}{\partial t}(\overline{\rho}\ \overline{u}_i) + \frac{\partial}{\partial x_j}(\overline{\rho}\ \overline{u}_i\ \overline{u}_j + \overline{\rho}\overline{u'_i u'_j}) = -\frac{\partial \overline{p}}{\partial x_i} - \frac{\partial}{\partial x_j}(\overline{\tau}_{ij}) + \overline{\rho} g_i, \tag{2.22}$$

met

$$\overline{\tau}_{ij} = -\mu\left(\frac{\partial \overline{u}_i}{\partial x_j} + \frac{\partial \overline{u}_j}{\partial x_i}\right) + \frac{2}{3}\mu \frac{\partial \overline{u}_k}{\partial x_k}\delta_{ij}. \tag{2.23}$$

2.3.3 Behoud van energie

In een turbulente stroming wordt de energiebalans geformuleerd als

$$\frac{\partial}{\partial t}(\overline{\rho}\overline{H} - \overline{p}) + \frac{\partial}{\partial x_j}(\overline{\rho}\ \overline{u}_j \overline{H} + \overline{\rho}\overline{u'_j H'}) = \frac{\partial}{\partial x_j}\left(\frac{\lambda}{c_p}\frac{\partial \overline{H}}{\partial x_j}\right) + \overline{S}_{rad} \tag{2.24}$$

2.3.4 Behoud van chemische stoffen

$$\frac{\partial}{\partial t}(\overline{\rho}\overline{M}_a) + \frac{\partial}{\partial x_j}(\overline{\rho}\ \overline{u}_j \overline{M}_a + \overline{\rho}\overline{u'_j M'_a}) = \frac{\partial}{\partial x_j}\left(\Gamma_a \frac{\partial \overline{M}_a}{\partial x_j}\right) + \overline{S}_a \tag{2.25}$$

2.3.5 Het standaard $k - \epsilon$ turbulentie model

Door het gebruik van de Reynoldsdecompositie ontstaan nieuwe onbekenden. Om de onbekende Reynoldstermen $\bar{\rho}\overline{u_i'u_j'}$, $\bar{\rho}\overline{u_j'H'}$ en $\bar{\rho}\overline{u_j'M_a'}$ te evalueren maken we gebruik van het $k - \epsilon$ model (zie [3]).

$$\bar{\rho}\overline{u_i'u_j'} = -\mu_t(\frac{\partial \bar{u}_i}{\partial x_j} + \frac{\partial \bar{u}_j}{\partial x_i}) + \frac{2}{3}\bar{\rho}k\delta_{ij}, \tag{2.26}$$

waarbij $k = \frac{1}{2}\overline{u_i'u_i'}$ de turbulente kinetische energie voorstelt, $\mu_t = \frac{C_\mu \bar{\rho} k^2}{\epsilon}$ is de turbulente viscositeit, met C_μ een empirische constante is en ϵ is de dissipatie van de turbulente kinetische energie:

$$\epsilon = \frac{\mu}{\bar{\rho}}\overline{\left(\frac{\partial u_i'}{\partial x_j}\frac{\partial u_i'}{\partial x_j}\right)}. \tag{2.27}$$

De turbulente kinetische energie en de dissipatie ervan voldoen aan

$$\frac{\partial}{\partial t}(\bar{\rho}k) + \frac{\partial}{\partial x_j}(\bar{\rho}\bar{u}_j k) = \frac{\partial}{\partial x_j}((\mu + \frac{\mu_t}{\sigma_k})\frac{\partial k}{\partial x_j}) + P_k - \bar{\rho}\epsilon, \tag{2.28}$$

$$\frac{\partial}{\partial t}(\bar{\rho}\epsilon) + \frac{\partial}{\partial x_j}(\bar{\rho}\bar{u}_j \epsilon) = \frac{\partial}{\partial x_j}((\mu + \frac{\mu_t}{\sigma_\epsilon})\frac{\partial \epsilon}{\partial x_j}) + (C_{\epsilon 1}P_k - C_{\epsilon 2}\bar{\rho}\epsilon)\frac{\epsilon}{k}, \tag{2.29}$$

waarbij σ_k en σ_ϵ de turbulente Prandtlgetallen voorstellen, $C_{\epsilon 1}$ en $C_{\epsilon 2}$ zijn empirische constanten en P_k is de productie term gegeven door

$$P_k = [\mu_t(\frac{\partial \bar{u}_i}{\partial u_j} + \frac{\partial \bar{u}_j}{\partial u_i}) - \frac{2}{3}\bar{\rho}k\delta_{ij}]\frac{\partial \bar{u}_i}{\partial u_j}. \tag{2.30}$$

Op analoge wijze kunnen de volgende Reynoldstermen gemodelleerd worden.

$$\bar{\rho}\overline{u_j'H'} = -\frac{\mu_t}{\sigma_H}\frac{\partial \overline{H}}{\partial x_j}, \tag{2.31}$$

$$\bar{\rho}\overline{u_j'M_a'} = -\frac{\mu_t}{\sigma_a}\frac{\partial \overline{M_a}}{\partial x_j}, \tag{2.32}$$

waarin σ_H het turbulente Prandtl getal is en σ_a het turbulente Schmidt getal voor chemische stoffen.

2.3.6 Favre-decompositie

Voor turbulente stromingen met een temperatuurafhankelijke dichtheid kunnen we de zogenoemde Favre-decompositie gebruiken (zie [5]). Het voordeel hiervan boven de Reynoldsdecompositie is het vermijden van de expliciete dichtheidscorrelaties (die bij de Reynoldsdecompositie verwaarloosd zijn). De Favre-decompositie van een variabele ϕ luidt als volgt

$$\phi(\underline{x},t) = \tilde{\phi}(\underline{x},t) + \phi''(\underline{x},t), \tag{2.33}$$

met

$$\tilde{\phi} = \frac{\overline{\rho\phi}}{\overline{\rho}}. \tag{2.34}$$

Het Favre gemiddelde van de fluctuatie ϕ'' is nul. waarbij $\overline{\rho\phi}$ en $\overline{\rho}$ Reynolds-gemiddelden zijn. Bovendien geldt er

$$\overline{\phi''} = -\frac{\overline{\rho'\phi'}}{\overline{\rho}}, \tag{2.35}$$

met ρ' de fluctuatie van ρ.

2.4 Begin- en randvoorwaarden

Een eenduidige oplossing voor de bewegingsvergelijkingen is alleen mogelijk als er begin- en randvoorwaarden gegeven zijn.

2.4.1 Beginvoorwaarden

Voor instationaire stromingen wordt op een begintijdstip t_0 de waarde van ϕ gegeven door een functie ϕ_0

$$\phi|_{t_0} = \phi_0. \tag{2.36}$$

2.4.2 Randvoorwaarden

We kunnen de volgende randvoorwaarden onderscheiden

$$\phi = f, \qquad (Dirichlet) \tag{2.37}$$

$$(\Gamma_\phi \frac{\partial \phi}{\partial x_j})n_j = f. \qquad (Neumann) \tag{2.38}$$

Voor een stilstaand randoppervlak reduceert de Dirichlet randvoorwaarde voor het snelheidsveld tot

$$u_t = 0 \quad \text{(no-slip conditie)}, \tag{2.39}$$

$$u_n = 0 \quad \text{(no-flow conditie)}, \tag{2.40}$$

waarin u_t en u_n respectievelijk de tangentiële en de normale snelheden voorstellen. De randvoorwaarden voor de temperatuur worden gegeven door de temperatuur op de rand

$$T = T_{rand}, \tag{2.41}$$

of door de warmteflux q_{rand} op de rand

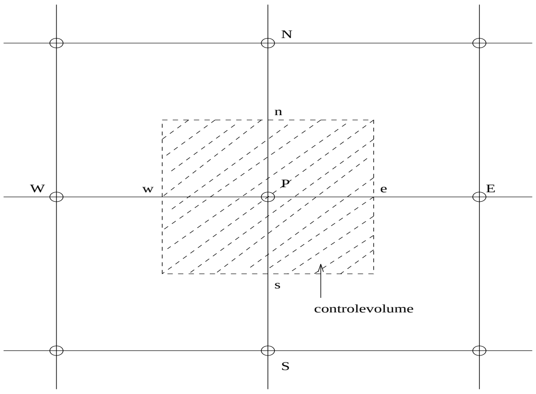
$$-k\frac{\partial T}{\partial n} = q_{rand}. \tag{2.42}$$

Lijst van behandelde onderwerpen:

- Incompressibele stromingen
  - Inleiding
  - Bewegingsvergelijkingen
    - Continuïteitsvergelijking
    - Navier-Stokes vergelijkingen
    - Stokes stromingen
    - Behoud van energie
    - Behoud van chemische stoffen
    - Toestandsvergelijking
  - Turbulente stromingen
    - Continuïteitsvergelijking
    - Navier-Stokes vergelijkingen
    - Behoud van energie
    - Behoud van chemische stoffen
    - Het standaard $k - \epsilon$ turbulentie model
    - Favre-decompositie
  - Begin- en randvoorwaarden
    - Beginvoorwaarden
    - Randvoorwaarden

3 Discretisatie

In deze sectie wordt aandacht besteed aan het oplossen van de vergelijkingen genoemd in Hoofdstuk 2. Omdat een analytische oplossing in het algemeen niet mogelijk is voor dit soort vergelijkingen, gebruiken we numerieke methoden. De gebruikte numerieke methode in dit verslag is de eindige volume methode (EVM) beschreven in [2]. Het drie-dimensionale gebied wordt in een aantal controle volumina verdeeld, zie figuur 3.1 voor het twee-dimensionale geval. Ieder controlevolume heeft een roosterpunt P als middelpunt waarop alle scalaire grootheden ϕ (druk p of temperatuur T) worden geëvalueerd. De roosterpunten voor de snelheidscomponenten zijn genomen op de zijvlakken van het controlevolume zodat ze verschoven zijn ten opzichte van het roosterpunt P. Met andere woorden we gebruiken het zogenaamde verschoven rooster (staggered grid). Om het hydrodynamische probleem (bestaande uit



Figuur 3.1: Controlevolume en roosterpunten voor het twee-dimensionale geval

de Navier-Stokes vergelijkingen en de continuïteitsvergelijking) op te lossen, dat wil zeggen om de snelheden en de druk te bepalen, zullen we gebruik maken van de SIMPLE methode die in Hoofdstuk 4 verder besproken zal worden (zie [2]).

De vergelijkingen (2.2), (2.3), (2.6) en (2.11) in Hoofdstuk 2 hebben de vorm van een convectie-diffusie vergelijking. De algemene vorm van zulke vergelijkingen voor een variabele ϕ luidt

$$\frac{\partial}{\partial t}(\rho\phi) + \frac{\partial}{\partial x_j}(\rho u_j \phi) = \frac{\partial}{\partial x_j}(\Gamma \frac{\partial \phi}{\partial x_j}) + S_\phi. \quad (3.1)$$

3.1 De convectie-diffusie vergelijking

De vergelijking (3.1) kunnen we ook als volgt formuleren

$$\frac{\partial}{\partial t}(\rho\phi) + \frac{\partial}{\partial x_1}(J_{x_1}) + \frac{\partial}{\partial x_2}(J_{x_2}) + \frac{\partial}{\partial x_3}(J_{x_3}) = S_\phi, \tag{3.2}$$

met

$$J_{x_1} = \rho u_1 \phi - \Gamma \frac{\partial \phi}{\partial x_1}, \tag{3.3}$$

$$J_{x_2} = \rho u_2 \phi - \Gamma \frac{\partial \phi}{\partial x_2}, \tag{3.4}$$

$$J_{x_3} = \rho u_3 \phi - \Gamma \frac{\partial \phi}{\partial x_3}, \tag{3.5}$$

waarin u_1, u_2 en u_3 de snelheidscomponenten respectievelijk in de x_1, x_2 en de x_3-richting voorstellen. Beschouw een roosterpunt P en zijn zes buurpunten (vier buurpunten in het twee-dimensionaal geval, zie figuur 3.1) E(East), W(West), N(North), S(South), T(Top) en B(Bottom). Op ieder zijvlak van het controlevolume kan de totale flux gedefinieerd worden, voor het zijvlak e (east) is die gedefinieerd door

$$J_e = \left[\rho u_1 \phi - \Gamma \frac{\partial \phi}{\partial x_1}\right]_e A_e, \tag{3.6}$$

waarbij A_e de oppervlakte is van het bovengenoemde zijvlak en $[\quad]_e$ de evaluatie aanduidt van alle grootheden in het bijbehorende zijvlak. Nu integreren we de vergelijking (3.2) over een controlevolume met volume $\Delta V = \Delta x \Delta y \Delta z$ en een tijdsinterval $[t, t+\Delta t]$, voor meer details zie [6]. Dus

$$\int_t^{t+\Delta t} \int_{\Delta V} \frac{\partial(\rho\phi)}{\partial t} dV dt + \int_t^{t+\Delta t} \int_{\Delta V} \left[\frac{\partial(J_{x_1})}{\partial x_1} + \frac{\partial(J_{x_2})}{\partial x_2} + \frac{\partial(J_{x_3})}{\partial x_3}\right] dV dt = $$
$$= \int_t^{t+\Delta t} \int_{\Delta V} S_\phi dV dt. \tag{3.7}$$

Voor de tijdsdiscretisatie wordt in dit hoofdstuk het volledig impliciete tijdschema genoemd. Voor de tijdsafhankelijke term wordt er aangenomen dat ρ en ϕ constant zijn over het hele controlevolume. De bronterm wordt op de gebruikelijke wijze gelineariseerd

$$S_\phi = S_C + S_P \phi_P, \tag{3.8}$$

hier staat de index P niet voor het roosterpunt P. Voor details verwijzen we naar Patankar [2]. Gebruikmakend van het divergentie-theorema van Gauss wordt de convectie-diffusie term gediscretiseerd

$$\int_{\Delta V} \frac{\partial}{\partial x_j}(J_j) dV = \int_{\partial V} J_j n_j dA, \tag{3.9}$$

waarbij ∂V de rand van ΔV voorstelt en n_j de naar buiten gerichte eenheidsnormaal van $\partial V_j, j = e, w, n, s, t, b$. Er volgt dus

$$\int_{\Delta V} \frac{\partial}{\partial x_j}(J_j)dV = (J_e - J_w) + (J_n - J_s) + (J_t - J_b). \tag{3.10}$$

Dus

$$\begin{aligned}\left[(\rho\phi)_P - (\rho\phi)_{P,0}\right]\frac{\Delta V}{\Delta t} + (J_e - J_w) + (J_n - J_s) + (J_t - J_b) = \\ = (S_c + S_P\phi_P)\Delta V,\end{aligned} \tag{3.11}$$

waarin $(\rho\phi)_{P,0}$ de waarde van $(\rho\phi)_P$ op het tijdstip t voorstelt, terwijl alle andere grootheden $(\phi_P, \phi_E, ..., \phi_B)$ de onbekenden op tijdstip $t + \Delta t$ zijn. De totale fluxen J_w, J_n, J_s, J_t en J_b worden op analoge wijze als J_e in (3.6) gedefinieerd. Nu gaan we de totale fluxen voor ieder zijvlak van het controlevolume evalueren. Beschouw de totale flux J_e in (3.6) bestaande uit een convectieve term $\rho u_1 \phi$ en een diffusieve term $\Gamma \frac{\partial \phi}{\partial x_1}$. Voor de diffusieve term wordt gebruik gemaakt van de centrale differentie methode. Benadering van de convectieve term met behulp van de centrale differentie methode leidt tot instabiliteit in de numerieke oplossing als het rooster Pecletgetal Pe (bijvoorbeeld voor e geldt $Pe = \frac{(\rho u_1)_e (\delta x_1)_e}{\Gamma_e}$, waarbij $(\delta x_1)_e$ de afstand tussen de roosterpunten P en E voorstelt) groter is dan 2. Om die reden wordt de convectieve term met behulp van een hybrid schema benaderd.
Met andere woorden

- Centrale differentie methode als $|Pe| \leq 2$.
- Upwind schema als $|Pe| > 2$.

Hieronder wordt het eerste orde upwind schema beschreven:
De waarde van ϕ_e wordt gegeven door

$$\phi_e = \phi_P \quad \text{als} \quad (u_1)_e > 0, \tag{3.12}$$
$$\phi_e = \phi_E \quad \text{als} \quad (u_1)_e \leq 0. \tag{3.13}$$

Er volgt dus

$$J_e = \left[(\rho u_1)_e \phi_P + \Gamma_e \frac{\phi_E - \phi_P}{(\delta x_1)_e}\right] A_e \quad \text{als} \quad (u_1)_e > 0, \tag{3.14}$$
$$J_e = \left[(\rho u_1)_e \phi_E + \Gamma_e \frac{\phi_E - \phi_P}{(\delta x_1)_e}\right] A_e \quad \text{als} \quad (u_1)_e \leq 0, \tag{3.15}$$

waarbij $A_e = \Delta y \Delta z$. De convectieve term kunnen we ook als volgt schrijven

$$(\rho u_1)_e \phi_e = \max\Big((\rho u_1)_e, 0\Big)\phi_P + \max\Big(-(\rho u_1)_e, 0\Big)\phi_E. \tag{3.16}$$

Voor de rand w geldt

$$(\rho u_1)_w \phi_w = \max\Big((\rho u_1)_w, 0\Big)\phi_W + \max\Big(-(\rho u_1)_w, 0\Big)\phi_P. \tag{3.17}$$

Op analoge manier worden de fluxen voor de zijvlakken n, s, t en b van het controlevolume gediscretiseerd. De discretisatie van de convectie diffusie vergelijking (3.1) kunnen we in de volgende vorm schrijven

$$a_P\phi_P = a_E\phi_E + a_W\phi_W + a_N\phi_N + a_S\phi_S + a_T\phi_T + a_B\phi_B + b, \tag{3.18}$$

met

$$a_E = \max\Big(-(\rho u_1)_e, 0\Big)A_e + \frac{\Gamma_e A_e}{(\delta x_1)_e}, \tag{3.19}$$

$$a_W = \max\Big((\rho u_1)_w, 0\Big)A_w + \frac{\Gamma_w A_w}{(\delta x_1)_w}, \tag{3.20}$$

$$a_N = \max\Big(-(\rho u_2)_n, 0\Big)A_n + \frac{\Gamma_n A_n}{(\delta x_2)_n}, \tag{3.21}$$

$$a_S = \max\Big((\rho u_2)_s, 0\Big)A_s + \frac{\Gamma_s A_s}{(\delta x_2)_s}, \tag{3.22}$$

$$a_T = \max\Big(-(\rho u_3)_t, 0\Big)A_t + \frac{\Gamma_t A_t}{(\delta x_3)_t}, \tag{3.23}$$

$$a_B = \max\Big((\rho u_3)_b, 0\Big)A_3 + \frac{\Gamma_b A_b}{(\delta x_3)_b}, \tag{3.24}$$

$$a_P = a_E + a_W + a_N + a_S + a_T + a_B + \rho_P\frac{\Delta V}{\Delta t} - S_P\Delta V, \tag{3.25}$$

$$b = S_C\Delta V + (\rho\phi)_{P,0}\frac{\Delta V}{\Delta t}. \tag{3.26}$$

De lineaire vergelijking (3.18) kan ook in de volgende vorm geschreven worden

$$a_P\phi_P = \sum_{nb} a_{nb}\phi_{nb} + b, \tag{3.27}$$

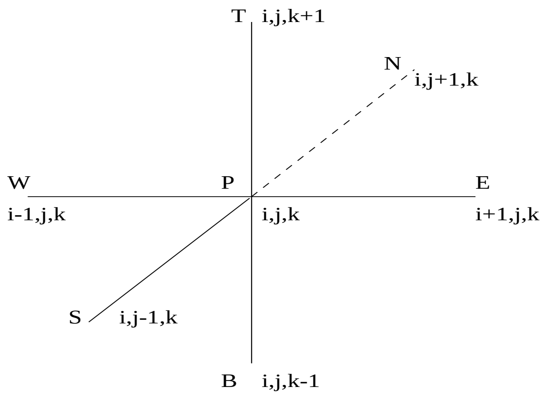
waarbij de sommatie uitgevoerd wordt over de buurpunten van het punt P.

Als we aan het punt P de index i, j, k toekennen, aan het punt E de index $i+1, j, k$ etc..., zie figuur 3.2, dan kunnen we (3.18) als volgt schrijven

$$\begin{aligned} -a^B_{ijk}\phi_{i,j,k-1} - a^S_{ijk}\phi_{i,j-1,k} - a^W_{ijk}\phi_{i-1,j,k} + a^P_{ijk}\phi_{i,j,k} + \\ - a^E_{ijk}\phi_{i+1,j,k} - a^N_{ijk}\phi_{i,j+1,k} - a^T_{ijk}\phi_{i,j,k+1} = b_{ijk} \end{aligned} \tag{3.28}$$

We zullen ons beperken tot de twee-dimensionaal stationaire stromingen met Dirichlet randvoorwaarden. De gediscretiseerde vergelijkingen luiden dan

$$a_P\phi_P = a_E\phi_E + a_W\phi_W + a_N\phi_N + a_S\phi_S + b, \tag{3.29}$$



Figuur 3.2: Indices van de buurpunten van het roosterpunt P

met

$$a_E = \max\Big(-(\rho u_1)_e, 0\Big)\Delta y + \frac{\Gamma_e \Delta y}{(\delta x_1)_e}, \tag{3.30}$$

$$a_W = \max\Big((\rho u_1)_w, 0\Big)\Delta y + \frac{\Gamma_w \Delta y}{(\delta x_1)_w}, \tag{3.31}$$

$$a_N = \max\Big(-(\rho u_2)_n, 0\Big)\Delta x + \frac{\Gamma_n \Delta x}{(\delta x_2)_n}, \tag{3.32}$$

$$a_S = \max\Big((\rho u_2)_s, 0\Big)\Delta x + \frac{\Gamma_s \Delta x}{(\delta x_2)_s}, \tag{3.33}$$

$$a_P = a_E + a_W + a_N + a_S - S_P \Delta x \Delta y, \tag{3.34}$$

$$b = S_C \Delta x \Delta y. \tag{3.35}$$

De vergelijking (3.29) kunnen we als volgt schrijven

$$-a_{ij}^S \phi_{i,j-1} - a_{ij}^W \phi_{i-1,j} + a_{ij}^P \phi_{i,j} - a_{ij}^E \phi_{i+1,j} - a_{ij}^N \phi_{i,j+1} = b_{ij}, \quad \text{voor} \quad 1 \le i \le n, 1 \le j \le m, \tag{3.36}$$

waarin n en m respectievelijk het aantal roosterpunten in x_1- en x_2-richting is. Uit (3.36) volgt een stelsel vergelijkingen van de vorm

$$\mathbf{A}\underline{\phi} = \underline{b}, \tag{3.37}$$

met

$$\underline{\phi} = (\phi_{11}, \phi_{21} \ldots, \phi_{n1}, \phi_{12}, \phi_{22}, \ldots, \phi_{n2}, \ldots\ldots, \phi_{1m}, \phi_{2,m}, \ldots, \phi_{nm})^T, \tag{3.38}$$

en de nm bij nm matrix $\mathbf{A}$ ziet er als volgt uit

$$\mathbf{A} = \begin{pmatrix} A_1 & N_1 & & & & \\ S_2 & A_2 & N_2 & & & \\ & \ddots & \ddots & \ddots & & \\ & & \ddots & \ddots & \ddots & \\ & & & S_{m-1} & A_{m-1} & N_{m-1} \\ & & & & S_m & A_m \end{pmatrix} \tag{3.39}$$

met

$$A_j = \begin{pmatrix} a^P_{1j} & -a^E_{1j} & & & & \\ -a^W_{2j} & a^P_{2j} & -a^E_{2j} & & & \\ & \ddots & \ddots & \ddots & & \\ & & \ddots & \ddots & \ddots & \\ & & & -a^W_{n-1j} & a^P_{n-1j} & -a^E_{n-1j} \\ & & & & -a^W_{nj} & a^P_{nj} \end{pmatrix} \tag{3.40}$$

voor $1 \leq j \leq m$ en

$$S_j = -\,\mathbf{Diag}(a^S_{1j}, a^S_{2j}, \ldots\ldots, a^S_{nj}), \quad \text{voor} \qquad 1 \leq j \leq m-1, \tag{3.41}$$

$$N_j = -\,\mathbf{Diag}(a^N_{1j}, a^N_{2j}, \ldots\ldots, a^N_{nj}), \quad \text{voor} \qquad 2 \leq j \leq m. \tag{3.42}$$

3.2 De Stokes vergelijkingen

Wij beperken ons tot een twee-dimensionaal stationaire stroming met constante dichtheid ρ. De continuïteitsvergelijking (2.2) reduceert tot

$$\frac{\partial u_i}{\partial x_i} = 0. \tag{3.43}$$

Het snelheidsveld is dus divergentievrij en dit reduceert (2.4) tot

$$\tau_{ij} = -\mu\left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i}\right). \tag{3.44}$$

Substitutie van (3.44) in (2.5) levert de Stokes vergelijkingen. Voor het gemak wordt de volgende notatie ingevoerd: $u := u_1, v := u_2, x := x_1$ en $y := x_2$. Om het Stokes probleem op te lossen zullen we dus de volgende vergelijkingen discretiseren.

$$\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} = 0, \tag{3.45}$$

$$\mu\left(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2}\right) - \frac{\partial P}{\partial x} = 0, \tag{3.46}$$

$$\mu\left(\frac{\partial^2 v}{\partial x^2} + \frac{\partial^2 v}{\partial y^2}\right) - \frac{\partial P}{\partial y} = 0, \tag{3.47}$$

waarbij $\nu = \frac{\mu}{\rho}$ de kinematische viscositeit voorstelt en P is de gemodificeerde druk en wordt gegeven door $p = p_0 + \rho g_i x_i + P$, waarin p_0 een referentiedruk voorstelt. We zullen in het vervolg voor de gemodificeerde druk weer de kleine letter p gebruiken.
De vergelijkingen (3.45),(3.46) en (3.47) worden in een dimensieloze vorm geschreven. Voer in een lengteschaal L en snelheidsschaal U. Met deze lengte- en snelheidsschaal definiëren we de volgende dimensieloze variabelen (voor het gemak blijft de notatie onveranderd).
$x := \frac{x}{L}, y := \frac{y}{L}, u := \frac{u}{U}, v := \frac{v}{U}, p := \frac{Rep}{\rho U^2}$. met $Re = \frac{UL}{\nu}$.

Substitutie van de ingevoerde variabelen in (3.45),(3.46) en (3.47) levert

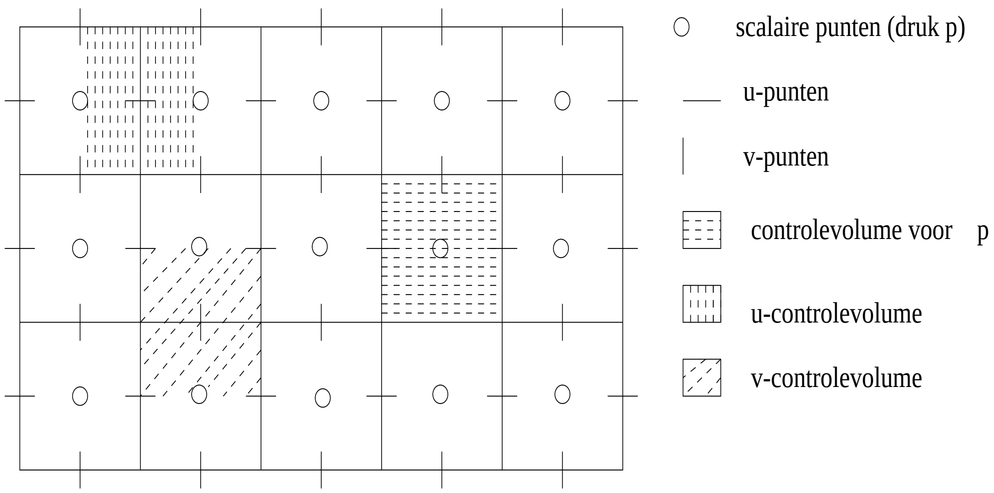
$$\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} = 0, \tag{3.48}$$

$$\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} - \frac{\partial p}{\partial x} = 0, \tag{3.49}$$

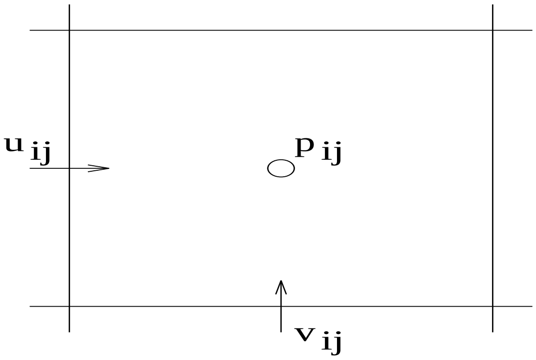
$$\frac{\partial^2 v}{\partial x^2} + \frac{\partial^2 v}{\partial y^2} - \frac{\partial p}{\partial y} = 0. \tag{3.50}$$

3.2.1 Discretisatie

Met behulp van de technieken behandeld in Hoofdstuk 2 zullen we (3.48), (3.49) en (3.50) discretiseren. We gebruiken een uniform rekenrooster G (zie figuur 3.3) met een stapgrootte h_1 in de x-richting en h_2 in de y-richting. Een cel met middelpunt $((i - \frac{1}{2})h_1, (j - \frac{1}{2})h_2)$ noemen we Ω_{ij}. De variabelen in het middelpunt van Ω_{ij} en in het middelpunt van de linker zijkant en de onderkant van Ω_{ij} zijn gelabeld met ij (zie figuur 3.4).



Figuur 3.3: Controlevolumina van een verschoven rooster



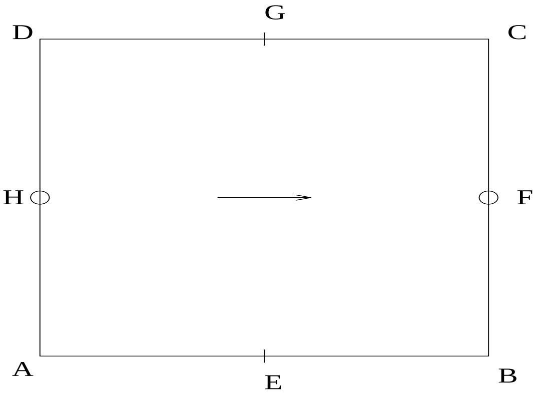
Figuur 3.4: Een cel gelabeld met ij

Eindige volume methode

De continuïteitsvergelijking (3.48) wordt over Ω_{ij} geïntegreerd. Dit levert

$$(u_{i+1,j} - u_{ij})h_2 + (v_{i,j+1} - v_{ij})h_1 = 0 \tag{3.51}$$

De Stokes vergelijking (3.49) in de x-richting wordt over een verschoven controlevolume (met middelpunt $((i-1)h_1, (i-\frac{1}{2})h_2)$) geïntegreerd.
Het verschoven controlevolume is gegeven in figuur 3.5. Dit levert als resultaat



Figuur 3.5: Verschoven controlevolume voor de Stokes vergelijking in x-richting

$$h_2\left[[\frac{\partial u}{\partial x}]_F - [\frac{\partial u}{\partial x}]_H\right] + h_1\left[[\frac{\partial u}{\partial y}]_G - [\frac{\partial u}{\partial y}]_E\right] - (p_F - p_H)h_2 = 0. \tag{3.52}$$

Bovendien worden de termen $\frac{\partial u}{\partial x}$ en $\frac{\partial u}{\partial y}$ benaderd door de centrale differentie methode. Dit leidt tot het volgende resultaat

$$\frac{h_2}{h_1}(u_{i-1,j} - 2u_{i,j} + u_{i+1,j}) + \frac{h_1}{h_2}(u_{i,j-1} - 2u_{i,j} + u_{i,j-1}) + \\ -(p_{i,j} - p_{i-1,j})h_2 = 0 \tag{3.53}$$

Toepassing van dezelfde technieken als hierboven op de Stokes vergelijking (3.50) in y-richting levert

$$\begin{aligned}\frac{h_2}{h_1}(v_{i-1,j}-2v_{i,j}+v_{i+1,j})+\frac{h_1}{h_2}(v_{i,j-1}-2v_{i,j}+v_{i,j+1})&+\\ -(p_{i,j}-p_{i,j-1})h_1&=0\end{aligned}\tag{3.54}$$

Randvoorwaarden

We beperken ons tot de Dirichlet randvoorwaarden, dat wil zeggen dat de snelheden op de randen gegeven zijn. Neem aan dat de cel ABCD in figuur 3.5 op de onderste rand ligt. Dus de snelheden op AB gegeven zijn, dan kunnen we u_{i0} elimineren door de volgende benadering in te voeren

$$u_{i0}=2u_E-u_{i1}.$$

Op de zijrand is de snelheid u_{1j} gegeven, dat wil zeggen

$$u_{1j}=u_{geg}.$$

Samenvatting van de gediscretiseerde vergelijkingen

De gediscretiseerde Stokes- en continuïteitsvergelijking kunnen we in de volgende vorm schrijven (voor de rangschikking van u, v en p zie (3.38))

$$\begin{pmatrix}\mathbf{Q}_1 & 0 & \mathbf{G}_1\\ 0 & \mathbf{Q}_2 & \mathbf{G}_2\\ \mathbf{G}_1^* & \mathbf{G}_2^* & 0\end{pmatrix}\begin{pmatrix}u\\ v\\ p\end{pmatrix}=\begin{pmatrix}b_1\\ b_2\\ b_3\end{pmatrix},\tag{3.55}$$

waarbij $\mathbf{G}_1^*$ en $\mathbf{G}_2^*$ respectievelijk de getransponeerde van $\mathbf{G}_1$ en $\mathbf{G}_2$ is en

$$\mathbf{Q}_1=\begin{pmatrix}-2(a+1/a) & a & & 1/a & \\ a & -2(a+1/a) & a & & \\ & & & & 1/a\\ 1/a & \ddots & \ddots & \ddots & \\ & & a & -2(a+1/a) & a\\ & 1/a & & a & -2(a+1/a)\end{pmatrix},$$

waarbij $a=\frac{h_2}{h_1}$,

$$\mathbf{G}_1=\begin{pmatrix}-h_2 & & & & \\ h_2 & -h_2 & & & \\ & \ddots & \ddots & & \\ & & \ddots & \ddots & \\ & & & h_2 & -h_2\end{pmatrix}.$$

In de matrices $\mathbf{Q}_1$ en $\mathbf{G}_1$ moeten de randvoorwaarden nog verwerkt worden. De matrices $\mathbf{Q}_2$ en $\mathbf{G}_2$ hebben respectievelijk een vergelijkbare gedaante als $\mathbf{Q}_1$ en $\mathbf{G}_1$. Dit stelsel kunnen we simpelweg als volgt schrijven

$$\mathbf{A}x = b. \tag{3.56}$$

3.3 De Navier-Stokes vergelijkingen

Na het discretiseren van de Stokes vergelijkingen zullen we de Navier-Stokes vergelijkingen (2.3) en (2.2) waarin niet-lineaire termen voorkomen, discretiseren met dezelfde technieken die gebruikt zijn in sectie 3.2. Voor niet-lineaire termen zullen we een hybrid schema gebruiken. Beschouw de stationaire tweedimensionale Navier-Stokes vergelijkingen. In deze sectie wordt aangenomen dat ρ constant is. Dat wil zeggen dat de volgende vergelijkingen gediscretiseerd moeten worden

$$\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} = 0, \tag{3.57}$$

$$\frac{\partial (u^2)}{\partial x} + \frac{\partial (uv)}{\partial y} = -\frac{1}{\rho}\frac{\partial p}{\partial x} + \nu(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2}), \tag{3.58}$$

$$\frac{\partial (uv)}{\partial x} + \frac{\partial (v^2)}{\partial y} = -\frac{1}{\rho}\frac{\partial p}{\partial y} + \nu(\frac{\partial^2 v}{\partial x^2} + \frac{\partial^2 v}{\partial y^2}). \tag{3.59}$$

Gebruikmakend van de dimensieloze variabelen ingevoerd in sectie 3.2 wordt de dimensieloze vorm van deze vergelijkingen verkregen.

$$\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} = 0, \tag{3.60}$$

$$\frac{\partial (u^2)}{\partial x} + \frac{\partial (uv)}{\partial y} = -\frac{\partial p}{\partial x} + \frac{1}{Re}(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2}), \tag{3.61}$$

$$\frac{\partial (uv)}{\partial x} + \frac{\partial (v^2)}{\partial y} = -\frac{\partial p}{\partial y} + \frac{1}{Re}(\frac{\partial^2 v}{\partial x^2} + \frac{\partial^2 v}{\partial y^2}). \tag{3.62}$$

We richten ons nu op de discretisatie van de niet lineaire term in (3.61) op hetzelfde verschoven rooster als in sectie 3.2. De term $\frac{\partial (u^2)}{\partial x} + \frac{\partial (uv)}{\partial y}$ wordt geïntegreerd over het verschoven controlevolume met volume ΔV
in figuur 3.5.

$$\int_{\Delta V} [\frac{\partial (u^2)}{\partial x} + \frac{\partial (uv)}{\partial y}] dV = h_2(u_F^2 - u_H^2) + h_1((uv)_G - (uv)_E). \tag{3.63}$$

Omdat de u en v niet gegeven zijn op E, F, G, H, wordt een verdere benadering verkregen door toepassing van een combinatie van centrale differenties en een upwind schema, ook wel hybrid schema genoemd, die we in de volgende paragraaf zullen bespreken (voor meer details verwijzen we naar [1]).

3.3.1 Hybrid schema

Voor het benaderen van de snelheid u_F definieren we het rooster Pecletgetal door

$$Pe_{1,i+1/2,j} = |u_{i+1/2,j}|h_1/\nu, \tag{3.64}$$

waarin

$$u_{i+1/2,j} = \frac{1}{2}(u_{ij} + u_{i+1,j}). \tag{3.65}$$

Definieer $s_{1,i+1/2,j}$ door

$$s_{1,i+1/2,j} = 1 \quad \text{als} \quad u_{i+1/2,j} > 0, \tag{3.66}$$

$$s_{1,i+1/2,j} = -1 \quad \text{als} \quad u_{i+1/2,j} < 0. \tag{3.67}$$

Het hybrid schema voor u_F^2 wordt als volgt gedefinieerd

$$u_F^2 = \frac{1}{2}(u_{ij}^2 + u_{i+1,j}^2) \quad \text{als} \quad Pe_{1,i+1/2,j} \leq 2, \tag{3.68}$$

$$u_F^2 = \frac{1}{2}[(1 + s_{1,i+1/2,j})u_{ij}^2 + (1 - s_{1,i+1/2,j})u_{i+1,j}^2] \quad \text{als} \quad Pe_{1,i+1/2,j} > 2. \tag{3.69}$$

De term u_H^2 wordt op dezelfde manier als u_F^2 benaderd. De term $(uv)_G$ wordt op de volgende wijze benaderd

$$(uv)_G = u_G v_{i-1/2,j+1}, \tag{3.70}$$

waarin

$$v_{i-1/2,j+1} = \frac{1}{2}(v_C + v_D) = \frac{1}{2}(v_{i,j+1} + v_{i-1,j+1}), \tag{3.71}$$

Definieer het rooster Pecletgetal

$$Pe_{2,i-1/2,j+1} = |v_{i-1/2,j+1}|h_2/\nu, \tag{3.72}$$

en het teken van $v_{i-1/2,j+1}$ door $s_{2,i-1/2.j+1}$. De term u_G wordt met het volgende hybride schema benaderd:

$$u_G = \frac{1}{2}(u_{ij} + u_{i,j+1}) \quad \text{als} \quad Pe_{2,i-1/2,j+1} \leq 2, \tag{3.73}$$

$$u_G = \frac{1}{2}[(1 + s_{2,i-1/2,j+1})u_{ij} + (1 - s_{2,i-1/2,j+1})u_{i,j+1}] \quad \text{als} \quad Pe_{2,i-1/2,j+1} > 0. \tag{3.74}$$

Het upwind schema wordt niet op v toegepast maar op u, omdat voor de Navier-Stokes vergelijking in de x-richting u als onbekende genomen wordt, terwijl v als een bekende coëfficient beschouwd wordt.

Linearisatie van de convectieve term

In de iteratieve oplossingsmethoden moet men de convectieve termen lineariseren. Dat wil zeggen dat de uitdrukkingen voor u_F^2 vervangen moet worden door (de superscript k duidt de k-de iteratiestap aan)

$$u_F^2 = \frac{1}{2}(u_{ij}^k u_{ij}^{k+1} + u_{i+1,j}^k u_{i+1,j}^{k+1}) \quad \text{als} \quad Pe_{1,i+1/2,j} \leq 2, \tag{3.75}$$

$$u_F^2 = \frac{1}{2}[(1+s_{1,i+1/2,j})u_{ij}^k u_{ij}^{k+1} + (1-s_{1,i+1/2,j})u_{i+1,j}^k u_{i+1,j}^{k+1}] \quad \text{als} \quad Pe_{1,i+1/2,j} > 2, \tag{3.76}$$

en $(uv)_G$ door

$$(uv)_G = \frac{1}{2}v_{i-1/2,j}^k[u_{ij}^{k+1} + u_{i,j+1}^{k+1}] \quad \text{als} \quad Pe_{2,i-1/2,j+1} \leq 2, \tag{3.77}$$

$$(uv)_G = \frac{1}{2}v_{i-1/2,j}^k[(1+s_{2,i-1/2,j+1})u_{ij}^{k+1} + (1-s_{2,i-1/2,j+1})u_{i,j+1}^{k+1}] \quad \text{als} \quad Pe_{2,i-1/2,j+1} > 0, \tag{3.78}$$

De convectieve term (bijvoorbeeld: u_F^2) kan ook op de volgende wijze gelineariseerd worden:

$$u_F^2 = \frac{1}{2}(u_{ij}^N u_{ij}^{k+1} + u_{i+1,j}^N u_{i+1,j}^{k+1}) \quad \text{als} \quad Pe_{1,i+1/2,j} \leq 2, \tag{3.79}$$

$$u_F^2 = \frac{1}{2}[(1+s_{1,i+1/2,j})u_{ij}^N u_{ij}^{k+1} + (1-s_{1,i+1/2,j})u_{i+1,j}^N u_{i+1,j}^{k+1}] \quad \text{als} \quad Pe_{1,i+1/2,j} > 2, \tag{3.80}$$

waarbij u_{ij}^N de nieuwste waarde van u_{ij} in het gegeven roosterpunt aanduidt. De Navier-Stokes bewegingsvergelijking in de y-richting wordt op een analoge manier gediscretiseerd.

3.3.2 Samenvatting van de gediscretiseerde vergelijkingen

De gediscretiseerde Navier-Stokes vergelijkingen kunnen we eenvoudigweg in de volgende vorm schrijven

$$\mathbf{Q}_1(\underline{u}^k)u^{k+1} + \mathbf{G}_1 p^{k+1} = b_1, \tag{3.81}$$

$$\mathbf{Q}_2(\underline{u}^k)v^{k+1} + \mathbf{G}_2 p^{k+1} = b_2, \tag{3.82}$$

$$\mathbf{G}_1^* u^{k+1} + \mathbf{G}_2^* v^{k+1} = b_3, \tag{3.83}$$

waarin $\underline{u} = \begin{pmatrix} u \\ v \end{pmatrix}$. De matrices $\mathbf{Q}_1$ en $\mathbf{Q}_2$ hangen dus van de snelheidscomponenten u en v af. Het stelsel bestaand uit (3.81), (3.82) en (3.83) kan samengevat worden door

$$\mathbf{A}x = b, \tag{3.84}$$

waarbij

$$\mathbf{A} = \begin{pmatrix} \mathbf{Q}_1 & \mathbf{0} & \mathbf{G}_1 \\ \mathbf{0} & \mathbf{Q}_2 & \mathbf{G}_2 \\ \mathbf{G}_1^* & \mathbf{G}_2^* & \mathbf{0} \end{pmatrix} \tag{3.85}$$

Lijst van behandelde onderwerpen:

- Discretisatie
  - Discretisatie van de convectie-diffusie vergelijking
  - Discretisatie van de Stokes vergelijkingen
    - Discretisatie van de continuïteitsvergelijking
  - Discretisatie van de Navier-Stokes vergelijkingen
    - Hybrid schema
    - Samenvatting van de gediscretiseerde vergelijkingen

4 Methoden voor het bepalen van de numerieke oplossing

De verkregen stelsels bij het discretiseren van de Stokes en Navier-Stokes vergelijkingen (zie (3.56) en (3.84)) zijn van de vorm $\mathbf{A}x = b$, waarvoor de matrix $\mathbf{A}$ de volgende vorm heeft

$$\mathbf{A} = \begin{pmatrix} \mathbf{Q}_1 & \mathbf{0} & \mathbf{G}_1 \\ \mathbf{0} & \mathbf{Q}_2 & \mathbf{G}_2 \\ \mathbf{G}_1^* & \mathbf{G}_2^* & \mathbf{0} \end{pmatrix} \tag{4.1}$$

Een geschikte splitsing van $\mathbf{A}$ van de vorm $\mathbf{A} = \mathbf{M} - \mathbf{N}$ zodat $\mathbf{M}$ eenvoudig inverteerbaar is, is niet mogelijk door de aanwezigheid van de nul-blokmatrix op de hoofddiagonaal. Bovendien hangt voor de Navier-Stokes vergelijkingen de matrix $\mathbf{A}$ van de snelheidscomponenten af. Voor het oplossen van deze stelsels wordt in WISH3D de iteratieve methode SIMPLE of SIMPLER gebruikt. Aan de hand van de distributieve iteratie methode (zie Wesseling [1]) zullen we de SIMPLE methode beschrijven.

Distributieve iteratie

In plaats van het oplossen van $\mathbf{A}x = b$ men kan ook het volgende stelsel oplossen

$$\mathbf{AB}y = b, \qquad x = \mathbf{B}y. \tag{4.2}$$

De matrix $\mathbf{B}$ wordt een postconditionering van $\mathbf{A}$ genoemd, en wordt gekozen zodat de splitsing $\mathbf{AB} = \mathbf{M} - \mathbf{N}$ leidt tot een convergente iteratieve methode. De iteratieve methode ziet er dan als volgt uit

$$x^{k+1} = x^k + \mathbf{BM}^{-1}(b - \mathbf{A}x^k). \tag{4.3}$$

4.1 SIMPLE methode

De gediscretiseerde Navier-Stokes vergelijkingen zien er als volgt uit (zie (3.84))

$$\mathbf{Q}_1 u + \mathbf{G}_1 p = b_1 \tag{4.4}$$

$$\mathbf{Q}_2 v + \mathbf{G}_2 p = b_2 \tag{4.5}$$

$$\mathbf{G}_1^* u + \mathbf{G}_2^* v = b_3 \tag{4.6}$$

Een oplossing van (4.4) en (4.5) is alleen mogelijk als de druk of een benadering daarvan bekend is. Uitgaand van een benadering p^* van de druk leveren de vergelijkingen (4.4) en (4.5) een oplossing u^* en v^* die voldoen aan

$$\mathbf{Q}_1 u^* + \mathbf{G}_1 p^* = b_1, \tag{4.7}$$
$$\mathbf{Q}_2 v^* + \mathbf{G}_2 p^* = b_2. \tag{4.8}$$

Gezocht wordt naar een manier om te zorgen dat de resulterende snelheden steeds aan de continuïteitsvergelijking voldoen. We nemen aan dat de correcte druk p verkregen wordt door $p = p^* + \delta p$, waarin δp de drukcorrectie genoemd wordt. Hetzelfde geldt voor de snelheidscomponenten, dat wil zeggen

$$u = u^* + \delta u, \tag{4.9}$$
$$v = v^* + \delta v. \tag{4.10}$$

Als we (4.7) van (4.4) aftrekken dan krijgen we de volgende vergelijking voor δu

$$\mathbf{Q}_1 \delta u + \mathbf{G}_1 \delta p = 0. \tag{4.11}$$

Op dezelfde manier volgt een vergelijking voor δv

$$\mathbf{Q}_2 \delta v + \mathbf{G}_2 \delta p = 0. \tag{4.12}$$

Om δu en δv in (4.11) en (4.12)te bepalen wordt in de SIMPLE aanpak de volgende benadering gemaakt (voor meer details zie [2])

$$\delta u = -\mathbf{D}_1^{-1} \mathbf{G}_1 \delta p, \tag{4.13}$$
$$\delta v = -\mathbf{D}_2^{-1} \mathbf{G}_2 \delta p, \tag{4.14}$$

waarbij

$$\mathbf{D}_1 = \mathrm{Diag}(\mathbf{Q}_1), \tag{4.15}$$
$$\mathbf{D}_2 = \mathrm{Diag}(\mathbf{Q}_2). \tag{4.16}$$

Door substitutie van u en v respectievelijk gegeven door (4.9) en (4.10) in de vergelijking (4.6), gebruikmakend van de uitdrukkingen (4.13) en (4.14) volgt

$$(-\mathbf{G}_1^* \mathbf{D}_1^{-1} \mathbf{G}_1 - \mathbf{G}_2^* \mathbf{D}_2^{-1} \mathbf{G}_2) \delta p = b_3 - \mathbf{G}_1^* u^* - \mathbf{G}_2^* v^*. \tag{4.17}$$

We hebben alle vergelijkingen die nodig zijn voor het bepalen van de snelheden en de druk geformuleerd. Nu zijn we in staat om het SIMPLE algoritme te beschrijven.

SIMPLE algoritme

1. kies een beginschatting p^*

2. los op $\mathbf{Q}_1 u^* + \mathbf{G}_1 p^* = b_1$
3. los op $\mathbf{Q}_2 v^* + \mathbf{G}_2 p^* = b_2$
4. los op $(-\mathbf{G}_1^* \mathbf{D}_1^{-1} \mathbf{G}_1 - \mathbf{G}_2^* \mathbf{D}_2^{-1} \mathbf{G}_2)\delta p = b_3 - \mathbf{G}_1^* u^* - \mathbf{G}_2^* v^*$

5. bereken $p := p^* + \delta p$
6. bereken $u := u^* - \mathbf{D}_1^{-1} \mathbf{G}_1 \delta p$
   bereken $v := v^* - \mathbf{D}_2^{-1} \mathbf{G}_2 \delta p$

7. neem $p^* := p$ als nieuwe beginschatting en herhaal dit proces vanaf stap 2 tot convergentie

De SIMPLE methode kunnen we ook afleiden door het postconditioneren van de matrix $\mathbf{A}$ door een matrix $\mathbf{B}$ zodat een geschikte splitsing van de vorm $\mathbf{AB} = \mathbf{M} - \mathbf{N}$ mogelijk is. De SIMPLE methode kan verkregen worden door de keuze

$$\mathbf{B} = \begin{pmatrix} \mathbf{I} & \mathbf{0} & -\mathbf{D}_1^{-1}\mathbf{G}_1 \\ \mathbf{0} & \mathbf{I} & -\mathbf{D}_2^{-1}\mathbf{G}_2 \\ \mathbf{0} & \mathbf{0} & \mathbf{I} \end{pmatrix}. \tag{4.18}$$

Dit levert met $\mathbf{R} = -\mathbf{G}_1^* \mathbf{D}_1^{-1} \mathbf{G}_1 - \mathbf{G}_2^* \mathbf{D}_2^{-1} \mathbf{G}_2$

$$\mathbf{AB} = \begin{pmatrix} \mathbf{Q}_1 & \mathbf{0} & \mathbf{G}_1 - \mathbf{Q}_1\mathbf{D}_1^{-1}\mathbf{G}_1 \\ \mathbf{0} & \mathbf{Q}_2 & \mathbf{G}_2 - \mathbf{Q}_2\mathbf{D}_2^{-1}\mathbf{G}_2 \\ \mathbf{G}_1^* & \mathbf{G}_2^* & \mathbf{R} \end{pmatrix}. \tag{4.19}$$

De matrix $\mathbf{M}$ wordt dan als volgt gedefinieerd

$$\mathbf{M} = \begin{pmatrix} \mathbf{Q}_1 & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{Q}_2 & \mathbf{0} \\ \mathbf{G}_1^* & \mathbf{G}_2^* & \mathbf{R} \end{pmatrix}. \tag{4.20}$$

Gebruikmakend van de distributieve iteratie methode (4.3), wordt het stelsel (3.84) opgelost met het volgende iteratieve proces

$$x^{k+1} = x^k + \mathbf{BM}^{-1}(b - \mathbf{A}x^k) \qquad k = 0, 1, 2, ... \tag{4.21}$$

Dit is equivalent met het oplossen van

for $k = 0, 1, ...$until convergence **do**
  $r^k = b - \mathbf{A}x^k$
  $\mathbf{M}\delta\tilde{x} = r^k$
  $\delta x = \mathbf{B}\delta\tilde{x}$
  $x^{k+1} = x^k + \delta x$
end for

Resumerend komen we tot het SIMPLE algoritme:

Algoritme 4.1

p^0 =beginschatting,
$u^0 = 0$
$v^0 = 0$

for $k = 0, 1, 2...$ until convergence **do**

/*** $r^k = b - \mathbf{A}x^k$ ***/
a. $r_1^k = b_1 - \mathbf{Q}_1 u^k - \mathbf{G}_1 p^k$
b. $r_2^k = b_2 - \mathbf{Q}_2 v^k - \mathbf{G}_2 p^k$
c. $r_3^k = b_3 - \mathbf{G}_1^* u^k - \mathbf{G}_2^* v^k$

/*** $\mathbf{M}\delta\tilde{x} = r^k$ ***/

d. los op $\mathbf{Q}_1 \delta\tilde{u} = r_1^k$
e. los op $\mathbf{Q}_2 \delta\tilde{v} = r_2^k$
f. los op $\mathbf{R}\delta\tilde{p} = r_3^k - \mathbf{G}_1^* \delta\tilde{u} - \mathbf{G}_2^* \delta\tilde{v}$

/*** $\delta x = \mathbf{B}\delta\tilde{x}$ ***/

g. $\delta u = \delta\tilde{u} - \mathbf{D}_1^{-1} \mathbf{G}_1 \delta\tilde{p}$
h. $\delta v = \delta\tilde{v} - \mathbf{D}_2^{-1} \mathbf{G}_2 \delta\tilde{p}$
i. $\delta p = \delta\tilde{p}$

/*** $x^{k+1} = x^k + \delta x$ ***/

j. $u^{k+1} := u^k + \delta u$
k. $v^{k+1} := v^k + \delta v$
l. $p^{k+1} := p^k + \delta p$

end for

Merk op dat Algoritme 4.1 en het SIMPLE algoritme identiek zijn. Immers, beschouw Algoritme 4.1 en substitueer (d), (e) en (f) in (a), (b) en (c). Gebruik de notatie $u^* := u^k + \delta\tilde{u}$, $v^* := v^k + \delta\tilde{v}$ en $p^* := p^k$ en substitueer vervolgens (g), (h) en (i) in (j), (k) en (l) dan volgt het SIMPLE algoritme.

In de praktijk worden de stelsels in de stappen d), e) en f) niet exact opgelost maar alleen benaderingen van de oplossingen van deze stelsels worden bepaald. Deze stelsels zijn in de volgende vorm te schrijven:

$$\phi_P = (\sum_{nb} a_{nb}\phi_{nb} + b)/a_P, \tag{4.22}$$

waarbij de sommatie uitgevoerd wordt over de buurpunten van het roosterpunt P. Dit kunnen we ook in de volgende matrix vorm schrijven:

$$\mathbf{Q}\underline{\phi} = \underline{b}, \tag{4.23}$$

waarbij de matrix $\mathbf{Q}$ de vorm heeft zoals gegeven is door (3.39) en (3.40) We introduceren de volgende splitsing van de matrix $\mathbf{Q}$:

$$\mathbf{Q} = -\mathbf{R} + \mathbf{D} - \mathbf{S} \tag{4.24}$$

Verschillende iteratieve methoden kunnen gebruikt worden voor dit doel.

De gebruikte methoden om deze benaderingen te bepalen in WISH3D zijn de TDMA solvers, dat wil zeggen dat we de matrix $\mathbf{M}$ in (4.20) anders kunnen schrijven

$$\mathbf{M} = \begin{pmatrix} \mathbf{M}_1 & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{M}_2 & \mathbf{0} \\ \mathbf{G}_1^* & \mathbf{G}_2^* & \mathbf{M}_3 \end{pmatrix}, \tag{4.25}$$

waarin $\mathbf{M}_1$, $\mathbf{M}_2$ en $\mathbf{M}_3$ respectievelijk een benaderning is van $\mathbf{Q}_1$, $\mathbf{Q}_2$ en $\mathbf{R}$ afhankelijk van de gebruikte methode.

4.2 WISH3d solvers

We zullen in deze paragraaf een aantal methoden bespreken die gebruikt worden bij de oplossing van (4.23). Daarna zullen we aangeven welke matrix $\mathbf{M}$ uit (4.25) hiermee correspondeert.

We onderscheiden de volgende methoden:

4.2.1 Jacobi

Deze methode evalueert de variabele ϕ in een roosterpunt P op de volgende wijze:

$$\phi_P = (\sum_{nb} a_{nb}\phi^*_{nb} + b)/a_P, \tag{4.26}$$

waarbij ϕ^*_{nb} de waarde van ϕ_{nb} van de vorige iteratie aanduidt. Dit is equivalent met het volgende:

$$\mathbf{D}\phi^{m+1} = (\mathbf{R} + \mathbf{S})\phi^m + b \tag{4.27}$$

4.2.2 Gauss-Seidel

De variabele ϕ wordt geëvalueerd in een roosterpunt als volgt:

$$\phi_P = (\sum_{nb} a_{nb}\phi^*_{nb} + b)/a_P, \tag{4.28}$$

waarin ϕ^*_{nb} de nieuwste waarde van ϕ_{nb} voorstelt. Dit is equivalent met het volgende:

$$(\mathbf{D} - \mathbf{R})\phi^{m+1} = \mathbf{S}\phi^m + b \tag{4.29}$$

4.2.3 TDMA solvers

We maken onderscheid tussen verschillende TDMA (Tri-Diagonal-Matrix-Algoritm) solvers, namelijk de Line-, Plane- en space-TDMA. Om de TDMA solvers toe te lichten zullen we hier volstaan met de Line-TDMA solver voor het tweedimensionaal geval, de Plane- en space-TDMA kunnen op analoge manier behandeld worden.

<u>Line-TDMA</u>

Beschouw een lijn j parallel aan de x-as, zie figuur 4.1.
De onbekende vector op de lijn j duiden we aan met ϕ_j, met

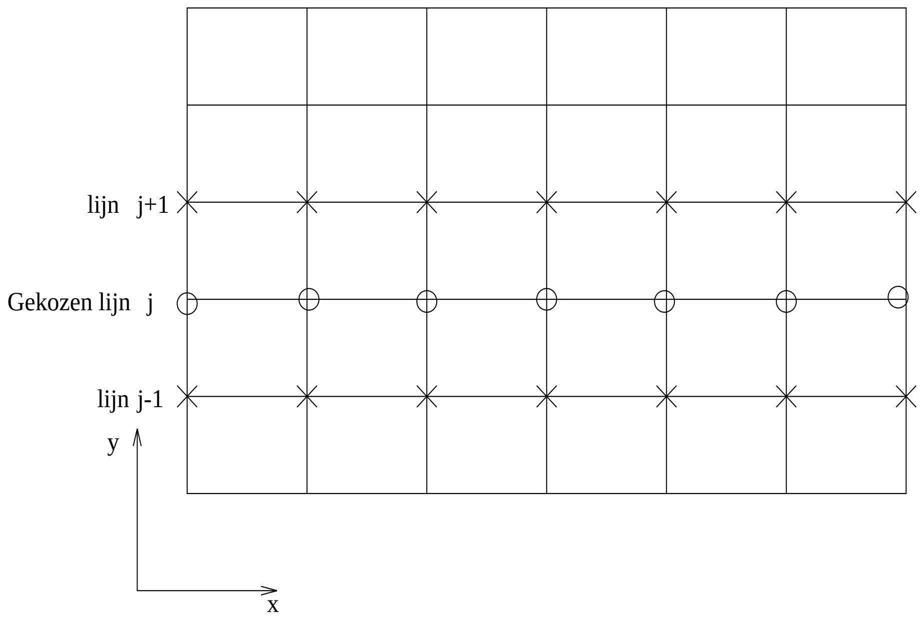
$$\phi_j = (\phi_{1j}, \phi_{2j}, \ldots, \phi_{nj})^T, \tag{4.30}$$

en

$$\phi = (\phi_1, \phi_2, \ldots, \phi_m)^T, \tag{4.31}$$

zie (3.38), waarbij n het aantal roosterpunten op de lijn j voorstelt. De waarden van ϕ, die zich niet op de lijn j bevinden, worden als bekend beschouwd. De vergelijking (4.22) kan geschreven worden als:

for $j = 1, m$ **do**
  for $i = 1, n$ **do**
    $-a^S_{ij}\phi_{i,j-1} - a^W_{ij}\phi_{i-1,j} + a^P_{ij}\phi_{i,j} - a^E_{ij}\phi_{i+1,j} - a^N_{ij}\phi_{i,j+1} = b_{ij},$
  end for
end for



Figuur 4.1: Representatie van Line-TDMA methode.

die te schrijven is als:

$$\begin{pmatrix} A_1 & N_1 & & & & \\ S_2 & A_2 & N_2 & & & \\ & \ddots & \ddots & \ddots & & \\ & & \ddots & \ddots & \ddots & \\ & & & S_{m-1} & A_{m-1} & N_{m-1} \\ & & & & S_m & A_m \end{pmatrix} \begin{pmatrix} \phi_1 \\ \phi_2 \\ \vdots \\ \vdots \\ \phi_{m-1} \\ \phi_m \end{pmatrix} = \begin{pmatrix} b_1 \\ b_2 \\ \vdots \\ \vdots \\ b_{m-1} \\ b_m \end{pmatrix} \tag{4.32}$$

De matrices $\mathbf{A}_j$, $\mathbf{S}_j$ en $\mathbf{N}_j$ zijn gegeven door (3.40) en (3.41). De diagonaal blokken $\mathbf{A}_j$ corresponderen met de roosterpunten op de gridlijn j evenwijdig aan de x-as. Het algoritme (4.2.3) is equivalent met:

for $j = 1, m$ **do**
  $\mathbf{A}_j\phi_j = \tilde{b}_j$,
end for

waarin $\tilde{b}_j = S_j\phi_{j-1} + N_j\phi_{j+1}$. Dit algoritme heet linezyx, hier spreken we over sweeps in de y-richting. Sweeps in de tegengestelde richting zijn ook mogelijk. Diverse line TDMA solvers worden gebruikt in WISH3D namelijk linezxy, lineyzx, enz... In het twee dimensionale geval speelt de z-richting geen rol. Naast de line TDMA solvers bestaan ook plane-TDMA en space-TDMA solvers.

Stel voor de eenvoud dat de methode van Jacobi zie sectie 4.2.1 wordt gebruikt en volstaan wordt met twee iteraties en een beginschating gelijk aan nul , dan ziet $\mathbf{M}$ er als volgt uit:

$$\mathbf{M} = \begin{pmatrix} (2\mathbf{I}-\mathbf{Q_1}\mathbf{D_1^{-1}})^{-1}\mathbf{D_1} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & (2\mathbf{I}-\mathbf{Q_2}\mathbf{D_2^{-1}})^{-1}\mathbf{D_2} & \mathbf{0} \\ \mathbf{G}_1^* & \mathbf{G}_2^* & (2\mathbf{I}-\mathbf{R}\mathbf{D_3^{-1}})^{-1}\mathbf{D_3} \end{pmatrix},$$

waarin $\mathbf{D}_1$ en $\mathbf{D}_2$ gegeven zijn door (4.15) en (4.16) en $\mathbf{D}_3 = \mathrm{Diag}(\mathbf{R})$.

Het algoritme 4.1 kan herschreven worden met de invoering van de nieuwe matrix $\mathbf{M}$ en dat resulteert in het volgende algoritme:

$$\begin{aligned} p^0 &= \text{beginschatting}, \\ u^0 &= 0 \\ v^0 &= 0 \end{aligned}$$

for $k = 0, 1, 2...$ until convergence **do**

$$\begin{aligned} r_1^k &= b_1 - \mathbf{Q}_1 u^k - \mathbf{G}_1 p^k \\ r_2^k &= b_2 - \mathbf{Q}_2 v^k - \mathbf{G}_2 p^k \\ r_3^k &= b_3 - \mathbf{G}_1^* u^k - \mathbf{G}_2^* v^k \end{aligned}$$

$$\begin{aligned} \delta\tilde{p} &= \mathbf{D}_3^{-1}(2\mathbf{I}-\mathbf{R}\mathbf{D_3}^{-1})\left[r_3^k - \textstyle\sum_{j=1}^2 \mathbf{G}_j^*\mathbf{D}_j^{-1}(2\mathbf{I}-\mathbf{Q}_j\mathbf{D}_j^{-1})r_j^k\right] \\ u^{k+1} &:= u^k + \mathbf{D}_1^{-1}(2\mathbf{I}-\mathbf{Q_1}\mathbf{D_1}^{-1})r_1^k - \mathbf{D}_1^{-1}\mathbf{G}_1\delta\tilde{p} \\ v^{k+1} &:= v^k + \mathbf{D}_2^{-1}(2\mathbf{I}-\mathbf{Q_2}\mathbf{D_2}^{-1})r_2^k - \mathbf{D}_2^{-1}\mathbf{G}_2\delta\tilde{p} \\ p^{k+1} &:= p^k + \delta\tilde{p} \end{aligned}$$

end for

4.3 Krylov methoden

Beschouw het lineaire stelsel $\mathbf{A}x = b, \quad \mathbf{A} \in \mathbb{R}^{n\times n}, \quad b \in \mathbb{R}^n$. Afhankelijk van de eigenschappen van de matrix $\mathbf{A}$, kunnen diverse Krylov methoden gebruikt worden. Als $\mathbf{A}$ symmetrisch positief definiet (SPD) is gebruiken we de PCG (Preconditioned Conjugate Gradient) methode. Voor lineaire stelsels waarvoor de matrix $\mathbf{A}$ niet symmetrisch is wordt in dit onderzoek gebruik gemaakt van de GCR (Generalized Conjugate Residual) methode of de gepreconditioneerde versie daarvan PGCR. Beschouw een niet singuliere matrix $\mathbf{P}^{-1}$ als preconditionering van de matrix $\mathbf{A}$, dat wil zeggen dat het stelsel $\mathbf{P}^{-1}\mathbf{A}x = \mathbf{P}^{-1}b$ wordt opgelost in plaats van $\mathbf{A}x = b$. Daarvoor wordt het volgende iteratieve proces geïntroduceerd

$$x^{k+1} = x^k + \mathbf{P}^{-1}(b - \mathbf{A}x^k), \qquad \text{voor} \quad k = 0,1,2,... \tag{4.33}$$

Voor de iteratiefout $x^{k+1} - x$ geldt:

$$x^{k+1} - x = (\mathbf{I} - \mathbf{P}^{-1}\mathbf{A})(x^k - x), \qquad \text{voor} \quad k = 0,1,2,... \tag{4.34}$$

waarbij x de exacte oplossing voorstelt.
Voer in $\mathbf{C} = \mathbf{P}^{-1}\mathbf{A}$, dan volgt

$$x^{k+1} = x + (\mathbf{I} - \mathbf{C})^{k+1}(x^0 - x), \qquad \text{voor} \quad k = 0,1,2,... \tag{4.35}$$

Definieer het residu door $r^k = b - \mathbf{A}x^k$ voor $k = 0,1,2,...$
Hiermee kan de iteratieformule herschreven worden tot

$$x^{k+1} = x - (\mathbf{I} - \mathbf{C})^{k+1}\mathbf{C}^{-1}\mathbf{P}^{-1}r^0, \qquad \text{voor} \quad k = 0,1,2,... \tag{4.36}$$

Er geldt dus

$$x^{k+1} = x - p_{k+1}(\mathbf{C})\mathbf{C}^{-1}\mathbf{P}^{-1}r^0, \tag{4.37}$$

waarbij p_k een k-de graads polynoom is met $p_k(0) = 1$. Of wel

$$x^{k+1} = x^0 + (\mathbf{I} - p_{k+1}(\mathbf{C}))\mathbf{C}^{-1}\mathbf{P}^{-1}r^0. \tag{4.38}$$

Omdat p_{k+1} een polynoom is van graad $(k+1)$ met $p_{k+1}(0) = 1$, is $(\mathbf{I} - p_{k+1}(\mathbf{C}))\mathbf{C}^{-1}$ een willekeurige polynoom van graad k. Hieruit volgt dat $x^{k+1} - x^0$ in de ruimte zit, opgespannen door de vectoren
$\{\mathbf{P}^{-1}r^0, \mathbf{C}\mathbf{P}^{-1}r^0, \ldots, \mathbf{C}^k\mathbf{P}^{-1}r^0\}$. Dit heet de Krylov ruimte van dimensie $(k+1)$ en noteren we met $\mathbf{K}^{k+1}\{\mathbf{C}, \mathbf{P}^{-1}r^0\}$. Merk op dat het inwendige product $(.,.)$, welke in het vervolg van dit onderzoek gebruikt wordt, gegeven wordt door

$$\|x\|_2^2 = (x,x), \quad x \in \mathbb{R}^n. \tag{4.39}$$

4.3.1 PCG methode

Neem nu aan dat de matrix $\mathbf{A}$ en de preconditioneringsmatrix $\mathbf{P}^{-1}$ symmetrisch en positief definiet zijn. Gegeven een ϵ $(0 < \epsilon \ll 1)$ en een beginschatting x^0. Het volgende algoritme berekent een vector x zodat $\|b - \mathbf{A}x\|_2 \approx \epsilon \|r^0\|_2$.

Algoritme 4.3.1

$r^0 = b - \mathbf{A}x^0$
$p^0 = \mathbf{P}^{-1} r^0$
$z^0 = p^0$
while $(\|r^k\|_2 > \epsilon \|r^0\|_2)$ **do**
$\quad \alpha^k = \dfrac{(r^k, z^k)}{(\mathbf{A}p^k, p^k)}$
$\quad x^{k+1} = x^k + \alpha^k p^k$
$\quad r^{k+1} = r^k - \alpha^k \mathbf{A} p^k$
$\quad z^{k+1} = \mathbf{P}^{-1} r^{k+1}$
$\quad \beta^{k+1} = \dfrac{(r^{k+1}, z^{k+1})}{(r^k, z^k)}$
$\quad p^{k+1} = z^{k+1} + \beta^{k+1} p^k$
end while

4.3.2 GCR methode

In deze sectie introduceren we de GCR (Generalized Conjugate Residuals) methode (zie [7]) voor het oplossen van lineaire stelsels waarvoor de matrix $\mathbf{A}$ niet symmetrisch is. Het GCR algoritme kan in het kort beschreven worden door

$r^0 = b - \mathbf{A}x^0, \quad s^0 = r^0$
for $k = 0, 1, \ldots$ **do**
$\quad \alpha^k = \dfrac{(r^k, \mathbf{A}s^k)}{(\mathbf{A}s^k, \mathbf{A}s^k)}$
$\quad x^{k+1} = x^k + \alpha^k s^k$
$\quad r^{k+1} = r^k - \alpha^k \mathbf{A} s^k$
$\quad$ **for** $i = 0, 1, 2, \ldots k$ **do**
$\quad\quad \beta^i_{(k+1)} = -\dfrac{(\mathbf{A}r^{k+1}, \mathbf{A}s^i)}{(\mathbf{A}s^i, \mathbf{A}s^i)}$
$\quad$ **end for**
$\quad s^{k+1} = r^{k+1} + \sum_{i=0}^{k} \beta^i_{(k+1)} s^i$
end for

We zullen nu een variant van het bovenstaande GCR algoritme beschouwen (voor meer details zie [12]). Algoritme GCR

$r^0 = b - \mathbf{A}x^0$
for $k = 0, 1, 2, \ldots$ **do**
$s^{k+1} = r^k$
$v^{k+1} = \mathbf{A}s^{k+1}$
for $i = 1, 2, \ldots k$ **do**
$\alpha = (v^{k+1}, v^i)$
$v^{k+1} := v^{k+1} - \alpha v^i$
$s^{k+1} := s^{k+1} - \alpha s^i$
end for
$v^{k+1} := v^{k+1}/\|v^{k+1}\|_2$
$s^{k+1} := s^{k+1}/\|v^{k+1}\|_2$
$x^{k+1} := x^k + (r^k, v^{k+1})s^{k+1}$
$r^{k+1} := r^k - (r^k, v^{k+1})v^{k+1}$
end for

Beschouw een niet singulier preconditioneringsmatrix $\mathbf{P}^{-1}$. In het bovenstaande GCR algoritme kiezen we de zoekrichting $s^{k+1} = \mathbf{P}^{-1}r^k$. Dit leidt tot Het PGCR (Preconditioned GCR) algoritme.

Algoritme 4.3.2

$r^0 = b - \mathbf{A}x^0$
for $k = 0, 1, 2, \ldots$ **do**
$s^{k+1} = \mathbf{P}^{-1}r^k$
$v^{k+1} = \mathbf{A}s^{k+1}$
for $i = 1, 2, \ldots k$ **do**
$\alpha = (v^{k+1}, v^i)$
$v^{k+1} := v^{k+1} - \alpha v^i$
$s^{k+1} := s^{k+1} - \alpha s^i$
end for
$v^{k+1} := v^{k+1}/\|v^{k+1}\|_2$
$s^{k+1} := s^{k+1}/\|v^{k+1}\|_2$
$x^{k+1} := x^k + (r^k, v^{k+1})s^{k+1}$
$r^{k+1} := r^k - (r^k, v^{k+1})v^{k+1}$
end for

Merk op dat de vectoren s^k en v^k in het geheugen opgeslagen moeten worden. Verder neemt de hoeveelheid werk in het Gram-Schmidt proces kwadratisch toe met het aantal iteraties. Om dit te verlagen kan men gebruik maken van de volgende technieken:

- Trunkeren: Er wordt gebruik gemaakt van slechts *ntrunc* zoekrichtingen. Als het aantal iteraties groter is dan *ntrunc* dan wordt een oude zoekrichting overschreven.

- Herstarten: Een ander alternatief is het herstarten van het GCR algoritme na k_s iteraties. Alle vectoren s^k en v^k worden verwijderd en het algoritme wordt herstart met een nieuwe beginschatting $x^0 = x^{k_s}$.

4.4 SIMPLE voor Stokes versnellen met GCR

De SIMPLE methode is een Gauss-Jacobi-achtige methode toegepast op het stelsel (4.40). Bekend is dat de Krylov versnelling vaak veel beter convergeert, dat die goedkoop is, dat wil zeggen het extra werk dat uitgevoerd moet worden door het gebruik van de Krylov versnelling (bv: orthogonalisatie) verwaarloosbaar is. Bovendien hebben de langzaam convergerende termen na verloop van de tijd geen invloed meer (superlineaire convergentie), en meestal is geen relaxatie nodig. Dit zijn redenen om het SIMPLE algoritme te versnellen met een Krylov methode.
Door een combinatie van de GCR methode en het SIMPLE algoritme zullen we proberen het SIMPLE algoritme te versnellen. Het probleem

$$\mathbf{A}x = b, \tag{4.40}$$

wordt met behulp van de volgende distributieve iteratie methode opgelost

$$x^{k+1} = x^k + \mathbf{B}\mathbf{M}^{-1}(b - \mathbf{A}x^k), \tag{4.41}$$

waarbij $\mathbf{B}$ gegeven is door (4.18) en $\mathbf{M}$ door (4.20). De GCR methode gegeven door Algoritme 4.3.2 wordt nu op (4.41) toegepast. Er wordt voor de zoekrichting s^{k+1} de keuze $s^{k+1} = \mathbf{B}\mathbf{M}^{-1}r^k$ gemaakt. Hieruit volgt
Algoritme GCR-SIMPLE

$r^0 = b - \mathbf{A}x^0$
for $k = 0, 1, 2, \dots$ **do**
  $\mathbf{M}z^{k+1} = r^k$
  $s^{k+1} = \mathbf{B}z^{k+1}$
  $v^{k+1} = \mathbf{A}s^{k+1}$
  for $i = 1, 2, ...k$ **do**
    $\alpha = (v^{k+1}, v^i)$
    $v^{k+1} := v^{k+1} - \alpha v^i$
    $s^{k+1} := s^{k+1} - \alpha s^i$
  end for
  $v^{k+1} := v^{k+1}/\|v^{k+1}\|_2$
  $s^{k+1} := s^{k+1}/\|v^{k+1}\|_2$
  $x^{k+1} := x^k + (r^k, v^{k+1})s^{k+1}$
  $r^{k+1} := r^k - (r^k, v^{k+1})v^{k+1}$
end for

In het vervolg zullen we gebruik maken van de volgende definities:
De vector x wordt vervangen door $\begin{pmatrix} u \\ v \\ p \end{pmatrix}$. Voor het gemak voeren we de volgende transformatie: $u_1 := u, \quad u_2 := v$.
Definieer:
$r = \begin{pmatrix} r_1 \\ r_2 \\ r_3 \end{pmatrix}$, $s = \begin{pmatrix} s_1 \\ s_2 \\ s_3 \end{pmatrix}$, $v = \begin{pmatrix} v_1 \\ v_2 \\ v_3 \end{pmatrix}$ en $z = \begin{pmatrix} z_1 \\ z_2 \\ z_3 \end{pmatrix}$. Dan volgt het GCR-SIMPLE algoritme:
Algoritme 4.4 (GCR-SIMPLE)

kies een beginschatting p^0,
$u_1^0 = 0$,
$u_2^0 = 0$,

$r_j^0 = b_j - \mathbf{Q}_j u_j^0 - \mathbf{G}_j p^0, \qquad j = 1,2$
$r_3^0 = b_3 - \mathbf{G}_1^* u_1^0 - \mathbf{G}_2^* u_2^0$

for $k = 0, 1, 2, \ldots, ngcr$ **do**

los op $\mathbf{Q}_j z_j^{k+1} = r_j^k, \qquad j = 1,2$
los op $\mathbf{R} z_3^{k+1} = r_3^k - \mathbf{G}_1^* z_1^{k+1} - \mathbf{G}_2^* z_2^{k+1}$

$s_j^{k+1} = z_j^{k+1} - \mathbf{D}_j^{-1} \mathbf{G}_j z_3^{k+1}, \qquad j = 1,2$
$s_3^{k+1} = z_3^{k+1}$

$v_j^{k+1} = \mathbf{Q}_j s_j^{k+1} + \mathbf{G}_j s_3^{k+1}, \qquad j = 1,2$
$v_3^{k+1} = \mathbf{G}_1^* s_1^{k+1} + \mathbf{G}_2^* s_2^{k+1}$

/*** begin Gram-Schmidt orthogonalisatie proces ***/
for $i = 1, 2, ...k$ **do**
$\alpha = \sum_{j=1}^{3} (v_j^{k+1}, v_j^i)$
$v_j^{k+1} := v_j^{k+1} - \alpha v_j^i, \qquad j = 1,2,3$
$s_j^{k+1} := s_j^{k+1} - \alpha s_j^i, \qquad j = 1,2,3$
end for
/*** end Gram-Schmidt orthogonalisatie proces ***/

$\beta = \sqrt{\sum_{j=1}^{3} (v_j^{k+1}, v_j^{k+1})}$
$v_j^{k+1} := v_j^{k+1} / \beta, \qquad j = 1,2,3$
$s_j^{k+1} := s_j^{k+1} / \beta, \qquad j = 1,2,3$
$\gamma = \sum_{j=1}^{3} (r_j^k, v_j^{k+1})$
$u_j^{k+1} := u_j^k + \gamma s_j^{k+1}, \qquad j = 1,2$
$p^{k+1} := p^k + \gamma s_3^{k+1}$
$r_j^{k+1} := r_j^k - \gamma v_j^{k+1}, \qquad j = 1,2,3$

end for

4.5 Matrix-schaling

In de vergelijking (3.8) is het concept van de bronterm linearisatie geïntroduceerd:

$$S_\phi = S_C + S_P \phi_P, \tag{4.42}$$

voor meer details verwijzen we naar [2]. We volstaan hier door te noemen dat een positieve S_P tot divergentie van de numerieke oplosmethoden kan leiden. Met andere woorden, het principe van een negatieve S_P is van belang voor de convergentie van de oplosmethoden. Bovendien, om de randvoorwaarden te verwerken wordt gebruik gemaakt van het volgende:

$$S_C = 10^{25} \phi_r and, \tag{4.43}$$

$$S_P = -10^{25}, \tag{4.44}$$

waarin 10^{25} groot genoeg is om de andere termen verwaarloosbaar te maken. En dat resulteert in:

$$S_C + S_P \phi_P \approx 0, \tag{4.45}$$

en dit levert

$$\phi_P = -\frac{S_C}{S_P} = \phi_r and, \tag{4.46}$$

In vergelijking (3.34) voor a_p en (3.35) voor b zien we dat de term S_P respektievelijk S_C voorkomt. Als we GCR-SIMPLE toepassen zien we een slechte convergentie en soms divergentie van het algoritme. Vandaar dat we het GCR-SIMPLE algoritme toepassen op de geschaalde vergelijkingen. Als we diagonaalschaling toepassen resulteert dit met $\mathbf{D} = \mathrm{Diag}(\mathbf{AB})$ in het volgende algoritme:

$r^0 = \mathbf{D}^{-1}(b - \mathbf{A}x^0)$
for $k = 0, 1, 2, \ldots$ **do**
  $\mathbf{D}^{-1}\mathbf{M}z^{k+1} = r^k$
  $s^{k+1} = \mathbf{B}z^{k+1}$
  $v^{k+1} = \mathbf{D}^{-1}\mathbf{A}s^{k+1}$
  for $i = 1, 2, \ldots k$ **do**
    $\alpha = (v^{k+1}, v^i)$
    $v^{k+1} := v^{k+1} - \alpha v^i$
    $s^{k+1} := s^{k+1} - \alpha s^i$
  end for
  $v^{k+1} := v^{k+1}/\|v^{k+1}\|_2$
  $s^{k+1} := s^{k+1}/\|v^{k+1}\|_2$
  $x^{k+1} := x^k + (r^k, v^{k+1})s^{k+1}$
  $r^{k+1} := r^k - (r^k, v^{k+1})v^{k+1}$
end for

In principe wijzigt de diagonaal-schaling niets aan het GCR-SIMPLE algoritme, toch zien we een betere convergentie van het algoritme. Deze verbetering is te danken aan twee redenen namelijk:

1. De diagonaal-schaling leidt tot een beter gedrag ten opzichte van de afrondfouten.
2. Toepassing van de TDMA solvers op $\mathbf{D}^{-1}\mathbf{M}z^{k+1} = \mathbf{D}^{-1}r^k$ in het GCR-SIMPLE algoritme levert een betere convergentie dan toepassing op $\mathbf{M}z^{k+1} = r^k$

Het bovenstaande algoritme kan herschreven worden met $\mathbf{D_3} = \mathrm{Diag}(\mathbf{R})$ in het volgende algoritme:

kies een beginschatting p^0,
$u_1^0 = 0$
$u_2^0 = 0$,
$r_j^0 = \mathbf{D}_j^{-1}(b_j - \mathbf{Q}_j u_j^0 - \mathbf{G}_j p^0), \quad j = 1,2$
$r_3^0 = \mathbf{D}_3^{-1}(b_3 - \mathbf{G}_1^* u_1^0 - \mathbf{G}_2^* u_2^0)$
for $k = 0,1,2,\ldots,ngcr$ **do**
los op $\mathbf{D}_j^{-1}\mathbf{Q}_j z_j^{k+1} = r_j^k, \quad j = 1,2$
los op $\mathbf{D}_3^{-1}\mathbf{R} z_3^{k+1} = r_3^k - \mathbf{D}_3^{-1}(\mathbf{G}_1^* z_1^{k+1} - \mathbf{G}_2^* z_2^{k+1})$

$s_j^{k+1} = z_j^{k+1} - \mathbf{D}_j^{-1}\mathbf{G}_j z_3^{k+1}, \quad j = 1,2$
$s_3^{k+1} = z_3^{k+1}$
$v_j^{k+1} = \mathbf{D}_j^{-1}\mathbf{Q}_j s_j^{k+1} + \mathbf{D}_j^{-1}\mathbf{G}_j s_3^{k+1}, \quad j = 1,2$
$v_3^{k+1} = \mathbf{D}_3^{-1}\mathbf{G}_1^* s_1^{k+1} + \mathbf{D}_3^{-1}\mathbf{G}_2^* s_2^{k+1}$

for $i = 1,2,...k$ **do**
$\alpha = \sum_{j=1}^{3}(v_j^{k+1}, v_j^i)$
$v_j^{k+1} := v_j^{k+1} - \alpha v_j^i, \quad j = 1,2,3$
$s_j^{k+1} := s_j^{k+1} - \alpha s_j^i, \quad j = 1,2,3$
end for

$\beta = \sqrt{\sum_{j=1}^{3}(v_j^{k+1}, v_j^{k+1}}$
$v_j^{k+1} := v_j^{k+1}/\beta, \quad j = 1,2,3$
$s_j^{k+1} := s_j^{k+1}/\beta, \quad j = 1,2,3$

$\gamma = \sum_{j=1}^{3}(r_j^k, v_j^{k+1})$
$u_j^{k+1} := u_j^k + \gamma s_j^{k+1}, \quad j = 1,2$
$p^{k+1} := p^k + \gamma s_3^{k+1}$
$r_j^{k+1} := r_j^k - \gamma v_j^{k+1}, \quad j = 1,2,3$
end for

4.6 SIMPLE voor Navier-Stokes versnellen met GCR

In het stelsel (3.84) hangt de matrix $\mathbf{A}$ van de snelheidscomponenten af, zoals aangegeven in (3.81) en(3.82). De aanpak is dat (4.47) wordt opgelost met een iteratief proces waarin alleen lineaire stelsels hoeven te worden opgelost (iteratiemethode van Picard). Dat is equivalent met het oplossen van stelsels van de vorm

$$\mathbf{A}(x^i)x^{i+1} = b. \tag{4.47}$$

Deze lineaire stelsels worden opgelost met de reeds behandelde methode in Algoritme 4.4. Uitgaande van een beginschatting x^0 wordt het stelsel $\mathbf{A}(x^0)x^1 = b$ met behulp van Algoritme 4.4 opgelost en de verkregen oplossing x^1 wordt gebruikt om een nieuwe matrix $\mathbf{A}(x^1)$ op te bouwen. Dit proces wordt herhaald totdat een oplossing is gevonden zodanig dat $\mathbf{A}(x^{i+1})x^{i+1} = b$. Dan komen we tot een algoritme voor het oplossen van de Navier-Stokes vergelijkingen:

Algoritme 4.6

kies een beginschatting x^0,
for $i = 0, 1, 2 \ldots, niter$ **do**
  pas Algoritme 4.4 toe op $\mathbf{A}(x^i)x^{i+1} = b$
end for

In het vervolg van dit verslag zullen we gebruik maken van de volgende notaties:

- niter : het aantal iteraties van het algoritme 4.6.
- ngcr : het aantal iteraties van het algoritme 4.4.
- ntrunc: het aantal zoekrichtingen in het Gram-Schmidt orthogonalisatie proces.

We hebben dus gezien hoe de SIMPLE methode voor de Navier-Stokes vergelijkingen versneld kan worden met behulp de GCR methode. Het is ook mogelijk om andere iteratieve methoden te gebruiken in plaats van de GCR methode (bv: GCR(k): GCR herstarten na k iteraties zie [7], MR: Minimal residual, GMRES: Generalized minimal residual zie [8] en [10], GMRESR zie [9] of een Krylov methode voor niet-lineaire stelsels zie [11], enz...).

Lijst van behandelde onderwerpen:

- Methoden voor het bepalen van numerieke oplossing
  - SIMPLE methode

- WISH3D solvers
- Krylov methoden
    * Gepreconditioneerde CG methode
    * GCR methode
- SIMLPE methode voor Stokes versnellen met GCR
- Diagonaalschaling
- SIMPLE methode voor Navier-Stokes versnellen met GCR

5 Toepassing van het GCR-SIMPLE algoritme

In dit hoofdstuk zullen we een korte inleiding geven over het toepassen van de SIMPLE methode en het GCR-SIMPLE algoritme op Stokes- (hoofdstuk 6), Navier-Stokes problemen (hoofdstuk 7) en Navier-Stokes gecombineerd een turbulentie model (hoofstuk 8). Aan de hand van een aantal testproblemen wordt het GCR-SIMPLE algoritme getest, de convergentie en de executietijd daarvan worden met die van de SIMPLE methode vergeleken. De variabelen die opgelost moeten worden zijn:

- De snelheid in de x-richting u of u_1,
- De snelheid in de y-richting v of u_2,
- De druk p.

- SIMPLE methode: Deze methode wordt beschreven door algoritme 4.1. In deze methode worden de matrices $\mathbf{Q}_1$, $\mathbf{Q}_2$, $\mathbf{R}$, $\mathbf{G}_1$, $\mathbf{G}_2$, $\mathbf{G}_1^*$ en $\mathbf{G}_2^*$ en de oplossing u, v en p bij iedere iteratie (*niter*) bepaald op de volgende wijze:

  for $i = 0, 1, 2, \ldots, niter$ **do**
  bepaal $\mathbf{Q}_1$ en $\mathbf{G}_1$,
  los op u
  bepaal $\mathbf{Q}_2$ en $\mathbf{G}_2$,
  los op v
  bepaal $\mathbf{R}$, $\mathbf{G}_1^*$ en $\mathbf{G}_2^*$,
  los op p
  end for

- GCR-SIMPLE : Deze methode maakt gebruik van het algoritme 4.6. Omdat het bepalen van de matrices $\mathbf{G}_1$, $\mathbf{G}_2$, $\mathbf{G}_1^*$ en $\mathbf{G}_2^*$ goedkoper is ten opzichte van $\mathbf{Q}_1$ en $\mathbf{Q}_2$, worden deze matrices bepaald op het moment dat ze nodig zijn. Op deze manier is er geen extra geheugen nodig. De matrices $\mathbf{Q}_1$, $\mathbf{Q}_2$ en $\mathbf{R}$ en de oplossing u, v en p worden op de volgende wijze bepaald:

  for $i = 0, 1, 2, \ldots, niter$ **do**
  bepaal de matrices $\mathbf{Q}_1$, $\mathbf{Q}_2$ en $\mathbf{R}$,
  for $k = 1, 2, \ldots, ngcr$ **do**
  los op u, v en p
  end for
  end for

Voor iedere variabele u, v en p wordt de bijbehorende vergelijking plus de randvoorwaarden opgelost. De stelsels voor u, v en p zowel in de SIMPLE methode als in het GCR-SIMPLE algoritme kunnen gerelaxeerd worden met een factor $relax$. Tenzij anders wordt vermeld wordt in het GCR-SIMPLE algoritme $relax(u) = relax(v) = 1$ en $relax(p) = 1$. In dit verslag wordt gebruik gemaakt van de notaties:

- $tijd$: De executietijd voor het oplossen van een bepaald probleem.
- $residu$: Voor een gegeven stelsel $\mathbf{Q}x = b$ wordt het $residu$ van $\tilde{x}$ gegeven door de formule: $residu = \frac{||b-\mathbf{Q}\tilde{x}||_1}{||\mathbf{D}\tilde{x}||_1}$, waarin $\mathbf{D} = Diag(Q)$.

Voor het bepalen van de executietijd wordt gekozen, tenzij anders wordt vermeld, voor het stopcriterium:
$residu(u) \leq 10^{-10}$, $residu(v) \leq 10^{-10}$ en $residu(p) \leq 10^{-10}$. Bovendien in het geval van turbulente stromingen ($k - \epsilon$ model), het stopcriterium voor de turbulente kinetische energie k en de dissipatie daarvan ϵ luidt, tenzij anders wordt vermeld: $residu(k) \leq 10^{-10}$ en $residu(epsilon) \leq 10^{-10}$.

De volgende notaties zoals gedefinieerd in hoofdstuk 4 zijn in het hele verslag van toepassing:

- $niter$: het aantal iteraties van het algoritme 4.6,
- $ngcr$: het aantal iteraties van het algoritme 4.4,
- $ntrunc$: het aantal zoekrichtingen in het Gram-Schmidt orthogonalisatie process.

Merk op dat $ntrunc = 1$ betekent dat er geen orthogonalisatie plaats vindt in het Gram-Schmidt orthogonalisatie process.

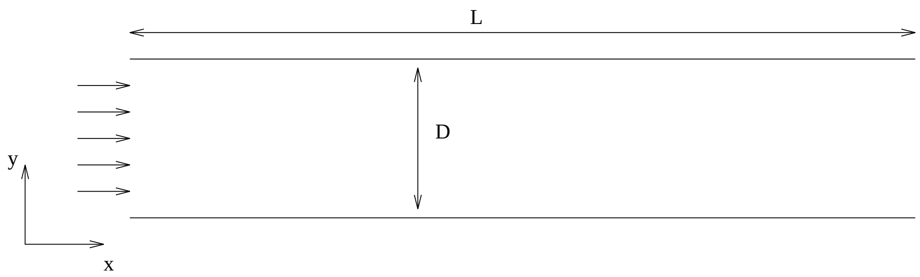
Het GCR-SIMPLE algoritme en de SIMPLE methode worden toegepast voor verschillende problemen namelijk, Stokes, Navier-Stokes en turbulentie problemen. Er wordt in alle testproblemen een twee dimensionale luchtstroming in de horizontale richting met instroomsnelheid u_0 beschouwd. De stroming is stationair en heeft de volgende eigenschappen:

- De dichtheid van de lucht $\rho = 1.16$ (kg/m^3),
- De dynamische viscositeit van de lucht $\mu = 1.824\ 10^{-5}$ (kg/ms).

Het Reynoldsgetal

De SIMPLE methode en het GCR-SIMPLE algoritme worden in de komende hoofdstukken toegepast voor een twee dimensionale stroming tussen twee vlakke platen en stroming over een backward facing step. Voor deze problemen wordt

de volgende definitie van het Reynolds getal gebruikt:
$Re = \frac{\rho u_0 d}{\mu}$, waarin:
d de karakteristieke lengte voorstelt.
Voor een stroming tussen twee vlakke platen wordt vaak d gelijk aan de hydraulische diameter genomen, met $d = \frac{4E}{C}$, E is het doorstroomde oppervlak en C is de natte omtrek. Voor een kanaal stroming wordt het Reynoldsgetal gedefinieerd door: $d = 2D$ zie figuur 5.1.

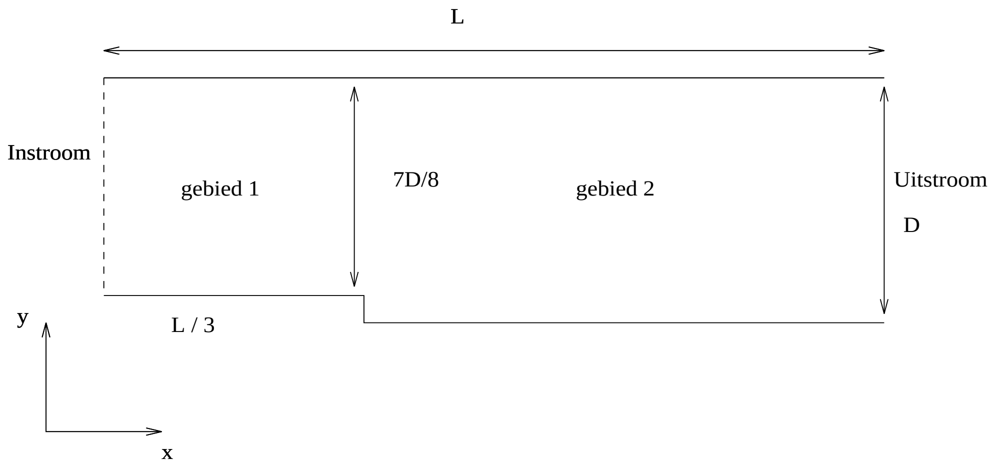


Figuur 5.1: Stroming tussen twee vlakke platen.

Dus

$$Re = \frac{2D\rho u_0}{\mu}. \tag{5.1}$$

Voor de stroming over een backward facing step zullen we het Reynoldsgetal in gebied 2 nemen zie figuur 5.2



Figuur 5.2: Stroming over een backward facing step.

In hoofdstuk 6 worden Stokes problemen behandeld. De stroming tussen twee vlakken platen wordt beschouwd. In hoofdstuk 7 en hoofdstuk 8 komen respectievelijk Navier-Stokes en turbulentie problemen aan de orde, hier wordt de stroming tussen twee vlakke platen en de stroming in een backward facing step beschouwd.

Lijst van behandelde onderwerpen:

- Toepassen van het GCR-SIMPLE algoritme
  - SIMPLE methode
  - GCR-SIMPLE methode
  - Het Reynoldsgetal

6 Toepassing van GCR-SIMPLE op Stokes problemen

Aan de hand van een testprobleem zullen we het GCR-SIMPLE algoritme testen. De stroming tussen twee vlakke platen wordt beschouwd. Dit probleem wordt opgelost met behulp van de SIMPLE methode het GCR-SIMPLE algoritme en de resultaten worden met elkaar vegeleken.

1. **SIMPLE**: Gebruikmakend van de SIMPLE methode worden de variabelen u, v en p op de volgen de manier opgelost:

   for $k = 0, 1, ..., niter$ **do**
   los op u
   los op v
   los op p
   end for

   Om de stelsels in het bovenstaande algoritme op te lossen gebruiken we de volgende solvers:

variabele	solver
u, v, p	planez

   Tabel 6.1: De randvoorwaarden.

   Voor de variabelen u, v, p wordt gebruik gemaakt van de volgende relaxatiefactoren (notatie: $relax$):

variabele	u	v	p
relax	0.6	0.6	1.0

   Tabel 6.2: De relaxatiefactoren voor de SIMPLE methode.

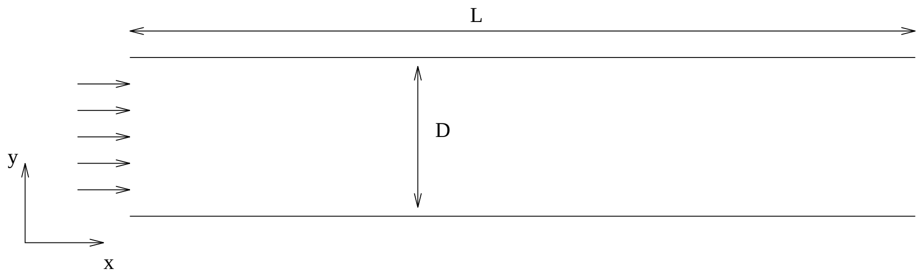
2. **GCR-SIMPLE**: Hier wordt gebruik gemaakt van het GCR-SIMPLE algoritme voor het oplossen van de variabelen u, v en p. Dit kunnen we samenvatten in het volgende algoritme:

kies een beginschatting
for $i = 0, 1, 2, \ldots, niter$ **do**
  los op u, v en v met behulp van het GCR-SIMPLE algoritme (gebruikmakend van planez).
end for

Voor de bepaling van de executietijd wordt gekozen tenzij anders wordt vermeld voor het stopcriterium: $residu(u) \leq 10^{-10}, residu(v) \leq 10^{-10}$ en $residu(p) \leq 10^{-10}$.

6.1 Stroming tussen twee vlakke platen

Beschouw een twee dimensionale stationaire luchtsroming in de x-richting tussen twee vlakke platen zoals geschetst in Figuur 6.1.



Figuur 6.1: Stroming tussen twee vlakke platen

De gebruikte afmetingen zijn:

$$D = 10\ cm,$$
$$L = 500\ cm.$$

De stroming is laminair met een instroomsnelheid u_0 en het Reynoldsgetal is gegeven door:

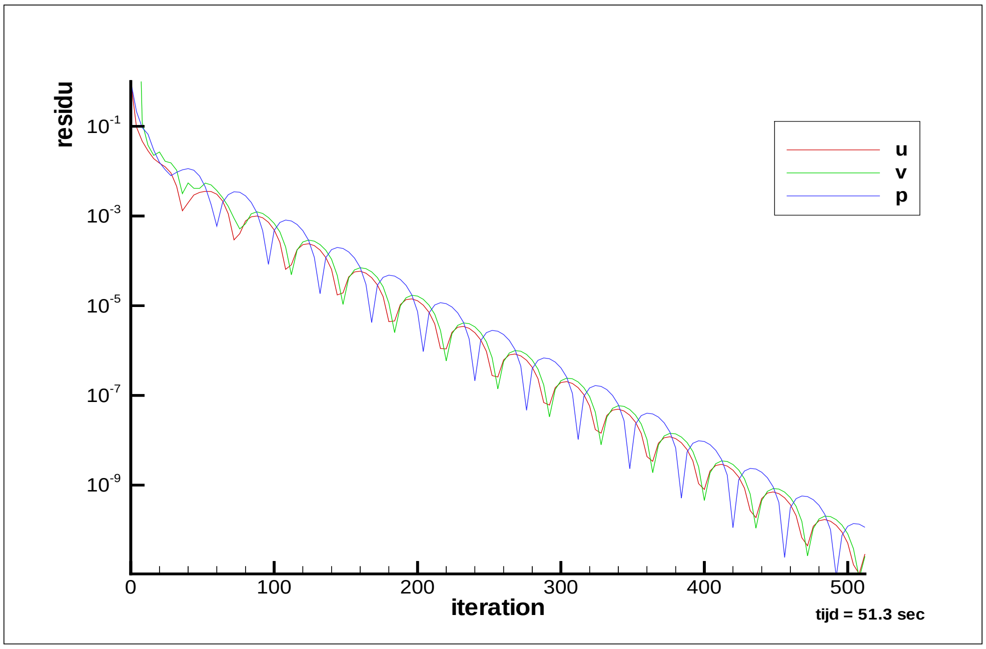
$$Re = \frac{2\rho D u_0}{\mu}. \tag{6.1}$$

Randvoorwaarden:
De gebruikte randvoorwaarden zijn gegeven in tabel 6.3:

variabele	instroomrand	uitstroomrand	vaste wanden
snelheid u	$u = u_0$	$\frac{\partial u}{\partial x} = 0$	$u = 0$
snelheid v	$v = 0$	$v = 0$	$v = 0$
druk p	$\frac{\partial p}{\partial x} = 0$	$p = 0$	$\frac{\partial p}{\partial y} = 0$

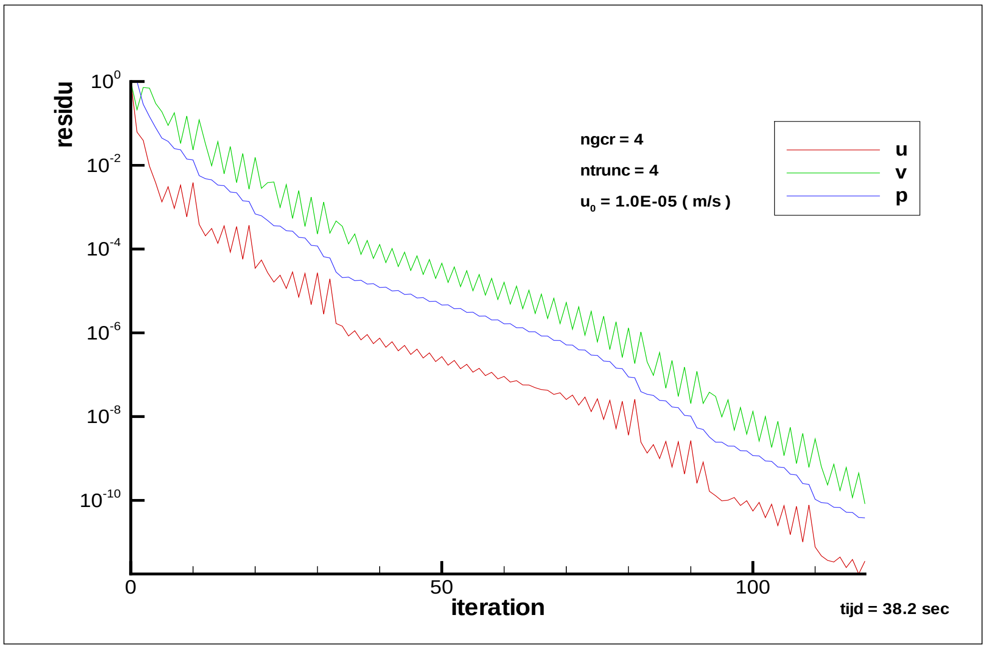
Tabel 6.3: Randvoorwaarden.

Beschouw een stroming met instroomsnelheid $u_0 = 10^{-5}\ m/s$. Het Reynolds-getal is $Re \approx 0.127$. Dat wil zeggen dat we te maken hebben met een Stokes stroming zoals beschreven in sectie 2.2.3. We lossen het gegeven probleem op voor een rekenrooster (20×10). We zullen ten eerste het gegeven probleem oplossen met behulp van de SIMPLE methode. Het convergentiegedrag wordt gegeven in figuur 6.2.

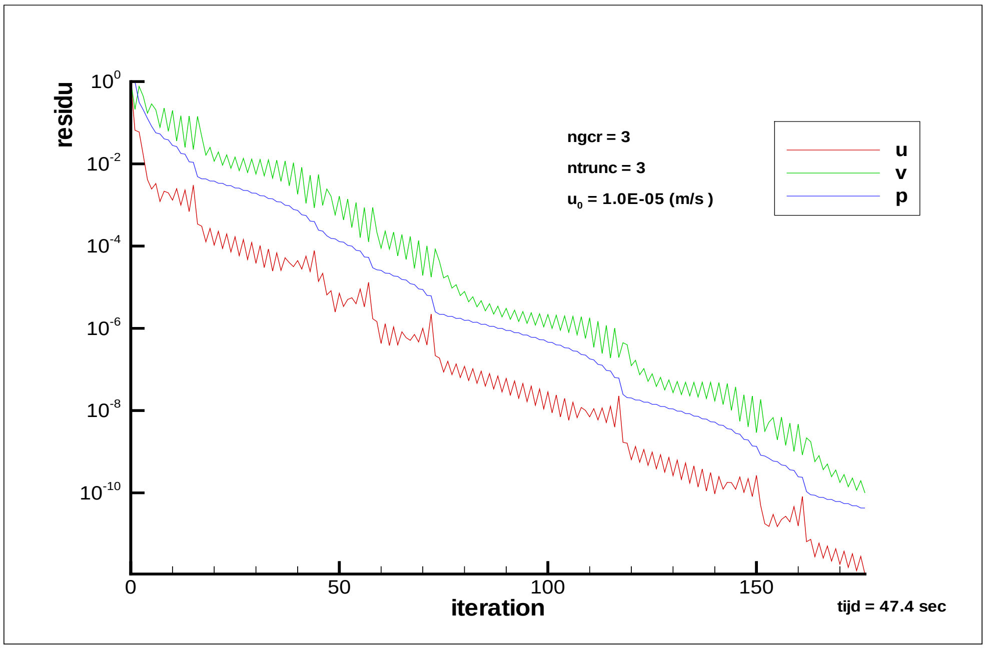


Figuur 6.2: SIMPLE methode: rooster (20×10).

Hetzelfde probleem gaan we nu oplossen met behulp van het GCR-SIMPLE algoritme. De resultaten zijn in figuur 6.3 en 6.4 en in tabel 6.5 gegeven.



Figuur 6.3: GCR-SIMPLE methode: rooster (20×10)



Figuur 6.4: GCR-SIMPLE methode: rooster (20×10)

SIMPLE	niter	tijd (sec)
	515	51.3

Tabel 6.4: Resultaten met behulp van de SIMPLE methode.

ngcr	ntrunc	niter	tijd (sec)
3	3	176	47.4
3	2	239	76.8
4	4	118	38.2
4	3	150	47.6
5	5	222	80.9

Tabel 6.5: Resultaten met behulp van het GCR-SIMPLE algoritme.

Opmerkingen:

- Uit de verkregen resultaten kunnen we concluderen dat het GCR-SIMPLE algoritme voor een gegeven $ngcr$ de beste resultaten geeft als $ngcr = ntrunc$.
- Het GCR-SIMPLE algoritme voor $ngcr = ntrunc = 4$ convergeert sneller dan de SIMPLE methode.
- Merk op dat het aantal buiteniteraties ($niter$) groter wordt als $ngcr$ groter wordt dan 4.

Lijst van behandelde onderwerpen:

- Toepassing van het GCR-SIMPLE algoritme op Stokes problemen
  - SIMPLE methode
  - GCR-SIMPLE methode
  - Stroming tussen twee vlakke platen

7 Toepassing van GCR-SIMPLE op Navier-Stokes problemen

In deze sectie zullen we aandacht besteden aan Navier-Stokes problemen. Deze problemen worden opgelost met behulp van de SIMPLE methode en het GCR-SIMPLE algoritme en de resultaten worden met elkaar vegeleken.

1. SIMPLE: Gebruikmakend van de SIMPLE methode worden de variabelen u, v en p op de volgende manier opgelost:
   for $k = 0, 1, ..., niter$ **do**
   los op u
   los op v
   los op p
   end for

   Om de stelsels in het bovenstaande algoritme op te lossen gebruiken we de volgende solvers:

variabele	solver
u, v, p	planez

   Tabel 7.1: TDMA solvers voor de SIMPLE methode.

   Voor de variabelen u, v en p wordt gebruik gemaakt tenzij anders wordt vermeld van de volgende default waarden van de relaxatiefactoren (notatie: $relax$). Deze waarden zijn mogelijk niet optimaal:

variabele	u	v	p
relax	0.6	0.6	1.0

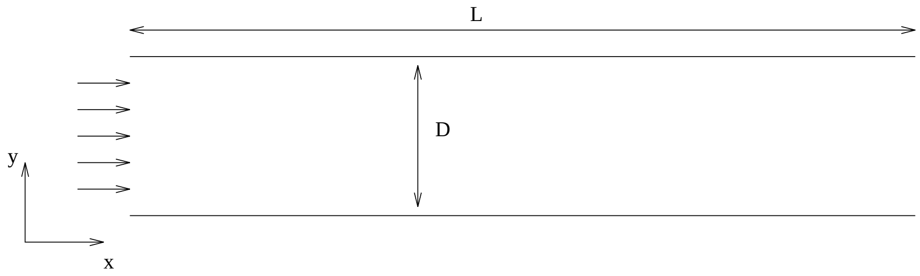
   Tabel 7.2: Relaxatiefactoren voor de SIMPLE methode.

2. GCR-SIMPLE: Hier wordt gebruik gemaakt van het GCR-SIMPLE algoritme voor het oplossen van de variabelen u, v en p. Dit kunnen we samenvatten in het volgende algoritme:
   kies een beginschatting
   for $i = 0, 1, 2, \ldots, niter$ **do**
   los op u, v en v met behulp van het GCR-SIMPLE algoritme (gebruikmakend van planez).
   end for

Voor de bepaling van de executietijd wordt tenzij anders wordt vermeld gekozen voor het stopcriterium: $residu(u) \leq 10^{-10}, residu(v) \leq 10^{-10}$ en $residu(p) \leq 10^{-10}$.

7.1 Stroming tussen twee vlakke platen

Beschouw een twee dimensionale stationaire luchtstroming in de x-richting tussen twee vlakke platen zoals geschetst in Figuur 7.1.



Figuur 7.1: Stroming tussen twee vlakke platen

De gebruikte afmetingen zijn:

$$D = 10 \ cm,$$
$$L = 500 \ cm.$$

De stroming is laminair met een instroomsnelheid $u_0 = 0.1 \ (m/s)$ en het Reynoldsgetal is gegeven door:

$$Re = \frac{2\rho D u_0}{\mu} = 1272. \tag{7.1}$$

7.1.1 GCR-SIMPLE algoritme zonder schaling

Om de overgang van het GCR-SIMPLE algoritme zonder schaling naar het GCR-SIMPLE algoritme met schaling te motiveren, zullen we het eerstgenoemde algoritme toepassen op de stroming tussen twee vlakke platen. De resultaten in tabel 7.3 zijn verkregen:

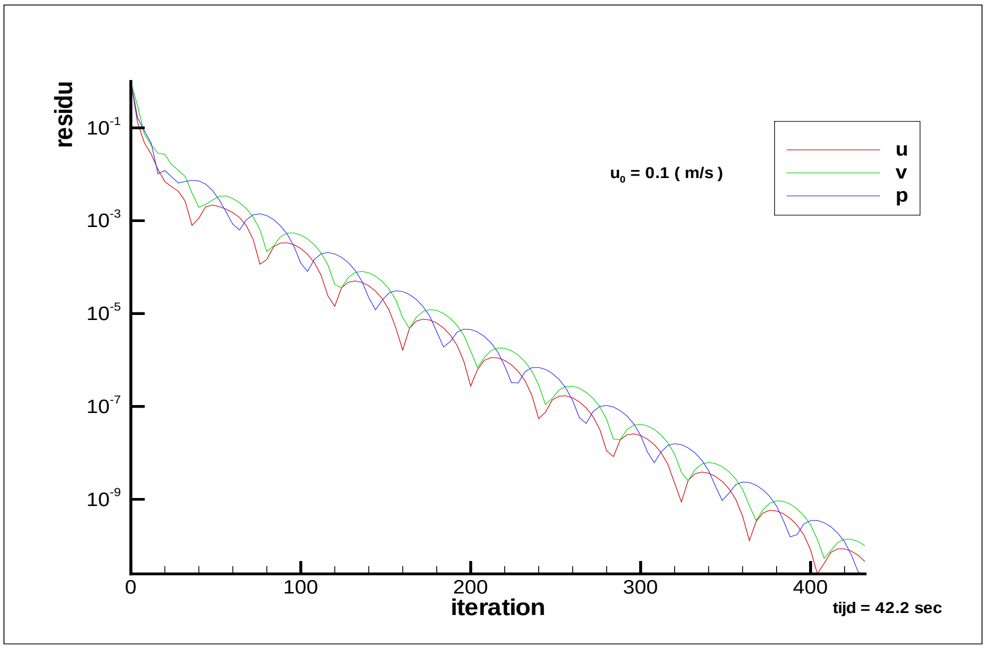
$ngcr$	1	1	2	2	3	3	3	3	4	4
$ntrunc$	1	1	2	2	3	3	3	3	4	4
$relax(u,v)$	1	0.8	1	0.6	1	0.6	0.8	0.4	1	0.6
$niter$	ddn	211	147	301	208	206	ddn	ddn	264	201
$tijd$ (sec)		38	33.4	65.7	56	55.3			82.5	63.2

Tabel 7.3: GCR-SIMPLE algoritme zonder schaling, rooster (20×10), ddn=deling door nul.

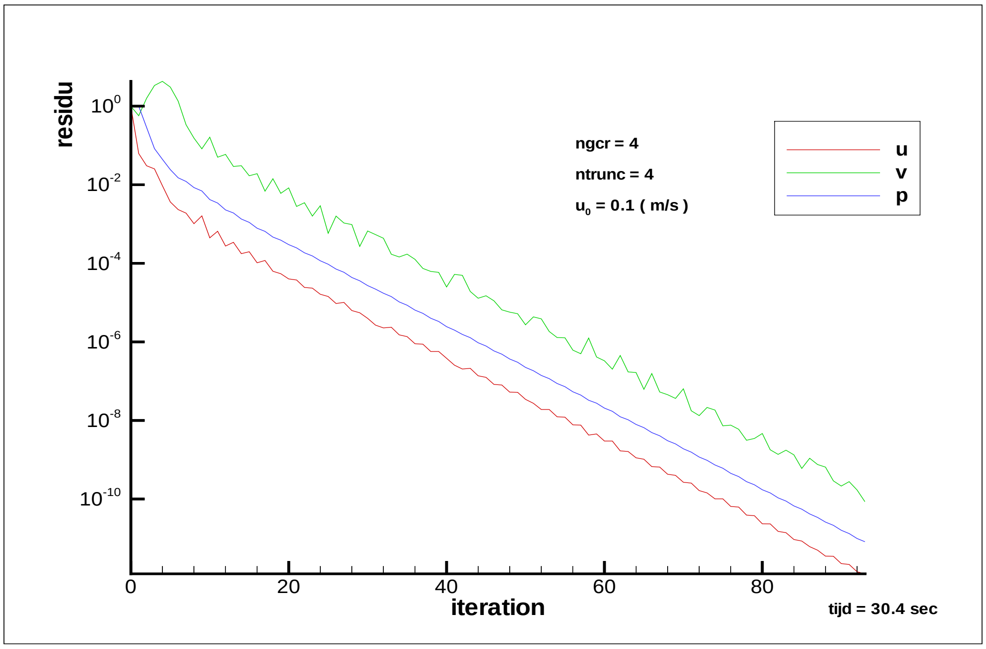
Uit de resultaten van tabel 7.3 concluderen we dat voor het GCR-SIMPLE algoritme zonder schaling voor bepaalde waarden van $ngcr$ een 'deling door nul' in WISH3D subroutines optreedt. Voor een fijn rooster bijvoorbeeld (40×20) dan treedt voor veel waarden van $ngcr$ een 'deling door nul' op. We zullen daarom voortaan het GCR-SIMPLE algoritme met schaling toepassen.

7.1.2 GCR-SIMPLE algoritme met schaling

In deze paragraaf passen we het GCR-SIMPLE algoritme met schaling toe op Navier-Stokes problemen. Gekeken zal word worden naar de invloed van de relaxatiefactoren en de invloed van de grootte van het rooster bij het toepassen-van de GCR-SIMPLE algoritme en de SIMPLE methode. Tenslotte wordt gekeken of het GCR-SIMPLE algoritme voor langgerekte cellen nog steeds convergeert.
We passen de SIMPLE methode en het GCR-SIMPLE algoritme op een rooster (20×10) toe. Het convergentiegedrag is gegeven in figuur 7.2 en 7.3



Figuur 7.2: SIMPLE methode: rooster (20×10).



Figuur 7.3: GCR-SIMPLE algoritme: rooster (20×10).

In tabel 7.4 zijn het aantal iteraties ($niter$) en de executietijd van de SIMPLE methode gegeven. In tabel 7.5 en tabel 7.6 zijn de resultaten van het GCR-SIMPLE algoritme gegeven voor verschillende waarden van $ngcr$ en $ntrunc$.

SIMPLE methode	niter = 415	tijd = 42.2 (sec)

Tabel 7.4: Resultaten van de SIMPLE methode, rooster (20×10), $relax(u, v) = 0.6$.

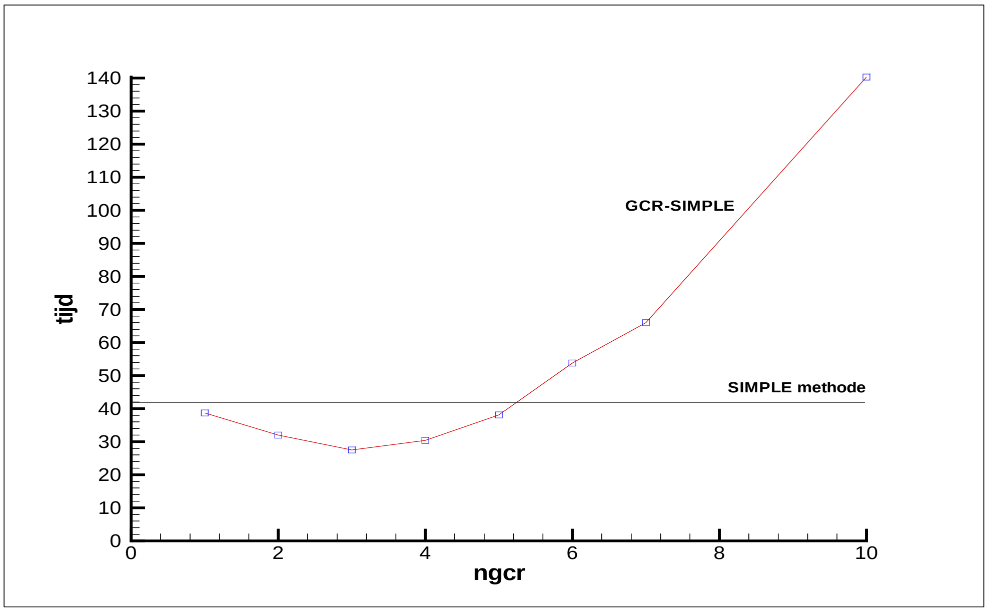
ngcr	2	3	3	4	4	4	5	5	5
ntrunc	2	2	3	2	3	4	2	3	4
niter	138	99	96	139	97	93	207	121	102
tijd	32	27.9	27.5	44.3	32.1	30.4	73.4	44.1	38.1

Tabel 7.5: Resultaten van het GCR-SIMPLE algoritme, rooster (20×10), $relax(u, v) = 1$.

ngcr	5	6	6	7	10
ntrunc	5	5	6	7	10
niter	100	130	125	137	215
tijd	36.9	55.5	53.8	66	140.3

Tabel 7.6: Resultaten van het GCR-SIMPLE algoritme, rooster (20×10), $relax(u, v) = 1$.

Het GCR-SIMPLE algoritme toegepast op Navier-Stokes problemen geeft voor een gegeven $ngcr$ de beste resultaten voor $ngcr = ntrunc$. We zullen in de volgende secties en hoofdstukken het GCR-SIMPLE algoritme met $ngcr = ntrunc$ toepassen.



Figuur 7.4: Rooster (20×10). De resultaten van het GCR-SIMPLE algoritme $(relax(u,v) = 1)$ en het resultaat van de SIMPLE methode met $(relax(u,v) = 0.6$ zie tabel 7.4 en tabel 7.5.

Opgemerkt kan worden uit figuur 7.4 dat zonder gebruik te maken van de relaxatiefactoren voor het GCR-SIMPLE algoritme voor een groot aantal parameters $ngcr$ goede resultaten geeft.

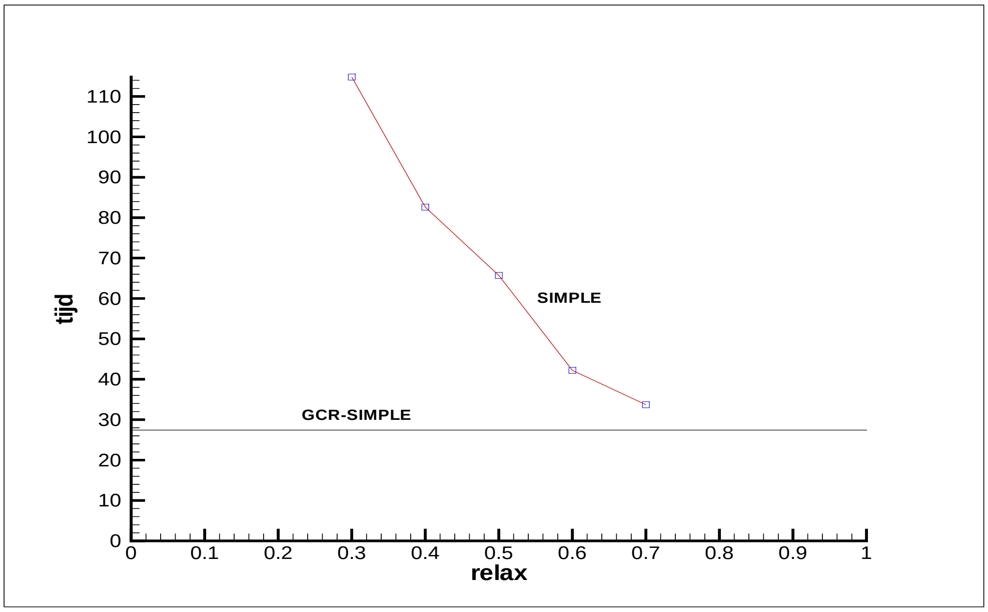
De SIMPLE methode en verschillende relaxatiefactoren

We gaan nu bekijken hoe de SIMPLE methode zich gedraagt ten opzichte van verschillende relaxatiefactoren: In tabel 7.7 zijn de resultaten gegeven:

relax u, v	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1
tijd	235	155	114.8	82.6	65.7	42.2	33.7	ddn	ddn	ddn

Tabel 7.7: SIMPLE methode voor verschillende relaxatiefactoren: ddn = deling door nul

Merk op dat de SIMPLE methode gevoelig is voor de juiste keuze van de relaxatiefactor. Dichtbij de optimale waarde 0.7 treedt een 'deling door nul' voor $relax = 0.8$ op.



Figuur 7.5: Rooster (20×10). De resultaten van de SIMPLE methode en het resultaat van de GCR-SIMPLE methode (met $relax(u, v) = 1$) zie tabel 7.4 en tabel 7.7.

In figuur 7.5 is duidelijk te zien dat voor alle relaxatiefactoren de SIMPLE methode meer tijd in beslag neemt dan de GCR-SIMPLE methode voor een breed interval van $ngcr$ (zie figuur 7.4). Merk op dat het GCR-SIMPLE algoritme gebruikt is zonder relaxatiefactoren.

GCR-SIMPLE algoritme en relaxatie

We volstaan met het volgende $relax(u) = relax(v)$. We passen dit toe voor het beste resultaat van het GCR-SIMPLE algoritme voor dit rooster, dat wil zeggen $ngcr = 3$ en $ntrunc = 3$, merk op dat de resultaten van tabel 7.5 zijn verkregen met $relax(u) = relax(v) = 1$), De resultaten zijn gegeven in tabel 7.8:

relax u, v	1	0.6	0.8	0.9	0.94	0.95	0.96	0.97	0.98	0.99
tijd (sec)	27.5	65.9	42.3	30.8	26	25.3	24.2	23.3	26.2	27.7

Tabel 7.8: GCR-SIMPLE en verschilende relaxatiefactoren.

Merk op dat de optimale waarde van de relaxatiefactor voor het GCR-SIMPLE algoritme zich tussen 1 en 0.8 bevindt.

Toepassen van het GCR-SIMPLE algoritme voor verschillende roosters

Definieer de versnellingsfactor $f = \frac{\text{tijd(SIMPLE)}}{\text{tijd(GCR-SIMPLE)}}$ Uit de testen is gebleken dat de versnelling afhankelijk is van de grootte van de rekenroosters. Beschouw de rekenroosters (20×10), (40×10) en (40×20). De SIMPLE methode en de GCR-SIMPLE methode worden op de verschillende rekenroosters toegepast. De resultaten zijn in tabel 7.9 samengevat.

Rooster	tijd (SIMPLE) $relax\ u,\ v = 0.6$	tijd(GCR-SIMPLE) $ngcr = 3,\ ntrunc = 3,\ relax(u,v) = 1$	f
(20×10)	42.2	27.5	1.53
(40×10)	303.6	152.3	2
(40×20)	470.8	218.7	2.15

Tabel 7.9: Afhankelijkheid van SIMPLE en GCR-SIMPLE voor verschillende groottes van het rooster.

Merk op voor de roosters in tabel 7.9 dat de winst groter wordt naarmate de roostergrootte toeneemt.

Langgerekte cellen

Om het GCR-SIMPLE op extreme gevallen toe te passen zullen we aandacht besteden aan de toepassing van dit algoritme voor rekenroosters met langgerekte cellen. Hiertoe delen we het kanaal op in 10 cellen in de x-richting en 20 cellen in de y-richting. Voor de SIMPLE methode is $relax(u) = relax(v) = 0.6$ gebruikt.

SIMPLE	$niter = 269$	$tijd = 29.6$

Tabel 7.10: De resultaten van de SIMPLE methode op het rekenrooster (10×20)

GCR-SIMPLE	$ngcr$	$ntrunc$	$niter$	$tijd\ (sec)$
	1	1	121	28.4
	2	2	61	18.5
	3	3	53	18.6
	4	4	73	28.4

Tabel 7.11: De resultaten van het GCR-SIMPLE op een rekenrooster (10×20).

Stel dat $h_1,\ h_2$ de stapgrootte van het rekenrooster (10×20) respectievelijk in de x-richting en in de y-richting dan geldt $h_1 = 50$ en $h_2 = 0.5$, anders gezegd $h_1 = 100h_2$. Uit tabel 7.11 concluderen we dat ook voor rekenroosters met langgerekte cellen het beter is om kleinere parameters $ngcr$ en $ntrunc$ te nemen voor het GCR-SIMPLE algoritme.

Opmerking: Rooster met kleine stapgroottes:
We passen de SIMPLE methode Voor een rooster (120×20) toe. De afmetingen zijn:

$$D = 10cm$$
$$L = 500cm$$

De resultaten zijn samengevat in tabel 7.12.

$relax(u,v)$	$niter$	$tijd$
0.7	div	
0.6	1022	854

Tabel 7.12: SIMPLE methode, rooster (120×20), $u_0 = 10^{-4}$ (m/s), div = divergeert.

Met behulp van het GCR-SIMPLE algoritme volgen de resultaten van tabel 7.13.

$ngcr$	$ntrunc$	$niter$	$tijd$
4	4	817	2426
8	8	274	1530

Tabel 7.13: GCR-SIMPLE algoritme met $ngcr = ntrunc = 4$, rooster (120×20), $u_0 = 10^{-4}$ (m/s).

Opmerking:
We zien een vertraging in de GCR-SIMPLE methode ten opzichte van de SIMPLE methode.

Stel dat we de volgende afmetingen nemen:

$$D = 50cm$$
$$L = 500cm$$

Topassing van de SIMPLE methode levert de resultaten van tabel 7.14.

$relax(u, v)$	$niter$	$tijd$
0.6	1191	980
0.7	836	692

Tabel 7.14: SIMPLE methode, rooster (120×20), $u_0 = 10^{-4}$ (m/s).

In tabel 7.15 zijn de resultaten met behulp van het GCR-SIMPLE algoritme gegeven.

$niter$	$tijd$
89	271

Tabel 7.15: GCR-SIMPLE algoritme met $ngcr = ntrunc = 4$, rooster (120×20), $u_0 = 10^{-4}$ (m/s).

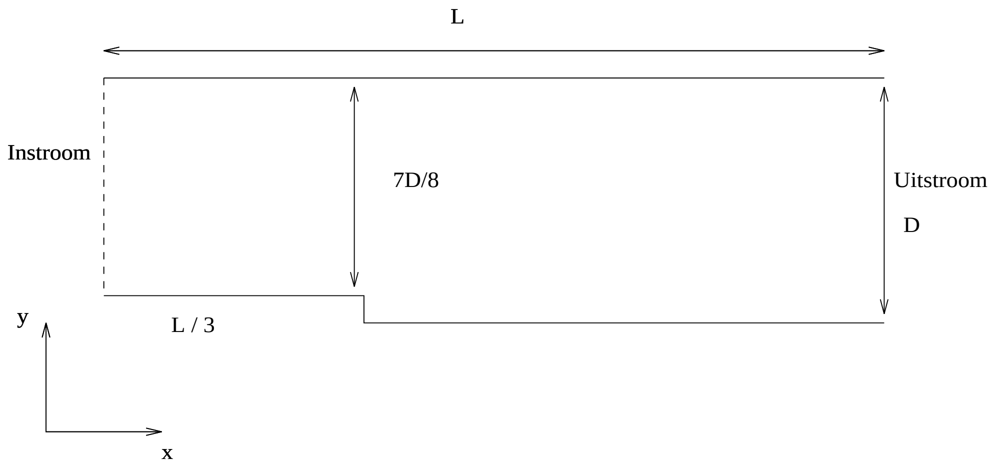
Opmerking:
Merk op nu dat het GCR-SIMPLE algoritme sneller convergeert dan de SIMPLE methode.

Dit aspect zou nog verder onderzocht moeten worden. De volgende punten kunnen nader onderzocht worden:

- Grotere waarden van $ngcr$ nemen.
- In plaats van planez andere varianten van TDMA solvers gebruiken.

7.2 Stroming in een Backward facing step

Beschouw een stationaire luchtstroming in het gebied zoals geschetst in figuur 7.6. De afmetingen zijn:



Figuur 7.6: Stroming in een backward facing step.

$$D = 20\ cm,$$
$$L = 500\ cm.$$

Randvoorwaarden.:
De gebruikte randvoorwaarden zijn in tabel 7.16 gegeven:

variabele	instroomrand	uitstroomrand	vaste wanden
snelheid u	$u = u_0$	$\frac{\partial u}{\partial x} = 0$	$u = 0$
snelheid v	$v = 0$	$v = 0$	$v = 0$
druk p	$\frac{\partial p}{\partial x} = 0$	$p = 0$	$\frac{\partial p}{\partial n} = 0$

Tabel 7.16: Randvoorwaarden

De solvers die zowel in SIMPLE als in GCR-SIMPLE worden gebruikt zijn:

variabele	solver
u, v, p	planex, planey, planez

Tabel 7.17: TDMA solvers voor de SIMPLE methode en het GCR-SIMPLE algoritme.

In tabel 7.17 hebben we voor een andere TDMA solver variant gekozen. Hier wordt voor iedere variabele het stelsel opgelost met planex, planey en planez. Op die manier kunnen we het GCR-SIMPLE algoritme testen voor verschillen TDMA solvers.

De volgende relaxatiefactoren worden in de SIMPLE methode gebruikt tenzij anders wordt vermeld:

variabele	u	v	p
relax	0.6	0.6	1.0

Tabel 7.18: Relaxatiefactoren voor de SIMPLE methode.

Voor de bepaling van de executietijd wordt tenzij anders wordt vermeld gekozen voor het stopcriterium: $residu(u) \leq 10^{-10}, residu(v) \leq 10^{-10}$ en $residu(p) \leq 10^{-10}$.

De resultaten voor een rooster (40×20):

$relax(u,v)$	$niter$	tijd (sec)
0.6	1122	545
0.7	ddn	

Tabel 7.19: Resultaten van het SIMPLE algoritme , rooster $(40 \times 20), u_0 = 0.01(m/s)$, $Re = 254$, ddd = deling door nul.

ngcr	1	2	3	4	5
ntrunc	1	2	3	4	5
niter	396	249	173	129	156
tijd (sec)	273	258.1	249.8	236.5	349.8

Tabel 7.20: Resultaten van het GCR-SIMPLE algoritme, rooster $(40 \times 20), u_0 = 0.01(m/s)$. $Re = 254$.

Merk op dat de optimale relaxatiefactor 0.7 voor de stroming tussen twee vlakke platen met een rooster (20×10) zie tabel 7.7 leidt nu tot een 'deling door nul' in de WISH3D subroutines zie tabel 7.19. Terwijl de waarden van *ngcr* nog steeds tot goede resultaten leiden zie tabel 7.5 en tabel 7.20.

Lijst van behandelde onderwerpen:

- Toepassing van het GCR-SIMPLE algoritme op Navier-Stokes problemen
  - Stroming tussen twee vlakke platen
    - GCR-SIMPLE algoritme zonder schaling
    - GCR-SIMPLE algoritme met schaling
    - De SIMPLE methode en verschillende relaxatiefactoren
    - GCR-SIMPLE en relaxatie
    - Toepassen van GCR-SIMPLE voor verschillende roosters
    - Langgerekte cellen
  - Stroming in een backward facing step

8 Toepassen van GCR-SIMPLE op turbulente stromingen

We zullen in dit hoofdstuk aandacht besteden aan turbulente stromingen. Hiervoor wordt gebruik gemaakt van het $k - \epsilon$ turbulentie model zie sectie 2.3.5.

Door de testen die uitgevoerd zijn bij het oplossen van de Navier-Stokes problemen in hoofstuk 7 hebben we opgemerkt dat het GCR-SIMPLE algoritme betere convergentie oplevert wanneer $ngcr = ntrunc$. Gemotiveerd door deze resultaten zullen we voor turbulente problemen gebruik maken van het GCR-SIMPLE algoritme met $ngcr = ntrunc$. Analoog aan de Navier-Stokes problemen zullen we voor de turbulente problemen ook twee testproblemen gebruiken namelijk de stroming tussen twee vlakke platen en de stroming over een backward facing step.

De variabelen die opgelost moeten worden zijn:

- Snelheid in x-richting u
- Snelheid in y-richting v
- Druk p
- Turbulente kinetische energie k
- Dissipatie van de turbulente kinetische energie ϵ

Voor de bepaling van de executietijd wordt gekozen voor het stopcriterium: $residu(u) \leq 10^{-6}, residu(v) \leq 10^{-6}, residu(p) \leq 10^{-6}, residu(k) \leq 10^{-6}$ en $residu(\epsilon) \leq 10^{-6}$.

Er wordt gebruik gemaakt van twee verschillende oplosmethoden namelijk:

1. **SIMPLE**: Gebruikmakend van de SIMPLE methode worden de variabelen u, v, p, k en ϵ op de volgende manier opgelost:
   for $k = 0, 1, ..., niter$ **do**
   los op u
   los op v
   los op p
   los op k
   los op ϵ
   end for

Om de stelsels in het bovenstaande algoritme op te lossen gebruiken we de volgende solvers:

variabele	solver
u, v, p, k, ϵ	planez

Tabel 8.1: TDMA solvers voor de SIMPLE methode.

Voor de variabelen u, v, p, k en ϵ wordt gebruik gemaakt tenzij anders wordt vermeld van de volgende default relaxatiefactoren (notatie: $relax$) in de SIMPLE methode.:

variabele	u	v	p	k	ϵ
$relax$	0.6	0.6	1.0	0.6	0.6

Tabel 8.2: Relaxatiefactoren voor de SIMPLE methode.

2. **GCR-SIMPLE**: Hier wordt gebruik gemaakt van het GCR-SIMPLE algoritme voor het oplossen van de variabelen u, v en p. De variabelen k en ϵ worden bepaald met behulp van TDMA solvers. Dit kunnen we samenvatten in het volgende algoritme:

   kies een beginschatting
   for $i = 0, 1, 2, \ldots, niter$ **do**
   los u, v en p op met behulp van het GCR-SIMPLE algoritme (gebruikmakend van planez)
   los k op met behulp van planez
   los ϵ op met behulp van planez
   end for

De bovengenoemde methoden worden op de volgende testproblemen toegepast:

- Stroming tussen twee vlakke platen.
- Stroming in een backward facing step.

8.1 Stroming tussen twee vlakke platen

Beschouw een twee-dimensionaal stationaire luchtstroming tussen twee vlakke platen zoals geschetst in figuur 7.1. De afmetingen zijn:

$$D = 20\ cm,$$
$$L = 500\ cm.$$

De stroming heeft een instroomsnelheid $u_0 = 1.0(m/s)$ en overige eigenschappen van de stroming zijn al gegeven in hoofdstuk 5. Er worden de in tabel 8.3 aangegeven randvoorwaarden gebruikt:

variabele	instroomrand	uitstroomrand	vaste wanden
u	$u = u_0$	$\frac{\partial u}{\partial x} = 0$	$u = 0$
v	$v = 0$	$v = 0$	$v = 0$
p	$\frac{\partial p}{\partial x} = 0$	$p = 0$	$\frac{\partial p}{\partial y} = 0$
k	k = 0.005400	$\frac{\partial k}{\partial x} = 0$	$wandfuncties$
ϵ	$\epsilon = 0.00543365$	$\frac{\partial \epsilon}{\partial x} = 0$	$wandfuncties$

Tabel 8.3: Randvoorwaarden.

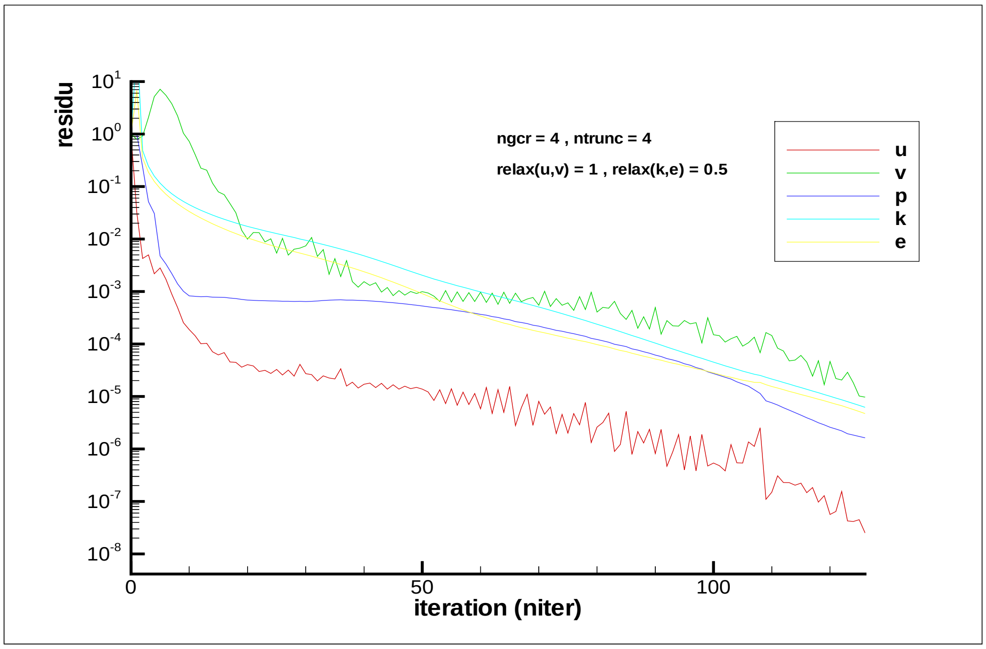
Wandfuncties:

Voor de turbulentie problemen wordt in WISH3D gebruik gemaakt van twee typen wandfuncties:

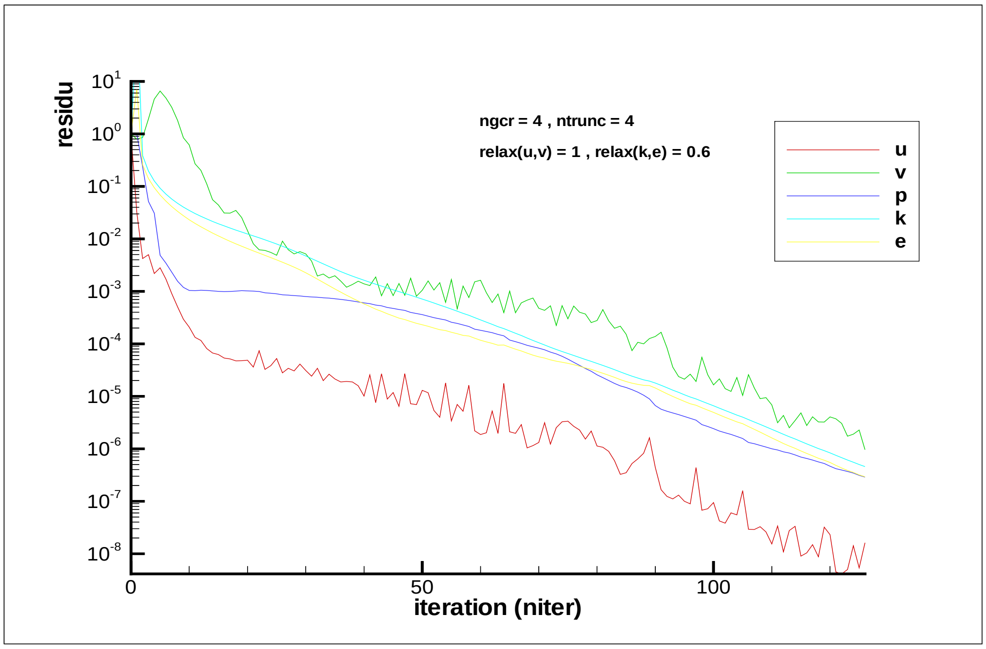
- Type 1: turbulent 'inertial' grenslaag, wordt toegepast als een alternatief voor Type 2. Er wordt aangenomen dat het naast de wand liggende roosterpunt in de 'inertial' sublaag ligt.
- Type 2: turbulente grenslaag, wordt toegepast voor alle vlakken die contact hebben met het fluidum. Aangenomen wordt dat de grenlaag tussen de wand en de 'fully turbulent core' verdeeld is in een visceuze sublaag en een 'inertial' sublaag. Voor ieder roosterpunt naast de wand wordt bepaald in welke sublaag het roosterpunt ligt, de bijbehorende wandfunctie wordt gebruikt.

Wandfuncties Type 1

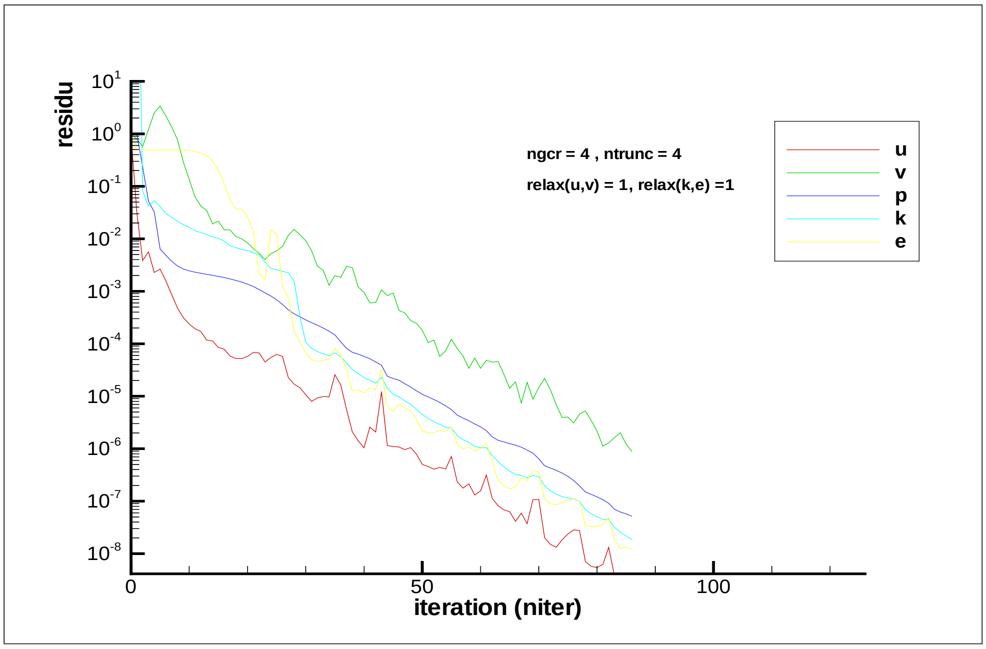
We lossen dit probleem met behulp van de bovengenoemde twee methoden op een rekenrooster (40×10).
We zullen ten eerste het GCR-SIMPLE algoritme toepassen voor verschillende relaxatiefactoren van k en ϵ. We volstaan hier met GCR-SIMPLE algoritme met $relax(u,v) = 1$ voor $ngcr = ntrunc = 4$ en $ngcr = ntrunc = 6$. De resultaten in figuur 8.1, 8.2, 8.3, 8.5 en 8.4 zijn verkregen.



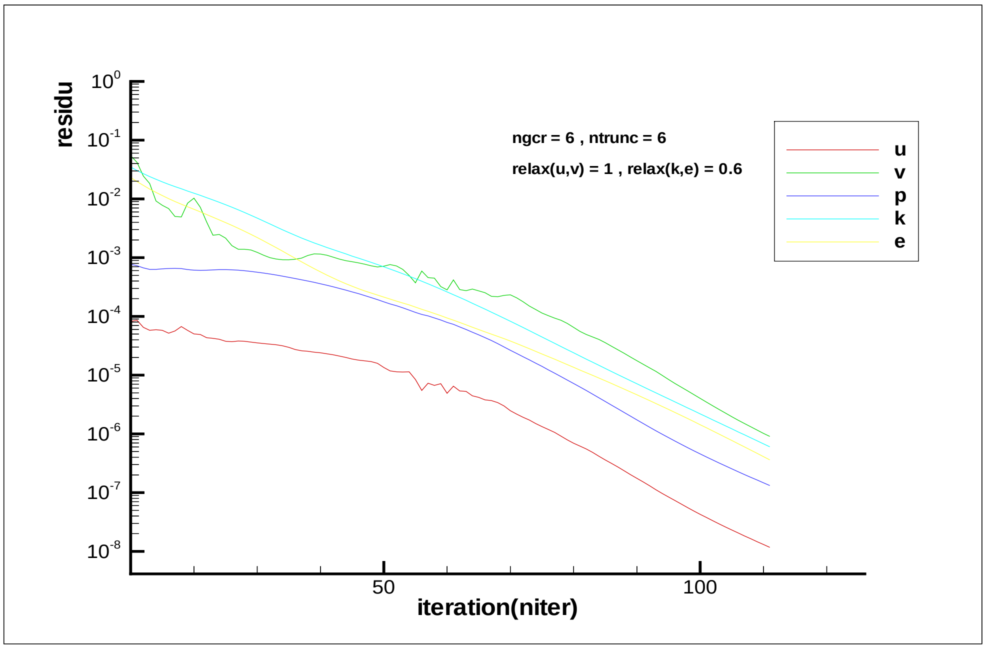
Figuur 8.1: GCR-SIMPLE methode, rooster (40×10), $tijd = 100(sec)$.



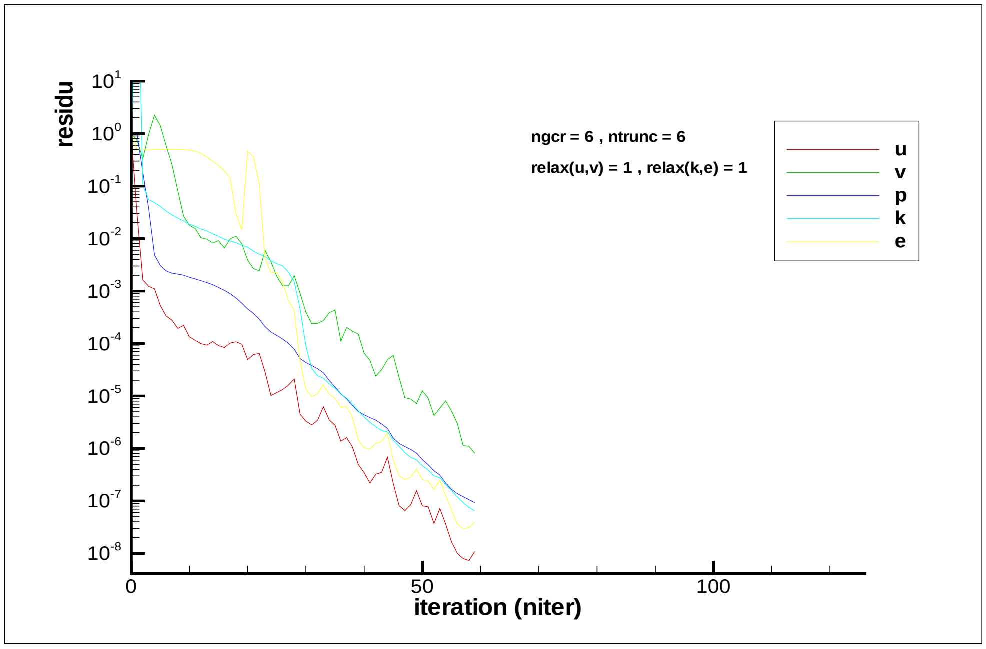
Figuur 8.2: GCR-SIMPLE methode, rooster (40×10), $tijd = 82(sec)$.



Figuur 8.3: GCR-SIMPLE methode, rooster (40×10), $tijd = 56(sec)$.



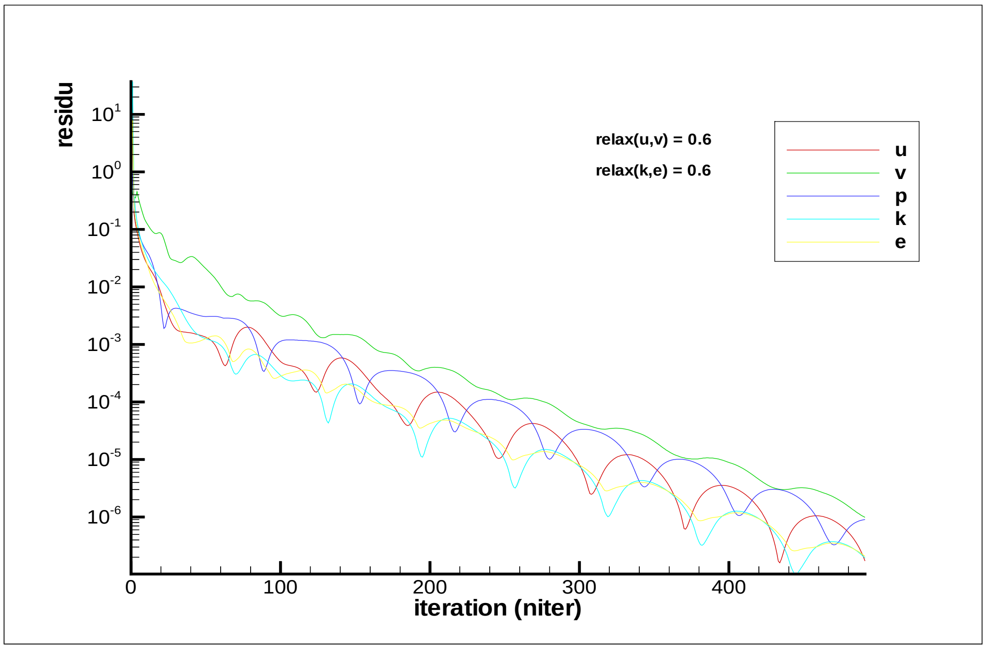
Figuur 8.4: GCR-SIMPLE methode, rooster (40×10), $tijd = 96.5(sec)$.



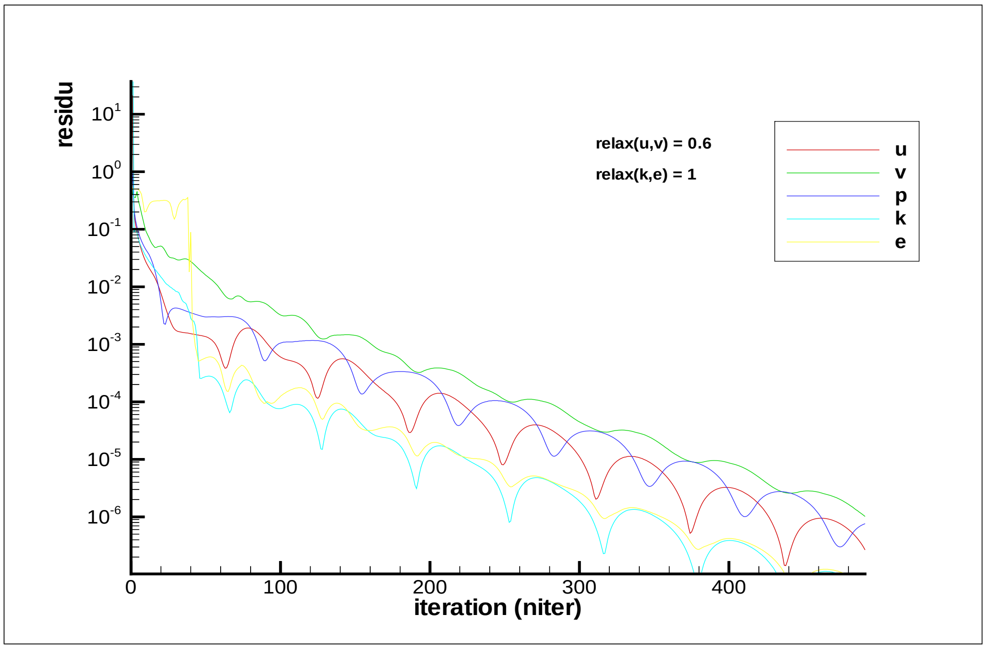
Figuur 8.5: GCR-SIMPLE methode, rooster (40×10), $tijd = 52(sec)$.

Merk op $relax(k,\epsilon)$ een groot invloed heeft op de convergentie van het GCR-SIMPLE algoritme. Dat wil zeggen dat de versnelling van de convergentie voor u, v en p wordt geremd door de convergentie van k en ϵ. Een mogelijke remedie hiervoor is de stelsels voor k en ϵ ook te versnellen.

We passen nu de SIMPLE methode voor $relax(k,\epsilon) = 0.6$ en $relax(k,\epsilon) = 1$ toe. De resultaten zijn in figuur 8.6 en 8.7 gegeven.



Figuur 8.6: SIMPLE methode, rooster (40×10),



Figuur 8.7: SIMPLE methode, rooster (40×10),

De verkregen resultaten met behulp van de SIMPLE methode (voor verschillende relaxatiefactoren) en het GCR-SIMPLE algoritme (voor verschillende waarden van $ngcr$) zijn respectievelijk weergegeven in tabel 8.4 en tabel 8.5

$relax(u,v)$	$relax(k,\epsilon)$	$niter$	$tijd$ (sec)
0.6	0.6	491	120
0.7	0.5	354	87
0.7	0.6	332	82

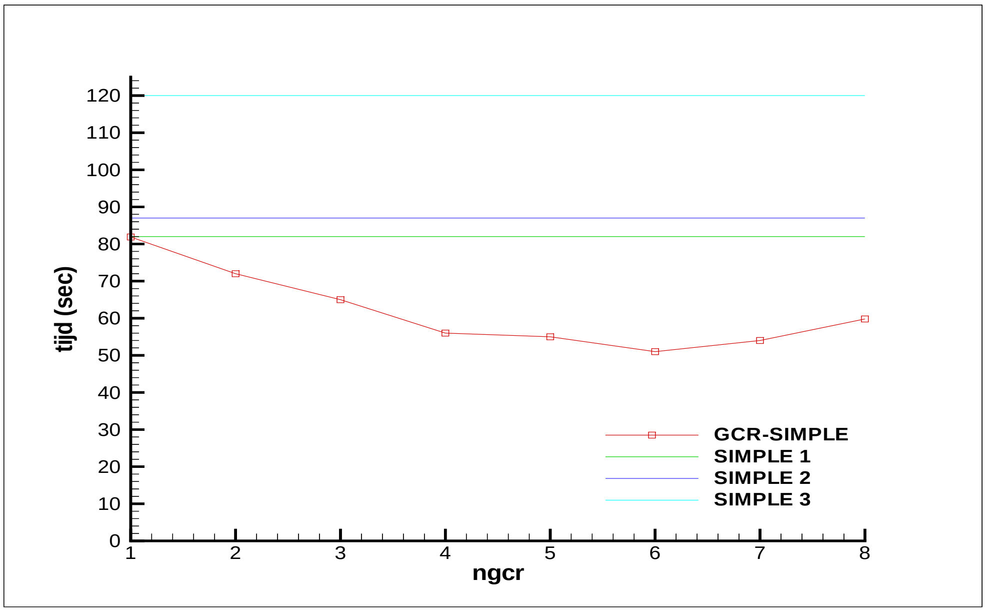
Tabel 8.4: Resultaten met de SIMPLE methode: stroming tussen twee vlakke platen, wandfuncties Type 1, rooster (40×10), $residu$ u, v, p, k, $\epsilon \leq 10^{-6}$.

Merk op dat we in tabel 8.4 naar de beste parameters ($relax$) voor SIMPLE hebben gezocht om de snelste convergentie te verkrijgen. De relaxatiefactoren 0.8, 0.9 en 1 leiden tot een 'deling door nul' in WISH3D subroutines. Door andere relaxatiefactoren tussen 0 en 1 neemt de SIMPLE methode nog meer tijd in beslag.

$ngcr$	$ntrunc$	$relax(u,v)$	$relax(k,\epsilon)$	$niter$	$tijd$ (sec)
1	1	1	1	206	81.9
2	2	1	1	146	72
3	3	1	0.6	149	85
3	3	1	1	113	65
4	4	1	1	86	56
4	4	1	0.6	129	86
5	5	1	1	70	55
6	6	1	1	57	51
7	7	1	1	53	54
8	8	1	1	51	59.8

Tabel 8.5: Resultaten met het GCR-SIMPLE algoritme: stroming tussen twee vlakke platen, wandfuncties Type 1, rooster (40×10), $residu$ u, v, p, k, $\epsilon \leq 10^{-6}$.

In tabel 8.5 is het GCR-SIMPLE algoritme voor verschillende waarden van $ngcr = ntrunc$ uitgevoerd. Maar er werd niet gezocht naar de beste relaxatiefactoren voor de op te lossen variabelen. In de meest gevallen is gebruik gemaakt van $relax(u,v,p,k,\epsilon) = 1$.



Figuur 8.8: Resultaten van GCR-SIMPLE en het beste resultaat van SIMPLE uit tabel 8.4 en tabel 8.5. SIMPLE 1 = (SIMPLE met $relax(u, v) = 0.7$ en $relax(k, \epsilon) = 0.6$), SIMPLE 2 = (SIMPLE met $relax(u, v) = 0.7$ en $relax(k, \epsilon) = 0.5$), SIMPLE 3 = (SIMPLE met $relax(u, v) = 0.6$ en $relax(k, \epsilon) = 0.6$).

Figuur 8.8 laat zien dat voor meerdere waarden van $ngcr$ levert het GCR-SIMPLE algoritme een snellere convergentie dan de SIMPLE methode (met de beste relaxatiefactor $relax = 0.7$ voor dit geval). We zullen later zien dat $relax = 0.7$ tot een 'deling door nul' in de WISH3D subroutines leidt. In de meeste gevallen wordt gebruik gemaakt van de default waarde ($relax = 0.6$) voor de relaxatiefactoren voor u en v.

Wandfuncties Type 2

We lossen dit probleem met behulp van de bovengenomde twee methoden op een rekenrooster (40×10). De verkregen resultaten met behulp van de SIMPLE methode en het GCR-SIMPLE algoritme zijn respectievelijk weergegeven in tabel 8.6 en tabel 8.7

$relax(u,v)$	$relax(k,\epsilon)$	$niter$	$tijd\ (sec)$
0.6	0.6	522	127
0.7	0.6	397	97

Tabel 8.6: Resultaten met de SIMPLE methode: stroming tussen twee vlakke platen, wandfuncties Type 2, rooster (40×10), $residu\ u,\ v,\ p,\ k,\ \epsilon \leq 10^{-6}$.

$ngcr$	$ntrunc$	$relax(u,v)$	$relax(k,\epsilon)$	$niter$	$tijd\ (sec)$
1	1	1	1	224	88
1	1	0.97	1	194	76
2	2	1	1	188	90
2	2	0.97	1	159	76
3	3	1	1	165	94
6	6	1	1	126	114

Tabel 8.7: Resultaten met het GCR-SIMPLE algoritme: stroming tussen twee vlakke platen, wandfuncties Type 2, rooster (40×10), $residu\ u,\ v,\ p,\ k,\ \epsilon \leq 10^{-6}$.

Uit de tabel 8.6 en tabel 8.7 is duidelijk te zien dat voor het geval van de wandfuncties Type 2 het GCR-SIMPLE algoritme (met $relax(k,\epsilon) = 1$) geen versnelling geeft behalve voor een goede keuze van de relaxatiefactoren. Andere waarde van $relax(k,\epsilon)$ kan betere resultaten opleveren. Dit aspect zou nog verder onderzocht moeten worden.

Verschillende roosters (wandfuncties Type 1):

Nu wordt bekeken hoe de versnelling van het GCR-SIMPLE zich gedraagt ten opzichte van de SIMPLE methode voor verschillende roosters. In de volgende tabel zien we het resultaat:

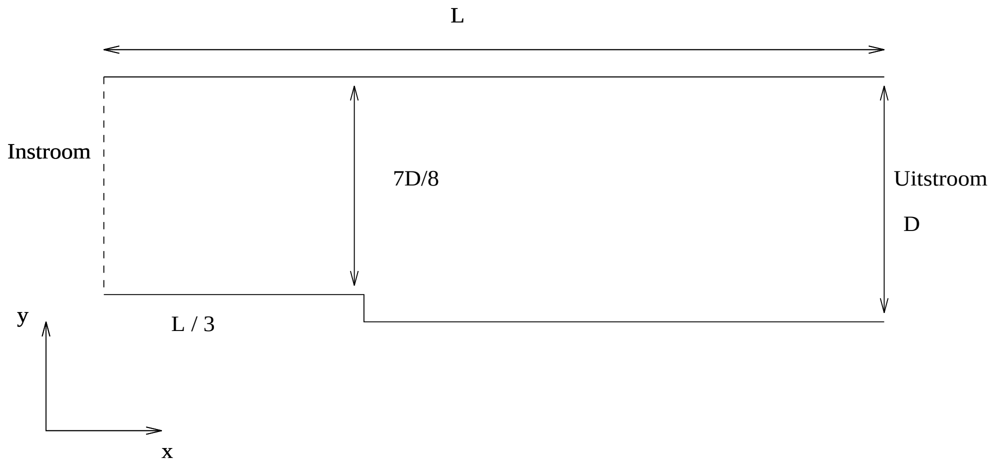
	rooster	niter	tijd (sec)
GCR-SIMPLE met $ngcr = 6,\ ntrunc = 6$	(40×10)	70	51
SIMPLE	(40×10)	332	82
GCR-SIMPLE met $ngcr = 6,\ ntrunc = 6$	(80×10)	128	209
SIMPLE	(80×10)	639	287

Tabel 8.8: Resultaten van de SIMPLE methode en het GCR-SIMPLE algoritme voor verschillende roostergroottes, $residu\ u,\ v,\ p,\ k,\ \epsilon \leq 10^{-6}$, voor GCR-SIMPLE $relax(u, v, k, \epsilon) = 1$, voor SIMPLE $relax(u, v) = 0.7$, $relax(p) = 1$ en $relax(k, \epsilon) = 0.6$, wandfuncties Type 1.

In tabel 8.8 is er niet gezocht naar de optimale waarde van $ngcr$ terwijl voor de SIMPLE methode de beste $relaxatiefactor$ wordt gebruikt.

8.2 Stroming in een backward facing step

Beschouw een stationaire luchtstroming in het gebied zoals geschetst in figuur 8.9.



Figuur 8.9: Stroming in een backward facing step.

De afmetingen zijn:

$$D = 20\ cm,$$
$$L = 500\ cm.$$

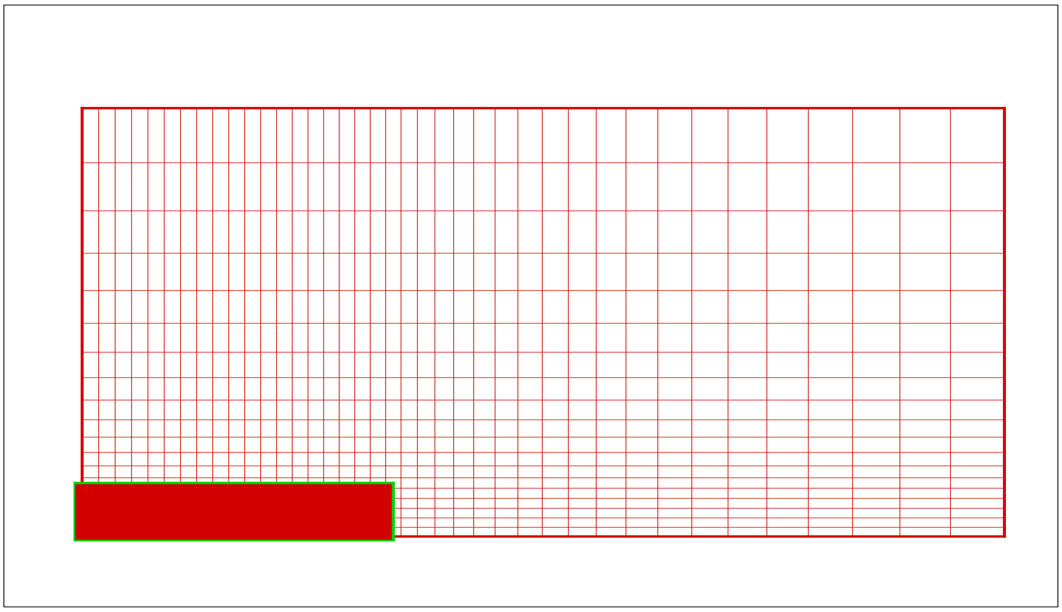
Randvoorwaarden:

variabele	instroomrand	uitstroomrand	vaste wanden
snelheid u	$u = u_0$	$\frac{\partial u}{\partial n} = 0$	$u = 0$
snelheid v	$v = 0$	$v = 0$	$v = 0$
druk p	$\frac{\partial p}{\partial x} = 0$	$p = 0$	$\frac{\partial p}{\partial n} = 0$
k	k = 0.005400	$\frac{\partial k}{\partial x} = 0$	wandfuncties
ϵ	$\epsilon = 0.00543365$	$\frac{\partial \epsilon}{\partial x} = 0$	wandfuncties

Tabel 8.9: Randvoorwaarden.

Voor de wandfuncties wordt in deze sectie gebruik gemaakt van Type 1.

We zullen de SIMPLE methode en het GCR-SIMPLE algoritme (zie [blz 53]) toepassen op dit testprobleem. Het rooster is gegeven in figuur 8.10



Figuur 8.10: Stroming in een backward facing step, Rooster (40×20).

Voor de SIMPLE methode en het GCR-SIMPLE algoritme zullen we de planez TDMA solver gebruiken.

De volgende resultaten zijn verkregen op een rooster (40×20).

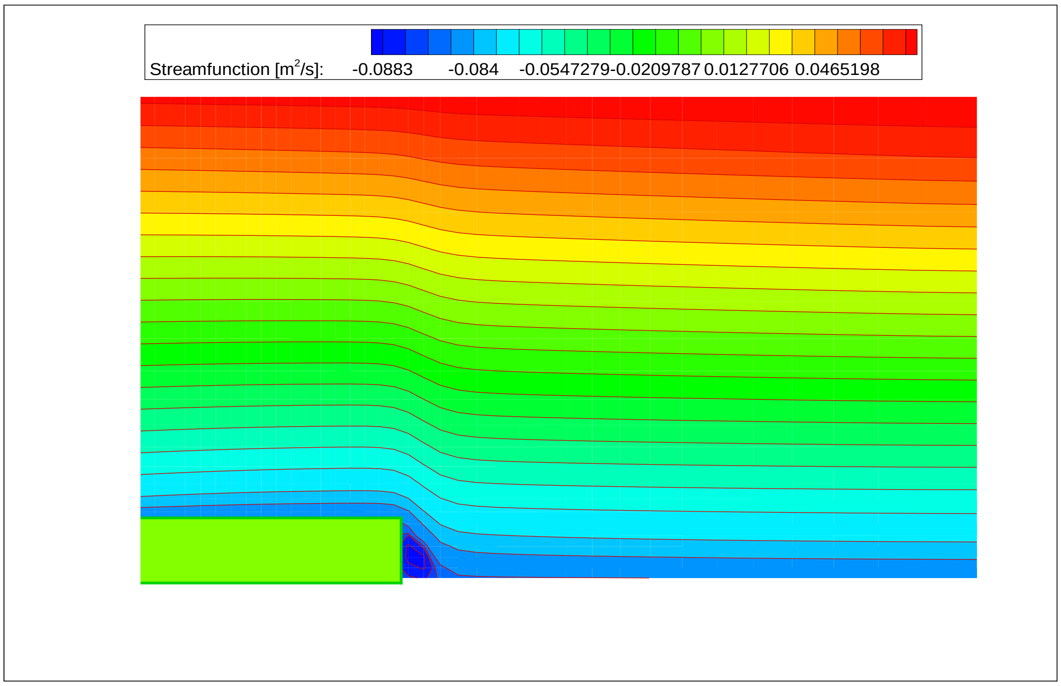
SIMPLE methode	niter = 559	tijd = 230 (sec)

Tabel 8.10: Resultaten van het SIMPLE algoritme, rooster (40×20), $relax(u, v) = 0.6$, $relax(k, \epsilon) = 0.6$.

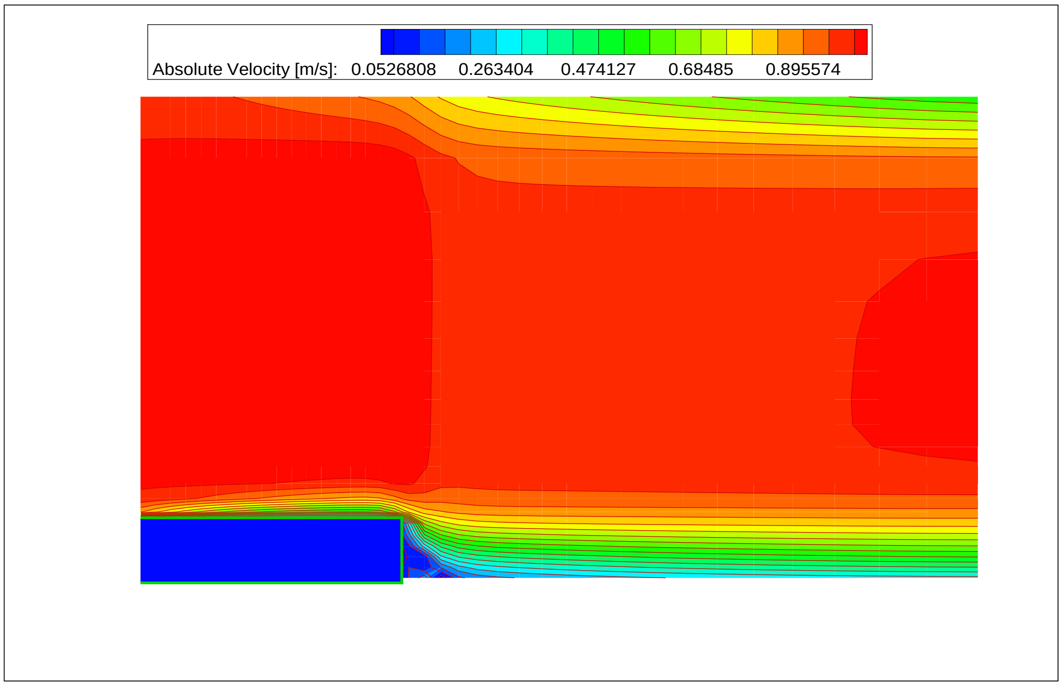
$ngcr$	2	3	4	5
$ntrunc$	2	3	4	5
$niter$	140	117	107	105
$tijd$	110	114	125	145

Tabel 8.11: Resultaten van het GCR-SIMPLE algoritme, rooster (40×20), $relax(u, v) = 1$, $relax(k, \epsilon) = 0.6$.

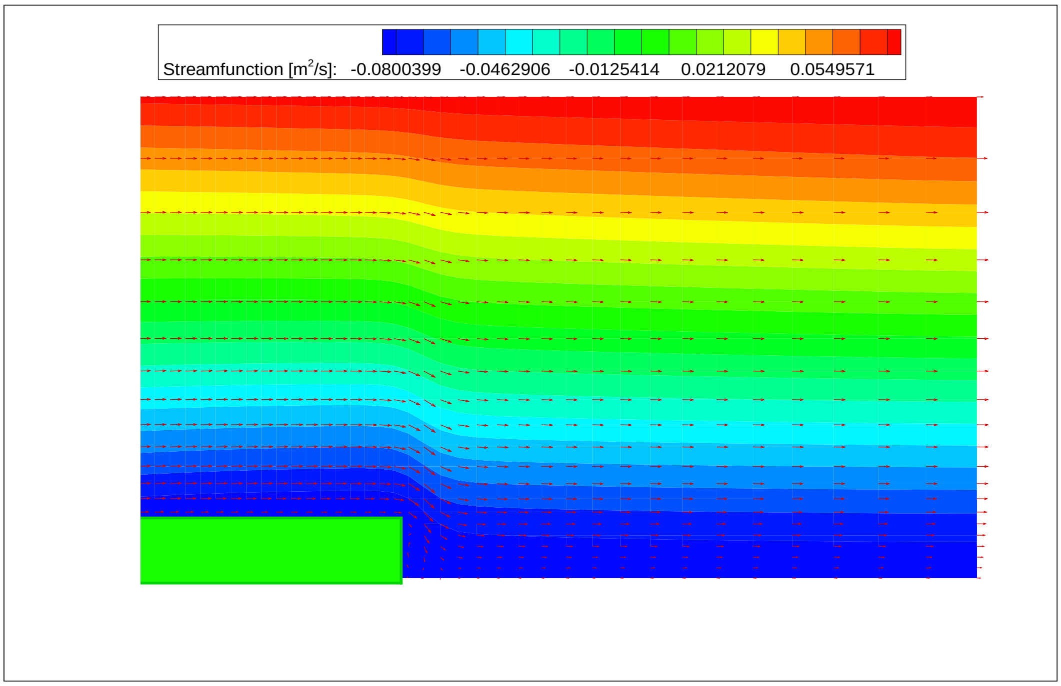
De absolute snelheid en de stroomfunctie berekend met het GCR-SIMPLE algoritme staan in de volgende figuren: De SIMPLE methode levert dezelfde resultaten.



Figuur 8.11: Stroming in een backward facing step, de stroomfunctie.



Figuur 8.12: Stroming in een backward facing step, de absolute snelheid.



Figuur 8.13: Stroming in een backward facing step, de stroomfunctie en een vector plot van de absolute snelheid.

Lijst van behandelde onderwerpen:

- Toepassing van het GCR-SIMPLE algoritme op turbulente stromingen
  - Stroming tussen twee vlakke platen
    - Wandfuncties Type 1
    - Wandfuncties Type 2
  - Stroming in een backward facing step

9 Conclusies en aanbevelingen voor verder onderzoek

9.1 Conclusies

In dit afstudeerwerk is onderzocht hoe de SIMPLE methode versneld kan worden met Krylov methoden. Er is onderzoek gedaan naar het oplossen van de gekoppelde en niet-lineaire stelsels van de vorm $\mathbf{A}(x)x = b$. Voor de Krylov methoden is in dit verslag voor de GCR (Generalized Conjugate Residuals) gekozen voor het oplossen van de lineaire stelsels van de vorm $\mathbf{A}x = b$. Voor de niet lineaire stelsels is gebruik gemaakt van Picard iteraties. Op deze manier is het GCR-SIMPLE algoritme ontwikkeld. Het is gebleken dat het aantal SIMPLE iteraties gereduceerd kan worden. We concluderen dat:

- Het GCR-SIMPLE algoritme leidt tot een snelle convergentie voor kleine waarden van $ntrunc$ ($\leq$ 10). Op deze manier is er niet veel extra geheugenruimte nodig voor het bewaren van $2 \times ntrunc$ vectoren in het Gram-Schmidt orthogonalisatie proces.
- Het GCR-SIMPLE algoritme leidt tot goede resultaten voor $ngcr = ntrunc$.
- Het toepassen van het GCR-SIMPLE algoritme op Stokes en Navier-Stokes problemen leidt tot een goede versnelling van de SIMPLE methode. Afhankelijk van het probleem, kan de GCR-SIMPLE methode ongeveer 2 tot 4 keer sneller zijn dan de SIMPLE methode zie hoofdstuk 7. In sommige gevallen is de GCR-SIMPLE methode niet beter doet dan de SIMPLE methode zie hoofstuk 7.
- Voor turbulentie problemen ($k - \epsilon$ model) heeft de convergentie van het k en ϵ invloed op de convergentie van het GCR-SIMPLE algoritme. De wandfuncties voor k en ϵ zijn ook van belang.

9.2 Aanbevelingen

In de loop van dit onderzoek en gemotiveerd door de verkregen resultaten zijn veel vragen aan de orde gekomen om het aantal SIMPLE iteraties nog verder te reduceren. De volgende onderwerpen kunnen verder onderzocht worden.

- Voor de Krylov methoden kan men naast de GCR methode ook de GM-RES (Generalized Minimal Residuals) methode gebruiken.

- In het GCR-SIMPLE algoritme de matrices $\mathbf{Q}_1$, $\mathbf{Q}_2$, $\mathbf{R}$, $\mathbf{G}_1$, $\mathbf{G}_2$, $\mathbf{G}_1^*$ en $\mathbf{G}_2^*$ zie [blz. 45] op de volgende manier bepalen:

  for $i = 0, 1, 2, \ldots, niter$ **do**
    bepaal de matrices $\mathbf{Q}_1$, $\mathbf{Q}_2$, $\mathbf{R}$, $\mathbf{G}_1$, $\mathbf{G}_2$, $\mathbf{G}_1^*$ en $\mathbf{G}_2^*$
    for $k = 1, 2, \ldots, ngcr$ **do**
      los op u, v en p
    end for
  end for

- De matrix $\mathbf{AB}$ in 4.19 preconditioneren door een geschikte preconditioneringsmatrix.

- Voor het oplossen van de niet lineaire stelsels kan men gebruik maken van de Newton methode in plaats van Picard iteraties. Het voordeel van de Newton iteratie methode is dat deze sneller convergeert. Beschouw een niet lineair stelsel vergelijkingen:

  $$F(x) = 0, \tag{9.1}$$

  waarin F een niet lineaire functie van $\mathbb{R}^{\mathbb{N}}$ naar $\mathbb{R}^{\mathbb{N}}$. De Newton methode toegepast op (9.1) resulteert in het volgende algoritme:

  kies $x^0 =$ beginschatting,
  for $n = 1, 2, \ldots,$tot convergentie, **do**
    a) los op $J(x^n)\delta = -F(x^n)$,
    b) $x^{n+1} = x^n + \delta$,
  end for

  waarin $J(x^n)$ de jacobiaan voorstelt. Om het stelsel a) op te lossen kunnen we gebruik maken van Krylov methoden (GCR, GMRES, enz...) zie [11].

- Het versnellen van de SIMPLER methode zie [6] met Krylov methoden.

- Uitbreiden van de twee dimensionale problemen naar het drie dimensionale geval.

- Gebruik maken van een Krylov methode voor niet linaire stelsels zie [13].

Lijst van Figuren

Lijst van Tabellen

10 Bibliografie

[1] **P. Wesseling**, *An Introduction To Multigrid Methods*, John Wiley and Sons, Chichester, 1992.

[2] **Patankar, S.V.**, *Numerical Heat Transfer and fluid Flow*, Hemisphere, New York, 1980.

[3] **L. Post**, *Modelling of Flow and Combustion in a Glass Melting Furnace*, Thesis, TU Delft, 1988.

[4] **F.T.M. Nieuwstadt**, Turbulentie: *Inleiding in de theorie en toepassingen van turbulente stromingen*, Epsilon, Utrecht, 1992.

[5] **G.P. Boerstoel**, *Modelling of Gas-Fired Furnaces*, Thesis, TU Delft, 1997.

[6] **H. Tazammourti**, *(Parallelle) Solvers voor CFD-code WISH3D*, Afstudeerverslag, TU Delft, 1997. http://dutita0.twi.tudelft.nl/users/vuik/numanal/tazammourti.html

[7] **S.C. Eisenstat, H.C. Elman and M.H. Schultz**, *Variational iterative methods for nonsymmetric syste ms of linear equations*, SIAM J. Numer. Anal., 20:345-357, 1983.

[8] **Y. Saad and M.H. Schultz**, *GMRES: A generalized minimal residual algorithm for solving nonsymmetric linear sys tems*, SIAM J. Sci. Stat. Comput., 7 (1986) 856-869.

[9] **H.A. van der Vorst and C. Vuik**, *GMRESR: A family of nested GMRES methods*, Num. Lin. Alg. Appl., 1:369-386, 1994.

[10] **H.A. van der Vorst and C. Vuik**, *The superlinear convergence behaviour of GMRES*, J. Comput. Appl. Math., 48:3 27-341, 1993.

[11] **P.N. Brown and Y. Saad**, *Hybrid Krylov methods for nonlinear systems of equations*, SIAM J. Sci. Stat. Comput ., 11:450-481, 1990.

[12] **H.A. van der Vorst and C. Vuik**, *Voortgezette numerieke lineaire algebra*, Collegediktaat, TU Delft, Oktober 19 96.

[13] **C.G. Broyden**, *A class of methods for solving nonlinear simultaneous equations*, Math. Comput. 19 pp. 577-593, 1965.