Het oplossen van Tikhonov regularisatie in algemene vorm met behulp van samengevoegde bidiagonalisatie

Jetske Kemper (5644488)
11 januari 2019
Bachelorscriptie
N.A Luiken en T. van Leeuwen

Universiteit Utrecht

Inhoudsopgave

1 Introductie **1**
1.1 Achtergrondkennis ontbindingen . . . 3

2 Het kleinstekwadraten probleem **5**
2.1 Standaard oplossingsmethode . . . 5
2.1.1 SVD Analyse . . . 7
2.2 Tikhonov Regularisatie . . . 7

3 Standaard Tikhonov regularisatie **10**
3.1 SVD analyse . . . 10
3.2 Projectie methode . . . 11
3.3 Lanczos bidiagonalisatie . . . 12
3.4 Projectiemethode met bidiagonalisatie . . . 14
3.5 Hybride methode . . . 15

4 Algemene Tikhonov regularisatie **18**
4.1 GSVD . . . 19
4.2 Tikhonov uitgedrukt in GSVD . . . 21
4.3 Samengevoegde bidiagonalisatie . . . 22
4.4 Tikhonov met samengevoegde bidiagonalisatie . . . 27
4.5 Een efficiëntere manier voor het gebruik . . . 29
4.5.1 Samengevoegde Lanczos bidiagonalisatie 2.0 . . . 29
4.5.2 Tikhonov met samengevoegde bidiagonalisatie 2.0 . . . 32
4.6 Juistheid oplossing . . . 35

5 Numerieke resultaten **37**
5.1 2D beeld vervaging probleem . . . 37
5.2 Helioseismologie . . . 40
5.2.1 Zonder ruis . . . 40
5.3 Met ruis . . . 41

6 Conclusie **42**

7 Referenties **43**

Samenvatting

In dit artikel geven we een algoritme voor het berekenen van een benadering van een slecht gesteld probleem wat geregulariseerd wordt door middel van Tikhonov regularisatie. Zowel de situatie waarbij $L = I$ en $L \neq I$ wordt behandeld. Voor de eerste situatie wordt gebruik gemaakt van een beneden bidiagonalisatie algoritme om de projectie ruimte te vinden. Daarna bekijken we een methode voor het oplossen van Tikhonov regularisatie met $L \neq I$. Hierbij maken we gebruik van een samengevoegd bidiagonalisatie algoritme. We bekijken daarbij vooral de resultaten van het artikel "A projection-based approach to gereral-form Tikhonov regularization" van Kilmer, Hanssen en Español waarbij we meer informatie en uitleg geven bij de bewijzen.

1 Introductie

In dit artikel zijn we geïnteresseerd in het vinden van een oplossing van de volgende vorm:

$$Ax \approx b = b_{\text{exact}} + e, \quad b_{\text{exact}} = Ax_{\text{exact}}.$$

Het oplossen van deze vergelijking is te herschrijven als

$$\min_x ||Ax - b||_2.$$

Dit probleem wordt het kleinstekwadraten probleem genoemd. Wanneer het probleem slecht gesteld is, zal de directe methode voor het oplossen van het kleinstekwadraten probleem geen goede benadering geven. Bij een slecht gesteld probleem dalen de singuliere waarden van A heel snel. Bovendien liggen de kleinste singuliere waarden rond machine precisie. Hierdoor kan de oplossing erg verstoord worden. Dit resulteert in een oplossing x die ver van x_{exact} afligt.

Een voorbeeld van een probleem is het *Gravity* probleem uit [1]. Dit model is een discretisatie van een *gravity surveying* probleem. Hierbij geeft $f(t) = \sin(\pi t) + 0.5\sin(2\pi t)$ de verdeling van de massa weer gelokaliseerd op diepte 0.25 en waarbij de verticale component het zwaartekracht veld $g(s)$ gemeten wordt aan de oppervlakte. Het probleem is een Fredholm integraal van eerste soort. Hierbij is de kern gelijk aan $K(s,t) = 0.25(0.25^2 + (s-t)^2)^{-\frac{3}{2}}$ waarbij het interval voor t en s allebei $[0,1]$ is. Er geldt dat $A \in \mathbb{R}^{1000\times 1000}$ en dat x en b bijbehorende lengte hebben.

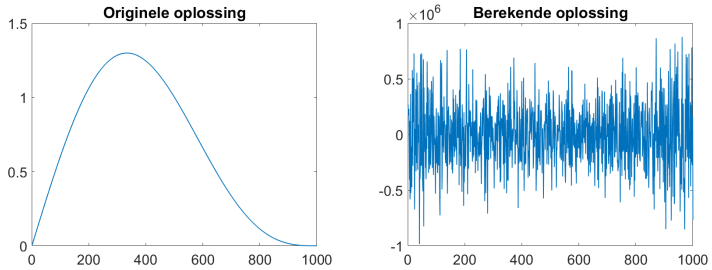
In figuur 1 zien we zowel de originele oplossing van het *Gravity* probleem en de berekende oplossing. Beide figuren komen totaal niet overeen.

Een veelvoorkomende methode om dit effect tegen te gaan is door gebruik te maken van Tikhonov regularisatie:

$$x^\lambda = \underset{x}{\operatorname{argmin}} \{ \ ||Ax - b||_2^2 + \lambda ||Lx||_2^2 \ \}. \tag{1}$$

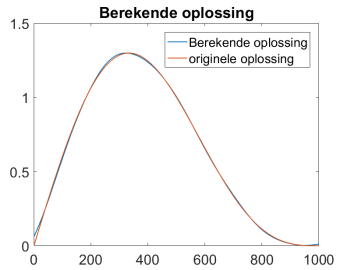
Hierbij wordt λ de regularisatie parameter genoemd en is $L \in \mathbb{R}^{p\times n}$. Belangrijk is dat x^λ een goede benadering is voor x_{exact} als λ goed gekozen wordt. Wanneer λ verkeerd gekozen wordt is de oplossing dat niet.

De rechtstreekse oplossingsmethode kan erg duur worden voor grote A en L. Dit heeft er vooral mee te maken dat het oplossen van (1) kostbaar is en dat hij vaak meerdere keren berekend moet worden om de juiste λ te vinden. Dit is de reden dat er methodes worden bedacht om het proces te verbeteren.



Figuur 1: De originele afbeelding van het probleem *Gravity* is links weergegeven. Rechts zien we de oplossing verkregen door het kleinstekwadraten probleem direct op te lossen. Aan de data b is een ruis vector e toegevoegd waarbij de waarden van e normaal verdeeld zijn met gemiddelde 0 en standaarddeviatie 10^{-2}.

De situatie met $L = I$ is al veel bestudeerd. Deze vorm van Tikhonov regularisatie wordt de algemene vorm genoemd. Een methode is om gebruik te maken van een projectie methode. Het probleem wordt dan geprojecteerd naar een probleem van lagere dimensie. Op deze manier kost het minder operaties om de Tikhonov vergelijking op te lossen. Het vinden van de juiste λ is hierdoor minder belastend. Nadat de juiste λ en de oplossing van het geprojecteerde probleem gevonden zijn, kan er terug gerekend worden naar de oorspronkelijke ruimte. Een van de methodes maakt gebruik van de beneden Lanczos bidiagonalisatie van de matrix A en start vector b. In figuur 2 zien we deze methode uitgevoerd op het *Gravity* probleem.

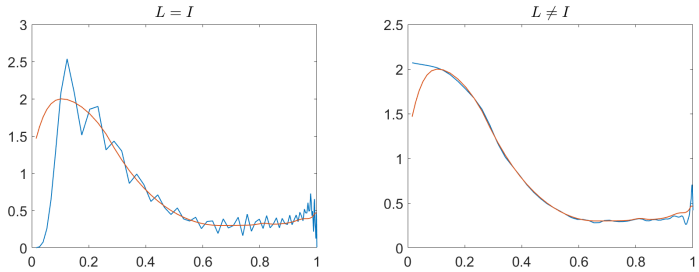


Figuur 2: De oplossing van het *Gravity* probleem verkregen door middel van het algoritme wat gebruik maakt van Lanczos bidiagonalisatie.

Er zijn echter problemen waarbij $L = I$ geen goede benadering geeft voor x_{exact}. Een voorbeeld is het probleem Helio uit [2]. Dit probleem is een probleem uit de helioseismologie. In figuur 3 zien we dat er een andere L bestaat ongelijk aan de identiteitsmatrix waarbij de reconstructie beter is.

We willen daarom kijken naar situaties waarbij $L \neq I$. We noemen dit de algemene vorm van Tikhonov regularisatie. Een van de meest gebruikte methodes om deze op te lossen is door het probleem om te schrijven naar de standaard vorm. Op deze vorm kan dan de bovenstaande tactiek uitgevoerd worden. Het probleem is dat de transformatie van algemeen naar standaard vorm ook erg duur kan zijn.

Dit is de reden dat we naar een ander algoritme bekijken. We zullen een algoritme bekijken dat voorgesteld is door Kilmer, Hanssen en Español in "A projection-based approach to gereral-form Tikhonov regularization" . Hierbij wordt gebruik gemaakt van een samengevoegd bidiagonalisatie



Figuur 3: Links zien we de beste oplossing van de Tikhonov vergelijking met $L = I$ en rechts zien we de beste oplossing van de Tikhonov vergelijking met $L \neq I$. Hierbij is voor L de discrete benadering van de tweede afgeleide genomen.

algoritme. Dit algoritme maakt gebruik van beneden Lanczos bidiagonalisatie en boven Lanczos bidiagonalisatie.

In hoofdstuk 2 geven we wat basisinformatie over het kleinstekwadraten probleem. We vertellen uitgebreider waarom regularisatie nodig is en hoe Tikhonov regularisatie eruit ziet. Vervolgens bekijken we in hoofdstuk 3 een methode om standaard vorm van Tikhonov regularisatie op te lossen. We baseren ons daarbij vooral op [3] en [4]. We zullen de resultaten hieruit benoemen en verder uitwerken. We leggen het principe van projectie uit en leggen de link met Krylov deelruimten. Vervolgens leggen we Lanczos bidiagonalisatie uit en passen deze toe op het oorspronkelijke kleinstekwadraten probleem. Hierna beschrijven we de methode met zowel Tikhonov regularisatie als projectie. We behandelen de standaard vorm zodat we meer inzicht hebben in de algemene vorm. In hoofdstuk 4 behandelen we de algemene vorm van Tikhonov regularisatie. We baseren ons hierbij op [5]. Wederom bekijken we hiervan de resultaten en geven we meer uitleg. We beginnen met het definiëren van de GSVD. Deze ontbinding zullen we veel gebruiken in het verloop van het hoofdstuk. Vervolgens definiëren we het samengevoegde bidiagonalisatie algoritme en vertellen we hoe we het kunnen gebruiken. Vervolgens maken we nog een aantal belangrijke opmerkingen om het efficiënt te implementeren. Als laatste bekijken we naar de ruimte waarin onze gevonden oplossing ligt. In hoofdstuk 5 bekijken we vervolgens een aantal numerieke resultaten van het samengevoegde bidiagonalisatie algoritme. We vergelijken het hierbij met het LSQR algoritme. Als laatste geven we in hoofdstuk 6 onze conclusie.

1.1 Achtergrondkennis ontbindingen

Definitie 1.1 (Singuliere waarden ontbinding)**.** [4, p.70] Stel dat $A \in \mathbb{R}^{m\times n}$ een matrix is met rang r. Dan geldt dat er matrices U, V en Σ bestaan zo dat

$$A = U\Sigma V^T.$$

Hierbij geldt dat $U \in \mathbb{R}^{m\times m}$ en $V \in \mathbb{R}^{n\times n}$ twee orthonormale matrices zijn en $\Sigma \in \mathbb{R}^{m\times n}$ een diagonaal matrix waarbij de eerste r elementen op de diagonaal aangegeven worden met $\sigma_1, \sigma_2, \ldots, \sigma_r$. Hierbij geldt dat $\sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_r \geq 0$. Deze elementen worden ook de singuliere waarden van A genoemd. Alle andere elementen op de diagonaal zijn nul. Deze ontbinding wordt de singuliere waarden ontbinding genoemd bekend onder de Engelse afkorting SVD.

Definitie 1.2 (QR ontbinding)**.** [6, p.154] Zij $A \in \mathbb{R}^{m\times n}$ met $m \geq n$ een matrix met rang n. Dan bestaan er een orthogonale matrix $Q \in \mathbb{R}^{m\times m}$ en een bovendriehoeksmatrix $R \in \mathbb{R}^{n\times n}$ zo dat:

$$A = Q \begin{bmatrix} R \\ 0 \end{bmatrix}.$$

Deze ontbinding wordt de QR ontbinding genoemd.

We merken op dat R een niet singuliere matrix is en dat zijn inverse dus bestaat.
Definitie 1.3 (Gereduceerde QR ontbinding)**.** [6, p.155] Zij $A \in \mathbb{R}^{m\times n}$ met $m \geq n$ een matrix met $\text{rang}(A) = n$. Dan bestaan er een matrix $Q \in \mathbb{R}^{m\times n}$ met orthonormale kolommen en een bovendriehoeksmatrix $R \in \mathbb{R}^{n\times n}$ zo dat:

$$A = QR$$

Deze ontbinding wordt de gereduceerde QR ontbinding genoemd.

Er geldt dat definitie 1.3 gebruik maakt van de de nulrijen in definitie 1.2. Op deze manier is het mogelijk om een matrix Q te vinden die niet meer vierkant is. We merken op dat wanneer $m > n$, Q niet meer inverteerbaar is. De gereduceerde QR ontbinding wordt zo genoemd omdat de matrix Q minder groot is en daardoor minder opslag nodig heeft. Ook kost het minder tijd om hem te reconstrueren.

2 Het kleinstekwadraten probleem

In dit hoofdstuk geven we meer informatie over het kleinstekwadraten probleem, waarom de voor de hand liggende niet werkt en hoe we dit kunnen oplossen met Tikhonov regularisatie. Dit hoofdstuk kan gezien worden als een inleiding op de komende twee hoofdstukken.

We zijn geïnteresseerd in het oplossen van het volgende lineaire systeem:

$$Ax = b$$

waarbij $A \in \mathbb{R}^{m \times n}$, $x \in \mathbb{R}^n$ en $b \in \mathbb{R}^m$ met $m \geq n$ en $\text{Rang}(A) = n$. Het probleem van het vinden van x wordt het inverse probleem genoemd. Wanneer A vierkant en inverteerbaar is, wordt x gegeven door $A^{-1}b$. Wanneer $m > n$ dan geldt dat het systeem overbepaald is. De vergelijking $Ax = b$ heeft dan vaak geen oplossing. We zoeken dan een x zo dat de afstand van Ax tot b minimaal is. In andere woorden: we willen dat de norm van het residu $r = Ax - b$ minimaal is. Dit leidt tot het volgende minimalisatie probleem:

$$\min_x ||Ax - b||_2. \tag{2}$$

2.1 Standaard oplossingsmethode

De meest voor de hand liggende methode om (2) op te lossen is door het minimum van $||r||_2$ te vinden. We volgen hierbij de redenatie van [6, p.142-143].

Als eerste merken we op dat (2) te herschrijven valt als:

$$\min_x \frac{1}{2}||Ax - b||_2^2. \tag{3}$$

Door de definities van de 2-norm uit te schrijven vinden we dat

$$\frac{1}{2}||Ax - b||_2^2 = \frac{1}{2}\sum_{i=1}^{m}\left(b_i - \sum_{j=1}^{n} a_{i,j}x_j\right)^2.$$

Om het minimum van $\frac{1}{2}||Ax - b||_2^2$ te vinden moet tenminste gelden dat de eerste afgeleide naar elke component van x gelijk is aan nul. Dit geeft voor $k = 1, 2, \ldots, n$:

$$\frac{\partial}{\partial x_k}\frac{1}{2}||Ax - b||_2^2 = \sum_{i-1}^{m}\left[\left(b_i - \sum_{j=1}^{n} a_{i,j}x_j\right)(-a_{i,k})\right] = 0$$

Dit laatste kan herschreven worden als

$$\sum_{i=1}^{m} a_{i,k}\sum_{j=1}^{n} a_{i,j}x_j = \sum_{i=1}^{m} a_{i,k}b_i.$$

In matrix notatie is dit gelijk aan $A^TAx = A^Tb$. Wanneer A volledige kolomrang heeft is dit inderdaad een uniek minimum. De vergelijking $A^TAx = A^Tb$ wordt ook wel de normaal vergelijking genoemd. Wanneer A volledige kolomrang heeft, betekent dit dat

$$x = (A^TA)^{-1}A^Tb. \tag{4}$$

Een begrip dat veel voorkomt is de Moore-Penrose inverse, ook wel de pseudoinverse genoemd. Een matrix A heeft alleen een inverse als hij niet singulier en vierkant is. De Moore-Penrose inverse is een unieke matrix X die bestaat voor elke eindige matrix A. Deze matrix voldoet aan de Penrose vergelijkingen en wordt vaak weergeven als $A^\dagger$. Wanneer A een inverse heeft dan geldt dat $A^{-1} = A^\dagger$. Wanneer A volledige kolomrang heeft geldt dat $A^\dagger = (A^T A)^{-1} A^T$.[7] We kunnen (4) dus ook herschrijven als $A^\dagger b = x$. We noemen deze oplossing de pseudoinverse oplossing en het oplossen van (4) noemen we de pseudoinverse methode.

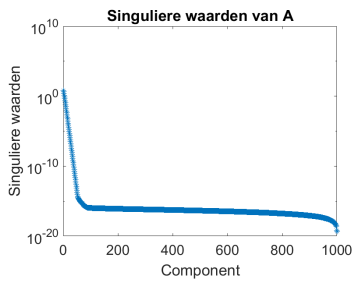
Vanwege onnauwkeurigheden in het verkrijgen van de data is b niet exact. Er is vaak sprake van een fout veroorzaakt door bijvoorbeeld meetfouten. We zeggen daarom dat $b = b_{\text{exact}} + e$ waarbij e een onbekende fout is. We definiëren b_{exact} als $b_{\text{exact}} = Ax_{\text{exact}}$. Hierdoor krijgen we $Ax = Ax_{\text{exact}} + e$.

De volgende ongelijkheid geeft een bovengrens voor de relatieve fout. [6, p.127-130]

$$\frac{||x_{\text{exact}} - x||_2}{||x_{\text{exact}}||_2} \leq \text{Cond}(A) \frac{||e||_2}{||b_{\text{exact}}||_2} \tag{5}$$

Hierbij is $\text{Cond}(A)$ het conditie nummer van A. Deze wordt bij gebruik van de 2-norm gegeven door $\frac{\sigma_1(A)}{\sigma_r(A)}$ waarbij $\sigma_1(A)$ en $\sigma_r(A)$ respectievelijk de grootste en de kleinste singuliere waarde van A zijn.

Bij slecht gestelde problemen is het conditie nummer heel hoog. Dit komt omdat het verschil in singuliere waarden heel groot is. Het gevolg hiervan is dat de gevonden x ver van de exacte waarde x_{exact} kan liggen. Ondanks dat (5) een bovengrens is is de praktijk dat x inderdaad ver van x_{exact} ligt. Hieruit volgt dat x geen nuttige oplossing is. Dit is het effect wat we zagen in 1 in de inleiding. De berekende oplossing is hierbij verkregen door de pseudoinverse. Het probleem *Gravity* is een slecht gesteld probleem. Dit zien we goed wanneer we naar de singuliere waarden van A kijken in figuur:



Figuur 4: De singuliere waarde ontbinding van de matrix A behorende bij het *Gravity* probleem.

2.1.1 SVD Analyse

Een andere manier om dit goed te zien is door middel van de SVD van A te bekijken. We herinneren ons dat de SVD van A gelijk is aan $A = U\Sigma V^T$. Wanneer we dit in (4) invullen zien we:

$$\begin{aligned} x &= (V\Sigma^T U^T U\Sigma V^T)^{-1} V\Sigma^T U^T b \\ &= (V\Sigma^T \Sigma V^T)^{-1} V\Sigma^T U^T b \\ &= V(\Sigma^T\Sigma)^{-1}\Sigma^T U^T b \\ &= V\Sigma^{-1}U^T b \\ &= \sum_{i=1}^{n} v_i \frac{u_i^T b}{\sigma_i} \end{aligned}$$

Bij slecht gestelde problemen geldt dat het verschil tussen de grootste singuliere waarden en de kleinste singuliere waarden heel groot is en vaak is er een sprong in de waardes van de singuliere waarden te zien. Voor de kleine waardes wordt de term $v_i \frac{u_i^T b}{\sigma_i}$ heel groot. Het probleem is dat deze σ_i vaak zo klein zijn dat afrond fouten een enorme invloed hebben. Het gevolg hiervan dat een oplossing verkregen door deze manier enorm gevoelig is voor verstoringen. Hierdoor kan de oplossing erg afwijken van de exacte oplossing.

Een oplossing hiervoor is regularisatie van het probleem. Bij regularisatie wordt het oorspronkelijke probleem vervangen door een probleem wat er erg op lijkt. Dit nieuwe probleem is dan wel goed oplosbaar. Vaak gebeurt dit door middel van nieuwe informatie die toegevoegd wordt aan het oorspronkelijke probleem. Deze informatie is vaak niet van te voren bekend.

2.2 Tikhonov Regularisatie

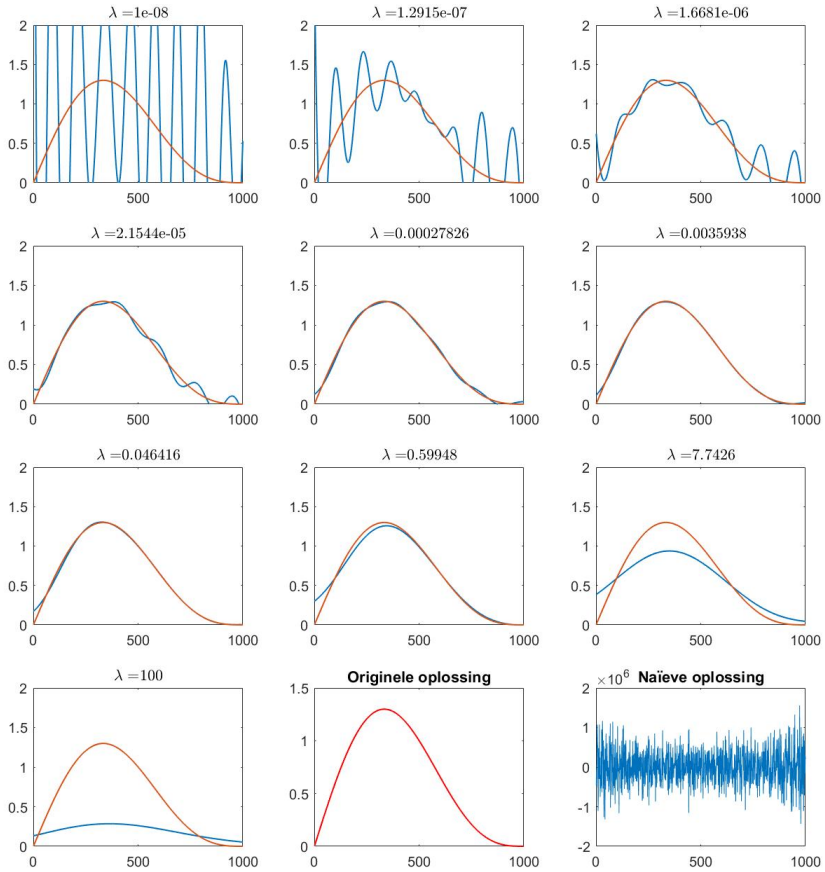
Een van de bekendste regularisatie methoden is Tikhonov regularisatie.
Definitie 2.1 (Tikhonov regularisatie)**.** [3, p.172] De algemene vorm van Tikhonov regularisatie wordt als volgt gegeven:

$$x^\lambda = \underset{x}{\operatorname{argmin}} \{ \ ||Ax - b||_2^2 + \lambda ||Lx||_2^2 \ \}$$

waarbij $\lambda \in \mathbb{R}$ en $L \in \mathbb{R}^{p\times n}$.

We zien dat de oplossing nu afhankelijk is van een positieve $\lambda \in \mathbb{R}$ en $L \in \mathbb{R}^{p\times n}$ waarbij er op p geen restricties zitten. Hierbij wordt λ de regularisatie parameter genoemd. Deze parameter bepaalt de verhouding tussen de termen $||Ax - b||_2^2$ en $||Lx||_2^2$. De correctheid van de oplossing x^λ hangt af van de gekozen λ. Wanneer deze niet goed gekozen wordt zal de oplossing niet juist zijn. Dit zien we in figuur 5. Voor een te grote λ heeft de regularisatie te veel effect. De de oplossing wordt te plat gestreken. Voor een te kleine λ heeft de regularisatie te weinig effect. Het probleem komt dan dichter bij de pseudoinverse oplossing te liggen. Het is dus belangrijk om een goede λ te kiezen. Helaas is dit eenvoudig. Een methode voor het vinden van de juiste λ is om verschillende waarden te proberen. Het probleem is dat de berekening vaak kostbaar is en dat er vaak meerdere λ geprobeerd moeten worden. Een andere manier zijn de zogenaamde parameter keuze methodes. Deze methodes bepalen welke λ het beste is. Een aantal veel voorkomende zijn het l-kromme criterium en het discrepantie principe.

Voorbeelden van L zijn de identiteit matrix en de discrete benadering van de eerste en de tweede afgeleide. Voor sommige problemen zorgt een goede keuze van L voor een aanzienlijk betere



Figuur 5: Het effect van λ op de oplossing van x^λ. Het weergegeven probleem is wederom *Gravity*. De rode lijn geeft de exacte oplossing weer. De grafiek in de rechter benedenhoek geeft de pseudoinverse oplossing weer.

reconstructie. Om deze reden wordt niet altijd voor de identiteit matrix gekozen. De keuze voor L is afhankelijk van het probleem en wordt hierop dan ook gebaseerd. In figuur 3 zien we dit terug. De vergelijking (7) is hierbij 2 keer opgelost. Een keer met $L = I$ en een keer waarbij L de discrete benadering is van de tweede afgeleide. Hierbij is λ bij beide zo gekozen zodat ze optimaal waren. Voor de eerste situatie is dit $\lambda = 1.5999 \cdot 10^{-6}$ en voor de tweede $\lambda = 2.5595 \cdot 10^{-4}$. We zien dat de situatie waarbij $L = I$ geen goede benadering geeft. De andere situatie geeft een aanzienlijk betere benadering.

We merken op dat we de Tikhonov vergelijking kunnen herschrijven als:

$$x^\lambda = \underset{x}{\operatorname{argmin}} \left\| \begin{bmatrix} A \\ \sqrt{\lambda} L \end{bmatrix} x - \begin{bmatrix} b \\ 0 \end{bmatrix} \right\|_2 \tag{6}$$

Bewijs: Voor het bewijs maken we gebruik van $||a||_2^2 = a^T a$ waarbij a een vector is. Dit geeft ons de volgende afleiding:

$$\begin{aligned}
\left|\left|\begin{bmatrix} A \\ \sqrt{\lambda}L \end{bmatrix} x - \begin{bmatrix} b \\ 0 \end{bmatrix}\right|\right|_2^2 &= \left|\left|\begin{bmatrix} Ax \\ \sqrt{\lambda}Lx \end{bmatrix} - \begin{bmatrix} b \\ 0 \end{bmatrix}\right|\right|_2^2 \\
&= \left|\left|\begin{bmatrix} Ax - b \\ \sqrt{\lambda}Lx \end{bmatrix}\right|\right|_2^2 \\
&= \begin{bmatrix} Ax - b \\ \sqrt{\lambda}Lx \end{bmatrix}^T \begin{bmatrix} Ax - b \\ \sqrt{\lambda}Lx \end{bmatrix} \\
&= (Ax - b)^T(Ax - b) + (\sqrt{\lambda}Lx)^T(\sqrt{\lambda}Lx) \\
&= (Ax - b)^T(Ax - b) + \lambda(Lx)^T(Lx) \\
&= ||Ax - b||_2^2 + \lambda||Lx||_2^2
\end{aligned}$$

Verder geldt dat

$$\min \left|\left|\begin{bmatrix} A \\ \sqrt{\lambda}L \end{bmatrix} x - \begin{bmatrix} b \\ 0 \end{bmatrix}\right|\right|_2^2 = \min_x \left|\left|\begin{bmatrix} A \\ \sqrt{\lambda}L \end{bmatrix} x - \begin{bmatrix} b \\ 0 \end{bmatrix}\right|\right|_2$$

Hieruit concluderen we dat we inderdaad de Tikhonov vergelijking kunnen schrijven als (6) □

We zullen vanaf nu vooral gebruik maken van (6) als weergave van het Tikhonov probleem waarbij we merken dat deze weergave ook een kleinstekwadraten probleem is.

Dit probleem heeft de volgende normaal vergelijking:

$$\begin{bmatrix} A \\ \sqrt{\lambda}L \end{bmatrix}^T \begin{bmatrix} A \\ \sqrt{\lambda}L \end{bmatrix} x = \begin{bmatrix} A \\ \sqrt{\lambda}L \end{bmatrix}^T \begin{bmatrix} b \\ 0 \end{bmatrix}.$$

Wanneer de matrix $\begin{bmatrix} A \\ L \end{bmatrix}$ volle kolomrang heeft bestaat de inverse van $\begin{bmatrix} A \\ \sqrt{\lambda}L \end{bmatrix}^T \begin{bmatrix} A \\ \sqrt{\lambda}L \end{bmatrix}$. Hierdoor kunnen we de normaal vergelijking herschrijven als:

$$\begin{aligned}
x^\lambda &= \left(\begin{bmatrix} A \\ \sqrt{\lambda}L \end{bmatrix}^T \begin{bmatrix} A \\ \sqrt{\lambda}L \end{bmatrix}\right)^{-1} \begin{bmatrix} A \\ \sqrt{\lambda}L \end{bmatrix}^T \begin{bmatrix} b \\ 0 \end{bmatrix} \\
&= (A^T A + \lambda L^T L)^{-1} A^T b. \qquad (7)
\end{aligned}$$

Deze vergelijking is computationeel gezien niet fijn aangezien we een inverse moeten berekenen. Om deze reden bekijken we in de komende hoofdstukken een andere methode om (6) op te lossen.

3 Standaard Tikhonov regularisatie

We zijn op zoek naar een oplossing voor (6) op een efficiënte manier voor algemene L gebaseerd op samengevoegde bidiagonalisatie. Om deze situatie beter te begrijpen bekijken we eerst de situatie waarbij $L = I$. Voor dit hoofdstuk volgen we grotendeels de redenatie van [3] waarbij de nadruk ligt op hoofdstuk 6. In dit hoofdstuk wordt ervan uitgegaan dat A lineair onafhankelijke kolommen heeft.

De situatie met $L = I$ wordt standaard Tikhonov regularisatie genoemd.
Definitie 3.1 (Standaard Tikhonov regularisatie)**.** De standaard vorm van Tikhonov regularisatie is gelijk aan de algemene vorm Tikhonov regularisatie met $L = I$ waarbij I de identiteit matrix is. Dit geeft:

$$x^\lambda = \underset{x}{\operatorname{argmin}} \{ \ ||Ax - b||^2 + \lambda ||x||^2 \ \}.$$

De vergelijking kan herschreven worden als:

$$x^\lambda = \underset{x}{\operatorname{argmin}} \left|\left| \begin{bmatrix} A \\ \sqrt{\lambda} I \end{bmatrix} x - \begin{bmatrix} b \\ 0 \end{bmatrix} \right|\right|_2 . \tag{8}$$

en er geldt

$$x^\lambda = (A^T A + \lambda I)^{-1} A^T b.$$

3.1 SVD analyse

Om verder inzicht in het probleem te krijgen bekijken we de SVD van A. We stoppen deze in (8). Dit geeft:

$$\begin{aligned} x^\lambda = (A^T A + \lambda I)^{-1} A^T b &= (V \Sigma^T U^T U \Sigma V^T + \lambda I)^{-1} (V \Sigma^T U^T) b \\ &= (V \Sigma^T U^T U \Sigma V^T + \lambda V^T V)^{-1} (V \Sigma^T U^T) \\ &= (V \Sigma^T \Sigma V^T + \lambda V^T V)^{-1} (V \Sigma^T U^T) \\ &= V (\Sigma^T \Sigma + \lambda I) V^T V \Sigma^T U^T b \\ &= V (\Sigma^T \Sigma + \lambda I) \Sigma^T U^T b \end{aligned}$$

We kunnen dit omschrijven naar somnotatie. Dit geeft

$$\begin{aligned} x^\lambda &= \sum_{i=1}^{n} \frac{\sigma_i}{\sigma_i^2 + \lambda} v_i u_i^T b \\ &= \sum_{i=1}^{n} \frac{\sigma_i^2}{\sigma_i^2 + \lambda} v_i \frac{u_i^T b}{\sigma_i} \end{aligned}$$

We zien dat wanneer $\sigma_i^2 \gg \lambda$ dat $\frac{\sigma_i^2}{\sigma_i^2+\lambda} \approx 1$ en dat wanneer $\sigma_i^2 \ll \lambda$ dat $\frac{\sigma_i^2}{\sigma_i^2+\lambda} \approx \frac{\sigma_i^2}{\lambda}$. We vergelijken dit met de SVD van het niet geregulariseerde probleem. We zien dat voor de singuliere waarden met $\sigma_i^2 \gg \lambda$ de Tikhonov componenten enorm op de componenten van de pseudoinverse componenten lijken. Wanneer $\sigma_i^2 \ll \lambda$ zien we echter dat $\frac{\sigma_i^2}{\sigma_i^2+\lambda}$ heel klein wordt. We zouden kunnen zeggen dat deze singuliere waarden en singuliere vectoren weinig bijdragen aan de SVD

van de Tikhonov benadering. Ze hebben veel minder invloed op de oplossing dan de grote singuliere waarden. Een andere opmerking is dat wanneer $\lambda = 0$ we de vergelijking krijgen behorende bij het oorspronkelijke probleem.

3.2 Projectie methode

Zoals eerder gezegd is Tikhonov regularisatie een goede methode maar kan het computationeel lastig zijn voor grote A. Hierdoor ontstaat het idee om het probleem te projecteren naar een lagere dimensie zodat het wel op te lossen is. We noemen de methode om dit te doen de projectie methode. We bekijken het probleem eerst zonder Tikhonov regularisatie. In deze paragraaf definiëren we bovendien de Krylov deelruimten.

We bekijken het probleem (2) nu zonder Tikhonov regularisatie. Een geprojecteerd probleem ziet er als volgt uit:

$$x_k = \underset{x}{\operatorname{argmin}} \{||Ax - b||_2^2\} \quad \text{zodat } x_k \in \mathcal{W}_k = \operatorname{span}\{w_1, \ldots, w_k\}. \tag{9}$$

Hierbij geldt dat de vectoren $w_1, \ldots, w_k$ een geschikte basis vormen voor de oplossingsruimte van (2). Verder stellen we dat de matrix $W_k \in \mathbb{R}^{n \times k}$ de matrix gegeven is door $W_k = (w_1, \ldots w_k)$.

We kunnen (9) herschrijven. De eis dat $x \in \mathcal{W}_k = \operatorname{span}\{w_1, \ldots, w_k\}$ is namelijk hetzelfde als $x_k = W_k y_k$ waarbij $y \in \mathbb{R}^k$. Wanneer dit gebruikt wordt, volgt het volgende probleem:

$$x_k = W_k y_k \quad y_k = \operatorname{argmin} \{||AW_k y - b||_2^2\}.$$

Het is belangrijk om op te merken dat AW_k een kleinere matrix is dan A. Hierdoor zal y_k makkelijker te berekenen zijn en ook het terugrekenen van y_k naar x_k kost niet veel werk als k klein.

Een andere belangrijke opmerking is dat de gevonden x nu een benadering is van x_{exact}. De vraag is nu hoe we een goede basis bepalen zodat deze benadering een goede benadering is. De ideale basis is gebaseerd op de SVD van A. Uit de voorgaande SVD analyse volgt dat de oplossing x^λ in de ruimte ligt opgespannen door de belangrijkste SVD componenten. Dit is een ruimte van lagere dimensie dan het oorspronkelijke probleem. Deze hebben bijna alle belangrijke informatie over het probleem in zich. Het probleem is echter het berekenen van de SVD wederom kostbaar is voor grote A. Een andere mogelijkheid is om een vaste basis te kiezen voor alle problemen. Hier is het probleem dat specifieke informatie over het probleem niet meegenomen wordt. Het voordeel is wel dat de basis niet elke keer opnieuw berekend hoeft te worden.

Een tussenweg vinden we met behulp van de zogenaamde Krylov deelruimten. Deze deelruimten bekijken we omdat het projectie probleem op deze manier fijne eigenschappen krijgt. De Krylov deelruimte wordt als volgt gedefinieerd:
Definitie 3.2 (Krylov Deelruimten)**.** [4, p.472] Zij $A \in \mathbb{R}^{n \times n}$ matrix en $b \in \mathbb{R}^n$. Dan wordt de Krylov deelruimte en Krylov matrix gegenereerd door A en b gegeven door:

$$\mathcal{K}_k(A, b) = \operatorname{span}\{b, Ab, A^2, \ldots, A^{k-1}b\}.$$

Definitie 3.3 (Krylov matrix)**.** [4, p.416] Zij $A \in \mathbb{R}^{n \times n}$ matrix en $b \in \mathbb{R}^n$. Dan wordt de Krylov matrix opgespannen door b en A gegeven door:

$$K_k(A, b) = [b, Ab, A^2, \ldots, A^{k-1}b].$$

Om te begrijpen waarom we naar de Krylov ruimten kijken maken we een uitstap naar tridiagonalisatie.

Stelling 3.1. *[4, p.414] Zij $A \in \mathbb{R}^{n\times n}$ een symmetrische matrix. Dan is het mogelijk om een orthogonale matrix $Q \in \mathbb{R}^{n\times n}$ te vinden zo dat*

$$Q^T AQ = T.$$

Hierbij is T een tridiagonale matrix. Dit houdt in dat de matrix de volgende vorm heeft:

$$T = \begin{bmatrix} \alpha_1 & \beta_1 & & & \\ \beta_1 & \alpha_2 & \beta_2 & & \\ & \ddots & \ddots & \ddots & \\ & & \beta_{n-2} & \alpha_{n-1} & \beta_{n-1} \\ & & & \beta_{n-1} & \alpha_n \end{bmatrix}.$$

Stelling 3.2. *[4, stelling 8.3.1] Zij $Q^T AQ = T$ de tridiagonale ontbinding van een symmetrische matrix $A \in \mathbb{R}^{n\times n}$, dan geldt dat $Q^T K(A, Q(;, 1), n) = R$ een bovendriekhoeks matrix is.*[...]

Stelling 3.2 geeft ons dat de Q uit de tridiagonalisatie van A ook gebruikt kan worden om de QR ontbinding van de Krylov matrix $K(A, Q(;, 1), n)$ te vinden. Aangezien voor de kolommen van Q ook geldt dat ze de oorspronkelijke matrix opspannen hebben we dus dat $K(A, Q(;, 1), n)$ opgespannen wordt door de kolommen van Q. Een belangrijke eigenschap van deze ontbinding is dat de grootste eigenwaarden van T een goede benadering zijn voor de grootste eigenwaarden van A.

We willen dit nu toepassen op ons kleinstekwadraten probleem. Een probleem is dat A in de bovenstaande stellingen een symmetrische matrix moet zijn. Zoiets is niet te zeggen over de matrix A uit ons probleem. Echter geldt er dat $A^T A$ wel symmetrisch is. Bovendien geldt, omdat A volledige kolomrang heeft, dat de singuliere waarden van A gelijk zijn aan de positieve wortel van $A^T A$. We weten dat de normaal vergelijking van (2) gelijk is aan

$$A^T Ax = A^T b.$$

Hierdoor ontstaat het idee om de Krylov deelruimte $\mathcal{K}_k(A^T A, A^T b)$ gebruiken als $\mathcal{W}_k$. We merken op dat we nu dus kijken naar de tridiagonalisatie

$$Q^T A^T AQ = T$$

We merken op dat deze ruimte gedefinieerd wordt door de vectoren die komen uit de verschillende stappen van de *power methode*. Deze methode benadert de grootste eigenwaarde en eigenvector van $A^T A$. Het gevolg hiervan is dat $(A^T A)^i A^T b$ steeds meer op elkaar gaan lijken. Hierdoor is deze basis die de Krylov deelruimte opspant geen goede basis om mee te werken. Om dit op te lossen zoeken we een orthonormale basis voor de ruimte. Het blijkt dat het Lanczos bidiagonalisatie algoritme uitgevoerd op A en b hiervoor zorgt.

3.3 Lanczos bidiagonalisatie

In deze paragraaf bekijken we wat Lanczos bidiagonalisatie precies inhoudt. Bovendien leggen we een link met de vorige paragraaf. We gebruiken hiervoor de redenatie uit [4, p.497-498].

Algorithm 1: Beneden Lanczos bidiagonalisatie

$\beta_1 = ||b||$
$\beta_1 u_1 = b$
$v_0 = 0$
voor k=1,2,...
  $\alpha_k v_k = A^T u_k - \beta_k v_{k-1}$
  $\beta_{k+1} u_{k+1} = A v_k - \alpha_k u_k$
einde

Voor nu zijn we geïnteresseerd in bidiagonalisatie van de volgende vorm:

$$U^T A V = B = \left[\begin{array}{ccccc} \alpha_1 & & & & \\ \beta_2 & \alpha_2 & & & \\ & \beta_3 & \ddots & & \\ & & \ddots & \alpha_n & \\ & & & \beta_{n+1} & \\ \hline & & 0 & & \end{array}\right].$$

Hierbij zijn $U \in \mathbb{R}^{m\times m}$ en $V \in \mathbb{R}^{n\times n}$ orthogonale matrices. De matrix B heeft $m-n$ nulrijen. We zullen deze ontbinding vanaf nu beneden bidiagonalisatie noemen.

Voor het berekenen van deze bidiagonalisatie leiden we het Lanczos bidiagonalisatie af. Hiervoor bekijken we $AV = UB$ door U en V in kolommen uit te schrijven:

$$A[v_1, v_2, \ldots, v_n] = [u_1, u_2, \ldots, u_m] \left[\begin{array}{ccccc} \alpha_1 & & & & \\ \beta_2 & \alpha_2 & & & \\ & \beta_3 & \ddots & & \\ & & \ddots & \alpha_n & \\ & & & \beta_{n+1} & \\ \hline & & 0 & & \end{array}\right].$$

We merken op dat de $i$de kolom van de matrix aan de linkerkant van het is teken gelijk is aan Av_i. De $i$de kolom van de matrix aan rechterkant aan $\alpha_i u_i + \beta_{i+1} u_{i+1}$. Dit is in te zien door te kijken naar de eerste kolom. Er geldt dat

$$[u_1, u_2, \ldots, u_m] \begin{bmatrix} \alpha_1 \\ \beta_2 \\ 0 \\ \vdots \\ 0 \end{bmatrix} = \begin{bmatrix} u_{1,1}\alpha_1 + u_{2,1}\beta_2 \\ u_{1,2}\alpha_1 + u_{2,2}\beta_2 \\ \vdots \\ u_{1,n}\alpha_1 + u_{2,n}\beta_2 \end{bmatrix} = \alpha_1 u_1 + \beta_2 u_2.$$

Hieruit volgt dat $Av_i = \alpha_i u_i + \beta_{i+1} u_{i+1}$. Op soortgelijke manier kan er naar $A^T U = V B^T$ gekeken worden. We verkrijgen dan de vergelijking $A^T u_i = \beta_i v_{i-1} + \alpha_i v_i$ waarbij $\beta_1 v_0 = 0$. Deze vergelijkingen vormen de basis voor het beneden Lanczos bidiagonalisatie algoritme zoals gegeven in algoritme 1. De parameters β_k en α_k worden zo gekozen dat de bijbehorende vectoren v_k en u_k genormaliseerd zijn. Wanneer α_i en β_{i+1} niet nul zijn voor $i = 1, \ldots, k$ is het algoritme goed gedefinieerd voor de eerste k stappen.

We definiëren nu een aantal matrices. Laat $V_k = (v_1, v_2, \ldots, v_k)$, $U_{k+1} = (u_1, u_2, \ldots, u_{k+1})$ en $B_k = B(1 : k+1, 1 : k)$. Er gelden nu de volgende gelijkheden:

$$AV_k = U_{k+1}B_k \tag{10}$$

$$A^T U_{k+1} = V_k B_k^T + \alpha_{k+1} v_{k+1} e_{k+1}^T. \tag{11}$$

Hierbij is e_{k+1} de $k+1$'de eenheidsvector van lengte $k+1$.

De eerste gelijkheid volgt direct uit de afleiding van het algoritme. Voor de tweede gelijkheid is meer werk nodig. We bekijken eerst $A^T U_{k+1}$ door het te vergelijken met het algoritme:

$$A^T[u_1, u_2, \ldots, u_{k+1}] = \begin{bmatrix} \alpha_1 v_1 \\ \alpha_2 v_2 + \beta_2 v_1 \\ \vdots \\ \alpha_k v_{k-1} + \beta_k v_k \\ \alpha_{k+1} v_{k+1} + \beta_{k+1} v_k \end{bmatrix}.$$

Nu bekijken we $V_k B_k^T$. Er geldt dat

$$[v_1, v_2, \ldots, v_k] \begin{bmatrix} \alpha_1 & \beta_2 & & & \\ & \alpha_2 & \ddots & & \\ & & \ddots & \beta_k & \\ & & & \alpha_k & \beta_{k+1} \end{bmatrix} = \begin{bmatrix} \alpha_1 v_1 \\ \beta_2 v_1 + \alpha_2 v_2 \\ \vdots \\ \alpha_k v_{k-1} + \beta_k v_k \\ \beta_{k+1} v_k \end{bmatrix}.$$

Wanneer we beide matrices vergelijken zien we dat ze bijna gelijk zijn. Alleen de laatste rij is anders. Om ze gelijk aan elkaar te laten zijn, moet er bij de laatste rij van $V_k B_k^T$ nog $\alpha_{k+1} v_{k+1}$ opgeteld worden. Dit resulteert in $A^T u_{k+1} = V_k B_k^T + \alpha_{k+1} v_{k+1} e_{k+1}^T$.

De kolommen van U_{k+1} en V_k moeten wegens constructie orthonormaal zijn. Dit is echter alleen theoretisch het geval. Wanneer algoritme 1 rechtstreeks wordt geïmplementeerd zullen de kolommen van U_k en V_k niet orthogonaal zijn. Om dit te verhelpen moet er een extra orthogonalisatie stap uitgevoerd worden. Dit kost extra rekenkracht.

We merken op dat met behulp van (10) de volgende gelijkheid op:

$$V_k^T A^T A V_k = B_k^T U_{k+1}^T U_{k+1} B_k = B_k^T B_k.$$

Het volgt door uitschrijven van $B_k^T B_k$ dat deze matrix een symmetrische tridiagonale matrix is. Verder geldt er dat v_1 de genormaliseerde vector is zo dat $v_1 = A^T b$ geldt. Hierdoor volgt met de redenatie uit de voorgaande paragraaf dat $\{v_1, v_2 \ldots, v_k\}$ de Krylov ruimte $\mathcal{K}_k(A^T A, A^T b)$ opspant.

3.4 Projectiemethode met bidiagonalisatie

We kijken nu weer naar het projectie probleem. We passen het in de vorige paragraaf gedefinieerde bidiagonalisatie toe op het projectie probleem en we kijken hoe het probleem er vervolgens uit komt te zien. Bovendien noemen we kort het LSQR algoritme

We nemen $W_k = V_k$. Dit doen we omdat de kolommen vna V_k een basis vormen voor $\mathcal{K}_k(A^TA, A^Tb)$. Dit geeft:

$$x_k = V_k y_k \quad y_k = \text{argmin}_y \ \{||AV_k y - b||_2\}.$$

We merken met behulp van (10) het volgende op:

$$\begin{aligned} ||AV_k y - b||_2 &= ||U_{k+1}B_k V_k^T V_k y - b||_2 \\ &= ||U_{k+1}B_k y - b||_2 \\ &= ||U_{k+1}(B_k y - \beta_1 e_1)||_2 \\ &= ||B_k y - \beta_1 e_1||_2 \end{aligned}$$

Hierbij is e_1 de eerste eenheidsvector van lengte $k+1$. De tweede gelijkheid geldt hierbij door de orthonormale kolommen van V_k en de derde gelijkheid door de relatie $U_{k+1}\beta_1 e_1 = \beta_1 u_1 = b$. De laatste gelijkheid geldt vanwege de orthonormale kolommen van U_{k+1}.

Het projectie probleem ziet er nu als volgt uit:

$$x_k = V_k y_k, \quad y_k = \text{argmin}_y \ \{||(B_k y - \beta_1 e_1)||_2\}.$$

We kunnen dit gebruiken om (2) op te lossen. Om dit te doen bereken je de k'de stap van het algoritme. Vervolgens bereken je de optimale λ en daarna y_k. Wanneer er convergentie van λ plaats vindt is dit een teken om te stoppen met het algoritme. Op dit punt kan x_k berekend worden. Wanneer dit niet het geval is ga je verder met $k+1$. We merken op dat deze methode heel erg vaag is. We kunnen ons immers afvragen wat "als λ niet veel meer verandert" precies inhoud. Het vinden van de juiste k is een probleem wat nog niet goed beantwoord is. Wanneer k te klein is, zal de methode te weinig informatie bevatten over het probleem en zal de oplossing niet goed zijn. Wanneer k te groot is zal er te veel ruis veroorzaakt worden door de kleine singuliere waarden van A. Het is dus belangrijk de goede te kiezen. Hieruit volgt dat deze methode ook een regularisatie methode is met regularisatie parameter k.

De bovenstaande afleidingen zijn de basis voor een iteratieve oplossingsmethode genaamd LSQR. Het algoritme wordt beschreven in [8]. We gaan niet verder in op de werking van het algoritme maar maken wel gebruik ervan tijdens de implementatie. Er geldt dat LSQR een iteratieve methode is.

Een belangrijk resultaat uit [8] is dat een oplossing van het minimalisatie probleem (2), verkregen door het LSQR algoritme in de ruimte $\mathcal{K}_k(A^TA, A^Tb)$ ligt.

3.5 Hybride methode

In deze paragraaf passen we de projectie methode toe op het Tikhonov probleem. We krijgen dan een hybride methode die gebruik maakt van twee regularisatie methodes. We zullen een aantal belangrijke resultaten geven.

Wanneer we de projectie methode toepassen op de Tikhonov vergelijking geeft dit het volgende probleem:

$$x_k^\lambda = \underset{x}{\text{argmin}} \quad \left|\left|\begin{bmatrix} A \\ \sqrt{\lambda} I \end{bmatrix} x - \begin{bmatrix} b \\ 0 \end{bmatrix}\right|\right|_2 \quad \text{zodat} \quad x_k^\lambda \in \mathcal{K}_k(A^TA, A^Tb).$$

Dit kunnen we herschrijven tot:

$$x_k^\lambda = V_k y_k^\lambda, \quad y_k^\lambda = \underset{y}{\operatorname{argmin}} \left|\left| \begin{bmatrix} A \\ \sqrt{\lambda} I \end{bmatrix} V_k y - \begin{bmatrix} b \\ 0 \end{bmatrix} \right|\right|_2$$
$$= \underset{y}{\operatorname{argmin}} \, ||AV_k y - b||_2^2 + \lambda ||V_k y||_2^2.$$

Aangezien V_k orthonormale kolommen heeft geldt dat $||V_k y||_2^2 = ||y||_2^2$. Samen met behulp van de analyse uit de voorgaande paragraaf, zien we dat het probleem te herschrijven is als volgt:

$$x_k^\lambda = V_k y_k^\lambda, \quad y_k^\lambda = \underset{y}{\operatorname{argmin}} \, \{||(B_k y - \beta_1 e_1)||_2^2 + \lambda ||y||_2^2\}.$$

De volgende stelling is van belang voor het vinden van de juiste λ.
Stelling 3.3. *[3, p.129] Gegeven de definities van de matrices zoals uit de algoritmes, dan gelden de volgende gelijkheden:*

$$||Ax_k^\lambda - b||_2 = ||B_k y_k^\lambda - \beta_1 e_1||_2$$
$$||x_k^\lambda||_2 = ||y_k^\lambda||_2$$

Bewijs: Het bewijs van de bovenstaande stelling volgt direct uit het invullen $V_k y_k^\lambda = x_k^\lambda$ en de afleiding van de vorige paragraaf. De tweede gelijkheid volgt door de orthonormale kolommen van V_k. □

Het gevolg van deze stelling is dat we de normen uit het niet geprojecteerde probleem direct kunnen berekenen door die van het geprojecteerde probleem. Er geldt dat hiervoor x_k^λ niet berekend hoeft te worden. Het gevolg hiervan is weer dat we een parameter keuze methode kunnen toepassen op y_k^λ.

We bekijken nog een keer de oplossing waarin x_k^λ zit. We hebben in de voorgaande paragrafen gezien dat wanneer we een probleem van vorm (2) oplossen met ons algoritme dat de oplossing in een Krylov deelruimte zit. In dit geval gaat het om de Krylov deelruimte

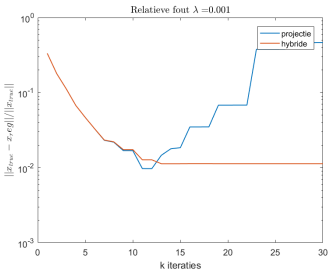
$$\mathcal{K}_k \left(\begin{bmatrix} A \\ \sqrt{\lambda} I \end{bmatrix}^T \begin{bmatrix} A \\ \sqrt{\lambda} I \end{bmatrix}, \begin{bmatrix} A \\ \sqrt{\lambda} I \end{bmatrix}^T \begin{bmatrix} b \\ 0 \end{bmatrix} \right) = \left\{ \left(\begin{bmatrix} A \\ \sqrt{\lambda} I \end{bmatrix}^T \begin{bmatrix} A \\ \sqrt{\lambda} I \end{bmatrix} \right)^i \begin{bmatrix} A \\ \sqrt{\lambda} I \end{bmatrix}^T \begin{bmatrix} b \\ 0 \end{bmatrix} \right\}_{i=0}^{k-1}$$
$$= \{(A^T A + \lambda I)^i A^T b\}_{i=0}^{k-1}.$$

Omdat λI een constante is volgt dat deze Krylov deelruimte gelijk is aan de $\mathcal{K}_k(A^T A, A^T b)$.

Het toepassen van het algoritme gebeurd op dezelfde manier als bij het probleem zonder regularisatie. We zitten hier met hetzelfde probleem wat betreft k. We moeten weten voor welke waarde van k we het algoritme moeten uitvoeren. We kunnen geen beter antwoord geven dan te kijken naar wanneer λ convergeert. De matrices die opgeslagen moeten worden zijn V_k en B_k. Vervolgens voer je een parameter keuze methode uit op het probleem met y_k. Wanneer de gevonden λ niet meer veel verandert kun stoppen met deze stap en kun je de laatste y_k omrekenen naar x_k.

De vraag ontstaat waarom er zowel projectie als Tikhonov regularisatie wordt toegepast. Waarom passen we niet alleen de projectie methode toe. Het antwoord heeft te maken met vinden van de juiste k. Omdat dit niet goed mogelijk is om dit precies te doen kan het zijn dat de oplossing

x_k niet de beste reconstructie is. Bovendien geldt dat de Krylov deelruimte afhangt van b. Zoals eerder besproken heeft b fouten. We baseren dus onze projectie methode op een vector met ruis. Hierdoor ontstaan er basisvectoren die niet nuttig zijn voor het probleem. Om het wat meer te stabiliseren gebruiken we daarom ook nog de Tikhonov regularisatie methode. Het blijkt namelijk dat ondanks dat het vinden van de juiste k hierbij ook een probleem kan zijn, het hier net iets minder van belang is. Dit komt omdat wanneer Tikhonov regularisatie toegepast wordt, λ de rol van regularisatie parameter overneemt van k. Dit zien we in figuur 6.



Figuur 6: Het verschil tussen de projectie methode en de projectie methode met Tikhonov regularisatie bij het *Gravity* probleem. Er is gebruik gemaakt van $\lambda = 0.001$

De oplossing in figuur 2 in de inleiding is berekend met behulp van deze methode. Hierbij hebben we $k = 15$ en $\lambda = 10^{-3}$ genomen. We zien dat dan na 15 iteraties de oplossing al heel dicht in de beurt ligt van de originele oplossing.

4 Algemene Tikhonov regularisatie

We hebben nu een werkend algoritme voor het standaard probleem. In dit hoofdstuk bekijken we het algemene probleem. We volgen hierbij grotendeels de redenatie van [5]. We zullen eerst uitleggen waarom we een algoritme willen ontwikkelen. Daarna zullen we de GSVD uitleggen en vervolgens definiëren we ons samengevoegd bidiagonalisatie algoritme en passen we deze toe op het projectie probleem. Daarna maken we wat opmerkingen over efficiëntie en als laatste kijken we naar de ruimte waarin de oplossing ligt.

Voordat we verder gaan maken we een aantal opmerkingen. We definiëren als de matrix $[A\ L] = \begin{bmatrix} A \\ L \end{bmatrix}$. We nemen verder aan in dit hoofdstuk dat $N(A) \cap N(L) = 0$ zodat $[A\ L]$ lineair onafhankelijke kolommen heeft. Hierdoor is $[A\ L]^T[A\ L]$ inverteerbaar en heeft (6) een unieke oplossing.

We willen nu de algemene Tikhonov regularisatie vergelijking oplossen.

$$x^\lambda = \underset{x}{\operatorname{argmin}} \ \{ \ ||Ax - b||_2^2 + \lambda ||Lx||_2^2 \ \}$$

Wanneer we de strategie met het projectie probleem van hoofdstuk 3 hier direct op toepassen bereiken we niet de gewenste efficiëntie die bij de standaard vorm wel bereikt wordt. Om dit te zien bekijken we de Krylov deelruimte waarin de oplossing zit.

Deze is gelijk aan:

$$\begin{aligned} \mathcal{K}_k &= \operatorname{span} \left\{ \left(\begin{bmatrix} A \\ \sqrt{\lambda} L \end{bmatrix}^T \begin{bmatrix} A \\ \sqrt{\lambda} L \end{bmatrix} \right)^i \begin{bmatrix} A \\ \sqrt{\lambda} L \end{bmatrix}^T \begin{bmatrix} b \\ 0 \end{bmatrix} \right\}_{i=0}^{k-1} \\ &= \operatorname{span} \left\{ (A^T A + \lambda L^T L)^i A^T b \right\}_{i=0}^{k-1} \end{aligned} \tag{12}$$

Wanneer $L = I$ geldt dat de opgespannen Krylov deelruimte onafhankelijk is van λ. Wanneer $L \neq I$ is dit niet meer het geval. Hierdoor moet voor elke λ die geprobeerd wordt een nieuwe basis gemaakt worden. Dit neemt een groot gedeelte van de efficiëntie weg. Om deze reden gaan we op zoek naar alternatieven.

Het is mogelijk om het algemene probleem om te schrijven naar het standaard probleem. Een van de mogelijkheden om dit te doen is door de transformatie $y = Lx$ en $\bar{A} = AL^{-1}$ toe te passen op het probleem. Dit kan alleen als L inverteerbaar is. Het probleem wordt dan:

$$y^\lambda = \underset{x}{\operatorname{argmin}} \ \{ \ ||\bar{A}y - b||_2^2 + \lambda ||y||_2^2 \ \}, \quad x^\lambda = L^{-1} y^\lambda$$

We kunnen nu het probleem behandelen als een standaard Tikhonov probleem. Deze transformatie is alleen mogelijk als L inverteerbaar is en de methode is alleen nuttig als de inverse makkelijk uitgerekend kan worden. Dit is in de praktijk vaak niet het geval.

Wanneer L niet inverteerbaar is kan er ook gebruik gemaakt worden van de zogenaamde A-weighted pseudoinverse van L [9]. Deze wordt gegeven door $L_A^\dagger = (I - (A(I - L^\dagger L))^\dagger A) L^\dagger$. De algemene vorm heeft nu de vorm:

$$y^\lambda = \underset{x}{\operatorname{argmin}} \ \{ \ ||AL_A^\dagger y - b||_2^2 + \lambda ||y||_2^2 \ \}, \quad x^\lambda = L_A^\dagger y^\lambda + x_0.$$

Hierbij is x_0 de component van de oplossing in de nulruimte is. Wederom kan dit probleem aangepakt worden zoals in hfst 3. Dit is nuttig wanneer $L_A^\dagger$ en de andere operaties makkelijk te berekenen zijn. Wederom is dit niet altijd het geval.

We bekijken een algoritme wat het idee van de bidiagonalisatie gebruikt. Dit zal leiden tot een algoritme dat gebruik maakt van samengevoegde bidiagonalisatie van A en L.

4.1 GSVD

Om inzicht in dit probleem te krijgen bekijken we de *generalized singular value decomposition* (GSVD) van $\{A, L\}$. Dit doen we omdat de SVD van A analyse hier minder nut heeft aangezien we ook nog een matrix L hebben. Bovendien willen we een ontbinden hebben waarbij de rechter singuliere vectoren hetzelfde zijn voor A en L. [3] We definiëren in deze paragraaf de GSVD door middel van de QR ontbinding en de CS ontbinding.

We beginnen met het bekijken van de gereduceerde QR ontbinding van $[A\ L]$ zoals die van definitie 1.3.

Deze wordt gegeven door:

$$\begin{bmatrix} A \\ L \end{bmatrix} = QR$$

waarbij $Q \in \mathbb{R}^{(m+p)\times n}$ een matrix met orthonormale kolommen en $R \in \mathbb{R}^{n\times n}$ een bovendriehoekse matrix is.

We geven QR ontbinding nu op een andere manier weer. We kiezen $Q_A \in \mathbb{R}^{m\times n}$ en $Q_L \in \mathbb{R}^{p\times n}$ zo dat ze een partitie vormen van Q. De QR ontbinding ziet er nu als volgt uit:

$$\begin{bmatrix} A \\ L \end{bmatrix} = QR = \begin{bmatrix} Q_A \\ Q_L \end{bmatrix} R.$$

Dit geeft $A = Q_A R$ en $L = Q_L R$. We leiden nu een gelijkheid af die geldt vanwege het feit dat Q een matrix is met orthonormale kolommen. Dit betekent dat $Q^T Q = I$. Bovendien geldt dat

$$Q^T Q = \begin{bmatrix} Q_A \\ Q_L \end{bmatrix}^T \begin{bmatrix} Q_A \\ Q_L \end{bmatrix} = Q_A^T Q_A + Q_L^T Q_L.$$

Er geldt dus dat $Q_A^T Q_A + Q_L^T Q_L = I$.

Voor we verder gaan bekijken we de CS ontbinding. Deze gebruiken we om een gegeneraliseerde versie van de SVD te vinden.

Stelling 4.1 (CS ontbinding)**.** *[10, stelling 1.1] Laat $Q \in \mathbb{R}^{(m+p)\times n}$ orthonormale kolommen hebben. Hierbij vormen $Q_A \in \mathbb{R}^{m\times n}$ en $Q_L \in \mathbb{R}^{p\times n}$ een partitie van Q zodanig dat*

$$Q = \begin{bmatrix} Q_A \\ Q_L \end{bmatrix}$$

Dan geldt dat er orthonormale matrices $P_A \in \mathbb{R}^{m\times m}$, $P_L \in \mathbb{R}^{p\times p}$ en $W \in \mathbb{R}^{n\times n}$ zo dat:

$$\begin{bmatrix} P_A^T & 0 \\ 0 & P_L^T \end{bmatrix} \begin{bmatrix} Q_A \\ Q_L \end{bmatrix} V = \begin{bmatrix} P_A^T Q_A W \\ P_L^T Q_L W \end{bmatrix} \tag{13}$$

een van de volgende vormen aanneemt:

- *als $p \geq n$ en $m \geq n$ dan heeft (13) de volgende vorm:*

$$\begin{array}{c} \\ n \\ m-n \\ n \\ p-n \end{array} \begin{array}{c} n \\ \left[\begin{array}{c} C \\ 0 \\ \hline S \\ 0 \end{array}\right] \end{array}$$

- *als $m \geq n$ en $p \leq n$ dan heeft (13) de volgende vorm:*

$$\begin{array}{c} \\ p \\ n-p \\ m-n \\ p \end{array} \begin{array}{c} \begin{array}{cc} p & n-p \end{array} \\ \left[\begin{array}{cc} C & 0 \\ 0 & I \\ 0 & 0 \\ \hline S & 0 \end{array}\right] \end{array}$$

- *als $m \leq n$ en $p \geq n$ dan heeft (13) de volgende vorm:*

$$\begin{array}{c} \\ p \\ p \\ n-p \\ m-n \end{array} \begin{array}{c} \begin{array}{cc} p & n-p \end{array} \\ \left[\begin{array}{cc} C & 0 \\ \hline S & 0 \\ 0 & I \\ O & 0 \end{array}\right] \end{array}$$

- *als $m \leq n$ en $p \leq n$ dan heeft (13) de volgende vorm:*

$$\begin{array}{c} \\ m+p-n \\ n-p \\ m+p-n \\ n-m \end{array} \begin{array}{c} \begin{array}{ccc} m+p-n & n-p & n-m \end{array} \\ \left[\begin{array}{ccc} C & 0 & 0 \\ 0 & I & 0 \\ \hline S & 0 & 0 \\ 0 & 0 & I \end{array}\right] \end{array}$$

Hierbij zijn C en S vierkante diagonaal matrices met niet negatieve elementen. De elementen op de diagonaal liggen tussen de 0 en de 1. Bovendien geldt dat $C^2 + S^2 = I$. Deze ontbinding wordt de CS ontbinding genoemd wat verwijst naar cosinus sinus ontbinding.

Het bewijs voor een gegeneraliseerde versie van deze stelling wordt gegeven in [11]. We maken alleen gebruik van de stelling en geven niet het bewijs. Voor onze situatie zijn de eerste 2 gevallen het belangrijkst aangezien we ervan uitgaan dan $m \geq n$.

Voor het verdere gebruik noemen we de deelmatrix boven de streep C_A en de matrix onder de streep S_L. We merken op dat er nog steeds geldt dat $C_A^T C_A + S_L^T S_L = I$ maar dat het niet meer mogelijk is om het kwadraat te nemen omdat de matrices niet meer vierkant zijn. Verder geldt dat $C_A \in \mathbb{R}^{m \times n}$ en $S_A \in \mathbb{R}^{p \times n}$. Uit stelling 4.1 volgt nu dat $P_A^T Q_A W = C_A$ en $P_L Q_L W = S_A$.

Met behulp van deze CS ontbinding kunnen we de volgende stelling bewijzen.
Stelling 4.2. *[5, p.317] Zij P_A, C_A, P_L, S_L en W de matrices verkregen uit de CS ontbinding. Dan geldt de volgende ontbinding:*

$$A = P_A C_A G^{-1}, \quad L = P_L S_L G^{-1} \quad \textit{waarbij} \quad G = R^{-1} W \tag{14}$$

Bewijs: De twee te bewijzen gelijkheden volgen vrij snel uit de QR ontbinding en de CS ontbinding. Zo geldt: $A = Q_A R = P_A C_A W^T R = P_A C_A G^{-1}$. De vergelijking voor L geldt op soortgelijke manier. $\square$

De ontbinding in (14) wordt de gegeneraliseerde singuliere waarde ontbinding van het matrix paar $\{A, L\}$ genoemd. We refereren naar deze ontbinding als de GSVD waarbij de afkorting komt van de Engelse term *Generalized Singular Value Decomposition.* De elementen op de diagonalen van C_A en S_L zijn hierbij op de volgende manier geordend:

$$1 \geq c_1 \geq \ldots \geq c_{\min(n,p)} \geq 0, \quad c_{\min(n,p)+1} = \ldots = c_n = 1$$
$$0 \leq s_1 \leq \ldots \leq s_{\min(n,p)} \leq 1.$$

De variabelen $\frac{c_i}{s_i}$ worden de gegeneraliseerde singuliere waarden genoemd. Wanneer $s_i = 0$ dan stellen we dat de gegeneraliseerde singuliere waarde gelijk is aan oneindig.

Op deze manier geldt wanneer $n \geq p$ dat $Q_A = P_A C_A W^T$ de SVD is van Q_A. Er geldt immers dat P_A en W beide orthonormale matrices zijn en C_A heeft alleen niet negatieve elementen die in van groot naar klein zijn geordend. Wanneer $p > n$ kan de SVD van Q_A ook verkregen worden. Hiervoor is alleen een permutatie van de elementen nodig zodat ze op de juiste manier geordend zijn.

4.2 Tikhonov uitgedrukt in GSVD

We willen nu een GSVD analyse toepassen op de Tikhonov vergelijking (7). Wanneer dit op uitgewerkt wordt op dezelfde manier als bij de SVD analyse van het standaard probleem vinden we

$$\begin{aligned} x^\lambda &= G(C_A^T C_A + \lambda S_L^T S_L)^{-1} C_A^T P_A^T b \\ &= \sum_{i=1}^{\min(n,p)} \frac{c_i}{c_i^2 + \lambda s_i^2} p_{A,i}^T b g_i + \sum_{i=\min(n,p)+1}^{n} p_{A,i}^T b g_i \\ &= \sum_{i=1}^{\min(n,p)} \frac{c_i^2}{c_i^2 + \lambda s_i^2} \frac{p_{A,i}^T b}{c_i} g_i + \sum_{i=\min(n,p)+1}^{n} p_{A,i}^T b g_i. \end{aligned}$$

Hierbij zijn $p_{A,i}$ en g_i de kolommen van P_A en G. De tweede term ligt in de nulruimte van L.

Bij de gewone SVD hebben we gekeken wat er gebeurt als de singuliere groot of klein waren ten opzichte van λ. Zoiets willen we nu ook doen. Echter hebben we nu niet meer te maken met een singuliere waarde. We kunnen echter wel kijken naar de gegeneraliseerde singuliere waarden. We merken op dat de volgende gelijkheid geldt:

$$\frac{c_i^2}{c_i^2 + \lambda s_i^2} = \frac{\left(\frac{c_i}{s_i}\right)^2}{\left(\frac{c_i}{s_i}\right)^2 + \lambda}. \tag{15}$$

Wanneer $\frac{c_i}{s_i} \ll \sqrt{\lambda}$ dan volgt dat (15) ongeveer gelijk is aan 1. Wanneer $\frac{c_i}{s_i} \gg \sqrt{\lambda}$ volgt dat (15) ongeveer gelijk is aan $\frac{c_i^2}{\lambda}$. We zien wederom dat (15) een filter effect heeft. De kleine c_i worden geassocieerd met de componenten die ruis veroorzaken. We zien dat (15) het effect heeft het effect van deze kleine c_i weg te nemen. [3]

4.3 Samengevoegde bidiagonalisatie

We ontwikkelen nu een algoritme dat ons helpt bij het oplossen van (6). Dit is gebaseerd op het Lanczos bidiagonalisatie algoritme uit het voorgaande hoofdstuk. We maken zowel gebruik van het idee achter deze methode als het daadwerkelijke algoritme zelf. In plaats van het het algoritme uit te voeren op A, wordt het nu uitgevoerd op Q_A. Bovendien passen we ook boven bidiagonalisatie toe op Q_L. Samen levert dit een samengevoegd bidiagonalisatie algoritme op.

Algorithm 2: Beneden Lanczos Bidiagonalisatie

$\beta_1 u_1 = b$
$\alpha_1 v_1 = Q_A^T u_1$
voor $i = 1, 2, ...n-1$
  $\beta_{i+1} u_{i+1} = Q_A v_i - \alpha_i u_i$
  $\alpha_{i+1} v_{i+1} = Q_A^T u_{i+1} - \beta_{i+1} v_i$
einde

Het beneden bidiagonalisatie algoritme 2 is hetzelfde als het algoritme 1 uit hoofdstuk 3. We voeren het nu echter uit op Q_A. We merken op dat de weergave iets anders is. Het aantal stappen wat doorlopen moet worden is anders dan bij het oorspronkelijke algoritme. We kiezen voor deze weergave omdat het later ervoor zorgt dat bepaalde afleidingen makkelijker gaan.

Voor Q_L gebruiken we boven bidiagonalisatie. Boven bidiagonalisatie lijkt hier heel erg op op beneden bidiagonalisatie. We gebruiken de volgende boven bidiagonalisatie voor Q_L:

$$\hat{U}^T Q_L \hat{V} = \hat{B}.$$

De precisie vorm van $\hat{B}$ is afhankelijk van n en p. Wanneer $n \geq p$ dan geldt dat

$$\hat{B} = \left[\begin{array}{ccccc|c} \hat{\alpha_1} & \hat{\beta_1} & & & & \\ & \hat{\alpha_2} & \hat{\beta_2} & & & \\ & & \ddots & \ddots & & 0 \\ & & & \ddots & \hat{\beta}_{p-1} & \\ & & & & \hat{\alpha_p} & \end{array}\right]$$

waarbij het aantal kolommen dat op het einde bestaat uit nullen gelijk is aan $n-p$ Wanneer $n \leq p$ dan geldt dat

$$\hat{B} = \left[\begin{array}{ccccc} \hat{\alpha_1} & \hat{\beta_1} & & & \\ & \hat{\alpha_2} & \hat{\beta_2} & & \\ & & \ddots & \ddots & \\ & & & \ddots & \hat{\beta}_{n-1} \\ & & & & \hat{\alpha_n} \\ \hline & & 0 & & \end{array}\right]$$

waarbij het aantal rijen dat op het einde bestaat uit nullen gelijk is aan $p-n$.

Op soortgelijke manier als bij de afleiding van de beneden bidiagonalisatie vinden we hiervoor het boven Lanczos bidiagonalisatie algoritme. Deze wordt gegeven in algoritme 3.

Algorithm 3: Boven Lanczos Bidiagonalisatie

$\hat{v}_1 = v_1$
$\hat{\alpha}_1 \hat{u}_1 = Q_L \hat{v}_1$
voor $i = 1, 2, ... \min(p, n) - 1$
  $\hat{\beta}_i \hat{v}_{i+1} = Q_L^T \hat{u}_i - \hat{\alpha}_i \hat{v}_i$
  $\hat{\alpha}_{i+1} \hat{u}_{i+1} = Q_L \hat{v}_{i+1} - \hat{\beta}_i \hat{u}_i$
einde

We bekijken beide bidiagonalisaties tegelijkertijd. Samen vormen ze de samengevoegde bidiagonalisatie van $\{Q_A, Q_L\}$. Voor α_i, β_i, $\hat{\alpha}_i$ en $\hat{\beta}_i$ geldt dat ze zo worden gekozen dat de bijbehorende vector genormaliseerd wordt. Na k iteraties zijn er 6 matrices gevormd. De eerste twee zijn $V_k \in \mathbb{R}^{n \times k}$ en $U_{k+1} \in \mathbb{R}^{m \times (k+1)}$ waarbij

$$V_k = (v_1, v_2, \ldots, v_k)$$

$$U_{k+1} = (u_1, u_2, \ldots, u_{k+1}).$$

De volgende twee zijn $\hat{U}_k \in \mathbb{R}^{p \times k}$ en $\hat{V}_k \in \mathbb{R}^{n \times k}$ waarbij

$$\hat{V}_k = (\hat{v}_1, \hat{v}_2, \ldots, \hat{v}_k)$$

$$\hat{U}_k = (\hat{u}_1, \hat{u}_2, \ldots, \hat{u}_k).$$

Door de constructie van u_k, v_k, $\hat{u}_k$ en $\hat{v}_k$ zijn de kolommen van U_k, V_k, $\hat{U}_k$ en $\hat{V}_k$ orthonormaal. De laatste twee matrices zijn een beneden bidiagonaal matrix $B_k \in \mathbb{R}^{(k+1) \times k}$ en een boven bidiagonaal matrix waarbij $\hat{B}_k \in \mathbb{R}^{k \times k}$.

$$B_k = \begin{bmatrix} \alpha_1 & & & \\ \beta_2 & \alpha_2 & & \\ & \beta_3 & \ddots & \\ & & \ddots & \alpha_k \\ & & & \beta_{k+1} \end{bmatrix} \quad \hat{B}_k = \begin{bmatrix} \hat{\alpha}_1 & \hat{\beta}_1 & & & \\ & \hat{\alpha}_2 & \hat{\beta}_2 & & \\ & & \ddots & \ddots & \\ & & & \hat{\alpha}_{k-1} & \hat{\beta}_{k-1} \\ & & & & \hat{\alpha}_k \end{bmatrix}.$$

De volgende twee vergelijkingen zijn gelijk aan de vergelijkingen (10) en (11) alleen dan met Q_A in plaats van A

$$Q_A V_k = U_{k+1} B_k \tag{16}$$

$$Q_A^T U_{k+1} = V_k B_k^T + \alpha_{k+1} v_{k+1} e_{k+1}^T. \tag{17}$$

Voor de boven bidiagonalisatie van Q_L zijn er op een soortgelijke manier als eerder vergelijkingen af te leiden. Deze worden als volgt gegeven:

$$Q_L \hat{V}_k = \hat{U}_k \hat{B}_k \tag{18}$$

$$Q_L^T \hat{U}_k = \hat{V}_k \hat{B}_k + \hat{\beta}_k \hat{v}_{k+1} e_k^T. \tag{19}$$

De volgende stelling geeft een opstapje voor het verband tussen de twee verschillende bidiagonalisatie algoritmes.

Stelling 4.3. *[5, stelling 2.1] Gegeven de samengevoegde bidiagonalisatie van Q_A en Q_L zoals beschreven in algoritme 2 en 3. Dan gelden de volgende gelijkheden:*

$$\hat{v}_{j+1} = (-1)^j v_{j+1} \quad \text{en} \quad \hat{\alpha}_j \hat{\beta}_j = \alpha_{j+1}\beta_{j+1}$$

waarbij $j = 1, 2, \ldots, \min(p, n) - 1$.

Bewijs: We zullen deze stelling met behulp van inductie bewijzen.

Als eerste bewijzen we dat de stelling geld voor $j = 1$. We moeten in dit geval bewijzen dat $\tilde{v}_2 = -v_2$ en $\hat{\alpha}_1\hat{\beta}_1 = \alpha_2\beta_2$.

We merken op dat in algoritme 3 de initialisatie geeft dat $\tilde{v}_1 = v_1$ en nemen aan dat $\hat{\alpha}_1$, $\hat{\beta}_1$, α_2 en β_2 strikt positief zijn.

Voor het bewijs wordt gebruik gemaakt van de gelijkheden die uit de algoritmes 2 en 3 volgen. De eerste die we gebruiken is $\hat{\beta}_1\hat{v}_2 = Q_L^T\hat{u}_1 - \hat{\alpha}_1\hat{v}_1$. Hieruit volgt:

$$\hat{\alpha}_1\hat{\beta}_1\hat{v}_2 = \hat{\alpha}_1 Q_L^T\hat{u}_1 - \hat{\alpha}_1^2\hat{v}_1. \tag{20}$$

Verder merken we op dat $\hat{\alpha}_1 Q_L^T\hat{u}_1 = Q_L^T\hat{\alpha}_1\hat{u}_1$. Uit algoritme 3 halen we dat $\hat{\alpha}_1\hat{u}_1 = Q_L\hat{v}_1$. Hieruit volgt dat (20) gelijk is aan:

$$\hat{\alpha}_1 Q_L^T\hat{u}_1 - \hat{\alpha}_1^2\hat{v}_1 = Q_L^T Q_L\hat{v}_1 - \hat{\alpha}_1^2\hat{v}_1.$$

Met behulp van de vergelijking $Q_A^T Q_A + Q_L^T Q_L = I$ en $\tilde{v}_1 = v_1$ vinden we vervolgens:

$$\begin{aligned}\hat{\alpha}_1\hat{\beta}_1\hat{v}_2 &= (I - Q_A^T Q_A)\hat{v}_1 - \hat{\alpha}_1^2\hat{v}_1. \\ &= \hat{v}_1 - Q_A^T Q\hat{v}_1 - \hat{\alpha}_1^2\hat{v}_1 \\ &= v_1 - Q_A^T Q v_1 - \hat{\alpha}_1^2 v_1\end{aligned}$$

Op een soortgelijke manier met behulp van $\alpha_2 v_2 = Q_A^T u_2 - \beta_2 v_1$, $\beta_2 u_2 = Q_A v_1 - \alpha_1 u_1$ en $\alpha_1 v = Q_A^T u_1$ vinden we dat:

$$\beta_2\alpha_2 v_2 = Q_A^T Q_A v_1 - \alpha_1^2 v_1 - \beta_2^2 v_1.$$

Wanneer beide gelijkheden bij elkaar opgeteld worden, krijgen we de volgende gelijkheid:

$$\begin{aligned}\hat{\alpha}_1\hat{\beta}_1\hat{v}_2 + \beta_2\alpha_2 v_2 &= v_1 - Q_A^T Q v_1 - \hat{\alpha}_1^2 v_1 + Q_A^T Q_A v_1 - \alpha_1^2 v_1 - \beta_2^2 v_1 \\ &= (1 - \hat{\alpha}_1^2 - \alpha_1^2 - \beta_2^2)v_1\end{aligned}$$

Door de manier waarop we v_1, v_2 en $\tilde{v}_2$ hebben geconstrueerd, geldt dat zowel v_2 en $\hat{v}_2$ orthogonaal zijn met v_1. Wanneer we de bovenstaande vergelijking met v_2^T of $\hat{v}_2^T$ vanaf links vermenigvuldigen wordt hierdoor de rechterkant van de vergelijking nul. Dit geeft voor de vermenigvuldiging met v_2^T:

$$\begin{aligned}\hat{\alpha}_1\hat{\beta}_1 v_2^T\hat{v}_2 + \beta_2\alpha_2 v_2^T v_2 &= (1 - \hat{\alpha}_1^2 - \alpha_1^2 - \beta_2^2)v_2^T v_1 \\ &= 0,\end{aligned}$$

ofwel:

$$\begin{aligned}\hat{\alpha}_1\hat{\beta}_1 v_2^T \hat{v}_2 &= -\beta_2\alpha_2 v_2^T v_2 \\ &= -\beta_2\alpha_2\end{aligned}$$

Wanneer we een deling uitvoeren vinden we de volgende gelijkheid:

$$v_2^T \hat{v}_2 = -\frac{\alpha_2\beta_2}{\hat{\alpha}_1\hat{\beta}_1}.$$

Op dezelfde manier vinden we door middel van een vermenigvuldiging met $\hat{v}_2^T$

$$\hat{v}_2^T v_2 = -\frac{\hat{\alpha}_1\hat{\beta}_1}{\alpha_2\beta_2}.$$

We kunnen de laatst genoemde vergelijkingen nu aan elkaar gelijk stellen. Door middel van herschrijven vinden we dat $(\hat{\alpha}_1\hat{\beta}_1)^2 = (\alpha_2\beta_2)^2$. Vanwege de aanname dat $\hat{\alpha}_1$, $\hat{\beta}_1$, α_2 en β_2 strikt positief zijn, is de enige oplossing dat $\alpha_2\beta_2 = \hat{\alpha}_1\hat{\beta}_1$. Hieruit volgt vervolgens dat $\hat{v}_2^T v_2 = -1$ ofwel $\hat{v}_2^T = -v_2$. Dit was wat we in de eerste stap wouden bewijzen.

We nemen nu aan dat de gelijkheden gelden voor alle indices tot en met $j-1$. Vervolgens moeten we bewijzen dat het ook waar is voor j, ofwel we moeten bewijzen dat $\hat{v}_{j+1} = (-1)^k v_{j+1}$ en $\hat{\alpha}_j\hat{\beta}_j = \alpha_{j+1}\beta_{j+1}$.

We merken op dat we de inductie hypothese $\hat{v}_{j+1} = (-1)^k v_{j+1}$ voor $j = 0, 1, \ldots, j-1$ kunnen weergeven met de volgende matrix notatie:

$$\hat{V}_j = V_j P_j.$$

Hierbij is $P \in \mathbb{R}^{j\times j}$ een diagonaal matrix met op de diagonaal de vector $(1, -1, 1, -1, \ldots)$.

We gebruiken nu wederom $I = Q_A^T Q_A + Q_L^T Q_L$ om een gelijkheid af te leiden die we verderop in het bewijs nodig hebben. Met behulp van (16) en (18) vinden we dat:

$$\begin{aligned}I &= Q_A^T Q_A + Q_L^T Q_L \\ &= (U_{j+1}B_j V_j^T)^T U_{j+1}B_j V_j^T + (\hat{U}_j\hat{B}_j\hat{V}_j^T)^T \hat{U}_j\hat{B}_j\hat{V}_j^T \\ &= V_j B_j^T U_{j+1}^T U_{j+1} B_j V_j^T + \hat{V}_j\hat{B}_j^T\hat{U}_j^T\hat{U}_j\hat{B}_j\hat{V}_j^T \\ &= V_j B_j^T B_j V_j^T + \hat{V}_j\hat{B}_j^T\hat{B}_j\hat{V}_j^T. \\ &= V_j B_j^T B_j V_j^T + V_j P_j \hat{B}_j^T \hat{B}_j P_j V_j^T\end{aligned}$$

Hierbij geldt de een-na-laatste gelijkheid door de orthonormale kolommen van U en $\hat{U}$ en de laatste door het eerste deel van de inductie hypothese en het feit dat $P_j^T = P_j$. De volgende stap is het vermenigvuldigen met V_j en V_j^T respectievelijk vanaf rechts en links. Omdat V_j orthonormale kolommen heeft geldt namelijk dat $V_j^T V_j = I_j$ Dit geeft:

$$\begin{aligned}I_j = V_j^T V_j &= V_j^T V_j B_j^T B_j V_j^T V_j + V_j^T V_j P_j \hat{B}_j^T \hat{B}_j P_j V_j^T V_j \\ &= B_j^T B_j + P_j \hat{B}_j^T \hat{B}_j P_j\end{aligned}$$

We bekijken nu (16). Deze vergelijking vermenigvuldigen we met Q_A^T. Vervolgens gebruiken we (17) om het te herschrijven. Dit geeft:

$$\begin{aligned} Q_A^T Q_A V_j &= Q_A^T U_{j+1} B_j \\ &= (V_j B_j^T + \alpha_{j+1} v_{j+1} e_{j+1}^T) B_j \\ &= V_j B_j^T B_j + \alpha_{j+1} v_{j+1} e_{j+1}^T B_j \end{aligned}$$

We merken op dat $e_{j+1}^T B_j$ gelijk is aan $\beta_{j+1} e_j^T$ vanwege de constructie van B_j. Wanneer we dit invullen krijgen we:

$$Q_A^T Q_A V_j = V_j B_j^T B_j + \alpha_{j+1} \beta_{j+1} v_{j+1} e_j^T.$$

Op soortgelijke manier vinden we met behulp van (18) en (19) en $e_j^T \hat{B}_j = \alpha e_j^T$ dat $Q_L^T Q_L \hat{V}_j = \hat{V}_j \hat{B}_j^T \hat{B}_j + \hat{\alpha}_j \hat{\beta}_j \hat{v}_{j+1} e_j^T$. Vervolgens passen we hierop de inductie hypothese toe en gebruiken we dat P een orthonormale matrix is met $e_j^T P_j = (-1)^{j+1} e_j^T$. Dit geeft:

$$\begin{aligned} Q_L^T Q_L V_j P_j &= V_j P_j \hat{B}_j^T \hat{B}_j + \hat{\alpha}_j \hat{\beta}_j \hat{v}_{j+1} e_j^T \\ Q_L^T Q_L V_j P_j P_j &= V_j P_j \hat{B}_j^T \hat{B}_j P_j + \hat{\alpha}_j \hat{\beta}_j \hat{v}_{j+1} e_j^T P_j \\ Q_L^T Q_L V_j &= V_j P_j \hat{B}_j^T \hat{B}_j P_j + \hat{\alpha}_j \hat{\beta}_j (-1)^{j+1} \hat{v}_{j+1} e_j^T. \end{aligned}$$

Wanneer we de twee verkregen gelijkheden bij elkaar optellen krijgen we het volgende:

$$(Q_A^T Q_A + Q_L^T Q_L) V_j = V_j (B_j^T B_j + P_j \hat{B}_j^T \hat{B} P_j) + \alpha_{j+1} \beta_{j+1} v_{j+1} e_j^T + \hat{\alpha}_j \hat{\beta}_j (-1)^{j+1} \hat{v}_{j+1} e_j^T$$

Met behulp van de eerder afgeleide gelijkheden vinden we nu dat:

$$I V_j = V_j I + \alpha_{j+1} \beta_{j+1} v_{j+1} e_j^T + \hat{\alpha}_j \hat{\beta}_j (-1)^{j+1} \hat{v}_{j+1} e_j^T.$$

Hieruit volgt dat de restterm gelijk is aan nul en dus moet gelden dat

$$\alpha_{j+1} \beta_{j+1} v_{j+1} + \hat{\alpha}_j \hat{\beta}_j (-1)^{j+1} \hat{v}_{j+1} = 0.$$

Met een soortgelijke redenatie als bij het geval $j = 2$ volgt nu dat $\hat{v}_{j+1} = (-1)^k v_{j+1}$ en $\hat{\alpha}_j \hat{\beta}_j = \alpha_{j+1} \beta_{j+1}$ hetgeen wat we wouden bewijzen. □

Merk op: stelling 4.3 kan herschreven worden als:

Stelling 4.4. *Gegeven de samengevoegde bidiagonalisatie van Q_A en Q_L zoals beschreven in algoritme 2 en 3. Dan gelden de volgende gelijkheden:*

$$\hat{V}_j = V_j P_j \quad \text{en} \quad \hat{\alpha}_j \hat{\beta}_j = \alpha_{j+1} \beta_{j+1}$$

waarbij $j = 1, 2, \ldots, \min(p, n) - 1$ en $P_j \in \mathbb{R}^{j \times j}$ een diagonaalmatrix met op de diagonaal de vector
$(1, -1, 1, \ldots, (-1)^{j+1})$ *is.*

De voorgaande stellingen kunnen we gebruiken om een stelling te definiëren die de twee bidiagonalisatie algoritmes samenvoegt om het matrix paar$\{A, L\}$ te bidiagonaliseren.

Stelling 4.5. *[5, stelling 2.2][Samengevoegde Bidiagonalisatie van A en L] Laat $A \in \mathbb{R}^{m\times n}$ en $L \in \mathbb{R}^{p\times n}$ met $m \geq n$. Dan bestaan er orthogonale matrices $U \in \mathbb{R}^{m\times m}$, $\hat{U} \in \mathbb{R}^{p\times p}$ en $V \in \mathbb{R}^{n\times n}$, een beneden bidiagonaal matrix $B \in \mathbb{R}^{m\times n}$, een boven bidiagonaal matrix $\bar{B} \in \mathbb{R}^{p\times n}$ en een inverteerbare matrix Z zo dat:*

$$A = UBZ^{-1}, \quad L = \hat{U}\bar{B}Z^{-1}, \quad Z = R^{-1}V.$$

Hierbij is R de bovendriehoeks matrix verkregen door de QR ontbinding van $\begin{bmatrix} A \\ L \end{bmatrix}$, $\bar{B} = \hat{B}P$, P de diagonaalmatrix met diagonaal $(1, -1, 1, \ldots, (-1)^{j+1})$ is en zijn alle andere matrices zoals in de bidiagonalistatie algoritmes gedefinieerd. Voor $p < n$ geldt dat de kolommen $p + 1$ tot n van $\bar{B}$ alleen nullen bevatten.

Bewijs: Het bewijs volgt door het uitwerken van de definities en de vergelijkingen verkregen door de volledige uitvoering van beide algoritmes. We merken wel op dat bij het volledig uitvoeren niet gelijk de gewenste dimensies ontstaan. Om dit op te lossen kunnen de missende rijen en kolommen van B en $\bar{B}$ gevormd worden door nullen. De missende kolommen van U, $\hat{U}$ en V kunnen gemaakt worden door een orthonormale vector van de juiste lengte. Op deze manier voldoen de matrices aan de gewenste eisen. Voor het verdere verloop heeft dit geen invloed. We gebruiken alleen het algoritme tot zijn volledige uitvoering. □

Het gevolg van stelling 4.5 is dat het matrix paar A, L samengevoegd gebidiagonaliseerd kan worden. De ontbinding heet de samengevoegde bidiagonalisatie van A en L en ziet er als volgt uit:

$$AZ_k = U_{k+1}B_k \quad \text{en} \quad LZ_k = \hat{U}_k\bar{B}_k \tag{21}$$

Voordat we verder gaan beantwoorden we eerst de vraag waarom we eerst de QR ontbinding van $[A\ L]$ bekijken en daarop de bidiagonalisatie toepassen. Waarom bidiaginaliseren we niet gelijk A en L? Het antwoord ligt in het bewijs van stelling 4.3. Hierin wordt gebruik gemaakt van de relatie $Q_A^TQ_A + Q_L^TQ_L = I$ welke komt van de orthonormale kolommen van Q. We kunnen zoiets niet zeggen over de A en L. Hierdoor geldt stelling 4.3 niet voor A en L. Er hoeft dus geen verband te zijn tussen de rechtstreekse beneden bidiagonalisatie van A en de boven bidiagonalisatie van L. Hierdoor is het niet mogelijk om een samengevoegd bidiagonalisatie ontbinding te vinden van dezelfde vorm.

4.4 Tikhonov met samengevoegde bidiagonalisatie

We passen nu het samengevoegde bidiagonalisatie algoritme toe op het Tikhonov probleem (6). Dit doen we door (6) te projecteren naar de ruimte $\mathcal{Z} = \text{span}\{z_1, z_2, \ldots, z_k\}$.

$$x_k^\lambda = \text{argmin}_{x\in\mathcal{Z}} \left\| \begin{bmatrix} A \\ \sqrt{\lambda}L \end{bmatrix} x - \begin{bmatrix} b \\ 0 \end{bmatrix} \right\|_2, \quad \mathcal{Z} = \text{span}\{z_1, z_2, \ldots, z_k\}$$

We stellen $x_k = Z_ky_k$ waarbij Z_k de matrix is met als kolommen de eerste k kolommen van matrix Z. Op deze manier zeggen we eigenlijk dat $x_k \in \mathcal{Z}$. De norm kan als volgt herschreven worden door middel van de samengevoegde bidiagonalisatie:

$$
\begin{aligned}
\left\|\begin{bmatrix} A \\ \sqrt{\lambda}L \end{bmatrix} x - \begin{bmatrix} b \\ 0 \end{bmatrix}\right\|_2 &= \left\|\begin{bmatrix} AZ_k \\ \sqrt{\lambda}LZ_k \end{bmatrix} y - \begin{bmatrix} b \\ 0 \end{bmatrix}\right\|_2 \\
&= \left\|\begin{bmatrix} U_{k+1}B_k \\ \sqrt{\lambda}\hat{U}_k\bar{B}_k \end{bmatrix} y - \begin{bmatrix} b \\ 0 \end{bmatrix}\right\|_2 \\
&= \left\|\begin{bmatrix} U_{k+1} & 0 \\ 0 & \hat{U}_k \end{bmatrix} \begin{bmatrix} B_k \\ \sqrt{\lambda}\bar{B}_k \end{bmatrix} y - \begin{bmatrix} b \\ 0 \end{bmatrix}\right\|_2 \\
&= \left\|\begin{bmatrix} U_{k+1} & 0 \\ 0 & \hat{U}_k \end{bmatrix} \left(\begin{bmatrix} B_k \\ \sqrt{\lambda}\bar{B}_k \end{bmatrix} y - \begin{bmatrix} \beta_1 e_1 \\ 0 \end{bmatrix}\right)\right\|_2 \\
&= \left\|\begin{bmatrix} B_k \\ \sqrt{\lambda}\bar{B}_k \end{bmatrix} y - \begin{bmatrix} \beta_1 e_1 \\ 0 \end{bmatrix}\right\|_2
\end{aligned}
$$

Hierbij geldt de laatste gelijkheid door de orthonormale kolommen van U_{k+1} en $\hat{U}_k$.

De een-na-laatste gelijkheid komt door de volgende twee eigenschappen die volgen uit de constructie van ons algoritme:

$$U_{k+1}(\beta_1 e_1) = \beta_1 u_1 = b \quad \text{en} \quad \beta_1 = ||b||.$$

Kortom we kunnen de algemene vorm van Tikhonov regularisatie als volgt herschrijven met behulp van de samengevoegde bidiagonalisatie :

$$x_k^\lambda = Z_k y_k^\lambda \quad \text{met} \quad y_k^\lambda = \operatorname{argmin}\left\|\begin{bmatrix} B_k \\ \sqrt{\lambda}\bar{B}_k \end{bmatrix} y - \begin{bmatrix} \beta_1 e_1 \\ 0 \end{bmatrix}\right\|_2 \tag{22}$$

We kunnen deze vergelijking gebruiken voor het oplossen van het probleem. Een manier om het aan te pakken is door voor $k = 1, 2, \ldots$ de samengevoegde bidiagonalisatie te berekenen. Daarna wordt y_k^λ berekend door (22) op te lossen. We zullen later zien dat we de normen in (6) direct kunnen berekenen als we y_k^λ en V_k weten. Hierdoor kunnen we λ direct uit y_k^λ berekend worden. Wanneer k groot genoeg is, kan x_k^λ berekend worden en hebben we de oplossing.

We gaan er hierbij van uit dat voor k groot genoeg, het geprojecteerde probleem de belangrijkste informatie van het oorspronkelijke probleem bevat. Alleen als dit het geval is kunnen we zeggen dat x_k^λ een goede benadering zal zijn voor de oplossing van (6). De vraag is echter wanneer dit het geval is. Het antwoord van deze vraag hebben we hier niet behandeld. Het is een onderwerp waar nog verder onderzoek naar gedaan kan worden.

4.5 Een efficiëntere manier voor het gebruik

De voorgaande paragraaf gaf weer hoe het samengevoegde bidiagonalisatie algoritme werkt en hoe het gebruikt kan worden bij het oplossen van de Tikhonov vergelijking. Wanneer dit algoritme rechtstreeks gebruikt wordt moet er een QR ontbinding uitgevoerd worden. Dit kost veel operaties en opslag. In dit hoofdstuk worden een aantal opmerkingen gemaakt wat betreft het efficiënter maken van het algoritme.

4.5.1 Samengevoegde Lanczos bidiagonalisatie 2.0

Voor het samengevoegde Lanczos bidiagonalisatie algoritme wat we nu hebben is de QR ontbinding van $\begin{bmatrix} A \\ L \end{bmatrix}$ nodig. Dit is een kostbare operatie zowel in operaties als in opslag. We willen dit daarom niet uitvoeren. Het probleem is dat we nu geen toegang tot Q_A en Q_L hebben en we kunnen algoritme 2 en 3 niet uitvoeren. Om dit op te lossen bekijken we beide algoritmes nog een keer. De bedoeling is dat we een nieuw algoritme krijgen dat niet meer afhangt van Q_A en Q_L en dat efficiënter te berekenen is.

Voordat we beginnen met het omschrijven van de algoritmes merken we het volgende op. We kunnen Q_A en Q_L schrijven als een product met Q. Dit ziet er als volgt uit:

$$Q_A = \begin{bmatrix} I_m \\ 0_{p\times m} \end{bmatrix}^T Q \quad \text{en} \quad Q_L = \begin{bmatrix} 0_{m\times p} \\ I_p \end{bmatrix}^T Q.$$

Verder definiëren we een aantal nieuwe vectoren:

$$\tilde{u}_j = \begin{bmatrix} I_m \\ 0_{p\times m} \end{bmatrix} u_j = \begin{bmatrix} u_j \\ 0_p \end{bmatrix}, \quad \tilde{v}_j = Qv_j, \quad \bar{u}_j = \begin{bmatrix} 0_{m\times p} \\ I_p \end{bmatrix} \hat{u}_j = \begin{bmatrix} 0_m \\ \hat{u}_j \end{bmatrix}, \quad \bar{v}_j = Q\hat{v}_j.$$

Met behulp van bovenstaande vergelijkingen gaan we de algoritmes een voor een omschrijven. We beginnen met het beneden bidiagonalisatie algoritme in 2. De eerste regel is $\beta_1 u_1 = b$. We vermenigvuldigen dit vanaf links met $\begin{bmatrix} I_m \\ 0_{p\times m} \end{bmatrix}^T$. Herschrijven met behulp van de eerste vergelijking levert het volgende op

$$\beta_1 \tilde{u}_1 = \begin{bmatrix} I_m \\ 0_{p\times m} \end{bmatrix}^T b.$$

De tweede regel van het algoritme is $\alpha_1 v = Q_A^T u_1$. Deze vermenigvuldigen we met Q. Dit geeft:

$$\begin{aligned} \alpha_1 Qv &= QQ_A^T u_1 \\ &= QQ^T \begin{bmatrix} I_m \\ 0_{p\times m} \end{bmatrix} u_1 \\ &= QQ^T \tilde{u}_1 \end{aligned}$$

Er geldt echter ook dat $\alpha_1 Qv = \alpha_1 \tilde{v}_1$. Hierdoor geldt:

$$\alpha_1 \tilde{v}_1 = QQ^T \tilde{u}_1$$

Op soortgelijke manier vinden we ook dat

$$\beta_{i+1}\tilde{u}_{i+1} = \begin{bmatrix} I_m \\ 0_{p\times m} \end{bmatrix} \begin{bmatrix} I_m \\ 0_{p\times m} \end{bmatrix}^T \tilde{v}_i - \alpha_i u_i$$

$$\alpha_{i+1}\tilde{v}_{i+1} = QQ^T\tilde{u}_{i+1} - \beta_{i+1}\tilde{v}_i.$$

Het aangepaste algoritme is te vinden in algoritme 4.

Algorithm 4: Beneden Lanczos Bidiagonalisatie aangepast

$\beta_1\tilde{u}_1 = \begin{bmatrix} I_m \\ 0_{p\times m} \end{bmatrix}^T b$
$\alpha_1\tilde{v}_1 = QQ^T\tilde{u}_1$
voor $i = 1, 2, ...n-1$
  $\beta_{i+1}\tilde{u}_{i+1} = \begin{bmatrix} I_m \\ 0_{p\times m} \end{bmatrix} \begin{bmatrix} I_m \\ 0_{p\times m} \end{bmatrix}^T \tilde{v}_i - \alpha_i u_i$
  $\alpha_{i+1}\tilde{v}_{i+1} = QQ^T\tilde{u}_{i+1} - \beta_{i+1}\tilde{v}_i$
einde

We bekijken nu de boven bidiagonalisatie in algoritme 3. Dit wordt op een soortgelijke manier aangepast. De eerste vergelijking in het algoritme is $\hat{v}_1 = v_1$. Wanneer we vanaf links vermenigvuldigen met Q vinden we $\bar{v}_1 = \tilde{v}_1$. De tweede vergelijking is $\hat{\alpha}_1\hat{u}_1 = Q_L\hat{v}_1$. Herschrijven levert het volgende op:

$$\begin{aligned} \hat{\alpha}_1 \begin{bmatrix} 0_{m\times p} \\ I_p \end{bmatrix}^T \hat{u}_1 &= \begin{bmatrix} 0_{m\times p} \\ I_p \end{bmatrix} Q_L\hat{v}_1 \\ \hat{\alpha}_1\bar{u}_1 &= \begin{bmatrix} 0_{m\times p} \\ I_p \end{bmatrix} \begin{bmatrix} 0_{m\times p} \\ I_p \end{bmatrix}^T Q\hat{v}_1 \\ &= \begin{bmatrix} 0_{m\times p} \\ I_p \end{bmatrix} \begin{bmatrix} 0_{m\times p} \\ I_p \end{bmatrix}^T \bar{v}_1. \end{aligned}$$

Voor het vervolg van het algoritme maken we gebruik van stelling 4.3. Hieruit volgt namelijk dat $\hat{\beta}_i = \frac{\alpha_{i+1}\beta_{i+1}}{\hat{\alpha}_i}$ en dat $\hat{v}_{i+1} = (-1)^i v_{i+1}$ wat in onze nieuwe notatie gelijk is aan $\bar{v}_{i+1} = (-1)^i\tilde{v}_{i+1}$. We hebben dus vergelijkingen voor $\hat{\beta}_i$ en $\bar{v}_{i+1}$ die berekend kunnen worden door deling en vermenigvuldiging met scalars die bekend zijn in plaats van met de matrix Q_L die onbekend is.

Er is alleen nog een uitdrukking nodig voor $\hat{\alpha}_{i+1}$ en $\hat{u}_{i+1}$. Deze vinden we met behulp van de laatste regel van het oorspronkelijke algoritme: $\hat{\alpha}_{i+1}\hat{u}_{i+1} = Q_L\hat{v}_{i+1} - \hat{\beta}_i\hat{u}_i$. Omschrijven op al vaker gebruikte manier levert op:

$$\hat{\alpha}_{i+1}\bar{u}_{i+1} = \begin{bmatrix} 0_{m\times p} \\ I_p \end{bmatrix} \begin{bmatrix} 0_{m\times p} \\ I_p \end{bmatrix}^T \bar{v}_{i+1} - \hat{\beta}_i\bar{u}_i.$$

Het aangepaste algoritme is compleet weergegeven in algoritme 5.

Wanneer we beide nieuwe algoritmes opnieuw rangschikken vinden we algoritme 6. We hebben hier gebruik gemaakt de vele nullen in de algoritmes. Ze passen als het ware precies in elkaar. Dit

Algorithm 5: Boven Lanczos Bidiagonalisatie aangepast

$\bar{v}_1 = \tilde{v}_1$
$\hat{\alpha}_1 \bar{u}_1 = \begin{bmatrix} 0_{m\times p} \\ I_p \end{bmatrix} \begin{bmatrix} 0_{m\times p} \\ I_p \end{bmatrix}^T \bar{v}_1$
voor $i = 1, 2, ... \min(p, n) - 1$
  $\hat{\beta}_i = \frac{\alpha_{i+1}\beta_{i+1}}{\hat{\alpha}_i}$
  $\bar{v}_{i+1} = (-1)^i \tilde{v}_{i+1}$
  $\hat{\alpha}_{i+1}\bar{u}_{i+1} = \begin{bmatrix} 0_{m\times p} \\ I_p \end{bmatrix} \begin{bmatrix} 0_{m\times p} \\ I_p \end{bmatrix}^T \bar{v}_{i+1} - \hat{\beta}_i \bar{u}_i$
einde

Algorithm 6: Samengevoegde bidiagonalisatie

$\beta_1 u_1 = b$
$\alpha_1 \tilde{v}_1 = QQ^T \begin{bmatrix} u_1 \\ 0 \end{bmatrix}$
$\hat{\alpha}_1 \hat{u}_1 = \tilde{v}_1(m+1 : m+p)$
voor $i = 1, 2, \ldots$
  $\beta_{i+1} u_{i+1} = \tilde{v}_i(1 : m) - \alpha_i u_i$
  $\alpha_{i+1}\tilde{v}_{i+1} = QQ^T \begin{bmatrix} u_{i+1} \\ 0_p \end{bmatrix} - \beta_{i+1}\tilde{v}_i$
  $\hat{\beta}_i = (\alpha_{i+1}\beta_{i+1})/\hat{\alpha}_i$
  $\alpha_{i+1}\hat{u}_{i+1} = (-1)^i \tilde{v}_{i+1}(m+1 : m+p) - \hat{\beta}_i \hat{u}_i$
einde

algoritme 6 hangt niet meer van Q_A en Q_L af. We zien wel dat er een aantal vermenigvuldigen zijn met QQ^T en een vector. Wanneer we dit direct willen berekenen moeten we alsnog de QR ontbinding vinden. Gelukkig kan deze vermenigvuldiging op een relatief makkelijkere manier uitgevoerd worden.

Voor de reden waarom en hoe bekijken we het volgende kleinstekwadraten probleem:

$$\min_x \left\| \begin{bmatrix} A \\ L \end{bmatrix} x - \begin{bmatrix} u_i \\ 0 \end{bmatrix} \right\|_2 .$$

We weten uit eerdere analyses dat we x als volgt kunnen uitdrukken:

$$x = \left(\begin{bmatrix} A \\ L \end{bmatrix}^T \begin{bmatrix} A \\ L \end{bmatrix} \right)^{-1} \begin{bmatrix} A \\ L \end{bmatrix}^T \begin{bmatrix} u_i \\ 0 \end{bmatrix} . \tag{23}$$

Voordat we verder gaan merken we op dat de volgende gelijkheid geldt:

$$\begin{aligned} \begin{bmatrix} A \\ L \end{bmatrix} \left(\begin{bmatrix} A \\ L \end{bmatrix}^T \begin{bmatrix} A \\ L \end{bmatrix} \right)^{-1} \begin{bmatrix} A \\ L \end{bmatrix}^T &= QR(R^T Q^T QR)^{-1} R^T Q^T \\ &= QR(R^T R)^{-1} R^T Q^T \\ &= QRR^{-1}(R^T)^{-1} R^T Q^T \\ &= QQ^T . \end{aligned}$$

Hierbij geldt dat de inverse van R bestaat aangezien $R \in \mathbb{R}^{n\times n}$ en $[A\ L]$ volledige kolomrang heeft.

We zien dat

$$\begin{bmatrix} A \\ L \end{bmatrix} \left(\begin{bmatrix} A \\ L \end{bmatrix}^T \begin{bmatrix} A \\ L \end{bmatrix} \right)^{-1} \begin{bmatrix} A \\ L \end{bmatrix}^T$$

bijna helemaal in de uitdrukking (23) voorkomt. We hoeven alleen (23) te vermenigvuldigen met $\begin{bmatrix} A \\ L \end{bmatrix}$ vanaf links en dan staat deze term voor $[u_i\ 0]$. Met andere woorden:

$$\begin{aligned} \begin{bmatrix} A \\ L \end{bmatrix} x &= \begin{bmatrix} A \\ L \end{bmatrix} \left(\begin{bmatrix} A \\ L \end{bmatrix}^T \begin{bmatrix} A \\ L \end{bmatrix} \right)^{-1} \begin{bmatrix} A \\ L \end{bmatrix}^T \begin{bmatrix} u_i \\ 0 \end{bmatrix} \\ &= QQ^T \begin{bmatrix} u_i \\ 0 \end{bmatrix} \end{aligned}$$

De laatste stap is het inzien dat $QQ^T[u_i\ 0]$ een orthogonale projectie is op het bereik van $[A\ L]$.[5] Hierdoor geldt dat $QQ^T[u_i\ 0] = [u_i\ 0]$. We zien dat $QQ^T[u_i\ 0]$ berekend kan worden door de vergelijking $\begin{bmatrix} A \\ L \end{bmatrix} x = \begin{bmatrix} u_i \\ 0 \end{bmatrix}$ op te lossen. Wanneer we de gevonden x vervolgens weer vermenigvuldigen met $[A\ L]$ krijgen we $QQ^T[u_i\ 0]$. Dit is de vector die we wouden berekenen. Een van de manieren om x te bepalen is door middel van het in paragraaf 3.3 genoemde LSQR algoritme. Op deze manier hoeven we QQ^T niet expliciet te berekenen. Belangrijk wel is dat er wel een stelsel opgelost moet worden. Wanneer L redelijk goed gesteld is, verwachten we dat dit geen probleem oplevert [5].

We hebben het samengevoegde bidiagonalisatie algoritme nu herschreven op een manier dat er geen QR ontbinding meer nodig is. Het huidige algoritme bestaat uit matrix vector vergelijkingen en het oplossen van een kleinstekwadraten probleem.

4.5.2 Tikhonov met samengevoegde bidiagonalisatie 2.0

In deze paragraaf vertellen we hoe je de samengevoegde bidiagonalisatie op een efficiënte manier kunt gebruiken om y_k^λ en x_k^λ te berekenen uit (22).

We beginnen met het vinden van y_k^λ. We willen deze verkrijgen uit (22). Om dit te doen nemen we de standaard QR ontbinding van $\begin{bmatrix} B_k \\ \sqrt{\lambda}\bar{B}_k \end{bmatrix}$. Deze schrijven we als

$$\begin{bmatrix} B_k \\ \sqrt{\lambda}\bar{B}_k \end{bmatrix} = Q_k^\lambda \begin{bmatrix} R_k^\lambda \\ 0 \end{bmatrix}.$$

Hierbij geldt dat $Q_k^\lambda \in \mathbb{R}^{(2k+1)\times(2k+1)}$ een orthogonale matrix is en $R_k^\lambda \in \mathbb{R}^{(k\times k)}$. We gebruiken deze notatie om te benadrukken dat Q_k^λ en R_k^λ van k en λ afhangen. We definiëren vervolgens vectoren f_k^λ en g_k^λ van respectievelijk lengte k en $k+1$ als volgt:

$$\begin{bmatrix} \beta_1 e_1 \\ 0 \end{bmatrix} = Q_k^\lambda \begin{bmatrix} f_k^\lambda \\ h_k^\lambda \end{bmatrix}.$$

Op deze manier vinden we:

$$\begin{bmatrix} B_k & \beta_1 e_1 \\ \sqrt{\lambda}\bar{B}_k & 0 \end{bmatrix} = Q_k^\lambda \begin{bmatrix} R_k^\lambda & f_k^\lambda \\ 0 & h_k^\lambda \end{bmatrix}.$$

In [5] en [4] wordt benoemd hoe met behulp van Givens rotaties, $\begin{bmatrix} R_k^\lambda & f_k^\lambda \\ 0 & h_k^\lambda \end{bmatrix}$ berekend kan worden zonder Q_k^λ te bepalen. Op deze manier hoeft de QR ontbinding niet bepaald te worden en geldt er dat R_k^λ een boven bidiagonaal matrix is. Bovendien kan de operatie die beschreven wordt in $O(k)$ operaties gedaan worden.

Wanneer we naar de normaal vergelijking behorende bij het probleem (22) kijken merken we het volgende op:

$$\begin{bmatrix} B_k \\ \bar{B}_k \end{bmatrix}^T \begin{bmatrix} B_k \\ \bar{B}_k \end{bmatrix} y_k^\lambda = \begin{bmatrix} B_k \\ \bar{B}_k \end{bmatrix}^T \begin{bmatrix} \beta_1 e_1 \\ 0 \end{bmatrix}$$
$$\begin{bmatrix} R_k^\lambda \\ 0 \end{bmatrix}^T (Q_k^\lambda)^T Q_k^\lambda \begin{bmatrix} R_k^\lambda \\ 0 \end{bmatrix} y_k^\lambda = \begin{bmatrix} R_k^\lambda \\ 0 \end{bmatrix}^T (Q_k^\lambda)^T \begin{bmatrix} \beta_1 e_1 \\ 0 \end{bmatrix}$$
$$\begin{bmatrix} R_k^\lambda \\ 0 \end{bmatrix}^T \begin{bmatrix} R_k^\lambda \\ 0 \end{bmatrix} y_k^\lambda = \begin{bmatrix} R_k^\lambda \\ 0 \end{bmatrix}^T \begin{bmatrix} f_k^\lambda \\ h_k^\lambda \end{bmatrix}$$
$$(R_k^\lambda)^T R_k^\lambda y_k^\lambda = (R_k^\lambda)^T f_k^\lambda$$

Hierbij hebben we gebruik gemaakt van de orthonormale eigenschappen van Q_k^λ.

We zien dat nu geldt dat:

$$R_k^\lambda y_k^\lambda = f_k^\lambda.$$

We hebben nu een vergelijking om y_k^λ te berekenen. Omdat R_k^λ bovenbidiagonaal is kan deze operatie ook in $O(k)$ operaties berekent worden [5].

We merken op dat voor deze berekening nog wel een λ nodig is. Het zou fijn zijn als we niet voor alle y_k terug hoeven te rekenen naar x_k voor het vinden van λ. Veel methodes die de optimale λ bepalen maken namelijk gebruik van de normen $||Ax_k^\lambda - b||_2$ en $||Lx_k^\lambda||_2$. Gelukkig is er de volgende stelling die zegt dat het berekenen van x_k^λ niet nodig is voor het berekenen van deze normen.

Stelling 4.6. *[5, stelling 3.1] Stel dat R_k^λ, y_k^λ, f_k^λ en h_k^λ voldoen aan de bovenstaande definities en dat B_k en $\bar{B}_k$ de matrices zijn zo als eerder. Dan gelden de volgende gelijkheden:*

$$||Ax_k^\lambda - b||_2 = ||B_k y_k^\lambda - \beta_1 e_1||_2 = (||h_k^\lambda||_2^2 - \lambda||\bar{B}_k y_k^\lambda||_2^2)^{\frac{1}{2}}$$

en

$$||Lx_k^\lambda||_2 = ||\bar{B}_k y_k^\lambda||_2.$$

Bewijs: We beginnen met de eerste gelijkheid. We gebruiken de variabelen zo als ze in de voorgaande paragrafen gedefinieerd zijn. We gebruiken in het speciaal vergelijking (21), $x_k^\lambda =$

$Z_k y_k^\lambda$ en de orthonormale kolommen van U_{k+1}.

$$\begin{aligned}
||Ax_k^\lambda - b||_2 &= ||U_{k+1}B_k Z_k^{-1} Z_k y_k^\lambda - b||_2 \\
&= ||U_{k+1}B_k y_k^\lambda - b||_2 \\
&= ||U_{k+1}B_k y_k^\lambda - U_{k+1}\beta_1 e_1||_2 \\
&= ||B_k y_k^\lambda - \beta_1 e_1||_2. \qquad (24)
\end{aligned}$$

Bovendien geldt dat

$$||B_k y_k^\lambda - \beta_1 e_1||_2^2 - \lambda ||B_k y_k^\lambda||_2^2 = ||h_k^\lambda||_2^2.$$

Dit is op te maken uit de volgende afleiding:

$$\begin{aligned}
||B_k y_k^\lambda - \beta_1 e_1||_2^2 - \lambda ||B_k y_k^\lambda||_2^2 &= \left|\left| \begin{bmatrix} B_k \\ \bar{B}_k \end{bmatrix} y_k^\lambda - \begin{bmatrix} \beta_1 e_1 \\ 0 \end{bmatrix} \right|\right|_2^2 \\
&= \left|\left| Q_k^\lambda \begin{bmatrix} R_k^\lambda \\ 0 \end{bmatrix} y_k^\lambda - Q_k^\lambda \begin{bmatrix} f_k^\lambda \\ h_k^\lambda \end{bmatrix} \right|\right|_2^2 \\
&= \left|\left| \begin{bmatrix} f_k^\lambda \\ 0 \end{bmatrix} - \begin{bmatrix} f_k^\lambda \\ h_k^\lambda \end{bmatrix} \right|\right|_2^2 \\
&= \left|\left| - \begin{bmatrix} 0 \\ h_k^\lambda \end{bmatrix} \right|\right|_2^2 \\
&= ||h_k^\lambda||_2^2
\end{aligned}$$

Met behulp van deze vergelijking krijgen we nu dat (24) gelijk is aan

$$(||h_k^\lambda||_2^2 - \lambda ||\bar{B}_k y_k^\lambda||_2^2)^{\frac{1}{2}}.$$

Voor de tweede gelijkheid maken we gebruik van (21), $x_k^\lambda = Z_k y_k^\lambda$ en de orthonormale kolommen van $\hat{U}_k$.

$$\begin{aligned}
||Lx_k^\lambda||_2 &= ||\hat{U}_k \bar{B}_k Z_k^{-1} Z_k y_k^\lambda||_2 \\
&= ||\hat{U}_k \bar{B}_k y_k^\lambda||_2 \\
&= ||\bar{B}_k y_k^\lambda||_2
\end{aligned}$$

We hebben nu bewezen wat we moesten bewijzen. □

Ook het terugrekenen van y_k^λ naar x_k^λ kan efficiënter. Bij het oorspronkelijke algoritme wordt x_k^λ berekend door $x_k^\lambda = Z_k y_k^\lambda$ op te lossen. Hierbij geldt dat Z_k de eerste k kolommen is van de matrix $R^{-1}V$. We hebben V niet volledig en we willen R ook niet berekenen. Gelukkig is er ook hiervoor een andere manier. Voor het terug gaan naar x_k^λ maken we gebruik van $\tilde{V}_k$. Deze matrix is namelijk wel bekend. Bovendien kunnen we de volgende afleiding maken:

$$\tilde{V}_k = QV_k = QRR^{-1}V_k = \begin{bmatrix} A \\ L \end{bmatrix} Z_k.$$

Door gebruik van $x_k^\lambda = Z_k y_k^\lambda$ vinden we

$$\tilde{V}_k y_k^\lambda = \begin{bmatrix} A \\ L \end{bmatrix} Z_k y_k^\lambda = \begin{bmatrix} A \\ L \end{bmatrix} x_k^\lambda.$$

De variabelen $\tilde{V}_k$, y_k^λ en $\begin{bmatrix} A \\ L \end{bmatrix}$ zijn bekend. De vermenigvuldiging $\tilde{V}_k y_k^\lambda$ kost $O((m+p)k)$ vermenigvuldigingen. Voor het vinden van x_k^λ kunnen we nu een iteratieve methode gebruiken zoals LSQR om de bovenstaande vergelijking op te lossen.

4.6 Juistheid oplossing

In deze paragraaf bekijken we hoe goed onze gevonden oplossing is. Hiervoor zoeken we de ruimte waar oplossing in ligt. Bovendien vergelijken we deze met die van een andere bekende oplossingsmethode, namelijk de LSQR methode uitgevoerd op de matrix $[A\ \sqrt{\lambda}L]$.

We beginnen met de ruimte van onze oplossing. We bekijken (6) nog eens nader met behulp van de QR ontbinding van $[A\ L]$. We zien dat we de volgende vergelijking krijgen:

$$x^\lambda = \underset{x}{\operatorname{argmin}} \left\| \begin{bmatrix} Q_A \\ \sqrt{\lambda} Q_L \end{bmatrix} Rx - \begin{bmatrix} b \\ 0 \end{bmatrix} \right\|_2 .$$

We stellen dat $w^\lambda = Rx^\lambda$. We krijgen nu het volgende probleem:

$$x^\lambda = R^{-1} w^\lambda, \quad w^\lambda = \underset{w}{\operatorname{argmin}} \left\| \begin{bmatrix} Q_A \\ \sqrt{\lambda} Q_L \end{bmatrix} w - \begin{bmatrix} b \\ 0 \end{bmatrix} \right\|_2 .$$

We hebben de samengevoegde bidiagonalisatie uitgevoerd op het matrix paar Q_A, Q_L. We bekijken wat dit betekent voor het probleem.

Ons samengevoegde bidiagonalisatie algoritme berekent onder andere de matrix V_k. Deze matrix is gebaseerd op Q_A en b. In paragraaf 3.3 hebben we gezien dat wanneer beneden Lanczos bidiagonalisatie wordt toegepast op A en b dat de kolommen van $V_{k,A}$ een basis vormen voor de Krylov deelruimte opgespannen door $\mathcal{K}_k(A^T A, A^T b)$. In ons geval betekent dat dus dat V_k een basis vormt voor de Krylov deelruimte $\mathcal{K}_k(Q_A^T Q_A, Q_A^T b)$ aangzien we nu het Lanczos bidiagonalisatie algoritme hebben toegepast op Q_A en b.

Het idee is nu dat we het voorgaande probleem projecteren op deze Krylov deelruimte. We hebben dan $w_k^\lambda = V_k y_k^\lambda$ ofwel we zoeken een $w_k^\lambda \in \mathcal{K}_k(Q_A^T Q_A, Q_A^T b)$. Het projectie probleem ziet er als volgt uit:

$$w_k^\lambda = \underset{w \in \mathcal{K}_k(Q_A^T Q_A, Q_A^T b)}{\operatorname{argmin}} \left\| \begin{bmatrix} Q_A \\ \sqrt{\lambda} Q_L \end{bmatrix} w - \begin{bmatrix} b \\ 0 \end{bmatrix} \right\|_2 .$$

We zien dat dit volgens de eerder genoemde redenaties hetzelfde is aan:

$$w_k^\lambda = V_k y_k^\lambda, \quad y_k^\lambda = \underset{y}{\operatorname{argmin}} \left\| \begin{bmatrix} B_k \\ \sqrt{\lambda} \bar{B}_k \end{bmatrix} y - \begin{bmatrix} \beta_1 e_1 \\ 0 \end{bmatrix} \right\|_2 .$$

De volgende stelling vat de zojuist gevoerde redenatie samen.

Stelling 4.7. *[5, stelling 4.3] De oplossing gegeven door ons algoritme is gelijk aan:*

$$x_k^\lambda = R^{-1} w_k^\lambda, \quad w_k^\lambda = \underset{w \in \mathcal{K}_k}{\operatorname{argmin}} \left\| \begin{bmatrix} Q_A \\ \sqrt{\lambda} Q_L \end{bmatrix} w - \begin{bmatrix} b \\ 0 \end{bmatrix} \right\|_2$$

waarbij $\mathcal{K}_k = (Q_A^T Q_A, Q_A^T b)$ de Krylov deelruimte opgespannen door $Q_A^T Q_A$ en $Q_A^T b$ is.

Het gevolg van stelling 4.7 is dat $w_k^\lambda \in \mathcal{K}_k(Q_A^T Q_A, Q_A^T b)$. Dit gebruiken we om te bepalen in welke ruimte x_k^λ ligt.

Hiervoor gebruiken we de CS ontbinding van $\{Q_A, Q_L\}$ waarbij vooral de ontbinding van Q_A nu van belang is. Deze is gelijk aan $Q_A = P_A C_A W^T$. Voor we verder gaan merken we op dat

$$\mathcal{K}_k(Q_A^T Q_A, Q_A^T b) = \text{span}\{(Q_A^T Q_A)^i Q_A^T b\}_{i=0}^{k-1}$$

Wanneer we de CS ontbinding invullen zien we het volgende:

$$\begin{aligned}\mathcal{K}_k(Q_A^T Q_A, Q_A^T b) &= \text{span}\{(W C_A^T P_A^T P_A C_A W^T)^i W C_A^T P_A^T b\}_{i=0}^{k-1} \\ &= \text{span}\{(W C_A^T C_A W^T)^i W C_A^T P_A^T b\}_{i=0}^{k-1} \\ &= \text{span}\{W (C_A^T C_A)^i C_A^T P_A^T b\}_{i=0}^{k-1}\end{aligned}$$

We hebben gedefinieerd dat $G = R^{-1} W$. Dit geeft de volgende ruimte:

$$\mathcal{K}_k(Q_A^T Q_A, Q_A^T b) = \text{span}\{R G (C_A^T C_A)^i C_A^T P_A^T b\}_{i=0}^{k-1}.$$

We herinneren dat w_k^λ in deze ruimte ligt en dat $w_k = R x_k^\lambda$. Hieruit volgt dat

$$x_k^\lambda \in \text{span}\{G (C_A^T C_A)^i C_A^T P_A^T b\}_{i=0}^{k-1}.$$

We merken nu op dat x_k^λ nu in een ruimte ligt die rechtstreeks opgespannen is door de GSVD van $\{A, L\}$ en dat hij niet afhankelijk is van λ.

We vergelijken dit nu met de oplossingsruimte van LSQR. Dit algoritme geeft een oplossing in de ruimte (12). Wanneer we dit in de GSVD van $\{A, L\}$ uitdrukken vinden we dat :

$$x_{\text{lsqr}} \in \text{span}\{G^{-T}((C_A^T C_A) + \lambda S_L^T S_L)(G^T G)^{-1})^i C_A^T P_A^T b\}_{i=0}^{k-1}.$$

We merken een tweetal dingen op. De eerste is dat de ruimte afhangt van λ. De tweede is dat de term G^{-1} erin voorkomt in plaats van G.

Vooral de laatste opmerking is van belang. De oplossingsruimte van x_{lsqr} is niet direct afhankelijk van de matrix G terwijl die van x_k^λ dat wel is. We zien dus dat wanneer de gsvd van $\{A, L\}$ een goede basis vormt voor de oplossing, dat het samengevoegde bidiagonalisatie algoritme een betere benadering geeft. [5]

5 Numerieke resultaten

We bekijken nu de numerieke resultaten van ons algoritme bij verschillende problemen. Telkens zal een poging gedaan worden de volgende vergelijking op te lossen

$$Ax = b.$$

Voor alle berekeningen is MATLAB gebruikt. De pseudoinverse oplossing wordt verkregen door het oplossen van de vergelijking (4). De oplossing verkregen door ons het algoritme met de samengevoegde bidiagonalisatie geven we aan met SBD. De samengevoegde bidiagonalisatie hebben we zelf geïmplementeerd. Hierbij hebben we de verbeterde versie 6 gebruikt. Voor het berekenen van de vermenigvuldiging met QQ^T is de functie lsqr uit MATLAB gebruikt waarbij we de tolerantie op 10^{-8} hebben gezet en het aantal stappen groot genoeg zodat deze tolerantie bereikt wordt. Voor het vergelijken met het LSQR algoritme benoemd in hoofdstuk 3, hebben we wederom lsqr uit MATLAB gebruikt. Hierbij hebben we de tolerantie op 10^{-8} gezet. Het aantal stappen hangt echter af van k.

We bekijken vooral de relatieve fout die we definiëren als:

$$\frac{||x_{\text{exact}} - x_{\text{reg}}||}{||x_{\text{exact}}||}.$$

5.1 2D beeld vervaging probleem

Dit voorbeeld is hetzelfde voorbeeld als in [5]. Bij dit probleem wordt geprobeerd uit een vervaagd beeld het oorspronkelijke beeld te vinden. Het beeld wordt weergegeven in een $N \times N$ matrix. Het vervaagde beeld is verkregen door een scherp beeld te vervagen door middel van ruis.

Voor dit probleem maken we gebruik van de afbeelding 'rice' uit de Image Processing Toolbox [12]. Voor dit probleem nemen we een 64 stukje uit de originele afbeelding. Het gaat om het gedeelte in de linker hoek. We maken van de matrix een vector door alle kolommen onder elkaar te zetten. De verkregen vector noemen we x_{exact}.

De matrix A is een $N^2 \times N^2$ dubbele Toeplitz matrix. Hij wordt gegeven door $A = (2\pi\sigma^2)^{-1} T \otimes T$. Hierbij is T een $N \times N$ symmetrische band Toeplitz matrix waarbij de eerste rij z gegeven wordt door:

$$z_i = \begin{cases} \exp(-\frac{(i-1)^2}{2\sigma^2}), & \text{als } i \leq \text{band} - 1 \\ 0 & \text{anders} \end{cases}$$

Hierbij is band een half bandbreedte van matrix T en σ een parameter die de breedte van de onderliggende *Gaussian point spread* functie $h(x, y)$ bepaalt

$$h(x, y) = \frac{1}{2\pi\sigma^2} \exp\left(-\frac{x^2 + y^2}{2\sigma^2}\right).$$

We nemen band $= 16$ en $\sigma = 2$. De matrix A kan verkregen worden door de functie blur uit de regularization tools [1].

We bepalen b_{exact} door $b_{\text{exact}} = Ax_{\text{exact}}$. Voor het bepalen van $b = b_{\text{exact}} + e$ kiezen we e zo dat $||e||_2/||b_{\text{exact}}||_2 = 10^{-2}$. Op deze manier kiezen we de ruis op dezelfde manier als het artikel [5].

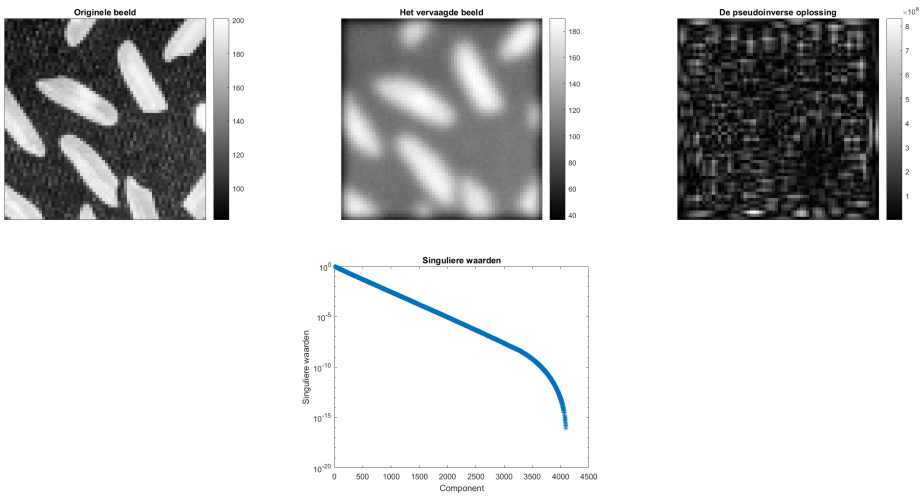
Voor L nemen we de volgende matrix:

$$L = \begin{bmatrix} I \otimes L_1 \\ L_1 \otimes I \end{bmatrix} \quad \text{waarbij} \quad L_1 = \begin{bmatrix} 1 & -1 & & & \\ & 1 & -1 & & \\ & & \ddots & \ddots & \\ & & & 1 & -1 \end{bmatrix}. \tag{25}$$

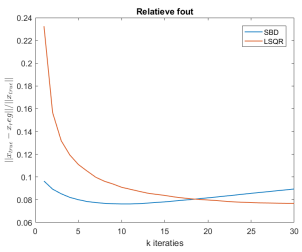
Hierbij is $L_1 \in \mathbb{R}^{63\times 64}$ de discrete benadering is van de eerste afgeleide op een standaard raster zonder aannames over de grensvoorwaarden en geeft $\otimes$ het Kronecker product tussen beide matrices weer.

We hebben het probleem herschaalt zodat $||A||_2 = 1, ||L||_2 = 1$.

In figuur 7 zien we zowel de originele afbeelding, de vervaagde afbeelding en de pseudoinverse oplossing. Er geldt dat deze laatst genoemde compleet niet in de richting komt van de originele afbeelding. In figuur 7 zien we ook de SVD van A. De singuliere waarden dalen op voor de grootste componenten heel snel. Bovendien liggen een aantal singuliere waarden rond machine precisie.



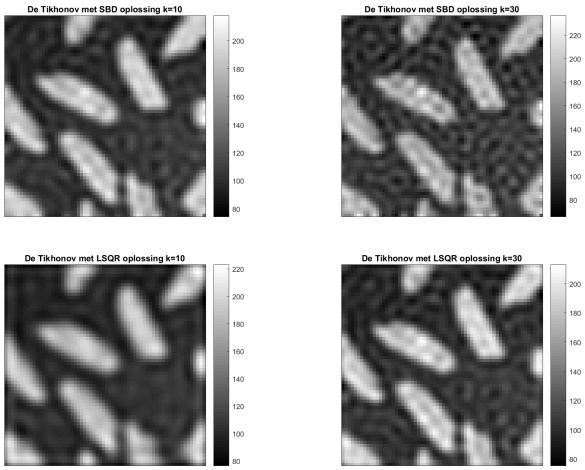
Figuur 7: Links zien we de originele beeld. Het middelste figuur geeft het vervaagde beeld. Het rechter figuur geeft de oplossing van de pseudoinverse methode. Het onderste figuur geeft de singuliere waarden van A.



Figuur 8: De relatieve fout bij $\lambda = 0.001$ en verschillende k.

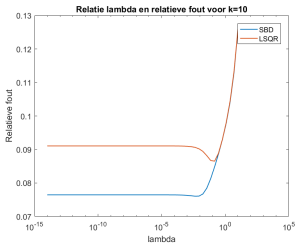
De vergelijking tussen SBD en LSQR is te zien in figuur 8. Wanneer we $\lambda = 0.001$ en k variëren

zien we dat de SBD oplossing eerst beter is. Bij ongeveer $k = 18$ zijn beide fouten gelijk en daarna wordt de LSQR oplossing beter. De SBD oplossing zal steeds slechter worden. De λ die hier gekozen is, ligt dicht bij de optimale λ van de SBD methode aan. We hebben deze verkregen door verschillende λ uit te proberen tussen 10^{-14} en 1.



Figuur 9: In dit figuur zijn verschillende oplossingen weergegeven met $\lambda = 0.001$. Op de bovenste rij zijn de oplossingen weergegeven van het SBD algoritme voor $k = 10$ en $k = 30$. Op de onderste rij zijn de oplossingen weergegeven van het LSQR algoritme voor $k = 10$ en $k = 30$.

In figuur 9 zien we dit effect daadwerkelijk bij de verkregen figuren. Wederom is $\lambda = 0.001$ genomen. We zien de uitkomst van zowel SBD en LSQR bij $k = 10$ en $k = 30$. Voor de SBD oplossing zien we dat voor $k = 10$ de oplossing beter is dan de LSQR oplossing. De benadering wordt echter slechter bij $k = 30$ en de oplossing behorende bij LSQR is nu beter dan de SBD oplossing.



Figuur 10: De relatieve fout als voor vaste k en verschillende λ.

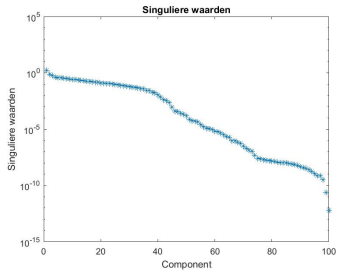
In figuur 10 zien we voor $k = 10$ de fout bij verschillend λ. De fout voor kleinere λ dan de optimale $\lambda = 0.001$ blijft ongeveer hetzelfde blijft voor SBD. Er zit een klein dalletje bij $\lambda = 0.001$ maar het is niet zo dat de fout enorm stijgt als λ kleiner is dan de optimale λ. Wanneer alleen Tikhonov gebruikt wordt is dit vaak wel het geval. Dit effect is ook te zien bij LSQR alleen hier stijgt de relatieve fout eerst iets voordat hij constant wordt bij kleiner wordende λ.

5.2 Helioseismologie

Het volgende probleem is een probleem uit de helioseismologie. De data voor dit probleem behoord bij [2]. We hebben $A \in \mathbb{R}^{212\times 100}$, $x \in \mathbb{R}^{100}$ en $b \in \mathbb{R}^{212}$. Voor onze analyse gebruiken we L gegeven door $L_2 \in \mathbb{R}^{98\times 100}$ waarbij

$$L_2 = \begin{bmatrix} -1 & 2 & -1 & & & & \\ & -1 & 2 & -1 & & & \\ & & -1 & 2 & -1 & & \\ & & & \ddots & \ddots & \ddots & \\ & & & & -1 & 2 & -1 \end{bmatrix}.$$

Hierbij is L_2 de discrete benadering van de tweede afgeleide.

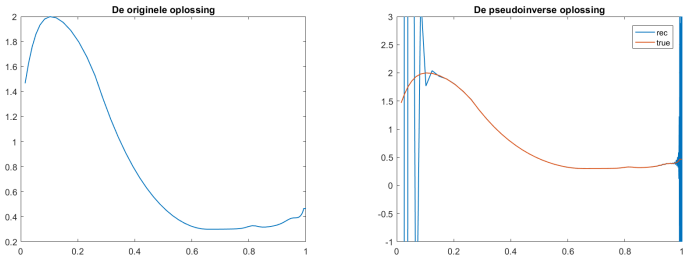


Figuur 11: De singuliere waarden van A.

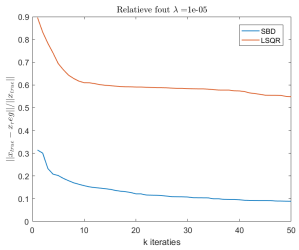
In figuur 11 zien we de de singuliere waarden van A. Deze waarden zijn redelijk mooi verdeeld. Het probleem is niet bijzonder slecht gesteld. Het verschil tussen de kleinste singuliere waarde en de grootste is echter wel erg hoog. Het conditienummer is dan ook gelijk aan $2.9132e12$.

5.2.1 Zonder ruis

We bekijken dit probleem eerst zonder ruis op de data. In figuur 12 zien we x_{exact} afgebeeld als het meest linker grafiek. De grafiek ernaast is de pseudoinverse oplossing. We zien dat de pseudoinverse oplossing geen goede oplossing is.

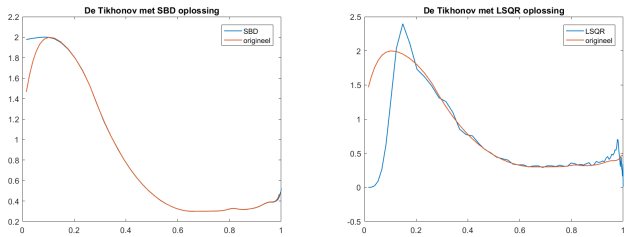


Figuur 12: Links zien we de originele oplossing van het probleem. Rechts zien we de oplossing van het probleem met behulp van de pseudoinverse methode zonder ruis.



Figuur 13: De relatieve fout bij het probleem zonder ruis met k gevarieerd en $\lambda = 0.00001$.

We houden nu $\lambda = 0.00001$ vast en bekijken wat er gebeurt met SBD en LSQR bij oplopende k. We laten zowel SBD als LSQR k keer itereren en dat vergelijken we met elkaar. Het resultaat is te zien in figuur 13. We zien dat de relatieve fout bij SBD kleiner is dan bij LSQR. De fout bij beide wordt steeds kleiner al daalt de SBD wel veel minder snel als k groter wordt.



Figuur 14: De oplossing van SBD (links) en LSQR (rechts) bij $\lambda = 0.00001$ en $k = 50$ bij het probleem zonder ruis.

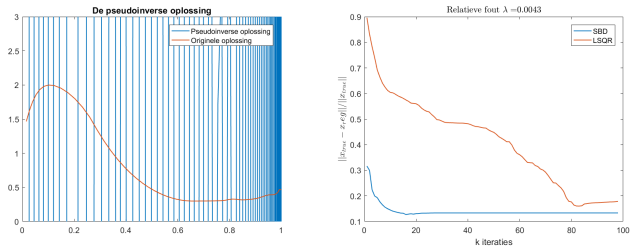
In figuur 14 zien we voor $\lambda = 0.00001$ en $k = 50$ de verkregen oplossing van SBD en LSQR. We zien dat de oplossing bij SBD beter is dan die van LSQR, wat ook te verwachten was na figuur 13.

5.3 Met ruis

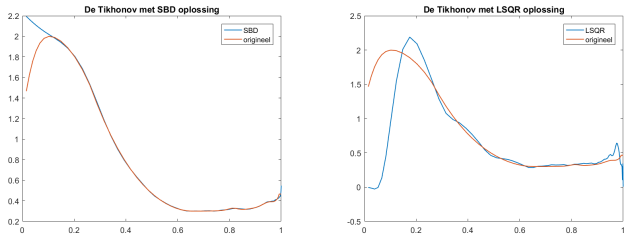
We bekijken de situatie nu met ruis. Voor het bepalen van $b = b_{\text{exact}} + e$ kiezen we e zo dat het vector is met normaal verdeelde getallen met gemiddelde 0 en standaard deviatie 10^{-3} .

We zien in figuur 15 dat de oplossing via de pseudoinverse methode wederom compleet verkeerd is. Door verschillende λ te testen op 7 en naar de oplossing te kijken bekijken vinden we dat we $\lambda = 0.0043$ ongeveer de beste λ is. In figuur 15 zien we wat er gebeurd bij deze λ voor verschillende k bij de twee verschillende methodes. De fout van de oplossing behorende bij SBD is voor kleine k al snel laag en vervolgens blijft de fout hetzelfde. Het LSQR algoritme heeft meer stappen nodig en deze bereikt niet dezelfde convergentie als SBD.

In figuur16 zien we de daadwerkelijke oplossingen voor $\lambda = 0.0043$ en $k = 15$. We zien inderdaad dat SBD beter is dan LSQR na 15 stappen en deze $\lambda = 0.0043$.



Figuur 15: Links geeft de pseudoinverse oplossing weer. Rechts geeft de fout bij $\lambda = 0.0043$ en verschillende k.



Figuur 16: De oplossing van SBD (links) en LSQR (rechts) bij $\lambda = 0.0043$ en $k = 15$ bij het probleem met ruis.

6 Conclusie

In dit artikel hebben we gezien hoe we met behulp van bidiagonalisatie, zowel de Tikhonov regularisatie vergelijking in algemene vorm en in standaard vorm kunnen oplossen voor grote L en A. Bij het oplossen van de standaard vorm hebben we gebruik gemaakt van een beneden Lanczos bidiagonalisatie algoritme. We hebben de methode geanalyseerd en bekeken hoe het gebruikt kan worden bij Tikhonov regularisatie. We zagen dat de methode met zowel Tikhonov regularisatie en projectie een goede oplossing geeft voor meerdere k. De goede keuze van k is minder van belang. Ook hebben we gekeken in welke ruimte de oplossing ligt. Dit bleek de Krylov ruimte te zijn opgespannen door $A^T A$ en $A^T b$.

Voor het oplossen van de algemene vorm hebben we een samengevoegd bidiagonalisatie algoritme bekeken. We hebben gekeken wat het algoritme inhoudt en hoe het gebruikt kan worden om de algemene vorm van Tikhonov regularisatie op te lossen. Verder hebben we bekeken in welke ruimte de oplossing lag en we zagen dat de oplossing met het algoritme met samengevoegde bidiagonalisatie een betere oplossing zal geven dan het LSQR algoritme uitgevoerd op $[A\ L]$ als de GSVD een goede basis is voor de oplossing.

We merken op dat we het vinden van de optimale k om te stoppen met de bidiagonalisatie niet behandeld hebben. Dit probleem heeft nog verder onderzoek nodig. Bovendien geldt dat tijdens het berekenen van de bidiagonalisatie de orthogonaliteit van U_k en V_k verloren gaat. Het effect hiervan hebben we niet bekeken. Verder is er op meerdere plekken in het algoritme een berekening van een ander kleinstekwadraten probleem nodig. We zijn ervan uitgegaan dat dit snel genoeg opgelost kan worden. Toch zal deze stap de benodigde tijd kosten.

7 Referenties

[1] Matlab regularization tools, versie 4.1 (R2016b). P.C. Hansen, DTU Compute, Technical University of Denmark.

[2] T.Sekii en H. Shibahashi P.C. Hansen. The modified truncated svd method for regularization in general form. *SIAM j. Sci. and Stat. Comput.*, 13(5):1142–1150, 1992.

[3] P.C.Hansen. *Discrete Inverse Problems.* SIAM, 2010.

[4] G.J. Golub en C.F van Loan. *Matrix Computations Third Edition.* The Johns Hopkins University Press, 2013.

[5] P.C.Hansen en M.I. Español M.E. Kilmer. A projection-based approach to general-form tikhonov regularization. *SIAM J. Sci. Comput*, 29(1):315–330, 2007.

[6] U.M. Ascher en C. Grief. *A First Course in Numerical Methods.* SIAM, 2011.

[7] A. Ben-Israel en T.N.E. Greville. *Generalized Inverses: Theory and Applications.* Springer, 2003.

[8] C.C. Paige en M.A. Saunders. Lsqr: An algorithm for sparse linear equations and sparse least squares. *ACM Transactions on Mathematical Software (TOMS)*, 8(1):43–71, 1982.

[9] L. Eldén. A weighted pseudoinverse, generalized singular values, and constrained least squares problems. *BIT*, 70(4):487–502, 1982.

[10] G.W.Stewart. Computing the CS decomposition of a partitioned orthonormal matrix. *Numerische Mathematik*, 40(3):297–306, 1982.

[11] C.c Paige en M.A Saunders. Toward a generalized singular value decomposition. *SIAM J. Numer. Anal*, 18(3):398–405, 1981.

[12] Matlab image processing toolbox, versie 9.5 (R2016b). The MathWorks, Natick, MA, USA.