Inhoudsopgave

Hoofdstuk 1

Inleiding

1.1 Context, situering en motivatie

Om in de ingenieurswetenschappen van idee tot product te komen, moet heel wat kennis en vaardigheid samengebracht worden:

- Vooreerst zijn er de basiswetenschappen: thermodynamica, scheikunde, mechanica, fysica, elektromagnetisme, enz. Die kennisdomeinen beschrijven de wereld zoals die in de wetenschap ontdekt is, en geven vaak aanleiding tot wiskundige beschrijvingen van de fenomenen die we rondom ons zien.
- Daarnaast zijn er technologische disciplines: materiaalkunde, chemische proceskunde, machinebouw, bouwkunde, elektrische schakelingen, digitale communicatie, computerwetenschappen, enz. Deze technologische disciplines *gebruiken* de wetenschappelijke kennis om iets te maken, om de wereld vooruit te helpen.

Een belangrijke rol om de basiswetenschappen met de technologie te verbinden is weggelegd voor de numerieke analyse en de toegepaste wiskunde. De numerieke analyse houdt zich bezig met het opstellen, het analyseren en het implementeren van methoden voor het oplossen van wiskundig geformuleerde problemen met een computer. Wanneer we de volledige ketting van taken bekijken, vertrekkend vanuit een welbepaald doel (denk bijvoorbeeld aan het ontwerp van een efficiëntere en/of stillere windturbine, of van een nauwkeurigere scanner voor medische beeldvorming), zijn een aantal deelstappen te onderscheiden:

1. Het beschrijven van het fysisch probleem, de *ontwerpvraag*;
2. Het opstellen van een *wiskundig model* dat ons systeem beschrijft;
3. Het opstellen van een *numeriek model*, waarvan de oplossing op de computer berekend kan worden;
4. Het opstellen van de *numerieke methodes* om die oplossing ook effectief te berekenen;
5. Het implementeren van die numerieke methodes in *software* die bruikbaar is voor ingenieurs in de specifieke toepassing.

In deze cursus gaan we dieper in op stappen 3 en 4, en in zekere mate ook op stap 5 (met name in de oefenzittingen en practica). We behandelen daarbij specifiek de volgende onderwerpen:

- Benaderingstheorie: het opstellen van benaderingen van functies, die ofwel gekend zijn (functiebenadering), ofwel ongekend zijn (en onderliggend aan data die wel beschikbaar zijn).
- Numerieke methodes voor lineaire algebra: de bouwstenen om de benaderingen die we opstellen efficiënt en nauwkeurig te berekenen.

We behandelen deze onderwerpen in een geünificeerd kader dat de onderliggende basisprincipes expliciet maakt en aantoont hoe ogenschijnlijk zeer verschillende onderwerpen toch op dezelfde onderliggende ideeën gebaseerd zijn. De aangehaalde onderwerpen dienen twee doelen. Ten eerste spelen ze een belangrijke rol voor benaderingsproblemen die courant opduiken in zowat alle ingenieursdisciplines. Daarnaast vormen de basisbegrippen uit de benaderingstheorie de hoeksteen voor zowat alle ontwikkelingen in de numerieke analyse en toegepaste wiskunde.

1.2 De componenten van een benaderingsprobleem

Doorheen de cursus zullen we ons steeds de volgende vragen moeten stellen:

1. *Wat* wensen we te benaderen? (Wat is de data, of de te benaderen functie?)
2. *Waarom* wensen we te benaderen? (Wat is het doel, of de taak?)
3. *Waarmee* wensen we te benaderen? (Wat is een geschikte benaderingsstrategie, gegeven het doel en de aard van de data?)
4. Hoe beoordelen we de kwaliteit van een mogelijke benadering? (Hoe kiezen we de beste benadering, wat is het *benaderingscriterium*?)
5. Hoe *berekenen* we de beste benadering? (Welk algoritme, hoe implementeren in bruikbare software?)

We gaan in deze sectie even dieper in op de eerste twee deelvragen. De drie laatste deelvragen zullen het onderwerp vormen van het grootste deel van deze cursus. In de rest van dit inleidende hoofdstuk zetten we eerst het kader (Sectie 1.3), en illustreren we even waarom met het voorbeeld van discrete en continue functiebenadering met veeltermen (Section 1.4).

1.2.1 Wat wensen we te benaderen?

Benaderingsproblemen komen in vele vormen en hoedanigheden voor. De te benaderen data kan *continu* zijn of *discreet*. In de meeste realistische toepassingen is opgemeten data discreet. Denk aan beeldmateriaal (foto's, filmpjes) of tijdsreeksen (elektromagnetische

signalen, sensormetingen). Toch kan het in zo'n geval zinvol zijn om de data te beschouwen als continu (een functie van de tijd, of een functie van ruimte en/of tijd), omdat dit toelaat na te denken over het te benaderen object, los van de resolutie waarmee het opgemeten wordt. Tegelijk zijn er ook situaties waarbij de te benaderen data al inherent continu is. Denk hierbij bijvoorbeeld aan het benaderen van transcendente functies door veeltermen, om ermee te kunnen rekenen op computers. (Dit voorbeeld komt terug in Sectie 1.4.1.)

Een andere manier om data te classificeren is volgens de dimensie van de benaderende functie. Gaat het om een ééndimensionale functie (zoals een tijdsreeks), een tweedimensionale functie (zoals een beeld), een driedimensionale (zoals een filmpje, of de materiaalconstantes in een bepaald volume)? Of vanuit de toepassing. Stelt de data een functie voor, een signaal of een beeld? Al die informatie over wat de data betekent zal een rol spelen bij de beslissingen die we nemen in het benaderingsproces.

Wiskundig gesproken zal de data steeds voorgesteld worden als een vector, die een punt voorstelt in een mogelijk hoogdimensionale vectorruimte. Belangrijk om weten dan is of de data exact is, dan wel meetfouten en/of ruis bevat.

1.2.2 Waarvoor dient de benadering?

Wanneer we grote hoeveelheden data gaan benaderen door een of andere functie, maken we verschillende zaken mogelijk die we niet zomaar met de ruwe data kunnen bekomen. Het is van belang om te weten wat het doel van de benadering is om de keuzes te motiveren die we tijdens het benaderingsproces maken.

Een eerste klasse van taken die we ons kunnen voorstellen is om de grote hoeveelheden data die we beschikbaar hebben compacter voor te stellen, een operatie die we *datacompressie* noemen. Wanneer we een benaderende functie opstellen die slechts van een beperkt aantal coëfficiënten afhankelijk is, kunnen we de nodige opslagcapaciteit heel wat verminderen. Uiteraard is het dan ook belangrijk om de oorspronkelijke data te kunnen terugvinden uit de gecomprimeerde data, hetzij exact, hetzij benaderend. (Deze taak heet *reconstructie*.)

Of we data exact of slechts benaderend willen reconstrueren, heeft ook te maken met de hoeveelheid ruis die op de data aanwezig is. Een belangrijke taak bij benadering is dan ook *ruisverwijdering*. Een opgemeten tijdsreeks zal, door de aanwezige meetruis, vaak veel minder zachtverlopend zijn dan de onderliggende functie die de data genereerde, zie bijvoorbeeld al Sectie 1.4.2 voor een illustratie.

Een volgend soort taak heeft te maken met de *interpretatie* van de data. Hierbij gaat het om het extraheren van informatie uit data, oftewel het geven van betekenis aan de opgemeten gegevens. Twee voorbeelden die hierbij het vermelden waard zijn:

- men kan bijvoorbeeld proberen het surfgedrag van een individuele surfer proberen te benaderen als een lineaire combinatie van een beperkt aantal vooraf gedefinieerde "profielen", om op die manier gericht advertenties te tonen.
- men kan voor een verzameling foto's pogen een benaderende functie te vinden die als invoer een foto heeft, en als uitvoer het antwoord op de vraag of er al dan niet een kat op de foto te zien is.

De taak uit dit laatste voorbeeld heet *classificatie*, en dit is een heel belangrijk soort taak in de wereld van *machine learning*.

Tot slot kunnen we ook proberen de data te *manipuleren*[1]. Ruisverwijdering is al aangehaald als voorbeeld, maar bijvoorbeeld het bekomen van de afgeleide of integraal van de benaderende functie, interpolatie of zelfs extrapolatie (*predictie*) worden mogelijk op basis van functies die een benadering vormen voor de onderliggende data. Denk aan het berekenen van het gemiddelde rendement van je zonnepanelen, het voorspellen van het weer van morgen op basis van de weergegevens van de afgelopen dagen, of het bepalen van het risico van een verzekeringnemer op basis van historische data over bepaalde ziektepatronen.

Het moge duidelijk zijn dat het niet overbodig is dat een ingenieur, of een andere gebruiker van benaderingstechnieken, enig inzicht heeft in de technieken die gebruikt worden om dit soort datagedreven taken uit te voeren en de onderliggende wiskundige veronderstellingen. Dit zijn de thema's waarvoor deze cursus de wiskundige basis legt. Waar deze cursus *niet* op ingaat, zijn de ethische implicaties van het gebruik van benaderingstheorie. Een voorbeeld: wanneer we op basis van grote hoeveelheden data individuele gezondheidsrisico's kunnen bepalen, kan de vraag rijzen in welke mate het principe van een verzekering uitgehold wordt, met name het collectief maken van risico's en het financiële risico spreiden. Bij het beantwoorden van die vraag zijn zowel het ethische kader, als het technische kader (hoe betrouwbaar zijn de gebruikte methodes? wat is meetbaar?) uitermate belangrijk.

1.3 Van data naar functiebenadering

In deze sectie poneren we eerst het probleem van benaderen van data door functies in een heel vage algemene setting, waarna we de algemene stellingen even illustreren met een aantal voorbeelden. We verwijzen ook naar waar die voorbeelden in deze cursus behandeld worden. Het idee van het benaderen van data door een wiskundig model is dat we de data voorstellen als zijnde een monster (steekproef), komende uit een onderliggende (vaak ongekende) functie. We gaan uit van volgende principes voor het opstellen van een wiskundig model:

1. We gebruiken kennis van het probleem om aan te geven wat voor soort functie we zoeken. Met kennis van het probleem bedoelen we zowel de aard van de beschikbare data als de structuur in de data waarnaar we op zoek zijn via de benaderingsmethode.

2. De functies die we beschouwen beelden een invoer af op een uitvoer en leggen de (typisch laag-dimensionale) structuur van het probleem bloot. We kunnen twee functies beschouwen: de *compressie* van de hoogdimensionale data op de laagdimensionale benadering, en de *reconstructie* van de hoogdimensionale data vanuit de laagdimensionale compressie.

Enkele voorbeelden verduidelijken misschien wat de bedoeling is. Elk van de voorbeelden leidt ons tot enkele hoofdstukken van deze cursus.

[1]Let op: we bedoelen hier niet "manipuleren" in de zin van iemands mening ongemerkt beïnvloeden, maar wel in de zin van "bewerkingen uitvoeren op de data".

1.3.1 Benadering van vectoren in eindigdimensionale ruimtes

Beschouw volgende twee problemen:

- We hebben een signaal $b = (b_1, b_2, \ldots, b_m) \in \mathbb{R}^m$, waarvan de componenten de waarde van een opgemeten signaal of tijdsreeks bevatten. We hebben tegelijk een aantal vooropgestelde signalen $a_1, \ldots, a_n \in \mathbb{R}^m$, waarvan we ons afvragen in welke mate ze aanwezig zijn in onze tijdsreeks.
- We hebben de score die een individu heeft gegeven aan een hele rij door ons voorgestelde films verzameld in een vector $b = (b_1, b_2, \ldots, b_m) \in \mathbb{R}^m$. We hebben tegelijk een aantal referentiepersonen uit verschillende doelgroepen $a_1, \ldots, a_n \in \mathbb{R}^m$, en we vragen ons af op welke manier het individu voorgesteld door de vector b samengesteld is uit de verschillende referentiepersonen.

Beide bovenstaande voorbeelden kunnen op dezelfde manier abstract beschreven worden als het zoeken van de beste benadering van de vector b in de deelruimte $\mathcal{D} \subseteq \mathbb{R}^m$, opgespannen door vectoren $a_1, \ldots, a_n$, in formule

$$\mathcal{D} =< a_1, \ldots, a_n > .$$

We veronderstellen $m > n$ en bestuderen de volgende vraag. Zoek de beste benadering voor een vector $b \in \mathbb{R}^m$ in de ruimte $\mathcal{D}$. Het resultaat wordt een benadering $x = (x_1, \ldots, x_n)$, die als volgt geïnterpreteerd kan worden:

$$b \approx \sum_{i=1}^{n} x_i \cdot a_i.$$

De vector x bevat dus de (laagdimensionale) voorstelling van de beste benadering van de (hoogdimensionale) vector b in de deelruimte $\mathcal{D}$. In Hoofdstuk 2 beschouwen we dit probleem in meer detail. We zullen dan bekijken in welke zin we de "beste" benadering kunnen definiëren en wat goede manieren zijn om die te berekenen.

1.3.2 Lineaire benadering van functies

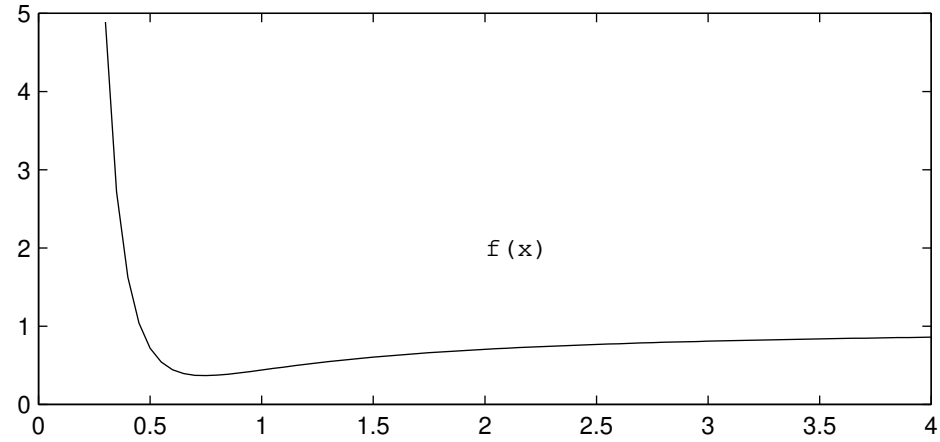
Een belangrijk probleem in de benaderingstheorie is de benadering van een gegeven functie $f(x)$ met een eindig aantal basisfuncties. Dit is van belang bij heel wat numerieke methodes, zoals het benaderen van de oplossingen van differentiaalvergelijkingen.

Vaak gaat men lineaire benaderingen nemen van de vorm

$$y_n(x) = \sum_{k=0}^{n} a_k \phi_k(x) \ . \tag{1.1}$$

waarin de $\phi_k(x)$ een stel gegeven lineair onafhankelijke functies zijn.

De keuze voor specifieke basisfuncties $\phi_k(x)$ kan ingegeven zijn door verschillende redenen. Een populaire keuze zijn veeltermen, zoals de monomialen $\phi_k(x) = x^k$. In Sectie 1.4 gaan we even dieper op die keuze in, en die keuze vormt ook het onderwerp van Hoofdstuk 4.



Figuur 1.1: Een functie met een niet-veeltermachtig verloop.

Een *eerste reden* kan de vorm van de gegeven kromme zijn. Zo zal men periodieke functies best benaderen door een lineaire samenstelling van de functies

$$1, \cos(\omega x), \cos(2\omega x), \ldots, \sin(\omega x), \sin(2\omega x), \ldots \quad \text{met} \;\; \omega = 2\pi/T \;, \tag{1.2}$$

waarbij T de periode voorstelt. Dit noemt men trigonometrische benaderingen. Ze kunnen bijvoorbeeld gevonden worden door de Fourier-reeks van de functie op te stellen en deze af te breken.

Een transcendente of experimenteel opgenomen kromme met een vorm zoals op figuur 1.1 zal men best benaderen door een lineaire samenstelling van negatieve machten van x, omdat de kromme sterk gelijkt op de karakteristieke krommen van de familie van de functies $y(x) = a + \frac{b}{x} + \frac{c}{x^2}$. De benadering zal dan van de volgende vorm zijn

$$y_n(x) = a_0 + a_1 x^{-1} + a_2 x^{-2} + \ldots + a_n x^{-n} \;. \tag{1.3}$$

Ze zal met een kleiner aantal termen dan de veeltermbenadering een even goede nauwkeurigheid opleveren. Soms zal men benaderen met halve machten van x:

$$y(x) = a_0 + a_1\sqrt{x} + a_2 x + a_3 x\sqrt{x} + a_4 x^2 + \ldots \tag{1.4}$$

Ook splinebenaderingen, waarbij het benaderingsinterval wordt opgesplitst in een aantal deelintervallen en een veeltermvoorstelling wordt gezocht op elk deelinterval, zijn geschikt wanneer de te benaderen functie zich niet veeltermachtig gedraagt. Splinebenadering wordt behandelt in Hoofdstuk 5.

Welke basisfuncties men zal verkiezen, is meestal een kwestie van ondervinding.

Een *tweede reden* om andere benaderingen te nemen, is meer van fysische aard. Veronderstel dat men experimenteel het verval meet van een radioactief mengsel. Vervalkrommen

vertonen een exponentieel karakter, vanwege de fysische wetten die het verschijnsel beheersen. In zo'n geval zal men liever een benadering nemen van de vorm

$$y(x) = \sum_{k=0}^{n} a_k \, e^{-\lambda_k t} \ . \tag{1.5}$$

Als we weten uit welke stoffen het mengsel is samengesteld, dan kennen we de vervalconstanten λ_k en komt het er dus op aan de coëfficiënten a_k zodanig te bepalen dat (1.5) een goede benadering is voor de opgemeten functie. De a_k die zo bekomen worden, geven de beginhoeveelheden aan van de verschillende stoffen in het mengsel. We hebben dan niet alleen een formeel benaderingsvraagstuk opgelost; we hebben ook een interpretatie van het fysisch experiment gevonden.

Een *derde reden* om andere basisfuncties te nemen dan de machten van x, is eerder van numerieke aard. Bij het oplossen van het minimalisatieprobleem zullen doorgaans lineaire stelsels moeten worden opgelost. Deze stelsels kunnen voor bepaalde basisfuncties bijzonder slecht geconditioneerd zijn. De afrondingsfouten kunnen zich soms heel sterk voortplanten, waardoor heel veel beduidende cijfers aan nauwkeurigheid verloren gaan.

1.3.3 Niet-lineaire benaderingen

Benaderingen die niet van de vorm (1.1) zijn, waarbij de $\phi_k(x)$ op voorhand gegeven functies zijn, noemt men *niet-lineair*. Als bijvoorbeeld in (1.5) de λ_k niet op voorhand gekend zijn, dan heeft men een niet-lineair benaderingsvraagstuk. Deze vraagstukken komen tamelijk veel voor bij het interpreteren van fysische experimenten. Ze zijn echter dikwijls slecht geconditioneerd, d.w.z. zeer gevoelig aan fouten op de gegevens.
Het best gekend zijn rationale benaderingen

$$y_{m,n}(x) = \frac{a_0 + a_1 x + \ldots + a_m x^m}{b_0 + b_1 x + \ldots + b_n x^n} \ . \tag{1.6}$$

Zij zijn zeer geschikt voor het benaderen in het complexe vlak, of voor het benaderen van reële functies met singuliere punten in het benaderingsinterval, of in de omgeving ervan.
Ook diepe neurale netwerken zijn niet-lineaire benaderingstechnieken, en komen aan bod in het derde deel van deze cursus.

1.4 Veeltermbenadering als illustratief voorbeeld

1.4.1 Benaderen van een continue functie

Nu we al even zijn ingegaan op de vraag *wat* we willen benaderen, *waarom*, en *waarmee*, bekijken we even een specifieke situatie om te illustreren waarom de vraag naar een benaderingscriterium en -algoritme belang hebben.

Om de zaken wat concreter te maken, bekijken we een aantal veeltermbenaderingen van de functie e^x over het interval $[-1, 1]$. Er komen in de discussie aspecten aan bod die ook meer in het algemeen geldig zijn, en die we in het vervolg van de cursus verder zullen uitdiepen.

1.4.1.1 Machtreeksontwikkeling

Om de exponentiële functie te berekenen in het interval $[-1, 1]$ kan men haar Maclaurinreeks opstellen en deze afbreken. Laten we dit bijvoorbeeld doen na de vijfde term. We bekomen dan een veelterm van de vierde graad:

$$\begin{aligned} y_4(x) &= 1 + x + \frac{1}{2}x^2 + \frac{1}{3!}x^3 + \frac{1}{4!}x^4 \\ &\simeq 1 + x + 0.5\,x^2 + 0.16666666\,x^3 + 0.04166666\,x^4 \ . \end{aligned}$$

In de onmiddellijke omgeving van de oorsprong zal deze benadering wellicht voldoen: voor de waarde $x = 0.1$ geeft de benadering een fout die kleiner is dan 10^{-7}. Voor $x = 1$ is de fout echter ongeveer gelijk aan 10^{-2}, wat voor sommige toepassingen veel te groot is. De foutenkromme

$$r(x) = e^x - y_4(x)$$

werd geschetst in figuur 1.2. Men bemerkt het volgende:

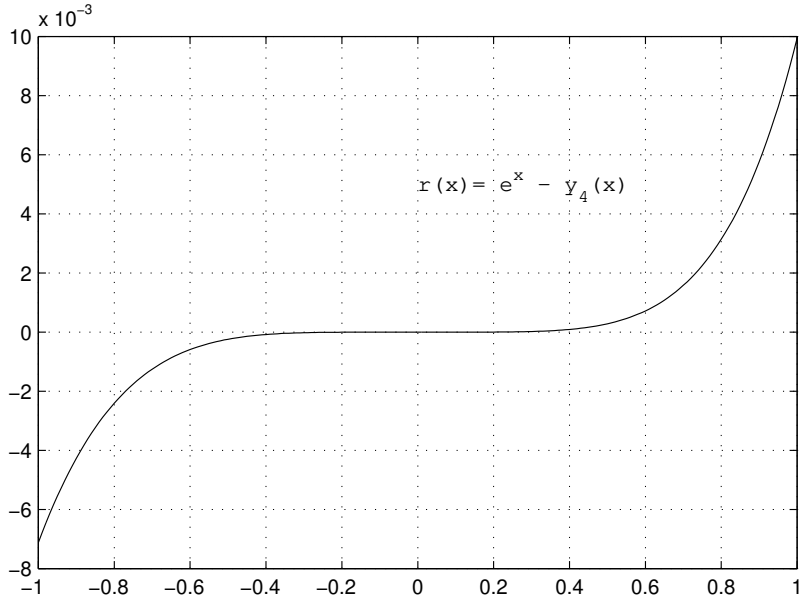
- de fout is nul in de oorsprong en stijgt monotoon in absolute waarde als x stijgt in absolute waarde;
- de foutenkromme vertoont weinig variatie in de omgeving van de oorsprong; dit komt doordat de eerste tot de vierde afgeleiden van de benadering gelijk zijn aan deze van de functie.
- de fout stijgt sneller naarmate men zich verder van de oorsprong verwijdert;
- de maximale waarde van de fout treedt op aan de intervalgrens $x = 1$.

Dit verloop is karakteristiek voor alle afgebroken Maclaurin- of Taylor-ontwikkelingen: in het punt waarrond de functie ontwikkeld wordt, is de waarde van de benadering gelijk aan die van de functie, en zijn ook de eerste afgeleiden gelijk. In de omgeving van dat punt zal de benadering dus zeer goed zijn. Aan de grenzen zal de benadering het slechtst zijn. Het nadeel van deze situatie is dat men dikwijls verplicht is een hoge graad van benadering te nemen om de fouten aan het uiteinde van het interval binnen redelijke grenzen te houden. De benadering in het midden van het interval is dan vaak veel nauwkeuriger dan gevraagd werd. Meestal wenst men een fout die meer gelijkmatig over heel het interval verspreid is.

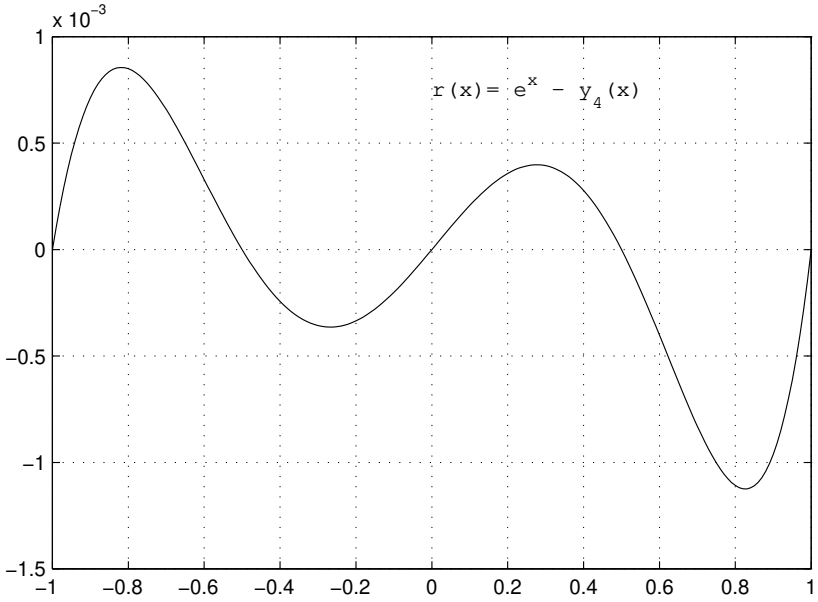
We vermelden nog dat het opstellen van een Taylor-benadering voornamelijk symbolisch rekenwerk vereist. Wanneer de transcendente functie een moeilijk voorschrift heeft, dan gebruikt men hiervoor best een formulemanipulator zoals Mathematica of Maple.

1.4.1.2 Veelterminterpolatie

Voor het opstellen van een benadering voor de exponentiële functie zou men kunnen vertrekken van een tabel met de waarde van e^x voor een aantal discrete abscissen $x_1, \ldots, x_N$.



Figuur 1.2: Fout van de Maclaurin-benadering van graad 4 voor e^x.



Figuur 1.3: Fout van een interpolerende benadering van graad 4 voor e^x.

Laten we de overeenkomende functiewaarden aanduiden met $f_1, \ldots, f_N$. De interpolerende veelterm van graad $N-1$ is dan de veelterm $y_{N-1}(x)$ die in de punten x_i de waarde f_i aanneemt, d.w.z. de veelterm $y_{N-1}(x)$ waarvoor geldt dat:

$$y_{N-1}(x_i) = f_i \quad \text{voor} \quad i = 1, \ldots, N \ . \tag{1.7}$$

Men bewijst dat de interpolerende veelterm bestaat en enig is. Er bestaan verschillende vormen voor deze veelterm, bijvoorbeeld met gedeelde differenties of met zogenaamde veeltermen van Lagrange. De interpolatieformule van Lagrange geeft de volgende expliciete uitdrukking voor de interpolerende veelterm:

$$y_{N-1}(x) = \sum_{k=1}^{N} l_k(x)\, f_k \ . \tag{1.8}$$

De basisfuncties $l_k(x)$ zijn de *Lagrange-veeltermen* voor de gegeven abscissen. Zo'n veelterm $l_k(x)$ is gedefinieerd als de veelterm van graad $N-1$ die de waarde 1 aanneemt in x_k en de waarde 0 in de andere punten. Die veelterm wordt gegeven door

$$l_k(x) = \frac{(x-x_1)\cdots(x-x_{k-1})(x-x_{k+1})\cdots(x-x_N)}{(x_k-x_1)\cdots(x_k-x_{k-1})(x_k-x_{k+1})\cdots(x_k-x_N)} \tag{1.9}$$

$$= \frac{\pi(x)}{(x-x_k)\pi'(x_k)} \tag{1.10}$$

met

$$\pi(x) = (x-x_1)\cdots(x-x_{N-1})(x-x_N) \ . \tag{1.11}$$

We selecteren nu een aantal punten waarin we de exponentiële functie evalueren. Om aan te sluiten bij het vorige voorbeeld nemen we vijf equidistante punten in het interval $[-1, 1]$, zoals vermeld in tabel 1.1. We bekomen zodoende een vierdegraadsbenadering voor de exponentiële functie :

$$y_4(x) = 1 + 0.997854\, x + 0.499645\, x^2 + 0.177347\, x^3 + 0.043436\, x^4 \ .$$

i	0	1	2	3	4
x_i	-1	-0.5	0	0.5	1
f_i	0.36787944	0.60653066	1.00000000	1.64872127	2.71828183

Tabel 1.1: Interpolatiepunten en functiewaarden voor de functie e^x.

De foutenkromme werd weergegeven in figuur 1.3. Wanneer men ze vergelijkt met die van de Maclaurin-reeks in figuur 1.2, dan ziet men dat de interpolerende veelterm van dezelfde graad reeds veel betere resultaten geeft, behalve in de onmiddellijke omgeving van de oorsprong. De maximale fout is nu van grootte-orde 0.001 in plaats van 0.01. Wellicht

kan de benadering nog verbeterd worden door een andere keuze van de interpolatiepunten. Er is a priori immers geen enkele reden om te veronderstellen dat equidistante interpolatie optimaal zou zijn (dat is trouwens niet zo). Hoe we de punten dan wel zouden moeten kiezen, is voorlopig niet duidelijk.

Om deze interpolatiemethode te gebruiken, dienen we te beschikken over een techniek om de te benaderen functie nauwkeurig uit te rekenen. De snelheid van die techniek is niet zo belangrijk. In bovenstaand voorbeeld zouden we de reeksontwikkeling van e^x kunnen gebruiken, afgebroken na een voldoende groot aantal termen.

1.4.1.3 Andere benaderingscriteria

We vermoeden dat er betere benaderingen bestaan dan de interpolerende benadering die hierboven werd opgesteld. Wanneer we van de volledige kromme van de functie e^x in het interval $[-1, 1]$ slechts vijf punten uitkiezen, dan laten we immers zeer veel informatie ongebruikt. We zouden een methode willen ontwerpen die alle functiewaarden van heel het interval in rekening brengt.

Wellicht het meest voor de hand liggende criterium is het volgende. We zoeken onder alle mogelijke vierdegraadsveeltermen de veelterm die het minst afwijkt van de exponentiële functie. Meer precies zoeken we de veelterm waarvoor de grootste afwijking $e^x - y_4(x)$ zo klein mogelijk is. Wiskundig wordt dit: bepaal a_0, a_1, a_2, a_3 en a_4 zodat

$$\max_{x\in[-1,1]} \mid e^x - (a_0 + a_1x + a_2x^2 + a_3x^3 + a_4x^4) \mid \quad \text{minimaal is.} \tag{1.12}$$

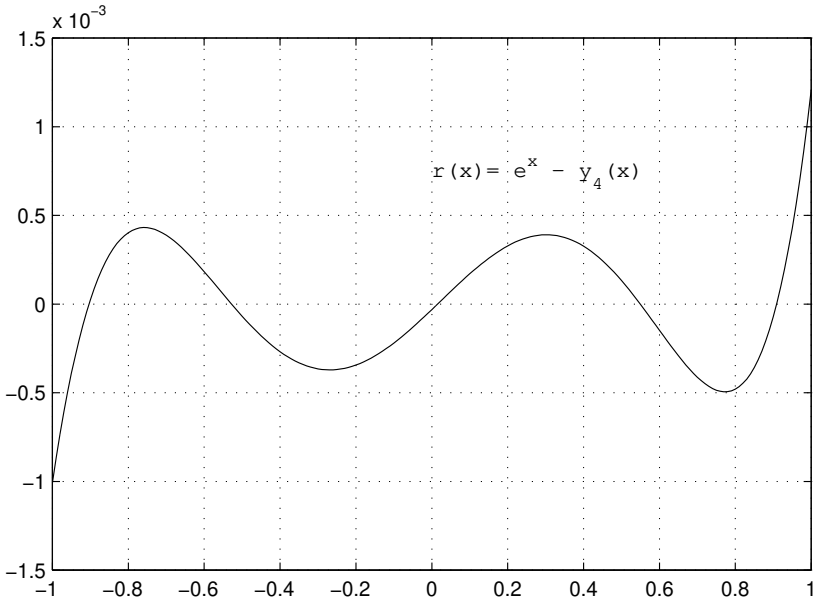
Dit benaderingscriterium noemt men het *minimaxcriterium*. Het geeft aanleiding tot een vrij moeilijk minimalisatieprobleem, in dit geval in de vijfdimensionale ruimte van componenten $(a_0, a_1, a_2, a_3, a_4)$.

Een ander criterium dat eveneens alle functiewaarden van de te benaderen functie over het volledige benaderingsinterval in rekening brengt, is het volgende:

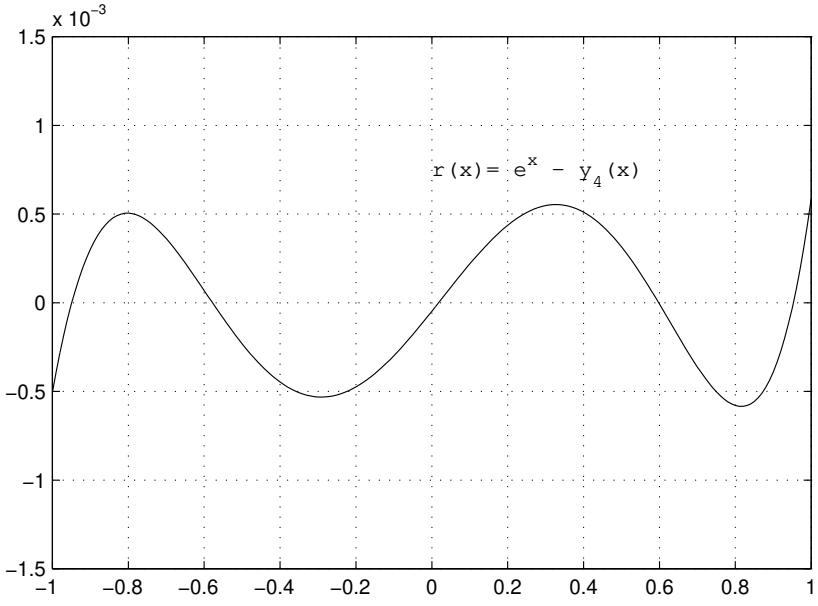
$$\text{minimaliseer} \quad \int_{-1}^{1} w(x) \left(e^x - (a_0 + a_1x + a_2x^2 + a_3x^3 + a_4x^4)\right)^2 \, dx \; . \tag{1.13}$$

Hierin is $w(x)$ een strikt positieve functie, die toelaat aan bepaalde punten of deelgebieden van het benaderingsinterval meer of minder gewicht toe te kennen. Daar waar $w(x)$ een relatief hoge waarde aanneemt, zal de benadering nauwer aansluiten bij de te benaderen functie. Men noemt $w(x)$ de *gewichtsfunctie* en het criterium wordt het *kleinstekwadratencriterium* genoemd. Ook dit criterium geeft aanleiding tot een minimalisatieprobleem. Zoals we verder zullen zien, is het echter veel gemakkelijker op te lossen dan het minimalisatieprobleem geassocieerd aan het minimaxcriterium.
We vermelden alvast de oplossing van het benaderingsprobleem. Wanneer we $w(x) \equiv 1$ nemen, dan verkrijgen we de volgende benadering:

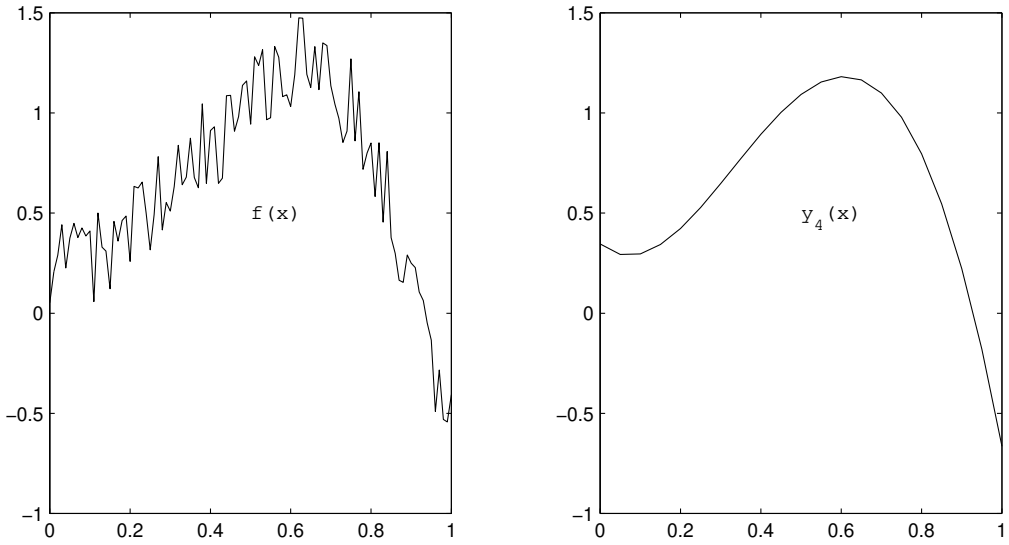
$$y_4(x) = 1.000031 + 0.997955\,x + 0.499352\,x^2 + 0.176139\,x^3 + 0.043597\,x^4 \; . \tag{1.14}$$



Figuur 1.4: Fout van de kleinste-kwadratenbenadering van graad 4 voor e^x met $w(x) \equiv 1$.



Figuur 1.5: Fout van de kleinste-kwadr.-benadering van graad 4 voor e^x met $w(x) = \frac{1}{\sqrt{1-x^2}}$.



Figuur 1.6: Een sterk door ruis verstoord signaal en een vierdegraadsbenadering.

De fout van deze benadering werd getekend in figuur 4.2. Merk op dat de fout nog steeds het grootst is aan de randen van het benaderingsinterval.

Ter vergelijking hebben we ook eens de benadering opgesteld voor de gewichtsfunctie

$$w(x) = \frac{1}{\sqrt{1-x^2}} \ . \tag{1.15}$$

Deze gewichtsfunctie kent een zeer groot gewicht toe aan de randen van het interval. Men noemt ze de *Chebyshev-gewichtsfunctie*. De bekomen benadering luidt

$$y_4(x) = 1.000048 + 0.997308\,x + 0.499197\,x^2 + 0.177347\,x^3 + 0.043794\,x^4 \ . \tag{1.16}$$

Bemerk dat de coëfficiënten in (1.16) slechts weinig verschillen van die in (4.26). Toch is de nauwkeurigheid heel wat beter. De maximale fout is ongeveer 0.00059 i.p.v. 0.0012. De foutenkromme werd getekend in figuur 4.3. De fout is zeer gelijkmatig verspreid over het interval.

1.4.2 Benaderen van een discrete functie

In vele praktische problemen heeft men functies gegeven in tabelvorm, met discrete abscissen $x_1, \ldots, x_N$ en bijbehorende ordinaten $f_1, \ldots, f_N$. Een dergelijke tabel zou functiewaarden kunnen bevatten afkomstig van de numerieke evaluatie van een continue functie. Het kan zijn dat die functie slechts impliciet gegeven is, bijv. als oplossing van een differentiaalvergelijking. De functiewaarden werden dan wellicht gevonden via een numerieke methode voor het oplossen van dergelijke vergelijkingen. De functiewaarden zullen in zo'n geval vrij nauwkeurig zijn. De tabel kan echter ook afkomstig zijn van een experimenteel

opgenomen kromme of van een bemonsterd analoog signaal. De meetwaarden kunnen dan onderhevig zijn aan meetfouten, en het signaal kan verstoord zijn door ruis. Voor een voorbeeld verwijzen we naar figuur 1.6.
Wanneer een tabel met N punten gegeven is, zou men een interpolerende veelterm van graad $N-1$ door de punten kunnen leggen. Er zijn echter goede redenen om dit niet te doen, maar liever een benadering van een lagere graad op te stellen.

- De interpolerende veelterm gaat in een dergelijk geval de meetfouten op de experimentele gegevens of de toevallige schommelingen veroorzaakt door de ruis op het discrete signaal exact volgen. Dat kan zeker niet de bedoeling zijn. Doorgaans wenst men immers een benadering op te stellen met een zacht verloop, zodat meetfouten en ruis 'weggefilterd' worden. Men spreekt in dit verband van het *vereffenen van krommen*, in het Engels *curve fitting*.

- Zelfs al zijn er geen grove meetfouten en is ruis te verwaarlozen, dan nog zijn de gegevens van de tabel wellicht niet nauwkeurig genoeg om een interpolatie van hoge graad te verrechtvaardigen. Stel dat de gegevens bijvoorbeeld vermeld zijn tot op 5 juiste cijfers, dan heeft het geen zin een benadering op te stellen die een hogere nauwkeurigheid zou bieden.

- Vaak is het de bedoeling bij het opstellen van een benadering om aan gegevensreductie of -compressie te doen. Men wil een uitgebreide tabel gaan benaderen met een functie waarvan het voorschrift zeer compact is. Men verlangt dan dat het aantal parameters of coëfficiënten in de benadering heel wat kleiner is dan het aantal gegevens die voor het opstellen van de benadering gebruikt werden.

- Zoals we verder nog zullen zien, geeft het opstellen en evalueren van veeltermen van zeer hoge graad vaak aanleiding tot numerieke problemen. Omwille van de eindige getallenvoorstelling en de zeer sterke voortplanting van afrondingsfouten, heeft het eindresultaat doorgaans slechts een beperkt aantal juiste beduidende cijfers.

Wanneer men in een grote en voldoende nauwkeurige tabel zou willen interpoleren, dan zou men het interval kunnen onderverdelen in kleinere deelintervallen waarop men interpolatie van een lagere graad toepast, of men zou een aantal punten kunnen uitkiezen uit de gegeven tabel. De eerste werkwijze veronderstelt het opstellen en bewaren van verschillende veeltermen, die op een gepaste manier aan elkaar dienen gekoppeld te zijn. De tweede werkwijze laat een gedeelte van de gegeven informatie ongebruikt; daarenboven is het niet meteen duidelijk welke punten het best als interpolatiepunten gekozen moeten worden.

Men zal liever trachten op een of andere manier alle punten van het interval bij het opstellen van de benaderende veelterm te betrekken. Dit is bijvoorbeeld het geval bij de zogenaamde *discrete kleinste-kwadratenbenadering*. Het criterium is dan analoog aan (1.13), maar de integraal wordt er vervangen door een discrete som over alle meetpunten:

$$\text{minimaliseer} \quad \sum_{i=1}^{N} w_i \left(f_i - (a_0 + a_1 x_i + a_2 x_i^2 + a_3 x_i^3 + a_4 x_i^4) \right)^2 \ . \tag{1.17}$$

Men gebruikt opnieuw een gewichtsfunctie, in dit geval een rij van N getalwaarden w_i. Dat laat toe aan bepaalde koppels (x_i, f_i) een hoger belang toe te kennen dan aan andere. Een vierdegraadsbenadering voor de discrete functie van figuur 1.6, opgesteld met het criterium (1.17) voor $w_i = 1$, vind je aan de rechterkant van figuur 1.6.

Indien de discrete gegevens voldoende nauwkeurig zijn, dan zou men er ook aan kunnen denken om de discrete variante van (1.12) te nemen als criterium. Men spreekt van het *discrete minimaxcriterium* wanneer men vereist dat de benadering die veelterm is die, onder alle veeltermen van een bepaalde graad (of lager), de uitdrukking

$$\max_{i=1,..,N} \mid f_i - (a_0 + a_1 x_i + a_2 x_i^2 + a_3 x_i^3 + a_4 x_i^4) \mid$$

minimaal maakt. Dat is opnieuw een minimalisatieprobleem.

1.5 De beste benadering als optimalisatieprobleem

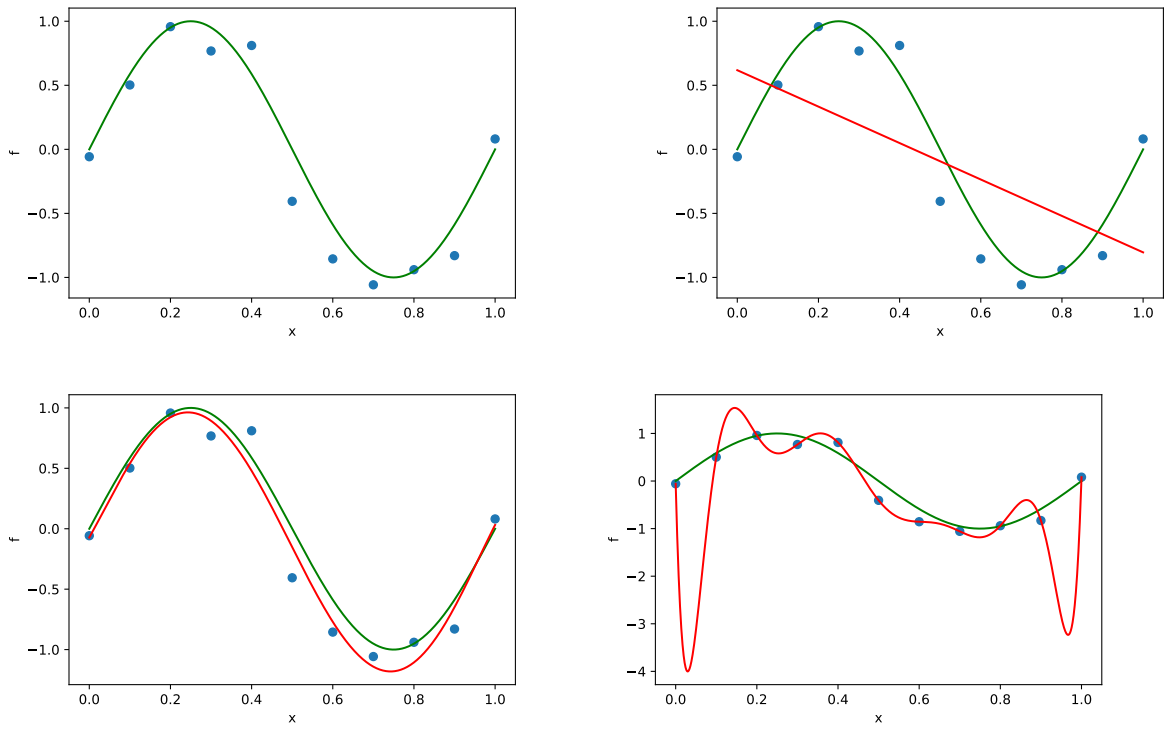
In de vorige secties bleek dat we het vinden van de beste benadering van een functie kunnen beschouwen als een optimalisatieprobleem: vind, binnen de verzameling mogelijke functies, diegene die de norm van het verschil tussen de data en de benaderende functiewaarde minimaal maakt. Wanneer we de gegeven data noteren als $(x_i, f_i)_{i=1}^N$ en de basis voor de benadering $\{\phi_k(x)\}_{k=1}^n$, wordt de vraag: vind de coëfficiënten $\{a_k\}_{k=1}^n$ die voldoen aan de eis:

$$\arg\min_{\{a_k\}} \left\| \boldsymbol{f} - \sum_{k=1}^{n} a_k \boldsymbol{\phi}_k \right\|, \tag{1.18}$$

met $\boldsymbol{f} = (f_i)_{i=1}^N, \boldsymbol{\phi}_k = (\phi_k(x_i))_{i=1}^N \in \mathbb{R}^N$. De norm $\| \cdot \|$ is in bovenstaande vergelijking niet gespecifieerd. Vele normen bestaan, en de keuze voor een bepaalde norm hangt sterk af van het doel van de benadering. In Hoofdstuk 3 zullen we dit soort problemen in een abstracte setting behandelen, waarbij we het ook zullen hebben over de eigenschappen waaraan normen moeten voldoen.

1.5.1 Dimensionaliteit van data en van model

De kwaliteit van een beste benadering hangt ook af van de gekozen basisfuncties, en van de dimensie van de benaderingsruimte (het aantal te bepalen coëfficiënten n) ten opzichte van de hoeveelheid data (het aantal datapunten N) en de hoeveelheid ruis op de data. Laat ons deze beschouwing even concreet maken met een nieuw voorbeeld. Veronderstel dat we op zoek zijn naar de functie $f(x) = \sin(2\pi x)$ op het interval $[0, 1]$. De data die we hebben zijn metingen op de punten $x_i = i\Delta x$, waarbij $\Delta x = 1 \cdot 10^{-1}$, voor $i = 0, \ldots, 100$. Elk meetpunt bevat een meetfout, en we hebben daardoor als data $f_i = f(x_i) + \xi_i$, waarbij ξ_i onafhankelijke normaal verdeelde ruis is met gemiddelde 0 en standaarddeviatie $2 \cdot 10^{-1}$. De data, alsook de onderliggende functie waarvan de data afkomstig is, zijn te zien in Figuur 1.7 (linksboven).



Figuur 1.7: Linksboven: onderliggende gezochte functie $f(x) = \sin(2\pi x)$ (groene lijn), samen met tien ruizige evaluaties van die functie (blauwe punten). Rechtsboven: zelfde gezochte functie en data, samen met een lineaire beste benadering. Linksonder: zelfde gezochte functie en data, samen met de beste benadering in de ruimte van veeltermen van vijfde graad. Rechtsonder: zelfde gezochte functie en data, samen met de beste benadering in de ruimte van veeltermen van graad 10 (tegelijk een interpolerende veelterm).

Als we op zoek gaan naar een goede benadering, dan willen we eigenlijk de functie $f(x) = \sin(2\pi x)$ terugvinden, want dat is de functie die de data genereerde. Alleen weten we niet welke functie dit is, of zelfs welk soort functie. Wanneer we nu besluiten om een veeltermbenadering te zoeken, gebaseerd op de opgemeten data, moeten we zelf nog een gepaste graad kiezen. Twee extreme situaties zijn:

- Het kiezen van een zeer hoge graad, bijvoorbeeld een benadering van graad 10, zie Figuur 1.7 (rechtsonder). Met zo'n benadering zullen we de data perfect kunnen reconstrueren. (Met 11 meetpunten is de beste benadering ook een interpolerende veelterm door de datapunten. Helaas biedt die benadering geen enkele nuttige informatie over de onderliggende functie *tussen* de meetpunten. De onderliggende functie is veel zachtverlopender dan de interpolerende veelterm kan zijn. Wanneer dit probleem zich voordoet, spreken we van *overfitting*.

- Het kiezen van een benadering van lagere graad kan ruis verwijderen, en wordt dus gebruikt om de benadering zachtverlopend te maken. (Dit heet *smoothing*.) We kunnen daar echter ook te ver in gaan. Neem bijvoorbeeld een lineaire benandering, zie Figuur 1.7 (rechtsboven). Met zo'n benadering zullen we, naast de ruis, ook veel informatie over de onderliggende functie weggooien. We maken onze functie wel zeer zachtverlopend. Dit probleem noemen we *oversmoothing*.

Wanneer we een goede balans gevonden hebben, kunnen we wel een goede benadering bekomen. In dit voorbeeld is een veelterm van graad 5 geschikt, zie Figuur 1.7 (links onder). Hoe goed de beste benadering is, zal dus ook afhangen van de hoeveelheid beschikbare data, de ruis op die data, en de functies waarmee we benaderen.

1.5.2 Het belang van regularisatie

In de praktijk is de functie die de data genereerde ongekend, en is het dus niet eenvoudig om na te gaan of we met oversmoothing of overfitting te maken hebben. We willen het model zo eenvoudig mogelijk houden, maar niet eenvoudiger dan mogelijk. In het voorbeeld hierboven: we willen de graad van de veelterm zo laag mogelijk, maar niet lager. Bovendien zijn lage-graadsbenaderingen makkelijker te interpreteren: het is makkelijker om de fysische betekenis van een eerste afgeleide aan te voelen dan van een vijfde afgeleide.

Tegelijk willen we wel de mogelijkheid hebben om problemen te benaderen met een aantal parameters dat in de buurt komt van het aantal meetpunten (zolang onze benadering maar niet overfit of hevig begint te oscilleren tussen de datapunten). Bij neurale netwerken is het probleem zelfs nog erger: heel vaak is het aantal te bepalen parameters daar zelfs (veel) groter dan het aantal datapunten! In zo'n geval is een beste benadering zelfs niet goed gedefinieerd. Het is mogelijk dat heel wat parametercombinaties de data exact reproduceren, maar dat slechts weinig benaderingen informatief zijn over de functie tussen de datapunten in…

Al deze problemen worden gevat onder de noemer "slecht gestelde problemen", en ze worden aangepakt door regularisatie. De klassieke technieken voor regularisatie bestaan er in om

een balans te vinden tussen de benadering van de data aan de ene kant en de complexiteit van het model aan de andere kant. Een typisch optimalisatieprobleem dat zo ontstaat, heeft vaak de vorm

$$\arg\min_{\{a_k\}} \left\| \boldsymbol{f} - \sum_{k=1}^{n} a_k \boldsymbol{\phi}_k \right\|_{\alpha} + \|\boldsymbol{a}\|_{\beta}\,, \tag{1.19}$$

waarin we de vector van ongekende coëfficiënten noteren als $\boldsymbol{a} = (a_k)_{k=1}^{n}$ en met de subscripts α en β aangeven dat de twee gebruikte normen niet noodzakelijk dezelfde zijn. Regularisatie zal aan bod komen in Hoofdstukken 7 en 10.

1.6 Besluit

Als samenvatting mogen we zeggen dat een benaderingsprobleem steeds de volgende vier componenten omvat:

- een *te benaderen functie*. Het begrip functie is hier heel algemeen. Het kan een klassieke continue functie zijn, maar ook een discreet signaal of een tweedimensionaal beeld (bijv. een ingescande foto of document). Vaak is de 'ware' functie – die functie die men echt wil benaderen – zelfs niet gekend. Dat is bijvoorbeeld zo in het geval van beeldrestauratie en bij methoden voor detectie en verwijdering van meetfouten.

- een *klasse van benaderingsfuncties*, waaruit op een of andere wijze een goede benadering zal worden geselecteerd. De klasse kan worden gekozen omwille van de eenvoud van verwerking en manipulatie (bijv. veeltermen), omwille van fysische redenen (bijv. trigonometrische of exponentiële functies), omwille van numerieke redenen (orthogonale functies),... Bij de compressie van beelden en signalen zal men dan weer een voorstelling kiezen waarbij het bekomen van een zo compact mogelijke representatie de keuze bepaalt (bijv. wavelets).

- een *benaderingscriterium*, dat moet aangeven welke 'kandidaat' in de klasse van benaderingsfuncties de meest geschikte is. Of men kiest voor een reeksontwikkeling, een interpolant, een kleinste-kwadraten- of minimaxbenadering, of nog een ander soort benadering, hangt meestal af van de eisen die men stelt aan de benadering, van wat men uiteindelijk met de benadering wenst te doen en wellicht ook van de inspanning die men wenst te besteden aan het opstellen van de benadering.

- een *benaderingsalgoritme*, dat aangeeft hoe de beste kandidaat volgens het vooropgestelde criterium kan worden gevonden. De implementatie van het algoritme zal in vele gevallen aanleiding geven tot een *computerprogramma* dat op een numerieke of symbolische wijze de benadering genereert.

In het vervolg van deze cursus zullen we op al deze aspecten dieper ingaan.

Deel I

Lineaire benaderingsproblemen

Hoofdstuk 2

Benadering van vectoren

In dit hoofdstuk behandelen we benaderingen van vectoren in deelruimten van de m-dimensionale vectorruimte $\mathbb{C}^m$ over de complexe getallen. In Sectie 2.1 illustreren we het belang van het gebruik van orthogonale basissen. In Sectie 2.3 bespreken we verscheidene algoritmen die toelaten om een basis van een deelruimte te vervangen door een orthonormale basis. In Sectie 2.4 beschouwen we tot slot het probleem van het bepalen van de *beste* benadering van een vector in een deelruimte. Centraal in de methodologie is het concept van orthogonale projectie, waar we dieper op ingaan in Sectie 2.2.

In de ruimte $\mathbb{C}^m$ kunnen we het volgende inwendig product definiëren,

$$(x, y) = x^* y = \sum_{i=1}^{m} \bar{x}_i y_i, \quad x, y \in \mathbb{C}^m,$$

met x^* de complex toegevoegd getransponeerde van x. De geassocieerde norm is de 2-norm,

$$\|x\|_2 = \sqrt{(x, x)} = \sqrt{x^* x} = \sqrt{\sum_{i=1}^{m} |x_i|^2}, \quad x \in \mathbb{C}^m,$$

waarbij $|\cdot|$ slaat op de modulus. Een overzicht van terminologie, definities en eigenschappen van complexe vectoren en matrices geven we in Appendix A.

De resultaten en afleidingen uit het hoofdstuk zijn rechtstreeks toepasbaar op de ingebedde Euclidische ruimte[1] $\mathbb{R}^m$. De factorisaties en algoritmen die we zullen tegenkomen zijn bijvoorbeeld zodanig dat als de invoer bestaat uit reële vectoren en matrices, ditzelfde geldt voor de uitvoer, en enkel bewerkingen met reële getallen nodig zijn. Bij de meetkundige interpretaties en illustraties zullen we in wat volgt ook steeds met reële matrices werken.

[1] Een Euclidische ruimte is een eindig-dimensionale vectorruimte over het veld van de reële getallen.

2.1 Orthogonale en scheve basissen

Beschouw een deel(vector)ruimte $\mathcal{D} \subseteq \mathbb{C}^m$. Een stel vectoren $\{a_1, \ldots, a_n\}$, met $n \leq m$ vormen een basis van $\mathcal{D}$ als deze lineair onafhankelijk zijn en bovendien

$$< a_1, \ldots, a_n >:= \left\{ \sum_{i=1}^{n} c_i a_i : \ c_i \in \mathbb{C}, \ i = 1, \ldots, n \right\} = \mathcal{D}. \tag{2.1}$$

We noemen n de dimensie van de deelruimte van $\mathcal{D}$. Wanneer we matrix $A \in \mathbb{C}^{m \times n}$ definiëren als $A = [a_1 \cdots a_n]$, dan kunnen we voorwaarde (2.1) alternatief uitdrukken als $\mathcal{R}(A) = \mathcal{D}$, met $\mathcal{R}(\cdot)$ het bereik of kolomruimte.

Definitie 2.1.1 *(orthogonale en orthonormale basis)*
We spreken van een orthogonale, respectievelijk orthonormale basis als de basisvectoren $\{a_1, \ldots, a_n\}$ orthogonaal, respectievelijk orthonormaal zijn. Als een basis niet orthogonaal is, spreken we van een scheve basis.

Het werken met een orthogonale basis heeft verschillende voordelen.

Ten eerste, voor een basis $\{a_1, \ldots, a_n\}$ van deelruimte $\mathcal{D}$ en twee vectoren $v \in \mathcal{D}, \ w \in \mathcal{D}$, ontbonden als

$$v = Ac = \sum_{i=1}^{n} c_i a_i, \quad w = Ad = \sum_{i=1}^{n} d_i a_i, \tag{2.2}$$

geldt dat

$$(v, w) = v^* w = \left(\sum_{i=1}^{n} \bar{c}_i a_i^* \right) \left(\sum_{j=1}^{n} d_i a_i \right) = c^* G d,$$

met

$$G = \begin{bmatrix} a_1^* a_1 & \cdots & a_1^* a_n \\ \vdots & & \vdots \\ a_n^* a_1 & \cdots & a_n^* a_n \end{bmatrix} = A^* A \tag{2.3}$$

de zogenaamde *grammatrix* horende bij de basis. Is deze basis orthogonaal, dan wordt G een diagonaalmatrix. Is de basis orthonormaal, dan is $G = I_n$. In dit laatste geval volgt dat

$$v^* w = c^* d = \sum_{i=1}^{n} \bar{c}_i d_i$$

en

$$\|v\|_2 = \sqrt{v^* v} = \sqrt{c^* c} = \|c\|_2. \tag{2.4}$$

Dus het inwendige product en norm kunnen bepaald worden op het niveau van de vectoren van coëfficiënten.

Ten tweede, stel dat we de coëfficiënten c_i in de eerste ontbinding in (2.2) niet kennen en willen berekenen. Vermenigvuldigen we de gelijkheid langs links met $a_j^*,\ j \in \{1,\ldots,n\}$ (i.e., maken we het inwendige product van linker- en rechterlid met a_j), dan bekomen we

$$\sum_{i=1}^{n}(a_j^* a_i)c_i = a_j^* v.$$

Als we deze nieuwe gelijkheid beschouwen voor $j = 1,\ldots,n$, dan verkrijgen we in matrix vorm

$$Gc = A^* v. \tag{2.5}$$

Om de coëfficiënten te bepalen moeten we dus een stelsel vergelijkingen oplossen (dat we later zullen identificeren als het zogenaamde normaalstelsel). Is de basis orthogonaal, dan hebben we echter een zeer eenvoudige en elegante uitdrukking voor de coëfficiënten. De j-de vergelijking herleidt zich dan tot $(a_j^* a)c_j = a_j^* v$, en we bekomen

$$c_j = \frac{a_j^* v}{a_j^* a_j} = \frac{a_j^* v}{\|a_j\|^2},\ j = 1,\ldots,n.$$

Is de basis bovendien orthonormaal, dan verkrijgen we $c_j = a_j^* v$ voor $j = 1,\ldots,n$.

Ten derde is het werken met een orthogonale basis aangewezen omwille van numerieke stabiliteit. Beschouw als voorbeeld voor basis $\{a_1,\ldots,a_n\}$ de volgende vraagstelling: gegeven de coëfficiënten c_i, bepaal $v = \sum_{i=1}^{n} c_i a_i$. Deze bewerking kan geïnterpreteerd worden als het matrix-vector product Ac, waarvoor het conditiegetal voor relatieve fouten begrensd wordt door het conditiegetal $\kappa(A) = \|A\|_2\|A^+\|_2$, zie Voorbeeld B.1 in Appendix B. Wanneer de basis niet orthogonaal is, kan dit probleem zeer slecht geconditioneerd zijn, zoals we zullen illustreren. Als gevolg worden ook bij het gebruik van een achterwaarts stabiel algoritme grote relatieve fouten verwacht. Bij het gebruik van een scheve basis is verder het probleem van het bepalen van de coëfficiënten c voor gegeven v slecht geconditioneerd als $\kappa(G)$ groot is. Bij een orthonormale basis daarentegen geldt $\kappa(A) = 1$ en $\kappa(G) = 1$.

Voorbeeld 2.1 *We beschouwen de deelruimte $\mathcal{D}$ van $\mathbb{R}^3$, gegeven door $\mathcal{D} = \{(v_1,v_2,v_3) \in \mathbb{R}^3 :\ v_2 = v_3\}$. Een basis wordt gegeven door $\{a_1,a_2\}$, met*

$$a_1 = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix},\ a_2 = \begin{bmatrix} 1 \\ \eta \\ \eta \end{bmatrix}, \tag{2.6}$$

waarbij $\eta > 0$ een parameter is. Deze basis is niet orthogonaal omdat $a_1^T a_2 = 1$. Merk dat voor kleine waarden van η de basisvectoren bijna dezelfde richting hebben. Om de

η	1.0000e-01	1.0000e-02	1.0000e-03	1.0000e-04	1.0000e-05
θ	1.4049e-01	1.4141e-02	1.4142e-03	1.4142e-04	1.4142e-05
$\kappa(A)$	1.4213e+01	1.4143e+02	1.4142e+03	1.4142e+04	1.4142e+05
$\kappa(G)$	2.0202e+02	2.0002e+04	2.0000e+06	2.0000e+08	2.0000e+10

Tabel 2.1: Conditiegetallen gerelateerd aan (2.6), met $A = [a_1\ a_2]$ en $G = A^*A$.

implicaties ervan te illustreren zetten we in de Tabel 2.1 de hoek[2] θ *uit tussen* a_1 *en* a_2*, alsook de conditiegetallen* $\kappa(A)$ *en* $\kappa(G)$*, in functie van* η*.*
Beschouw nu vector $v = [1\ 1\ 1]^T \in \mathcal{D}$*, die we kunnen ontbinden als*

$$v = \underbrace{\left(1 - \frac{1}{\eta}\right)}_{c_1} a_1 + \underbrace{\frac{1}{\eta}}_{c_2} a_2. \tag{2.7}$$

Voor kleine η *zijn beide coëfficiënten groot en hebben een verschillend teken. Voor* $\eta = 10^{-4}$ *bijvoorbeeld is* $c_1 = -9999$ *en* $c_2 = 10000$*. Wanneer de coëfficiënten als gegeven beschouwd worden en* v *bepaald moet worden, dan geeft een relatieve fout van* 0.01 *op* c_1 *aanleiding tot een relatieve fout op* v *gelijk aan* $-99.99/\sqrt{3}$*! Bij het uitrekenen van het rechterlid van (2.7) treedt er namelijk een zgn. gevaarlijke aftrekking*[3] *op. De basis* $\{q_1, q_2\}$*, met*

$$q_1 = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix},\ q_2 = \begin{bmatrix} 0 \\ \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} \end{bmatrix},$$

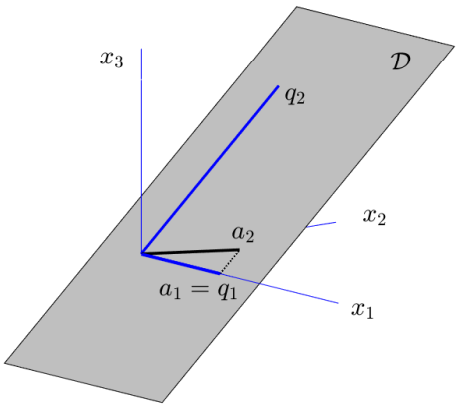
is daarentegen orthonormaal, wat $\kappa([q_1\ q_2]) = 1$ *impliceert. We kunnen nu* v *ontbinden als* $v = d_1q_1 + d_2q_2$ *met* $(d_1, d_2) = (1, \sqrt{2})$*. Merk dat* $d_1^2 + d_2^2 = 5 = \|v\|_2^2$*, wat in overeenstemming is met (2.4). Een relatieve fout van* 0.01 *op* c_1 *leidt nu tot een relatieve fout van* $0.01/\sqrt{3}$ *op* v*.*
In Figuur 2.1 worden de deelruimte $\mathcal{D}$ *en de twee basissen gevisualiseerd, waarbij* $\eta = 0.1$*.*

Voorbeeld 2.2 *In Hoofdstuk 9 zullen we het algoritme van Arnoldi behandelen, een iteratief algoritme voor het berekenen van extreme eigenwaarden van een grote matrix* $M \in \mathbb{R}^{m\times m}$*, dat gebaseerd is op het zoeken van benaderingen van eigenvectoren in een zogenaamde Krylov ruimte*

$$\mathcal{K}_n(M, v_0) =< v_0, Mv_0, M^2v_0, \ldots, M^{n-1}v_0 >,$$

[2]De hoek tussen vectoren $x,\ y \in \mathbb{R}^n$ kan berekend worden als $\theta = \arccos\left(\frac{x^T y}{\|x\|_2\|y\|_2}\right)$.

[3]Een gevaarlijke aftrekking (in het Engels *catastrophic cancellation)* bestaat uit het aftrekken van twee ongeveer even grote getallen waar fouten op zitten. Dit leidt tot het opblazen van de relatieve fout op het resultaat.



Figuur 2.1: Twee basissen voor de deelruimte gekenmerkt door $x_2 = x_3$.

met v_0 een willekeurig gekozen startvector en $n \ll m$. Een voor de hand liggende basis voor de Krylov ruimte bestaat uit de genormaliseerde vectoren $a_i = \frac{1}{\|M^{i-1}v_0\|_2} M^{i-1}v_0$, $i = 1, \ldots, n$, die we kunnen groeperen in matrix

$$A_n = \left[\frac{1}{\|v_0\|_2} v_0 \;\; \frac{1}{\|Mv_0\|_2} Mv_0 \;\; \ldots \;\; \frac{1}{\|M^{n-1}v_0\|_2} M^{n-1}v_0 \right].$$

Stel als numeriek experiment dat $M = QDQ^T$ met $Q \in \mathbb{R}^{102\times 102}$ een orthogonale matrix, D een diagonaalmatrix met diagonaalelementen $200, 150, 100, 99, \ldots, 2, 1$, en dat $Q^T v_0 = [1 \;\; \cdots 1]^T$. Dan kan je eenvoudig nagaan dat $\kappa(M) = 200$, wat duidt op een goede conditie. In volgende tabel tonen we het conditiegetal van A_n in functie van n

n	5	6	7	8	9	10
$\kappa(A_n)$	1.877e+02	1.097e+03	9.838e+03	9.780e+04	9.923e+05	1.013e+07

De kolommen van A_n hebben weliswaar lengte 1 maar zijn dus verre van orthogonaal, en met toenemende n neemt het conditiegetal van A_n sterk toe. Om numerieke problemen te vermijden bestaat een belangrijke stap in het algoritme van Arnoldi uit het opstellen van een orthogonale basis van $\mathcal{K}_n(M, v_0)$.

In Sectie 2.3 gaan we dieper in op algoritmen voor het construeren van orthogonale basissen. Maar eerst behandelen we projectoren, een klasse van matrices die een belangrijke rol spelen bij de constructie en interpretatie van deze algoritmen.

2.2 Begrip projector

Definitie 2.2.2 *(projector)*
Een projector *is een matrix $P \in \mathbb{C}^{m\times m}$ die idempotent is, dit is $P^2 = P$.*

Stel $v \in \mathbb{C}^m$ een willekeurige vector, dan is $Pv \in \mathcal{R}(P)$ volgens de definitie van het bereik, en is $(I-P)v \in \mathcal{N}(P)$ omdat $P(I-P)v = (P-P^2)v = (P-P)v = 0$. We kunnen dus v ontbinden in componenten volgens $\mathcal{R}(P)$ en $\mathcal{N}(P)$ als

$$v = \underbrace{Pv}_{\in \mathcal{R}(P)} + \underbrace{(I-P)v}_{\in \mathcal{N}(P)}. \tag{2.8}$$

Aan volgende eigenschappen is voldaan.

- Als $v \in \mathcal{R}(P)$, dan is $Pv = v$. Inderdaad, als $v \in \mathcal{R}(P)$, dan $\exists u : \ v = Pu$, en dus is $Pv = P^2u = Pu = v$.
- Er geldt $\mathcal{R}(P) \cap \mathcal{N}(P) = \{0\}$. Immers, stel $x \in \mathcal{R}(P)$ en $x \in \mathcal{N}(P)$. Er volgt dat $x = Px = 0$.
- Er geldt $\dim(\mathcal{R}(P)) + \dim(\mathcal{N}(P)) = m$, volgend uit zowel de eerste dimensiestelling als de vorige eigenschap.
- De ontbinding in componenten volgens $\mathcal{R}(P)$ en $\mathcal{N}(P)$ is uniek. Stel $v = x_1 + y_1 = x_2 + y_2$, met $x_1, \ x_2 \in \mathcal{R}(P)$ en $y_1, \ y_2 \in \mathcal{N}(P)$. Er geldt voor $i \in \{1,2\}$ dat $Pv = Px_i + Py_i = Px_i = x_i$. Hieruit volgt dat $x_1 = x_2$.

De meetkundige betekenis is als volgt. Matrix P projecteert een vector op de ruimte $\mathcal{R}(P)$, waarbij de richting bepaald wordt door $\mathcal{N}(P)$.
Matrix $\widetilde{P} = I - P$ is ook een projector; immers

$$\widetilde{P}^2 = (I-P)^2 = I - P - P + P^2 = I - P = \widetilde{P}.$$

Verder is voldaan aan

$$\mathcal{R}(I-P) = \mathcal{N}(P) \ , \ \mathcal{N}(I-P) = \mathcal{R}(P).$$

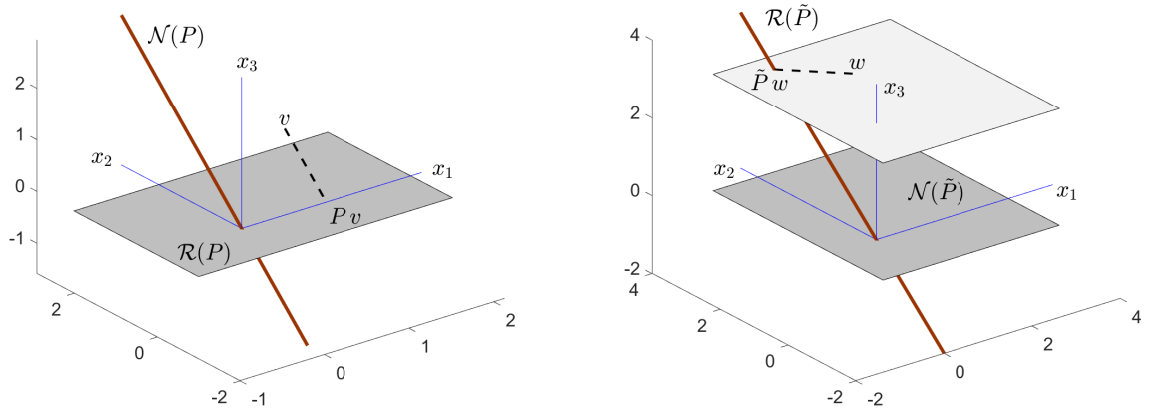
De ontbinding (2.8) kan nu ook geschreven worden als

$$v = \underbrace{(I-\widetilde{P})v}_{\in \mathcal{R}(P)} + \underbrace{\widetilde{P}v}_{\in \mathcal{N}(P)} .$$

Matrix $\widetilde{P}$ projecteert dus op $\mathcal{N}(P)$ waarbij de richting bepaald wordt door $\mathcal{R}(P)$. Deze matrix wordt daarom de *complementaire projector* van P genoemd.

Voorbeeld 2.3 *Matrix*

$$P = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & -1 \\ 0 & 0 & 0 \end{bmatrix} \tag{2.9}$$



Figuur 2.2: Meetkundige interpretatie van projector (2.9) en de bijhorende complementaire projector.

voldoet aan de definitie van een projector omdat $P^2 = P$. *Er geldt*

$$\mathcal{R}(P) = \left\{ [x_1 \ x_2 \ x_3]^T : \ x_3 = 0 \right\}, \quad \mathcal{N}(P) = \left\{ k\,[0 \ 1 \ 1]^T : \ k \in \mathbb{C} \right\}. \tag{2.10}$$

Dus P *projecteert in* $\mathbb{R}^3$ *op het horizontale vlak, in de richting van vector* $[0 \ 1 \ 1]^T$. *Met* $v = [1 \ 1 \ 1]^T$ *verkrijgen we* $Pv = [1 \ 0 \ 0]$. *Merk dat* $v - Pv = [0 \ 1 \ 1]^T$. *Dit wordt gevisualiseerd in het linkerluik van Figuur 2.2.*
De complementaire projector wordt gegeven door

$$\widetilde{P} = I - P = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 1 \end{bmatrix}.$$

Met $w = [1 \ 2 \ 3]^T$ *krijgen we* $\tilde{P}w = [0 \ 3 \ 3]^T \in \mathcal{N}(P)$. *De richting van projectie is voor deze vector*

$$w - \widetilde{P}w = \begin{bmatrix} 1 \\ -1 \\ 0 \end{bmatrix} \in \mathcal{R}(P),$$

zie het rechterluik van Figuur 2.2.

Vanuit praktisch oogpunt is het volgende, omgekeerde probleem interessanter: gegeven twee deelruimten $\mathcal{S}_1$ en $\mathcal{S}_2$ die voldoen aan

$$\dim(\mathcal{S}_1) = n, \quad \dim(\mathcal{S}_2) = m - n, \quad \mathcal{S}_1 \cap \mathcal{S}_2 = \{0\}, \tag{2.11}$$

bepaal een idempotente matrix $P_{\mathcal{S}_1,\mathcal{S}_2}$ die projecteert op $\mathcal{S}_1$ in de richting van $\mathcal{S}_2$. Stel vectoren $\{a_1,\ldots,a_n\}$ vormen een basis voor $\mathcal{S}_1$ en $\{b_1,\ldots,b_n\}$ een basis voor het orthogonale complement van $\mathcal{S}_2$, $\mathcal{S}_2^{\perp}$. Vorm de $m\times n$ matrices A en B met als kolommen $a_1,\ldots,a_n$, respectievelijk $b_1,\ldots,b_n$. Neem een willekeurige vector $v\in\mathbb{C}^m$. Vector $v_1 = P_{\mathcal{S}_1,\mathcal{S}_2}v$ moet tot $\mathcal{S}_1$ behoren en v_1 kan dus geschreven worden als een lineaire combinatie van de kolommen van A: $v_1 = Ay$. We eisen

$$(v-v_1)\in\mathcal{S}_2 \Leftrightarrow (v-v_1)\perp\mathcal{S}_2^{\perp} =< b_1,\ldots,b_n >$$

Dus

$$b_j^*(v-v_1)=0\ ,\ j=1,2,\ldots,n.$$

$$B^*(v-v_1)=0,$$

$$B^*(v-Ay)=0,$$

$$B^*v = B^*Ay.$$

Bewijs zelf dat onder voorwaarden (2.11) de n-bij-n matrix B^*A inverteerbaar is. We verkrijgen dan

$$y=(B^*A)^{-1}B^*v\ ,\ v_1=Ay=A(B^*A)^{-1}B^*v$$

en dus is

$$P_{\mathcal{S}_1,\mathcal{S}_2} = A(B^*A)^{-1}B^*. \tag{2.12}$$

Deze matrix is inderdaad idempotent.

Voorbeeld 2.4 *We hernemen het vorige voorbeeld. Vertrekken we van ruimten (2.10) is een mogelijke keuze voor A en B*

$$A=\begin{bmatrix}1&0\\0&1\\0&0\end{bmatrix},\ B=\begin{bmatrix}1&1\\0&1\\0&-1\end{bmatrix}.$$

Invullen in formule (2.12) levert (2.9) op.

We definiëren nu een orthogonale projector.

Definitie 2.2.3*(orthogonale projector)*
Een projector P is orthogonaal *indien $\mathcal{R}(P)$ en $\mathcal{N}(P)$ onderling orthogonale ruimten zijn.*

Een projector die niet orthogonaal is, noemen we een scheve projector.

Eigenschap 2.1 *Een projector P is orthogonaal als en alleen als $P=P^*$.*

Bewijs De voorwaarde $P = P^*$ is nodig. Beschouw een orthonormale basis $\{q_1, \ldots, q_n\}$ van $\mathcal{R}(P)$ en een orthonormale basis $\{q_{n+1}, \ldots, q_m\}$ van $\mathcal{N}(P)$. Omdat volgens de definitie beide ruimten orthogonaal zijn, is $Q = [q_1 \ \cdots \ q_n \ q_{n+1} \ \cdots \ q_m]$ een unitaire matrix. We verkrijgen

$$PQ = [q_1 \ \cdots \ q_n \ 0 \cdots 0] \ \Rightarrow Q^*PQ = \mathrm{diag}(1, \ldots, 1, 0, \ldots, 0).$$

Vermits Q^*PQ dus reëel is, geldt $Q^*PQ = (Q^*PQ)^* = Q^*P^*Q$, waaruit volgt dat $P = P^*$.
De voorwaarde is voldoende. Neem willekeurige $x = Pu \in \mathcal{R}(P)$ en $y \in \mathcal{N}(P)$. Dan is

$$x^*y = (Pu)^*y = u^*P^*y = u^*Py = 0.$$

De ruimten $\mathcal{R}(P)$ en $\mathcal{N}(P)$ zijn dus orthogonaal. □

Laten we nu een projector $P_{\mathcal{D}}$ construeren die orthogonaal projecteert op een gegeven ruimte $\mathcal{D}$ van dimensie n, vertrekkende van basisvectoren $\{a_1, \ldots, a_n\}$ van $\mathcal{S}$, die de kolommen van matrix A vormen. De afleiding is analoog als deze voor een scheve projector. Nu eisen we dat

$$(v - v_1) \perp \mathcal{D} =< a_1, \ldots, a_n >,$$

wat leidt tot

$$P_{\mathcal{D}} = A(A^*A)^{-1}A^*. \tag{2.13}$$

Merk dat $P_{\mathcal{D}} = P_{\mathcal{D}}^*$. Je kan verder aantonen dat $\|P_{\mathcal{D}}\|_2 = 1$ (bijvoorbeeld door gebruik te maken van de singuliere-waardenontbinding van A).
De projector $P_{\mathcal{D}^\perp}$ op het orthogonaal complement van de deelruimte $\mathcal{D}$ wordt gegeven door de complementaire projector

$$P_{\mathcal{D}^\perp} = I - P_{\mathcal{D}} = I - A(A^*A)^{-1}A^*.$$

Voorbeeld 2.5 *Laten we een orthogonale projector construeren op*

$$\mathcal{D} =< \begin{bmatrix} 1 \\ 0 \\ 1 \end{bmatrix}, \begin{bmatrix} 1 \\ 1 \\ 0 \end{bmatrix} > .$$

We kiezen $A = \begin{bmatrix} 1 & 1 \\ 0 & 1 \\ 1 & 0 \end{bmatrix}$ *zodat* $A^TA = \begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix}$, $(A^TA)^{-1} = \begin{bmatrix} 2/3 & -1/3 \\ -1/3 & 2/3 \end{bmatrix}$. *Zo bekomen we*

$$P_{\mathcal{D}} = A(A^TA)^{-1}A^T = \begin{bmatrix} 1 & 1 \\ 0 & 1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} 2/3 & -1/3 \\ -1/3 & 2/3 \end{bmatrix} \begin{bmatrix} 1 & 0 & 1 \\ 1 & 1 & 0 \end{bmatrix} = \begin{bmatrix} 2/3 & 1/3 & 1/3 \\ 1/3 & 2/3 & -1/3 \\ 1/3 & -1/3 & 2/3 \end{bmatrix}.$$

Merk op dat $P_{\mathcal{D}}$ *symmetrisch is en* $P_{\mathcal{D}}^2 = P_{\mathcal{D}}$, *en dat de rang van* $P_{\mathcal{D}}$ *gelijk is aan 2.*

Stel $v = \begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix}$, *dan geldt* $v_1 = P_{\mathcal{D}}\, v = \begin{bmatrix} 7/3 \\ 2/3 \\ 5/3 \end{bmatrix} = 5/3 \begin{bmatrix} 1 \\ 0 \\ 1 \end{bmatrix} + 2/3 \begin{bmatrix} 1 \\ 1 \\ 0 \end{bmatrix}$. *Merk* $v_2 = v - P_{\mathcal{D}}\, v = \begin{bmatrix} -4/3 \\ 4/3 \\ 4/3 \end{bmatrix}$, *waarbij* $\begin{bmatrix} 1 & 0 & 1 \end{bmatrix} v_2 = 0$, $\begin{bmatrix} 1 & 1 & 0 \end{bmatrix} v_2 = 0$.

Laten we tenslotte kijken naar een aantal speciale gevallen van (2.13).

- Wanneer $n = 1$ is en er dus maar 1 basisvector a is ($\mathcal{D} =< a >$), wordt A een $m \times 1$ matrix en $A^*A = a^*a$. We verkrijgen dan

$$P_{<a>} = \frac{aa^*}{a^*a} = \frac{aa^*}{\|a\|_2^2}, \quad P_{<a>^\perp} = I - \frac{aa^*}{a^*a}. \tag{2.14}$$

- Als een basis $\{q_1, q_2, \ldots, q_n\}$ voor $\mathcal{D}$ orthonormaal is en $Q = [q_1 \ \cdots \ q_n]$, dan is Q^*Q de eenheidsmatrix en verkrijgen we

$$P_{\mathcal{D}} = QQ^*, \quad P_{\mathcal{D}^\perp} = I - QQ^*, \tag{2.15}$$

  waaruit volgt

$$P_{\mathcal{D}}\, v = QQ^*v = \sum_{i=1}^{n} (q_i^* v)\, q_i, \quad v \in \mathbb{C}^m. \tag{2.16}$$

2.3 Orthogonalisatieprocedures

In Sectie 2.1 hebben we het belang van een orthogonale basis geïllustreerd. In deze sectie leggen we uit hoe we orthonormale basissen kunnen opstellen. In §2.3.1 wordt het verband aangetoond tussen het construeren van een geneste orthogonale basis en een fundamentele matrix-ontbinding in de numerieke lineaire algebra, namelijk de QR-factorisatie. Vervolgens worden verscheidene algoritmen besproken in §2.3.2-§2.3.4.

2.3.1 Relatie met de QR-factorisatie

We vertrekken van een basis $\{a_1, \ldots, a_n\}$ van een n-dimensionale deelruimte van $\mathbb{C}^m$, waarmee we een m-bij-n matrix A kunnen associëren, $A = \begin{bmatrix} a_1 & a_2 & \ldots & a_n \end{bmatrix}$. We wensen deze basis te vervangen door een orthonormale basis $\{q_1, \ldots, q_n\}$. Bovendien leggen we op dat deze basis *genest* is, in de volgende betekenis:

$$\begin{aligned} &< a_1 > = < q_1 > \\ &< a_1, a_2 > = < q_1, q_2 > \\ &< a_1, a_2, a_3 > = < q_1, q_2, q_3 > \\ &\qquad\qquad \ddots \\ &< a_1, a_2, \ldots, a_n > = < q_1, q_2, \ldots, q_n > \end{aligned} \tag{2.17}$$

Er volgt:
a_1 ligt in de deelruimte opgespannen door q_1: $a_1 = r_{11}q_1$
a_2 ligt in de deelruimte opgespannen door q_1 en q_2: $a_2 = r_{12}q_1 + r_{22}q_2$
a_3 ligt in de deelruimte opgespannen door q_1, q_2 en q_3: $a_3 = r_{13}q_1 + r_{23}q_2 + r_{33}q_3$
$\vdots$
$a_n = r_{1n}q_1 + r_{2n}q_2 + \ldots + r_{nn}q_n$
Deze relaties kunnen we in matrixvorm gieten, namelijk

$$A = [a_1\ a_2\ \ldots\ a_n] = [q_1\ q_2\ \ldots\ q_n] \begin{bmatrix} r_{11} & r_{12} & r_{13} & \ldots & r_{1n} \\ 0 & r_{22} & r_{23} & \ldots & r_{2n} \\ 0 & 0 & r_{33} & \ldots & r_{3n} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \ldots & r_{nn} \end{bmatrix} := \hat{Q}\hat{R}, \tag{2.18}$$

met $\hat{Q} \in \mathbb{C}^{m\times n}$ en $\hat{R} \in \mathbb{C}^{n\times n}$. De ontbinding van A door het product $\hat{Q}\hat{R}$ wordt de *onvolledige QR-factorisatie* van A genoemd. Er geldt:

$$\hat{Q}^*\hat{Q} = I_n. \tag{2.19}$$

Voor $m \neq n$ is $\hat{Q}$ geen unitaire matrix omdat zij niet vierkant is, maar zij heeft wel orthonormale kolommen. In §2.3.2 zullen we constructief aantonen dat de onvolledige QR-factorisatie uniek is als we het argument van de diagonaalelementen van $\hat{R}$ (of teken bij de ontbinding van een reële matrix in reële factoren) opleggen.

Opmerking 2.1 *Het gegeven dat $\{a_1, \ldots, a_n\}$ een basis is, impliceert dat A van volle kolomrang is. Vertrekken we echter van een matrix A die niet van volle kolomrang is, dan kan niet meer voldaan worden aan (2.17) maar bestaat een factorisatie van de vorm (2.18)-(2.19) nog steeds. De rang van $\hat{R}$ is dan gelijk aan de rang van A.*

Wanneer we matrix $\hat{Q}$ uitbreiden met $m-n$ kolommen $q_{n+1}, q_{n+2}, \ldots, q_m$ die orthonormaal zijn op de kolommen van $\hat{Q}$ bekomen we een unitaire matrix

$$Q = [\ q_1\ \ldots\ q_n\ q_{n+1}\ \ldots\ q_m] = [\hat{Q}\ q_{n+1}\ \ldots\ q_m].$$

Tegelijkertijd breiden we $\hat{R}$ uit met $m-n$ 0-rijen:

$$R = \begin{bmatrix} r_{11} & r_{12} & \ldots & r_{1n} \\ 0 & r_{22} & \ldots & r_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \ldots & r_{nn} \\ 0 & 0 & \ldots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \ldots & 0 \end{bmatrix} = \begin{bmatrix} \hat{R} \\ 0 \end{bmatrix}. \tag{2.20}$$

We verkrijgen zo een *volledige QR-factorisatie* van de matrix A,

$$A = QR, \tag{2.21}$$

met Q een $(m \times m)$ unitaire matrix en R een $(m \times n)$ bovendriehoeksmatrix.

Vertrekkende van een reële matrix A kunnen Q en R als reële matrices gekozen worden, en de algoritmen die we in wat volgt behandelen geven dan aanleiding tot een factorisatie in reële matrices. Matrix Q, die voldoet aan $Q^T Q = I$, is dan een *orthogonale matrix*.

2.3.2 Gram-Schmidt orthogonalisatie

We starten met een algebraïsche afleiding. Vertrekkende van de kolommen van A bepalen we de kolommen van $\hat{Q}$ en de kolommen van $\hat{R}$ in opeenvolgende stappen. In stap j bepalen we de j-de kolom van $\hat{Q}$ en de j-de kolom van $\hat{R}$.

stap 1: $r_{11} = \|a_1\|_2$, $q_1 = a_1/r_{11}$.

stap j: Stel $q_1, q_2, ..., q_{j-1}$ en de $j-1$ eerste kolommen van $\hat{R}$ gekend.
$a_j = r_{1j}q_1 + r_{2j}q_2 + \ldots + r_{j-1j}q_{j-1} + r_{jj}q_j$.
Wegens de orthogonaliteit van vectoren q_i geldt: $q_i^* a_j = r_{ij}$. Verder is
$r_{jj}q_j = a_j - r_{1j}q_1 - r_{2j}q_2 - \cdots - r_{j-1j}q_{j-1} := v_j$,
dus q_j is gekend op een constante na en we weten dat $\|q_j\|_2 = 1$.
We kiezen $r_{jj} = \|v_j\|_2$ en $q_j = v_j/r_{jj}$.

Dit leidt tot Algoritme 1.

Algoritme 1 Klassiek Gram-Schmidt algoritme

```
1: for j = 1 to n do
2:    v_j = a_j
3:    for i = 1 to j − 1 do
4:       r_ij = q_i^* a_j
5:       v_j = v_j − r_ij q_i
6:    end for
7:    r_jj = ‖v_j‖_2
8:    q_j = v_j / r_jj
9: end for
```

De meetkundige interpretatie is als volgt. Stel $Q_j = [q_1, \ldots, q_j]$. In de j-de iteratie van de buitenste lus wordt v_j opgebouwd tot

$$\begin{aligned} v &= a_j - r_{1j}q_1 - r_{2j}q_2 - \cdots - r_{j-1j}q_{j-1} \\ &= a_j - q_1(q_1^* a_j) - q_2(q_2^* a_j) - \cdots - q_{j-1}\left(q_{j-1}^* a_j\right) \\ &= a_j - Q_{j-1}Q_{j-1}^* a_j \\ &= a_j - P_{<q_1,\ldots,q_{j-1}>} a_j, \end{aligned} \tag{2.22}$$

waarbij de laatste stap volgt uit (2.15). We nemen de nieuwe kolom a_j, we projecteren deze vector orthogonaal op de ruimte opgespannen door $q_1, \ldots, q_{j-1}$ en trekken het resultaat er van af (wat ook geïnterpreteerd kan worden als het projectoren van a_j op het orthogonaal complement van $< q_1, \ldots, q_{j-1} >$). Op deze manier verwijderen we uit a_j de componenten in de richtingen van $< q_1, \ldots, q_{j-1} >$. Vector v staat dus orthogonaal op deze ruimte. De

richting van q_j wordt dan bepaald door v, we moeten enkel nog normaliseren. Merk dat de normalisatie in elke stap uniek is op een complex getal met modulus 1 na. Met de gemaakte keuze leggen we op dat de diagonaalelementen van $\hat{R}$ reëel en positief zijn.

Het algoritme van Gram-Schmidt is niet numeriek stabiel. Wat stabiliteitsanalyse betreft past de QR-factorisatie niet helemaal in het schema in Appendix B omdat het resultaat niet enkel uit factoren $\hat{Q}$ en $\hat{R}$ bestaat maar er ook een orthogonaliteitseis op $\hat{Q}$ ligt. In numerieke experimenten komen de berekende $\hat{Q}$ en $\hat{R}$ weliswaar overeen met een factorisatie van een perturbatie van A van ordegrootte machineprecisie, maar bij toenemende n gaat de *orthogonaliteit* van de kolommen van Q snel verloren. Dit kan grotendeels verklaard worden door de opbouw van v in (2.22), en is meer uitgesproken wanneer $\|v\|_2$ klein is ten opzichte van a_j, waarbij dan onvermijdelijk gevaarlijke aftrekkingen optreden. Merk dat een kleine waarde van $\|v\|_2$ impliceert dat a_j bijna in de kolomruimte ligt opgespannen door $\{a_1, \ldots, a_{j-1}\}$, dus hoe sterker de basis afwijkt van een orthogonale basis, des te sneller gaat orthogonaliteit van de berekende $\hat{Q}$ in eindige precisie verloren. Een extreem geval treedt op als A niet van volle rang is, waarbij in theorie $v_j = 0$ voor een waarde van $j \in \{1, \ldots, n\}$.
Bij het zogenaamde gewijzigde Gram-Schmidt algoritme, met pseudocode in Algoritme 2, is het verlies aan orthogonaliteit nog steeds aanwezig maar wat minder uitgesproken. Enkel in lijn 4 verschilt de pseudocode van deze voor het klassieke Gram-Schmidt algoritme.

Algoritme 2 Gewijzigd Gram-Schmidt algoritme

```
1: for j = 1 to n do
2:     v_j = a_j
3:     for i = 1 to j − 1 do
4:         r_ij = q_i^* v_j
5:         v_j = v_j − r_ij q_i
6:     end for
7:     r_jj = ‖v_j‖_2
8:     q_j = v_j / r_jj
9: end for
```

Laten we even ingaan op de achterliggende principes. De twee stappen in de binnenste lus kan je interpreteren als het uitrekenen van $(I - P_{<q_i>})v_j$. Beschouw nu de k-de kolom van A, a_k. In het klassieke Gram-Schmidt algoritme worden de componenten volgens $\{q_1, \ldots, q_{k-1}\}$ in één keer verwijderd: we berekenen

$$v = (I - P_{<q_1,\ldots,q_{k-1}>})a_k, \tag{2.23}$$

waarna v genormaliseerd wordt tot q_k. In het gewijzigde algoritme wordt eerst de component volgens q_1 verwijderd, vervolgens de component volgens q_2 enz., wat je kan voorstellen door

$$v = (I - P_{<q_{k-1}>}) \ldots (I - P_{<q_2>})(I - P_{<q_1>})a_k. \tag{2.24}$$

Je kan gemakkelijk nagaan dat (2.23) en (2.24) wiskundig equivalent zijn. Het grote verschil tussen beide algoritmen zit in de berekening van r_{ij}. Bij het klassieke Gram-Schmidt algoritme wordt het inwendige product van q_i en a_j direct uitgerekend. Bij het gewijzigde algoritme rekent men het inwendige product uit van q_i en een vector v_j, bekomen uit a_j door de componenten volgens $q_1, \ldots, q_{i-1}$ te verwijderen. In theorie maakt dit geen verschil maar in eindige precisie heeft dit een gunstig effect op de voortplanting van fouten, waarvoor $\|v_j\|_2 \leq \|a_j\|_2$ een aanwijzing is.

In praktijk kan de stabiliteit van Gram-Schmidt algoritmen significant verbeterd worden door een zogenaamde *herorthogonalisatie*, met twee varianten waarvan de eerste de meest gebruikelijke is.

1. *Stapsgewijse variant.* Bij Algoritme 1 komt dit neer op het vervangen van lijnen 2-6 door:

   $v_j = a_j$
   for $i = 1$ to $j - 1$ **do**
   $r_{ij} = q_i^* a_j$
   $v_j = v_j - r_{ij} q_i$
   end for
   $w_j = v_j$
   for $i = 1$ to $j - 1$ **do**
   $s_{ij} = q_i^* w_j$
   $v_j = v_j - s_{ij} q_i$
   $r_{ij} = r_{ij} + s_{ij}$
   end for

2. *Simultane variant.* Na het berekenen van de onvolledige QR-factorisatie, die resulteert in factoren $\hat{Q}_1$ en $\hat{R}_1$ wordt het algoritme opnieuw toegepast met als input de eerste factor, wat resulteert in $\hat{Q}_1 \approx \hat{Q}_2 \hat{R}_2$. We bepalen dan $\hat{Q} = \hat{Q}_2$ en $\hat{R} = \hat{R}_2 \hat{R}_1$. Bij Algoritme 2 is dit meestal voldoende om orthogonaliteit van de kolommen van $\hat{Q}$ te realiseren, bij Algoritme 1 is soms het meermaals herhalen van deze procedure noodzakelijk.

Verder merken we op

- algoritmen 1 en 2 vragen beide $\sim (2mn^2)$ elementaire bewerkingen[4].
- Beide algoritmen, gecombineerd met herorthogonalisatie, zijn bijzonder geschikt voor grootschalige problemen waarbij $m >> n$ en enkel een *onvolledige* QR-factorisatie gewenst is, i.e., een orthogonalisatie van de kolommen van A. Merk hierbij op dat voor een vaste waarde van n, het aantal bewerkingen *lineair* schaalt met dimensie m, in tegenstelling tot de algoritmen die we in §2.3.3-§2.3.4 zullen behandelen. Algoritmes 1-2 hebben als voordeel dat alle kolommen van A niet op voorhand gekend

[4]Hiermee bedoelen we optellingen, aftrekkingen, vermenigvuldigingen en delingen met complexe getallen. Indien uitsluitend gewerkt wordt met data uit de ingebedde ruimte $\mathbb{R}^m$, gaat het om elementaire bewerkingen met reële getallen.

moeten zijn, en de orthonormale basis "on the fly" uitgebreid kan worden. Dit is onder meer belangrijk bij zogenaamde Krylov methodes, waarvan het algoritme van Arnoldi, behandeld in Hoofdstuk 9, een voorbeeld is.

- In principe zouden we met algoritmen 1 en 2 ook een volledige QR-factorisatie kunnen berekenen. Het volstaat daarbij om A eerst uit te breiden tot een vierkante (inverteerbare) m-bij-m matrix, en achteraf de R-factor te beperken tot de eerste n kolommen. Er bestaan echter betere, stabiele methodes gebaseerd op orthogonale transformaties, die we in wat volgt behandelen.

2.3.3 QR-factorisatie met Givens-rotaties

De volledige QR-factorisatie (2.21) kan geschreven worden als $Q^*A = R$, wat de volgende methodologie suggereert: zoek een unitaire transformatie die in matrix A nullen creëert onder de diagonaal. Zulke transformatie zullen we construeren, in de vorm van een samenstelling van elementaire unitaire transformaties.
We beschouwen als startpunt een vector $x = \begin{bmatrix} x_1 & x_2 \end{bmatrix}^T \in \mathbb{R}^2$ met $x_2 \neq 0$ en zoeken een orthogonale matrix $G \in \mathbb{R}^{2\times 2}$ zodat de transformatie beschreven door[5] G^T de tweede component van x op nul brengt. Omdat een orthogonale transformatie de Euclidische norm bewaart, kunnen we dit schrijven als

$$G^T x = \begin{bmatrix} \pm\sqrt{x_1^2 + x_2^2} \\ 0 \end{bmatrix}. \tag{2.25}$$

Je kan nagaan dat matrix

$$G = \begin{bmatrix} c & -s \\ s & c \end{bmatrix}, \text{ met } c = \frac{x_1}{\sqrt{x_1^2 + x_2^2}}, \quad s = \frac{x_2}{\sqrt{x_1^2 + x_2^2}}, \tag{2.26}$$

orthogonaal is en voldoet aan (2.25), met keuze voor het plusteken. De meetkundige interpretatie is als volgt. Matrix G beschrijft een *rotatie in het vlak*. Variabelen c en s komen overeen met cosinus en sinus van de rotatiehoek, die gelijk is aan de hoek tussen de positieve x_1-as en vector x. De afbeelding beschreven door $G^T = G^{-1}$ draait vector x in wijzerzin tot deze in de richting van de positieve x_1-as komt te liggen.

Voorbeeld 2.6 *Met $x = [0\ 1]^T$ wordt (2.26)*

$$G = \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix}. \tag{2.27}$$

Deze orthogonale matrix heeft eigenwaarden $\pm\imath = e^{\pm\frac{\pi}{2}\imath}$. Matrices G en G^T roteren een vector over $90°$ tegen uurwerkzin, respectievelijk in uurwerkzin. Er geldt $G^T x = [1\ 0]^T$.

[5] Omwille van consistentie met gebruikelijke notatie in de literatuur.

We kunnen bovenstaande methodologie algebraïsch veralgemenen naar de vectorruimte $\mathbb{C}^m$ over de complexe getallen van dimensie $m \geq 2$. Een m-dimensionale *Givens-rotatie* is een unitaire matrix van de vorm:

$$G_{i,j} = \begin{bmatrix} 1 & \cdots & 0 & 0 & & \cdots & & 0 & 0 & \ldots & 0 \\ \vdots & \ddots & \vdots & \vdots & & \ddots & & \vdots & \vdots & \ddots & \vdots \\ 0 & \cdots & 1 & 0 & & \cdots & & 0 & 0 & \cdots & 0 \\ 0 & \cdots & 0 & c & 0 & \cdots & 0 & -\bar{s} & 0 & \cdots & 0 \\ 0 & \cdots & 0 & 0 & 1 & \cdots & 0 & 0 & 0 & \cdots & 0 \\ \vdots & \ddots & \vdots & \vdots & & \ddots & & \vdots & \vdots & \ddots & \vdots \\ 0 & \cdots & 0 & 0 & 0 & \cdots & 1 & 0 & 0 & \cdots & 0 \\ 0 & \cdots & 0 & s & 0 & \cdots & 0 & \bar{c} & 0 & \ldots & 0 \\ 0 & \cdots & 0 & 0 & & \cdots & & 0 & 1 & \cdots & 0 \\ \vdots & \ddots & \vdots & \vdots & & \ddots & & \vdots & \vdots & \ddots & \vdots \\ 0 & \cdots & 0 & 0 & & \cdots & & 0 & 0 & \cdots & 1 \end{bmatrix} \in \mathbb{C}^{m \times m}, \tag{2.28}$$

met

$$|c|^2 + |s|^2 = 1 \tag{2.29}$$

en de c's en s'en op de snijdingen van rijen en kolommen i en j. Gegeven een vector $x = [\ x_1\ \ldots\ x_{i-1}\ x_i\ x_{i+1}\ \ldots\ x_{j-1}\ x_j\ x_{j+1}\ \ldots\ x_m\ ]^T \in \mathbb{C}^m$ dan kan men $G_{i,j}$ construeren zo dat de j-de component van $G_{i,j}^* x$ gelijk is aan 0. Kies daartoe

$$c = \frac{x_i}{\sqrt{|x_i|^2 + |x_j|^2}} \text{ en } s = \frac{x_j}{\sqrt{|x_i|^2 + |x_j|^2}}, \tag{2.30}$$

wat leidt tot

$$G_{i,j}^* x = [\ x_1\ \ldots\ x_{i-1}\ \sqrt{|x_i|^2 + |x_j|^2}\ x_{i+1}\ \ldots\ x_{j-1}\ 0\ x_{j+1}\ \ldots\ x_m\ ]^T. \tag{2.31}$$

Wanneer men een vector y vooraan vermenigvuldigt met $G_{i,j}^*$ blijven de componenten $1, \ldots, i-1, i+1, \ldots, j-1, j+1, \ldots, m$ ongewijzigd, terwijl

$$(G_{i,j}^* y)_i = \bar{c}\, y_i + \bar{s}\, y_j$$

en

$$(G_{i,j}^* y)_j = -s\, y_i + c\, y_j.$$

Wanneer men een matrix A vooraan vermenigvuldigt met $G_{i,j}^*$ blijven de rijen $1, \ldots, i-1, i+1, \ldots, j-1, j+1, \ldots, m$ ongewijzigd.
Rij i van $G_{i,j}^* A = (\bar{c} \times \text{rij } i \text{ van } A) + (\bar{s} \times \text{rij } j \text{ van } A)$. Rij j van $G_{i,j}^* A = (-s \times \text{rij } i \text{ van } A)\ + (c \times \text{rij } j \text{ van } A)$.

Wanneer men een matrix A achteraan vermenigvuldigt met $G_{i,j}$ blijven de kolommen

$1, \ldots, i-1, i+1, \ldots, j-1, j+1, \ldots, m$ ongewijzigd. Kolom i van $AG_{i,j} = (c \times$ kolom i van $A) + (s \times$ kolom j van $A)$. Kolom j van $AG_{i,j} = (-\bar{s} \times$ kolom i van $A) + (\bar{c} \times$ kolom j van $A)$.

Men kan Givens-rotaties dus gebruiken om op een bepaalde plaats in een matrix een 0 te creëren door vooraan te vermenigvuldigen met een Givens-matrix. Stel bijvoorbeeld dat we het element in A op plaats (j,k) willen 0 maken. Kies dan een element in dezelfde kolom, bijvoorbeeld op plaats (i,k), en maak $G_{i,j}$ met

$$c = \frac{a_{ik}}{\sqrt{|a_{ik}|^2 + |a_{jk}|^2}}, \quad s = \frac{a_{jk}}{\sqrt{|a_{ik}|^2 + |a_{jk}|^2}}, \tag{2.32}$$

dan heeft $G_{i,j}^* A$ een 0 op plaats (j,k).

Vertrekkende van een matrix $A \in \mathbb{C}^{m \times n}$ kan men door vooraan te vermenigvuldigen met Givens-rotaties matrix A omvormen tot een bovendriehoeksmatrix R. Voor een (4×3) matrix kan dit als volgt worden gevisualiseerd, waarbij elementen die in een stap aangepast worden in het vet worden weergegeven.

$$A = \begin{bmatrix} x & x & x \\ x & x & x \\ x & x & x \\ x & x & x \end{bmatrix} \underset{\rightarrow}{\left(G_{3,4}^{(1)}\right)^*} \begin{bmatrix} x & x & x \\ x & x & x \\ \mathbf{x} & \mathbf{x} & \mathbf{x} \\ 0 & \mathbf{x} & \mathbf{x} \end{bmatrix} \underset{\rightarrow}{\left(G_{2,3}^{(1)}\right)^*} \begin{bmatrix} x & x & x \\ \mathbf{x} & \mathbf{x} & \mathbf{x} \\ 0 & \mathbf{x} & \mathbf{x} \\ 0 & x & x \end{bmatrix}$$

$$\underset{\rightarrow}{\left(G_{1,2}^{(1)}\right)^*} \begin{bmatrix} \mathbf{x} & \mathbf{x} & \mathbf{x} \\ 0 & \mathbf{x} & \mathbf{x} \\ 0 & x & x \\ 0 & x & x \end{bmatrix} \underset{\rightarrow}{\left(G_{3,4}^{(2)}\right)^*} \begin{bmatrix} x & x & x \\ 0 & x & x \\ 0 & \mathbf{x} & \mathbf{x} \\ 0 & 0 & \mathbf{x} \end{bmatrix} \underset{\rightarrow}{\left(G_{2,3}^{(2)}\right)^*} \begin{bmatrix} x & x & x \\ 0 & \mathbf{x} & \mathbf{x} \\ 0 & 0 & \mathbf{x} \\ 0 & 0 & x \end{bmatrix}$$

$$\underset{\rightarrow}{\left(G_{3,4}^{(3)}\right)^*} \begin{bmatrix} x & x & x \\ 0 & x & x \\ 0 & 0 & \mathbf{x} \\ 0 & 0 & 0 \end{bmatrix} := R.$$

Dus er geldt

$$\left(G_{3,4}^{(3)}\right)^* \left(G_{2,3}^{(2)}\right)^* \left(G_{3,4}^{(2)}\right)^* \left(G_{1,2}^{(1)}\right)^* \left(G_{2,3}^{(1)}\right)^* \left(G_{3,4}^{(1)}\right)^* A = R$$

of, aangezien alle matrices $G_{i,j}$ unitair zijn, volgt

$$A = \left(G_{3,4}^{(1)} G_{2,3}^{(1)} G_{1,2}^{(1)} G_{3,4}^{(2)} G_{2,3}^{(2)} G_{3,4}^{(3)}\right) R := QR.$$

Als we bovenstaande redenering in pseudocode zetten, komen we tot Algoritme 3. We maken hierbij nog twee opmerkingen.

Algoritme 3 QR-factorisatie met Givens-rotaties

1: $Q = I, R = A$
2: **for** $j = 1$ to n **do**
3: **for** $i = m$ downto $j + 1$ **do**
4: $c = \frac{r_{i-1\,j}}{\sqrt{|r_{i-1\,j}|^2+|r_{ij}|^2}}, \quad s = \frac{r_{ij}}{\sqrt{|r_{i-1\,j}|^2+|r_{ij}|^2}}$
5: $r_{ij} = 0,\ r_{i-1\,j} = \sqrt{|r_{i-1\,j}|^2 + |r_{ij}|^2}$
6: **for** $k = j + 1$ to n **do**
7: $\begin{bmatrix} r_{i-1\,k} \\ r_{ik} \end{bmatrix} = \begin{bmatrix} \bar{c} & \bar{s} \\ -s & c \end{bmatrix} \begin{bmatrix} r_{i-1\,k} \\ r_{ik} \end{bmatrix}$
8: **end for**
9: **for** $k = 1$ to m **do**
10: $[\ q_{k\,i-1}\ q_{ki}] = [\ q_{k\,i-1}\ q_{ki}] \begin{bmatrix} c & -\bar{s} \\ s & \bar{c} \end{bmatrix}$
11: **end for**
12: **end for**
13: **end for**

- Stappen 6-8 en stappen 9-11 kunnen geschreven (en geïmplementeerd) worden als operaties op rijen, respectievelijk kolommen.
- In Algoritme 3 wordt Q expliciet uitgerekend. Deze matrix kan ook in een gefactoriseerde vorm bewaard worden, waarbij enkel de opeenvolgende waarden van c en s opgeslagen worden.

Voorbeeld 2.7 *We beschouwen de reële matrix*

$$A = \begin{bmatrix} 1 & -2 & 3 \\ 1 & -3 & 5 \\ 2 & 3 & 1 \\ -3 & 4 & 2 \end{bmatrix}. \tag{2.33}$$

Alle bewerkingen gebeuren nu met reële getallen, en de unitaire transformaties herleiden zich tot orthogonale transformaties. We creëren één voor één nullen, als volgt.

$$A \xrightarrow{G_{3,4}^{(1)^T}} \begin{bmatrix} 1.0000 & -2.0000 & 3.0000 \\ 1.0000 & -3.0000 & 5.0000 \\ 3.6056 & -1.6641 & -1.1094 \\ 0 & 4.7150 & 1.9415 \end{bmatrix}$$

$$\xrightarrow{G_{2,3}^{(1)^T}} \begin{bmatrix} 1.0000 & -2.0000 & 3.0000 \\ 3.7417 & -2.4054 & 0.2673 \\ 0 & 2.4461 & -5.1146 \\ 0 & 4.7150 & 1.9415 \end{bmatrix} \xrightarrow{G_{1,2}^{(1)^T}} \begin{bmatrix} 3.8730 & -2.8402 & 1.0328 \\ 0 & 1.3111 & -2.8293 \\ 0 & 2.4461 & -5.1146 \\ 0 & 4.7150 & 1.9415 \end{bmatrix}$$

$$G_{3,4}^{(2)^T} \to \begin{bmatrix} 3.8730 & -2.8402 & 1.0328 \\ 0 & 1.3111 & -2.8293 \\ 0 & 5.3117 & -0.6320 \\ 0 & 0 & 5.4341 \end{bmatrix} \quad G_{2,3}^{(2)^T} \to \begin{bmatrix} 3.8730 & -2.8402 & 1.0328 \\ 0 & 5.4711 & -1.2916 \\ 0 & 0 & 2.5954 \\ 0 & 0 & 5.4341 \end{bmatrix}$$

$$G_{3,4}^{(3)^T} \to \begin{bmatrix} 3.8730 & -2.8402 & 1.0328 \\ 0 & 5.4711 & -1.2916 \\ 0 & 0 & 6.0220 \\ 0 & 0 & 0 \end{bmatrix} \tag{2.34}$$

Tenslotte vermelden we nog een aantal kenmerken.

- Algoritme 3 berekent inherent een *volledige* QR factorisatie, Q wordt opgebouwd als product van (vierkante) unitaire matrices.
- Algoritme 3 zonder stappen 9-11 (enkel R gewenst) vraagt $\sim (3mn^2 - n^3)$ elementaire bewerkingen, inclusief stappen 9-11 vraagt het

  $$\sim (6m^2n - n^3) \tag{2.35}$$

  bewerkingen[6]. Om dit verifiëren zijn volgende reeksen nuttig:

  $$1 + 2 + \cdots + n = \tfrac{n(n+1)}{2} \sim \tfrac{1}{2}n^2,$$
  $$1^2 + 2^2 + \cdots + n^2 = \tfrac{n(n+1)(2n+1)}{6} \sim \tfrac{1}{3}n^3.$$

- Zoals bij een achterwaarts stabiel algoritme komen de berekende Q en R overeen met een factorisatie van een perturbatie van A met relatieve fout van ordegrootte de machineprecisie. Bovendien is ook $Q^*Q \approx I$, met fout van ordegrootte van de machineprecisie. De sleutel tot de stabiliteit is dat het algoritme bestaat uit een aaneenschakeling van unitaire transformaties, gekenmerkt door $\kappa(G_{i,j}) = 1$.
- Algoritme 3 blijft ook toepasbaar wanneer A niet van volle rang is of zeer kleine singuliere waarden heeft (in dit laatste geval zeggen we dat A *numeriek* van lage rang is).

2.3.4 QR-factorisatie met Householder-transformaties

We halen opnieuw inspiratie uit afbeeldingen in een vlak, zoals in de inleiding van §2.3.3. Naast een rotatie is er nog een tweede type orthogonale transformatie, namelijk een spiegeling[7]. Hiermee kan ook een nul gecreëerd worden. Laten we terugkomen op Voorbeeld 2.6.

[6] We beschouwen enkel de termen met de hoogste totale graad in m en n. Verder veronderstellen we dat we een algoritme hebben voor het evalueren van de functie $\mathbb{R} \ni x \mapsto \sqrt{x}$ waarvan het aantal bewerkingen begrensd wordt door een constante onafhankelijk van x.

[7] Men kan bewijzen dat in $\mathbb{R}^n$ elke orthogonale transformatie ontbonden kan worden in een opeenvolging van elementaire rotaties en spiegelingen.

Voorbeeld 2.8 *Ook orthogonale matrix*

$$F = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$$

beeldt $x = [0\ 1]^T$ *af op* $[1\ 0]^T$*. De matrix heeft eigenwaarde* 1*, met eigenvector* $V_1 = [1\ 1]^T$ *en eigenwaarde* -1*, met eigenvector* $V_2 = [1\ -1]^T$*. Vector* V_1 *beschrijft de as ten opzichte waarvan er gespiegeld wordt, vector* V_2 *bepaalt de richting.*

We kunnen de methodologie opnieuw veralgemenen naar de vectorruimte $\mathbb{C}^m$. Beschouw voor een vector $v \in \mathbb{C}^m$ de projector op het orthogonaal complement van $< v >$ (voor een reële driedimensionale vectorruimte bijvoorbeeld is dit het vlak waarvan v de normaal is). Deze projector wordt gegeven door

$$P_{<v>^\perp} = I - \frac{vv^*}{v^*v},$$

zie (2.14). De reflector of spiegeling $R_{<v>^\perp}$ ten opzichte van het orthogonaal complement van $< v >$, en in de richting van v, beeldt een vector x af op $x + 2(P_{<v>^\perp}x - x)$, dus

$$R_{<v>^\perp} = I - 2\frac{vv^*}{v^*v}. \tag{2.36}$$

Een grafische interpretatie wordt gegeven in Figuur 2.3. Er kan gemakkelijk nagegaan worden dat matrix $R_{<v>^\perp}$ unitair is.

We construeren nu een reflector om nullen te creëren in een vector (en in latere instantie in een matrix). In tegenstelling tot Givens-rotaties is het mogelijk om tegelijkertijd meerdere nullen te creëren. Laten we, gegeven een vector $x \in \mathbb{C}^m$, een reflector zoeken die de laatste $m-1$ componenten van x op nul afbeeldt, wat uitgedrukt kan worden als

$$R_{<v>^\perp}\, x = \pm\|x\|_2 \mathbf{e}_1, \tag{2.37}$$

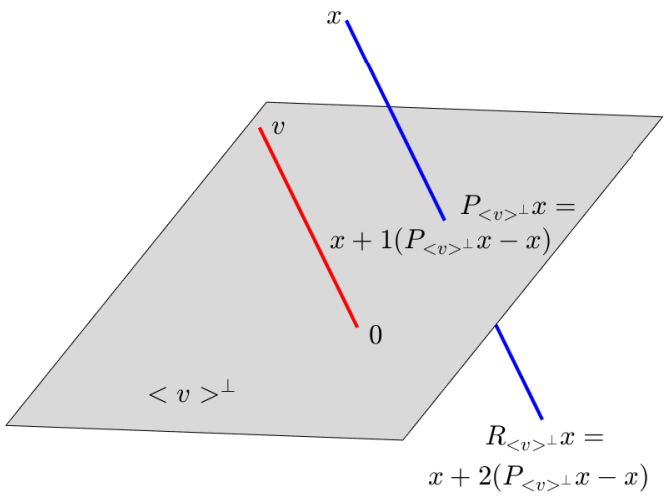
met $\mathbf{e}_1 = [1\ 0 \cdots\ 0]^T$ de eerste eenheidsvector. De richting v van de reflectie is deze van $x - R_{<v>^\perp}\, x$, wat leidt tot

$$v = x \mp \|x\|_2 \mathbf{e}_1. \tag{2.38}$$

Er kan geverifieerd worden dat er met beide keuzes voor v voldaan is aan (2.37).
Tenslotte leggen we het teken in (2.38) vast. Omwille van numerieke stabiliteit (vermijden van een 'gevaarlijke aftrekking') kiezen we het teken zodat $\|v\|_2$ maximaal wordt. Definiëren we hiertoe de tekenfunctie door $\text{sign}(x) = 1$ voor $x \geq 0$ en $\text{sign}(x) = -1$ voor $x < 0$, brengt dit ons uiteindelijk tot de unitaire transformatie

$$F_m := I_m - 2\frac{vv^*}{v^*v}, \quad \text{met} \quad v = x + \text{sign}(x_1)\|x\|_2 \mathbf{e}_1, \tag{2.39}$$

die bekend staat als een *Householder-transformatie*. Merk dat $F_m = F_m^*$ zodat $F_m^{-1} = F_m$, wat ook consistent is met de meetkundige interpretatie.



Figuur 2.3: Grafische interpretatie van een reflectie in $\mathbb{R}^3$.

Laten we nu de reflectoren gebruiken in een algoritme om een QR-factorisatie uit te rekenen. Vertrekkende van matrix A kunnen we kolom per kolom nullen onder de diagonaal maken door transformaties beschreven door $Q_{1,m}, \ldots, Q_{n,m}$ langs links toe te passen, waarbij $Q^*_{1,m} = F_m$,

$$Q^*_{2,m} = \begin{bmatrix} 1 & \\ & F_{m-1} \end{bmatrix}, \quad Q^*_{3,m} = \begin{bmatrix} I_2 & \\ & F_{m-2} \end{bmatrix}, \cdots, \quad Q^*_{n,m} = \begin{bmatrix} I_{n-1} & \\ & F_{m-n+1} \end{bmatrix}.$$

Bij een 4-bij 3 matrix is bijvoorbeeld de flow van bewerkingen om tot een bovendriehoeksmatrix te komen als volgt:

$$A = \begin{bmatrix} x & x & x \\ x & x & x \\ x & x & x \\ x & x & x \end{bmatrix} \overset{Q^*_{1,4}}{\rightarrow} \begin{bmatrix} \mathbf{x} & \mathbf{x} & \mathbf{x} \\ 0 & \mathbf{x} & \mathbf{x} \\ 0 & \mathbf{x} & \mathbf{x} \\ 0 & \mathbf{x} & \mathbf{x} \end{bmatrix} \overset{Q^*_{2,4}}{\rightarrow} \begin{bmatrix} x & x & x \\ 0 & \mathbf{x} & \mathbf{x} \\ 0 & 0 & \mathbf{x} \\ 0 & 0 & \mathbf{x} \end{bmatrix} \overset{Q^*_{3,4}}{\rightarrow} \begin{bmatrix} x & x & x \\ 0 & x & x \\ 0 & 0 & \mathbf{x} \\ 0 & 0 & 0 \end{bmatrix}.$$

We kunnen dit samenvatten als

$$Q^*_{3,4}Q^*_{2,4}Q^*_{1,4}A = R \ \rightarrow A = (Q_{1,4}Q_{2,4}Q_{3,4})R := QR.$$

Voorbeeld 2.9 *Vertrekkende van matrix (2.33) bepalen we in de eerste stap*

$$x = a_1,\ v = x + \text{sign}(x_1)\|x\|_2\,\mathbf{e}_1 = x + \sqrt{15}\begin{bmatrix}1\\0\\0\\0\end{bmatrix} = \begin{bmatrix}4.8730\\1.0000\\2.0000\\-3.0000\end{bmatrix},$$

zodat

$$Q_{1,4}^*\,a_1 = \begin{bmatrix}-3.8730\\0\\0\\0\end{bmatrix} = -\text{sign}(x_1)\ \|x\|_2\,\mathbf{e}_1.$$

Het effect op matrix A wordt beschreven door

$$A \overset{Q_{1,4}^*}{\rightarrow} \begin{bmatrix}-3.8730 & 2.8402 & -1.0328\\ 0 & -2.0067 & 4.1724\\ 0 & 4.9865 & -0.6552\\ 0 & 1.0202 & 4.4827\end{bmatrix} \overset{Q_{2,4}^*}{\rightarrow} \begin{bmatrix}-3.8730 & 2.8402 & -1.0328\\ 0 & 5.4711 & -1.2916\\ 0 & 0 & 2.9885\\ 0 & 0 & 5.2282\end{bmatrix}$$

$$\overset{Q_{3,4}^*}{\rightarrow} \begin{bmatrix}-3.8730 & 2.8402 & -1.0328\\ 0 & 5.4711 & -1.2916\\ 0 & 0 & -6.0220\\ 0 & 0 & 0\end{bmatrix}.$$

In vergelijking met (2.34) hebben de eerste en derde rij van de R-factor een ander teken (alsook de overeenkomstige kolommen van de niet weergegeven Q-factor). Dit teken wordt bepaald door de gemaakte keuze in (2.39).

Bij het implementeren van het algoritme is het om redenen van efficiëntie belangrijk op te merken dat het product tussen F_m en een vector q kan uitgerekend worden als

$$F_m\ q = q - 2\frac{(v^*q)}{v^*v}v. \tag{2.40}$$

Merk dat we enkel v nodig hebben en F_m niet expliciet moeten vormen. Dit is niet alleen van belang bij het berekenen van factor R, maar ook bij het bepalen van Q. Indien enkel matrix-vector producten met Q nodig zijn, kunnen we het product Qw, met $w \in \mathbb{C}^m$ berekenen als

$$\begin{aligned} Qw &= Q_{1,m}\ \ldots Q_{n-1,m}Q_{n,m}\ w \\ &= \left(F_m \begin{bmatrix}1 & \\ & F_{m-1}\end{bmatrix} \cdots \left(\begin{bmatrix}I_{n-2} & \\ & F_{m-n+2}\end{bmatrix}\left(\begin{bmatrix}I_{n-1} & \\ & F_{m-n+1}\end{bmatrix} w\right)\right)\cdots\right), \end{aligned}$$

waarbij we dan telkens gebruik kunnen maken van (2.40). Indien Q in expliciete vorm gevraagd wordt, kunnen we bijvoorbeeld Qw voor $w \in \{\mathbf{e}_1, \ldots, \mathbf{e}_m\}$ uitrekenen, met $\mathbf{e}_j$ de j-de eenheidsvector. Met de notatie $A_{i:j,k}$ voor de vector bestaande uit het i-de tot j-de

Algoritme 4 QR-factorisatie met Householder-transformaties

1: $R = A$
2: **for** $j = 1$ to n **do**
3: $\quad x = R_{j:m,j}$
4: $\quad v_j = x + \text{sign}(x_1)\|x\|_2\ \mathbf{e}_1$
5: $\quad v_j = v_j/\|v_j\|_2$
6: $\quad R_{jj} = -\text{sign}(x_1)\|x\|_2,\ R_{j+1:m,j} = 0$
7: $\quad$ **for** $k = j + 1$ to n **do**
8: $\quad\quad R_{j:m,k} = R_{j:m,k} - 2\left(v_j^*\ R_{j:m,k}\right) v_j$
9: $\quad$ **end for**
10: **end for**
11: **for** $i = 1$ to m **do**
12: $\quad w = \mathbf{e}_i$
13: $\quad$ **for** $k = n$ downto 1 **do**
14: $\quad\quad w_{k:m} = w_{k:m} - 2\left(v_k^*\ w_{k:m}\right) v_k$
15: $\quad$ **end for**
16: $\quad Q_{1:m,i} = w$
17: **end for**

element van kolom k van matrix A en $w_{i:v}$ de vector bestaande uit het i-de tot j-de element van vector w, komen we op deze manier tot Algoritme 4.
Om enkel R te berekenen (lijnen 1-10 van Algoritme 4) voldoet het aantal elementaire bewerking aan $2mn^2 - \frac{2}{3}n^2$. Om zowel R als Q te bepalen wordt dit

$$4m^2n - \frac{2}{3}n^3. \tag{2.41}$$

Een vergelijking met (2.35) leert dat het gelijktijdig creëren van meerdere nullen per kolom een significante reductie van het rekenwerk ten opzichte van Algoritme 3 oplevert. Onder meer daardoor wordt bij volle matrices zonder specifieke structuur Algoritme 4 geprefereerd. Bovendien is bij het bepalen van (2.41) nog niet in rekening gebracht dat de structuur van w op lijn 12 uitgebuit kan worden. Voor $w = \mathbf{e}_1$ bijvoorbeeld moet de lus 13-15 enkel voor $k = 1$ uitgevoerd worden.

Opmerking 2.2 *Bij de algoritmen voor de QR-factorisatie gebaseerd op unitaire transformaties (algoritmen 3-4) vertrekken we van matrix A en vormen deze geleidelijk om tot een (pseudo)driehoeksmatrix R door vermenigvuldiging langs links, wat neerkomt op het uitvoeren van operaties op rijen. We bouwen matrix Q^* op zodat $Q^*A = R$. Bij Gram-Schmidt gebaseerde algoritmen (algoritmen 1-2) daarentegen vormen we A geleidelijk om naar matrix $\hat{Q}$ door bewerkingen op kolommen uit te voeren, wat je kan interpreteren als het vermenigvuldigen langs rechts met een matrix. We construeren dus impliciet $\hat{R}$ zodat $A\hat{R}^{-1} = \hat{Q}$ (wat ook de numerieke problemen verklaart als A numeriek niet van volle kolomrang is).*

2.4 Beste benadering van een vector in een deelruimte

We beschouwen een deelruimte $\mathcal{D}$ in $\mathbb{C}^m$, opgespannen door vectoren $a_1, \ldots, a_n$, in formule

$$\mathcal{D} =< a_1, \ldots, a_n > .$$

We veronderstellen $m > n$ en bestuderen de volgende vraag:
Zoek de beste benadering voor een vector $b \in \mathbb{C}^m$ in de ruimte $\mathcal{D}$ volgens de 2-norm.

Stellen we $A = [a_1 \ldots a_n]$ en maken we gebruik van de eigenschap $\mathcal{R}(A) = \mathcal{D}$, dan kunnen we dit probleem ook op de volgende manier beschrijven: bepaal $\hat{x} \in \mathbb{C}^n$ zodat het minimum

$$\min_{x \in \mathbb{C}^n} \|Ax - b\|_2 \tag{2.42}$$

bereikt wordt voor $x = \hat{x}$. Vector $\hat{y} = A\hat{x}$ is dan een beste benadering van b.
Het zoeken naar $\hat{x}$ wordt het *kleinste-kwadratenprobleem genoemd* en komen we onder meer tegen in de context van het vereffenen van meetgegevens en wiskundige modellering.

Voorbeeld 2.10 *Stel dat we een aantal meetwaarden (x_i, y_i) , $i = 1, 2, \ldots, m$, hebben en we willen de achterliggende continue functie vinden of benaderen in de vorm van een lineaire combinatie van gegeven basisfuncties $\{g_1, \ldots, g_n\}$. We zoeken met andere woorden coëfficiënten $\{c_1, \ldots, c_n\}$ zodat*

$$f(x) \approx V(x) := c_1 g_1(x) + c_2 g_2(x) + \cdots + c_n g_n(x). \tag{2.43}$$

Voor $m > n$ is de interpolatie-eis $V(x_i) = y_i$, $i = 1, \ldots, m$, te streng. We zwakken hem af tot $V(x_i) \approx y_i$, wat we in matrixvorm kunnen schrijven als

$$\underbrace{\begin{bmatrix} g_1(x_1) & g_2(x_1) & \cdots & g_n(x_1) \\ g_1(x_2) & g_2(x_2) & \cdots & g_n(x_2) \\ \vdots & & & \vdots \\ g_1(x_m) & g_2(x_m) & \cdots & g_n(x_m) \end{bmatrix}}_{A \in \mathbb{R}^{m \times n}} \underbrace{\begin{bmatrix} c_1 \\ c_2 \\ \vdots \\ c_n \end{bmatrix}}_{x} \approx \underbrace{\begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_m \end{bmatrix}}_{b} .$$

Een manier om het overgedetermineerde stelsel $Ax = b$ benaderend op te lossen bestaat uit het zoeken naar $x \in \mathbb{R}^n$ zodat de doelfunctie in (2.42) minimaal wordt. Dit heeft een geometrische interpretatie omdat

$$\|Ax - b\|_2^2 = \sum_{i=1}^{n} \left(y_i - (c_1 g_1(x_i) + c_2 g_2(x_i) + \cdots + c_n g_n(x_i))\right)^2 = \sum_{i=1}^{n} \left(y_i - V(x_i)\right)^2 .$$

In woorden, we minimaliseren de som van de kwadraten van de afwijkingen tussen meetwaarden y_i en de voorspelling ervan op basis van het vooropgesteld model (bepaald door V). Vandaar de naam kleinste kwadraten *...*

2.4.1 Karakterisering door het normaalstelsel

We beginnen met een stelling.

STELLING 2.4.1
Vector $\hat{y}$ is een beste benadering in deelruimte $\mathcal{D}$ voor b als $b-\hat{y}$ orthogonaal is ten opzichte van $\mathcal{D}$.

Bewijs Neem een willekeurige $y \in \mathcal{D}$. Vermits $y - \hat{y}$ in $\mathcal{D}$ ligt en dus orthogonaal is ten opzichte van $b - \hat{y}$ geldt, volgens de stelling van Pythagoras,

$$\|b - y\|_2^2 = \|b - \hat{y}\|_2^2 + \|y - \hat{y}\|_2^2 \geq \|b - \hat{y}\|_2^2, \tag{2.44}$$

dus y benadert b niet beter dan $\hat{y}$. □

Omwille van het volgende resultaat kunnen we in wat volgt spreken van *de* beste benadering.

STELLING 2.4.2 *(orthogonale projectiestelling)*
De beste benadering in deelruimte $\mathcal{D}$ voor vector b bestaat en is uniek. Ze wordt gegeven door

$$\hat{y} = P_{\mathcal{D}}\, b, \tag{2.45}$$

met $P_{\mathcal{D}}$ de orthogonale projector op $\mathcal{D}$.

Bewijs Zoals we gezien hebben in Sectie 2.2 kan b ontbonden worden als

$$b = \underbrace{P_{\mathcal{D}}\, b}_{\in \mathcal{R}(P_{\mathcal{D}})} + \underbrace{(I - P_{\mathcal{D}})\, b}_{\in \mathcal{N}(P_{\mathcal{D}})},$$

waaruit volgt dat $b - \hat{y} = b - P_{\mathcal{D}}\, b \in \mathcal{N}(P_{\mathcal{D}}) = \mathcal{R}(P_{\mathcal{D}^\perp})$. Volgens Stelling 2.4.1 is $\hat{y}$ een beste benadering. De uniciteit volgt uit ongelijkheid (2.44). □

Laten we nu vector $\hat{x}$ bepalen, die gekarakteriseerd wordt als oplossing van een stelsel met als matrix de grammatrix $G = A^*A$, horende bij de basis $\{a_1, \ldots, a_n\}$.

STELLING 2.4.3 *(normaalstelsel)*
Vector $\hat{x}$ is een oplossing van (2.42) als en alleen als $x = \hat{x}$ voldoet aan het zgn. normaalstelsel

$$A^*A\, x = A^*b. \tag{2.46}$$

Bewijs De voorwaarde is nodig. Veronderstel dat $A\hat{x} = \hat{y}$, met $\hat{y}$ gegeven door (2.45). We weten dat $(b - A\hat{x}) \perp \mathcal{D} =< a_1 \ldots, a_n >$ waaruit volgt

$$(a_i, (b - A\hat{x})) = 0, \Leftrightarrow a_i^*(b - A\hat{x}) = 0,\ i = 1, \ldots, n.$$

In matrix vorm geschreven drukt dit uit dat $\hat{x}$ voldoet aan (2.46).
De voorwaarde is voldoende. Gelijkheid (2.46) kan geschreven worden als $A^*(A\hat{x} - b)$, wat $(A\hat{x} - b) \perp \mathcal{D}$ impliceert. Uit vorige twee stellingen volgt dat $A\hat{x} = \hat{y}$. □

Indien vectoren $\{a_1, \ldots, a_n\}$ een basis vormen van $\mathcal{D}$, of, equivalent daarmee, indien matrix A van volle kolomrang is, dan is matrix A^*A inverteerbaar. We kunnen dan de oplossing van (2.46) in expliciete vorm schrijven als

$$\hat{x} = (A^*A)^{-1}A^*b = A^+b, \tag{2.47}$$

waaruit volgt

$$\hat{y} = A\hat{x} = A(A^*A)^{-1}A^*\, b,$$

wat consistent is met (2.45) en (2.13). Als A niet van volle rang is, dan hebben vergelijkingen (2.46) oneindig veel oplossingen. We komen hierop terug op het einde van Sectie 2.4.2.
Indien de vectoren $\{a_1, \ldots, a_n\}$ orthogonaal zijn, wordt het normaalstelsel diagonaal zodat

$$\hat{x}_j = \frac{a_j^*b}{a_j^*a_j},\ j = 1, \ldots, n,$$

en

$$\hat{y} = \sum_{j=1}^{n} \frac{a_j^*b}{a_j^*a_j}\, a_j. \tag{2.48}$$

We zijn deze formules reeds tegengekomen in Sectie 2.1. Toen werd $v \in \mathcal{D}$ verondersteld en leverden ze, toegepast op v, een (exacte) ontbinding van v op. Nu leggen we de voorwaarde $b \in \mathcal{D}$ niet op. Dezelfde formules toegepast op b geven dan de beste benadering van b weer volgens de 2-norm.
Tenslotte benadrukken we, aansluitend bij Sectie 2.1, een andere belangrijke eigenschap van orthogonale basissen. Als we de ruimte waarin we de beste benadering zoeken uitbreiden (en overeenkomstig een basisvector toevoegen) moeten we in (2.48) enkel een term toevoegen. De andere coëfficiënten blijven namelijk dezelfde. Dit laatste is *niet* het geval bij een scheve basis, zoals we nu illustreren met een voorbeeld.

Voorbeeld 2.11 *We herbekijken Voorbeeld 2.1 voor* $\eta = 0.1$ *en breiden de twee basissen uit tot*

$$a_1 = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix},\ a_2 = \begin{bmatrix} 1 \\ 0.1 \\ 0.1 \end{bmatrix}, a_3 = \begin{bmatrix} 1 \\ 0.1 \\ 0.3 \end{bmatrix}$$

en

$$q_1 = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix},\ q_2 = \begin{bmatrix} 0 \\ \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} \end{bmatrix},\ q_3 = \begin{bmatrix} 0 \\ \frac{1}{\sqrt{2}} \\ -\frac{1}{\sqrt{2}} \end{bmatrix}.$$

	$P_{<a_1,\ldots,a_i>}\, b$	*scheve basis*	*orthogonale basis*
$i=1$	$\begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}$	$c_1^{(1)} = 1$	$d_1^{(1)} = 1$
$i=2$	$\begin{bmatrix} 1 \\ 1.5 \\ 1.5 \end{bmatrix}$	$c_1^{(2)} = -14$ $c_2^{(2)} = 15$	$d_1^{(2)} = 1$ $d_2^{(2)} = \frac{3}{\sqrt{2}}$
$i=3$	$\begin{bmatrix} 1 \\ 1 \\ 2 \end{bmatrix}$	$c_1^{(3)} = -9$ $c_2^{(3)} = 5$ $c_3^{(3)} = 5$	$d_1^{(3)} = 1$ $d_2^{(3)} = \frac{3}{\sqrt{2}}$ $d_3^{(3)} = -\frac{1}{\sqrt{2}}$

Tabel 2.2: Ontbindingen van (beste benaderingen van) $b = [1\ 1\ 2]^T$in orthogonale en scheve basissen.

Merk dat $< a_1 >=< q_1 >$, $< a_1, a_2 >=< q_1, q_2 >$ *en uiteraard* $< a_1, a_2, a_3 >=< q_1, q_2, q_2 >$. *Basis* $\{q_1, q_2, q_3\}$ *is een orthonormale basis van* $\mathbb{C}^3$, *basis* $\{a_1, \ldots, a_3\}$ *een scheve basis. We berekenen de beste benadering van* $b = [1\ 1\ 2]^T$ *in de ruimte* $< a_1, \ldots, a_i >$, *die we ontbinden als*

$$\mathcal{P}_{<a_1,\ldots,a_i>} b = c_1^{(i)} a_1 + \ldots + c_i^{(i)} a_i = d_1^{(i)} q_1 + \ldots + d_i^{(i)} q_i,\ i \in \{1, 2, 3\}.$$

De resultaten worden weergegeven in Tabel 2.2

2.4.2 Oplossing via orthogonale transformaties

Het rechtstreeks oplossen van de normaalvergelijkingen wordt best vermeden omdat $\kappa(A^*A) = \kappa(A)^2$, zie ook Tabel 2.1. We komen hier nog verder op terug in §2.4.3. We presenteren eerst andere methodes, gebaseerd op een unitaire transformatie.
We vertrekken van de volledige QR-factorisatie van A, namelijk $A = QR$, waarbij $Q \in \mathbb{C}^{m\times m}$ unitair is en $R \in \mathbb{C}^{m\times n}$ een pseudo-driehoeksmatrix,

$$R = \begin{bmatrix} \hat{R} \\ 0 \end{bmatrix},$$

met $\hat{R} \in \mathbb{R}^{n\times n}$ een bovendriehoeksmatrix. Merk dat $\hat{R}$ inverteerbaar is als en alleen als A van volle kolomrang is. Vermits een unitaire transformatie de 2-norm bewaart, geldt

$$\|b - Ax\|_2 = \|b - QRx\|_2 = \|Q^*(b - QRx)\|_2 = \|Q^*b - Rx\|_2. \tag{2.49}$$

Stellen we

$$Q^*b = \begin{bmatrix} b_1 \\ b_2 \end{bmatrix},$$

met $b_1 \in \mathbb{C}^n$ en $b_2 \in \mathbb{C}^{m-n}$ dan verkrijgen we

$$\min_{x\in\mathbb{C}^n} \|b - Ax\|_2 = \min_{x\in\mathbb{C}^n} \left\| \begin{bmatrix} b_1 \\ b_2 \end{bmatrix} - \begin{bmatrix} \hat{R} \\ 0 \end{bmatrix} x \right\|_2 = \min_{x\in\mathbb{C}^n} \left\| \begin{bmatrix} b_1 - \hat{R}x \\ b_2 \end{bmatrix} \right\|_2 . \tag{2.50}$$

Merk dat we enkel de eerste n elementen van de vector in de laatste uitdrukking kunnen beïnvloeden door keuze van x. De kleinste-kwadratenoplossing $\hat{x}$ kunnen we dus berekenen door het driehoekige stelsel

$$\hat{R}x = b_1 \tag{2.51}$$

op te lossen. De rekenkost, bovenop het bepalen van een QR-factorisatie van matrix A, bedraagt $\sim (mn)+ \sim O(n^2)$ elementaire bewerkingen.

Opmerking 2.3 *Hoewel we bij de afleiding en meer bepaald in (2.49) gebruik gemaakt hebben van de volledige QR-factorisatie, hebben we voor het berekenen van b_1 enkel de eerste n kolommen van Q nodig. Bijgevolg kunnen we in een implementatie gebruik maken van een onvolledige QR-factorisatie, $A = \hat{Q}\hat{R}$.*

Uit (2.50) kunnen we ook de minimale fout halen, namelijk $\min_{x\in\mathbb{C}^n} \|b - Ax\|_2 = \|b_2\|_2$. In de toepassing uit Voorbeeld 2.10 is $\|b_2\|_2$ een maatstaf voor de kwaliteit van de berekende kleinste-kwadratenbenadering.

Opmerking 2.4 *De oplossing via de QR-factorisatie van A en het oplossen van (2.51) kan ook als volgt geïnterpreteerd worden:*

1. *Vervang de basis $\{a_1, \ldots, a_n\}$ door een orthonormale basis $\{q_1, \ldots, q_n\}$. Overeenkomstig (2.48) wordt de beste benadering van b bepaald als $\sum_{i=1}^{n} (q_i^* b)\, q_i$. Het samenvoegen van de coëfficiënten in deze ontbinding, die we als coördinaten kunnen interpreteren, levert vector b_1 op.*

2. *Transformeer deze coördinaten terug naar de basis $\{a_1, \ldots, a_n\}$, door het stelsel (2.51) op te lossen.*

Bij de QR-factorisatie bepalen we een orthogonale basis van de kolomruimte van A. Indien A zeer slecht geconditioneerd is, kan het omwille van numerieke stabiliteit voordelig zijn om gebruik te maken van de singuliere-waardenontbinding, die een orthogonale basis definieert van zowel de kolom-als rijruimte. De prijs hiervoor is echter een significante toename van het rekenwerk. Concreet, vertrekken we van de ontbinding $A = U\Sigma V^*$, met $U \in \mathbb{C}^{m\times m}$ en $V \in \mathbb{C}^{n\times n}$ unitaire matrices, krijgen we

$$\|Ax - b\|_2 = \|(U\Sigma V^*)x - b\|_2 = \|\Sigma V^* x - U^* B\|_2 = \left\| \Sigma y - \hat{b} \right\|_2 ,$$

met $y = V^*x$ en $\hat{b} = U^*b := [\hat{b}_1 \ \cdots \ \hat{b}_m]^T$. Als A van volle kolomrang is, wordt de kleinste-kwadratenoplossing gegeven door

$$\hat{y}_i = \frac{\hat{b}_i}{\sigma_i},\ i = 1, \ldots, n,$$

waarna we $\hat{x}$ kunnen bepalen als $V\hat{y}$.

Opmerking 2.5 *Als A van rang $r < n$ is, dan is*

$$\|\Sigma y - \hat{b}\|_2 = \left\| \begin{bmatrix} \sigma_1 & & & & & & \\ & \ddots & & & & & \\ & & \sigma_r & & & & \\ & & & 0 & & & \\ & & & & \ddots & & \\ & & & & & 0 & \\ 0 & & & & & 0 & \\ \vdots & & & & & \vdots & \\ 0 & & & & & 0 & \end{bmatrix} \begin{bmatrix} y_1 \\ \vdots \\ y_n \end{bmatrix} - \begin{bmatrix} \hat{b}_1 \\ \vdots \\ \hat{b}_m \end{bmatrix} \right\|_2 .$$

Er zijn nu oneindig veel oplossingen die $\|Ax-b\|_2 = \left\|\Sigma y - \hat{b}\right\|_2$ minimaal maken, namelijk

$$y_1 = \frac{\hat{b}_1}{\sigma_1}, \ldots, y_r = \frac{\hat{b}_r}{\sigma_r},\ y_{r+1} \in \mathbb{C}, \ldots,\ y_n \in \mathbb{C}.$$

De oplossing met $y_{r+1} = \cdots = y_n = 0$ is dan degene met minimale 2-norm. In praktijk is het meestal zo dat er door afrondingsfouten toch kleine niet-nul singuliere waarden verschijnen. Als deze niet expliciet als perturbaties van nul geïnterpreteerd worden kan dat leiden tot numerieke problemen omdat $\frac{1}{\sigma_i}$ voor $i > r$ dan zeer groot zal zijn, en het bepalen van een drempelwaarde is niet evident. Echter, als in een bepaalde toepassing de rang van A op voorhand gekend is, dan weten we welke singuliere waarden we als nul moeten beschouwen.

2.4.3 Verdere reflectie rond conditie en stabiliteit

We kijken eerst in meer detail naar de conditie van het benaderingsprobleem, waarin we de effecten van relatieve fouten op gegevens A en b op de beste benadering $\hat{y} \in \mathcal{D}$ en de kleinste-kwadratenoplossing $\hat{x}$ nagaan, gemeten met de 2-norm. Belangrijke parameters daarbij zijn

$$\theta = \arccos\left(\frac{\|\hat{y}\|_2}{\|b\|_2}\right), \quad \eta = \frac{\|A\|_2\|\hat{x}\|_2}{\|A\hat{x}\|_2}.$$

Merk dat in de reële vectorruimte $\mathbb{R}^n$, parameter θ de hoek voorstelt tussen vectoren b en projectie $\hat{y}$, en meer algemeen, de hoek tussen b en de deelruimte $\mathcal{D}$. Parameter $\eta \in [1,\ \kappa(A)]$ drukt uit hoe klein $\|A\hat{x}\|_2/\|\hat{x}\|_2$ is ten opzichte van $\|A\|_2 = \max_x \|Ax\|_2/\|x\|_2$. In wat volgt worden conditiegetallen genoteerd met de letter K, waarbij de beschouwde in- en uitgang worden aangeduid in het sub- en superscript.

Beschouwen we vooreerst een perturbatie op b. Overeenkomstig (2.45) en uitdrukking (B.1) voor het conditiegetal van een matrix-vector vermenigvuldiging, krijgen we

$$K_b^{\hat{y}} = \|P_{\mathcal{D}}\|_2 \frac{\|b\|_2}{\|P_{\mathcal{D}}b\|_2} = \frac{\|b\|_2}{\|\hat{y}\|_2} = \frac{1}{\cos(\theta)}.$$

Op een gelijkaardige manier vinden we uit (2.47) dat

$$K_b^{\hat{x}} = \|A^+\|_2 \frac{\|b\|_2}{\|\hat{x}\|_2} = \kappa(A) \frac{\|b\|_2}{\|A\|_2 \|\hat{x}\|_2} = \frac{\kappa(A)}{\eta \cos(\theta)}.$$

Voor een perturbatie op A wordt in Hoofdstuk 18 van [TB97] afgeleid dat

$$K_A^{\hat{y}} \leq \frac{\kappa(A)}{\cos(\theta)}, \quad K_A^{\hat{x}} \leq \kappa(A) + \frac{\kappa(A)^2 \tan(\theta)}{\eta}. \tag{2.52}$$

Zoals bij stelsels lineaire vergelijkingen (waarbij $m = n$) speelt het conditiegetal van A een belangrijke rol in de voortplanting van fouten op de gegevens. Als b bijna orthogonaal is ten opzichte van de kolomruimte $\mathcal{D}$ van A, dan is de projectie $\hat{y} = P_{\mathcal{D}} b$ bijna de nulvector. Relatieve fouten op b, en ook A, worden dan sterk opgeblazen, wat consistent is met de factoren die afhangen van θ. Tenslotte, als η groot is, dan is in zekere zin $\hat{x}$ groot ten opzichte van $\hat{y}$, wat een gunstig effect heeft op de relatieve fout.

De algoritmen besproken in §2.4.2 zijn achterwaarts stabiel (op voorwaarde dat de matrix-factorisaties met een achterwaarts stabiel algoritme worden bepaald). Bijgevolg geven bovenstaande conditiegetallen, vermenigvuldigd met de machineprecisie ϵ_{mach}, een zeer goede indicatie van de te verwachten fout op de numerieke oplossing voor $\hat{x}$ en $\hat{y}$.

Beschouw nu ter vergelijking een algoritme gebaseerd op de normaalvergelijkingen, bestaande uit het vormen van A^*A, gevolgd door het oplossen van stelsel (2.46). Laten we voor deze laatste stap een achterwaarts stabiel algoritme gebruiken (bijvoorbeeld gebaseerd op de Cholesky-factorisatie). Dan kan de bekomen oplossing $\tilde{x}$ beschouwd worden als exacte oplossing van een geperturbeerd probleem,

$$(A^*A + \delta A)\tilde{x} = (A^*b + \delta b),$$

met $\|\delta A\|_2 / \|A^*A\|_2 = \mathcal{O}(\epsilon_{\text{mach}})$, $\|\delta b\|_2 / \|A^*b\|_2 = \mathcal{O}(\epsilon_{\text{mach}})$. Bijgevolg wordt grootte-orde

$$\frac{\|\tilde{x} - \hat{x}\|_2}{\|\hat{x}\|} = \mathcal{O}(\kappa(A^*A)\epsilon_{\text{mach}}) \tag{2.53}$$

verwacht, waarbij $\kappa(A^*A) = \kappa(A)^2$. Bovendien zitten er sowieso al fouten op de matrix van het stelsel, door het uitrekenen van A^*A in de eerste stap.

Een vergelijking van (2.53) met de conditiegetallen leert ons dat een algoritme gebaseerd op de normaalvergelijkingen *niet* achterwaarts stabiel is, tenzij in speciale gevallen waarbij het conditiegetal $\kappa(A)$ klein is of $\frac{\tan(\theta)}{\eta}$ voldoende groot is. Bij het vereffenen van meetgegevens, aangehaald in (2.10), betekent het bestaan van een goede fit dat θ klein is. De factor in (2.52) die afhangt van $\kappa(A)^2$ is dan in de regel niet dominant.

Hoofdstuk 3

Benaderen van functies

In vorig hoofdstuk beschouwden we benaderingsproblemen in $\mathbb{R}^n$ en $\mathbb{C}^n$. In dit hoofdstuk zullen we gelijkaardige problemen zeer algemeen en abstract formuleren. Zo'n abstracte aanpak heeft verschillende voordelen. Abstractie zorgt voor een grotere eenheid in de behandeling omdat men tot een synthese van onderwerpen komt die op het eerste gezicht weinig met elkaar te maken hebben. Abstractie zorgt voor een bondige formulering van stellingen en eigenschappen en leidt tot een diepgaander inzicht. Verder brengt een abstracte behandeling vaak essentiële en minimale voorwaarden voor een resultaat of methode aan het licht, wat dan weer de toepasbaarheid vergroot.
In Sectie 3.1 behandelen we metrische ruimten, waar we enkel het bestaan van een axiomatisch gedefinieerde afstandsfunctie opleggen. Vervolgens beperken we ons tot abstracte vectorruimten. In Sectie 3.2 leggen we op de vectorruimte het bestaan van een norm op, in Sectie 3.3 van een scalair product, beiden ook axiomatisch gedefinieerd, wat ons brengt tot een zgn. genormeerde of unitaire ruimte. Zoals we zullen zien, definieert een norm op een natuurlijke manier een afstand, en een scalair product op een natuurlijke manier een norm. In die zin leggen we doorheen het hoofdstuk steeds meer vereisten op aan de ruimte, tot we komen aan unitaire ruimten, waartoe we de benaderingstheorie en -algoritmen van Hoofdstuk 2 in grote mate kunnen veralgemenen. Een unificerend element doorheen de verschillende secties is het zoeken van een beste benadering in een deelverzameling of deelruimte.
Hoewel de resultaten van dit hoofdstuk toepasbaar zijn op willekeurige ruimten die aan bovenvermelde voorwaarden voldoen, zullen we ze voornamelijk toepassen op en illustreren aan de hand van functieruimten, als tegengewicht voor de klassieke vectorruimten uit vorig hoofdstuk. Een belangrijk probleem is bijvoorbeeld het benaderen van een gegeven functie door een lineaire combinatie van gekende basisfuncties. Deze oriëntatie wordt gereflecteerd in de misschien wat misleidende titel van het hoofdstuk.

3.1 Metrische ruimte en afstand

Het woord *ruimte* is op zichzelf in de wiskunde niet gedefinieerd. Het komt echter wel voor in samengestelde woorden zoals 'topologische ruimte' en 'projectieve ruimte'. Ruimte bete-

kent dan meestal een verzameling waarop een zekere structuur gedefinieerd is die zodanig is dat ze één of andere meetkundige interpretatie mogelijk maakt. Bij het werken in dergelijke ruimten maakt men dan ook veelvuldig gebruik van meetkundige terminologie. De elementen van zo'n ruimte noemt men doorgaans de *punten* van de ruimte. De structuur van een ruimte kan ook louter algebraïsch zijn, d.w.z. gedefinieerd door samenstellingswetten. Een voorbeeld van zo'n een ruimte is een vectorruimte, ook wel *lineaire* ruimte genoemd. Men kan ook een zogenaamde topologische structuur definiëren. Een typisch voorbeeld van een verzameling met een topologische structuur is een metrische ruimte.

3.1.1 Het begrip afstand

We kennen het begrip afstand uit de meetkunde. De afstand tussen twee reële getallen (of tussen twee punten op de getallenrechte) wordt meestal gedefinieerd als

$$\rho(x,y) \;=\; |x-y| \;. \tag{3.1}$$

In het vlak definieert men een afstand tussen twee punten $p_1(x_1,y_1)$ en $p_2(x_2,y_2)$ als

$$\rho(p_1,p_2) \;=\; \sqrt{(x_1-x_2)^2 \;+\; (y_1-y_2)^2} \;. \tag{3.2}$$

Om het begrip *afstand* bruikbaar te maken voor 'niet-meetkundige' ruimten, zoals bijvoorbeeld functieruimten, zal men er een algemene definitie van moeten geven. Die definitie zal een distillatie zijn van de voornaamste eigenschappen van wat men intuïtief onder afstand verstaat; het zal dus een axiomatische definitie zijn. De hierboven vermelde meetkundige afstanden zullen aan die definitie moeten voldoen. De definitie moet ook voldoende elementen bevatten zodat de belangrijkste eigenschappen waarin het meetkundig begrip afstand een rol speelt, afgeleid kunnen worden.

Definitie 3.1.1 *(afstand)*
Men zegt dat over een verzameling A een afstand *gedefinieerd is als er met elk paar elementen $x,y \in A$ een reëel getal $\rho(x,y)$ overeenstemt dat voldoet aan de volgende eigenschappen:*

$$\begin{array}{ll} 1. & \rho(x,y) \geq 0 \\ 2. & \rho(x,y) = 0 \quad \textit{als en slechts als } x = y \\ 3. & \rho(x,y) = \rho(y,x) \\ 4. & \rho(x,y) \leq \rho(z,x) \;+\; \rho(z,y) \end{array} \tag{3.3}$$

Een afstandsfunctie is bijgevolg een functionaal van de productverzameling $A \times A$ naar de verzameling $\mathbb{R}$ van de reële getallen ('oneindig' is geen reëel getal). De eerste twee eigenschappen zeggen dat de afstand een positief definiete functie is. De derde eigenschap zegt dat het een symmetrische functie is in x en y. De vierde eigenschap noemt men de driehoeksongelijkheid, zie Figuur 3.1.

3.1.2 Het begrip beste benadering

We beginnen deze paragraaf met enkele aanvullende naamgevingen en definities.

1. De verzameling A met een afstandsfunctie ρ noemt men een *metrische ruimte*. De afstandsfunctie noemt men ook wel de *metriek* van de ruimte. Op eenzelfde verzameling A kan men verschillende afstandsfuncties ρ definiëren. Wanneer dit het geval is, dan moet men een notatie gebruiken die dit onderscheid weergeeft. Men stelt de ruimte dan voor door (A, ρ), eventueel met een omschrijving voor ρ.

2. Wanneer $\rho(x, y) < \rho(x, z)$, dan zegt men dat het punt x *dichter* bij y ligt dan bij z.

3. Zij D een deelverzameling van een metrische ruimte (A, ρ). Dan definieert men de *afstand van het punt $x \in A$ tot de deelverzameling D* als

$$\rho(x, D) = \inf\{\rho(x, y) : \ y \in D \ \} \ . \tag{3.4}$$

DEFINITIE 3.1.2 *(beste benadering)*
Zij D een deelverzameling van een metrische ruimte (A, ρ). Een element $d \in D$ noemt men een beste benadering *van een gegeven element $x \in A$, als er geen enkel ander element van D dichter bij x gelegen is dan d.*

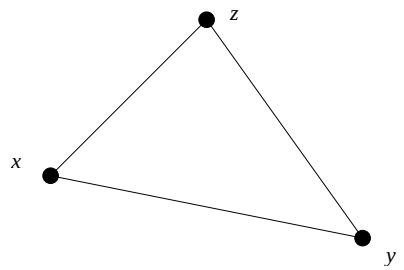
Uit deze definitie volgt dat de grootste ondergrens in (3.4) *bereikt* wordt in het element d,

$$\rho(x, D) = \min\{\rho(x, y) : \ y \in D \ \} = \rho(x, d) \ . \tag{3.5}$$

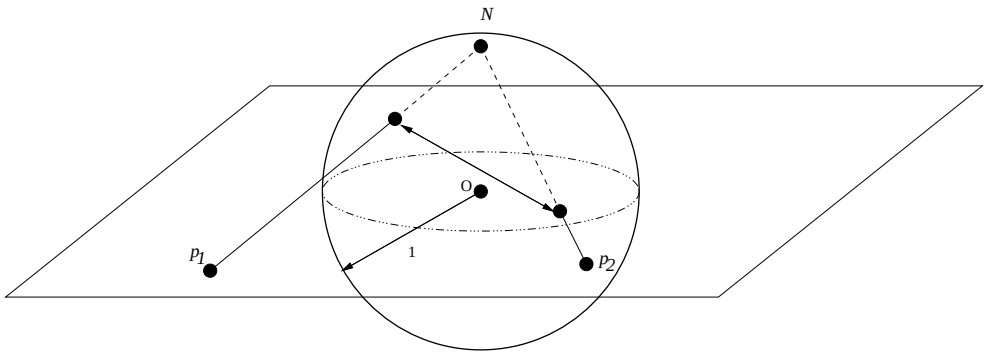
Zo'n beste benadering bestaat niet altijd. Neem bijvoorbeeld $D = (0, 1)$ in $\mathbb{R}$ en $x = 2$. Dan is $\rho(x, D) = 1$. Deze ondergrens wordt echter niet bereikt want er is geen enkel punt $y \in (0, 1)$ waarvoor $\rho(y, 2) = 1$. Het punt $x = 2$ heeft bijgevolg geen beste benadering in het open interval $(0, 1)$. Ook is het zo dat een beste benadering niet altijd uniek is. Bijvb., alle punten op de eenheidscirkel liggen even ver van de oorsprong. Ze zijn alle 'beste benaderingen' voor het punt $(0, 0)$.

3.1.3 Voorbeelden van afstandsfuncties

De functies (3.1) en (3.2) voldoen aan de definitie van afstand. We zullen (3.1) de *klassieke afstand* noemen in de ruimte van de reële getallen. Ook voor complexe getallen is (3.1), de modulus van het verschil van de twee getallen, een klassieke afstand.



Figuur 3.1: Driehoeksongelijkheid: $\rho(x, y) \leq \rho(z, x) \ + \ \rho(z, y)$.



Figuur 3.2: Chordale afstand tussen de punten p_1 en p_2.

3.1.3.1 Enkele niet-klassieke afstandsfuncties

1. In de complexe functieleer gebruikt men soms de zogenaamde *chordale* afstand. Dit is de lengte van de koorde tussen twee complexe getallen die voorgesteld zijn op de bol van Riemann, zie Figuur 3.2.

2. Elke niet-ledige verzameling kan 'gemetriseerd' worden door het invoeren van een afstand. Definieer hiertoe de *triviale* afstand

$$\rho(x, y) = 1 \quad \text{als} \quad x \neq y \ . \tag{3.6}$$

   Praktische toepassingen heeft deze afstandsfunctie niet, maar hij is wel dienstig als voorbeeld (of tegenvoorbeeld) bij allerlei eigenschappen en stellingen.

3. Zij S een verzameling van verzamelingen A, B, …die elk een eindig aantal elementen bevatten. De functie

$$\rho(A, B) = \#\{(A \cup B) \setminus (A \cap B)\} \tag{3.7}$$

   voldoet aan de definitie van afstand. Dit is de *Silverman*-afstand.

4. In de code-theorie definieert men de *Hamming*-afstand tussen twee n-koppels als het aantal plaatsen waarin hun componenten van elkaar verschillen. Bijvoorbeeld

$$\rho(\ (1,0,0,1,1,0,1)\ ,\ (0,1,0,1,0,1,1)\ )\ =\ 4\ .$$

5. In de ruimte van de complexe $n \times n$ matrices wordt de *Frobenius*-afstand gedefinieerd:

$$\rho(A, B) = \sqrt{\sum_{i=1}^{n} \sum_{j=1}^{n} |A_{ij} - B_{ij}|^2}, \quad \text{met} \quad A, B \in \mathbb{C}^{n \times n} \ .$$

3.1.3.2 Afstandsfuncties in continue functieruimten

We geven enkele voorbeelden. Bij de notatie $1 \leq p < \infty$ is de achterliggende veronderstelling dat p een *reëel* getal is.

1. Zij $w(x)$ een positieve gewichtsfunctie die begrensd is over $[a, b]$. De functie

$$\rho(f, g) = \left[\int_a^b w(x) \, |f(x) - g(x)|^2 \, dx \right]^{1/2} \tag{3.8}$$

   is een afstandsfunctie in $C[a, b]$, de verzameling van alle continue reële functies op het interval $[a, b]$. Men spreekt van de *kwadratische* afstand. De metrische ruimte die zo ontstaat, stelt men dikwijls voor door $C([a, b], w(x))$.

2. De kwadratische afstand is een speciaal geval van de zogenaamde L_p-afstand

$$\rho(f, g) = \left[\int_a^b w(x) \, |f(x) - g(x)|^p \, dx \right]^{1/p} \quad \text{met} \quad 1 \leq p < \infty \quad \text{in} \quad C[a, b] \; . \tag{3.9}$$

3. Verder is er ook de *Chebyshev*-afstand, *gelijkmatige* afstand of *maximum*afstand,

$$\rho(f, g) = \sup_{x \in [a,b]} \; | \; f(x) - g(x) \; |, \tag{3.10}$$

   in $B[a, b]$, de verzameling van alle begrensde reële functies op het interval $[a, b]$. In $C[a, b]$ wordt dit vereenvoudigd tot

$$\rho(f, g) = \max_{x \in [a,b]} \; | \; f(x) - g(x) \; | \; . \tag{3.11}$$

4. Beschouwen we de verzameling van alle reële functies f waarvoor de integraal

$$\int_a^b |f(x)|^p \, dx \quad \text{met} \quad 1 \leq p < \infty \tag{3.12}$$

   bestaat (d.w.z. een eindige waarde heeft). In deze verzameling definieert

$$\rho(f, g) = \left[\int_a^b |f(x) - g(x)|^p \, dx \right]^{1/p} \tag{3.13}$$

   geen afstandsfunctie! Inderdaad, wanneer twee functies f en g slechts in één punt verschillen, dan zou $\rho(f, g)$ de waarde nul aannemen, terwijl toch geldt dat $f \neq g$.

   Om hieraan te verhelpen gaat men het begrip functie aanpassen. Men spreekt af geen onderscheid meer te maken tussen functies die slechts in een enkel punt of in een 'beperkt' aantal punten van elkaar verschillen. Het worden dan verschillende vertegenwoordigers van dezelfde functieklasse. Wiskundig zegt men dat men functies

identificeert die slechts verschillen in een verzameling met *maat nul*[1]. Hiermee bedoelt men ruwweg dat de verzameling punten zo klein is, dat ze een totale 'lengte' nul heeft. De gelijkheid van twee functies wordt nu gedefinieerd als

$$f = g \quad \text{als en slechts als} \quad f(x) = g(x) \ b.o. \tag{3.14}$$

Hierbij staat de afkorting '*b.o.*' voor 'bijna overal'. In de Engelstalige literatuur gebruikt men de afkorting '*a.e.*' (almost everywhere).

Om de integraal van dergelijke functies te berekenen schiet het klassieke begrip *Riemann-integraal* te kort, en moet men zijn toevlucht nemen tot een meer algemeen soort integraal, de *Lebesgue-integraal*. Voor de geïnteresseerde lezer verwijzen we bijvoorbeeld naar [NS82, App. D]. Hier volstaat het echter aan te nemen dat het klassieke integraalbegrip uitgebreid kan worden, zodat bijv. ook functies met een oneindig aantal discontinuïteiten geïntegreerd kunnen worden. De zo ontstane verzameling van functieklassen stelt men voor door $L_p[a, b]$. Deze verzameling van alle 'functies' waarvoor de p-de macht Lebesgue-integreerbaar is, is een metrische ruimte. De afstand kan worden veralgemeend door invoering van een gewichtsfunctie $w(x)$.

Bewijzen dat (3.8), (3.9) en (3.13) aan de definitie van afstand voldoen, is niet zo gemakkelijk in dit stadium van de bespreking. We zullen dit later hernemen.

3.1.3.3 Afstandsfuncties in discrete functieruimten

1. Beschouw de verzameling van alle reële functies gedefinieerd in een eindig stel punten $x_1, x_2, \ldots, x_n$. Zo een functie f is volledig gedefinieerd door de functiewaarden $f_1, f_2, \ldots, f_n$. In deze verzameling van 'discrete functies' voldoet

$$\rho(f, g) = \left[\sum_{i=1}^{n} |f_i - g_i|^p \right]^{1/p} \quad \text{met} \quad 1 \leq p < \infty \tag{3.15}$$

aan de definitie van afstand. Ook hier zijn de speciale gevallen $p = 1$, $p = 2$ en het limietgeval voor $p \to \infty$ het belangrijkst. Dit laatste geeft aanleiding tot de *discrete Chebyshev-afstand*, gedefinieerd als

$$\rho(f, g) = \max_i |f_i - g_i| \ . \tag{3.16}$$

Merk op dat (3.15) en (3.16) geen afstandsfuncties zijn in de verzameling van continue functies gedefinieerd over het *interval* $[x_1, x_n]$.

2. Bekijken we de verzameling van alle oneindige rijen van reële of complexe getallen, $x = (x_1, x_2, \ldots)$ die p-sommeerbaar zijn, d.w.z.

$$\sum_{i=1}^{\infty} |x_i|^p < \infty \ ,$$

[1] Zij $I_k = (a_k, b_k), k = 1, 2, \ldots$, een rij open intervallen in $\mathbb{R}$. Definieer de totale lengte van de rij door $\Sigma_{k=1}^{\infty}(b_k - a_k)$. Een verzameling $E \subset \mathbb{R}$ is een verzameling met maat nul, als voor elke $\epsilon > 0$ er een rij $\{I_k\}$ van intervallen met totale lengte kleiner dan ϵ bestaat zodat $E \subset \cup_k I_k$.

met p een waarde die voldoet aan $1 \leq p < \infty$. Deze verzameling stelt men voor door l_p. Ze wordt gemetriseerd door de afstandsfunctie

$$\rho(f,g) = \left[\sum_{i=1}^{\infty} |f_i - g_i|^p\right]^{1/p} . \tag{3.17}$$

In de limiet voor $p \to \infty$ verkrijgen we de ruimte l_∞ van alle oneindige, *begrensde* reële of complexe rijen, met als afstandsfunctie

$$\rho(f,g) = \sup_i |f_i - g_i| . \tag{3.18}$$

3.2 Genormeerde ruimte en lengte

In de rest van het hoofdstuk veronderstellen we dat de beschouwde ruimte V een zogenaamde *vectorruimte*, ook genoemd *lineaire ruimte*, is. We herhalen even de definitie van een vectorruimte over een veld $\mathbb{F}$. In deze cursus zullen we steeds $\mathbb{F} = \mathbb{C}$ of $\mathbb{F} = \mathbb{R}$ veronderstellen.

DEFINITIE 3.2.3*(vectorruimte)*
Een vectorruimte V over het veld $\mathbb{F}$ is een verzameling van elementen (zgn. vectoren), waarop twee bewerkingen zijn gedefinieerd: optelling en een scalaire vermenigvuldiging met een element uit $\mathbb{F}$. Deze bewerkingen moeten voldoen aan volgende voorwaarden.

1. $\vec{u} + \vec{v} \in V, \quad \forall \vec{u},\ \vec{v} \in V$
2. $\vec{u} + (\vec{v} + \vec{w}) = (\vec{u} + \vec{v}) + \vec{w}, \quad \forall \vec{u},\ \vec{v},\ \vec{w} \in V$
3. *Er bestaat een element* $\vec{0} \in V$ *zodat* $\vec{0} + \vec{v} = \vec{v},\ \forall \vec{v} \in V$
4. *Voor elke* $\vec{v} \in V$ *bestaat er element* $-\vec{v} \in V$ *zodat* $\vec{v} + (-\vec{v}) = 0$
5. $\vec{u} + \vec{v} = \vec{v} + \vec{u},\ \forall \vec{u},\ \vec{v} \in V$
6. $f\vec{v} \in V,\ \forall f \in \mathbb{F},\ \forall \vec{v} \in V$
7. $f(g\vec{v}) = (fg)\vec{v},\ \forall f,\ g \in \mathbb{F},\ \forall \vec{v} \in V$
8. *Als* 1 *het eenheidselement in* $\mathbb{F}$ *is geldt dat* $1\vec{v} = \vec{v},\ \forall \vec{v} \in V$
9. $f(\vec{u} + \vec{v}) = f\vec{u} + f\vec{v},\ \forall f \in \mathbb{F}, \forall \vec{u},\ \vec{v} \in V$
10. $(f + g)\vec{v} = f\vec{v} + g\vec{v},\ \forall f,\ g \in \mathbb{F},\ \forall \vec{v} \in V$

In Hoofdstuk 2 hebben we reeds twee vectorruimten beschouwd, namelijk de vectorruimte $\mathbb{C}^m$ over de complexe getallen en de vectorruimte $\mathbb{R}^m$ over de reële getallen. We wensen in deze paragraaf het begrip norm met de bijhorende noties van lengte en afstand te veralgemenen naar abstracte vectorruimten. Voor de duidelijkheid worden elementen uit

een vectorruimte vanaf nu aangeduid met de vector-notatie, zoals in de definitie. De concrete betekenis van bijvoorbeeld $\vec{x}$ hangt af van de beschouwde ruimte. In $\mathbb{R}^n$ heeft $\vec{x}$ weliswaar de betekenis van een klassieke vector, in een functieruimte komt $\vec{x}$ overeen met een functie.
Een formele definitie van een genormeerde ruimte is als volgt.

DEFINITIE 3.2.4*(norm en genormeerde ruimte)*
Een norm *over de vectorruimte V is een functionaal van V naar $\mathbb{R}$ waarvan de beelden $\| \vec{x} \|$ voldoen aan de volgende vier eigenschappen:*

$$\begin{aligned} &1.\ \| \vec{x} \| \geq 0 \\ &2.\ \| \vec{x} \| = 0 \ \text{ als en alleen als } \ \vec{x} = \vec{0} \\ &3.\ \| a\,\vec{x} \| = |a| \ \| \vec{x} \| \\ &4.\ \| \vec{x} + \vec{y} \| \leq \| \vec{x} \| + \| \vec{y} \| \end{aligned} \tag{3.19}$$

Een vectorruimte waarover een norm gedefinieerd is, is een genormeerde *ruimte.*

De eerste twee eigenschappen zeggen dat de norm een positief definiete functionaal is. De derde eis noemt men de homogeniteitseigenschap; de vierde eis is de driehoeksongelijkheid. Uit de driehoeksongelijkheid volgt verder ook de volgende eenvoudige eigenschap

$$| \ \| \vec{x} \| - \| \vec{y} \| \ | \leq \| \ \vec{x} - \vec{y} \ \| \, . \tag{3.20}$$

3.2.1 Verband met metrische ruimten

Hierboven werd reeds vermeld dat de klassieke normfunctie in het vlak ook een afstand tussen punten genereert. We zullen die eigenschap in deze paragraaf veralgemenen. Zo zullen we aantonen dat elke norm een afstandsfunctie definieert; anderzijds is echter niet elke afstandsfunctie afkomstig van een normfunctie.

STELLING 3.2.1
Als een vectorruimte genormeerd is, dan is ze ook metrisch. De functie

$$\rho(\vec{x}, \vec{y}) = \| \vec{x} - \vec{y} \| \tag{3.21}$$

voldoet aan de definitie van afstand.

Bewijs De eerste drie voorwaarden voor een afstandsfunctie zijn gemakkelijk te verifiëren. De te bewijzen driehoeksongelijkheid luidt

$$\| \vec{x} - \vec{y} \| \leq \| \vec{x} - \vec{z} \| + \| \vec{y} - \vec{z} \| \ . \tag{3.22}$$

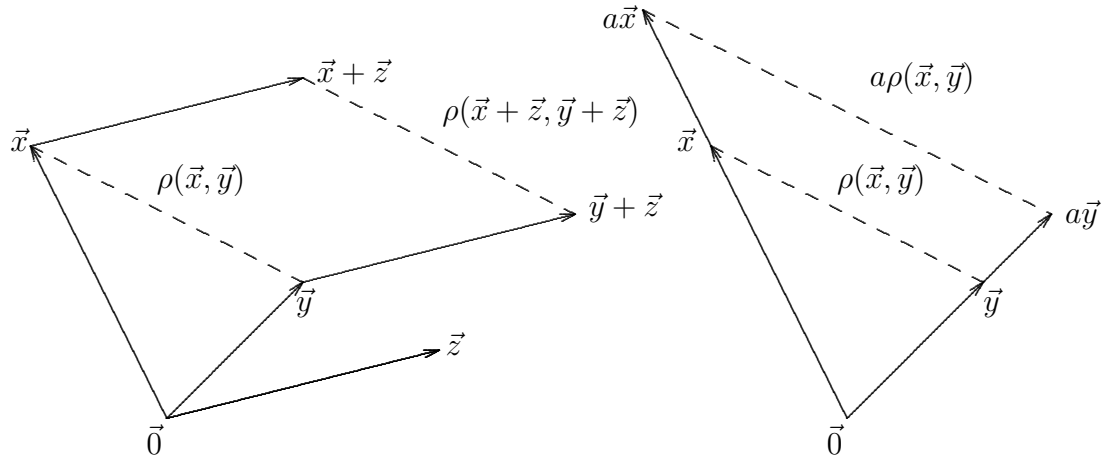
Welnu voor normen geldt de driehoeksongelijkheid $\| \vec{\alpha} + \vec{\beta} \| \leq \| \vec{\alpha} \| + \| \vec{\beta} \|$; stel hierin $\vec{\alpha} = \vec{x} - \vec{z}$ en $\vec{\beta} = \vec{z} - \vec{y}$. □

In een willekeurige metrische ruimte kan men niet altijd een norm definiëren die met de afstand door (3.21) verbonden is. De metrische ruimte moet op zijn minst een vectorruimte zijn (dus een *metrische vectorruimte*). Zoals in de volgende stelling wordt geponeerd, moet de afstandsfunctie ook nog aan een aantal bijkomende voorwaarden voldoen.

STELLING 3.2.2
Een metrische vectorruimte kan genormeerd worden met een norm die voldoet aan (3.21), als en slechts als de afstandsfunctie voldoet aan

1. $\rho(\vec{x},\vec{y}) = \rho(\vec{x}+\vec{z},\vec{y}+\vec{z})$ *voor alle* $\vec{x}, \vec{y}, \vec{z} \in V$
2. $\rho(a\vec{x},a\vec{y}) = a\rho(\vec{x},\vec{y})$ *voor alle* $a \in \mathbb{R}^+$ *en alle* $\vec{x}, \vec{y} \in V$.

Een afstand in een metrische vectorruimte die aan de eerste eigenschap voldoet, noemt men een *translatie-invariante* of *supermetrische* afstand. Een afstand die aan de tweede eigenschap voldoet noemt men *homogeen*. Voor een illustratie, zie Figuur 3.3. We laten de (technische) details van het bewijs achterwege. Het bewijs bestaat uit twee stappen. Men toont eerst aan dat $\| \vec{x} - \vec{y} \|$ steeds een homogene en supermetrische afstand is. De omgekeerde implicatie bewijst men door aan te tonen dat $\rho(\vec{x},\vec{0})$ een normfunctie is.



Figuur 3.3: Translatie-invariantie en homogeniteit van een afstand in een vectorruimte.

Uit bovenstaande stellingen en uit het feit dat de verzamelingen vectorruimten moeten zijn, volgt dat volgende afstanden niet tot een norm aanleiding geven:

- de chordale afstand
- de triviale afstand (3.6)
- de Silverman afstand (3.7)
- de supermetrische maar niet-homogene afstand

$$\rho(x,y) = \frac{|x-y|}{1+|x-y|} \ , \quad x,y \in \mathbb{R}.$$

Men kan in een vectorruimte allerlei normen en afstanden definiëren. Als men in een genormeerde ruimte echter over afstand spreekt, dan bedoelt men daarmee steeds de afstand die met de norm verbonden is volgens (3.21). We noemen die de *natuurlijke* of *geïnduceerde* afstand.

3.2.2 Enkele voorbeelden van genormeerde ruimten

1. In de lineaire algebra wordt aangetoond dat de volgende functionalen normen zijn in de eindigdimensionale ruimte van de reële of complexe n-koppels $\vec{x} = (x_1, x_2, ..., x_n)$

$$\| \vec{x} \|_\infty = \max\{|x_1|, |x_2|, ..., |x_n|\} \tag{3.23}$$

$$\| \vec{x} \|_1 = |x_1| + |x_2| + ... + |x_n| \tag{3.24}$$

$$\| \vec{x} \|_p = (|x_1|^p + |x_2|^p + ... + |x_n|^p)^{1/p} \quad \text{voor} \quad p \geq 1 \ . \tag{3.25}$$

   We noemen (3.23) de ∞-norm, de discrete maximumnorm of max-norm, en (3.25) de discrete p-norm. Norm (3.24) wordt soms de *Manhattan-norm* genoemd.

2. In $B[a, b]$ voldoet $\sup_{x\in[a,b]} |f(x)|$ aan de definitie van norm.

   Om dit aan te tonen moeten we eerst vermelden dat het supremum inderdaad bestaat, anders zou f geen begrensde functie zijn. De functionaal is dus inderdaad gedefinieerd in elk 'punt' f van $B[a, b]$.

   De eerste drie normeigenschappen zijn eenvoudig te controleren. Om de driehoeksongelijkheid te bewijzen, gaan we als volgt tewerk. Voor elke $x \in [a, b]$ geldt dat

$$|f(x)| \leq \sup_{x\in[a,b]} |f(x)| \quad \text{en} \quad |g(x)| \leq \sup_{x\in[a,b]} |g(x)| \ .$$

   Dus geldt

$$|f(x) + g(x)| \leq |f(x)| + |g(x)| \leq \sup_{x\in[a,b]} |f(x)| + \sup_{x\in[a,b]} |g(x)| \ .$$

   Het rechterlid is eindig. Vermits de bovengrens geldt voor alle $x \in [a, b]$, volgt

$$\sup_{x\in[a,b]} |f(x) + g(x)| \leq \sup_{x\in[a,b]} |f(x)| + \sup_{x\in[a,b]} |g(x)| \ .$$

3. In $C[a, b]$ is onderstaande functionaal een norm:

$$\| f \| = \max_{x\in[a,b]} |f(x)| \ . \tag{3.26}$$

   Het is de continue *maximumnorm* of *max-norm*. Hiermee is ook bewezen dat (3.11) een afstand is in $C[a, b]$. Dat hadden we vroeger niet bewezen.

4. In de ruimte $C[a,b]$ geldt de volgende norm

$$\| f \|_p = \left[\int_a^b w(x)|f(x)|^p dx\right]^{1/p}, \quad \text{voor} \quad p \geq 1 \quad \text{en} \quad w(x) > 0 \; . \tag{3.27}$$

De driehoeksongelijkheid volgt hier onmiddellijk uit de ongelijkheid van Minkowski die in de analyse bewezen wordt (bijv. [NS82, p.548]):

$$\left[\int_a^b w(x)|f(x)+g(x)|^p dx\right]^{1/p} \leq \left[\int_a^b w(x)|f(x)|^p dx\right]^{1/p} + \left[\int_a^b w(x)|g(x)|^p dx\right]^{1/p}$$

Voor het geval $p = 2$ geven we in een volgende paragraaf een eenvoudig bewijs.

5. In de ruimte $L_p[a,b]$ met $1 \leq p < \infty$ is (3.27) met $w(x) \equiv 1$ een klassieke norm. Men spreekt van de L_p-norm, met als speciale geval de kwadratische norm ($p = 2$). Indien $w(x) \not\equiv 1$, dan spreekt men van een *gewogen L_p-norm.*

6. In de ruimte l_∞ van begrensde rijen (d.w.z. met $\sup_i |x_i| < \infty$) voldoet $\sup_i |x_i|$ aan de definitie van norm.

7. In de ruimte l_p van alle p-sommeerbare rijen geldt volgende norm

$$\| x \|_p = \left[\sum_{i=1}^{\infty} |x_i|^p\right]^{1/p} \quad \text{voor} \quad 1 \leq p < \infty \; . \tag{3.28}$$

Deze eigenschap is gebaseerd op een discrete Minkowski-ongelijkheid. Meer bepaald,

$$\left[\sum_{i=1}^{\infty} |f_i + g_i|^p\right]^{1/p} \leq \left[\sum_{i=1}^{\infty} |f_i|^p\right]^{1/p} + \left[\sum_{i=1}^{\infty} |g_i|^p\right]^{1/p} \; .$$

8. Ook alle eindigdimensionale deelvectorruimten van bovenvermelde genormeerde ruimten zijn genormeerd. Men toont in de lineaire algebra aan dat alle eindigdimensionale genormeerde ruimten in zekere zin analoog zijn: elke n-dimensionale vectorruimte over $\mathbb{R}$ ($\mathbb{C}$) is isomorf met de vectorruimte $\mathbb{R}^n$ ($\mathbb{C}^n$). Ook wat hun normen betreft is er een equivalentie: in een eindigdimensionale vectorruimte volgt uit convergentie volgens een norm ook convergentie volgens gelijk welke andere norm. Men noemt dit laatste soms de stelling van de *gelijkwaardigheid* of *equivalentie van normen.*

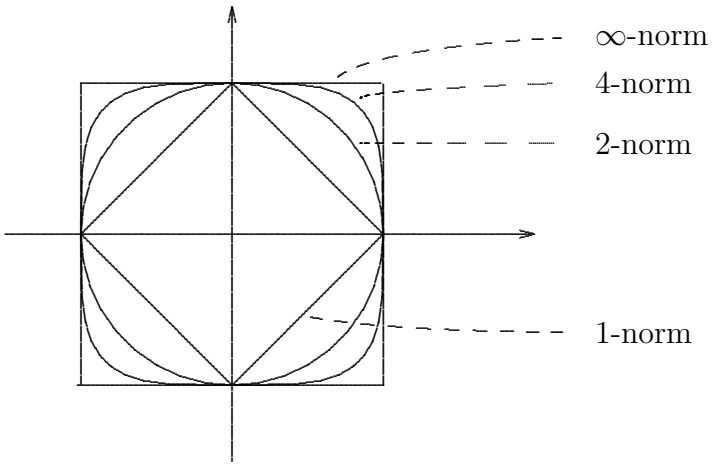
3.2.3 Beste benadering in een deelruimte

We karakteriseren benaderingen van een vector in een deelruimte. Daarbij speelt de eenheidsbol,

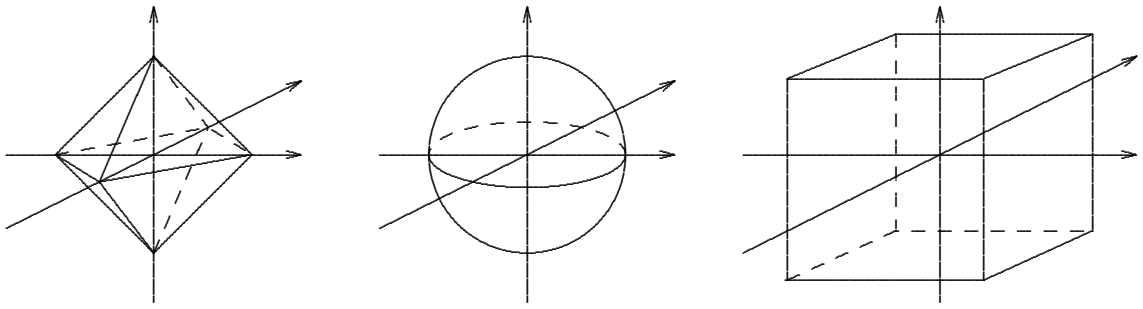
$$\widetilde{B}(\vec{0},1) = \{\vec{y} : \| \vec{y} \| = 1\},$$

een belangrijke rol.
Om eigenschappen van het benaderingsprobleem te concretiseren vertrekken we van meetkundige interpretaties in $\mathbb{R}^2$ en $\mathbb{R}^3$.

Voorbeeld 3.1 *In Figuur 3.4 en in Figuur 3.5 tonen we de 'eenheidsbol' in* $\mathbb{R}^2$ *en* $\mathbb{R}^3$*, voor verschillende discrete p-normen.*

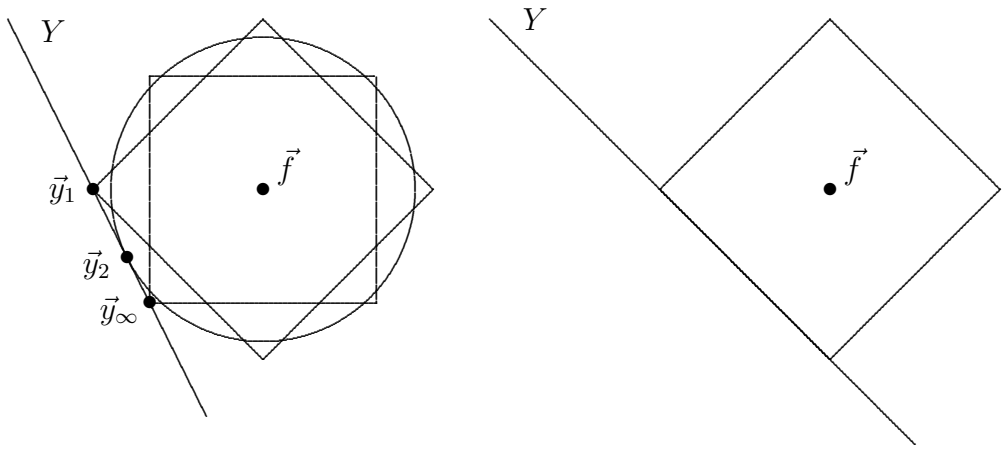


Figuur 3.4: Eenheidscirkels in het vlak.

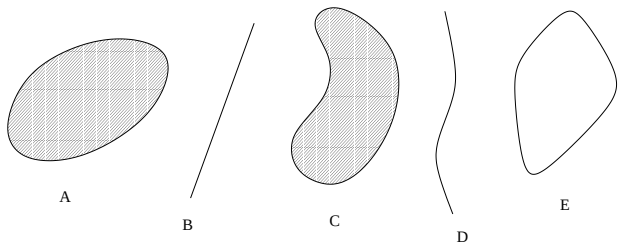


Figuur 3.5: Eenheidsbollen in de driedimensionale ruimte.

Het benaderen van een punt $\vec{f}$ *van de tweedimensionale ruimte door een punt* $\vec{y}$ *van een gegeven rechte* Y *door de oorsprong kan men opvatten als volgt. Men zoekt het punt waar de 'cirkel' met* $\vec{f}$ *als middelpunt en een nader te bepalen 'straal'* r*, de rechte* Y *juist 'raakt'. Men kan inzien dat men dan inderdaad het punt* $\vec{y}$ *bekomt met de kleinst mogelijke natuurlijke afstand tot* $\vec{f}$*. Uit de meetkundige voorstelling in Figuur 3.6 volgt dat de beste benadering van* $\vec{f}$ *in* Y *volgens de 2-norm steeds bestaat en enig is. De beste benaderingen volgens de 1- en de* ∞*-normen bestaan steeds, maar zijn niet altijd enig. De figuur laat bijvoorbeeld zien dat er op een welbepaalde rechte oneindig veel beste benaderingen kunnen bestaan volgens de 1-norm. Ook in hoogdimensionale ruimten zullen de normen zich typisch gedragen als de p-norm met* $1 < p < \infty$ *(hoogstens één beste benadering) of als de 1- of* ∞*-norm (mogelijkerwijs oneindig veel beste benaderingen).*



Figuur 3.6: De linkerfiguur toont de beste benadering voor het punt $\vec{f}$ volgens de 1-norm ($\vec{y}_1$), 2-norm ($\vec{y}_2$) en ∞-norm ($\vec{y}_\infty$). De rechterfiguur illustreert de mogelijke meervoudigheid van de beste benadering in de 1-norm.



Figuur 3.7: Convexe (A, B) en niet-convexe (C, D, E) verzamelingen.

Uit Voorbeeld 3.1 blijkt dat het al dan niet convex zijn van de eenheidsbol van een ruimte een belangrijke invloed heeft op het al dan niet uniek zijn van een beste benadering. Daarom zullen we het begrip convexiteit in het algemeen specificeren.

DEFINITIE 3.2.5*(convexe en strikt convexe verzameling)*
Een deelverzameling C van een vectorruimte V is convex *wanneer voor alle $\lambda > 0$ en $\mu > 0$ met $\lambda + \mu = 1$ en voor alle $\vec{x}_1, \vec{x}_2 \in C$ geldt dat alle punten $\lambda\,\vec{x}_1 + \mu\,\vec{x}_2 \in C$. Wanneer al deze punten tot het inwendige van C behoren, dan noemt men C* strikt convex.

Meetkundig betekent dit dat elk open lijnstuk $L(\vec{x}_1, \vec{x}_2)$ dat twee punten $\vec{x}_1$ en $\vec{x}_2$ van C verbindt, volledig tot C behoort, of tot het inwendige van C in geval van strikte convexiteit. In Figuur 3.7 worden enkele voorbeelden in $\mathbb{R}^2$ gegeven. Met het arceren van A en C bedoelen we dat zowel de rand als het inwendige tot het gebied gerekend worden. E is niet gearceerd, waarmee we bedoelen dat het inwendige niet tot het gebied behoort.

EIGENSCHAPPEN

– In een genormeerde ruimte is elke gesloten bol convex.

Inderdaad, zij $\vec{x}_1$ en $\vec{x}_2$ twee punten van $B(\vec{a}, r)$. Dan moeten we aantonen dat $\lambda\vec{x}_1 + (1-\lambda)\vec{x}_2$ tot de bol behoort, of nog dat $\parallel \lambda\vec{x}_1 + (1-\lambda)\vec{x}_2 - \vec{a} \parallel \leq r$. Welnu $\parallel \lambda\vec{x}_1 + (1-\lambda)\vec{x}_2 - \vec{a} \parallel \leq \lambda \parallel \vec{x}_1 - \vec{a} \parallel + (1-\lambda) \parallel \vec{x}_2 - \vec{a} \parallel \leq \lambda\, r + (1-\lambda)\, r = r$.

– In een genormeerde ruimte is de eenheidsbol $B(\vec{0}, 1)$ dus ook convex. Dit klopt met Figuren 3.4 en 3.5. Voor $p < 1$ zouden we een niet-convexe bol bekomen; met deze p-waarden stemmen dus geen normen overeen.

DEFINITIE 3.2.6 *(strikt genormeerde ruimte en strikte norm)*
Een genormeerde ruimte is strikt *genormeerd als de eenheidsbol van die ruimte strikt convex is. Men spreekt dan van een* strikte *norm.*

Neem twee punten $\vec{x}_1$ en $\vec{x}_2$ van het boloppervlak van de eenheidsbol. Dan is $L(\vec{x}_1, \vec{x}_2)$ een koorde van die bol (de eindpunten niet inbegrepen). Als de eenheidsbol strikt convex is dan behoort geen enkel punt van die koorde tot het boloppervlak. D.w.z. dat het boloppervlak geen 'rechte' lijnstukken kan bevatten. Wiskundig kunnen we dit ook schrijven als

$$\text{als } \vec{x}_1 \neq \vec{x}_2 \text{ en } \parallel \vec{x}_1 \parallel = \parallel \vec{x}_2 \parallel = 1 \text{ dan geldt } \parallel \vec{x}_1 + \vec{x}_2 \parallel < 2\ . \tag{3.29}$$

De 1-norm en de ∞-norm in $\mathbb{R}^n$ zijn geen strikte normen. Je kan nagaan dat de eenheidsbol in $C[a, b]$ niet strikt convex is.

Het belang van een strikt genormeerde ruimte bij het benaderingsprobleem in een deelruimte wordt door volgende stelling onderstreept.

STELLING 3.2.3 *(beste benadering in een deelruimte)*
Zij $\mathcal{D}$ een eindigdimensionale deelruimte van een strikt genormeerde ruimte V en zij $\vec{v} \in V$. Dan bestaat de beste benadering van $\vec{v}$ in $\mathcal{D}$ en is deze uniek.

Bewijs Existentie. Noem $d = \inf\{\|\vec{v} - \vec{w}\| : \vec{w} \in \mathcal{D}\}$. We tonen aan dat dit infimum in feite een minimum is. Volgens de definitie van een infimum bestaat er een rij van vectoren $\{\vec{w}_k\}_{k\geq 1}$ in $\mathcal{D}$ zodat $\{\|\vec{v} - \vec{w}_k\|\}_{k\geq 1}$ een dalende rij is die convergeert naar d. De rij $\{\vec{w}_k\}_{k\geq 1}$ is bovendien uniform begrensd omdat

$$\|\vec{w}_k\| = \|(\vec{w}_k - \vec{v}) + \vec{v}\| \leq \|\vec{w}_k - \vec{v}\| + \|\vec{v}\| \leq \|\vec{w}_1 - \vec{v}\| + \|\vec{v}\|,\ \forall k \geq 1. \tag{3.30}$$

Stel n gelijk aan de dimensie van $\mathcal{D}$ en beschouw een basis $\{\vec{a}_1, \ldots, \vec{a}_n\}$ van $\mathcal{D}$. Dan kunnen we $\vec{w}_k,\ k \geq 1$, ontbinden als

$$\vec{w}_k = \sum_{i=1}^{n} c_{ki}\, \vec{a}_i.$$

Uit (3.30) volgt dat de rij $\{(c_{k1}, \ldots, c_{kn})\}_{k\geq 1}$ begrensd is. Deze rij heeft bijgevolg steeds een convergente deelrij (stelling van Weierstrass-Bolzano) waarvan we de limiet $(\hat{c}_1, \ldots, \hat{c}_n)$ noemen. Daarom kunnen we, zonder algemeenheid in te boeten, in wat volgt veronderstellen dat de rij $\{(c_{k1}, \ldots, c_{kn})\}_{k\geq 1}$ convergeert naar $(\hat{c}_1, \ldots, \hat{c}_n)$. Definieer

$$\vec{\zeta} = \sum_{i=1}^{n} \hat{c}_i\, \vec{a}_i \in \mathcal{D},$$

dan geldt er voor alle $k \geq 1$ dat

$$\|\vec{v} - \vec{\zeta}\| \leq \underbrace{\|\vec{v} - \vec{w}_k\|}_{\to d} + \underbrace{\|\vec{w}_k - \vec{\zeta}\|}_{\to 0},$$

wat $\|\vec{v} - \vec{\zeta}\| = d$ impliceert. Vector $\vec{\zeta}$ is dus een beste benadering.
Uniciteit. Het bewijs is uit het ongerijmde. Veronderstel dat er twee verschillende beste benaderingen zijn, $\vec{v}_1$ en $\vec{v}_2$, zodat

$$\|\vec{v} - \vec{v}_1\| = \|\vec{v} - \vec{v}_2\| = d.$$

Merk dat $\vec{e}_i := \frac{1}{d}(\vec{v} - \vec{v}_i)$ op de eenheidsbol in V ligt voor $i = 1, 2$. Omdat de eenheidsbol strikt convex is geldt

$$\left\| \vec{v} - \frac{\vec{v}_1 + \vec{v}_2}{2} \right\| = d \left\| \underbrace{\frac{1}{2}}_{\lambda} \vec{e}_1 + \underbrace{\frac{1}{2}}_{\mu} \vec{e}_2 \right\| < d.$$

Dus $\frac{1}{2}(\vec{v}_1 + \vec{v}_2) \in \mathcal{D}$ is een betere benadering dan $\vec{v}_1$, wat in tegenspraak is met de veronderstelling. □

3.3 Unitaire ruimte en orthogonaliteit

3.3.1 Het begrip scalair product

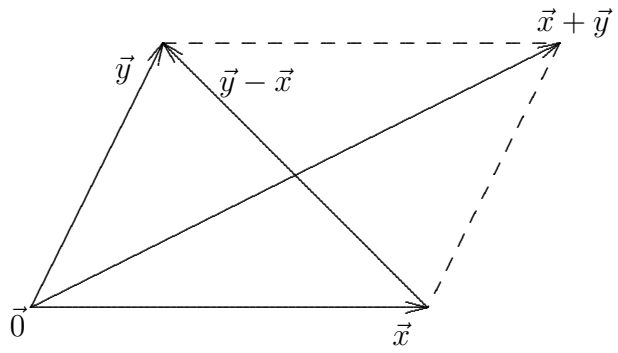
Een unitaire ruimte is een vectorruimte, uitgerust met een scalair product. De formele definitie voor een unitaire vectorruimte over het veld $\mathbb{C}$ is als volgt.

Definitie 3.3.7 *(scalair product en unitaire ruimte)*
Men noemt een vectorruimte V over de complexe getallen unitair *als er met elk paar elementen $\vec{x}, \vec{y} \in V$ een complex getal $(\vec{x}, \vec{y})$ overeenstemt dat voldoet aan de volgende eigenschappen:*

$$\begin{aligned}
&1.\quad (\vec{x}, a\vec{y}) = a(\vec{x}, \vec{y}) \quad \textit{voor alle } a \in \mathbb{C} \\
&2.\quad (\vec{x} + \vec{y}, \vec{z}) = (\vec{x}, \vec{z}) + (\vec{y}, \vec{z}) \\
&3.\quad (\vec{x}, \vec{y}) = \overline{(\vec{y}, \vec{x})} \\
&4.\quad (\vec{x}, \vec{x}) > 0 \quad \textit{als } \vec{x} \neq \vec{0}.
\end{aligned} \tag{3.31}$$

Men noemt $(\vec{x}, \vec{y})$ het scalair product *van $\vec{x}$ en $\vec{y}$.*

Uit de derde eigenschap volgt dat $(\vec{x}, \vec{x}) \in \mathbb{R}$ en dus is het gebruik van het '>'-symbool in de vierde eigenschap verantwoord. Uit de eerste en derde eigenschap volgt bijvoorbeeld nog dat $(a\vec{x}, \vec{y}) = \overline{a}\,(\vec{x}, \vec{y})$ voor $a \in \mathbb{C}$.
Voor een vectorruimte over de reële getallen geldt er een consistente definitie.



Figuur 3.8: Parallellogramgelijkheid in het vlak.

DEFINITIE 3.3.8*(scalair product en unitaire ruimte)*
Men noemt een vectorruimte V over de reële getallen unitair *als er met elk paar elementen $\vec{x}, \vec{y} \in V$ een reëel getal $(\vec{x}, \vec{y})$ overeenstemt dat voldoet aan de volgende eigenschappen:*

$$\begin{aligned}
&1.\quad (\vec{x}, a\vec{y}) = a(\vec{x}, \vec{y}) \quad \textit{voor alle } a \in \mathbb{R} \\
&2.\quad (\vec{x}+\vec{y}, \vec{z}) = (\vec{x}, \vec{z}) + (\vec{y}, \vec{z}) \\
&3.\quad (\vec{x}, \vec{y}) = (\vec{y}, \vec{x}) \\
&4.\quad (\vec{x}, \vec{x}) > 0 \quad \textit{als } \vec{x} \neq \vec{0}.
\end{aligned} \tag{3.32}$$

Men noemt $(\vec{x}, \vec{y})$ het scalair product *van $\vec{x}$ en $\vec{y}$.*

Merk dat voor een vectorruimte over het veld $\mathbb{R}$ het scalair product symmetrisch is, in tegenstelling tot een vectorruimte over het veld $\mathbb{C}$.
In algemene unitaire ruimten geldt de zogenaamde ongelijkheid van Cauchy-Schwarz, die afgeleid kan worden uit de axioma's van het scalair product:

$$|(\vec{x}, \vec{y})| \leq \sqrt{(\vec{x}, \vec{x})}\ \sqrt{(\vec{y}, \vec{y})}\ . \tag{3.33}$$

3.3.2 Verband met genormeerde ruimten

De relatie tussen genormeerde en unitaire ruimten is vrij gelijkaardig aan die tussen metrische en genormeerde ruimten. We zullen aantonen dat elke unitaire ruimte tevens een genormeerde ruimte is, maar niet omgekeerd. Een genormeerde ruimte is slechts unitair wanneer de norm aan een welbepaalde voorwaarde voldoet. Tot slot zullen we bewijzen dat een unitaire ruimte genormeerd is met een *strikte* norm. Deze eigenschap impliceert het bestaan en de uniciteit van beste benaderingen in deelruimten, zie Stelling 3.2.3.

STELLING 3.3.4
Als een vectorruimte unitair is, dan is ze ook genormeerd. De functie

$$\| \vec{x} \| = \sqrt{(\vec{x}, \vec{x})} \tag{3.34}$$

voldoet aan de definitie van norm.

Bewijs De eerste drie normeigenschappen zijn gemakkelijk te bewijzen. De driehoeksongelijkheid volgt (voor $\vec{x}+\vec{y}\neq\vec{0}$) uit de ongelijkheid van Cauchy-Schwarz, als volgt:

$$\begin{aligned} 0 \leq (\vec{x}+\vec{y},\vec{x}+\vec{y}) &= (\vec{x},\vec{x}+\vec{y})+(\vec{y},\vec{x}+\vec{y}) \\ &\leq \sqrt{(\vec{x},\vec{x})}\sqrt{(\vec{x}+\vec{y},\vec{x}+\vec{y})}+\sqrt{(\vec{y},\vec{y})}\sqrt{(\vec{x}+\vec{y},\vec{x}+\vec{y})} \end{aligned}$$

en dus $\sqrt{(\vec{x}+\vec{y},\vec{x}+\vec{y})} \leq \sqrt{(\vec{x},\vec{x})}+\sqrt{(\vec{y},\vec{y})}$.

Voor het geval $\vec{x}+\vec{y}=\vec{0}$ is de driehoeksongelijkheid vanzelfsprekend. □

STELLING 3.3.5
Een genormeerde vectorruimte is een unitaire ruimte met een scalair product dat voldoet aan (3.34), als en slechts als de norm voldoet aan

$$\| \vec{x}+\vec{y} \|^2 + \| \vec{x}-\vec{y} \|^2 = 2\,(\,\| \vec{x} \|^2 + \| \vec{y} \|^2\,)\ . \tag{3.35}$$

Bewijs Het nodig zijn wordt als volgt aangetoond:

$$\| \vec{x}+\vec{y} \|^2 + \| \vec{x}-\vec{y} \|^2 = (\vec{x}+\vec{y},\vec{x}+\vec{y}) + (\vec{x}-\vec{y},\vec{x}-\vec{y}) = 2(\vec{x},\vec{x}) + 2(\vec{y},\vec{y}).$$

Voor een reële vectorruimte wordt het voldoende zijn bewezen door aan te tonen dat

$$(\vec{x},\vec{y}) = \frac{1}{4}\{\| \vec{x}+\vec{y} \|^2 - \| \vec{x}-\vec{y} \|^2\} \tag{3.36}$$

een scalair product is, en dat de natuurlijke norm van dit scalair product de oorspronkelijk gegeven norm is. Het bewijs is nogal technisch en laten we achterwege. □

Formule (3.35) wordt de *parallellogramgelijkheid* genoemd. In het Euclidische vlak komt deze gelijkheid overeen met het feit dat de som van de kwadraten van de diagonalen van een parallellogram gelijk is aan de som van de kwadraten van de vier zijden, zie figuur 3.8.

STELLING 3.3.6
De eenheidsbol in een unitaire ruimte is strikt convex.

Bewijs Het volstaat aan te tonen dat (3.29) geldt. Welnu, neem $\vec{x}\neq\vec{y}$ met $\| \vec{x} \| = 1$ en $\| \vec{y} \| = 1$. Dan volgt uit de parallellogramgelijkheid

$$\| \vec{x}+\vec{y} \|^2 = - \| \vec{x}-\vec{y} \|^2 + 2(\| \vec{x} \|^2 + \| \vec{y} \|^2) = - \| \vec{x}-\vec{y} \|^2 + 4 < 4\ .$$

En dus is $\| \vec{x}+\vec{y} \| < 2$. □

In een unitaire ruimte kunnen allerhande normen gedefinieerd worden. We zullen (3.34) de *natuurlijke* of *geïnduceerde* norm noemen. Als we van norm spreken in een unitaire ruimte, dan bedoelen we daarmee steeds deze natuurlijke norm. Zo kunnen we vanaf nu de ongelijkheid van Cauchy-Schwarz schrijven als

$$|(\vec{x},\vec{y})| \leq \| \vec{x} \|\,\| \vec{y} \|\ . \tag{3.37}$$

Als we van afstand spreken, dan bedoelen we daarmee de natuurlijke afstand van de natuurlijke norm. Unitaire ruimten zijn dus een speciaal geval van genormeerde ruimten, zoals genormeerde ruimten een speciaal geval zijn van metrische ruimten.

3.3.3 Enkele voorbeelden

1. Zij $(\vec{x}, \vec{y})$ een scalair product in een willekeurige reële n-dimensionale vectorruimte V. Kiezen we een basis $\vec{e}_1, \vec{e}_2, ..., \vec{e}_n$ in V. Dan is

$$(\vec{x}, \vec{y}) = (\sum_{i=1}^{n} x_i \vec{e}_i, \sum_{j=1}^{n} y_j \vec{e}_j) = \sum_{i=1}^{n} \sum_{j=1}^{n} x_i y_j (\vec{e}_i, \vec{e}_j) \ .$$

   Veronderstel dat alle scalaire producten van de basisvectoren gegeven zijn. We noemen ze $g_{ij} = (\vec{e}_i, \vec{e}_j)$. Dan kan men het scalair product van $\vec{x}$ en $\vec{y}$ schrijven als

$$(\vec{x}, \vec{y}) = \sum_{i=1}^{n} \sum_{j=1}^{n} g_{ij} x_i y_j \quad \text{of, in matrixnotatie,} \quad (\vec{x}, \vec{y}) = X^T G Y \ , \tag{3.38}$$

   waarbij X en Y de kolomvectoren zijn van de componenten van $\vec{x}$ en $\vec{y}$. In het geval van een *complexe* ruimte verkrijgt men als scalair product iets van de vorm

$$(\vec{x}, \vec{y}) = \sum_{i=1}^{n} \sum_{j=1}^{n} g_{ij} \bar{x}_i y_j \quad \text{of, in matrixnotatie,} \quad (\vec{x}, \vec{y}) = X^* G Y \ . \tag{3.39}$$

   Men noemt G de *fundamentele metrische matrix* van het scalair product t.o.v. de gekozen basis. We zullen verder aantonen dat deze matrix symmetrisch positief definiet is in reële ruimten, en Hermitiaans positief definiet in complexe.

   Indien de basisvectoren een orthonormaal stel vormen, dan is G een eenheidsmatrix. In dat geval spreken we van het *klassieke* scalair product. In $\mathbb{R}^n$ stemt de natuurlijke norm van dit scalair product dan overeen met de 2-norm. Merk ook op dat de p-norm met $p \neq 2$ geen aanleiding geeft tot een scalair product. De grootheid

$$\| \vec{x} \|_p = \left[\sum_{i=1}^{n} |x_i|^p \right]^{1/p}$$

   voldoet aan de definitie van norm, maar niet aan de parallellogramgelijkheid. Neem als tegenvoorbeeld in (3.35): $\vec{x} = (1, 1, 0, \ldots, 0)$ en $\vec{y} = (1, -1, 0, \ldots, 0)$.

2. De ruimte l_2 van kwadratisch sommeerbare rijen in $\mathbb{C}$ is een unitaire ruimte met

$$(\vec{x}, \vec{y}) = \sum_{i=1}^{\infty} \overline{x_i}\, y_i \ . \tag{3.40}$$

   De genormeerde ruimten l_p met $p \neq 2$ zijn geen unitaire ruimten. Er is niet voldaan aan de parallellogramgelijkheid. Illustreer dit voor l_1 en l_∞.

3. De maximumnorm in $C[a, b]$ geeft geen aanleiding tot een scalair product. De parallellogramgelijkheid zou immers luiden

$$\left[\max_{x\in[a,b]}|f(x)+g(x)|\right]^2+\left[\max_{x\in[a,b]}|f(x)-g(x)|\right]^2=2\left[\max_{x\in[a,b]}|f(x)|\right]^2+2\left[\max_{x\in[a,b]}|g(x)|\right]^2.$$

Ga zelf na dat bijvoorbeeld voor $f(x)=1$ en $g(x)=x$ in $[0,1]$ hieraan niet voldaan is.

De norm (3.27) voor $p=2$ geeft wel aanleiding tot een scalair product. Men bewijst inderdaad gemakkelijk dat in $C([a,b],w(x))$

$$(f,g)=\int_a^b w(x)\overline{f(x)}g(x)\,dx \tag{3.41}$$

aan de definitie van scalair product voldoet. In reële ruimten valt het 'complex toegevoegde'-teken weg.

4. De verzameling $L_2[a,b]$ van de kwadratisch (Lebesgue-)integreerbare reële functies is een functieruimte die groter is dan de ruimte $C[a,b]$, omdat ze ook discontinue functies toelaat. In deze ruimte voldoet de integraal (3.41), geïnterpreteerd als een Lebesgue-integraal, aan de definitie van scalair product. Men bewijst gemakkelijk dat aan de vier voorwaarden voldaan is.

   De ruimten $L_p[a,b]$ met $p\neq 2$ zijn geen unitaire ruimten voor de L_p-norm. Er is immers opnieuw niet voldaan aan de parallellogramgelijkheid.

3.3.4 Het begrip orthogonaliteit

Door middel van het scalair product kunnen we de notie van orthogonaliteit veralgemenen.

DEFINITIE 3.3.9*(orthogonaliteit)*
Twee vectoren $\vec{x}$ en $\vec{y}$ in unitaire ruimte V noemt men orthogonaal*, genoteerd als $\vec{x}\perp\vec{y}$, wanneer $(\vec{x},\vec{y})=0$.*

Men zegt dat een vector $\vec{x}$ orthogonaal is tot (of ten opzichte van) een deelverzameling $\mathcal{D}$ van een unitaire ruimte V, genoteerd als $\vec{x}\perp\mathcal{D}$, als $\vec{x}$ orthogonaal is tot alle vectoren van $\mathcal{D}$. Een gevolg van de definitie van orthogonaliteit is de stelling van Pythagoras, die nu veralgemeend kan worden tot willekeurige unitaire ruimten.

STELLING 3.3.7 *(Pythagoras)*
Wanneer $\vec{x}\perp\vec{y}$ is in een unitaire ruimte V, dan geldt t.o.v. de natuurlijke norm in V dat

$$\|\vec{x}+\vec{y}\|^2=\|\vec{x}\|^2+\|\vec{y}\|^2\ . \tag{3.42}$$

Bewijs Het gestelde volgt triviaal uit een expansie van $\|\vec{x}+\vec{y}\|^2=(\vec{x}+\vec{y},\vec{x}+\vec{y})$. □

3.4 Benaderen in een unitaire ruimte

We beschouwen beste benaderingen in een eindigdimensionale deelruimte van een eventueel oneindigdimensionale unitaire ruimte. Denk bijvoorbeeld aan de ruimte $P_n[a,b]$, de ruimte van alle veeltermen op $[a,b]$ met graad kleiner of gelijk aan n, als deelruimte van $C[a,b]$. Stap voor stap laten we zien hoe concepten, resultaten en algoritmen behandeld in Hoofdstuk 2 veralgemenen naar een abstracte unitaire ruimte.

3.4.1 De grammatrix van een stel vectoren

DEFINITIE 3.4.10 *(grammatrix)*
De grammatrix van een stel vectoren $\{\vec{a}_1, \vec{a}_2, \ldots, \vec{a}_n\}$ is de matrix

$$G(\vec{a}_1, \vec{a}_2, \ldots, \vec{a}_n) = \begin{bmatrix} (\vec{a}_1, \vec{a}_1) & (\vec{a}_1, \vec{a}_2) & \cdots & (\vec{a}_1, \vec{a}_n) \\ (\vec{a}_2, \vec{a}_1) & (\vec{a}_2, \vec{a}_2) & \cdots & (\vec{a}_2, \vec{a}_n) \\ \cdot & \cdot & \cdots & \cdot \\ \cdot & \cdot & \cdots & \cdot \\ (\vec{a}_n, \vec{a}_1) & (\vec{a}_n, \vec{a}_2) & \cdots & (\vec{a}_n, \vec{a}_n) \end{bmatrix}. \tag{3.43}$$

Deze matrix, die we verkort voorstellen door G, laat toe om na te gaan of vectoren $\{\vec{a}_1, \ldots, \vec{a}_n\}$ lineair onafhankelijk zijn.

STELLING 3.4.8
Grammatrix (3.43) is Hermitiaans positief definiet als en alleen als vectoren $\{\vec{a}_1, \vec{a}_2, \ldots, \vec{a}_n\}$ lineair onafhankelijk zijn.

Bewijs Uit de derde voorwaarde voor een scalair product in (3.31) volgt dat $G = G^*$, dus G is een Hermitiaanse matrix. Zij $\upsilon = [v_1 \ \cdots \ v_n]^T$ een willekeurige vector in $\mathbb{C}^n$. Steunend op de lineariteit van het scalair product kunnen we de uitdrukking $\upsilon^* G \upsilon$ herschrijven als

$$\begin{aligned} \upsilon^* G \upsilon &= [\bar{v}_1 \ \cdots \ \bar{v}_n] \begin{bmatrix} (\vec{a}_1, \vec{a}_1) & (\vec{a}_1, \vec{a}_2) & \cdots & (\vec{a}_1, \vec{a}_n) \\ (\vec{a}_2, \vec{a}_1) & (\vec{a}_2, \vec{a}_2) & \cdots & (\vec{a}_2, \vec{a}_n) \\ \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot \\ (\vec{a}_n, \vec{a}_1) & (\vec{a}_n, \vec{a}_2) & \cdots & (\vec{a}_n, \vec{a}_n) \end{bmatrix} \begin{bmatrix} v_1 \\ v_2 \\ . \\ . \\ v_n \end{bmatrix} \\ &= [\bar{v}_1 \ \cdots \ \bar{v}_n] \begin{bmatrix} (\vec{a}_1, v_1\vec{a}_1 + v_2\vec{a}_2 + \cdots + v_n\vec{a}_n) \\ (\vec{a}_2, v_1\vec{a}_1 + v_2\vec{a}_2 + \cdots + v_n\vec{a}_n) \\ \cdot \\ \cdot \\ (\vec{a}_n, v_1\vec{a}_1 + v_2\vec{a}_2 + \cdots + v_n\vec{a}_n) \end{bmatrix} \\ &= (v_1\vec{a}_1 + v_2\vec{a}_2 + \cdots + v_n\vec{a}_n, v_1\vec{a}_1 + v_2\vec{a}_2 + \cdots + v_n\vec{a}_n) \\ &= \| \, v_1\vec{a}_1 + v_2\vec{a}_2 + \cdots + v_n\vec{a}_n \, \|^2 \ . \end{aligned}$$

Als $\{\vec{a}_1, \vec{a}_2, \ldots, \vec{a}_n\}$ lineair onafhankelijk zijn, dan geldt voor $\upsilon \neq 0$ dat $v_1\vec{a}_1 + v_2\vec{a}_2 + \cdots + v_n\vec{a}_n \neq 0$ en bijgevolg $\upsilon^* G \upsilon > 0$. Dit bewijst dat G positief definiet is.

Als de n vectoren $\{\vec{a}_1, \ldots, \vec{a}_n\}$ lineair afhankelijk zijn, dan bestaat er $\hat{v} = (\hat{v}_1, \ldots, \hat{v}_n) \neq 0$ zodat

$$\hat{v}_1\vec{a}_1 + \hat{v}_2\vec{a}_2 + \cdots + v_n\vec{a}_n = 0.$$

Bijgevolg is $\hat{v}^* G\hat{v} = 0$ wat $G\hat{v} = 0$ impliceert. Matrix G is dus singulier en niet Hermitiaans positief definiet. □

GEVOLG
Grammatrix (3.43) is inverteerbaar als en alleen als vectoren $\{\vec{a}_1, \vec{a}_2, \ldots, \vec{a}_n\}$ lineair onafhankelijk zijn.

Merk dat in de ruimte $\mathbb{C}^m$ de grammatrix zich herleidt tot $G = A^*A$, waarbij de kolommen van A overeenkomen met de basisvectoren, zie (2.3).

3.4.2 Orthogonale projector

We beginnen met een wiskundige definitie.

DEFINITIE 3.4.11 *(projector)*
Een projector *is een lineaire operator*[2] $P : V \to V$ *die idempotent is, dit wil zeggen* $P\,(P\,\vec{v}) = P\,\vec{v}$.

Elke vector $\vec{v} \in V$ kunnen we ontbinden als

$$\vec{v} = P\,\vec{v} + (I - P)\vec{v},$$

met I de identiteitsoperator. Je kan gemakkelijk volgende eigenschappen nagaan: er geldt $P\,\vec{v} \in \mathcal{R}(P)$ en $(I - P)\;\vec{v} \in \mathcal{N}(P)$, waarbij $\mathcal{R}(P) = \{P\,\vec{x} :\; \vec{x} \in V\}$ en $\mathcal{N}(P) = \{\vec{x} \in V : P\,\vec{x} = 0\}$. Bovendien geldt $\mathcal{R}(P) \cap \mathcal{N}(P) = \{\vec{0}\}$, is bovenstaande ontbinding van $\vec{v}$ uniek, en geldt dat $\mathcal{N}(P) = \mathcal{R}(I - P)$. Dus P projecteert op $\mathcal{R}(P)$ volgens de richting van $\mathcal{N}(P)$.

DEFINITIE 3.4.12 *(orthogonale projector)*
Een projector P *is* orthogonaal *indien* $\mathcal{R}(P)$ *en* $\mathcal{N}(P)$ *onderling orthogonale ruimten zijn.*

De veralgemening van Eigenschap 2.1 is als volgt.

Eigenschap 3.1 *Een projector* P *is orthogonaal als en alleen als*

$$(P\;\vec{w}, \vec{v}) = (\vec{w}, P\;\vec{v}), \quad \forall\vec{v},\; \vec{w} \in V. \tag{3.44}$$

Bewijs Voorwaarde (3.44) is nodig. Als $\mathcal{R}(P)$ en $\mathcal{N}(P)$ orthogonaal zijn, dan is voor alle $\vec{v}, \vec{w} \in V$ voldaan aan

$$(P\,\vec{w}, (I - P)\,\vec{v}) = \vec{0} = ((I - P)\,\vec{w}, P\,\vec{v}),$$

[2]Een operator beeldt een element van een ruimte af op een element van een (ev. andere) ruimte. Een lineaire operator veralgemeent een matrix. Het beeld van $\vec{v}$ door operator P noteren we als $P\,\vec{v}$.

waaruit volgt dat $(P\,\vec{w}, \vec{v}) = (P\,\vec{w}, P\,\vec{v}) = (\vec{w}, P\,\vec{v})$.
Voorwaarde(3.44) is voldoende. Neem willekeurige $\vec{x} = P\,\vec{u} \in \mathcal{R}(P)$ en $\vec{y} \in \mathcal{N}(P)$. Dan is

$$(\vec{x}, \vec{y}) = (P\,\vec{u}, \vec{y}) = (\vec{u}, P\,\vec{y}) = (\vec{u}, \vec{0}) = \vec{0},$$

wat de orthogonaliteit bewijst. □

Zij $\mathcal{D}$ een gegeven eindigdimensionale deelvectorruimte van een (eventueel oneindigdimensionale) vectorruimte V. Veronderstel dat de dimensie van $\mathcal{D}$ gelijk is aan n en een basis gegeven wordt door $\{\vec{a}_1, \ldots, \vec{a}_n\}$. We construeren nu de projector $P_{\mathcal{D}}$ die orthogonaal projecteert op $\mathcal{D}$. Hiervoor tonen we aan dat een vector $\vec{v} \in V$ uniek ontbonden kan worden als

$$\vec{v} = \vec{v}_1 + \vec{v}_2 \;\text{ met }\; \vec{v}_1 \in \mathcal{D} \;\text{ en }\; \vec{v}_2 \perp \mathcal{D}. \tag{3.45}$$

De afbeelding van $\vec{v}$ naar $\vec{v}_1$ zal lineair blijken zodat $\vec{v}_1$ dan gelijk is aan $P_{\mathcal{D}}\,\vec{v}$.
Vermits $\vec{v}_1 \in \mathcal{D}$ kunnen we deze vector ontbinden als

$$\vec{v}_1 = c_1\vec{a}_1 + c_2\vec{a}_2 + \ldots + c_n\vec{a}_n. \tag{3.46}$$

Als we $(\vec{v} - \vec{v}_1) \perp \mathcal{D}$ opleggen, levert ons dat n vergelijkingen op, namelijk

$$(\vec{a}_k, \vec{v} - \vec{v}_1) = 0, \quad k = 1, \ldots, n.$$

Wanneer we hierin $\vec{v}_1$ vervangen door (3.46) en gebruik maken van de lineariteit van het scalair product, dan bekomen we volgend stelsel:

$$\left\{\begin{array}{lcl} c_1(\vec{a}_1, \vec{a}_1) + c_2(\vec{a}_1, \vec{a}_2) + \cdots + c_n(\vec{a}_1, \vec{a}_n) &=& (\vec{a}_1, \vec{v}) \\ c_1(\vec{a}_2, \vec{a}_1) + c_2(\vec{a}_2, \vec{a}_2) + \cdots + c_n(\vec{a}_2, \vec{a}_n) &=& (\vec{a}_2, \vec{v}) \\ \vdots & & \vdots \\ c_1(\vec{a}_n, \vec{a}_1) + c_2(\vec{a}_n, \vec{a}_2) + \cdots + c_n(\vec{a}_n, \vec{a}_n) &=& (\vec{a}_n, \vec{v}) \end{array}\right. . \tag{3.47}$$

Dit stelsel noemt men het *normaalstelsel*. Omdat vectoren $\{\vec{a}_1, \ldots, \vec{a}_n\}$ een basis vormen en dus lineair onafhankelijk zijn, is de matrix van het stelsel, die we identificeren als de grammatrix G, inverteerbaar, zie Stelling 3.4.8, en heeft het stelsel een unieke oplossing:

$$\begin{bmatrix} c_1 \\ \vdots \\ c_n \end{bmatrix} = G^{-1} \begin{bmatrix} (\vec{a}_1, \vec{v}) \\ \vdots \\ (\vec{a}_n, \vec{v}) \end{bmatrix}. \tag{3.48}$$

We concluderen dat de orthogonale projector op $\mathcal{D} =< \vec{a}_1, \ldots, \vec{a}_n >$ gedefinieerd wordt door

$$P_{\mathcal{D}}\;\vec{v} = \sum_{i=1}^{n} c_i\;\vec{a}_i, \qquad \vec{v} \in V, \tag{3.49}$$

waarbij de coëfficiënten $c_i,\ i = 1, \ldots, n$, gegeven zijn door (3.48).

Merk dat (3.48)-(3.49) een directe veralgemening is van uitdrukking (2.13), die

$$P_{\mathcal{D}}\ v = A(A^*A)^{-1}(A^*v), \quad v \in \mathbb{C}^m, \tag{3.50}$$

impliceert. Immers, het product van A in (3.50) met wat volgt veralgemeent tot ontbinding (3.49) in basisvectoren, matrix A^*A veralgemeent tot de grammatrix, en het product A^*v tot de scalaire producten tussen de basisvectoren en vector $\vec{v}$.

Als de basis orthogonaal is, dan wordt de matrix van het stelsel (3.47) een diagonaalmatrix en kunnen we de oplossing uitdrukken als

$$c_i = \frac{(\vec{a}_i, \vec{v})}{\| \vec{a}_i \|^2},\ i = 1, \ldots, n.$$

De orthogonale projectie wordt dan

$$P_{\mathcal{D}}\ \vec{v} = \sum_{i=1}^{n} \frac{(\vec{a}_i, \vec{v})}{\| \vec{a}_i \|^2}\ \vec{a}_i, \quad \vec{v} \in V. \tag{3.51}$$

Is de basis bovendien orthonormaal verkrijgen we tenslotte

$$P_{\mathcal{D}}\ \vec{v} = \sum_{i=1}^{n} (\vec{a}_i, \vec{v})\ \vec{a}_i, \quad \vec{v} \in V, \tag{3.52}$$

zie ter vergelijking (2.16).

3.4.3 Orthogonalisatieprocedures

We vertrekken van een basis $\{\vec{a}_1, \ldots, \vec{a}_n\}$ van een n-dimensionale deelruimte $\mathcal{D}$ van V en wensen deze te vervangen door een *geneste orthonormale basis* $\{\vec{q}_1, \ldots, \vec{q}_n\}$, in de betekenis van (2.17). Deze gewenste eigenschap kunnen we uitdrukken in de vorm

$$\begin{array}{rcl} \vec{a}_1 & = & r_{11}\vec{q}_1 \\ \vec{a}_2 & = & r_{12}\vec{q}_1 + r_{22}\vec{q}_2 \\ \vec{a}_3 & = & r_{13}\vec{q}_1 + r_{23}\vec{q}_2 + r_{33}\vec{q}_3 \\ \vdots & & \ddots \\ \vec{a}_n & = & r_{1n}\vec{q}_1 + r_{2n}\vec{q}_2 + \cdots + r_{nn}\vec{q}_n, \end{array} \tag{3.53}$$

met $r_{ij} \in \mathbb{C}$ ($r_{ij} \in \mathbb{R}$ bij een vectorruimte over $\mathbb{R}$). Merk dat een ontbinding van de vorm (3.53) een directe veralgemening is van de (onvolledige) QR-factorisatie van een matrix.

We kunnen de orthogonale basis en de coëfficiënten rechtstreeks berekenen, gebruik makend van de opgelegde orthogonaliteitseis op de nieuwe basis:

stap 1: $r_{11} = \|\vec{a}_1\|$, $\vec{q}_1 = \vec{a}_1/r_{11}$.
stap j: Stel $\vec{q}_1, \vec{q}_2, ..., \vec{q}_{j-1}$ en r_{ik}, met $k < j$, gekend.
$\vec{a}_j = r_{1j}\ \vec{q}_1 + \vec{r}_{2j}\ \vec{q}_2 + \cdots + r_{j-1j}\ \vec{q}_{j-1} + r_{jj}\ \vec{q}_j$.
Wegens de orthogonaliteit van vectoren $\vec{q}_i$ geldt: $(\vec{q}_i, \vec{a}_j) = r_{ij}$. Verder is
$r_{jj}\vec{q}_j = \vec{a}_j - r_{1j}\ \vec{q}_1 - \vec{r}_{2j}\ \vec{q}_2 - \cdots - r_{j-1j}\ \vec{q}_{j-1} := \vec{v}_j$,
dus $\vec{q}_j$ is gekend op een constante na en we weten dat $\|\vec{q}_j\| = 1$.
We kiezen $r_{jj} = \|\vec{v}_j\|$ en $\vec{q}_j = \vec{v}_j/r_{jj}$.

Deze stappen leiden tot Algoritme 5, dat het klassieke Gram-Schmidt algoritme veralgemeent. In vergelijking met Algoritme 1 zijn inwendige producten tussen vectoren in $\mathbb{C}^m$ vervangen door scalaire producten in V, en de Euclidische norm door de natuurlijke norm geïnduceerd door het scalair product, dit is $\|\cdot\| = \sqrt{(\cdot,\cdot)}$.

Algoritme 5 Klassiek Gram-Schmidt algoritme in unitaire ruimte V

```
1: for j = 1 to n do
2:    v_j = a_j
3:    for i = 1 to j − 1 do
4:       r_ij = (q_i, a_j)
5:       v_j = v_j − r_ij q_i
6:    end for
7:    r_jj = ||v_j||
8:    q_j = v_j/r_jj
9: end for
```

Ook de meetkundige interpretatie die we in Hoofdstuk 2 gemaakt hebben blijft geldig. In de j-de stap van Algoritme 5 wordt $\vec{v}_j$ opgebouwd tot

$$\begin{aligned}\vec{v} &= \vec{a}_j - r_{1j}\vec{q}_1 - r_{2j}\vec{q}_2 - \cdots - r_{j-1j}\vec{q}_{j-1} \\ &= \vec{a}_j - (\vec{q}_1, \vec{a}_j)\vec{q}_1 - (\vec{q}_2, \vec{a}_j)\vec{q}_2 - \cdots - (\vec{q}_{j-1}, \vec{a}_j)\,\vec{q}_{j-1} \\ &= \vec{a}_j - P_{<\vec{q}_1,\ldots,\vec{q}_{j-1}>}\vec{a}_j,\end{aligned}$$

waarbij de laatste stap volgt uit (3.52). De procedure kan dus als volgt begrepen worden.

- Vector $\vec{q}_1$ wordt bekomen uit $\vec{a}_1$ door normalisatie.
- Om $\vec{q}_2$ te bepalen projecteren we $\vec{a}_2$ orthogonaal op $< \vec{q}_1 >$. We trekken deze projectie af van $\vec{a}_2$, zodat de resulterende vector $\vec{v}_2$ orthogonaal is t.o.v. $\vec{q}_1$. Na normalisatie volgt $\vec{q}_2$.
- Stel dat we $\vec{q}_1, \ldots, \vec{q}_{j-1}$ berekend hebben. Om $\vec{q}_j$ te bepalen projecteren we $\vec{a}_j$ loodrecht op $< \vec{q}_1, \ldots, q_{j-1} >$. We trekken deze projectie af van $\vec{a}_j$, zodat de resulterende vector $\vec{v}_j$ orthogonaal is t.o.v. de ruimte $< \vec{q}_1, \ldots, \vec{q}_{j-1} >$. Na normalisatie volgt de nieuwe basisvector $\vec{q}_j$.

Voorbeeld 3.2 *Beschouw de ruimte $L_2[-1, 1]$ met het scalair product (3.41) waarbij $w(x) \equiv 1$. Een lineair onafhankelijk stel vectoren in deze ruimte is het stel { 1, x, x^2, x^3*

}. Het stel is een basis voor de deelruimte $P_3[-1,1]$. We zullen voor deze deelruimte een orthonormale basis opstellen. De basisvectoren zullen we aanduiden met $P_0(x), P_1(x), P_2(x)$ en $P_3(x)$. We gebruiken het Gram-Schmidt algoritme.

- *Eerste iteratiestap:*

$$r_{11} = \sqrt{\int_{-1}^{1} 1\,dt} = \sqrt{2}, \quad P_0(x) = \frac{1}{\sqrt{2}}$$

- *Tweede iteratiestap:*

$$\begin{aligned} r_{12} &= (P_0(x), x) = \textstyle\int_{-1}^{1} \frac{t}{\sqrt{2}}\,dt = 0 \\ v_2 &= x - 0\frac{1}{\sqrt{2}} = x \\ r_{22} &= \sqrt{\textstyle\int_{-1}^{1} t^2\,dt} = \sqrt{\frac{2}{3}}, \quad P_1(x) = \sqrt{\frac{3}{2}}x \end{aligned}$$

- *Derde iteratiestap:*

$$\begin{aligned} r_{13} &= (P_0(x), x^2) = \textstyle\int_{-1}^{1} \frac{t^2}{\sqrt{2}}\,dt = \frac{2}{3\sqrt{2}}, \quad r_{23} = (P_1(x), x^2) = 0 \\ v_3 &= x^2 - \frac{2}{3\sqrt{2}}\frac{1}{\sqrt{2}} = x^2 - \frac{1}{3} \\ r_{33} &= \sqrt{\textstyle\int_{-1}^{1} (t^2 - \frac{1}{3})^2\,dt} = \frac{2\sqrt{2}}{3\sqrt{5}}, \quad P_2(x) = \frac{\sqrt{5}}{2\sqrt{2}}\ (\ 3x^2 - 1) \end{aligned}$$

- *Vierde iteratiestap:*

$$\begin{aligned} r_{14} &= 0,\ r_{24} = (P_1(x), x^3) = \textstyle\int_{-1}^{1} \sqrt{\frac{3}{2}} t^4\,dt = \frac{\sqrt{6}}{5},\ r_{34} = 0 \\ v_4 &= x^3 - \frac{\sqrt{6}}{5}\sqrt{\frac{3}{2}}x = x^3 - \frac{3}{5}x \\ r_{44} &= \sqrt{\textstyle\int_{-1}^{1} \left(t^3 - \frac{3}{5}t\right)^2\,dt} = \frac{2\sqrt{2}}{5\sqrt{7}}, \quad P_3(x) = \frac{\sqrt{7}}{\sqrt{2}}\left(\frac{5}{2}x^3 - \frac{3}{2}x\right) \end{aligned}$$

De twee sets van basisfuncties zijn afgebeeld in Figuur 3.9. De coëfficiënten r_{ij}, horende bij de genomen lineaire combinaties van vectoren, kunnen we groeperen in matrix

$$\hat{R} = \begin{bmatrix} \sqrt{2} & 0 & \frac{2}{3\sqrt{2}} & 0 \\ 0 & \frac{\sqrt{2}}{\sqrt{3}} & 0 & \frac{\sqrt{6}}{5} \\ 0 & 0 & \frac{2\sqrt{2}}{3\sqrt{5}} & 0 \\ 0 & 0 & 0 & \frac{2\sqrt{7}}{5\sqrt{7}} \end{bmatrix}. \tag{3.54}$$

Ook het gewijzigde Gram-Schmidt algoritme (Algoritme 2) en het proces van *herorthogonalisatie* blijven toepasbaar. De aanpassingen zijn zoals deze aan Algoritme 1.

Opmerking 3.1 *In tegenstelling tot de Gram-Schmidt gebaseerde algoritmen kunnen de orthogonalisatieprocedures gebaseerd op unitaire transformaties uit §2.3.4 niet veralgemeend worden naar abstracte unitaire ruimten. De reden is aangehaald in Opmerking 2.2. Bij Gram-Schmidt algoritmen vormt men matrix A om tot matrix $\hat{Q}$ met orthonormale kolommen door lineaire combinaties te nemen van de kolommen, wat op triviale wijze veralgemeent naar lineaire combinaties van basisvectoren (in het voorbeeld hierboven komt dit neer op lineaire combinaties van functies). Bij de methodes uit §2.3.4 vormt men echter matrix A om tot matrix R door lineaire combinaties te nemen van rijen. In de abstracte setting worden de kolommen van A bijvoorbeeld vervangen door functies, terwijl $\hat{R}$ een (klassieke) matrix van coëfficiënten r_{ij} blijft, zie (3.54), wat al een compatibiliteitsprobleem oplevert.*

3.4.4 Beste benadering in een deelruimte

Met Stelling 3.2.3 hebben we reeds aangetoond dat de beste benadering voor een vector $\vec{v} \in V$ in een n-dimensionale deelruimte $\mathcal{D} =< \vec{a}_1, \ldots, \vec{a}_n >$ bestaat en uniek is. In tegenstelling tot het bewijs van Stelling 2.4.1 hebben we daarvoor het bestaan van een scalair product en de gerelateerde notie van orthogonaliteit niet gebruikt, enkel de eigenschap dat de beschouwde norm een strikte norm is. Herinner dat aan deze voorwaarde voldaan is voor de natuurlijke norm in een unitaire ruimte.
In een unitaire ruimte kunnen we een stap verder gaan.

Stelling 3.4.9 *(orthogonale projectiestelling)*
De beste benadering van vector $\vec{v}$ in deelruimte $\mathcal{D}$ van unitaire ruimte V met betrekking tot de natuurlijke norm wordt gegeven door

$$\vec{\hat{y}} = P_{\mathcal{D}}\, \vec{v},$$

met $P_{\mathcal{D}}$ de orthogonale projector op $\mathcal{D}$, gegeven door (3.48)-(3.49) in het algemene geval, door (3.51) voor een orthogonale basis en door (3.52) voor een orthonormale basis.

Zoals reeds geïllustreerd in Sectie 2.1 en Sectie 2.4, bieden orthogonale basissen heel wat voordelen, en wordt het direct oplossen van het normaalstelsel best vermeden. We geven nog een voorbeeld waar benaderingen gezocht worden in een functieruimte, dat aansluit bij Voorbeeld 3.2.

Voorbeeld 3.3 *Beschouw de ruimte $L_2[-1,1]$ met het scalair product (3.41) waarbij $w(x) \equiv 1$. We beschouwen de deelruimte $P_n[-1,1]$, waarbij basisvector $\vec{a}_i$ overeenkomst met de functie x^{i-1} voor $i = 1, \ldots, n+1$. In Tabel 3.1 wordt het conditiegetal van de grammatrix $G_n(\vec{a}_1, \ldots, \vec{a}_{n+1})$ getoond in functie van n. We merken de sterke groei op.*

Als het vertrekpunt van het benaderingsprobleem een niet-orthogonale basis $\{\vec{a}_1, \ldots, \vec{a}_n\}$ is, bestaat een aangewezen algoritme uit de volgende stappen.

- Vervang de basis door een orthonormale basis, $\{\vec{q}_1, \ldots, \vec{q}_n\}$. Deze laatste kan bepaald worden door Algoritme 5, eventueel uitgebreid met herorthogonalisatie.
  De beste banadering van $\vec{v}$ wordt dan gegeven door $\vec{\hat{y}} = \sum_{i=1}^{n} d_i \vec{q}_i$, met $d_i = (\vec{q}_i, \vec{v})$.

n	3	6	9	12	15	18
$\kappa(G_n)$	6.760e+01	1.023e+04	1.685e+06	2.960e+08	5.285e+10	9.676e+12

Tabel 3.1: Conditiegetal van de grammatrix, voor de monomiale veeltermbasis.

- (Enkel) indien de coördinaten $c = [c_1 \ \cdots \ c_n]^T$ gevraagd worden in de oorspronkelijke basis, dit is $\vec{\hat{y}} = \sum_{i=1}^{n} c_i \vec{a}_i$, los dan het stelsel $\hat{R}c = d$ op, met $d = [d_1 \ \cdots \ d_n]^T$ en

$$\hat{R} = \begin{bmatrix} r_{11} & \cdots & r_{1n} \\ & \ddots & \vdots \\ 0 & & r_{nn} \end{bmatrix}.$$

Conceptueel is deze aanpak dezelfde als deze voor het oplossen van het kleinstekwadratenprobleem in §2.4.2, zie Opmerking 2.4.

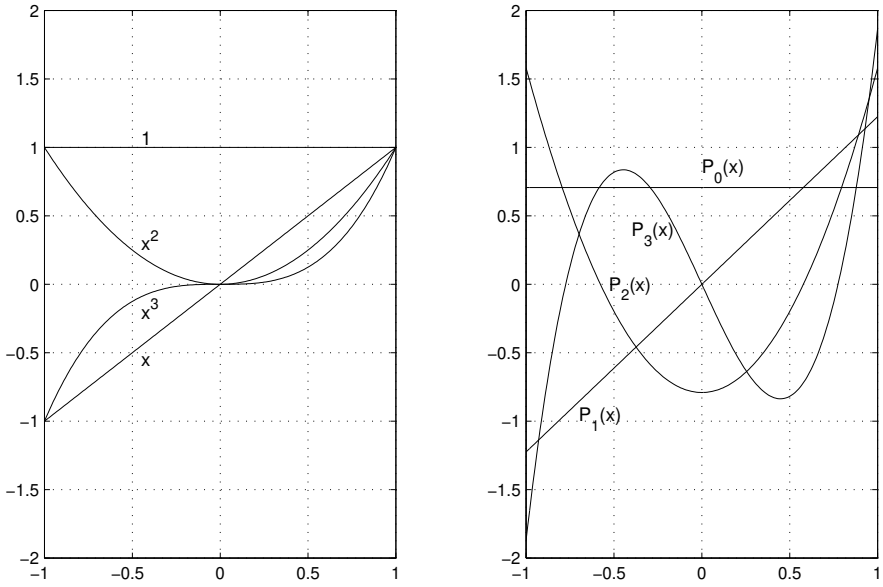
Voorbeeld 3.4 *We gaan verder met Voorbeeld 3.2. De beste benadering, g, van een gegeven functie, f, door een veelterm van graad drie in de beschouwde unitaire ruimte wordt gegeven door*

$$g(x) = \sum_{i=1}^{4} d_i \, P_{i-1}(x), \quad \text{met } d_i = (P_{i-1}(x), f(x)) = \int_{-1}^{1} P_{i-1}(x) f(x) \, dx.$$

In de oorspronkelijke basis geldt

$$g(x) = \sum_{i=1}^{n} c_i \, x^{i-1},$$

waarbij $c = \hat{R}^{-1} d$ en $\hat{R}$ gegeven is door (3.54).



Figuur 3.9: Een 'schuine' basis (links) en een orthonormale basis (rechts) voor $P_3[-1,1]$.

Hoofdstuk 4

Benadering door middel van veeltermen

4.1 Inleiding

4.1.1 Het kleinste-kwadratencriterium

Het benaderen van een functie bestaat erin een andere functie te zoeken die voor verdere bewerkingen beter geschikt is en die goed genoeg met de gegeven functie overeenstemt. In dit hoofdstuk zullen we vooropstellen dat de benadering een veelterm moet zijn, dit omwille van de eenvoud van evaluatie en manipulatie.
De benadering moet 'goed genoeg' met de gegeven functie overeenstemmen, en meestal wenst men zelfs de 'beste' benadering uit de gegeven verzameling. De benamingen 'goed genoeg' en 'beste' benadering hebben maar zin als men een criterium opgeeft dat als maat kan dienen voor de afwijking tussen de te benaderen functie $f(x)$ en de mogelijke benaderingen $y(x)$. Zo'n criterium moet realistisch zijn in viervoudig opzicht.

- De gestelde eis moet een praktische betekenis hebben. In sommige gevallen wordt die eis opgelegd uit fysische overwegingen. In de meeste gevallen echter zal ze volgen uit intuïtieve bedenkingen omtrent wat men bedoelt met een goede benadering.

- Het criterium moet zodanig zijn dat het mogelijk is in de verzameling Y van alle potentiële benaderingen uit te maken of een benadering $y_1(x)$ beter aan het criterium voldoet dan een andere benadering $y_2(x)$. Het criterium moet dus een ordening van de verzameling Y teweegbrengen. Hieraan zal bijvoorbeeld voldaan zijn wanneer het criterium afgeleid is van een afstandsfunctie in een metrische ruimte.

- De eis die door het criterium gesteld wordt, mag niet te streng zijn. Er moet een redelijke kans zijn dat men eraan kan voldoen. Zo zou het onredelijk zijn van een benaderende functie te eisen dat zij in elk punt van een interval een kleinere afwijking zou geven dan de andere functies uit de verzameling mogelijke benaderingen. Een dergelijke eis is zo streng dat men er in de praktijk nooit zal kunnen aan voldoen.

– De beste benadering moet niet alleen bestaan, ze moet ook kunnen gevonden worden. Het moet mogelijk zijn een algoritme op te stellen dat toelaat de beste benadering te bepalen; soms zal dit in een eindig aantal berekeningsstappen kunnen gebeuren, soms zal men zich moeten tevreden stellen met een iteratief algoritme.

In dit hoofdstuk beperken we ons tot het benaderen volgens het kleinste-kwadratencriterium. Dit criterium steunt op de gewogen kwadratische afstand in een unitaire functieruimte van continue functies. Als afwijking of afstand tussen de te benaderen functie $f(x)$ en de mogelijke benaderingen $y(x)$ beschouwt men de integraal

$$\int_a^b w(x)\ |f(x) - y(x)|^2\ dx. \tag{4.1}$$

Dit criterium voldoet aan de vier voorwaarden die hierboven opgesomd werden.

– De praktische betekenis is er voornamelijk in gelegen dat men rekening houdt met de waarde van de fout in heel het interval of in alle gegeven punten.

– Vermits dit criterium uitgedrukt wordt in termen van een integraal die een reëel getal als waarde heeft, kan men steeds van twee benaderingen uitmaken welke het best aan het criterium voldoet, namelijk deze waarvoor de integraal het kleinst is.

– Eén van de eigenschappen van het kleinste-kwadratencriterium is wel dat het onder zeer brede voorwaarden voor $w(x)$, $f(x)$ en de verzameling Y tot een oplossing leidt, en dat deze oplossing enig is. Dit heeft te maken met het feit dat de afstandsfunctie (4.1) afkomstig is van een scalair product.

– Het criterium is zeer gemakkelijk toe te passen.

4.1.2 Een unitaire ruimte

In het beschouwde *continue benaderingsprobleem* zullen we meestal functies benaderen uit de ruimte van de continue functies $C[a, b]$ of de meer algemene ruimte $L_2[a, b]$, de verzameling van functies f waarvoor $\int_a^b |f(x)|^2\, dx$ bestaat en eindig is. Het interval $[a, b]$ kan een eindig, half-oneindig of oneindig interval zijn. We zullen ons beperken tot *reële* functieruimten. In deze ruimten is de volgende uitdrukking een scalair product

$$(f, g) = \int_a^b w(x)\, f(x)\, g(x)\, dx, \tag{4.2}$$

met $w(x)$ een gegeven, positieve en integreerbare gewichtsfunctie,

$$w(x) > 0 \quad \text{voor} \quad x \in [a, b] \quad \text{en} \quad \int_a^b w(x)\, dx < \infty. \tag{4.3}$$

Indien het benaderingsinterval een half-oneindig of oneindig interval is, dan zullen we eveneens veronderstellen dat alle uitdrukkingen van de vorm

$$m_k = \int_a^b w(x)\, x^k\, dx\ , \quad k = 0, 1, 2, \ldots \tag{4.4}$$

bestaan. Deze integralen noemt men de *momenten* van de gewichtsfunctie. Het bestaan ervan garandeert dat alle veeltermen tot de gewogen L_2-ruimte behoren. Dit is een noodzakelijke voorwaarde om aan veeltermbenadering te kunnen doen.
Merk dat (4.1) de natuurlijke afstand is met betrekking tot scalair product (4.2). Het probleem dat centraal staat in dit hoofdstuk is het volgende:

Zoek een veelterm van graad n die de afstand (4.1) minimaal maakt.

Zulke veelterm kan uitgedrukt worden als

$$y_n(x) = \sum_{k=0}^{n} a_k \phi_k(x),$$

waarbij $\{\phi_0(x), \phi_1(x), \ldots, \phi_n(x)\}$ een basis is van de ruimte $P_n[a, b]$.

4.1.3 Gebruik van de klassieke monomiale basis

De meest voor de hand liggende keuze voor de basisfuncties is de klassieke *monomiale basis*, waarbij

$$\phi_k(x) = x^k \quad \text{voor} \quad k = 0, \ldots, n. \tag{4.5}$$

Deze functies zijn lineair onafhankelijk en spannen een eindigdimensionale ruimte op. Vermits we te maken hebben met unitaire ruimten mogen we de theorie van Sectie 3.3 zonder meer toepassen. De coëfficiënten van de benadering in de basis (4.5) worden gevonden als oplossing van het normaalstelsel (3.47).

In Voorbeeld 3.3 hebben we reeds geïllustreerd dat de monomiale veeltermbasis best vermeden wordt. We geven nog een tweede illustratie, waarbij een ander interval beschouwd wordt.

Voorbeeld 4.1 *We wensen een functie $f(x)$ te benaderen door een veelterm van de n-de graad, en dit over het interval $[0, 1]$. We veronderstellen $w(x) \equiv 1$. Dan is*

$$(\phi_k, \phi_j) = \int_0^1 x^{j+k}\, dx = \frac{1}{j+k+1}.$$

De coëfficiëntenmatrix van het normaalstelsel, de grammatrix, wordt dan

$$G = \begin{bmatrix} 1 & 1/2 & 1/3 & \cdots & 1/(n+1) \\ 1/2 & 1/3 & 1/4 & \cdots & 1/(n+2) \\ \vdots & \vdots & \vdots & & \vdots \\ 1/(n+1) & 1/(n+2) & 1/(n+3) & \cdots & 1/(2n+1) \end{bmatrix}. \tag{4.6}$$

Dit is een zogenaamde Hilbert-matrix van orde $n+1$. Hilbert-matrices zijn uiterst slecht geconditioneerde matrices. Voor $n=4$, bijvoorbeeld, is de inverse:

$$\begin{bmatrix} 1 & 1/2 & 1/3 & 1/4 & 1/5 \\ 1/2 & 1/3 & 1/4 & 1/5 & 1/6 \\ 1/3 & 1/4 & 1/5 & 1/6 & 1/7 \\ 1/4 & 1/5 & 1/6 & 1/7 & 1/8 \\ 1/5 & 1/6 & 1/7 & 1/8 & 1/9 \end{bmatrix}^{-1} = \begin{bmatrix} 25 & -300 & 1050 & -1400 & 630 \\ -300 & 4800 & -18900 & 26880 & -12600 \\ 1050 & -18900 & 79380 & -117600 & 56700 \\ -1400 & 26880 & -117600 & 179200 & -88200 \\ 630 & -12600 & 56700 & -88200 & 44100 \end{bmatrix}.$$

Deze inverse bevat zeer grote waarden, wisselend van teken. Dit geeft reeds aan dat afrondingsfouten in het rechterlid van het normaalstelsel zich zeer sterk zullen voortplanten. Wanneer we het conditiegetal berekenen, dan vinden we $\kappa(G) = 47661$. *Voor* $n = 10$ *vindt men* $\kappa(G) = 10^{13}$. *Dit betekent dat men bij het oplossen van het stelsel tot* 13 *decimale cijfers aan nauwkeurigheid kan verliezen!*

4.1.4 Gebruik van een orthogonale veeltermbasis

Wanneer bij het benaderen in een unitaire vectorruimte gebruik gemaakt wordt van een orthogonale basis is er, overeenkomstig Stelling 3.4.9, een expliciete uitdrukking voor de beste benadering, namelijk (3.51). In de notaties van dit hoofdstuk schrijven we

$$y_n(x) = \sum_{k=0}^{n} a_k \phi_k(x) \quad \text{met} \quad a_k = \frac{(\phi_k, f)}{(\phi_k, \phi_k)} \ . \tag{4.7}$$

Als we formule (4.7) wensen te gebruiken, dan moeten we eerst een orthogonale veeltermbasis met betrekking tot scalair product (4.2) opstellen. Hiervoor kunnen we bijvoorbeeld het Gram-Schmidt algoritme toepassen, vertrekkende van de monomiale basis (4.5). We verwijzen naar Algoritme 5 voor de pseudocode en Voorbeeld 3.2 voor een illustratie.

Merk dat we bij het opstellen van een rij orthogonale veeltermen nog een zekere keuzevrijheid hebben. Immers, wanneer $\psi_0(x), \psi_1(x), \psi_2(x), \ldots$ een orthogonale rij is, dan zijn ook de veeltermen van de rij met elementen $\phi_k(x) = \lambda_k \psi_k(x)$, met $\lambda_k \in \mathbb{R}_0$, orthogonaal. Algoritme 5 leidt tot een *orthonormale* basis in strikte zin, waarbij $\|\phi_k\| = 1$ voor $k = 0, \ldots, n$, met $\|\cdot\|$ de natuurlijke norm horende bij het scalair product. Andere veelgebruikte manieren om de basisfuncties te normaliseren zijn de volgende:

– Men kan λ_k zo kiezen dat $\phi_k(x)$ voor een zekere x een welbepaalde waarde aanneemt. Voor orthogonale veeltermen over $[a, b]$ legt men dikwijls de voorwaarde op dat

$$\phi_k(b) = 1 \ . \tag{4.8}$$

Aan deze voorwaarde kan voldaan worden als b geen wortel is van de vergelijking $\phi_k(x) = 0$. Het zal later blijken dat aan deze voorwaarde steeds voldaan is (Stelling 4.2.4).

– Soms neemt men λ_k zo dat de coëfficiënt van de hoogste macht van x in $\phi_k(x)$ een gegeven waarde aanneemt. Voor deze waarde neemt men dikwijls 1, d.w.z.

$$\phi_k(x) = x^k + c_{k-1}x^{k-1} + \cdots c_1 x + c_0 \ . \tag{4.9}$$

Dan verkrijgt men *monische* veeltermen. Deze normalisatie is slechts mogelijk als elke $\phi_k(x)$ van strikte graad k is. Ook aan deze voorwaarde zal steeds voldaan zijn.

Opmerking 4.1 *In Algoritme 5 kunnen we meteen (4.8) of (4.9) opleggen door lijn 7 aan te passen, waar de normalisatie gebeurt. In dat geval moeten we ook lijn 4 aanpassen tot* $r_{ij} = (\vec{q}_i, \vec{q}_j)/(\vec{q}_i, \vec{q}_i)$.

Eenmaal de orthogonale veeltermen bepaald, gebruikt men formule (4.7) voor het berekenen van de beste benadering van functie f. In het speciale geval waarbij f zelf een veelterm is van graad kleiner of gelijk aan n, is het rechterlid van (4.7) geen benadering maar een expansie van f.

4.2 Eigenschappen van orthogonale veeltermen

We zullen enkele eigenschappen bewijzen waaraan orthogonale veeltermen voldoen. Tenzij expliciet anders aangegeven, zullen de eigenschappen onafhankelijk zijn van de gewichtsfunctie en de ligging van het interval.
We zullen het steeds hebben over rijen van orthogonale veeltermen waarbij *opeenvolgende veeltermen* van *opeenvolgende graad* zijn. De eerste veelterm in de rij is van graad nul. De veeltermen zullen worden voorgesteld als $\phi_0(x), \phi_1(x), \phi_2(x), \ldots$ waarbij de index de graad van de veelterm aangeeft. Merk dat het resultaat van Algoritme 5, vertrekkende van de (geordende) monomiale basis, altijd aan deze voorwaarde voldoet.

4.2.1 Orthogonale veeltermen als basis in een unitaire ruimte

Stelling 4.2.1
Het stel orthogonale veeltermen $\phi_0(x), \phi_1(x), \ldots, \phi_n(x)$ *vormt een basis van de ruimte* $P_n[a, b]$.

Bewijs Het stel van $n+1$ veeltermen is een lineair onafhankelijk stel —dit volgt uit de inverteerbaarheid van de grammatrix (3.43) – in een ruimte met dimensie $n+1$. Bijgevolg is het stel een basis. □

Elke veelterm y_n van graad kleiner of gelijk aan n kan steeds geschreven worden in de vorm (4.7).

Stelling 4.2.2
Een veelterm die behoort tot een rij orthogonale veeltermen is ook orthogonaal tot alle veeltermen van een lagere graad.

Het bewijs van deze stelling is triviaal. De veelterm $\phi_k(x)$ is dus niet alleen orthogonaal tot alle veeltermen $\phi_i(x)$ voor $i < k$, maar ook tot de veeltermen $1, x, x^2, \ldots, x^{k-1}$, en alle lineaire combinaties ervan.

4.2.2 Een fundamentele recursiebetrekking

De rij orthogonale veeltermen van opeenvolgende graad kan worden opgesteld met behulp van de reeds geziene orthogonalisatieprocedures. Men vertrekt dan van de rij niet-orthogonale veeltermen x^k en men past het klassieke (Algoritme 5) of het verbeterde Gram-Schmidt algoritme toe. Beide rekenschema's vergen ongeveer evenveel rekenwerk. Het aantal uit te rekenen scalaire producten is van de orde $\mathcal{O}(n^2)$, waarbij n het aantal te berekenen veeltermen voorstelt.

We zullen nu aantonen dat orthogonale veeltermen van opeenvolgende graad voldoen aan een zeer eenvoudige recursiebetrekking. Deze betrekking laat toe de orthogonale basis op een veel efficiëntere manier op te stellen, en dit met een hoeveelheid rekenwerk (gemeten in aantal uit te rekenen scalaire producten) van orde $\mathcal{O}(n)$.

STELLING 4.2.3
De orthogonale veeltermen voldoen aan een drietermsrecursiebetrekking:

$$\begin{aligned} \phi_0(x) &= \lambda_0 \ , \quad \phi_1(x) = \lambda_1 \left(x - \frac{(x,1)}{(1,1)} \right) \phi_0(x), \\ \phi_k(x) &= \lambda_k \left((x - \alpha_k)\, \phi_{k-1}(x) - \beta_k\, \phi_{k-2}(x) \right) \quad \textit{voor} \ \ k \geq 2, \end{aligned} \tag{4.10}$$

waarbij

$$\alpha_k = \frac{(x\phi_{k-1}, \phi_{k-1})}{(\phi_{k-1}, \phi_{k-1})} \ , \quad \beta_k = \frac{(x\phi_{k-1}, \phi_{k-2})}{(\phi_{k-2}, \phi_{k-2})} \tag{4.11}$$

en λ_k volgt uit de gekozen normalisatievoorwaarde.

Bewijs Veeltermen $\phi_0(x)$ en $\phi_1(x)$ verkrijgt men door orthogonalisatie van 1 en x met de Gram-Schmidt methode. De waarden van λ_0 en λ_1 volgen uit de normalisatievoorwaarde. Vermits de veelterm $x\phi_{k-1}(x)$ van graad k is, kan ze ontbonden worden als een lineaire combinatie van de orthogonale veeltermen $\phi_0(x), \phi_1(x), \ldots, \phi_k(x)$. Formule (4.7) geeft

$$x\phi_{k-1}(x) = b_0\phi_0(x) + b_1\phi_1(x) + \cdots + b_k\phi_k(x), \tag{4.12}$$

met $b_l = \dfrac{(x\phi_{k-1}, \phi_l)}{(\phi_l, \phi_l)} = \dfrac{(\phi_{k-1}, x\phi_l)}{(\phi_l, \phi_l)}$ voor $l \leq k$.

De veelterm $x\phi_l(x)$ is van graad $l+1$. Uit Stelling 4.2.2 volgt dan dat b_l gelijk is aan nul wanneer $k-1 > l+1$, of nog, wanneer $l < k-2$. In het rechterlid van (4.12) blijven slechts 3 termen over,

$$x\phi_{k-1}(x) = b_{k-2}\phi_{k-2}(x) + b_{k-1}\phi_{k-1}(x) + b_k\phi_k(x) \ . \tag{4.13}$$

Dit herschrijven we als

$$\phi_k(x) = \frac{1}{b_k} \left((x - b_{k-1})\phi_{k-1}(x) - b_{k-2}\phi_{k-2}(x) \right) \ . \tag{4.14}$$

Identificatie met de vooropgestelde vorm in (4.10) levert de waarden van parameters α_k en β_k op. □

Opmerkingen

– Formule (4.10) is een betrekking die $\phi_k(x)$ uitdrukt als een combinatie van de twee vorige orthogonale veeltermen $\phi_{k-1}(x)$ en $\phi_{k-2}(x)$. Men spreekt daarom van een drietermsrecursiebetrekking of een recursiebetrekking van tweede orde. Bemerk dat $\phi_{k-1}(x)$ en $\phi_{k-2}(x)$ lineair optreden in deze betrekking. Men noemt de recursiebetrekking daarom lineair.

– De formules werden afgeleid voor een algemeen scalair product gedefinieerd op een vectorruimte over de reële getallen. Met gebruik van het continue scalair product voor een gewichtsfunctie $w(x)$ over een interval $[a, b]$ kunnen de coëfficiënten van de recursiebetrekking voluit geschreven worden als

$$\alpha_k = \frac{\int_a^b w(x)\, x\, \phi_{k-1}^2(x)\, dx}{\int_a^b w(x)\, \phi_{k-1}^2(x)\, dx}\ , \quad \beta_k = \frac{\int_a^b w(x)\, x\, \phi_{k-1}(x)\, \phi_{k-2}(x)\, dx}{\int_a^b w(x)\, \phi_{k-2}^2(x)\, dx}. \tag{4.15}$$

– Om het proces te starten bepaalt men λ_0 uit de normalisatievoorwaarde voor de nuldegraadsveelterm en bepaalt men vervolgens $\phi_1(x)$ volgens het orthogonalisatieprocédé van Gram-Schmidt. Men kan echter ook vertrekken van de formele veelterm $\phi_{-1}(x) \equiv 0$ en de nuldegraadsveelterm $\phi_0(x) \equiv \lambda_0$, om daarna de recursiebetrekking te gebruiken vanaf $k = 1$. Dat levert dezelfde orthogonale rij op.

Voorbeeld 4.2 *Nemen we weer het geval van orthogonale veeltermen voor $w(x) \equiv 1$ op $[-1, 1]$. In plaats van de algemene notatie ϕ_k zullen we de voor deze veeltermen klassieke notatie P_k aannemen. Als normalisatievoorwaarde eisen we dat de veeltermen monisch zijn.*

De recursiebetrekking wordt:

$$P_k(x) = \lambda_k \left((x - \alpha_k)P_{k-1}(x) - \beta_k P_{k-2}(x)\right)$$

met

$$\alpha_k = \frac{\int_{-1}^1 x\, P_{k-1}^2(x)\, dx}{\int_{-1}^1 P_{k-1}^2(x)\, dx} \quad en \quad \beta_k = \frac{\int_{-1}^1 x\, P_{k-1}(x)\, P_{k-2}(x)\, dx}{\int_{-1}^1 P_{k-2}^2(x)\, dx}.$$

- *We starten met $P_0(x) = 1$ en $P_1(x) = x$.*
- *Vervolgens stellen we $P_2(x) = \lambda_2 \left((x - \alpha_2)P_1(x) - \beta_2\right)$. We vinden $\alpha_2 = 0$ en*

$$\beta_2 = \frac{\int_{-1}^1 x^2\, dx}{\int_{-1}^1 1\, dx} = \frac{1}{3}\ .$$

De normalisatievoorwaarde voor een monische veelterm leidt tot $\lambda_2 = 1$.

Dus : $P_2(x) = x^2 - \frac{1}{3}$.

- *We nemen dan* $P_3(x) = \lambda_3\ ((x-\alpha_3)P_2(x) - \beta_3 P_1(x))$. *We vinden* $\alpha_3 = 0$ *en*

$$\beta_3 = \frac{\int_{-1}^{1} x\, P_2(x)\, P_1(x)\, dx}{\int_{-1}^{1} P_1^2(x)\, dx} = \frac{\int_{-1}^{1} x^2\, (x^2 - \frac{1}{3})\, dx}{\int_{-1}^{1} x^2\, dx} = \frac{4}{15}\ .$$

Er volgt dus dat $P_3(x) = \lambda_3(x\, P_2(x) - \frac{4}{15}) = \lambda_3(x^3 - \frac{3}{5}x)$. *Wanneer we de normalisatievoorwaarde in rekening brengen, krijgen we* $\lambda_3 = 1$ *en*

$$P_3(x) = x^3 - \frac{3}{5}x\ .$$

- *Op dezelfde wijze vindt men voor de veelterm van de vierde graad*

$$\alpha_4 = 0\ ,\ \beta_4 = \frac{9}{35}\ ,\ \lambda_4 = 1 \quad en \quad P_4(x) = x^4 - \frac{6}{7}x^2 + \frac{3}{35}\ . \tag{4.16}$$

- *De berekeningen voor de Legendre-veelterm van graad vijf geven*

$$\alpha_5 = 0\ ,\ \beta_5 = \frac{16}{63}\ ,\ \lambda_5 = 1 \quad en \quad P_5(x) = x^5 - \frac{10}{9}x^3 + \frac{5}{21}x\ . \tag{4.17}$$

Dit zijn, op evenredigheidsfactoren na, de veeltermen die met de methode van Gram-Schmidt afgeleid werden in het vorige hoofdstuk, zie Voorbeeld 3.2. Dit zijn eveneens, op evenredigheidsfactoren na, de veeltermen van Legendre.

4.2.3 De nulpunten van orthogonale veeltermen

We formuleren en bewijzen enkele belangrijke eigenschappen van nulpunten van orthogonale veeltermen.

Stelling 4.2.4
De veelterm $\phi_k(x)$ *die behoort tot een stel veeltermen dat orthogonaal is over een interval* $[a, b]$, *heeft* k *enkelvoudige reële nulpunten, gelegen in het open interval* (a, b).

Bewijs Voor $k = 0$ is de stelling triviaal; het bewijs dat hier volgt, geldt voor $k > 0$.
Daar $\phi_k(x)$ een veelterm is van de k-de graad, heeft hij hoogstens k reële nulpunten. Hij kan dus hoogstens k maal van teken veranderen in (a, b), en dit enkel als alle nulpunten reëel en enkelvoudig zijn. Als we dus kunnen bewijzen dat deze veelterm k maal van teken verandert in (a, b), dan is de stelling bewezen.
Veronderstel dat $\phi_k(x)$ slechts m keer van teken verandert met $m < k$. Noem de punten waar dit gebeurt $x_1, x_2, \ldots, x_m$. Beschouw dan de veelterm

$$\psi(x) = \phi_k(x)(x - x_1)(x - x_2)\cdots(x - x_m)\ .$$

Telkens als $\phi_k(x)$ van teken wisselt, bijvoorbeeld in x_i, wordt dit opgeheven door de aanwezigheid van een factor $(x - x_i)$ die er ook van teken wisselt. De functie $\psi(x)$ verandert dus nergens van teken in (a, b). Vanwege de continuïteit is $\psi(x)$ dus overal positief of overal negatief. De integraal

$$\int_a^b w(x)\,\phi_k(x)\,(x - x_1)(x - x_2)\cdots(x - x_m)\,dx$$

is bijgevolg verschillend van nul.
Welnu, $(x - x_1)(x - x_2)\cdots(x - x_m)$ is een veelterm van graad $m < k$, en dus orthogonaal tot $\phi_k(x)$; dus moet de integraal wel nul zijn. De aanname $m < k$ is bijgevolg onjuist. □

Opmerkingen

- Uit deze stelling volgt dat orthogonale veeltermen een sterk oscillerend verloop hebben in het interval $[a, b]$, en monotoon stijgen of dalen daarbuiten.

- De randpunten van $[a, b]$ kunnen geen nulpunten van de orthogonale veelterm zijn. Hieruit volgt dat de normalisatievoorwaarde $\phi_k(a) = 1$ of $\phi_k(b) = 1$ een zinvolle voorwaarde is: door een scalering van de veelterm kan er steeds aan worden voldaan.

In heel wat numerieke toepassingen is het van belang de waarde van de nulpunten van een orthogonale veelterm nauwkeurig te kennen. Dit is bijvoorbeeld zo bij het oplossen van partiële differentiaalvergelijkingen met de *Gauss-collocatiemethode*, bij het numeriek berekenen van integralen met *Gauss-kwadratuurformules* en bij het numeriek benaderen van functies met bepaalde optimale interpolatiemethoden. De orthogonale veelterm wordt zelf volledig bepaald door de waarden van α_k, β_k en λ_k. Ook de nulpunten liggen dus in principe vast wanneer die parameters gegeven zijn. Voor het bepalen van de waarde van de nulpunten kan men de onderstaande eigenschap gebruiken.

Stelling 4.2.5
De nulpunten van de veelterm $\phi_n(x)$ zijn de eigenwaarden van de $n \times n$ tridiagonale matrix `tridiag`$(\nu_{k-1}; \alpha_k; \mu_k)$ *met $\nu_k = \beta_{k+1}$ en $\mu_k = 1/\lambda_k$, waarbij α_k, β_k en λ_k gegeven worden door (4.11) en de normalisatievoorwaarde.*

Bewijs De fundamentele recursiebetrekking kan herschreven worden als

$$\beta_k\phi_{k-2}(x) + \alpha_k\phi_{k-1}(x) + \frac{1}{\lambda_k}\phi_k(x) = x\phi_{k-1}(x)\ .$$

Stel nu $\nu_k = \beta_{k+1}$ en $\mu_k = 1/\lambda_k$, dan wordt dit

$$\nu_{k-1}\phi_{k-2}(x) + \alpha_k\phi_{k-1}(x) + \mu_k\phi_k(x) = x\phi_{k-1}(x)\ . \tag{4.18}$$

Definieer de $n \times n$ tridiagonale matrix A en de n-vector $\Phi(x)$ als

$$A = \begin{bmatrix} \alpha_1 & \mu_1 & & & & & \\ \nu_1 & \alpha_2 & \mu_2 & & & & \\ & \nu_2 & \alpha_3 & \mu_3 & & & \\ & & \nu_3 & \alpha_4 & \mu_4 & & \\ & & & \cdot & \cdot & \cdot & \\ & & & & \nu_{n-2} & \alpha_{n-1} & \mu_{n-1} \\ & & & & & \nu_{n-1} & \alpha_n \end{bmatrix} \quad \text{en} \quad \Phi(x) = \begin{bmatrix} \phi_0(x) \\ \phi_1(x) \\ \phi_2(x) \\ \phi_3(x) \\ \vdots \\ \phi_{n-2}(x) \\ \phi_{n-1}(x) \end{bmatrix} . \tag{4.19}$$

Gebruikmakend van (4.18), kan men het matrix-vectorproduct $A\Phi(x)$ vereenvoudigen tot

$$A\Phi(x) = x \begin{bmatrix} \phi_0(x) \\ \phi_1(x) \\ \phi_2(x) \\ \vdots \\ \phi_{n-2}(x) \\ \phi_{n-1}(x) \end{bmatrix} - \mu_n \begin{bmatrix} 0 \\ 0 \\ 0 \\ \vdots \\ 0 \\ \phi_n(x) \end{bmatrix} = x\,\Phi(x) - \mu_n\,\Psi(x) \; . \tag{4.20}$$

Evalueren we deze betrekking in een nulpunt x_k van de veelterm $\phi_n(x)$, dan vinden we

$$A\Phi(x_k) = x_k\,\Phi(x_k) \; . \tag{4.21}$$

Dit wil zeggen dat x_k een eigenwaarde is van A, met $\Phi(x_k)$ als eigenvector. Dit geldt voor alle nulpunten $x_1, x_2, \ldots, x_n$ van $\phi_n(x)$. Vermits een $n \times n$ matrix hoogstens n verschillende eigenwaarden heeft, moeten alle eigenwaarden nulpunten zijn (en omgekeerd). □

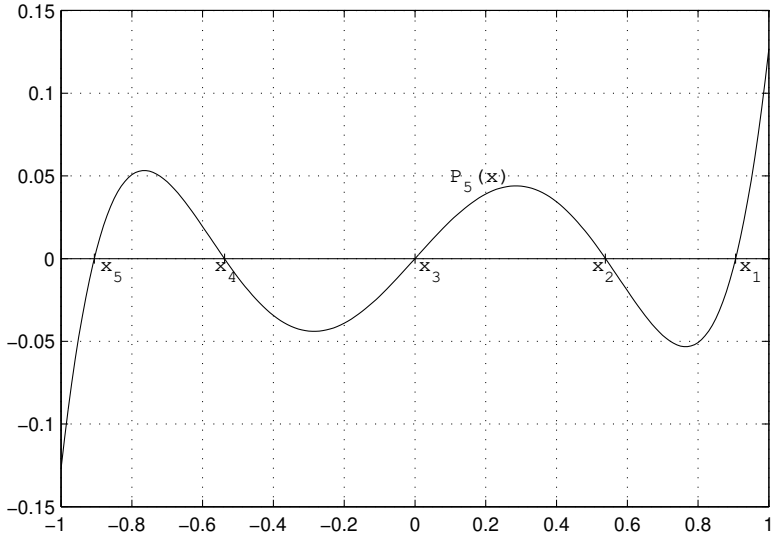
Voorbeeld 4.3 *We berekenen de nulpunten van de veelterm van graad 5, opgesteld in Voorbeeld 4.2. De waarde van de verschillende parameters worden in de onderstaande tabel opgesomd.*

k	*1*	*2*	*3*	*4*	*5*
λ_k	*1*	*1*	*1*	*1*	*1*
α_k	*0*	*0*	*0*	*0*	*0*
β_k	*0*	*1/3*	*4/15*	*9/35*	*16/63*

De matrix A en de bijhorende eigenwaarden worden gegeven door

$$A = \begin{bmatrix} 0 & 1 & & & \\ 1/3 & 0 & 1 & & \\ & 4/15 & 0 & 1 & \\ & & 9/35 & 0 & 1 \\ & & & 16/63 & 0 \end{bmatrix} \quad en \quad \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \end{bmatrix} = \begin{bmatrix} 0.90617984593866 \\ 0.53846931010568 \\ 0 \\ -0.53846931010568 \\ -0.90617984593866 \end{bmatrix} .$$

De veelterm werd getekend in Figuur 4.1. Ook de ligging van de nulpunten werd aangeduid.



Figuur 4.1: Veelterm van de vijfde graad, met nulpunten $x_1, \ldots, x_5$. Op een constante na is dit de Legendre-veelterm van graad 5.

4.3 Continue kleinste-kwadratenbenadering

We gaan nu wat dieper in op enkele aspecten van kleinste-kwadratenbenaderingen.

4.3.1 Foutenkromme

Zij een orthogonaal stel veeltermen $\phi_0(x), \ldots, \phi_n(x)$ en een continue te benaderen functie $f(x)$ gegeven. De kleinste-kwadratenbenadering van de n-de graad is dan gelijk aan

$$y_n(x) = \sum_{k=0}^{n} a_k \phi_k(x) \quad \text{met} \quad a_k = \frac{\int_a^b w(x)\, f(x)\, \phi_k(x)\, dx}{\int_a^b w(x)\, \left(\phi_k(x)\right)^2\, dx}, \tag{4.22}$$

zie (4.7). De fout of het *residu* van de kleinste-kwadratenbenadering t.o.v. de gegeven functie is

$$r_n(x) = f(x) - y_n(x). \tag{4.23}$$

Onder milde voorwaarden op de te benaderen functie convergeert de rij $\{a_k\}_{k\geq 0}$ zeer snel naar nul [Tre19]. We mogen daarom vermoeden dat men een goede schatting van de fout bekomt door de eerstvolgende term in de reeks (4.22) te beschouwen,

$$r_n(x) \simeq -a_{n+1}\phi_{n+1}(x)\ . \tag{4.24}$$

Bovenstaande formule laat verwachten dat de fout van de n-de graadsbenadering net als $\phi_{n+1}(x)$ nul zal worden in $n{+}1$ punten van het open interval (a, b). We zullen bewijzen dat dit voor continue functies inderdaad het geval is.

STELLING 4.3.6 *(interpolerende eigenschap van de kleinste-kwadratenbenadering)*
Zij f een continue functie op het interval $[a, b]$. Dan geldt dat de fout van de n-de graads-benadering nul wordt in minstens $n+1$ punten van het open interval (a, b).

Bewijs We tonen aan dat het residu minstens $n+1$ tekenwisselingen ondergaat in het open interval (a, b). Uit de continuïteit van $f(x)$ volgt dan dat het residu minstens $n+1$ nulpunten heeft.
Veronderstel dat $r_n(x)$ slechts m tekenwisselingen zou ondergaan, met $m \leq n$, namelijk in de punten $x_1 < x_2 < \ldots < x_m$, gelegen in het open interval (a, b). Dan heeft de functie

$$r_n(x)\,(x - x_1)(x - x_2)\cdots(x - x_m)$$

geen enkele tekenverandering in $[a, b]$ en is dus

$$\int_a^b w(x)\, r_n(x)\,(x - x_1)(x - x_2)\cdots(x - x_m)\,dx \neq 0\ . \tag{4.25}$$

Nu werd echter in Hoofdstuk 2 aangetoond dat het residu van een beste benadering in een deelruimte orthogonaal staat tot de benaderingsruimte, in dit geval tot alle veeltermen van graad n of lager. De functie $r_n(x)$ staat dus ook orthogonaal tot de veelterm

$$V_j(x) = (x - x_1)(x - x_2)\cdots(x - x_m)\ .$$

Dit is in tegenstrijd met (4.25). De veronderstelling $m \leq n$ is dus verkeerd. □

Voorbeeld 4.4 *We zoeken de kleinste-kwadratenbenadering y_4 van graad 4 van de functie $f(x) = e^x$ op het interval $[-1,\ 1]$. Voor de gewichtsfunctie $w(x) \equiv 1$ kan deze benadering berekend worden via formule (4.22), met $n = 4$ en $\phi_k(x) = P_k(x),\ k = 0, \ldots, 4$. In de monomiale basis is dit de veelterm*

$$y_4(x) = 1.000031 + 0.997955\,x + 0.499352\,x^2 + 0.176139\,x^3 + 0.043597\,x^4. \tag{4.26}$$

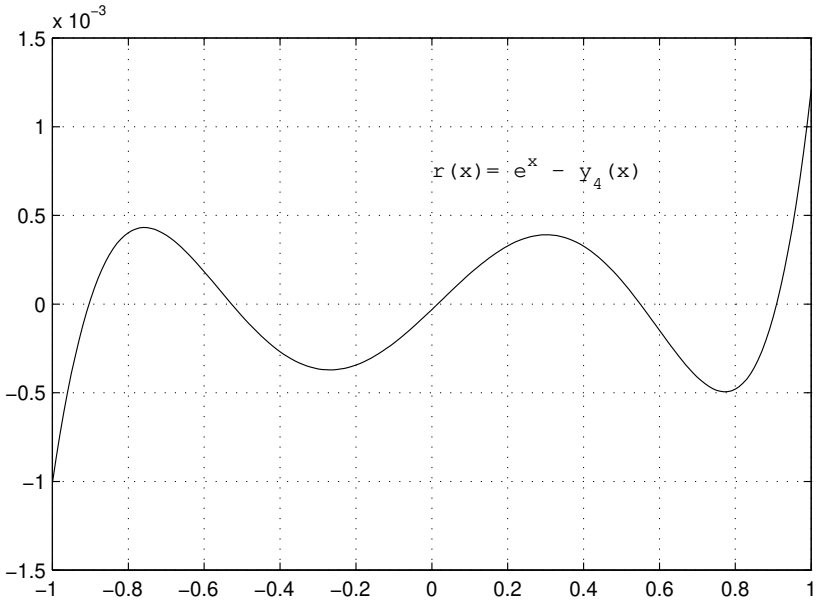
In Figuur 4.2 wordt de foutenkromme getoond. Overeenkomstig Stelling 4.3.6 is de fout nul in vijf punten, wat betekent dat de functie f daar geïnterpoleerd wordt.
De fout is groter aan de uiteinden van het interval. We kunnen in het benaderingscriterium meer gewicht geven aan de uiteinden van het interval door de gewichtsfunctie

$$w(x) = \frac{1}{\sqrt{1 - x^2}} \tag{4.27}$$

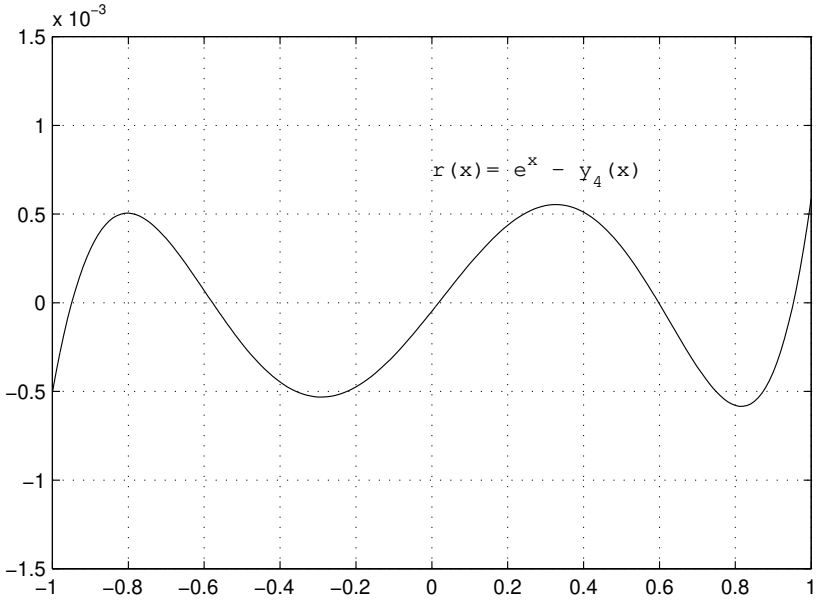
te beschouwen. De foutencurve horende bij deze gewichtsfunctie wordt getoond in Figuur 4.3.

Een orthogonale rij van veeltermen op $[-1,\ 1]$ met betrekking tot gewichtsfunctie (4.27) zullen we later identificeren als de Chebyshev-veeltermen.

Opmerking 4.2 *Als in de opgave van Voorbeeld 4.4 het beschouwde interval aangepast wordt, dan is het aangewezen om een transformatie van de variabele x uit te voeren die het beschouwde interval afbeeldt op $[-1,\ 1]$.*



Figuur 4.2: Fout van de kleinste-kwadratenbenadering van graad 4 voor e^x met $w(x) \equiv 1$.



Figuur 4.3: Fout van de kleinste-kwadr.-benadering van graad 4 voor e^x met $w(x) = \frac{1}{\sqrt{1-x^2}}$.

4.3.2 Het evalueren van kleinste-kwadratenbenaderingen

De meest voor de hand liggende manier voor het evalueren van een kleinste-kwadratenbenadering van de vorm (4.22) voor waarde(n) van x bestaat erin de benadering te rangschikken naar opeenvolgende machten van x, d.w.z. de coëfficiënten c_k te bepalen zodat

$$y_n(x) = \sum_{k=0}^{n} c_k x^k, \tag{4.28}$$

om daarna het rekenschema van Horner toe te passen. Een alternatieve manier bestaat erin uitdrukking (4.22) direct te evalueren, zonder omzetting naar de vorm (4.28). Hiertoe moet de kennis van de coëfficiënten a_k en de parameters λ_k, α_k en β_k van de fundamentele recursiebetrekking (4.10) voldoende zijn.
We zullen aantonen dat $y_n(x)$ kan worden berekend door volgend recursieschema:

$$\left\{\begin{array}{lcl} b_{n+2} & = & b_{n+1} = 0 \\ b_k & = & a_k + \lambda_{k+1}(x-\alpha_{k+1})b_{k+1} - \lambda_{k+2}\beta_{k+2}b_{k+2}\ , \quad k = n, n-1, \ldots, 0 \\ y_n(x) & = & b_0\lambda_0 \end{array}\right. \tag{4.29}$$

Dit schema noemt men het *rekenschema van Smith*. De correctheid bewijzen we als volgt. De recursiebetrekking (4.29) is ook te schrijven als

$$\left\{\begin{array}{lcl} a_n & = & b_n \\ a_{n-1} & = & b_{n-1} - \lambda_n(x-\alpha_n)\, b_n \\ a_k & = & b_k - \lambda_{k+1}(x-\alpha_{k+1})b_{k+1} + \lambda_{k+2}\beta_{k+2}b_{k+2}\ , \quad k = n-2, n-3, \ldots, 0 \end{array}\right.$$

Links en rechts vermenigvuldigen met een orthogonale veelterm levert volgende tabel:

$$\begin{array}{rcl} a_n\phi_n(x) & = & b_n\phi_n(x) \\ a_{n-1}\phi_{n-1}(x) & = & b_{n-1}\phi_{n-1}(x) - \lambda_n(x-\alpha_n)b_n\phi_{n-1}(x) \\ a_{n-2}\phi_{n-2}(x) & = & b_{n-2}\phi_{n-2}(x) - \lambda_{n-1}(x-\alpha_{n-1})b_{n-1}\phi_{n-2}(x) + \lambda_n\beta_n b_n\phi_{n-2}(x) \\ & \vdots & \\ a_2\phi_2(x) & = & b_2\phi_2(x) - \lambda_3(x-\alpha_3)b_3\phi_2(x) + \lambda_4\beta_4 b_4\phi_2(x) \\ a_1\phi_1(x) & = & b_1\phi_1(x) - \lambda_2(x-\alpha_2)b_2\phi_1(x) + \lambda_3\beta_3 b_3\phi_1(x) \\ a_0\phi_0(x) & = & b_0\phi_0(x) - \lambda_1(x-\alpha_1)b_1\phi_0(x) + \lambda_2\beta_2 b_2\phi_0(x) \\ - - - - - & & - \\ y_n(x) & = & b_0\phi_0(x) \\ & & +b_1[\phi_1(x) - \lambda_1(x-\alpha_1)\phi_0(x)] \\ & & +b_2[\phi_2(x) - \lambda_2(x-\alpha_2)\phi_1(x) + \lambda_2\beta_2\phi_0(x)] \\ & & \vdots \\ & & +b_n[\phi_n(x) - \lambda_n(x-\alpha_n)\phi_{n-1}(x) + \lambda_n\beta_n\phi_{n-2}(x)] \end{array}$$

De veeltermen $\phi_0(x)$, $\phi_1(x)$, $\phi_2(x)$, ... voldoen aan de recursiebetrekking (4.10). Hierdoor zijn de coëfficiënten van b_1, b_2, ..., b_n alle nul. Er blijft dus over

$$y_n(x) = b_0\phi_0(x) = b_0\lambda_0\ .$$

Een laatste manier, die gerelateerd is aan het rekenschema van Smith, bestaat uit het eerst uitrekenen van $\phi_0(x),\ \phi_1(x),\ \ldots,\ \phi_n(x)$ voor de beschouwde waarde(n) van x door middel van de recursiebetrekking (4.10). Vervolgens kan dan (4.28) rechtstreeks geëvalueerd worden.

4.3.3 Een benadering van de kleinste-kwadratenbenadering

In §4.3.1 behandelden we de interpolerende eigenschap van kleinste-kwadratenbenaderingen. We toonden aan dat de benadering van graad n de functie $f(x)$ interpoleert in minstens $n+1$ punten. Uit intuïtieve beschouwingen weten we dat die interpolatiepunten in de omgeving van de nulpunten van $\phi_{n+1}(x)$ zullen gelegen zijn. Voor de residufunctie vonden we immers (4.24) als benadering. We kunnen ons nu de vraag stellen wat er gebeurt als we exact interpoleren door de nulpunten van $\phi_{n+1}(x)$. We zullen dan wellicht een benadering krijgen die niet te sterk afwijkt van de echte kleinste-kwadratenbenadering.

Voorbeeld 4.5 *De nulpunten van de Legendre-veelterm van graad vijf werden berekend in §4.2.3. De veelterm van graad vier die de functie e^x in deze nulpunten interpoleert, is gelijk aan*

$$y_4(x) = 1.000000 + 0.997963x + 0.499663x^2 + 0.176126x^3 + 0.043235x^4. \tag{4.30}$$

Deze veelterm is een goede benadering voor de echte vierdegraads kleinste-kwadratenbenadering (4.26).

Ter illustratie stelden we de interpolerende veeltermen op van verschillende graden n voor de functie e^x en voor de iets moeilijker te benaderen functie $1/(1+6x^2)$. We namen telkens twee stellen interpolatiepunten: een equidistant stel en het stel overeenkomend met de nulpunten van de Legendre-veelterm van graad $n+1$. In de Figuren 4.4 en 4.5 werd de norm van het residu uitgezet als functie van de graad van de interpolerende veelterm. De puntlijn stemt overeen met equidistante interpolatie, de volle lijn met interpolatie in de Legendre-punten. Voor beide benaderingsproblemen blijken duidelijk de voordelen van de niet-equidistante interpolatie. De functie $1/(1+6x^2)$ is een klassiek voorbeeld van een functie die zeer moeilijk te benaderen valt met een interpolerende veelterm door equidistante punten. Zoals blijkt uit Figuur 4.5 neemt de fout zelfs toe bij toenemende graad.

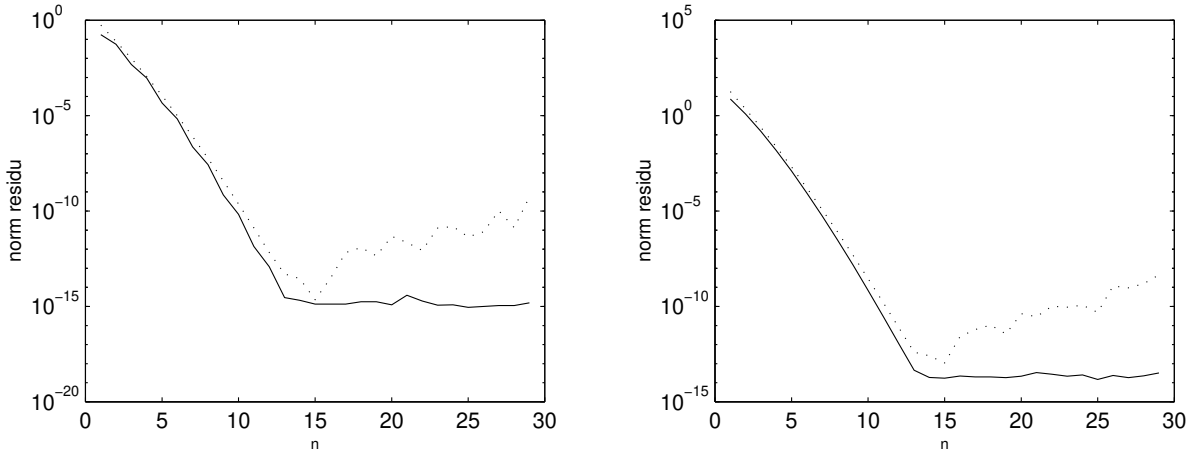
4.4 Klassieke orthogonale veeltermen

4.4.1 Inleiding

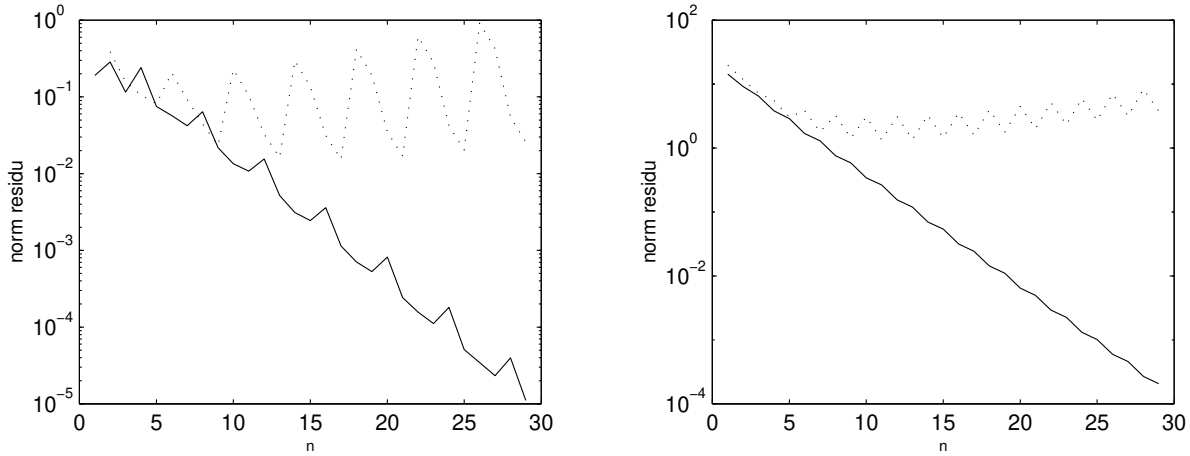
Er bestaat een uitgebreide literatuur over de orthogonale veeltermen gegenereerd door

$$(p, q) = \int_a^b w(x)\, p(x)\, q(x)\, dx \tag{4.31}$$

voor enkele welbepaalde keuzes van $[a, b]$ en $w(x)$. Zo kent men



Figuur 4.4: Maximumnorm (links) en 2-norm (rechts) van het residu van een interpolerende benadering van graad n voor de functie e^x over het interval $[-1,1]$: equidistante interpolatiepunten (puntlijn) en Legendre-interpolatiepunten (volle lijn).



Figuur 4.5: Cfr. Figuur 4.4, nu voor de functie $1/(1+6x^2)$

(i) de *Jacobi*-veeltermen $P_k^{(\alpha,\beta)}(x)$: $-a=b=1$ en $w(x)=(1-x)^\alpha(1+x)^\beta$, met $\alpha,\beta>-1$

(ii) de *Laguerre*-veeltermen $L_k^{(\alpha)}(x)$: $a=0$, $b=\infty$ en $w(x)=x^\alpha e^{-x}$, met $\alpha>-1$

(iii) de *Hermite*-veeltermen $H_k(x)$: $-a=b=\infty$ en $w(x)=e^{-x^2}$

Bij de Jacobi-veeltermen maakt men nog een verder onderscheid afhankelijk van de waarde van de parameters α en β:

- $\alpha=\beta$: *Gegenbauer*-veeltermen of *ultrasferische* veeltermen $P_k^{(\alpha)}(x)$

- $\alpha = 0$, $\beta = 0$: *Legendre*-veeltermen $P_k(x)$
- $\alpha = -1/2$, $\beta = -1/2$: *Chebyshev*-veeltermen van de eerste soort $T_k(x)$
- $\alpha = 1/2$, $\beta = 1/2$: *Chebyshev*-veeltermen van de tweede soort $U_k(x)$

We zullen een aantal eigenschappen van deze klassieke orthogonale veeltermen bestuderen.

4.4.2 Legendre-veeltermen

4.4.2.1 Definitie, constructie en eigenschappen

DEFINITIE 4.4.1 *(Legendre-veeltermen)*
De veeltermen $P_0(x), P_1(x), P_2(x), \ldots$ met $P_k(1) = 1$ die een orthogonale rij vormen voor het scalair product (4.31) t.o.v. de gewichtsfunctie $w(x) \equiv 1$ in het interval $[-1, 1]$, noemt men de veeltermen van Legendre.

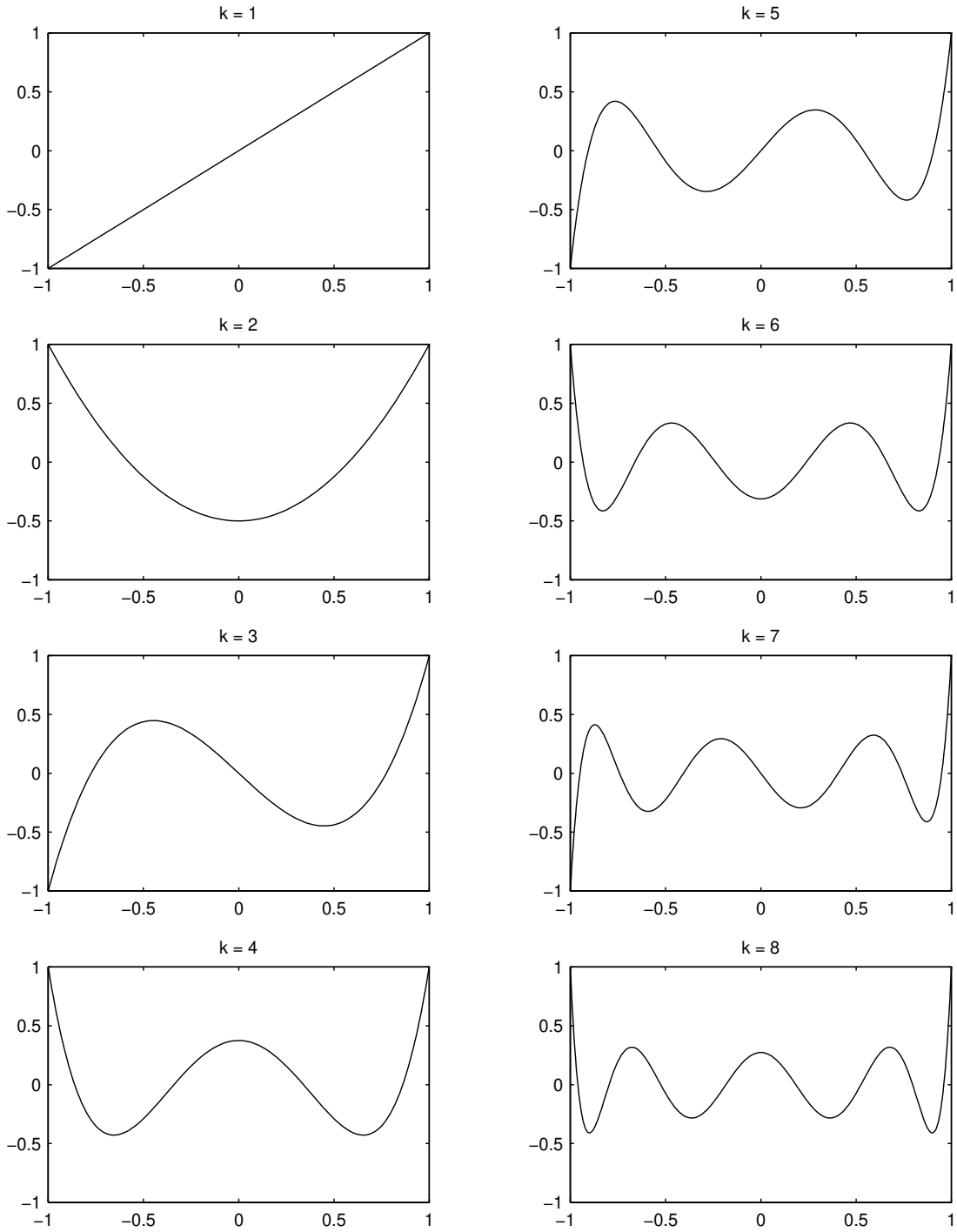
De eenvoudigste manier om de veeltermen te berekenen, is gebruik te maken van de zogenaamde differentiaalvergelijking van Rodrigues. Dit levert,

$$P_k(x) = \frac{1}{2^k k!} \cdot \frac{d^k}{dx^k}(x^2 - 1)^k \ . \tag{4.32}$$

Uitwerking van deze formule geeft onder meer:

$$\begin{aligned}
P_0(x) &= 1 \\
P_1(x) &= x \\
P_2(x) &= \frac{1}{2}(3x^2 - 1) \\
P_3(x) &= \frac{1}{2}(5x^3 - 3x) \\
P_4(x) &= \frac{1}{8}(35x^4 - 30x^2 + 3) \\
P_5(x) &= \frac{1}{8}(63x^5 - 70x^3 + 15x) \\
P_6(x) &= \frac{1}{16}(231x^6 - 315x^4 + 105x^2 - 5) \\
P_7(x) &= \frac{1}{16}(429x^7 - 693x^5 + 315x^3 - 35x) \\
P_8(x) &= \frac{1}{128}(6435x^8 - 12012x^6 + 6930x^4 - 1260x^2 + 35)
\end{aligned}$$

We merken op dat de veeltermen van even graad enkel even machten bevatten, en deze van oneven graad enkel oneven machten. De Legendre-veeltermen van graad 1 tot en met graad 8 werden getekend in Figuur 4.6.



Figuur 4.6: Legendre-veeltermen van graad 1 tot en met graad 8.

Mits enig rekenwerk kan men de volgende algemene uitdrukking afleiden:

$$P_k(x) = \frac{1}{2^k} \sum_{j=0}^{\lfloor k/2 \rfloor} (-1)^j \left(\begin{array}{c} 2k-2j \\ k \end{array} \right) \left(\begin{array}{c} k \\ j \end{array} \right) x^{k-2j} \ . \tag{4.33}$$

Hierin staat $\lfloor r \rfloor$ voor het grootste geheel getal kleiner dan of gelijk aan r.
De hoogstegraadscoëfficiënt A_k wordt gegeven door

$$A_k = \frac{(2k)!}{2^k \, (k!)^2} \tag{4.34}$$

en de (natuurlijke) norm door

$$\| \ P_k \ \| = \sqrt{\frac{2}{2k+1}} \ . \tag{4.35}$$

Tenslotte kan men bewijzen dat

$$|P_k(x)| \leq 1 \quad \text{voor alle} \quad x \in [-1, 1] \ . \tag{4.36}$$

We vermelden nog de klassieke recursiebetrekking voor de Legendre-veeltermen,

$$P_k(x) = \frac{2k-1}{k} x P_{k-1}(x) - \frac{k-1}{k} P_{k-2}(x), \tag{4.37}$$

die volgt uit Stelling 4.2.3.

4.4.2.2 Het opstellen van Legendre-benaderingen

De Legendre-benadering over het interval $[-1, 1]$ van een functie $f(x)$ wordt gegeven door

$$y_n(x) = \sum_{k=0}^{n} a_k P_k(x) \ , \quad \text{met} \quad a_k = \frac{2k+1}{2} \int_{-1}^{1} f(x) P_k(x) dx \ . \tag{4.38}$$

De integralen die in (4.38) voorkomen kunnen berekend worden met methodes uit de numeriek integratie, waarbij voor de evaluatie van P_k in de knooppunten van de gehanteerde kwadratuurformules recursiebetrekking (4.37) gebruikt kan worden. Een alternatief bestaat erin om $P_k(x)$ expliciet in machten van x te schrijven en de $k+1$ termen afzonderlijk te berekenen. Men zal dus eerst de momenten

$$I_k = \int_{-1}^{1} x^k \, f(x) \, dx$$

van $f(x)$ berekenen voor $k = 0, 1, \ldots, n$. Men kan dan deze waarden gebruiken voor het bepalen van de coëfficiënten a_k. Laten we de a_k-waarden uitdrukken in functie van de I_k:

$$\begin{aligned}
a_k &= \frac{2k+1}{2} \int_{-1}^{1} f(x)\, P_k(x)\, dx \\
&= \frac{2k+1}{2} \int_{-1}^{1} f(x)\, \frac{1}{2^k} \sum_{j=0}^{\lfloor k/2 \rfloor} (-1)^j \left(\begin{array}{c} 2k-2j \\ k \end{array} \right) \left(\begin{array}{c} k \\ j \end{array} \right) x^{k-2j}\, dx \\
&= \frac{2k+1}{2^{k+1}} \sum_{j=0}^{\lfloor k/2 \rfloor} (-1)^j \left(\begin{array}{c} 2k-2j \\ k \end{array} \right) \left(\begin{array}{c} k \\ j \end{array} \right) \int_{-1}^{1} f(x)\, x^{k-2j}\, dx \\
&= \frac{2k+1}{2^{k+1}} \sum_{j=0}^{\lfloor k/2 \rfloor} (-1)^j \left(\begin{array}{c} 2k-2j \\ k \end{array} \right) \left(\begin{array}{c} k \\ j \end{array} \right) I_{k-2j} \qquad (4.39)
\end{aligned}$$

Ter illustratie werken we enkele van deze uitdrukkingen verder uit:

$$\begin{aligned}
a_0 &= \frac{1}{2} I_0 \\
a_1 &= \frac{3}{2} I_1 \\
a_2 &= \frac{5}{4}(3I_2 - I_0) \\
a_3 &= \frac{7}{4}(5I_3 - 3I_1) \\
a_4 &= \frac{9}{16}(35I_4 - 30I_2 + 3I_0) \\
a_5 &= \frac{11}{16}(63I_5 - 70I_3 + 15I_1) \\
a_6 &= \frac{13}{32}(231I_6 - 315I_4 + 105I_2 - 5I_0) \\
a_7 &= \frac{15}{32}(429I_7 - 693I_5 + 315I_3 - 35I_1) \\
a_8 &= \frac{17}{256}(6435I_8 - 12012I_6 + 6930I_4 - 1260I_2 + 35I_0)
\end{aligned}$$

Voorbeeld 4.6 *Hernemen we het voorbeeld van de functie $f(x) = e^x$ in $[-1, 1]$. Hiervoor is*

$$I_k = \int_{-1}^{1} x^k e^x dx \ .$$

Dit geeft onder meer

$$\begin{aligned} I_0 &= \int_{-1}^{1} e^x dx = e - e^{-1} \\ I_1 &= \int_{-1}^{1} x e^x dx = [x e^x]_{-1}^{1} - \int_{-1}^{1} e^x dx = e + e^{-1} - (e - e^{-1}) = 2e^{-1} \\ I_k &= \int_{-1}^{1} x^k e^x dx = [x^k e^x]_{-1}^{1} - \int_{-1}^{1} k x^{k-1} e^x dx = e - (-1)^k e^{-1} - k I_{k-1} \ . \end{aligned}$$

In dit voorbeeld kunnen alle momenten worden uitgedrukt als lineaire combinaties van de getallen e en e^{-1}. We zullen verder alle berekeningen doorvoeren in een uitgebreide precisie, met 25 beduidende decimale cijfers. Voor de twee constanten nemen we:

$$\begin{aligned} e &= 2.718281828459045235360287 \\ e^{-1} &= 0.367879441171442321595523 7 \end{aligned}$$

Voor de coëfficiënten a_0 tot a_{10} vinden we:

$$\begin{aligned} a_0 &= 1.1752011936438014568823 8 \\ a_1 &= 1.1036383235143269647865 71 \\ a_2 &= 0.3578143506473724604790 5 \\ a_3 &= 0.0704556336684890278 15 \\ a_4 &= 0.0099651281488691785 2 \\ a_5 &= 0.0010995861272075085 \\ a_6 &= 0.0000994543391134250 \\ a_7 &= 0.0000762054130886 \\ a_8 &= 0.0000050647197 45 \end{aligned}$$

Hierbij hebben we enkel de juiste cijfers gegeven. Zoals men ziet, gaan er bij het berekenen van de coëfficiënten heel wat beduidende cijfers verloren. Deze fouten zijn te wijten aan het feit dat in de formules expliciet of impliciet grote coëfficiënten voorkomen. Neem bijvoorbeeld a_{10}. Deze coëfficiënt is een lineaire combinatie van e en e^{-1}, een lineaire combinatie die na enig rekenwerk aanleiding geeft tot volgend verschil

$$a_{10} = \frac{21}{2}(551439881.732110137222799 - 551439881.732110137074141) \ .$$

De twee getallen van dit verschil hebben 18 cijfers gemeen. Het resultaat van de aftrekking heeft hoogstens 7 beduidende cijfers nauwkeurig. Voor de benadering van de vierde graad vindt men

$$\begin{aligned} y_4(x) &= 1.1752012 P_0(x) + 1.1036383 P_1(x) + 0.3578144 P_2(x) \\ &\quad + 0.0704556 P_3(x) + 0.0099651 P_4(x) \ . \end{aligned}$$

Als we dit rangschikken naar machten van x, dan bekomen we:

$$y_4(x) = 1.000031 + 0.997955x + 0.499352x^2 + 0.176139x^3 + 0.043597x^4 \ .$$

Dit is de benadering die we reeds vroeger opstellen in (4.26).

4.4.3 Chebyshev-veeltermen

4.4.3.1 Definitie en constructie van de orthogonale rij

DEFINITIE 4.4.2 *(Chebyshev-veeltermen)*
De veeltermen $T_0(x), T_1(x), T_2(x), \ldots$ *met* $T_k(1) = 1$ *die een orthogonale rij vormen voor het scalair product (4.31) t.o.v. de gewichtsfunctie* $w(x) = 1/\sqrt{1-x^2}$ *in het interval* $[-1, 1]$*, noemt men de* Chebyshev-veeltermen (van de eerste soort)*.*

Deze veeltermen worden gebruikt voor het opstellen van benaderingen die de integraal

$$\int_{-1}^{1} \frac{1}{\sqrt{1-x^2}} \left(f(x) - y_n(x)\right)^2 dx \tag{4.40}$$

zo klein mogelijk maken. De gewichtsfunctie is gelijk aan 1 in het midden van het interval en stijgt dan tot ∞ aan de twee uiteinden van het interval.
We zullen verifiëren dat de gezochte veeltermen worden gegeven door:

$$T_k(x) = \cos(k \arccos(x)) \ , \quad \text{voor} \quad x \in [-1, 1] \ . \tag{4.41}$$

Eerst gaan we de orthogonaliteit na. Hiertoe berekenen we het scalair product

$$(T_k, T_j) = \int_{-1}^{1} \frac{T_k(x) T_j(x)}{\sqrt{1-x^2}} \, dx = \int_{-1}^{1} \frac{\cos(k \arccos(x)) \cos(j \arccos(x))}{\sqrt{1-x^2}} \, dx \ .$$

Wanneer men overgaat op een nieuwe veranderlijke volgens $x = \cos(\theta)$, dan wordt bovenstaande uitdrukking

$$(T_k, T_j) = \int_{0}^{\pi} \cos(k\theta) \cos(j\theta) \, d\theta \ .$$

Gebruikmakend van de goniometrische identiteit

$$\cos(k\theta) \cos(j\theta) = \frac{1}{2} \left(\cos((k+j)\theta) + \cos((k-j)\theta) \right) \ ,$$

volgt hieruit onmiddellijk dat

$$(T_k, T_j) = \begin{cases} 0 & \text{als} \quad j \neq k \\ \frac{\pi}{2} & \text{als} \quad j = k \neq 0 \\ \pi & \text{als} \quad j = k = 0 \end{cases} \tag{4.42}$$

Dat betekent dat de functies $T_k(x)$ en $T_j(x)$ orthogonaal zijn voor alle $j \neq k$. De functies voldoen ook aan de normalisatievoorwaarde $T_k(1) = 1$.
We moeten dus enkel nog aantonen dat die functies veeltermen zijn. Welnu,

$$T_0(x) = \cos(0) = 1 \tag{4.43}$$
$$T_1(x) = \cos(\arccos(x)) = x \tag{4.44}$$
$$T_2(x) = \cos(2 \arccos(x)) = 2 \cos^2(\arccos(x)) - 1 = 2x^2 - 1 \tag{4.45}$$

Dat zijn inderdaad veeltermen in de veranderlijke x. Om dit meer in het algemeen te bewijzen, toont men aan dat de functies $T_k(x)$ voldoen aan de drietermsrecursiebetrekking

$$T_{k+1}(x) = 2\,x\,T_k(x) - T_{k-1}(x) \quad k > 0\ . \tag{4.46}$$

Deze gelijkheid volgt onmiddellijk uit de identiteit

$$\cos((k+1)\theta) + \cos((k-1)\theta) = 2\cos(\theta)\cos(k\theta).$$

Uit (4.46) en het voorschrift voor de start-'waarden' van deze recursiebetrekking, (4.43) en (4.44), blijkt dat dus alle $T_k(x)$ inderdaad veeltermen zijn. Uit het enig zijn (op een factor na) van de rij orthogonale veeltermen voor een gegeven interval en gewichtsfunctie mogen we besluiten dat die veeltermen inderdaad de gezochte Chebyshev-veeltermen zijn.

Gebruikmakend van de recursiebetrekking vindt men gemakkelijk volgende uitdrukkingen:

$$\begin{aligned}
T_0(x) &= 1\\
T_1(x) &= x\\
T_2(x) &= 2x^2 - 1\\
T_3(x) &= 4x^3 - 3x\\
T_4(x) &= 8x^4 - 8x^2 + 1\\
T_5(x) &= 16x^5 - 20x^3 + 5x\\
T_6(x) &= 32x^6 - 48x^4 + 18x^2 - 1\\
T_7(x) &= 64x^7 - 112x^5 + 56x^3 - 7x\\
T_8(x) &= 128x^8 - 256x^6 + 160x^4 - 32x^2 + 1
\end{aligned}$$

De Chebyshev-veeltermen van graad 1 tot en met graad 8 werden getekend in Figuur 4.7. De algemene uitdrukking is als volgt

$$T_k(x) = \frac{k}{2}\sum_{j=0}^{\lfloor k/2\rfloor}(-1)^j \frac{(k-j-1)!}{j!\,(k-2j)!}\,(2x)^{k-2j}\ . \tag{4.47}$$

De hoogstegraadscoëfficiënt wordt gegeven door

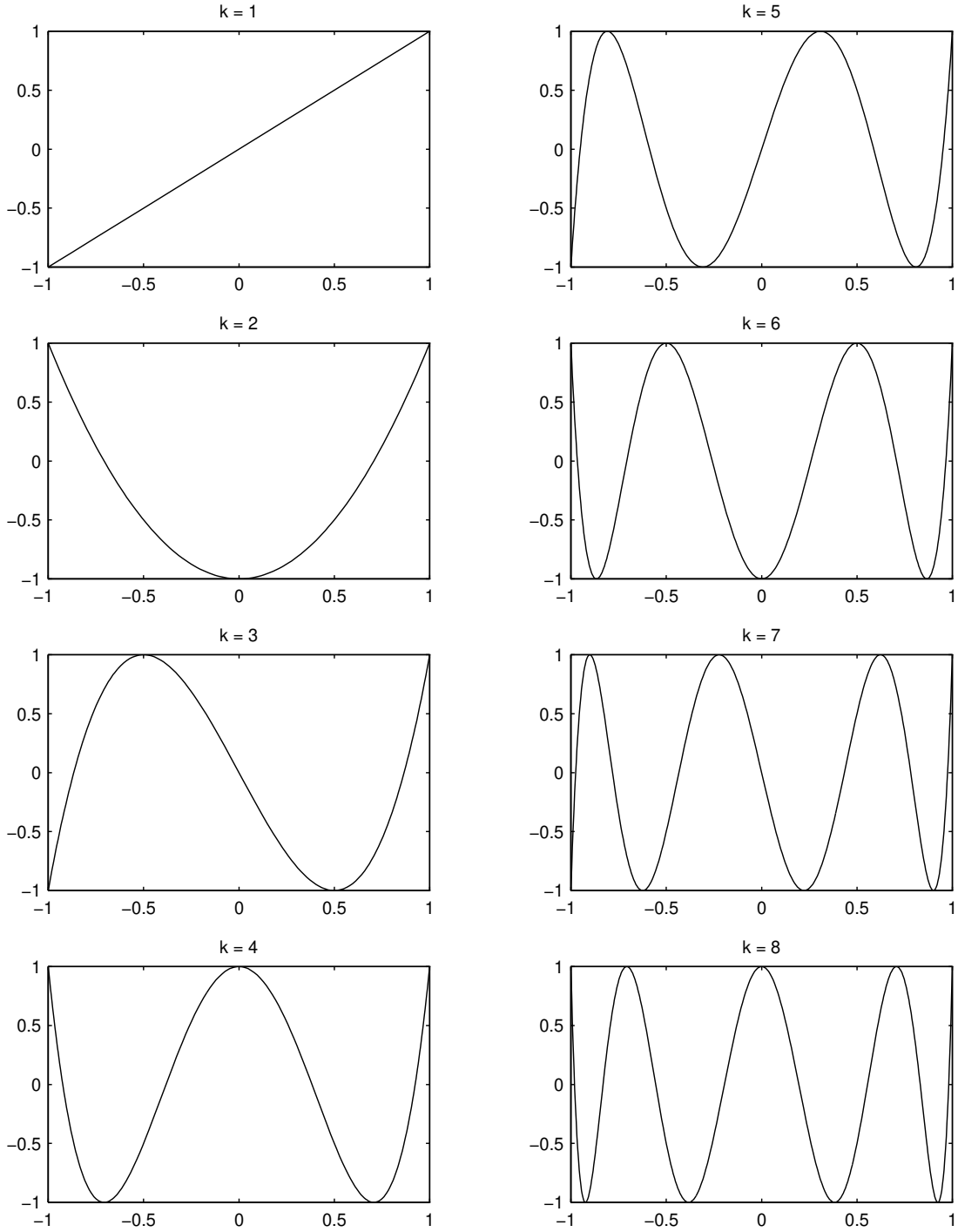
$$A_k = 2^{k-1} \quad \text{voor} \quad k \geq 1\ . \tag{4.48}$$

De (natuurlijke) norm is gelijk aan

$$\|T_k\| = \begin{cases} \sqrt{\pi/2} & \text{voor} \quad k \neq 0 \\ \sqrt{\pi} & \text{voor} \quad k = 0 \end{cases} \tag{4.49}$$

Stellen we $x = \cos(\theta)$, dan is

$$T_k(x)\,T_j(x) = \cos(k\theta)\,\cos(j\theta) = \frac{1}{2}\left(\cos((k+j)\theta) + \cos((k-j)\theta)\right)\ .$$



Figuur 4.7: Chebyshev-veeltermen van graad 1 tot en met graad 8.

Hieruit leiden we een eenvoudige formule af voor het product van twee Chebyshev-veeltermen

$$T_k(x)\,T_j(x) \;=\; \frac{1}{2}(T_{k+j}(x) \;+\; T_{|\,k-j\,|}(x))\;. \tag{4.50}$$

Voor de onbepaalde integraal van $T_k(x)$ met $k \geq 2$ vinden we

$$\begin{aligned}\int T_k(x)\,dx \;&=\; -\int \cos(k\theta)\,\sin(\theta)\,d\theta \\ &=\; -\frac{1}{2}\int (\sin((k+1)\theta) - \sin((k-1)\theta))\;d\theta \\ &=\; \frac{1}{2}\left(\frac{T_{k+1}(x)}{k+1} - \frac{T_{k-1}(x)}{k-1}\right)\end{aligned} \tag{4.51}$$

4.4.3.2 Het opstellen van Chebyshev-benaderingen

De Chebyshev-kleinste-kwadratenveelterm van graad n voor een functie $f(x)$ is gelijk aan

$$y_n(x) = \sum_{k=0}^{n} a_k T_k(x) \;\;\text{met}\;\; a_k = \frac{1}{\|T_k\|^2}\int_{-1}^{1} \frac{f(x)\,T_k(x)}{\sqrt{1-x^2}}\,dx\;. \tag{4.52}$$

Gebruikmakend van (4.49) vindt men voor de coëfficiënten de volgende uitdrukkingen:

$$a_0 = \frac{1}{\pi}\int_{-1}^{1} \frac{f(x)\,T_0(x)}{\sqrt{1-x^2}}\,dx \quad\text{en}\quad a_k = \frac{2}{\pi}\int_{-1}^{1} \frac{f(x)\,T_k(x)}{\sqrt{1-x^2}}\,dx \;\;\text{voor}\;\; k>0\;. \tag{4.53}$$

Vaak schrijft men de Chebyshev-reeks met een licht gewijzigd sommatiesymbool,

$$y_n(x) = {\sum_{k=0}^{n}}' a_k\,T_k(x) \;:=\; \frac{a_0}{2}T_0(x) + a_1T_1(x) + a_2T_2(x) + \ldots\;. \tag{4.54}$$

Men kan dan in het algemeen schrijven dat

$$a_k = \frac{2}{\pi}\int_{-1}^{1} \frac{f(x)\,T_k(x)}{\sqrt{1-x^2}}dx\;. \tag{4.55}$$

In tegenstelling tot (4.53) is deze formule nu geldig voor alle k-waarden, ook $k=0$. Met de transformatie $x=\cos(\theta)$, kan bovenstaande uitdrukking voor a_k worden omgezet tot de vorm

$$a_k = \frac{2}{\pi}\int_{0}^{\pi} f(\cos(\theta))\,\cos(k\theta)\,d\theta. \tag{4.56}$$

Doordat de singulariteiten in de eindpunten $x=\pm 1$ verwijderd zijn, leent deze vorm zich beter tot numerieke integratie.

4.4.3.3 Kleinste-kwadraten Chebyshev vs. minimax-benaderingen

De vereiste nauwkeurigheid van een veeltermbenadering wordt in de praktijk vaak niet rechtstreeks gegeven in de vorm van een toegelaten kwadratische fout (4.1), maar in de vorm van een toegelaten *maximale fout*. Men wil zeker zijn dat de benadering in elk punt van het interval een fout heeft die kleiner is dan een toegelaten afwijking. Het criterium dat aangewezen is om dit vraagstuk op te lossen, is het zogenaamde *criterium van de minimale maximale afwijking*, ook het *minimaxcriterium* genoemd. We zoeken hierbij een veelterm y_n van opgegeven graad n waarvoor

$$E_n = \max_{x\in[a,b]} |f(x) - y_n(x)| \tag{4.57}$$

minimaal is. Als de functie f continue is, bestaat zulke veelterm.

Stelling 4.4.7 *(Borel)*
Voor elke functie $f(x)$ die continu is over het compacte interval $[a, b]$, bestaat er een veelterm $y_n(x)$ van graad n of lager waarvoor geldt dat

$$\max_{x\in[a,b]} |\, f(x) - y_n(x)\,| \leq \max_{x\in[a,b]} |\, f(x) - p_n(x)\,| \quad \textit{voor alle} \quad p_n \in P_n[a,b]\ . \tag{4.58}$$

De punten in het interval $[a,\, b]$ waar het maximum E_n in (4.57) wordt bereikt, worden *extremaalpunten* genoemd. Een punt waar $f(x) - y_n(x) = E_n$ noemt men een $+$punt, waar $f(x) - y_n(x) = -E_n$ een $-$punt. De volgende stelling geeft een karakterisering van de beste benadering.

Stelling 4.4.8 *(equioscillatiestelling)*
Zij $f(x)$ een continue functie op een compact interval $[a, b]$ en zij $y_n(x)$ een veelterm van graad n. Dan is y_n een beste benadering van graad n volgens het minimaxcriterium als en alleen als er in het interval $[a, b]$ een rij van $n + 2$ extremaalpunten bestaat die afwisselen tussen $+$punten en $-$punten

Hoewel er algoritmen zijn om minimaxbenaderingen te berekenen, gebaseerd op Stelling 4.4.8, worden deze in de praktijk maar weinig gebruikt. De reden wordt in wat volgt uiteengezet.
We vermeldden reeds in §4.3.1 dat de benaderingsfout of residu $r_n(x) = f(x) - y_n(x)$ van een kleinste-kwadratenbenadering kan worden geschat door de volgende term in de reeks (4.22). Voor de Chebyshev-benadering (4.52) wordt dit

$$r_n(x) \simeq -a_{n+1}T_{n+1}(x)\ .$$

De Chebyshev veelterm T_{n+1}, *beperkt* tot $[-1,\ 1]$, heeft precies $n+2$ extrema die afwisselen tussen 1 en -1. Vaak is de benadering van $r_n(x)$ door $-a_{n+1}T_{n+1}(x)$ zeer nauwkeurig. Dit impliceert het volgende.

- De kleinste-kwadraten Chebyshev-benadering wijkt maar weinig af van de minimax-benadering. Dit wordt geïllustreerd in Figuur 4.3, waar het residu bijna aan de voorwaarden in de equioscillatiestelling voldoet.

- De veelterm van graad n die interpoleert in de $n+1$ nulpunten van $T_{n+1}(x)$ wijkt maar weinig af van de minimaxbenadering.

Deze laatste observatie en de eigenschap dat er snelle FFT[1]-gebaseerde algoritmen bestaan om te schakelen tussen functiewaarden in Chebyshev punten (dit zijn de nulpunten van een Chebyshev-veelterm) en de coëfficiënten van de interpolerende veelterm uitgedrukt in een Chebyshev basis, liggen aan de basis van het veelvuldig gebruik van interpolatie in Chebyshev punten in numerieke algoritmen en software voor het berekenen, benaderen en manipuleren van functies, zoals in het pakket Chebfun[2].

[1] Fast Fourier Transform
[2] https://www.chebfun.org

Hoofdstuk 5

Benaderen door middel van splinefuncties

5.1 Inleiding

Soms hebben de continue of discrete functies die men wenst te benaderen, een zeer verschillend karakter in verschillende delen van het benaderingsinterval. Op bepaalde plaatsen kan de functie bijv. sterk variabel of oscillerend zijn, terwijl ze elders eerder zachtverlopend en vlak is. Ook kan het zijn dat de te benaderen functie uit verschillende zachtverlopende stukken bestaat die aan elkaar gekoppeld zijn. In zo'n geval zal de functie wel continu zijn, maar ze zal misschien sterke discontinuïteiten vertonen in de afgeleiden. Veeltermen of trigonometrische functies zijn dan niet echt aangewezen als benaderingsfuncties, zie bijv. figuur 5.1. Om een goede nauwkeurigheid te bereiken zou men immers een zeer hoge benaderingsgraad moeten nemen of men zou heel veel trigonometrische functies (ook hoogfrequente) in de benaderingsruimte moeten opnemen. Dat heeft hoge rekentijden tot gevolg en veroorzaakt een sterke voortplanting van afrondingsfouten. Wanneer veeltermen met hoge graad gebruikt worden voor het interpoleren of benaderen van discrete functies, zijn die vaak sterk schommelend tussen de discrete meetwaarden. Ook dat is een reden om op zoek te gaan naar andere benaderingsfuncties.

In zo'n geval gaat men het benaderingsinterval opsplitsen in deelintervallen. Op elk van die deelintervallen zoekt men dan een veeltermbenadering van lagere graad. Men zorgt er ook voor dat de verschillende benaderingen op de gepaste manier aan elkaar worden gekoppeld, d.w.z. met de gepaste graad van continuïteit. Wenst men bijv. een benadering zonder zichtbare knikken, dan leert de ondervinding dat de afgeleiden tot en met de tweede orde in de grenspunten van de deelintervallen continu moeten zijn. Het oog is immers in staat om discontinuïteiten in de kromming —de tweede afgeleide dus— van een functie te onderscheiden. Vertoont de te benaderen functie ergens een duidelijke knik, dan kan men ervoor zorgen dat de benadering die ook heeft, door in dat punt enkel continuïteit te eisen.

Een dergelijke aaneenkoppeling van veeltermen noemt men een *splinefunctie*. In het woordenboek vindt men als vertaling van 'spline': lat, strooklat of elastisch tekenliniaal. Zo'n tekenliniaal werd vroeger veel gebruikt door tekenaars om vloeiende lijnen te tekenen door

Figuur 5.1: Enkele functies die moeilijk door veeltermen te benaderen zijn.

een stel opgegeven punten.

5.2 Enkele basisbegrippen omtrent splinefuncties

5.2.1 Definitie van splinefunctie

In de literatuur vindt men verschillende equivalente definities van splinefuncties. We vertrekken van de zogenaamde *beschrijvende definitie*.

DEFINITIE 5.2.1 *(splinefunctie)*
Zij een strikt stijgende rij van reële getallen gegeven, die voldoet aan

$$a = t_0 < t_1 < t_2 < \ldots < t_{n-1} < t_n = b \ .$$

Een splinefunctie $s(x)$ *van graad* $k > 0$ *(of orde* $k+1$*) met* knooppunten $t_0, t_1, \ldots, t_n$*, is een functie gedefinieerd over* $[a, b]$*, met volgende eigenschappen:*

1. *in elk interval* $[t_{i-1}, t_i]$ *is* $s(x)$ *een veelterm van graad* k *of lager;*
2. *de functie* $s(x)$ *en haar afgeleiden tot en met orde* $k-1$ *zijn continu in* $[a, b]$.

OPMERKINGEN

- Men kan ook splinefuncties van graad nul definiëren. In dat geval laat men de tweede voorwaarde vallen, en vervangt men in de eerste voorwaarde het gesloten interval $[t_{i-1}, t_i]$ door het halfopen interval $[t_{i-1}, t_i)$. De waarde $s(b)$ is dan onbepaald, of moet afzonderlijk worden gedefinieerd. Een splinefunctie van graad nul is dus een stapfunctie.

- Een splinefunctie van graad één is een continue gebroken lijn, met breekpunten in de knooppunten $t_1, t_2, \ldots, t_{n-1}$.

– Een splinefunctie van graad twee noemt men een kwadratische splinefunctie. Ze bestaat uit parabolen, aaneengeschakeld in de knooppunten $t_1, t_2, \ldots, t_{n-1}$ zodanig dat de afgeleiden van twee naburige parabolen er gelijk zijn aan elkaar.

– Om aan de eis van visueel vloeiende krommen te voldoen, om dus te bekomen dat de krommingen van twee naburige veeltermen dezelfde zijn in het gemeenschappelijk knooppunt, zal men met splinefuncties van graad drie of hoger moeten werken. Splinefuncties van graad drie noemt men kubische splinefuncties.

– Elke veelterm over $[a, b]$ van graad k is ook een splinefunctie van graad k over het interval $[a, b]$. In het algemeen zal een splinefunctie van graad k echter gegeven worden door een verschillende veelterm in elk deelinterval. Indien $s^{(k)}(x)$ een discontinuïteit heeft in het inwendig knooppunt t_i, dan zegt men dat t_i een *actief* knooppunt is. Een veelterm is een splinefunctie zonder actieve knooppunten.

– De definitie vereist een strikt stijgende knooppuntenrij. Verder zullen we die voorwaarde verzwakken en ook splinefuncties definiëren met samenvallende knooppunten. Dan laat men toe dat $t_{i-1} < t_i = \ldots = t_{i+l} < t_{i+l+1}$ voor $l \leq k$. In een dergelijk $(l+1)$-voudig knooppunt vereist men dat $s(x)$ continue afgeleiden heeft tot en met orde $k-l-1$. Door het laten samenvallen van knooppunten bekomt men splinefuncties die minder zachtverlopend zijn.

Men kan verschillende randvoorwaarden aan splinefuncties opleggen. Twee bijzondere klassen van splinefuncties moeten hier zeker worden vermeld.

Definitie 5.2.2 *(natuurlijke splinefunctie)*
Een natuurlijke splinefunctie *is een splinefunctie van oneven graad $k = 2m+1$, met $m \geq 1$, waarvoor geldt dat*

$$s^{(j)}(a) = s^{(j)}(b) = 0 \quad \textit{voor} \quad j = m+1, \ldots, 2m \ . \tag{5.1}$$

Definitie 5.2.3 *(periodieke splinefunctie)*
Een periodieke splinefunctie *is een splinefunctie die voldoet aan*

$$s^{(j)}(a) = s^{(j)}(b) \quad \textit{voor} \quad j = 0, \ldots, k-1 \ . \tag{5.2}$$

Dergelijke splines worden bijvoorbeeld gebruikt voor het modelleren van gesloten krommen.

5.2.2 Over het voorstellen van splinefuncties

Er zijn meerdere manieren om splinefuncties voor te stellen. De meest voor de hand liggende bestaat er in de splinefunctie in elk interval voor te stellen als een veelterm,

$$p_i(x) = a_{i0} + a_{i1}x + \ldots + a_{ik}x^k \ . \tag{5.3}$$

Laten we onderzoeken hoeveel coëfficiënten vrij kunnen worden gekozen en hoeveel er door de definitie van de splinefunctie zijn vastgelegd. Door de knooppunten wordt het interval

$[a, b]$ opgesplitst in n deelintervallen. Er zijn dus $n \cdot (k+1)$ veeltermcoëfficiënten. In elk van de $n-1$ inwendige knooppunten worden echter k voorwaarden opgelegd. Dit leert ons dat er $n \cdot (k+1) - (n-1) \cdot k = n + k$ vrijheidsgraden overblijven. We formuleren dit onder de vorm van eens stelling.

STELLING 5.2.1
De vectorruimte van de splinefuncties van graad k met knooppunten $t_0, t_1, \ldots, t_n$ heeft dimensie $n + k$.

Om een basis voor de splineruimte te zoeken, starten we met de klassieke veeltermvoorstelling voor een gegeven splinefunctie $s(x)$ in het eerste knooppunteninterval,

$$s(x) = a_0 + a_1 x + a_2 x^2 + \ldots + a_k x^k \ , \quad x \in [t_0, t_1) \ . \tag{5.4}$$

We gaan dan op zoek naar een voorschrift voor $s(x)$, geldig in de unie van de eerste twee intervallen. Daartoe schrijven we $s(x)$ als de veelterm uit (5.4) waarbij een nog te bepalen 'correctieterm' wordt opgeteld,

$$s(x) = a_0 + a_1 x + a_2 x^2 + \ldots + a_k x^k \ + q_k(x), \quad x \in [t_0, t_2) \ . \tag{5.5}$$

De functie $q_k(x)$ moet aan een reeks voorwaarden voldoen: ze moet identiek nul zijn in $[t_0, t_1)$; ze moet een veelterm zijn van graad k in $[t_1, t_2)$, en opdat de afgeleiden van $s(x)$ tot en met orde $k-1$ continu zouden zijn, moet gelden dat

$$q_k^{(j)}(t_1) = 0 \quad \text{voor} \quad j = 0, \ldots, k-1 \ .$$

Door deze voorwaarden is de functie $q_k(x)$ volledig bepaald. We vinden dat ze evenredig moet zijn met een *afgeknotte-machtsfunctie*,

$$q_k(x) = c_1 (x - t_1)_+^k \ \text{ met } \ (x - t_1)_+^k = \begin{cases} 0 & \text{als } \ x \le t_1 \\ (x - t_1)^k & \text{als } \ x > t_1 \end{cases} \tag{5.6}$$

De constante c_1 is een vrije parameter. Ter illustratie tekenden we in figuur 5.2 een aantal afgeknotte-machtsfuncties voor verschillende waarden van k. Men kan de vorige redenering herhalen om een splinevoorstelling te vinden die geldig is in drie intervallen, daarna in vier intervallen, enzovoort. Uiteindelijk bekomt men een zeer compacte voorstelling:

$$s(x) = \sum_{i=0}^{k} a_i x^i + \sum_{i=1}^{n-1} c_i (x - t_i)_+^k \ . \tag{5.7}$$

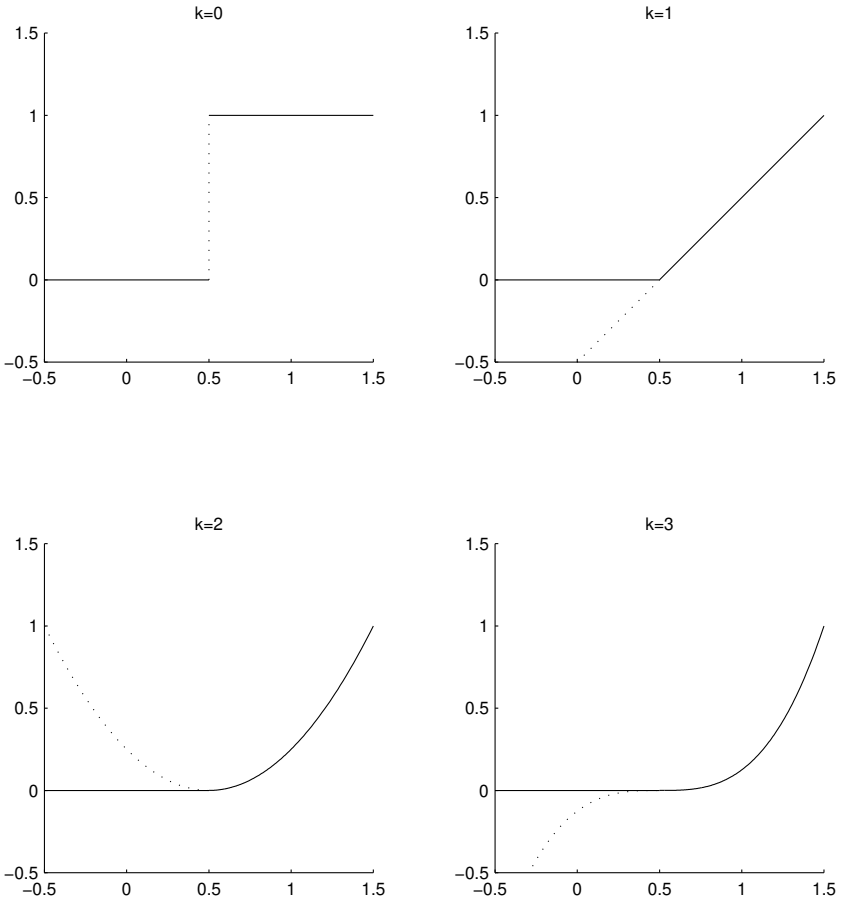
Formule (5.7) bevat slechts $n+k$ coëfficiënten. Dat is inderdaad het kleinst mogelijk aantal. Het stel lineair onafhankelijke functies

$$1, x, x^2, \ldots, x^k, (x - t_1)_+^k, \ldots, (x - t_{n-1})_+^k \tag{5.8}$$

vormt dan ook een basis van de splineruimte. Met een analoge redenering toont men aan dat ook het volgende stel functies een basis is

$$1, x, x^2, \ldots, x^k, (t_1 - x)_+^k, \ldots, (t_{n-1} - x)_+^k \ . \tag{5.9}$$

Met de ondervinding die we reeds hebben opgedaan, vermoeden we echter dat de basissen (5.8) en (5.9) verre van orthogonaal zullen zijn. Dat is inderdaad het geval. Bij berekeningen geeft dit aanleiding tot numerieke instabiliteit. De voorstelling (5.7) wordt dan ook in de praktijk weinig gebruikt.



Figuur 5.2: Afgeknotte-machtsfuncties van graad $k = 0, 1, 2, 3$.

5.2.3 Interpolerende splinefuncties

5.2.3.1 Het opstellen van natuurlijke, interpolerende, kubische splinefuncties

Onder een *interpolerende splinefunctie* verstaat men een splinefunctie die in de knooppunten $t_0, t_1, \ldots, t_n$ bepaalde opgegeven waarden $f_0, f_1, \ldots, f_n$ aanneemt. De meest courant gebruikte interpolerende splinefuncties zijn *natuurlijke, interpolerende splinefuncties*. Een dergelijke functie behoort tot de splineruimte van oneven graad $k = 2m+1$ en voldoet aan

$$s(t_i) = f_i \quad \text{voor} \quad i = 0, 1, \ldots, n \tag{5.10}$$

en

$$s^{(j)}(a) = s^{(j)}(b) = 0 \quad \text{voor} \quad j = m+1, \ldots, 2m\ . \tag{5.11}$$

Dat zijn $(n+1)+2m = n+k$ bijkomende voorwaarden. Vermits de splineruimte dimensie $n+k$ heeft, mogen we vermoeden dat die natuurlijke, interpolerende splinefunctie eenduidig gedefinieerd is. Dat is inderdaad het geval.

In grafische toepassingen worden voornamelijk *kubische* splinefuncties gebruikt. Die zijn immers visueel voldoende zachtverlopend. Een vaak terugkerend praktisch probleem is het tekenen van een vloeiende curve door een aantal opgegeven punten. In de terminologie van dit hoofdstuk kunnen we deze vraag formuleren als volgt: zoek de natuurlijke, interpolerende splinefunctie $s(x)$ van graad 3 die door de punten (x_i, f_i), $i = 0, \ldots, n$ loopt. We beperken ons tot het geval waarin de interpolatiepunten als knooppunten worden gekozen.

De gezochte splinefunctie bestaat uit n opeenvolgende veeltermen $p_1(x), \ldots, p_n(x)$ van graad drie. Veelterm $p_i(x)$ is gedefinieerd in het interval $[x_{i-1}, x_i]$. Uit de continuïteitsvoorwaarden mogen we afleiden dat de tweede afgeleide van de splinefunctie een continue gebroken lijn is. $p_i''(x)$ is dus de interpolerende veelterm van de eerste graad tussen de punten $(x_{i-1}, s''(x_{i-1}))$ en $(x_i, s''(x_i))$. Zijn voorschrift is

$$p_i''(x) = \frac{(x - x_{i-1})s''(x_i) - (x - x_i)s''(x_{i-1})}{\Delta x_i} \quad \text{met} \quad \Delta x_i = x_i - x_{i-1} \ . \tag{5.12}$$

We kunnen (5.12) tweemaal integreren, en vinden dat

$$p_i(x) = \frac{(x - x_{i-1})^3}{6\Delta x_i} s''(x_i) - \frac{(x - x_i)^3}{6\Delta x_i} s''(x_{i-1}) + c_{1i}(x - x_{i-1}) + c_{2i}(x_i - x) \ . \tag{5.13}$$

Hierin zijn de waarden $s''(x_i)$ en de integratieconstanten c_{1i} en c_{2i} nog onbekend. Om deze grootheden te bepalen, beschikken we over de interpolatievoorwaarden en de eigenschap dat de eerste afgeleide $s'(x)$ continu moet zijn. De continuïteit van de tweede afgeleide werd reeds in rekening gebracht in (5.12).

De interpolatievoorwaarde voor $p_i(x)$ in het punt x_i levert

$$p_i(x_i) = \frac{(x_i - x_{i-1})^3}{6\Delta x_i} s''(x_i) + c_{1i}(x_i - x_{i-1}) = f_i \ .$$

Hieruit haalt men onmiddellijk de waarde van de onbekende constante c_{1i} :

$$c_{1i} = \frac{f_i}{\Delta x_i} - \frac{\Delta x_i}{6} s''(x_i) \ . \tag{5.14}$$

De interpolatievoorwaarde $p_i(x_{i-1}) = f_{i-1}$ is te schrijven als

$$p_i(x_{i-1}) = \frac{(x_i - x_{i-1})^3}{6\Delta x_i} s''(x_{i-1}) + c_{2i}(x_i - x_{i-1}) = f_{i-1} \ ,$$

en levert op analoge wijze een uitdrukking voor c_{2i},

$$c_{2i} = \frac{f_{i-1}}{\Delta x_i} - \frac{\Delta x_i}{6} s''(x_{i-1}) \ . \tag{5.15}$$

Wanneer men c_{1i} en c_{2i} elimineert uit (5.13), dan krijgt men

$$\begin{aligned} p_i(x) \ = \ & \frac{f_i(x - x_{i-1}) + f_{i-1}(x_i - x)}{\Delta x_i} + \frac{1}{6}\left[\frac{(x - x_{i-1})^3}{\Delta x_i} - \Delta x_i(x - x_{i-1})\right] s''(x_i) \\ & - \frac{1}{6}\left[\frac{(x - x_i)^3}{\Delta x_i} + \Delta x_i(x_i - x)\right] s''(x_{i-1}) \ . \end{aligned}$$

In de volgende stap zullen we de continuïteit van de eerste afgeleiden gebruiken. Hiertoe stellen we eerst een formule op voor de afgeleide van $p_i(x)$,

$$p_i'(x) = \frac{f_i - f_{i-1}}{\Delta x_i} + \frac{1}{6}\left[\frac{3(x-x_{i-1})^2}{\Delta x_i} - \Delta x_i\right] s''(x_i) - \frac{1}{6}\left[\frac{3(x-x_i)^2}{\Delta x_i} - \Delta x_i\right] s''(x_{i-1})\,.$$

Uit de continuïteit van de afgeleide volgt dat

$$p_i'(x_i) = p_{i+1}'(x_i)\ . \tag{5.16}$$

We rekenen eerst het linkerlid uit,

$$\begin{aligned} p_i'(x_i) &= \frac{\Delta f_i}{\Delta x_i} + \frac{1}{6}\left[\frac{3(\Delta x_i)^2}{\Delta x_i} - \Delta x_i\right] s''(x_i) + \frac{1}{6}\Delta x_i s''(x_{i-1}) \\ &= \frac{\Delta f_i}{\Delta x_i} + \frac{\Delta x_i}{6}\left(s''(x_{i-1}) + 2s''(x_i)\right)\ . \end{aligned}$$

Op dezelfde manier vindt men voor het rechterlid

$$p_{i+1}'(x_i) = \frac{\Delta f_{i+1}}{\Delta x_{i+1}} - \frac{\Delta x_{i+1}}{6}\left(2s''(x_i) + s''(x_{i+1})\right)\ .$$

Invullen in (5.16) geeft de volgende gelijkheid

$$\frac{\Delta f_i}{\Delta x_i} + \frac{\Delta x_i}{6}(s''(x_{i-1}) + 2s''(x_i)) = \frac{\Delta f_{i+1}}{\Delta x_{i+1}} - \frac{\Delta x_{i+1}}{6}(2s''(x_i) + s''(x_{i+1}))\ .$$

Vermenigvuldigen met 6 en rangschikken naar de $s''(x_i)$ levert:

$$\Delta x_i s''(x_{i-1}) + 2(\Delta x_i + \Delta x_{i+1})s''(x_i) + \Delta x_{i+1} s''(x_{i+1}) = 6(\frac{\Delta f_{i+1}}{\Delta x_{i+1}} - \frac{\Delta f_i}{\Delta x_i})\ . \tag{5.17}$$

Vanwege het natuurlijk zijn van $s(x)$ heeft men nog dat

$$s''(x_0) = 0 \quad \text{en} \quad s''(x_n) = 0\ . \tag{5.18}$$

Men vindt de $s''(x_i)$-waarden als oplossing van een stelsel met coëfficiëntenmatrix:

$$\begin{pmatrix} 2(\Delta x_1 + \Delta x_2) & \Delta x_2 & & & & \\ \Delta x_2 & 2(\Delta x_2 + \Delta x_3) & \Delta x_3 & & & \\ & \Delta x_3 & 2(\Delta x_3 + \Delta x_4) & \Delta x_4 & & \\ & & . & . & . & \\ & & & \Delta x_{n-1} & 2(\Delta x_{n-1} + \Delta x_n) & \end{pmatrix} \tag{5.19}$$

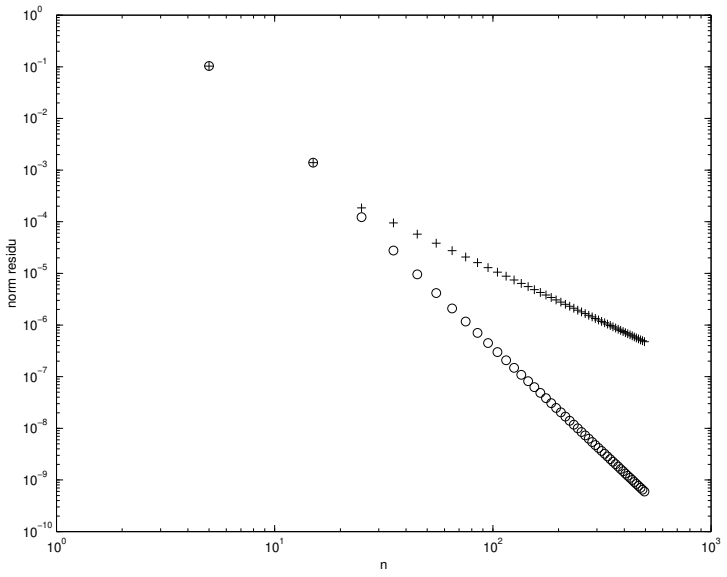
Het rechterlid wordt gegeven door de volgende vector:

$$6\ \left(\frac{\Delta f_2}{\Delta x_2} - \frac{\Delta f_1}{\Delta x_1}\ ;\ \frac{\Delta f_3}{\Delta x_3} - \frac{\Delta f_2}{\Delta x_2}\ ;\ \cdots\ ;\ \frac{\Delta f_n}{\Delta x_n} - \frac{\Delta f_{n-1}}{\Delta x_{n-1}}\right)^T\ . \tag{5.20}$$

Als resultaat vinden we voor ieder deelinterval een veeltermvoorstelling van de vorm (5.13), met de berekende getalwaarden voor $s''(x_i)$ en $s''(x_{i-1})$. De getallen c_{1i} en c_{2i} worden gevonden uit (5.14) en (5.15).

Opmerking 5.1 *We hebben het natuurlijk zijn van de splinefunctie slechts helemaal op het einde in rekening gebracht, via (5.18). Indien we om een of andere reden over de exacte waarde van de tweede afgeleide in de intervaluiteinden zouden beschikken, dan zouden we die kunnen gebruiken voor het opstellen van een meer nauwkeurige interpolerende benadering. De coëfficiëntenmatrix blijft dan dezelfde; enkel de eerste en de laatste component van de rechterlidvector moet worden aangepast.*

Voorbeeld 5.1 *We berekenden interpolerende, kubische splinebenaderingen voor de functie* $1/(1+6x^2)$ *over het interval* $[-1,1]$. *Er werden* $n+1$ *equidistante interpolatiepunten gebruikt. Met die keuze van interpolatiepunten wordt de coëfficiëntenmatrix van het stelsel een tridiagonale matrix met constante diagonalen. In figuur 5.3 geven we de maximale waarde van het residu als functie van de waarde* n. *De "+"-symbolen stemmen overeen met de resultaten voor de natuurlijke spline, dus met* $s''(-1) = s''(1) = 0$. *Voor de resultaten aangeduid met de "∘"-symbolen, gebruikten we de exacte waarde voor de tweede afgeleide in de eindpunten:* $s''(-1) = s''(1) = 204/343$. *Die resultaten zijn duidelijk superieur. Vergelijk deze figuur ook met figuur 4.5, waar dezelfde functie werd benaderd met een interpolerende veelterm.*



Figuur 5.3: Maximale benaderingsfout van interpolerende, kubische splinefuncties met equidistante interpolatiepunten: natuurlijke splines (+) en splines met exacte randvoorwaarde voor de tweede-orde afgeleide (∘).

5.2.3.2 Het opstellen van periodieke, interpolerende, kubische splinefuncties

De redenering van de vorige paragraaf kan worden herhaald voor het opstellen van *periodieke*, interpolerende, kubische splinefuncties. In dat geval is $s''(x)$ een periodieke gebroken lijn, waarbij de waarden $s''(x_0)$ en $s''(x_n)$ aan elkaar gelijk zijn, maar onbekend. We dienen

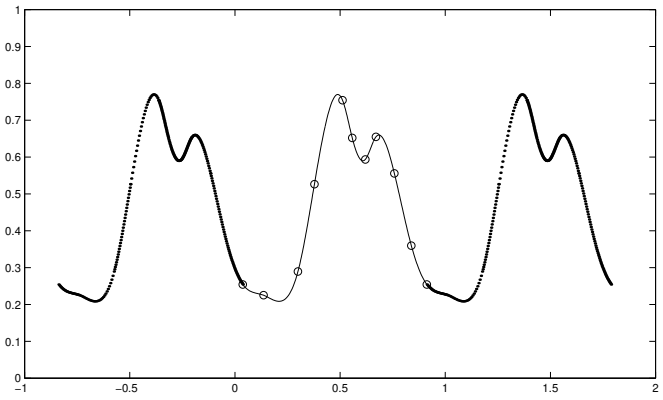
rekening te houden met het feit dat ook $s'(x_0)$ en $s'(x_n)$ aan elkaar gelijk zijn, en met het feit dat $s(x_0) = s(x_n)$, d.w.z. $f_0 = f_n$. Als coëfficiëntenmatrix vinden we

$$\begin{pmatrix} 2(\Delta x_n + \Delta x_1) & \Delta x_1 & & & & \Delta x_n \\ \Delta x_1 & 2(\Delta x_1 + \Delta x_2) & \Delta x_2 & & & \\ & \Delta x_2 & 2(\Delta x_2 + \Delta x_3) & \Delta x_3 & & \\ & & . & . & . & \\ \Delta x_n & & & \Delta x_{n-1} & 2(\Delta x_{n-1} + \Delta x_n) & \end{pmatrix} \tag{5.21}$$

Het rechterlid wordt

$$6 \left(\frac{\Delta f_1}{\Delta x_1} - \frac{\Delta f_n}{\Delta x_n} \; ; \; \frac{\Delta f_2}{\Delta x_2} - \frac{\Delta f_1}{\Delta x_1} \; ; \; \cdots \; ; \; \frac{\Delta f_n}{\Delta x_n} - \frac{\Delta f_{n-1}}{\Delta x_{n-1}} \right)^T . \tag{5.22}$$

Ter illustratie verwijzen we naar figuur 5.4, waar we een periodieke, interpolerende, kubische spline tekenden door 11 gegeven punten. De spline werd getekend over drie perioden.

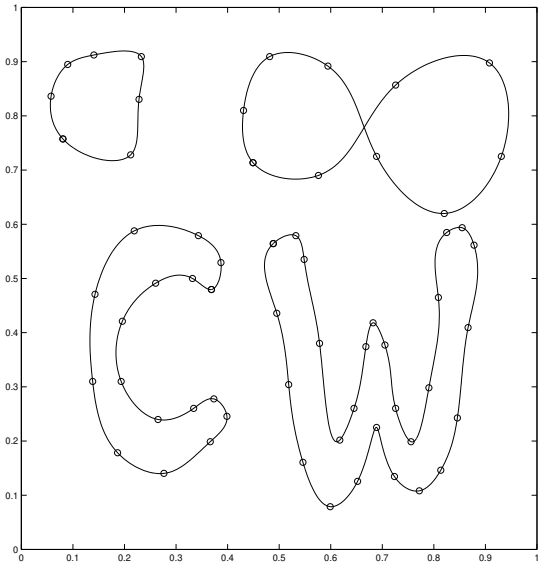


Figuur 5.4: Een periodieke, interpolerende, kubische spline, getekend over drie perioden.

5.2.3.3 Het opstellen van interpolerende, kubische splinecurven

Interpolerende, kubische splinefuncties worden ook vaak gebruikt bij het benaderen of modelleren van *curven*. Een curve in het (x, y)-vlak kan worden voorgesteld in parametervorm als $x = x(t)$ en $y = y(t)$, voor $t \in [a, b]$. Indien de functies $x(t)$ en $y(t)$ periodiek zijn met periode $b - a$, dan beschrijven ze een *gesloten* curve.
Veronderstel dat een stel punten gegeven is in het vlak: $(x_0, y_0), (x_1, y_1), \ldots, (x_n, y_n)$. Men wenst een curve te bepalen die exact door deze punten gaat. Wenst men een gesloten curve, dan zullen we veronderstellen dat $x_0 = x_n$ en $y_0 = y_n$. Vooreerst dient men aan elk punt een parameterwaarde t_i toe te kennen. Deze waarde kunnen we zelf kiezen. Men kan bijvoorbeeld $t_i = i$ nemen. Beter neemt men als t_i de lengte langsheen de gebroken

lijn die de verschillende punten met elkaar verbindt. Wanneer dit is gebeurd, stelt men twee periodieke, interpolerende, kubische splinefuncties op, met de methoden die hierboven werden uitgelegd. In figuur 5.5 werden enkele gesloten kubische splinecurven getekend.



Figuur 5.5: Enkele gesloten, periodieke, interpolerende, kubische splinecurven.

5.3 De B-splinevoorstelling van splinefuncties

5.3.1 Gedeelde differenties

De B-splinebasisfuncties zullen in deze tekst worden gedefinieerd aan de hand van *gedeelde differenties*. Deze laatste werden in een vroegere cursus over numerieke wiskunde reeds behandeld. We herhalen de definitie en de belangrijkste eigenschappen.

DEFINITIE 5.3.4*(gedeelde differentie)*
De gedeelde differentie van orde nul *van een functie f in een punt x_i is gelijk aan*

$$f[x_i] = f(x_i) = f_i \ . \tag{5.23}$$

De gedeelde differentie van orde $j-i$ *voor $j > i$ van een functie f in de punten $x_i, x_{i+1}, \ldots, x_j$ (die alle verschillend zijn) wordt gedefinieerd als*

$$f[x_i, x_{i+1}, \ldots, x_j] = \frac{f[x_{i+1}, \ldots, x_j] - f[x_i, \ldots, x_{j-1}]}{x_j - x_i} \ . \tag{5.24}$$

Soms gebruikt men ook andere notaties voor het neerschrijven van gedeelde differenties. We zullen verder nog onderstaande notatie gebruiken:

$$\Delta_t^k(x_i, x_{i+1}, \ldots, x_{i+k}) f(t) \ . \tag{5.25}$$

Die notatie is wel wat redundant, maar ze geeft heel duidelijk de orde van de differentie aan, de verschillende punten en de onafhankelijk veranderlijke.
Een gedeelde differentie kan gemakkelijk worden uitgerekend met een *differentietabel*. Als voorbeeld geven we de tabel voor het uitrekenen van $f[x_1, x_2, x_3, x_4, x_5]$.

x_1	$f(x_1)$				
x_2	$f(x_2)$	$f[x_1, x_2]$			
x_3	$f(x_3)$	$f[x_2, x_3]$	$f[x_1, x_2, x_3]$		
x_4	$f(x_4)$	$f[x_3, x_4]$	$f[x_2, x_3, x_4]$	$f[x_1, x_2, x_3, x_4]$	
x_5	$f(x_5)$	$f[x_4, x_5]$	$f[x_3, x_4, x_5]$	$f[x_2, x_3, x_4, x_5]$	$f[x_1, x_2, x_3, x_4, x_5]$

De berekening van een waarde in de tabel vraagt twee waarden uit de vorige kolom: die in dezelfde rij en die erboven.

EIGENSCHAPPEN

1. De gedeelde differentie $f[x_i, x_{i+1}, \ldots, x_j]$ is lineair in f, d.w.z.

$$(af + bg)[x_i, \ldots, x_j] = a\, f[x_i, \ldots, x_j] + b\, g[x_i, \ldots, x_j] \ . \tag{5.26}$$

2. De interpolerende veelterm van graad $j - i$ door $(x_i, f_i), \ldots, (x_j, f_j)$ is gelijk aan:

$$\begin{aligned} p_{j-i}(x) \ &= \ a_0 + a_1(x - x_i) + a_2(x - x_i)(x - x_{i+1}) + \ldots \\ &\qquad \ldots + \ a_{j-i}(x - x_i)(x - x_{i+1}) \cdots (x - x_{j-1}) \ , \end{aligned} \tag{5.27}$$

   waarbij de coëfficiënten worden gegeven door

$$a_k = f[x_i, x_{i+1}, \ldots, x_{i+k}] \ . \tag{5.28}$$

   Deze schrijfwijze voor de interpolerende veelterm noemt men de *Newton-vorm*.

3. De gedeelde differentie $f[x_i, x_{i+1}, \ldots, x_j]$ is continu in de argumenten $x_i, x_{i+1}, \ldots, x_j$ als $f(x)$ $(j - i)$-maal differentieerbaar is met continue $(j - i)$-de afgeleide.

4. De gedeelde differentie van orde $j - i$ van een veelterm $p_m(x)$ van graad m, met $m < j - i$, heeft de waarde nul,

$$p_m[x_i, x_{i+1}, \ldots, x_j] = 0 \quad \text{voor} \quad j - i > m \ . \tag{5.29}$$

5. De gedeelde differentie $f[x_i, x_{i+1}, \ldots, x_j]$ is een lineaire samenstelling van de functiewaarden $f_i, f_{i+1}, \ldots, f_j$,

$$f[x_i, x_{i+1}, \ldots, x_j] = \sum_{k=i}^{j} \lambda_k f_k \ . \tag{5.30}$$

6. De *formule van Leibniz* voor de gedeelde differentie van een product van twee functies luidt als volgt: als $f(x) = g(x) \cdot h(x)$ dan is

$$f[x_i, \ldots, x_j] = \sum_{r=i}^{j} g[x_i, \ldots, x_r] \cdot h[x_r, \ldots, x_j] \ . \tag{5.31}$$

5.3.2 Definitie van B-splines

DEFINITIE 5.3.5*(gewone en genormaliseerde B-spline)*
De gewone B-spline *van graad k (of orde $k+1$) wordt gegeven door*

$$M_{i,k+1}(x) = \Delta_t^{k+1}(t_i, t_{i+1}, \ldots, t_{i+k+1})(t-x)_+^k \ . \tag{5.32}$$

De genormaliseerde B-spline *van graad k (of orde $k+1$) wordt gegeven door*

$$N_{i,k+1}(x) = (t_{i+k+1} - t_i)\, M_{i,k+1}(x) \ . \tag{5.33}$$

De gewone B-spline is dus de gedeelde differentie van orde $k+1$ naar de variabele t van de afgeknotte-machtsfunctie $(t-x)_+^k$ over de punten $t_i, \ldots, t_{i+k+1}$. We zullen voorlopig veronderstellen dat de rij $t_i, t_{i+1}, \ldots$ een strikt stijgende rij is. Bij het berekenen van de gedeelde differentie moeten we x als een constante parameter beschouwen. Een gedeelde differentie is een getal. Dat getal zal natuurlijk afhangen van de waarde van de parameter x. De B-spline $M_{i,k+1}(x)$ is dus een functie van x.

STELLING 5.3.2
De functies $M_{i,k+1}(x)$ en $N_{i,k+1}(x)$ zijn splinefuncties.

Bewijs We kunnen (5.30) toepassen om aan te tonen dat

$$M_{i,k+1}(x) = f[t_i, t_{i+1}, ..., t_{i+k+1}] = \sum_{s=i}^{i+k+1} \lambda_s\, f_s \quad \text{met} \quad f_s = (t_s - x)_+^k \ .$$

De functie $M_{i,k+1}(x)$ is dus een lineaire samenstelling van afgeknotte-machtsfuncties van graad k, van de vorm (5.9). We hebben dus te maken met een splinefunctie. Eenzelfde redenering geldt voor $N_{i,k+1}(x)$. □

De afgeknotte-machtsfuncties $(t_s - x)_+^k$ zijn slechts een deel van de splinebasis. Er stelt zich dus het probleem of er voldoende B-splines zijn als basis voor de splineruimte, voor gegeven k en n. Welnu, de dimensie van de splineruimte over de gegeven knooppunten is $n+k$. Met de knooppunten $t_0, t_1, \ldots, t_n$ kunnen we reeds $n-k$ B-splines construeren, namelijk $M_{i,k+1}(x)$ met $i = 0, \ldots, n-k-1$. De laatst gebruikte abscis is dan gelijk aan $t_{(n-k-1)+k+1} = t_n$. We hebben dus nog $(n+k)-(n-k) = 2k$ onafhankelijke basisfuncties nodig om tot een volledige basis te komen. In plaats van machten van x erbij te betrekken, of andere functies, zullen we ons behelpen door nog $2k$ bijkomende knooppunten in te schakelen: k knooppunten links van t_0 en k knooppunten rechts van t_n. We krijgen dus een stel van $n+2k+1$ knooppunten:

$$t_{-k}, t_{-k+1}, \ldots t_{-1}, t_0, t_1, \ldots, t_{n-1}, t_n, t_{n+1}, \ldots, t_{n+k} \ . \tag{5.34}$$

Hierbij horen de $n+k$ B-splines

$$M_{i,k+1}(x) \quad \text{met} \quad i = -k, \ldots, n-1 \ . \tag{5.35}$$

5.3.3 Eigenschappen van B-splines

Met behulp van de gedeelde-differentietabel kunnen we $M_{i,k+1}(x)$ en ook $N_{i,k+1}(x)$ evalueren voor verschillende waarden van x. We deden dit voor een knooppuntenrij met $t_i = i$ en voor verschillende graden k. We tekenden de resulterende B-splines in figuur 5.6. In deze paragraaf gaan we verschillende eigenschappen van die B-splines afleiden.

Eigenschap 5.1

$$M_{i,1}(x) = \begin{cases} \dfrac{1}{t_{i+1} - t_i} & \textit{voor} \quad t_i \leq x < t_{i+1} \\ 0 & \textit{elders} \end{cases} \tag{5.36}$$

$$N_{i,1}(x) = \begin{cases} 1 & \textit{voor} \quad t_i \leq x < t_{i+1} \\ 0 & \textit{elders} \end{cases} \tag{5.37}$$

Bewijs Per definitie van $M_{i,k+1}(x)$ geldt

$$M_{i,1}(x) = \Delta_t^1(t_i, t_{i+1})(t-x)_+^0 = \frac{(t_{i+1} - x)_+^0 - (t_i - x)_+^0}{t_{i+1} - t_i} \ .$$

We vinden (5.36) door het precies in rekening brengen van de formule

$$(t-x)_+^0 = \begin{cases} 0 & \text{voor} \quad t \leq x \\ 1 & \text{voor} \quad t > x \end{cases}$$

Steunend op de definitie van $N_{i,k+1}(x)$ vinden we uit (5.36) onmiddellijk (5.37). □

Eigenschap 5.2

$$M_{i,k+1}(x) = 0 \quad \textit{voor} \quad x \leq t_i \quad \textit{en} \quad x \geq t_{i+k+1} \qquad (k \geq 1) \ . \tag{5.38}$$

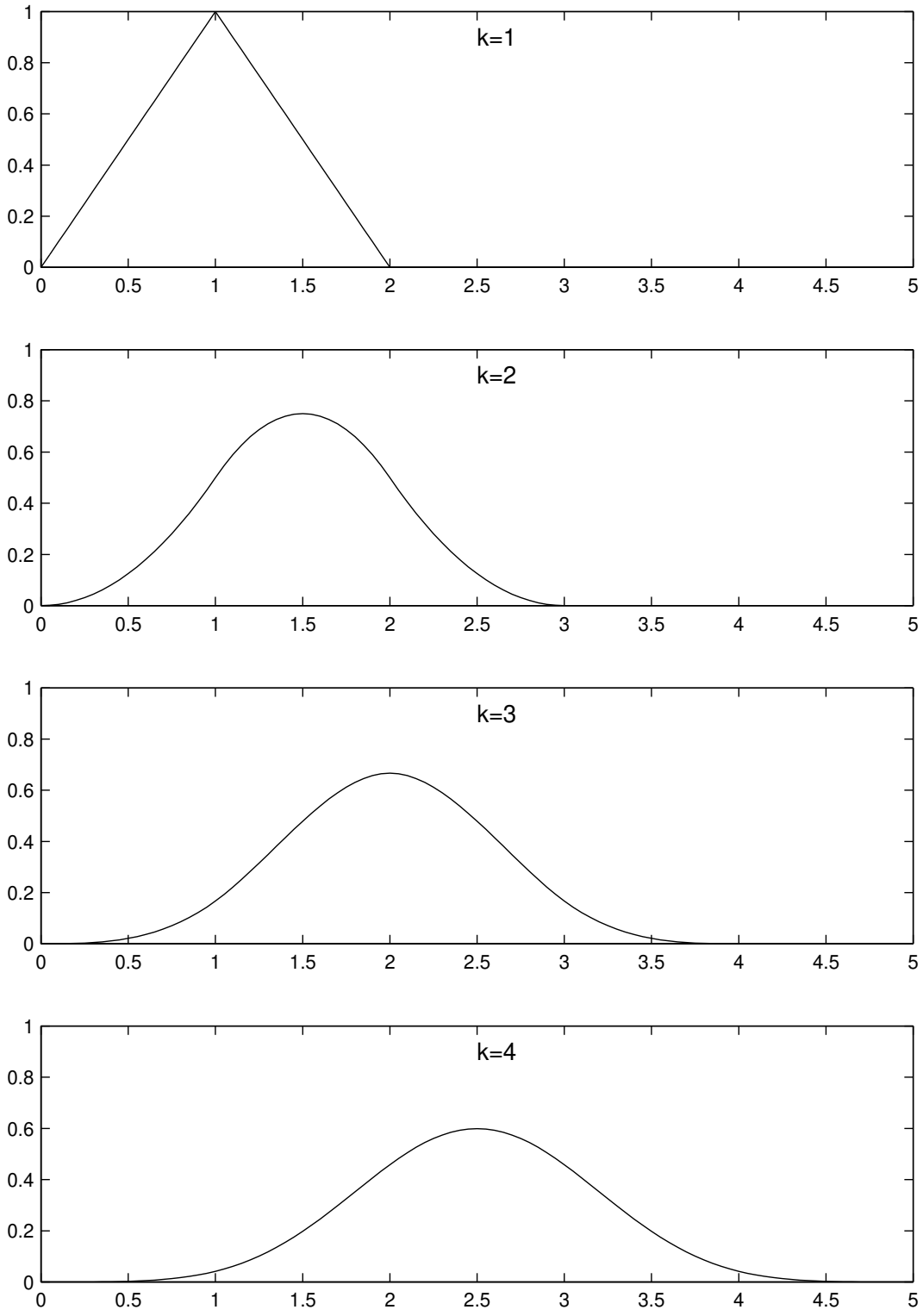
Bewijs We leidden vroeger reeds af dat

$$M_{i,k+1}(x) = \sum_{s=i}^{i+k+1} \lambda_s f_s \quad \text{met} \quad f_s = (t_s - x)_+^k \ .$$

Wanneer nu $x \geq t_{i+k+1}$, dan zijn alle f_s in bovenstaande formule gelijk aan nul.
Als $x \leq t_i$, mogen we in de uitdrukking voor f_s het "+"-teken weglaten, en dit voor elke s. De gewone B-spline wordt dan

$$M_{i,k+1}(x) = \Delta_t^{k+1}(t_i, t_{i+1}, \ldots, t_{i+k+1})(t-x)^k \ .$$

Dat is identiek nul, want differentie van orde $k+1$ van een veelterm van graad k. □



Figuur 5.6: Genormaliseerde B-splines van graad $k = 1, 2, 3, 4$.

Eigenschap 5.3 *Voor de gewone en de genormaliseerde B-splines gelden volgende recursiebetrekkingen:*

$$M_{i,k+1}(x) = \frac{x - t_i}{t_{i+k+1} - t_i} M_{i,k}(x) + \frac{t_{i+k+1} - x}{t_{i+k+1} - t_i} M_{i+1,k}(x) \ , \tag{5.39}$$

$$N_{i,k+1}(x) = \frac{x - t_i}{t_{i+k} - t_i} N_{i,k}(x) + \frac{t_{i+k+1} - x}{t_{i+k+1} - t_{i+1}} N_{i+1,k}(x) \ . \tag{5.40}$$

Bewijs Voor $k \geq 1$ kunnen we schrijven dat

$$(t - x)_+^k = (t - x)_+^{k-1} \cdot (t - x) \ .$$

We vullen dit in in (5.32) en passen de formule van Leibniz toe. Omwille van de factor $(t - x)$, een veelterm van graad 1 in t, bevatten de meeste termen in de som (5.31) een factor die gelijk is aan nul. Inderdaad,

$$\begin{aligned} M_{i,k+1}(x) &= \Delta_t^{k+1}(t_i, t_{i+1}, \ldots, t_{i+k+1})\{(t - x)_+^{k-1} \cdot (t - x)\} \\ &= \Delta_t^{k+1}(t_i, t_{i+1}, \ldots, t_{i+k+1})(t - x)_+^{k-1} \cdot \Delta_t^0(t_{i+k+1})(t - x) \\ &\qquad + \Delta_t^k(t_i, t_{i+1}, \ldots, t_{i+k})(t - x)_+^{k-1} \cdot \Delta_t^1(t_{i+k}, t_{i+k+1})(t - x) \\ &= \Delta_t^{k+1}(t_i, t_{i+1}, \ldots, t_{i+k+1})(t - x)_+^{k-1} \cdot (t_{i+k+1} - x) + M_{i,k}(x) \cdot 1 \end{aligned}$$

Men gebruikt nu de recursieve definitie van gedeelde differentie, om de differentie van orde $k+1$ in bovenstaande uitdrukking te schrijven als een lineaire combinatie van twee gedeelde differenties van orde k. Rekening houdend men de definitie van B-spline, verkrijgt men

$$M_{i,k+1}(x) = \frac{M_{i+1,k}(x) - M_{i,k}(x)}{t_{i+k+1} - t_i} \cdot (t_{i+k+1} - x) + M_{i,k}(x) \ .$$

Recursiebetrekking (5.39) volgt hieruit. Steunend op de definitie van $N_{i,k+1}(x)$, werkt men (5.39) gemakkelijk om tot de recursiebetrekking (5.40). □

Eigenschap 5.4

$$M_{i,k+1}(x) > 0 \quad \textit{voor} \quad t_i < x < t_{i+k+1} \quad (k \geq 1) \ . \tag{5.41}$$

Bewijs We geven een bewijs gebaseerd op de recursiebetrekking en maken gebruik van volledige inductie. Voor $k = 1$ luidt de recursiebetrekking

$$M_{i,2}(x) = \frac{x - t_i}{t_{i+2} - t_i} M_{i,1}(x) + \frac{t_{i+2} - x}{t_{i+2} - t_i} M_{i+1,1}(x) \ .$$

Schrijven we het rechterlid als $A \cdot B + C \cdot D$, dan is $A > 0$ voor $x \in (t_i, \infty)$, $B > 0$ voor $x \in [t_i, t_{i+1})$ en nul daarbuiten, $C > 0$ voor $x \in (-\infty, t_{i+2})$, $D > 0$ voor $x \in [t_{i+1}, t_{i+2})$ en nul daarbuiten. Uit dit alles volgt dat $M_{i,2}(x) > 0$ als $x \in (t_i, t_{i+2})$.

Voor de inductiestap bekijken we de recursiebetrekking voor $M_{i,k+1}(x)$,

$$M_{i,k+1}(x) = \frac{x - t_i}{t_{i+k+1} - t_i} M_{i,k}(x) + \frac{t_{i+k+1} - x}{t_{i+k+1} - t_i} M_{i+1,k}(x) \ .$$

Schrijven we ook hier het rechterlid als $A \cdot B + C \cdot D$, dan vinden we $A > 0$ voor $x \in (t_i, \infty)$, $B > 0$ als $x \in (t_i, t_{i+k})$ en nul daarbuiten, $C > 0$ voor $x \in (-\infty, t_{i+k+1})$, $D > 0$ als $x \in (t_{i+1}, t_{i+k+1})$ en nul daarbuiten. Er volgt dat $M_{i,k+1}(x) > 0$ als $x \in (t_i, t_{i+k+1})$. □

Eigenschap 5.5

$$M^{(j)}_{i,k+1}(t_i) = M^{(j)}_{i,k+1}(t_{i+k+1}) = 0 \quad \textit{voor} \quad j = 0, \ldots, k-1 \quad (k \geq 1) \ . \tag{5.42}$$

Bewijs We toonden reeds aan dat $M_{i,k+1}(x)$ een splinefunctie is. Het gevraagde volgt dan ook onmiddellijk uit de tweede voorwaarde in de definitie van splinefuncties en uit (5.38). □

Eigenschap 5.6

$$\sum_{i=-k}^{n-1} N_{i,k+1}(x) = 1 \quad \textit{voor} \quad x \in [t_0, t_n] \quad (k \geq 1) \ . \tag{5.43}$$

Bewijs We gebruiken de recursiebetrekking en (5.38), om aan te tonen dat voor $x \in [t_j, t_{j+1})$ geldt

$$\begin{aligned}
\sum_{i=-k}^{n-1} N_{i,k+1}(x) &= \sum_{i=j-k}^{j} N_{i,k+1}(x) \\
&= \sum_{i=j-k}^{j} \left\{ \frac{x - t_i}{t_{i+k} - t_i} N_{i,k}(x) + \frac{t_{i+k+1} - x}{t_{i+k+1} - t_{i+1}} N_{i+1,k}(x) \right\} \\
&= \frac{x - t_{j-k}}{t_j - t_{j-k}} N_{j-k,k}(x) + \sum_{i=j-(k-1)}^{j} \left\{ \frac{x - t_i}{t_{i+k} - t_i} + \frac{t_{i+k} - x}{t_{i+k} - t_i} \right\} N_{i,k}(x) \\
&\qquad + \frac{t_{j+k+1} - x}{t_{j+k+1} - t_{j+1}} N_{j+1,k}(x) \ .
\end{aligned}$$

Uit (5.38) volgt dat $N_{j-k,k}(x)$ en $N_{j+1,k}(x)$ identiek nul zijn voor $t_j \leq x < t_{j+1}$. Dus

$$\sum_{i=j-k}^{j} N_{i,k+1}(x) = \sum_{i=j-(k-1)}^{j} N_{i,k}(x) \ .$$

We kunnen op analoge manier verder gaan. We vinden

$$\sum_{i=j-k}^{j} N_{i,k+1}(x) = \ldots = \sum_{i=j-1}^{j} N_{i,2}(x) = N_{j,1}(x) \ .$$

Voor $x \in [t_j, t_{j+1})$ is het rechterlid gelijk aan 1. Dat bewijst de stelling voor alle $x \in [t_0, t_n)$. Het geval $x = t_n$ volgt uit de continuïteit van de functie $\sum_{i=-k}^{n-1} N_{i,k+1}(x)$ voor $k \geq 1$. □

Eigenschap 5.7 *De (rechter)afgeleide van $N_{i,k+1}(x)$ wordt gegeven door*

$$N'_{i,k+1}(x) = k\left(\frac{N_{i,k}(x)}{t_{i+k}-t_i} - \frac{N_{i+1,k}(x)}{t_{i+k+1}-t_{i+1}}\right) \qquad (k \geq 1)\ . \tag{5.44}$$

Bewijs In het bewijs zullen we de volgende eigenschap gebruiken:

$$\frac{d}{dx}(t-x)_+^k = -k(t-x)_+^{k-1}\ .$$

Deze formule is steeds geldig, uitgezonderd voor het berekenen van de afgeleide in het punt $x = t$ voor $k = 1$. In dat geval geeft de formule de waarde van de rechterafgeleide.
We bewijzen nu (5.44).

$$\begin{aligned}
N'_{i,k+1}(x) &= \frac{d}{dx}\{(t_{i+k+1}-t_i)\,\Delta_t^{k+1}(t_i,t_{i+1},\ldots,t_{i+k+1})(t-x)_+^k\} \\
&= (t_{i+k+1}-t_i)\,\Delta_t^{k+1}(t_i,t_{i+1},\ldots,t_{i+k+1})\frac{d}{dx}(t-x)_+^k \\
&= -k(t_{i+k+1}-t_i)\,\Delta_t^{k+1}(t_i,t_{i+1},\ldots,t_{i+k+1})(t-x)_+^{k-1} \\
&= -k\left(\Delta_t^k(t_{i+1},\ldots,t_{i+k+1})(t-x)_+^{k-1} - \Delta_t^k(t_i,\ldots,t_{i+k})(t-x)_+^{k-1}\right) \\
&= k\left(\frac{N_{i,k}(x)}{t_{i+k}-t_i} - \frac{N_{i+1,k}(x)}{t_{i+k+1}-t_{i+1}}\right)
\end{aligned}$$

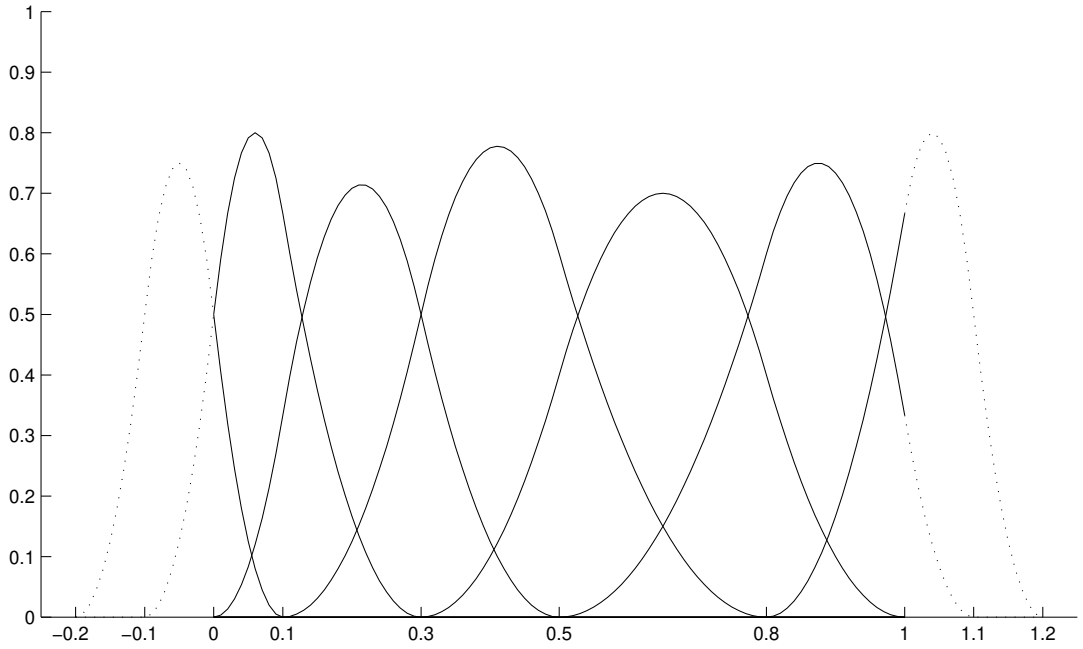
□

Voorbeeld 5.2 *We beschouwen de volgende knooppuntenrij:* 0, 0.1, 0.3, 0.5, 0.8, 1*; dus met $n = 5$. De ruimte van de kwadratische splinefuncties gedefinieerd over deze knooppuntenrij heeft dimensie $n + k = 7$. De ruimte van de kubische splinefuncties heeft dimensie 8.*

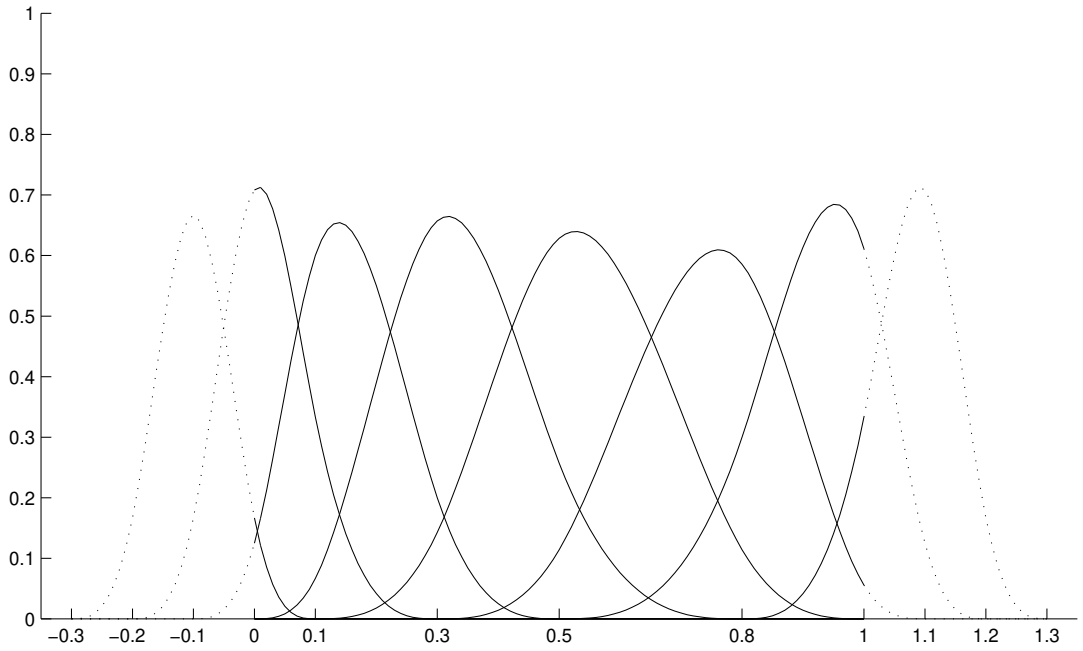
Voor het bepalen van een basis van de ruimte van kwadratische splinefuncties bepaald door de gegeven knooppunten, vulden we de rij aan met twee knooppunten links en twee knooppunten rechts: -0.2, -0.1 *en* 1.1, 1.2*. Dat geeft een totaal van $n+2k+1 = 10$ punten. Die knooppunten bepalen 7 B-splines van graad twee. Het zijn de volgende B-splines: $N_{-2,3}(x), N_{-1,3}(x), \ldots, N_{4,3}(x)$. Ze werden getekend in figuur 5.7. Enkel de gedeelten van de B-splines gelegen binnen het interval $[0, 1]$ zijn relevant.*

Voor het construeren van een basis van de ruimte van kubische splinefuncties bepaald door de gegeven knooppunten, vulden we de rij aan met de volgende waarden: $-0.3, -0.2, -0.1$ *en* $1.1, 1.2, 1.3$*. Dat levert een totaal van 12 punten, die de 8 kubische B-splines $N_{-3,4}(x)$, $N_{-2,4}(x)$ tot $N_{4,4}(x)$ volledig vastleggen. Ze werden getekend in figuur 5.8.*

De figuren laten ons toe om visueel een aantal eigenschappen van B-splines na te gaan: elke B-spline verschilt van de nulfunctie in slechts $k + 1$ intervallen, en wordt bepaald door $k + 2$ knooppunten. In een willekeurig punt van het interval $[t_0, t_n]$ zijn hoogstens $k + 1$ B-splines verschillend van nul. Het zijn er juist $k + 1$, indien het punt geen knooppunt is; indien het wel een knooppunt is, dan zijn er slechts k B-splines verschillend van nul.



Figuur 5.7: Een basis van kwadratische B-splines.



Figuur 5.8: Een basis van kubische B-splines.

5.3.4 Operaties op splinefuncties in B-splinevoorstelling

De splinefunctie $s(x)$ kan worden voorgesteld als een lineaire combinatie van $n+k$ B-splines,

$$s(x) = \sum_{i=-k}^{n-1} c_i N_{i,k+1}(x) \ . \tag{5.45}$$

Deze voorstelling leent zich goed tot een snelle evaluatie en differentiatie van de spline.

5.3.4.1 Het evalueren van een splinefunctie

Een zeer efficiënt en stabiel algoritme voor het evalueren van $s(x)$ zonder eerst de B-splines afzonderlijk te evalueren, werd in 1972 door Carl de Boor voorgesteld, zie [dB78]. Het algoritme maakt gebruik van de recursiebetrekking voor genormaliseerde B-splines.

STELLING 5.3.3 *(de Boor)*
Zij $x \in [t_j, t_{j+1})$. Dan geldt $s(x) = c_j^{[k]}$. De constante $c_i^{[0]} = c_i$, en $c_i^{[r]}$ wordt gevonden uit

$$c_i^{[r]} = \alpha_{i,r}\, c_i^{[r-1]} + (1-\alpha_{i,r})\, c_{i-1}^{[r-1]} \quad \textit{met} \quad \alpha_{i,r} = \frac{x - t_i}{t_{i+k+1-r} - t_i} \ . \tag{5.46}$$

Bewijs

$$\begin{aligned}
s(x) &= \sum_{i=-k}^{n-1} c_i \left(\frac{x-t_i}{t_{i+k}-t_i}\, N_{i,k}(x) + \frac{t_{i+k+1}-x}{t_{i+k+1}-t_{i+1}}\, N_{i+1,k}(x) \right) \\
&= \sum_{i=-(k-1)}^{n-1} c_i \frac{x-t_i}{t_{i+k}-t_i}\, N_{i,k}(x) + \sum_{i=-k}^{n-2} c_i \frac{t_{i+k+1}-x}{t_{i+k+1}-t_{i+1}}\, N_{i+1,k}(x) \\
&= \sum_{i=-(k-1)}^{n-1} \left(c_i \frac{x-t_i}{t_{i+k}-t_i} + c_{i-1} \frac{t_{i+k}-x}{t_{i+k}-t_i} \right) N_{i,k}(x) \\
&= \sum_{i=-(k-1)}^{n-1} c_i^{[1]}\, N_{i,k}(x)
\end{aligned}$$

Schrijven we $c_i^{[0]} = c_i$, dan vinden we de $c_i^{[1]}$-coëfficiënten als

$$c_i^{[1]} = \alpha_{i,1}\, c_i^{[0]} + (1-\alpha_{i,1})\, c_{i-1}^{[0]} \quad \text{met} \quad \alpha_{i,1} = \frac{x-t_i}{t_{i+k}-t_i} \ . \tag{5.47}$$

Op een analoge manier kunnen we verder gaan en vinden dat

$$s(x) = \sum_{i=-(k-r)}^{n-1} c_i^{[r]} N_{i,k+1-r}(x) \ = \cdots = \sum_{i=0}^{n-1} c_i^{[k]} N_{i,1}(x) \ .$$

waarbij $c_i^{[r]}$ voldoet aan (5.46). Voor $x \in [t_j, t_{j+1})$ zijn alle B-splines van eerste orde gelijk aan nul, op $N_{j,1}(x)$ na. Die neemt er de waarde 1 aan. We vinden dus: $s(x) = c_j^{[k]}$. □

De berekeningen van het de Boor-algoritme kunnen in een tabel worden geordend.

$$
\begin{array}{llllll}
c_{j-k}^{[0]} = c_{j-k} & & & & & \\
c_{j-k+1}^{[0]} = c_{j-k+1} & c_{j-k+1}^{[1]} & & & & \\
c_{j-k+2}^{[0]} = c_{j-k+2} & c_{j-k+2}^{[1]} & c_{j-k+2}^{[2]} & & & \\
\vdots & \vdots & \vdots & \ddots & & \\
c_{j-1}^{[0]} = c_{j-1} & c_{j-1}^{[1]} & \cdots & \cdots & c_{j-1}^{[k-1]} & \\
c_{j}^{[0]} = c_{j} & c_{j}^{[1]} & \cdots & \cdots & c_{j}^{[k-1]} & c_{j}^{[k]} = s(x)
\end{array}
$$

Voor de evaluatie van $s(x)$ in het punt b, voert men het algoritme uit met waarde $j = n-1$.

5.3.4.2 Het differentiëren van een splinefunctie

We stelden reeds een uitdrukking op voor de afgeleide van een genormaliseerde B-spline. Die uitdrukking kan nu worden gebruikt voor het opstellen van een formule voor de afgeleiden van splinefuncties in B-splinevoorstelling.

STELLING 5.3.4
De afgeleide van orde r van een splinefunctie $s(x)$ voldoet aan

$$s^{(r)}(x) = \sum_{i=-(k-r)}^{n-1} c_i^{(r)} N_{i,k+1-r}(x) \tag{5.48}$$

met $c_i^{(0)} = c_i$ en

$$c_i^{(r)} = (k+1-r)\,\frac{c_i^{(r-1)} - c_{i-1}^{(r-1)}}{t_{i+k+1-r} - t_i}\ . \tag{5.49}$$

Indien $r = k$, dan geeft (5.48) de rechterafgeleide.

Bewijs We bewijzen het geval $r = 1$. De formule voor de hogere afgeleiden volgt door inductie.

$$
\begin{aligned}
s'(x) &= \sum_{i=-k}^{n-1} c_i^{(0)} N'_{i,k+1}(x) \\
&= \sum_{i=-k}^{n-1} c_i^{(0)}\, k\, \left(\frac{N_{i,k}(x)}{t_{i+k} - t_i} - \frac{N_{i+1,k}(x)}{t_{i+k+1} - t_{i+1}} \right) \\
&= \sum_{i=-(k-1)}^{n-1} c_i^{(1)} N_{i,k}(x)
\end{aligned}
$$

waarbij de coëfficiënt $c_i^{(1)}$ gelijk is aan

$$c_i^{(1)} = k\,\frac{c_i^{(0)} - c_{i-1}^{(0)}}{t_{i+k} - t_i}\ . \tag{5.50}$$

□

De r-de afgeleide van een splinefunctie van graad k is dus een splinefunctie van graad $k-r$, waarvan de coëfficiënten recursief worden gevonden met behulp van uitdrukking (5.49). Wenst men de afgeleide te evalueren in een punt $x \in [t_j, t_{j+1})$, dan stelt men de coëfficiënten van de B-splines die in x verschillen van nul, onder elkaar in een kolom. De berekeningen worden dan geordend in een trapeziumvormige tabel; de laatste kolom van die tabel geeft de coëfficiënten van de afgeleide splinefunctie.

$$\begin{array}{llll}
c_{j-k}^{(0)} = c_{j-k} & & & \\
c_{j-k+1}^{(0)} = c_{j-k+1} & c_{j-k+1}^{(1)} & & \\
c_{j-k+2}^{(0)} = c_{j-k+2} & c_{j-k+2}^{(1)} & \ddots & \\
\vdots & \vdots & \vdots & c_{j-k+r}^{(r)} \\
\vdots & \vdots & \vdots & \vdots \\
c_j^{(0)} = c_j & c_j^{(1)} & \dots & c_j^{(r)}
\end{array}$$

Voor het evalueren van die afgeleide, vertrekt men van de laatste kolom van bovenstaande tabel, en men past er het rekenschema van de Boor op toe. Wenst men de linkerafgeleide te berekenen in het punt b, dan past men het algoritme toe met $j = n-1$.

5.4 Enkele uitbreidingen

5.4.1 Splinefuncties met samenvallende knooppunten

Splinefuncties kunnen eveneens worden gedefinieerd in het geval dat meerdere knooppunten met elkaar samenvallen. Hiertoe moeten we eerst de definitie van gedeelde differenties, en meer bepaald formule (5.24), aanpassen. We veronderstellen verder dat de x_i geordend werden volgens toenemende grootte. De nieuwe definitie luidt dan als volgt:

$$f[x_i, x_{i+1}, \dots, x_j] = \begin{cases} \dfrac{f[x_{i+1}, \dots, x_j] - f[x_i, \dots, x_{j-1}]}{x_j - x_i} & x_i < x_j \\ \dfrac{f^{(j-i)}(x_i)}{(j-i)!} & x_i = \dots = x_j \end{cases} \tag{5.51}$$

Als voorbeeld stellen we de tabel op voor het uitrekenen van de waarde $f[a,a,a,b,b]$:

a	$f(a)$				
a	$f(a)$	$f'(a)$			
a	$f(a)$	$f'(a)$	$f''(a)/2$		
b	$f(b)$	$f[a,b]$	$f[a,a,b]$	$f[a,a,a,b]$	
b	$f(b)$	$f'(b)$	$f[a,b,b]$	$f[a,a,b,b]$	$f[a,a,a,b,b]$

We kunnen nu onmiddellijk een aantal belangrijke eigenschappen van gedeelde differenties zoals gezien in §5.3.1, veralgemenen. We doen dit hier zonder bewijs. De eerste eigenschap rechtvaardigt de keuze van de aangepaste definitie (5.51). De tweede eigenschap is belangrijk voor het afleiden van de continuïteitseigenschappen van splinefuncties gedefinieerd voor samenvallende knooppunten.

Eigenschappen

1. Zij $a_k = f[x_i, x_{i+1}, \ldots, x_{i+k}]$, dan geldt dat de veelterm

$$\begin{aligned} p_{j-i}(x) &= a_0 + a_1(x - x_i) + a_2(x - x_i)(x - x_{i+1}) + \ldots \\ &\qquad \ldots + a_{j-i}(x - x_i)(x - x_{i+1}) \cdots (x - x_{j-1}) \ , \end{aligned}$$

   de veelterm is die in de knooppunten x_s voldoet aan

$$p^{(r)}_{j-i}(x_s) = f^{(r)}(x_s) \text{ met } \ r = 0, \ldots, l \ ,$$

   wanneer knooppunt x_s een meervoudigheid $l+1$ heeft ($x_s = x_{s+1} = \ldots = x_{s+l}$). Deze eigenschap noemt men de *interpolatie-eigenschap* van gedeelde differenties.

2. De gedeelde differentie $f[x_i, \ldots, x_j]$ is een lineaire combinatie van de functiewaarden $f(x_s)$ (in het geval van een enkelvoudig knooppunt x_s) en van de functiewaarden en afgeleiden $f^{(r)}(x_s),\ r = 0, \ldots, l$ (in het geval van een $(l+1)$-voudig knooppunt x_s). Deze eigenschap wordt de *lineaire-samenstellingseigenschap* genoemd.

Een *B-spline* kan op dezelfde wijze gedefinieerd worden als voorheen. Meer bepaald,

$$M_{i,k+1}(x) = \Delta_t^{k+1}(t_i, t_{i+1}, \ldots, t_{i+k+1})(t - x)_+^k \ ,$$

waarbij nu de veralgemeende definitie voor gedeelde differenties wordt gebruikt. Uit de tweede eigenschap volgt dat $M_{i,k+1}(x)$ een lineaire combinatie is van termen van de vorm $(t_s - x)_+^{k-r},\ r = 0, \ldots, l$. Hierbij veronderstellen we dat het knooppunt t_s een $(l+1)$-voudig knooppunt is.

Een *splinefunctie* tenslotte is het gemakkelijkst te definiëren als een lineaire combinatie van dergelijke B-splines. Uit de continuïteitseigenschappen van de afgeleiden van de afgeknotte-machtsfuncties kunnen de continuïteitseigenschappen van de splinefunctie zelf worden afgeleid.

- In een enkelvoudig knooppunt is de splinefunctie continu, alsook zijn afgeleiden tot en met orde $k-1$.
- In een l-voudig knooppunt met $l \leq k$ zijn de afgeleiden van de splinefunctie continu tot en met orde $k-l$.
- In een $(k+1)$-voudig knooppunt is de splinefunctie mogelijk discontinu.

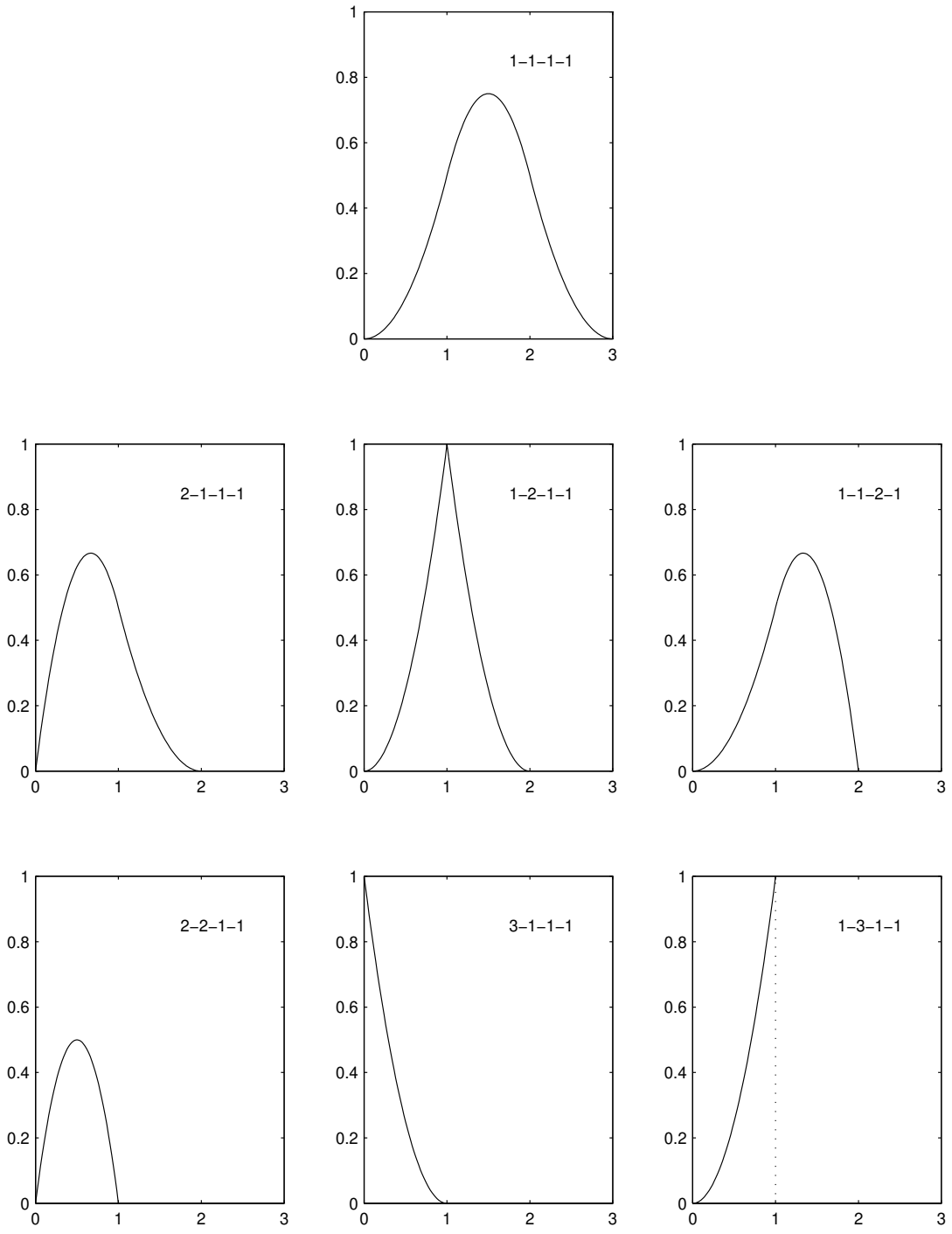
– Wanneer een knooppunt meervoudigheid $k+2$ of hoger heeft, dan is met dat punt een B-spline geassocieerd die volledig nul is. Dat geval zullen we daarom verder uitsluiten.

Deze verschillende gevallen werden geïllustreerd in de figuren 5.9 en 5.10.

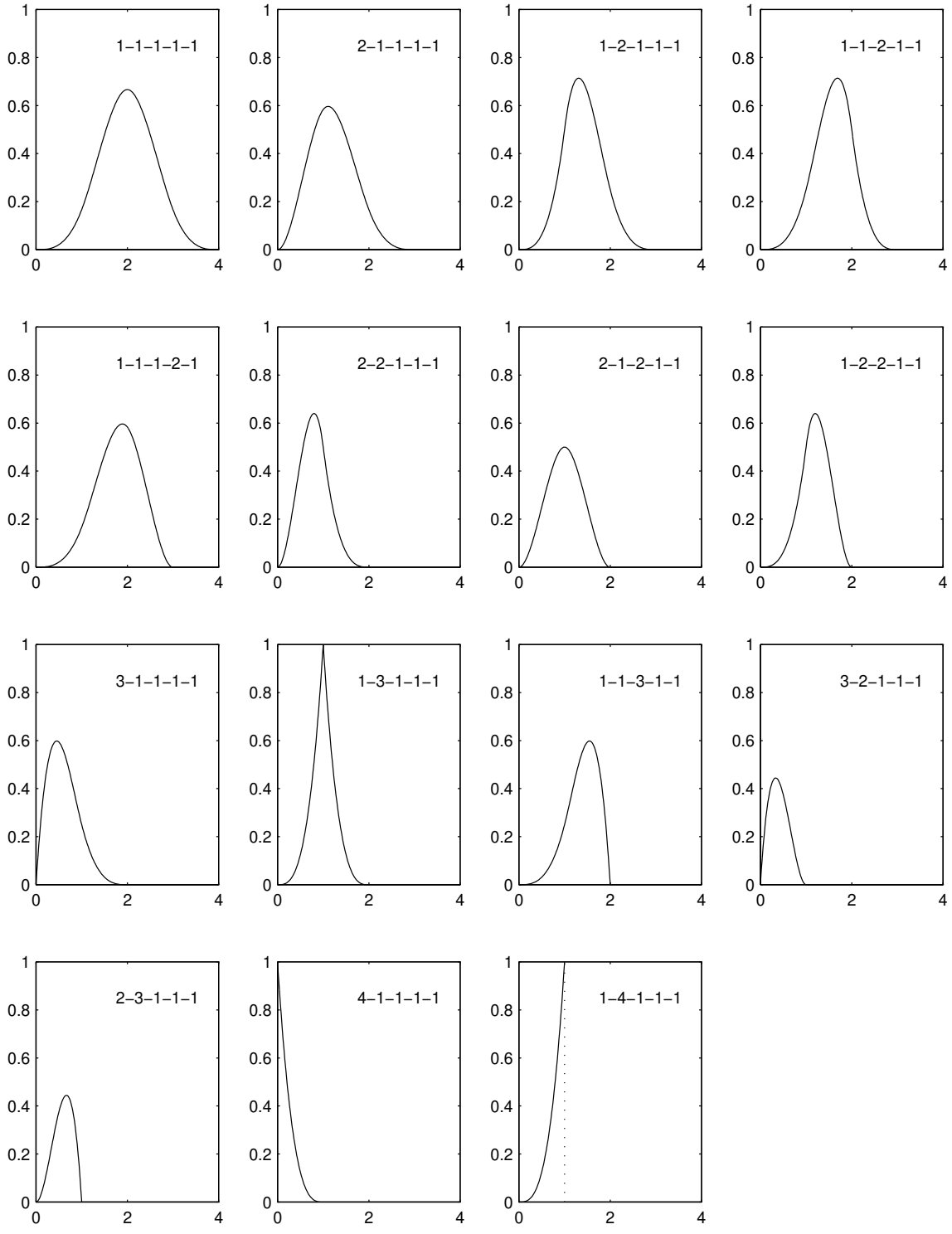
De splinefuncties kunnen worden geëvalueerd met de vroeger besproken methodes. De afleiding van de verschillende evaluatiemethodes blijft immers geldig wanneer we steunen op de veralgemeende definitie van gedeelde differenties. Voor elk evaluatiepunt x bepaalt men eerst een knooppunteninterval zodat $x \in [t_j, t_{j+1})$. Daarna kan men bijvoorbeeld het recursieschema van de Boor toepassen. In figuur 5.11 werden enkele splinefuncties getekend met samenvallende knooppunten. Dergelijke splinefuncties laten toe functies en curven te te modelleren met discontinuïteiten in de functiewaarde of de afgeleiden.

5.4.2 Splinefuncties in meerdere veranderlijken

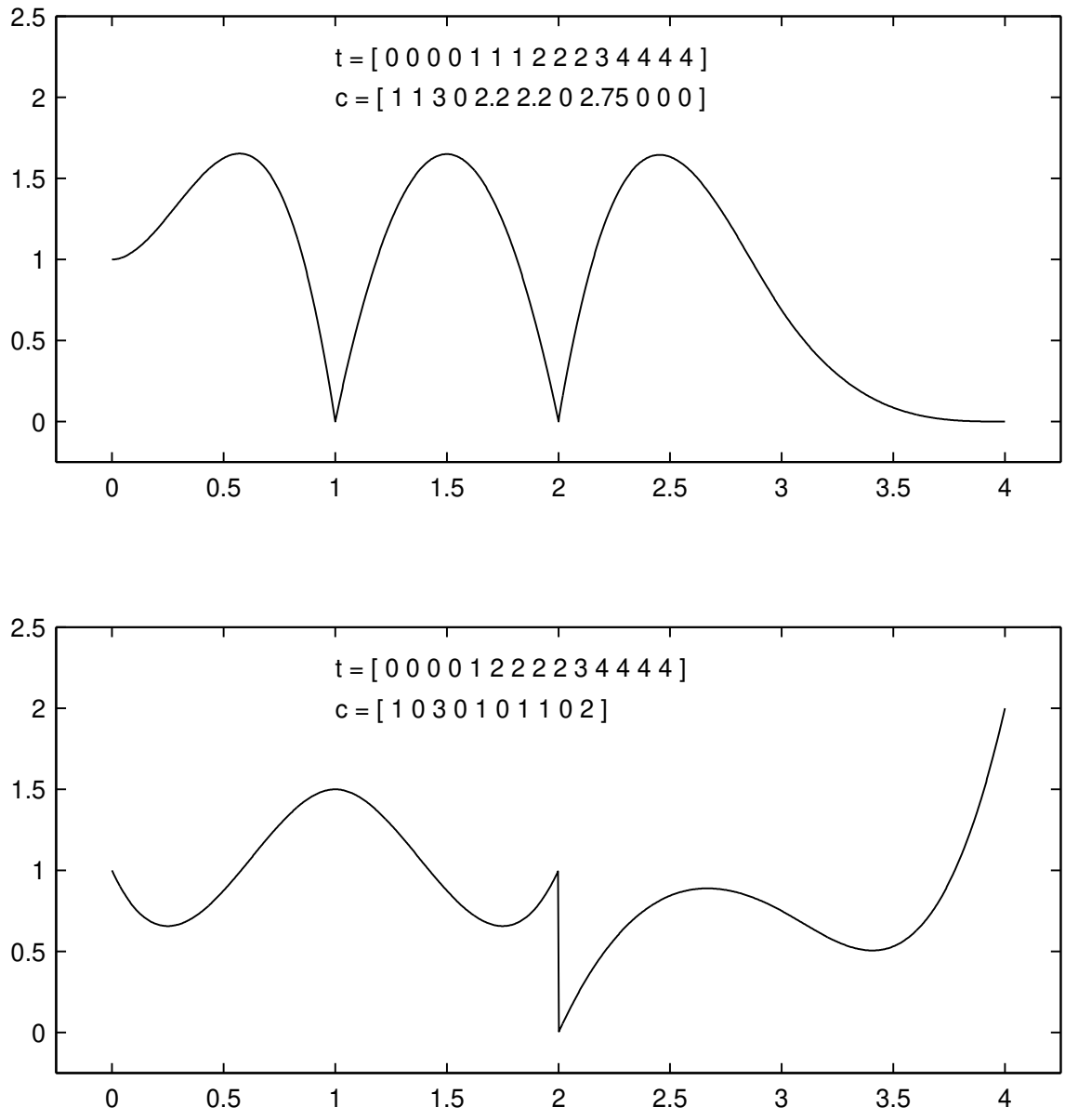
Wat werd gezegd omtrent splinefuncties in één veranderlijke, kan gemakkelijk worden uitgebreid tot splines in meerdere veranderlijken. We beperken ons hier tot bivariate splinefuncties, gedefinieerd op een regelmatig rechthoekig rooster van knooppunten. Dergelijke splines noemt men *tensorproduct-splinefuncties*. Voor meer algemene splinefuncties, gedefinieerd op driehoekige roosters of op meerdimensionale simplices, verwijzen we naar het boek [Die95].



Figuur 5.9: Kwadratische B-splines met samenvallende knooppunten. De knooppunten zijn: 0, 1, 2 en 3. Hun meervoudigheid wordt rechtsboven in elke figuur gegeven. Telkens werd de genormaliseerde B-spline $N_{-2,3}(x)$ getekend.



Figuur 5.10: Kubische B-splines met samenvallende knooppunten. De knooppunten zijn: 0, 1, 2, 3 en 4. Hun meervoudigheid wordt rechtsboven in elke figuur gegeven. Telkens werd de genormaliseerde B-spline $N_{-3,4}(x)$ getekend.



Figuur 5.11: Enkele kubische splinefuncties met samenvallende knooppunten. In elke figuur wordt de knooppuntenrij t vermeld, alsook de rij van coëfficiënten c.

DEFINITIE 5.4.6 *(tensorproduct-splinefunctie)*
Beschouw het rechthoekig gebied $R = [a, b] \times [c, d]$ *en twee strikt stijgende rijen*

$$a = \lambda_0 < \lambda_1 < \ldots < \lambda_g = b \quad en \quad c = \mu_0 < \mu_1 < \ldots < \mu_h = d \ .$$

De functie $s(x, y)$*, gedefinieerd over* R*, noemt men een* tensorproduct-splinefunctie *van graad* $k > 0$ *in* x *en graad* $l > 0$ *in* y*, met knooppunten* $\{\lambda_i\}_{i=0}^{g}$ *in de* x*-richting en* $\{\mu_j\}_{j=0}^{h}$ *in de* y*-richting, als voldaan is aan de volgende twee voorwaarden:*

1. *In elke deelrechthoek* $[\lambda_i, \lambda_{i+1}] \times [\mu_j, \mu_{j+1}]$ *is* $s(x, y)$ *een veelterm van graad* k *in* x *en graad* l *in* y*.*

2. *Alle partiële afgeleiden* $\dfrac{\partial^{i+j} s(x, y)}{\partial x^i \partial y^j}$*, voor* $0 \leq i < k$ *en* $0 \leq j < l$*, zijn continu in* R*.*

We beperkten ons in bovenstaande definitie tot het geval van niet-samenvallende knooppunten. De uitbreiding naar samenvallende knooppunten ligt voor de hand. Voor de meeste toepassingen in meerdere dimensies gebruikt men splines van graad $k = 3$. In twee dimensies spreekt men van *bikubische* splines.

Men gaat gemakkelijk na dat de dimensie van de splineruimte gelijk is aan $(g + k) \cdot (h + l)$. Voor het opstellen van een B-splinebasis, voegen we $2\,k$ knooppunten toe in de x-richting,

$$\lambda_{-k} \leq \lambda_{-k+1} \leq \ldots \leq \lambda_1 \leq \lambda_0 = a \quad \text{en} \quad b = \lambda_g \leq \lambda_{g+1} \leq \ldots \leq \lambda_{g+k} \ ,$$

en $2\,l$ knooppunten in de y-richting,

$$\mu_{-l} \leq \mu_{-l+1} \leq \ldots \leq \mu_1 \leq \mu_0 = c \quad \text{en} \quad d = \mu_h \leq \mu_{h+1} \leq \ldots \leq \mu_{h+l} \ .$$

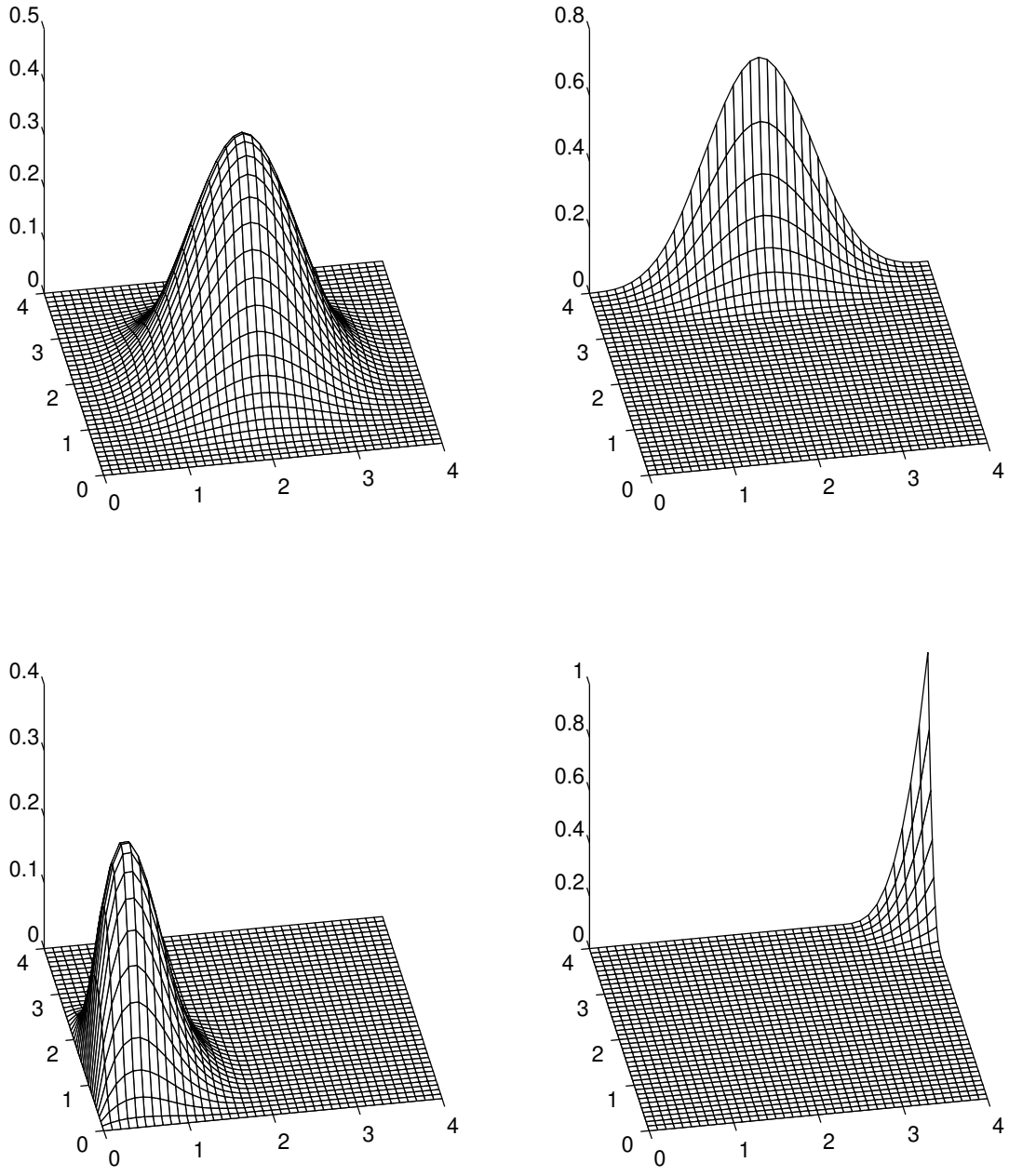
Elke splinefunctie $s(x, y)$ kan dan worden geschreven als een lineaire combinatie van de kruisproducten van de ééndimensionale B-splines $N_{i,k+1}^{\lambda}(x)$ en $N_{j,l+1}^{\mu}(y)$. (We voegen de bovenindices λ en μ toe om aan te duiden welke knooppuntenrij beschouwd wordt). Dus,

$$s(x, y) = \sum_{i=-k}^{g-1} \sum_{j=-l}^{h-1} c_{i,j}\, N_{i,k+1}^{\lambda}(x)\, N_{j,l+1}^{\mu}(y) \ . \tag{5.52}$$

Ter illustratie werden enkele tweedimensionale B-splines getekend in figuur 5.12.

Evaluatie en differentiatie van tensorproduct-splinefuncties gebeurt met een meerdimensionale uitbreiding van de gekende eendimensionale algoritmes. Net als bij de bespreking van meerdimensionale veelterm- en trigonometrische benadering, kunnen de meerdimensionale operaties ook hier via herhaalde eendimensionale operaties gebeuren. Veronderstel bijvoorbeeld dat we een spline $s(x, y)$ wensen te evalueren in een punt $(x, y) \in [\lambda_r, \lambda_{r+1}) \times [\mu_s, \mu_{s+1})$. Dan volgt dat (5.52) kan geschreven worden als

$$s(x, y) = \sum_{i=r-k}^{r} a_i\, N_{i,k+1}^{\lambda}(x) \ , \tag{5.53}$$



Figuur 5.12: Enkele tweedimensionale kubische B-splines.

met

$$a_i = \sum_{j=s-l}^{s} c_{i,j}\, N^{\mu}_{j,l+1}(y) \ , \qquad i = r-k, \ldots, r \ . \tag{5.54}$$

Dit kan gezien worden als de evaluatie van een eendimensionale spline in de variabele x, waarvan de coëfficiënten zelf berekend worden als waarden van k+1 eendimensionale splines in de variabele y. We kunnen dus k+1 'de Boor'-tabellen opstellen voor het berekenen van de a_i en dit laten volgen door één bijkomende tabel voor het berekenen van $s(x, y)$.

5.4.3 Periodieke splinefuncties

Periodieke splinefuncties worden gebruikt voor het benaderen en modelleren van gesloten krommen en gesloten oppervlakken. Eendimensionale periodieke splines over een interval $[a, b]$ werden in §5.2.1 gedefinieerd als splinefuncties die aan de randen van het interval voldoen aan de volgende periodiciteitsvoorwaarden:

$$s^{(j)}(a) = s^{(j)}(b) \quad \text{voor} \quad j = 0, \ldots, k-1 \ .$$

Deze k bijkomende beperkingen zorgen ervoor dat de dimensie van de splineruimte van graad k herleid wordt van dimensie $n+k$ tot dimensie n. We zullen verder veronderstellen dat $n \geq k$, zodat elke B-spline in hoogstens 1 van beide randpunten verschilt van 0.
Voor het construeren van een basis voor deze splineruimte, neemt men links en rechts van het interval $[a, b]$ telkens k bijkomende knooppunten, die aan volgende gelijkheden voldoen:

$$t_{-i} = t_{n-i} - (b-a) \quad \text{en} \quad t_{n+i} = t_i + (b-a) \quad i = 1, \ldots, k \ . \tag{5.55}$$

De ligging van de extra knooppunten wordt bepaald door de ligging van de inwendige knooppunten. We kunnen ze dus niet willekeurig kiezen. Uit die aanname volgt dat

$$N_{-i,k+1}(x) = N_{n-i,k+1}(x+b-a) \quad i = 1, \ldots, k \ . \tag{5.56}$$

Beide splinefuncties hebben precies dezelfde vorm, maar ze zijn onderling verschoven over een afstand gelijk aan de lengte van het interval. Hieruit volgt dat bovenstaande splines dezelfde functiewaarde en afgeleiden hebben in a resp. b, d.w.z.

$$N^{(j)}_{-i,k+1}(a) = N^{(j)}_{n-i,k+1}(b) \quad i = 1, \ldots, k \ , \ j = 0, \ldots, k-1 \ .$$

Een basis van n splinefuncties vinden we nu als volgt:

$$B_{i,k+1}(x) = \begin{cases} N_{i,k+1}(x) & i = 0, \ldots, n-k-1 \\ N_{i,k+1}(x) + N_{i-n,k+1}(x) & i = n-k, \ldots, n-1 \end{cases} \tag{5.57}$$

waarbij de functies steeds begrensd worden tot het interval $[a, b]$. De $B_{i,k+1}(x)$ zijn splinefuncties die aan de periodiciteitsvoorwaarden voldoen. Ze vormen een basis van de splineruimte. Dit wordt geïllustreerd in figuur 5.13, waar een basis werd getekend voor een ruimte van kubische periodieke splinefuncties.

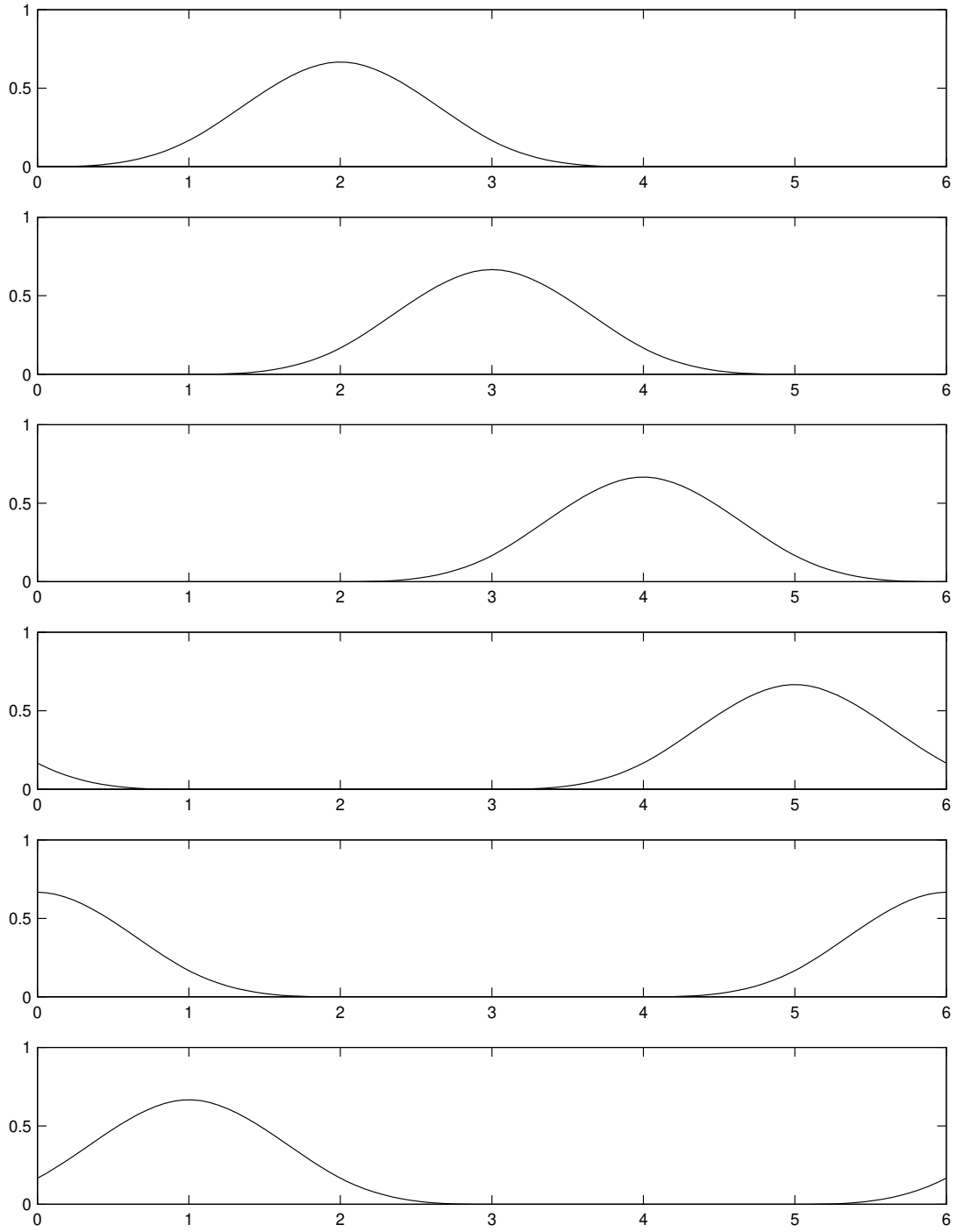
Een splinefunctie $s(x)$ zal periodiek zijn, wanneer ze kan geschreven worden als

$$s(x) = \sum_{i=0}^{n-1} c_i B_{i,k+1}(x) \ . \tag{5.58}$$

We kunnen de spline ook schrijven als een lineaire combinatie van de klassieke B-splines,

$$s(x) = \sum_{i=-k}^{n-1} d_i N_{i,k+1} \quad \text{met} \quad \begin{cases} d_i = c_i & i = 0, \ldots, n-k-1 \\ d_i = d_{i-n} = c_i & i = n-k, \ldots, n-1 \end{cases} \tag{5.59}$$

Van de $n + k$ splinecoëfficiënten kunnen er bijgevolg slechts n vrij worden gekozen.



Figuur 5.13: Een basis voor de ruimte van kubische periodieke splinefuncties gedefinieerd door de knooppuntenrij: 0, 1, 2, 3, 4, 5 en 6.

Hoofdstuk 6

Discrete benadering op basis van meetdata

Nu we het algemene kader geschetst hebben voor het benaderen van functies, en we wat dieper ingegaan zijn op het benaderen van *gekende* functies aan de hand van veeltermen en splines, kunnen we overgaan tot het benaderen van een discrete verzameling meetgegevens aan de hand van continue functies. We zullen dit zowel doen met veeltermen (Sectie 6.1) als met splines (Sectie 6.2). Vervolgens bekijken we specifiek hoe we de beschouwde technieken kunnen gebruiken voor de benadering van functies van twee onafhankelijke veranderlijken (Sectie 6.3).

6.1 Discrete veeltermbenadering

6.1.1 Een unitaire ruimte

In het *discrete benaderingsprobleem* beschouwen we discrete functies, d.w.z. functies gedefinieerd in een stel abscissen $a \leq x_1 < x_2 \ldots < x_{N-1} < x_N \leq b$. We kunnen de ruimte van discrete functies voorstellen door $l_2(N)$, waarbij N de dimensie van de ruimte aangeeft. Deze ruimte is voorzien van het discrete scalair product

$$(f, g) = \sum_{i=1}^{N} w_i \, f_i \, g_i \ , \tag{6.1}$$

met een gegeven, positieve discrete gewichtsfunctie,

$$w_i > 0 \quad \text{voor} \quad i = 1, \ldots, N \ . \tag{6.2}$$

We zullen ons in deze tekst beperken tot eindigdimensionale discrete ruimten. De ruimte $l_2(N)$ is natuurlijk isomorf met de vectorruimte $\mathbb{R}^N$ voorzien van een (gewogen) Euclidisch scalair product. De ruimte is ook isomorf met de ruimte $P_{N-1}[a, b]$, de ruimte van alle veeltermen van graad $N-1$ of lager. Met elke discrete functie van de ruimte $l_2(N)$ stemt juist één enkele veelterm overeen, namelijk de interpolerende veelterm $p(x)$ die voldoet aan

$$p(x_i) = f_i \quad \text{voor} \quad i = 1, \ldots, N \ .$$

Ook het omgekeerde geldt: met elke veelterm stemt een unieke discrete functie overeen, bekomen door de veelterm in het stel abscissen te evalueren. Uit deze discussie volgt dat we veeltermen tot en met graad $N-1$ mogen beschouwen als elementen van de discrete ruimte. Dit laat ons toe om te spreken over veeltermbenadering in de l_2-ruimte.

6.1.2 Opstellen van het normaalstelsel

Zij een stel punten (x_1, f_1), ..., (x_N, f_N) gegeven. De abscissen x_i worden foutloos verondersteld. De ordinaten f_i kunnen onderhevig zijn aan meetfouten, afrondingsfouten of andere fouten. Het op te lossen vraagstuk bestaat erin een veelterm te zoeken die zo goed mogelijk tussen de gegeven punten door loopt. Het criterium zegt dat we onder alle veeltermen van graad n die moeten nemen die de volgende uitdrukking minimaal maakt

$$\sum_{i=1}^{N} w_i (f_i - y_n(x_i))^2 \ . \tag{6.3}$$

Om de kleinste-kwadratenbenadering te bepalen kan men het normaalstelsel (3.47) opstellen, gebruikmakend van een of andere basis. Er zijn verschillende mogelijkheden.

- De eenvoudigste keuze is natuurlijk de monomiale basis $1, x, x^2, \ldots, x^n$. De coëfficiënten van de benaderende veelterm $y_n(x) = c_0 + c_1 x + c_2 x^2 + \ldots + c_n x^n$ worden gevonden als oplossing van een stelsel met een volle coëfficiëntenmatrix. Zoals reeds vermeld, zijn aan het gebruik van deze basis heel wat nadelen verbonden. Het stelsel zal in de meeste gevallen slecht geconditioneerd zijn. Het oplossen van het normaalstelsel vraagt veel rekenwerk en bij het veranderen van de graad moet een groot gedeelte van het rekenwerk worden hernomen.

- Men zou ook klassieke orthogonale veeltermen als basisfuncties kunnen nemen, zoals de Legendre-veeltermen. Dan schrijft men de benadering in de vorm

  $$y_n(x) = b_0 P_0(x) + b_1 P_1(x) + \ldots + b_n P_n(x) \ .$$

  De coëfficiënten van het normaalstelsel worden nu:

  $$g_{kl} = (P_k, P_l) = \sum_{i=1}^{N} w_i \, P_k(x_i) \, P_l(x_i) \ . \tag{6.4}$$

  De coëfficiëntenmatrix van het normaalstelsel is nog geen diagonaalmatrix, omdat we veeltermen gebruiken die orthogonaal zijn voor het continue scalair product, en niet voor het discrete scalair product. Wanneer de punten x_i tamelijk regelmatig over het benaderingsinterval $[-1, 1]$ verspreid liggen, dan zal de matrix echter wel sterk diagonaaldominant zijn. Het stelsel zal dan tamelijk goed geconditioneerd zijn.

- Om de coëfficiëntenmatrix werkelijk tot een diagonaalmatrix te herleiden, moet men als basis een stel veeltermen $p_k(x)$ nemen die orthogonaal zijn t.o.v. sommatie:

  $$\sum_{i=1}^{N} w_i \, p_k(x_i) \, p_l(x_i) = 0 \quad \text{voor} \quad l \neq k \ . \tag{6.5}$$

Men zoekt dan de benadering in de vorm

$$y_n(x) = a_0 p_0(x) + a_1 p_1(x) + \ldots + a_n p_n(x) \ . \tag{6.6}$$

Het principe van deze methode werd voorgesteld door Householder in 1953, verder uitgewerkt door Stiefel in 1955 en volledig op punt gesteld door Forsythe in 1957. De methode wordt bijna algemeen *de methode van Forsythe* genoemd.

6.1.3 Een overgedetermineerd stelsel

Men kan het benaderingsprobleem vervangen door de eis dat $y_n(x)$ zou voldoen aan

$$y_n(x_i) = f_i \quad \text{voor} \quad i = 1, \ldots, N \ .$$

Wanneer men $y_n(x)$ voorstelt in de monomiale basis, dan levert dit een overgedetermineerd stelsel in de onbekenden $c_0, c_1, \ldots, c_n$:

$$\left\{ \begin{array}{rcl} c_0 + c_1 x_1 + c_2 x_1^2 + ... + c_n x_1^n &=& f_1 \\ c_0 + c_1 x_2 + c_2 x_2^2 + ... + c_n x_2^n &=& f_2 \\ & \vdots & \\ c_0 + c_1 x_N + c_2 x_N^2 + ... + c_n x_N^n &=& f_N \end{array} \right. \tag{6.7}$$

We stellen dit stelsel verder voor als $Ac = f$. Een dergelijk stelsel kan slechts benaderend worden opgelost, bijvoorbeeld door de vector c te zoeken die de uitdrukking

$$\|Ac - f\|^2 \quad \text{of} \quad \|D(Ac - f)\|^2 \tag{6.8}$$

minimaal maakt. Hierin neemt men D doorgaans als een diagonaalmatrix met strikt positieve diagonaalelementen. Wanneer men als norm de klassieke Euclidische vectornorm neemt en wanneer men de diagonaalelementen van D schrijft als $\sqrt{w_i}$, dan ziet men in dat

$$\|D(Ac - f)\|^2 = \sum_{i=1}^{N} w_i \left(y_n(x_i) - f_i\right)^2 \ . \tag{6.9}$$

Het minimalisatieprobleem (6.8) is dan identiek aan het discrete kleinste-kwadratenprobleem (6.3). In de numerieke lineaire algebra wordt aangetoond dat de oplossing van (6.8) de oplossing is van het volgende stelsel

$$A^T W A c = A^T W f \quad \text{met} \quad W = D^2 \ . \tag{6.10}$$

Dit is niets anders dan het klassieke normaalstelsel uit de benaderingstheorie. We hebben in Sectie 2.4 al numerieke methodes bekeken om dit stelsel op te lossen.

6.1.4 Enkele praktische aspecten van de methode van Forsythe

6.1.4.1 Het bepalen van de rij van orthogonale veeltermen

Zij $p_0(x), p_1(x), p_2(x), \ldots$ een rij veeltermen van opeenvolgende graad die orthogonaal staan tot elkaar t.o.v. sommatie over het stel punten $x_1, \ldots, x_N$. Deze veeltermen voldoen aan de eigenschappen van §4.2, wanneer we het discrete scalaire product (6.1) gebruiken. Zo weet men bijv. dat elke veelterm $p_k(x)$ orthogonaal staat tot alle veeltermen van lagere graad. Het recursieschema wordt nu:

$$\begin{cases} p_{-1}(x) \equiv 0; \quad p_0(x) \equiv \lambda_0 \\ p_k(x) = \lambda_k \left((x - \alpha_k) p_{k-1}(x) - \beta_k p_{k-2}(x) \right) \quad \text{voor} \quad k = 1, 2, \ldots \end{cases} \tag{6.11}$$

$$\alpha_k = \frac{\sum_{i=1}^{N} w_i \, x_i \, p_{k-1}^2(x_i)}{\sum_{i=1}^{N} w_i \, p_{k-1}^2(x_i)} \quad \text{en} \quad \beta_k = \frac{\sum_{i=1}^{N} w_i \, x_i \, p_{k-1}(x_i) \, p_{k-2}(x_i)}{\sum_{i=1}^{N} w_i \, p_{k-2}^2(x_i)} \; . \tag{6.12}$$

Voor elke veelterm $p_k(x)$ vraagt dit het berekenen van 3 sommen: de noemer van β_k is immers ook de noemer van α_{k-1}, en moet dus niet opnieuw worden berekend.
De λ_k kunnen vrij gekozen worden om de veeltermen op een of andere wijze te normaliseren. Meestal neemt men alle veeltermen monisch, dus $\lambda_k = 1$.

6.1.4.2 Het bepalen van de benadering

De discrete benadering wordt volgens (4.7) gegeven door

$$y_n(x) = \sum_{k=0}^{n} a_k \, p_k(x) \quad \text{met} \quad a_k = \frac{\sum_{i=1}^{N} w_i \, p_k(x_i) \, f_i}{\sum_{i=1}^{N} w_i \, p_k^2(x_i)} \; . \tag{6.13}$$

De teller van a_k is nieuw; de noemer is deze van α_{k+1}. Per graad k moeten dus in totaal slechts drie sommen worden opgebouwd.
Voor grote k kunnen de $p_k(x_i)$-waarden tamelijk groot worden en van afwisselend teken, zodat fouten op de f_i zich sterk voortplanten. Men vervangt (6.13) daarom door

$$a_k = \frac{\sum_{i=1}^{N} w_i \, p_k(x_i) \left(f_i - \sum_{j=0}^{k-1} a_j p_j(x_i) \right)}{\sum_{i=1}^{N} w_i \, p_k^2(x_i)} = \frac{\sum_{i=1}^{N} w_i \, p_k(x_i) \, r_{k-1,i}}{\sum_{i=1}^{N} w_i \, p_k^2(x_i)} \; . \tag{6.14}$$

Hierin is $r_{k-1,i}$ het residu van de benadering van graad $k-1$. Deze formule is theoretisch identiek met (6.13), maar geeft in de praktijk betere resultaten. De invloed van afrondingsfouten wordt sterk verminderd omdat de $r_{k-1,i}$ afnemen met toenemende k.

6.1.4.3 Het bewaren van de orthogonale veeltermen

De orthogonale veeltermen $p_k(x)$ zou men kunnen schrijven in de vorm

$$p_k(x) = b_{k0} + b_{k1} x + \cdots + b_{kk} x^k \; . \tag{6.15}$$

Om de b_{kj} te bepalen kan men in de recursiebetrekking (6.11) uitdrukkingen van de vorm (6.15) substitueren, het rechterlid rangschikken naar opeenvolgende machten van x, en de coëfficiënten van gelijke machten in linker- en rechterlid aan elkaar gelijk stellen.
De formules voor α_k, β_k en a_k hebben echter enkel de *waarden* van de veeltermen nodig in de punten x_i. Dat zou het evalueren van de veeltermen in al deze punten vereisen, enorm veel rekenwerk vragen en numerieke moeilijkheden oproepen. Men kan zich immers aan grote coëfficiënten b_{kj} verwachten wat numerieke instabiliteit met zich meebrengt. Forsythe stelde daarom voor om niet de coëfficiënten b_{kj} te berekenen en te bewaren, maar wel de *waarde* van de veeltermen $p_k(x)$ *in de punten* x_i. Deze waarden $p_k(x_i)$ kunnen rechtstreeks uit de $p_{k-1}(x_i)$ en $p_{k-2}(x_i)$ worden berekend door de recursiebetrekking

$$p_k(x_i) = \lambda_k(x_i - \alpha_k)p_{k-1}(x_i) - \beta_k p_{k-2}(x_i) \tag{6.16}$$

toe te passen voor $i = 1, \ldots, N$.
Men bewaart in het geheugen de $p_{k-1}(x_i)$ en de $p_{k-2}(x_i)$. Zodra $p_k(x_i)$ berekend is, mag men $p_{k-2}(x_i)$ overschrijven, want die waarde heeft men dan niet meer nodig. De voordelen van deze methode zijn een belangrijke tijdsbesparing (vooral voor grote n) en een verbetering van de numerieke stabiliteit.

6.1.4.4 Keuze van de gewichten

De gewichten w_i kunnen worden bepaald op statistische gronden. Veronderstel dat men f_i in het punt x_i niet één keer zou meten maar m keer. Die meetwaarden zijn dan in feite toevallige waarden uit een statistische verdeling. Men verkrijgt dan een stel waarden $f_{i,k}$, met een gemiddelde f_i en spreiding s_i die gegeven worden door

$$f_i = \sum_{k=1}^{m} f_{i,k}/m \quad \text{en} \quad s_i = \sqrt{\sum_{k=1}^{m} (f_{i,k} - f_i)^2/(m-1)} \tag{6.17}$$

Men kan aantonen dat men een soort statistisch 'optimale' benadering bekomt indien

$$w_i = 1/s_i^2 \ . \tag{6.18}$$

Meestal moeten de s_i^2 niet worden berekend door het benaderingsprogramma, maar worden ze door de experimentator geleverd. Ze zijn dikwijls op voorhand gekend door het uittesten van de apparatuur. Wanneer men niet op de hoogte is van het statistisch gedrag van de meetapparatuur, dan zal men bijv. alle $w_i = 1$ nemen. Wenst men een meer gelijkmatig verloop van de fout over het interval, dan kan men bijv.

$$w_i = (1 - x_i^2)^{-1/2}$$

nemen. De punten moeten dan wel eerst naar het interval $[-1, 1]$ worden herleid.
In sommige toepassingen zijn de relatieve fouten belangrijk. Men wenst dan

$$\sum_{i=1}^{N} w_i \left(\frac{f_i - y_n(x_i)}{f_i} \right)^2 \tag{6.19}$$

te minimiseren. Dit kan gebeuren door een klassieke, absolute kleinste-kwadratenbenadering op te stellen gebruikmakend van de gewijzigde gewichten

$$w_i^\star = w_i / f_i^2 \ . \tag{6.20}$$

De methode gaat niet op als één of meerdere f_i nul zijn. Wenst men zo'n benadering op te stellen voor een functie met een nulpunt in het benaderingsinterval, dan moet men eisen dat de benadering ook de waarde nul aanneemt in de nulpunten van f.

6.2 Discrete benadering met splinefuncties

6.2.1 Probleemstelling

Zij gegeven een stel meetpunten $(x_r, f_r),\ r = 1, 2, \ldots, N$. We wensen een functie $y(x)$ te bepalen waarvan de grafiek min of meer goed aansluit bij deze gegevens. Als de vorm van $y(x)$ niet op voorhand vast ligt door de specifieke toepassing, zal men meestal voor een veelterm of een splinefunctie kiezen. Voor moeilijke problemen is men aangewezen op splines. Dat impliceert dan dat we de volgende parameters moeten bepalen:

- de graad van de splinefunctie,
- het aantal en de ligging van de knooppunten,
- de coëfficiënten in de gekozen voorstelling van de splinefunctie: bijv. de coëfficiënten van de B-splinebasisfuncties of van de afgeknotte-machtsfuncties.

De keuze van de graad stelt meestal geen problemen. Men zal in de praktijk bijna altijd werken met kubische splines. Enkel wanneer men ook geïnteresseerd is in hogere afgeleiden, zal men splines van een hogere graad (bijv. graad 5) moeten gebruiken.

6.2.2 Het kleinste-kwadratencriterium en het normaalstelsel

Voor de eenvoud zullen we hier veronderstellen dat de knooppunten t_j gegeven zijn. We kunnen dan $s(x)$ bepalen zodanig dat

$$\sum_{r=1}^{N} w_r \left(f_r - s(x_r)\right)^2 \tag{6.21}$$

minimaal is. De gewichten w_r worden opgegeven door de gebruiker. Hiermee kan die bijv. aangeven dat bepaalde meetpunten (x_r, f_r) nauwkeuriger zijn dan andere.
Gebruiken we de B-splinevoorstelling, dan komt het benaderingsprobleem neer op het oplossen van het normaalstelsel $G\,c = b$. Dat is een stelsel van dimensie $n + k$, met

$$g_{i,j} = \sum_{r=1}^{N} w_r N_{i,k+1}(x_r) N_{j,k+1}(x_r) \quad \text{en} \quad b_i = \sum_{r=1}^{N} w_r N_{i,k+1}(x_r)\, f_r \ . \tag{6.22}$$

De vector $c = (c_{-k}, c_{-k+1}, \ldots, c_{n-1})^T$ is de vector van de te zoeken coëfficiënten. De matrix van dit stelsel is symmetrisch en positief, maar niet altijd definiet. Men toont aan dat de matrix ook regulier is (en dus een SPD-matrix), als een geordende deelverzameling van de meetpunten aan de zogenaamde *Schoenberg-Whitney-voorwaarden* voldoet:

$$\exists\, \{u_{-k}, u_{-k+1}, \ldots, u_{n-1}\} \subset \{x_1, x_2, \ldots, x_N\}\ , \quad u_j < u_{j+1} \tag{6.23}$$

zodat

$$t_j < u_j < t_{j+k+1}\ , \quad j = -k, -k+1, \ldots, n-1\ . \tag{6.24}$$

Er dient dus minstens één meetpunt te liggen in het basisgebied van elke B-spline.
Bemerk dat de normaalmatrix een bandstructuur heeft. Het is gemakkelijk in te zien dat

$$g_{i,j} = 0 \quad \text{als} \quad |i-j| > k\ . \tag{6.25}$$

Het stelsel kan efficiënt worden opgelost met behulp van de eliminatiemethode van Cholesky voor symmetrische bandmatrices.

6.2.3 Een overgedetermineerd stelsel

Het normaalstelsel kan ook worden geschreven als volgt:

$$A^T W A\, c = A^T W f \quad \text{met} \quad W = D^2 = \text{diag}(w_1, \ldots, w_N)\ . \tag{6.26}$$

Hierin is A een rechthoekige matrix van dimensie $N \times (n+k)$ en vector $f = (f_1, \ldots, f_N)^T$. Dit is het normaalstelsel voor het *overgedetermineerd lineair stelsel* $DAc = Df$, of

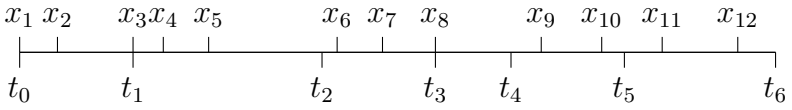
$$\sqrt{w_r} \sum_{i=-k}^{n-1} c_i\, N_{i,k+1}(x_r) = \sqrt{w_r}\, f_r\ , \quad r = 1, 2, \ldots, N\ . \tag{6.27}$$

We vermeldden in hoofdstuk 3 reeds dat de oplossing van (6.26) gelijk is aan de benaderende oplossing van (6.27), wanneer hiervoor de kleinste-kwadratenmethode uit de numerieke lineaire algebra wordt gebruikt. Een numeriek stabiele manier voor het oplossen van dit stelsel, zonder het expliciet construeren van de normaalmatrix, is gebaseerd op de QR-factorisatie van de rechthoekige matrix DA. Dat kan gebeuren met Givens-rotaties of Householder-transformaties. Ook hier kan men rekening houden met de speciale structuur van de matrix. Uit de eigenschappen van B-splines volgen immers onderstaande kenmerken.

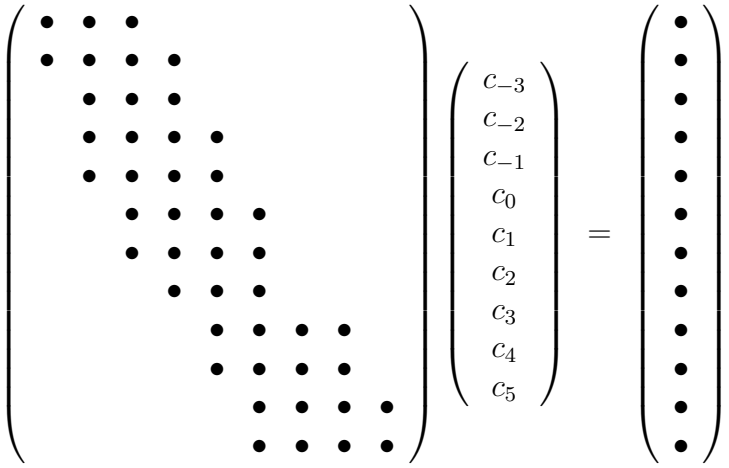
1. De r-de rij van matrix DA bevat slechts $k+1$ van nul verschillende elementen. De rij bevat slechts k elementen wanneer x_r samenvalt met een knooppunt. Deze niet-nul elementen staan in opeenvolgende kolommen in de rij van de matrix.

2. Indien de x_r-waarden in stijgende volgorde gerangschikt zijn, dan ligt het eerste van nul verschillende element van de r-de rij nooit meer naar links dan het eerste van nul verschillende element in rij $r-1$.

Voorbeeld 6.1 *We beschouwen een benaderingsprobleem met 12 meetpunten. Men koos 7 knooppunten en wenst te benaderen met kubische splines. De ligging van de abscissen x_r en van de knooppunten t_j is zoals aangeduid in onderstaande figuur.*



Ga na dat de structuur van het stelsel er als volgt uitziet:



Die speciale structuur maakt het gebruik van Givens-rotaties bijzonder aantrekkelijk. De methode van Givens laat toe de elementen onder de hoofddiagonaal van DA één na één te elimineren. De spaarsheid kan hierbij volledig worden uitgebuit. Na de QR-factorisatie vinden we een $N \times (n+k)$ matrix

$$R = \begin{pmatrix} \bar{R} \\ 0 \end{pmatrix}$$

waarbij $\bar{R}$ een $(n+k) \times (n+k)$ bovendriehoeksmatrix is met bovenbandbreedte $k+1$. Op elke rij zijn er hoogstens $k+1$ van nul verschillende elementen. Uit het stelsel

$$R\,c = Q^T D f, \tag{6.28}$$

waarvan de coëfficiëntematrix een driehoeksmatrix is, bepalen we dan gemakkelijk de gezochte kleinste-kwadratenoplossingsvector c.

OPMERKINGEN

– Men moet zelf het aantal knooppunten en de ligging ervan bepalen. Dat is niet altijd een eenvoudige opgave. Legt men te weinig knooppunten in bepaalde deelgebieden, dan zal de benadering daar onvoldoende aansluiten bij de meetgegevens. Legt men teveel knooppunten, dan zal de benadering teveel de fouten op de gegevens volgen.

  Het belang van de ligging van de knooppunten wordt grafisch geïllustreerd in figuur 6.1. Een dertigtal opgemeten punten werden er benaderd met een kubische spline, op een knooppuntenreeks met 4 inwendige knooppunten ($n = 5$). De knooppunten aan de randen werden telkens viervoudig genomen, d.w.z. $t_{-3} = t_{-2} = t_{-1} = t_0$ en $t_n = t_{n+1} = t_{n+2} = t_{n+3}$.

– Het feit dat men de knooppunten zelf moet kiezen, kan in sommige gevallen ook een voordeel inhouden. Immers, door het laten samenvallen van l knooppunten in een punt zal $s(x)$ ter hoogte van dat punt een discontinuïteit vertonen in de afgeleide van orde $k+1-l$. Dat kan interessant zijn indien de te benaderen functie een analoge discontinuïteit heeft.

– Normaal gezien zal het vereiste aantal knooppunten heel wat kleiner zijn dan het aantal opgegeven meetpunten. Dat is interessant wat rekentijd en gegevensreductie betreft.

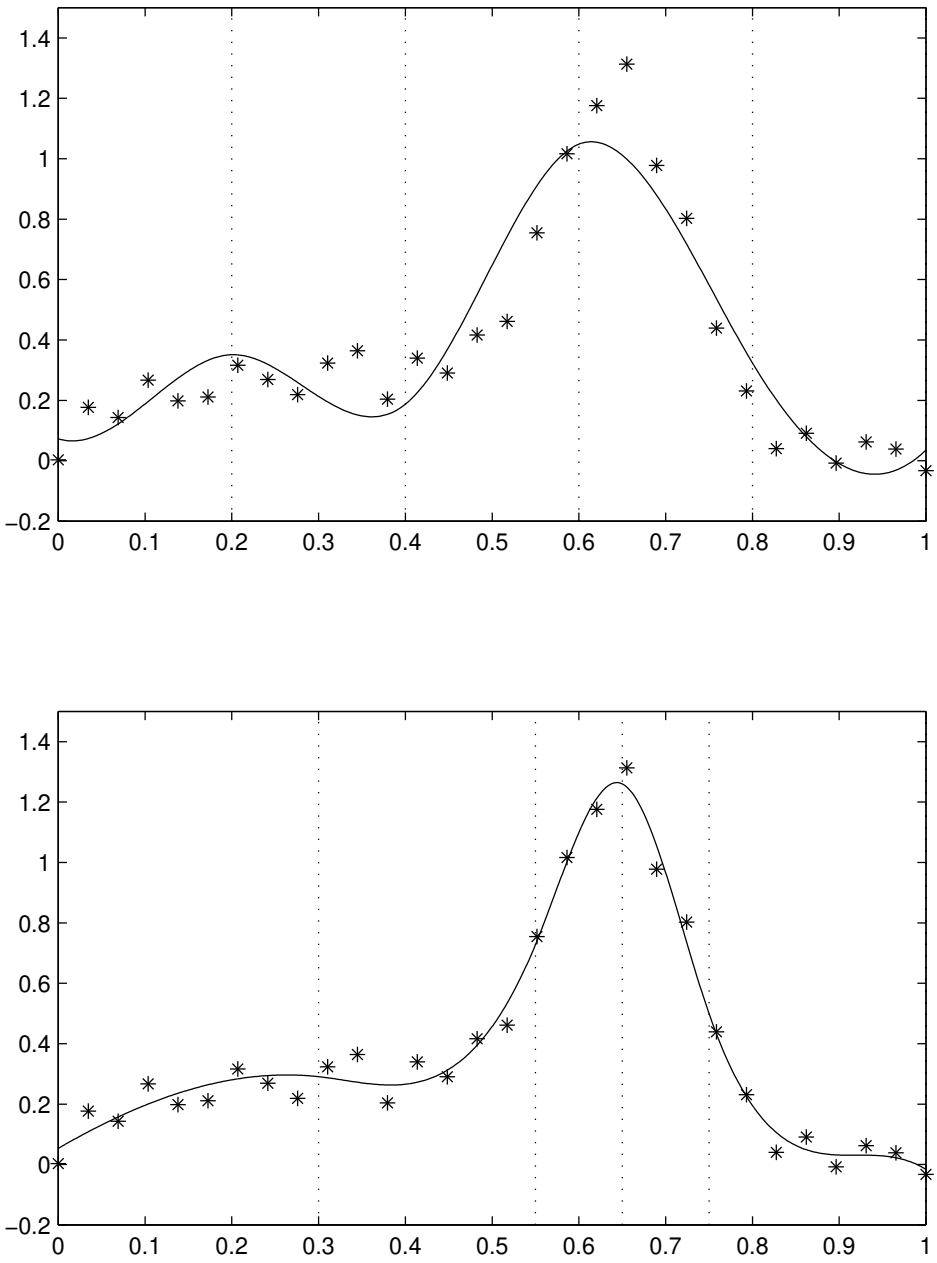
– Een aantal kleinste-kwadratenprogramma's trachten ook de ligging van de knooppunten automatisch te bepalen. Men neemt dan een vast aantal knooppunten en beschouwt in (6.21) naast de coëfficiënten ook de inwendige knooppunten t_j als veranderlijken.

  Dergelijke programma's vragen veel rekentijd, en succes is niet altijd gewaarborgd. Het kleinste-kwadratenprobleem is nu immers een niet-lineair optimalisatieprobleem. Men vindt dat de te minimaliseren functionaal heel wat lokale extrema en zadelpunten heeft. Men heeft dus goede startwaarden nodig voor $t_1, t_2, \ldots t_{n-1}$. Bovendien is het een minimalisatieprobleem met beperkingen. Er moet immers steeds gelden dat $t_0 \leq t_1 \leq t_2 \leq \ldots \leq t_n$. Voor verdere informatie over dit onderwerp verwijzen we naar [Die95, Ch. 4].

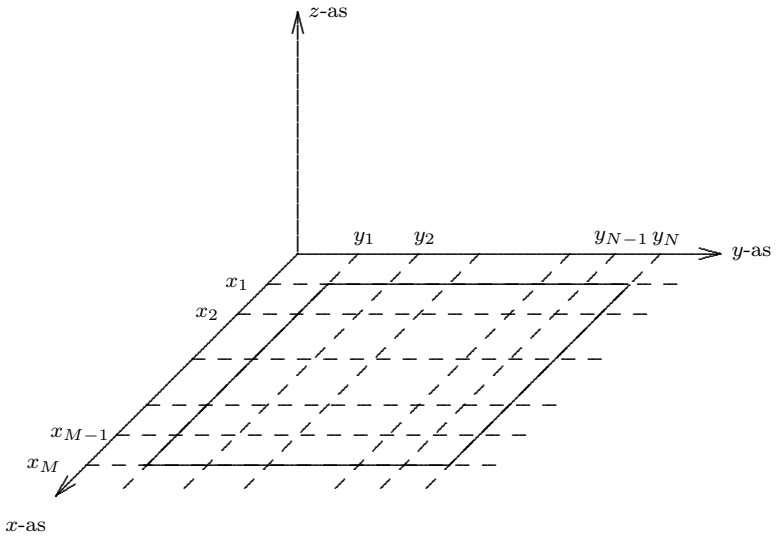
6.3 Discrete benadering in twee ruimtelijke dimensies

6.3.1 Het benaderingsprobleem

We beperken ons in deze tekst tot het tweedimensionale benaderingsprobleem, d.w.z. tot het benaderen van oppervlakken. We beschouwen daarenboven enkel het meest eenvoudige geval: dit waarbij de functiewaarden gegeven zijn op een regelmatig rechthoekig puntenrooster. Dit rooster bestaat uit een stel punten (x_i, y_j) met $i = 1, \ldots, M$ en $j = 1, \ldots, N$, gelegen op de snijpunten van rechten evenwijdig met de coördinaatassen, zie figuur 6.2.



Figuur 6.1: Benaderen met kubische splinefuncties. De ligging van de inwendige knooppunten wordt aangegeven door een verticale stippellijn: equidistant (bovenaan) en niet-equidistant (onderaan).



Figuur 6.2: Een regelmatig rechthoekig puntenrooster.

In de roosterpunten zijn functiewaarden f_{ij} of $f(x_i, y_j)$ gegeven. Men vraagt een veeltermbenadering op te stellen in de veranderlijken x en y,

$$z(x,y) = \sum_{k=0}^{m} \sum_{l=0}^{n} c_{k,l}\, x^k y^l \ , \tag{6.29}$$

volgens het discrete kleinste-kwadratencriterium, d.w.z. zodanig dat

$$\sum_{i=1}^{M} \sum_{j=1}^{N} w_{i,j} \left(f_{i,j} - z(x_i, y_j)\right)^2 \tag{6.30}$$

minimaal wordt. We zullen voor de eenvoud verder veronderstellen dat de componenten van de gewichtsfunctie kunnen worden geschreven als

$$w_{i,j} = w_i^{(1)}\, w_j^{(2)} \ .$$

Een dergelijke gewichtsfunctie noemt men *separabel*. Wanneer we aan meerdimensionale *continue* benadering zouden doen, dan zou een separabele gewichtsfunctie $w(x, y)$ kunnen worden geschreven als $w^{(1)}(x)w^{(2)}(y)$.
Dit vraagstuk is in wezen niet echt verschillend van het eendimensionale benaderingsprobleem. Laten we veronderstellen dat de ruimte van de veeltermen van graad m in x en graad n in y opgespannen wordt door een stel basisveeltermen $\phi_{k,l}(x, y)$, met $k = 0, \ldots, m$ en $l = 0, \ldots, n$. We zouden bijvoorbeeld $\phi_{k,l}(x, y) = x^k y^l$ kunnen nemen. De coëfficiënten worden samengevoegd tot een vector met $(n + 1) \cdot (m + 1)$ componenten,

$$c = \left(c_{0,0}, c_{0,1}, \ldots, c_{0,n}, c_{1,0}, c_{1,1}, \ldots, c_{1,n}, \ldots, c_{m,0}, c_{m,1}, \ldots, c_{m,n}\right) \ .$$

Het normaalstelsel $Gc=b$ is dan een stelsel met $(n+1)\cdot(m+1)$ vergelijkingen en onbekenden, waarbij de componenten van G en b gegeven worden door

$$G_{u(n+1)+v+1,k(n+1)+l+1} = (\phi_{k,l},\phi_{u,v}) = \sum_{i=1}^{M}\sum_{j=1}^{N} w_{i,j}\,\phi_{k,l}(x_i,y_j)\,\phi_{u,v}(x_i,y_j) \tag{6.31}$$

$$b_{u(n+1)+v+1} = (f,\phi_{u,v}) = \sum_{i=1}^{M}\sum_{j=1}^{N} w_{i,j}\,f_{i,j}\,\phi_{u,v}(x_i,y_j) \tag{6.32}$$

Wanneer we de klassieke basis nemen, dan mogen we opnieuw verwachten dat het normaalstelsel bijzonder slecht geconditioneerd zal zijn. Daarom zouden we graag over veeltermen beschikken die orthogonaal zijn t.o.v. sommatie over alle gegeven punten. Bij gebruik van zo'n veeltermen heeft het normaalstelsel bovendien een diagonale coëfficiëntenmatrix.

6.3.2 Gebruik van orthogonale veeltermen in twee veranderlijken

De gezochte orthogonale veeltermen moeten voldoen aan

$$\sum_{i=1}^{M}\sum_{j=1}^{N} w_{i,j}\,\phi_{k,l}(x_i,y_j)\phi_{u,v}(x_i,y_j) = 0\ , \quad \text{tenzij} \quad k=u \quad \text{en} \quad l=v\ . \tag{6.33}$$

Men ziet gemakkelijk in dat het product van orthogonale veeltermen in de twee veranderlijken afzonderlijk een orthogonale veelterm $\phi_{k,l}(x,y)$ in x en y oplevert. Inderdaad, zij $\phi_{k,l}(x,y)$ een tweedimensionale veelterm van graad k in x en graad l in y, van de vorm

$$\phi_{k,l}(x,y) = p_k(x)\,q_l(y) \tag{6.34}$$

waarbij de factoren voldoen aan

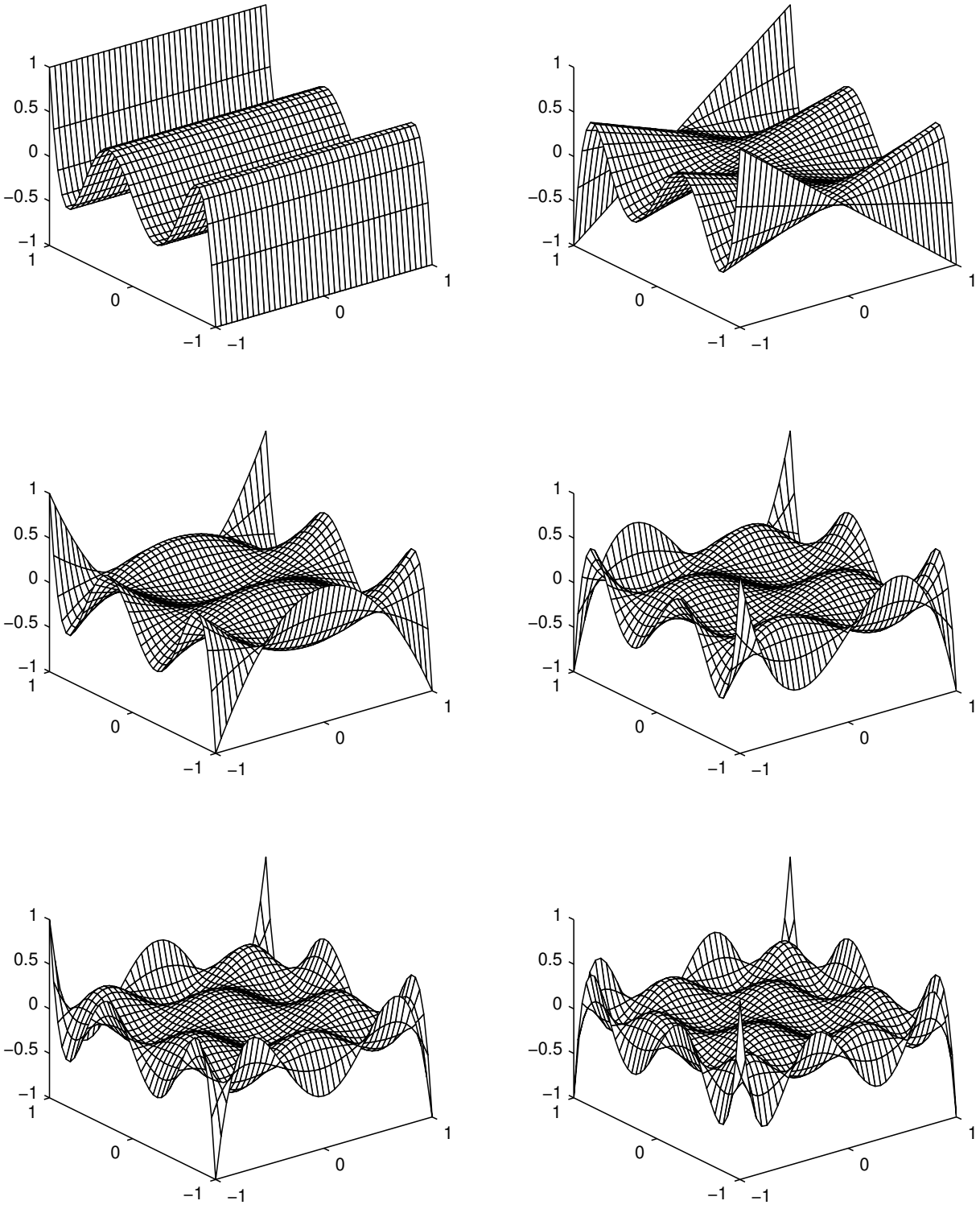
$$\sum_{i=1}^{M} w_i^{(1)}\,p_k(x_i)\,p_u(x_i) = 0 \ \text{ als } \ k\neq u \quad \text{en} \quad \sum_{j=1}^{N} w_j^{(2)}\,q_l(y_j)\,q_v(y_j) = 0 \ \text{ als } \ l\neq v\ .$$

Gebruikmakend van het separabel zijn van de gewichtsfunctie vinden we dat

$$\sum_{i=1}^{M}\sum_{j=1}^{N} w_{i,j}\,\phi_{k,l}(x_i,y_j)\,\phi_{u,v}(x_i,y_j) = \sum_{i=1}^{M} w_i^{(1)}\,p_k(x_i)\,p_u(x_i)\ \cdot \sum_{j=1}^{N} w_j^{(2)}\,q_l(y_j)\,q_v(y_j)\ .$$

Deze uitdrukking neemt de waarde nul aan, tenzij $k=u$ en $l=v$. Een aantal orthogonale veeltermen in twee veranderlijken van de vorm (6.34) werden getekend in figuur 6.3.
Vanwege de bijzondere vorm van de orthogonale veeltermen zal blijken dat het benaderen over een rechthoekig rooster veel eenvoudiger wordt dan het benaderen over een willekeurig stel punten. De benadering wordt nu in de vorm

$$z(x,y) = \sum_{k=0}^{m}\sum_{l=0}^{n} a_{k,l}\,p_k(x)\,q_l(y) \tag{6.35}$$



Figuur 6.3: Discrete orthogonale veeltermen in de veranderlijken x en y. Getekend werden de veeltermen $\phi_{k,5}(x, y)$, voor $k = 0, 1, \ldots 5$ (linksboven tot rechtsonder).

geschreven. De coëfficiënten $a_{k,l}$ worden gegeven door

$$\begin{aligned} a_{k,l} &= \frac{\sum_{i=1}^{M}\sum_{j=1}^{N} w_i^{(1)}\, w_j^{(2)}\, f_{i,j}\, p_k(x_i)\, q_l(y_j)}{\sum_{i=1}^{M}\sum_{j=1}^{N} w_i^{(1)}\, w_j^{(2)}\, p_k^2(x_i)\, q_l^2(y_j)} \\ &= \frac{\sum_{i=1}^{M}\sum_{j=1}^{N} w_i^{(1)}\, w_j^{(2)}\, f_{i,j}\, p_k(x_i)\, q_l(y_j)}{\|p_k\|^2\ \|q_l\|^2} \end{aligned} \tag{6.36}$$

De veeltermen $p_k(x)$ en $q_l(y)$ kunnen worden gevonden met de methode van Forsythe.

6.3.3 Herhaalde eendimensionale benadering

De kost van het uitrekenen van de $(m+1)\cdot(n+1)$ dubbele sommen van $M \cdot N$ getalwaarden voor het bepalen van de coëfficiënten $a_{k,l}$ kan sterk worden gereduceerd. Formule (6.36) kan worden geschreven als

$$a_{k,l} = \sum_{j=1}^{N} w_j^{(2)} \left(\frac{\sum_{i=1}^{M} w_i^{(1)}\, f_{i,j}\, p_k(x_i)}{\|p_k\|^2} \right) \frac{q_l(y_j)}{\|q_l\|^2} \tag{6.37}$$

$$= \frac{\sum_{j=1}^{N} w_j^{(2)}\, b_{k,j}\, q_l(y_j)}{\|q_l\|^2} \quad \text{met} \quad b_{k,j} = \frac{\sum_{i=1}^{M} w_i^{(1)}\, f_{i,j}\, p_k(x_i)}{\|p_k\|^2}\ . \tag{6.38}$$

De berekening kan nu geformuleerd worden als een proces in twee stappen.
Eerst berekent men voor elke j de benadering voor de M punten $(x_1, y_j), (x_2, y_j), \ldots, (x_M, y_j)$. Men bepaalt dus de kleinste-kwadratenbenadering voor een doorsnede van het te benaderen oppervlak met een vlak loodrecht op de y-as. Dit levert N benaderingen van de vorm

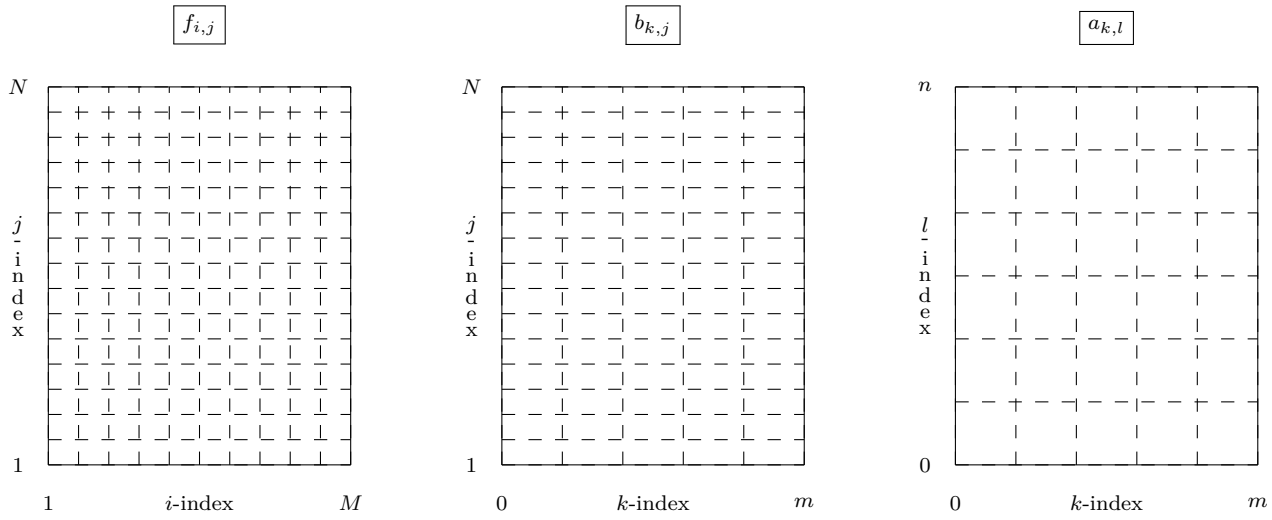
$$g_j(x) = \sum_{k=0}^{m} b_{k,j}\, p_k(x) \quad \text{met} \quad b_{k,j} = \frac{\sum_{i=1}^{M} w_i^{(1)}\, f_{i,j}\, p_k(x_i)}{\|\, p_k(x)\,\|^2}\ .$$

Daarna stelt men de N coëfficiënten $b_{k,1}, b_{k,2}, \ldots, b_{k,N}$ op een rij. Zo heeft men $m+1$ rijen, die kunnen gezien worden als rijen van functiewaarden bij de punten $y_1, y_2, \ldots, y_N$. Elk van deze rijen gaat men benaderen door een veelterm $h_k(y)$ van graad n:

$$h_k(y) = \sum_{l=0}^{n} a_{k,l}\, q_l(y) \quad \text{met} \quad a_{k,l} = \frac{\sum_{j=1}^{N} w_j^{(2)}\, b_{k,j}\, q_l(y_j)}{\|\, q_l\,\|^2}\ .$$

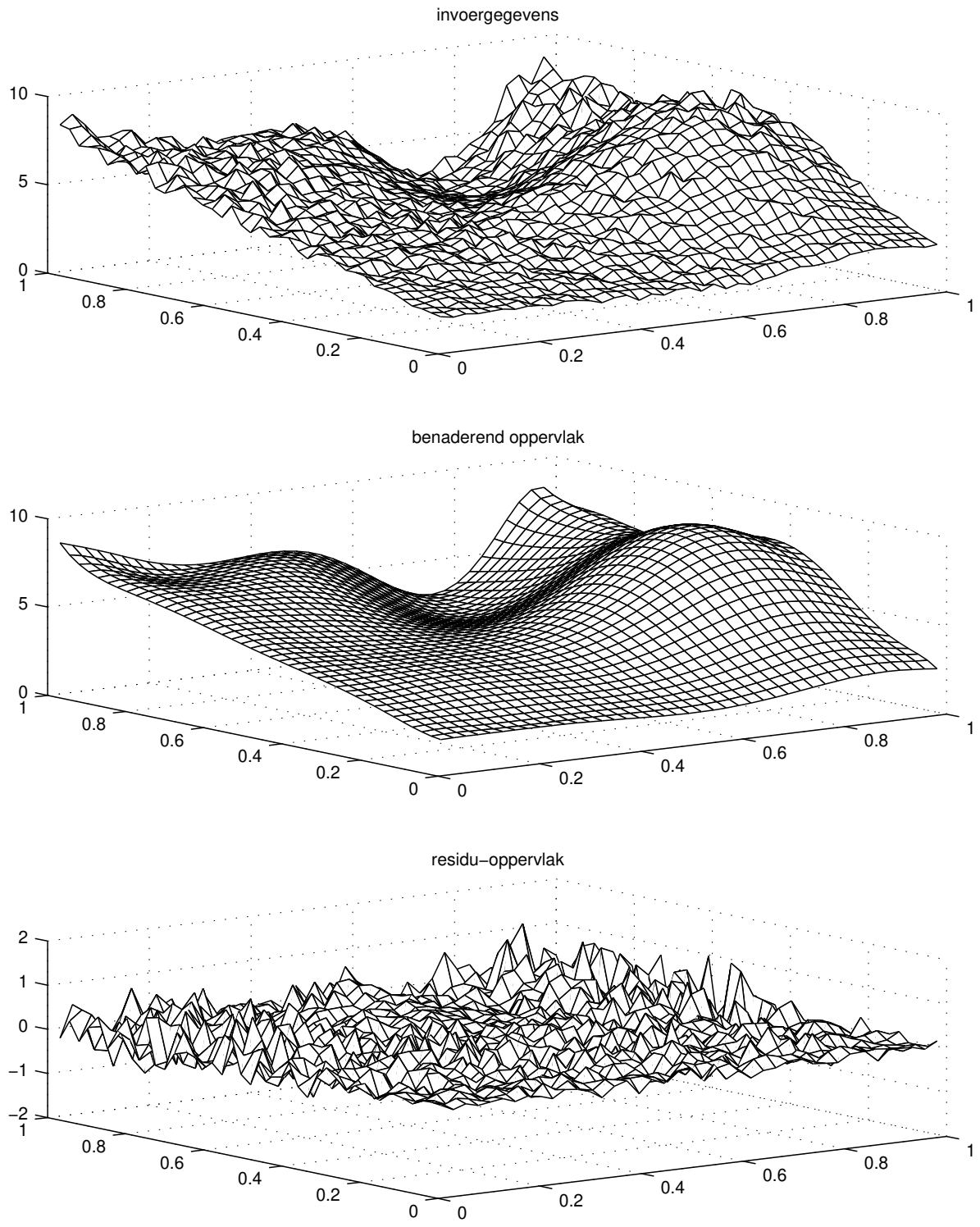
Deze werkwijze kan worden opgevat als een herhaalde eendimensionale benadering.

- Men berekent eerst N kleinste-kwadratenbenaderingen van graad m voor discrete functies van lengte M, langs lijnen evenwijdig met de x-as.
- Daarna berekent men $m+1$ kleinste-kwadratenbenaderingen van graad n voor discrete functies van lengte N langs lijnen evenwijdig met de y-as.



Figuur 6.4: Gegevensstructuren voor het tweedimensionale benaderingsalgoritme.

Men hoeft dus geen volledig nieuw programma te schrijven wanneer men reeds het algoritme van Forsythe heeft geïmplementeerd. Men kan dit laatste twee keer na elkaar toepassen: eerst voor het berekenen van alle $b_{k,j}$, dan voor de $a_{k,l}$. Dit wordt geïllustreerd in figuur 6.4. Men kan verder nog heel wat werk besparen door de discrete orthogonale veeltermen slechts éénmaal uit te rekenen. Ook het evalueren van de benadering (6.35) kan efficiënt gebeuren met een algoritme gebaseerd op herhaalde eendimensionale evaluatie. Ga zelf na hoe een dergelijk algoritme er zou uitzien, gebruikmakend van het evaluatie-algoritme van Smith. Ook de uitbreiding naar hogerdimensionale benadering ligt voor de hand. Tot slot geven we een voorbeeld van een tweedimensionale benadering in figuur 6.5.



Figuur 6.5: Tweedimensionale benadering: invoergegevens (41×41 datapunten), benaderend oppervlak van graad 7 in x en y (8×8 coëfficiënten), en residu van de benadering.

Hoofdstuk 7

Regularisatietechnieken

7.1 Inleiding en motivatie

In voorgaande hoofdstukken hebben we een aantal aspecten van benaderingstheorie behandeld, zowel in eindige dimensie (benaderen van vectoren) als in oneindige dimensie (benaderen van functies). Bij functiebenadering vertrokken we ofwel van een gegeven functie $f(x)$ (continue benadering), ofwel van een aantal meetpunten $(x_i, f_i)_{i=1}^N$ (discrete benadering). We bleven echter op de vlakte over *waarmee* we de gegeven functie of data zouden benaderen. We kozen voor veeltermen of voor splines, maar de graad van de veeltermen, of de plaats van de knopen en de graad van de splines, beschouwden we als gegeven.

We formuleerden bovenstaand soort problemen heel algemeen. We beschouwden de datavector, of de te benaderen functie, als een punt in een vectorruimte, die we voorzagen van een afstand, norm, of (liefst zelfs) een scalair product, en definieerden zelf een deelruimte waarin we een benadering wensten te vinden. Vervolgens formuleerden we in algemene termen het probleem als een beste-benaderingsprobleem: zoek het element van de deelruimte dat het "dichtst" ligt bij de te benaderen functie of data.

Wanneer we een scalair product ter beschikking hebben, konden we die beste benadering vervolgens berekenen door het opstellen van het normaalstelsel, dat uitdrukt dat de fout van onze benadering orthogonaal moet staan op de deelruimte waarin we benaderen. We beschouwen de functie die we willen benaderen als *gekend*, en associëren er een scalair product mee. In het geval we vertrekken van discrete data (die we interpreteren als functie-evaluaties), associëren we met de meetpunten $(x_i, f_i)_{i=1}^N$ een gegeven discrete gewichtsfunctie w_i, $i = 1, \ldots, N$ en het discrete scalair product

$$(f, g) = \sum_{i=1}^{N} w_i f_i g_i. \tag{7.1}$$

We zoeken dan een benadering

$$y_n = \sum_{k=0}^{n} c_k \phi(x)^k \tag{7.2}$$

waarbij we de coëfficiënten $c = [c_0 \dots c_n]^T$ zo kiezen dat ze de uitdrukking

$$E(c) = \sum_{i=1}^{N} w_i (f_i - y_n(x_i))^2. \tag{7.3}$$

minimaliseren. We hebben $E(c)$ de kostfunctie genoemd, en de coëfficiënten

$$c^* = \arg\min_c E(c), \tag{7.4}$$

die de beste benadering definiëren worden bekomen door het opstellen van het normaalstelsel. Zie vorig hoofdstuk, waar we het ook hadden over de keuze van de gewichten om de nauwkeurigheid van de metingen in rekening te brengen.

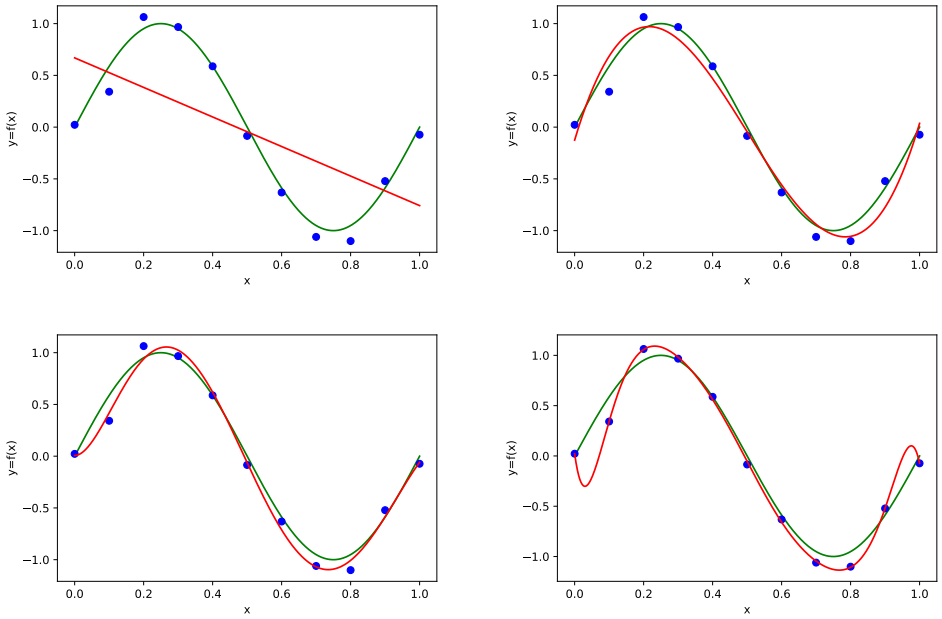
Wanneer we data willen benaderen, moeten we echter nog een stap verder gaan. Idealiter moeten we ook de deelruimte identificeren waarin we de beste benadering willen zoeken. Dit hoofdstuk beschouwt één manier om die vraag te omzeilen: regularisatie. Dit hoofdstuk is als volgt opgebouwd. We beschouwen *overfitting* als een belangrijk risico bij het benaderen van discrete data met continue functies. In Sectie 7.2 bekijken we hoe overfitting ontstaat en hoe het zich manifesteert. Vervolgens bekijken we, in Sectie 7.3, hoe we kunnen nagaan (*detecteren*) of er van overfitting sprake is. Daarna, in Sectie 7.4, proberen we overfitting te vermijden, en we zullen vaststellen dat regularisatie een oplossing kan bieden. De analyse zal leiden tot twee soorten regularisatie: Tikhonovregularisatie (in Sectie 7.5), wat weinig aanpassing vraagt aan het normaalstelsel, en LASSO-regularisatie (in Sectie 7.6), waarvoor fundamenteel andere algoritmes nodig zijn.

7.2 Het probleem van overfitting

In deze sectie bekijken we hoe overfitting ontstaat in een situatie waarbij we de onderliggende continue functie kennen waar we een benadering voor willen opstellen. We veronderstellen dat we een *ongekende functie* $y = f(x)$ wensen te benaderen op basis van data $(x_i, f_i)_{i=1}^N$ en gewichten w_i, $i = 1, \dots, N$, volgens het kleinste-kwadratencriterium. We veronderstellen dat de data $f_i = f(x_i) + \eta_i$, met η_i een normaal verdeelde meetfout met gemiddelde 0 en standaardafwijking σ.

7.2.1 Benadering van ruizige data door veeltermen

We beschouwen als te benaderen functie $y = f(x) = \sin(2\pi x)$, op het interval $x \in [0, 1]$. We veronderstellen dat we $f(x)$ niet kennen, en wensen te benaderen op basis van $N = 11$ metingen in de punten $x_i = i\Delta x$, $i = 0, \dots, N$ met $\Delta x = 1 \cdot 10^{-1}$. Als ruisgrootte kiezen we $\sigma = 1 \cdot 10^{-1}$. We benaderen de ruizige data volgens het kleinste-kwadratencriterium, achtereenvolgens met veeltermen van graden 1, 3, 6 en 9. De resultaten staan op Figuur 7.1.



Figuur 7.1: Kleinste-kwadratenbenadering (rode lijn) van ruizige data (blauwe punten), bekomen door evaluatie van de functie $y = \sin(2\pi x)$ (groene lijn) en toevoeging van normaal verdeelde meetfout met standaardafwijking $\sigma = 1 \cdot 10^{-1}$. Graden zijn achtereenvolgens $n = 1$ (linksboven), $n = 3$ (rechtsboven), $n = 6$ (linksonder) en $n = 9$ (rechtsonder). Er is een duidelijke tendens van onderfitting voor te lage graad naar overfitting voor te hoge graad.

We maken de volgende vaststellingen:

- Voor een te lage graad is de benaderende veelterm geen goede benadering van de data, en ook niet van de onderliggende functie die de data genereerde.
- Voor hoger wordende graden sluit de benaderende veelterm initieel steeds beter aan bij zowel de metingen als de onderliggende functie.
- Voor de hoogste graad sluit de benaderende veelterm zeer sterk aan bij de opgemeten data, maar minder sterk bij de onderliggende functie. Er is sprake van overfitting.

Wanneer we de graad nog verder zouden laten stijgen, zou het aantal te bepalen coëfficiënten gelijk wordt aan het aantal datapunten. Van zodra dat gebeurt, stellen we in feite een interpolerende veelterm op. De conclusie is dat de benaderende veeltermen van hoge graad weinig 'beschrijvend' zijn voor de onderliggende functie. Dit effect noemt men overfitting: hoewel de meetdata zeer goed benaderd worden, is het opgesteld model geen goede benadering van de onderliggende functie. Doordat zowel de benaderingen van graad $n = 3$ als $n = 6$ er 'behoorlijk' uitzien, is het echter moeilijk om op zicht te besluiten welke benadering de beste is voor ons doel.

7.3 Detecteren van overfitting: generalisatiefout

In bovenstaande voorbeeld was de overfitting met het blote oog vast te stellen. Dat is echter in het algemeen niet mogelijk, en dit om drie (evidente) redenen:

- Het visueel vaststellen van overfitting vereist dat we de onderliggende functie kunnen *evalueren* in veel meer punten dan diegene die we gebruikt hebben om de benadering op te stellen. Dat is zeker niet altijd mogelijk, en bovendien kan je je dan de vraag stellen of we die extra data dan niet beter meenemen bij het berekenen van de beste benadering.

- Het visueel vaststellen van overfitting vereist dat we onze benadering kunnen *grafisch voorstellen*, wat niet mogelijk is in hogere dimensies.

- Het is onmogelijk om een dergelijke visuele inspectie aan de computer over te laten en dus te automatiseren.

Deze redenen kunnen geïllustreerd worden door het volgende benaderingsprobleem te beschouwen. Beschouw als data een aantal foto's, voorgesteld als een rij pixels. De data is dus een lange vector $x \in \mathbb{R}^D$, met D de dimensie van de vector. De benaderende functie die we willen opstellen is een functie $\mathbb{R}^D \to \mathbb{R} : x \mapsto y = f(x)$ die elke foto zou moeten afbeelden op een getal tussen 0 en 1, een getal dat de kans voorstelt dat op de foto een kat te zien is. De beschikbare data hiervoor is een verzameling foto's x_i met voor elke foto i de bijhorende functiewaarde $f_i = 1$ als een mens op die foto een kat gezien heeft, en $f_i = 0$ anders. Dit soort functies zijn zowel niet makkelijk te evalueren als niet makkelijk te visualiseren.

Het probleem van automatiseren zonder visuele inspectie is op te lossen door een evalueerbaar criterium te definiëren dat kwantificeert hoe groot de fout op de onderliggende functie is. Hiervoor zullen we echter extra functie-evaluaties onmogelijk kunnen vermijden (omdat het net de fout op die extra functie-evaluaties is die we willen kennen). Wanneer een functie goed benaderd wordt in punten die niet gebruikt zijn voor het opstellen van de benadering, zeggen we dat de benadering goed *veralgemeent* (of *generalizeert*). We zullen dit opnieuw doen met een kleinste-kwadratencriterium, maar nu met data die niet gebruikt is voor het opstellen van de benadering.

Herinner je dat we onze benadering opstelden op basis van datapunten $(x_i, f_i)_{i=1}^N$. Voortaan, en in analogie met de terminologie die gehanteerd wordt in *machine learning*, zullen we deze data de *trainingsdata* noemen. Om na te gaan of onze benadering goed generaliseert, zullen we een tweede dataset introduceren, $(\tilde{x}_i, \tilde{f}_i)_{i=1}^{\tilde{N}}$, die we *testdata* zullen noemen. De fout die we met onze kleinste-kwadratenbenadering minimaliseren,

$$E_{\text{training}} = \sum_{i=1}^{N} w_i \left(f_i - y_n(x_i)\right)^2, \tag{7.5}$$

noemen we dan de *trainingsfout*, terwijl we de fout op de testdata

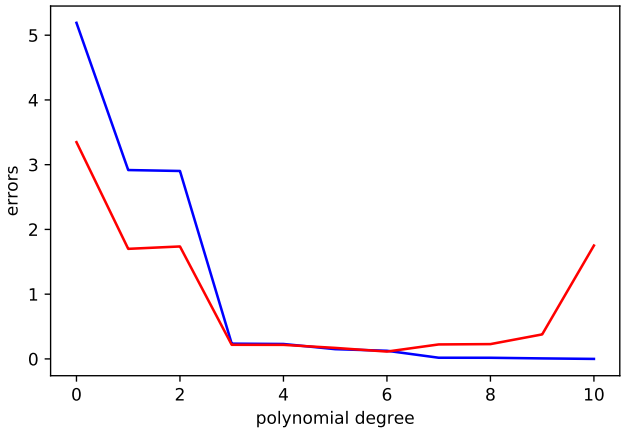
$$E_{\text{test}} = \sum_{i=1}^{\tilde{N}} \tilde{w}_i \left(\tilde{f}_i - y_n(\tilde{x}_i) \right)^2, \tag{7.6}$$

de *generalizatiefout* noemen.

De vraag die we ons nu moeten stellen is: vanaf wanneer is er sprake van overfitting? Wat we tot nu toe weten is het volgende:

- Wanneer we de graad van de benadering laten toenemen, zal de trainingsfout monotoon dalen, omwille van de eigenschappen van beste benaderingen in groeiende deelruimtes.
- Wanneer we de graad van de benadering laten toenemen, verwachten we dat de generalizatiefout plots terug zal beginnen stijgen.

Het lijkt dan ook redelijk om die benadering te nemen die de generalisatiefout minimaliseert. We bekijken dit even voor het voorbeeld uit de vorige sectie. We berekenen achtereenvolgens de beste veeltermbenadering op basis van de data tot en met graad $n = 10$. Voor de testdata nemen we $\tilde{N} = 11$ willekeurige punten in het interval $[0, 1]$, waarbij we op dezelfde manier ruis toevoegen als bij de trainingsdata. Figuur 7.2 toont dat de testfout terug begint te stijgen vanaf graad $n = 6$, ook al blijft de trainingsfout monotoon dalen.



Figuur 7.2: Trainingsfout (blauw) en generalisatiefout (rood) van een kleinste-kwadratenbenadering op basis van ruizige data (blauwe punten), bekomen door evaluatie van de functie $y = \sin(2\pi x)$ (groene lijn) en toevoeging van normaal verdeelde meetfout met standaardafwijking $\sigma = 1 \cdot 10^{-1}$, als functie van de graad van de benadering.

Opmerking 7.1 (Bias-variance trade-off) *De kleinste-kwadratenfout die bekomen wordt bij training is een deterministische grootheid: het gaat om het verschil tussen de waarde van*

de benaderende functie en de meting in de meetpunten. Om die reden wordt de trainingsfout soms een vertekening *genoemd (Engels:* bias*). De generalisatiefout wordt berekend met behulp van testdata* in willekeurig gekozen extra meetpunten*, en is dus een toevalsvariabele. De generalisatiefout is dus een monster (Engels:* sample*) uit een verdeling van mogelijke generalisatiefouten. Bij overfitting kunnen we hevige schommelingen verwachten in de generalisatiefout als functie van het meetpunt. De grootheid* $E_{\texttt{test}}$ *meet dan ook de variantie op die generalisatiefouten. Dit is de reden dat de keuze voor de graad van benadering in de machine learning gemeenschap vaak de* bias-variance trade-off *genoemd wordt.*

Opmerking 7.2 (Alternatieven voor validatie) *Het genereren van een grote hoeveelheid testdata*[1] *voelt ergens aan als een verspilling. Er is dan ook heel wat onderzoek gedaan (en bezig) naar alternatieve manieren. Eén manier is de zogenaamde "leave-one-out" kruisvalidatie. Hierbij worden meerdere benaderingen opgesteld, waarbij telkens één enkel datapunt niet wordt meegenomen. Vervolgens wordt gekeken hoe die benaderingen veralgemenen naar het genegeerde punt. Dit principe gedetailleerd wiskundig uitwerken leidt ons (helaas) te ver.*

7.4 Overfitting vermijden: principe van regularisatie

Het principe van regularisatie vertrekt vanuit een eenvoudige vaststelling: een model met meer vrijheidsgraden (bijvoorbeeld een hogere-graadsveelterm) kan ook modellen met minder vrijheidsgraden voorstellen (bijvoorbeeld lagere-graadsveeltermen) door de coëfficiënten die horen bij de hoogste-graadstermen expliciet op nul te zetten. Dit is een directe oplossing voor het probleem van overfitting, want het probleem blijft om te kiezen welke termen we meenemen en welke niet. Alleen brengt deze manier van kijken ons wel op een idee: wat als we zouden proberen een evenwicht te vinden tussen de "beste" benadering, en een benadering die de coëfficiënten van de benadering klein houdt?

Om overfitting te vermijden, wijzigen we dus de kostfunctie van het benaderingsprobleem door een extra term toe te voegen die de grootte van de coëfficiënten van de veeltermbenadering penaliseert. We schrijven opnieuw de benadering

$$y_n(x) = \sum_{k=0}^{n} c_k \phi_k(x), \tag{7.7}$$

met $\{\phi_k\}_{k=1}^n$ de (veelterm-)basis en $\boldsymbol{c} = [c_0, \ldots, c_n]$ de vector van coëfficiënten. We noemen E_{reg} de geregulariseerde kostfunctie, en definiëren ze als volgt:

$$\begin{aligned} E_{\text{reg}}(c) &= \sum_{i=1}^{N} w_i (f_i - y_n(x_i))^2 + \lambda F(\boldsymbol{c}) \\ &= \|D(Ac - f)\|^2 + \lambda F(\boldsymbol{c}), \end{aligned} \tag{7.8}$$

[1] Of, alternatief, het niet gebruiken van alle beschikbare data voor training.

waarbij λ een zogenaamde hyperparameter (methode-parameter) is die ons in staat stelt de invloed van de regularisatie op de kostfunctie te regelen.

Op dit moment kunnen we drie zaken vaststellen:

- We hebben een parameter λ ingevoerd die nog moet gekozen worden. Hoe groter λ, hoe harder grote coëfficiënten worden bestraft. Indien $\lambda = 0$, dan vallen we terug op de beste benadering die we voordien hadden, en speelt de regularisatieterm geen rol meer. Indien λ naar oneindig gaat, dan convergeert de oplossing naar $c \to 0$. Elke positieve waarde voor λ zal tot resultaat hebben dat de coëfficiënten krimpen ten opzichte van de coëfficiënten van de beste benadering zonder de regularisatieterm. Een "goede" waarde voor λ zit daar tussenin, en kan gevonden worden door de generalisatiefout te vergelijken van oplossingen die bekomen worden met verschillende waarden voor λ.

- We hebben de functie $F(\boldsymbol{c})$ nog niet bepaald. Indien we de 2-norm kiezen,

$$F(\boldsymbol{c}) = \|\boldsymbol{c}\|_2^2 = \sum_{k=0}^{n} c_k^2, \tag{7.9}$$

  dan bekomen we (een vereenvoudigde versie van) Tikhonovregularisatie, ook *ridge regression* genoemd in de machine learning literatuur. Indien we de 1-norm kiezen,

$$F(\boldsymbol{c}) = \|\boldsymbol{c}\|_1 = \sum_{k=0}^{n} |c_k|\,, \tag{7.10}$$

  dan bekomen we LASSO-regularisatie[2]. Tikhonovregularisatie komt aan bod in Sectie 7.5, en LASSO-regularisatie in Sectie 7.6.

- We zagen in voorgaande hoofdstukken dat de beste benadering in een bepaalde benaderingsruimte (bv., de ruimte van veeltermen van graad n) uniek is, ongeacht welke basis er gekozen wordt om de elementen van die ruimte voor te stellen[3] Indien we regularisatie gebruiken, echter, heeft de keuze van de basis een effect in de benadering die je berekent: de basis waarin het benaderingsprobleem wordt opgelost bepaalt de grootte van de coëfficiënten $\boldsymbol{c}$: dezelfde veelterm in verschillende basissen leiden tot verschillende waarden voor $F(\boldsymbol{c})$. De oplossing van het regularisatieprobleem kan dus afhangen van de gekozen basis.

[2]LASSO staat voor *least absolute shrinkage and selection operator*, en het lijkt er wel een beetje op dat de uitvinder, Robert Tibshirani, wel wat moeite deed om een aangenaam acroniem te vinden.

[3]Uiteraard heeft de keuze van basis een invloed op numerieke eigenschappen van de methode, en dus de nauwkeurigheid waarmee de beste benadering gevonden wordt, maar niet op de *definitie* van de beste benadering. Bijvoorbeeld, de monomiaalbasis leidt tot een Grammatrix met een groot conditiegetal, terwijl Chebychev veeltermen typisch leiden tot een Grammatrix met lager conditiegetal.

7.5 Tikhonovregularisatie of L_2-regularisatie

We gaan op zoek naar de beste benadering, met begrensde coëfficiënten, waarbij we de grootte van de coëfficiënten meten in de 2-norm. Hiervoor wijzigen we de kostfunctie

$$E_{\text{reg}}(c) = \|D(Ac - f)\|^2 + \lambda\|c\|^2, \tag{7.11}$$

waarbij λ een te kiezen hyperparameter (methodeparameter) is en waarbij D een diagonaalmatrix met elementen $(\sqrt{w_i})_{i=1}^N$. We kunnen de kostfunctie (7.11) ook als volgt schrijven (met $W = D^2$):

$$\begin{aligned} E_{\text{reg}}(c) &= (Ac - f)^T W(Ac - f) + \lambda c^T c \\ &= c^T(A^T WA + \lambda I)c - f^T WAc - c^T A^T Wf \\ &= c^T(A^T WA + \lambda I)c - 2f^T WAc + f^T Wf, \end{aligned} \tag{7.12}$$

waarbij we in de laatste stap gebruikten dat $f^T WAc = (f, Ac) = (Ac, f) = c^T A^T Wf$ voor reële f, A en c. Deze doelfunctie wordt minimaal in $c = c^*$ als

$$\nabla_c E_{\text{reg}}(c^*) = 0. \tag{7.13}$$

We passen dus de productregel toe:

$$\begin{aligned} \nabla_c E_{\text{reg}}(c) &= \nabla_c \left(c^T(A^T WA + \lambda I)c - 2f^T WAc\right), \\ &= \nabla_c \left(c^T(A^T WA + \lambda I)c\right) - 2f^T WA, \\ &= \nabla_c \left(c^T(A^T WA + \lambda I)\right) c + c^T(A^T WA + \lambda I)\nabla_c c - 2f^T WA \\ &= 2c^T(A^T WA + \lambda I) - 2f^T WA. \end{aligned}$$

Als optimaliteitsvoorwaarde bekomen we dus een lineair stelsel voor de coëfficiënten c^* (zie ook vergelijking (6.10))

$$(A^T WA + \lambda I)c^* = A^T Wf \tag{7.14}$$

Wanneer we het bovenstaande lineaire stelsel vergelijken met datgene wat we bekwamen bij het opstellen van het normaalstelsel voor het vinden van de beste benadering is, blijkt dat slechts één iets is veranderd ten opzichte van de discrete benaderingstechniek in (6.10): bij L2-regularisatie, is een extra term toegevoegd op de diagonaal. Dit heeft een belangrijk praktisch voordeel: de technieken voor het oplossen van de lineaire stelsels die we gebruikten voor het oplossen van het normaalstelsel kunnen ook hier gebruikt worden.

Opmerking 7.3 (Tihkonovregularisatie) *De kostfunctie van l_2-regularisatie* (7.11) *stelt ons in staat om de grootte van de coëficiënten te penaliseren, maar de methode geeft hetzelfde gewicht aan alle parameters. Men kan verschillende parameters op een verschillende manier penaliseren door aan de penalisatie van de coëfficiënten gewichten toe te kennen:*

$$E_{reg,\,Tikh} = \|D(Ac - f)\|^2 + \|\Gamma c\|_2^2 \tag{7.15}$$

waarbij Γ *een diagonaalmatrix is met gewichten* $(\Gamma_k)_{k=0}^n$. *Een nadeel van deze methode is het feit dat we meer hyperparameters dienen te selecteren: in plaats van een enkele parameter* λ *dienen we nu een diagonaalmatrix* Γ *te kiezen. In feite is deze veralgemeende vorm de echte Tikhonovregularisatie. De vereenvoudigde versie met een scalaire hyperparameter heet strikt genomen "ridge regression", maar de termen worden vaak door elkaar gebruikt.*

7.6 LASSO-regularisatie of L_1-regularisatie

Hoewel Tikhonovregularisatie en ridge regressie er in slagen om overfitting te vermijden (of te verminderen), wijkt hun gedrag toch wel wat af van wat we in Sectie 7.4 beoogden, met name het op nul zetten van overbodige coëfficiënten. De reden daarvoor is dat de penalisatiefunctie $F(c)$ de neiging zal hebben om *alle* coëfficiënten klein te houden, eerder dan *enkele* coëfficiënten *exact nul* te maken. Een voorname reden voor dit gedrag van Tikhonovregularisatie ligt in het kwadratische karakter van de penalisatiefunctie, dat grote coëfficiënten zwaarder bestraft dan lage coëfficiënten.

In deze sectie stellen we een alternatief voor op basis van de 1-norm, om dit euvel te proberen te verhelpen. Het doel van l_1-regularisatie is om vaker een aantal coëfficiënten exact nul te krijgen bij regularisatie. Met dit doel definiëren we de volgende doelfunctie

$$\begin{aligned} E_{\text{reg},l_1} &= \|D(Ac-f)\|_2^2 + \lambda\|c\|_1 \\ &= \sum_{i=1}^{N} w_i(f_i - y_n(x_i))^2 + \lambda\sum_{k=0}^{n}|c_k| \end{aligned} \tag{7.16}$$

Belangrijk om op te merken is dat de optimaliteitscriteria in dit geval niet langer aanleiding geven tot een lineair stelsel, en dat we dus alternatieve methodes voor optimalisatie zullen nodig hebben.

7.7 Regularisatie als optimalisatie met beperkingen

Om in te zien hoe LASSO-regularisatie verschilt van Tikhonovregularisatie maken we een omweg langs de definitie van optimalisatieproblemen met ongelijkheidsbeperkingen. Veronderstel dat we een optimalisatieprobleem[4] met beperkingen hebben van de volgende vorm:

$$\min_c E(c) = \sum_{i=1}^{N} w_i(f_i - y_n(x_i))^2, \tag{7.17}$$

$$\text{s.t.} \qquad F(c) \leq K. \tag{7.18}$$

De oplossing van dit optimalisatieprobleem geeft een set coëfficiënten c^* die zodanig is dat ze, binnen de toegestane set van coëfficiënten, de doelfunctie minimaliseert. Beschouw

[4] "s.t." is een afkorting van "subject to" en betekent zoveel als "onderhevig aan". De tweede lijn definieert dus de beperking.

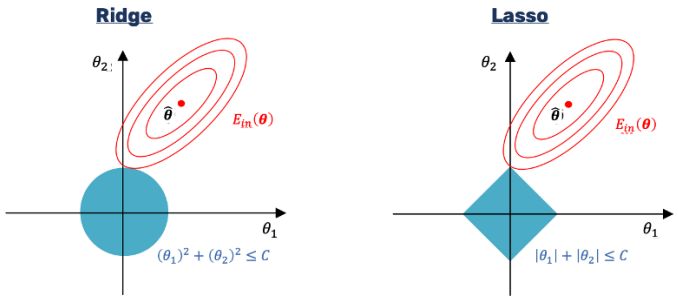
eerst de beste benadering c^*_{kkb}, die de oplossing is van het optimalisatieprobleem *zonder* de beperking. Die zal ook de oplossing zijn van het optimalisatieprobleem mét beperking, op voorwaarde dat K groot genoeg is, met name $K > F(c^*_{\text{kkb}})$. Pas wanneer de beste benadering niet langer de oplossing is van bovenstaand optimalisatieprobleem, speelt de beperking een rol.

We gaan er in deze sectie daarom van uit dat de beperking $F(c) = K$ een gelijkheidsbeperking is geworden. (We noemen de beperking dan actief.) In dat geval weten we dat we de oplossing kunnen vinden door de beperking met een Lagrangevermenigvuldiger toe te voegen aan de doelfunctie en het resulterende optimalisatieprobleem zonder beperkingen op te lossen:

$$\min_c E(c) = \sum_{i=1}^{N} w_i(f_i - y_n(x_i))^2 + \lambda(F(c) - K), \tag{7.19}$$

wat op een constante na gelijk is aan het regularisatieprobleem (7.8). Er is natuurlijk een belangrijk verschil: bij het regularisatieprobleem (7.8) ligt de regularisatieparameter λ vast. Bij het optimalisatieprobleem met beperking (7.17) wordt de waarde van de Lagrangevermenigvuldiger λ bepaald bij het oplossen ervan, en hangt de waarde ervan af van de waarde van K. Dit betekent dat de oplossing van het regularisatieprobleem (7.8) voor een gekozen waarde van de regularisatieparameter λ overeenkomt met de oplossing van het optimalisatieprobleem met beperkingen *voor een passende waarde van* K, die volgt uit de gekozen waarde van λ.

Het bovenstaande betekent ook dat de oplossing van het regularisatieprobleem en die van het optimalisatieprobleem met beperking dezelfde is. Dat laat ons toe om grafisch een intuïtie te verwerven voor de sterkere neiging van LASSO-regularisatie om meer coëfficiënten exact nul te maken. Dit is te zien in Figuur 7.3. (Deze figuur is overgenomen vanop deze link[5].)



Figuur 7.3: Schets van het regularisatieprobleem als een optimalisatieprobleem met beperkingen. Links: ridge regressie; rechts: LASSO. Beide figuren tonen een eenheidsbol voor de parameters, en contourlijnen van de kleinste-kwadraten-doelfunctie. Parameters c in de tekst zijn genoteerd als θ op de figuur.

[5] https://cal.unibg.it/wp-content/uploads/DSI/slide/Lecture-04-Overfitting-and-regularization.pdf

Deel II

Data, grafen en eigenwaarden

Hoofdstuk 8

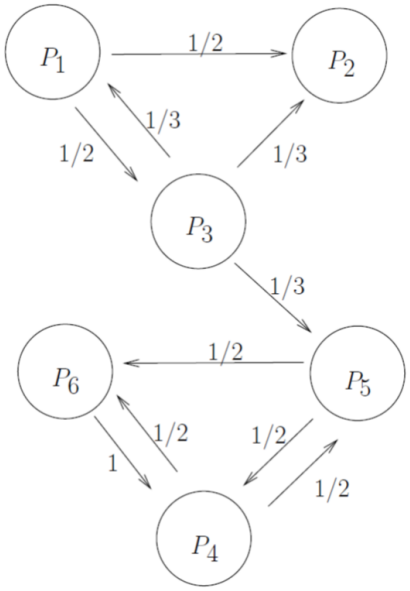
Grafen en eigenwaarden in datawetenschappen

We bespreken enkele toepassingen van (numerieke) lineaire algebra, en in het bijzonder het berekenen van eigenwaarden, in de context van netwerken en grafen. De eerste toepassing betref het zogenaamde *PageRank* algoritme dat de basis vormt van (de eerste versie van) Google's zoekmachine. Zoals we zullen zien bestaat de kern van het algoritme uit het berekenen van de eigenvector horende bij de dominante eigenwaarde van een matrix. De tweede toepassing betreft het berekenen van de meest centrale knoop in een netwerk, wat opgelost kan worden door eerst een eigenwaardenontbinding te bepalen. De derde toepassing betreft het partitioneren van een graaf, waarbij de eigenvector horende bij de tweede-kleinste eigenwaarde een centrale rol speelt.

8.1 PageRank

Het PageRank algoritme is gebaseerd op het netwerk dat ontstaat doordat webpagina's naar elkaar verwijzen via hyperlinks. Concreet brengen robot crawlers dit netwerk in kaart en indexeren ze de gevonden webpagina's. Vervolgens wordt er op basis van de structuur van het netwerk op regelmatige basis een score berekend en toegekend aan elke webpagina. Wanneer een gebruiker van de zoekmachine termen ingeeft, worden op basis daarvan webpagina's gesuggereerd waar die termen in voorkomen, gesorteerd volgens de toegekende score aan de webpagina's.
Wat de eerste stap betreft worden in feite een graaf en geassocieerde matrix $A \in \mathbb{R}^{N\times N}$ opgesteld die de relaties tussen de N bezochte websites uitdrukken. Elke knoop stelt een webpagina voor. De graaf is gericht en bevat een pijl van knoop i naar knoop j als er op pagina P_i een hyperlink staat die verwijst naar pagina P_j. Met N_i het aantal links op pagina P_i, kunnen we matrix A definiëren op de volgende manier: $A_{ij} = 1/N_i$ als er een link is van P_i naar P_j en $A_{ij} = 0$ anders. Matrix A komt dus overeen met een gewogen verbindingsmatrix van de graaf.

Voorbeeld 8.1 *We beschouwen het didactische voorbeeld uit [LM05], met* 6 *webpagina's waarvoor de overeenkomstige graaf wordt weergegeven in Figuur 8.1. Matrix A wordt in*



Figuur 8.1: Graaf gebaseerd op hyperlinks op 6 pagina's. De geassocieerde matrix A wordt gegeven door (8.1).

dit geval gegeven door

$$A = \begin{bmatrix} 0 & 1/2 & 1/2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 1/3 & 1/3 & 0 & 0 & 1/3 & 0 \\ 0 & 0 & 0 & 0 & 1/2 & 1/2 \\ 0 & 0 & 0 & 1/2 & 0 & 1/2 \\ 0 & 0 & 0 & 1 & 0 & 0 \end{bmatrix}. \tag{8.1}$$

Merk dat deze matrix ook een interpretatie heeft in termen van een Markov model voor het internet. De getallen $1/2$ *op de eerste rij bijvoorbeeld drukken dan uit dat een crawler of gebruiker, die van pagina naar pagina navigeert door op links te klikken zonder persoonlijke voorkeur, met* 50% *kans op pagina* 2 *en met* 50% *kans op pagina* 3 *terecht komt.*

Gebaseerd op matrix A wordt aan elke webpagina P_i een score $r(P_i) \geq 0$ toegekend, voor $i = 1, \ldots, N$, waarbij $r(\cdot)$ staat voor ranking. Daarbij worden de volgende principes gehanteerd:

- als veel andere pagina's naar een bepaalde pagina verwijzen begunstigt dit de score van deze laatste;
- als een verwijzende pagina zelf een hoge score heeft is dat begunstigend voor een hoge score van een pagina waarnaar verwezen wordt;
- als een verwijzende pagina weinig links heeft is dat begunstigend voor een hoge score van de pagina waarnaar verwezen wordt.

Hiermee consistent vertrekt men van volgende wiskundige formulering,

$$r(P_i) \approx \sum_{j \in B_i} \frac{r(P_j)}{N_j}, \quad i = 1, \ldots, N, \tag{8.2}$$

met B_i de verzameling indices van pagina's die verwijzen naar P_i. Mits we Π definiëren als de vector van scores of de zgn. *PageRank vector*,

$$\Pi = [r(P_1)\ r(P_2)\ \cdots\ r(P_N)]^T,$$

kunnen we uitdrukkingen (8.2) formuleren in temen van matrix A:

$$\Pi^T \approx \Pi^T A$$

of

$$A^T \Pi \approx \Pi. \tag{8.3}$$

In (8.2) en (8.3) gebruiken we het symbool voor benadering ($\approx$) in plaats van gelijkheid ($=$), omdat bij een gelijkheid de resulterende vergelijkingen mogelijk inconsistent zijn: immers $r(P_i)$ wordt gedefinieerd in functie van $r(P_j),\ j \neq i$, en vice-versa. Om inconsistenties te vermijden, en tegelijkertijd de weg te openen naar een snel algoritme om een goede oplossing van (8.3) te vinden, worden kleine aanpassingen aan matrix A gemaakt die we beschrijven in wat volgt.
Vooreerst wordt matrix $\tilde{A}$ bekomen uit A door elke nul-rij te vervangen door een rij met alle elementen gelijk aan $1/N$. Voor grote N heeft dit weinig effect op het rechterlid van (8.2), door de weging met $1/N$. De grafische interpretatie is dat men aan een pagina zonder links virtuele links (pijlen in de graaf) toevoegt naar alle andere pagina's. Doordat in matrix $\tilde{A}$ alle rijsommen gelijk zijn aan 1 en alle elementen groter of gelijk zijn aan nul behoort deze tot de klasse van *rij-stochastische* matrices. Vervolgens bepaalt men matrix $\hat{A}$ als een convexe combinatie van $\tilde{A}$ en een matrix die op elke positie getal $1/N$ bevat,

$$\hat{A} = (1-\alpha)\tilde{A} + \alpha\ \left(\frac{1}{N}\mathbf{1}\,\mathbf{1}^T\right),$$

waarbij $\mathbf{1} = [1\ 1\ \cdots\ 1]^T$ en $\alpha \in (0,1)$ dicht bij nul gekozen wordt. De tweede term heeft de interpretatie van het toevoegen van virtuele links met een klein gewicht tussen gelijk welke pagina's. Door deze tweede aanpassing krijgt matrix $\hat{A}$, die rij-stochastisch blijft, de bijkomende eigenschap *irreduceerbaar* te zijn. Dit laatste betekent dat de matrix d.m.v. een permutatie niet tot een blok-driehoeksmatrix met meer dan 1 diagonaal blok omgevormd kan worden of, equivalent hiermee, dat de geassocieerde graaf sterk geconnecteerd is. "Sterk geconnecteerd"wil zeggen dat we vanuit elke knoop elke andere knoop van de graaf kunnen bereiken, rekening houdend met de richting van de pijlen. Het belang van de twee bovenvermelde aanpassingen komt tot uiting in de volgende stelling.

Stelling 8.1.1
Als $\hat{A}$ een irreduceerbare rij-stochastische matrix is, dan is de dominante eigenwaarde enkelvoudig en gelijk aan 1. *Bovendien kan de overeenkomstige eigenvector van $\hat{A}^T$ steeds gekozen worden zodat al haar elementen groter of gelijk zijn aan nul.*

In het PageRank algoritme wordt tenslotte (8.3) vervangen door

$$\hat{A}^T \Pi = 1 \; \Pi. \tag{8.4}$$

Gebaseerd op Stelling 8.1.1 en gebruik makend van de eigenschap dat de eigenwaarden van $\hat{A}$ en $\hat{A}^T$ aan elkaar gelijk zijn, kunnen we besluiten dat probleem (8.4) goed gedefinieerd is (immers (8.4) drukt uit dat de vector met scores Π een eigenvector is horende bij eigenwaarde 1 waarvan het bestaan gegarandeerd wordt door Stelling 8.1.1). Bovendien hebben de elementen van Π hetzelfde teken, wat een ordening toelaat.
Het oplossen van (8.4) komt neer op het bepalen van de eigenvector horende bij de *dominante* eigenwaarde van $\hat{A}$, waarvoor (varianten van) de *methode van de machten* gebruikt worden. In bepaalde literatuur wordt naar Google's eigenwaardenprobleem (8.4) verwezen als "het grootste eigenwaardeprobleem ter wereld".

Opmerking 8.1 *In de interpretatie van $\hat{A}$ als transitiematrix horende bij een Markov keten stelt de gezochte vector Π, genormaliseerd met de 1-norm, een stationaire distributie voor. Dit is een kansverdeling om in een bepaalde knoop te zitten, die invariant is in de (discrete) tijd.*

Voorbeeld 8.2 *We komen terug op het didactische voorbeeld gevisualiseerd in Figuur 8.1. De aanpassingen van A resulteren in*

$$\tilde{A} = \begin{bmatrix} 0 & 1/2 & 1/2 & 0 & 0 & 0 \\ 1/6 & 1/6 & 1/6 & 1/6 & 1/6 & 1/6 \\ 1/3 & 1/3 & 0 & 0 & 1/3 & 0 \\ 0 & 0 & 0 & 0 & 1/2 & 1/2 \\ 0 & 0 & 0 & 1/2 & 0 & 1/2 \\ 0 & 0 & 0 & 1 & 0 & 0 \end{bmatrix}$$

en voor $\alpha = 0.1$,

$$\hat{A} = \frac{9}{10}\tilde{A} + \frac{1}{10}\frac{1}{6}\mathbf{1}\,\mathbf{1}^T = \begin{bmatrix} 1/60 & 7/15 & 7/15 & 1/60 & 1/60 & 1/60 \\ 1/6 & 1/6 & 1/6 & 1/6 & 1/6 & 1/6 \\ 19/60 & 19/60 & 1/60 & 1/60 & 19/60 & 1/60 \\ 1/60 & 1/60 & 1/60 & 1/60 & 7/15 & 7/15 \\ 1/60 & 1/60 & 1/60 & 7/15 & 1/60 & 7/15 \\ 1/60 & 1/60 & 1/60 & 11/12 & 1/60 & 1/60 \end{bmatrix}.$$

De genormaliseerde dominante eigenvector van $\hat{A}$, de PageRank vector, is dan

$$\Pi = \begin{bmatrix} 0.037 \\ 0.054 \\ 0.042 \\ 0.375 \\ 0.206 \\ 0.286 \end{bmatrix}.$$

Stel nu dat een gebruiker van de zoekmachine 2 *termen intypt die voorkomen in volgende webpagina's:*
term 1: pag. 1, pag. 4, pag. 6,
term 2: pag. 1, pag. 3,
dan worden volgende resultaten teruggeven, daarbij de ranking volgend: pag. 4, pag. 6, pag. 3, pag. 1.

8.2 Meest centrale knoop

We bespreken een mogelijke definitie en een algoritme voor de berekening van de meest centrale knoop in een netwerk (de "spil" in het netwerk), gebaseerd op Sectie 2.1 van [EH10]. De graaf van het netwerk bestaat uit N knopen, waarbij we een directe verbinding tussen twee knopen een boog noemen. We beperken ons tot een enkelvoudige graaf, i.e., een ongerichte graaf waarin niet meer dan één boog tussen twee knopen voorkomt, en geen boog voorkomt van een knoop naar zichzelf. Ze kan voorgesteld worden door een binaire verbindingsmatrix A, waarbij $A_{ij} = 1$ als knoop i verbonden is met knoop j d.m.v. een boog, en $A_{ij} = 0$ in het andere geval. Omdat de graaf ongericht is, is matrix A symmetrisch. Een eerste indicatie over hoe centraal knoop i is, wordt gegeven door het aantal bogen vanuit die knoop, wat neerkomt op $\sum_{j=1}^{N} A_{ij}$. Omdat verbindingsmatrix A binair is en symmetrisch, kan deze uitdrukking herschreven worden als

$$\begin{aligned} \sum_{j=1}^{N} A_{ij} &= \sum_{j=1}^{N} A_{ij}^2 \\ &= \sum_{j=1}^{N} A_{ij}A_{ji} \\ &= (AA)_{ii}. \end{aligned} \tag{8.5}$$

Het aantal bogen vanuit knoop i kan ook geïnterpreteerd worden als het aantal lussen[1] startend in knoop i met lengte 2 (bijvoorbeeld van knoop i naar een bepaalde knoop j en terug). Uitdrukking (8.5) kan als volgt veralgemeend worden.

Stelling 8.2.2
Het aantal lussen van lengte n startend in knoop i wordt gegeven door het diagonaalelement

$$(A^n)_{i,i}\,,$$

met

$$A^n = \underbrace{A\ A\ \cdots\ A}_{n \text{ keer}}.$$

Men zou kunnen opmerken dat het aantal lussen van lengte 2, of het aantal vertrekkende bogen, niet alles zegt in verband met het belang van een knoop. Inderdaad, als een knoop

[1] Een lus van lengte n startend in knoop i is geordende lijst van knopen $(i,\ k_1,\ k_2, \ldots,\ k_{n-1},\ i)$, met $k_j \in \{1, \ldots, N\},\ j = 1, \ldots, n-1$, zodat opeenvolgende knopen d.m.v. een boog verbonden zijn.

verbonden is met andere knopen van waaruit veel bogen vertrekken is de situatie verschillend van het geval waarbij die andere knopen nauwelijks verbindingen hebben. Enerzijds is het daarom aangewezen om in de definitie van meest centrale knoop ook de lussen van lengte 2, 3, ... mee te nemen. Anderzijds is het duidelijk dat, hoe groter de lengte van de lussen, hoe minder gewicht we eraan moeten geven. Een mogelijke weging, met $1/2!, 1/3!, \ldots$, en een toepassing van Stelling 8.2.2 leiden ons tot

$$\frac{1}{2!}\left(A^2\right)_{i,i} + \frac{1}{3!}\left(A^3\right)_{i,i} + \cdots + \frac{1}{n!}\left(A^n\right)_{i,i} + \cdots$$

Vermits de diagonaal van A enkel nullen bevat en alleen de mate van centraal zijn van de knopen ten opzichte van elkaar belangrijk is, kunnen we even goed kijken naar

$$\left(I + \frac{A}{1!} + \frac{A^2}{2!} + \cdots + \frac{A^n}{n!} + \cdots\right)_{i,i},$$

waarbij de uitdrukking tussen haakjes de (matrix) exponentiële van A genoemd wordt, aangeduid met e^A. Dit brengt ons tot de volgende definitie.

DEFINITIE 8.2.1 *(centraliteit)*
De mate van centraliteit van knoop i wordt gegeven door $\left(e^A\right)_{i,i}$, voor $i = 1, \ldots, N$. De meest centrale knoop in het netwerk is de knoop waarvoor de mate van centraliteit maximaal is.

De meeste centrale knoop, volgens bovenstaande definitie, kan onder meer bepaald worden door de matrix exponentiële van A te berekenen, en vervolgens de waarden op de diagonaal te vergelijken. Men kan gebruik maken van volgende eigenschap: als

$$A = X \begin{bmatrix} \lambda_1 & & \\ & \ddots & \\ & & \lambda_N \end{bmatrix} X^{-1}$$

overeenkomt met een eigenwaardenontbinding, dan is

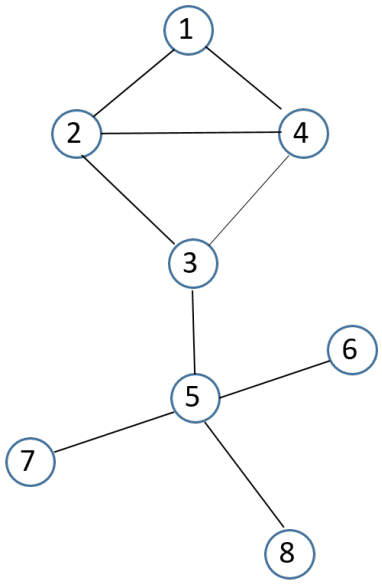
$$e^A = X \begin{bmatrix} e^{\lambda_1} & & \\ & \ddots & \\ & & e^{\lambda_N} \end{bmatrix} X^{-1}.$$

De kern van een klasse van algoritmen bestaat bijgevolg uit het berekenen van een volledige eigenwaardenontbinding van verbindingsmatrix A, waarvoor het *impliciet verschoven QR-algoritme* uit Sectie 9.2 gebruikt kan worden.

Voorbeeld 8.3 *Beschouw de enkelvoudige graaf met verbindingsmatrix*

$$A = \begin{bmatrix} 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \end{bmatrix}, \tag{8.6}$$

weergegeven in Figuur 8.2. Als we de diagonaal van A^2 uitrekenen verkrijgen we



Figuur 8.2: Ongerichte graaf met 8 knopen en verbindingsmatrix (8.6).

$$\text{diag}\left(A^2\right) = [2\ 3\ 3\ 3\ 4\ 1\ 1\ 1]^T.$$

Deze getallen komen inderdaad overeen met het aantal bogen vertrekkende uit de verschillende knopen. De diagonaal van e^A voldoet aan

$$\text{diag}\left(e^A\right) = [3.0279\ 4.3403\ 3.9914\ 4.3403\ 3.9126\ 1.6945\ 1.6945\ 1.6945]^T.$$

Bijgevolg zijn volgens Definitie 8.2.1 Knopen 2 en 4 meest centraal, hoewel ze minder directe verbindingen met andere knopen hebben dan Knoop 5. De lussen met lengte groter dan 2 geven de doorslag.

8.3 Graafpartitionering

Voor heel wat problemen gerelateerd aan het partitioneren van grafen en het clusteren van data, (deels) voorgesteld door middel van grafen, bestaan eigenwaardengebaseerde algoritmen. We illustreren dit aan de hand van een eenvoudig probleem, waarbij een graaf met een even aantal knopen in twee grafen met eenzelfde aantal knopen gesplitst wordt en daarbij getracht wordt zo weinig mogelijk bogen te doorbreken. We baseren ons op artikel [PSL90], waarvan de theoretische basis teruggaat op [Fie73].
We beschouwen de setting van Sectie 8.2 en vertrekken van een enkelvoudige graaf $\mathcal{G}$ met N knopen (N even) en binaire verbindingsmatrix A. We noemen de verzameling van knopen K en de verzameling van bogen B, wiskundig beschreven door

$$K = \{1, \ldots, N\}, \quad B = \{(i,j) \in K \times K : \ i < j, \text{ knoop } i \text{ is verbonden met knoop } j\}.$$

In de definitie van B leggen we de voorwaarde $i < j$ op omdat de graaf ongericht is, en we zo dubbels vermijden. De graad van knoop i, aangeduid met $g(i)$, is het aantal bogen vanuit knoop i. We kunnen nu de *graaf-Laplaciaan* $L = [L_{ij}]_{i,j=1}^{N}$ op de volgende manier definiëren:

$$L_{ij} = \begin{cases} -1, & \text{als } (i,j) \in B \text{ of } (j,i) \in B, \\ g(i), & \text{als } i = j, \\ 0, & \text{anders.} \end{cases}$$

Merk dat $L = D - A$, met $D = \text{diag}(g(1), \ldots, g(N))$. Matrix L heeft volgende eigenschappen.

Stelling 8.3.3
Zij L de graaf-Laplaciaan van een enkelvoudige graaf $\mathcal{G}$. Dan geldt:

1. *L is symmetrisch positief semi-definiet;*
2. *L heeft steeds een eigenwaarde nul en $\mathbf{1} = [1 \ 1 \ \cdots \ 1]^T$ is een bijhorende eigenvector;*
3. *Het aantal verbonden componenten van G is gelijk aan de multipliciteit van eigenwaarde nul van L.*

In wat volgt veronderstellen we dat de graaf geconnecteerd is, oftewel niet opsplitsbaar in twee grafen. De eigenwaarden van L voldoen dan aan

$$0 = \lambda_N < \lambda_{N-1} \leq \lambda_{N-2} \leq \cdots \leq \lambda_1.$$

De derde eigenschap in Stelling 8.3.3 suggereert al dat de grootte van λ_{N-1} een maat is voor de connectiviteit van de graaf, in de zin van hoe kleiner λ_{N-1}, hoe gemakkelijker het is om de graaf in twee deelgrafen te verdelen. We gaan deze uitspraak nu formaliseren.
We wensen de graaf op te delen in twee grafen $\mathcal{G}_1$ en $\mathcal{G}_2$, met knopenverzamelingen K_1 en K_2, waarbij $K_1 \cup K_2 = K$, $K_1 \cap K_2 = \phi$. Zulke partitie kunnen we voorstellen door een vector $x = [x_1 \ \cdots \ x_N]^T$, met

$$x_i = \begin{cases} 1, & i \in K_1, \\ -1, & i \in K_2. \end{cases}$$

De voorwaarde $\#(K_1) = \#(K_2)$, met $\#(\cdot)$ het aantal elementen, kunnen we dan uitdrukken als

$$\mathbf{1}^T x = 0.$$

Als we $B_x \subseteq B$ de verzameling noemen van alle bogen van $\mathcal{G}$ tussen een knoop van K_1 en een knoop van K_2, kunnen we uitrekenen dat

$$\begin{aligned} \tfrac{1}{4} x^T L x &= \tfrac{1}{4} \sum_{i=1}^{N} \sum_{j=1}^{N} L_{ij} x_i x_j \\ &= \tfrac{1}{2} \sum_{(i,j)\in B} (-x_i x_j + x_i^2) \\ &= \tfrac{1}{2} \sum_{(i,j)\in B_x} (-x_i x_j + x_i^2) \\ &= \tfrac{1}{2} \sum_{(i,j)\in B_x} 2 \\ &= \#(B_x). \end{aligned}$$

Het zoeken naar een partitie, waarbij $\#(K_1) = \#(K_2)$ en het aantal verbindingen tussen de twee deelgrafen minimaal is, kan bijgevolg beschreven worden door het optimalisatieprobleem

$$\begin{cases} \min_{x\in\{-1,1\}^N} \frac{1}{4} x^T L x, \\ \mathbf{1}^T x = 0. \end{cases} \tag{8.7}$$

Het moeilijke aan (8.7) is de binaire beperking $x \in \{-1, 1\}^N$, die het probleem discreet maakt. Het aantal mogelijke partities van de N knopen in twee groepen met eenzelfde aantal elementen is gelijk aan $\frac{1}{2}\binom{N}{\frac{N}{2}}$, waarvan de schaalbaarheid met N geïllustreerd wordt door volgende tabel:

N	16	32	64	128	256	512
$\frac{1}{2}\binom{N}{\frac{N}{2}}$	6.4350e+03	3.0054e+08	9.1631e+17	1.1976e+37	2.8843e+75	2.3628e+152

Een brute-force aanpak is daarom niet mogelijk voor grote waarden van N.

Een praktische manier die meestal leidt tot een goede partitie voor een aanvaardbare hoeveelheid rekenwerk bestaat erin om optimalisatieprobleem (8.7) te vereenvoudigen, door toe te laten dat de componenten van x reële getallen zijn en de beperking $x \in \{-1, 1\}^N$ te vervangen door $\|x\|_2^2 = x_1^2 + \cdots + x_N^2 = N$, wat ons brengt tot

$$\begin{cases} \min_{x\in\mathbb{R}^N} \frac{1}{4} x^T L x, \\ \|x\|_2^2 = N, \quad \mathbf{1}^T x = 0. \end{cases} \tag{8.8}$$

De oplossing wordt beschreven door volgende stelling.

STELLING 8.3.4
Veronderstel dat $\lambda_{N-1} < \lambda_{N-2}$. Dan wordt de oplossing van (8.8) gekenmerkt door minimum $\frac{1}{4} N \lambda_{N-1}$ en bereikt door v_{N-1}, de eigenvector van L horende bij eigenwaarde λ_{N-1}.

Bewijs Vermits L symmetrisch is, is L diagonaliseerbaar, en heeft de matrix een eigenwaardenontbinding van de vorm

$$L = V \operatorname{diag}(\lambda_1, \ldots, \lambda_{N-1}, \lambda_N)\, V^T,$$

met $V = [v_1 \ \cdots \ v_N]$ een orthogonale matrix die eigenvectoren groepeert. Omwille van het feit dat λ_N nul is en omwille van Stelling 8.3.3 geldt dat

$$\mathbf{1}^T x = v_N^T x.$$

Doen we de substitutie $x = Vy$, en maken we gebruik van het feit dat een orthogonale transformatie de 2-norm bewaart en van $v_N^T x = v_N^T V y = [0 \ \cdots \ 0 \ 1]y = y_N$, wordt (8.8) omgezet naar

$$\begin{cases} \min_{y \in \mathbb{R}^N} \frac{1}{4} \sum_{i=1}^N \lambda_i y_i^2, \\ \|y\|_2^2 = N, \quad y_N = 0. \end{cases}$$

De oplossing wordt bekomen door alle gewicht aan y_{N-1} te geven. Het minimum is dus $\frac{1}{4}\lambda_{N-1}N$ en wordt bereikt door $y = [0 \ \cdots \ 0 \ \sqrt{N} \ 0]^T$, wat overeenkomt met $x = \sqrt{N}\, v_{N-1}$. □

Door de vereenvoudiging van (8.7) door (8.8) gaat het binaire karakter van de optimale x, in de betekenis van $x_i \in \{-1, 1\}$, verloren: de elementen van v_{N-1} zijn in het algemeen niet even groot. Daarom maken we een partitie gebaseerd op het teken van de elementen van v_{N-1}. Samengevat komen we tot het volgende algoritme.

1. Bereken de eigenvector v_{N-1}, horende bij de tweede-kleinste eigenwaarde van L.

2. Partitioneer de graaf op basis van het teken van de elementen van v_{N-1}.

Voor het vinden van de *tweede-kleinste eigenwaarde* van de typisch grote en spaarse symmetrische matrix L, is Lanczos' algoritme aangewezen, dat we zullen behandelen in §9.1.3.5.

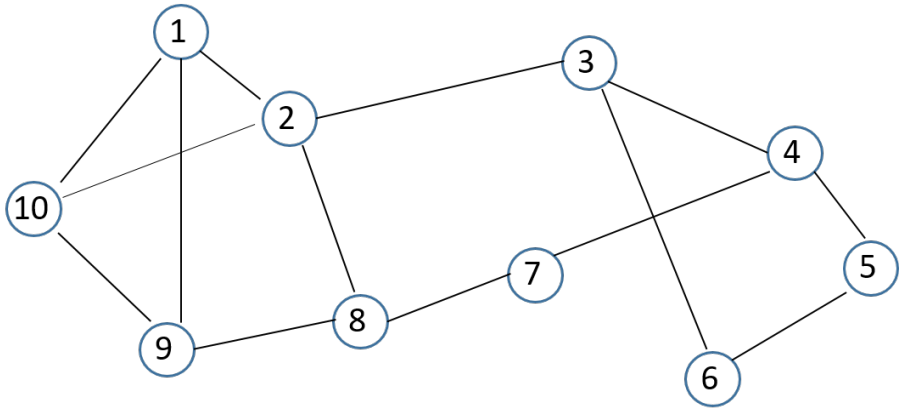
Opmerking 8.2 *Doordat we de beperkingen "verzacht" hebben, is het minimum van (8.8) kleiner of gelijk aan datgene van (8.7). Bijgevolg is het minimaal aantal bogen dat bij een gelijke partitie doorbroken moet worden langs onder begrensd door het minimum van (8.8), zijnde* $\frac{N}{4}\lambda_{N-1}$. *De tweede-kleinste eigenwaarde* λ_{N-1} *van* L *wordt de* algebraïsche connectiviteit *van* $\mathcal{G}$ *genoemd en de bijhorende eigenvector de* Fiedler vector.

Voorbeeld 8.4 *Beschouw een graaf met* $N = 10$ *knopen, waarbij*

$$A = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 \\ 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix}, L = \begin{bmatrix} 3 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & -1 \\ -1 & 4 & -1 & 0 & 0 & 0 & 0 & -1 & 0 & -1 \\ 0 & -1 & 3 & -1 & 0 & -1 & 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & 3 & -1 & 0 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 & 2 & -1 & 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 & -1 & 2 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 & 0 & 0 & 2 & -1 & 0 & 0 \\ 0 & -1 & 0 & 0 & 0 & 0 & -1 & 3 & -1 & 0 \\ -1 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 3 & -1 \\ -1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 3 \end{bmatrix}.$$

De drie kleinste eigenwaarden van L *zijn*

$$\lambda_{10} = 0, \ \lambda_9 = 0.45112801, \ \lambda_8 = 1.22935459.$$



Figuur 8.3: Graaf horende bij Voorbeeld 8.4.

De eigenvector horende bij de tweede-kleinste eigenwaarde is

$$v_9 = \begin{bmatrix} 0.35792561 \\ 0.19777282 \\ -0.20707804 \\ -0.28943830 \\ -0.46846236 \\ -0.43614992 \\ -0.06220078 \\ 0.19309724 \\ 0.35660813 \\ 0.35792561 \end{bmatrix}$$

Op basis hiervan delen we de graaf op in twee deelgrafen, met knopen $K_1 = \{1, 2, 8, 9, 10\}$ *en* $K_2 = \{3, 4, 5, 6, 7\}$. *Dit is consistent met Figuur 8.3 die de graaf visualiseert.*
Zoals beschreven in Opmerking 8.2 zijn $\frac{N}{4}\lambda_{N-1} = 1.12782004$ *en, door afronding naar boven, het getal twee een ondergrens op het minimum aantal te doorbreken bogen. Bij de berekende oplossing wordt de ondergrens twee gerealiseerd.*

Hoofdstuk 9

Eigenwaardenalgoritmes

In dit hoofdstuk komen veelgebruikte algoritmes voor het oplossen van eigenwaardenproblemen aan bod.
In Sectie 9.1 behandelen we algoritmes voor het berekenen van één of een beperkt aantal eigenwaarden. Deze algoritmes zijn toepasbaar op zéér grote matrices waarbij het berekenen van alle eigenwaarden niet haalbaar is. Specifiek voor deze algoritmes is dat deze geen bewerkingen zoals transformaties uitvoeren op de matrix zelf, maar enkel gebruik maken van matrix-vectorproducten. Merk dat bij een m-bij-m matrix de computationele kost van een matrix vectorproduct $\sim 2m^2$ elementaire bewerkingen bedraagt, in contrast met bijvoorbeeld $\sim 4m^3$ bewerkingen voor een QR-factorisatie. Wanneer de matrix veel nullen bevat, kan de kost van een matrix-vectorvermenigvuldiging bovendien verder naar beneden gehaald worden.
In Sectie 9.2 beschrijven we vervolgens het standaard algoritme voor het bepalen van *alle* eigenwaarden van een matrix, dat volledig gebaseerd is op orthogonale transformaties.

Alle beschreven algoritmes zijn ofwel inherent iteratief (deze van Sectie 9.1), ofwel hebben ze een component die iteratief is (het algoritme van Sectie 9.2). Hier is een fundamentele reden voor, die in volgende stelling uiteengezet wordt.

Stelling 9.0.1 *(adaptatie van Theorem 25.1 uit [TB97])*
Voor elke $m \geq 5$ bestaat er een matrix $A \in \mathbb{Q}^{m\times m}$ die een reële eigenwaarde r heeft, waarbij r niet uitgedrukt kan worden in termen van rationale getallen en de bewerkingen optellen, aftrekken, vermenigvuldigen, delen, alsook het nemen van $\sqrt[k]{\cdot}$, met $k \in \mathbb{N}$.

Hoewel de voorgestelde algoritmes algemeen toepasbaar zijn, zullen we in de wiskundige analyse steeds de veronderstelling maken dat de beschouwde matrix $A \in \mathbb{C}^{m\times m}$ diagonaliseerbaar is:

$$A = XDX^{-1}, \tag{9.1}$$

waarbij de kolommen van $X = [x_1 \ \cdots \ x_m]$ eigenvectoren zijn en de diagonaal van $D = \text{diag}(\lambda_1, \ldots, \lambda_m)$ de overeenkomstige eigenwaarden bevat, met de ordening

$$|\lambda_1| \geq |\lambda_2| \geq \cdots \geq |\lambda_m|. \tag{9.2}$$

Verder zullen we in dit hoofdstuk bij de analyse van algoritmes vaak gebruik maken van functies van matrices, en meer bepaald veeltermen. Gegeven een veelterm $p(z) = \sum_{i=0}^{n} c_i z^i$ en matrix $A \in \mathbb{C}^{m\times m}$, definiëren we $p(A)$ als

$$p(A) = c_0 I_m + c_1 A + c_2 A^2 + \cdots + c_n A^n.$$

Bestaat de eigenwaardenontbinding (9.1) dan kan je gemakkelijk nagaan dat $A^k = X D^k X^{-1}$ enz., zodat

$$p(A) = X\, p(D)\, X^{-1}, \quad \text{met} \quad p(D) = \operatorname{diag}(p(\lambda_1), \ldots, p(\lambda_m)). \tag{9.3}$$

Dus, de eigenwaarden van $p(A)$ bekom je door functie p toe te passen op de eigenwaarden van A. Dit resultaat, dat ook geldt als A niet diagonaliseerbaar is en voor meer algemene functies[1], wordt het *spectral mapping theorem* genoemd. Tenslotte, als p_A de karakteristieke veelterm van A is, $p_A(z) = \det(zI - A)$, dan geldt $p_A(A) = 0$. Dit resultaat, dat direct uit (9.3) volgt, wordt de *stelling van Cayley-Hamilton* genoemd.

9.1 Berekening van selecte eigenwaarden

We beginnen met de methode van de machten voor het berekenen van de eigenvector horende bij de dominante eigenwaarde. Vervolgens breiden we deze methode uit op twee manieren. Bij zgn. deelruimte-iteratie itereren we niet op één vector maar op een verzameling van vectoren die een basis vormen van een deelruimte. Bij de methode van Arnoldi zoeken we benaderingen van eigenvectoren in de ruimte opgespannen door vectoren die met de methode van de machten gegenereerd worden.

9.1.1 Methode van de machten

Voor matrix $A \in \mathbb{C}^{m\times m}$, startvector $x^{(0)} \in \mathbb{C}^m$ en een norm $\|\cdot\|$ op $\mathbb{C}^m$, kan de basisiteratie als volgt beschreven worden:

$$p^{(k)} = A x^{(k-1)}, \quad x^{(k)} = \frac{p^{(k)}}{\|p^{(k)}\|}, \ k = 1, 2, 3, \ldots \tag{9.4}$$

Wanneer $|\lambda_1| > |\lambda_2|$ is, convergeert $\{x^{(k)}\}_{k\geq 1}$ onder milde voorwaarden naar de eigenvector horende bij de dominante eigenwaarde λ_1. Dit kan als volgt ingezien worden. De m eigenvectoren van A vormen een basis. Stel nu

$$x^{(0)} = \sum_{i=1}^{m} c_i x_i,$$

waarbij $c_1 \neq 0$. Omdat de tweede stap in de basisiteratie slechts een normalisatie is, berekenen we in de k-de stap op een constante na $A^k x^{(0)}$. Nu is $A^k x^{(0)} = \sum_{i=1}^{m} c_i \lambda_i^k x_i$ en

[1] We zijn de exponentiële van een matrix bijvoorbeeld reeds tegengekomen in Sectie 8.2.

dus

$$\frac{1}{c_1\lambda_1^k}A^k x^{(0)} = x_1 + \sum_{i=2}^{m} \frac{c_i}{c_1}\left(\frac{\lambda_i}{\lambda_1}\right)^k x_i,$$

waarbij alle termen behalve de eerste naar nul convergeren als $k \to \infty$. We krijgen dus lineaire convergentie, met in het generieke geval als convergentiefactor $|\lambda_2|/|\lambda_1|$, de verhouding van moduli van de tweede grootste en grootste eigenwaarde. We geven nog enkele opmerkingen.

- Een schatting van de eigenwaarde bekom je via het Rayleigh quotiënt $r(x^{(k)})$, met $r(x) = \frac{x^*Ax}{x^*x}$.

- Voor een willekeurig gekozen startvector is de voorwaarde $c_1 \neq 0$ voldaan met waarschijnlijkheid[2] 1. In het speciale geval waarbij $c_1 = 0$, wordt vaak alsnog een component volgens x_1 gegenereerd door afrondingsfouten.

- Vervang je A door A^{-1} in de basisiteratie, wat je implementeert door het stelsel $Ap^{(k)} = x^{(k-1)}$ op te lossen, bereken je de eigenvector horende bij de eigenwaarde met kleinste modulus. Dit wordt *inverse iteratie* genoemd.

- Vervang je A door $(\sigma I - A)^{-1}$, met $\sigma \in \mathbb{C}$, bereken je de eigenvector horende bij de eigenwaarde dichtst bij σ. Dit wordt *inverse iteratie met verschuiving* genoemd. Wordt in elke iteratie σ aangepast met het Rayleigh quotiënt $r(x^{(k-1)})$, spreken we van Rayleigh quotiënt iteratie. Als de Rayleigh quotiënt iteratie convergeert is de asymptotische convergentiesnelheid onder milde voorwaarden *kubisch*.

We geven nog een didactisch voorbeeld.

Voorbeeld 9.1 *We beschouwen*

$$A = \begin{bmatrix} 1.0 & 0.5 & 2.0 \\ 2.0 & 0.0 & 3.0 \\ -1.0 & 2.0 & 4.0 \end{bmatrix}$$

en startvector $x^{(0)} = [1 \;\; 1 \;\; 1]^T = \begin{bmatrix} x_1^{(0)} & x_2^{(0)} & x_3^{(0)} \end{bmatrix}^T$.
Als we de normalisatiestap aanpassen zodat de eerste component 1 *wordt, is het resultaat van de methode van de machten:*

[2] Als een voorwaarde voldaan is met waarschijnlijkheid 1, betekent dit niet dat er geen situaties bestaan waarbij de voorwaarde niet voldaan is.

i	$x_1^{(i)}$	$x_2^{(i)}$	$x_3^{(i)}$	*berekende eigenwaarde*
0	*1.0*	*1.000000000*	*1.000000000*	
1	*1.0*	*1.428571429*	*1.428571429*	*3.500000000*
2	*1.0*	*1.375000000*	*1.656250000*	*4.571428571*
3	*1.0*	*1.393750000*	*1.675000000*	*5.000000000*
4	*1.0*	*1.391950464*	*1.681733746*	*5.046875000*
5	*1.0*	*1.392485620*	*1.682168645*	*5.059442724*
6	*1.0*	*1.392430472*	*1.682345828*	*5.060580100*
7	*1.0*	*1.392445590*	*1.682355443*	*5.060906892*
8	*1.0*	*1.392443919*	*1.682360112*	*5.060933681*
9	*1.0*	*1.392444347*	*1.682360315*	*5.060942183*
10	*1.0*	*1.392444307*	*1.682360439*	*5.060942804*
11	*1.0*	*1.392444309*	*1.682360443*	*5.060943026*
12	*1.0*	*1.392444308*	*1.682360446*	*5.060943040*
13	*1.0*	*1.392444308*	*1.682360446*	*5.060943046*
14	*1.0*	*1.392444308*	*1.682360446*	*5.060943046*

De eigenwaarden van de matrix zijn $\lambda_1 = 5.060943046$, $\lambda_2 = -0.862636908$ *en* $\lambda_3 = 0.801693862$. *De convergentie naar* λ_1 *is lineair, met snelheid bepaald door* $\left|\frac{\lambda_2}{\lambda_1}\right| = 0.17$.

9.1.2 Deelruimte-iteratie

In plaats van op één vector te itereren zoals bij de methode van de machten, itereren we nu op een volledige deelruimte opgespannen door een stel van n orthonormale vectoren $\{q_1^{(0)}, q_2^{(0)}, \ldots, q_n^{(0)}\}$, met typisch $n \ll m$. We bepalen van de deelruimte opgespannen door $\{Aq_1^{(0)}, Aq_2^{(0)}, \ldots, Aq_n^{(0)}\}$ een orthonormale basis $\{q_1^{(1)}, q_2^{(1)}, \ldots, q_n^{(1)}\}$ en itereren dan daarmee opnieuw.
In Sectie 2.3 hebben we gezien dat een orthogonalisatie van vectoren geïnterpreteerd kan worden in termen van een onvolledige QR-factorisatie. Definiëren we matrix $\underline{Q}_k = [q_1^{(k)} \ \cdots \ q_n^{(k)}]$ voor $k \geq 0$, dan wordt de basisiteratie als volgt beschreven: startend met beginmatrix $\underline{Q}_0$, voeren we iteraties uit van de vorm

$$P_k = A\,\underline{Q}_{k-1}, \quad \underline{Q}_k \underline{R}_k = P_k, \quad k = 1, 2, 3, \ldots \tag{9.5}$$

Merk dat de normalisatie in (9.4) vervangen is door het bepalen van de Q-factor uit de onvolledige QR-factorisatie. Immers, wanneer we A toepassen op een stel vectoren, moeten we nadien niet alleen de lengte van de vectoren terugbrengen tot 1, maar ook de orthogonaliteit herstellen.

Als $|\lambda_n| > |\lambda_{n+1}|$ is in (9.2), dan convergeert voor bijna alle initiële vectoren de ruimte $< q_1^{(k)}, \ldots, q_n^{(k)} >$ naar de ruimte opgespannen door de eigenvectoren horende bij de n grootste eigenwaarden in modulus, wat we kunnen noteren als

$$\lim_{k\to\infty} < q_1^{(k)} \ , \ q_2^{(k)} \ , \ \ldots \ , \ q_n^{(k)} > = < x_1 \ , \ x_2 \ , \ \ldots \ , \ x_n > . \tag{9.6}$$

We kunnen dus met de methode eigenvectoren overeenkomstig *n dominante eigenwaarden* berekenen. We geven nog een technische verduidelijking van dit resultaat. Omdat bij een orthogonalisatie van vectoren het bereik niet verandert, geldt

$$< q_1^{(k)} \ , \ q_2^{(k)} \ , \ \ldots \ , \ q_n^{(k)} >=< A^k q_1^{(0)} \ , A^k \ q_2^{(0)} \ , \ \ldots \ , A^k \ q_n^{(0)} > .$$

Stel nu dat een vector q ontbonden is als $q = \sum_{i=1}^{m} c_i x_i$ dan geldt

$$\frac{1}{c_n \lambda_n^k} A^k q = \sum_{i=1}^{n-1} \frac{c_i}{c_n} \left(\frac{\lambda_i}{\lambda_n} \right)^k x_i + x_n + \sum_{i=n+1}^{m} \frac{c_i}{c_n} \left(\frac{\lambda_i}{\lambda_n} \right)^k x_i.$$

In woorden, door vermenigvuldiging met A^k groeien de componenten volgens $x_1, \ldots, x_n$ voor toenemende k of blijven even groot, terwijl de componenten volgens $x_{n+1}, \ldots, x_m$ allen naar nul convergeren. Vermits de dimensie van $< q_1^{(k)} \ , \ldots , \ q_n^{(k)} >$ gelijk blijft aan n doorheen de iteraties, moet (9.6) gelden, waarbij de convergentie wordt bepaald door de verhouding $\frac{|\lambda_{n+1}|}{|\lambda_n|}$. We merken nog op:

- Als je bij het uitvoeren van de basisiteratie (9.5) enkel kijkt naar de eerste kolom van matrices $\underline{Q}_k$, komt wat je ziet overeen met iteraties van de methode van de machten. Als je enkel kijkt naar de eerste twee kolommen van deze matrices, krijg je de iteraties van deelruimte-iteratie met twee startvectoren enz. Zijn de grootste $n+1$ eigenwaarden van A verschillend in modulus, dan geldt dus niet alleen (9.6), maar ook

  $$\begin{array}{l} \lim_{k\to\infty} < q_1^{(k)} > = < x_1 > \\ \lim_{k\to\infty} < q_1^{(k)} \ , \ q_2^{(k)} > = < x_1 \ , \ x_2 > \\ \qquad\qquad\qquad\qquad \ddots \\ \lim_{k\to\infty} < q_1^{(k)} \ , \ q_2^{(k)} \ , \ \ldots \ , \ q_{n-1}^{(k)} > = < x_1 \ , \ x_2 \ , \ \ldots \ , \ x_{n-1} > . \end{array}$$

  De achterliggende reden is dat een QR-factorisatie een *geneste* orthonormale basis opstelt, zie (2.17).

- Benaderingen van de n dominante eigenwaarden in de k-de iteratiestap worden gegeven door de eigenwaarden van matrix $\underline{Q}_k^* A \underline{Q}_k \in \mathbb{C}^{n\times n}$. Immers, stel dat convergentie bereikt is en dat $Q \in \mathbb{C}^{m\times n}$ voldoet aan $\mathcal{R}(Q) =< x_1, \ldots, x_n >$. Voor $i \in \{1, \ldots, n\}$ geldt dan $x_i = Q y_i$ voor één of andere $y_i \in \mathbb{C}^n$ en $AQy_i = \lambda_i Q y_i$. Vermits $Q^*Q = I_n$ impliceert dit laatste $Q^*AQ\, y_i = \lambda_i y_i$. Door het kleine eigenwaardenprobleem

  $$Q^*AQ\, y = \lambda y$$

  op te lossen vinden we bijgevolg de dominante eigenwaarden van A, en de bijhorende eigenvectoren via de relatie $x_i = Q y_i$.
  Voor het gereduceerde eigenwaardenprobleem kan het algoritme van Sectie 9.2 gebruikt worden.

- Van uitbreidingen van de methode van de machten zoals inverse iteratie met ev. verschuiving kunnen ook deelruimte-varianten worden opgesteld.

We eindigen opnieuw met een *didactisch* voorbeeld.

Voorbeeld 9.2 *We beschouwen matrix*

$$A = \begin{bmatrix} -22.2 & -19.2 & -11.6 \\ -19.6 & -14.6 & -9.8 \\ 83.8 & 67.8 & 42.9 \end{bmatrix},$$

waarvoor het volgende geldt.

Eigenwaarden	*Eigenvectoren*
5	$[0.2357 \;\; 0.2357 \;\; -0.9428]^T = x_1$
1	$[0.4472 \;\; 0.0000 \;\; -0.8944]^T = x_2$
0.1	$[0.3369 \;\; 0.1684 \;\; -0.9264]^T = x_3.$

Iteraties van (9.5) leiden tot

$$\underline{Q}_0 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 0 & 0 \end{bmatrix}, \quad \underline{Q}_1 = \begin{bmatrix} -0.2498 & 0.6973 \\ -0.2205 & -0.7166 \\ 0.9429 & 0.0171 \end{bmatrix},$$

$$\underline{Q}_2 = \begin{bmatrix} -0.2391 & 0.7054 \\ -0.2323 & -0.7088 \\ 0.9428 & 0.0043 \end{bmatrix}, \quad \underline{Q}_3 = \begin{bmatrix} -0.2358 & 0.7071 \\ -0.2356 & -0.7072 \\ 0.9428 & 0.0002 \end{bmatrix},$$

$$\underline{Q}_4 = \begin{bmatrix} -0.2357 & 0.7071 \\ -0.2357 & -0.7071 \\ 0.9428 & 0.0000 \end{bmatrix} \approx [-x_1 \quad -3x_1 + \sqrt{10}x_2].$$

9.1.3 De methode van Arnoldi

Bij de methode van de machten, gestart met $x^{(0)} = b$, worden in opeenvolgende iteraties vectoren $\{b, Ab, A^2b, \ldots\}$ berekend, op schaalfactoren na. Slechts de laatst bekomen vector wordt gebruikt in de uiteindelijke schatting van de dominante eigenwaarde. Het basisidee achter de methode van Arnoldi is om in de k-de iteratie benaderende eigenvectoren te zoeken in de ruimte

$$\mathcal{K}_k(A, b) =< b, Ab, A^2b, \ldots, A^{k-1}b >, \tag{9.7}$$

waardoor beter gebruik gemaakt wordt van de beschikbare informatie. We zullen zien dat we op deze manier in de k-de iteratiestap k benaderingen van eigenwaarden kunnen bepalen. Dit biedt een dubbel voordeel: het aantal iteraties verhogen is niet alleen nuttig om de nauwkeurigheid van een eigenwaardenbenadering te verbeteren, maar leidt ook tot benaderingen van méér eigenwaarden en laat op die manier toe om het spectrum wat af te tasten.
De ruimte (9.7) is een zogenaamde Krylov-ruimte.

DEFINITIE 9.1.1 *(Krylov-ruimte)*
De deelruimte (9.7) van $\mathbb{C}^m$ noemt men de Kylov-ruimte van dimensie k, gegenereerd door matrix A en startvector b.

9.1.3.1 Orthonormale basis van Krylov-ruimte

Om op een robuuste manier op zoek te gaan naar benaderingen van eigenvectoren in de Krylov-ruimte (9.7) hebben we een goede basis nodig. Vectoren $\{b, Ab, A^2b, \ldots, A^{k-1}b\}$ zijn echter verre van geschikt. Immers, overeenkomstig de methode van de machten geldt $\lim_{k\to\infty} < A^k b >=< x_1 >$. Dit betekent dat opeenvolgende vectoren allemaal benaderingen zijn van de dominante eigenvector en die benadering beter wordt bij toenemende k. In Voorbeeld 2.2 hebben we reeds geïllustreerd dat zulke basis verre van orthogonaal is, zelfs als de basisvectoren individueel genormaliseerd zijn.
De oplossing bestaat erin om een (geneste) orthonormale basis $\{q_1, \ldots, q_k\}$ van $\mathcal{K}_k(A, b)$ op te stellen. Omdat we in een setting zitten waarbij m typisch zeer groot is (grote matrix), $k \ll m$ (we zoeken een beperkt aantal eigenwaarden) en we in opeenvolgende iteraties (bij stijgende k) de basis 'on the fly' willen uitbreiden, ligt een Gram-Schmidt gebaseerd algoritme voor de hand. Dit brengt ons dan tot $k-1$ iteraties van Algoritme 6.
De meetkundige interpretatie van dit algoritme is als volgt. De eerste basisvector q_1 is evenredig met b. Vervolgens berekenen we $v_1 = Ab/\|b\|_2$. Zoals in het Gram-Schmidt algoritme verwijderen we de component volgens q_1. Wat overblijft wordt genormaliseerd tot q_2. In de j-de stap, $j \geq 2$, berekenen we Aq_j, we verwijderen de componenten volgens $q_1, \ldots, q_j$ die gegeven worden door $P_{<q_1,\ldots,q_j>}v_j$. Wat overblijft is orthogonaal met betrekking tot deze vectoren, en wordt genormaliseerd tot q_{j+1}. Merk dat $v_j = Aq_j$ numeriek gezien een veel betere kandidaat is om de ruimte $< q_1, \ldots, q_j >$ uit te breiden dan $A^j b$. Immers, voor grote j is deze laatste vector een goede benadering van eigenvector x_1, die op zijn beurt goed benaderd wordt in $< q_1, \ldots, q_j >$. In Stap 6 zouden dus 'gevaarlijke aftrekkingen' optreden. Door de gemaakte keuze van v_j is het enige grote verschil met Algoritme 1 dat de te orthogonaliseren vectoren (kolommen) niet op voorhand vast liggen.

Opmerking 9.1 *We hebben impliciet de veronderstelling gemaakt dat, voor de beschouwde stappen, $h_{j+1\,j}$, berekend in lijn 8, verschillend van nul is. In §9.1.3.3 beschouwen we het geval waarbij dit niet het geval is.*

Opmerking 9.2 *In een praktisch algoritme gebeurt er omwille van numerieke stabiliteit na stappen 4-7 een herorthogonalisatie (zie het einde van §2.3.2).*

Uit Algoritme 6 halen we voor $j = 1$ dat $Aq_1 - h_{11}q_1 = h_{21}q_2$ of

$$Aq_1 = q_1 h_{11} + q_2 h_{21}.$$

Op een gelijkaardige manier vinden we voor $j = 2$ dat $Aq_2 - h_{12}q_1 - h_{22}q_2 = h_{32}q_3$ of

$$Aq_2 = q_1 h_{12} + q_2 h_{22} + q_3 h_{32}.$$

Algoritme 6 Arnoldi's algoritme - orthonormale basis van Krylov-ruimte

1: $q_1 = \frac{b}{\|b\|_2}$
2: **for** $j = 1, 2, 3, \ldots$ **do**
3: $\quad v_j = Aq_j$
4: $\quad$ **for** $i = 1$ to j **do**
5: $\quad\quad h_{ij} = q_i^* v_j$
6: $\quad\quad v_j = v_j - h_{ij} q_i$
7: $\quad$ **end for**
8: $\quad h_{j+1\,j} = \|v_j\|_2$
9: $\quad q_{j+1} = v_j / h_{j+1\,j}$
10: **end for**

We kunnen dit proces verder zetten tot $j = k$. Stellen we tenslotte

$$Q_k = [q_1\ q_2\ \ldots q_k] \in \mathbb{C}^{m\times k},\ H_k = \begin{bmatrix} h_{11} & h_{12} & & \cdots & h_{1k} \\ h_{21} & h_{22} & & \cdots & h_{2k} \\ 0 & h_{32} & h_{33} & \cdots & h_{3k} \\ \vdots & \ddots & \ddots & & \vdots \\ 0 & \cdots & 0 & h_{k\,k-1} & h_{kk} \end{bmatrix} \in \mathbb{C}^{k\times k}$$

en $\mathbf{e}_k = \begin{bmatrix} 0 & \cdots & 0 & 1 \end{bmatrix}^T \in \mathbb{C}^k$, dan krijgen we in matrix-notatie

$$AQ_k = Q_k H_k + q_{k+1}\mathbf{e}_k^T h_{k+1\,k}. \tag{9.8}$$

Vergelijking (9.8) wordt de *recursiebetrekking* van de Arnoldi-iteratie genoemd. Een matrix met de structuur van H_k wordt een Hessenberg-matrix genoemd.

9.1.3.2 Benaderen van eigenwaarden

We willen bij de k-de iteratie een benadering (λ, q) bepalen van een eigenwaarde-eigenvector paar, waarbij $q \in \mathcal{K}_k(A, b)$, oftewel $q = Q_k y$ met $y \in \mathbb{C}^k$. We wensen dat

$$r(\lambda, y) := (A - \lambda I)Q_k y \approx 0,$$

met r het zogenaamde residu. Een gelijkheidseis is veel te streng omdat dit zou impliceren dat $Aq \in \mathcal{K}_k(A, b)$. Wat we daarom opleggen is dat de orthogonale projectie van het residu op de Krylov-ruimte nul is, dat is $P_{<q_1,\ldots,q_k>} r(\lambda, y) = 0$. Overeenkomstig (2.15) is $P_{<q_1,\ldots,q_k>} = Q_k Q_k^*$ wat ons brengt tot

$$\begin{aligned} &Q_k Q_k^* (A - \lambda I) Q_k y = 0 \\ &\qquad\Downarrow \\ &Q_k \left(Q_k^* A Q_k y - \lambda y\right) = 0 \\ &\qquad\Downarrow \\ &\left(Q_k^* A Q_k\right) y = \lambda y. \end{aligned}$$

We moeten dus het eigenwaardenprobleem voor de k-bij-k matrix $Q_k^* A Q_k$ oplossen. Een groot voordeel van Arnoldi's algoritme is dat we deze matrix niet meer expliciet moeten uitrekenen. Uit de recursiebetrekking (9.8) volgt namelijk dat

$$Q_k^* A Q_k = H_k.$$

We besluiten als volgt.
In de k-de iteratiestap van Algoritme 6 kunnen we eigenwaardenbenaderingen voor matrix $A \in \mathbb{C}^{m\times m}$ bekomen door alle eigenwaarden van matrix $H_k \in \mathbb{C}^{k\times k}$ te berekenen (bijvoorbeeld met het algoritme van Sectie 9.2), waarvan de elementen door het Gram-Schmidt orthogonalisatieproces bepaald zijn. Deze eigenwaarden worden *Ritz-waarden* genoemd. Merk verder op dat voor alle $1 \leq k \leq m-1$, matrix H_k een deelmatrix is van H_{k+1}.

Voorbeeld 9.3 *Stel* $A = \mathrm{diag}(0, 0.01, 0.02, \ldots, 1.99, 2, 2.5, 3) \in \mathbb{R}^{203\times 203}$ *en* $b \in \mathbb{R}^{203}$ *een vector met alle elementen gelijk aan* 1. *De opeenvolgende Ritz-waarden zijn - we geven slechts de 3 grootste (vanaf* $k \geq 3$*):*

k	$\lambda_{(k-2)}^{(k)}$	$\lambda_{(k-1)}^{(k)}$	$\lambda_{(k)}^{(k)}$
1			*1.0172414*
2		*0.4630208*	*1.6737321*
3	*0.2879595*	*1.2830703*	*2.2281918*
4	*0.8569504*	*1.6886660*	*2.8191520*
5	*1.2853930*	*1.8716956*	*2.9569284*
6	*1.6377214*	*2.1094416*	*2.9863859*
7	*1.8084728*	*2.3914197*	*2.9974232*
8	*1.8782509*	*2.4814854*	*2.9997497*
9	*1.9163200*	*2.4971726*	*2.9999807*
10	*1.9397762*	*2.4995774*	*2.9999986*

Merk dat we in de 10-de iteratiestap een klein eigenwaardenprobleem van dimensie 10-bij-10 moeten oplossen, terwijl het originele eigenwaardenprobleem dimensie 203 heeft.

9.1.3.3 Afbreken van de iteratie

Wanneer bij het uitvoeren van Algoritme 6 blijkt dat

$$h_{21} \neq 0,\ h_{32} \neq 0, \ldots, h_{k\,k-1} \neq 0,\ h_{k+1\,k} = 0$$

en $k < m$, dan kunnen we de iteratie niet verder zetten. We noemen dit een *prematuur afbreken* van de Arnoldi-iteratie. Als dit gebeurt, dan herleid (9.8) zich tot $AQ_k = Q_k H_k$, voluit

$$\left\{\begin{array}{rcl} Aq_1 & = & h_{11}q_1 + h_{21}q_2 \\ \vdots & \vdots & \ddots \\ Aq_{k-1} & = & h_{1\,k-1}q_1 + h_{2\,k-1}q_2 + \cdots + h_{k\,k-1}q_k \\ Aq_k & = & h_{1k}q_1 + h_{2k}q_2 + \cdots + h_{kk}q_k. \end{array}\right.$$

Dit betekent dat de ruimte $\mathcal{K}_k(A,b)$ een A-invariante deelruimte[3] is van $\mathbb{C}^m$.
Men kan aantonen dat elke A-invariante deelruimte opgespannen wordt door eigenvectoren en eventueel hoofdvectoren van A. Vermits we veronderstellen dat A diagonaliseerbaar is, betekent het premature afbreken dat

$$\mathcal{K}_k(A,b) =< v_1, \ldots, v_k >, \tag{9.9}$$

met $v_i,\ i = 1, \ldots, k$, eigenvectoren van A. Men kan dan nagaan dat de bijhorende eigenwaarden samenvallen met de Ritz-waarden.
Uitdrukking (9.9), en het feit dat q_1 een veelvoud van b is, impliceren dat

$$b = \sum_{i=1}^{k} d_i v_i,$$

met $d_i \in \mathbb{C},\ i = 1, \ldots, k$. Het prematuur afbreken kan dus enkel wanneer de startvector b slechts componenten heeft volgens een beperkt aantal eigenvectoren van A. Met een willekeurig gekozen startvector is de kans dat deze situatie zich voordoet nul.

9.1.3.4 Convergentie-eigenschappen

Enkele kenmerken van het algoritme van Arnoldi zijn als volgt.

- De convergentie van individuele Ritz-waarden naar eigenwaarden verloopt kwalitatief lineair, in de zin dat uiteindelijk de fout in elke iteratie met een constante factor afneemt. We kunnen echter niet spreken over lineaire convergentie in strikte zin, omdat dat een asymptotische eigenschap is, terwijl na een eindig aantal stappen (namelijk $m-1$) de groeiende Krylov-ruimte samenvalt met $\mathbb{C}^m$ (of met een A-invariante deelruimte van $\mathbb{C}^m$ in het geval van het afbreken van de iteratie) en de Ritz waarden gelijk worden aan de eigenwaarden. Het algoritme is echter bedoeld voor grote matrices, waarbij meestal voor een aantal Ritz-waarden convergentie van het residu bereikt wordt tot ordegrootte machineprecisie na slechts $k \ll m$ iteraties.

- De eigenwaarden die in eerste instantie teruggevonden worden zijn *extreme eigenwaarden*, in de betekenis van kwalitatief goed gescheiden eigenwaarden aan de *rand van het spectrum*. Merk dat eigenwaarden aan de rand niet noodzakelijk grootste eigenwaarden in modulus zijn.

Een gedetailleerde analyse van de convergentie-eigenschappen is zeer technisch en buiten de scope van deze cursus, maar we formuleren wel twee stellingen, die inzicht geven in de werking van het algoritme en het tweede punt verduidelijken. We beginnen met een invariantie-eigenschap.

STELLING 9.1.2 *(invariantie voor verschuivingen)*
Als Arnoldi's algoritme, toegepast op het paar (A,b), in de k-de iteratie (met $k \in \{1,\ldots,m\}$), leidt tot Ritz-waarden $\{r_1^{(k)}, \ldots, r_k^{(k)}\}$, dan leidt Arnoldi's algoritme, toegepast op het paar $(A-\sigma I, b)$, met $\sigma \in \mathbb{C}$, tot de verschoven Ritz waarden $\{r_1^{(k)} - \sigma, \ldots, r_k^{(k)} - \sigma\}$.

[3]Een deelruimte $\mathcal{D}$ van $\mathbb{C}^m$ is A-invariant als $A\,\mathcal{D} \subseteq \mathcal{D}$, met $A\,\mathcal{D} = \{Ax : x \in \mathcal{D}\}$.

Bewijs Uit de recursiebetrekking (9.8) volgt

$$(A - \sigma I)Q_k = Q_k(H_k - \sigma I) + q_{k+1}\mathbf{e}_k^T h_{k+1\,k},$$

wat als een recursiebetrekking voor het paar $(A - \sigma I, b)$ geïnterpreteerd kan worden. □

Het korte bewijs is algebraïsch van aard. De achterliggende geometrische reden is dat de opgebouwde Krylov-ruimten door (A, b) en $(A - \sigma I, b)$ dezelfde zijn, in formule

$$\mathcal{K}_k(A - \sigma I, b) = \mathcal{K}_k(A, b),\ \forall k \in \{1, \ldots, m\}.$$

Merk op dat een verschuiving eigenvectoren niet verandert, enkel eigenwaarden. We illustreren de relatie tussen de stelling en de convergentie van Ritz-waarden met een voorbeeld.

Voorbeeld 9.4 *We beschouwen een reële symmetrische matrix A met alle eigenwaarden tussen $\lambda_{\min}$ en $\lambda_{\max}$, met $0 < \lambda_{\min} < \lambda_{\max}$. Enerzijds verwachten we intuïtief, doordat Arnoldi's algoritme de methode van de machten veralgemeent, dat de eigenvector horende bij de grootste eigenwaarde $\lambda_{\max}$ snel gevonden wordt. Anderzijds heeft de verschoven matrix $A - \sigma I$, met bijvoorbeeld $\sigma = \lambda_{\max}$, als grootste eigenwaarde in absolute waarde precies $\lambda_{\min} - \sigma$, waardoor we volgens dezelfde redenering verwachten dat de eigenvector horende bij $\lambda_{\min}$ snel gevonden wordt. Het algoritme van Arnoldi differentieert dus niet tussen $\lambda_{\min}$ en $\lambda_{\max}$, in tegenstelling tot de methode van de machten en deelruimte-iteratie.*

De volgende stelling legt een relatie bloot met veeltermbenadering en kleinste kwadraten.

STELLING 9.1.3 *(relatie met veeltermbenadering)*
Veronderstel dat de Arnoldi-iteratie niet afbreekt en beschouw Q_k, H_k overeenkomstig (9.8). Dan is de karakteristieke veelterm van H_k, $k \in \{1, \ldots, m\}$, de unieke minimizer van het probleem

$$\min_{p \in M_k} \|p(A)b\|_2, \tag{9.10}$$

met M_k de verzameling van alle monische[4] veeltermen van graad k.

Bewijs Als we m iteraties van Algoritme 6 uitvoeren, herleidt (9.8) zich tot $AQ_m = Q_m H_m$, met Q_m een unitaire matrix. We veronderstellen in wat volgt dat $k < m$.
Voor $p \in M_k$ kunnen we $p(A)b$ uitdrukken als $A^k b - Q_k y$, waarbij $y \in \mathbb{C}^k$ uniek bepaald is door p. Dit komt omdat $\mathcal{R}(Q_k) =< b, Ab, \ldots, A^{k-1}b >$. We kunnen (9.10) dus herformuleren als

$$\min_{y \in \mathbb{C}^k} \|A^k b - Q_k y\|_2.$$

Omdat Q_k van volle kolomrang is, heeft dit kleinstekwadraten-probleem een unieke oplossing.

[4]Herinner dat een veelterm monisch is als (in de monomiale basis) de coëfficiënt bij de hoogste macht één is.

Noem $\hat{p}$ de karakteristieke veelterm van H_k en bepaal $\hat{y}$ zodat $\hat{p}(A)b = A^k b - Q_k\hat{y}$. Uit de stelling van Cayley-Hamilton volgt dat $\hat{p}(H_k) = 0$. Bijgevolg is ook

$$Q_k^* Q_m \, \hat{p}(H_m) \, Q_m^* b = 0. \tag{9.11}$$

Immers, $Q_k^* Q_m = [I_k \; 0]$ en $Q_m^* b$ is een niet-nul veelvoud van de eerste eenheidsvector (vermenigvuldigen langs voor met $Q_k^* Q_m$ en langs achter met $Q_m^* b$ levert dus op een constante na de eerste k elementen uit de eerste kolom). Bovendien komen de eerste k elementen uit de eerste kolom van $\hat{p}(H_m)$ en $\hat{p}(H_k)$ overeen. Uit (9.11) volgt dat

$$Q_k^* \left(\hat{p}(Q_m H_m Q_m^*)b\right) = Q_k^* \left(\hat{p}(A)b\right) = 0, \text{ of } Q_k^*(A^k b - Q_k\hat{y}) = 0.$$

Het residu bij $\hat{y}$ is dus orthogonaal met betrekking tot $\mathcal{R}(Q_k)$. Overeenkomstig Stelling 2.4.1 is $\hat{y}$ de oplossing van het kleinstekwadratenprobleem. □

Gebruik makend van (9.3) kunnen we (9.10) herschrijven als

$$\min_{p \in M_k} \left\| X \begin{bmatrix} p(\lambda_1) & & \\ & \ddots & \\ & & p(\lambda_m) \end{bmatrix} X^{-1} b \right\|_2 . \tag{9.12}$$

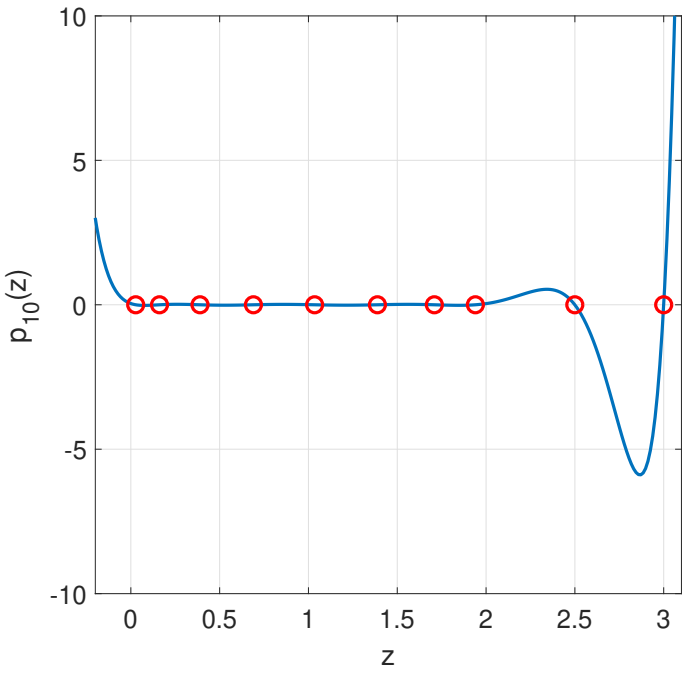
Het bepalen van de eigenwaarden van H_k in de k-de stap van Arnoldi's algoritme kan dus geïnterpreteerd worden als het zoeken naar een monische veelterm van graad k die de doelfunctie in (11.18) minimaliseert. Intuïtief gebeurt dit door er voor te zorgen dat $|p(\lambda_i)|$ klein is voor alle i, wat kan door het spectrum van A zo goed mogelijk af te dekken met de k nulpunten van A. De relatie tot convergentie naar extreme eigenwaarden maken we opnieuw duidelijk met een voorbeeld, waarbij we Voorbeeld 9.3 opnieuw opnemen.

Voorbeeld 9.5 *De matrix heeft geïsoleerde eigenwaarden* 2.5 *en* 3*, alsook een cluster van eigenwaarden tussen* 0 *en* 2*. We beschouwen de* 10*-de iteratie (*$k = 10$*). Een manier om met een monische veelterm* p *van graad* 10 *te voldoen aan* $p(\lambda_i) \approx 0$*,* $i = 1, \ldots, 203$*, bestaat erin om één nulpunt dicht bij* 2.5 *te leggen, één nulpunt dicht bij* 3*, en de andere* 8 *nulpunten zoals Chebyshev-punten te verspreiden over de cluster van eigenwaarden, zodat* $|p(z)|$ *uniform klein is in het interval waarop de cluster ligt. Dit is precies hoe de eigenwaarden van* H_{10}*, de Ritz waarden, gelokaliseerd zijn, zie Figuur 9.1. Na* 10 *iteraties levert Arnoldi's algoritme dus goede benaderingen op van eigenwaarden* 2.5 *en* 3*, alsook Ritz waarden verdeeld over de cluster.*

9.1.3.5 Lanczos' algoritme voor Hermitiaanse matrices

Indien matrix A Hermitiaans is, $A = A^*$, dan zijn matrices H_k, gegenereerd door Arnoldi's algoritme, ook Hermitiaans en tridiagonaal. Dit kan als volgt ingezien worden. Voor $k = m$ reduceert recursiebetrekking (9.8) tot $AQ_m = Q_m H_m$ met $Q_m \in \mathbb{C}^{m \times m}$, waaruit

$$Q_m^* A Q_m = H_m$$



Figuur 9.1: De karakteristieke veelterm van matrix H_{10}, bekomen na 10 iteraties van Arnoldi's algoritme op de matrix gespecificeerd in Voorbeeld 9.3. De Ritz-waarden zijn gemarkeerd.

volgt. Het linkerlid is Hermitiaans omdat $(Q_m^* A Q_m)^* = Q_m^* A^* Q_m = Q_m^* A Q_m$ en dus ook H_m. Een Hermitiaanse Hessenberg-matrix is tridiagonaal. Vermits H_k, met $k < m$, een deelmatrix is van H_m, en meer bepaald een diagonaal blok, is H_k ook tridiagonaal.
Hiermee rekening houdende wordt Algoritme 6 vereenvoudigd tot Algoritme 7. We spreken nu over *Lanczos' algoritme*.

Algoritme 7 Lanczos' algoritme - orthonormale basis van Krylov-ruimte

Require: A Hermitiaans

$q_1 = \frac{b}{\|b\|_2}$
$v_1 = Aq_1$
$h_{11} = q_1^* v_1$
$v_1 = v_1 - h_{11} q_1$
$h_{21} = \|v_1\|_2$
$q_2 = v_1 / h_{21}$
for $j = 2, 3, \ldots$ **do**
  $v_j = Aq_j$
  $h_{j-1\,j} = \overline{h_{j\,j-1}}$
  $h_{jj} = q_j^* v_j$
  $v_j = v_j - h_{j-1\,j} q_{j-1} - h_{jj} q_j$
  $h_{j+1\,j} = \|v_j\|_2$
  $q_{j+1} = v_j / h_{j+1\,j}$
end for

De vereenvoudiging kan ook als volgt geïnterpreteerd worden. Om Av_j te orthogonaliseren ten opzichte van $\{q_1, \ldots, q_j\}$, is het voldoende deze vector te orthogonaliseren ten opzichte van $\{q_{j-1}, q_j\}$.

9.2 Berekenen van alle eigenwaarden

We beschrijven *het* standaard algoritme voor het berekenen van alle eigenwaarden van een matrix, dat bekend staat als het *impliciet verschoven QR-algoritme*. De beschrijving is gebaseerd op [Wat11, Wat08]. Dit algoritme kan rechtstreeks toegepast worden op kleine en middelgrote matrices (dimensie m van ordegrootte $\mathcal{O}(10^3)$). Bij grote matrices kan het algoritme gebruikt worden om het gereduceerde eigenwaardenprobleem bij bijvoorbeeld deelruimte-iteratie of Arnoldi's methode op te lossen (het eigenwaardenprobleem voor matrix H_k in het laatste geval).
Het algoritme bestaat uit twee grote stappen, die besproken worden in §9.2.1 en §9.2.2. In de eerste stap wordt de matrix, door middel van Householder-reflecties, getransformeerd naar een gelijkvormige Hessenberg-matrix. We zullen zien dat dit in zekere zin de structuur is die het dichtst aanleunt bij een bovendriehoeksmatrix (waarbij we eigenwaarden simpelweg kunnen aflezen op de diagonaal). In de tweede stap worden iteraties uitgevoerd om de Hessenberg-matrix stap voor stap om te vormen naar een (blok) bovendriehoeksmatrix. Zoals reeds vermeld ligt de noodzakelijkheid van een iteratief proces in Stelling 9.0.1.

9.2.1 Bepalen van een gelijkvormige Hessenberg-matrix

Om matrix $A \in \mathbb{C}^{m\times m}$ om te vormen naar een gelijkvormige Hessenberg-matrix H,

$$H = \begin{bmatrix} h_{11} & h_{12} & & \cdots & h_{1m} \\ h_{21} & h_{22} & & \cdots & h_{2m} \\ 0 & h_{32} & h_{33} & \cdots & h_{3m} \\ \vdots & \ddots & \ddots & & \vdots \\ 0 & \cdots & 0 & h_{m-1\,m} & h_{mm} \end{bmatrix}, \tag{9.13}$$

moeten we nullen creëren onder de subdiagonaal. Hiervoor kunnen zowel Givens-rotaties als Householder-transformaties gebruikt worden. In wat volgt beperken we ons ons tot de tweede manier, die minder rekenwerk vraagt.
Met de notatie van §2.3.4 bepalen we eerst orthogonale matrix $Q_{2,m}$ zodat matrix $(Q_{2,m}^* A)$ nullen heeft in de eerste kolom en onder de eerste subdiagonaal. Omdat de eigenwaarden niet mogen veranderen, moeten we vervolgens langs rechts vermenigvuldigen met $\left(Q_{2,m}^*\right)^{-1} = Q_{2,m}$. Immers A en $Q_{2,m}^* A Q_{2,m}$ zijn gelijkvormig, wat niet gezegd kan worden van A en $Q_{2,m}^* A$. Vervolgens creëren we nullen in de tweede kolom, en zetten dit proces verder tot een matrix van de vorm (9.13) bekomen wordt. Bij een 4-bij-4 matrix is bijvoorbeeld de flow van bewerkingen om tot een Hessenberg-matrix te komen als volgt:

$$A = \begin{bmatrix} x & x & x & x \\ x & x & x & x \\ x & x & x & x \\ x & x & x & x \end{bmatrix} \begin{matrix} Q_{2,4}^*\times \\ \rightarrow \end{matrix} \begin{bmatrix} x & x & x & x \\ \mathbf{x} & \mathbf{x} & \mathbf{x} & \mathbf{x} \\ 0 & \mathbf{x} & \mathbf{x} & \mathbf{x} \\ 0 & \mathbf{x} & \mathbf{x} & \mathbf{x} \end{bmatrix} \begin{matrix} \times Q_{2,4} \\ \rightarrow \end{matrix} \begin{bmatrix} x & \mathbf{x} & \mathbf{x} & \mathbf{x} \\ x & \mathbf{x} & \mathbf{x} & \mathbf{x} \\ 0 & \mathbf{x} & \mathbf{x} & \mathbf{x} \\ 0 & \mathbf{x} & \mathbf{x} & \mathbf{x} \end{bmatrix}$$

$$\begin{matrix} Q_{3,4}^*\times \\ \rightarrow \end{matrix} \begin{bmatrix} x & x & x & x \\ x & x & x & x \\ 0 & \mathbf{x} & \mathbf{x} & \mathbf{x} \\ 0 & 0 & \mathbf{x} & \mathbf{x} \end{bmatrix} \begin{matrix} \times Q_{3,4} \\ \rightarrow \end{matrix} \begin{bmatrix} x & x & \mathbf{x} & \mathbf{x} \\ x & x & \mathbf{x} & \mathbf{x} \\ 0 & x & \mathbf{x} & \mathbf{x} \\ 0 & 0 & \mathbf{x} & \mathbf{x} \end{bmatrix}.$$

We kunnen dit samenvatten als

$$Q_{3,4}^* Q_{2,4}^* A Q_{2,4} Q_{3,4} = H \ \rightarrow A = (Q_{2,4}Q_{3,4}) H (Q_{2,4}Q_{3,4})^* := QHQ^*.$$

Merk op dat de overgang van A naar H via de relatie QHQ^* een gelijkvormigheidstransformatie is, die de eigenwaarden bewaart.
Doordat we *tweezijdige* operaties uitvoeren kunnen we matrix A niet rechtstreeks herleiden tot een bovendriehoeksmatrix. Door vooraan te vermenigvuldigen kunnen we weliswaar nullen creëren direct onder de diagonaal, maar deze gaan verloren na de overeenkomstige vermenigvuldiging achteraan.
Gebruik makend van formule (2.39) kunnen we de geschetste procedure omzetten tot Algoritme 8. Stappen 6-9 implementeren de pre-multiplicatie met $Q_{j+1,m}^*$, stappen 10-12 de post-multiplicatie met $Q_{j+1,m}$. Indien matrix Q in expliciete vorm gewenst is, moeten stappen 14-20 nog uitgevoerd worden. Zij zijn gelijkaardig als deze in Algoritme 4.

De rekenkost van Algoritme 8 bedraagt $\sim \frac{10}{3}m^3$ elementaire bewerkingen als enkel H gewenst is en $\sim \frac{16}{3}m^3$ bewerkingen als zowel H als Q in expliciete vorm gewenst zijn.

Algoritme 8 Impliciet verschoven QR-algoritme - gelijkvormige Hessenberg-matrix via Householder-transformaties

$H = A$
for $j = 1$ to $m - 2$ **do**
  $x = H_{j+1:m,j}$
  $v_j = x + \text{sign}(x_1)\|x\|_2\, \mathbf{e}_1$
  $v_j = v_j/\|v_j\|_2$
  $H_{j+1\,j} = -\text{sign}(x_1)\|x\|_2,\ H_{j+2:m,j} = 0$
  for $k = j + 1$ to m **do**
    $H_{j+1:m,k} = H_{j+1:m,k} - 2\left(v_j^*\, H_{j+1:m,k}\right) v_j$
  end for
  for $k = 1$ to m **do**
    $H_{k,j+1:m} = H_{k,j+1:m} - 2\left(H_{k,j+1:m} v_j\right) v_j^*$
  end for
end for
for $i = 1$ to m **do**
  $w = \mathbf{e}_i$
  for $k = m - 2$ downto 1 **do**
    $w_{k+1:m} = w_{k+1:m} - 2\left(v_k^*\, w_{k+1:m}\right) v_k$
  end for
  $Q_{1:m,i} = w$
end for

Opmerking 9.3 *Hessenberg-matrix H kan ook bekomen worden door m iteraties van Arnoldi's algoritme, Algoritme 6, uit te voeren. Immers, voor $k = m$ breekt de recursie (9.8) af en krijgen we $AQ_m = Q_m H_m$, oftewel $Q_m^* A Q_m = H_m$. Algoritme 8 is echter numeriek stabieler.*
In feite is de situatie voor matrix transformaties volledig analoog aan deze bij de QR-factorisatie. In de setting van Sectie 2.3: Wenst men een volledige QR-factorisatie uit te voeren, dan is een algoritme gebaseerd op Givens-rotaties of Householder-reflecties aangewezen. Wenst men slechts n vectoren te orthogonaliseren, met $n \ll m$, is een Gram-Schmidt gebaseerd algoritme aangewezen (en onvermijdbaar indien alle vectoren niet op voorhand gekend zijn). In de setting van dit hoofdstuk: Wenst men een matrix volledig te herleiden tot een gelijkvormige Hessenberg-matrix, dan ligt Algoritme 8 voor de hand. Kan of wenst men slechts een $k \times k$ submatrix H_k van H en de eerste k kolommen van Q, vervat in Q_k, te bepalen , met $k \ll m$, dan ligt het Gram-Schmidt gebaseerde algoritme van Arnoldi voor de hand.

We sluiten af met een voorbeeld.

Voorbeeld 9.6 *We beschouwen matrix*

$$A = \begin{bmatrix} 3.354814 & 0.197482 & -1.720636 & 1.209042 \\ 5.141004 & 1.662580 & 2.565090 & 3.838912 \\ -3.188632 & 0.355978 & 1.217444 & -2.270835 \\ 3.950505 & -0.527946 & 4.301668 & 4.765161 \end{bmatrix}, \tag{9.14}$$

waarvoor geldt

$$Q_{2,4}^* A = \begin{bmatrix} 3.354814 & 0.197482 & -1.720636 & 1.209043 \\ -7.225218 & -0.737223 & -3.639878 & -6.339118 \\ 0 & 0.974767 & 2.817396 & 0.353571 \\ 0 & -1.294586 & 2.319433 & 1.513695 \end{bmatrix},$$

$$Q_{2,4}^* A Q_{2,4} = \begin{bmatrix} 3.354814 & -1.560930 & -1.267230 & 0.647301 \\ -7.225218 & 2.384225 & -4.444744 & -5.341942 \\ 0 & 0.356470 & 2.976824 & 0.156050 \\ 0 & 1.117119 & 1.697575 & 2.284137 \end{bmatrix},$$

$$Q_{3,4}^* Q_{2,4}^* A Q_{2,4} = \begin{bmatrix} 3.354814 & -1.560930 & -1.267229 & 0.647301 \\ -7.225218 & 2.384225 & -4.444744 & -5.341942 \\ 0 & -1.172615 & -2.522176 & -2.223475 \\ 0 & 0 & -2.319885 & 0.545703 \end{bmatrix},$$

$$Q_{3,4}^* Q_{2,4}^* A Q_{2,4} Q_{3,4} = \begin{bmatrix} 3.354814 & -1.560930 & -0.231434 & 1.404033 \\ -7.225218 & 2.384225 & 6.440310 & 2.610461 \\ 0 & -1.172615 & 2.884976 & 1.726883 \\ 0 & 0 & 0.185358 & 2.375984 \end{bmatrix} := H.$$

9.2.2 Iteratief bepalen van de eigenwaarden

Vertrekkende van een Hessenberg-matrix leggen we uit hoe stap voor stap de eigenwaarden bepaald kunnen worden. Eerst behandelen we in §9.2.2.1 een variant van deelruimte-iteratie die *as such* niet gebruikt wordt, maar ons wel zal toelaten om de werking van het impliciet verschoven QR-algoritme te begrijpen.

9.2.2.1 Preliminaire resultaten

We beschouwen ℓ zogenaamde verschuivingen $\rho_i \in \mathbb{C},\ i = 1, \ldots, \ell$, met ℓ een klein natuurlijk getal (bijvoorbeeld 1, 2 of 4). Met deze verschuivingen, waarvan we de rol later duidelijk maken, definiëren we de veelterm

$$p(z) = \Pi_{i=1}^{\ell}(z - \rho_i).$$

We passen nu het algoritme van deelruimte-iteratie aan op twee manieren. Ten eerste itereren we op de volledige ruimte (waarbij $n = m$), ten tweede itereren we niet met A, maar met

$$p(A) = (A - \rho_1 I)(A - \rho_2 I) \cdots (A - \rho_\ell I).$$

Dit resulteert in het volgende algoritme. Startend met beginmatrix $\underline{Q}_0 = I_m$, berekenen we matrices $\underline{Q}_k = [q_1^{(k)} \cdots\ q_m^{(k)}]$ voor $k \geq 1$, via de iteratie

$$\begin{aligned} &P_k = p(A)\, \underline{Q}_{k-1}, \\ &\underline{Q}_k \underline{R}_k = P_k, \quad k = 1, 2, 3, \ldots \end{aligned} \tag{9.15}$$

De gegenereerde vectoren en matrices voldoen aan

$$\begin{aligned} < q_1^{(k)}, \ldots, q_j^{(k)} > &= < (p(A))^k \mathbf{e}_1, \ldots, (p(A))^k \mathbf{e}_j > \\ &= (p(A))^k < \mathbf{e}_1, \ldots, \mathbf{e}_j >, \quad j = 1, \ldots, m, \end{aligned} \tag{9.16}$$

met $\mathbf{e}_i,\ i \in \{1, \ldots, m\}$, de i-de eenheidsvector of de i-de kolom van $\underline{Q}_0$.

Met (9.1) de eigenwaardenontbinding van A, geldt

$$p(A) = X \begin{bmatrix} p(\lambda_1) & 0 & \cdots & 0 \\ 0 & p(\lambda_2) & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & p(\lambda_m) \end{bmatrix} X^{-1},$$

dus, in overeenstemming met het zgn. spectral mapping theorem, blijven de eigenvectoren dezelfde, terwijl de functie p individueel op de eigenwaarden wordt toegepast. Nu kunnen we een betekenis geven aan de verschuivingen.

Stel dat de ℓ verschuivingen goede schattingen zijn van precies ℓ eigenwaarden van A. Dan is $|p(z)|$ geëvalueerd in deze eigenwaarden klein, maar niet in de andere eigenwaarden. Als we dan een *alternatieve ordening* van eigenwaarden veronderstellen zodat

$$|p(\lambda_1)| \geq |p(\lambda_2)| \geq \cdots \geq |p(\lambda_m)|,$$

komen $\lambda_{m-\ell+1}, \ldots, \lambda_m$ overeen met de ℓ eigenwaarden dichtst bij de verschuivingen, en $\lambda_1, \ldots, \lambda_{m-\ell}$ met de andere eigenwaarden. Voeren we nu iteraties uit van (9.15), dan zal

$$\lim_{k\to\infty} < q_1^{(k)}, \ldots, q_{m-\ell}^{(k)} > = < x_1, \ldots, x_{m-\ell} >,$$

met convergentiefactor

$$\rho = \left| \frac{p(\lambda_{m-\ell+1})}{p(\lambda_{m-\ell})} \right|.$$

Dus, des te beter de verschuivingen benaderingen zijn van eigenwaarden, des te kleiner ρ, en des te sneller de convergentie. Eens convergentie bereikt is, kunnen we de ℓ eigenwaarden $\lambda_{m-\ell+1}, \ldots, \lambda_m$ met volle precisie berekenen, wat volgt uit volgende stelling waarvan we het bewijs als oefening laten.

STELLING 9.2.4 *(afsplitsen van ℓ eigenwaarden)*
Beschouw matrix $A \in \mathbb{C}^{m\times m}$ met eigenvectoren $x_1, \ldots, x_m$, horende bij eigenwaarden $\lambda_1, \ldots, \lambda_m$, en $\ell \in \mathbb{N}$, $\ell < m$. Zij $Q \in \mathbb{C}^{m\times m}$ een unitaire matrix, waarvan de eerste $m - \ell$ kolommen een basis vormen van $< x_1, \ldots, x_{m-\ell} >$, dan is

$$Q^* A Q = \begin{bmatrix} S_{11} & S_{12} \\ 0 & S_{22} \end{bmatrix}, \tag{9.17}$$

met $S_{11} \in \mathbb{C}^{(m-\ell)\times(m-\ell)}$, $S_{12} \in \mathbb{C}^{(m-\ell)\times\ell}$ en $S_{22} \in \mathbb{C}^{\ell\times\ell}$.

Uit ontbinding (9.17) volgt dat de eigenwaarden van A overeenkomen met deze van S_{11}, samengevoegd met deze van S_{22}. Omdat ℓ zeer klein is, kunnen we deze laatste rechtstreeks bepalen. We zouden dan de hele procedure kunnen herhalen met matrix S_{11} waarvan de dimensies ten opzichte van A met ℓ afgenomen zijn, weer een blok met ℓ eigenwaarden afscheiden enz. In §9.2.2.2 en §9.2.2.3 beschrijven we een proces dat hiermee wiskundig equivalent is, maar veel minder rekenwerk vraagt. Het buit sterk uit dat de matrix al op voorhand in Hessenberg-vorm gebracht is, die over de iteraties behouden blijft.

9.2.2.2 Eén iteratiestap van het impliciet verschoven QR-algoritme

Gegeven matrix $A^{(k-1)}$ in Hessenberg-vorm, leggen we uit hoe matrix $A^{(k)}$, ook in Hessenberg-vorm, berekend wordt. We gaan er van uit dat $m > \ell$.
We kiezen ℓ verschuivingen $\rho_1^{(k)}, \ldots, \rho_\ell^{(k)}$ en definiëren $p^{(k)}(z) = \left(z - \rho_1^{(k)}\right) \cdots \left(z - \rho_\ell^{(k)}\right)$.
We starten met het berekenen van $x = \gamma\, p^{(k)}(A^{(k-1)})\mathbf{e}_1$, met $\mathbf{e}_1$ de eerste eenheidsvector van compatibele dimensie en $\gamma > 0$ een schaalfactor:

$$x = \gamma \left(A^{(k-1)} - \rho_1^{(k)} I\right)\left(A^{(k-1)} - \rho_2^{(k)} I\right) \cdots \left(A^{(k-1)} - \rho_\ell^{(k)} I\right)\mathbf{e}_1,$$

waarbij we γ kiezen zodat $\|x\|_2 = 1$. Door de Hessenberg-structuur van $A^{(k-1)}$ zijn enkel de eerste $\ell + 1$ elementen van x verschillend van nul, die een vector $\tilde{x} \in \mathbb{C}^{\ell+1}$ vormen. We bepalen vervolgens een Householder-transformatie $Q_{1,\ell+1} \in \mathbb{C}^{(\ell+1)\times(\ell+1)}$ zodat

$$Q_{1,\ell+1}^* \tilde{x} = \mathbf{e}_1.$$

We stellen dan

$$Q_1 = \begin{bmatrix} Q_{1,\ell+1} & 0 \\ 0 & I_{m-(\ell+1)} \end{bmatrix}$$

en berekenen de volgende matrix gelijkvormig met $A^{(k-1)}$:

$$Q_1^* A^{(k-1)} Q_1.$$

Merk dat matrix Q_1^* combinaties maakt van de eerste $\ell + 1$ rijen en Q_1 van de eerste $\ell + 1$ kolommen. Voor $m > \ell + 1$ zijn daarom in het algemeen de elementen uit de submatrix bestaande uit de eerste $\ell + 2$ rijen en kolommen van $Q_1^* A^{(k-1)} Q_1$ verschillend van nul, wat de Hessenberg-structuur verstoort. We voeren daarom een tweede transformatie door, beschreven door unitaire matrix Q_2, die de Hessenberg-structuur herstelt, wat resulteert in

$$A^{(k)} = Q_2^* Q_1^* A^{(k-1)} Q_1 Q_2.$$

Je kan nagaan dat je hiervoor Q_2 kan kiezen in de vorm van opeenvolgende Householder-transformaties:

$$Q_2 = Q_{2,\underline{\ell+2}}\, Q_{3,\underline{\ell+3}}\, Q_{4,\underline{\ell+4}} \cdots Q_{m-1,m},$$

met $\underline{k} = \min(k, m)$, waarbij $Q_{2,\underline{\ell+2}}$ nullen creëert in de eerste kolom, $Q_{3,\underline{\ell+3}}$ in de tweede kolom enz. Omdat Q_2 niet inwerkt op de eerste rij, geldt $Q_2\mathbf{e}_1 = \mathbf{e}_1$.
Stellen we $Q^{(k)} = Q_1Q_2$, dan voldoet de totale transformatie-matrix aan

$$Q^{(k)}\mathbf{e}_1 = Q_1(Q_2\mathbf{e}_1) = Q_1\mathbf{e}_1 = \begin{bmatrix} Q_{1,\ell+1}\mathbf{e}_1 \\ 0 \\ \vdots \\ 0 \end{bmatrix} = \begin{bmatrix} \tilde{x} \\ 0 \\ \vdots \\ 0 \end{bmatrix} = x = p^{(k)}(A^{(k-1)})\mathbf{e}_1.$$

In woorden, de eerste kolom van $Q^{(k)}$ is gelijk aan $p^{(k)}(A^{(k-1)})\mathbf{e}_1$.

Laten we even recapituleren. We hebben Hessenberg-matrix $A^{(k-1)}$ omgevormd tot matrix

$$A^{(k)} = {Q^{(k)}}^* A^{(k-1)} Q^{(k)}, \tag{9.18}$$

met de volgende eigenschappen:

- Matrix $A^{(k)}$ is gelijkvormig met $A^{(k-1)}$ en heeft bijgevolg dezelfde eigenwaarden.
- Matrix $A^{(k)}$ is ook in Hessenberg-vorm.
- De eerste kolom van $Q^{(k)}$ is gelijk aan $p^{(k)}(A^{(k-1)})\mathbf{e}_1$.

Voorbeeld 9.7 *We gaan verder met Voorbeeld 9.6. We stellen $A^{(0)} = H$, kiezen $\ell = 1$ en*

$$p^{(1)}(z) = (z - H_{44}) = z - 2.375984.$$

De verschuiving is dus het element onderaan rechts in matrix $A^{(0)}$. Dan is

$$x = \begin{bmatrix} 0.134248 \\ -0.990948 \\ 0 \\ 0 \end{bmatrix}, \quad Q_1^T A^{(0)} Q_1 = \begin{bmatrix} 3.570562 & -7.195990 & -6.413080 & -2.398342 \\ -1.531702 & 2.168478 & -0.635257 & -1.741772 \\ 1.162000 & 0.157421 & 2.884976 & 1.726883 \\ 0 & 0 & 0.185358 & 2.375984 \end{bmatrix}.$$

De Hessenberg-structuur is inderdaad verloren gaan. Om een tweede nul te creëren in de eerste kolom voeren we een Householder-transformatie uit:

$$Q_{2,3}^T \left(Q_1^T A^{(0)} Q_1\right) Q_{2,3} = \begin{bmatrix} 3.570562 & 1.856928 & -9.458419 & -2.398342 \\ 1.922591 & 2.660293 & 0.805711 & 2.431362 \\ 0 & 0.013032 & 2.393162 & 0.323070 \\ 0 & 0.112029 & 0.147672 & 2.375984 \end{bmatrix}.$$

Een bijwerking is echter dat we in de tweede kolom een nul kwijt zijn. De uitstulping ('bulge'*) ten opzichte van de Hessenberg-vorm is dus verplaatst langs de diagonaal. We kunnen dit remediëren met een tweede Householder-transformatie:*

$$\begin{aligned} A^{(1)} &= \underbrace{Q_{3,4}^T Q_{2,3}^T}_{Q_2^T} \left(Q_1^T A^{(0)} Q_1\right) \underbrace{Q_{2,3} Q_{3,4}}_{Q_2} \\ &= \begin{bmatrix} 3.570562 & 1.856928 & 3.475144 & 9.117955 \\ 1.922590 & 2.660293 & -2.508172 & -0.519385 \\ 0 & -0.112785 & 2.430241 & 0.143359 \\ 0 & 0 & 0.318757 & 2.338905 \end{bmatrix}. \end{aligned}$$

De totale transformatie wordt beschreven door

$$Q^{(1)} = Q_1 Q_2 = \begin{bmatrix} 0.134248 & 0.789475 & 0.069202 & 0.594910 \\ -0.990948 & 0.106953 & 0.009375 & 0.080595 \\ 0 & 0.604393 & -0.092052 & -0.791351 \\ 0 & 0 & -0.993302 & 0.115544 \end{bmatrix},$$

met de eerste kolom gelijk aan $x = p^{(1)}\left(A^{(0)}\right)\mathbf{e}_1$.

9.2.2.3 Afsplitsen van eigenwaarden

De volgende stelling zegt dat matrix $Q^{(k)}$, gegenereerd tijdens de k-de iteratiestap van het impliciet verschoven QR-algoritme, geïnterpreteerd kan worden als een stap van de variant van deelruimte-iteratie beschreven in §9.2.2.1.

STELLING 9.2.5 *(relatie met deelruimte-iteratie)*
Zij Hessenberg-matrices $A^{(k)},\ A^{(k-1)} \in \mathbb{C}^{m\times m}$ *en unitaire matrix* $Q^{(k)} \in \mathbb{C}^{m\times m}$ *gegeven die voldoen aan (9.18) en waarbij de eerste kolom van* $Q^{(k)}$ *gelijk is aan* $p^{(k)}(A^{(k-1)})\mathbf{e}_1$. *Veronderstel dat alle elementen van* $A^{(k)}$ *net onder de diagonaal verschillend zijn van nul. Dan voldoet matrix* $Q^{(k)} = \left[q_1^{(k)}\ q_2^{(k)}\ \cdots\ q_m^{(k)}\right]$ *aan*

$$< q_1^{(k)}, \ldots, q_j^{(k)} >= p^{(k)}(A^{(k-1)}) < \mathbf{e}_1, \ldots, \mathbf{e}_j >, \quad \forall j \in \{1, \ldots, m\},$$

met $\mathbf{e}_i,\ i \in \{1, \ldots, m\}$, *de* i*-de eenheidsvector in* $\mathbb{C}^m$.

Bewijs Omdat matrix $A^{(k)}$ een Hessenberg-matrix is en men (9.18), geschreven als

$$A^{(k-1)}Q^{(k)} = Q^{(k)}A^{(k)},$$

kan interpreteren als recursiebetrekking van een Arnoldi-iteratie met matrix $A^{(k-1)}$, is er voor elke $j \in \{1, \ldots, m\}$ voldaan aan

$$< q_1^{(k)}, \ldots, q_j^{(k)} >= \mathcal{K}_j\left(A^{(k-1)}, q_1^{(k)}\right),$$

zie Definitie 9.7 van een Krylov-ruimte. Verder is

$$\begin{aligned} \mathcal{K}_j\left(A^{(k-1)}, q_1^{(k)}\right) &= \mathcal{K}_j\left(A^{(k-1)}, p^{(k)}(A^{(k-1)})\mathbf{e}_1\right) \\ &= p^{(k)}(A^{(k-1)})\ \mathcal{K}_j\left(A^{(k-1)}, \mathbf{e}_1\right), \end{aligned}$$

waarbij we gebruik maakten van de eigenschap $A^{(k-1)}p^{(k)}\left(A^{(k-1)}\right) = p^{(k)}\left(A^{(k-1)}\right)A^{(k-1)}$. Tenslotte geldt omwille van de Hessenberg-structuur van $A^{(k-1)}$, met elementen op posities $(2,1), \ldots, (m, m-1)$ verschillend van nul, dat

$$\mathcal{K}_j(A^{(k-1)}, \mathbf{e}_1) =< \mathbf{e}_1, \ldots, \mathbf{e}_j > .$$

Leggen we de drie gelijkheden samen, volgt het gestelde. □

Kijken we naar het *gezamenlijk effect* van iteraties $1, 2, \ldots, k$, dan zien we

$$\begin{aligned} A^{(k)} &= Q^{(k)^*} A^{(k-1)} Q^{(k)} \\ &= Q^{(k)^*} Q^{(k-1)^*} A^{(k-2)} Q^{(k-1)} Q^{(k)} \\ &\vdots \\ &= \mathbf{Q}^{(k)^*} A^{(0)} \mathbf{Q}^{(k)}, \end{aligned}$$

met $\mathbf{Q}^{(k)} = Q^{(1)} \cdots Q^{(k)}$.
Stellen we nu $\mathbf{Q}^{(k)} = \left[\mathbf{q}_1^{(k)} \ \cdots \ \mathbf{q}_m^{(k)}\right]$ voor $k \geq 1$, dan geldt voor elke $j \in \{1, \ldots, m\}$ dat

$$\begin{aligned} < \mathbf{q}_1^{(k)} \ \cdots \ \mathbf{q}_j^{(k)} > &= \mathbf{Q}^{(k-1)} < q_1^{(k)}, \ldots, q_j^{(k)} > \\ &= \mathbf{Q}^{(k-1)} p^{(k)}(A^{(k-1)}) < \mathbf{e}_1, \ldots, \mathbf{e}_j >, \end{aligned}$$

waarbij we Stelling 9.2.5 toegepast hebben. Verder weten we dat

$$p^{(k)}(A^{(k-1)}) = p^{(k)}\left(\mathbf{Q}^{(k-1)^*} A^{(0)} \mathbf{Q}^{(k-1)}\right) = \mathbf{Q}^{(k-1)^*} \ p^{(k)}\left(A^{(0)}\right) \ \mathbf{Q}^{(k-1)}.$$

Vullen we dit in en herhalen we dezelfde argumenten, dan vinden we

$$\begin{aligned} < \mathbf{q}_1^{(k)}, \ldots, \mathbf{q}_j^{(k)} > &= p^{(k)}\left(A^{(0)}\right) \mathbf{Q}^{(k-1)} < \mathbf{e}_1, \ldots, \mathbf{e}_j > \\ &= p^{(k)}\left(A^{(0)}\right) < \mathbf{q}_1^{(k-1)}, \ldots, \mathbf{q}_j^{(k-1)} > \\ &= p^{(k)}\left(A^{(0)}\right) p^{(k-1)}\left(A^{(0)}\right) < \mathbf{q}_1^{(k-2)}, \ldots, \mathbf{q}_j^{(k-2)} > \\ &\vdots \\ &= p^{(k)}\left(A^{(0)}\right) \cdots p^{(2)}\left(A^{(0)}\right) < \mathbf{q}_1^{(1)}, \ldots, \mathbf{q}_j^{(1)} > \\ &= p^{(k)}\left(A^{(0)}\right) \cdots p^{(2)}\left(A^{(0)}\right) p^{(1)}\left(A^{(0)}\right) < \mathbf{e}_1, \ldots, \mathbf{e}_j >, \\ & \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad j \in \{1, 2, \ldots, m\}. \end{aligned}$$

Een vergelijking met (9.16) leert dat we impliciet deelruimte-iteratie toepassen op de volledige ruimte $\mathbb{C}^m$, met wat extra flexibiliteit ingebouwd doordat we verschuivingen (en dus de functies $p^{(k)}$) niet constant veronderstellen over de iteraties.

Uit de Analyse in §9.2.2.1 volgt dat als de verschuivingen goede benaderingen zijn van precies ℓ eigenwaarden, er snel convergentie optreedt van $< \mathbf{q}_1^{(k)}, \ldots, \mathbf{q}_{m-\ell}^{(k)} >$ naar $< x_1, \ldots, x_{m-\ell} >$, wat ons volgens Stelling 9.2.4 toelaat om uiteindelijk een blok van ℓ eigenwaarden rechts onderaan af te splitsen. Wanneer convergentie bijna bereikt is, is $A^{(k-1)}$ bij benadering een blok-bovendriehoeksmatrix, waardoor de eigenwaarden van de deelmatrix gevormd door de laatste ℓ rijen en laatste ℓ kolommen goede benaderingen zijn van eigenwaarden van $A^{(k-1)}$, en dus van $A^{(0)}$. De eigenwaarden van deze deelmatrix kunnen als nieuwe verschuivingen gebruikt worden, en doordat in de volgende iteratiestap de verschuivingen een update kunnen krijgen op basis van dezelfde deelmatrix van $A^{(k)}$, kan de convergentie nog versneld worden. Het mooie aan het QR-algoritme is dat ook wanneer er geen goede schattingen van eigenwaarden voorhanden zijn, deze keuze voor $\ell = 1$ of $\ell = 2$ bijna altijd leidt tot het afsplitsen van eigenwaarden. Dit brengt ons tot Algoritme 9.

De voorwaarde op lijn 4 drukt uit dat een blok rechts onderaan kan afgesplitst worden. Uit de bovenstaande analyse verwachten we dat zulk blok dimensies ℓ-bij-ℓ heeft, maar het kan in de praktijk ook voorkomen dat het afgesplitste blok kleiner of groter is (het eerste geval zullen we illustreren en verklaren met Voorbeeld 9.9). Vandaar dat niet alleen naar de grootte van $|a_{m-\ell+1\ m-\ell}|$ gekeken wordt. Na het afsplitsen herleidt het bepalen van de eigenwaarden zich tot het bepalen van de eigenwaarden van de twee diagonale blokken. Heeft zulk diagonaal blok dimensies ℓ-bij-ℓ of kleiner, kunnen we deze rechtstreeks bepalen. In het andere geval passen we Algoritme 9 toe op dat diagonale blok, waarvan de dimensies kleiner zijn dan deze van de originele matrix. Zo ontstaat een recursief proces waarbij het originele eigenwaardenprobleem uiteindelijk opgesplitst is in dat van matrices met maximale dimensie ℓ-bij-ℓ. We geven twee voorbeelden.

Algoritme 9 Impliciet verschoven QR-algoritme - afsplitsen van eigenwaarden

1: **input** Hessenberg-matrix $A^{(0)} \in \mathbb{C}^{m\times m}$, $\ell \in \mathbb{N},\ \ell < m$, tolerantie $\epsilon > 0$
2: **for** $k = 1, 2, \ldots$ **do**
3: Bepaal matrix $A^{(k)} = [a_{ij}]_{i,j=1}^{m}$ uit $A^{(k-1)}$ via de procedure in §9.2.2.2
4: **if** $(|a_{21}| < \epsilon\|A\|_F \vee\ |a_{32}| < \epsilon\|A\|_F \vee\ \cdots \vee\ |a_{m\,m-1}| < \epsilon\|A\|_F)$ **then**
5: **break**
6: **end if**
7: **end for**

Voorbeeld 9.8 *We gaan verder met Voorbeeld 9.7. We hebben in feite reeds één iteratie gedaan van Algoritme 9 voor $\ell = 1$. Het uitvoeren van meerdere iteraties leidt tot onder andere*

$$A^{(3)} = \begin{bmatrix} 4.9861e{+}00 & 1.8075e{-}01 & 3.9614e{+}00 & 6.3183e{+}00 \\ 4.6709e{-}01 & 1.1410e{+}00 & -5.5859e{+}00 & -3.8235e{+}00 \\ 0 & -3.6989e{-}02 & 2.8775e{+}00 & 3.7978e{-}01 \\ 0 & 0 & -8.9351e{-}03 & 1.9954e{+}00 \end{bmatrix}$$

en

$$A^{(6)} = \left[\begin{array}{cccc} 4.9997e{+}00 & -1.7576e{-}01 & -3.1907e{+}00 & 5.8076e{+}00 \\ -1.4977e{-}02 & 8.6702e{-}01 & 5.9211e{+}00 & -4.6191e{+}00 \\ 0 & -4.8244e{-}02 & 3.1333e{+}00 & -5.9405e{-}01 \\ \hline 0 & 0 & 1.9477e{-}19 & 2.0000e{+}00 \end{array}\right]. \tag{9.19}$$

Opeenvolgende waarden van het element op positie $(4, 3)$ zijn

$$3.1876\text{e}{-}01,\ 2.7305\text{e}{-}01,\ -8.9351\text{e}{-}03,\ 3.5613\text{e}{-}05,\ -6.4951\text{e}{-}10,\ 1.9477\text{e}{-}19,$$

wat duidt op kwadratische convergentie naar nul.
In (9.19) hebben we dus de eigenwaarde 2 afgesplitst. Om de andere eigenwaarden te bepalen kunnen we Algoritme 9 toepassen op de 3-bij-3 deelmatrix links boven in $A^{(6)}$, die we aanduiden met $B^{(0)}$. Opeenvolgende iteraties met $\ell = 1$ leiden tot

$$B^{(1)} = \begin{bmatrix} 4.9997e{+}00 & -2.3422e{-}01 & 3.2004e{+}00 \\ -4.7067e{-}03 & 9.9222e{-}01 & -5.9592e{+}00 \\ 0 & 2.7150e{-}03 & 3.0080e{+}00 \end{bmatrix}$$

en

$$B^{(4)} = \left[\begin{array}{rr|r} 5.0003e{+}00 & -2.2910e{-}01 & 3.2142e{+}00 \\ 4.7652e{-}03 & 9.9973e{-}01 & -5.9543e{+}00 \\ \hline 0 & 4.4834e{-}20 & 3.0000e{+}00 \end{array}\right]. \tag{9.20}$$

Opnieuw hebben we een eigenwaarde afgesplitst, namelijk 3*. Passen we ten slotte Algoritme 9 met* $\ell = 1$ *toe op* $C^{(0)}$*, de* 2*-bij-*2 *deelmatrix links boven in* $B^{(4)}$*, krijgen we al na één iteratiestap*

$$C^{(1)} = \left[\begin{array}{r|r} 5.0000e{+}00 & -2.3386e{-}01 \\ \hline 1.5443e{-}15 & 1.0000e{+}00 \end{array}\right].$$

De laatste twee eigenwaarden zijn bijgevolg 1 *en* 5*.*
We besluiten dat het spectrum van matrix (9.14) bestaat uit eigenwaarden $\{1, 2, 3, 5\}$*.*

Systematisch kiezen voor $\ell = 1$ in Algoritme 9 is te restrictief voor een robuust algemeen toepasbaar eigenwaardenalgoritme. Een reële niet-symmetrische matrix kan bijvoorbeeld paren hebben van complex toegevoegde eigenwaarden. Voor $\ell = 1$ is de verschuiving dan een reëel getal, en bijgevolg kan een eigenwaarde behorend tot zulk paar niet snel afgesplitst worden. Het invoeren van een complexe verschuiving kan dit probleem oplossen, maar heeft als nadeel dat vanaf dan alle bewerkingen met complexe getallen moeten gebeuren, en het buit de symmetrie van het spectrum ten opzichte van de reële as niet uit. Een betere aanpak bestaat erin om Algoritme 9 toe te passen met $\ell = 2$ (of hoger), wat toelaat om een 2-bij-2 matrix af te splitsen waarvan de eigenwaarden zowel reëel kunnen zijn als een complex toegevoegd paar.

Voorbeeld 9.9 *We beschouwen matrix* $A = A^{(0)}$*, reeds in Hessenberg vorm,*

$$A^{(0)} = \begin{bmatrix} -5.4499e{+}00 & 1.0553e{+}00 & -1.1841e{+}00 & -3.0171e{-}01 & 2.7492e{+}00 \\ -4.1239e{+}01 & 8.9965e{+}00 & -1.6655e{+}01 & 1.1153e{+}00 & 1.1521e{+}01 \\ 0 & 7.9402e{-}01 & -1.0598e{+}00 & 2.6791e{+}00 & 4.3793e{+}00 \\ 0 & 0 & -1.4016e{+}00 & -1.6771e{+}00 & -1.0108e{+}01 \\ 0 & 0 & 0 & 7.1803e{-}01 & 5.1902e{+}00 \end{bmatrix}.$$

We passen Algoritme 9 toe met $\ell = 2$*, wat leidt tot*

$$A^{(5)} = \left[\begin{array}{rrrr|r} 5.1655e{-}02 & -2.4177e{+}01 & -1.6917e{+}00 & -3.8503e{+}01 & -9.6964e{+}00 \\ 5.3419e{-}01 & 3.9335e{+}00 & 1.9082e{+}00 & 8.0128e{+}00 & -8.3983e{+}00 \\ 0 & 1.2689e{-}01 & -3.0737e{-}01 & 4.4849e{+}00 & 5.4575e{+}00 \\ 0 & 0 & -3.3862e{-}01 & -1.6777e{+}00 & 2.8893e{+}00 \\ \hline 0 & 0 & 0 & 4.9304e-31 & 4.0000e{+}00 \end{array}\right].$$

Hoewel op dit moment geen 2*-bij-*2 *blok is afgescheiden, vinden we al wel* 1 *eigenwaarde terug, namelijk* 4*. We passen nu Algoritme 9 toe op de* 4*-bij-*4 *deelmatrix boven links, die we* $B^{(0)}$ *noemen. We vinden, opnieuw met* $\ell = 2$*, dat*

$$B^{(1)} = \begin{bmatrix} 6.8113e{+}00 & -2.3399e{+}01 & -7.0692e{+}00 & 3.8595e{+}01 \\ 1.3739e{+}00 & -2.8111e{+}00 & 9.4063e{-}01 & 3.3709e{+}00 \\ 0 & -1.7226e{-}04 & 1.5488e{-}01 & -4.0257e{+}00 \\ 0 & 0 & 5.7976e{-}01 & -2.1550e{+}00 \end{bmatrix},$$

$$B^{(3)} = \left[\begin{array}{rr|rr} 8.1385e{+}00 & -2.2717e{+}01 & 3.8899e{+}01 & -2.2871e{+}00 \\ 2.0548e{+}00 & -4.1385e{+}00 & 5.6530e{+}00 & 1.2135e{+}00 \\ \hline 0 & -2.8442e{-}20 & -1.6932e{+}00 & 3.4768e{-}01 \\ 0 & 0 & -4.2581e{+}00 & -3.0684e{-}01 \end{array}\right].$$

Er is nu wel onderaan een 2*-bij-*2 *blok afgesplitst, met eigenwaarden* $-1 \pm \imath$*. De resterende eigenwaarden zijn deze van het* 2*-bij-*2*-blok links bovenaan, namelijk* $2 \pm 3\imath$*.*

We besluiten dat de eigenwaarden van A gegeven worden door $\{4, -1 \pm \imath, 2 \pm 3\imath\}$*. Omdat we gerekend hebben met reële matrices en de twee verschuivingen eigenwaarden zijn van een deelmatrix, zijn deze ofwel beide reëel, ofwel vormen ze een complex toegevoegd paar. Matrix A heeft slechts* 1 *reële eigenwaarde, waardoor twee reële verschuivingen nooit twee eigenwaarden tegelijkertijd goed kunnen benaderen. Dit is de reden waarom in de eerste toepassing van Algoritme 9 slechts* 1 *eigenwaarde wordt afgesplitst, ondanks dat* $\ell = 2$*.*

Opmerking 9.4 *In Stelling 9.2.5, de theoretische basis, veronderstelden we dat alle elementen net onder de diagonaal van* $A^{(k)}$ *verschillend zijn van nul. Indien dit niet het geval is, is* $A^{(k)}$ *een blokdriehoeksmatrix en kan het eigenwaardenprobleem meteen opsplitst worden in dat voor de diagonale blokken.*

We eindigen het hoofdstuk met enkele beschouwingen.

- Het rekenwerk voor 1 iteratie van Algoritme 9, uiteengezet in §9.2.2.2, bedraagt $\sim 6(\ell+1)m^2$ elementaire bewerkingen. Dit in vergelijking met $\sim \frac{10}{3}m^3$ bewerkingen voor de initiële reductie van de matrix tot een Hessenberg-matrix (lijnen 1-13 van Algoritme 8). Bovendien is het aantal benodigde iteraties om eigenwaarden af te splitsen meestal zeer klein, en wordt na het afsplitsen van eigenwaarden Algoritme 9 toegepast of matrices van steeds kleinere dimensies. In de praktijk gedraagt het impliciet verschoven QR-algoritme (incl. het omvormen tot een Hessenberg-matrix) zich als een directe methode met een rekenkost van $\mathcal{O}(m^3)$ bewerkingen, hoewel het een iteratieve component heeft.

- We hebben het tot nog toe niet gehad over het berekenen van eigenvectoren. Eens de matrix tot een (blok)bovendriehoeksmatrix herleid is, waarvan de eigenwaarden op de diagonaal staan (of overeenkomen met deze van de diagonale blokken), kunnen de eigenvectoren efficiënt berekend worden doordat de matrix met de overeenkomstige eigenvectoren ook een (blok)bovendriehoeksmatrix is. Om de eigenvectoren in de oorspronkelijke basis te vinden, moeten al de uitgevoerde orthogonale transformaties op de matrix in omgekeerde volgorde op de eigenvectoren toegepast worden.

  Merk dat het in dat geval niet volstaat Algoritme 9 uit te voeren op een deelmatrix na het afsplitsen van een blok. De achterliggende transformatie van de deelmatrix moet dan gerealiseerd worden door een transformatie van de volledige matrix, die ook impact heeft op de niet-diagonale blokken.

Deel III

Niet-lineaire benaderingsproblemen

Hoofdstuk 10

Niet-lineaire benaderingsproblemen

In dit hoofdstuk bestuderen we de benadering van functies die een niet-lineaire parameterafhankelijkheid hebben van de te bepalen parameter. In tegenstelling tot lineaire benaderingsproblemen, die kunnen opgelost worden via een lineair stelsel (een overgedetermineerd stelsel of het normaalstelsel), vereisen niet-lineaire benaderingsproblemen (iteratiev) optimalisatietechnieken. Dit hoofdstuk maakt de overgang van lineaire naar niet-lineaire benaderingsproblemen. In Sectie 10.1 bekijken we benaderingsproblemen met niet-lineaire parameterafhankelijkheid. In Sectie 10.2 bekijken we de wiskundige definitie van diepe neurale netwerken.

10.1 Benaderingsproblemen met niet-lineaire parameterafhankelijkheid

In de vorige hoofdstukken gaf het benaderingsprobleem steeds aanleiding tot *lineaire stelsels*, omdat we de te benaderen functie steeds voorstelden als een lineaire combinatie van volledig gespecifieerde basisfuncties. Dat hoeft echter niet steeds het geval te zijn. We beschouwen twee voorbeelden van benaderingsproblemen waarbij de *basisfuncties zelf* ook afhangen van parameters die we wensen te bepalen.

Voorbeeld 10.1 (Benadering van een periodiek signaal) *Stel dat we signaal opmeten en voorstellen als $(t_i, y_i)_{i=1}^N$, met t_i het tijdstip en y_i de opgemeten waarde van het signaal. Als we veronderstellen dat het signaal periodiek is, kunnen we een benadering met n periodieke functies voorstellen van de vorm*

$$y_n(x) = \sum_{k=0}^{n} a_k \sin(\omega_k t + \phi_k) \tag{10.1}$$

waarbij ω_k de frequentie van de golf is en ϕ_k een faseverschuiving. Als we $(\omega_k)_{k=0}^n$ en $(\phi_k)_{k=0}^n$ vast kiezen, en enkel de coefficiënten $(a_k)_{k=0}^n$ wensen te bepalen, dan vallen we terug op het lineaire benaderingsprobleem van het eerste deel van de cursus. Als we ook de frequenties $(\omega_k)_{k=0}^n$ en/of fases $(\phi_k)_{k=0}^n$ optimaal wensen te kiezen (en dus als onbekend beschouwen),

dan hebben we een niet-lineair benaderingsprobleem. De niet-lineariteit ontstaat door de aanwezigheid van de sin*-functie, en doordat de parameters* $(a_k)_{k=0}^n$*,* $(\omega_k)_{k=0}^n$*, en* $(\phi_k)_{k=0}^n$ *met elkaar gekoppeld worden in elk van de termen.*
Een kleinste-kwadraten benadering bestaat erin de coëfficiënten als volgt te kiezen:

$$(a_k, \omega_k, \phi_k)_{k=0}^n = \arg \min_{a_k,\omega_k,\phi_k} \sum_{i=1}^N |y_i - y_n(t_i)|^2, \tag{10.2}$$

wat een niet-lineair optimalisatieprobleem is.

Voorbeeld 10.2 (Straling) *Stel dat je een vat aantreft met een aantal stoffen in die allemaal radioactief verval hebben. Het signaal* $(t_i, y_i)_{i=1}^N$ *dat in dit geval wordt opgemeten stelt dan de aangetroffen hoeveelheid straling voor. We kunnen dan die staling benaderen door een lineaire combinatie van dalende exponentiëlen als*

$$y_n(t) = \sum_{k=0}^n a_k e^{-\lambda_k t}, \tag{10.3}$$

waarbij de coëfficiënten $(a_k)_{k=0}^n$ *aangeven hoeveel er aanwezig is van stof* k*, en waarbij de parameters* $(\lambda_k)_{k=0}^n$ *verwant zijn aan de halfwaardetijd van stof* k*. Wanneer de aanwezige stoffen gekend zijn, kunnen de parameters* $(\lambda_k)_{k=0}^n$ *als gekend verondersteld worden. Het benaderingsprobleem wordt dan opnieuw een lineair benaderingsprobleem. Moeten we ook de parameters* $(\lambda_k)_{k=0}^n$ *bepalen, bekomen we opnieuw een niet-lineair benaderingsprobleem.*

Opmerking 10.1 (Niet-lineaire transformaties) *Hoewel het soms mogelijk kan zijn om via een gepaste transformatie een niet-lineair benaderingsprobleem alsnog lineair te maken, is dit vaak geen goed idee. Een eenvoudig voorbeeld. Stel dat we een functie wensen te benaderen door een enkele dalende exponentiële. Dan is is het voorschrift*

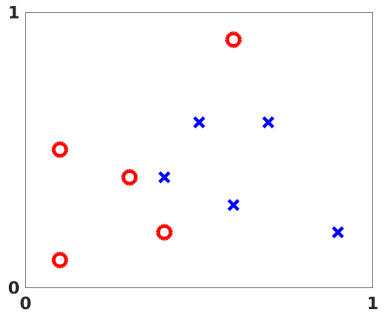
$$y(t) = ae^{-\lambda t} \tag{10.4}$$

equivalent met de getransformeerde voorstelling.

$$\log(y) = \log(a) - \lambda t \tag{10.5}$$

We zouden dan in de verleiding kunnen komen om de data te transformeren tot $(t_i, \log(y_i))_{i=1}^N$*, en vervolgens een lineair kleinste-kwadratenprobleem op te lossen. Helaas is dat geen goed idee. De kleinste-kwadratenbenadering veronderstelt namelijk iets over de meetfouten, en de structuur van de meetfouten verandert door de niet-lineaire transformatie.*

Opmerking 10.2 (Principal component analysis) *Een strategie die ergens tussen lineaire en niet-lineaire benaderingsmethodes in zit, is principal component analysis, zie ook hoofdstuk 12. Bij principal component analysis wordt lineaire beste benadering gezocht in een hoog-dimensionale deelruimte, waarna slechts enkele termen (die met de grootste coëfficiënt) geselecteerd worden. Strikt genomen is die selectiestap een niet-lineaire operatie.*



Figuur 10.1: Gelabelde datapunten in $\mathbb{R}^2$. Cirkels stellen punten uit categorie A voor. Kruisjes zijn punten uit categorie B.

Beide bovenstaande problemen geven aanleiding tot niet-lineaire optimalisatieproblemen van de vorm

$$c = \arg\min_c F(c), \tag{10.6}$$

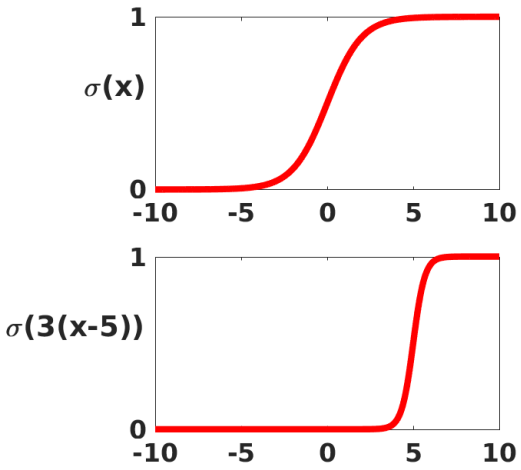
met c de te bepalen parameters. In Hoofdstuk 11 bespreken we numerieke methodes voor het oplossen van dat soort problemen.

10.2 Diepe neurale netwerken

In deze Sectie worden artificiële neurale netwerken benaderd als een probleem waarin we discrete data willen benaderen aan de hand van een specifieke (niet-lineaire) benaderingsfunctie. Als concreet voorbeeld beschouwen we de verzameling punten in Figuur 10.1. De figuur toont *gelabelde gegevens*: sommige punten, aangegeven door cirkels, vallen in categorie A, en de rest, aangegeven door kruisjes, valt in categorie B. De gegevens kunnen bijvoorbeeld twee biometrische patiëntgegevens zijn, en categorie A stelt dan de zieke patiënten voor. Kunnen we deze data gebruiken om voor een nieuw bekomen datapunt te bepalen of het in categorie A of B zal zitten? Onze taak is om een transformatie te maken die voor elk punt in RR^2 een een cirkel of een vierkant voorstelt. Dit benaderingsprobleem noemen we *gesuperviseerd leren*, omdat we zelf een boel data met labels ter beschikking hebben.

10.2.1 Activatiefuncties

Natuurlijk zijn er vele redelijke manieren om zo'n transformatie te construeren. Artificiële neurale netwerken gebruiken hiervoor herhaalde toepassing van een eenvoudige, mogelijks niet-lineaire, functie, die we *activatiefunctie* noemen. In de literatuur zijn veel dergelijke activatiefuncties gedefinieerd, en de keuze voor een activatiefunctie hangt sterk af van de toepassing en de gewoontes in een bepaald vakgebied. Enkele voorbeelden zijn:



Figuur 10.2: Bovenaan: sigmoide (10.8). Onderaan: sigmoide met verschoven en geschaalde invoer.

- lineair: $f(x) = x$
- binaire stap:

$$f(x) = \begin{cases} 0 & x \leq 0 \\ 1 & x \geq 0 \end{cases} \tag{10.7}$$

- sigmoide:

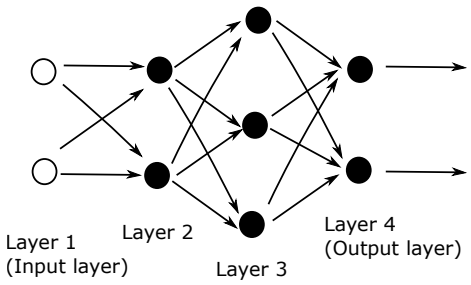
$$f(x) = \frac{1}{1 + e^{-x}} \tag{10.8}$$

- $f(x) = \tanh(x)$
- $f(x) = \arctan(x)$
- Rectified Linear Unit (RELU), stuksgewijs continue eerstegraadsveelterm

Merk op dat we voor de sigmoide functie $f(x) = \frac{1}{1+e^{-x}}$ een eenvoudig functievoorschrift hebben voor de afgeleide,

$$f'(x) = f(x)(1 - f(x)) \tag{10.9}$$

In dit geval is het evalueren van de afgeleide dus goedkoop. We kunnen de locatie en helling van de activatiefunctie aanpassen via een lineaire transformatie op de invoer van de functie. We berekenen dan $f(ax + b)$, waarbij a een schaalfactor is en b een verschuiving (*bias*). Figuur 10.2 toont de sigmoide functie met en zonder schaling en verschuiving.



Figuur 10.3: A network with four layers.

10.2.2 Lagen van neuronen

Nu we een activatiefunctie hebben, kunnen we lagen neuronen opzetten, zie Figuur 10.3. In deze figuur is elk bolletje een neuron, en elke kolom neuronen een laag. Het idee is dat, in elke laag, elk neuron een enkel reëel getal afgeeft, dat wordt doorgegeven aan elk neuron in de volgende laag. In de volgende laag vormt elk neuron zijn eigen gewogen combinatie van deze inkomende waarden, voegt vervolgens zijn eigen bias toe, en past tenslotte de sigmoïde functie toe op die gewogen en verschoven waarde.

We introduceren wat notatie. Als de reële getallen, geproduceerd door de neuronen inéén laag worden verzameld in een vector x, die als input dienen voor de huidige laag, dan is de vector van outputs van die huidige laag van de vorm

$$\sigma(Wx + b), \tag{10.10}$$

met W een matrix en b een vector. We zeggen dat W de *gewichten* bevat en b de *bias*. Het aantal kolommen in W komt overeen met het aantal neuronen die in de vorige laag de vector x produceerden. Het aantal rijen in W komt overeen met het aantal neuronen in de huidige laag. Ook het aantal componenten in b komt overeen met het aantal neuronen in de huidige laag. Om de rol van het $i$de neuron in (10.10) even te belichten, kunnen we de i-de component eruit pikken als

$$\sigma\left(\sum_j w_{ij}x_j + b_i\right),$$

waarbij de som loopt over alle entries in x. (Merk op dat we hier de notatie $\sigma(\cdot)$ zowel gebruiken voor een functie die een scalaire input omzet in een scalaire output, als voor de gevectoriseerde vorm van die functie. Dit is een licht misbruik van notatie dat het vervolg van dit hoofdstuk veel aangenamer om lezen zal maken.)

10.2.3 Een uitgewerkt voorbeeld

In de volgende Sectie zullen we een algemene notatie introduceren. Voor we daarmee beginnen, werken we eerst het voorbeeld van Figuur 10.3 in detail uit. Figuur 10.3 stelt een

artificieel neuraal netwerk voor met vier lagen. We zullen dit netwerk gebruiken voor het probleem gedefinieerd door Figuur 10.1. Voor het netwerk in Figuur 10.3 wordt de eerste (input) laag vertegenwoordigd door twee neuronen. Dit komt doordat onze invoerdata twee componenten heeft: de coördinaten van het datapunt in het vlak. De tweede laag bestaat uit twee neuronen. De pijlen van laag één naar laag twee geven aan dat beide componenten van de invoerdata beschikbaar worden gesteld aan de twee neuronen in laag twee. Aangezien de invoerdata de vorm $x \in \mathbb{R}^2$ heeft, kunnen de gewichten en biases voor laag twee worden weergegeven door een matrix $W^{[2]} \in \mathbb{R}^{2\times 2}$ en een vector $b^{[2]} \in \mathbb{R}^2$, respectievelijk. De output van laag twee heeft dan de vorm

$$\sigma(W^{[2]}x + b^{[2]}) \in \mathbb{R}^2.$$

Laag drie heeft drie neuronen, elk met invoer in $\mathbb{R}^2$. Daarom kunnen de gewichten en biases voor laag drie worden weergegeven door een matrix $W^{[3]} \in \mathbb{R}^{3\times 2}$ en een vector $b^{[3]} \in \mathbb{R}^3$, respectievelijk. De output van laag drie heeft dan de vorm

$$\sigma\left(W^{[3]}\sigma(W^{[2]}x + b^{[2]}) + b^{[3]}\right) \in \mathbb{R}^3.$$

De vierde (output) laag heeft twee neuronen, elk met invoer in $\mathbb{R}^3$. Daarom kunnen de gewichten en biases voor deze laag worden weergegeven door een matrix $W^{[4]} \in \mathbb{R}^{2\times 3}$ en een vector $b^{[4]} \in \mathbb{R}^{[2]}$, respectievelijk. De output van laag vier, en daarmee van het gehele netwerk, heeft de vorm

$$F(x) = \sigma\left(W^{[4]}\sigma\left(W^{[3]}\sigma(W^{[2]}x + b^{[2]}) + b^{[3]}\right) + b^{[4]}\right) \in \mathbb{R}^2. \tag{10.11}$$

10.2.4 Een niet-lineair optimalisatieprobleem

De uitdrukking (10.11) definieert een functie $F : \mathbb{R}^2 \to \mathbb{R}^2$ in termen van zijn 23 parameters, namelijk de elementen in de gewichtsmatrices en biasvectoren. Vanaf nu groeperen we de parameters in de vector $c \in \mathbb{R}^{23}$. We noteren dan de functie $F(x; c)$, waarbij $x \in \mathbb{R}^2$ een punt in het vlak is en de puntkomma het verschil aangeeft tussen de onafhankelijke variabele x en de parameters c.

Herinner je dat ons doel is om alle punten x in het vlak te classificeren als cirkel of kruisje, op basis van de gegevens in Figuur 10.1. We doen dit door de functie $F(x; c)$ te optimaliseren over de parameters c van het neurale netwerk. We eisen dat $F(x; c)$ dicht bij $[1, 0]^T$ ligt voor datapunten in categorie A en dichtbij $[0, 1]^T$ voor datapunten in categorie B. Dan, gegeven een nieuw punt $x \in \mathbb{R}^2$, zou het redelijk zijn om het te classificeren op basis van de grootste component van $F(x; c)$: in categorie A als $F_1(x; c) > F_2(x; c)$ en in categorie B als $F_1(x) < F_2(x)$, met een regel om te beslissen in geval $F_1(x) = F_2(x)$. Deze eis aan F kan worden gespecificeerd met behulp van een *kostfunctie*. We duiden de tien datapunten in Figuur 10.1 aan met $\{x^{\{i\}}\}_{i=1}^{10}$, en we gebruiken $y(x^{\{i\}})$ als de te benaderen functiewaarde,

die we definiëren als volgt:

$$y(x^{\{i\}}) = \begin{cases} \begin{bmatrix} 1 \\ 0 \end{bmatrix} & \text{als } x^{\{i\}} \text{ in categorie A,} \\ \\ \begin{bmatrix} 0 \\ 1 \end{bmatrix} & \text{als } x^{\{i\}} \text{ in categorie B.} \end{cases} \tag{10.12}$$

Wanneer we alle te bepalen parameters verzamelen in een vector

$$c = \left[W^{[2]}, W^{[3]}, W^{[4]}, b^{[2]}, b^{[3]}, b^{[4]}\right],$$

kunnen we onze kostfunctie definiëren als

$$\text{Kost}(c)\,(c) = \frac{1}{10}\sum_{i=1}^{10} \frac{1}{2}\left|y(x^i) - F(x^i; c)\right|_2^2. \tag{10.13}$$

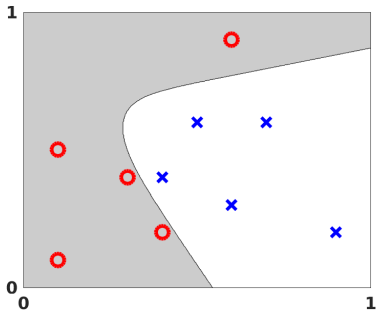
Hier is de factor 1/2 toegevoegd voor het gemak; die factor valt dan weg bij differentiatie. We benadrukken dat Kost enkel een functie is van de parameters c, i.e., gewichten en biases – de datapunten zijn vast. Je herkent hier opnieuw een niet-lineair kleinste-kwadratenprobleem. In de taal van optimalisatie is Kost onze *doelfunctie*. De factor 1/10 zorgt ervoor dat we de *gemiddelde* kwadratische afwijking minimaliseren. Deze normalisatiefactor zorgt ervoor dat de grootte-orde van de doelfunctie onafhankelijk wordt van het aantal datapunten. Stel dat we N datapunten hebben, ofwel *trainingspunten*, waarvoor doeluitvoer $y(x^i)i = 1^N$ wordt gegeven. Dan veralgemeent (10.13) zich als volgt:

$$\text{Kost} = \frac{1}{N}\sum_{i=1}^{N} \frac{1}{2}|y(x^i) - a^{[L]}(x^i)|_2^2, \tag{10.14}$$

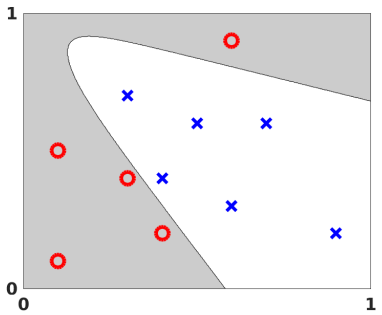
De normalisatie heeft enkel invloed op de waarde van de kostfunctie, niet op de waarde van c waarvoor de kost geminimaliseerd wordt. Het kiezen van de gewichten en biases op een manier die de kostfunctie minimaliseert, wordt *trainen* van het netwerk genoemd.

Voor de gegevens in Figuur 10.1 hebben we de MATLAB Optimization Toolbox gebruikt om de kostfunctie (10.13) te minimaliseren over de 23 parameters die $W^{[2]}$, $W^{[3]}$, $W^{[4]}$, $b^{[2]}$, $b^{[3]}$ en $b^{[4]}$ definiëren. Meer specifiek hebben we de niet-lineaire kleinste-kwadratenmethode `lsqnonlin` gebruikt. Voor het getrainde netwerk laat Figuur 10.4 de grens zien waar $F_1(x) > F_2(x)$. Met deze aanpak zou elk punt in het gearceerde gebied worden toegewezen aan categorie A en elk punt in het niet-gearceerde gebied aan categorie B. Figuur 10.5 laat zien hoe het netwerk reageert op extra trainingsgegevens. Hier hebben we nog één extra punt van categorie B toegevoegd, aangegeven door het extra kruisje op $(0.3, 0.7)$, en hebben we de optimalisatieroutine opnieuw uitgevoerd.

Het voorbeeld dat wordt geïllustreerd in Figuur 10.4 is qua schaal bijzonder klein in vergelijking met de huidige deep learning-tools die miljoenen tot miljarden parameters kunnen bevatten. Echter, het onderliggende optimalisatieprobleem, het minimaliseren van een



Figuur 10.4: Visualisatie van de uitvoer van een artificieel neuraal netwerk getrained op de data in Figuur 10.1.



Figuur 10.5: Herhaling van het experiment in Figuur 10.4 met een extra datapunt.

niet-convexe doelfunctie over 23 variabelen, is fundamenteel moeilijk. We kunnen niet exhaustief zoeken in een 23-dimensionale parameterruimte en we kunnen niet garanderen dat we het globale minimum van een niet-convexe functie vinden. Sterker nog, enig experimenteren met de locatie van de datapunten in Figuur 10.4 en met de keuze van de initiële schattingen voor de gewichten en biases zou snel tonen dat `lsqnonlin`, met zijn standaardinstellingen, niet altijd een acceptabele oplossing kan vinden. Dit motiveert het materiaal in Hoofdstuk 11, waar we ons specifike richten op numerieke methodes voor het optimalisatieprobleem.

10.3 Uitbreidingen bij neurale netwerken

10.3.1 Convolutionele neurale netwerken en beeldclassificatie

In beeldverwerking wordt standaard een speciale klasse van artificiële neurale netwerken gebruikt, de *convolutionele neurale netwerken*. Om die netwerken te motiveren, merken we op dat het algemene kader beschreven in de voorgaande sectie niet goed schaalt in het geval van digitale beelddata. Stel een kleurenafbeelding voor die bestaat uit 200 bij 200 pixels,

elk met een rode, groene en blauwe component. Dit komt overeen met een invoervector van dimensie $n_1 = 200 \times 200 \times 3 = 120000$, en dus een gewichtsmatrix $W^{[2]}$ op niveau 2 met 120000 kolommen. Als we algemene gewichten en biases toestaan, is deze benadering duidelijk onuitvoerbaar. Convolutionele neurale netwerken (CNNs) omzeilen dit probleem door het aantal niet-nul waarden in die matrices te beperken. In plaats van een enkele volledige lineaire transformatie passen CNNs herhaaldelijk een kleinschalige lineaire kernel of filter toe op delen van hun invoerdata: men kiest uiterst ijle en sterk gestructureerde gewichtsmatrices.

Om te begrijpen waarom deze aanpak nuttig kan zijn, kun je beschouwen dat we een invoervector in $\mathbb{R}^6$ vermenigvuldigen met de matrix

$$\begin{bmatrix} 1 & -1 & & & & \\ & 1 & -1 & & & \\ & & 1 & -1 & & \\ & & & 1 & -1 & \\ & & & & 1 & -1 \end{bmatrix} \in \mathbb{R}^{5\times 6}. \tag{10.15}$$

Dit levert een vector op in $\mathbb{R}^5$ die bestaat uit verschillen tussen aangrenzende waarden.

Geschikte veralgemeningen van deze matrix naar het geval van 2D-beelden kunnen worden gebruikt om *randen* in een afbeelding te detecteren. Dit wordt gedaan door een grote absolute waarde terug te geven wanneer er een abrupte verandering is in aangrenzende pixelwaarden. Op een gelijkaardige manier, kunnen ook andere kenmerken worden onthuld, zoals bepaalde soorten curven of vlekken van dezelfde kleur. Door een bepaald sparsiteitspatroon vooraf op te leggen, kunnen we het trainingsproces toestaan om de gewichten te leren als een middel om nuttige structuren uit beelden te extraheren. Het woord "convolutioneel" komt voort uit het feit dat de lineaire transformaties die hierbij betrokken zijn, kunnen worden geschreven in de vorm van een convolutie. In het geval van 1D wordt de k-de component van de convolutie van de vector $x \in \mathbb{R}^p$ met de filter $g_{1-p}, g_{2-p}, \ldots, g_{p-2}, g_{p-1}$ gegeven door

$$y_k = \sum_{n=1}^{p} x_n g_{k-n}.$$

Het voorbeeld hierboven wordt gegeven door de filter met componenten $g_0 = 1$, $g_{-1} = -1$ en alle andere $g_k = 0$.

In de praktijk wordt beeldgegevens doorgaans beschouwd als een driedimensionale tensor: elke pixel heeft twee ruimtelijke coördinaten en een rood/groen/blauw-waarde. Met deze benadering heeft de filter de vorm van een kleine tensor die achtereenvolgens wordt toegepast op patches van de invoertensor, en de bijbehorende convolutiebewerking is meerdimensionaal. Vanuit een berekeningsperspectief is een belangrijk voordeel van CNNs dat de matrix-vectorproducten die betrokken zijn bij de voorwaartse en achterwaartse doorvoer door het netwerk uiterst efficiënt kunnen worden berekend met behulp van snelle transformatietechnieken.

10.3.2 Overfitting vermijden en regularisatie

Zoals besproken in Hoofdstuk 7, treedt overfitting op wanneer een getraind netwerk nauwkeurig presteert op de gegeven invoerdata, maar niet goed kan veralgemenen naar nieuwe gegevens. De technieken uit Hoofdstuk 7 kunnen dus (uiteraard!) ook ingezet worden bij de training van neurale netwerken. De kostfunctie (10.14 kan bijvoorbeeld worden uitgebreid tot

$$\text{Kost} = \frac{1}{N}\sum_{i=1}^{N} \|y(x^{\{i\}}) - a^{[L]}(x^{\{i\}})\|_2^2 + \frac{\lambda}{N}\sum_{l=2}^{L} \| W^{[l]} \|_2^2. \tag{10.16}$$

Hier is $\lambda > 0$ de regularisatieparameter. Een motivatie voor (10.16) is dat grote gewichten kunnen leiden tot neuronen die gevoelig zijn voor hun invoer en daardoor minder betrouwbaar zijn wanneer nieuwe gegevens worden gepresenteerd. Dit argument lijkt minder belangrijk de biases, die meestal niet worden opgenomen in zo'n regularisatieterm. Hoewel we hier in deze cursus niet dieper op ingaan, is het eenvoudig te controleren dat het gebruik van (10.16) in plaats van (10.14) een zeer kleine en goedkope wijziging aanbrengt in de optimalisatieroutines.

Hoofdstuk 11

Optimalisatie-algoritmes

In dit Hoofdstuk behandelen we een aantal algoritmes voor het oplossen van optimalisatieproblemen zonder beperkingen, geschreven als

$$\min_{x\in\mathbb{R}^n} f(x),$$

waarbij verondersteld wordt dat alle afgeleiden van functie $f : \mathbb{R}^n \to \mathbb{R}$ tot orde twee bestaan en continue zijn. We beperken ons tot eerste-orde algoritmes, die enkel gebruik maken van evaluaties van de functie en haar gradiënt. In Sectie 11.1 herhalen we enkele basisbegrippen en resultaten van optimalisatie. In Sectie 11.2 laten we varianten van het algoritme van de steilste helling aan bod komen. In Sectie 11.3 stellen we het algoritme van de toegevoegde gradiënten voor. Tenslotte bespreken we in Sectie 11.4 kort de Gauss-Newton methode voor niet-lineaire kleinste-kwadratenproblemen.

11.1 Inleidende begrippen

Als f een scalaire functie is van n veranderlijken, $f : \mathbb{R}^n \to \mathbb{R}$, dan definiëren we de *gradiënt* van f als

$$\nabla f(x) := \left[\frac{\partial f(x)}{\partial x_1} \;\cdots\; \frac{\partial f(x)}{\partial x_n}\right]^T,$$

waarbij $x = [x_1 \;\cdots\; x_n]^T$. In het geval $n = 1$ is de gradiënt niets anders dan de eerste afgeleide. Voor een functie K van $\mathbb{R}^n \to \mathbb{R}^m$,

$$K(x) : \begin{cases} k_1(x_1,\ldots,x_n), \\ \vdots \\ k_m(x_1,\ldots,x_n), \end{cases}$$

definiëren we de *Jacobiaan* van K als

$$J_K(x) := \left[\frac{\partial k_i(x_1,\ldots,x_n)}{\partial x_j}\right]_{i,j=1}^{m,n} = \begin{bmatrix} \nabla k_1(x)^T \\ \vdots \\ \nabla k_m(x)^T \end{bmatrix} \in \mathbb{R}^{m\times n}.$$

We kunnen ook een afgeleide van f berekenen in een bepaalde richting. Een richting in $\mathbb{R}^n$ zullen we meestal aangeven door een eenheidsvector s. Als lengte zullen we in dit hoofdstuk steeds de Euclidische lengte gebruiken. Dus

$$\|s\|_2^2 = s^T s = \sum_{i=1}^{n} s_i^2 \quad \text{met} \quad s = [s_1 \ \cdots \ s_n]^T.$$

Als x zich vanuit het punt x^* in de richting s over een afstand α beweegt, dan beschrijft

$$x(\alpha) := x^* + \alpha s$$

een rechte lijn in $\mathbb{R}^n$. De functie $F(\alpha) := f(x(\alpha))$ wordt op die manier een functie van slechts één (scalaire) veranderlijke α. De afgeleide van f in de richting s in het punt $x(\alpha)$ is dan

$$\begin{array}{rcl} \frac{dF(\alpha)}{d\alpha} & = & \sum_{i=1}^{n} \frac{\partial f(x)}{\partial x_i} \frac{dx_i(\alpha)}{d\alpha} = \sum_{i=1}^{n} s_i \frac{\partial f(x)}{\partial x_i} \\ & = & s^T \nabla f(x) = (\nabla f(x))^T s. \end{array} \tag{11.1}$$

Dit noemt men de *richtingsafgeleide* of de *helling* (in de richting s). Uit uitdrukking (11.1) kan men volgende belangrijke besluiten trekken.

Eigenschap 11.1 *Van alle richtingen die men in een punt x kan beschouwen is de helling in de richting van de gradiënt het grootst (meest positief in de $+\nabla f(x)$ richting en meest negatief in de $-\nabla f(x)$ richting). Indien $\nabla f(x) \neq 0$ dan geldt bovendien:*

- *Indien de hoek tussen s en $\nabla f(x)$ kleiner (groter) is dan $\frac{\pi}{2}$, dan is de helling in de richting s positief (negatief).*
- *De gradiënt $\nabla f(x)$ is orthogonaal met betrekking tot de niveau-verzameling[1] (niveaulijn voor $n = 2$) door het punt x.*

De *Hessiaan* van functie $f : \ \mathbb{R}^n \to \mathbb{R}$ is een matrix die alle tweede-orde afgeleiden verzamelt,

$$\nabla^2 f(x) := \left[\frac{\partial^2 f(x)}{\partial x_i \partial x_j} \right]_{i,j=1}^{n} \in \mathbb{R}^{n \times n}.$$

Merk dat deze matrix symmetrisch is. De tweede afgeleide van f in de richting s, nl.

$$\frac{d^2 F(\alpha)}{d\alpha^2} \quad = \quad \frac{d}{d\alpha} \left[\sum_{i=1}^{n} s_i \frac{\partial f(x)}{\partial x_i} \right] = \sum_{i=1}^{n} \sum_{j=1}^{n} s_i \frac{\partial^2 f(x)}{\partial x_j \partial x_i} s_j = s^T \nabla^2 f(x) s,$$

duidt de *kromming* aan in het punt x in de richting s.

Het is welbekend dat een functie van slechts één veranderlijke die differentieerbaar is in een minimum een afgeleide heeft gelijk aan nul. Als de afgeleide nul is wil dit nog niet zeggen dat men altijd een minimum heeft. Men weet dan enkel dat de raaklijn horizontaal is en dan kan dit punt nog altijd een maximum zijn of een buigpunt. Maar wanneer de tweede afgeleide positief is hebben we steeds een minimum. De veralgemening naar functies van meerdere veranderlijken wordt beschreven door volgende stellingen.

[1] De niveauverzameling van f horende bij niveau c is de verzameling $\{x \in \mathbb{R}^n : \ f(x) = c\}$.

STELLING 11.1.1 *(nodige voorwaarde voor een minimum)*
Als f een lokaal minimum heeft in x^*, dan is $\nabla f(x^*) = 0$.

STELLING 11.1.2 *(voldoende voorwaarde voor een lokaal minimum)*
Als $\nabla f(x^*) = 0$ en de Hessiaan-matrix $\nabla^2 f(x^*)$ is positief definiet, dan bereikt f een geïsoleerd lokaal minimum in x^*.

Bewijs Uit $\nabla f(x^*) = 0$ volgt dat in het punt x^* in elke richting s de richtingsafgeleide nul is. Voorwaarde $\nabla^2 f(x^*) > 0$ impliceert dat $s^T \nabla^2 f(x^*) s > 0$ voor all $s \neq 0$. In elke richting is de kromming dus strikt positief. □

11.2 Methode van de steilste afdaling

In een iteratieve methode om een minimum van f te vinden zullen we doorgaans een iteratiepunt $x^{(k-1)}$ verbeteren door twee bewerkingen te doen:

1. We berekenen een zoekrichting $p^{(k)}$ waarvoor ${p^{(k)}}^T \nabla f(x^{(k-1)}) < 0$.

2. We zoeken een stapgrootte $\alpha^{(k)} > 0$ zodanig dat $x^{(k)} = x^{(k-1)} + \alpha^{(k)} p^{(k)}$ een (voldoende) lagere functiewaarde geeft dan $x^{(k-1)}$.

In elke stap kunnen we een afdaling verzekeren omdat we op deze manier een monotoon dalende rij van functiewaarden genereren. We verkrijgen aldus een algemene vorm van een iteratief algoritme voor een methode die men een *afdalingsmethode* zullen noemt. Het *algoritme van de steilste afdaling* bekomen we met de keuze $p^{(k)} = -\nabla f(x^{(k-1)})$, waarbij de richtingsafgeleide het kleinst is (zie Stelling 11.1).

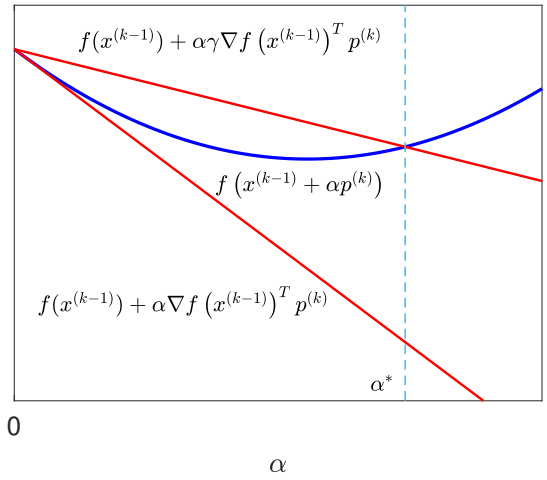
Als $p^{(k)}$ een daalrichting is, hebben we nog geen garantie dat $f\left(x^{(k-1)} + p^{(k)}\right) < f(x^{(k-1)})$ omdat het bepalen van $p^{(k)}$ gebaseerd is op lokale informatie in $x^{(k-1)}$ en een lokale lineaire benadering. We hebben echter wel de garantie dat voldaan is aan $f(x^{(k-1)} + \alpha p^{(k)}) < f(x^{(k-1)})$ voor voldoende kleine $\alpha > 0$. De methode voor het vinden van een geschikte waarde van α wordt een *lijnzoekmethode* genoemd. We vermelden twee klassen.

- Bij een *exacte lijnzoekmethode* gebruikt men de optimale α als stapgrootte, d.w.z.

  $$f\left(x^{(k-1)} + \alpha^{(k)} p^{(k)}\right) = \min_{\alpha > 0} f(x^{(k-1)} + \alpha p^{(k)}).$$

  Dit geeft echter niet altijd een vlugge convergentie. We moeten bovendien een scalair optimalisatieprobleem oplossen, wat bij de meeste doelfuncties iteratief moet gebeuren en een groot aantal functie-evaluaties vereist.

- In praktijk past men meestal *backtracking* toe. Dit betekent dat men vertrekt van de oorspronkelijke stapgrootte, bepaald door $\alpha = 1$, en men deze indien nodig met een constante factor laat afnemen, d.w.z. $\alpha \leftarrow q\alpha$ met $q < 1$, totdat $f\left(x^{(k-1)} + \alpha p^{(k)}\right)$



Figuur 11.1: Voor $\alpha \in (0, \alpha^*)$ is aan de Armijo-voorwaarde voldaan, waarbij $\gamma = \frac{1}{3}$. Richting $p^{(k)}$ is een daalrichting in $x^{(k-1)}$.

aan een bepaalde voorwaarde voor afname voldoet, zoals de zogenaamde *Armijo voorwaarde*

$$f\left(x^{(k-1)} + \alpha p^{(k)}\right) \leq f\left(x^{(k-1)}\right) + \nabla f(x^{(k-1)})^T \alpha\gamma p^{(k)}, \tag{11.2}$$

waarbij $\gamma \in (0, 1)$ een parameter is van het algoritme. Eens aan (11.2) voldaan stellen we $\alpha^{(k)} = \alpha$. Merk dat de raaklijn aan de functie $F(\alpha) = f\left(x^{(k-1)} + \alpha p^{(k)}\right)$ in $\alpha = 0$ gegeven wordt door $f\left(x^{(k-1)}\right) + \nabla f(x^{(k-1)})^T \alpha p^{(k)}$. Het rechterlid van (11.2) wordt bekomen door de helling met een factor γ aan te passen, zie Figuur 11.1.

De Armijo voorwaarde garandeert monotoniciteit, $f(x^{(k)}) < f(x^{(k-1)})$, en omdat de backtracking stopt zodra er aan voldaan is, worden veel te kleine stappen vermeden (die het risico inhouden om stil te vallen voor een minimum bereikt is) Voor meer detail verwijzen we naar [NW06].

11.3 Algoritme van de toegevoegde gradiënten

In §11.3.1 beschrijven we de basisiteratie van het algoritme van de toegevoegde gradiënten, dat als een verbetering van de methode van de steilste afdaling geïnterpreteerd kan worden. In §11.3.2 passen we het algoritme toe op het minimaliseren van een strikt convexe kwadratische functie of, equivalent hiermee, op het oplossen van een stelsel lineaire vergelijkingen met een symmetrische positief definiete matrix. We maken daarbij een gedetailleerde analyse van de convergentie, waarbij de Krylov-ruimte (9.7) een belangrijke rol speelt.

11.3.1 Principe en basisiteratie

Het algoritme, waarvan de basis in Algoritme 10 geschetst wordt, vertrekt van het algoritme van de steilste afdaling met een exact lijnzoekmethode. De enige aanpassing is dat als zoekrichting $p^{(k)}$ in stap 8 een bepaalde lineaire combinatie wordt genomen van de richting van de steilste afdaling (bepaald door $-\nabla f(x^{(k)})$ en de vorige zoekrichting, $p^{(k-1)}$ (die orthogonaal is met betrekking tot $\nabla f(x^k)$ door de exacte lijnzoekmethode). De factor $\beta^{(k)}$ is gekozen om tot een snellere convergentie te komen.
Merk dat het algoritme zéér eenvoudig te implementeren is. Tegelijk kan de winst in performantie door de schijnbaar kleine aanpassing spectaculair kan zijn, wat we voor een convexe kwadratische doelfunctie zullen aantonen in §11.3.2.

Algoritme 10 Algoritme van de toegevoegde gradiënten (basis)

Require: $x^{(0)}$
1: $p^{(0)} = -\nabla f(x^{(0)})$
2: **for** $k = 1, 2, \ldots$ **do**
3: $\quad \alpha^{(k)} = \arg\min_\alpha f\left(x^{(k-1)} + \alpha p^{(k-1)}\right)$
4: $\quad x^{(k)} = x^{(k-1)} + \alpha^{(k)} p^{(k-1)}$
5: $\quad \beta^{(k)} = \dfrac{\left\|\nabla f(x^{(k)})\right\|^2}{\left\|\nabla f(x^{(k-1)})\right\|^2}$
6: $\quad p^{(k)} = -\nabla f(x^{(k)}) + \beta^{(k)} p^{(k-1)}$
7: **end for**

11.3.2 Toepassing op stelsels vergelijkingen

De methode van de toegevoegde gradiënten wordt vaak gebruik voor het oplossen van stelsels vergelijkingen van de vorm

$$Ax = b, \tag{11.3}$$

met $A \in \mathbb{R}^{m\times m}$, $b \in \mathbb{R}^m$, waarbij m zeer groot is en matrix A symmetrisch positief definiet, genoteerd $A > 0$. Onder deze laatste voorwaarde maakt de oplossing, $x = x^*$, de kwadratische functie

$$f(x) := \frac{1}{2}x^T Ax - x^T b \tag{11.4}$$

minimaal. Dit volgt uit Stelling 11.1.2, rekening houdend met

$$\nabla f(x) = Ax - b, \quad \nabla^2 f(x) = A.$$

We specificeren nu stap voor stap de verschillende stappen uit Algoritme 10, toegepast op een kwadratische doelfunctie van de vorm (11.4).

In **stap 3** zoeken we het minimum op de lijn $x^{(k-1)} + \alpha p^{(k-1)}$. We kunnen dit vertalen naar een punt op deze lijn waar de richtingsafgeleide van f in richting $p^{(k-1)}$ nul is. In formules wordt dit

$$p^{(k-1)^T} \left(\nabla f \left(x^{(k-1)} + \alpha^{(k)} p^{(k-1)} \right) \right) = 0$$

of

$$p^{(k-1)^T} \left(A \left(x^{(k-1)} + \alpha^{(k)} p^{(k-1)} \right) - b \right) = 0,$$

waaruit volgt

$$\alpha^{(k)} = \frac{p^{(k-1)^T} r^{(k-1)}}{p^{(k-1)^T} A p^{(k-1)}}, \tag{11.5}$$

met

$$r^{(k)} = b - Ax^{(k)}$$

het *residu* in de k-de stap. In **stap 4** kunnen we naast x ook het residu updaten,

$$\begin{aligned} x^{(k)} &= x^{(k-1)} + \alpha^{(k)} p^{(k-1)}, \\ r^{(k)} &= r^{(k-1)} - \alpha^{(k)} A p^{(k-1)}, \end{aligned}$$

en in **stap 5** berekenen we parameter $\beta^{(k)}$ als

$$\beta^{(k)} = \frac{r^{(k)^T} r^{(k)}}{r^{(k-1)^T} r^{(k-1)}},$$

waarbij we gebruik maken van $\nabla f(x^{(k)}) = -r^{(k)}$. Kiezen we tenslotte $x^{(0)} = 0$ als startwaarde, komen we tot Algoritme 11

Algoritme 11 Algoritme van de toegevoegde gradiënten voor stelsels (basis)

1: $p^{(0)} = r^{(0)} = b,\ x^{(0)} = 0$
2: **for** $k = 1, 2, \ldots$ **do**
3: $\quad \alpha^{(k)} = \frac{p^{(k-1)^T} r^{(k-1)}}{p^{(k-1)^T} A p^{(k-1)}}$
4: $\quad x^{(k)} = x^{(k-1)} + \alpha^{(k)} p^{(k-1)}$
5: $\quad r^{(k)} = r^{(k-1)} - \alpha^{(k)} A p^{(k-1)}$
6: $\quad \beta^{(k)} = \frac{r^{(k)^T} r^{(k)}}{r^{(k-1)^T} r^{(k-1)}}$
7: $\quad p^{(k)} = r^{(k)} + \beta^{(k)} p^{(k-1)}$
8: **end for**

De variabelen die gegenereerd worden door iteraties van Algoritme 11 voldoen aan volgende eigenschappen, waarvan de eerste direct volgt uit de beschrijving van de stappen.

Eigenschap 11.2 *Als de iteratie van Algoritme 11 na $k-1$ stappen niet volledig geconvergeerd is, d.w.z. $r^{(k-1)} \neq 0$, dan is er voldaan aan*

$$\begin{aligned} \mathcal{K}_k(A,b) &= < x^{(1)}, x^{(2)}, \ldots, x^{(k)} > \\ &= < p^{(0)}, p^{(1)}, \ldots, p^{(k-1)} > \\ &= < r^{(0)}, r^{(1)}, \ldots, r^{(k-1)} > \end{aligned} . \tag{11.6}$$

Eigenschap 11.3 *Als $r^{(k-1)} \neq 0$, dan voldoen de gegenereerde residu's door Algoritme 11 en de zoekrichtingen aan volgende orthogonaliteitseigenschappen:*

$$r^{(k)^T} r^{(j)} = 0, \quad j = 1, \ldots, k-1, \tag{11.7}$$

$$p^{(k)^T} A p^{(j)} = 0, \quad j = 1, \ldots, k-1. \tag{11.8}$$

Bewijs Het bewijs is door inductie op k. De relaties (11.7)-(11.8) gelden voor $k = 1$, wat je zelf kan nagaan. Het volstaat dan (11.7)-(11.8) na te gaan voor een bepaalde waarde van k, gegeven dat aan deze betrekkingen voor lagere waarden van index k voldaan is. Voor (11.7) maken we het inwendig product van de uitdrukking op lijn 5 met $r^{(j)}$,

$$r^{(k)^T} r^{(j)} = r^{(k-1)^T} r^{(j)} - \alpha^{(k)} r^{(j)^T} A p^{(k-1)}. \tag{11.9}$$

We veronderstellen eerst $j < k-1$. Uit (11.6) volgt dat $r^{(j)} \in \mathcal{K}_{j+1}(A,b) =< p^{(0)}, \ldots, p^{(j)} >$. Het rechterlid van (11.9) is dan nul o.w.v. de inductiehypothese. Voor $j = k-1$ wordt het rechterlid van (11.9) nul op voorwaarde dat

$$\alpha^{(k)} = \frac{r^{(k-1)^T} r^{(k-1)}}{r^{(k-1)^T} A p^{(k-1)}}. \tag{11.10}$$

We tonen aan dat dit inderdaad het geval is. Vermits $r^{(k-1)} = p^{(k-1)} - \beta^{(k-1)} p^{(k-2)}$ en $p^{(k-2)} \in \mathcal{K}_{k-1}(A,b) =< r^{(0)}, \ldots, r^{(k-2)} >$ volgt uit de inductiehypothese dat

$$r^{(k-1)^T} r^{(k-1)} = r^{(k-1)^T} p^{(k-1)}. \tag{11.11}$$

Op een gelijkaardige manier kan aangetoond worden dat $r^{(k-1)^T} A p^{(k-1)} = p^{(k-1)^T} A p^{(k-1)}$, zodat (11.10) consistent is met de definitie van $\alpha^{(k)}$ op lijn 3.

Voor (11.8) maken we het inwendig product van de uitdrukking op lijn 7 met $Ap^{(j)}$,

$$p^{(k)^T} A p^{(j)} = r^{(k)^T} A p^{(j)} + \beta^{(k)} p^{(j)^T} A p^{(k-1)}. \tag{11.12}$$

We veronderstellen eerst $j < k-1$. Uit (11.6) volgt dat $Ap^{(j)} \in \mathcal{K}_{j+2}(A,b) =< r^{(0)}, \ldots, r^{(j+1)} >$. Het rechterlid van (11.12) is dan nul o.w.v. (11.7) en de inductiehypothese. Voor $j = k-1$ wordt het rechterlid van (11.12) nul op voorwaarde dat

$$\beta^{(k)} = -\frac{p^{(k-1)^T} A r^{(k)}}{p^{(k-1)^T} A p^{(k-1)}}. \tag{11.13}$$

We tonen aan dat dit inderdaad het geval is. Uit $r^{(k)} = r^{(k-1)} - \alpha^{(k)} A p^{(k-1)}$ en (11.7) volgt dat ${r^{(k)}}^T r^{(k)} = -\alpha^{(k)} {r^{(k)}}^T A p^{(k-1)}$. Bovendien geldt

$$ {r^{(k-1)}}^T r^{(k-1)} = {r^{(k-1)}}^T \left(r^{(k)} + \alpha^{(k)} A p^{(k-1)} \right) = \alpha^{(k)} {r^{(k-1)}}^T A p^{(k-1)} = \alpha^{(k)} {p^{(k-1)}}^T A p^{(k-1)}, $$

zodat de uitdrukking voor $\beta^{(k)}$ in lijn 6 consistent is met (11.13). □

Opmerking 11.1 *De naam* toegevoegde gradiënten *komt van eigenschap (11.8), men zegt dat de residu's (of gradiënten) A-toegevoegd zijn.*

Wanneer we in de uitdrukking voor $\alpha^{(k)}$ in lijn 3 van Algoritme 11 de substitutie (11.11) maken en een stopcriterium toevoegen, komen we tot de standaard vorm van het algoritme van de toegevoegde gradiënten, Algoritme 12.
Merk dat bij een implementatie enkel producten nodig zijn van A met vectoren, wat de methode zeer geschikt maakt voor problemen waarbij m groot is en matrix A veel nullen bevat.

Algoritme 12 Algoritme van de toegevoegde gradiënten voor stelsels (standaard vorm)

Require: $A > 0,\ b,\ k_{\max},\ \epsilon > 0$
1: $p^{(0)} = r^{(0)} = B,\ x^{(0)} = 0$
2: **for** $k = 1, 2, \ldots, k_{\max}$ **do**
3: $\quad \alpha^{(k)} = \frac{{r^{(k-1)}}^T r^{(k-1)}}{{p^{(k-1)}}^T A p^{(k-1)}}$
4: $\quad x^{(k)} = x^{(k-1)} + \alpha^{(k)} p^{(k-1)}$
5: $\quad r^{(k)} = r^{(k-1)} - \alpha^{(k)} A p^{(k-1)}$
6: $\quad$ **if** $\|r^{(k)}\|_2 < \epsilon$ **then**
7: $\quad\quad$ **return** $x^{(k)}$
8: $\quad$ **end if**
9: $\quad \beta^{(k)} = \frac{{r^{(k)}}^T r^{(k)}}{{r^{(k-1)}}^T r^{(k-1)}}$
10: $\quad p^{(k)} = r^{(k)} + \beta^{(k)} p^{(k-1)}$
11: **end for**

Stellen we $e^{(k)} = x^{(k)} - x^*$ en definiëren we de A-norm via $\|x\|_A = \sqrt{x^* A x}$, dan kunnen we de convergentie van Algoritme 11 (en Algoritme 12) als volgt karakteriseren.

Stelling 11.3.3 *(monotoniciteit van de fout)*
Als $r^{(k-1)} \neq 0$, dan voldoet de fout $e^{(k)}$ voor Algoritme 11 aan

$$ {e^{(k)}}^T A e^{(k)} = \min_{x \in \mathcal{K}_k(A,b)} (x - x^*)^T A (x - x^*), \tag{11.14} $$

dus $x^{(k)}$ is een element uit $\mathcal{K}_k(A, b)$ waarvoor de fout, gemeten met de A-norm, minimaal is. Bijgevolg is

$$ \|e^{(k)}\|_A \leq \|e^{(k-1)}\|_A \tag{11.15} $$

en is $e^{(n)} = 0$ voor een waarde van $n \geq m$.

Bewijs Uit (11.6) volgt dat $x^{(k)} \in \mathcal{K}_k(A,b)$. Elke $x \in \mathcal{K}_k(A,b)$ kan dan geschreven worden als $x = x^{(k)} + \Delta x$, met ook $\Delta x \in \mathcal{K}_k(A,b)$. Bovendien is dan $x - x^* = \Delta x + e^{(k)}$, zodat

$$\begin{aligned} &\min_{x\in\mathcal{K}_k(A,b)}(x-x^*)^T A(x-x^*) \\ &= \min_{\Delta x\in\mathcal{K}_k(A,b)}(\Delta x + e^{(k)})^T A(\Delta x + e^{(k)}) \\ &= \min_{\Delta x\in\mathcal{K}_k(A,b)} {e^{(k)}}^T A e^{(k)} + 2{e^{(k)}}^T A\Delta x + \Delta x^T A \Delta x. \end{aligned} \tag{11.16}$$

Nu is $Ae^{(k)} = -r^{(k)}$. Uit (11.7) volgt dan dat $Ae^{(k)} \perp\!\!< r^{(0)}, \ldots, r^{(k-1)} >\!\!= \mathcal{K}_k(A,b)$, overeenkomstig (11.6), wat leidt tot $Ae^{(k)} \perp \Delta$. De tweede term in de doelfunctie in (11.16) is dus altijd nul, terwijl de derde term voldoet aan $\Delta x^T A \Delta x \geq 0$. Het minimum wordt dus bereikt voor $\Delta x = 0$ of $x = x^{(k)}$.
De monotoniciteit van de fout (11.15) volgt uit $\mathcal{K}_{k-1}(A,b) \subseteq \mathcal{K}_k(A,b)$. Tenslotte, als na $m-1$ iteraties de fout nog steeds niet nul zou zijn, equivalent met $r^{(m-1)} \neq 0$, dan is dit het geval in de m-de iteratie omdat dan $\mathcal{K}_m(A,b) =< r^{(0)}, \ldots, r^{(m-1)} >= \mathbb{R}^m$ (de residu's zijn immers orthogonaal). □

Stelling 11.3.3 maakt duidelijk dat de methode van de toegevoegde gradiënten een Krylovmethode is, zoals het algoritme van Arnoldi voor het berekenen van eigenwaarden uit Sectie 9.1.3. Dit laatste algoritme zoekt benaderingen van eigenvectoren in een Krylovruimte, waarbij het residu orthogonaal is ten opzichte van de ruimte. Het algoritme van de toegevoegde gradiënten bepaalt een benaderende oplossing van het stelsel in de Krylovruimte, waarbij de fout in de A-norm minimaal is. Als tegenhanger van Stelling 9.1.3 is er ook nu een verband met veeltermbenadering.

STELLING 11.3.4 *(relatie met veeltermbenadering)*
Als $r^{(k-1)} \neq 0$, dan voldoet de fout $e^{(k)}$ voor Algoritme 11 aan

$$\|e^{(k)}\|_A^2 = \min_{p\in N_k} \left\|p(A)e^{(0)}\right\|_A^2, \tag{11.17}$$

met N_k de verzameling van alle veeltermen van graad kleiner of gelijk aan k die voldoen aan $p(0) = 1$.

Bewijs Voor $x \in \mathcal{K}_k(A,b)$ geldt, rekening houdend met $b = Ax^* = -A(x^{(0)} - x^*) = -Ae^{(0)}$, dat

$$x = c_1 b + c_2 Ab + \cdots + c_k A^{k-1} b = -c_1 Ae^{(0)} - c_2 A^2 e^{(0)} - \cdots - c_k A^k e^{(0)}$$

voor bepaalde $c_1, \ldots, c_k$. Vermits $e^{(0)} = x^{(0)} - x^* = -x^*$ volgt dat

$$\begin{aligned} x - x^* &= e^{(0)} - c_1 Ae^{(0)} - c_2 A^2 e^{(0)} - \cdots - c_k A^k e^{(0)} \\ &= \left(I - c_1 A - c_2 A^2 - \cdots - c_k A^k\right) e^{(0)} := \hat{p}(A) e^{(0)}, \end{aligned}$$

voor één of andere $\hat{p} \in N_k$. Het gestelde volgt dan uit (11.14). □

Gebruik makend van ontbinding (9.3) kunnen we (11.17) herschrijven als

$$\|e^{(k)}\|_A^2 = \min_{p\in N_k} \left\| X \begin{bmatrix} p(\lambda_1) & & \\ & \ddots & \\ & & p(\lambda_m) \end{bmatrix} X^{-1} e^{(0)} \right\|_A^2. \tag{11.18}$$

Hoe klein de fout na k iteraties is, wordt dus voor een groot deel bepaald door hoe goed het spectrum van matrix A "afgedekt" kan worden door een veelterm $p \in N_k$ van graad maximaal k. Als matrix A slechts k verschillende eigenwaarden heeft, met $k < m$, dan volgt uit (11.18) rechtstreeks dat volledige convergentie al optreedt in de k-iteratie - het minimum nul wordt dan bereikt door een veelterm met nulpunten in de eigenwaarden. Uit (11.18) volgt verder dat het voor snelle convergentie bevorderlijk is als de eigenwaarden van A gegroepeerd zijn in een klein aantal clusters.

Voorbeeld 11.1 *We beschouwen het stelsel (11.3) met $m = 1000$ en matrices gespecificeerd door*

$$b_i = 1,\ A_{ij} = \begin{cases} \frac{m-|i-j|}{4}, & i \neq j, \\ m + 2i, & i = j, \end{cases} \tag{11.19}$$

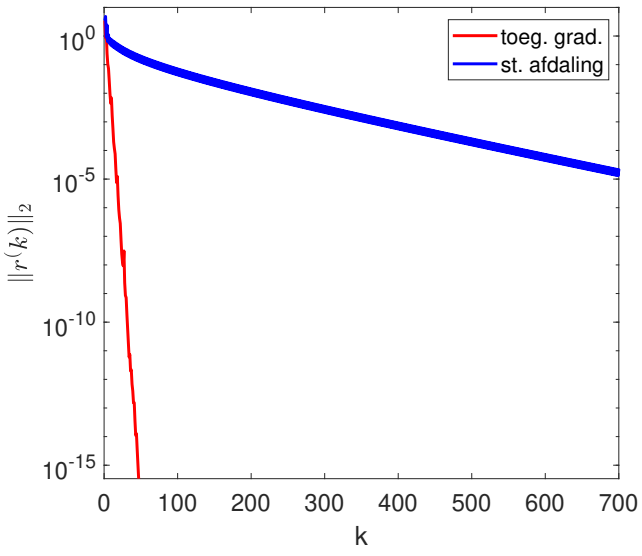
voor $i,\ j \in \{1, \ldots, m\}$. We vergelijken het algoritme van de steilste afdaling, met startwaarde $x^{(0)} = 0$ en een exacte lijnzoekmethode, met de methode van de toegevoegde gradiënten. Voor doelfunctie (11.4) herleidt Algoritme 12 zich tot het algoritme van de steilste afdaling door lijn 10 te vervangen door $p^{(k)} = r^{(k)}$.
In Figuur 11.2 en het linker luik van Figuur 11.3 wordt de norm van het residu $r^{(k)}$ getoond in functie van iteratienummer k - het enige verschil tussen beide plots is het bereik van k. Merk dat de methode van de steilste afdaling zeer traag convergeert, de combinatie van de (lokaal) steilste afdalingsrichting en een exacte lijnzoekmethode is dus verre van ideaal. Het algoritme van de toegevoegde gradiënten convergeert zeer snel, en dit enkel door als zoekrichting een goede lineaire combinatie te maken van de richting van steilste afdaling en de vorige zoekrichting
In het rechter luik van Figuur 11.3 wordt de A-norm van de fout weergegeven in functie van het iteratienummer. Deze functie daalt monotoon, overeenkomstig (11.15). Merk dat de waarde van deze norm niet als stopcriterium gebruikt kan worden, omdat de fout $e^{(k)}$ op x afhangt van de gezochte oplossing x^*.

Merk tenslotte dat voor elke inverteerbare matrix $D \in \mathbb{R}^{m \times m}$ het stelsel

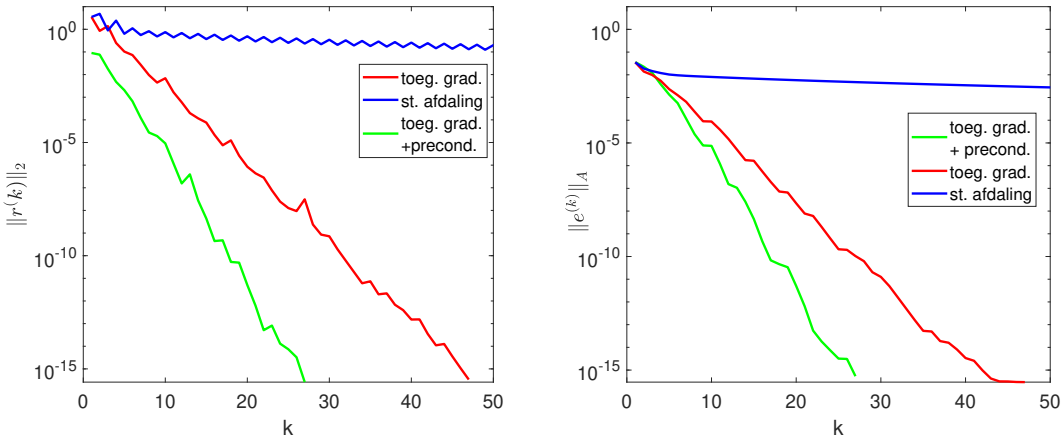
$$(D^{-1}AD^{-T})y = D^{-1}b,$$

waarbij $D^{-T}y = x$, equivalent is met (11.3) en $(D^{-1}AD^{-T}) > 0$ als $A > 0$. Als het matrix vector-product met D^{-1} en D^{-T} gemakkelijk kan worden berekend, kan men ook Algoritme 12 toepassen met als input matrices $(D^{-1}AD^{-T})$ en $D^{-1}b$. De eigenwaarden van $D^{-1}AD^{-T}$ zijn in het algemeen verschillend van deze van A en als deze laatste beter geclusterd zijn, kan dit de convergentie versnellen, volgend uit (11.18). Matrix D wordt een *preconditioner* genoemd. We gaan hier niet verder op in en geven enkel een illustratie.

Voorbeeld 11.2 *In matrix A van het voorbeeld (11.19) zijn de diagonaalelementen relatief groot in vergelijking met niet-diagonaalelementen, wat een indicatie is dat de meeste eigenwaarden in de buurt liggen van de diagonaalelementen. Door D te kiezen als een diagonaalmatrix met $D_{ii} = \sqrt{A_{ii}},\ i = 1, \ldots, m$, worden alle diagonaalelementen getransformeerd naar 1. Intuïtief wordt daar een grote cluster van eigenwaarden verwacht. Het effect op de convergentiesnelheid wordt getoond in Figuur 11.3.*



Figuur 11.2: Norm van de residu's bij het toepassen van het algoritme van de steilste afdaling en dat van de toegevoegde gradiënten, voor stelsel (11.3) met matrices (11.19).



Figuur 11.3: Norm van de residu's bij het toepassen van het algoritme van de steilste afdaling en het (voorgeconditioneerde) algoritme van de toegevoegde gradiënten, voor stelsel (11.3) met matrices (11.19).

11.4 De methode van Gauss-Newton

In Voorbeeld 2.10 beschouwden we het vereffenen van meetgegevens $\{(t_i, y_i)\}_{i=1}^m$, waarbij het vooropgestelde model een lineaire combinatie was van n basisfuncties, met $n < m$. Dit gaf aanleiding tot een overgedetermineerd stelsel van de vorm $Ax = b$, met $A \in \mathbb{R}^{m\times n}$.
Stel nu dat het model op een niet-lineaire manier afhangt van de parameters $c_1, \ldots, c_n$, waarbij $g(t_i, c_1, \ldots, c_n)$ de voorspelling uitdrukt van y_i op basis van een model met parameters $c_1, \ldots, c_n$. Om deze laatste te bepalen kunnen we opnieuw de som van de kwadraten van de afwijkingen minimaliseren:

$$\min_{c_1,\ldots,c_n} \sum_{i=1}^{m} \left(y_i - g(t_i, c_1, \ldots, c_n)\right)^2,$$

wat een speciaal geval is van het optimalisatieprobleem

$$\min_c \|F(c)\|_2^2, \tag{11.20}$$

met $F: \mathbb{R}^n \to \mathbb{R}^m$ en $m > n$. Specifiek is voor bovenvermelde toepassing $c = [c_1 \ \cdots \ c_n]^T$ en

$$F(c) = \begin{bmatrix} y_1 - g(x_1, c_1, \ldots, c_n) \\ \vdots \\ y_m - g(x_m, c_1, \ldots, c_n) \end{bmatrix}.$$

Het idee achter de Gauss-Newton methode is om een benadering $c^{(k-1)}$ van een oplossing van (11.20) te verbeteren door de functie F, binnen de norm, te lineariseren en het lineaire kleinste-kwadratenprobleem op te lossen. De oplossing ervan definieert dan $c^{(k)}$, en als de iteratie convergeert, kan men aantonen dat de oplossing overeenkomt met een (lokaal) minimum van (11.20).
Concreet, vertrekkende vanuit $c^{(k-1)}$ benaderen we F door $F\left(c^{(k-1)}\right) + J_F(c^{(k-1)})(c - c^{(k-1)})$, met J_F de Jacobiaan van F, en lossen we

$$\min_p \left\| F\left(c^{(k-1)}\right) + J_F\left(c^{(k-1)}\right) p \right\|_2^2 \tag{11.21}$$

op, met $p = c - c^{(k-1)}$. Noemen we de oplossing $p^{(k)}$, dan stellen we $c^{(k)} = c^{(k-1)} + p^{(k)}$.
Het deelprobleem (11.21) is een lineair kleinste-kwadratenprobleem met $A = J_F\left(c^{(k-1)}\right)$ en $b = -F(c^{(k-1)})$, dat we met de methodologie van Sectie 2.4.2 kunnen oplossen.

Een convergentie-analyse van de Gauss-Newton methode is buiten de scope van de cursus. Hiervoor verwijzen we de geïnteresseerde lezer naar [NW06].

11.5 Optimalisatie bij neurale netwerktraining

We hebben in Hoofdstuk 10 gezien dat het trainen van een netwerk overeenkomt met het kiezen van de parameters, dat wil zeggen de gewichten en biases, die de kostfunctie

minimaliseren. De gewichten en biases hebben de vorm van matrices en vectoren, maar voor de optimalisatie is het handig om ze voor te stellen als een enkele vector die we c noemen. Het voorbeeld in Figuur 10.3 heeft in totaal 23 gewichten en biases. Dus in dat geval geldt $c \in \mathbb{R}^{23}$. Over het algemeen veronderstellen we dat $c \in \mathbb{R}^n$, en schrijven we de kostfunctie in (10.14) als Kost(c) om de afhankelijkheid van de parameters te benadrukken. Dus Kost : $\mathbb{R}^n \to \mathbb{R}$. In deze Sectie neemt die kostfunctie de rol van $f(x)$ in vorige Secties.

In dit hoofdstuk hebben we een aantal methodes gezien om dergelijke niet-lineaire optimalisatieproblemen op te lossen. In de context van diepe neurale netwerken wordt vooral de methode van de steilste helling (en varianten daarop) gebruikt. We noteren die methode als

$$c^{(k+1)} = c^{(k)} - \eta \nabla \text{Kost}(c^{(k)}). \tag{11.22}$$

Hier is η een kleine stapgrootte die in deze context bekend staat als de *learning rate* (leersnelheid). We kiezen een initiële vector en itereren met (11.22) totdat een stopcriterium is bereikt of totdat een maximaal aantal iteraties bereikt is.

Een belangrijk aspect bij het trainen van neurale netwerken is het berekenen van de gradiënt. Dit kan efficiënt met een methode die *backpropagation* heet. Bij *backpropagation* wordt een gelineariseerde versie van het neurale netwerk achterwaarts doorlopen. Dit blijkt neer te komen op een systematisch gebruik van de kettingregel voor afgeleiden. Dit aspect leggen we hier niet verder uit.

Onze kostfunctie (10.14) omvat een som van individuele termen die over de trainingsgegevens loopt. Hieruit volgt dat de partiële afgeleide $\nabla \text{Cost}(p)$ een som is over de trainingsgegevens van individuele partiële afgeleiden. Meer precies, laat ons noteren

$$C_{x^{\{i\}}} = \|y(x^{\{i\}}) - a^{[L]}(x^{\{i\}})\|_2^2. \tag{11.23}$$

Dan, vertrekkend van (10.14), bekomen we

$$\nabla \text{Cost}(p) = \frac{1}{N} \sum_{i=1}^{N} \nabla C_{x^{\{i\}}}(p). \tag{11.24}$$

Wanneer we een groot aantal parameters en trainingspunten hebben, kan het berekenen van de gradiëntvector (11.24) bij elke iteratie van de steilste afdalingsmethode (11.22) zeer veel rekenkracht vereisen. Een veel goedkoper alternatief is om het gemiddelde van individuele gradiënten over alle trainingspunten te vervangen door de gradiënt bij een willekeurig gekozen trainingspunt. Dit leidt tot de eenvoudigste vorm van de zogenaamde *stochastische gradiëntmethode*. Een enkele stap kan als volgt worden samengevat:

1. Kies een willekeurig geheel getal i uit de verzameling $1, 2, 3, \ldots, N$.

2. Werk c bij met de volgende formule:

$$c^{(k+1)} \to c^{(k)} - \eta \nabla C_{x^{\{i\}}}(c^{(k)}). \tag{11.25}$$

Bij elke stap gebruikt de stochastische gradiëntmethode één willekeurig gekozen trainingspunt om de volledige trainingsset te vertegenwoordigen. Naarmate de iteratie vordert, komt de methode meer trainingspunten tegen, wat hoop biedt dat de vermindering in kosten per iteratie uiteindelijk de moeite waard zal zijn.

Het is belangrijk op te merken dat zelfs met een kleine leersnelheid η de update (11.25) niet gegarandeerd de algehele kostfunctie verlaagt. Door het gebruik van één enkel monster in plaats van het gemiddelde, introduceren we willekeur in het optimalisatieproces. Daarom geven we er de voorkeur aan om deze methode simpelweg *stochastische gradiënt* te noemen, ook al wordt vaak de methode de stochastische methode van de steilste afdaling genoemd.

De versie van de stochastische gradiëntmethode die we introduceerden in (11.25) is de eenvoudigste variant van een breed scala aan mogelijkheden. In het bijzonder werd de index i in (11.25) gekozen door middel van een *steekproef met teruglegging*: nadat een trainingspunt is gebruikt, wordt het teruggeplaatst in de trainingsset en is het even waarschijnlijk als elk ander punt om bij de volgende stap te worden gekozen. Een alternatief is om zonder teruglegging te werken, wat betekent dat je elk van de N trainingspunten beurtelings selecteert in een willekeurige volgorde. Het uitvoeren van N stappen op deze manier, wat een *epoch* wordt genoemd, kan als volgt worden samengevat:

1. Permuteer de indices $\{1, 2, 3, \ldots, N\}$ naar een nieuwe volgorde, $\{k_1, k_2, k_3, \ldots, k_N\}$.
2. for $i = 1$ upto N, update

$$c^{(k+1)} = c^{(k)} - \eta \nabla C_{x^{\{k_i\}}}(c^{(k)}). \tag{11.26}$$

Als we de stochastische gradiëntmethode beschouwen als een benadering van het gemiddelde over alle trainingspunten in (11.24) door een enkele steekproef, dan is het natuurlijk om een compromis te overwegen waarbij we een klein steekproefgemiddelde gebruiken. Voor een bepaald $m \ll N$ kunnen we stappen nemen van de volgende vorm:

1. Kies m indices, $k_1, k_2, \ldots, k_m$, uniform en willekeurig vanuit $\{1, 2, 3, \ldots, N\}$.
2. Update

$$c^{(k+1)} = c^{(k)} - \eta \frac{1}{m} \sum_{i=1}^{m} \nabla C_{x^{\{k_i\}}}(c^{(k)}). \tag{11.27}$$

In deze procedure wordt de verzameling $\{x^{\{k_i\}}\}_{i=1}^{n}$ de *mini-batch* genoemd. Er is een alternatief *zonder teruglegging* waarbij we, ervan uitgaande dat $N = Km$ voor een bepaald K, de trainingsset willekeurig opsplitsen in K verschillende mini-batches en die sequentieel doorlopen.

De keuzes binnen de stochastische gradiëntmethode, zoals de grootte van de mini-batches en de vorm van randomisatie, worden vaak bepaald door de eisen van high-performance computing-architecturen, omdat deze methode meestal wordt toegepast binnen de context van grootschalige berekeningen. Daarnaast is het uiteraard mogelijk om deze keuzes,

samen met andere factoren zoals de leersnelheid, dynamisch te variëren naarmate het trainingproces vordert, in een poging om de convergentie te versnellen. Dit stelt het algoritme in staat om zich aan te passen aan de eigenschappen van de specifieke dataset en mogelijke uitdagingen tijdens het trainingsproces. Het vinden van de optimale set van parameters en hyperparameters is vaak een iteratief proces waarbij verschillende combinaties worden uitgeprobeerd om de beste prestaties te bereiken.

Hoofdstuk 12

Ijle representaties en benaderingen

In dit hoofdstuk behandelen we een klasse van technieken die toelaten om goede benaderingen op te stellen van functies of om het aantal variabelen te reduceren. Deze zijn gebaseerd op transformaties of factorisaties die een ijle structuur aan het licht brengen, in de zin dat belangrijke van minder belangrijke informatie gescheiden kan worden. Vooreerst behandelen we in Sectie 12.1 de singuliere-waardenontbinding, die een belangrijke rol speelt bij een aantal van deze technieken. Vervolgens bespreken we in Sectie 12.2 methodes om goede lage-rangbenaderingen te bekomen van data of functies voorgesteld door matrices. In Sectie 12.3 komt principal component analysis aan bod, een veelgebruikte methode om de dimensie van een dataset te reduceren. In Section 12.4 tenslotte behandelen we bondig andere benaderingstechnieken, gebaseerd op een transformatie naar het frequentiedomein of op basis van resolutie.

12.1 De singuliere-waardenontbinding

We herhalen de definitie en enkele belangrijke eigenschappen, waarna computationele aspecten aan bod komen.

12.1.1 Definitie en eigenschappen

De singuliere-waardenontbinding van matrix $A \in \mathbb{C}^{m\times n}$ is een ontbinding van de vorm

$$A = U\Sigma V^*,$$

met $U \in \mathbb{C}^{m\times m}$ en $V \in \mathbb{C}^{n\times n}$ unitaire matrices en met $\Sigma = \text{diag}(\sigma_1, \ldots, \sigma_m)$ voor $m = n$,

$$\Sigma = \begin{bmatrix} \sigma_1 & 0 & \cdots & 0 \\ 0 & \sigma_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \sigma_n \\ 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 \end{bmatrix} \in \mathbb{R}^{m\times n}$$

voor het geval $m > n$ en

$$\Sigma = \begin{bmatrix} \sigma_1 & 0 & \cdots & 0 & 0 & \cdots & 0 \\ 0 & \sigma_2 & \cdots & 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \sigma_m & 0 & \cdots & 0 \end{bmatrix} \in \mathbb{R}^{m \times n}$$

voor het geval $m < n$. De kolommen van U worden linker singuliere vectoren genoemd, de kolommen van V rechter singuliere vectoren. De singuliere waarden $\sigma_i \geq 0$ zijn geordend zodat

$$\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_p \geq 0,$$

met $p = \min(m, n)$. De rang r van A wordt bepaald door het aantal singuliere waarden verschillend van nul. Voor $r < p$ geldt dat $\sigma_1 \geq \cdots \geq \sigma_r > 0$ en $\sigma_{r+1} = \cdots = \sigma_p = 0$. We formuleren enkele eigenschappen.

- De eerste r kolommen van U vormen een orthonormale basis van de kolomruimte $\mathcal{R}(A)$, de eerste r rijen van V^* een orthonormale basis van de rijruimte van A.
- De laatste $n - r$ kolommen van V vormen een orthonormale basis van de nulruimte $\mathcal{N}(A)$.
- We kunnen A ontbinden in r matrices van rang 1, namelijk

  $$A = \sum_{i=1}^{r} \sigma_i u_i v_i^*,$$

  waarbij $U = [u_1 \ \cdots \ u_m]$ en $V = [v_1 \ \cdots \ v_n]$. We kunnen A ook schrijven als

  $$A = YZ^*, \text{ met } Y \in \mathbb{C}^{m \times r}, \ Z \in \mathbb{C}^{n \times r}.$$

  Dat kan bijvoorbeeld met de keuze $Y = [\sigma_1 u_1 \ \cdots \ \sigma_r u_r]$ en $Z = [v_1 \ \cdots \ v_r]$.
- Uit de relatie

  $$A^*AV = V\Sigma^*\Sigma \tag{12.1}$$

  volgt dat vector $v_i, \ i \in \{1, \ldots, n\}$, een eigenvector is van A^*A. Voor $i \leq r$ is de bijhorende eigenwaarde σ_i^2, voor $i > r$ nul.
- Matrix Σ is steeds uniek bepaald. In het geval waarbij $\sigma_1 > \sigma_2 > \cdots > \sigma_r$ is vector $v_i, \ i \in \{1, \ldots, r\}$, uniek, op een complexe constante met modulus 1 na. Eens deze constante vastligt is ook u_i uniek bepaald omdat uit de ontbinding $AV = U\Sigma$, oftewel

  $$Av_i = \sigma_i u_i \tag{12.2}$$

  volgt.

- Er geldt

$$\|A\|_2 = \sigma_1, \quad \|A\|_F = \sqrt{\sum_{i=1}^{r} \sigma_i^2}.$$

Als $A \in \mathbb{R}^{m\times n}$, dan kunnen matrices U en V in de ontbinding als reële matrices gekozen worden. Het zijn dan *orthogonale* matrices, zodat $U^TU = I_m,\ V^TV = I_n$.

12.1.2 Berekening van de ontbinding

Omdat de singuliere-waardenontbinding van A volgt uit deze van A^* en vice versa, kunnen we, zonder algemeenheid in te boeten, veronderstellen dat $m \geq n$ is. In een eerste stap kan matrix A, door vermenigvuldiging met orthogonale matrices links en rechts, omgevormd worden tot een bi-diagonale matrix, in formule

$$Q_l^* A Q_r = \left[\begin{array}{ccccc} b_{11} & b_{12} & & \cdots & 0 \\ 0 & b_{22} & b_{23} & & \vdots \\ \vdots & & \ddots & \ddots & 0 \\ & & & b_{n-1\ n-1} & b_{n-1\ n} \\ 0 & \cdots & & 0 & b_{nn} \\ \hline 0 & & \cdots & & 0 \\ \vdots & & & & \vdots \\ 0 & & \cdots & & 0 \end{array}\right] := \left[\begin{array}{c} A_0 \\ \hline 0 \end{array}\right].$$

In tegenstelling tot de aanpak in §9.2.1 (eerste stap bij het berekenen van alle eigenwaarden), kunnen matrices Q_l en Q_r onafhankelijk van elkaar gekozen worden. Dit verklaart waarom een reductie tot een bi-diagonale matrix altijd mogelijk is (in plaats van slechts een Hessenberg-matrix, waarbij de elementen rechts bovenaan niet nul hoeven te zijn). Ga dit na! In de tweede stap wordt de singuliere-waardenontbinding van de *vierkante* bi-diagonale matrix A_0 met een iteratief algoritme bepaald (wat neerkomt op het verder herleiden van de matrix tot een diagonale matrix). Dit resulteert in

$$A_0 = U_0 \Sigma_0 V_0^*. \tag{12.3}$$

De singuliere-waardenontbinding van A wordt dan gegeven door

$$A = \underbrace{Q_l \left[\begin{array}{c} U_0 \\ \hline 0 \end{array}\right]}_{U} \underbrace{\Sigma_0}_{\Sigma} \underbrace{V_0^* Q_r^*}_{V^*}.$$

We gaan nu wat dieper in op de berekening van (12.3), waarvoor eigenwaardenalgoritmen gebruikt kunnen worden. Relatie (12.1) suggereert een aanpak gebaseerd op het berekenen van alle eigenwaarden en eigenvectoren van $A_0^*A_0$. Deze aanpak is echter niet numeriek

stabiel[1]. Beter is om eigenwaarden en eigenvectoren te berekenen van de Hermitiaanse matrix

$$H = \begin{bmatrix} 0 & A_0^* \\ A_0 & 0 \end{bmatrix},$$

waarbij het product van matrices vermeden wordt. Je kan gemakkelijk nagaan dat

$$H \begin{bmatrix} V_0 & V_0 \\ U_0 & -U_0 \end{bmatrix} = \begin{bmatrix} V_0 & V_0 \\ U_0 & -U_0 \end{bmatrix} \begin{bmatrix} \Sigma_0 & 0 \\ 0 & -\Sigma_0 \end{bmatrix},$$

wat je kan interpreteren als een eigenwaardenontbinding (zie Appendix A.4). De eigenwaarden van H zijn dus $\pm\sigma_1, \ldots, \pm\sigma_n$, en uit de overeenkomstige eigenvectoren kunnen zowel linker als rechter singuliere vectoren gehaald worden.

12.2 Lage-rangbenaderingen

We beschouwen het benaderingsprobleem van een gegeven matrix $A \in \mathbb{C}^{m\times n}$ van rang r door een matrix van rang kleiner of gelijk aan k, met $k < r$. We zoeken met andere woorden een benadering van A door een element van de verzameling

$$\mathcal{M}_k(m,n) := \left\{ B \in \mathbb{C}^{m\times n} : \ \mathrm{rang}(B) \leq k \right\}.$$

De kwaliteit van een benadering $B \in \mathcal{M}_k(m,n)$ van matrix A kunnen we kwantificeren door middel van $\|B - A\|$, met $\|\cdot\|$ een matrix norm.
Lage-rangbenaderingen hebben heel wat toepassingen in de context van signaal- en beeldverwerking, datacompressie, het voorspellen van ontbrekende multi-variate data (bijvoorbeeld bij aanbevelingssystemen). Ook vormen ze vaak een essentieel onderdeel van algoritmen voor het oplossen van partiële differentiaalvergelijkingen omwille van de compacte representatie van (benaderende) oplossingen. De achterliggende reden is tweevoudig. Ten eerste kan een m-bij-n matrix B van rang k voorgesteld worden als $M = XY^*$, met $X \in \mathbb{C}^{m\times k}$ en $Y \in \mathbb{C}^{n\times k}$. Het bewaren en manipuleren van de factoren X en Y (in plaats van matrix B zelf) kan een aanzienlijke winst in geheugen- en rekenkost opleveren als $k \ll \min(m,n)$. Ten tweede zijn heel wat fysische systemen en processen achter data vaak inherent laag-dimensionaal, in de zin dat men met relatief lage k al een goede benadering kan bekomen.
Voor het bepalen van beste lage-rangbenaderingen is de algemene theorie uit Hoofdstukken 2-3 niet bruikbaar, hoewel de verzameling van matrices van gegeven dimensies een vectorruimte vormt en de Frobenius-norm matrices behandelt als ware het vectoren. De reden is dat verzameling $\mathcal{M}_k(m,n)$ geen deel(vector)ruimte is van $\mathbb{C}^{m\times n}$. Ze is bijvoorbeeld niet gesloten onder de optelling, wat als volgt geïllustreerd kan worden. Matrices

$$B_1 = \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix}, \ B_2 = \begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix}$$

[1]Men kan aantonen dat bij het gebruik van een stabiel algoritme de fout op een individuele eigenwaarde van een Hermitiaanse matrix van dezelfde grootte-orde is als de norm van de matrix (zie [TB97]). Ongunstig in dit opzicht is dat $\|A_0^* A_0\|_2 = \|A_0\|_2^2$.

behoren tot $\mathcal{M}_1(2,2)$, terwijl $B_1 + B_2 = I_2$ van rang twee is en dus niet behoort tot $\mathcal{M}_1(2,2)$.
In wat volgt bespreken we twee methodes om goede lage-rangbenaderingen te bekomen, gebaseerd op de QR-factorisatie, respectievelijk de singuliere-waardenontbinding. We tonen aan dat de tweede methode leidt tot beste benaderingen volgens de spectrale norm en de Frobenius-norm.

12.2.1 Benaderingen via de QR-factorisatie met kolomverwisselingen

Een intuïtieve manier om een rang-k benadering op te stellen bestaat erin om na het bepalen van een QR-factorisatie, $A = QR$, enkel k kolommen van Q en k overeenkomstige rijen van R te behouden. Om er voor te zorgen dat belangrijke informatie behouden blijft, kunnen zowel Algoritme 3 als Algoritme 4 als volgt uitgebreid worden: tussen lijnen 2 en 3 wordt de 2-norm bepaald van alle kolommen van de deelmatrix van R, bestaande uit de rijen j tot m en kolommen j tot n. Vervolgens wordt er voor gezorgd dat de grootste kolom de j-de kolom wordt, door indien nodig een kolomverwisseling toe te passen op de volledige matrix R.
Bij iteratie $j = 1$ wordt de grootste kolom van $R = A$ vooraan geplaatst, waarna nullen gecreëerd worden op posities 2 tot m. Omdat orthogonale transformaties de 2-norm behouden is daarna $|r_{11}|$ gelijk aan de norm van de grootste kolom van A en geldt $|r_{11}| \geq |r_{1j}|,\ j = 2, \ldots, n$. Voor iteratie $j = 2$ wordt dezelfde procedure toegepast op de deelmatrix bekomen door de eerste kolom en de eerste rij van R te verwijderen. Na afloop is $|r_{22}|$ de norm van de grootste kolom van deze deelmatrix, wat $|r_{11}| \geq |r_{22}|$ impliceert. Verder geldt ook $|r_{22}| \geq |r_{2j}|,\ j = 3, \ldots, n$. Deze redenering kan verdergezet worden tot $j = n$. Het omwisselen van kolommen kunnen we interpreteren als het vermenigvuldigen langs rechts met een permutatiematrix. Noemen we het product van alle permutatiematrices P, dan bekomen we met de gewijzigde algoritmen een factorisatie van de vorm

$$AP = QR, \text{ of } A = QRP^T, \tag{12.4}$$

met Q een unitaire matrix, P een permutatiematrix en

$$R = \begin{bmatrix} r_{11} & r_{12} & \ldots & r_{1n} \\ 0 & r_{22} & \ldots & r_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \ldots & r_{nn} \\ 0 & 0 & \ldots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \ldots & 0 \end{bmatrix},$$

waarbij

$$\begin{aligned} &|r_{11}| \geq |r_{22}| \geq \cdots \geq |r_{nn}|, \\ &|r_{ii}| \geq |r_{ij}|, \qquad\qquad i, j \in \{1, \ldots, n\},\ i \leq j. \end{aligned} \tag{12.5}$$

Overeenkomstig (12.5) wegen de eerste rijen van R meer door in de factorisatie dan de andere, in de zin dat ze een belangrijkere bijdrage tot A leveren. Een rang-k benadering A_k^{qr} van A kan dan bekomen worden in de vorm

$$A_k^{\text{qr}} = Q_k R_k P^T,$$

waarbij $Q_k \in \mathbb{C}^{m\times k}$ bestaat uit de eerste k kolommen van Q en R_k uit de eerste k rijen van R. Merk dat P niet expliciet als matrix moet worden opgeslagen. Het volstaat te coderen welke kolommen van plaats gewisseld werden.
Een illustratie van de kwaliteit van de benadering zal gegeven worden in Voorbeeld 12.1.

12.2.2 Benaderingen via de singuliere-waardenontbinding

Ook via de singuliere-waardenontbinding kunnen we een *rang-k benadering* (met $k < r$) van matrix A opstellen, namelijk

$$A_k^{\text{swo}} = U \begin{bmatrix} \sigma_1 & 0 & \cdots & 0 & 0 & \cdots & 0 \\ 0 & \sigma_2 & \cdots & 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \sigma_k & 0 & \cdots & 0 \\ 0 & 0 & \cdots & 0 & 0 & \ldots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 & 0 & \ldots & 0 \end{bmatrix} V^* = \sum_{i=1}^{k} \sigma_i u_i v_i^*. \tag{12.6}$$

De benadering A_k^{swo} is dus volledig bepaald door de eerste k kolommen van U en V, en de grootste k singuliere waarden. De gemaakte fout voldoet aan

$$A_k^{\text{swo}} - A = -\sum_{i=k+1}^{r} \sigma_i u_i v_i^* = \sum_{i=k+1}^{r} \sigma_i (-u_i) v_i^*,$$

waarbij je de meest rechtse uitdrukking als een singuliere-waardenontbinding kan interpreteren. Hieruit volgt dat

$$\|A_k^{\text{swo}} - A\|_2 = \sigma_{k+1}, \quad \|A_k^{\text{swo}} - A\|_F = \sqrt{\sigma_{k+1}^2 + \cdots + \sigma_r^2}. \tag{12.7}$$

De benadering van A door A_k^{swo} is een *beste* rang-k benadering volgens zowel de spectrale norm als de Frobenius-norm, wat in de volgende stelling geformaliseerd wordt.

STELLING 12.2.1 *(beste rang-k benadering - stelling van Eckart–Young–Mirsky)*
Er geldt dat

$$\|A_k^{\text{swo}} - A\|_2 = \min_{B\in\mathcal{M}_k(m,n)} \|B - A\|_2$$

en

$$\|A_k^{\text{swo}} - A\|_F = \min_{B\in\mathcal{M}_k(m,n)} \|B - A\|_F.$$

Bewijs We splitsen het bewijs op in twee delen.
Bewijs voor de spectrale norm. Neem een willekeurige matrix $B \in \mathcal{M}_k(m,n)$, die we kunnen uitdrukken als $B = XY^*$, met $X \in \mathbb{C}^{m\times k}$ en $Y \in \mathbb{C}^{n\times k}$. Er bestaat een vector $w \in \mathbb{C}^n$, $\|w\|_2 = 1$, zodat tegelijkertijd voldaan is aan $w \in \mathcal{R}([v_1 \ \cdots \ v_{k+1}])$ en $w \in \mathcal{N}(Y^*)$, omdat de eerste ruimte dimensie $k+1$ heeft en de tweede ruimte dimensie $n-k$, waarbij de som van de dimensies groter is dan n. We kunnen w uitdrukken als $w = \sum_{i=1}^{k+1} c_i v_i$. Omdat de verzameling $\{v_1, \ldots, v_{i+1}\}$ orthonormaal is, geldt dat $1 = \|w\|_2^2 = \sum_{i=1}^{k+1} c_i^2$. Nu is, omdat de spectrale norm geïnduceerd is door de 2-norm, voldaan aan

$$\|B - A\|_2 \geq \frac{\|(B-A)w\|_2}{\|w\|_2} = \|(XY^* - A)w\|_2 = \|Aw\|_2.$$

Bijgevolg geldt

$$\|B - A\|_2 \geq \left\| \sum_{i=1}^{k+1} c_i A v_i \right\|_2 = \left\| \sum_{i=1}^{k+1} c_i \sigma_i u_i \right\|_2 \geq \sigma_{k+1} \left\| \sum_{i=1}^{k+1} c_i u_i \right\|_2 = \sigma_{k+1} \sqrt{\sum_{i=1}^{k+1} c_i^2} = \sigma_{k+1},$$

waarbij we gebruik gemaakt hebben van de orthonormaliteit van de kolommen van U. Een vergelijking met (12.7) leert ons dat B geen betere benadering is dan A_k^{swo}.

Bewijs voor de Frobenius-norm. Opnieuw beschouwen we een willekeurige matrix $B \in \mathcal{M}_k(m,n)$ en stellen we $p = \min(m,n)$. Voor $i \in \{2,3,\ldots,p-k\}$ geldt de volgende eigenschap, waarbij we met subscript ℓ en $(\cdot)_\ell$ en superscript `swo` de beste rang-ℓ benadering van een matrix volgens de spectrale norm aanduiden, en met $\sigma_i(\cdot)$ de i-de singuliere waarde van het matrix-argument bedoelen:

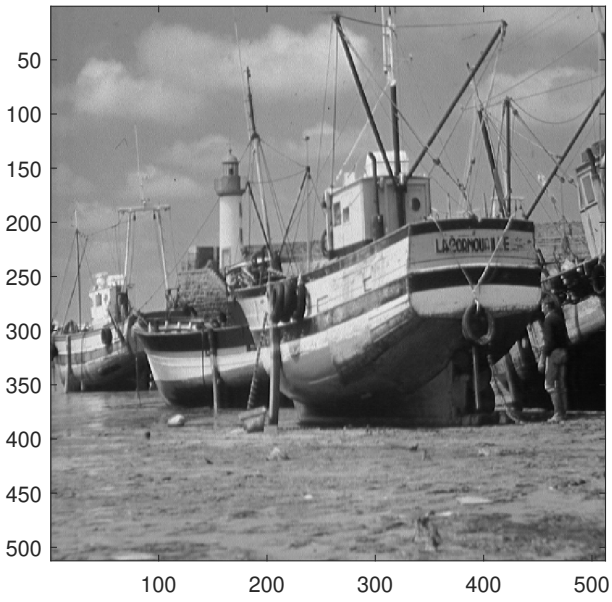
$$\begin{aligned}
\sigma_i(B-A) &= \left\|(A-B) - (A-B)_{i-1}^{\text{swo}}\right\|_2 \\
&= \|A - \underbrace{\left(B + (A-B)_{i-1}^{\text{swo}}\right)}_{\text{rang}(\cdot)\leq i+k-1} \|_2 \\
&\geq \|A - A_{i+k-1}^{\text{swo}}\|_2 \\
&= \sigma_{k+i}(A).
\end{aligned}$$

Eveneens geldt $\sigma_1(B-A) \geq \sigma_{k+1}(A)$. Hiervan gebruikmakend vinden we dat

$$\begin{aligned}
\|B-A\|_F^2 &= \textstyle\sum_{i=1}^{p} \sigma_i(B-A)^2 \\
&\geq \textstyle\sum_{i=1}^{p-k} \sigma_i(B-A)^2 \\
&\geq \textstyle\sum_{i=1}^{p-k} \sigma_{k+i}(A)^2 \\
&= \textstyle\sum_{i=k+1}^{p} \sigma_i(A)^2 = \|A_k^{\text{swo}} - A\|_F^2.
\end{aligned}$$

We komen weer tot de conclusie dat de benadering B van A niet beter kan zijn dan de benadering A_k^{swo}. □

Een gevolg van Stelling 12.2.1 is dat voor elke k voldaan is aan $\|A_k^{\text{swo}} - A\| \leq \|A_k^{\text{qr}} - A\|$ voor zowel de spectrale als de Frobenius-norm. Echter, de rekenkost voor het bepalen van A_k^{qr} is significant kleiner.



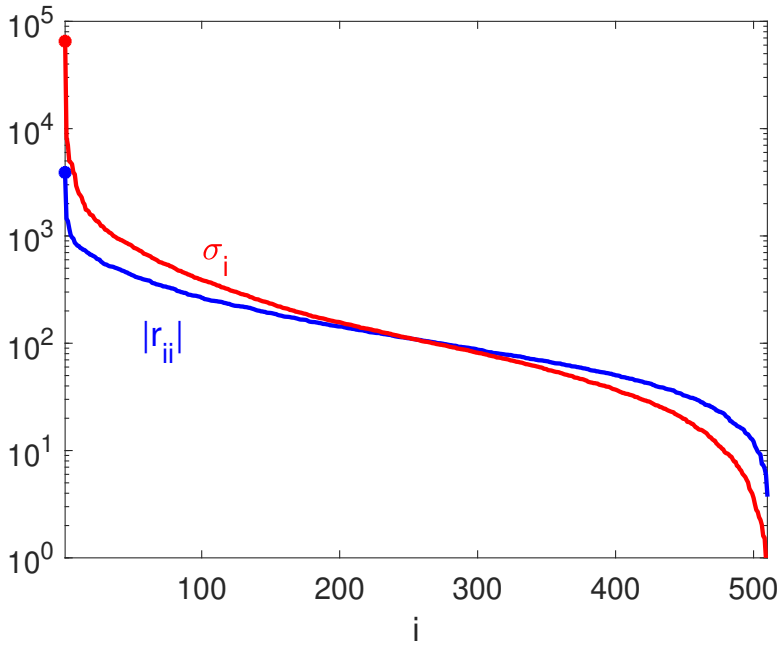
Figuur 12.1: 'Fishing boat' benchmark uit het domein van de beeldverwerking, zwart-wit versie, resolutie 512×512.

Voorbeeld 12.1 *In Figuur 12.1 wordt een zwart-wit beeld*[2]*, weergegeven met een resolutie van* 512*-bij-*512 *pixels, dat voorgesteld kan worden door een matrix* $A \in \mathbb{R}^{512\times512}$*. Met elk element van* A *komt een grijswaarde overeen, voorgesteld door een natuurlijk getal tussen* 0 *(zwart) en* 255 *(wit). Merk dat zulk getal voorgesteld kan worden door* 8 *bits of* 1 *byte.*
Van matrix A *berekenen we zowel een QR-factorisatie met kolomverwisselingen (dit is ontbinding (12.4)) als een singuliere-waardenontbinding. In Figuur 12.2 worden de diagonaalelementen van matrix* R *en de singuliere waarden weergegeven. De sterke daling in functie van index* i *duidt op potentieel om de data voorgesteld door matrix* A *te comprimeren.*
We beschouwen de lage-rangbenaderingen A_k^{qr} *en* A_k^{swo}*. In Figuur 12.3 worden in functie van de rang* k *van de benaderingen de relatieve fouten*

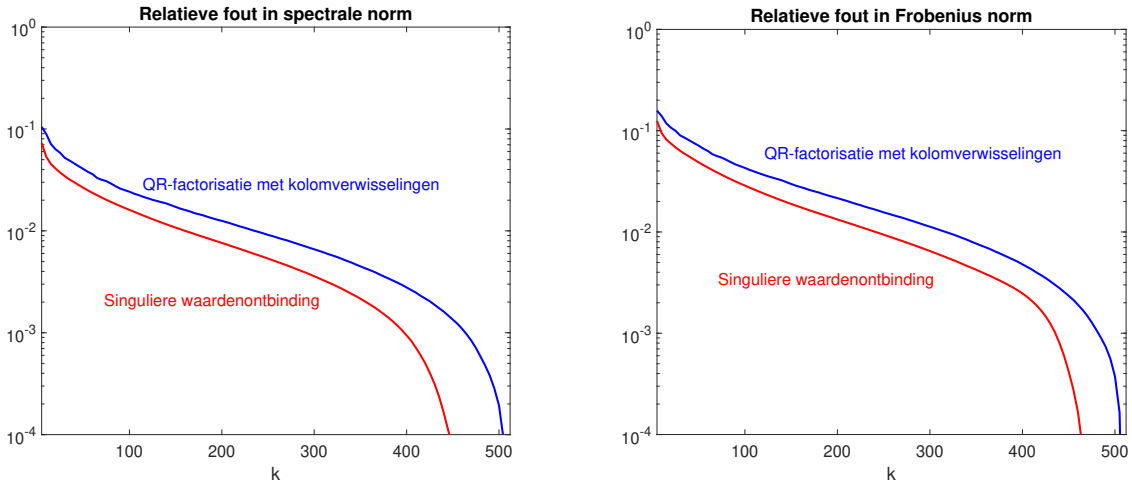
$$\frac{\|\tilde{A}_k^{\text{qr}} - A\|}{\|A\|}, \quad \frac{\|\tilde{A}_k^{\text{swo}} - A\|}{\|A\|}, \text{ met} \|\cdot\| = \|\cdot\|_F \text{ en } \|\cdot\|_2, \tag{12.8}$$

weergegeven, waarbij $\tilde{A}_k^{\text{qr}}$ *en* $\tilde{A}_k^{\text{swo}}$ *uit* A_k^{qr} *en* A_k^{swo} *bekomen worden door alle elementen af te ronden op een getal tussen* 0 *en* 255*. Voor eenzelfde rang leidt de getrunceerde singuliere-waardenontbinding tot een kleinere fout, wat consistent is met Stelling 12.2.1.*
In Figuur 12.4 tonen we het gereconstrueerde beeld op basis van benadering A_k^{swo}*, waarbij*

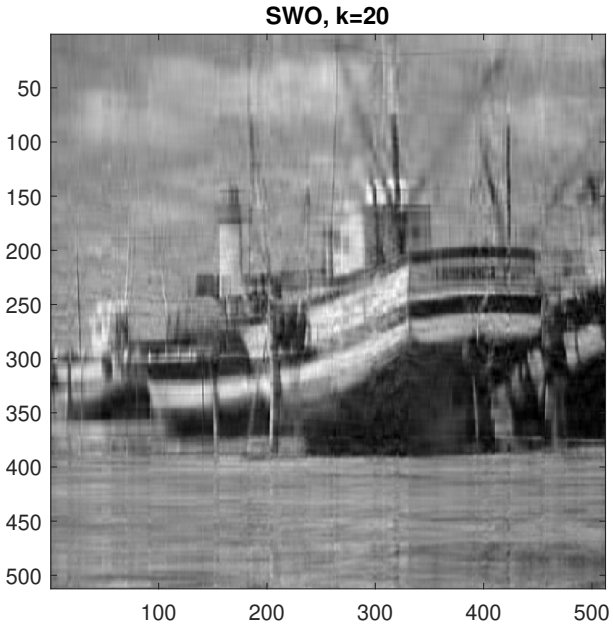
[2]De zgn. 'Fishing boat' benchmark, USC-SIPI Image Database, Signal and Image Processing Institute, Viterbi School of Engineering, University of Southern California



Figuur 12.2: Absolute waarde van de diagonaalelementen van R in ontbinding (12.4) en singuliere waarden van matrix A, overeenkomstig Figuur 12.1.



Figuur 12.3: Relatieve fouten (12.8) van de lage-rangbenaderingen A_k^{qr} en A_k^{swo} van matrix A, overeenkomstig Figuur 12.1.



Figuur 12.4: Benadering van Figuur 12.1, bekomen uit de grijswaarden in matrix $\tilde{A}_k^{\text{swo}}$, voor $k = 20$.

$k = 20$. *Matrix A bestaat uit* $512 \times 512 = 262144$ *getallen. De drie factoren van* A_{20}^{swo} *kunnen voorgesteld worden door* $512 \times 20 + 20 + 20 \times 512 = 20500$ *getallen.*

12.3 Principal component analysis

Principal component analysis (PCA) is een belangrijke techniek voor het reduceren van de dimensie van (discrete) data-sets, die we in deze sectie toelichten.

We vertrekken van een data-matrix $X \in \mathbb{R}^{m \times n}$, met $m \geq n$. Hierbij komt elke rij overeen met een experiment (of een individu, een patiënt, een sample,..., afhankelijk van de dataset). Elke kolom komt overeen met een meting van een "quantity of interest", een uitlezing door een bepaalde sensor, een score gegeven voor een bepaald feature etc. Matrix X heeft dus de vorm

$$X = \begin{bmatrix} x_1^{(1)} & x_2^{(1)} & \cdots & x_n^{(1)} \\ x_1^{(2)} & x_2^{(2)} & \cdots & x_n^{(2)} \\ \vdots & \vdots & & \vdots \\ x_1^{(m)} & x_2^{(m)} & \cdots & x_n^{(m)} \end{bmatrix},$$

waarbij $x^{(i)} = \begin{bmatrix} x_1^{(i)} & \cdots & x_n^{(i)} \end{bmatrix}^T$ bijvoorbeeld alle metingen voorstelt horende bij het i-de experiment. We maken de volgende veronderstelling, waaraan we steeds kunnen voldoen na eventuele verschuivingen van meetwaarden in elke kolom.

Veronderstelling 12.1 *Het (sample) gemiddelde van elke kolom van X is gelijk aan nul.*

De waarnemingen bij de verschillende experimenten kunnen we voorstellen door een puntenwolk in $\mathbb{R}^n$. We zoeken nu naar belangrijke richtingen in deze ruimte met betrekking tot spreiding van de puntenwolk, wat ons toelaat om de dimensie n te reduceren. In wat volgt wordt dit mathematisch gespecificeerd.
Een richting kunnen we voorstellen door een vector $r = (r_1, \ldots, r_n) \in \mathbb{R}^n$ waarvoor $\|r\|_2 = 1$. De orthogonale projectie van het i-de datapunt (overeenkomstig experiment i) op de ruimte $< r >$ wordt gegeven door $(r^T x^{(i)})\ r$, waarvan de afstand tot de oorsprong $r^T x^{(i)}$ bedraagt.

- De voornaamste richting r_1, of principal component, is de richting waarvoor de variantie ("sample variance") maximaal is over de dataset. Deze richting wordt beschreven door

$$r_1 = \arg \max_{\|r\|_2=1} \sum_{i=1}^{m} \left(r^T x^{(i)}\right)^2 = \arg \max_{\|r\|_2=1} \|Xr\|_2^2 = \arg \max_{\|r\|_2=1} r^T X^T X r.$$

  Maken we gebruik van de singuliere-waardenontbinding, $X = U\Sigma V^T$, en stellen we $r = Vy$, dan kunnen we het optimalisatieprobleem herschrijven als

$$\hat{y} = \arg \max_{\|y\|_2=1} y^T \Sigma^T \Sigma y,$$

  met oplossing $\hat{y} = [1\ 0\ \cdots\ 0]^T$ en dus $r_1 = v_1$, de rechter singuliere vector horende bij de grootste singuliere waarde.
  Merk dat we $\{r_1^T x^{(i)}\}_{i=1}^m = \{v_1^T x^{(i)}\}_{i=1}^m$ ook kunnen interpreteren als meetwaarden voor een *virtueel feature*, in de vorm van een lineaire combinatie $r^T x$ van de originele features met $\|r\|_2 = 1$, waarvoor de variantie maximaal is.

- De tweede richting, r_2, is de richting orthogonaal tot r_1, waarvoor de variantie maximaal is. We kunnen dit wiskundig uitdrukken als

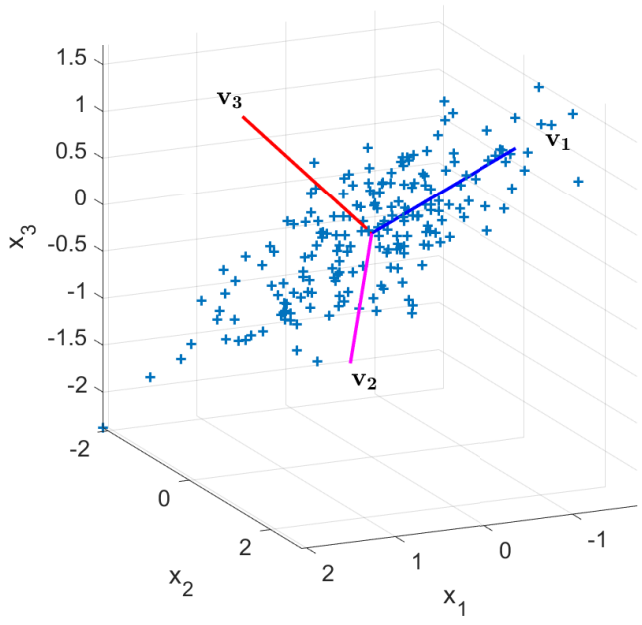
$$r_2 = \arg \max_{\|r\|_2=1,\ r^T v_1=0} r^T X^T X r.$$

  Stellen we opnieuw $r = Vy$, dan verkrijgen we

$$\tilde{y} = \arg \max_{\|y\|_2=1,\ y_1=0} y^T \Sigma^T \Sigma y,$$

  met als oplossing $\tilde{y} = [0\ 1\ 0\ \cdots\ 0]^T$ en bijgevolg $r_2 = v_2$, de rechter singuliere vector horende bij de tweede singuliere waarde. Merk dat we een gelijkaardig optimalisatieprobleem reeds hebben opgelost in de context van graafpartitionering.

- De derde richting wordt bepaald als richting orthogonaal tot $< v_1, v_2 >$, waarvoor de variantie maximaal is, wat ons brengt tot de rechter singuliere vector v_3. We kunnen dit proces verder zetten tot een nieuwe basis $< v_1, \ldots, v_n >$ is opgebouwd.



Figuur 12.5: Dataset met waarden in $\mathbb{R}^3$, met aanduiding van de hoofdrichtingen.

De hoofdrichting*en* of principal component*s* zijn dus, in afnemende mate van belangrijkheid: $v_1,\ v_2\ , v_3,\ \ldots$

Voorbeeld 12.2 *Figuur 12.5 visualiseert een dataset, waarbij $X \in \mathbb{R}^{200\times 3}$. De drie hoofdrichtingen worden aangeduid.*

Uit de singuliere-waardenontbinding van X halen we dat

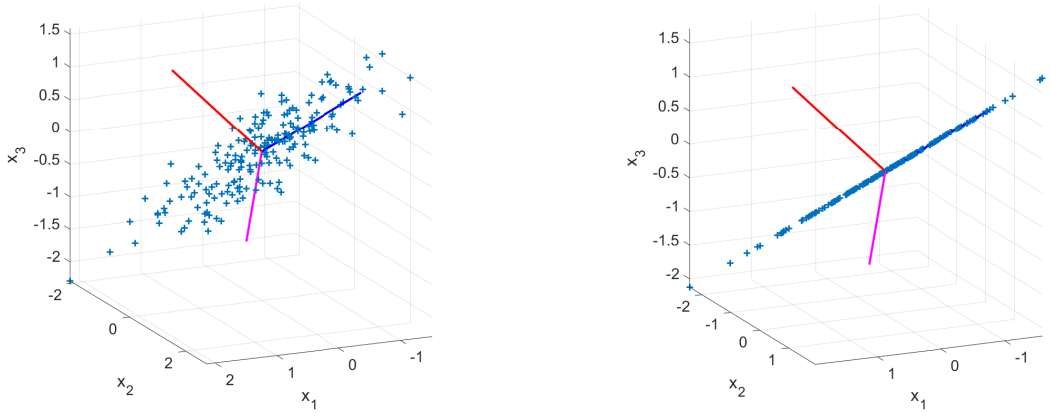
$$\left[\ x^{(1)}\quad \cdots\quad x^{(m)}\ \right] = X^T = \sum_{i=1}^{n} v_i \left(\sigma_i u_i^T\right), \tag{12.9}$$

waaruit volgt

$$x^{(j)} = (\sigma_1 u_{1j})\, v_1 + (\sigma_2 u_{2j})\, v_2 + \cdots + (\sigma_n u_{nj})\, v_n, \quad j = 1, \ldots, m. \tag{12.10}$$

Uit de linker singuliere vectoren, gewogen met de singuliere waarden, halen we dus de coördinaten van de datapunten in de basis $\{v_1, v_2, \ldots, v_n\}$. Deze weging, samen met het feit $\|u_i\|_2 = 1,\ \forall i$, zijn consistent met de betekenis van hoofdrichtingen.
Een goede manier om de dimensie n van de dataset te reduceren, bestaat erin om in de ontbindingen (12.9)-(12.10) enkel de k eerste termen te beschouwen, met $k < n$. Merk dat dit overeenkomt met het afknotten van de singuliere waardenontbinding van X, i.e. de benadering $X \approx X_k^{\text{swo}}$.

Voorbeeld 12.3 *We komen terug op didactisch Voorbeeld 12.2. In het linker, respectievelijk rechter paneel van Figuur 12.6 tonen we de datasets X_2^{swo} en X_1^{swo}, bekomen door*



Figuur 12.6: Datasets X_2^{swo} (links) en X_1^{swo} (rechts), bekomen uit dataset X, gevisualiseerd in Figuur 12.5.

de singuliere-waardenontbinding van X af te kappen na twee termen, respectievelijk één term. Dit betekent dat in het eerste geval enkel de componenten in de richting van v_1 en v_2 beschouwd worden, in het tweede geval enkel de component volgens v_1.

Een veel voorkomende toepassing van PCA bestaat uit het verminderen van het aantal variabelen bij het schatten van een functie. Stel dat we op basis van meetwaarden $(x_1, \ldots, x_n)$ een voorspelling willen maken van grootheid y, wat neerkomt op het schatten van een onbekende functie $y = f(x_1, \ldots, x_n)$. Hoe hoger het aantal variabelen, hoe moeilijker deze taak wordt en hoe meer training data nodig is (in bepaalde gevallen stijgt deze exponentieel met de dimensie). Echter in de meeste toepassingen is de distributie van meetwaarden niet uniform, zijn er bij benadering afhankelijkheden tussen verschillende metingen en kan met PCA een efficiënte reductie gemaakt worden. Gebruik makend van de benadering van de dataset in Figuur 12.6 (rechts) moeten we dus bijvoorbeeld slechts een functie in één veranderlijke bepalen, $y = F(w)$, met w de component volgens v_1.

12.4 Andere ijle representaties en benaderingen

We hernemen de setting van Sectie 12.2 en beschouwen de benadering van een discrete functie, voorgesteld door de elementen van matrix $A \in \mathbb{R}^{m \times n}$.

12.4.1 Fourier-benaderingen

De tweedimensionale discrete cosinustransformatie (DCT) laat toe om de gegeven functie te ontbinden in periodieke basisfuncties, wat beschreven wordt door[3]

$$A_{ij} = \sum_{k=0}^{m-1} \sum_{l=0}^{n-1} c_k d_l B_{kl} \cos\left((i+\tfrac{1}{2})k\tfrac{\pi}{m}\right) \cos\left((j+\tfrac{1}{2})l\tfrac{\pi}{n}\right), \quad 0 \le i \le m-1, \;\; 0 \le j \le n-1, \tag{12.11}$$

met

$$c_k = \begin{cases} \frac{1}{\sqrt{m}}, & k = 0, \\ \sqrt{\frac{2}{m}}, & \text{anders} \end{cases} \quad \text{en } d_l = \begin{cases} \frac{1}{\sqrt{n}}, & l = 0, \\ \sqrt{\frac{2}{n}}, & \text{anders} \end{cases} .$$

De coëfficiënten B_{kl} horende bij de basisfunctie $\cos((\cdot + 1/2)k\pi/m)\cos((\cdot + 1/2)l\pi/n)$ kunnen berekend worden als

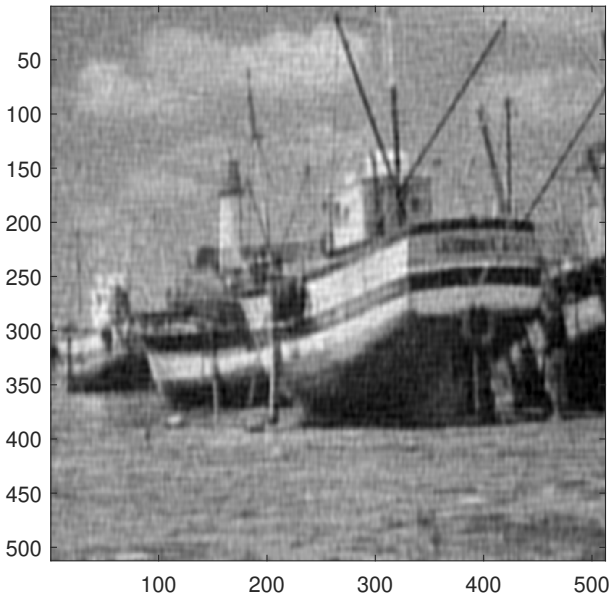
$$B_{kl} = c_k d_l \sum_{i=0}^{m-1} \sum_{j=0}^{n-1} A_{ij} \cos\left((i+\tfrac{1}{2})k\tfrac{\pi}{m}\right) \cos\left((j+\tfrac{1}{2})l\tfrac{\pi}{n}\right), \quad 0 \le k \le m-1, \;\; 0 \le l \le n-1.$$

De transformatie van matrix A naar matrix B brengt vaak spaarsheid aan het licht, in de zin dat de meeste elementen van B zeer klein zijn ten opzichte van de dominante elementen. Om dit te illustreren beschouwen we opnieuw matrix A, overeenkomstig de foto van Figuur 12.1. We bereken matrix B en vervangen alle elementen door nul, behalve de $64 \times 64 = 4096$ grootste in absolute waarde, en voeren vervolgens de inverse transformatie door via formules (12.11). De resulterende benadering wordt gevisualiseerd in Figuur 12.7. Voor beelcompressie is de DCT duidelijk efficiënter dan lage-rangbenaderingen.
Als n en m machten zijn van twee kunnen de DCT en de inverse DCT uitgerekend worden met een algoritme gebaseerd op de Fast Fourier Transform (FFT).
De oorspronkelijke jpeg-standaard maakt gebruik van de DCT, toegepast op 8-bij-8 blokken waarin de matrix wordt ingedeeld.

12.4.2 Wavelets

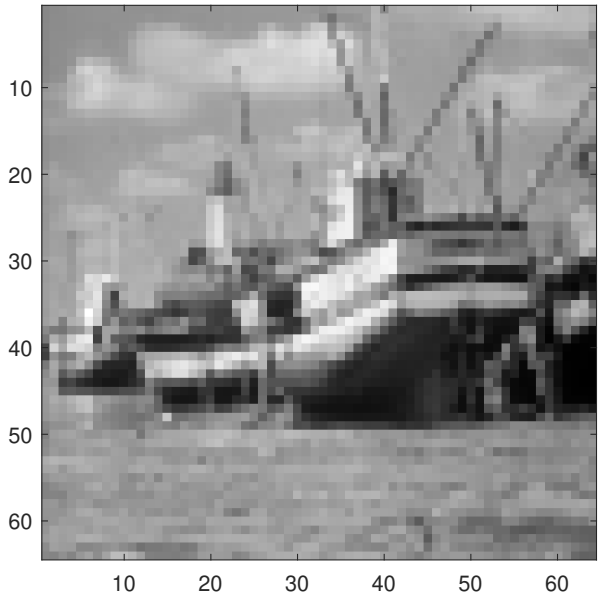
Een wavelettransformatie leidt tot een ontbinding op basis van resolutie. Daarbij zijn alle basisfuncties gescaleerde en/of verschoven kopieën van één functie die slechts op een eindig interval verschillend is van nul, de zogenaamde *mother wavelet*. Voor een wiskundige theorie verwijzen we naar [Dau92]. We illustreren enkel wat basisprincipes aan de hand van de transformatie van een zwart-wit beeld met de zogenaamde Haar-wavelettransformatie, de eenvoudigste wavelettransformatie.
We veronderstellen voor de eenvoud $m = n$ en dat m een macht van twee is. De transformatie kan opgesplitst worden in verschillende stappen. In de eerste stap bekomen we een benadering van het beeld door de grijswaarden van blokken van 2-bij-2 pixels uit te

[3]Let op: in tegenstelling tot de conventie bij matrixrekenen beginnen de indices met nul. In de context van beeldverwerking is dit natuurlijk omdat coëfficiënt B_{00} bijvoorbeeld overeenkomt met een constante component in de ontbinding, de zgn. DC component.



Figuur 12.7: Benadering van Figuur 12.1 op basis van de DCT, waarbij de 64×64 grootste coëfficiënten (elementen van matrix B) weerhouden werden.

middelen. Het aantal coëfficiënten om die benadering voor te stellen is $m/2 \times m/2$. De andere $3m^2/4$ coëfficiënten bevatten informatie (in de vorm van verschilwaarden) die toelaat om het originele beeld te reconstrueren. Als we enkel de eerste reeks van coëfficiënten zouden bijhouden of doorsturen naar een gebruiker, kan op basis daarvan een beeld met resolutie $m/2 \times m/2$ bekomen worden. In de tweede stap passen we dezelfde bewerkingen toe op de $m/2$-bij-$m/2$ deelmatrix bestaande uit de eerste reeks van coëfficiënten: uitmiddelen in blokken van vier pixels en informatie voor de reconstructie coderen. Twee keer uitmiddelen resulteert in een benadering van resolutie $m/4 \times m/4$. Omgekeerd zorgt één reconstructiestap voor een beeld met resolutie $m/2 \times m/2$, twee reconstructiestappen voor het originele beeld. We kunnen de hele procedure herhalen tot er uiteindelijk over alle pixels uitgemiddeld is. In Figuur 12.8 wordt de benadering van Figuur 12.1 weergegeven, na drie stappen van de transformatie.
De jpeg2000 standaard is gebaseerd op de zogenaamde Deaubechies-wavelettransformatie.



Figuur 12.8: Benadering van Figuur 12.1 met resolutie 64×64, op basis van drie stappen van de Haar-wavelettransformatie.

Deel IV

Appendices

Bijlage A

Vectoren en matrices

In deze appendix herhalen we belangrijke terminologie en leggen we uit hoe basisdefinities en eigenschappen van reële matrices veralgemenen naar complexe matrices.

A.1 Elementaire definities en notaties

Gegeven een getal $a \in \mathbb{C}$ dat ontbonden kan worden als $a = x + \imath y$, met $x, y \in \mathbb{R}$ en $\imath$ de imaginaire eenheid, noemen we $\bar{a} = x - \imath y$ het complex toegevoegde getal van a en $|a| = \sqrt{x^2 + y^2}$ de modulus. Voor $a \neq 0$ kunnen we het getal op een unieke manier schrijven in polaire vorm,

$$a = |a|e^{\imath\theta} = |a|(\cos(\theta) + \imath \sin(\theta))$$

met $\theta \in (-\pi,\ \pi]$ het argument van a.
Een vector x in $\mathbb{C}^n$ stellen we voor als

$$x \in \mathbb{C}^n,\ x = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix},\ x_i \in \mathbb{C},\ i = 1, 2, \ldots, n,$$

en een m-bij-n matrix A als

$$A \in \mathbb{C}^{m\times n},\ A = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{bmatrix} = \begin{bmatrix} a_1 & a_2 & \ldots & a_n \end{bmatrix},$$

$$a_{ij} \in \mathbb{C},\ i = 1, 2, \ldots, m,\ j = 1, 2, \ldots, n.$$

In de tweede voorstelling staat $a_i \in \mathbb{C}^m,\ i \in \{1, 2, \ldots, n\}$, voor de i-de kolom van A.

Een *matrix-vectorvermenigvuldiging* Ax kunnen we aanzien als een matrix die inwerkt op een vector om een nieuwe vector te bekomen:

$$A : \mathbb{C}^n \longrightarrow \mathbb{C}^m : x \longmapsto b = Ax, \text{ met } b_i = \sum_{j=1}^{n} a_{ij}x_j, \ i = 1, 2, \ldots, m.$$

Dit komt overeen met een matrix A als representatie van een lineaire afbeelding tussen $\mathbb{C}^n$ en $\mathbb{C}^m$. We kunnen matrix-vectorvermenigvuldiging Ax ook interpreteren als vector x die inwerkt op matrix A. Het product is een *lineaire combinatie* van de kolommen van A. De coëfficiënten van de lineaire combinatie zijn de coëfficiënten van vector x:

$$x : \mathbb{C}^{m\times n} \longrightarrow \mathbb{C}^m : A = \left[\begin{array}{cccc} a_1 & a_2 & \ldots & a_n \end{array} \right] \longmapsto b = Ax = \sum_{j=1}^{n} x_j a_j \ . \tag{A.1}$$

Het product van twee matrices $A \in \mathbb{C}^{m\times l}$ en $B \in \mathbb{C}^{l\times n}$ is een matrix $C \in \mathbb{C}^{m\times n}$, waarbij $c_{ij} = \sum_{k=1}^{l} a_{ik}b_{kj}$. Ook hier kan je de matrix-matrix vermenigvuldiging interpreteren als een bewerking op de kolommen van A. De j-de kolom van matrix $C = AB$ is namelijk een lineaire combinatie van de kolommen van A. De coëfficiënten van deze combinatie staan in de j-de kolom van B.
De *getransponeerde van vector* $x \in \mathbb{C}^{n\times 1}$ is de vector $x^T = [x_1 \ x_2 \ \ldots \ x_n] \in \mathbb{C}^{1\times n}$. De *complex toegevoegd getransponeerde van vector* $x \in \mathbb{C}^{n\times 1}$ is de vector $x^* = [\bar{x}_1 \ \bar{x}_2 \ \ldots \ \bar{x}_n] \in \mathbb{C}^{1\times n}$.
De *getransponeerde en complex toegevoegd getransponeerde van matrix* $A \in \mathbb{C}^{m\times n}$ worden aangeduid als A^T, respectievelijk A^*, waarbij

$$A^T = \begin{bmatrix} a_{11} & a_{21} & \cdots & a_{m1} \\ a_{12} & a_{22} & \cdots & a_{m2} \\ \vdots & \vdots & \ddots & \vdots \\ a_{1n} & a_{2n} & \cdots & a_{mn} \end{bmatrix} \in \mathbb{C}^{n\times m}, \quad A^* = \begin{bmatrix} \bar{a}_{11} & \bar{a}_{21} & \cdots & \bar{a}_{m1} \\ \bar{a}_{12} & \bar{a}_{22} & \cdots & \bar{a}_{m2} \\ \vdots & \vdots & \ddots & \vdots \\ \bar{a}_{1n} & \bar{a}_{2n} & \cdots & \bar{a}_{mn} \end{bmatrix} \in \mathbb{C}^{n\times m}.$$

Er geldt: $(Ax)^* = x^*A^*$ en $(AB)^* = B^*A^*$, alsook $(Ax)^T = x^TA^T$ en $(AB)^T = B^TA^T$.
Een matrix is *vierkant* als $m = n$.
Een vierkante matrix A is *Hermitiaans* als $A^* = A$ en symmetrisch als $A = A^T$.
Als A een reële matrix is, dan is $A^* = A^T$.

A.2 Bereik, nulruimte, rang, inverse

Het *bereik* of de *kolomruimte* van matrix $A \in \mathbb{C}^{m\times n}$ is de verzameling van alle vectoren die kunnen geschreven worden als lineaire combinatie van de kolommen, dus als Ax voor één of andere x:

$$\mathcal{R}(A) = \{y \in \mathbb{C}^m : \ \exists x \in \mathbb{C}^n, y = Ax\}.$$

Definiëren we

$$< a_1, a_2, \ldots, a_n >= \left\{ y \in \mathbb{C}^m : \ \exists (c_1, \ldots, c_n) \in \mathbb{C}^n, \ y = \sum_{i=1}^{n} c_i a_i \right\},$$

dan geldt $\mathcal{R}(A) =< a_1, \ldots, a_n >$.
De *nulruimte* van matrix $A \in \mathbb{C}^{m \times n}$ is

$$\mathcal{N}(A) = \{x \in \mathbb{C}^n : Ax = 0\}.$$

De *rang* (of *kolomrang*) van een matrix is de dimensie van de kolomruimte. Meestal beschouwen we in deze cursus matrices met $m \geq n$ en zeggen we dat matrix A van volle rang is als de rang gelijk is aan n, d.w.z. dat de kolommen van A lineair onafhankelijk zijn.
De *eenheidsmatrix* van orde m, genoteerd als I_m (of I als duidelijk is wat de orde is) is de matrix

$$I = \begin{bmatrix} 1 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 \end{bmatrix} \in \mathbb{C}^{m \times m}.$$

Een *niet-singuliere* matrix A is een vierkante matrix ($A \in \mathbb{C}^{m \times m}$) van volle rang. Zulke matrix heeft een *inverse*, d.w.z. een matrix genoteerd met A^{-1} zodat

$$AA^{-1} = A^{-1}A = I.$$

Voor $A \in \mathbb{C}^{m \times m}$ zijn volgende voorwaarden equivalent:

1. A heeft een inverse A^{-1},
2. $\text{rang}(A) = m$,
3. $\mathcal{R}(A) = \mathbb{C}^m$,
4. $\mathcal{N}(A) = \{0\}$, met 0 de nulvector,
5. (het getal) 0 is geen eigenwaarde van A,
6. (het getal) 0 is geen singuliere waarde van A.

Voor inverteerbare vierkante matrices A en B van gelijke dimensies geldt: $(AB)^{-1} = B^{-1}A^{-1}$. Voor $(A^{-1})^* = (A^*)^{-1}$ en $(A^{-1})^T = (A^T)^{-1}$ gebruikt men soms de verkorte notatie A^{-*} en A^{-T}.

Als matrix $A \in \mathbb{C}^{m \times n}$, met $m \geq n$, van volle kolomrang is, dan noemen we

$$A^+ = (A^*A)^{-1}A^* \in \mathbb{C}^{n \times m} \tag{A.2}$$

de pseudo-inverse van A. Deze matrix is een *linker*-inverse omdat $A^+A = I_n$. Voor $m > n$ geldt echter niet dat $AA^+ = I_m$, matrix AA^+ is dan niet eens van volle rang.

A.3 Inwendig product, orthogonaliteit, normen

Het *inwendig product* van twee vectoren x en $y \in \mathbb{C}^m$ is het product van de complex toegevoegd getransponeerde van x met y:

$$(x, y) = x^* y = \sum_{i=1}^{m} \bar{x}_i y_i.$$

Bij reële vectoren herleidt het inwendig product zich tot $(x, y) = x^T y$. De *Euclidische lengte* van een vector $x \in \mathbb{C}^m$ wordt gedefinieerd als

$$\|x\| = \sqrt{x^* x} = \sqrt{\sum_{i=1}^{m} |x_i|^2}. \tag{A.3}$$

Twee vectoren x en y zijn *orthogonaal* als $(x, y) = x^* y = 0$. Vector x is orthogonaal tot (of ten opzichte van) deelverzameling $\mathcal{D} \subset \mathbb{C}^m$, genoteerd $x \perp \mathcal{D}$, als x orthogonaal is tot elk element van $\mathcal{D}$, dit wil zeggen $x^* y = 0,\ \forall y \in \mathcal{D}$. Twee verzamelingen van vectoren $X = \{x_1, \ldots, x_n\}$ en $Y = \{y_1, \ldots, y_m\}$ zijn *orthogonaal* als elke vector van X orthogonaal is tot elke vector van Y:

$$x_i^* y_j = 0,\ \forall i \in \{1, \ldots, n\},\ \forall j \in \{1, \ldots, m\}.$$

Een verzameling vectoren $X = \{x_1, x_2, \ldots, x_n\}$ is *orthogonaal* als elk element orthogonaal is op elk ander element van de verzameling:

$$x_i^* x_j = 0\ ,\ i \neq j.$$

Een verzameling vectoren $X = \{x_1, x_2, \ldots, x_n\}$ is *orthonormaal* als ze orthogonaal is en voor alle $x_i \in X$ geldt dat $\|x_i\| = 1$. We noteren dit met de Kronecker-δ als $x_i^* x_j = \delta_{ij}$, waarbij

$$\delta_{ij} = \begin{cases} 0 & \text{als } i \neq j, \\ 1 & \text{als } i = j. \end{cases}$$

Een vierkante matrix $U \in \mathbb{C}^{m \times m}$ is *unitair* als hij orthonormale kolommen heeft. Er geldt dan dat $U^* U = I$ of $U^{-1} = U^*$. Er geldt dan automatisch ook dat $UU^* = I$, dus de rijen van U zijn ook orthonormaal. Een reële unitaire matrix, d.w.z. een matrix $Q \in \mathbb{R}^{m \times m}$ die voldoet aan $Q^T Q = I$, noemen we een *orthogonale matrix*. Merk op dat een zgn. orthogonale matrix *orthonormale* kolommen heeft.

Gegeven een deelruimte $\mathcal{D}$ van $\mathbb{C}^m$, noemen we de deelruimte

$$\mathcal{D}^\perp = \{x \in \mathbb{C}^m :\ x \perp \mathcal{D}\} = \{x \in \mathbb{C}^m :\ x^* y = 0, \forall y \in \mathcal{D}\}$$

het *orthogonaal complement* van $\mathcal{D}$. Is $\mathcal{D} =< a_1, \ldots, a_n >$ een deelruimte van dimensie $n < m$ en

$$A = [a_1\ \cdots\ a_n] = QR$$

een volledige QR-factorisatie, met $Q = [q_1 \ \ldots \ q_m]$, dan is $\mathcal{D}^\perp =< q_{n+1}, \ldots, q_m >$.

Belangrijke *vectornormen* zijn:

$$\|x\|_1 = \sum_{i=1}^{m} |x_i|, \quad \|x\|_2 = \sqrt{\sum_{i=1}^{m} |x_i|^2} = \sqrt{x^* x}, \quad \|x\|_\infty = \max_{1 \le i \le m} |x_i|,$$

waarbij $x \in \mathbb{C}^m$. Meer algemeen wordt de vector p-norm, met $p \in [1, \infty)$ gedefinieerd door

$$\|x\|_p = \left(\sum_{i=1}^{m} |x_i|^p \right)^{\frac{1}{p}}.$$

De 2-norm is eveneens de norm bepaald door bovenvermeld inwendig product, zie (A.3).

Elke vector p-norm definieert een matrix p-norm via de formule

$$\|A\|_p = \max_{x \neq 0} \frac{\|Ax\|_p}{\|x\|_p},$$

wat de *geïnduceerde matrix p-norm* genoemd wordt. Intuïtief is deze de maximale versterking, gemeten volgens de vector p-norm, die bekomen wordt bij de afbeelding beschreven door matrix A. De meest gebruikte is de geïnduceerde 2-norm, ook wel *spectrale norm* genoemd,

$$\|A\|_2 = \max_{x \neq 0} \frac{\|Ax\|_2}{\|x\|_2}.$$

Er geldt $\|A\|_2 = \sigma_1$, met σ_1 de grootste singuliere waarde van A. De zogenaamde Frobeniusnorm,

$$\|A\|_F = \sqrt{\sum_{i=1}^{m} \sum_{j=1}^{n} |a_{ij}|^2}$$

is geen geïnduceerde norm. Als alle elementen van de matrix in één vector geplaatst worden, komt de Frobenius norm overeen met de 2-norm van deze vector.

Een *unitaire transformatie behoudt de 2-norm en de Frobeniusnorm*. Meer bepaald geldt voor matrices en vectoren met compatibele dimensies het volgende:

$$U \in \mathbb{C}^{m \times m}, U^* U = I \ \Rightarrow \ \|Ux\|_2 = \|x\|_2, \quad \|UA\|_2 = \|A\|_2, \quad \|UA\|_F = \|A\|_F.$$

Werken we louter in $\mathbb{R}^m$, dan wordt deze eigenschap: *een orthogonale transformatie behoudt de 2-norm en de Frobeniusnorm*.

A.4 Eigenwaarden en eigenvectoren

Zij $A \in \mathbb{C}^{m\times m}$ een vierkante matrix. Een vector $x \in \mathbb{C}^m$, $x \neq 0$, waarvoor geldt

$$Ax = \lambda x,$$

met $\lambda \in \mathbb{C}$, wordt een *eigenvector* van A genoemd. De waarde λ is de bijhorende *eigenwaarde*. Het homogene stelsel $(A-\lambda I)x = 0$ heeft dan een van nul verschillende oplossing. Daartoe moet $\det(\lambda I - A)$ nul zijn. De eigenwaarden zijn dus nulpunten van de m-de graadsveelterm $F(z) = \det(zI - A)$. Deze veelterm heeft steeds m nulpunten (die eventueel meervoudig kunnen zijn). Bij elk verschillend nulpunt hoort minstens één eigenvector. Bij meervoudige eigenwaarden kan het aantal lineair onafhankelijke eigenvectoren variëren van 1 tot de meervoudigheid van het nulpunt. Eigenvectoren zijn slechts bepaald op een constante na.
Zij T een niet-singuliere vierkante $m \times m$ matrix. Matrix A en $C = TAT^{-1}$ noemen we *gelijkvormige matrices* (of *equivalente matrices*). Gelijkvormige matrices hebben dezelfde eigenwaarden, zij stellen immers dezelfde lineaire afbeelding voor in een andere basis.
Als matrix A m lineair onafhankelijke eigenvectoren heeft noemen we de matrix *niet-defectief* of *diagonaliseerbaar*. Stel dat dit het geval is en beschouw zulke verzameling van eigenvectoren $\{x_1, \ldots, x_m\}$, waarbij x_i een eigenvector is horende bij eigenwaarde λ_i. Matrix

$$X = \left[\begin{array}{cccc} x_1 & x_2 & \ldots & x_m \end{array}\right] \in \mathbb{C}^{m\times m}$$

is dan inverteerbaar en

$$AX = X \begin{bmatrix} \lambda_1 & 0 & \cdots & 0 \\ 0 & \lambda_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \lambda_m \end{bmatrix} = X\,\mathrm{diag}(\lambda_1, \lambda_2, \ldots, \lambda_m) := XD \tag{A.4}$$

of

$$A = XDX^{-1}.$$

Dit noemt men een *eigenwaardenontbinding* van matrix A. Is de matrix defectief, dan bestaat er een gelijkaardige ontbinding, waarbij de kolommen van X bestaan uit eigenvectoren en veralgemeende eigenvectoren (ook genoemd hoofdvectoren) en D een zgn. Jordancanonieke vorm heeft.

Indien matrix $A \in \mathbb{C}^{m\times m}$ Hermitiaans is, dan zijn alle eigenwaarden reëel. Een Hermitiaanse matrix die voldoet aan

$$x^*Ax > 0,\ \forall x \neq 0,$$

noemen we *Hermitiaans positief definiet* (semi-definiet als $>$ vervangen wordt door $\geq$). De notatie hiervoor is $A > 0$ ($A \geq 0$). Een matrix is Hermitiaans positief definiet (semi-definiet) als en alleen als hij Hermitiaans is en alle eigenwaarden strikt positief (positief

of nul) zijn. Een reële Hermitiaans-positief definiete matrix noemen we een *symmetrisch-positief definiete matrix*, vaak verkort tot simpelweg *positief definiete matrix*.
Een Hermitiaanse matrix is diagonaliseerbaar door een unitaire transformatie, meer specifiek is X in ontbinding (A.4) unitair. Eigenvectoren horende bij verschillende eigenwaarden zijn dan orthogonaal.

De singuliere-waardenontbinding wordt uitvoerig behandeld in Sectie 12.1.

Bijlage B

Conditie en stabiliteit

We herhalen en illustreren begrippen gerelateerd aan de conditie van een numeriek probleem en de stabiliteit van een algoritme.

B.1 Conditie

We vertrekken van een numeriek probleem dat beschreven kan worden als een afbeelding f die één of ander gegeven $x \in X$ afbeeldt op een resultaat $f(x) \in Y$. De conditie van het probleem drukt uit hoe gevoelig het resultaat is voor veranderingen (perturbaties) aan het gegeven, en kan gekarakteriseerd worden aan de hand van conditiegetallen.
Het *conditiegetal voor de relatieve fout* wordt gedefinieerd als

$$K = \lim_{\delta \to 0+} \sup_{\|\delta x\| \leq \delta} \frac{\frac{\|\delta f(x)\|}{\|f(x)\|}}{\frac{\|\delta x\|}{\|x\|}},$$

waarbij $\delta f(x) = f(x + \delta x) - f(x)$ en $\|\cdot\|$ een norm in de ruimten onder beschouwing[1] is. In het speciale geval waarbij f een afleidbare functie is van $\mathbb{R}^n$ naar $\mathbb{R}^m$ is

$$K = \frac{\|J_f(x)\|\|x\|}{\|f(x)\|},$$

met $J_f \in \mathbb{R}^{m \times n}$ de Jacobiaan-matrix van f.
Het numerieke probleem is goed (slecht) geconditioneerd als het conditiegetal klein (groot) is. Merk dat het conditiegetal afhangt van de beschouwde x. Het probleem is goed geconditioneerd voor $\Omega \subseteq X$, als het goed geconditioneerd is voor elke $x \in \Omega$.

Voorbeeld B.1 *We beschouwen de matrix-vector vermenigvuldiging, $y = Ax$, met $A \in \mathbb{C}^{m \times n}$, waarbij we x als het gegeven beschouwen en y als het resultaat. Als we werken met de 2-norm in $\mathbb{C}^n$ en $\mathbb{C}^m$ krijgen we*

$$K = \lim_{\delta \to 0+} \sup_{\|\delta x\| \leq \delta} \frac{\|A(x + \delta x) - Ax\|_2}{\|\delta x\|_2} \frac{\|x\|_2}{\|Ax\|_2} = \frac{\|A\|_2 \|x\|_2}{\|Ax\|_2}. \tag{B.1}$$

[1] Gegevens en resultaat kunnen tot een andere ruimte behoren.

Als A van volle kolomrang is, kunnen we door gebruik te maken van de eigenschap $x = A^+Ax$, met A^+ de pseudo-inverse (A.2), een bovengrens afleiden:

$$K = \frac{\|A\|_2 \, \|A^+(Ax)\|_2}{\|Ax\|_2} \leq \frac{\|A\|_2 \, \|A^+\|_2 \, \|Ax\|_2}{\|Ax\|_2} = \kappa(A),$$

met

$$\kappa(A) := \|A\|_2 \left\|A^+\right\|_2 = \frac{\sigma_1}{\sigma_n}$$

het conditiegetal *van matrix A. Voor $m = n$ is $\kappa(A) = \|A\|_2 \|A^{-1}\|_2$.*

Voorbeeld B.2 *We beschouwen het oplossen van het stelsel $Ax = B$ met $A \in \mathbb{C}^{m\times m}$ en $B \in \mathbb{C}^{m\times 1}$, waarbij we A en B als de gegevens beschouwen en x als het resultaat. In de cursus Numerieke Wiskunde werd afgeleid dat*

$$\frac{\|\delta x\|_2}{\|x\|_2} \leq \kappa(A)\frac{\|\delta A\|_2}{\|A\|_2} + \kappa(A)\frac{\|\delta B\|_2}{\|B\|_2}.$$

Het conditiegetal van matrix A is dus een bovengrens voor het conditiegetal van het probleem van het oplossen van het stelsel voor perturbaties op A en B afzonderlijk, terwijl bovenstaande ongelijkheid ook gecombineerde effecten in rekening brengt.

Het conditiegetal van een matrix is steeds groter of gelijk aan 1. De eigenschap

$$U^*U = I \;\Rightarrow\; \kappa(U) = 1$$

onderstreept het belang van het werken met *unitaire transformaties* in numerieke algoritmen.

B.2 Stabiliteit

We beschouwen opnieuw het abstracte probleem waarbij gegeven $x \in X$ wordt afgebeeld op resultaat $f(x) \in Y$, met f een achterliggende functie. Om $f(x)$ te berekenen gebruiken we een algoritme met invoer $x \in X$. Omwille van de gemaakte fouten modelleren we het algoritme als een andere functie $\tilde{f}$, die invoer x afbeeldt op uitvoer $\tilde{f}(x)$.
We veronderstellen dat er gerekend wordt in eindige precisie, met een bewegende-kommavoorstelling gekarakteriseerd door machineprecisie ϵ_{mach} en verzameling van voorstelbare reële getallen O_{real}.
De nauwkeurigheid van het algoritme kunnen we meten aan de hand van de relatieve fout op het resultaat, de voorwaartse fout genoemd, wat ons brengt tot de definitie van voorwaartse stabiliteit.

DEFINITIE B.2.1 *(voorwaartse stabiliteit)*
Algoritme $\tilde{f}$ voor probleem f is (voorwaarts) stabiel[2] *als voor alle $x \in X$ geldt dat*

$$\frac{\left\|\tilde{f}(x) - f(x)\right\|}{\|f(x)\|} = \mathcal{O}(\epsilon_{\text{mach}}). \tag{B.2}$$

[2] In de terminologie van het boek [TB97] komt dit overeen met de notie van nauwkeurigheid.

Hoewel een kleine relatieve fout op het resultaat in eerste instantie beoogd wordt, zijn er maar weinig algoritmen die voldoen aan (B.2). De reden is dat de moeilijkheid van het probleem, waarvan de conditie een indicator is, niet in rekening gebracht wordt. Vaak is het bijvoorbeeld zo dat afrondingsfouten in de eerste stappen van een algoritme zich gedragen als fouten op de gegevens, en is het probleem slecht geconditioneerd, dan kan je moeilijk een kleine fout op het berekende resultaat verwachten.
Een mogelijke manier om de fout te wegen ten opzichte van de conditie van het probleem bestaat uit het beschouwen van de zogenaamde achterwaartse fout (zie de cursus Numerieke Wiskunde), wat ons leidt tot de notie van achterwaartse stabiliteit.

DEFINITIE B.2.2 *(achterwaartse stabiliteit)*
Algoritme $\tilde{f}$ voor probleem f is achterwaarts stabiel is als er voor elke $x \in X$ een element $\tilde{x} \in X$ bestaat zodat

$$\tilde{f}(x) = f(\tilde{x}), \text{ en } \frac{\|\tilde{x} - x\|_2}{\|\tilde{x}\|} = \mathcal{O}(\epsilon_{\text{mach}}). \tag{B.3}$$

Hier wordt $\tilde{x} - x$ de achterwaartse fout genoemd. Vereiste (B.3) drukt uit dat het berekende resultaat voor invoer x beschouwd kan worden als het *exacte resultaat* voor een wijziging $\tilde{x}$ van de invoer, met relatieve afwijking van grootte-orde de machineprecisie. Bijgevolg voldoet de voorwaartse fout aan

$$\frac{\|\tilde{f}(x) - f(x)\|}{\|f(x)\|} = \mathcal{O}\left(K(x)\epsilon_{\text{mach}}\right),$$

met K het conditiegetal van f, waarbij we de afhankelijkheid van x door de notatie benadrukken. Definitie B.2.2 is dus meer (minder) tolerant naar voorwaartse fouten toe als de conditie van het probleem slechter (beter) is.
We merken tenslotte nog op dat, voor gegeven x, een element $\tilde{x} \in X$ zodat $\tilde{f}(x) = f(\tilde{x})$ niet altijd bestaat en, als het bestaat, het niet altijd uniek is.

We geven nu twee voorbeelden van achterwaarts stabiele algoritmen, met een bewijs. Daarbij gaan we er van uit dat elementaire bewerkingen met elementen uit O_{real} exact gebeuren, waarna er een afronding tot het dichtstbijzijnde getal in O_{real} plaats vindt. We gaan er ook van uit dat de invoer op de machine exact voorgesteld is.

Voorbeeld B.3 *Het algoritme dat het inwendig product van twee gegeven vectoren $x, y \in \mathbb{R}^m$ term per term van links naar rechts uitrekent, is achterwaarts stabiel. Noem $z = x^T y$. Wat berekend wordt is*

$$\tilde{z} = [\cdots[(x_1y_1)(1+\eta_1) + (x_2y_2)(1+\eta_2)](1+\epsilon_2) + \cdots + (x_my_m)(1+\eta_m)](1+\epsilon_m),$$

waarbij de gemaakte relatieve fouten bij de elementaire bewerkingen voldoen aan

$$|\eta_i| \leq \epsilon_{\text{mach}},\ i = 1, \ldots, m, \quad |\epsilon_j| \leq \epsilon_{\text{mach}},\ j = 2, \ldots, m.$$

Na het uitwerken van de producten en verwaarlozen van hogere-orde termen in ϵ_j en η_i verkrijgen we

$$\begin{aligned}\tilde{z} &= x_1y_1(1+\eta_1+\epsilon_2+\cdots+\epsilon_m)+x_2y_2(1+\eta_2+\epsilon_2+\cdots+\epsilon_m)\\ &\quad +\cdots+x_my_m(1+\eta_m+\epsilon_m).\end{aligned}$$

Dit resultaat kan je bijvoorbeeld interpreteren als het exacte inwendig product van vector x met vector $\tilde{y}$, waarbij $\tilde{y}_1 = y_1(1+\eta_1+\epsilon_2+\cdots+\epsilon_m)$ enz.

Voorbeeld B.4 *Een algoritme, gebaseerd op achterwaartse substitutie, voor het oplossen van het driehoekige stelsel $Rx = b$, voluit*

$$\begin{bmatrix} r_{11} & r_{12} & \cdots & r_{1m} \\ & r_{22} & \cdots & r_{2m} \\ & & \ddots & \vdots \\ & & & r_{mm} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_m \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_m \end{bmatrix},$$

is achterwaarts stabiel. Hierbij vormen $R \in \mathbb{R}^{m\times m}$ en $b \in \mathbb{R}^m$ het gegeven, x het resultaat. In de eerste stap bekomen we

$$\tilde{x}_m = \frac{b_m}{r_{mm}}(1+\eta_1),$$

met $|\eta_1| \leq \epsilon_{\text{mach}}$. We weten dat $1+\eta_1 = \frac{1}{1-\eta_1} + \mathcal{O}(\eta_1^2)$. Vullen we dit in en verwaarlozen we de hogere-orde termen, dan verkrijgen we

$$\tilde{r}_{mm}\tilde{x}_m = b_m, \tag{B.4}$$

met $\tilde{r}_{mm} = r_{mm}(1-\eta_1)$ en $\frac{|\tilde{r}_{mm}-r_{mm}|}{|r_{mm}|} \leq \epsilon_{\text{mach}}$.
In de tweede stap berekenen we

$$\tilde{x}_{m-1} = \frac{(b_{m-1} - r_{m-1\,m}\tilde{x}_m(1+\eta_2))\,(1+\epsilon_1)}{r_{m-1\,m-1}}(1+\eta_3),$$

waarbij $|\eta_2| \leq \epsilon_{\text{mach}}$, $|\eta_3| \leq \epsilon_{\text{mach}}$ en $|\epsilon_1| \leq \epsilon_{\text{mach}}$. Expanderen we $1+\epsilon_1 = \frac{1}{1-\epsilon_1} + \mathcal{O}(\epsilon_1^2)$ en $1+\eta_3 = \frac{1}{1-\eta_3} + \mathcal{O}(\eta_3)$ en verwaarlozen we opnieuw de hogere-orde termen, levert ons dat

$$\tilde{x}_{m-1} = \frac{(b_{m-1} - r_{m-1\,m}(1+\eta_2)\tilde{x}_m)}{r_{m-1\,m-1}(1-\epsilon_1-\eta_3)}$$

op. Deze uitdrukking kunnen we ook schrijven als

$$\tilde{r}_{m-1\,m-1}\,\tilde{x}_{m-1} + \tilde{r}_{m-1\,m}\,\tilde{x}_m = b_{m-1}, \tag{B.5}$$

met $\tilde{r}_{m-1\,m-1} = r_{m-1\,m-1}(1-\epsilon_1-\eta_3)$ en $\tilde{r}_{m-1\,m} = r_{m-1\,m}(1+\eta_2)$, zodat

$$\frac{|\tilde{r}_{m-1\,m-1} - r_{m-1\,m-1}|}{|r_{m-1\,m-1}|} \leq 2\epsilon_{\text{mach}}, \quad \frac{|\tilde{r}_{m-1\,m} - r_{m-1\,m}|}{|r_{m-1\,m}|} \leq \epsilon_{\text{mach}}.$$

Dit proces kunnen we verder zetten. Gebruik makend van uitdrukkingen als (B.4) en (B.5) vinden we uiteindelijk dat de berekende oplossing $\tilde{x} = [\tilde{x}_1 \ \cdots \ \tilde{x}_m]^T$ *voldoet aan*

$$(R + \delta R)\tilde{x} = b,$$

met

$$\begin{bmatrix} |\delta r_{11}| & |\delta r_{12}| & |\delta r_{13}| & \cdots & |\delta r_{1m}| \\ & |\delta r_{22}| & |\delta r_{23}| & \cdots & |\delta r_{2m}| \\ & & \ddots & & \vdots \\ & & & |\delta r_{m-1\,m-1}| & |\delta r_{m-1\,m}| \\ & & & & |\delta r_{mm}| \end{bmatrix} \overset{\text{ew}}{\leq} \begin{bmatrix} m & 1 & 2 & \cdots & m-1 \\ & m-1 & 1 & \cdots & m-2 \\ & & \ddots & & \vdots \\ & & & 2 & 1 \\ & & & & 1 \end{bmatrix} \epsilon_{\text{mach}},$$

waarbij $\overset{\text{ew}}{\leq}$ *op een element-gewijze ongelijkheid duidt. Dus* $\tilde{x}$ *kan aanzien worden als exacte oplossing voor invoer* $(R + \delta R, b)$*.*

Opmerking B.1 *De beschouwde algoritmen voor het inwendig product en voor het oplossen van een driehoekig stelsel zijn ook achterwaarts stabiel wanneer met complexe getallen gerekend wordt. De analyse is echter complexer omdat bewerkingen met complexe getallen omgezet worden tot bewerkingen met reële getallen.*

Bibliografie

[Dau92] I. Daubechies. *Ten Lectures on Wavelets*. Society for Industrial and Applied Mathematics, 1992.

[dB78] C. de Boor. *A practical guide to splines*. Springer, New York, 1978.

[Die95] P. Dierckx. *Curve and Surface Fitting with Splines*. Clarendon Press, Oxford, 1995.

[EH10] E. Estrada and D. Higham. Network properties revealed through matrix functions. *SIAM Review*, 52(4):696–714, 2010.

[Fie73] M. Fiedler. Algebraic connectivity of graphs. *Czechoslovak Mathematical Journal*, 23:298–305, 1973.

[LM05] A.N. Langville and C.D. Meyer. A survey of eigenvector methods for web information retrieval. *SIAM Review*, 47(1):135–161, 2005.

[NS82] A. Naylor and G. Sell. *Linear Operator Theory in Engineering and Science*. Springer-Verlag, Berlin, 1982.

[NW06] J. Nocedal and S.J. Wright. *Numerical Optimization*. Springer, 2006.

[PSL90] A. Pothen, H. Simon, and K.-P. Liou. Partitioning sparse matrices with eigenvectors of graphs. *SIAM Journal on Matrix Analysis and its Applications*, 11:430–452, 1990.

[TB97] L.N. Trefethen and D. Bau. *Numerical Linear Algebra*. SIAM, Philadelphia, 1997.

[Tre19] L.N. Trefethen. *Approximation theory and approximation practice, Extended edition*. SIAM, 2019.

[Wat08] D.S. Watkins. The QR algorithm revisited. *SIAM Review*, 50(1):133–145, 2008.

[Wat11] D.S. Watkins. Francis's algorithm. *The American Mathematical Monthly*, 118(5):387–403, 2011.