Chapter 1

Foutenanalyse

1.1 Foutmeting

Stel dat x de exacte waarde is, en $\bar{x}$ de benadering van van de exacte waarde, dan is de:

- Absolute fout: $\Delta x = \bar{x} - x$
- Relatieve fout: $\delta x = \frac{\bar{x}-x}{x}$

De absolute fout is echter betekenisloos zonder context, en er wordt vaker naar de relatieve fout gekeken.

1.2 Klassieke voorstelling van getallen

Elk getal x kan voorgesteld worden als:

$$x \;=\; \pm\,(c_n \;\; c_{n-1} \ldots c_0 . c_{-1} \;\; c_{-2} \;\; c_{-3} \ldots)_b$$

Als $c_n = c_{n-1} = \cdots = c_{k+1} = 0$ en $c_k \neq 0$, dan noemt men de cijfers c_n, c_{n-1}, $\ldots$, c_{k+1} leidende nullen. Het cijfer c_k wordt het eerste beduidende cijfer genoemd. Alle volgende cijfers c_{k-1}, c_{k-2}, $\ldots$ zijn ook beduidende cijfers.

Wanneer c_k het eerste beduidende cijfer is, dan geldt:

$$b^k \;\leqslant |\, x \,| < \; b^{k+1} \tag{1.1}$$

Dus de index l van het eerste beduidende cijfer bevat informatie over de grootte van een getal x.

1.3 Bewegende kommavoorstelling van een getal

Elk getal x kan voorgesteld worden als:

$$x \;=\; \pm \; y \;.\; b^e$$

met y de mantisse
b de basis
e de exponent
b^e de schaalfactor

Om de voorstelling uniek te maken, wordt de schaalfactor b^e zo gekozen dat $y = (c_0 \;.\; c_{-1}\; c_{-2}\; c_{-3})_b$ en $c_0 \neq 0$ (als $x \neq 0$). Dit is de *genormaliseerde bewegende kommavoorstelling.*

1.3.1 Juiste cijfers

We noemen het cijfer $\bar{c}_i$ van $\bar{x}$ een **juist cijfer** t.o.v. de exacte waarde x als:

$$\mid x - \bar{x} \mid \;=\; \mid \Delta x \mid \;\leqslant\; \frac{1}{2} b^i$$

Eigenschappen

- Als $\bar{c}_i$ een juist cijfer is van de benadering $\bar{x}$, dan zijn $\bar{c}_{i+1}$, $\bar{c}_{i+2}$, ... ook juiste cijfers.
- Als $\bar{c}_i$ een juist cijfer is, dan geldt:
  - $\bar{x} \in [x - \frac{1}{2}b^i, x + \frac{1}{2}b^i]$
  - $x \in [\bar{x} - \frac{1}{2}b^i, \bar{x} + \frac{1}{2}b^i]$
- Als $\bar{c}_{j+1}$ het laatste juiste cijfer is, dan is $\bar{c}_j$ geen juist cijfer meer en geldt: $\frac{1}{2}b^j \;< \mid \bar{x} - x \mid \;\leqslant\; \frac{1}{2}b^{j+1}$
- Als $\bar{c}_{\bar{n}}$ het eerste cijfer is van $\bar{x}$ en $\bar{c}_{j+1}$ het laatste cijfer, dan is het *aantal juiste cijfers* gelijk aan $\max(0, \bar{n} - j)$.

Afronden en juiste cijfers

TODO

1.3.2 Het verband tussen de relatieve fout en het aantal juiste beduidende cijfers

Neem een benaderend getal $\bar{x}$:

$$\bar{x} \;=\; \pm(\bar{c}_k \;\; \bar{c}_{k-1} \;\; \ldots \;\; \bar{c}_{j+1} \;\; \bar{c}_j \;\; \ldots)_b$$

met $\bar{c}_k$ het eerste beduidende cijfer
$\bar{c}_{j+1}$ het laatste juiste cijfer

Het *aantal juiste cijfers* is dan gelijk aan $q = k - j$. Omdat $\bar{c}_k$ het eerste beduidende cijfer is, geldt:

$$b^k \;\leqslant |\, \bar{x} \,| \;<\; b^{k+1} \;\Leftrightarrow\; b^{-k-1} \;<\; \frac{1}{|\, \bar{x} \,|} \;\leqslant\; b^{-k} \tag{1.2}$$

Omdat $\bar{c}_{j+1}$ het laatste juiste cijfer is, geldt:

$$\frac{1}{2}b^j \;< |\, \bar{x} - x \,| \leqslant\; \frac{1}{2}b^{j+1} \tag{1.3}$$

Neem deze twee uitdrukkingen samen:

$$\frac{1}{2}b^{-k+j-1} \;<\; \frac{|\, \bar{x} - x \,|}{\bar{x}} \;\leqslant\; \frac{1}{2}b^{-k+j+1} \tag{1.4}$$

$$\Leftrightarrow\; q - 1 \;\leqslant\; \log_b \frac{1}{2}\left|\frac{\bar{x}}{\bar{x} - x}\right| \;<\; q + 1 \tag{1.5}$$

$$\Leftrightarrow\; q \;\approx\; \log_b\left(\frac{1}{2}\left|\frac{\bar{x}}{\bar{x} - x}\right|\right) \tag{1.6}$$

Als $\bar{x} \approx x$, dan wordt de uitdrukking herleid tot $q \approx \log_b\left(\frac{1}{2|\delta x|}\right)$.

1.3.3 Voorstelling van getallen in computersystemen

De functie fl

Een getal $x \in \mathbb{R}$ kan enkel benaderd voorgesteld worden door een computersysteem, met p beduidende cijfers verschillend van 0 in de mantisse. We nemen aan dat afronden gebruikt wordt:

$$fl: \;\; \mathbb{R} \to \mathcal{F}: \;\; x \mapsto fl(x)$$

Deze functie geeft als resultaat de bewegende kommavoorstelling van x, afgerond op p beduidende cijfers. D.w.z. dat $fl(x)$ minstens p juiste beduidende cijfers heeft. Omdat voor het aantal juiste beduidende cijfers q geldt dat

$$\frac{1}{2}b^{-q-1} \;<\; \left|\frac{fl(x)-x}{fl(x)}\right| \;\leqslant\; \frac{1}{2}b^{-q+1}$$

Hieruit volgt dat:

$$\left|\frac{fl(x)-x}{fl(x)}\right| \;\leqslant\; \frac{1}{2}b^{-q+1} \;\leqslant\; \boxed{\frac{1}{2}b^{-p+1} \;=\; \varepsilon_{mach}}$$

Dit kan men schrijven als:

$$x \;=\; fl(x)(1+\varepsilon) \quad \text{met } |\varepsilon| \;\leqslant\; \varepsilon_{mach}$$

Omdat er geldt dat $(1+\varepsilon)^{-1} = 1-\varepsilon$, kan men $fl(x)$ schrijven als:

$$fl(x) \;=\; x\frac{1}{1+\varepsilon} \;\approx\; x(1-\varepsilon)$$

Aangezien men naar de relatieve absolute fout kijkt, kan men de term $(1-\varepsilon)$ ook schrijven als $(1+\varepsilon')$ met $\varepsilon' = -\varepsilon$:

$$\boxed{fl(x) \;=\; x(1+\varepsilon') \quad \text{met } |\varepsilon'| \;\leqslant\; \varepsilon_{mach}}$$

Deze formule vormt de basis van de foutenanalyse voor een algoritme uitgevoerd op een computersysteem.

- Als $|x| > M, fl(x) = \pm\infty$ waarbij het teken bepaald wordt door het teken van x. Er wordt een *overflow flag* gezet.
- Als $0 < |x| < m, fl(x) = \pm 0$ waarbij het teken bepaald wordt door het teken van x. Er wordt een *underflow flag* gezet.
- Er geldt: $fl(0) = 0$.

1.3.4 Elementaire bewerkingen op een computersysteem

We nemen aan dat $x, y \in \mathcal{F}$, dan geldt:

$$x \oplus y \;=\; fl(x+y) \;=\; (x+y)(1+\varepsilon_+)$$

$$x \ominus y \;=\; fl(x-y) \;=\; (x-y)(1+\varepsilon_-)$$

$$x \otimes y \;=\; fl(x \times y) \;=\; (x \times y)(1+\varepsilon_\times)$$
$$x \oslash y \;=\; fl(x \div y) \;=\; (x \div y)(1+\varepsilon_\div)$$

Dit wilt dus zeggen dat elke elementaire bewerking een fout ε introduceert die kleiner is dan (of gelijk aan) de machineprecisie ε_{mach}. Als bijvoorbeeld het uitrekenen van $\sin(x)$ zo geïmplementeerd is dat men het kan opvatten als een primitieve functie, dan geldt:

$$\boxed{\sin}(x) \;=\; fl(\sin(x)) \;=\; \sin(x)(1+\varepsilon_{\sin})$$

Merk op dat hier geldt: $|\varepsilon_+|$, $|\varepsilon_-|$, $|\varepsilon_\times|$, $|\varepsilon_\div|$, $|\varepsilon_{\sin}| \leqslant \varepsilon_{mach}$

Foutenanalyse van een sommatie-algoritme

Neem een algoritme die de volgende waarde berekent en teruggeeft: $s = \sum_{i=1}^{n} a_i$ voor n getallen, met $a_i \in \mathcal{F}$. Het is belangrijk om te weten hoe het algoritme is geïmplementeerd. Neem voor dit voorbeeld het volgende: TODO ALGORITME De berekende waarde $\bar{s}$ is dan:

$$\begin{aligned}
\bar{s} &= fl(\ldots fl(fl(a_1+a_2)+a_3)\cdots+a_n) \\
&= (\ldots((a_1+a_2)(1+\varepsilon_2)+a_3)(1+\varepsilon_3)+\cdots+a_n)(1+\varepsilon_n) \\
&= (a_1+a_2)(1+\varepsilon_2)(1+\varepsilon_3)+\cdots+(1+\varepsilon_n) \\
&\quad + \qquad\qquad a_3(1+\varepsilon_3)+\cdots+(1+\varepsilon_n) \\
&\quad + \;\ldots\; + \qquad\qquad\qquad a_n(1+\varepsilon_n) \\
&\approx (a_1+a_2)(1+\varepsilon_2+\varepsilon_3+\cdots+\varepsilon_n) \\
&\quad + \quad a_3\,(1+\quad \varepsilon_3+\cdots+\varepsilon_n) \\
&\quad + \;\ldots\; + \qquad\quad a_n(1+\varepsilon_n)
\end{aligned}$$

Waarbij er in de laatste stap de hogere orde termen (bv. ε_i^2, $\varepsilon_i\varepsilon_j$, $\ldots$) worden weggelaten, aangezien de fout ε_i op een machine met een precisie van 16 cijfers van de grootteorde 16 of kleiner is, en deze hogere orde termen dus van grootteorde 32 of kleiner (verwaarloosbaar). Hieruit volgt:

$$\begin{aligned}
\bar{s}-s \;&\approx \varepsilon_2(a_1+a_2) \\
&\quad + \;\varepsilon_3(a_1+a_2+a_3) \\
&\quad + \;\ldots\varepsilon_n(a_1+a_2+a_3+\cdots+a_n)
\end{aligned}$$

Aangezien voor alle ε_i geldt dat $\varepsilon_i \leqslant \varepsilon_{mach}$, kunnen we de volgende ongelijkheid beschouwen:

$$|\bar{s}-s| \;\leqslant\; \Big(|a_1+a_2|+|a_1+a_2+a_3|+\cdots+|a_1+a_2+a_3+\ldots a_n|\Big)\varepsilon_{mach}$$

Als alle getallen a_i positief zijn:

$$|\bar{s} - s| \leqslant \Big((n-1)(a_1 + a_2) + (n-2)a_3 + (n-3)a_4 + \cdots + 1.a_n\Big)\varepsilon_{mach}$$

Hieruit besluiten we dat de bovengrens voor de absolute fout klein gemaakt kan worden door de getallen te sommeren van klein naar groot.

1.3.5 Verschillende soorten fouten

- **Afrondingsfouten**: elk getal wordt afgerond naar het dichtstbijzijnd computergetal door de functie $fl(x) = x(1 + \varepsilon)$ met $|\varepsilon| \leqslant \varepsilon_{mach}$
- **Inherente of geïnduceerde fouten**: gegevens die met slechts een eindige precisie gekend zijn (conditie van het numeriek probleem).
- **Afbrekings- of discretisatiefouten**: een computer kan slechts eindige berekeningen doen. Sommige wiskundige bewerkingen zoals bijvoorbeeld de integraal moeten dan benaderd worden a.d.h.v. een eindige som.

  $$\int_a^b f(x)dx = \sum_{k=0}^{N} f(x_k).w_k + I$$

  Hierbij is x_k de gediscretiseerde waarde van x.

Inherente fouten bij optelling

Inherente of geïnduceerde fouten op exacte gegevens zorgen uiteraard voor fouten op resultaten. Neem twee exacte gegevens x en y, en als gewenst resultaat de som van de twee getallen, $s = x + y$. Indien er op die getallen een fout staat, worden ze voorgesteld met een streep erboven:

$$\bar{x} = x + \Delta x \qquad\qquad \bar{y} = y + \Delta y$$

Dan geldt voor de som:

$$\begin{aligned} \bar{s} &= (x + \Delta x) + (y + \Delta y) \\ &= s + (\Delta x + \Delta y) \end{aligned}$$

De *absolute fout op de som* is dan:

$$\Delta s = \bar{s} - s = \Delta x + \Delta y \tag{1.7}$$

De *relatieve fout op de som* wordt gegeven door:

$$\delta s \;=\; \frac{\Delta s}{s} \;=\; \frac{\Delta x + \Delta y}{x + y} \tag{1.8}$$

Merk op dat $|\delta s|$ zeer groot kan worden als $s = x + y \approx 0$ oftewel $x \approx y$. Dit noemt men *catastrophic cancellation*. Dit wilt echter niet zeggen dat de fout in absolute waarde zeer groot word. Neem als voorbeeld een functie $f(x) = x$ en een functie die op elke waarde x een constante fout 10^{-8} toevoegt, dus $\bar{f}(x) = \bar{x} = f(x) + 10^{-8}$. Hoewel de absolute fout (gegeven door $\bar{x} - x = x + 10^{-8} - x = 10^{-8}$) constant is, dus de berekende waarde in zowel 0 als 1 als elk ander punt even veel verschilt van de echte waarde, is de relatieve fout niet overal gelijk (gegeven door $\delta x = \frac{10^{-8}}{x}$). Dit wordt geplot in 1.1.

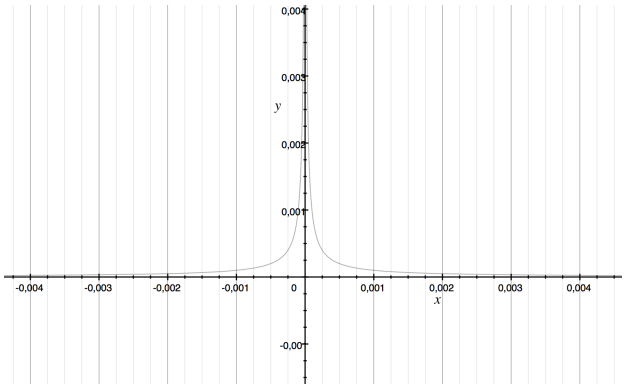


Figure 1.1: De relatieve fout gegeven door $\delta x = 10^{-8}/x$

Inherente fouten bij vermenigvuldiging

Neem nu als resultaat de vermenigvuldiging van twee getallen, $p = x.y$. De relatieve fouten op x en y zijn:

$$\bar{x} \;=\; x(1 + \delta x) \qquad\qquad \bar{y} \;=\; y(1 + \delta y)$$

Het resultaat van de gegevens $\bar{x}$ en $\bar{y}$ is:

$$\begin{aligned} \bar{p} \;&=\; \bar{x}.\bar{y} \\ &=\; x(1 + \delta x).y(1 + \delta y) \\ &=\; xy(1 + \delta x + \delta y + \delta x.\delta y) \\ &\approx\; p(1 + \delta p) \end{aligned}$$

met $\delta p \approx \delta x + \delta y$. TODO Foutvoortplanting

1.4 Conditie van een numeriek probleem

Als een wiskundig verband tussen de gegevens g en het resultaat r exact is (d.w.z. dat het in oneindige precisie uitgerekend kan worden), wordt dit genoteerd als $r = F(g)$. De *conditie* van een numeriek probleem zegt iets over hoe kleine veranderingen of perturbaties Δx op een gegeven x het resultaat beïnvloeden. Beschouw alle mogelijke perturbaties kleiner dan ε op een gegeven g (de grijze cirkel in 1.2), en alle resultaten van de verzameling van perturbaties op g.

De *absolute conditie* zegt nu wat de grootst mogelijke absolute fout op het

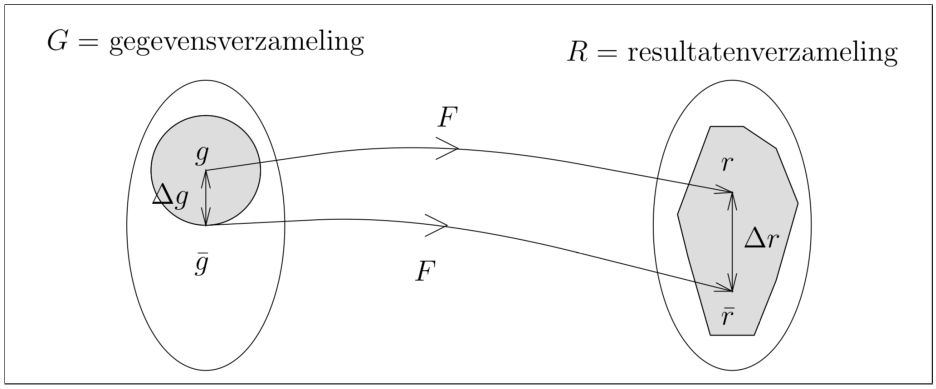


Figure 1.2: De conditie van een numeriek probleem

resultaat is, $||\Delta r||$ indien men de grootst mogelijke perturbatie beschouwt op het gegeven, $||\Delta g||$. Dit wordt uitgedrukt als:

$$\sup_{\Delta||g||\leqslant\varepsilon} \frac{||\Delta r||}{||\Delta g||}$$

Het *absoluut conditiegetal* geeft dan de verhouding tussen de fout op het resultaat en de fout op de gegevens indien de perturbatie op de gegevens zeer klein wordt. Aangezien de perturbatie kleiner moet zijn dan ε, kunnen we de limiet van $\varepsilon \to 0$ nemen om de perturbatie infinitesimaal klein te maken. Dus geldt:

$$\kappa_A(F, g) = \lim_{\varepsilon\to 0} \left(\sup_{\Delta||g||\leqslant\varepsilon} \frac{||\Delta r||}{||\Delta g||} \right) \tag{1.9}$$

Het *relatief conditiegetal* wordt gegeven door:

$$\kappa_R(F, g) = \lim_{\varepsilon\to 0} \left(\sup_{\delta||g||\leqslant\varepsilon} \frac{||\delta r||}{||\delta g||} \right) \tag{1.10}$$

Indien $F(g)$ een differentieerbare functie is in één of meer veranderlijken, kunnen de conditiegetallen ook geschreven worden als:

$$\kappa_A(F,g) \;=\; ||F'(g)|| \qquad\qquad \kappa_R(F,g) \;=\; ||F'(g)|| \cdot \frac{||g||}{||r||} \tag{1.11}$$

waarbij $F'(g)$ de Jacobiaan is, gedefinieerd als:

$$J \;=\; \begin{bmatrix} \frac{\partial F}{\partial x_1} & \cdots & \frac{\partial F}{\partial x_n} \end{bmatrix} \;=\; \begin{bmatrix} \frac{\partial F_1}{\partial x_1} & \cdots & \frac{\partial F_1}{\partial x_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial F_m}{\partial x_1} & \cdots & \frac{\partial F_m}{\partial x_n} \end{bmatrix} \qquad \text{met } J_{ij} = \frac{\partial f_i}{\partial x_j}$$

Een conditiegetal zegt dus hoe gevoelig de resultaten zijn voor perturbaties van de gegevens. Een kleine κ zegt dus dat het probleem *goed gecondition-eerd* is, oftewel weinig gevoelig aan veranderingen in de gegevens. Merk op dat het hier gaat over de conditie van een numeriek probleem (de mapping van gegevens naar hun resultaat gegeven door F), en dit niet afhankelijk is van de algoritmische implementatie van het probleem.

Voorbeeld: evalueren van $f(x,y) = x^2 - y^2$
De gegevens zijn $g = (x,y)$, en de resultaten $r = F(g) = f(x,y)$. De perturbaties op de gegevens zijn $\Delta g = (\Delta x, \Delta y)$ met als eenheidsnorm $||\Delta g||_1 = |\Delta x| + |\Delta y|$, en zorgen voor een absolute fout op de resultaten Δr gegeven door:

$$\begin{aligned} \Delta r \;&=\; f(x+\Delta x, y+\Delta y) - f(x,y) \\ &\approx\; f(x,y) + \frac{\partial f(x,y)}{\partial x}.\Delta x + \frac{\partial f(x,y)}{\partial y}.\Delta y - f(x,y) \\ &\approx\; 2x.\Delta x - 2y.\Delta y \\ &=\; \begin{bmatrix} 2x & -2y \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta y \end{bmatrix} \end{aligned}$$

Het absoluut conditiegetal is dan:

$$\kappa_A \;=\; \lim_{\varepsilon \to 0} \left(\sup_{||\Delta g|| \leqslant \varepsilon} \left(\frac{|2x\Delta x - 2y\Delta y|}{|\Delta x| + |\Delta y|} \right) \right)$$

en het relatief conditiegetal:

$$\begin{aligned} \kappa_R \;&\leqslant\; \lim_{\varepsilon \to 0} \left(\sup_{||\Delta g|| \leqslant \varepsilon} \left(\frac{||(2x,-2y)||_1.\cancel{||(\Delta x, \Delta y)||_1}.||(x,y)||_1}{\cancel{||(\Delta x, \Delta y)||_1}|x^2 - y^2|} \right) \right) \\ &\leqslant\; \frac{2(|x|+|y|)^2}{x^2 - y^2} \end{aligned}$$

Dit is dus een slecht geconditioneerd probleem als $x \approx y$: het relatief conditiegetal wordt dan zeer groot.

1.5 Stabiliteit van een algoritme

Bij de conditie van een numeriek probleem was het algoritme dat gebruikt werd voor het probleem niet van belang. Algoritmes zorgen echter voor afrondings- en discretisatiefouten van het resultaat bovenop de geïnduceerde fouten op de gegevens, die niet verwaarloosd mogen worden.

- We vertrekken van de exacte gegevens g die een exact resultaat r geven met behulp van de mapping F in oneindige precisie.
- Die exacte gegevens worden omgezet naar computergetallen. Dit zijn dan inexacte gegevens $\bar{g}$ die in oneindige precisie naar $\bar{r}$ gemapt worden door F.
- Nu moet F ook omgezet worden naar een numeriek algoritme. De eerste stap is het discretiseren van het numeriek probleem (bv. een integraal met eindige sommen benaderen). De mapping $\tilde{F}$ zet de computergetallen om naar $\tilde{r}$ met discretisatiefouten.
- Ten slotte moet het algoritme $\tilde{F}$ nog in eindige precisie geïmplementeerd worden, wat voor afrondingsfouten zorgt. Het uiteindelijk algoritme $\bar{\tilde{F}}$ zet $\bar{g}$ om in $\bar{\tilde{r}}$.

Op dit resultaat zit nu een geïnduceerde fout, discretisatiefout en afrondingsfout. *Numerieke stabiliteit* meet de afrondingsfout die gemaakt wordt door het implementeren van $\tilde{F}$ in eindige precisie (waar de conditie een maatstaf was voor de geïnduceerde fout).

1.5.1 Voorwaartse of sterke stabiliteit

Een algoritme is voorwaarts stabiel als het numeriek berekend resultaat weinig verschilt van het echt resultaat. Sterke stabiliteit voor de absolute fout is:

$$||\bar{\tilde{F}}(\bar{g}) - \tilde{F}(\bar{g})||$$

en voor de relatieve fout:

$$\frac{||\bar{\tilde{F}}(\bar{g}) - \tilde{F}(\bar{g})||}{||\tilde{F}(\bar{g})||}$$

Indien deze klein zijn, is het algoritme *voorwaarts stabiel*.

1.5.2 Achterwaartse stabiliteit

Bij achterwaartse stabiliteit kijkt men naar hoe groot de afwijking op de gegevens mag zijn om hetzelfde resultaat te bekomen. Neem twee niet-exacte gegevens $\bar{g}$ en $\bar{\bar{g}}$ waarvoor geldt dat $\bar{F}(\bar{g}) = \tilde{F}(\bar{\bar{g}}) = \bar{\tilde{r}}$. Als er geldt dat:

$$\frac{||\bar{\bar{g}} - \bar{g}||}{||\bar{g}||} \approx \varepsilon_{mach}$$

dan wilt dit zeggen dat het algoritme *achterwaarts stabiel* is, en dat de (eventueel) grote afwijkingen op het resultaat kan liggen aan een slecht geconditioneerd probleem. Indien er geldt dat:

$$\frac{||\bar{\bar{g}} - \bar{g}||}{||\bar{g}||} \gg \varepsilon_{mach}$$

dan is het algoritme niet achterwaarts stabiel is.

1.5.3 Zwakke stabiliteit

Indien een algoritme zwak stabiel is, geldt:

$$\frac{||\bar{\bar{r}} - \tilde{r}||}{||\bar{\bar{r}} - \tilde{r}||} \approx 1$$

Indien een algoritme niet zwak stabiel is, geldt:

$$\frac{||\bar{\bar{r}} - \tilde{r}||}{||\bar{\bar{r}} - \tilde{r}||} \gg 1$$

Voorbeeld: evalueer $f(x) = \frac{1-\cos(x)}{x^2}$ voor $x \approx 0$.
Neem aan dat er geen discretisatiefouten aanwezig zijn ($\tilde{f}(x) = f(x)$ en $\tilde{x} = x$). We voeren eerst de conditieonderzoek uit:

$$\kappa_R = \left|\frac{f'(x).x}{f(x)}\right| = \left|\frac{2 - \sin(x).x - 2\cos(x)}{\cos(x) - 1}\right|$$

Dit berekend voor de waarde $x \to 0$:

$$\lim_{x \to 0} \kappa_R(x) = 0$$

Deze is dus zeer klein, en het probleem is goed geconditioneerd voor $x \approx 0$. Als we nu de stabiliteitsonderzoek uitvoeren, beginnend bij zwakke stabiliteit:

$$\begin{aligned}
\bar{f}(x) &= \frac{(1-\cos(x)(1+\varepsilon_1))(1+\varepsilon_2)}{x^2(1+\varepsilon_3)}(1+\varepsilon_4) \\
&\approx \left(\frac{1-\cos(x)}{x^2}\right) + \varepsilon_1\left(\frac{-\cos(x)}{x^2}\right) + (\varepsilon_2-\varepsilon_3+\varepsilon_4)\left(\frac{1-\cos(x)}{x^2}\right) \\
&\Updownarrow \\
\frac{\bar{f}(x)-f(x)}{f(x)} &\approx \varepsilon_1\left(\frac{-\cos(x)}{x^2}\right)\left(\frac{x^2}{1-\cos(x)}\right) + (\varepsilon_2-\varepsilon_3+\varepsilon_4) \\
&\Updownarrow \\
\left|\frac{\bar{f}(x)-f(x)}{f(x)}\right| &\leqslant \left(\left|\frac{\cos(x)}{1-\cos(x)}\right| + 3\right)\varepsilon_{mach}
\end{aligned}$$

Voor $x \approx 0$ geldt dat $\cos(x) \approx 1$ en dus wordt de noemer $1-\cos(x) \approx 0$. Bij de laatste ongelijkheid wordt de bovengrens dus zeer groot, en is dit numeriek probleem hierdoor niet zwak stabiel.

EINDE HOOFDSTUK: Nog extra notities van de les.

$$\delta f(x) = \frac{\Delta f(x)}{f(x)} \approx \frac{f'(x).\Delta x}{f(x)} \tag{1.12}$$

$$\approx \frac{f'(x).x}{f(x)}.\frac{\Delta x}{x} \tag{1.13}$$

Hier is de term voor de maal de vermenigvuldigingsfactor op de fout van het gegeven (relatieve fout). Er is een norm $||g||_G$ nodig in de gegevensverzameling G (en voor de resultatenverzameling R). Deze norm moet kleiner zijn dan ε, dan krijgen we een verzameling rond een gegeven met alle mogelijke licht gewijzigde gegevens. Dan gaan we van al deze gegevens het overeenkomstig resultaat bekijken (dit is opnieuw een verzameling). Dan beschouwen we in deze begrensde resulatenverzameling de grootst mogelijke afstand tussen twee elementen in die verzameling. Dit is het *conditiegetal*. Als deze groot is (groot is hier ook relatief) ...
De norm van de Jacobiaan moet genomen worden bij een functie in meerdere veranderlijken.

$$(x+y)(1+\varepsilon_+) = x(1+\varepsilon_+) + y(1+\varepsilon_+)$$

Alle bewerkingen zijn dus achterwaarts stabiel in de computer, maar toch kunnen grote fouten voorkomen door de foute conditie.

EXTRA: fout bij $2\frac{sin(\frac{1}{2}x)^2}{x^2}$

$$\frac{(1+\varepsilon_{\sin})(1+\varepsilon_{kwad})}{(1+\varepsilon_2)}(1+\varepsilon_1) \;=\; (1+\varepsilon_{\sin}+\varepsilon_{kwad}+\varepsilon_1-\varepsilon_2)$$

Dus goed geconditioneerd.

Chapter 2

Stelsels lineaire vergelijkingen

Voor stelsels lineaire vergelijkingen van de vorm:

$$\begin{cases} a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n = b_1 \\ a_{12}x_1 + a_{22}x_2 + \cdots + a_{2n}x_n = b_2 \\ \quad \vdots \qquad \vdots \qquad \vdots \qquad \vdots \\ a_{i1}x_1 + a_{i2}x_2 + \cdots + a_{in}x_n = b_i \\ \quad \vdots \qquad \vdots \qquad \vdots \qquad \vdots \\ a_{n1}x_1 + a_{n2}x_2 + \cdots + a_{nn}x_n = b_n \end{cases}$$

waarbij a_{ij} en b_i gegeven zijn voor $i, j = 1, 2, \ldots, n$ en x_j een onbekende, is de complexiteit voor het berekenen de onbekenden van grootte-orde $\mathcal{O}(n^3)$. Deze kan echter verlaagd worden als er gebruik gemaakt wordt van bepaalde eigenschappen van het stelsel.

Chapter 3

Veeltermінterpolatie

Dit hoofdstuk gaat over het zoeken van een functie f dat door de punten punten (x_k, f_k) gaat, met $k = 0, 1, \ldots, n$. Dus er moet gelden dat $f(x_k) = f_k$. Er zijn uiteraard oneindig veel functies, maar als restrictie nemen we dat f een veelterm moet zijn. Deze zijn eenvoudig te evalueren, differentiëren en integreren. De veelterm $p(x)$ heeft $n + 1$ interpolatievoorwaarden (punten waar p door moet gaan), en is van graad n:

$$p(x) \;=\; a_0 + a_1 x + a_2 x^2 \cdots + a_n x^n \tag{3.1}$$

De coëfficiënten a_i moeten zo bepaald worden dat de veelterm in elke x_i evalueert tot f_i. Dit kan expliciet geschreven worden als een stelsel van $n + 1$ vergelijkingen in $n + 1$ onbekenden:

$$\begin{array}{ccccccccccc}
a_0 & + & a_1 x_0 & + & a_2 x_0^2 & + & \ldots & + & a_n x_0^n & = & f_0 \\
a_0 & + & a_1 x_1 & + & a_2 x_1^2 & + & \ldots & + & a_n x_1^n & = & f_1 \\
 & & & & & & \vdots & & & & \\
a_0 & + & a_1 x_n & + & a_2 x_n^2 & + & \ldots & + & a_n x_n^n & = & f_n
\end{array}$$

In matrixvorm kan men dit schrijven als $V.a = f$:

$$\begin{bmatrix}
1 & x_0 & x_0^2 & \ldots & x_0^n \\
1 & x_1 & x_1^2 & \ldots & x_1^n \\
\vdots & \vdots & \vdots & \ddots & \vdots \\
1 & x_n & x_n^2 & \ldots & x_n^n
\end{bmatrix}
\begin{bmatrix} a_0 \\ a_1 \\ \vdots \\ a_n \end{bmatrix}
\;=\;
\begin{bmatrix} f_0 \\ f_1 \\ \vdots \\ f_n \end{bmatrix}$$

waarbij V een Vandermonde-matrix is. Deze heeft als eigenschap dat

zijn determinant gelijk is aan:

$$\begin{aligned}\det V &= \prod_{\substack{1 \leqslant k \leqslant n \\ j < k}} (x_k - x_j) \\ &= (x_1 - x_0). \\ &\quad (x_2 - x_0)(x_2 - x_1). \\ &\quad (x_3 - x_0)(x_3 - x_1)(x_3 - x_2). \\ &\quad \dots \\ &\quad (x_n - x_0)(x_n - x_1)\dots(x_n - x_{n-1})\end{aligned}$$

Deze is verschillend van nul als $x_i \neq x_j$, met $i \neq j$. De matrix is dan *singulier*. Dit wilt zeggen dat de *interpolerende veelterm bestaat en uniek is*. Het vinden van de veelterm heeft een complexiteit van $O(n^3)$ indien men de methode van Gauss toepast.

Bij het ontwerpen van een interpolerende veelterm ontstaan er echter twee problemen: het coëfficiëntenprobleem, en het waardeprobleem.

3.1 Coëfficiëntenprobleem

Er geldt dat $p(x) \in P_n$, waarbij P_n gedefinieerd is als:

$$P_n = \{p(x) \mid \deg p(x) \leqslant n\}$$

Dit is een vectorruimte van dimensie $n+1$, dus met een basis bestaande uit $n+1$ veeltermen, met elementen van de vorm $p(x) = a_0 + a_1 x + \dots + a_n x^n$. Het *coëfficiëntenprobleem* gaat vooral over het vinden van een veelterm, en niet zozeer de waarde ervan in x. Men kan de vectorruimte van veeltermen van graad n veralgemenen naar een ruimte met basis $\{\phi_i(x)\}_{i=0}^n$. Dan zijn de veeltermen van de vorm:

$$y_n = c_0\phi_0(x) + c_1\phi_1(x) + \dots + c_n\phi_n(x)$$

Het probleem bestaat nu uit het zoeken naar de coëfficiënten $\{c_k\}_{i=0}^n$ ten opzichte van de basis $\{\phi_i\}$. Als men bijvoorbeeld $\phi_k(x) = x^k$ neemt, dan zijn de coëfficiënten $c_k = a_k$ en wordt het probleem herleidt naar 3.1. Een paar andere basissen zijn:

$$\begin{aligned}\phi_k(x) &= (x - x_0)^k \\ \phi_k(x) &= (x - x_0)(x - x_1)\dots(x - x_{k-1}) \\ \phi_k(x) &= (x - x_0)(x - x_1)\dots(x - x_{k-1})(x - x_{k+1})\dots(x - x_n)\end{aligned}$$

3.1.1 Conditie van het coëfficiëntenprobleem

In matrixvorm kan het coëfficiëntenprobleem veralgemeend worden tot:

$$\begin{bmatrix} \phi_0(x_0) & \phi_1(x_0) & \dots & \phi_n(x_0) \\ \phi_0(x_1) & \phi_1(x_1) & \dots & \phi_n(x_1) \\ \vdots & \vdots & \ddots & \vdots \\ \phi_0(x_n) & \phi_1(x_n) & \dots & \phi_n(x_n) \end{bmatrix} \begin{bmatrix} c_0 \\ c_1 \\ \vdots \\ c_n \end{bmatrix} = \begin{bmatrix} f_0 \\ f_1 \\ \vdots \\ f_n \end{bmatrix}$$

Het probleem, alsook de (relatieve) conditie, is dus afhankelijk van het oplossen van het stelsel. Het stelsel is ook afhankelijk van de gekozen basis. Neem als basis de machten van x, dan is de matrix een Vandermonde-matrix. Indien de afstanden tussen de interpolatiepunten x_i klein worden, wordt de determinant klein. Dit kan dan wijzen op een slechte conditie. De conditie is dus $\kappa(V)$.

3.1.2 Stabiliteit van het coëfficiëntenprobleem

Het oplossen van een stelsel brengt geen discretisatiefouten teweeg. Er moet dus enkel rekening gehouden worden met afrondingsfouten.

3.2 Waardeprobleem

Bij het waardeprobleem wilt men, voor de gegevens $(x_k, f_k = f(x_k))$ en een $\bar{x}$, een goede benadering voor $f(\bar{x})$ in de vorm van $y_n(\bar{x})$. Hiervoor gebruikt men een *interpolatieproces*, waarbij de toenemende graad van $y_i(\bar{x})$ naar de uiteindelijke $f(\bar{x})$ zou moeten convergeren (in het ideale geval).

3.2.1 Conditie van het waardeprobleem

De conditie zegt hoe goed dat de waarde $f(\bar{x})$ benaderd wordt. Bij discontinue functies kan de conditie oneindig slecht worden. We beperken ons tot de klasse van veeltermen voor de bespreking van conditie, aangezien deze afhankelijk is van de gebruikte functieklasse.

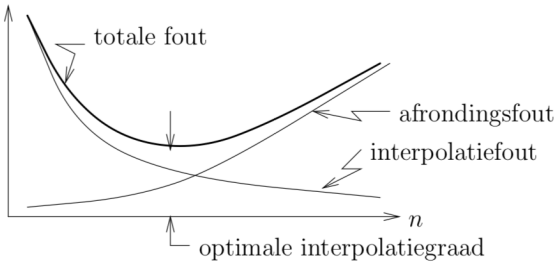
Indien het verschil tussen $\tilde{y}(\bar{x})$ en $y(\bar{x})$ groot is, zegt men dat het probleem slecht geconditioneerd is.

3.2.2 Stabiliteit van het waardeprobleem

De totale absolute fout $|f(\bar{x}) - y_n(\bar{x})|$ wordt veroorzaakt door

- *interpolatiefouten*: deze neemt af naarmate het aantal gebruikte punten (of de graad i van $y_i(x)$) stijgt.
- *afrondingsfouten*: deze neemt toe naarmate de graad stijgt en er meer en meer berekeningen gemaakt moeten worden, en de fouten zich voortplanten.

Hierdoor ontstaat er een optimale interpolatiegraad n waardoor de totale fout minimaal wordt, en $f(\bar{x})$ het best benaderd wordt.



3.3 De interpolatiefout

De interpolatiefout wordt gegeven door:

$$E_n(x) \;=\; \frac{f^{(n+1)}(\xi(x))}{(n+1)!}(x-x_0)(x-x_1)\ldots(x-x_n) \tag{3.2}$$

Indien er enkel sprake is van een interpolatiefout, is het verband tussen f en y_n:

$$f(x) = y_n(x) + E_n(x) \quad \text{met } x \in [a,b]$$

Hier is ξ een getal in het interval $[\min(x_i), \max(x_i)]$, maar omdat deze afhankelijk is van de gekozen waarde x waarin men interpoleert, noteert men dit vaak als $\xi(x)$.

Omdat de formule zo weinig bruikbaar is, wordt deze vaak gebruikt als bovengrens voor de maximale fout:

$$|f(x) - P_n(x)| \;\leqslant\; \frac{\max\left(|f^{(n+1)}(\xi)|\right)}{(n+1)!}\max\left(|(x-x_0)\ldots(x-x_n)|\right) \tag{3.3}$$

3.4 Interpolerende veelterm volgens Lagrange

Bij de Lagrangemethode wordt als basis $\{l_i(x)\}$ genomen, waarbij $\deg l_i(x) = n$, en voor de coëfficiënten $\{f_i\}$. De *interpolerende veelterm volgens Lagrange* is dan:

$$y_n(x) \;=\; l_0(x)f_0 \;+\; l_1(x)f_1 \;+\; \ldots \;+\; l_n(x)f_n \tag{3.4}$$

Om aan de *interpolatie-eis* $y_n(x_i) = f_i$ te voldoen, moet er gelden dat:

$$l_i(x_j) \;=\; \begin{cases} 1 \;\text{ als } i = j \\ 0 \;\text{ als } i \neq j \end{cases}$$

De definitie van $l_i(x_j)$ is dus gelijk aan dat van de Kronecker-delta δ_{ij}. De Kronecker-delta is echter geen 'functie', dus wordt l_i als volgt gedefinieerd:

$$l_i(x) \;=\; c_i \,\frac{(x-x_0)(x-x_1)\ldots(x-x_n)}{(x-x_i)}$$

Uit de voorwaarde $l_i(x_i) = 1$ halen we de waarde van c_i:

$$l_i(x_i) \;=\; 1 \;=\; c_i \,\frac{(x_i-x_0)(x_i-x_1)\ldots(x_i-x_i)\ldots(x_i-x_n)}{(x_i-x_i)}$$

$$\Updownarrow$$

$$c_1 \;=\; \frac{1}{(x_i-x_1)\ldots(x_i-x_{i-1})(x_i-x_{i+1})\ldots(x_i-x_n)}$$

Definieer nu:

$$\begin{aligned} \pi(x) \;&=\; (x-x_1)(x-x_2)\ldots(x-x_n) \\ \pi'(x_i) \;&=\; (x_i-x_1)\ldots(x_i-x_{i-1})(x_i-x_{i+1})\ldots(x_i-x_n) \end{aligned}$$

Hieruit volgt de formule voor de *Lagrange-veelterm*:

$$l_i(x) \;=\; \frac{\pi(x)}{\pi'(x_i)(x-x_i)} \left(\;=\; \prod_{\substack{j=0 \\ j\neq i}}^{n} \frac{(x-x_j)}{(x_i-x_j)} \right) \tag{3.5}$$

Voor het evalueren van de veelterm in een punt x kunnen we het volgend algoritme gebruiken.

Invoer : $x_0, \ldots, x_n$; $f_0, \ldots, f_n$; x
Uitvoer: $y_n(x)$
$y_n(x) \leftarrow 0$;
for $i = 0, 1, \ldots, n$ **do**
  $t \leftarrow f_i$;
  for $j = 0, 1, \ldots, n$ **do**
    if $j \neq i$ **then**
      $t \leftarrow t \cdot \frac{x - x_j}{x_i - x_j}$
    end
  end
  $y_n(x) \leftarrow y_n(x) + t$
end

Het algoritme heeft een complexiteit van $\mathcal{O}(n^2)O + \mathcal{O}(n^2)V$.

3.4.1 Lagrange veeltermen voor equidistante punten

Voor equidistante punten geldt: $x_k = x_0 + k \cdot h$, met h de afstand tussen twee opeenvolgende punten. We voeren een nieuwe veranderlijke u in:

$$u(x) \;=\; \frac{x - x_0}{h}$$

De Lagrange veeltermen worden gegeven door

$$\tilde{l}_i(u) \;=\; l_i(x) \;=\; \frac{u(u-1)(u-2)\ldots(u-i+1)(u-i-1)\ldots(u-n)}{i(i-1)(i-2)\ldots(1)(-1)\ldots(i-n)} \tag{3.6}$$

en de interpolatiefout door

$$E_n(x) \;=\; \frac{u(u-1)\ldots(u-n)}{(n+1)!} h^{n+1} f^{(n+1)}(\xi(x)) \tag{3.7}$$

3.4.2 Nut van de Lagrange veeltermen

Indien de x_i onveranderd blijven, blijven ook de $l_i(x)$ ongewijzigd. Men moet de interpolerende veelterm dus in de praktijk maar één keer evalueren. Het toevoegen van een punt zal er echter voor zorgen dat er een nieuwe term $l_{n+1}(x)f_{n+1}$ toegevoegd moet worden, en alle voorgaande termen herberekend moeten worden. De Lagrangemethode is dus geschikt voor het coëfficiëntenprobleem, maar niet voor het waardeprobleem (interpolatieproces).

3.5 Interpolerende veelterm volgens Newton

Als basis voor P_n nemen we nu

$$\phi_k(x) \;=\; (x - x_0)(x - x_1)\dots(x - x_{k-1}) \tag{3.8}$$

Het coëfficiëntenprobleem $[\phi_j(x_i)][c_j] = [f_i]$ ziet er als volgt uit:

$$\begin{bmatrix} 1 & 0 & 0 & \dots & 0 \\ 1 & (x_1 - x_0) & 0 & \dots & 0 \\ 1 & (x_2 - x_0) & (x_2 - x_0)(x_2 - x_1) & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & (x_n - x_0) & (x_n - x_0)(x_n - x_1) & \dots & (x_n - x_0)(x_n - x_{n-1}) \end{bmatrix} \begin{bmatrix} c_1 \\ c_2 \\ c_3 \\ \vdots \\ c_n \end{bmatrix} = \begin{bmatrix} f_0 \\ f_1 \\ f_2 \\ \vdots \\ f_n \end{bmatrix} \tag{3.9}$$

Merk op dat $[\phi_j(x_i)]$ een onderdriehoeksmatrix is dat met *voorwaartse substitutie* opgelost kan worden in $\mathcal{O}(n^2)$ tijd. TODO

3.6 Gedeelde differenties

Gedeelde differenties worden als volgt gedefinieerd:

0^{de} orde: $$f[x_i] \;=\; f_i$$

1^{e} orde: $$f[x_i, fx_{i+1}] \;=\; \frac{f_{i+1} - f_i}{x_{i+1} - x_i} \quad \left(= \frac{f[x_{i+1}] - f[x_i]}{x_{i+1} - x_i}\right)$$

$(j-i)^{\text{de}}$ orde: $$f[x_i, x_{i+1}, \dots, x_j] \;=\; \frac{f[x_{i+1}, \dots, x_j] - f[x_i, \dots, x_{j-1}]}{x_j - x_i}$$

De coëfficiënten c_i van de Newton-veeltermen zijn dus gebaseerd op de eerste x_i punten, en worden gegeven door:

$$c_i \;=\; f[x_0, x_1, \dots, x_i]$$

Chapter 4

Numerieke Integratie

Neem de functie $f(x)$ waarvan we de integraal numeriek gaan benaderen. Neem de Lagrangevoorstelling van $f(x)$, hieruit gaan we de *kwadratuurformule* 4.1 afleiden.

$$
\begin{aligned}
& f(x) = y_n(x) + E_n(x) \\
\Leftrightarrow \quad & f(x) = \sum_{k=0}^{n} l_k(x) f(x_k) + E_n(x) \\
\Leftrightarrow \quad & \int_a^b f(x)dx = \int_a^b \left(\sum_{k=0}^{n} l_k(x) f(x_k) \right) dx + \int_a^b E_n(x)dx \\
\Leftrightarrow \quad & \int_a^b f(x)dx = \sum_{k=0}^{n} \left(\int_a^b l_k(x)dx \right) f(x_k) + F_n \\
\Leftrightarrow \quad & \int_a^b f(x)dx = \sum_{k=0}^{n} H_k f(x_k) + F_n \qquad (4.1)
\end{aligned}
$$

met H_k : gewichten
x_k : abscissen
F_n : integratiefout

4.1 Conditie

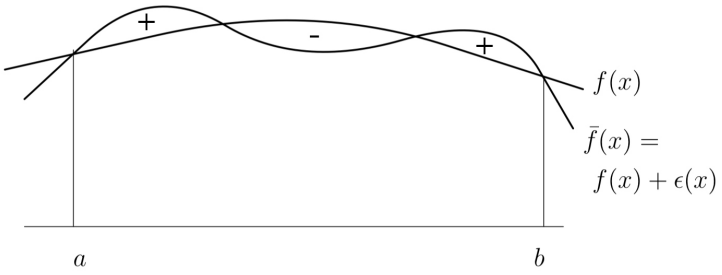
Indien we een kleine fout $\varepsilon(x)$ op de integrand $f(x)$ aanbrengen, geldt er:

$$\int_a^b \left(f(x) + \varepsilon(x) \right) dx = \int_a^b f(x)dx + \int_a^b \varepsilon(x)dx$$

De wijziging op het resultaat is dan ook relatief klein, namelijk:

$$\int_a^b \varepsilon(x)dx$$

Grafisch kan dit als volgt voorgesteld worden. De conditie van het integratieprobleem is dus veel beter dan dat van de differentiatieprobleem.



4.2 Stabiliteit

De stabiliteit hangt af van

- de afbrekings- of discretisatiefout: hier de integratiefout F_n.
- de afrondingsfouten bij het berekenen van $\sum H_i f_i$

TODO

4.3 Integratiefout

De integratiefout werd in de afleiding van 4.1 gegeven door:

$$F_n \;=\; \int_a^b E_n(x)dx \;=\; \int_a^b \frac{f^{(n+1)}(\xi(x))}{(n+1)!}(x-x_0)\ldots(x-x_n)dx$$

Voor $n = 1$, $x_0 = a$ en $x_1 = b$ geldt er:

$$F_1 \;=\; \frac{1}{12}f''(\eta)(b-a)^3 \;, \qquad \eta \in (a,b)$$

Voor *equidistante abscissen* kan men de volgende formules gebruiken:

$$F_n \;=\; \frac{f^{(n+2)}(\eta)}{(n+2)!}\int_a^b x.\pi(x)dx \qquad n \text{ even} \tag{4.2}$$

$$F_n \;=\; \frac{f^{(n+1)}(\eta)}{(n+1)!}\int_a^b \pi(x)dx \qquad n \text{ oneven} \tag{4.3}$$

4.4 Nauwkeurigheidsgraad

Een kwadratuurformule heeft nauwkeurigheidsgraad n als

- alle veeltermen van graad $\leqslant n$ exact geïntegreerd worden.
- er minstens één veelterm van graad $n+1$ bestaat die niet exact geïntegreerd wordt.

Dit begrip heeft enkel zin wanneer de functies een veeltermachtig verloop hebben. De integraal opsplitsen in deelintegralen kan de nauwkeurigheidsgraad doen stijgen.

Bij een even n zegt de term $f^{(n+2)}(\eta)$ in 4.2 dat de kwadratuurformule een nauwkeurigheidsgraad heeft van $n + 1$, oftewel dat elke veelterm van graad $n+1$ exact geïntegreerd wordt, terwijl deze bij de oneven n slecht van graad n is.

Chapter 5

Iteratieve methoden

- x^*: exacte oplossing
- $x^{(0)}$: startwaarde
- $x^{(k+1)}$: de $k+1^e$ benadering afhankelijk van de k vorige benaderingen.
  - $x^{(k+1)} = F_{k+1}(x^{(0)}, x^{(1)}, \ldots, x^{(k)})$
- F_k: de iteratieformule

Het stopcriterium is dat $x^{(k)}$ een voldoende goede benadering is voor x^*. De benaderingen $x^{(0)}, x^{(1)}, \ldots, x^{(k)}, \ldots$ convergeren naar x^*

Chapter 6

Oplossen van niet-lineaire vergelijkingen

Gegeven is een functie $f : D \subset \mathbb{R} \to \mathbb{R}$. We zoeken nu één of meerdere waarden voor x zodat $f(x) = 0$. Er geldt dan voor een $(n+1)$-voudig nulpunt dat:

$$f(x^*) \;=\; \frac{df(x^*)}{dx} \;=\; \frac{d^{(n)}f(x^*)}{dx^n} \;=\; 0 \qquad \text{en} \qquad \frac{d^{(n+1)}f(x^*)}{dx^{n+1}} \;\neq\; 0$$

Indien $n = 0$, dan spreekt men over een enkelvoudig nulpunt.

De efficientie van iteratieve methodes wordt bepaald door:

- het aantal functie-evaluaties (en of men ook afgeleiden moeten berekenen).
- het aantal iteratiestappen.

6.1 Bissectiemethode

Vereisten

- f is continu in het interval $[a, b]$ dat we gaan bekijken.
- $f(a)f(b) < 0$, dan kan men met zekerheid zeggen dat er een nulpunt is in dat interval.

Indien $f(a)f(b) > 0$, kan f nog steeds nulpunten hebben in dat interval, maar dan zijn het nulpunten met een even meervoudigheid, een even aantal nulpunten met een oneven meervoudigheid of een combinatie van de twee.

Methode
Bij elke iteratiestap wordt het onzekerheidsinterval door 2 gedeeld (lineaire complexiteit, robuust maar traag). Neem $x^{(0)} = a$, $x^{(1)} = b$, dan is:

$$x^{(2)} = \frac{x^{(0)} + x^{(1)}}{2}$$

Zij $f_2 = f(x^{(2)})$, dan zijn er 3 mogelijke gevallen:

- $f_2 = 0$: dan is $x^{(2)} = x^*$ (komt zelden voor).
- $f_2 f_0 < 0$: dan bevat het half zo groot interval $[x^{(0)}, x^{(2)}]$ de wortel.
- $f_2 f_1 < 0$: dan bevat het half zo groot interval $[x^{(1)}, x^{(2)}]$ de wortel.

Merk op dat de punten niet allemaal bijgehouden moeten worden aangezien enkel het laatste punt nodig is en ook vaak het nauwkeurigst.

Stopcriterium
Wanneer de lengte van het interval $|x^{(k+1)} - x^{(k)}| < 2\varepsilon$.

Hoeveelheid rekenwerk
Slechts 1 functie-evaluatie per iteratiestap.

6.2 Secant-methode (lineaire interpolatie)

Vereisten
Een interpolerende eerstegraadsveelterm van de vorm:

$$y_{10} \;=\; f_1 + (x - x^{(1)})\frac{f_1 - f_0}{x^{(1)} - x^{(0)}} \quad \Big(\;=\; f_1 + (x - x^{(1)}) f[x^{(0)}, x^{(1)}] \Big)$$

Ook mag er maar een nulpunt zijn tussen de beginwaarden $x^{(0)}$ en $x^{(1)}$.
Methode
Het nulpunt van de interpolerende veelterm is:

$$x^{(k+1)} \;=\; x^{(k)} - \frac{x^{(k)} - x^{(k-1)} f_k}{f_k - f_{k-1}} \tag{6.1}$$

Dit is van de vorm:

$$x^{(k+1)} \;=\; F(x^{(k-1)}, x^{(k)})$$

Stopcriterium
Men is nu niet zeker dat x^* tussen $x^{(k-1)}$ en $x^{(k)}$ ligt. Dat $|x^{(k)} - x^{(k-1)}| < \varepsilon$ wilt nu niet zeggen dat ook $|x^{(k)} - x^*| < \varepsilon$. Problemen met de secantmethode zijn:

- Er kan divergentie optreden na een bepaalde iteratie.
- Als f_{k-1} en f_k bijna dezelfde waarde hebben, wordt de noemer in 6.1 bijna 0 en $x^{(k+1)}$ kan dan buiten het domein van de functie vallen.
- Als $x^{(k+m)} = x^{(k)}$ en $x^{(k+m+1)} = x^{(k+1)}$, dan zal de methode in een oneindige lus terecht komen.

Men legt daarom een grens K_{max} op het aantal iteratiestappen.

6.3 Regula falsi methode

Vereisten

- Combineert de lineaire interpolatie van de secant-methode, en het zekerheidsinterval waar de wortel in zit van de bissectiemethode.
- Enkel voor wortels met oneven meervoudigheid.

Methode
Men gebruikt de secant-stap voor het bepalen van $x^{(k+1)}$:

$$x^{(k+1)} = x^{(k)} - \frac{f(x^{(k)})(x^{(k)} - x^{(k-1)})}{f(x^{(k)}) - f(x^{(k-1)})}$$

Er zijn dan 3 mogelijkheden:

- $f(x^{(2)}) = 0$: dan is $x^{(2)} = x^*$.
- $f(x^{(0)})f(x^{(2)}) < 0$: dan is $x^* \in (x^{(0)}, x^{(2)})$.
- $f(x^{(1)})f(x^{(2)}) < 0$: dan is $x^* \in (x^{(1)}, x^{(2)})$.

Voor elk verkleind interval voert men de secant-stap opnieuw uit. De methode convergeert dus gegarandeerd. De iteratieformule is dus:

$$x^{(k+1)} = F_{\text{secant}}(x^{(k-1)}, x^{(k)})$$

Methode
Er zijn twee stopcondities, elk gebaseerd op de oorspronkelijke methodes:

- Een maximum aantal iteratiestappen K_{max}. Deze kan niet berekend worden zoals bij de bissectiemethode, maar dient als een bovengrens voor de regentijd.

- Een nauwkeurigheid van $|x^{(k+1)} - x^{(k)}| < 2\varepsilon$. Merk op dat de methode can convergeren zonder deze nauwkeurigheid te bereiken. Daarom de eerste stopconditie.

Hoeveelheid rekenwerk
Slechts 1 functie-evaluatie per iteratiestap.

6.4 Newton-Raphson methode (lineaire Hermite interpolatie)

Vereisten
Hier wordt eveneens een eerstegraads interpolerende veelterm gebruikt, maar deze keer een eerste-orde Hermite veelterm die door het punt $(x^{(k)}, f(x^{(k)})$ gaat raakt aan de kromme $f(x)$. De functie moet ook afleidbaar zijn.

$$y_{00} = f(x^{(0)}) + f'(x^{(0)})(x - x^{(0)})$$

Methode
Het nulpunt hiervan wordt gebruikt om de volgende punten te berekenen:

$$x^{(k+1)} = x^{(k)} - \frac{f(x^{(k)})}{f'(x^{(k)})} \tag{6.2}$$

Dit is een iteratieformule dat slecht van 1 vorig punt afhankelijk is:

$$x^{(k+1)} = F(x^{(k)})$$

Stopconditie
Convergentie is niet gegarandeerd, maar als er convergeert gebeurt dit zeer snel. Het aantal beduidende cijfers neemt $\mathcal{O}(n^2)$ toe, en het algoritme stopt als de nauwkeurigheid bereikt is of het aantal iteratiestappen K_{max} overschreden is.

Hoeveelheid rekenwerk
Per evaluatiestap moet zowel de functie als de afgeleide berekend worden, wat kostelijk kan zijn.

6.5 Methode van Whittaker

Vereisten
De afgeleide bij de Newton-Raphsonmethode kan men ook benaderen door

een differentie ($f'(x^{(k)}) = f[x^{(k-1)}, x^{(k)}]$).

Methode
Indien $x^{(0)}$ niet te ver van x^* afligt, kan men veronderstellen dat $f'(x^{(k)}$ niet sterk zal variëren. Er geldt dan:

$$x^{(0)} \approx x^* : \quad f'(x^{(k)}) = f'(x^{(0)}) = m$$

Men kan voor de benadering van de afgeleide $m_k = f[x^{(k-1)}, x^{(k)}]$ nemen, en deze dan constant houden gedurende enkele iteraties.

Stopcriterium
Zelfde als bij Newton-Raphson.

Hoeveelheid rekenwerk
Slechts 1 functie-evaluatie per iteratiestap, en m_k om de paar stappen berekenen (bv. om de 5 stappen).

6.6 Methode van Muller

Vereisten
Steunt op een interpolerende veelterm van graad 2 (betere benadering maar nog steeds eenvoudig te berekenen). Er zijn dus 3 startwaarden nodig: $x^{(0)}$, $x^{(1)}$ en $x^{(2)}$. De interpolerende veelterm wordt gegeven door:

$$y_{210}(x) = f_2 + f[x^{(2)}, x^{(1)}](x - x^{(2)}) + f[x^{(2)}, x^{(1)}, x^{(0)}](x - x^{(2)})(x - x^{(1)})$$

$$\Updownarrow$$

$$y_{210}(x) = c + b(x - x^{(2)}) + a(x - x^{(2)})^2 \tag{6.3}$$

met
$$c = f_2$$
$$b = f[x^{(2)}, x^{(1)}] + (x^{(2)} - x^{(2)}) f[x^{(2)}, x^{(1)}, x^{(0)}]$$
$$a = f[x^{(2)}, x^{(1)}, x^{(0)}]$$

Methode
De 2 nulpunten van deze parabool worden gegeven door:

$$w = x^{(2)} + \frac{-b \pm \sqrt{b^2 - 4ac}}{2a} \tag{6.4}$$

We stellen $x^{(3)}$ gelijk aan het nulpunt dat het dichtst bij x^* ligt. Om numerieke stabiliteit te behouden, herschrijven we 6.4 om catastrophic cancellation te voorkomen:

$$w = x^{(2)} + \frac{2c}{b \pm \sqrt{b^2 - 4ac}}$$

Om de breuk zo klein mogelijk te maken in absolute waarde, nemen we het onderstaand nulpunt voor het volgend iteratiepunt $x^{(3)}$:

$$x^{(3)} = x^{(2)} + \frac{2c}{b + \text{sign}(b)\sqrt{b^2 - 4ac}} \tag{6.5}$$

Stopconditie
Als $|x^{(2)} - x^{(1)}| < \varepsilon$, of als er geen convergentie optreedt na K_{max} stappen.

Hoeveelheid rekenwerk
Slechts 1 functie-evaluatie per iteratiestap, namelijk het berekenen van $a_{k+1}, b_{k+1}, c_{k+1} \leftarrow a_k, b_k, c_k$.

6.6.1 Inverse interpolatie

Interpolerende veeltermen gebruiken met een graad $n > 3$ stelt een paar problemen:

- Complexiteit van het berekenen van deze nulpunten.
- Een keuze maken tussen deze nulpunten voor het volgend iteratiepunt.

Men kan oftewel overstappen naar rationele interpolatie, of inverse interpolatie gebruiken. TODO

6.7 Methode van Halley (rationele interpolatie)

Vereisten
Rationele interpolerende functie gebruiken i.p.v. een veelterm. De graad van de teller kan laag gehouden worden en men heeft nog 3 vrijheidsgraden voor het bepalen van het nulpunt. Men gebruikt een eerste- of tweedegraads Hermite veelterm in teller en noemer:

$$y(x) = \frac{a(x - x^{(0)}) - b}{c(x - x^{(0)}) - d} \tag{6.6}$$

Aangezien de functie niet veranderd als men de teller en noemer door een constante deelt, kan men $d = 1$ nemen als $d \neq 0$.

$$y(x) \;=\; \frac{a(x - x^{(0)}) - b}{c(x - x^{(0)}) - 1} \tag{6.7}$$

Methode

Er blijven in 6.7 nog 3 vrijheidsgraden over. De drie voorwaarden die we gaan opleggen zijn:

$$y(x^{(0)}) \;=\; f(x^{(0)}), \qquad y'(x^{(0)}) \;=\; f'(x^{(0)}), \qquad y''(x^{(0)}) \;=\; f''(x^{(0)})$$

Hieruit kunnen we de parameters a, b en c halen:

$$f(x^{(0)}) \;=\; b, \qquad f'(x^{(0)}) \;=\; bc - a, \qquad f''(x^{(0)}) \;=\; 2c(bc - a)$$

$$\Updownarrow$$

$$b \;=\; f(x^{(0)}), \qquad c \;=\; \frac{f''(x^{(0)})}{2f'(x^{(0)})}, \qquad a \;=\; \frac{f''(x^{(0)})f(x^{(0)})}{2f'(x^{(0)})} - f'(x^{(0)})$$

Aangezien het nulpunt van de benadering gegeven wordt door $x^{(0)} + \frac{b}{a}$, vinden we:

$$x^{(1)} = x^{(0)} - \frac{2f(x^{(0)})f'(x^{(0)})}{2(f(x^{(0)}))^2 - f(x^{(0)})f''(x^{(0)})} \tag{6.8}$$

Ook hier is $x^{(k+1)} = F(x^{(k)})$ slechts afhankelijk van 1 vorig interpolatiepunt. Dit is de *interpolatieformule van Halley*.

Hoeveelheid rekenwerk

Aangezien deze methode gebruik maakt van de functiewaarde en de eerste en tweede afgeleide, zal deze zeer snel convergeren evenredig met $\mathcal{O}(n^3)$.

6.8 Indeling van iteratieve methodes

Een iteratiemethode is afhankelijk van een rij van iteratiefuncties $F_1, F_2, \ldots$. Deze functies zijn op hun beurt afhankelijk van een rij van iteratiepunten $x^{(0)}, x^{(1)}, \ldots$. Indien $x^{(k+1)}$ enkel afhankelijk is van de s vorige iteratiepunten $x^{(k-s)}$ t/m $x^{(k)}$, noemt men dit een s-stapsmethode (de eeste punten zijn ook vaak overbodig aangezien ze slechts een ruwe schatting zijn). Zo is de secant-methode een 2-stapsmethode en de methode van Muller een 3-stapsmethode.

Wanneer F_k onafhankelijk is van k, spreekt men van *stationaire methodes*. Hieronder vallen de methodes van Newton-Raphson, Halley, Muller en de secant-methode. Ook de Whittaker-methode indien $m_k = m$ constant gehouden wordt.

6.9 Substitutiemethodes

Stationaire één-stapsmethoden worden *substitutiemethodes* genoemd, en zijn van de vorm:

$$x^{(k+1)} \;=\; F(x^{(k)})$$

Als F continu is in de buurt van x^* en als de methode convergeert, dan zal x^* een oplossing zijn van de vergelijking:

$$x - F(x) = 0 \qquad \Leftrightarrow \qquad x = F(x)$$

M.a.w. wordt x^* op zichzelf afgebeeld door F. Men zegt dan dat x^* een *vast punt* is van F. Substitutiemethodes worden daarom ook *vaste puntsmethodes* genoemd.

6.9.1 Meetkundige interpretatie van substitutiemethodes

Substitutiemethodes gaan dus op zoek naar een vast punt van de iteratiefunctie $F(x)$, oftewel het snijpunt van $F(x)$ en de bissectrice. Het berekenen van $x^{(k+1)} = F(x^{(k)})$ kan opgevat worden als het als een verticale beweging van $x^{(k)}$ naar de kromme $F(x)$, gevolgd door een horizontale beweging naar de bissectrice. De volgende mogelijkheden kunnen optreden:

- $0 < F'(x^*) < 1$: monotone convergentie, de rij $x^{(k)}$ nadert x^* monotoon (6.1a).
- $-1 < F'(x^*) < 0$: spiraalvormige convergentie, $x^{(k)}$ ligt afwisselend links en rechts van x^* (6.1b).
- $|F'(x^*)| > 1$: divergentie (6.1c en d).

In de eerste twee gevallen is x^* een aantrekkingspunt, in het geval van divergentie een afstotingspunt.

6.10 Convergentiesnelheid van rijen

De convergentiefactor in de k^{de} stap is:

$$\rho \;=\; \frac{\varepsilon^{(k)}}{\varepsilon^{(k-1)}} \tag{6.9}$$

Voldoende voorwaarde opstellen opdat de methode van Whittaker gaat convergeren.

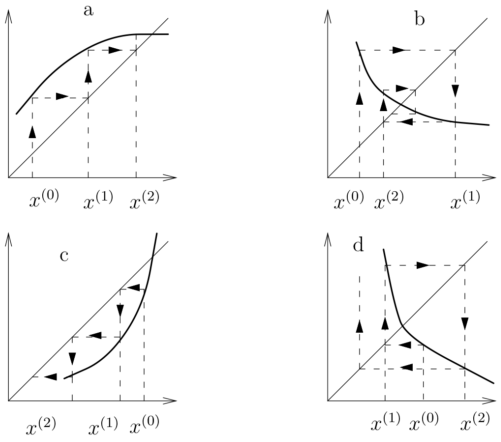


Figure 6.1: Substitutiemethode