Oplossingen oefenzittingen

Numerieke Wiskunde - G0N90a

Dit document bevat de einduitkomsten van (een deel van) de oefeningen van de oefenzittingen. Zorg ervoor dat je deze zelf kunt bekomen én dat je de oplossingen ook kan interpreteren.

In oefenzittingen 13, 14 en 16 wordt er gevraagd de convergentiefactor af te lezen uit een grafiek van de benaderingsfout voor verschillende iteratieve processen. De convergentiefactor is gedefinieerd als

$$\rho = \lim_{k\to\infty} \frac{\epsilon^{(k+1)}}{\epsilon^{(k)}}.$$

Merk op dat uit de definitie van orde (zie handboek) volgt dat als de orde groter is dan 1, de convergentiefactor ρ gelijk is aan 0.

Als de orde 1 is, dan mag je er voor k "voldoende groot" (meestal) vanuit gaan dat

$$\rho \approx \frac{\epsilon^{(k+1)}}{\epsilon^{(k)}} \quad \Longrightarrow \quad \rho^m \approx \frac{\epsilon^{(k+m)}}{\epsilon^{(k)}}. \tag{1}$$

In één stap wordt de fout ongeveer vermenigvuldigd met ρ, dus in m stappen wordt de fout ongeveer vermenigvuldigd met ρ^m. Indien je dus de waarde van de fout kan aflezen in stappen k en $k+m$, dan vind je de benadering voor ρ

$$\rho \approx \left(\frac{\epsilon^{(k+m)}}{\epsilon^{(k)}}\right)^{\frac{1}{m}}. \tag{2}$$

Door m groot genoeg te nemen, middel je afleesfouten uit en krijg je een nauwkeurigere benadering.

Eigenlijk komt dit neer op het schatten van de richtingscoëfficiënt van de rechte die je bekomt in een grafiek van de fout met logaritmische schaal op de y-as. Uit vergelijking (1) volgt namelijk dat

$$\log\left(\epsilon^{(k+1)}\right) \approx \log(\rho) + \log\left(\epsilon^{(k)}\right)$$

en bijgevolg

$$\log\left(\epsilon^{(k)}\right) \approx k\log(\rho) + \log\left(\epsilon^{(0)}\right).$$

Hieraan zie je dat je een rechte bekomt met richtingscoëfficiënt $\log(\rho)$. De richtingscoëfficiënt kan je nu schatten als

$$\log(\rho) \approx \frac{\log\left(\epsilon^{(k+m)}\right) - \log\left(\epsilon^{(k)}\right)}{m}.$$

Deze formule is iets minder handig dan formule (2), want je moet de log van de fout aflezen, en nadien moet je nog ρ halen uit $\log(\rho)$. Toon zelf aan dat deze laatste formule equivalent is met formule (2).

2 Bewegende kommavoorstelling en foutenanalyse

P1. EP getallen: 24 bit, DP getallen: 53 bit

P2. $2^{52} - 1$ tussen de getallen 1 en 2. $2^{50} + 2^{49} - 1$ tussen de getallen 7 en 9.

P3. $x_n = 1000$ voor $n \geq 1000$.

P4.

- **bepaal_b:** Vind een getal in het interval $[b^p, b^{p+1})$. In dit interval liggen twee opeenvolgende getallen op een afstand b

$$.\times\times\times\cdots\times\times \;\cdot\; b^{p+1} \quad \longrightarrow \quad \text{afstand } .000\cdots01 \cdot b^{p+1} = b.$$

De eerste lus vindt zo'n getal A. De tweede lus zoekt het eerstvolgende getal en b wordt dan gevonden als het verschil.

Als er $A \leftarrow A{+}1$ zou staat, dan is het triviaal dat er een getal in $[b^p, b^{p+1})$ gevonden wordt. De regel $A \leftarrow 2 * A$ is uiteraard veel efficiënter. (Hoeveel stappen zijn er bijvoorbeeld maar nodig om tot het getal $A = 10^{10}$ te geraken?) We bewijzen dat er met deze aanpassing zeker nog een getal gevonden wordt in $[b^p, b^{p+1})$:

$$2^k \in [b^p, b^{p+1}) \Longleftrightarrow k \in \big[\log_2(b) \cdot p, \log_2(b) \cdot (p+1)\big).$$

Omdat $\log_2(b) \geq 1$ voor $b \geq 2$ vinden we altijd zo'n getal. We zien ook dat de eerste lus $\mathcal{O}(p)$ stappen vergt i.p.v. $\mathcal{O}(b^p)$ indien er $A \leftarrow A + 1$ zou staan.

- **bepaal_p:** Aan het begin van de lus geldt er altijd dat $z = b^p$, waarbij p de variabale voorstelt. De conditie van de lus is niet meer voldaan vanaf dat $z \in [b^{p^*}, b^{p^*+1})$, waarbij p^* de precisie voorstelt. Bijgevolg moet $p = p^*$.

P5.

$$\left|\frac{\bar{y} - y}{y}\right| \leq \left(\frac{3}{2}\frac{\sqrt{x+1}}{\sqrt{x+1}+1} + 2\right)\epsilon_{mach} \leq \frac{7}{2}\,\epsilon_{mach}$$

P6.

$$\left|\frac{\bar{y} - y}{y}\right| \leq \left(\left|\frac{\cos(x)}{1-\cos(x)}\right| + 3\right)\epsilon_{mach}$$

P8.

(a) $$\left|\frac{\bar{y} - y}{y}\right| \leq 2\,\epsilon_{mach}$$

(b) $$\left|\frac{\bar{y} - y}{y}\right| \leq \left(\left|\frac{\cos(x)}{1-\cos(x)}\right| + 3\right)\epsilon_{mach}$$

(c) $$\left|\frac{\bar{y} - y}{y}\right| \leq \left(\left|\frac{e^{-2x}}{1-e^{-2x}}\right| + 2\right)\epsilon_{mach}$$ (Geen fout voor 2*x in basis 2.)

(d) $$\left|\frac{\bar{y} - y}{y}\right| \leq \left(\left|\frac{1}{x}\right| + |\log(y)| + 1\right)\epsilon_{mach}$$

(e) $$\left|\frac{\bar{y} - y}{y}\right| \leq \left(\left|\frac{1}{2}\frac{e^x}{e^x-1}\right| + \frac{3}{2}\right)\epsilon_{mach}$$

(f) $$\left|\frac{\bar{y} - y}{y}\right| \leq \left(\left|\frac{1}{x}\cot\left(\frac{1}{x}\right)\right| + 1\right)\epsilon_{mach}$$

(g) $$\left|\frac{\bar{y} - y}{y}\right| \leq \left(\left|\log(y) + \frac{x^4}{1+x^2}\right| + x^2 + 1\right)\epsilon_{mach}$$ (Dezelfde ϵ_i voor de bewerking x^2!)

(h) $$\left|\frac{\bar{y} - y}{y}\right| \leq \left(\left|\frac{e^{x^2}+e^{-x^2}}{e^{x^2}-e^{-x^2}}\cdot x^2 - 1\right| + \left|\frac{2e^{x^2}}{e^{x^2}-e^{-x^2}}\right| + 2\right)\epsilon_{mach}$$ (id.)

P9.

$$\left|\frac{\overline{P}-P}{P}\right| \leq (n-1)\,\epsilon_{mach}$$

$$\begin{aligned}\left|\overline{SP}-SP\right| &= \left|\sum_{i=1}^{n}\epsilon_i(a_ib_i)+\sum_{i=1}^{n-1}\delta_i\left(\sum_{k=1}^{i+1}a_kb_k\right)\right| \\ &\leq \left(\sum_{i=1}^{n}|a_ib_i|+\sum_{i=1}^{n-1}\left|\sum_{k=1}^{i+1}a_kb_k\right|\right)\epsilon_{mach}\end{aligned}$$

3 (PC) Bewegende kommavoorstelling en foutenanalyse

P2. $(0.1)_{10} = (1.100110011001\ldots)_2 \times 2^{-4}$, $fl(0.1) = (1.10011001\ldots10011010)_2 \times 2^{-4}$ (52 cijfers na de komma)

$\implies fl(0.1) - 0.1 \approx (0.00000000\ldots000000001)_2 \times 2^{-4}$ met 1 het 53ste cijfer na de komma
$= 2^{-53}\cdot 2^{-4} = 2^{-57}$

$\implies$ relatieve fout $= 2^{-57}/0.1 \approx 6.9\cdot 10^{-17}$

P3.

(a) Vanaf $k = 11$ is de term $x^{k-1}/k!$ in `t(k)` kleiner dan `eps(y(k-1))/2`.

(b) $\delta y = 4.0e-16$

(c) `semilogy` geeft een rechte.

(d) $fout_k \sim \dfrac{x^{k+1}}{(k+1)!} =$ `t(k+2)`

P4. Iets tragere convergentie. Vanaf $k = 16$ is de fout exact nul, waardoor ze niet meer wordt weergegeven. Merk op dat de echte fout niet exact nul is, want je vergelijkt met MATLAB `exp(x)`, wat ook een benadering is.

P5.

(a) De termen $x^k/k!$ nemen pas af vanaf $k = 21$ en het duurt nog langer voor de termen echt klein worden.

(b) $\Delta y = 1.2e-7, \delta y = 2.5e-16$

(c) Goede benadering, want $\delta y \approx \epsilon_{mach}$. De absolute fout is veel groter, maar dit komt omdat het getal $\exp(20) \approx 4.8e+8$.

(d) $\bar{S} - S \approx \sum_{i=2}^{n}\epsilon_i\,(a_1+\ldots+a_i)$. Alle termen zijn hier positief, dus $\bar{S}-S \leq \epsilon_{mach}\cdot n\sum_{j=1}^{n}a_j = \epsilon_{mach}\cdot n\cdot S$. Numeriek blijkt dat dit een serieuze overschatting is van de absolute fout.

P6.

(a) $\Delta y = 2.1e-9, \delta y = 1.02$

(b) Grote relatieve fout (1 of geen juiste beduidende cijfers), slechte benadering. Het probleem is dat de absolute fout vrij groot is, en dat ditmaal het getal $\exp(-20) \approx 2.06e-9$ redelijk klein is, waardoor de relatieve fout opblaast.

(c) $\bar{S} - S \approx \sum_{i=2}^{n} \epsilon_i \sum_{j=1}^{i} a_j$, met $\sum_{j=1}^{i} a_j =$ `y(i-1)`. De grootste getallen in de vector `y` zijn van grootte-orde 10^7, bijgevolg is $\bar{S} - S$ van grootte-orde $10^7 \epsilon_{mach} \approx 1e-9$.

Je krijgt een zeer slecht numeriek resultaat, omdat je grote absolute fouten maakt en op het einde deelt door een klein getal. Dit gebeurt hier als de som van grote getallen met een wisselend teken en is een mooi voorbeeld van een gevaarlijke aftrekking.

P8. Zoek ook eens op het internet.

P9.

(a) **lus1:** $\dfrac{\bar{y}-y}{y} \approx \dfrac{1}{2^{39}}\epsilon_1 + \dfrac{1}{2^{38}}\epsilon_2 + \ldots + \epsilon_{40}$

$$\Rightarrow \left|\frac{\bar{y}-y}{y}\right| \leq \epsilon_{mach} \frac{1-\left(\frac{1}{2}\right)^{40}}{1-\frac{1}{2}} \leq 2\,\epsilon_{mach}$$

lus2: $\dfrac{\bar{y}-y}{y} \approx 2^{40}\delta + 2^{39}\epsilon_1 + 2^{38}\epsilon_2 + \ldots + \epsilon_{40}$

(d) Nee, want het gegeven voor de tweede lus is de output van de eerste lus, dus er zit een relatieve fout op van grootte-orde ϵ_{mach}. Uit de foutenanalyse leid je af dat de tweede lus een slecht geconditioneerd probleem is. Kleine fouten op het gegeven worden opgeblazen, ook als er exact wordt gerekend.

P10.

(a) Een verband zoals $\mathcal{O}(x^{-2})$ plot je het best met een `loglog` grafiek, waarom?

(b) $f(x) = 1 + \dfrac{x^4}{3!} + \dfrac{x^8}{5!} + \dfrac{x^{12}}{7!} + \mathcal{O}(x^{16})$, de eerste vier termen volstaan om voor de gegeven x-waarden een fout te hebben van grootte-orde ϵ_{mach}.

(c) $$\left|\frac{\bar{y}-y}{y}\right| \leq \left(\left|\frac{e^{x^2}+e^{-x^2}}{e^{x^2}-e^{-x^2}} \cdot x^2 - 1\right| + \left|\frac{2e^{x^2}}{e^{x^2}-e^{-x^2}}\right| + 2\right)\epsilon_{mach}$$ (Dezelfde ϵ_i voor de bewerking x^2!)

$$\sim \left(\left|\frac{2+x^4}{2x^2} \cdot x^2 - 1\right| + \left|\frac{2}{2x^2}\right| + 2\right)\epsilon_{mach} \sim \left|\frac{1}{x^2}\right| \quad x \to 0$$

waarbij we gebruik hebben gemaakt van Taylor reeksen.

4 Conditie en stabiliteit

P1. $\dfrac{f'(x)x}{f(x)} = \dfrac{2x^2+4x+3}{(1+x)(3+2x)}$, slechte conditie voor $x \approx -1$ en $x \approx \frac{-3}{2}$

1. $$\left|\frac{\bar{y}-y}{y}\right| \leq \left(1 + \frac{|1+2x||1+x|}{|3+2x||x|} + \frac{2}{|3+2x||x|}\right)\epsilon_{mach}$$ (geen fout bij vermenigvulding met 2)
   - onstabiel voor $x \approx 0$
   - zwak stabiel voor $x \approx \frac{-3}{2}$
   - voorwaarts stabiel voor andere waarden van x

   Vraag: geeft dit algoritme een nauwkeurig resultaat voor $x \approx 1$?

2. $$\left|\frac{\bar{y}-y}{y}\right| \leq 4\,\epsilon_{mach}$$ (geen fout bij vermenigvulding met 2)

- voorwaarts stabiel voor alle waarden van x

$\rightarrow$ beter algoritme

Opmerking: wanneer je op een machine met basis 2 vermenigvuldigt met of deelt door 2, dan maak je geen relatieve fout, waarom? Als je hier toch een relatieve fout zou doorvoeren dan moet je in de tweede term van 1. $|1+2x|$ vervangen door $(|1+2x|+|2x|)$ en in 2. staat er dan $(4+|2x|/|3+2x|)\epsilon_{mach}$, waardoor algoritme 2. zwak stabiel is voor $x \approx -3/2$.

P2.

- $\delta_c y = \big(1 + x\cot(x)\big)\delta x$, slechte conditie voor $x \approx k\pi$, $k \neq 0$. (Waarom $k \neq 0$?)
  $\left|\dfrac{\bar{y}-y}{y}\right| \leq 2\,\epsilon_{mach}$, voorwaarts stabiel algoritme

- $\delta_c f = \dfrac{1}{2}\dfrac{\sqrt{a}\,\delta a + \sqrt{b}\,\delta b}{\sqrt{a}-\sqrt{b}}$, slechte conditie voor $a \approx b$, nog eens versterkt wanneer a en b heel groot zijn.

  1. **eval1:** $\left|\dfrac{\bar{y}-y}{y}\right| \leq \left(1 + \dfrac{\sqrt{a}+\sqrt{b}}{\sqrt{a}-\sqrt{b}}\right)\epsilon_{mach}$, zwak stabiel voor $a \approx b$
  2. **eval2:** $\left|\dfrac{\bar{y}-y}{y}\right| \leq 4\,\epsilon_{mach}$, voorwaarts stabiel algoritme

P3. $g(x)$ goed geconditioneerd $\Rightarrow g(x+\delta x) = g(x)(1+C_1\delta x)$, met C_1 een niet al te grote constante. Stabiel algoritme $\hat{g}(x) \Rightarrow \hat{g}(x) = g(x)(1+C_2\epsilon)$, met $|\epsilon| \leq \epsilon_{mach}$ en C_2 een niet al te grote constante. Bijgevolg

$$\hat{g}(x(1+\delta x)) = g(x)(1+C\epsilon')$$

met C een niet al te grote constante en $|\epsilon'| \leq \epsilon_{mach}$ als $|\delta x| \leq \epsilon_{mach}$. Uitwerken geeft als eindresultaat

$$\begin{aligned}
\frac{\bar{y}-y}{y} &= \frac{g(x+h)}{g(x+h)-g(x)}\delta_1 - \frac{g(x)}{g(x+h)-g(x)}\delta_2 + \epsilon_1 + \epsilon_2 \\
&= \frac{g(x+h)}{g'(\xi)h}\delta_1 - \frac{g(x)}{g'(\xi)h}\delta_2 + \epsilon_1 + \epsilon_2, \qquad x \leq \xi \leq x+h \\
\Rightarrow \left|\frac{\bar{y}-y}{y}\right| &\leq \mathcal{O}(\frac{1}{h})\,\epsilon_{mach},
\end{aligned}$$

onder de voorwaarde dat g continu afleidbaar is in een omgeving van x.

P5.

- $\delta_c y = \dfrac{2(1+x)}{2+x}$

  1. **eval1** $\left|\dfrac{\bar{y}-y}{y}\right| \leq \left(\dfrac{3(1+x)^2}{|2+x||x|}+1\right)\epsilon_{mach}$
  2. **eval2** $\left|\dfrac{\bar{y}-y}{y}\right| \leq 2\,\epsilon_{mach}$

- $\delta_c y = \dfrac{-2e^{2x}}{e^{2x}-1}\delta x$, slechte conditie in $x \approx 0$

1. **eval1** $\left|\frac{\bar{y}-y}{y}\right| \leq \left(2\left|\frac{e^{2x}}{e^{2x}-1}\right| + (e^x+1) + |e^x-1| + 1\right)\epsilon_{mach}$
   - zwak stabiel voor $x \approx 0$
   - onstabiel voor $x \to \infty$ (in de praktijk voor $x >> 1$)
2. **eval2** $\left|\frac{\bar{y}-y}{y}\right| \leq \left(\left|\frac{e^{2x}}{e^{2x}-1}\right| + 2\right)\epsilon_{mach}$
   - zwak stabiel voor $x \approx 0$
   - voorwaarts stabiel voor andere waarden van x

5 (PC) Afrondings- en benaderingsfouten

P1-P2-P3 samenvattende analyse: Formule (1) heeft orde 1, want de benaderingsfout neemt af zoals $\mathcal{O}(h)$ als $h \to 0$. Bewijs dit m.b.v een Taylor reeks van $f(h)$ rond 0. Bewijs ook dat de benaderingsfout voor formule (2) afneemt zoals $\mathcal{O}(h^2)$ als $h \to 0$ en de orde dus gelijk is aan 2.

Als de benaderingsfout voldoet aan $E_{benadering} = Ch^p$ [1] voor een constante C, dan geldt er dat $\log(E_{benadering}) = \log(C) + p\log(h)$ en dan krijg je in een `loglog` grafiek van $E_{benadering}$ i.f.v. h dus een rechte met richtingscoëfficiënt gelijk aan p. Als de richtingscoëfficiënt van de rechte die overeenkomt met de benaderingsfout van formule (1) gelijk is aan 1, dan is het duidelijk uit de figuur dat de afrondingsfouten ongeveer op een rechte liggen met richtingscoëfficiënt gelijk aan -1. De afrondingsfouten nemen dus toe zoals $\mathcal{O}(h^{-1})$ als $h \to 0$. Dit volgt inderdaad uit de foutenanalyse in **P3** van oefenzitting 4. Bijgevolg geldt er dat voor een constante D

$$E_{afronding} \approx Dh^{-1}\epsilon_{mach}.$$

Wat is nu de kleinste fout die je kan bekomen met beide formules? De fout bestaat uit de benaderingsfout en de afrondingsfouten

$$E \approx Ch^p + Dh^{-1}\epsilon_{mach}.$$

Als h groot is domineert de eerste term en als h klein is de tweede. We vinden de kleinste fout als beide termen ongeveer even groot zijn

$$Ch^p \approx Dh^{-1}\epsilon_{mach} \Leftrightarrow h \approx \left(\frac{D}{C}\right)^{\frac{1}{p+1}} (\epsilon_{mach})^{\frac{1}{p+1}}.$$

Voor formule 1 is $p = 1$, en krijgen we dus de kleinste fout bij $h \approx (\epsilon_{mach})^{\frac{1}{2}} \approx 10^{-8}$ en de fout is dan $E \approx 10^{-8}$, voor formule (2) krijgen we $h \approx (\epsilon_{mach})^{\frac{1}{3}} \approx 10^{-5}$ en de fout is dan $E \approx 10^{-10}$. Dit komt overeen met de observaties op de figuur. Formule (2) geeft een nauwkeuriger resultaat, voor een grotere waarde van h.
(Omdat enkel de grootte ordes ons interesseren, hebben we de constanten C en D van grootte orde $\mathcal{O}(1)$ genomen. Voor ϵ_{mach} hebben we 10^{-16} genomen voor formule (1) en 10^{-15} voor formule (2) om het rekenwerk te vereenvoudigen.)

6 Veelterminterpolatie en numerieke integratie

P1.

(a) Beide methoden geven $y_1(x) = 4x - 3$.

[1] Dit is ongeveer hetzelfde als $E_{benadering} = \mathcal{O}(h^p)$.

(b) $[-3\ \ 4]^T$ is een oplossing van het Vandermonde stelsel.

(c) $f''(x) = 4 \Longrightarrow$ formule (4.10) : $E_1(x) = 2(x+1)(x-1) = f(x) - y_1(x)$.

P2. $y_2(x) = f(x)$, $f^{(3)}(x) = 0 \Longrightarrow$ formule (4.10) : $E_2(x) = 0 = f(x) - y_2(x)$. (Leg het verband met Probleem 4.)

P3. $y_3(x) = f(x)$. Verklaar.

P4. Als $f = p$ een veelterm is van graad $\leq n$, dan geldt er dat $y_n = p$. Neem $p(x) = 1$ en $p(x) = x^k$.

P5. Bewijs dat het rechterlid de interpolerende veelterm van graad n is. Dit houdt in dat y_n een veelterm moet zijn van graad n die voldoet aan de interpolatievoorwaarden.

P6. Gebruik de hint. Op de functiewaarde bij $x = 1.70$ staat een absolute fout van 0.1234.

P7. De afgeleide van een product is gelijk aan ...

P10. $H_0 = H_1 = 1$, de nauwkeurigheidsgraad is 1. De formule is interpolerend.

P11. $H_0 = -\frac{2}{3}h, H_{-\frac{1}{2}} = H_{\frac{1}{2}} = \frac{4}{3}h$, de nauwkeurigheidsgraad is 3. Hint: gebruik i.p.v. de basis $1, x, x^2, \ldots$ de equivalente basis $1, (x-a), (x-a)^2, \ldots$ om het rekenwerk te vereenvoudigen. Waarom mag dit?

P12. $H_{-1} = H_1 = \frac{7}{15}, H_0 = \frac{16}{15}, H'_{-1} = \frac{1}{15}, H'_1 = -\frac{1}{15}$, de nauwkeurigheidsgraad is 5.

7 (PC) Het oplossen van stelsels lineaire vergelijkingen

P1. (a) $L * U \approx A$ tot op machine-nauwkeurigheid. Indien je L en U kent, dan kan je de determinant van A berekenen als `prod(diag(U))`.
(b) L is niet benedendriehoeks. Dit komt omdat er pivotering is toegepast zoals in Algoritme 3.4 in het handboek. Je krijgt $PA = LU \Longleftrightarrow A = P^{-1}LU$, en de `L` die je van Matlab terugkrijgt is gelijk aan $P^{-1}L$. De matrix P^{-1} permuteert enkel de rijen van L. Hoe weet je dat P^{-1} net zoals P een permutatiematrix is?

P2. P3. De onderstaande tabel geeft de relatieve fouten, residu's en conditiegetallen weer. Merk op dat dit grootte-ordes zijn en dat de precieze waarden kunnen verschillen per uitvoer, omdat de matrices randomheid bevatten. De tabel leert ons het volgende:

- Voor de eerste matrix zijn alledrie de methoden achterwaarts stabiel, want de residu's zijn klein. Dit geldt ook voor de laatste matrix. Voor de tweede matrix is gauss1 echter niet meer achterwaarts stabiel, wat je kan zien aan het 'veel grotere' residu. De instabiliteiten zijn hier te wijten aan het feit dat gauss1 geen optimale rijpivotering toepast, en de tweede matrix op een bepaald moment een zeer kleine pivot zal bevatten. Wanneer optimale pivotering wel wordt toegepast, in gauss2, dan krijgen we wel een stabiele methode[2].

- Het conditiegetal van de matrix A geeft een verband tussen het residu en de relatieve fout op de berekende oplossing x, de ongelijkheid in formule (2) in de opgave. Je kan dit verband inderdaad nagaan voor de waarden in de tabel.

- Indien het conditiegetal van de matrix A groot is, zoals voor de derde matrix, dan kan zelfs een achterwaarts stabiele methode een grote relatieve fout op de berekende oplossing geven.

[2]Ter info: gauss2 is voor bijna alle matrices achterwaarts stabiel, het algoritme met qr is voor alle matrices achterwaarts stabiel.

	$\kappa_2(A)$		gauss1	gauss2	qr
genmatrix1		$\|\|\delta x\|\|_2$	10^{-15}	10^{-15}	10^{-15}
	10^2	$\|\|r\|\|_2/\|\|b\|\|_2$	10^{-16}	10^{-16}	10^{-16}
genmatrix2		$\|\|\delta x\|\|_2$	10^{-7}	10^{-15}	10^{-15}
	10^1	$\|\|r\|\|_2/\|\|b\|\|_2$	10^{-8}	10^{-16}	10^{-16}
genmatrixc		$\|\|\delta x\|\|_2$	10^{-5}	10^{-6}	10^{-5}
	10^{11}	$\|\|r\|\|_2/\|\|b\|\|_2$	10^{-16}	10^{-17}	10^{-16}

11 Substitutiemethodes en Newton-Raphson

P1. Uit $\epsilon^{(k)} = O\left(\left(\epsilon^{(k-1)}\right)^p\right)$ volgt via de definitie van 'Grote O' dat $\lim_{k\to\infty} \left|\frac{\epsilon^{(k)}}{\left(\epsilon^{(k-1)}\right)^p}\right| \leq M$ met $M < \infty$. Uit $\epsilon^{(k)} \neq o\left(\left(\epsilon^{(k-1)}\right)^p\right)$ volgt dat $\lim_{k\to\infty} \left|\frac{\epsilon^{(k)}}{\left(\epsilon^{(k-1)}\right)^p}\right| > 0$. Hieruit haal je dat

$$\lim_{k\to\infty} \frac{\varepsilon^{(k+1)}}{\left[\varepsilon^{(k)}\right]^p} = \rho_p, \quad \text{met} \quad 0 < \rho_p < \infty.$$

We kunnen dan het volgende schrijven:

$$\begin{aligned}
\lim_{k\to\infty} \frac{\varepsilon^{(k+1)}}{\left[\varepsilon^{(k)}\right]^n} &= \lim_{k\to\infty} \left(\frac{\varepsilon^{(k+1)}}{\left[\varepsilon^{(k)}\right]^p} \frac{1}{\left[\varepsilon^{(k)}\right]^{n-p}}\right) \\
&= \lim_{k\to\infty} \left(\frac{\varepsilon^{(k+1)}}{\left[\varepsilon^{(k)}\right]^p}\right) \lim_{k\to\infty} \left(\frac{1}{\left[\varepsilon^{(k)}\right]^{n-p}}\right) \\
&= \rho_p \lim_{k\to\infty} \left(\frac{1}{\left[\varepsilon^{(k)}\right]^{n-p}}\right).
\end{aligned}$$

Aangezien het hier gaat over een convergerende rij van benaderingen is $\lim_{k\to\infty} \varepsilon^{(k)} = 0$, dus is de bovenstaande limiet gelijk aan ∞ voor $n > p$ en gelijk aan 0 voor $n < p$.

P2. (a) volledig consistent, spiraalvormige divergentie, $F'(x^*) = -1.763\ldots$
(b) volledig consistent, spiraalvormige convergentie, $F'(x^*) = -0.5671\ldots$

P3. Voor deze oefening moet je eerst $F(x)$ bepalen volgens de methode van Newton-Raphson, $F(x) = x - f(x)/f'(x)$. Uit de gevallenstudie van Newton-Raphson volgt dat de methode altijd consistent is, maar over reciproke consistentie weet je nog niets. Er volgt ook uit dat de convergentie voor enkelvoudige nulpunten kwadratisch is.

- $f(x) = 1 - \frac{a}{x^2}$, $F(x) = \frac{x}{2}\left(3 - \frac{x^2}{a}\right)$, consistent, niet reciprook consistent, want $x = 0$ is een vast punt van F maar geen nulpunt van f.
- De convergentiefactor is gelijk aan $F'(x^*) = 0$, dus de orde is 2.
- Je zou m.b.v. een zelfgetekende figuur toch zeker het gebied van divergentie $\{x : |x| > \sqrt{5a}\}$ moeten kunnen aanduiden en de gebieden van convergentie $\{x : x > 0, x < \sqrt{3a}\}$ en $\{x : x < 0, x > -\sqrt{3a}\}$. Als je de figuur goed kan interpreteren zijn de precieze waarden van de intervallen minder belangrijk.

P4. Gebruik de afleiding in het handboek bij de gevalstudie van de methode van Newton-Raphson. Voor de duidelijkheid noemen we $\hat{m}$ de multipliciteit van de wortel x^* van f en m de waarde van de parameter in de formule voor $F(x)$. Het is duidelijk dat

$$F'(x) = 1 - \left(m \frac{f(x)}{f'(x)} \right)'.$$

Bij Newton-Raphson zou m gelijk zijn aan 1 in bovenstaande formule en volgens de gevallenstudie geldt er dat $F'(x^*) = 1 - 1/\hat{m}$. Bijgevolg geldt er voor de aangepaste methode dat

$$F'(x^*) = 1 - \frac{m}{\hat{m}}.$$

Indien $m = \hat{m}$ dan is de methode kwadratisch. **Antwoord denkvraag:** Meestal weet men echter niet op voorhand wat de multipliciteit zal zijn van de wortels die men zoekt en kan men de waarde van m moeilijk vastleggen. Indien men bovendien $m > 1$ zou nemen, dan geldt er voor enkelvoudige wortels ($\hat{m} = 1$) dat $F'(x^*) = 1 - m <= -1$, en dan is er zelfs geen convergentie.

P5. (a) volledig consistent, monotone convergentie, $F'(x^*) = 0.216\ldots$
(b) consistent want $x = 0$ is een vast punt, maar geen nulpunt[3], spiraalvormige convergentie, $F'(x^*) = -0.3816\ldots$ Merk op dat de methode hier convergeert voor alle waarden van $x \in (0, 1]$. Dit volgt niet uit een stelling, maar kan je grafisch afleiden.

P6.

- $f(x) = x - \dfrac{a}{x}, F(x) = \dfrac{2ax}{x^2 + a}$, consistent, niet reciprook consistent, want $x = 0$ is een vast punt van F maar geen nulpunt van f
- De convergentiefactor is $F'(x^*) = 0$, dus de orde is 2.
- Je zou m.b.v. een zelfgetekende figuur moeten kunnen zien dat de convergentie monotoon verloopt naar de nulpunten en dat er voor alle $x \neq 0$ convergentie is naar $\sqrt{a}$ voor $x > 0$ en naar $-\sqrt{a}$ voor $x < 0$.

13 Iteratieve methoden voor het oplossen van stelsels

P1. $x^{(0)}, x^{(1)}, f(x^{(0)})$ zijn vectoren in $\mathbb{R}^n$, terwijl de functie $J(x)$ die je meegeeft als input, een matrix in $\mathbb{R}^{n \times n}$ teruggeeft. I.p.v. een deling door de afgeleide in Newton-Raphson voor één veranderlijke, krijg je nu een formule met de inverse van de Jacobiaan. Hier los je uiteraard een stelsel op. Tenslotte gebruik je $||x^{(1)} - x^{(0)}|| < \epsilon$ als absoluut stopcriterium of $||x^{(1)} - x^{(0)}|| \leq \epsilon ||x^{(0)}||$ als relatief stopcriterium of een combinatie van beiden.

P2.

(a) $J = \begin{bmatrix} 2x + 1 & -2y \\ -2x\cos(x^2) & 1 \end{bmatrix}$

(b) orde 2, want in figuur 1 zie je dat (voor k groot genoeg) de fout in elke stap wordt gekwadrateerd. Door de log schaal op de verticale as komt dit erop neer dat de afstand op de grafiek van 10^0 tot de waarde van de fout in elke stap verdubbelt. (Verklaar dit!) Als de orde 2 is, dan moet de convergentiefactor ρ gelijk zijn aan 0. In tabel 1 zie je ongeveer een verdubbeling van het aantal juiste beduidende cijfers. Dit aantal is voor $x^{(k)}$ (ongeveer) gelijk aan $1, 3, 6, 12$ voor respectievelijk $k = 6, 7, 8, 9$, waarbij we vergelijken met de waarde voor $k = 10$.

[3]Je kan ook argumenteren dat $F(x)$ niet gedefinieerd is voor $x = 0$, waardoor dit geen vast punt is. Indien je $F(0)$ definieert als de waarde van de limiet van F in 0, dan is $x = 0$ wel een vast punt van F.

(c) Evalueren van $\det(J)$ voor $(x, y) = \left(x^{(10)}, y^{(10)}\right)$ geeft ongeveer de waarde 1.1897. Omdat de Jacobiaan is duidelijk regulier is in de oplossing, is de convergentie kwadratisch.

(d) In de figuur kan je goed zien dat het sterk afhangt van de startwaarde of er al dan niet convergentie zal optreden. Zelfs voor een startwaarde dichtbij de oplossingen is divergentie mogelijk.

P3.

(a) $J = \begin{bmatrix} 2x & 8y \\ y^2 - 1 & 2xy \end{bmatrix}$

(b) orde 1, want in figuur 2 zie je dat (voor k groot genoeg) de fout in elke stap met een vast getal kleiner dan 1 wordt vermenigvuldigd. Door de log schaal op de verticale as komt dit erop neer dat de afstand op de grafiek van 10^0 tot de waarde van de fout in elke stap vermeerdert met een vast getal. (Verklaar dit!) Je kan een schatting van de convergentiefactor aflezen van de grafiek, je bekomt $\rho \approx 0.5$. In tabel 2 zie je dat de fout in elke stap inderdaad ongeveer halveert.

(c) $J(2, \sqrt{3})) = \begin{bmatrix} 4 & 8\sqrt{3} \\ 2 & 4\sqrt{3} \end{bmatrix} \longrightarrow \det(J) = 0$, omdat de Jacobiaan singulier is in de oplossing is de convergentie lineair. De partiële afgeleiden van een functie in een punt bepalen de richtingscoëfficiënt van de raaklijn aan die functie in dat punt. Als twee functies in een snijpunt proportionele partiële afgeleiden hebben, d.w.z. de Jacobiaan is singulier, dan hebben ze in dit snijpunt dus dezelfde raaklijn. In figuur 2 zie je dat de twee functies elkaar inderdaad raken.

P4. Uit de grafische illustratie (zie ook het handboek) volgt dat de methoden convergeren. Indien de vergelijkingen van plaats worden gewisseld, dan divergeren de methoden.

P6. Voor de exacte oplossing X van het stelsel geldt er $AX = B$ en dus $UX + DX + LX = B$. Trek deze vergelijking (na wat herschikken) af van de iteratieformules voor Jacobi en Gauss-Seidel in matrix notatie. Je bekomt, respectievelijk voor Jacobi en Gauss-Seidel, een verband tussen de opeenvolgende iteratiefouten. Hieruit haal je gemakkelijk het te bewijzen.

P7. Jacobi: $G = \begin{bmatrix} 0 & 1/2 \\ 1/2 & 0 \end{bmatrix}$, $||G||_\infty = 1/2 < 1 \Longrightarrow$ convergentie. Dit moet uiteraard ook blijken uit de spectraalradius. De eigenwaarden van G kan je berekenen als $\lambda = \pm 1/2$, bijgevolg $\rho(G) = 1/2 < 1$.

Gauss-Seidel: $G = \begin{bmatrix} 0 & 1/2 \\ 0 & 1/4 \end{bmatrix}$, $||G||_\infty = 1/2 < 1 \Longrightarrow$ convergentie. De eigenwaarden van G kan je berekenen als 0 en $1/4$, bijgevolg $\rho(G) = 1/4 < 1$.

P8. Jacobi: $G = \begin{bmatrix} 0 & -3/2 \\ -1/2 & 0 \end{bmatrix}$, $||G||_\infty = 3/2 > 1 \longrightarrow$ zegt niets over convergentie. De eigenwaarden van G kan je berekenen als $\lambda = \pm\sqrt{3}/2$, bijgevolg $\rho(G) = \sqrt{3}/2 < 1$. De methode van Jacobi convergeert dus, hoewel $||G||_\infty > 1$. Hier zie je duidelijk dat dit een voldoende, maar niet nodige voorwaarde is.

Gauss-Seidel: $G = \begin{bmatrix} 0 & -3/2 \\ 0 & 3/4 \end{bmatrix}$, $||G||_\infty = 3/2 > 1 \longrightarrow$ zegt niets over convergentie. De eigenwaarden van G kan je berekenen als 0 en $-3/4$, bijgevolg $\rho(G) = 3/4 < 1$. De methode van Gauss-Seidel convergeert dus, hoewel $||G||_\infty > 1$.

Uit de figuur kan je de convergentiefactoren aflezen. Voor Jacobi kan je de fouten aflezen voor $k = 0$ en $k = 80$ als respectievelijk 10^0 en 10^{-5}. Hiermee bekom je een schatting $\rho \approx 0.866$. Voor Gauss-Seidel kan je de fouten aflezen voor $k = 40$ en $k = 80$ als respectievelijk 10^{-5} en 10^{-10}. Hiermee bekom je een schatting $\rho \approx 0.75$. De convergentiefactoren zijn dus precies de spectraalradii!

14 (PC) Iteratieve methoden voor het oplossen van stelsels

P1. Als je de fouten plot met de juiste schaal op de assen zie je twee rechten. De convergentie is dus lineair. De methode van Gauss-Seidel convergeert sneller dan de methode van Jacobi. De spectraalradius voor Jacobi is $\rho(G) \approx 0.3536$ en voor Gauss-Seidel $\rho(G) \approx 0.125$. Dit zijn precies de convergentiefactoren die je ook kan aflezen van de grafieken of kan schatten m.b.v. MATLAB.

P3. (a)(b) De totale stapmethode en de enkelvoudige stapmethode kunnen allebei convergeren naar de onderste oplossing, mits de startwaarde goed genoeg gekozen is. Het is niet mogelijk om te convergeren naar de bovenste oplossing.
(c) De convergentie is lineair, dus de orde is 1. De convergentiefactoren zijn ongeveer gelijk aan 0.3 voor de totale stapmethode en 0.09 voor de enkelvoudige stapmethode.
(d) Als je x berekent uit de tweede en y uit de eerste vergelijking, dan is er convergentie mogelijk naar de bovenste oplossing. In het algemeen moet je, om te convergeren naar een bepaalde oplossing, x oplossen uit de vergelijking die in deze oplossing de steilste helling heeft.

P4. Vervang z door $x + iy$ en werk de complexe vergelijking uit. Je krijgt iets van de vorm $u(x, y) + iv(x, y) = 0$. Je bekomt het stelsel

$$\begin{cases} u(x, y) = 0 \\ v(x, y) = 0. \end{cases}$$

P5. De Cauchy-Riemann voorwaarden staan in het handboek.

16 (PC) Nulpunten van een veelterm

P1. Een formule voor Δc kan als volgt bekomen worden, door gebruik te maken van Taylor reeksen en door tweede orde termen te verwaarlozen (we veronderstellen dat de coëfficiënten van $\Delta p(x)$ veel kleiner zijn dan die van $p(x)$):

$$\begin{aligned} 0 = \overline{p}(c + \Delta c) &= p(c + \Delta c) + \Delta p(c + \Delta c) \\ &\approx p(c) + p'(c)\Delta c + \Delta p(c) + \Delta p'(c)\Delta c \\ &\approx p'(c)\Delta c + \Delta p(c) \end{aligned}$$

en bijgevolg

$$\Delta c \approx -\frac{\Delta p(c)}{p'(c)} = \frac{-\sum_{j=0}^{n} \Delta a_j c^{n-j}}{p'(c)}. \tag{3}$$

We hebben twee keer een Taylor benadering van eerste orde gebruikt. Vermits $\Delta p(x)$ een kleine perturbatie is van een veelterm, is ook de afgeleide $\Delta p'(x)$ een veelterm met kleine coëfficiënten van ongeveer dezelfde grootte-orde als $\Delta p(x)$. Bijgevolg is $\Delta p'(c)\Delta c$ een tweede orde term die we mogen verwaarlozen t.o.v. de term $\Delta p(c)$.

P2. Indien c een m-voudig nulpunt is van $p(x)$, dan is $p'(c) = 0$, dus geldt de eerste orde benadering die we in de vorige oefening gebruikt hebben niet meer. We hebben wel opnieuw

$$0 = \overline{p}(c + \Delta c) = p(c + \Delta c) + \Delta p(c + \Delta c)$$

Er geldt dat $p^{(k)}(c) = 0$, $k = 0, \ldots, m-1$, dus we krijgen

$$p(c + \Delta c) = \frac{p^{(m)}(c)}{m!}(\Delta c)^m + O\left((\Delta c)^{m+1}\right),$$

en bijgevolg is $\Delta p(c + \Delta c) = O\left((\Delta c)^m\right)$. Er geldt opnieuw

$$\Delta p(c + \Delta c) \approx \Delta p(c) + \Delta p'(c)\Delta c \approx \Delta p(c).$$

We besluiten dat

$$(\Delta c)^m = -\frac{\Delta p(c) m!}{p^{(m)}(c)}.$$

Dit is een m-de graadsveelterm in Δc met m oplossingen

$$\Delta c = \left(-\frac{\Delta p(c) m!}{p^{(m)}(c)}\right)^{1/m} e^{i\frac{k2\pi}{m}}, \quad k = 0, \ldots, m-1, \tag{4}$$

die symmetrisch rond 0 liggen in het complexe vlak.

De afschatting (7) ligt voor de hand. Uit deze formule besluiten we het volgende voor de conditie van een meervoudig nulpunt: de conditie van een nulpunt c met meervoudigheid m wordt slechter wanneer

- m verhoogt, door de factor $\epsilon^{1/m}$ in de bovengrens. Stel bvb. dat $\epsilon \approx 10^{-16}$, dan krijg je voor een 2-voudig nulpunt dat $\epsilon^{1/m} \approx 10^{-8}$, terwijl voor een 4-voudig nulpunt $\epsilon^{1/m} \approx 10^{-4}$!
- $p^{(m)}(c)$ verkleint, dus als er andere wortels dichtbij c liggen
- n verhoogt, door de factor met $|c|^n$, indien $|c| > 1$.

P3.

1. $\Delta c \approx -2.5 \cdot 10^{-7}$
2. Omdat $p^{(2)}(1) = 2!$ bekom je voor de foutenschatter $\Delta c \approx (-\Delta t)^{1/2}$, of volgens de meer nauwkeurigere formule (4): $\Delta c \approx (-\Delta t)^{1/2} e^{i(k\pi)} = \pm(-\Delta t)^{1/2} = \pm 0.001i$. Dit is ook precies wat je in de praktijk bekomt, op afrondingsfouten na. Toon zelf aan, dat je in dit geval eigenlijk nergens een benadering hebt moeten gebruiken in het bewijs van de foutenschatter, en dat de foutenschatter de exacte waarde van Δc teruggeeft!

P4. De berekende nulpunten liggen in het complexe vlak, symmetrisch rond het exacte 6-voudige nulpunt $c = 2$, met een fout van ongeveer $|\Delta c| \approx 1e-4$. De achterwaartse fout bekom je m.b.v. het commando `poly` op de berekende nulpunten en is van grootte orde 1e-13. (Als je de relatieve achterwaartse fout berekent, bekom je iets van grootte orde 1e-16, wat aangeeft dat `roots` een achterwaarts stabiel algoritme is.)

Als je de achterwaartse foutveelterm hebt, dan kan je $\Delta p(c)$ berekenen m.b.v. het commando `polyval`. De foutenschatter vereenvoudigt opnieuw tot $(-\Delta p(c))^{1/6} e^{i(k2\pi/6)}$, $k = 0, \ldots, 5$. Je bekomt een benadering nauwkeurig tot op ongeveer twee decimale cijfers.

P5. Uit deze oefening blijkt dat je ook een (heel) slechte conditie kan hebben voor enkelvoudige nulpunten. Als je voor de Wilkinson veelterm de absolute fouten op de berekende nulpunten juist bepaalt, dan zie je dat je de maximale fout bekomt voor het nulpunt $c = 14$ van ongeveer $6.5e-2$.

De absolute achterwaartse fout, berekend als `poly(r)-c` waarbij `r` de berekende nulpunten zijn en `c=poly(1:20)`, is zeer groot voor sommige coëfficiënten. Neem je echter de relatieve fout, dan bekom je iets van grootte orde 1e-15, wat weer aangeeft dat `roots` een achterwaarts stabiel algoritme is.

17 (PC) Het berekenen van eigenwaarden

P1. De matrix A heeft eigenwaarden 100, 1 en 1, wat je ziet met het commando `[V,D] = eig(A)`. Wanneer je de eerste startvector gebruikt, dan bekom je lineaire convergentie, waarbij je de convergentiefactor kan aflezen van de grafiek of kan berekenen als $\rho \approx 0.01$. Dit komt overeen met de theoretische waarde $|\lambda_2|/|\lambda_1|$. (Merk op dat je een goede schaal op de assen moet nemen voor de grafiek.)

Gebruik je de tweede startvector, dan zie je in de grafiek de convergentie pas inzetten in de 9-de stap. Dit komt omdat de tweede startvector een eigenvector van A is bij de eigenwaarde 1 en dus α_1 gelijk is aan 0 in de eerste formule van de opgave. Door afrondingsfouten zal echter de iteratievector ook een component van de eigenvector horende bij eigenwaarde 100 krijgen, wat erop neerkomt dat α_1 een heel kleine waarde krijgt verschillend van nul. Hierdoor zal de eerste term van $A^k X^{(0)}$ pas na een aantal stappen beginnen domineren.

B heeft eigenwaarden 1, 2 en 3. Voor de eerste startvector krijg je lineaire convergentie met $\rho = 2/3$, wat opnieuw overeenkomt met de theorie. De tweede startvector is de eigenvector bij de eigenwaarde 3, waardoor de methode al na 1 stap geconvergeerd is.

P2. Indien de eigenwaarden $\{\lambda_i\}_{i=1}^n$ herordend zijn als $\{\hat{\lambda}_i\}_{i=1}^n$ zodat

$$|\sigma - \hat{\lambda}_1| < |\sigma - \hat{\lambda}_2| \leq |\sigma - \hat{\lambda}_3| \leq \dots$$

dan is bij de methode van de inverse machten de convergentiefactor gelijk aan

$$\rho = \frac{\sigma - \hat{\lambda}_1}{\sigma - \hat{\lambda}_2}.$$

Je krijgt dus lineaire convergentie die des te sneller is wanneer σ dichter bij een bepaalde eigenwaarde ligt. Verklaar zelf deze formule voor ρ!

P3. Als je de iteratiematrix G berekend hebt en de eigenwaarden en eigenvectoren, dan kan je de spectraalradius bekomen als de grootste eigenwaarde in absolute waarde

$$\rho(G) = \max_i \left(|\lambda_i(G)|\right) = 1.33\ldots = \frac{4}{3}.$$

Uit de opgave van een van de vorige oefenzittingen besluit je dat de methode van Jacobi niet convergeert voor elke mogelijke keuze van de startvector. Inderdaad, kijk je naar de formule van $E^{(k)}$

$$E^{(k)} = G^k E^{(0)} = \alpha_1 \lambda_1^k V_1 + \alpha_2 \lambda_2^k V_2 + \ldots + \alpha_n \lambda_n^k V_n,$$

dan zie je dat de eerste term oneindig groot zal worden voor $k \to \infty$, op voorwaarde dat $\alpha_1 \neq 0$. Merk op dat er startvectoren zijn waarvoor $\alpha_1 = 0$. Als doorheen de berekening deze component nul blijft, dan krijg je convergentie als de tweede grootste eigenwaarde (in modulus) voldoet aan $|\lambda_2| < 1$. Merk echter op dat door afrondingsfouten, de eerste component op een bepaald moment verschillend kan worden van 0!

Bekijk je nu beide startvectoren, dan bepaal je de initiële fouten $E^{(0)}$ door de exacte oplossing hiervan af te trekken. Veronderstel dat je deze initiële fouten `e0a` en `e0b` noemt. De convergentie hangt af van de waarde van α_1 in de ontbinding van de fouten als lineaire combinatie van de eigenvectoren van de iteratiematrix G. De coëfficiënten α_i kan je vinden als oplossing van de stelsels

```
V\e0a          V\e0b
```

met als oplossingen, respectievelijk,

$$\begin{bmatrix} 0 \\ 1.3556\ldots \\ -2.0401\ldots \end{bmatrix} \quad \text{en} \quad \begin{bmatrix} 0.0000577\ldots \\ 1.3557\ldots \\ -2.0401\ldots \end{bmatrix}.$$

Voor de eerste startvector krijg je dus $\alpha_1 = 0$, waardoor de convergentie afhangt van de tweede grootste eigenwaarde van G, die in dit geval kleiner is dan 1. De methode van Jacobi convergeert, want $|\lambda_2| < 1$. Voor de tweede startvector krijg je een kleine waarde van α_1, waardoor er divergentie optreedt, maar de term die divergeert begint pas na een bepaald aantal stappen te domineren over de andere, convergerende termen.

P4. In de methode `invmachten_adaptief` zal de matrix $(\sigma_i I - A)^{-1}$ singulier worden, wanneer σ_i een eigenwaarde van A tot op machine-nauwkeurigheid benadert. De waarde van `x0` zal op dat moment de bijhorende eigenvector benaderen tot op machine nauwkeurigheid. In de stappen

```
x1 = (sigma(i)*eye(size(A,1))-A)\x0;
mu = 1/norm(x1);
```

zal `x1` daarom oneindig groot worden (verklaar dit!), waardoor in Matlab de waarde `Inf` wordt toegekend aan elementen van `x1`, en bijgevolg krijgt `mu` de waarde nul. Bekijk de rest van de code. Doordat $\mu = 0$, blijft $\sigma_{i+1} = \sigma_i$ en zal `x0` gelijk worden aan de nulvector. In de volgende stap wordt dan een stelsel opgelost met als matrix een singuliere matrix en als rechterlid de nulvector, wat niet gedefinieerd is en `NaN` (Not a number) waarden geeft in Matlab. Dit is analoog aan de operatie $0/0$.

De convergentie is duidelijk kwadratisch.