KU Leuven

Verzameling

Examenvragen en -antwoorden

Numerieke Wiskunde [G0N90B]

Auteur:
Tom Sydney Kerckhove

Professor:
Professor M. Van Barel

Met dank aan:
Dennis Frett
Karel Domin
Jonas Devlieghere

Gestart: 25 mei 2014
Gecompileerd: 30 mei 2019

Deel 1

Directe Methodes

Vraag 1 Verschillende basis

V 1 Opgave

Gegeven de volgende definitie van e willen we e benaderen.

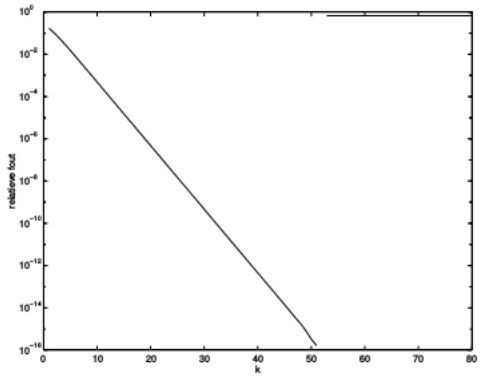
$$\lim_{x\to\infty}\left(1+\frac{1}{x}\right)^x = e$$

We zullen e benaderen op twee manieren door een iteratieproces waarbij de k-de benadering er als volgt uit ziet.

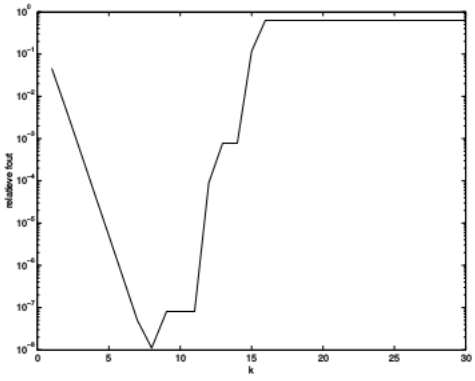
$$e_k = \left(1+\frac{1}{x_k}\right)^{x_k}$$

1. met $x_k = 2^k$
2. met $x_k = 10^k$

We plotten de relatieve fout $\left|\frac{e_k - e}{e}\right|$ in functie van k in figuur 1.1 We bekomen volgende grafieken voor de relatieve fout.



(a) Relatieve fout



(b) Relatieve fout

Figuur 1.1: Relatieve fout

1. Waarom zijn de twee grafieken zo verschillend?
2. Kan men hieruit afleiden met welke basis en met hoeveel beduidende cijfers de computer werkt?
3. Verklaar in detail je antwoord

V 1 Antwoord

Resultaten

In de eerste grafiek zien we dat de relatieve fout afneemt tot k groter wordt dan 52. We zien bovendien dat de relatieve fout niet kleiner wordt dan 10^{-16}. Daarna wordt de relatieve fout plot van grootteorde 10^0.

In de tweede grafiek zien we dat de relatieve fout (enigszins sneller) afneemt tot k groter wordt dan 7, daarna wordt de fout weer groter tot ze ook van grootteorde 10^0 wordt.

Verklaring

De benadering e_k van e wordt in het begin steeds beter omdat de theoretische formule (met de limiet) geldt. Op een bepaald punt worden de afrondingsfouten op x_k echter groter dan de benadering beter wordt. De relatieve fout zal dan opnieuw stijgen.

Vanaf een bepaalde groote van x_k wordt $\frac{1}{x_k}$ zo klein dat dit $1 + \frac{1}{x_k=1}$ geeft. e_k wordt dan ook 1 waardoor de absolute fout $e - 1$ wordt en de relatieve fout bijgevolg van grootteorde 10^0.

Antwoord op de vraag

Op de verschillende x_k worden verschillende fouten gemaakt. Bij de eerste worden er zelfs bijna geen fouten gemaakt op het berekenen van de x_k.

De computer werkt met basis 2 omdat 2^k nagenoeg geen afrondingsfouten geeft. De computer werkt met ongeveer 16 beduidende cijfers, of met 52 bits in de mantisse.

Vraag 2 Het raadsel van 0.3

V 2 Opgave

Gegeven volgend programma met de output. Leg in detail uit waarom het programma besluit dat y niet gelijk is aan 0.3.

```
1 x = 0.1
2 y = 3 * 0.1
3 if y == 0.3
4 then print "y is gelijk aan 0.3"
5 else print "y is niet gelijk aan 0.3"
```

```
1 x = 1.0000000000E-1
2 y = 3.0000000000E-1
3 y is niet gelijk aan 0.3
```

V 2 Informatie

- Boek pagina 13: Hoofdstuk 2 Foutenanalyse

V 2 Antwoord

Er worden in de eerste drie lijnen al drie fouten gemaakt in de machine. In lijn 1 moet de machine 0.1 afronden omdat 0.1 binair niet correct voorgesteld kan worden in eindige precisie.

$$\overline{x} = x(1 + \epsilon_1) \text{ met } |\epsilon_1| \leq \epsilon_{mach}$$

Waneer x wordt afgedrukt wordt x eerst terug omgezet naar het decimaal talstelsel. De fout op x is te klein (kleiner dan de machineprecisie) om een verschil te maken in het decimaal talstelsel in de machine. In de tweede lijn worden twee fouten gemaakt. Eerst wordt dezelfde fout gemaakt als in lijn 1. 0.1 wordt omgezet naar binair. Vervolgens wordt 0.1 met 3 vermenigvuldigd.

$$\overline{y} = 0.3(1 + \epsilon_1)(1 + \epsilon_2) \text{ met } |\epsilon_1|, |\epsilon_2| \leq \epsilon_{mach}$$

In de derde lijn wordt 0.3 omgezet naar binair. Op dit moment wordt er opnieuw een fout gemaakt.

$$\overline{0.3} = 0.3(1 + \epsilon_3) \text{ met } |\epsilon_3| \leq \epsilon_{mach}$$

Tenslotte wordt in de derde lijn $\overline{y}$ met $\overline{0.3}$ vergeleken.

$$0.3(1 + \epsilon_1)(1 + \epsilon_2) - 0.3(1 + \epsilon_3) = \epsilon_1 + \epsilon_2 - \epsilon_3 \neq 0$$

Het verschil tussen $\overline{y}$ en $\overline{0.3}$ is groot genoeg om opgemerkt te worden door de machine. De machine beweert daarom dat y niet gelijk is aan 0.3.

Vraag 3 Programma verschil: Verklaar

V 3 Opgave

Gegeven volgend programma met de output. Verklaar waarom het verschil plots niet meer gelijk is aan 0. Waarvoor staat de nieuwe waarde van het verschil?

```
1 som = 0.0
2 for i = 0.0:0.1:1.0
3    verschil = i - som
4    print(verschil)
5    som = som + 0.1
6 end
```

```
1  verschil = 0
2  verschil = 0
3  verschil = 0
4  verschil = 0
5  verschil = 0
6  verschil = 0
7  verschil = 1.1.. e-16
8  verschil = 1.1.. e-16
9  verschil = 1.1.. e-16
10 verschil = 1.1.. e-16
11 verschil = 1.1.. e-16
```

V 3 Informatie

- Boek pagina 22: 5.3 De machinenauwkeurigheid
- Oefenzitting 2: Probleem 1 tot 4

V 3 Antwoord

We werken op de computer met een getallenvoorstelling in basis $b = 2$ en een mantisse met een eindig aantal bits. Het getal 0.1 kan dus niet exact worden voorgesteld. 0.1 ziet er binair als volgt uit.

`0.0001100110011001100110011001100110011001100110011001100110...`

In de machine moet dit getal dus worden afgerond of afgekapt, zodat er een fout gemaakt wordt. Die fout is klein maar bestaande. Telkens wanneer er 0.1 bij 'som' wordt opgeteld stapelt diezelfde fout zich op. In iteratie j ziet 'som' er wiskundig gezien zo uit.

$$som = j(0.1)(1 + \epsilon) = 0.1j + j\epsilon$$

Hier is bovendien ϵ, in absolute waarde, kleiner dan de machineprecisie ϵ_{mach}. (ϵ blijkt negatief te zijn.) Wanneer de waarde van 'verschil' berekend wordt in iteratie j, blijven er eigenlijk enkel nog de opgestapelde fouten over.

$$verschil = 0.1j - som = 0.1j - 0.1j - j\epsilon = -j\epsilon$$

Wanneer 'verschil' wordt afgedrukt wordt de waarde opnieuw omgezet naar het decimaal talstelsel. Als de opgestapelde fout dan nog kleiner is dan de machineprecisie ϵ_{mach} zal de uitvoer 0 geven. Vanaf een bepaalde iteratie zal de opgestapelde fout $j\epsilon$ echter groter worden dan de machineprecisie ϵ_{mach}. Bij de omzetting naar het decimaal stelsel zal de uitvoer niet meer 0 geven, maar het eerstvolgende getal dat voorgesteld kan worden in de machine.

Op het moment dat de opgestapelde fouten te groot worden, dus wanneer de uitvoer $1.1E-16$ geeft, is 'som' (ongeveer) gelijk aan 0.5. De afstand tussen 0.5 en het volgende voorstelbaar getal in de machine is $eps(0.5) = 1.1E-16$. Dit is precies hetgeen de uitvoer geeft.

Vraag 4 Stabiliteit van een eenvoudig algoritme

V 4 Opgave

Gegeven volgende functie f en volgend algoritme a om f te evalueren in x.

$$f(x) = e^{x^2} - 1 - x^2$$

```
1 a = x^{2}
2 b = e^a
3 f = b-1-a
```

Is dit algoritme numeriek stabiel? Hoe zit het met de conditie van dit probleem?

V 4 Informatie

- Boek pagina 33: 7.3 Numerieke stabiliteit van een methode
- Oefenzitting 2, 3 en 4 : Bewegende kommavoorstelling en foutenanalyse
- Tutorial: Foutenanalyse

V 4 Oplossing

$$f'(x) = 2xe^{x^2} - 2x$$

Conditie

$$\Delta_c y = f'(x)\Delta x = 2x\Delta x\left(e^{x^2} - 1\right)$$

Dit probleem is duideiljk slecht geconditioneerd voor grote waarden van x.

Stabiliteit

1.

$$\overline{y} = fl\left(fl\left(fl\left(e^{fl(x^2)}\right) - 1\right) - x^2\right)$$

$$\overline{y} = \left(\left(\left(e^{(x^2)(1+\epsilon_4)}\right)(1+\epsilon_3) - 1\right)(1+\epsilon_2) - x^2\right)(1+\epsilon_1)$$

- $$\frac{\delta\overline{y}}{\delta\epsilon_1}(\epsilon_1, 0, 0, 0) = e^{x^2} - 1 - x^2$$
- $$\frac{\delta\overline{y}}{\delta\epsilon_2}(0, \epsilon_2, 0, 0) = e^{x^2} - 1$$
- $$\frac{\delta\overline{y}}{\delta\epsilon_3}(0, 0, \epsilon_3, 0) = e^{x^2}$$
- $$\frac{\delta\overline{y}}{\delta\epsilon_4}(0, 0, 0, \epsilon_4) = x^2e^{x^2(1+\epsilon_4)}$$

$$\frac{\delta\overline{y}}{\delta\epsilon_4}(0, 0, 0, 0) = x^2e^{x^2}$$

2.

$$\overline{y} \approx y + \left(e^{x^2} - 1 - x^2\right)\epsilon_1 + \left(e^{x^2} - 1\right)\epsilon_2 + \left(e^{x^2}\right)\epsilon_3 + \left(x^2e^{x^2}\right)\epsilon_4$$

$$\overline{y} \approx y + y\epsilon_1 + \left(e^{x^2} - 1\right)\epsilon_2 + \left(e^{x^2}\right)\epsilon_3 + \left(x^2e^{x^2}\right)\epsilon_4$$

3. De absolute fout:

$$\overline{y} - y \approx \left(e^{x^2} - 1 - x^2\right)\epsilon_1 + \left(e^{x^2} - 1\right)\epsilon_2 + \left(e^{x^2}\right)\epsilon_3 + \left(x^2 e^{x^2}\right)\epsilon_4$$

De relatieve fout:

$$\frac{\overline{y} - y}{y} \approx \epsilon_1 + \frac{e^{x^2} - 1}{e^{x^2} - 1 - x^2}\epsilon_2 + \frac{e^{x^2}}{e^{x^2} - 1 - x^2}\epsilon_3 + \frac{x^2 e^{x^2}}{e^{x^2} - 1 - x^2}\epsilon_4$$

4.

$$\delta_s y = \frac{\overline{y} - y}{y} \leq \epsilon_{mach}\left(1 + \frac{e^{x^2} - 1}{e^{x^2} - 1 - x^2} + \frac{e^{x^2}}{e^{x^2} - 1 - x^2} + \frac{x^2 e^{x^2}}{e^{x^2} - 1 - x^2}\right)$$

$$\delta_s y = \frac{\overline{y} - y}{y} \leq \left(1 + \frac{(x^2 + 2)e^{x^2} - 1}{e^{x^2} - 1 - x^2}\right)\epsilon_{mach}$$

Dit algoritme is dus onstabiel voor grote x en wanneer $e^{x^2} \approx x^2 + 1$ geldt.

Omdat $\Delta_s y$ niet klein is en $\Delta_c y$ en $\Delta_s y$ van dezelfde grootteorde zijn, besluiten we dat het algoritme zwak stabiel is.

Vraag 5 Veeltermen met een zo laag mogelijke graad

V 5 Opgave

Gegeven volgende informatie over een veelterm p.

- $p(-1) = p(0) = p(1)$
- $p'(0) = 1$

V 5 Informatie

- Boek pagina 122: 11.1 De methode der onbepaalde coëfficiënten.

V 5 Antwoord

We zoeken een veeltem van zo laag mogelijke graad. Voor elke graad komt er in de methode der onbepaalde coëfficienten één interpolatievoorwaarde bij. We hebben vier interpolatievoorwaarden gegeven. Dit is misschien niet meteen duidelijk omdat we maar drie gelijkheden gegeven hebben, maar het zijn wel degelijk vier voorwaarden: (c is een onbekende constante.)

$$\begin{cases} p(-1) = c \\ p(0) = c \\ p(1) = c \\ p'(c) = 1 \end{cases}$$

We zoeken dus een veelterm van graad 3.

$$p(x) = a_0 + a_1 x + a_2 x^2 + a_3 x^3$$

We kunnen nu een stelsel opstellen dat overeenkomt met de interpolatievoorwaarden.

$$\begin{pmatrix} 1 & -1 & 1 & -1 \\ 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & -1 \\ 0 & 1 & 0 & 0 \end{pmatrix} \begin{pmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \end{pmatrix} = \begin{pmatrix} c \\ c \\ c \\ 1 \end{pmatrix}$$

De oplossing van dit stelsel rekenen we met de hand uit.

$$\begin{pmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \end{pmatrix} = \begin{pmatrix} c \\ 1 \\ 0 \\ -1 \end{pmatrix}$$

Alle mogelijke veeltermen van de laagst mogelijke graad die aan de gegeven voorwaarden voldoen zijn de volgenden.

$$\{-x^3 + x + c \mid c \in \mathbb{R}\}$$

Vraag 6 Verschillende basis

V 6 Opgave

Gegeven volgende informatie over een functie f. Bepaalde Hermitisch interpolerende veelterm van graad 3.

- $f(x_0)$
- $f(x_1)$
- $f'(x_0)$
- $f'(x_1)$

V 6 Informatie

- Boek pagina 122: 11 Hermite interpolatie

V 6 Antwoord

We kunnen dit oplossen met de methode der onbepaalde coëfficiënten. We leggen vier interpolatievoorwaarden op voor de interpolerende veelterm van graad 3.

$$\begin{cases} a_0 + a_1x_0 + a_2x_0^2 + a_3x_0^3 = f(x_0) \\ a_0 + a_1x_1 + a_2x_1^2 + a_3x_1^3 = f(x_1) \\ a_1 + 2a_2x_0 + 3a_3x_0^2 = f'(x_0) \\ a_1 + 2a_2x_1 + 3a_3x_1^2 = f'(x_1) \end{cases}$$

$$\begin{pmatrix} 1 & x_0 & x_0^2 & x_0^3 \\ 1 & x_0 & x_0^2 & x_0^3 \\ 0 & 1 & 2x_0 & 3x_0^2 \\ 0 & 1 & 2x_0 & 3x_0^2 \end{pmatrix} \begin{pmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \end{pmatrix} = \begin{pmatrix} f(x_0) \\ f(x_1) \\ f'(x_0) \\ f'(x_1) \end{pmatrix}$$

Dit stelsel kunnen we met de hand uitrekenen.

Vraag 7 Interpolatie van een sinus

V 7 Opgave

De sinusfunctie op het interval $]-\pi, \pi[$ wordt geïnterpoleerd door een veelterm p van graad n in $n+1$ interpolatiepunten. Geef een bovengrens voro de interpolatie-fout

V 7 Informatie

- Boek pagina 94: 6 De interpolatiefout

V 7 Antwoord

De interpolatiefout wordt gegeven door volgende formule.

$$E_n(x) = \frac{f^{n-1}(\xi)}{(n+1)!} \prod_{i=0}^{n} (x - x_i)$$

Een bovengrens $|E_{max}|$ voor de interpolatiefout ziet er dan als volgt uit.

$$|E_{max}| = \underset{x \in]-\pi,\pi[}{max} \frac{|\prod_{i=0}^{n}(x - x_i)|}{(n+1)!} \underset{x \in]-\pi,\pi[}{max} |f^{(n+1)}(x)|$$

De tweede factor is maximaal 1 omdat een afgeleide van de sinus steeds een sinus of een cosinus is. De eerste factor kunnen we niet meer vereenvoudigen zonder extra aannames.

Vraag 8 Functiewaarden gegeven, bepaal coëefficient.

V 8 Opgave

Gegeven volgende informatie over een veelterm p van graad 4.

$$p(x) = a_0 + a_1x + a_2x^2 + a_3x^3 + a_4x^4$$

- $p(0) = 0$
- $p(1) = 9$
- $p(2) = 15$
- $p(3) = 18$

Alle gedeelde differenties van de vierde graad zijn bovendien 1. Bereken de waarde van de derde coëfficieënt a_3.

V 8 Informatie

- Boek pagina 103: 9.3 Gedeelde differenties

V 8 Antwoord

Alle gedeelde differenties van de vierde graad zijn 1. Dus ook de gedeelde differentie van de eerste vijf coëfficienten.

$$p[x_0, x_1, x_2, x_3, x_4] = 1$$

De nulde differenties zijn ook gegeven voor vier verschillende waarden van x. $(x_0, \cdots, x_3)$

- $p[0] = 0$
- $p[1] = 9$
- $p[2] = 15$
- $p[3] = 18$

We kunnen nu de gedeelde differenties van de eerste graad berekenen ...

$$p[x, y] = \frac{p(y) - p(x)}{y - x}$$

- $p[0, 1] = 4$
- $p[1, 2] = 6$
- $p[2, 3] = 3$

... alsook de gedeelde differenties van de tweede graad.

- $p[0, 1, 2] = 1$
- $p[1, 2, 3] = -\frac{3}{2}$

Tenslotte berekenen we nog de gedeelde differenties van de derde graad.

$$p[0, 1, 2, 3] = -\frac{5}{6}$$

We kunnen nu p berekenen met interpolatie volgens newton.

$$p(x) = 5 + 4(x - 0) + 1(x - 0)(x - 1) - \frac{5}{6}(x - 0)(x - 1)(x - 2) + 1(x - 0)(x - 1)(x - 2)(x - 3)$$

Werk dit uit om de derde coëfficient a_3 af te lezen.

$$a_3 = -6.833$$

Vraag 9 Functiewaarden gegeven, bepaal coëefficient.

V 9 Opgave

Gegeven de volgende informatie over een functie f.

- $x_0 = 0$
- $f(x_0) = 2$
- $f'(x_1) = 0$
- $f''(x_1) = 0$
- $x_1 = 1$
- $f'(x_0) = 0$
- $f''(x_0) = 0$
- $f[x_0, x_0, x_1, x_1] = 1$

Bereken de waarde van $f(x_1)$.

V 9 Informatie

- Boek pagina 102: 9 Interpolatie volgens Newton
- Boek pagina 122: 11.2 Confluente interpolatiepunten

V 9 Antwoord

Let op: We weten niet of de functie $f(x)$ een veeltermfunctie is of niet! De gedeelde differentie heeft natuurlijk iets te maken met een afgeleide.

$$\lim_{x\to y} f[x,y] = \lim_{x\to y} \frac{f(y)-f(x)}{y-x} = f'(y)$$

Op deze manier zullen we de gegeven waarden voor de afgeleiden kunnen gebruiken. We zullen de gegeven derdegraads gedeelde differentie schrijven als een limiet, om bovenstaande formule te kunnen gebruiken.

$$f[x_0,x_0,x_1,x_1] = \lim_{x_2\to x_0, x_3\to x_1} f[x_2,x_0,x_1,x_3]$$

We zullen nu de gegeven gedeelde differentie van graad 3 uitwerken volgens de definitie, om een vergelijking te bekomen met $f(x_1)$ als onbekende.

$$\lim_{x_2\to x_0, x_3\to x_1} f[x_2,x_0,x_1,x_3] = \lim_{x_2\to x_0, x_3\to x_1} \frac{f[x_2,x_0,x_1]-f[x_0,x_1,x_3]}{x_0-x_1}$$

$$= \frac{\lim_{x_2\to x_0} f[x_2,x_0,x_1] - \lim_{x_3\to x_1} f[x_0,x_1,x_3]}{x_0-x_1} = \frac{\lim_{x_2\to x_0} \frac{f[x_2,x_0]-f[x_0,x_1]}{x_2-x_1} - \lim_{x_3\to x_1} \frac{f[x_0,x_1]-f[x_1,x_3]}{x_0-x_3}}{x_0-x_1}$$

$$= \frac{\frac{\lim_{x_2\to x_0} f[x_2,x_0]-f[x_0,x_1]}{x_0-x_1} - \frac{f[x_0,x_1]-\lim_{x_3\to x_1} f[x_1,x_3]}{x_0-x_1}}{x_0-x_1}$$

$$= \frac{\frac{f'(x_0)-\frac{f(x_0)-f(x_1)}{x_0-x_1}}{x_0-x_1} - \frac{\frac{f(x_0)-f(x_1)}{x_0-x_1}-f'(x_1)}{x_0-x_1}}{x_0-x_1}$$

Van deze uitdrukking weten we dat ze gelijk is aan 1. (gegeven) Vul nu in wat verder nog gegeven is en werk uit om $f(x_1)$ te bekomen.

$$1 = \frac{\frac{f'(x_0)-\frac{f(x_0)-f(x_1)}{x_0-x_1}}{x_0-x_1} - \frac{\frac{f(x_0)-f(x_1)}{x_0-x_1}-f'(x_1)}{x_0-x_1}}{x_0-x_1} = \frac{\frac{0-\frac{2-f(x_1)}{-1}}{-1} - \frac{\frac{2-f(x_1)}{-1}-0}{-1}}{-1}$$

$$1 = 2 - f(x_1) + 2 - f(x_1) = 4 - 2f(x_1)$$

$$f(x_1) = \frac{3}{2}$$

Vraag 10 Voortplantende fout in veelterm interpolatie

V 10 Opgave

Gegeven een tabel $\{x_i, f(x_i)\}$ van waarden x_i en hun afbeelding $f(x_i)$ door een functie f. Er zit een fout in de tabel, op de afbeelding van de k-de waarde. Wanneer je de tabel van (voorwaartse) gedeelde differenties opstelt kan je zien hoe de fout zich propageert. Welk patroon herken je? Je mag ervan uitgaan dat de differenties met een hogere graad naar nul gaan. Hoe kan je vinden op welk punt de fout zat en hoe groot die is?

V 10 Informatie

- Boek pagina 103: 9.3 Gedeelde differenties

V 10 Antwoord

Noem de fout op $f(x_k)$ ϵ. We kunnen nu de tabel der gedeelde differenties opstellen.

$$
\begin{array}{cc}
x_0 & f(0) \\
x_1 & f(1) \\
\vdots & \\
x_{k-2} & f(k-2) \\
 & \qquad f[x_{k-2}, x_{k-1}] \\
x_{k-1} & f(k-1) \qquad\qquad f[x_{k-2}, x_{k-1}, x_k] + \frac{\epsilon}{(x_k - x_{k-1}(x_k - x_{k-2})} \\
 & \quad f[x_{k-1}, x_k] + \frac{\epsilon}{x_k - x_{k-1}} \\
x_k & f(k) + \epsilon \qquad\qquad f[x_{k-1}, x_k, x_{k+1}] - \frac{\frac{\epsilon}{x_k - x_{k-1}} + \frac{\epsilon}{x_{k+1} - x_k}}{x_{k+1} - x_{k-1}} \\
 & \quad f[x_k, x_{k+1}] - \frac{\epsilon}{x_{k+1} - x_k} \\
x_{k+1} & f(k+1) \qquad\qquad f[x_k, x_{k+1}, x_{k+2}] + \frac{\epsilon}{(x_{k+1} - x_k)(x_{k+2} - x_k)} \\
 & \qquad f[x_{k+1}, x_{k+2}] \\
x_{k+2} & f(k+2) \\
\vdots & \\
x_{n-1} & f(n-1) \\
x_n & f(n)
\end{array}
$$

Wanneer we nu deze tabel vereenvoudigen (en veronderstellen dat de punten equidistant zijn met breedte h) zodat we enkel nog de fout in de tabel zetten, dan wordt het veel makkelijker om een patroon op te merken.

$$
\begin{array}{cccc}
\vdots & & & \\
x_{k-3} & 0 & & \\
 & & 0 & \\
x_{k-2} & 0 & & 0 \\
 & & 0 & \qquad \frac{\epsilon}{6h^3} \\
x_{k-1} & 0 & \qquad \frac{\epsilon}{2h^2} & \\
 & & \frac{\epsilon}{h} & \qquad -\frac{3\epsilon}{6h^3} \\
x_k & \epsilon & \qquad -\frac{2\epsilon}{2h^2} & \\
 & & -\frac{\epsilon}{h} & \qquad \frac{3\epsilon}{6h^3} \\
x_{k+1} & 0 & \qquad \frac{\epsilon}{2h^2} & \\
 & & 0 & \qquad -\frac{e}{6h^3} \\
x_{k+2} & 0 & \qquad 0 & \\
 & & 0 & \\
x_{k+3} & 0 & & \\
\vdots & & &
\end{array}
$$

Nu kunnen we wel degelijk een patroon zien. In een gedeelde differentie van de i-de graad zien we een fout van grootteorde $O(\frac{1}{i!h^i})$. Het teken alterneert op de diagonalen in de tabel. De fout wordt bovendien vermenigvuldigd met een factor die overeen komt met een waarde uit de tabel van het binomium van Newton en de driehoek van Pascal.

Om te zien waar de fout gemaakt is in de tabel moeten we inderdaad veronderstellen dat gedeelde differenties van een hogere graad naar nul gaan. In absolute waarde zien we dat de gedeelde differenties symmetrisch zijn rond een bepaalde waarde x_j. De fout zit dan op de functiewaarde $f(x_j)$ van x_j.

Vraag 11 Numerieke differentiatie volgens Lagrange

V 11 Opgave

Zij f een functie die we interpoleren op het interval $]-h,h[$ met de volgende formule.

$$f'(x) = \frac{1}{h^2}\left(\frac{2x-h}{2}f(-h) - 2xf(0) + \frac{2x+h}{2}f(h)\right) + D(x)$$

Geef een uitdrukking voor $D(x)$ en een bovengrens. Waar is de differentiatiefout het grootst?

V 11 Informatie

- Boek pagina 131: Hoofdstuk 5 Numerieke differentiatie

V 11 Antwoord

Dit valt allemaal letterlijk in de cursus op te zoeken. De differentiatiefout wordt gegeven door volgende formule.

$$D_n(x_i) = \frac{\pi'(x_i)}{(n+1)!}f^{(n+1)}(\xi(x_i))$$

We hebben echter te maken met equidistante interpolatiepunten.

$$D_n(x_i) = (-1)^{n-i}\frac{i!(n-i)!}{(n+1)!}h^n f^{(n+1)}(\xi(x_i))$$

Een bovengrens voor de interpolatiefout is dan de volgende.

$$|D_n(x_i)| \leq \frac{|\pi'(x_i)|}{(n+1)!}\max_{x\in]-h,h[}|f^{(n+1)}(x)$$

De differentiatiefout is het grootst in de buurt van de interpolatiepunten.

Vraag 12 Kwadratuurformule met een zo hoog mogelijke nauwkeurigheidsgraad

V 12 Opgave

Bepaal de gewichten H_i van volgende kwadratuurformule zodat ze de hoogst mogelijke nauwkeurigheidsgraad heeft. Welke nauwkeurigheidsgraad is dat?

$$\int_{a-h}^{a+h} f(x)dx = H_{-\frac{1}{2}} f\left(a - \frac{h}{2}\right) + H_0 f(a) + H_{\frac{1}{2}} f\left(a + \frac{h}{2}\right)$$

V 12 Informatie

- Tutorial: Kwadratuurformules
- Boek pagina 139: 1 Interpolerende kwadratuurformules

V 12 Antwoord

Noem d de minimale nauwkeurigheidsgraad. Noem de exacte integraal $I(f)$ en de kwadratuurformule $I_k(f)$. Voor elke veelterm p van een graad kleiner dan d geldt dat de kwadratuurformule de integraal exact benadert. Er bestaat minstens één veelterm van graad $d+1$ die niet exact wordt benaderd door de kwadratuurformule. We kunnen a in de oorsprong leggen zonder verlies van algemeenheid (door de grafiek mee te verschuiven). Er zijn drie gewichten in de kwadratuurformule, dus we leggen een nauwkeurigheidsgraad van $d = 2$ op. We lossen nu volgend stelsel op.

$$\begin{cases} I_k(1) & = H_{-\frac{1}{2}} + H_0 + H_{\frac{1}{2}} & = 2h \\ I_k(x) & = -\frac{h}{2}H_{-\frac{1}{2}} + \frac{h}{2}H_{\frac{1}{2}} & = 0 \\ I_k(x^2) & = \frac{h^2}{4}H_{-\frac{1}{2}} + \frac{h^2}{4}H_{\frac{1}{2}} & = \frac{2h^3}{3} \end{cases}$$

Lossen we nu dit stelsel op, dan krijgen we voor de gewichten de volgende waarden.

$$H_{-\frac{1}{2}} = H_{\frac{1}{2}} = \frac{4h}{3} \text{ en } H_0 = -\frac{2h}{3}$$

Deze kwadratuurformule heeft dus minstens een nauwkeurigheidsgraad van 2.

$$\int_{a-h}^{a+h} f(x)dx = \frac{4h}{3} f\left(a - \frac{h}{2}\right) - \frac{2h}{3} f(a) + \frac{4h}{3} f\left(a + \frac{h}{2}\right)$$

Het blijkt zo te zijn dat de formule veeltermen van graad drie ook nog exact integreert, maar niet alle veeltermen van graad vier. De nauwkeurigheidsgraad van deze formule is dus 3.

Deel 2

Iteratieve Methodes

Vraag 1 Conditie van nulpunten

V 1 Opgave

Gegeven de volgende tweedegraads functie $f(x)$.

$$f(x) = x^2 + 2x + c \text{ met } c < 1$$

Bespreek de conditie van de nulpunten van deze functie $f(x)$.

V 1 Informatie

- Boek pagina 238: 18 De conditite van een wortel
- Boek pagina 239: Figuur 2.11 Conditie van het snijpunt x^*

V 1 Antwoord

Hoe dichter de afgeleide van de functie $f'(x)$ bij nul komt rond x^*, hoe slechter het nulpunt geconditioneerd is. Met andere woorden, hoe vlakker de functie bij het nulpunt, hoe slechter de conditie. We onderzoeken hoe de afgeleide van de functie zich gedraagt bij de nulpunten. Eerst berekenen we de nulpunten.

$$x = \frac{-2 \pm \sqrt{4 - 4c}}{2}$$

Vervolgens berekenen we de afgeleide van $f(x)$.

$$f'(x) = 2x + 2$$

Tenslotte vullen we het nulpunt in in de afgeleide om te zien hoe die zich er gedraagt.

$$f'(x^*) = 2\frac{-2 \pm \sqrt{4 - 4c}}{2} + 2 = -2 \pm \sqrt{4 - 4c} + 2 = \pm 2\sqrt{1 - c}$$

Hier zien we dat de nulpunten slecht geconditioneerd is wanneer c dicht bij 1 ligt. Dit is natuurlijk precies wanneer het nulpunt tweevoudig is.

Vraag 2 Convergentie van een substitutiemethode

V 2 Opgave

Gegeven volgende substutiemethode.

$$x_{k+1} = F(x_k) = (x_k - a)^2 - 1$$

Voor welke reële waarden van x_0 en a treedt er convergentie op? Naar welke waarde voor x convergeert de methode dan? In welke gevallen gebeurt de convergentie quadratisch?

V 2 Informatie

- Boek pagina 221: 12.1 Meetkundige interpretatie van substitutiemethodes
- Boek pagina 229: De convergentiefactor voor een substitutiemethode
- Boek pagina 231: 15.2 Orde van convergentie

V 2 Antwoord

Allereerst berekenen we het vast punt van de iteratiefunctie.

$$x = F(x) \Leftrightarrow x = (x - a)^2 - 1$$

$$\Leftrightarrow 0 = x^2 - (1 + 2a)x + a^2 - 1$$

$$D = (1 + 2a)^2 - 4 \cdot 1 \cdot (a^2 - 1) = 1 + 4a + 4a^2 - 4a^2 + 4 = 4a + 5$$

Als de discriminant D kleiner dan nul is, dit is wanneer $a < -\frac{5}{4}$ geldt, is er geen vast punt. Als de discriminant nul is, wanneer $a = -\frac{5}{4}$ geldt, is er precies één vast punt.

$$x^* = \frac{1 + 2a}{2}$$

Anders zijn er twee vaste punten x_1^* en x_2^*.

$$x_1^* = \frac{2a + 1 + \sqrt{4a + 5}}{2} \text{ en } x_2^* = \frac{2a + 1 - \sqrt{4a + 5}}{2}$$

In wat volgt gaan we er dus van uit dat a groter of gelijk is aan $-\frac{5}{4}$. Om nu de convergentie te berekenen, hebben we de afgeleide van de iteratiefunctie nodig.

$$F'(x) = 2x - 2a$$

Wanneer deze afgeleide in het vast punt, in absolute waarde, groter is dan één is er divergentie. Wanneer ze negatief is is er spiraalconvergentie terwijl een positieve waarde voor de afgeleide in het vast punt op monotone convergentie duidt

$$F'(x_1^*) = 1 + \sqrt{4a + 5} \text{ en } F'(x_2^*) = 1 - \sqrt{4a + 5}$$

- x_1^*

  $$|1 + \sqrt{4a + 5}| > 1 \Leftrightarrow 1 + \sqrt{4a + 5} > 1 \vee 1 + \sqrt{4a + 5} < -1$$

  $$True \vee False \Leftrightarrow True$$

  De substitutiemethode convergeert dus niet voor het eerste vaste punt x_1^*.

- x_2^*

  $$|1 - \sqrt{4a + 5}| > 1 \Leftrightarrow 1 - \sqrt{4a + 5} > 1 \vee 1 - \sqrt{4a + 5} < -1$$

  $$False \vee a > -\frac{1}{4} \Leftrightarrow a > -\frac{1}{4}$$

  De substitutiemethode divergeert dus ook voor het tweede vaste punt, maar enkel als a groter is dan $-\frac{1}{4}$. Wanneer a kleiner is dan $-\frac{1}{4}$ maken we nog een onderscheid tussen spiraalconvergentie en monotone convergentie.

  $$1 - \sqrt{4a + 5} < 0 \wedge a < -\frac{1}{4} \Leftrightarrow a \in]-1, -\frac{1}{4}[$$

  Voor $a \in]-1, -\frac{1}{4}[$ is er dus spiraalconvergentie.

  $$1 - \sqrt{4a + 5} > 0 \wedge a < -\frac{1}{4} \Leftrightarrow a \in]-\frac{5}{4}, -1[$$

  Voor $a \in]-\frac{5}{4}, -1[$ is er monotone convergentie.

De waarde van de afgeleide in het vast punt $F'(x^*)$ is bovendien gelijk aan de convergentiefactor. Wanneer die waarde nul is convergeert de methode kwadratisch ($a = -1$), anders convergeert ze lineair.

Samenvatting

- $a < -\frac{5}{4}$: Geen vaste punten.
- $a = -\frac{5}{4}$: één vast punt $\frac{1+2a}{2}$
- $a < -\frac{5}{4}$: Twee vaste punten.

$$x_1^* = \frac{2a + 1 + \sqrt{4a + 5}}{2} \text{ en } x_2^* = \frac{2a + 1 - \sqrt{4a + 5}}{2}$$

Er is steeds divergentie voor x_1^*. Voor x_2^* is er convergentie wanneer $a < -\frac{1}{4}$ geldt.

- $a \in]-1, -\frac{1}{4}[$: Spiraalconvergentie
- $a = 1$: Kwadratische convergentie
- $a \in]-\frac{5}{4}, -1[$: Monotone convergentie

Vraag 3 Convergentie van een substitutiemethode

V 3 Opgave

Gegeven volgende substutiemethode.

$$x_{k+1} = F(x_k) = (x_k - a)^2 + 1 \text{ met } a > 0$$

Voor welke reële waarden van x_0 en a treedt er convergentie op? Naar welke waarde voor x convergeert de methode dan? In welke gevallen gebeurt de convergentie quadratisch?

V 3 Informatie

- Boek pagina 221: 12.1 Meetkundige interpretatie van substitutiemethodes
- Boek pagina 229: De convergentiefactor voor een substitutiemethode
- Boek pagina 231: 15.2 Orde van convergentie

V 3 Antwoord

Allereerst berekenen we het vast punt van de iteratiefunctie.

$$x = F(x) \Leftrightarrow x = (x - a)^2 + 1$$
$$\Leftrightarrow 0 = x^2 - (1 + 2a)x + a^2 + 1$$
$$D = (1 + 2a)^2 - 4 \cdot 1 \cdot (a^2 + 1) = 1 + 4a + 4a^2 - 4a^2 - 4 = 4a - 3$$

Deze discriminant is positief wanneer $a \geq \frac{3}{4}$ geldt. Wanneer a precies gelijk is aan $\frac{3}{4}$, is er één vast punt.

$$x^* = \frac{1 + 2a}{2}$$

Als a groter is dan $\frac{3}{4}$ zijn er twee vaste punten x_1^* en x_2^*.

$$x_1^* = \frac{2a + 1 + \sqrt{4a - 3}}{2} \text{ en } x_2^* = \frac{2a + 1 - \sqrt{4a - 3}}{2}$$

In wat volgt gaan we er dus van uit dat a groter of gelijk is aan $\frac{3}{4}$. Om nu de convergentie te berekenen, hebben we de afgeleide van de iteratiefunctie nodig.

$$F'(x) = 2x - 2a$$

Wanneer deze afgeleide in het vast punt, in absolute waarde, groter is dan één is er divergentie. Wanneer ze negatief is is er spiraalconvergentie terwijl een positieve waarde voor de afgeleide in het vast punt op monotone convergentie duidt

$$F'(x_1^*) = 1 + \sqrt{4a - 3} \text{ en } F'(x_2^*) = 1 - \sqrt{4a - 3}$$

- x_1^*

$$|1 + \sqrt{4a - 3}| > 1 \Leftrightarrow 1 + \sqrt{4a - 3} > 1 \vee 1 + \sqrt{4a - 3} < -1$$
$$True \vee False \Leftrightarrow True$$

  De substitutiemethode convergeert dus niet voor het eerste vaste punt x_1^*.

- x_2^*

$$|1 - \sqrt{4a - 3}| > 1 \Leftrightarrow 1 - \sqrt{4a - 3} > 1 \vee 1 - \sqrt{4a - 3} < -1$$
$$False \vee a > \frac{7}{4} \Leftrightarrow a > \frac{7}{4}$$

  De substitutiemethode divergeert dus ook voor het tweede vaste punt, maar enkel als a groter is dan $\frac{7}{4}$. Wanneer a kleiner is dan $\frac{7}{4}$ maken we nog een onderscheid tussen spiraalconvergentie en monotone convergentie.

$$1 - \sqrt{4a - 3} < 0 \wedge a < \frac{7}{4} \Leftrightarrow a \in]1, \frac{7}{4}[$$

  Voor $a \in]1, \frac{7}{4}[$ is er dus spiraalconvergentie.

$$1 - \sqrt{4a - 3} > 0 \wedge a < \frac{7}{4} \Leftrightarrow a \in]0, 1[$$

  Voor $a \in]0, 1[$ is er monotone convergentie.

De waarde van de afgeleide in het vast punt $F'(x^*)$ is bovendien gelijk aan de convergentiefactor. Wanneer die waarde nul is convergeert de methode kwadratisch ($a = 1$), anders convergeert ze lineair.

Samenvatting

- $a < \frac{3}{4}$: Geen vaste punten.
- $a = \frac{3}{4}$: één vast punt $\frac{1+2a}{2}$
- $a < \frac{3}{4}$: Twee vaste punten.

$$x_1^* = \frac{2a + 1 + \sqrt{4a - 3}}{2} \text{ en } x_2^* = \frac{2a + 1 - \sqrt{4a - 3}}{2}$$

Er is steeds divergentie voor x_1^*. Voor x_2^* is er convergentie wanneer $a < \frac{7}{4}$ geldt.

- $a \in]1, \frac{7}{4}[$: Spiraalconvergentie
- $a = 1$: Kwadratische convergentie
- $a \in]0, 1[$: Monotone convergentie

Maak zeker ook een tekening, op het examen, van $y = x$ en $y = F(x)$. Begin bij het punt $(x_0, F(x_0))$ en trek een lijn evenwijdig met de horizontale as naar de rechte $y = x$. Trek nu een rechte van het snijpunt waar je bent aangekomen terug naar de grafiek van $y = F(x)$, deze keer evenwijdig met de verticale as. Ga zo door tot je een convergentiepunt bereikt. Het is zo vaak ook makkelijk te zien of er bij een bepaald convergentiepunt divergentie of convergentie optreedt.

Vraag 4 Newton-Raphson vijfdegraads veelterm

V 4 Opgave

Gegeven een Maple worksheet. (Deze is niet bijgevoegd) Er is een vijfdegraadsveelterm $p(x)$ gegeven met nulpunt $x^* = -0.31$. Er wordt Newton-Rapson gebruikt om dat nulpunt te berekenen en je krijgt een logaritmische plot van de fout. Het plot is een parabool, een typisch plot voor kwadratische convergentie.
Verklaar de grafiek van de relatieve fout. Wat is de convergentiesnelheid?

Bijvraag

Het aantal juiste beduidende cijfers verdubbelt bij elke stap. Hoe zie je dat in de grafiek?

V 4 Informatie

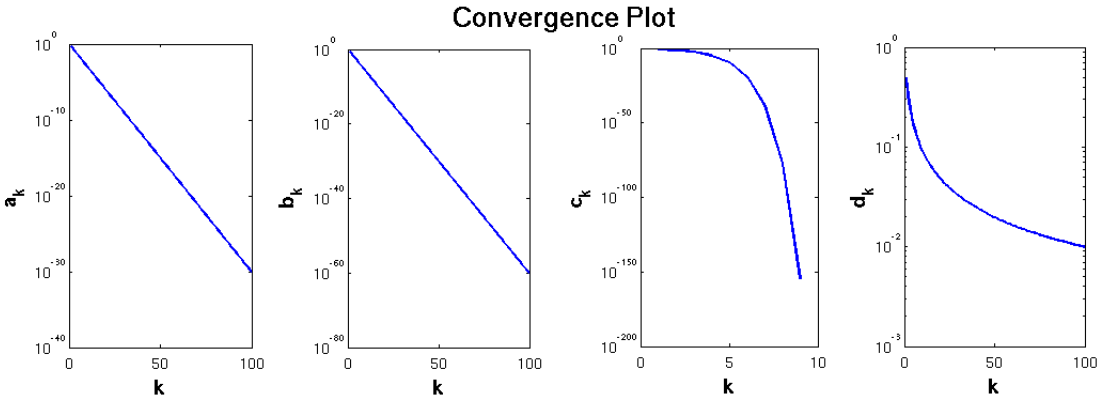
- Boek pagina 209: 6 De methode van Newton-Raphson
- Boek pagina 228: 15 Convergentiesnelheid van rijen
- Boek pagina 233: Tabel 2.9 Convergentiesnelheden

V 4 Antwoord

De convergentiesnelheid van Newton-Raphson is gekend.

- Kwadratisch als x^* een enkelvoudig nulpunt is
- Lineair als x^* een meervoudig nulpunt is met als convergentiefactor $\rho = 1 - \frac{1}{m}$

We kunnen aan de grafiek zien welke convergentiesnelheid de methode met het gegeven nulpunt heeft en daaraan zien wat de multipliciteit van het nulpunt x^* is.



Figuur 2.1: Absolute fout

In figuur 2.1 staan vier plots van absolute fouten. Achtereenvolgens zijn het plots van lineaire, lineaire, quadratische en sublineaire convergentie. Het aantal juiste beduidende cijfers ver q- voudigt in elke iteratiestap, met q gelijk aan de convergentieorde (als de convergentiefactor nul is).

Vraag 5 Newton Raphson: Exacte waarde na één iteratie

V 5 Opgave

Gegeven een Maple worksheet (die niet bijgevoegd zijn). De worksheet toont een uitvoering van de methode van Newton-Raphson. De methode vindt de exacte oplossing (een nulpunt) in dit geval na slechts één iteratiestap.

- Verklaar waarom de methode in dit geval de exacte oplossing vindt na één iteratiestap.
- Wat is de orde van convergentie en de convergentiefactor?
- Verklaar in detail waarom de totale stap vereenvoudigde Newton-Raphson zo traag convergeert.

V 5 Informatie

- Boek pagina 209: 6 De methode van Newton-Raphson
- Boek pagina 233: Tabel 2.9 Convergentiesnelheden
- Boek pagina 261: 3 De Newton-Raphson methode
- Boek pagina 263: 4 Vereenvoudigde Newton-Raphson methodes

V 5 Antwoord

- Newton-Raphson vindt de exacte oplossing in één iteratiestap, ofwel toevallig (als de startwaarde zo gekozen is en de functie het toelaat), ofwel, en dit is hier het geval, is de functie een eerstegraads veelterm. Zij $p(x)$ een eerstegraads veelterm met x^* als nulpunt en $x_0 = c$ de startwaarde voor Newton-Raphson.

  $$p(x) = ax + b \text{ dus } x^* = -\frac{b}{a}$$

  De waarde die Newton-Raphson vindt na één iteratiestap x_1 is precies x^*.

  $$x_1 = F(x_0) = x_c = c - \frac{ac + b}{a} = -\frac{b}{a} = x^*$$

- De convergentiefactor voor Newton-Raphson is 0 met orde 2 als x^* enkelvoudig is en $1 - \frac{1}{m}$ met orde 1 als x^* m-voudig is.
- De vereenvoudigde methodes van Newton-Raphson ruilen rekentijd in voor convergentiesnelheid. De vereenvoudigde methodes convergeren veel trager (in het aantal stappen dat ze nodig hebben), maar de stappen kunnen veel sneller uitgerekend worden.

Vraag 6 Nulpunten met Jacobi

V 6 Opgave

Gegeven een tweedimensionaal lineair stelsel $Ax = b$.

$$A = \begin{pmatrix} \alpha + 1 & 1 \\ \alpha & 1 \end{pmatrix}$$

Bij welke waarden voor α convergeert de methode van Jacobi om nulpunten te vinden?

V 6 Informatie

- Boek pagina 271: 2 Convergentie van methodes

V 6 Antwoord

De methode van Jacobi en Gauss-Seidel convergeren als A *diagonaal dominant* is. Dit betekent dat volgende ongelijkheid geldt.

$$\forall k \in \{1, \cdots, n\} : |a_{kk}| \geq \sum_{j=1, j \neq k} |a_{kj}|$$

In mensentaal betekent dit dat een element op de diagonaal in absolute waarde groter is dan de elementen op dezelfde rij samen. Voor elke rij krijgen we dus een ongelijkheid die moet gelden opdat de methode van Jacobi convergeert.

$$\begin{cases} |\alpha + 1| > 1 \\ 1 > |\alpha| \end{cases}$$

We werken dit stelsel voorzichtig uit. Er zitten namelijk absolute-waardetekens in die wel een last kunnen geven.

$$|\alpha + 1| > 1 \Leftrightarrow \alpha + 1 > 1 \vee \alpha + 1 < -1 \Leftrightarrow \alpha > 0 \vee \alpha < -2$$

$$|\alpha| < 1 \leftrightarrow \alpha < 1 \wedge \alpha > -1$$

Nemen we deze samen, dan krijgen we het volgende interval voor α.

$$(\alpha > 0 \vee \alpha < -2) \wedge (\alpha < 1 \wedge \alpha > -1) \Leftrightarrow \alpha \in]0, 1[\cup]-\infty, -2[$$

Vraag 7 Matrix met dominante eigenwaarde

V 7 Opgave

Gegeven de bijgevoegde Maple worksheet. Verklaar de grafieken. Wat is de orde van convergentie en de convergentiefactor? Wat zou er gebeuren als we in de methode geen normalisatie zouden gebruiken?

V 7 Informatie

- Boek pagina 287: Hoofdstuk 6 Het berekenen van eigenwaarden.
- Boek pagina 290: Opmerking 5 (normalisatie)
- Boek pagina 292: 6 Slotbemerkingen en samenvatting puntje 5 (convergentie- orde en factor)

V 7 Antwoord

In de Maple worksheet wordt de methode van de machten (met normalisatie) uitgevoerd op een matrix A met een beginvector x_0

$$A = \begin{pmatrix} 9 & 1 & -1 \\ 0 & 10 & -5 \\ 0 & 0 & 8 \end{pmatrix} \text{ en } x_0 = \begin{pmatrix} -1 \\ 1 \\ 1 \end{pmatrix}$$

$$x_k = A^k x_0$$

We berekenen eerst de eigenwaarden en eigenvectoren. De eigenwaarden vallen af te lezen op de diagonaal van A. Ze zijn reël en positief.

$$\lambda_1 = 10 \text{ en } E_1 = \begin{pmatrix} 1 \\ 1 \\ 0 \end{pmatrix}$$

$$\lambda_2 = 9 \text{ en } E_2 = \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}$$

$$\lambda_3 = 8 \text{ en } E_3 = \begin{pmatrix} -3 \\ 5 \\ 2 \end{pmatrix}$$

De eigenvectoren vormen een basis, dus x_0 kan uitgedrukt worden als een lineaire combinatie van de eigenvectoren E_i met coëfficienten α_i.

$$x_0 = \alpha_1 E_1 + \alpha_2 E_2 + \alpha_3 E_3$$

We vinden voor deze α_i de volgende waarden.

$$\alpha_1 = -\frac{3}{2},\ \alpha_2 = 2 \text{ en } \alpha_3 = \frac{1}{2}$$

Let op: $\alpha_1 \neq 0$. Dit is een voorwaarde om de methode van de machten te kunnen gebruiken. Controleer deze zeker! We kunnen nu de uitdrukking voor x_k uitwerken.

$$x_k = A^k x_0 = A^k(\alpha_1 E_1 + \alpha_2 E_2 + \alpha_3 E_3) = (\alpha_1 A^k E_1 + \alpha_2 A^k E_2 + \alpha_3 A^k E_3) = \alpha_1 \lambda_1^k E_1 + \alpha_2 \lambda_2^k E_2 + \alpha_3 \lambda_3^k E_3)$$

$$x_k = \lambda_1^k \left(\alpha_1 E_1 + \left(\frac{\lambda_2}{\lambda_1}\right)^k \alpha_2 E_2 + \left(\frac{\lambda_3}{\lambda_1}\right)^k \alpha_3 E_3 \right)$$

In deze uitdrukking is λ_1^k duidelijk dominant. Na voldoende iteraties zal er iets overblijven in x_k van de volgende vorm.

$$x_k = \lambda_1^k \alpha_1 E_1 + o(1)$$

In eerste Maple worksheet, wordt de eigenwaarde iteratief berekend met in de j-de iteratiestap λ_j. Dit zal inderdaad naar λ_1 convergeren.

$$\lambda_{1_j} = \frac{\|x_j\|}{\|x_{j-1}\|}$$

In de andere Maple worksheet zien we de relatieve fout. Deze verkleint zoals verwacht.

Convergentiefactor

De convergentiefactor berekenen we als $\frac{\lambda_2}{\lambda_1}$

$$\rho = \frac{9}{10}$$

We kunnen dit ook uit de grafiek aflezen.

$$\epsilon_{40} = 10^{-3} \text{ en } \epsilon_{100} = 10^{-6}$$

De verhouding van de fout in iteratiestap j ten opzichte van de fout in iteratiestap i is (ongeveer) de $j-i$-de macht van ρ.

$$\rho^{100-40} \approx \frac{10^{-6}}{10^{-3}} = 10^{-3}$$
$$\rho \approx 0.89125$$

Convergentieorde

De convergentiefactor is niet nul, en de fout daalt superlineair, dus de orde van convergentie is 1. De methode van de machten convergeert lineair. Dit staat trouwens letterlijk in het boek op pagina 293.

Zonder normalisatie

Zonder normalisatie zou x_k groeien als $O(n^k)$. Dit zou vroeg of laat overloop met zich meebrengen. Dan valt er niets meer te zeggen over λ_1. Met normalisatie heeft x_k steeds een lengte van 1.

Zorg dat je na 1 uur minstens 1 vraag opgelost hebt en na 2 uur 2 vragen !

Vraag: 4

Methode van de machten

(Dit is de afdruk van een Maple Worksheet)

We rekenen met 16 beduidende decimale cijfers.

```
> restart:with(linalg):lp:=16:Digits:=lp;
```

```
Warning, the protected names norm and trace have been redefined and
unprotected
```

$$Digits := 16$$

We beschouwen de matrix A.

```
> A:=array(1..3,1..3,[[9,1,-1],[0,10,-5],[0,0,8]]);
```

$$A := \begin{bmatrix} 9 & 1 & -1 \\ 0 & 10 & -5 \\ 0 & 0 & 8 \end{bmatrix}$$

Omdat de matrix A bovendriehoeks is, vinden we de eigenwaarden van A op de hoofddiagonaal: 9, 10, 8. De dominante eigenwaarde is dus 10. We gaan deze eigenwaarde proberen te benaderen met behulp van de methode van de machten. We nemen als startvector x0. We zullen K=100 iteraties uitvoeren.

```
> x0:=array(1..3,1..1,[[-1.0],[1.0],[1]]);K:=100;
```

$$x0 := \begin{bmatrix} -1.0 \\ 1.0 \\ 1 \end{bmatrix}$$

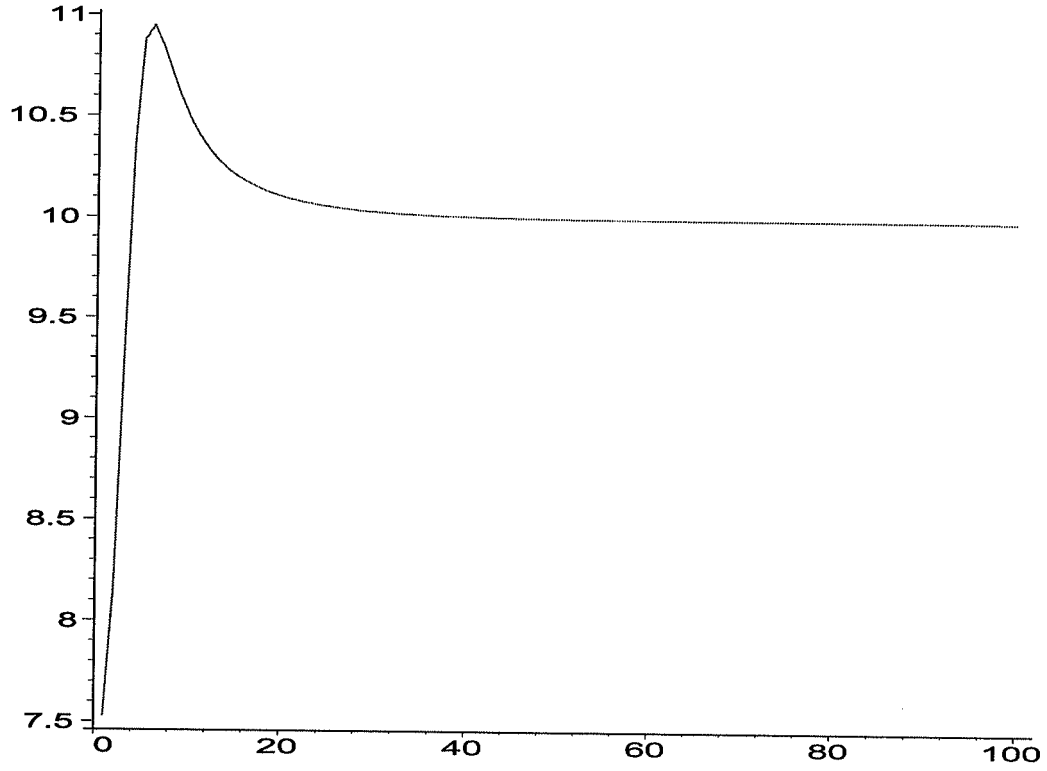
$$K := 100$$

We voeren de methode van de machten met normalisatie uit.

```
> Y:=x0/norm(x0,2):
> muvec:=array(1..K):
> for i from 1 to K do
> Z:=evalm(A&*Y):
> mu:=norm(Z,2);
> Y:=Z/mu:
> muvec[i]:=[i,mu]:
> od:
```

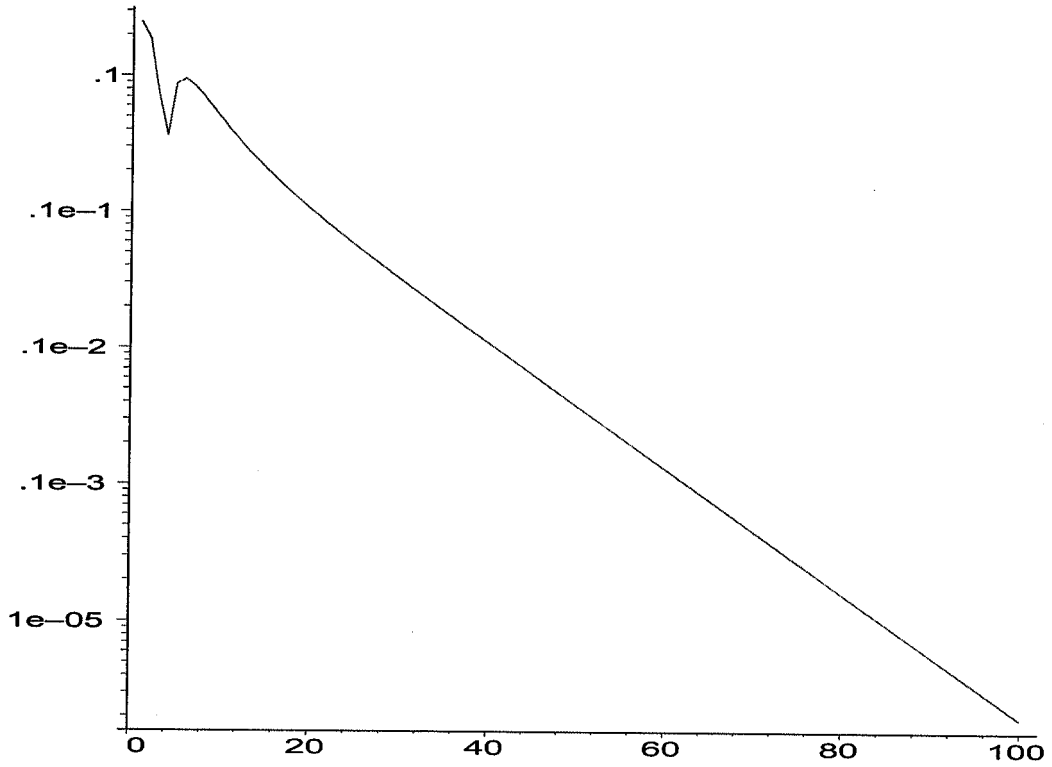
In elke iteratiestap werd er een waarde van mu berekend. Deze waarde wordt hieronder geplot.

```
> with(plots):plot(muvec,thickness=2);
```



We zien dat deze waarde naar 10 konvergeert. In de hiernavolgende grafiek plotten we de relatieve fout.

```
> relerr:=array(1..K):
> for i from 1 to K do
> relerr[i]:=[i,abs(muvec[i][2]-10.0)/10.0];
> od:
> logplot(relerr,thickness=2);
```



Vraag 8 Matrix met dominante eigenwaarde

V 8 Opgave

Gegeven volgende matrix A en startvector x_0.

$$A = \begin{pmatrix} 2 & 1 & -1 \\ 0 & 3 & -5 \\ 0 & 0 & -2 \end{pmatrix} \text{ en } x_0 = \begin{pmatrix} -1 \\ 1 \\ 1 \end{pmatrix}$$

Wat gebeurt als we de methode van 'Von Mises' uitvoeren op de matrix A met startvector x_0.

V 8 Informatie

- Boek pagina 287: Hoofdstuk 6 Het berekenen van eigenwaarden.
- Boek pagina 290: Opmerking 5 (normalisatie)
- Boek pagina 292: 6 Slotbemerkingen en samenvatting puntje 5 (convergentie- orde en factor)

V 8 Antwoord

De methode van 'Von Mises' vermenigvuldigt de vector x_0 herhaalderlijk met A tot er een dominante eigenvector overblijft. De eigenwaarden van A zijn af te lezen op de hoofddiagonaal: 3, 2 en -2. De methode zal convergeren naar de eigenvector die bij 3 hoort.

$$\left(\begin{array}{ccc|c} -1 & 1 & -1 & 0 \\ 0 & 0 & -5 & 0 \\ 0 & 0 & -5 & 0 \end{array}\right) \longrightarrow \left(\begin{array}{ccc|c} -1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \end{array}\right) \longrightarrow E = t \begin{pmatrix} 1 \\ 1 \\ 0 \end{pmatrix}$$

Converentie snelheid

De convergentiefactor ziet er als volgt uit.

$$\frac{\lambda_2}{\lambda_1} = \pm\frac{2}{3}$$

Omdat de convergentiefactor niet nul is, is de convergentie hoogstens lineair. Er kan nog normalisatie gebruikt worden om overloop te voorkomen.

Vraag 9 Methode van de machten

V 9 Opgave

Gegeven volgende matrix A.

$$\begin{pmatrix} a & b \\ c & d \end{pmatrix}$$

- $a = 10^{-5}$
- $b = 10^{-5}$
- $c = 3 - a$
- $d = ab - \frac{2}{b}$

Wat is het condititiegetal van deze matrix? Als we de methode van de machten gebruiken, wat is dan de orde van convergentie, en de convergentiefactor?

V 9 Informatie

- Boek pagina 67: 9.3 Het condititiegetal $\kappa(A)$
- Boek pagina 292: 6 Slotbemerkingen en samenvatting puntje 5

V 9 Antwoord

Condititegetal

Het conditiegetal van een matrix A ziet er als volgt uit.

$$\kappa(A) = \|A\|\|A^{-1}\|$$

Lees $\|A\|$ als 'de norm van A'. We kiezen hiervoor de éénnorm. De éénnorm het maximum van de somaties van de absolute waarden van de elementen per kolom.

$$\|A\|_1 = \max_{1 \leq i \leq n} \sum_{j=1}^{n} |a_{ij}|$$

In dit geval ziet dat er als volgt uit.

$$\|A\|_1 = \max\{|a| + |c|, |b| + |d|\} = |b| + |d| = b - ab + \frac{2}{b} \approx 2 \cdot 10^5$$

Om de norm van de inverse matrix $\|A^{-1}\|$ van A te berekenen moeten we natuurlijk eerst de inverse matrix A^{-1} van A berekenen.

$$A^{-1} = \frac{1}{ad - bc} \begin{pmatrix} d & -b \\ -c & a \end{pmatrix}$$

$$\|A^{-1}\|_1 = \max\left\{ \frac{|d| + |c|}{|ad - bc|}, \frac{|b| + |a|}{|ad - bc|} \right\}$$

$$|ad - bc| = |10^{-15} - 2 - 3 \cdot 10^{-5} + 10^{-10}| \approx 2$$

$$\|A^{-1}\|_1 \approx 10^5$$

Het conditiegetal van A is van grootteorde 10^{10}. Deze matrix A is dus verschrikkelijk geconditioneerd. Hier valt niets aan te doe.

Eigenwaarden

We zetten A in echelonvorm om de eigenwaarden af te lezen op de hoofddiagonaal.

$$\begin{pmatrix} a & b \\ c & d \end{pmatrix} \longrightarrow \begin{pmatrix} a & b \\ 0 & \frac{da}{c} - b \end{pmatrix}$$

De eigenwaarden van A zijn dus de volgende λ_i.

$$\lambda_1 = a = 10^{-5}$$

$$\lambda_2 = \frac{da}{c} - b = \frac{(10^{-5}10^{-5} - \frac{2}{10^{-5}})10^{-5}}{3 - 10^{-5}} - 10^{-5} = \frac{10^{-15} - 2 - 3 \cdot 10^{-5} + 10^{-10}}{3 - 10^{-5}} \approx -\frac{2}{3}$$

De tweede eigenwaarde λ_2 is hier dominant.

Convergentie

De methode van de machten zal convergeren naar de eigenvector E_2 die bij λ_2 hoort. De convergentiefactor ρ ziet er dan als volgt uit.

$$\frac{\lambda_1}{\lambda_2} \approx \frac{-3 \cdot 10^{-5}}{2}$$

De convergentieorde is één.

KU Leuven

Samenvatting van Deel 1 in het boek

Numerieke Wiskunde

Auteur:
Tom Sydney Kerckhove

Professor:
Prof. Dr. A. Bultheel

Gestart: 26 februari 2014
Gecompileerd: 30 mei 2019

Hoofdstuk 1

Inleiding

Fouten:

- Gegevens + voortplantingen
- Stabiliteit
- Afrondingsfouten

Hoofdstuk 2

Foutenanalyse

Definitie 1. *Klassieke voorstelling.*

$$x = \sum_{i=m}^{n} c_i r^i \qquad -\infty \leq m \leq 0 \leq n \leq \infty$$

Grondtal (radix) r, Getallen voor de komma n, Getallen na de komma $|m|$.

Definitie 2. *Bewegende kommavoorstelling.*

$$x = yb^e$$

Mantisse m, Basis b, Exponent e. (b^e is de schaalfactor)

Definitie 3. *Exacte waarde van x: x.*

$$x = \sum_{i=m}^{n} c_i b^i$$

Mantisse m, Basis b, Exponent e.

Definitie 4. *Benadering voor x: $\overline{x}$.*

$$x = \sum_{i=m}^{n} \overline{c_i} b^i$$

Mantisse m, Basis b, Exponent e.

Definitie 5. *Absolute fout Δx.*

$$\Delta x = \overline{x} - x$$

Definitie 6. *Relatieve fout δx.*

$$\delta x = \frac{\overline{x} - x}{x}$$

Definitie 7. *Juist cijfer c_i:*

$$|\overline{x} - x| \leq \frac{1}{2} b^i$$

Definitie 8. *Verband tussen absolute fout en aantal juiste cijfers na de komma.*

$$\frac{1}{2} b^{-p-1} < |\overline{x} - x| \leq \frac{1}{2} b^{-p}$$

Aantal juiste cijfers na de komma p.

Definitie 9. *Verband tussen relatieve fout en aantal juiste cijfers na de komma.*

$$\frac{1}{2}b^{j-k-1} < \frac{|\overline{x}-x|}{\overline{x}} \leq \frac{1}{2}b^{j-k+1} \quad of \quad \frac{1}{2}b^{-q-1} < \frac{|\overline{x}-x|}{\overline{x}} \leq \frac{1}{2}b^{-q+1}$$

Positie van het eerste beduidende cijfer k, Positie van het laatste beduidende cijfer $j+1$, Aantal juiste beduidende cijfers $q = k - j$.

Definitie 10. *Machineprecisie ϵ_{mach}.*

$$\epsilon_{mach} = \frac{1}{2}b^{1-p}$$

Definitie 11. *Het getallenbereik O_{real}: Alle y zodat:*

$$y = mb^r \text{ is voorstelbaar in de machine}$$

Mantisse m, basis b en exponent e.

Definitie 12. *Elementaire bewerking τ.*

$$fl(x \ \tau \ y) = (x \ \tau \ y)(1+\eta)$$

$(|\eta| \leq \epsilon_{mach} \text{ en } x, y \in O_{real})$

Definitie 13. *Absolute fout op een som.*

$$\Delta\left(\sum_{i=1}^{n} x_i\right) = \sum_{i=1}^{n} \Delta x_i$$

Bovengrens:

$$\left|\Delta\left(\sum_{i=1}^{n} x_i\right)\right|_{max} = n\epsilon_+ \leq n\epsilon$$

Relatieve fout op een som.

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

Zie p 27 voor meer uitleg.

Definitie 14. *Zij $\overline{x} = x(1+\delta x)$ en $\overline{y} = x(1+\delta y)$.*
Absolute fout op een vermenigvuldiging.

$$\Delta p = y\Delta x + x\Delta y$$

Relatieve fout op een vermenigvuldiging.

$$\Delta xy = xy(\delta x + \delta y + \delta x \delta y) \approx \delta x + \delta y$$

Bovengrens:

$$|\delta(xy)| \leq 2\epsilon. \text{ en } \left|\delta\left(\frac{x}{y}\right)\right| \leq 2\epsilon.$$

Definitie 15. *Absolute fout op differentieerbare functies.*

$$\Delta f(x) = f(\overline{x}) - f(x) = \Delta x f'(x') \approx f'(\overline{x})\Delta x \text{ met } x' \text{ tussen } x \text{ en } \overline{x}$$

Bovengrens:

$$|\Delta f(x)|_{max} \approx |\Delta x|_{max} \max_t |f'(t)|$$

$$|\Delta f(x_1, ..., x_n)|_{max} \approx \sum_{i=1}^{n} |\Delta x_i|_{max} \max_{t_1,...,t_n} |f_i'(t_1, ..., t_n)|$$

Zie p 30 voor voorbeelden.

Definitie 16. *Norm* $\|\cdot\|$.
De norm van een mathematisch object geeft de 'grootte' ervan aan.

Definitie 17. *Gegevens en resultaten.*
Exact gegeven g.
Gewijzigd gegeven $\overline{g}$.
Absoluut verschil in gegevens Δg.

$$\Delta g = \overline{g} - g$$

Relatief verschil in gegevens δg.

$$\delta g = \frac{\overline{g} - g}{\|g\|} = \frac{\Delta g}{\|g\|}$$

Exact resultaat r.

$$r = F(g)$$

Berekend resultaat $\overline{r}$

$$\overline{r} = F(\overline{g})$$

Absoluut verschil in resultaat Δr

$$\Delta r = F(\overline{g}) - r = \overline{r} - r$$

Relatief verschil in resultaat δr

$$\delta r = \frac{F(\overline{g}) - r}{\|r\|} = \frac{\Delta r}{\|r\|}$$

Definitie 18. *Conditie van een probleem.*
Absoluut Conditiegetal k_A.

$$k_A = \lim_{\epsilon \to 0} \sup_{\|\Delta g\| \leq \epsilon} \frac{\|\Delta r\|}{\|\Delta g\|}$$

Relatief Conditiegetal k_R

$$k_R = \lim_{\epsilon \to 0} \sup_{\|\Delta g\| \leq \epsilon} \frac{\|\delta r\|}{\|\delta g\|}$$

Het conditiegetal van een probleem geeft aan hoeveel een fout op de gegevens wordt opgeblazen in de resultaten.

Voorbeeld: Als het relatieve conditiegetal van een probleem F a is, dan worden fouten op de gegevens met een factor a opgeblazen in het slechtste geval.

Definitie 19. *Benaderingen.*

- *exacte gegevens:* g
- *exact resultaat:* r
- *exact verband:* F
- *Het exact verband tussen het exact resultaat* r *van exact* F *toegepast op exacte gegevens* g.

$$r = F(g)$$

- *Inexacte gegevens:* $\overline{g} = g + \Delta g = g(1 + \delta g)$

- *De exacte berekening van het resultaat $\overline{r}$(, met geïnduceerde fout Δr, δr,) gebaseerd op een gewijzigd gegeven $\overline{g}$.*

$$r + \Delta r = F(g + \Delta g) \rightarrow r(1 + \delta r) = F(g(1 + \delta g)) \rightarrow \overline{r} = F(\overline{g})$$

- *Benaderd resultaat bij eindige discretisatie $\widetilde{r}$*
- *Eindige discretisatie van F, exact berekend: $\widetilde{F}$. Discretisatiefout $err_{dis} = \widetilde{F} - F$.*

$$\widetilde{F} = F + err_{dis}$$

$$\widetilde{r} = \widetilde{F}(\overline{g})$$

  $\widetilde{F}$ verschilt dus enkel van F in de discretisatie fouten.

- *Resultaat bij een echte berekening $\overline{\widetilde{r}}$ bij een berekening gebaseerd op inexacte gegevens $\overline{g}$.*
- *Implementatie van $\widetilde{F}$ in eindige precisie.*

$$\overline{\widetilde{r}} = \overline{F}(\overline{g})$$

  De waarde van de gegevens als $\overline{\widetilde{r}}$ een exacte berekening zou zijn: $\overline{\overline{g}}$.

$$\overline{F}(\overline{g}) = \overline{\widetilde{r}} = \widetilde{F}(\overline{\overline{g}})$$

- *Het resultaat van het exact verband F toegepast op inexacte gegevens $\overline{g}$: $\overline{\overline{r}}$*

Definitie 20. *Stabiliteit van een methode*

- *Voorwaartse/Sterke Stabiliteit.*
  Absolute voorwaartse stabiliteit: De absolute grootte van het verschil tussen het berekende resultaat van de methode toegepast op een gewijzigd resultaat $\overline{g}$ in eindige precisie: $\overline{F}(\overline{g})$ en het exacte resultaat van de methode toegepast op een gewijzigd resultaat $\overline{g}$: $\widetilde{F}(\overline{g})$.

$$\|\overline{F}(\overline{g}) - \widetilde{F}(\overline{g})\| = \|\overline{\widetilde{r}} - \widetilde{r}\|$$

  Relatieve voorwaartse stabiliteit: De relatieve grootte van het verschil tussen het berekende resultaat van de methode toegepast op een gewijzigd resultaat $\overline{g}$ in eindige precisie: $\overline{F}(\overline{g})$ en het exacte resultaat van de methode toegepast op een gewijzigd resultaat $\overline{g}$: $\widetilde{F}(\overline{g})$.

$$\frac{\|\overline{F}(\overline{g}) - \widetilde{F}(\overline{g})\|}{\|\widetilde{F}(\overline{g})\|} = \frac{\|\overline{\widetilde{r}} - \widetilde{r}\|}{\|\widetilde{r}\|}$$

  Kleine waarde(n) $\Rightarrow$ Voorwaarts stabiele methode $\widetilde{F}$.

- *Achterwaartse Stabiliteit.*
  Absolute achterwaartse stabiliteit: De grootte van het verschil tussen de gegevens zoals ze zouden zijn als $\overline{\widetilde{r}}$ een exact resultaat was $\overline{\overline{g}}$ en de echte (gewijzigde) gegevens $\overline{g}$.

$$\|\overline{\overline{g}} - \overline{g}\|$$

  Relatieve achterwaartse stabiliteit: De relatieve grootte van het verschil tussen de gegevens zoals ze zouden zijn als $\overline{\widetilde{r}}$ een exact resultaat was $\overline{\overline{g}}$ en de echte (gewijzigde) gegevens $\overline{g}$.

$$\frac{\|\overline{\overline{g}} - \overline{g}\|}{\|\overline{g}\|}$$

  Relatieve waarde is $O(\epsilon_{mach}) \Rightarrow$ Achterwaarts stabiele methode.

- *Zwakke Stabiliteit.*
  Het de grootte van het verschil tussen $\overline{\overline{r}}$ en $\widetilde{r}$. ten opzichte van de grootte van het verschil tussen $\overline{\widetilde{r}}$ en $\widetilde{r}$: S.

$$S = \frac{\|F(\overline{g}) - \widetilde{F}(\overline{g})\|}{\|\overline{F}(\overline{g}) - \widetilde{F}(\overline{g})\|} = \frac{\|\overline{\overline{r}} - \widetilde{r}\|}{\|\overline{\widetilde{r}} - \widetilde{r}\|}$$

  $S \approx 1 \Rightarrow$ *zwak stabiele methode.*

Merk op: dit zou niet zo specifiek gevraagd mogen worden, de tekst is niet duidelijk genoeg.

Hoofdstuk 3

Stelsels Lineaire Vergelijkingen

3.1 Algoritmes voor oplossen

3.1.1 Achterwaartse substitutie $O(n^2)$

Input

Een bovendriehoekig stelsel $A \in \mathbb{R}^{n \times n}$ en een vector $b \in \mathbb{B}^n$.

$$A = \begin{pmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ 0 & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & a_{nn} \end{pmatrix} \quad b = \begin{pmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{pmatrix}$$

Output

Een vector X, als oplossing voor het stelsel $AX = b$.

3.1.2 Voorwaartse substitutie $O(n^2)$

Input

Een beneden driehoekig stelsel $A \in \mathbb{R}^{n \times n}$ en een vector $b \in \mathbb{B}^n$.

$$A = \begin{pmatrix} a_{11} & 0 & \cdots & 0 \\ a_{21} & a_{22} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nn} \end{pmatrix} \quad b = \begin{pmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{pmatrix}$$

Output

Een vector X, als oplossing voor het stelsel $AX = b$.

3.1.3 Eliminatie $O(n^3)$

Input

Een matrix $A \in \mathbb{R}^{n \times m}$ met $m \geq n$.

Output

Een boventrapeziummatrix C. Merk op dat c_{ij} de volgende waarde heeft.

$$c_{ij} = c_{ji} - \sum_{l=i+1}^{m} c_{jl}\frac{c_{ji}}{c_{ii}}$$

3.1.4 Eenvoudige Gauseliminatie $O(n^3)$

Input

Een matrix $A \in \mathbb{R}^{n\times m}$ met $m > n$ en een vector $b \in \mathbb{B}^n$.
We beschouwen de matrix $C = [A|b]$.

Output

Een vector X, als oplossing voor het stelsel $AX = b$.

Procedure

Voer Eliminatie uit op de uitgebreide matrix C. Voer daarna voorwaartse substitutie uit op het resultaat om de vector X te bekomen.

3.1.5 Gaus-Jordan $O(n^3)$

Input

Een matrix $A \in \mathbb{R}^{n\times m}$ met $m > n$ en een vector $b \in \mathbb{B}^n$.
We beschouwen de matrix $C = [A|b]$.

Output

Een vector X, als oplossing voor het stelsel $AX = b$.

Procedure

Voer de eliminatie en substitutie tegelijk uit. Ga zo kolom per kolom af tot de originele A in een (pseudo-)eenheidsmatrix is veranderd.

3.1.6 Gauss met pivotering $O(n^3)$

Input

Een matrix $A \in \mathbb{R}^{n\times m}$ met $m > n$ en een vector $b \in \mathbb{B}^n$.
We beschouwen de matrix $C = [A|b]$.

Output

Een vector X, als oplossing voor het stelsel $AX = b$.

Procedure

Voer de methode van Gauss uit, maar verwissel rijen zodat het spilelement niet nul is.

3.1.7 Determinant berekenen $O(n^3)$

Input

Een vierkante matrix $A \in \mathbb{R}^{n \times n}$.

Output

De determinant van A.

Procedure

Voer de eliminatiemethode uit op A, en bereken het product van de spilelement.

3.1.8 Gauseliminatie met pivotering en factorizatie $O(n^3)$

Input

Een matrix $C = [A|B|l]$ met $l = (1\ 2\ \cdots\ n)$:

$$C = \left(\begin{array}{c|c|c} & & 1 \\ A & B & \vdots \\ & & n \end{array}\right)$$

Output

Een benedendriehoeksmatrix L, een boventrapeziummatrix R en een vector Y zodat $AX = B$ equivalent is met $RX = Y$. Ook nog een matrix P zodat $PA = LR$ en $PB = LY$.

3.1.9 Gauseliminatie met optimale pivotering $O(n^4)$

Als spilelement kiezen we elke keer het grootste element op die kolom en verwisselen dan die rijen. Zo wordt de stabiliteit van dit algoritme verhoogt.

3.1.10 Methode van Crout

Input

Een matrix $A \in \mathbb{R}^{n \times m}$ met $m > n$ en een vector $b \in \mathbb{B}^n$.

Output

Procedure

Bereken een LR ontbinding door middel van eliminatie.

$$C = LR$$

$$l_{ij} = c_{ij} - \sum_{k=1}^{j-1} l_{ik} r_{kj}$$

$$r_{ij} = \frac{c_{ij} - \sum_{k=1}^{i-1} l_{ik} r_{kj}}{l_{ii}}$$

Los daarna $RX = Y$ op via achterwaartse substitutie.

Samenvatting

Methode	Complexiteit
Voorwaartse substitutie	$O(n^2)$
Achterwaartse substitutie	$O(n^2)$
Eliminatie	$O(n^3)$
Eenvoudige Gauseliminatie	$O(n^3)$
Gaus-Jordan	$O(n^3)$.
Methode van Cramer	$O(n^4)$
Gauseliminatie met pivotering	$O(n^3)$
Determinant berekenen	$O(n^3)$
Gauseliminatie met pivotering en factorizatie	$O(n^3)$
Oplossen van meerdere stelsels met	$O(n)$
dezelfde coëfficienten na factorisatie	Per vector
Gauseliminatie met optimale pivotering	$O(n^4)$

3.2 Conditie van een stelsel lineaire vergelijkingen

Norm van een matrix A: $\|A\|$

$$\|A\| = \underset{X\neq 0}{max}\frac{\|AX\|}{\|X\|}$$

Het conditiegetal van een matrix A: $\kappa(A)$

$$\kappa(A) = \|A\|\|A^{-1}\|$$

Residu van een stelsel lineaire vergelijkingen $AX = B$: R

$$R = AX - B$$

Onthoudt:

$$\|A\|\|A^{-1}\|$$

Bovenstaande uitdrukking is het conditiegetal van een matrix A.

3.3 Nullen Creëren in een matrix

3.3.1 Givens Transformatie

$$G = \begin{pmatrix} c & s \\ s & -c \end{pmatrix} = \begin{pmatrix} \cos\theta & \sin\theta \\ \sin\theta & -\cos\theta \end{pmatrix}$$

Zoek y en y zodat:

$$\begin{pmatrix} c & s \\ s & -c \end{pmatrix}\begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} z \\ 0 \end{pmatrix}$$

Oplossing:

$$\begin{cases} c &= \frac{x}{\pm\sqrt{x^2+y^2}} \\ s &= \frac{y}{\pm\sqrt{x^2+y^2}} \end{cases}$$

3.3.2 Householder Transformatie

$$P = \mathbb{I} - \frac{2vv^T}{v^Tv}$$

3.3.3 QR-factorisatie van een matrix

Gegeven een matrix $A \in \mathbb{R}^{m \times n}$ zoek Q en R zodat.

$$A = Q \cdot R \text{ en } Q^T Q = I$$

met givens en householder transformaties.

Hoofdstuk 4

Veeltermінterpolatie

Definitie 21. *Gegeven $n+1$ punten $(x_i, f(x_i))$ is de veelterm van graad n: $y_n(x)$ de interpolerende veelterm.*

$$y_n(x) = a_0 + a_1 x + \cdots + a_n x^n = \sum_{i=0}^{n} a_i x^i$$

Voor y_n geldt:

$$y_n(x_i) = f_i$$

Het vinden van de waarde van de interpolerende veelterm kan via de coëfficienten en een evaluatie of rechtstreeks.

4.1 Interpolatie volgens Gauss

Los het Vandermonde-stelsel op dat de interpolatievoorwaarden bepalen.

$$\begin{pmatrix} 1 & x_0 & x_0^2 & \cdots & x_0^n \\ 1 & x_1 & x_2^2 & \cdots & x_1^n \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & x_n & x_n^2 & \cdots & x_n^n \end{pmatrix} \begin{pmatrix} a_0 \\ a_1 \\ \vdots \\ a_n \end{pmatrix} = \begin{pmatrix} f(x_0) \\ f(x_1) \\ \vdots \\ f(x_n) \end{pmatrix}$$

Dit kan in $O(n^3)$ tijd het coëfficiëntenprobleem oplossen. We zullen deze methode enkel gebruiken om te tonen dat de interpolerende veelterm uniek is.

4.2 Conditie en stabiliteit

De conditie van beide problemen kan willekeurig slecht zijn. Wanneer het proces convergeert(, dus wanneer de interpolatiefout naar nul gaat,)zal de afrondingsfout stijgen. We zoeken dus een optimale graad n voor interpolatie zodat de totale fout minimaal is.

We zien dat de interpolatiefout kleiner is in het middel van het interpolatieinterval en groter aan de uiteinden.

4.3 Interpolatie volgens Lagrange (coëfficienten)

$$y_n(x) = l_0(x)f(x_0) + l_1(x)f_1(x_1) + \cdots + l_n(x)f(x_n) = \sum_{i=0}^{n} l_i(x)f(x_i)$$

Hierbij is l_0 de lagrange veelterm van graad n.

$$l_i = \prod_{j=0, j\neq i}^{n} \frac{x - x_j}{x_i - x_j} = \frac{\pi(x)}{\pi'(x_i)(x - x_i)}$$

Deze veelterm kan berekend worden in $O(n^2)$ tijd, daarna kan ze geëvalueerd worden in $O(n)$ tijd. Deze vorm van interpolatie is geschikt als oplossing voor het coëfficientenprobleem, maar niet voor het waardeprobleem.

4.4 Interpolatie volgens Newton (beide)

Om het waardenprobleem op te lossen willen we een methode die $y_{n+1}(x)$ makkelijk kan berekenen uit $y_n(x)$.

$$y_n(x) = \sum_{i=0}^{n} f[x_0, x_1, ..., x_i] \prod_{j=0}^{i-1} (x - x_j)$$

Het waardeprobleem kan hiermee opgelost worden in $O(n)$ tijd.

4.4.1 Gedeelde differenties

Definitie 22. *Een gedeelde differentie is een waarde die de verandering van een functie voorstelt per afstand tussen opeenvolgende interpolatiepunten.*

$$f[x_i] = f(x_i)$$

$$f[x_i, x_{i+1}, ..., x_j] = \frac{f[x_{i+1}, ..., x_j] - f[x_i, ..., x_{j-1}]}{x_j - x_i}$$

We kunnen deze gedeelde differenties berekenen door middel van de tabel der gedeelde differenties. De alternatieve tabel der gedeelde differenties dient om makkelijk de graad te kunnen verhogen.

Hoofdstuk 5

Numerieke differentiatie

5.1 Het principe

Gegeven een functie in tabelvorm $\{(x_i, f(x_i))\}_{i=0}^{n}$ waarvan we de afgeleide $f'(x)$ willen berekenen. Zoek de interpolerende veelterm $y_n(x)$ en leidt deze af om een benadering te vinden voor de afgeleide van f.

5.2 Conditie en stabiliteit

Differentiatie is een slecht geconditioneerd probleem. De conditie kan willekeurig slecht zijn.

De stabiliteit van deze methode is zeer slecht omdat er een verschrikkelijk instabiele stap in voorkomt. De differentiatiefout is het grootst bij de interpolatiepunten en het kleinst in het midden van de intervallen.

Hoofdstuk 6

Numerieke Integratie

6.1 Het principe

Gegeven een functie in tabelvorm $\{(x_i, f(x_i))\}_{i=0}^{n}$ waarvan we een integraal $\int_a^b f(x)$ willen berekenen. Zoek de interpolerende veelterm $y_n(x)$ en integreer deze over $[a, b]$ om een benadering te vinden voor de integraal van f.

6.2 Conditie en stabiliteit

Integratie is een relatief goed geconditioneerd probleem. De stabiliteit van deze methode relatief goed, zeker als we ervoor zorgen dat we nergens verschillen berekenen (en dat is mogelijk). De stabiliteit wordt bepaald door de discretisatiefout F_n en de afrondingsfouten bij het evalueren van $\Sigma H_i f(x_i)$

6.3 Newton-Cotes

Niet echt nuttig voor grote n.

Tutorial: Foutenanalyse

Tom Sydney Kerckhove

Started: 27 februari 2014
Compiled: 30 mei 2019

Inhoudsopgave

1 Voorkennis

- Calculus

2 Theorie

2.1 Absolute en relatieve fout

De absolute fout is gedefinieerd als het verschil tussen de berekende waarde en de echte waarde.

$$err_{abs} = \Delta y = \overline{y} - y$$

De relatieve fout is de absolute fout relatief ten opzichte van het echte resultaat.

$$err_{rel} = \frac{\Delta y}{y} = \frac{\overline{y} - y}{y}$$

2.2 $\overline{y}$ exact berekenen

Bij elke bewerking maakt de machine een afrondingsfout, begrensd door de machine precisie.

$$\epsilon_i \leq \epsilon_{mach}$$

Zij $\circ$ een elementaire bewerking, dan geldt het volgende.

$$fl(a \circ b) = (a \circ b)(1 + \epsilon_\circ)$$

2.3 Maclaurin reeks

Een Maclaurin reeks rond is een reeks die een functie benadert in de buurt van 0. Zij $f(x)$ de te benaderen functie met x een variabele.

$$f(x) = \sum_{i=0}^{\infty} \frac{f^{(i)}(0)}{i!} x^i = f(0) + f'(0)x + \frac{1}{2} f''(0)x^2 + ...$$

Of nog algemener, met n variabelen $x_1, ..., x_n$.

$$f(x_1, ..., x_n) = \sum_{i=0}^{\infty} \sum_{j=0}^{n} \frac{x_j^i}{i!} \frac{\delta f}{\delta x_j}(0, ..., 0)$$

2.4 $\overline{y}$ benaderen met een Maclaurin reeks

Eens we $\overline{y}$ exact berekent hebben kunnen we $\overline{y}$ (vrij goed) benaderen zodat de vergelijking eenvoudiger wordt. Zij n het aantal berekeningen om $\overline{y}$ te bekomen. $\overline{y}$ is dus een functie in $\epsilon_1, ..., \epsilon_n$ met x als parameter. We gebruiken een Maclaurin reeks om $\overline{y}$ te benaderen.

$$\overline{y} = y + \sum_{i=1}^{n} \frac{\delta \overline{y}}{\delta \epsilon_i}(0, ..., 0)\epsilon_i$$

We laten de hogere orde termen weg, omdat een term met ϵ_i^2 een verwaarloosbare invloed heeft.

3 Algoritme

Gegeven een getal y dat we berekenen met een machine zodat we $\overline{y}$ bekomen. We berekenen nu de absolute en relatieve fout op deze berekening.

1. Bereken $\overline{y}$ exact.

- Duid de bewerkingen aan.
- Zet rond elke bewerking $fl(...)$.
- Voeg aan het resultaat van elke bewerking $(1+\epsilon_i)$ toe.

2. Bepaal de waarde van de partiële afgeleide van $\overline{y}$ naar elke ϵ_i en evalueer deze in $(0,...,0)$. Hint: de variabelen naar dewelke u niet afleidt kan u meteen invullen.

3. Benader $\overline{y}$ met een Maclaurin reeks.

4. Benader err_{abs} en err_{rel}.

4 Voorbeeld

$$y = \frac{1-\cos(x)}{x^2}$$

1.
$$\overline{y} = fl\left(\frac{fl(1-fl(\cos(x)))}{fl(x^2)}\right)$$
$$\overline{y} = \frac{(1-(\cos(x))(1+\epsilon_1))(1+\epsilon_2)}{(x^2)(1+\epsilon_3)}(1+\epsilon_4)$$

2.
$$\frac{\delta\overline{y}}{\delta\epsilon_1}(\epsilon_1,0,0,0) = \frac{\delta}{\delta\epsilon_1}\frac{1-(\cos(x))(1+\epsilon_1)}{x^2} = -\frac{\cos(x)}{x^2}$$
$$\frac{\delta\overline{y}}{\delta\epsilon_2}(0,\epsilon_2,0,0) = \frac{\delta}{\delta\epsilon_2}\frac{(1-\cos(x))(1+\epsilon_2)}{x^2} = \frac{(1-\cos(x))}{x^2}$$
$$\frac{\delta\overline{y}}{\delta\epsilon_3}(0,0,\epsilon_3,0) = \frac{\delta}{\delta\epsilon_3}\frac{1-\cos(x)}{(x^2)(1+\epsilon_3)} = \frac{1-\cos(x)}{(x^2)}\frac{-1}{(1+\epsilon_3)^2}$$
$$\frac{\delta\overline{y}}{\delta\epsilon_3}(0,0,0,0) = -\frac{1-\cos(x)}{x^2}$$
$$\frac{\delta\overline{y}}{\delta\epsilon_4}(0,0,0,\epsilon_4) = \frac{\delta}{\delta\epsilon_4}\frac{(1-\cos(x))(1+\epsilon_4)}{x^2} = \frac{1-\cos(x)}{x^2}$$

3.
$$\overline{y} \approx y - \frac{\cos(x)}{x^2}\epsilon_1 + \frac{(1-\cos(x))}{x^2}\epsilon_2 - \frac{1-\cos(x)}{x^2}\epsilon_3 + \frac{1-\cos(x)}{x^2}\epsilon_4$$
$$\overline{y} \approx y - \frac{\cos(x)}{x^2}\epsilon_1 + y\epsilon_2 - y\epsilon_3 + y\epsilon_4$$

4.
$$\overline{y} - y \approx -\frac{\cos(x)}{x^2}\epsilon_1 + y\epsilon_2 - y\epsilon_3 + y\epsilon_4$$
$$\frac{\overline{y}-y}{y} \approx -\frac{\cos(x)}{x(1-\cos(x))}\epsilon_1 + \epsilon_2 - \epsilon_3 + \epsilon_4$$

Tutorial: Conditie en Stabiliteit

Tom Sydney Kerckhove

Started: 9 maart 2014
Compiled: 30 mei 2019

Inhoudsopgave

1 Voorkennis

- Calculus
- Foutenanalyse

2 Theorie

2.1 Conditie van een probleem

De conditie van een probleem is een maat voor hoe gevoelig de oplossing is aan veranderingen in de gegevens.

$$\delta_c y = \frac{y'x}{y}\delta x$$

De conditie van een probleem is inherent. Als een probleem slecht geconditioneerd is valt daar niets aan te doen.

2.2 Stabiliteit van een algoritme

De stabiliteit van een algoritme is een maat voor hoe hard fouten in de gegevens worden opgeblazen doorheen het algoritme.

$$\delta_s y = \frac{\overline{y} - y}{y}$$

2.3 Conclusie

- Instabiel:
  $\delta_s y$ is groot, maar $\delta_c y$ is klein. Het probleem is goed geconditioneerd, maar toch kunnen er relatief grote fouten zijn in de oplossing.
- Zwak stabiel:
  $\delta_s y$ en $\delta_c y$ zijn ongeveer even groot.
- Voorwaarts Stabiel: $\delta_s y$ is klein (ten opzichte van de machine precisie).

3 Onderzoek

3.1 Algoritme

3.1.1 Gegeven

y en een algoritme om $\overline{y}$ te berekenen.

3.1.2 Stappenplan

1. Conditie: $\delta_c y$
   (a) Leidt y af, bereken y'.
   (b) Bereken $\delta_c y$
   $$\delta_c y = \frac{y'x}{y}\delta x$$
   (c) Bepaal eventueel de waarden voor x waarvoor $\delta_c y$ groot kan worden.
2. Stabiliteit: $\delta_s y$
   (a) Maak een foute analyse. (Zie Tutorial)

(b) Bereken de relatieve fout

(c) Bepaal eventueel de waarden voor x waarvoor $\delta_s y$ groot kan worden

3. Conclusie

3.2 Voorbeeld

3.2.1 Opgave

$$y = \frac{1 - \cos(x)}{x^2}$$

Eval:

1. $y \leftarrow \cos(x)$
2. $y \leftarrow 1 - y$
3. $z \leftarrow x$
4. $z \leftarrow zx$
5. $y \leftarrow \frac{y}{z}$

3.2.2 Oplossing

1. Conditie

   (a) Afgeleide
   $$\frac{dy}{dx} = \frac{x^2 \sin(x) - 2x(1 - cos(x))}{x^2}$$

   (b) $\delta_c y$
   $$\delta_c y = \frac{y'x}{y}\delta x = \frac{x^3 \sin(x) - 2x^2(1 - cos(x))}{1 - \cos(x)}\delta_x$$

   (c) $\delta_c y$ kan groot worden als $x \approx k\pi$. Daar is het probleem dus slecht geconditioneerd.

2. Stabiliteit

3. Zie het voorbeeld uit de tutorial over foutenanalyse.
   $$\delta_s y = -\frac{\cos(x)}{x(1 - \cos(x))}\epsilon_1 + \epsilon_2 - \epsilon_3 + \epsilon_4$$

4. $\delta_s y$ wordt groot voor $x \approx k\pi$. Daar is het algoritme dus onstabiel.

5. Conclusie Dit algoritme is voorwaarts stabiel voor alle $x \not\approx 0$. Voor $x \approx 0$ is het algoritme onstabiel (of zwak stabiel).

Oefeningen Numerieke Wiskunde

Oefenzitting 1: Een eerste kennismaking met MATLAB

1 Inleiding

MATLAB is een pakket dat voornamelijk interactief wordt gebruikt en dat vooral geschikt is om te werken met matrices. Vele functies uit welgekende pakketten voor lineaire algebra (LAPACK, ...) worden in MATLAB aangeboden. Zo zijn er bv. commando's beschikbaar voor het berekenen van de eigenwaarden en eigenvectoren van een matrix of voor de LU-decompositie van een matrix.

Deze tekst is een inleiding voor MATLAB. Eerst worden de basiscommando's overlopen. Sectie 3 geeft dan een redelijk uitgebreid overzicht van alle commando's die volgens ons gekend moeten zijn om enigszins vlot met het pakket te kunnen werken. Sectie 4 overloopt de belangrijkste program flow constructies en in Sectie 5 wordt het schrijven van MATLAB functies en scripts besproken. Sectie 6 geeft tenslotte een overzicht van de belangrijkste commando's om figuren te maken.

Het is uiteraard niet de bedoeling om onmiddellijk alle commando's in deze tekst van buiten te kennen. We hopen echter dat de lezer een idee krijgt van wat er allemaal kan en dit document later kan gebruiken om de belangrijke commando's op te zoeken.

MATLAB bevat een uitgebreide helpfunctie en het is een goede gewoonte om deze steeds te raadplegen bij het gebruik van nieuwe commando's. Door het intypen van

```
>> doc
```

verschijnt het helpvenster, dat ook een ingebouwde zoekfunctie heeft. Om te weten te komen wat de functie exp precies doet, typt men

```
>> doc exp
```

Je kan in plaats van doc ook het commando

```
>> help exp
```

gebruiken. De documentatie verschijnt hierdoor in de Command Window waarin je aan het werken bent, wat vaak handiger is. We verwijzen verder ook naar de help functie online, op `http://www.mathworks.nl/help/matlab/`.

2 Basiscommando's

Bij het werken met MATLAB krijg je op het scherm telkens een prompt. Dit is: >> . Hierachter kan je de gewenste bevelen intikken. Als je op de enter-toets drukt, wordt het bevel uitgevoerd. Merk op dat je enkel achtereenvolgende commando's kan ingeven. Het is niet mogelijk om bijvoorbeeld met de muis op een eerder uitgevoerd commando te klikken

om dit te wijzigen. Er bestaat wel de mogelijkheid om met de pijltjes toetsen te navigeren doorheen de commando's die je eerder ingegeven hebt. Zo kan je eenvoudig een commando opnieuw onder de cursor halen en eventueel aanpassen.

2.1 Het invoeren van variabelen

Getallen

```
< naam-van-de-variabele > = < waarde >
```

Voorbeeld:

```
>> x = 0.65
```

heeft als resultaat dat MATLAB aan de variabele x de waarde 0.65 toekent. MATLAB antwoordt :

```
x =

    0.6500
```

Vectoren

Vectoren worden aangemaakt door getallen gescheiden door een spatie, een komma of een puntkomma te plaatsen tussen vierkante haken. Als de getallen gescheiden worden door een puntkomma, is de vector een kolomvector, anders is het een rijvector.

```
>> v = [1 8 6 7]

v =
    1   8   6   7

>> v = [1; 8 ;6 ;7]

v =
    1
    8
    6
    7
```

Vectoren die bestaan uit equidistante elementen kan je aanmaken door gebruik te maken van de dubbelpuntoperator. Voorbeeld:

```
a:b:c
```

maakt een rijvector aan met als elementen:

```
a, a+b, a+2b, ..., c
```

Om een kolomvector te maken gebruik je de transpose operator '

```
(a:b:c)'
```

Matrices

Matrices worden rij per rij gedefinieerd. Een vierkante haak [maakt MATLAB duidelijk dat er een matrix volgt. Een matrix wordt afgesloten met]. Je kan een matrix op 2 manieren invoeren. Voorbeeld: Je wil

$$A = \begin{pmatrix} 0 & 1 & 2 \\ 3 & 4 & 5 \\ 6 & 7 & 8 \end{pmatrix}$$

dan tik je ofwel

```
>> A = [ 0 1 2 ; 3 4 5 ; 6 7 8 ]
```

ofwel

```
>> A = [ 0 1 2
         3 4 5
         6 7 8 ]
```

In beide gevallen antwoordt MATLAB :

```
A =

     0     1     2
     3     4     5
     6     7     8
```

De rijen worden dus gescheiden door een ; of door een enter, de verschillende elementen van een rij worden gescheiden door een spatie of door een komma.

2.2 De uitvoer

De inhoud van een variabele kan je opvragen door gewoon de naam van de variabele in te tikken, en dan op de enter-toets te drukken.

```
>> x
```

MATLAB antwoordt :

```
x =

    0.6500
```

Dit is het standaard uitvoerformaat van MATLAB. Wetenschappelijke notatie verkrijg je met het bevel:

```
>> format short e
```

Wil je meer cijfers van je getal zien, dan kan dat met de bevelen

```
>> format long
```

of

```
>> format long e
```

Na het intikken van dit laatste bevel, beantwoordt MATLAB

```
>> x
```

met

```
x =

     6.500000000000000e-01
```

Meerdere opties wat uitvoer van getallen betreft, kan je opvragen met het commando

```
>> doc format
```

Merk op dat MATLAB steeds uitvoer geeft als een commando wordt ingetypt. Om dit te vermijden (wat bijvoorbeeld handig is als men grote matrices als uitvoer verwacht) kan je achter het commando een ; plaatsen.

2.3 Selecteren van elementen

Je kan elementen uit een vector selecteren door gebruik te maken van ronde haakjes:

`v(i)`	geeft het i-de element van een rij- of kolom-vector terug.
`v(end)`	geeft het laatste element van een rij- of kolom-vector terug.
`v([1,3,4])`	geeft het eerste, derde en vierde element van de vector v terug

Je kan matrix-elementen als volgt selecteren :

`A(i,j)`	geeft het element op de i-de rij en de j-de kolom van A.
`A(i,:)`	geeft de hele i-de rij van A.
`A(:,j)`	geeft de hele j-de kolom van A.
`A(i:j,k:l)`	geeft de deelmatrix A(i ... j , k ... l).
`A(end,end)`	geeft het laatste element in de matrix.

Voorbeeld:

```
>> M = A(1:2,2:3)

M =

     1     2
     4     5
```

2.4 Rekenkundige operatoren

De volgende rekenkundige bewerkingen kunnen zowel voor scalars als voor vectoren en matrices gebruikt worden:

`+` optelling
`-` aftrekking
`*` vermenigvuldiging
`/` rechtse deling (`a/b` is theoretisch equivalent met `a*inv(b)`)
`\` linkse deling (`a\b` is theoretisch equivalent met `inv(a)*b`)
`^` machtsverheffing

Voorbeeld:

```
>> N = A(1:2,1:2);
>> P = N * M
P =
     4     5
    19    26
```

berekent het matrixproduct van de matrices `N` en `M`. Uiteraard moeten deze matrices de juiste dimensies hebben. Om de dimensie van de matrix `M` op te vragen kan men het commando

```
>> size(M)
```

gebruiken.

De volgende rekenkundige bewerkingen kunnen zowel voor vectoren als voor matrices gebruikt worden:

`.*` elementsgewijze vermenigvuldiging
`./` elementsgewijze deling
`.^` elementsgewijze machtsverheffing

Voorbeeld:

```
>> v = [ 9 5 4 ];
>> w = [ 6 7 0 ];
>> u = v .* w

u =
    54    35     0
```

Als je het resultaat van een bewerking niet toekent aan een variabele, wordt hiervoor automatisch de variabele ans gebruikt.

```
>> 2^5

ans =

    32

>> ans - 5

ans =

    27
```

2.5 Relationele en logische operatoren

Relationele operatoren		Logische operatoren	
==	gelijkheid	&	elementsgewijze AND
~=	ongelijkheid	|	elementsgewijze OR
<	kleiner dan	~	elementsgewijze NOT
<=	kleiner dan of gelijk	&&	"Short-Circuit" AND
>	groter dan	||	"Short-Circuit" OR
>=	groter dan of gelijk		

Een overzicht van deze operatoren krijg je met

```
>> help relop
```

Het resultaat van een logische uitdrukking is 1 als ze waar is en 0 als ze onwaar is.
Voorbeeld:

```
>> 8 == ( 2^3 )

ans =
     1

>> 2 > 6

ans =
     0
```

Merk op dat ronde haakjes kunnen worden gebruikt om de volgorde van operatoren op te leggen. Strikt genomen zijn deze in bovenstaand voorbeeld overbodig omdat de machtsverheffing een hogere prioriteit heeft dan de gelijkheid. Meer informatie over de volgorde van operatoren kan men bekomen op `http://www.mathworks.nl/help/matlab/matlab_prog/operator-precedence.html`

Je kan relationele operatoren ook toepassen op vectoren of matrices. Het resultaat is een logische array. De logische operatoren &, | en ~ kunnen toegepast worden op deze logische arrays.
Voorbeeld:

```
>> [1 2 3 4 3 5] <= 3

ans =
     1     1     1     0     1     0

>> [1 2 3 4 3 5] <= 3 & [1 2 3 4 3 5] >= 2

ans =
     0     1     1     0     1     0
```

De logische operatoren && en || zijn vooral van belang bij `while` lussen en `if else` structuren, zie Sectie 4.

2.6 Selecteren van element met logische arrays

Naast het selecteren van elementen in vectoren en matrices door exacte indices op te geven zoals beschreven in Sectie 2.3, kan je ook logische indices opgeven. Hiermee kan je bijvoorbeeld alle elementen van een matrix nul maken die aan een bepaalde voorwaarde voldoen:

```
>> A = [1 2 3; 4 5 6]

A =
     1     2     3
     4     5     6

>> A( A<=3 | A==6 ) = 0

A =
     0     0     0
     4     5     0
```

Met de functie `find` kan je dan de indices opvragen van elementen die aan een bepaalde voorwaarde voldoen. Merk op dat je eigenlijk een logische array aan `find` geeft.

```
>> a = [1 3 4 -1 5 -2]

a =
     1     3     4    -1     5    -2

>> find(a<0)

ans =
     4     6
```

Je kan dus ook elementen selecteren m.b.v. `find`

```
>> A(find( A<=3 | A==6 )) = 0
```

maar dit is eigenlijk een omweg.

2.7 Oefeningen

Oef 1. Maak een rijvector v met 100 elementen van de vorm: $1, \frac{1}{2}, \frac{1}{3}, \frac{1}{4}, \ldots$

Oef 2. Maak een kolomvector met als elementen $1024, 512, \ldots, 16, 8, 4, 2, 1$

Oef 3. Maak voor de volgende functies f de vector $[f(0), f(0.1), f(0.2), \ldots, f(2)]$.

(a) $f(x) = x^2 + 2x + 1$.

(b) $f(x) = \dfrac{x + 3^x}{x^{1/2} + x^{-2}}$

Oef 4. Maak twee matrices

$$G = \begin{pmatrix} 1 & 2 & 3 & 4 \\ 5 & 6 & 7 & 8 \\ 9 & 10 & 11 & 12 \end{pmatrix} \text{ en } H = \begin{pmatrix} 1 & 1 & 1 & 1 \\ 2 & 2 & 2 & 2 \\ 3 & 3 & 3 & 3 \end{pmatrix}$$

(a) Bereken het verschil tussen G en H.

(b) Vermenigvuldig G met H elementsgewijs.

(c) Vermenigvuldig G met H^T .

(d) Bewaar de eerste drie kolommen van de matrix G in G_3 en bereken G_3^2.

Oef 5. Gegeven $x = \begin{bmatrix} 3 & 15 & 9 & 12 & -1 & 0 & -12 & 9 & -6 & 1 \end{bmatrix}$, voer het commando uit dat

(a) de negatieve waarden van x gelijk stelt aan 0;

(b) de waarden van x groter of gelijk aan 9 stockeert in de vector y.

(c) het element in x na het element 12 gelijk stelt aan 100 (gebruik makend van `find`).

3 Functies in MATLAB

In wat volgt worden verschillende functies overlopen die nuttig zijn voor het vak Numerieke Wiskunde. We proberen een redelijk volledig overzicht te geven.

Let op: je gebruikt best geen functienamen als naam voor je variabelen, want dan werkt de functie niet meer. Zie oefening 6.

3.1 Elementaire functies in MATLAB

Het argument geef je in tussen ronde haakjes. Voorbeeld:

```
>> log(1)

ans =
     0
```

Een overzicht vindt je met

```
>> doc elfun
```

Belangrijke functies zijn

```
sqrt, log, log10, exp, sin, cos, tan, asin, sinh, asinh, ...
abs, round, ceil, floor, sign, rem ...
```

3.2 Algemene functies in MATLAB

Resultaten van berekeningen in MATLAB worden vaak elders opnieuw gebruikt. De waarden van variabelen kunnen bewaard worden in een .mat-bestand. Dit gebeurt met het commando

```
>> save filename
```

waardoor alle gebruikte variabelen worden bewaard in het bestand met naam 'filename.mat' De extensie .mat hoeft niet opgegeven te worden. Hoeven er maar een aantal variabelen te worden opgeslagen, dan dienen die gespecificeerd te worden na de bestandsnaam. Het opgeslagen bestand kan in MATLAB worden ingelezen met de opdracht

```
>> load filename
```

waarbij de extensie .mat mag worden weggelaten. De variabelen met hun waarden kunnen nu weer gebruikt worden. **Let op:** alle variabelen in de workspace die dezelfde naam hebben worden overschreven.

Volgende tabel geeft een overzicht van verschillende algemene MATLAB-functies:

`who`/`whos`	geeft een lijst van de variabelen die je gebruikt.
`clear`	geeft de gebruikte geheugenruimte terug vrij.
`clc`	wist de Command Window.
`save`	bewaart de variabelen van de huidige MATLAB-sessie in een .mat-bestand.
`load`	leest de variabelen uit een .mat-bestand terug in zodat deze terug de waarden hebben van net voor de laatste save.
`pause`	laat Matlab wachten op de gebruiker
`path`	zet hierin de padnamen van de directories waar MATLAB naar functies (.m-bestanden) moet zoeken.
`pwd`	geeft de huidige map in een string.
`disp`	geeft variabele of string weer in Command Window
`fprintf`	schrijft geformatteerde string naar bestand of Command Window
`tic`/`toc`	meet de uitvoeringstijd van de commando's tussen `tic` en `toc`

De functie `fprintf` is zeer handig voor het printen van output naar de Command Window. Deze output is een geformatteerde string. Voorbeeld:

```
>> fprintf('Dit is iteratie %i: x = %0.3f, fout = %0.3e \n', ...
      3,0.12345,0.00098765)
Dit is iteratie 3: x = 0.123, fout = 9.877e-04
```

De variabelen (of in dit geval getallen) die worden meegegeven worden geformatteerd naar:

`%i`	een geheel getal
`%0.3f`	een komma getal met 3 cijfers na de komma
`%0.3e`	een getal in wetenschappelijke notatie met 3 cijfers na de komma

Merk ook de drie puntjes `...` op. Deze dienen om een regel te splitsen over meerdere lijnen.

3.3 Voorgedefinieerde variabelen

`pi`	het getal pi.
`Inf`	oneindig
`NaN`	Not a Number, om resultaten als $\frac{0}{0}$ en $\frac{\infty}{\infty}$ voor te stellen

3.4 Functies voor vectoren en/of matrices

Handige tools

`length`	geeft het aantal elementen van een vector terug.
`size`	geeft het aantal rijen en kolommen van een matrix terug.
`zeros(m,n)`	geeft een matrix van dimensie m x n met allemaal nullen.
`ones(m,n)`	geeft een matrix van dimensie m x n met allemaal eentjes.
`eye(n)`	geeft de eenheidsmatrix terug van dimensie n.
`rand(m,n)`	geeft een (m x n)-matrix met randomgetallen, uniform verdeeld tussen 0 en 1.
`linspace(a,b,N)`	geeft een rijvector van N equidistante punten tussen a en b terug
`reshape(A,m,n)`	hervormt de matrix A naar dimensie $m \times n$, kolom per kolom.
`find`	zoekt indices van niet-nul elementen
`fliplr(A)`	draait de matrix `A` om in de links/rechts richting. (noot: dit is equivalent met `A(:,end:-1:1)`)
`flipud(A)`	draait de matrix A om in de onder/boven richting.
`diag(A)`	als A een vector: diagonaalmatrix met de elementen van A op de diagonaal. als A een matrix: vector met de diagonaalelementen van A.
`tril(A)`	geeft het onderdriehoeksdeel terug van A.
`triu(A)`	geeft het bovendriehoeksdeel terug van A.
`A'`	geeft de getransponeerde van de matrix A.
`A(:)`	zet alle kolommen van A onder elkaar in een vector.
`sort`	sorteert de elementen van een vector of matrix.
`sortrows`	sorteert de elementen van een matrix.
`sum`	sommeert de elementen van een matrix volgens rijen of kolommen
`prod`	vermenigvuldigt de elementen van een matrix volgens rijen of kolommen
`max` en `min`	geeft maximum/mininum terug en ook de bijhorende index

Numerieke algoritmen

`inv`	berekent de inverse matrix (vierkante matrix !).
`det`	berekent de determinant (vierkante matrix !).
`cond`	berekent het conditiegetal van een matrix.
`rank`	bepaalt de rang van een matrix.
`norm`	berekent de norm van een matrix of van een vector.
`A\b`	berekent de oplossing van het stelsel $Ax = b$ als de matrix A vierkant is (gebruikt verschillende algoritmen naargelang het type matrix A, voor een algemene volle matrix is dit Gauss eliminatie)
`lu`	geeft de factoren terug na Gauss-eliminatie (met optimale rij-pivotering).
`chol`	berekent de Cholesky-factorizatie.
`qr`	berekent de QR factorizatie.
`svd`	berekent de singuliere waarden ontbinding.
`eig`	berekent eigenwaarden en eigenvectoren.
`polyval`	evalueert een veelterm.
`poly`	berekent veelterm met gegeven nulpunten
`roots`	berekent nulpunten van een veelterm

Voorbeelden

Na het intikken van

```
>> [L,U] = lu(A)
```

geeft MATLAB twee matrices L en U terug; L is een benedendriehoeksmatrix (op een permutatie van de rijen na) met 1-tjes op de diagonaal en U is een bovendriehoeksmatrix. Om een stelsel Ax = b op te lossen, gebruik je de MATLAB-deling

```
>> x = A \ b
```

Hierbij wordt impliciet ook Gauss-eliminatie met optimale rij-pivotering gebruikt. Als A geen vierkante matrix is (dit betekent dat het aantal vergelijkingen verschilt van het aantal onbekenden), dan krijg je de kleinste kwadraten oplossing.

```
>> [V,D] = eig(A)
```

geeft een diagonaal matrix D met de eigenwaarden en een volle matrix V waarvan de kolommen de eigenvectoren zijn zodat A V = V D. (Hier moet A een vierkante matrix zijn!)

Merk op dat `x = eig(A)` een andere output geeft dan `[V,D] = eig(A)`. Je moet dus expliciet het aantal output argumenten meegeven dat je wil terugkrijgen.

3.5 Oefeningen

Oef 6. Deze oefening laat zien waarom je best geen functienamen gebruikt als naam voor je variabelen. Voer de volgende opdrachten uit in de onderstaande volgorde:

(a) `>> sqrt`

(b) `>> sqrt(2)`

(c) `>> a = sqrt(2)`

(d) `>> sqrt = sqrt(2)`

(e) `>> sqrt(1)`

(f) `>> sqrt(4)`

(g) `>> clear sqrt`

(h) `>> sqrt(4)`

Op welke lijnen (a t/m h) wordt `sqrt` als een functie gezien en op welke als een variabele? Waarom krijg je bij (e) geen foutmelding en bij (f) wel?

Oef 7. Bereken de volgende uitdrukkingen met MATLAB:

- $\sin(\frac{\pi}{4})$
- e^2
- $\frac{\tan(x)}{\sqrt{1+x^2}}$, met $x = 4$

Oef 8. Voer de twee opdrachten

```
>> f = 2
>> g = 3 + f
```

uit. Bekijk de variabelen in het geheugen met `whos`. Sla de variabelen f en g op in het bestand `probeer.mat`. Wis alle variabelen met `clear` en controleer met `whos`. Laad de file `probeer.mat`. Kijk of de variabelen f en g bekend zijn. Waar is het bestand `probeer.mat` terecht gekomen?

Oef 9. Evaluatie van de opdracht pause(30) heeft als effect dat MATLAB 30 seconden pauze houdt. Voer deze opdracht uit en pas dan de toetscombinatie ctrl+C toe. Wat is het effect?

Oef 10. Maak met behulp van `rand` een random 10 bij 10 matrix R en bereken zijn inverse. Controleer of hun product gelijk is aan de eenheidsmatrix.

Oef 11. Gebruik het commando `diag` om een diagonaalmatrix met de elementen $10, 20, 30, \ldots 80$ aan te maken. Bereken daarna de determinant van deze matrix en controleer dat deze gelijk is aan het product van de elementen op de diagonaal. (Zie de functies `prod` en `det` .)

4 Constructies: if, for, while, switch

Deze constructies kunnen zowel interactief in de Command Window als in een .m-bestand (zie Sectie 5) worden gebruikt.

4.1 if - elseif - else - end

Syntaxis:

```
if condition
  statements;
elseif condition
  statements;
else
  statements;
end
```

Voorbeeld:

```
if i<=n && fout > 1E-6
  i=j+1;
  j=i/2;
elseif i == j+1
  j=0;
else
  j=2*i;
end
```

Het voorbeeld slaat op weinig. De eerste conditie is wel een goed voorbeeld van de operator `&&`. De regels erna worden enkel uitgevoerd als beide condities waar zijn. Indien echter de eerste conditie `i<=n` onwaar is, dan wordt onmiddellijk naar de `elseif` overgegaan.

4.2 for - end

Syntaxis:

```
for variable = vector
  statements;
end
```

Voorbeeld:

```
x=0;
for i=1:5
   x=x+i;
end
```

Het is mogelijk om met het commando `break` bij het voldaan zijn van een bepaalde conditie de lus te verlaten:

```
x=0;
for i=1:5
   x=x+i;
   if abs(x) < 10
      break
   end
end
```

4.3 while - end

Syntaxis:

```
while condition
  statements;
end
```

Voorbeeld:

```
while x <= 5
  x=x+1;
  y=y+x;
end
```

Ook hier kan je gebruik maken van `break`.

4.4 switch - case - otherwise - end

Syntaxis:

```
switch variable
  case value1
    statements;
  case value2
   statements;
 ...
  case valueN
   statements;
 otherwise
   statements;
end
```

Voorbeeld:

```
switch a
  case 0
    x = A\b;
  case 1
    x = eig(A);
  case 2
    x = sum(A,1);
 otherwise
   disp('Error')
end
```

5 Zelf functies en scripts schrijven

5.1 Functies

In MATLAB kan je zelf ook functies schrijven om complexe berekeningen uit te voeren. Zodoende kan je het pakket naar je eigen behoefte verder uitbouwen. Een dergelijke functie wordt meestal met de MATLAB editor geschreven en opgeslagen (in ASCII-formaat) in een bestand met extensie '.m'. De eerste regel van een functie bevat bijvoorbeeld:

```
function [x,y] = naam(a,b,c)
```

Als je dit bestand opslaat met de naam naam.m, kan je deze functie in MATLAB oproepen als:

```
>> [x,y] = naam(a,b,c)
```

Als je

```
>> x = naam(a,b,c)
```

doet, dan krijg je standaard enkel de eerste output variabele terug.

De parameters a, b, c zijn de gegevens die aan de functie worden doorgegeven. Het resultaat moet binnen de functie worden toegekend aan de variabelen x en y. Uiteraard moeten de namen van de variabelen en parameters binnen het .m-bestand en bij het oproepen van de functie niet hetzelfde zijn. Let op dat de **uitvoerparameters tussen vierkante haken** staan en de **invoervariabelen tussen ronde haken**. Als er één of geen enkele uitvoerparameter is, kan je de vierkante haken weglaten. De ronde haken kan je weglaten als er geen invoer nodig is.

Na deze functiehoofding voeg je best wat commentaar toe over het doel van de functie, en bijvoorbeeld de volgorde waarin je de argumenten moet geven. Dit kan je doen door je regel te laten beginnen met een `%` -teken. Deze eerste commentaarlijnen verschijnen als antwoord op

```
>> help naam
```

Je kan uiteraard ook op andere plaatsen in je functie commentaar schrijven.

Vergeet niet om achter elk commando een ; te schrijven, anders krijg je bij het uitvoeren van de functie alle tussenresultaten op je scherm. Om de functie te verlaten voor de laatste opdracht is uitgevoerd, bv. bij het slagen van een bepaalde test, gebruik je `return`.

5.2 Scripts

Indien je een functie wenst te schrijven zonder invoer en uitvoerparameters, dan hoef je de eerste regel

```
>> function [x,y] = naam(a,b,c)
```

niet toe te voegen aan je bestand. Het resulterende bestand is nu geen functie maar een script dat je kan uitvoeren door in de editor op de run knop te duwen of door de naam van het bestand in te geven in het Command Window. Het voordeel aan een script is dat alle variabelen die gebruikt zijn na het uitvoeren van het script nog steeds in je workspace zitten. (Je kan dit checken met `whos`.)

6 Grafieken

`figure`	open een nieuwe figuur
`plot(x,y)`	plot vector `y` t.o.v. vector `x`
`semilogx(x,y)`	zoals `plot`, maar met een logaritmische schaal op de x-as
`semilogy(x,y)`	zoals `plot`, maar met een logaritmische schaal op de y-as
`loglog(x,y)`	zoals `plot`, maar met een logaritmische schaal op de x-as én de y-as
`hold on`	houdt huidige plot en schaal van de assen (lineair, logaritmisch, ...) vast zodat nieuwe plots worden toegevoegd i.p.v. de oude te overschrijven
`hold all`	zoals `hold on`, maar elke nieuwe plot is in een andere kleur
`hold off`	doet `hold on` teniet
`xlabel`	voeg label toe aan x-as
`ylabel`	voeg label toe aan y-as
`title`	voeg titel toe
`legend`	voeg legende toe
`subplot`	toon meerdere plots naast en/of boven elkaar in dezelfde figuur
`clf`	wis de plot op de huidige figuur

Voorbeelden

Plot de exponentiële functie in $[-1, 1]$:

```
>> x = linspace(-1,1,100);
>> figure
>> plot(x,exp(x))
>> xlabel('x')
>> ylabel('y')
>> title('exponential function')
```

Plot een sinus en een cosinus in een figuur met 'handle' 3 en voeg een legende toe:

```
>> x = linspace(-2*pi,2*pi,100);
>> figure(3)
>> plot(x,sin(x))
>> hold all
>> plot(x,cos(x))
>> legend('sin','cos')
```

Plot 2^{-x} in $x = 0, 1, 2, \ldots, 30$ met rode sterretjes en met logaritmisch geschaalde y-as in figuur 3. Als figuur 3 al bestond, wis dan eerst alle plots:

```
>> x = 0:30;
>> figure(3),clf
>> semilogy(x,2.^(-x),'*r')
```

6.1 Oefeningen

Oef 12. Maak een script `oefening_logplots.m`. Plot hierin de functies $f(x) = 2^x$ voor $x \in [0, 30]$ en $g(x) = x^3$ voor $x \in [0, 100]$. Doe dit in verschillende figuren, waarbij je de commando's `plot`, `loglog` en `semilogy` vergelijkt. Wat valt je op?

Oef 13. Schrijf een script `oefening_legende.m` waarin de functies sin(t) en cos(t) getekend worden op het interval $0 \leq t \leq 2$ in een figuur. Zorg ervoor dat de sinus in het rood en de cosinus in het groen getekend wordt. Benoem de assen en voorzie een legende.

Oef 14. Maak een script `oefening_subplots.m`. Maak een figuur met twee subplots boven elkaar. In de eerste subplot teken je de functie $f(x) = \dfrac{\sin(x) + 2x}{1 + x^2}$ op het interval $[0, 2\pi]$, in de tweede subplot plot je een random vector met lengte 1000.

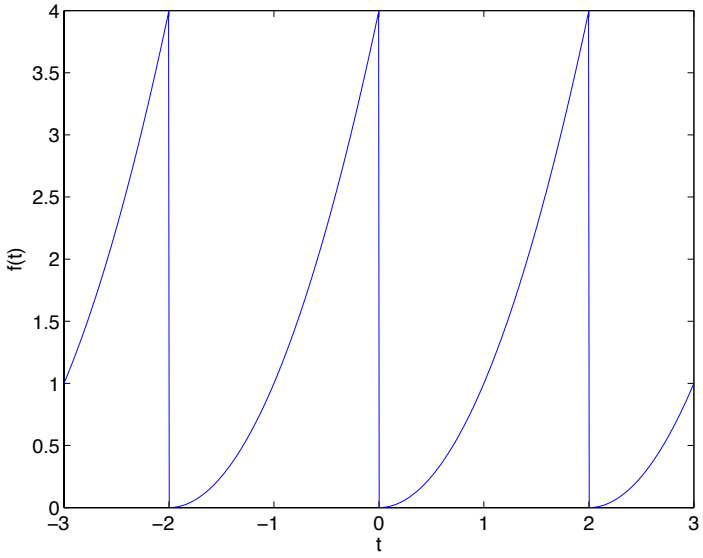
Oef 15. De periodieke functie f met periode 2 is gegeven door:

$$f(t) = \begin{cases} t^2 & , \quad 0 \leq t \leq 2 \\ f(t-2) & , \quad 2 \leq t \\ f(t+2) & , \quad t < 0 \end{cases}$$

Definieer deze functie in MATLAB. Zorg ervoor dat de functie ook voor vectoren werkt. (Hint: er zijn minstens drie mogelijke implementaties. De snelste implementatie maakt gebruik van `rem`.) Ga na dat je functie werkt door de commando's

```
>> t = linspace(-3,3,1000);
>> figure
>> plot(t,f(t))
```

uit te voeren en te vergelijken met Figuur 1.



Figuur 1: $f(t)$ van Oefening 15

Oefeningen Numerieke Wiskunde

Oefenzitting 2: Bewegende kommavoorstelling en foutenanalyse

1 Bewegende kommavoorstelling

Probleem 1. De IEEE standaard voorziet voor basis $b = 2$ in enkelvoudige-precisiegetallen en dubbele-precisiegetallen (zie boek H2 §5.5). Deze hebben, respectievelijk, een 32-bit en een 64-bit voorstelling. Hoeveel bits worden hiervan gebruikt voor de mantisse?

Probleem 2. Hoeveel dubbele-precisiegetallen kunnen er worden voorgesteld

- tussen de getallen 1 en 2?
- tussen de getallen 7 en 9?

Probleem 3. Hoe ziet de rij $x_1, x_2, \ldots$, met

$$x_1 = 1; \quad x_{n+1} = fl(x_n + 1),\ n = 1, 2, \ldots,$$

er voor grote waarden van n uit, op een machine met 3 decimale cijfers voor de mantisse?

Probleem 4. Analyseer de volgende algoritmen voor het bepalen van de basis, b, en het aantal cijfers in de mantisse, p:

bepaal_b <uit: b >
1. $A \leftarrow 1$
2. while $(A + 1) - A = 1$
   2.1. $A \leftarrow 2 * A$
3. $i \leftarrow 1$
4. while $(A + i) = A$
   4.1. $i \leftarrow i + 1$
5. $b \leftarrow (A + i) - A$

bepaal_p <in: b; uit: p >
1. $p \leftarrow 1$
2. $z \leftarrow b$
3. while $(z + 1) - z = 1$
   3.1. $p \leftarrow p + 1$
   3.2. $z \leftarrow z * b$

- Ga de werking van de algoritmen na voor $b = 10$ en $p = 3$.
- (extra opgave) Toon formeel de werking van de algoritmen aan voor om het even welke b en p.
  (Hint: op regel 2.1 van **bepaal_b** zou er ook $A \leftarrow A+1$ mogen staan. Waarom staat er wat er nu staat?)

2 Foutenanalyse

Het doel van een formele foutenanalyse is het vinden van een a priori bovengrens op het effect van de afrondingsfouten en hun voortplanting doorheen een berekening $y = f(x)$. We zullen dit effect meten met de relatieve fout op het eindresultaat.

Veronderstel dat de afrondingsfouten die gemaakt worden voldoen aan $\text{fl}(x) = x(1+\epsilon)$ met $|\epsilon| \leq \epsilon_{\text{mach}}$ en dat iedere elementaire bewerking exact wordt uitgevoerd en dan wordt afgerond. D.w.z.,

$$\begin{aligned} \text{fl}(a \circ b) &= (a \circ b)(1+\epsilon), \\ \text{fl}(\Box a) &= (\Box a)(1+\epsilon), \end{aligned}$$

waarbij $\circ$ een van de bewerkingen $+, -, *$ of $\div$ voorstelt en $\Box$ een elementaire functie zoals $\sqrt{}, \sin, \exp, \ldots$. Door het berekenen van een eerste orde benadering van de fout, bekom je uiteindelijk iets van de vorm

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

waarbij $E(x)$ een functie is die enkel afhangt van x.

Bij het doen van een formele foutenanalyse kan je twee methodes volgen om een eerste orde benadering te bekomen van de fout. Eerst bekijken we een methode waarbij je de eerste orde benadering in één keer berekent via een Taylor benadering in meerdere veranderlijken. In Sectie 3 wordt een alternatieve methode voorgesteld die je zelf thuis kan bekijken.

2.1 Voorbeeld

Als voorbeeld voeren we een foutenanalyse uit voor de berekening van

$$y = \sqrt{1+x} - 1,$$

waarbij men de berekeningen uitvoert volgens de uitdrukking hierboven. Men veronderstelt dat x exact voorgesteld kan worden op de machine. Op die manier worden enkel de afrondingsfouten in rekening gebracht die gemaakt worden bij de berekeningen.

De berekende waarde voor y is dan

$$\bar{y} = fl\left(fl(\sqrt{fl(1+x)}) - 1\right) = \left((\sqrt{(1+x)(1+\epsilon_1)})(1+\epsilon_2) - 1\right)(1+\epsilon_3)$$

Hierbij stellen $\epsilon_i,\ i = 1, 2, 3$, de relatieve fouten voor die gemaakt worden bij afronding na iedere bewerking. We weten dat

$$|\epsilon_i| \leq \epsilon_{mach},\ i = 1, 2, 3.$$

We interpreteren, in de uitdrukking voor $\bar{y}$, x als een parameter en $\epsilon_i,\ i = 1, 2, 3$, als variabelen,

$$\bar{y} = F(\epsilon_1, \epsilon_2, \epsilon_3).$$

Vermits de ϵ_i zeer klein zijn, laat $F(\epsilon_1, \epsilon_2, \epsilon_3)$ zich goed benaderen door de Taylorontwikkeling rond $(0,0,0)$ afgebroken na de lineaire termen

$$\bar{y} = F(\epsilon_1, \epsilon_2, \epsilon_3) \approx F(0,0,0) + \epsilon_1 \frac{\partial F}{\partial \epsilon_1}(0,0,0) + \epsilon_2 \frac{\partial F}{\partial \epsilon_2}(0,0,0) + \epsilon_3 \frac{\partial F}{\partial \epsilon_3}(0,0,0).$$

De constante term $F(0,0,0)$ in de lineaire benadering is de exacte waarde y. Voor de berekening van de partiële afgeleiden gaan we als volgt te werk

$$\begin{aligned} F(\epsilon_1, 0, 0) &= \sqrt{(1+x)(1+\epsilon_1)} - 1 = \sqrt{1+x}(1+\epsilon_1)^{\frac{1}{2}} - 1 \\ \frac{\partial F}{\partial \epsilon_1}(\epsilon_1, 0, 0) &= \sqrt{1+x}\frac{1}{2}(1+\epsilon_1)^{-\frac{1}{2}} \\ \frac{\partial F}{\partial \epsilon_1}(0,0,0) &= \frac{1}{2}\sqrt{1+x} \end{aligned}$$

Na uitwerking van de andere partiële afgeleiden vinden we de lineaire benadering voor $\bar{y}$ en een benaderende formule voor de absolute en de relatieve fout

$$\begin{aligned} \bar{y} &\approx y + \frac{1}{2}\sqrt{1+x}\,\epsilon_1 + \sqrt{1+x}\,\epsilon_2 + y\epsilon_3 \\ \bar{y} - y &\approx \frac{1}{2}\sqrt{1+x}\,\epsilon_1 + \sqrt{1+x}\,\epsilon_2 + y\epsilon_3 \\ \frac{\bar{y}-y}{y} &\approx \frac{\sqrt{1+x}}{2(\sqrt{1+x}-1)}\epsilon_1 + \frac{\sqrt{1+x}}{\sqrt{1+x}-1}\epsilon_2 + \epsilon_3. \end{aligned}$$

In de limiet voor $x \to 0$ worden de coëfficiënten vóór ϵ_1 en ϵ_2 oneindig groot. Dit betekent dat voor kleine x de relatieve fout op de berekende y zeer groot kan zijn. Voor grote x daarentegen is het berekende resultaat nauwkeurig.

Een bovengrens voor de relatieve fout vinden we als volgt

$$\begin{aligned} \left|\frac{\bar{y}-y}{y}\right| &\approx \left|\frac{\sqrt{1+x}}{2(\sqrt{1+x}-1)}\epsilon_1 + \frac{\sqrt{1+x}}{\sqrt{1+x}-1}\epsilon_2 + \epsilon_3\right| \\ &\leq \left|\frac{\sqrt{1+x}}{2(\sqrt{1+x}-1)}\epsilon_1\right| + \left|\frac{\sqrt{1+x}}{\sqrt{1+x}-1}\epsilon_2\right| + |\epsilon_3| \\ &= \left|\frac{\sqrt{1+x}}{2(\sqrt{1+x}-1)}\right||\epsilon_1| + \left|\frac{\sqrt{1+x}}{\sqrt{1+x}-1}\right||\epsilon_2| + |\epsilon_3| \\ &\leq \epsilon_{mach}\left(\left|\frac{\sqrt{1+x}}{2(\sqrt{1+x}-1)}\right| + \left|\frac{\sqrt{1+x}}{\sqrt{1+x}-1}\right| + 1\right) \\ &= \epsilon_{mach}\left(\left|\frac{3\sqrt{1+x}}{2(\sqrt{1+x}-1)}\right| + 1\right) \end{aligned}$$

2.2 Oefeningen

Probleem 5. Voer de foutenanalyse uit voor de berekening van y van het voorbeeld indien de berekeningen verlopen volgens de wiskundig equivalente formule

$$y = \frac{x}{\sqrt{x+1}+1}.$$

Verklaar dat dit rekenschema nauwkeuriger is dan het vorige wanneer x klein is.

Probleem 6. Voer de foutenanalyse uit voor de berekening van

$$y = \frac{1-\cos(x)}{x^2}.$$

Naar welk getal convergeert y voor heel kleine x? Is de formule dan nog nauwkeurig?

Probleem 7. Beschouw volgend algoritme voor het berekenen van de som van n getallen:

som <in: $a_1, \ldots, a_n$; uit: $S = \sum_{i=1}^{n} a_i$ >
1. $S \leftarrow a_1$
2. for $i = 2 : 1 : n$
   2.1. $S \leftarrow S + a_i$

Toon aan dat de absolute fout van dit algoritme gelijk is aan

$$\bar{S} - S \approx \epsilon_2(a_1 + a_2) + \epsilon_3(a_1 + a_2 + a_3) + \ldots + \epsilon_n(a_1 + a_2 + \ldots + a_n)$$

of

$$\bar{S} - S \approx \sum_{k=2}^{n} \epsilon_k \, b_k, \quad \text{met} \quad b_k = \sum_{i=1}^{k} a_i.$$

(Zie ook slides en handboek.)

3 Alternatieve methode en extra oefeningen

3.1 Alternatieve methode

Bij deze methode ga je waar nodig de stelling van Taylor in één veranderlijke toepassen om zo in een aantal stappen een eerste orde benadering te bekomen.

We illustreren de methode op hetzelfde voorbeeld: de berekening van

$$y = \sqrt{1+x} - 1.$$

De berekende waarde voor y is

$$\bar{y} = fl\left(fl(\sqrt{fl(1+x)}) - 1\right) = \left((\sqrt{(1+x)(1+\epsilon_1)})(1+\epsilon_2) - 1\right)(1+\epsilon_3) \qquad (1)$$

met $|\epsilon_i| \leq \epsilon_{mach},\ i = 1, 2, 3.$

Ditmaal herschrijven we $\bar{y}$ als $\bar{y} \approx y(1+\delta)$ door in (1) gebruik te maken van de Taylor reeks rond nul

$$f(\epsilon) = f(0) + f'(0)\epsilon + f''(0)\frac{\epsilon^2}{2} + \dots$$

en hogere orde termen te verwaarlozen. Bv.

$$\sqrt{1+\epsilon} \approx 1 + \frac{1}{2}\epsilon, \quad \text{en} \quad (1+\epsilon_1)(1+\epsilon_2) \approx 1 + \epsilon_1 + \epsilon_2,$$

waarbij we veronderstellen dat ϵ, ϵ_1 en ϵ_2 klein genoeg zijn. We krijgen dus

$$\begin{aligned}
\bar{y} &= \left((\sqrt{(1+x)(1+\epsilon_1)})(1+\epsilon_2) - 1 \right)(1+\epsilon_3) \\
&= \left(\sqrt{1+x}\,\sqrt{1+\epsilon_1}\,(1+\epsilon_2) - 1 \right)(1+\epsilon_3) \\
&\approx \left(\sqrt{1+x}\,(1+\frac{1}{2}\epsilon_1)\,(1+\epsilon_2) - 1 \right)(1+\epsilon_3) \\
&\approx \left((\sqrt{1+x})(1+\frac{1}{2}\epsilon_1 + \epsilon_2) - 1 \right)(1+\epsilon_3) \\
&= \left(y + (\sqrt{1+x})(\frac{1}{2}\epsilon_1 + \epsilon_2) \right)(1+\epsilon_3) \\
\implies \bar{y} - y &\approx \frac{1}{2}\sqrt{1+x}\,\epsilon_1 + \sqrt{1+x}\,\epsilon_2 + y\epsilon_3 \\
\frac{\bar{y}-y}{y} &\approx \frac{\sqrt{1+x}}{2(\sqrt{1+x}-1)}\epsilon_1 + \frac{\sqrt{1+x}}{\sqrt{1+x}-1}\epsilon_2 + \epsilon_3.
\end{aligned}$$

waarbij we steeds de tweede orde termen verwaarloosd hebben. Dit is uiteraard hetzelfde resultaat als voor de eerste methode.

3.2 Extra oefeningen

Probleem 8. Voer een foutenanalyse uit voor de volgende berekeningen. Waar verwacht je numerieke moeilijkheden?

(a) $y = x\sin(x)$

(b) $y = \dfrac{1-\cos(x)}{\sin(x)}$

(c) $y = \dfrac{1-e^{-2x}}{x}$

(d) $y = (1+x)^{\frac{1}{x}}$

(e) $y = \sqrt{e^x - 1}$

(f) $y = \sin(\frac{1}{x})$

(g) $y = \left(1 + x^2\right)^{x^2}$

(h) $y = \dfrac{e^{x^2} - e^{-x^2}}{2x^2}$

Probleem 9. Beschouw volgende algoritmen voor het berekenen van product en scalair product:

- **product** <in: $a_1, \ldots, a_n$; uit: $P = \Pi_{i=1}^n a_i >$
  1. $P \leftarrow a_1$
  2. voor $i = 2 : 1 : n$
     2.1 $P \leftarrow P * a_i$

- **scalair_product** <in: $a_1, \ldots, a_n, b_1, \ldots, b_n$; uit: $SP = \sum_{i=1}^n a_i b_i >$
  1. $SP \leftarrow a_1 b_1$
  2. voor $i = 2 : 1 : n$
  2.1 $SP \leftarrow SP + a_i * b_i$

Maak een afschatting van de absolute fout m.b.v. een foutenanalyse.

Probleem 10. Analyseer het volgende recursieve algoritme:
SOM <in: $a_1, \ldots, a_n$; uit: $S = \sum_{i=1}^n a_i >$
1. als $n = 1$
   1.1 $S \leftarrow a_1$
   1.2 anders
   $S \leftarrow S(a_1, \ldots, a_{n\text{div}2}) + S(a_{n\text{div}2+1}, \ldots, a_n)$

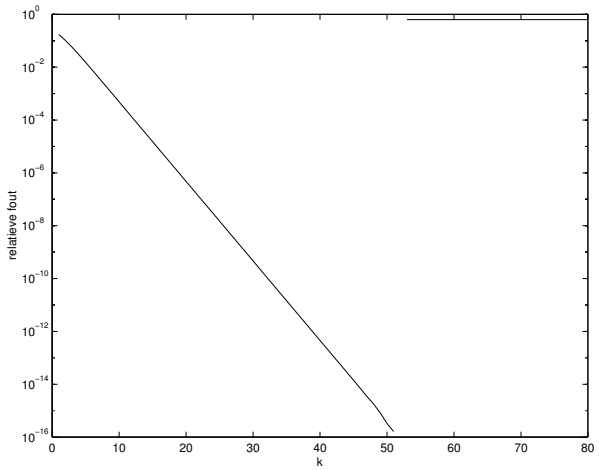
De bewerking 'div' levert het quotient van de gehele deling, dus $(2n+1)\text{div}2 = (2n)\text{div}2 = n,\ n \in \mathbb{N}$.
Toon aan dat in eindige precisie, de fout $S - \bar{S}$ voldoet aan:

$$n \leq 2^k,\ k \in \mathbb{N} \rightarrow |S - \bar{S}| \leq k\epsilon_{\text{mach}} \left(|a_1| + |a_2| + \cdots + |a_n|\right).$$

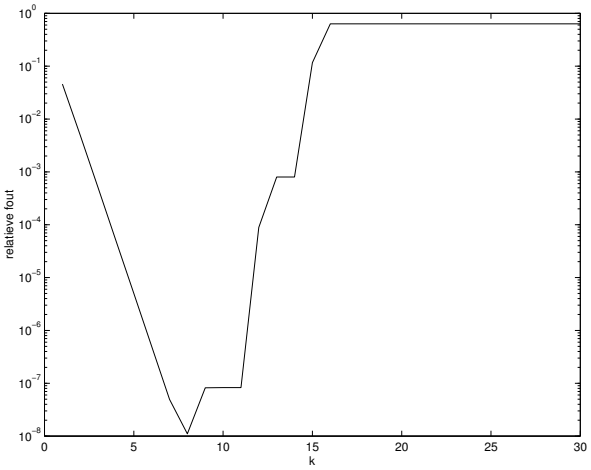
Een recursief algoritme leent zich tot een bewijs door ...

Probleem 11. (Examenvraag)
We weten dat $\lim_{x\to\infty}(1 + 1/x)^x = e^1$. We gaan de waarde e benaderen door $\tilde{e}_k = (1 + 1/x_k)^{x_k}$ uit te rekenen voor

- $x_k = 2^k$ en
- $x_k = 10^k$.



(a) Relatieve fout



(b) Relatieve fout

In de figuren hieronder geven we de relatieve fout weer, d.w.z. $\left|\frac{\tilde{e}_k - e}{e}\right|$. De eerste figuur geeft de relatieve fout voor de machten van 2 terwijl de tweede figuur de fouten voor machten van 10 weergeeft.

Waarom zijn de twee grafieken zo verschillend?
Kan men hieruit afleiden met welke basis en met hoeveel beduidende cijfers de computer werkt?
Verklaar in detail je antwoord.

Oefeningen Numerieke Wiskunde

Oefenzitting 3 (PC): Bewegende kommavoorstelling en foutenanalyse

Voor het uitwerken van de opgaven heb je de '.m'-bestanden nodig die je kan vinden op Toledo.

1 Bewegende kommavoorstelling

Probleem 1. (Berekening van ϵ_{mach}) Gebruik `bepaalb.m` om de basis te berekenen van het talstelsel waarmee MATLAB werkt. Met `bepaalp.m` kan je het aantal cijfers in de mantisse berekenen. Bekijk deze bestanden. Zet de `fprintf` regels uit commentaar en roep beide functies opnieuw aan.

Vergelijk de berekende waarde ϵ_{mach} (formule handboek) met de voorgedefinieerde variabele `eps` in MATLAB. Opgelet: `eps` stelt niet de machineprecisie voor, maar wel het verschil tussen 1 en het kleinste getal groter dan 1, voorstelbaar in de floating point-voorstelling. Controleer dit door 1 af te trekken van $(1+\mathbf{eps})$ en daarna 1 af te trekken van $(1+\frac{\mathbf{eps}}{3})$. Wat gebeurt er als je 1 aftrekt van $(1+0.70*\mathbf{eps})$ en hoe verklaar je dat?

Probleem 2. (Floating Point) Het programma `dumpfp` geeft de binaire floating point voorstelling van een decimaal getal terug zoals op een i386 architectuur. Het geeft dus terug hoe je computer getallen bitsgewijs bijhoudt. Gebruik `dumpfp` voor de getallen:

0.125, 0.25, 0.5, 1, 2, 4, 8 en

0.001, 0.01, 0.1, 1, 10, 100, 1000.

Welke getallen worden correct bijgehouden? Welke niet? Waarom? Hoe groot is de fout ongeveer op de voorstelling van 0.1[1]? En als je zou werken met een geheugencel waarvoor de mantisse slechts 3 bits telt?

2 Taylorreeks van e^x

De exponentiële functie kan geschreven worden als zijn Taylorreeks

$$e^x = \sum_{k=0}^{\infty} \frac{x^k}{k!}\,.$$

[1]Beschouw enkel de eerste verwaarloosde bit.

De reeks convergeert theoretisch gezien voor alle $x \in \mathbb{R}$. In wat volgt gaan we na hoe snel de reeks convergeert en hoe goed de reeks convergeert in de praktijk. We berekenen voor een bepaalde waarde van x de afgebroken Taylorreeks

$$y_n = \sum_{k=0}^{n} \frac{x^k}{k!} .$$

Dit is een goede benadering voor e^x als de termen $\frac{x^k}{k!}$ klein zijn voor $k > n$. Er geldt dan voor de absolute fout dat

$$|e^x - y_n| = O\left(\frac{x^{n+1}}{(n+1)!}\right). \tag{1}$$

Dit geeft de orde-grootte van de fout weer. In MATLAB kan je de termen van de reeks berekenen met

```
t = x.^(0:n)./factorial(0:n)
```

Met het commando `cumsum` kan je nu in één keer $y_0, y_1, \ldots, y_n$ berekenen

```
y = cumsum(t)
```

Maak een nieuw script `probleem_exp.m` waarin je de volgende opdrachten uitvoert:

Probleem 3. Bereken de benaderingen y_k, $k = 0, \ldots, n$ voor $x = 0.1$ en $n = 20$.

(a) Bekijk de vector `y'` (transpose van `y`) met `format long`. Verklaar waarom de waarde van y_k vanaf een bepaalde k niet meer wijzigt. Bekijk hiervoor ook de vector `t'` en gebruik ook eens `format long e`.

(b) Bereken de absolute en de relatieve fout van de uiteindelijke benadering.

(c) Plot de absolute en relatieve fout in functie van k. Welke schaal (`plot`, `semilogx`, `semilogy`, `loglog`) geeft het best weer hoe snel de reeks convergeert?

(d) Leg het verband tussen de grafiek en de convergentie die je theoretisch verwacht. Plot de orde-grootte van de fout in dezelfde figuur met `plot(0:n-1,t(2:n+1),'r--')`.

Probleem 4. Herhaal Probleem 2 voor $x = 0.8$. Wat is er anders? Verklaar waarom de grafiek niet volledig wordt weergegeven tot en met $n = 20$. Is de benadering beter dan voor $x = 0.1$?

Probleem 5. Herhaal Probleem 2 voor $x = 20$ en $n = 100$.

(a) De reeks begint pas na een tijd te convergeren. Verklaar.

(b) Bereken opnieuw de absolute en de relatieve fout van de uiteindelijke benadering.

(c) Is dit numeriek een goede benadering? Zo nee, wat is het probleem?

(d) Bekijk de foutenanalyse van de som van de vorige oefenzitting. Wat is de grootte-orde van de absolute fout in dit geval en komt dit overeen met wat je bekomt?

Probleem 6. Herhaal Probleem 2 voor $x = -20$ en $n = 100$. Ook hier begint de reeks pas na een tijd te convergeren.

(a) Bereken opnieuw de absolute en de relatieve fout van de uiteindelijke benadering.

(b) Is dit numeriek een goede benadering? Zo nee, wat is het probleem?

(c) Bekijk de foutenanalyse van de som van de vorige oefenzitting. Wat is de grootte-orde van de absolute fout in dit geval en komt dit overeen met wat je bekomt?

3 Benadering van een limiet

Maak een script `probleem_limiet.m` waarin je de code van volgende oefening zet.

Probleem 7. We hebben de volgende limiet

$$\lim_{x\to 0} f(x) := \lim_{x\to 0} \frac{1-\cos(x)}{x^2} = \frac{1}{2}\,.$$

(a) Ga de waarde van de limiet na met de regel van Hôpital.

(b) Evalueer $f(x)$ voor $x = 10^k$, waarbij je de exponent k in stapjes van 0.1 laat variëren tussen 0 en -10. Bekijk $f(x)$ als een kolomvector met `format long`. Wat neem je waar?

(c) Plot de absolute fout $|f(x) - 1/2|$ i.f.v. x. Welke schaal (`plot`, `semilogx`, `semilogy`, `loglog`) gebruik je hier best en waarom?

(d) Als we $\cos(x)$ schrijven als zijn Taylorreeks rond $x = 0$, $\cos(x) = 1 - x^2/2! + x^4/4! - x^6/6! - \ldots$, dan krijgen we

$$f(x) = \frac{1}{2} - \frac{x^2}{4!} + \frac{x^4}{6!} - \ldots$$

Toon aan dat de fout zich gedraagt als $g(x) = \frac{x^2}{4!}$ en ga dit na door $g(x)$ in dezelfde figuur te plotten met een rode stippellijn.

(e) In de vorige oefenzitting hebben we gezien dat de absolute fout bij het berekenen van $f(x)$ voldoet aan

$$\delta y \approx y\left(-\frac{\cos(x)}{1-\cos(x)}\epsilon_1 + \epsilon_2 - \epsilon_3 + \epsilon_4\right), \qquad |\epsilon_i| \leq \epsilon_{mach}$$

Toon aan dat door het vervangen van $\cos(x)$ door zijn Taylorreeks rond $x = 0$, de dominante term van de fout zich gedraagt als $\sim x^{-2}\epsilon_1$. Ga dit na door $x^{-2}\epsilon_{mach}$ in dezelfde figuur te plotten met een groene stippellijn.

4 Extra oefeningen

Probleem 8. (Een onschadelijke afronding?) Op 25 februari, 1991, tijdens de Golfoorlog, faalt een Amerikaans Patriot raketafweersysteem, operationeel in Dharan, Saudi Arabië, bij het opsporen en onderscheppen van een inkomende Iraakse Scud.[2] De Scud slaat vervolgens in op een legerkazerne, waarbij 28 doden en 98 gewonden vallen. De oorzaak blijkt een onnauwkeurige berekening van de tijd sinds het opstarten, door een afrondingsfout naar een fixed-point getal.

Het raketafweersysteem berekent de positie van de vijandige raket aan de hand van de vorige gemeten positie van de raket, de snelheid van de raket en de tijd. De interne klok van het systeem houdt de tijd sinds het opstarten bij in tienden van seconden als een geheel getal (bv. 250 = 25 sec). Om de nieuwe positie van de raket te berekenen moet de tijd in seconden gekend zijn als een reëel getal. Gewoon een simpele vermenigvuldiging met 0.1 dus...
Er is gegeven dat

- het systeem het getal 0.1 opslaat als een 24-bits binair fixed-point getal. (Hint: 1/8 als 6 bits binair fixed-point getal is 0.00100.)
- het systeem al 100 uur opstaat.
- een scud 1.676 km/s vliegt.

Pas dumpfp nogmaals toe op 0.1.

1. Naar welk getal wordt 0.1 afgekapt en geef een uitdrukking voor de absolute en relatieve fout.
2. Hoe groot is de fout op de berekende tijd? (*antwoord: 0.3433 sec*)
3. Hoe groot is de fout op de berekende positie?

Probleem 9. (Onschadelijke berekening?) Volgende berekeningen uitvoeren voor x

$$
\begin{array}{l}
for\ \ i = 1:40 \\
\quad x = \sqrt{x} \\
end \\
for\ \ i = 1:40 \\
\quad x = x^2 \\
end
\end{array}
$$

laat in theorie elke $x \geq 0$ ongewijzigd, tenminste wanneer er geen afrondingsfouten worden gemaakt.

[2] NAVO-codenaam voor een geleide middellange-afstandsraket van Sovjet makelij.

(a) Voer een foutenanalyse uit. (Hint: doe dit voor beide lussen afzonderlijk, waarbij je voor de tweede lus een relatieve fout δ op het gegeven zet. Deze δ stelt het effect voor van alle afrondingsfouten die gemaakt worden in de eerste lus.)

(b) Vul het script `oefening_forlus.m` aan. Wat is de relatieve fout op het eindresultaat? Komt dit overeen met de foutenanalyse?

(c) Bereken ook de relatieve fouten op de tussenresultaten, waarbij je de op machineprecisie na exacte waarden krijgt in `exacte_x.mat`. Plot deze fouten en verklaar je resultaten. Maak hierbij gebruik van een logaritmische schaal waar nodig.

(d) Stel dat er bij het berekenen van de machten in de tweede lus geen afrondingsfouten zouden optreden. Zou je dan een nauwkeurig eindresultaat krijgen?

Probleem 10. (Evaluatie van een functie) Beschouw de functie

$$f(x) = \frac{e^{x^2} - e^{-x^2}}{2x^2}.$$

(a) Evalueer deze functie voor $x = 10^{-1}, 10^{-2}, \ldots, 10^{-10}$. Vergelijk met de op machineprecisie na exacte waarden in `f_eval_exact.mat`. Plot de relatieve fout en visualiseer dat de fout toeneemt als $\mathcal{O}(x^{-2})$.

(b) Schrijf een Matlab functie om $f(x)$ wel correct te evalueren. (Hint: stelling van Taylor.)

(c) Toon m.b.v. een foutenanalyse aan dat de relatieve fout voor kleine waarden van x toeneeemt als $\mathcal{O}(x^{-2})$.

Probleem 11. (Evaluatie van een functie) Wanneer je de functie

$$f(x) = e^{2x}(1 - \tanh(x))$$

evalueert voor grote waarden van x heb je niet enkel het probleem van gevaarlijke aftrekkingen, maar tevens zal e^{2x} al vlug een overflow genereren. Herschrijf deze uitdrukking zodat beide problemen voorkomen worden.

Oefeningen Numerieke Wiskunde

Oefenzitting 4: Conditie en stabiliteit

Het doel van deze oefenzitting is het onderzoeken van de conditie van een probleem en de stabiliteit van een algoritme. Het bestreken deel van de cursus is *Hoofdstuk 2, Foutenanalyse*.

1 Theorie

1.1 Conditie van een probleem

De conditie van een probleem geeft aan hoe gevoelig de oplossing van het probleem is voor fouten op de gegevens.

Indien het probleem erin bestaat $y = f(x)$ te evalueren voor een gegeven x, dan is

$$\Delta_c y \approx f'(x)\Delta x.$$

De absolute waarde van $f'(x)$ is dus een maat voor de conditie van het probleem met betrekking tot de absolute fout. De conditie met betrekking tot de relatieve fout volgt uit de formule

$$\delta_c y \approx \frac{f'(x)x}{f(x)}\,\delta x. \tag{1}$$

Wij zullen vooral aandacht hechten aan deze laatste betrekking, omdat de relatieve fout meer zegt over de nauwkeurigheid van het resultaat (aantal juiste beduidende cijfers). Merk op dat voor dit probleem, de uitdrukkingen overeenkomen met het absoluut en relatief conditiegetal, respectievelijk:

$$\kappa_A = |f'(x)| \qquad \text{en} \qquad \kappa_R = \left|\frac{f'(x)x}{f(x)}\right|.$$

Indien het probleem erin bestaat een functie $y = f(x_1, \ldots, x_n)$ van meerdere veranderlijken te evalueren, dan is

$$\Delta_c y \approx \sum_{i=1}^{n} \frac{\partial f}{\partial x_i}(x_1, \ldots, x_n)\Delta x_i$$

$$\delta_c y = \frac{\Delta_c y}{y} \approx \sum_{i=1}^{n} \frac{\frac{\partial f}{\partial x_i} \cdot x_i}{f}\delta x_i$$

De conditie van dit probleem kan je dus nagaan door de uitdrukkingen voor δx_i uit te werken en na te gaan waar deze groot worden.

1.2 Stabiliteit van een algoritme

Om een numeriek probleem op te lossen gebruiken we een algoritme. De stabiliteit van een algoritme zegt iets over de (relatieve) fout op het eindresultaat veroorzaakt door de voortplanting van afrondingsfouten. We kunnen de stabiliteit van een algoritme dus nagaan d.m.v. een foutenanalyse zoals in de vorige oefenzitting. Herinner dat we bij een foutenanalyse reeds veronderstelden dat de gegevens machinegetallen zijn en dus exact kunnen worden voorgesteld. De reden is dat bij een stabiliteitsonderzoek enkel afrondingsfouten in rekening worden gebracht die gemaakt worden bij bewerkingen van het algoritme en **niet** de fouten op de gegevens. (Als je de fouten op de gegevens ook in rekening brengt, dan onderzoek je eigenlijk ook de conditie.)

We berekenen dus $\bar{y}$ m.b.v. een foutenanalyse zoals in de vorige oefenzitting en kijken naar de relatieve fout

$$\delta_s y = \frac{\bar{y} - y}{y}.$$

Deze relatieve fout vergelijken we met de waarde van $\delta_c y$ voor $|\delta x| \approx \epsilon_{mach}$. We maken voor deze oefenzitting een onderscheid tussen de volgende drie gevallen:

1. $|\delta_s y|$ is groot en $|\delta_c y|$ is klein (goede conditie): het algoritme is **onstabiel**.
2. $|\delta_s y|$ is groot en $|\delta_c y|$ is ongeveer even groot (slechte conditie): het algoritme is **zwak stabiel**.
3. $|\delta_s y|$ is klein: het algoritme is **voorwaarts stabiel**.

Opmerkingen:

1. **(Definitie zwak stabiel)** Een algoritme is zwak stabiel als

   $$\frac{|\delta_s y|}{|\delta_c y|} \lesssim 1,$$

   waarbij $|\delta x| \approx \epsilon_{mach}$ in (1), dus als $|\delta_s y|$ kleiner dan of ongeveer gelijk is aan $|\delta_c y(\epsilon_{mach})|$. Merk op dat voor een voorwaarts stabiel algoritme $|\delta_s y|$ klein is, en er steeds aan deze voorwaarde voldaan is. Voorwaartse stabiliteit impliceert dus zwakke stabiliteit.
2. Merk op dat we dus twee gevallen onderscheiden wanneer $|\delta_s y|$ groot is en dat dit afhangt van de conditie van het probleem.

2 Een voorbeeld.

Onderzoek de conditie van het volgende probleem: gegeven x, evalueer

$$y = f(x) = \sqrt{1+x} - 1 = \frac{x}{\sqrt{1+x}+1},$$

en onderzoek de stabiliteit van de volgende algoritmen:

$$\bar{y}_1 = \bar{f}_1(x) = fl\left(fl(\sqrt{fl(1+x)}) - 1\right),$$
$$\bar{y}_2 = \bar{f}_2(x) = fl\left(\frac{x}{fl\left(fl(\sqrt{fl(1+x)}) + 1\right)}\right).$$

- **Conditie**

$$\begin{aligned} \Delta_c y &\approx f'(x)\Delta x = \frac{1}{2\sqrt{1+x}}\Delta x \\ \delta_c y &\approx \frac{f'(x)x}{f(x)}\delta x = \frac{\sqrt{1+x}+1}{2\sqrt{1+x}}\delta x. \end{aligned}$$

  De conditie (m.b.t. de relatieve fout) is slecht als x in de buurt komt van -1 en goed voor andere waarden van x, ook voor zeer grote positieve of negatieve waarden.[1]

- **Stabiliteit** In de vorige oefenzitting zagen we dat het aandeel van de relatieve fout op het resultaat, veroorzaakt door algoritme $\bar{f}_1$, gelijk is aan

$$\delta_s y_1 \approx \frac{\sqrt{1+x}}{2(\sqrt{1+x}-1)}\epsilon_1 + \frac{\sqrt{1+x}}{\sqrt{1+x}-1}\epsilon_2 + \epsilon_3.$$

  met $|\epsilon_i| \leq \epsilon_{mach}$. $|\delta_s y_1|$ kan dus groot zijn wanneer x in de buurt ligt van 0. We besluiten hieruit het volgende:

  - Het algoritme $\bar{f}_1$ is onstabiel voor $x \approx 0$, want daar is het probleem goed geconditioneerd, maar kan $|\delta_s y_1|$ zeer groot worden.
  - Het algoritme is voorwaarts stabiel voor andere waarden van x. Dus ook voor $x \approx -1$ waar de conditie van het probleem slecht is!

  Voor $\bar{f}_2$ was het resultaat van de foutenanalyse in de vorige oefenzitting

$$|\delta_s y_2| \leq \frac{7}{2}\epsilon_{mach}$$

  en bijgevolg is algoritme $\bar{f}_2$ voorwaarts stabiel voor alle waarden van x. Dit is een voorbeeld van een stabiel algoritme en een onstabiel algoritme voor hetzelfde probleem.

[1] Conditie en stabiliteit moeten ook nagegaan worden voor $|x| \to \infty$.

3 Opgaven

Probleem 1. Onderzoek de conditie van de evaluatie van de functie

$$f(x) = 1 + 2x - \frac{1}{1+x} = \frac{(3+2x)x}{1+x}$$

en onderzoek de stabiliteit van de volgende algoritmen om de functie $f(x)$ te evalueren, waarbij je rekening houdt met het feit dat je machine met basis 2 werkt (i.e. $fl(2x) = 2x$):

eval1 <in: x; uit: y >
1. $y \leftarrow 2 * x$
2. $y \leftarrow 1 + y$
3. $z \leftarrow 1 + x$
4. $z \leftarrow \frac{1}{z}$
5. $y \leftarrow y - z$

eval2 <in: x; uit: y >
1. $y \leftarrow 2 * x$
2. $y \leftarrow 3 + y$
3. $y \leftarrow y * x$
4. $z \leftarrow 1 + x$
5. $y \leftarrow \frac{y}{z}$

Probleem 2. Onderzoek de conditie van het onderstaande probleem en de stabiliteit van het bijhorende algoritme.

(a) $y = x \sin(x)$
eval <in: x; uit: y >
1. $y \leftarrow \sin(x)$
2. $y \leftarrow x * y$

(b) $y = \sqrt{a} - \sqrt{b}$
eval1 <in: a, b; uit: f >
1. $f \leftarrow \sqrt{a} - \sqrt{b}$

eval2 <in: a, b; uit: f >
1. $f \leftarrow \dfrac{a-b}{\sqrt{a}+\sqrt{b}}$

Probleem 3. Stel dat het evalueren van $g(x)$ goed geconditioneerd is en dat je een stabiel algoritme $\bar{g}(x)$ hebt ter evaluatie. Dit betekent dat

$$g(x(1+\delta x)) = g(x)(1+C_1\delta x)$$

met C_1 een niet al te grote constante, en dat

$$\bar{g}(x) = g(x)(1+C_2\epsilon)$$

met $|\epsilon| \leq \epsilon_{mach}$ en C_2 een niet al te grote constante. Indien $|\delta x| \leq \epsilon_{mach}$, dan

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

met $|\epsilon'| \leq \epsilon_{mach}$ en C een niet al te grote constante

Toon aan dat $g'(x)$ benaderen met de volgende formule geen nauwkeurig resultaat geeft:

$$y = \frac{\bar{g}(x+h) - \bar{g}(x)}{h}$$

(Theoretisch wordt y een betere benadering naarmate $h \to 0$.) Doe hiervoor een fouten-analyse. Toon aan dat

$$\left| \frac{\bar{y} - y}{y} \right| \leq \mathcal{O}(\frac{1}{h})\, \epsilon_{mach}$$

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

Probleem 4. Onderzoek de conditie van de wortels van de kwadratische vergelijking in x

$$x^2 - 2x + t = 0$$

met betrekking tot fouten op de coëfficiënt t indien t in de buurt van 1 ligt (bv. $t = 1 - \epsilon$). Illustreer grafisch. Wat verwacht je in het algemeen over de conditie van een meervoudige wortel van een veelterm?

Probleem 5. Onderzoek de conditie van het onderstaande probleem en de stabiliteit van de bijhorende algoritmen.

(a) $y = (1+x)^2 - 1 = (2+x)x$

eval1 <in: x; uit: y>
1. $y \leftarrow 1 + x$
2. $y \leftarrow y^2$
3. $y \leftarrow y - 1$

eval2 <in: x; uit: y>
1. $y \leftarrow 2 + x$
2. $y \leftarrow y * x$

(b) $y = \dfrac{1}{e^x - 1} - \dfrac{1}{e^x + 1} = \dfrac{2}{e^{2x} - 1}$

eval1 <in: x; uit: y >
1. $z \leftarrow \exp(x)$
2. $v \leftarrow z - 1$
3. $v \leftarrow 1/v$
4. $w \leftarrow z + 1$
5. $w \leftarrow 1/w$
6. $y \leftarrow v - w$

eval2 <in: x; uit: y >
1. $y \leftarrow 2 * x$
2. $y \leftarrow \exp(y)$
3. $y \leftarrow y - 1$
4. $y \leftarrow 1/y$
5. $y \leftarrow 2 * y$

Oefeningen Numerieke Wiskunde

Oefenzitting 5 (PC): Afrondings- en benaderingsfouten

1 Numerieke differentiatie: benaderingsfouten en afrondingsfouten

Een differentiequotiënt is een eenvoudige benadering voor de afgeleide $f'(x)$ van een functie $f(x)$. In de volgende oefeningen maken we een Matlab script waarin we de afgeleide van de functie $f(x) = 5e^x$ in $x = 0$ benaderen met volgende differentieformules

$$y_1 = \frac{f(x+h) - f(x)}{h}, \tag{1}$$

$$y_2 = \frac{f(x+h) - f(x-h)}{2h}. \tag{2}$$

We zullen beide formules evalueren voor steeds kleiner wordende waarden van h

```
n = -1:-0.1:-15
h = 10.^n
```

Maak een script `numdiff.m` waarin je de volgende opdrachten uitvoert.

Probleem 1. Reken beide formules uit voor de gegeven waarden van h.

(a) Bereken de relatieve fout, waarbij je vergelijkt met de exacte waarde van $f'(0)$.

(b) Bekijk de fout in de Command Window met `format long`. Wat valt je op? Wat is het maximum aantal juiste beduidende decimale cijfers dat je bekomt voor elke formule?

(c) Plot de fout voor beide formules in dezelfde figuur, in functie van de waarde van h of in functie van de waarde van de exponent n. Gebruik een geschikte schaal voor de assen.

(d) Welke componenten van de fout (benadering, afronding, etc.) neem je waar?

Probleem 2. (Orde van de benaderingsfout) Een benadering $g(h;x)$ voor $f'(x)$ is van orde k als

$$f'(x) = g(h;x) + O(h^k), \quad \text{voor} \quad h \to 0. \tag{3}$$

Dit betekent dat de absolute fout van de benadering afneemt zoals h^k wanneer $h \to 0$.

(a) Lees van de grafiek de orde van beide benaderingen af.

(b) Plot in dezelfde figuur de grafiek van h^k met een streepjeslijn (`'--'`), met k de orde van elke benadering.

(c) Geef een wiskundig bewijs van je observaties, gebruik makende van een Taylorreeksontwikkeling. (Hint: schrijf voor (2) een ontwikkeling voor $f(x+h)$ en $f(x-h)$.)

Probleem 3. (Afrondingsfouten) In de vorige oefenzitting heb je aangetoond dat voor formule (1), de relatieve fout op het berekende resultaat $\bar{y}_1$ door afrondingsfouten zich gedraagt als

$$\left| \frac{\bar{y}_1 - y_1}{y_1} \right| \leq \mathcal{O}(\frac{1}{h})\, \epsilon_{mach} \tag{4}$$

(a) Plot in dezelfde figuur de grafiek van $h^{-1}\epsilon_{mach}$ met een streepjeslijn.

(b) Toon aan dat ook voor formule (2), de relatieve fout door afrondingsfouten zich gedraagt als (4).

(c) Wat weet je nu over de differentieformules? Kan je deze gebruiken om de afgeleide van een functie te benaderen tot op machineprecisie? Welke formule zou je verkiezen?

2 Conditie en stabiliteit van de kwadratische vergelijking

Voor het uitwerken van deze opgaven heb je de '.m'-bestanden nodig die je kan vinden op Toledo.

Probleem 4. (Vierkantsvergelijking) Gebruik het bestand `vierkant.m` om de wortels van een vierkantsvergelijking te berekenen. Ga na hoe beide algoritmes de wortels berekenen en pas die dan toe op

$$x^2 - bx + 2 = 0,$$

voor $b = 10^7, 10^8, 10^9$. Verklaar de verschillen. Welk algoritme is het nauwkeurigst?

Probleem 5. (Wortel van een complex getal) Maak gebruik van `vierkant.m` om de vierkantswortel $x + iy$ van een complex getal $a + ib$ te berekenen. Hint: $a + ib = (x + iy)^2$ en stel reële en imaginaire delen gelijk aan mekaar.

Schrijf hiervoor een functie `[x1,y1,x2,y2] = cwortel(a,b)` waarbij de output overeenkomt met de verschillende versies van `vierkant.m`. Bereken nu de vierkantswortel van $2 - 3i$, $-2 \cdot 10^3 - 3 \cdot 10^{-3}i$ en $2 \cdot 10^3 - 3 \cdot 10^{-3}i$ en vergelijk het resultaat van beide versies met de output van het Matlab commando `sqrt`. (Noot: de output van `sqrt` heeft altijd een positief reëel deel.) Voor welk getal en voor welke versie krijg je een grote relatieve fout en waarom enkel voor dit getal?

Probleem 6. (Conditieonderzoek: dubbel nulpunt) Beschouw het probleem van het zoeken van de nulpunten van de vierkantsvergelijking

$$x^2 - 2x + t = 0.$$

Onderzoek de conditie m.b.t. fouten op de parameter t. Gebruik `vierkant.m` om voor $t \approx 1$ de nulpunten te berekenen en deze te vergelijken met de nulpunten wanneer je een kleine perturbatie op t aanbrengt. Neem bv. $t_1 = 1 - 1 \cdot 10^{-10}$ en $t_2 = t_1 + 1 \cdot 10^{-14}$. Verklaar de relatieve fouten op de berekende nulpunten. Maak ook een grafiek van het (relatief) conditiegetal en van de nulpunten voor $t \in [0, 1]$. Verklaar de resultaten.

Oefeningen Numerieke Wiskunde

Oefenzitting 6: Veelterminterpolatie en numerieke integratie

Het doel van deze oefenzitting is de theorie en de algoritmische aspecten van veeltermsinterpolatie en de theorie over numerieke integratie inoefenen. Het bestreken deel van de cursus is *Hoofdstuk 4 - Veelterminterpolatie* en *Hoofdstuk 6 - Numerieke integratie*.

1 Veelterminterpolatie

Theorie

Gegeven een tabel $\{(x_i, f_i)\}_{i=0}^n$, met $x_i \neq x_j$ als $i \neq j$. De **unieke** interpolerende veelterm van graad n kan op verschillende manieren geschreven worden. In de Lagrange-basis wordt ze geschreven als

$$y_n(x) = l_0(x)f_0 + l_1(x)f_1 + \ldots + l_n(x)f_n,$$

waarbij

$$l_i(x) = \frac{\prod_{k=0,k\neq i}^n (x - x_k)}{\prod_{k=0,k\neq i}^n (x_i - x_k)}$$

de i-de Lagrangeveelterm van graad n is. In de Newton-basis krijgen we

$$y_n(x) = b_0 + b_1(x - x_0) + \ldots + b_n(x - x_0)(x - x_1)\ldots(x - x_{n-1}),$$

waarbij

$$b_i = f[x_0, x_1, \ldots, x_i]$$

de gedeelde differentie is van orde n. In de klassieke basis krijgen we

$$y_n(x) = a_0 + a_1x + a_2x^2 + \ldots + a_nx^n$$

waarbij de coëfficiënten de oplossing zijn van het Vandermonde stelsel

$$\begin{bmatrix} 1 & x_0 & \cdots & x_0^n \\ 1 & x_1 & \cdots & x_1^n \\ \vdots & \vdots & & \vdots \\ 1 & x_n & \cdots & x_n^n \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ \vdots \\ a_n \end{bmatrix} = \begin{bmatrix} f_0 \\ f_1 \\ \vdots \\ f_n \end{bmatrix}.$$

Opgaven

Probleem 1. Beschouw de 2^e-graadsveelterm $f(x) = 2x^2 + 4x - 5$.

(a) Bereken de interpolerende veelterm $y_1(x)$ in de punten $\{-1, 1\}$ met de methode van Lagrange en met de methode van Newton.

(b) Stel het Vandermonde stelsel op en ga na dat je oplossing hieraan voldoet.

(c) Bereken de interpolatiefout $E_1(x)$ met formule (4.10) uit het handboek en maak een grafiek van deze fout. Hoe gedraagt de fout zich buiten het interval $[-1, 1]$?

Probleem 2. Beschouw dezelfde 2^e-graadsveelterm $f(x) = 2x^2 + 4x - 5$ en herhaal Probleem 1 met de interpolatiepunten $\{-1, 0, 1\}$ waarbij je voor de methode van Newton de twee soorten tabellen van gedeelde differenties opstelt. Wat is de waarde van $E_2(x)$? Verklaar.

Probleem 3. Beschouw dezelfde 2^e-graadsveelterm $f(x) = 2x^2 + 4x - 5$ en herhaal Probleem 1 met de interpolatiepunten $\{-1, 0, 0.5, 1\}$. Gebruik enkel de methode van Newton en stel de twee soorten tabellen van gedeelde differenties op, waarbij je de tabellen uit Probleem 2 hergebruikt. Verklaar je resultaat.

Probleem 4. Bewijs

1.
$$\sum_{i=0}^{n} l_i(x) = 1, \quad \forall x.$$

2.
$$\sum_{i=0}^{n} l_i(x) {x_i}^k = x^k, \quad k \leqslant n.$$

Probleem 5. Bewijs

$$y_n(x) = y_{n-1}(x) + (f_n - y_{n-1}(x_n)) l_n(x).$$

(Hint: controleer of er aan de interpolatievoorwaarden voldaan is.)

Probleem 6*. Hieronder staat een tabel met gedeelde differenties van een veelterm $f(x)$ van de derde graad, cfr. Tabel 4.1 in het handboek. Eén van de functiewaarden is echter verkeerd. Welke functiewaarde? Wat is de absolute fout op deze functiewaarde?
(Hint: Gedeelde differenties zijn lineair in de functiewaarden f_i. Waarom? Het effect van een fout op een functiewaarde kan dus apart onderzocht worden. Stel een tabel op waarbij $f_k = \epsilon$ en alle andere functiewaarden gelijk zijn aan 0.)

x_i	f_i			
0.00	0.8622			
		0.1127		
1.00	0.9749		3.0705	
		4.1044		0.5377
1.30	2.2062		3.8770	
		6.0429		0.5377
1.50	3.4148		4.1996	
		7.3028		0.5377
1.60	4.1450		4.3878	
		7.9609		123.9377
1.65	4.5431		29.1753	
		10.8785		-246.2623
1.70	5.0870		-20.0771	
		7.8669		123.9377
1.80	5.8737		10.9073	
		10.0484		-20.0290
1.90	6.8785		4.8986	
		11.0281		0.5377
2.00	7.9814		5.1137	
		12.5622		
2.20	10.4938			

Probleem 7*. Bewijs dat $l_i(x)$ ook kan geschreven worden als

$$l_i(x) = \frac{\pi(x)}{\pi'(x_i)(x - x_i)}$$

met $\pi(x) = (x - x_0)(x - x_1)\dots(x - x_n)$.

Probleem 8*. (Interpolatie volgens Neville)

1. Bewijs dat
$$y_{012} = \frac{x - x_2}{x_0 - x_2} y_{01}(x) + \frac{x - x_0}{x_2 - x_0} y_{12}(x).$$
de interpolerende veelterm is in de punten $\{(x_i, f_i)\}_{i=0}^{2}$.

2. Bewijs de algemene formule:
$$y_{i,\dots,j}(x) = \frac{x - x_i}{x_j - x_i} y_{i+1,\dots,j}(x) + \frac{x - x_j}{x_i - x_j} y_{i,\dots,j-1}(x).$$

3. Bereken het aantal bewerkingen in het algoritme van Neville.

4. Hoe zou je het aantal bewerkingen kunnen verminderen en hoeveel bewerkingen moet je dan minder doen?

(Hint: Ga eens na welke bewerkingen meer dan één keer worden gedaan en hoe je dit kan vermijden.)

2 Numerieke integratie

Theorie

De bepaalde integraal van een functie f over een interval $[a, b]$ wordt benaderd door een **kwadratuurformule**. Dit is een gewogen som van de functiewaarden

$$\int_a^b f(x)dx \approx H_0 f(x_0) + H_1 f(x_1) + \cdots + H_n f(x_n). \tag{1}$$

De punten x_k worden de abscissen van de kwadratuurformule genoemd en de coëfficiënten H_k de gewichten.

De kwadratuurformule wordt **interpolerend** genoemd indien aan de volgende, onderling equivalente voorwaarden voldaan is:

1. Voor elke functie f geldt

$$H_0 f(x_0) + H_1 f(x_1) + \cdots + H_n f(x_n) = \int_a^b y_n(x)dx \tag{2}$$

   waarbij $y_n(x)$ de interpolerende veelterm van graad $\leq n$ voorstelt door de punten $(x_k, f(x_k)),\ k = 0, 1, \ldots, n$.

2. Voor elke veelterm p van graad $\leq n$ geldt

$$H_0 p(x_0) + H_1 p(x_1) + \cdots + H_n p(x_n) = \int_a^b p(x)dx. \tag{3}$$

3. De gewichten van de kwadratuurformule zijn gelijk aan de integralen van de Lagrangeveeltermen

$$H_k = \int_a^b l_k(x)dx \quad k = 0, 1, \ldots, n. \tag{4}$$

We zeggen dat de **nauwkeurigheidsgraad** van de kwadratuurformule (1) gelijk is aan d indien

$$H_0 p(x_0) + H_1 p(x_1) + \cdots + H_n p(x_n) = \int_a^b p(x)dx, \tag{5}$$

voor elke veelterm p van graad $\leq d$ en indien er een veelterm van graad $d + 1$ bestaat waarvoor deze gelijkheid niet meer geldt. Vermits elke veelterm kan geschreven worden als

een lineaire combinatie van de eentermen $1, x, x^2, \ldots$, komt dit neer op de voorwaarden

$$\begin{aligned} H_0 + H_1 + \cdots + H_n &= \int_a^b dx \\ H_0 x_0 + H_1 x_1 + \cdots + H_n x_n &= \int_a^b x dx \\ &\cdots \\ H_0 x_0^d + H_1 x_1^d + \cdots + H_n x_n^d &= \int_a^b x^d dx \\ H_0 x_0^{d+1} + H_1 x_1^{d+1} + \cdots + H_n x_n^{d+1} &\neq \int_a^b x^{d+1} dx \end{aligned}$$

Opgaven

Probleem 9. Toon de onderlinge equivalentie aan van de 3 voorwaarden opdat een kwadratuurformule interpolerend zou zijn. (Hint: 1. $\Rightarrow$ 2. $\Rightarrow$ 3. $\Rightarrow$ 1.).

Probleem 10. Bepaal de gewichten van de kwadratuurformule

$$\int_{-1}^{1} f(x)dx \approx H_0 f(-\frac{1}{2}) + H_1 f(\frac{1}{2})$$

zo dat haar nauwkeurigheidsgraad zo hoog mogelijk is.
Hoe hoog is de nauwkeurigheidsgraad? Is de kwadratuurformule interpolerend?

Probleem 11. Bepaal de gewichten $H_{-\frac{1}{2}}, H_0$ en $H_{\frac{1}{2}}$ zodanig dat de kwadratuurformule

$$\int_{a-h}^{a+h} f(x)dx \approx H_{-\frac{1}{2}} f(a - \frac{h}{2}) + H_0 f(a) + H_{\frac{1}{2}} f(a + \frac{h}{2})$$

een zo hoog mogelijke nauwkeurigheidsgraad heeft. Hoe hoog is de nauwkeurigheidsgraad?

Probleem 12*. Bepaal de gewichten van de kwadratuurformule

$$\int_{-1}^{1} f(x)dx \approx H_{-1} f(-1) + H_0 f(0) + H_1 f(1) + H'_{-1} f'(-1) + H'_1 f'(1)$$

zodat deze een zo hoog mogelijke nauwkeurigheidsgraad heeft. Hoe hoog is de nauwkeurigheidsgraad?

Oefeningen Numerieke Wiskunde

Oefenzitting 7 (PC): Het oplossen van stelsels lineaire vergelijkingen

In deze oefenzittingen ga je m.b.v. Matlab aspecten zoals conditie, stabiliteit en complexiteit onderzoeken voor stelsels vergelijkingen. Het relevante deel van de cursus is *Hoofdstuk 3 'Stelsels lineaire vergelijkingen'*, de slides over QR, SVD en kleinste kwadraten benaderingen en de slides over het creëren van nullen in een matrix. Voor het uitwerken van de opgaven gebruik je de `.m`-bestanden die je kan vinden op Toledo.

1 MATLAB–functies voor deze oefenzitting

Genereren van matrices

De volgende functies genereren een stelsel $Ax = b$, waarvan de oplossingsvector x bestaat uit natuurlijke getallen. Ieder commando genereert een matrix $M = [A\ b]$ met A een $(n \times n)$–matrix en b een $(n \times 1)$–vector.

- `M = genmatrix1(n)` : De matrix A is een goed geconditioneerde random matrix.
- `M = genmatrix2(n)` : De matrix A is een goed geconditioneerde random matrix met een specifiek permutatiegedrag.
- `M = genmatrixc(n)` : De matrix A is een slecht geconditioneerde random matrix.

Stelsels oplossen

- `G = gauss1(M)` waarbij $M = [A\ b]$ met A een $n \times n$–matrix en b een $n \times 1$–vector. Deze functie maakt de matrix

$$\left(\begin{array}{c|c|c} & & 1 \\ A & b & \vdots \\ & & n \end{array}\right)$$

  aan en past Gauss–eliminatie toe (zie handboek, Algoritme 3.4).

- `G = gauss2(M)` is analoog aan `gauss1`, maar met optimale rij–pivotering (zie handboek, Algoritme 3.5). Merk op dat dit equivalent is met `A\b` in Matlab voor vierkante matrices A.

- `[Q,R] = qr(M)` berekent een QR-factorisatie van de matrix M.
- `x = asubst(G)` voert achterwaartse substitutie uit op het resultaat van `gauss1` of `gauss2` (`x = asubst(R)` resp. op het resultaat van `qr`), en geeft de oplossing x van het stelsel $Ax = b$.

2 Conditie en achterwaartse stabiliteit

Een algoritme $\hat{F}(x)$ om $F(x)$ te berekenen is achterwaarts stabiel als

$$\hat{F}(x) = F(x + \Delta x), \quad \text{én} \quad \frac{||\Delta x||}{||x||} \leq C \cdot \epsilon_{mach}.$$

Dit betekent dat het effect van de voortplanting van afrondingsfouten gemaakt in $\hat{F}(x)$ kan worden teruggebracht tot een kleine relatieve perturbatie op de gegevens. In het geval van het oplossen van stelsels is een algoritme $\hat{x}(A, b)$ dus achterwaarts stabiel als

$$(A + \Delta A)\hat{x} = b + \Delta b, \quad \text{met} \quad \frac{||\Delta A||}{||A||} \leq C_1 \cdot \epsilon_{mach}, \frac{||\Delta b||}{||b||} \leq C_b \cdot \epsilon_{mach}. \tag{1}$$

We zullen in deze oefenzitting stelsels oplossen met verschillende algoritmes en niet enkel de relatieve fouten $\|\hat{x} - x\| / \|x\|$ op de berekende $\hat{x}$ vergelijken, maar ook de residu's

$$r = A\hat{x} - b.$$

Als het residu r klein is t.o.v. het rechterlid b, dan geldt (1) met $\Delta A = 0$ en $\Delta b = r$ en dan is het algoritme dus achterwaarts stabiel voor de gebruikte gegevens. Ga dit na!

In het algemeen geldt er voor een achterwaarts stabiel algoritme dat

$$\frac{||\hat{F}(x) - F(x)||}{||F(x)||} = \frac{||F(x + \Delta x) - F(x)||}{||F(x)||} \leq \kappa_F \frac{||\Delta x||}{||x||} \overset{(*)}{\leq} C \cdot \kappa_F \cdot \epsilon_{mach},$$

dus de voorwaartse fout is begrensd door het conditiegetal en de machine-nauwkeurigheid. Merk op dat de achterwaartse stabiliteit eigenlijk enkel de laatste ongelijkheid $(*)$ impliceert. De gelijkheid definieert gewoon de achterwaartse fout Δx en de eerste ongelijkheid volgt uit de definitie van het conditiegetal. Ga na dat, in het geval van het oplossen van stelsels, er uit de definitie van de achterwaartse fout en het conditiegetal van een matrix volgt dat

$$\frac{||\hat{x} - x||}{||x||} \leq \kappa_A \left(\frac{||\Delta b||}{||b||} + \frac{||\Delta A||}{||A||} \right), \tag{2}$$

en dat bijgevolg voor een achterwaarts stabiel algoritme geldt dat

$$\frac{||\hat{x} - x||}{||x||} \leq C \cdot \kappa_A \cdot \epsilon_{mach}.$$

3 Oefeningen

Probleem 1. (De LU ontbinding)

(a) Stel

$$A = \begin{pmatrix} 7 & 8 & 0 \\ 1 & 2 & 3 \\ 4 & 5 & 6 \end{pmatrix}.$$

Bereken de LU-ontbinding van A met `[L,U] = lu(A)`. Wat is $L*U$? Geef de Matlab-bevelen om uit L en U de determinant van A te berekenen.

(b) Stel

$$A = \begin{pmatrix} 1 & 4 & 3 \\ 4 & 3 & 5 \\ 9 & 8 & 0 \end{pmatrix}.$$

Bereken opnieuw de LU-ontbinding van A. Wat merk je op als je de matrices L en U bekijkt? Geef een verklaring voor wat er gebeurd is. (Hint: `doc lu`.)

Probleem 2. (Gauss-eliminatie) Gebruik de functie `genmatrix1` om een 6×7–matrix $M = [A\ b]$ te genereren. Los het stelsel $Ax = b$ op m.b.v. de functies `gauss1`, `gauss2` en `qr`. Je bekomt dus drie oplossingen voor hetzelfde stelsel.

(a) Bereken de relatieve fout van de oplossingen en de verhouding van de residu's t.o.v het rechterlid b, indien je weet dat de exacte oplossing een vector met natuurlijke getallen is. Vul in de onderstaande tabel de grootte-orde (bvb. 10^3, 10^6, 10^{11}, ...) van de fouten en de relatieve residu's in. (Hint: gebruik `norm`.)

(b) Wat kan je zeggen over de stabiliteit van de methodes?

Doe nu hetzelfde, maar met de functie `genmatrix2`.

(c) Wat is het verschil t.o.v. je resultaten met het eerste stelsel? Verklaar.

(d) Wat kan je nu zeggen over de stabiliteit van de methodes?

(Tip: schrijf een Matlab script zodat je gemakkelijk je resultaten kan herberekenen. Deze zullen telkens anders zijn, want we werken met random matrices. De grootte-ordes zullen echter redelijk gelijk blijven.)

Probleem 3. (Conditie en achterwaartse stabiliteit) Voer opnieuw (a) uit van de vorige oefening, maar nu voor de functie `genmatrixc`. Vul de tabel verder aan.

(a) Bereken met `cond` of met `norm` het 2-norm conditiegetal van de matrix A voor de drie verschillende stelsels. Vul telkens de grootte-orde in de tabel in.

(b) Ga na dat de relatieve fouten van de oplossingen en de relatieve residu's voldoen aan formule (2).

(c) Wat is je algemene besluit over de stabiliteit van de methodes?

	$\kappa_2(A)$		gauss1 gauss2 qr
genmatrix1		$\|\hat{x}-x\|_2 / \|x\|_2$ $\|\|r\|\|_2/\|\|b\|\|_2$	
genmatrix2		$\|\hat{x}-x\|_2 / \|x\|_2$ $\|\|r\|\|_2/\|\|b\|\|_2$	
genmatrixc		$\|\hat{x}-x\|_2 / \|x\|_2$ $\|\|r\|\|_2/\|\|b\|\|_2$	

Probleem 4. (Implementatie-opdracht) Schrijf een functie `[Q,R]=qrstep(M)`, die een QR-factorisatie berekent van een matrix $M = [A\ B]$ van de volgende structuur:

$$M = \left[\begin{array}{ccccc|ccc} a_{1,1} & a_{1,2} & \cdots & a_{1,n-1} & a_{1,n} & b_{1,1} & \cdots & b_{1,p} \\ 0 & a_{2,2} & & a_{2,n-1} & a_{2,n} & b_{2,1} & & b_{2,p} \\ \vdots & & \ddots & \vdots & \vdots & \vdots & & \vdots \\ 0 & \cdots & 0 & a_{n-1,n-1} & a_{n-1,n} & b_{n-1,1} & \cdots & b_{n-1,p} \\ a_{n,1} & \cdots & a_{n,n-2} & a_{n,n-1} & a_{n,n} & b_{n,1} & \cdots & b_{n,p} \end{array}\right]$$

De implementatie moet gebeuren met maximaal $n-1$ Givens rotaties. Een Givens rotatie kan berekend worden met de standaard Matlab functie `planerot`.

Test je algoritme (is Q orthogonaal, R bovendriehoeks en is $M = Q * R$?). Vergelijk met het resultaat van de Matlab functie `qr`. Merk op dat dit niet noodzakelijk hetzelfde is, waaruit nogmaals blijkt dat de QR-factorisatie essentieel uniek is. D.w.z. dat de kolommen van Q uniek zijn op het teken na. Wat betekent dit voor de rijen van R?

Hints:

- `A([1 3],:) = G*A([1 3],:)` past de Givens rotatie G toe op de eerste en derde rij van de matrix A.
- Een matrix M met de bovenstaande structuur kan je bv. genereren met

```
M = triu(rand(n,n+p));
M(n,1:n-1) = rand(1,n-1);
```

Probleem 5*. (QR-factorisatie) Door je algoritme $n-1$ keer toe te passen kan je een QR-factorisatie van een willekeurige $n \times (n+p)$-matrix berekenen. Schrijf zulke routine. Nu heb je een alternatief voor het schema uit je cursus, met hetzelfde aantal Givens rotaties.

Oefeningen Numerieke Wiskunde

Oefenzitting 8 (PC): Programmeren in MATLAB

Meer over functies, debuggen en plotten

Op Toledo vind je de benodigde MATLAB bestanden.

1 Functies, function handles en anonieme functies

Probleem 1. Schrijf een functie `evalueer_lagrange.m` waarin je Algoritme 4.1 uit het handboek implementeert om de interpolerende veelterm volgens Lagrange op te stellen en te evalueren. De functie heeft volgende signatuur:

```
y = evalueer_lagrange( x,f,z )
```

waarbij `x` en `f` vectoren zijn met interpolatiepunten en functiewaarden, en `z` een vector met waarden waarin de veelterm geëvalueerd moet worden.

Test of je functie werkt m.b.v. het script `test_lagrange.m`.

Probleem 2. (Function handles)

(a) Hernoem de voorgaande functie tot `evalueer_lagrange2.m` en pas deze aan zodat de invoer `f` nu een function handle is i.p.v. een vector met functiewaarden. (Zie `help function_handle` voor meer informatie, bekijk zeker de voorbeelden).

(b) Schrijf een functie `f1.m` die $f(x) = 1/(1 + 25x^2)$ implementeert.

(c) Hernoem nu `test_lagrange.m` tot `test_lagrange2.m` en pas dit script aan zodat je overal `f1.m` en `evalueer_lagrange2.m` gebruikt.

Probleem 3. (Anonieme functies) Je kan i.p.v. functies te maken met een m-bestand ook anonieme functies gebruiken. Als je bijvoordeeld een van de volgende regels invoert

```
f = @exp
f = @(x) exp(x)
```

dan kan je deze functie evalueren in een vector `z` met het commando `y = f(z)`.

Hernoem het bovenstaande script tot `test_lagrange3.m`. Maak een anonieme functie voor $f(x) = 1/(1 + 25x^2)$ en gebruik deze i.p.v. `f1.m`.

2 De MATLAB debugger

Probleem 4. De Collatz functie van een positief natuurlijk getal n is een sequentie die begint bij n en eindigt bij 1. Gegeven een getal g_i in deze sequentie kan het volgende getal g_{i+1} als volgt bepaald worden:

1. g_i is even: $g_{i+1} = \frac{g_i}{2}$
2. g_i is oneven: $g_{i+1} = 3g_i + 1$

Ga naar `http://www.mathworks.nl/help/matlab/debugging-code.html` en klik op 'Ways to Debug MATLAB Files'. In deze handleiding wordt het debug-proces in MATLAB en enkele eigenschappen uitgelegd aan de hand van de bestanden `collatz.m` en `collatzplot.m`, die je met behulp van de volgende commando's in MATLAB kan openen:

```
open(fullfile(matlabroot,'help','techdoc','matlab_env','examples',
    'collatz.m'))
open(fullfile(matlabroot,'help','techdoc','matlab_env','examples',
    'collatzplot.m'))
```

Bewaar een kopie van deze bestanden in je Current Directory in MATLAB. De functie `collatz` berekent voor een gegeven getal de Collatz sequentie en de functie `collatzplot` tekent voor verschillende waarden het aantal elementen in de Collatz sequentie. Er is echter een (stomme) fout gemaakt bij het schrijven van deze programma's.

Probleem 5. Maak gebruik van de MATLAB debugger om de verschillende waarden die de teller `i` in deze lus aanneemt te bepalen:

```
for i=200:-25:100
    x=i+1;
    a=-2*x+5^3+400;
    i=a;
end
```

Probleem 6. (Tracken van de hoeksnelheid: vind de fout!) Bekijk het script `filming_the_rocket.m` en voer het uit. De positie van een raket wordt weergegeven met daaronder de benaderde hoeksnelheid van een camera in de oorsprong die de raket moet volgen. Gebruik de debugger om het vreemde gedrag van de hoeksnelheid te verklaren. Corrigeer de code zodat de hoeksnelheid wel juist wordt weergegeven.

3 Plotten met de juiste schaal

Het plotten van een vector `y` t.o.v. een vector `x` doe je met `plot(x,y)`. Het is echter vaak handig om een logaritmische schaal op de x-as en/of y-as te gebruiken. Je kan dit uiteraard doen met `plot(x,log10(y))` en varianten, maar MATLAB heeft hier de ingebouwde functies `semilogx`, `semilogy` en `loglog` voor. Het voordeel van deze functies is dat de waarden die bij de assen staan de oorspronkelijke waarden zijn en niet de logaritmen. Vergelijk maar eens de volgende plot-commando's:

```
x = 1:20;
y = exp(x);
plot(x,log10(y))
semilogy(x,y)
```

Probleem 7. (Veeltermgedrag) Beschouw de veelterm $p(x) = a_1x + a_2x^2 + \ldots + a_nx^n$, en dus $p(0) = 0$. De veelterm $p(x)$ is asymptotisch equivalent met de volgende functies

$$\begin{aligned} p(x) &\sim a_n x^n, \qquad && x \to \infty \\ p(x) &\sim a_1 x, && x \to 0 \end{aligned}$$

(a) Genereer voor $n = 6$ de coëfficiënten a_i van de veelterm $p(x)$ met `rand`.

(b) Plot de veelterm $p(x)$ in een bereik van heel kleine tot heel grote waarden van x. (Hint: gebruik `polyval`.) Kies hierbij zelf hoe je de vector `x` maakt en kies de schaal voor de assen die het meeste informatie geeft.

(c) Plot ook de functies a_1x en a_nx^n in stippellijn. Moet x heel groot en heel klein worden om het asymptotisch gedrag met het blote oog waar te nemen?

Probleem 8. (Exponentieel gedrag) Beschouw de functie $f(x) = 0.1^x + 0.01^x + 0.001^x$.

(a) Plot de functie $f(x)$ in een bereik van negatieve en positieve waarden van x. Kies hierbij zelf hoe je de vector `x` maakt en kies de schaal voor de assen die het meeste informatie geeft.

(b) Plot ook de functies waarmee $f(x)$ asymptotisch equivalent is voor $x \to \infty$ en $x \to -\infty$ in stippellijn. Moet $|x|$ heel groot worden om het asymptotisch gedrag met het blote oog waar te nemen?

Oefeningen Numerieke Wiskunde

Oefenzitting 9 (PC): Splines

Op Toledo vind je de benodigde MATLAB bestanden. Maak voor deze oefenzitting een script `oef9.m` waarin je alle code van de verschillende oefeningen bundelt. Dit is handig als je nadien oplossingen wil vergelijken. Maak eventueel verschillende secties in je script met `%%`.

1 Splinefunctie in de klassieke basis

Gegeven zijn de reële knooppunten $a = t_0 < t_1 < \ldots < t_n = b$. Een kubische splinefunctie $s(x)$ (of een splinefunctie van graad 3) is een functie die:

- in elk deelinterval $[t_j, t_{j+1})$ gelijk is aan een veelterm $p_j(x)$ van graad 3;
- in elk knooppunt t_j continu is in zijn 0^e, 1^e en 2^e afgeleide.

Een eenvoudige manier om een splinefunctie te definiëren is d.m.v. de veeltermcoëfficiënten in de klassieke basis $\{1, x, x^2, x^3\}$ in elk deeltinterval:

$$s(x) = p_j(x) = a_j x^3 + b_j x^2 + c_j x + d_j, \quad x \in [t_j, t_{j+1}), \quad j = 0, \ldots, n-1. \tag{1}$$

Om te kunnen spreken van een splinefunctie moeten deze coëfficiënten voldoen aan de continuïteitsvoorwaarden:

$$\begin{aligned} p_{j-1}(t_j) &= p_j(t_j) \\ p'_{j-1}(t_j) &= p'_j(t_j) \\ p''_{j-1}(t_j) &= p''_j(t_j) \end{aligned} \qquad \text{voor } \; j = 1, \ldots, n-1 \tag{2}$$

Als we de coëfficiënten voorstellen in de kolomvector

$$c = [a_0 \;\; b_0 \;\; c_0 \;\; d_0 \;\; a_1 \;\; b_1 \;\; c_1 \;\; d_1 \;\; \cdots \;\; a_{n-1} \;\; b_{n-1} \;\; c_{n-1} \;\; d_{n-1}]^T, \tag{3}$$

dan kunnen we de voorwaarden (2) schrijven als een stelsel

$$Ac = 0.$$

Probleem 1.

(a) Schrijf de voorwaarden (2) uit in functie van de veeltermcoëfficiënten.

(b) Bestudeer de functie `splinematrix.m`. Controleer dat elke rij van de output matrix A in het stelsel $Ac = 0$ overeenkomt met een continuïteitsvoorwaarde.

(c) Voer de functie uit voor `t=linspace(-1,1,10)`, en bekijk de matrix A met het commando `spy`.

(d) Bereken met `size` de dimensies van A en met `rank` de rang. Controleer dat de dimensie van de nulruimte van A gelijk is aan $n+3$. Merk op dat de rang gelijk is aan het aantal rijen, want de continuïteitsvoorwaarden zijn lineair onafhankelijk.

Dit toont aan dat voor $n+1$ knooppunten, de dimensie van de ruimte van kubische splinefuncties gelijk is aan $n+3$. Als we dus nog $n+3$ voorwaarden opleggen, dan bekomen we een unieke splinefunctie. Indien we een interpolatieprobleem oplossen, dan krijgen we de volgende $n+1$ interpolatievoorwaarden:

$$s(t_j) = f_j, \qquad j = 0, \ldots, n. \tag{4}$$

Deze worden aangevuld met nog twee voorwaarden, bijvoorbeeld de natuurlijke voorwaarden:

$$s''(t_0) = 0, \qquad s''(t_n) = 0. \tag{5}$$

Probleem 2.

(a) Schrijf de interpolatievoorwaarden en de natuurlijke voorwaarden uit in functie van de veeltermcoëfficiënten.

(b) Bestudeer de functie `splinestelsel.m` met als output een matrix `A` en een vector `b`. Controleer dat elke rij van het stelsel $Ac = b$ overeenkomt met een continuïteitsvoorwaarde, een interpolatievoorwaarde of een natuurlijke voorwaarde.

(c) Voer de functie uit voor `t=linspace(-1,1,10)` en `f=@exp`, en bekijk opnieuw de matrix A met het commando `spy`.

Probleem 3.

(a) Schrijf een functie met signatuur

```
y = evalspline(t, c, x)
```

die gegeven de knooppunten `t` en de veeltermcoëfficiënten `c` zoals in (3), de splinefunctie evalueert in de punten van de vector `x`. (Hint: gebruik `polyval` voor het evalueren van de veeltermen.)

(b) Stel het stelsel $Ac = b$ op voor `t=-1:0.25:1` en $f(x) = 1/(1+25x^2)$. Bereken de coëfficiënten van de splinefunctie met `c = A\b`. Plot $f(x)$ en de splinefunctie voor $x = [-1, 1]$ samen in een figuur.

2 Splinefunctie met B-splines

In wat volgt stellen we een kubische splinefunctie voor als een lineaire combinatie van $n+3$ genormaliseerde B-splines

$$s(x) = \sum_{j=-3}^{n-1} c_j N_{j,3}(x). \tag{6}$$

Vanaf nu laten we het subscript van de graad weg: $N_j(x) := N_{j,3}(x)$.

De genormaliseerde B-splines $N_j(x)$ zijn lineair onafhankelijk en voldoen aan de continuïteitsvoorwaarden (2). Ze vormen bijgevolg een basis voor de ruimte van kubische splinefuncties. Hierdoor hoeven er slechts $n+3$ coëfficiënten bepaald te worden op basis van de interpolatievoorwaarden (4) en nog twee bijkomende voorwaarden. Met de natuurlijke voorwaarden (5) bekomen we het volgende stelsel:

$$\begin{bmatrix} N''_{-3}(t_0) & N''_{-2}(t_0) & N''_{-1}(t_0) & & \\ N_{-3}(t_0) & N_{-2}(t_0) & N_{-1}(t_0) & & \\ & N_{-2}(t_1) & N_{-1}(t_1) & N_0(t_1) & \\ & \ddots & \ddots & \ddots & \\ & & N_{n-3}(t_n) & N_{n-2}(t_n) & N_{n-1}(t_n) \\ & & N''_{n-3}(t_n) & N''_{n-2}(t_n) & N''_{n-1}(t_n) \end{bmatrix} \begin{bmatrix} c_{-3} \\ c_{-2} \\ c_{-1} \\ \vdots \\ c_{n-2} \\ c_{n-1} \end{bmatrix} = \begin{bmatrix} 0 \\ f_0 \\ f_1 \\ \vdots \\ f_n \\ 0 \end{bmatrix}$$

Met de knooppunten $a = t_0, t_1, \ldots, t_n = b$ kunnen we maar $n-3$ B-splines construeren. Om aan een volledige basis van $n+3$ B-splines te komen, moeten we nog 6 knooppunten toevoegen als volgt:

$$t_{-3},\ t_{-2},\ t_{-1},\ t_0,\ \ldots, t_n,\ t_{n+1},\ t_{n+2},\ t_{n+3}$$

De functie `y = evalBspline( t, c, x )` evalueert de splinefunctie (6) met coëfficiënten `c` en knooppunten `t` in een vector met punten `x`. Hierbij is het van belang dat als $n+7$ de lengte is van `t`, dat $n+3$ de lengte is van `c`. Probeer bijvoorbeeld eens

```
t = -3:8;
N = length(t);
c = rand(1,N-4);
x = linspace(-5,10,1000);
figure
plot(x,evalBspline(t,c,x))
```

Probleem 4. Neem knooppunten `t=-3:8`. Maak een figuur waarin je enkele B-splines $N_j(x)$ plot in het interval $[-3, 8]$. Begin bijvoorbeeld met $N_{-3}(x)$. (Hint: je kan de coëfficiënten `c` bekomen als een kolom uit de eenheidsmatrix die je bekomt met het commando `eye`.) Merk op dat deze B-splines een basis vormen voor kubische splinefuncties in het interval $[0, 5]$!

Probleem 5. Ga voor knooppunten `t=-3:8` na wat de waarde is van $N_j(x)$, $j = -3, -2, \ldots, 4$ in de knooppunten $t_{-3}, t_{-2}, \ldots, t_8$.

Probleem 6. De waarden uit Probleem 5 zijn altijd hetzelfde voor equidistante knooppunten.

(a) Bestudeer de functie `Bsplinestelsel.m` met als output een matrix `A` en een vector `b`. Controleer dat elke rij van het stelsel $Ac = b$ overeenkomt met ofwel een interpolatievoorwaarde, ofwel een natuurlijke voorwaarde.

(b) Voer de functie uit voor `t = -1.75:0.25:1.75` en `f` een function handle voor $f(x) = 1/(1 + 25x^2)$, en bekijk de matrix A met het commando `spy`.

(c) Bereken de coëfficiënten van de splinefunctie als `c = A\b`. Plot $f(x)$ en de splinefunctie voor $x = [-1, 1]$ samen in een figuur.

(d) Ga na dat je dezelfde splinefunctie bekomen bent als in Probleem 3 (b). Je kan bijvoorbeeld beide splinefuncties evalueren in dezelfde punten `x` en dan kijken naar de norm het verschil met `norm`.

Oefeningen Numerieke Wiskunde

Oefenzitting 11: substitutiemethodes en Newton-Raphson

1 Theorie

We benaderen de wortels van de niet-lineaire vergelijking $f(x) = 0$ m.b.v. substitutiemethodes. Een substitutiemethode zet de vergelijking $f(x) = 0$ om in een vast punt vergelijking $x = F(x)$. De iteratieformule ziet er dan uit als

$$x^{(k+1)} = F\left(x^{(k)}\right). \tag{1}$$

Een punt x^* dat voldoet aan $x^* = F(x^*)$ heet een vaste punt van F.

Consistentie

Door het gebruik van een substitutiemethode kunnen er nulpunten verdwenen of bijgevoegd zijn. In dit verband worden de volgende begrippen gedefinieerd:

(i) $F(x)$ is **consistent** met de vergelijking $f(x) = 0$ als alle nulpunten van $f(x)$ ook vaste punten van $F(x)$ zijn $\longrightarrow \quad f(x) = 0 \Longrightarrow F(x) = x$

(ii) $F(x)$ is **reciprook consistent** met de vergelijking $f(x) = 0$ als alle vaste punten van $F(x)$ ook nulpunten van $f(x)$ zijn $\longrightarrow \quad f(x) = 0 \Longleftarrow F(x) = x$

(iii) $F(x)$ is **volledig consistent** als $F(x)$ consistent en reciprook consistent is $\longrightarrow \quad f(x) = 0 \Longleftrightarrow F(x) = x.$

Convergentie

Voor een differentieerbare $F(x)$ geldt **Stelling 14.2**: het iteratieproces (1) convergeert naar een vast punt x^* van $F(x)$ indien $|F'(x)| \leqslant M < 1$ in een bepaald interval rond $x^{(0)}$ en x^*. Het teken van $F'(x)$ bepaalt of de convergentie monotoon zal verlopen $(+)$ of volgens een vierkante spiraal $(-)$.

Zij $\{x^{(k)}\}_0^\infty$ een rij van getallen die convergeert naar x^* . De fout van de k-de benadering is $\varepsilon^{(k)} = x^{(k)} - x^*$. Zij

$$\rho^{(k)} = \frac{x^{(k)} - x^*}{x^{(k-1)} - x^*} = \frac{\varepsilon^{(k)}}{\varepsilon^{(k-1)}}. \tag{2}$$

Op voorwaarde dat de limiet bestaat noemen we $\rho = \lim_{k\to\infty} \rho^{(k)}$ de **convergentiefactor** van het benaderingsproces. Voor differentieerbare $F(x)$ met vast punt x^* en iteratieproces (1) is de convergentiefactor gelijk aan $\rho = F'(x^*)$ (zie **Stelling 15.2**).

Voor convergentie moet $|\rho| < 1$. Hoe kleiner $|\rho|$ is, des te sneller de convergentie. Als $\rho = 0$, dan kan men convergentiesnelheden vergelijken door de convergentieorde te definiëren. De **orde van convergentie** van het benaderingsproces is het getal $p,\ 1 \leq p \in \mathbb{R}$, zodat[1]

$$\varepsilon^{(k+1)} = O([\varepsilon^{(k)}]^p) \quad \text{en} \quad \varepsilon^{(k+1)} \neq o([\varepsilon^{(k)}]^p). \tag{3}$$

Dit getal p bepaalt ergens een grens. Zij

$$\lambda(n) = \lim_{k\to\infty} \frac{\varepsilon^{(k+1)}}{[\varepsilon^{(k)}]^n},$$

dan geldt er

$$\lambda(n) = \begin{cases} 0 & n < p \\ \rho_p & n = p \\ \infty & n > p \end{cases} \tag{4}$$

met $0 \leq \rho_p \leq \infty$.

Als de orde 1 is, dan is ρ_1 de convergentiefactor zoals wij hem hierboven gedefinieerd hebben. De orde van een substitutiemethode is altijd een natuurlijk getal. Om de orde van een substitutiemethode te bepalen, kan je **Stelling 15.3** gebruiken. We noemen een benaderingsproces van orde 1 lineair, een benaderingsproces van orde 2 kwadratisch en een benaderingsproces van orde 3 kubisch.

De methode van Newton-Raphson is een substitutiemethode met

$$F(x) = x - \frac{f(x)}{f'(x)},$$

zoals uitgelegd in Sectie 6. In Sectie 14.4 wordt de consistentie en convergentie van deze methode nagegaan.

2 Oefeningen

Probleem 1. Bewijs vergelijking (4).

Probleem 2. Beschouw de volgende substitutiemethodes voor de wortel van $f(x) = x + \log(x) = 0$.

(a) $x^{(k)} = -\log(x^{(k-1)})$

(b) $x^{(k)} = e^{-x^{(k-1)}}$

Ga voor beide methodes de consistentie en de convergentie na. Schets $F(x)$ en geef grafisch de convergentie weer voor de startwaarde $x^{(0)} = 0.9$. (De wortel is $x^* = 0.56714$.)

[1] Zie de Appendix van Hoofdstuk 2 van Deel 1 van het handboek voor de definities van *grote O* en *kleine O*.

Probleem 3. ($\sqrt{a}$ m.b.v. Newton-Raphson) Pas Newton-Raphson toe op de vergelijking $f(x) = 1 - \dfrac{a}{x^2}$.

(a) Onderzoek de consistentie.

(b) Onderzoek de convergentie. Wat is de orde?

(c) Schets $F(x)$.

(d) Voor welke startwaarden is er convergentie?

Probleem 4. Als x^* een m-voudige wortel is van $f(x) = 0$, dan geldt voor de Newton-Raphson methode dat ze lineair is als $m > 1$ en minstens kwadratisch als $m = 1$. Toon aan dat de volgende aangepaste methode minstens van tweede orde is

$$F(x) = x - m\frac{f(x)}{f'(x)},$$

als m de multipliciteit is van de wortel x^* van $f(x) = 0$. Maak gebruik van de afleiding in het handboek bij de gevalstudie van de methode van Newton-Raphson (Sectie 14.4) om niet al het rekenwerk opnieuw te moeten doen.

(Denkvraagje: Als deze methode hogere orde heeft dan NR, waarom wordt ze dan niet verkozen boven NR?)

Probleem 5. Beschouw de volgende substitutiemethodes voor de wortel van $f(x) = x + \log(x) = 0$.

(a) $x^{(k)} = \dfrac{x^{(k-1)} + e^{-x^{(k-1)}}}{2}$

(b) $x^{(k)} = \sqrt{-x^{(k-1)} \log(x^{(k-1)})}$

Ga voor beide methodes de consistentie en de convergentie na. Schets $F(x)$ en geef grafisch de convergentie weer voor de startwaarde $x^{(0)} = 0.9$. (De wortel is $x^* = 0.56714$.)

Probleem 6. ($\sqrt{a}$ m.b.v. Newton-Raphson) Pas Newton-Raphson toe op de vergelijking $f(x) = x - \dfrac{a}{x}$.

(a) Onderzoek de consistentie.

(b) Onderzoek de convergentie. Wat is de orde?

(c) Schets $F(x)$.

(d) Voor welke startwaarden is er convergentie?

Oefeningen Numerieke Wiskunde

Oefenzitting 12 (PC): Oplossen van niet-lineaire vergelijkingen

Het Maple bestand voor Probleem 1 en de `.m`-bestanden voor Probleem 4 kan je vinden op Toledo.

Probleem 1. (Convergentiesnelheid m.b.v. Maple) Open het Maple bestand **methods.mws**. De nulpunten van een gegeven functie worden gezocht met de volgende methodes: `bissectie`, `secant`, `newton`, `muller` en `halley`.

Voor alle methodes wordt het verloop van de benadering en van de relatieve fout $\epsilon^{(k)}$ weergegeven. Ook wordt voor elke methode het verloop van $\rho^{(k)} = \epsilon^{(k)}/[\epsilon^{(k-1)}]^p$ weergegeven, waarbij steeds de gekende orde van convergentie p gebruikt wordt die je kan terugvinden in Tabel 2.9 op p. 233 van het handboek.

- Bekijk rustig het Maple-werkblad. Over welke functie gaat het, wat wordt er precies geïmplementeerd, ...?
- Komen de convergentiefactoren en ordes overeen met wat er in de cursus staat?
- Verander de functie in $f(x) = x^2 - 2$, $f(x) = (x - 1.1)^2$ en $f(x) = (x - 1.4)^3$.
- Wat gebeurt er met de convergentiefactoren?
- Welke methodes falen en waarom?
- Verander de functie in $f(x) = \cos(x)$ en bekijk de resultaten van Newton-Raphson. Ga in Maple na wat de orde is. Bewijs dit ook theoretisch.

Herinnering: als je per iteratiestap een constant aantal cijfers wint, dan convergeert de methode lineair. Als het aantal cijfers verdubbelt, dan zegt men dat het algoritme kwadratisch convergeert. Ligt het ertussen, dan spreekt men van een superlineair convergentiegedrag. Het nagaan van de convergentie-orde kan natuurlijk alleen gebeuren indien de getallen met voldoende cijfers worden weergegeven.

Opmerking: als je iets verandert aan het Maple-werkblad, begin dan steeds terug bovenaan met `restart`. Je kan het hele werkblad opnieuw uitvoeren door op **!!!** te drukken.

Probleem 2. (Gulden snede) In deze opgave worden methoden onderzocht om de gulden snede (i.e., $(\sqrt{5}-1)/2$) numeriek te berekenen waarbij enkel gebruik gemaakt wordt van elementaire bewerkingen (optellen en vermenigvuldigen). De gulden snede kan berekend worden als de positieve oplossing van de vergelijking

$$x^2 + x - 1 = 0.$$

Implementeer hiervoor onderstaande substitutieformules:

(a) $x^{(k)} = 1 - \left(x^{(k-1)}\right)^2$,

(b) $x^{(k)} = \dfrac{3\left(x^{(k-1)}\right)^2 - x^{(k-1)} + 1}{4x^{(k-1)}}$,

(c) $x^{(k)} = \dfrac{4 - x^{(k-1)}}{4x^{(k-1)} + 3}$,

(d) $x^{(k)} = x^{(k-1)} - \dfrac{\left(x^{(k-1)}\right)^2 + x^{(k-1)} - 1}{2x^{(k-1)} + 1}$.

Neem als startwaarde $x^{(0)} = 0.7$.

- Ga telkens na of er convergentie of divergentie optreedt. Komt dit overeen met de theorie?
- Rangschik de formules volgens snelheid van convergentie. Wat zijn de convergentiefactoren? Wat is de convergentie orde?
- Hoe zie je of de convergentie al dan niet monotoon is?

Probleem 3. (Newton-Raphson) De iteratiefunctie voor de methode van Newton-Raphson is

$$F(x) = x - \frac{h(x)}{h'(x)}.$$

Gebruik deze methode om $\sqrt{2}$ te berekenen. Dit kan bv. door de oplossing te zoeken van één van onderstaande vergelijkingen. Ga de convergentie na van elk van die vergelijkingen. Gebruik hiervoor de startwaarde $x^{(0)} = 2$. Herhaal daarna de berekeningen met de startwaarde $x^{(0)} = 6$.

$$x^2 - 2 = 0$$
$$1 - \frac{2}{x^2} = 0$$
$$x - \frac{2}{x} = 0$$

$\sqrt{2}$ voldoet ook aan volgende vergelijking: $x^4 - 4x^2 + 4 = 0$. Bereken deze wortel met behulp van de Newton-Raphson methode en m.b.v. de substitutieformule

$$x^{(k)} = x^{(k-1)} - 2\frac{f(x^{(k-1)})}{f'(x^{(k-1)})}.$$

Waarom convergeert de tweede methode sneller? Kijk hiervoor naar een oefening van de vorige oefenzitting.

Probleem 4. Bereken met behulp van `secant.m`, `bisect.m`, `dekker.m` en je eigen Newton-Raphson implementatie het nulpunt van $f(x) = \tan(x) - x - 1$ gelegen in het interval $[0, \frac{\pi}{2}]$. Wat is de invloed van de startwaarden op de convergentie van de methode?

Oefeningen Numerieke Wiskunde

Oefenzitting 13: Iteratieve methoden voor het oplossen van stelsels

1 Niet-lineaire stelsels

Probleem 1. Pas Algoritme 2.5 voor Newton-Raphson van één niet-lineaire vergelijking aan voor stelsels van niet-lineaire vergelijkingen. Veronderstel dat de functie $f(x)$ en de Jacobiaan van deze functie $J(x)$ gegeven zijn, zodat je deze in het algoritme kunt evalueren. Welke fouten zou je kunnen gebruiken voor het stopcriterium? Schrijf naast het algoritme de dimensies van alle variabelen, bvb. scalar, $n \times 1$ vector, $n \times n$ matrix, enz.

Probleem 2. Beschouw het stelsel

$$\begin{cases} x^2 + x - y^2 - 1 &= 0 \\ y - \sin(x^2) &= 0 \end{cases}$$

(a) Bereken de Jacobiaan van dit stelsel.

(b) Het algoritme uit Probleem 1 wordt uitgevoerd met verschillende startwaarden. Het resultaat van de iteraties wordt grafisch weergegeven in Figuur 1 evanals de 2-norm van de relatieve fout van de iteratiepunten van een convergerende rij punten. Deze relatieve fout wordt berekend t.o.v. een referentie oplossing berekend in hogere precisie. Tabel 1 toont de iteratiepunten van deze rij. Wat is de orde van convergentie? Wat is de convergentiefactor? Hoe zie je dit aan de iteratiepunten in Tabel 1?

(c) Leid uit de Jacobiaan die je in (a) berekend hebt, de orde van convergentie af. (Hint: bekijk Opmerking 3 op p. 262 van het handboek.)

(d) Wat kan je in het algemeen afleiden uit de figuur omtrent de keuze van de startwaarde van de Newton Raphson methode voor stelsels niet-lineaire vergelijkingen?

Probleem 3. Beschouw het stelsel

$$\begin{aligned} x^2 + 4y^2 - 16 &= 0 \\ xy^2 - x - 4 &= 0 \end{aligned}$$

met als oplossingen $(2, \pm\sqrt{3})$ en $(-4, 0)$.

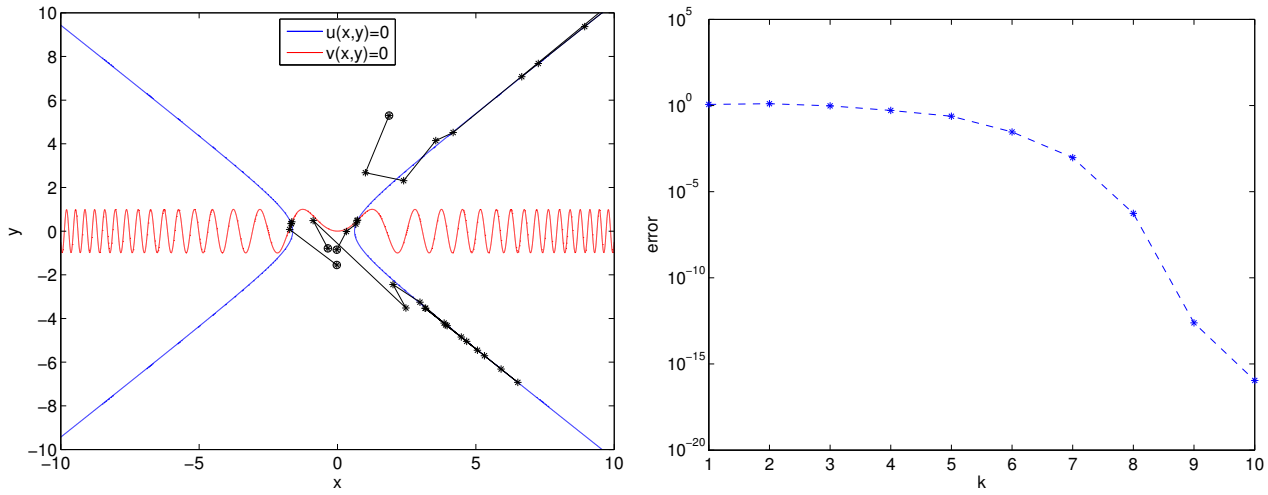
(a) Bereken de Jacobiaan van dit stelsel.

(b) Het algoritme uit Probleem 1 wordt uitgevoerd met de startwaarde $(4, 4)$. Het resultaat van de iteraties wordt grafisch weergegeven in Figuur 2 evanals de 2-norm van de relatieve fout van de iteratiepunten (t.o.v. een referentie oplossing berekend in hogere precisie). Tabel 2 toont de iteratiepunten van deze rij. Wat is de orde van convergentie? Wat is de convergentiefactor? Hoe zie je dit aan de iteratiepunten in Tabel 2?

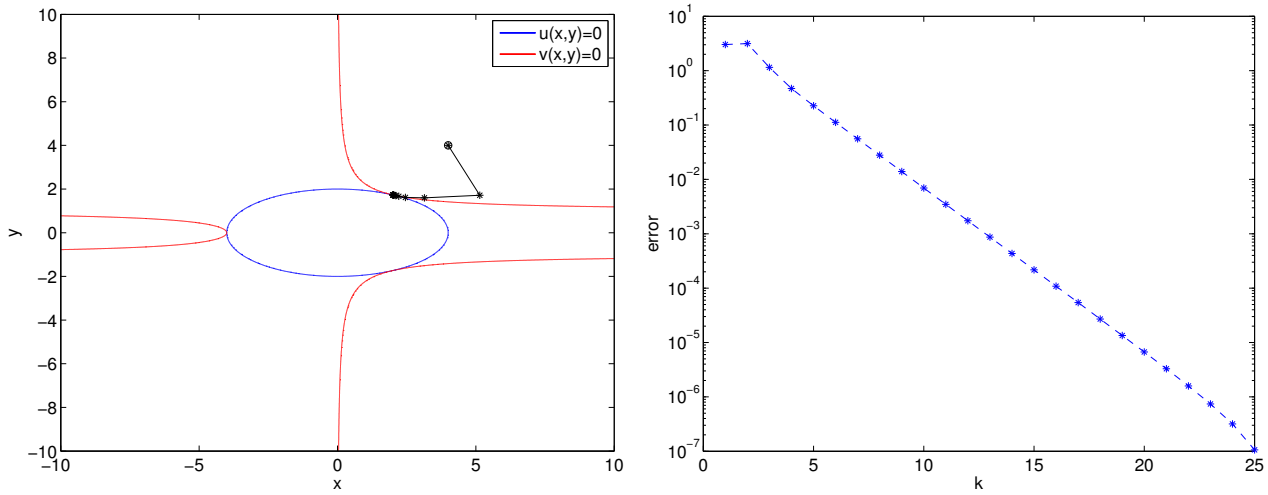
(c) Evalueer de Jacobiaan in het nulpunt waarnaar de methode convergeert. Wat kan je zeggen over de Jacobiaan en wat is het verband met de convergentie-orde? Hoe zie je dit ook aan de grafiek van de twee krommen?

k	$x^{(k)}$	$y^{(k)}$
1	7.6036866359446620e-01	1.6666666666666643e+00
2	1.5111331325429989e+00	1.5026360900263263e+00
3	1.2321741830888697e+00	1.3078763881079765e+00
4	9.6062004482353225e-01	9.6347404648365309e-01
5	5.5959998404791311e-01	3.3224417286278374e-01
6	7.1655089903159652e-01	4.7517566358048569e-01
7	7.2537904572941048e-01	5.0220141452690092e-01
8	7.2595044187617486e-01	5.0294603478122302e-01
9	7.2595072614305434e-01	5.0294650106230854e-01
10	7.2595072614319200e-01	5.0294650106250838e-01

Tabel 1: Newton Raphson Probleem 2



Figuur 1: Newton Raphson Probleem 2



Figuur 2: Newton Raphson Probleem 3

k	$x^{(k)}$	$y^{(k)}$
1	4.0000000000000000e+00	4.0000000000000000e+00
2	5.1428571428571423e+00	1.7142857142857144e+00
3	3.1423404804959958e+00	1.5956255920089555e+00
4	2.4552290895289555e+00	1.6159850096045763e+00
5	2.2168012409335196e+00	1.6699002600533723e+00
6	2.1075302011977848e+00	1.7010390282638144e+00
7	2.0535903269181404e+00	1.7165840255422320e+00
8	2.0267536289666963e+00	1.7243281169512277e+00
9	2.0133666713454845e+00	1.7281922334432744e+00
10	2.0066808291350657e+00	1.7301222247238226e+00
11	2.0033397915418467e+00	1.7310866935943789e+00
12	2.0016697404625843e+00	1.7315687951157588e+00
13	2.0008348314600086e+00	1.7318098124972439e+00
14	2.0004174060442890e+00	1.7319303128244170e+00
15	2.0002087006023777e+00	1.7319905608945905e+00
16	2.0001043496969459e+00	1.7320206844060908e+00
17	2.0000521746952660e+00	1.7320357460317017e+00
18	2.0000260873057409e+00	1.7320432768123817e+00
19	2.0000130436481811e+00	1.7320470421919831e+00
20	2.0000065218042851e+00	1.7320489248861475e+00
21	2.0000032608742067e+00	1.7320498662355768e+00
22	2.0000016304864339e+00	1.7320503368879867e+00
23	2.0000008153854134e+00	1.7320505721873833e+00
24	2.0000004072705941e+00	1.7320506899999837e+00
25	2.0000002037286335e+00	1.7320507487574865e+00
26	2.0000001027308829e+00	1.7320507779130259e+00

Tabel 2: Newton Raphson Probleem 3

2 Lineaire stelsels

Probleem 4. Beschouw het stelsel

$$\begin{cases} 2x_1 - x_2 &= 1 \\ -x_1 + 2x_2 &= 2 \end{cases}$$

Illustreer voor dit stelsel grafisch de methodes van Jacobi en Gauss-Seidel met de startvector

$$x^{(0)} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}.$$

Teken hiervoor de twee rechten van het stelsel en bepaal grafisch de iteratiepunten $x^{(1)}$, $x^{(2)}$ ... Convergeren de methoden?

Probleem 5. Stel dat de matrix A wordt opgedeeld in zijn diagonaal D en de overblijvende benedendriehoek L en de bovendriehoek U

$$A = U + D + L.$$

Dan kan de Jacobi methode in matrix notatie geschreven worden als

$$Dx^{(k+1)} = b - (L + U)x^{(k)}$$

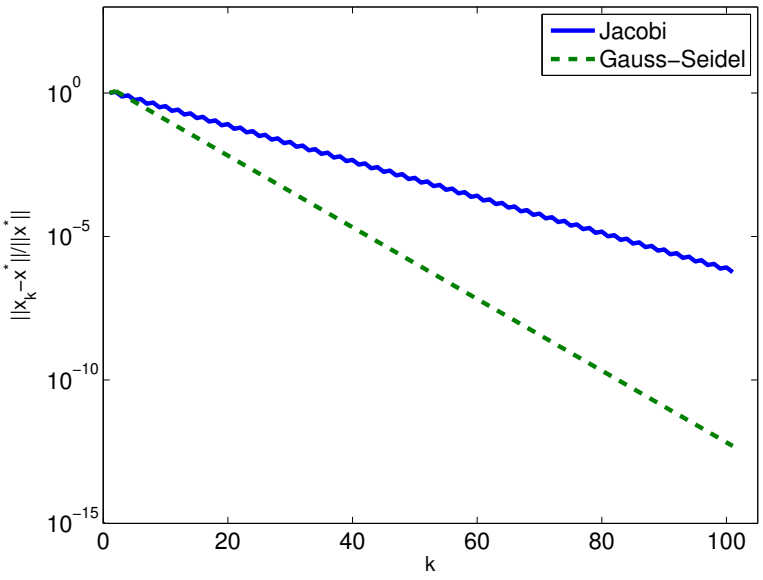
en de Gauss-Seidel methode als

$$Dx^{(k+1)} = b - Lx^{(k+1)} - Ux^{(k)}.$$

Toon aan dat dit overeenkomt met de formules in Algoritmes 4.1 en 4.2.

Probleem 6. Stel $e^{(k)} = x^{(k)} - x^*$ de k-de iteratiefout, bewijs dan dat

$$e^{(k)} = Ge^{(k-1)}, \qquad \text{met} \quad \begin{cases} G = -D^{-1}(U + L) & \text{voor Jacobi} \\ G = -(L + D)^{-1}U & \text{voor Gauss-Seidel} \end{cases}$$



Figuur 3: Jacobi en Gauss-Seidel, Probleem 8

Probleem 7. De spectraalradius van een matrix A wordt gedefinieerd als $\rho(A) = \max_i(|\lambda_i(A)|)$. Voor de convergentie van de methodes van Jacobi en Gauss-Seidel bestaan de volgende voorwaarden:

- $\|G\| < 1$ (voldoende voorwaarde voor convergentie),
- $\rho(G) < 1$ (nodige en voldoende voorwaarde voor convergentie).

Ga de convergentie van Jacobi en Gauss-Seidel na voor het stelsel uit Probleem 4 door $\|G\|_\infty$ en $\rho(G)$ te berekenen.

Probleem 8. Beschouw het stelsel

$$\begin{bmatrix} 2 & 3 \\ 1 & 2 \end{bmatrix} x = \begin{bmatrix} 100 \\ 101 \end{bmatrix}.$$

Bereken opnieuw $\|G\|_\infty$ en $\rho(G)$ voor beide methodes. Wat besluit je over de convergentie? In Figuur 3 wordt de relatieve fout getoond voor beide methodes. Wat zijn de convergentiefactoren? Wat valt er je op?

Oefeningen Numerieke Wiskunde

Oefenzitting 14 (PC): Iteratieve methoden voor het oplossen van stelsels

Voor het uitwerken van de opgaven moet je de `.m`-bestanden gebruiken die je kan vinden op Toledo.

1 Lineaire stelsels

Stel dat de matrix A wordt opgedeeld in zijn diagonaal D en de overblijvende benedendriehoek L en de bovendriehoek U

$$A = U + D + L.$$

Dan kan de Jacobi methode in matrix notatie geschreven worden als

$$Dx^{(k+1)} = b - (L + U)x^{(k)}$$

en de Gauss-Seidel methode als

$$Dx^{(k+1)} = b - Lx^{(k+1)} - Ux^{(k)}.$$

Indien $e^{(k)} = x^{(k)} - x^*$ de k-de iteratiefout is, dan geldt er dat

$$e^{(k)} = Ge^{(k-1)}, \qquad \text{met} \quad \begin{cases} G = -D^{-1}(U + L) & \text{voor Jacobi} \\ G = -(L + D)^{-1}U & \text{voor Gauss-Seidel.} \end{cases}$$

De spectraalradius van de matrix G is gedefinieerd als $\rho(G) = \max_i(|\lambda_i(G)|)$.

Probleem 1. (Jacobi vs Gauss-Seidel) Gebruik de methodes van Jacobi en Gauss-Seidel, geïmplementeerd in resp. `jacobi.m` en `gs.m`, om het volgende stelsel op te lossen. Gebruik als startwaarde $x_1 = x_2 = x_3 = 1$ en doe 20 iteraties. De gegeven MATLAB functies berekenen de fout t.o.v. een referentie oplossing, die je kan berekenen met `\`.

$$\begin{cases} 4x_1 - x_2 = 1 \\ -x_1 + 4x_2 - x_3 = 0 \\ - x_2 + 4x_3 = 1 \end{cases}$$

(a) Maak in één figuur voor elke methode een grafiek van de fout en voorzie een legende.

(b) Bereken voor elke methode met MATLAB een schatting van de convergentiefactor.

(c) Bereken voor elke methode de spectraalradius m.b.v het commando `eig`. Wat besluit je?

2 Niet-lineaire stelsels

Probleem 2. (Newton-Raphson) In deze oefening implementeer je de methode van Newton-Raphson en pas je deze toe op de volgende stelsels niet-lineaire vergelijkingen:

$$\begin{cases} x^2 + x - y^2 - 1 = 0 \\ y - \sin(x^2) = 0 \end{cases} \qquad \begin{cases} x^2 + 4y^2 - 16 = 0 \\ xy^2 - x - 4 = 0 \end{cases}$$

(a) Vul de functie `newton.m` aan zodat deze werkt zoals aangegeven in de commentaar.

(b) Bestudeer het script `oef2.m` en voer het uit om te zien of je functie werkt.

(c) Vul de anonieme functies voor `F` en `J` aan voor het tweede stelsel.

(d) Experimenteer met verschillende startwaarden en met de parameters `tol` en `Kmax`.

(e) Ga voor beide stelsels en de verschillende oplossingen na of de Jacobiaan regulier of singulier is.

Probleem 3. (Vereenvoudigde Newton-Raphson) In deze oefening pas je de methode van vereenvoudigde Newton-Raphson toe op het volgende stelsel:

$$\begin{cases} y - \sin(\pi x) = 0 \\ (x-1)^2 + (y-1)^2 = \dfrac{1}{3} \end{cases}$$

Bestudeer het script `oef3.m`.

(a) Voer het script verschillende keren uit met telkens een andere startwaarde. Naar welke oplossing kan de totale stapmethode convergeren mits een goede startwaarde?

(b) Vul het script aan zodat ook de enkelvoudige stapmethode uitgevoerd wordt. Naar welke oplossing kan deze methode convergeren?

(c) Bepaal voor elke methode de convergentie orde en de convergentiefactor.

(d) Pas de anonieme functies `vNRx` en `vNRy` aan zodat nu x berekend wordt uit de tweede vergelijking en y uit de eerste. Naar welke oplossing is er nu convergentie?

Probleem 4. (Complexe nulpunten - I) Gebruikt de methode van Newton-Raphson om de complexe nulpunten van onderstaande vergelijkingen te vinden. Beschouw een complex getal $z = x + iy$ als twee variabelen en genereer telkens een stelsel.

(a) $2z^2 + z + 1 = 0$

(b) $z - e^z = 0$

Probleem 5. (Complexe nulpunten - II) Wanneer de complexe functie $f : \mathbb{C} \to \mathbb{C}$ analytisch is, zal de iteratieformule

$$z^{(k+1)} = z^{(k)} - \frac{f(z^{(k)})}{f'(z^{(k)})}, \quad z^{(0)} \in \mathbb{C} \tag{1}$$

dezelfde eigenschappen behouden als de Newton-Raphson formule voor reële nulpunten. Formule (1) kan direct geïmplementeerd worden omdat Matlab rekent met complexe getallen. Zoek de complexe wortels van de functies uit de vorige opgave met behulp van deze methode. Merk wel dat je daarbij alleen naar een complex nulpunt kan convergeren indien de startwaarde complex is. Bewijs m.b.v. de Cauchy-Riemann voorwaarden dat beide methoden in feite identiek zijn.

Oefeningen Numerieke Wiskunde

Oefenzitting 16 (PC): Nulpunten van een veelterm

1 Theorie

Stel

$$p(x) = \sum_{i=0}^{n} a_i x^{n-i}, \quad p(c) = 0, \tag{1}$$

$$\bar{p}(x) = \sum_{i=0}^{n} \bar{a}_i x^{n-i}, \quad \bar{p}(\bar{c}) = 0, \tag{2}$$

$$\bar{c} = c + \Delta c, \quad \bar{a}_i = a_i + \Delta a_i, \quad \bar{p}(x) = p(x) + \Delta p(x), \tag{3}$$

$$\Delta p(x) = \sum_{i=0}^{n} \Delta a_i x^{n-i}. \tag{4}$$

Dus, c is een nulpunt van een veelterm $p(x)$ en $\bar{c}$ is een nulpunt van een geperturbeerde velteerm $\bar{p}(x)$.

Probleem 1. (Conditie van een enkelvoudig nulpunt) Stel dat c een enkelvoudig nulpunt van $p(x)$ is. Bewijs (door verwaarlozen van tweede-orde-termen):

$$\Delta c \approx \frac{-\sum_{i=0}^{n} \Delta a_i c^{n-i}}{p'(c)} = -\frac{\Delta p(c)}{p'(c)}. \tag{5}$$

Waarom geldt deze formule niet meer voor meervoudige nulpunten?

Probleem 2. (Conditie van een meervoudig nulpunt) Stel dat c een nulpunt is van $p(x)$ met meervoudigheid m. Bewijs dat

$$\Delta c \approx \left(\frac{-\sum_{i=0}^{n} \Delta a_i c^{n-i} m!}{p^{(m)}(c)} \right)^{1/m} = \left(-\frac{\Delta p(c) \cdot m!}{p^{(m)}(c)} \right)^{1/m}. \tag{6}$$

Stel dat $|\Delta a_i| < \varepsilon$. Dan zal

$$|\Delta c| \leq \left(\frac{\varepsilon m!}{|p^{(m)}(c)|} \sum_{i=0}^{n} |c|^{n-i} \right)^{1/m}. \tag{7}$$

Wat kun je uit (6) en (7) halen i.v.m. de conditie?

2 Praktijk

Gebruik het commando `roots` om de wortels van een veelterm te berekenen. Het commando `poly` berekent de coëfficiënten van de veelterm met gegeven wortels.

Probleem 3. Beschouw de veelterm in x met parameter t

$$p(x) = x^2 - 2x + t.$$

1. Zij $t = -3$. Breng een fout $\Delta t = 10^{-6}$ aan op t een controleer de foutenschatter (5) voor de wortel $c = 3$.

2. Zij $t = 1$. De veelterm heeft nu een dubbele wortel $c = 1$. Observeer dat de wortel nu veel gevoeliger is voor een fout op t en controleer de foutenschatter (6).

Probleem 4. Construeer een veelterm met een zesvoudig nulpunt $c = 2$ en bereken numeriek zijn wortels.

- Hoe nauwkeurig zijn ze? Teken de berekende nulpunten in het complexe vlak (gebruik hiervoor de commando's `real` en `imag`).
- Bereken de voorwaartse fout en de achterwaarts fout op de berekende nulpunten. (Hint: de achterwaartse fout is een fout op de coëfficiënten van de veelterm, zodat de nulpunten van de geperturbeerde veelterm de berekende nulpunten zijn.)
- Ga de geldigheid van formule (6) na m.b.v. `polyval`.

Probleem 5. (Veelterm van Wilkinson) De veelterm van Wilkinson is gekend als

$$p(x) = (x-1)(x-2)\dots(x-20) = x^{20} - 210x^{19} + \dots + 20!.$$

Construeer deze veelterm met het commando `poly(1:20)`. Bereken de nulpunten.

- Bereken de voorwaartse fout voor alle nulpunten en plot deze fout in functie van de nulpunten (met een gepaste schaal op de assen).
- Bereken ook de achterwaartse fout(veelterm) en plot deze fout.
- Ga formule (5) na voor alle nulpunten. Bereken hiervoor voor ieder nulpunt de teller en de noemer van (5). Voor welke nulpunten is de fout het grootst? Verklaar.

De Wilkinson veelterm is dus (zeer) slecht geconditioneerd voor het berekenen van sommige nulpunten. Merk op dat dit allemaal enkelvoudige nulpunten zijn.

Probleem 6*. (Eigenwaarden - I) Een primitieve methode voor het berekenen van alle eigenwaarden van een matrix A bestaat uit het berekenen van alle nulpunten van de karakteristieke vergelijking $\det(\lambda I - A) = 0$.

Construeer een diagonaalmatrix met de getallen $1, 2, \ldots, 20$ op de diagonaal (`A = diag(1:20)`). Bereken met het commando `roots(poly(A))` de wortels van de karakteristieke vergelijking daarvan. Hoe nauwkeurig zijn de resultaten?

Nochtans is het probleem goed geconditioneerd. Ga dit na door de eigenwaarden te berekenen van de geperturbeerde matrix `B = A + 1.0e-10 * rand(20)` met het stabiele Matlab-commando `eig(B)`.

Waaraan zijn de fouten te wijten als je weet dat zowel `poly` als `roots` afzonderlijk stabiel zijn?

Probleem 7*. (Eigenwaarden - II) Men kan het probleem van het berekenen van alle wortels van een veelterm omzetten tot het berekenen van alle eigenwaarden van een matrix, waarvoor stabiele algoritmen bestaan. Stel $p(x) = x^n + a_1 x^{n-1} + \ldots + a_n$ met nulpunten $\alpha_1,\ \alpha_2, \ldots,\ \alpha_n$. Beschouw de companion-matrix C_p van de veelterm $p(x)$:

$$C_p = \begin{pmatrix} -a_1 & -a_2 & \ldots & -a_{n-1} & -a_n \\ 1 & 0 & \ldots & 0 & 0 \\ 0 & 1 & & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \ldots & 1 & 0 \end{pmatrix}.$$

1. Bewijs dat de nulpunten van de veelterm $p(x)$ de eigenwaarden zijn van C_p.
2. Wat zijn de bijhorende eigenvectoren? ($C_p X_i = \alpha_i X_i$)

Probleem 8*. (Deflatie) Men noemt het wegdelen van reeds gevonden nulpunten een *deflatie*. Construeer een veelterm met nulpunten $x = 1, x = 10, x = 100, x = 1000, x = 10000$ en $x = 100000$. Bereken hiervan de nulpunten. Deel vervolgens met de routine `horner.m` het grootste berekende nulpunt weg. Herhaal dit tot je enkel het kleinste nulpunt overhoudt. Doe nu hetzelfde maar deel de nulpunten weg van klein naar groot. Welke resultaten bekom je? Waarom? Is deflatie een stabiele methode?

Oefeningen Numerieke Wiskunde

Oefenzitting 17 (PC): Het berekenen van eigenwaarden

In de oefenzitting ga je m.b.v. Matlab verschillende methodes voor het berekenen van de eigenwaarden van een matrix uitproberen op een voorbeeld. Het relevante deel van de cursus is Hoofdstuk 6 van Deel 2: 'Het berekenen van eigenwaarden'. Je past ook de methode van Jacobi uit Hoofdstuk 4 van Deel 2 toe op een stelsel van lineaire vergelijkingen en je onderzoekt het verband tussen de convergentie van deze methode en de methode van de machten voor eigenwaarden.

Voor het uitwerken van de opgaven gebruik je `.m`-bestanden die je kan vinden op Toledo.

Theorie

Berekenen van eigenwaarden Een willekeurige startvector $X^{(0)}$ is te schrijven als een lineaire combinatie van de eigenvectoren $\{V_k\}_{k=1}^n$ van een matrix A die van eenvoudige structuur is. Er geldt dan dat

$$A^k X^{(0)} = \lambda_1^k \left(\alpha_1 V_1 + \alpha_2 \left(\frac{\lambda_2}{\lambda_1}\right)^k V_2 + \ldots + \alpha_n \left(\frac{\lambda_n}{\lambda_1}\right)^k V_n \right).$$

We nemen aan dat

$$|\lambda_1| > |\lambda_2| \geq |\lambda_3| \geq \cdots \geq |\lambda_n|.$$

Bij de methode van de machten zal hierdoor de benadering van de eigenvector convergeren naar V_1 en men kan aantonen dat de convergentiefactor gelijk is aan $\rho = \lambda_2/\lambda_1$. Dit is ook de convergentiefactor voor de opeenvolgende benaderingen van de eigenwaarde λ_1.

Bij de methode van de inverse machten met verschuiving σ past men de methode van de machten toe op de matrix $(\sigma I - A)^{-1}$. Deze matrix heeft dezelfde eigenvectoren als de matrix A en heeft eigenwaarden $1/(\sigma - \lambda_i),\ i = 1, \ldots, n$. Hoe beter σ een goede benadering is van een bepaalde eigenwaarde, des te sneller de convergentie.

Indien men bij de methode van de inverse machten de verschuiving adaptief maakt, en voor σ steeds de nieuwe benadering van de eigenwaarde neemt, dan wordt de convergentie zelfs kwadratisch.

Iteratief oplossen van stelsels lineaire vergelijkingen Voor de methoden van Jacobi en Gauss-Seidel voor het oplossen van een stelsel $AX = B$ geldt er dat

$$E^{(k)} = GE^{(k-1)}, \qquad \text{met} \quad \begin{cases} G = -D^{-1}(U + L) & \text{voor Jacobi} \\ G = -(L + D)^{-1}U & \text{voor Gauss-Seidel} \end{cases}$$

met $E^{(k)} = X^{(k)} - X^*$ de k-de iteratiefout. Indien we $E^{(0)}$ schrijven als een linaire combinatie van de eigenvectoren $\{V_k\}_{k=1}^n$ van G, dan krijgen we

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

Hieruit ziet men dat als $\alpha_1 \neq 0$ de fout afneemt zoals $\mathcal{O}(\lambda_1^k)$, en dat de convergentiefactor in modulus gelijk is aan de spectraalradius $\rho(G) = \max_i (|\lambda_i(G)|)$.

Beschrijving van de Matlab-bestanden

- `[x, mu] = machten(A, x0, N)` past de methode van de machten met scalering toe op de matrix `A` met `x0` als startvector en `N` het aantal iteraties. Als output krijg je de benaderde eigenvector `x` en de opeenvolgende benaderingen `mu` van de overeeenkomstige eigenwaarde.
- `[x,mu] = invmachten(A, x0, sigma, N)` past de methode van de inverse machten met verschuiving `sigma` toe op de matrix `A` met `x0` als startvector en `N` het aantal iteraties. Dit komt overeen met het toepassen van de methode van de machten op $(\sigma I - A)^{-1}$. Hoe beter de schatting σ voor een bepaalde eigenwaarde, hoe sneller de methode zal convergeren. Als output krijg je de benaderde eigenvector `x` en de opeenvolgende benaderingen `mu` van de overeeenkomstige eigenwaarde.
- `[x,mu] = invmachten_adaptief(A, x0, sigma, N)` past de methode van de inverse machten met adaptieve verschuiving toe op de matrix `A` met `x0` als startvector en `N` het aantal iteraties. Als output krijg je de benaderde eigenvector `x` en de opeenvolgende benaderingen `mu` van de overeeenkomstige eigenwaarde.
- `fout = jacobi(A, b, x0, xe, n)` past de methode van Jacobi toe voor het iteratief oplossen van een stelsel $Ax = b$ met startvector `x0`, exacte oplossing `xe` en aantal iteraties `n` . Als output krijg je de normsgewijze relatieve fout van de verschillende iteraties, berekend t.o.v. de exacte oplossing `xe`.

Oefeningen

Probleem 1. (Methode van de machten) Pas de methode van de machten toe om de dominante eigenwaarde te berekenen van onderstaande matrices A en B. Gebruik hierbij als startvector `x0` achtereenvolgens $[\,1\ 0\ 0\,]^T$ en $[\,1\ 1\ 1\,]^T$.

$$A = \begin{pmatrix} 102 & -201 & 100 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix}, \quad B = \begin{pmatrix} -1 & 4 & 0 \\ -2 & 5 & 0 \\ 0 & 1 & 2 \end{pmatrix}.$$

(a) Plot telkens de relatieve fouten (vergelijk met de output van `eig`).

(b) Indien je convergentie hebt, bereken dan numeriek een schatting voor de convergentiefactor en vergelijk deze waarde met de theoretische convergentiefactor.

(c) Verklaar eventueel onverwacht numeriek gedrag.

Probleem 2. (Methode van de inverse machten) Gebruik `invmachten.m` om van bovenstaande matrix B met startvector `x0` $= [\,1\ 0\ 0\,]^T$ sneller de eigenwaarden te vinden door aan `sigma` een benadering van de gezochte eigenwaarde toe te kennen. Gebruik de waarden $2.8, 2.9$ en 2.99 voor `sigma`.

(a) Plot de fouten in één figuur samen met de fout voor de methode van de machten.

(b) Bepaal de theoretische convergentiefactoren ρ en voeg de grafieken van ρ^k toe aan je figuur.

Probleem 3. (Gevoeligheid voor initiële condities bij Jacobi) Doe dertig Jacobi-iteraties om het volgende stelsel op te lossen:

$$\begin{cases} 3x_1 - 2x_2 + 2x_3 = 3 \\ -2x_1 + 3x_2 - 2x_3 = -1 \\ 2x_1 - 2x_2 + 3x_3 = 3 \end{cases}$$

Neem als startwaarden $\begin{bmatrix} 3 & 2 & 0 \end{bmatrix}^T$ en $\begin{bmatrix} 3.0001 & 2 & 0 \end{bmatrix}^T$ en plot de fouten. Verklaar wat er gebeurt:

(a) Bereken de iteratiematrix G en bereken de eigenwaarden en eigenvectoren.

(b) Wat is de spectraalradius $\rho(G)$? Wat besluit je over de convergentie?

(c) Bepaal de initiële foutvector $E^{(0)}$. Verklaar nu in detail het convergentiegedrag dat je bekomt. (Hint: maak gebruik van de eigenvectoren van G.)

Probleem 4. (Methode van de inverse machten met adaptieve verschuiving) Pas `invmachten_adaptief` toe op de matrix B met startvector `x0` $= [1\ 0\ 0]^T$, $N = 10$ en een eerste schatting voor `sigma` gelijk aan 2.8.

(a) Bereken en plot de relatieve fout. Bekijk ook de uitvoer `mu` en de fout in de Command Window. Welke nauwkeurigheid bereik je?

(b) Wat is de orde van convergentie?

(c) Verklaar waarom je vanaf een bepaalde iteratie enkel nog `NaN` waarden krijgt.

(d) Probeer ook eens waarden voor `sigma` die dichter bij de andere eigenwaarden van B liggen.

Oefeningen Numerieke Wiskunde: Oefenzitting 11

Tom Sydney Kerckhove

7 april 2014

1 Probleem 1

Zie verbetering van Toledo.

2 Probleem 2

(a)

Consistentie

`http://www.wolframalpha.com/input/?i=-+log+x`

$$x + \log(x) = 0 \Leftrightarrow x = -\log(x) \Leftrightarrow F(x) = x$$

We spreken dus van volledige consistentie.

Convergentie

$$|F'(x^*)| = |\frac{1}{-x^*}| = |\frac{1}{0.56714}| \approx 1.7632 > 1$$

Er gebeurt hier dus spiraalvorigme divergentie.

(b)

Consistentie

`http://www.wolframalpha.com/input/?i=-e^x`

$$x + \log(x) = 0 \Leftrightarrow x = e^{-x} \Leftrightarrow F(x) = x$$
$$\Leftrightarrow -x = \log(x) \Leftrightarrow f(x) = 0$$

Er is op nieuw volledige consistentie.

Convergentie

$$|F'(x^*)| = | - e^{-x^*}| = e^{-0.56714} < 1$$

Hier hebben we dus spiraalvormige convergentie want de afgeleide is kleiner dan nul.

Oefeningen Numerieke Wiskunde: Oefenzitting 12

Tom Sydney Kerckhove

23 april 2014

1 Probleem 1

Te Bewijzen:

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

Bewijs.

$$F(x) = x - \frac{-\cos(x)}{-\sin(x)} = x + \cot x$$

Zie p 323 (bewijs)

$$F'(x) = 1 - \csc^2(x) \to F'(x^*) = 0$$

$$F''(x) = 2\cot(x)\csc^2(x) \to F''(x^*) = 0$$

$$F'''(x) = -2(\cos(2x) + 2)\csc^4(x) \to F'''(x^*) = -2$$

Het proces is dus van orde 3. □

KOMT OP EXAMEN

Oefeningen Numerieke Wiskunde: Oefenzitting 11

Tom Sydney Kerckhove

7 april 2014

1 Probleem 1

Zie verbetering van Toledo.

2 Probleem 2

(a)

Consistentie

`http://www.wolframalpha.com/input/?i=-+log+x`

$$x + \log(x) = 0 \Leftrightarrow x = -\log(x) \Leftrightarrow F(x) = x$$

We spreken dus van volledige consistentie.

Convergentie

$$|F'(x^*)| = |\frac{1}{-x^*}| = |\frac{1}{0.56714}| \approx 1.7632 > 1$$

Er gebeurt hier dus spiraalvorigme divergentie.

(b)

Consistentie

`http://www.wolframalpha.com/input/?i=-e^x`

$$x + \log(x) = 0 \Leftrightarrow x = e^{-x} \Leftrightarrow F(x) = x$$
$$\Leftrightarrow -x = \log(x) \Leftrightarrow f(x) = 0$$

Er is op nieuw volledige consistentie.

Convergentie

$$|F'(x^*)| = | - e^{-x^*}| = e^{-0.56714} < 1$$

Hier hebben we dus spiraalvormige convergentie want de afgeleide is kleiner dan nul.

Oefeningen Numerieke Wiskunde: Oefenzitting 2

Tom Sydney Kerckhove
Verbeteringen door Ward Schodts

19 februari 2014

1 Probleem 1

De exponent gaat van -126 tot 127. Dit betekent dat hier 8 bits voor nodig zijn. Er blijven dus nog $32 - 8 = 23$ bits over voor de mantisse want er is ook nog een tekenbit. Een andere manier om dit resultaat te bekomen is de formule voor de machine precisie te gebruiken.

$$\epsilon_{mach} = \frac{b^{1-p}}{2}$$

Hierin is b de basis en p het aantal juiste cijfers na de komma. Uit deze formule halen we de p.

$$1 - p = \log_b(2\epsilon_{mach})$$

$$p = 1 - \log_2(2 \cdot 2^{-24}) = 24 \text{ (waarvan 1 tekenbit, dus) } p = 23$$

Volgens dezelfde redenering vinden we dat het aantal bits voor het teken, de mantisse en de exponent bij de dubbele-nauwkeurigheidsgetallen.

2 Probleem 2

- We weten dat 1 en 2 respectievelijk als volgt voorgesteld worden.

  $$1 = .100\ldots00 \cdot 2^1$$

  $$2 = .100\ldots00 \cdot 2^2$$

  We hebben hier weer een mantisse van 52 cijfers.
  Hier zitten dus $2^m - 1 = 2^{52} - 1$ getallen tussen.

- Voor de gemakkelijkheid bereken we eerst van 7 tot 8, en dan van 8 tot 9. We weten dat 7, 8 en 9 respectievelijk als volgt voorgesteld worden.

  $$7 = .11100\ldots00 \cdot 2^3$$

  $$8 = .10000\ldots00 \cdot 2^4$$

  $$9 = .10010\ldots00 \cdot 2^4$$

  Hier zitten dus $2^{50} + 2^{49} - 1$ getallen tussen.

3 Probleem 3

Het laatste zinvolle getal in deze rij is 1000. De overgang van 999 naar 1000 resulteert niet in een fout want 1000 wordt als volgt voorgesteld.

$$0.100 \cdot 10^4$$

Als we hier echter nog 1 bij optellen gebeurt er een absolute afrondingsfout van precies -1. Vanaf dan is de beschreven rij dus constant (1000).

4 Probleem 4

- – **bepaal_b** We beginnen met $A = 1$, dan vermenigvuldigen we A met 2 tot er een afrondingsfout gebeurd wanneer je $(A + 1) - A = 1$ evalueert. A is nu gelijk aan 2^{10}. Vervolgens initialiseren we i op 1 en hogen i op zolang $(A+i) = A$. i is dus net groot genoeg om A op de volgende afronding te laten stoppen. Dit is wanneer i 6 wordt. b is dus $.103 \cdot 10^4 - .102 \cdot 10^4 = 10$.
  – **bepaal_p** We initialiseren p op 1 en z op 10. Nu hogen we p en de exponent van 10^1 op tot $(z + 1) - z = 1$ een afrondingsfout geeft. Dit is wanneer $p = 3$ geldt.
- Extra:
  – **bepaal_b**
  Te Bewijzen

  Bewijs. □ bewijs dit
  – **bepaal_p**
  Te Bewijzen

  Bewijs. □ bewijs dit

5 Probleem 5

1.
$$\overline{y} = fl\left(\frac{x}{fl\left(fl\left(\sqrt{fl\,(x+1)}\right)+1\right)}\right)$$
$$= \frac{x(1+\epsilon_1)}{\left(\left(\sqrt{(x+1)\,(1+\epsilon_4)}\right)(1+\epsilon_3)+1\right)(1+\epsilon_2)}$$
$$\overline{y} = y + \sum_{i=1}^{4} \epsilon_i \frac{\delta \overline{y}}{\delta \epsilon_i}(0,0,0,0)$$

2. •
$$\frac{\delta \overline{y}}{\delta \epsilon_1}(\epsilon_1,0,0,0) = \frac{x}{\sqrt{x+1}+1}$$

- $$\frac{\delta \overline{y}}{\delta \epsilon_2}(0, \epsilon_2, 0, 0) = \frac{-x}{\left(\sqrt{x+1}+1\right)(1+\epsilon_2)^2}$$
$$\frac{\delta \overline{y}}{\delta \epsilon_2}(0, 0, 0, 0) = \frac{-x}{\sqrt{x+1}+1}$$
- $$\frac{\delta \overline{y}}{\delta \epsilon_3}(0, 0, \epsilon_3, 0) = \frac{-x\sqrt{1+x}}{\left((\sqrt{1+x})(1+\epsilon_3)+1\right)^2}$$
$$\frac{\delta \overline{y}}{\delta \epsilon_3}(0, 0, 0, 0) = \frac{-x\sqrt{1+x}}{\left(\sqrt{1+x}+1\right)^2}$$
- $$\frac{\delta \overline{y}}{\delta \epsilon_4}(0, 0, 0, \epsilon_4) = \frac{-x(x+1)}{2\sqrt{(x+1)(1+\epsilon_4)}\left(\sqrt{(x+1)(1+\epsilon_4)}+1\right)^2}$$
$$\frac{\delta \overline{y}}{\delta \epsilon_4}(0, 0, 0, 0) = \frac{-x(x+1)}{2\sqrt{x+1}\left(\sqrt{x+1}+1\right)^2}$$
$$= \frac{-x(x+1)}{(2\sqrt{(x+1)^3}+4(x+1)+2\sqrt{x+1})}$$

3. $$\overline{y} \approx y + \frac{x}{\sqrt{x+1}+1}\epsilon_1 + \frac{-x}{\sqrt{x+1}+1}\epsilon_2$$
$$+\frac{-x\sqrt{1+x}}{\left(\sqrt{1+x}+1\right)^2}\epsilon_3 + \frac{-x(x+1)}{(2\sqrt{(x+1)^3}+4(x+1)+2\sqrt{x+1})}\epsilon_4$$
$$\overline{y} \approx y + y\epsilon_1 - y\epsilon_2 + y\frac{\sqrt{1+x}}{\sqrt{1+x}+1}\epsilon_3 - y\frac{1}{2}\frac{\sqrt{1+x}}{\sqrt{1+x}+1}\epsilon_4$$
$$\overline{y} - y \approx y\epsilon_1 - y\epsilon_2 + y\frac{\sqrt{1+x}}{\sqrt{1+x}+1}\epsilon_3 - y\frac{1}{2}\frac{\sqrt{1+x}}{\sqrt{1+x}+1}\epsilon_4$$
$$\frac{\overline{y}-y}{y} \approx \epsilon_1 - \epsilon_2 + \frac{\sqrt{1+x}}{\sqrt{1+x}+1}\epsilon_3 - \frac{1}{2}\frac{\sqrt{1+x}}{\sqrt{1+x}+1}\epsilon_4$$

6 Probleem 6

$$y = \frac{1-\cos(x)}{x^2}$$

1. $$\overline{y} = fl\left(\frac{fl(1-fl(\cos(x)))}{fl(x^2)}\right)$$
$$\overline{y} = \frac{(1-(\cos(x))(1+\epsilon_1))(1+\epsilon_2)}{(x^2)(1+\epsilon_3)}(1+\epsilon_4)$$

2.

$$\frac{\delta\overline{y}}{\delta\epsilon_1}(\epsilon_1,0,0,0) = \frac{\delta}{\delta\epsilon_1}\frac{1-(\cos(x))(1+\epsilon_1)}{x^2} = -\frac{\cos(x)}{x^2}$$
$$\frac{\delta\overline{y}}{\delta\epsilon_2}(0,\epsilon_2,0,0) = \frac{\delta}{\delta\epsilon_2}\frac{(1-\cos(x))(1+\epsilon_2)}{x^2} = \frac{(1-\cos(x))}{x^2}$$
$$\frac{\delta\overline{y}}{\delta\epsilon_3}(0,0,\epsilon_3,0) = \frac{\delta}{\delta\epsilon_3}\frac{1-\cos(x)}{(x^2)(1+\epsilon_3)} = \frac{1-\cos(x)}{(x^2)}\frac{-1}{(1+\epsilon_3)^2}$$
$$\frac{\delta\overline{y}}{\delta\epsilon_3}(0,0,0,0) = -\frac{1-\cos(x)}{x^2}$$
$$\frac{\delta\overline{y}}{\delta\epsilon_4}(0,0,0,\epsilon_4) = \frac{\delta}{\delta\epsilon_4}\frac{(1-\cos(x))(1+\epsilon_4)}{x^2} = \frac{1-\cos(x)}{x^2}$$

3.

$$\overline{y} \approx y - \frac{\cos(x)}{x^2}\epsilon_1 + \frac{(1-\cos(x))}{x^2}\epsilon_2 - \frac{1-\cos(x)}{x^2}\epsilon_3 + \frac{1-\cos(x)}{x^2}\epsilon_4$$
$$\overline{y} \approx y - \frac{\cos(x)}{x^2}\epsilon_1 + y\epsilon_2 - y\epsilon_3 + y\epsilon_4$$

4.

$$\overline{y} - y \approx -\frac{\cos(x)}{x^2}\epsilon_1 + y\epsilon_2 - y\epsilon_3 + y\epsilon_4$$
$$\frac{\overline{y}-y}{y} \approx -\frac{\cos(x)}{x(1-\cos(x))}\epsilon_1 + \epsilon_2 - \epsilon_3 + \epsilon_4$$

Indien x naar nul nadert wordt de factor bij ϵ_1 zodanig groot dat we te maken hebben met een grote absolute en relatieve fout.

7 Probleem 7

$$S = \sum_{i=1}^{n} a_i$$

1.

$$\overline{S} = fl(fl(...fl(fl(fl(fl(a_1+a_2)+a_3)+a_4)...)+a_n)$$
$$= ((...((((a_1+a_2)(1+\epsilon_2))+a_3)(1+\epsilon_3))...)+a_n)(1+\epsilon_n)$$

- $$\frac{\delta\overline{y}}{\delta\epsilon_2}(\epsilon_2,0,...,0) = a_1+a_2$$
- $$\frac{\delta\overline{y}}{\delta\epsilon_3}(0,\epsilon_3,0,...,0) = a_1+a_2+a_3$$
- $$\frac{\delta\overline{y}}{\delta\epsilon_i}(0,...,0,\epsilon_i,0,...,0) = \sum_{j=1}^{i} a_j$$

2.

$$\overline{S} \approx y + \sum_{k=2}^{n}\epsilon_k\sum_{i=1}^{k} a_i$$

3.

$$\overline{S} - S \approx \sum_{k=2}^{n}\epsilon_k\sum_{i=1}^{k} a_i$$

8 Probleem 8

(a)

$$y = x\sin(x)$$

1.

$$\overline{y} = fl(xfl(\sin(x)))$$
$$\overline{y} = (x\sin(x)(1+\epsilon_1))(1+\epsilon_2)$$

2. - $$\frac{\delta\overline{y}}{\delta\epsilon_1}(\epsilon_1, 0) = x\sin(x)$$
   - $$\frac{\delta\overline{y}}{\delta\epsilon_2}(0, \epsilon_2) = x\sin(x)$$

3.

$$\overline{y} \approx y + x\sin(x)\epsilon_1 + x\sin(x)\epsilon_2$$
$$\overline{y} \approx y + y\epsilon_2 + y\epsilon_3$$

4.

$$\overline{y} - y = y\epsilon_1 + y\epsilon_2$$
$$\frac{\overline{y} - y}{y} = \epsilon_1 + \epsilon_2$$

(b)

$$y = \frac{1-\cos(x)}{\sin(x)}$$

1.

$$\overline{y} = fl\left(\frac{fl(1 - fl(\cos(x)))}{fl(\sin(x))}\right)$$
$$\overline{y} = \left(\frac{(1-(\cos(x))(1+\epsilon_1))(1+\epsilon_2)}{(\sin(x))(1+\epsilon_3)}\right)(1+\epsilon_4)$$

2. - $$\frac{\delta\overline{y}}{\delta\epsilon_1}(\epsilon_1, 0, 0, 0) = \frac{\delta\overline{y}}{\delta\epsilon_1}\frac{1-\cos(x)(1+\epsilon_1)}{\sin(x)} = \frac{-\cos(x)}{\sin(x)}$$
   - $$\frac{\delta\overline{y}}{\delta\epsilon_2}(0, \epsilon_2, 0, 0) = \frac{\delta\overline{y}}{\delta\epsilon_2}\frac{(1-\cos(x))(1+\epsilon_2)}{\sin(x)} = \frac{1-\cos(x)}{\sin(x)}$$
   - $$\frac{\delta\overline{y}}{\delta\epsilon_3}(0, 0, \epsilon_3, 0) = \frac{\delta\overline{y}}{\delta\epsilon_3}\frac{1-\cos(x)}{(\sin(x))(1+\epsilon_3)} = \frac{\cos(x)-1}{\sin(x)(1+\epsilon_3)^2}$$
     $$\frac{\delta\overline{y}}{\delta\epsilon_3}(0, 0, 0, 0) = \frac{\cos(x)-1}{\sin(x)}$$
   - $$\frac{\delta\overline{y}}{\delta\epsilon_4}(0, 0, 0, \epsilon_4) = \frac{\delta\overline{y}}{\delta\epsilon_4}\left(\frac{1-\cos(x)}{\sin(x)}\right)(1+\epsilon_4) = \frac{1-\cos(x)}{\sin(x)}$$

3.

$$\overline{y} \approx y + \frac{-\cos(x)}{\sin(x)}\epsilon_1 + \frac{1-\cos(x)}{\sin(x)}\epsilon_2 + \frac{\cos(x)-1}{\sin(x)}\epsilon_3 + \frac{1-\cos(x)}{\sin(x)}\epsilon_4$$
$$\overline{y} \approx y + \frac{-\cos(x)}{\sin(x)}\epsilon_1 + y\epsilon_2 - y\epsilon_3 + y\epsilon_4$$

4.

$$\overline{y} - y \approx \frac{-\cos(x)}{\sin(x)}\epsilon_1 + y\epsilon_2 - y\epsilon_3 + y\epsilon_4$$

$$\frac{\overline{y} - y}{y} \approx \frac{-\cos(x)}{1-\cos(x)}\epsilon_1 + \epsilon_2 - \epsilon_3 + \epsilon_4$$

(c)

$$y = \frac{1 - e^{-2x}}{x}$$

1.

$$\overline{y} = fl\left(\frac{fl\left(1 - fl(e^{fl(-2x)})\right)}{x}\right)$$

$$\overline{y} = \left(\frac{\left(1 - ((e^{(-2x)(1+\epsilon_1)})(1+\epsilon_2))\right)(1+\epsilon_3)}{x}\right)(1+\epsilon_4)$$

2.

- $$\frac{\delta\overline{y}}{\delta\epsilon_1}(\epsilon_1, 0, 0, 0) = \frac{\delta\overline{y}}{\delta\epsilon_1}\frac{1 - e^{(-2x)(1+\epsilon_1)}}{x} = \frac{2xe^{(-2x)(1+\epsilon_1)}}{x}$$

  $$\frac{\delta\overline{y}}{\delta\epsilon_1}(0, 0, 0, 0) = \frac{2xe^{-2x}}{x}$$

- $$\frac{\delta\overline{y}}{\delta\epsilon_2}(0, \epsilon_2, 0, 0) = \frac{\delta\overline{y}}{\delta\epsilon_2}\frac{1 - (e^{-2x})(1+\epsilon_2)}{x} = \frac{-e^{-2x}}{x}$$

- $$\frac{\delta\overline{y}}{\delta\epsilon_3}(0, 0, \epsilon_3, 0) = \frac{\delta\overline{y}}{\delta\epsilon_3}\frac{(1 - e^{-2x})(1+\epsilon_3)}{x} = \frac{1 - e^{-2x}}{x}$$

- $$\frac{\delta\overline{y}}{\delta\epsilon_4}(0, 0, 0, \epsilon_4) = \frac{\delta\overline{y}}{\delta\epsilon_4}\left(\frac{1 - e^{-2x}}{x}\right)(1+\epsilon_4) = \frac{1 - e^{-2x}}{x}$$

3.

$$\overline{y} \approx y + \frac{2xe^{-2x}}{x} + \frac{-e^{-2x}}{x} + \frac{1 - e^{-2x}}{x} + \frac{1 - e^{-2x}}{x}$$

$$\overline{y} \approx y + \frac{2xe^{-2x}}{x}\epsilon_1 + \frac{-e^{-2x}}{x}\epsilon_2 + y\epsilon_3 + y\epsilon_4$$

4.

$$\overline{y} - y \approx \frac{2xe^{-2x}}{x}\epsilon_1 + \frac{-e^{-2x}}{x}\epsilon_2 + y\epsilon_3 + y\epsilon_4$$

$$\frac{\overline{y} - y}{y} \approx \frac{2xe^{-2x}}{1 - e^{-2x}}\epsilon_1 + \frac{-e^{-2x}}{1 - e^{-2x}}\epsilon_2 + \epsilon_3 + \epsilon_4$$

(d)

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

1.

$$\overline{y} = fl\left((1+x)^{\frac{1}{x}}\right)$$

$$\overline{y} = \left(((1+x)(1+\epsilon_2))^{\left(\frac{1}{x}\right)(1+\epsilon_1)}\right)(1+\epsilon_3)$$

2.
- $$\frac{\delta \overline{y}}{\delta \epsilon_1}(\epsilon_1, 0, 0) = \frac{\delta \overline{y}}{\delta \epsilon_1}(1+x)^{(\frac{1}{x})(1+\epsilon_1)} = \frac{(x+1)^{\frac{\epsilon_1+1}{x}} ln(x+1)}{x}$$
$$\frac{\delta \overline{y}}{\delta \epsilon_1}(0, 0, 0) = \frac{(x+1)^{\frac{1}{x}} ln(x+1)}{x}$$
- $$\frac{\delta \overline{y}}{\delta \epsilon_2}(0, \epsilon_2, 0) = \frac{\delta \overline{y}}{\delta \epsilon_2}((1+x)(1+\epsilon_2))^{\frac{1}{x}} = \frac{(1+x)}{a}((1+x)(1+\epsilon_2))^{\frac{1}{x}-1}$$
$$\frac{\delta \overline{y}}{\delta \epsilon_2}(0, 0, 0) = \frac{(1+x)}{a}(1+x)^{\frac{1}{x}-1} = \frac{(1+x)^{\frac{1}{x}}}{a}$$
- $$\frac{\delta \overline{y}}{\delta \epsilon_3}(0, 0, \epsilon_3) = \frac{\delta \overline{y}}{\delta \epsilon_3}\left((1+x)^{\frac{1}{x}}\right)(1+\epsilon_3) = \left((1+x)^{\frac{1}{x}}\right)$$

3. $$\overline{y} \approx y + \frac{(x+1)^{\frac{1}{x}} ln(x+1)}{x}\epsilon_1 + \frac{(1+x)^{\frac{1}{x}}}{a}\epsilon_2 + \left((1+x)^{\frac{1}{x}}\right)\epsilon_3$$
$$\overline{y} \approx y + \frac{ln(x+1)}{x}y\epsilon_1 + \frac{1}{a}y\epsilon_2 + y\epsilon_3$$

4. $$\overline{y} - y \approx \frac{ln(x+1)}{x}y\epsilon_1 + \frac{1}{a}y\epsilon_2 + y\epsilon_3$$
$$\frac{\overline{y} - y}{y} \approx \frac{ln(x+1)}{x}\epsilon_1 + \frac{1}{a}\epsilon_2 + \epsilon_3$$

(e) $$y = \sqrt{e^x - 1}$$

1. $$\overline{y} = fl\left(\sqrt{fl(fl(e^x) - 1)}\right)$$
$$\overline{y} = \left(\sqrt{((e^x)(1+\epsilon_1) - 1)(1+\epsilon_2)}\right)(1+\epsilon_3)$$

2.
- $$\frac{\delta \overline{y}}{\delta \epsilon_1}(\epsilon_1, 0, 0) = \frac{\delta \overline{y}}{\delta \epsilon_1}\sqrt{(e^x)(1+\epsilon_1) - 1} = \frac{e^x}{2\sqrt{(e^x)(1+\epsilon_1) - 1}}$$
$$\frac{\delta \overline{y}}{\delta \epsilon_1}(0, 0, 0) = \frac{e^x}{2\sqrt{e^x - 1}}$$
- $$\frac{\delta \overline{y}}{\delta \epsilon_2}(0, \epsilon_2, 0) = \frac{\delta \overline{y}}{\delta \epsilon_2}\sqrt{(e^x - 1)(1+\epsilon_2)} = \frac{e^x - 1}{2\sqrt{(e^x - 1)(1+\epsilon_2)}}$$
$$\frac{\delta \overline{y}}{\delta \epsilon_2}(0, 0, 0) = \frac{e^x - 1}{2\sqrt{e^x - 1}} = \frac{1}{2}\sqrt{e^x - 1}$$
- $$\frac{\delta \overline{y}}{\delta \epsilon_3}(0, 0, \epsilon_3) = \frac{\delta \overline{y}}{\delta \epsilon_3}(1+\epsilon_3)\sqrt{e^x - 1} = \sqrt{e^x - 1}$$

3. $$\overline{y} \approx y + \frac{e^x}{2\sqrt{e^x - 1}}\epsilon_1 + \frac{1}{2}\sqrt{e^x - 1}\epsilon_2 + \sqrt{e^x - 1}\epsilon_3$$
$$\overline{y} \approx y + +y\frac{e^x}{2e^x - 2}\epsilon_1 + y\frac{1}{2}\epsilon_2 + y\epsilon_3$$

4.

$$\overline{y} - y \approx y\frac{e^x}{2e^x - 2}\epsilon_1 + y\frac{1}{2}\epsilon_2 + y\epsilon_3$$
$$\frac{\overline{y} - y}{y} \approx \frac{e^x}{2e^x - 2}\epsilon_1 + \frac{1}{2}\epsilon_2 + \epsilon_3$$

(f)

$$y = \sin\left(\frac{1}{x}\right)$$

1.

$$\overline{y} = \left(\sin\left(\frac{(1+\epsilon_1)}{x}\right)\right)(1+\epsilon_2)$$
$$\overline{y} = ...$$

2. •

$$\frac{\delta\overline{y}}{\delta\epsilon_1}(\epsilon_1, 0) = \frac{\delta\overline{y}}{\delta\epsilon_1}\sin\left(\frac{(1+\epsilon_1)}{x}\right) = \frac{1}{x}\cos\left(\frac{(1+\epsilon_1)}{x}\right)$$
$$\frac{\delta\overline{y}}{\delta\epsilon_1}(0,0) = \frac{1}{x}\cos\left(\frac{1}{x}\right)$$

•

$$\frac{\delta\overline{y}}{\delta\epsilon_2}(0, \epsilon_2) = \frac{\delta\overline{y}}{\delta\epsilon_2}\left(\sin\left(\frac{1}{x}\right)\right)(1+\epsilon_2) = \sin\left(\frac{1}{x}\right)$$

3.

$$\overline{y} \approx y + \frac{1}{x}\cos\left(\frac{1}{x}\right)\epsilon_1 + \sin\left(\frac{1}{x}\right)\epsilon_2$$
$$\overline{y} \approx y + y\frac{1}{x}\cot\left(\frac{1}{x}\right)\epsilon_1 + y\epsilon_2$$

4.

$$\overline{y} - y \approx y\frac{1}{x}\cot\left(\frac{1}{x}\right)\epsilon_1 + y\epsilon_2$$
$$\frac{\overline{y} - y}{y} \approx \frac{1}{x}\cot\left(\frac{1}{x}\right)\epsilon_1 + \epsilon_2$$

(g)

$$y = (1+x^2)^{x^2}$$

1.

$$\overline{y} = fl\left(\left(fl(1 + fl(x^2))\right)^{fl(x^2)}\right)$$
$$\overline{y} = \left(\left((1 + (x^2)(1+\epsilon_2))(1+\epsilon_3)\right)^{(x^2)(1+\epsilon_1)}\right)(1+\epsilon_4)$$

2. •

$$\frac{\delta\overline{y}}{\delta\epsilon_1}(\epsilon_1, 0, 0, 0) = \frac{\delta\overline{y}}{\delta\epsilon_1}(1+x^2)^{(x^2)(1+\epsilon_1)} = x^2(x^2+1)^{x^2(1+\epsilon_1)}ln(x^2+1)$$
$$\frac{\delta\overline{y}}{\delta\epsilon_1}(0,0,0,0) = x^2(x^2+1)^{x^2}ln(x^2+1)$$

•

$$\frac{\delta\overline{y}}{\delta\epsilon_2}(0, \epsilon_2, 0, 0) = \frac{\delta\overline{y}}{\delta\epsilon_2}(1 + (x^2)(1+\epsilon_2))^{x^2} = x^4(1 + (x^2)(1+\epsilon_2))^{x^2-1}$$
$$\frac{\delta\overline{y}}{\delta\epsilon_2}(0,0,0,0) = x^4(1+x^2)^{x^2-1}$$

- $$\frac{\delta \overline{y}}{\delta \epsilon_3}(0,0,\epsilon_3,0) = \frac{\delta \overline{y}}{\delta \epsilon_3}((1+x^2)(1+\epsilon_3))^{x^2}$$
$$= (1+x^2)x^2((1+x^2)(1+\epsilon_3))^{x^2-1}$$
$$\frac{\delta \overline{y}}{\delta \epsilon_3}(0,0,0,0) = (1+x^2)x^2(1+x^2)^{x^2-1}$$

- $$\frac{\delta \overline{y}}{\delta \epsilon_4}(0,0,0,\epsilon_4) = \frac{\delta \overline{y}}{\delta \epsilon_4}\left((1+x^2)^{x^2}\right)(1+\epsilon_4) = (1+x^2)^{x^2}$$

3. $$\overline{y} \approx y + x^2(x^2+1)^{x^2} ln(x^2+1)\epsilon_1 + x^4(1+x^2)^{x^2-1}\epsilon_2$$
$$+(1+x^2)x^2(1+x^2)^{x^2-1}\epsilon_3 + (1+x^2)^{x^2}\epsilon_4$$
$$y = (1+x^2)^{x^2}$$
$$\overline{y} \approx y + x^2 y ln(x^2+1)\epsilon_1 + \frac{x^4 y}{1+x^2}\epsilon_2 + yx^2\epsilon_3 + y\epsilon_4$$

4. $$\overline{y} - y \approx x^2 y ln(x^2+1)\epsilon_1 + \frac{x^4 y}{1+x^2}\epsilon_2 + yx^2\epsilon_3$$
$$\frac{\overline{y}-y}{y} \approx x^2 ln(x^2+1)\epsilon_1 + \frac{x^4}{1+x^2}\epsilon_2 + x^2\epsilon_3$$

(h) $$y = \frac{e^{x^2} - e^{-x^2}}{2x^2}$$

1. $$\overline{y} = fl\left(\frac{fl\left(fl\left(e^{fl(x^2)}\right) - fl\left(e^{fl(-fl(x^2))}\right)\right)}{fl(2fl(x^2))}\right)$$
$$\overline{y} = \frac{\left(e^{(x^2)(1+\epsilon_1)} - e^{(-(x^2)(1+\epsilon_1))(1+\epsilon_2)}\right)(1+\epsilon_4)}{(2(x^2)(1+\epsilon_1))(1+\epsilon_3)}(1+\epsilon_5)$$

2. - $$\frac{\delta \overline{y}}{\delta \epsilon_1}(\epsilon_1,0,0,0,0) = \frac{\delta \overline{y}}{\delta \epsilon_1}\frac{e^{(x^2)(1+\epsilon_1)} - e^{-(x^2)(1+\epsilon_1)}}{2(x^2)(1+\epsilon_1)}$$
$$= \frac{e^{-x^2(\epsilon_1+1)}\left(x^2(\epsilon_1+1)\left(e^{2x^2(\epsilon_1+1)}+1\right)\right) - e^{2x^2(\epsilon_1+1)} + 1}{2x^2(\epsilon_1+1)^2}$$

   AWW HELL NAW

9 Probleem 9

- Product

$$y = \prod_{i=1}^{n} a_i$$

1.

$$\overline{y} = fl(fl(...fl(fl(fl(a_1a_2)a_3)a_4)...a_{n-1})a_n)$$
$$\overline{y} = ((...((a_1a_2)(1+\epsilon_2)a_3)(1+\epsilon_3)...a_{n-1})(1+\epsilon_{n-1})a_n)(1+\epsilon_n)$$
$$= \left(\prod_{i=1}^{n} a_i\right)\left(\prod_{i=2}^{n}(1+\epsilon_i)\right)$$

2. –

$$\frac{\delta\overline{y}}{\delta\epsilon_1}(\epsilon_1, 0, ..., 0) = \frac{\delta\overline{y}}{\delta\epsilon_1} = \left(\prod_{i=1}^{n} a_i\right)(1+\epsilon_1)$$
$$\frac{\delta\overline{y}}{\delta\epsilon_1}(0, ..., 0) = \prod_{i=1}^{n} a_i$$

–

$$\frac{\delta\overline{y}}{\delta\epsilon_2}(0, \epsilon_2, ..., 0) = \frac{\delta\overline{y}}{\delta\epsilon_2} = \left(\prod_{i=1}^{n} a_i\right)(1+\epsilon_2)$$
$$\frac{\delta\overline{y}}{\delta\epsilon_2}(0, ..., 0) = \prod_{i=1}^{n} a_i$$

–

$$\frac{\delta\overline{y}}{\delta\epsilon_j}(0, ..., \epsilon_j, ..., 0) = \frac{\delta\overline{y}}{\delta\epsilon_j} = \left(\prod_{i=1}^{n} a_i\right)(1+\epsilon_j)$$
$$\frac{\delta\overline{y}}{\delta\epsilon_j}(0, ..., 0) = \prod_{i=1}^{n} a_i$$

3.

$$\overline{y} \approx y + \sum_{j=1}^{n}\epsilon_j \prod_{i=1}^{n} a_i$$
$$\overline{y} \approx y + \sum_{j=1}^{n}\epsilon_j y$$

4.

$$\overline{y} - y \approx \sum_{j=1}^{n}\epsilon_j y$$
$$\frac{\overline{y} - y}{y} \approx \sum_{j=1}^{n}\epsilon_j$$

- Scalair Product

$$y = \sum_{i=1}^{n} a_i b_i$$

1.
$$\overline{y} = fl(fl(...fl(fl(fl(a_1b_1) + fl(a_2b_2)) + fl(a_3 + b_3))...) + fl(a_nb_n))$$
$$\overline{y} = (((a_1b_1(1+\epsilon_{1.}) + a_2b_2(1+\epsilon_{2.}))(1+\epsilon_{2+}) + a_3b_3(1+\epsilon_{3.})(1+\epsilon_{3+})...)(1+\epsilon_{n+})$$

2. –
$$\frac{\delta\overline{y}}{\delta\epsilon_{1.}}(0,...,\epsilon_{1.},...,0) = a_1b_1$$
$$\frac{\delta\overline{y}}{\delta\epsilon_{2.}}(0,...,\epsilon_{2.},...,0) = a_2b_2$$
$$\frac{\delta\overline{y}}{\delta\epsilon_{i.}}(0,...,\epsilon_{i.},...,0) = a_ib_i$$

–
$$\frac{\delta\overline{y}}{\delta\epsilon_{2+}}(0,...,\epsilon_{2+},...,0) = a_1b_1 + a_2b_2$$
$$\frac{\delta\overline{y}}{\delta\epsilon_{3+}}(0,...,\epsilon_{3+},...,0) = a_1b_1 + a_2b_2 + a_3b_3$$
$$\frac{\delta\overline{y}}{\delta\epsilon_{i+}}(0,...,\epsilon_{i+},...,0) = \sum_{j=1}^{i} a_jb_j$$

3.
$$\overline{y} \approx y + \sum_{i=1}^{n} a_ib_i\epsilon_{i.} + \sum_{i=2}^{n}\epsilon_{i+}\sum_{j=1}^{i} a_jb_j$$

10 Probleem 10

$$y = \sum_{i=1}^{n} a_i$$

Bewijs. Bewijs door volledige inductie op n
Voor $n = 1$ is de stelling triviaal.
Stel dat de stelling klopt voor een bepaalde $n = k$, dan bewijzen we nu dat de stelling geldt voor $n = k + 1$.

1.
$$\overline{y} = fl\left(fl\left(fl\left(...\right) + fl\left(...\right)\right) + fl\left(fl\left(...\right) + fl\left(...\right)\right)\right)$$

LOLNOPE

□

11 Probleem 11

Een vermenigvuldiging met 2 geeft bijna geen fout in een computer. Een vermenigvuldiging met 10 daarentegen geeft wel degelijk een fout zoals we zien in de tweede grafiek. De machine werkt duidelijk met basis 2. De machine werkt met ongeveer 8 beduidende cijfers.

Todo list

Oefeningen Numerieke Wiskunde: Oefenzitting 3

Tom Sydney Kerckhove
Verbeteringen door Ward Schodts

26 februari 2014

1 Bewegende kommavoorstelling

1.1 Probleem 1

$$b = 2 \text{ en } p = 53$$
$$eps = 2.2204e-16 \text{ en } \epsilon_{mach} = 1.1102e-16$$

Voorbeeld van een afrondingsfout:

$$(1 + 0.70 \cdot eps) - 1 = 2.2204e-16$$

0.70 voor eps is net groot genoeg om eps naar boven laten af te ronden, vandaar dat eps wordt teruggegeven

1.2 Probleem 2

- Correct:
  0.125, 0.25, 0.5, 1, 2, 8, 1, 10 ,100, 1000.
- Incorrect:
  0.001, 0.01, 0.1

0.1 wordt voorgesteld als

```
+1.10011001100110011001100110011001100110011001100110011010 2^01111111011 (2^-4)
```

De fout is dus $fl(0.1) - 0.1$.

```
-0.00000000000000000000000000000000000000000000000000000100110011...  2^01111111011 (2^-4)
```

$$\Delta(0.1) = (1.00110011..)_2 \cdot 2^{-56}$$

2 Taylorreeks van e^x

2.1 Probleem 3

(a) De waarde van k verandert niet meer na een bepaalde iteratie omdat de component die nog toegevoegd wordt kleiner is dan voorstelbaar.

(b) Met matlab, zie abs_err en rel_err.

(c) LogLog of Semilogy geeft het mooiste resultaat.

(d) Zie matlab voor illustratie.

2.2 Probleem 4

Voor grote n delen we door een voorgestelde nul in de machine. De benadering convergeert beter, maar is ongeveer even goed.

2.3 Probleem 5

(a) Voor grote x duurt het langer voor x^k kleiner wordt dan $k!$.

(b) Zie matlab voor illustratie.

(c) Ja, de relatieve fout is van de grootteorde 10^{-15}.

(d) De absolute fout is van grootteorde 10^{-7}. Aangezien een de bovengrens voor de som van n getallen deels gegeven wordt door: $\epsilon_{mach} \cdot \sum_i^n a_i$. Bij deze waarden is de grootste waarde van t al reeds 10^8. We weten ook dat de machine precisie gelijk is aan $1.110223024625157e^{-16}$. Dus we zien dat de fout groter wordt als de bovengrens want $10^{-16} \cdot 10^8 = 10^{-8} < 10^{-7}$

2.4 Probleem 6

(a) Zie matlab voor illustratie.

(b) Nee, de relatieve fout is van de grootteorde 10^1.

(c) Ja.

$$\begin{aligned}\overline{S} - S &\approx \sum_{k=2}^{n} \epsilon_k \sum_{i=1}^{k} a_i \\ &= \sum_{k=2}^{n} \epsilon_k \sum_{i=1}^{k} \frac{x^n}{n!}\end{aligned}$$

Zoals we zien is de absolute fout in het begin heel groot omdat het even duur voordat x^n kleiner wordt dan $n!$.

3 Probleem 7

(a)

$$\lim_{x\to 0} \frac{1-\cos(x)}{x^2} = \lim_{x\to 0} \frac{\sin(x)}{2x0} = \lim_{x\to 0} \frac{\cos(x)}{2} = \frac{1}{2}$$

(b) Vanaf een bepaalde waarde wordt de benadering terug slechter, en daarna gaat alles naar de maan.

(c) LogLog werkt het best, omdat zowel x als abs_err een logaritmisch verloop hebben.

(d) De absolute fout is:

$$f(x) - \frac{1}{2} = -\frac{1}{2} + \frac{1}{2} - \frac{x^4}{4!} + \frac{x^6}{6!} - ... =\sim \frac{x^4}{4!}$$

(e) Vervang als volgt:

$$\cos(x) = 1 - \frac{x^2}{2!} + \frac{x^4}{4!} - \frac{x^6}{6!} + \cdots$$

$$\delta y \approx y\left(-\frac{1 - \frac{x^2}{2!} + \frac{x^4}{4!} - \frac{x^6}{6!} + \cdots}{\frac{x^2}{2!} - \frac{x^4}{4!} + \frac{x^6}{6!} - \cdots}\epsilon_1 + \epsilon_2 - \epsilon_3 + \epsilon_4\right) =\sim \frac{1}{x^2}\epsilon_1$$

4 Probleem 8

1. 0.1 wordt naar het volgende getal afgekapt:

```
+0.00011001100110011001100
```

$$= 0.09999990463256835937$$

De absolute fout is dus de volgende uitdrukking

```
+0.00000000000000000000000110011001100110011...
=
9.536743164617612e-8
```

De relatieve fout is dus de volgende.

```
9.536743164617612e-7
```

2. Vermenigvuldigen we de relatieve fout met 100 uur $(60 * 60 * 100)$, dan krijgen we de fout op de berekende tijd.

$$0.343$$

3.

$$0.343 * 1676 = 574.868m$$

5 Probleem 9

(a) (a) Wortels

$$y = \sqrt[2^{40}]{x}$$

i.

$$\overline{y} = fl\left(\sqrt{...fl\left(\sqrt{fl(\sqrt{x})}\right)...}\right)$$

$$\overline{y} = (1+\epsilon_{40})\sqrt{...(1+\epsilon_2)\sqrt{(1+\epsilon_1)\sqrt{x}}}$$

$$= (1+\epsilon_{40})(1+\epsilon_{39})^{\frac{1}{2}}(1+\epsilon_{38})^{\frac{1}{4}}...(1+\epsilon_2)^{\frac{1}{2^{38}}}(1+\epsilon_1)^{\frac{1}{2^{39}}}\ \sqrt[2^{40}]{x}$$

$$= \sqrt[2^{40}]{x}\prod_{i=1}^{40}(1+\epsilon_i)^{\frac{1}{2^{40-i}}}$$

ii.

$$\frac{\delta\overline{y}}{\delta\epsilon_i}(0,...,\epsilon_i,...,0) = \frac{\delta}{\delta\epsilon_i}\ \sqrt[2^{40}]{x}(1+\epsilon_i)^{\frac{1}{2^{40-i}}} = \sqrt[2^{40}]{x}\frac{(1+\epsilon_i)^{\frac{1}{2^{40-i}}-1}}{2^{40-i}}$$

$$\frac{\delta\overline{y}}{\delta\epsilon_i}(0,...,0) = \frac{\sqrt[2^{40}]{x}}{2^{40-i}}$$

iii.

$$\overline{y} \approx y + \sum_{i=1}^{40}\frac{\sqrt[2^{40}]{x}}{2^{40-i}}\epsilon_i$$

$$\overline{y} \approx y + \sum_{i=1}^{40}\frac{y}{2^{40-i}}\epsilon_i$$

iv.

$$\overline{y} - y \approx \sum_{i=1}^{40} \frac{y}{2^{40-i}} \epsilon_i$$

$$\frac{\overline{y} - y}{y} \approx \sum_{i=1}^{40} \frac{1}{2^{40-i}} \epsilon_i \leq \left(\frac{2^{40} - 1}{2^{40}} + 1 \right) \epsilon_{mach}$$

(b) Kwadraten

$$y = x^{2^{40}}$$

i.

$$\overline{y} = fl\left(fl\left(fl\left(fl\left(fl\left(x^2 \right)^2 \right)^2 \right)^2 \right)^2 \right)$$

$$\overline{y} = (1 + \epsilon_{40}) \left(...(1 + \epsilon_3) \left((1 + \epsilon_2) \left((1 + \epsilon_1) x^2 \right)^2 \right)^2 ... \right)^2$$

$$\overline{y} = x^{2^{40}} \sum_{i=1}^{40} (1 + \epsilon_i)^{2^{40-i}}$$

ii.

$$\frac{\delta \overline{y}}{\delta \epsilon_i}(0, ..., \epsilon_i, ..., 0) = \frac{\delta}{\delta \epsilon_i} x^{2^{40}} (1 + \epsilon_i)^{2^{40-i}} = x^{2^{40}} 2^{40-i} (1 + \epsilon_i)^{2^{40-i} - 1}$$

$$\frac{\delta \overline{y}}{\delta \epsilon_i}(0, ..., 0) = x^{2^{40}} 2^{40-i}$$

iii.

$$\overline{y} \approx y + \sum_{i=1}^{40} x^{2^{40}} 2^{40-i} \epsilon_i = x^{2^{40}} \sum_{i=1}^{40} 2^{40-i} \epsilon_i$$

$$\overline{y} \approx y + y \sum_{i=1}^{40} 2^{40-i} \epsilon_i$$

iv.

$$\overline{y} - y \approx y \sum_{i=1}^{40} 2^{40-i} \epsilon_i$$

$$\frac{\overline{y} - y}{y} \approx \sum_{i=1}^{40} 2^{40-i} \epsilon_i \leq \left(2^{40} - 1 \right) \epsilon_{mach}$$

We kunnen nu de fout berekenen.

$$y = x \text{ maar } \overline{y} \neq x$$

$$\overline{y} = x \left(1 + \sum_{i=1}^{40} \frac{1}{2^{40-i}} \epsilon_i \right) \left(1 + \sum_{j=1}^{40} 2^{40-j} \epsilon_j \right)$$

(b) Zie matlab

(c) Zie matlab

(d) Ja, op het eerste deel komen vrij weinig fouten.

6 Probleem 10

(a) Zie matlab

(b) Zie matlab

$$e^{x^2} = \sum_{k=0}^{\infty} \frac{x^{2^k}}{k!},\ e^{-x^2} = \sum_{k=0}^{\infty} \frac{-x^{2^k}}{k!}$$

$$f(x) = \frac{e^{x^2} - e^{-x^2}}{2x^2} = \frac{\sum_{k=0}^{\infty} \frac{x^{2^k}}{k!} - \sum_{k=0}^{\infty} \frac{-x^{2^k}}{k!}}{2x^2} = \frac{\sum_{k=0}^{\infty} \frac{2x^{2+4k}}{\frac{2+4k}{2}!}}{2x^2}$$

(c) LOLNOPE

7 Probleem 11

probleen 11

Oefeningen Numerieke Wiskunde: Oefenzitting 4

Tom Sydney Kerckhove
Verbeterd door Ward Schodts

5 maart 2014

1 Probleem 1

$$y_1 = 1 + 2x - \frac{1}{1+x} \text{ en } y_2 = \frac{(3+2x)x}{1+x}$$

$$\overline{y_1} = fl\left(fl(1+fl(2x)) - fl\left(\frac{1}{fl(1+x)}\right)\right) \text{ en } \overline{y_2} = fl\left(\frac{fl((fl(3+fl(2x)))x)}{fl(1+x)}\right)$$

1.1 Conditie

$$\Delta_c y_1 \approx f'(x)\Delta x = 2 + \frac{1}{(1+x)^2}\Delta x$$

$$\Delta_c y_2 \approx f'(x)\Delta x = \frac{(1+x)(3+4x) - (3x+2x^2)}{(1+x)^2} = \frac{(3+4x+2x^2)}{(1+x)^2}\Delta x$$

Dit is natuurlijk gelijk.

$$\delta_c y \approx \frac{f'(x)x}{f(x)}\delta x = \frac{2x^2+4x+3}{(1+x)(3+2x)}\delta x$$

Slecht geconditioneerd voor $x \approx -1$, $x \approx -\frac{3}{2}$.

1.2 Stabiliteit

eval1:

1. $$\overline{y_1} = \left((1+2x)(1+\epsilon_1) - \left(\frac{1}{(1+x)(1+\epsilon_2)}\right)(1+\epsilon_3)\right)(1+\epsilon_4)$$

2. - $$\frac{\delta \overline{y}}{\delta \epsilon_1}(\epsilon_1,0,0,0) = \frac{\delta}{\delta \epsilon_1}(1+2x)(1+\epsilon_1) - \frac{1}{1+x} = 1+2x$$
   - $$\frac{\delta \overline{y}}{\delta \epsilon_2}(0,\epsilon_2,0,0) = \frac{\delta}{\delta \epsilon_2}1 + 2x - \frac{1}{(1+x)(1+\epsilon_2)} = \frac{1}{1+x}\frac{1}{(1+\epsilon_2)^2}$$

     $$\frac{\delta \overline{y}}{\delta \epsilon_2}(0,0,0,0) = \frac{1}{1+x}$$
   - $$\frac{\delta \overline{y}}{\delta \epsilon_3}(0,0,\epsilon_3,0) = 1 + 2x - (\frac{1}{1+x})(1+\epsilon_3) = -\frac{1}{1+x}$$

- $$\frac{\delta\overline{y}}{\delta\epsilon_4}(0,0,0,\epsilon_4) = \frac{\delta}{\delta\epsilon_4}(1+2x-\frac{1}{1+x})(1+\epsilon_4) = 1+2x-\frac{1}{1+x}$$

3. $$\overline{y} \approx y + (1+2x)\epsilon_1 + \frac{1}{1+x}\epsilon_2 - \frac{1}{1+x}\epsilon_3 + \left(1+2x-\frac{1}{1+x}\right)\epsilon_4$$

4. $$\overline{y} - y \approx (1+2x)\epsilon_1 + \frac{1}{1+x}\epsilon_2 - \frac{1}{1+x}\epsilon_3 + \left(1+2x-\frac{1}{1+x}\right)\epsilon_3$$
$$\delta_s y_1 = \frac{\overline{y}-y}{y} \approx \frac{(1+2x)(1+x)}{(3+2x)x}\epsilon_1 + \frac{1}{(3+2x)x}\epsilon_2 - \frac{1}{(3+2x)x}\epsilon_3 + \epsilon_4$$

5. $$\delta_s y_1 \leq \frac{3+3x+4x^2}{(3+2x)x}\epsilon_{mach}$$

Dit algoritme is onstabiel voor $x \approx 0$ en voor $x \approx x = -\frac{2}{3}$ maar stabiel voor andere x.

eval2:

1. $$\overline{y_2} = \frac{(3+2x)x}{1+x}$$
$$\overline{y_2} = \left(\frac{(3+2x)(1+\epsilon_1)x}{(1+x)(1+\epsilon_2)}\right)(1+\epsilon_3)$$

2. - $$\frac{\delta\overline{y}}{\delta\epsilon_1}(\epsilon_1,0,0) = \frac{\delta}{\delta\epsilon_1}\frac{(3+2x)(1+\epsilon_1)x}{1+x} = \frac{(3+2x)x}{1+x}$$
   - $$\frac{\delta\overline{y}}{\delta\epsilon_2}(0,\epsilon_2,0) = \frac{\delta}{\delta\epsilon_2}\frac{(3+2x)x}{(1+x)(1+\epsilon_2)} = \frac{(3+2x)x}{1+x}\frac{1}{(1+\epsilon_2)^2}$$
   $$\frac{\delta\overline{y}}{\delta\epsilon_2}(0,...,0) = \frac{(3+2x)x}{1+x}$$
   - $$\frac{\delta\overline{y}}{\delta\epsilon_3}(0,0,\epsilon_3) = \frac{\delta}{\delta\epsilon_3}\frac{(3+2x)x}{1+x}(1+\epsilon_3) = \frac{(3+2x)x}{1+x}$$

3. $$\overline{y} \approx y + \frac{(3+2x)x}{1+x}\epsilon_1 + \frac{(3+2x)x}{1+x}\epsilon_2 + \frac{(3+2x)x}{1+x}\epsilon_3$$
$$\overline{y} \approx y + y\epsilon_1 + y\epsilon_2 + y\epsilon_3$$

4. $$\overline{y} - y \approx y\epsilon_1 + y\epsilon_2 + y\epsilon_3$$
$$\delta_s y_2 = \frac{\overline{y}-y}{y} \approx \epsilon_1 + \epsilon_2 + \epsilon_3$$

5. $$\delta_s y_2 \leq 3\epsilon_{mach}$$

Dit algoritme is overal even stabiel en voorwaarts stabiel.

2 Probleem 2

(a)

$$y = x\sin(x)$$

Conditie

$$f'(x) = \sin(x) + x\cos(x)$$
$$\delta_c y \approx \frac{f'(x)x}{f(x)}\delta x = \frac{x\sin(x) + x^2\cos(x)}{x\sin(x)} = 1 + \frac{x\cos(x)}{\sin(x)}$$

Slecht geconditioneerd bij $x = k\pi$. Bij $x = 0$ gaat $\delta_c y$ naar 2.

Stabiliteit

1.
$$\overline{y} = fl(xfl(\sin(x))$$
$$\overline{y} = (x(sin(x))(1+\epsilon_1))(1+\epsilon_2)$$

2.
$$\frac{\delta\overline{y}}{\delta\epsilon_1} = \frac{\delta}{\delta\epsilon_1} = x(sin(x))(1+\epsilon_1) = x(sin(x))$$
$$\frac{\delta\overline{y}}{\delta\epsilon_2} = \frac{\delta}{\delta\epsilon_2} = x(sin(x))(1+\epsilon_2) = x(sin(x))$$

3.
$$\overline{y} \approx y + x(sin(x))\epsilon_1 + x(sin(x))\epsilon_2$$
$$\overline{y} \approx y + y\epsilon_1 + y\epsilon_2$$

4.
$$\overline{y} - y \approx x(sin(x))\epsilon_1 + x(sin(x))\epsilon_2$$
$$\delta_s y = \frac{\overline{y} - y}{y} \approx \epsilon_1 + \epsilon_2$$

5.
$$\delta_s y \leq 2\epsilon_{mach}$$

Dit algoritme is overal even stabiel en voorwaarts stabiel.

(b)

Conditie

$$\frac{\delta f}{\delta a} = \frac{1}{2\sqrt{a}}$$
$$\frac{\delta f}{\delta b} = -\frac{1}{2\sqrt{b}}$$
$$\delta_c y = \frac{\frac{a}{2\sqrt{a}}}{\sqrt{a}-\sqrt{b}}\delta a - \frac{\frac{b}{2\sqrt{b}}}{\sqrt{a}-\sqrt{b}}\delta b = \frac{\sqrt{a}}{2(\sqrt{a}-\sqrt{b})}\delta a - \frac{\sqrt{b}}{2(\sqrt{a}-\sqrt{b})}\delta b$$

Dit probleem is slecht geconditioneerd voor $a \approx b$.

Stabiliteit

$$y = \sqrt{a} - \sqrt{b}$$

1.

$$\overline{y_1} = fl(fl(\sqrt{a}) - fl(\sqrt{b}))$$
$$\overline{y_1} = ((\sqrt{a})(1+\epsilon_1) - (\sqrt{b})(1+\epsilon_2))(1+\epsilon_3)$$

2.

$$\frac{\delta \overline{y_1}}{\delta \epsilon_1}(\epsilon_1, 0, 0) = \frac{\delta}{\delta \epsilon_1}\left(\sqrt{a}(1+\epsilon_1) - \sqrt{b}\right) = \sqrt{a}$$
$$\frac{\delta \overline{y_1}}{\delta \epsilon_2}(0, \epsilon_2, 0) = \frac{\delta}{\delta \epsilon_2}\left(\sqrt{a} - \sqrt{b}(1+\epsilon_2)\right) = -\sqrt{b}$$
$$\frac{\delta \overline{y_1}}{\delta \epsilon_3}(0, 0, \epsilon_3) = \frac{\delta}{\delta \epsilon_3}\left(\sqrt{a} - \sqrt{b}\right)(1+\epsilon_3) = \sqrt{a} - \sqrt{b}$$

3.

$$\overline{y} \approx y + \sqrt{a}\epsilon_1 - \sqrt{b}\epsilon_2 + (\sqrt{a} - \sqrt{b})\epsilon_3$$
$$\overline{y} \approx y + y\frac{\sqrt{a}}{\sqrt{a} - \sqrt{b}}\epsilon_1 - y\frac{\sqrt{b}}{\sqrt{a} - \sqrt{b}}\epsilon_2 + y\epsilon_3$$

4.

$$\delta_s y_1 = \overline{y} - y \approx \sqrt{a}\epsilon_1 - \sqrt{b}\epsilon_2 + (\sqrt{a} - \sqrt{b})\epsilon_3$$
$$\frac{\overline{y} - y}{y} \approx \frac{\sqrt{a}}{\sqrt{a} - \sqrt{b}}\epsilon_1 - \frac{\sqrt{b}}{\sqrt{a} - \sqrt{b}}\epsilon_2 + \epsilon_3$$

5.

$$\delta_s y_1 \leq \frac{2\sqrt{a} + 2\sqrt{b}}{\sqrt{a} - \sqrt{b}}\epsilon_{mach}$$

Dit algoritme is onstabiel wanneer $a \approx b$ geldt.

$$y = \frac{a-b}{\sqrt{a} + \sqrt{b}}$$

1.

$$\overline{y_2} = fl\left(\frac{fl(a-b)}{fl(fl(\sqrt{a}) + fl(\sqrt{b}))}\right)$$
$$\overline{y_2} = \frac{(a-b)(1+\epsilon_4)}{((\sqrt{a})(1+\epsilon_1) + (\sqrt{b})(1+\epsilon_2))(1+\epsilon_3)}(1+\epsilon_5)$$

2.

$$\frac{\delta \overline{y_1}}{\delta \epsilon_1}(\epsilon_1, 0, 0, 0, 0) = \frac{\delta}{\delta \epsilon_1}\frac{a-b}{\sqrt{a}(1+\epsilon_1) + \sqrt{b}} = \frac{\sqrt{a}(a-b)}{(\sqrt{a}(1+\epsilon_1) + \sqrt{b})^2}$$
$$\frac{\delta \overline{y_1}}{\delta \epsilon_1}(0, 0, 0, 0, 0) = \frac{\sqrt{a}(a-b)}{(\sqrt{a} + \sqrt{b})^2}$$
$$\frac{\delta \overline{y_1}}{\delta \epsilon_2}(0, \epsilon_2, 0, 0, 0) = \frac{\delta}{\delta \epsilon_2}\frac{a-b}{\sqrt{a} + \sqrt{b}(1+\epsilon_2)} = \frac{\sqrt{b}(a-b)}{(\sqrt{a} + \sqrt{b}(1+\epsilon_2))^2}$$

$$\frac{\delta\overline{y_1}}{\delta\epsilon_2}(0,0,0,0,0) = \frac{\sqrt{b}(a-b)}{(\sqrt{a}+\sqrt{b})^2}$$

$$\frac{\delta\overline{y_1}}{\delta\epsilon_3}(0,0,\epsilon_3,0,0) = \frac{\delta}{\delta\epsilon_3}\frac{a-b}{(\sqrt{a}+\sqrt{b})(1+\epsilon_3)} = \frac{(a-b)(\sqrt{a}+\sqrt{b})}{((\sqrt{a}+\sqrt{b})(1+\epsilon_3))^2}$$

$$\frac{\delta\overline{y_1}}{\delta\epsilon_3}(0,0,0,0,0) = \frac{(a-b)(\sqrt{a}+\sqrt{b})}{(\sqrt{a}+\sqrt{b})^2} = \frac{a-b}{\sqrt{a}+\sqrt{b}}$$

$$\frac{\delta\overline{y_1}}{\delta\epsilon_4}(0,0,0,\epsilon_4,0) = \frac{\delta}{\delta\epsilon_4}\frac{(a-b)(1+\epsilon_4)}{\sqrt{a}+\sqrt{b}} = \frac{a-b}{\sqrt{a}+\sqrt{b}}$$

$$\frac{\delta\overline{y_1}}{\delta\epsilon_5}(0,0,0,0,\epsilon_5) = \frac{\delta}{\delta\epsilon_5}\frac{a-b}{\sqrt{a}+\sqrt{b}}(1+\epsilon_5) = \frac{a-b}{\sqrt{a}+\sqrt{b}}$$

3.

$$\overline{y} \approx y + \frac{\sqrt{a}(a-b)}{(\sqrt{a}+\sqrt{b})^2}\epsilon_1 + \frac{\sqrt{b}(a-b)}{(\sqrt{a}+\sqrt{b})^2}\epsilon_2 + \frac{a-b}{\sqrt{a}+\sqrt{b}}\epsilon_3 + \frac{a-b}{\sqrt{a}+\sqrt{b}}\epsilon_4 + \frac{a-b}{\sqrt{a}+\sqrt{b}}\epsilon_5$$

$$\overline{y} \approx y + \frac{y\sqrt{a}}{(\sqrt{a}+\sqrt{b})}\epsilon_1 + \frac{y\sqrt{b}}{(\sqrt{a}+\sqrt{b})}\epsilon_2 + y\epsilon_3 + y\epsilon_4 + y\epsilon_5$$

4.

$$\overline{y} - y \approx +\frac{\sqrt{a}(a-b)}{(\sqrt{a}+\sqrt{b})^2}\epsilon_1 + \frac{\sqrt{b}(a-b)}{(\sqrt{a}+\sqrt{b})^2}\epsilon_2 + \frac{a-b}{\sqrt{a}+\sqrt{b}}\epsilon_3 + \frac{a-b}{\sqrt{a}+\sqrt{b}}\epsilon_4 + \frac{a-b}{\sqrt{a}+\sqrt{b}}\epsilon_5$$

$$\delta_s y_2 = \frac{\overline{y}-y}{y} \approx +\frac{\sqrt{a}}{(\sqrt{a}+\sqrt{b})}\epsilon_1 + \frac{\sqrt{b}}{(\sqrt{a}+\sqrt{b})}\epsilon_2 + \epsilon_3 + \epsilon_4 + \epsilon_5$$

5.

$$\delta_s y_2 \leq \frac{4(a-b)}{(\sqrt{a}+\sqrt{b})}\epsilon_{mach}$$

Dit algoritme is stabiel.

3 Probleem 3

Voor de duidelijkheid:

$$\overline{g} = g(x)(x + C_2\epsilon)$$

$$\overline{g}((x+h)(1+\epsilon)) = g(x+h)(1+C_1\epsilon)$$

Foutenanalyse:

1.

$$\overline{g'}(x) = \overline{y} = \frac{\overline{g}((x+h)(1+\epsilon_1)) - \overline{g}(x)}{h}(1+\epsilon_2)(1+\epsilon_3)$$

$$= \frac{g(x+h)(1+C_1\epsilon_1) - g(x)(x+C_2\epsilon_4)}{h}(1+\epsilon_2)(1+\epsilon_3)$$

2.

$$\frac{\delta \overline{y}}{\delta \epsilon_1}(0,0,0,0) = \frac{g(x+h)}{h}$$
$$\frac{\delta \overline{y}}{\delta \epsilon_2}(0,0,0,0) = -\frac{g(x)}{h}$$
$$\frac{\delta \overline{y}}{\delta \epsilon_3}(0,0,0,0) = \frac{g(x+h)-g(x)}{h}$$
$$\frac{\delta \overline{y}}{\delta \epsilon_4}(0,0,0,0) = \frac{g(x+h)-g(x)}{h}$$

3.

$$\overline{y} \approx y + \frac{g(x+h)}{h}\epsilon_1 - \frac{g(x)}{h}\epsilon_2 + \frac{g(x+h)-g(x)}{h}\epsilon_3 + \frac{g(x+h)-g(x)}{h}\epsilon_4$$
$$\overline{y} \approx y + \frac{g(x+h)}{h}\epsilon_1 - \frac{g(x)}{h}\epsilon_2 + y\epsilon_3 + y\epsilon_4$$

4.

$$\overline{y} - y \approx \frac{g(x+h)}{h}\epsilon_1 - \frac{g(x)}{h}\epsilon_2 + y\epsilon_3 + y\epsilon_4$$
$$\frac{\overline{y}-y}{y} \approx \frac{g(x+h)}{g(x+h)-g(x)}\epsilon_1 - \frac{g(x)}{g(x+h)-g(x)}\epsilon_2 + \epsilon_3 + \epsilon_4$$

Middelwaardestelling

$$\frac{\overline{y}-y}{y} \approx \frac{g(x+h)}{g'(\xi)h}\epsilon_1 - \frac{g(x)}{g'(\xi)h}\epsilon_2 + \epsilon_3 + \epsilon_4$$

De relatieve fout is dus inderdaad van orde $\frac{1}{h}$

4 Probleem 4

Zij y_1 en y_1 de wortels van deze vergelijking.

$$y_1 = \frac{2+\sqrt{4-4t}}{2} y_2 = \frac{2-\sqrt{4-4t}}{2}$$
$$\frac{dy_1}{dt} = \frac{1}{\sqrt{4-4t}} \text{ en } \frac{dy_2}{dt} = -\frac{1}{\sqrt{4-4t}}$$
$$\delta_c y_2 \approx \frac{y_1' t}{y_1}\delta t = (2\sqrt{1-t}+2-2t)\delta t$$
$$\delta_c y_2 \approx \frac{y_2' t}{y_2}\delta t = (-2\sqrt{1-t}+2-2t)\delta t$$

REDO, foutje!

5 Probleem 5

5.1 (a)

5.1.1 Conditie

$$y' = 2x$$
$$\delta_c y = \frac{2x^2}{2x+x^2}\delta x = \frac{2x}{x+2}\delta x$$

Dit probleem is goed geconditioneerd voor kleine x.

5.1.2 Stabiliteit

Eval1:

1.
$$\overline{y} = fl\left(fl\left(fl(1+x)^2\right)-1\right)$$
$$\overline{y} = (((1+x)(1+\epsilon_1))^2(1+\epsilon_2)-1)(1+\epsilon_3)$$

2.
$$\frac{\delta\overline{y}}{\delta\epsilon_1}(\epsilon_1,0,0) = \frac{\delta}{\delta\epsilon_1}(((1+x)(1+\epsilon_1))^2-1) = 2(1+x)^2(1+\epsilon_1)$$
$$\frac{\delta\overline{y}}{\delta\epsilon_1}(0,0,0) = 2(1+x)^2$$
$$\frac{\delta\overline{y}}{\delta\epsilon_2}(0,\epsilon_2,0) = \frac{\delta}{\delta\epsilon_2}((1+\epsilon_2)(1+x)^2-1) = (1+x)^2$$
$$\frac{\delta\overline{y}}{\delta\epsilon_3}(0,0,\epsilon_3) = \frac{\delta}{\delta\epsilon_3}((1+x)^2-1)(1+\epsilon_3) = (1+x)^2-1$$

3.
$$\overline{y} \approx y + 2(1+x)^2\epsilon_1 + (1+x)^2\epsilon_2 + (1+x)^2 - 1\epsilon_3$$
$$\overline{y} \approx y + y\frac{2(1+x)^2}{(1+x)^2-1}\epsilon_1 + y\frac{(1+x)^2}{(1+x)^2-1}\epsilon_2 + y\epsilon_3$$

4.
$$\overline{y} - y \approx 2(1+x)^2\epsilon_1 + (1+x)^2\epsilon_2 + (1+x)^2 - 1\epsilon_3$$
$$\frac{\overline{y}-y}{y} \approx +\frac{2(1+x)^2}{(1+x)^2-1}\epsilon_1 + \frac{(1+x)^2}{(1+x)^2-1}\epsilon_2 + \epsilon_3$$

5. Dit algoritme is onstabiel voor $x \approx 0$

Eval2:

1.
$$\overline{y} = fl(xfl(2+x))$$
$$\overline{y} = x(2+x)(1+\epsilon_1)(1+\epsilon_2)$$

2.
$$\frac{\delta\overline{y}}{\delta\epsilon_1}(\epsilon_1,0) = \frac{\delta}{\delta\epsilon_1}x(2+x)(1+\epsilon_1) = x(2+x)$$
$$\frac{\delta\overline{y}}{\delta\epsilon_2}(0,\epsilon_2) = \frac{\delta}{\delta\epsilon_2}x(2+x)(1+\epsilon_2) = x(2+x)$$

3.
$$\overline{y} \approx y + x(2+x)\epsilon_1 + x(2+x)\epsilon_2$$
$$\overline{y} \approx y + y\epsilon_1 + y\epsilon_2$$

4.
$$\overline{y} - y \approx +x(2+x)\epsilon_1 + x(2+x)\epsilon_2$$
$$\frac{\overline{y}-y}{y} \approx \epsilon_1 + \epsilon_2$$

5. Dit algoritme is overal even stabiel en voorwaarts stabiel.

5.2 (b)

$$y = \frac{1}{e^x - 1} - \frac{1}{e^x + 1} = \frac{2}{e^{2x} - 1}$$

5.2.1 Conditie

$$y' = \frac{-4e^{2x}}{(e^{2x} - 1)^2}$$

$$\delta_c y \approx \frac{-4xe^{2x}}{(e^{2x} - 1)^2} \frac{e^{2x} - 1}{2} = \frac{-2xe^{2x}}{e^{2x} - 1}$$

Dit probleem is slecht geconditioneerd voor $x \approx 0$.

5.2.2 Stabiliteit

Eval1

1. $$\overline{y} = fl\left(\frac{1}{fl(fl(e^x) - 1)}\right) - fl\left(\frac{1}{fl(fl(e^x) + 1)}\right)$$

$$\overline{y} = \left(\frac{1 + \epsilon_3}{(e^x(1 + \epsilon_1) - 1)(1 + \epsilon_2)} - \frac{1 + \epsilon_5}{(e^x(1 + \epsilon_1) + 1)(1 + \epsilon_4)}\right)(1 + \epsilon_6)$$

   LOL NOPE

Eval2

1. $$\overline{y} = fl\left(2fl\left(\frac{1}{fl\left(fl\left(e^{fl(2x)}\right) - 1\right)}\right)\right)$$

$$\overline{y} = \frac{2(1 + \epsilon_4)(1 + \epsilon_5)}{\left((1 + \epsilon_2)e^{2x(1+\epsilon_1)} - 1\right)(1 + \epsilon_3)}$$

   LOL NOPE

Oefeningen Numerieke Wiskunde: Oefenzitting 5

Tom Sydney Kerckhove

11 maart 2014

1 Probleem 1

Zie de matlab scripts. Voor de eerste formule bekomen we maximaal 8 juiste beduidende cijfers, voor de tweede bekomen we er maximaal 11. We plotten de relatieve fout ten opzichte van h in een loglog-plot en zien duidelijk twee componenten in de fout. Eenderzijds links de afrondingsfouten en anderzijds rechts de benaderingsfouten.

2 Probleem 2

De ordes van de benaderingen zijn respectievelijk 1 en 2. Ontwikkel $f(x+h)$ en $f(x-h)$

$$f(x+h) = f(x) + f'(x)h + f''(x)h^2 + \cdots$$

$$f(x-h) = f(x) - f'(x)h + f''(x)h^2 + \cdots$$

Vul nu in:
Voor y_1:

$$y_1 = \frac{f(x+h) - f(x)}{h} = \frac{f(x) + f'(x)h + f''(x)h^2 + \cdots - f(x)}{h} = f'(x) + f''(x)h + \cdots$$

$$= f'(x) + O(h)$$

Voor y_2:

$$y_2 = \frac{f(x+h) - f(x-h)}{2h}$$

$$= \frac{f(x) + f'(x)h + f''(x)h^2 + f'''(x)h^3 + \cdots - f(x) + f'(x)h - f''(x)h^2 + +f'''(x)h^3 + \cdots}{2h}$$

$$f'(x) + f'''(x)h^2 + \cdots = f'(x) + O(h^2)$$

3 Probleem 3

Het bewijs is volledig analoog.

4 Probleem 4

Zie matlab: er is geen verschil.

Oefeningen Numerieke Wiskunde: Oefenzitting 6

Tom Sydney Kerckhove

31 maart 2014

1 Probleem 1

(a)

$$f(-1) = -7 \text{ en } f(1) = 1$$

Lagrange

Bereken eerste de lagrange basis.

$$l_0(x) = \prod_{j=0, j\neq 0}^{1} \frac{x - x_j}{x_0 - x_j} = \frac{x-1}{-1-1} = -\frac{1}{2}x + \frac{1}{2}$$

$$l_1(x) = \prod_{j=0, j\neq 1}^{1} \frac{x - x_j}{x_1 - x_j} = \frac{x+1}{1+1} = \frac{1}{2}x + \frac{1}{2}$$

De interpolerende veelterm is dan de volgende:

$$y_1(x) = \sum_{i=0}^{1} l_i(x) f(x_i) = \left(-\frac{1}{2}x + \frac{1}{2}\right)(-7) + \left(\frac{1}{2}x + \frac{1}{2}\right)(1) = 4x - 3$$

Newton

Bereken eerst de newton basis.

$$b_0 = f[x_0] = f(x_0) = -7$$

$$b_1 = f[x_0, x_1] = \frac{f(x_1) - f(x_0)}{x_1 - x_0} = 4$$

De interpolerende veelterm is dan de volgende:

$$y_1(x) = \sum_{i=0}^{1} b_i \prod_{j=0}^{i} (x - x_j) = b_0 + b_1(x - x_0) = -7 + 4(x+1) = 4x - 3$$

(b)

Het vandermonde stelsel:

$$\begin{cases} a_0 - 1a_1 &= -7 \\ a_0 + a_2 &= 1 \end{cases}$$

Test:

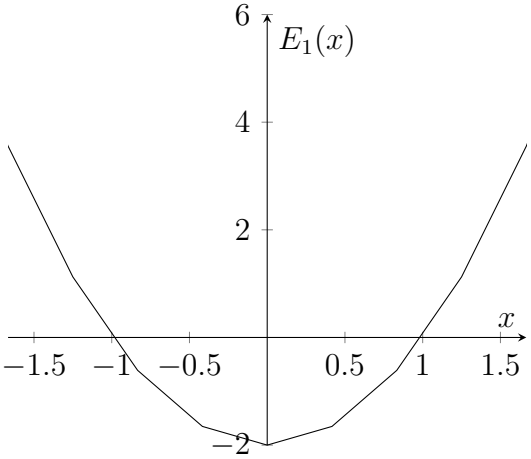
$$\begin{cases} -3 - 4 &= -7 \\ -3 + 4 &= 1 \end{cases} \rightarrow OK$$

(c)

$$E_1(x) = \frac{f^{(2)}(\zeta)}{2!}(x - x_0)(x - x_1)$$

We weten dat $f_{(2)}(x) = 4$ constant is.

$$E_1(x) = \frac{4}{2}(x + 1)(x - 1) = 2x^2 - 2$$



Buiten $[-1, 1]$ gedraagt $E_1(x)$ zicht als iets van $O(x^2)$.

2 Probleem 2

De tabel del gedeelde differentiaties:

$$\begin{array}{c|lll} -1 & -7 & & \\ 0 & -5 & \frac{-5+7}{0+1} = 2 & \\ 1 & 1 & \frac{1+5}{1+0} = 6 & \frac{6-2}{1+1} = 2 \end{array}$$

$$y_2(x) = \sum_{i=0}^{2} f[x_0, ..., x_i] \prod_{j=0}^{i-1}(x - x_j) = -7 + 2(x + 1) + 2(x + 1)(x - 0) = 2x^2 + 4x - 5$$

Alternatieve tabel:

$$\begin{array}{c|lll} -1 & -7 & & \\ 0 & -5 & \frac{-5+7}{0+1} = 2 & \\ 1 & 1 & \frac{1+7}{1+1} = \frac{8}{2} & = 2 \end{array}$$

$E_2(x) = 0$ omdat y en y_2 beide van graad twee zijn en y_2 uniek is.

3 Probleem 3

$$\begin{array}{c|llll} -1 & -7 & & & \\ 0 & -5 & 2 & & \\ 0.5 & -2.5 & 5 & 2 & \\ 1 & 1 & 7 & 2 & 0 \end{array}$$

$$y_3(x) = \sum_{i=0}^{3} f[x_0, ..., x_i] \prod_{j=0}^{i-1}(x - x_j)$$

$$= -7 + 2(x+1) + 2(x+1)(x-0) + 0(x+1)(x-0)(x-0.5) = 2x^2 + 4x - 5$$

Alternatieve tabel:

-1	-7			
0	-5	2		
0.5	-2.5	3	2	
1	1	4	2	0

We zien dat deze veelterm van graad twee is. We kunnen immers geen interpolerende veelterm opstellen van graad drie voor een veelterm van graad twee.

4 Probleem 4

1.

$$\sum_{i=1}^{n} \prod_{j=0, j\neq i}^{n} \frac{x - x_j}{x_i - x_j} = 1$$

Niet zomaar beginnen uitrekenen, dat wordt een hel.

Bewijs. Stel de interpolerende veelterm (van graad n) op van de constante functie $y = 1$.

$$y_n = \sum_{i=0}^{n} l_i(x) f(x_i)$$

We weten dat $f(x_i) = 1$ geldt voor elke i.

$$y_n = \sum_{i=0}^{n} l_i(x) \text{ en } y_n(x) = 1$$

We weten dat y_n y exact benadert omdat y van graad 0 is en $n \geq 0$.

$$y_n = \sum_{i=0}^{n} l_i(x) = 1$$

□

2.

$$\sum_{i=1}^{n} l_i(x) x_i^k = x^k \text{ met } k \leq n$$

Bewijs. Als we de interpolerende veelterm opstellen van $y = x^k$ wordt deze exact benaderd omdat $k \leq n$. □

5 Probleem 9

Bewijs. Bewijs in delen

- (1) → (2) Triviaal: als het voor elke functie geldt geldt het ook voor een veelterm.

- (2) $\rightarrow$ (3) We weten:

$$\int_a^b p(x)dx = \sum_{i=0}^{n} H_i f(x_i)$$

$$p(x) = \sum_{i=0}^{n} l_i(x) f(x_i) =$$

$$\int_a^b p(x)dx = \int_a^b \sum_{i=0}^{n} l_i(x) f(x_i)$$

$$= \int_a^b \sum_{i=0}^{n} l_i(x) f(x_i)dx = \sum_{i=0}^{n} f(x_i) \int_a^b l_i(x)dx$$

$$\Rightarrow H_i = \int_a^b l_i(x)dx$$

- (3) $\rightarrow$ (1)

$$\sum_{i=0}^{n} H_i f(x_i) = \sum_{i=0}^{n} f(x_i) \int_a^b l_i(x)dx = \int_a^b \sum_{i=0}^{n} l_i(x) f(x_i)dx = \int_a^b y_n dx$$

□

6 Probleem 10

We weten dat de gewichten H_i de integralen zijn van de lagrange veeltermen l_i.

$$H_i = \int_a^b l_i dx$$

We kunnen de gewichten dus gewoon berekenen.

$$H_0 = \int_{-1}^{1} l_0 dx = \int_{-1}^{1} \frac{x - \frac{1}{2}}{-\frac{1}{2} - \frac{1}{2}} = \int_{-1}^{1} \left(\frac{1}{2} - x\right) dx = 1$$

$$H_1 = \int_{-1}^{1} l_1 dx = \int_{-1}^{1} \frac{x + \frac{1}{2}}{\frac{1}{2} + \frac{1}{2}} = \int_{-1}^{1} \left(\frac{1}{2} + x\right) dx = 1$$

We kunnen nu nakijken tot welke graad de integraal exact wordt berekend.

$$(d = 0): \ 1 + 1 = 2 = \int_{-1}^{1} dx \rightarrow OK$$

$$(d = 1): \ -\frac{1}{2} + \frac{1}{2} = 0 = \int_{-1}^{1} x dx \rightarrow OK$$

$$(d = 2): \ \frac{1}{4} + \frac{1}{4} = \frac{1}{2} \neq \int_{-1}^{1} x^2 dx \rightarrow NOK$$

De nauwkeurigheidsgraad is dus 1. Nauwkeurigheidsgraad $= 1 \geq n$ dus interpolerend.

7 Probleem 11

Alternatieve methode

$$\begin{cases} H_0 & +H_1 & = \int_{-1}^{1} dx & = 2 \\ -\frac{1}{2}H_0 & +\frac{1}{2}H_1 & = \int_{-1}^{1} x dx & = 0 \end{cases}$$

waarom werkt dit?

We vinden voor H_0 en H_1 beide 1.

Oefeningen Numerieke Wiskunde: Oefenzitting 7

Tom Sydney Kerckhove

22 maart 2014

Oefeningen Numerieke Wiskunde: Oefenzitting 9

Tom Sydney Kerckhove

29 maart 2014

1 Probleem 1

$$a_0 t_1^3 + b_0 t_1^2 + c_0 t_1 + d_0 - a_1 t_1^3 + b_1 t_1^2 + c_1 t_1 + d_1 = 0$$

Het stelsel A ziet er dus als volgt uit.

$$A = \begin{pmatrix}
t_1^3 & t_1^2 & t_1 & 1 & -t_1^3 & -t_1^2 & -t_1 & -1 & 0 & 0 & 0 & 0 & \cdots \\
0 & 0 & 0 & 0 & t_2^3 & t_2^2 & t_2 & 1 & -t_2^3 & -t_2^2 & -t_2 & -1 & \cdots \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & t_3^3 & t_3^2 & t_3 & 1 & \cdots \\
\vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \\
3t_1^2 & 2t_1 & 1 & 0 & -3t_1^2 & -2t_1 & -1 & 0 & 0 & 0 & 0 & 0 & \cdots \\
0 & 0 & 0 & 0 & 3t_2^2 & 2t_2 & 1 & 0 & -3t_2 & -2t_2 & -1 & 0 & \cdots \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 3t_3^2 & 2t_3 & 1 & 0 & \cdots \\
\vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \\
6t_1 & 2 & 0 & 0 & -6t_1 & -2 & 0 & 0 & 0 & 0 & 0 & 0 & \cdots \\
0 & 0 & 0 & 0 & 6t_2 & 2 & 0 & 0 & -6t_2 & -2 & 0 & 0 & \cdots \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 6t_3 & 2 & 0 & 0 & \cdots \\
\vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots &
\end{pmatrix}$$

2 Probleem 2

Zie matlab

3 Probleem 3

evalsplin

4 Probleem 4

Zie matlab

5 Probleem 5

fix scripts

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.

Spline-functies

1 Definities

In het hoofdstuk over veelterminterpolatie hebben we gezien dat de interpolatiefout niet altijd kleiner wordt voor toenemende graad van de interpolerende veelterm. Om convergentie te garanderen zullen we functies benaderen niet met veeltermen maar met zogenaamde *spline-functies*.

Definitie 1 (spline functie) *Zij $k \in \mathbb{N} = \{0, 1, 2, \ldots\}$ en $a = t_0 < t_1 < \cdots < t_n = b$ met $t_i \in \mathbb{R}$. Dan is de functie $s(x)$ een* spline *van graad k (of van orde $k+1$) over het interval $[a, b]$ met als knooppunten $t_0, t_1, \ldots, t_n$ als en slechts als*

- *$s(x)$ over elk deelinterval $[t_j, t_{j+1})$, $j = 0(1)n-2$ en over het interval $[t_{n-1}, t_n]$ een veelterm is van graad $\leq k$;*
- *alle afgeleiden $s^{(j)}(x)$, $j = 0(1)k-1$ continu zijn over het interval $[a, b]$.*

Voor $k = 0$ is automatisch aan de tweede voorwaarde voldaan. Dus een spline van graad 0 is een stapfunctie. Een spline van graad 1 is een continue gebroken lijn.

Merk op dat de verzameling van alle veeltermen (over het interval $[a, b]$) van graad $\leq k$ een deelverzameling is van de verzameling van de splines van graad k.

Voor een gegeven stel knooppunten is de verzameling van de splines van graad k een vectorruimte. We zullen deze vectorruimte noteren als $\mathcal{V}_k(t_0, t_1, \ldots, t_n)$.

2 De voorstelling van splines

Volgens de definitie wordt de spline $s(x)$ in elk deelinterval $[t_j, t_{j+1})$ gegeven door een veelterm. D.w.z. dat we in elk deelinterval de spline kunnen voorstellen door een voorstelling te nemen van de corresponderende veelterm. Een voor de hand liggende voorstelling is de klassieke voorstelling

$$s(x) = p_j(x) = a_{j,k}x^k + a_{j,k-1}x^{k-1} + \cdots + a_{j,0}, \quad \forall x \in [t_j, t_{j+1}), \quad j = 0(1)n-1.$$

De continuïteitsvoorwaarden zorgen voor $k(n-1)$ lineair onafhankelijke vergelijkingen tussen de $n(k+1)$ coëfficiënten $a_{j,i}$ van de veeltermvoorstellingen. Hieruit volgt dat de dimensie van de vectorruimte $\mathcal{V}_k(t_0, t_1, \ldots, t_n)$ gelijk is aan $n(k+1) - k(n-1) = n + k$. In wat volgt gaan we op zoek naar een basis voor $\mathcal{V}_k(t_0, t_1, \ldots, t_n)$ wat ons een voorstelling oplevert waarbij de

koördinaten in tegenstelling tot de coëfficiënten $a_{j,i}$ onafhankelijk van elkaar kunnen gekozen worden. Een eerste basis wordt gevormd door gebruik te maken van afgeknotte machtsfuncties. In de volgende sectie zullen we een basis opstellen met behulp van de zogenaamde B-splines.

Definitie 2 (afgeknotte machtsfuncties) *De afgeknotte machtsfunctie x_+^k van graad k wordt gedefiniëerd als*

$$\begin{aligned} x_+^k &= x^k, \qquad \textit{voor } x > 0 \\ &= 0, \qquad \textit{voor } x \leq 0 \end{aligned}$$

Merk op dat voor elk van de knooppunten t_j de functie $(t_j - x)_+^k$ ook een spline van graad k is. Het is nu eenvoudig na te gaan dat elke spline $s(x)$ van graad k met als knooppunten $t_0, t_1, \ldots, t_n$ kan voorgesteld worden als

$$s(x) = \sum_{i=0}^{k} a_j x^j + \sum_{j=1}^{n-1} c_j (t_j - x)_+^k.$$

Hieruit volgt dat de verzameling functies

$$\left\{1, x, x^2, \ldots, x^k, (t_1 - x)_+^k, \ldots, (t_{n-1} - x)_+^k\right\}$$

een basis vormt voor de vectorruimte $\mathcal{V}_k(t_0, t_1, \ldots, t_n)$. Het gebruik van deze basis zorgt echter voor numeriek onstabiele berekeningen. Daarom gaan we in de volgende sectie op zoek naar een andere basis gebaseerd op B-splines die resulteert in meer nauwkeurige resultaten.

3 B-splines

We hebben gedeelde differenties gebruikt bij de Newton-voorstelling van de interpolerende veelterm. We herhalen de definitie.

Definitie 3 (gedeelde differentie) *Zij gegeven de koppels $(x_i, f_i) \in \mathbb{R}^2$ voor $i = 0(1)n$ waarbij de x_i allemaal verschillend zijn van elkaar. Dan worden de gedeelde differenties op een recursieve manier gedefiniëerd als*

- $f[x_i] = f_i$, $i = 0(1)n$ *(gedeelde differentie van orde 0);*
- $f[x_i, x_{i+1}, \ldots, x_{i+k}] = \dfrac{f[x_i, x_{i+1}, \ldots, x_{i+k-1}] - f[x_{i+1}, x_{i+2}, \ldots, x_{i+k}]}{x_i - x_{i+k}}$, $k = 1(1)n$, $i = 0(1)n - k$ *(gedeelde differentie van orde k).*

We zullen de volgende twee eigenschappen van gedeelde differenties nodig hebben.

Eigenschap 1 (eigenschappen van gedeelde differenties)

1. *Met* $\pi(x) = \prod_{i=0}^{n}(x - x_i)$, *krijgen we*

$$f[x_0, x_1, \ldots, x_n] = \sum_{j=0}^{n} \frac{f_j}{\pi'(x_j)}. \tag{1}$$

2. *(formule van Leibnitz) Als* $f(x) = g(x)h(x)$, *geldt er*

$$f[x_0, x_1, \ldots, x_n] = \sum_{j=0}^{n} g[x_0, x_1, \ldots, x_j]\, h[x_j, x_{j+1}, \ldots, x_n]. \tag{2}$$

Wanneer we een functie $f(t, x)$ hebben in twee veranderlijken, gebruiken we de notatie $\Delta_x^k[x_i, x_{i+1}, \ldots, x_{i+k}]f(t, x)$ voor de gedeelde differentie voor de koppels $(x_j, f(x_j, t))$ en $\Delta_t^k[t_i, t_{i+1}, \ldots, t_{i+k}]f(t, x)$ voor de koppels $(t_i, f(x, t_i))$. Merk op dat $\Delta_x^k[x_i, x_{i+1}, \ldots, x_{i+k}]f(t, x)$ een functie is in t en $\Delta_t^k[t_i, t_{i+1}, \ldots, t_{i+k}]f(t, x)$ een functie in x.

Definitie 4 (B-spline) *Een B-spline van graad k is elke functie*

$$M_{i,k}(x) = \Delta_t^{k+1}[t_i, t_{i+1}, \ldots, t_{i+k+1}](t - x)_+^k.$$

We zullen de B-splines normaliseren:

$$N_{i,k}(x) = (t_{i+k+1} - t_i)M_{i,k}(x).$$

Door deze normalisatie geldt

$$\sum_i N_{i,k}(x) \equiv 1.$$

Uit formule (1) volgt dat

$$M_{i,k}(x) = \sum_{j=0}^{k+1} \frac{(t_{i+j} - x)_+^k}{\pi'_{i,k+1}(t_{i+j})}$$

met $\pi_{i,k+1}(t) = \prod_{j=0}^{k+1}(t - t_{i+j})$.
Hieruit volgt dat B-splines ook splines zijn.

4 De B-spline voorstelling van spline-functies

Met de knooppunten $t_0, t_1, \ldots, t_n$ kunnen we $n-k$ verschillende B-splines van graad k definiëren, nl.

$$N_{i,k}(x) \quad \text{voor} \quad i = 0(1)n-k-1.$$

Deze $n-k$ B-splines moeten we nog aanvullen met $2k$ bijkomende splines om een basis voor $\mathcal{V}_k(t_0, t_1, \ldots, t_n)$ te bekomen. Daarom voegen we zowel links als rechts van het interval nog k knooppunten toe:

$$\begin{aligned} t_{-k} < t_{-k+1} < \ldots < t_0 \\ t_n < t_{n+1} < \ldots < t_{n+k} \ . \end{aligned}$$

Dan kunnen we elke spline $s(x)$ van graad k met als knooppunten $t_0, t_1, \ldots, t_n$ op een unieke wijze schrijven als

$$s(x) = \sum_{j=-k}^{n-1} c_j N_{j,k}(x).$$

5 Eigenschappen van B-splines

De recursieve definitie van gedeelde differenties samen met formule (2) leidt tot de volgende recursiebetrekkingen voor B-splines.

Eigenschap 2 (recursiebetrekking)

$$\begin{aligned} M_{i,k}(x) &= \frac{x-t_i}{t_{i+k+1}-t_i} M_{i,k-1}(x) + \frac{t_{i+k+1}-x}{t_{i+k+1}-t_i} M_{i+1,k-1}(x) \ . \\ N_{i,k}(x) &= \frac{x-t_i}{t_{i+k}-t_i} N_{i,k-1}(x) + \frac{t_{i+k+1}-x}{t_{i+k+1}-t_{i+1}} N_{i+1,k-1}(x) \ . \end{aligned}$$

Voor het evalueren van splines is het van belang dat B-splines verschillend zijn van nul in een beperkt interval.

Eigenschap 3

$$M_{i,k}(x) = 0 \quad \textit{als} \quad x < t_i \quad \textit{of} \quad x \geq t_{i+k+1} \ .$$

Bewijs: Omdat $M_{i,k}(x)$ een gedeelde differentie is in de punten $t_i, t_{i+1}, \ldots, t_{i+k+1}$ kunnen we formule (1) toepassen:

$$M_{i,k}(x) = \sum_{j=0}^{k+1} \frac{(t_{i+j}-x)_+^k}{\pi'(t_{i+j})}$$

met

$$\pi(t) = \prod_{j=0}^{k+1} (t - t_{i+j}) \ .$$

Als $x \geq t_{i+k+1}$ zijn alle afgeknotte machtsfuncties $(t_{i+j} - x)_+^k$ gelijk aan nul. Als $x < t_i$ geldt dat

$$(t - x)_+^k = (t - x)^k \quad , \quad t = t_i, t_{i+1}, \ldots, t_{i+k+1} \ .$$

De B-spline $M_{i,k}(x)$ is gedefinieerd als

$$M_{i,k}(x) = \Delta_t^{k+1}(t_i, t_{i+1}, \ldots, t_{i+k+1})(t - x)_+^k \ ,$$

d.w.z. als een gedeelde differentie in de punten

$$t_i, t_{i+1}, \ldots, t_{i+k+1} \quad \text{voor de functie} \quad (t - x)_+^k \ .$$

Wanneer $x < t_i$ kunnen we $M_{i,k}(x)$ ook schrijven als

$$M_{i,k}(x) = \Delta_t^{k+1}(t_i, \ldots, t_{i+k+1})(t - x)^k \ ,$$

wat een gedeelde differentie is van orde $k+1$ van een veelterm van graad k. Hieruit volgt dat $M_{i,k}(x) = 0$ voor $x < t_i$. □

Eigenschap 4

$$M_{i,k}(x) > 0 \qquad \textit{als } t_i < x < t_{i+k+1}$$

Bewijs: We bewijzen de eigenschap door volledige inductie.

De eigenschap geldt voor $M_{i,0}$ want

$$\begin{aligned} M_{i,0}(x) &= \frac{1}{t_{i+1} - t_i} \qquad \text{voor} \quad t_i \leq x < t_{i+1} \\ &= 0 \qquad \text{voor} \quad x < t_i \quad \text{of} \quad x \geq t_{i+1} \ . \end{aligned}$$

Dus $M_{i,0}(x)$ is strikt positief tussen t_i en t_{i+1}.

De recursiebetrekking voor B-splines kan geschreven worden als

$$M_{i,k}(x) = \alpha \ M_{i,k-1}(x) + \beta \ M_{i+1,k-1}(x)$$

met

$$\alpha = \frac{x - t_i}{t_{i+k+1} - t_i} \quad \text{en} \quad \beta = \frac{t_{i+k+1} - x}{t_{i+k+1} - t_i} \ .$$

Als $t_i < x < t_{i+k+1}$, geldt er

$$\begin{aligned} \alpha &> 0 \\ \beta &> 0 \\ \alpha + \beta &= 1. \end{aligned}$$

M.a.w. $M_{i,k}$ is een strikt convexe combinatie van $M_{i,k-1}$ en $M_{i+1,k-1}$, dus ook strikt positief. □

Uit eigenschap 3 volgt dat in elk punt x maximum $k+1$ B-splines van graad k verschillend zijn van nul.

De normalisatievoorwaarden kunnen dus ook geschreven worden als

$$\sum_i N_{i,k}(x) = \sum_{i=j-k}^{j} N_{i,k}(x) \equiv 1 \quad \text{als} \quad t_j \leq x < t_{j+1} \ .$$

6 Evalueren van spline-functies

We willen de waarde berekenen van de spline-functie $s(x)$ gegeven in zijn B-spline voorstelling

$$s(x) = \sum_{i=-k}^{n-1} c_i N_{i,k}(x) \ . \tag{3}$$

We nemen aan dat $t_j \leq x < t_{j+1}$.
Van de $n+k$ basisfuncties $N_{i,k}(x)$ in de som van formule (3) zijn er maximum $k+1$ verschillend van nul, nl.

$$N_{i,k}(x) \qquad , \qquad i = j - k(1)j \ .$$

Hieruit volgt dat

$$s(x) = \sum_{i=j-k}^{j} c_i N_{i,k}(x) \quad \text{als} \quad t_j \leq x < t_{j+1} \ .$$

Een eerste mogelijkheid om $s(x)$ te evalueren bestaat erin om de waarden $N_{i,k}(x)$, $i = j-k(1)j$ te berekenen door $k+1$ differentietabellen op te stellen. Voor elke $N_{i,k}(x)$ ziet zo'n differentietabel eruit als volgt:

$$\begin{array}{cccc} (t_i - x)_+^k & & & \\ (t_{i+1} - x)_+^k & \ddots & & \\ \vdots & \ddots & \ddots & \\ (t_{i+k+1} - x)_+^k & \cdots & \cdots & \frac{N_{i,k}(x)}{t_{i+k+1} - t_i} \end{array} \ .$$

Deze methode is echter numeriek onstabiel.

Een tweede methode maakt gebruik van de recursiebetrekking voor B-splines om $N_{i,k}(x)$, $i = j - k(1)j$ uit te rekenen. We weten dat er maximum slechts $l+1$ B-splines van graad l verschillend zijn van nul voor een vaste waarde van x. Hieruit volgt dat er een driehoekige tabel overblijft met volgende elementen:

$$\begin{array}{lllll} M_{j,0}(x) = 1/(t_{j+1} - t_j) & M_{j-1,1}(x) & M_{j-2,2}(x) & \ldots & M_{j-k,k}(x) \\ & M_{j,1}(x) & M_{j-1,2}(x) & \ldots & M_{j-k+1,k}(x) \\ & & M_{j,2}(x) & \ldots & M_{j-k+2,k}(x) \\ & & & \ddots & \vdots \\ & & & & M_{j,k}(x) \end{array}$$

Deze methode is numeriek stabieler want er worden altijd convexe combinaties van positieve getallen gevormd.

Een derde methode (algoritme van de Boor (1972)) berekent de driehoekige tabel

$$\begin{array}{llll} c_j^{[0]}(x) & & & \\ c_{j-1}^{[0]}(x) & c_j^{[1]}(x) & & \\ \vdots & \vdots & \ddots & \\ c_{j-k}^{[0]}(x) & c_{j-(k-1)}^{[1]}(x) & \ldots & c_j^{[k]}(x) \end{array}$$

met

$$c_j^{[i]}(x) = \begin{cases} c_j & \text{als} \quad i = 0 \\ \dfrac{c_j^{[i-1]}(x)(x - t_j) + c_{j-1}^{[i-1]}(x)(t_{j+k+1-i} - x)}{t_{j+k+1-i} - t_j} & \text{als} \quad i > 0 \; . \end{cases}$$

De waarde van $s(x)$ is dan

$$s(x) = \sum_{i=-k}^{n-1} c_i N_{i,k}(x) = c_j^{[k]}(x) \; .$$

De laatste methode kunnen we ook gebruiken om de waarde van $N_{i,k}(x)$ te berekenen door $c_i = 1$ te stellen en alle andere $c_j = 0$, $j \neq i$.

7 Interpolatie met kubische splines

In deze sectie zullen we de tabel van punten $\{(x_i, f_i)\}$, $i = 0(1)n$ interpoleren met een spline-functie $s(x)$ i.p.v. met een veelterm $y_n(x)$ van graad n. We zullen de knooppunten t_i gelijk nemen aan de x_i-waarden: $t_i = x_i$, $i = 0(1)n$. In sectie 2, hebben we gezien dat de dimensie van de vectorruimte

$\mathcal{V}_k(t_0, t_1, \ldots, t_n)$ van de splines van graad k gelijk is aan $n + k$. Dit betekent dat we naast de $n + 1$ interpolatievoorwaarden

$$s(x_i) = f_i, \qquad i = 0(1)n \tag{4}$$

nog $k - 1$ bijkomende voorwaarden moeten opleggen om de spline $s(x)$ uniek te kunnen bepalen.

Dikwijls wordt de graad k van de interpolerende spline-functie $s(x)$ gelijk aan drie genomen. Dit wil zeggen dat de spline, de eerste en de tweede afgeleide van de spline continue functies zijn. Het menselijk oog ervaart de spline-functie dan als een vloeiende kromme. De splines van graad drie worden *kubische splines* genoemd. Naast de $n + 1$ interpolatievoorwaarden hebben we dan nog $k - 1 = 2$ bijkomende voorwaarden nodig. Meestal wordt er een *natuurlijke* of een *periodieke* spline genomen.

- Een natuurlijke spline-functie voldoet aan de volgende twee voorwaarden:
$$s^{(2)}(t_0) = s^{(2)}(t_n) = 0.$$

- Een periodieke kubische spline-functie voldoet aan de volgende twee voorwaarden:
$$s^{(j)}(t_0) = s^{(j)}(t_n), \qquad \text{voor} \quad j = 1, 2.$$
Men veronderstelt dan ook meestal dat $f_0 = f_n$ zodat ook $s(t_0) = s(t_n)$.

We bepalen de coördinaten c_j van de interpolerende kubische spline $s(x)$ in zijn B-spline voorstelling

$$s(x) = \sum_{j=-3}^{n-1} c_j N_{j,3}(x)$$

door de interpolatievoorwaarden in te vullen:

$$\begin{aligned} s(x_i) &= \sum_{j=-3}^{n-1} c_j N_{j,3}(x_i) \\ &= \sum_{j=i-3}^{i-1} c_j N_{j,3}(x_i) \\ &= f_i. \end{aligned}$$

In wat volgt gebruiken we de notatie $N_j(x) = N_{j,3}(x)$. Nemen we de natuurlijke voorwaarden om de spline-functie enig te maken, dan krijgen we de volgende voorwaarden:

$$\begin{aligned} \sum_{j=-3}^{-1} c_j N_j^{(2)}(x_0) &= 0 \qquad \text{en} \\ \sum_{j=n-3}^{n-1} c_j N_j^{(2)}(x_n) &= 0. \end{aligned}$$

Dit leidt tot het volgende stelsel van $n+3$ lineaire vergelijkingen in de $n+3$ onbekenden $c_{-3}, c_{-2}, \ldots, c_{n-2}, c_{n-1}$:

$$\begin{bmatrix} N_{-3}^{(2)}(x_0) & N_{-2}^{(2)}(x_0) & N_{-1}^{(2)}(x_0) & & & & & & \\ N_{-3}(x_0) & N_{-2}(x_0) & N_{-1}(x_0) & & & & & & \\ & N_{-2}(x_1) & N_{-1}(x_1) & N_0(x_1) & & & & & \\ & & \ddots & \ddots & \ddots & & & & \\ & & & N_{n-4}(x_{n-1}) & N_{n-3}(x_{n-1}) & N_{n-2}(x_{n-1}) & & & \\ & & & & N_{n-3}(x_n) & N_{n-2}(x_n) & N_{n-1}(x_n) & & \\ & & & & N_{n-3}^{(2)}(x_n) & N_{n-2}^{(2)}(x_n) & N_{n-1}^{(2)}(x_n) & & \end{bmatrix} \begin{bmatrix} c_{-3} \\ c_{-2} \\ c_{-1} \\ \vdots \\ c_{n-2} \\ c_{n-1} \end{bmatrix}$$

$$= \begin{bmatrix} 0 \\ f_0 \\ f_1 \\ \vdots \\ f_{n-1} \\ f_n \\ 0 \end{bmatrix}.$$

Het derde element uit de eerste rij kan gemakkelijk geëlimineerd worden m.b.v. de tweede rij. Analoog kan het op twee na laatste element op de laatste rij gemakkelijk geëlimineerd worden m.b.v. de voorlaatste rij. Het resulterende stelsel is een tridiagonaal-stelsel dat met $O(n)$ bewerkingen kan opgelost worden.

8 Oefeningen

- In dit hoofdstuk werden verschillende resultaten niet expliciet afgeleid. Probeer zelf enkele van deze resultaten te bewijzen, bijv. de recursiebetrekking voor B-splines.
- Bewijs dat de afgeleiden van een spline $s(x)$ van graad k met knooppunten $t_{-k} < t_{-k+1} < \cdots < t_{n-1} < t_n$ kunnen berekend worden m.b.v.

de volgende formule

$$s^{(\nu)}(x) = \prod_{i=1}^{\nu}(k+1-i) \sum_{j=-(k-\nu)}^{n-1} c_i^{(\nu)} N_{i,k-\nu}(x)$$

waarbij de $c_j^{(i)}$ voldoen aan de volgende recursiebetrekkingen

$$c_j^{(i)} = \begin{cases} c_j & \text{als } i = 0 \\ \dfrac{c_j^{(i-1)} - c_{j-1}^{(i-1)}}{t_{j+k+1-i} - t_j} & \text{als } i > 0 \ . \end{cases}$$

1 QR Factorisatie

1.1 Householder Transformatie

Gegeven een matrix A :

$$A = \begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{pmatrix}$$

Om nullen te maken in de eerste kolom vanaf de tweede rij, selecteer de kolomvector waar de nullen moeten komen. Als er nullen moeten komen vanaf de i+1'de rij, behoudt dan de elementen uit de kolomvector vanaf het i-de element (alle elementen die nul moeten worden en het element erboven). Noem deze vector x :

$$x = \begin{pmatrix} a_{11} \\ a_{21} \\ a_{31} \end{pmatrix}$$

Bereken v (is later nodig), $sgn(x_1)$ is hier de sign-functie van het eerste element uit x, in ons geval is die a_{11}

$$\mathbf{v} = \mathbf{x} + sgn(x_i)||x||.e_i$$

Hiermee berekenen we P_1

$$P_1 = I - \frac{2vv^T}{v^T v}$$

Deze P_1 heeft als eigenschap dat $P_1 x = (sgn(x_1)||x||, 0, ..., 0)^T$

Dus in ons voorbeeld wordt $\mathbf{P}x = (sgn(a_{11})||x||, 0, ..., 0)^T$

P_1A levert een nieuwe matrix op waarbij de gewenste elementen nul zijn :

$$P_1 A = \begin{pmatrix} sgn(a_{11})||x|| & * & * \\ 0 & b_{22} & b_{23} \\ 0 & b_{32} & b_{33} \end{pmatrix}$$

Hierbij zijn de * nieuwe waarden, maar onbelangrijk, en de b_{ij} nieuwe waarden die nog gebruikt worden.

Om in de volgende kolom nullen te creëren doen we hetzelfde, maar x is hier kleiner :

$$x = \begin{pmatrix} b_{22} \\ b_{32} \end{pmatrix}$$

We bereken v exact zoals hiervoor, en P_2' zoals P_1.
Stel we bekomen P_2' :

$$P_2' = \begin{pmatrix} p_{22} & p_{23} \\ p_{32} & p_{33} \end{pmatrix}$$

Om nu de benodigde P_2 te bekomen moet P_2' de juiste dimensie krijgen, die gebeurt als volgt :

$$P_2 = \begin{pmatrix} 1 & 0 & 0 \\ 0 & p_{22} & p_{23} \\ 0 & p_{32} & p_{33} \end{pmatrix}$$

Vermenigvuldigen we nu P_1A met P_2 dan bekomen we de gewenste bovendriehoeksmatrix :

$$P_2P_1A = \begin{pmatrix} sgn(a_{11})||x_1|| & * & * \\ 0 & sgn(b_{22})||x_2|| & * \\ 0 & 0 & c_{33} \end{pmatrix}$$

Stellen we nu :

$$Q^T = P_2P_1$$

Dan (De P's zijn symmetrisch):

$$Q = P_1P_2$$

Dan krijgen we de QR factorisatie van A als volgt :

$$R = P_2P_1A = Q^TA$$
$$QR = A$$

Deze procedure is volledig uitbreidbaar voor grotere (of kleinere) dimensies van A.

Oefeningen Numerieke Wiskunde

Foutenanalyse: twee verschillende aanpakken

Bij het doen van een formele foutenanalyse kan je twee methodes volgen om een eerste orde benadering te bekomen van de fout. Bij de eerste methode ga je waar nodig de stelling van Taylor in 1 veranderlijke toepassen om zo in een aantal stappen een eerste orde benadering te bekomen. Bij de tweede methode bereken je de eerste orde benadering in één keer via een Taylor benadering in meer veranderlijken.

Beide methoden geven hetzelfde resultaat. Welke methode het makkelijkst is, hangt af van probleem tot probleem en is ook afhankelijk van je persoonlijke voorkeur.

1 Voorbeeld

Als voorbeeld voeren we een foutenanalyse uit voor de berekening van

$$y = \sqrt{1+x} - 1,$$

waarbij men de berekeningen uitvoert volgens de uitdrukking hierboven. Men veronderstelt dat x exact voorgesteld kan worden op de machine.

De berekende waarde voor y is dan

$$\bar{y} = fl\left(fl(\sqrt{fl(1+x)}) - 1\right) = \left((\sqrt{(1+x)(1+\epsilon_1)})(1+\epsilon_2) - 1\right)(1+\epsilon_3) \tag{1}$$

Hierbij stellen $\epsilon_i,\ i = 1,2,3$, de relatieve fouten voor die gemaakt worden bij afronding na iedere bewerking. We weten dat

$$|\epsilon_i| \leq \epsilon_{mach},\ i = 1,2,3.$$

2 Methode I

We herschrijven $\bar{y}$ als $\bar{y} \approx y(1+\delta)$ door in (1) gebruik te maken van de Taylor reeks rond nul

$$f(\epsilon) = f(0) + f'(0)\epsilon + f''(0)\frac{\epsilon^2}{2} + \ldots$$

en hogere orde termen te verwaarlozen. Bv.

$$\sqrt{1+\epsilon} \approx 1 + \frac{1}{2}\epsilon,$$

waarbij we veronderstellen dat ϵ klein genoeg is.

We krijgen dus

$$\begin{aligned}
\bar{y} &\approx \left((\sqrt{1+x})(1+\frac{1}{2}\epsilon_1+\epsilon_2) - 1\right)(1+\epsilon_3) \\
&= \left(y + (\sqrt{1+x})(\frac{1}{2}\epsilon_1+\epsilon_2)\right)(1+\epsilon_3) \\
&\approx y\left(1 + \frac{\sqrt{1+x}}{\sqrt{1+x}-1}(\frac{1}{2}\epsilon_1+\epsilon_2) + \epsilon_3\right),
\end{aligned}$$

waarbij we steeds de tweede orde termen verwaarloosd hebben.

3 Methode II

We interpreteren, in de uitdrukking voor $\bar{y}$, x als een parameter en $\epsilon_i,\ i = 1, 2, 3$, als variabelen,

$$\bar{y} = F(\epsilon_1, \epsilon_2, \epsilon_3).$$

Vermits de ϵ_i zeer klein zijn, laat $F(\epsilon_1, \epsilon_2, \epsilon_3)$ zich goed benaderen door de Taylorontwikkeling rond $(0, 0, 0)$ afgebroken na de lineaire termen

$$\bar{y} = F(\epsilon_1, \epsilon_2, \epsilon_3) \approx F(0,0,0) + \epsilon_1 \frac{\partial F}{\partial \epsilon_1}(0,0,0) + \epsilon_2 \frac{\partial F}{\partial \epsilon_2}(0,0,0) + \epsilon_3 \frac{\partial F}{\partial \epsilon_3}(0,0,0).$$

De constante term $F(0, 0, 0)$ in de lineaire benadering is de exacte waarde y. Voor de berekening van de partiële afgeleiden gaan we als volgt te werk

$$\begin{aligned} F(\epsilon_1, 0, 0) &= \sqrt{(1+x)(1+\epsilon_1)} - 1 = \sqrt{1+x}(1+\epsilon_1)^{\frac{1}{2}} - 1 \\ \frac{\partial F}{\partial \epsilon_1}(\epsilon_1, 0, 0) &= \sqrt{1+x}\frac{1}{2}(1+\epsilon_1)^{-\frac{1}{2}} \\ \frac{\partial F}{\partial \epsilon_1}(0, 0, 0) &= \frac{1}{2}\sqrt{1+x} \end{aligned}$$

Na uitwerking van de andere partiële afgeleiden vinden we de lineaire benadering voor $\bar{y}$ en een benaderende formule voor de absolute en de relatieve fout

$$\begin{aligned} \bar{y} &\approx y + \frac{1}{2}\sqrt{1+x}\,\epsilon_1 + \sqrt{1+x}\,\epsilon_2 + y\epsilon_3 \\ \bar{y} - y &\approx \frac{1}{2}\sqrt{1+x}\,\epsilon_1 + \sqrt{1+x}\,\epsilon_2 + y\epsilon_3 \\ \frac{\bar{y}-y}{y} &\approx \frac{\sqrt{1+x}}{2(\sqrt{1+x}-1)}\epsilon_1 + \frac{\sqrt{1+x}}{\sqrt{1+x}-1}\epsilon_2 + \epsilon_3. \end{aligned}$$

We krijgen uiteraard hetzelfde resultaat als met Methode I.

4 Oefeningen

Probeer beide methodes eens uit voor de volgende berekeningen:

1. $$y = \sqrt{e^x - 1}$$

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

3. $$y = (1+x^2)^{x^2}$$

DERIVATIVES AND INTEGRALS

Basic Differentiation Rules

1. $\frac{d}{dx}[cu] = cu'$
2. $\frac{d}{dx}[u \pm v] = u' \pm v'$
3. $\frac{d}{dx}[uv] = uv' + vu'$
4. $\frac{d}{dx}\left[\frac{u}{v}\right] = \frac{vu' - uv'}{v^2}$
5. $\frac{d}{dx}[c] = 0$
6. $\frac{d}{dx}[u^n] = nu^{n-1}u'$
7. $\frac{d}{dx}[x] = 1$
8. $\frac{d}{dx}[|u|] = \frac{u}{|u|}(u'), \quad u \neq 0$
9. $\frac{d}{dx}[\ln u] = \frac{u'}{u}$
10. $\frac{d}{dx}[e^u] = e^u u'$
11. $\frac{d}{dx}[\log_a u] = \frac{u'}{(\ln a)u}$
12. $\frac{d}{dx}[a^u] = (\ln a)a^u u'$
13. $\frac{d}{dx}[\sin u] = (\cos u)u'$
14. $\frac{d}{dx}[\cos u] = -(\sin u)u'$
15. $\frac{d}{dx}[\tan u] = (\sec^2 u)u'$
16. $\frac{d}{dx}[\cot u] = -(\csc^2 u)u'$
17. $\frac{d}{dx}[\sec u] = (\sec u \tan u)u'$
18. $\frac{d}{dx}[\csc u] = -(\csc u \cot u)u'$
19. $\frac{d}{dx}[\arcsin u] = \frac{u'}{\sqrt{1 - u^2}}$
20. $\frac{d}{dx}[\arccos u] = \frac{-u'}{\sqrt{1 - u^2}}$
21. $\frac{d}{dx}[\arctan u] = \frac{u'}{1 + u^2}$
22. $\frac{d}{dx}[\operatorname{arccot} u] = \frac{-u'}{1 + u^2}$
23. $\frac{d}{dx}[\operatorname{arcsec} u] = \frac{u'}{|u|\sqrt{u^2 - 1}}$
24. $\frac{d}{dx}[\operatorname{arccsc} u] = \frac{-u'}{|u|\sqrt{u^2 - 1}}$
25. $\frac{d}{dx}[\sinh u] = (\cosh u)u'$
26. $\frac{d}{dx}[\cosh u] = (\sinh u)u'$
27. $\frac{d}{dx}[\tanh u] = (\operatorname{sech}^2 u)u'$
28. $\frac{d}{dx}[\coth u] = -(\operatorname{csch}^2 u)u'$
29. $\frac{d}{dx}[\operatorname{sech} u] = -(\operatorname{sech} u \tanh u)u'$
30. $\frac{d}{dx}[\operatorname{csch} u] = -(\operatorname{csch} u \coth u)u'$
31. $\frac{d}{dx}[\sinh^{-1} u] = \frac{u'}{\sqrt{u^2 + 1}}$
32. $\frac{d}{dx}[\cosh^{-1} u] = \frac{u'}{\sqrt{u^2 - 1}}$
33. $\frac{d}{dx}[\tanh^{-1} u] = \frac{u'}{1 - u^2}$
34. $\frac{d}{dx}[\coth^{-1} u] = \frac{u'}{1 - u^2}$
35. $\frac{d}{dx}[\operatorname{sech}^{-1} u] = \frac{-u'}{u\sqrt{1 - u^2}}$
36. $\frac{d}{dx}[\operatorname{csch}^{-1} u] = \frac{-u'}{|u|\sqrt{1 + u^2}}$

Basic Integration Formulas

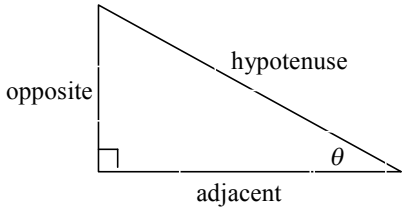
1. $\int kf(u)\,du = k\int f(u)\,du$
2. $\int [f(u) \pm g(u)]\,du = \int f(u)\,du \pm \int g(u)\,du$
3. $\int du = u + C$
4. $\int a^u\,du = \left(\frac{1}{\ln a}\right)a^u + C$
5. $\int e^u\,du = e^u + C$
6. $\int \sin u\,du = -\cos u + C$
7. $\int \cos u\,du = \sin u + C$
8. $\int \tan u\,du = -\ln|\cos u| + C$
9. $\int \cot u\,du = \ln|\sin u| + C$
10. $\int \sec u\,du = \ln|\sec u + \tan u| + C$
11. $\int \csc u\,du = -\ln|\csc u + \cot u| + C$
12. $\int \sec^2 u\,du = \tan u + C$
13. $\int \csc^2 u\,du = -\cot u + C$
14. $\int \sec u \tan u\,du = \sec u + C$
15. $\int \csc u \cot u\,du = -\csc u + C$
16. $\int \frac{du}{\sqrt{a^2 - u^2}} = \arcsin\frac{u}{a} + C$
17. $\int \frac{du}{a^2 + u^2} = \frac{1}{a}\arctan\frac{u}{a} + C$
18. $\int \frac{du}{u\sqrt{u^2 - a^2}} = \frac{1}{a}\operatorname{arcsec}\frac{|u|}{a} + C$

Trig Cheat Sheet

Definition of the Trig Functions

Right triangle definition

For this definition we assume that $0 < \theta < \frac{\pi}{2}$ or $0^\circ < \theta < 90^\circ$.



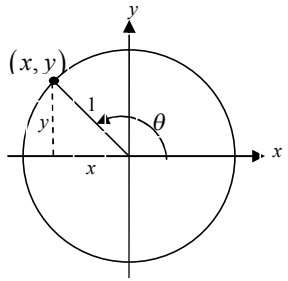
$$\sin\theta = \frac{\text{opposite}}{\text{hypotenuse}} \qquad \csc\theta = \frac{\text{hypotenuse}}{\text{opposite}}$$

$$\cos\theta = \frac{\text{adjacent}}{\text{hypotenuse}} \qquad \sec\theta = \frac{\text{hypotenuse}}{\text{adjacent}}$$

$$\tan\theta = \frac{\text{opposite}}{\text{adjacent}} \qquad \cot\theta = \frac{\text{adjacent}}{\text{opposite}}$$

Unit circle definition

For this definition θ is any angle.



$$\sin\theta = \frac{y}{1} = y \qquad \csc\theta = \frac{1}{y}$$

$$\cos\theta = \frac{x}{1} = x \qquad \sec\theta = \frac{1}{x}$$

$$\tan\theta = \frac{y}{x} \qquad \cot\theta = \frac{x}{y}$$

Facts and Properties

Domain

The domain is all the values of θ that can be plugged into the function.

$\sin\theta$, θ can be any angle

$\cos\theta$, θ can be any angle

$\tan\theta$, $\theta \neq \left(n+\frac{1}{2}\right)\pi$, $n = 0, \pm1, \pm2, \ldots$

$\csc\theta$, $\theta \neq n\pi$, $n = 0, \pm1, \pm2, \ldots$

$\sec\theta$, $\theta \neq \left(n+\frac{1}{2}\right)\pi$, $n = 0, \pm1, \pm2, \ldots$

$\cot\theta$, $\theta \neq n\pi$, $n = 0, \pm1, \pm2, \ldots$

Range

The range is all possible values to get out of the function.

$-1 \le \sin\theta \le 1$ $\qquad \csc\theta \ge 1$ and $\csc\theta \le -1$

$-1 \le \cos\theta \le 1$ $\qquad \sec\theta \ge 1$ and $\sec\theta \le -1$

$-\infty < \tan\theta < \infty$ $\qquad -\infty < \cot\theta < \infty$

Period

The period of a function is the number, T, such that $f(\theta+T) = f(\theta)$. So, if ω is a fixed number and θ is any angle we have the following periods.

$$\sin(\omega\theta) \rightarrow T = \frac{2\pi}{\omega}$$

$$\cos(\omega\theta) \rightarrow T = \frac{2\pi}{\omega}$$

$$\tan(\omega\theta) \rightarrow T = \frac{\pi}{\omega}$$

$$\csc(\omega\theta) \rightarrow T = \frac{2\pi}{\omega}$$

$$\sec(\omega\theta) \rightarrow T = \frac{2\pi}{\omega}$$

$$\cot(\omega\theta) \rightarrow T = \frac{\pi}{\omega}$$

Formulas and Identities

Tangent and Cotangent Identities

$$\tan\theta = \frac{\sin\theta}{\cos\theta} \qquad \cot\theta = \frac{\cos\theta}{\sin\theta}$$

Reciprocal Identities

$$\csc\theta = \frac{1}{\sin\theta} \qquad \sin\theta = \frac{1}{\csc\theta}$$

$$\sec\theta = \frac{1}{\cos\theta} \qquad \cos\theta = \frac{1}{\sec\theta}$$

$$\cot\theta = \frac{1}{\tan\theta} \qquad \tan\theta = \frac{1}{\cot\theta}$$

Pythagorean Identities

$$\sin^2\theta + \cos^2\theta = 1$$

$$\tan^2\theta + 1 = \sec^2\theta$$

$$1 + \cot^2\theta = \csc^2\theta$$

Even/Odd Formulas

$$\sin(-\theta) = -\sin\theta \qquad \csc(-\theta) = -\csc\theta$$

$$\cos(-\theta) = \cos\theta \qquad \sec(-\theta) = \sec\theta$$

$$\tan(-\theta) = -\tan\theta \qquad \cot(-\theta) = -\cot\theta$$

Periodic Formulas

If n is an integer.

$$\sin(\theta+2\pi n) = \sin\theta \qquad \csc(\theta+2\pi n) = \csc\theta$$

$$\cos(\theta+2\pi n) = \cos\theta \qquad \sec(\theta+2\pi n) = \sec\theta$$

$$\tan(\theta+\pi n) = \tan\theta \qquad \cot(\theta+\pi n) = \cot\theta$$

Double Angle Formulas

$$\sin(2\theta) = 2\sin\theta\cos\theta$$

$$\cos(2\theta) = \cos^2\theta - \sin^2\theta = 2\cos^2\theta - 1 = 1 - 2\sin^2\theta$$

$$\tan(2\theta) = \frac{2\tan\theta}{1-\tan^2\theta}$$

Degrees to Radians Formulas

If x is an angle in degrees and t is an angle in radians then

$$\frac{\pi}{180} = \frac{t}{x} \quad \Rightarrow \quad t = \frac{\pi x}{180} \quad \text{and} \quad x = \frac{180t}{\pi}$$

Half Angle Formulas

$$\sin^2\theta = \frac{1}{2}\left(1-\cos(2\theta)\right)$$

$$\cos^2\theta = \frac{1}{2}\left(1+\cos(2\theta)\right)$$

$$\tan^2\theta = \frac{1-\cos(2\theta)}{1+\cos(2\theta)}$$

Sum and Difference Formulas

$$\sin(\alpha \pm \beta) = \sin\alpha\cos\beta \pm \cos\alpha\sin\beta$$

$$\cos(\alpha \pm \beta) = \cos\alpha\cos\beta \mp \sin\alpha\sin\beta$$

$$\tan(\alpha \pm \beta) = \frac{\tan\alpha \pm \tan\beta}{1 \mp \tan\alpha\tan\beta}$$

Product to Sum Formulas

$$\sin\alpha\sin\beta = \frac{1}{2}\left[\cos(\alpha-\beta) - \cos(\alpha+\beta)\right]$$

$$\cos\alpha\cos\beta = \frac{1}{2}\left[\cos(\alpha-\beta) + \cos(\alpha+\beta)\right]$$

$$\sin\alpha\cos\beta = \frac{1}{2}\left[\sin(\alpha+\beta) + \sin(\alpha-\beta)\right]$$

$$\cos\alpha\sin\beta = \frac{1}{2}\left[\sin(\alpha+\beta) - \sin(\alpha-\beta)\right]$$

Sum to Product Formulas

$$\sin\alpha + \sin\beta = 2\sin\left(\frac{\alpha+\beta}{2}\right)\cos\left(\frac{\alpha-\beta}{2}\right)$$

$$\sin\alpha - \sin\beta = 2\cos\left(\frac{\alpha+\beta}{2}\right)\sin\left(\frac{\alpha-\beta}{2}\right)$$

$$\cos\alpha + \cos\beta = 2\cos\left(\frac{\alpha+\beta}{2}\right)\cos\left(\frac{\alpha-\beta}{2}\right)$$

$$\cos\alpha - \cos\beta = -2\sin\left(\frac{\alpha+\beta}{2}\right)\sin\left(\frac{\alpha-\beta}{2}\right)$$

Cofunction Formulas

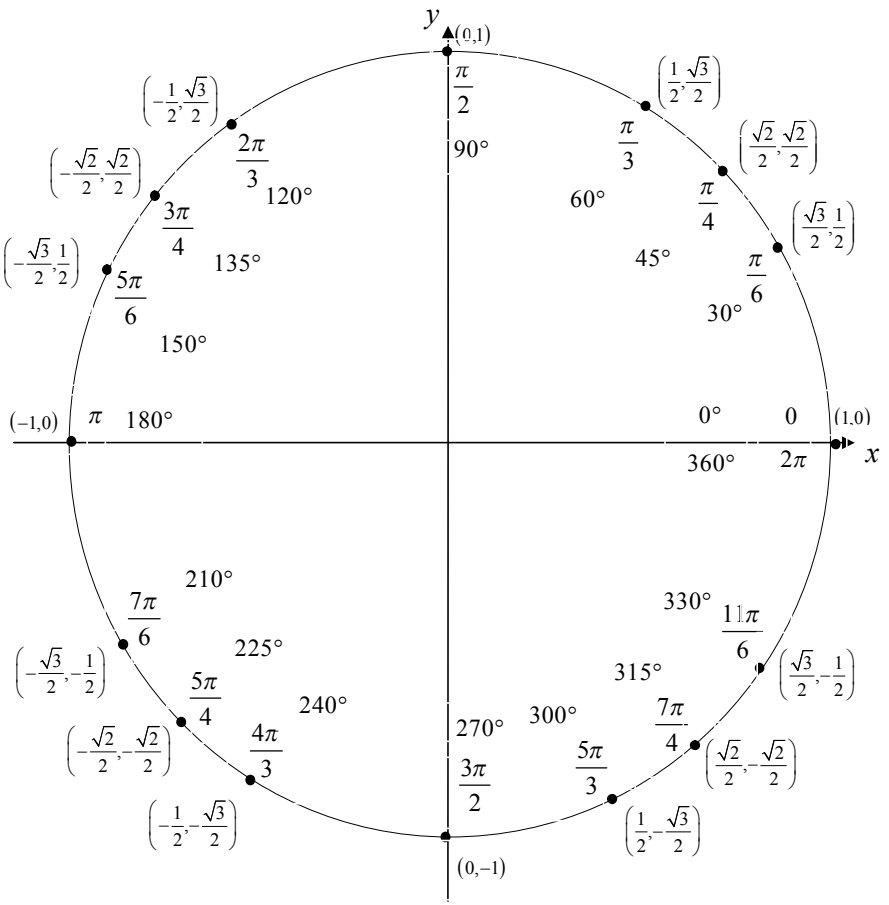
$$\sin\left(\frac{\pi}{2}-\theta\right) = \cos\theta \qquad \cos\left(\frac{\pi}{2}-\theta\right) = \sin\theta$$

$$\csc\left(\frac{\pi}{2}-\theta\right) = \sec\theta \qquad \sec\left(\frac{\pi}{2}-\theta\right) = \csc\theta$$

$$\tan\left(\frac{\pi}{2}-\theta\right) = \cot\theta \qquad \cot\left(\frac{\pi}{2}-\theta\right) = \tan\theta$$

© 2005 Paul Dawkins

Unit Circle



For any ordered pair on the unit circle (x, y) : $\cos\theta = x$ and $\sin\theta = y$

Example

$$\cos\left(\frac{5\pi}{3}\right) = \frac{1}{2} \qquad \sin\left(\frac{5\pi}{3}\right) = -\frac{\sqrt{3}}{2}$$

Inverse Trig Functions

Definition

$y = \sin^{-1} x$ is equivalent to $x = \sin y$

$y = \cos^{-1} x$ is equivalent to $x = \cos y$

$y = \tan^{-1} x$ is equivalent to $x = \tan y$

Inverse Properties

$\cos\left(\cos^{-1}(x)\right) = x$ $\quad \cos^{-1}\left(\cos(\theta)\right) = \theta$

$\sin\left(\sin^{-1}(x)\right) = x$ $\quad \sin^{-1}\left(\sin(\theta)\right) = \theta$

$\tan\left(\tan^{-1}(x)\right) = x$ $\quad \tan^{-1}\left(\tan(\theta)\right) = \theta$

Domain and Range

Function	Domain	Range
$y = \sin^{-1} x$	$-1 \le x \le 1$	$-\frac{\pi}{2} \le y \le \frac{\pi}{2}$
$y = \cos^{-1} x$	$-1 \le x \le 1$	$0 \le y \le \pi$
$y = \tan^{-1} x$	$-\infty < x < \infty$	$-\frac{\pi}{2} < y < \frac{\pi}{2}$

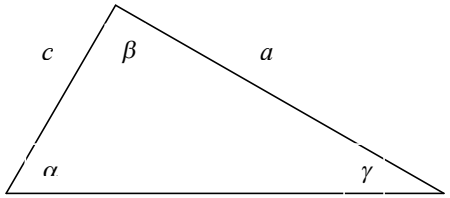
Alternate Notation

$\sin^{-1} x = \arcsin x$

$\cos^{-1} x = \arccos x$

$\tan^{-1} x = \arctan x$

Law of Sines, Cosines and Tangents



Law of Sines

$$\frac{\sin\alpha}{a} = \frac{\sin\beta}{b} = \frac{\sin\gamma}{c}$$

Law of Cosines

$$a^2 = b^2 + c^2 - 2bc\cos\alpha$$

$$b^2 = a^2 + c^2 - 2ac\cos\beta$$

$$c^2 = a^2 + b^2 - 2ab\cos\gamma$$

Mollweide's Formula

$$\frac{a+b}{c} = \frac{\cos\frac{1}{2}(\alpha-\beta)}{\sin\frac{1}{2}\gamma}$$

Law of Tangents

$$\frac{a-b}{a+b} = \frac{\tan\frac{1}{2}(\alpha-\beta)}{\tan\frac{1}{2}(\alpha+\beta)}$$

$$\frac{b-c}{b+c} = \frac{\tan\frac{1}{2}(\beta-\gamma)}{\tan\frac{1}{2}(\beta+\gamma)}$$

$$\frac{a-c}{a+c} = \frac{\tan\frac{1}{2}(\alpha-\gamma)}{\tan\frac{1}{2}(\alpha+\gamma)}$$

© 2005 Paul Dawkins

Examenvragen Numerieke Wiskunde 2012

Dennis Frett, Karel Domin, Jonas Devlieghere

21 juni 2012

Inhoudsopgave

1 Programma verschil, verklaar afwijking

Gegeven: Programma:

```
1 som = 0.0
2 for i = 0.0:0.1:1.0
3 verschil = i - som % == 0
4 som = som + 0.1
5 end
```

Output:

```
1 verschil = 0
2 verschil = 0
3 ..
4 verschil = 1.1.. e-15
```

(Syntaxverduidelijking: % en alles wat erachter komt is commentaar, geen modulo ofzo.)

Gevraagd: Verklaar waarom het verschil plots niet meer gelijk is aan 0. Waarvoor staat dat getal?

Informatie: Boek pagina 22, PC-zitting over foutenanalyse

Antwoord: Het getal 0.1 kan niet exact worden voorgesteld. We werken op de computer met een getallenvoorstelling met basis = 2.

0.1 in het binair = 0.000110011... dit is niet eindig voor te stellen, er zal dus gebruik gemaakt worden van afkapping of afronding. Hierdoor zal bijvoorbeeld intern $10 * 0.1 \neq .1$ zijn. Het gevolg hiervan is dat er een kleine fout zal gemaakt worden bij het intern voorstellen (met basis 2 dus). Bij het outputten wordt er weer omgezet naar decimaal talstelsel en zal men in het begin toch nog de waarde 0 krijgen. Dit komt doordat de gemaakte fout kleiner is dan de machineprecisie.
We combineren dit met onze kennis over Matlab:

- De machineprecisie is de **grootste** relatieve fout die je kan maken wanneer je een getal voorstelt met de computer.

- eps is de afstand tussen 1 en het eerstvolgende machinegetal (= voorstelbaar getal)
- hieruit volgt: emach = eps/2
- eps(2) = de afstand tussen 2 en het eerstvolgende machinegetal = eps(1)*2 enz.
- Emach houdt rekening met afronding, eps niet.

Het getal dat we uitkomen is dus eps/2

2 Matrix met dominante eigenwaarde

Gegeven: Maple afdruk: laatste vraag van de examenvragen in de winabundel (Die over het bepalen van eigenwaarden met de methode van de machten).

Uit de matrix A = [2 1 -1; 0 3 -5; 0 0 -2] wordt de dominante eigenwaarde berekend. De startwaarden zijn: [-1.00001 1.00002 1]. Op de grafiek is zichtbaar dat er eerst naar 2 lijkt te convergeren, maar uiteindelijk toch de juiste eigenwaarde 3 gekozen wordt. Het berekenen gebeurt met de methode van de machten met normalisatie.

Gevraagd:

- Hoe komt het dat er eerst naar 2 geconvergeerd wordt?
- Waarom uiteindelijk toch naar 3?
- Wat als er geen normalisatie gebruikt zou worden?

Antwoord: We hebben gegeven:
$A = \begin{bmatrix} 2 & 1 & -1 \\ 0 & 3 & -5 \\ 0 & 0 & -2 \end{bmatrix}$ $X_0 = \begin{bmatrix} -1.00001 \\ 1.00002 \\ 1 \end{bmatrix}$
De matrix A is een bovendriehoeksmatrix, dit wil zeggen dat we de eigenwaarden van A vinden op de hoofddiagonaal: $\lambda_1 = 2$,$\lambda_2 = 3$, $\lambda_3 = -2$. Hieruit leiden we af dat de dominante eigenwaarde 3 is. De methode van de machten zal dus normaal eerst naar de dominante eigenwaarde convergeren. Indien we de eigenvectoren van A berekenen, horende bij de eigenwaarden, dan vinden we:

$$E1 = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} \; E2 = \begin{bmatrix} 1 \\ 1 \\ 0 \end{bmatrix} \; E3 = \begin{bmatrix} 0 \\ 1 \\ 1 \end{bmatrix}$$

(We berekenden deze eigenvectoren met de formule: $(A - \lambda I) = 0$) voorbeeld voor $\lambda_1 = 2$:

$$A = \begin{bmatrix} 0 & 1 & -1 \\ 0 & 1 & -5 \\ 0 & 0 & -4 \end{bmatrix} \begin{bmatrix} X_1 \\ X_2 \\ X_3 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$$

We lossen dit stelsel op:
$-4X_3 = 0$
$X_2 - 5X_3 = 0$
$X_2 - X_3 = 0$
We merken op dat X_1 een vrije variabele is, voor de gemakkelijkheid stellen we deze gelijk aan 1 achteraf. We krijgen dan:
$X_1 = 1$
$X_1 = X_3 = 0$
Wat ons de uitgekomen eigenvector E1 geeft. De werkwijze voor de overige 2 eigenvectoren is identiek.
We merken nu dat de startvector $X_0 = \begin{bmatrix} -1.00001 \\ 1.00002 \\ 1 \end{bmatrix}$ ongeveer een lineaire combinatie is van de 2 eigenvectoren E1 en E3. Hierdoor zit de iteratie van het algoritme in het begin vast in het vlak van deze 2 eigenvectoren. Door de lichte afwijking op de vector (de .00001) komt de vector na genoeg iteraties toch uit het vlak en gaan we naar 3 itereren. Moest de startvector exact een lineaire combinatie van 2 eigen vectoren zijn dan zou men nooit naar 3 convergeren. (zie ook oefenzitting 10, Matlab sessie, daar hebben we ongeveer hetzelfde gedaan). De grafiek van de norm van de gevonden vector is ook gegeven en daar zie je dat hij eerst een tijd op 2 staat en dan pas na 20 stappen begint te schommelen en toch naar 3 gaat. Waarom? Omdat hij pas de afwijking van de ideale waarden (de .00001) gaat zien nadat de iteratieve methode de juiste precisie heeft bereikt. Dat wil zeggen na 20 stappen (1 stap is 1 bit en 3 bits per getal nauwkeurig => 20 stappen voor 6 getallen nauwkeurig). We maken gebruik van een genormaliseerde vorm om overloop of onderloop te vermijden. Door normalisatie gaat de norm van een vector namelijk beperkt zijn en is er dus minder kans op overloop/onderloop. De methode zal enkel naar λ convergeren als λ dominant is en de startvector een component heeft overeenkomstig met de eigenvector van λ. In de praktijk hang de bruikbaarheid van de von Mises methode af van de verhouding $\frac{|\lambda_2|}{|\lambda_1|}$, de convergentiefactor. De methode kan falen om verschillende redenen:

- Startvector heeft geen component in de richting van de dominante eigenvector. (dit is als α=0) In de praktijk zal dit probleem niet vaak voorkomen omdat afrondingsfouten vaak toch een component in die richting garanderen.

- Er kunnen meerde eigenwaarden zijn met dezelfde (maximum) modules. De methode kan dan convergeren naar een lineaire combinatie van de overeenkomstige eigenvectoren.
- Voor een reële matrix en startvector, kan de methode nooit convergeren naar een complexe vector.

Zie ook vraag 4 van het opgeloste examen door van Barel zelf (ongeveer gelijke vraag).

3 Functiewaarden gegeven, bepaal factor

Gegeven: Er is een functie van de vorm p(x) $= a_0 + a_1x + a_2x^2 + ... + a_nx^n$ (n is niet gekend)

- p(0) = 5
- p(1) = 9
- p(2) = 15
- p(3) = 18

Alle gedeelde differenties van de vierde graad 1 zijn.

Gevraagd: Geef a_3.

Informatie: Boek paginas 103-105

Antwoord: Gegeven:
$f[x0, x1, x2, x3, x4] = 1$

$f[x_i] = p(i)$

$f[x_0, x_1] = (f(x_1) - f(x_0))/(x_1 - x_0) = (9 - 5)/(1 - 0) = 4$
$f[x_1, x_2] = 6$
$f[x_2, x_3] = 3$

$f[x_0, x_1, x_2] = (f[x_1, x_2] - f[x_0, x_1])/(x_2 - x_0) = (6 - 4)/2 = 1$
$f[x_1, x_2, x_3] = (f[x_2, x_3] - f[x_1, x_2])/(x_3 - x_1) = (3 - 6)/2 = -1.5$
$f[x0, x1, x2, x3] = (-1.5 - 1)/3 = -0.833333...$

$y_n(x) = f[x_0]+f[x_0, x_1](x-x_0)+f[x_0, x_1, x_2](x-x_0)(x-x_1)+f[x_0, x_1, ..., x_n](x-x_0)(x - x_1)...(x - x_{n-1})$

$y_n(x) = 5 + 4(x - 0) + 1(x - 0)(x - 1) + (-0.8333...)(x - 0)(x - 1)(x - 2) + 1(x - 0)(x - 1)(x - 2)(x - 3)$

Uitgewerkte vorm:

$y_n(x) = x^4 - 6.83333...x^3 + 14.5x^2 - 4.6666...x + 5$

Dus:
Coefficient van $x^3 = -6.83333$

4 Nulpunten met Jacobi

Gegeven: Een 2-dimensionaal lineair stelsel $Ax = b$ met

$$\begin{pmatrix} \alpha + 1 & 1 \\ \alpha & 1 \end{pmatrix}$$

We gebruiken de methode van Jacobi om een nulpunt te vinden.

Gevraagd: Bepaal alle waarden van alfa waarvoor de methode van Jacobi convergeert (voor alle startwaarden).

Informatie: Boek pagina 272

Antwoord: Methoden van Jacobi en Gauss-Seidel convergeren enkel indien de matrix A van het stelsel *diagonaal dominant is*.
Dus: een element op de diagonaal moet in absolute waarde groter zijn dan de som van de absolute waarden van alle andere elementen die zich op *dezelfde rij* bevinden. Dit is voldoende maar niet altijd *nodig* voor convergentie.

Dus:
$|\alpha + 1| > 1$ en
$1 > |\alpha|$

Dit geldt voor: $\alpha \in]0, 1[$
En voor: $\alpha \in]-2, -\infty[$ (Deze was vergeten in de wiki-oplossingen)

5 NR: Vijfdegraadsveelterm

Gegeven: Een hoop maple-uitvoer. Het gaat over een vijfdegraadsveelterm met een nulpunt in -0.31. Er wordt Newton-Raphson gebruikt om dat nulpunt te berekenen, en je krijgt een logaritmische plot van de fout. De plot is een heel normale, typische plot voor kwadratische convergentie.

Vraag: Verklaar deze grafiek (van de fout dus). Wat is de convergentiesnelheid? Als bijvraag kreeg ik het aantal juiste beduidende cijfers verdubbelt bij elke stap, hoe zie je dat in de grafiek ?

Informatie: Boek pagina 228

Antwoord: De convergentiesnelheid van Newton-Raphson is gekend:

- Kwadratisch als x^* enkelvoudig is
- Lineair als x^* een meervoudig nulpunt is

We weten bovendien dat als de functie F afleidbaar is dat geldt:

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

Waarmee vinden daarmee de **convergentiefactor** voor Newton-Raphson:

$$\rho = 1 - \frac{1}{m} = 1 - m^{-1}$$

Deze geeftaan hoeveel de benaderingsfout verkleint in de $k-$de benaderingsstap als $k \to \infty$. Hoe kleiner ρ, hoe sneller de functie convergeert.

De convergentiefactor is echter niet voldoende. We moeten ook de orde van convergentie (p) bepalen. Het proces is van orde n als $F^n(x^*) \neq 0$ en er geldt:

$$\rho_n = \frac{F^n(x^*)}{n!}$$

Enkelvoudig nulpunt: Dan geldt dat $m = 1$ en over het algemeen is $F''(x^*) \neq 0$ als $F(x) = \frac{x - f(x)}{f'(x)}$. De orde is dus bijgevolg **kwadratisch** indien het nulpunt enkelvoudig is.

Meervoudig nulpunt: Als $m > 1$ hebben we meervoudige nulpunten en is de methode bijgevolg slechts lineair.

6 Stabiliteit Methodes oplossen Matrices

Gegeven: A,b en twee berekende x matrices: Ax=b. De resultaten liggen ver uit elkaar.

Gevraagd: Bespreek stabiliteit van de methodes als machinenauwkeurigheid 10^{-15} is.

Antwoord: We bespreken de stabiliteit in het algemeen voor *Gauss-eliminatie* en *optimale pivotering*.

Gausselimiatie: Aan de hand van het voorbeeld in het boek kunnen we zien dat de volgorde van de vergelijkingen bij het gebruik van Gauss *zonder pivotering* de nauwkeurigheid sterk beïnvloedt.

Een probleem treedt op wanneer het pivotelement a_{11} zeer klein is. Bekijken we het voorbeeld in het boek dan zien we:

$$x_1 = \frac{1}{a_{11}}(b_1 - x_2)$$

Door te delen door dit getal bekomen we juist een zeer groot getal. In het voorbeeld zoeken we een waarde $x_1 \approx 1$. Dit wil zeggen dat de rechter factor zeer klein zal moeten zijn. Omdat de getallen b_1 en x_2 van dezelfde grote-orde zijn als x_1 zelf, vormt hier zich een zeer grote relatieve fout.

Indien de matrices gegeven zijn, kunnen we erop Gauss-eliminatie toepassen en die waarden gebruiken voor bovenstaande analyse.

Gausselimiatie met optimale pivotering: Hierbij gaan we de absolute waarde van de spilelementen zo groot mogelijk proberen maken om bovenstaande problematiek te voorkomen. We kunnen in dit geval de **residus** gebruiken als maat voor de stabiliteit. Deze is gedefinieerd als:

$$R = A\bar{X} - B = A(X + \Delta X) - B$$

Als ΔX klein is, is het residu. Het omgekeerde is **niet** altijd waar. Als het probleem slecht geconditioneerd is, kunnen de residuvectoren toch groot worden. Veronderstellen we een perturbatie R:

$$A(X + \Delta X) = B + R$$

Dan geldt uit de analyse van paragraaf 9.2 in het boek:

$$\frac{||\Delta X||}{||X||} \leq \kappa(A) \cdot \frac{||R||}{||B||}$$

Kleine (relatieve) residu's kunnen dus toch afkomstig zijn van grote (relatieve) fouten op X door een slechte conditie.

7 Veeltermen met zo laag mogelijke graad

Gegeven: Veelterm $p(x)$ met $p(1) = p(0) = p(1)$ en $p'(0) = 1$

Gevraagd: Geef alle veeltermen van zo laag mogelijke graad die hieraan voldoen.

Informatie: Zie p.122, methode der onbepaalde cofficinten

Antwoord: We gebruiken de methode der onbepaalde coëfficiënten:
Men kan de coëfficiënten van de interpolerende veeltermbepalen door expliciet de interpolatievoorwaarden op te leggen. We zoeken een veelterm van zo laag mogelijke graad die aan de bovenstaande voorwaarden voldoet. Er zijn 3 punten gegeven, P(-1) , P(0) en P(1) en één afgeleide. Dat zijn in totaal 4 interpolatievoorwaarden dus we zoeken een veelterm van graad 3 (deze heeft namelijk 4 te bepalen coëficienten). We zoeken dus via de methode der onbepaalde coëfficiënten voor n=3 interpolatiepunten, graad is dus 3.
We nemen een algemene veelterm van graad 3:

$a_0 + a_1x + a_2x^2 + a_3x^3$

We gaan nu een stelsel opstellen:
We beginnen met P(-1) = c (met c een waarde die we niet kennen). We krijgen onze eerste vergelijking:

$a_0 - a_1 + a_2 - a_3 = c$

Voor P(0) weten we dat P(0) = P(-1) = c. We krijgen:

$a_0 = c$

Voor P(1) weten we ook dat P(1) = c. We krijgen de 3de vergelijking:

$a_0 + a_1 + a_2 - a_3 = c$

Als laatste weten we nog dan P'(0) = 1. De afgeleide van onze algemene functie $= a_1 + 2a_2x + 3a_3x^2$
Hier vullen we 0 in, dan moet dit gelijk zijn aan 1:

$a_1 = 1$

Hiermee kunnen we volgend stelsel opstellen:

$$\sigma(s,i) = \begin{cases} a_0 - a_1 + a_2 - a_3 = c \\ \qquad\qquad\quad a_0 = c \\ a_0 + a_1 + a_2 - a_3 = c \\ \qquad\qquad\quad a_1 = 1 \end{cases}$$

Hieruit halen we dat $a_1 = 1$
Wanneer we dit stelsel verder uitwerken krijgen we volgende waarden: $a_1 = 1$, $a_3 = 1$, $a_2 = 0$ en P(0) = c
Dit levert:
$c + x - x^3$

8 Methode van het Midden

Gegeven: Grafiek

Gevraagd: Bespreek (conditie etc).

Informatie Boek pagina 238 (Conditie van een wortel)

Antwoord: Als x^* een wortel is van f(x), dus f(x^*) = 0 en we brengen een kleine wijziging aan op de gegevens, hoe verandert dan de wortel x^*?
Als $|f'(x^*)|$ klein is of 0 is het probleem slecht geconditioneerd, omgekeerd, als $|f'(x^*)|$ groot is, is het probleem goed geconditioneerd.
Check fig 2.11 op p239 om te zien waarom dit zo is.

9 NR: Convergentiefactor en orde

Gegeven: Verschillende grafieken van NR en vereenvoudigde NR

Gevraagd: Bespreek en geef convergentiefactor en orde.

Informatie: Boek pagina 261

Antwoord:

10 Hermitisch interpolerende veelterm

Gegeven: f_0, f'_0, f_1, f'_1

Gevraagd: Bepaal de Hermitisch interpolerende veelterm van graad 3

Informatie: Boek pagina 122 (letterlijk)

Antwoord: Er zijn twee methodes om dit probleem aan te pakken: de *methode der onbepaalde coefficienenten* en de *methode met confluente interpolatiepunten*. Mij leek de eerste methode eenvoudiger.

We kunnen de veelterm bepalen door expliciet interpolatievoorwaarden op te leggen. Er moet in dit geval voldaan worden aan:

$$\sigma(s,i) = \begin{cases} a_0 + a_1x_0 + a_2x_0^2 + a_3x_0^3 = f_0 \\ a_0 + a_1x_1 + a_2x_1^2 + a_3x_1^3 = f_1 \\ a_1 + 2a_2x_0 + 3a_3x_0^2 = f'_0 \\ a_1 + 2a_2x_1 + 3a_3x_1^2 = f'_1 \end{cases}$$

De oplossing van dit stelsel geeft de Hermite-interpolerende veelterm.

$$a_0 + a_1x + a_2x^2 + a_3x^3$$

De oplossing is uiteraard onderhevig aan de conditie van dit probleem en de stabiliteit van de gekozen methode voor het bepalen van de oplossing van het stelsel.

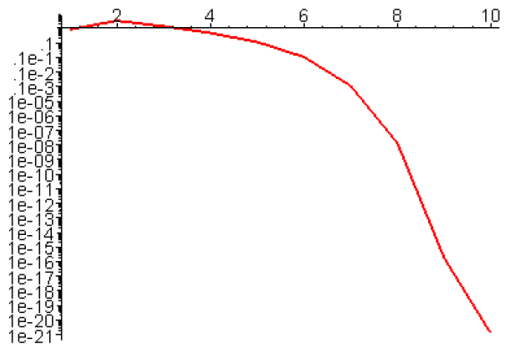
11 Bespreken grafiek niet-lineair stelsel

Gegeven: Grafieken van niet-lineair stelsel

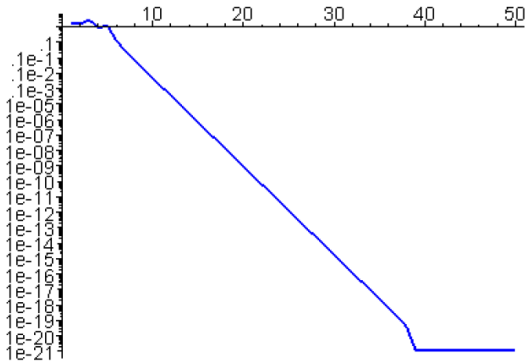
Gevraagd: Bespreek de grafieken

Informatie: Boek pagina 261

Antwoord: (Ik vermoed dat dit vindbaar is in de slides?)
Newton-Raphson:



Vereenvoudigde Newton-Raphson:



We zien dat dat vereenvoudigde lineair convergeert, hoewel deze daar meer stappen voor nodig heeft. Dit is omdat de niet-vereenvoudigde veel meer berekeningen doet per stap.

12 Voorstelling 0.3

Gegeven: Een Mapleprogramma voert onderstaande instructies uit

```
1 x = 0.1
2 y = 3*0.1
3 if y == 0.3
4 then print "y is gelijk aan 0.3"
5 else print "y is niet gelijk aan 0.3".
```

Output:

```
1 x = 1.0000000000e-1
2 y = 3.0000000000e-1
3 y is niet gelijk aan 0.3
```

Gevraagd: Leg in detail uit waarom het programma besluit dat y niet gelijk is aan 0.3

Antwoord: Getal 0.1 wordt binair door een repeterend patroon (ofzo) voorgesteld en dus als we inlezen wordt er een deel van dat patroon afgebroken en dus een kleine fout gemaakt. een geheel getal zoals 3 kan wel exact worden voorgesteld, en dus wordt de formule met fouten:

$$fl(x) = 0.1(1 + e)$$

$$y = 3 * x(1 + e')(1 + e)$$

$$fl(y) = y(1 + e)$$

en dus wordt er drie keer een foutje gemaakt en zal y dus niet meer exact gelijk aan 0.3 zijn (dat trouwens ook niet exact kan worden voorgesteld maar ook, zoals x, wordt afgebroken)

Bij het uitschrijven van y, wordt de binaire info terug omgezet naar decimaal talstelsel en vermits de fout die wordt gemaakt kleiner is als de machineprecisie zie je die eerst niet.

13 Stabiliteit functie

Gegeven: $f(x) : e^{x^2} - 1 - x^2$. Voor het berekenen van de waarde gebruiken we volgend algoritme:

```
1 a = x^2
2 b = e^a
3 f = b-1-a
```

Gevraagd: Is deze numeriek stabiel? Bereken $f(10^{-4})$ met 10 beduidende juiste cijfers.

Antwoord: Om de fout uit de macht te halen kunnen we twee benaderingen hanteren:

- Tailor met het verwaarlozen van hogere orde termen
- Partieel afleiden naar ϵ_i

Ik heb het niet uitgewerkt.
Maar:
(met dank aan Gust Verbruggen hebben we de uitwerking)

We willen volgende expressie berekenen

$$y = e^{x^2} - 1 - x$$

Hier zal de benadering, na het uitvoeren van de verschillende stappen, gegeven worden door

$$\bar{y} = (e^{x^2(1+\epsilon_1)}(1+\epsilon_2) - 1 - x^2(1+\epsilon_1))(1+\epsilon_3)$$

Als we x als een parameter zien, kunnen we de expressie zien in functie van de epsilons

$$\bar{y} \approx F(\epsilon_1, \epsilon_2, \epsilon_3)$$

, dewelke we rond (0,0,0) kunnen benaderen dmv. Taylorontwikkeling (omdat de ϵ_i zeer klein zullen zijn). We berekenen $\frac{\partial F}{\partial \epsilon_1}(\epsilon_1, 0, 0)$, $\frac{\partial F}{\partial \epsilon_2}(0, \epsilon_2, 0)$ en $\frac{\partial F}{\partial \epsilon_3}(0, 0, \epsilon_3)$ (dit mag zo gebeuren omdat we ze uiteindelijk toch zullen evalueren in (0,0,0); de vermenigvuldiging met $(1 + \epsilon_i)$ mag verwaarloosd worden

voor de constante factoren en zo kunnen we de af te leiden functie op voorhand vereenvoudigen).

$$\frac{\partial F}{\partial \epsilon_1}(\epsilon_1, 0, 0) = x^2(e^{x^2(1+\epsilon_1)} - 1)$$

$$\frac{\partial F}{\partial \epsilon_2}(0, \epsilon_2, 0) = e^{x^2}$$

$$\frac{\partial F}{\partial \epsilon_3}(0, 0, \epsilon_3) = e^{x^2} - 1 - x^2$$

Ontwikkeling rond (0,0,0) wordt dan

$$\bar{y} \approx y + \epsilon_1 x^2(e^{x^2} - 1) + \epsilon_2 e^{x^2} + \epsilon_3 y$$

en dit is de benaderde waarde. Hieruit kan je dan verder de absolute en relatieve fout berekenen.

14 Interpolatie sinus

Gegeven: De functie $f(x) = sin(x)$ in het interval $(-\pi, \pi)$.

Gevraagd: Geef een bovengrens op de interpolatie-fout als ge weet dat uw interpolerende veelterm p is die in $n-1$ interpolatiepunten interpoleert.

Informatie: p. 94 e.v.

Antwoord: De bovengrens wordt gegeven door de formule

$$E_n = max_{x\in[-\pi,\pi]}|E_n(x)| \leq max_{x\in[-\pi,\pi]}|\frac{|(x-x_0)(x-x_1)...(x-x_{n-2})|}{(n-2+1)!}\cdot max_{x\in[-\pi,\pi]}|f^{(n-2+1)})(x)|$$

Aangezien de afgeleide van de sinus ofwel een cosinus ofwel een sinus is is het maximum van $|f^{(n-2+1)})(x)| = 1$. Veronderstellen we bijvoorbeeld equidistante punten

$$x_i = -\pi + i \cdot (\frac{2\pi}{n})$$

geeft dit

$$E_n \leq \frac{|\prod_{0<i<n-2}|}{(n-1)!}$$

15 Convergentiegetal en -orde

Gegeven:

$$\begin{pmatrix} a & b \\ c & d \end{pmatrix}$$

met als waarden

- $a = 10^{-}5$
- $b = 10^{-}5$
- $c = 3 - a$
- $d = ab - 2/b$

$$\text{startwaarde} = \begin{bmatrix} 1 \\ 1 \end{bmatrix} ???$$

Gevraagd: Zoek convergentiegetal en orde en waarom is er zo'n grote fout?

Informatie: Boek pagina 290

Antwoord: Het convergentiegetal is $\frac{\lambda_2}{\lambda_1}$.
De convergentieorde van de methode van de machten is lineair (p290 puntje 3).
Het conditiegetal van de matrix is van de grootte-orde 10^{10}, en is dus zeer slecht geconditioneerd.
Dit zou kunnen verklaren dat er zo'n grote fout op de oplossing zit omdat diezelfde matrix telkens opnieuw vermenigvuldigd wordt.

16 Lagrange interpolatie

Gegeven: $-h < x < h$ en

$$f'(x) = \frac{1}{h^2}[\frac{2x-h}{2} \cdot f(-h) - 2x \cdot f(0) + \frac{2x+h}{2} \cdot f(h)] + D(x)$$

met $D(x)$ de differentiatiefout.

Gevraagd: Uitdrukking voor $D(x)$. Waar is die het grootst?

Informatie: Boek pagina 131

Antwoord: De differentiefout wordt gegeven door de formule:

$$D_n(x) = \frac{\pi'(x_i)}{(n+1)!} f^{(n+1)}(\xi(x_i))$$

De wordt maximaal als $\pi'(x_i)$ maximaal wordt. $\pi(x)$ wordt gegeven door:

$$\pi(x) = (x-x_0)(x-x_1)...(x-x_n)$$

$\pi'(x) = 1(x-x_1)...(x-x_n) + (x-x_0)[(x-x_2)...(x-x_)$
$+(x-x_1)[(x-x_3)...(x-x_n) + (x-x_2)[...]]$
$\pi'(x) = (x-x_1)...(x-x_n) + (x-x_0)(x-x_2)...(x-x_n)+$
$(x-x_0)(x-x_1)(x-x_3)...(x-x_n) + ...+$
$(x-x_0)(x-x_1)...(x-x_{n-2})(x-x_{n-1})$

Deze formule is maximaal indien de factoren maximaal zijn. Dit is het geval als de punten op gelijke afstand van elkaar gelegen zijn.

Dan geldt:

$$D_n(x_i) = (-1)^{n-i} \frac{i!(n-i)!}{(n+1)!} h^n f^{(n+1)}(\xi(x_i))$$

17 Equidistance punten met een fout epsilon

Gegeven: Enkele equidistante punten Xi, f(Xi). Er staat een fout epsilon op 1 van die punten, namelijk op Xk, f(Xk). Als je de tabel van voorwaartse (=gedeelde) differenties opstelt kan je zien hoe de fout zich propageert. Herken je een bepaald patroon? (duh, anders zou hij het niet vragen) Je mag er van uit gaan dat de differenties met hogere graad naar 0 gaan.

Gevraagd: Hoe kan je vinden op welk punt Xk de fout zat en hoe groot die is?

Antwoord: Stel gewoon een tabel op voor N elementen:

```
1 X1        f(X1)
2 ...
3 Xk-2      f(Xk-2)
4 Xk-1      f(Xk-1)
5 Xk        f(Xk)+epsilon
6 Xk+1      f(Xk+1)
7 Xk+2      f(Xk+2)
8 ...
9 Xn        f(Xn)
```

en reken die dan een paar stappen uit. Je zal zien dat de fout epsilon groter wordt en zich uitbreidt. Ook wisselt de fout steeds van teken: dit komt omdat als je de gedeelde differenties uitrekent, je soms -(f(Xk)+epsilon) moet doen, wat de epsilon negatief maakt. Als je vervolgens nog eens moet aftrekken, -(f(Xk)-epsilon) dus, dan wordt epsilon weer positief. Uiteindelijk bekom je volgend patroon in de fouten, dat eigenlijk de driehoek van pascal/binonium van newton is!

```
1                                   eps
2                           eps
3                   eps                     -4*eps
4           eps                     -3*eps
5 eps                -2*eps                  6*eps
6           -eps                    3*eps
7                   eps                     -4*eps
8                           -eps
9                                   eps
```

enzoverder. De fout verbreedt dus, maar de tabel wordt kleiner. Uiteindelijk zal je dus met 1 getal overblijven (want de f(x)'en worden allemaal 0 als je blijft doorrekenen. Stel dat dat getal bijvoorbeeld -10*eps is. Je zoekt in de driehoek van pascal op waar die "10"staat, en zie je dat op de 6e rij staat tussen 1 5 10 12 10 5 1. De 10 staat op de 3e plaats links, maar ook op 3e plaats rechts. Dit komt overeen met k of n-k = 3, dus het punt waar de fout op zit staat op derde plaats in de tabel OF op de n-3 plaats.

18 NR na 1 iteratiestap

Gegeven: Maple prints.

Gevraagd:

- Verklaar waarom de methode van Newton Raphson in 1 iteratiestap op afrondingsfouten na de exacte oplossing vindt!
- Wat is de orde van de convergentie en wat is de convergentiefactor? = reeds beantwoord
- Verklaar in detail waarom de totale stap vereenvoudigde Newton Raphson zo traag convergeert.

Antwoord: Veronderstel dat we het nulpunt willen vinden van een lineaire functie in één variabele, f(x) = ax+b. We weten dat het nulpunt gelijk is aan $\frac{-b}{a}$. Hoe zou NR dit nu vinden? stel a=3 en b=2, en we starten met startwaarde x_0=4. We hebben f(4)=14 en f'(4)=3. Dit wil zeggen dat als we X verhogen met 1, we f(x) verhogen met 3 (interpretatie afgeleide). Om vanaf de beginwaarde x_0 tot aan nul te raken, moeten we een vermindering van 14 hebben, daarvoor moet de volgende benadering $\frac{14}{3}$ lager liggen dan de huidige benadering. We passen nu NR toe
$x_1 = x_0$ - $\frac{f(x_0}{f'(x_0)}$
dit geeft 4 - $\frac{14}{3} = \frac{-2}{3}$ voor onze volgende benadering, wat gelijk is aan het nulpunt. Voor lineaire functies is het dus duidelijk dat deze benadering altijd zal werken in één stap.

19 Conditie van nulpunten

Gegeven: $x^2 + 2x + c$ $(c < 1)$

Gevraagd: Bespreek de conditie van de nulpunten. Is de conditie goed/slecht voor de abolute of relatieve fout?

Informatie: P.238 ev

Antwoord: Een nulpunt (x^*) is slecht geconditioneerd als $|f'(x^*)|$ klein is, en goed geconditioneerd anders.
$f(x) = x^2 + 2x + c$
Nulpunt: $x^* = \frac{-2 \pm \sqrt{4-4c}}{2}$
$f'(x) = 2x + 2$
Nulpunt invullen in afgeleide:
$f'(x^*) = 2(\frac{-2 \pm \sqrt{4-4c}}{2}) + 2$
$=> -2 \pm \sqrt{4-4c} + 2 = \pm\sqrt{4-4c}$ voor $c < 1$
Als c ≈ 1 is $\sqrt{4-4c}$ bijna 0, en is de wortel slecht geconditioneerd. (Absolute fout)
Voor de relatieve fout kunnen we de conditie hier niet nagaan, want we moeten delen door $f(x^*) = 0$.

20 Schommeling fout NR

Gegeven: de functie $f(x) = (x - 1,1)^5$ (maar uitgewerkt, dus $f(x) = x^5 - 5,5 * x^4 + ...$, en er stond niet bij dat dit gelijk was aan $(x - 1,1)^5$). Er is een Maplecode gegeven waarmee we via Newton-Raphson het nulpunt 1,1 willen bepalen. Men tekent de relatieve fouten en de grafiek daalt lineair, en schiet plots omhoog, daalt weer en terug omhoog, ... ||||...

Gevraagd: Verklaar wat je ziet in de grafiek en geef een variant van Newton-Raphson waar dit probleem niet opduikt.

Antwoord: een fuctie f(x) van de 6de graad x = 1.1 is een nulpunt een hoop matlab-code die ik niet verstond (alle matlab-oefenzittingen gebrost :-s) uit de commentaar was duidelijk dat ze Newton-Raphson toepasten een grafiek waarin duidelijk is dat de fout telkens kleiner wordt tot ongeveer 40 iteratiestappen, en daarna terug omhoog schiet, en terug zacht naar beneden gaat (en zo herhaalt zich dat). Gevraagd: leg in detail deze grafiek uit kun je een betere methode bedenken? oplossing : als ge de afgeleide van de gegeven functie berekent, dan is deze voor het opgegeven nulpunt ook nul, net zoals de derde en vierde afgeleide. We hebben dus te maken met een nulpunt met multipliciteit. =¿ gevolg: newton-raphson = 1 - 0/0 Alternatief =¿ Whittaker

21 Vragen opgelost in handgeschreven nota's

21.1 Examen 7 Juni 2010

21.1.1 Vraag 1

Gegeven: Gegeven de volgende gegevens: $f[x0, x0, x1, x1] = 1$; $f''(x0) = 0$; $f''(x1) = 0$; $f'(x_0) = 0$; $f'(x1) = 0$; $f(x_0) = 2$; $x_0 = 0$; $x_1 = 1$;

Gevraagd: Kan je hiermee f(x1) uitrekenen?

21.1.2 Vraag 2

Gegeven: Gegeven de iteratieformule $x(k+1) = (x(k)-a)^2-1$, met $a > 0$.

Gevraagd: Voor welke rele waarden van a en $x(0)$ is er convergentie, en naar welke waarden van x gebeurt dit? Wanneer treedt er kwadratische convergentie op?

21.1.3 Vraag 4

Gegeven: Gegeven maple code waar methode van de machten werd op toegepast. De gegevens werden erin genormeerd en mu werd uitgezet op de grafiek. Daarop was te zien hoe die convergeerde naar de dominante eigenwaarde. Ook werd er een grafiek gegeven die de relatieve fout van mu (in logaritmische schaal) uitzet tov het aantal iteratiestappen. Die was dalend en in 'rechte'-vorm

Gevraagd: Je moest de grafieken bespreken en een bijvraag was hoe je op de laatste grafiek de convergentiesnelheid kon zien.

21.2 Examen 8 Juni 2010

21.2.1 Vraag 1

Gegeven: Een fuctie f(x) van de 6de graad x = 1.1 is een nulpunt een hoop matlab-code die ik niet verstond (alle matlab-oefenzittingen gebrost :-s) uit de commentaar was duidelijk dat ze Newton-Raphson toepasten een grafiek waarin duidelijk is dat de fout telkens kleiner wordt tot ongeveer 40

iteratiestappen, en daarna terug omhoog schiet, en terug zacht naar beneden gaat (en zo herhaalt zich dat).

Gevraagd: Leg in detail deze grafiek uit kun je een betere methode bedenken?

21.2.2 Vraag 2

Gegeven: Veelterm van nen bepaalde graad (5de graad?) De exacte integraal van deze veelterm van -1 tot 1 is een bepaalde waarde (waarde is gegeven) We gaan de integraal bepalen met twee kwadratuurformules. Weer matlabcode (:-s) x1 = -alfa, x2 = 0, x3 = alfa H1, H2 en H3 gegeven

Eerste kwadratuurformule: alfa heeft bepaalde waarde de waarde van de kwadratuurformule is exact de opgegeven waarde
Tweede kwadratuurformule: ander alfa waarde, de rest hetzelfde. de waarde van de kwadratuurfomule geeft nu een andere oplossing dan hierboven

Gevraagd:

- Kunnen we hieruit besluiten dat de nauwkeurigheidsgraad van de eerste kwadratuurformule beter is dan de tweede? (uiteraard niet... anders zou hij't zo niet vragen)
- Bereken de nauwkeurigheidsgraad van de kwadratuurforumules

21.2.3 Vraag 3

Gegeven: $f(x) = x - x + 1$ Weeral matlabcode (:-s) $x* = 1$ Door middel van substitutiemethodes word deze functie benaderd, ne keer langs links, en langs rechts (ik dacht voor x=0,9 en x=1,1) Twee grafieken gegeven, op de ene (x=0,9) zie je fout kleiner worden, op andere zie je fout groter worden. Zie dus figuur 2.10 op pagina 221. bepalen ofzo

Gevraagd: Verklaar. Bijvraag was iets van convergentiefactor.

21.2.4 Vraag 4

Gegeven:

$$\begin{pmatrix} 2 & 1 & -1 \\ 0 & 3 & -5 \\ 0 & 0 & -2 \end{pmatrix}$$

en X = (-1,1,1)

Gevraagd: Wat gebeurd er als we *von Mises* op een matrix A uitvoeren met en een startvector X.

Zorg dat je na 1 uur minstens 1 vraag opgelost hebt en na 2 uur 2 vragen !

Vraag: 1

We hebben de volgende gegevens van de functie $f(x)$.

- $f[x_0, x_0, x_1, x_1] = 1$
- $f''(x_0) = 0$
- $f''(x_1) = 0$
- $f'(x_0) = 0$
- $f'(x_1) = 0$
- $f(x_0) = 2$
- $x_0 = 0$
- $x_1 = 1$

Kunnen we met behulp van deze gegevens de waarde van de functie bepalen in x_1, d.w.z., $f(x_1)$?

Numerieke Wiskunde: Voorbeeldexamen.

Marc Van Barel

Vraag 1

↳ Gedeelde differentie:

$$f[x_0, x_0, x_1, x_1] = 1$$

$$\lim_{\substack{x_2 \to x_0 \\ x_3 \to x_1}} f[x_2, x_0, x_2, x_3] = 1$$

$$\lim_{x_0 \to x_1} f[x_0, x_1] = \lim_{x_0 \to x_1} \frac{f(x_0) - f(x_1)}{x_0 - x_1} = f'(x_1)$$

→ Geen veelterminterpolatie → het kan eender welke functie zijn.

$$f[x_2, x_0, x_1, x_3] = \frac{f[x_2, x_0, x_1] - f[x_0, x_1, x_3]}{x_2 - x_3}$$

$$= \frac{\left(\frac{f[x_2, x_0] - f[x_0, x_1]}{x_2 - x_1}\right) - \left(\frac{f[x_0, x_1] - f[x_1, x_3]}{x_0 - x_3}\right)}{x_2 - x_3}$$

$$= \frac{\left(\frac{0 - (2-\alpha)}{-1}\right) - \left(\frac{(2-\alpha) - 0}{-1}\right)}{-1}$$

$$f[x_0, x_1] = \frac{f(x_0) - f(x_1)}{x_0 - x_1} = \frac{2 - \alpha}{-1}$$

$$\boxed{\alpha = \frac{3}{2}}$$

Gedeelde Differenties geven enkel info over de waarde van de functie bij welbepaalde x → geen info over soort functie.

Vraag: 2

Bepaal de gewichten $H_{-\frac{1}{2}}$, H_0 en $H_{\frac{1}{2}}$ zodanig dat de kwadratuurformule

$$\int_{a-h}^{a+h} f(x)dx \approx H_{-\frac{1}{2}} f(a - \frac{h}{2}) + H_0 f(a) + H_{\frac{1}{2}} f(a + \frac{h}{2})$$

een zo hoog mogelijke nauwkeurigheidsgraad heeft.
Wat is deze nauwkeurigheidsgraad?

Vraag 2

nauwkeurigheidsgraad = d

kwadratuurformule $I_k(f)$

exacte integraal $I(f)$

$\forall p$: p veelterm van graad $\leq d$ $\quad I_k(p) = I_p$

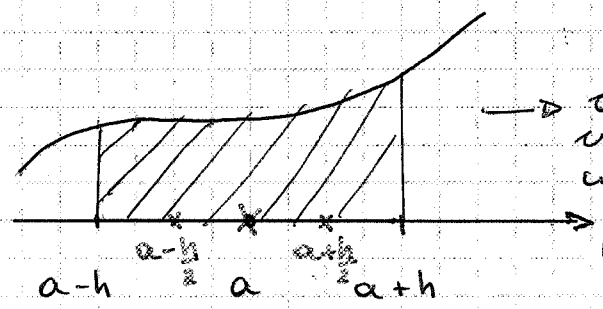
$\exists q$: q graad $d+1$ $\quad I_k(q) \neq I(q)$

basis van alle veeltermen van graad $\leq d$

$1, X, X^2, \ldots, X^d$

↳ Rekenwerk kan verminderd worden door het kiezen van een andere basis

$$I_k(f) = \alpha f\left(a - \frac{h}{2}\right) + \beta f(a) + \gamma f\left(a + \frac{h}{2}\right) \approx \int_{a-h}^{a+h} f(x)\,dx$$



→ verschuiven van grafiek o verandert niets aan de integraal indien het interval mee verschuift wordt.

Opm: nieuwe basis: $1, x-a, (x-a)^2, \ldots$

Of

leg a in 0 oorsprong en neem oorspronkelijke basis

$$I_k(1) = \alpha 1 + \beta 1 + \gamma \cdot 1 = \int_{-h}^{h} 1 \cdot dx = 2h$$

$$I_k(x) = \alpha\left(-\frac{h}{2}\right) + \beta(0) + \gamma\left(\frac{h}{2}\right) = \int_{-h}^{h} x\,dx = \frac{2h^2}{2} = 0$$

$$I_k(x^2) = \alpha\left(-\frac{h}{2}\right)^2 + \beta(0) + \gamma\left(\frac{h}{2}\right)^2 = \int_{-h}^{h} x^2\,dx = \frac{2h^3}{3}$$

} Stelsel oplossen

$\alpha = \gamma$

$\frac{2\alpha}{4} = \frac{2h^3}{3} \Rightarrow \alpha = \frac{4h}{3}$

$\beta = \frac{-2h}{3}$

$$I_k(x^3) = \alpha\left(-\frac{h}{2}\right)^3 + \beta 0 + \gamma\left(\frac{h}{2}\right)^3 = 0 = \int_{-h}^{h} x^3\,dx$$

$I_k(x^4) \neq I(x^4) \Rightarrow$ graad 3

Vraag: 3

Zij het iteratieproces $x^{(k+1)} = (x^{(k)} - a)^2 + 1$, $a > 0$ gegeven. Voor welke reële waarden van a en $x^{(0)}$ is er konvergentie naar een reële waarde x ? Wanneer treedt er kwadratische konvergentie op ?

Vraag 3

↳ Iteratieve methoden

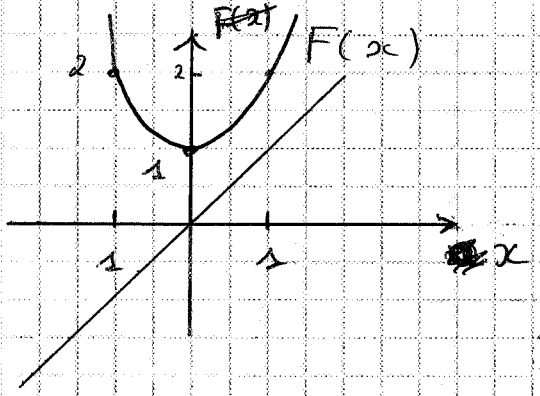
$x^{(k+1)} = F(x^{(k)})$ ← grote F!!!

$F(x) = (x-a)^2 + 1$

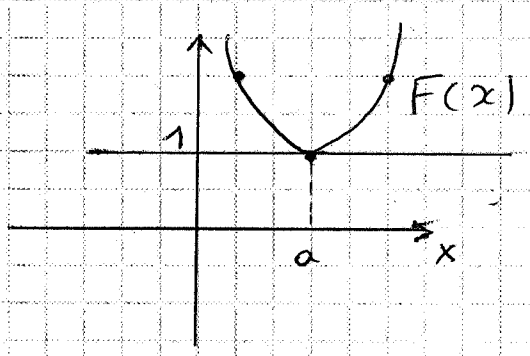
$a > 0$

Maak een grafiek van $F(x)$

$\boxed{a = 0} \rightarrow x^2 + 1$



Voor eender welke a



$X^* = F(x^*)$

$X = (x-a)^2 + 1$

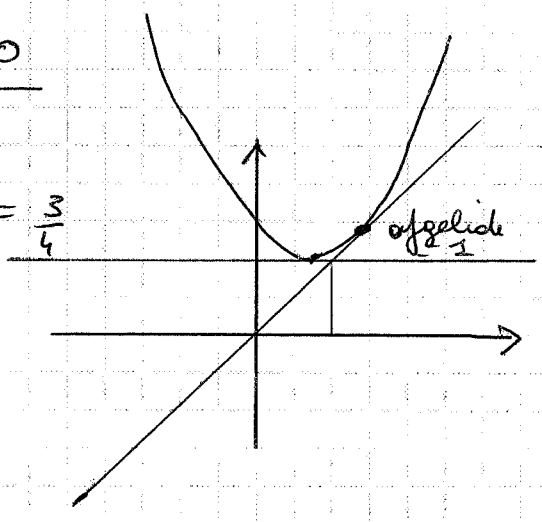
$X^2 - 2aX - X + 1 + a^2 = 0$

$x^2 - (2a+1)x + (1+a^2) = 0$

$$x^*_{1,2} = \frac{(2a+1) \pm \sqrt{(2a+1)^2 - 4(1+a^2)}}{2}$$

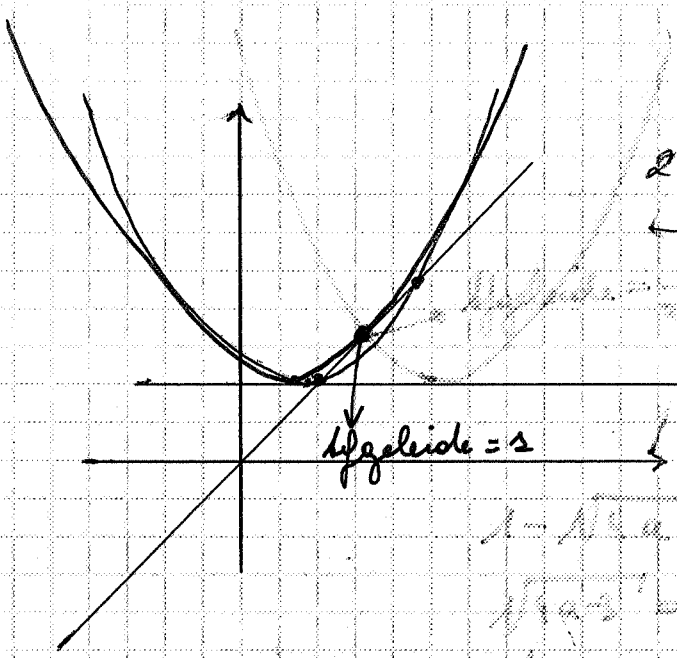
$$= \frac{(2a+1) \pm \sqrt{\underbrace{4a-3}_{\geq 0}}}{2}$$

$4a - 3 \geq 0 \Rightarrow \boxed{a \geq \frac{3}{4}} \rightarrow a = \frac{3}{4}$

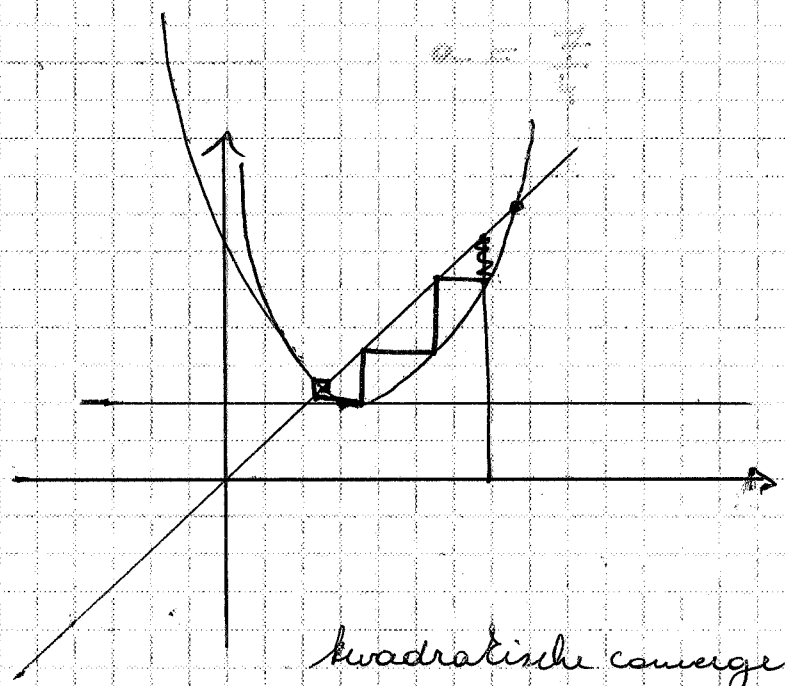


$F'(x^*_{1,2}) = 1 \pm \sqrt{4a-3}$

$F'(x) = 2(x-a)$



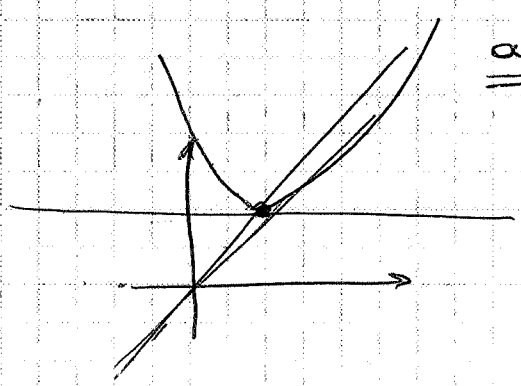
$$\frac{2\left(\frac{3}{4}\right)+\left(\frac{4}{4}\right)}{2}=\frac{5}{4}$$



kwadratische convergentie als $\rho=0$!

Convergentiefactor $\rho = F'(x^*_{1,2}) = 1 \mp \sqrt{4a-3} = 0$

$a = 1$



Zorg dat je na 1 uur minstens 1 vraag opgelost hebt en na 2 uur 2 vragen !

Vraag: 4

Methode van de machten

(Dit is de afdruk van een Maple Worksheet)

We rekenen met 16 beduidende decimale cijfers.

```
> restart:with(linalg):lp:=16:Digits:=lp;
```

```
Warning, the protected names norm and trace have been redefined and
unprotected
```

$$Digits := 16$$

We beschouwen de matrix A.

```
> A:=array(1..3,1..3,[[9,1,-1],[0,10,-5],[0,0,8]]);
```

$$A := \begin{bmatrix} 9 & 1 & -1 \\ 0 & 10 & -5 \\ 0 & 0 & 8 \end{bmatrix}$$

Omdat de matrix A bovendriehoeks is, vinden we de eigenwaarden van A op de hoofddiagonaal: 9, 10, 8. De dominante eigenwaarde is dus 10. We gaan deze eigenwaarde proberen te benaderen met behulp van de methode van de machten. We nemen als startvector x0. We zullen K=100 iteraties uitvoeren.

```
> x0:=array(1..3,1..1,[[-1.0],[1.0],[1]]);K:=100;
```

$$x0 := \begin{bmatrix} -1.0 \\ 1.0 \\ 1 \end{bmatrix}$$

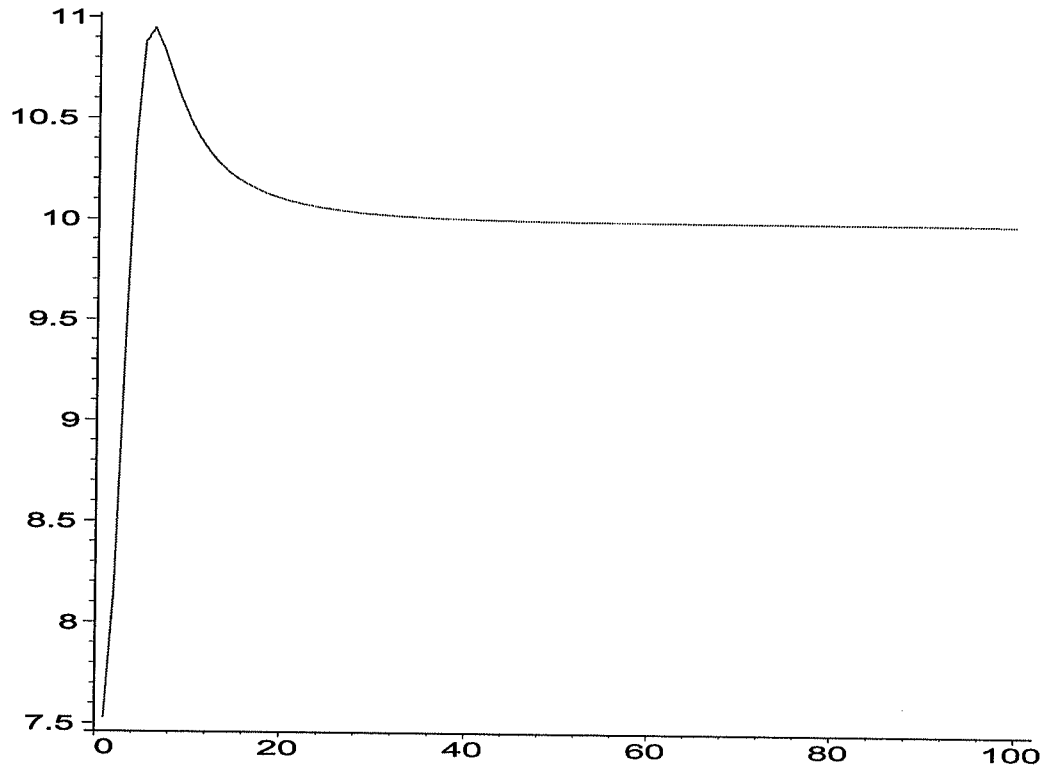
$$K := 100$$

We voeren de methode van de machten met normalisatie uit.

```
> Y:=x0/norm(x0,2):
> muvec:=array(1..K):
> for i from 1 to K do
> Z:=evalm(A&*Y):
> mu:=norm(Z,2);
> Y:=Z/mu:
> muvec[i]:=[i,mu]:
> od:
```

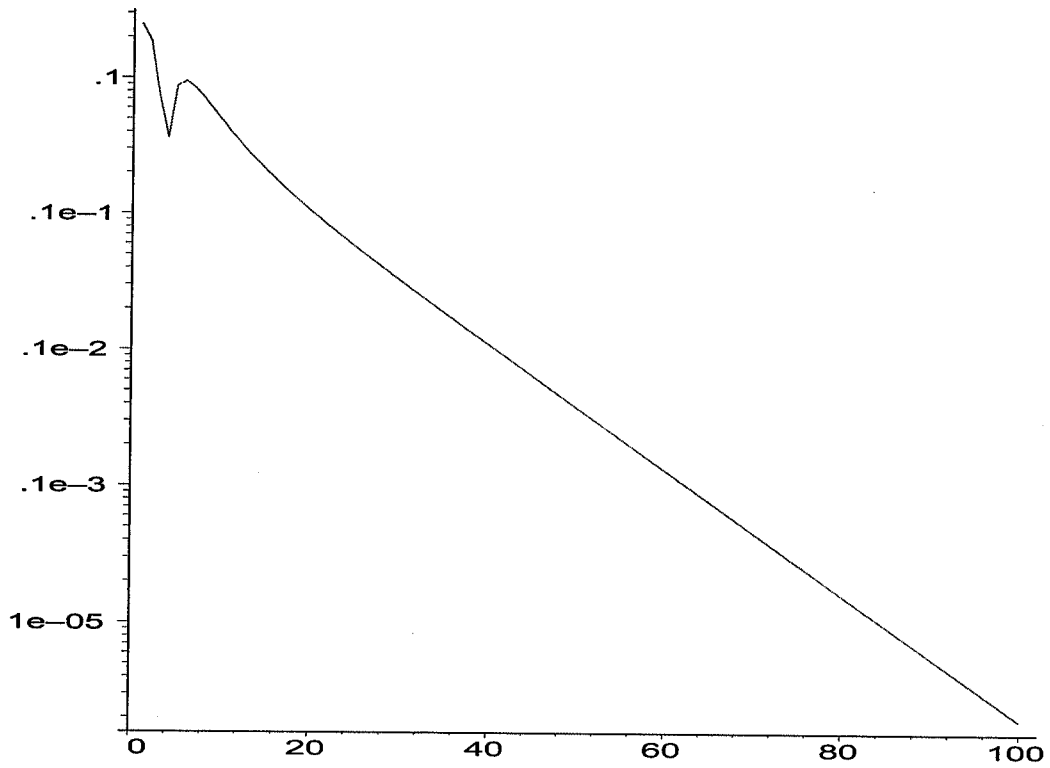
In elke iteratiestap werd er een waarde van mu berekend. Deze waarde wordt hieronder geplot.

```
> with(plots):plot(muvec,thickness=2);
```



We zien dat deze waarde naar 10 konvergeert. In de hiernavolgende grafiek plotten we de relatieve fout.

```
> relerr:=array(1..K):
> for i from 1 to K do
> relerr[i]:=[i,abs(muvec[i][2]-10.0)/10.0];
> od:
> logplot(relerr,thickness=2);
```



Verklaar de grafieken die hierboven gegenereerd werden. Wat is de orde van konvergentie en wat is de konvergentiefactor?

Wat zou er gebeuren als we in de methode hierboven geen normalisatie zouden gebruiken?

Vraag 4

De fout daalt superlineair
↳ lineaire convergentie

$$A = \begin{bmatrix} 9 & 1 & -1 \\ 0 & 10 & -5 \\ 0 & 0 & 8 \end{bmatrix} \qquad X_0 = \begin{bmatrix} -1 \\ 1 \\ 1 \end{bmatrix}$$

$\lambda_1 = 10 \qquad \lambda_2 = 9 \qquad \lambda_3 = 8$

$(A - \lambda I)\, X = 0$

$\lambda_1 = 10$

$$\begin{bmatrix} -1 & 1 & -1 \\ 0 & 0 & -5 \\ 0 & 0 & -2 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = 0 \Rightarrow x_3 = 0$$

$x_1 = x_2$

$x_1 = 1$

$x_2 = 1$

$$E_1 = \begin{bmatrix} 1 \\ 1 \\ 0 \end{bmatrix}$$

$\lambda_2 = 9$

$$\begin{bmatrix} 0 & 1 & -1 \\ 0 & 1 & -5 \\ 0 & 0 & -1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$$

$x_3 = 0$

$x_2 = 0$

$x_1 = 1$

$$E_2 = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}$$

$\lambda_3 = 8$

$$\begin{bmatrix} 1 & 1 & -1 \\ 0 & 2 & -5 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} \Rightarrow E_3 = \begin{bmatrix} -3 \\ 5 \\ 2 \end{bmatrix}$$

~~$x_3 = 1 \Rightarrow x_2 = \frac{5}{2}$~~

~~$x_1 = \frac{7}{2}$~~ ~~$\Rightarrow E_3 = \begin{bmatrix} 7 \\ 5 \\ 2 \end{bmatrix}$~~ $E_3 = \begin{bmatrix} -3 \\ 5 \\ 2 \end{bmatrix}$

$$x_k = A^k x_0$$

$$x_0 = \alpha_1 E_1 + \alpha_2 E_2 + \alpha_3 E_3$$

$$\boxed{\alpha_1 = -\frac{3}{2}}$$

$$\alpha_2 = 2$$

$$\alpha_3 = \frac{1}{2}$$

$$x_k = A^k x_0 = \alpha_1 \lambda_1^k E_1 + \alpha_2 \lambda_2^k E_2 + \alpha_3 \lambda_3^k E_3$$

$$= \lambda_1^k \left(\alpha_1 E_1 + \left(\frac{\lambda_2}{\lambda_1}\right)^k \alpha_2 E_2 + \left(\frac{\lambda_3}{\lambda_1}\right)^k \alpha_3 E_3 \right)$$

Dominant

convergentiefactor $= \frac{9}{10} = \frac{\lambda_2}{\lambda_1}$

λ_1

$\lambda_2 = 10^{-3}$

$\lambda_3 = 2 \cdot 10^{-6}$

$$\rho^{60} = \frac{2 \cdot 10^{-6}}{2 \cdot 10^{-3}} \approx 0,906$$

Wat zonder normalisatie?

Overloop: vector groeien als 10^k groeit!

Op examen mogelijk niet generiek gedraagt.

$\boxed{\alpha_1 = 0}$ controleren