

Samenvatting CPN v3

Origineel door Kjell De Mars. Gebaseerd op het cursusboek *Computergesteund Probleemoplossen in de Natuurkunde, 10e editie*, en de lessen van prof. dr. Ronald Cools. Andere bronnen zijn vermeld in de tekst. Maak er je eigen ding van, en sharing is caring (maar vermeld je bronnen!).

Aangepast door: ... (vul aan)

Opmerking: Dit is niet meer gewoon een "samenvatting" te noemen maar ook een uitgebreide poging om een ietwat vage cursus op te helderen en de hoofdzaken netjes en uitgediept op een rij te zetten. Let op, er zullen ongetwijfeld nog fouten in staan. Aan het einde is er een lijst van nuttige MATLAB bevelen en hoe ze te gebruiken voorzien.

Plekken waar ik zelf vind dat er uitleg ontbreekt of er aan twijfel zijn aangeduid met een "(?)".

HS1: Inleiding

1.3 Getallen in een computer

❓ Wat is een voorstelbaar getal en hoe wordt het voorgesteld? >

Een computer kan maar een eindig aantal getallen voorstellen (die natuurlijk in een eindig *bereik* liggen), de **voorstelbare getallen**.

Een voorstelbaar getal x wordt in basis b geschreven als

$$x = b^e \cdot m,$$

waarbij de **exponent** $e \in \{e_{\min}, \dots, e_{\max}\}$ en de **mantisse** $m \in \{(-1)^s 0.d_1 \dots d_p \mid s \in \{0, 1\}, d_i \in \{0, 1, \dots, b-1\}\}$ met p de zogenaamde **precisie**. We kunnen dus ook noteren

$$x = b^e \cdot (-1)^s \sum_{j=1}^p d_j b^{-j}.$$

Bijgevolg heeft x niet noodzakelijk een unieke voorstelling, wat men oplost door af te spreken dat $d_1 \neq 0$.

📖 **DEFINITIE:** genormaliseerde bewegende kommagetallen >

De verzameling voorstelbare getallen aldus bekomen, uitgebreid met 0.
Genoteerd als \mathbb{F}_N .

② Wat is het probleem met genormaliseerde bewegende kommagetallen? > Hoe wordt het opgelost? (?)

Getallen met $d_1 \neq 0$ die zo klein zijn dat $e < e_{\min}$ kunnen zo niet voorgesteld worden. Dit is een enorm probleem als we bijvoorbeeld zoeken naar nulpunten van een vergelijking, want we kunnen deze vaak enkel benaderen en het loont als we dat zo precies mogelijk kunnen doen. We kunnen ze terug toevoegen door te eisen dat $d_1 \neq 0$ enkel mag als $e = e_{\min}$.

① DEFINITIE: Gedenormaliseerde of subnormale getallen >

Bewegende kommagetallen met een dusdanig kleine absolute waarde dat ze niet als genormaliseerd getal kunnen voorgesteld worden. Genoteerd als \mathbb{F}_D .

Ze behoren allemaal tot het (open) interval $(-b^{e_{\min}-1}, b^{e_{\min}-1})$ en hebben $m \in [b^{-p}, b^{-1} - b^{-p}] = [0.0 \dots 01, 0.0\delta \dots \delta]$ (met $\delta = b - 1$).

We maken het onderscheid tussen genormaliseerd en gedenormaliseerd m.b.v. een **hidden bit**. In een binair talstelsel is $d_1 \neq 0$ uitzonderlijk als $d_1 = 1$. We kunnen de informatie dat we met een genormaliseerd getal te doen hebben dus onthouden in een niet gebruikte waarde van de exponent in plaats van "de bit[patroon(?)] van d_1 ", gewoonlijk de bit van $e_{\min} - 1$. (?)

② Hoe ziet de afstand tussen genormaliseerde getallen er uit en waarom? >

De mantisse van genormaliseerde getallen kan elke waarde aannemen van $m_{\min} = b^{-1}$, in stapjes van $\Delta m = b^{-p}$, tot en met $m_{\max} = 1 - b^{-p}$. Bijgevolg is het aantal voorstelbare getallen in $\mathbb{F}_N \cap [b^{e-1}, b^e]$ onafhankelijk van e ($> e_{\min} - 1$).

De afstand tussen opeenvolgende getallen uit $[b^e, b^{e+1}] \subset \mathbb{F}_N$ is $\Delta x = b^e \Delta m = b^{e-p}$, waardoor de *relatieve* afstand ongeveer constant blijft, en bovendien naar boven begrensd is door de machinenauwkeurigheid:

$$\frac{\Delta x}{x} = \frac{b^{e-p}}{m(x) b^e} = \frac{b^{-p}}{m(x)} \leq \frac{b^{-p}}{m_{\min}} = b^{1-p}.$$

DEFINITIE: machinenauwkeurigheid/machineprecisie >

Meestal gedefinieerd als kleinste getal ϵ waarvoor $1 + \epsilon$ als groter dan 1 wordt voorgesteld. Dit is echter problematisch omdat bij optelling afrondingsfouten gemaakt worden.

Opmerkingen:

- De afstand tussen een voorstelbaar reëel getal R en het volgende voorstelbaar getal is dan ongeveer ϵR .
- Indien geen afronding of extra cijfers gebruikt worden zou $\epsilon = b^{1-p}$, wat met moderne machines ongeveer $10^{-16} = 10^{-14} \%$ is.

Hoe kan je deze benaderen in MatLab? >

```
eps = 1;  
while eps + 1 > 1  
    eps /= 2;  
eps *= 2
```

1.5 Afrondingen

DEFINITIE: afrondingsregel >

Afbeelding

$$\square : \mathbb{R} \rightarrow \mathbb{F} : x \mapsto \square x.$$

Deze voldoen praktisch aan:

- $\forall x \in \mathbb{F} : \square x = x$,
- $\forall x, y \in \mathbb{R} : x \leq y \Rightarrow \square x \leq \square y$,

waaruit volgt dat er een $\hat{x} \in [x_1, x_2]$ tussen twee opeenvolgende bewegende kommagetallen $x_1 < x_2$ bestaat zodat

- $x \in [x_1, x_2] \wedge x < \hat{x} \Rightarrow \square x = x_1$, en
- $x \in [x_1, x_2] \wedge x > \hat{x} \Rightarrow \square x = x_2$.

Wat er met \hat{x} zelf gebeurt hangt af van de specifieke afrondingsregel.

Enkele varianten zijn:

☰ Afkappen >

Ook wel *round towards zero* genoemd. Hier is

$$\hat{x} := \text{sgn}(x) \max\{|x_1|, |x_2|\}.$$

☰ Optimale afronding >

$$\hat{x} := \frac{x_1 + x_2}{2}$$

Dit komt neer op het afronden naar het dichtstbijzijnde bewegende kommagetal. Er zijn hier wel twee mogelijkheden om \hat{x} af te ronden:

- Afronden weg van nul.
- Afronden naar het getal met als minste beduidende cijfer in de mantisse een even getal. Dit wordt *unbiased* genoemd.

1.6 Absolute en relatieve fout

🔍 Geef de definities. >

📘 DEFINITIE: absolute afrondingsfout

$$\varepsilon(x) := \square x - x$$

📘 DEFINITIE: relatieve afrondingsfout

$$\rho(x) := \frac{\varepsilon(x)}{x}$$

De volgende stelling is goed om abstractie te maken van de binaire getalvoorstelling. Ze zegt dat een getal wordt afgerond tot maximaal de machinenauwkeurigheid. De meeste resultaten over machinenauwkeurigheid volgen uit volgende stelling. (?)

📘 STELLING >

Voor elke $x \in \mathbb{R}$ binnen het bereik van \mathbb{F}_N bestaat er een $\rho \in \mathbb{R}$ zo dat

$$\square x = x(1 + \rho) \quad \text{met} \quad |\rho| \leq \epsilon,$$

met ϵ de machinenauwkeurigheid.

1.7 Foutenvoortplanting

② Geef de 2 belangrijkste voorbeelden van problemen van rekenen met bewegende kommagetallen, de reden ervan, en hun oplossingen. >

In feite is rekenen met bewegende kommagetallen noch associatief, commutatief, als distributief. Daar komt bij dat decimalen niet eindig binair voorstelbaar zijn.

- Als men rechtstreeks vele kleine getallen optelt loopt men het gevaar een behoorlijk onnauwkeurig resultaat te bekomen. Men kan best optellen van klein naar groot (in absolute waarde), omdat kleine bewegende kommagetallen dichter bij elkaar liggen. Zo maakt men zo nuttig mogelijk gebruik van de relatief hoge resolutie op kleine voorstelbare getallen.
- Het verschil van twee bijna gelijke getallen kan een onnauwkeurig resultaat leveren, omdat de relatieve fout $\rho_{x+y} \propto \frac{1}{x+y}$. Dit zegt dat hoe kleiner de echte afstand tussen de echte getallen is, hoe meer informatie we daarover verliezen wegens de (relatief grote) afstand tussen voorstelbare getallen. Dit kan men alleen vermijden door de berekening anders te formuleren.

1.8 Conditie van een probleem

② Motiveer/definieer het begrip *conditiegetal*. (?) >

Een typisch probleem heeft invoer en uitvoer, gegevens en gevraagd—beiden een stel getallen met de uitvoer éénduidig bepaald door de invoer. We kunnen de invoer samen voegen in een vector $x \in \mathbb{R}^m$ en de uitvoer in een vector $y \in \mathbb{R}^n$, en het probleem dan bekijken als een zwarte doos met als invoer x en als uitvoer $y = f(x)$.

Het **conditiegetal** (vaak κ) geeft in zekere zin een ruwe bovengrens (worst-case scenario) voor de factor waarmee de fout op de invoer door de eigenschappen van (een linearisatie van) het probleem zelf wordt vergroot. Verder zegt $k = O(\kappa)$ ook ruwweg dat je zo'n k beduidende cijfers per toepassing van f kan verliezen bovenop alle andere benaderingsfouten. (?)

- Een probleem met een "laag" conditiegetal wordt *goed geconditioneerd* genoemd.

- Een probleem met een "hoog" conditiegetal wordt *slecht geconditioneerd* genoemd.

DEFINITIE: (relatieve) conditiegetal van een scalaire functie op de reële getallen >

$$(\text{cond} f)(x) := \left| \frac{x f'(x)}{f(x)} \right|,$$

waar we veronderstellen dat f afleidbaar is.

Afleiding:

$$\begin{aligned} \delta y &= f(x + \delta x) - f(x) \approx f'(x) \delta x \\ &\quad \Downarrow \\ \frac{\delta y}{y} &\approx \frac{x f'(x)}{f(x)} \frac{\delta x}{x}. \end{aligned}$$

DEFINITIE: (relatieve) conditiegetallen van een regulier stelsel (?) >

De constanten zijn het rechterlid in het stelsel $Ax = b$.

Stel dat we enkel perturbatie hebben op b . Dan hebben we te maken met een afbeelding $f : \mathbb{R}^n \rightarrow \mathbb{R}^n : b \mapsto A^{-1}b$. Met wat fantasie leiden we uit de vorige definitie af dat $\frac{\partial f}{\partial b} = A^{-1}$ (?) zodat

$$(\text{cond} f)(b) = \frac{\|b\| \|A^{-1}\|}{\|A^{-1}b\|} = \frac{\|Ax\| \|A^{-1}\|}{\|x\|}.$$

Om iets te zeggen over de invloed van perturbaties op A definiëren we het conditiegetal van A als de slechtst mogelijke variant hiervan:

$$\text{cond} A = \kappa_{\text{norm}}(A) := \max_{x \neq 0} (\text{cond} f)(b) = \|A\| \|A^{-1}\|.$$

De keuze van norm is hier niet belangrijk omdat enkel de ordegroottes tellen, en de conclusies vergelijkbaar zullen zijn onder verschillende normen (?).

HS2: Stelsels lineaire vergelijkingen

2.2 Vierkante stelsels

We beschouwen in deze sectie exclusief stelsels van de vorm

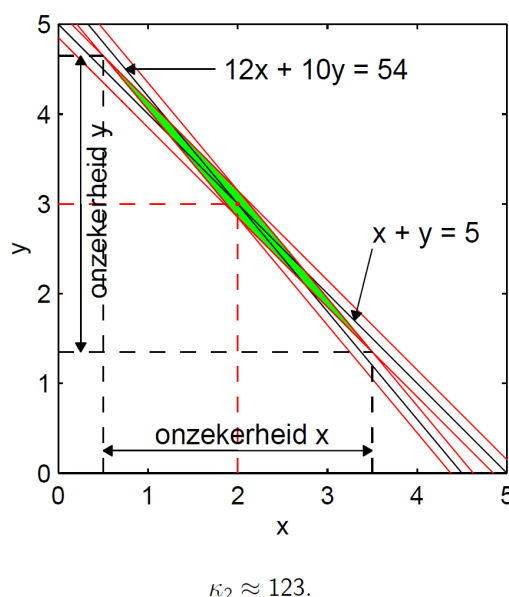
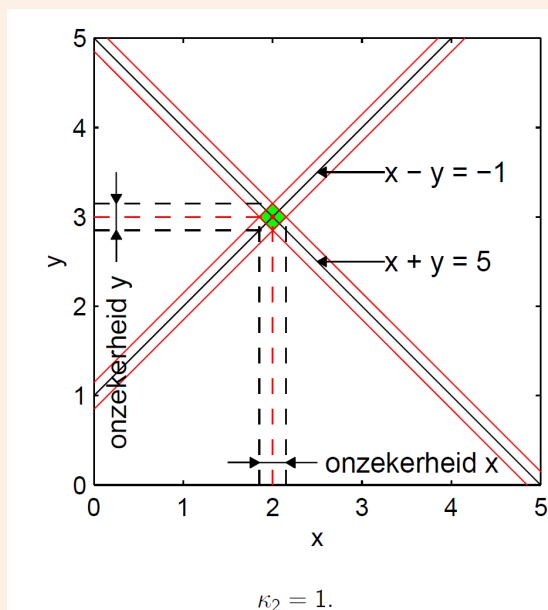
$$Ax = b, \quad A \in \text{GL}_n(\mathbb{R}), \quad x, b \in \mathbb{R}^n.$$

2.2.1 Grafische interpretatie conditiegetal

🔗 Beschrijf hoe dit er grafisch uit ziet. >

We kunnen A beschouwen als lineaire transformatie tussen vectorruimten. Door afrondingsfouten is er een onzekerheid op de gegeven vector x , die je kan voorstellen als een bolletje rond x in het domein. Dit bolletje wordt dan afgebeeld op een ellipsoïde waar Ax ergens in zit. Hoe groter deze ellipsoïde, hoe slechter het conditiegetal.

We kunnen dit preciezer maken op de volgende manier: De oplossing van een stelsel in n variabelen kan voorgesteld worden als het unieke gemeenschappelijke punt van alle $(n - 1)$ -dimensionale hypervlakken in de n -dimensionale ruimte gedefinieerd door de vergelijkingen er in. Omdat deze vlakken niet exact voorgesteld kunnen worden kan hun onzekerheid voorgesteld worden als een bepaalde dikte die deze vlakken hebben. Hoe kleiner de spreiding tussen deze vlakken, hoe meer hun onzekerheid zal overlappen, en dus hoe onzekerder de oplossing.



2.2.2 Gauss eliminatie

In numerieke software gebruikt men meestal een kleine variant van de *Gauss-eliminatie* methode (herleid A tot een bovendriehoeksmatrix U via Gauss-eliminatie en vind x via achterwaartse substitutie).

🔗 Wat is de orde van de Gauss-eliminatie methode? Wat betekent dit? Wat is een maat voor de stabiliteit ervan? >

De orde is n^3 (je hebt 3 lussen nodig). Dit betekent dat als je matrix 10x groter wordt, je 1000x meer operaties moet uitvoeren.

Numerieke instabiliteit van Gauss eliminatie met partiële pivoting kan optreden wanneer $\|U\|_a/\|A\|_a$ erg groot wordt, dus wanneer U erg grote elementen bevat in vergelijking met A .

❓ Beschrijf de variant. Welk voordeel biedt deze? >

Deze is genaamd **LU-decompositie**. We ontbinden $A = LU$, met L benedendriehoeks en U bovendriehoeks, en lossen het stelsel dan op door $Ly = b$ en $Ux = y$ in die volgorde op te lossen. Dit vraagt precies evenveel rekenwerk als Gauss eliminatie maar geeft ons het gereedschap om andere stelsels $Ax' = b'$ ook meteen op te lossen.

❓ Waarom is de variant niet altijd uitvoerbaar? Hoe kunnen we dat oplossen? >

Dit is enkel mogelijk als we A kunnen rij-herleiden zonder rijen van plaats te wisselen. (?)

Daarom volgende stelling.

ℹ️ STELLING >

Voor elke reguliere matrix A bestaat er een permutatiematrix P zo dat

$$PA = LU$$

bestaat. We zoeken bijgevolg oplossingen van $Ly = Pb$ en $Ux = y$.

[Verhoogt deze oplossing het rekenwerk ook niet weer aanzienlijk door telkens Pb te moeten berekenen(?)]

❓ In de praktijk zal men echter nog net iets anders doen. Wat en waarom? >

Naast fouten op de gegevens zijn er ook de fouten ten gevolge van rekenen met bewegende kommagetallen. Daarom zal men in de praktijk niet enkel rijen omwisselen wanneer een pivot 0 wordt, maar zal men bij het werken op de i -de kolom het absoluut grootste element ervan op of onder de diagonaal op de diagonaal plaatsen. Voor de reden zie sectie 2.2.4.

2.3 Overgedetermineerde stelsels

① DEFINITIE: overgedetermineerd stelsel >

Stelsel met meer vergelijkingen dan onbekenden.

① DEFINITIE: (lineaire) kleinste kwadraten benadering >

Methode voor het minimaliseren van de afstand $\sum r_i^2$ tussen een model en een verzameling datapunten.

② Wat kan de kleinste kwadraten benadering in deze context voor ons betekenen? Hoe wordt dit in de praktijk gedaan en waarom? >

Stel dat we een stelsel

$$Ax = b, A \in \mathbb{R}^{m \times n}, x \in \mathbb{R}^n, b \in \mathbb{R}^m, m > n$$

zonder exacte oplossing hebben. We kunnen dan nog steeds een kleinste kwadraten benadering zoeken.

Dit komt neer op het minimaliseren van $\|Ax - b\|$, wat neerkomt op het zoeken van de $y_0 = Ax_0$ in de kolomruimte van A die op minimale afstand van b ligt. Dit is evident de loodrechte projectie van b op de kolomruimte van A , waaruit volgt dat $y_0 - b$ loodrecht op dit vlak moet staan:

$$\begin{aligned} \forall x \in \mathbb{R}^n : \\ \langle Ax, Ax_0 - b \rangle &= (Ax)^T (Ax_0 - b) = 0 \\ \Leftrightarrow x^T (A^T Ax_0 - A^T b) &= 0. \end{aligned}$$

Aangezien dit moet gelden voor alle x vinden we dat de

① DEFINITIE: normaalvergelijking

$$A^T Ax_0 = A^T b$$

moet opgelost worden naar Ax_0 .

In de praktijk zal men, behalve voor kleine n , de normaalvergelijking niet rechtstreeks oplossen maar via haar speciale eigenschappen eerst omvormen tot een ander probleem, $Rx = Q^T b$ met $A = QR$ (QR-ontbinding). Hier zijn twee redenen voor:

- Het conditiegetal van R is gelijk aan dat van A , terwijl $\kappa_2(A^T A)$ van het normaalstelsel gelijk is aan $(\kappa_2(A))^2$.

- Bij het numeriek vormen van $A^T A$ kan informatie verloren gaan, zelfs zodanig dat voor reguliere A dit singulier wordt.

2.4 Overgedetermineerde stelsels met beperkingen

Deze komen voor als we de fouten op onze onafhankelijke veranderlijke niet verwaarlozen, en zullen we opnieuw via QR-ontbinding moeten oplossen.

🔍 Waarom kiest men ervoor beperking op deze stelsels te leggen? (?) >

Bij slecht geconditioneerde problemen kan de oplossing opblazen, en door beperkingen op te leggen wordt dit voorkomen. De beperking dat de norm van de coëfficiëntenvector vast is wordt (Tikhonov) regularisatie genoemd.

(Niet zeker wat hier meer over te zeggen valt.) (?)

HS3: Niet-lineaire benaderingen

3.1 Veeltermbenaderingen

🔍 Waar moet je voor opletten bij veeltermbenaderingen? Waarom? >

- Goede veeltermbenaderingen zijn niet noodzakelijk bruikbaar om de afgeleide van een functie in een punt, of zijn extrema, te benaderen. Dit omdat ze tussen de interpolatiepunten een heel ander gedrag kan vertonen dan de benadering. Voor het benaderen van een bepaalde integraal is dit minder problematisch.
- De punten waardoor geïnterpoleerd wordt zijn van groot belang voor de nauwkeurigheid! Gelijk verspreide punten zijn vaak de slechtste keuze, omdat een veelterm buiten het interval met de interpolatiepunten naar $\pm\infty$ wil gaan (horizontale asymptoten zijn dus als zeker problematisch), zodat ze naar de uiteindes toe de neiging zal hebben sterk te schommelen en de pan uit te vliegen. Daarom is het beter van meer punten aan de randen van het interval te nemen.

Opmerking: zie h3_weierstrass.

3.1.1 Interpolerende veeltermen

📖 DEFINITIE: Interpolerende veelterm >

Unieke veelterm van graad d in 1 veranderlijke die $d + 1$ (één meer dan de graad om de veelterm uniek te maken!) specifieke punten verbindt.

❓ **Welk probleem moet je oplossen om een interpolerende veelterm door de punten (x_i, y_i) te bepalen? Wat is de graad?** >

Doormiddel van het oplossen van het stelsel $Ac = y$, met c de coëfficiënten en $A_{ij} = x_i^j$. Zo'n A wordt een *Vandermonde matrix* genoemd. De resulterende graad is gelijk aan het aantal punten min één.

❓ **Wanneer is het gebruik van een interpolerende veelterm (niet) rechtvaardig?** >

- Enkel bruikbaar voor relatief lage graden omdat een Vandermonde matrix snel zeer slecht geconditioneerd wordt.
- Het gebruik van de interpolerende veelterm om te extrapoleren is zeer precair.

3.1.2 Kleinste kwadraten veeltermen

❓ **Formuleer de probleemstelling en algemene oplossing. (?)** >

Veelal zullen we m meetpunten (x_i, y_i) , $i = 1, \dots, m$ willen benaderen door een veelterm van graad $d < m - 1$. Dit komt neer op het berekenen van de kleinste kwadraten benadering van het overgedetermineerde stelsel $Ac = y$ met c de coëfficiënten en $A_{ij} = x_i^j$. (?)

❓ **Hoe kies je de optimale graad voor je veeltermbenadering?** >

Het antwoord op die vraag komt uit de statistiek. Men verhoogt de graad zo lang er een beduidende daling te merken is in de variantie

$$\sigma^2 = \frac{\sum e_i^2}{m - d - 1},$$

met $e_i := y_i - y(x_i)$.

Merk op dat $\sum e_i^2$ zal dalen als men d verhoogt, maar omdat men dit door d deelt zal σ^2 niet noodzakelijk mee dalen.

3.2 Niet-veelterm benaderingen

Veel voorkomende verbanden zijn exponentieel van aard, zoals $y = cx^p$ of $y = ce^{ax}$.

② **Hoe kan je (in MATLAB) nagaan of er een exponentieel verband tussen twee variabelen bestaat?** >

Visueel door de gegevens in een grafiek met (semi)logaritmische schaal te plotten.

- In het eerste geval zal een loglog plot het lineair verband $\log y = p(\log x + \log c)$ tussen $\log y$ en $\log x$ onthullen.
- In het tweede geval zal een semilogy plot het lineair verband $\log y = ax + \log c$ tussen $\log y$ en x onthullen.

② **Hoe kan je dergelijke verbanden eenvoudig proberen benaderen en waar moet je voor opletten?** >

Stel $z = \log y$ en zoek een lineaire benadering voor resp. $\log x$ of x . Men dient er zich wel bewust van te zijn dat men nu afwijkingen op $\log y$ minimaliseert.

HS4: Beginwaardeproblemen voor gewone differentiaalvergelijkingen

4.2 Het algemene kader

Een differentiaalvergelijking numeriek oplossen kan [of moet(?), cf. 4.5] beginnen bij het benaderen van de afgeleide.

4.2.1 Het benaderen van afgeleiden

$$y'(x) = \lim_{h \rightarrow 0} \frac{y(x+h) - y(x)}{h}$$

② **Waar hangt de nauwkeurigheid van het benaderen van deze afgeleide van af? Wat zijn de implicaties?** >

- De fout in de benadering daalt als h daalt.
- De fout ten gevolge van het aftrekken van twee bijna gelijke getallen stijgt als h daalt.

De fout zal dus dalen zolang het eerste fenomeen domineert. Gelukkig moeten we niet voor elk probleem de optimale h bepalen. We hebben niet altijd de best mogelijke nauwkeurigheid nodig, en dus worden meestal technieken gebruikt die automatisch de juiste staplengte kiezen opdat het resultaat de vereiste nauwkeurigheid zou hebben.

Een andere overweging is dat een kleinere staplengte meer stappen impliceert, wat niet altijd wenselijk is.

4.2.2 Voorwaartse Euler

We beschouwen een eerste orde beginwaardeprobleem

$$\begin{cases} \dot{y}(x) = f(y, t) \\ y(0) = y_0, \end{cases} \quad (x, y) \in (a, b) \times \mathbb{R}^n$$

met f en y_0 gegeven.

🔗 Beschrijf de methode. >

Benader $y(t)$ op discrete tijdstippen t_n met als staplengte $t_{n+1} - t_n = h \in \mathbb{F}_0^+$ door

$$\begin{aligned} y(t_{n+1}) &\approx y_{n+1} = y_n + (t_{n+1} - t_n) \frac{y_{n+1} - y_n}{t_{n+1} - t_n} \\ &= y_n + hf_n, \end{aligned}$$

waar $f_n := f(y_n, t_n)$.

🔗 Bespreek de orde van de methode. >

De *orde* is de grootteorde van h in de fout op de benadering. Hoe hoger hoe meer een kleinere staplengte zal lonen.

We kunnen dit illustreren (intuïtie kweken want dit is moeilijk te veralgemenen) a.d.h.v. het (lineaire) modelvoorbeeld $\dot{y} = \alpha y$. De methode geeft $y_{n+1} = y_n + hf_n = (1 + h\alpha) y_n$, wat overeenkomt met linearisatie, zoals blijkt uit de Taylor-reeks van de exacte oplossing:

$$\begin{aligned} y(t_{n+1}) &= e^{\alpha(t_n+h)} = e^{\alpha h} y(t_n) \\ &= \left(1 + \alpha h + \frac{(\alpha h)^2}{2} + O(h^3) \right) y(t_n). \end{aligned}$$

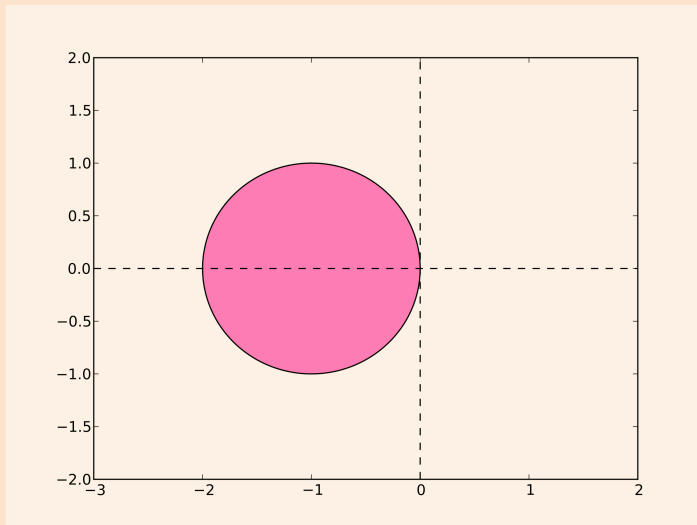
De *lokale fout* (de fout gemaakt in elke stap afzonderlijk, dus in de veronderstelling dat $y_n = y(t_n)$) is

$$y_{n+1} - y(t_{n+1}) = O(h^2).$$

Dit betekent dat als we h halveren de fout in elk stapje door 4 gedeeld wordt. De *globale fout* zal echter ook slechts gehalveerd worden, omdat we nu twee maal zoveel stappen doen!

🔗 Wat is de lineaire stabiliteitsregio van de methode? Wat gebeurt er precies als $\alpha \in \mathbb{R}_0^-$? >

Uit $y_{n+1} = (1 + h\alpha) y_n$ volgt dat de benadering niet zoals de exacte oplossing naar nul convergeert wanneer $h\alpha \notin \{z \in \mathbb{C} \mid |1 + z| \leq 1\}$. Bijgevolg is de lineaire stabiliteitsregio het roze gebied in onderstaande afbeelding:



Bron: https://en.wikipedia.org/wiki/Euler_method, 11 juni 2024

Als $\alpha \in \mathbb{R}_0^-$:

- In het algemeen neemt de (lokale) fout toe naarmate $|\alpha h|$ toeneemt. (?) p.58
- Als $\alpha h < 1$ dan zal de benadering oscilleren.
- Als $\alpha h < 2$ dan neemt de benadering zelfs toe in absolute waarde, in tegenstelling tot de exacte oplossing.

4.5 Diverse methodes

Beschouw opnieuw $(\hat{} \hat{}) (\hat{} \hat{}) \hat{} \hat{} (\hat{} \hat{})$ (tja kon niet kiezen...). We kunnen de oplossing ook benaderen door te kijken naar een integraal. De exacte oplossing wordt gegeven door

$$y(t_{n+1}) = y(t_n) + \int_{t_n}^{t_{n+1}} f(y, t) \, dt.$$

4.5.1, 4.5.2 Euler

Beschouw opnieuw $(\text{♠} \text{♣}) (\text{♠}^{\circ} \text{♣}^{\circ}) \text{♣} \heartsuit (\text{♣}^{\circ}) \spadesuit$.

② Beschrijf hoe we hiermee zowel de voorwaartse als *achterwaartse* Euler methodes kunnen bekomen.

Benaderen we de integraal door een *linker rechthoeksregel*

$$\int_{t_n}^{t_{n+1}} f(y, t) dt \approx (t_{n+1} - t_n) f_n,$$

dan bekomen we met $h = t_{n+1} - t_n$ terug de voorwaartse Euler.

Benaderen we de integraal echter door een *rechter rechthoeksregel*

$$\int_{t_n}^{t_{n+1}} f(y, t) dt \approx (t_{n+1} - t_n) f_{n+1},$$

dan bekomen we met $h = t_{n+1} - t_n$ de zogenaamde **achterwaartse Euler** methode:

$$y_{n+1} = y_n + h f_{n+1}.$$

🔗 Bespreek de orde van de achterwaartse Euler methode. >

Als we deze methode toepassen op de testvergelijking $\dot{y} = \alpha y$, krijgen we

$$\begin{aligned} y_{n+1} &= y_n + h \alpha y_{n+1} \\ &= \frac{1}{1 - \alpha h} y_n \\ &= (1 + \alpha h + (\alpha h)^2 + O(h^3)) y_n. \end{aligned}$$

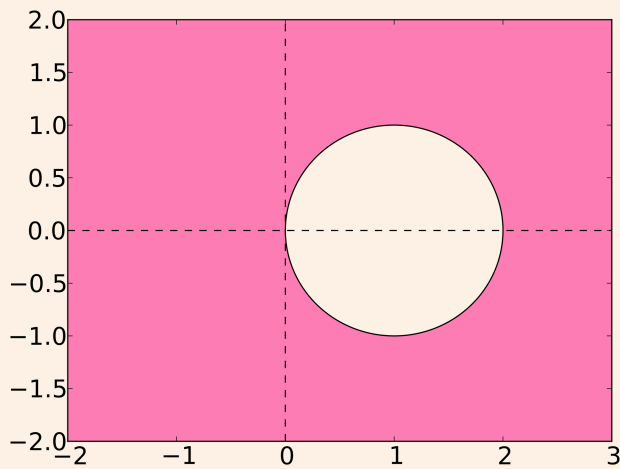
De lokale fout is

$$y_{n+1} - y(t_{n+1}) = O(h^2).$$

Dezelfde redenering als voor de voorwaartse Euler leert ons dat de globale fout bijgevolg ook hier lineaire orde heeft.

🔗 Wat is de lineaire stabiliteitsregio van de methode? Wat zegt dit over deze methode als $\alpha \in \mathbb{R}_0^-$? >

Uit $y_{n+1} = \frac{1}{1 - h\alpha} y_n$ volgt dat de benadering niet zoals de exacte oplossing naar nul convergeert wanneer $h\alpha \notin \{z \in \mathbb{C} \mid |1 - z| \geq 1\}$. Bijgevolg is de (lineaire) *stabiliteitsregio* het roze gebied in onderstaande afbeelding:



Bron: https://en.wikipedia.org/wiki/Backward_Euler_method, 11 juni 2024

Bijgevolg is de achterwaartse Euler methode wat *absoluut stabiel* wordt genoemd.

DEFINITIE: absoluut stabiele methode

Numerieke methode die ook asymptotisch absoluut *daalt* als de exacte oplossing dit doet. (Dit impliceert niet dat ander asymptotisch gedrag wordt behouden!)

In dit specifieke voorbeeld betekent dit dat $R(\alpha) < 0 \Rightarrow \lim_{n \rightarrow \infty} |y_n| = 0$.

Welke voorbeelden zijn in de cursus besproken waarvoor deze methodes problematisch zijn? Hoe komt dit? (?)

- Simulaties van systemen met energiebehoud: zie 4.4. De energie zal respectievelijk onder- en overschat worden.
- Voorwaartse Euler ook niet voor stijve stelsels (zie 4.6) omdat het een expliciete methode is.

4.5.3 Trapeziumregel

Beschouw opnieuw $(\hat{\sim} \hat{\sim}) (\hat{\sim} \hat{\sim}) \hat{\sim} \hat{\sim}$.

Beschrijf de methode. Wat is de orde? Vergelijk met de Euler methodes. (?)

De integraal wordt benaderd door het gemiddelde van de voorwaartse en achterwaartse Euler benaderingen:

$$\int_{t_n}^{t_{n+1}} f(y, t) dt \approx h \frac{f_n + f_{n+1}}{2}.$$

Deze methode heeft een lokale fout $O(h^3)$ en een globale fout $O(h^2)$ en is dus nauwkeuriger dan zowel voorwaartse als achterwaartse Euler. Ze is bovendien ook absoluut stabiel.

In het voorbeeld van de slinger overschat de voorwaartse Euler methode de energie, terwijl de achterwaartse ze onderschat, met als gevolg dat deze toeneemt resp. afneemt over de tijd. De trapeziumregel houdt de energie niet perfect constant, maar restaureert ze iedere periode wel. [veralgemening(?)]

4.5.4 Runge-Kutta methodes

🔍 Wat zijn de voor- en nadelen van deze methodes? (?) >

Voordelen:

- Hogere orde dan de Euler methodes, en in veel gevallen zelfs dan de trapeziummethode.
- Vaak expliciete methodes. [kleinere kost(?)]

Nadelen:

- (Vaak) niet absoluut stabiel.
- Energie ook niet goed behouden [of hoort dat bij vorige punt(?)].

4.5.5 Classificatie van methodes

🔍 Classificeer de voorwaartse en achterwaartse Euler methodes en vergelijk ze. (?) >

📖 DEFINITIE: k -steps (impliciete/expliciete) methode

Methode die beschreven wordt door

$$\sum_{i=0}^k a_i y_{n+1-(k-i)} = h \sum_{i=0}^k b_i f_{n+1-(k-i)},$$

met $a_k \neq 0$ en $a_0 \vee b_0 \neq 0$. (In de praktijk zal men beginnen met een 1-steps methode, dan een stap met een 2-steps methode doen, enz.)

Indien $b_k = 0$ spreekt men van een *expliciete methode*, omdat er dan in het rechterlid enkel bekende termen staan (en in het linkerlid één onbekende term,

$a_k y_{n+1}$). Anders van een *impliciete methode*.

① DEFINITIE: impliciete/expliciete methode (?)

Bij **expliciete methodes** kent men een toekomstige toestand van een systeem [onmiddellijk(?)] expliciet in termen van gekende toestanden (en moeten we dus enkel voorwaarts rekenen). **Impliciete methodes** daarentegen vergen het oplossen van een vergelijking die [initieel(?)] onbekende toestanden in beide leden bevat.

Wiskundig, als $Y(t)$ de huidige toestand is, en $Y(t + h)$ de toestand een beetje later, dan is voor expliciete methodes

$$Y(t + h) = F(Y(t)),$$

terwijl men voor impliciete methodes *minstens* één extra berekening nodig heeft (om als dat kan $Y(t + h)$ naar één kant te brengen) om de vergelijking

$$G(Y(t), Y(t + h)) = 0$$

op te lossen naar $Y(t + h)$.

Bron: https://en.wikipedia.org/wiki/Explicit_and_implicit_methods, 12 juni 2024

Voorwaartse Euler:

- $a_0 = -1, a_1 = 1, b_0 = 1, b_1 = 0 \Rightarrow$ expliciet

Achterwaartse Euler:

- $a_0 = -1, a_1 = 1, b_0 = 0, b_1 = 1 \Rightarrow$ impliciet

Beide 1-steps omdat $a_1 \neq 0$ terwijl $a_i = 0$ voor $i > 1$, en omdat $a_0 \vee b_0 \neq 0$.

Vergelijking: (?)

- De achterwaartse Euler benadering zal steeds positief zijn als de exacte oplossing positief is. p.66(?)
- Impliciete methodes zijn meestal duurder, hoewel niet altijd (zie bv. sectie 4.6).

4.6 Stijfheid

② Wat zijn *stijve differentiaalvergelijkingen*? Welk type methode is er het meest gepast voor, en waarom? (?)

>

Deze kunnen een heel zacht verlopende oplossing hebben.

Natuurlijk zijn hogere orde methodes voordelig omdat deze de globale fout verminderen. (?) Daarenboven—omdat ze een zacht verloop hebben—is de staplengte is hier veelal beperkt door stabiliteitsbeperkingen, en niet door de lokale fout, waardoor een grotere staplengte wenselijk is teneinde de kost (en duur) van de berekeningen te verminderen. Aangezien impliciete methodes vaak absoluut stabiel zijn (terwijl expliciete methodes dit nooit zijn) en dus [vaak(?)] een grotere staplengte toelaten, zijn deze hier het meest geschikt.

HS5: Oplossen van niet-lineaire vergelijkingen

5.1 Inleiding

Probleemstelling: Gegeven een functie $f : D \subseteq \mathbb{R} \rightarrow \mathbb{R}$, zoek alle $x^* \in D$ waarvoor $f(x^*) = 0$.

DEFINITIES: enkelvoudig en m-voudig nulpunt >

- **enkelvoudig nulpunt:** $x^* \in \mathbb{R}$ waarvoor $f(x^*) = 0 \neq f'(x^*)$.
- **m-voudig nulpunt:** $x^* \in \mathbb{R}$ waarvoor $f(x^*) = f'(x^*) = \dots = f^{(m-1)}(x^*) = 0 \neq f^{(m)}(x^*)$.

5.2 Conditie van een nulpunt

Wat is het absolute conditiegetal van een enkelvoudig nulpunt? Wat betekent dit? Wat zegt het over meervoudige nulpunten? >

DEFINITIE: absolute conditiegetal van een enkelvoudig nulpunt

$$\frac{1}{|f'(x^*)|}$$

Motivering >

We brengen een kleine perturbatie $\varepsilon g(x)$ aan op f , waardoor het gezochte nulpunt x^* ook verstoord is:

$$f(x^*(\varepsilon)) + \varepsilon g(x^*(\varepsilon)) = 0.$$

De fout op het nulpunt is $\delta x^* = x^*(\varepsilon) - x^*$, wat een Taylor-expansie van $x^*(\varepsilon)$ rond 0 suggereert. Voor een linearisatie van de fout hebben we niet meer nodig dan de eerste afgeleide $\frac{dx^*}{d\varepsilon}$, welke we vinden in de differentiatie van

de vorige vergelijking naar ε :

$$f'(x^*(\varepsilon)) \frac{dx^*}{d\varepsilon} + g(x^*(\varepsilon)) + \varepsilon g'(x^*(\varepsilon)) \frac{dx^*}{d\varepsilon} = 0.$$

In $\varepsilon = 0$ bekomen we zo

$$\frac{dx^*}{d\varepsilon} = -\frac{g(x^*)}{f'(x^*)}$$

zodat de gezochte Taylor-reeks van de vorm

$$x^*(\varepsilon) = x^* - \varepsilon \frac{g(x^*)}{f'(x^*)} + O(\varepsilon^2)$$

is. De linearisatie van de fout wegens de perturbatie is dan

$$\delta x^* = -\frac{1}{f'(x^*)} \delta f(x^*)$$

en we bekomen als conditiegetal

$$\left| \frac{\delta x^*}{\delta f(x^*)} \right| = \frac{1}{|f'(x^*)|}.$$

De hoek waarmee $f(x)$ de x -as snijdt bepaalt dus de conditie van het probleem. Hoe horizontaler, hoe slechter de conditie.

Hoe hoger de meervoudigheid, hoe beter de functie aansluit bij de x -as in de omgeving van x^* , en dus hoe slechter de conditie zal zijn. Een voorbeeld waarbij dit problemen kan veroorzaken is $\sin x - x = 0$.

5.3 Bisectiemethode (Intervalherleiding)

🔍 Beschrijf de methode. >

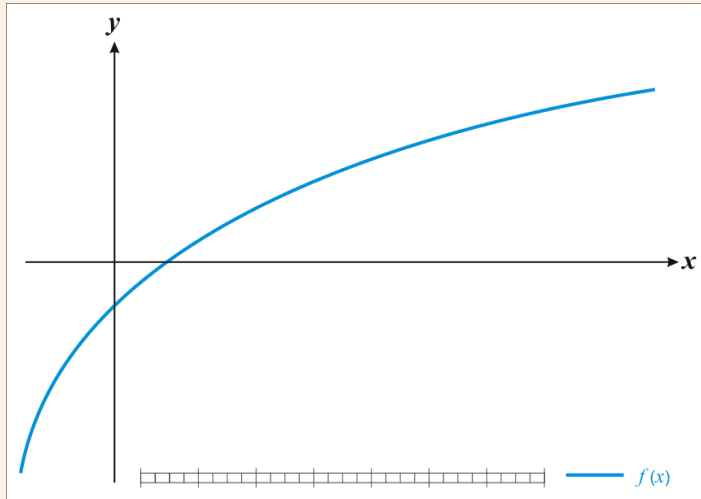
Stel dat f continu is op $[a, b] \subset \mathbb{R}$ en dat $f(a)f(b) < 0$. Dan volgt hieruit dat er een (oneven aantal) nulpunt(en) in (a, b) moet liggen.

De bisectiemethode deelt het interval op in twee gelijke delen $[a, c]$ en $[c, b]$, met $c = \frac{a+b}{2}$, en gaat eventueel verder op zoek in het deelinterval waarvan f op de eindpunten van teken verschilt. Het herhaalt dit proces een begrensd aantal keer tot het middelpunt goed genoeg een nulpunt benadert. Twee mogelijke stopcriteria zijn:

- $|f(c_n)| < \text{een gegeven nauwkeurigheid}$.
- $|a_n - c_n| < \text{een gegeven nauwkeurigheid}$. Men kan hiermee op voorhand het

benodigd aantal stappen berekenen.

Deze methode is zeer *robuust* want in elke stap wordt het onzekerheidsinterval gehalveerd. Ze heeft *lineaire convergentie*.



Bron: <https://de.wikipedia.org/wiki/Bisektion>, 14 juni 2024

5.4 Methodes gebaseerd op lineaire interpolatie

② **Wat doen deze anders dan de bisectiemethode? Welke voor- en nadelen heeft dat? Beschrijf de drie voorbeelden uit de cursus en geef hun convergentieordes. (?)** >

De bisectiemethode maakt enkel gebruik van het teken van $f(x)$. Indien dat alle beschikbare informatie is, is dat de snelste manier om een oplossing te vinden. Men kan echter snellere convergentie verwachten als men meer informatie gebruikt, maar convergentie is daarvoor niet altijd gegarandeerd zoals bij de bisectiemethode. Bijgevolg (?) kan de kost ook groter zijn (zeker voor het berekenen van afgeleiden voor NR) (?).

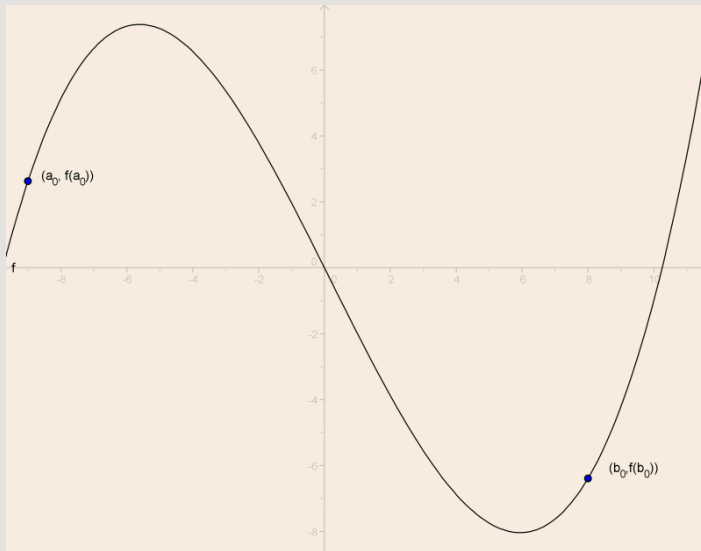
Drie voorbeelden waarin we een opeenvolging van wortels van linearisaties van het probleem gebruiken om een wortel te benaderen:

③ **Lineaire interpolatie (regula falsi)** >

Stel $x_0 < x_1$ met $f(x_0)f(x_1) < 0$. Interpoleer een rechte door $(x_0, f(x_0))$ en $(x_1, f(x_1))$ en zoek hiervan de wortel x_2 . Als $f(x_0)f(x_2) < 0$ ga dan verder met $[x_0, x_2]$ en anders met $[x_2, x_1]$, en herhaal met dit deelinterval het beschreven proces tot een gegeven criterium is bereikt.

Deze methode heeft *lineaire* convergentie, en is zeer robuust (in elke stap wordt het zoekgebied verkleind, zodat convergentie gegarandeerd is), zoals de

bisectiemethode. Er is wel geen garantie van snellere convergentie daardan [lol, nieuw woordje].

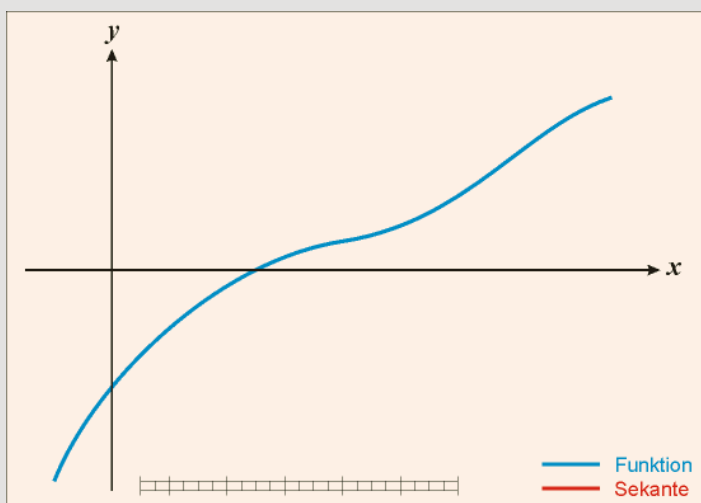


Bron: https://nl.wikipedia.org/wiki/Regula_falsi, 11 juni 2024

Secant methode >

Ongeveer hetzelfde als de lineaire interpolatie, maar waar telkens het vorige nulpunt samen met het huidige nulpunt als nieuwe interval worden gebruikt. Bijgevolg kan het dat $f(x_k)f(x_{k+1}) > 0$, wat een *lineaire extrapolatie* stap wordt genoemd.

Lineaire extrapolatie stappen ontnemen de garantie op convergentie. Als deze methode convergeert naar een enkelvoudig nulpunt dan doet ze dit *superlineair* (orde ~ 1.62).

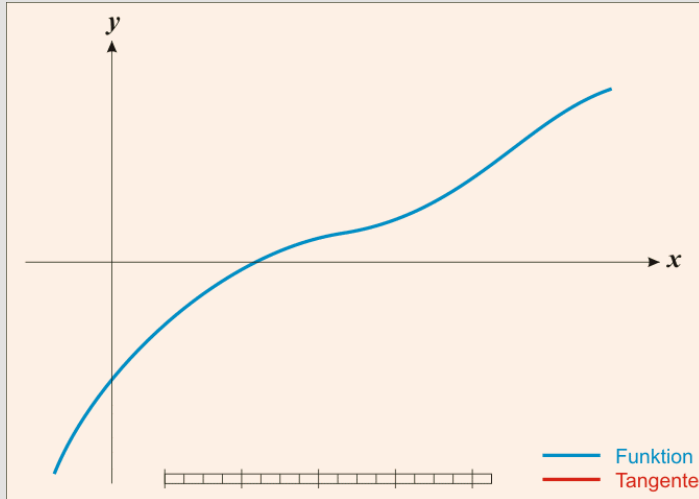


Bron: <https://nl.wikipedia.org/wiki/Secant-methode>, 11 juni 2024

Newton-Rhapson >

Neem voor x_{k+1} de wortel van de raaklijn aan $(x_k, f(x_k))$.

Ook hier is convergentie niet gegarandeerd. Als deze methode convergeert naar een enkelvoudig nulpunt dan heeft ze *kwadratische* convergentie.



Bron: https://nl.wikipedia.org/wiki/Methode_van_Newton-Raphson, 11 juni 2024

HS6: Oplossen van partiële differentiaalvergelijkingen

Beschouw de temperatuursverdeling $u(x, t)$ in een staaf van lengte 1. Het eenvoudigste model bestaat uit de

DEFINITIE: 1-dimensionale warmte/diffusievergelijking

$$\frac{\partial u}{\partial t} = \frac{d^2 u}{dx^2}.$$

Neem voor beginvoorwaarde

$$u(x, 0) = g(x), \quad 0 \leq x \leq 1$$

en voor (Dirichlet) randvoorwaarden

$$u(0, t) = a(t), \quad u(1, t) = b(t), \quad 0 \leq t.$$

We beperken ons tot methodes gebaseerd op eindige differenties.

6.4 Een expliciete methode

② Leg uit hoe een *discretisatie* werkt en wanneer het een expliciete methode is. (?) >

We zoeken $u(x, t)$ op een beperkt aantal plaatsen x_i en tijdstippen t_i . De eenvoudigste discretisatie neemt gelijk verspreide punten op het ruimte- en tijdsdomein.

Zoals eerder gedefinieerd is het een expliciete methode als we de waarden op tijdstip $j + 1$ expliciet in functie van de waarden op tijdstip j hebben (we moeten dus enkel voorwaarts rekenen):

$$v_{i,j} = F(v_{1,j-1}, \dots, v_{n,j-1}).$$

② Bespreek de stabiliteit van deze methode. (?) >

We beschouwen opnieuw een modelvoorbeeld, het speciale geval met $a(t) = b(t) = 0$. Dan zal $u(x, t \rightarrow \infty) = 0$. Een methode is absoluut stabiel als de benadering asymptotisch absoluut daalt als de exacte oplossing dat doet.

De expliciete methode kan geschreven worden als

$$V_j = AV_{j-1},$$

met $v_{i,j}$ de temperatuur op het roosterpunt (x_i, t_j) . Dit impliceert dat

$$V_j = \dots = A^j V_0,$$

waarvoor dus moet gelden dat

$$\lim_{j \rightarrow \infty} A^j V = 0$$

voor alle beginvoorwaarden V . Dit gebeurt uitzonderlijk als voor de **spectraalradius** $\rho(A) := \|A\|_2 = \max \{|\lambda| \mid \lambda \in \text{Spec}(A)\}$ geldt dat

$$\rho(A) < 1.$$

Men kan dit eenvoudig aantonen voor diagonaliseerbare [was dit niet zo(?)] $A = P\Lambda P^{-1}$. Dan is namelijk

$$\lim_{j \rightarrow \infty} A^j = \lim_{j \rightarrow \infty} P\Lambda^j P^{-1} = 0$$

uitzonderlijk als alle eigenwaarden naar 0 gaan.

6.5 Een impliciete methode

② Leg uit wanneer een discretisatie een impliciete methode is, en hoe dit concreet gedaan wordt. (?) >

Door achterwaartse differentie te gebruiken bekomen we een recursieformule die meer onbekenden dan bekenden bevat. Dit is dan een impliciete methode. We kunnen dit oplossen door voor elke stap een stelsel lineaire vergelijkingen op te lossen, wat men kan doen met LU-decompositie.

🔗 Bespreek de stabiliteit van deze methode. (?) >

We beschouwen opnieuw het speciale geval met $a(t) = b(t) = 0$, zodat opnieuw $u(x, t \rightarrow \infty) = 0$. We kunnen de oplossing schrijven als

$$V_j = A^{-1}V_{j-1} = \dots = A^{-j}V_0.$$

Dezelfde redenering als bij de expliciete methode leert ons dat het algoritme absoluut stabiel is als

$$\rho(A^{-1}) = (\rho(A))^{-1} < 1.$$

HS7: Toevalsgeneratoren

7.2 Willekeurige getallen

7.2.1 Definitie en eigenschappen

We beschouwen in dit deel trekkingen $R_i \sim \mathcal{U}(0, 1)$ met als wdf $f(x) = \mathbb{1}_{x \in [0, 1]}$. Hieruit volgt dat $E[R_i] = \frac{1}{2}$ en $\text{Var}[R_i] = \frac{1}{12}$.

🔗 Welke twee eigenschappen moet een rij willekeurige getallen uit de continue uniforme verdeling hebben? >

- **Uniformiteit:**

Als het interval $[0, 1]$ opgesplitst wordt in n deelintervallen van gelijke lengte, dan is het verwachte aantal geobserveerde punten in elk deelinterval gelijk aan N/n , waarbij N het totaal aantal observaties is.

- **Onafhankelijkheid:**

De kans om een punt waar te nemen in een bepaald interval is onafhankelijk van de eerder waargenomen punten.

7.2.2 Genereren van willekeurige getallen

📄 3 types generatoren >

- Tabellen

- Hardware-generatoren (zoals een dobbelsteen of instrumenten die gebruik maken van thermische witte ruis of radioactief verval)
- Software-generatoren, ook *toevalsgeneratoren* genoemd

DEFINITIE: pseudo-willekeurige getallen >

Getallen die de uitvoer zijn van een software-generator voor willekeurige getallen.

Aan welke criteria moet een toevalsgenerator zo goed mogelijk voldoen? >

- De *statistische eigenschappen* van echt willekeurige getallen nabootsen.
- *Lange periode* hebben.
- *Reproduceerbare* uitvoer geven (geïnitieerd kunnen worden).
- *Draagbaar* zijn (dezelfde getallen produceren op verschillende computers en in verschillende talen).
- *Snel en economisch* werken.

Geef enkele voorbeelden van hoe een rij pseudo-willekeurige getallen kan afwijken van een ideale rij willekeurige getallen? (?) >

- Niet uniform verdeeld zijn.
- Afhangelijkheid tussen de getallen hebben.
- Een gemiddelde hebben dat te ver afwijkt van het ideaal.
- Een variantie hebben die te ver afwijkt van het ideaal. (Een constant gemiddelde kan niet, want dan zou men altijd het gemiddelde moeten trekken.)
- Discreet i.p.v. continu verdeeld zijn. [Dit is sowieso een probleem, omdat men een eindige voorstelling heeft. Bovendien liggen niet alle voorstelbare getallen even ver uit elkaar, en wil ze dus zelfs niet allemaal gebruiken. Dit betekent natuurlijk ook dat de periode altijd eindig zal zijn.] (?)

7.2.3 Toevalsgeneratoren

DEFINITIE: lineaire congruentiële generatoren (LCG) >

1. Kies een grote $m \in \mathbb{N}$.
2. Telkens een willekeurig getal $U \sim \mathcal{U}[0, 1]$ nodig is trekt men willekeurig een $X \in \{1, \dots, m-1\}$ (alsof uit een urne) en stelt men $U = X/m$.

Opmerkingen:

- In principe moet een getrokken getal terug in de urne gestoken worden. In de praktijk doet men dit niet. Gelukkig maakt dit weinig uit als m groot is.
- Merk op dat men hiermee 0 en 1 niet kan genereren, dit is belangrijk voor het vermijden van enkele problemen.
- Men kan het resultaat van verschillende LCG's combineren om de statistische eigenschappen te verbeteren.

📌 DEFINITIE: lineaire congruentiële methode (LCM) >

Men produceert getallen tussen 1 en $m - 1$ volgens de recursiebetrekking

$$X_{i+1} = [(aX_i + c) \bmod m], \quad i = 0, 1, 2, \dots$$

en deelt deze daarna door m .

De startwaarde X_0 noemt men de *kiem*, a de *vermenigvuldiger*, c het *increment*, en m de *modulus*.

🔗 Geef enkele implementatieproblemen van LCG's en de LCM? >

- Het getal $a(m - 1)$ moet exact voorgesteld kunnen worden om de recursiebetrekking voor de LCM direct te kunnen implementeren.
- Willen we gegenereerde getallen x_1, x_2, \dots combineren tot n -dimensionale punten $(x_1, \dots, x_n), (x_{n+1}, \dots, x_{2n}), \dots$ [correct(?)], dan zullen deze echter altijd in een beperkt aantal $(n - 1)$ -dimensionale vlakken vallen, en dat kunnen er heel weinig zijn.

7.3 Testen voor toevalsgeneratoren

🔗 Welke statistische testen worden gebruikt om na te gaan of een rij pseudo-willekeurige getallen de nodige statistische eigenschappen heeft? Waarvoor moet je met deze testen opletten? (?) >

- **Frequentietests:**
Gebruik van de Kolmogorov-Smirnov-test of de chi-kwadraat-test om de verdeling van de gegenereerde getallen te vergelijken met de uniforme verdeling.
Problemen(?)
- **Autocorrelatietest:**
Ga na of de correlatie tussen gegenereerde punten zoals verwacht gelijk is aan 0. Men kijkt daarbij niet enkel naar opeenvolgende getallen. Het probleem met

deze test is dat hoe langer je zoekt hoe groter de kans wordt dat je een correlatie vindt, ookal is ze insignificant.

7.4 Toevalsgeneratoren voor discrete verdelingen

② Hoe kunnen we discrete willekeurige variabelen verdeeld volgens eender welke pdf genereren, uitgaande van uniform verdeelde willekeurige getallen? >

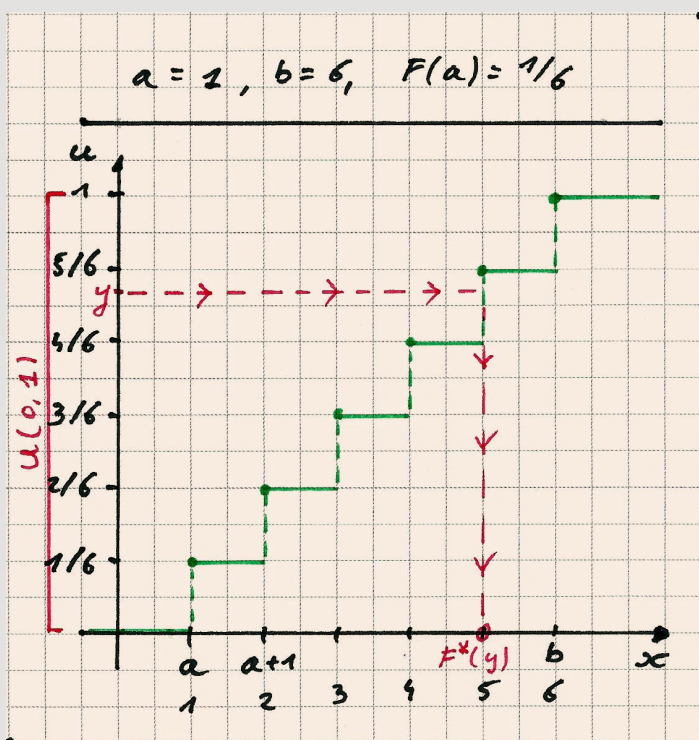
Aan de hand van volgende definitie en stelling:

① **DEFINITIE:** inverse distributiefunctie (idf) van een discrete verdeling

Stel dat X een discrete willekeurige variabele is met cdf F . De **idf** van X is

$$F^* : (0, 1) \rightarrow \Omega : u \mapsto F^*(u) = \min\{x \in \Omega \mid u < F(x)\}.$$

Bijvoorbeeld voor een dobbelsteen:



① **STELLING**

Stel dat X een discrete willekeurige variabele is met idf F^* , en $U \sim \mathcal{U}(0, 1)$. Zij $Z := F^*(U)$ dan zijn Z en X identiek verdeeld.

7.5 Toevalsgeneratoren voor continue verdelingen

- ② Hoe kunnen we continue willekeurige variabelen verdeeld volgens eender welke pdf genereren, uitgaande van uniform verdeelde willekeurige getallen? >

Aan de hand van volgende definitie en stelling:

① **DEFINITIE: inverse distributiefunctie (idf) van een continue verdeling**

Stel dat X een continue willekeurige variabele is met inverteerbare cdf F . De idf van X is de inverse functie F^{-1} van F .

① **STELLING**

Stel dat X een continue willekeurige variabele is met idf F^{-1} , en $U \sim \mathcal{U}(0, 1)$. Zij $Z := F^{-1}(U)$ dan zijn Z en X identiek verdeeld.

Het continue geval lijkt eenvoudiger dan het discrete omdat men niet met ongelijkheden moet werken om de idf op te stellen. Anderzijds is het niet altijd mogelijk om F^{-1} eenvoudig uit te drukken.

- ② Beschrijf de acceptance-rejection methode. Geef de grote voor- en nadelen en bespreek. >

Dit is een techniek om niet-uniforme stochastische variabelen te genereren zonder gebruik te maken van de cdf of idf.

① **DEFINITIE: acceptance-rejection methode**

Gegeven een wdf $p(x)$. Stel dat we een functie $q(x)$ kennen die voldoet aan $q(x) \geq p(x)$ voor alle x , en dat we een manier hebben om steekproeven te nemen volgens de wdf

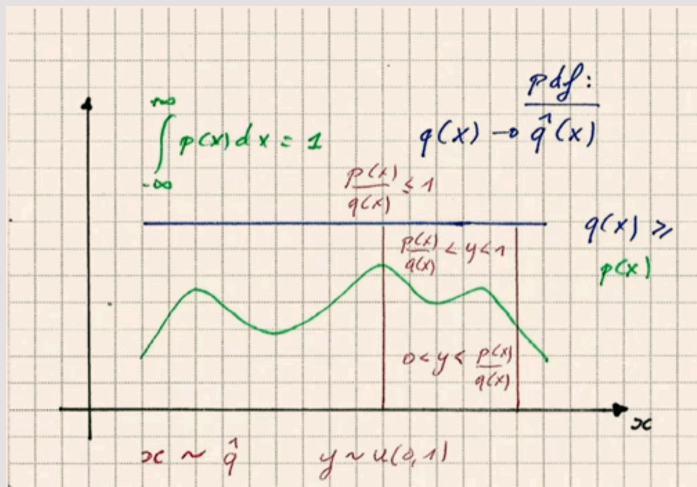
$$\hat{q}(x) = \frac{q(x)}{\int_{\mathbb{R}} q(t) dt}.$$

Aangezien het nodig is om een generator voor \hat{q} -verdeelde variabelen ter beschikking te hebben neemt men in de praktijk voor $q(x)$ meestal een constante functie op een eindig interval.

Neem twee willekeurige getallen: $x \sim \hat{q}$, en $y \sim \mathcal{U}(0, 1)$. Het acceptance-rejection algoritme steunt nu op de volgende beslissing:

- **accept** x indien $0 < y < \frac{p(x)}{q(x)}$,
- **reject** x indien $\frac{p(x)}{q(x)} < y < 1$.

De punten x die aanvaard worden zijn dan volgens de wdf $p(x)$ verdeeld.



- Het grote voordeel is dat men geen idf nodig heeft.
- Het grote nadeel is dat er soms veel pogingen nodig zijn vooraleer een punt aanvaard wordt, omdat er minstens twee willekeurige getallen x en y gegenereerd moeten worden om één p -verdeeld getal te bekomen. Daarom is het best van een q te gebruiken die zo nauw mogelijk aansluit bij p (bijvoorbeeld $q = \max p(\mathbb{R})$), dan zullen we minder weggooien.

HS8: Monte Carlo simulaties

Tot nu toe beschouwden we systemen die volledig deterministisch zijn. In dit hoofdstuk beschouwen we systemen waarvan het gedrag stochastisch is. Een typisch voorbeeld is diffusie.

8.4 Monte Carlo integratie

② Geef twee soorten problemen die zich dienen aan simulatie met willekeurige getallen.



- Het beschrijven van veel-deeltjessystemen (zoals de diffusie van room in koffie).
- Het bepalen van een verwachte waarde (zoals de verhouding willekeurige punten die binnen een cirkel landen, t.o.v. erbuiten, op een uniform verdeeld vierkant dat de cirkel omhelst).

Beide hebben eigenlijk te maken met informatie over het gemiddelde gedrag van een groot aantal monsters 🧟 🧟 🧟 🧟 🧟 🧟 🧟 🧟 🧟 .

🔗 Wat is de *klassieke Monte Carlo methode* en hoe kan ze gebruikt worden om een één-dimensionale integraal te benaderen? (?) >

De gemiddelde waarde van $f(x)$ in het interval $[0, 1]$ is $I = \int_0^1 f(x) dx$. Bijgevolg is

$$I \approx \frac{1}{N} \sum_{i=1}^N f(x_i),$$

met $x_1, \dots, x_N \in [0, 1]$ [uniform verdeelde(?)] steekproeven.

We kunnen dit veralgemenen naar de integraal

$$I = \int_{\mathbb{R}} f(x) p(x) dx$$

waar $p(x)$ een wdf is. Door gebruik te maken van een steekproef verdeeld volgens p kunnen we

$$I \approx \frac{1}{N} \sum_{i=1}^N f(x_i)$$

benaderen.

Deze methode is echter verre van optimaal voor dit soort problemen, en deze voorbeelden dienen louter als introductie.

🔗 Wat is de *convergentieorde* van Monte Carlo integratie? Hoe kan de fout verminderd worden? >

De centrale limietstelling leert ons dat, in Monte Carlo integratie, de fout $\varepsilon \propto \sigma / \sqrt{N}$. Dit betekent dat we 100x meer monsters nodig hebben om een 10x accurater resultaat te krijgen. Een dergelijke convergentieorde is typisch bij Monte Carlo methodes, wat bekend staat als de " $1/\sqrt{N}$ wet". Het zou dus lonen van deze orde negatiever te krijgen (maar daar gaan we niet verder op in), of de variantie σ^2 te doen dalen.

8.5 Variantiereductie

Beschouw opnieuw

$$I = \int_0^1 f(x) dx.$$

② Wat zijn *primaire* en *secundaire schatters* voor I ? (?) >

In een Monte-Carlo integratie is elke functie-evaluatie

$$f(u)$$

met u getrokken uit $\mathcal{U}[0, 1]$ een schatter voor I , een **primaire schatter** genoemd. De onzekerheid van het resultaat wordt uitgedrukt m.b.v. de variantie

$$\sigma_{\text{prim}}^2 = \int_0^1 (f(u) - I)^2 dx, \text{ welke evident enkel van het probleem afhangt.}$$

De basis (?) van de Monte Carlo methode is deze variantie te reduceren door meer steekproeven te nemen en hun primaire schatters te combineren in **secundaire schatters**

$$\frac{1}{N} \sum_{i=1}^N f(u_i),$$

het gemiddelde van een aantal primaire schatters.

② Beschrijf de drie manieren die we hebben gezien om variantiereductie te verkrijgen. (?) >

① *Stratified sampling.* (?) >

We willen een steekproef uniform willekeurig verspreide punten over een lengte, vlak, of volume, bekomen. Echter, hoe minder punten we nemen, hoe meer gaten en ophopingen we mee zullen achterblijven. Stratificatie is een techniek om deze variantie in de spreiding van punten tegen te gaan.

Neem bijvoorbeeld het interval $\Omega = [0, 1]$. We kunnen dit opsplitsen in M deelintervallen

$$\Omega_k = \left[\frac{k-1}{M}, \frac{k}{M} \right], \quad k = 1, 2, \dots, M$$

van gelijke lengte, en in elk deelgebied N/M punten trekken die verdeeld zijn volgens de wdf

$$p(x) = \begin{cases} M & x \in \Omega_k \\ 0 & x \notin \Omega_k. \end{cases}$$

Het succes van deze werkwijze hangt af van de keuze van de strata. Een steekproef in zo'n stratum wordt verondersteld "representatief te zijn voor de functie voor dat deelgebied" (?). Inzicht in het probleem is dus belangrijk om deze deelgebieden te bepalen.

Importance sampling. >

Hier neemt men punten daar waar het belangrijkst is. Men gaat ze selecteren volgens een wdf p die lijkt op de gegeven functie f . Als de functie bv. ergens een scherp verloop heeft, zal men een wdf nemen die op dezelfde plaats een piek heeft, zodat daar in verhouding meer punten getrokken zullen worden. (Hier is het dus belangrijk om punten volgens willekeurige wdf's te kunnen genereren, zoals besproken in 7.5.)

Hoe meer men weet over f , hoe beter men p kan kiezen en hoe kleiner de fout. De kostprijs per punt stijgt over het algemeen en hangt af van de complexiteit van p .

Control variates. >

Het idee achter deze methode is om een integrand g te gebruiken die gelijkaardig is met f maar waarvoor de exacte waarde van de integraal gekend is. De gezochte integraal wordt dan herschreven als

$$\int_0^1 f(x) \, dx = \int_0^1 (f(x) - g(x)) \, dx + \int_0^1 g(x) \, dx.$$

Vervolgens wordt de Monte Carlo methode gebruikt om de eerste integraal te benaderen.

8.6 Dronkemanswandeling

DEFINITIE: dronkemanswandeling/toevalsbeweging >

Wiskundige formalisering van een traject dat bestaat uit opeenvolgende willekeurige stappen.

Bron: <https://nl.wikipedia.org/wiki/Toevalsbeweging>, 12 juni 2024

Wat is de link met de diffusievergelijking? >

Om inzicht te krijgen in het macroscopisch gedrag van veel-deeltjessystemen proberen we beter niet alle bewegingsvergelijkingen nauwkeurig op te lossen, maar kunnen we wegens het feit van Brownse beweging het gedrag van de deeltjes heel eenvoudig voorstellen als wandelende dronken mannen (wat een feest!). Simulaties maken het dan heel makkelijk om statistische uitspraken te doen over de veranderende verdeling

van de posities van de deeltjes. Op deze manier (door ons op de dichtheid van de deeltjes te concentreren) kunnen we de diffusievergelijking simulatief oplossen.

DEFINITIE: diffusieconstante (van de dronken mannen) >

In alle dimensies stijgt de variantie van de positie lineair met de tijd. De **diffusieconstante** wordt gedefinieerd als de evenredigheidsconstante (van de variantie met de tijd).

Nuttige MATLAB bevelen

Getallen, vectoren, matrices

Een matrix is een vector met vectorcomponenten. Een getal is een vector met één component.

- `j`
Dat gekke ding met $j^2 = -1$.
- `rand`; `randn`
...
- `exp`; `log`; `log2`; `log10`; `sin`; `asin`; `abs`; `max`; `min`; `sum`; `mean`; `length`
Spreken voor zich of even opzoeken.
- `a:h:b`; `colon(a,h,b)`
Rijvector met componenten gaande van a tot en met b in stapjes van h.
`a:b` — Volstaat voor `h=1`.
- `linspace(a, b, k)`
Rijvector met componenten gaande van a tot en met b in k gelijke stappen.
- `zeros(n)`; `zeros(m, n)`; `ones(n)`; `ones(m, n)`
 $n \times n$ resp. $m \times n$ nulmatrix resp. matrix met enen.
- `eye(n)`; `eye(m, n)`
 $n \times n$ resp. $m \times n$ matrix met enen op de hoofddiagonaal en nullen elders.
- `[B1; B2; ...]`; `[B1 B2 ...]`
Voegt de matrices B_i van boven naar onder resp. links naar rechts samen, zolang hun rijen resp. kolommen even lang zijn.
`[]` — Een lege matrix.
- `A(x, y)`
De deelmatrix van A met de elementen op de $x(i)$ 'de rijen en $y(j)$ 'de kolommen. Tellen begint bij 1.
`A(:, y)`; `A(x, :)` — Om alle rijen resp. kolommen mee te nemen.
Merk op dat de `:` notatie verwijst naar de syntax die gelijk verspreide punten genereert, zoals hierboven beschreven.

- $A(x)$ — (met A een matrix en x een vector indices)
De kolomvector met de $x(i)$ 'de elementen, van de kolomvector met alle kolommen van A achtereenvolgens in één kolom geschikt. Tellen begint bij 1.
 $A(x) = v$ — Vervangt dit deel door v (waar v een vector met $n-m+1$ elementen is).
 $A(\text{end})$ — Het laatste element van deze kolom.
 $A(:)$ — Alle kolommen van A achtereenvolgens in één kolom.
 Merk op dat zo $v(:)$ verzekert dat je v als kolomvector gebruikt.
- $A.'; A'$
Transpose resp. Hermitische/toegevoegde transpose van A .
- $\text{flip}(A); \text{fliplr}(A)$
Spiegel A verticaal (of voor een rijvector horizontaal) resp. horizontaal.
- $x.!A$ — (met x een getal, A een $m \times n$ matrix, en $!$ een operatie op getallen)
 $m \times n$ matrix B met $B(i) = x!A(i)$.
 Voor alles behalve delen en exponentiëren kan het punt weggelaten worden.
- $A.!x$ — (met A een $m \times n$ matrix, x een getal, en $!$ een operatie op getallen)
 $m \times n$ matrix B met $B(i) = A(i)!x$.
 Voor alles behalve exponentiëren kan het punt weggelaten worden.
- $A.!B$ — (met A en B $m \times n$ matrices, en $!$ een operatie op getallen)
 $m \times n$ matrix C met $C(i) = A(i)!B(i)$.
- $A * B$ — (met A een $m \times n$ en B een $n \times p$ matrix)
Matrixproduct van A en B .
- $\text{norm}(A)$
2-norm van de vector of matrix A .
 $\text{norm}(A, p)$ — De p -norm van A .
- $\text{det}(A); \text{inv}(A); \text{cond}(A)$
Spreken voor zich.
- $A \setminus b$ — (met A een matrix en b een kolomvector)
Ruwweg $A^{-1}b$ (of b/A if you will) of de kleinste-kwadratenbenadering.
- $V = \text{vander}(v)$
 $n \times n$ matrix met $V(:, j) = v(:).^{(n-j)}$.

Functies en differentiaalvergelijkingen

- $f = @ (v) f(v)$
Stelt een functie op met als invoerargument de vector v , en als uitvoer de vector $f(v)$.
 Tip: Leer de $!.$ notatie goed te gebruiken, dan kan je deze functie meteen in een heel aantal punten evalueren door voor elke component van v een vector mee te geven.
- $\text{polyfit}(x, y, d)$
Rijvector met de coëfficiënten (in dalende orde) van een polynoom van graad d die de punten $(x(i), y(i))$ met de kleinste kwadratenmethode het best benadert.

- `polyval(c, x)`
Vector met de functiewaarden in de punten $x(i)$ van de polynomiaal met coëfficiënten c (in dalende orde).
- `roots(c)`
Alle wortels van de polynomiaal met coëfficiënten c .
- `fminsearch(f, x0)`
Begint bij x_0 om de functie f lokaal te trachten minimaliseren.

Plotten

- `figure`
Maakt een figuur.
- `plot`; `semilogx`; `semilogy`; `loglog`
Verschillende methodes om te plotten.
- `xlabel`; `xticks`; `ylabel`; `yticks`; `title`; `legend`
Opmaak van de figuur.
- `hold on`; `hold off`
Teken verder op de laatst gemaakte figuur of niet.

Algemeen en extra's

- `clear`; `clear all`
Verwijdert alle variabelen uit het geheugen.
- `clc`
Maakt het 'command window' leeg.
- `fprintf('hello')`
Print 'hello' in het command window.
- `x = input('Waarde voor x: ')`
Vraagt een waarde voor x in te geven aan de gebruiker.
- `nargin`
Geeft het aantal invoerargumenten ingevuld door de gebruiker.
- `eps`
Geeft de waarde van het kleinste bewegende kommagetal groter dan 1. Dit is niet de machinenauwkeurigheid!
- `format`
...
- `feval`
...
- `ceil(A)`; `floor(A)`; `round(A)`
De elementen in A afronden naar $+\infty$ resp. $-\infty$, resp. dichtstbijzijnde geheel.

- `real(A)`; `imag(A)`
Reële resp. imaginaire delen van de vector of matrix A.
- `isreal(A)`
1 als de vector of matrix A geen imaginaire componenten heeft. Anders 0.
- `C = {a b ...}`
Cell array met $C\{1\}=a$, $C\{2\}=b$, etc. (zoals een list in python)

Slotwoordje

Ik graaf geen konijnenholen die snel instorten, ik verdiep een hele site stukje bij beetje, telkens na elk beetje onomgekeerde stenen achterlatend. Zo blijf ik graag bezig. - Kjell