NumIntro - Uge 1 Onsdag

Nick Laursen

Københavns Universitet

7. september 2020

1. Program
2. Rettelse
3. C.P 1.3.1
4. C.P 1.3.3 - Teori
5. Lav selv
6. Pseudokode
7. C.P 1.3.5
8. Besvarelse af opgaver

Program

Rettelse af **Opgave 1.3.13 a)**
Gennemgå **C.P 1.3.1**
Gennemgang af teori for **C.P 1.3.3**
Lav selv **C.P 1.3.3**
Lav selv **C.P 1.3.4 (a-b)**
Gennemgang af general pseudokode
Gennemgang af **C.P 1.3.4 (c) Og 1.3.5**
Besvarelse af opgaver som lovet.

Opgave 1.3.13 a)

Vi ser at $x_{n+1} - nx_n = 0$ IKKE er en homogen differens ligning. Ergo løser vi den med substitution.

$$x_{n+1} = nx_n = n \cdot (n-1)\, x_{n-1} = \ldots = \prod_{i=1}^{n} i.$$

$\square$

C.P 1.3.1

Hvorfor opfører vores graf os sådan?

Svar: Computer fejl i dens beregning. Husk **Opgave 1.3.27**

C.P 1.3.3 - Teori

Vi har $4x_{n+2} - 8x_{n+1} + 3x_n = 0$. Lad os vise om den er stabil eller ej. Lad os starte med at finde rødderne på det karakteristiske polynomium. Vi har

$$p(\lambda) = 4\lambda^2 - 8\lambda + 3 = 0 \Rightarrow \lambda \in \left\{\frac{1}{2}, \frac{3}{2}\right\}.$$

Vi bruger **Thm. 3, p.33** til at vise den er stabil. Vi ser at roden $\left|\frac{3}{2}\right| > 1$, så differens ligningen er ikke stabil.

Den generelle løsning bliver

$$z_n = \alpha \left(\frac{1}{2}\right)^n + \beta \left(\frac{3}{2}\right)^n.$$

Lav selv

Lav selv følgende programmering:

C.P 1.3.3
C.P 1.3.4 *a)*
C.P 1.3.4 *b)*

Pseudokode

Algorithm 1 Code for C.P 1.3.4 c) Unit Roundoff Error

procedure $\mathrm{URE}(\quad)$
    $\varepsilon = 1.0$

    while $1.0 + 0.5 \cdot \varepsilon \neq 1.0$ **do**
        $\varepsilon = 0.5 \cdot \varepsilon$
    return ε

C.P 1.3.5 - Teori

Relativ fejl:

$$\varepsilon = \frac{x - x_e}{x}.$$

Vi har $x_{n+2} = \left(\pi + \pi^{-1}\right) x_{n+1} - x_n$, hvor $x_0 = 1$ og $x_1 = \pi$. Lad os beregne det næste par x_i

$$\begin{aligned}
x_2 &= \left(\pi + \pi^{-1}\right) x_1 - x_0 = \pi^2 + 1 - 1 = \pi^2 \\
x_3 &= \left(\pi + \pi^{-1}\right) x_2 - x_1 = \left(\pi + \pi^{-1}\right) \pi^2 - \pi = \pi^3 + \pi - \pi = \pi^3 \\
&\vdots \\
x_{50} &= \pi^{50}.
\end{aligned}$$

Lad os nu begynde at programmere.

C.P 1.3.5 Teori anden del

Nu har vi $x_{n+2} = (\pi + \pi^{-1})\, x_{n+1} - x_n$, hvor $x_0 = 1$ og $x_1 = \pi^{-1}$.
Nu får vi så:

$$
\begin{aligned}
x_2 &= \left(\pi + \pi^{-1}\right) x_1 - x_0 = 1 + \pi^{-2} - 1 = \pi^{-2} \\
x_3 &= \left(\pi + \pi^{-1}\right) x_2 - x_1 = \pi^{-1} + \pi^{-3} - \pi^{-1} = \pi^{-3} \\
&\vdots \\
x_{50} &= \pi^{-50}.
\end{aligned}
$$

Lad os nu programmere igen.

C.P 1.3.1 - Forklaring

Så hvorfor ser vi at vores funktions værdier går mod uendelig? Vi har $1-\sqrt{3}$. En computer kan nemt repræsentere 1 uden fejl. Når det kommer til $\sqrt{3}$, så ser vi at en computer skulle have uendelig hukomelse samt processor kraft for at kunne repæsentere dette tal. Så vores computer approksimere tallet og det giver os en fejl ε. Når vi så har $x_{n+2} = 2\left(x_{n+1} + x_1\right)$, så ser vi at den fejl vi har fra $x_2 = 1 - \sqrt{3}$ vil kommer over til x_3 dog bare større. Da

$$x_3 = 2 \cdot 1 + 2 \cdot \left(1 - \overline{\sqrt{3}} + \varepsilon\right) = 2 + 2 - 2\overline{\sqrt{3}} + 2\varepsilon.$$

Hvis vi forsætter dette vil vi ende med $2^{n-1} \cdot \varepsilon$ fejl fra hvad vi burde ende med.

C.p 1.3.4 - Forklaring

Vi ser at **a)** har samme forklaring som ved **C.P 1.3.1**

For **b)** så ser vi at vi kan beskrive denne udvikling som en bionomal udvikling, i.e

$$\begin{aligned}\left(1-\sqrt{3}\right)^{n-1} &= \sum_{i=0}^{n-1} \frac{(n-1)!}{i!\,((n-1)-i)!} 1^i \left(-\sqrt{3}\right)^{(n-1)-i} \\ &= \sum_{i=0}^{n} \frac{n!}{i!\,(n-i)!} 1^i \left(-\sqrt{3}+\varepsilon\right)^{n-i}.\end{aligned}$$

Vi husker at ε er et lille tal. Så vi vil får i vores ligning hvor hvert led vil have $\{\varepsilon^0, \varepsilon^1, \ldots, \varepsilon^{n-1}\}$. Vi ser at ledet hvor $n = 0$, vil svare til x_0 og det er defineret som "0" i vores tilfælde. Så vi vil faktisk kun have en fejl ved hvert led som er $\{\varepsilon^1, \ldots, \varepsilon^{n-1}\}$. (*Faktisk ikke* ε^1)

C.P 1.3.4 - Forklaring

For **c)** ser vi at $\alpha \neq 0$, da vores computer prøver at beskrive 0 med den bedste approksimation. Da $\alpha \neq 0$ så vil $\alpha \left(1 + \sqrt{3}\right)^n \to \infty$ for $n \to \infty$.

C.P 1.3.5 - Forklaring

For π^{50} så har vi en fejl ligesom i **C.P 1.3.4 b)**. Altså vi vil have en fejl $\{\varepsilon, \varepsilon^2, \ldots, \varepsilon^n\}$. Vi ved at ε er lille og derfor vil $\varepsilon^n \to 0$ for $n \to \infty$. Så vi vil have en mindre fejl for hvert led, men i princppet vil den totale fejl stadig eksistere.

For $\frac{1}{\pi^{50}}$, så kan vi approksimere fejlen for hvert led til at være

$$\{\varepsilon^{-1}, \varepsilon^{-2}, \ldots, \varepsilon^{-n}\}.$$

Derfor vil fejlen blive større og større for hvert led.

ADVARSEL: Denne måde at "regne" på fejl, er et meget simplificeret måde at regne fejl på. Pas på med dette. Der er flere ting som spiller ind i hvorfor der er fejl. Hver gang vi laver en beregning kan der ske ydeligere fejl som enten kan forvære eller forbedre fejlen. Dette er bare gennemtænkt som et simpel svar til hvorfor computere laver fejl, uden at dykke ned i detaljer.