# Matteoblig

Helene Haga Pedersen

8. april 2025

## Numerisk derivasjon og presisjon

1

Vi har utrykket

$$f'(x) \approx \frac{f(x+h) - f(x)}{h}$$

som gjelder så lengde grenseverdien eksisterer. Dersom vi lar $f(x) = e^x$ og $h = 0.1, 0.01, ..., 10^{-20}$. Får vi at dette blir differeansen mellom det analytiske utrykket og det tilnærmede:

| $h$ | Differanse |
|---|---|
| $10^{-1}$ | 0.23174447 |
| $10^{-2}$ | 0.02248333 |
| $10^{-3}$ | 0.00224159 |
| $10^{-4}$ | 0.00022409 |
| $10^{-5}$ | 0.00002241 |
| $10^{-6}$ | 0.00000224 |
| $10^{-7}$ | 0.00000022 |
| $10^{-8}$ | 0.00000001 |
| $10^{-9}$ | 0.00000000 |
| $10^{-10}$ | 0.00000062 |
| $10^{-11}$ | 0.00000594 |
| $10^{-12}$ | 0.00005921 |
| $10^{-13}$ | 0.00094741 |
| $10^{-14}$ | 0.03611195 |
| $10^{-15}$ | 0.84738145 |
| $10^{-16}$ | 4.48168907 |
| $10^{-17}$ | 4.48168907 |
| $10^{-18}$ | 4.48168907 |
| $10^{-19}$ | 4.48168907 |
| $10^{-20}$ | 4.48168907 |

Ut i fra dette kan vi se at for små h går det greit, og feilen er lineær og proposjonal med h. Men når $h$ blir mindre, omlag $10^{-9}$ begynner det å oppstå støy. Men den går helt at skogen når den blir altfor liten ($10^{-16}$). Dataen bruker IEEE 754 som kun bruker 64-bits per tall som vil gi 15-16 sikre desimaler.

2

Her blir eksprimentet gjentatt, men denne gangen benyttes formelen for den symmertrisk deriverte:

$$\frac{f(x+h)-f(x-h)}{2h}$$

| $h$ | Differanse |
|---|---|
| $10^{0}$ | 0.78519727 |
| $10^{-1}$ | 0.0586405 |
| $10^{-2}$ | 0.0004600 |
| $10^{-3}$ | 0.0000036 |
| $10^{-4}$ | 0.0000000 |
| $10^{-5}$ | 0.0000000 |
| $10^{-6}$ | 0.0000000 |
| $10^{-7}$ | 0.0000000 |
| $10^{-8}$ | 0.0000000 |
| $10^{-9}$ | 0.0000159 |
| $10^{-10}$ | 0.0000623 |
| $10^{-11}$ | 0.0001281 |
| $10^{-12}$ | 0.00034024 |
| $10^{-13}$ | 0.00379797 |
| $10^{-14}$ | 0.12070167 |
| $10^{-15}$ | 0.98814184 |
| $10^{-16}$ | 4.48168907 |
| $10^{-17}$ | 4.48168907 |
| $10^{-18}$ | 4.48168907 |
| $10^{-19}$ | 4.48168907 |
| $10^{-20}$ | 4.48168907 |

Dette fungerte litt bedre. Nå kan h være så lav som $10^{-17}$ før ting blir helt krise. Feilen er kvadratisk. Hvorfor dette skjer kan vi forklare ved bruk av Taylorrekker da funksjonen er analytisk. Først ønsker vi å approksimere verdien i $f(x+h)$ ved å utvikle funksjonen rundt $x$:

$$f(x+h) = f(x) + f'(x)h + \frac{f''(x)h}{2!}h^2 + \frac{f'''(x)h}{3!}h^3 + ...$$

Så approksimerer vi verdien i $f(x-h)$ ved å utvikle funksjonen rundt $x$:

$$f(x+h) = f(x) - f'(x)h + \frac{f''(x)h}{2!}h^2 - \frac{f'''(x)h}{3!}h^3 + ...$$

Setter så dette inn i formelen for den symmertrisk deriverte og kansellerer:

$$\frac{2f'(x)h + \frac{2f'''(x)}{3!}h^3 + ...}{2h} = f'(x) + \frac{f'''(x)}{3!}h^2 + ...$$

Feilen her, eller halen om du vil, er proposjonal med $h^2$, og ikke bare $h$. Derfor fungerer den bedre.

3

Vi kan prøve det samme igjen med formelen

$$f'(x) = \frac{f(x-2h) - 8f(x-h) + 8f(x+h) - f(x+2h)}{12h}$$

som kommer fra taylorutviklinger for $f(x+2h), f(x-2h), f(x+h), f(x-h)$. og som har $O(h^4)$

| $h$ | Differanse |
|---|---|
| $10^{0}$ | 0.16825072 |
| $10^{-1}$ | 0.00000921 |
| $10^{-2}$ | 0.00000000 |
| $10^{-3}$ | 0.00000000 |
| $10^{-4}$ | 0.00000000 |
| $10^{-5}$ | 0.00000000 |
| $10^{-6}$ | 0.00000000 |
| $10^{-7}$ | 0.00000000 |
| $10^{-8}$ | 0.00000000 |
| $10^{-9}$ | 0.00000004 |
| $10^{-10}$ | 0.00000269 |
| $10^{-11}$ | 0.00000872 |
| $10^{-12}$ | 0.00001796 |
| $10^{-13}$ | 0.00065712 |
| $10^{-14}$ | 0.00379797 |
| $10^{-15}$ | 0.12070167 |
| $10^{-16}$ | 1.89978302 |
| $10^{-17}$ | 9.62714490 |
| $10^{-18}$ | 62.56559493 |
| $10^{-19}$ | 660.15482738 |
| $10^{-20}$ | 7405.96851990 |

Teoretisk er dette veldig mye bedre. Men i praksis kommer det ikke så godt frem på grunn av datamaskinen. Dersom man absolutt må ha 17 desimaler eller mer bør man faktisk heller bruke en av de andre metodene eller vurdere 128 bits flyttall eller noe annet lurt.

## Numeriske løsninger av varmelikningen

Vi har varmelikningen

$$\dot{u}(x,t) = u''(x,t)$$

med randkrav

$$u(0,t) = u(1,t) = 0$$

og initialkrav

$$u(x,0) = f(x)$$

og den andre ordens differanseformelen for $x$:

$$u''(x,t) \approx \frac{u(x+h,t) - 2u(x,t) + u(x-h,t)}{h^2}$$

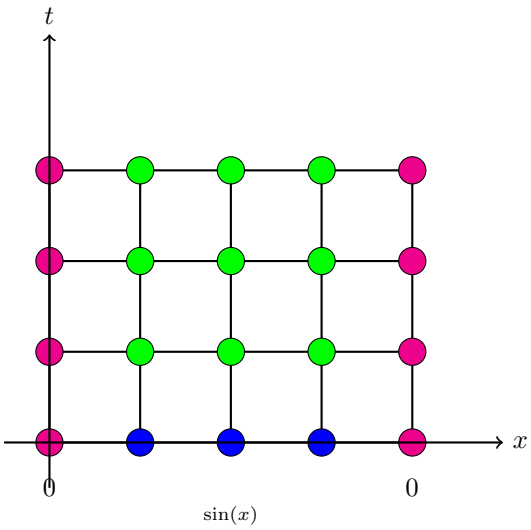
Vi ønsker å løse systemet

$$\dot{u}(x_i,t) = \frac{u_{i+1}(t) - 2u_i(t) + u_{i-1}(t)}{h^2}$$

4 Først løser vi varmelikningen med eksplisitt skjema og initialkrav $u(x,0) = \sin(x)$

$$\frac{u_{i,j+1} - u_{i,j}}{k} = \frac{u_{i+1,j} - 2u_{i,j} + u_{i-1,j}}{h^2}$$

$$u_{i,j+1} = u_{i,j} + \frac{k}{h^2}(u_{i+1,j} - 2u_{i,j} + u_{i-1,j})$$




Dette er et litt forenklet bilde på gitterpunktene vi ønsker å finne. Essensen er at de blå og rosa punktene er kjente da de er gitt av initial- og randkravene, og vi kan se for oss at vi benytter gitterpunktene $(x_0, t_0), (x_1, t_0), (x_2, t_0)$ for å beregne $(x_1, t_1)$ også videre til vi vet verdien til alle de grønne.

Etter å ha testet ulike verdier i koden *Matteoblig 4* for $h$ og $k$ blir det tydelig at Von Neumann[1] hadde rett og at stabilitetsbetingelsen $\frac{k}{h^2} \leq \frac{1}{2}$ gjelder her. Intuitivt gir dette mening, vi kan ikke prøve å umiddelbart hoppe for lagt fremover i tid.

5 Nå implementeres den implisitte metoden, la $\lambda = \frac{k}{h^2}$

$$-\lambda u_{i-1,j+1} + (1-2\lambda)u_{i.j+1} - \lambda u_{i+1,j+1} = u_{i,j}$$

Her benytter vi oss av de blå gitterpunktene og $(x_1, t_1), (x_4, t_1)$ for å finne verdien til alle de grønne på samme rad i ett jafs. Man må løse et lineært likningssystem på formen $A \cdot \mathbf{u}^{j+1} = \mathbf{u}^j$ hvor

$$A = \begin{bmatrix} 1+2\lambda & -\lambda & & & \\ -\lambda & 1+2\lambda & -\lambda & & \\ & \ddots & \ddots & \ddots & \\ & & -\lambda & 1+2\lambda & -\lambda \\ & & & -\lambda & 1+2\lambda \end{bmatrix}$$

I koden *Matteoblig 5* ser vi at denne metoden er stabil for ulike $\frac{k}{h^2}$. Oppløsningen blir bedre for lave h-verdier. Dette kan også forklares ved Von Neumann stabilitetsanalyse.

[1] [1]

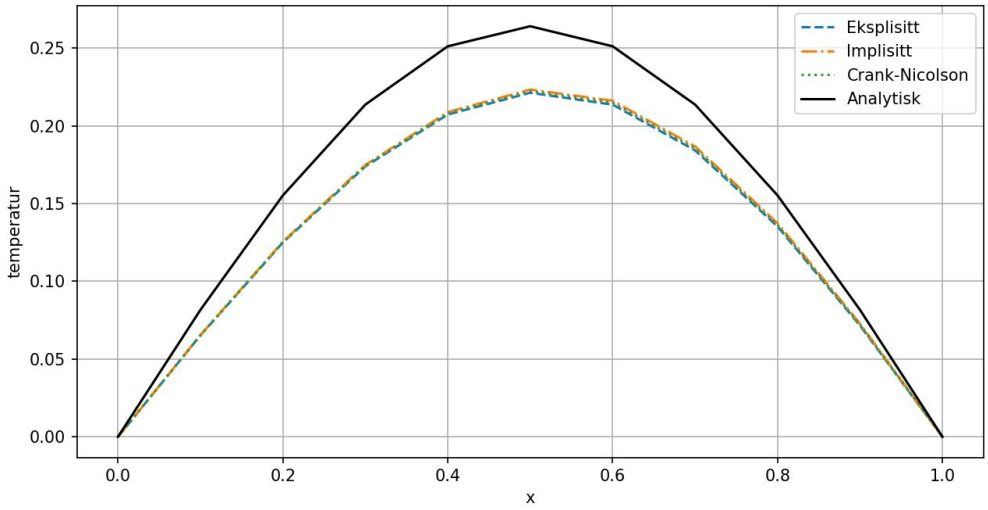
6 Nå bruker vi Crank-Nicolson, en blanding av implisitt og eksplisitt metode, $\lambda = \frac{k}{h^2}$

$$u_{i,j+1} + \frac{\lambda}{2}\left(2u_{i,j+1} - u_{i+1,j+1} - u_{i-1,j+1}\right) = u_{i,j} + \frac{\lambda}{2}\left(u_{i+1,j} - 2u_{i,j} + u_{i-1,j}\right)$$

Vi må løse systemet $Au^{j+1} = Bu^j$ hvor

$$A = \begin{bmatrix} 1+\lambda & -\lambda/2 & 0 & \cdots & 0 \\ -\lambda/2 & 1+\lambda & -\lambda/2 & \cdots & 0 \\ 0 & \ddots & \ddots & \ddots & 0 \\ 0 & \cdots & -\lambda/2 & 1+\lambda & -\lambda/2 \\ 0 & \cdots & 0 & -\lambda/2 & 1+\lambda \end{bmatrix} \quad B = \begin{bmatrix} 1-\lambda & \lambda/2 & 0 & \cdots & 0 \\ \lambda/2 & 1-\lambda & \lambda/2 & \cdots & 0 \\ 0 & \ddots & \ddots & \ddots & 0 \\ 0 & \cdots & \lambda/2 & 1-\lambda & \lambda/2 \\ 0 & \cdots & 0 & \lambda/2 & 1-\lambda \end{bmatrix}$$

I koden *Matteoblig 6* ble denne metoden testet for ulike $h$ og $k$ verdier og den funket hver gang. Dette kan også forklares ved bruk av stabilitetsanalyse.




Etter å ha sammenliknet metodene med den analytiske løsningen kommer vi frem til at Crank-Nicolson eller den implisitte i dette tilfellet er best. Hadde forventet at Crank-Nicolson var bedre fordi det er en annenordens metode i tid, og er betinget stabil.

# Referanser


[1] *Von Neumann stability analysis*. 2025. URL: https://en.wikipedia.org/wiki/Von_Neumann_stability_analysis.