Numerisk løsning for varmelikningen i 2 romlige koordinater.

Lorang Strand

1. februar 2024

Varmelingen $u(t, \boldsymbol{x})$ er en partiell differensiallikning som tar inn både en tidsvariabel t og en posisjonsvariabel $\boldsymbol{x}$.

$$\dot{u}(t, \boldsymbol{x}) = \alpha \Delta u(t, \boldsymbol{x}) \tag{1}$$

hvor α er en konstant. For enkelhetens skyld settes $\alpha = 1$ og fjernes fra videre beregninger. Denne kan lett implementeres senere ved behov.

For varmeflyt over to koordinater (planet) vil likningen se slik ut:

$$\frac{d}{dt}u(t, x, y) = \frac{d^2}{dx^2}u(t, x, y) + \frac{d^2}{dy^2}u(t, x, y) \tag{2}$$

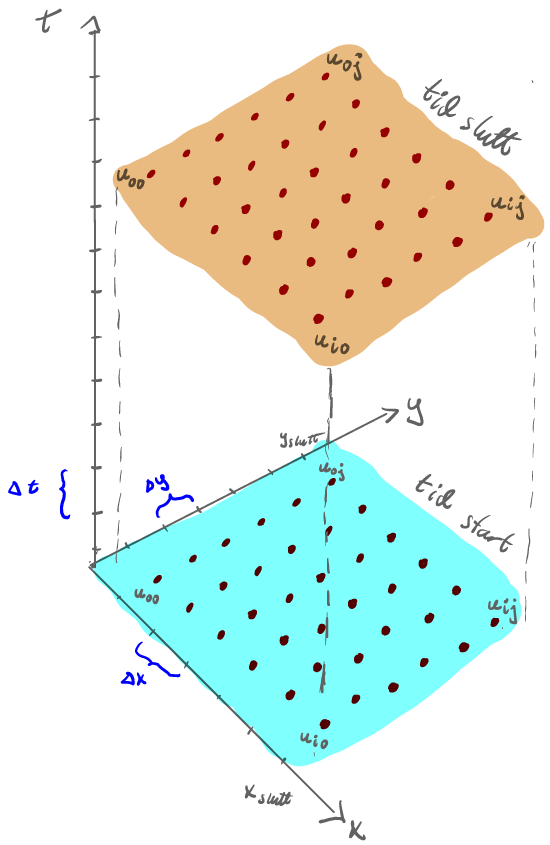
Ved å ta utgangspunkt i definisjonen av den deriverte og sentraldifferans for andredreriverte [1] ser vi at løsningne for én romlig dimensjon blir:

$$\frac{u(t + \Delta t, x) - u(t, x)}{\Delta t} = \frac{u(t, x + \Delta x) - 2u(t, x) + u(t, x - \Delta x)}{\Delta x^2} \tag{3}$$

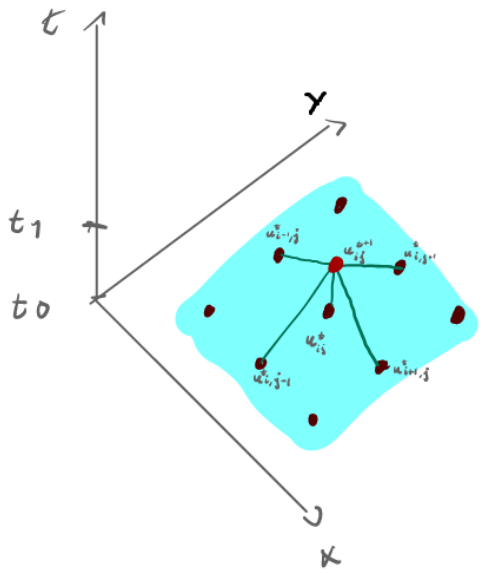
Kombinerer vi denne med løsningen for to romlige dimensjoner og får:

$$\begin{aligned} \frac{u(t + \Delta t, x, y) - u(t, x, y)}{\Delta t} =& \frac{u(t, x + \Delta x, y) - 2u(t, x, y) + u(t, x - \Delta x, y)}{\Delta x^2} \\ &+ \frac{u(t, x, y + \Delta y) - 2u(t, x, y) + u(t, x, y - \Delta y)}{\Delta y^2} \end{aligned} \tag{4}$$

Dette er den eksplistte numeriske løsningen for varmelikningen i planet. Metoden illustreres i figur 1. Planet dannes av en $[n \times m]$ matrise hvor hvert element gjenspeiler et punkt i planet. Denne matrisen utvides for alle i tidssteg, slik at sluttproduktet er en $[i \times n \times m]$ matrise; for alle punkter i planet over all tid. Tempraturen til ett punkt ved neste tidssteg er gitt av likning (4) over. Dette illustreres grafisk illustreres i figur 2.



Figur 1: Planet dannes av punkter satt i forhold til hverandre. Nye plan dannes oppover kjeden når tid forløper.



Figur 2: Neste punkt blir beregnet fra en sum av de nærtliggende foregående punktene.

A Kodesnutt

```
# Obligatisk innlevering for matematikk 4 våren 2024
# skrevet av Lorang Strand
# Numerisk løsning av varmelikningen i to romlige dimensjoner.
# Dokumenatasjon er vedlagt i sepperat notat.

import numpy as np
import matplotlib.pyplot as plt
from matplotlib import cm
import matplotlib.animation as animation

# antall tidssted (oppløsning for t-aksen)
i = 100
# antall posisjonssteg i x-retning (oppløsning for x-aksen)
n = 20
# antall posisjonssteg i y-retning (oppløsning for y-aksen)
m = 20

# start og slutt for tid
```

```
t_start = 0
t_slutt = 1
k = (t_slutt - t_start) / i

# start og slutt for x-akse
x_start = 0
x_slutt = 10
g = (x_slutt - x_start) / n

# start og slutt for y-akse
y_start = 0
y_slutt = 10
h = (y_slutt - y_start) / m

# Det er alltid lurt å verifisere at x_gamma og y_ gamma
# er mindre enn 0.5 for eksplistt metode.
x_gamma = k/h**2
y_gamma = k/g**2
print(x_gamma, '\n',y_gamma)


# matrise som utgjør alle punktene i planet for u gjennom alle tidssteg m.
u = np.zeros((i, m, n))

# Initialkrav: 5 spredt punkter med alle 100 grader
u[0, m//2, n//2] = 100
u[0, m//4, n//2] = 100
u[0, 3*m//4, n//2] = 100
u[0, m//2, n//4] = 100
u[0, m//2, 3*n//4] = 100

# Randkrav: Alle tempraturer langs ytterkanten er alltid lik 0.
# Dette verifiseres ved å ikke regne med første og siste element i planet for
# x- og y- retning i løkken nedenfor.

# Eksplisitt metode
# Gjennomgang av alle tids-"plan"
for a in range(0, i-1):
    # For alle y-akser
    for b in range (1, m-1):
        # For alle punkter i x-retning innenfor gitt y-akse.
        for c in range (1, n-1):
            u[a+1, b, c] = x_gamma * (u[a, b, c+1] - 2*u[a, b, c] + u[a, b, c-1])
            + y_gamma * (u[a, b-1, c] - 2*u[a, b, c] + u[a, b+1, c]) + u[a, b, c]



# plotting av grafene
```

```
x = np.arange (x_start, x_slutt, g)
y = np.arange (y_start, y_slutt, h)
X, Y = np.meshgrid(x, y)

fig = plt.figure()
ax = fig.add_subplot(projection="3d")

# lager en itererbar liste hvor alle separate tidsplot blir lagret.
figures = []
for a in range(len(u)):
    surf = ax.plot_surface (X, Y, u[a], cmap=cm.coolwarm)
    figures.append([surf])

# Fargeskalering av temperatur
fig.colorbar(surf, shrink=0.5, aspect=5)

# Animasjon som iterer gjennom listen og plotter alle elementene
# i tur etter et gitt tidsinterval
anim = animation.ArtistAnimation(
    fig, figures, interval=100, repeat=True
)

# For å lagre animasjonen som en GIF brukes Matplotlib Pillow
writer = animation.PillowWriter(fps=10,
                                metadata=dict(artist='Lorang Strand'),
                                bitrate=1800)
anim.save('animasjon.gif', writer=writer)

plt.show()
```

Referanser

[1] M. andreas Nome, "NUMERISK LØSNING AV PARTIELLE DIFFERENSIALLIKNINGER I," årg. TMA4121, nr. 4.1, 2024.