NTNU

elsys
ELEKTRONISK SYSTEMDESIGN & INNOVASJON

TMA4106

Tittel: Analyse av relasjon mellom diskretisering og varmeligningen.

Skrevet av: Thomas Haug

Gruppe: Bare meg

Dato: 22. april 2024

Sammendrag

Denne rapporten undersøker implementasjonen og sammenligningen av tre numeriske løsningsmetoder for varmeligningen: den eksplisitte, den implisitte, og Crank-Nicolson-metoden.

Formålet med studien var å evaluere metodene under forskjellige betingelser relatert til steglengde i tid og rom sammenlignet med analytisk metode.

Ved å benytte Python og bibliotekene NumPy og Matplotlib, ble de tre metodene implementert for å løse varmeligningen på et diskretisert domene, og resultatene ble visualisert for å sammenligne temperaturfordelingen over tid. Resultatene indikerer at mens den eksplisitte metoden er rask og enkel å implementere, lider den av stabilitetsproblemer når tidssteget økes. Den implisitte metoden, til tross for å være mer beregningskrevende, viste seg å være stabil under nesten alle testforholdene. Crank-Nicolson-metoden som er mest beregningskrevende viste seg å holde seg nærmest analytisk løsning for alle testforholdene.

Studien konkluderer med at valg av numerisk metode bør veiledes av spesifikke krav til stabilitet og nøyaktighet, samt beregningsressurser tilgjengelig for prosjektet. Videre arbeid kan inkludere undersøkelse av flere verdier for initialkrav og andre numeriske løsninger.

Innhold

1 Innledning

Teorien for varmeligningen ble først utviklet i 1822 av en lite kjent matematikker kalt Joseph Fourier.

Teorien var hvordan varme diffunderer gjennom en gitt region. Varmeligningen er uttrykket slikt som under.

$$\frac{\partial u}{\partial t} = \alpha \nabla^2 u \tag{1}$$

Dette er et skummelt matematisk uttrykk, noe som tyder på at det tar lengre tid å regne ut varmeligningen analytisk for hånd enn å designe et programvare som gjør det for oss.

Dermed går denne rapporten ut på å analysere programvare som bruker diskretisering, og approksimasjoner av derivasjon. Numeriske løsningene som blir brukt er eksplisitt, implisitt og Cranker Nicolson-metoden.

For en bedre forståelse se på Figur 1

Figur 1: Hovedmålet for rapporten

Problemet med approksimasjon ligger i ordet, det er en approksimasjon. Dette medfører feil i utregningene. Derfor skal det undersøkes om de numeriske metodene har en approksimasjon-feil som er vesentlig.

Dette bidrar til da om man kan bruke numeriske løsninger i fremtidige utregninger uten å bekymre seg med for store feil.

2

%Teori er viktig.

2.1 Varmeligningen

Varmeligningen er ligningen som skal analyseres, og den er representert som i Ligning 2.

$$\frac{\partial u}{\partial t} = \alpha \nabla^2 u \tag{2}$$

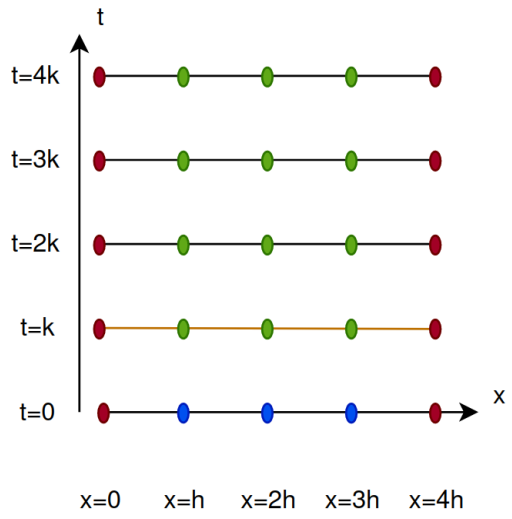
Der u er temperaturen, også kalt termisk profil som funksjon av plass og tid. Variabelen t er for tiden og ∂t er endringen med hensyn til tiden.

α er den termiske diffusivitet til et medium. Det forklarer hvor raskt varme diffunderer gjennom medium. Det holdes konstant her, men en bør være obs siden den er ikke alltid konstant. Noe jeg lærte da jeg tok mat utifra ovnen med våt ovnsvott.

En kjendis blant variablene er ∇^2, dette er Laplace-operatoren, og representerer summen av de andrederiverte med hensyn til romlige koordinater.

2.2 Diskretisering

For numerisk løsning diskretiseres varmeligningen ved bruk av et gitter. Tidsaksen t og romaksen x deles inn med henholdsvis gitteravstand k og h.



Figur 2: Gittersystem for diskretisering

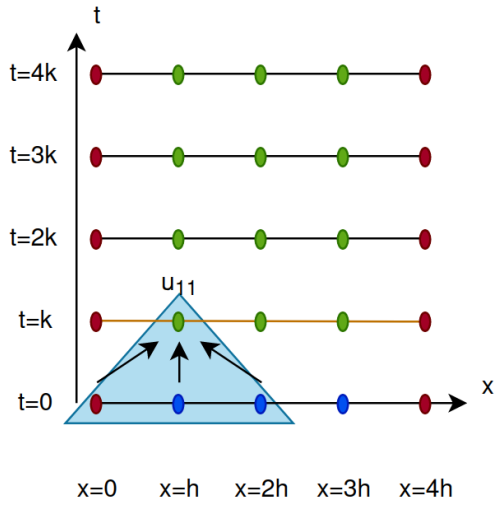
Røde punkter representerer randkrav. Blå punkter representerer initialkrav. Derav skal de grønne punktene løses for. De representeres ved $u(x_i, t_j) \approx u_{ij}$.

2.3 Eksplisitt

Den eksplisitte metoden for løsning av varmeligningen er gitt ved:

$$\frac{u_{i,j+1} - u_{i,j}}{k} = \frac{u_{i+1,j} - 2u_{i,j} + u_{i-1,j}}{h^2} \tag{3}$$

Denne metoden krever kun kjente verdier fra initialkrav eller randkrav for å beregne temperaturen videre. Dette kan representeres som den blå trekanten i Figur 3.



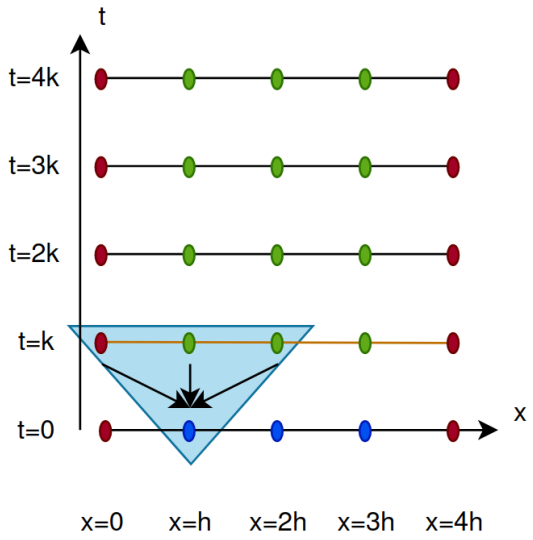
Figur 3: Gitterpunkt brukt i ligningen for eksplisitt.

2.4 Implisitt

I den implisitte metoden inngår også fremtidige verdier, som gjør beregningene mer stabil men også mer beregningskrevende:

$$\frac{u_{i,j+1} - u_{i,j}}{k} = \frac{u_{i+1,j+1} - 2u_{i,j+1} + u_{i-1,j+1}}{h^2} \tag{4}$$

De to ukjente er representert ved grønne sirkler innad i den blå trekanten i Figur 4



Figur 4: Gitterpunkt brukt i ligningen for implisitt.

Her må man løse et sett av ligninger for å finne de ukjente verdiene i hvert tidssteg. Dette blir en matrise. Der $\lambda = k/h^2$

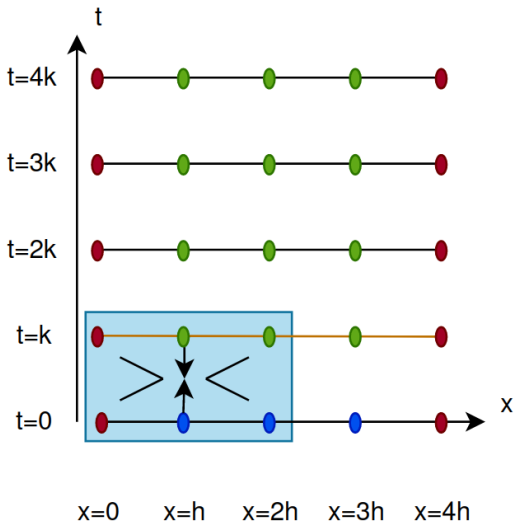
$$\begin{bmatrix} 1+2\lambda & -\lambda & 0 \\ -\lambda & 1+2\lambda & -\lambda \\ 0 & -\lambda & 1+2\lambda \end{bmatrix} \begin{bmatrix} u_{1,1} \\ u_{2,1} \\ u_{3,1} \end{bmatrix} = \begin{bmatrix} u_{1,0} + \lambda u_{0,1} \\ u_{2,0} \\ u_{3,0} + \lambda u_{4,1} \end{bmatrix} \tag{5}$$

2.5 Crank-Nicolson

Crank-Nicolson-metoden er en hybrid mellom eksplisitt og implisitt metode, som gir en god balanse mellom stabilitet og beregningskraft:

$$\frac{u_{i,j+1} - u_{i,j}}{k} = \frac{u_{i+1,j+1} - 2u_{i,j+1} + u_{i-1,j+1}}{2h^2} + \frac{u_{i+1,j} - 2u_{i,j} + u_{i-1,j}}{2h^2} \tag{6}$$

Det samme skjer ved at det er to ukjente innad den blå firkanten.



Figur 5: Gitterpunkt brukt i ligningen for Cranker Nicolson.

Denne metoden krever også løsning av flere ligninger samtidig for å finne de ukjente temperaturene.

$$\begin{aligned} &\begin{bmatrix} 2(1+\lambda) & -\lambda & 0 \\ -\lambda & 2(1+\lambda) & -\lambda \\ 0 & -\lambda & 2(1+\lambda) \end{bmatrix} \begin{bmatrix} u_{1,1} \\ u_{2,1} \\ u_{3,1} \end{bmatrix} \\ &= \begin{bmatrix} 2(1-\lambda) & \lambda & 0 \\ \lambda & 2(1-\lambda) & \lambda \\ 0 & \lambda & 2(1-\lambda) \end{bmatrix} \begin{bmatrix} u_{1,0} \\ u_{2,0} \\ u_{3,0} \end{bmatrix} + \begin{bmatrix} \lambda(u_{0,1}+u_{0,0}) \\ 0 \\ \lambda(u_{4,0}+u_{4,1}) \end{bmatrix} \end{aligned} \tag{7}$$

3 Metode

3.1 Numerisk implementering i Python

For analysering benyttes Python for å implementere og sammenligne tre numeriske løsningsmetoder for varmeligningen. Det brukes 'numpy' for matriseoperasjoner og 'matplotlib' for å visualisere løsningene.

3.1.1 Initialisering og parametere

Man definerte et romlig og et tidsmessig gitter ved hjelp av parametrene h for romlig skrittstørrelse og k for tidsintervall. Stabiliteten til den eksplisitte metoden ble sjekket ved å beregne forholdet $\frac{k}{h^2}$, og en advarsel ble gitt hvis forholdet overskred 0,5, noe som indikerer potensiell ustabilitet i løsningen.

```
import numpy as np
import matplotlib.pyplot as plt
import matplotlib.cm as cm

h = 0.1  # Romlig skrittstørrelse
k = 0.01  # Tidsintervall
factor = k/(h**2)
if factor > 0.5:
    print("Factor is too large, the solution will be unstable for explicit scheme.")
space = 1  # Total lengde i rommet
time = 0.5  # Total tid

x = np.arange(0, space + h, h)
t = np.arange(0, time + k, k)
```

3.1.2 Definisjon av Grense- og Initialbetingelser

For å simulere varmespredning langs en stav med lengde 1 meter, ble null-temperaturbetingelser satt ved begge endene av staven. Den initiale temperaturfordelingen ble satt som en sinusformet funksjon over stavens lengde.

```
boundaryCondition = [0, 0]
initialConditions = np.sin(np.pi * x)
```

3.1.3 Numerisk Løsning

Det implementeres funksjoner for å beregne temperaturen ved hver tidspunkt og posisjon på gitteret for de tre metodene. Resultatene ble lagret i separate matriser for hver metode.

```
def heat_equation_all(h, k, boundaryCondition, initialConditions, time, space):
    factor = k/(h**2)
    if factor > 0.5:
        print("Factor is too large, the solution will be unstable for
        explicit scheme.")
    x = np.arange(0, space+h, h)
    t = np.arange(0, time+k, k)
    n = len(x)
    m = len(t)
    T = np.zeros((n, m))
    T[0, :] = boundaryCondition[0]
    T[-1, :] = boundaryCondition[1]
    T[:, 0] = initialConditions
    T2 = T.copy()
    T3 = T.copy()
    # Compute solutions with explicit, implicit, and Crank-Nicolson methods
```

Derav koden for eksplisitt utregning for temperatur matrisen.

```
    for j in range(1, m):
        for i in range(1, n-1):
            T[i, j] = factor*(T[i+1, j-1] - 2*T[i, j-1] + T[i-1, j-1]) + T[i, j-1]
```

Koden for implisitt utregning for temperatur matrisen.

```
    A = np.diag([1+2*factor]*(n-2)) + np.diag([-factor]*(n-3), -1)
    + np.diag([-factor]*(n-3), 1)
    for j in range(1, m):
        b = T[1:-1, j-1].copy()
        b[0] = b[0] + factor*T[0, j]
        b[-1] = b[-1] + factor*T[-1, j]
        solution = np.linalg.solve(A, b)
        T2[1:-1, j] = solution
```

Koden for Cranker Nicolson utrregning for temperatur matrisen.

```
    A = np.diag([2+2*factor]*(n-2)) + np.diag([-factor]*(n-3), -1)
    + np.diag([-factor]*(n-3), 1)
    B = np.diag([2-2*factor]*(n-2)) + np.diag([factor]*(n-3), -1)
    + np.diag([factor]*(n-3), 1)
    for j in range(0, m-1):
        b = T[1:-1, j].copy()
        b = np.dot(B, b)
        b[0] = b[0] + factor*(T[0, j] + T[0, j+1])
        b[-1] = b[-1] + factor*(T[-1, j] + T[-1, j+1])
```

```
        solution = np.linalg.solve(A, b)
        T3[1:-1, j+1] = solution
```

3.1.4 Visualisering

Resultatene fra hver metode ble visualisert i samme plot for direkte sammenligning. Forskjellige fargekart ble brukt for å skille mellom løsningene fra de forskjellige metodene. En analytisk løsning ble også plottet for referanse.

```
plt.legend(loc="upper right")
plt.xlabel("Distance [m]")
plt.ylabel("Temperature [°C]")
plt.title("Comparison of Numerical Schemes for the Heat Equation")
plt.show()
```

3.1.5 Kjøring av Simuleringen

Simuleringen ble kjørt ved å kalle funskjonen med definerte parametere, og resultatene ble presentert gjennom grafene generert av matplotlib.

```
heat_equation_all(h, k, boundaryCondition, initialConditions, time, space)
```

Dermed har man en visuell representasjon av hvordan temperaturprofilen utvikler seg over tid under de forskjellige numeriske løsningsmetodene. Dette bidrar til visuell sammenligning av metodene mot analytisk metode.

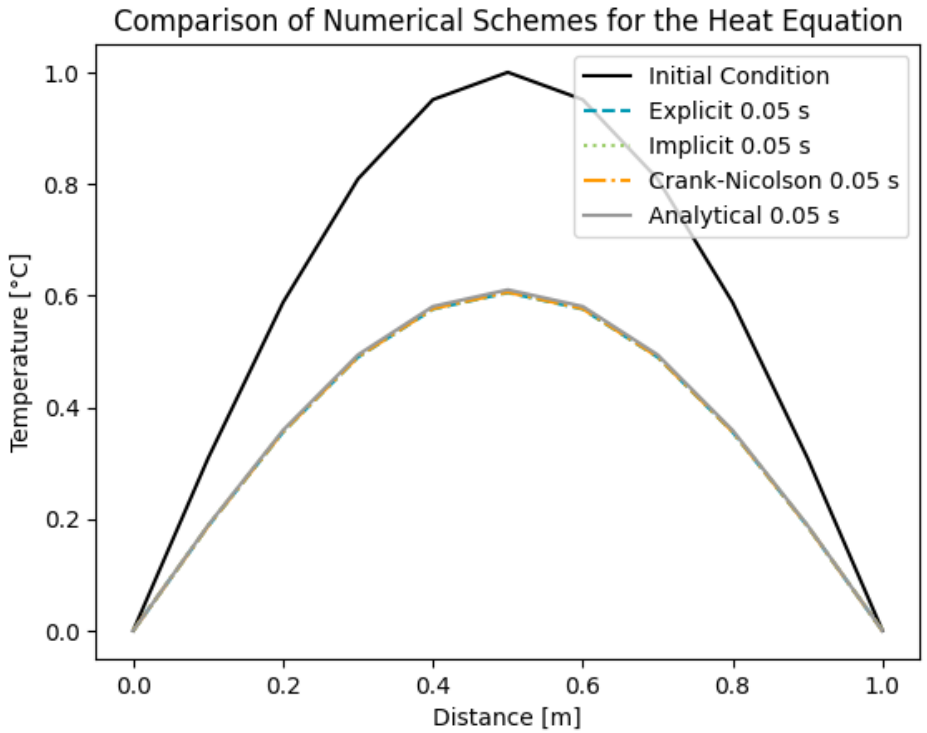
4 Resultater

Denne delen av rapporten presenterer en sammenligning av løsningsmetodene eksplisitt, implisitt og Crank-Nicolson for varmeligningen ved forskjellige λ-verdier, som reflekterer forholdet $\frac{k}{h^2}$. Dette gir innsikt i hver metodes stabilitet og nøyaktighet under forskjellige numeriske forhold.

4.1 Sammenligning mellom eksplisitt, implisitt, og Crank-Nicolson

4.1.1 $\lambda = 0.5$

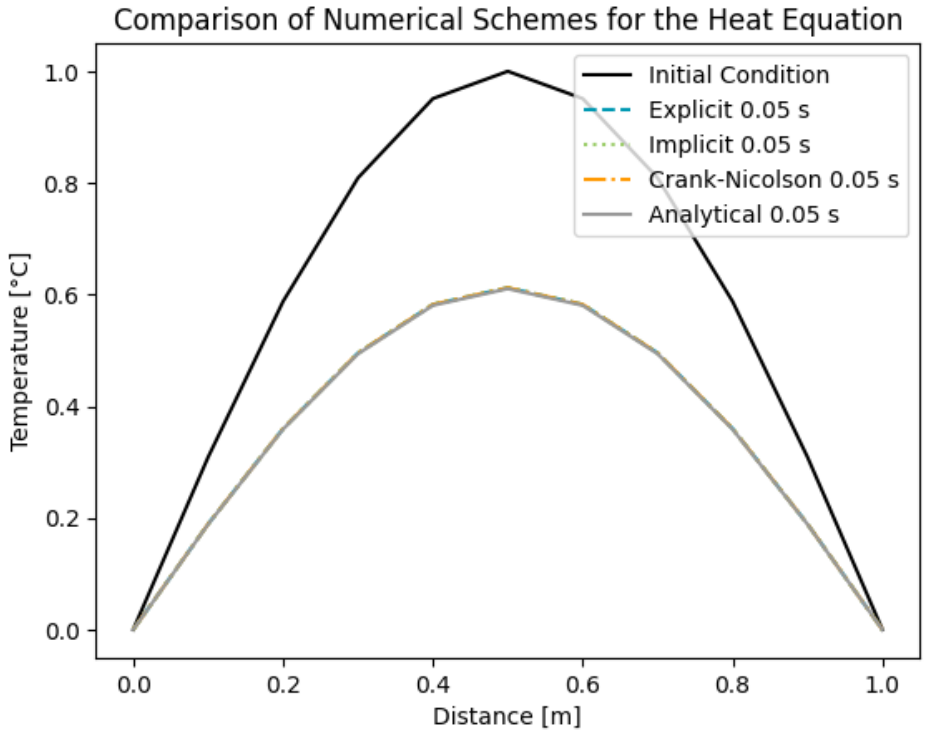
Over dette segmentet presenteres resultater ved et λ som nærmer seg grensen for stabilitet i eksplisitte metoden.



Figur 6: Sammenligning ved $\lambda = 0.5$.

4.1.2 $\lambda = 0.05$

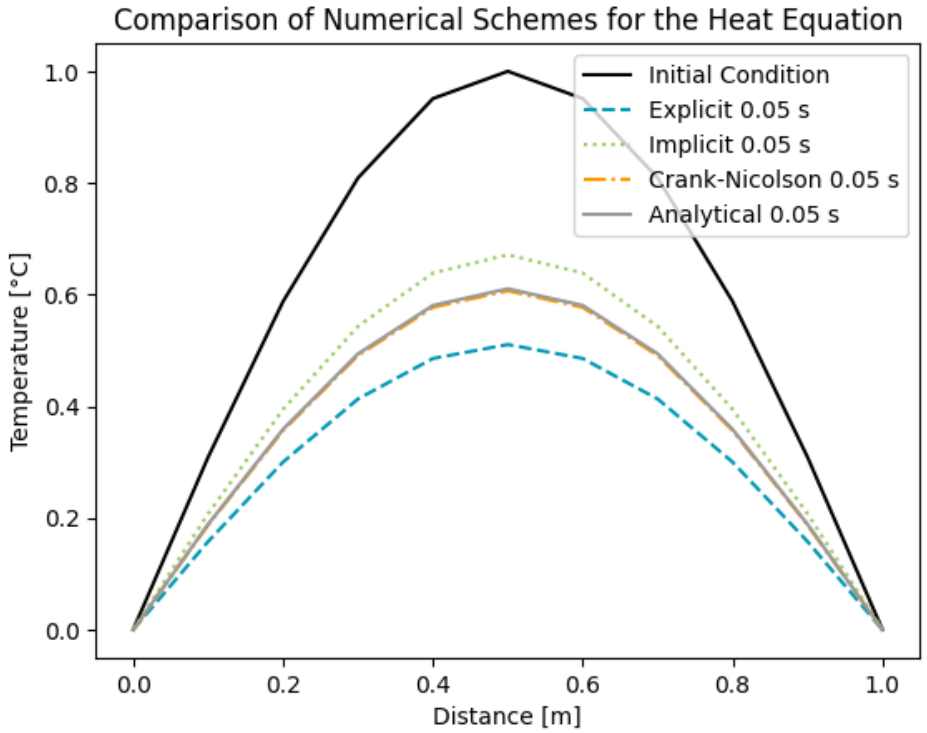
Dette z fokuserer på et lavt λ, hvor alle metodene forventes å oppføre seg stabilt.



Figur 7: Sammenligning ved $\lambda = 0.05$.

4.1.3 $\lambda = 5$

Her utforskes en høy λ-verdi, der eksplisitte metoden viser signifikant ustabilitet. Overraskende viser implisitt en relativ stor forskjell fra den analytiske verdien. Crank-Nicolson viser seg å ligge omtrent oppå den analytiske løsningen.



Figur 8: Sammenligning ved $\lambda = 5$.

Resultatene indikerer tydelige forskjeller i ytelse og stabilitet mellom de numeriske metodene, noe som understreker viktigheten av riktig valg av metode basert på de spesifikke kravene til numerisk stabilitet og nøyaktighet for en gitt applikasjon.

5 Konklusjon

Denne studien har utforsket og sammenlignet ytelsen til tre numeriske løsningsmetoder for varmeligningen: den eksplisitte metoden, den implisitte metoden, og Crank-Nicolson-metoden. Gjennom anvendelse av disse metodene på en standard varmetransportmodell, har vi oppnådd innsikt i deres stabilitet, nøyaktighet og anvendelighet under forskjellige betingelser.

Den eksplisitte metoden viste seg å være den enkleste å implementere, men den var også den minst stabile, spesielt ved større tidssteg. Denne metoden er passende for simuleringer hvor små tidssteg kan anvendes uten at det fører til uforholdsmessig høy beregningskostnad. Den implisitte metoden, derimot, viste betydelig større stabilitet og var ikke følsom for størrelsen på tidssteget, noe som gjør den ideell for lengre simuleringer hvor større tidssteg er nødvendig.

Crank-Nicolson-metoden kombinerer fordeler fra både den eksplisitte og implisitte metoden, og tilbyr en optimal balanse mellom stabilitet og beregningskostnad. Denne metoden produserte konsistente og pålitelige resultater, og viste seg å være spesielt verdifull i situasjoner hvor både nøyaktighet og numerisk effektivitet er kritisk.

Basert på disse observasjonene, anbefales Crank-Nicolson-metoden for de fleste praktiske anvendelser av varmeligningen, spesielt når både nøyaktighet og beregningseffektivitet er av betydning. For fremtidige arbeider vil det være fordelaktig å utforske adaptive tidsstegsteknikker for å ytterligere forbedre effektiviteten til de eksplisitte og implisitte metodene, samt å implementere mer avanserte modeller for termisk diffusivitet som tar hensyn til variabler som endrer seg med temperatur eller andre miljøfaktorer.

Denne studien bekrefter viktigheten av nøye valg av numerisk metode basert på spesifikke krav og forhold i simuleringsscenarier. Videre forskning og utvikling innen numeriske metoder vil fortsette å spille en nøkkelrolle i å forbedre vår forståelse og evne til å nøyaktig modellere komplekse fysiske systemer.

Referanser

[1] Egner, Torbjørn: *Folk og røvere i Kardemomme By.* Cappelen Damn, 1980, ISBN 9788202045944.

A Vedlegg A