Datorlab. 2: Värmeledning

FFM234 Vektorfält och klassisk fysik

Jonatan Haraldsson Oscar Lindberg

Chalmers tekniska högskola
Oktober 2021

Inledning

Värmeledning eller konduktion beskriver hur värme transporteras i ett material, givet olika egenskaper hos materialet. I denna rapport undersöks olika aspekter av endimensionell värmeledning i marken genom numeriska och analytiska metoder. Huvudsyftet är att för två olika fall av villkor finna en stationär, alltså en tidsoberoende, lösning till värmeekvationen

$$\frac{\partial T}{\partial t} = \frac{\lambda}{c\rho}\nabla^2 T + u \qquad \text{(värmeekvationen)}$$

under en kall vinterdag. I värmeekvationen är T ett temperaturfält som funktion av tid och läge, λ värmeledningsförmågan, c värmekapacitiviteten och ρ densiteten. I forsättningen kommer kvoten $\lambda/(c\rho) := k$. Termen $u = s/(c\rho)$ är ett skalärt fält som beskriver hur en värmekälla skulle påverka temperaturfördelningen.

De två olika fall som analyseras är (1) endimensionell temperaturfördelning med Dirichlets randvillkor och (2) endimensionell temperaturfördelning med värmekälla (och Dirichlets randvillkor). I båda fallen betraktas området från markytan ner till ett tjälfritt djup på 1,0 m. Variabeln x, vilken definieras med positiv riktning nedåt, beskriver djupet i meter. Den initiala temperaturen ($t = 0$) antas vara $T = 0\,^\circ\mathrm{C}$, förutom vid markytan och vid en meters djup där $T(x = 0) = -10\,^\circ\mathrm{C}$ respektive $T(x = 1) = 0\,^\circ\mathrm{C}$ antas gälla under hela förloppet i enlighet med Dirichlets randvillkor. Vidare antas att $\lambda = 1{,}0\,\mathrm{W\,m^{-1}\,K^{-1}}$, $c = 1000\,\mathrm{J\,kg^{-1}\,K^{-1}}$ och $\rho = 1500\,\mathrm{kg\,m^{-3}}$. I fall 1 genomförs analysen utan någon värmekälla, medan en homogen värmekälla med $s = 100\,\mathrm{W\,m^{-3}}$ antas vara närvarande i fall 2.

Metod

Inledningsvis konstaterades att konstanterna k och u kan bestämmas till $k = \frac{\lambda}{c\rho} = \frac{2}{3} \cdot 10^{-6}\,\mathrm{m^2\,s^{-1}}$ respektive $u = \frac{s}{c\rho} = \frac{2}{3} \cdot 10^{-4}\,^\circ\mathrm{C\,m^{-1}}$. Randvillkoren, vilka är nödvändiga för att bestämma en lösning till värmeekvationen, identifierades till $T(x = 0)$=$-10\,^\circ\mathrm{C}$ vid marknivån och $T(x = 1)$ $0\,^\circ\mathrm{C}$ vid en meters djup.

Numerisk lösning

För de numeriska beräkningarna användes `Python` tillsammans med biblioteken `numpy` och `matplotlib`. Källkoden för fall 1 och fall 2 presenteras i bilagan. För att finna en numerisk

lösning till värmeekvationen genomfördes iterativa beräkningar av temperaturen över tiden och djupet. Tidssteget sattes till

$$\delta t = K\frac{\delta x^2}{k},$$

där konstanten $K = 0,25$. För varje tidssteg beräknades temperaturen i varje punkt i mellan 0 till 1 meters djup (totalt 28 punkter, exklusive randpunkterna) från värdena från den föregående iterationen enligt formeln

$$T_i(t+\delta t) = T_i(t) + \delta t k\frac{T_{i+1}(t) - 2T_i(t) + T_{i-1}(t)}{\delta x^2} + \delta t u_i(t),$$

där den sista termen i fall 1 blir 0 eftersom det saknas en källa ($u = 0$). Under varje tidssteg approximerades tidsderivatan i respektive punkt genom

$$\frac{\partial T_i}{\partial t} = \frac{T_i(t+\delta t) - T_i(t)}{\delta t}.$$

När normen av matrisen innehållande approximationen av tidsderivatan av temperaturen understigit ett värde av $\varepsilon = 5 \cdot 10^{-6}$ avbröts iterationerna och en ungefärlig stationärlösning ansågs vara uppnådd.

Två typer av grafer togs fram för att visualisera temperaturfältets utveckling för respektive fall. Den första bestod av en djup-tid-plot över vilken temperaturen visualiserades genom `contourf`. Den andra bestod istället av 15 djup-temperatur-kurvor där tidsutvecklingen visualiserades genom en färgkodning. För att undersöka huruvida den numeriska lösningen närmade sig den analytiska inkluderades även denna.

Analytisk lösning

En stationär lösning till värmeekvationen svarar mot en lösning som ej har ett tidsberoende, alltså då $\frac{\partial T}{\partial t} = 0$. Eftersom problemet endast är endimensionellt gäller det att $\nabla^2 T = \frac{\partial T^2}{\partial x^2}$. Således kan värmeekvationen skrivas

$$\frac{-u}{k} = \frac{\partial T^2}{\partial x^2}$$

och sedan lösas analytiskt med upprepad integration, vilket ger

$$T(x) = -\frac{u}{2k}x^2 + Cx + D. \tag{1}$$

Konstanterna C med enhet $\mathrm{K\,m^{-1}}$ och D med enhet K i ekvation 1 bestäms med randvillkoren $T(0) = -10$ och $T(1) = 0$. Sammantaget erhålls lösningen

$$T(x) = -\frac{u}{2k}x^2 + \left(10 + \frac{u}{2k}\right)x - 10. \tag{2}$$

I fall 1 finns ingen värmekälla i marken, vilket svarar mot $u = 0$ i värmeekvationen. Det följer från ekvation 2 att $T(x) = 10(x-1)$ är analytisk lösning i fall 1.

Flödet ϕ_q på ett djup x ges av

$$\phi_q = -\lambda\frac{\partial T}{\partial x}.$$

Vid stationärt tillstånd kan flödet därför uttryckas

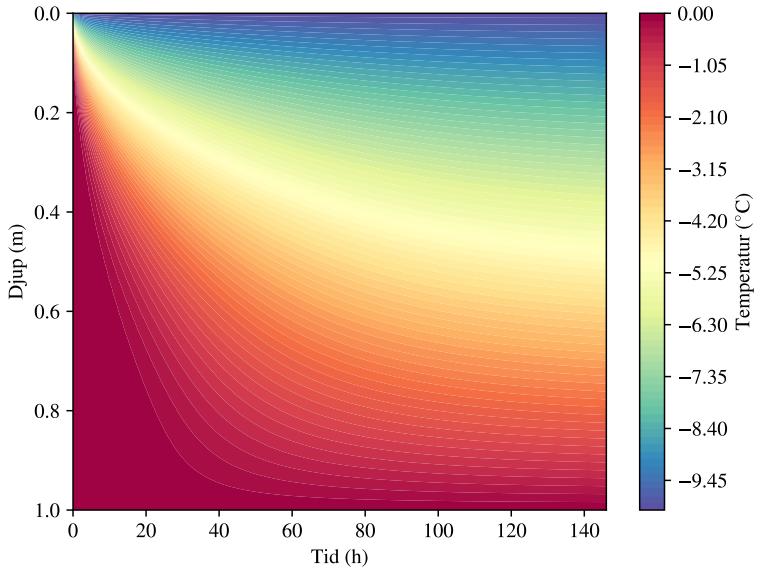
$$\phi_q = \lambda \left(\frac{u}{k} x - 10 - \frac{u}{2k} \right). \tag{3}$$

Flödet i randpunkterna kan från ekvation 3 bestämmas till $\phi_{q0} = -\lambda \left(10 + \frac{u}{2k}\right)$ och $\phi_{q1} = \lambda \left(\frac{u}{2k} - 10\right)$ För att finna totalt flöde beräknas differensen mellan ϕ_{q1} och ϕ_{q0}. Det ger ett totalt flöde $\lambda \frac{u}{k}$ ut i båda fallen.

Resultat och diskussion

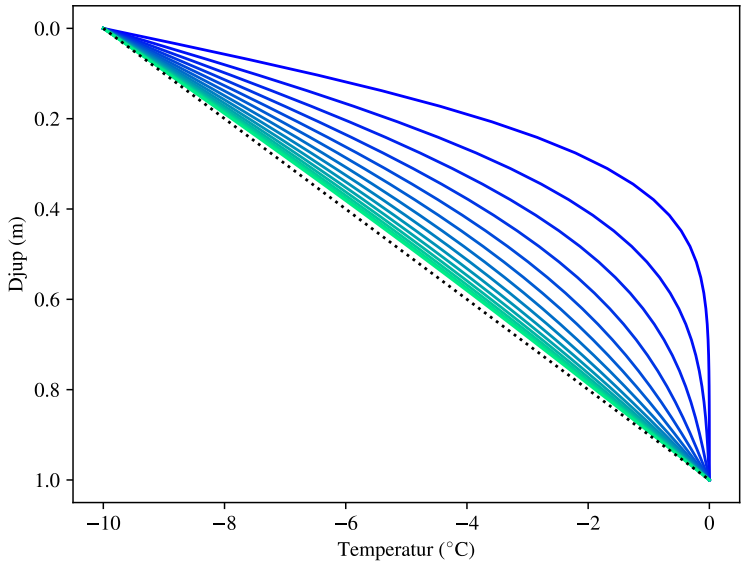
Fall 1 utan värmekälla

Tidsförloppet över temperaturen visualiseras i figur 1. Allteftersom t ökar jämnar temperaturfördelningen ut sig mot en jämn färgförändring. Temperaturförändringen sker fortare de första 60 timmarna för att sedan plana av. Det är en naturlig följd av värmeekvationen då $\nabla^2 T \propto \frac{\partial T}{\partial t}$ ger att stora temperaturskillnader i rummet orsakar en snabbare temperaturförändring i tiden.



Figur 1: Temperaturfältets utveckling över tiden för fall 1 (utan värmekälla).

I figur 2 visas 15 numeriskt beräknade lösningskurvor genom tidsförloppet. Färgen på kurvorna går från blå för små t och blir grönare för större t. Den streckade kurvan i samma graf är den analytiska lösningen till $T(x)$. Från figur 2 följer det att de numeriska lösningarna närmar sig den analytiska för större och större t.



Figur 2: 15 lösningskurvor från iterationerna i fall 1 (utan värmekälla). Blå färg svarar mot små t, medan grön färg svarar mot större t ($t_{max} \approx 146{,}2\,\mathrm{h}$). Den streckade kurvan är den analytiska lösningen vid stationärt tillstånd.

För att uppnå en stationär lösning numeriskt avbröts iterationerna när $\frac{\partial T}{\partial t} < \varepsilon$. För valet av $\varepsilon = 5 \cdot 10^{-6}$ gavs en total tid på $146{,}2\,\mathrm{h} \approx 6\,\mathrm{d}$, motsvarande 1180 iterationer, innan temperaturfördelningen ansågs stabil.

Flödet kunde bestämmas till $\phi_{q1} = \lambda\left(\frac{u}{2k} - 10\right)$ och $\phi_{q0} = -\lambda\left(10 + \frac{u}{2k}\right)$ för $x = 1$ respektive $x = 0$. Eftersom $u = 0$ i fall 1 ges ett flöde $\phi_{q0} = -10\lambda = \{\lambda = 1\} = -10\,\mathrm{W m^{-2}}$ och ett flöde på $\phi_{q1} = -10\lambda = -10\,\mathrm{W m^{-2}}$. Värmeströmmen är således lika stor vid båda randpunkterna och det flödar alltså in lika mycket som flödar ut. Det totala flödet ut, vilket bestämdes till $\lambda\frac{u}{k}$, blir i fall 1 lika med noll.

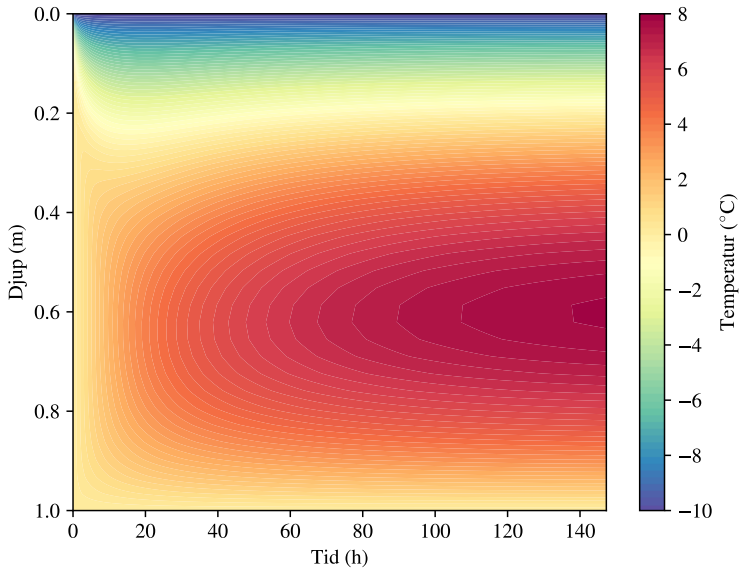
Fall 2 med konstant värmekälla

På motsvarande sätt som i fall 1 presenteras temperaturfältets utveckling över tiden i figur 3. Här syns att det finns en värmekälla som påverkar markens temperatur över tiden. Förändringen är som störst i början och närmar sig sedan ett konstant tillstånd.

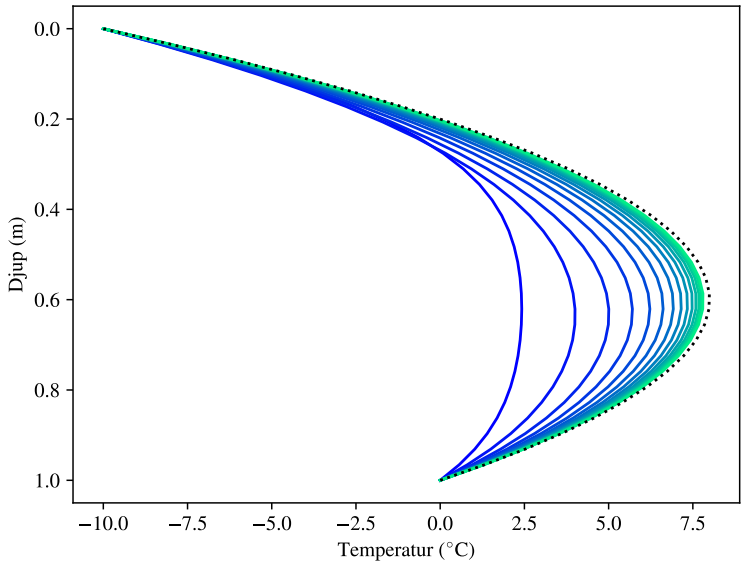
I figur 4 visas på samma sätt som tidigare 15 lösningskurvor, samt den analytiska lösningen. Det är tydligt att kurvorna närmar sig den analytiska lösningen när tiden ökar.

Med samma val av ε som tidigare avbröts iterationerna efter en simulering av $147{,}4\,\mathrm{h} \approx 6\,\mathrm{d}$, motsvarande 1190 iterationer.

Det totala flödet ut beräknades genom ekvation 3 till $\phi_q = \lambda\frac{u}{k} = s = 100\,\mathrm{W\,m^{-2}}$.



Figur 3: Temperaturfältets utveckling över tiden för fall 2 (med värmekälla).



Figur 4: 15 lösningskurvor från iterationerna i fall 2 (med värmekälla). Blå färg svarar mot små t, medan grön färg svarar mot större t ($t_{max} \approx 147{,}4\,\text{h}$). Den streckade kurvan är den analytiska lösningen vid stationärt tillstånd.

Bilaga

Källkod del 1 (Python)

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

# LaTeX font
plt.rcParams['mathtext.fontset'] = 'cm'
plt.rcParams['font.family'] = 'STIXGeneral'

# Definitions
rho = 1500  # Density
c = 1000  # Specific heat capacity
l = 1  # Thermal conductivity
k = l/(c*rho)
K = 0.25
m = 30  # Number of points
d = 1  # Total depth
s = np.zeros(m)
u = s/(c*rho)
x = np.linspace(0, d, m)
dx = d/(m-1)
dt = K*(dx**2/k)

T_old = np.zeros(m)  # Temperature (old)
T_new = np.zeros(m)  # Temperature (new)
T_old[0] = -10
T_old[m-1] = 0
T_all = []
T_plots = []
T_prime = np.zeros(m)
T_prime_all = []
T_prime_plots = []

t = 0
t_vec = []
eps = 5e-6

# Loop
cont = True
while cont:

    for i in range(x.size):
        if i == 0 or i == 29:
            T_new[i] = T_old[i]
        else:
            T_new[i] = T_old[i] + dt*k*(T_old[i+1]-2*T_old[i]+T_old[i-1])/dx**2 + dt*u[i]
```

```
        t_vec.append(t)
        t += dt
        T_prime = (T_new - T_old)/dt
        T_all.append(T_new.copy())
        T_prime_all.append(T_prime.copy())

        if np.linalg.norm(T_prime) < eps:
            cont = False

        T_old = np.copy(T_new)

print('Time (s):', t)
print('Time (h):', t/3600)
print('Time (d):', t/(3600*24))
print('Number of iterations:', int(t/dt))

### PLOTS ###
T_steps = np.linspace(0, len(T_prime_all), 15).astype(int)
T_steps[-1] -= 1

for i in range(len(T_steps)):
    T_prime_plots.append(T_prime_all[T_steps[i]])
    T_plots.append(T_all[T_steps[i]])

# Depth-time-temperature contourf
xp = np.linspace(0, d, len(t_vec))
X, Y = np.meshgrid(x, t_vec)
CS = plt.contourf(Y/3600, X, T_all, 75, cmap=plt.cm.Spectral_r)
cb = plt.colorbar(CS)
cb.set_label('Temperatur ($^{\circ}$C)', fontsize=11)
cb.ax.tick_params(labelsize=11)
plt.xlabel('Tid (h)', fontsize=11)
plt.ylabel('Djup (m)', fontsize=11)
plt.xticks(fontsize=11)
plt.yticks(fontsize=11)
ax = plt.gca()
ax.set_ylim(ax.get_ylim()[::-1])
plt.show()

# Depth-temp-time graphs
colormap = plt.cm.winter
plt.gca().set_prop_cycle(plt.cycler('color', colormap(np.linspace(0, 1, len(T_plots)))))

for i in range(1, len(T_plots)):
    plt.plot(T_plots[i], x)

plt.plot(10*x-10, x, 'k:')  # Analytic solution
```

```
plt.xlabel('Temperatur ($^{\circ}$C)', fontsize=11)
plt.ylabel('Djup (m)', fontsize=11)
plt.xticks(fontsize=11)
plt.yticks(fontsize=11)
ax = plt.gca()
ax.set_ylim(ax.get_ylim()[::-1])
plt.show()
```

Källkod del 2 (Python)

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

# LaTeX font
plt.rcParams['mathtext.fontset'] = 'cm'
plt.rcParams['font.family'] = 'STIXGeneral'

# Definitions
rho = 1500  # Density
c = 1000  # Specific heat capacity
l = 1  # Thermal conductivity
k = l/(c*rho)
K = 0.25
m = 30  # Number of points
d = 1  # Total depth
s = np.ones(m)*100
u = s/(c*rho)
x = np.linspace(0, d, m)
dx = d/(m-1)
dt = K*(dx**2/k)

T_old = np.zeros(m)  # Temperature (old)
T_new = np.zeros(m)  # Temperature (new)
T_old[0] = -10
T_old[m-1] = 0
T_all = []
T_plots = []
T_prime = np.zeros(m)
T_prime_all = []
T_prime_plots = []

t = 0
t_vec = []
eps = 5e-6

# Loop
cont = True
while cont:
```

```
    for i in range(x.size):
        if i == 0 or i == 29:
            T_new[i] = T_old[i]
        else:
            T_new[i] = T_old[i] + dt*k*(T_old[i+1]-2*T_old[i]+T_old[i-1])/dx**2 + dt*u[i]

    t_vec.append(t)
    t += dt
    T_prime = (T_new - T_old)/dt
    T_all.append(T_new.copy())
    T_prime_all.append(T_prime.copy())

    if np.linalg.norm(T_prime) < eps:
        cont = False

    T_old = np.copy(T_new)

print('Time (s):', t)
print('Time (h):', t/3600)
print('Time (d):', t/(3600*24))
print('Number of iterations:', int(t/dt))

### PLOTS ###
T_steps = np.linspace(0, len(T_prime_all), 15).astype(int)
T_steps[-1] -= 1

for i in range(len(T_steps)):
    T_prime_plots.append(T_prime_all[T_steps[i]])
    T_plots.append(T_all[T_steps[i]])

# Depth-time-temperature contourf
xp = np.linspace(0, d, len(t_vec))
X, Y = np.meshgrid(x, t_vec)
CS = plt.contourf(Y/3600, X, T_all, 75, cmap=plt.cm.Spectral_r)
cb = plt.colorbar(CS)
cb.set_label('Temperatur ($^{\circ}$C)', fontsize=11)
cb.ax.tick_params(labelsize=11)
plt.xlabel('Tid (h)', fontsize=11)
plt.ylabel('Djup (m)', fontsize=11)
plt.xticks(fontsize=11)
plt.yticks(fontsize=11)
ax = plt.gca()
ax.set_ylim(ax.get_ylim()[::-1])
plt.show()

# Depth-temp-time graphs
colormap = plt.cm.winter
plt.gca().set_prop_cycle(plt.cycler('color', colormap(np.linspace(0, 1, len(T_plots)))))
```

```
for i in range(1, len(T_plots)):
    plt.plot(T_plots[i], x)

plt.plot(-1/2*x**2*u/k+(10+1/2*u/k)*x-10, x, 'k:')  # Analytic solution

plt.xlabel('Temperatur ($^{\circ}$C)', fontsize=11)
plt.ylabel('Djup (m)', fontsize=11)
plt.xticks(fontsize=11)
plt.yticks(fontsize=11)
ax = plt.gca()
ax.set_ylim(ax.get_ylim()[::-1])
plt.show()
```