AVKJØLING

python™ & PASCO
koding med mening

I dette forsøket skal vi undersøke Newtons avkjølingslov, ved å gjennomføre en måleserie som viser hvordan temperaturen endres i et objekt som kjøles ned i forhold til omgivelsene. Forsøket kan enten gjøres ved at man varmer opp en temperatursensor, og lar denne avkjøle, eller ved at man setter sensoren i noe varmt (f.eks. en kaffekopp).

Etter å ha gjennomført målingene og lagret dataene i SPARKvue eller Capstone, trekker vi fra rom-temperaturen i alle dataene, slik at temperaturen går mot 0^oC. Vi laster deretter inn dataene i Python, hvor vi analyserer dataene videre og sammenligner med teorien for dette forsøket.

I dette eksempelet bruker vi et eget sett med funksjoner skrevet for å lette lesing av data fra PASCO-programvare. Det også mulig å gjøre dette med egne Python-biblioteker, som f.eks. pandas,

```
# LES DATA (CSV-fil fra Capstone) med pasco-biblioteket
# Hent ut kolonne 1 og 3: tid og temperatur
datatabell = pasco.lesData('newton-avkjolingslov.csv')
tid = [rad[1] for rad in datatabell]
temp = [rad[3] for rad in datatabell]
```

Etter å ha lastet inn måledataene, konverterer vi den målte temperaturen til en rett linje y, ved å ta logaritmen av temperaturene: $y = \log(T)$. Disse punktene tilpasser vi med en lineær regresjon (med *polyfit* i numpy-biblioteket), hvor vi finner koeffisientene *a* og *b* og likningen $y = at + b$.

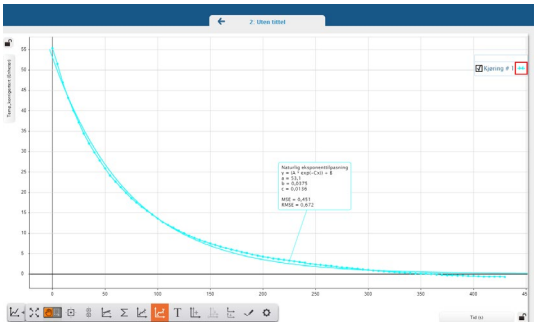
```
# KURVETILPASNING
log_temp = np.log(temp)
a,b = np.polyfit(tid,log_temp,1)
fit_temp = [np.exp(a*t + b) for t in tid]
```

På siste linje i koden over, konverterer vi data-punktene fra regresjonen, *y,* tilbake til en eksponensiell funksjonstilpasning for temperaturen

$$T_{fit}(t) = e^{y(t)}$$

Fortsetter på neste side…

Utstyr:
PS-3201 Temperatursensor, SPARKvue eller Capstone.



Måledata fra en oppvarmet temperatursensor, etter å ha trukket fra romtemperaturen.

Biblioteker i Python

Python inneholder mye ferdigskrevet kode som er samlet i **biblioteker**. Om du lærer deg hvor de er og hvordan de brukes, kan kodejobben bli både kort og enkel.

NumPy er et bibliotek som inneholder flere av de mest relevante kodene vi trenger når vi skal jobbe med tall. Vi importerer biblioteket med linjen «import numpy as np» og kaller på funksjoner, som f.eks. en eksponensialfunksjon e^x, med «np.exp(x)».

Det er brukt en håndfull biblioteker i disse eksemplene, samt et egenutviklet bibliotek som vi har kalt «pasco», som forenkler lesing av csv- og txt-filer.

AVKJØLING

python™ PASCO
koding med mening

Her ligger også alt til rette for å verifisere og analysere temperaturdataene ved å simulere (integrere) Newtons avkjølingslov

$$\frac{dT}{dt} = -k(T - T_0)$$

Siden vi trakk fra romtemperaturen fra dataene i SPARKvue, før vi lastet de over til Python, kan vi sette $T_0 = 0$ og få en enda enklere simuleringsjobb.

```
# SIMULERING
k = 0.01    # Gjett ulike verdier for k
dt = 1.0e0  # Velg ulike verdier for dt
# Beregn antall punkter i simuleringen fra dt
nsim = int((tid[len(tid)-1]-tid[0])/dt) + 1
# Sett første verdiene i tid og temperatur-listene
sim_tid = [];   sim_tid.append(tid[0])
sim_temp = [];  sim_temp.append(temp[0])
# Beregn alle verdiene deretter
for i in range(1,nsim):
    sim_tid.append(tid[0]+i*dt) #Lagre tid
    sim_temp.append(sim_temp[i-1]*(1.0-k*dt)) #Lagre T
```

Helt til sist plotter vi dataene vi har lest fra PASCO-sensorene (temp), kurvetilpasningen (fit_temp) og resultatet av simuleringer over (sim_temp) sammen.

```
# PLOTTING
# Plott data fra capstone som punkter
# Plott kurvetilpasning (temp_fit) som linje
plot(tid,temp,".") #Plott PASCO-data
plot(tid,fit_temp) #Plott kurvetilpasning
plot(sim_tid,sim_temp) #Plott simulering
xlabel("Tid (s)") #Tekst på x-aksen
ylabel("Temp (Celcius)") #Tekst på y-aksen
#Lag en boks i plottet som forklarer hva som er plottet
legend(["PASCO data", "Kurvetilpasning", "Simulering"])
show() #Vis fram plottet
```

Det er noe avvik mellom teori, modell og data i plottet, men i store trekk (og spesielt etter 100 sekunder) viser de at forsøket stemmer godt med teorien. Det er mulig at vi kunne gjettet bedre verdi for *k*, eller forbedret den numeriske løsningen av Newtons avkjølingslov. Men hvordan, det lar vi stå ubesvart, slik at elevene kan komme med forslag på løsninger selv.

Differensiallikninger og integrering

Selv om vi i den siste versjonen av læreplanene, ser at både differensial-likninger og integrering av fått mindre plass i enkelt fag, er det mulig å legge opp til noen enkle forsøk som løser problemer ved hjelp av enkle differensiallikninger og integraler. For tross alt er disse, i all sin enkelthet, uttrykk for en forandring og en sum, som vi ønsker å beregne for en serie med flere tidssteg eller datapunkter.

I dette eksempelet, summerer vi bidragene fra en forenklet Newtons avkjølingslov:

$$\Delta T = -kT \cdot \Delta t$$

Som vi skriver om og regner ut på formen

$$T_{t+1} = \Delta T + T_t = T_t(1 - k\Delta t)$$

for hvert tidssteg ΔT.