Av: Ellen E & Geir Magne Fløe

Opplegg 13 – **Modellering med luftmotstand**

Utledning av uttrykk

Dersom du slipper et objekt, f.eks. en ball, vil det påvirkes av tyngdekraften G og luftmotstanden L. I et slikt tilfelle vil vi få følgende uttrykk for akselerasjonen (positiv retning er nedover):

$\Sigma F = G - L$	Newtons 2. lov
$G - L = ma$	
$mg - kv^2 = ma$	Setter $L=kv$^2
$a = g - kv^2 / m$	Deler på m

Programmering

Vi starter med å definere grunnleggende størrelser og initialbetingelser. Deretter deler vi tiden opp i små tidsintervaller, oppretter matriser og setter initialbetingelsene inn på første plass i matrisene. Til sist bruker vi en for-løkke med Eulers metode og plotter grafene.

Puslespill

Sett sammen kodebitene til et ferdig program.

```
v[0] = v0
s[0] = s0
t[0] = t0
```

→ Setter initialbetingelser inn i matrisene

```
from pylab import *

g = 9.81
m = 1.0
k = 0.1
```

→ Definerer grunnleggende variabler.

```
plot(t,v)
xlabel("Tid (s)")
ylabel("Fart (m/s)")
grid()
title("Fartsgraf")
figure()

plot(t,a)
xlabel("Tid (s)")
ylabel("Akselerasjon (m/s)")
grid()
title("Akselerasjonsgraf")
```

→ Plotter grafene.

```
a = zeros(N)
v = zeros(N)
t = zeros(N)
s = zeros(N)
```

→ Oppretter matriser

```
N = 1000
tid = 4
dt = tid/N
```

→ Deler tiden opp i små tidsintervaller.

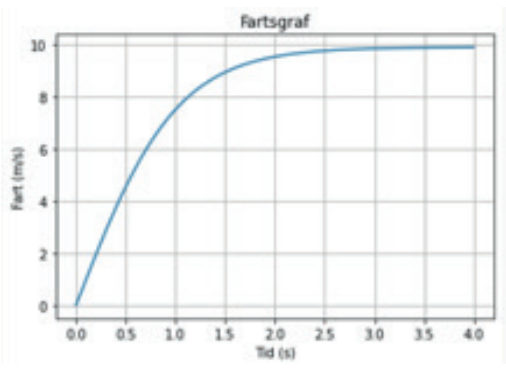
```
v0 = 0
s0 = 0
t0 = 0
```

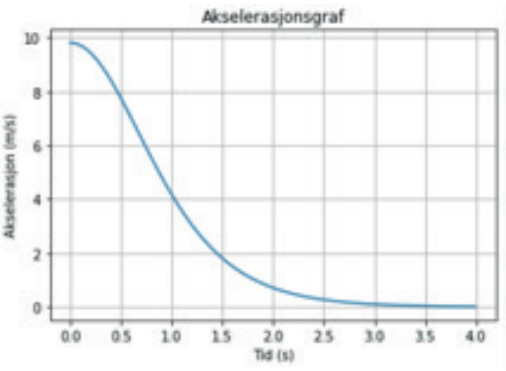
→ Setter initialbetingelser.

```
for i in range(N-1):
    a[i]=g-k/m*v[i]**2
    s[i+1]=s[i]+v[i]*dt+0.5*a[i]*dt**2
    v[i+1]=v[i]+a[i]*dt
    t[i+1]=t[i]+dt

a[N-1]=g-k/m*v[i]**2
```

For-løkke med Eulers metode. Her beregnes først akselerasjonen som har indeks i. Deretter beregnes de øvrige størrelsene som ligger et tidssteg foran, og har indeksen i+1. Uttrykkene her er bevegelsesformlene fra fysikken og er basert på verdiene til s, v, og t for indeksen i. I aller siste linje fyller vi ut siste plass i akselerasjons-matrisen.



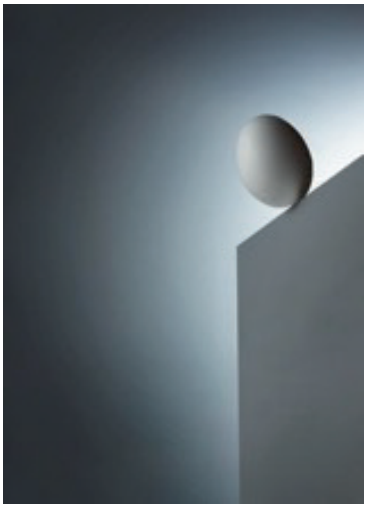


Redd det fallende egget!

- mål tid og akselerasjon med micro:bit og lagre til fil

Oppgave

Lag noe som forhindrer egget fra å bli knust når det slippes fra ca. 3 meters høyde. Mål tiden og akselerasjonen mens egget faller ved å feste en micro:bit som lagrer data til en fil, på egget. Dette datasettet med tid og akselerasjon skal dere bruke for å modellere eggets fart ved programmering.



Fase 1: Hvordan redde egget? Søk gjerne på «egg drop challenge» for tips.

Fase 2: Det er viktig at dere er åpne for alle slags ideer og ikke er for kritiske, da kan nyttige forslag bli avfeid for tidlig.

1. Tenk selv først og tegn gjerne skisser fra forskjellige vinkler.
2. Forklar ideen din for de andre på gruppa. Bruk gjerne skissene i forklaringen.
3. Hele gruppa diskuterer de ulike ideene, og lager en felles plan for bygging.

Fase 3: Gjennomfør planen deres for å lage anti-eggeknuseren og lag programmet for å måle akselerasjonen og lagre på micro:biten. Se tips på side 3.

Gjennomfør fase 4 – 7.

Programmeringsoppgave med målte verdier

Dere skal bruke de målte verdiene for tiden, og den tilhørende akselerasjonen for å finne eggets fart mens det faller.

1. Hvordan skal dere gjøre om akselerasjonsmålingene til riktig enhet, m/s^2 ?
2. Modeller eggets fart og lag en fartsgraf – se tips på side 4.
3. Hvilken momentanfart skulle egget hatt ved bakken ifølge denne modellen?
4. Beregn hvilken fart egget skulle hatt ved bakken ifølge energibevaring i tyngdefelt.
    a. Hvilke antagelser ligger til grunn for formelen dere brukte?
    b. Hvorfor blir denne verdien så forskjellig fra den programmet beregner?

Programmeringsoppgaver – teoretisk modellering

1. Lag et program som modellerer akselerasjonen til egget når det faller, og plotter resultatet som en graf – se tips på forrige side.
2. Inkluder de målte verdiene for akselerasjonen i samme graf og diskuter hvorfor de er ulike.
3. Utvid programmet til å modellere farten til egget, og plott resultatet som en fartsgraf.
4. Inkluder fartsverdiene dere beregnet i programmeringsoppgaven der dere brukte de målte akselerasjonsverdiene, i samme grafen.
    a. På hvilken måte er grafene forskjellige?
    b. Hva kunne dere ha gjort eksperimentelt for at fartsgrafene skulle blitt likere hverandre? Er det mulig?

Skrive til fil på micro:bit med Python Mu

I slutten av kapittel 9 brukte vi datasett i form av tekstfiler for å lage matematiske modeller. Hadde det ikke vært kult om vi kunne fått micro:biten til å lage slike filer av sine målinger? Da kan vi bruke micro:biten til å samle inn data over en viss periode, uten å være koblet til en datamaskin. Datasettet blir lagret i micro:biten som en .txt-fil, og vi kan legge den over i datamaskinen, før vi plotter resultatene eller lager en matematisk modell med Python.

Det aller første vi må gjøre, er å opprette en tekstfil, det vil i en fil som slutter på .txt, og det er enklest å gjøre i Mu. Lagre den samme i samme mappe som resten av dine Python Mu-filer.

Deretter må den fila du har laget, kopieres over på micro:biten. Dette gjør du på samme måten som for sonaren.

```
with open ('filnavn.txt', 'w') as fil:
```

Åpner fila som heter filnavn.txt og tildeler den til et objekt som vi kaller fil. Siden vi har 'w' (write) i kommandoen gjøres det klar til å skrive inn i fila.

```
with open ('filnavn.txt', 'r') as fil:
```

Tilsvarende som linja over, men siden vi har 'r' (read) i kommandoen gjøres det klar til å lese fila. Det er valgfritt å ta med 'r'.

```
fil.write(y+'\n')
```

Skriver variabelen y til fila vår. +'n' gjør at det blir et linjeskift etter hvert tall, og det må vi ha for at python skal kunne lese filene.
Må stå med innrykk.

```
y = str(x)
```

Gjør om variabelen x til en streng (tekstverdi).
Må stå med innrykk.

```
print(z)
```

Skriver variabelen z til skjermen på PC. Må IKKE stå med innrykk.

```
z = fil.read()
```

Leser det som står i fila vår, og legger det inn i variabel z. Må stå med innrykk.

Da gjenstår det å lage selve programmet. Under er et eksempel der micro:biten måler temperaturen og legger målingen i en .txt-fil ved navn datasett.txt. Det kan være greit å kunne sjekke hvordan datasett.txt-fila ser ut uten å overføre til datamaskinen og åpne den, derfor er dette også med i puslespill-oppgaven.

Puslespill-oppgave

```
print(innhold)
```

```
t = temperature()
```

```
while not button_a.is_pressed():
```

```
sleep(1000)
```

```
fil.write(str(t)+'\n')
```

```
with open('datasett.txt') as fil:
```

```
with open('datasett.txt', 'w') as fil:
```

```
innhold = fil.read()
```

```
from microbit import *
```

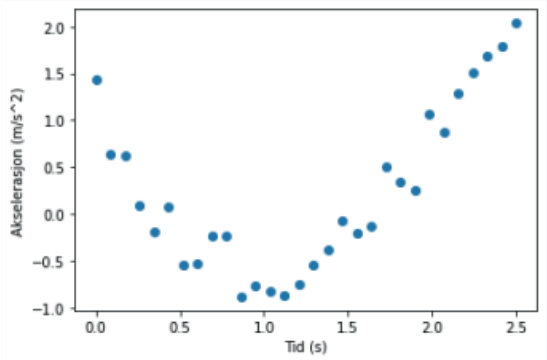
Ekstraoppgave

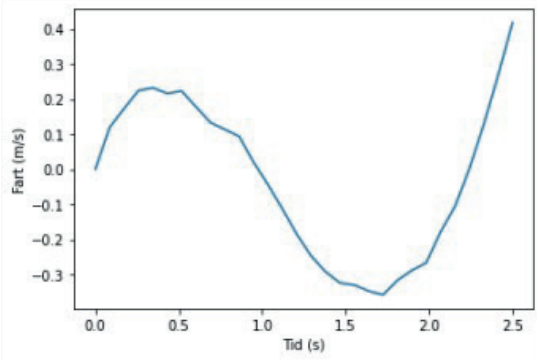
Lag et program som bruker en sensor du kobler på micro:biten for å samle inn data som du lagrer i en fil på micro:biten. Hva må du forandre på?
Hva må du forandre i programmet for å lage en fil med to kolonner der den første kolonnen er hvor lang tid det har gått før målingen er gjort?

Modellering av fart med målt akselerasjon

Dersom vi har et datasett med målte verdier for akselerasjonen, kan vi bruke disse verdiene for å beregne farten i hver av punktene. Vi vil da få grafer som vist i figurene nedenfor. Figuren til venstre viser de målte datapunktene, mens figuren til høyre viser farten. Denne har vi beregnet numerisk.

For å få til dette, må man importere et datasett med akselerasjonsverdier (a) og tilhørende tider (t) i to lister. Deretter bruker vi en løkke med bevegelsesformelen $v\_(i+1)=v\_i+at$ for å beregne fartene.





Ved å bruke kodebitene under, kan man lage et program som plotter akselerasjonen og farten.

```
figure()
plot(t,v)
xlabel("Tid (s)")
ylabel("Fart (m/s)")
```

Plotter fartsgrafen. Funksjonen «figure» brukes når man allerede har plottet en graf og ønsker å åpne et nytt grafvindu

Vi deler opp den totale tiden i små tidssteg og kaller et slikt tidssteg «dt».

```
dt = t[N-1] / N
```

```
scatter(t,a)
xlabel("Tid (s)")
ylabel("Akselerasjon (m/s^2)")
```

```
N = len(t)
```

Måler lengden av lista «t», dvs teller antall observerte data

Plotter akselerasjonen. «Scatter» brukes når man vil ha punkter, ikke sammenhengende graf.

```
v = zeros(N)
```

Oppretter liste for farter.

```
for i in range(N-1):
    v[i+1] = v[i] + a[i]*dt
```

Beregner farter ved hjelp av ei løkke.

Diskusjonsoppgaver

1. Sammenlign figurene. Hva skjer med grafen til farten når akselerasjonen er positiv? Og negativ?
2. Hva er akselerasjonen i topp- og bunnpunktet til farten? Hvorfor?