Forelesningsnotater, FYS1100 uke 9

Joakim Bergli

1 Rakettlikningen

Bevaring av driv gir oss en fin måte å beskrive en situasjon der massen til et legeme forandrer seg med tida ved at noe masse kastes ut eller leges til, som for eksempel en rakett.

1.1 Utledning av rakettlikningen

Legemet vi følger har massen $m(t)$ og farta $\vec{v}(t)$. Vi endrer massen med en liten mengde Δm (for en rakett når massen avtar er $\Delta m < 0$. Drivet før dette skjer er

$$\vec{p}(t) = m\vec{v}.$$

Her skriver vi ikke $m(t)$ og $\vec{v}(t)$ på høyre side selv om det er det vi mener. Det blir lettere å lese likningene på denne måten, men vi må huske at vi mener disse størrelsene ved et bestemt tidspunkt, og at de forandrer seg. Etter at massen er endret er

$$\vec{p}(t + \Delta t) = (m + \Delta m)(\vec{v} + \Delta\vec{v}) - \Delta m\vec{u}$$

der $\vec{u}$ er farta til den massen som sendes ut. Dermed er endringen i driv

$$\Delta\vec{p} = \vec{p}(t + \Delta t) - \vec{p}(t) = m\Delta\vec{v} + \Delta m\vec{v} + \Delta m\Delta\vec{v} - \Delta m\vec{u} \approx m\Delta\vec{v} + \Delta m\vec{v} - \Delta m\vec{u}.$$

Leddet $\Delta m\Delta\vec{v} \approx 0$ siden det er et produkt av to små størrelser. Vi har da når $\Delta t \to 0$ at summen av eksterne krefter er

$$\vec{F}_{ext} = \frac{dp}{dt} = m\frac{d\vec{v}}{dt} + (\vec{v} - \vec{u})\frac{dm}{dt}.$$

1.2 Eksempel: En rakett

Her er det ingen ytre kraft, $\vec{F}_{ext} = 0$. Vi lar bevegelsen være i en dimensjon, og antar at gassen som sendes ut av raketten har en fast relativ hastighetv_{rel} i forhold til raketten, så $v - u = v_{rel}$ og denne varierer ikke med tida. Da er

$$m\frac{dv}{dt} = -v_{rel}\frac{dm}{dt}$$

eller

$$\frac{dv}{dt} = -v_{rel}\frac{1}{m}\frac{dm}{dt}.$$

Vi integrerer begge sider og får

$$\int_0^t dt'\frac{dv}{dt'} = -v_{rel}\int_0^t dt'\frac{1}{m}\frac{dm}{dt'} = -v_{rel}\int_{m_0}^{m_1}\frac{dm}{m}.$$

Der m_0 er startmassen og m_1 er sluttmassen. Dermed er

$$v(t) = v(0) - v_{rel} \ln \frac{m_1}{m_0}.$$

Husk at $m_1 < m_0$ så $\ln \frac{m_1}{m_0} < 0$ og sluttfarta blir større enn startfarta.

1.3 Eksempel: Sand i vogn

Dette er en utvidelse av eksempel 12.7.1 i læreboka. En jernbanevogn beveger seg rett fram mens det lastes sand oppi. Sanda faller ned med en konstant rate $R = \frac{dm}{dt}$ samtidig som den trekkes med en kraft F.

1.3.1 Konstant fart

I eksempelet i boka er det bare funnet hvor stor F må være for at farta skal være konstant. Da er

$$F = m\frac{dv}{dt} + v\frac{dm}{dt} = m\frac{dv}{dt} + vR.$$

Konstant fart betyr $\frac{dv}{dt} = 0$, så krafta må være $F = vR$.

1.3.2 Oppbremsing uten ytre kraft

Her ser vi på $F = 0$, og en startfart $v(0) = v_0$.

$$m\frac{dv}{dt} + vR = 0.$$

Her er det viktig å huske at $m = m(t)$ er en funksjon av tida og ikke konstant. Hvis massen i starten er $m(0) = m_0$ er $m = m_0 + Rt$. Likninga er separabel og vi har

$$\frac{1}{v}\frac{dv}{dt} = -\frac{R}{m} = -\frac{R}{m_0 + Rt}.$$

Dermed har vi

$$\int_{v_0}^{v} \frac{dv'}{v'} = -R\int_0^t \frac{dt'}{m_0 + Rt'}$$

som gir

$$\ln \frac{v}{v_0} = -\ln \frac{m_0 + Rt}{m_0} = -\ln\left(1 + \frac{Rt}{m_0}\right).$$

Dermed er farta som funksjon av tida

$$v(t) = \frac{v_0}{1 + \frac{Rt}{m_0}}.$$

Når $t \to \infty$ blir $\lim_{t\to\infty} v(t) = 0$, så vogna stoppes opp fordi drivet er konstant, men massen øker og dermed må fara avta.

1.3.3 Hvordan varierer farta når $F \neq 0$?

Vi kan også finne farta som funksjon av tida hvis krafta ikke er akkurat den som gir konstant fart.

$$F = Rv + m\frac{dv}{dt}, \qquad m = m_0 + Rt. \tag{1}$$

Så lenge F er konstant er denne likninga separabel.

Alternativ 1: Løsning ved separasjon

Vi kan skrive om likninga til

$$\frac{R}{F - Rv}\frac{dv}{dt} = \frac{R}{m_0 + Rt}$$

og integrere

$$\int_{v_0}^{v} \frac{Rdv'}{F - Rv'} = \int_0^t \frac{Rdt'}{m_0 + Rt'}$$

$$-\ln\frac{Rv - F}{Rv_0 - F} = \ln\left(1 + \frac{Rt}{m_0}\right)$$

$$v(t) = \frac{F}{R} + \frac{v_0 - F/R}{1 + \frac{Rt}{m_0}}. \tag{2}$$

Alternativ 2: Løsning ved integrerende faktor

Vi kan skrive om likning (1) som

$$\frac{dv}{dt} + \frac{R}{m}v = \frac{F}{m}$$

Vi regner ut

$$\int \frac{R}{m_o + Rt}dt = \ln|m_0 + Rt|$$

Den integrerende faktoren vi må gange med er

$$e^{\int p(t)dt} = m_0 + Rt$$

Det gir

$$(m_0 + Rt)\frac{dv}{dt} + Rv = F$$

som vi ser er lik

$$\frac{d}{dt}\left[(m_0 + Rt)v\right] = F.$$

Dermed er

$$(m_0 + Rt)v = \int Fdt = Ft + C \tag{3}$$

Hvis initialbetingelsen er $v(0) = v_0$ er $C = m_0v_0$. Dermed er

$$v(t) = \frac{Ft + m_0v_0}{Rt + m_0}.$$

Dette er lik det vi fant over i likning (2) (sjekk det!).

For å sjekke om dette virker rimelig kan vi se at det gir riktig enhent (m/s). Vi kan også sjekke at $v(0) = \frac{m_0 v_0}{m_0} = v_0$ som det skal. Vi kan også se hva som skjer etter lang tid. Da vil konstantleddene i teller og nevner være ubetydlige og vi har

$$v(t) \approx \frac{Ft}{Rt} = \frac{F}{R}.$$

Dette er nettopp den farta som vi fant i avsnitt 1.3.1 at er konstant med den gitte krafta.

Vi får altså samme svar med begge metodene, men dersom krafta ikke er kosntant, men en gitt funksjon av tida, $F(t)$, vil ikke likninga være separabel, og vi kan ikke bruke den metoden. Med integrerende faktor vil vi kunne finne en løsning så lenge vi klarer å løse integralet i likning (3).

1.4 Eksempel: Vanndråpe i fuktig luft

En vanndråpe har massen $m(t)$ som endrer seg med tida ettersom dråpa vokser. Den faller gjennom fuktig luft inne i en sky der den treffer mange små vanndråper som fester seg til den hvis de blir truffet. Vi antar at disse små dråpene ligger i ro før dråpa treffer dem. Hvordan øker farta som funksjon av tida? Ytre kraft er tyngden $G = mg$ og rakettlikninga gir

$$mg = m\frac{dv}{dt} + v\frac{dm}{dt}. \tag{4}$$

Hvis tettheten av små vanndråper er slik at det er en gjennomsnittlig massetetthet av vann i lufta på ρ har vi at

$$\frac{dm}{dt} = \rho A v$$

Der $A = 4\pi r^2$ er tverrsnittsarealet og v farta og r radien til den fallende dråpa. Vi kan relatere dette til massen ved at volumet er $V = \frac{4}{3}\pi r^3$ og massen er $m = \rho_v V = \frac{4}{3}\pi \rho_v r^3$ hvis ρ_v er massetettheten til vann. Dermed er

$$r = \left(\frac{4}{3}\pi\rho_v\right)^{-1/3} m^{1/3}$$

og vi får at

$$\frac{dm}{dt} = Km^{2/3}v, \qquad K = 4\pi\rho\left(\frac{4}{3}\pi\rho_v\right)^{-2/3}. \tag{5}$$

Går vi tilbake til likning (4) får vi

$$\frac{dv}{dt} = g - Kv^2 m^{-1/3}. \tag{6}$$

Likningene (5) og (6) er et sett av koblede difflikninger som må løses sammen fordi m inngår i likninga for v og motsatt. De er ikke-lineære fordi m og v ikke bare inngår lineært. Det er antagelig umulig å løse disse analytisk (si fra hvis noen klarer det!), så vi vil heller løse dem numerisk. Dette er neste tema.

2 Eulers metode

Vi ser på en likning av typen

$$\dot{x} = f(x,t), \qquad x(0) = x_0 \tag{7}$$

der $x = x(t)$ er den ukjente funksjonen og x_0 er initialbetingelsen. $f(x,t)$ er en kjent funksjon som gir den tidsderiverte $\dot{x}$ når vi kjenner x of t. Hvis vi tilnærmer den deriverte

$$\dot{x} \approx \frac{x(t+\Delta t) - x(t)}{\Delta t}$$

får vi

$$x(t+\Delta t) \approx x(t) + f(x,t)\Delta t. \tag{8}$$

Hvis vi starter fra x_0 ved $t = 0$ kan vi bruke denne likninga gjentatte ganger for å finne en tilnærming til $x(t)$. Vi definerer

$$t_i = i\Delta t, \qquad x_i = x(t_i)$$

og kan da skrive

$$x_{i+1} = x_i + f(x_i, t_i)\Delta t.$$

Dette kaller vi Eulers metode for løsning av difflikninger.

2.1 Eksempel

Vi ser på likninga

$$\dot{x} = x\cos t, \qquad x(0) = x_0. \tag{9}$$

La oss først løse den analytisk, så vi kan sammenlikne den numeriske løsninga med den analytiske. Likninga er separabel, og vi får

$$\int_{x_0}^{x} \frac{dx'}{x} = \int_0^t \cos t' dt'.$$

Løser vi integralene får vi

$$\ln \frac{x}{x_0} = \sin t$$

eller

$$x(t) = x_0 e^{\sin t}.$$

Vi lager et program som implementerer Eulers metode for dette eksemplet.

```
# Eulers metode for \dot x = x cos t. Vi sammenlikner numerisk og analytisk losning.

import numpy as np
import matplotlib.pyplot as plt


x0 = 1
```

```
dt = 0.1
tMax = 20

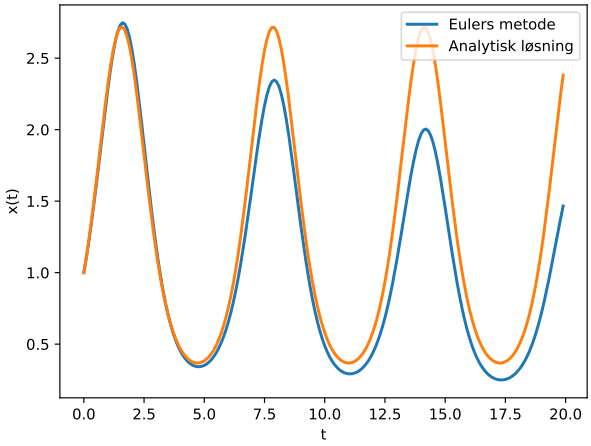
n = int(tMax/dt)
t = np.zeros(n)
dx = np.zeros(n)
x = np.zeros(n)

x[0] = x0
t[0] = 0

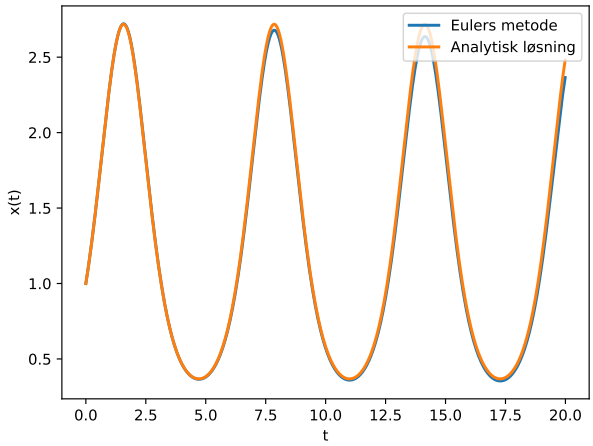
for i in range(n-1):
    t[i+1] = t[i] + dt
    dx[i] = np.cos(t[i])*x[i]
    x[i+1] = x[i] + dx[i]*dt

plt.plot(t,x,linewidth=2, label='Eulers metode')
plt.plot(t,x0*np.exp(np.sin(t)),linewidth=2, label='Analytisk løsning')
plt.legend(loc='upper right')
plt.xlabel('t')
plt.ylabel('x(t)')
plt.savefig('euler.pdf')
plt.show()
```

Med steglengden Med $\Delta t = 0.1$ får vi



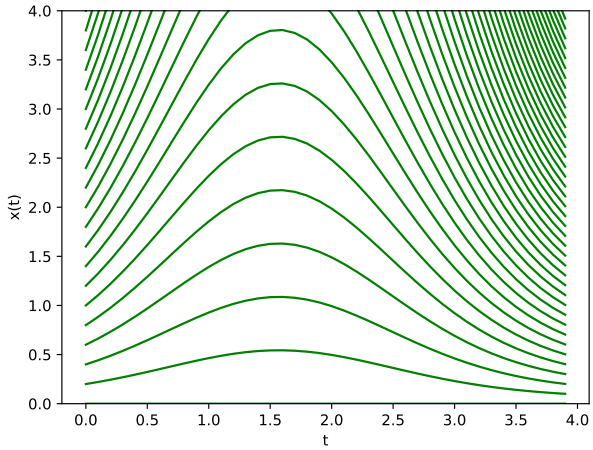
Vi ser at den numeriske løsninga følger den analytiske en stund, men blir etter hvert mer og mer gal. Vi kan gjøre dette bedre ved å redusere steglengden, men det gjør samtidig at vi trenger flere integrasjonssteg for å nå samme tid, og dermed tar beregninga lengre tid. Med $\Delta t = 0.01$ blir resultatet



Vi ser at feilen blir mindre og den numeriske løsninga er nær den analytiske lengre.

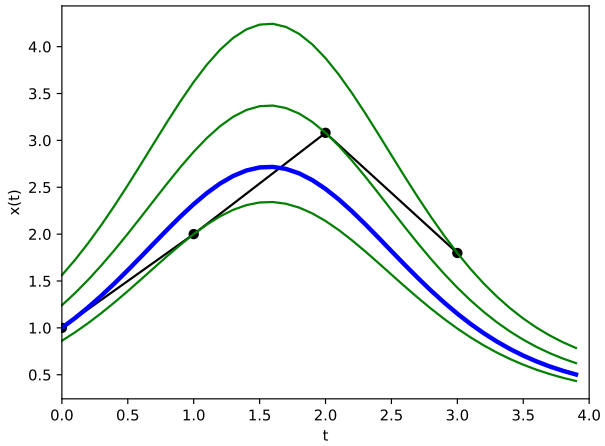
2.2 Geometrisk bilde

Vi kan gi et geometrisk bilde av hvordan Eulers metode tilnærmet følger en løsning til en difflikning. For å forstå det må vi huske på at en difflikning ikke definerer en entydig løsning i seg selv, men en hel familie av løsninger. Når vi velger en initialbetinglese, plukker vi ut den bestemte løsninga som stemmer med initialbetinglesen. Figuren under viser forskjellige løsninger til likninga (**??**) med forskjellige initialbetingelser.



Gjennom ethvert punkt (t, x) går det en entydig bestemt løsning.

Tilnærmingen i likning (**??**) innebærer at vi følger tangenten til funksjonen $x(t)$ i t i hele intervallet $[t, t + \Delta t]$. Så lenge funksjonen ikke er lineær vil den bøye av fra tangenten, og metoden gir en feil. For å forstå hvordan dette utvikler seg fra steg til steg må vi huske at difflikninga har mange løsninger avhengig av initialbetingelsen x_0. Når vi gjør en feil i første steg treffer vi isteden en løsning som svarer til en annen initialbetingelse, og i neste steg følger vi tangenten til denne funksjonen isteden. Slik fortsetter vi slik vist under, der den blå kurven er den løsninga med riktig initialbetingelse mens de grønne kurvene er andre løsninger som vi treffer underveis.



2.3 Feil som funksjon av Δt

Vi så over at feilen blir mindre når vi gjør Δt mindre. Samtidig tar det lengre tid å gjøre beregninga når Δt er liten, for vi må gjøre flere tidssteg for å nå samme sluttid. Derfor er det best å bruke Δt så stor som mulig men som samtidig gir nøyaktig nok svar. Vi skal gjøre en liten analyse av hvordan feilen avtar med Δt, og sjekke numerisk at dette stemmer.

Hvis $x(t)$ er en løsning av difflikninga (**??**) kan vi bruke Taylors formel med restledd

$$x(t+\Delta t) = x(t) + \dot{x}(t)\Delta t + \frac{1}{2}\ddot{x}(u)\Delta t^2 = x(t) + f(x,t)\Delta t + \frac{1}{2}\ddot{x}(u)\Delta t^2$$

med $u \in [t, t+\Delta t]$. Eulers metode med $x_i = x(t)$ og $t_i = t$ tilnærmer $x(t+\Delta t)$ med

$$x_{i+1} = x_i + f(x_i, t_i)\Delta t.$$

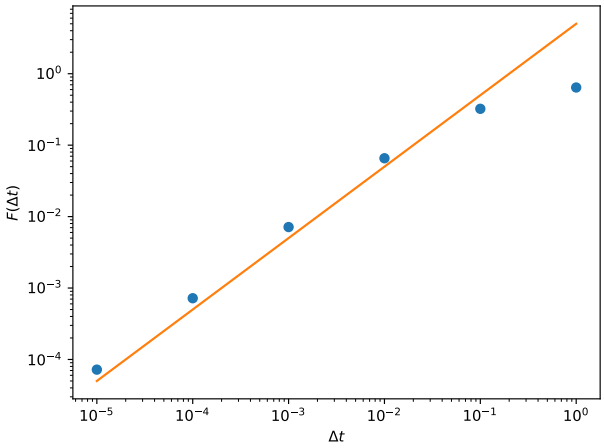
Feilen i ett tidssteg er da

$$F_1(\Delta t) = x(t+\Delta t) - x_{i+1} = \frac{1}{2}\ddot{x}(u)\Delta t^2.$$

For hvert tidssteg er altså feilen proporsjonal med Δt^2. Hvis vi numerisk løser likninga fra starttida $t = 0$ til en sluttid T er antallet steg vi gjør $N = T/\Delta t$. Dermed forventer vi at feilen ved tida T er proporsjonal med

$$F_T(\Delta t) \propto N\Delta t^2 = T\Delta t.$$

Vi kan sjekke at dette stemmer for eksempelet vårt over, likning (**??**). Vi bruker $x_0 = 1$ og ser på løsninga ved tida $T = 80$ ved ulike Δt (koden ligger i fila `eulerVariabelDt.py`)



De blå punktene er feilen i simuleringene sammenliknet med den eksakte løsninga. Den rette linja er ei linje med stigningstall 1 i de logaritmiske aksene. Husk at om vi har en potenssammenheng

$$F_T(\Delta t) = A\Delta t^n$$

har vi for logaritmene

$$\ln F_T(\Delta t) = \ln A + n \ln \Delta t$$

slik at det blir ei rett linje med stigningstall n. I vårt tilfelle forventer vi at $n = 1$, og vi ser at dette stemmer siden punktene følger samme stigning om linja.

3 Implisitt Eulers metode

Denne metoden bruker vi ikke noen gang for seg selv, men den inngår i Euler-Cromer metoden (avsnitt **??**), så det er nyttig å se litt hvordan den virker.

Eulers metode bruker den deriverte i startpunktet, $f(x_i, t_i)$, i hele intervallet $[t_i, t_{i+1}]$. Men vi kunne isteden tenke at vi bruker den deriverte i sluttpunktet, $f(x_{i+1}, t_{i+1})$. Dette gir følgende formel for å regne ut en tilnærmet løsning

$$x_{i+1} = x_i + f(x_{i+1}, t_{i+1})\Delta t.$$

Ulempen med denne formen er at når vi har regnet ut x_i vet vi ikke x_{i+1} aom vi trenger for å regne ut $f(x_{i+1}, t_{i+1})$. Uttrykket over er altså en likning som vi må løse for å finne x_{i+1}. Derfor kalles denne metoden implisitt Eulers metode. Hvis vi ser på eksempelet fra likning (**??**) har vi

$$x_{i+1} = x_i + x_{i+1} \cos t_{i+1}\Delta t$$

som vi løser og finner

$$x_{i+1} = \frac{x_i}{1 - \cos t_{i+1}\Delta t}.$$

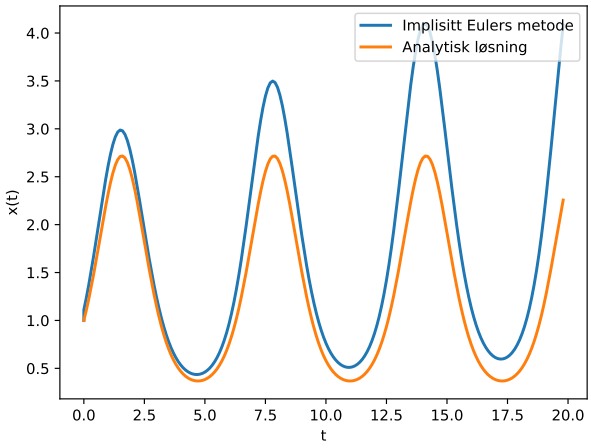
Vi implementerer dette ved å bytte ut kodelinjene

```
dx[i] = np.cos(t[i])*x[i]
x[i+1] = x[i] + dx[i]*dt
```

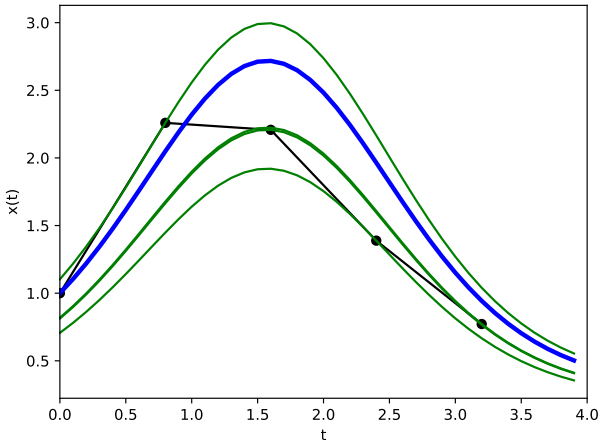
med

```
x[i+1] = x[i]/(1-np.cos(t[i+1])*dt)
```

Dette gir



Vi ser at denne metoden har samme type svakhet som Eulers metode, men den tenderer til å feile i motsatt retning. Igjen kan vi forstå geometrisk hvordan den virker. Her finner vi et punkt på en kurve slik at tangenten går tilbake til det forrige punktet i sekvensen.



4 Eulers midtpunktmetode

Vi har sett at vi kan tilnærme den deriverte i intervallet $[t, t + \Delta t]$ på to forskjellige måter. Om vi bruker

$$\dot{x} \approx \frac{x(t + \Delta t) - x(t)}{\Delta t}$$

får vi Eulers metode, mends om vi bruker

$$\dot{x} \approx \frac{x(t) - x(t - \Delta t)}{\Delta t}$$

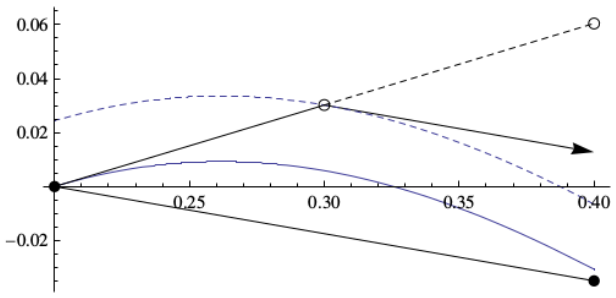
får vi implisitt Eulers metode. Begge disse har feil av orden Δt^2. Men det finnes en måte å tilnærme den deriverte på der feilen er av orden Δt^3. Dette får vi ved å ta den symmetriske kombinasjonen

$$\dot{x} \approx \frac{x(t + \Delta t/2) - x(t - \Delta t/2)}{\Delta t}.$$

Geometrisk svarer det til at vi bruker den deriverte i punktet $t + \Delta t/2$ for hele intervallet $[t, t + \Delta t]$. Intuitivt er det ikke vanskelig å tro at dette gir en bedre tilnærming. Vi kunne altså ønske å regne ut

$$x_{i+1} = x_i + f(x(t_i + \Delta t/2), t_i + \Delta t/2)\Delta t.$$

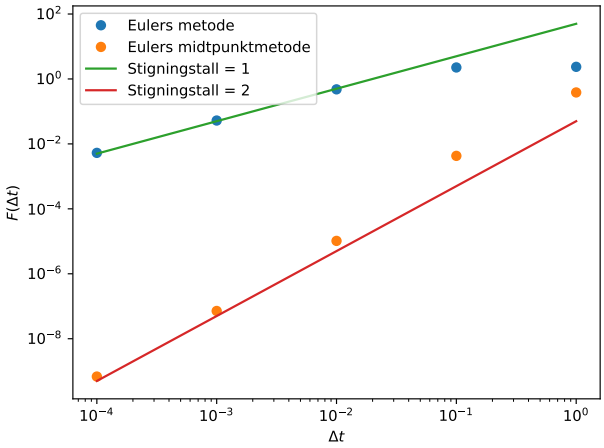
Problemet er at vi ikke kjenner $x(t_i + \Delta t/2)$, da måtte vi jo løse difflikninga først. Dette tilnærmer vi ved å bruke Eulers metode med tidssteg $\Delta t/2$ og finne en tilnærming til $x(t_i + \Delta t/2)$. Så bruker vi den deriverte i det punktet isteden



Det gir oss følgende to-stegs prosess

$$x_{i+1/2} = x_i + f(x_i, t_i)\frac{\Delta t}{2}$$
$$x_{i+1} = x_i + f(x_{i+1/2}, t_i + \Delta t/2)\Delta t$$

Man kan vise at dette teoretisk gir en feil $F_T(\Delta t) \propto \Delta t^2$ for hele prosessen. Vi sjekker numerisk at dette stemmer (koden ligger i fila `eulerMidtpunktVariabelDt.py`)



Vi ser at feilen avtar omtrent som Δt^2 som forventet, og at den blir mye mindre enn for Eulers metode.

5 Høyere ordens difflikninger kan omgjøres til sett av koblede første ordens likninger

I mekanikk er det ofte vi trenger å løse andre ordens difflikninger fordi vi ofte kjenner kraften som virker i et punkt, og denne gir oss akselerasjonen som er den andrederiverte av posisjonen. For eksempel har vi en harmonisk oscillator som har likninga

$$\ddot{x} = -\omega^2 x, \qquad x(0) = x_0, \qquad \dot{x}(0) = v_0. \tag{10}$$

Et vanlig tilfelle er at vi starter i ro med maks utslag, slik at $v_0 = 0$, og da er løsninga

$$x(t) = x_0 \cos \omega t.$$

En slik likning kan gjøres om til to koblede førsteordenslikninger ved å betrakte den førstederiverte, $v(t) = \dot{x}(t)$ som en funksjon i seg selv. Likning (**??**) er dermed ekvivalent med

$$\dot{x} = v$$
$$\dot{v} = -\omega^2 x.$$

To slike likninger kan løses numerisk like lett som en. Eulers metode for disse gir uttrykkene

$$v_{i+1} = v_i - \omega^2 x_i \Delta t$$
$$x_{i+1} = x_i + v_i \Delta t.$$

Dette implementerer vi slik

```
# Eulers metode for harmonisk oscillator

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

```
x0 = 2
v0 = 0
dt = 0.05
tMax = 50

# a,v,x er akselerasjon, fart, posisjon

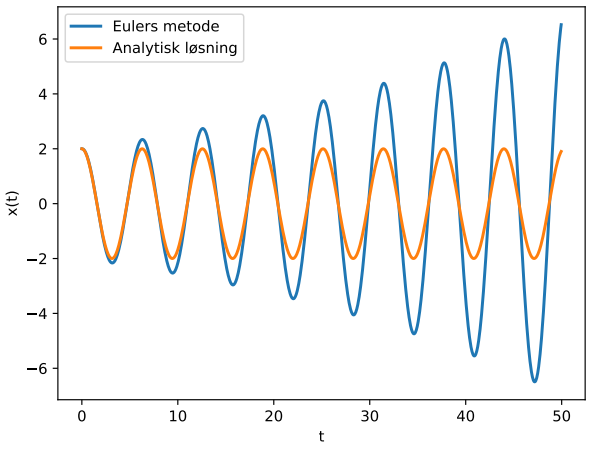
n = int(tMax/dt)
t = np.zeros(n)
a = np.zeros(n)
v = np.zeros(n)
x = np.zeros(n)

x[0] = x0
v[0] = v0

for i in range(n-1):
    t[i+1] = (i+1)*dt
    a[i] = -x[i]
    v[i+1] = v[i] + a[i]*dt
    x[i+1] = x[i] + v[i]*dt

plt.plot(t,x,linewidth=2, label='Eulers metode')
plt.plot(t,x0*np.cos(t),linewidth=2, label='Analytisk løsning')
plt.legend(loc='upper left')
plt.xlabel('t')
plt.ylabel('x(t)')
plt.savefig('HOEuler.pdf')
plt.show()
```

Med steglengden $\Delta t = 0.05$ får vi



Som vi ser følger vi den analytiske løsningen til en viss grad, men amplituden til svingningene øker hele tiden. Ved å gjøre Δt mindre går denne økningen saktere (på bekostning av lengre beregningstid). En økning i amplituden betyr at energien øker. Eulers metode er ikke spesielt god på denne typen mekanikkproblemer fordi den ikke gir bevaring av energi som vi vet er en egenskap ved den eksakte løsningen.

6 Euler-Cromer metoden

Det viser seg at for mange mekanikkproblemer (særlig de som involverer periodiske svingninger av en eller annen type) er det veldig effektivt å bruke en blanding av Eulers og implisitt Eulers metode. Denne kalles ofte Euler-Cromer metoden, og er lik med Euler for å finne farten, men implisitt Euler for å finne posisjonen. Det betyr at vi bruker

$$v_{i+1} = v_i - \omega^2 x_i \Delta t$$
$$x_{i+1} = x_i + v_{i+1} \Delta t.$$

6.1 Eksempel: Harmonisk oscillator

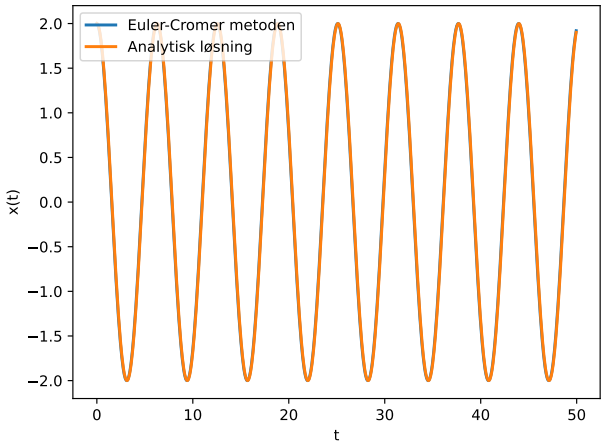
Euler-Cromer metoden er like lett som Eulers metode å implementere, vi bytter ut

```
    x[i+1] = x[i] + v[i]*dt
```

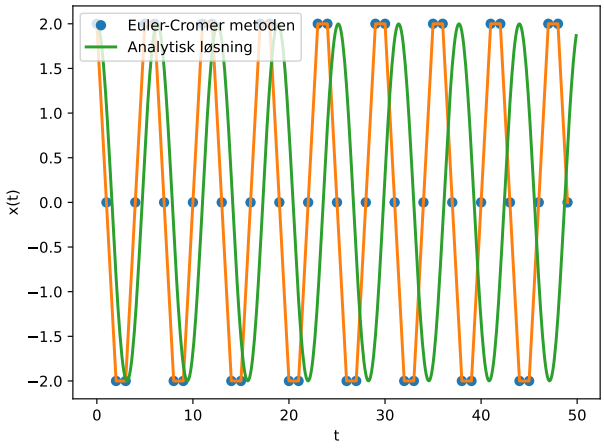
med

```
    x[i+1] = x[i] + v[i+1]*dt
```

Med steglengden $\Delta t = 0.05$ som over får vi

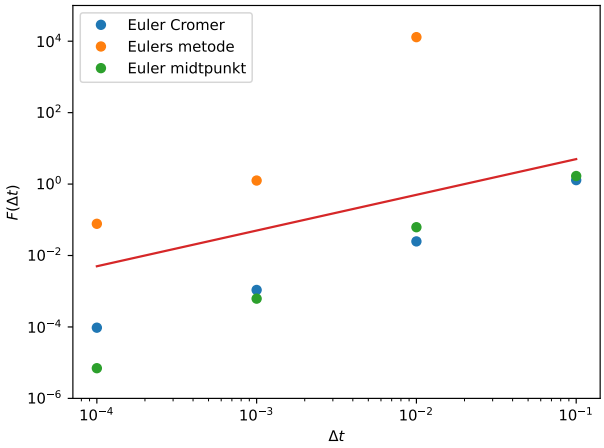


Her kan vi ikke se forskjell på den analytiske og numeriske løsningen. Euler-Cromer metoden er mye bedre enn Eulers metode for denne typen likninger. Man kan vise at den ikke beverer energien eksakt, men at energien ikke vokser ubegrenset slik som med Eulers metode, men har små svingninger om den riktige verdien. Dette gjør at metoden er stabil for mye større steglengder. Hvis vi setter $\Delta t = 1$ (og vi har valgt $\omega = 1$ slik at perioden er 2π, så med $\Delta t = 1$ er det urealistisk å følge bevegelsen særlig godt) får vi:



Vi ser at vi ikke lengre klarer å følge bevegelsen i detalj, og perioden stemmer ikke lengre så godt. Men selv nå er bevegelsen stabil fordi den respekterer energibevaring i snitt.

Vi kan også undersøke hvor stor feil vi får som funksjon av Δt. Her har vi brukt Eulers metode, Eulers midtpunktmetode og Euler-Cromer metoden (koden ligger i fila `HOVariabelDt.py`).



Som vi ser har Euler-Cromer metoden en feil som er proporsjonal med Δt slik som Eulermetoden, men feilen er mye mindre. Eulers midtpunktmetode er en andreordens metode, og feilen er proporsjonal med Δt^2. For små Δt gir dette mindre feil.