

# ELE130 Anvendt matematikk og fysikk i robotprogrammering

## Øving 2

### Introduksjon til simulering i Simulink<sup>1</sup>

Levert av John David McGurk

---

<sup>1</sup>Tilhørende filer: [oving2\\_skallfiler.zip](#) og L<sup>A</sup>T<sub>E</sub>X-mal i [oving2\\_Overleaf.zip](#)

## Om øvingen og innleveringen

- For å bli kjent med Simulink på en effektiv måte, er alle oppgavene i denne øvingen svært nyttige, og de vil fungere som et oppslagsverk for deg.
- For å tilrettelegge for at du skal gjøre flest mulig oppgaver, så skal du laste ned og pakke opp **oving2\_skallfiler.zip** som inneholder:
  - Simulink-skallfilen **oving2.slx** hvor deloppgavene a)-p) implementeres i egne subsystem
  - den ferdige modellen **oving2\_oppg\_q.slx** for deloppgave q)
  - skallfilen **oving2\_data.m** som du må kjøre for 2i), 2o), 2p) og 2q).

Simulink-skallfilene er spesifisert med trapesmetoden `ode2` (Heun) som integrasjonsmetode og tidsskritt  $T_s$  lik 0.01 sekund.

- Resultater i scope kan du overføre til en MATLAB-figur med menyvalget `[File -> Print to Figure]`, som du igjen kan bruke **LagreMinFigur** på, se prosedyre-boksen nedenfor.
- **For å få øvingen godkjent må du gjøre følgende oppgaver:**  
**a), c), d), f), g), i), o), p), q)**
  - Oppgavene **b), e), h), j), k), l), m), n)** er også lærerike og nyttige, men frivillige. Dette er markert i toppteksten.
  - **For å ta eksamen i emnet ELE130 så må denne øving være godkjent. Husk at øvingene må leveres individuelt.**
  - Basis for denne øvingen er **kapittel 4 og 5** i kompendiet.
  - På samme måte som i øving 1 finner du en L<sup>A</sup>T<sub>E</sub>X-mal i filen **oving2\_Overleaf.zip**.

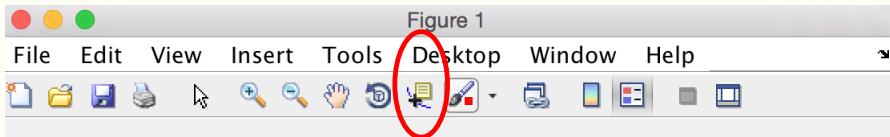
Så langt det er mulig skal du benytte blokker fra Eget bibliotek slik at nyttig informasjon om blokkene vises i blokkdiagrammet. Du skal benytte Scope-blokkene som heter **Scope**, **tynne linjer** eller **Scope, tykke linjer** fra Eget bibliotek til å vise resultatene i. Øvingen og skallfilen er organisert i følgende deler/subsystem:

- “Om Integrator-blokken”, oppgave 2a) - 2e)
- “Om Sine Wave-blokken”, oppgave 2f) - 2h)
- “Om Lookup Table-blokken”, oppgave 2i) - 2l)
- “Litt diverse”, oppgave 2m) - 2p)
- “PID-regulatoren”, oppgave 2q)

I mange av oppgavene skal du inkludere simuleringsresultatet i innleveringen din og samtidig lese av verdier i MATLAB-figuren. Prosedyren for å gjøre dette er som følger:

Prosedyre for å lage figur fra Simulink-scope og deretter lese av verdier

- Velg `[File -> Print to Figure]` fra scopet. Vær klar over at eventuelle avlesningene ved bruk av `Cursor Measurements` ikke blir med i MATLAB-figuren.
- Det å avlese/markere i MATLAB-figuren som er laget med utgangspunkt i et Simulinks scope gjøres ikke på samme måte som i vanlige MATLAB-figuren. Du må nemlig du bruke `Data Cursor`-verktøyet vist i rød ring figuren under. For å avlese mer enn ett punkt, husk å holde inne `[Shift]`-knappen.



**Figur 1:** Trykk på knappen i rød ring for å lese av verdier på grafen i en figur som er eksportert fra Simulink.

- Benytt deretter `LagreMinFigur`-funksjonen til å lage en .pdf av figuren.

## Forskjellig L<sup>A</sup>T<sub>E</sub>X-kode

Under er det gitt litt forskjellig kode som du kan kopiere inn i svarene under de oppgavene du gjør. Husk bare på å oppdatere bokmerkene til nye, unike navn, så slipper du feilreferering.

- Resultatet fra scopet er vist i figur ??.



**Figur 2:** Resultat av blokkskjemaet i modellen.

- Svaret er gitt i ligning (??). For å bruke kommandoen \underline må du bruke pakken `ulem` som `\usepackage{ulem}`.

$$y(25) = \underline{\underline{4.2}} \quad (1)$$

- Eksempel på bruk av `align` er vist i ligning (??).

$$\begin{aligned} y(t) &= \int_0^t U \sin(\omega \cdot \tau) d\tau + y(0) \\ &= -\frac{U}{\omega} \cdot \cos(\omega \cdot \tau) \Big|_0^t + y(0) \\ &= -\frac{U}{\omega} \cdot \cos(\omega \cdot t) - \left( -\frac{U}{\omega} \cos(0) \right) + y(0) \\ &= -\frac{U}{\omega} \cdot \cos(\omega \cdot t) + \frac{U}{\omega} + y(0) \end{aligned} \quad (2)$$

- Du finner også mye L<sup>A</sup>T<sub>E</sub>X-kode i hver av .tex-filene til delxoppgavene.

## Om Integrator-blokken

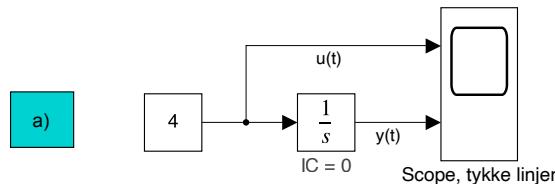
Alle modellene i denne delen av øvingen skal implementeres i subsystemet

**Om Integrator-blokken, oppgave 2a)-2e)** i skallfilen **oving2.slx**, og alle modellene skal benytte integratorblokken som numeriske integrerer et inngangssignal  $u(t)$  i henhold til

$$y(t) = \int_0^t u(\tau)d\tau + y(0) \quad (3)$$

hvor  $y(0)$  er initialverdien. Vær klar over at alle signalene vi jobber med er diskrete som  $\{u_k\}$ , men at vi for enkelhets skyld skriver signalene som kontinuerlige som  $u(t)$ .

- a) I denne deloppgaven er  $u(t)$  en konstant, og du skal relatere simuleringsresultatet til integralregning fra matematikken. Implementer modellen vist under.



Utgangen fra Constant-blokken er  $u(t)$  og utgangen fra Integrator-blokken er  $y(t)$ . La initialverdien til integratorblokken være  $y(0)=0$ . **La simuleringstiden være 25 sekund.**

Svar på følgende spørsmål:

- a1) Bruk ligning (??) og la  $u(t)=4$ . Hva blir det matematiske uttrykket for  $y(t)$  når  $y(0)=0$ ?
- a2) Bruk det matematiske uttrykket til å beregne verdien av  $y(t)$  ved tidspunkt  $t=25$  sekund.
- a3) Simuler modellen og svar samtidig på følgende spørsmål: "Stemmer simuleringsresultatet med svaret du fant i forrige delspørsmål?"  
Ta med simuleringsresultatet i innleveringen ved bruke prosedyren på side ??.
- a4) Endre initialverdien til  $y(0)=10$  og simuler. Hvordan endres det matematiske uttrykket til  $y(t)$ ? Vis at simuleringsresultatet overstemmer med det matematiske uttrykket. Ta med figur i innleveringen.

**Svar**

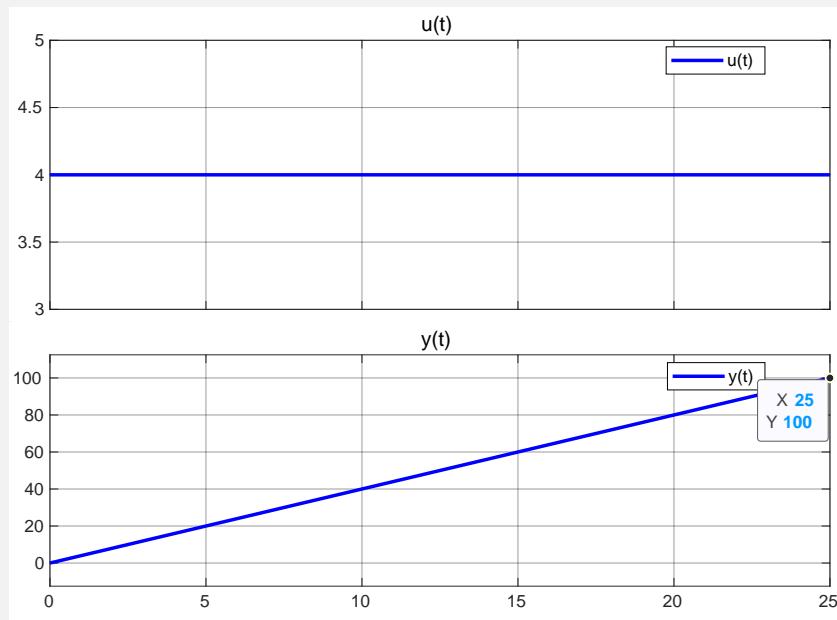
- a1) Med utgangspunkt i ligning (??) og  $u(t)=4$ , er det matematiske uttrykket for  $y(t)$  gitt som

$$\begin{aligned}
 y(t) &= \int_0^t u(\tau)d\tau + y(0) \\
 &= \int_0^t 4d\tau + 0 \\
 &= 4\tau \Big|_0^t \\
 &= 4t - 4 \cdot 0 \\
 &= 4t
 \end{aligned} \tag{4}$$

a2)

$$\begin{aligned}
 y(25) &= 4t \\
 &= 4 \cdot 25 \\
 &= 100
 \end{aligned} \tag{5}$$

- a3) Vi ser på grafen at når  $x = 25$  så er  $y = 100$ , som stemmer med det vi fant ut i deloppgave a2.

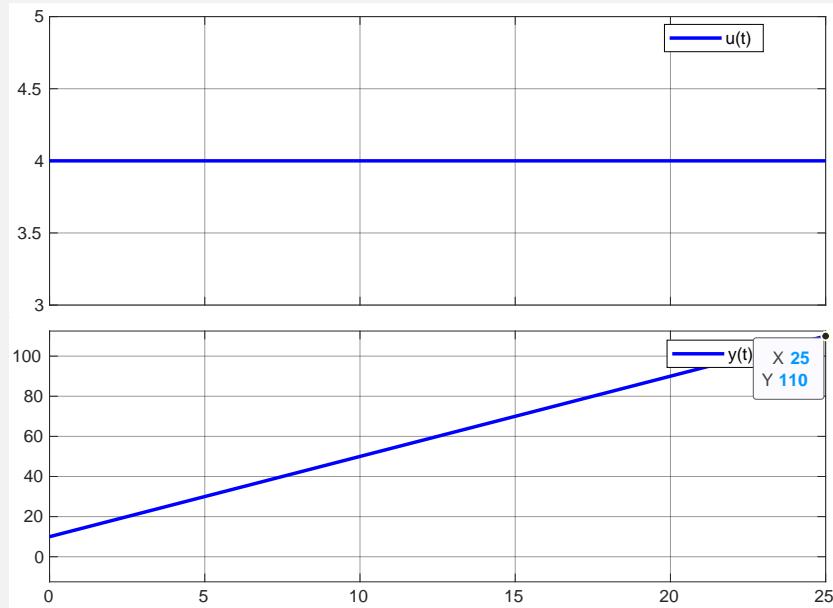


**Figur 3:** Resultat av blokkskjemaet i modellen.

a4)

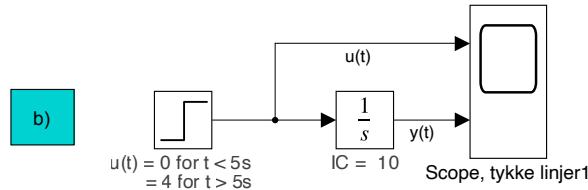
$$\begin{aligned}y(t) &= \int u(t)dt + y(0) \\&= \int 4dt + 10 \\&= 4t + 10 \\y(25) &= 4 \cdot 25 = 10 \\&= 110\end{aligned}\tag{6}$$

Vi ser at dette stemmer med resultatet fra koden i figuren vist under.



**Figur 4:** Resultat av blokkskjemaet i modellen.

- b) Kopier modellen fra a) og bytt ut Constant-blokk med en Step-blokk som går fra 0 til 4 ved  $t=5$  som vist under



Uttrykket for  $u(t)$  er

$$u(t) = \begin{cases} 0 & \text{for } t < 5 \\ 4 & \text{for } t \geq 5 \end{cases} \quad (7)$$

La simuleringstiden fortsatt være 25 sekund.

Svar på følgende spørsmål:

- La initialverdien til integratoren være  $y(0)=10$ . Simuler modellen og ta med simuleringsresultatet i innleveringen din. Hva er verdien av  $y(t)$  i  $t=25$  sekund? Ta med figuren og avlesning ved å bruke prosedyren på side ??.
- Basert på visuell avlesning, hvor stort er arealet under  $u(t)$ ? Hvordan henger denne verdien sammen med verdien av  $y(t)$ ?
- Endre  $u(t)$  til

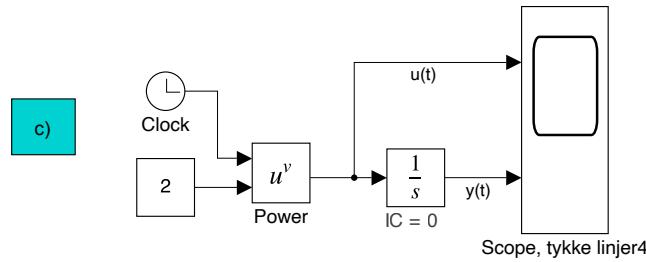
$$u(t) = \begin{cases} -2 & \text{for } t < 5 \\ 2 & \text{for } t \geq 5 \end{cases} \quad (8)$$

Hva blir nå verdien av  $y(t)$  i  $t=25$  sekund? Basert på visuell avlesning, hvor stort er nå arealet under  $u(t)$ ?

### Svar (frivillig)

- b1)
- b2)
- b3)

- c) Kopier modellen fra a) og bygg den som vist under, hvor initialverdien til integratoren er  $y(0)=0$ .



La simuleringstiden fortsatt være 25 sekund.

Svar på følgende spørsmål:

- c1) Hva er det analytiske uttrykket for  $u(t)$  som er modellert?
- c2) Hva blir det analytiske uttrykket for  $y(t)$ ?
- c3) Simuler modell og ta med simuleringsresultatet i innleveringen din ved å bruke prosedyren på side ??.
- c4) Vis/forklar at simuleringsresultatet stemmer overens med det analytiske uttrykket for  $y(t)$ .

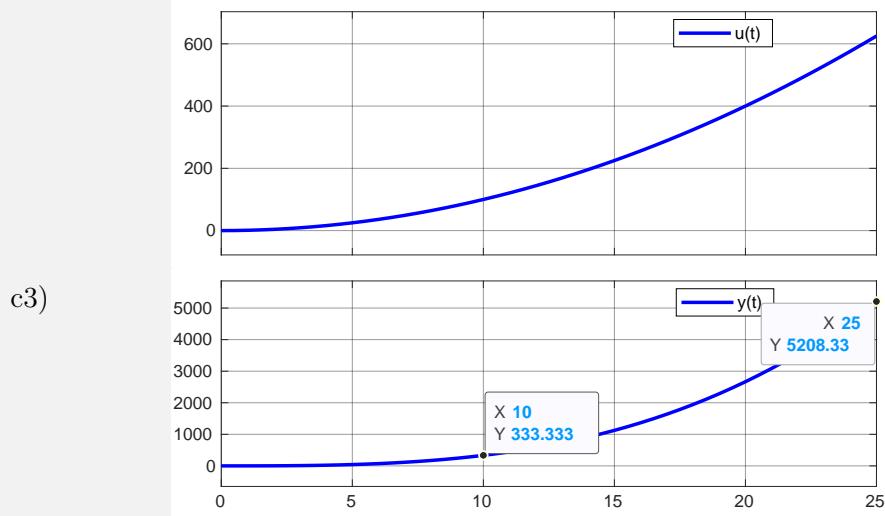
### Svar

c1)

$$u(t) = t^2$$

c2)

$$\begin{aligned}
 y(t) &= \int u(t)dt \\
 &= \int t^2 dt \\
 &= \frac{1}{3}t^3
 \end{aligned} \tag{9}$$

**Figur 5:** Resultat av blokkskjemaet i modellen.

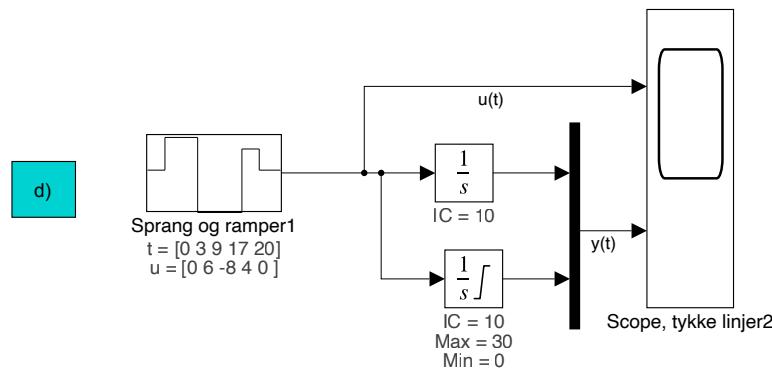
- c4) Det er visuelt tydelig at grafen vist i a3 er eksponentielt stigende. Vi kan sjekke at den stemmer med det analytiske uttrykket ved å sette inn  $x$  verdiene fra grafen inn i uttrykket.

$$y(t) = \frac{1}{3}t^3 \quad (10)$$

$$\begin{aligned} y(10) &= \frac{1}{3} \cdot 10^3 \\ &= \frac{1000}{3} \\ &= 333.33 \end{aligned} \quad (11)$$

$$\begin{aligned} y(25) &= \frac{1}{3} \cdot 25^3 \\ &= \frac{15625}{3} \\ &= 5208.33 \end{aligned} \quad (12)$$

d) Lag følgende modell.



Signalet  $u(t)$  er modellert med Sprang og ramper-blokken med følgende verdier

- Tidspunkter som  $[0 \ 3 \ 9 \ 17 \ 20]$
- Utgangsverdier som  $[0 \ 6 \ -8 \ -4 \ 0]$

som betyr at

$$u(t) = \begin{cases} 0 & \text{for } 0 \leq t < 3 \\ 6 & \text{for } 3 \leq t < 9 \\ -8 & \text{for } 9 \leq t < 17 \\ 4 & \text{for } 17 \leq t < 20 \\ 0 & \text{for } t \geq 20 \end{cases} \quad (13)$$

Velg interpolasjonsmetode som 'Trappetrinn' fra rullegardinmenyen i blokken. Den svarte avlange blokken er en 'Mux' som multiplekser flere signal inn i samme "ledning". Den summerer altså ikke signalene. I blokken Integrator med begrensning er utgangen begrenset nedad til  $y_{min}=0$  og oppad til  $y_{max}=30$ . Begge integratorene har initialverdi  $y(0)=10$ . La simuleringstiden fortsatt være 25 sekund.

**Svar på følgende spørsmål:**

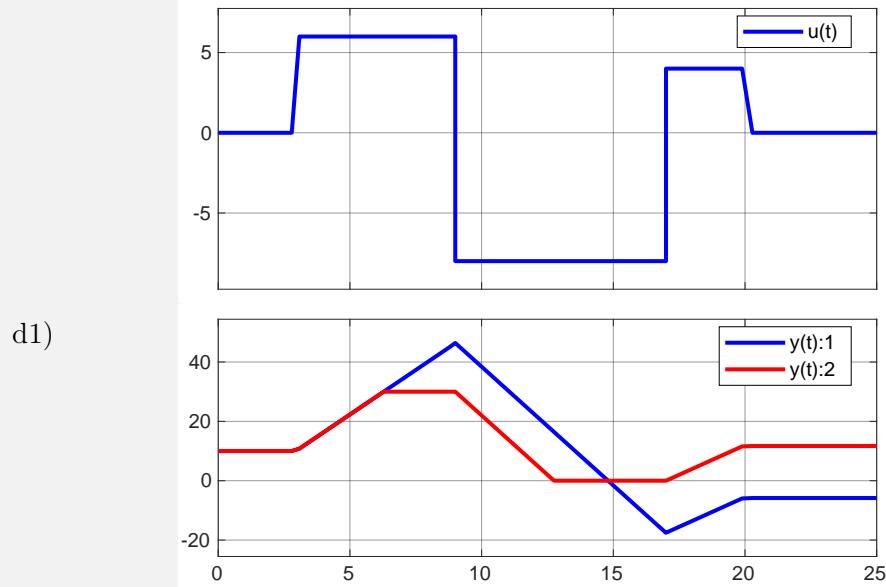
- d1) Simuler modellen og ta med simuleringsresultatet i innleveringen.
- d2) Forklar med ord hva simuleringsresultatet viser. Forklar først den blå kurven, og deretter den røde kurven.
- d3) Hvordan ville du prinsipielt implementert integratorbegrensning i kode?
- d4) Forklar hva som skjer dersom du prøver å sette initialverdien til 40 i Integrator med begrensning-blokken?
- d5) Hvorfor beholder  $y(t)$  sin verdi når  $u(t)$  går til 0 fra  $t>20$  sekund? Hva er den intuitive forklaringen?

Vær klar over at du kan skrive både matematiske uttrykk, kommentarer, funksjoner og kode i alle feltene i alle Simulink-blokkene. Under vises noen eksempler på hva du kan skrive i feltet **Utgangsverdier**:

- `[0 4 0 -5 2]*0.1`
- `[0 -3 2 0 1] %[0 4 0 -5 2]`
- `randn(1,5)`
- `1:0.1:1.4`

Uansett hvilke tall du skriver, så må alltid antall elementer i **Tidspunkter** og **Utgangsverdier** være likt.

### Svar

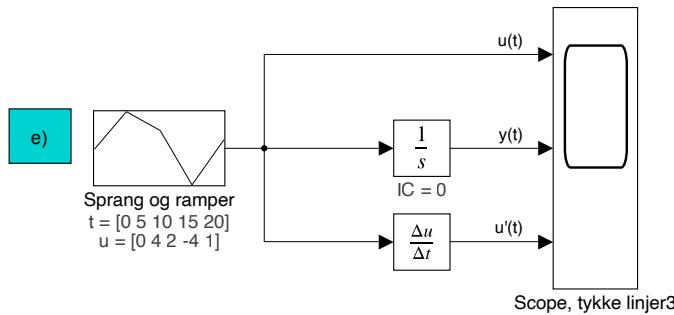


**Figur 6:** Resultat av blokkskjemaet i modellen.

- d2) Den blå kurven viser plottet av den antideriverte/integrasjonen av funksjonen  $u(t)$ . Den røde viser det samme, men det er satt en maks og min grense for funksjonens resultat.
- d3) Én måte å implementere integratorbegrensning i kode kunne være å bruke en if setning for å sjekke om resultatsverdien er over maks eller under min, og om så bytt verdien ut hendholdsvis med maks eller min verdien.

- d4) Hvis initialverdien er satt til 40 på integrator med begrensning, vil resultatet av  $y(t)$  forbli på 0 frem til  $u(t)$  går under 0 (dette skjer ved  $t = 9$  i denne simuleringen). Etter det punktet fortsetter den som i tidligere deloppgaver.
- d5) Den intuitive forklaringen er at  $u(t)$  beskriver stigningstallet til  $y(t)$ , altså hvor mye funksjonen stiger eller minker ved en gitt tidspunkt. Når  $u(t)$  går til 0 vil dette si stigningstallet til  $y(t)$  er 0, det vil si den vil verken stige eller minke.

- e) Lag følgende modell hvor den nye blokken er *Derivative*.



Signalet  $u(t)$  er modellert med Sprang og ramper-blokken med følgende verdier

- Tidspunkter som [0 5 10 15 20]
- Utgangsverdier som [0 4 2 -4 1]

og med interpolasjonsmetode som [Linear] fra rullegardinmenyen i blokken. La simuleringstiden fortsatt være 25 sekund.

Gjør følgende oppgaver / svar på følgende spørsmål:

- Simuler modellen og ta med simuleringsresultatet i innleveringen din.
- Beregn arealet under  $u(t)$  (ta hensyn til både positiv og negativt areal).
- Hvordan henger dette arealet sammen responsen i  $y(t)$ ?
- Hva viser resultatet for den deriverete  $u'(t)$ ? Forklar sammenhengen mellom  $u(t)$  og  $u'(t)$ .

**Svar (frivillig)**

- e1)
- e2)
- e3)
- e4)

## Om Sine Wave-blokken

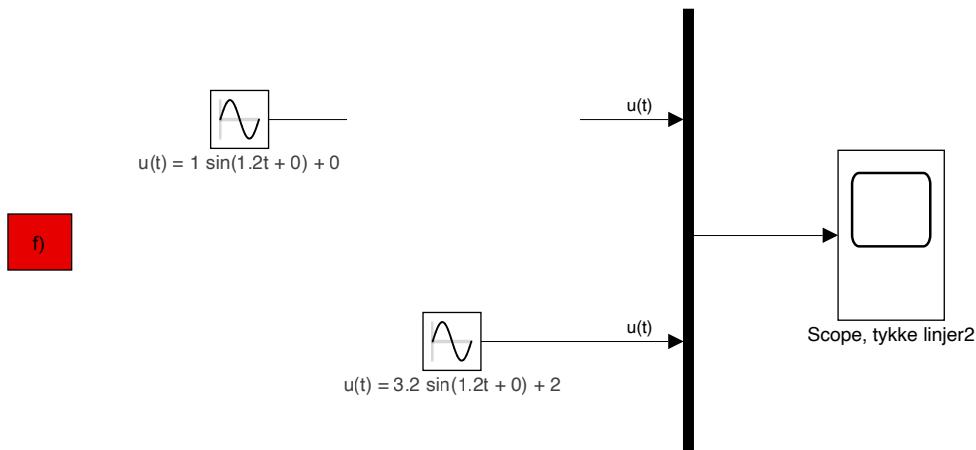
Alle modellene i denne delen av øvingen skal implementeres i subsystemet

**Om Sine Wave-blokken, oppgave 2f)-2h)** i skallfilen **oving2.slx**.

f) I denne oppgaven skal du implementere signalet  $u(t)$

$$u(t) = 3.2 \sin(1.2t + 0) + 2 \quad (14)$$

på 2 måter som vist i figuren under. Ved å sammenligne kurvene i scopet vil du kunne se at de gir samme resultatet.



- Den øverste varianten tar utgangspunkt i en Sine Wave-blokk som kun gir ut  $\sin(1.2t)$  med amplitud lik 1 og ingen likevektsverdi (bias). Det betyr at du må bygge opp signalet  $u(t)$  ved å bruke et utvalg av andre blokker, som Gain, Constant og Sum. Det er flere måter å gjøre dette på, og detaljene om dette er skjult i modellen under.
- Den nederste varianten er en Sine Wave-blokk hvor amplituden og likevektsverdien til  $u(t)$  er spesifisert direkte i blokken.

La simuleringstiden fortsatt være 25 sekund.

Gjør følgende oppgaver / svar på følgende spørsmål:

- Bygg ferdig modellen og ta et skjermdump av modellen i innleveringen din.
- Simuler modellen og ta med simuleringssresultatet i innleveringen din. For å vise at kurvene ligger over hverandre MÅ du endre den røde kurven til å være stiplet slik at det er mulig å se at de ligger over hverandre.
- Uttrykket for  $u(t)$  i ligning (??) kan generelt skrives som

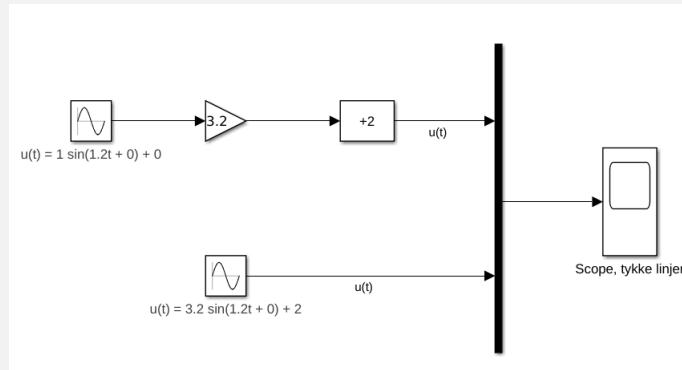
$$u(t) = U \cdot \sin(\omega \cdot t) + u_A \quad (15)$$

Anta at du ikke kjenner til uttrykket for  $u(t)$ , men bare har kjennskap til figuren med resultatet ditt. Vis hvordan du ved hjelp av kun avlesninger fra kurven i scopet kan beregne estimerer av følgende størrelser:

- amplituden  $U$
- frekvensen  $\omega$
- likevektsverdien  $u_A$

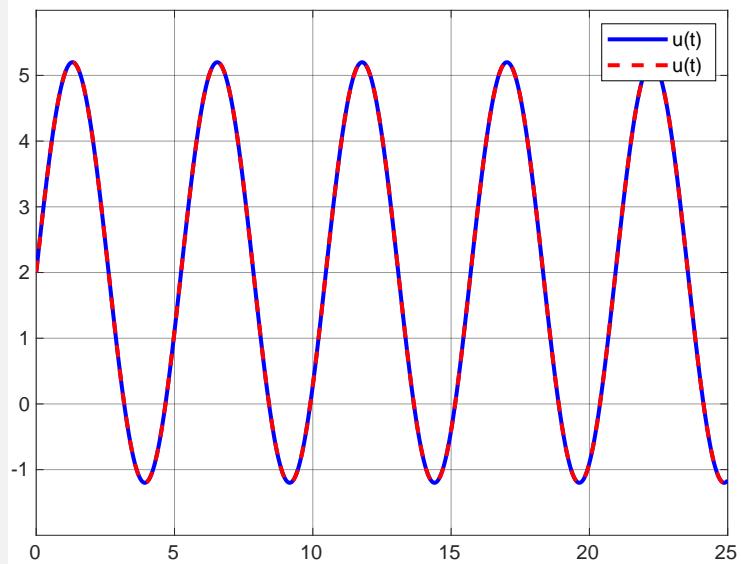
### Svar

f1)



**Figur 7:** Skjermdump av blokkskjemaet i modellen.

f2)

**Figur 8:** Resultat av blokkskjemaet i modellen.

- f3) Amplituden  $U$  kan regnes ut ved å først lese av høy og lavpunkter fra grafen og så regne ut  $\frac{\text{høy}+\text{lav}}{2}$ . Fra denne graven kan vi se høy = 5.2 og lav = 1.2. Setter vi dette inn så får vi:

$$\begin{aligned}
 U &= \frac{5.2 + 1.2}{2} \\
 &= \frac{6.4}{2} \\
 &= 3.2
 \end{aligned} \tag{16}$$

Frekvensen kan regnes ut ved å lese av periodetiden til kurven (enklest i dette tilfellet å lese av hvor kurven først går til 0 med positiv stigning), og så regner vi ut  $\frac{2\pi}{\text{periode}}$ . Perioden er circa 5.24 sekunder her.

- g) I denne oppgaven skal du implementere følgende 3 signal ved bruk av en Ramp-blokk og to Sine Wave-blokker.

$$v_1(t) = 3.2 \sin(1.2t) + 2 \quad (17)$$

$$v_2(t) = \begin{cases} 1 & \text{for } 0 \leq t < 15 \text{ sekund} \\ 1 + 0.3 \cdot (t - 15) & \text{for } t \geq 15 \text{ sekund} \end{cases} \quad (18)$$

$$v_3(t) = 1.7 \cos(0.7t) + 3 \quad (19)$$

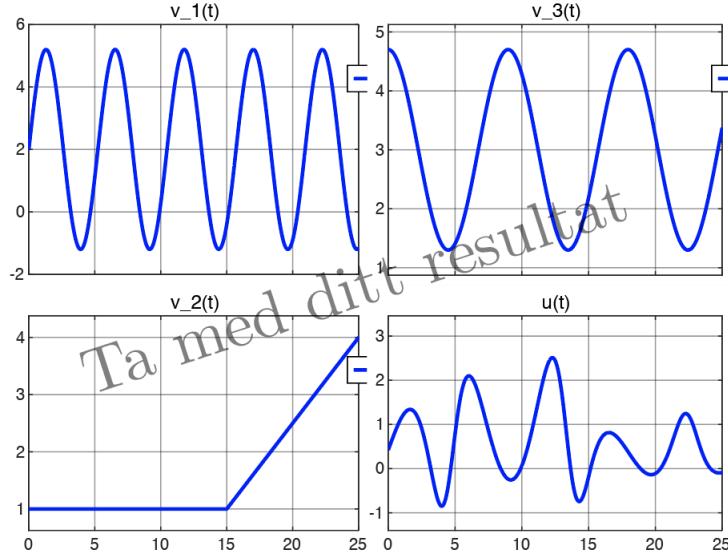
Legg merke til at  $v_3$  er et cosinus-signal. Lag deretter et signal  $u(t)$  som baserer seg på de 3 signalene på følgende måte.

$$u(t) = \frac{v_1(t)}{v_2(t) \cdot v_3(t)} \quad (20)$$

Bruk kun én Produkt-blokk til å lage  $u(t)$ . Send alle 4 signalene til et scope. **La simuleringstiden fortsatt være 25 sekund.**

**Gjør følgende oppgaver / svar på følgende spørsmål:**

- g1) Bygg modellen og ta med skjermdump av den i innleveringen din.
- g2) Simuler modellen og ta med simuleringssresultatet i innleveringen din. Vis at du får en figur lik denne.



**Figur 9:** Simuleringsresultatet

- g3) Velg et vilkårlig tidspunkt etter  $t=15$  sekund og les av en verdi fra  $u(t)$ . Sett inn samme tidspunkt i ligning (??) innsatt  $v_1(t)$ ,  $v_2(t)$  og  $v_3(t)$ , og vis at resultatet stemmer med simuleringen.

**Svar**

g1)

g2)

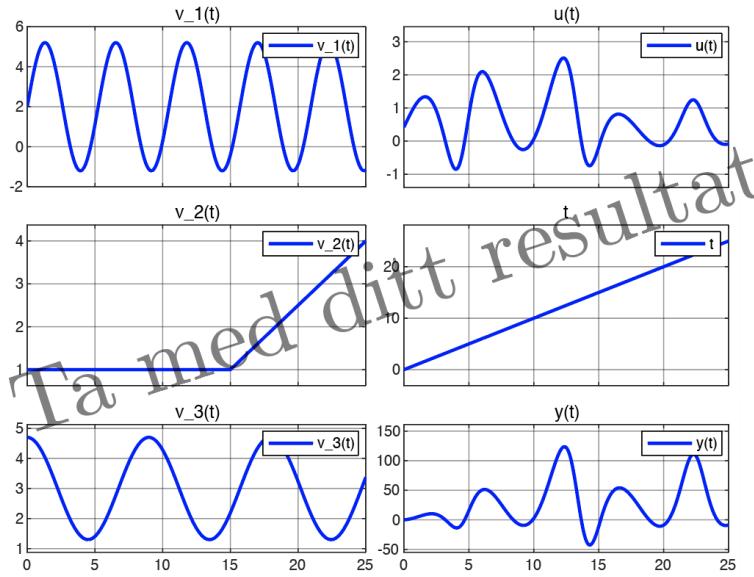
g3)

h) Kopier modellen fra g) og lag et nytt signal som

$$y(t) = 4 \cdot u(t) \cdot t \quad (21)$$

ved å bruke en Gain-blokk, en Clock-blokk og en Product-blokk.

Utvid Scopet til 6 innganger med 6 delfigurer for å vise  $v_1(t)$ ,  $v_2(t)$ ,  $v_3(t)$ ,  $u(t)$ ,  $t$  og  $y(t)$ . La simuleringstiden fortsatt være 25 sekund. Simuler modellen og ta med simuleringsresultatet i innleveringen din ved å bruke prosedyren på side ??.



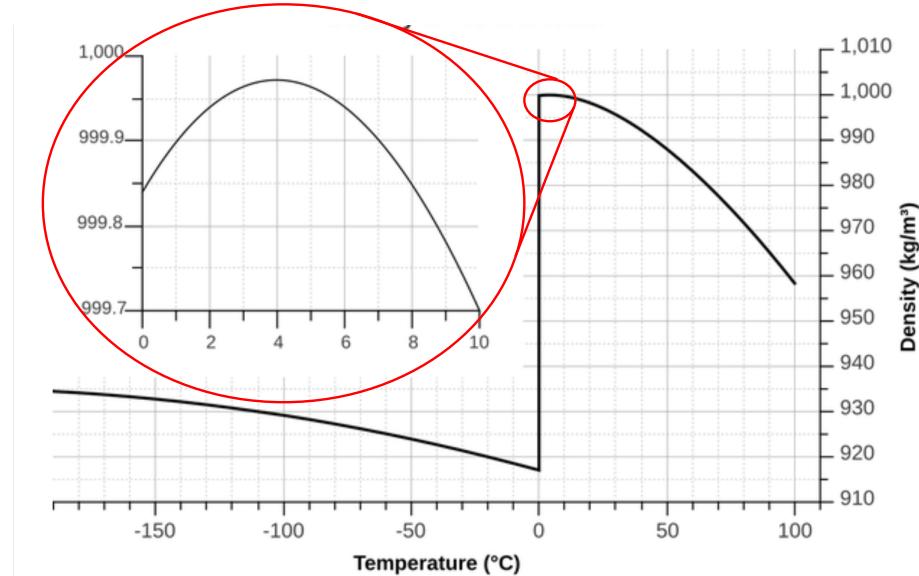
**Figur 10:** Simuleringsresultatet

Svar (frivillig)

## Om Lookup Table-blokken

Alle modellene i denne delen av øvingen skal implementeres i subsystemet Om Lookup Table-blokken, oppgave 2i)-2l) i skallfilen `oving2.slx`.

- i) I denne oppgaven skal du begynne på en ny modell. Tettheten til vann som funksjon av temperatur er vist i figuren under



**Figur 11:** Tettheten vann som funksjon av temperatur. Detaljene mellom 0 og 10 grader er vist i egen delfigur.

Målet med oppgaven er å implementere sammenhengen i figur ?? i en 1-D Lookup table-blokk som tar temperatur som et inngangssignal og gir ut tetthet som utgangssignal. Denne type blokk brukes for å implementere sammenhenger som ikke kan uttrykkes matematisk med en ligning. Blokken må spesifiseres med to vektorer med tallverdier som representerer kombinasjoner av verdier fra henholdsvis x-aksen og y-aksen. Ved å interpolere mellom punktene får du en kontinuerlig sammenhengen mellom inngangssignalet og utgangssignalet.

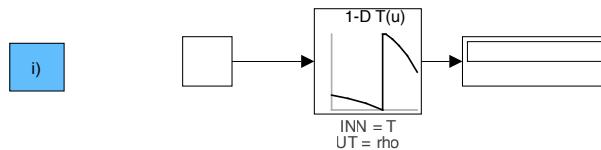
For å få med endringene i tetthet rundt 0°C samt detaljene mellom 0 og 10°C vist i delfiguren i figur ??, så tar vi utgangspunkt i følgende vektor med temperaturer fra -150°C til 100°C. Om du vil kan du legge inn flere punkter selv.

$$T = [-150, -100, -50, -0.001, 0, 2, 4, 6, 8, 10, 30, 50, 70, 100]$$

- Åpne skallfilen `oving2_data.m` og fullfør den tilhørende vektoren `rho` med tettheter avlest fra figur ?. Dette innebærer at for hver verdi i vektoren `T` ovenfor, så leser du av tilhørende tetthet fra figuren.

Ved å kjøre .m-filen får du en figur som viser sammenhengen slik at du får bekreftet at du har avlest riktige verdier og riktig antall verdier. Hensikten med å bruke en .m-fil først er at det er lettere å oppdage feil der enn i selve Lookup Table-blokken.

- I Simulinkmodellen henter du inn en 1-D Lookup table-blokk hvor du benytter disse to vektorene til å spesifisere blokken. Du kan nå velge å enten kopiere inn vektorene med tallverdier i feltene eller skrive **[rho]** og **[T]** i feltene. Dette betinger at du kjører .m-filen først slik at disse variablene er tilgjengelig i Workspace.
- Måten 1-D Lookup table-blokk interpolerer på er bestemt i fanen Algorithm under Interpolation method. Defaultverdien er som du ser Linear point slope.
- Fronten på blokken vil vise sammenhengen mellom temperatur og tetthet som vist under



Inngangen kommer fra en Constant-blokk som representerer temperaturen, og utgangen går til en Display-blokk som viser tilhørende tetthet. Du kan dobbelklikke på Display-blokkene og velge **Numeric display format** som **Long**.

#### Gjør følgende oppgaver / svar på følgende spørsmål:

- Skriv inn en av temperaturene fra vektoren  $T$  og sjekk at resultatet gir den verdien du har lest ut fra grafen. Hvilken temperatur valgte du, og hvilken tetthetsverdi fikk du ut?
- Benytt figur ?? til å estimere hva tettheten er ved  $T = -20^\circ\text{C}$ .
- Verifiser dette ved å skrive inn  $-20$  i konstantblokka. Hvilken verdi leses av? Ta med 3 desimaler. Husk at du må simulere modellen for å lese av.
- Skriv inn en svært høy temperatur, f.eks.  $T=200^\circ\text{C}$ . Hvilken tetthet får du da? Hva er det som skjer i blokken siden du får ut et tall? Er resultatet fornuftig?
- Hvilken tetthet gir  $T=2000^\circ\text{C}$ ? Kommenter gjerne resultatet.

**Svar**

i1)

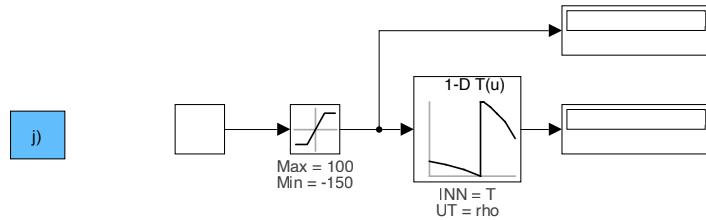
i2)

i3)

i4)

i5)

- j) Kopier modellen fra i). Siden temperaturer utenfor definisjonsområdet mellom  $-150^{\circ}\text{C}$  til  $100^{\circ}\text{C}$  gir ufysiske tettheter skal du nå begrense verdien på det som sendes inn ved å benytte en **Saturation**-blokk som vist under



Spesifiser nedre grense lik  $-150$  og øvre grense lik  $100$ , og plasser blokken som vist over.

Legg gjerne til en ny **Display**-blokk som viser hvilken verdi som sendes inn til **1-D Lookup table**-blokken.

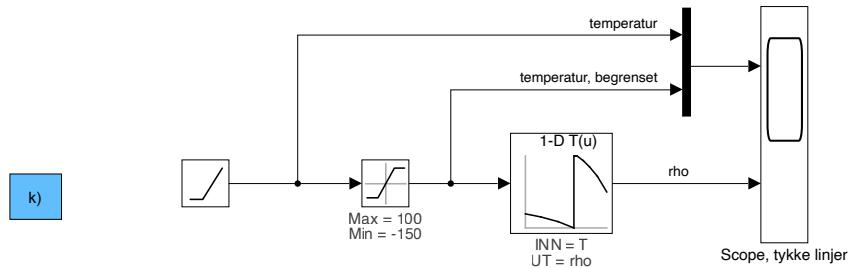
Test ut med å skrive inn f.eks.  $-200$ ,  $200$  eller  $1000$  i **Constant**-blokken.

La simuleringstiden fortsatt være **25 sekund**.

Ta med skjermdump av resultatet.

**Svar (frivillig)**

- k) Kopier modellen fra j), og erstatt Constant-blokk med en Ramp-blokk og bygg modellen om som vist under.



Spesifiser rampen på en slik måte at temperaturen stiger fra  $-200^{\circ}\text{C}$  til  $+200^{\circ}\text{C}$  i løpet av simuleringstiden på 25 sekund ut fra følgende sammenheng

$$T(t) = -200 + a \cdot t \quad (22)$$

La simuleringstiden fortsatt være 25 sekund.

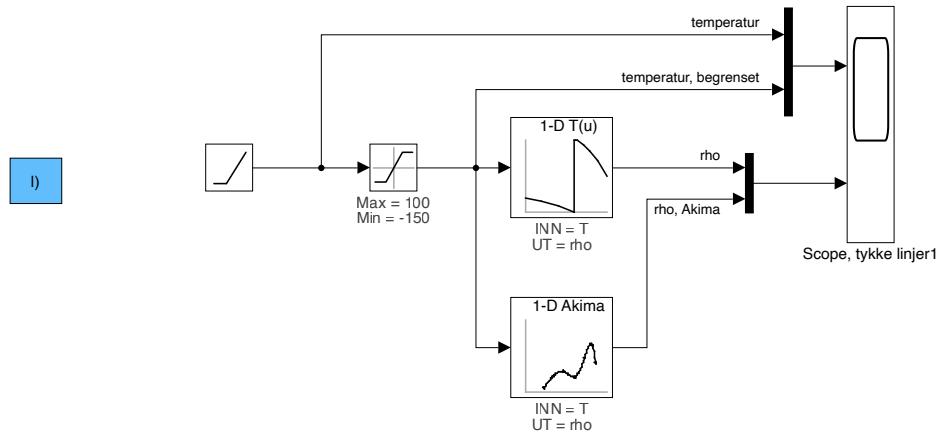
Svar på følgende spørsmål:

- k1) Hva blir verdien av  $a$ ?
- k2) Simuler modellen og ta med resultatet i innleveringen din. Forklar hva som skjer i simuleringen.
- k3) Ved hvilket tidspunktet er temperaturen  $T = -20^{\circ}\text{C}$ ? Er avlest tetthet den samme som i oppgave i)?
- k4) Forstør området mellom  $12 < t < 17$  sekund på samme måte som vist med blå kurve i figur ?. Få frem at kurven for tettheten består av lineære elementer som viser den lineære interpolasjonen som blokken utfører.

**Svar (frivillig)**

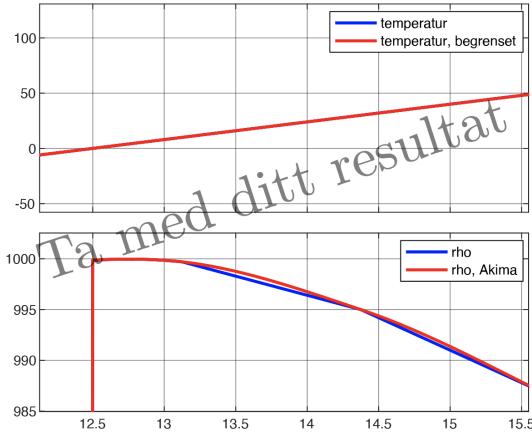
- k1)
- k2)
- k3)
- k4)

- 1) Kopier modellen fra deloppgave k). For å se forskjellen mellom to aktuelle interpolasjonsmetoder, skal du kopiere den allerede eksisterende 1-D Lookup Table-blokka og legge den i parallel med den første som vist under.



Endre deretter interpolasjonsmetoden i den kopierte blokka til Akima Spline, sjekk [https://en.wikipedia.org/wiki/Akima\\_spline](https://en.wikipedia.org/wiki/Akima_spline). La simuleringstiden fortsatt være 25 sekund.

Simuler modellen på ny, forstør et område rundt der hvor tettheten er  $\rho=1000$ , og vis at du får resultatet vist under.



**Figur 12:** Resultat som viser forskjellen på interpolasjonsmetodene.

Hva viser resultatet om forskjellen på interpolasjonsmetodene?

**Svar (frivillig)**

## Litt diverse

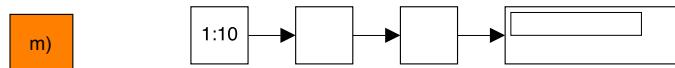
Alle modellene i denne delen av øvingen skal implementeres i subsystemet

Litt diverse, oppgave 2m)-2p) i skallfilen oving2.slx.

- m) Implementer en Simulinkmodell for ligning (??)

$$a = \sum_{n=1}^{10} n^2 \quad (23)$$

Ta utgangspunkt i følgende struktur



hvor du ser at i Constant-blokken er tallene for  $n$  som [1:10] angitt. Bruk ellers en Math Function-blokk (velg operasjon fra menyen), en Sum of elements-blokk og en Display-blokk.

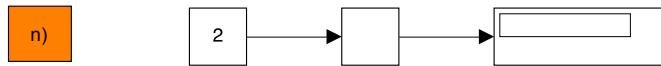
La simuleringstiden fortsatt være 25 sekund. Ta med skjermdump av modellen og resultatet.

Svar (frivillig)

- n) Implementer en modell som bruker eksponentialfunksjonen

$$e^x \quad (24)$$

Ta utgangspunkt i følgende struktur hvor konstantblokken med verdi 2 innebærer at  $x = 2$ .

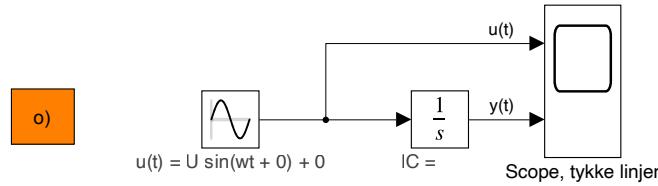


Bruk igjen en **Math Function**-blokk (velg riktig funksjon i menyen) og en **Display**-blokk.

La simuleringstiden fortsatt være **25 sekund**. Ta med skjermdump av modellen og resultatet.

**Svar (frivillig)**

- o) Kjør først filen `oving2_data.m`, og lag deretter modellen vist under



Som du ser har vi spesifiser  $u(t)$  som

$$u(t) = U \sin(\omega \cdot t) \quad (25)$$

hvor  $U=0.5$  og  $\omega=0.3$  rad/s gitt i .m-filen. Ved å integrere  $u(t)$  med initialverdi  $y(0)=0$  finner vi uttrykket for  $y(t)$  som

$$y(t) = -\frac{U}{\omega} \cdot \cos(\omega \cdot t) + \frac{U}{\omega} \quad (26)$$

La simuleringstiden nå være 100 sekund, og ikke 25 sekund som tidligere.

Gjør følgende oppgaver / svar på følgende spørsmål:

- o1) Sett initialverdien til 0 i integratoren. Simuler modellen og vis at responsen til  $y(t)$  alltid er større enn 0. Gi en forklaring på hvorfor det er slik. Ta med simuleringsresultatet i innleveringen din ved å bruke prosedyren på side ??.
- o2) Vi ønsker å få  $y(t)$ -kurven til å svinge omkring  $y=0$ , og den eneste måten å få dette til på er å redusere initialverdien  $y(0)$  i integratoren. Bestem ny initialverdi som gjør at  $y(t)$  svinger omkring  $y=0$ , og simuler på ny. Ta med simuleringsresultatet i innleveringen din.

**Svar**

o1)

o2)

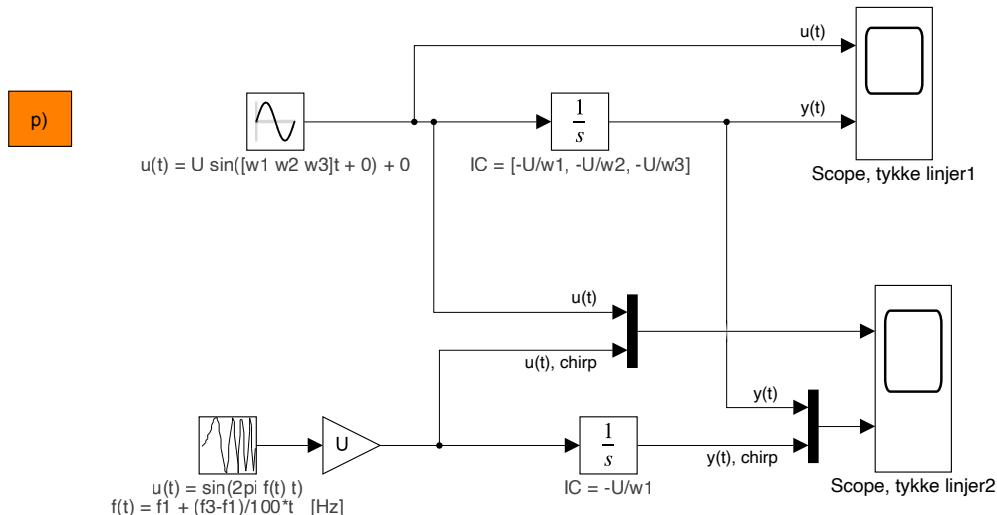
p) I denne oppgaven skal du integrerer følgende 3 signal.

$$u_1(t) = 0.5 \sin(0.3t) \quad (27)$$

$$u_2(t) = 0.5 \sin(0.55t) \quad (28)$$

$$u_3(t) = 0.5 \sin(0.8t) \quad (29)$$

Hensikten med oppgaven er å vise hvordan disse 3 sinussignalene henger sammen med Chirp-blokken. Kjør først filen `oving2_data.m`, og lag deretter modellen vist under



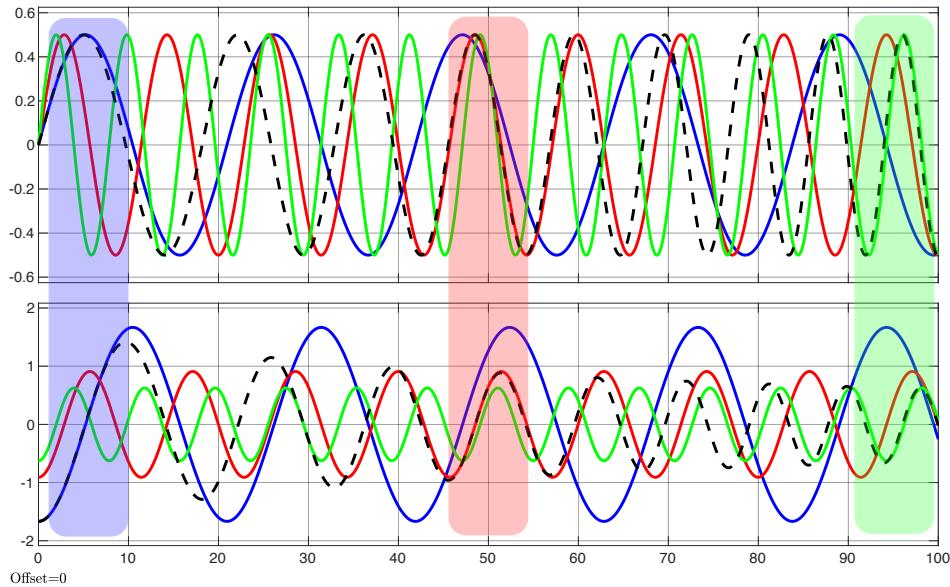
La simuleringstiden nå være 100 sekund.

- I Sine Wave-blokken spesifiserer du Frequency-feltet som  $[w1, w2, w3]$ .
- I Chirp-blokken spesifiserer du
  - $f1$  som Initial frequency,
  - $f3$  som Target frequency og
  - 100 som Target time (tilsvarer simuleringstiden).
- Spesifiser initialverdiene som vist i figuren slik at cosinuskurvene svinger omkring  $y=0$ .

Gjør følgende oppgaver / svar på følgende spørsmål:

- p1) Simuler modellen og vis at du får et resultat som ligner på responsen i figur ??, hvor vi har endret den svarte kurven til stiplet slik at det er lettere å identifisere Chirp-signalet.

Ta med simuleringsresultatet i innleveringen din ved å bruke prosedyren på side ??.



**Figur 13:** Simuleringsresultat. De fargelagte områdene er gjort i tegneprogram etterpå.

Som du ser “overlapper” det svart-stippled chirp-signalen

- med den blå sinuskurven i det blå feltet i begynnelsen,
- med den røde sinuskurven i det røde feltet i midten,
- med den grønne sinuskurven i det grønne feltet i slutten

og dette gjelder både i den øverste og nederste delfiguren.

- p2) Som du ser så reduseres amplituden  $Y$  i integralet  $y(t)$  når frekvensen  $\omega$  øker. Forklar, ut fra et matematisk ståsted, hvorfor dette skjer? I forklaringen din kan du ta utgangspunkt i at uttrykket for  $y(t)$  er gitt av følgende sammenheng.

$$\begin{aligned}
 y(t) &= \int_0^t u(\tau)d\tau + y(0) \\
 &= \int_0^t U \cdot \sin(\omega \cdot \tau)d\tau + y(0) \\
 &= \dots \\
 &= \dots
 \end{aligned}$$

Fortsett på denne utledningen og finn det analytiske uttrykket for  $y(t)$ , og bruk dette til å forklare hvorfor amplituden til  $y(t)$  synker med økende frekvens  $\omega$ .

**Svar**

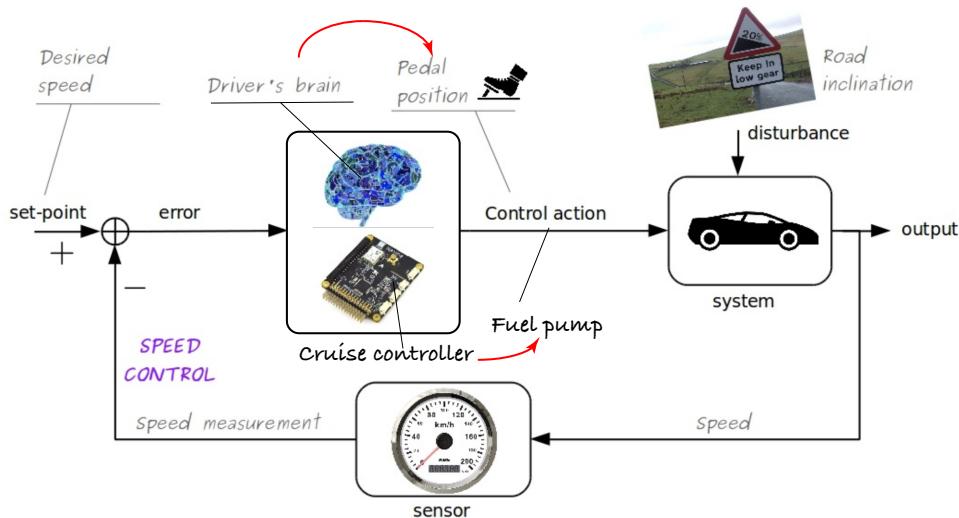
p1)

p2)

## PID-regulatoren

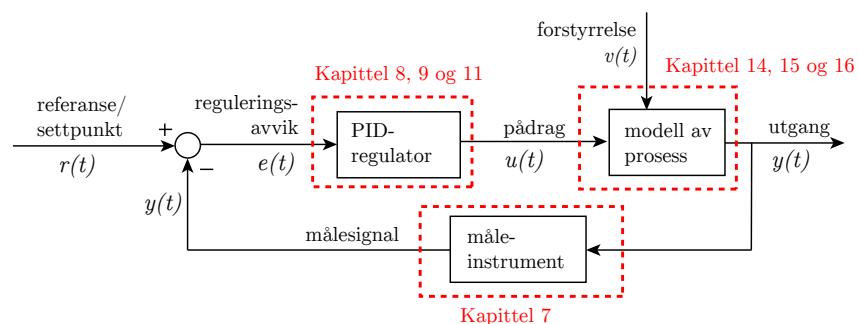
- q) I denne oppgaven skal du kun bruke en ferdig modell og ikke implementere noe som helst. Hensikten er å gi deg litt innsikt i PID-regulatoren som du skal jobbe med i det kommende Legoprosjektet.

PID-regulatoren er den mest brukte regulatoren i industrien, og for å forklare virkemåten skal vi benytte en cruisekontroller som eksempel og sammenligne med hva som skjer i hjernen vår når vi kjører selv, se figur ??.



**Figur 14:** Prinsippet med negativ tilbakekopling i bilkjøring. Enten er det du selv via hjernen din (regulator), øynene dine (måleinstrument) og føten din (pådrag) som bestemmer farten, eller så er det cruisekontrolleren. Hentet fra <https://alphaville.github.io/qub/pid-101/>.

Strukturen i figur ?? kalles negativ tilbakekopling, og reguleringsløypen kan skjematisk presenteres som i figur ??, hvor vi har henviser til relevante kapitler i kompendiet.



**Figur 15:** Prinsippet for negativ tilbakekopling.

Negativ tilbakekopling innebærer at utgangen  $y(t)$  som vi ønsker å styre/regulere, samples og måles ved hjelp av et måleinstrument, og dette signalet “føres tilbake” og sammenlignes med referansen. Siden målingene subtraheres som markert med minusstegnet, oppstår ordet “negativ”<sup>2</sup>, og vi beregner det såkalte *reguleringsavviket*  $e(t)$  som

$$e(t) = r(t) - y(t) \quad (30)$$

Reguleringsavviket er faktisk den eneste informasjonen regulatoren har tilgang til. **Regulatoren har ingen kjennskap til verken ønsket verdi  $r(t)$  eller utgangens verdi  $y(t)$ !** Det betyr at cruisekontrolleren ikke kjenner verken målt fart eller ønsket fart. Den kjenner kun til avviket fra den vilkårlige hastigheten du ønsker å kjøre i. Poenget er at når avviket  $e(t)=0$ , så “vet” regulatoren indirekte at den ukjente målte hastigheten  $y(t)$  er lik den ukjente ønskede hastigheten  $r(t)$ . En liten kommentar om notasjon: Strengt tatt er  $y(t)$ ,  $r(t)$  og  $e(t)$  egentlig diskret verdier gitt som  $\{y_k\}$ ,  $\{r_k\}$  og  $\{e_k\}$ .

Cruisekontrollen har med andre bare ett mål i livet; nemlig å sørge for at  $e(t)=0$  uansett situasjon. Dette gjelder også i situasjoner hvor bilen påvirkes av en forstyrrelse  $v(t)$ . Tenk på hva som skjer med hastigheten når du kommer til en oppoverbakke, som representerer en hindring som regulereringssystemet ikke har kontroll på. Cruisekontrollen oppnår målet i livet sitt ved å beregne et gasspådrag  $u(t)$  som påvirker bilen i en slik retning at målt hastighet  $y(t)$  nærmer seg ønsket hastighet  $r(t)$ , og dermed at  $e(t) \rightarrow 0$ . Når du ikke bruker cruisekontroll fungerer du selv som regulator, men du har jo kontroll på både fart og ønsket fart, men strengt tatt så gjør du en sammenligning og bruker indirekte avviket til å enten gi gass eller bremse.

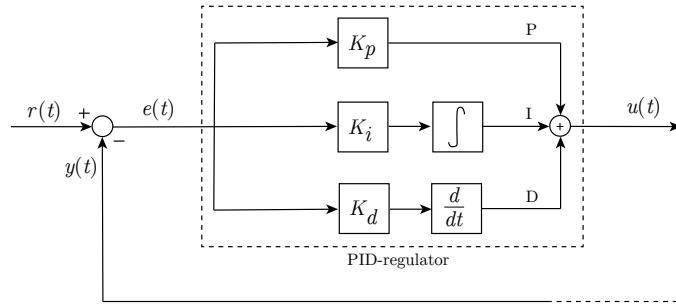
Matematisk kan PID-regulatoren uttrykkes som ligning (??), hvor “P” står for *proposjonalvirkning*, “I” står for *integralvirkning* og “D” står for *derivativirkning*.

$$u(t) = \underbrace{K_p \cdot e(t)}_{P} + \underbrace{\int_0^t K_i \cdot e(\tau) d\tau}_{I} + \underbrace{\frac{d}{dt}(K_d \cdot e(t))}_{D} \quad (31)$$

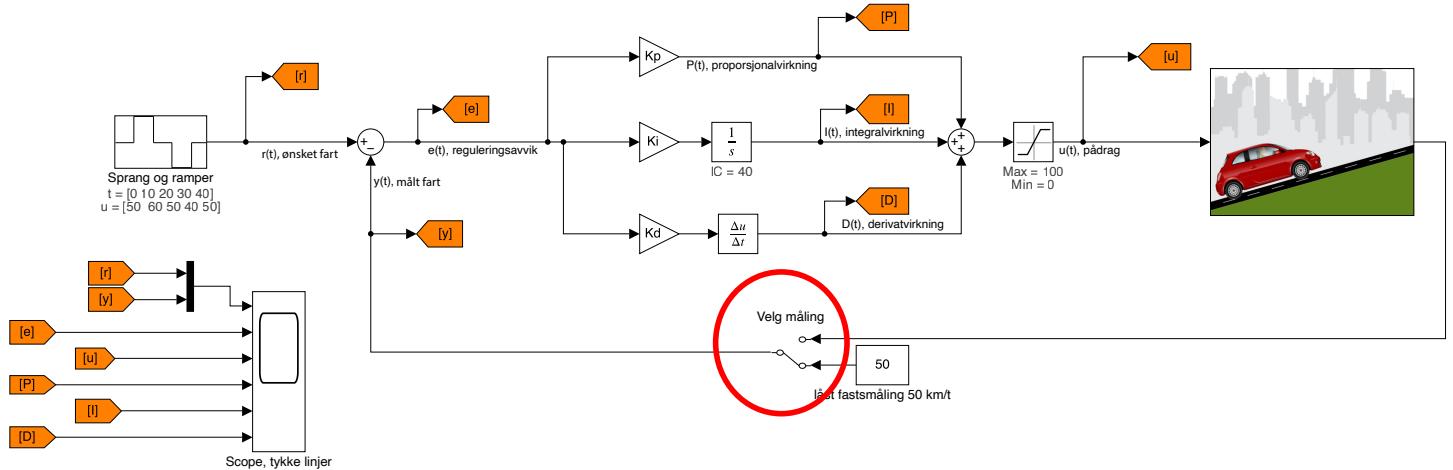
hvor  $K_p$ ,  $K_i$  og  $K_d$  er forsterkningsfaktorer som kalles regulatorparametre. Ligning (??) kan skjematiske presenteres som figur ??, hvor P-, I- og D-delene er representert som tre parallele bidrag.

---

<sup>2</sup>Hadde du addert  $y(t)$  til  $r(t)$  ville du hatt positiv tilbakekopling.

**Figur 16:** Blokkskjema over PID-regulatoren i ligning (??).

I simulinkmodellen `oving2_oppg_q.slx` vist under har vi implementert denne PID-strukturen sammen med en modell av en bil i motbakke.



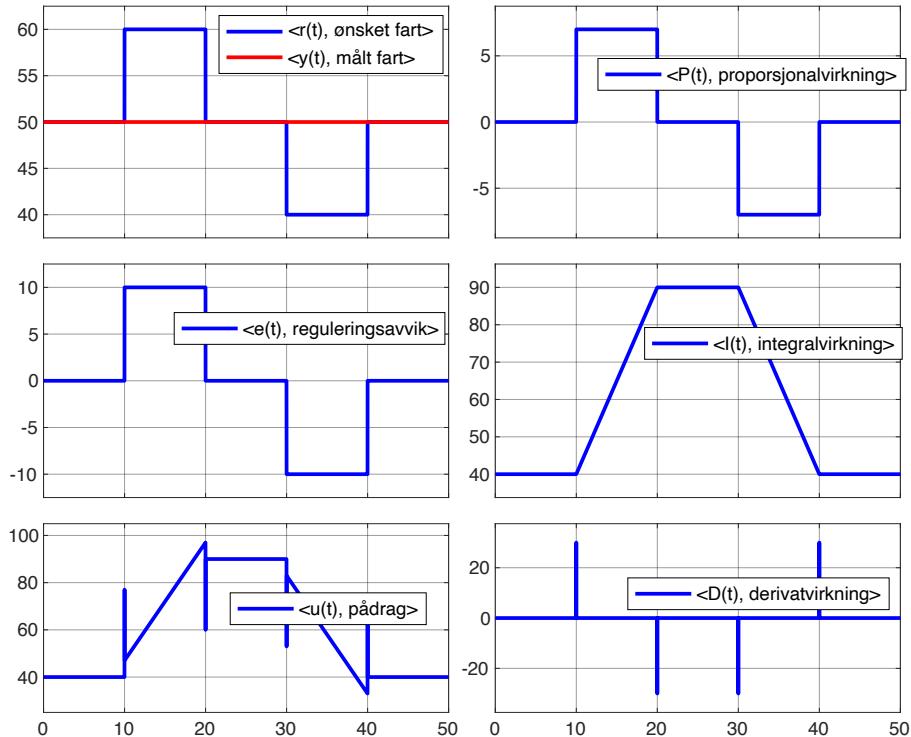
Situasjonen som er modellert er at bilen kjører i 50 km/t oppover en lang bakke, og pådraget for å oppnå dette er  $u(t)=40$  (tilsvarer initialverdien til integratoren).

Vi har benyttet oss av Goto- og From-blokker (vist i oransje) slik at modellen blir mindre spaghetti-aktig. Referansen  $r(t)$ , ønsket fart helt til venstre varierer mellom 40 og 60 km/t i løpet av simuleringsiden på 50 sekund.

Den røde ringen markerer en manuell bryter som skifter retning når du dobbelt-klikker på den. Slik bryteren står i figuren antar vi at målt hastighet er 50 km/t uansett hva cruisekontrolleren gjør, og hensikten må å låse hastigheten er at du skal få anledning til å undersøke hvordan P-, I-, og D-leddene fungerer før du kopler inn cruisekontrolleren.

Gjør følgende oppgaver:

- q1) Kjør filen `oving2_data.m`, og simuler modellen og bekrefte for din egen del at du får responsen i figur ??.



**Figur 17:** Simuleringsresultat med  $K_p=0.7$ ,  $K_i=0.5$  og  $K_d=0.03$ .

Forklar med dine egne ord hvordan P-, I-, og D-leddene fungerer.

- q2) Dobbeklikk på bryteren slik at du måler hastigheten fra bilen og at cruisekontrolleren nå er aktiv. Simuler modellen og ta med figuren du får i innleveringen. Forklar med ord hva resultatet viser. Fungerer cruisekontrollen slik du kjenner den fra din egen erfaring?
- q3) Endre verdiene av  $K_p$ ,  $K_i$  og  $K_d$  i m-filen og studere effekten på resultatet. Hvordan endres kurvene når regulatorparametriene endres?
- q4) Som er frivillig ekstraoppgave kan du prøve å lage din egen PID-blokk med din egen meny ved å følge prosedyren i vedlegg C i kompendiet.

**Svar**

- q1)
- q2)
- q3)
- q4)