Prosjektoppgave i UNIK4500

Edmund Brekke

November 2005

Innhold

1 Simuleringsmodell

Vi har det kontinuerlig-diskrete systemet

$$\begin{aligned}
\dot{\boldsymbol{x}} &= \boldsymbol{F}\boldsymbol{x} + \boldsymbol{L}u + \boldsymbol{G}v \\
z_k &= \boldsymbol{H}\boldsymbol{x}_k + w_k \\
& \boldsymbol{x}_0 \sim \mathcal{N}(\boldsymbol{0}, \hat{\boldsymbol{P}}_0),\ v \sim \mathcal{N}(0, \tilde{Q}\delta(t-\tau)),\ w_k \sim \mathcal{N}(0, \tilde{R}\delta_{kl})
\end{aligned} \tag{1}$$

med matrisene

$$\boldsymbol{F} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & -\frac{1}{T_2} & \frac{1}{T_2} \\ 0 & 0 & -\frac{1}{T_3} \end{bmatrix},\ \boldsymbol{L} = \begin{bmatrix} 0 \\ 0 \\ \frac{1}{T_3} \end{bmatrix},\ \boldsymbol{G} = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix},\ \boldsymbol{H} = \begin{bmatrix} 1 & 0 & 0 \end{bmatrix}.$$

Tilstandsvektoren $\boldsymbol{x}$ består av variablene x_1 = posisjon, x_2 = hastighet og x_3 = ankerstrøm. Vi bruker tallverdiene $T_2 = 5$s, $T_3 = 1$s, $\hat{\boldsymbol{P}}_0 = \text{diag}(1, 0.1^2, 0.1^2)$, $\tilde{Q} = 2 \cdot 0.1^2$, $\tilde{R} = 1$, $t_0 = 0$ og $t_{\text{final}} = 100$.

2 Diskretisering

For å kunne gjøre simuleringer må vi representere det kontinuerlige systemet i Likning (1) med et diskretisert system

$$\boldsymbol{x}_{k+1} = \boldsymbol{\Phi}\boldsymbol{x}_k + \boldsymbol{\Lambda}\boldsymbol{u}_k + \boldsymbol{\Gamma}\boldsymbol{v}_k,\ \boldsymbol{v}_k \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{Q}\delta_{kl})$$

der

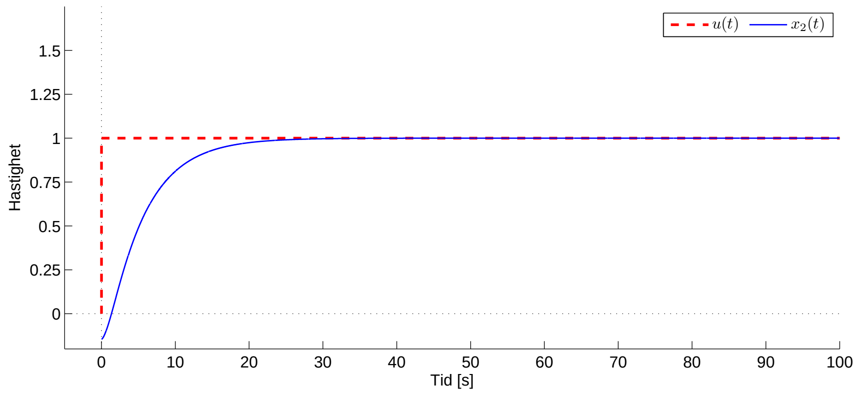
$$\begin{aligned}
\boldsymbol{\Phi} &= \int_0^{\Delta t} e^{\boldsymbol{F}s} \mathrm{d}s, \\
\boldsymbol{\Lambda} &= \int_0^{\Delta t} e^{\boldsymbol{F}s} u(s) \mathrm{d}s, \\
\boldsymbol{\Gamma}\boldsymbol{Q}\boldsymbol{\Gamma}^T &= \int_0^{\Delta t} e^{\boldsymbol{F}s} \boldsymbol{G}\boldsymbol{Q}\boldsymbol{G}^T (e^{\boldsymbol{F}s})^T \mathrm{d}s.
\end{aligned}$$

Merk at vi kan regne ut $\boldsymbol{\Lambda}$ og $\boldsymbol{\Gamma}\boldsymbol{Q}\boldsymbol{\Gamma}^T$ uten å inkludere u og v siden disse er tidsinvariante.

Diskretiseringen kan gjøres på flere måter med ulik grad av nøyaktighet. Det enkleste vil være å bruke en første ordens tilnærming,

$$\boldsymbol{\Phi} \approx \boldsymbol{I} + \boldsymbol{F}\Delta t,\ \boldsymbol{\Lambda} \approx \boldsymbol{L}\Delta t,\ \boldsymbol{\Gamma} \approx \boldsymbol{G}\Delta t,\ Q = \frac{\tilde{Q}}{\Delta t}.$$

Dersom man kan regne ut matriseeksponentialfunksjonen kan man også regne ut de ovenstående integralene numerisk med for eksempel en trapesformel.



Figur 1: Simulert system uten støy

Vi vil her regne ut integralene analytisk i henhold til en oppskrift gitt av Loan (1978). Her vises det at pådragsmatrisen $\mathbf{\Lambda}$ kan finnes via

$$\exp\left(\begin{bmatrix} \boldsymbol{F} & \boldsymbol{L} \\ \mathbf{0} & \mathbf{0} \end{bmatrix} \cdot \Delta t\right) = \begin{bmatrix} \times & \mathbf{\Lambda} \\ \mathbf{0} & \times \end{bmatrix},$$

mens støymatriseproduktet $\mathbf{\Gamma}\boldsymbol{Q}\mathbf{\Gamma}^T = \boldsymbol{A}^T\boldsymbol{B}$ der $\boldsymbol{A}$ og $\boldsymbol{B}$ finnes via

$$\exp\left(\begin{bmatrix} -\boldsymbol{F} & \boldsymbol{G}\tilde{\boldsymbol{Q}}\boldsymbol{G}^T \\ \mathbf{0} & \boldsymbol{F}^T \end{bmatrix} \cdot \Delta t\right) = \begin{bmatrix} \times & \boldsymbol{B} \\ \mathbf{0} & \boldsymbol{A} \end{bmatrix}.$$

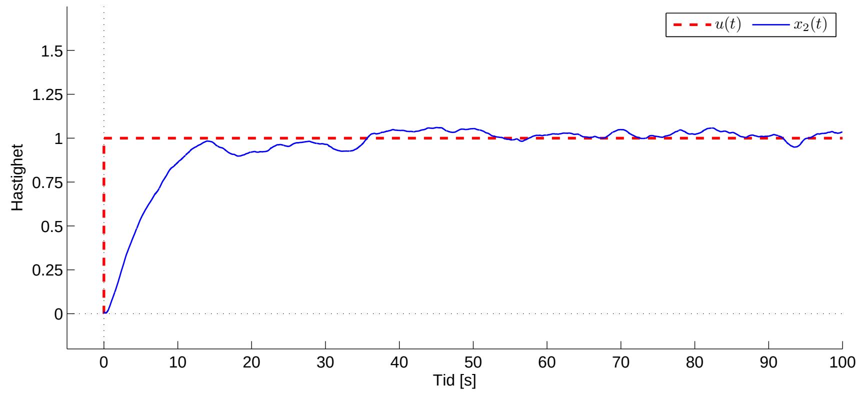
Siden det i denne oppgaven ikke er nødvendig med separate funksjoner for å beregne de ulike matrisene lar jeg alle beregningene bli gjort av en enkelt funksjon kalt diskretize.m. Denne er vedlagt i Listing 1. Diskretiseringen er gjort med en steglengde på $\Delta t = 0.01$s. De resulterende matrisene er som følger,

$$\mathbf{\Phi} = \begin{bmatrix} 1 & 0.01 & 9.9601 \cdot 10^{-6} \\ 0 & 0.998 & 0.002 \\ 0 & 0 & 0.99 \end{bmatrix}, \mathbf{\Lambda} = \begin{bmatrix} 0 \\ 0 \\ 0.01 \end{bmatrix},$$

$$\mathbf{\Gamma} = \begin{bmatrix} 6.3035 \cdot 10^{-8} & 0 & 0 \\ 1.5738 \cdot 10^{-5} & 4.0743 \cdot 10^{-6} & 0 \\ 0.0105 & 0.0081 & 0.0047 \end{bmatrix}.$$

3 Simulering

Vi simulerer her deterministisk ($\tilde{Q} = 0$) og stokastisk system med en heavysidefunksjon som pådrag. Figur 1 viser deterministisk system mens Figur 2 viser stokastisk system. Simuleringene er gjort ved hjelp av den samme kildekoden som er brukt i Oppgave 4.



Figur 2: Simulert system med støy

4 Optimalt Kalmanfilter

Gitt målinger basert på simuleringene i Oppgave 3 ønsker vi å finne et estimat $\hat{\boldsymbol{x}}_k$ og påfølgende prediksjoner $\bar{\boldsymbol{x}}_k$ av tilstandsvektoren for tidspunkter $k > 0$. Dette gjøres med et Kalmanfilter beskrevet av følgende likninger (Gelb 1974)

$$\begin{aligned}
\bar{\boldsymbol{x}}_{k+1} &= \boldsymbol{\Phi}\bar{\boldsymbol{x}}_k + \boldsymbol{\Lambda}\boldsymbol{u}_k \\
\bar{\boldsymbol{P}}_{k+1} &= \boldsymbol{\Phi}\bar{\boldsymbol{P}}_k\boldsymbol{\Phi}^T + \boldsymbol{\Gamma}\boldsymbol{Q}\boldsymbol{\Gamma}^T \\
\hat{\boldsymbol{x}}_k &= \bar{\boldsymbol{x}}_k + \boldsymbol{K}(z_k - \boldsymbol{H}\bar{\boldsymbol{x}}_k) \\
\hat{\boldsymbol{P}}_k &= (\boldsymbol{I} - \boldsymbol{K}_k\boldsymbol{H})\bar{\boldsymbol{P}}_k \\
\boldsymbol{K}_k &= \bar{\boldsymbol{P}}_k\boldsymbol{H}^T(\boldsymbol{H}\bar{\boldsymbol{P}}\boldsymbol{H}^T + \boldsymbol{R})^{-1}.
\end{aligned} \tag{2}$$

De to første formlene i Likning (2) utgjør tidsoppdateringen og gjøres med en frekvens på 100Hz. Når det foreligger målinger brukes de tre siste formlene, som utgjør måleoppdateringen. Dette skjer hvert sekund. Måleoppdateringen fungerer som en initialbetingelse for tidsoppdateringen.

Figur 3 viser både den sanne hastigheten $x_2(t)$ som en blå kurve og den estimerte hastigheten $\bar{x_2}(t)$ som en grønn kurve. Måleoppdateringene $\hat{x}_2$ er markert med svarte stjerner. I tillegg er pådraget $u(t)$ plottet som en stiplet rød kurve. Ved nærmere inspeksjon av Figur 3 kan vi se hvordan dynamikken i filterestimatet alltid henger etter dynamikken i den sanne hastigheten. Utslagene fra likevektstilstanden $\lim_{t\to\infty} x_2 = 1$ er allikevel små grunnet den relativt høye måleusikkerheten.

I Figur 4 ser vi residualene $\bar{x}_2(t) - x(t)$ plottet som en svart kurve mens standardavviket $\bar{s}_2(t) = \sqrt{[P]_{22}}$ fra tidsoppdateringen er plottet som en blå kurve. Tilsvarende viser Figur 5 residualer og standardavvik for måleoppdatering.

Vi merker oss at residualene i omlag 2/3 av tilfellene ligger innenfor båndet utspent av $\pm s_2$. Dette demonstrerer at Kalmanfilteret gir et optimalt estimat på usikkerheten.

I tillegg til funksjonen discretize.m ble funksjonen kalman.m samt scriptet oppgave4.m brukt for å generere disse resultatene. Kildekoden er vedlagt i Listing 2 og 4.

5 Monte Carlo-simulering av optimalt system

For å finne forventning $E\{\hat{\boldsymbol{e}}_k\}$ og kovarians $\text{Cov}\{\hat{\boldsymbol{e}}_k\}$ til estimeringsfeilen må vi generelt bruke Monte Carlo-simulering. Dette gjøres ved å simulere og filtrere gjentatte ganger for. For hver kjøring trekkes $\boldsymbol{x}_0$, $\boldsymbol{v}_k$ og $\boldsymbol{w}_k$ på nytt fra sine respektive fordelinger.

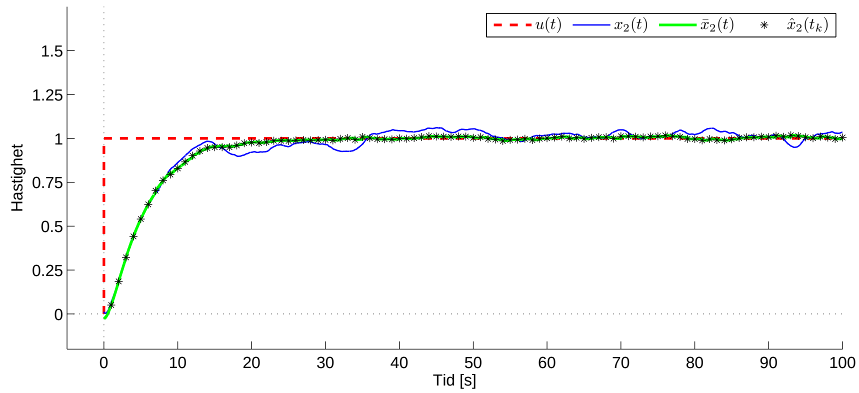
Vi definerer målefeilen ved tidspunkt k for kjøring nummer j som $\boldsymbol{e}_k^j = \boldsymbol{x}_k^j - \hat{\boldsymbol{x}}_k^j$ og regner ut estimatene $\boldsymbol{m}_k^N \approx E\{\hat{\boldsymbol{e}}_k\}$ og $\boldsymbol{P}_k^N \approx \text{Cov}\{\hat{\boldsymbol{e}}_k\}$ rekursivt ved hjelp av formlene

$$\begin{aligned}
\Delta_k^N &= \hat{\boldsymbol{e}}_k^N - \hat{\boldsymbol{e}}_k^{N-1} \\
\hat{\boldsymbol{m}}_k^N &= \hat{\boldsymbol{m}}_k^{N-1} + \frac{1}{N}\Delta_k^N,\ \hat{\boldsymbol{m}}_k^0 = \mathbf{0}, \\
\hat{\boldsymbol{P}}_k^N &= \frac{N-2}{N-1}\hat{\boldsymbol{P}}_k^{N-1} + \frac{1}{N}\Delta_k^N(\Delta_k^N)^T,\ \hat{\boldsymbol{P}}_k^0 = \mathbf{0},\ \hat{\boldsymbol{P}}_k^1 = \mathbf{0}.
\end{aligned}$$

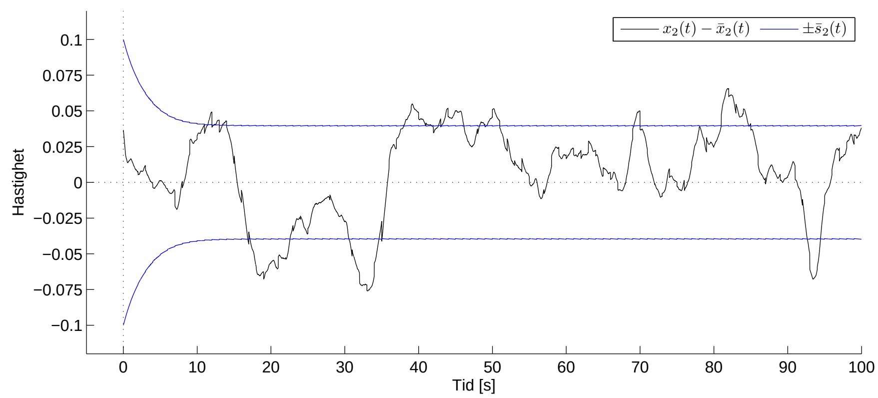
Figur 6 viser Kalmanfilterets estimat $\hat{x}_2(t)$ for 10 forskjellige måleserier der $v(t)$ og $w(t)$ er trukket på nytt fra sine respektive fordelinger. Figur 7 viser de tilhørende estimeringsfeilene.

Figur 8 og Figur 9 viser begge Monte-Carlo-beregnet feil og standardavvik for x_2 basert på henholdsvis 10 og 100 måleserier,samt det teoretisk beregnede standardavviket fra Kalmanfilteret. Legg merke til hvordan $\sqrt{\text{Cov}\{\hat{e}_2(t)\}}$ svinger rundt $\bar{s}_{22}(t)$. Utslaget blir mindre dess større N er.

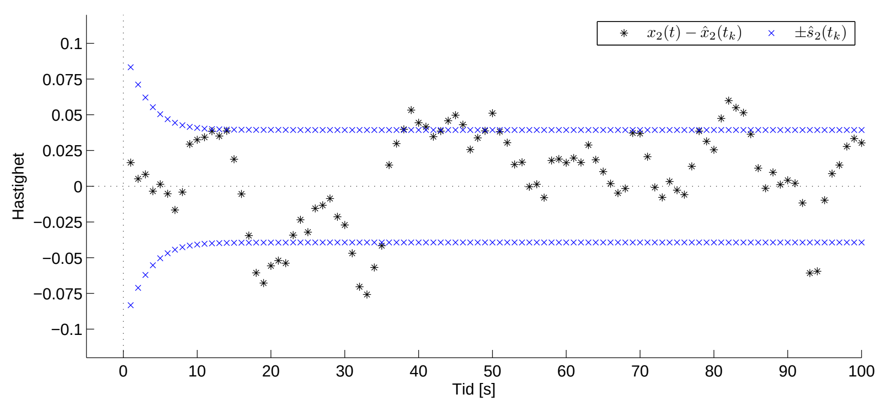
Kildekoden som ble brukt for å generere disse figurene finnes i Listing 5.



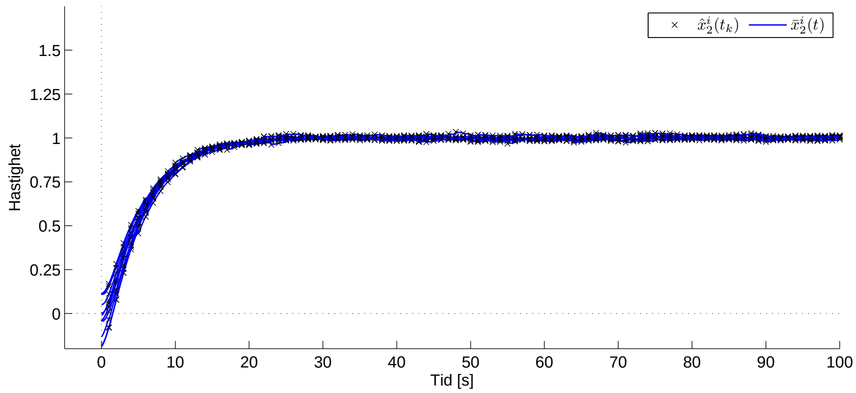
Figur 3: Sann og estimert hastighet for optimalt Kalmanfilter



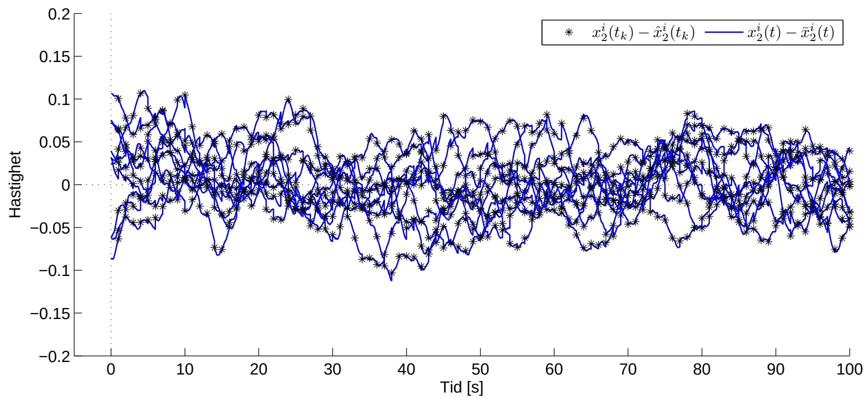
Figur 4: Residualer fra tidsoppdatering for optimalt Kalmanfilter



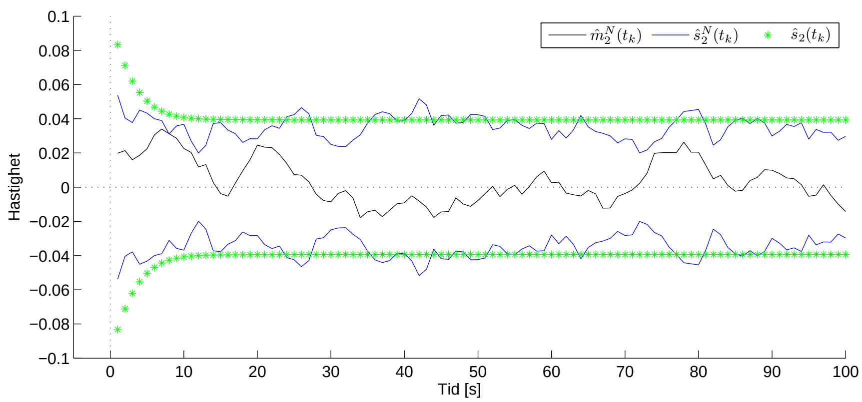
Figur 5: Residualer fra måleoppdatering for optimalt Kalmanfilter



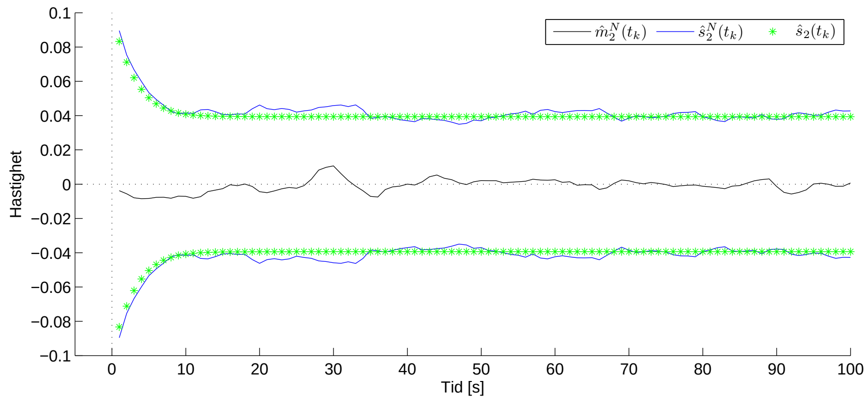
Figur 6: Hastighetsestimater for 10 forskjellige måleserier



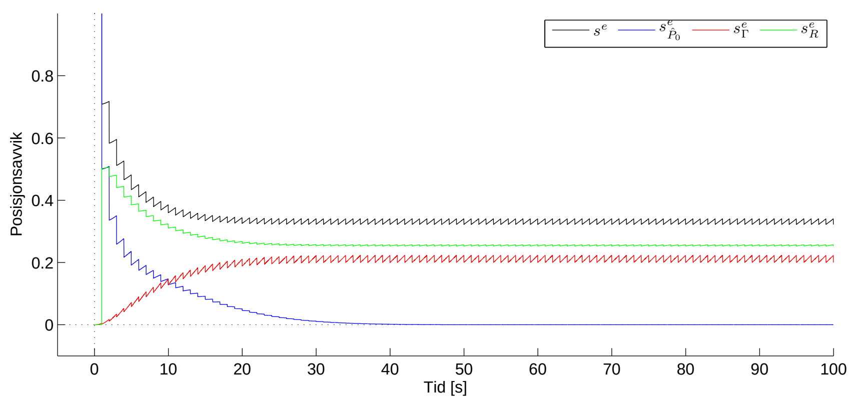
Figur 7: Kalmanfilterets hastighetsfeil for 10 forskjellige måleserier



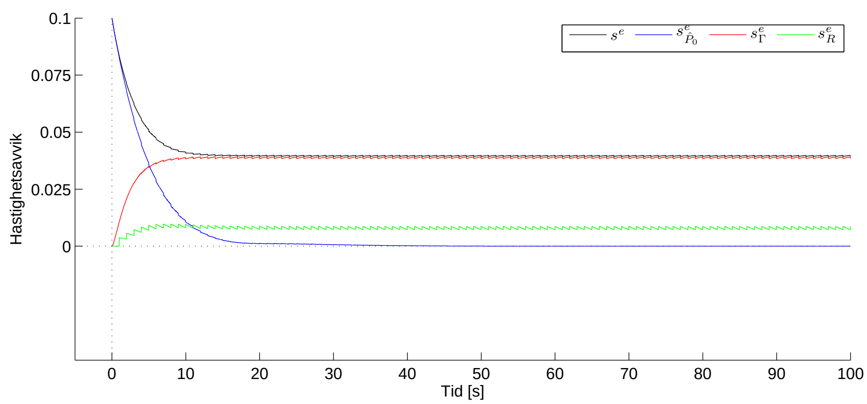
Figur 8: Teoretisk og Monte-Carlo-beregnet standardavvik for N = 10



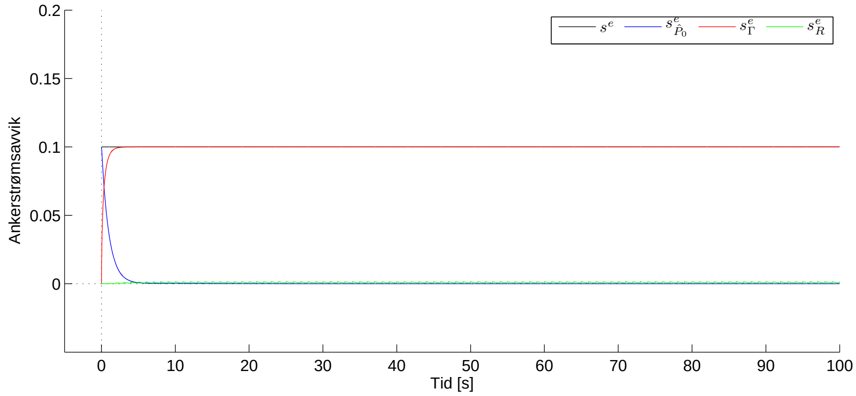
Figur 9: Teoretisk og Monte-Carlo-beregnet standardavvik for N = 100



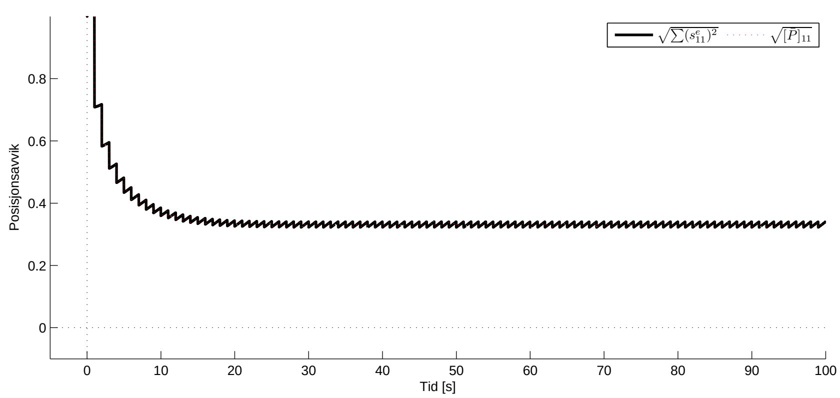
Figur 10: Feilbudsjett for posisjon



Figur 11: Feilbudsjett for hastighet



Figur 12: Feilbudsjett for ankerstrøm



Figur 13: RMS-sum av standaravvik for posisjonsfeil

6 Feilbudsjett for optimalt Kalmanfilter

I feilbudsjettet studerer vi hvordan de tre feilkildene initiell kovarians, prosesstøy og målestøy bidrar til totalfeilen. Siden simuleringsmodellen og filtermodellen er like blir feillikningene i diskret tid

$$
\begin{aligned}
\bar{\boldsymbol{P}}^e_{k+1} &= \boldsymbol{\Phi}\boldsymbol{P}^e_k\boldsymbol{\Phi} + \boldsymbol{\Gamma}\boldsymbol{Q}\boldsymbol{\Gamma} && (3)\\
\hat{\boldsymbol{P}}^e_k &= (\boldsymbol{I} - \boldsymbol{K}_k\boldsymbol{H})\bar{\boldsymbol{P}}^e_k(\boldsymbol{I} - \boldsymbol{K}_k\boldsymbol{H})^T + \boldsymbol{K}_k\boldsymbol{R}\boldsymbol{K}_k^T && (4)\\
\hat{\boldsymbol{P}}^e_0 &= \bar{\boldsymbol{P}}^*_{\boldsymbol{0}}
\end{aligned}
$$

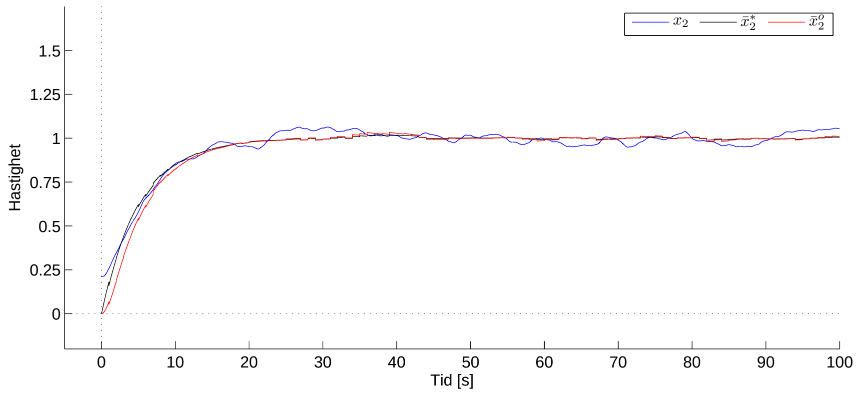
hvor måleoppdateringen i Likning (4) fungerer som initialbetingelse for tidsoppdateringen i Likning (3), jfr. Gelb (1974) side 250. $\boldsymbol{K}_k$ er forsterkningene beregnet av Kalmanfilteret, dvs. tilhørende filtermodellen. Vi lager feilbudsjettet ved å kjøre ovenstående likninger 3 ganger med henholdsvis $\bar{\boldsymbol{P}}_{\boldsymbol{0}}$, $\boldsymbol{\Gamma}\boldsymbol{Q}\boldsymbol{\Gamma}$ og $\boldsymbol{R}$ ulik null.

Figur 10, Figur 11 og Figur 12 viser feilbudsjett for henholdsvis posisjon, hastighet og ankerstrøm. De blå kurvene representerer feilbidrag fra initiell kovarians, de røde representerer feilbidrag fra prosesstøy og de grønne representerer feilbidrag fra målestøy. Totalfeilen er representert ved de svarte kurvene.

Vi ser at for alle tilstandsvariablene synker feilbidraget fra initiell kovarians hurtig mot null. Dette er som ventet da målinger etterhvert gir informasjon som gjør kjennskap til starttilstanden overflødig. Prosesstøyen har stor innvirkning på ankerstrømsfeilen og liten virkning på posisjonsfeilen, mens det forholder seg omvendt for målestøyen. Dette skyldes at vi måler posisjonen direkte, mens vi får kjennskap til hastighet og ankerstrøm kun gjennom prosessmodellen.

I Figur 13 er RMS-summen av standardavvikene for posisjonsfeilen vist som en svart kruve mens det tilsvarende standardavviket $\bar{s}_1 1(t)$ beregnet av Kalmanfilteret er vist som en stiplet rød linje. Siden systemmodellen er like filtermodellen blir $\bar{\boldsymbol{P}}^e$ lik $\bar{\boldsymbol{P}}^*$ og kurvene sammenfaller.

For å beregne feilbudsjettene har vi brukt funksjonen covupdate.m som utelukkende gjør tids- og måleoppdateringer av kovariansmatrisen gitt Kalmanfilterforsterkningene $\boldsymbol{K}_k$. Denne er vedlagt i Listing 3. Resten av kildekoden brukt for å lage feilbudsjett finnes i Listing 6.



Figur 14: Predikert hastighet for optimalt og suboptimalt Kalmanfilter

7 Suboptimalt Kalmanfilter

7.1 Suboptimalt Kalmanfilter basert på modellreduksjon

Simuleringsmodellen er som før, men filtermodellen forenkles til

$$\begin{aligned}\dot{\boldsymbol{x}}^* &= \boldsymbol{F}^*\boldsymbol{x}^* + \boldsymbol{L}^* u + \boldsymbol{G}^* v^* \\ z_k &= \boldsymbol{H}^*\boldsymbol{x}_k^* + w_k^*\end{aligned}$$

med matrisene

$$\boldsymbol{F}^* = \begin{bmatrix} 0 & 1 \\ 0 & -\frac{1}{T_2}\end{bmatrix}, \boldsymbol{L}^* = \begin{bmatrix} 0 \\ \frac{1}{T_2}\end{bmatrix}, \boldsymbol{G}^* = \begin{bmatrix} 0 \\ \frac{1}{T_2}\end{bmatrix}, \boldsymbol{H}^* = \begin{bmatrix} 1 & 0 \end{bmatrix}.$$

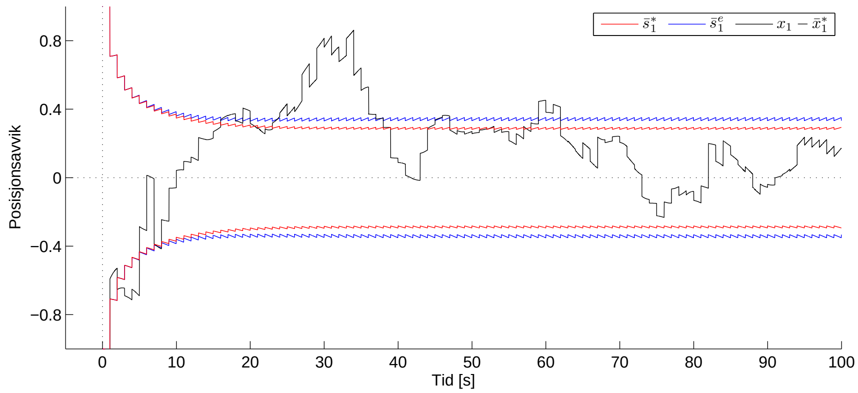
og prosesstøyen modellert som $v^* \sim \mathcal{N}(0, T_3\tilde{Q}/2)$. Resultatene fra et filter basert på denne modellen er vist sammen med resultatene fra et optimalt Kalmanfilter i Figur 14. Vi ser at forskjellen er negliserbar. Man kan imidlertid spørre seg hvorvidt noen av filterne gjør noe estimering av betydning. Målestøyen er så høy filterne må oppføre seg forsiktig og først og fremst forholde seg til modellen.

For å beregne standardavvik til estimeringsfeilene bruker vi en augmentert tilstandsvektor definert ved

$$\boldsymbol{x}_a = \begin{bmatrix} \boldsymbol{e} \\ \boldsymbol{x}\end{bmatrix} = \begin{bmatrix} \boldsymbol{N}\boldsymbol{x} - \boldsymbol{x}^* \\ \boldsymbol{x}\end{bmatrix} \quad \text{der } \boldsymbol{N} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0\end{bmatrix} \text{ i dette tilfellet.}$$

Videre lar vi

$$\boldsymbol{F}_a = \begin{bmatrix} \boldsymbol{F}^* & \boldsymbol{N}\boldsymbol{F} - \boldsymbol{F}^*\boldsymbol{N} \\ 0 & \boldsymbol{F}\end{bmatrix}, \boldsymbol{G}_a = \begin{bmatrix} \boldsymbol{N}\boldsymbol{G} \\ \boldsymbol{G}\end{bmatrix} = \begin{bmatrix} \boldsymbol{0} \\ \boldsymbol{G}\end{bmatrix},$$



Figur 15: Sann og estimert usikkerhet for posisjon

$$\boldsymbol{H}_a = \begin{bmatrix} \boldsymbol{H}^* & \boldsymbol{H} - \boldsymbol{H}^*\boldsymbol{N} \end{bmatrix}, \, \boldsymbol{K}_a(t_k) = \begin{bmatrix} \boldsymbol{K}^*(t_k) \\ \boldsymbol{0} \end{bmatrix}.$$

Da vil estimeringsfeilens kovarians kunne beregnes via

$$\dot{\bar{\boldsymbol{P}}}_a = \boldsymbol{F}\bar{\boldsymbol{P}}_a + \bar{\boldsymbol{P}}_a\boldsymbol{F}^T + \boldsymbol{G}_a\tilde{Q}\boldsymbol{G}_a^T, \; \bar{\boldsymbol{P}}_a(t_k) = \hat{\boldsymbol{P}}_a(t_k) \tag{5}$$

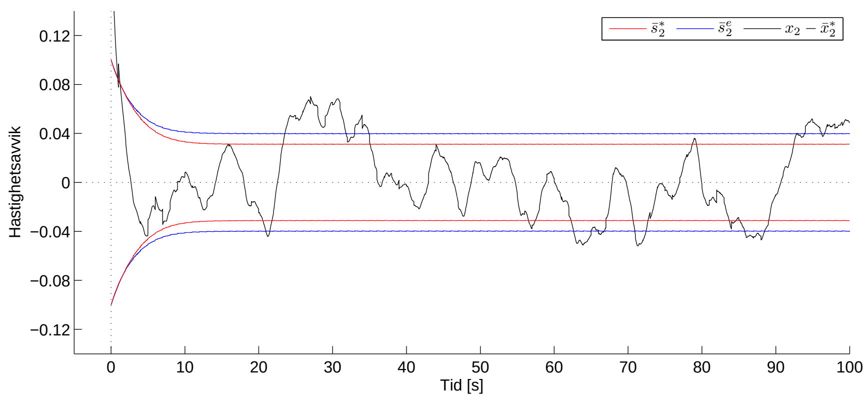
$$\begin{aligned} \hat{\boldsymbol{P}}_a(t_k) = {} & (\boldsymbol{I} - \boldsymbol{K}_a(t_k)\boldsymbol{H}_a)\bar{\boldsymbol{P}}_a(t_k)(\boldsymbol{I} - \boldsymbol{K}_a(t_k)\boldsymbol{H}_a)^T \\ & + \boldsymbol{K}_a(t_k)R\boldsymbol{K}_a(t_k)^T. \end{aligned} \tag{6}$$

Likning (5) må selvsagt diskretiseres for praktiske formål, og vi får da et utrykk på samme form som Likning (3). Vi finner Cov$\{\boldsymbol{e}(t)\}$ ved å plukke ut delmatrisen øverst til venstre i $\boldsymbol{P}_a$. Denne blir av dimensjon 2×2 i vårt tilfelle. Rekursjonen starter med initialverdien

$$\bar{\boldsymbol{P}}_a(t_0) = \begin{bmatrix} \boldsymbol{N}\hat{\boldsymbol{P}}_0\boldsymbol{N} & \boldsymbol{N}\hat{\boldsymbol{P}}_0 \\ \hat{\boldsymbol{P}}_0\boldsymbol{N}^T & \hat{\boldsymbol{P}}_0 \end{bmatrix}.$$

Resultater av kovariansanalysen er vist i Figur 15 og Figur 16. De røde kurvene viser s_i^* (den usikkerheten det suboptimale Kalmanfilteret selv beregner) mens de blå kurvene viser s_i^e (den sanne usikkerheten som beregnes via Likning (5) og Likning (6)). Vi ser at det suboptimale Kalmanfilteret vurderer sin egen usikkerhet som mindre enn den faktisk er. Forskjellen er som ventet av Figur 14 ikke særlig stor.

For å beregne $\boldsymbol{P}_a$ har vi også her brukt funksjonen covupdate.m fra Listing 3. Resten av kildekoden finnes i Listing 6.



Figur 16: Sann og estimert usikkerhet for hastighet

7.2 Suboptimalt Kalmanfilter basert på måling som pådrag

Modellen i Likning (1) skal simulere en vogn i rettlinjet bevegelse drevet av den elektriske strømmen x_3. Vi undersøker her hvordan posisjon og hastighet kan estimeres dersom vi kan måle strømmen direkte. Det er rimelig å anta at vi kan måle strømmen med frekvens på 100Hz og med usikkerhet på rundt 1%. Under disse forutsetningene blir x_3 å regne som tilnærmet kjent. Siden x_3 ikke avhenger av x_1 eller x_2 kan strømmen modelleres som et pådrag på resten av systemet.

Vi kan altså angripe den nye situasjonen på blant annet to måter: Enten så måler vi både x_1 og x_3 og bruker målingene i et vanlig Kalmanfilter, eller så kan vi bruke målingen av x_3 som et pådrag i et Kalmanfilter som formelt sett kun måler x_1. Det siste leder til den suboptimale filtermodellen beskrevet i Likning (7) og Likning (7). Simuleringsmodellen blir

$$\begin{aligned}\dot{\boldsymbol{x}} &= \boldsymbol{F}\boldsymbol{x} + \boldsymbol{L}u + \boldsymbol{G}v \\ \boldsymbol{z}_k &= \boldsymbol{H}\boldsymbol{x}_k + \boldsymbol{w}_k\end{aligned}$$

hvor

$$H = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} \text{ og } \boldsymbol{w}_k \sim \mathcal{N}\left(\boldsymbol{0}, \boldsymbol{R}^*\delta(t-\tau)\right) = \mathcal{N}\left(\boldsymbol{0}, \begin{bmatrix} 1 & 0 \\ 0 & 0.01^2 \end{bmatrix}\delta(t-\tau)\right).$$

Merk at her måles strømmen med en frekvens på 1Hz.

For å lage filtermodellen utnytter vi imidlertid antakelsen om at strømmen kan måles hyppigere (100Hz) til å beregne et tilnærmet kontinuerlig pådrag

$u^*(t) = x_3(t) + v^*$ der $v^* \sim \mathcal{N}(0, \tilde{Q}^*\delta(t-\tau))$ og $\tilde{Q}^* = [\boldsymbol{R}]_{22}$. Dette gir oss filtermodellen

$$\begin{aligned} \dot{\boldsymbol{x}}^* &= \boldsymbol{F}^*\boldsymbol{x}^* + \boldsymbol{L}^*u^* = \boldsymbol{F}^*\boldsymbol{x}^* + \boldsymbol{L}^*(\boldsymbol{A}\boldsymbol{x} + v^*) \\ \boldsymbol{B}\boldsymbol{z}_k &= \boldsymbol{H}^*\boldsymbol{x}_k^* + \boldsymbol{B}\boldsymbol{w}_k^* \end{aligned}$$

der vi har benyttet matrisene $\boldsymbol{A} = [0, 0, 1]$ og $\boldsymbol{B} = [1, 0]$ for å uttrykke modellen på en kompakt måte. De andre matrisene er som i seksjon 7.1.

Feillikningene og kovariansutviklingen må i dette tilfellet modifiseres grunnet den nye koblingen mellom simuleringsmodellen og filtermodellen. Tidsoppdateringen av estimeringsfeilen blir

$$\begin{aligned} \dot{\bar{\boldsymbol{e}}} &= \boldsymbol{N}\dot{\boldsymbol{x}} - \dot{\bar{\boldsymbol{x}}} \\ &= \boldsymbol{N}\boldsymbol{F}\boldsymbol{x} + \boldsymbol{N}\boldsymbol{L}u + \boldsymbol{N}\boldsymbol{G}v - \boldsymbol{F}^*\bar{\boldsymbol{x}} - \boldsymbol{A}\boldsymbol{x} \\ &= \boldsymbol{F}^*\left(\boldsymbol{N}\boldsymbol{x} - \boldsymbol{x}^*\right) + \left(\boldsymbol{N}\boldsymbol{F} - \boldsymbol{F}^*\boldsymbol{N} - \boldsymbol{A}\right)\boldsymbol{x} + \boldsymbol{N}\boldsymbol{L}u + \boldsymbol{N}\boldsymbol{G}v \\ &= \boldsymbol{F}^*\bar{\boldsymbol{e}} + \left(\boldsymbol{N}\boldsymbol{F} - \boldsymbol{F}^*\boldsymbol{N} - \boldsymbol{A}\right)\boldsymbol{x} \end{aligned}$$

mens måleoppdateringen av estimeringsfeilen blir

$$\begin{aligned} \hat{\boldsymbol{e}}_k &= \boldsymbol{N}\boldsymbol{x}_k - \hat{\boldsymbol{x}}_k^* \\ &= \boldsymbol{N}\boldsymbol{x}_k - \bar{\boldsymbol{x}}_k^* - \boldsymbol{K}_k^*\left(\boldsymbol{B} * \boldsymbol{z}_k - \boldsymbol{H}^*\bar{\boldsymbol{x}}_k\right) \\ &= \left(\boldsymbol{I} - \boldsymbol{K}_k^*\boldsymbol{H}^*\right)\bar{\boldsymbol{e}}_k - \boldsymbol{K}_k^*\left(\boldsymbol{B}\boldsymbol{H} - \boldsymbol{H}^*\boldsymbol{N}\right)\boldsymbol{x}_k - \boldsymbol{K}_k^*\boldsymbol{B}\boldsymbol{w}_k. \end{aligned}$$

Ved å la

$$\boldsymbol{F}_a = \begin{bmatrix} \boldsymbol{F}^* & \boldsymbol{N}\boldsymbol{F} - \boldsymbol{F}^*\boldsymbol{N} - \boldsymbol{A} \\ 0 & \boldsymbol{F} \end{bmatrix} \text{ og } \boldsymbol{H}_a = \begin{bmatrix} \boldsymbol{H}^* & \boldsymbol{B}\boldsymbol{H} - \boldsymbol{H}^*\boldsymbol{N} \end{bmatrix}$$

kan vi også her bruke Likning (5) og Likning (6) til å beregne den sanne estimeringsfeilens kovarians.

Figur 17 til Figur 19 viser resultatene når optimalt og suboptimalt Kalmanfilter kjøres på det ovenstående systemet med $R = \mathrm{diag}(1,\ 0.01^2)$. For å illustrere klarere hva som skjer viser vi også resultater for $R = \mathrm{diag}(1,\ 0.1^2)$ og $\tilde{Q} = 10 \cdot 0.1^2$ i Figur 20 til Figur 22. Blå kurver i feilplottene viser sann usikkerhet mens grønne kurver viser Monte-Carlo beregnet usikkerhet for det suboptimale filteret. Filterets egen beregning av usikkerheten er vist som røde kurver. Det optimale filterets usikkerhet er også vist som svarte kurver. I Figur 17 og Figur 20 viser blå kurve den sanne verdien av x_2, mens den svarte viser det optimale filterets prediksjon og den røde viser det suboptimale filterets prediksjon.

Det første som slår en er at vårt suboptimale filter fungerer bedre enn det optimale og dermed snarere er superoptimalt. Videre ser en i Figur 22 at hastighetsfeilen faktisk synker under tidsoppdatering og stiger under måleoppdatering. Forklaringen på denne uventede oppførselen ligger selvsagt i at filteret gis mer informasjon enn den som kommer via de formelle målingene $\boldsymbol{Bz}_k$. Likningen for x_2^* viser at vi over tid vil å få en svært god prediksjon $\bar{x}_2$ uten å bruke måleoppdatering overhodet:

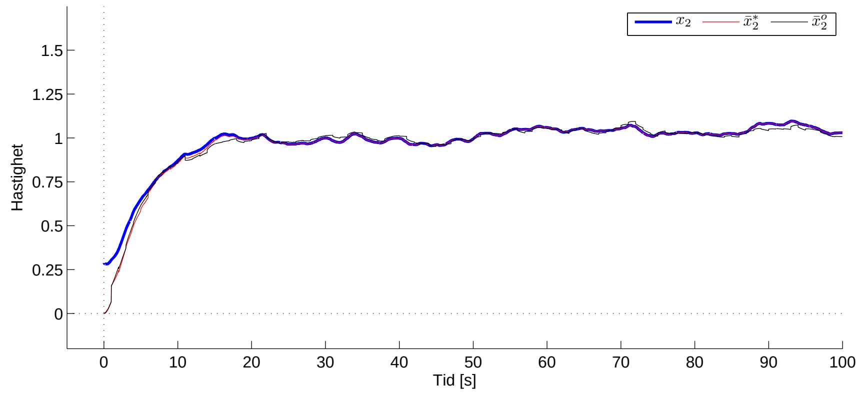
$$\begin{aligned}
\dot{x}_2^* &= -\frac{1}{T_2}x_2^* + \frac{1}{T_2}(x_3 + v^*) \\
\Rightarrow x_2^*(t) &= e^{-\frac{t}{T_2}}x_2^*(0) + \frac{1}{T_2}\int_0^t e^{-\frac{t}{T_2}}x_3(s)\mathrm{d}s + \frac{1}{T_2}\int_0^t e^{-\frac{t}{T_2}}v^*(s)\mathrm{d}s \\
&\to \frac{1}{T_2}\int_0^t e^{-\frac{t}{T_2}}x_3(s)\mathrm{d}s + \frac{1}{T_2}\int_0^t e^{-\frac{t}{T_2}}v^*(s)\mathrm{d}s.
\end{aligned}$$

Med andre ord så avhenger $\bar{x}_2^*$ i det lange løp kun av x_3 og ikke av intialbetingelsen $\hat{x}_2^*(0)$. Prosesstøyen i Likning (1) er innbakt i x_3 og bidrar følgelig ikke til usikkerheten. Usikkerheten i $\bar{x}_2^*$ skyldes derfor kun den beskjedne målestøyen knyttet til målingen av x_3.

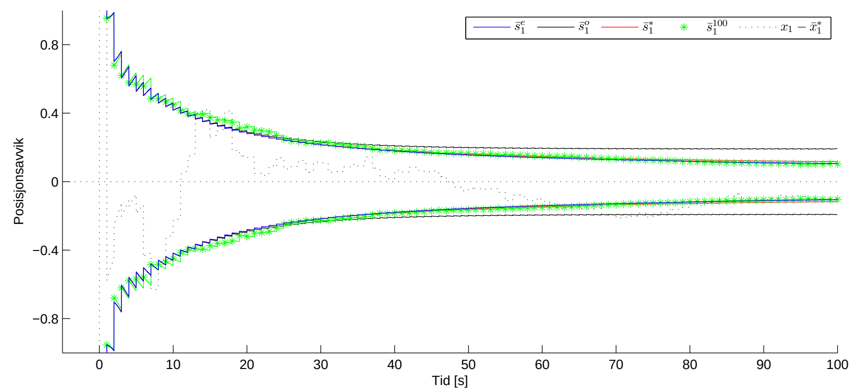
For x_1^* sin del har vi imidlertid ingen dempning og filtrering må til for å estimere denne. Når Kalmanfilteret gjør en måleoppdatering for å estimere $\hat{x}_1^*$ vil det også oppdatere x_2^*, uten å ta hensyn til at vi allerede gjennom tidsoppdateringen har fått masse informasjon om x_2^*. Derfor øker måleoppdateringen usikkerheten for x_2^*.

Vi har som sagt antatt at pådraget u^* kan betraktes som kontinuerlig. En naturlig videreførelse av det som er gjort her ville være å erkjenne at u^* egentlig er diskret og modellere denne som stykkevis konstant mellom hver måling av x_3. Dette er problematisk siden vi da ville måtte ta hensyn til tidskorrelasjoner, og følgelig ikke gjort her.

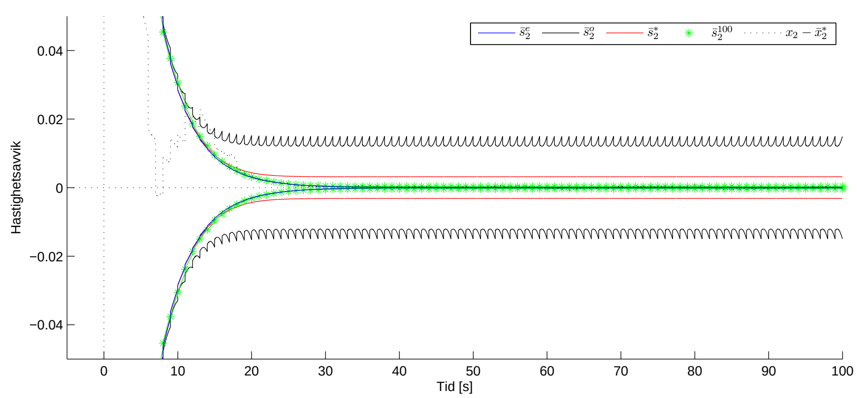
Resultatene i denne deloppgaven ble generert med Matlab-scriptet i Listing 8.



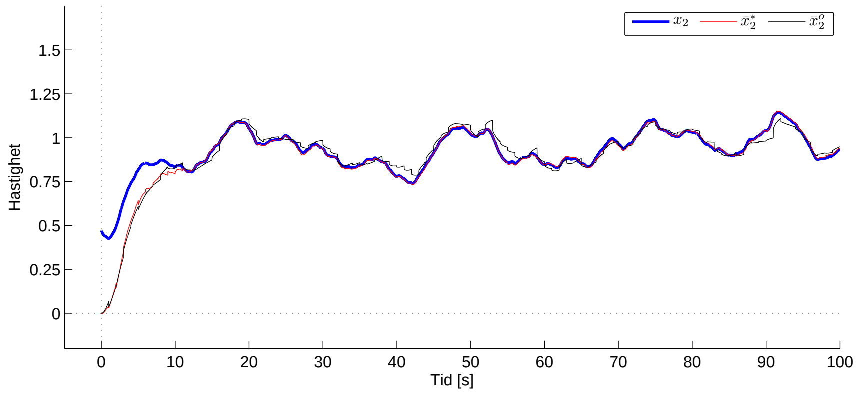
Figur 17: Prediktert hastighet for optimalt og suboptimalt Kalmanfilter



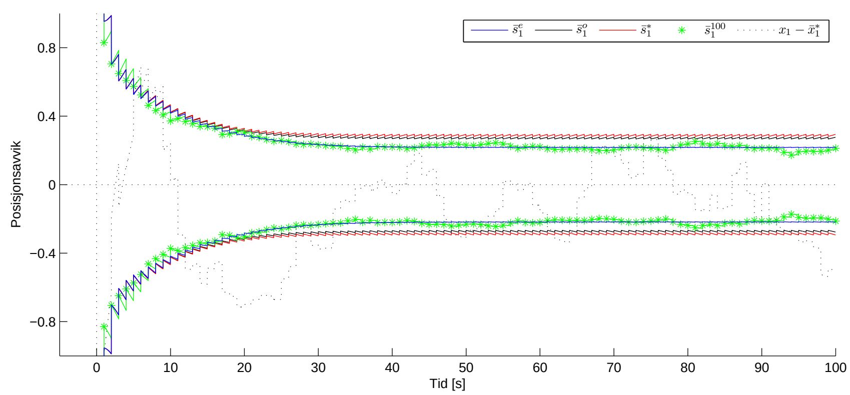
Figur 18: Sann og estimert usikkerhet for posisjon



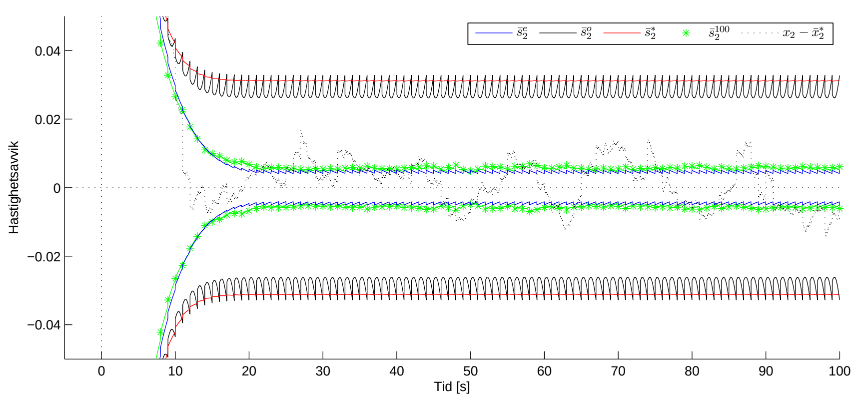
Figur 19: Sann og estimert usikkerhet for hastighet



Figur 20: Predikteret hastighet for optimalt og suboptimalt Kalmanfilter



Figur 21: Sann og estimert usikkerhet for posisjon



Figur 22: Sann og estimert usikkerhet for hastighet

Referanser

Gelb, A. (1974), *Applied Optimal Estimation*, MIT Press.

Loan, C. F. V. (1978), 'Computing integrals involving the matrix exponential', *IEEE Transactions on Automatic Control* **23**, 395–404.

8 Kildekode

All Matlab-kode brukt i dette prosjektet er gjengitt her.

Listing 1: discretize.m

```
function [Phi,Lambda,S,Ga] = discretize(F,L,G,Q,d)

%Flow matrix Phi
Phi = expm(F*d);

%Input matrix Lambda
sf = size(F);
sl = size(L);
A = [F,L;zeros(sl(2),sf(2)+sl(2))];
Loan1 = expm(A*d);
Lambda = Loan1(1:sl(1),sf(2)+1:sf(2)+sl(2));

%Noise matrices Gamma and S
Qc = G*Q*G';
Qc = (Qc+Qc')/2;
dim = size(F,1);
Loan2 = expm([-F,G*Q*G';zeros(dim),F']*d);
G2 = Loan2(1:dim,dim+1:2*dim);
F3 = Loan2(dim+1:2*dim,dim+1:2*dim);

%Calculate S = Gamma*Gamma'
A = F3'*G2;
S =(A+A')/2;
if(nargout == 4)
    %Cholesky factorization to obtain Gamma
    %If not possible use only S
    Ga = chol(A)';
end
```

Listing 2: kalman.m

```
function [xbar,Pbar,xhat,Phat,Karray] = kalman(z,u,xhatzero,Phatzero,Phi,Lambda,S,R,H,mp,M)


%Check consistency of time parameters
if(mod(M,mp)~= 0)
    error('# M is not a multiple of measurement period');
end

d = size(xhatzero,1);

%Initialise resulting arrays
xbar = zeros(d,M);
Pbar = zeros(d,d,M);
Pbar(:,:,1) = Phatzero;
Phat = zeros(d,d,floor(M/mp));
Phat(:,:,1) = Phatzero;
xhat = zeros(d,floor(M/mp));
xhat(:,1) = xhatzero;
xbar(:,1) = xhatzero;
Karray =  zeros(size(Phatzero,1),size(H,1),floor(M/mp));
Karray(:,:,1) = Phatzero*H'*inv(H*Phatzero*H'+R);

%Iterative filtering
i = 1;
```

```
for(k=2:M)

    %If previous time step contained a measurement update from MO
    if(mod(k-1,mp) == 0)
        xbar(:,k) = Phi*xhat(:,i) + Lambda*u(:,k);
        Pbar(:,:,k) = Phi*squeeze(Phat(:,:,i))*Phi' + S;

    %If not update from previous TO
    else
        xbar(:,k) = Phi*xbar(:,k-1) + Lambda*u(:,k);
        Pbar(:,:,k) = Phi*squeeze(Pbar(:,:,k-1))*Phi' + S;
    end

    %If time step contains a measurement use MO
    if(mod(k,mp) == 0)
        i = floor(k/mp);
        P = squeeze(Pbar(:,:,k));
        K = P*H'*inv(H*P*H'+R);
        xhat(:,i) = xbar(:,k) + K*(z(:,i)-H*xbar(:,k));
        Phat(:,:,i) = (eye(d)-K*H)*P;
        Karray(:,:,i) = K;
    end
end
```

Listing 3: covupdate.m

```
function Pbar = covupdate(Karray,Phi,H,M,Pzero,Sa,R,mp)

n = size(Karray,1);
Pbar = zeros(size(Pzero,1),size(Pzero,2),M);
Pbar(:,:,1) = Pzero;

%Check consistency of time parameters
if(mod(M,mp)~= 0)
    error('# M is not a multiple of measurement period');
end

for(k=2:M)

    %TO
    P = Phi*squeeze(Pbar(:,:,k-1))*Phi' + Sa;
    Pbar(:,:,k) = P;

    %MO
    if(mod(k,mp) == 0)
        i = floor(k/mp);
        K = squeeze(Karray(:,:,i));
        Pbar(:,:,k) = (eye(n)-K*H)*P*(eye(n)-K*H)' + K*R*K';
    end
end
```

Listing 4: oppgave4.m

```
%Declare constants
T2 = 5;
T3 = 1;
Q = 2*0.1^2;
R = 1;
tzero = 0;
tf = 100;
d = 0.01;
```

```
%Declare matrices
F = [0,1,0;0,-1/T2,1/T2;0,0,-1/T3];
L = [0;0;1/T3];
G = 1*[0;0;1];
H = [1,0,0];
Pzero = [1,0,0;0,0.1^2,0;0,0,0.1^2];

%Get matrices for discrete system
[Phi,Lambda,S,Ga] = discretize(F,L,G,Q,d);

%-----------------------------------------------------------------------
%Simulate using discretized system--------------------------------------
%-----------------------------------------------------------------------
M = 10000;
x = zeros(3,M);
u = ones(1,M);
v = randn(3,M);
xzero = chol(Pzero)*randn(3,1);
x = simulate(xzero,Pzero,u,v,Phi,Lambda,Ga,M);

%-----------------------------------------------------------------------
%Estimate true states using Kalman filter-------------------------------
%-----------------------------------------------------------------------
mp = 100;
w = sqrt(R)*randn(1,floor(M/mp));
z = H*x(:,find(mod([1:M],mp) == 0))+w;
xhatzero = chol(Pzero)*randn(3,1);
Phatzero = Pzero;
[xbar,Pbar,xhat,Phat,K] = kalman(z,u,xhatzero,Phatzero,Phi,Lambda,S,R,H,mp,M);

%-----------------------------------------------------------------------
%Plot results-----------------------------------------------------------
%-----------------------------------------------------------------------

%Plot x_2 from Kalman filter
fig1 = figure;
hold on;
xticks = [0:1000:M];
yticks = [0:0.25:1.5];
xmarks = [0:10:100];
ymarks = yticks;
xlab = 'Tid [s]';
ylab = 'Hastighet';
plotrange = [-500,10000,-0.2,1.75];
axis(plotrange);
xlabel(xlab,'Fontsize',12);
ylabel(ylab,'Fontsize',12);
set(gca,'XTick',xticks);
set(gca,'YTick',yticks);
set(gca,'XTickLabel',xmarks,'Fontsize',12);
set(gca,'YTickLabel',ymarks);
line(plotrange(1:2),[0,0],'Linestyle',':','color','k');
line([0,0],plotrange(3:4),'Linestyle',':','color','k');
h2 = line([0,0],[0,1],'Linestyle','--','color','r','linewidth',2);
line([0,10000],[1,1],'Linestyle','--','color','r','linewidth',2);
h1 = plot(x(2,:),'linewidth',1);
h3 = plot(xbar(2,:),'g','linewidth',2);
h4 = plot([100:100:10000],xhat(2,:),'k*');
hold off;
L1 = legend([h2,h1,h3,h4],'$u(t)$','$x_2(t)$','$\bar{x}_2(t)$','$\hat{x}_2(t_k)$',...
    'Orientation','Horizontal');
```

```
set(L1,'Interpreter','latex','Fontsize',14); % LaTeX in figures
set(fig1,'Position',[100,678,1000,420]);
%title('Estimert og sann hastighet','FontSize',14);

%Plot x_2-\bar{x}_2 from Kalman filter
fig2 = figure;
hold on;
plotrange = [-500,10000,-0.12,0.12];
yticks = [-0.1:0.025:0.1];
ymarks = yticks;
xlabel(xlab,'Fontsize',12);
ylabel(ylab,'Fontsize',12);
axis(plotrange);
set(gca,'XTick',xticks);
set(gca,'YTick',yticks);
set(gca,'XTickLabel',xmarks,'Fontsize',12);
set(gca,'YTickLabel',ymarks);
line(plotrange(1:2),[0,0],'Linestyle',':','color','k');
line([0,0],plotrange(3:4),'Linestyle',':','color','k');
f1 = plot(x(2,:)-xbar(2,:),'k');
f2 = plot(sqrt(squeeze(Pbar(2,2,:))),'b');
f3 = plot(-sqrt(squeeze(Pbar(2,2,:))),'b');
hold off;
set(fig2,'Position',[100,678,1000,420]);
L2 = legend([f1,f2],'$x_2(t)-\bar{x}_2(t)$','$\pm␣\bar{s}_2(t)$','Orientation','Horizontal');
set(L2,'Interpreter','latex','Fontsize',14);
%title('Residualer for tidsoppdatering av hastighet','FontSize',14);

%Plot x_2-\hat{x}_2 from Kalman filter
xextracted = x(:,find(mod([1:M],mp) == 0));
fig3 = figure;
hold on;
plotrange = [-5,100,-0.12,0.12];
yticks = [-0.1:0.025:0.1];
xticks = xmarks;
ymarks = yticks;
xlabel(xlab,'Fontsize',12);
ylabel(ylab,'Fontsize',12);
axis(plotrange);
set(gca,'XTick',xticks);
set(gca,'YTick',yticks);
set(gca,'XTickLabel',xmarks,'Fontsize',12);
set(gca,'YTickLabel',ymarks);
line(plotrange(1:2),[0,0],'Linestyle',':','color','k');
line([0,0],plotrange(3:4),'Linestyle',':','color','k');
g1 = plot(xextracted(2,:)-xhat(2,:),'k*');
g2 = plot(sqrt(squeeze(Phat(2,2,:))),'bx');
g3 = plot(-sqrt(squeeze(Phat(2,2,:))),'bx');
hold off;
set(fig3,'Position',[100,678,1000,420]);
L3 = legend([g1,g2],'$x_2(t)-\hat{x}_2(t_k)$','$\pm␣\hat{s}_2(t_k)$','Orientation','Horizonta
set(L3,'Interpreter','latex','Fontsize',14);
%title('Residualer for måleoppdatering av hastighet','FontSize',14);

%Plot x_2 from simulation
fig4 = figure;
hold on;
xticks = [0:1000:M];
yticks = [0:0.25:1.5];
xmarks = [0:10:100];
ymarks = yticks;
xlab = 'Tid␣[s]';
```

```
ylab = 'Hastighet';
plotrange = [-500,10000,-0.2,1.75];
axis(plotrange);
xlabel(xlab,'Fontsize',12);
ylabel(ylab,'Fontsize',12);
set(gca,'XTick',xticks);
set(gca,'YTick',yticks);
set(gca,'XTickLabel',xmarks,'Fontsize',12);
set(gca,'YTickLabel',ymarks);
line(plotrange(1:2),[0,0],'Linestyle',':','color','k');
line([0,0],plotrange(3:4),'Linestyle',':','color','k');
h2 = line([0,0],[0,1],'Linestyle','--','color','r','linewidth',2);
line([0,10000],[1,1],'Linestyle','--','color','r','linewidth',2);
h1 = plot(x(2,:),'linewidth',1);
hold off;
L1 = legend([h2,h1],'$u(t)$','$x_2(t)$',...
    'Orientation','Horizontal');
set(L1,'Fontsize',16,'Interpreter','latex'); % LaTeX in figures
set(fig4,'Position',[100,678,1000,420]);
%title('Hastighet i tilfelle med støy','FontSize',16);
```

Listing 5: oppgave5.m

```
%Declare constants
T2 = 5;
T3 = 1;
Q = 2*0.1^2;
R = 1;
tzero = 0;
tf = 100;
d = 0.01;
M = 10000;
mp = 100;

%Declare matrices
F = [0,1,0;0,-1/T2,1/T2;0,0,-1/T3];
L = [0;0;1/T3];
G = [0;0;1];
H = [1,0,0];
Pzero = [1,0,0;0,0.1^2,0;0,0,0.1^2];
xzero = chol(Pzero)*randn(3,1);
Phatzero = Pzero;

%Plotting stuff
xticks = [0:1000:M];
xmarks = [0:10:100];
xlab = 'Tid [s]';

%Get matrices for discrete system
[Phi,Lambda,S,Ga] = discretize(F,L,G,Q,d);

%Declare variables
x = zeros(3,M);
u = ones(1,M);
v = randn(3,M);


%-------------------------------------------------------------------------
%Monte Carlo Simulation---------------------------------------------------
%-------------------------------------------------------------------------
N = 10;
ehatsim = zeros(1,floor(M/mp));
```

```
shatsim = zeros(1,floor(M/mp));
xhatsimarray = zeros(N,floor(M/mp));
xbarsimarray = zeros(N,M);
xmcarray = zeros(N,M);

for(trajectory = 1:N)
    vmc = randn(3,M);
    wmc = sqrt(R)*randn(1,floor(M/mp));
    xzeromc = chol(Pzero)*randn(3,1);

    %Simulate
    xmc = simulate(xzeromc,Pzero,u,vmc,Phi,Lambda,Gamma,M);
    xmcarray(trajectory,:) = xmc(2,:);

    %Get measurements
    zmc = H*xmc(:,find(mod([1:M],mp) == 0)) + wmc;

    %Kalman filter
    [xbar,Pbar,xhat,Phat,K] = kalman(zmc,u,xzeromc,Phatzero,Phi,Lambda,S,R,H,mp,M);

    %Putting stuff into arrays
    xhatsimarray(trajectory,:) = xhat(2,:);
    xbarsimarray(trajectory,:) = xbar(2,:);

    %Computing error means and covariances recursively
    deltavector = xmc(2,find(mod([1:M],mp) == 0))-xhat(2,:) - ehatsim;
    ehatsim = ehatsim + deltavector/trajectory;
    if(trajectory < 2)
        ;
    else
        shatsim = ((trajectory-2)/(trajectory-1))*shatsim + deltavector.*deltavector/trajecto
    end
end

%----------------------------------------------------------------------
%Plot results----------------------------------------------------------
%----------------------------------------------------------------------

%Plot velocity simulations
fig4 = figure;
hold on;
xticks = [0:1000:M];
yticks = [0:0.25:1.5];
xmarks = [0:10:100];
ymarks = yticks;
xlab = 'Tid [s]';
ylab = 'Hastighet';
plotrange = [-500,10000,-0.2,1.75];
axis(plotrange);
xlabel(xlab,'Fontsize',12);
ylabel(ylab,'Fontsize',12);
set(gca,'XTick',xticks);
set(gca,'YTick',yticks);
set(gca,'XTickLabel',xmarks,'Fontsize',12);
set(gca,'YTickLabel',ymarks);
line(plotrange(1:2),[0,0],'Linestyle',':','color','k');
line([0,0],plotrange(3:4),'Linestyle',':','color','k');
h2 = plot(xbarsimarray(1,:),'linewidth',1);
h1 = plot([100:100:10000],xhatsimarray(1,:),'kx');
for(i=2:N)
    plot(xbarsimarray(i,:),'linewidth',1);
    plot([100:100:10000],xhatsimarray(i,:),'kx');
```

```
end
hold off;
L1 = legend([h1,h2],'$\hat{x}_2^i(t_k)$','$\bar{x}_2^i(t)$',...
    'Orientation','Horizontal');
set(L1,'Interpreter','latex','Fontsize',14);
set(fig4,'Position',[100,678,1000,420]);

%Plot velocity difference simulations
fig5 = figure;
hold on;
xticks = [0:1000:M];
plotrange = [-500,10000,-0.20,0.20];
yticks = [-0.20:0.05:0.20];
xmarks = [0:10:100];
ymarks = yticks;
xlab = 'Tid [s]';
ylab = 'Hastighet';
axis(plotrange);
xlabel(xlab,'Fontsize',12);
ylabel(ylab,'Fontsize',12);
set(gca,'XTick',xticks);
set(gca,'YTick',yticks);
set(gca,'XTickLabel',xmarks,'Fontsize',12);
set(gca,'YTickLabel',ymarks);
line(plotrange(1:2),[0,0],'Linestyle',':','color','k');
line([0,0],plotrange(3:4),'Linestyle',':','color','k');
h2 = plot(xmcarray(1,:) - xbarsimarray(1,:),'linewidth',1);
h1 = plot([100:100:10000],xmcarray(1,find(mod([1:M],mp) == 0)) - xhatsimarray(1,:),'k*');
for(i=2:N)
    plot(xmcarray(i,:) - xbarsimarray(i,:),'linewidth',1);
    plot([100:100:10000],xmcarray(i,find(mod([1:M],mp) == 0)) - xhatsimarray(i,:),'k*');
end
hold off;
L1 = legend([h1,h2],'$x_2^i(t_k)-\hat{x}_2^i(t_k)$','$x_2^i(t)-\bar{x}_2^i(t)$',...
    'Orientation','Horizontal');
set(L1,'Interpreter','latex','Fontsize',14);
set(fig5,'Position',[100,678,1000,420]);

%Plot velocity error and covariance for N=10;
fig6 = figure;
hold on;
plotrange = [-500,10000,-0.1,0.1];
yticks = [-0.1:0.02:0.1];
ymarks = yticks;
axis(plotrange);
xlabel(xlab,'Fontsize',12);
ylabel(ylab,'Fontsize',12);
set(gca,'XTick',xticks);
set(gca,'YTick',yticks);
set(gca,'XTickLabel',xmarks,'Fontsize',12);
set(gca,'YTickLabel',ymarks);
line(plotrange(1:2),[0,0],'Linestyle',':','color','k');
line([0,0],plotrange(3:4),'Linestyle',':','color','k');
f1 = plot([100:100:10000],ehatsim,'k');
f2 = plot([100:100:10000],sqrt(shatsim),'b');
plot([100:100:10000],-sqrt(shatsim),'b');
f3 = plot([100:100:10000],sqrt(squeeze(Phat(2,2,:))),'g*');
plot([100:100:10000],-sqrt(squeeze(Phat(2,2,:))),'g*');
hold off;
set(fig6,'Position',[100,678,1000,420]);
L2 = legend([f1,f2,f3],'$\hat{m}^N_2(t_k)$','$\hat{s}^N_2(t_k)$','$\hat{s}_2(t_k)$','Orientation','Hor
set(L2,'Interpreter','latex','Fontsize',14);
```

Listing 6: oppgave6.m

```
%Declare constants
T2 = 5;
T3 = 1;
Q = 2*0.1^2;
R = 1;
tzero = 0;
tf = 100;
d = 0.01;
M = 10000;
mp = 100;

%Declare matrices
F = [0,1,0;0,-1/T2,1/T2;0,0,-1/T3];
L = [0;0;1/T3];
G = [0;0;1];
H = [1,0,0];
Pzero = [1,0,0;0,0.1^2,0;0,0,0.1^2];
xzero = chol(Pzero)*randn(3,1);
Phatzero = Pzero;

%Get matrices for discrete system
[Phi,Lambda,S,Ga] = discretize(F,L,G,Q,d);

%Declare variables
x = zeros(3,M);
u = ones(1,M);
v = randn(3,M);


%------------------------------------------------------------------------
%Make error budget-------------------------------------------------------
%------------------------------------------------------------------------

%Run Kalman filter to get position standard deviation and Kalman gain
x = simulate(xzero,Pzero,u,v,Phi,Lambda,Gamma,M);
w = sqrt(R)*randn(1,floor(M/mp));
z = H*x(:,find(mod([1:M],mp) == 0))+w;
xhatzero = chol(Pzero)*randn(3,1);
Phatzero = Pzero;
[xbar,Pbar,xhat,Phat,K] = kalman(z,u,xhatzero,Phatzero,Phi,Lambda,S,R,H,mp,M);
std1 = squeeze(Pbar(1,1,:));

%Update error covariances based on Kalman gain
ErrTotal = covupdate(K,Phi,H,M,Pzero,S,R);
ErrPzero = covupdate(K,Phi,H,M,Pzero,0*S,0*R);
ErrGamma = covupdate(K,Phi,H,M,0*Pzero,S,0*R);
ErrR = covupdate(K,Phi,H,M,0*Pzero,0*S,R);

%------------------------------------------------------------------------
%Plot results------------------------------------------------------------
%------------------------------------------------------------------------

%Plot error budget for position
fig8 = figure;
xticks = [0:1000:M];
xmarks = [0:10:100];
xlab = 'Tid [s]';
hold on;
plotrange = [-500,10000,-0.1,1];
yticks = [0.0:0.2:0.8];
ymarks = yticks;
```

```
xlabel(xlab,'Fontsize',12);
ylabel('Posisjonsavvik','Fontsize',12);
set(gca,'XTick',xticks);
set(gca,'YTick',yticks);
set(gca,'XTickLabel',xmarks,'Fontsize',12);
set(gca,'YTickLabel',ymarks);
axis(plotrange);
line(plotrange(1:2),[0,0],'Linestyle',':','color','k');
line([0,0],plotrange(3:4),'Linestyle',':','color','k');
f1 = plot(sqrt(squeeze(ErrTotal(1,1,:))),'k');
f2 = plot(sqrt(squeeze(ErrPzero(1,1,:))),'b');
f3 = plot(sqrt(squeeze(ErrGamma(1,1,:))),'r');
f4 = plot(sqrt(squeeze(ErrR(1,1,:))),'g');
hold off;
set(fig8,'Position',[100,678,1000,420]);
L2 = legend([f1,f2,f3,f4],'$s^e$','$s^e_{\hat{P}_0}$','$s^e_{\Gamma}$','$s^e_{R}$',...
    'Orientation','Horizontal');
set(L2,'Interpreter','latex','Fontsize',14);
%title('Feilbudsjett for posisjon','FontSize',14);

%Plot error budget for velocity
fig9 = figure;
hold on;
plotrange = [-500,10000,-0.05,0.1];
yticks = [0.0:0.025:0.1];
ymarks = yticks;
xlabel(xlab,'Fontsize',12);
ylabel('Hastighetsavvik','Fontsize',12);
set(gca,'XTick',xticks);
set(gca,'YTick',yticks);
set(gca,'XTickLabel',xmarks,'Fontsize',12);
set(gca,'YTickLabel',ymarks);
axis(plotrange);
line(plotrange(1:2),[0,0],'Linestyle',':','color','k');
line([0,0],plotrange(3:4),'Linestyle',':','color','k');
f1 = plot(sqrt(squeeze(ErrTotal(2,2,:))),'k');
f2 = plot(sqrt(squeeze(ErrPzero(2,2,:))),'b');
f3 = plot(sqrt(squeeze(ErrGamma(2,2,:))),'r');
f4 = plot(sqrt(squeeze(ErrR(2,2,:))),'g');
hold off;
set(fig9,'Position',[100,678,1000,420]);
L2 = legend([f1,f2,f3,f4],'$s^e$','$s^e_{\hat{P}_0}$','$s^e_{\Gamma}$','$s^e_{R}$',...
    'Orientation','Horizontal');
set(L2,'Interpreter','latex','Fontsize',14);
%title('Feilbudsjett for hastighet','FontSize',14);

%Plot error budget for x_3
fig10 = figure;
hold on;
plotrange = [-500,10000,-0.05,0.2];
yticks = [0.0:0.05:0.2];
ymarks = yticks;
xlabel(xlab,'Fontsize',12);
ylabel('Ankerstrømsavvik','Fontsize',12);
set(gca,'XTick',xticks);
set(gca,'YTick',yticks);
set(gca,'XTickLabel',xmarks,'Fontsize',12);
set(gca,'YTickLabel',ymarks);
axis(plotrange);
line(plotrange(1:2),[0,0],'Linestyle',':','color','k');
line([0,0],plotrange(3:4),'Linestyle',':','color','k');
f1 = plot(sqrt(squeeze(ErrTotal(3,3,:))),'k');
```

```
f2 = plot(sqrt(squeeze(ErrPzero(3,3,:))),'b');
f3 = plot(sqrt(squeeze(ErrGamma(3,3,:))),'r');
f4 = plot(sqrt(squeeze(ErrR(3,3,:))),'g');
hold off;
set(fig10,'Position',[100,678,1000,420]);
L2 = legend([f1,f2,f3,f4],'$s^e$','$s^e_{\hat{P}_0}$','$s^e_{\Gamma}$','$s^e_{R}$',...
    'Orientation','Horizontal');
set(L2,'Interpreter','latex','Fontsize',14);
%title('Feilbudsjett for ankerstrøm','FontSize',14);

%Plot total error for x_1
fig11 = figure;
hold on;
plotrange = [-500,10000,-0.1,1];
yticks = [0.0:0.2:0.8];
ymarks = yticks;
xlabel(xlab);
ylabel('Posisjonsavvik');
axis(plotrange);
set(gca,'XTick',xticks);
set(gca,'YTick',yticks);
set(gca,'XTickLabel',xmarks);
set(gca,'YTickLabel',ymarks);
line(plotrange(1:2),[0,0],'Linestyle',':','color','k');
line([0,0],plotrange(3:4),'Linestyle',':','color','k');
f1 = plot(sqrt(squeeze(Pbar(1,1,:))),'k','Linewidth',2);
f5 = plot(sqrt(squeeze(ErrR(1,1,:))+squeeze(ErrGamma(1,1,:))+squeeze(ErrPzero(1,1,:))),...
    'r','Linestyle',':');
hold off;
set(fig11,'Position',[100,678,1000,420]);
L2 = legend([f1,f5],'$\sqrt{\sum_(s^e_{11})^2}$','$\sqrt{[\bar{P}]_{11}}$',...
    'Orientation','Horizontal');
set(L2,'Interpreter','latex','Fontsize',14);
%title('RMS-sum sammenlighet med standardavvik fra KF','FontSize',14);
```

Listing 7: oppgave7.m

```
%Declare constants
T2 = 5;
T3 = 1;
Q = 2*0.1^2;
R = 1;
tzero = 0;
tf = 100;
d = 0.01;
mp = 100;
M = 10000;

%Plotting stuff
xticks = [0:1000:M];
xmarks = [0:10:100];
xlab = 'Tid_[s]';


%Declare matrices
F = [0,1,0;0,-1/T2,1/T2;0,0,-1/T3];
L = [0;0;1/T3];
G = [0;0;1];
H = [1,0,0];
R = eye(size(H,1));
Pzero = [1,0,0;0,0.1^2,0;0,0,0.1^2];
xzero = chol(Pzero)*randn(3,1);
```

```
dim = size(F,1);

%Get matrices for discrete system
[Phi,Lambda,S,Ga] = discretize(F,L,G,Q,d);

%------------------------------------------------------------------------
%Simulate using discretized system---------------------------------------
%------------------------------------------------------------------------
M = 10000;
x = zeros(3,M);
u = ones(1,M);
v = randn(dim,M);
x = simulate(xzero,Pzero,u,v,Phi,Lambda,Ga,M);

%------------------------------------------------------------------------
%Estimate true states using Kalman filter--------------------------------
%------------------------------------------------------------------------
mp = 100;
w = sqrt(R)*randn(size(H,1),floor(M/mp));
z = H*x(:,find(mod([1:M],mp) == 0))+w;
xhatzero = 0*chol(Pzero)*randn(3,1);
Phatzero = Pzero;
[xbar,Pbar,xhat,Phat,K] = kalman(z,u,xhatzero,Phatzero,Phi,Lambda,S,R,H,mp,M);

%------------------------------------------------------------------------
%Suboptimal model--------------------------------------------------------
%------------------------------------------------------------------------

%Declare matrices.
Fsub = [0,1;0,-1/T2];
Lsub = [0;1/T2];
Gsub = [0;1/T2];
Hsub = [1,0];
Qsub = T3*Q/2;
Rsub = 1;
Pzsub = Pzero(1:2,1:2);
xzsub = 0*chol(Pzsub)*randn(2,1);
dimsub = size(Fsub,1);

%Discretise suboptimal system.
[Phisub,Lambdasub,Ssub,Gammasub] = discretize(Fsub,Lsub,Gsub,Qsub,d);

%No simulation for filter model:
%Use measurements from optimal system also for suboptimal filter.
zsub = z;

%Filter.
Phzsub = Pzero(1:2,1:2);
xhzsub = chol(Phzsub)*randn(dimsub,1);
[xbarsub,Pbarsub,xhatsub,Phasubt,Ksub]...
    = kalman(zsub,u,xhzsub,Phzsub,Phisub,Lambdasub,Ssub,Rsub,Hsub,mp,M);


%------------------------------------------------------------------------
%Estimate true standard deviations s^e-----------------------------------
%------------------------------------------------------------------------
N = [1,0,0;0,1,0];
Paz = [N*Phatzero*N',N*Phatzero;Phatzero*N',Phatzero];
Ka = cat(1,Ksub,K*0);
Ga = [N*G;G];
deltaH = H-Hsub*N;
deltaF = N*F-Fsub*N;
```

```
Ha = [Hsub, deltaH];
Fa = [Fsub, deltaF; deltaF'*0, F];

Phia = expm(Fa*d);
Sa = kp2dpGa(Fa, Ga, Q, d);
Pbara = covupdate(Ka, Phia, Ha, M, Paz, Sa, R, mp);


%-------------------------------------------------------------------------
%Plot stuff--------------------------------------------------------------
%-------------------------------------------------------------------------


%Plot position standard deviations
fig11 = figure;
hold on;
plotrange = [-500,10000,-1,1];
yticks = [-0.8:0.4:0.8];
xticks = [0:1000:M];
xmarks = [0:10:100];
ymarks = yticks;
xlab = 'Tid [s]';
xlabel(xlab, 'Fontsize', 12);
ylabel('Posisjonsavvik', 'Fontsize', 12);
axis(plotrange);
set(gca, 'XTick', xticks);
set(gca, 'YTick', yticks);
set(gca, 'XTickLabel', xmarks, 'Fontsize', 12);
set(gca, 'YTickLabel', ymarks, 'Fontsize', 12);
line(plotrange(1:2), [0,0], 'Linestyle', ':', 'color', 'k');
line([0,0], plotrange(3:4), 'Linestyle', ':', 'color', 'k');
f5 = plot(sqrt(squeeze(Pbara(1,1,:))), 'b');
plot(-sqrt(squeeze(Pbara(1,1,:))), 'b');
f1 = plot(x(1,:) - xbarsub(1,:), 'k')
f2 = plot(sqrt(squeeze(Pbarsub(1,1,:))), 'r');
plot(-sqrt(squeeze(Pbarsub(1,1,:))), 'r');
hold off;
set(fig11, 'Position', [100,478,1000,420]);
L2 = legend([f2, f5, f1], '$\bar{s}_1^*$', '$\bar{s}_1^e$', '$x_1 - \bar{x}_1^*$', ...
    'Orientation', 'Horizontal');
set(L2, 'Interpreter', 'latex', 'Fontsize', 14);
%title('Standardavvik for posisjon for optimalt og suboptimalt filter', 'FontSize', 14);


%Plot velocity standard deviations
fig12 = figure;
hold on;
plotrange = [-500,10000,-0.14,0.14];
yticks = [-0.12:0.04:0.12];
ymarks = yticks;
xlabel(xlab, 'Fontsize', 12);
ylabel('Hastighetsavvik', 'Fontsize', 12);
axis(plotrange);
set(gca, 'XTick', xticks);
set(gca, 'YTick', yticks);
set(gca, 'XTickLabel', xmarks, 'Fontsize', 12);
set(gca, 'YTickLabel', ymarks, 'Fontsize', 12);
line(plotrange(1:2), [0,0], 'Linestyle', ':', 'color', 'k');
line([0,0], plotrange(3:4), 'Linestyle', ':', 'color', 'k');
f5 = plot(sqrt(squeeze(Pbara(2,2,:))), 'b');
plot(-sqrt(squeeze(Pbara(2,2,:))), 'b');
f1 = plot(x(2,:) - xbarsub(2,:), 'k')
```

```
f2 = plot(sqrt(squeeze(Pbarsub(2,2,:))),'r');
plot(-sqrt(squeeze(Pbarsub(2,2,:))),'r');
hold off;
set(fig12,'Position',[100,478,1000,420]);
L2 = legend([f2,f5,f1],'$\bar{s}_2^*$','$\bar{s}_2^e$','$x_2 - \bar{x}_2^*$',...
    'Orientation','Horizontal');
set(L2,'Interpreter','latex','Fontsize',14);
%title('Standardavvik for hastighet for optimalt og suboptimalt filter','FontSize',14);

%Plot velocities
fig13 = figure;
hold on;
plotrange = [-500,10000,-0.2,1.75];
yticks = [0:0.25:1.5];
ymarks = yticks;
xlabel(xlab,'Fontsize',12);
ylabel('Hastighet','Fontsize',12);
axis(plotrange);
set(gca,'XTick',xticks);
set(gca,'YTick',yticks);
set(gca,'XTickLabel',xmarks,'Fontsize',12);
set(gca,'YTickLabel',ymarks,'Fontsize',12);
line(plotrange(1:2),[0,0],'Linestyle',':','color','k');
line([0,0],plotrange(3:4),'Linestyle',':','color','k');
f5 = plot(x(2,:),'b');
f1 = plot(xbarsub(2,:),'k');
f2 = plot(xbar(2,:),'r');
hold off;
set(fig13,'Position',[100,478,1000,420]);
L2 = legend([f5,f1,f2],'$x_2$','$\bar{x}_2^*$','$\bar{x}_2^o$',...
    'Orientation','Horizontal');
set(L2,'Interpreter','latex','Fontsize',14);
%title('Sann hastighet og hastighet for optimalt og suboptimalt filter','FontSize',14);
```

Listing 8: oppgave72.m

```
%Declare constants
T2 = 5;
T3 = 1;
Q = 2*0.1^2;
Ruz = 1*0.01^2;
tzero = 0;
tf = 100;
d = 0.01;
mp = 100;
M = 10000;
iterations = 100;

%Plotting stuff
xticks = [0:1000:M];
xmarks = [0:10:100];
xlab = 'Tid [s]';

%Declare matrices
F = [0,1,0;0,-1/T2,1/T2;0,0,-1/T3];
L = [0,0,0;0,0,0;0,0,1/T3];
G = [0;0;1];
H = [1,0,0;0,0,1];
R = [1,0;0,0.0001];
Pzero = 10*[1,0,0;0,0.1^2,0;0,0,0.1^2];
xzero = chol(Pzero)*randn(3,1);
xzero(2) = 0.4;
```

```
dim = size(F,1);

%Get matrices for discrete system
[Phi,Lambda,S,Gam] = discretize(F,L,G,Q,d);

%------------------------------------------------------------------------
%Simulate using discretized system---------------------------------------
%------------------------------------------------------------------------

x = zeros(3,M);
u = [zeros(2,M);ones(1,M)];
v = randn(dim,M);
x = simulate(xzero,Pzero,u,v,Phi,Lambda,Gam,M);

%------------------------------------------------------------------------
%Estimate true states using Kalman filter--------------------------------
%------------------------------------------------------------------------
w = sqrt(R)*randn(size(H,1),floor(M/mp));
z = H*x(:,find(mod([1:M],mp) == 0))+w;
xhatzero = 0*chol(Pzero)*randn(3,1);
Phatzero = Pzero;
[xbar,Pbar,xhat,Phat,K] = kalman(z,u,xhatzero,Phatzero,Phi,Lambda,S,R,H,mp,M);

%------------------------------------------------------------------------
%Make suboptimal Kalman filter-------------------------------------------
%------------------------------------------------------------------------
Fsub = [0,1;0,-1/T2];
Lsub = [0,0;0,1/T2];
Gsub = [0;1/T2];
Hsub = [1,0];
Rsub = 1;
Qsub = Ruz;% + T3*Q/(2*T2^2);

usub = [zeros(1,M);x(3,:) + chol(Ruz)*randn(size(x(3,:)))];

zsub = z(1,:);

[Phisub,Lambdasub,Ssub,Gammasub] = discretize(Fsub,Lsub,Gsub,Qsub,d);

Phzsub = Pzero(1:2,1:2);
xhzsub = 0*chol(Phzsub)*randn(2,1);
[xbarsub,Pbarsub,xhatsub,Phasubt,Ksub] = kalman(zsub,usub,xhzsub,Phzsub,Phisub,Lambdasub,Ssub


%------------------------------------------------------------------------
%Covariance analysis-----------------------------------------------------
%------------------------------------------------------------------------

N = [1,0,0;0,1,0];
A = [0,0,0;0,0,F(2,3)];
B = [1,0];
Paz = [N*Phatzero*N',N*Phatzero;Phatzero*N',Phatzero];
Ka = [Ksub;zeros(size(K,1),1,size(K,3))];
Ga = [N*G;G];
%Ga = cat(2,[Gsub;zeros(3,1)],[zeros(2,1);G])
deltaH = B*H-Hsub*N;
deltaF = N*F-Fsub*N-A;
Ha = [Hsub,deltaH];
Fa = [Fsub,deltaF;deltaF'*0,F];
Phia = expm(Fa*d);
Qa = Q;
%Qa = blkdiag(Ruz,Q);
```

```
Sa = kp2dpGa(Fa,Ga,Q,d);
Ra = B*R*B';
Pbara = covupdate(Ka,Phia,Ha,M,Paz,Sa,Ra,mp);


%-----------------------------------------------------------------------------------------------
%Monte Carlo Simulation to test covariance analysis-----------------------------------------
%-----------------------------------------------------------------------------------------------


ehatsim1 = zeros(1,floor(M/mp));
ehatsim2 = zeros(1,floor(M/mp));
ebarsim1 = zeros(1,M);
ebarsim2 = zeros(1,M);
shatsim1 = zeros(1,floor(M/mp));
shatsim2 = zeros(1,floor(M/mp));
sbarsim1 = zeros(1,M);
sbarsim2 = zeros(1,M);
xhatsimarray1 = zeros(iterations,floor(M/mp));
xhatsimarray2 = zeros(iterations,floor(M/mp));
xbarsimarray1 = zeros(iterations,M);
xbarsimarray2 = zeros(iterations,M);
xmcarray1 = zeros(iterations,M);
xmcarray2 = zeros(iterations,M);
tic
for(trajectory = 1:iterations)
    vmc = randn(dim,M);
    wmc = sqrt(R)*randn(size(H,1),floor(M/mp));
    xzeromc = 0*chol(Pzero)*randn(dim,1);

    %Simulate
    xmc = simulate(xzeromc,Pzero,u,vmc,Phi,Lambda,Gam,M);
    xmcarray1(trajectory,:) = xmc(1,:);
    xmcarray2(trajectory,:) = xmc(2,:);

    %Get measurements
    zmc = H*xmc(:,find(mod([1:M],mp) == 0)) + wmc;
    zmcsub = zmc(1,:);
    umcsub = [zeros(1,M);xmc(3,:) + chol(Ruz)*randn(size(xmc(3,:)))];

    %Kalman filter


    xhzsub = 0*chol(Phzsub)*randn(2,1);

    [xbarmc,Pbarmc,xhatmc,Phatmc,Kmc] = kalman(zmcsub,umcsub,xhzsub,Phzsub,Phisub,Lambdasub,Ssub,Rsub,

    %Putting stuff into arrays - only storing velocities
    xhatsimarray1(trajectory,:) = xhatmc(1,:);
    xhatsimarray2(trajectory,:) = xhatmc(2,:);
    xbarsimarray1(trajectory,:) = xbarmc(1,:);
    xbarsimarray2(trajectory,:) = xbarmc(2,:);

    %Computing error means and covariances recursively
    deltahatvector1 = xmc(1,find(mod([1:M],mp) == 0))-xhatmc(1,:) - ehatsim1;
    deltahatvector2 = xmc(2,find(mod([1:M],mp) == 0))-xhatmc(2,:) - ehatsim2;
    deltabarvector1 = xmc(1,:) - xbarmc(1,:) - ebarsim1;
    deltabarvector2 = xmc(2,:) - xbarmc(2,:) - ebarsim2;
    ehatsim1 = ehatsim1+ deltahatvector1/trajectory;
    ehatsim2 = ehatsim2+ deltahatvector2/trajectory;
    ebarsim1 = ebarsim1+ deltabarvector1/trajectory;
    ebarsim2 = ebarsim2+ deltabarvector2/trajectory;
```

```
    if(trajectory < 2)
        ;
    else
        shatsim1 = ((trajectory -2)/(trajectory -1))*shatsim1 + deltahatvector1.*deltahatvector
        shatsim2 = ((trajectory -2)/(trajectory -1))*shatsim2 + deltahatvector2.*deltahatvector
        sbarsim1 = ((trajectory -2)/(trajectory -1))*sbarsim1 + deltabarvector1.*deltabarvector
        sbarsim2 = ((trajectory -2)/(trajectory -1))*sbarsim2 + deltabarvector2.*deltabarvector
    end
end
toc

%-----------------------------------------------------------------------------------
%Plot stuff---------------------------------------------------------------------------
%-----------------------------------------------------------------------------------

%Plot velocities;
fig13 = figure;
hold on;
plotrange = [-500,10000,-0.2,1.75];
yticks = [0:0.25:1.5];
ymarks = yticks;
xlabel(xlab,'Fontsize',12);
ylabel('Hastighet','Fontsize',12);
axis(plotrange);
set(gca,'XTick',xticks);
set(gca,'YTick',yticks);
set(gca,'XTickLabel',xmarks,'Fontsize',12);
set(gca,'YTickLabel',ymarks,'Fontsize',12);
line(plotrange(1:2),[0,0],'Linestyle',':','color','k');
line([0,0],plotrange(3:4),'Linestyle',':','color','k');
f5 = plot(x(2,:),'b','linewidth',2);
f1 = plot(xbarsub(2,:),'r');
f2 = plot(xbar(2,:),'k');
hold off;
set(fig13,'Position',[100,478,1000,420]);
L2 = legend([f5,f1,f2],'$x_2$','$\bar{x}_2^*$','$\bar{x}_2^o$',...
    'Orientation','Horizontal');
set(L2,'Interpreter','latex','Fontsize',14);
%title('Hastighet for optimalt og suboptimalt filter','FontSize',14);

%Plot positions;
fig14 = figure;
hold on;
plotrange = [-500,10000,-10,100];
yticks = [0:10:100];
ymarks = yticks;
xlabel(xlab,'Fontsize',12);
ylabel('Posisjon','Fontsize',12);
axis(plotrange);
set(gca,'XTick',xticks);
set(gca,'YTick',yticks);
set(gca,'XTickLabel',xmarks,'Fontsize',12);
set(gca,'YTickLabel',ymarks,'Fontsize',12);
line(plotrange(1:2),[0,0],'Linestyle',':','color','k');
line([0,0],plotrange(3:4),'Linestyle',':','color','k');
f5 = plot(x(1,:),'b','linewidth',2);
f1 = plot(xbarsub(1,:),'r');
f2 = plot(xbar(1,:),'k');
hold off;
set(fig14,'Position',[100,478,1000,420]);
L2 = legend([f5,f1,f2],'$x_1$','$\bar{x}_1^*$','$\bar{x}_1^o$',...
    'Orientation','Horizontal');
```

```
set(L2,'Interpreter','latex','Fontsize',14);
%title('Posision for optimalt og suboptimalt filter','FontSize',14);


%Plot position standard deviations
fig11 = figure;
hold on;
plotrange = [-500,10000,-1,1];
yticks = [-0.8:0.4:0.8];
xticks = [0:1000:M];
xmarks = [0:10:100];
ymarks = yticks;
xlab = 'Tid_[s]';
xlabel(xlab);
ylabel('Posisjonsavvik');
axis(plotrange);
set(gca,'XTick',xticks);
set(gca,'YTick',yticks);
set(gca,'XTickLabel',xmarks);
set(gca,'YTickLabel',ymarks);
line(plotrange(1:2),[0,0],'Linestyle',':','color','k');
line([0,0],plotrange(3:4),'Linestyle',':','color','k');
%line(plotrange(1:2),[min(sqrt(squeeze(Pbara(1,1,:)))),min(sqrt(squeeze(Pbara(1,1,:))))],'color','g');
f1 = plot(sqrt(squeeze(Pbar(1,1,:))),'k');
plot(-sqrt(squeeze(Pbar(1,1,:))),'k');
f2 = plot(sqrt(squeeze(Pbarsub(1,1,:))),'r');
plot(-sqrt(squeeze(Pbarsub(1,1,:))),'r');
f4 = plot([mp:mp:M],sqrt(shatsim1),'g*');
plot([mp:mp:M],-sqrt(shatsim1),'g*');
f3 = plot(sqrt(sbarsim1),'g');
plot(-sqrt(sbarsim1),'g');
f6 = plot(x(1,:) - xbarsub(1,:),'k:')
f5 = plot(sqrt(squeeze(Pbara(1,1,:))),'b');
plot(-sqrt(squeeze(Pbara(1,1,:))),'b');
hold off;
set(fig11,'Position',[100,478,1000,420]);
L2 = legend([f5,f1,f2,f4,f6],'$\bar{s}_1^e$','$\bar{s}_1^o$','$\bar{s}_1^*$','$\bar{s}_1^{100}$','$x_1
    'Orientation','Horizontal');
set(L2,'Interpreter','latex','Fontsize',14);
%title('Standardavvik for posisjon for optimalt og suboptimalt filter','FontSize',14);

%Plot velocity standard deviations
fig12 = figure;
hold on;
plotrange = [-500,10000,-0.05,0.05];
yticks = [-0.04:0.02:0.04];
ymarks = yticks;
xlabel(xlab);
ylabel('Hastighetsavvik');
axis(plotrange);
set(gca,'XTick',xticks);
set(gca,'YTick',yticks);
set(gca,'XTickLabel',xmarks);
set(gca,'YTickLabel',ymarks);
line(plotrange(1:2),[0,0],'Linestyle',':','color','k');
line([0,0],plotrange(3:4),'Linestyle',':','color','k');
f1 = plot(sqrt(squeeze(Pbar(2,2,:))),'k');
plot(-sqrt(squeeze(Pbar(2,2,:))),'k');
f2 = plot(sqrt(squeeze(Pbarsub(2,2,:))),'r');
plot(-sqrt(squeeze(Pbarsub(2,2,:))),'r');
f4 = plot([mp:mp:M],sqrt(shatsim2),'g*');
plot([mp:mp:M],-sqrt(shatsim2),'g*');
```

```
f3 = plot(sqrt(sbarsim2),'g');
plot(-sqrt(sbarsim2),'g');
f6 = plot(x(2,:) - xbarsub(2,:),'k:')
f5 = plot(sqrt(squeeze(Pbara(2,2,:))),'b');
plot(-sqrt(squeeze(Pbara(2,2,:))),'b');
hold off;
set(fig12,'Position',[100,478,1000,420]);
L2 = legend([f5,f1,f2,f4,f6],'$\bar{s}_2^e$','$\bar{s}_2^o$','$\bar{s}_2^*$','$\bar{s}_2^{100
    'Orientation','Horizontal');
set(L2,'Interpreter','latex','Fontsize',14);
%title('Standardavvik for hastighet for optimalt og suboptimalt filter','FontSize',14);
```