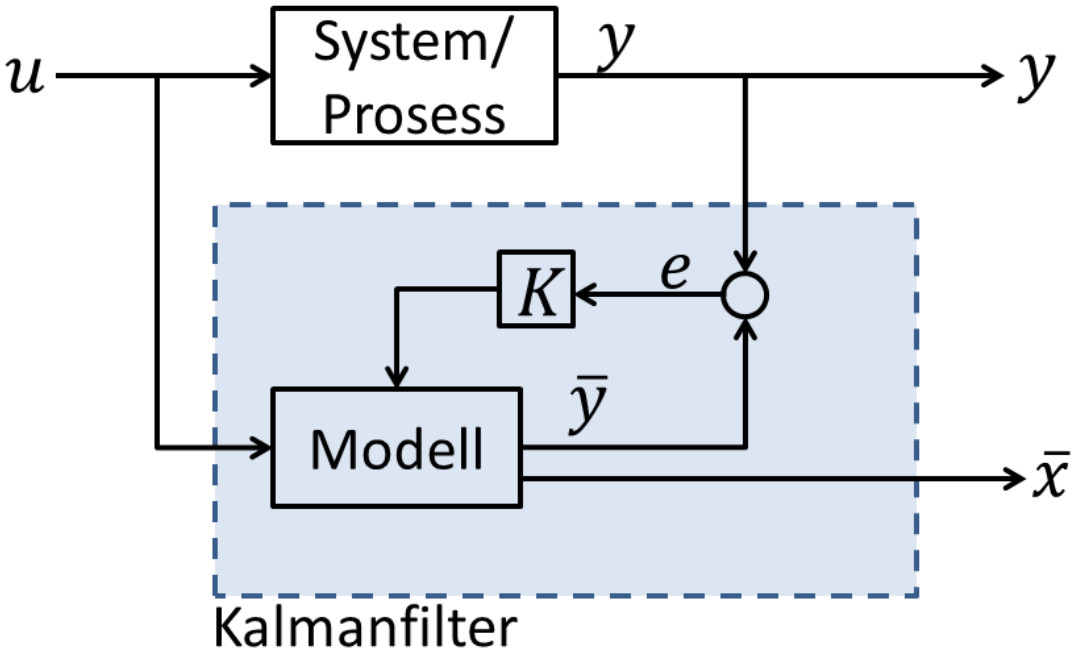
University College of Southeast Norway

Lecture Notes

Kalmanfilter

HANS-PETTER HALVORSEN, 2016.11.01



http://home.hit.no/~hansha

Forord

Dette dokumentet tar for seg grunnleggende modellbasert regulering over temaet Kalmanfilter. Noen forenklinger er gjort underveis da det legges vekt på praktisk forståelse og implementering.

Notatet tar for seg praktisk bruk av Kalmanfilter og hvordan dette kan implementeres i LabVIEW og MathScript. LabVIEW og MathScript er gode engineering-verktøy som er velegnet til formålet da det finnes innebygd støtte for modellering, regulering, datainnsamling, m.m.. Det er også veldig enkelt å komme i gang med LabVIEW og MathScript, samt at disse kan brukes om hverandre eller integreres tett sammen.

I dette dokumentet vil det gis en rask innføring i Kalmanfilter. Målet er å sette leseren i stand til å bruke og implementere et Kalmanfilter i praksis vha en datamaskin. Kalmanfilteret er tross alt ikke noe annet enn en algoritme som må implementeres i en datamaskin. Vi går igjennom konkrete eksempler på hvordan dette gjøres i **LabVIEW**.

Kort fortalt er et Kalmanfilter følgende:

Kalmanfilteret er en tilstandsestimator, dvs. man ønsker å finne systemets tilstandsvariable. Man kan si at Kalmanfilteret er en optimal rekursiv algoritme som implementeres vha et dataprogram.

Utgangspunktet for et Kalmanfilter er en **matematisk modell** av det (stokastiske) systemet en står ovenfor. Jo bedre modellen er, jo bedre resultat får man. En må derimot alltid foreta avveininger mellom kompleksitet og ytelse.

Kalmanfilter – Hva er det? (oppsummert):

- Optimal tilstandsestimator (dvs. man finner optimale verdier på estimatene)
- Bruker en matematisk modell
- Gjelder for stokastiske systemer (Eksempler: bølger, vind, havstrøm, osv.)
- Modellbasert algoritme for implementering i en datamaskin

Fordeler - Fordelene med å benytte en tilstandsestimator er mange, f.eks.:

- **Ikke fysisk målbart -** En tilstandsestimator kan beregne flere av systemets tilstandsvariable enn de som er fysisk målbare. F.eks. Temperaturen i en kjernereaktor.
- **kostbart måleutstyr** - En tilstandsestimator kan også beregne målbare tilstander fremfor å kjøpe inn kostbart måleutstyr.

- **Støy på målinger** - En tilstandsestimator kan gi bedre estimat av målte tilstander da disse ofte er belagt med støy.
- **Feil med måleutstyr** - En tilstandsestimator vil fortsette å gi fornuftige tilstandsestimater selv en tid etter at en eller flere målinger faller ut (prediksjon).

Eksempler:

- **Dynamisk Posisjonering (DP)**: En tilstandsestimator kan f.eks. beregne et skips hastighet selv om en kun måler skipets posisjon. Ved dynamisk posisjonering estimeres vannets strømningshastighet for bruk i en foroverkopling i posisjonsreguleringssystemet for fartøyene. Og de øvrige tilstandene posisjon og hastighet estimeres for bruk i situasjoner der det er manglende (feilaktige eller sjeldne) sensorsignaler fra f.eks. satellittbaserte sensorer (GPS).
- **Atomreaktor**: F.eks. temperaturen i en atomreaktor er fysisk umulig å måle pga. høye temperaturer.
- **Raketter og månelanding**: Det første Kalmanfilteret i praksis var et feilhåndterings-system som ble brukt til å kontrollere kursen til rakettene i Apollo-programmet ifm. månelandingsprosjektet på 60-tallet. Som kjent landet Apollo 11 på månen 21. Juli 1969.
- **Annet**: Rakettstyring, værvarsling, havvarsling, oljeleting, bærekraftig utnytting av ressursene i havet og varsling av influensa-epidemier har minst én ting felles: Det handler om å bruke usikker informasjon (målinger) til å fatte mest mulig sikre beslutninger. I disse anvendelsene er tilstandsestimering (for eksempel Kalmanfilter) svært nyttig.

Kalmanfilteret er utledet fra stokastisk teori og egner ser derfor til bruk i forbindelse med stokastiske (tilfeldige) systemer (ikke-deterministiske). I et deterministisk system er alle eksitasjoner og initialbetingelser kjente i både fortid, nåtid og fremtid, mens de i et stokastisk system bare vil kunne være statistisk beskrevet. Eksempler på stokastiske prosesser kan være vind, havstrøm, bølger, etc.

I dette dokumentet vil vi fokusere på den diskrete versjonen av Kalmanfilteret da vi ønsker å implementere dette i en datamaskin for å få en praktisk forståelse av Kalmanfilter og tilstandsestimering generelt.

Merk! Forskjellige notasjon blir brukt om hverandre i ulik litteratur:

$x(k)$

$x(t_k)$

x_k

→ Alle disse betyr det samme (dvs. verdien av tilstanden x i tidsskrittet k). Du kan bruke de som du synes er mest hensiktsmessig. I dette dokumentet blir de brukt litt om hverandre.

Kalmanfilter-algoritmen er utviklet av Rudolf Kalman i 1960.

Innholdsfortegnelse

Forord ii

Innholdsfortegnelse v

1 Innledning 7

1.1 Eksempel 7

2 Kalmanfilter 8

2.1 Testing 13

2.2 Alternativer - Observers 13

3 Observerbarhet 15

3.1.1 Utledning av Observerbarhetsmatrisen 15

3.1.2 Eksempler 16

4 Kalmanfilter-algoritmen 20

4.1 Utledning 20

4.1.1 Fysisk Prosess/System 20

4.1.2 Estimator 22

4.1.3 Avvik 22

4.1.4 Optimalt estimat 22

4.1.5 Optimal Kalman forsterkning 24

4.2 Stasjonær Kalmanfilter forsterkning 25

4.3 Kalmanfilter algoritmen 25

4.4 Ulike typer algoritmer 28

4.4.1 Prediktor-type 28

4.4.2 Prediktor-korrektor-type 28

5 Tuning av Kalmanfilteret 30

6 Praktisk Implementering 32

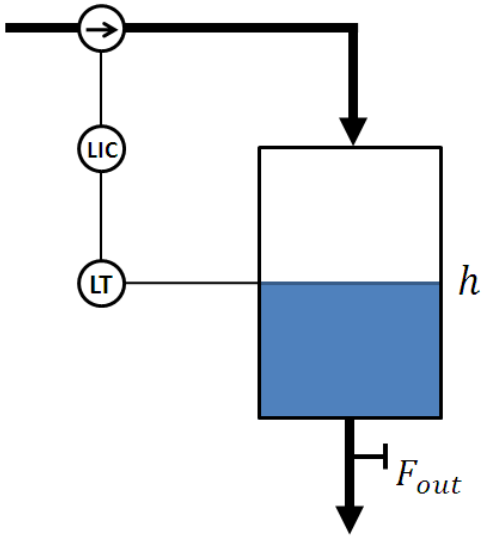
6.1 Observerbarhet 32
6.2 Statisk Kalmanfilter-forsterkning 33
6.3 Kalmanfilter-algoritmer 34
6.4 LabVIEW Eksempel: Diskret Kalmanfilter 35
Appendiks A: Statistikk 38
6.5 Formler 38
6.5.1 Middelverdi/Gjennomsnitt 38
6.5.2 Forventningsverdi 39
6.5.3 Varians 39
6.5.4 Standardavvik 40
6.5.5 Kovarians, Autokovarians 40
6.6 Begreper 42
6.6.1 Stokastisk variabel 42
6.6.2 Normalfordeling (Gauskurven) 42
6.6.3 Hvit støy 43
6.6.4 Farget støy 43
Appendiks B: Matriser 45
6.7 Determinant 45
6.7.1 2x2 Systemer 45
6.7.2 3x3 Systemer 45
Referanser 47

1 Innledning

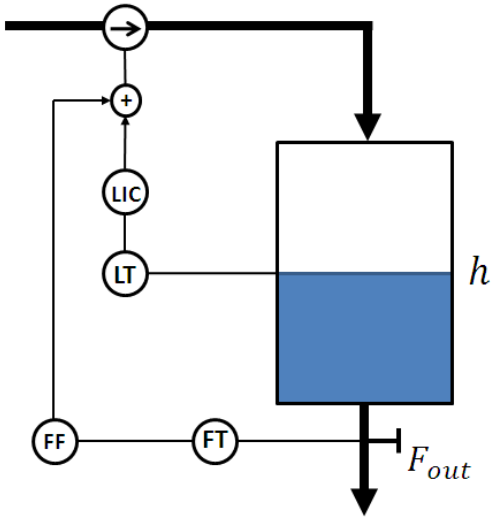
1.1 Eksempel

Vi ønsker å forbedre reguleringen av en vanntank, dvs. regulere nivået h i en vanntank. Vi måler nivået h i tanken og bruker en vanlig PID regulator til å regulere nivået (bildet til venstre). Det renner vann ut fra tanken i bunnen F_{out} som kan betraktes som en forstyrrelse og som gjør det vanskeligere å regulere nivået.

Tilbakekobling:



Foroverkobling:



Matematisk modell for systemet:

$$A\dot{h} = K_p u - F_{out}$$

der A er arealet i tanken, K_p er pumpeforsterkning, u er pådraget, h er nivået i tanken, F_{out} er utstømningen fra tanken.

Vi ønsker å forbedre reguleringen av nivået ved å innføre en foroverkobling fra forstyrrelsen F_{out}, se figur til høyre. Problemet er at vi med dagens prosess ikke måler utstrømningen F_{out}, dermed ønsker vi å estimere denne, slik at dette estimatet kan inngå som en del av reguleringen.

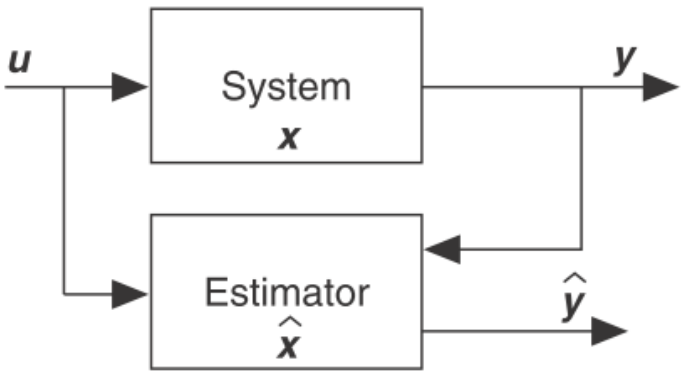
2 Kalmanfilter

Teorien i dette kapitlet er delvis basert på [F. Haugen, Advanced Dynamics and Control: TechTeach, 2010].

Et Kalmanfilter brukes til å estimere tilstandsvariabler i en gitt prosess basert på en matematisk modell av prosessen. Vi ønsker å finne disse tilstandsvariablene fordi de av ulike årsaker ikke måles (for dyrt eller ikke fysisk mulig på grunn av høye temperaturer, osv.).

Kort fortalt er Kalmanfilter en metode (modellbasert algoritme) for å estimere tilstandsvariablene i en prosess som er forutsatt påvirket av stokastiske (tilfeldige) forstyrrelser og av stokastisk (tilfeldig) støy (dvs. hvit normalfordelt støy).

Slik virker et Kalmanfilter grovt sett:

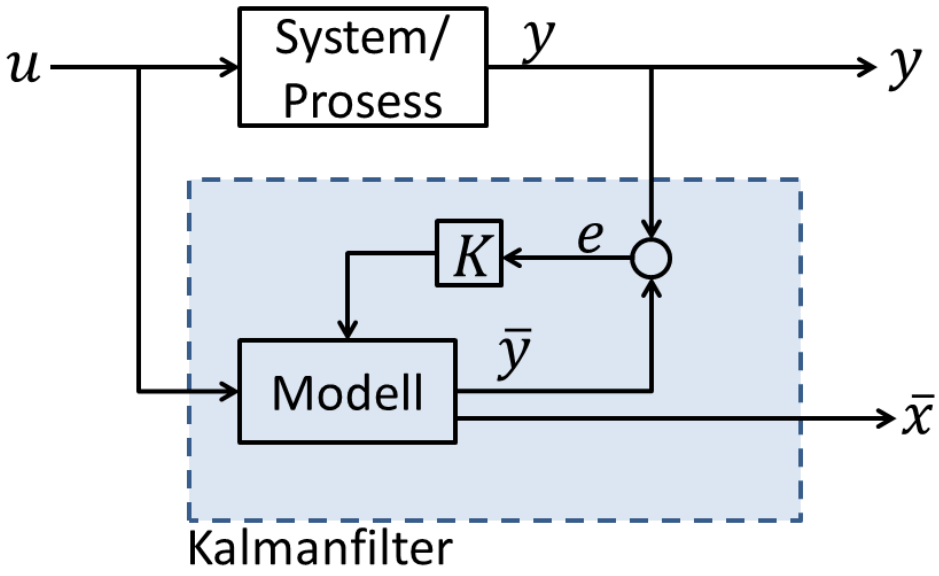


[Figure: F. Haugen, Advanced Dynamics and Control: TechTeach, 2010]

→ En matematisk modell av en gitt prosess kjører i parallell med prosessen. Brukes for å estimere de tilstandene i prosessen som vi ikke måler (av ulike årsaker).

→ Vi bruker $\bar{x}$ og $\hat{x}$ som symboler på at det er et estimat.

Eller litt mer detaljert skisse:



→ Dvs vha en matematisk algoritme (Kalmanfilter algoritmen) finner vi avviket mellom målt verdi og estimatet. Dette avviket brukes til å oppdatere modellen vha. en forsterkningsmatrise K.

Prosess:

Vi tar utgangspunkt i følgende system (virkelig prosess):

$$\boldsymbol{x_{k+1} = Ax_k + Bu_k + w_k}$$

$$\boldsymbol{y_k = Cx_k + v_k}$$

der w er prosess-støy og v er målestøy. Det antas at w og v er hvit og normalfordelt støy som er uavhengige av hverandre, dvs.:

$$p(w) \sim N(0, Q)$$

$$p(v) \sim N(0, R)$$

der Q er prosesstøy-kovariansen og R er målestøy-kovariansen. Disse kan i prinsippet variere med tiden men vi antar at de er konstante.

Dvs. Q sier noe om hvordan prosess-støyen varierer statistisk sett, mens R sier noe om hvordan måle-støyen varierer statistisk sett.

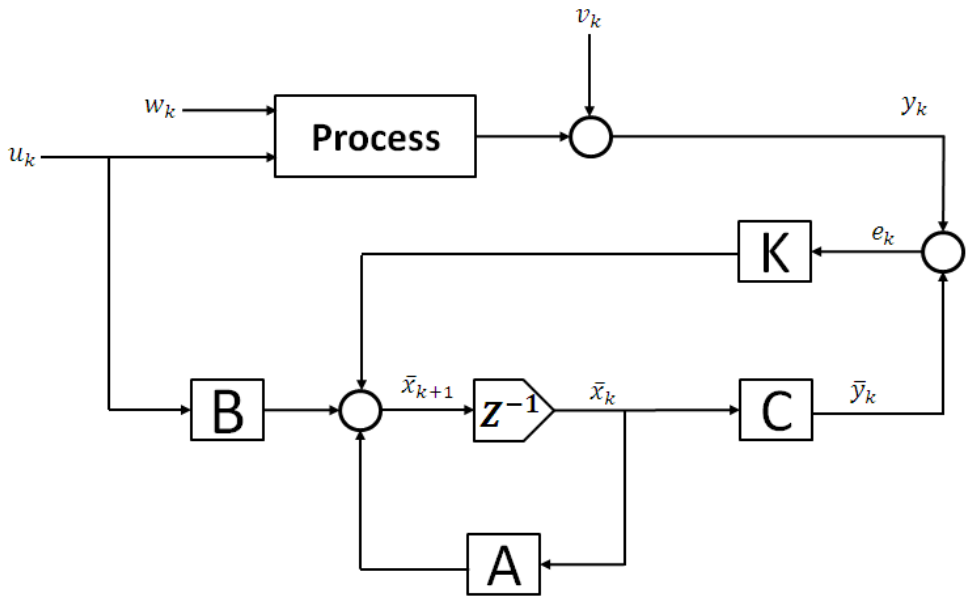
Estimator:

Vi bruker følgende estimator (modell):

$$\boldsymbol{\bar{x}_{k+1} = A\bar{x}_k + Bu_k + K(y_k - \bar{y}_k)}$$

$$\boldsymbol{\bar{y}_k = C\bar{x}_k}$$

Da kan vi tegne Kalmanfilteret på følgende måte (lineært system):



→ Dvs. vi lager en matematisk modell som kjører i parallell med den virkelige prosessen. Modellen oppdateres vha. en forsterkningsmatrise K.

Merk følgende:

- Vi må ha minst en fysisk måling for at Kalmanfilteret skal virke.
- Pådraget u_k påtrykkes både den fysiske prosessen og estimatoren (modellen)
- z er den diskrete versjonen av Laplace-operatoren s (z-transformasjon). Blokka med z^{-1} er rett og slett symbolet for en diskret integrator.

Estimatoravvik:

Vi ønsker at avviket skal være minst mulig.

Vi tar utgangspunkt i estimatoravviket (e_k):

$$e_k = x_k - \bar{x}_k$$

Kalmanfilteret er en optimal tilstandsestimator i den forstand at vi ønsker at variansen til estimator-avviket blir minimalt. Dvs.:

$$\frac{\partial}{\partial K_k}\left(E\left[e_k e_k^T\right]\right) = 0$$

Vi definerer:

$$P_k \equiv E\left[e_k e_k^T\right]$$

Merk P_k er en matrise (kovariansmatrise)

→ Dvs Kalmanfilter-algoritmen går ut på å løse følgende for hvert tidsskritt (setter den deriverte lik null for å finne minimum, dvs. vi ønsker at avviket skal være på et minimum):

$$\frac{\partial P_k}{\partial K_k} = 0$$

Dette gir 2 likninger:

$$K_{k,opt} = f(A, C, R, P_k)$$

der P_k finnes fra følgende likning:

$$P_{k+1} = f(A, C, G, Q, R, P_k)$$

(Vi går ikke inn på detaljer her, da det er litt komplisert å utlede disse likningene)

→ Dvs disse må løses rekursivt for hvert tidsskritt for å finne den optimale K i hvert tidsskritt.

→ Merk! Dette er matriselikninger

→ Der Q og R vil være Kalmanfilterets "tuningsparametre"

$\boldsymbol{Q}$ er autokovarians-matrisen til prosess-støyen w

$$Q = E(ww^T)$$

$$p(w) \sim N(0, Q)$$

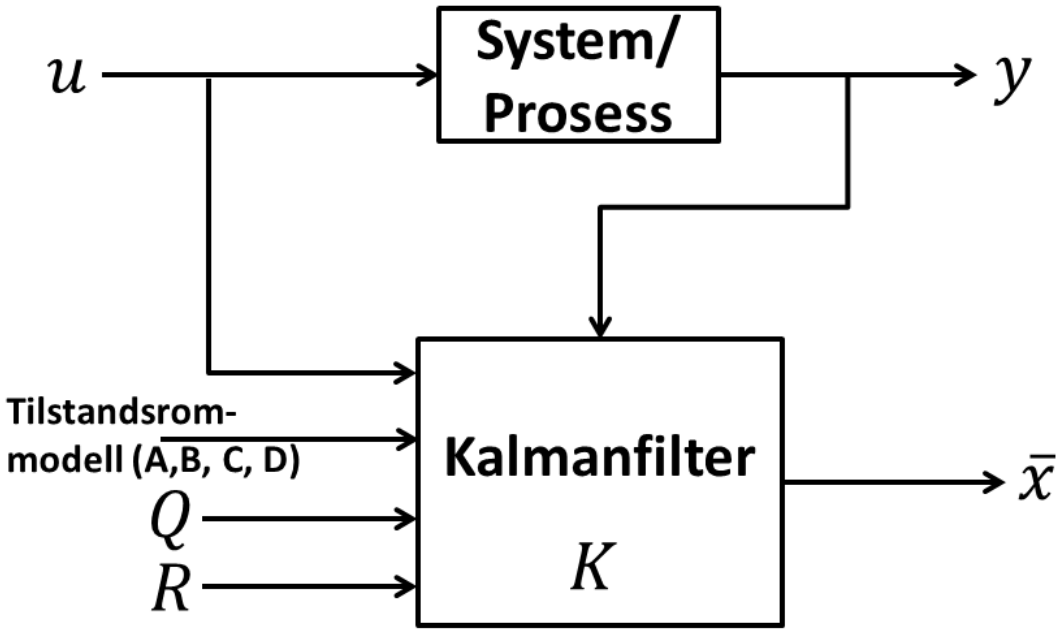
$\boldsymbol{R}$ er autokovarians-matrisen til målestøyen v

$$R = E(vv^T)$$

$$p(v) \sim N(0, R)$$

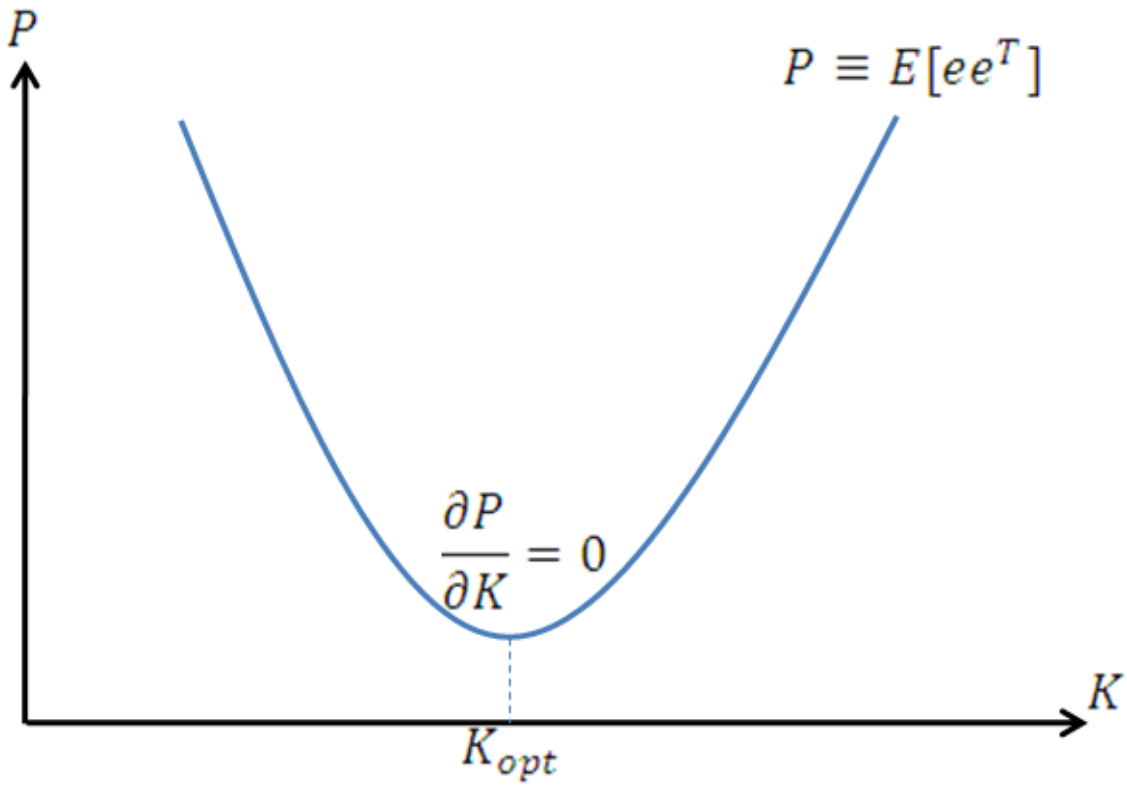
→ dvs. disse gir en statistisk beskrivelse av hvordan støyen i systemet varierer.

Kalmanfilteret kan dermed illustreres slik:



Algoritmen må implementeres i en løkke, hvor man for hvert tidsskritt vil finne en ny optimal verdi av K:

Den optimale løsningen kan illustreres slik:



NB! Dette er en enkel skisse for å illustrere prinsippet, da K og P er matriser (multidimensjonalt) vil det i praksis være mer komplisert å skissere dette.

2.1 Testing

Det lønner seg å teste ut Kalmanfilteret på en simulert prosess før du tar det i bruk på den virkelige prosessen. Du kan bruke f.eks. bruke LabVIEW, MATLAB eller andre simulerings- og programmeringsspråk.

Du bør teste på følgende måte:

1. **Simulering**: Du bør teste ut på en modell med både prosess- og målestøy
2. **Modellfeil**: Du bør innføre noen modellfeil ved å forandre den simulerte prosessen litt – dette for å sjekke om Kalmanfilteret fremdeles gir brukbare estimater selv om modellen ikke er riktig.

2.2 Alternativer - Observers

Et alternativ til Kalmanfilteret er f.eks. ”**Observers**” (Tilstandsestimator med egenverdidesign). ”Observers” har den samme strukturen som Kalmanfilteret. ”Observers” baserer seg på at du kan bestemme hvor raskt estimatene skal konvergere til de virkelige verdiene basert på å bestemme estimatorens egenverdier.

Fordeler/ulemper Kalmanfilter/”Observers”:

- Teori og implementering for ”Observers” er enklere enn for Kalmanfilter.
- Det er ikke rett frem å implementere ”Observers” for systemer som har flere enn en måling, dette er derimot rett frem ifm. Kalmanfilter.
- Observers tar ikke hensyn til støyforholdene.

Oppsummert får kan vi sette opp følgende:

Kalmanfilter	Observer
Mål: Finne optimal K	Mål: Finne K basert på egenverdier
Teorien og matematikken bak Kalmanfilteret er ganske komplisert	Teorien bak Observers er forholdsvis enkel
Rett frem å implementere når man har flere enn en måling	Mer komplisert å implementere når man har flere enn en måling
Tar hensyn til støy	Tar ikke hensyn til støy
Tuningsparametere:	Tuningsparameter:

Q R (Mye prøv og feil for å finne gode verdier)	T_r (Enkelt å forholde seg til)

3 Observerbarhet

Teorien i dette kapitlet er delvis basert på [F. Haugen, Advanced Dynamics and Control: TechTeach, 2010].

En viktig betingelse ifm. Kalmanfilteret er Observerbarhet.

En nødvendig betingelse for at Kalmanfilteret skal virke ordentlig er at systemet er observerbart.

Gitt følgende diskrete og lineære tilstandsrommodell:

$$x_{k+1} = Ax_k + Bu_k$$

$$y_k = Cx_k + Du_k$$

Observerbarhetsmatrisen er definert som:

$$\mathcal{O} = \begin{bmatrix} C \\ CA \\ \vdots \\ CA^{n-1} \end{bmatrix}$$

hvor n er systemets orden, dvs. antall tilstander i tilstandsrommodellen.

→ Et system med orden n er observerbart hvis $\mathcal{O}$ har full rang, dvs. rangen til $\mathcal{O}$ er lik n.

$$rang(\mathcal{O}) = n \rightarrow Observerbart$$

→ Rangen kan sjekkes ved å finne determinanten til $\mathcal{O}$. Hvis determinanten er ulik null, har $\mathcal{O}$ full rang og systemet er dermed observerbart.

$$det(\mathcal{O}) \neq 0 \rightarrow Observerbart$$

3.1.1 Utledning av Observerbarhetsmatrisen

Observerbarhet kan defineres slik:

Systemet:

$$x_{k+1} = Ax_k + Bu_k$$

$$y_k = Cx_k + Du_k$$

er observerbart hvis det finnes en endelig k slik at kjennskap til $u(0), \dots, u(k-1)$ og $(0), \dots, y(k-1)$ er tilstrekkelig til å bestemme systemets initialtilstand $x(0)$.

Utledning:

Vi antar $u(k) = 0$, det gir:

$$x_{k+1} = Ax_k$$

$$y_k = Cx_k$$

Vi får da:

	$y(0) = Cx(0)$
$x(1) = Ax(0)$	$y(1) = Cx(1) = CAx(0)$
$x(2) = Ax(1) = AAx(0) = A^2x(0)$	$y(2) = Cx(2) = CA^2x(0)$
…	…
$x(n-1) = A^{n-1}x(0)$	$y(n-1) = Cx(n-1) = CA^{n-1}x(0)$

Dette gir:

$$\begin{bmatrix} y(0) \\ y(1) \\ \vdots \\ y(n-1) \end{bmatrix} = \underbrace{\begin{bmatrix} C \\ CA \\ \vdots \\ CA^{n-1} \end{bmatrix}}_{\equiv \mathcal{O}} x(0)$$

3.1.2 Eksempler

Eksempel:

Gitt følgende system:

$$x_{k+1} = \underbrace{\begin{bmatrix} 1 & 1 \\ -1 & 2 \end{bmatrix}}_{A} x_k + \underbrace{\begin{bmatrix} 1 \\ 2 \end{bmatrix}}_{B} u_k$$

$$y_k = \underbrace{\begin{bmatrix} 2 & 1 \end{bmatrix}}_{C} x_k + \underbrace{[0]}_{D} u_k$$

Får da ($n = 2$):

$$\mathcal{O} = \begin{bmatrix} C \\ CA \\ \vdots \\ CA^{n-1} \end{bmatrix} = \begin{bmatrix} C \\ CA \end{bmatrix} = \begin{bmatrix} [2 & 1] \\ [2 & 1] \begin{bmatrix} 1 & 1 \\ -1 & 2 \end{bmatrix} \end{bmatrix} = \begin{bmatrix} 2 & 1 \\ (2 \cdot 1 + 1 \cdot (-1)) & (2 \cdot 1 + 1 \cdot 2) \end{bmatrix} = \underline{\underline{\begin{bmatrix} 2 & 1 \\ 1 & 4 \end{bmatrix}}}$$

I **MathScript** kan vi bruke funksjonene **obsv() det()** og **rank()**

MathScript kode:

```
A = [1, 1; -1, 2]
B = [1, 2]'
C = [2, 1]
D = 0
system = ss(A, B, C, D)

Ob = obsvmx(system)
d = det(Ob)
r = rank(Ob)
```

som gir:

```
Ob = 2   1

     1   4

d = 7 (determinant)

r = 2 (rang)
```

[Slutt på eksempel]

Eksempel:

Vi ønsker å estimere tilstandene i en elektrisk likestrømsmotor vha en Observer.

Motoren kan beskrives ved følgende tilstandsrommodell:

$$\begin{bmatrix}\dot{x}_1\\ \dot{x}_2\end{bmatrix} = \begin{bmatrix}0 & 1\\ 0 & -1\end{bmatrix}\begin{bmatrix}x_1\\ x_2\end{bmatrix} + \begin{bmatrix}0\\ 1\end{bmatrix}u$$

$$y = \begin{bmatrix}1 & 0\end{bmatrix}\begin{bmatrix}x_1\\ x_2\end{bmatrix}$$

der x_1 er posisjonen (til akslingen) og x_2 er hastigheten til motoren. Vi måler kun posisjonen og dermed ønsker vi å estimere motorens hastighet.

Vi må derfor finne ut om systemet er observerbart.

Observerbarhetsmatrisen er definert som følger:

$$\mathcal{O} = \begin{bmatrix}C\\ CA\\ \vdots\\ CA^{n-1}\end{bmatrix}$$

der n er antall tilstander.

Observerbarhetsmatrisen blir som følger for dette systemet:

$$\mathcal{O} = \begin{bmatrix} \mathrm{C} \\ \mathrm{CA} \end{bmatrix}$$

Vi får da:

$$C = [1 \quad 0]$$

$$CA = [1 \quad 0]\begin{bmatrix} 0 & 1 \\ 0 & -1 \end{bmatrix} = [0 \quad 1]$$

Dette gir følgende Observerbarhetsmatrise:

$$\mathcal{O} = \begin{bmatrix} \mathrm{C} \\ \mathrm{CA} \end{bmatrix} = \underline{\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}}$$

Et system med orden n er observerbart hvis O har full rang, dvs rangen til O er lik n.

$$rang(\mathcal{O}) = n \rightarrow Observerbart$$

Rangen kan sjekkes ved å finne determinanten til O. Hvis determinanten er ulik null, har O full rang og systemet er dermed observerbart.

$$det(\mathcal{O}) \neq 0 \rightarrow Observerbart$$

Slik finner vi determinanten for et $2x2$ system:

$$A = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix}$$

$$det(A) = |A| = a_{11} \cdot a_{22} - a_{21} \cdot a_{12}$$

Vi finner determinanten til $\mathcal{O}$:

$$\det(\mathcal{O}) = \begin{vmatrix} 1 & 0 \\ 0 & 1 \end{vmatrix} = 1 \cdot 1 - 0 \cdot 0 = 1$$

→ Vi ser at $\det(\mathcal{O}) \neq 0$ → <u>Systemet er observerbart</u>

Anta

$$y = x_2$$

der x_2 er hastigheten til motoren.

Vi ønsker å finne observerbarhetsmatrisen og sjekke om systemet er observerbart nå.

Observerbarhetsmatrisen blir som følger:

$$\mathcal{O} = \begin{bmatrix} \mathrm{C} \\ \mathrm{CA} \end{bmatrix}$$

Vi får da:

$$C = [0 \quad 1]$$

$$CA = [0 \quad 1]\begin{bmatrix} 0 & 1 \\ 0 & -1 \end{bmatrix} = [0 \quad -1]$$

Dette gir følgende observerbarhetsmatrise:

$$\mathcal{O} = \begin{bmatrix} \mathrm{C} \\ \mathrm{CA} \end{bmatrix} = \underline{\begin{bmatrix} 0 & 1 \\ 0 & -1 \end{bmatrix}}$$

Determinanten blir da:

$$\det(\mathcal{O}) = \begin{vmatrix} 0 & 1 \\ 0 & -1 \end{vmatrix} = 0 \cdot (-1) - 0 \cdot 1 = \underline{0}$$

→ Vi ser at $\det(\mathcal{O}) = 0$ → Systemet er ikke observerbart!

Forklaring:

Av definisjonen har vi at et system er observerbart hvis systemet initialtilstander $x(t_0)$ kan bestemmes fra $y(t)$ over et endelig tidsintervall $[t_0, t_1]$.

Altså:

$y = x_2$: I dette tilfellet måler vi altså hastigheten, men dette gir ingen informasjon om posisjonen. Dette viser altså at systemet ikke er observerbart når vi krever informasjon om begge tilstandsvariablene.

$y = x_1$: I dette tilfellet måler vi posisjonen. Hastigheten kan vi enkelt finne ved å ta den deriverte av posisjonen. Systemet er observerbart fordi hastigheten påvirker posisjonen og dermed måleverdien.

[Slutt på eksempel]

4 Kalmanfilter-algoritmen

Teorien i dette kapitlet er delvis basert på [F. Haugen, Advanced Dynamics and Control: TechTeach, 2010].

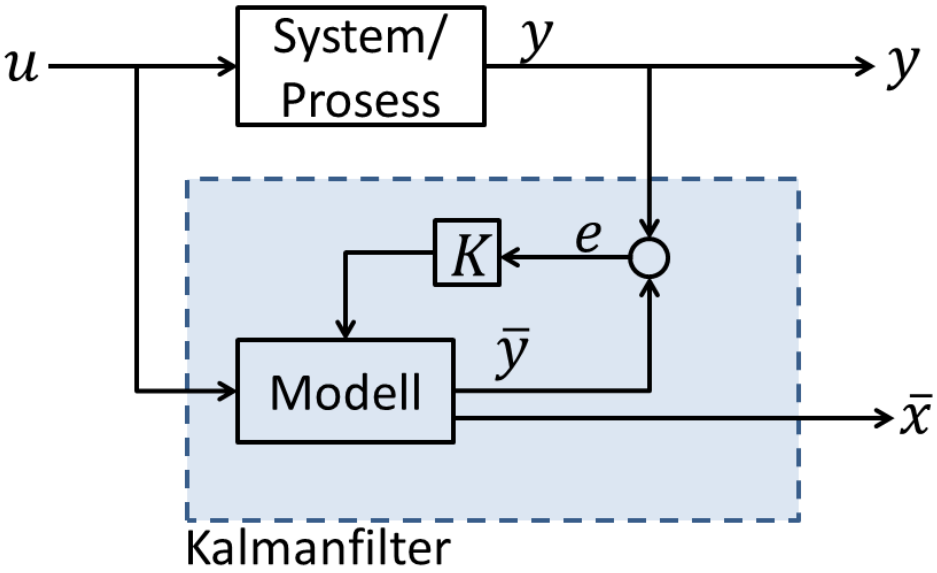
Det finnes flere versjoner av Kalmanfilter-algoritmen. I utledningen under tar vi utgangspunkt i den opprinnelige versjonen.

4.1 Utledning

I utgangspunktet er Kalmanfilteret utledet for lineære systemer, men kan også brukes for ulineære systemer kalles da utvidet Kalmanfilter (Extended Kalmanfilter, EKF).

Det finnes også ulike versjoner av algoritmen, uten at vi skal gå nærmere inn på forskjellene mellom disse.

Vi antar følgende oppstilling:



4.1.1 Fysisk Prosess/System

Anta følgende lineære diskrete system:

$$\boldsymbol{x_{k+1} = Ax_k + Bu_k + Gw_k}$$

$$\boldsymbol{y_k = Cx_k + Du_k + Hv_k}$$

eller på generell form:

$$x_{k+1} = f(x_k, u_k) + Gw_k$$

$$y_k = g(x_k, u_k) + Hv_k$$

der:

x er tilstandsvektoren med n tilstander	$x = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}$
u er pådragsvektoren med m pådrag	$u = \begin{bmatrix} u_1 \\ u_2 \\ \vdots \\ u_m \end{bmatrix}$
y er målevektoren med r målinger	$y = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_r \end{bmatrix}$
w er en vektor med tilfeldig hvit prosess-støy	$w = \begin{bmatrix} w_1 \\ w_2 \\ \vdots \\ w_n \end{bmatrix}$
v er en vektor med tilfeldig hvit målestøy	$v = \begin{bmatrix} v_1 \\ v_2 \\ \vdots \\ v_r \end{bmatrix}$

og:

A er systemmatrisen	$n\ x\ n$
B er pådragsmatrisen	$n\ x\ m$
G er prosesstøymatrise	Vanligvis settes G lik identitetsmatrisen I: $G = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & \ddots & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$
C er målematrisen	$r\ x\ n$
D er pådragsmatrisen som virker direkte på målingen	Vanligvis settes $D = [0]$
H er målestøymatrisen	Vanligvis settes $H = [0]$

4.1.2 Estimator

Vi antar følgende estimator:

$$\bar{x}_{k+1} = A\bar{x}_k + Bu_k + K(y_k - \bar{y}_k)$$

$$\bar{y}_k = C\bar{x}_k + Du_k$$

der K kalles Kalmanfilter forsterkningen, som brukes til å oppdatere estimeringsavviket.

4.1.3 Avvik

Avviket mellom estimatoren og virkelig prosess er gitt ved:

$$e_k = x_k - \bar{x}_k$$

Vi setter utrykket for $\bar{x}_k$ og x_k inn i $e_k = x_k - \bar{x}_k$ som gir:

$$e_{k+1} = x_{k+1} - \bar{x}_{k+1} = Ax_k + Bu_k + Gw_k - [A\bar{x}_k + Bu_k + K(y_k - \bar{y}_k)]$$

videre:

$$\begin{aligned} e_{k+1} &= x_{k+1} - \bar{x}_{k+1} \\ &= Ax_k + Bu_k + Gw_k - [A\bar{x}_k + Bu_k + K(\underbrace{Cx_k + Du_k + Hv_k}_{y_k} \\ &\quad - (\underbrace{C\bar{x}_k + Du_k}_{\bar{y}_k}))] \end{aligned}$$

som gir (NB! Vi ser at alle pådragsleddene forsvinner!):

$$e_{k+1} = Ax_k + Gw_k - A\bar{x}_k - K(Cx_k + Hv_k + C\bar{x}_k)$$

videre:

$$e_{k+1} = Ax_k + Gw_k - A\bar{x}_k - KCx_k - \underbrace{KH}_{\tilde{H}} v_k - KC\bar{x}_k$$

videre:

$$e_{k+1} = A\underbrace{(x_k - \bar{x}_k)}_{e_k} - KC\underbrace{(x_k - \bar{x}_k)}_{e_k} + Gw_k - \tilde{H}v_k$$

som gir:

$$\underline{e_{k+1} = (A - KC)e_k + Gw_k - \tilde{H}v_k}$$

4.1.4 Optimalt estimat

Kalmanfilteret er en tilstandsestimator som beregner det optimale estimatet i den forstand at estimeringsavvikets varians får en minimal verdi.

$$\boldsymbol{P_k \equiv E[e_k e_k^T]}$$

Vi definerer dette som kovariansmatrisen P_k

Ut fra dette finner man en optimal matrise $\boldsymbol{K}$ $(n\ x\ r)$, kalt **Kalmanfilter forsterkningen** som brukes i Kalmanfilter algoritmen til å oppdatere estimatene.

Vi har følgende:

$$P_{k+1} = E[e_{k+1} e_{k+1}^T]$$

Innsatt får vi:

$$P_{k+1} = E\left[\left((A - KC)e_k + Gw_k - \tilde{H}v_k\right)(\cdot)^T\right]$$

Som gir:

$$P_{k+1} = (A - KC)E[e_k e_k^T](A - KC)^T + G\underbrace{E[w_k w_k^T]}_{\equiv Q}G^T + \tilde{H}\underbrace{E[w_k w_k^T]}_{\equiv R}\tilde{H}^T$$

Dette gir:

$$\underline{P_{k+1} = (A - KC)P_k(A - KC)^T + GQG^T + \tilde{H}R\tilde{H}^T}$$

Vi har dermed innført autokovarians matrisene Q og R:

Q er autokovarians matrisen til prosess-støyen w $Q = E(ww^T)$ $E\{w\}$ er forventningsverdien ("middelverdien") til prosess-støy vektoren w.	Vanligvis antar vi at Q er en diagonalmatrise: $\boldsymbol{Q} = \begin{bmatrix} \boldsymbol{q_{11}} & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{q_{22}} & \boldsymbol{0} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{\ddots} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{q_{nn}} \end{bmatrix}$ der q_{ii} er variansen til w_i
R er autokovarians matrisen til målestøyen v $R = E(vv^T)$	Vanligvis antar vi at R er en diagonalmatrise: $\boldsymbol{R} = \begin{bmatrix} \boldsymbol{r_{11}} & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{r_{22}} & \boldsymbol{0} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{\ddots} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{r_{rr}} \end{bmatrix}$

$E\{v\}$ er forventningsverdien ("middelverdien") til målestøy vektoren v.	der r_{ii} er variansen til v_i

→ Q og R brukes som "tuningsparametre" ("vektmatriser") i Kalmanfilteret, nesten på samme måte som vi bruker P, I og D som tuningparametre i en PID regulator.

NB! Hvis vi antar at de enkelte støykomponentene ikke påvirker hverandre blir Q og R diagonalmatriser.

4.1.5 Optimal Kalman forsterkning

Vi ønsker å finne den optimale K, dvs vi deriverer P_{k+1} mtp. K_k som settes lik null:

$$\boxed{\frac{\partial P_{k+1}}{\partial K_k} = 0}$$

Vha. matrisederivasjon, m.m. finner vi at (vil ikke bli utledet her):

$$\underline{K_k = AP_kC^T[CP_kC^T + R]^{-1}}$$

NB! K varier med tiden! – derfor K_k

Hvis vi setter uttrykket for K_k inn i uttrykket for P_{k+1} får vi:

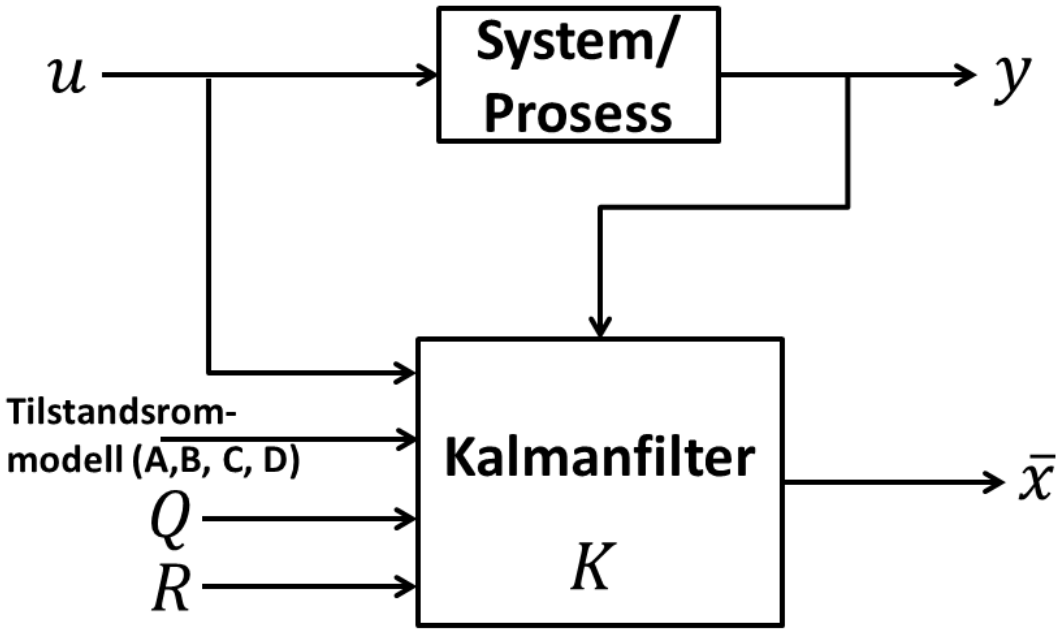
$$P_{k+1} = AP_kA^T + GQG^T + AP_kC^T[CP_kC^T + R]^{-1}CP_kA^T$$

Denne ligningen betegnes som den såkalte diskrete **Riccati-likningen**. Dette er en differenselikning (diskret differensiallikning).

→ Denne må løses for hvert tidsskritt for å kunne den optimale K for hvert tidsskritt.

K vil variere med tiden, men da det er "regnekrevende" å gjøre dette, brukes vanligvis den stasjonere verdien av K, kalt K_s. Samt at K konvergerer ganske raskt mot K_s

Dette kan illustreres med følgende skisse:

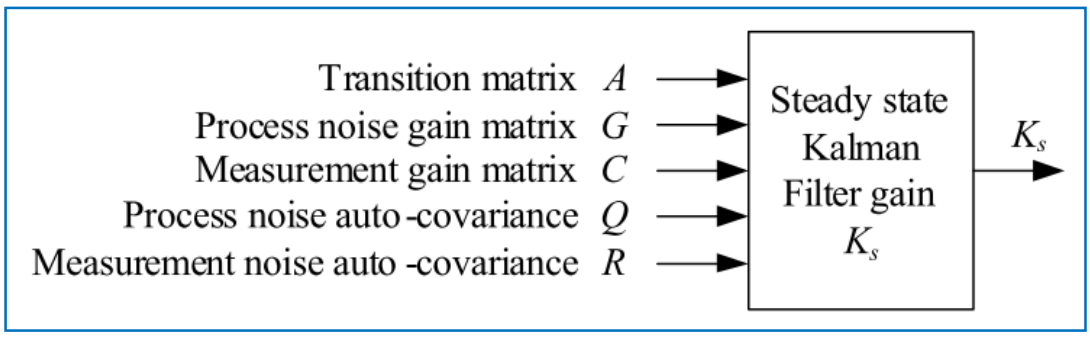


4.2 Stasjonær Kalmanfilter forsterkning

K vil variere med tiden, men da det er "regnekrevende" å gjøre dette, brukes vanligvis den stasjonere verdien av K, kalt K_s. Samt at K konvergerer ganske raskt mot K_s

→ Det stasjonære forholdet finner vi ved $P_{k+1} = P_k = P_\infty$ som gir en stasjonær K.

Følgende informasjon er nødvendig for å finne/beregne den stasjonære Kalmanfilter forsterkningen K_s:



[Figure: F. Haugen, Advanced Dynamics and Control: TechTeach, 2010].

I MathScript/MATLAB og LabVIEW finnes det ferdige funksjoner for dette.

4.3 Kalmanfilter algoritmen

Det eksisterer flere versjoner av Kalmanfilter algoritmen. Her bruker vi en oppdatert versjon (i forhold til den originale algoritmen til Rudolf Kalman fra 1960). Kalmanfilter algoritmen som vi ønsker å implementere i en datamaskin er som følger:

Finn stasjonær Kalman forsterkning K. K varierer med tiden, men vanligvis kan vi bruke den stasjonære versjonen av K.

Finn det innledende tilstandsestimatet

$$\bar{x}_0 = x_0$$

Steg 1: **Finn måleestimatet**

Generelt:

$$\bar{y}_k = g(\bar{x}_k, u_k)$$

For lineære systemer:

$$\boxed{\bar{\boldsymbol{y}}_{\boldsymbol{k}} = \boldsymbol{C}\bar{\boldsymbol{x}}_{\boldsymbol{k}} + \boldsymbol{D}\boldsymbol{u}_{\boldsymbol{k}}}$$

Steg 2: **Finn estimator-avviket**

$$\boxed{\boldsymbol{e}_{\boldsymbol{k}} = \boldsymbol{y}_{\boldsymbol{k}} - \bar{\boldsymbol{y}}_{\boldsymbol{k}}}$$

Steg 3: **Finn det korrigerte tilstandsestimatet**

$$\boxed{\hat{\boldsymbol{x}}_{\boldsymbol{k}} = \bar{\boldsymbol{x}}_{\boldsymbol{k}} + \boldsymbol{K}\boldsymbol{e}_{\boldsymbol{k}}}$$

Steg 4: **Finn det predikerte (fremtidige) tilstandsestimatet**

Generelt:

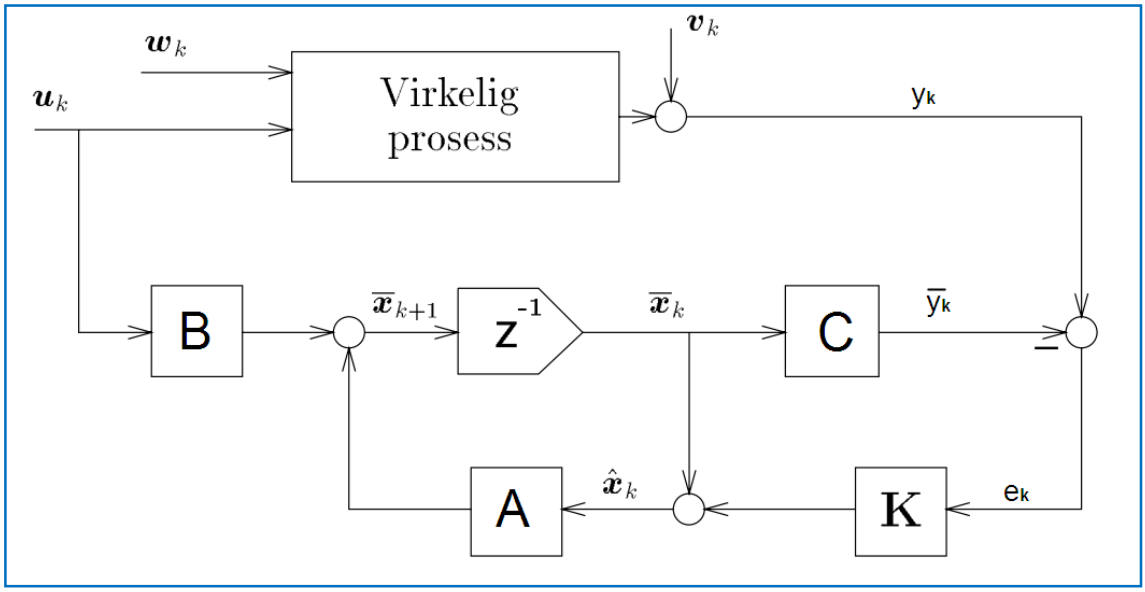
$$\bar{x}_{k+1} = f(\hat{x}_k, u_k)$$

For lineære systemer:

$$\boxed{\bar{\boldsymbol{x}}_{\boldsymbol{k+1}} = \boldsymbol{A}\hat{\boldsymbol{x}}_{\boldsymbol{k}} + \boldsymbol{B}\boldsymbol{u}_{\boldsymbol{k}}}$$

Steg 1- 4 går i loop i et dataprogram

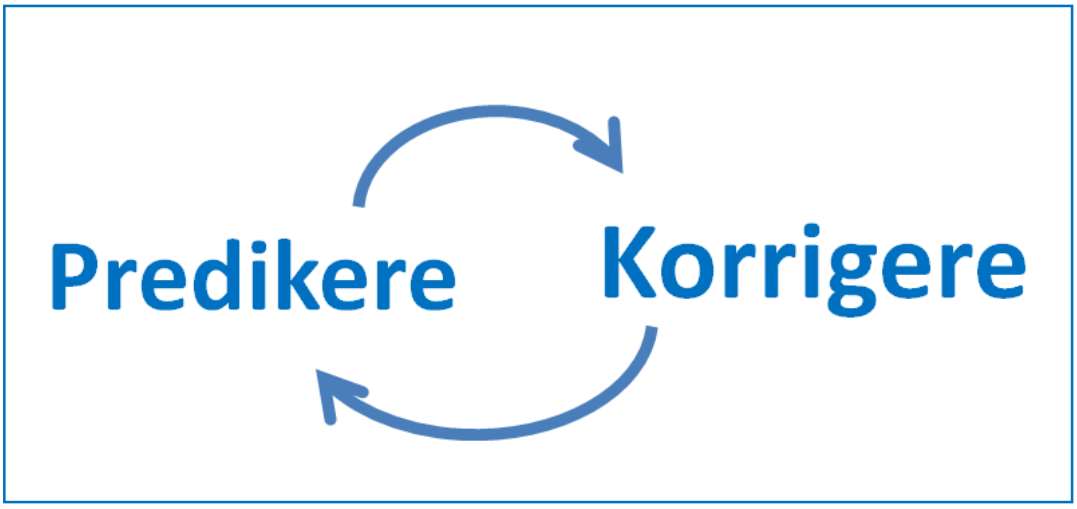
Dette kan skisseres slik i et blokkdiagram (for det lineære tilfellet):



[Figure: Di Ruscio, D. (2010). State estimation and the Kalman filter, Lecture notes]

→ Vi ser at Kalmanfilteret inneholder en modell som går i parallell med den virkelige prosessen. Kalmanfilter modellen skal altså gjengi den virkelige prosessen. Det er derimot umulig å modellere helt nøyaktig, og i tillegg vil prosessen alltid være utsatt for støy. Derfor ser vi at modellen blir oppdatert ved at differansen mellom virkelig og estimert måling multipliseres med Kalmanfilter forsterkningen K.

Hovedprinsippet med Kalmanfilteret kan illustreres slik:



→ Dvs. følgende skjer i en ”evig” løkke i Kalmanfilter algoritmen:

- Prediksjon av ny tilstand
- Oppdatere/Korrigere med siste måling

4.4 Ulike typer algoritmer

Vi opererer med to forskjellige estimater:

$\bar{x}$ **- apriori-estimatet**. Beregnes før nåværende måling er tatt. Kalles også "**Time Update**" estimat eller **Predikert estimat** (x_p).

$\hat{x}$ – **aposteriori-estimatet**. Beregnes etter at nåværende måling er tatt. Kalles også "**Measurement Update**" estimat eller "**Corrected**" estimat (x_c).

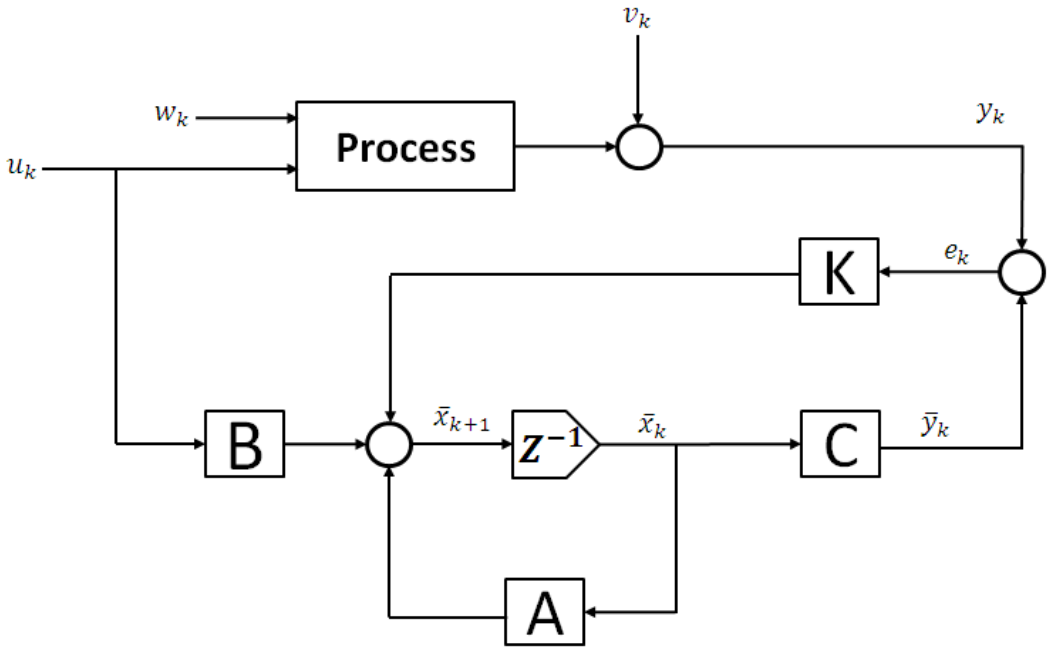
Nedenfor gjennomgår vi disse 2 variantene.

4.4.1 Prediktor-type

Dette er den opprinnelige versjonen. Denne typen skiller ikke mellom apriori-estimatet ($\bar{x}$) og aposteriori-estimatet ($\hat{x}$), dvs det er samme variabel.

Ulempe: Det er en tidsforsinkelse på et tidsskritt mellom $y(k)$ og $\bar{x}(k+1)$.

Blokkdiagram for Prediktor-type Kalmanfilter:

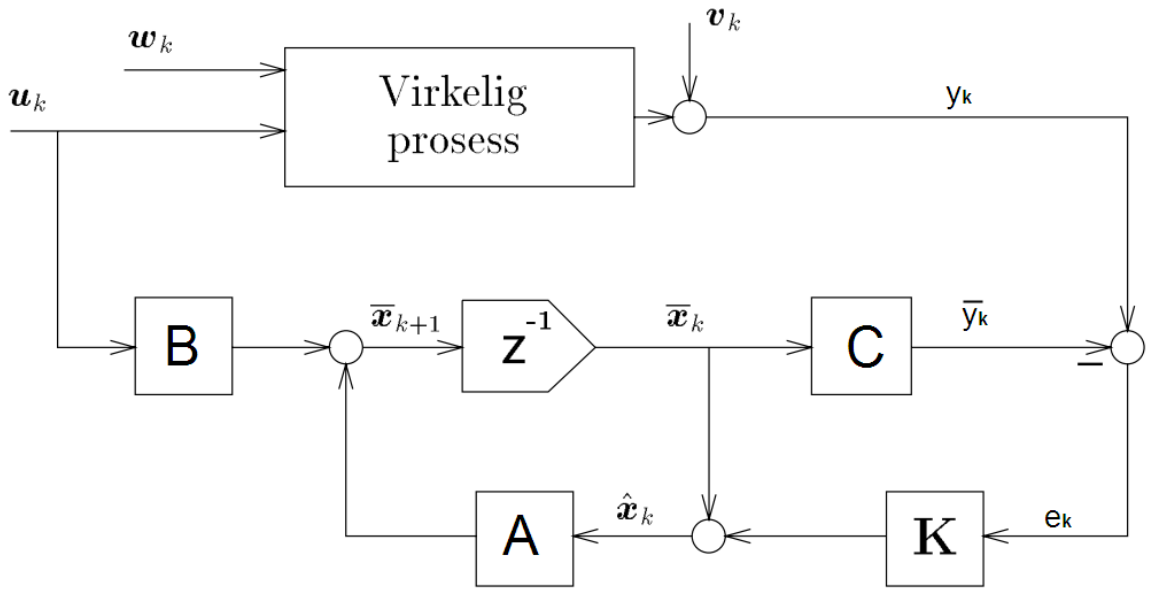


[Figure: Di Ruscio, D. (2010). State estimation and the Kalman filter, Lecture notes]

4.4.2 Prediktor-korrektor-type

Dette er en oppdatert versjon av algoritmen. Denne typen skiller mellom apriori-estimatet ($\bar{x}$) og aposteriori-estimatet ($\hat{x}$), Dette er den mest vanlige i praktiske anvendelser i dag.

Blokkdiagram for Prediktor-korrektor-type Kalmanfilter:



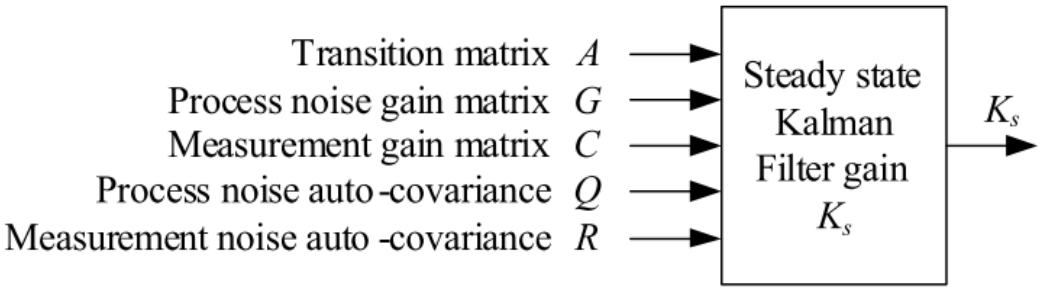
[Figure: Di Ruscio, D. (2010). State estimation and the Kalman filter, Lecture notes]

5 Tuning av Kalmanfilteret

Q og R brukes som "tuningsparametre" ("vektmatriser") i Kalmanfilteret, nesten på samme måte som vi bruker P, I og D som tuningparametre i en PID regulator.

PID regulator	Kalmanfilter
P	Q
I	R
D	

For å finne (stasjonær) K trengs følgende informasjon:



[Figure: F. Haugen, Advanced Dynamics and Control: TechTeach, 2010].

Som vi ser, så avhenger Kalmanfilter forsterkningen av Q og R, dvs vi bruker disse som "tuningsparametre" i Kalmanfilteret.

Q er autokovarians matrisen til prosess-støyen w

R er autokovarians matrisen til målestøyen v

R kan enkelt beregnes fra en tidsserie med målinger ved å bruke en varians funksjon i f.eks LabVIEW eller MathScript. Det er også vanlig for kommersielle måleinstrumenter at denne er oppgitt fra leverandøren sin side.

Jo større Q, jo større Kalmanfilter-forsterkning og sterkere oppdatering av estimatene, men det gjør at estimatene blir mer støyfylte. Så vi må ha en gylden middelvei.

Så hovedregelen er:

Velg Q så stor som mulig uten at estimatene blir for støyfylte.

Som nevnt tidligere kan vi definere Q slik:

$$\boldsymbol{Q} = \begin{bmatrix} \boldsymbol{q_{11}} & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{q_{22}} & \boldsymbol{0} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{\ddots} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{q_{nn}} \end{bmatrix}$$

der verdien q_{11} vil påvirke tilstandsestimatet $\bar{x}_1$, q_{22} vil påvirke tilstandsestimatet $\bar{x}_2$, osv.

6 Praktisk Implementering

Det finnes ferdige funksjoner i LabVIEW og MathScript for å finne blant annet den statiske Kalmanfilter-forsterkningen samt fullstendige Kalmanfilter-algoritmer. Vi vil se litt nærmere på noen av disse funksjonene som finnes.

6.1 Observerbarhet

En nødvendig betingelse for at Observeren skal virke ordentlig er at systemet er observerbart, derfor bør du alltid sjekke om systemet er observerbart først.

Observerbarhetsmatrisen er definert som:

$$O = \begin{bmatrix} C \\ CA \\ \vdots \\ CA^{n-1} \end{bmatrix}$$

hvor n er systemets orden, dvs antall tilstander i tilstandsrommodellen.

→ Rangen kan sjekkes ved å finne determinanten til O. Hvis determinanten er ulik null, har O full rang og systemet er dermed observerbart.

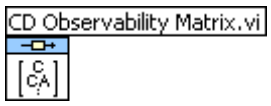
$$\boldsymbol{det(O) \neq 0 \rightarrow Observerbart}$$

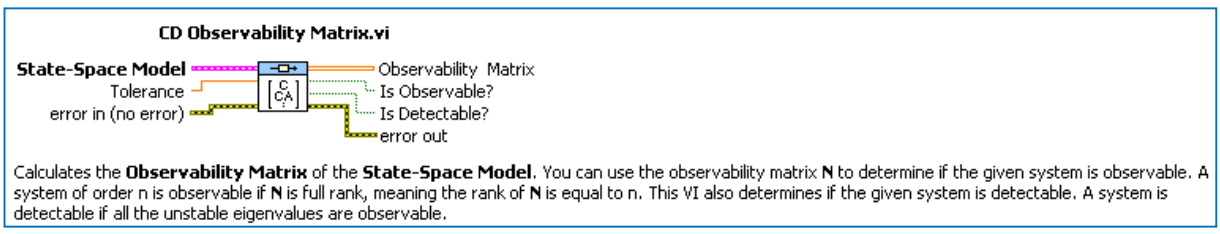
LabVIEW:

I "LabVIEW Control Design and Simulation Module" kan vi bruke "**Observability Matrix.vi**" for å finne observerbarhetsmatrisen og sjekke om systemet er observerbart.

Funksjonen finnes i funksjonspaletten til LabVIEW:

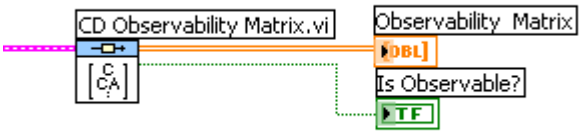
Control Design & Simulation → Control Design → State-Space Model Analysis → CD Observability Matrix.vi





Merk! LabVIEW bruker betegnelsen N på Observerbarhetsmatrisa.

LabVIEW:



MathScript:

I MathScript kan du bruke funksjonen **obsvmx()** for å finne Observerbarhetsmatrisa. Deretter kan du bruke funksjonene **rank()** eller **det()** for å sjekke om systemet er observerbart.

Eksempel:

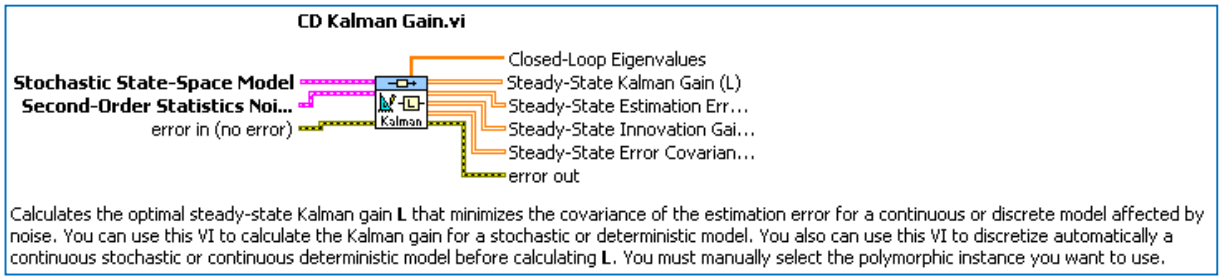
```
% Check for Observability:
O = obsvmx (discretemodel)
r = rank(O)
```

d = det(O)

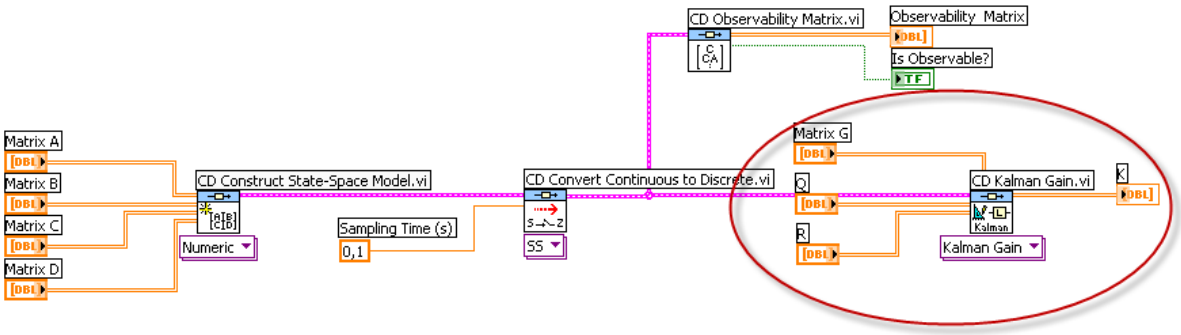
6.2 Statisk Kalmanfilter-forsterkning

Kalman Gain.vi

Bestemmelse av statisk kalmanfilter-forsterkning



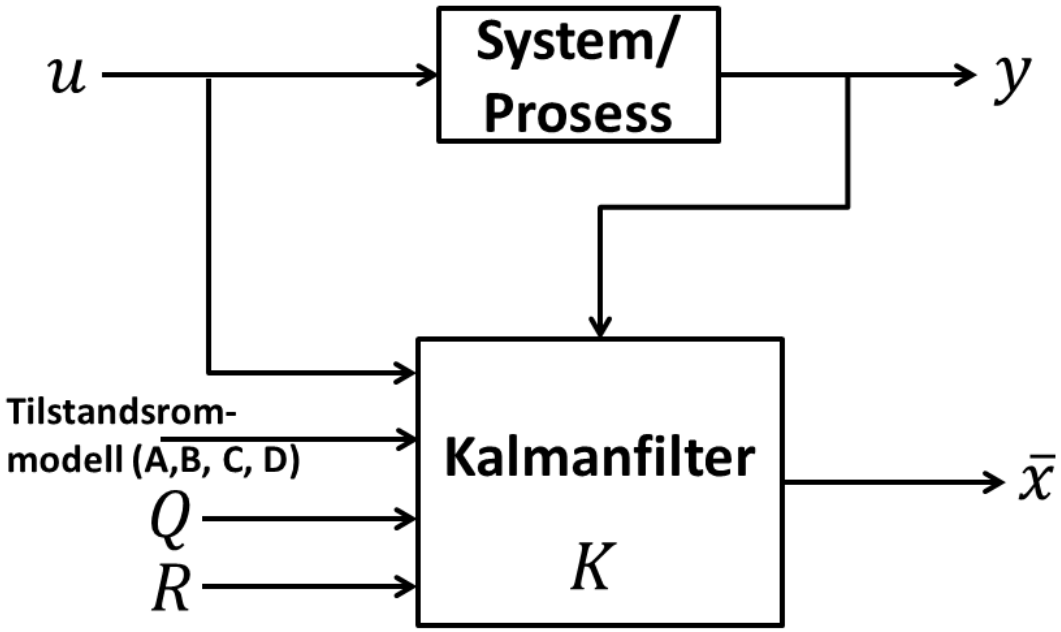
Eksempel:



6.3 Kalmanfilter-algoritmer

LabVIEW tilbyr flere ferdige Kalmanfilter algoritmer.

Felles for disse er at de implementeres som følger:

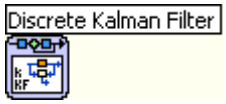


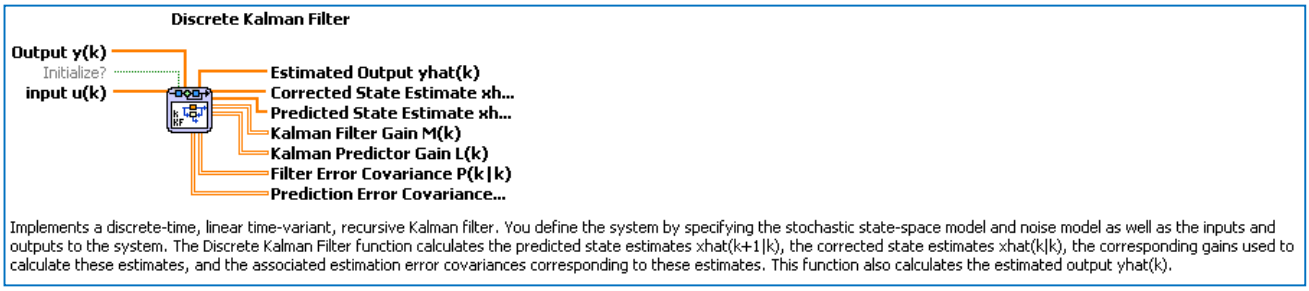
Dvs. Kalmanfilteret kjører i parallell med systemet/prosessen og beregner fortløpende nye estimater basert på oppdaterte måleverdier fra systemet/prosessen.

LabVIEW inneholder både kontinuerlige og diskrete Kalmanfilter-algoritmer som vi kan benytte.

Discrete Kalman Filter:

LabVIEW Functions Palette: Control Design & Simulation → Simulation → Estimation → Discrete Kalman Filter

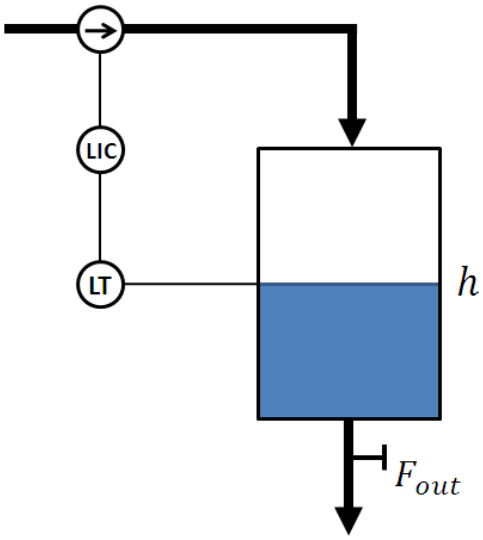




6.4 LabVIEW Eksempel: Diskret Kalmanfilter

Eksempel:

Gitt følgende system:



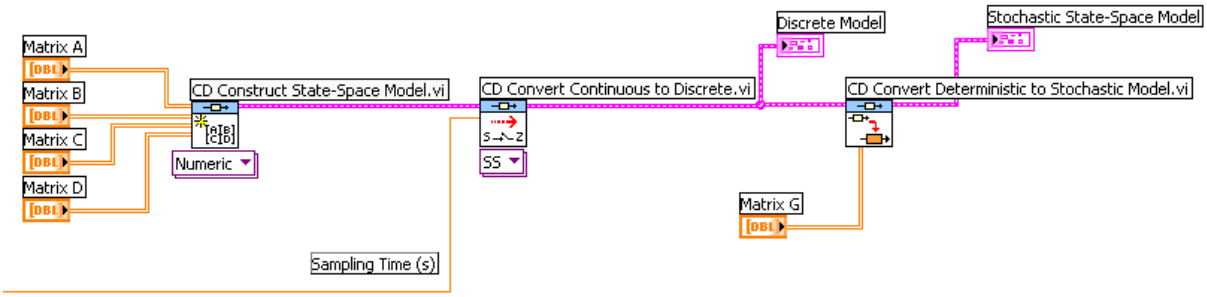
Tilstandsrommodellen av vanntanken er som følger:

$$\begin{bmatrix}\dot{x}_1\\ \dot{x}_2\end{bmatrix} = \underbrace{\begin{bmatrix}0 & -10\\ 0 & 0\end{bmatrix}}_{A}\begin{bmatrix}x_1\\ x_2\end{bmatrix} + \underbrace{\begin{bmatrix}0.02\\ 0\end{bmatrix}}_{B} u$$

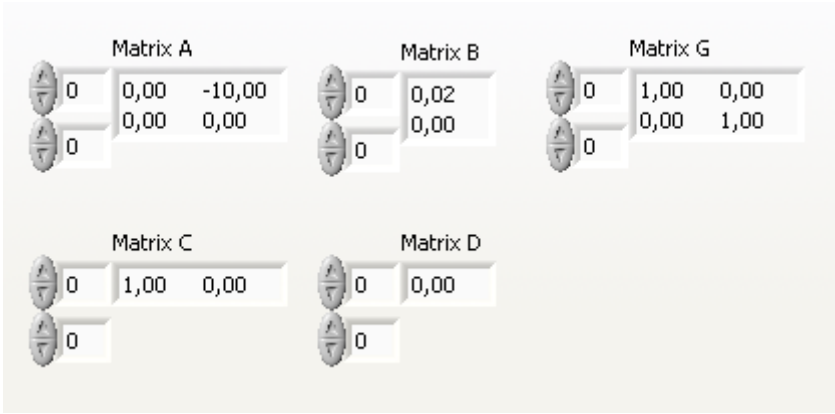
$$y = \underbrace{\begin{bmatrix}1 & 0\end{bmatrix}}_{C}\begin{bmatrix}x_1\\ x_2\end{bmatrix} + \underbrace{[0]}_{D} u$$

x_1 er nivået i tanken (h), mens x_2 er utstrømmingen av tanken (F_{out}). Bare nivået x_1 blir målt.

Steg 1: Først oppretter vi modellen i LabVIEW:

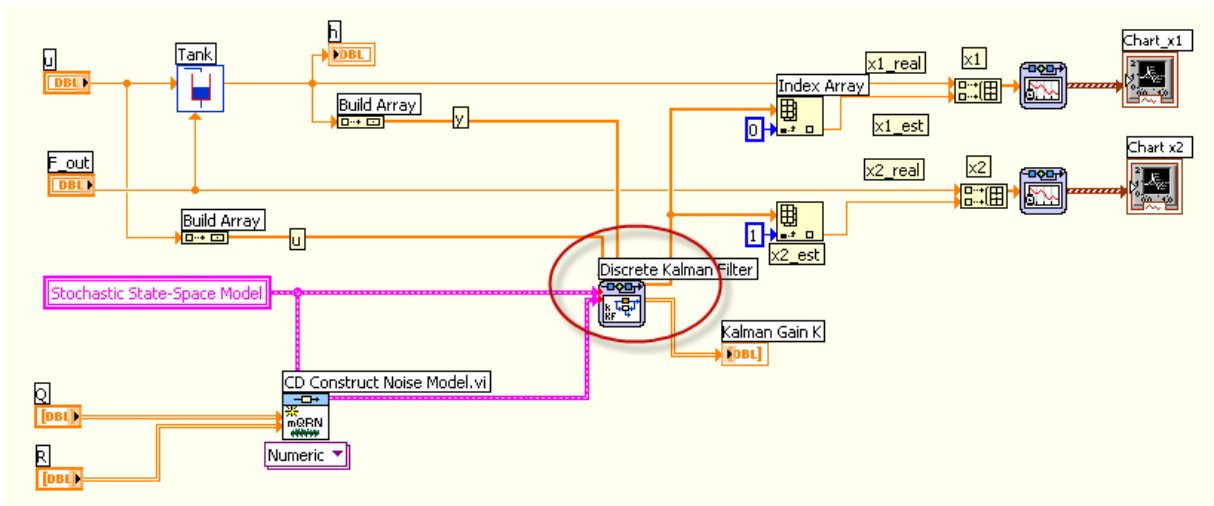


Hvor matrisene A, B, D and D er definert som i tilstandsrommodellen over:



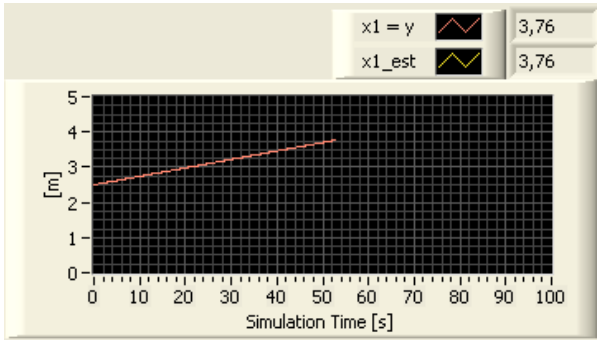
Merk! Den diskrete Kalmanfilter funksjonen i LabVIEW krever en stokastisk tilstandsrommodell (ihht teorien om Kalmanfilteret), så vi må lage en stokastisk modell som vist i LabVIEW koden over.

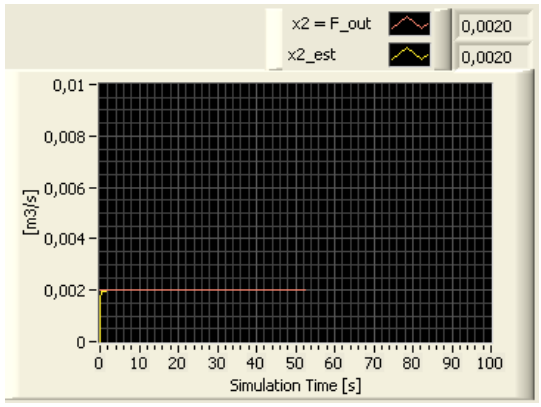
Steg 2: Nå er vi klare til å bruke den innebygde diskrete Kalmanfilter-algoritmen i LabVIEW:



Den diskrete Kalmanfilter-algoritmen i LabVIEW krever også en støymodell, så vi må lage en støymodell fra Q og R matrisene våres.

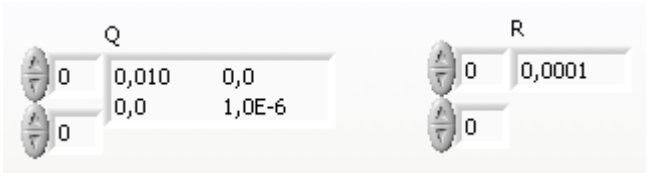
Resultatet blir som følger:





→ Vi ser at resultatet blir ganske bra.

Følgende verdier på Q og R er brukt:



[Slutt på eksempel]

Appendiks A: Statistikk

Siden Kalmanfilteret er basert på stokastisk teori, trenger vi å kjenne til noen begreper innenfor statistikk og stokastisk teori.

Her oppsummerer vi kort de nødvendigste statistiske begrepene som brukes ifm Kalmanfilteret.

6.5 Formler

Vi vil gå gjennom følgende:

- Middelverdi/gjennomsnitt
- Forventningsverdi
- Varians
- Standardavvik
- Kovarians/Autokovarians

6.5.1 Middelverdi/Gjennomsnitt

Middelverdien μ (”mean”/”average”) til en tallrekke/datasett $x_1, x_2, \cdots, x_N$ er gitt som følger:

$$\mu = \bar{x} = \frac{x_1 + x_2 + \cdots + x_N}{N} = \frac{1}{N}\sum_{i=1}^{N} x_i$$

(Det er vanlig å bruke den greske bokstaven ”Mu” μ som symbol på standardavviket)

Eksempel:

Gitt følgende datasett: 2.2, 4.5, 6.2, 3.6, 2.6

I MathScript kan vi bruke den innebygde funksjonen mean():

```
x=[2.2, 4.5, 6.2, 3.6, 2.6]
middelverdi = mean(x)
```

Svaret blir: 3.82

→ Sjekk svaret ved håndregning.

[Slutt på eksempel]

6.5.2 Forventningsverdi

Engelsk: ”Expectation value” (derfor brukes $E(\cdot)$)

For stokastiske variable bruker vi begrepet forventningsverdien istedenfor middelverdien. For en stokastisk variabel X, skriver man $E[X]$ for forventningsverdien til X.

Hvis X er en diskret stokastisk variabel, og antar verdiene $x_1, x_2, \ldots$ med sannsynlighet henholdsvis $p_1, p_2, \ldots$ så er forventningsverdien E(X) gitt ved:

$$E(x) = \sum_{i=1}^{n} p_i x_i$$

der p er sannsynligheten.

Altså: **Forventningen til en stokastisk variabel er en verdi, slik at hvis man gjentar eksperimentet som ligger til grunn for variabelen mange ganger, vil gjennomsnittet av utfallene nærme seg forventningen.** I det diskrete tilfellet er forventningen lik summen av sannsynligheten for hvert utfall, multiplisert med verdien av dette utfallet.

6.5.3 Varians

Varians er et mål på variasjonen i en statistisk fordeling. Vi bruker ofte σ^2 som symbol for variansen. For en stokastisk variabel X er variansen definert som:

$$Var(X) = \sigma^2 = E[(X - \mu)^2]$$

For et gitt antall verdier (datasett) har vi:

$$Var(x) = \frac{1}{N} \sum_{i=1}^{N} (x_i - \bar{x})^2$$

der E(X) er forventningsverdien.

→ Variansen utrykker hvor stort det kvadratiske avviket til observasjonene er i gjennomsnitt.

Eksempel:

Gitt følgende datasett: 2.2, 4.5, 6.2, 3.6, 2.6

I MathScript kan vi bruke den innebygde funksjonen var():

```
x=[2.2, 4.5, 6.2, 3.6, 2.6]
varians = var(x)
```

Svaret blir: 2.572

→ Sjekk svaret ved håndregning.

[Slutt på eksempel]

6.5.4 Standardavvik

Standardavviket er et mål for spredningen av verdiene i et datasett eller av verdien av en stokastisk variabel. Den er definert som kvadratroten av variansen.

Standardavvik σ er gitt som følger:

$$\sigma = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(x_i - \bar{x})^2}$$

(Det er vanlig å bruke den greske bokstaven "Sigma" σ som symbol på standardavviket)

Vi har at:

$$\sigma^2 = Var(X) \Leftrightarrow \sigma = \sqrt{Var(X)}$$

Eksempel:

Gitt følgende datasett: 2.2, 4.5, 6.2, 3.6, 2.6

I MathScript kan vi bruke den innebygde funksjonen std():

```
x=[2.2, 4.5, 6.2, 3.6, 2.6]
standardavvik = std(x)
```

Svaret blir: 1.6037

→ Sjekk svaret ved håndregning.

[Slutt på eksempel]

6.5.5 Kovarians, Autokovarians

Kovarians er et mål på den lineære avhengigheten mellom to varierende størrelser (stokastiske variable).

For to stokastiske variabler X og Y er kovariansen (krysskovariansen) definert som:

$$Cov(X,Y) = E[(X - E[X])(Y - E[Y])]$$

der E er forventningsverdien.

For et gitt antall verdier (datasett) har vi:

$$Cov(X,Y) = \frac{1}{n}\sum_{i=1}^{n}(X - \mu_X)(Y - \mu_Y)$$

der μ er middelverdien.

Kovarians-matrise:

En kovarians-matrise er en matrise hvor elementet (i,j) er kovariansen mellom element i og element j for en vektor med stokastiske variable.

Fra før har vi at for en stokastisk variabel X er variansen definert som:

$$Var(X) = \sigma^2 = E[(X - \mu)^2]$$

der E(X) er forventningsverdien.

Kovariansmatrisen (nxn matrise) for følgende (en vektor med stokastiske variable):

$$X = \begin{bmatrix} X_1 \\ X_2 \\ \vdots \\ X_n \end{bmatrix}$$

blir da:

$$C_X = E[(X - E[X])(X - E[X])^T]$$

Dette blir følgende matrise:

$$\begin{bmatrix} E[(X_1 - \mu_1)(X_1 - \mu_1)] & \cdots & E[(X_1 - \mu_1)(X_n - \mu_n)] \\ \vdots & \ddots & \vdots \\ E[(X_n - \mu_n)(X_1 - \mu_1)] & \cdots & E[(X_n - \mu_n)(X_n - \mu_n)] \end{bmatrix}$$

→ Dvs kovarans-matrisen er en generalisering av varians-begrepet

6.6 Begreper

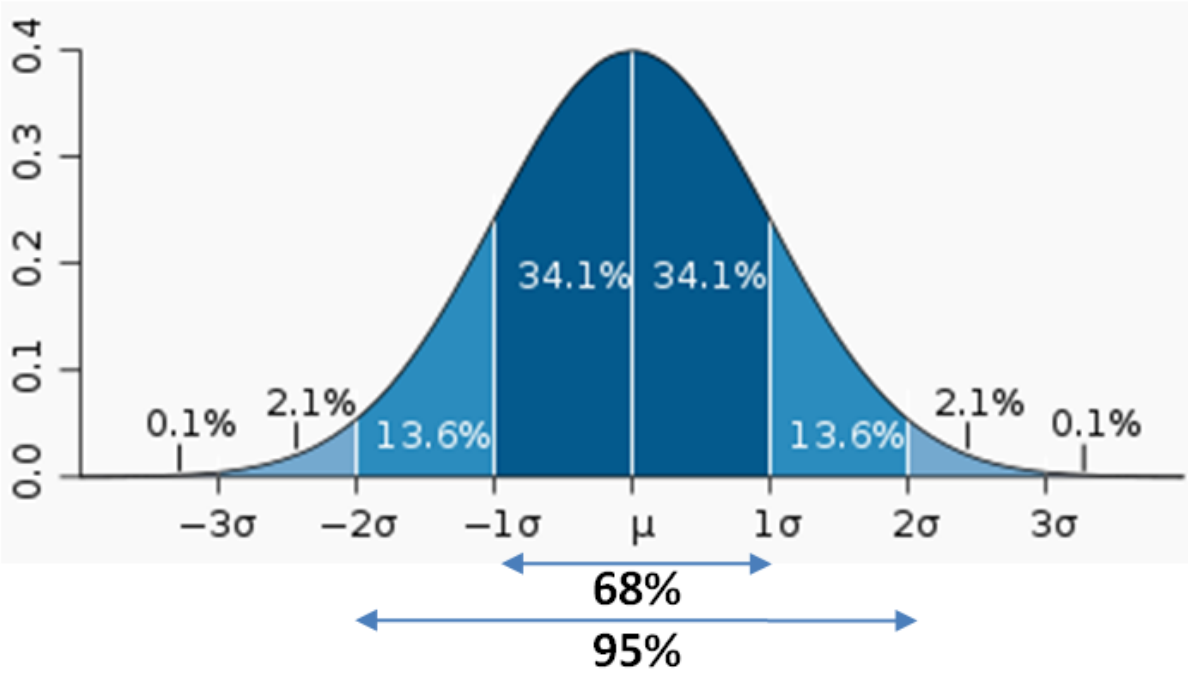
6.6.1 Stokastisk variabel

En stokastisk variabel (vi bruker vanligvis bokstaven X) er et begrep i sannsynlighetsteori og sannsynlighetsregning. Det er en funksjon som tilordner verdier til elementer i utfallsrommet til et tilfeldig eksperiment. For eksempel kan en stokastisk variabel beskrive utfallet av et terningkast, og de mulige utfallene er da elementene i mengden {1,2,3,4,5,6}.

Merk at en stokastisk variabel er en funksjon, og ikke en variabel i den vanlige matematiske betydningen av ordet.

6.6.2 Normalfordeling (Gauskurven)

Normalfordelingen, Gausskurven, er viktig innenfor sannsynlighetsteori og statistikk. En normalfordelt variabel antar ofte verdien som ligger nær middelverdien, og sjelden verdien som har stor avvikelse. Derfor ser normalfordelingen ut som en klokke (bjelle).



[Figure: Wikipedia]

Normalfordelingen er gitt med følgende funksjon:

$$f(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

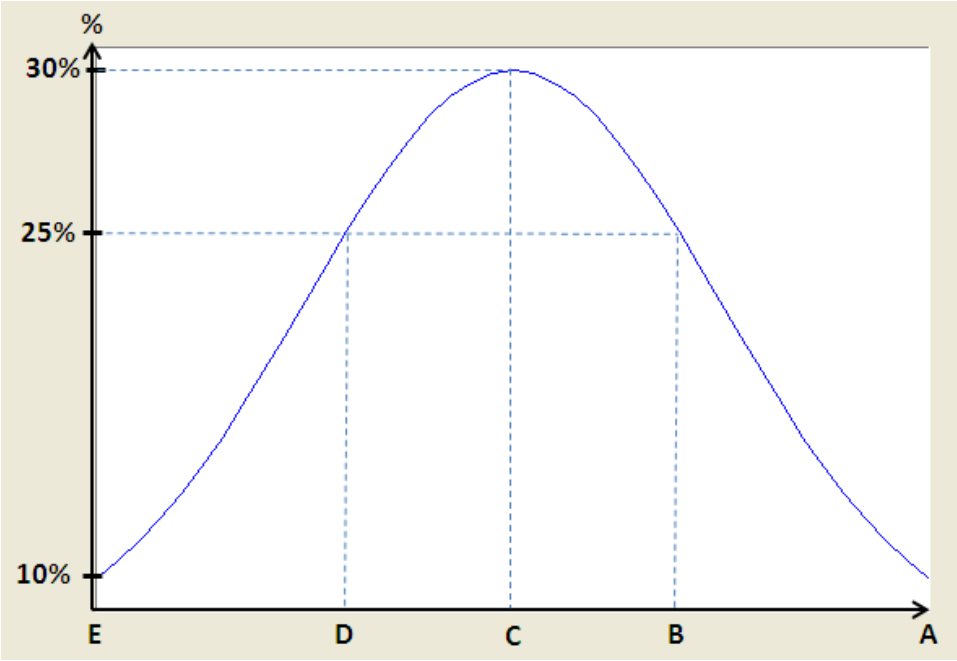
der μ er forventningsverdien, σ er standardavviket.

Denne normalfordelingen kan betegnes:

$$N(\mu, \sigma)$$

Eksempel:

F.eks. på landsbasis skal det tilstrebes en normalfordeling av karakterene over tid, med denne fordelingen: A (10 %), B (25 %), C (30 %), D (25 %), E (10 %). Det betyr ikke at det skal være en normalfordeling innenfor ett emne, innenfor ett semester, eller innenfor ett studiested. Det er snakk om at det skal være en normalfordeling på landsbasis over tid.



6.6.3 Hvit støy

Hvit støy er et stokastisk signal, dvs. støy som varierer helt tilfeldig. I Kalmanfilteret forutsettes det at støyen er hvit.

Man skiller mellom hvit støy, der alle frekvenser er likt representert og støy der fordelingen er annerledes. Støysignalet er kjennetegnet av å være tilfeldig, det vil si at å kjenne signalet fram til et gitt tidspunkt ikke forteller hvordan det vil se ut i neste øyeblikk. Dette i motsetning til f.eks. en sinusbølge. Det eneste som er kjent er sannsynlighetsfordelingen for de forskjellige frekvensenes energitettheter.

For hvit støy er middelverdien lik null, $\mu = 0$

6.6.4 Farget støy

Som en motsetning til hvit støy har vi farget støy som ikke varierer helt tilfeldig. Det finnes flere typer.

Appendiks B: Matriser

6.7 Determinant

Determinanten er bare definert for kvadratiske matriser. Determinanten til en kvadratisk matrise A er definert ved den skalare "tallverdien".

$$det(A) = |A|$$

6.7.1 2x2 Systemer

Slik finner vi determinanten for et 2x2 system:

$$A = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix}$$

$$\boxed{det(A) = |A| = a_{11} \cdot a_{22} - a_{21} \cdot a_{12}}$$

Eksempel:

$$A = \begin{bmatrix} 3 & -2 \\ 5 & 1 \end{bmatrix}$$

$$\det(A) = |A| = 3 \cdot 1 - 5 \cdot (-2) = 3 + 10 = \underline{\underline{13}}$$

I MathScript kan vi gjøre følgende:

```
A=[3 -2;5 1];
det(A)
rank(A)
```

Som gir følgende svar:

```
ans = 13 (determinant)

ans = 2 (rang)
```

[Slutt på eksempel]

6.7.2 3x3 Systemer

Det blir litt mer komplisert for systemer med større orden, men vi vil maks håndregne på 3x3 systemer. For større systemer bruker vi et dataprogram til å beregne dette, f.eks MathScript.

Slik finner vi determinanten for et 3x3 system:

$$A = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix}$$

Vi utvikler determinanten langs en rekke eller en kolonne.

Her utvikler vi determinanten langs første kolonne:

$$\det(A) = a_{11}\begin{vmatrix} a_{22} & a_{23} \\ a_{32} & a_{33} \end{vmatrix} - a_{21}\begin{vmatrix} a_{12} & a_{13} \\ a_{32} & a_{33} \end{vmatrix} + a_{31}\begin{vmatrix} a_{12} & a_{13} \\ a_{22} & a_{23} \end{vmatrix}$$

Vi ser at determinanten til et høyere ordens system kan uttrykkes som en sum av lavere ordens determinanter.

Eksempel:

$$A = \begin{bmatrix} -1 & 3 & 0 \\ 2 & 1 & -5 \\ 1 & 4 & -2 \end{bmatrix}$$

$$\det(A) = (-1)\begin{vmatrix} 1 & -5 \\ 4 & -2 \end{vmatrix} - 2\begin{vmatrix} 3 & 0 \\ 4 & -2 \end{vmatrix} + 1\begin{vmatrix} 3 & 0 \\ 1 & -5 \end{vmatrix}$$

Dette gir:

$$\begin{vmatrix} 1 & -5 \\ 4 & -2 \end{vmatrix} = -2 - (-20) = 18$$

$$\begin{vmatrix} 3 & 0 \\ 4 & -2 \end{vmatrix} = -6 - 0 = -6$$

$$\begin{vmatrix} 3 & 0 \\ 1 & -5 \end{vmatrix} = -15 - 0 = -15$$

Som gir:

$$\det(A) = -18 + 12 - 15 = \underline{\underline{-21}}$$

I MathScript kan vi gjøre følgende:

```
A=[-1 3 0; 2 1 -5; 1 4 -2];
det(A)
rank(A)
```

Som gir følgende svar:

```
ans = -21 (determinant)

ans = 3 (rang)
```

[Slutt på eksempel]

Referanser

Di Ruscio, D. (2003). Matrisemetoder og lineær algebra, Forelesningsnotater.

Di Ruscio, D. (2010). State estimation and the Kalman filter, Lecture notes.

Ergon, R. (1996). Diskret regulering av dynamiske systemer, Del 2: Optimal regulering av stokastiske systemer.

Haugen, F. (1996). Regulering av dynamiske systemer 2, Tapir.

Haugen, F. (2010). Advanced Dynamics and Control, TechTeach.

Skretting, K. (2010). MIK 130 Systemidentifikasjon, Forelesningsnotater

Welch, G and Bishop, G. (2001). An Introduction to the Kalman Filter, Lecture notes.

www.wikipedia.org (2011). Kalman Filter.

Hans-Petter Halvorsen, M.Sc.

E-mail: hans.p.halvorsen@hit.no

Blog: http://home.hit.no/~hansha/

University College of Southeast Norway

www.usn.no
