A Crash Course on Kalman Filtering

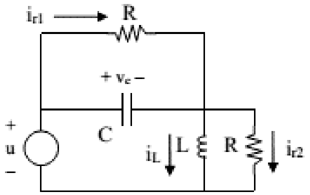
Dan Simon

Cleveland State University

Fall 2014

Outline

- Linear Systems
- Probability
- State Means and Covariances
- Least Squares Estimation
- The Kalman Filter
- Unknown Input Estimation
- The Extended Kalman Filter



KVL: $u = v_c + L\, di_L/dt$ KCL: $i_c + i_{r1} = i_{r2} + i_L$

$i_c = C\, dv_c/dt$

$i_{r1} = v_c/R$

$i_{r2} = (u - v_c)/R$

$\Longrightarrow dv_c/dt = -2v_c/RC + i_L/C + u/RC$

Define $x^T = \begin{bmatrix} v_c & i_L \end{bmatrix}$

$$\dot{x} = \begin{bmatrix} -2/RC & 1/C \\ -1/L & 0 \end{bmatrix} x + \begin{bmatrix} 1/RC \\ 1/L \end{bmatrix} u = Ax + Bu$$

What is the "output"?

Suppose $y = v_{r2} = L\, di_L/dt = \begin{bmatrix} -1 & 0 \end{bmatrix} x + u = Cx + Du$

Let $R = C = L = 1$ and $u(t) =$ step function.
How can we simulate the system in Matlab?

- Control System Toolbox
- Simulink
- m-file

- Control System Toolbox:
  ```
  R = 1; L = 1; C = 1;
  A = [-2/R/C, 1/C ; -1/L, 0];
  B = [1/R/C ; 1/L];
  C = [-1, 0];
  D = 1;
  sys = ss(A, B, C, D);
  step(sys)
  ```
- Simulink: LinearRLC1Model.slx
- m-file: LinearRLC1.m
  What time step should we use?

Linear Systems: Discretization

$$\begin{aligned}
\text{Continuous time: } \dot{x} &= Ax + Bu, \quad y = Cx + Du \\
\text{Discrete time: } x_{k+1} &= Fx + Gu, \quad y = Cx + Du \\
F &= \exp(A\Delta t) \\
G &= (F - I)A^{-1}B
\end{aligned}$$

where Δt is the integration step size; I is the identity matrix

Linear Systems: Discrete-Time Simulation

- Control System Toolbox:

```
dt = 0.1;
F = expm(A*dt);
G = (F - eye(2)) / A * B;
sysDiscrete = ss(F, G, C, D, dt);
step(sysDiscrete)
```

- Simulink: LinearRLC1DiscreteModel.slx
- m-file: LinearRLC1Discrete.m

Outline

- Linear Systems
- Probability
- State Means and Covariances
- Least Squares Estimation
- The Kalman Filter
- Unknown Input Estimation
- The Extended Kalman Filter

Cumulative Distribution Function

$$
\begin{aligned}
X &= \text{random variable} \\
\text{CDF: } F_X(x) &= P(X \leq x) \\
F_X(x) &\in [0, 1] \\
F_X(-\infty) &= 0 \\
F_X(\infty) &= 1 \\
F_X(a) &\leq F_X(b) \quad \text{if } a \leq b \\
P(a < X \leq b) &= F_X(b) - F_X(a)
\end{aligned}
$$

Probability Density Function

$$
\begin{aligned}
\text{PDF: } f_X(x) &= \frac{dF_X(x)}{dx} \\
F_X(x) &= \int_{-\infty}^{x} f_X(z)\, dz \\
f_X(x) &\geq 0 \\
\int_{-\infty}^{\infty} f_X(x)\, dx &= 1 \\
P(a < x \leq b) &= \int_a^b f_X(x)\, dx \\
\text{Expected value: } E[g(X)] &= \int_{-\infty}^{\infty} g(x) f_X(x)\, dx \\
E(X) &= \bar{x} = \mu_x = \text{ mean} \\
E\left[(X - \bar{x})^2\right] &= \sigma_x^2 = \text{ variance} \\
\sigma_x &= \text{standard deviation}
\end{aligned}
$$

Probability

- Random numbers in Matlab: `rand` and `randn`
  - Random number seed
  - How can we create a random vector with given covariance R?
- Probability Density Functions
  - Uniform Distribution
  - Gaussian, or Normal, Distribution

$$\begin{aligned} \text{CDF: } F_{XY}(x,y) &= P(X \le x, Y \le y) \\ \text{PDF: } f_{XY}(x,y) &= \frac{\partial^2 F_{XY}(x,y)}{\partial x \partial y} \end{aligned}$$

Independence:

$$P(X \le x, Y \le y) = P(X \le x)P(Y \le y) \qquad \text{for all } x,\ y$$

$$\begin{aligned} \text{Covariance: } C_{XY} &= E[(X-\bar{X})(Y-\bar{Y})] \\ &= E(XY) - \bar{X}\bar{Y} \\ \text{Correlation: } R_{XY} &= E(XY) \end{aligned}$$

Random Vectors

$$\begin{aligned} X &= \begin{bmatrix} X_1 & X_2 \end{bmatrix} \\ \text{CDF: } F_X(x) &= P(X_1 \leq x_1, X_2 \leq x_2) \\ \text{pdf: } f_X(x) &= \frac{\partial^2 F_X(x)}{\partial x_1 \partial x_2} \end{aligned}$$

- Autocorrelation: $R_X = E[XX^T] > 0$
- Autocovariance: $C_X = E[(X - \bar{X})(X - \bar{X})^T] > 0$
- Gaussian RV:

$$\begin{aligned} \text{PDF}(x) &= \frac{1}{(2\pi)^{n/2}|C_X|^{1/2}} \exp\left[\frac{-1}{2}(x-\bar{x})^T C_X^{-1}(x-\bar{x})\right] \\ \text{If } Y &= AX + b, \text{ then } Y \sim N(A\bar{x} + b, AC_XA^T) \end{aligned}$$

Stochastic Processes

- A stochastic process $X(t)$ is an RV that varies with time
- If $X(t_1)$ and $X(t_2)$ are independent $\forall\, t_1 \neq t_2$ then $X(t)$ is *white*
- Otherwise, $X(t)$ is *colored*

Examples:

- The high temperature on a given day
- The closing price of the stock market
- Measurement noise in a voltmeter
- The amount of sleep you get each night

Outline

- Linear Systems
- Probability
- State Means and Covariances
- Least Squares Estimation
- The Kalman Filter
- Unknown Input Estimation
- The Extended Kalman Filter

State Means and Covariances

$$
\begin{aligned}
x_k &= F_{k-1}x_{k-1} + G_{k-1}u_{k-1} + w_{k-1} \\
w_k &\sim (0, Q_k) \\
\bar{x}_k &= E(x_k) \\
&= F_{k-1}\bar{x}_{k-1} + G_{k-1}u_{k-1} \\
(x_k - \bar{x}_k)(\cdots)^T &= (F_{k-1}x_{k-1} + G_{k-1}u_{k-1} + w_{k-1} - \bar{x}_k)(\cdots)^T \\
&= [F_{k-1}(x_{k-1} - \bar{x}_{k-1}) + w_{k-1}][\cdots]^T \\
&= F_{k-1}(x_{k-1} - \bar{x}_{k-1})(x_{k-1} - \bar{x}_{k-1})^T F_{k-1}^T + \\
&\quad\; w_{k-1}w_{k-1}^T + F_{k-1}(x_{k-1} - \bar{x}_{k-1})w_{k-1}^T + \\
&\quad\; w_{k-1}(x_{k-1} - \bar{x}_{k-1})^T F_{k-1}^T \\
P_k &= E\left[(x_k - \bar{x}_k)(\cdots)^T\right] \\
&= F_{k-1}P_{k-1}F_{k-1}^T + Q_{k-1}
\end{aligned}
$$

This is the discrete-time Lyapunov Equation, or Stein Equation

Outline

- Linear Systems
- Probability
- State Means and Covariances
- Least Squares Estimation
- The Kalman Filter
- Unknown Input Estimation
- The Extended Kalman Filter

Least Squares Estimation

Suppose x is a constant vector

$$\begin{aligned} \text{Vector measurement at time } k\text{: } y_k &= H_k x + v_k, \quad v_k \sim (0, R_k) \\ \text{Estimate: } \hat{x}_k &= \hat{x}_{k-1} + K_k(y_k - H_k \hat{x}_{k-1}) \end{aligned}$$

This is a *recursive* estimator

re·cur·sive: adjective, meaning *recursive*

Our goal: Find the “best” estimator gain K_k

Least Squares Estimation

What is the mean of the estimation error?

$$\begin{aligned} E(\epsilon_{x,k}) &= E(x - \hat{x}_k) \\ &= E[x - \hat{x}_{k-1} - K_k(y_k - H_k\hat{x}_{k-1})] \\ &= E[\epsilon_{x,k-1} - K_k(H_k x + v_k - H_k\hat{x}_{k-1})] \\ &= E[\epsilon_{x,k-1} - K_k H_k(x - \hat{x}_{k-1}) - K_k v_k] \\ &= (I - K_k H_k)E(\epsilon_{x,k-1}) - K_k E(v_k) \end{aligned}$$

$E(\epsilon_{x,k}) = 0$ if $E(v_k) = 0$ and $E(\epsilon_{x,k-1}) = 0$, regardless of K_k
Unbiased estimator

Least Squares Estimation

$$
\begin{aligned}
\text{Objective function: } J_k &= E[(x_1 - \hat{x}_1)^2] + \cdots + E[(x_n - \hat{x}_n)^2] \\
&= E\left(\epsilon_{x1,k}^2 + \cdots + \epsilon_{xn,k}^2\right) \\
&= E\left(\epsilon_{x,k}^T \epsilon_{x,k}\right) \\
&= E\left[\mathrm{Tr}(\epsilon_{x,k} \epsilon_{x,k}^T)\right] \\
&= \mathrm{Tr}\, P_k
\end{aligned}
$$

Least Squares Estimation

$$
\begin{aligned}
P_k &= E(\epsilon_{x,k}\epsilon_{x,k}^T) \\
&= E\left\{[(I - K_k H_k)\epsilon_{x,k-1} - K_k v_k][\cdots]^T\right\} \\
&= (I - K_k H_k)E(\epsilon_{x,k-1}\epsilon_{x,k-1}^T)(I - K_k H_k)^T - \\
&\quad K_k E(v_k \epsilon_{x,k-1}^T)(I - K_k H_k)^T - (I - K_k H_k)E(\epsilon_{x,k-1} v_k^T)K_k^T + \\
&\quad K_k E(v_k v_k^T)K_k^T \\
&= (I - K_k H_k)P_{k-1}(I - K_k H_k)^T + K_k R_k K_k^T
\end{aligned}
$$

Least Squares Estimation

Recall that $\frac{\partial \mathrm{Tr}(ABA^T)}{\partial A} = 2AB$ if B is symmetric

$$
\begin{aligned}
\frac{\partial J_k}{\partial K_k} &= 2(I - K_k H_k)P_{k-1}(-H_k^T) + 2K_k R_k \\
&= 0 \\
K_k R_k &= (I - K_k H_k)P_{k-1}H_k^T \\
K_k(R_k + H_k P_{k-1} H_k^T) &= P_{k-1}H_k^T \\
K_k &= P_{k-1}H_k^T(H_k P_{k-1} H_k^T + R_k)^{-1}
\end{aligned}
$$

Recursive least squares estimation of a constant

1. Initialization: $\hat{x}_0 = E(x), \quad P_0 = E[(x - \hat{x}_0)(x - \hat{x}_0)^T]$
   If no knowledge about x is available before measurements are taken, then $P_0 = \infty I$. If perfect knowledge about x is available before measurements are taken, then $P_0 = 0$.
2. For $k = 1, 2, \cdots$, perform the following.
   1. Obtain measurement y_k:

      $$y_k = H_k x + v_k$$

      where $v_k \sim (0, R_k)$ and $E(v_i v_k) = R_k \delta_{k-i}$ (white noise)
   2. Measurement update of estimate:

      $$\begin{aligned} K_k &= P_{k-1} H_k^T (H_k P_{k-1} H_k^T + R_k)^{-1} \\ \hat{x}_k &= \hat{x}_{k-1} + K_k (y_k - H_k \hat{x}_{k-1}) \\ P_k &= (I - K_k H_k) P_{k-1} (I - K_k H_k)^T + K_k R_k K_k^T \end{aligned}$$

Alternate Estimator Equations

$$
\begin{aligned}
K_k &= P_{k-1} H_k^T (H_k P_{k-1} H_k^T + R_k)^{-1} \\
&= P_k H_k^T R_k^{-1}
\end{aligned}
$$

$$
\begin{aligned}
P_k &= (I - K_k H_k) P_{k-1} (I - K_k H_k)^T + K_k R_k K_k^T \\
&= (P_{k-1}^{-1} + H_k^T R_k^{-1} H_k)^{-1} \\
&= (I - K_k H_k) P_{k-1} \qquad \text{(Valid only for optimal } K_k\text{)}
\end{aligned}
$$

Example: `RLS.m`

Outline

- Linear Systems
- Probability
- State Means and Covariances
- Least Squares Estimation
- The Kalman Filter
- Unknown Input Estimation
- The Extended Kalman Filter

The Kalman filter

$$
\begin{aligned}
x_k &= F_{k-1}x_{k-1} + G_{k-1}u_{k-1} + w_{k-1} \\
y_k &= H_k x_k + v_k \\
w_k &\sim (0, Q_k) \\
v_k &\sim (0, R_k) \\
E[w_k w_j^T] &= Q_k \delta_{k-j} \\
E[v_k v_j^T] &= R_k \delta_{k-j} \\
E[v_k w_j^T] &= 0
\end{aligned}
$$

$$
\begin{aligned}
\hat{x}_k^+ &= E[x_k | y_1, y_2, \cdots, y_k] = \textit{ a posteriori} \text{ estimate} \\
P_k^+ &= E[(x_k - \hat{x}_k^+)(x_k - \hat{x}_k^+)^T] \\
\hat{x}_k^- &= E[x_k | y_1, y_2, \cdots, y_{k-1}] = \textit{ a priori} \text{ estimate} \\
P_k^- &= E[(x_k - \hat{x}_k^-)(x_k - \hat{x}_k^-)^T] \\
\hat{x}_{k|k+N} &= E[x_k | y_1, y_2, \cdots, y_k, \cdots, y_{k+N}] = \text{ smoothed estimate} \\
\hat{x}_{k|k-M} &= E[x_k | y_1, y_2, \cdots, y_{k-M}] = \text{ predicted estimate}
\end{aligned}
$$

Time Update Equations

$$\begin{aligned}
\text{Initialization: } \hat{x}_0^+ &= E(x_0) \\
\hat{x}_1^- &= F_0 \hat{x}_0^+ + G_0 u_0 \\
P_1^- &= F_0 P_0^+ F_0^T + Q_0 \\
\text{Generalize: } \hat{x}_k^- &= F_{k-1} \hat{x}_{k-1}^+ + G_{k-1} u_{k-1} \\
P_k^- &= F_{k-1} P_{k-1}^+ F_{k-1}^T + Q_{k-1}
\end{aligned}$$

These are the Kalman filter *time update* equations

Measurement Update Equations

Recall the RLS estimate of a constant x:

$$\begin{aligned} K_k &= P_{k-1} H_k^T (H_k P_{k-1} H_k^T + R_k)^{-1} \\ \hat{x}_k &= \hat{x}_{k-1} + K_k (y_k - H_k \hat{x}_{k-1}) \\ P_k &= (I - K_k H_k) P_{k-1} (I - K_k H_k)^T + K_k R_k K_k^T \end{aligned}$$

$\hat{x}_{k-1}, P_{k-1}$ = estimate and covariance *before* measurement y_k
$\hat{x}_k, P_k$ = estimate and covariance *after* measurement y_k

Least squares estimator		Kalman filter
$\hat{x}_{k-1}$ = estimate before y_k	$\Longrightarrow$	$\hat{x}_k^-$ = *a priori* estimate
P_{k-1} = covariance before y_k	$\Longrightarrow$	P_k^- = *a priori* covariance
$\hat{x}_k$ = estimate after y_k	$\Longrightarrow$	$\hat{x}_k^+$ = *a posteriori* estimate
P_k = covariance after y_k	$\Longrightarrow$	P_k^+ = *a posteriori* covariance

Measurement Update Equations

Recursive Least Squares:

$$
\begin{aligned}
K_k &= P_{k-1} H_k^T (H_k P_{k-1} H_k^T + R_k)^{-1} \\
\hat{x}_k &= \hat{x}_{k-1} + K_k (y_k - H_k \hat{x}_{k-1}) \\
P_k &= (I - K_k H_k) P_{k-1} (I - K_k H_k)^T + K_k R_k K_k^T
\end{aligned}
$$

Kalman Filter:

$$
\begin{aligned}
K_k &= P_k^- H_k^T (H_k P_k^- H_k^T + R_k)^{-1} \\
\hat{x}_k^+ &= \hat{x}_k^- + K_k (y_k - H_k \hat{x}_k^-) \\
P_k^+ &= (I - K_k H_k) P_k^- (I - K_k H_k)^T + K_k R_k K_k^T
\end{aligned}
$$

These are the Kalman filter *measurement update* equations

Kalman Filter Equations

1. State equations:

$$x_k = F_{k-1}x_{k-1} + G_{k-1}u_{k-1} + w_{k-1}$$
$$y_k = H_k x_k + v_k$$
$$E(w_k w_j^T) = Q_k \delta_{k-j}, E(v_k v_j^T) = R_k \delta_{k-j}, E(w_k v_j^T) = 0$$

2. Initialization: $\hat{x}_0^+ = E(x_0)$, $P_0^+ = E[(x_0 - \hat{x}_0^+)(x_0 - \hat{x}_0^+)^T]$
3. For each time step $k = 1, 2, \cdots$

$$\begin{aligned}
P_k^- &= F_{k-1}P_{k-1}^+ F_{k-1}^T + Q_{k-1} \\
K_k &= P_k^- H_k^T (H_k P_k^- H_k^T + R_k)^{-1} = P_k^+ H_k^T R_k^{-1} \\
\hat{x}_k^- &= F_{k-1}\hat{x}_{k-1}^+ + G_{k-1}u_{k-1} = \text{ \textit{a priori} state estimate} \\
\hat{x}_k^+ &= \hat{x}_k^- + K_k(y_k - H_k \hat{x}_k^-) = \text{ \textit{a posteriori} state estimate} \\
P_k^+ &= (I - K_k H_k)P_k^-(I - K_k H_k)^T + K_k R_k K_k^T \\
&= \left[(P_k^-)^{-1} + H_k^T R_k^{-1} H_k\right]^{-1} \\
&= (I - K_k H_k)P_k^-
\end{aligned}$$

Kalman Filter Properties

Define estimation error $\tilde{x}_k = x_k - \hat{x}_k$
Problem: $\min E\left[\tilde{x}_k^T S_k \tilde{x}_k\right]$, where $S_k > 0$

- If $\{w_k\}$ and $\{v_k\}$ are Gaussian, zero-mean, uncorrelated, and white, then the Kalman filter solves the problem.
- If $\{w_k\}$ and $\{v_k\}$ are zero-mean, uncorrelated, and white, then the Kalman filter is the best *linear* solution to the problem.
- If $\{w_k\}$ and $\{v_k\}$ are correlated or colored, then the Kalman filter can be easily modified to solve the problem.
- For nonlinear systems, the Kalman filter can be modified to approximate the solution to the problem.

Kalman Filter Example: `DiscreteKFEx1.m`

$$
\begin{aligned}
\begin{bmatrix} \dot{r} \\ \dot{v} \\ \dot{a} \end{bmatrix} &= \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} r \\ v \\ a \end{bmatrix} + w \Longrightarrow \dot{x} = Ax + w \\
x_{k+1} &= F x_k + w_k \\
F &= \exp(AT) = \begin{bmatrix} 1 & T & T^2/2 \\ 0 & 1 & T \\ 0 & 0 & 1 \end{bmatrix} \\
w_k &\sim (0, Q_k) \\
\hat{x}_k^- &= F \hat{x}_{k-1}^+ \\
P_k^- &= F P_{k-1}^+ F^T + Q_{k-1} \\
y_k &= H_k x_k + v_k \\
&= \begin{bmatrix} 1 & 0 & 0 \end{bmatrix} x_k + v_k \\
v_k &\sim (0, R_k), R_k = \sigma^2
\end{aligned}
$$

Kalman Filter Divergence

Kalman filter theory is based on several assumptions.
How to improve filter performance in the real world:

- Increase arithmetic precision
- Square root filtering
- Use a fading-memory Kalman filter
- Use fictitious process noise
- Use a more robust filter (e.g., H-infinity)

Modeling Errors

True System:

$$
\begin{aligned}
x_{1,k+1} &= x_{1,k} + x_{2,k} \\
x_{2,k+1} &= x_{2,k} \\
y_k &= x_{1,k} + v_k \\
v_k &\sim (0, 1)
\end{aligned}
$$

Incorrect Model:

$$
\begin{aligned}
x_{1,k+1} &= x_{1,k} \\
y_k &= x_k + v_k \\
w_k &\sim (0, Q), \qquad Q = 0 \\
v_k &\sim (0, 1)
\end{aligned}
$$

Modeling Errors

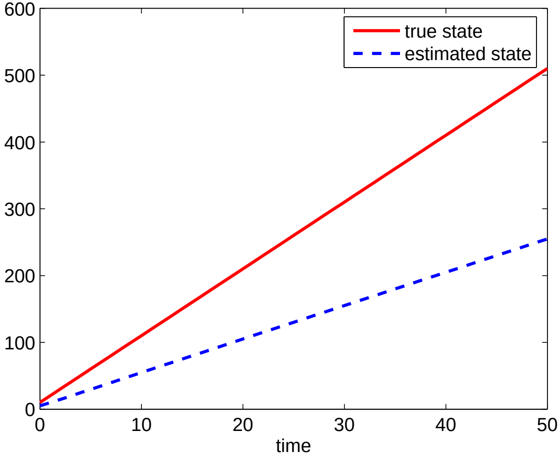


Figure: Kalman filter divergence due to mismodeling

Fictitious Process Noise

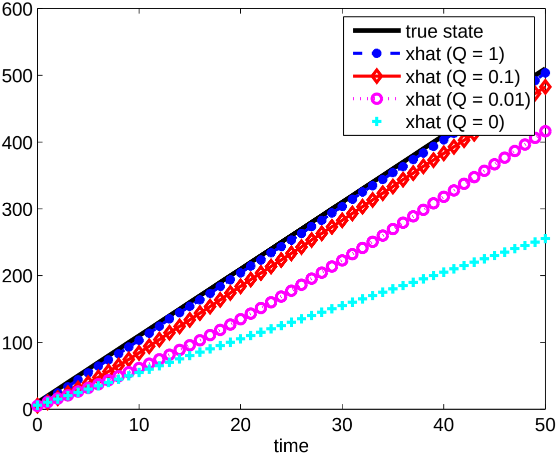


Figure: Kalman filter improvement due to fictitious process noise

Fictitious Process Noise

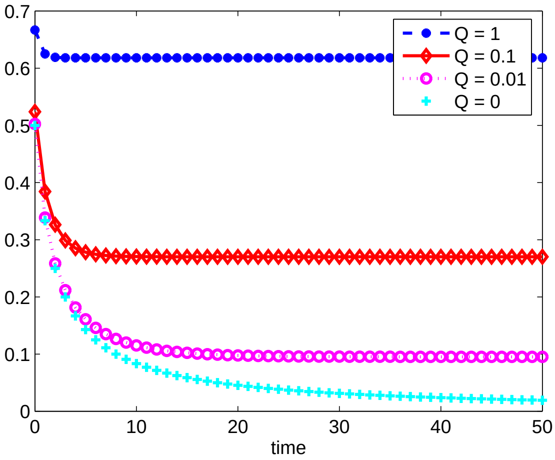


Figure: Kalman gain for various values of process noise

The Continuous-Time Kalman Filter

$$\begin{aligned} \dot{x} &= Ax + Bu + w \\ y &= Cx + v \\ w &\sim (0, Q_c) \\ v &\sim (0, R_c) \end{aligned}$$

Discretize:

$$\begin{aligned} x_k &= Fx_{k-1} + Gu_{k-1} + w_{k-1} \\ y_k &= Hx_k + v_k \\ F &= \exp(AT) \approx (I + AT) \text{ for small } T \\ G &= (\exp(AT) - I)A^{-1}B \approx BT \text{ for small } T \\ H &= C \\ w_k &\sim (0, Q), \qquad Q = Q_c T \\ v_k &\sim (0, R), \qquad R = R_c / T \end{aligned}$$

The Continuous-Time Kalman Filter

Recall the discrete-time Kalman gain:

$$\begin{aligned}
K_k &= P_k^- H^T (H P_k^- H^T + R)^{-1} \\
&= P_k^- C^T (C P_k^- C^T + R_c / T)^{-1} \\
\frac{K_k}{T} &= P_k^- C^T (C P_k^- C^T T + R_c)^{-1} \\
\lim_{T \to 0} \frac{K_k}{T} &= P_k^- C^T R_c^{-1}
\end{aligned}$$

The Continuous-Time Kalman Filter

Recall the discrete-time estimation error covariance equations:

$$
\begin{aligned}
P_k^+ &= (I - K_k H) P_k^- \\
P_{k+1}^- &= F P_k^+ F^T + Q \\
&= (I + AT) P_k^+ (I + AT)^T + Q_c T, \text{ for small } T \\
&= P_k^+ + (A P_k^+ + P_k^+ A^T + Q_c) T + A P_k^+ A^T T^2 \\
&= (I - K_k C) P_k^- + A P_k^+ A^T T^2 + \\
&\quad [A(I - K_k C) P_k^- + (I - K_k C) P_k^- A^T + Q_c] T \\
\frac{P_{k+1}^- - P_k^-}{T} &= \frac{-K_k C P_k^-}{T} + A P_k^+ A^T T + \\
&\quad (A P_k^- + A K_k C P_k^- + P_k^- A^T - K_k C P_k^- A^T + Q_c) \\
\dot{P} &= \lim_{T \to 0} \frac{P_{k+1}^- - P_k^-}{T} \\
&= -P C^T R_c^{-1} C P + A P + P A^T + Q_c
\end{aligned}
$$

The Continuous-Time Kalman Filter

Recall the discrete-time state estimate equations:

$$
\begin{aligned}
\hat{x}_k^- &= F\hat{x}_{k-1}^+ + Gu_{k-1} \\
\hat{x}_k^+ &= \hat{x}_k^- + K_k(y_k - H\hat{x}_k^-) \\
&= F\hat{x}_{k-1}^+ + Gu_{k-1} + K_k(y_k - HF\hat{x}_{k-1}^+ - HGu_{k-1}) \\
&\approx (I + AT)\hat{x}_{k-1}^+ + BTu_{k-1} + \\
&\quad K_k(y_k - C(I + AT)\hat{x}_{k-1}^+ - CBTu_{k-1}), \text{ for small } T \\
&= \hat{x}_{k-1}^+ + AT\hat{x}_{k-1}^+ + BTu_{k-1} + \\
&\quad PC^TR_c^{-1}T(y_k - C\hat{x}_{k-1}^+ - CAT\hat{x}_{k-1}^+ - CBTu_{k-1}) \\
\dot{\hat{x}} &= \lim_{T\to 0} \frac{\hat{x}_k^+ - \hat{x}_{k-1}^+}{T} \\
&= A\hat{x} + Bu + PC^TR_c^{-1}(y - C\hat{x}) \\
\dot{\hat{x}} &= A\hat{x} + Bu + K(y - C\hat{x}) \\
K &= PC^TR_c^{-1}
\end{aligned}
$$

The Continuous-Time Kalman Filter

- Continuous-time system dynamics and measurement:

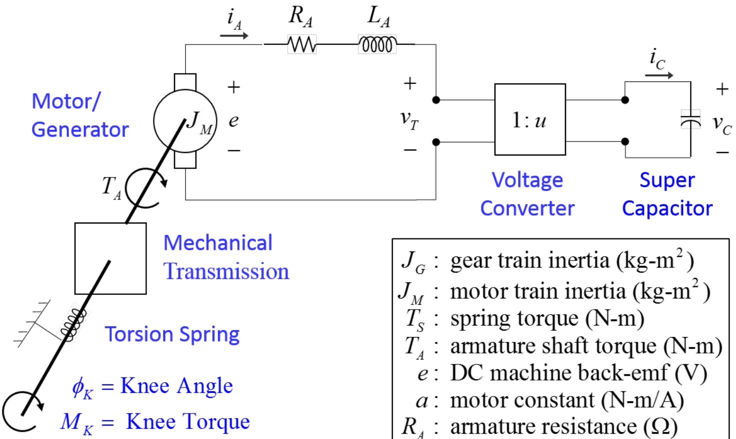
$$\begin{aligned} \dot{x} &= Ax + Bu + w \\ y &= Cx + v \\ w &\sim (0, Q_c) \\ v &\sim (0, R_c) \end{aligned}$$

- Continuous-time Kalman filter equations:

$$\begin{aligned} \hat{x}(0) &= E[x(0)] \\ P(0) &= E[(x(0) - \hat{x}(0))(x(0) - \hat{x}(0))^T] \\ K &= PC^T R_c^{-1} \\ \dot{\hat{x}} &= A\hat{x} + Bu + K(y - C\hat{x}) \\ \dot{P} &= -PC^T R_c^{-1} CP + AP + PA^T + Q_c \end{aligned}$$

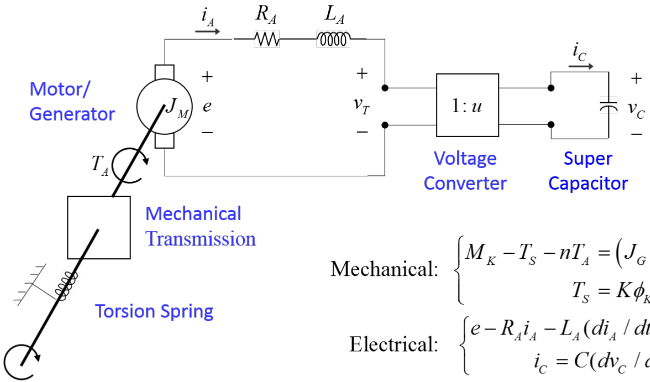
- What if y includes the input also? That is, $y = Cx + Du$?

Example: Regenerative Knee Prosthesis



R. Rarick et al., "Optimal Design of a Transfemoral Prosthesis with Energy Storage and Regeneration," *American Control Conference*, June 2014

- J_G : gear train inertia (kg-m^2)
- J_M : motor train inertia (kg-m^2)
- T_S : spring torque (N-m)
- T_A : armature shaft torque (N-m)
- e : DC machine back-emf (V)
- a : motor constant (N-m/A)
- R_A : armature resistance (Ω)
- C : capacitance (F)
- L_A : armature inductance (H)
- i_A : armature current (A)
- i_C : capacitor current (A)
- u : ideal transformer ratio
- n : transmission ratio



ϕ_K = Knee Angle

M_K = Knee Torque

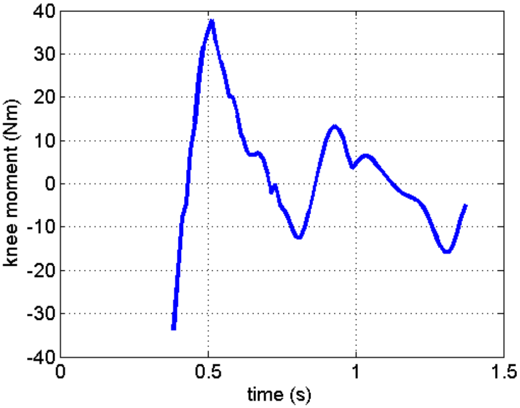
Mechanical: $$\begin{cases} M_K - T_S - nT_A = \left(J_G + n^2 J_M\right)\ddot{\phi}_K \\ T_S = K\phi_K \end{cases}$$

Electrical: $$\begin{cases} e - R_A i_A - L_A(di_A/dt) - v_T = 0 \\ i_C = C(dv_C/dt) \end{cases}$$

DC Transformer: $$\begin{cases} v_T = uv_C \\ i_C = ui_A \end{cases}$$

Electromechanical: $$\begin{cases} e = an\dot{\phi}_K \\ T_A = ai_A \end{cases}$$

Input Torque M



- D. Winter, Biomechanics and Motor Control of Human Movement, 4th Edition, Wiley, 2009, Appendix A
- www.wiley.com/WileyCDA/WileyTitle/productCd-0470398183.html
- `bcs.wiley.com/he-bcs/Books?action=resource&bcsId=5453&itemId=0470398183&resourceId=19492`

State Equations

$$
\begin{aligned}
x_1 &= \phi_k \\
x_2 &= \dot{\phi}_k \\
x_3 &= i_A \\
x_4 &= v_C \\
J_T &= J_G + n^2 J_M \\
\dot{x} &= \begin{bmatrix} 0 & 1 & 0 & 0 \\ -K/J_T & 0 & -na/J_T & 0 \\ 0 & an/L_A & -R_A/L_A & -u/L_A \\ 0 & 0 & u/C & 0 \end{bmatrix} x + \begin{bmatrix} 0 \\ 1/J_T \\ 0 \\ 0 \end{bmatrix} M_K + w \\
y &= \begin{bmatrix} 1 & 0 & 0 & 0 \end{bmatrix} x + v
\end{aligned}
$$

Matlab program: `RegenerationKalman.m`

Outline

Unknown Input Estimation

- Continuous-time system dynamics and measurement:

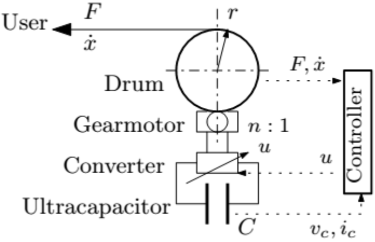
$$\begin{aligned} \dot{x} &= Ax + Bu + f + w \\ y &= Cx + v \\ w &\sim (0, Q_c),\, v \sim (0, R_c) \end{aligned}$$

- Consider f as a state:

$$\begin{aligned} z &= \begin{bmatrix} x^T & f^T \end{bmatrix}^T \\ \dot{z} &= \begin{bmatrix} A & I \\ 0 & 0 \end{bmatrix} z + \begin{bmatrix} B \\ 0 \end{bmatrix} u + \begin{bmatrix} w \\ w' \end{bmatrix} \\ y &= \begin{bmatrix} C & 0 \end{bmatrix} z + \begin{bmatrix} v \\ 0 \end{bmatrix} \\ w &\sim (0, \tilde{Q}), \qquad \tilde{Q} = \mathrm{diag}(Q_c, Q') \\ v &\sim (0, \tilde{R}), \qquad \tilde{R} = \mathrm{diag}(R_c, 0) \end{aligned}$$

- w' is fictitious process noise, and Q' is a tuning parameter

Unknown Input Estimation: Rowing Machine



- θ = position, ω = velocity, q = capacitor charge
- k = spring constant, J = inertia, a = motor constant
- R = resistance, u = power converter ratio, C = capacitance
- r = radius, ϕ = friction

Unknown Input Estimation: Rowing Machine

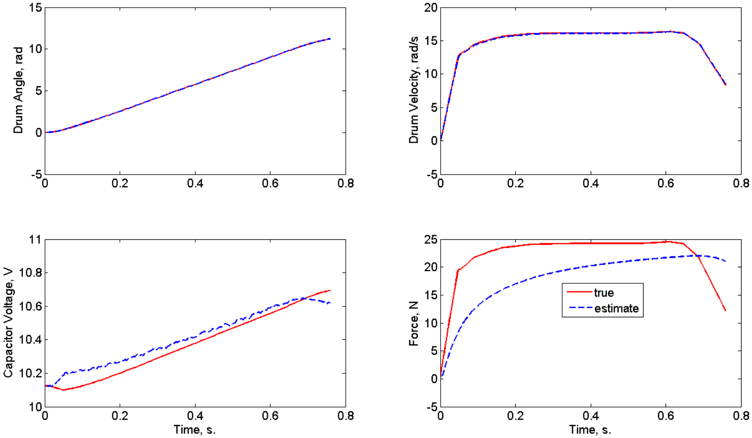
- System model:

$$
\begin{aligned}
\dot{\theta} &= \omega \\
\dot{\omega} &= -\frac{k}{J}\theta - \frac{a^2}{RJ}\omega + \frac{au}{RCJ}q + \frac{r}{J}F - \frac{\phi(\theta,\omega)}{J} \\
\dot{q} &= \frac{au}{R}\omega - \frac{u^2}{RC}q, \quad \phi(\cdot,\cdot) = 0.12\text{sign}(\omega)
\end{aligned}
$$

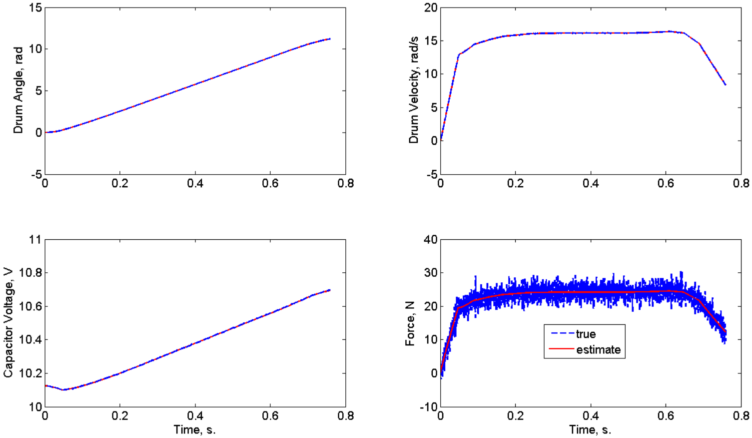
- State space model, assuming $\omega > 0$:

$$
\begin{aligned}
\dot{x} &= \begin{bmatrix} 0 & 1 & 0 & 0 \\ -k/J & -a^2/RJ & au/RCJ & r/J \\ 0 & au/R & u^2/RC & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} x + \begin{bmatrix} 0 \\ -0.12 \\ 0 \\ 0 \end{bmatrix} + w \\
w &\sim (0, Q), \quad Q = \text{diag}\begin{bmatrix} q_1, & q_2, & q_3, & q_4 \end{bmatrix} \\
y &= Cx + v = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} x + v \\
v &\sim (0, R), \quad R = \text{diag}\begin{bmatrix} 0.01^2, & 0.01^2, & (0.01C)^2 \end{bmatrix} (q = CV)
\end{aligned}
$$

Unknown Input Estimation: Rowing Machine

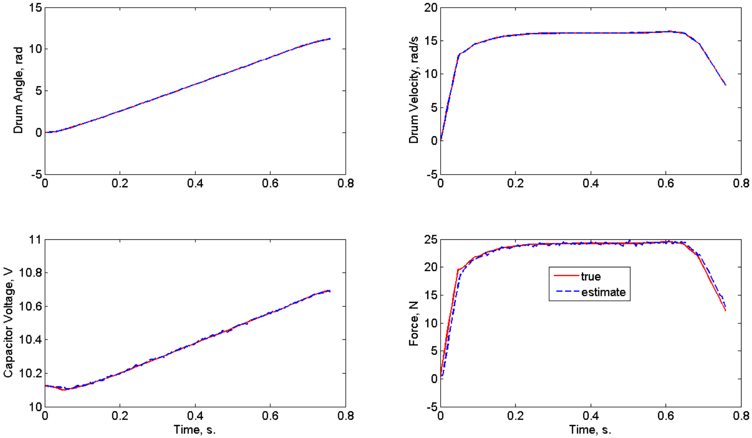


$Q = \text{diag}([0.01^2, 0.01^2, 1^2, 1^2])$ - not responsive enough



$Q = \text{diag}([0.01^2, 0.01^2, 1^2, 10000^2])$ - too responsive

Unknown Input Estimation: Rowing Machine



$Q = \text{diag}([0.01^2, 0.01^2, 1^2, 100^2])$ - just about right

Unknown Input Estimation: Rowing Machine

- How can we improve our results?
- We have modeled F as a noisy constant: $\dot{F} = w$
- Instead we can model F as a ramp:

$$\begin{aligned} \dot{F} &= F_v + w_1 \\ \dot{F}_v &= w_2 \end{aligned}$$

- This increases the number of states by 1 but gives the Kalman filter more flexibility to estimate a value for F that matches the measurements
- RMS force estimation error decreases from 0.8 N to 0.4 N

Outline

Nonlinear Kalman Filtering

- Nonlinear system:

$$\begin{aligned}\dot{x} &= f(x, u, w, t) \\ y &= h(x, v, t) \\ w &\sim (0, Q) \\ v &\sim (0, R)\end{aligned}$$

- Linearization:

$$\begin{aligned}\dot{x} &\approx f(x_0, u_0, w_0, t) + \left.\frac{\partial f}{\partial x}\right|_0 (x - x_0) + \left.\frac{\partial f}{\partial u}\right|_0 (u - u_0) + \\ &\quad \left.\frac{\partial f}{\partial w}\right|_0 (w - w_0) \\ &= f(x_0, u_0, w_0, t) + A\Delta x + B\Delta u + L\Delta w \\ y &\approx h(x_0, v_0, t) + \left.\frac{\partial h}{\partial x}\right|_0 (x - x_0) + \left.\frac{\partial h}{\partial v}\right|_0 (v - v_0) \\ &= h(x_0, v_0, t) + C\Delta x + M\Delta v\end{aligned}$$

Nonlinear Kalman Filtering

$$
\begin{aligned}
\dot{x}_0 &= f(x_0, u_0, w_0, t) \\
y_0 &= h(x_0, v_0, t) \\
\Delta\dot{x} &= \dot{x} - \dot{x}_0 \\
\Delta y &= y - y_0 \\
\Delta\dot{x} &= A\Delta x + Lw \\
&= A\Delta x + \tilde{w} \\
\tilde{w} &\sim (0, \tilde{Q}), \qquad \tilde{Q} = LQL^T \\
\Delta y &= C\Delta x + Mv \\
&= C\Delta x + \tilde{v} \\
\tilde{v} &\sim (0, \tilde{R}), \qquad \tilde{R} = MRM^T
\end{aligned}
$$

We have a linear system with state Δx and measurement Δy

The Linearized Kalman Filter

- System equations:

$$\begin{aligned} \dot{x} &= f(x, u, w, t), \qquad w \sim (0, Q) \\ y &= h(x, v, t), \qquad v \sim (0, R) \end{aligned}$$

- Nominal trajectory:

$$\dot{x}_0 = f(x_0, u_0, 0, t), y_0 = h(x_0, 0, t)$$

- Compute partial derivative matrices:

$$A = \partial f / \partial x|_0 \, , L = \partial f / \partial w|_0 \, , C = \partial h / \partial x|_0 \, , M = \partial h / \partial v|_0$$

- Compute $\tilde{Q} = LQL^T$, $\tilde{R} = MRM^T$, $\Delta y = y - y_0$
- Kalman filter equations:

$$\begin{aligned} \Delta\hat{x}(0) &= 0, P(0) = E\left[(\Delta x(0) - \Delta\hat{x}(0))(\Delta x(0) - \Delta\hat{x}(0))^T\right] \\ \Delta\dot{\hat{x}} &= A\Delta\hat{x} + K(\Delta y - C\Delta\hat{x}), K = PC^T\tilde{R}^{-1} \\ \dot{P} &= AP + PA^T + \tilde{Q} - PC^T\tilde{R}^{-1}CP \\ \hat{x} &= x_0 + \Delta\hat{x} \end{aligned}$$

The Extended Kalman Filter

- Combine the $\dot{x}_0$ and $\Delta\dot{\hat{x}}$ equations:

$$\dot{x}_0 + \Delta\dot{\hat{x}} = f(x_0, u_0, w_0, t) + A\Delta\hat{x} + K[y - y_0 - C(\hat{x} - x_0)]$$

- Choose $x_0(t) = \hat{x}(t)$, so $\Delta\hat{x}(t) = 0$ and $\Delta\dot{\hat{x}}(t) = 0$
- Then the nominal measurement becomes

$$\begin{aligned} y_0 &= h(x_0, v_0, t) \\ &= h(\hat{x}, v_0, t) \end{aligned}$$

and the first equation above becomes

$$\dot{\hat{x}} = f(\hat{x}, u, w_0, t) + K\left[y - h(\hat{x}, v_0, t)\right]$$

The Extended Kalman Filter

- System equations:

$$\begin{aligned} \dot{x} &= f(x,u,w,t), \qquad w \sim (0,Q) \\ y &= h(x,v,t), \qquad v \sim (0,R) \end{aligned}$$

- Compute partial derivative matrices:

$$A = \partial f/\partial x|_{\hat{x}}\,, L = \partial f/\partial w|_{\hat{x}}\,, C = \partial h/\partial x|_{\hat{x}}\,, M = \partial h/\partial v|_{\hat{x}}$$

- Compute $\tilde{Q} = LQL^T$, $\tilde{R} = MRM^T$
- Kalman filter equations:

$$\begin{aligned} \hat{x}(0) &= E[x(0)], \quad P(0) = E\left[(x(0)-\hat{x}(0))(x(0)-\hat{x}(0))^T\right] \\ \dot{\hat{x}} &= f(\hat{x},u,0,t) + K\left[y - h(\hat{x},0,t)\right], \quad K = PC^T\tilde{R}^{-1} \\ \dot{P} &= AP + PA^T + \tilde{Q} - PC^T\tilde{R}^{-1}CP \end{aligned}$$

Robot State Estimation

- Robot dynamics:

$$u = M\ddot{q} + C\dot{q} + g + R + F$$

- u = control forces/torques, q = joint coordinates
- $M(q)$ = mass matrix, $C(q, \dot{q})$ = Coriolis matrix
- $g(q)$ = gravity vector, $R(\dot{q})$ = friction vector
- $F(q)$ = external forces/torques
- State space model:

$$\begin{aligned}
\ddot{q} &= M^{-1}(u - C\dot{q} - g - R - F) \\
x &= \begin{bmatrix} q_1 & q_2 & q_3 & \dot{q}_1 & \dot{q}_2 & \dot{q}_3 \end{bmatrix}^T = \begin{bmatrix} x_1^T & x_2^T \end{bmatrix}^T \\
\dot{x} &= \begin{bmatrix} x_2 \\ M^{-1}(u - C\dot{q} - g - R - F) \end{bmatrix} = f(x, u, w, t) \\
y &= q_3 = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 & 0 \end{bmatrix} x + v = h(x, v, t)
\end{aligned}$$

- The detailed model is available at:

  www.sciencedirect.com/science/article/pii/S0307904X14003096

  dynamicsystems.asmedigitalcollection.asme.org/article.aspx?articleid=1809665

- System equations:

$$\begin{aligned}\dot{x} &= f(x,u,w,t), \qquad w \sim (0,Q)\\ y &= h(x,v,t), \qquad v \sim (0,R)\end{aligned}$$

- Compute partial derivative matrices:

$$A = \partial f/\partial x|_{\hat{x}}\,, L = \partial f/\partial w|_{\hat{x}}\,, C = \partial h/\partial x|_{\hat{x}}\,, M = \partial h/\partial v|_{\hat{x}}$$

- Compute $\tilde{Q} = LQL^T$, $\tilde{R} = MRM^T$
- Kalman filter equations:

$$\begin{aligned}\hat{x}(0) &= E[x(0)], \quad P(0) = E\left[(x(0)-\hat{x}(0))(x(0)-\hat{x}(0))^T\right]\\ \dot{\hat{x}} &= f(\hat{x},u,0,t) + K\left[y - h(\hat{x},0,t)\right], \quad K = PC^T\tilde{R}^{-1}\\ \dot{P} &= AP + PA^T + \tilde{Q} - PC^T\tilde{R}^{-1}CP\end{aligned}$$

Robot State Estimation: `Robot.zip`

- First we write a simulation for the dynamic system model: `simGRF.mdl` and `statederCCFforce.m`
- Then we write a controller: `PBimpedanceCCFfull.m`
- Then we calculate the A matrix: `CalcFMatrix.m` and `EvaluateFMatrix.m`
- Then we write a Kalman filter: `zhatdot.m`
- Run the program:
  - Run `setup.m`
  - Run `simGRF.mdl`
  - Look at the output plots:
    - Run `plotter.m` to see control performance
    - Open "Plotting - 1 meas" block to see estimator performance
    - Open the "q1, q1hat" scope to see hip position
    - Open the "q2, q2hat" scope to see thigh angle
    - Open the "q3meas, q3, q3hat" scope to see knee angle

Additional Estimation Topics

- Nonlinear estimation
  - Iterated EKF
  - Second-order EKF
  - Unscented Kalman filter
  - Particle filter
  - Many others ...
- Parameter estimation
- Smoothing
- Adaptive filtering
- Robust filtering (H_∞ filtering)
- Constrained filtering