1. Derive discrete linear Kalman Filter
2. Steady state (optional)
3. Cramer Rao (optional)

Truth model:

$$\underline{x}_{k+1} = \Phi_k \underline{x}_k + \Gamma_k \underline{u}_k + \Upsilon_k \underline{w}_k \qquad (3.27)$$

$$\tilde{\underline{y}}_k = H_k \underline{x}_k + \underline{v}_k$$

$\underline{v}_k$ & w_k are zero-mean Gaussian noise processes:

(3.28-9)

$$E\left\{ \underline{v}_k \underline{v}_j^T \right\} = \begin{cases} 0 & k \neq j \\ R_k & k = j \end{cases}$$

$$E\left\{ \underline{w}_k \underline{w}_j^T \right\} = \begin{cases} 0 & k \neq j \\ Q_k & k = j \end{cases}$$

(block) diagonal structure

$$E\left\{ \underline{v}_k \underline{w}_k^T \right\} = 0 \text{ for all } k$$

Assume structure:

$$\hat{\underline{x}}^{-}_{k+1} = \Phi \hat{\underline{x}}^{+}_{k} + \Gamma_k \underline{u}_k \quad \text{prediction}$$

$$\hat{x}^{+}_{k} = \hat{x}^{-}_{k} + K_k \left[\tilde{\underline{y}}_k - H_k \hat{\underline{x}}^{-}_{k} \right] \quad \text{update} \tag{3.30}$$

<u>Find optimal K.</u>

First, define error covariances:

$$P^{-}_{k+1} \equiv E\left\{ \tilde{\underline{x}}^{-}_{k+1} \tilde{\underline{x}}^{-T}_{k+1} \right\}$$

$$P^{+}_{k+1} \equiv E\left\{ \tilde{\underline{x}}^{+}_{k+1} \tilde{\underline{x}}^{+T}_{k+1} \right\} \tag{3.31}$$

$\ominus$: prediction

$\oplus$: update

$$\underbrace{\tilde{\underline{x}}^{-}_{k+1}}_{\text{error}} = \underbrace{\hat{\underline{x}}^{-}_{k+1}}_{\text{est.}} - \underbrace{\underline{x}_{k+1}}_{\text{true}} \tag{3.32}$$

Derive expressions for $P_{k+1}^{+/-}$

plug 3.27 & 3.30 into 3.32

$$\underline{\tilde{x}}_{k+1}^{-} = \Phi_k \underline{\tilde{x}}_k^{+} - \gamma_k \underline{w}_k \qquad (3.33)$$

note: not a function of u_k, because that is known exactly.

now we can derive P_{k+1}^{-}

$$P_{k+1}^{-} = E\left\{ \underline{\tilde{x}}_{k+1}^{-} \, \underline{\tilde{x}}_{k+1}^{-\,T} \right\} \qquad (3.34)$$

$$= E\left\{ \Phi_k \underline{\tilde{x}}_k^{+} \, \Phi_k \underline{\tilde{x}}_k^{+\,T} \right\}$$

$$- \cancel{E\left\{ \Phi_k \underline{\tilde{x}}_k^{+} w_k^{T} \gamma_k^{T} \right\}} = 0$$

$$- \cancel{E\left\{ \gamma_k \underline{w}_k \underline{\tilde{x}}_k^{+\,T} \Phi_k^{T} \right\}} = 0$$

$$+ E\left\{ \gamma_k \underline{w}_k \underline{w}_k^{T} \gamma_k^{T} \right\}$$

From 3.27 we see $\underline{w}_k$ & $\tilde{x}_k^+$ are uncorrelated. ($\tilde{\underline{x}}_{k+1}$ depends on $\underline{w}_k$)

$$\Rightarrow E\{\tilde{x}_k^+ w_k^T\} = E\{w_k \tilde{x}_k^{+T}\} = 0$$

Simplifying & plugging in 3.28-9:

$$P_{k+1}^- = \Phi_k P_k^+ \Phi_k^T + \gamma_k Q_k \gamma_k^T \qquad (3.35)$$

& initial cond: $P_0^- = E\{\tilde{\underline{x}}_0^- \tilde{\underline{x}}_0^{-T}\}$

Now develop expr. for P_k^+

plug 3.27 into 3.30, and that into 3.32:

$$\tilde{\underline{x}}_k^+ = (I - K_k H_k)\hat{\underline{x}}_k^- + K_k H_k \underline{x}_k + K_k \underline{v}_k - x_k$$

replace w/ $\tilde{\underline{x}}_k^-$

$$\tilde{\underline{x}}_k^+ = (I - K_k H_k)\tilde{\underline{x}}_k^- + K_k \underline{v}_k$$

Following similar approach as with P_{k+1}^-

$$P_k^+ = E\{\underline{\tilde{x}}_k^+ \ \underline{\tilde{x}}_k^{+T}\} \quad (3.39)$$

$$= [I - K_k H_k] P_k^- [I - K_k H_k]^T + K_k R_k K_k^T$$

To find K_k, minimize trace (P_k^+)

(equivalent to minimizing length of the estimation error vector) minimum variance approach

$$K_k = P_k^- H_k^T [H_k P_k^- H_k^T + R_k]^{-1} \quad (3.42)$$

&

$$P_k^+ = [(P_k^-)^{-1} + H_k^T R_k^{-1} H_k]^{-1} \quad (3.45)$$

Note: 3.45 shows that update decrease covariance, while 3.35 shows that propagation increases covariance.

$$K_k = P_k^+ H_k^T R_k^{-1} \quad \leftarrow \text{alt. form.}$$

Table 3.1: Discrete-Time Linear Kalman Filter

Model	$x_{k+1} = \Phi_k x_k + \Gamma_k u_k + \Upsilon_k w_k, \quad w_k \sim N(0, Q_k)$ $\tilde{y}_k = H_k x_k + v_k, \quad v_k \sim N(0, R_k)$
Initialize	$\hat{x}(t_0) = \hat{x}_0$ $P_0 = E\{\tilde{x}(t_0)\tilde{x}^T(t_0)\}$
Gain	$K_k = P_k^- H_k^T [H_k P_k^- H_k^T + R_k]^{-1}$
Update	$\hat{x}_k^+ = \hat{x}_k^- + K_k[\tilde{y}_k - H_k \hat{x}_k^-]$ $P_k^+ = [I - K_k H_k] P_k^-$
Propagation	$\hat{x}_{k+1}^- = \Phi_k \hat{x}_k^+ + \Gamma_k u_k$ $P_{k+1}^- = \Phi_k P_k^+ \Phi_k^T + \Upsilon_k Q_k \Upsilon_k^T$

Implement

Extend to handle missing measurement.

Steady State KF

Propagating the $n \times n$ matrix for $P^{+/-}$ is computationally expensive.

It typically converges quickly, and a steady state value for P, K can be determined.

Table 3.2: Discrete and Autonomous Linear Kalman Filter

Model	$x_{k+1} = \Phi x_k + \Gamma u_k + \Upsilon w_k, \quad w_k \sim N(0,Q)$ $\tilde{y}_k = H x_k + v_k, \quad v_k \sim N(0,R)$
Initialize	$\hat{x}(t_0) = \hat{x}_0$
Gain	$K = PH^T[HPH^T + R]^{-1}$
Covariance	$P = \Phi P \Phi^T - \Phi P H^T [HPH^T + R]^{-1} H P \Phi^T + \Upsilon Q \Upsilon^T$
Estimate	$\hat{x}_{k+1} = \Phi \hat{x}_k + \Gamma u_k + \Phi K [\tilde{y}_k - H \hat{x}_k]$

Discrete time algebraic Riccati eqn

Can be solved for P,

See $\boxed{3.34}$

Cramer-Rao Lower Bound

$$E\left\{ (\hat{x}_k^+ - x_k)(\hat{x}_k^+ - x_k)^T \right\} \geq \mathcal{F}_k^{-1}$$

matrices of state vectors over time

"trajectory information" matrix

$$\mathcal{F}_k = -E\left\{ \frac{\partial^2}{\partial x_k \partial x_k^T} \ln\left[p(\tilde{Y}_k, x_k) \right] \right\}$$

note: joint probability, not conditional probability, b/c the state is stochastic $(Q \neq 0)$

if zero process noise, then $p(\tilde{Y}_k, x_k) = p(\tilde{Y}_k \mid x_k)$

$\mathcal{F}_k$ is a $(kn) \times (kn)$ matrix

k time points

n states

We only really care about lower right $n \times n$ block, which corresponds to the current state.

$$\mathcal{J}_k = \left[\begin{array}{c|c} & \\ \hline & J_k \end{array}\right]$$

Skipping derivation...

for discrete time linear KF:

$$J_{k+1} = Q_k^{-1} - Q_k^{-1}\Phi_k\left(J_k + \Phi_k^T Q_k^{-1}\Phi_k\right)^{-1}\Phi_k^T Q_k^{-1} \qquad (3.149)$$
$$+ H_{k+1}^T R_{k+1}^{-1} H_{k+1}$$

$$J_0 = -E\left\{\frac{\partial^2}{\partial \underline{x}_0 \partial \underline{x}_0^T} \ln\left[p(\underline{x}_0)\right]\right\} = P_0^{-1}$$

It can be shown that for the KF,

$$\left(P_{k+1}^{+}\right)^{-} = \left(J_{k+1}\right)^{-} \Rightarrow$$ KF is an "efficient" filter.

It is not possible to do any better.

Implement:

Check that your KF is efficient