MODULE 5
THE DISCRETE KALMAN FILTER
- Like the Wiener filter, the Kalman filter is a solution to the MMSE estimation problem.

- There are two main differences between the Kalman filter and the Wiener filter:
  1. For the Kalman filter, the problem is formulated in state space.
  2. Whereas the Wiener filter is the optimal LSI estimator, the Kalman filter is time varying in general.

- Thus, we begin by reviewing the relationship between our standard spectral representation of a stationary stochastic process and the "controllable canonical form" state space model.

- Consider a linear system

  \[ \begin{align*}
  u(t) & \rightarrow \text{system} \rightarrow y(t) \\
  \end{align*} \]

  where \( u(t) \) and \( y(t) \) could be scalars or vectors.

  Each element of \( u(t) \) is a white noise process with unit-magnitude power spectrum.

- A general state space model for the system is

  \[ \begin{align*}
  \dot{x} &= Fx + Gu \\
  y &= Bx \\
  \end{align*} \]
where:

- $F, G$ are matrices of appropriate size.  
- $B$ is a matrix or a vector of appropriate size.  
- $X$ is a vector of intermediate variables called "state variables" or "phase variables".  

$\rightarrow X$ is called the "state vector" of the system.

- The state space model is not unique. For any given system, there exist uncountably infinitely many models of the form $(\ast)$ on PAGE 5.1.

- For a system with a transfer function that is rational, a finite dimensional model of the form $(\ast)$ on PAGE 5.1 always exists.

- For the moment, let us assume that $u(t)$ and $y(t)$ are scalar processes and that the PSD $S_y(s)$ of $y(t)$ is known.

- We want to model $y(t)$ as the response of a linear system, where the system input is the unity power spectrum white noise $u(t)$.

Note: this implies that $E[x(t)] = 0$. 

...
- We begin by spectrally factorizing \( S_y(s) \):

\[
S_y(s) = S_y^+(s) S_y^-(s)
\]

where

\( S_y^+(s) \) has left half-plane poles and zeros only.

\( S_y^-(s) \) has right half-plane poles and zeros only.

- Now, the shaping filter required to generate \( y(t) \) with nontrivial correlation structure from the process \( x(t) \) with trivial correlation structure is simply \( S_y^+(s) \).

- Suppose \( S_y^+(s) = \frac{b_m s^m + b_{m-1} s^{m-1} + \ldots + b_0}{s^n + a_{n-1} s^{n-1} + \ldots + a_0} \), where \( m = n-1 \).

- We have then

\[
\begin{array}{ccc}
\text{u}(t) & \rightarrow & \frac{b_m s^m + b_{m-1} s^{m-1} + \ldots + b_0}{s^n + a_{n-1} s^{n-1} + \ldots + a_0} \\
& & \rightarrow \ y(t)
\end{array}
\]
- We first "break apart" the denominator and numerator of $S_y(s)$.

- Between the denominator and numerator, we introduce the intermediate process $r(t)$:

$$U(t) \xrightarrow{s^n + a_{n-1}s^{n-1} + \ldots + a_0} r(t) \xrightarrow{b_m s^m + b_{m-1} s^{m-1} + \ldots + b_0} y(t)$$

- To be concrete, let us assume that $n=4$, $b=3$.

- We next define the state variables according to:

$$X_1 = r(t)$$
$$X_2 = \dot{r}(t) = \dot{x}_1$$
$$X_3 = \ddot{r}(t) = \dot{x}_2$$
$$X_4 = \dddot{r}(t) = \dot{x}_3$$

- In the Laplace domain, the relationship between $U(s)$ and $R(s)$ is

$$U(s) = s^4R(s) + a_3s^3R(s) + a_2s^2R(s) + a_1sR(s) + a_0R(s)$$

- so

$$u(t) = \dddot{r}(t) + a_3\ddot{r}(t) + a_2\dot{r}(t) + a_1r(t) + a_0r(t)$$
- We have

\[ F(t) = -a_0 r(t) - a_1 \dot{r}(t) - a_2 \ddot{r}(t) - a_3 \dddot{r}(t) + u(t) \]

\[ \dot{x}_4 = -a_0 x_1 - a_1 x_2 - a_2 x_3 - a_3 x_4 + u(t) \]

- Writing this last equation together with equations (4) on page 5.4 using matrix notation, we have

\[
\begin{bmatrix}
\dot{x}_1 \\
\dot{x}_2 \\
\dot{x}_3 \\
\dot{x}_4
\end{bmatrix} =
\begin{bmatrix}
0 & 1 & 0 & 0 \\
0 & 0 & 1 & 0 \\
0 & 0 & 0 & 1 \\
-a_0 & -a_1 & -a_2 & -a_3
\end{bmatrix}
\begin{bmatrix}
x_1 \\
x_2 \\
x_3 \\
x_4
\end{bmatrix} +
\begin{bmatrix}
0 \\
0 \\
0 \\
1
\end{bmatrix} u(t) \tag{4}
\]

\[ \dot{X} = FX + GU \]

- Now, we redraw the block diagram on page 5.4 as

\[ u(t) \rightarrow \frac{1}{s^4 + a_3 s^3 + a_2 s^2 + a_1 s + a_0} \rightarrow \begin{bmatrix} b_0 \\ b_1 s \\ b_2 s^2 \\ b_3 s^3 \end{bmatrix} \rightarrow \Sigma \rightarrow y(t) \]

\[ y(t) = b_0 x_1 + b_1 x_2 + b_2 x_3 + b_3 x_4 \]
In matrix notation,

\[ y(t) = \begin{bmatrix} b_0 & b_1 & b_2 & b_3 & b_4 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} \]

\[ y = B^\top x \]

Together, the state equation (*) on PAGE 5.5 and the output equation (*) above constitute the controllable canonical state variable model for the system.

In general, this model is given by

\[
\begin{bmatrix}
\dot{x}_1 \\
\dot{x}_2 \\
\vdots \\
\dot{x}_n
\end{bmatrix} =
\begin{bmatrix}
0 & 1 & 0 & \cdots \\
0 & 0 & 1 & \cdots \\
\vdots & \ddots & \ddots & \ddots \\
-a_0 & -a_1 & \cdots & -a_n
\end{bmatrix}
\begin{bmatrix}
x_1 \\
x_2 \\
\vdots \\
x_n
\end{bmatrix} +
\begin{bmatrix}
0 \\
0 \\
\vdots \\
1
\end{bmatrix} u(t)
\]

\[ y(t) = \begin{bmatrix} b_0 & b_1 & \cdots & b_m \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_m \end{bmatrix} \]

Carefully study examples 5.1 and 5.2 on pages 194-196 of the book.
The fundamental requirement that determines whether a stochastic process $y(t)$ can be formulated as a state space model of the type we have just examined is this:

such a model exists if $y(t)$ can be related to a white noise through a linear differential equation.

Several examples are given on pages 196-198 of the book. Study them carefully.

**Discrete-Time State Space Model**

- First, we look at the relationship between the controllable canonical form and the ARMA model.
- Let $W_k$ be a white noise vector with known diagonal covariance matrix $Q_k$ given by

$$E[W_kW_i^T] = Q_k \delta_{i-k}.$$  

**NOTE:** In general, the diagonal matrix $Q_k$ will be time varying.
- Assume for the moment that \( w_k \) is a scalar process and let the scalar process \( y_k \) be related to \( u_k \) through an LSI shaping filter with a rational transfer function.

- We want to construct the controllable canonical state space model for the process \( y_k \).

- In the Kalman filtering literature, it is customary to write \( w(k) \) and \( y(k) \) for the discrete-time processes \( w_k \) and \( y_k \).

- We have

\[
W(k) \rightarrow \frac{\beta_m z^m + \beta_{m-1} z^{m-1} + \ldots + \beta_0}{z^n + \alpha_{n-1} z^{n-1} + \ldots + \alpha_0} \rightarrow y(k),
\]

where \( m = n - 1 \).

- To be concrete, let us assume that \( n = 4 \), so that \( m = 3 \).

- As before, we pull apart the numerator and denominator of the transfer function.

- Between them, we define an intermediate stochastic process \( r(k) \).
- Define the state variables according to (with n=4):

\[
\begin{align*}
    x_1(k) &= r(k) \\
    x_2(k) &= r(k+1) = x_1(k+1) \\
    x_3(k) &= r(k+2) = x_2(k+1) \\
    x_4(k) &= r(k+3) = x_3(k+1)
\end{align*}
\]

- The relationship between \( W(z) \) and \( R(z) \) is

\[
\begin{align*}
    W(z) &= \alpha_0 R(z) + \alpha_1 z R(z) + \alpha_2 z^2 R(z) + \alpha_3 z^3 R(z) + z^4 R(z) \\
    w(k) &= \alpha_0 r(k) + \alpha_1 r(k+1) + \alpha_2 r(k+2) + \alpha_3 r(k+3) + r(k+4) \\
    r(k+4) &= -\alpha_0 r(k) - \alpha_1 r(k+1) - \alpha_2 r(k+2) - \alpha_3 r(k+3) + w(k) \\
    x_4(k+1) &= -\alpha_0 x_1(k) - \alpha_1 x_2(k) - \alpha_2 x_3(k) - \alpha_3 x_4(k) + w(k)
\end{align*}
\]

- From the figure above, we have also that

\[
y(k) = \beta_0 x_1(k) + \beta_1 x_2(k) + \beta_2 x_3(k) + \beta_3 x_4(k)
\]
Writing equations (**) and equation (***) on PAGE 5.9 together using matrix notation, we obtain the controllable canonical state variable form

\[
\begin{bmatrix}
    x_1(k+1) \\
    x_2(k+1) \\
    x_3(k+1) \\
    x_4(k+1)
\end{bmatrix} =
\begin{bmatrix}
    0 & 1 & 0 & 0 \\
    0 & 0 & 1 & 0 \\
    0 & 0 & 0 & 1 \\
    -\alpha_0 & -\alpha_1 & -\alpha_2 & -\alpha_3
\end{bmatrix}
\begin{bmatrix}
    x_1(k) \\
    x_2(k) \\
    x_3(k) \\
    x_4(k)
\end{bmatrix} +
\begin{bmatrix}
    0 \\
    0 \\
    0 \\
    1
\end{bmatrix}
\begin{bmatrix}
    w(k)
\end{bmatrix}
\]

\[y(k) = [\beta_0 \beta_1 \beta_2 \beta_3]
\begin{bmatrix}
    x_1(k) \\
    x_2(k) \\
    x_3(k) \\
    x_4(k)
\end{bmatrix}\]

OR,

\[x_{k+1} = \Phi_k x_k + w_k\]
\[y_k = B_k x_k\]

where
\[x_k = \text{state vector at time } k,\]
\[\Phi_k = \text{state transition matrix for } x_k \rightarrow x_{k+1},\]
\[B_k = \text{output equation gain matrix}\]
- From the figure on PAGE 5.8, we have

\[
\frac{Y(z)}{W(z)} = \frac{\beta_m z^m + \beta_{m-1} z^{m-1} + \cdots + \beta_0}{z^n + \alpha_{n-1} z^{n-1} + \cdots + \alpha_0}
\]

- Thus, the ARMA model relating \( y(k) \) and \( w(k) \) is:

\[
y(k+n) + \alpha_{n-1} y(k+n-1) + \alpha_{n-2} y(k+n-2) + \cdots + \alpha_0 y(k) = \beta_m w(k+n) + \beta_{m-1} w(k+n-1) + \beta_{m-2} w(k+n-2) + \beta_0 w(k)
\]

- The corresponding controllable canonical state space model is:

\[
\begin{bmatrix}
x_1(k+1) \\
x_2(k+1) \\
\vdots \\
x_n(k+1)
\end{bmatrix} = \begin{bmatrix}
0 & 1 & 0 & 0 & \cdots \\
0 & 0 & 1 & 0 & \cdots \\
\vdots & \vdots & \ddots & \ddots & \vdots \\
-\alpha_0 & -\alpha_1 & -\alpha_2 & \cdots & -\alpha_{n-1}
\end{bmatrix} \begin{bmatrix}
x_1(k) \\
x_2(k) \\
\vdots \\
x_n(k)
\end{bmatrix} + w(k)
\]

\[
y(k) = \begin{bmatrix}
\beta_0 & \beta_1 & \beta_2 & \cdots & \beta_n
\end{bmatrix} \begin{bmatrix}
x_1(k) \\
x_2(k) \\
\vdots \\
x_n(k)
\end{bmatrix}
\]
Sampling of a Continuous-Time System

- Sampling of the continuous-time output equation

\[ y = Bx \]

is obvious, so we concentrate on the state equation.

- The continuous-time state equation is

\[ \dot{x} = Fx + Gu \]

- We sample this equation at discrete times \( t_0, t_1, t_2, \ldots, t_k \) \ldots to obtain

\[ x(t_{k+1}) = \Phi(t_{k+1}, t_k) x(t_k) + \int_{t_k}^{t_{k+1}} \Phi(t_{k+1}, \tau) G(\tau) u(\tau) d\tau \]

\[ \Phi_k \]

\[ w_k \]

\[ \Rightarrow x_{k+1} = \Phi_k x_k + w_k \] (8.3)

- It is necessary to solve for \( \Phi_k \) and \( Q_k \) from the specification of the continuous-time system.

- If the matrix \( F \) is not time varying and if the sampling instants are equally spaced by a time \( \Delta t \), then

\[ \Phi_k = \left[ L^{-1} \{ (sI - F)^{-1} \} \right]_{t=\Delta t} \]
- The input covariance matrix is given by

\[ E\left[w_k w_i^T\right] = Q_k \delta(k-i) \], where

\[
Q_k = E \left[ \int_{t_k}^{t_{k+1}} \Phi(t_{k+1}, \theta) G(\theta) u(\theta) d\theta \left( \int_{t_k}^{t_{k+1}} \Phi(t_{k+1}, \psi) G(\psi) u(\psi) d\psi \right)^T \right]
\]

\[
= \int_{t_k}^{t_{k+1}} \int_{t_k}^{t_{k+1}} \Phi(t_{k+1}, \theta) G(\theta) E\left[ u(\theta) u^T(\psi) \right] G^T(\psi) \Phi^T(t_{k+1}, \psi) d\psi d\theta.
\]

**NOTE:** Since \( u(t) \) is a white noise,

\[ E\left[ u(\theta) u^T(\psi) \right] \] is a diagonal matrix.

\[ \rightarrow \] The diagonal entries are Dirac deltas.

- Study example 5.3 on page 200 of the book.

- A numerical method for finding \( Q_k \) is given on page 204 of the book.

**NOTE:** if \( F \) is time varying, \( \Phi_k \) must be obtained by solving the differential equation

\[
\dot{\Phi}(t, t_k) = F(t) \Phi(t, t_k) \quad ; \quad \Phi(t_k, t_k) = I.
\]

- If \( F(t) \) does not vary from \( t_k \) to \( t_{k+1} \), then

\[ \Phi_k = e^{F(t_{k+1} - t_k)} \] (matrix exponential).

**PAGE 5.13**
A Development of the Discrete Kalman Filter

- In a nutshell:
  - There is a discrete linear system.
  - The input is white noise.
  - Your observations are the system output plus a white noise called the "measurement noise."
  → The system input noise and the measurement noise are uncorrelated with each other.
  - You know:
    - The state space model for the system.
    - The 2nd order statistics of the input noise.
    - The 2nd order statistics of the measurement noise.
  - The problem:
    - Given the noisy observations of the output, find MMSE estimates of the system state vector.

NOTE: if you can get MMSE estimates of the state vector, then it is trivial to turn these into MMSE estimates of an arbitrary affine function of the state vector... which is a statement of the general Kalman filtering problem.
The system output itself is an affine function of the state vector. Thus, one of the most straight-forward Kalman filtering problems is:

given a noisy observation of the output, find MMSE estimates of the output.

The system state space model:

\[ x_{k+1} = \Phi_k x_k + w_k \]

\[ y_k = H_k x_k \]
The state transition vector $x_k$ is the system output $y_k$ plus the input measurement noise $v_k$:

$$Z_k = H_k x_k + v_k$$

- The covariance structures of the input noise and measurement noise are known:

$$E[W_k w_i^T] = \begin{cases} Q_k, & i=k \\ 0, & i \neq k \end{cases}$$

$$E[v_k v_i^T] = \begin{cases} R_k, & i=k \\ 0, & i \neq k \end{cases}$$

- The input noise and measurement noise are uncorrelated:

$$E[W_k v_i^T] = 0 \quad (\forall i, k)$$

All together:

$$x_{k+1} = \phi_k x_k + w_k \quad (5.5.1)$$

$$Z_k = H_k x_k + v_k \quad (5.5.2)$$

$$E[w_k w_i^T] = Q_k \delta(i-k) \quad (5.5.3)$$

$$E[v_k v_i^T] = R_k \delta(i-k) \quad (5.5.4)$$

$$E[w_k v_i^T] = 0 \quad (5.5.5)$$
What makes the Kalman filter different from anything that came before is the recursive way it is formulated.

In this regard, it is kind of like a mathematical proof by induction.

Assume that somehow we have obtained a prediction for the state vector at time \( k \), and that this estimate is based on the first \( k-1 \) observations.

In other words, assume that we have an estimate of \( \hat{X}_k \) given \( \ldots, Z_{k-3}, Z_{k-2}, Z_{k-1} \).

This is often written \( \hat{X}_{k\mid k-1} \).

But in the book it is written \( \hat{X}_k^- \).

Note: \( \hat{X}_k^- \) is an "a priori estimate" or "prior" of \( X_k \), the true state vector at time \( k \).

Note: The assumption that we "somehow" have this prediction \( \hat{X}_k^- \) is not a problem: in the worst case, we can just set \( \hat{X}_k^- = E[X_k] \), which is usually zero...
**NOTE:** \( E[X_k] \) is computable (solvable). If it turns out to be nonzero, then it is customary to subtract it out of the model. This leads to a modified system model where \( E[X_k] = 0 \).

**NOTE:** \( \hat{X}_k^- \) might be a good estimate or a poor estimate... this doesn't matter. What does matter is that we need \( \hat{X}_k^- \) to be unbiased. That is, we need it to be true that

\[
E[\hat{X}_k^-] = E[X_k].
\]

- So the error in the prediction \( \hat{X}_k^- \) is

\[
E_k^- = X_k - \hat{X}_k^- \quad (5.5.6)
\]

with mean

\[
E[e_k^-] = E[X_k - \hat{X}_k^-] = E[X_k] - E[\hat{X}_k] = 0.
\]

- \( e_k^- \) is the "predicted state vector error."
- The predicted state vector error covariance matrix is:

\[
P_k^- = E[e_k^- e_k'^-] = E[(X_k - \hat{X}_k^-)(X_k - \hat{X}_k^-)^T] \quad (5.5.7)
\]
- Now, given $\hat{x}_k$ and the observation $z_k$, we want to "refine" the prediction $\hat{x}_k$ by using the new information in $z_k$ in an optimal way.

- This will give us an "a posteriori" estimate of $x_k$ given $k$ observations (a "filtered" estimate).

- Often, this is written $\hat{x}_{k|k}$.

- In the book, it is written $\hat{x}_k$.

- The strategy is to make the new & improved estimate $\hat{x}_k$ a linear combination, or "blending" of the old estimate $\hat{x}_k$ and the new observation $z_k$.

- The "blending coefficients" are going to be written in a weird way that will prove useful a little bit later:

\[
\hat{x}_k = \left( I - K_k H_k \right) \hat{x}_k^- + K_k z_k
\]

- blending factor for the old a priori estimate $\hat{x}_k^-$

- blending factor for the new observation $z_k$. 

PAGE 5.19
\[ \hat{x}_k = \hat{x}_k^- + K_k (z_k - H_k \hat{x}_k^-) \] (5.5.8)

- We will seek to find a value for \( k \) that makes \( \hat{x}_k \) optimal in the MMSE sense.

**NOTE:** in (5.5.8), the term \( z_k - H_k \hat{x}_k^- \) may be considered as a "residual"... because, according to the measurement equation (5.5.2) on PAGE 5.16, \( H_k \hat{x}_k^- \) is in fact an a priori estimate for \( z_k \).

The error in this estimate, \( z_k - H_k \hat{x}_k^- \), has two parts:

- The part due to the measurement noise \( \nu_k \) (undesirable).

- The part due to **new information** that is present in \( z_k \) but was not present in ... \( z_{k-3}, z_{k-2}, z_{k-1} \) (desirable!)
The error in the a posteriori filtered state vector estimate $\hat{x}_k$ is

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

true estimated

The filtered state vector error covariance matrix is

$$P_k = E[e_k e_k^T]$$

$$= E\left[ (x_k - \hat{x}_k)(x_k - \hat{x}_k)^T \right]$$

use (5.5.8) $= E\left\{ \left[ x_k - \hat{x}_k - K_k(z_k - H_k \hat{x}_k^-) \right][\text{same}]^T \right\}$

use (5.5.2) $= E\left\{ \left[ x_k - \hat{x}_k^- - K_k(H_k x_k + v_k - H_k \hat{x}_k^-) \right][\text{same}]^T \right\}$

$= E\left\{ \left[ x_k - \hat{x}_k^- - K_k H_k (x_k - \hat{x}_k^-) - K_k v_k \right][\text{same}]^T \right\}$

$= E\left\{ \left[ (1 - K_k H_k)(x_k - \hat{x}_k^-) - K_k v_k \right] \right\}$

$\times \left[ (x_k - \hat{x}_k^-)^T (1 - K_k H_k)^T - v_k^T K_k^T \right]$
\[ P_k = ... \]
\[ = E \{ (I - k_k H_k)(x_k - \hat{x}_k^-)(x_k - \hat{x}_k^-)^T (I - k_k H_k)^T \]
\[ - (I - k_k H_k)(x_k - \hat{x}_k^-) V_k^T k_k^T \]
\[ - k_k V_k (x_k - \hat{x}_k^-)^T (I - k_k H_k)^T + k_k V_k V_k^T k_k^T \} \]
\[ = (I - k_k H_k) E \{ (x_k - \hat{x}_k^-)(x_k - \hat{x}_k^-)^T \} (I - k_k H_k)^T \]
\[ - (I - k_k H_k) E \{ (x_k - \hat{x}_k^-) V_k^T \} k_k^T \]
\[ - k_k E \{ V_k (x_k - \hat{x}_k^-)^T \} (I - k_k H_k)^T \]
\[ + k_k E \{ V_k V_k^T \} k_k^T \]
\[ = (I - k_k H_k) P_k^- - (I - k_k H_k)^T + k_k R_k K_k^T \]

\[ P_k = (I - k_k H_k) P_k^- (I - k_k H_k)^T + k_k R_k K_k^T \] (5.5.11)

- This is true for any choice of \( k_k \). The question is: which choice is optimal?
- Recall:
  - \( \hat{X}_k \) is the filtered, or "updated" state vector estimate obtained using \( k \) observations.
  - The error in \( \hat{X}_k \) is \( E_k = X_k - \hat{X}_k \).
  - Assuming that \( \hat{X}_k \) is an unbiased estimate of \( X_k \), the error the filtered state vector error covariance matrix is
    \[
    P_k = E[e_k e_k^T] = (5.5.11) \ldots \text{which depends on } k_k.
    \]
  - The main diagonal entries of \( P_k \) are the variances of the estimation errors in each component of \( \hat{X}_k \).
  - So what we want is a choice for the gain or "blending factor" \( K_k \) that will minimize the sum of the main diagonal entries in \( P_k \).
  - So we will take the derivative of the trace \( \text{Tr} P_k \) with respect to \( K_k \), set this equal to zero, and solve for the optimal \( K_k \).
**NOTE:** \( \text{Tr} A \) is a scalar... it is the sum of the main diagonal elements of the matrix \( A \).

**Note:** The derivative of a scalar \( \theta \) with respect to an \( m \times n \) matrix \( A = \begin{bmatrix} a_{1,1} & a_{1,2} & \cdots & a_{1,n} \\ a_{2,1} & a_{2,2} & \cdots & a_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m,1} & a_{m,2} & \cdots & a_{m,n} \end{bmatrix} \) is given by

\[
\frac{d\theta}{dA} = \begin{bmatrix}
\frac{d\theta}{da_{1,1}} & \frac{d\theta}{da_{1,2}} & \cdots & \frac{d\theta}{da_{1,n}} \\
\frac{d\theta}{da_{2,1}} & \frac{d\theta}{da_{2,2}} & \cdots & \frac{d\theta}{da_{2,n}} \\
\vdots & \vdots & \ddots & \vdots \\
\frac{d\theta}{da_{m,1}} & \frac{d\theta}{da_{m,2}} & \cdots & \frac{d\theta}{da_{m,n}}
\end{bmatrix}
\]

**FACTS From Matrix Theory:**

- if \( A \) and \( B \) are conformable matrices and \( AB \) is a square matrix, then

\[
\frac{d(\text{Tr}AB)}{dA} = B^T
\]
Matrix Theory Facts...

- If $A$ and $C$ are conformable matrices and $C$ is a symmetric matrix, then

$$\frac{d}{dA} (\text{Tr } AC A^T) = 2AC$$

- Now let us return to the expression (5.5.11) for $P_k$ on page 5.22, and rewrite it as

$$P_k = (I - K_k H_k) P_k^- (I - K_k H_k)^T + K_k R_k K_k^T \quad (5.5.11)$$

$$= (P_k^- - K_k H_k P_k^-) (I - H_k^T K_k^T) + K_k R_k K_k^T$$

$$= P_k^- - P_k^- H_k^T K_k^T - K_k H_k P_k^- + K_k H_k P_k^- H_k^T K_k^T + K_k R_k K_k^T$$

$$= P_k^- - K_k H_k P_k^- - P_k^- H_k^T K_k^T + K_k (H_k P_k^- H_k^T + R_k) K_k^T$$

- Now differentiate \( \text{Tr } P_k \) with respect to $K_k$:

$$\frac{d}{dK_k} (\text{Tr } P_k) = \frac{d}{dk_k} \text{Tr } P_k^- - \frac{d}{dk_k} \text{Tr } K_k H_k P_k^- - \frac{d}{dk_k} \text{Tr } P_k^- H_k^T K_k^T$$

$$+ \frac{d}{dk_k} \text{Tr } K_k (H_k P_k^- H_k^T + R_k) K_k^T$$
\[
= - (H_K P_K^-)^T - H_K^T K_K^T + 2 K_K (H_K P_K^- H_K^T + R_K^-) \\
\approx (H_K K_K)^T \\
= -2 (H_K P_K^-)^T + 2 K_K (H_K P_K^- H_K^T + R_K) \quad (5.5.16)
\]

So the optimal "Kalman gain" $K_K$ is found by solving

\[-2 (H_K P_K^-)^T + 2 K_K (H_K P_K^- H_K^T + R_K) = 0
\]

\[(H_K P_K^-)^T = K_K (H_K P_K^- H_K^T + R_K) \]

\[K_K = (H_K P_K^-)^T (H_K P_K^- H_K^T + R_K)^{-1} \quad (5.5.17)
\]
- We started with the state model and observation equation:

\[ x_{k+1} = \phi_k x_k + w_k \quad (5.5.1) \]

\[ z_k = H_k x_k + v_k \quad (5.5.2) \]

- We assumed that we were given an initial a priori state vector estimate \( \hat{x}_k^- \) with error

\[ e_k^- = x_k - \hat{x}_k^- \quad (5.5.6) \]

and predicted state vector error covariance matrix

\[ P_k^- = E[ e_k^- e_k^{\top} ] = E[(x_k - \hat{x}_k^-)(x_k - \hat{x}_k^-)^\top]. \quad (5.5.7) \]

- We assumed an update equation to revise the prediction \( \hat{x}_k^- \) using the observation \( z_k \):

\[ \hat{x}_k = \hat{x}_k^- + K_k (z_k - H_k \hat{x}_k^-) \quad (5.5.8) \]

- Giving a smoothed state vector error covariance matrix of

\[ P_k = (I - K_k H_k) P_k^- (I - K_k H_k)^\top + K_k R_k K_k^\top \quad (5.5.11) \]

\[ (5.5.18) \]
- We found the optimal MMSE Kalman gain vector $K_k$ by solving
\[ \frac{\partial \text{Tr}(P_k)}{\partial K_k} = 0 \]
for $K_k$, giving
\[ K_k = P_k^{-1}H_k^T(H_kP_k^{-1}H_k^T + R_k)^{-1} \] (5.5.17)

Now, we use this to simplify the general expression (5.5.11) for $P_k$ in the special case that $K_k$ is the optimal MMSE gain solution:
\[ P_k = (I - K_kH_k)P_k^{-1}(I - K_kH_k)^T + K_kR_kK_k^T \] (5.5.11) (5.5.18)
\[ = (P_k^{-1} - K_kH_kP_k^{-1})(I - H_k^TK_k^T) + K_kR_kK_k^T \]
\[ = P_k^{-1} - K_kH_kP_k^{-1} - P_k^{-1}H_k^TK_k^T + K_kH_kP_k^{-1}H_k^TK_k^T + K_kR_kK_k^T \]
\[ = P_k^{-1} - K_kH_kP_k^{-1} - P_k^{-1}H_k^TK_k^T + K_k(H_kP_k^{-1}H_k^T + R_k)K_k^T \]

Substituting in (5.5.17) for $K_k$ yields:
\[ P_k = (I - K_kH_k)P_k^{-1} \] (5.5.22)

PAGE 5.28
We now have four different equations for $P_k$:

\[
P_k = (I - k_k H_k) P_k^- (I - k_k H_k)^T + k_k R_k k_k^T \quad (5.5.11)
\]

→ good for any choice of $k_k$

\[
P_k = P_k^- - P_k^- H_k^T (H_k P_k^- H_k^T + R_k)^{-1} H_k P_k^- \quad (5.5.20)
\]

\[
P_k = P_k^- - k_k (H_k P_k^- H_k^T + R_k) k_k^T \quad (5.5.21)
\]

\[
P_k = (I - k_k H_k) P_k^- \quad (5.5.22)
\]

→ valid only for the optimal $k_k$.

→ Intermediate results obtained in going from (5.5.11/18) to (5.5.22).

Any of these four expressions can be used in practice to obtain $P_k$.

→ In various practical situations with finite precision arithmetic, one or the other of these may give superior numerical performance and/or convergence behavior.

→ Important, since it is often necessary to compute $P_k$ "on the fly" as the Kalman filter runs.
For now, we will assume that it's sufficient to calculate $P_k$ using

$$P_k = (I - K_k H_k) P_{k-1} \quad (5.5.22)$$

- Given that we now have $\hat{x}_k$ and $\hat{x}_k$, we need to "age" the state vector estimate by predicting it ahead to time $k+1$.

$\rightarrow$ Since the input $w_k$ is assumed zero mean, this is done as:

$$\hat{x}_{k+1}^- = \phi_k \hat{x}_k \quad (5.5.23).$$

- We are now in a position to repeat the whole procedure recursively.

- First, we will find the estimation error and error covariance matrix associated with the new "aged" prediction $\hat{x}_{k+1}^-$:

$$e_{k+1}^- = x_{k+1} - \hat{x}_{k+1}^-$$

$$= (\phi_k x_k + w_k) - \phi_k \hat{x}_k$$

$$= \phi_k (x_k - \hat{x}_k) + w_k$$

$$= \phi_k e_k + w_k \quad (5.5.24)$$
\[ p_{k+1} = E[e_{k+1} e_{k+1}^T] \]
\[ = E[(\phi_k e_k + w_k)(\phi_k e_k + w_k)^T] \]
\[ = E[(\phi_k e_k + w_k)(e_k^T \phi_k^T + w_k^T)] \]
\[ = E[\phi_k e_k e_k^T \phi_k^T + \phi_k e_k w_k^T + w_k e_k^T \phi_k^T + w_k w_k^T] \]
\[ = \phi_k E[e_k e_k^T] \phi_k^T + \phi_k E[e_k w_k^T] + E[w_k e_k^T] \phi_k^T \]
\[ + E[w_k w_k^T] \]
\[ \approx \phi_k p_k \phi_k^T + Q_k \quad (5.5.25) \]
The recursive procedure:

**START**: \( \hat{x}_0, P_0 \)

\[ k_k = P_k H_k^T (H_k P_k H_k^T + R_k)^{-1} \quad (5.5.17) \]

\[ \hat{x}_k = \hat{x}_k^- + K_k (z_k - H_k \hat{x}_k^-) \quad (5.5.8) \]

**Update error covariance**:
\[ P_k = (I - K_k H_k) P_k^- \quad (5.5.22) \]

**Age (update) prediction and predicted state vector error covariance matrix**
\[ \hat{x}_k^- = \phi_k \hat{x}_k \quad (5.5.23) \]
\[ P_{k+1}^- = \phi_k P_k \phi_k^T + Q_k \quad (5.5.25) \]

\[ k = k + 1 \]

Repeat

"Discrete Kalman Filter"
State Model:

\[ X_{k+1} = \Phi_k X_k + W_k \]
\[ Z_k = H_k X_k + V_k \]

\( X_k \): (nx1) state vector at time \( t_k \)
\( \Phi_k \): (nxn) state transition matrix
\( W_k \): (nx1) forcing function
\( Z_k \): (mx1) measurement vector
\( H_k \): (mxn) output matrix
\( V_k \): (mx1) measurement noise

The System:

Kalman predictor update equation:

\[ \hat{X}_{k+1} = \Phi_k \hat{X}_k \quad (5.5.23) \]
\[ = \phi_k \left[ \hat{X}_k + K_k (Z_k - H_k \hat{X}_k) \right] \quad (5.5.8) \]
\[ = \phi_k K_k (Z_k - H_k \hat{X}_k) + \phi_k \hat{X}_k \]
Let $\hat{X}_k^- = \text{"a priori" estimate of } X_k$, given only $k-1$ observations

$= \hat{X}_k^1 | z_{k-1}, z_{k-2}, ... = \hat{X}_k^{k-1}$.

Then $E_k^- = X_k - \hat{X}_k^- : \text{error in the predicted state vector.}$

The predictor will turn out to be a system that looks much like the actual system.

It will have a state vector... namely $\hat{X}_k^-$.

The Predictor:

$\rightarrow \hat{X}_{k+1}^- = \phi_k K_k (z_k - H_k \hat{X}_k^-) + \phi_k \hat{X}_k^- \quad \text{(from page 5.33)}$
The filter also has a state vector $\hat{x}_k$,

$$\hat{x}_{k+1} = \hat{x}_{k+1}^- + K_{k+1} (z_{k+1} - H_{k+1} \hat{x}_{k+1}^-)$$  \hspace{1cm} (5.5.8)

$$= \phi_k \hat{x}_k + K_{k+1} \left[ z_{k+1} - H_{k+1} \phi_k \hat{x}_k \right]$$  \hspace{1cm} (5.5.23)

$$= K_{k+1} (z_{k+1} - H_{k+1} \phi_k \hat{x}_k) + \phi_k \hat{x}_k$$

$$= P_{k+1}^{-1} H_{k+1}^T \left( H_{k+1} P_{k+1}^{-1} H_{k+1}^T + R_{k+1} \right)^{-1} (z_{k+1} - H_{k+1} \phi_k \hat{x}_k) + \phi_k \hat{x}_k$$  \hspace{1cm} (5.5.17)

$$= (\phi_k P_k \phi_k^T + Q_k) H_{k+1}^T \left[ H_{k+1} \left( \phi_k P_k \phi_k^T + Q_k \right) H_{k+1}^T + R_{k+1} \right]^{-1} (z_{k+1} - H_{k+1} \phi_k \hat{x}_k)$$  \hspace{1cm} (5.5.25)

$$= \left( \phi_k P_k \phi_k^T H_{k+1}^T + Q_k H_{k+1}^T \right) \left( H_{k+1} \phi_k P_k \phi_k^T H_{k+1}^T + H_{k+1} Q_k H_{k+1}^T + R_{k+1} \right)^{-1} (z_{k+1} - H_{k+1} \phi_k \hat{x}_k) + \phi_k \hat{x}_k$$

$$\overset{\text{def}}{=} a_k$$

$$= a_k (z_{k+1} - H_{k+1} \phi_k \hat{x}_k) + \phi_k \hat{x}_k$$

5.35
The filter: (or "smoother")

\[ Z_{k+1} \xrightarrow{+} A_k \xrightarrow{+} X_{k+1} \xrightarrow{+} Z^{-1} \xrightarrow{+} H_k \xrightarrow{+} \hat{Z}_k \]

→ The noncausality will have to be addressed.

- We must solve for the optimal \( K_k \) and \( A_k \).

\( K_k \) depends on \( P_k^- \), \( H_k \), and \( R_k \).
→ \( H_k \) and \( R_k \) are known .
→ We must still solve for \( P_k^- \).

\( A_k \) depends on \( \phi_k \), \( P_k \), \( H_{k+1} \), \( Q_k \), \( R_{k+1} \).
→ \( \phi_k \), \( H_{k+1} \), \( Q_k \), \( R_{k+1} \) are known.
→ We must still solve for \( P_k \).
Using the innovations representation and the orthogonality principle, it is possible to develop complete and independent representations for the filter and the predictor.

- This involves expressing $P_{k+1}$ as a function of $P_k^-$ and expressing $P_{k+1}$ as a function of $P_k$.

- The book does not do this, so we will skip it.

- We can get around:

  1. The "noncausality" problem with the filter.

  2. The problem of not having independent recursive formulations for $P_{k+1}$ in terms of $P_k$ and $P_{k+1}^-$ in terms of $P_k^-$.

- By making use of the equations that relate quantities in the filter and the predictor to each other:

\[
\hat{X}_K = \hat{X}_K^- + K_k (Z_k - H_k \hat{X}_K^-) \quad (5.5.8)
\]
\[
K_k = P_k^- H_k^T (H_k P_k^- H_k^T + R_k)^{-1} \quad (5.5.17)
\]
\[
P_k = (I - K_k H_k) P_k^- \quad (5.5.22)
\]
\[
P_{k+1}^- = \Phi_k P_k \Phi_k^T + Q_k \quad (5.5.25)
\]
\[
\hat{X}_{k+1}^- = \Phi_k \hat{X}_K \quad (5.5.23)
\]

- This is what is usually done in practical implementations. It leads to the procedure in Fig. 5.8 of the book.
Initialization

- In some cases, the dynamics of the system being estimated may give insight into the problem of initializing \( \hat{x}_0^- \) and \( P_0^- \). More on this later...

- In other cases, we may be able to deduce or may be given the initial state vector covariance matrix

\[
\Pi_0 = E[X_0X_0^T].
\]

\( \Rightarrow \) Then we can take

\[
\hat{x}_0^- = \mathbf{0}
\]

\[
P_0^- = \Pi_0.
\]

- An alternative procedure can be used if we consider the first observation to be \( z_{-1} \) and if \( \Pi_{-1} \) is known and if \( z_{-1} \) and \( x_{-1} \) are known to be zero-mean Gaussians.

- In this case, it may be shown with some effort that

\[
E(x_{-1} | z_{-1}) = E(x_{-1}) + \text{Cov}(x_{-1}, z_{-1}) \text{Cov}^{-1}(z_{-1}) \left[ z_{-1} - E(z_{-1}) \right]
\]

\[
= \text{Cov}(x_{-1}, z_{-1}) \text{Cov}^{-1}(z_{-1}) z_{-1}
\]

\( \Rightarrow \)
- we then take

\[
\hat{X}_{-1} = E(x_{-1} | Z_{-1})
= E(x_{-1} z_{-1}^T) \{ E(z_{-1} z_{-1}^T) \}^{-1} z_{-1}
= E \left[ x_{-1} (H_{-1} x_{-1} + V_{-1})^T \right] \{ E \left[ (H_{-1} x_{-1} + V_{-1}) (H_{-1} x_{-1} + V_{-1})^T \right] \}^{-1} z_{-1}
= E \left[ x_{-1} (x_{-1}^T H_{-1} + V_{-1}^T) \right] \{ E \left[ (H_{-1} x_{-1} + V_{-1}) (x_{-1}^T H_{-1} + V_{-1}^T) \right] \}^{-1} z_{-1}
= E \left[ x_{-1} x_{-1}^T H_{-1}^T + x_{-1} V_{-1}^T \right] \{ E \left[ H_{-1} x_{-1} x_{-1}^T H_{-1}^T + H_{-1} x_{-1} V_{-1}^T + V_{-1} x_{-1}^T H_{-1}^T + V_{-1} V_{-1}^T \right] \}^{-1} z_{-1}
= (\pi_{-1} H_{-1}^T + 0) \{ H_{-1} \pi_{-1} H_{-1}^T + 0 + 0 + R_{-1} \}^{-1} z_{-1}
= \pi_{-1} H_{-1}^T \left[ H_{-1} \pi_{-1} H_{-1}^T + R_{-1} \right]^{-1} z_{-1}.
\]

- It may also be shown using innovations and orthogonality that

\[
P_{-1} = \pi_{-1} - E \left[ \hat{X}_{-1} \hat{X}_{-1}^T \right]
= \pi_{-1} - \pi_{-1} H_{-1}^T \left[ H_{-1} \pi_{-1} H_{-1}^T + R_{-1} \right]^{-1} H_{-1} \pi_{-1}
\]

- $\hat{X}_0$ and $P_0$ are then obtained from $\hat{X}_{-1}$ and $P_{-1}$ using

\[
\hat{X}_{k+1}^{-} = \phi_k \hat{X}_k \quad (5.5.23)
\]

\[
P_{k+1}^{-} = \phi_k P_k \phi_k^T + Q_k \quad (5.5.25)
\]
Example

- We have an infrared camera mounted in a ball gimbal on an aircraft.
- It is a military aircraft, and we expect the enemy to fire threatening infrared tracking missiles at the aircraft.
- Our objective is to detect and track the missiles in our images.
- In each image where a missile is present, we want to compute two things:
  1) Our best estimate of the missile position in frame k given k observations... so we have the best possible battlefield situational awareness.
  2) Our best estimate of where the missile position will be in frame k+1 given k observations, so we can point the ball gimbal to maintain target lock.

⇒ Obviously, this problem is perfect for Kalman filtering.

Assumptions:

1. We will acquire the target when it is still far away. Thus, the target will appear as a single very bright pixel, perhaps surrounded by several relatively dimmer pixels that are somewhat brighter than the background.

2. The frame rate of our camera is very fast compared to the kinematics of the target. Thus, the target is capable of only negligible accelerations from frame to frame.
3. The horizontal and vertical motions of the target are independent. Thus, we can consider the "x" and "y" observations of the target to be noisy measurements of the outputs of two independent systems.

Target Extraction:

- We will first apply a nonlinear filter to detect the target.

- The main idea is this: at each pixel, formulate an estimate of the background. Subtract the background estimate from the pixel.

\[
\begin{array}{ccc}
I_{i,j-1} & I_{i,j} & I_{i,j+1} \\
I_{i-1,j} & I_{i,j} & I_{i+1,j} \\
I_{i-1,j+1} & I_{i,j+1} & I_{i+1,j+1}
\end{array}
\]

Background Estimate = MED[Eight Nearest Neighbors]

= MED\[I_{i-1,j-1}, I_{i,j-1}, I_{i+1,j-1}, I_{i-1,j}, I_{i,j}, I_{i+1,j}, \\
I_{i-1,j+1}, I_{i,j+1}, I_{i+1,j+1}\]

= Average of the "middle" two eight-neighbors

= \(M_{i,j}\).

Filter output: \(Y_{i,j} = I_{i,j} - M_{i,j}\).
- The median filtered image is then thresholded.
  - we could use an absolute threshold, a relative threshold (based, e.g., on the variance of the neighborhood), or both.
  - Pixels that pass the threshold are called "exceedances". They are candidate targets, denoted $E_{ij}$.

- Each time we receive a new frame, there are two things that need to be done:
  1) Associate exceedances with existing tracks and use the associated exceedances as observations to update the existing tracks.
  2) Consider starting new tracks on any exceedances that do not associate with already existing tracks.

Gating:

- The "track gate" is a rectangular window centered on the estimated position of an existing track or a candidate new track.
- Let $G$ be the width and height of the gate:

$$\text{Gate: } \begin{array}{c}
\hline
G \text{ pixels} \\
\hline
\end{array}$$
- Typically, the gate size is varied during tracking.
- We will stipulate minimum and maximum values for \( G \).
- Typically, for a new track, \( G \) starts out at \( G_{\text{max}} \).
- Why do this? (Vary the gate size)
  - For a candidate new track, our knowledge of the true track position is typically poor. So we want a large, or "loose" gate to be sure we pick up all the exceedances associated with the track.
  - For an existing track, if the measurements have not been agreeing well with the predictions, we also want a large gate for the same reasons as above.
  - For an existing track where the measurements and predictions have been in good agreement, however, we want the gate to be small, or "tight".

\( \Rightarrow \) Since only exceedances within the gate will be included in the observed threshold calculation, a tight gate means less noise in the centroid calculation.

5.43
Note: Henceforth, we consider only tracking in the horizontal direction. Vertical tracking is analogous.

Gating and Association: Update of existing tracks:

- For each existing track, we have a predicted position \( \hat{\mathbf{x}}_k = \hat{\mathbf{x}}_{k|k-1} \).

- When frame \( k \) arrives, we apply the spatial filter and threshold(s) to detect exceedances.

- For each track, we place the track gate at the predicted location \( \hat{\mathbf{x}}_k \) and compute the centroid of the exceedances in the gate. This centroid is the observation \( z_k \) for the track.

- The measurement noise arises from the discrete nature of the camera focal plane array detector, thermal noise, and imperfections in the gating and target extraction processing.
Centroid Calculation:

\[ Z_K = \sum_{m,n \in \text{Gate}} m E_{m,n} \]

where

\[ E_{ij} = \begin{cases} Y_{ij}, & Y_{ij} > \text{Threshold} \\ 0, & \text{otherwise} \end{cases} \]

\[ Y_{ij} = \text{spatial filter output.} \]

Note: the standard deviation of the exceedances in the gate is also sometimes used in the calculation of \( G \), the gate size.

- For each existing track, \( i_{K}^{0} \) is used as the observation \( Z_k \) to update the Kalman filter.

New Track Starts:

- In any given frame, there may be exceedances that do not fall near the gate of any existing track (i.e., do not "associate").
- A default sized gate is placed around such exceedances and they become candidates for new tracks.
- Since spurious bright spots may occur due to reflections, sun glints, etc., we don't start a new track right away.
- Instead, we require the candidate track to persist for some small number of frames before we actually start a new track. This is called the "persistance test" or the "temporal correspondence test".

Track Coasting:

- Sometimes a target is temporarily obscured.
- In this case, there will be no exceedances in the track gate of the existing track.
- When this occurs, we need to "coast" the track.

- This involves two steps:
  1. Open the track gate up to the maximum size.
  2. Allow the state equation to evolve with no input.

- If the number of missed detections exceeds a set limit, then the track is terminated. This is called "track loss" or "track deletion."
- It may be helpful to restart the Kalman filter upon reacquisition.
Track Crossings:

- Sometimes two targets will cross one another.
- When this happens, the track gates overlap.
- In this case, it may not be possible to resolve the individual targets.
- The best thing we can do is coast both tracks and hope to reacquire the targets when they separate.
- It may be helpful to restart the Kalman gains upon reacquisition.

Track Merge:

- Sometimes two tracks will cross and never come apart again.
- In this case, we merge the tracks.
- The simplest way to do this is to keep the track that best agrees with the observations and delete the other track.
- This is called a "track merge".
- It should be noted in the history of the deleted track that it was deleted due to a merge. Later offline analysis of both tracks may reveal insight into what actually happened.
Track Filter (Kalman Filter): 

- The horizontal position of the target is \( i_k \).
- We denote the horizontal velocity of the target in frame \( k \) by \( \frac{\partial}{\partial t} i_k = V_k \).
- Since we are assuming a "constant velocity" model from frame to frame, the position (horizontal) of the target in frame \( k+1 \) is given by
  \[
  i_{k+1}^c = i_k^c + \Delta V_k, \quad (\star)
  \]
  \( \Delta = \text{frame time} \).
- Now, it is obvious that, over many frames, the velocity cannot remain truly constant. Thus, we allow a small "velocity drift" from frame to frame:
  \[
  V_{k+1} = V_k + u_k, \quad (\star \star)
  \]
  where \( u_k = \text{white "velocity drift" noise with} \)
  \[
  E[u_k] = 0, \quad E[u_k u_i] = \sigma_v^2 \delta_{k-i}.
  \]
- Since we have used the symbol "\( V_k \)" for the target velocity, we will use "\( \eta_k \)" for the measurement noise (a white noise process), with
  \[
  E[\eta_k] = 0, \quad E[\eta_k \eta_i] = 0, \quad E[\eta_k \eta_i^*] = \sigma_n^2 \delta_{k-i}.
  \]
- The observation is then given by \( Z_k = i_k^c + \eta_k \).

(\star \star \star)
The system model is obtained by writing equations (5.4), (5.5), and (5.4.4) on page 5.48 together in matrix form:

\[
\begin{bmatrix}
    i_{k+1}^c \\
    v_{k+1}
\end{bmatrix} =
\begin{bmatrix}
    1 & A \\
    0 & 1
\end{bmatrix}
\begin{bmatrix}
    i_k^c \\
    v_k
\end{bmatrix} +
\begin{bmatrix}
    0 \\
    1
\end{bmatrix} u_k
\]

\[
Z_k =
\begin{bmatrix}
    1 & 0
\end{bmatrix}
\begin{bmatrix}
    i_k^c \\
    v_k
\end{bmatrix} + n_k
\]

With reference to equations (5.5.1) and (5.5.2) in the book, we have:

\[
\phi_k =
\begin{bmatrix}
    1 & A
\end{bmatrix} \quad \chi_k =
\begin{bmatrix}
    i_k^c \\
    v_k
\end{bmatrix}
\]

\[
\psi_k =
\begin{bmatrix}
    0 \\
    u_k
\end{bmatrix}
\]

\[
\eta_k =
\begin{bmatrix}
    1 & 0
\end{bmatrix}
\]

\[
v_k = n_k
\]

"book"

With reference to equations (5.5.3) and (5.5.4) in the book, we have:

\[
Q_k = E \{[u_k][0 u_i]\} =
\begin{bmatrix}
    0 & 0 \\
    0 & \sigma_i \delta_{i-k}
\end{bmatrix}
\]

\[
R_k = E[n_k n_i] = \sigma_n^2 \delta_{k-i}
\]

5.49
- At \( k = -2 \), we measure \( z_{-2} = i_{-2} + n_{-2} \).
- At \( k = -1 \), we measure \( z_{-1} = i_{-1} + n_{-1} \).

\[ \Rightarrow \text{we take } \hat{i}_{-1} = z_{-1} \]

\[ \text{and } \hat{v}_{-1} = \frac{z_{-1} - z_{-2}}{\Delta} \]

\( \text{note: the averaging reduces the effects of the measurement noise } n_k. \)

This gives us

\[ \begin{bmatrix} \hat{i}_{-1} \\ \hat{v}_{-1} \end{bmatrix} = \begin{bmatrix} z_{-1} \\ z_{-1} - z_{-2} \end{bmatrix} \]

- Now, \( \text{note: NOT } e_{-1}!! \)

\[ P_{-1} = E[e_{-1} e_{-1}^T] = E \left\{ (x_{-1} - \hat{x}_{-1}) (x_{-1} - \hat{x}_{-1})^T \right\} \]

\[ = E \left\{ \left( \begin{bmatrix} i_{-1} \\ v_{-1} \end{bmatrix} - \begin{bmatrix} \hat{i}_{-1} \\ \hat{v}_{-1} \end{bmatrix} \right) \left( \begin{bmatrix} i_{-1} \\ v_{-1} \end{bmatrix} - \begin{bmatrix} \hat{i}_{-1} \\ \hat{v}_{-1} \end{bmatrix} \right)^T \right\} \]

\[ = E \left\{ \begin{bmatrix} i_{-1} - \hat{i}_{-1} \\ v_{-1} - \hat{v}_{-1} \end{bmatrix} \begin{bmatrix} i_{-1} - \hat{i}_{-1} \\ v_{-1} - \hat{v}_{-1} \end{bmatrix}^T \right\} \]

\[ = \begin{bmatrix} i_{-1} - \hat{i}_{-1} \\ v_{-1} - \hat{v}_{-1} \end{bmatrix} \begin{bmatrix} i_{-1} - \hat{i}_{-1} \\ v_{-1} - \hat{v}_{-1} \end{bmatrix}^T \]

\( \text{(*)} \)
Now,
\[
\hat{\xi}_1 - \hat{\xi}_2 = \xi_1 - \xi_2 - \xi_2.
\]

\[
= i\xi_1 - \left( i\xi_1 + n_1 \right)
\]

\[
= -n_1 \quad (\times)
\]

\[
\nu - \hat{\nu}_1 = \nu - \frac{Z_1 - Z_2}{\Delta}
\]

\[
= \nu - \frac{\left( i\xi_1 + n_1 \right) - \left( i\xi_2 + n_2 \right)}{\Delta}
\]

\[
= \nu - \frac{\left( i\xi_1 - i\xi_2 \right) + \left( n_1 - n_2 \right)}{\Delta}
\]

\[
= \nu - \frac{\left( (i\xi_2 + \Delta \nu - i\xi_2) + (n_1 - n_2) \right)}{\Delta}
\]

\[
= \left( \nu + u_2 \right) - \frac{1}{\Delta} \left( \Delta \nu + n_1 - n_2 \right)
\]

\[
= \nu + u_2 - v_2 - \frac{n_1 - n_2}{\Delta}
\]

\[
= u_2 - \frac{n_1 - n_2}{\Delta} \quad (\times \times)
\]
We next use the results on page 19 to compute the quantities needed in equation (4) on page 18:

$$E\left[(\hat{i}_{c1} - \hat{i}_{c1})^2\right] = E\left[(-\hat{n}_{-1})^2\right] = E[n_{21}^2] = \sigma_n^2 \quad (\ast)$$

$$E\left[(\hat{i}_{c1} - \hat{i}_{c1})(\hat{v}_{-1} - \hat{v}_{-1})\right] = E\left[(-n_{-1})(u_{-2} - \frac{n_{-1} - n_{-2}}{\Delta})\right]$$
$$= E\left[-n_{-1}u_{-2} + \frac{n_{-1}}{\Delta}(n_{-1} - n_{-2})\right]$$
$$= E\left[-n_{-1}u_{-2} + \frac{n_{21}^2}{\Delta} - \frac{n_{-1}n_{-2}}{\Delta}\right]$$
$$= -E\left[n_{-1}u_{-2}\right] + \frac{1}{\Delta} E[n_{21}^2] - \frac{1}{\Delta} E[n_{-1}n_{-2}]$$
$$= \frac{\sigma_n^2}{\Delta} \quad (\ast\ast)$$

$$E\left[(\hat{v}_{-1} - \hat{v}_{-1})(\hat{i}_{c1} - \hat{i}_{c1})\right] = E\left[(\hat{i}_{c1} - \hat{i}_{c1})(\hat{v}_{-1} - \hat{v}_{-1})\right] = \frac{\sigma_n^2}{\Delta} \quad (\ast\ast)$$
\[
E \left[ (v_{-1} - \hat{v}_{-1})^2 \right] = E \left[ (u_{-2} - \frac{n_{-1} - n_{-2}}{\Delta})^2 \right] \\
= E \left[ u_{-2}^2 - \frac{2u_{-2}}{\Delta} (n_{-1} - n_{-2}) + \frac{(n_{-1} - n_{-2})^2}{\Delta^2} \right] \\
= E \left[ u_{-2}^2 - \frac{2}{\Delta} (u_{-2} n_{-1} - u_{-2} n_{-2}) + \frac{1}{\Delta^2} (n_{-1}^2 - 2n_{-1} n_{-2} + n_{-2}^2) \right] \\
= E[u_{-2}^2] - \frac{2}{\Delta} E[u_{-2} n_{-1}] + \frac{2}{\Delta} E[u_{-2} n_{-2}] + \frac{1}{\Delta^2} E[n_{-1}^2] \\
- \frac{2}{\Delta^2} E[n_{-1}^2] + \frac{1}{\Delta^2} E[n_{-2}^2] \\
= \sigma_v^2 + \frac{1}{\Delta^2} \sigma_n^2 + \frac{1}{\Delta^2} \sigma_n^2 \\
= \frac{2\sigma_n^2}{\Delta^2} + \sigma_v^2 
\]

\text{So,}

\[
P_{-1} = \begin{bmatrix} 5.50 (*) & 5.52 (**) \\ 5.52 (**) & 5.53 (***) \end{bmatrix} \\
= \begin{bmatrix} \sigma_n^2 & \sigma_n^2 \\ \sigma_n^2 & \frac{2\sigma_n^2}{\Delta^2} + \sigma_v^2 \end{bmatrix} 
\]

\text{5.53}
\[ \hat{x}_0^- = \phi_{-1} \hat{x}_{-1} \quad (5.5.23) \]
\[
= \begin{bmatrix} 1 & \Delta \\ 0 & 1 \end{bmatrix} \begin{bmatrix} \hat{x}_{-1} \\ \hat{v}_{-1} \end{bmatrix}
\]

\[ P_0^- = \phi_{-1} P_{-1} \phi_{-1}^T + Q_{-1} \quad (5.5.25) \]
\[
= \begin{bmatrix} 1 & \Delta \\ 0 & 1 \end{bmatrix} \begin{bmatrix} \sigma_n^2 & \frac{\sigma_n^2}{\Delta} \\ \frac{\sigma_n^2}{\Delta} & \frac{2\sigma_n^2}{\Delta^2} + \sigma_v^2 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ \Delta & 1 \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ 0 & \sigma_v^2 \end{bmatrix}
\]
\[
= \begin{bmatrix} 2\sigma_n^2 & \frac{3\sigma_n^2}{\Delta} + \Delta \sigma_v^2 \\ \frac{\sigma_n^2}{\Delta} & \frac{2\sigma_n^2}{\Delta^2} + \sigma_v^2 \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ 0 & \sigma_v^2 \end{bmatrix}
\]
\[
= \begin{bmatrix} 5\sigma_n^2 + \Delta^2 \sigma_v^2 & \frac{3\sigma_n^2}{\Delta} + \Delta \sigma_v^2 \\ \frac{3\sigma_n^2}{\Delta} + \Delta \sigma_v^2 & \frac{2\sigma_n^2}{\Delta^2} + \sigma_v^2 \end{bmatrix}
\]
\[
= \begin{bmatrix} 5\sigma_n^2 + \Delta^2 \sigma_v^2 & \frac{3\sigma_n^2}{\Delta} + \Delta \sigma_v^2 \\ \frac{3\sigma_n^2}{\Delta} + \Delta \sigma_v^2 & 2(\frac{\sigma_n^2}{\Delta^2} + \sigma_v^2) \end{bmatrix}
\]

\[ \Rightarrow \text{We are now ready to enter the Kalman filtering loop given in Fig. 5.8 on page 219 of the book.} \]
We will now look more carefully at the equations in Fig. 5.8 and see that, in this case (constant velocity target model), they reduce to the well-known $\alpha$-$\beta$ filter:

\[ \hat{x}_k = \hat{x}^-_k + K_k (z_k - H_k \hat{x}^-_k) \quad (5.5.8) \]

\[ \begin{bmatrix} \hat{i}^-_k \\ \hat{v}_k \end{bmatrix} = \begin{bmatrix} \hat{i}^-_k \\ \hat{v}^-_k \end{bmatrix} + K_k (z_k - [1 0] \begin{bmatrix} \hat{i}^-_k \\ \hat{v}^-_k \end{bmatrix}) \]

\[ \begin{bmatrix} \hat{i}^-_k \\ \hat{v}_k \end{bmatrix} = \begin{bmatrix} \hat{i}^-_k \\ \hat{v}^-_k \end{bmatrix} + K_k (z_k - \hat{i}^-_k) \]

\[ \rightarrow \text{Disagreement between observed position and predicted position.} \]

We now write $K_k = \begin{bmatrix} \alpha_k \\ \beta_k \end{bmatrix}$ to obtain scalar equations for the "smoothed" or "updated" estimates of the state vector entries:

\[ \begin{cases} \hat{\xi}_k^- = \hat{\xi}_k^- + \alpha_k (z_k - \hat{\xi}_k^-) \\ \hat{v}_k = \hat{v}^-_k + \beta_k (z_k - \hat{\xi}_k^-) \end{cases} \quad (6) \]

\[ \rightarrow \]

5.55
The "update" or "aging" equation for the predicted state vector is

$$\hat{x}_{k+1} = \Phi_k \hat{x}_k$$  (5.5.23)

For our "constant velocity" model, this gives us

$$\begin{bmatrix}
\hat{\xi}_{k+1}^- \\
\hat{\nu}_{k+1}^-
\end{bmatrix} = \begin{bmatrix} 1 & \Delta \\ 0 & 1 \end{bmatrix} \begin{bmatrix}
\hat{\xi}_k^- \\
\hat{\nu}_k^-
\end{bmatrix} = \begin{bmatrix}
\hat{\xi}_k^- + \Delta \hat{\nu}_k \\
\hat{\nu}_k^-
\end{bmatrix}
$$

Or, as scalar equations,

$$\begin{cases}
\hat{\xi}_{k+1}^- = \hat{\xi}_k^- + \Delta \hat{\nu}_k^- \\
\hat{\nu}_{k+1}^- = \hat{\nu}_k^-
\end{cases}$$  (†)

Combining (†) above with (‡) on p. 5.55 we obtain the classical η-β filter equations:

$$\begin{cases}
\hat{\xi}_K = \hat{\xi}_K^- + \alpha_K (z_K - \hat{\xi}_K^-) \\
\hat{\nu}_K = \hat{\nu}_K^- + \beta_K (z_K - \hat{\xi}_K^-) \\
\hat{\xi}_{k+1}^- = \hat{\xi}_K^- + \Delta \hat{\nu}_K^- \\
\hat{\nu}_{k+1}^- = \hat{\nu}_K^-
\end{cases}$$  (‡ ‡)
Now we must address the computation of the so-called "Kalman gains" $\alpha_k$ and $\beta_k$.

NOTE:

$\rightarrow P_0^-$ on page 5.54 does not depend on the observations $z_k$.

$$\begin{cases}
K_k = P_k^- H_k^T (H_k P_k^- H_k^T + R_k)^{-1} \\
p_k = (I - K_k H_k) P_k^- \\
p_{k+1}^- = \Phi_k P_k \Phi_k^T + Q_k
\end{cases} \quad (5.5.17) \quad (5.5.22) \quad (5.5.25)$$

$\Rightarrow$ The Kalman gains depend on the predicted state vector error covariance matrix, but not directly on the observations.

$\Rightarrow$ The predicted state vector error covariance matrix depends on the filtered state vector error covariance matrix, but not directly on the observations.

$\Rightarrow$ The filtered state vector error covariance matrix depends on the predicted state vector error covariance matrix and on the Kalman gains, but not directly on the observations.
Thus, starting with the expression for \( P_0 \) on page 5.54 and iterating equations (\#) on page 5.57, we see that none of the Kalman gains and state vector error covariance matrices will ever depend on the observations.

This means that we can \textsc{precompute} all the Kalman gains offline and store them in a file.

Implementation of the Kalman filter is then reduced to processing the observations \( z_k \) using equations (\#\#) on page 5.56 subject to the initial conditions

\[
\begin{align*}
\hat{x}_{-1} &= \hat{z}_{-1} \\
\hat{v}_{-1} &= \frac{z_{-1} - z_{-2}}{\Delta} \\
\lambda_{-1} &= \phi_{-1} X_{-1} = \begin{bmatrix} 1 & \Delta \\ 0 & 1 \end{bmatrix} \begin{bmatrix} \hat{x}_{-1} \\ \hat{v}_{-1} \end{bmatrix}
\end{align*}
\]

given on pages 5.50 and 5.54.
Computation of the Gains:

- Let $P_k = \begin{bmatrix} P_{k}(1,1) & P_{k}(1,2) \\ P_{k}(2,1) & P_{k}(2,2) \end{bmatrix}$

$$P_k^- = \begin{bmatrix} P_{k}^{-}(1,1) & P_{k}^{-}(1,2) \\ P_{k}^{-}(2,1) & P_{k}^{-}(2,2) \end{bmatrix}$$

- The $P_k^-$ iteration is initialized with the expression for $P_0^-$ given on page 5.55

- Kalman gains:

$$K_k = P_k^- H_k^T (H_k P_k^- H_k^T + R_k)^{-1} \quad (5.5.17)$$

$$\begin{bmatrix} \alpha_k \\ \beta_k \end{bmatrix} = \begin{bmatrix} P_{k}^{-}(1,1) & P_{k}^{-}(1,2) \\ P_{k}^{-}(2,1) & P_{k}^{-}(2,2) \end{bmatrix} \begin{bmatrix} 1 \\ 0 \end{bmatrix} \left( \begin{bmatrix} P_{k}^{-}(1,1) & P_{k}^{-}(1,2) \\ P_{k}^{-}(2,1) & P_{k}^{-}(2,2) \end{bmatrix} \begin{bmatrix} 1 \\ 0 \end{bmatrix} + \sigma_n^2 \right)^{-1}$$

$$= \begin{bmatrix} P_{k}^{-}(1,1) \\ P_{k}^{-}(2,1) \end{bmatrix} \left( \begin{bmatrix} P_{k}^{-}(1,1) & P_{k}^{-}(1,2) \\ P_{k}^{-}(2,1) & P_{k}^{-}(2,2) \end{bmatrix} \begin{bmatrix} 1 \\ 0 \end{bmatrix} + \sigma_n^2 \right)^{-1}$$

$$= \begin{bmatrix} P_{k}^{-}(1,1) \\ P_{k}^{-}(2,1) \end{bmatrix} \left( P_{k}^{-}(1,1) + \sigma_n^2 \right)^{-1}$$

$$= \begin{bmatrix} \frac{P_{k}^{-}(1,1)}{P_{k}^{-}(1,1) + \sigma_n^2} \\ \frac{P_{k}^{-}(2,1)}{P_{k}^{-}(2,1) + \sigma_n^2} \end{bmatrix} \quad \rightarrow$$

5-59
\[ \alpha_k = \frac{P_{k-1}^{(1,1)}}{P_{k-1}^{(1,1)} + \sigma_n^2} \quad (\star) \]

\[ \beta_k = \frac{P_{k-1}^{(2,1)}}{P_{k-1}^{(1,1)} + \sigma_n^2} \]

- Filtered State Vector Error Covariance Matrix:

\[ P_k = (I - K_k H_k) P_{k-1}^e \quad (5.5.22) \]

\[
\begin{bmatrix}
P_{k-1}^{(1,1)} & P_{k-1}^{(1,2)} \\
P_{k-1}^{(2,1)} & P_{k-1}^{(2,2)}
\end{bmatrix}
= \left( \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} - \begin{bmatrix} \alpha_k \\ \beta_k \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \right)
\begin{bmatrix}
P_{k-1}^{(1,1)} & P_{k-1}^{(1,2)} \\
P_{k-1}^{(2,1)} & P_{k-1}^{(2,2)}
\end{bmatrix}
\]

\[ = \left( \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} - \begin{bmatrix} \alpha_k & 0 \\ \beta_k & 0 \end{bmatrix} \right)
\begin{bmatrix}
P_{k-1}^{(1,1)} & P_{k-1}^{(1,2)} \\
P_{k-1}^{(2,1)} & P_{k-1}^{(2,2)}
\end{bmatrix}
\]

\[ = \begin{bmatrix}
1 - \alpha_k & 0 \\
-\beta_k & 1
\end{bmatrix}
\begin{bmatrix}
P_{k-1}^{(1,1)} & P_{k-1}^{(1,2)} \\
P_{k-1}^{(2,1)} & P_{k-1}^{(2,2)}
\end{bmatrix}
\]

\[ = \begin{bmatrix}
(1-\alpha_k) P_{k-1}^{(1,1)} & (1-\alpha_k) P_{k-1}^{(1,2)} \\
(1-\beta_k) P_{k-1}^{(2,1)} & (1-\beta_k) P_{k-1}^{(2,2)}
\end{bmatrix} \quad (\star\star) \]
Predicted State Vector Error Covariance Matrix:

\[
P_{k+1}^- = \Phi_k P_k \Phi_k^T + Q_k \quad (5.5.25)
\]

\[
\begin{bmatrix}
P_{k+1}^- (1,1) & P_{k+1}^- (1,2) \\
P_{k+1}^- (2,1) & P_{k+1}^- (2,2)
\end{bmatrix}
= \begin{bmatrix}
1 & \Delta \\
0 & 1
\end{bmatrix}
\begin{bmatrix}
P_k (1,1) & P_k (1,2) \\
P_k (2,1) & P_k (2,2)
\end{bmatrix}
\begin{bmatrix}
1 & 0 \\
\Delta & 1
\end{bmatrix} + \begin{bmatrix}
0 & 0 \\
0 & 0\sigma_v^2
\end{bmatrix}
\]

\[
= \begin{bmatrix}
P_k (1,1) + \Delta P_k (2,1) & P_k (1,2) + \Delta P_k (2,2) \\
P_k (2,1) & P_k (2,2)
\end{bmatrix}
\begin{bmatrix}
1 & 0 \\
\Delta & 1
\end{bmatrix} + \begin{bmatrix}
0 & 0 \\
0 & 0\sigma_v^2
\end{bmatrix}
\]

\[
\begin{cases}
P_{k+1}^- (1,1) = P_k (1,1) + \Delta P_k (2,1) + \Delta^2 P_k (2,2) = P_k (1,1) + 2\Delta P_k (1,2) \\
P_{k+1}^- (1,2) = P_k (1,2) + \Delta^2 P_k (2,2)
\end{cases}
\]

\[
\begin{cases}
P_{k+1}^- (2,1) = P_k (2,1) + \Delta P_k (2,2) = P_k (1,2) + \Delta P_k (2,2) \\
P_{k+1}^- (2,2) = P_k (2,2) + \sigma_v^2
\end{cases}
\]

\underline{Note:} For covariance matrices, the (2,1) element and (1,2) element must be equal.
Thus, to precompute the gains:

1. Begin with $P_0$ as given on page 5.54
2. $k = 0$

3. Compute $\alpha_k$ and $\beta_k$ using (4) on page 5.60
4. Compute $P_k$ using (4.1) on page 5.60
5. Compute $P_{k+1}$ using (4) on page 5.61
6. $k = k + 1$

- Typically, the Kalman gains will rapidly converge to asymptotic values.
- The iteration above need only be carried out until convergence occurs.
Asymptotic Behavior

- We would like to know for sure when the Kalman gains have converged.
- Thus, it is of interest to compute the asymptotic values of the gains.

- Eq. (4) on 5.60 can be substituted into (**) on 5.60 to obtain $P_k$ in terms of $P_{k-1}$.

- The above result can be substituted into (4) on 5.61 to get $P_{k+1}$ in terms of $P_k$.

- In the limit as $k \to \infty$, $P_{k+1} = P_k = P_\infty$. Plugging this into the above gives three equations in three unknowns: $P_\infty(1,1)$, $P_\infty(2,2)$, and $P_\infty(1,2) = P_\infty(2,1)$.

- These can be solved. The solution can then be plugged into (4) on 5.60 to solve for the asymptotic gains, $\alpha_0$ and $\beta_0$. 5.63
Nonwhite Forcing Function

- This example is from Section 5.7 of the book.

- Shipboard INS; we wish to track INS drift.

\[ \psi_x = \text{longitude position error (E-w)} \]
\[ \psi_y = \text{latitude position error (N-S)} \]
\[ \psi_z = \text{azimuthal error} - \psi_x \tan(\text{latitude}) \]
\[ \Omega_x = \Omega \cos(\text{latitude}) \]
\[ \Omega_z = \Omega \sin(\text{latitude}) \]

\( x = \text{North axis} \)
\( y = \text{West axis} \)
\( z = \text{up axis} \)

\( \varepsilon_x, \varepsilon_y, \varepsilon_z = \text{INS gyro drift rates.} \)
The error dynamics are given by

\[ \begin{align*}
\dot{y}_x &= -\Omega_z y_y + \varepsilon_x \\
\dot{y}_y &= -\Omega_z y_x + \Omega_x y_z + \varepsilon_y \\
\dot{y}_z &= -\Omega_x y_y + \varepsilon_z
\end{align*} \]

- For state variables, we take \( y_x, y_y, y_z \).
- We could write the state model

\[
\begin{bmatrix}
\dot{y}_x \\
\dot{y}_y \\
\dot{y}_z
\end{bmatrix} =
\begin{bmatrix}
0 & -\Omega_z & 0 \\
-\Omega_z & 0 & \Omega_x \\
0 & -\Omega_x & 0
\end{bmatrix}
\begin{bmatrix}
y_x \\
y_y \\
y_z
\end{bmatrix} +
\begin{bmatrix}
\varepsilon_x \\
\varepsilon_y \\
\varepsilon_z
\end{bmatrix}
\]

- The problem with this is: the drift rates \( \varepsilon_x, \varepsilon_y, \varepsilon_z \) change very slowly in time.

\( \Rightarrow \) They are each highly self-correlated over short time intervals.

\( \Rightarrow \) So the above model doesn't satisfy the "uncorrelated forcing function" requirement of the Kalman filter.
- But, $\ddot{x}$, $\ddot{y}$, $\ddot{z}$ are expected to be uncorrelated,

$$\ddot{x} = f_x$$
$$\ddot{y} = f_y$$
$$\ddot{z} = f_z$$

- We get around the problem by adding $\ddot{x}$, $\ddot{y}$, $\ddot{z}$ to the state vector. This is called "Augmenting the state vector".

$$\begin{bmatrix}
\dot{\psi}_x \\
\dot{\psi}_y \\
\dot{\psi}_z \\
\dot{\psi}_x \\
\dot{\psi}_y \\
\dot{\psi}_z
\end{bmatrix} =
\begin{bmatrix}
0 & \Omega_z & 0 & 1 & 0 & 0 \\
-\Omega_z & 0 & \Omega_y & 0 & 1 & 0 \\
0 & -\Omega_x & 0 & 0 & 0 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0
\end{bmatrix}
\begin{bmatrix}
\psi_x \\
\psi_y \\
\psi_z \\
\dot{\psi}_x \\
\dot{\psi}_y \\
\dot{\psi}_z
\end{bmatrix}
+ \begin{bmatrix}
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0
\end{bmatrix}
\begin{bmatrix}
f_x \\
f_y \\
f_z \\
f_x \\
f_y \\
f_z
\end{bmatrix}$$
- Generally, for a nonwhite forcing function, we can find the "shaping filter" that relates the nonwhite function to white noise.

- We can then use white noise as the system input, incorporate the nonwhite forcing function into the state vector, and add the shaping filter equations to the system equations.

- A similar procedure can be used for nonwhite measurement noise.

  1. Remove nonwhite $v_k$ entries from $z_k = H_k x_k + v_k$

  2. Put the removed entries into the state vector.

  3. Add white noise entries to $w_k$ and add the required shaping filter equations to the system equations.
- Section 5.8 of the book starts to hint around the actual original Kalman filter solution a little bit.

- While it is not said in the book, this is again related to the innovations process and the orthogonality principle.

⇒ Chap. 5 up to here in a nutshell:

1. Usual Kalman filter problem: statement
2. Assume the a priori prediction \( \hat{x}_0 \) is given.
3. Assume that \( \hat{x}_k \) is linearly related to \( \hat{x}_k^e \) and \( z_k \). Find MMSE gains.

⇒ This is a quite restricted notion of the Kalman filter. In fact, the Kalman filter is much more powerful and much more profound than the above implies.
Here is the development given in the book with some added clarifications:

- Assume the standard Kalman problem statement and assume further that the input noise \( w_k \) and measurement noise \( v_k \) are Gaussian.

- Let \( z_k^* \) be the vector of all observations from time 0 up until time \( k \):

\[
    z_k^* = [z_0, z_1, \ldots, z_k]^T
\]

- Keeping all the notation unchanged from earlier in the chapter, we have the true state vector \( x_k \) and the filtered estimate \( \hat{x}_k \) (as yet unspecified).

- **Note:** \( \hat{x}_k \) doesn't depend on the true \( x_k \).

  \( \hat{x}_k \) depends only on the observations \( z_k^* \).

  → So, conditioned on \( z_k^* \), \( \hat{x}_k \) is deterministic, not stochastic.

- The MSE of \( \hat{x}_k \) conditioned on \( z_k^* \) is

\[
    E[(x_k - \hat{x}_k)(x_k - \hat{x}_k) \mid z_k^*] = E[x_kx_k - x_k^T \hat{x}_k + \hat{x}_k^T x_k + \hat{x}_k^T \hat{x}_k \mid z_k^*]
\]

→
\[
= E[\mathbf{x}_k \mathbf{x}_k | \mathbf{z}_k^*] - E[\mathbf{x}_k | \mathbf{z}_k^*] \hat{x}_k - \hat{x}_k^T E[\mathbf{x}_k | \mathbf{z}_k^*] \\
+ \hat{x}_k^T \hat{x}_k \quad (5.8.1)
\]

(Complete the square)

\[
= E[\mathbf{x}_k \mathbf{x}_k | \mathbf{z}_k^*] + \left\{ \hat{x}_k - E[\mathbf{x}_k | \mathbf{z}_k^*] \right\}^T \left\{ \hat{x}_k - E[\mathbf{x}_k | \mathbf{z}_k^*] \right\} \\
- E[\mathbf{x}_k^T | \mathbf{z}_k^*] E[\mathbf{x}_k | \mathbf{z}_k^*] \quad (5.8.2)
\]

⇒ Only the middle term depends on \( \hat{x}_k \).
⇒ In designing an optimal \( \hat{x}_k \), we can only hope to impact the middle term.
⇒ The middle term is minimized by choosing

\[
\hat{x}_k = E[\mathbf{x}_k | \mathbf{z}_k^*], \quad (5.8.3)
\]

the conditional mean of \( \mathbf{x}_k \) given \( \mathbf{z}_k^* \).
Let $x_{kl|k-1} = x_k$ "given" $z_{k-1}^*$ = $x_k$ conditioned on $z_{k-1}^*$.

Let $P_{kl|k-1} = E\left[(x_{kl|k-1} - \hat{x}_k^-)(x_{kl|k-1} - \hat{x}_k^-)^T\right]$.

Then $x_{kl|k-1} \sim N(\hat{x}_k^-, P_{kl|k-1})$ (5.8.4) "is distributed"

Since $z_k = H_k x_k + v_k$ (5.8.5) = (5.5.2), we have $z_k \sim N(H_k \hat{x}_k^-, H_k P_{kl|k-1} H_k^T + R_k)$

Also, it is clear that

$$P(z_k | x_{kl|k-1}) \sim N(H_k x_{kl|k-1}, R_k)$$ (5.8.7)

→ b/c once $x_{kl|k-1}$ is known, the error covariance matrix $P_k^-$ contains all zeros.
Applying Bayes rule,

\[ p(x_{k|k-1} | z_k) = \frac{p(z_k | x_{k|k-1}) p(x_{k|k-1})}{p(z_k)} \]  

(5.8.8).

Analytical expressions for each quantity on the RHS of (5.8.8) were given on page 5.70.

Now, it may be seen that the conditional RV \( x_{k|k-1} | z_k \) in (5.8.8) is identical to the optimal estimate

\[ \hat{x}_k = \mathbb{E}[x_k | z_k^*] \]  

(5.8.3) on page 5.69.

The density distribution of the optimal estimate \( \hat{x}_k \) may be computed using (5.5.8) above.
There results:

\[ E[\hat{x}_k] = \hat{x}_k + P_k^{-1} H_k^T (H_k P_k^{-1} H_k^T + R_k)^{-1} (z_k - H_k x_k) \]

\[ (5.8.10) = (5.5.8) + (5.5.17) \]

\[ \text{Cov}(\hat{x}_k) = \left[ (P_k^{-1})^{-1} + H_k^T R_k^{-1} H_k \right]^{-1} \]

\[ (5.8.1) \]

\[ \text{nontrivial} \]

\[ (I - K_k H_k) P_k^{-1} \]

\[ (5.5.22). \]

\[ \Rightarrow \text{This is the same answer we got before in Section 5.5.} \]

\[ \text{But here, we did not assume that the estimator had to be linear.} \]

\[ \Rightarrow \text{For Gaussian statistics on } w_k \text{ and } v_k, \]

\[ \text{the Kalman filter is the } \boxed{\text{optimal estimator}} \]

\[ \text{period}. \]

\[ \Rightarrow \text{There is no better estimator, including all possible linear and nonlinear estimators.} \]
Moreover, for Gaussian statistics,
\[ \hat{x}_k = \mathbb{E}[x_k | z^*_k] \]
is also the maximum likelihood estimator.

- It occurs at the mode (maximum) of the distribution (which is unimodal).

Therefore, \( \hat{x}_k \) also minimizes the expected value of practically any nondecreasing function of \( x_k - \hat{x}_k \).

\[ \Rightarrow \] Thus, under Gaussian statistics, the Kalman filter is optimal

- In the MMSE sense
- In the MMAE sense

etc.