Observer design

Observability matrix:

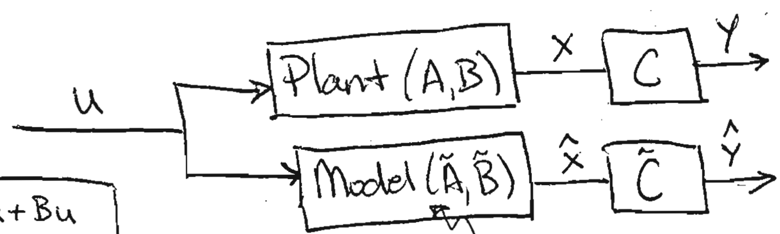
(A is $n \times n$.)

$$\mathcal{O} \equiv \begin{bmatrix} C \\ CA \\ CA^2 \\ \vdots \\ CA^{n-1} \end{bmatrix}$$

If states we don't measure directly can be __estimated__ with an "observer" (also called an "estimator"), $\text{rank}(\mathcal{O}) = n$.

in MATLAB: O = obsv(A, C)

Basic Idea:



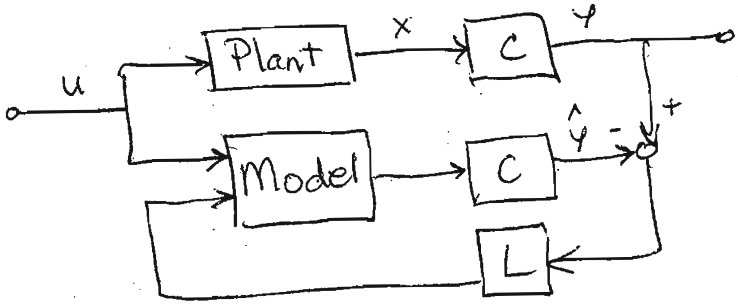
__Plant__: $\dot{x} = Ax + Bu$
$y = Cx$

__Model__: $\dot{\hat{x}} = A\hat{x} + Bu$
$\hat{y} = C\hat{x}$

model keeps track of state estimates, $\hat{x}$, over time...

Imagine we have a good idea of the actual plant dynamics. We can run a __simulation__ of what we __expect__ the plant to do, $\hat{x}$, along with corresponding, expected output, $\hat{y}$.

Now, if y (actual) & $\hat{y}$ (expect) do not agree, we can use <u>feedback</u> to correct our model's estimate of $\hat{x}$:



Now, model updates states as:

$$\boxed{\begin{aligned} \dot{\hat{x}} &= A\hat{x} + Bu + L(y - \hat{y}) \\ \hat{y} &= C\hat{x} \end{aligned}}$$

$$\begin{aligned} \dot{\hat{x}} &= A\hat{x} + Bu + L(y - C\hat{x}) \\ &= (A - LC)\hat{x} + Bu + Ly \end{aligned}$$

$$\boxed{\begin{aligned} e_x &= x - \hat{x} \\ \dot{e}_x &= \dot{x} - \dot{\hat{x}} \end{aligned}}$$ ← error in estimate...

Show that $\boxed{\dot{e}_x = (A - LC)e_x}$.

$$\dot{\hat{x}} = (A-LC)(x-e_x) + Bu + Ly = \dot{x} - \dot{e}_x$$

$$\dot{e}_x = (A-LC)e_x - (A-LC)x - Bu - Ly + \dot{x}$$

$$\dot{x} = Ax + Bu$$

$$\dot{e}_x = (A-LC)e_x - Ax + LCx - Bu - Ly + Ax + Bu$$

$$\dot{e}_x = (A-LC)e_x + LCx - Ly$$

$$\dot{e}_x = (A-LC)e_x + L(Cx - y)$$

$$y = Cx$$

$$\boxed{\dot{e}_x = (A-LC)e_x}$$ ← Dynamics of the error.

→ Set poles of $[sI - (A-LC)]$ to set "speed" of observer.

→ Rule of thumb, set poles to be 2-6x <u>faster</u> than "dominant poles" of the plant.

Example, pendulum: $\left(\begin{array}{c} J\ddot{\theta} + k\theta = \tau \\ \ddot{\theta} + \omega_n^2\theta = \tau \end{array} \right)$

$$\dot{x} = \begin{bmatrix} 0 & 1 \\ -\omega_n^2 & 0 \end{bmatrix} x + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u$$

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

Find $L = \begin{bmatrix} l_1 \\ l_2 \end{bmatrix}$

$$\det\left[sI - (A - LC)\right] \leftarrow \text{poles of estimator dynamics}$$

$$\begin{bmatrix} s & 0 \\ 0 & s \end{bmatrix} - \begin{bmatrix} 0 - l_1 \, , & 1 - 0 \\ -\omega_n^2 - l_2 \, , & 0 \end{bmatrix}$$

$$= \begin{bmatrix} s + l_1 \, , & -1 \\ \omega_n^2 + l_2 \, , & s \end{bmatrix}$$

$$\det \rightarrow s(s + l_1) + (\omega_n^2 + l_2)$$

$$\boxed{s^2 + s l_1 + \omega_n^2 + l_2} \leftarrow \text{char. eq.} \rightarrow = 0$$

Now, say we want: $s_1 = -10\omega_n$, $s_2 = -10\omega_n$

$$(s + 10\omega_n)(s + 10\omega_n) = 0$$

$$\boxed{s^2 + 20\omega_n s + 100\omega_n^2} = 0$$

$$\therefore \boxed{\begin{aligned} l_1 &= 20\omega_n \\ l_2 &= 99\omega_n^2 \end{aligned}} \qquad L = \begin{bmatrix} l_1 \\ l_2 \end{bmatrix}$$

OR: $L = \text{acker}(A', C', pe)'$

where: $pe = \begin{bmatrix} -\omega_n^2 \\ -\omega_n^2 \end{bmatrix}$

Kalman filters

Given a process model: $x_k = Ax_{k-1} + Bu_{k-1} + w$ ← process noise

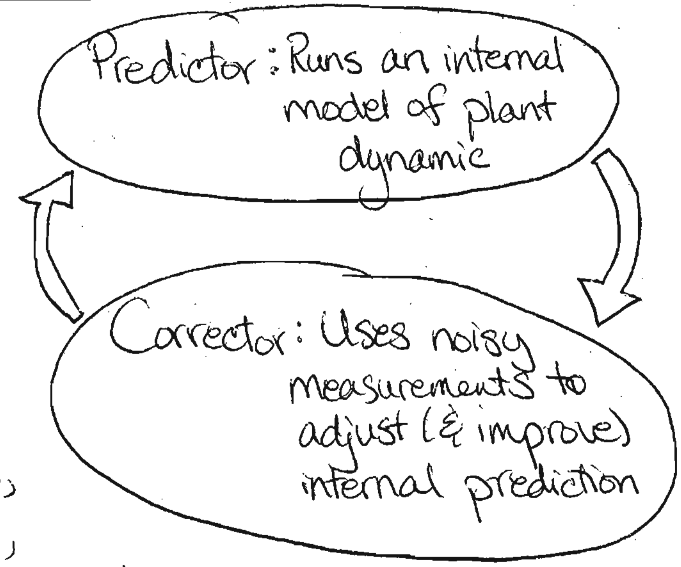
And a measurement model: $z_k = Hx_{k-1} + v$ ← measurement noise

Iterate to find a least-squares estimate of the state, $\hat{x}$.

* Assumptions: w and v are both Gaussian.

Overview:

- Kalman filtering is an iterative loop with 5 basic steps.



- To visualize the updates, consider a simple case, where x has one state and $H=1$.

Algorithm: Predictor ("time update") ← Blindly simulates dynamics

(1) $\hat{x}_k^- = A\hat{x}_{k-1} + Bu_{k-1}$

(2) $P_k^- = AP_{k-1}A^T + Q$ ← P: error covariance matrix of state estimates: $(x-\hat{x})(x-\hat{x})^T$

Corrector ("measurement update") ← Use sensor data to tweak prediction.

(3) $K_k = P_k^- H^T (HP_k^- H^T + R)^{-1}$

"Kalman gain"; when $H=1$, $K_k = \left(\frac{P_k^-}{P_k^- + R}\right)$

(4) $\hat{x}_k = \hat{x}_k^- + K_k(z - H\hat{x}_k^-)$

(5) $P_k = P_k^- - K_k H P_k^-$