16.323 Lecture 11

Estimators/Observers

- Bryson
- Gelb – Optimal Estimation

Estimators/Observers

- **Problem:** So far we have assumed that we have full access to the state $\mathbf{x}(t)$ when we designed our controllers.
  - Most often all of this information is not available.
  - And certainly there is usually error in our knowledge of $\mathbf{x}$.

- Usually can only feedback information that is developed from the sensors measurements.
  - Could try "output feedback" $\mathbf{u} = K\mathbf{x} \quad \Rightarrow \quad \mathbf{u} = \hat{K}\mathbf{y}$
  - But this is type of controller is hard to design.

- **Alternative approach:** Develop a replica of the dynamic system that provides an "estimate" of the system states based on the measured output of the system.

- **New plan:** called a "separation principle"
  1. Develop estimate of $\mathbf{x}(t)$, called $\hat{\mathbf{x}}(t)$.
  2. Then switch from $\mathbf{u} = -K\mathbf{x}(t)$ to $\mathbf{u} = -K\hat{\mathbf{x}}(t)$.

- Two key questions:
  - How do we find $\hat{\mathbf{x}}(t)$?
  - Will this new plan work? (yes, and very well)

Estimation Schemes

- Assume that the system model is of the form:

$$\begin{aligned}\dot{\mathbf{x}} &= A\mathbf{x} + B\mathbf{u} \;, \quad \mathbf{x}(0) \text{ unknown} \\ \mathbf{y} &= C_y\mathbf{x}\end{aligned}$$

  where

  – A, B, and C_y are known – possibly time-varying, but that is suppressed here.

  – $\mathbf{u}(t)$ is known

  – Measurable outputs are $\mathbf{y}(t)$ from $C_y \neq I$

- **Goal:** Develop a dynamic system whose state

$$\hat{\mathbf{x}}(t) = \mathbf{x}(t)$$

  for all time $t \geq 0$. Two primary approaches:

  – Open-loop.

  – Closed-loop.

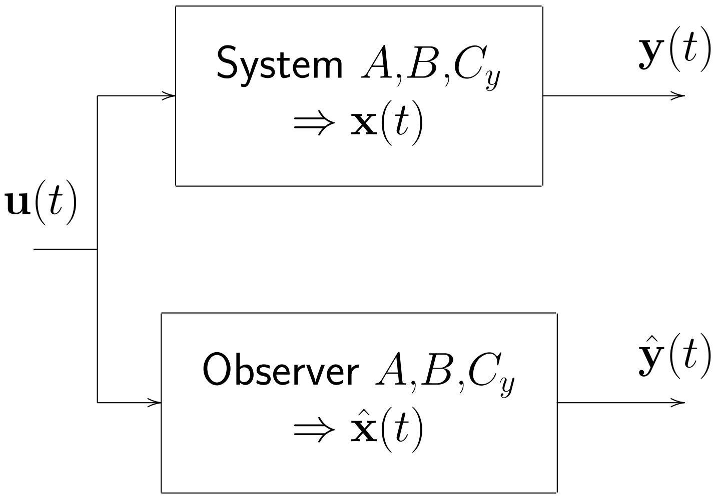
Open-loop Estimator

- Given that we know the plant matrices and the inputs, we can just perform a simulation that runs in parallel with the system

$$\dot{\hat{\mathbf{x}}}(t) = A\hat{\mathbf{x}} + B\mathbf{u}(t)$$

  – Then $\hat{\mathbf{x}}(t) \equiv \mathbf{x}(t) \ \forall \ t$ provided that $\hat{\mathbf{x}}(0) = \mathbf{x}(0)$

- **Major Problem:** We do not know $\mathbf{x}(0)$



- To analyze this case, start with:

$$\begin{aligned}\dot{\mathbf{x}}(t) &= A\mathbf{x}(t) + B\mathbf{u}(t)\\ \dot{\hat{\mathbf{x}}}(t) &= A\hat{\mathbf{x}}(t) + B\mathbf{u}(t)\end{aligned}$$

- Define the **estimation error**: $\tilde{\mathbf{x}}(t) = \mathbf{x}(t) - \hat{\mathbf{x}}(t)$.
  – Now want $\tilde{\mathbf{x}}(t) = 0 \ \forall \ t$, but is this realistic?

- Subtract to get:

$$\frac{d}{dt}(\mathbf{x} - \hat{\mathbf{x}}) = A(\mathbf{x} - \hat{\mathbf{x}}) \quad \Rightarrow \quad \dot{\tilde{\mathbf{x}}}(t) = A\tilde{\mathbf{x}}$$

which has the solution

$$\tilde{\mathbf{x}}(t) = e^{At}\tilde{\mathbf{x}}(0)$$

– Gives the estimation error in terms of the initial error.

- Does this guarantee that $\tilde{\mathbf{x}} = 0 \;\forall\; t$?
Or even that $\tilde{\mathbf{x}} \rightarrow 0$ as $t \rightarrow \infty$? (which is a more realistic goal).

– Response is fine if $\tilde{\mathbf{x}}(0) = 0$. But what if $\tilde{\mathbf{x}}(0) \neq 0$?

- If A stable, then $\tilde{x} \rightarrow 0$ as $t \rightarrow \infty$, but the dynamics of the estimation error are completely determined by the open-loop dynamics of the system (eigenvalues of A).
  – Could be very slow.
  – No obvious way to modify the estimation error dynamics.

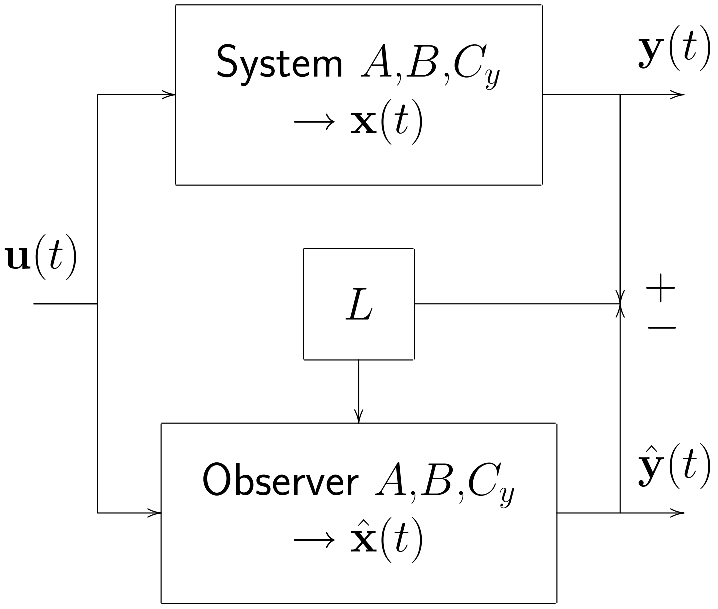
- Open-loop estimation **does not seem to be a very good idea.**

Closed-loop Estimator

- Obvious fix to problem: use the additional information available:
  – How well does the estimated output match the measured output?

  Compare: $\mathbf{y} = C_y \mathbf{x}$ with $\hat{\mathbf{y}} = C_y \hat{\mathbf{x}}$

  – Then form $\tilde{\mathbf{y}} = \mathbf{y} - \hat{\mathbf{y}} \equiv C_y \tilde{\mathbf{x}}$



- **Approach:** Feedback $\tilde{y}$ to improve our estimate of the state. Basic form of the estimator is:

$$
\begin{aligned}
\dot{\hat{\mathbf{x}}}(t) &= A\hat{\mathbf{x}}(t) + B\mathbf{u}(t) + \boxed{L\tilde{\mathbf{y}}(t)} \\
\hat{\mathbf{y}}(t) &= C_y \hat{\mathbf{x}}(t)
\end{aligned}
$$

  where L is a **user selectable gain matrix.**

- **Analysis:**

$$
\begin{aligned}
\dot{\tilde{\mathbf{x}}} = \dot{\mathbf{x}} - \dot{\hat{\mathbf{x}}} &= [A\mathbf{x} + B\mathbf{u}] - [A\hat{\mathbf{x}} + B\mathbf{u} + L(\mathbf{y} - \hat{\mathbf{y}})] \\
&= A(\mathbf{x} - \hat{\mathbf{x}}) - L(C\mathbf{x} - C_y \hat{\mathbf{x}}) \\
&= A\tilde{\mathbf{x}} - LC_y \tilde{\mathbf{x}} = (A - LC_y)\tilde{\mathbf{x}}
\end{aligned}
$$

- So the closed-loop estimation error dynamics are now

$$\dot{\tilde{\mathbf{x}}} = (A - LC_y)\tilde{\mathbf{x}} \quad \text{with solution} \quad \tilde{\mathbf{x}}(t) = e^{(A-LC_y)t}\,\tilde{\mathbf{x}}(0)$$

- **Bottom line:** Can select the gain L to attempt to improve the convergence of the estimation error (and/or speed it up).
  – But now must worry about observability of the system model.

- Note the similarity:
  – **Regulator Problem:** pick K for $A - BK$
  ◇ Choose $K \in \mathcal{R}^{1\times n}$ (SISO) such that the closed-loop poles

$$\det(sI - A + BK) = \Phi_c(s)$$

  are in the desired locations.
  – **Estimator Problem:** pick L for $A - LC_y$
  ◇ Choose $L \in \mathcal{R}^{n\times 1}$ (SISO) such that the closed-loop poles

$$\det(sI - A + LC_y) = \Phi_o(s)$$

  are in the desired locations.

  – These problems are obviously very similar – in fact they are called **dual problems**.

Estimation Gain Design

- For regulation, concerned with controllability of $[A, B]$, $\Rightarrow$ **For controllable system, can place eigenvalues of $A - BK$ arbitrarily**.

- For estimation, concerned with observability of $[A, C_y]$, $\Rightarrow$ **For observable system, can place eigenvalues of $A - LC_y$ arbitrarily.**

- Test using the observability matrix (SISO case):

$$\texttt{rank}\ \mathcal{M}_o \triangleq \texttt{rank} \begin{bmatrix} C_y \\ C_y A \\ C_y A^2 \\ \vdots \\ C_y A^{n-1} \end{bmatrix} = n$$

- Procedure for selecting L similar to that used for regulator design.
  – Note: poles of $(A - LC_y)$ and $(A - LC_y)^T$ are identical.
  – Also have that $(A - LC_y)^T = A^T - C_y^T L^T$
  – So designing L^T for this transposed system looks like a standard regulator problem $(A - BK)$ where

$$\begin{array}{ccc} A & \Rightarrow & A^T \\ B & \Rightarrow & C_y^T \\ K & \Rightarrow & L^T \end{array}$$

  So we can use $K_e = \texttt{acker}(A^T, C_y^T, P)\ , \quad L \equiv K_e^T$

- Note that the estimator equivalent of Ackermann's formula is that

$$L = \Phi_e(s) \mathcal{M}_o^{-1} \begin{bmatrix} 0 & \cdots & 0 & 1 \end{bmatrix}^T$$

Estimator Example 11–1

- Simple system (see page 11-21)

$$A = \begin{bmatrix} -1 & 1.5 \\ 1 & -2 \end{bmatrix}, \quad B = \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \quad x(0) = \begin{bmatrix} -0.5 \\ -1 \end{bmatrix}$$
$$C_y = \begin{bmatrix} 1 & 0 \end{bmatrix}, \quad D = 0$$

  – Assume that the initial conditions are not well known.
  – System stable, but $\lambda_{\max}(A) = -0.18$
  – Test observability:

$$\texttt{rank}\begin{bmatrix} C_y \\ C_yA \end{bmatrix} = \texttt{rank}\begin{bmatrix} 1 & 0 \\ -1 & 1.5 \end{bmatrix}$$

- Use open and closed-loop estimators. Since the initial conditions are not well known, use $\hat{x}(0) = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$

- Open-loop estimator:

$$\begin{aligned} \dot{\hat{\mathbf{x}}} &= A\hat{\mathbf{x}} + B\mathbf{u} \\ \hat{\mathbf{y}} &= C_y\hat{\mathbf{x}} \end{aligned}$$

- Closed-loop estimator:

$$\begin{aligned} \dot{\hat{\mathbf{x}}} &= A\hat{\mathbf{x}} + B\mathbf{u} + L\tilde{\mathbf{y}} = A\hat{\mathbf{x}} + B\mathbf{u} + L(\mathbf{y} - \hat{\mathbf{y}}) \\ &= (A - LC_y)\hat{\mathbf{x}} + B\mathbf{u} + L\mathbf{y} \\ \hat{\mathbf{y}} &= C_y\hat{\mathbf{x}} \end{aligned}$$

  – Dynamic system with poles $\lambda_i(A - LC_y)$ that takes the measured plant outputs as an input and generates an estimate of $\mathbf{x}$.

- Typically simulate both systems together for simplicity

- Open-loop case:

$$\begin{aligned}\dot{\mathbf{x}} &= A\mathbf{x} + B\mathbf{u}\\ \mathbf{y} &= C_y\mathbf{x}\\ \dot{\hat{\mathbf{x}}} &= A\hat{\mathbf{x}} + B\mathbf{u}\\ \hat{\mathbf{y}} &= C_y\hat{\mathbf{x}}\end{aligned}$$

$$\Rightarrow \begin{bmatrix}\dot{\mathbf{x}}\\ \dot{\hat{\mathbf{x}}}\end{bmatrix} = \begin{bmatrix}A & 0\\ 0 & A\end{bmatrix}\begin{bmatrix}\mathbf{x}\\ \hat{\mathbf{x}}\end{bmatrix} + \begin{bmatrix}B\\ B\end{bmatrix}\mathbf{u}\ , \quad \begin{bmatrix}\mathbf{x}(0)\\ \hat{\mathbf{x}}(0)\end{bmatrix} = \begin{bmatrix}-0.5\\ -1\\ 0\\ 0\end{bmatrix}$$

$$\begin{bmatrix}\mathbf{y}\\ \hat{\mathbf{y}}\end{bmatrix} = \begin{bmatrix}C_y & 0\\ 0 & C_y\end{bmatrix}\begin{bmatrix}\mathbf{x}\\ \hat{\mathbf{x}}\end{bmatrix}$$

- Closed-loop case:

$$\begin{aligned}\dot{\mathbf{x}} &= A\mathbf{x} + B\mathbf{u}\\ \dot{\hat{\mathbf{x}}} &= (A - LC_y)\hat{\mathbf{x}} + B\mathbf{u} + LC_y\mathbf{x}\end{aligned}$$

$$\Rightarrow \begin{bmatrix}\dot{\mathbf{x}}\\ \dot{\hat{\mathbf{x}}}\end{bmatrix} = \begin{bmatrix}A & 0\\ LC_y & A - LC_y\end{bmatrix}\begin{bmatrix}\mathbf{x}\\ \hat{\mathbf{x}}\end{bmatrix} + \begin{bmatrix}B\\ B\end{bmatrix}\mathbf{u}$$

- Example uses a strong $\mathbf{u}(t)$ to shake things up

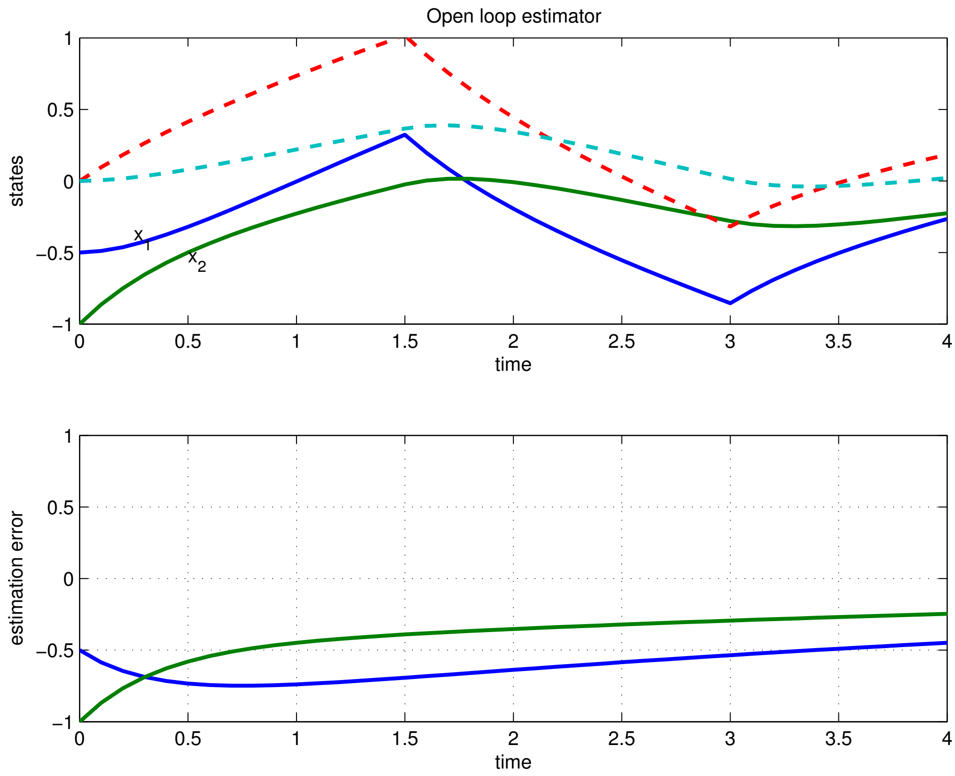


Figure 1: Open-loop estimator. Estimation error converges to zero, but very slowly.

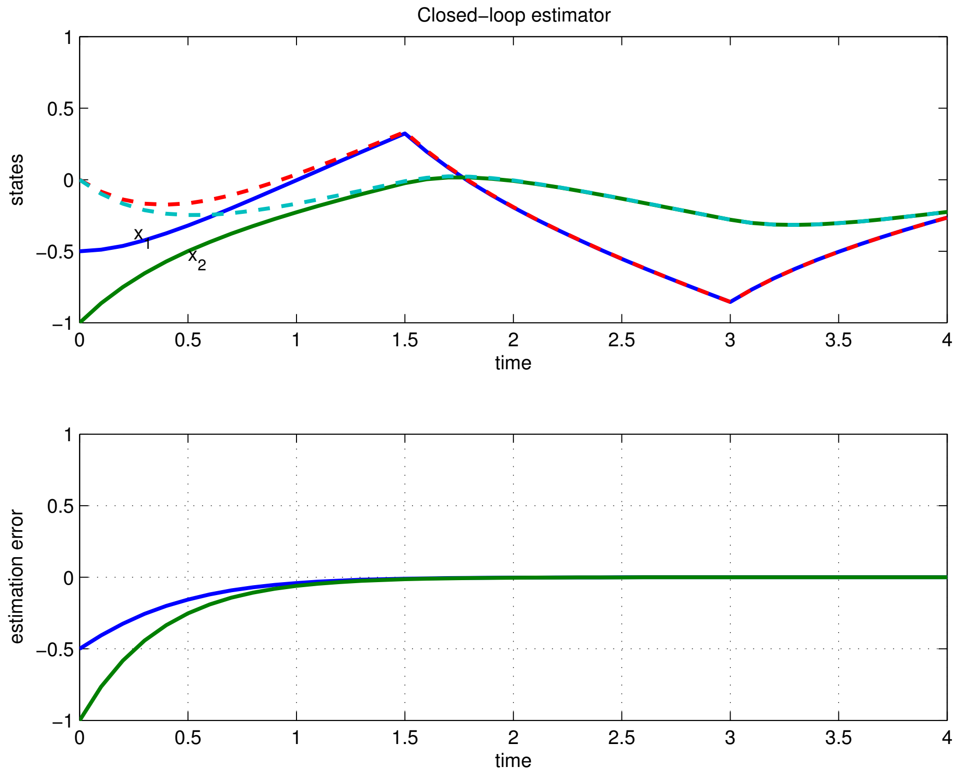


Figure 2: Closed-loop estimator. Convergence looks much better.

Estimator Poles?

- Location heuristics for poles still apply – use Bessel, ITAE, ...
  - Main difference: probably want to make the estimator faster than you intend to make the regulator – should enhance the control, which is based on $\hat{\mathbf{x}}(t)$.
  - ROT: Factor of 2–3 in the time constant $\zeta\omega_n$ associated with the regulator poles.

- **Note:** When designing a regulator, were concerned with "bandwidth" of the control getting too high ⇒ often results in control commands that *saturate* the actuators and/or change rapidly.

- Different concerns for the estimator:
  - Loop closed inside computer, so saturation not a problem.
  - However, the measurements y are often "noisy", and we need to be careful how we use them to develop our state estimates.

⇒ **High bandwidth estimators** tend to accentuate the effect of sensing noise in the estimate.
  - State estimates tend to "track" the measurements, which are fluctuating randomly due to the noise.

⇒ **Low bandwidth estimators** have lower gains and tend to rely more heavily on the plant model
  - Essentially an open-loop estimator – tends to ignore the measurements and just uses the plant model.

Optimal Estimator

- Can also develop an **optimal estimator** for this type of system.
  - Given the duality of the regulator and estimator seen so far, would expect to see close connection between the optimal estimator and the optimal regulator (LQR)

- Key step is to **balance** the effect of the various types of random noise in the system on the estimator:

$$\begin{aligned} \dot{\mathbf{x}} &= A\mathbf{x} + B\mathbf{u} + B_w\mathbf{w} \\ \mathbf{y} &= C_y\mathbf{x} + \mathbf{v} \end{aligned}$$

  - $\mathbf{w}$: "process noise" – models uncertainty in the system model.
  - $\mathbf{v}$: "sensor noise" – models uncertainty in the measurements.

- Typically assume that $w(t)$ and $v(t)$ are
  - Zero mean: $E[\mathbf{w}(t)] = 0$
  - Gaussian white random noises: no correlation between the noise at one time instant and another

$$\begin{aligned} E[\mathbf{w}(t_1)\mathbf{w}(t_2)^T] =& R_{ww}(t_1)\delta(t_1 - t_2) \quad \Rightarrow \mathbf{w}(t) \sim N(0, R_{ww}) \\ E[\mathbf{v}(t_1)\mathbf{v}(t_2)^T] =& R_{vv}(t_1)\delta(t_1 - t_2) \quad \Rightarrow \mathbf{v}(t) \sim N(0, R_{vv}) \\ E[\mathbf{w}(t_1)\mathbf{v}(t_2)^T] =& 0 \end{aligned}$$

- **Goal:** develop an estimator $\hat{\mathbf{x}}(t)$ which is a linear function of the measurements $\mathbf{y}(\tau)$ $(0 \le \tau \le t)$ and minimizes the function

$$E\left[(\mathbf{x}(t) - \hat{\mathbf{x}}(t))(\mathbf{x}(t) - \hat{\mathbf{x}}(t))^T\right]$$

which is the covariance for the estimation error.

- **Solution** is a closed-loop estimator

$$\dot{\hat{\mathbf{x}}}(t) = A\hat{\mathbf{x}} + L(t)(\mathbf{y}(t) - C_y\hat{\mathbf{x}}(t))$$

  where $L(t) = Q(t)C_y^T R_{vv}^{-1}$ and $Q(t) \geq 0$ solves

$$\dot{Q} = AQ + QA^T + B_w R_{ww} B_w^T - QC_y^T R_{vv}^{-1} C_y Q$$

  – Note that $\hat{x}(0)$ and $Q(0)$ are known
  – Differential equation for Q **solved forward in time**.
  – This is the filter form of the differential matrix Riccati equation for the error covariance.

- Called **Kalman-Bucy Filter** – **linear quadratic estimator (LQE)**

- Note that an increase in Q corresponds to **increased uncertainty in the state estimate**. $\dot{Q}$ has several contributions:
  – $AQ + QA^T$ is the homogeneous part
  – $B_w R_{ww} B_w^T$ increase due to the process measurements
  – $QC_y^T R_{vv}^{-1} C_y Q$ decrease due to measurements

- The estimator gain is $L(t) = Q(t)C_y^T R_{vv}^{-1}$
  – If the uncertainty about the state is high, then Q is large, and so the innovation $\mathbf{y} - C_y\hat{\mathbf{x}}$ is weighted heavily ($L \uparrow$)
  – If the measurements are very accurate $R_{vv} \downarrow$, then the measurements are heavily weighted

- With noise in the system, the model is of the form:

$$\dot{\mathbf{x}} = A\mathbf{x} + B\mathbf{u} + B_w\mathbf{w} \ , \quad \mathbf{y} = C_y\mathbf{x} + \mathbf{v}$$

  – And the estimator is of the form:

$$\dot{\hat{\mathbf{x}}} = A\hat{\mathbf{x}} + B\mathbf{u} + L(\mathbf{y} - \hat{\mathbf{y}}) \ , \quad \hat{\mathbf{y}} = C_y\hat{\mathbf{x}}$$

- **Analysis:** in this case:

$$\begin{aligned}
\dot{\tilde{\mathbf{x}}} &= \dot{\mathbf{x}} - \dot{\hat{\mathbf{x}}} = [A\mathbf{x} + B\mathbf{u} + B_w\mathbf{w}] - [A\hat{\mathbf{x}} + B\mathbf{u} + L(\mathbf{y} - \hat{\mathbf{y}})] \\
&= A(\mathbf{x} - \hat{\mathbf{x}}) - L(C_y\mathbf{x} - C_y\hat{\mathbf{x}}) + B_w\mathbf{w} - L\mathbf{v} \\
&= A\tilde{\mathbf{x}} - LC_y\tilde{\mathbf{x}} + B_w\mathbf{w} - L\mathbf{v} \\
&= (A - LC_y)\tilde{\mathbf{x}} + B_w\mathbf{w} - L\mathbf{v}
\end{aligned}$$

- This equation of the estimation error **explicitly** shows the conflict in the estimator design process. Must balance between:
  – Speed of the estimator decay rate, which is governed by $\lambda_i(A-LC_y)$
  – Impact of the sensing noise $\mathbf{v}$ through the gain L

- Fast state reconstruction requires rapid decay rate (typically requires a large L), but that tends to magnify the effect of $\mathbf{v}$ on the estimation process.
  – The effect of the process noise is always there, but the choice of L will tend to mitigate/accentuate the effect of $\mathbf{v}$ on $\tilde{\mathbf{x}}(t)$.

- **Kalman Filter** provides an optimal balance between the two conflicting problems for a given "size" of the process and sensing noises.

Steady State

- Assume that
  1. $R_{vv} > 0$, $R_{ww} > 0$
  2. All plant dynamics are constant in time
  3. $[A, C_y]$ detectable
  4. $[A, B_w]$ stabilizable

- Then, as with the LQR problem, the covariance of the LQE quickly settles down to a constant Q_{ss} independent of $Q(0)$, as $t \to \infty$ where

$$AQ_{ss} + Q_{ss}A^T + B_w R_{ww} B_w^T - Q_{ss}C_y^T R_{vv}^{-1} C_y Q_{ss} = 0$$

  – Stabilizable/detectable gives a unique $Q_{ss} \geq 0$
  – $Q_{ss} > 0$ iff $[A, B_w]$ controllable
  – $L_{ss} = Q_{ss}C_y^T R_{vv}^{-1}$

- If Q_{ss} exists, the steady state filter

$$\begin{aligned}\dot{\hat{\mathbf{x}}}(t) &= A\hat{\mathbf{x}} + L_{ss}(\mathbf{y}(t) - C_y\hat{\mathbf{x}}(t)) \\ &= (A - L_{ss}C_y)\hat{\mathbf{x}}(t) + L_{ss}\mathbf{y}(t)\end{aligned}$$

  is asymptotically stable iff (1)–(4) above hold.

Filter Interpretation

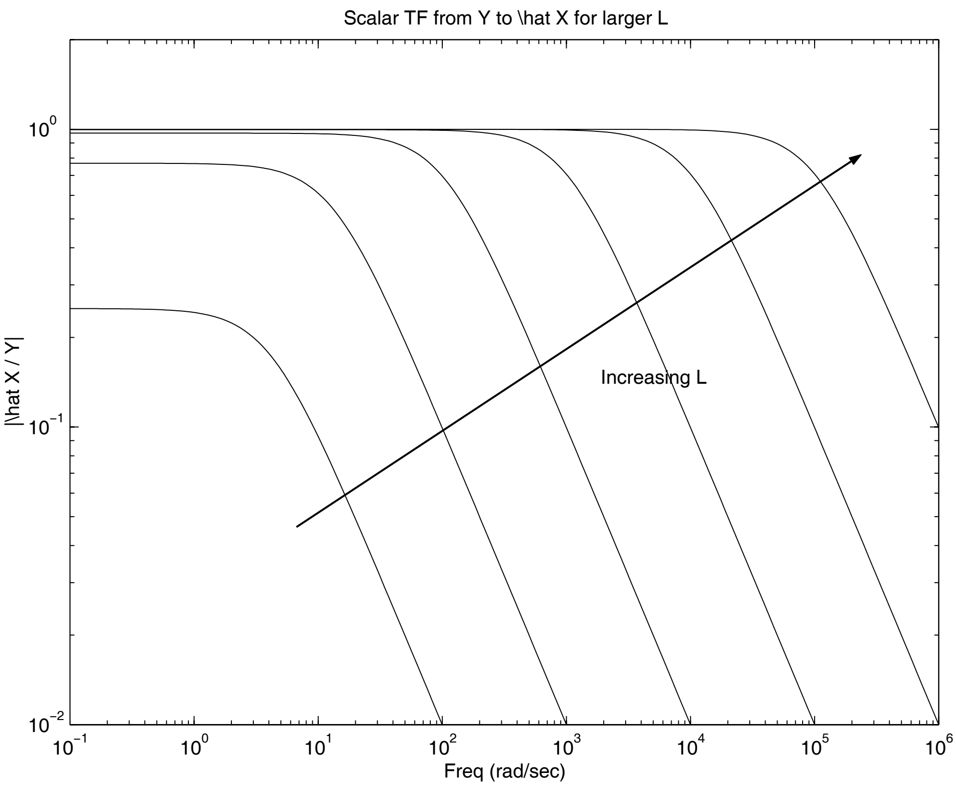
- Given that $\dot{\hat{\mathbf{x}}} = (A - LC_y)\hat{\mathbf{x}} + L\mathbf{y}$

- Consider a scalar system, and take the Laplace transform of both sides to get:

$$\frac{\hat{X}(s)}{Y(s)} = \frac{L}{sI - (A - LC_y)}$$

- This is the transfer function from the "measurement" to the "estimated state"
  – It looks like a low-pass filter.

- Clearly, by lowering R_{vv}, and thus increasing L, we are pushing out the pole.
  – DC gain asymptotes to $1/C_y$ as $L \to \infty$



Example 11–2

- Lightly Damped Harmonic Oscillator

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -\omega_0^2 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} w$$

and $y = x_1 + v$, where $R_{ww} = 1$ and $R_{vv} = r$.

– Can sense the position state of the oscillator, but want to develop an estimator to reconstruct the velocity state.

- **Symmetric root locus** exists for the optimal estimator. Can find location of the optimal poles using a SRL based on the TF

$$G_{yw}(s) = \begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} s & -1 \\ \omega_0^2 & s \end{bmatrix}^{-1} \begin{bmatrix} 0 \\ 1 \end{bmatrix} = \frac{1}{s^2 + \omega_0^2} = \frac{N(s)}{D(s)}$$

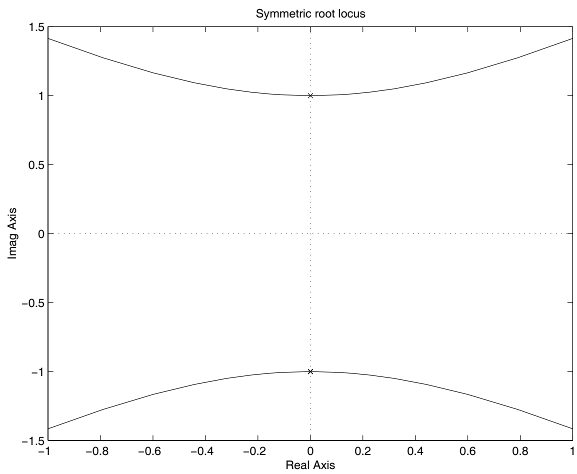
– SRL for the closed-loop poles $\lambda_i(A - LC)$ of the estimator which are the LHP roots of:

$$D(s)D(-s) \pm \frac{R_{ww}}{R_{vv}} N(s)N(-s) = 0$$

– Pick sign to ensure that there are no poles on the jω-axis (other than for a gain of zero)

– So we must find the LHP roots of

$$\left[s^2 + \omega_0^2\right] \left[(-s)^2 + \omega_0^2\right] + \frac{1}{r} = (s^2 + \omega_0^2)^2 + \frac{1}{r} = 0$$



- Note that as $r \to 0$ (clean sensor), the estimator poles tend to ∞ along the ± 45 deg asymptotes, so the poles are approximately

$$s \approx \frac{-1 \pm j}{\sqrt{r}} \quad \Rightarrow \quad \Phi_e(s) = s^2 + \frac{2}{\sqrt{r}} s + \frac{2}{r} = 0$$

- Can use these estimate pole locations in `acker`, to get that

$$\begin{aligned} L &= \left(\begin{bmatrix} 0 & 1 \\ -\omega_0^2 & 0 \end{bmatrix}^2 + \frac{2}{\sqrt{r}} \begin{bmatrix} 0 & 1 \\ -\omega_0^2 & 0 \end{bmatrix} + \frac{2}{r} I \right) \begin{bmatrix} C \\ CA \end{bmatrix}^{-1} \begin{bmatrix} 0 \\ 1 \end{bmatrix} \\ &= \begin{bmatrix} \frac{2}{r} - \omega_0^2 & \frac{2}{\sqrt{r}} \\ -\frac{2}{\sqrt{r}}\omega_0^2 & \frac{2}{r} - \omega_0^2 \end{bmatrix} \begin{bmatrix} \begin{bmatrix} 1 & 0 \end{bmatrix} \\ \begin{bmatrix} 0 & 1 \end{bmatrix} \end{bmatrix}^{-1} \begin{bmatrix} 0 \\ 1 \end{bmatrix} = \begin{bmatrix} \frac{2}{\sqrt{r}} \\ \frac{2}{r} - \omega_0^2 \end{bmatrix} \end{aligned}$$

- Given L, A, and C, we can develop the estimator transfer function from the measurement y to the $\hat{x}_2$

$$\begin{aligned} \frac{\hat{x}_2}{y} &= \begin{bmatrix} 0 & 1 \end{bmatrix} \left(sI - \begin{bmatrix} 0 & 1 \\ -\omega_0^2 & 0 \end{bmatrix} + \begin{bmatrix} \frac{2}{\sqrt{r}} \\ \frac{2}{r} - \omega_0^2 \end{bmatrix} \begin{bmatrix} 1 & 0 \end{bmatrix} \right)^{-1} \begin{bmatrix} \frac{2}{\sqrt{r}} \\ \frac{2}{r} - \omega_0^2 \end{bmatrix} \\ &= \begin{bmatrix} 0 & 1 \end{bmatrix} \begin{bmatrix} s + \frac{2}{\sqrt{r}} & -1 \\ \frac{2}{r} & s \end{bmatrix}^{-1} \begin{bmatrix} \frac{2}{\sqrt{r}} \\ \frac{2}{r} - \omega_0^2 \end{bmatrix} \\ &= \begin{bmatrix} 0 & 1 \end{bmatrix} \begin{bmatrix} s & 1 \\ \frac{-2}{r} & s + \frac{2}{\sqrt{r}} \end{bmatrix} \begin{bmatrix} \frac{2}{\sqrt{r}} \\ \frac{2}{r} - \omega_0^2 \end{bmatrix} \frac{1}{s^2 + \frac{2}{\sqrt{r}} s + \frac{2}{r}} \\ &= \frac{\frac{-2}{r}\frac{2}{\sqrt{r}} + (s + \frac{2}{\sqrt{r}})(\frac{2}{r} - \omega_0^2)}{s^2 + \frac{2}{\sqrt{r}} s + \frac{2}{r}} \approx \frac{s - \sqrt{r}\omega_0^2}{s^2 + \frac{2}{\sqrt{r}} s + \frac{2}{r}} \end{aligned}$$

- Filter zero asymptotes to $s = 0$ as $r \to 0$ and the two poles $\to \infty$
- Resulting estimator looks like a "band-limited" differentiator.
  - This was expected because we measure position and want to estimate velocity.
  - Frequency band over which we are willing to perform the differentiation determined by the "relative cleanliness" of the measurements.

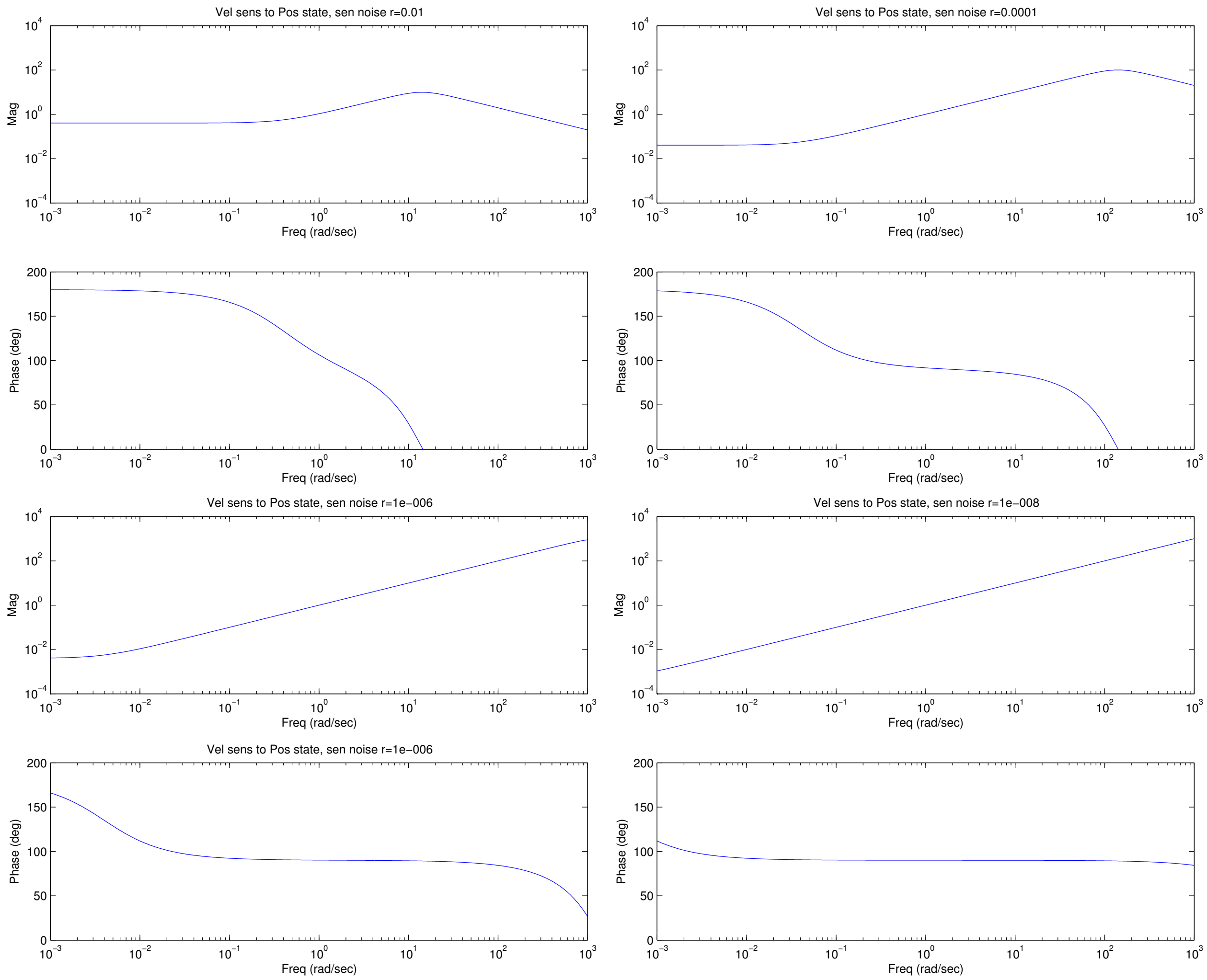


Figure 3: Bandlimited differentiation of the position measurement from LQE: $r = 10^{-2}$, $r = 10^{-4}$, $r = 10^{-6}$, and $r = 10^{-8}$

Final Thoughts

- Note that the feedback gain L in the estimator only stabilizes the estimation error.
  - If the system is unstable, then the state estimates will also go to ∞, with zero error from the actual states.

- Estimation is an important concept of its own.
  - Not always just "part of the control system"
  - Critical issue for guidance and navigation system

- More complete discussion requires that we study stochastic processes and optimization theory.

- **Estimation is all about which do you trust more: your measurements or your model.**

- Strong duality between LQR and LQE problems

$$\begin{array}{ccc} A & \longrightarrow & A^T \\ B & \longrightarrow & C_y^T \\ C_z & \longrightarrow & B_w^T \\ R_{zz} & \longrightarrow & R_{ww} \\ R_{uu} & \longrightarrow & R_{vv} \\ K(t) & \longrightarrow & L^T(t_f - t) \\ P(t) & \longrightarrow & Q(t_f - t) \end{array}$$

Basic Estimator (examp1.m) (See page 11-8)

```
% Examples of estimator performance
% Jonathan How, MIT
% 16.333 Fall 2005
%
% plant dynamics
%
a=[-1 1.5;1 -2];b=[1 0]';c=[1 0];d=0;
%
% estimator gain calc
%
l=place(a',c',[-3 -4]);l=l'
%
% plant initial cond
xo=[-.5;-1];
% extimator initial cond
xe=[0 0]';
t=[0:.1:10];
%
% inputs
%
u=0;u=[ones(15,1);-ones(15,1);ones(15,1)/2;-ones(15,1)/2;zeros(41,1)];
%
% open-loop extimator
%
A_ol=[a zeros(size(a));zeros(size(a)) a];
B_ol=[b;b];
C_ol=[c zeros(size(c));zeros(size(c)) c];
D_ol=zeros(2,1);
%
% closed-loop extimator
%
A_cl=[a zeros(size(a));l*c a-l*c];B_cl=[b;b];
C_cl=[c zeros(size(c));zeros(size(c)) c];D_cl=zeros(2,1);

[y_cl,x_cl]=lsim(A_cl,B_cl,C_cl,D_cl,u,t,[xo;xe]);
[y_ol,x_ol]=lsim(A_ol,B_ol,C_ol,D_ol,u,t,[xo;xe]);

figure(1);clf;subplot(211)
plot(t,x_cl(:,[1 2]),t,x_cl(:,[3 4]),'--','LineWidth',2);axis([0 4 -1 1]);
title('Closed-loop estimator');ylabel('states');xlabel('time')
text(.25,-.4,'x_1');text(.5,-.55,'x_2');subplot(212)
plot(t,x_cl(:,[1 2])-x_cl(:,[3 4]),'LineWidth',2)
%setlines;
axis([0 4 -1 1]);grid on
ylabel('estimation error');xlabel('time')

figure(2);clf;subplot(211)
plot(t,x_ol(:,[1 2]),t,x_ol(:,[3 4]),'--','LineWidth',2);axis([0 4 -1 1])
title('Open loop estimator');ylabel('states');xlabel('time')
text(.25,-.4,'x_1');text(.5,-.55,'x_2');subplot(212)
plot(t,x_ol(:,[1 2])-x_ol(:,[3 4]),'LineWidth',2)
%setlines;
axis([0 4 -1 1]);grid on
ylabel('estimation error');xlabel('time')

print -depsc -f1 est11.eps; jpdf('est11')
print -depsc -f2 est12.eps; jpdf('est12')
```

Filter Interpretation

```
%
% Simple LQE example showing SRL
% 16.323 Spring 2006
% Jonathan How
%
a=[0 1;-4 0];
c=[1 0]; % pos sensor
c2=[0 1]; % vel state out
f=logspace(-3,3,500);

r=1e-2;
l=polyvalm([1 2/sqrt(r) 2/r],a)*inv([c;c*a])*[0 1]'
[nn,dd]=ss2tf(a-l*c,l,c2,0); % to the vel estimate
g=freqresp(nn,dd,f*j);
[r roots(nn)]
figure(1)
subplot(211)
loglog(f,abs(g))
%hold on;fill([5e2 5e2 1e3 1e3 5e2]',[1e4 1e-4 1e-4 1e4 1e4]','c');hold off
xlabel('Freq (rad/sec)')
ylabel('Mag')
title(['Vel sens to Pos state, sen noise r=',num2str(r)])
axis([1e-3 1e3 1e-4 1e4])
subplot(212)
semilogx(f,unwrap(angle(g))*180/pi)
xlabel('Freq (rad/sec)')
ylabel('Phase (deg)')
axis([1e-3 1e3 0 200])

figure(2)
r=1e-4;
l=polyvalm([1 2/sqrt(r) 2/r],a)*inv([c;c*a])*[0 1]'
[nn,dd]=ss2tf(a-l*c,l,c2,0); % to the vel estimate
g=freqresp(nn,dd,f*j);
[r roots(nn)]
subplot(211)
loglog(f,abs(g))
%hold on;fill([5e2 5e2 1e3 1e3 5e2]',[1e4 1e-4 1e-4 1e4 1e4]','c');hold off
xlabel('Freq (rad/sec)')
ylabel('Mag')
title(['Vel sens to Pos state, sen noise r=',num2str(r)])
axis([1e-3 1e3 1e-4 1e4])
subplot(212)
semilogx(f,unwrap(angle(g))*180/pi)
xlabel('Freq (rad/sec)')
ylabel('Phase (deg)')
%bode(nn,dd);
axis([1e-3 1e3 0 200])

figure(3)
r=1e-6;
l=polyvalm([1 2/sqrt(r) 2/r],a)*inv([c;c*a])*[0 1]'
[nn,dd]=ss2tf(a-l*c,l,c2,0); % to the vel estimate
g=freqresp(nn,dd,f*j);
[r roots(nn)]
subplot(211)
loglog(f,abs(g))
%hold on;fill([5e2 5e2 1e3 1e3 5e2]',[1e4 1e-4 1e-4 1e4 1e4]','c');hold off
xlabel('Freq (rad/sec)')
ylabel('Mag')
title(['Vel sens to Pos state, sen noise r=',num2str(r)])
axis([1e-3 1e3 1e-4 1e4])
subplot(212)
semilogx(f,unwrap(angle(g))*180/pi)
xlabel('Freq (rad/sec)')
ylabel('Phase (deg)')
%bode(nn,dd);
```

```
title(['Vel sens to Pos state, sen noise r=',num2str(r)])
axis([1e-3 1e3 0 200])

figure(4)
r=1e-8;
l=polyvalm([1 2/sqrt(r) 2/r],a)*inv([c;c*a])*[0 1]'
[nn,dd]=ss2tf(a-l*c,l,c2,0); % to the vel estimate
g=freqresp(nn,dd,f*j);
[r roots(nn)]
subplot(211)
loglog(f,abs(g))
%hold on;fill([5e2 5e2 1e3 1e3 5e2]',[1e4 1e-4 1e-4 1e4 1e4]','c');hold off
xlabel('Freq (rad/sec)')
ylabel('Mag')
title(['Vel sens to Pos state, sen noise r=',num2str(r)])
axis([1e-3 1e3 1e-4 1e4])
title(['Vel sens to Pos state, sen noise r=',num2str(r)])
subplot(212)
semilogx(f,unwrap(angle(g))*180/pi)
xlabel('Freq (rad/sec)')
ylabel('Phase (deg)')
%bode(nn,dd);
axis([1e-3 1e3 0 200])

print -depsc -f1 filt1.eps; jpdf('filt1')
print -depsc -f2 filt2.eps;jpdf('filt2')
print -depsc -f3 filt3.eps;jpdf('filt3')
print -depsc -f4 filt4.eps;jpdf('filt4')
```