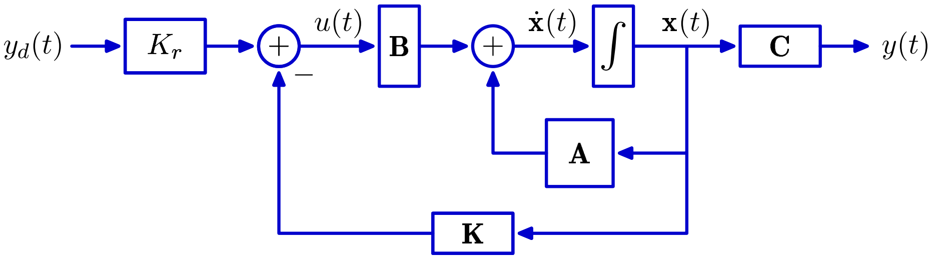
# 6.3100: Dynamic System Modeling and Control Design

## State-Space Control with Observers

*November 20, 2024*

## Full-State Feedback

One of the most powerful features of the state-space approach to control is the ability to incorporate feedback from **all** of the states of the plant.

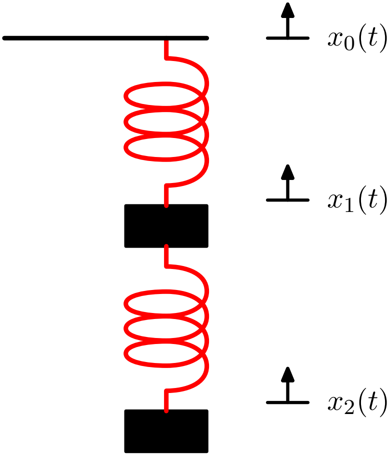


This can be especially important in systems with internal states that are difficult to control.

Example: two-spring system

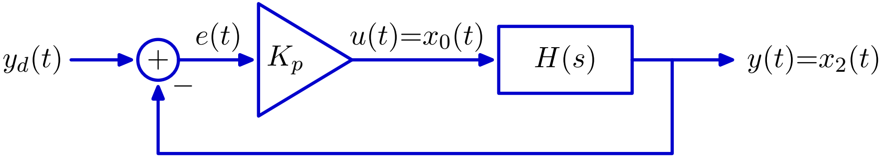
# Example: Two-Spring System

The **plant** consists of two springs and two masses. Use the input $u(t) = x_0(t)$ to move the bottom mass to the desired location $x_2(t) = y_d(t)$.

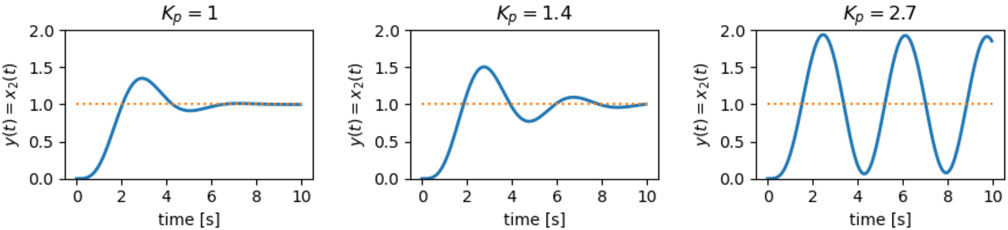


# Proportional Control

This system is difficult to control. A proportional controller converges slowly, has large overshoots, and oscillates.



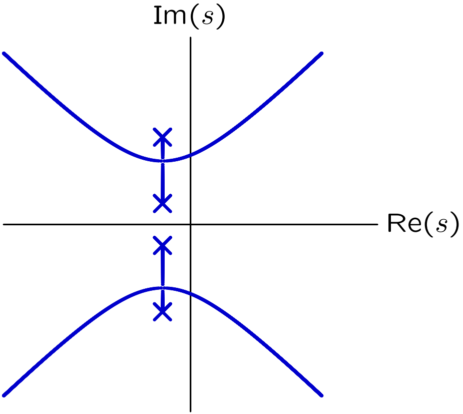
Step responses (mass $m = 1$, stiffness $k = 2$, damping $b = 1.4$):



This feedback system is stable for only a small range of gains: $K_p < 2.7$.

# Classical Control

Root Locus: As $K_p$ increases, the lower and higher frequency poles converge with no change in damping, then split and approach asymptotic trajectories at angles of $\pm\pi/4$ and $\pm 3\pi/4$. Unstable when poles enter right half-plane.

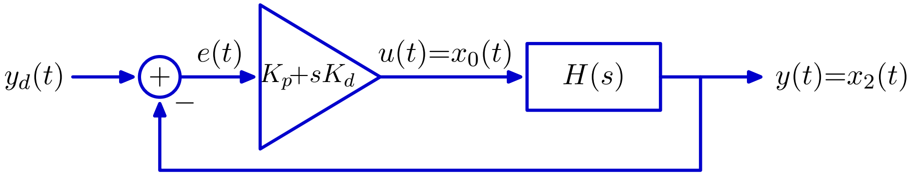


Good explanation of what happened.

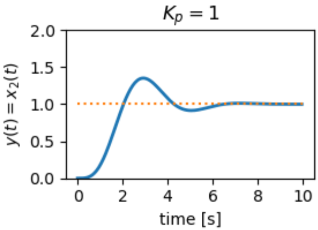
**Try proportional plus derivative control.**

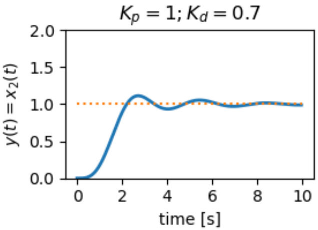
# Proportional Plus Derivative Control

Proportional plus derivative performance is better.



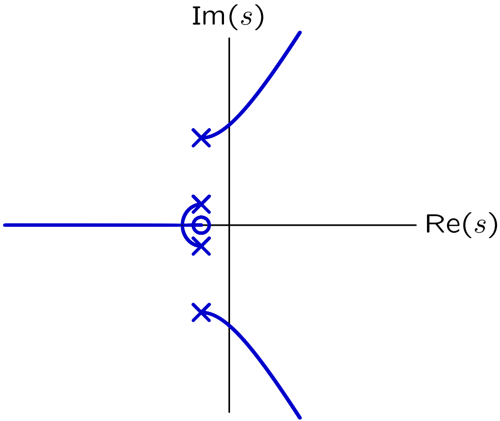
Step responses:





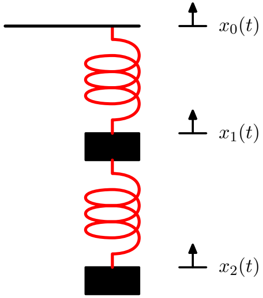
# Classical Control

Root Locus: Increase $K_p$ while holding $K_d = K_p/0.7$. Derivative term adds a zero and changes the asymptotic behavior, but closed-loop system still goes unstable.

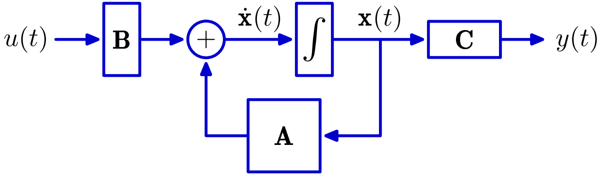


Good explanation of what happened – but how do we make it faster? Try state-space approach.

# State-Space Control with Full-State Feedback

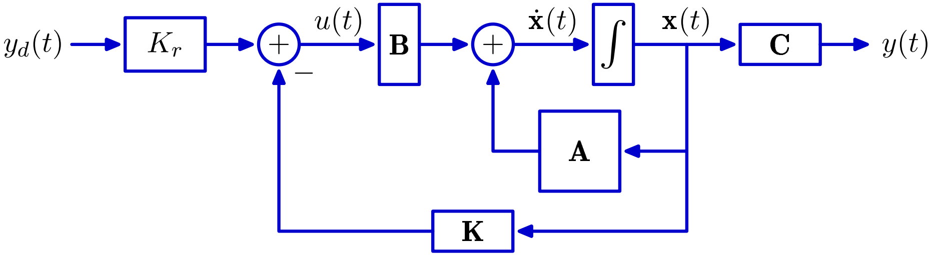


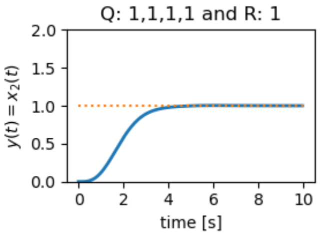
$$f_{m1} = m\ddot{x}_1(t) = k\Big(x_0(t) - x_1(t)\Big) - k\Big(x_1(t) - x_2(t)\Big) - b\dot{x}_1(t)$$
$$f_{m2} = m\ddot{x}_2(t) = k\Big(x_1(t) - x_2(t)\Big) - b\dot{x}_2(t)$$



# State-Space Control

A state-space controller can work better.

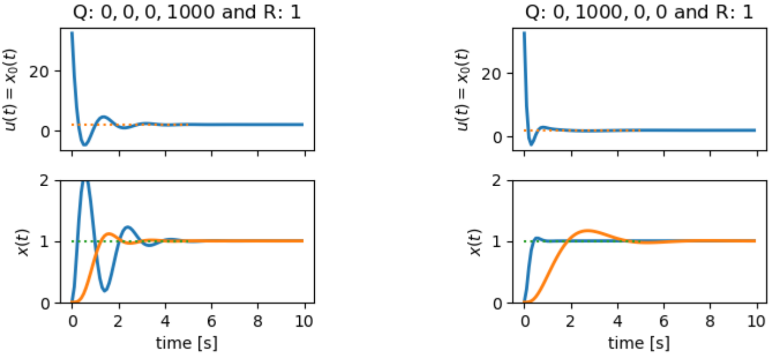




Although a bit slow, the response is at least monotonic.

# State-Space Control

Better performance results with feedback from the internal mass ($x_1$) than with feedback from the output mass ($x_2$)!



Left panels show results when feedback is from lower (output) mass only. Right panels show results when feedback is from upper (internal) mass only.

- displacement of upper (internal) mass shown in blue
- displacement of lower (output) mass shown in gold

# Beyond State-Space Control

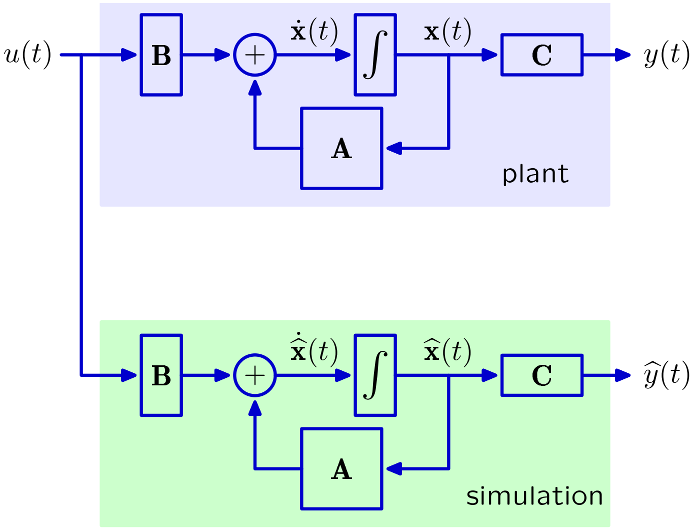
However, to feed back information about $x_1(t)$, we must **measure** $x_1(t)$.

What if it's not possible to measure $x_1(t)$.

**Idea:** Could we simulate the unmeasured states?

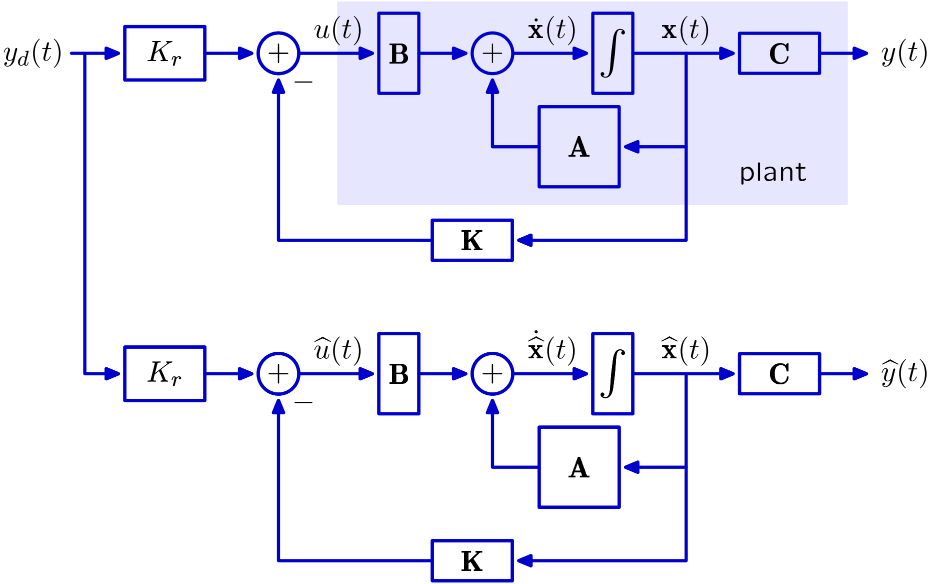
# Observers

An **observer** is a **simulation** of the plant that can provide information about unmeasured states. This **simulation** will be part of the controller!



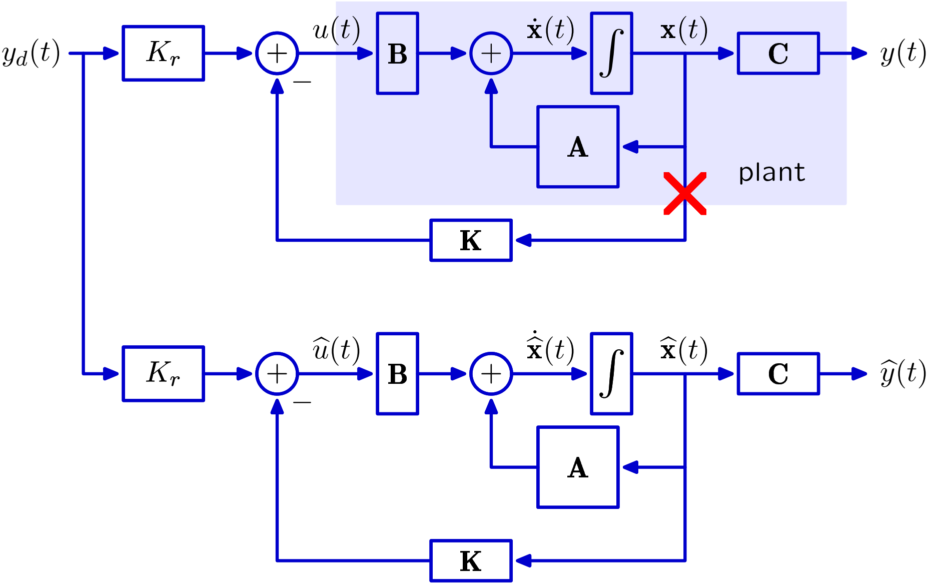
# Observers

We can build state-space **controllers** for both the plant and the simulation. The simulation then provides estimates of the state ($\widehat{\mathbf{x}}(t)$) and input ($\widehat{u}(t)$).



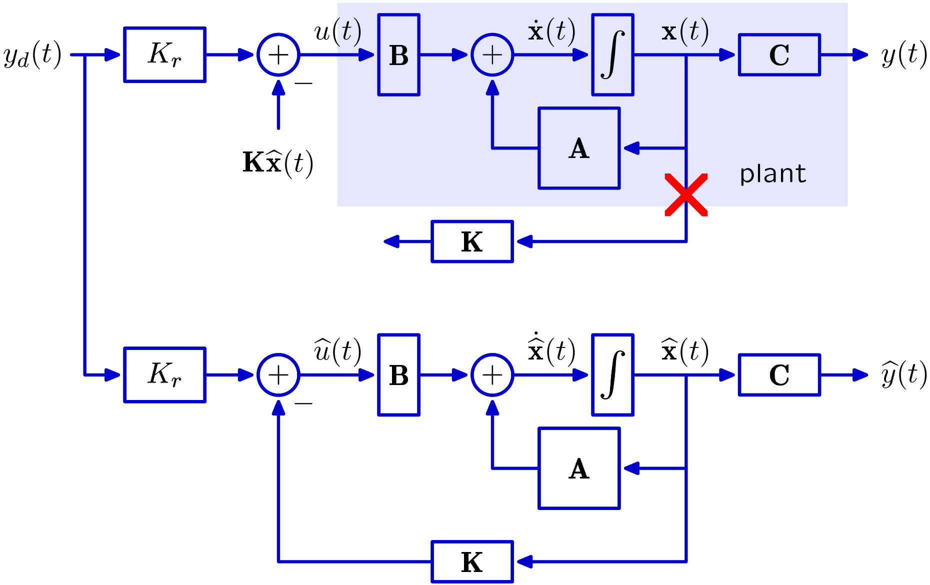
# Observers

Then if feedback from all of the states in $\mathbf{x}(t)$ is not possible ...



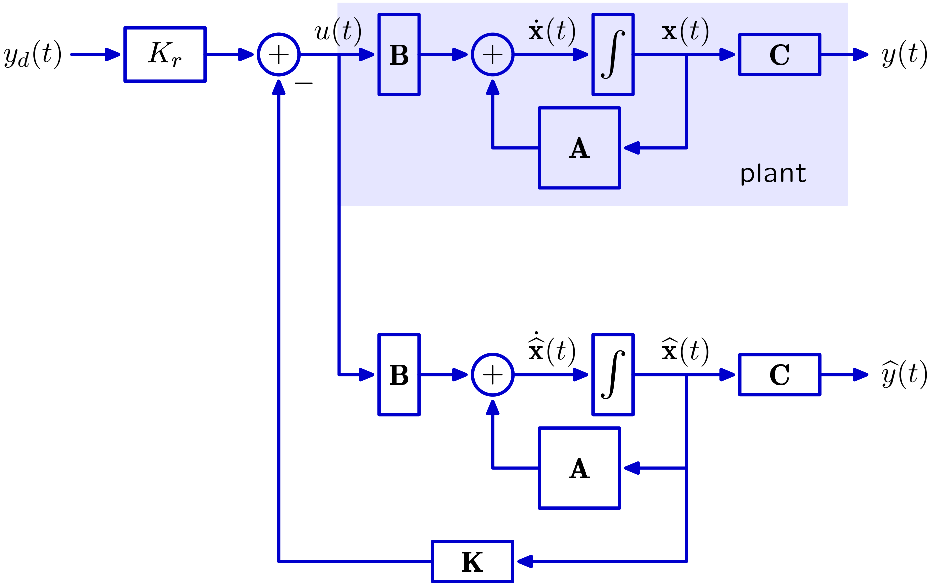
# Observers

… we can substitute the simulated states $\widehat{\mathbf{x}}(t)$ for the missing states $\mathbf{x}(t)$. Similarly, we can substitute $\widehat{u}(t)$ for $u(t)$ as well.



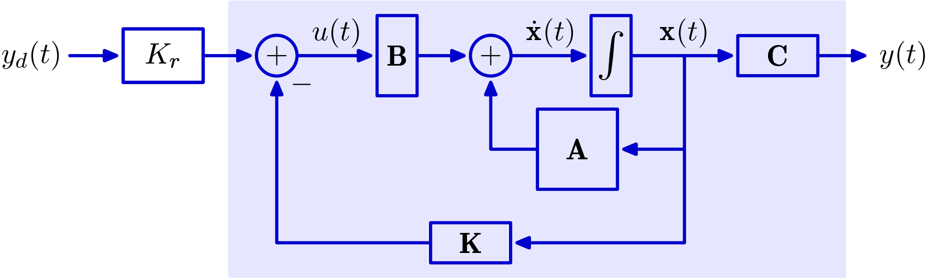
## Observers

The resulting controller takes advantage of all of the states in $\mathbf{x}(t)$ without measuring them. Really? Sounds a bit too good to be true!



# Last Time: Using Feedback to Reduce Tracking Errors

Recall the problem we discussed last time with using $K_r$ to adjust the overall gain of a state-space controller.

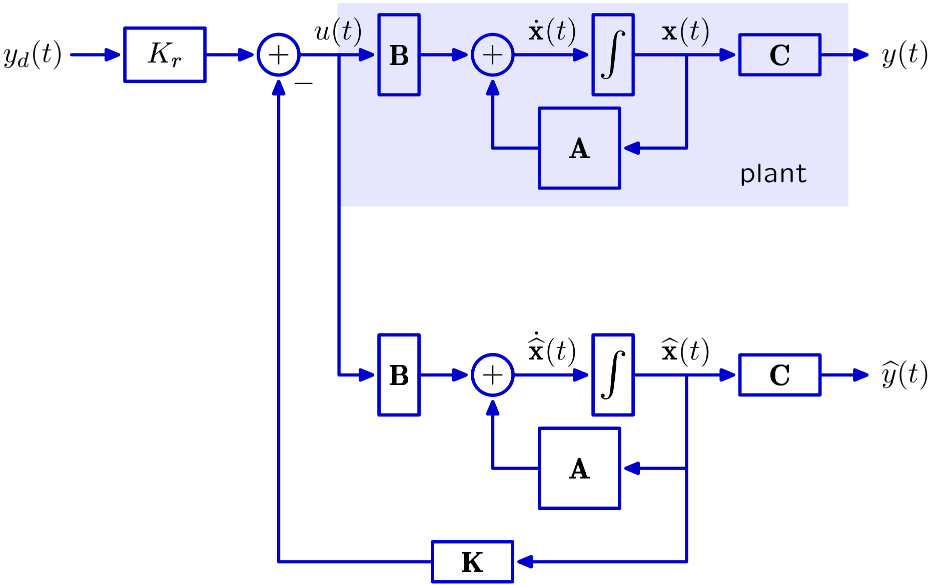


This method uses $K_r$ to anticipate and pre-correct unwanted offsets in the rest of the system.

Using $K_r$ to eliminate tracking errors is a **feed-forward** approach!
A better approach is to incorporate a new state variable $w(t)$ to monitor tracking errors and then **use feedback** to reduce $w(t)$ and thereby tracking error to zero.

## Observers

Here, it's not only $K_r$ but also the entire simulation of the plant that is intended to anticipate and pre-correct deficiencies of the plant.

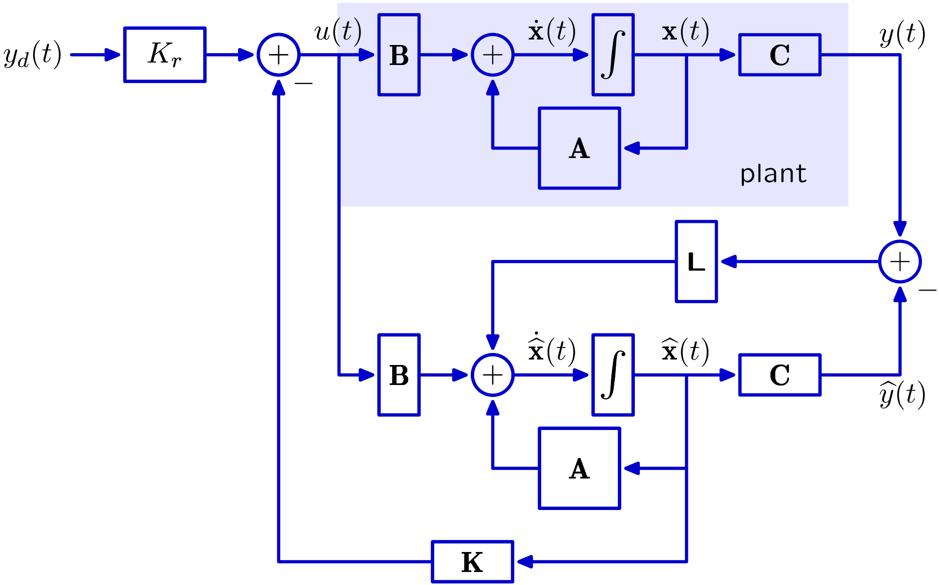


## Observers

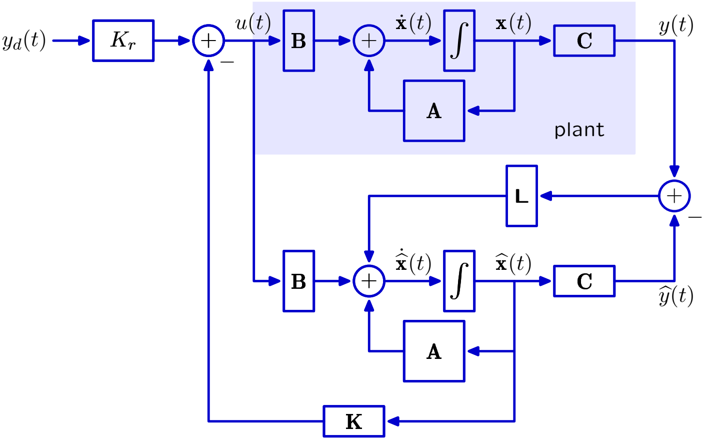
Fortunately, we can use **feedback** to correct simulation errors!

Calculate the difference between $y(t)$ and $\widehat{y}(t)$.

Then use that signal (times $\mathbf{L}$) to correct $\dot{\widehat{\mathbf{x}}}(t)$.



# Analyzing the Observer Model



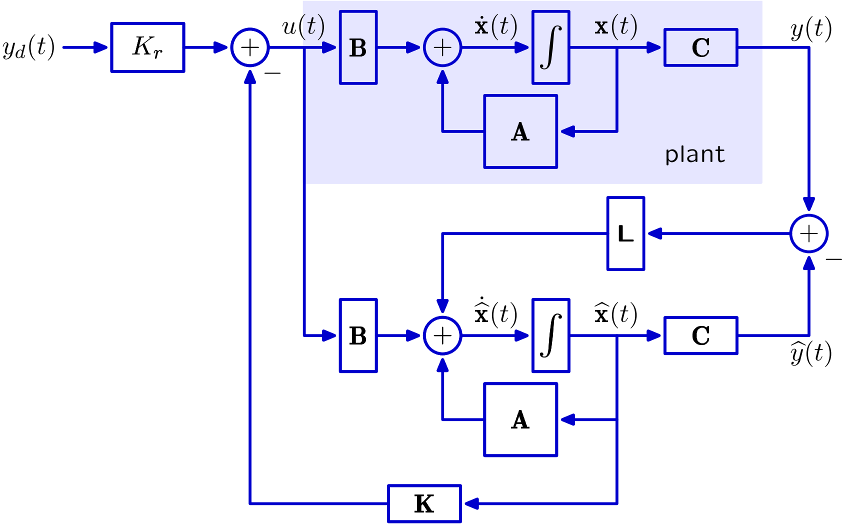
Which (if any) of the following expressions are correct?

1. $\dot{\mathbf{x}}(t) = \mathbf{A}\widehat{\mathbf{x}}(t) - \mathbf{B}\mathbf{K}\mathbf{x}(t) + \mathbf{B}K_r y_d(t)$
2. $\dot{\mathbf{x}}(t) = \mathbf{A}\mathbf{x}(t) - \mathbf{B}\mathbf{K}\mathbf{x}(t) + \mathbf{B}K_r y_d(t)$
3. $\dot{\widehat{\mathbf{x}}}(t) = \mathbf{A}\widehat{\mathbf{x}}(t) - \mathbf{B}\mathbf{K}\widehat{\mathbf{x}}(t) + \mathbf{B}K_r y_d(t) + \mathbf{L}(y(t) - \widehat{y}(t))$
4. $\dot{\widehat{\mathbf{x}}}(t) = \mathbf{A}\widehat{\mathbf{x}}(t) - \mathbf{B}\mathbf{K}\widehat{\mathbf{x}}(t) + \mathbf{B}K_r y_d(t)$
5. $\dot{\widehat{\mathbf{x}}}(t) = \dot{\mathbf{x}}(t)$

# Observers

Plant dynamics: $\dot{\mathbf{x}}(t) = \mathbf{A}\mathbf{x}(t) - \mathbf{B}\mathbf{K}\widehat{\mathbf{x}}(t) + \mathbf{B}K_r y_d(t)$

Simulation dynamics: $\dot{\widehat{\mathbf{x}}}(t) = \mathbf{A}\widehat{\mathbf{x}}(t) - \mathbf{B}\mathbf{K}\widehat{\mathbf{x}}(t) + \mathbf{B}K_r y_d(t) + \mathbf{L}(y(t) - \widehat{y}(t))$



We can analyze the model with a combined state vector $\mathbf{x}_+(t) = \begin{bmatrix} \mathbf{x}(t) \\ \widehat{\mathbf{x}}(t) \end{bmatrix}$

# Observers

Combined dynamics of the plant and observer.

$$\dot{\mathbf{x}}(t) = \mathbf{A}\mathbf{x}(t) - \mathbf{B}\mathbf{K}\widehat{\mathbf{x}}(t) + \mathbf{B}K_r y_d(t)$$

$$\dot{\widehat{\mathbf{x}}}(t) = \mathbf{A}\widehat{\mathbf{x}}(t) - \mathbf{B}\mathbf{K}\widehat{\mathbf{x}}(t) + \mathbf{B}K_r y_d(t) + \mathbf{L}\Big(y(t) - \widehat{y}(t)\Big)$$

Define $\mathbf{e}(t)$ to be the difference between the plant and simulation states:

$$\mathbf{e}(t) = \mathbf{x}(t) - \widehat{\mathbf{x}}(t)$$

Subtract $\dot{\widehat{\mathbf{x}}}(t)$ from $\dot{\mathbf{x}}(t)$ to find the derivative of $\mathbf{e}(t)$:

$$\dot{\mathbf{e}}(t) = \mathbf{A}\mathbf{e}(t) - \mathbf{L}\Big(y(t) - \widehat{y}(t)\Big) = \mathbf{A}\mathbf{e}(t) - \mathbf{L}\mathbf{C}\mathbf{e}(t)$$

Append the $\dot{\mathbf{x}}(t)$ and $\dot{\mathbf{e}}(t)$ to make a new **combined** state vector:

$$\begin{bmatrix} \dot{\mathbf{x}}(t) \\ \dot{\mathbf{e}}(t) \end{bmatrix} = \begin{bmatrix} \mathbf{A}-\mathbf{B}\mathbf{K} & \mathbf{B}\mathbf{K} \\ \mathbf{0} & \mathbf{A}-\mathbf{L}\mathbf{C} \end{bmatrix} \begin{bmatrix} \mathbf{x}(t) \\ \mathbf{e}(t) \end{bmatrix} + \begin{bmatrix} \mathbf{B} \\ \mathbf{0} \end{bmatrix} K_r y_d(t)$$

Notice that the resulting matrix equation has the same form as the original state evolution equation:

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

where $\mathbf{A}, \mathbf{B}$, and $\mathbf{x}(t)$ have been extended to include error terms.

# Observers

Combined dynamics of the plant and observer.

$$\begin{bmatrix} \dot{\mathbf{x}}(t) \\ \dot{\mathbf{e}}(t) \end{bmatrix} = \begin{bmatrix} \mathbf{A}{-}\mathbf{BK} & \mathbf{BK} \\ \mathbf{0} & \mathbf{A}{-}\mathbf{LC} \end{bmatrix} \begin{bmatrix} \mathbf{x}(t) \\ \mathbf{e}(t) \end{bmatrix} + \begin{bmatrix} \mathbf{B} \\ \mathbf{0} \end{bmatrix} K_r y_d(t)$$

The poles of this system are the roots of its characteristic equation:

$$\left| s\mathbf{I} - \begin{bmatrix} \mathbf{A}{-}\mathbf{BK} & \mathbf{BK} \\ \mathbf{0} & \mathbf{A}{-}\mathbf{LC} \end{bmatrix} \right| = 0$$

Because the evolution matrix has block triangular form, the characteristic equation can be factored into two parts:

$$\left| s\mathbf{I} - \begin{bmatrix} \mathbf{A}{-}\mathbf{BK} & \mathbf{BK} \\ \mathbf{0} & \mathbf{A}{-}\mathbf{LC} \end{bmatrix} \right| = \left| s\mathbf{I} - (\mathbf{A}{-}\mathbf{BK}) \right| \times \left| s\mathbf{I} - (\mathbf{A}{-}\mathbf{LC}) \right| = 0$$

and the poles of the augmented system are the union of the poles of the plant and simulation dynamics.

Furthermore, the poles of the plant and observer can be chosen independently, so we can pick an $\mathbf{L}$ to give fast decay of observer state errors (going from $\mathbf{x}(t)$ to $\widehat{\mathbf{x}}(t)$) relative to tracking errors (going from $y_d(t)$ to $y(t)$).

# Linear Quadratic Regulator (LQR)

The LQR method minimizes a **cost function** $J$ that describes the relative cost (or badness) of inputs $\mathbf{u}(t)$ and responses $\mathbf{x}(t)$.

The cost function $J$ is the time integral of a weighted sum of the squares of state variables $\mathbf{x}(t)$ and input $\mathbf{u}(t)$

$$J = \int_0^\infty \Big(\mathbf{x}^\mathbf{T}(t)\,\mathbf{Q}\mathbf{x}(t) + \mathbf{u}^\mathbf{T}(t)\,\mathbf{R}\mathbf{u}(t)\Big)dt$$

where $\mathbf{u}(t)$ and $\mathbf{x}(t)$ are related

- by the state transition equation: $\dot{\mathbf{x}}(t) = \mathbf{A}\mathbf{x}(t) + \mathbf{B}\mathbf{u}(t)$ and
- by the feedback constraint: $\mathbf{u}(t) = -\mathbf{K}\mathbf{x}(t)$.

and $\mathbf{Q}$ and $\mathbf{R}$ represent weights.

The “optimal” $\mathbf{K}$ is given by

$$\mathbf{K} = \mathbf{R}^{-\mathbf{1}}\mathbf{B}^\mathbf{T}\mathbf{S}$$

where $\mathbf{S}$ is the symmetric $n\times n$ solution to the **algebraic Riccati equation**:

$$\mathbf{A}^\mathbf{T}\mathbf{S} + \mathbf{S}\mathbf{A} - \mathbf{S}\mathbf{B}\mathbf{R}^{-\mathbf{1}}\mathbf{B}^\mathbf{T}\mathbf{S} + \mathbf{Q} = \mathbf{0}$$

# Linear Quadratic Regulator (LQR) – Redux!

The LQR method minimizes a **cost function** $J'$ that describes the relative cost (or badness) of output errors $y(t) - \widehat{y}(t)$ and state errors $\widehat{\mathbf{e}}(t)$.

The cost function $J'$ is the time integral of a weighted sum of the squares of the state errors $\mathbf{e}(t)$ and output variables $y(t) - \widehat{y}(t)$

$$J' = \int_0^\infty \left( \mathbf{e}(t)^T \mathbf{Q}\, \mathbf{e}(t) + (y(t) - \widehat{y}(t))^T\, \mathbf{R}\, (y(t) - \widehat{y}(t)) \right) dt$$

where $(y(t) - \widehat{y}(t))$ and $\mathbf{e}(t)$ are related

- by a state transition equation: $\dot{\mathbf{e}}(t) = \mathbf{A}\mathbf{e}(t) - \mathbf{L}(y(t) - \widehat{y}(t))$ and
- by the feedback constraint: $y(t) - \widehat{y}(t) = \mathbf{C}\mathbf{e}(t)$.

and $\mathbf{Q}$ and $\mathbf{R}$ represent weights.

The "optimal" $\mathbf{L}$ is given by

$$\mathbf{L}^{\mathbf{T}} = \mathbf{R}^{-\mathbf{1}}\mathbf{C}\mathbf{S}$$

where $\mathbf{S}$ is the symmetric $n \times n$ solution to the **algebraic Riccati equation**:

$$\mathbf{A}\mathbf{S} + \mathbf{S}\mathbf{A}^{\mathbf{T}} - \mathbf{S}\mathbf{C}^{\mathbf{T}}\mathbf{R}^{-\mathbf{1}}\mathbf{C}\mathbf{S} + \mathbf{Q} = \mathbf{0}$$

# Choosing L

Since optimizing **K** and **L** can be cast into problems with the same form, the optimizations can be solved using the same methods.

```
K = place(A,B,[poles])
L = place(A.’,C.’,[poles]).’
```

or

```
K = lqr(A,B,Qk,Rk)
L = lqr(A.’,C.’,Ql,Rl).’
```

# Summary

Today we formulated a new approach to control based on **observers**.

- An observer is a simulation of the plant that is part of the controller.
- The biggest challenge in designing an observer is keeping its state up-to-date with that of the plant.
- We can feedback the difference between the measured and simulated outputs to correct the simulated states.