Lecture 5 –Sampled Time Control

- Sampled time vs. continuous time: implementation, aliasing
- Linear sampled systems modeling refresher, DSP
- Sampled-time frequency analysis
- Sampled-time implementation of the basic controllers
  - I, PI, PD, PID
  - 80% (or more) of control loops in industry are digital PID

Sampled Time Models

- Time is often sampled because of the digital computer use
  - digital (sampled time) control system

$$x(t+d) = f(x,u,t)$$

$$y = g(x,u,t)$$

- Numerical integration of continuous-time ODE

$$x(t+d) \approx x(t) + d \cdot f(x,u,t), \qquad t = kd$$

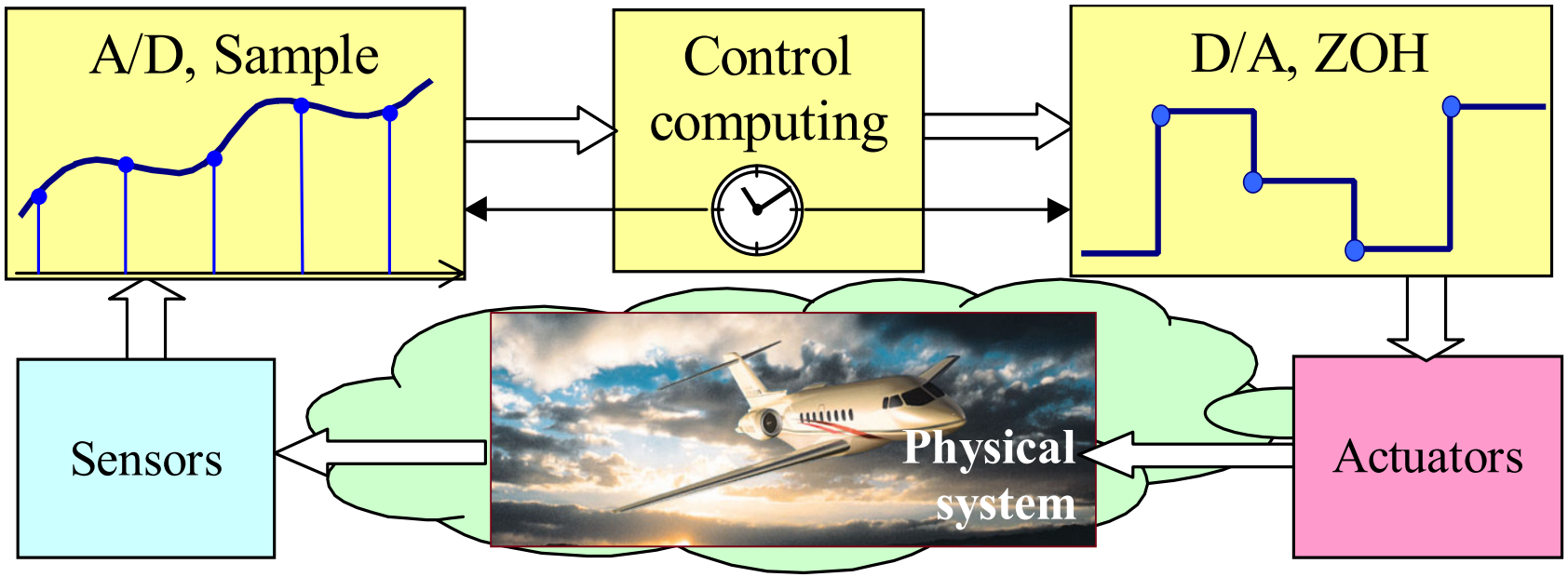
- Time can be sampled because this is how a system works
- Example: bank account balance
  - $x(t)$ - balance in the end of day t
  - $u(t)$ - total of deposits and withdrawals that day
  - $y(t)$ - displayed in a daily statement

$$x(t+1) = x(t) + u(t)$$

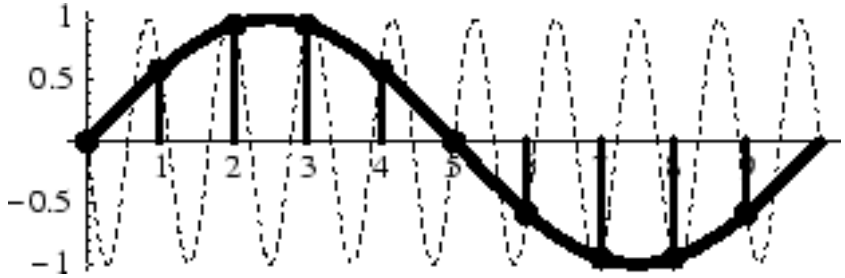
$$y = x$$

Sampling: continuous time view

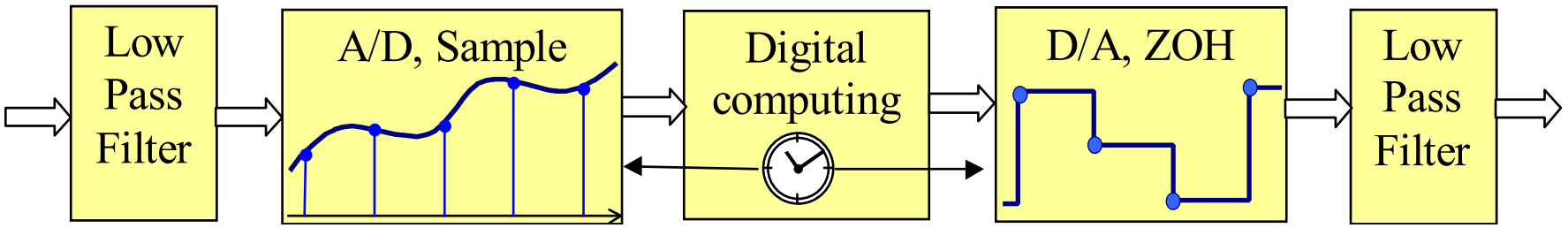
- Sampled and continuous time together
- Continuous time physical system + digital controller
  - ZOH = Zero Order Hold



Signal sampling, aliasing



- Nyquist frequency:
  $\omega_N = \frac{1}{2}\omega_S$; $\omega_S = 2\pi/T$
- Frequency folding: $k\omega_S \pm \omega$ map to the same frequency ω
- Sampling Theorem: sampling is Ok if there are no frequency components above ω_N
- Practical approach to anti-aliasing: low pass filter (LPF)
- Sampled→continuous: impostoring



Linear state space model

- Generic state space model:

$$x(t+1) = f(x,u,t)$$

$$y = g(x,u,t)$$

- LTI state space model

$$\boxed{\begin{aligned} x(t+1) &= Ax(t) + Bu(t) \\ y(t) &= Cx(t) \end{aligned}}$$

  - Physics-based linear system model
  - Obtained by sampling a continuous time model
  - Zero-order hold (ZOH)

$$\begin{aligned} \dot{x}(t) &= A_c x(t) + B_c u(t) \\ y(t) &= C_c x(t) \end{aligned}$$

$$x(t+T) = \underbrace{e^{A_c T} x(t)}_{A} + \underbrace{\left(\int_0^T e^{A_c t} B_c dt \right)}_{B} u(t)$$

$$y(t) = C_c x(t)$$

- Matlab commands for model conversion: **`help ltimodels`**

Disk read-write head example

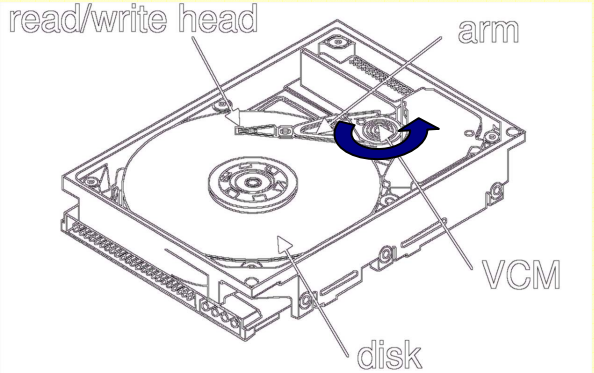
- Sampled time model for T=0.0001 (use Matlab `ss` and `c2d`)

$$x(k+1) = \underbrace{\begin{bmatrix} 0.998 & 0 \\ 10^{-4} & 0 \end{bmatrix}}_{A} \cdot x(k) + \underbrace{\begin{bmatrix} 10^{-3} \\ 5 \cdot 10^{-8} \end{bmatrix}}_{B} \cdot u(k)$$

$$y(k) = \underbrace{\begin{bmatrix} 0 & 1 \end{bmatrix}}_{C} \cdot x(k)$$

$$\boxed{\begin{aligned} x(k+1) &= Ax(k) + Bu(t) \\ y(k) &= Cx(k) \end{aligned}}$$

Read-write head control



$$\ddot{y} + b\dot{y} = gu \qquad b = 20, g = 10$$

$$y = \frac{10}{s(s+20)}u$$

$$\frac{dx}{dt} = \underbrace{\begin{bmatrix} -20 & 0 \\ 1 & 0 \end{bmatrix}}_{A} \cdot x + \underbrace{\begin{bmatrix} 10 \\ 0 \end{bmatrix}}_{B} \cdot u$$

$$y = \underbrace{\begin{bmatrix} 0 & 1 \end{bmatrix}}_{C} \cdot x$$

Sampled-time system stability

$$x(k+1) = Ax(k), \qquad x(0) = x_0$$

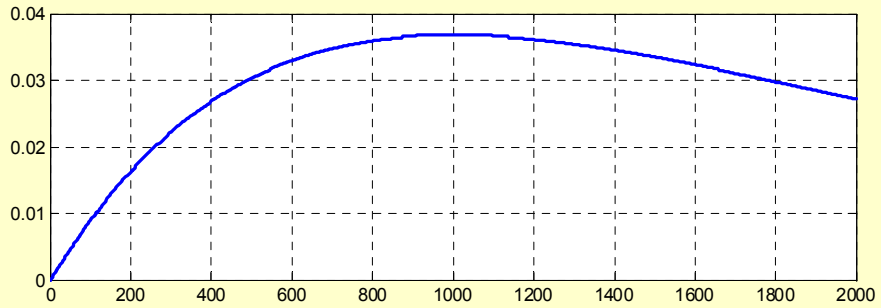
- Initial condition response

$$x(k) = A^k x_0$$

- Exponential stability condition

$$\{\lambda_j\} = \text{eig}(A)$$

$$|\lambda_j| < 1$$

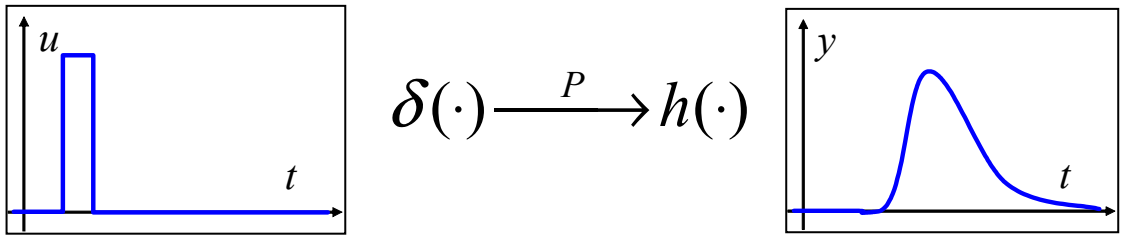


Example:

```
% Take A, B, C
% from the R-W head example
>> eig(A)
>> ans =
        1.0000
        0.9980
>>
>> K = 10
>> Ak = A - K*B*C;
>> eig(Ak)
>> ans =
        0.9990
        0.9990
>>
>> N  = 2000;
>> x(:,1) = [1; 0];
>> for t=2:N,
>>    x(:,t) = Ak*x(:,t-1);
>> end
>>
>> plot(1:N,x(2,:))
```

Impulse response

- Response to an input impulse



- Sampled time: $t = 1, 2, \ldots$
- Control history = linear combination of the impulses $\Rightarrow$ system response = linear combination of the impulse responses

$$u(t) = \sum_{k=0}^{\infty} \delta(t-k)u(k)$$

$$y(t) = \sum_{k=0}^{\infty} h(t-k)u(k) = (h * u)(t)$$

Convolution representation of a sampled-time system

- Convolution

$$y(t) = \sum_{k=-\infty}^{t} h(t-k)u(k)$$

signal processing notation $y = h * u$

- Impulse response

$$u(t) = \delta(t) \quad \Rightarrow \quad y(t) = h(t)$$

- Step response: $u = 1$ for $t > 0$

$$g(t) = \sum_{k=0}^{t} h(t-k) = \sum_{j=0}^{t} h(j) \qquad h(t) = g(t) - g(t-1)$$

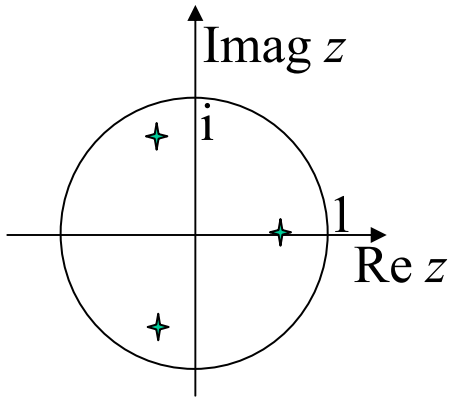
Matlab commands: `g = cumsum(h);` `h = diff(g);`

z-transform vs. Laplace transform

- Discrete (z-transform) transfer function:

$$H(z) = \sum_{k=0}^{\infty} h(k) z^{-k}$$

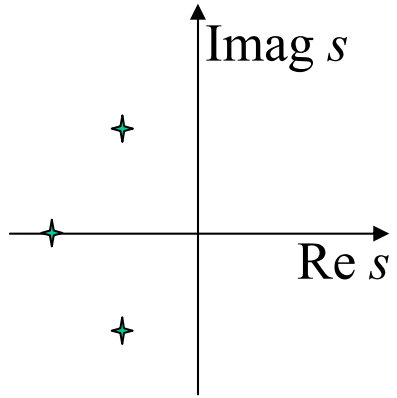
  - function of complex variable z
  - $z \rightarrow$ forward shift operator
  - analytical outside the circle $|z| \geq r$
  - all poles are inside the circle
  - for a stable system $r \leq 1$



- Laplace transform transfer function:

$$H(s) = \int_{-\infty}^{\infty} h(t) e^{st} dt$$

  - function of complex variable s
  - $s \rightarrow$ differentiation operator
  - analytical in a half plane $\mathrm{Re}\, s \leq a$
  - for a stable system $a \leq 1$



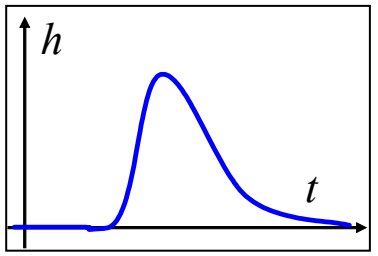
Sampled time vs. continuous time

- Continuous time analysis (digital implementation of a continuous time controller)
  - Tustin's method = trapezoidal rule of integration for $H(s) = \frac{1}{s}$

    $$H(s) \to H_s(z) = H\left(s = \frac{2}{T} \cdot \frac{1 - z^{-1}}{1 + z^{-1}} \right)$$
  - Matched Zero Pole: map each zero and a pole in accordance with $s = e^{zT}$
- Sampled time analysis (Sampling of continuous signals and system)
- Systems analysis is often performed continuous time - this is where the controlled plant is.

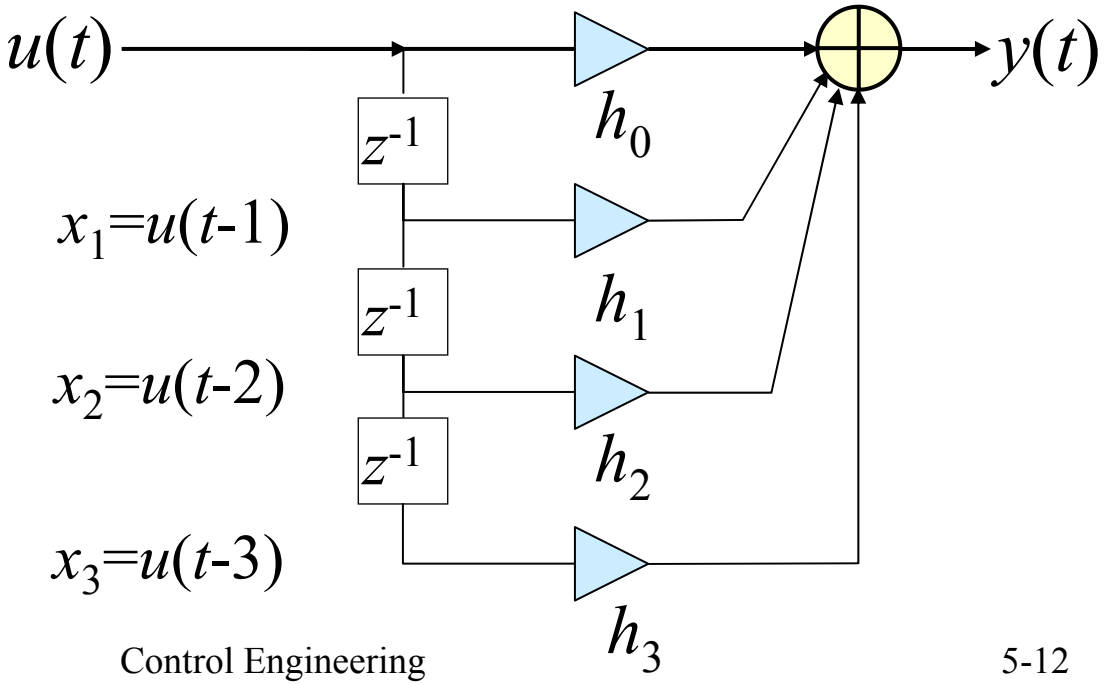
FIR model

$$y(t) = \sum_{k=0}^{N} h_{FIR}(t-k)u(k) = (h_{FIR} * u)(t)$$



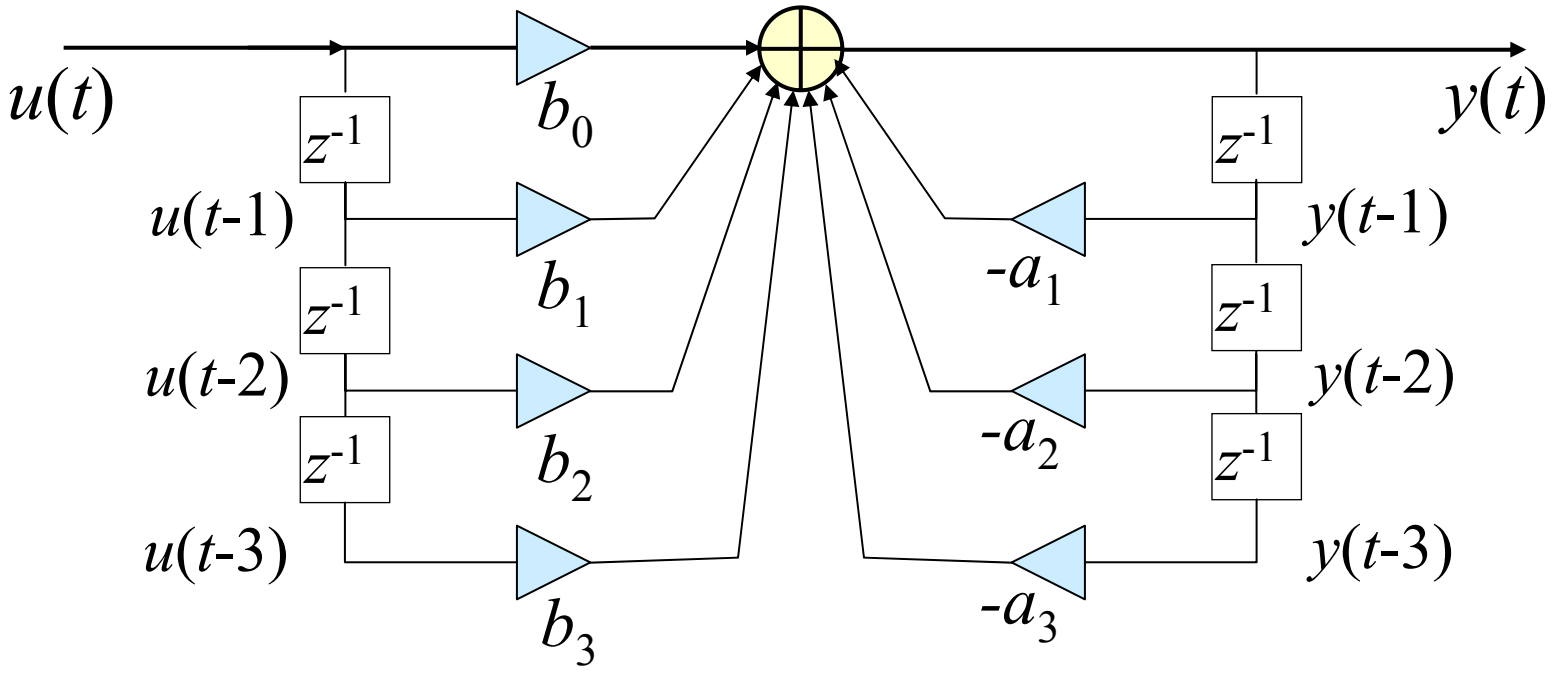
- FIR = Finite Impulse Response
- Cut off the trailing part of the impulse response to obtain FIR
- FIR filter state x is the shift register content

$$\begin{aligned} x(t+1) &= Ax(t) + Bu(t) \\ y(t) &= Cx(t) \end{aligned}$$



IIR model

- IIR model: $y(t) = -\sum_{k=1}^{n_a} a_k y(t-k) + \sum_{k=0}^{n_b} b_k u(t-k)$
- Filter states: $y(t\text{-}1), \ldots, y(t\text{-}n_a), u(t\text{-}1), \ldots, u(t\text{-}n_b)$



IIR model

- Matlab implementation of an IIR model: **`filter`**
- Transfer function realization: unit delay operator z^{-1}

$$y(t) = H(z)u(t)$$

$$H(z) = \frac{B(z)}{A(z)} = \frac{b_0 z^N + b_1 z^{N-1} + ... + b_N}{z^N + a_1 z^{N-1} + ... + a_N}$$

Control convention
Matlab `tf, step` etc

$$= \frac{b_0 + b_1 z^{-1} + ... + b_N z^{-N}}{1 + a_1 z^{-1} + ... + a_N z^{-N}}$$

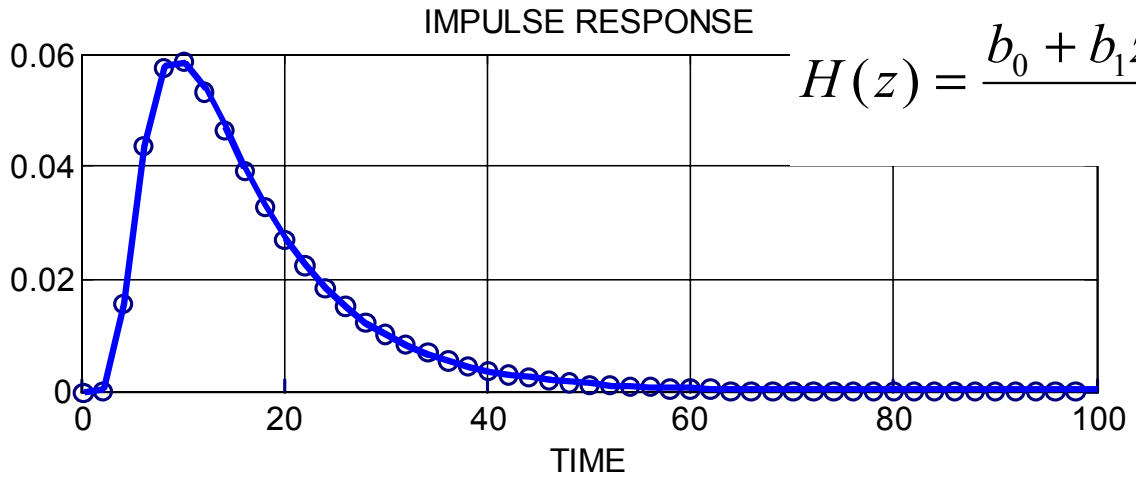
Signal processing convention
Matlab `filter` etc

$$\underbrace{\left(1 + a_1 z^{-1} + ... + a_N z^{-N}\right)}_{A(z)} y(t) = \underbrace{\left(b_0 + b_1 z^{-1} + ... + b_N z^{-N}\right)}_{B(z)} u(t)$$

- FIR model is a special case of an IIR with $A(z) = 1$ (or z^N)

IIR approximation example

- Low order IIR approximation of impulse response: (**`prony`** in Matlab Signal Processing Toolbox)
- Fewer parameters than a FIR model
- Example: sideways heat transfer
  - impulse response $h(t)$
  - approximation with IIR filter $a = [a_1\ a_2]$, $b=[b_0\ b_1\ b_2\ b_3\ b_4]$



$$H(z) = \frac{b_0 + b_1 z^{-1} + b_2 z^{-2} + b_3 z^{-3} + b_4 z^{-4}}{1 + a_1 z^{-1} + a_2 z^{-2}}$$

Linear state space model

- LTI state space model

$$\boxed{\begin{aligned} x(t+1) &= Ax(t) + Bu(t) \\ y(t) &= Cx(t) + Du(t) \end{aligned}}$$

- Unit delay operator z^{-1}: $z^{-1}\,x(t) = x(t\text{-}1)$
- Transfer function of an LTI model
  - defines an IIR representation

$$y = \left[\left(Iz - A\right)^{-1} B + D\right] \cdot u$$

$$H(z) = C\left(Iz - A\right)^{-1} B + D$$

- Stability: the poles are inside the unit circle

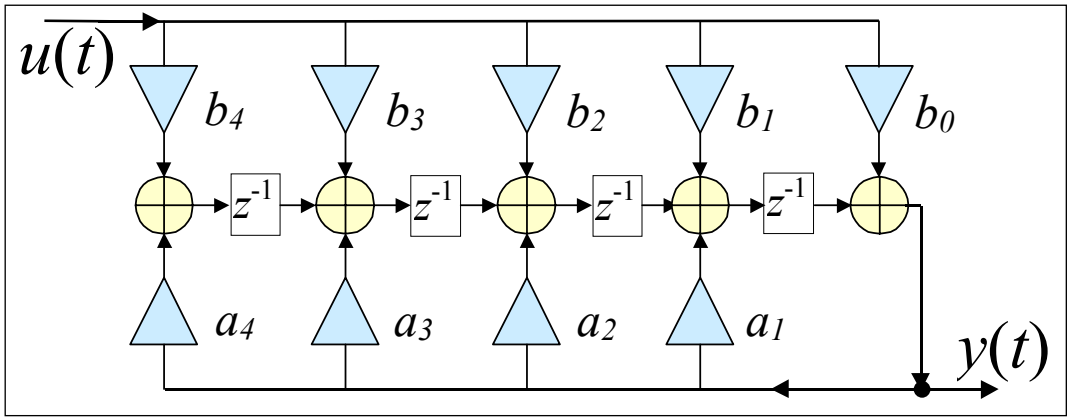
Stability analysis

- Transfer function poles tell you everything about stability
- Model-based analysis for a simple feedback example:

$$\left.\begin{aligned} y &= H(z)u \\ u &= -K(y - y_d) \end{aligned}\right\} \Longrightarrow y = \frac{H(z)K}{1 + H(z)K} y_d = L(z) y_d$$

- If $H(z)$ is a rational transfer function describing an IIR model
- Then $L(z)$ also is a rational transfer function describing an IIR model

State space realization of IIR filter



$$y(t) = H(z)u(t)$$

$$H(z) = \frac{B(z)}{A(z)} = \frac{b_0 + b_1 z^{-1} + \ldots + b_N z^{-N}}{1 + a_1 z^{-1} + \ldots + a_N z^{-N}}$$

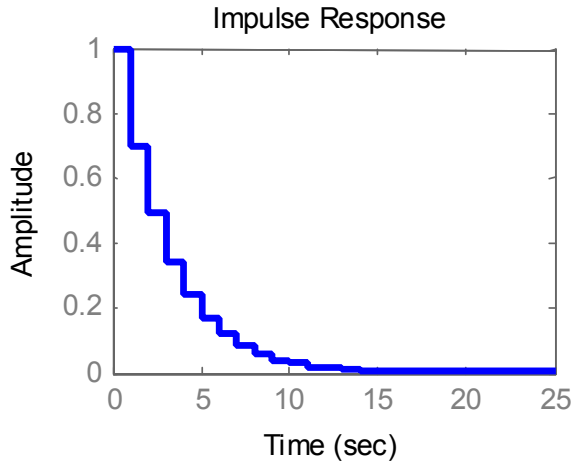
$$\begin{bmatrix} x_1(t+1) \\ x_2(t+1) \\ x_3(t+1) \\ x_4(t+1) \end{bmatrix} = \begin{bmatrix} -a_1 & -a_2 & -a_3 & -a_3 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \cdot \begin{bmatrix} x_1(t) \\ x_2(t) \\ x_3(t) \\ x_4(t) \end{bmatrix} + \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix} u(t)$$

$$y(t) = \begin{bmatrix} b_1 - a_1 b_0 & b_2 - a_2 b_0 & b_3 - a_3 b_0 & b_4 - a_4 b_0 \end{bmatrix} \cdot \begin{bmatrix} x_1(t) \\ x_2(t) \\ x_3(t) \\ x_4(t) \end{bmatrix} + b_0 u(t)$$

Poles and Zeros ←→ System

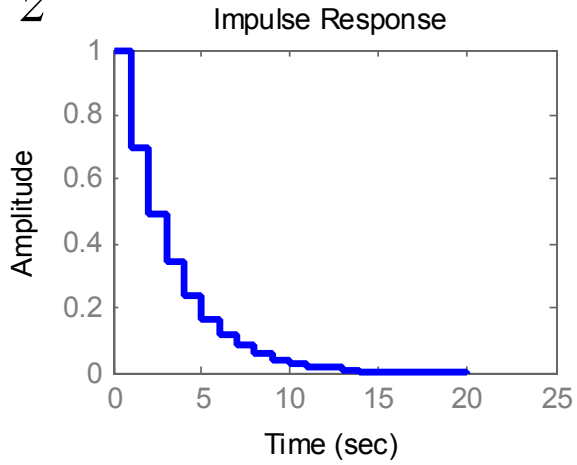
- …**not quite so!**
- Example:

$$y = H(z)u = \frac{z}{z - 0.7}$$



- FIR model - truncated IIR

$$y = H_{FIR}(z)u = \frac{z^{19} + 0.7z^{18} + 0.49z^{17} + \ldots + 0.001628\,z + 0.00114}{z^{19}}$$



IIR/FIR example - cont'd

- Feedback control:

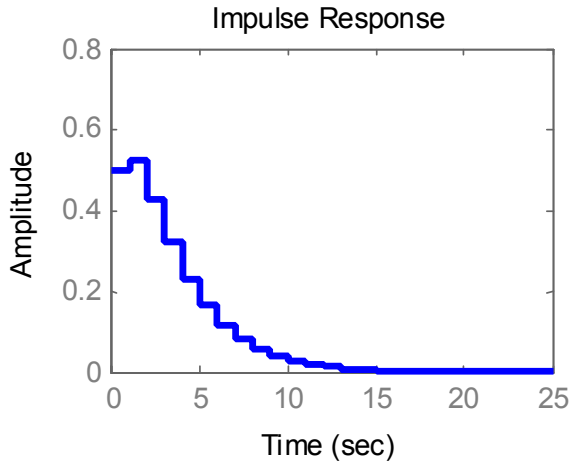
$$y = H(z)u = \frac{z}{z - 0.7}$$

$$u = -K(y - y_d) = -(y - y_d)$$

- Closed loop:

$$y = \frac{H(z)}{1 + H(z)}u = L(z)u$$

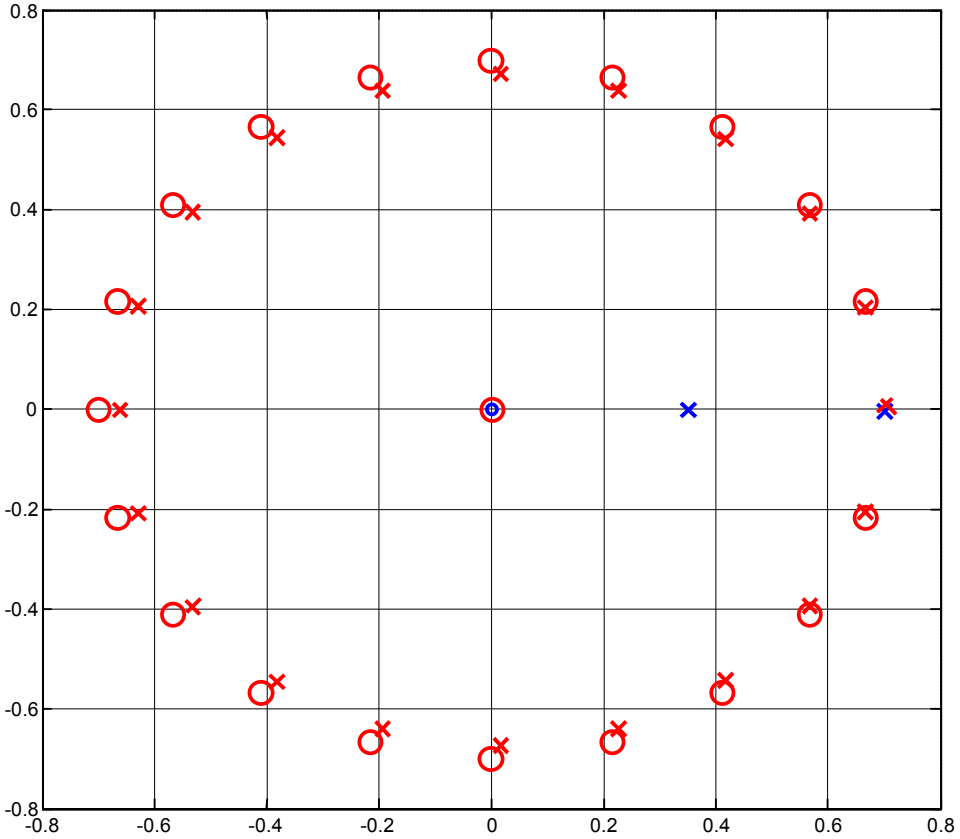
$$y = \frac{H_{FIR}(z)}{1 + H_{FIR}(z)}u = L_{FIR}(z)u$$



IIR/FIR example - cont'd
Poles and zeros

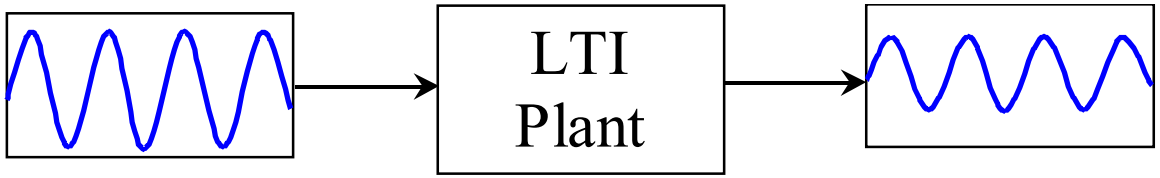
- **Blue:** Loop with IIR model poles x and zeros o

- **Red**: Loop with FIR model poles x and zeros o



Frequency domain description

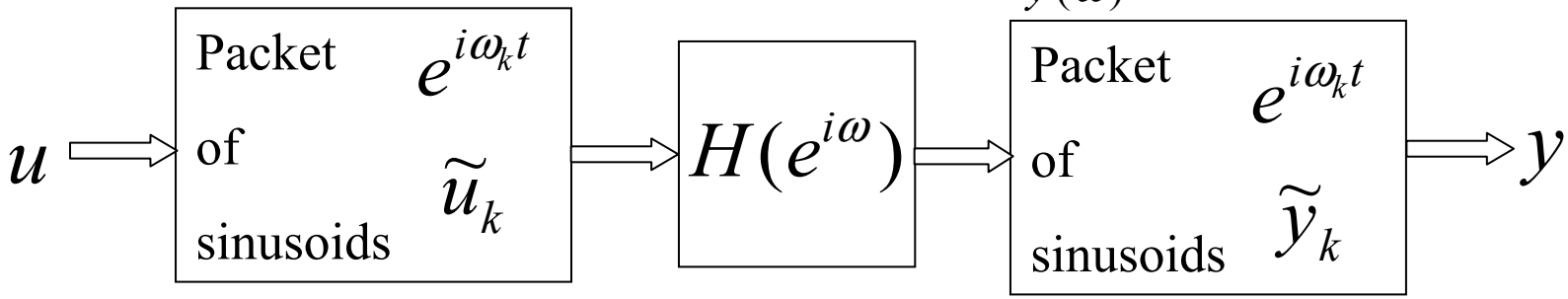
- Sinusoids are eigenfunctions of an LTI system: $y = H(z)u$



$$ze^{i\omega t} = e^{i\omega(t+1)} = e^{i\omega}e^{i\omega t} \qquad z \to e^{i\omega}$$

- Frequency domain analysis ←→ system diagonalization

$$u = \sum \tilde{u}_k e^{i\omega_k t} \Rightarrow y = \sum \underbrace{H(e^{i\omega_k})\tilde{u}_k}_{\tilde{y}(\omega)} e^{i\omega_k t}$$



Frequency domain description

- Bode plots:

$$u = e^{i\omega t}$$

$$y = H(e^{i\omega})e^{i\omega t}$$

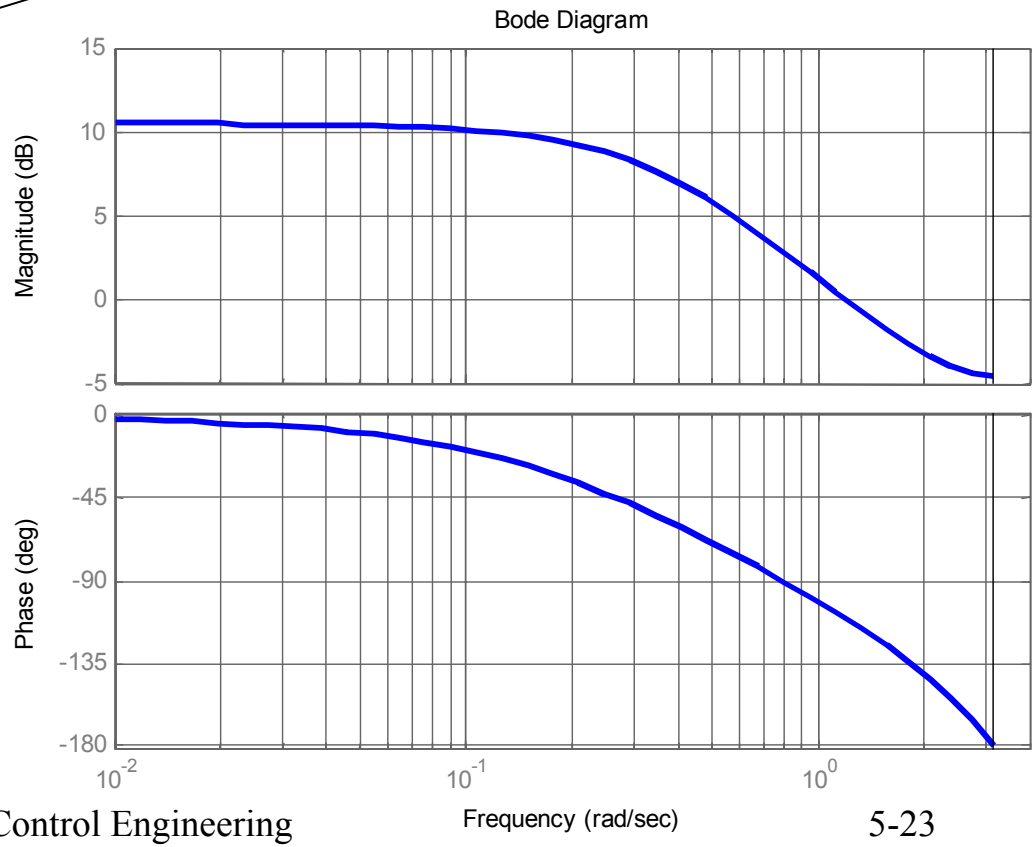
$$M(\omega) = \left|H(e^{i\omega})\right|$$

$$\varphi(\omega) = \arg H(e^{i\omega})$$

- Example:

$$H(z) = \frac{1}{z - 0.7}$$

- $|H|$ is often measured in dB



Fourier transform for sampled signals

$$\tilde{x}(\omega) = \sum_{t=-\infty}^{\infty} x(t) e^{i\omega t}$$

- Fourier transform of a series $[-\infty : 1 : \infty] \rightarrow [0, 2\pi]$

$$x(t) = \frac{1}{2\pi} \int_0^{2\pi} \tilde{x}(\omega) e^{-i\omega t} d\omega$$

- Inverse Fourier transform $[0, 2\pi] \rightarrow [-\infty : 1 : \infty]$

- If $x(t)$ is a smoothly varying signal, the sum approximates the integral for continuous FT
- $\tilde{x}(\omega)$ approximates the continuos FT up to the Nyquist frequency of $\omega = \pi$
- If sampling frequency is other than unity, need to scale

Discrete Fourier Transform = DFT

DFT is computed by Matlab **`FFT`**

$$\tilde{x}(\omega_k) = \sum_{t=1}^{N} x(t) e^{i\omega_k t}, \qquad \omega_k = \frac{2\pi}{N} k$$

`f=FFT(x)`

$$x(t) = \frac{1}{N} \sum_{k=0}^{N-1} \tilde{x}(\omega_k) e^{-i\omega_k t}$$

`x=iFFT(f)`

- DFT
  $[0:1:N] \rightarrow [0:(2\pi/N):2\pi]$
- Inverse DFT
  $[0:(2\pi/N):2\pi] \rightarrow [0:1:N]$

- DFT = FT of a sequence: …, 0, $x(1)$, $x(2)$, …, $x(N)$, 0, 0, ...
- The sum in Inverse DFT approximates the infinite sequence inverse FT up to the 'Nyquist time' of $N/2$

Sampled-time LTI models - recap

- Linear system can be described by impulse response
- Linear system can be described by frequency response = Fourier transform of the impulse response
- FIR, IIR, State-space models can be used to obtain close approximations of sampled-time linear system
- A pattern of poles and zeros can be very different for a small change in an input-output model.
- Impulse response, step response, and frequency response do not change much for small model changes

Sampled time I control

- Step to step update:

$$y(t) = g \cdot u(t) + d(t)$$
$$u(t) = u(t-1) + v(t-1)$$
$$v(t) = k_I \left[y(t) - y_d \right]$$

sampled time integrator

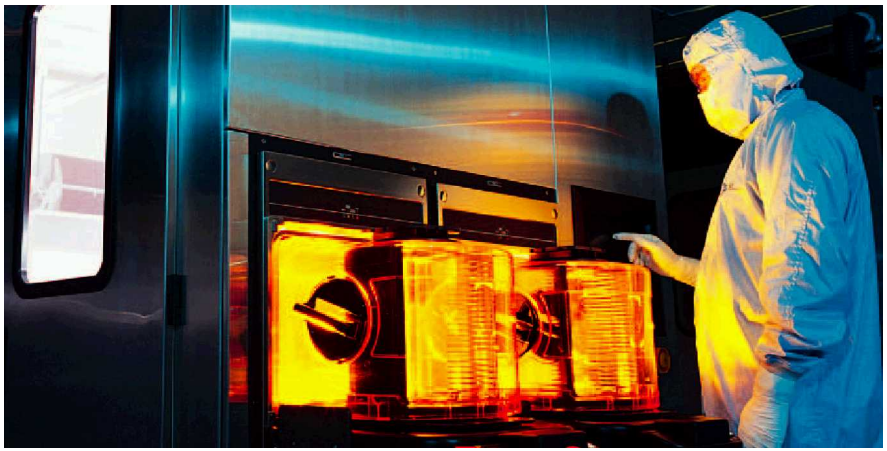
- Closed-loop dynamics

$$y = g \cdot u + d$$
$$u = \frac{k_I}{z-1}\left[y - y_d \right] \quad \Longrightarrow \quad y = \frac{g k_I}{z - 1 + g k_I} y_d + \frac{z-1}{z - 1 + g k_I} d$$

- Deadbeat control: $g k_I = 1 \quad \Longrightarrow \quad y = z^{-1} y_d + \left(1 - z^{-1}\right) d$

Run-to-run (R2R) control

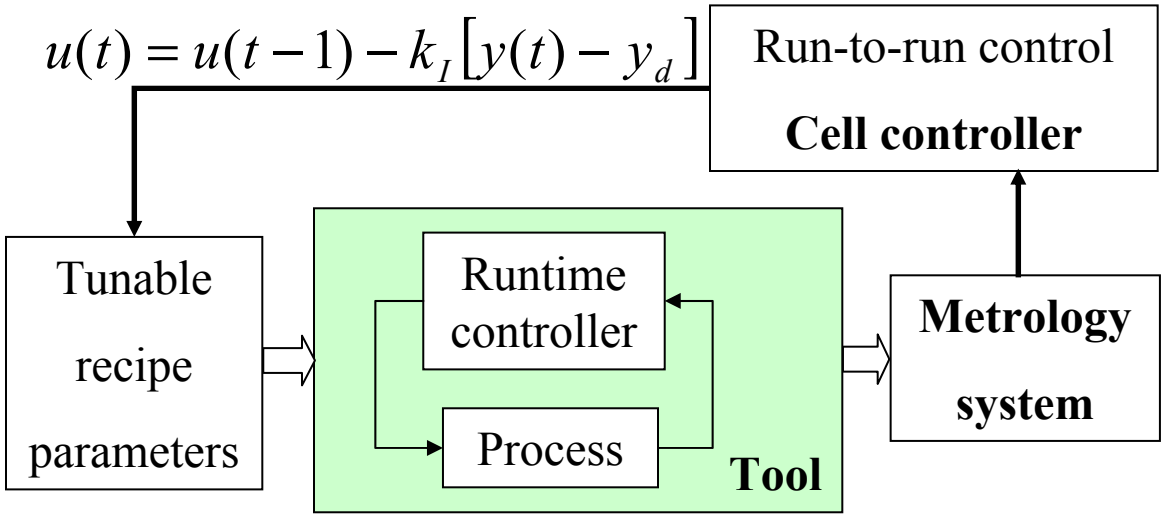
- Main APC (Advanced Process Control) approach in semiconductor processes
- Modification of a product recipe between tool "runs"



- Processes:
  - vapor phase epitaxy
  - lithography
  - chemical mechanical planarization (CMP)
  - plasma etch

$$y(t) = g \cdot u(t) + d(t)$$

$$u(t) = u(t-1) - k_I \left[y(t) - y_d \right]$$



PI control

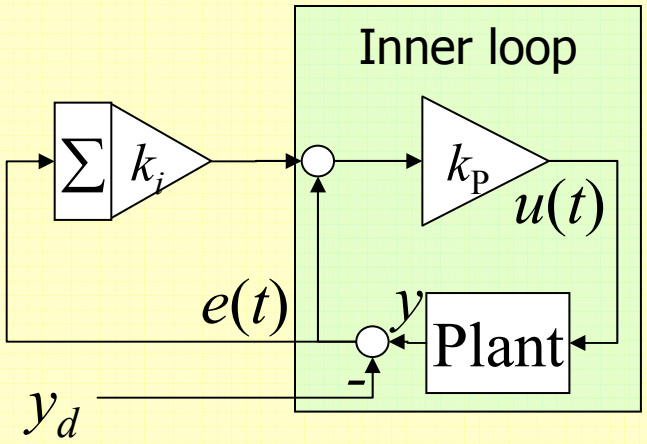
- P Control + Integrator for cancelling steady state error

$$e = y - y_d$$
$$v(t+1) = v(t) + e(t)$$
$$u(t) = -k_I v(t) - k_P e(t)$$

- Velocity form of the controller

$$u(t) = u(t-1) - k_I e(t) - k_P [e(t) - e(t-1)]$$

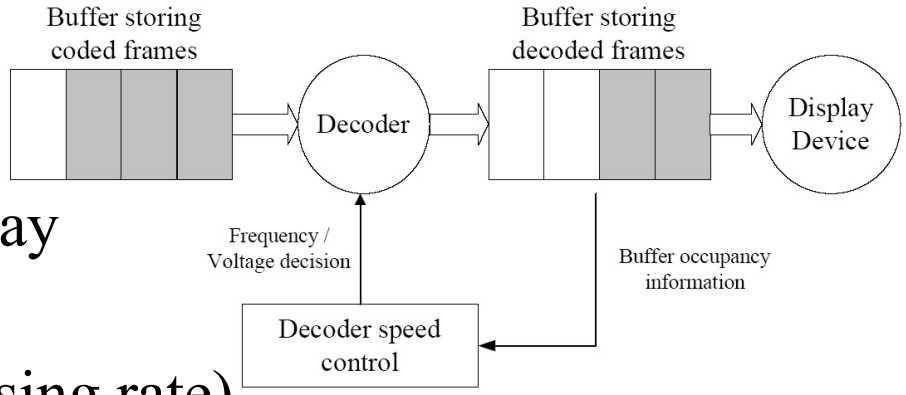
Cascade loop interpretation:



$$k_I = k_i \cdot k_P$$

QoS Control Example

- QoS control for video server
- QoS = Quality of Service
- Need to maintain constant delay
  - for audio synchronization etc
- Control codec quality (processing rate)
- Regulate buffer queue
- Model:



Assume $T = 1$

Rate $\quad r(t+1) = u(t)$

Buffer $\quad b(t+1) = b(t) - Tr(t) + w(t)$

$$x(k+1) = \underbrace{\begin{bmatrix} 0 & 0 \\ -1 & 1 \end{bmatrix}}_{A} \cdot x(k) + \underbrace{\begin{bmatrix} -1 \\ 0 \end{bmatrix}}_{B} \cdot u(k) + \underbrace{\begin{bmatrix} 0 \\ 1 \end{bmatrix}}_{F} \cdot w(k)$$

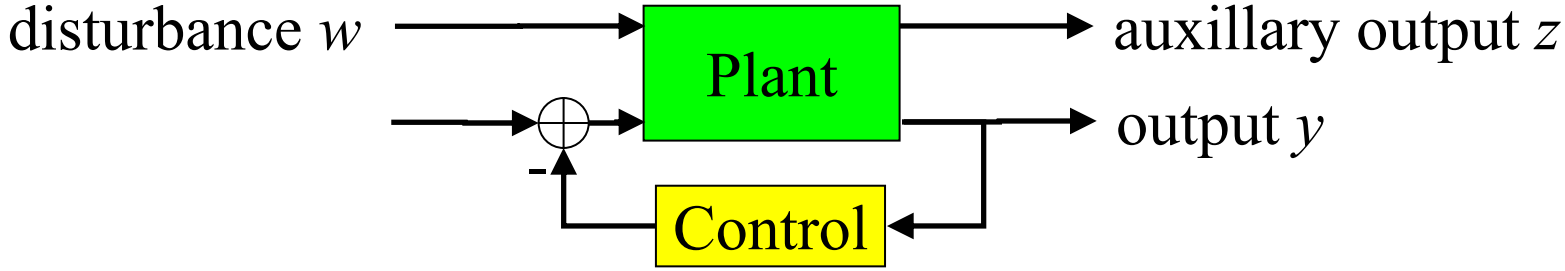
$$y(k) = \underbrace{\begin{bmatrix} 0 & 1 \end{bmatrix}}_{C} \cdot x(k)$$

QoS Control Example, cont'd

- Dominant integrator dynamics, use P control

$$x(k+1) = Ax(k) + Bu(k) + Fw(k) \qquad u(k) = -k_P\left(y(k) - y_d\right)$$

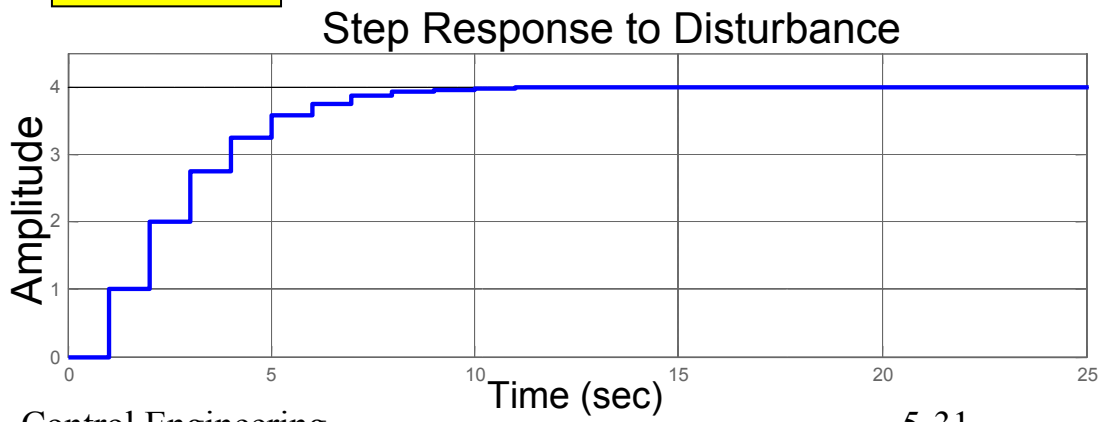
$$y(k) = Cx(k)$$



Matlab command:

```
S=feedback(P,C,iI,iO);
```

Use `tf` to convert model



QoS Control Example, cont'd

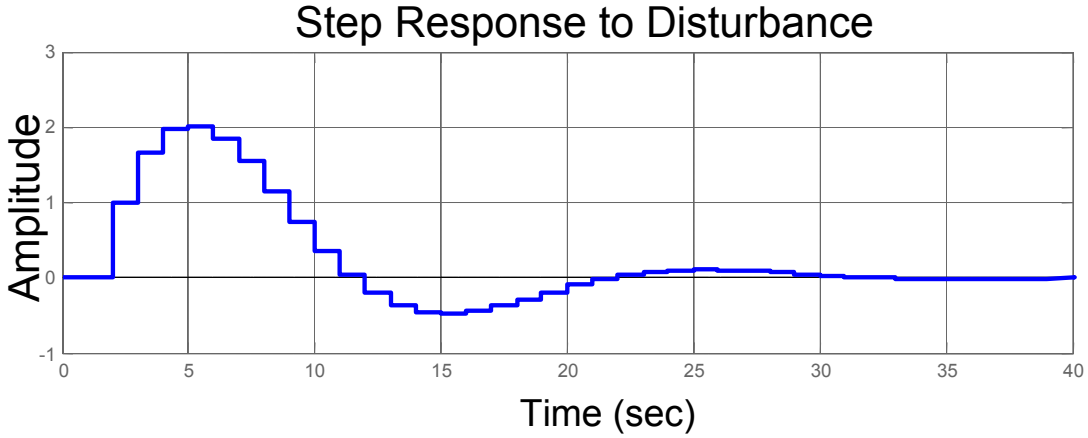
- P control provides poor disturbance rejection
- Use PI control

$$e(k) = y(k) - y_d$$

$$u(k) = u(k-1) - k_I e(k) - k_P [e(k) - e(k-1)]$$

- PI has better disturbance rejection

$$u = -\left[k_P + \frac{k_I z}{z-1}\right] \cdot e$$



PID Control

- PID: three-term control

$$e = y - y_d$$

$$u = -k_P e - k_I \int e \cdot dt - k_D \dot{e}$$

- Sampled-time PID

$$u = -k_P e - k_I \frac{1}{1 - z^{-1}} e - k_D \left(1 - z^{-1}\right) \cdot e$$

present *past* *future*

- PID controller in velocity form
  - bumpless transfer between manual and automatic

$$\Delta u = -k_D \Delta^2 e - k_P \Delta e - k_I e$$

$$\Delta = 1 - z^{-1}$$

$$u(t) = u(t-1) - k_I e(t) - k_P \left[e(t) - e(t-1)\right] - k_D \left[e(t) - 2e(t-1) + e(t-2)\right]$$

PID Controller Tuning

- Tune continuous-time PID controller, e.g. by Ziegler-Nichols rule, and set up the sampled time PID to approximate the continuous-time PID
- Cascaded loop design (continuous time structure)
  - Design P
  - Cascade I, retune P
  - Add D, retune PI
- Optimize the performance parameters by repeated simulation runs – search through the {P, I, D} space
- Loopshaping – Lectures 7-8
- Advanced control design – formal specs, other courses

Industrial PID Controller

- A box, not an algorithm
- Auto-tuning functionality:
  - pre-tune
  - self-tune
- Manual/cascade mode switch
- Bumpless transfer between different modes, setpoint ramp
- Loop alarms
- Networked or serial port connection

Plant Type

- Constant gain - I control
- Integrator - P control
- First order system - PI control
- Double integrator or second order system - PD control
- Generic response with delay - PID control