Digital Control

ELEC 3004: **Digital Linear Systems**: Signals & Controls
Dr. Surya Singh

Lecture 9

elec3004@itee.uq.edu.au
http://robotics.itee.uq.edu.au/~elec3004/

May 6, 2014

http://elec3004.com

Lecture Schedule:

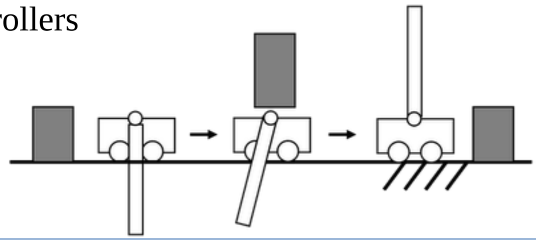
Week	Date	Lecture Title
1	4-Mar	Introduction & Systems Overview
	6-Mar	[Linear Dynamical Systems]
2	11-Mar	Signals as Vectors & Systems as Maps
	13-Mar	[Signals]
3	18-Mar	Sampling & Data Acquisition & Antialiasing Filters
	20-Mar	[Sampling]
4	25-Mar	System Analysis & Convolution
	27-Mar	[Convolution & FT]
5	1-Apr	Frequency Response & Filter Analysis
	3-Apr	[Filters]
6	8-Apr	Discrete Systems & Z-Transforms
	10-Apr	[Z-Transforms]
7	15-Apr	Introduction to Digital Control
	17-Apr	[Feedback]
8	29-Apr	Digital Filters
	1-May	[Digital Filters]
9	6-May	Digital Control Design
	8-May	[Digitial Control]
10	13-May	Stability of Digital Systems
	15-May	[Stability]
11	20-May	State-Space
	22-May	Controllability & Observability
12	27-May	PID Control & System Identification
	29-May	Digitial Control System Hardware
13	3-Jun	Applications in Industry & Information Theory & Communications
	5-Jun	Summary and Course Review

Digital control

Once upon a time…

- Electromechanical systems were controlled by electromechanical compensators
  - Mechanical flywheel governors, capacitors, inductors, resistors, relays, valves, solenoids (fun!)
  - But also complex and sensitive!

- Humans developed sophisticated tools for designing reliable analog controllers



Digital control

Once upon a time…

- Electromechanical systems were controlled by electromechanical compensators
  - Mechanical flywheel governors, capacitors, inductors, resistors, relays, valves, solenoids (fun!)
  - But also complex and sensitive!

➔Idea: Digital computers in real-time control
  - Transform approach (classical control)
    - Root-locus methods (pretty much the same as METR 3200)
    - Bode's frequency response methods (these change compared to METR 3200)
  - State-space approach (modern control)

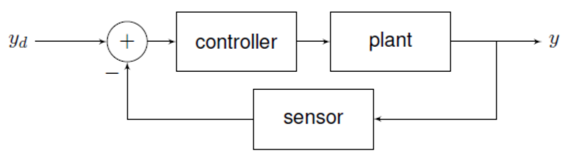
→ Model Making: Control of frequency response as well as Least Squares Parameter Estimation

Many advantages

- Practical improvement over analog control:
  - **Flexible**; reprogrammable to implement different control laws for different systems
  - **Adaptable**; control algorithms can be changed on-line, during operation
  - **Insensitive** to environmental conditions; (heat, EMI, vibration, etc)
  - **Compact**; handful of components on a PCB
  - **Cheap**

Feedback Control

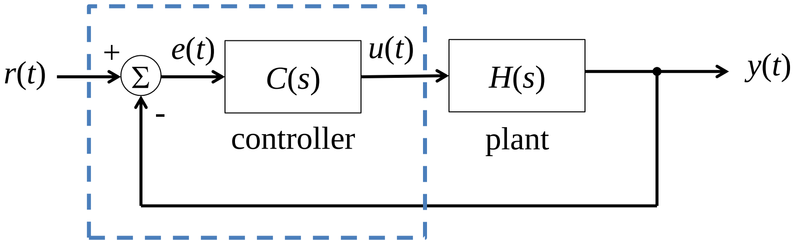
(Simple) control systems have three parts:



- The plant is the system to be controlled (e.g. the robot).
- The sensor measures the output of the plant.
- The controller sends an input command to the plant based on the difference from the actual output and the desired output.

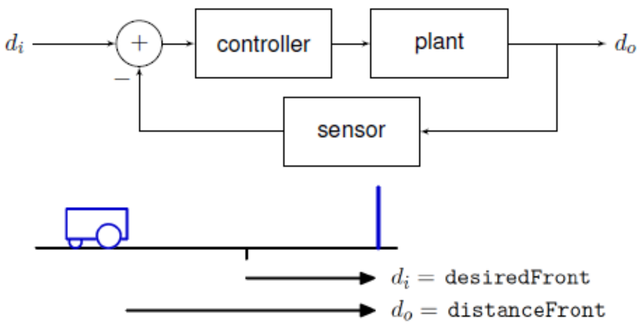
Archetypical control system

- Consider a continuous control system:



- The functions of the controller can be entirely represented by a discretised computer system

Simple Controller Goes Digital

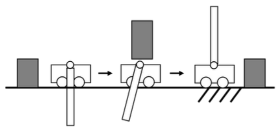


$$
\begin{aligned}
\text{plant:} \quad & y[n] = y[n-1] - Tu[n-1] \\
\text{sensor:} \quad & y[n] = u[n-1] \\
\text{controller:} \quad & y[n] = Ku[n]
\end{aligned}
$$

Complex system behaviors, depending on K

Digital Control

Wikipedia,
Cart and pole



$$L = \frac{1}{2}Mv_1^2 + \frac{1}{2}mv_2^2 - mg\ell\cos\theta$$

where v_1 is the velocity of the cart and v_2 is the velocity of the point mass m. v_1 and v_2 can be expressed in terms of x and θ by writing the velocity as the first derivative of the position;

$$v_1^2 = \dot{x}^2$$
$$v_2^2 = \left(\frac{d}{dt}(x - \ell\sin\theta)\right)^2 + \left(\frac{d}{dt}(\ell\cos\theta)\right)^2$$

Simplifying the expression for v_2 leads to:

$$v_2^2 = \dot{x}^2 - 2\ell\dot{x}\dot{\theta}\cos\theta + \ell^2\dot{\theta}^2$$

The Lagrangian is now given by:

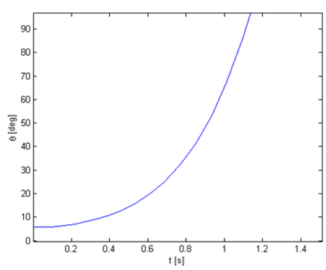
$$L = \frac{1}{2}(M+m)\dot{x}^2 - m\ell\dot{x}\dot{\theta}\cos\theta + \frac{1}{2}m\ell^2\dot{\theta}^2 - mg\ell\cos\theta$$

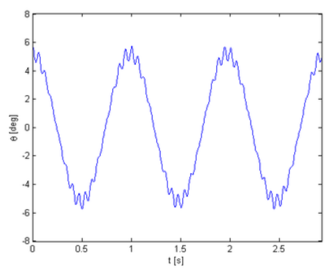
and the equations of motion are:

$$\frac{\mathrm{d}}{\mathrm{d}t}\frac{\partial L}{\partial\dot{x}} - \frac{\partial L}{\partial x} = F$$
$$\frac{\mathrm{d}}{\mathrm{d}t}\frac{\partial L}{\partial\dot{\theta}} - \frac{\partial L}{\partial\theta} = 0$$

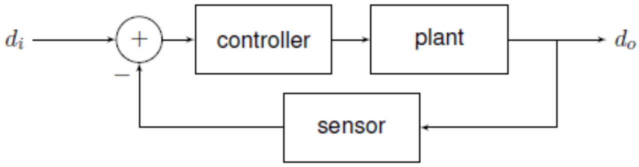
substituting L in these equations and simplifying leads to the equations that describe the motion o

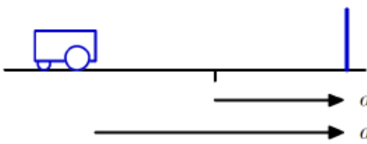
$$(M+m)\ddot{x} - m\ell\ddot{\theta}\cos\theta + m\ell\dot{\theta}^2\sin\theta = F$$
$$\ell\ddot{\theta} - g\sin\theta = \ddot{x}\cos\theta$$





Simple Controller Goes Digital



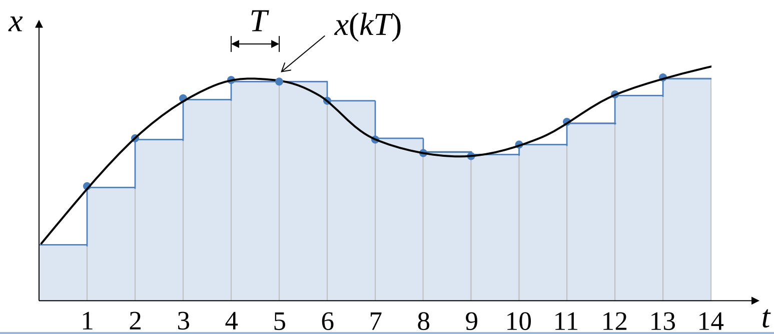


plant: $y[n] = y[n-1] - Tu[n-1]$
sensor: $y[n] = u[n-1]$
controller: $y[n] = Ku[n]$

Complex system behaviors, depending on K

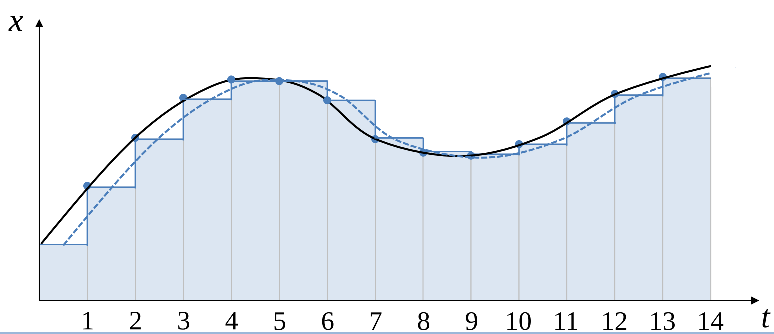
Return to the discrete domain

- Recall that continuous signals can be represented by a series of samples with period *T*



Zero Order Hold

- An output value of a synthesised signal is held constant until the next value is ready
  - This introduces an effective delay of $T/2$



How to Handle the Digitization?

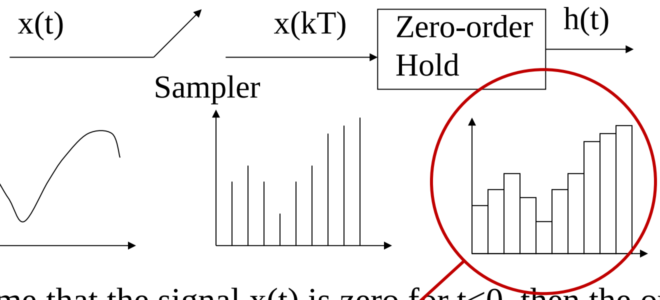
(z-Transforms)

The *z*-transform

- In practice, you'll use look-up tables or computer tools (ie. Matlab) to find the *z*-transform of your functions

$\boldsymbol{F(s)}$	$\boldsymbol{F(kt)}$	$\boldsymbol{F(z)}$
$\frac{1}{s}$	1	$\frac{z}{z-1}$
$\frac{1}{s^2}$	kT	$\frac{Tz}{(z-1)^2}$
$\frac{1}{s+a}$	e^{-akT}	$\frac{z}{z-e^{-aT}}$
$\frac{1}{(s+a)^2}$	kTe^{-akT}	$\frac{zTe^{-aT}}{(z-e^{-aT})^2}$
$\frac{1}{s^2+a^2}$	$\sin(akT)$	$\frac{z\sin aT}{z^2-(2\cos aT)z+1}$

Zero-order-hold (ZOH)



- Assume that the signal x(t) is zero for t<0, then the output h(t) is related to x(t) as follows:

$$h(t) = x(0)[1(t) - 1(t-T)] + x(T)[1(t-T) - 1(t-2T)] + \cdots$$
$$= \sum_{k=0}^{\infty} x(kT)[1(t - kT) - 1(t - (k+1)T)]$$

Transfer function of Zero-order-hold (ZOH)

- Recall the Laplace Transforms ($\mathcal{L}$) of:

$$\mathcal{L}[\delta(t)] = 1 \qquad \mathcal{L}[f(t-kT)] = F(s)e^{-kTs}$$
$$\mathcal{L}\,[\delta(t-kT)] = e^{-kTs} \qquad \mathcal{L}[1(t-kT)] = \frac{e^{-kTs}}{s}$$

- Thus the $\mathcal{L}$ of h(t) becomes:

$$\mathcal{L}[h(t)] = \mathcal{L}[\sum_{k=0}^{\infty} x(kT)[1(t - kT) - 1(t - (k+1)T)]]$$
$$= \sum_{k=0}^{\infty} x(kT)\mathcal{L}[1(t - kT) - 1(t - (k+1)T)] = \sum_{k=0}^{\infty} x(kT)[\frac{e^{-kTs}}{s} - \frac{e^{-(k+1)Ts}}{s}]$$
$$= \sum_{k=0}^{\infty} x(kT)\frac{e^{-kTs} - e^{-(k+1)Ts}}{s} = \sum_{k=0}^{\infty} x(kT)\frac{1-e^{-Ts}}{s}e^{-kTs} = \frac{1-e^{-Ts}}{s}\sum_{k=0}^{\infty} x(kT)e^{-kTs}$$

Transfer function of Zero-order-hold (ZOH)

… Continuing the $\mathcal{L}$ of h(t) …

$$\mathcal{L}[h(t)] = \mathcal{L}[\sum_{k=0}^{\infty} x(kT)[1(t - kT) - 1(t - (k+1)T)]]$$

$$= \sum_{k=0}^{\infty} x(kT)\mathcal{L}[1(t - kT) - 1(t - (k+1)T)] = \sum_{k=0}^{\infty} x(kT)[\frac{e^{-kTs}}{s} - \frac{e^{-(k+1)Ts}}{s}]$$

$$= \sum_{k=0}^{\infty} x(kT)\frac{e^{-kTs} - e^{-(k+1)Ts}}{s} = \sum_{k=0}^{\infty} x(kT)\frac{1-e^{-Ts}}{s}e^{-kTs} = \frac{1-e^{-Ts}}{s}\sum_{k=0}^{\infty} x(kT)e^{-kTs}$$

$$\rightarrow X(s) = \mathcal{L}\left[\sum_{k=0}^{\infty} x(kT)\delta(t-kT)\right] = \sum_{k=0}^{\infty} x(kT)e^{-kTs}$$

$$\therefore H(s) = \mathcal{L}[h(t)] = \frac{1-e^{-Ts}}{s}\sum_{k=0}^{\infty} x(kT)e^{-kTs} = \frac{1-e^{-Ts}}{s}X(s)$$

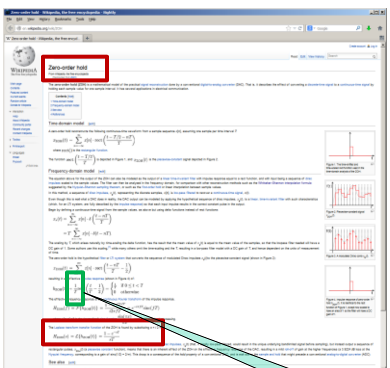
➔ Thus, giving the transfer function as:

$$\boxed{G_{\mathrm{ZOH}}(s) = \frac{H(s)}{X(s)} = \frac{1-e^{-Ts}}{s}} \xrightarrow{\mathcal{Z}} \boxed{G_{ZOH}(z) = \frac{\left(1-e^{-aT}\right)}{z-e^{-aT}}}$$

$\mathcal{L}$(ZOH)=??? : What is it?

$$\frac{1-e^{-Ts}}{Ts}$$

- Wikipedia



The "difference" is that Wikipedia pre-multiplies by 1/T

$$\frac{1-e^{-Ts}}{s}$$

- Lathi
- Franklin, Powell, Workman
- Franklin, Powell, Emani-Naeini
- Dorf & Bishop
- Oxford Discrete Systems: (Mark Cannon)
- MIT 6.002 (Russ Tedrake)
- Matlab

Proof!

Coping with complexity

- Transfer functions help control complexity
  - Recall the Laplace transform:

$$\mathcal{L}\{f(t)\} = \int_0^\infty f(t)e^{-st}dt = F(s)$$

where

$$\mathcal{L}\{\dot{f}(t)\} = sF(s)$$

$$x(t) \longrightarrow \boxed{H(s)} \longrightarrow y(t)$$

Is there a something similar for sampled systems?

The *z*-transform

- The discrete equivalent is the *z*-Transform†:

$$\mathcal{Z}\{f(k)\} = \sum_{k=0}^{\infty} f(k)z^{-k} = F(z)$$

and

$$\mathcal{Z}\{f(k-1)\} = z^{-1}F(z)$$

$$x(k) \longrightarrow \boxed{F(z)} \longrightarrow y(k)$$

Convenient!

†This is not an approximation, but approximations are easier to derive

The z-transform

- Some useful properties
  - **Delay by $\boldsymbol{n}$ samples**: $\mathcal{Z}\{f(k-n)\} = z^{-n}F(z)$
  - **Linear**: $\mathcal{Z}\{af(k) + bg(k)\} = \text{a}F(z) + bG(z)$
  - **Convolution**: $\mathcal{Z}\{f(k) * g(k)\} = F(z)G(z)$

So, all those block diagram manipulation tools you know and love will work just the same!

The z-transform

- In practice, you'll use look-up tables or computer tools (ie. Matlab) to find the z-transform of your functions

$\boldsymbol{F(s)}$	$\boldsymbol{F(kt)}$	$\boldsymbol{F(z)}$
$\frac{1}{s}$	1	$\frac{z}{z-1}$
$\frac{1}{s^2}$	kT	$\frac{Tz}{(z-1)^2}$
$\frac{1}{s+a}$	e^{-akT}	$\frac{z}{z-e^{-aT}}$
$\frac{1}{(s+a)^2}$	kTe^{-akT}	$\frac{zTe^{-aT}}{(z-e^{-aT})^2}$
$\frac{1}{s^2+a^2}$	$\sin(akT)$	$\frac{z\sin aT}{z^2-(2\cos aT)z+1}$

Why z-Transform

- Makes it easy to analyse feedback systems governed by difference equations
- For any complex number $z = re^{j\omega}$ $y[n] \stackrel{\mathcal{Z}}{\longleftrightarrow} Y(z)$
- Forward Analysis: $Y(z) = \sum_{k=-\infty}^{\infty} h[k] z^{-k}$
- Backward Synthesis
  (for any fixed r > 0 on which the Z-transform converges):

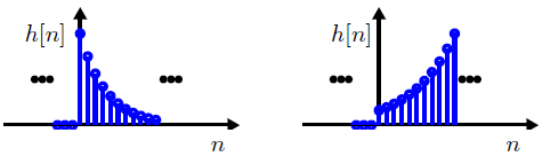
$$y[n] = \frac{1}{2\pi} \int_{2\pi} Y(re^{j\omega})(re^{j\omega})^n d\omega$$

z-Transforms for Difference Equations

- First-order linear constant coefficient difference equation:

First-order linear constant coefficient difference equation:

$$y[n] = ay[n-1] + bu[n]$$



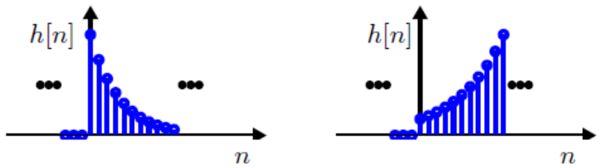
$$h[n] = \begin{cases} ba^n & n \geq 0, \\ 0 & \text{otherwise.} \end{cases}$$

$$H(z) = \sum_{k=0}^{\infty} ba^k z^{-k} = b \sum_{k=0}^{\infty} \left(\frac{a}{z}\right)^k = \frac{b}{1 - az^{-1}}, \quad \text{when } |z| > |a|.$$

z-Transforms for Difference Equations

First-order linear constant coefficient difference equation:

$$y[n] = ay[n-1] + bu[n]$$

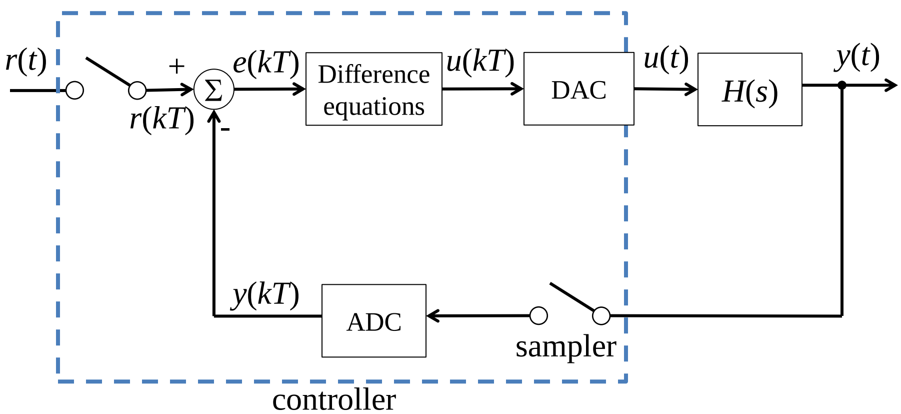


$$y[n] - ay[n-1] = bu[n]$$
$$\updownarrow$$
$$Y(z) - az^{-1}Y(z) = bU(z)$$

$$H(z) = \frac{Y(z)}{U(z)} = \frac{b}{1 - az^{-1}},$$ when does it converge?

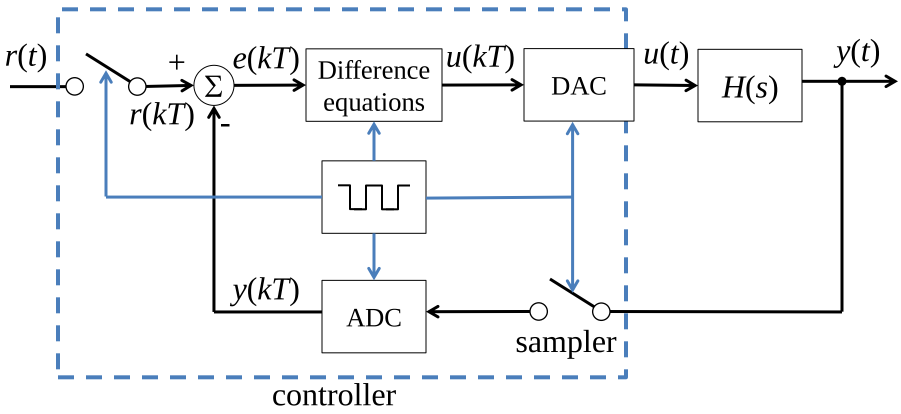
Digitisation

- Continuous signals sampled with period T
- kth control value computed at $t_k = kT$



Digitisation

- Continuous signals sampled with period T
- kth control value computed at $t_k = kT$



Difference equations

- How to represent differential equations in a computer? Difference equations!
- The output of a difference equation system is a function of current and previous values of the input and output:

$$y(t_k) = D(x(t_k), x(t_{k-1}), \ldots, x(t_{k-n}), y(t_{k-1}), \ldots, y(t_{k-n}))$$

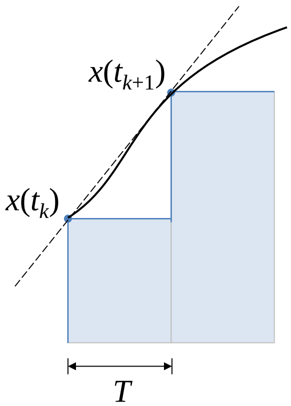
  – We can think of x and y as parameterised in k

  Useful shorthand: $x(t_{k+i}) \equiv x(k+i)$

Euler's method*

- Dynamic systems can be approximated† by recognising that:

$$\dot{x} \cong \frac{x(k+1) - x(k)}{T}$$



- As $T \rightarrow 0$, approximation error approaches 0

*Also known as the forward rectangle rule
†Just an approximation – more on this later

An example!

Convert the system $\frac{Y(s)}{X(s)} = \frac{s+2}{s+1}$ into a difference equation with period T, using Euler's method.

1. Rewrite the function as a dynamic system:

$$sY(s) + Y(s) = sX(s) + 2X(s)$$

Apply inverse Laplace transform:

$$\dot{y}(t) + y(t) = \dot{x}(t) + 2x(t)$$

2. Replace continuous signals with their sampled domain equivalents, and differentials with the approximating function

$$\frac{y(k+1) - y(k)}{T} + y(k) = \frac{x(k+1) - x(k)}{T} + 2x(k)$$

An example!

Simplify:

$$y(k+1) - y(k) + Ty(k) = x(k+1) - x(k) + 2Tx(k)$$
$$y(k+1) + (T-1)y(k) = x(k+1) + (2T-1)x(k)$$

$$y(k+1) = x(k+1) + (2T-1)x(k) - (T-1)y(k)$$

We can implement this in a computer.

Cool, let's try it!

Back to the future

A quick note on causality:

- Calculating the "(k+1)th" value of a signal using

$$y(k+1) = x(k+1) + Ax(k) - By(k)$$

relies on also knowing the next (future) value of $x(t)$.

future value current values

(this requires very advanced technology!)

- Real systems always run with a delay:

$$y(k) = x(k) + Ax(k-1) - By(k-1)$$

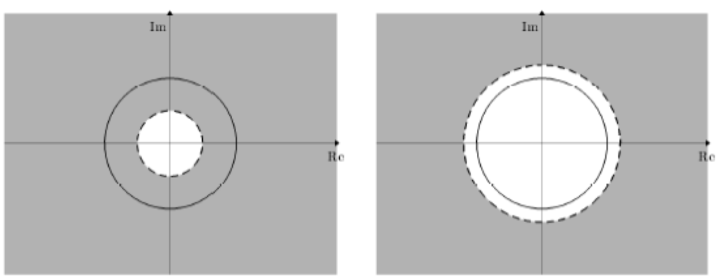
Back to the example!

```
T = 0.02;  //period of 50 Hz, a number pulled from thin air
A = 2*T-1; //pre-calculated control constants
B = T-1;

...

while(1)
   {
      if(interrupt_flag)          //this triggers every 20 ms
      {
         x0 = x;                  //save previous values
         y0 = y;
         x = update_input();      //get latest x value
         y = x + A*x0 - B*y0;     //do the difference equations
         update_output(y);        //write out current value
      }
   }
```

(The actual maths bit)

Region of Convergence (ROC) Plots

$$H(z) = \frac{Y(z)}{U(z)} = \frac{b}{1 - az^{-1}}, \quad |z| > |a|$$



$a = .5$ $a = 1.2$

Properties of the ROC

➔The ROC is always defined by circles centered around the origin.

$h[k]r^{-k}$ is absolutely summable, where $r = |z|$.

➔Right-sided signals have "outsided" ROCs.

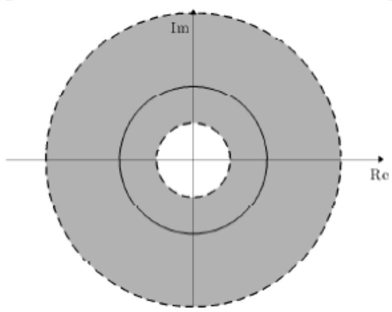
if $\exists n_0$ such that $h[n] = 0\ \forall n < n_0$, then if $r_0 \in$ ROC, then $\forall r$ with $r_0 < r < \infty$ are also in the ROC.

➔Left-sided signals have "insided" ROCs. (with ∀r within $0<r<r_0$)

Combinations of Signals

$$y_1[n] = \begin{cases} ba^n & n \geq 0 \\ 0 & n < 0 \end{cases} \qquad y_2[n] = \begin{cases} 0 & n \geq 0 \\ -ba^n & n < 0 \end{cases}$$

$a = .5$ $\qquad a = 2$



ROC for $\alpha_1 y_1[n] + \alpha_2 y_2[n]$

Higher-order difference equations

$$y[n] = a_1 y[n-1] + a_2 y[n-2] + a_3 y[n-3] + b_0 u[n] + b_1 u[n-1] + \ldots$$

Easy to take the Z-transform

$$Y(z) = a_1 z^{-1} Y(z) + a_2 z^{-2} Y(z) + a_3 z^{-3} Y(z) + b_0 U(z) + \ldots$$

$$H(z) = \frac{b_0 + b_1 z^{-1} + b_2 z^{-1} + \ldots}{1 - a_1 z^{-1} - a_2 z^{-2} - a_3 z^{-3} + \ldots}$$

Final value theorem

- An important question: what is the steady-state output a stable system at $t = \infty$?
  - For continuous systems, this is found by:
    $$\lim_{t \to \infty} x(t) = \lim_{s \to 0} sX(s)$$
  - The discrete equivalent is:
    $$\lim_{k \to \infty} x(k) = \lim_{z \to 1} (1 - z^{-1}) X(z)$$

(Provided the system is stable)

An example!

- Back to our difference equation:

$$y(k) = x(k) + Ax(k-1) - By(k-1)$$

becomes

$$Y(z) = X(z) + Az^{-1}X(z) - Bz^{-1}Y(z)$$
$$(z+B)Y(z) = (z+A)X(z)$$

which yields the transfer function:

$$\frac{Y(z)}{X(z)} = \frac{z+A}{z+B}$$

Note: It is also not uncommon to see systems expressed as polynomials in z^{-n}

This looks familiar…

- Compare:

$$\frac{\mathrm{Y(s)}}{X(s)} = \frac{s+2}{s+1} \text{ vs } \frac{Y(z)}{X(z)} = \frac{z+A}{z+B}$$

How are the Laplace and *z* domain representations related?

Consider the simplest system

- Take a first-order response:

$$f(t) = e^{-at} \Rightarrow \mathcal{L}\{f(t)\} = \frac{1}{s+a}$$

- The discrete version is:

$$f(kT) = e^{-akT} \Rightarrow \mathcal{Z}\{f(k)\} = \frac{z}{z - e^{-aT}}$$

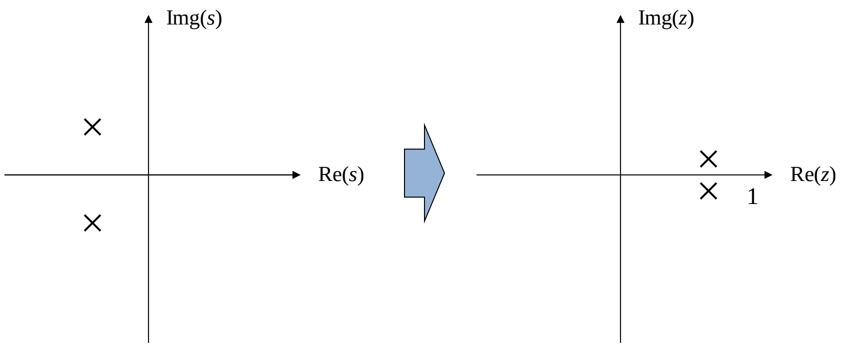
The equivalent system poles are related by

$$z = e^{sT}$$

That sounds somewhat profound… but what does it mean?

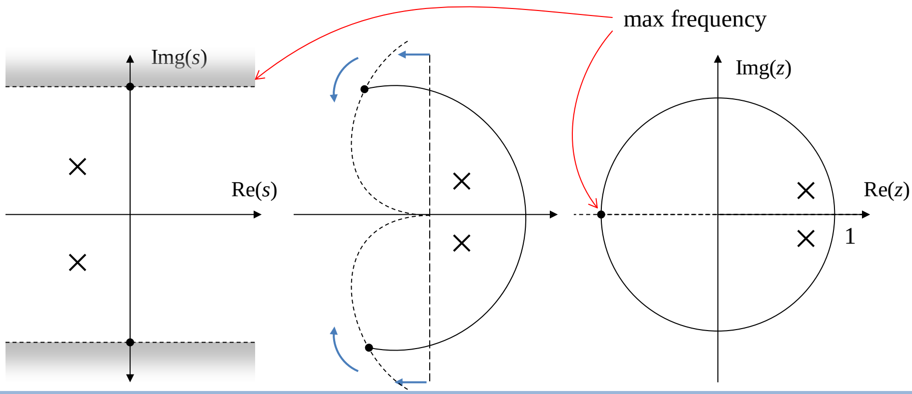
The z-Plane

- z-domain poles and zeros can be plotted just like s-domain poles and zeros:



Deep insight #1

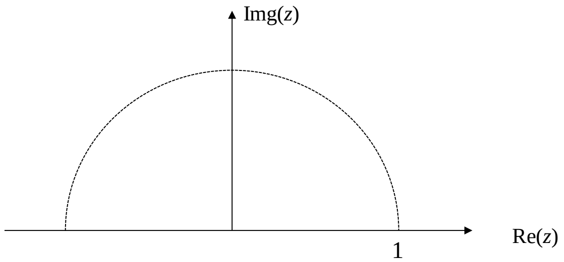
The mapping between continuous and discrete poles and zeros acts like a distortion of the plane



The *z*-plane

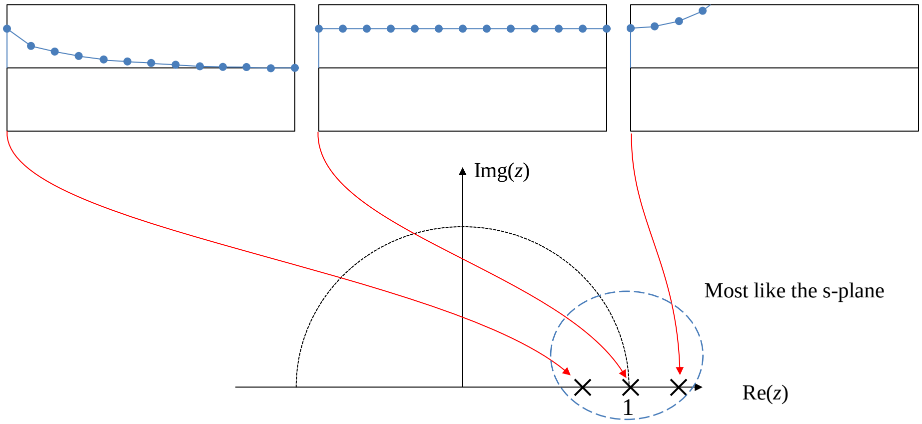
- We can understand system response by pole location in the *z*-plane

[Adapted from Franklin, Powell and Emami-Naeini]



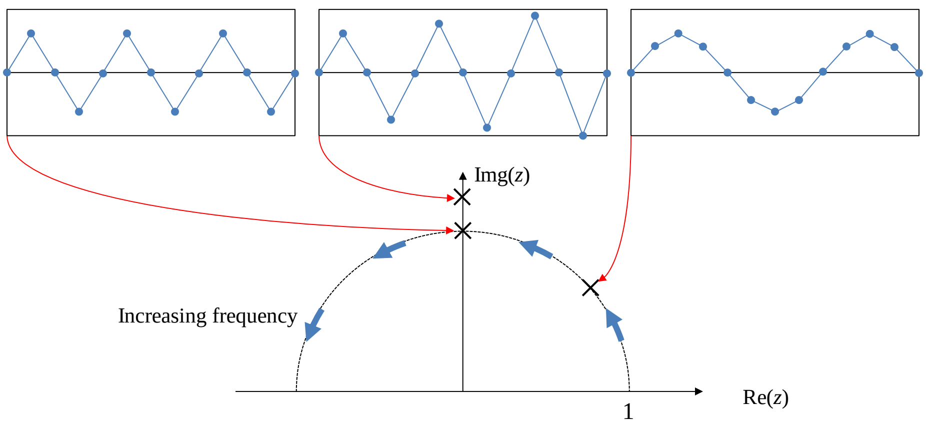
Effect of pole positions

- We can understand system response by pole location in the z-plane



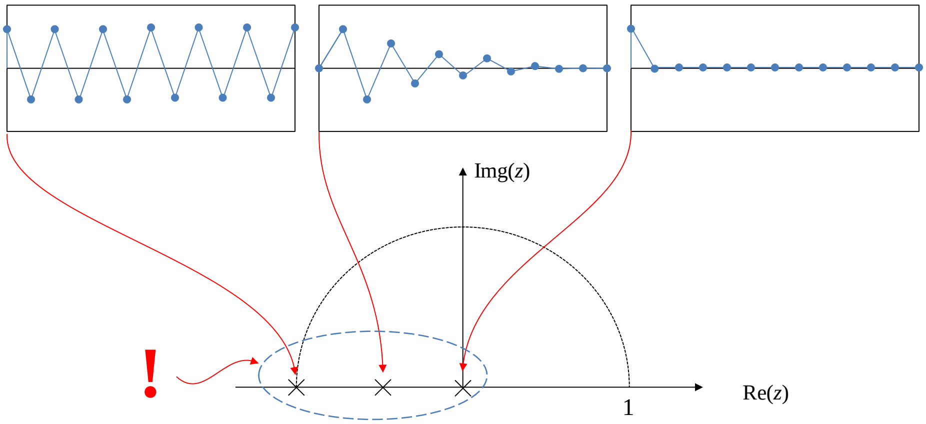
Effect of pole positions

- We can understand system response by pole location in the z-plane



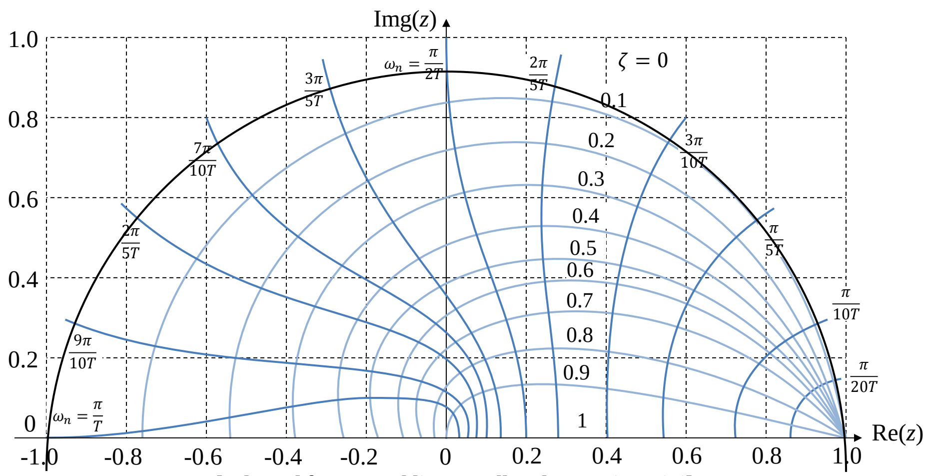
Effect of pole positions

- We can understand system response by pole location in the z-plane



Damping and natural frequency

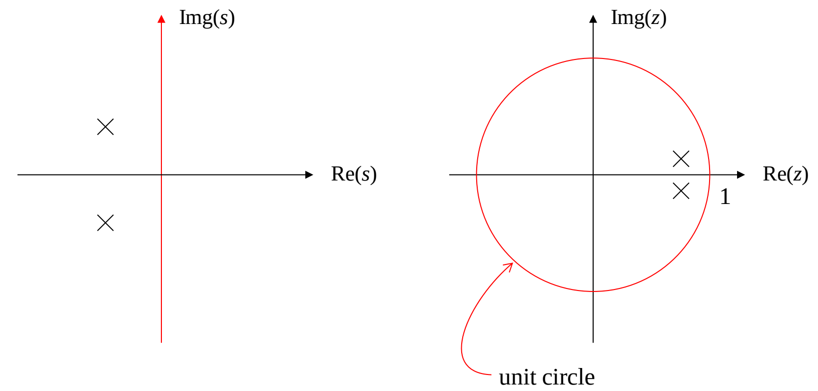
$$z = e^{sT} \text{ where } s = -\zeta\omega_n \pm j\omega_n\sqrt{1-\zeta^2}$$



[Adapted from Franklin, Powell and Emami-Naeini]

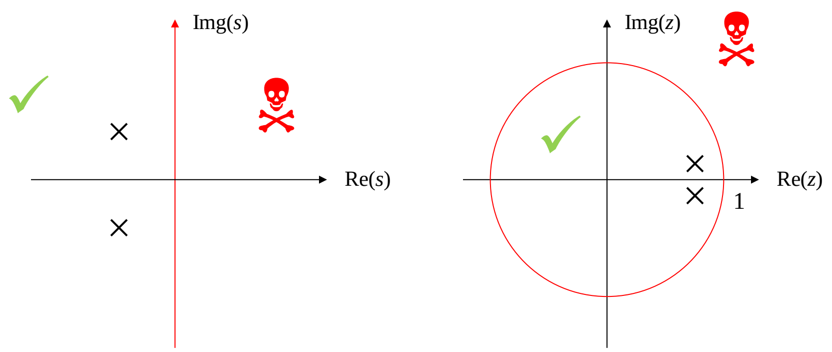
z-plane stability

- In the z-domain, the unit circle is the system stability bound



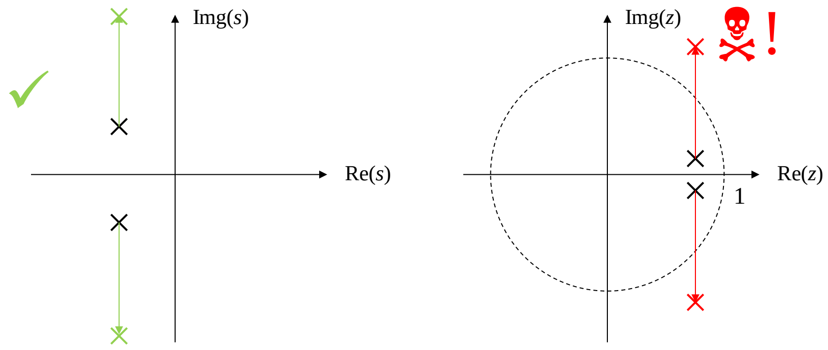
z-plane stability

- In the z-domain, the unit circle is the system stability bound



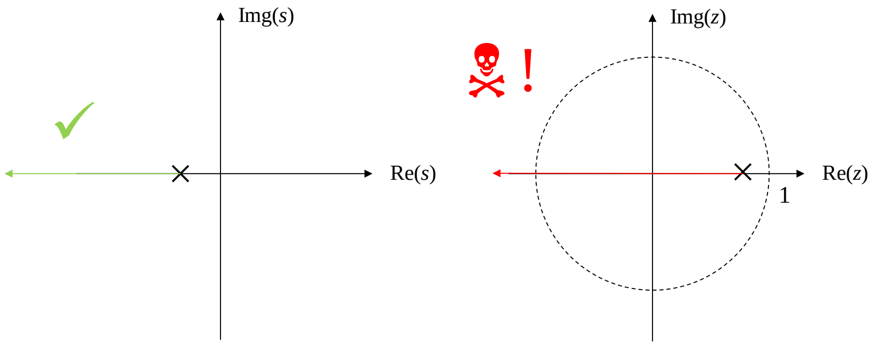
z-plane stability

- The z-plane root-locus in closed loop feedback behaves just like the s-plane:



Deep insight #2

Gains that stabilise continuous systems can actually *destabilise* digital systems!



Example:

- Is this system stable?

$$u(k) = 0.9u(k-1) - 0.2u(k-2)$$

- Time-shift it:

$$u(k+2) = 0.9u(k+1) - 0.2u(k)$$

- z-Transform:

$$(1)z^2 - 0.9z + 0.2 = 0$$

- Characteristic Roots:
  z=0.5, z=0.4 ➔ STABLE!