Digital Control: Stability

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

Lecture 10

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

May 13, 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

Announcements:

- Lab 3: Next week (Weeks 10-11)
- PS 3: Due Tomorrow
- PS 2: Initial Finalization done
- ELEC 7312 Students:
  - Final Paper Review Due 23/6/2014 by 11:59pm
  - Details to be out later this week
- Final Exam:
  - 15 Questions (60% Short Answer, 40% Regular Problems)
  - 3 Hours
  - Closed-book
  - Yes, it has an unexpected twist at the end, but you'll like it. ☺
- Final Exam Logistics:
  - Saturday 21/6/2014 at 4:30pm
  - Location: TBA
  - Should we do this METR4202 Style?

PS 2 Bloopers

Ummm... 5?

Answer for (answered 2 times, estimated grade: 10 / 10):

Marker	Grade	Comment	Confidence
(Peer)	2 / 10	No effort is seem. ...	X / 5
(Peer)	0 / 10	Need more effort...	5 / 5
(Peer)	0 / 10	...	5 / 5
(Tutor)	0 / 10	...	-

Recap from METR4201 / Paul's Lecture: Some cases for control design

The system…

- Isn't fast enough
- Isn't damped enough
- Overshoots too much
- Requires too much control action

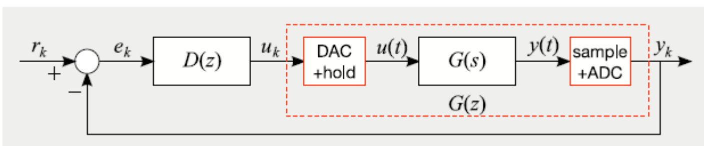
("Performance")

- Attempts to spontaneously disassemble itself

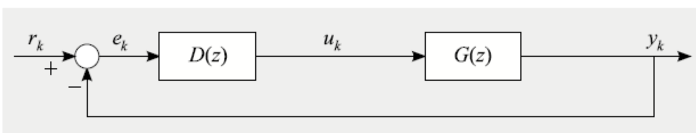
("Stability")

Designing in the Purely Discrete…

Analyse/design a discrete controller $D(z)$:



by considering the purely discrete time system:



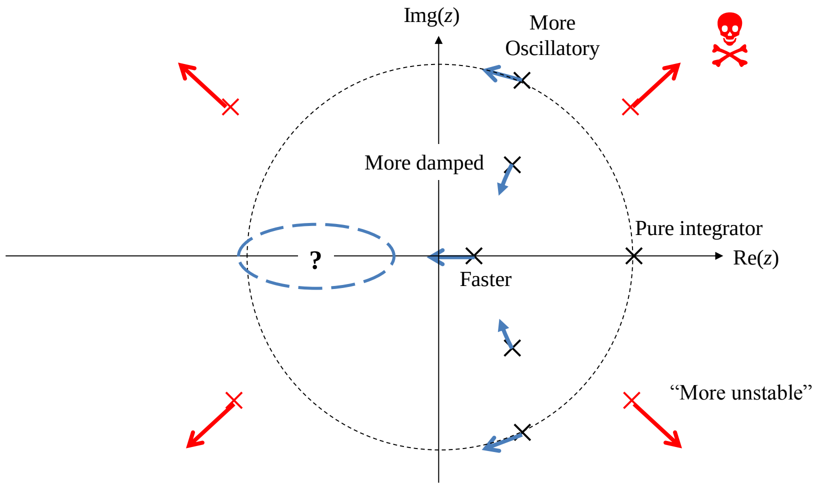
Closed loop system tranfer function: $\dfrac{Y(z)}{R(z)} = \dfrac{G(z)D(z)}{1 + G(z)D(z)}$

How do the closed loop poles relate to $\rightarrow$ stability?

$\rightarrow$ performance?

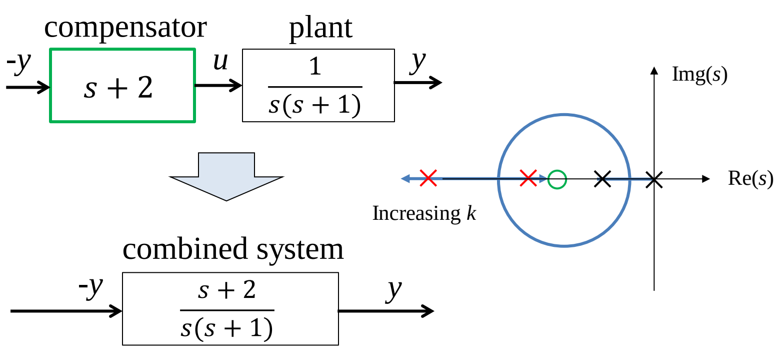
Recall dynamic responses

- Ditto the z-plane:



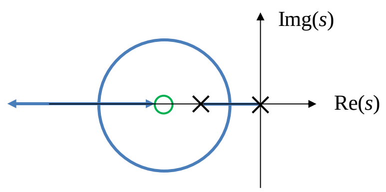
Dynamic compensation

- We can do more than just apply gain!
  - We can add dynamics into the controller that alter the open-loop response



But what dynamics to add?

- Recognise the following:
  - A root locus starts at poles, terminates at zeros
  - “Holes eat poles”
  - Closely matched pole and zero dynamics cancel
  - The locus is on the real axis to the left of an odd number of poles (treat zeros as ‘negative’ poles)



Some standard approaches

- Control engineers have developed time-tested strategies for building compensators
- Three classical control structures:
  - Lead
  - Lag
  - Proportional-Integral-Derivative (PID)
    (and its variations: P, I, PI, PD)

How do they work?

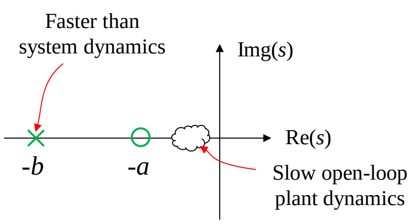
Lead/lag compensation

- Serve different purposes, but have a similar dynamic structure:

$$D(s) = \frac{s+a}{s+b}$$

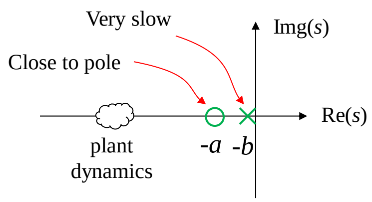
Note:
Lead-lag compensators come from the days when control engineers cared about constructing controllers from networks of op amps using frequency-phase methods. These days pretty much everybody uses PID, but you should at least know what the heck they are in case someone asks.

Lead compensation: $a < b$



- Acts to decrease rise-time and overshoot
  - Zero draws poles to the left; adds phase-lead
  - Pole decreases noise
- Set a near desired ω_n; set b at ~3 to 20x a

Lag compensation: $a > b$



- Improves steady-state tracking
  - Near pole-zero cancellation; adds phase-lag
  - Doesn't break dynamic response (too much)
- Set b near origin; set a at ~3 to 10x b

PID – the Good Stuff

- Proportional-Integral-Derivative control is the control engineer's hammer*
  - For P,PI,PD, etc. just remove one or more terms

$$C(s) = k\left(\underbrace{1}_{\text{Proportional}} + \underbrace{\frac{1}{\tau i s}}_{\text{Integral}} + \underbrace{\tau d s}_{\text{Derivative}}\right)$$

*Everything is a nail. That's why it's called "Bang-Bang" Control ☺

PID – the Good Stuff

- PID control performance is driven by three parameters:
  - k: system gain
  - τ_i: integral time-constant
  - τ_d: derivative time-constant

You're already familiar with the effect of gain.
What about the other two?

Integral

- Integral applies control action based on accumulated output error
  - Almost always found with P control
- Increase dynamic order of signal tracking
  - Step disturbance steady-state error goes to zero
  - Ramp disturbance steady-state error goes to a constant offset

Let's try it!

Integral

- Consider a first order system with a constant load disturbance, w; (recall as $t \to \infty$, $s \to 0$)

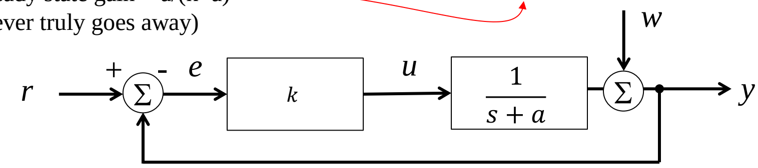
$$y = k\frac{1}{s+a}(r-y) + w$$

$$(s+a)y = k\,(r-y) + (s+a)w$$

$$(s+k+a)y = kr + (s+a)w$$

$$y = \frac{k}{s+k+a}r + \underbrace{\frac{(s+a)}{s+k+a}w}$$

Steady state gain = a/(k+a)
(never truly goes away)



Now with added integral action

$$y = k\left(1 + \frac{1}{\tau_i s}\right)\frac{1}{s+a}(r-y) + w$$

$$y = k\frac{s+\tau_i^{-1}}{s}\boxed{\frac{1}{s+a}}(r-y) + w$$

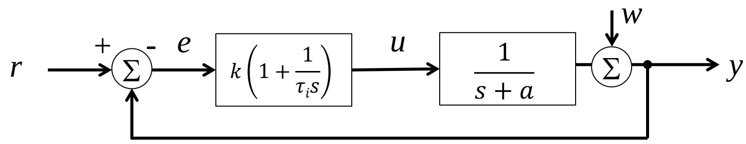
Same dynamics

$$s(s+a)y = k(s+\tau_i^{-1})(r-y) + s(s+a)w$$

$$(s^2 + (k+a)s + \tau_i^{-1})y = k(s+\tau_i^{-1})r + s(s+a)w$$

$$y = \frac{k(s+\tau_i^{-1})}{\boxed{(s^2+(k+a)s+\tau_i^{-1})}}r + \frac{\boxed{s(s+a)}}{k(s+\tau_i^{-1})}w$$

Must go to zero
for constant w!



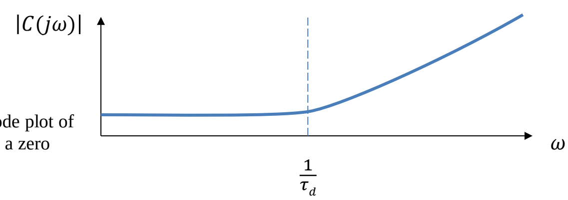
Derivative

- Derivative uses the rate of change of the error signal to anticipate control action
  - Increases system damping (when done right)
  - Can be thought of as ‘leading’ the output error, applying correction predictively
  - Almost always found with P control*

**What kind of system do you have if you use D, but don’t care about position?* Is it the same as P control in velocity space?

Derivative

- It is easy to see that PD control simply adds a zero at $s = -\frac{1}{\tau_d}$ with expected results
  - Decreases dynamic order of the system by 1
  - Absorbs a pole as $k \to \infty$
- Not all roses, though: derivative operators are sensitive to high-frequency noise



Bode plot of a zero

PID

- Collectively, PID provides two zeros plus a pole at the origin
  - Zeros provide phase lead
  - Pole provides steady-state tracking
  - Easy to implement in microprocessors
- Many tools exist for optimally tuning PID
  - Zeigler-Nichols
  - Cohen-Coon
  - Automatic software processes

Be alert

- If gains and time-constants are chosen poorly, all of these compensators can induce oscillation or instability.

- However, when used properly, PID can stabilise even very complex unstable third-order systems

Now in discrete

- Naturally, there are discrete analogs for each of these controller types:

Lead/lag: $\frac{1-\alpha z^{-1}}{1-\beta z^{-1}}$

PID: $k\left(1 + \frac{1}{\tau_i(1-z^{-1})} + \tau_d(1 - z^{-1})\right)$

But, where do we get the control design parameters from?
The s-domain?

Emulation vs Discrete Design

- Remember: polynomial algebra is the same, whatever symbol you are manipulating:

eg. $s^2 + 2s + 1 = (s + 1)^2$

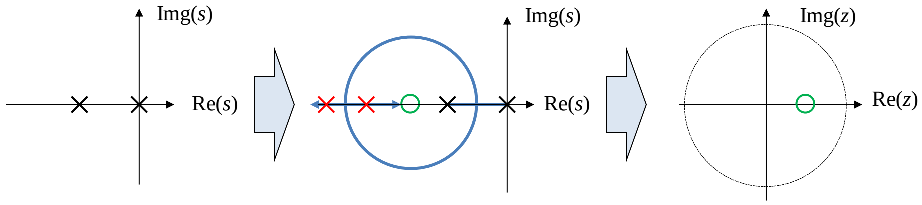
$z^2 + 2z + 1 = (z + 1)^2$

Root loci behave the same on both planes!

- Therefore, we have two choices:
  - Design in the s-domain and digitise (emulation)
  - Design only in the z-domain (discrete design)

Emulation design process

1. Derive the dynamic system model ODE
2. Convert it to a continuous transfer function
3. Design a continuous controller
4. Convert the controller to the z-domain
5. Implement difference equations in software



Emulation design process

- Handy rules of thumb:
  - Use a sampling period of 20 to 30 times faster than the closed-loop system bandwidth
  - Remember that the sampling ZOH induces an effective $T/2$ delay
  - There are several approximation techniques:
    - Euler's method
    - Tustin's method
    - Matched pole-zero
    - Modified matched pole-zero

Tustin's method

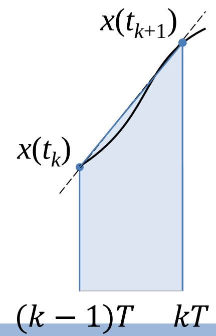
- Tustin uses a trapezoidal integration approximation (compare Euler's rectangles)
- Integral between two samples treated as a straight line:

$$u(kT) = \frac{T}{2}\left[x(k-1) + x(k)\right]$$

Taking the derivative, then z-transform yields:

$$s = \frac{2}{T}\frac{z-1}{z+1}$$

which can be substituted into continuous models



Matched pole-zero

- If $z = e^{sT}$, why can't we just make a direct substitution and go home?

$$\frac{Y(s)}{X(s)} = \frac{s+a}{s+b} \Rightarrow \frac{Y(z)}{X(z)} = \frac{z-e^{-aT}}{z-e^{-bT}}$$

- Kind of!
  - Still an approximation
  - Produces quasi-causal system (hard to compute)
  - Fortunately, also very easy to calculate.

Matched pole-zero

The process:

1. Replace continuous poles and zeros with discrete equivalents:

$$(s+a) \Rightarrow (z-e^{-aT})$$

2. Scale the discrete system DC gain to match the continuous system DC gain

3. If the order of the denominator is higher than the enumerator, multiply the numerator by $(z+1)$ until they are of equal order*

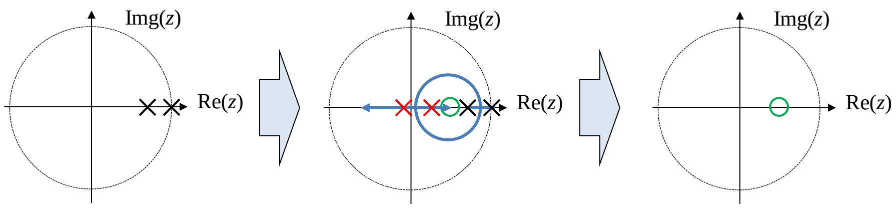
   * This introduces an averaging effect like Tustin's method

Modified matched pole-zero

- We're prefer it if we didn't require instant calculations to produce timely outputs
- Modify step 2 to leave the dynamic order of the numerator one less than the denominator
  - Can work with slower sample times, and at higher frequencies

Discrete design process

1. Derive the dynamic system model ODE
2. Convert it to a discrete transfer function
3. Design a digital compensator
4. Implement difference equations in software
5. Platypus Is Divine!



Discrete design process

- Handy rules of thumb:
  - Sample rates can be as low as twice the system bandwidth
    - but 5 to 10× for "stability"
    - 20 to 30 × for better performance
  - A zero at $z = -1$ makes the discrete root locus pole behaviour more closely match the s-plane
  - Beware "dirty derivatives"
    - dy/dt terms derived from sequential digital values are called 'dirty derivatives' – these are especially sensitive to noise!
    - Employ actual velocity measurements when possible

Direct Design:
Second Order Digital Systems

Consider the z-transform of a decaying exponential signal:

$y(t) = e^{-at}\cos(bt)\,\mathcal{U}(t)$ ($\mathcal{U}(t)$ = unit step)

- sample: $y(kT) = r^k \cos(k\theta)\,\mathcal{U}(kT)$ with $r = e^{-aT}$ & $\theta = bT$
- transform: $Y(z) = \frac{1}{2}\frac{z}{(z - re^{j\theta})} + \frac{1}{2}\frac{z}{(z - re^{-j\theta})}$

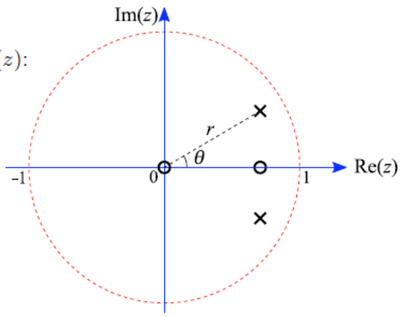
$$= \frac{z(z - r\cos\theta)}{(z - re^{j\theta})(z - re^{-j\theta})}$$

- e.g. y_k is the pulse response of $G(z)$:

$$G(z) = \frac{z(z - r\cos\theta)}{(z - re^{j\theta})(z - re^{-j\theta})}$$

poles: $\begin{cases} z = re^{j\theta} \\ z = re^{-j\theta} \end{cases}$

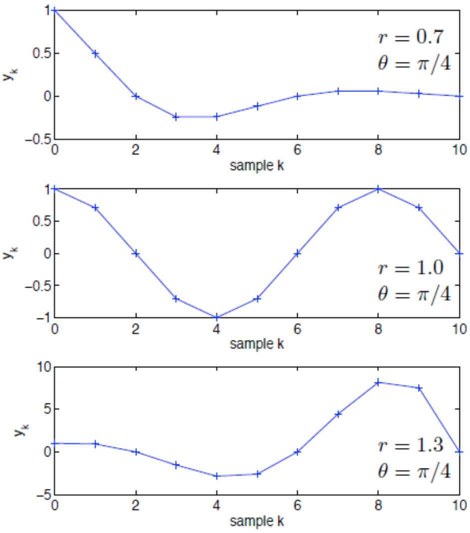
zeros: $\begin{cases} z = 0 \\ z = r\cos\theta \end{cases}$



Response of 2nd order system [1/3]

Responses for varying r:

- $r < 1$
  $\Downarrow$
  exponentially decaying envelope
- $r = 1$
  $\Downarrow$
  sinusoidal response with $2\pi/\theta$ samples per period
- $r > 1$
  $\Downarrow$
  exponentially increasing envelope



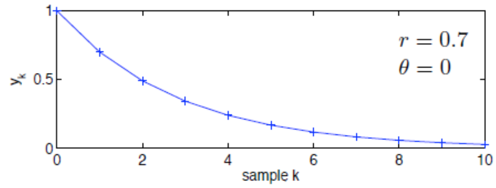
Response of 2nd order system [2/3]

Responses for varying θ:

- $\theta = 0$

  $\Downarrow$

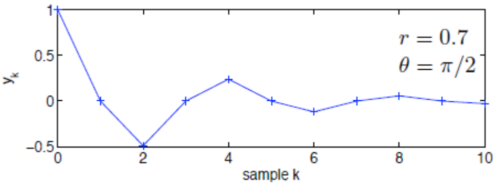
  decaying exponential



- $\theta = \pi/2$

  $\Downarrow$

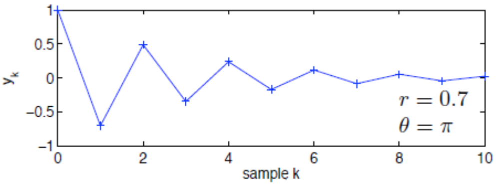
  $2\pi/\theta = 4$ samples per period



- $\theta = \pi$

  $\Downarrow$

  2 samples per period



Response of 2nd order system [3/3]

Some special cases:

- for $\theta = 0$, $Y(z)$ simplifies to:

$$Y(z) = \frac{z}{z - r}$$

$\Longrightarrow$ exponentially decaying response

- when $\theta = 0$ and $r = 1$:

$$Y(z) = \frac{z}{z - 1}$$

$\Longrightarrow$ unit step

- when $r = 0$:

$$Y(z) = 1$$

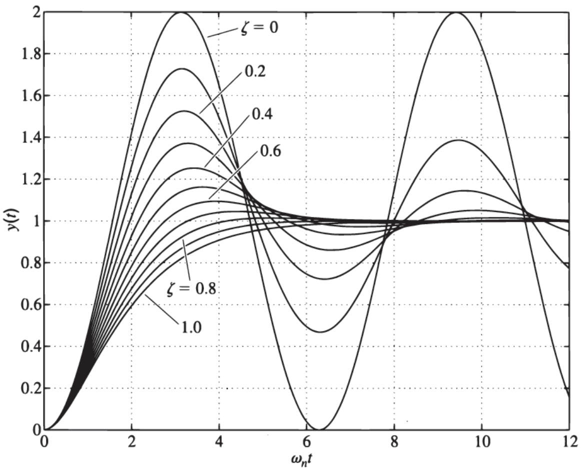
$\Longrightarrow$ unit pulse

- when $\theta = 0$ and $-1 < r < 0$:

  samples of alternating signs

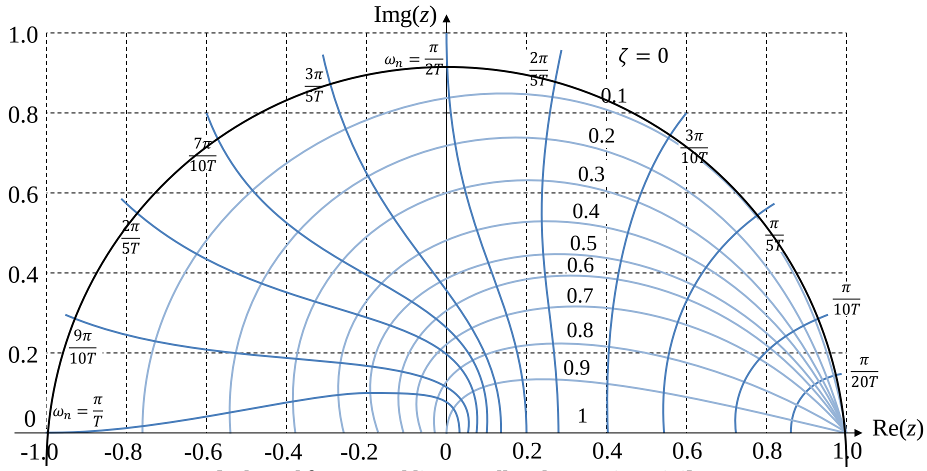
2nd Order System Response

- Response of a 2nd order system to increasing levels of damping:



Damping and natural frequency

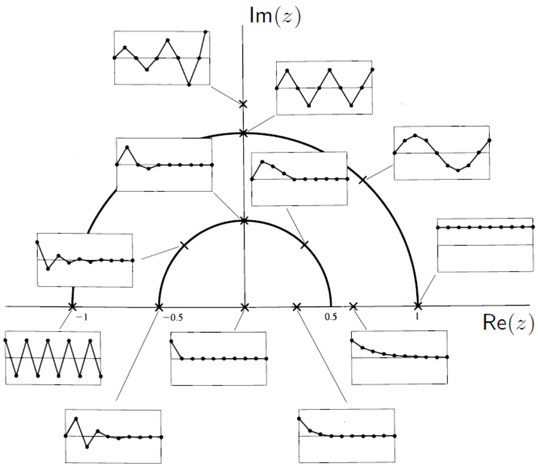
$$z = e^{sT} \text{ where } s = -\zeta\omega_n \pm j\omega_n\sqrt{1-\zeta^2}$$



[Adapted from Franklin, Powell and Emami-Naeini]

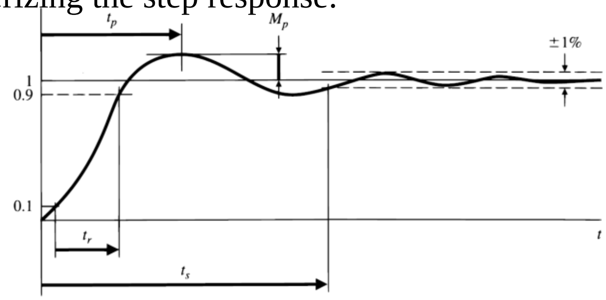
Pole positions in the z-plane

- Poles inside the unit circle are **stable**
- Poles outside the unit circle **unstable**
- Poles on the unit circle are oscillatory
- Real poles at 0 < z < 1 give exponential response
- Higher frequency of oscillation for larger
- Lower apparent damping for larer and r



2nd Order System Specifications

Characterizing the step response:

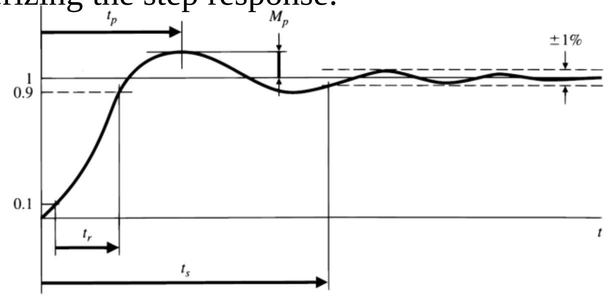


- Rise time (10% → 90%): $t_r \approx \frac{1.8}{\omega_0}$
- Overshoot: $M_p \approx \frac{e^{-\pi\zeta}}{\sqrt{1-\zeta^2}}$
- Settling time (to 1%): $t_s = \frac{4.6}{\zeta\omega_0}$
- Steady state error to unit step: e_{ss}
- Phase margin:

$$\phi_{PM} \approx 100\zeta$$

2nd Order System Specifications

Characterizing the step response:



- Rise time (10% → 90%) & Overshoot:
  t_r, M_p → ζ, ω_0 : Locations of dominant poles
- Settling time (to 1%):
  t_s → radius of poles: $|z| < 0.01^{\frac{T}{t_s}}$
- Steady state error to unit step:
  e_{ss} → final value theorem $e_{ss} = \lim_{z\to 1} \{(z-1)F(z)\}$

Ex: System Specifications → Control Design [1/4]

Design a controller for a system with:

- A continuous transfer function: $G(s) = \dfrac{0.1}{s(s+0.1)}$
- A discrete ZOH sampler
- Sampling time (T_s): T_s= 1s
- Controller:

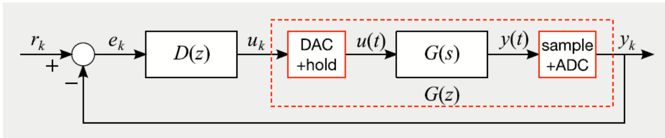
$$u_k = -0.5u_{k-1} + 13(e_k - 0.88e_{k-1})$$

The closed loop system is required to have:

- $M_p < 16\%$
- $t_s < 10$ s
- $e_{ss} < 1$

Ex: System Specifications → Control Design [2/4]

1. (a) Find the pulse transfer function of $G(s)$ plus the ZOH



$$G(z) = (1 - z^{-1})\mathcal{Z}\left\{\frac{G(s)}{s}\right\} = \frac{(z-1)}{z}\mathcal{Z}\left\{\frac{0.1}{s^2(s+0.1)}\right\}$$

e.g. look up $\mathcal{Z}\{a/s^2(s+a)\}$ in tables:

$$\begin{aligned} G(z) &= \frac{(z-1)}{z}\,\frac{z\Big((0.1 - 1 + e^{-0.1})z + (1 - e^{-0.1} - 0.1e^{-0.1})\Big)}{0.1(z-1)^2(z-e^{-0.1})} \\ &= \frac{0.0484(z+0.9672)}{(z-1)(z-0.9048)} \end{aligned}$$

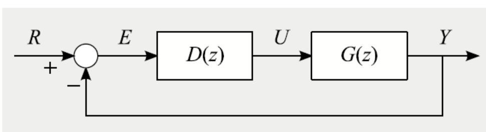
(b) Find the controller transfer function (using z = shift operator):

$$\frac{U(z)}{E(z)} = D(z) = 13\frac{(1-0.88z^{-1})}{(1+0.5z^{-1})} = 13\frac{(z-0.88)}{(z+0.5)}$$

Ex: System Specifications → Control Design [3/4]

2. Check the steady state error e_{ss} when r_k = unit ramp

$$e_{ss} = \lim_{k\to\infty} e_k = \lim_{z\to 1}(z-1)E(z)$$



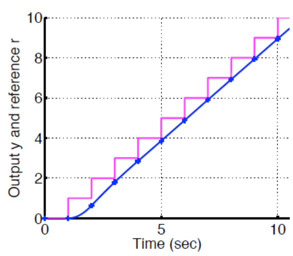
$$\frac{E(z)}{R(z)} = \frac{1}{1+D(z)G(z)}$$

$$R(z) = \frac{Tz}{(z-1)^2}$$

so $$e_{ss} = \lim_{z\to 1}\left\{(z-1)\frac{Tz}{(z-1)^2}\,\frac{1}{1+D(z)G(z)}\right\} = \lim_{z\to 1}\frac{T}{(z-1)D(z)G(z)}$$

$$= \lim_{z\to 1}\frac{T}{(z-1)\dfrac{0.0484(z+0.9672)}{(z-1)(z-0.9048)}D(1)}$$

$$= \frac{1-0.9048}{0.0484(1+0.9672)D(1)} = 0.96$$



$\Longrightarrow$ $e_{ss} < 1$ (as required)

Ex: System Specifications → Control Design [4/4]

3. Step response: overshoot $M_p < 16\% \implies \zeta > 0.5$

   settling time $t_s < 10 \implies |z| < 0.01^{1/10} = 0.63$

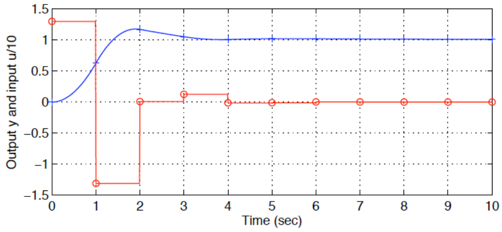
   The closed loop poles are the roots of $1 + D(z)G(z) = 0$, i.e.

$$1 + 13\frac{(z-0.88)}{(z+0.5)}\,\frac{0.0484(z+0.9672)}{(z-1)(z-0.9048)} = 0$$

$$\implies z = 0.88,\ -0.050 \pm j0.304$$

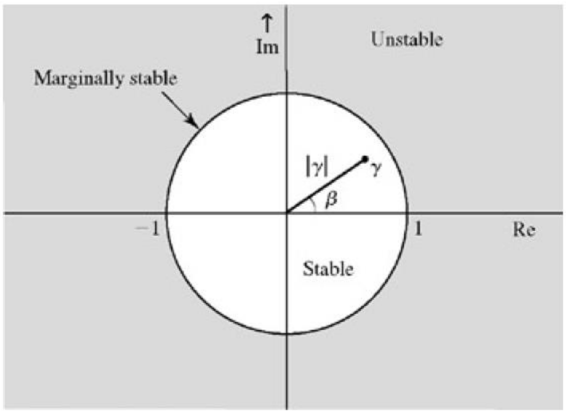
   But the pole at $z = 0.88$ is cancelled by controller zero at $z = 0.88$, and

$$z = -0.050 \pm j0.304 = re^{\pm j\theta} \implies \begin{cases} r = 0.31,\ \theta = 1.73 \\ \zeta = 0.56 \end{cases}$$

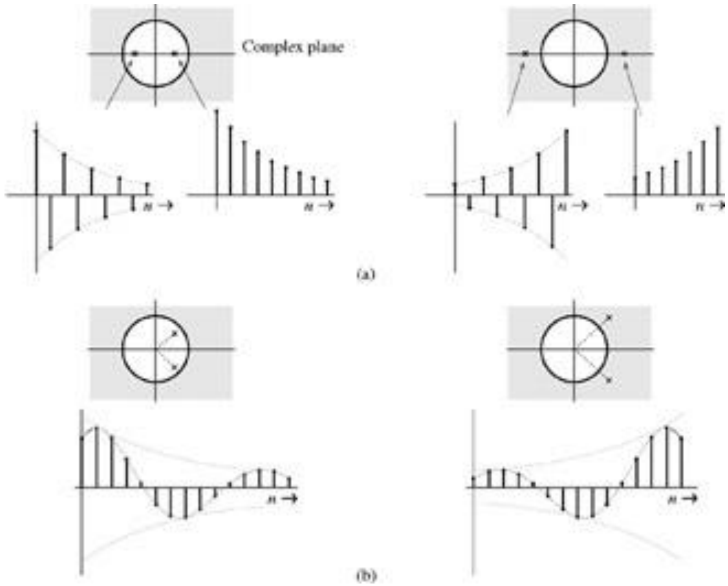


all specs satisfied!

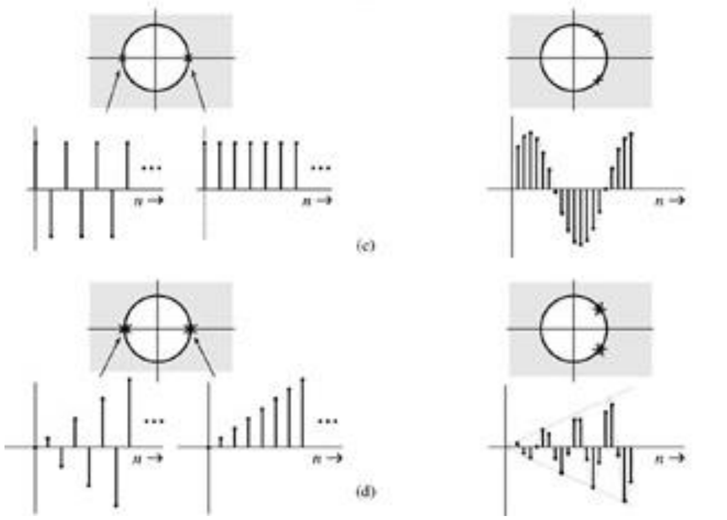
LTID Stability



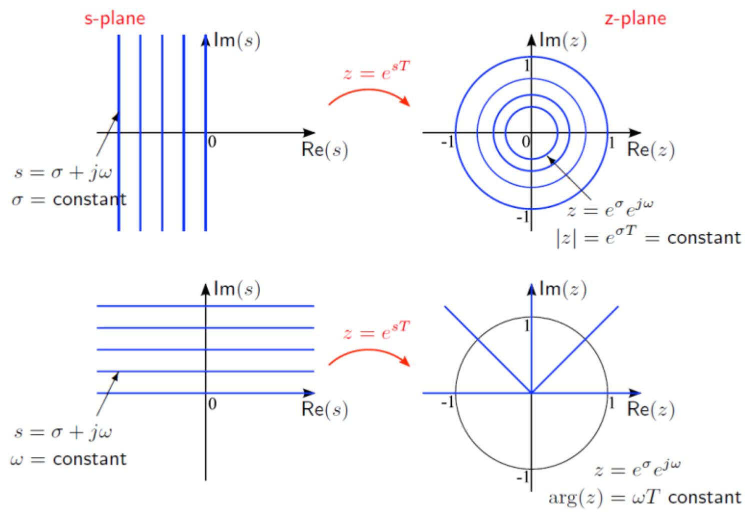
Characteristic roots location and the corresponding characteristic modes [1/2]



Characteristic roots location and the corresponding characteristic modes [2/2]



S-Plane to z-Plane [1/2]



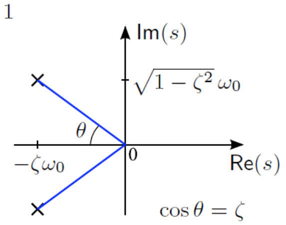
S-Plane to z-Plane [2/2]

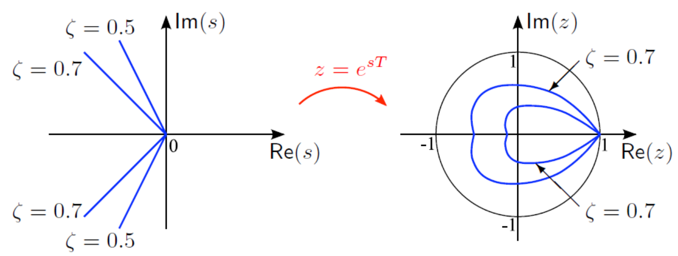
Pole locations for constant damping ratio $\zeta < 1$

$$s^2 + \zeta\omega_0 s + \omega_0^2 = 0$$

$$\Downarrow$$

$$s = -\zeta\omega_0 \pm j\sqrt{1-\zeta^2}\,\omega_0$$





$s = -\zeta\omega_0 + j\sqrt{1-\zeta^2}\,\omega_0$: ζ = constant $\qquad z = e^{-\zeta\omega_0 T} e^{-j\sqrt{1-\zeta^2}\omega_0 T}$

Relationship with s-plane poles and z-plane transforms

If $F(s)$ has a pole at $s = a$

then $F(z)$ has a pole at $z = e^{aT}$

$\uparrow$

consistent with $z = e^{sT}$

What about transfer functions?

$$G(z) = (1 - z^{-1})\mathcal{Z}\left\{\frac{G(s)}{s}\right\}$$

$\downarrow$

If $G(s)$ has poles $s = a_i$

then $G(z)$ has poles $z = e^{a_i T}$

but the zeros are unrelated

$\mathcal{F}(s)$	$f(kT)$	$F(z)$
$\frac{1}{s}$	$1(kT)$	$\frac{z}{z-1}$
$\frac{1}{s^2}$	kT	$\frac{Tz}{(z-1)^2}$
$\frac{1}{s+a}$	e^{-akT}	$\frac{z}{z-e^{-aT}}$
$\frac{1}{(s+a)^2}$	kTe^{-akT}	$\frac{Tze^{-aT}}{(z-e^{-aT})^2}$
$\frac{a}{s(s+a)}$	$1-e^{-akT}$	$\frac{z(1-e^{-aT})}{(z-1)(z-e^{-aT})}$
$\frac{b-1}{(s+a)(s+b)}$	$e^{-akT} - e^{-bkT}$	$\frac{(e^{-aT}-e^{-bT})z}{(z-e^{-aT})(z-e^{-bT})}$
$\frac{a}{s^2+a^2}$	$\sin akT$	$\frac{z \sin aT}{z^2 - (2\cos aT)z + 1}$
$\frac{b}{(s+a)^2+b^2}$	$e^{-akT}\sin bkT$	$\frac{ze^{-aT}\sin bT}{z^2 - 2e^{-aT}(\cos bT)z + e^{-2aT}}$

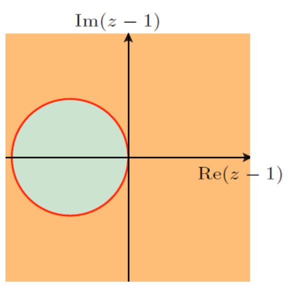
Fast sampling revisited

- For small T:

$$z = e^{sT} = 1 + sT + \frac{(sT)^2}{2} + \ldots \approx 1 + sT$$

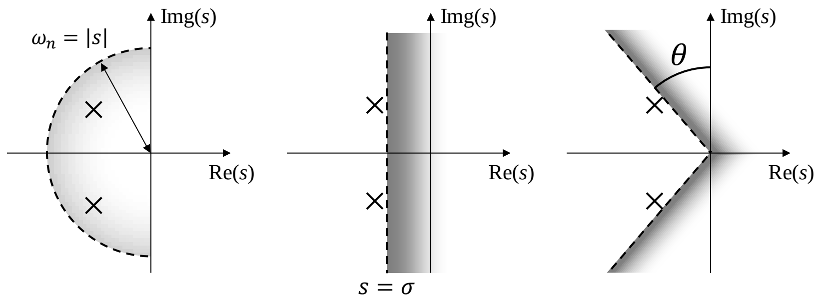
$$\rightarrow z \approx 1 + sT \rightarrow s = \frac{z-1}{T}$$

- Hence, the unit circle under the map from z to s-plane becomes:



Specification bounds

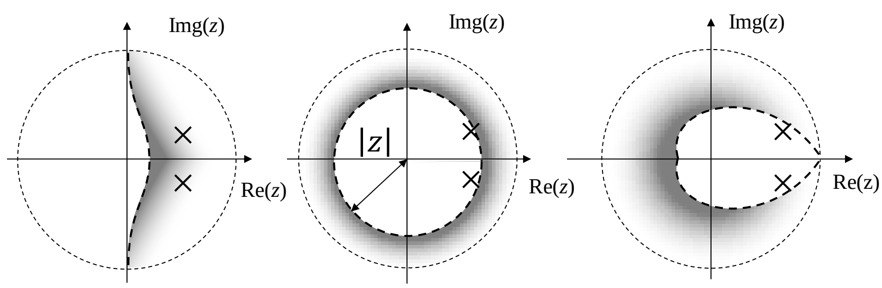
- Recall in the continuous domain, response performance metrics map to the s-plane:



$|s| = \frac{1.8}{t_r}$ $\qquad$ $s = \frac{4.6}{t_s}$ $\qquad$ $\theta = \sin^{-1}\zeta$

Discrete bounds

- These map to the discrete domain:



$|z| = e^{-t_s T}$

In practice, you'd use Matlab to plot these, and check that the spec is satisfied

Example Code:

```
%% Input System Model G
numg=5; deng=[1 20 0]; sysg=tf(numg, deng);

%% Approximate the ZOH (1-e^-sT)/(s)
[nd, dd]=pade(1,2);  %pade gives us the "hold" or -e^-sT of a ZOH
sysp=tf(nd, dd); sysi=tf([1],[1,0]); %Now we need the "1/s" portion
sys1=series(1-sysp, sysi); % Approximation as a series

%% Open loop response
syso=series(sys1, sysg); % computer the open loop G with the ZOH
sys=feedback(syso,1); % Computer the unity feedback response
step(sys) % Display the step response
```