CDS 101/110a: Lecture 9-1
PID Control

Douglas G. MacMynowski

Goals:

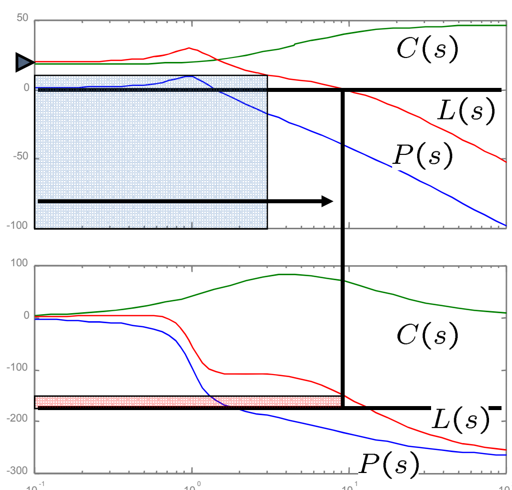
- Show how to use “loop shaping” using PID (Proportional + Integral + Derivative) to achieve a performance specification

Reading:

- Åström and Murray, *Feedback Systems,* Ch 10
- *Advanced*: Lewis, Chapters 12-13

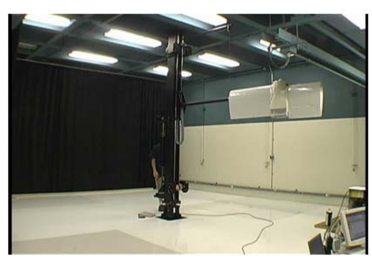
Last week: Loop Shaping

- Loop Shaping for Stability & Performance
  - Steady state error, bandwidth, tracking

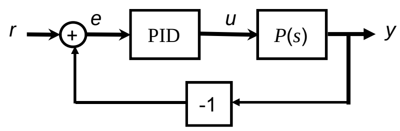


Main ideas

- Performance specs give bounds on loop transfer function
- Use controller to shape response
- Gain/phase relationships constrain design approach
- Lead compensator useful to add phase



Overview: PID control

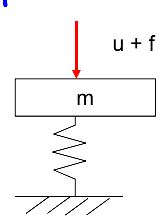


$$u = k_p e + k_i \int e \, \mathrm{d}t + k_d \dot{e}$$

$$C(s) = \frac{k_i}{s} + k_p + k_d s$$

- Intuition
  - Proportional term: provides inputs that correct for "current" errors
  - Integral term: ensures steady state error goes to zero
  - Derivative term: provides "anticipation" of upcoming changes
- A bit of history on "three term control"
  - First appeared in 1922 paper by Minorsky: "Directional stability of automatically steered bodies" under the name "three term control"
  - Also realized that "small deviations" (linearization) could be used to understand the (nonlinear) system dynamics under control
- Utility of PID
  - PID control is most common feedback structure in engineering systems
  - For many systems, only need PI or PD (special case)
  - Many tools for tuning PID loops and designing gains

Time-domain motivation



- Mass-spring system: $m\ddot{z} + c\dot{z} + kz = u + f$
- PD control: $u = -(k_p z + k_d \dot{z})$
- Closed-loop: $m\ddot{z} + (c + k_d)\dot{z} + (k + k_p)z = f$
  - Derivative gain acts like increasing damping
    - Increases system stability (greater phase margin)
  - Proportional gain acts like increasing stiffness
    - No matter how large the stiffness, still a non-zero response to disturbance force
  - Steady-state (for constant disturbance force *f*)

$$\dot{z} = 0 \quad \Rightarrow \quad \lim_{t\to\infty} z(t) = \frac{1}{k + k_p} f, \text{ and } \lim_{t\to\infty} u(t) = -\frac{k_p}{k + k_p} f$$

- Integral control: $\dot{q} = z$

$$u = -(k_i q + k_p z + k_d \dot{z})$$

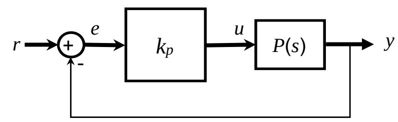
  - Steady-state (assuming stability) then for constant *f*

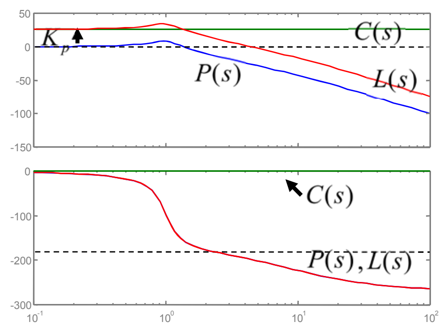
$$\dot{q} = 0 \quad \Rightarrow \quad \lim_{t\to\infty} z = 0, \text{ and } \lim_{t\to\infty} u(t) = -f$$

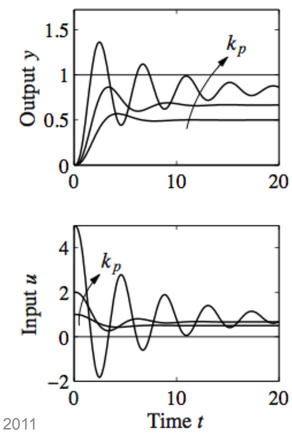
Proportional Feedback

$k_p > 0$ if $P(0) > 0$

- Simplest controller choice: $u = k_p e$
  - Effect: lifts gain with no change in phase
  - Good for plants with low phase up to desired bandwidth
  - Bode: shift gain up by factor of k_p
  - Step response: better steady state error, but with decreasing stability

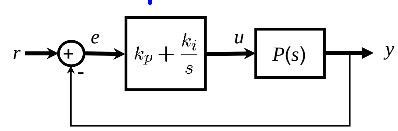




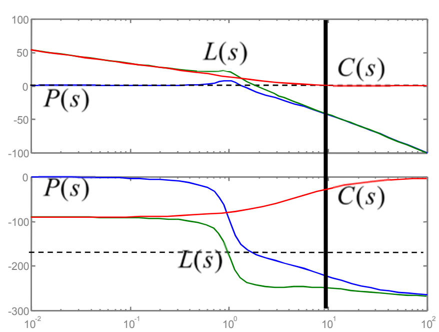


Proportional + Integral Compensation

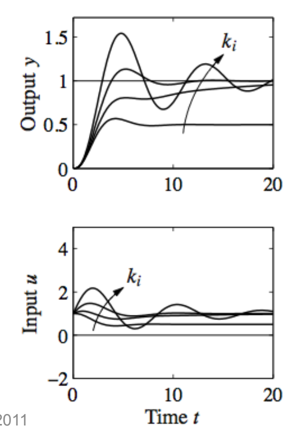
- Use to eliminate steady state error
  - Effect: lifts gain at low frequency
  - Gives *zero* steady state error
  - Bode: infinite SS gain + phase lag
  - Step response: zero steady state error, with smaller settling time, but more overshoot



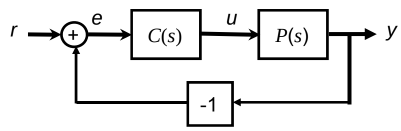
$k_p > 0$, $k_i > 0$



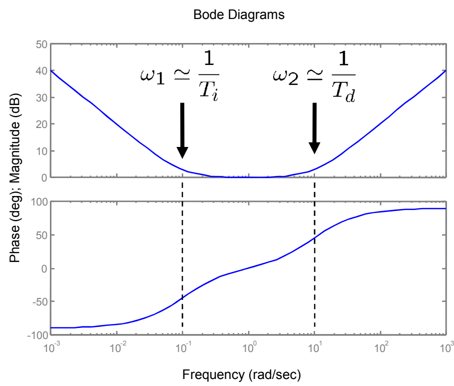
$\omega_z = k_i / k_p$

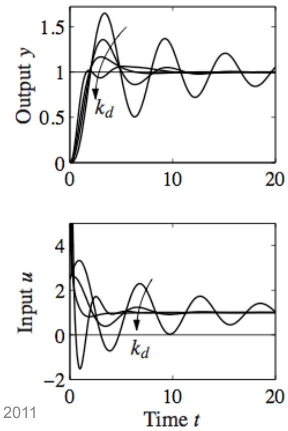


Proportional + Integral + Derivative (PID)



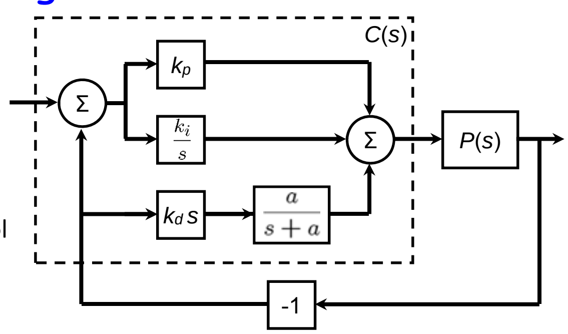
$$C(s) = k_p + k_i \frac{1}{s} + k_d s$$
$$= k\left(1 + \frac{1}{T_i s} + T_d s\right)$$
$$= (kT_d)\,\frac{(s+\alpha_i)(s+\alpha_d)}{s}$$

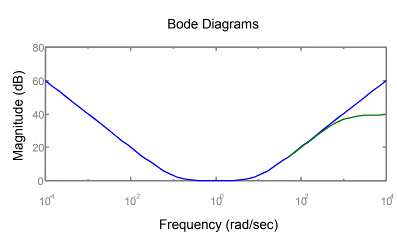




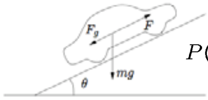
Implementing Derivative Action

- Problems with derivatives
  - High frequency noise amplified by derivative term
  - Step inputs in reference can cause large inputs
  - Shows up in Gang of Four...
- Solution: modified PID control
  - Use high frequency rolloff in derivative term
    - first order filter will give finite gain at high frequency
    - use higher order filter if needed
  - Don't feed reference signal through derivative block
    - Useful when reference has unwanted high frequency content
    - Alternative solution: reference shaping via two DOF design (*F*(*s*) block)
  - Many other variations (see AM08 + refs)

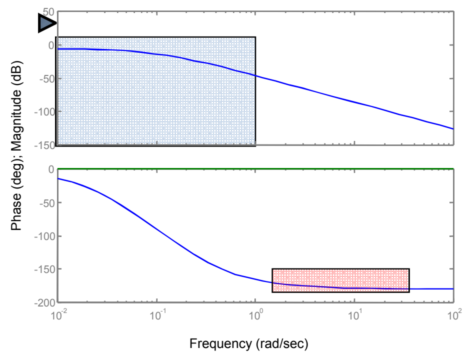




Example: Cruise Control using PID - Specification



$$P(s) = \frac{1/m}{s + b/m} \times \frac{r}{s + a}$$



- Performance Specification

  $\leq$ 1% steady state error

  $\Rightarrow$ Zero frequency gain > 100

  $\leq$ 10% tracking error up to 1 rad/sec

  $\Rightarrow$ Gain > 10 from 0-1 rad/sec

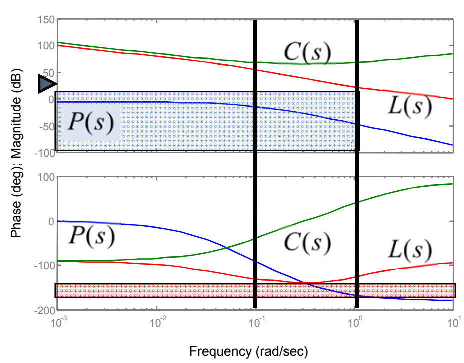
  $\geq$ 45˚ phase margin

  $\Rightarrow$ Gives good relative stability

  $\Rightarrow$ Provides robustness to uncertainty
- Observations
  - Purely proportional gain won't work: to get gain above desired level will not leave adequate phase margin
  - Need to increase the phase from ~0.5 to 2 rad/sec and increase gain as well

Example: Cruise Control using PID - Design

$$P(s) = \frac{1/m}{s + b/m} \times \frac{r}{s + a}$$



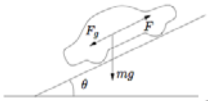
- Approach
  - Use proportional gain to give desired tracking performance
  - Use integral gain to make steady state error small (zero, in fact)
  - Use derivative action to increase phase lead in the cross over region
- Controller
  - Ti = 1/0.1; Td = 1/1; k = 2000

$$C(s) = 2000\frac{s^2 + 1.1s + 0.1}{s}$$

$$= 2200 + \frac{200}{s} + 2000s$$

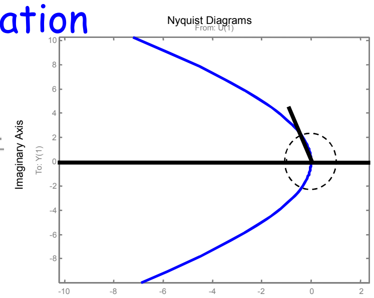
- Closed loop system
  - Very high steady state gain
  - Adequate tracking @ 1 rad/sec
  - ~80° phase margin
  - Verify with Nyquist + Gang of 4

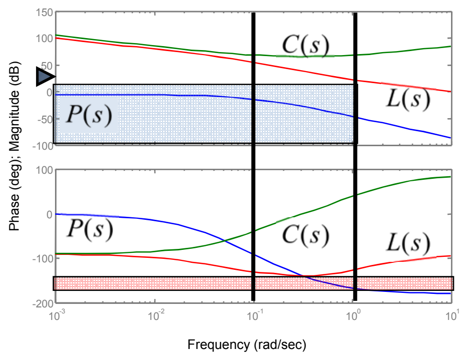
Example: Cruise Control using PID - Verification

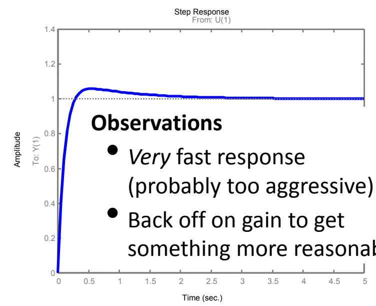


$$P(s) = \frac{1/m}{s + b/m} \times \frac{r}{s+a}$$

$$C(s) = 2000\frac{s^2 + 1.1s + 0.1}{s}$$



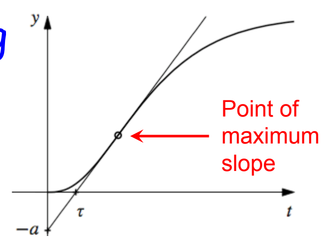




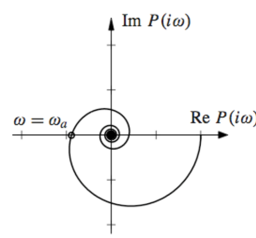
Observations

- *Very* fast response (probably too aggressive)
- Back off on gain to get something more reasonable

Automated PID Tuning



- Zeigler-Nichols step response method
  - Design PID gains based on step response
  - Measure maximum slope + intercept
  - Works OK for many plants (but underdamped)
  - Good way to get a first cut controller
- Ziegler-Nichols frequency response method
  - Increase gain until system goes unstable
  - Use critical gain and frequency as parameters



- Variations
  - Modified formulas (see text) give better response
  - Relay feedback: provides automated way to obtain critical gain, frequency

Type	k_p	T_i	T_d
P	1/a		
PI	0.9/a	3τ	
PID	1.2/a	2τ	0.5τ

(a) Step response method

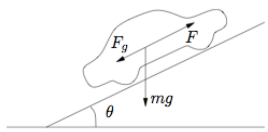
Type	k_p	T_i	T_d
P	$0.5k_c$		
PI	$0.4k_c$	$0.8T_c$	
PID	$0.6k_c$	$0.5T_c$	$0.125T_c$

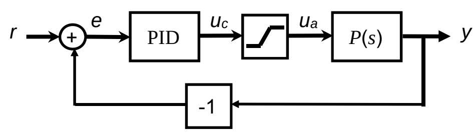
(b) Frequency response method

$$k_p = \frac{0.15\tau + 0.35T}{K\tau} \quad \left(\frac{0.9T}{K\tau}\right), \quad k_i = \frac{0.46\tau + 0.02T}{K\tau^2} \quad \left(\frac{0.3T}{K\tau^2}\right),$$

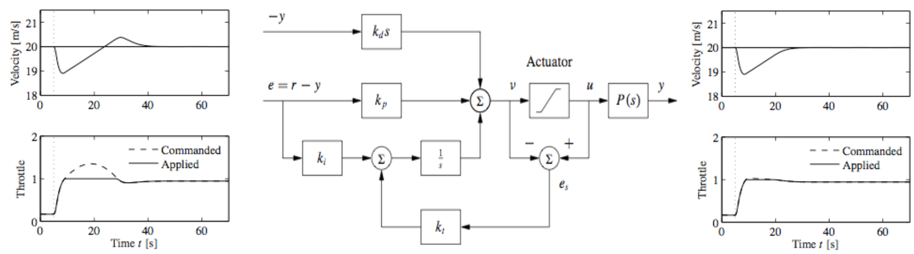
$$k_p = 0.22k_c - \frac{0.07}{K} \quad \left(0.4k_c\right), \quad k_i = \frac{0.16k_c}{T_c} + \frac{0.62}{KT_c} \quad \left(\frac{0.5k_c}{T_c}\right).$$

Windup and Anti-Windup Compensation





- Problem
  - Limited magnitude input (saturation)
  - Integrator "winds up" ⇒ overshoot
- Solution
  - Compare commanded input to actual
  - Subtract off difference from integrator

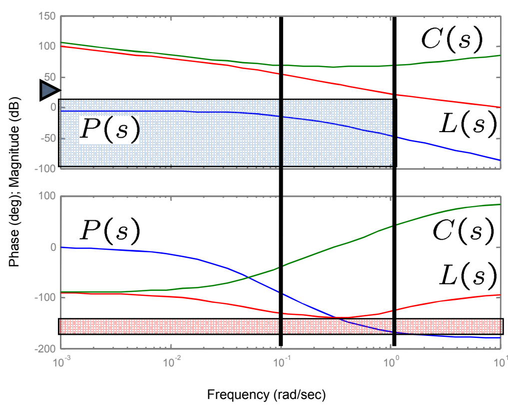


(a) Windup (b) Anti-windup

Summary: Frequency Domain Design using PID

- Loop Shaping for Stability & Performance
  - Steady state error, bandwidth, tracking

$$H_{ue}(s) = k_p + k_i \frac{1}{s} + k_d s$$



- Main ideas
  - Performance specs give bounds on loop transfer function
  - Use controller to shape response
  - Gain/phase relationships constrain design approach
  - Standard compensators: proportional, PI, PID