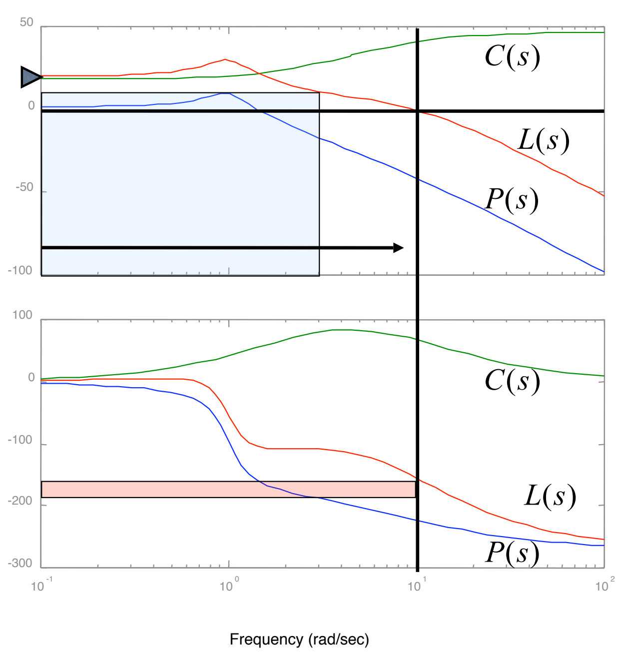
CDS 101/110a: Lecture 9-1
PID Control

Richard M. Murray
24 November 2008

Goals:
- Show how to use "loop shaping" using PID to achieve a performance specification
- Discuss the use of integral feedback and antiwindup compensation

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

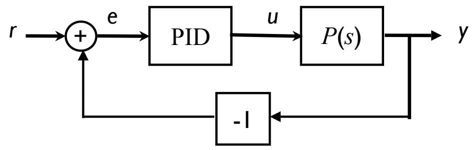
Overview of Loop Shaping



Performance specification
- ▶ Steady state error
- ☐ Tracking error
- → Bandwidth
- ☐ Relative stability

Approach: "shape" loop transfer function using $C(s)$
- $P(s)$ + specifications given
- $L(s) = P(s)\,C(s)$
  - Use $C(s)$ to choose desired shape for $L(s)$
- Important: can't set gain and phase independently

Overview: PID control



$$u = k_p e + k_i \int e\,dt + k_d \dot{e}$$

Intuition

- Proportional term: provides inputs that correct for "current" errors
- Integral term: insures 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

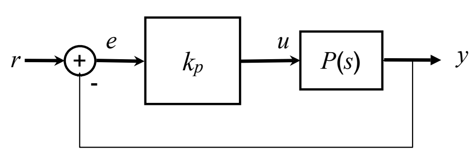
Utility of PID

- 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

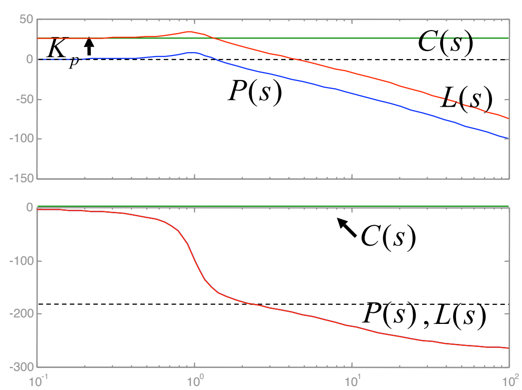
Proportional Feedback

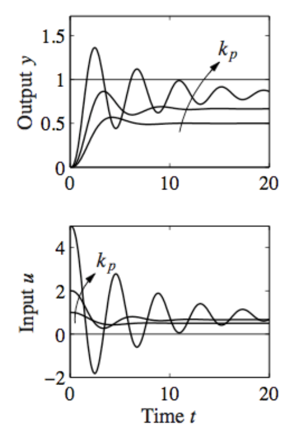
Simplest controller choice: $u = k_p e$

- 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



$k_p > 0$

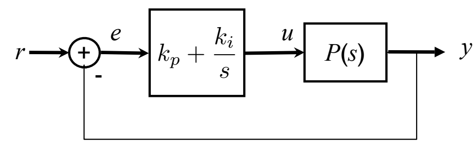




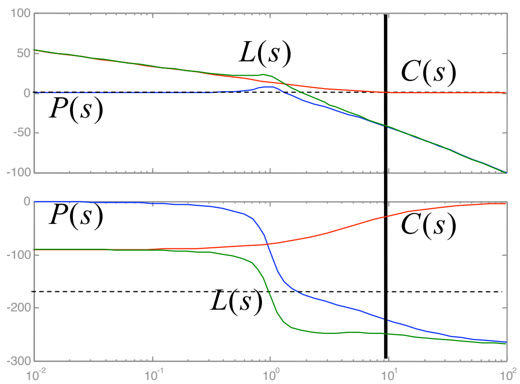
Proportional + Integral Compensation

Use to eliminate steady state error

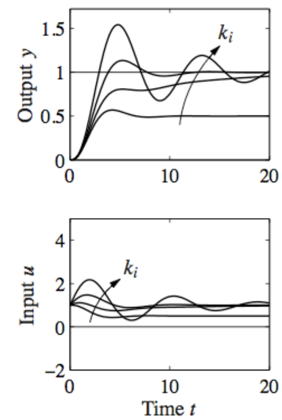
- Effect: lifts gain at low frequency
- Gives *zero* steady state error
- Bode: infinite SS gain + phase lag
- Step response: zero steady state error, with smaller settling time, but more overshoot



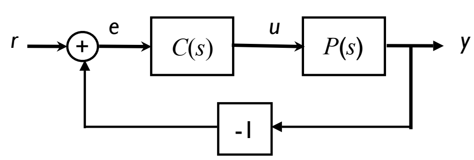
$$k_p > 0, \quad k_i > 0$$



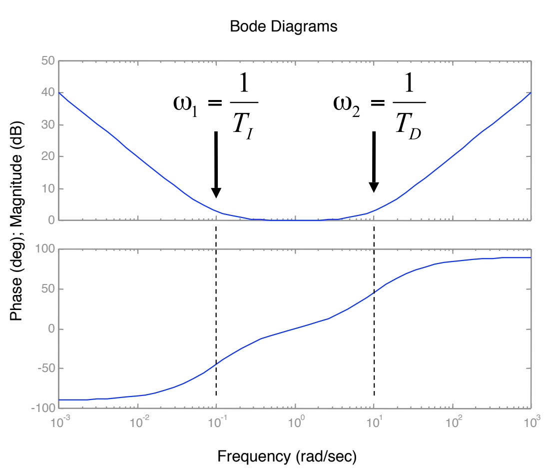
$\omega_z = K_I / K_p$

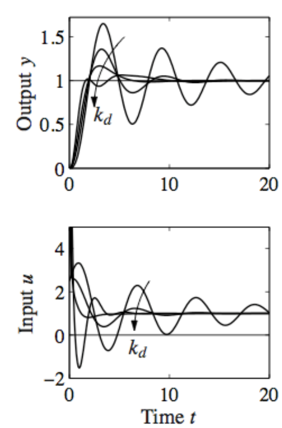


Proportional + Integral + Derivative (PID)



$$\begin{aligned} C(s) &= k_p + k_i\frac{1}{s} + k_d s \\ &= k(1 + \frac{1}{T_i s} + T_d s) \\ &= \frac{kT_d}{T_i}\frac{(s + 1/T_i)(s + 1/T_d)}{s} \end{aligned}$$





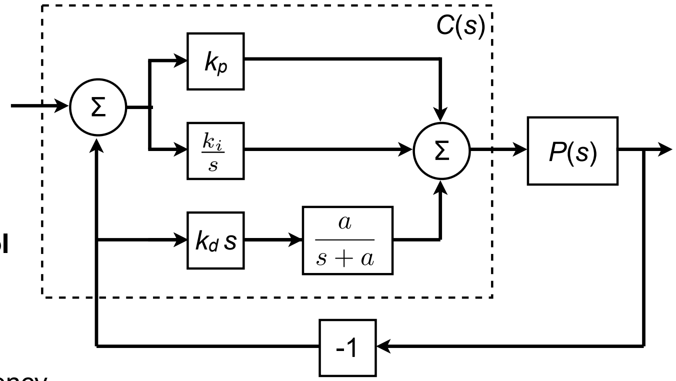
Implementing Derivative Action

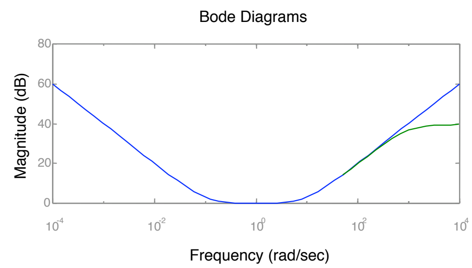
Problems with derivatives

- High frequency noise amplified by derivative term
- Step inputs in reference can cause large inputs
- Show up in Gang of Four...

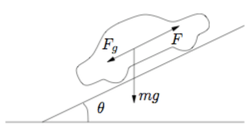
Solution: modified PID control

- 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
  - Better solution: reference shaping via two DOF design ($F(s)$ block)
- Many other variations (see AM08 + refs)

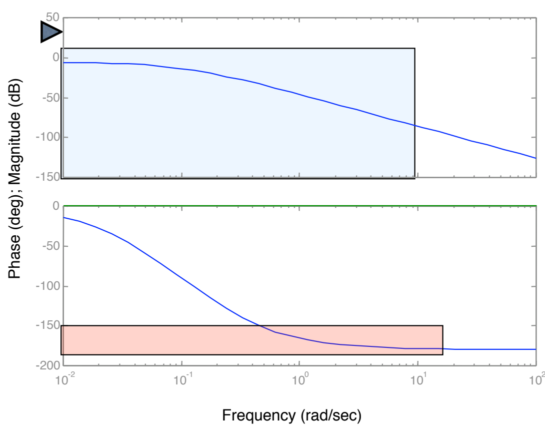




Example: Cruise Control using PID - Specification



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



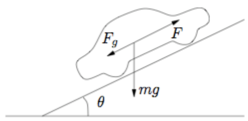
Performance Specification

- ≤ 1% steady state error
  - Zero frequency gain > 100
- ≤ 10% tracking error up to 10 rad/sec
  - Gain > 10 from 0-10 rad/sec
- ≥ 45˚ phase margin
  - Gives good relative stability
  - Provides robustness to uncertainty

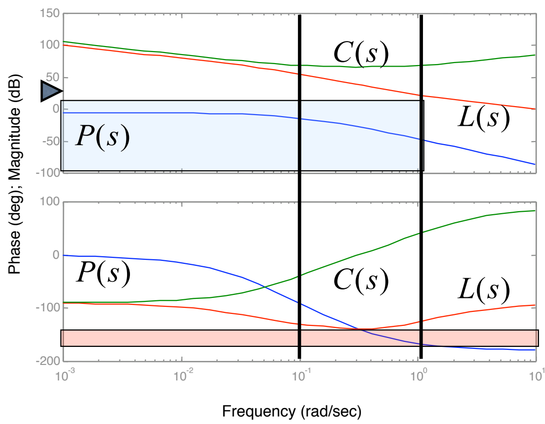
Observations

- Purely proportional gain won't work: to get gain above desired level will not leave adequate phase margine
- 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 integral gain to make steady state error small (zero, in fact)
- Use derivative action to increase phase lead in the cross over region
- Use proportional gain to give desired bandwidth

Controller

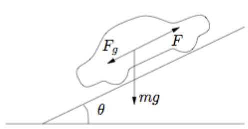
- Ti = 1/0.1; Td = 1/1; k = 2000

$$C(s) = 2000\frac{s^2 + 1.1s + 0.1}{s} = 2200 + \frac{200}{s} + 2000s$$

Closed loop system

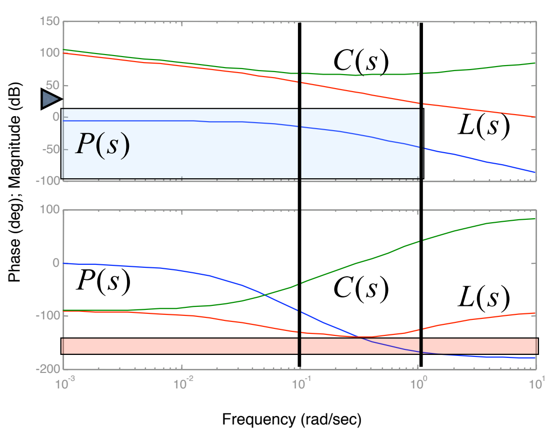
- Very high steady state gain
- Adequate tracking @ 1 rad/sec
- ~80° phase margin
- Verify with Nyquist + Gang of 4

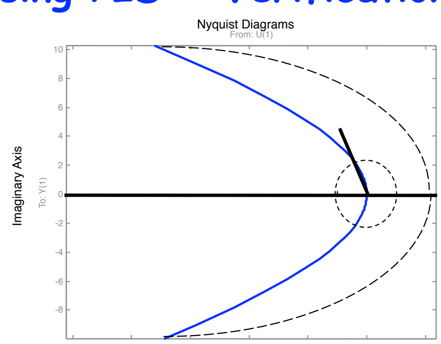
Example: Cruise Control using PID - Verification

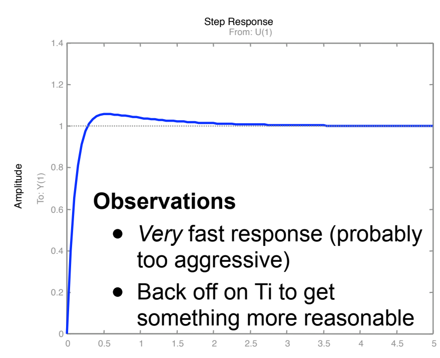


$$P(s) = \frac{1/m}{s + b/m} \times \frac{r}{s+a}$$

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







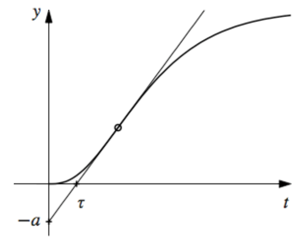
Observations

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

PID Tuning

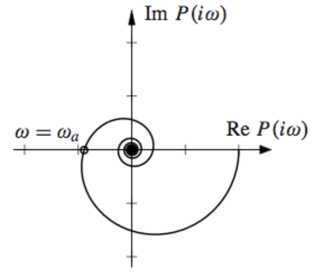
Zeigler-Nichols step response method

- 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

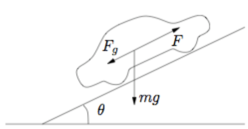
Type	k_p	T_i	T_d
P	$0.5k_c$		
PI	$0.4k_c$	$0.8T_c$	
PID	$0.6k_c$	$0.5T_c$	$0.125T_c$

(b) Frequency response method

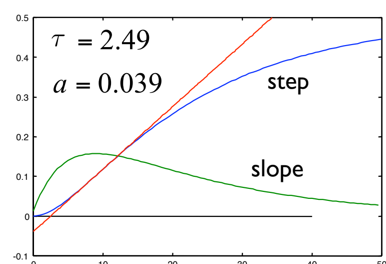
$$k_p = \frac{0.15\tau + 0.35T}{K\tau} \quad \left(\frac{0.9T}{K\tau}\right), \qquad k_i = \frac{0.46\tau + 0.02T}{K\tau^2} \quad \left(\frac{0.3T}{K\tau^2}\right),$$

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

Example: PID cruise control

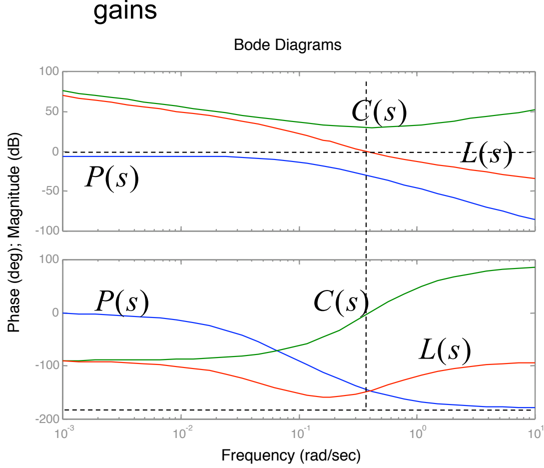


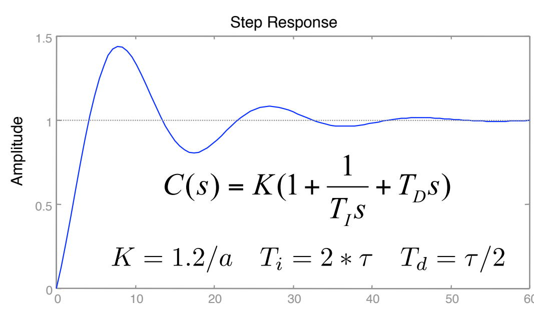
$$P(s) = \frac{1/m}{s + b/m} \times \frac{r}{s+a}$$



Ziegler-Nichols design for cruise controller

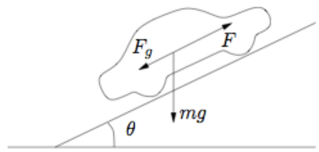
- Plot step response, extract τ and a, compute gains

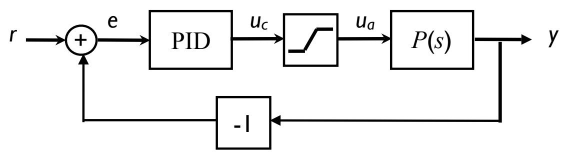




- Result: *sluggish* ⇒ increase loop gain + more phase margine (shift zero)

Windup and Anti-Windup Compensation



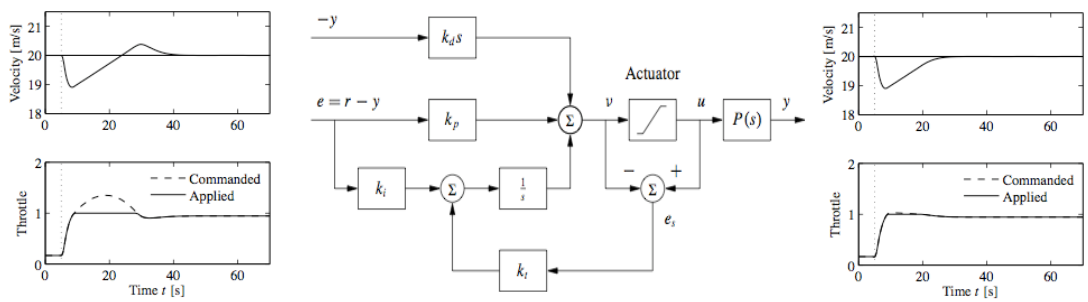


Problem

- Limited magnitude input (saturation)
- Integrator "winds up" => overshoot

Solution

- Compare commanded input to actual
- Subtract off difference from integrator



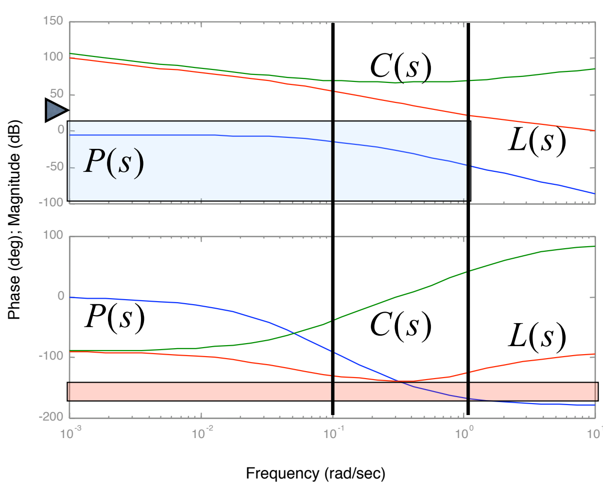
(a) Windup (b) Anti-windup

Summary: Frequency Domain Design using PID

Loop Shaping for Stability & Performance

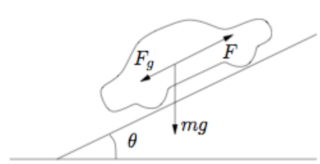
- Steady state error, bandwidth, tracking

$$H_{ue}(s) = K_p + K_I \times \frac{1}{s} + K_D s$$



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



```
% L8_1_pid.m - MATLAB commands for Lecture 8.1
% RMM, 13 Nov 04
%
% Required files: none

%%
%% Sample system for control design
%%
%% Use second order system + lag for plant; second order lead as control
%%
sys = tf([1], [1 2*0.2*1 1^2]) * tf([1], [0.1, 1]);

%% Proportional gain
kp = tf([20], [1]);
bode(sys, sys*kp, kp);

%% Integral compensation
integ = tf([1 5], [1 0]);
bode(sys, sys*integ, integ);
nyquist(sys, sys*integ, {0.28,1e5});

%%
%% Cruise control example
%%

% Parameter definitions
m = 1000;                                       % mass of the car, kg
b = 50;                                         % damping coefficient, N sec/m
a = 0.2;                                        % engine lag coefficient
r = 5;                                          % transmission gain

% Dynamics
veh = tf([1/m], [1 b/m]);                       % vehicle
eng = tf([r], [1 a]);                           % engine

% Plot the open loop dynamics and specifications
bode(veh*eng, tf(40), {0.01, 100});
print -dmeta cruise_open.wmf

% Controller description - use frequency break points
wI = 0.1;
wD = 1;
K = 2e4;

% Now compute PID transfer functoin using formula on slide 9
ctr = K*wI/wD * tf([1 (wD + wI) wD*wI], [1 0]);

bode(veh*eng, ctr, veh*eng*ctr);
print -dmeta cruise_closed.wmf

% Now check the performance of everything

nyquist(veh*eng*ctr);
print -dmeta cruise_nyquist.wmf

sys = feedback(veh*eng*ctr, 1);
step(sys, 5); print -dmeta cruise_step.wmf
```