CDS 101/110a: Lecture 7-1
Loop Analysis of Feedback Systems

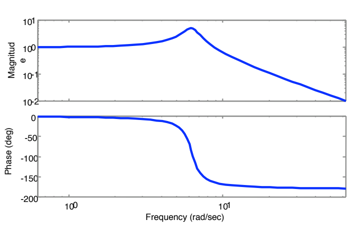
Richard M. Murray
10 November 2008

Goals:
- Show how to compute closed loop stability from open loop properties
- Describe the Nyquist stability criterion for stability of feedback systems
- Define gain and phase margin and determine it from Nyquist and Bode plots

Reading:
- Åström and Murray, *Feedback Systems,* Ch 9
- *Advanced*: Lewis, Chapters 7
- CDS 210: DFT, Ch 3

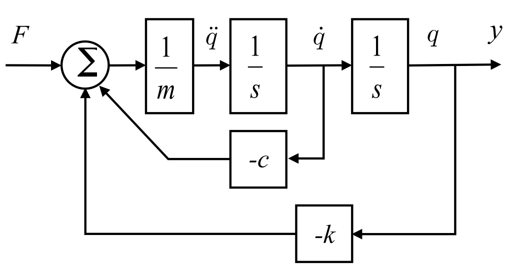
Review From Last Week

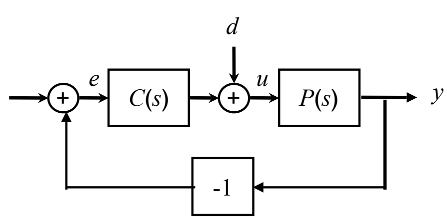
$$u = A\sin(\omega t) \;\longrightarrow\; \begin{array}{c} \dot{x} = Ax + Bu \\ y = Cx + Du \\ x(0) = 0 \end{array} \;\longrightarrow\; y_{ss} = A\cdot|G(i\omega)|\times \sin(\omega t + \arg G(i\omega))$$



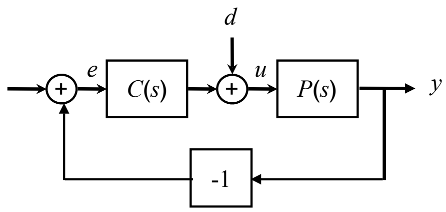
$$G(s) = C(sI - A)^{-1}B + D$$

$$G_{y_2u_1} = G_{y_2u_2}G_{y_1u_1} = \frac{n_1n_2}{d_1d_2}$$





Closed Loop Stability



Q: how do open loop dynamics affect the closed loop stability?

- Given open loop transfer function $C(s)P(s)$ determine when system is stable

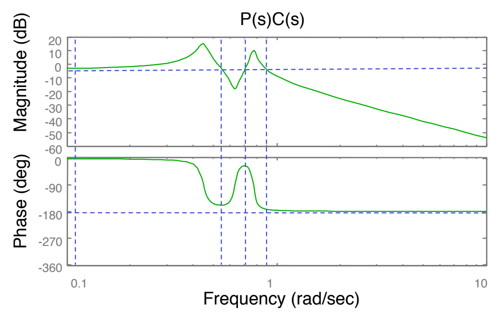
Brute force answer: compute poles closed loop transfer function

$$H_{yr} = \frac{PC}{1+PC} = \frac{n_p n_c}{d_p d_c + n_p n_c}$$

- Poles of H_{yr} = zeros of 1 + PC
- Easy to compute, but not so good for design

Alternative: look for conditions on *PC* that lead to instability

- Example: if $PC(s) = -1$ for some $s = i\omega$, then system is *not* asymptotically stable
- Condition on *PC* is much nicer because we can *design* *PC*(*s*) by choice of *C*(*s*)
- However, checking $PC(s) = -1$ is not enough; need more sophisticated check

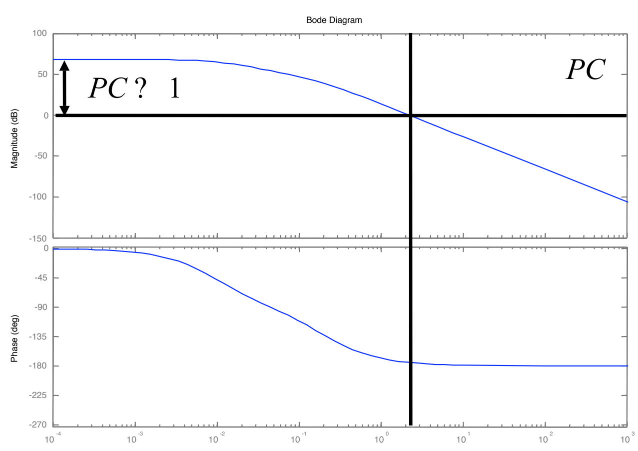


Game Plan: Frequency Domain Design

Goal: figure out how to *design* C(*s*) so that 1+*C*(*s*)*P*(*s*) is stable *and* we get good performance

$$H_{yr} = \frac{PC}{1+PC}$$

- Poles of H_{yr} = zeros of 1 + PC
- Would also like to "shape" H_{yr} to specify performance at differenct frequencies



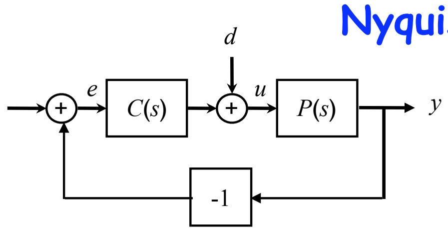
- Low frequency range:

$$PC \,?\, 1 \Rightarrow \frac{PC}{1+PC} \approx 1$$

(good tracking)

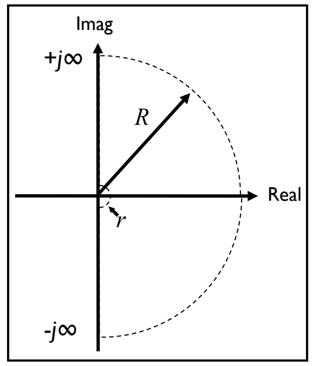
- Bandwidth: frequency at which closed loop gain = ½ ⇒ open loop gain ≈ 1
- Idea: use *C*(*s*) to *shape PC* (under certain constraints)
- Need tools to analyze stability and performance for closed loop given *PC*

Nyquist Criterion



Determine stability from (open) loop transfer function, $L(s) = P(s)C(s)$.

- Use "principle of the argument" from complex variable theory (see reading)



- Nyquist "D" contour
- Take limit as $r \to 0$, $R \to \infty$
- Trace from $-\infty$ to $+\infty$ along imaginary axis

Thm (Nyquist). Consider the Nyquist plot for loop transfer function $L(s)$. Let

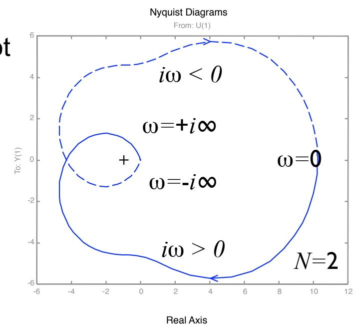
P # RHP poles of $L(s)$

N # clockwise encirclements of -1

Z # RHP zeros of $1 + L(s)$

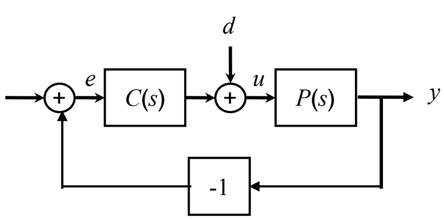
Then

$$Z = N + P$$



- Trace frequency response for $L(s)$ along the Nyquist "D" contour
- Count net # of clockwise encirclements of the -1 point

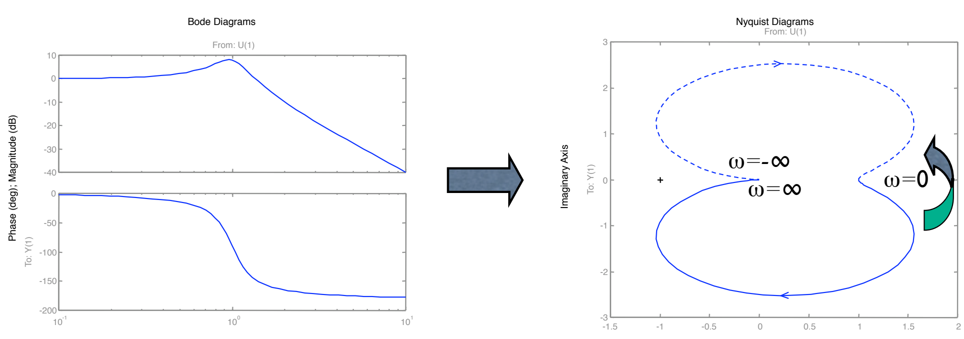
Simple Interpretation of Nyquist



Basic idea: avoid positive feedback

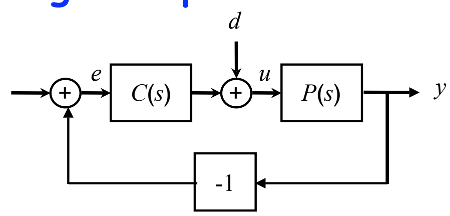
- If $L(s)$ has 180˚ phase (or greater) and gain greater than 1, then signals are amplified around loop
- Use when phase is monotonic
- General case requires Nyquist

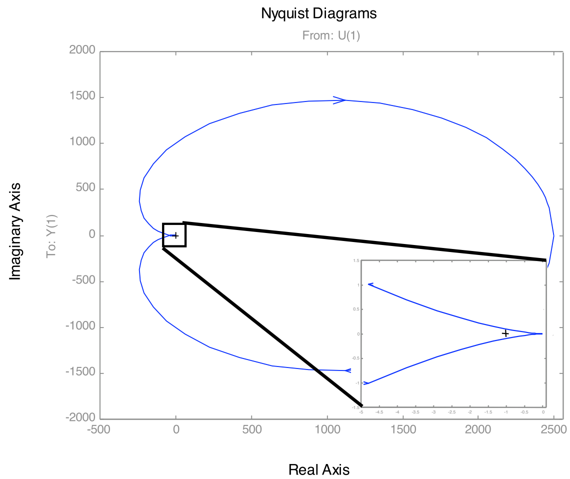
Can generate Nyquist plot from Bode plot + reflection around real axis



ambode(sys) [or bode(sys) in dB] amnyquist(sys)

Example: Proportional + Integral* speed controller





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

$$C(s) = K_p + \frac{K_i}{s + 0.01}$$

Remarks

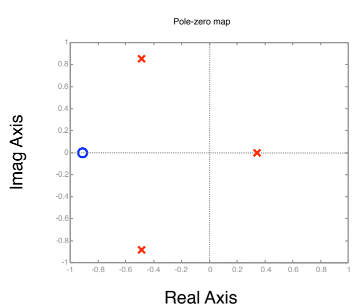
- $N = 0, P = 0 \Rightarrow Z = 0$ (stable)
- Need to zoom in to make sure there are no net encirclements
- Note that we don't have to compute closed loop response

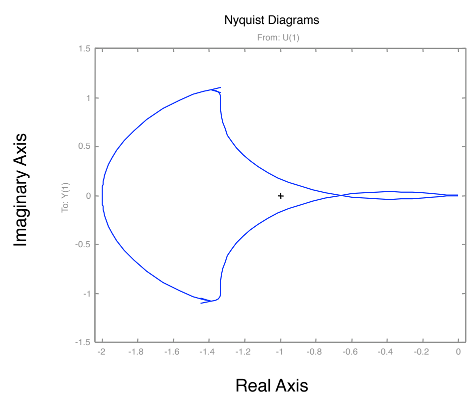
* slightly modified; more on the design of this compensator in next week's lecture

More complicated systems

What happens when open loop plant has RHP poles?

- 1 + PC has singularities inside D countour ⇒ these must be taken into account





$$L(s) = \frac{s + 1}{s - 0.5} \times \frac{1}{s^2 + s + 1}$$

unstable pole

$N = -1, P = 1 \Rightarrow Z = N + P = 0$ (stable)

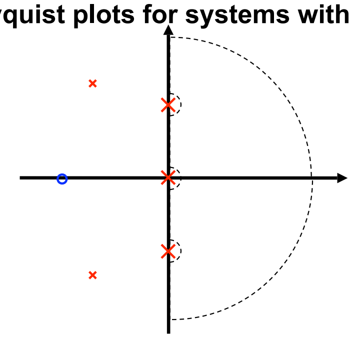
$$\frac{1}{1 + L} = \frac{s + 1}{(s + 0.35)(s + 0.07 + 1.2j)(s + 0.07 - 1.2j)}$$ ✓

Comments and cautions

Why is the Nyquist plot *useful*?

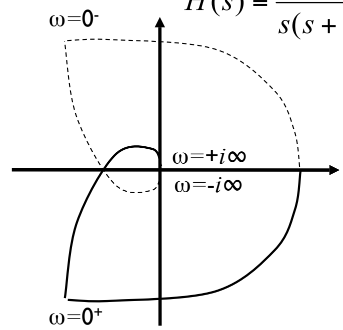
- Old answer: easy way to compute stability (before computers and MATLAB)
- Real answer: gives *insight* into stability and robustness; very useful for reasoning about stability

Nyquist plots for systems with poles on the $j\omega$ axis



- chose contour to avoid poles on axis
- need to carefully compute Nyquist plot at these points
- evaluate $H(\varepsilon+0i)$ to determine direction

$$H(s) = \frac{1}{s(s+1)^2}$$



Cautions with using MATLAB

- MATLAB doesn't generate portion of plot for poles on imaginary axis
- These must be drawn in by hand (make sure to get the orientation right!)

Robust stability: gain and phase margins

Nyquist plot tells us if closed loop is stable, but not *how* stable

Gain margin

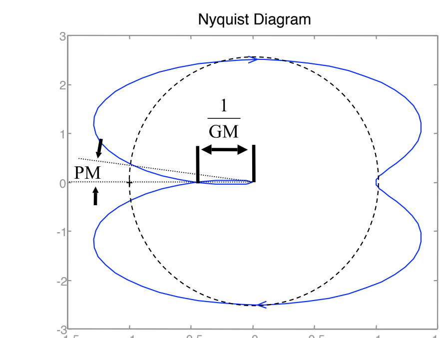
- How much we can modify the *loop gain* and still have the system be stable
- Determined by the location where the loop transfer function crosses 180˚ phase

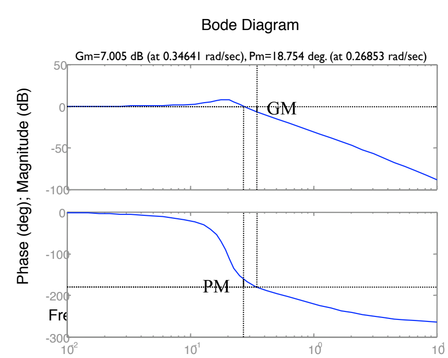
Phase margin

- How much we can add "phase delay" and still have the system be stable
- Determined by the phase at which the loop transfer function has unity gain

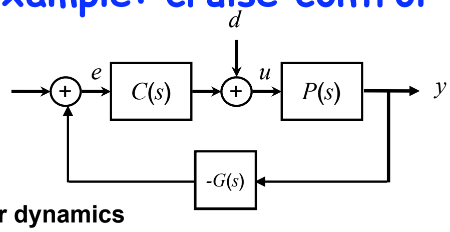
Bode plot interpretation

- Look for gain = 1, 180˚ phase crossings
- MATLAB: margin(sys)





Example: cruise control



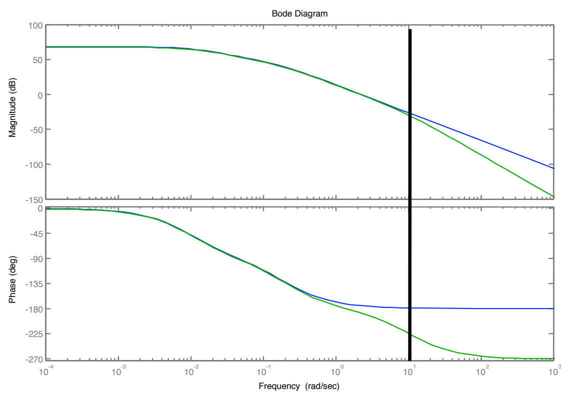
$$P(s) = \frac{1/m}{s + b/m} \times \frac{r}{s+a}$$

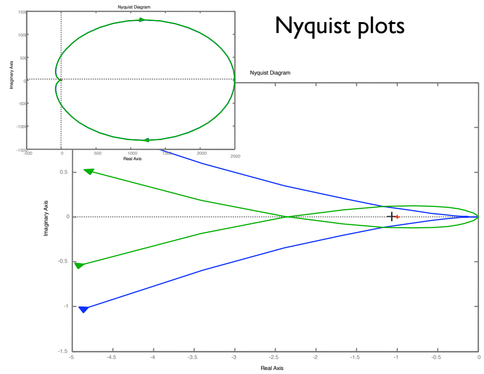
$$C(s) = K_p + \frac{K_i}{s + 0.01}$$

$$G(s) = \frac{10}{s+10}$$

Effect of additional sensor dynamics

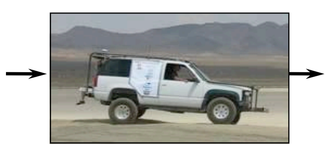
- New speedometer has pole at s = 10 (very fast); problems develop in the field
- What's the problem? A: insufficient phase margin in original design (not robust)

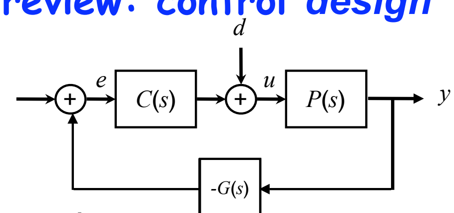




Nyquist plots

Preview: control *design*





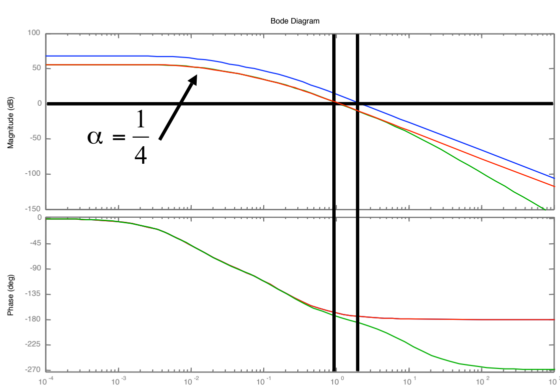
$$P(s) = \frac{1/m}{s + b/m} \times \frac{r}{s+a}$$

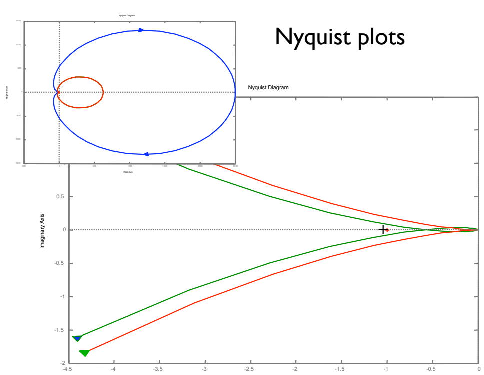
$$C(s) = \alpha\left(K_p + \frac{K_i}{s + 0.01}\right)$$

$$G(s) = \frac{10}{s+10}$$

Approach: Increase phase margin

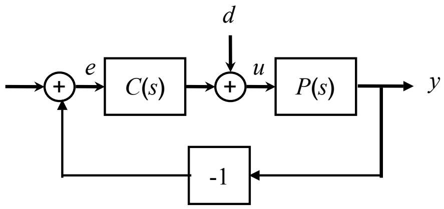
- Increase phase margin by reducing gain ⇒ can accommodate new sensor dynamics
- Tradeoff: lower gain at low frequencies ⇒ less bandwidth, larger steady state error

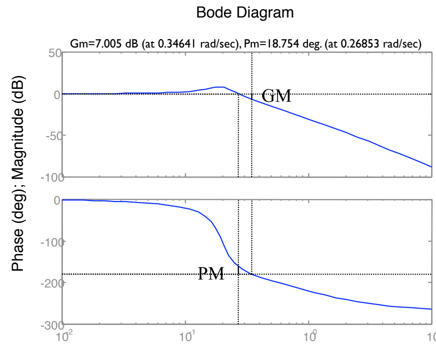




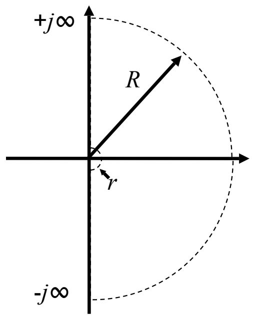
Nyquist plots

Summary: Loop Analysis of Feedback Systems





- Nyquist criteria for loop stability
- Gain, phase margin for robustness



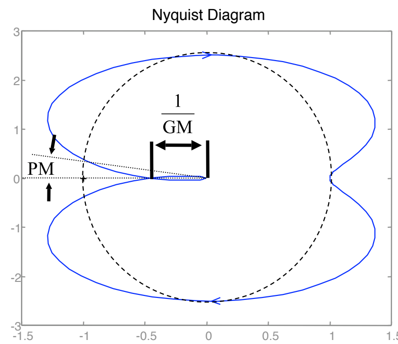
Thm (Nyquist).

P # RHP poles of $L(s)$

N # CW encirclements

Z # RHP zeros

$$Z = N + P$$



```
% L7_1_loopanal.m
% RMM, 8 Nov 02
%
% Required files: none

%%
%% Cruise controller
%%
%% This is the cruise controller that we studied in HW #2, 3, 4.  It uses
%% a modified PD control law.  The main modification is replacing the
%% integrator with a high gain, low pass filter to make the plots show
%% the features more clearly.
%%

% 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
Ki = 50;                                % integral gain
Kp = 1000;                              % proportional gain

% Dynamics
veh = tf([1/m], [1 b/m]);               % vehicle
eng = tf([r], [1 a]);                   % engine
ctr = tf([Kp Ki], [1 0.01]);            % control: PI w/ LF pole
cruise = ctr*eng*veh;                   % loop transfer function

%% Plot out the Nyquist plot for the system
global AM_NYQUIST_PLAIN;
figure(1); amnyquist(cruise);                   % standard plot
figure(2); amnyquist(cruise, {1,1e5});          % zoomed plot

%% Speed sensor dynamics (use standard MATLAB command this time)
figure(3); lag = tf([10], [1 10]);              % G(s) = 10/(s+10)
figure(4); bode(cruise, cruise*lag);            % Plot old and new Bode
figure(5); nyquist(cruise, cruise*lag);         % Nyquist plots for old and new
figure(6); nyquist(cruise, cruise*lag, {1,1e5});% Zoomed version

%% Design example - change the gain on the plots
figure(7); bode(cruise, 0.25*cruise*lag, 0.25*cruise);
figure(8); nyquist(cruise, 0.25*cruise*lag, 0.25*cruise*lag);
figure(9); nyquist(0.25*cruise*lag, 0.25*cruise*lag, {0.5,1e5});
```