ME 4555 - Lecture 32 - More control design with Bode

So far, we saw several techniques for control and several performance metrics that can be used:

transient response

- peak time t_p
- percent overshoot PO
- settling time t_s

use root locus or step response

frequency response

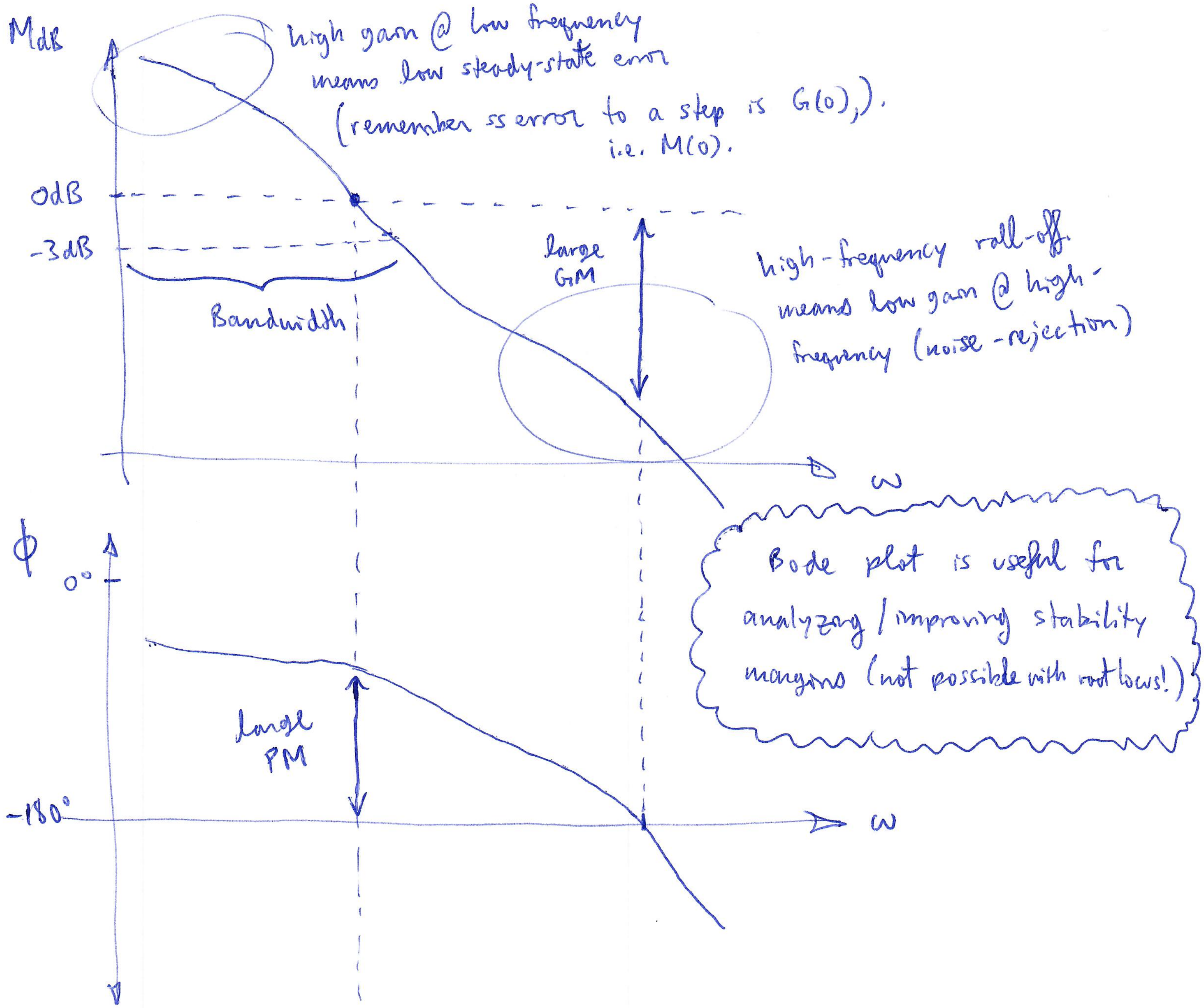
- bandwidth / gain crossover frequency
- gain / phase margin.

use open-loop Bode plot

The one-stop-shop for all of this information is the "control system designer" in Matlab, accessible using the command "sisotool". By default, it includes:

- CL step response (r2y) (for t_p, PO, t_s)
- root locus plot (to see pole locations)
- open-loop Bode plot (for bw, stability margins).

desirable frequency response (typical)



large PM + GM means sensitivity to errors.
[we are more robust to model errors, loop delays, etc.].

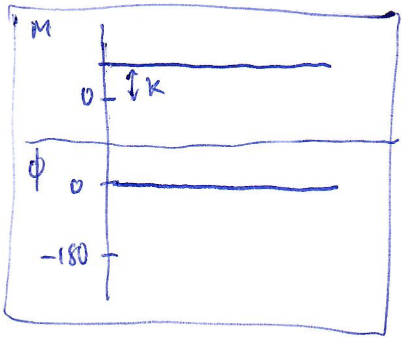
Bandwidth can go either way:

- High : very responsive / fast system, but very susceptible to high-frequency noise or unmodeled high-frequency dynamics.
- Low : slow / sluggish system, and unaffected by high-frequency noise.

How does a PID controller affect frequency response?

P-control : raises magnitude plot ; phase is unaffected.

$C(s) = K$



- can reduce PM + GM
- make system more unstable (depending on K)

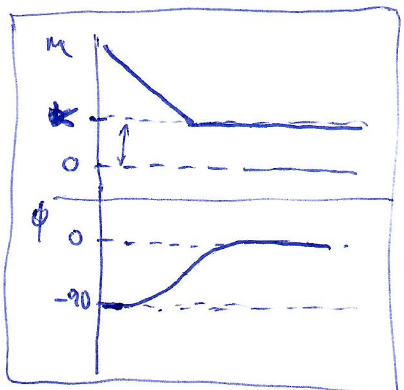
\+ improve steady-state error

± increase bandwidth

- increase high-frequency gain.

PI-control

$C(s) = \frac{K(s+z)}{s}$



\+ improve steady-state error

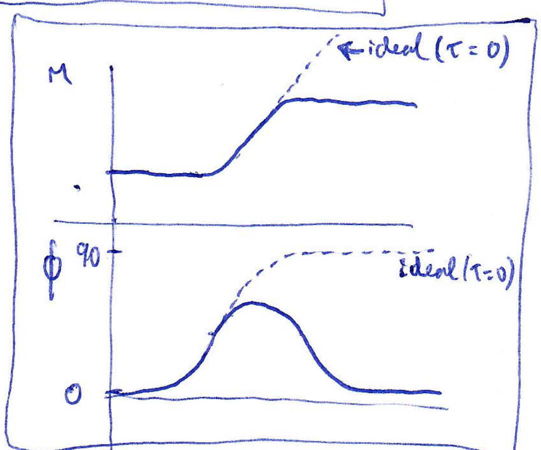
- can make system more unstable (depending on K)

± same issues as P control at high frequencies.

- can reduce PM.

PD-control

$C(s) = \frac{K(s+z)}{(\tau s+1)}$



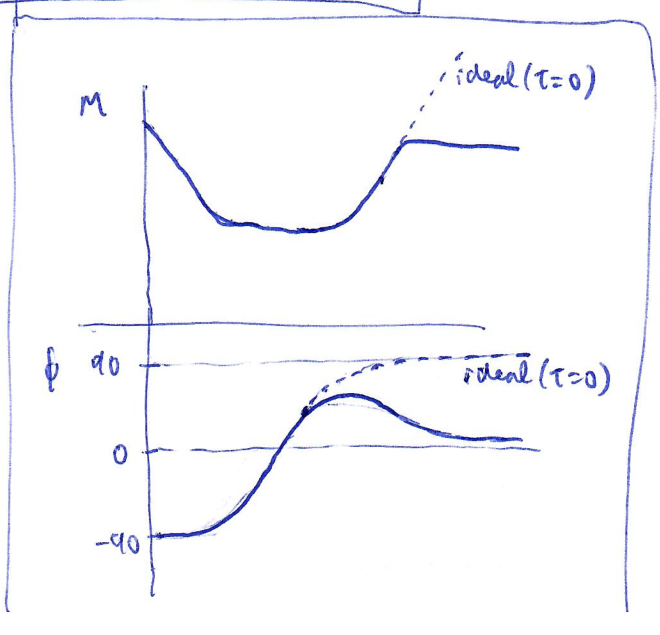
- can increase high-frequency gain

\+ can improve phase margin.

± may reduce bandwidth

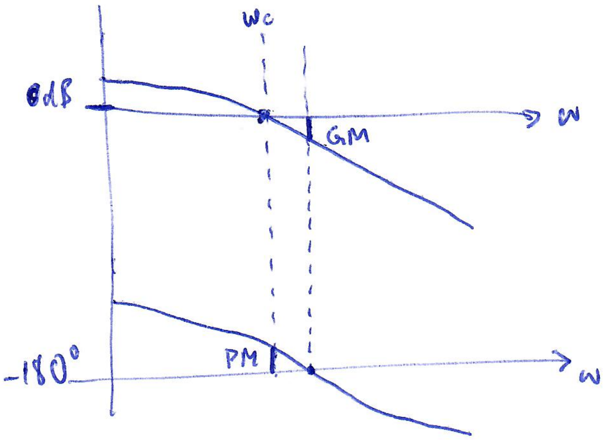
PID

$C(s) = \frac{K(s+z_1)(s+z_2)}{s(\tau s+1)}$



Combination of all effects above, depending on how poles/zeros/gain are selected.

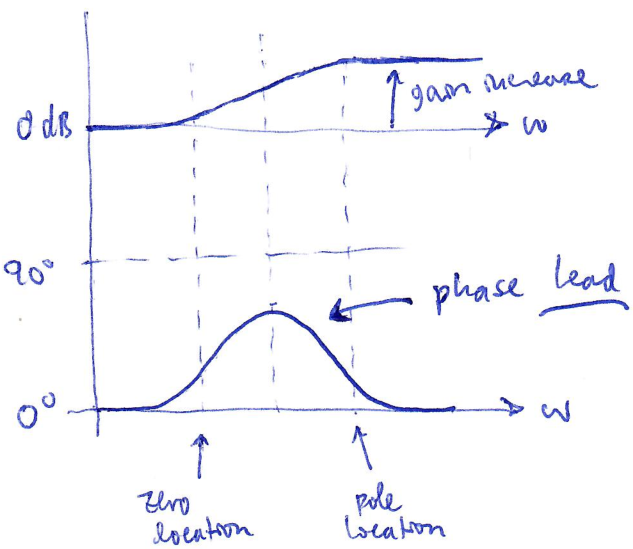
two ways to improve stability margins



if we want larger phase + gain margins we can just decrease gain K, but this will worsen the steady-state error (we want high gain @ low freq.)

Lead compensator

$C(s) = K\left(\frac{s+a}{s+b}\right)$ with $|a| < |b|$ (low freq. zero, high freq. pole).

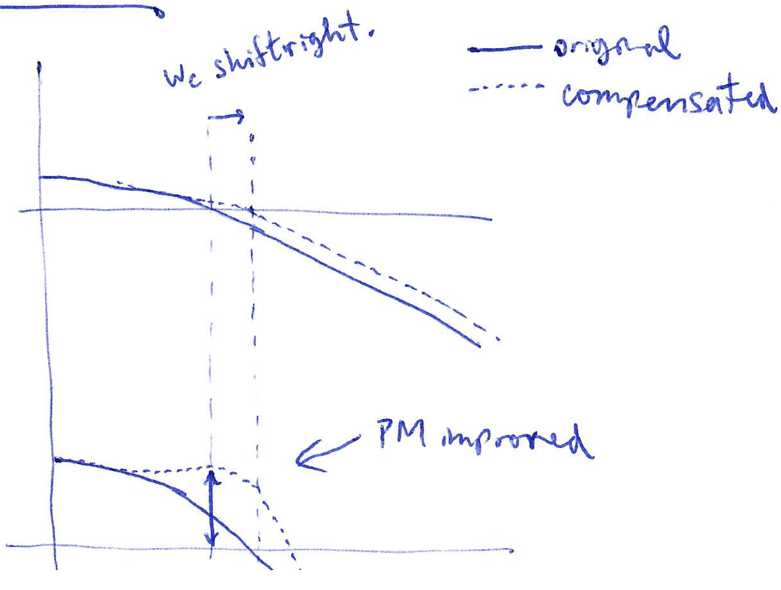


→ pick zero + poles on either side of ω_c to increase phase margin (increase ϕ @ ω_c)

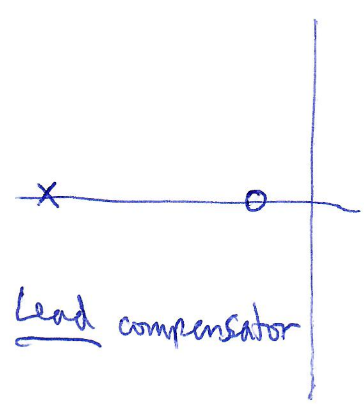
→ will typically increase gain also, so ω_c will shift right (increased bw)

→ may increase high-frequency noise gain (usually bad).

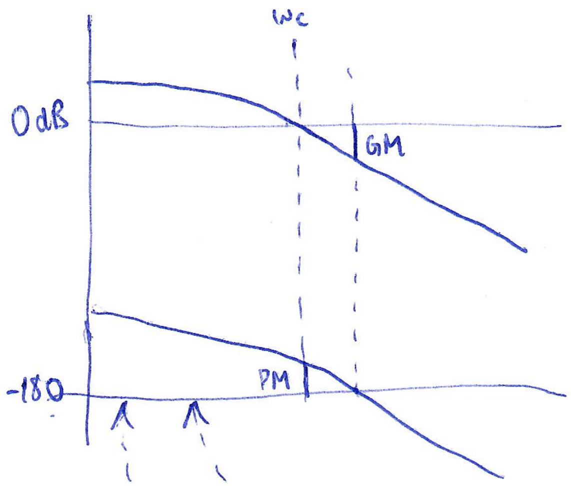
Net result.



root locus adds p+z :

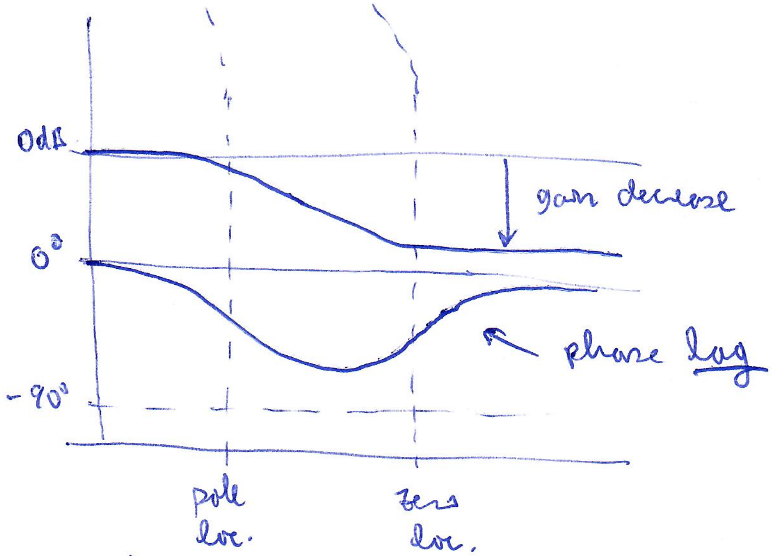


original system



Instead of boosting the phase at w_c to increase PM, we can instead try to decrease gain to improve GM and also shift w_c left (slower) so that PM improves. We do this with a lag compensator.

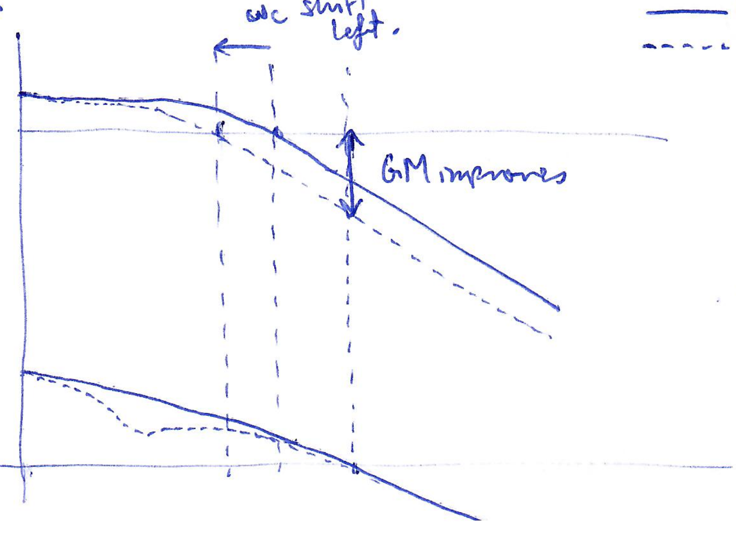
Lag compensator $C(s) = K\left(\frac{s+a}{s+b}\right)$ with $|a| > |b|$ (low freq pole, high freq zero)



place to the LEFT of w_c!

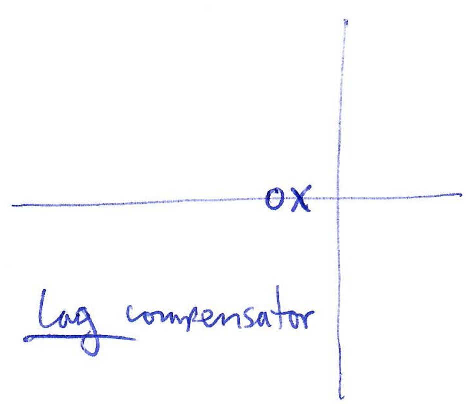
- → pick pole + zero to the left of w_c
  - → phase dip will not affect PM significantly.
- → gain will go down, so this moves w_c left, so decreased bandwidth (slower response)
- → GM will improve.

result

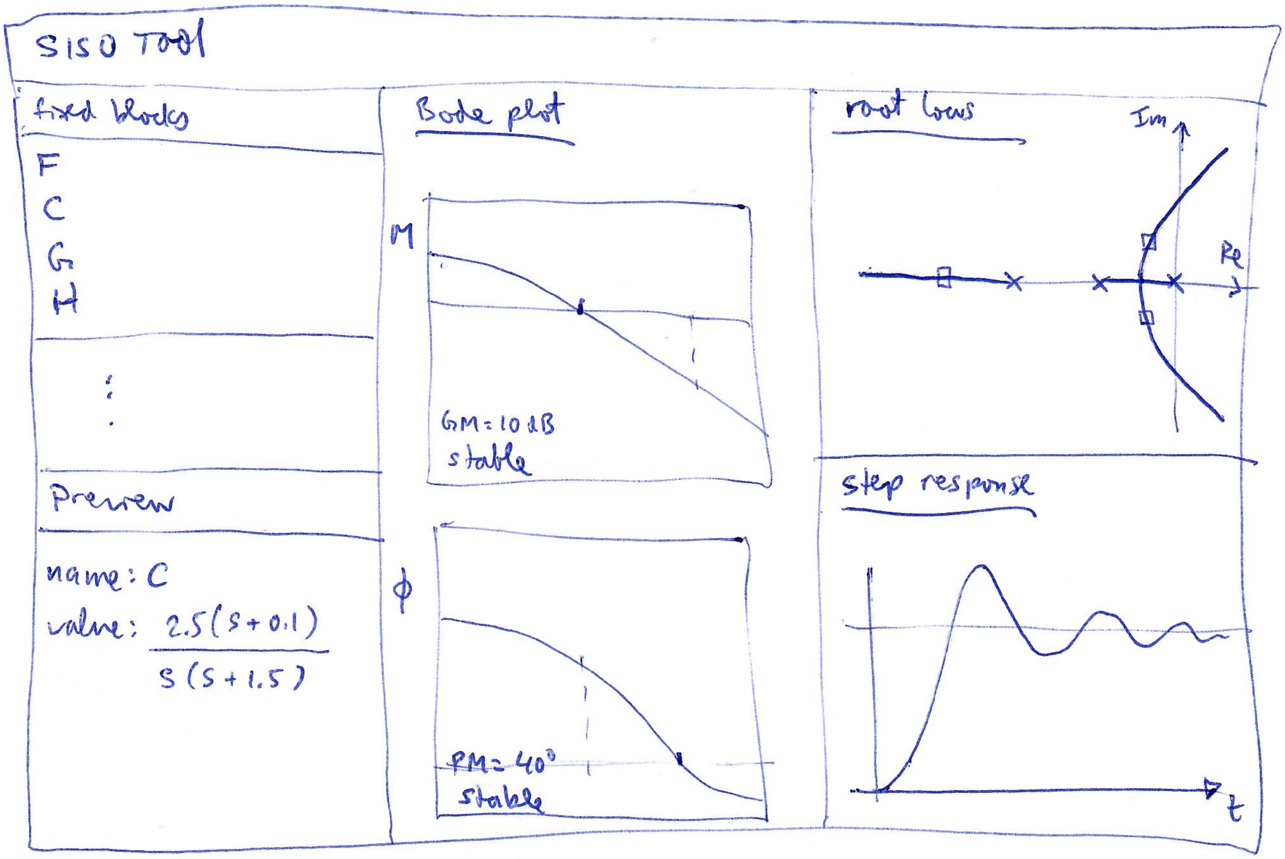


——— original
- - - - compensated

root locus adds p+z:



Design in Matlab allows easy graphical tuning via the "sisotool" command:



Bode plot: - gives stability margins, bandwidth.
- plot can be dragged up/down to change gain.
- can add compensator poles/zeros.

Root locus: - shows CL pole locations in complex plane.
- gives pole angles, shows stability
- can add compensator poles/zeros.

Step response: - shows transient response
- get performance: t_p, PO, t_s