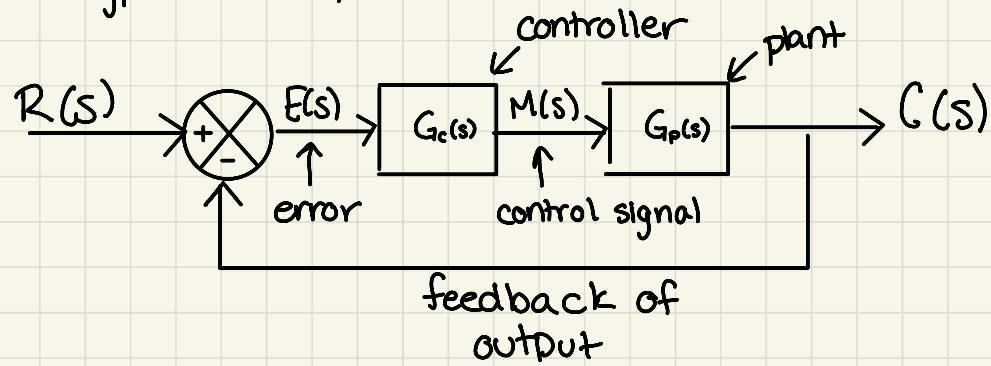
Let's take a moment and look at the block diagram for a typical simple control architecture



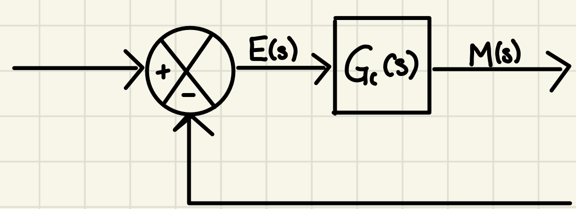
$$\frac{C(s)}{R(s)} = \frac{G_c(s)G_p(s)}{1 + G_c(s)G_p(s)}$$

We've learned how to derive transfer functions for dynamic systems

⇒ these will be our "plant"

We also will have blocks to represent our controllers

Let's talk about our most basic controller forms. These three controllers will all operate on the error signal, meaning they will be represented in the block diagram as



1) Porportional Control $\frac{M(s)}{E(s)} = G_c(s) = K_p$

porportional gain

- the control signal is adjusted as a product of K_p and the error.
- As the desired value approaches, the error goes to 0 and thus the controller reduces it's influence on the system

2) Integral Control $\frac{M(s)}{E(s)} = G_c(s) = \frac{K_i}{s}$ ← integral gain

- operates on the integral of the error
- accelerates progress toward the goal state
- prone to overshoot → once overshoot occurs the integral starts to reduce

3) Derivative Control $\frac{M(s)}{E(s)} = G(s) = K_d s$

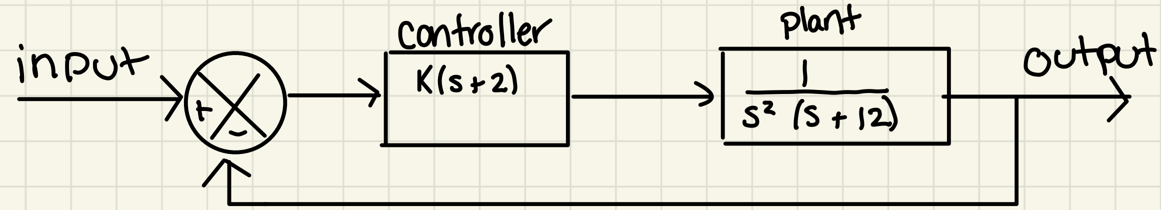
derivative gain

- operates on the derivative (slope) of the error
- works to smooth out oscillations and reduce overshoot

Common controller combos:

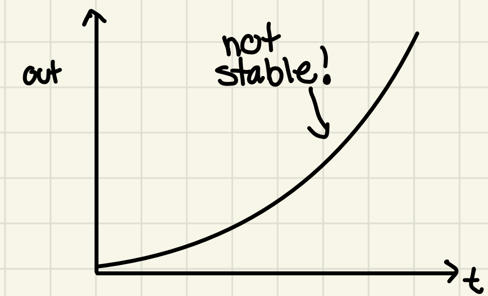
P, I, PI, PD, PID

Let's look at a quick example:



To start → let's look at the plant alone. Let's put this in Matlab and see how it responds to a step input.

```
t=0:.01:10
num=[1]
den =[1  12  0  0]
sys = tf(num, den)
y = step(sys, t)
plot (t,y)
grid
```



Now let's use the block diagram to find the transfer function with the controller added

$$G(s) = \frac{K(s+2)\left(\frac{1}{s^2(s+12)}\right)}{1 + K(s+2)\left(\frac{1}{s^2(s+12)}\right)} = \frac{K(s+2)}{s^2(s+12) + K(s+2)} = \frac{K(s+2)}{s^3 + 12s^2 + Ks + 2K}$$

Now, let's try this one with some different values of K

open firstcontrollerexample.m try K=1, 2, 5, 10, 20

1) What Kind of controller is this?
2) What did you observe as K increases?

A few things to notice:

- We observe that adding the controller changed the form of the response - zeros, poles, behavior

- We observe that changing our gain K improved our stability
  - → reduced time to steady state
  - → frequency of response / decay rate
  - → amplidue of oscillations

- We improved our controller performance by trial & error

- Ideally, we'd like to quantify this

  →analysis to quantify the transient response

  →analysis to help us tune the controller

The matlab is up on Canvas - feel free to play on your own!

→Change the coefficients in the controller
→Change the form of the controller
→try a different plant

Let's do a quick pause and review what we've covered so far.

- We want to design control systems. To do this, we said we had to be able to:

  1) Understand our system
  2) Model our system
  3) Analyze/monitor our system
  4) Influence our system

Up to this point, we've focused on 1 & 2, with a little bit of teasers on 3

We have a set of tools to help us develop models in both the time domain and s-domain, and we have a way to build models for systems from their components.

We also have taken some time to try to understand the relationship of these models to the physical world

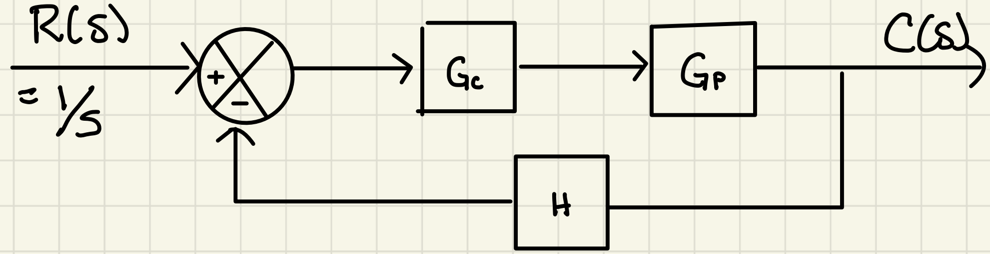
Now we are going to move onto analyzing, but with the context of the goal of controlling these systems

Before we launch into this, let's do a motivating example!

Let's do one more!

Matlab
oursecondcontroller.m

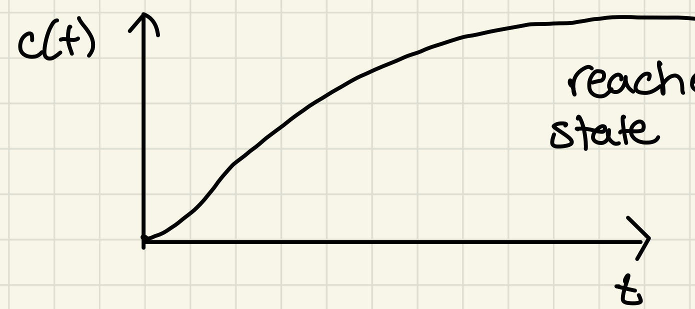
desired
→ state $c(t) = 1$



$$G_p = \frac{1}{s^2 + 3s + 1}$$

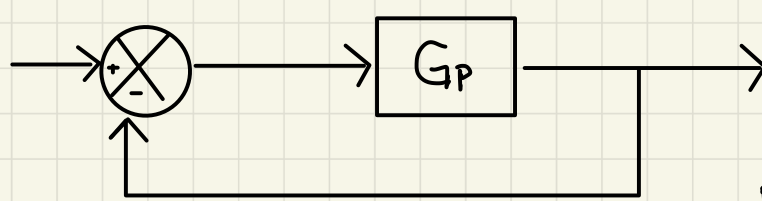
poles $s = -2.62, -.382$

the response of G_p to a step is



reaches steady state in ~15 sec
$c(t) = 1$

now let's add feedback, but no control
(H = 1)

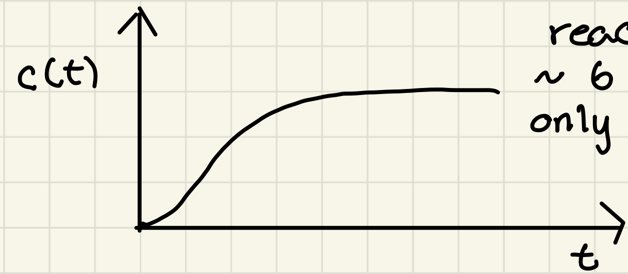


$$M_{nc} = \frac{1}{s^2 + 3s + 2}$$

poles: $s = -2, -1$

Let's find the transfer function M_nocontrol

In response to a step of 1

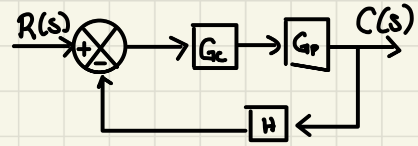


reaches SS in ~ 6 sec, but only gets to .5!

↳ this has what we call steady state error of .5

So let's add the controller

$$G_c = K_p + \frac{K_i}{s} + K_d s$$



and make a new transfer function.

Let's play with K_p, K_i, K_d and see what happens! (incrament & run)

K_p	1	2	3	16	10	10	10	10
K_i	0	0	0	0	1	4	4	4
K_d	0	0	0	0	0	0	1	3

↑ same as no control

note the transfer function changes

oursecond controller.m

⇒ We need to be able to quantify and predict key response features!

Stop & Think:

Based on your observations, what kinds of things might we want to able to quantify? (and influence!)

- SS error
- overshoot
- time to SS
- frequency of oscillations
- rates of rowth /decay

⇒ AND we need to understand now these values relate to controller architecture and gains

⇒ Can we use it all to determine our controller design insted of using trial & error?

=> spoiler alert → yes!