goal: techniques for analysis, design, & implementation of the most ubiquitous control architecture

* this is my last lecture
* Thu Dec 5: in-class review
* Tue Dec 10: project due @ 11:59p
* Thu Dec 12: final exam @ 10:30a

Based on a survey of over eleven thousand controllers in the refining, chemicals and pulp and paper industries, 97% of regulatory controllers utilize a PID feedback control algorithm.

L. Desborough and R. Miller, 2002 [DM02a].

1°. essentials of feedback control — [AMv2 Ch 11]

1¹. a simple controller — [Nu7 Ch 9.4]

1². implementation issues

1°. essentials of feedback control

- take a step back and reflect on:

what feedback does	& how it does it
– make output (y) track reference (r)	– the measured signal y and commanded u, r including their time histories
– reject disturbances to inputs (v) and outputs (w)	– prior knowledge, eg of process dynamics, statistics of disturbances, model uncertainty
– provide robustness to model inaccuracies, uncertain design parameters (i.e. spec sheet tolerances)	PID design tries not to rely on this

---

1¹. a simple controller

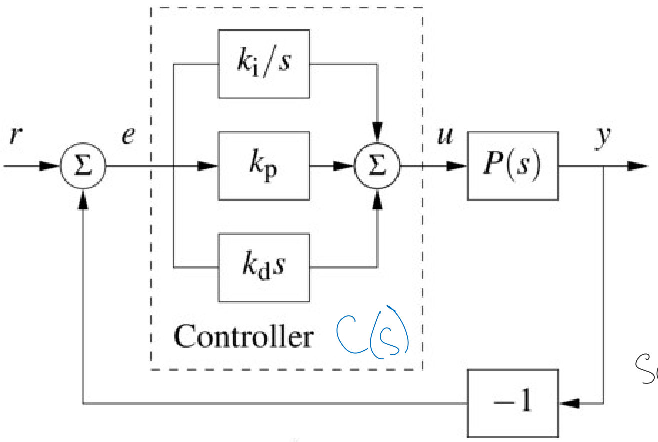
1!. a simple controller

- consider the simplest controller that doesn't rely on prior knowledge

→ use linear combination of: P – present error $e = r - y$

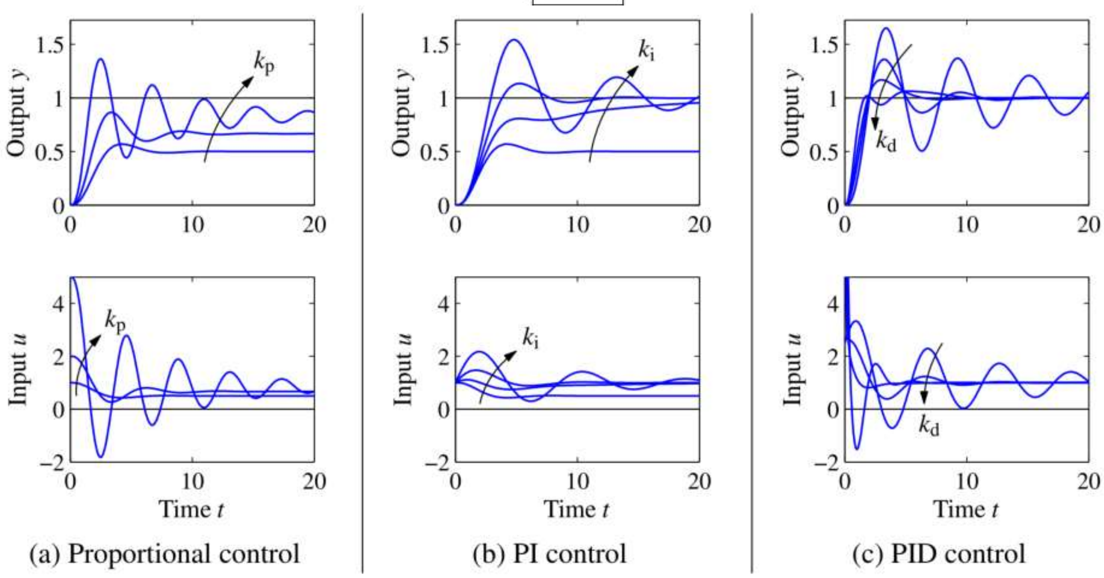
I – (integral of) past error

D – (prediction of) future error using linear extrapolation



- $C(s) = k_P + \frac{k_I}{s} + k_D s$

so $u(t) = k_P e(t) + k_I \int_0^t e(\tau) d\tau + k_D \frac{de(t)}{dt}$



→ describe trends in transient & steady-state as k_P, k_I, k_D vary

Figure 11.2: Responses to step changes in the reference value for a system with a proportional controller (a), PI controller (b) and PID controller (c). The process has the transfer function $P(s) = 1/(s+1)^3$, the proportional controller has parameters $k_p = 1$, 2 and 5, the PI controller has parameters $k_p = 1$, $k_i = 0$, 0.2, 0.5, and 1, and the PID controller has parameters $k_p = 2.5$, $k_i = 1.5$ and $k_d = 0$, 1, 2, and 4.

1². implementation issues

- we have an elegant mathematical formula:) signals lost / interrupted / etc

$u(t) = k_P e(t)$ proportional

$+ k_I \int_0^t e(\tau) d\tau$ integral

– initial error $e(0)$?

– "windup", – overflow, large inputs

$+ k_I \int_0^t e(\tau)\, d\tau$ integral

$+ k_D \frac{d}{dt} e(t)$ derivative

— "windup" — overflow, large inputs

— real signals, eg ref., aren't differentiable!

— measurement noise

→ what practical issues could arise when this formula is implemented on actual hardware, eg digital microcontroller or analogue RLC / op-amp circuit?

* how to choose gains k_P, k_I, k_D in reality?

1°. use a simulation/model of your system

2°. use a gain-tuning procedure

- the first / most widely-used rules were developed by Ziegler & Nichols in the 1940s

— heuristic rules hand-designed by trial & error

— guaranteed to work for $P(s) = \frac{e^{-sT}}{s+a}$ — delay duration T — one pole @ $-a$

1°. set $k_I, k_D = 0$

2°. increase k_P until system oscillates w/ period T_c → k_P^*

3°. Nyquist stability criterion tells us that open-loop transfer function $L(s) = C(s)P(s) = k_P^* P(s)$ passes through critical point $-1 \in \mathbb{C}$ at frequency $\omega_c = \frac{2\pi}{T_c}$

→ use the following table to choose gains:

Type	k_p	$T_i = 1/k_I$	$T_d = 1/k_D$
P	$0.5k_p^*$		

Type	k_p	$T_i = 1/k_I$	$T_d = 1/k_D$
P	$0.5k_c^*$		
PI	$0.4k_c^*$	$0.8T_c$	
PID	$0.6k_c^*$	$0.5T_c$	$0.125T_c$

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