PID Control & Tuning Rules for PID Controllers

Textbooks:

Modern Control Engineering, K. Ogata

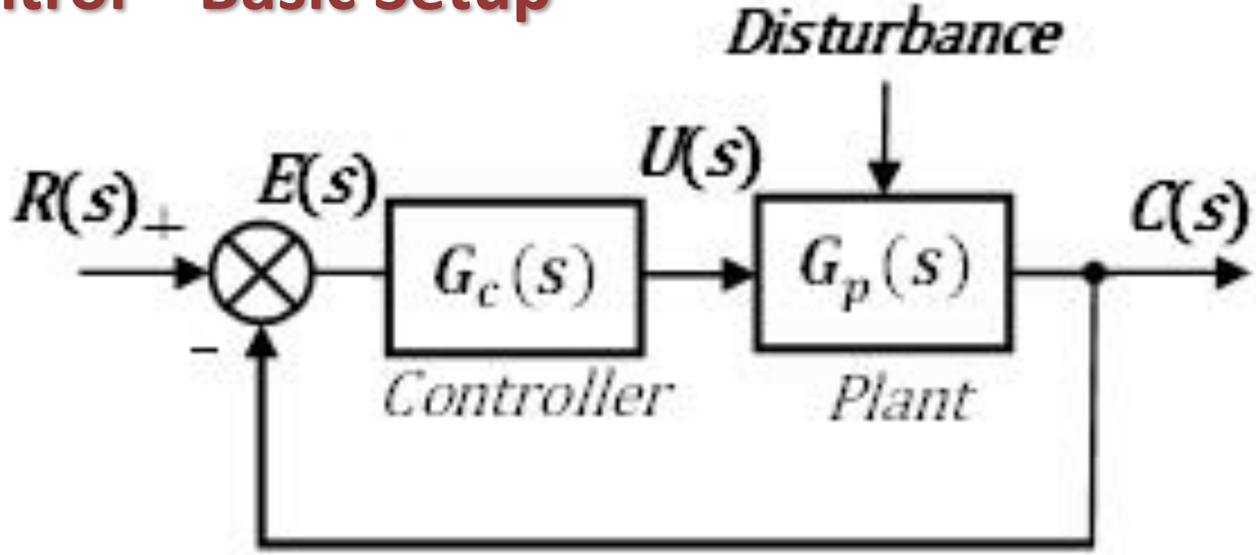
Feedback Control of Dynamic Systems, Franklin et al.

Instructor: Dr. Şeref Naci Engin

PID Control – Basics

- Currently, more than half of the controllers used in industry are PID controllers.
- In the past, many of these controllers were analog: OPAMPS & passive components
- However, many of today's controllers use digital signals and computers: DSP, μC, μP, PLC
- When a **mathematical model of a system is available**, the parameters of the controller can be **explicitly determined**.
- When a mathematical model is **unavailable**, the parameters must be determined **experimentally**.
- *Controller tuning* is the process of determining the controller parameters which produce the desired output.
- *Controller tuning* allows for optimization of a process and minimizes the error between the output variable of the process and its set point.

PID Control – Basic Setup



$$G_c(s)=\frac{U(s)}{E(s)} = K_p + \frac{K_i}{s} + K_d s = K_p\left(1 + \frac{1}{\frac{K_p}{K_i}s} + \frac{K_d}{K_p}s\right)$$

$$G_c(s)= K_p\left(1 + \frac{1}{T_i s} + T_d s\right)$$

Equivalently,

$$u(t) = K_p\left(e(t) + \frac{1}{T_i}\int_0^t e(t)dt + T_d\frac{de(t)}{dt}\right) + b$$

Introduction to PID Tuning

- If a mathematical model of the plant can be derived, then it is possible to apply various design techniques for determining parameters of the controller that will meet the transient and steady-state specifications of the closed-loop system.
- However, if the plant is so complicated that its mathematical model cannot be easily obtained, then an analytical approach to the design of a PID controller is not possible.
- Then we must switch to experimental approaches to the tuning of PID controllers.

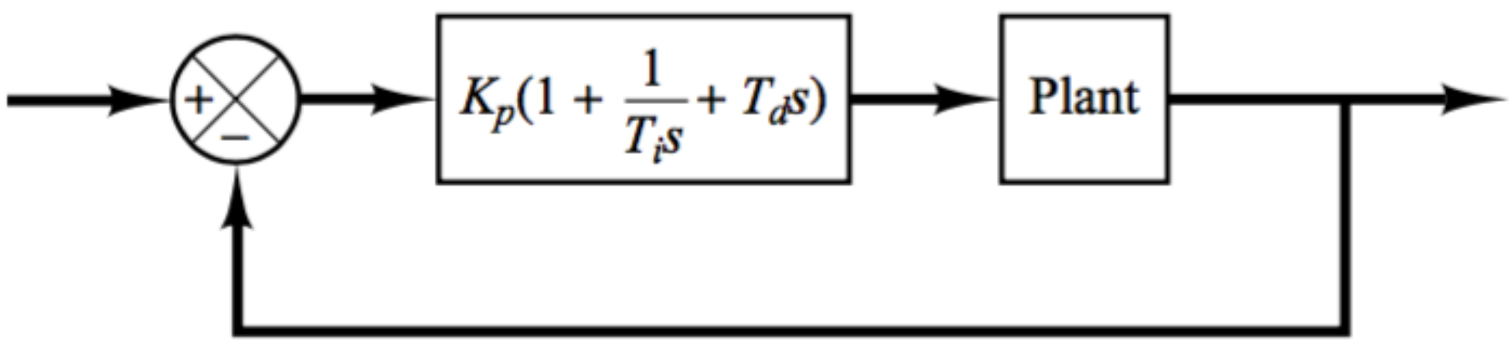
PID Control – Tuning

- Types of controller tuning methods include the **trial and error** method, and **process reaction curve** methods.
- The most common classical controller tuning methods are the **Ziegler-Nichols** and **Cohen-Coon methods**.
- These methods are often used when the mathematical model of the system is **not available**.
- The Ziegler-Nichols method can be used for both closed and open loop systems, while Cohen-Coon is typically used for open loop systems.
- A closed-loop control system is a system, which uses feedback control. In an open-loop system, the output is not compared to the input.

- **Two Ziegler-Nichols methods are available to be used:**
  - For the systems that **cannot produce oscillations** and
  - The ones that can produce **full oscillations** for a particular value of gain.

Introduction to PID Tuning…

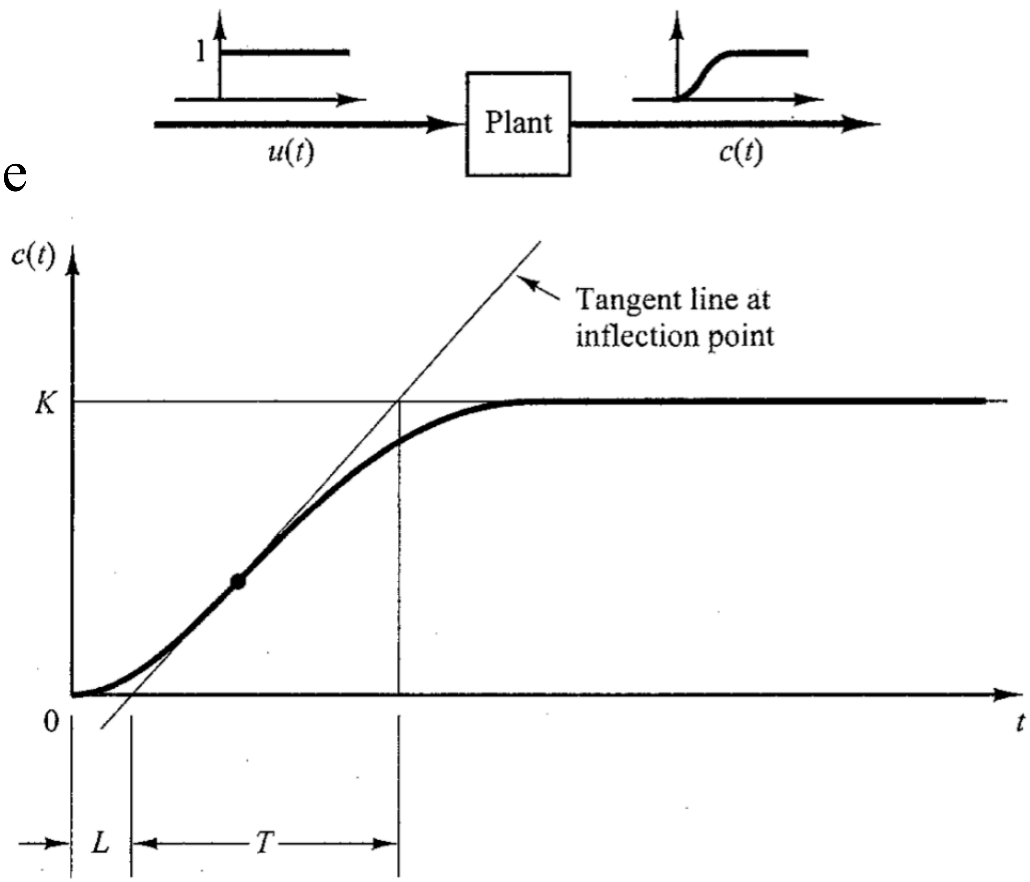
- The process of selecting the controller parameters to meet given performance specifications is known as **controller tuning**.
- **Ziegler and Nichols** suggested rules for tuning PID controllers (meaning to set values K_p, T_i, and T_d,) based on experimental step responses or based on the value of K_p, that results in marginal stability when only proportional control action is used.
- Ziegler-Nichols rules, which are briefly presented in the following slides, are useful when mathematical models of plants are not known.
- The initial values of PID coefficients can be estimated with **Ziegler-Nichols** rules. Then they can then be fine tuned.



Ziegler-Nichols Rules for Tuning PID Controllers-1

First Method:

- The response of the plant to a unit-step input is obtained experimentally.
- If the plant involves neither integrator(s) nor dominant complex-conjugate poles, then such a unit-step response curve may look *S*-shaped, as shown in the figure.
- This method applies if the response to a step input exhibits an *S*-shaped curve.
- Such step-response curves may be generated experimentally or from a dynamic simulation of the plant.



Ziegler-Nichols Rules for Tuning PID Controllers-2

First Method – *cont'd*:

- The S-shaped curve may be characterized by two constants, delay time L and time constant T.
- The delay time and time constant are determined by drawing a tangent line at the inflection point of the S-shaped curve and determining the intersections of the tangent line with the time axis and line $c(t) = K$, as shown in the figure.
- The transfer function $C(s)/U(s)$ may then be approximated by a first-order system with a transport lag as follows:

$$\frac{C(s)}{U(s)} = \frac{Ke^{-Ls}}{Ts+1}$$

- Ziegler and Nichols suggested to set the values of $\boldsymbol{K_p}$, $\boldsymbol{T_i}$, and $\boldsymbol{T_d}$ according to the formula shown in the following table.

Ziegler-Nichols Rules for Tuning PID Controllers-3

First Method – *cont'd*:

Ziegler-Nichols Tuning Rule Based on Step Response of Plant (First Method)

Type of Controller	K_p	T_i	T_d
P	$\frac{T}{L}$	∞	0
PI	$0.9\frac{T}{L}$	$\frac{L}{0.3}$	0
PID	$1.2\frac{T}{L}$	$2L$	$0.5L$

Ziegler-Nichols Rules for Tuning PID Controllers-4

First Method – *cont'd*:

Notice that the PID controller tuned by the first method of Ziegler-Nichols rules gives

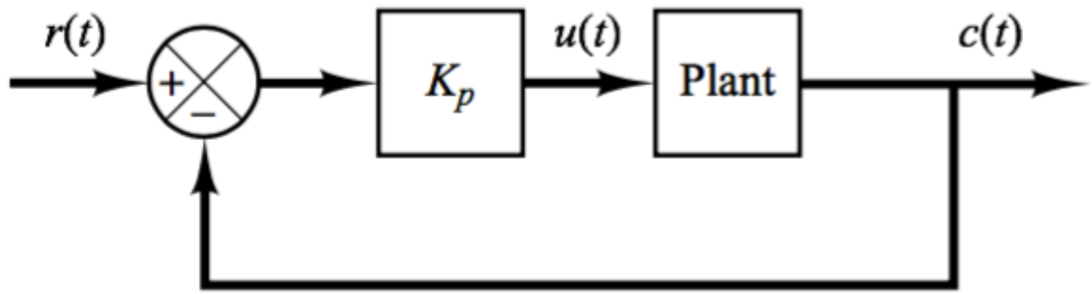
$$
\begin{aligned}
G_c(s) &= K_p\left(1 + \frac{1}{T_i s} + T_d s\right) \\
&= 1.2\frac{T}{L}\left(1 + \frac{1}{2Ls} + 0.5Ls\right) \\
&= 0.6T\frac{\left(s + \frac{1}{L}\right)^2}{s}
\end{aligned}
$$

Thus, the PID controller has a pole at the origin and double zeros at $s = -1/L$

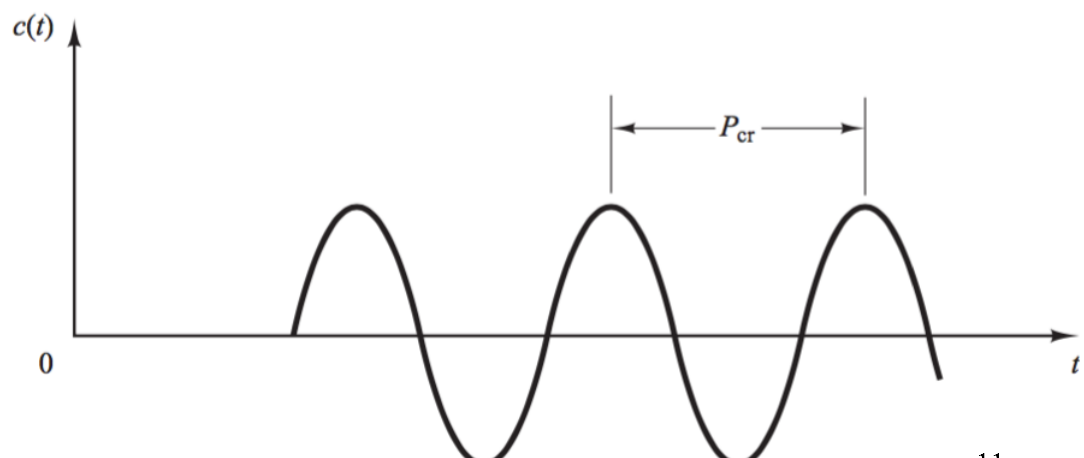
Ziegler-Nichols Rules for Tuning PID Controllers-5

Second Method:

- We first set $T_i = \infty$ and $T_d = 0$. Using the proportional control action only,



- Then increase K_p, from 0 to a critical value K_{cr} at which the output exhibits sustained oscillations.
- If the output does not exhibit sustained oscillations for whatever value K_p may take, then this method does not apply.
- The critical gain K_{cr}, and the corresponding period P_{cr} are determined experimentally.



Ziegler-Nichols Rules for Tuning PID Controllers-6

Second Method, *cont'd*:

Ziegler-Nichols Tuning Rule Based on The critical gain K_{cr} and the critical period P_{cr} (Second Method)

Type of Controller	K_p	T_i	T_d
P	$0.5K_{cr}$	∞	0
PI	$0.45K_{cr}$	$\frac{1}{1.2}P_{cr}$	0
PID	$0.6K_{cr}$	$0.5P_{cr}$	$0.125P_{cr}$

Ziegler-Nichols Rules for Tuning PID Controllers-7

Second Method, *cont'd*:

Notice that the PID controller tuned by the second method of Ziegler-Nichols rules gives

$$G_c(s) = K_p\left(1 + \frac{1}{T_i s} + T_d s\right)$$

$$= 0.6K_{\mathrm{cr}}\left(1 + \frac{1}{0.5P_{\mathrm{cr}}s} + 0.125P_{\mathrm{cr}}s\right)$$

$$= 0.075K_{\mathrm{cr}}P_{\mathrm{cr}}\frac{\left(s + \frac{4}{P_{\mathrm{cr}}}\right)^2}{s}$$

Thus, the PID controller has a pole at the origin and double zeros at $s = -\frac{4}{P_{cr}}$

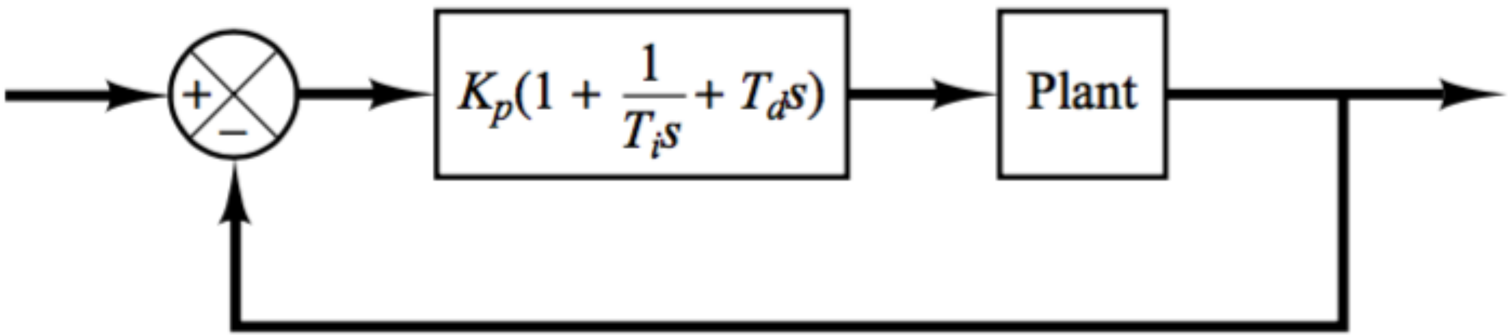
Example-1

Design a PID controller for the system where, $G_p(s) = \frac{6}{(s+1)(s+2)(s+3)}$

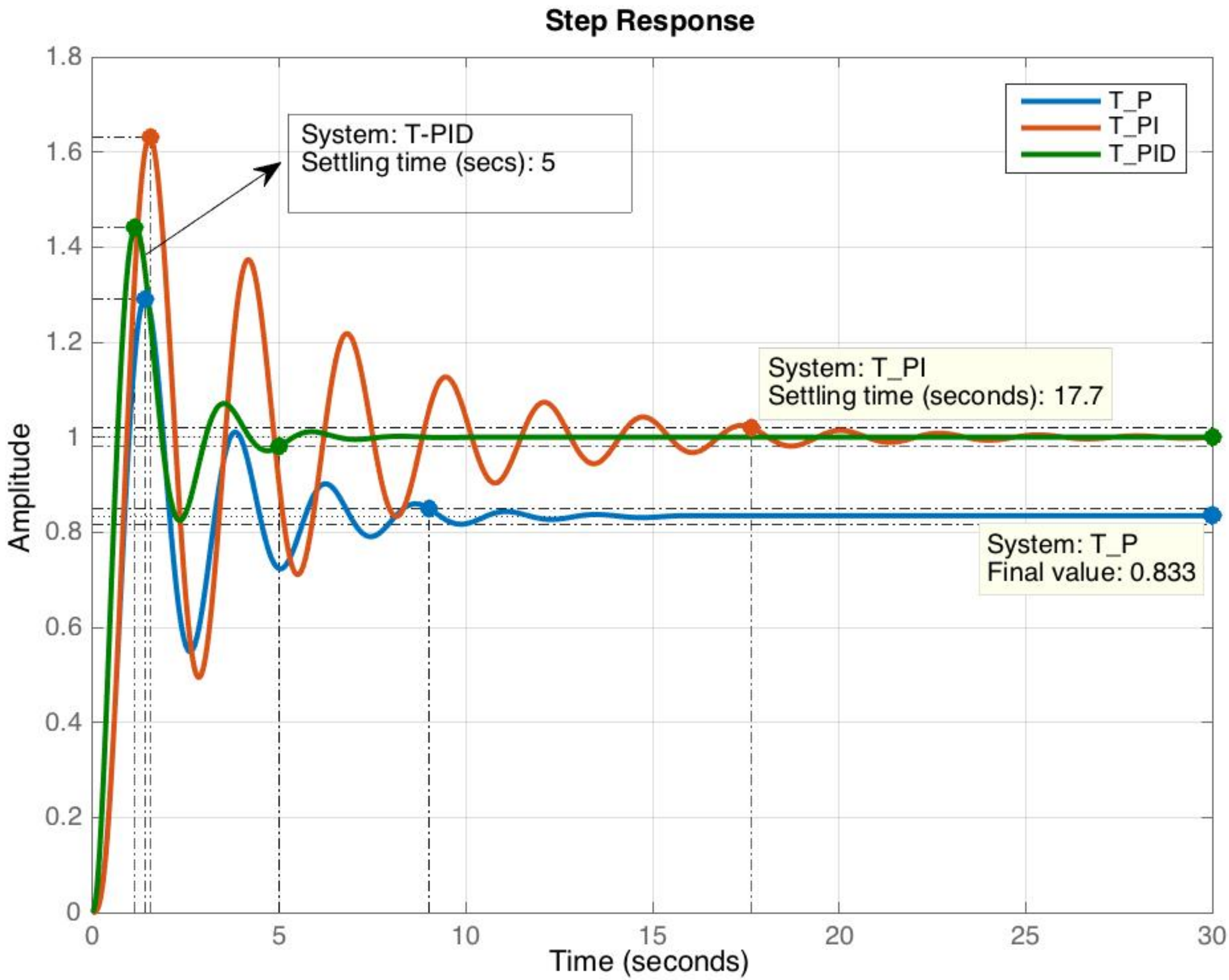
Answer: $K = 6, T_i = 0.947 \text{ sec}, T_d = 0.237 \text{ sec} \quad (K_{cr} = 10; P_{cr} = 1.8945)$

$$G_P(s) = 5; G_{PI}(s) = \frac{4.5(s+0.63)}{s}$$

$$G_{PID}(s) = \frac{1.42(s^2 + 4.2245s + 4.461)}{s} = \frac{1.42(s+2.112)^2}{s}$$



See the results in the next two slides, the P, PI, PID controller output plots and their analyses presented in a table.



Step Response
T_P
T_PI
T_PID
System: T-PID
Settling time (secs): 5
System: T_PI
Settling time (seconds): 17.7
System: T_P
Final value: 0.833
Amplitude
Time (seconds)
0
0.2
0.4
0.6
0.8
1
1.2
1.4
1.6
1.8
0
5
10
15
20
25
30

Ziegler-Nichols Rules based PID Tuning Results: Analyses of the plots

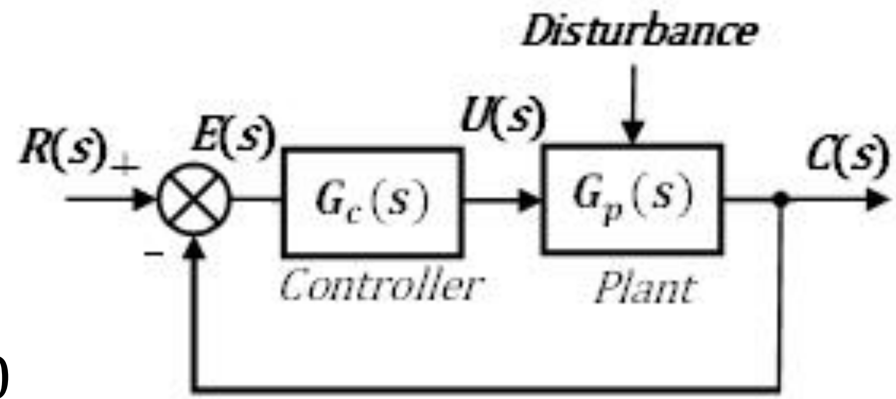
	stepinfo(T_P)	stepinfo(T_PI)	stepinfo(T_PID)
Rise Time:	0.5042	0.5362	0.4461
Settling Time:	9.0105	17.6602	4.9941
Overshoot:	54.9281	63.0729	44.0759
Peak:	1.2911	1.6307	1.4408
Peak Time:	1.3954	1.5341	1.1336

As seen, the output of the PID controller exhibits the best performance in every aspect.

Example-2

Design a P, PI and PID controller for the system given, where

$$G_p = \frac{1}{s(s+1)(s+5)}$$



Solution

The characteristic equation for the P-controlled system will be,

$$D(s) = s^3 + 6s^2 + 5s + K_p = 0$$

s^3	1	5
s^2	6	K_p
s^1	$30 - K_p$	0
s^0	K_p	

$K_p = K_{cr} = 30$

$6s^2 + 30 = 0,$

$s_{1,2} = \pm j2.236$

$\omega_{cr} = 2.236 = \frac{2\pi}{T_{cr}}$

$T_{cr} = P_{cr} = 2.81 \text{ sec}$

The alternative way to find ω_{cr}:

The characteristic equation for $K_p = K_{cr} = 30$,

$$s^3 + 6s^2 + 5s + 30 = 0$$

If we substitute $s = j\omega$

$$\omega = \omega_{cr} = 2.236$$

➔ $T_{cr} = P_{cr} = 2.81 \text{ sec}$

Example-2, *cont'd*…

Solution – ***cont'd…***

For $K_{cr} = 30$ and $P_{cr} = 2.81$, the PID coefficients can be determined via the table of the second method,

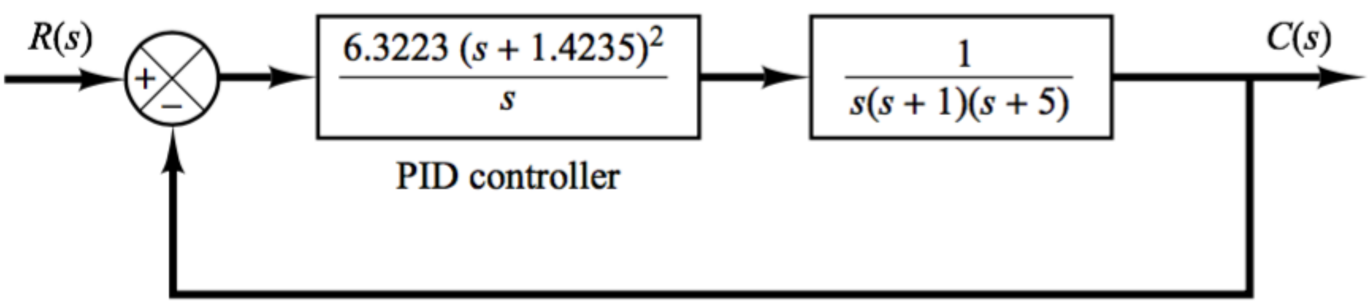
$$K_p = 0.6K_{cr} = 18$$

$$T_i = 0.5P_{cr} = 1.405$$

$$T_d = 0.125P_{cr} = 0.35124$$

The transfer function of the PID controller is thus

$$\begin{aligned} G_c(s) &= K_p\left(1 + \frac{1}{T_i s} + T_d s\right) \\ &= 18\left(1 + \frac{1}{1.405s} + 0.35124s\right) \\ &= \frac{6.3223(s + 1.4235)^2}{s} \end{aligned}$$



Problem with the Derivative Action

- When implementing the PID controller it is necessary to augment the derivative control part with a **Low Pass Filter** to avoid the non-causality:
  $\boldsymbol{G_c(s) = K_P + \frac{K_I}{s} + K_D s}$ or $\boldsymbol{G_c(s) = K_P\left(1 + \frac{1}{T_i s} + T_d s\right)}$, where $\boldsymbol{T_i = \frac{K_P}{K_I}}$ and $\boldsymbol{T_d = K_D/K_P}$, referred to as integral (reset) time and derivative (rate) time, respectively.
- Now, lets add an LPF of $\boldsymbol{\frac{N}{s+N}}$ so that the derivative part can be implemented physically,
$$\boldsymbol{G_c(s) = K_P + \frac{K_I}{s} + K_D s \frac{N}{s+N}}$$
- Derivative action can be problematic due to the noise added to the error signal. Noise signal may have a low amplitude but contains high frequency components. Derivative of a HF component yields an undesirable contribution to the output.
- More examples are left for class exercises and computer simulations.
- Another problem that has to be solved is integral windup... →

Integral windup, from Control System Design by Karl Johan Åström

$$u(t) = K\left(e(t) + \frac{1}{T_i}\int_0^t e(\tau)d\tau + T_d\frac{de(t)}{dt}\right)$$

- Although many aspects of a control system can be understood based on linear theory, some nonlinear effects must be accounted for in practically all controllers.
- Windup is such a phenomena, which is caused by the interaction of integral action and saturations.
- All actuators have limitations: a motor has limited speed, a valve cannot be more than fully opened or fully closed, etc.
- For a control system with a wide range of operating conditions, it may happen that the control variable reaches the actuator limits.
- When this happens the feedback loop is broken and the system runs as an open loop because the actuator will remain at its limit independently of the process output.
- If a controller with integrating action is used, the error will continue to be integrated. This means that the integral term may become very large or, colloquially, it “winds up”. It is then required that the error has opposite sign for a long period before things return to normal.
- The consequence is that any controller with integral action may give large transients when the actuator saturates.

https://www.cds.caltech.edu/~murray/courses/cds101/fa02/caltech/astrom-ch6.pdf

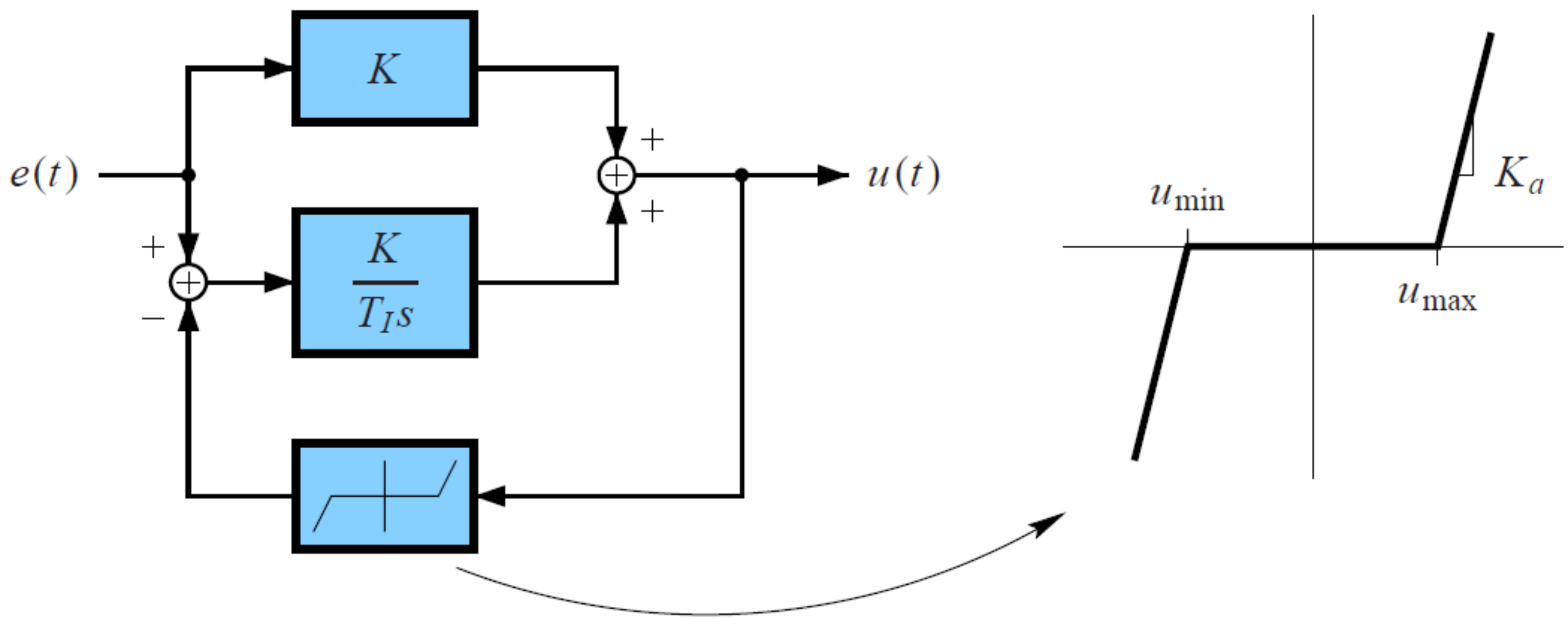
Integral windup, *cont'd…*

- **Integral windup**, also known as **integrator windup** or **reset windup**, refers to the situation in a PID feedback controller where a large change in set-point occurs (say a positive change) and the integral terms accumulates a significant error during the rise (windup),
- Thus overshooting and continuing to increase as this accumulated error is unwound (offset by errors in the other direction).
- The result is the excess overshooting.
- **This problem can be addressed by**
  - Initializing the controller integral to a desired value, for instance to the value before the problem
  - Increasing the set-point in a suitable ramp
  - Disabling the integral function until the to-be-controlled process variable (PV) has entered the controllable region
  - Preventing the integral term from accumulating above or below pre-determined bounds
  - Back-calculating the integral term to constrain the process output within feasible bounds.

Windup problem / Anti-windup mechanism

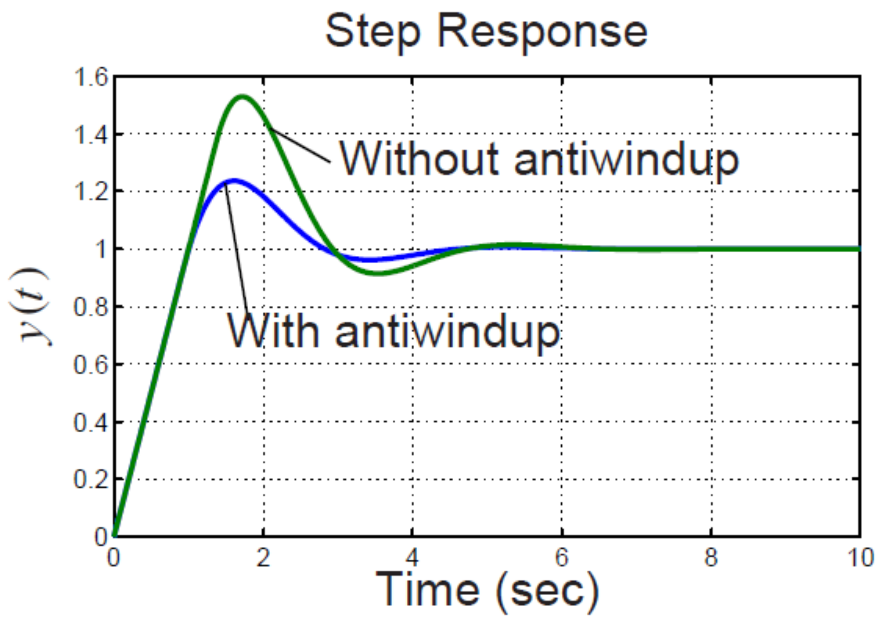
- Practical problem in PID controllers: <u>Integrator Overload</u> or <u>Integral Windup</u>
- Integrator in PI or PID control can cause problems.
- For example, suppose there is <u>saturation in the actuator</u>
  - Error will not decrease.
  - Integrator will integrate a constant error and its value will "blow up."
- **Solution:** "integrator anti-windup."

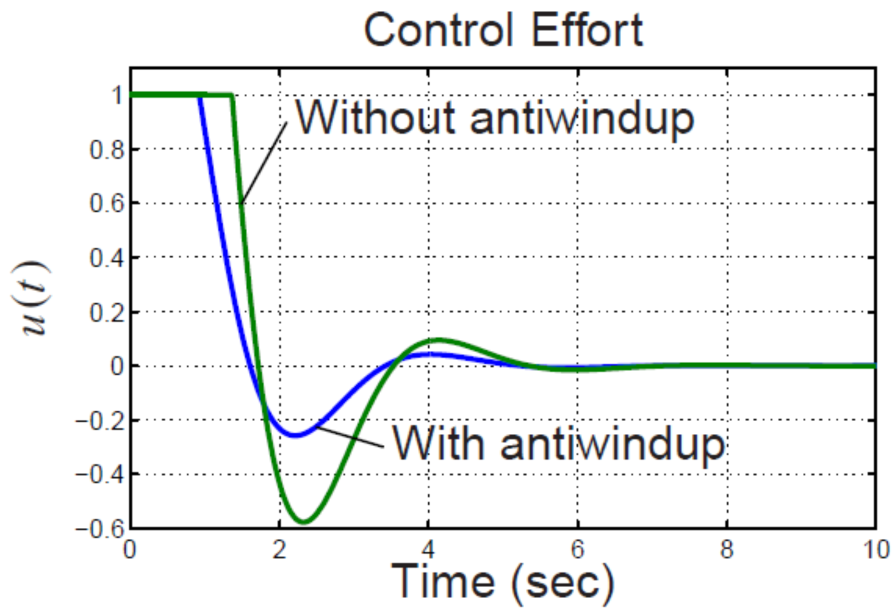
  Turn off integration when actuator saturates



Windup problem / Anti-windup mechanism

- Doing this (anti-windup) is NECESSARY in any practical implementation.
- Omission leads to bad response, instability.





End of PID Tuning, Problem with Derivative Action and Windup