Internal Model Control

Open loop Control

Consider,

$$y_{sp} \longrightarrow \boxed{q} \longrightarrow \boxed{g} \longrightarrow y$$

For, $y(s) = y_{sp}(s)$, ideally, $q(s) = \frac{1}{g(s)}$;

However, the inherent process dynamics does not allow the output y to track y_{sp} immediately.

So, $\frac{y(s)}{y_{sp}(s)} = \frac{1}{(\lambda s+1)^n}$ is used. So, $q(s) = \frac{1}{g(s)} \frac{1}{(\lambda s+1)^n} = \frac{f(s)}{g(s)}$

In actual practice, $g(s)$ is estimated as $g_m(s)$,

So, open loop control law should be $q(s) = f(s)/g_m(s)$

Open loop control

f(s) is called filter and should be chosen such a way that

- *q(s) should not become improper.*
- *Process dynamic characteristics are not violated.*

Since, inverse of $g_m(s)$ is required for the open loop control law, it should also be noted that it may not be always possible to get inverse of $g_m(s)$.

So, factorize $g_m(s) = g_m^-(s)\, g_m^+(s)$ where

$g_m^-(s)$ is invertible part and $g_m^+(s)$ is non-invertible part.

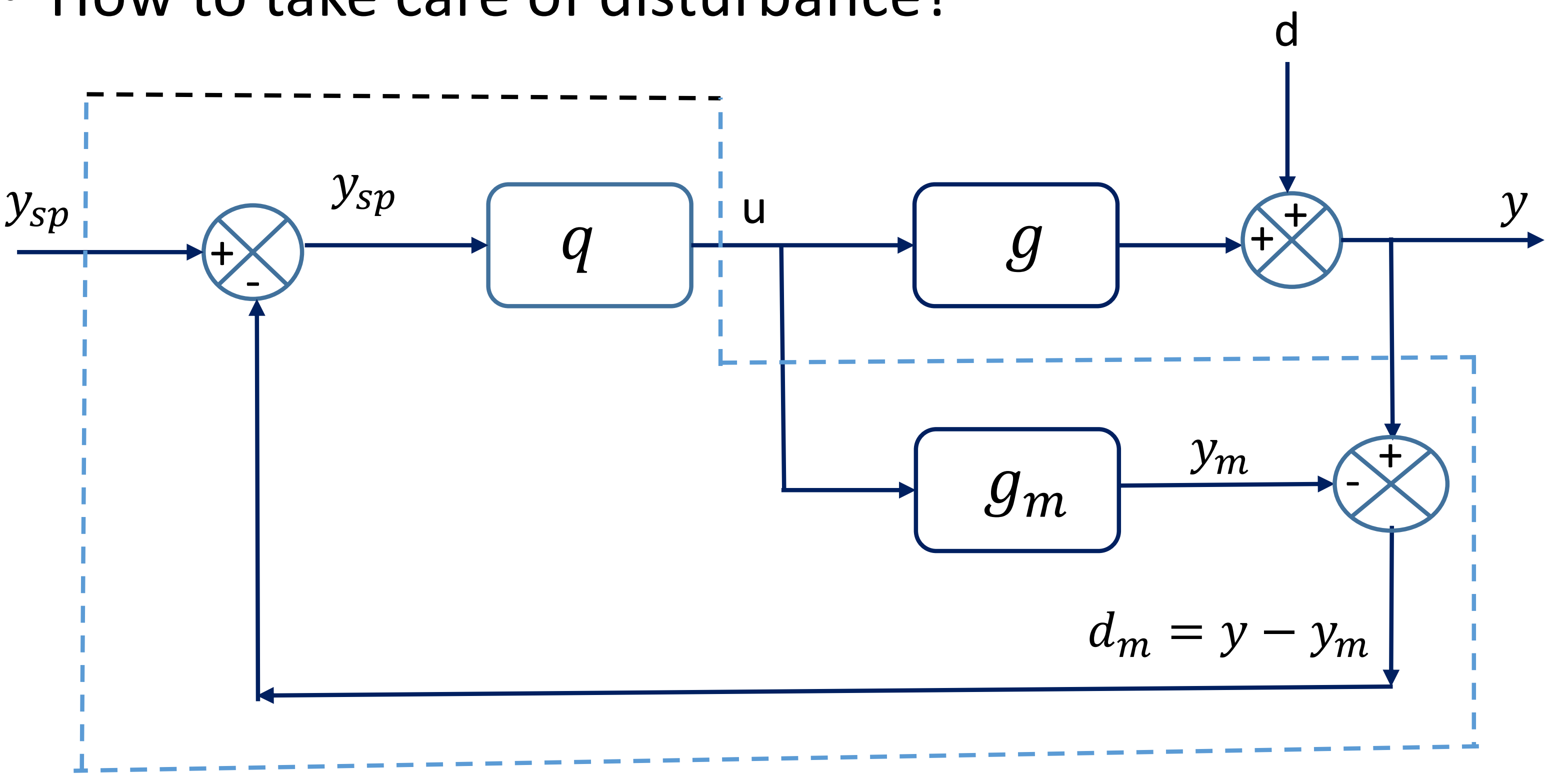
So, finally, open loop control law is $q(s) = \dfrac{f(s)}{g_m^-(s)}$

Open loop control is always stable if the process is stable.

Disturbance to the system → How to handle?

Internal Model Control

- Open loop control law discussed earlier can not take care of disturbances to the process.
- How to take care of disturbance?



- Disturbance, d is estimated internally using a process model
- The above is called Internal Model Control structure.

Internal Model Control

- IMC Equation:
- $y(s) = \frac{g(s)q(s)}{1+q(s)\left(g(s)-g_m(s)\right)}\ y_{sp}(s)$

$$+ \ \left[\frac{1 - g_m(s)q(s)}{1 + q(s)\left(g(s) - g_m(s)\right)}\right] d(s)$$

- The ideal IMC control law $q(s) = \frac{1}{g_m(s)}$ ensures both servo and regulatory control in spite of process/model mismatch $(g \neq g_m)$ and presence of disturbance $(d(s) \neq 0)$.
- In actual practice, design of IMC control law is same as earlier used for open loop control.

IMC Design Procedure

- Factor the process model into invertible and non-invertible component

$$g_m(s) = g_m^-(s)\, g_m^+(s)$$

- Form the idealized IMC controller which is the inverse of the invertible component of the process model.

$$q_I(s) = \frac{1}{g_m^-(s)}$$

- Add a filter $f(s) = \frac{1}{(\lambda s+1)^n}$ to make the controller atleast semi-proper

$$q(s) = q_I(s)\, f(s) = \frac{f(s)}{g_m^-(s)}$$

- Adjust λ to vary the speed of response of the control system.

Example-1: FODT Process

Consider the process $g(s) = \frac{1}{10s+1} e^{-3s}$

Assuming no process/model mismatch, $g_m(s) = g(s)$

Using the IMC design steps:

1. Factorize g_m to invertible and non-invertible parts.

$$g_m^-(s) = \frac{1}{10s+1} \text{ and } g_m^+(s) = e^{-3s}$$

2. Idealized IMC : $\quad q_I(s) = 10s + 1$

3. IMC: $\quad q(s) = q_I(s)f(s) = \frac{10s+1}{\lambda s+1}$

So, the controlled response will be

$$y(s) = g_m^+(s)f(s)y_{sp}(s) = \frac{e^{-3s}}{\lambda s+1} y_{sp}(s)$$

Example-2: SO with RHP zero

Consider the process $g(s) = \frac{-9s+1}{(15s+1)(3s+1)}$

Assuming no process/model mismatch, $g_m(s) = g(s)$

Using the IMC design steps:

1. Factorize g_m to invertible and non-invertible parts.

$g_m^-(s) = \frac{9s+1}{(15s+1)(3s+1)}$ and $g_m^+(s) = \frac{-9s+1}{9s+1}$

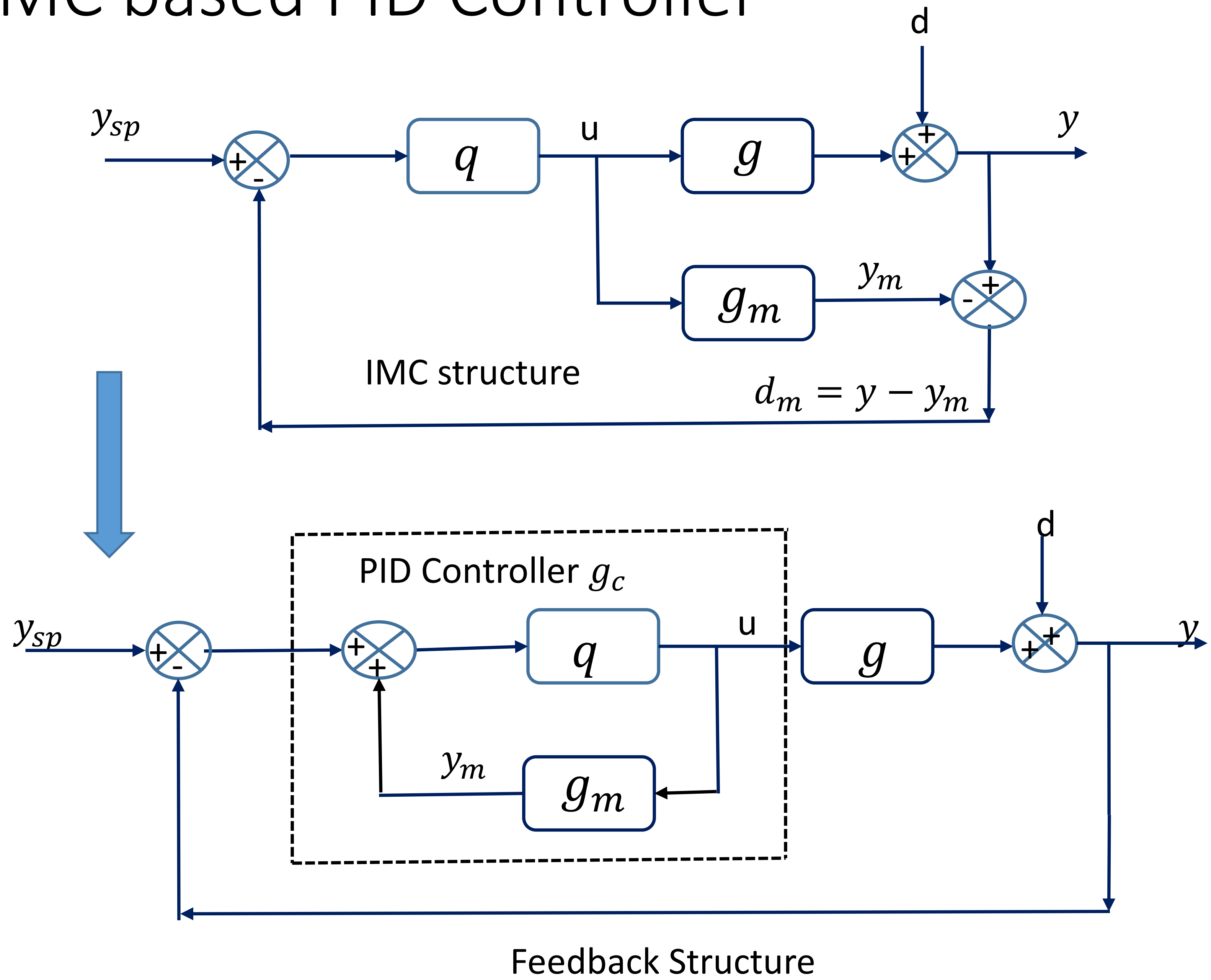
2. Idealized IMC: $q_I(s) = \frac{(15s+1)(3s+1)}{9s+1}$

3. IMC: $q(s) = q_I(s)f(s) = \frac{(15s+1)(3s+1)}{9s+1}\frac{1}{\lambda s+1}$

So, the controlled response will be

$$y(s) = g_m^+(s)f(s)y_{sp}(s) = \frac{-9s+1}{(9s+1)(\lambda s+1)}y_{sp}(s)$$

IMC based PID Controller



IMC based PID Controller design

- From the IMC and its equivalent Feedback block diagram, we can relate $g_c(s) = \frac{q(s)}{1-q(s)g_m(s)}$
- Where, $g_c(s)$ is a PID controller.
- **For a first order process** $g(s) = \frac{K}{\tau s+1}$
- IMC controller : $q(s) = \frac{\tau s+1}{K(\lambda s+1)}$
- Equivalent PID Controller

$$g_c(s) = \frac{q(s)}{1 - q(s)g_m(s)} = \frac{\dfrac{\tau s + 1}{K(\lambda s + 1)}}{1 - \dfrac{K}{\tau s + 1}\dfrac{\tau s + 1}{K(\lambda s + 1)}} = \frac{\tau s + 1}{K\lambda s}$$

So, $g_c(s) = \frac{\tau}{K\lambda}(1 + \frac{1}{\tau s})$ is PI controller with $K_c = \frac{\tau}{K\lambda}$ and $\tau_I = \tau$

IMC based PID Controller Design

Second Order Process $g(s) = \frac{K}{(\tau_1 s+1)(\tau_2 s+1)}$

- IMC controller: $q(s) = \frac{(\tau_1 s+1)(\tau_2 s+1)}{K(\lambda s+1)}$ Here, we allow improper q(s) to get ideal PID controller.
- Equivalent PID controller:
- $g_c(s) = \frac{q(s)}{1-g_m(s)q(s)} = \frac{\frac{(\tau_1 s+1)(\tau_2 s+1)}{K(\lambda s+1)}}{1-\frac{K}{(\tau_1 s+1)(\tau_2 s+1)}\frac{(\tau_1 s+1)(\tau_2 s+1)}{K(\lambda s+1)}}$
- or,

$$g_c(s) = \frac{\tau_1\tau_2 s^2 + (\tau_1 + \tau_2)s + 1}{K\lambda s} = \frac{\tau_1 + \tau_2}{K\lambda}\left(1 + \frac{1}{(\tau_1 + \tau_2)s} + \frac{\tau_1\tau_2}{\tau_1 + \tau_2}s\right)$$

IMC based PID Controller Design

FODT Process: $g(s) = \frac{K}{\tau s+1} e^{-\theta s}$

IMC controller: $q(s) = \frac{\tau s+1}{K(\lambda s+1)}$

Equivalent PID: $g_c(s) = \frac{\frac{\tau s+1}{K(\lambda s+1)}}{1-\frac{K}{\tau s+1}e^{-\theta s}\frac{\tau s+1}{K(\lambda s+1)}} = \frac{\tau s+1}{K(\lambda s+1-e^{-\theta s})}$

$e^{-\theta s}$ can be approximated by either first or 2nd order Taylor series.

For 1st order: $e^{-\theta s} = 1 - \theta s$

so, $g_c(s) = \frac{\tau s+1}{K(\lambda+\theta)s} = \frac{\tau}{K(\theta+\lambda)}(1 + \frac{1}{\tau s})$

IMC based PID Controller Design

- For 2nd order approximation: $e^{-\theta s} = \frac{1-\frac{\theta}{2}s}{1+\frac{\theta}{2}s}$

$$g(s) = \frac{K(1-0.5\theta s)}{(\tau s+1)(1+0.5\theta s)}$$

IMC controller: $q(s) = \frac{(\tau s+1)(1+0.5\theta s)}{K(\lambda s+1)}$

Equivalent PID controller: $g_c(s) = \frac{q(s)}{1-g_m(s)q(s)}$

i.e, $g_c(s) = \frac{(\tau s+1)(1+0.5\theta s)}{K(\lambda+0.5\theta)s}$

$$= \frac{\tau+0.5\theta}{K(\lambda+0.5\theta)}\left(1 + \frac{1}{(\tau+0.5\theta)s} + \frac{\tau\theta}{2\tau+\theta}s\right)$$

IMC based PID Controller Design

Controller : $g_c(s) = K_c(1 + \frac{1}{\tau_I s} + \tau_D s)(\frac{1}{\tau F\ s+1})$

Process	Filter	K_c	τ_I	τ_D	τ_F
$\frac{k_p}{\tau_p s+1}$	$\frac{1}{\lambda s+1}$	$\frac{\tau_p}{k_p \lambda}$	τ_p		
$\frac{k_p}{\tau_p s+1}$	$\frac{\gamma s+1}{(\lambda s+1)^2}$	$\frac{2\tau_p - \lambda}{k_p \lambda}$	$\frac{2\tau_p \lambda - \lambda^2}{\tau_p}$		
$\frac{k_p}{(\tau_1 s+1)(\tau_2 s+1)}$	$\frac{1}{\lambda s+1}$	$\frac{\tau_1 + \tau_2}{k_p \lambda}$	$\tau_1 + \tau_2$	$\frac{\tau_1 \tau_2}{\tau_1 + \tau_2}$	
$\frac{k_p}{\tau^2 s^2 + 2\zeta\tau s+1}$	$\frac{1}{\lambda s+1}$	$\frac{2\zeta\tau}{k_p \lambda}$	$2\zeta\tau$	$\frac{\tau}{2\zeta}$	
$\frac{k_p}{\tau^2 s^2 + 2\zeta\tau s+1}$	$\frac{1}{(\lambda s+1)^2}$	$\frac{\zeta\tau}{k_p \lambda}$	$2\zeta\tau$	$\frac{\tau}{2\zeta}$	$\frac{\lambda}{2}$

IMC based PID Controller Design

Controller : $g_c(s) = K_c(1 + \frac{1}{\tau_I s} + \tau_D s)(\frac{1}{\tau F\ s+1})$

Process	Filter	K_c	τ_I	τ_D	τ_F
$\frac{k_p(-\beta s+1)}{\tau^2 s^2 + 2\zeta\tau s + 1}$	$\frac{-\beta s+1}{(\beta s+1)(\lambda s+1)}$	$\frac{2\zeta\tau}{k_p(2\beta+\lambda)}$	$2\zeta\tau$	$\frac{\tau}{2\zeta}$	$\frac{\beta\lambda}{2\beta+\lambda}$
$\frac{k_p(-\beta s+1)}{\tau^2 s^2 + 2\zeta\tau s + 1}$	$\frac{-\beta s+1}{\lambda s+1}$	$\frac{2\zeta\tau}{k_p(\beta+\lambda)}$	$2\zeta\tau$	$\frac{\tau}{2\zeta}$	
$\frac{k}{s}$	$\frac{1}{\lambda s+1}$	$\frac{1}{k\lambda}$			
$\frac{k}{s}$	$\frac{2\lambda s+1}{(\lambda s+1)^2}$	$\frac{2}{k\lambda}$	2λ		
$\frac{k}{s(\tau s+1)}$	$\frac{1}{\lambda s+1}$	$\frac{1}{k\lambda}$		τ	
$\frac{k}{s(\tau s+1)}$	$\frac{2\lambda s+1}{(\lambda s+1)^2}$	$\frac{2\lambda+\tau}{k\lambda^2}$	$2\lambda + \tau$	$\frac{2\lambda\tau}{2\lambda+\tau}$	

Controller : $g_c(s) = K_c(1 + \frac{1}{\tau_I s} + \tau_D s)(\frac{1}{\tau F\ s+1})$

Process	K_c	τ_I	τ_D	τ_F
$\frac{k_p e^{-\theta s}}{\tau_p s+1}$	$\frac{\tau_p + \frac{\theta}{2}}{k_p(\theta+\lambda)}$	$\tau_p + \frac{\theta}{2}$	$\frac{\tau_p\theta}{2\tau_p+\theta}$	$\frac{\lambda\theta}{2(\lambda+\theta)}$
$\frac{k_p e^{-\theta s}}{\tau_p s+1}$	$\frac{\tau_p + \frac{\theta}{2}}{k_p\left(\lambda+\frac{\theta}{2}\right)}$	$\tau_p + \frac{\theta}{2}$	$\frac{\tau_p\theta}{2\tau_p+\theta}$	
$\frac{ke^{-\theta s}}{s}$	$\frac{2\lambda+\theta}{k(\lambda+\theta)^2}$	$2\lambda + \theta$		
$\frac{ke^{-\theta s}}{s}$	$\frac{2}{k\left(\lambda+\frac{\theta}{2}\right)}$	$2\lambda + \theta$	$\frac{\lambda\theta+\frac{\theta^2}{4}}{2\lambda+\theta}$	

IMC based PID for unstable process

- Find the IMC controller transfer function, $q(s)$, which includes a filter, $f(s)$, to make $q(s)$ semiproper. An additional requirement is that the value of $f(s)$ at $s = p_u$ (where p_u is an unstable pole) must be 1.
- Morari and Zafiriou (1989) recommend a filter transfer function that has the form $f(s) = \frac{(\gamma s+1)}{(\lambda s+1)^n}$
- A value of γ is found that satisfies the filter requirement $f(s = p_u) = 1$.
- Find the equivalent standard feedback controller using the transformation $g_c(s) = \frac{q(s)}{1-g_m(s)q(s)}$

Example

- Let us consider an unstable process $g(s) = \frac{K}{-\tau s+1}$
- IMC controller: $q(s) = \frac{-\tau s+1}{K} \frac{(\gamma s+1)}{(\lambda s+1)^2}$
- Solve for γ so that $f\left(s = \frac{1}{\tau}\right) = \frac{(\gamma/\tau+1)}{(\lambda/\tau+1)^2} = 1$
- This gives, $\gamma = \lambda\left(\frac{\lambda}{\tau} + 2\right)$
- $g_c(s) = \frac{q(s)}{1-g_m(s)q(s)} = \frac{\gamma}{K(2\lambda-\gamma)} \frac{(\gamma s+1)}{\gamma s}$
- $K_c = \frac{\gamma}{K(2\lambda-\gamma)} = -\frac{\lambda+2\tau}{K\lambda}$
- $\tau_I = \gamma = \lambda\left(\frac{\lambda}{\tau} + 2\right)$

Selection of λ

- The choice of design parameter λ is a key decision in the IMC design methods.

- In general, increasing λ produces a more conservative controller because K_c decreases while λ increases.

- Several IMC guidelines for λ have been published for the FODT model
  - $\frac{\lambda}{\theta} > 0.8$ and $\lambda > 0.1\,\tau$ - Rivera et. Al. (1986)
  - $\tau > \lambda > \theta$ - Chien and Fruehauf, (1990)
  - $\lambda = \theta$ - Skogestad, (2003)

Tuning for Lag-Dominant Models

- First- or second-order models with relatively small time delays $(\theta/\tau \ll 1)$ are referred to as *lag-dominant models*.

- The IMC and DS methods provide satisfactory set-point responses, but very slow disturbance responses, because the value of τ_I is very large.

- Fortunately, this problem can be solved in three different ways.

Method 1: Integrator Approximation

Approximate $\tilde{G}(s) = \dfrac{Ke^{-\theta s}}{\tau s + 1}$ by $\tilde{G}(s) = \dfrac{K * e^{-\theta s}}{s}$

where $K * \triangleq K/\tau$.

- Then use the IMC tuning rules to specify the controller settings.

Method 2. Limit the Value of τ_I

- For lag-dominant models, the standard IMC controllers for first-order and second-order models provide sluggish disturbance responses because τ_I is very large.

- For example, controller G_c has $\tau_I \geq \tau$ for most of the cases where τ is very large.

- As a remedy, Skogestad (2003) has proposed limiting the value of τ_I:

$$\tau_I = \min\left\{\tau_1, 4\left(\lambda + \theta\right)\right\}$$

where τ_1 is the largest time constant (if there are two).

Method 3. Design the Controller for Disturbances, Rather Set-point Changes

- The desired CLTF is expressed in terms of $(Y/D)_{\text{des}}$, rather than $(Y/Y_{sp})_{\text{des}}$

- *Reference*: Chen & Seborg (2002)

Example

Consider a lag-dominant model with $\theta / \tau = 0.01$:

$$\tilde{G}(s) = \frac{100}{100s+1} e^{-s}$$

Design four PI controllers:

a) IMC $(\tau_c = 1)$

b) IMC $(\tau_c = 2)$ based on the integrator approximation

c) IMC $(\tau_c = 1)$ with Skogestad's modification

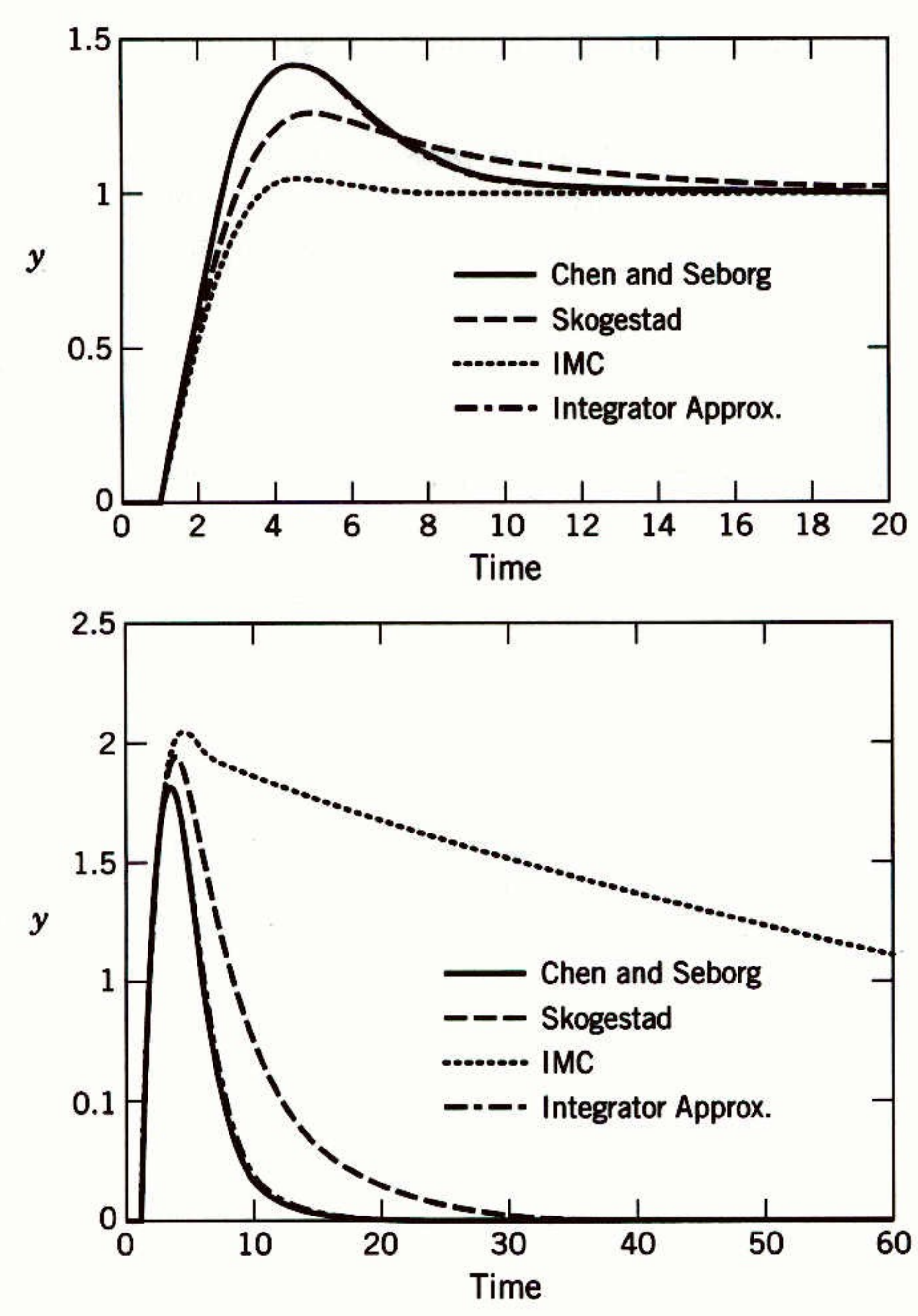
d) Direct Synthesis method for disturbance rejection (Chen and Seborg, 2002): The controller settings are $K_c = 0.551$ and $\tau_I = 4.91$.

Evaluate the four controllers by comparing their performance for unit step changes in both set point and disturbance. Assume that the model is perfect and that $G_d(s) = G(s)$.

Solution

The PI controller settings are:

Controller	K_c	τ_I
(a) IMC	0.5	100
(b) Integrator approximation	0.556	5
(c) Skogestad	0.5	8
(d) DS-d	0.551	4.91



Comparison of set-point responses (top) and disturbance responses (bottom). The responses for the Chen and Seborg and integrator approximation methods are essentially identical.

Time Delay Processes

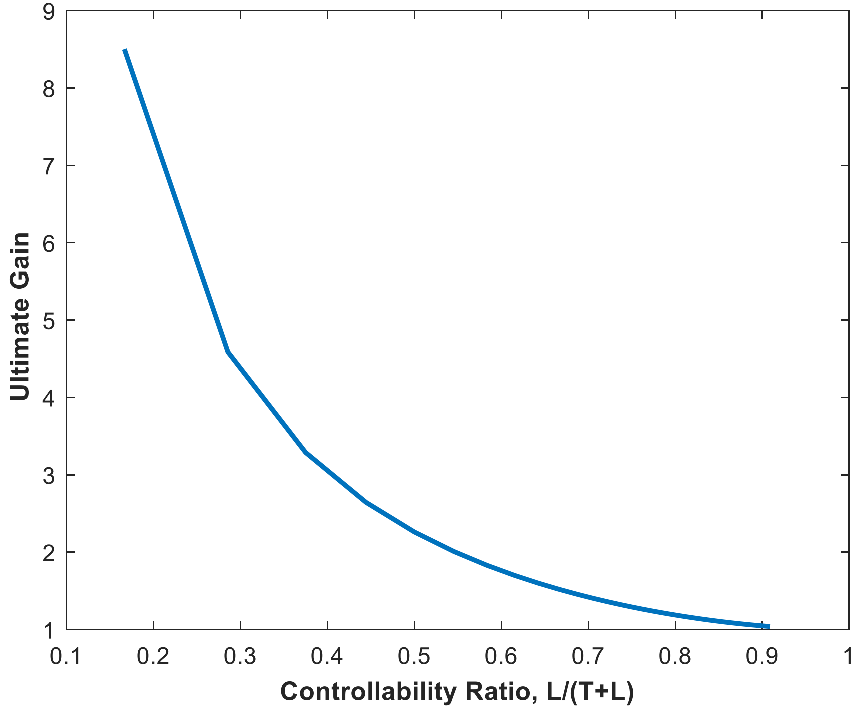
- Dead times, or time delays, are found in many processes in industry.
- Dead times are mainly caused by the
  - accumulation of time lags in a number of dynamic systems connected in series (e.g. distillation column)
  - Transportation lag (long pipelines)
  - Slow measuring device: GC
  - Sampling delays introduced by computer control
- For processes exhibiting dead time, every action executed in the manipulated variable of the process will only affect the controlled variable after the process dead time.

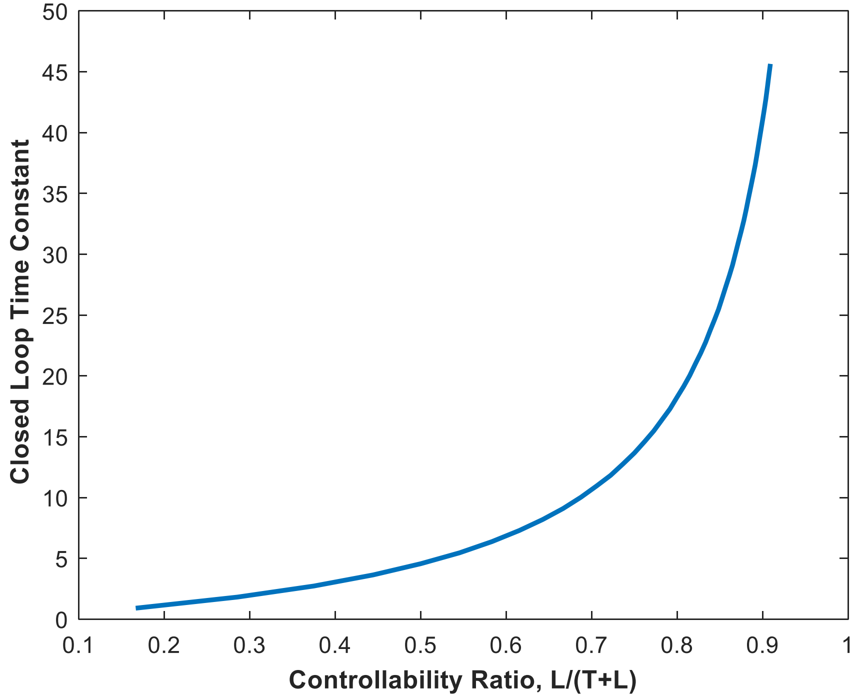
Effect of Time Delay

- Processes with large dead time are difficult to control by pure feedback control (PID) alone because
  - Effect of disturbances is not seen by controller for a while
  - Effect of control action is not seen at the output for a while. This causes controller to take additional compensation unnecessary.
  - This results in unnecessary oscillation and loss of feedback loop controllability.
- Dead times produce a decrease in the system phase and also give rise to a non-rational transfer function of the system, making them more difficult to analyse and control.

Controllability Ratio

- Controllability Ratio (CR) is defined as $\frac{\theta}{\tau+\theta}$ where θ is deadtime or timedelay and τ is dominant time constant of the process.

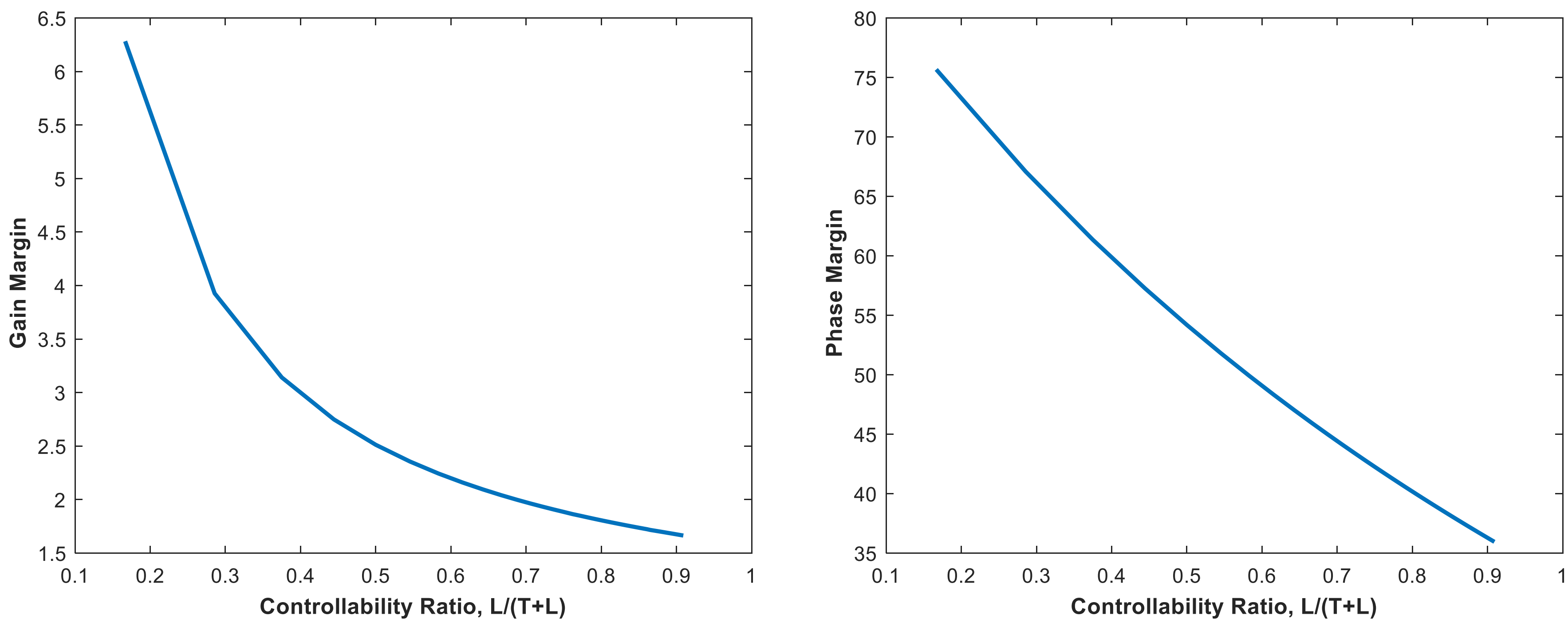




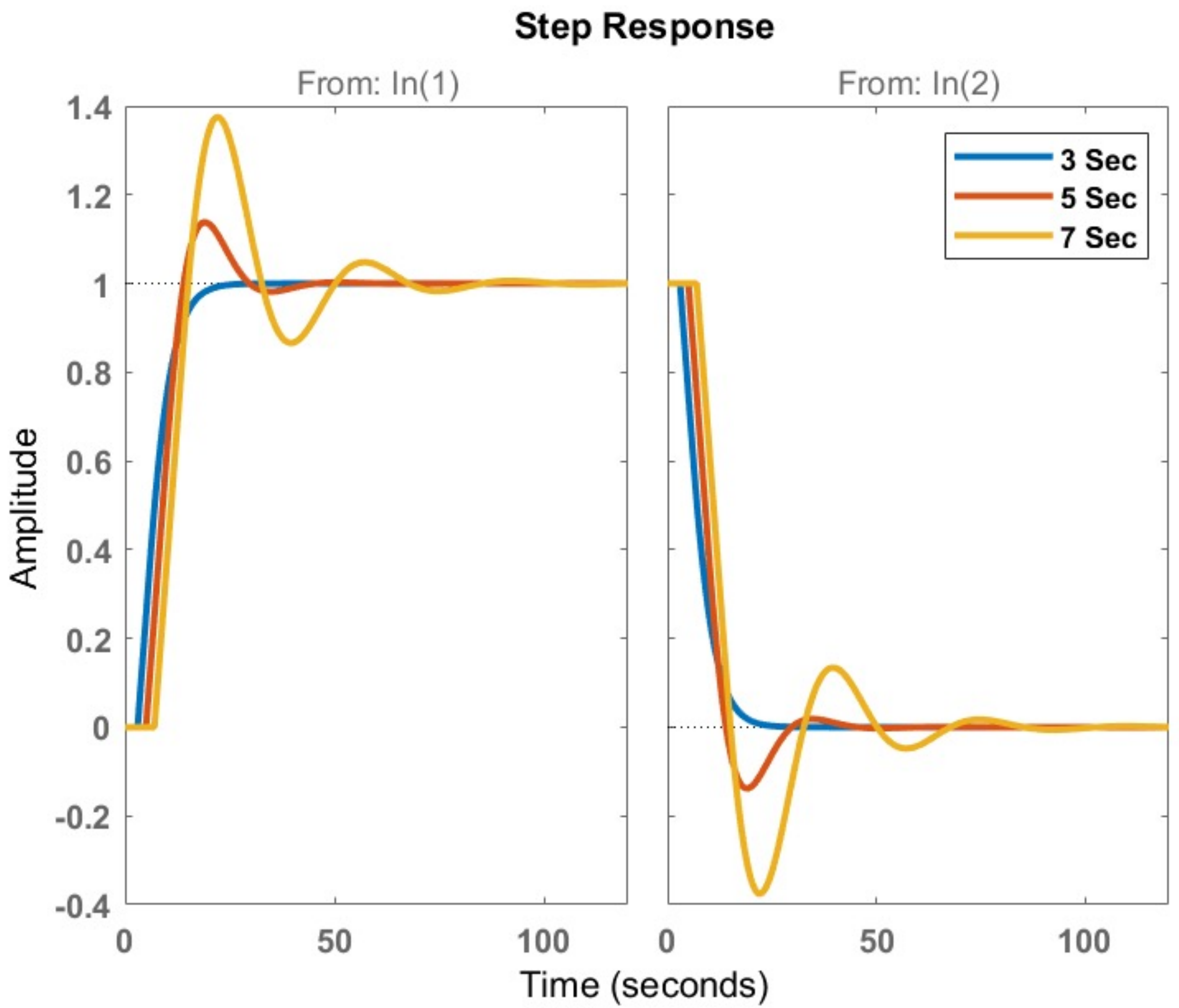
CR =1 represents pure delay processes. As CR increases, the stability of the control loop decreases. PID controller has to be tuned with lower gain (Z-N tuning) resulting sluggish response.

Effect of Controllability Ratio

- Robustness of the controller decreases.
- Both Gain Margin and Phase Margin reduced

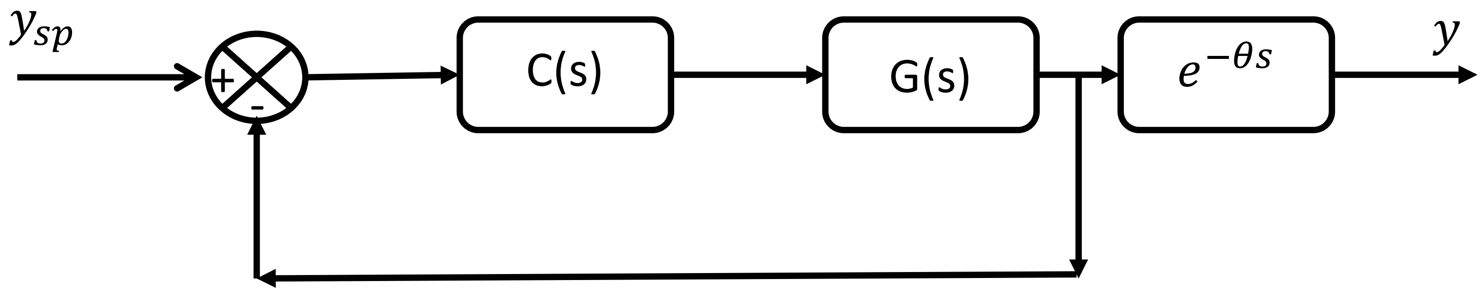


Effect of Deadtime on output response



Predictor based solution

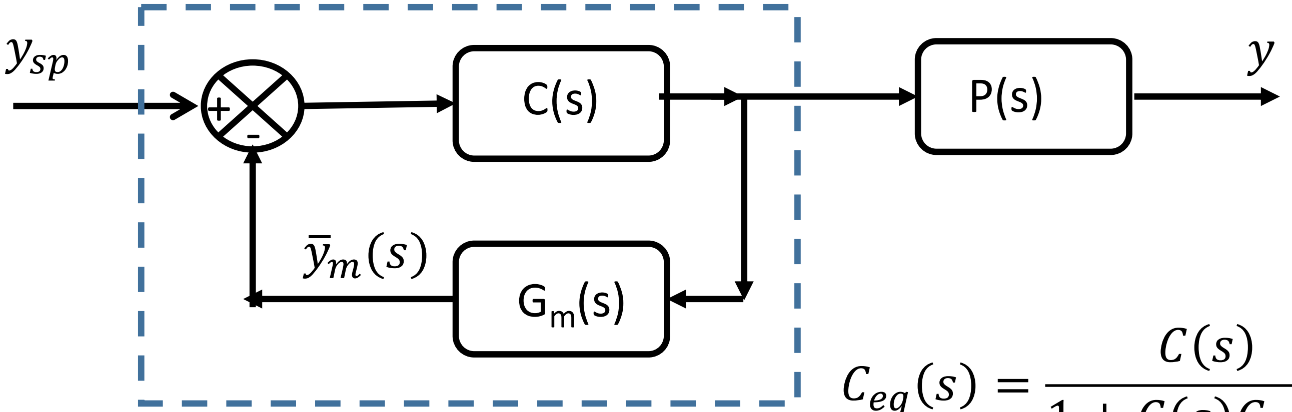
Ideal solution is to send the undelayed output signal to the controller as shown in the figure.



The real implementation of this solution is not possible in practice mainly because the sensor cannot be installed in the desired position and/or the process dead time is not caused by mass transportation.

A simple solution can be the use of process model without deadtime element. If process P(s) = G(s)$e^{-\theta s}$ is approximated by $G_m(s)e^{-\theta_m s}$ and $G_m(s)$ is used to send output signal.

Predictor based solution



$$C_{eq}(s) = \frac{C(s)}{1 + C(s)G_m(s)}$$

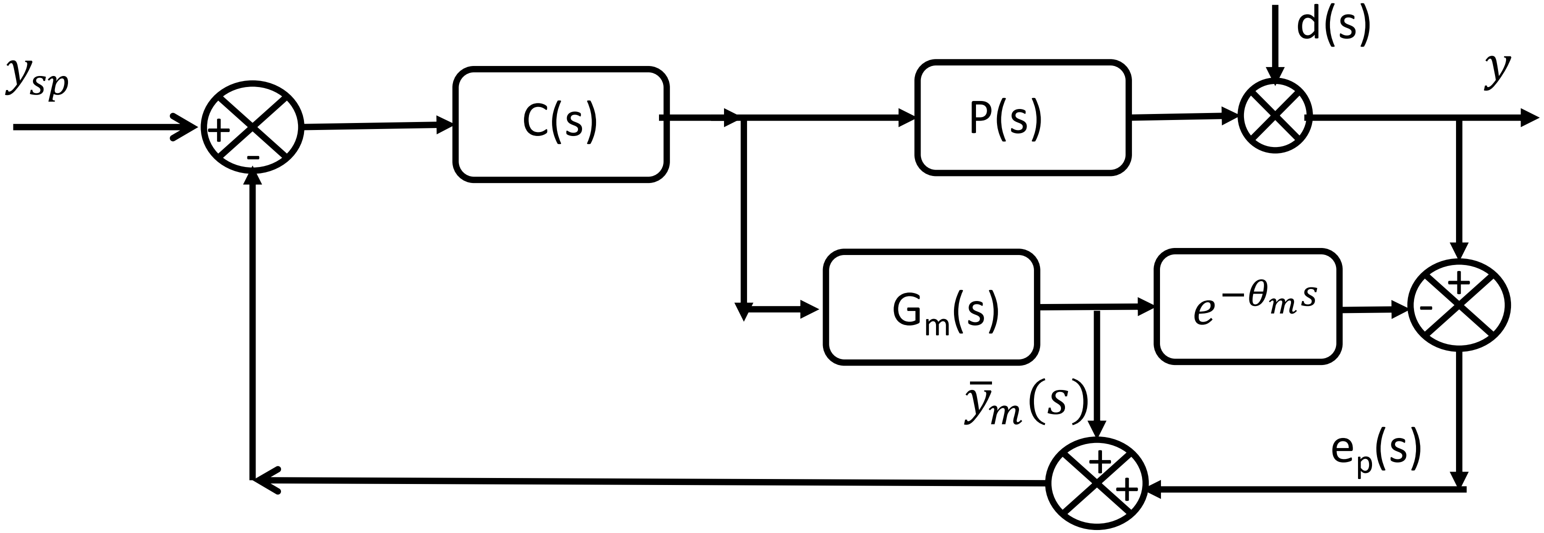
This is an open-loop predictor control system.

Control loop equation :

$$y(s) = \frac{C(s)P(s)}{1 + C(s)G_m(s)} y_{sp}(s) = \frac{C(s)G(s)e^{-\theta s}}{1 + C(s)G_m(s)} y_{sp}(s)$$

Limitation: Effect of disturbance and Process/Model mismatch is not taken care of in this design

Smith Predictor

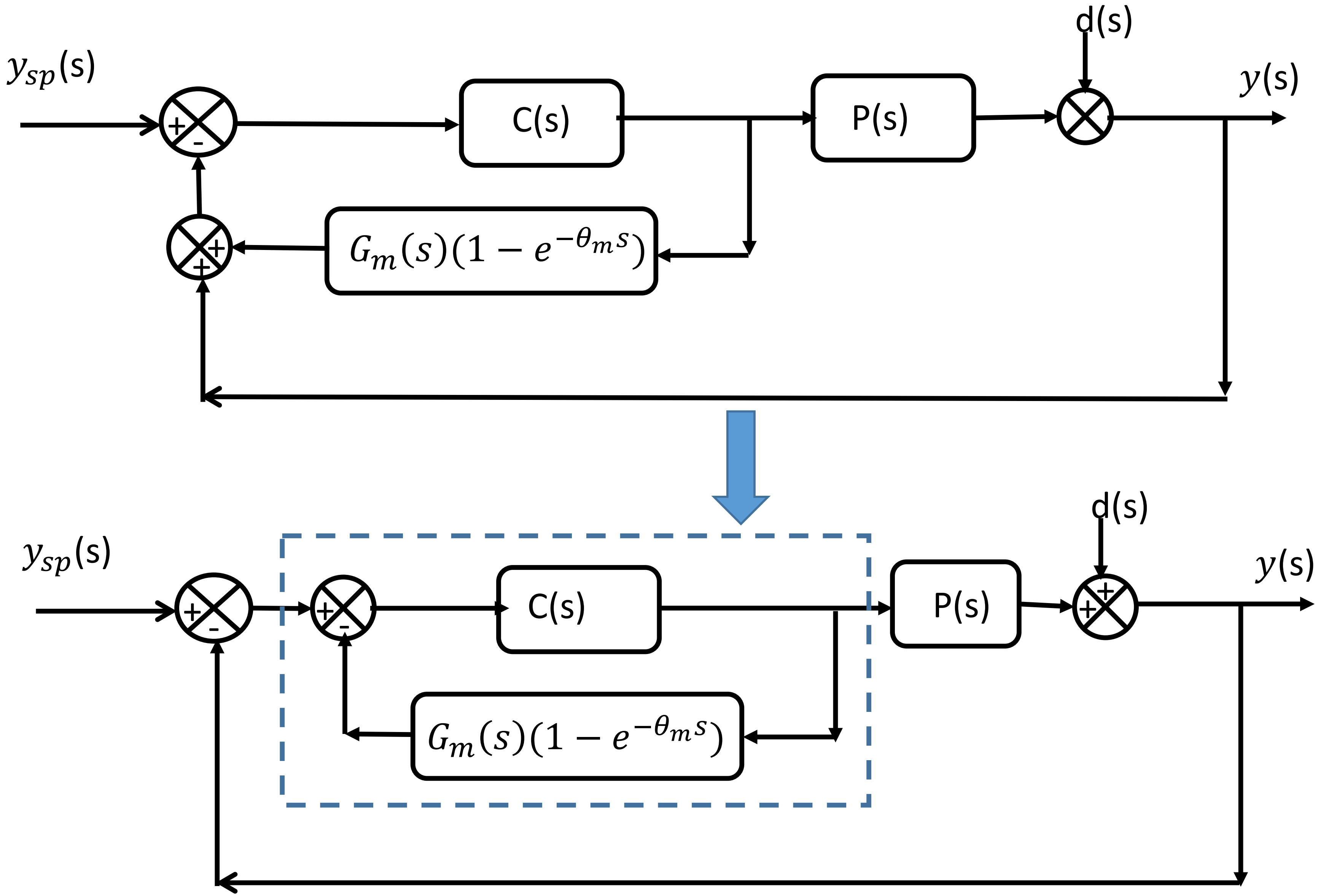


With this structure, if there are no modelling errors or disturbances, the error between the current process output and the model output ($e_p(t)$) will be null and the controller can be tuned as if the plant had no dead time.

Thus, in the nominal case this structure gives the same performance as the ideal solution i.e, open loop predictor.

This structure was developed by Smith (1950) and is called Smith Predictor.

Smith Predictor : Feedback Structure



Characteristic Equation

- Smith Compensator: $C_s(s) = \frac{C(s)}{1+C(s)g_{m(s)}\left(1-e^{-\theta_m s}\right)}$
- Feedback Equation:
- $y(s) = \frac{C_s(s)P(s)}{1+C_s(s)P(s)} y_{sp}(s) + \frac{1}{1+C_s(s)P(s)} d(s)$
- Characteristic Equation:
- $1 + C_s(s)P(s) = 1 + \frac{C(s)}{1+C(s)g_{m(s)}\left(1-e^{-\theta_m s}\right)}\ g(s)e^{-\theta s} = 0$
- $1 + C(s)g_m(s) - C(s)g_m(s)e^{-\theta_m s} + C(s)g(s)e^{-\theta s} = 0$
- No mismatch in Process/Model: $1 + C(s)g(s) = 0$

Smith compensator in IMC-PID

- Consider, FODT process model : $\frac{K_m}{\tau_m s+1} e^{-\theta_m s}$
- IMC- Controller : $\mathrm{C}(s) = \frac{\tau_m s+1}{K_m(\lambda s+1)}$
- IMC-PID Controller: $g_c(s) = \frac{\tau_m s+1}{K_m(\lambda s+1-e^{-\theta_m s})} = \frac{u(s)}{e(s)}$
- So, $K_m\left(\lambda s + 1 - e^{-\theta_m s}\right)u(s) = (\tau_m s + 1)e(s)$
- Or, $\frac{K_m\,\lambda s}{\tau_m s+1} u(s) + \frac{K_m}{\tau_m s+1} u(s) - \frac{K_m e^{-\theta_m s}}{\tau_m s+1} u(s) = e(s)$

Or, $\frac{K_m\,\lambda s}{\tau_m s+1} u(s) + g_m(s)\,u(s) - g_m(s) e^{-\theta_m s} u(s) = e(s)$

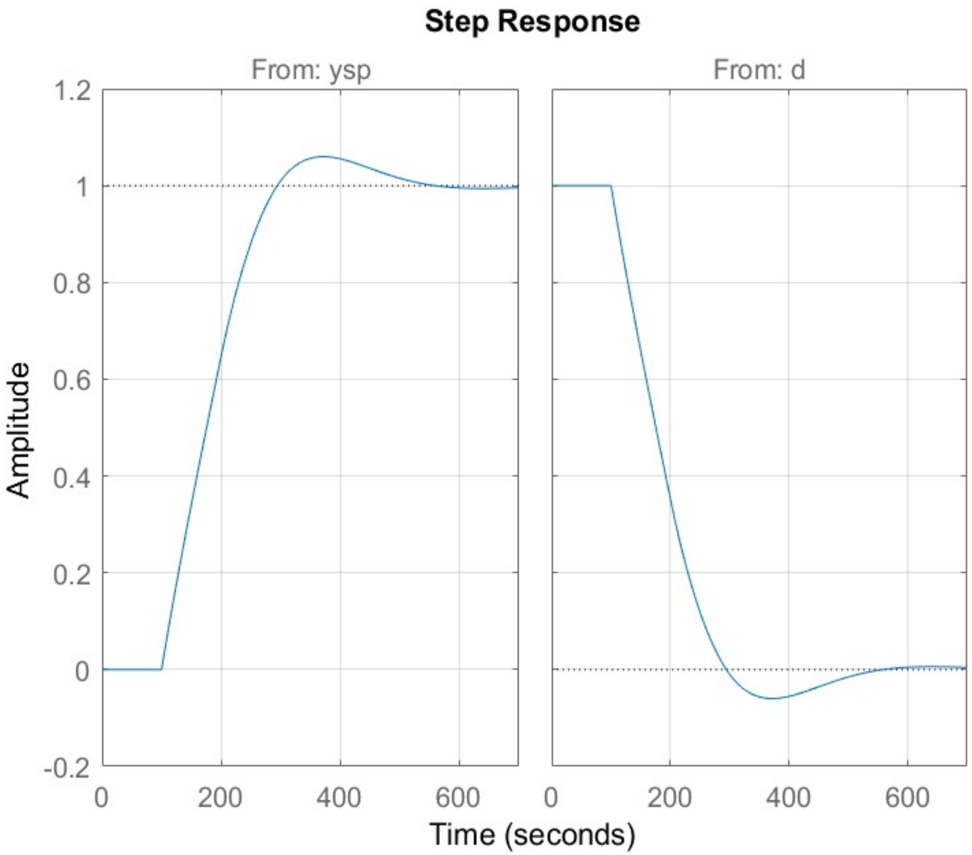
Or, $\boxed{u(s) = \frac{\tau_m s+1}{K_m \lambda s}\left[e(s) - g_m(s)\left(1 - e^{-\theta_m s}\right)u(s)\right]}$

Matlab Implementation

Process: $P(s) = \frac{6}{43s+1} e^{-100s}$ and $g(s) = P(s)$

Feedback control of Process:

```
s=tf('s');
P=6/(43*s+1)*exp(-100*s);
P.InputName='u';
P.OutputName='y0';
[C,info]=pidtune(P,pidstd(1,1));
Info
Pcl=feedback([P*C,1],1,1,1);
Pcl.InputName={'ysp', 'd'};
step(Pcl), grid on;
```



Bandwidth : 0.0059 rad/s

Phase Margin : 60 degree

Matlab Implementation

```
% Prediction model
Gp = 6/(45*s+1); Gp.InputName = 'u'; Gp.OutputName = 'yp';
 Dp = exp(-100*s); Dp.InputName = 'yp'; Dp.OutputName = 'y1';
 % Develop Process with Smith Predictor
 S1 = sumblk('dy = y0 - y1'); S2 = sumblk('ym = yp + dy');
Plant = connect(P,Gp,Dp,S1,S2,'u','ym');
 % Design PI controller with 60 degrees phase margin
[C,info] = pidtune(Plant,pidstd(1,1)); C.InputName = 'e';C.OutputName = 'u';
% Assemble closed-loop model from [y_sp,d] to y
Sum1 = sumblk('e = ysp - yp - dy'); Sum2 = sumblk('y = y0 + d'); Sum3 = sumblk('dy = y - y1');
T = connect(P,Gp,Dp,C,Sum1,Sum2,Sum3,{'ysp','d'},'y');
% Design PI controller with 85 degrees phase margin
Opt = pidtuneOptions('PhaseMargin',85);
[C1,info] = pidtune(Plant,pidstd(1,1),Opt);
C1.InputName = 'e'; C1.OutputName = 'u';
T1 = connect(P,Gp,Dp,C1,Sum1,Sum2,Sum3,{'ysp','d'},'y');
% Use STEP to compare the Smith Predictor (blue) with the PI controller (red):
step(T1,'b',T,'b--',gcl,'r--'), grid on
```

Matlab Implementation