Riassunti

▼ **Solution of LTI continuous time systems computation**

Manual procedure:

1. Compute $X(s)$, $Y(s)$ in the Laplace domain
2. Derive the Heaviside's partial fraction expansion (PFE) of $X(s)$ and $Y(s)$
3. Compute the residues of the PFE
4. Obtain $x(t)$ and $y(t)$ via Laplace inverse transform of $X(s)$ and $Y(s)$

MatLab procedure:

▼ Case 1: **real poles and zeros**

```
s = tf('s');

% Define the system input and initial condition
U = 2/s;
x0 = [2;2];

% Define the system matrices
A = [0 1; -2 -3];
B = [1; 0];

% Compute X(s) = (sI-A)^-1 * [x(0) + BU(s)]
X = zpk(minreal(inv(s * eye(2) - A) * (B * U + x0), 1e-2))

% For each of the 2 components of X(s), compute the PFE
[num_x1, den_x1] = tfdata(X(1), 'v')
```

```
[r1, p1] = residue(num_x1, den_x1)

[num_x2, den_x2] = tfdata(X(2), 'v')
[r2, p2] = residue(num_x2, den_x2)
```

Finally compute the inverse Laplace transform in the form:

$x_1(t) = r_1 e^{p_1 t} + r_2 e^{p_2 t} + ...$

▼ Case 2: **a couple of complex conjugates poles**

```
s = tf('s');

% Define the system input and initial condition
U = 1/s;
x0 = [1;1];

% Define the system matrices
A = [-3 2; -2 -3];
B = [1; 0];
C = [0 1];

% Compute X(s) = C * (sI-A)^-1 * [x(0) + BU(s)]
Y = zpk(minreal(C * inv(s * eye(2) - A) * (B * U + x0), 1e-2))

% For each of the 2 components of X(s), compute the PFE
[num_y, den_y] = tfdata(Y, 'v')
[r, p] = residue(num_y, den_y)

% Compute magnitude and phase of the residue corresponding to the complex root with pos
itive imaginary part
M = abs(r(1))
phi = angle(r(1))
```

Finally compute $y(t)$ in the form:

$y(t) = [2 * M e^{\mathrm{Re}[p_1]t} \cos(\mathrm{Im}[p_1]t + \phi)]\epsilon(t)$

▼ **Internal stability of LTI systems**

An LTI system is:

- **Internally stable** if the zero input state response $x_{zi}(t)$ is bounded for any initial state x_0
  - $\mathrm{Re}[\lambda_i(A)] \leq 0$, for $i = 1, ..., n$ and $\mu'(\lambda_i(A)) = 1$ for all the eigenvalues such that $\mathrm{Re}[\lambda_j(A)] = 0$

- **Asymptotically stable** if the zero input state response $x_{zi}(t)$ converges to 0 as $t \to \infty$, for any initial state x_0
  - $\text{Re}[\lambda_i(A)] < 0$ for $i = 1, ..., n$
- **Unstable** if it's not stable
  - $\exists i : \text{Re}[\lambda_i(A)] > 0$ or $\text{Re}[\lambda_i(A)] \leq 0$ for $i = 1, ..., n$ and $\mu'(\lambda_i(A)) > 1$ for the eigenvalues such that $\text{Re}[\lambda_j(A)] = 0$

▼ Natural modes classification

A natural mode associated with the eigenvalue λ is said:

- **Convergent** if $\lim_{t \to \infty} |m(t)| = 0$
  - $\text{Re}[\lambda_i(A)] < 0$
- **Bounded** if $\exists M \in R : \forall t \geq 0, 0 \leq |m(t)| \leq M < \infty$
  - $\text{Re}[\lambda_i(A)] = 0$ and $\mu'(\lambda_i(A)) = 1$
- **Divergent** if $\lim_{t \to \infty} |m(t)| = \infty$
  - $\text{Re}[\lambda_i(A)] > 0$ or $\text{Re}[\lambda_i(A)] = 0$ and $\mu'(\lambda_i(A)) > 1$

▼ MatLab example

Analyze the internal stability properties of the LTI system having the state matrix:

$$A = \begin{bmatrix} 0 & 1 & 0 \\ -1 & 0 & 0 \\ 0 & 0 & -1 \end{bmatrix}$$

```
A = [0 1 0; -1 0 0; 0 0 -1];

% Compute the eigenvalues of A
eigenvalues = eig(A)
```

▼ **BIBO stability of LTI systems**

A SISO LTI system is **BIBO** stable if the zero state output response is bounded for **all** bounded inputs:

$$\forall u_M \in (0, \infty), \exists y_m \in (0, \infty) : |u(t)| \leq u_M, \forall t \geq 0 \implies |y(t)| \leq y_M, \forall t \geq 0$$

Practical result: An LTI system is **BIBO stable** if and only if all the poles of the transfer function have **strictly negative** real parts.

Notice: Asymptotical stability **implies** BIBO stability, so: if $\mathrm{Re}[\lambda_i(A)] < 0$ for $i = 1, ..., n$, then the system is also BIBO stable!

▼ MatLab example

Analyze the BIBO stability of an LTI system having the following state space representation:

$$\dot{x}(t) = \begin{bmatrix} -1 & 2 \\ 1 & 0 \end{bmatrix} x(t) + \begin{bmatrix} 2 \\ 0 \end{bmatrix} u(t)$$
$$y(t) = \begin{bmatrix} 0.5 & -0.5 \end{bmatrix} x(t)$$

```
A = [-1 2; 1 0];
B = [2; 0];
C = [0.5 -0.5];
D = 0;

s = tf('s');

% Compute the H transfer function
sys = ss(A, B, C, D);
H = tf(sys);
H = zpk(minreal(H, 1e-2))

% Compute the poles of H and check the real part.
pole(H)
```

▼ Steady state response

▼ Step input

The steady state response $y_{ss}(t)$ due to a step input signal of amplitude $\bar{u}$: $u(t) = \bar{u}\epsilon(t)$ is given by $y_{ss}(t) = \bar{y}\epsilon(t)$, where $\bar{y} = \bar{u} \cdot H(0)$

▼ MatLab example

Compute, if possible, the steady state output response of an LTI system described by the following transform function in the presence of the input $u(t) = 2\epsilon(t)$

$H(s) = \frac{1}{s^3+2s^2+5.25s+4.25}$

```
s = tf('s');
H = 1/(s^3 + 2*s^2 + 5.25*s + 4.25);
H = zpk(minreal(H, 1e-2))
poles = pole(H)
u_bar_step = 2;
K = dcgain(H) * u_bar_step
```

Firstly we need to check if we **can** compute the steady state response (if the system is internally stable), all the poles of H have strictly negative real part, so the system is internally stable and we can proceed.

We get: $y_{ss}(t) = K\bar{u}\epsilon(t) = 0.4706\epsilon(t)$

▼ Sinusoidal input

The steady state response $y_{ss}(t)$ due to a sinusoidal input signal of shape $u(t) = \bar{u}\sin(\omega_0 t)$ is given by $y_{ss}(t) = \bar{y}(\omega_0)\sin(\omega_0 t + \phi(\omega_0))$, where:

- $\bar{y}(\omega_0) = \bar{u} \cdot |H(j\omega_0)|$
- $\phi(\omega_0) = \arg(H(j\omega_0))$

▼ MatLab example

Compute, if possible, the steady state output response of an LTI system described by the following transform function in the presence of the input $u(t) = 3\sin(0.1t)\epsilon(t)$

```
s = tf('s');
H = 1/(s^3 + 2*s^2 + 5.25*s + 4.25);
H = zpk(minreal(H, 1e-2))
poles = pole(H)
u_bar_sin = 3;
w0 = 0.1;
[mag, phi] = bode(H, w0)

% We should convert from degrees to radiants
phi_rad = phi / 180*pi
y_bar = mag * u_bar_sin
```

Firstly we need to check if we **can** compute the steady state response (if the system is internally stable), all the poles of H have strictly negative real part, so the system is internally stable and we can proceed.

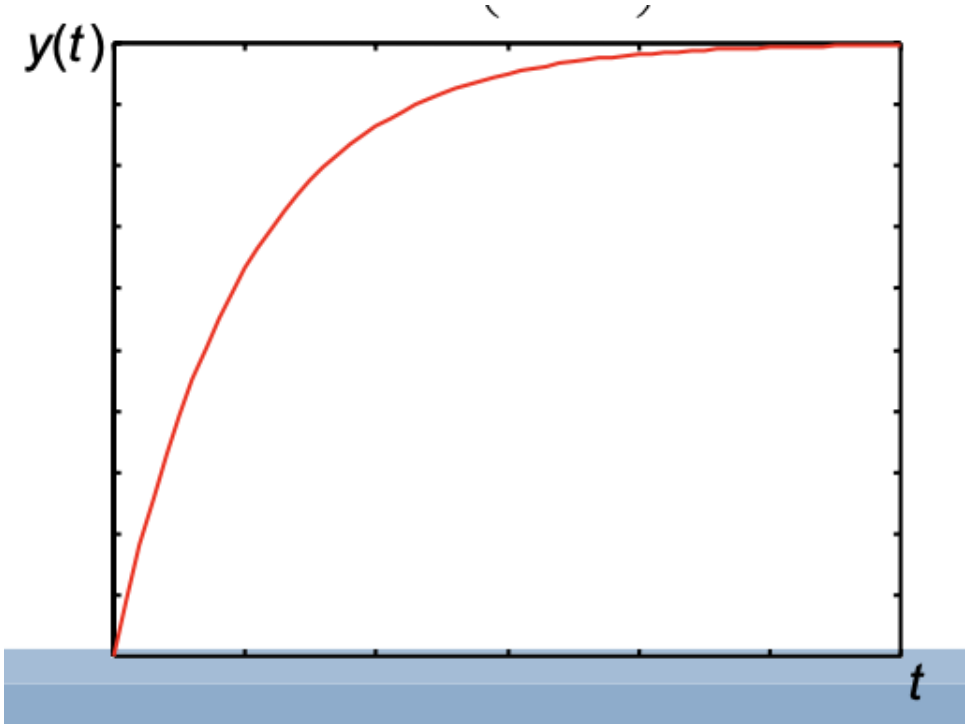
We get: $y_{ss}(t) = \bar{y}\sin(\omega_0 t + \phi)\epsilon(t) = 0.7038\sin(0.1t - 0.1232)\epsilon(t)$

▼ Analysis of the step response of prototype 1^{st} and 2^{nd} order systems

▼ First order system

Consider a first order system described by the transfer function $H(s) = \frac{K^*}{s-p}$, and let
$\tau = |\frac{1}{p}|$ and $K = -\frac{K^*}{p}$.
We can rewrite $H(s) = \frac{K}{1+\tau s}$.

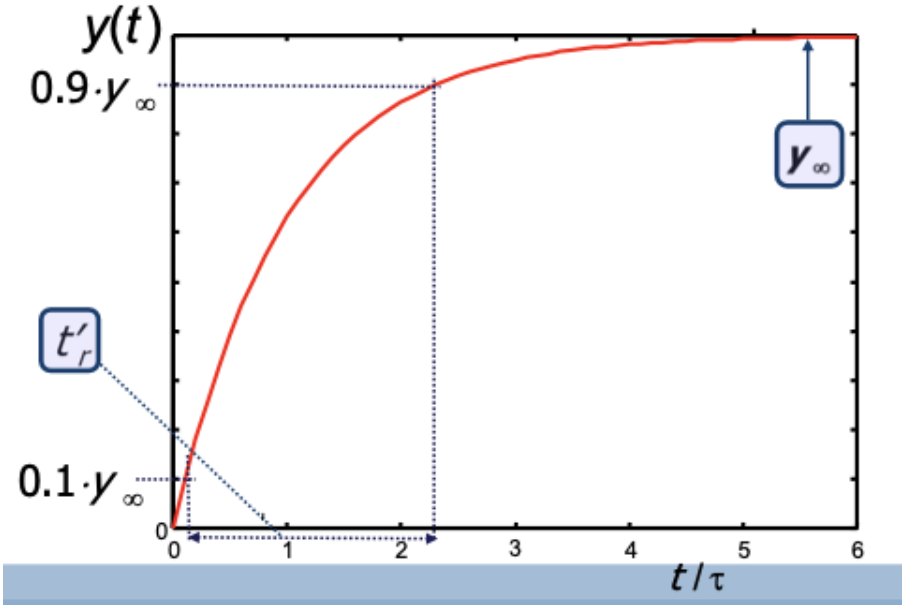
In the presence of a step input $u(t)$ with amplitude $\bar{u}$ the output response can be written as:
$y(t) = \bar{u}K(1 - e^{-\frac{t}{\tau}})$, for $t \geq 0$.



Graphical course of y(t)

We can compute its characteristics:

- **Steady state value**: $y_\infty = \lim_{t\to\infty} y(t) = K \cdot \bar{u}$
- **10%-90% rise time:** t_r' is the time required to the step response to go from the 10% to the 90% of the steady state value y_∞

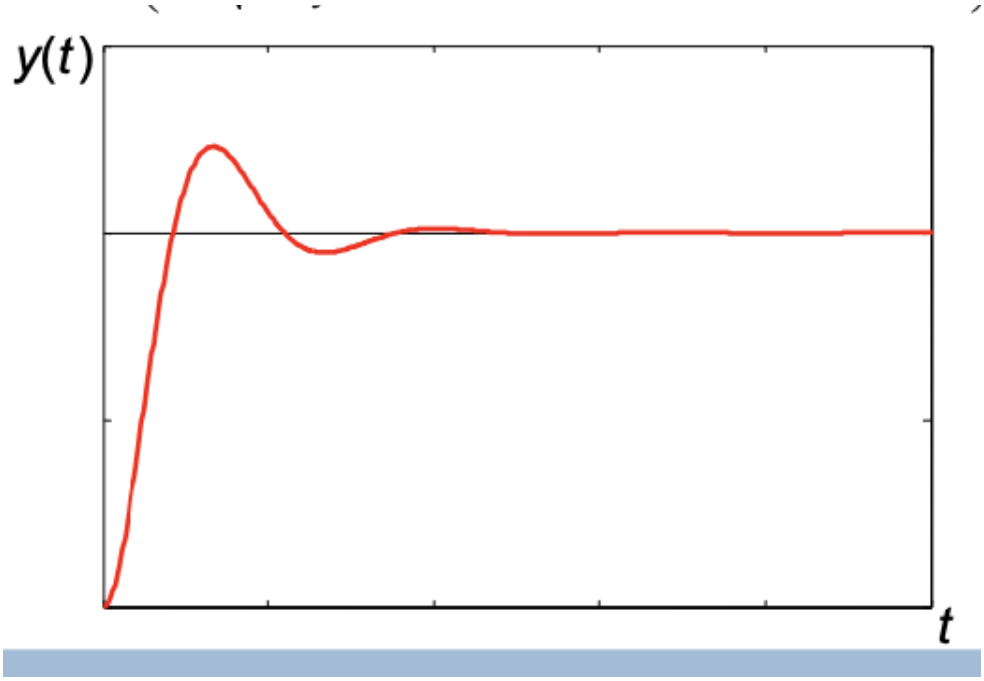


- **Settling time** $\pm\alpha\%$**:** $t_{s,\alpha\%}$ is the amount of time required to the step response to reach and stay within the $\pm\alpha\%$ of the y_∞

▼ Second order system

Consider a stable second order system described by the transfer function: $H(s) = K\frac{1}{1+s\frac{\zeta}{\omega_n}s+\frac{s^2}{\omega_n^2}} = K\frac{\omega_n^2}{s^2+2\zeta\omega_n s+\omega_n^2}$ and let $\tau = \frac{1}{\zeta\omega_n}$.

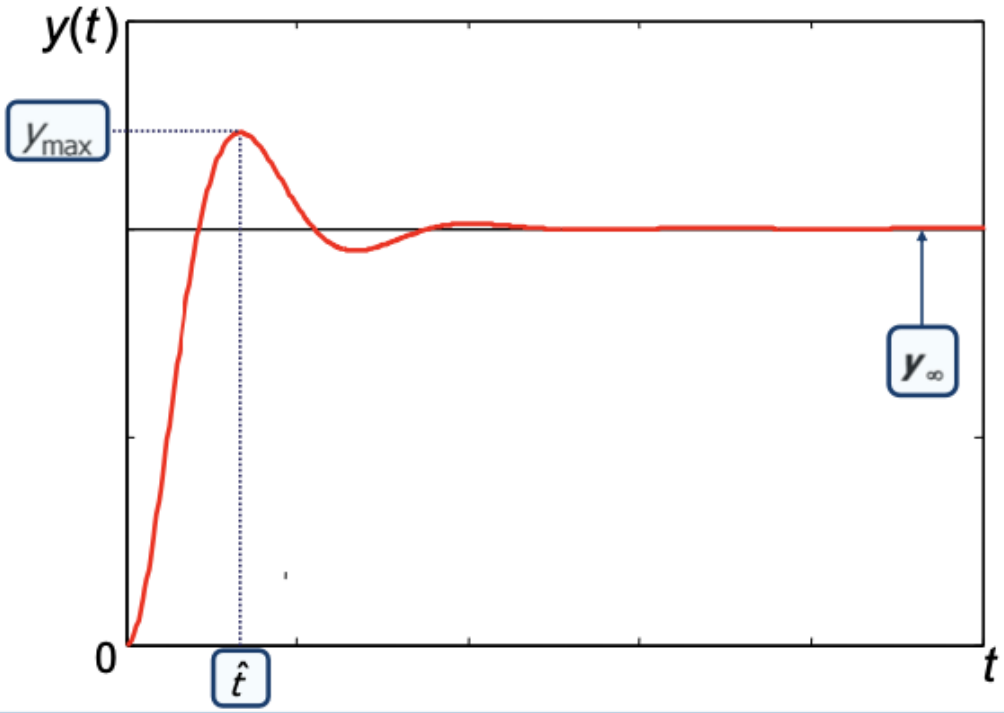
In the presence of a step input $u(t)$ with amplitude $\bar{u}$, the output response can be written as:
$y(t) = \bar{u}K(1 - \frac{1}{\sqrt{1-\zeta^2}}e^{-\zeta\omega_n t}\sin(\omega_n\sqrt{1-\zeta^2}t + \arccos(\zeta)))$, for $t \geq 0$



Graphical course of y(t)

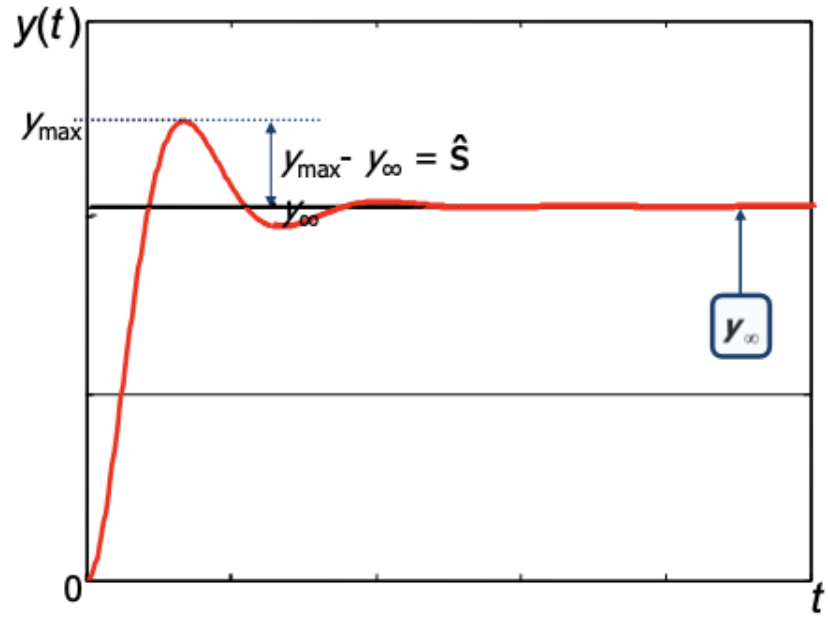
We can compute its characteristics:

- **Steady state value**: $y_\infty = \lim_{t\to\infty} y(t) = K \cdot \bar{u}$
- **Peak value**: $y_{max} = \max(y(t))$
- **Peak time**: $\hat{t}$ is the time instant when $y(t)$ reaches y_{max}.
  $\hat{t} = \frac{\pi}{\omega_n \sqrt{1-\zeta^2}}$

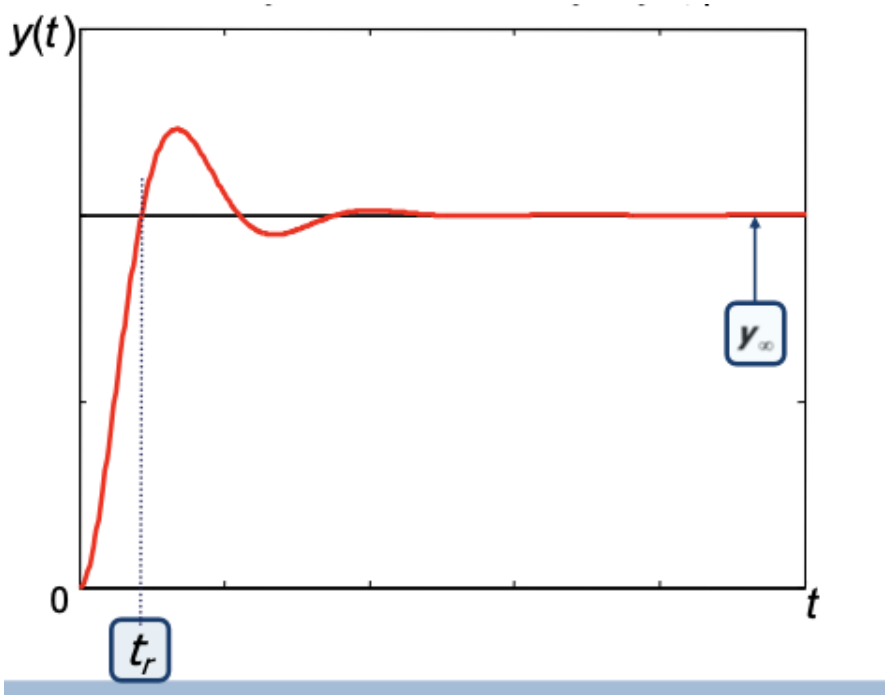


- **Maximum overshoot**: $\hat{s} = \frac{y_{max} - y_\infty}{y_\infty}$

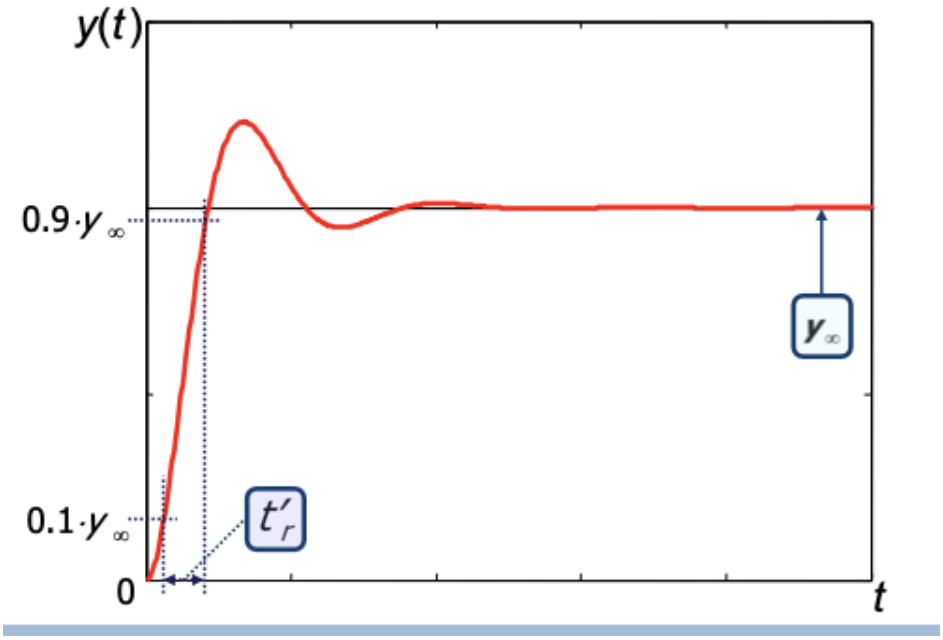
  $\hat{s} = e^{-\frac{\pi\zeta}{\sqrt{1-\zeta^2}}}$



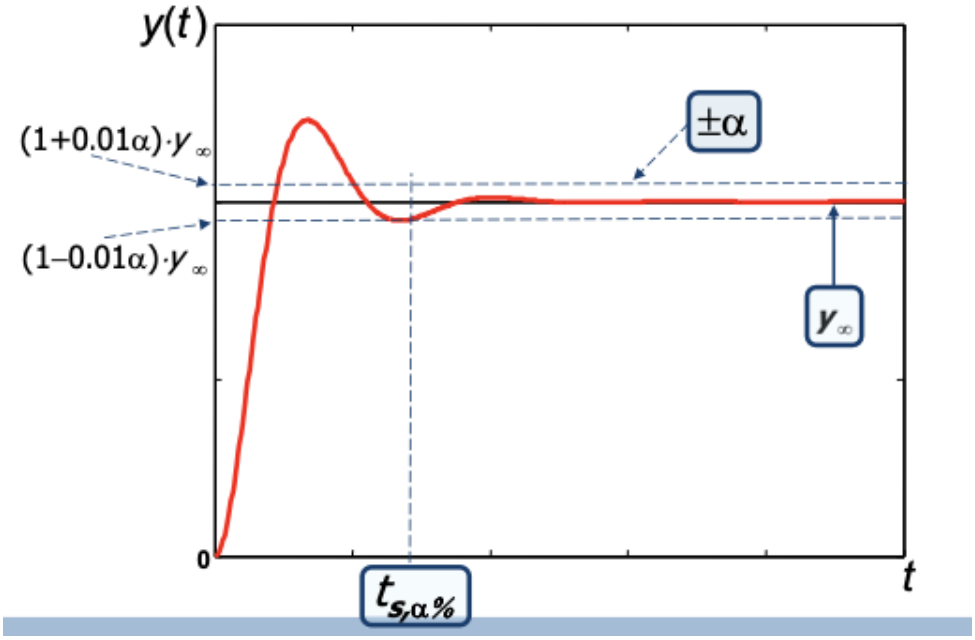
- **Rise time**: t_r is the time required to the step response to reach for the first time the steady state value y_∞
  $t_r = \frac{1}{\omega_n \sqrt{1-\zeta^2}}(\pi - \arccos(\zeta))$



- **10%-90% rise time**: t'_r is the time required to the step response to go from 10% to the 90% of y_∞
  $t'_r = \frac{2.16\zeta + 0.6}{\omega_n}$



- **Settling time** $\pm\alpha\%$**:** $t_{s,\alpha\%}$ is the amount of time required to the step response to reach and stay within the $\pm\alpha\%$ of y_∞

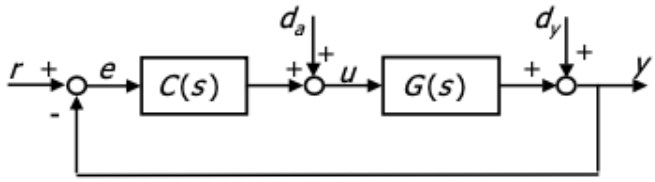


▼ Parameters formulas

$$\zeta = \frac{|\ln(\hat{s})|}{\sqrt{\pi^2+\ln^2(\hat{s})}}$$

$$\omega_n = \frac{\pi}{\hat{t}\sqrt{1-\zeta^2}}$$

▼ Feedback Control Systems



To analyze the stability and performance of the control architecture above we need to evaluate the effect of each input (r, d_a, d_y) on the output of each summing junction (y, u, e).
To do this we need to compute the transfer functions between each of the inputs r, d_a, d_y and the outputs y, u, e.

▼ Computation of the effect of input r on output y ($T(s)$)

$$y(s) = \frac{G(s)C(s)}{1+G(s)c(s)}r(s)$$

▼ Computation of the effect of input d_a on output y ($Q(s)$)

$$y(s) = \frac{G(s)}{1+G(s)C(s)}d_a(s)$$

▼ Computation of the effect of input d_y on output y ($S(s)$)

$$y(s) = \frac{1}{1+G(s)C(s)}d_y(s)$$

We can apply the principle of the superposition and compute
$y(s) = T(s)r(s) + Q(s)d_a(s) + S(s)d_y(s)$

In the same way we can compute the transfer functions for the other outputs:

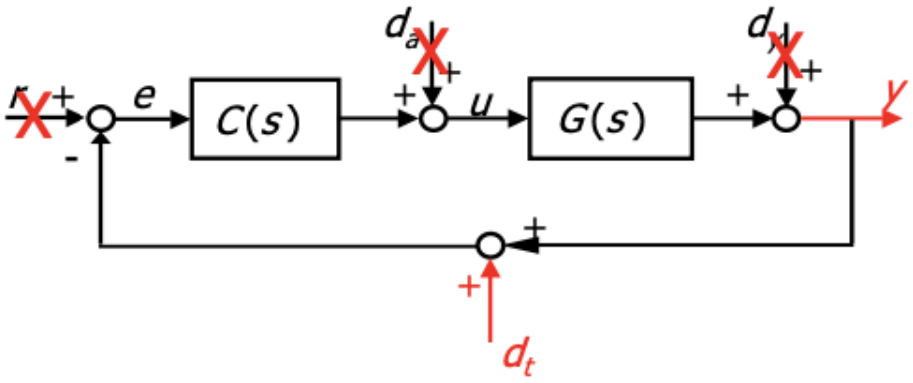
$e(s) = S(s)r(s) - Q(s)d_a(s) - S(s)d_y(s)$
$u(s) = R(s)r(s) + S(s)d_a(s) - R(s)d_y(s)$, in this case we consider $R(s) = \frac{C(s)}{1+G(s)C(s)}$

We can derive the transfer functions from the **loop transfer function** $L(s) = C(s)G(s)$:
$T(s) = \frac{L(s)}{1+L(s)}$, $Q(s) = \frac{G(s)}{1+L(s)}$, $S(s) = \frac{1}{1+L(s)}$, $R(s) = \frac{C(s)}{1+L(s)}$

▼ Sinusoidal disturbance attenuation

▼ Sensor noise

Consider the following feedback control feedback, supposed stable:



We want to attenuate the effect steady state of the sensor noise d_t on the controlled output y; in particular we focus on the class of **sinusoidal** signals in the form:
$d_t(t) = \delta_t \sin(\omega t), \omega \geq \omega_t$, given δ_t and ω_t.

At steady state, we have $y_{ss}(t) = \delta_t |T(j\omega)| \sin(\omega t + \arg(-T(j\omega)))$.

The steady state output error $|y_{d_t}^{\infty}| = \max_t |y_{ss}(t)| = \delta_t |T(j\omega)|$.

$|y_{d_t}^{\infty}|$ is required to be bounded by a given constant $|y_{d_t}^{\infty}| \leq \rho_t, \rho_t > 0$.

A design constraint on $|T(j\omega)|$ is obtained as $|T(j\omega)| \leq \frac{\rho}{\delta_t} = M_T^{HF}, \forall \omega \geq \omega_t$.

Note: a disturbance attenuation is obtained if $|T(j\omega)| << 1$

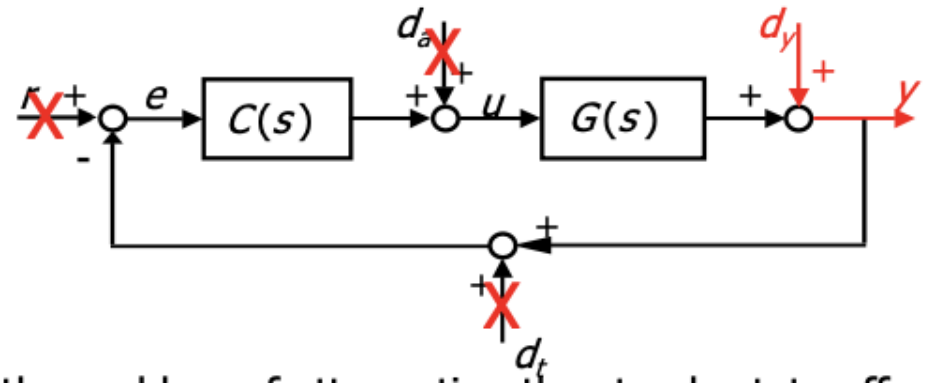
Since $|T(j\omega)| << 1$ holds for $\omega >> \omega_c$, the requirement on $y_{d_t}^{\infty}$ introduces a constaint on the value of ω_c such that $\omega_c << \omega_t$

Rule of thumb: $\omega_c \leq 0.1\omega_t$

On the Nichols plane, we can represent the constraint on $T(j\omega)$ as a constant magnitude locus; the points of $L(j\omega)$ corresponding to frequencies greater than ω_t must lie **below** the constant magnitude locus defined by $\frac{\rho_t}{\delta_t} = M_T^{HF}$

▼ Output disturbance

Consider the following feedback control system, supposed stable:



At steady state, in the presence of d_y we have
$y_{ss}(t) = \delta_y |S(j\omega)| \sin(\omega t + \arg(S(j\omega)))$

The steady state output error is defined as $|y_{d_y}^{\infty}| = \max_t(y_{ss}(t)) = \delta_y |S(j\omega)|$

It is required to be bounded by a given constant $|y_{d_y}^{\infty}| \leq \rho_y$, given $\rho_y > 0$

A design constraint on $|S(j\omega)|$ is obtained as $|S(j\omega)| \leq \frac{\rho_y}{\delta_y} = M_S^{LF} \; \forall \omega \leq \omega_y$

Note: the disturbance attenuation is obtained if $|S(j\omega)| << 1$

This relation holds for $\omega << \omega_c$, so we just introduced a requirement on ω_c:
$\omega_c >> \omega_y$

Rule of thumb: $\omega_c \geq 10\omega_y$

On the Nichols plane the constraint on $S(j\omega)$ can be represented as a constant magnitude locus; the points of $L(j\omega)$ corresponding to frequencies smaller than ω_y must lie **above** the constant magnitude locus defined by $\frac{\rho_y}{\delta_y} = M_S^{LF}$

Design Part

To design a loop function for our system, we should follow some steps:

▼ **Steady state design**

We study the steady state properties of

- The tracking error $e(t)$ in the presence of the reference r
- The output $y(t)$ in the presence of the disturbance $d_a(t)$ and $d_y(t)$

- For $|e_r^{\infty}|$:

$|e_r^\infty|$

$h \rightarrow$ $g \downarrow$	0 $r(t) = \rho\epsilon(t)$	1 $r(t) = \rho t\epsilon(t)$	2 $r(t) = \rho\frac{t^2}{2}\epsilon(t)$
0	$\left\|\frac{\rho}{1+K_0}\right\|$	∞	∞
1	0	$\left\|\frac{\rho}{K_1}\right\|$	∞
2	0	0	$\left\|\frac{\rho}{K_2}\right\|$

$$K_g = \lim_{s\to 0} s^g L(s)$$

- For $|y_{d_y}^\infty|$:

$|y_{dy}^\infty|$

$h \rightarrow$ $g \downarrow$	0 $d_y(t) = \delta_y\epsilon(t)$	1 $d_y(t) = \delta_y t\epsilon(t)$	2 $d_y(t) = \delta_y\frac{t^2}{2}\epsilon(t)$
0	$\left\|\frac{\delta_y}{1+K_0}\right\|$	∞	∞
1	0	$\left\|\frac{\delta_y}{K_1}\right\|$	∞
2	0	0	$\left\|\frac{\delta_y}{K_2}\right\|$

$$K_g = \lim_{s\to 0} s^g L(s)$$

- For $|y_{d_a}^{\infty}|$:

$|y_{d_a}^{\infty}|$

$h \rightarrow$ $g_c \downarrow$	0 $d_a(t) = \delta_a \epsilon(t)$	1 $d_a(t) = \delta_a t\epsilon(t)$	2 $d_a(t) = \delta_a \frac{t^2}{2}\epsilon(t)$
0	$\left\lvert\frac{\delta_a}{K_0}\right\rvert$	∞	∞
1	0	$\left\lvert\frac{\delta_a}{K_1}\right\rvert$	∞
2	0	0	$\left\lvert\frac{\delta_a}{K_2}\right\rvert$

Where $K_0 = \begin{cases} K_c & \text{if } G(s) \text{ has poles in } 0 \\ \frac{1+K_C K_G}{K_G} & \text{if } G(s) \text{ has not poles in } 0 \end{cases}$, $K_{g_c} = \lim_{s\to 0} s^{g_c} C(s)$

The steady state requirements depend on the **system type** (number of poles at the origin of the $L(s) = C(s)G(s)$) and on the **gain** $K_g = \lim_{s\to 0} s^g L(s)$.

Since $G(s)$ is fixed, the requirements can be translated on suitable choices of the values K_c and g_c.

We can express $C(s)$ as $C(s) = C_{ss}(s)C_T(s)$, where $C_{ss} = \frac{K_c}{s^g C}$ is the **steady state controller** and $C_T(s) = \frac{N_C'(s)}{D_c'(s)}, \lim_{s\to 0} \frac{N_C'(s)}{D_C'(s)} = 1$ is the **transient controller**.

To shape C_{ss} we should:

- Add a suitable number of poles at the origin to get the required value of g (the system type).
  In the tables above, h is the **system type**, while g is the number of poles to add.
- Tune the value of K_C in order to get the required value of $K_g = K_C K_G$.
  Remember that, since K_G is fixed, the required value of K_g can be obtained through a suitable choice of K_C.
  We can express $C(s) = C_{ss}(s)C_T(s)$, where:
  $C_{ss}(s) = \frac{K_C}{s^{g_c}}, C_T = \frac{N_c'(s)}{D_c'(s)}$ as stated above.

▼ How to choose the sign of K_C?

- If the plant transfer function $G(s)$ does not include any pole/zero with **strictly positive** real part, then the sign of Kc should be chosen such that $K_g = K_G * K_C > 0$
- If $G(s)$ includes poles/zeros with **strictly positive** real part, then the sign of K_c is chosen according to the following procedure:
  1. Draw the Nyquist diagram of $L'(s) = C_{ss}G(s)$ using the sign of K_c such that $K_g = K_G * K_C > 0$
  2. If $L'(s)$ leads to a stable feedback system (the number of poles of $T(s) = \frac{L(s)}{1+L(s)}$ with **strictly positive real part** is 0) then this is the correct sign choice, provided that $C_T(s)$ will be designed to avoid significant significant modifications of the frequency response $L'(s)$ near the critical point.
  3. If $L'(s)$ leads to un unstable feedback system, discuss whether a suitable choice of $C_T(s)$ may be able to stabilize the feedback system (e.g. through a phase lead action) around the critical point
  4. If the previous steps fail, repeat the procedure changing the sign of K_C to verify that it's the correct choice

▼ **Transient design**

Transient requirements of a feedback system are defined considering the controlled output response $y(t)$ when the reference signal $r(t)$ is a step function.

The following parameters are used to define the transient performance of a feedback control system for the time response:

- Maximum overshoot $\hat{s}$ → ***accuracy***
- Rise time t_r → ***Trigger off quickness***
- Settling time $t_{s,\alpha\%}$ → ***extinction quickness***

These can be translated into **frequency response** requirements:

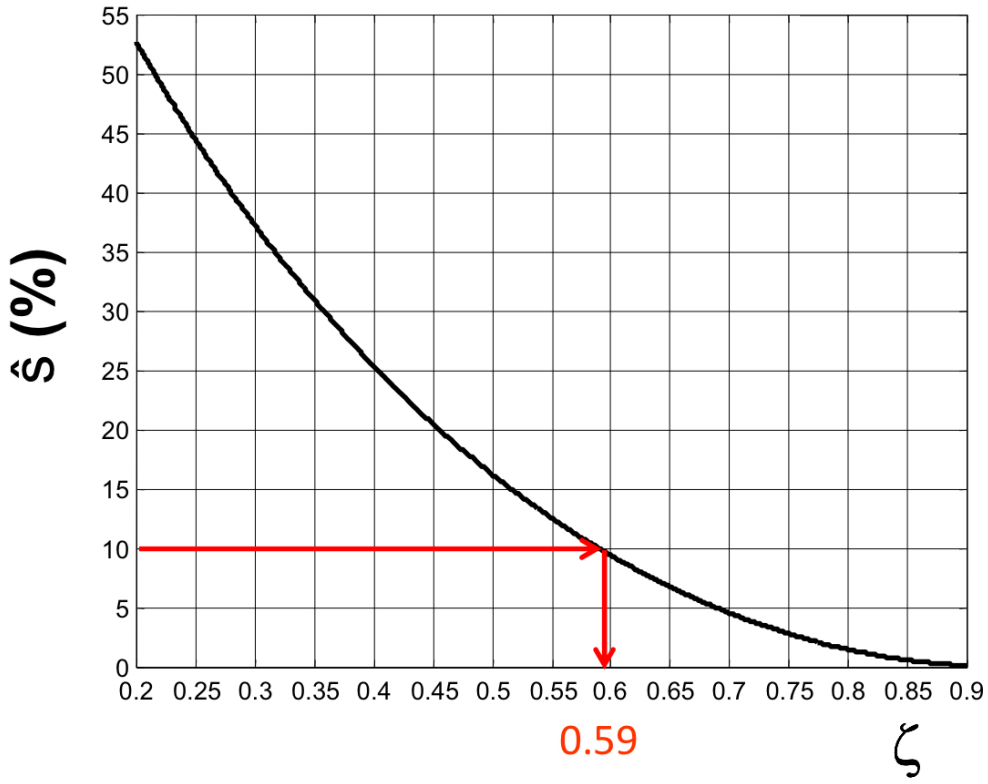
- Resonant peak of the complementary sensitivity function T_p
- Resonant peak of the sensitivity function S_p
- Crossover frequency ω_c

▼ **Computation of T_p and S_p**

Using a graphical procedure we can obtain the **minimum dampening coefficient** ζ from the requirement on the maximum value of the overshoot $\hat{s}$.

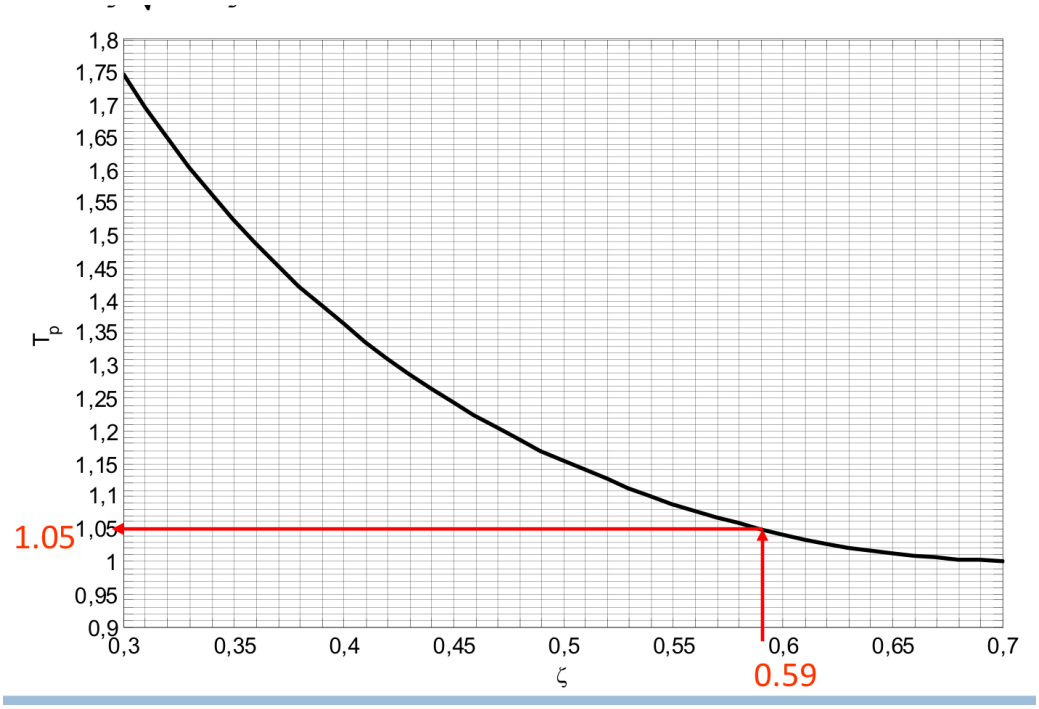
- Formula: $\zeta = \frac{|\ln(\hat{s})|}{\sqrt{\pi^2+\ln^2(\hat{s})}}$
- Graphical procedure:

$$\hat{S} = e^{-\frac{\pi\zeta}{\sqrt{1-\zeta^2}}}$$

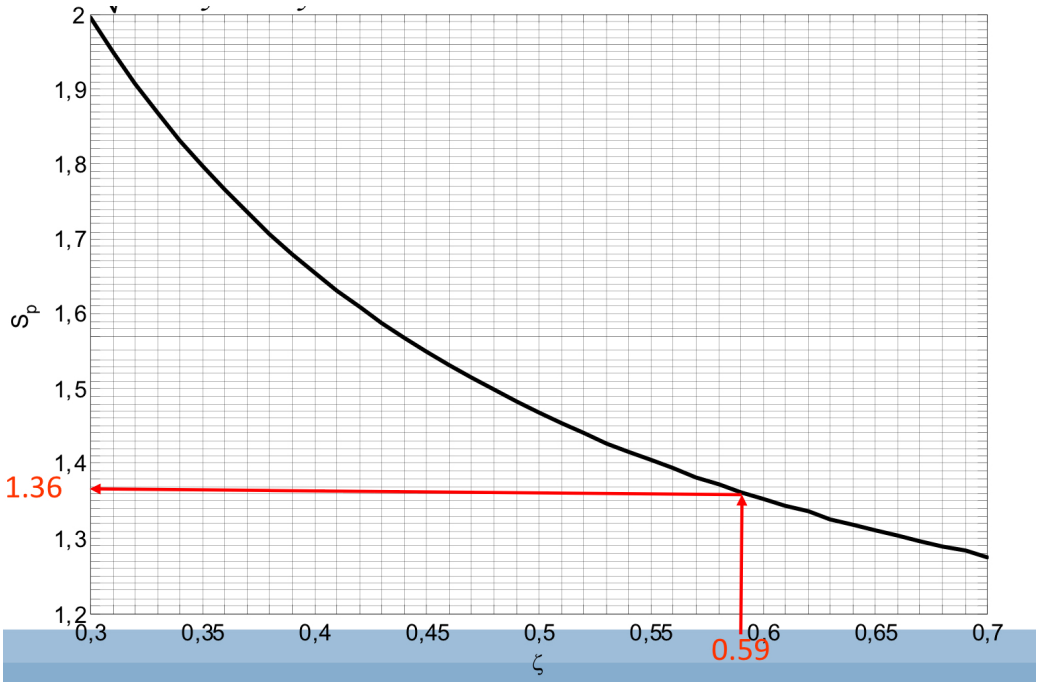


From ζ, using the formulas or the graphical approach we can get the desired values of T_p and S_p.

- Formulas: $T_p = \frac{1}{2\zeta\sqrt{1-\zeta^2}}$, $S_p = \frac{2\zeta\sqrt{2+4\zeta^2+2\sqrt{1+8\zeta^2}}}{\sqrt{1+8\zeta^2}+4\zeta^2-1}$
- Graphical procedure:
  - T_p:



- S_p:



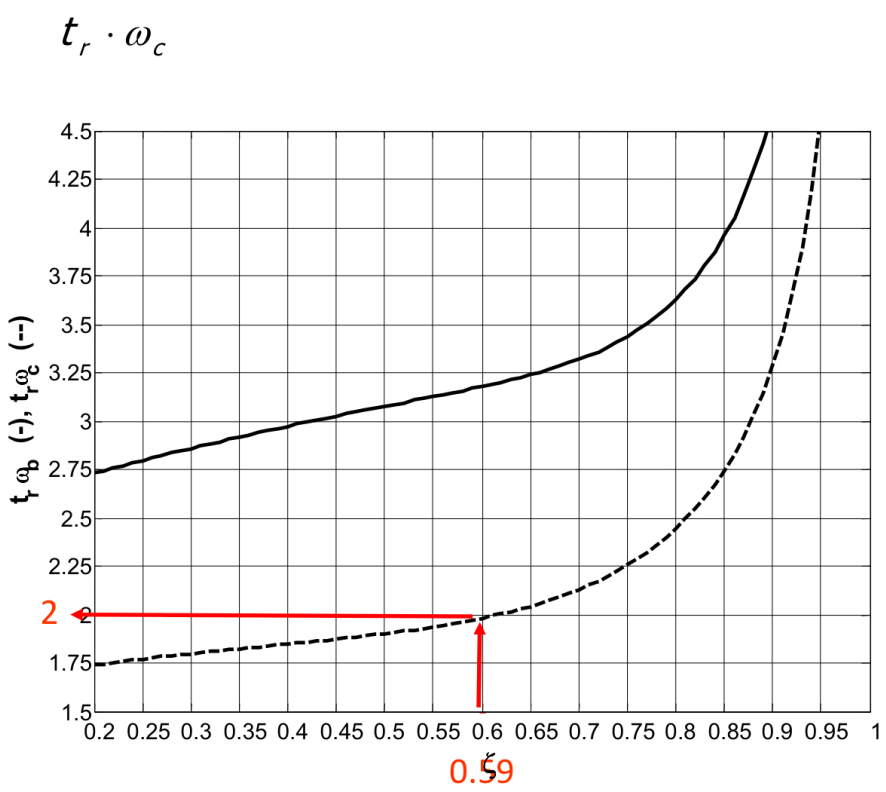
Note: Remember that the obtained values of T_p and S_p are not expressed in dB, and they need to be converted using the formula $T_{p_{dB}} = 20\log(T_p)$.

▼ **Computation of ω_c**

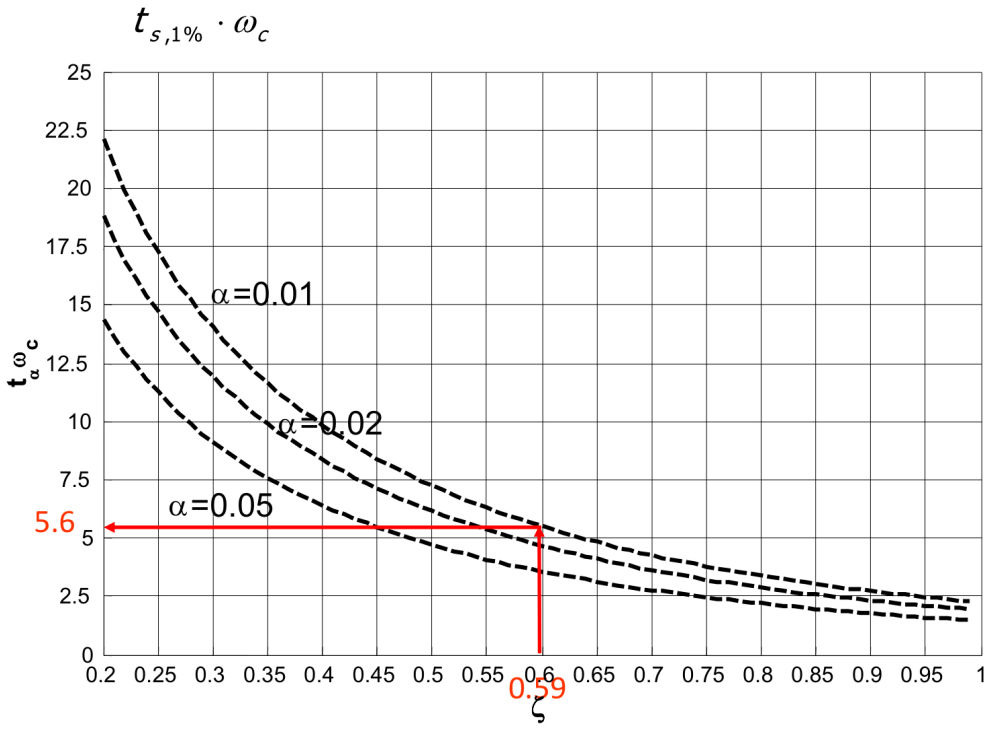
The rise time t_r and the settling time $t_{s,\alpha\%}$ requirements can be translated in a crossover frequency ω_c requirement.

Using the formulas or the graphical procedure we get 2 different values of ω_c and we should keep **the maximum** between the two: $\omega_{c,des} = \max(\omega_{c,t_r}, \omega_{c,t_{s,\alpha\%}})$.

- Using t_r requirement:
  - Formula: $t_r \cdot \omega_c = \frac{1}{\sqrt{1-\zeta^2}}(\pi - \arccos(\zeta))\sqrt{\sqrt{1+4\zeta^4} - 2\zeta^2}$
  - Graphical procedure:



- Using $t_{s,\alpha\%}$ requirement:
  - Formula: $t_{s,\alpha\%} \cdot \omega_c = \frac{\ln(\frac{100}{\alpha})}{\zeta}\sqrt{\sqrt{1+4\zeta^2} - 2\zeta^2}$
  - Graphical procedure:



▼ **Transient controller design**

After obtaining the desired values of T_p, S_p, ω_c, but before the design of the transient controller C_T we must follow some preliminary steps:

- Consider the Loop transfer function $L'(s) = C_{ss}(s)G(s)$ obtained during the steady-state design.
  Plot the frequency response on the Nichols plane
- Mark the point corresponding to ω_c
- Plot the constant magnitude loci T_p, S_p

```
L1 = Css * G;
figure, nichols(L1)
hold on
T_grid(Tp)
S_grid(Sp)
hold off
```

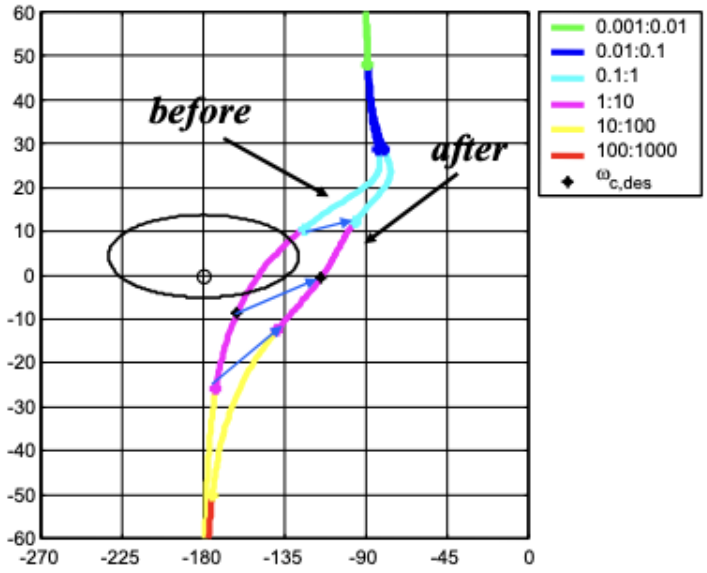
We need to shape $L'(s)$ in order for the point corresponding to ω_c to be **tangent** to the constant magnitude loci and to be on the x-axis.

We have multiple approaches to do this:

▼ **Lead network:** phase lead

$$C_D(s) = \frac{1+\frac{s}{\omega_D}}{1+\frac{s}{m_D\omega_D}}, \omega_D > 0, m_D > 1$$

The lead network introduces a desired **phase lead** (shift) and a **possibly magnitude increase**.



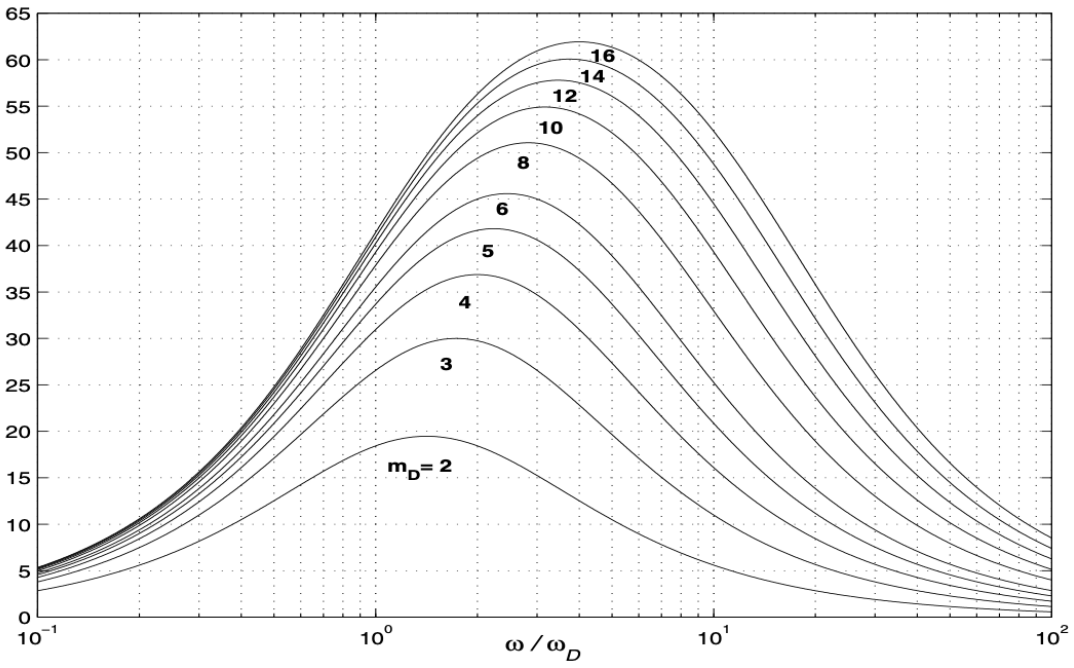
Effect of a phase lead

The greated m_D the greater the phase lead.

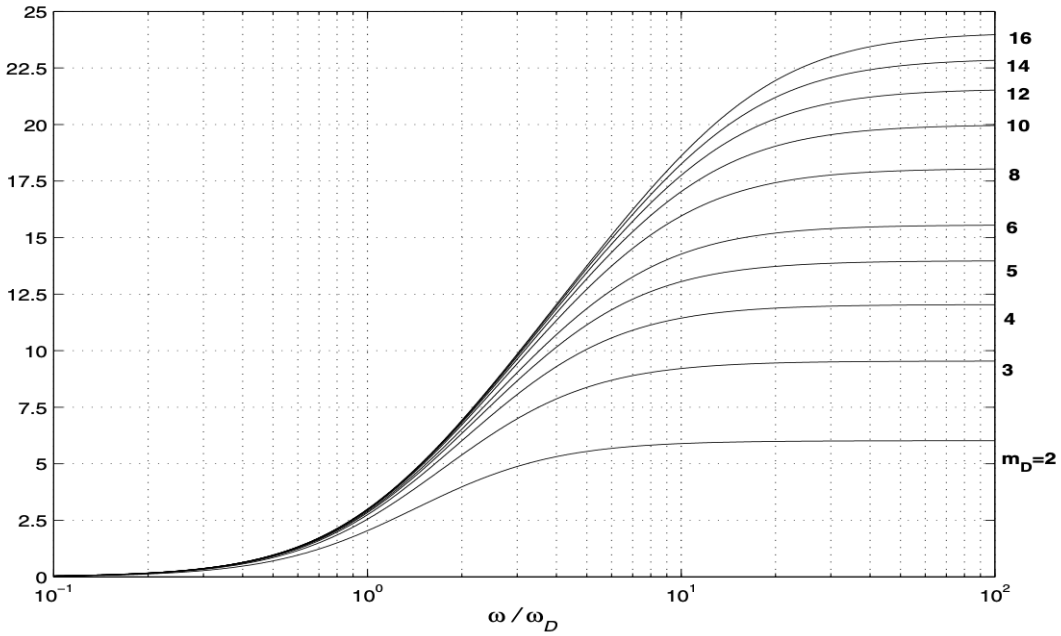
Guidelines for the design:

- Quantify the amount of the phase lead $\Delta\phi$ needed at ω_c in order to shift the value of $\arg(L'(j\omega_c))$ outside the influence of the magnitude loci.
- m_D is chosen on the basis of the required value of $\Delta\phi$
- ω_D is fixed to obtain that the phase lead $\Delta\phi$ occurs exactly at ω_c

We use the following network diagrams to choose m_D and ω_D.



On the left: the desired phase lead



On the left: the magnitude increase (side effect), on the right the value of mD

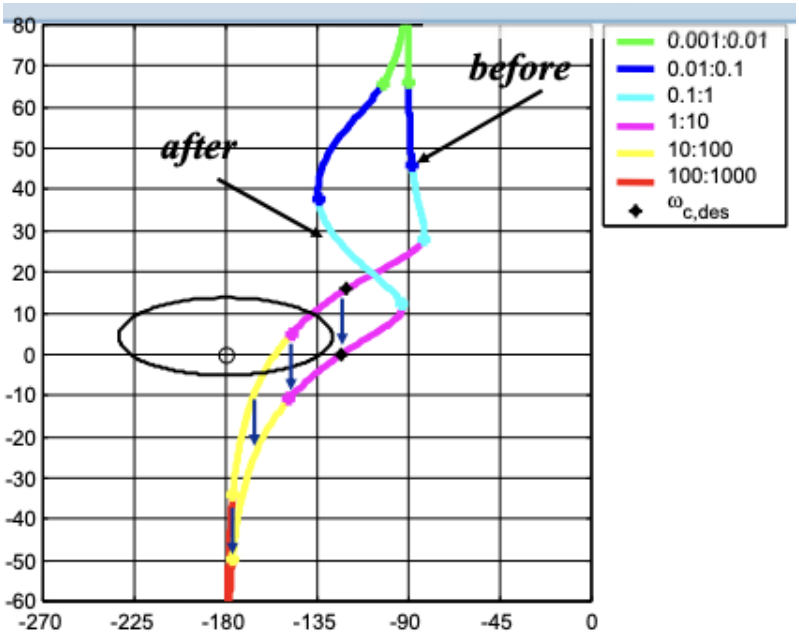
Notice: for leads greater than 60°, it is recommended to use multiple lead network with the formula:

$$C_D(s) = \frac{1+\frac{s}{\omega_{D1}}}{1+\frac{s}{m_{D1}\omega_{D1}}} \frac{1+\frac{s}{\omega_{D2}}}{1+\frac{s}{m_{D2}\omega_{D2}}} \dots$$

Notice: Remember that $\omega_{norm} = \frac{\omega}{\omega_D}$ and $\omega_D = \frac{\omega_c}{\omega_{norm}}$.
We choose the phase lead and m_D, then ω_{norm} is a consequence from the charts above.

▼ **Lag network**: magnitude decrease

$$C_I(s) = \frac{1+\frac{s}{m_I\omega_I}}{1+\frac{s}{\omega_I}}, \omega_I > 0, m_I > 1$$



The effect of a lag network

The **lag network** introduces a desired **magnitude decrease** and a **possible undesired phase lag.**
The greater m_I the greater the magnitude decrease.

Guidelines for the design:

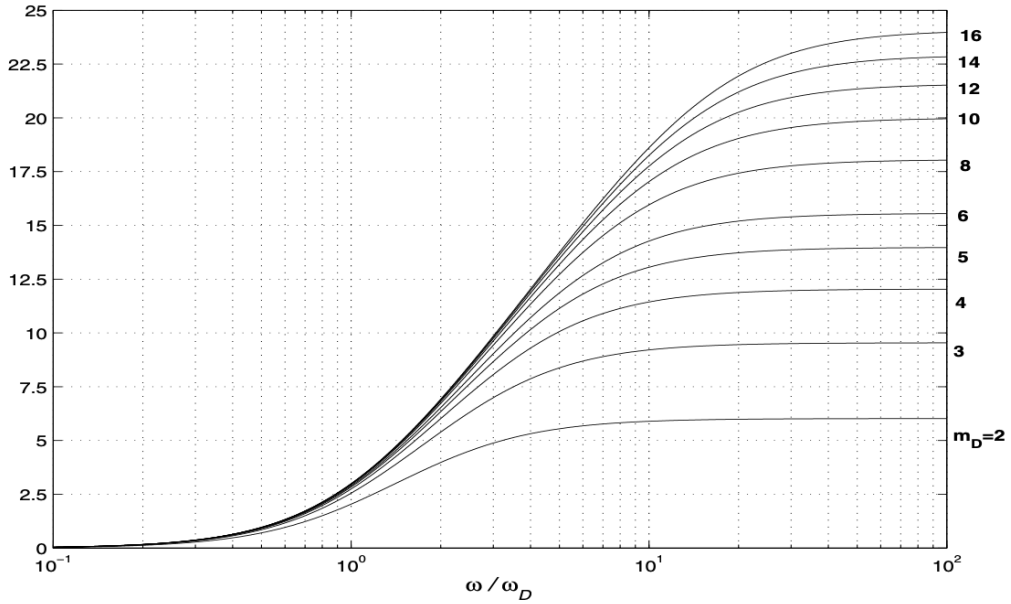
- Quantify the amount of magnitude decrease $\Delta\phi$ needed at ω_c in order to shift the value of $\arg(L'(j\omega_c))$ outside the influence of the magnitude loci
- m_I is chosen on the basis of the required value of $\Delta\phi$
- ω_I is fixed to obtain that the magnitude decrease $\Delta\phi$ occurs exactly at ω_c

We can use 2 different methods to decide the value of ω_i, m_I:

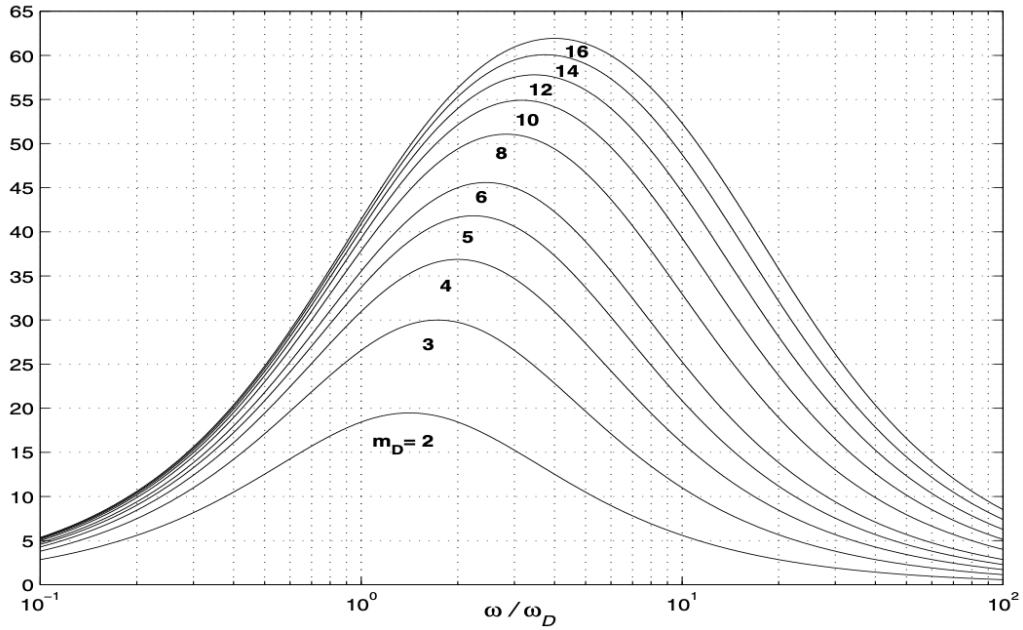
1. **Empyrical method**: To limit the phase lag at ω_c, the zero of the lag network $m_I\omega_I$ should be *sufficiently far* from ω_c.

We have $\omega_c \approx \alpha m_I \omega_I$, $\alpha >> 1 \rightarrow \omega_I = \frac{\omega_c}{\alpha m_I}$.
As a rule of thumb, start with $\alpha = 10$ and try to go down with its value.
Increasing α decreases the phase lag introduced at ω_c.

2. **Precise method**: using the tables below we get m_I and $\omega_I = \frac{\omega_c}{\omega_{norm}}$



On the left: the magnitude **decrease**, on the right the value of ml

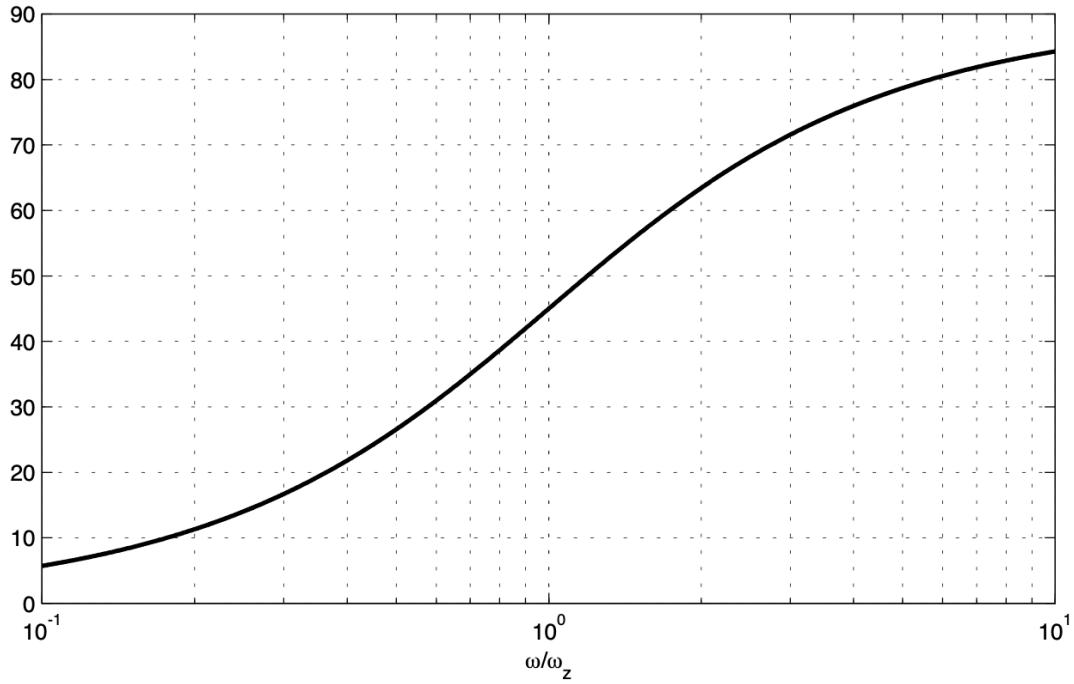


On the left: the desired phase lag

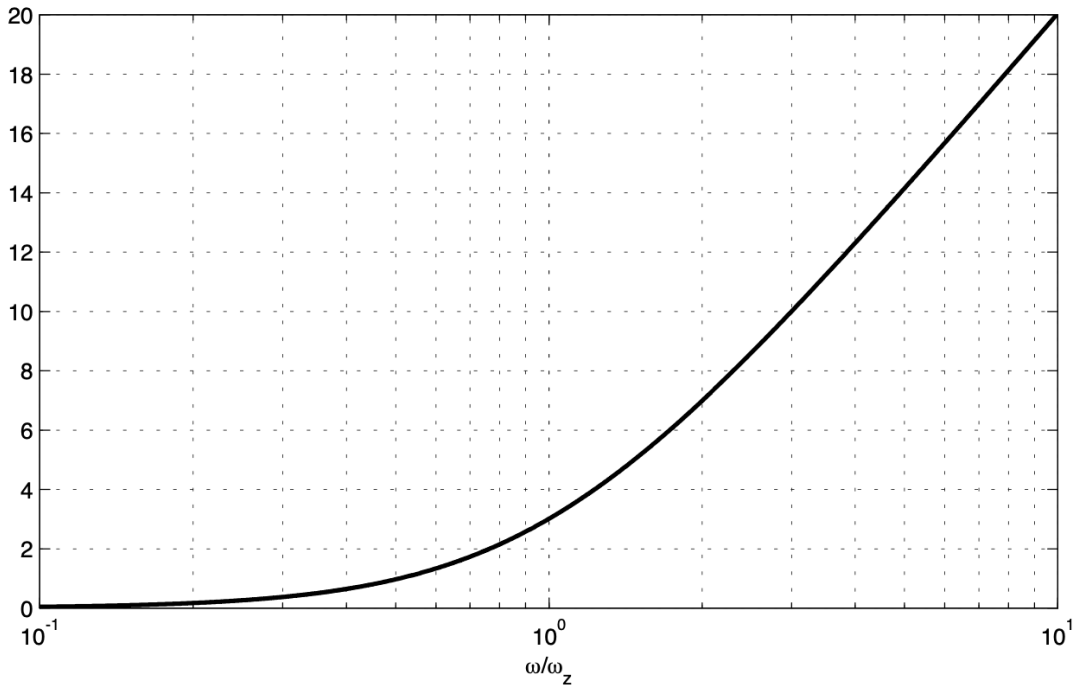
▼ **Real negative zero**: phase lead

$C_Z(s) = 1 + \frac{s}{\omega_Z}, \omega_z > 0$

ω_Z can be choosen using the tables below:



Phase lead



Magnitude increase

$\omega_{norm} = \frac{\omega}{\omega_Z} \rightarrow \omega_Z = \frac{\omega_c}{\omega_{norm}}$

Notice: for leads greater than 90°, it is recommended to use multiple real negative zero with the formula: $C_Z(s) = (1 + \frac{s}{\omega_{Z1}})(1 + \frac{s}{\omega_{Z2}})...$

Issues: since the transfer function of a real negative zero is **not proper**, it's not guaranteed that in general the final overall controller $C(s) = C_{ss}(s)C_Z(s)$ is proper.
If $C(s)$ is not proper, we need to add **closure poles** that have to be placed at higher frequencies with respect to highest frequency of the controller zeros.
$C(s) = \frac{K_C(1+\frac{s}{\omega_{Z1}})(1+\frac{s}{\omega_{Z2}})}{s(1+\frac{s}{\omega_P})}, \omega_P >> \max(\omega_{Z1}, \omega_{Z2})$

Remember: let's consider the controller
$C(s) = C_{ss}(s)C_z(s) = \frac{K_C(1+\frac{s}{\omega_{Z1}})(1+\frac{s}{\omega_{Z2}})...(1+\frac{s}{\omega_{Zn_z}})}{s^{g_c}}$
It is **proper** if $g_c \geq n_Z$ (the degree of the denominator is greater than or equal the degree of the numerator).

▼ Sensor noise: Steady State Response to Sinusoidal Disturbances

At steady state, in the presence of **sinusoidal** $d_t = \delta_t \sin(\omega_t)$ we have
$y_{ss} = \delta_t |T(j\omega)| \sin(\omega t + \arg(-T(j\omega)))$.

The steady state output error is $|y_{d_t}^{\infty}| = \max(|y_{ss}(t)|) = \delta_t |T(j\omega)|$.

The steady state output error is required to be bounded by a given constant $|y_{d_t}^{\infty}| \leq \rho_t$.

This constant can be translated into a Constant Magnitude Locus $M_T^{HF} = \frac{\rho_t}{\delta_t}$, and this introduces a **constraint** on the value of ω_c such that $\omega_c \leq 0.1\omega_t$.

▼ **Output Disturbance: Steady State Response to Sinusoidal Disturbances**

At steady state, in the presence of **sinusoidal** $d_y = \delta_y \sin(\omega_y)$ we have
$y_{ss} = \delta_y |S(j\omega)| \sin(\omega t + \arg(S(j\omega)))$.
The steady state output error is $|y_{d_y}^{\infty}| = \max(|y_{ss}(t)|) = \delta_y |S(j\omega)|$.

The steady state output error is required to be bounded by a given constant $|y_{d_y}^{\infty}| \leq \rho_y$.

This constant can be translated into a Constant Magnitude Locus $M_S^{LF} = \frac{\rho_y}{\delta_y}$, and this introduces a **constraint** on the value of ω_c such that $\omega_c \geq 10\omega_y$.

▼ **Actuator Saturation**

Physical limitations actuator devices impose hard constraints on the control input $u(t)$.

The above described actuator limitations, impose to $u(t)$ a **saturation constraint** of the form $|u(t)| \leq u_M, \ \forall t \geq 0$.

The saturation constraints can be described as a nonlinear static function of the control input.

To satisfy this requirement the only approach is "a posteriori". We need to **simulate** the behaviour of the system (with a step input) and check if the requirement is met.

If it's not, a common procedure is to **reduce the value of the actual crossover frequency ω_c.**

Keep in mind: reducing the value of ω_c may cause unsatisfaction of transient requirements.

Computer Engineering only

▼ **Equilibrium solutions of dynamical systems**

A constant state $\bar{x}$ is an **equilibrium solution** of the nonlinear dynamical system $\dot{x}(t) = f(x(t), u(t)), \ y(t) = g(x(t), u(t))$ if, in the presence of the **constant** input $u(t) = \bar{u}$ and the initial condition $x(0) = \bar{x}$, it results
$x(t) = \bar{x} \ \forall t \geq 0$.

- The input $\bar{u}$ is said **equilibrium input**

- The couple $(\bar{x}, \bar{u})$ is said **equilibrium point**
- The output $\bar{y} = g(\bar{x}, \bar{u})$ is said **equilibrium output**

An equilibrium point $(\bar{x}, \bar{u})$ of the dynamical system $\dot{x}(t) = f(x(t), u(t))$ satisfies the condition: $f(\bar{x}, \bar{u}) = 0$.

For LTI systems $\dot{x}(t) = Ax(t) + Bu(t)$ the equilibrium condition $f(\bar{x}, \bar{u})$ becomes $A\bar{x} + B\bar{u} = 0$, $B\bar{u} \in \mathcal{I}m(A\bar{x}) \implies \bar{x} = -A^{-1}B\bar{u}$ if $\det(A) \neq 0$.

▼ **Linearization of nonlinear systems**

Real world systems are, in general, **nonlinear**, but in the neighbourhood of a given solution their behaviour can be approximated through suitable linear models known as **linearized models**.

The objective is to compute a linear dynamical model that can accurately approximate the behaviour of a nonlinear system in a neighbourhood of an equilibrium solution.

- **Motivation**: Consider the following problem:
  Given the nonlinear dynamical system:
  $\dot{x}(t) = f(x(t), u(t))$
  $y(t) = g(x(t), u(t))$
  compute the solution in the presence of an input of the form:
  $u(t) = \bar{u} + \delta u(t), ||\delta u(t)|| << ||\bar{u}||$ and of the initial condition:
  $x(0) = \bar{x} + \delta x(0), ||\delta x(0)|| << ||\bar{x}||$, where $(\bar{x}, \bar{u})$ is a given equilibrium point of the considered system.
  In order to compute the required solution, we need to solve the following Cauchy problem: $\dot{x}(t) = f(x(t), u(t)), u(t) = \bar{u} + \delta u(t), x(0) = \bar{x} + \delta x(0)$. It is in general very difficult to compute the solution to it.
- **Preliminaries**: a viable approach is to compute an approximation $\tilde{x} = \bar{x} + \delta x(t) \approx x(t)$ where $\delta x(t)$ is the solution to a suitable LTI model of the form $\delta\dot{x}(t) = A\delta x(t) + B\delta u(t)$ in the presence of the input $\delta u(t)$ and of the initial condition $\delta x(0)$.
  Such a linear model will be derived through a ***linearization*** process applied to the nonlinear state representation $\dot{x}(t) = f(x(t), u(t))$
  $y(t) = g(x(t), u(t))$ in a neighbourhood of the equilibrium point $(\bar{x}, \bar{u})$.
- **System variations**: To derive the linearized model, let's define the **state variation** $\delta x(t)$ with respect to the solution $x(t)$ to the considered Cauchy problem:
  $\dot{x}(t) = f(x(t), u(t)), u(t) = \bar{u} + \delta u(t), x(0) = \bar{x} + \delta x(0)$ and the equilibrium

state $\bar{x}$: $\delta x(t) = x(t) - \bar{x} \in \mathcal{R}^n$.
Similarly, input and output variations are defined as:
$\delta u(t) = u(t) - \bar{u} \in \mathcal{R}^p, \delta y(t) = y(t) - \bar{y} \in \mathcal{R}^q$.

- **Computing the system variation**: The dynamic behaviour of the state variation $\delta x(t)$ is governed by the differential equation $\delta \dot{x}(t) = \dot{x}(t) - \dot{\bar{x}} = f(x(t), u(t))$.
  $f(x(t), u(t))$ can be approximated by its Taylor expansion in a neighbourhood of $f(\bar{x}, \bar{u})$ truncated at the **linear** term:
  $f(x(t), u(t)) = \frac{\partial f(x,u)}{\partial x}|_{x=\bar{x},u=\bar{u}} \delta x(t) + \frac{\partial f(x,u)}{\partial u}|_{x=\bar{x},u=\bar{u}} \delta u(t)$
  Therefore, the approximate behaviour of the state variation $\delta x(t)$ can be computed through the solution to the following linear differential equation:
  $\delta \dot{x}(t) = \frac{\partial f(x,u)}{\partial x}|_{x=\bar{x},u=\bar{u}} \delta x(t) + \frac{\partial f(x,u)}{\partial u}|_{x=\bar{x},u=\bar{u}} \delta u(t) = A\delta x(t) + B\delta u(t)$.
  A similar procedure can be applied to compute $\delta y(t)$:
  $\delta y(t) = y(t) - \bar{y} = g(x(t), u(t)) - g(\bar{x}, \bar{u}) = \frac{\partial g(x,u)}{\partial x}|_{x=\bar{x},u=\bar{u}} \delta x(t) + \frac{\partial g(x,u)}{\partial u}|_{x=\bar{x},u=\bar{u}} \delta u(t) = C\delta x(t) + D\delta u(t)$
- **Linearized system**: The linearized system is given by:
  $\delta \dot{x}(t) = A\delta x(t) + B\delta u(t),\ \delta x(0) = x(0) - \bar{x}$
  $\delta y(t) = C\delta x(t) + D\delta u(t),\ \delta u(t) = u(t) - \bar{x}$
  $$\begin{cases} A = \frac{\partial f(x(t),u(t))}{\delta x}|_{x=\bar{x},u=\bar{u}} & \text{Jacobian of } f \text{ wrt } x \\ B = \frac{\partial f(x(t),u(t))}{\delta u}|_{x=\bar{x},u=\bar{u}} & \text{Jacobian of } f \text{ wrt } u \\ C = \frac{\partial g(x(t),u(t))}{\delta x}|_{x=\bar{x},u=\bar{u}} & \text{Jacobian of } g \text{ wrt } x \\ D = \frac{\partial g(x(t),u(t))}{\delta u}|_{x=\bar{x},u=\bar{u}} & \text{Jacobian of } g \text{ wrt } u \end{cases}$$

▼ **Stability of equilibrium solutions of nonlinear dynamical systems**

An equilibrium point is **stable** if all the solutions which start near $\bar{x}$ (meaning that the initial condition lies in a neighbourhood of $\bar{x}$) remain near $\bar{x}$ for all time.

$\bar{x}$ is said **asymptotically stable** if it's stable and all the solutions starting near $\bar{x}$ tend towards $\bar{x}$ as $t \to \infty$.

To study the equilibrium stability we need to analyze if the time course of the perturbed solution $x_p(t)$:

- evolves inside a bounded neighbourhood of $\bar{x}$ (**stability**)

  Formal definition: The equilibrium state $\bar{x}$ is **stable** if
  $\forall \epsilon > 0, \exists \delta = \delta(\epsilon) > 0 : \forall x : ||x_0 - \bar{x}|| \leq \delta \implies ||x_p(t) - \bar{x}|| \leq \epsilon,\ \forall t \geq 0$
  Roughly speaking: it is stable if we can select a bound δ on initial condition that will result in trajectories that remain within a given arbitrarily small finite limit ϵ.

- asymptotically converges to $\bar{x}$ (**asymptotic stability**)

  Formal definition: The equilibrium state $\bar{x}$ is **asymptotically stable** if it's stable AND δ can be chosen such that: $\lim_{t\to\infty} ||x_p(t) - \bar{x}|| = 0$.
  Roughly speaking: it is asymptotically stable if it's stable and if the state approaches $\bar{x}$ as the time approaches ∞.

- evolves far away from $\bar{x}$ (**instability**)

  Formal definition: The equilibrium state $\bar{x}$ is **unstable** if the perturbed solution $x_p(t)$ does not satisfy the stability condition.

Special case: for LTI systems of the form $\dot{x}(t) = Ax(t) + Bu(t)$ we have the following:

$$\begin{cases} \text{internal stability} \Leftrightarrow \text{stability of every equilibrium solution } \bar{x} \\ \text{asymptotic stabilty} \Leftrightarrow \text{asymptotic stability of every equilibrium solution } \bar{x} \\ \text{unstability} \Leftrightarrow \text{unstability of every equilibrium solution } \bar{x} \end{cases}$$

▼ **Stability of an equilibrium solution through the linearized model**

Let $(\bar{x}, \bar{u})$ be an equilibrium point of the dynamical system $\dot{x}(t) = f(x(t), u(t))$ where $f(\cdot) : \mathcal{D} \to \mathcal{R}^n$ is a C^1 function and $\mathcal{D}$ is a neighbourhood of $\bar{x}$.

Let $A = \frac{\partial f(x)}{\partial x}|_{x=\bar{x}, u=\bar{u}}$, then:

- The equilibrium $\bar{x}$ is asymptotically stable if $\text{Re}[\lambda_i(A)] < 0, i = 0, ..., n$
- The equilibrium $\bar{x}$ is unstable if $\text{Re}[\lambda_i(A)] > 0$, for some i

Remember: if $\text{Re}[\lambda_i(A)] \leq 0, i = 1, ..., n$ the **no conclusion can be drawn** about the stability properties of the equilibrium $\bar{x}$.

Eigenvalues $\lambda_i(A)$ of the linearized system	Stability properties of the equilibrium
$\forall i : \mathbb{R}\text{e}(\lambda_i(A)) < 0$	**Asymptotic stability**
$\exists i : \mathbb{R}\text{e}(\lambda_i(A)) > 0$	**Unstability**
$\forall i : \mathbb{R}\text{e}(\lambda_i(A)) \leq 0$	**No conclusion can be drawn**

▼ **Discrete time LTI dynamical systems**

A discrete-time signal is made up by a sequence of real numbers; therefore, the signal is not defined between two sampling instants.

- Unit impulse sequence

$$\delta(k) = \begin{cases} 0 & k \neq 0 \\ 1 & k = 0 \end{cases}$$

- Unit step sequence

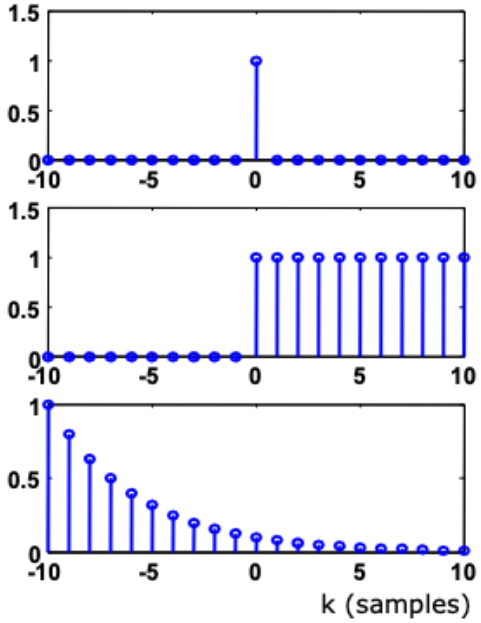
$$\varepsilon(k) = \begin{cases} 0 & k < 0 \\ 1 & k \geq 0 \end{cases}$$

- "Geometric" sequence

$$e^{aTk} = \alpha^k, |\alpha| < 1$$

$$\uparrow$$

$$e^{aT} = \alpha$$



They're used to describe:

- Phenomena whose events are defined in discrete time instants only
- The sampling of continuous time signals

A finite dimensional, discrete time ($k \in Z^n$), LTI dynamical system can be described through a state space representation made up by:

- A system of finite difference equations
- A static output equation

$x(k) \in R^n, u(k) \in R^p, y(k) \in R^q$

$$\begin{cases} x(k+1) = Ax(k) + Bu(k) & \text{state equation} \\ y(k) = Cx(k) + Du(k) & \text{output equation} \end{cases}'$$

$A \in R^{n,n}, B \in R^{n,p}, C \in R^{q,n}, D \in R^{q,p}$

▼ Solution of discrete time dynamical systems through $\mathcal{Z}$-transform

Consider the state space description

$\begin{cases} x(k+1) = Ax(k) + Bu(k) \\ y(k) = Cx(k) + Du(k) \end{cases}$, given $u(k), x(0)$ compute $x(k), y(k)\ \forall k \geq 0$

The solution can be computed using the time domain iterative relations:

$x(k) = A^k x(0) + \sum_{i=0}^{k-1} A^{k-i-1} B u(i) = x_{zi}(k) + x_{zs}(k)$
$y(k) = CA^k x(0) + C\sum_{i=0}^{k-1} A^{k-i-1} B u(i) + Du(k) = y_{zi}(k) + y_{zs}(k)$

Analysis and solution of the time dynamical system can be performed using the Z-transform operator.

I'll just put a little table of some known transformations:

$f(k)$	$F(z)$
a^k	$\frac{z}{z-a}$
$\binom{k}{\ell} a^{k-\ell}, \ell > 0$	$\frac{z}{(z-a)^{\ell+1}}$
$\sin(\vartheta k), \vartheta \in \mathbb{R}$	$\frac{z\sin(\vartheta)}{z^2 - 2\cos(\vartheta)z + 1}$
$\cos(\vartheta k), \vartheta \in \mathbb{R}$	$\frac{z(z-\cos(\vartheta))}{z^2 - 2\cos(\vartheta)z + 1}$
$A^k, A \in \mathbb{R}^{n,n}$	$z(zI - A)^{-1}$

▼ Example in MatLab

$x(k+1) = \begin{pmatrix} 3 & 0 \\ -3.5 & -0.5 \end{pmatrix} x(k) + \begin{pmatrix} 1 \\ -2 \end{pmatrix}$
$y(k) = \begin{pmatrix} 1 & -1 \end{pmatrix} x(k)$

Compute $x_{zi}(k), x(k)$ when: $u(k) = 2\epsilon(k), x(0) = \begin{pmatrix} 1 \\ -2 \end{pmatrix}$

```
A = [3 0; -3.5 -0.5];
B = [1; 2];
C = [1 -1];
```

```
x0 = [1; -2];
z = tf('z', 1);

U = 2 * z / (z - 1);

% Compute zero-input state response
X_i = minreal(zpk((z * inv(z * eye(2) - A)* x0)), 1e-3);

% Compute the residues of PFE, for z-domain use residuez instead
% of residue!!!
[num_X_i_1, den_X_i_1] = tfdata(X_i(1), 'v'); % 1st state
[r1, p1, k1] = residuez(num_X_i_1, den_X_i_1)

[num_X_i_2, den_X_i_2] = tfdata(X_i(2), 'v'); % 2nd state
[r2, p2, k2] = residuez(num_X_i_2, den_X_i_2)

% Compute the zero state response
X_f = minreal(zpk((inv(z * eye(2) - A) * B) * U), 1e-3)

% Compute the residues of PFE, for z-domain use residuez instead
% of residue!!!
[num_X_f_1, den_X_f_1] = tfdata(X_f(1), 'v'); % 1st state
[r1, p1, k1] = residuez(num_X_f_1, den_X_f_1)

[num_X_f_2, den_X_f_2] = tfdata(X_f(2), 'v'); % 2nd state
[r2, p2, k2] = residuez(num_X_f_2, den_X_f_2)
```

▼ Transfer function of a discrete time LTI system

The transfer function of a discrete time LTI system is expressed as a real rational function of the form: $H(z) = \frac{N_H(z)}{D_H(z)}, m \leq n$:

- $m < n$ → **strictly proper**
- $m = n$ → **proper**

Hint: in MatLab, use the function $\mathrm{zpkdata}$ to compute **zeros**, **poles, gain** of a zero-pole-gain form for a given transfer function.

Consider a discrete-time LTI system described by a difference equation
$y(k) = -a_1 y(k-1) - a_0 y(k-2) + b_1 u(k) + b_0 u(k-1).$

Applying the Z-transform we obtain
$Y(z) = (-a_1 z^{-1} - a_0 z^{-2})Y(z) + (b_1 + b_0 z^{-1})U(z)$

The corresponding transfer function is $H(z) = \frac{Y(z)}{U(z)} = \frac{b_1 + b_0 z^{-1}}{1 + a_1 z^{-1} + a_0 z^{-2}}$

▼ Example in MatLab

Given the following tf of an LTI dynamical system: $H(z) = \frac{z}{z^2+0.5z+0.25}$, compute the output response $y(k)$ when $u(k) = \delta(k)$.

```
z = tf('z', 1);

H = z / (z^2 + 0.5 * z + 0.25);

U = 1;

Y = minreal(zpk(H * U), 1e-3)
[num, den] = tfdata(Y, 'v');
[r, p, k] = residuez(num, den)

nu = abs(p(1))
theta = angle(p(1))
M = abs(r(1)), M2 = 2 * M
phi = angle(r(1))
```

Then we can compute $y(k) = 2M\nu^k \cos(\theta k + \phi)$

▼ **Stability of discrete time dynamical systems**

Stability deals with the following problems:

- Is the zero input ($u(k) = 0$) state response $x_{zi}(k)$ bounded for any initial state $x_0 \in \mathcal{R}^n$? → **internal stability**
- Is the zero state ($x_0 = 0$) system output response $y(k)$ bounded for any bounded input $u(k)$? → **BIBO stability**

▼ Internal stability of LTI systems

- An LTI system is **internally stable** if the zero input state response $x_{zi}(t)$ is **bounded** for any initial state x_0.
- An LTI system is **asymptotically stable** if the zero input response $x_{zi}(t)$ converges to 0, as $k \to \infty$, for any initial state x_0.
- An LTI system is **unstable** if it's not stable.

▼ Natural modes of LTI systems

The zero input state response is a linear combination of the natural modes of the system.

A natural mode $m(k)$ is said:

- **Convergent** if $\lim_{k\to\infty} |m(k)| = 0$

- **Bounded** if $\exists M \in \mathcal{R} : \forall k \geq 0 \rightarrow 0 \leq |m(k)| < \infty$
- **Divergent** if $\lim_{k \to \infty} |m(k)| = \infty$

Denote with $\lambda_i(A), i = 1, ..., n$ the i^{th} eigenvalue of matrix A, then the natural mode associated to the eigenvalue λ_i is:

- **Bounded** if $|\lambda_i(A)| = 1$ **and** $\mu'(\lambda_i(A)) = 1$
- **Convergent** if $|\lambda_i(A)| < 1$
- **Divergent** if $|\lambda_i(A)| > 1$ **or** $|\lambda_i(A)| = 1$ **and** $\mu'(\lambda_i(A)) > 1$

We can now redefine the stability characteristics of an LTI system.

> An LTI system is **internally stable** if and only if all the system's eigenvalues have magnitude ≤ 1 and those with magnitude $= 1$ have $\mu'(\lambda_i) = 1$ (the natural modes are **bounded**).

> An LTI system is **asymptotically stable** if and only if all the system's eigenvalues have magnitude < 1 (the natural modes are **convergent**).

> An LTI system is **unstable** if and only if either there is an eigenvalue with magnitude > 1 or all the systems eigenvalues have magnitude ≤ 1 and those with magnitude $= 1$ have $\mu'(\lambda_i) > 1$.

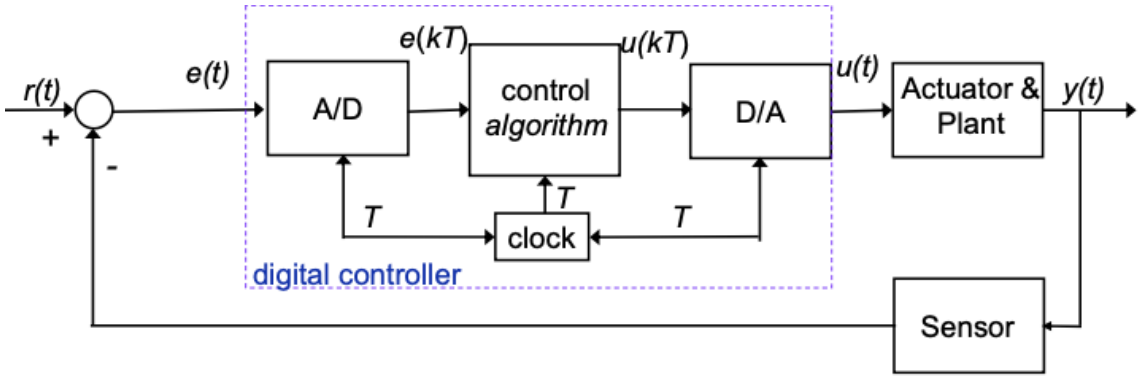
▼ BIBO stability of LTI systems

A Single-Input Single-Output LTI system is BIBO stable if the zero state output response is bounded for all bounded inputs:
$\forall u_M \in (0, \infty), \exists y_M \in (0, \infty) : |u(k)| \leq u_M, \forall k \geq 0 \implies |y(k)| \leq y_M, \forall k \geq 0$

> A discrete-time LTI system is **BIBO stable** if and only if all the poles of the transfer function $H(z)$ lie strictly inside the unit circle: $|p_i| < 1, i = 1, ..., n$

▼ Sampled data systems

▼ The structure of a sampled data systems



In order to study sampled data systems, we need to describe:

- Signal sampling (A/D conversion)
- Data reconstruction (D/A conversion)
- Digital control algorithm

▼ Signal sampling (A/D conversion)

We can describe the sampling systems as a switch that closes at every **sampling time** T and generates the sampled signal $f(kT)$, in Math this means multiplying the signal to be sampled $f(t)$ by a train of impulses $\delta_T(t)$; we represent the result as
$f^*(t) = \sum_{k=0}^{\infty} f(kT)\delta(t-kT)$.

We can also represent $f^*(t)$ through the Fourier series expansion of the train of impulses: $f^*(t) = f(t)\sum_{k=-\infty}^{\infty} C_k e^{jk\omega_s}$, where:

- $C_k = \frac{1}{T}\int_{-\frac{T}{2}}^{\frac{T}{2}} \sum_{k=-\infty}^{\infty} \delta(t-kT)e^{jk\omega_s}dt = \frac{1}{T}$
- $\omega_s = \frac{2\pi}{T}$

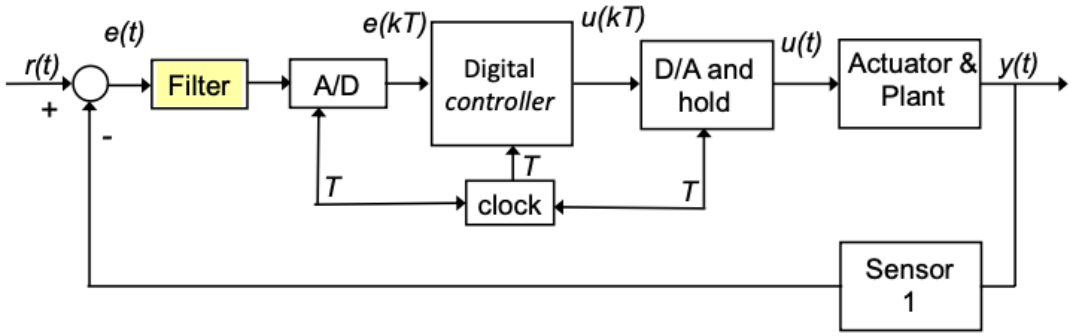
Note: The Laplace transform of $f^*(t)$ is: $\mathcal{L}[f^*(t)] = \frac{1}{T}\sum_{k=-\infty}^{\infty} F(s-jk\omega_s)$

▼ Aliasing and Shannon's theorem

Since it'd be long and difficult to insert all the theorem by hand, please refer to the PDF *05 - Sampled data systems*, from page 12 to page 27, or to the following PDF:

https://s3-us-west-2.amazonaws.com/secure.notion-static.com/77fae281-65fb-431b-9284-a550b4d7b6c3/05-_Sampled_Data_Systems.pdf

To sum up, by the way, we can describe the structure of the sampled data feedback control system with anti-aliasing filter with the following scheme:



▼ Data reconstruction (D/A conversion)

Ideally, we could reconstruct the original signal $f(t)$ with a low-pass filter.

However, since the ideal filters cannot be obtained through physically realizable systems, they can be approximated using data hold devices.

The simplest and most widely used is the **zero-order hold (ZOH) filter**.

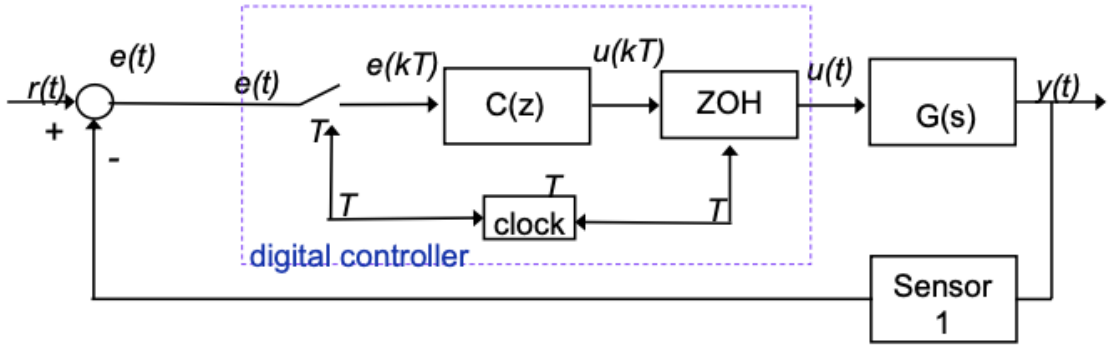
The ZOH clamps the output to a value equal to the input at the current sampling instant and keeps it until the next sampling.

The ZOH transfer function is given by $G_{ZOH}(s) = \frac{1-e^{-Ts}}{s}$ and it approximates an ideal low pass filter.

By analyzing its frequency response, we get that the sampling time T should be so that:
$T \leq \frac{\pi}{\omega_0}$.

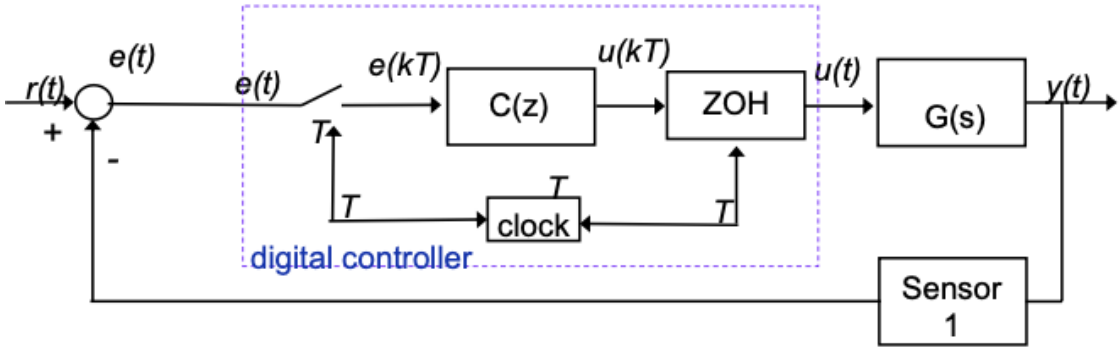
▼ Digital controller

The digital controller is realized through a proper discrete time transfer function in the $\mathcal{Z}$-transform domain.



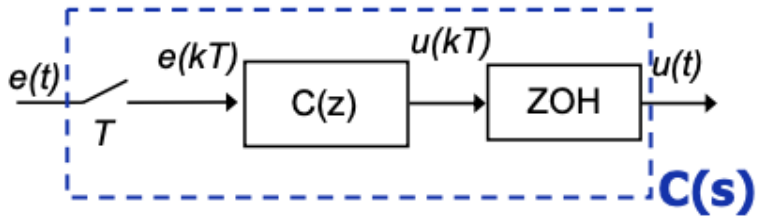
▼ Resume

Consider the following data control system structure:



To analyze its performance, there are different approaches:

1. Convert the system $G(s)$ to its discrete-time equivalent $G(z)$ ignoring its inter-sampling behaviour → discrete-time analysis.
2. Model the digital controller in continuous time → continuous time analysis
   In order to do it, we need to define:
   a. A/D conversion: $E^*(e^{j\omega T}) = \frac{1}{T}\sum_{h=-\infty}^{\infty} E(j(\omega + h\omega_s))$
   b. Digital controller: $C(z) = C(e^{j\omega T})E^*(e^{j\omega T})$
   c. D/A conversion: $U(j\omega) = G_{ZOH}(j\omega)U^*(e^{j\omega T})$

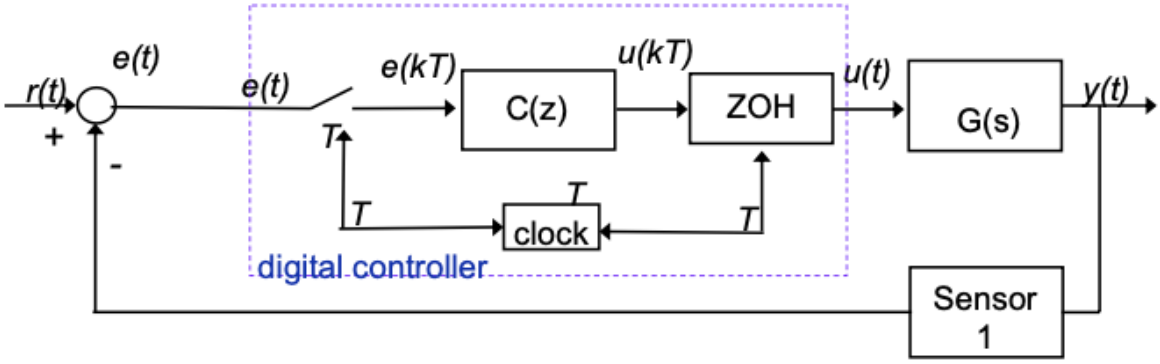


d. At the end, we get: $U(j\omega) = G_{ZOH}(j\omega)C(e^{j\omega T})\frac{1}{T}E_S(j\omega)$, and so the digital controller is equivalent to an analog controller of the form: $C(s) = G_{ZOH}(s)C(e^{sT})\frac{1}{T} \rightarrow G_{A/D}(s) = \frac{1}{t}$

3. Use numerical simulation (Simulink) → it only provides an answer for the specific case considered in the simulation.

▼ **Digital control design**

We'll consider the problem of designing the controller $C(z)$ for the sampled data control system structure below:



A suitable procedure for the designing of the digital tf $C(z)$ consists in:

1. Looking for an analog controller $C_0(s)$ such that the loop function frequency response
   $L_0(j\omega) = C(j\omega)G(j\omega) \approx e^{-\frac{j\omega T}{2}}C_0(j\omega)G(j\omega), \omega \in [0, \omega_N]$ satisfies the control requirements.
2. Computing a discrete equivalent $C(z)$ of $C_0(s)$ so that
   $C(e^{j\omega T}) \approx C_0(j\omega), \omega \in [0, \omega_N]$

This design procedure is referred as **emulation**.

▼ Emulation

The steps to follow are:

1. Choose a suitable sampling period T
2. Design an analog controller $C_0(s)$ using any method (e.g. frequency response loop-shaping)
3. Obtain through a suitable discretization procedure the digital controller $C(z)$
4. Verify through simulation time domain performance
5. If needed, design anti-aliasing filter
6. If needed, perform modifications of $C_0(s)$ in order to meet the requirements

▼ Sampling time selection

Several aspects have to be taken into account:

- **Sampling theorem**
  - The sampling frequency must be at least **twice** the value of the highest significant frequency in the signal.
    In a feedback system the highest frequency of all the signals in the loop is the system bandwidth ω_B, therefore a lower bound for the sampling frequency ω_s is $\omega_s > 2\omega_B \rightarrow \omega_s > 3\omega_c$
- **ZOH filter**
  - The D/A converter introduces a phase lag of $\arg(G_{ZOH}(j\omega)) = -\frac{\omega T}{2}$.
    In order to limit the phase lag at the cross-over frequency to small values, the following bound should be considered:
    $\arg(G_{ZOH}(j\omega)) = -\frac{\omega T}{2} = -\frac{\omega \pi}{\omega_s}$
    $-10^\circ < -\frac{\omega \pi}{\omega_s} < -5^\circ \implies 0.087\text{rad} < \frac{\omega \pi}{\omega_s} < 0.17\text{rad} \implies$
    $18\omega_c < \omega_s < 36\omega_c$
- **Suitable sampling of the transient**
  - A suitable number of samples must be considered for describing the behaviour of the transient phase.
    Typically 10 to 50 samples can be employed for a suitable description of the transient behaviour.
    This leads to the condition: $11.42\omega_c < \omega_s < 57.2\omega_c$
- **Limitations → small T**
  - improve conversion accuracy and destabilizing effects

- increase the cost of A/D and D/A devices
- emphasizes quantization effects

We need to trade among these criteria to choose T.

A final practical rule of thumb is to choose T within the interval:
$20\omega_c < \omega_s < 50\omega_c \implies \frac{0.12}{\omega_c} < T < \frac{0.3}{\omega_c}$

▼ **Analog controller design**

We can use any method to design $C_0(s)$, but if it's known a priori that the controller has to be realized through a digital computer, the design of $C_0(s)$ is performed taking into account the dynamics introduced by the A/D and the ZOH D/A transfer function $G_{A/D}(s) = \frac{1}{T}$, $G_{ZOH}(s) = \frac{1-e^{-Ts}}{s}$.

Thus the analog controller design should be designed considering the plant transfer function $G'(s) = \frac{1}{T}G(s)\frac{1-e^{-Ts}}{s}$.

Remark: in Matlab we should use the 1st Padé approximation:
$G_{ZOH}(s) \approx \frac{T}{1+s\frac{T}{2}} \implies G'(s) = G(s)\frac{1}{1+s\frac{T}{2}}$

▼ **Discretization**

Again, the discretization chapter includes a lot of (useless) theory, I'll insert the PDF pages below or you can find them into *06 - Digital Control Design*, from page 14 to page 29.

https://s3-us-west-2.amazonaws.com/secure.notion-static.com/dc962249-45b4-403b-b30b-12cf83173b67/06_-_Digital_Control_Design.pdf

By the way, I'll include the useful part of this section: the Matlab code to perform the discretization of a transfer function:

```
C = c2d(C_0, METHOD)
```

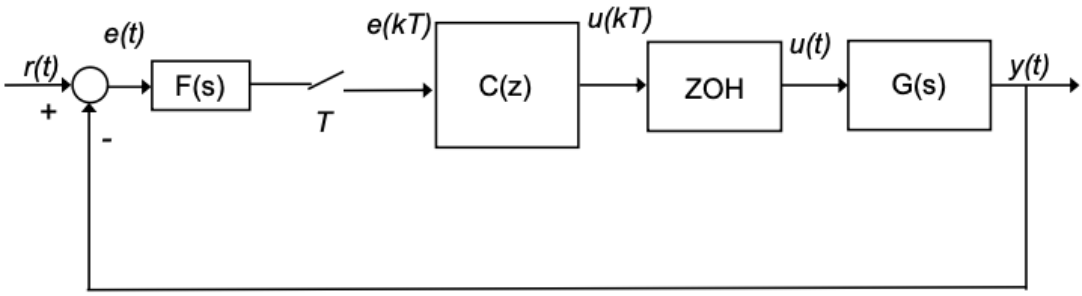
Where METHOD can be one of the following:

'zoh'	Zero-order hold
'foh'	First-order hold
'imp'	Impulsive-invariant
'tustin'	Tustin
'prewarp'	Tustin approximation with frequency prewarping
'matched'	Matched pole-zero

The default is 'zoh' when the parameter is omitted.

▼ Simulation

Verify through Simulink time domain performance and requirements.



Note: in the Simulink scheme, blocks corresponding to A/D and D/A may be omitted.

▼ Final example

A basic example of the design process can be found in the following PDF file:

https://s3-us-west-2.amazonaws.com/secure.notion-static.com/8f09bac8-17c7-4d31-b0e5-7d3d41c93212/06_-_Digital_Control_Design.pdf

https://s3-us-west-2.amazonaws.com/secure.notion-static.com/bcc3bee9-932e-4359-b7e4-3922650ee44e/06_-_Digital_Control_Design.pdf

▼ **Control systems design via static state feedback**

Let's consider an LTI **continuous** time system described by the following ss representation:

$$\begin{cases} \dot{x}(t) = Ax(t) + Bu(t) \\ y(t) = Cx(t) \end{cases}, x \in \mathcal{R}^n, u, y \in \mathcal{R}^p$$

The dynamics of the LTI system depend on the eigenvalues of matrix A, so we need to find an input that can affect the structure of A.

We can consider the input $u(t) = -Kx(t) + Nr(t), K \in \mathcal{R}^{p,n}, r(t) \in \mathcal{R}^p, N \in \mathcal{R}^{p,p}$

In the presence of it, the state equation becomes $\dot{x}(t) = Ax(t) + Bu(t) = (A - BK)x(t) + BNr(t)$ and the dynamical properties of the controlled system depend on the eigenvalues of the matrix $A - BK$.

The matrix K is referred to as the **state gain**.

A suitable choice of K allows us to modify the system eigenvalues and improve the dynamic properties of the system (stabilize it).

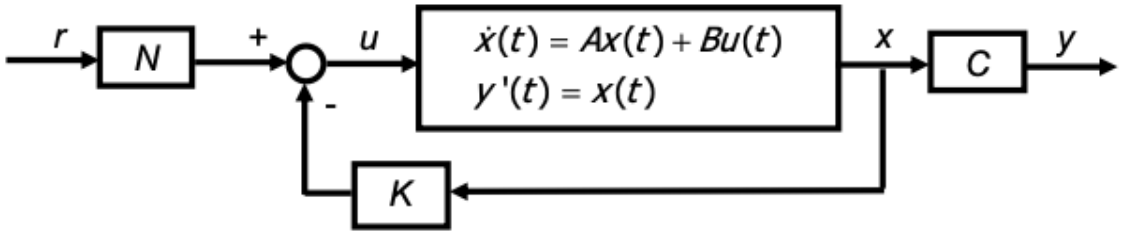
Note: A matrix $K \in \mathcal{R}^{p,n}$ exists such that the n eigenvalues of $A - BK$ can be assigned to arbitrary, real or complex-conjugate, locations if and only if $\rho(M_R) = \rho([B\ AB\ ...\ A^{n-1}B]) = n$

The matrix $M_R = [B\ AB\ ...\ A^{n-1}B]$ is the **reachability matrix**.

If $\rho(M_R) = n$ the dynamic system $\dot{x}(t) = Ax(t) + Bu(t)$ is said **reachable**.

```
Mr = ctrb(A, B);
rho_Mr = rank(Mr)
```

The input $u = -Kx + Nr$ represents a **static state feedback** control law



- Typically, K is computed by placing the eigenvalues (poles) of the controlled system to obtain, besides asymptotic stability, good damping coefficient and rapidity properties of the transient.

```
lambda_des = [-1 -2];
K = place(A, B, lambda_des)

% OR
lambda_des = [-1 -1];
K = acker(A, B, lambda_des)
```

- N can be choosen to make unitary the dc-gain of the controlled system guaranteeing zero steady state tracking error in the presence of a constant reference signal
  $r(t) = \bar{r}\epsilon(t)$.

```
sys_cont_c = ss(A-BK, B, C, 0);
N = 1 / dcgain(sys_cont_c)
```

▼ Requirements analysis

It is possible to choose the eigenvalues directly from the requirements, converting them to ζ and ω_n requirements.

Then we can obtain the eigenvalues to be assigned by the state feedback controller: we convert ζ and ω_n to the corresponding couple of complex conjugate values:

$\lambda_{1,2} = \sigma_0 \pm j\omega_0 = -\zeta\omega_n \pm j\omega_n\sqrt{1-\zeta^2}$

Another option is to design the control law avoiding oscillations of the output step response.

In this case, a prototype 2^{nd}order model with coincident real poles can be used to fix the eigenvalues to assign.

$u = -Kx + Nr$

K is designed to assign $n = \dim(x)$ eigenvalues.

if $n > 2$, the additional eigenvalues are chosen with a faster time constant τ_{add} wrt the ones corresponding to the prototype second order system eigenvalues.

Note: $\tau_{add} << \frac{1}{\zeta\omega_n}$, in particular, as a rule of thumb: $\tau_{add} = (0.1 \dot{-} 0.2)\frac{1}{\zeta\omega_n}$

▼ Complete Matlab example

Consider the following problem:

- Given the LTI system

$$\begin{cases} \dot{x}(t) = \begin{bmatrix} -0.1 & -1 \\ 1 & 0 \end{bmatrix} x(t) + \begin{bmatrix} 1 \\ 0 \end{bmatrix} u(t) \\ y(t) = \begin{bmatrix} 0 & 1 \end{bmatrix} x(t) \end{cases}$$

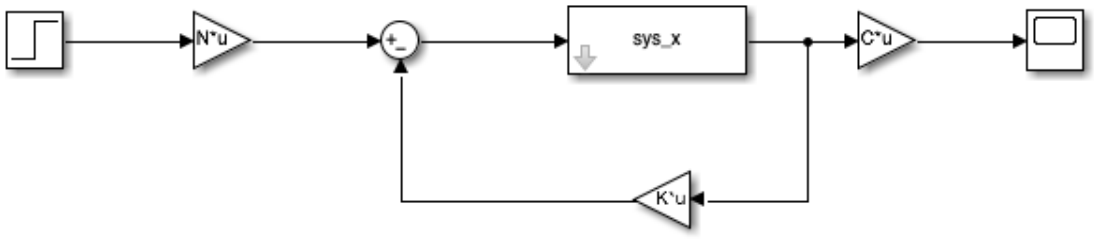
- Design a state feedback controller of the form

$$u(t) = -Kx(t) + Nr(t)$$

to meet the following requirements:

1. $| e_r^{\infty} | = 0$ for $r(t) = \varepsilon(t)$
2. $\hat{s} \leq 10\%$
3. $t_{s,2\%} \leq 6.5$ s

Firstly, let's design the Simulink scheme:

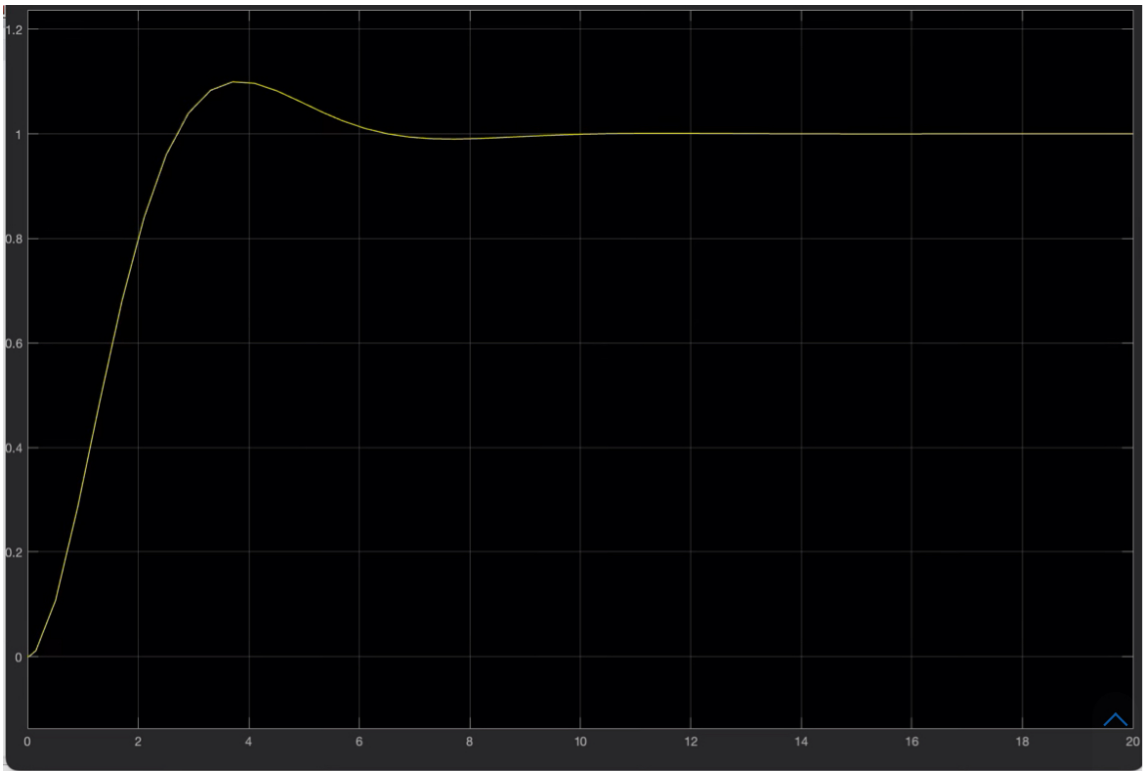


Then we can implement the Matlab code to run it:

```
% Define the state matrices
A = [-0.1 -1; 1 0];
B = [1; 0];
C = [0 1];
D = 0;
x0 = [0; 0];

% The following is needed to represent the system in Simulink
sys_x = ss(A, B, eye(2), 0);

% Check the reachability of the system
```

```
M_r = ctrb(A, B) % Reachability matrix
rho_M = rank(M_r) % Rank of the reachability matrix

% Since rho_M is > 0 (in particular it is 2), we can proceed because the
% system is reachable

% Define the requirements
s_hat = 0.1; % Max overshoot
t_s2 = 6.5; % Settling time 2%
zeta = abs(log(s_hat)) / (sqrt(pi^2 + (log(s_hat))^2)) % Dampling coefficient
wn = log(0.02^(-1)) / (zeta * t_s2)

% Define eigenvalues to assign
lambda_1 = -zeta * wn + j * wn * sqrt(1 - zeta^2)
lambda_2 = -zeta * wn - j * wn * sqrt(1 - zeta^2)

lambda_des = [lambda_1 lambda_2]; % Vector of the desired eigenvalues

% Compute K
K = place(A, B, lambda_des)

% State matrices of the controlled system
A_c = A - B * K;
B_c = B;
C_c = C;
D_c = 0;

sys_c = ss(A_c, B_c, C_c, D_c);

% Compute N
N = 1/dcgain(sys_c)

% Simulation parameters
rho = 1;
t_sim = 20;

% Run the simulation
sim('state_feedback.slx')
```

Simulation results:



From the simulation results we see that:

- $\hat{s} \approx 10\%$, requirement satisfied!
- $t_{s,2\%} \approx 5.8s$, requirement satisfied!

▼ **Asymptotic state observers**

▼ State observers

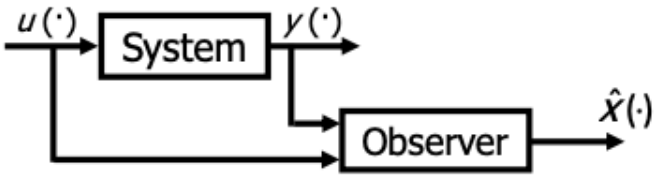
Let's consider an LTI continuous time system described by the state space representation: $\begin{cases} \dot{x}(t) = Ax(t) + Bu(t) \\ y(t) = Cx(t) \end{cases}$

A state feedback control law of the form $u = -Kx + Nr$ can be evaluated only when the system state can measured.

However in general, only the measurement of the system output is available.

Under suitable assumptions, an estimate of the system state can be obtained based on the knowledge of the provided input and the measurement of the output signal.

A **state observer** is a device that provides an estimate $\hat{x}$ of the system state x exploiting the knowledge of the system input u and the measurement of the output y.



For a continuous time dynamic system, the state estimation error e can be defined as $e(t) = \hat{x}(t) - x(t)$.

An observer such that $\lim_{t\to\infty} |e(t)| = \lim_{t\to\infty} |\hat{x}(t) - x(t)| = 0$ is referred to as **asymptotic state observer**.

To obtain an asymptotic observer, a term depending on the output estimation error $\hat{y} - y$ is included: $\dot{e}(t) = (A - LC)e(t)$.

A suitable choice of L allows us to modify the observer eigenvalues and improve its dynamic properties.

However L exists if and only if $\rho(M_O) = \rho(\begin{pmatrix} C \\ CA \\ \dots \\ CA^{n-1} \end{pmatrix}) = n$

The matrix $\begin{pmatrix} C \\ CA \\ \dots \\ CA^{n-1} \end{pmatrix}$ is the **observability matrix.**

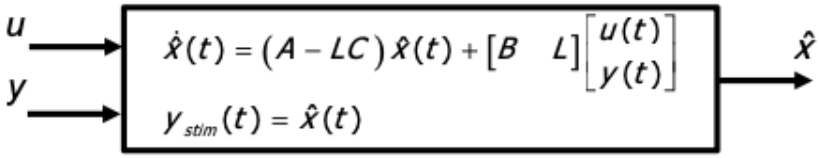
If $\rho(M_O) = n$ the dynamic system is said **observable**.

```
Mo = obsv(A, C);
rho_Mo = rank(Mo)
```

The state equation of the state observer is $\dot{\hat{x}} = (A - LC)\hat{x}(t) + Bu(t) + Ly(t)$.

The matrix L is referred to as the **observer gain** $\dim(L) = n \times q$.

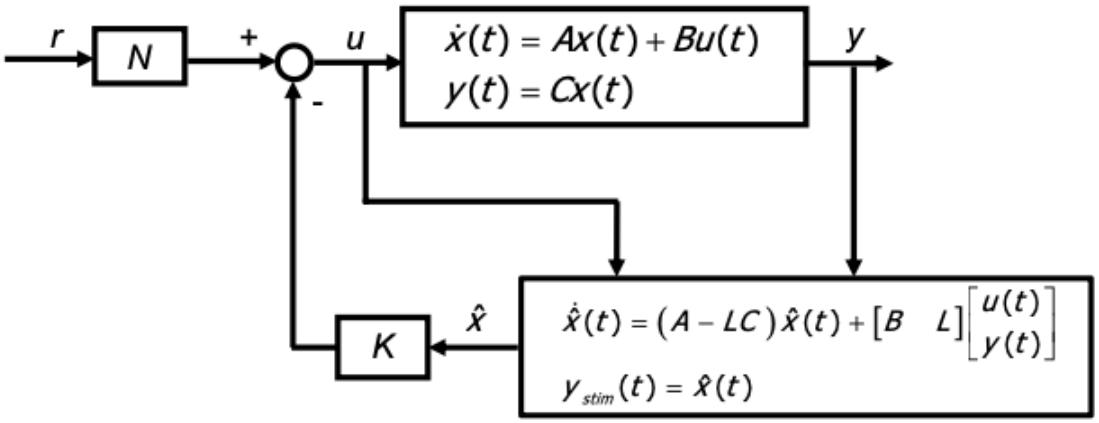
The observer is thus a dynamical system with inputs $u(t)$ and $y(t)$ and output $\hat{x}(t)$:



The matrix L can be computed through the Matlab place and acker statements:

```
lambda_obsv_des = [-1 -2];
L = place(A', C', lambda_obsv_des)'

% OR
lambda_obsv_des = [-1 -1];
L = acker(A', C', lambda_obsv_des)'
```

The architecture of the feedback control system is modified as:



To describe the overall system dynamics, we need to compute complete state space representation of the controlled system.

The ingredients are:

- $\dot{x}(t) = Ax(t) + Bu(t)$
- $y(t) = Cx(t)$
- $\hat{x}(t) = (A - LC)\hat{x}(t) + Bu(t) + Ly(t)$
- $u(t) = -K\hat{x}(t) + Nr(t)$

The design procedure can be resumed with the steps below:

- Check the reachability and observability properties
- Compute the state feedback gain K to assign the eigenvalues of $A - BK$ at the desired locations
- Compute the dc-gain correction matrix N

- Compute the observer gain L to assign the eigenvalues of $A - LC$ at the desired locations. Typically the observer eigenvalues are chosen such that the corresponding time constants are faster wrt the ones imposed by the control law, for instance $\tau_{A-LC} << \tau_{A-BK}$

▼ Complete Matlab example

Consider the following problem:

- Given the LTI system

$$\begin{cases} \dot{x}(t) = \begin{bmatrix} -0.4 & -1 \\ 1 & 0 \end{bmatrix} x(t) + \begin{bmatrix} 1 \\ 0 \end{bmatrix} u(t) \\ y(t) = \begin{bmatrix} 0 & 1 \end{bmatrix} x(t) \end{cases}$$

- Design a state feedback controller of the form

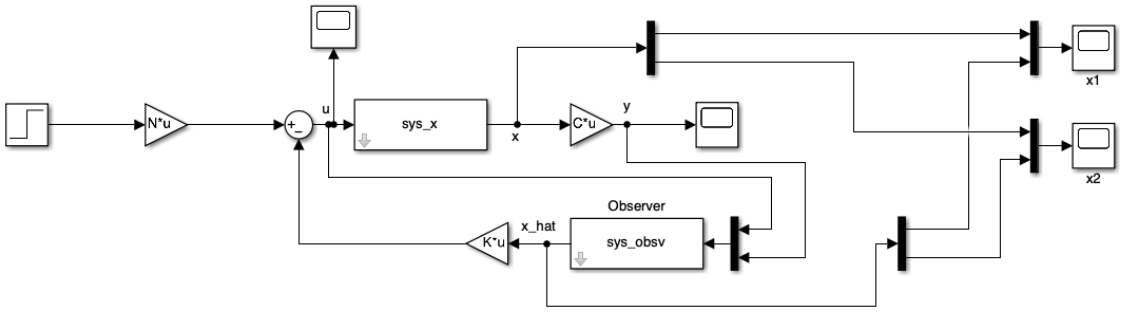
$$u(t) = -K\hat{x}(t) + Nr(t)$$

to meet the following requirements

1. $| e_r^{\infty} | = 0$ for $r(t) = \varepsilon(t)$
2. $\hat{s} \leq 8\%$
3. $t_{s,1\%} \leq 4$ s

$$x(0) = \begin{bmatrix} -0.5 & 0 \end{bmatrix}^T,$$
$$\hat{x}(0) = \begin{bmatrix} 0 & 0 \end{bmatrix}^T$$

Firstly, let's implement the Simulink scheme:



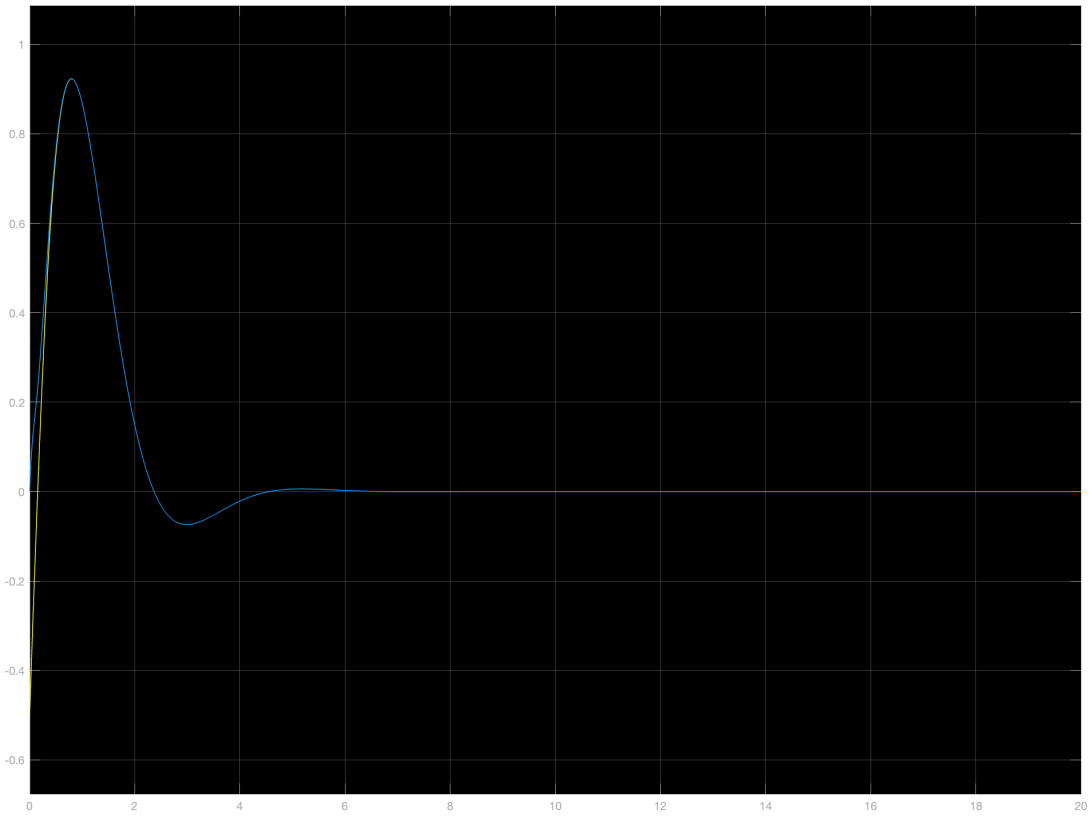
Then, let’s write the Matlab code to compute the parameters and run the simulation:

```
clear all
close all
clc

% Define the matrices
A = [-0.4 -1; 1 0];
B = [1; 0];
C = [0 1];
D = 0;
x0 = [-0.5; 0];

sys_x = ss(A, B, eye(2), 0);

% Check the reachability and the observability
M_r = ctrb(A, B)
rho_M_r = rank(M_r)
M_o = obsv(A, C)
rho_M_o = rank(M_o)

% State feedback design
 % Requirements
s_hat = 0.08;
t_s_1 = 4;
zeta = abs(log(s_hat))/(sqrt(pi^2+(log(s_hat)^2)))
wn = log((1/100)^(-1)) / (zeta * t_s_1)
lambda_1 = -zeta * wn + j * wn * sqrt(1-zeta^2)
lambda_2 = -zeta * wn - j * wn * sqrt(1-zeta^2)

lambda_des = [lambda_1 lambda_2];

K = place(A, B, lambda_des)

sys_c = ss(A - B * K, B, C, 0);

N = 1/dcgain(sys_c)

% Define the observer's eigenvalues to assign
lambda_obsv_des = [-zeta * wn -zeta * wn] * 10; % Vector of the desired
%                                               eigenvalues, 10 times faster

% Compute L
L = acker(A', C', lambda_obsv_des)'

% Define the observer system
sys_obsv = ss(A - L * C, [B L], eye(2), 0);
x0_hat = [0; 0]; % Estimated initial state

% Simulation parameters
rho = 1;
t_sim = 20;
```

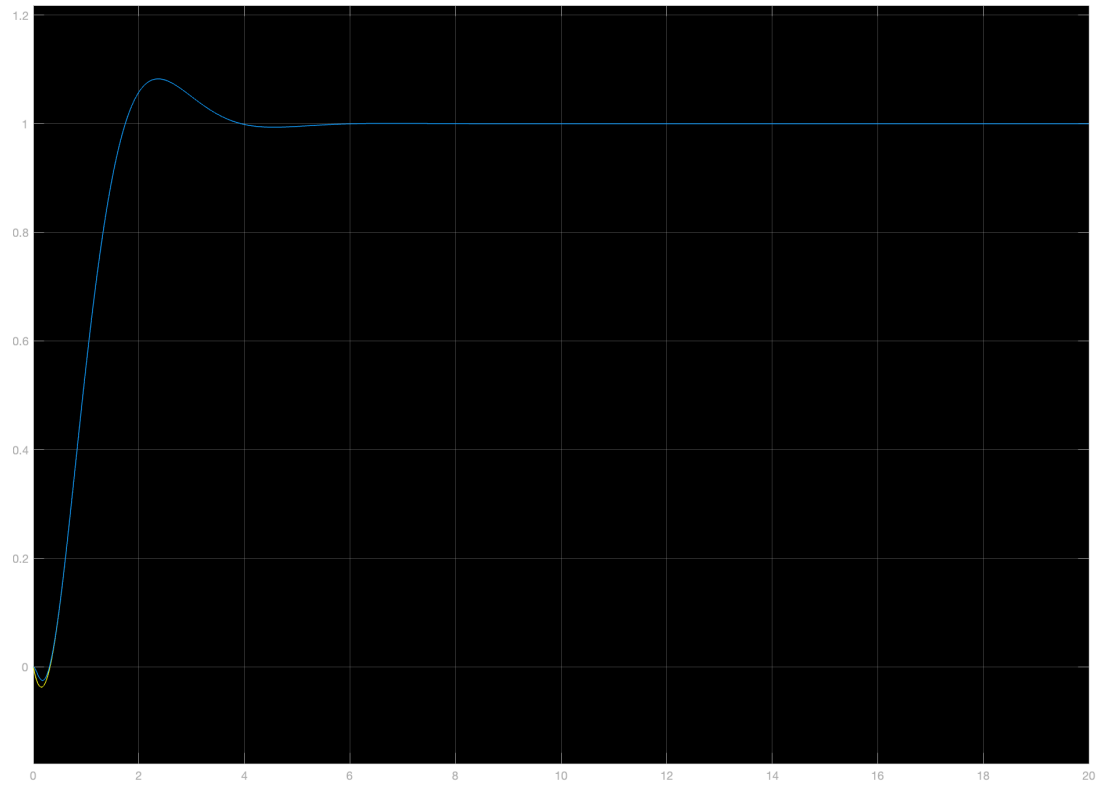
```
% Run the simulation
sim('estimated_states.slx')
```

The results of the simulation are:

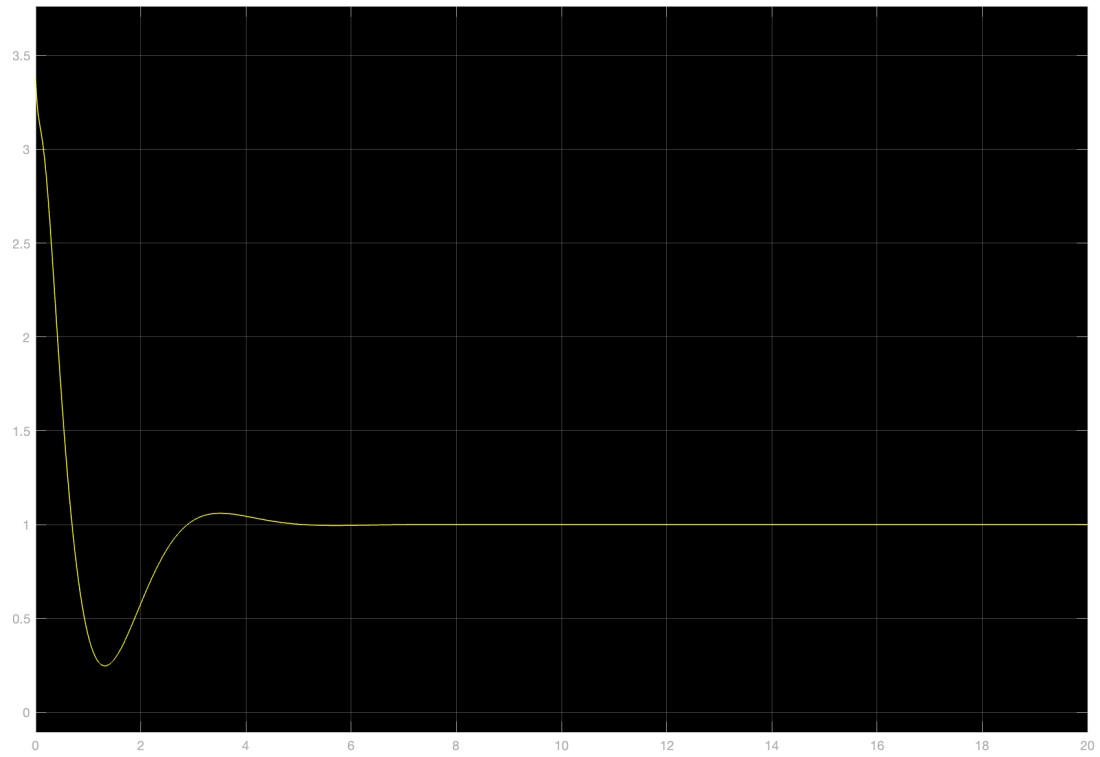
- x_1



- x_2


1.2
1
0.8
0.6
0.4
0.2
0
0
2
4
6
8
10
12
14
16
18
20

- $u(t)$


3.5
3
2.5
2
1.5
1
0.5
0
0
2
4
6
8
10
12
14
16
18
20

From the simulation results we can check the requirements:

- $\hat{s} \approx 0$, requirement satisfied!
- $t_{s,1\%} = 3.75s$, requirement satisfied!