Apoyo en el desarrollo del Proyecto de Grado MIACON

MÓDULO 2: Identificación y control de una planta de primer orden

- **Hecho por:** Diego Andrés García Díaz.
- **Código:** 2195533.

- **Fecha:** 26/09/2025.
- **Asignatura:** Control II (Adelanto de Nota).

Módelo para un Sitema Térmico (Primer Orden)

$$G_P(s) = \frac{K_p}{T_{\mathrm{p1}} \cdot s + 1} \cdot e^{-T_d \cdot s};$$

$$G_P(s) = \text{Actuador} + \text{Planta} + \text{Sensor};$$

$$K = K_p = \text{Ganancia};$$

$$K = \frac{\Delta Y \text{ (salida)}}{\Delta U \text{ (entrada)}};$$

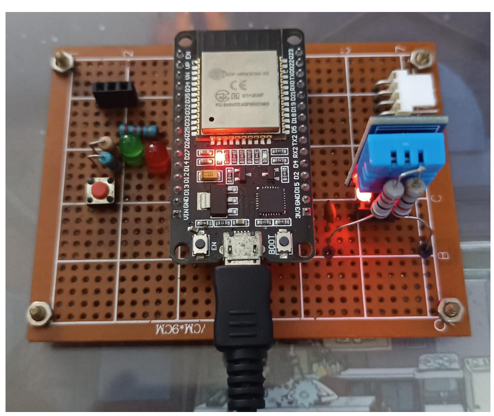
$$T = T_{\mathrm{p1}} = \text{Constante de tiempo};$$

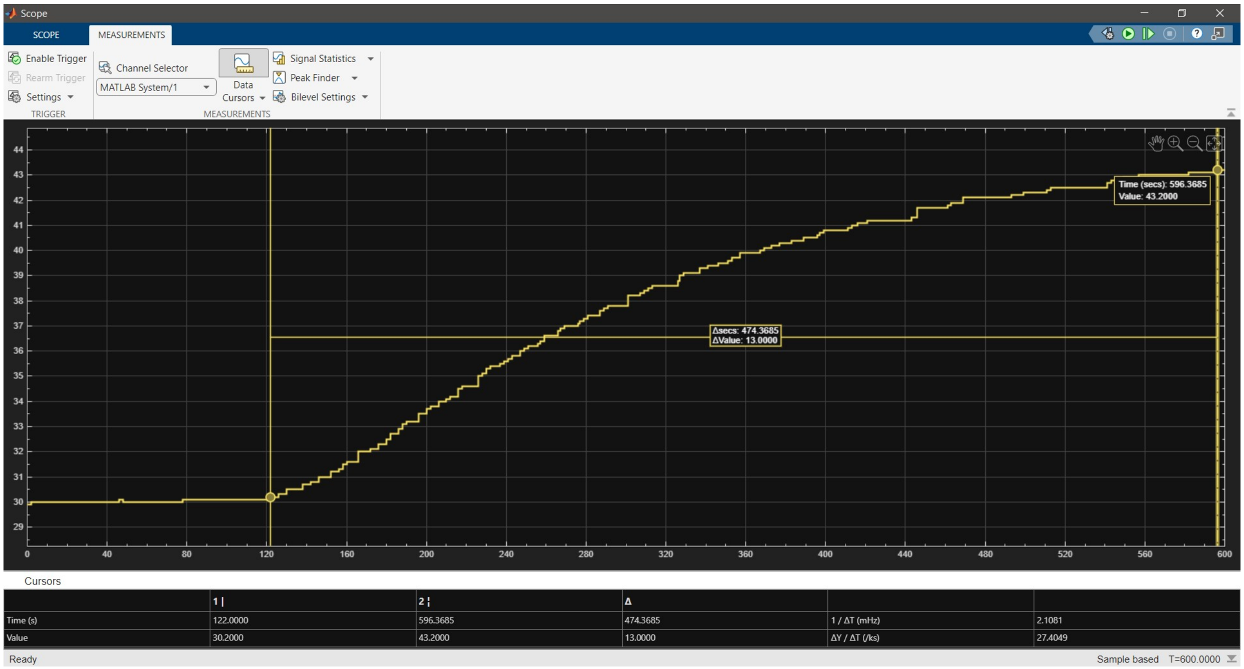
$$T = \frac{3}{2}(t_2 - t_1);$$

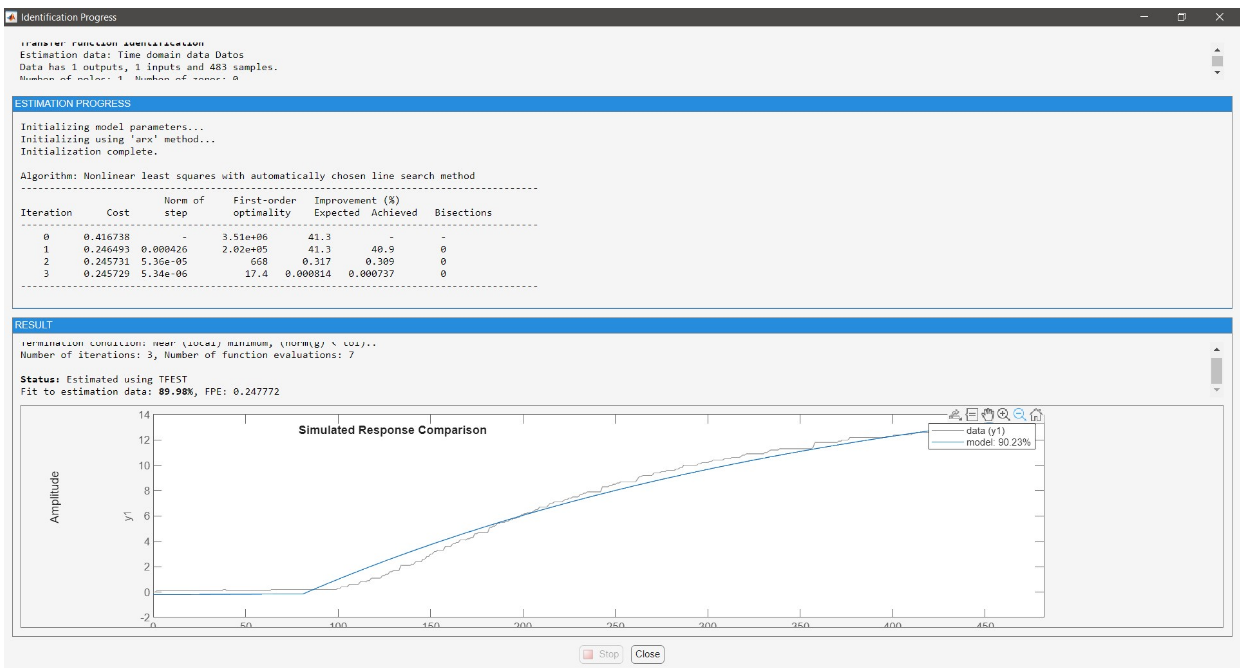
$$L = T_d = \text{Tiempo muerto (Delay)};$$

$$L = t_2 - T$$

Modelo identificado siguiendo las instrucciones de la página web:



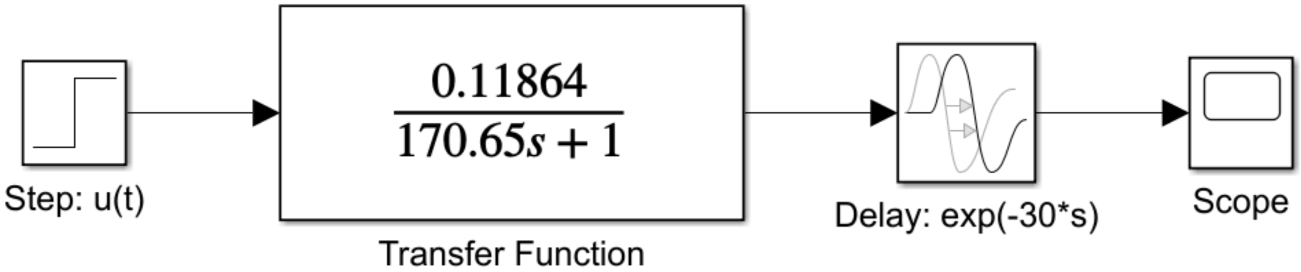


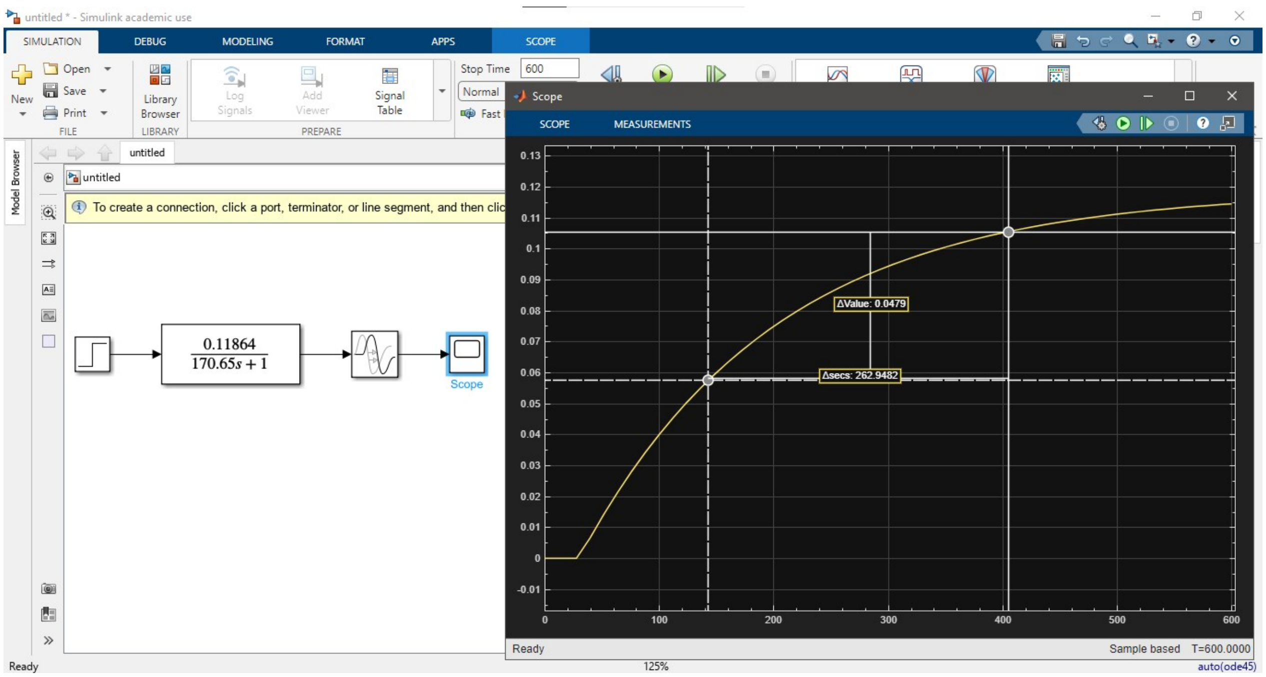


$$\mathrm{P1D} \quad \rightarrow \quad G_{P_1}(s) = P(s) = \frac{0.11864}{170.65 \cdot s + 1} \cdot e^{-30 \cdot s}$$

Ajuste de 90.23% > 80%

Validación en Simulink:





$$G_P(s) = \frac{0.1144}{158.7423 \cdot s + 1} \cdot e^{-31.7485 \cdot s}$$

$$t_1 = L + \frac{1}{3} \cdot T = 84.6626\,[s];\; t_2 = L + T = 190.4908\,[s]\;;$$

$$\text{Ganancia} \;\rightarrow\; K = \frac{\Delta Y\ (\text{salida})}{\Delta U\ (\text{entrada})} = 0.1144;$$

$$\text{Constante de tiempo} \;\rightarrow\; T = \frac{3}{2}(t_2 - t_1) = 158.7423\,[s];$$

$$\text{Retardo} \;\rightarrow\; L = t_2 - T = 31.7485\,[s]$$

A continuación se presenta el desarrollo para el respectivo deseño de los controladores P, PI, PD y PID:

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

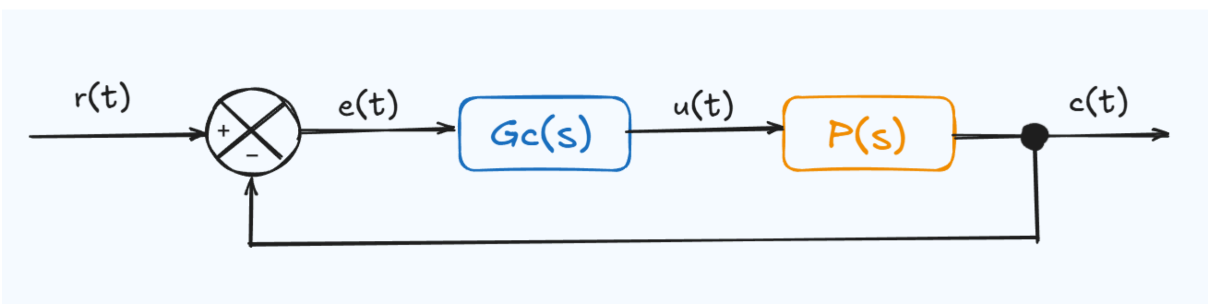
Tabla 1. *Ecuaciones necesarias para el diseño de controladores usando el método de Ziegler-Nichols.*

En este caso se definieron las constantes para un **controlador PI**.

$$\text{Funciona mejor cuando} \quad \rightarrow \quad 0.1 < \frac{L}{T} < 0.6$$

```
% Criterio de Ziegler-Nichols
kp_ZN = (0.9*T)/(k*L)
Ti_ZN = L/0.3
Td_ZN = 0
ki_ZN = kp_ZN/Ti_ZN
```

Figura 4. *Ecuaciones evaluadas en el software MATLAB.*



El siguiente procedimiento corresponde al cálculo del **controlador proporcional (P)**:

$$G_{\mathrm{LC}}(s) = \frac{C(s)}{R(s)} = \frac{G_C(s) \cdot P(s)}{1 + G_C(s) \cdot P(s)} \approx \frac{\omega_n^2}{s^2 + 2 \cdot \zeta \cdot \omega_n \cdot s + \omega_n^2}$$

$$G_C(s) = \frac{U(s)}{E(s)} = K\,\frac{(s+z_1)\cdot(s+z_2)\ldots(s+z_m)}{(s+p_1)\cdot(s+p_2)\ldots(s+p_n)} \quad \rightarrow \quad \text{Función de Transferencia del Controlador Generalizada}$$

$$P(s) = \frac{N(s)}{D(s)} \quad \rightarrow \quad \text{Función de Transferencia del Sistema } o \text{ Planta}$$

$$G_{LC}(s) = \frac{C(s)}{R(s)} = \frac{K\frac{(s+z_1)\cdot(s+z_2)\ldots(s+z_m)}{(s+p_1)\cdot(s+p_2)\ldots(s+p_n)}\cdot\frac{N(s)}{D(s)}}{1+K\frac{(s+z_1)\cdot(s+z_2)\ldots(s+z_m)}{(s+p_1)\cdot(s+p_2)\ldots(s+p_n)}\cdot\frac{N(s)}{D(s)}}$$

$$G_{LC}(s) = \frac{K\cdot(s+z_1)\cdot(s+z_2)\ldots(s+z_m)\cdot N(s)}{(s+p_1)\cdot(s+p_2)\ldots(s+p_n)\cdot D(s)+K\cdot(s+z_1)\cdot(s+z_2)\ldots(s+z_m)\cdot N(s)}$$

$$\frac{K\cdot(s+z_1)\cdot(s+z_2)\ldots(s+z_m)\cdot N(s)}{(s+p_1)\cdot(s+p_2)\ldots(s+p_n)\cdot D(s)+K\cdot(s+z_1)\cdot(s+z_2)\ldots(s+z_m)\cdot N(s)} \approx \frac{\omega_n^2}{s^2+2\cdot\zeta\cdot\omega_n\cdot s+\omega_n^2}$$

$$(170.65\cdot s+1)+K_P\cdot(0.11864\cdot e^{-30\cdot s}) = s^2+2\cdot\zeta\cdot\omega_n\cdot s+\omega_n^2$$

El término $e^{-30\cdot s}$ no se puede igualar directamente con los coeficientes, ya que la parte izquierda de la igualación, en su forma actual, tiene un "orden infinito" debido a la exponencial. Una solución a este inconveniente es **aproximar el retardo** con una **"aproximación racional"**, es decir, una fraccción de dos polinomios. La más común y utilizada es la **APROXIMACIÓN DE PADÉ** de primer orden.

Cabe resaltar que si se usa un orden mayor en la aproximación de Padé, también aumenta el orden del polinomio resultante para hacer la respectiva igualación.

$$e^{-\theta\cdot s} \approx \frac{1-\frac{\theta}{2}\cdot s}{1+\frac{\theta}{2}\cdot s} \quad \rightarrow \quad e^{-30\cdot s} = \frac{1-\frac{30}{2}\cdot s}{1+\frac{30}{2}\cdot s} = \frac{1-15\cdot s}{1+15\cdot s}$$

Entonces se obtiene lo siguiente:

$$(170.65\cdot s+1)+K_P\cdot\left(0.11864\cdot\left(\frac{1-15\cdot s}{1+15\cdot s}\right)\right) = s^2+2\cdot\zeta\cdot\omega_n\cdot s+\omega_n^2$$

Operando la parte izquierda se obtiene lo siguiente:

$$s^2+(0.07253-0.00069\cdot K_P)\cdot s+(0.0003907+0.0000463\cdot K_P) = s^2+2\cdot\zeta\cdot\omega_n\cdot s+\omega_n^2$$

Se hacen las respectivas igualaciones:

$$2\cdot\zeta\cdot\omega_n = 0.07253-0.00069\cdot K_P$$

$$\zeta\cdot\omega_n = 0.03627-0.00035\cdot K_P$$

Se definen y calculan parámetros para el diseño del controlador y poder despejar la primera igualación:

$$\textit{Parámetro}\text{ que se asume para el diseño del controlador} \quad \rightarrow \quad \text{OV (Overshoot)} \leq 20\,\%$$

$$\zeta \quad \rightarrow \quad \text{Factor de Amortiguamiento}$$

$$\zeta \geq \sqrt{\frac{\ln(\text{OV})^2}{\pi^2 + \ln(\text{OV})^2}} \quad \rightarrow \quad \zeta = 0.456$$

$$\zeta < 1 \quad \rightarrow \quad \text{Subamortiguado}$$

$$t_{s(2\%)} = \frac{4}{\zeta \cdot \omega_n} \quad \rightarrow \quad \text{Tiempo de Establecimiento}$$

Ahora, se puede despejar ω_n y obtener una **`Ecuación #1`**:

$$\zeta \cdot \omega_n = 0.03627 - 0.00035 \cdot K_P$$

$$(0.456) \cdot \omega_n = 0.03627 - 0.00035 \cdot K_P$$

$$\omega_n = 0.03627 - 0.00035 \cdot K_P \quad \rightarrow \quad \text{Ecuación \#1}$$

Finalmente, se hace la última igualación y obtener una **`Ecuación #2`**:

$$\omega_n^2 = 0.0003907 - 0.0000463 \cdot K_P \quad \rightarrow \quad \text{Ecuación \#2}$$

El siguiente paso es **reemplazar** la **`Ecuación #1`** en la **`Ecuación #2`**, agrupar términos semejantes y **despejar** la respectiva constante **(**K_P**)** del controlador, haciendo las respectivas operaciones se obtiene lo siguiente:

$$(0.07954 - 0.00077 \cdot K_P)^2 = 0.0003907 + 0.0000463 \cdot K_P$$

$$K_{P_1} = 243.5872$$

$$K_{P_2} = 41.1009$$

Se usa la **`Ecuación #2`** para obtener ω_n:

$$\omega_{n_1} = 0.10802 \ \frac{\text{rad}}{s}$$

$$\omega_{n_2} = 0.04789 \ \frac{\text{rad}}{s}$$

Por último, se puede calcular el **Tiempo de Establecimiento**:

$$t_{s(2\%)_1} = 81.20653 \ [s]$$

$$t_{s(2\%)_2} = 183.1683\ [s]$$

El siguiente paso a realizar corresponde a la validación en SIMULINK, la sintonización del controlador usandola teoría de Ziegler-Nichols y el uso de una herramienta para sintonizar el controlador de forma más "automática" y realizar las respectivas comparaciones. Finalmente, se recalca que este sería el mismo procedimiento a realizar para el diseño de otro controlador de forma analítica.

El siguiente código se usa para el diseño y comparación de los diferentes controladores diseñados:

```
% Hecho por: Diego Andrés García Díaz.
% Código: 2195533.
% Fecha: 24/09/2025.
% Asignatura: Control II (Adelanto de Nota).
%
% ------------------------------------------------------------------------------
% -------------- Diseño Controladores para Control de Temperatura --------------
% ------------------------------------------------------------------------------

% This section clears the command window, workspace, and closes any open figures.
% It also defines s as a transfer function variable for later use.
clc; clear; close all;
s = tf('s');

% Definición de parámetros del sistema (Planta) identificada
Kp = 0.11864; % Ganancia del sistema
Tp1 = 170.65; % Tiempo de establecimiento
L = 30; % Tiempo muerto o Delay
P_s = ( (Kp * exp(-L*s) ) / ((Tp1*s) + 1) )    % Modelo identificado del sistema
(Planta)
```

```
P_s =

                  0.1186
  exp(-30*s) * -----------
               170.7 s + 1

Continuous-time transfer function.
Model Properties
```

```
P_s_fact = zpk(P_s)    % P_s factorizada
```

```
P_s_fact =

               0.00069522
  exp(-30*s) * -----------
               (s+0.00586)

Continuous-time zero/pole/gain model.
Model Properties
```

```
% Parámetros del Criterio de Ziegler-Nichols para controlador PI
kp_ZN = (0.9*Tp1)/(Kp*L)
```

```
kp_ZN =
43.1516
```

```
Ti_ZN = L/0.3
Td_ZN = 0
ki_ZN = kp_ZN/Ti_ZN


% Different types of controllers are designed using the calculated Ziegler-Nichols
% parameters. A proportional controller (C_p), a PI controller (C_pi), a PD
controller
% (C_pd), and a PID controller (C_pid) are created.

% Controlador P
% C_p = kp_ZN
C_p = pid(kp_ZN, 0, 0)
```

```
C_p =

  Kp = 43.2

P-only controller.
Model Properties
```

```

% Controlador PI
C_pi = pid(kp_ZN, ki_ZN)
```

```
C_pi =

             1
  Kp + Ki * ---
             s

  with Kp = 43.2, Ki = 0.432

Continuous-time PI controller in parallel form.
Model Properties
```

```
% C_pi = pid(kp_ZN, ki_ZN, 0);

% Controlador PD
C_pd = pid(kp_ZN, 0, Td_ZN)
```

```
C_pd =

  Kp = 43.2

P-only controller.
Model Properties
```

```

% Controlador PID
```

```
C_pid = pid(kp_ZN, ki_ZN, Td_ZN)
```

```
C_pid =

              1
  Kp + Ki * ---
              s

  with Kp = 43.2, Ki = 0.432

Continuous-time PI controller in parallel form.
Model Properties
```

```
% Finally, the continuous controllers are converted to their digital forms using a
% specified sampling time (Ts). The c2d function with 'zoh' (zero-order hold)
% method is used for this conversion, making the controllers suitable for
% implementation in digital control systems.

% Controladores digitales (suponiendo un muestreo de Ts)
Ts = 1; % Tiempo de muestreo
C_p_digital = c2d(C_p, Ts, 'zoh');
C_pi_digital = c2d(C_pi, Ts, 'zoh');
C_pd_digital = c2d(C_pd, Ts, 'zoh');
C_pid_digital = c2d(C_pid, Ts, 'zoh');


% Open the SISO Tool for controller design and analysis
% sisotool(P_s)

% -------------------------------------------------------------------------------
```

```
% s = tf('s');
%
% % Definición de parámetros del sistema (Planta) identificada
% Kp = 0.11864; % Ganancia del sistema
% Tp1 = 170.65; % Tiempo de establecimiento
% L = 30; % Tiempo muerto o Delay
% P_s = ( (Kp * exp(-L*s) ) / ((Tp1*s) + 1) )    % Modelo identificado del sistema
(Planta)
% P_fact = zpk(P_s)    % P_s factorizada
%
% pade_firstOrder = pade(P_s,1)
%
% zpk(pade_firstOrder)
```

```
% s = tf('s');
%
%
% rta = (170.65*s + 1) + (0.11864 * ((-15*s + 1)/(15*s + 1)))
```

```
%
% G_s = tf([0.11864], [170.65 1], 'InputDelay', 30)   % Otra forma de definir una FT
%
% G_aprox = pade(G_s, 1)  % El '1' indica el orden de la Aproximación de Padé
```