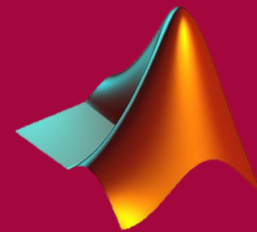


MATLAB[®] & SIMULINK[®]

UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

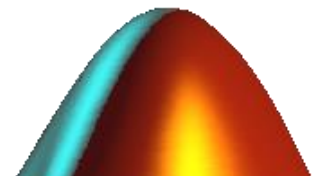
*Facultad de Estudios Superiores
Cuautitlán*

Manual de uso MATLAB



Autor: Rosa Estela Segura Méndez
Revisión: Dr. David Tinoco Varela
Mayo, 2020

Teoría del Control y
Robótica



Índice

1. INICIAR MATLAB...3

- I. Reconocimiento del entorno

2. DIAGRAMA DE BLOQUES...4

- I. Serie
- II. Paralelo
- III. Retroalimentación

3. MODELO DE SISTEMAS EN MATLAB SIMULINK...8

- I. Señal de entrada con escalón unitario
- II. Función de Transferencia en Lazo Abierto
- III. Función de Transferencia en Lazo cerrado
- IV. Sistema de Segundo Orden

4. TRANSFORMADA Y ANTITRANSFORMADA DE LAPLACE... 17

- I. Transformada de Laplace
 - Ejemplo 1
 - Ejemplo 2
 - Ejemplo 3
- II. Antitransformada de Laplace
 - Ejemplo 1

5. FRACCIONES PARCIALES... 22

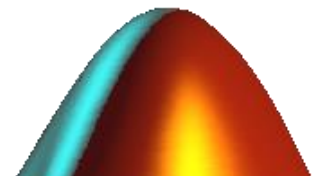
- Ejemplo 1
- Ejemplo 2
- Ejemplo 3

6. SISTEMAS DE PRIMER ORDEN...24

- I. Representación de función de transferencia de forma general
- II. Comparación de constantes de k respecto al tiempo
- III. Sistema de Primer Orden con cero nulo o derivador filtrado

7. SISTEMAS DE SEGUNDO ORDEN...30

- I. Representación de función de transferencia de forma general
- II. Análisis de Funcionamiento con variación de datos



8. DIAGRAMA DE BODE...36

- I. Ejemplo 1
- II. Ejemplo II
- III. Ejemplo II

9. CONTROLADORES P, PI Y PID...45

- I. Controlador Proporcional
- II. Controlador Proporcional Integral
- III. Controlador Proporcional Integral Derivativo

10. ESTABILIDAD...71

- I. Ejemplo 1
- II. Ejemplo II
- III. Ejemplo II

11. LUGAR GEOMÉTRICO DE LAS RAÍCES...79

- I. Ejemplo 1
- II. Ejemplo II
- III. Ejemplo II



1. INICIAR MATLAB

Por su versatilidad y uso extendido el paquete de software MATLAB se presenta como una herramienta potente para la resolución de problemas matemáticos, físicos y de ingeniería en general. En este manual abarcaremos temas referentes a Teoría de Control en donde nos apoyaremos de esta herramienta para realizar diagramas de bloques, cálculos de funciones de transferencia, análisis e interpretación de gráficos como soporte para el aprendizaje de esta materia.

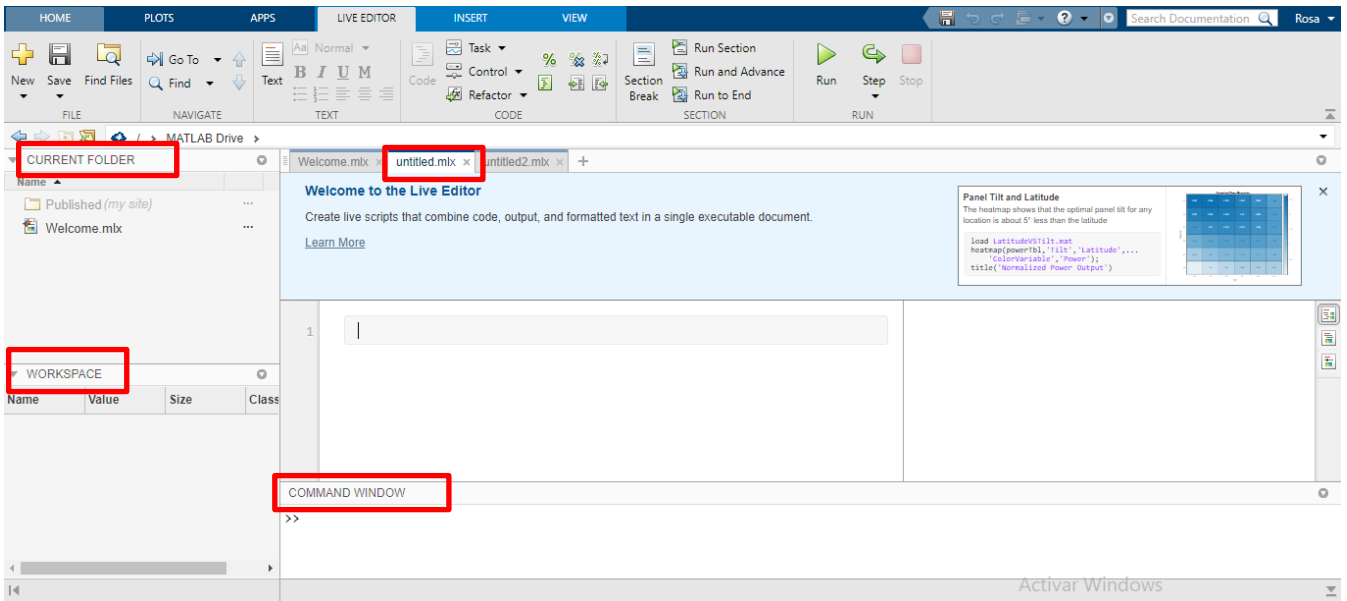
MATLAB (MATrix LABoratory) es un lenguaje de alto nivel y un entorno interactivo para cálculo numérico, visualización y programación. Permite al ingeniero analizar datos, desarrollar algoritmos y crear modelos y aplicaciones propias. Sus puntos fundamentales son:

- Entorno interactivo para la exploración, diseño y solución de problemas.
- Amplia librería de funciones para cálculo científico que permiten trabajar de forma natural con problemas de algebra lineal, estadística, análisis de Fourier o cálculo numérico.
- Biblioteca de funciones gráficas para la visualización en 2D y 3D de datos. Incluye herramientas que permiten la generación de gráficos personalizados.
- Lenguaje de programación de alto nivel.
- Una gran variedad de librerías adicionales (toolboxes) especializadas en campos específicos de la ingeniería y la ciencia.

I. Reconocimiento del entorno

MATLAB cuenta con un entorno con 4 ventanas principales:

- **Command Window:** Es la ventana en la que se escriben las instrucciones que se quieren ejecutar
- **Current Folder:** Muestra el contenido de la carpeta de trabajo. La dirección de la carpeta de trabajo se puede cambiar mediante la barra desplegable que aparece encima de las ventanas.
- **Workspace:** Muestra la información sobre las variables y objetos definidos
- **Editor/ Command History:** Permite ingresar una serie de comandos que se ejecutarán de manera secuencial de tal manera que nos evita ejecutar cada momento de manera individual, esta ventana también puede mostrar los últimos comandos (instrucciones) ejecutados.



1.1 Entorno de trabajo en Matlab

2. DIAGRAMAS DE BLOQUES

En la teoría de control se suele recurrir al concepto de bloques a la hora de representar gráficamente los sistemas, interconectando los distintos bloques que lo forman y dando lugar a lo que se conoce como una estructura de control. Algunos sistemas pueden dar lugar a estructuras de control complejas que interesará, en la mayoría de los casos, reducir o simplificar. Las operaciones más comunes para la simplificación de estas estructuras o diagramas de bloques se resuelven en MATLAB, principalmente, con cuatro comandos: *series*, *parallel*, *feedback*, *minreal*.

Tomando como ejemplo las siguientes funciones de transferencia:

$$G1(s) = \frac{s}{s + 2}; \quad G2(s) = \frac{9}{s^2 + 1.5s + 9}; \quad H1(s) = \frac{0.5}{s + 1}$$

A través del comando `tf` para función de transferencia declaramos cada una de las variables guiándonos de la siguiente secuencia:

`G# = tf (Numerador, [Denominador])`

Tomar en cuenta que los números deben de ir separados para ser agregados a cada variable correspondiente de la función.

Nota: Es importante colocar un ";" al término de cada instrucción ya que de esta manera indicamos al programa que la acción en la línea está terminada

```

1 G1=tf(2,[1 2 ])
2 G2=tf(9,[1 1.5 9])
3 H1=tf(0.5,[1 1])

```

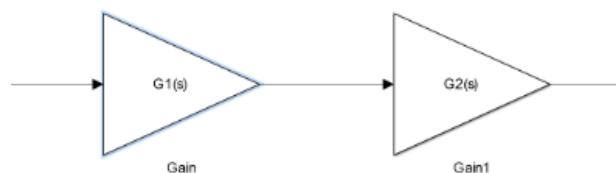
G1 =
 $\frac{2}{s + 2}$
Continuous-time transfer function.
G2 =
 $\frac{9}{s^2 + 1.5 s + 9}$
Continuous-time transfer function.
H1 =
 $\frac{0.5}{s + 1}$
Continuous-time transfer function.

2.1 Funciones de transferencia ejemplo

I. Serie

El comando *series* obtiene la función de transferencia equivalente del conjunto formado por dos bloques conectados en serie. Al comando se le pasan como argumentos las dos funciones de transferencia correspondiente a los sistemas, separadas por comas. Agrupaciones en serie de más de dos bloques se deberán realizar por partes, ya que no es posible realizar en un solo paso la asociación en serie de más de dos bloques.

Una alternativa al comando *series* consiste en realizar directamente la operación de multiplicar, sin ninguna limitación en el número de bloques conectados en serie.



2.2 Bloques conectados en serie

Ejemplo 1.

Diagrama de bloques en serie

Para realizar este ejemplo tomamos la función *series*, el cual multiplica las funciones de transferencia que en este caso es G1 y G2 previamente declaradas.

G#=series (Var1, Var2)

G#=series (Var1*Var2)

→ Si se desea agregar un comentario al flujo lo podemos realizar entre signos de porcentaje

%Comentario%

```
G1=tf(2,[1 2 ])
G2=tf(9,[1 1.5 9])
H1=tf(0.5,[1 1])
```

```
%Serie %
```

```
G12_s=series(G1,G2)
G12_s=(G1*G2)
```

```
Continuous-time transfer function.
G12_s =
```

$$\frac{18}{s^3 + 3.5 s^2 + 12 s + 18}$$

```
Continuous-time transfer function.
G12_s =
```

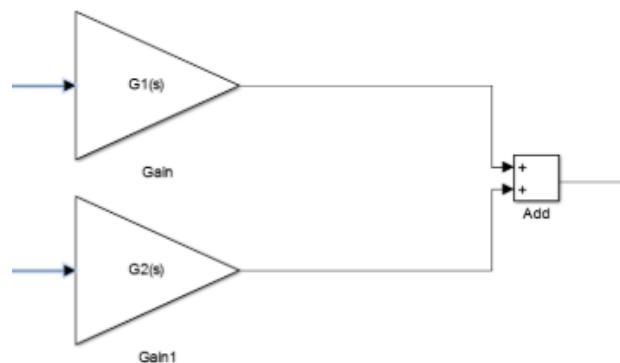
$$\frac{18}{s^3 + 3.5 s^2 + 12 s + 18}$$

```
Continuous-time transfer function.
```

2.3 Instrucciones para diagrama de bloques en serie

II. Paralelo

El comando *parallel* obtiene la función de transferencia del conjunto formado por dos bloques en paralelo. Al comando se le pasan como argumentos las dos funciones de transferencias correspondientes a los sistemas, separadas por comas. Agrupaciones en paralelo de más de dos objetos se deberán realizar por partes. También en este caso, una alternativa al comando *parallel* consiste en realizar directamente la operación de sumar, sin ninguna limitación en el número de bloques conectados en paralelo. En el caso de que los bloques se resten a la salida, puede emplearse el comando *parallel* cambiando el signo de la función de transferencia correspondiente, o bien, recurrir directamente a la operación de restar.



2.3 Diagrama de bloques en paralelo

Ejemplo 2

Para poder realizar este diagrama podemos optar por el comando *parallel* o simplemente por la suma de las funciones que buscamos como se muestra a continuación:

```
G#=parallel(Var1,Var2)
```

```
G#=Var1+Var2
```

```

G1=tf(2,[1 2 ])
G2=tf(9,[1 1.5 9])
H1=tf(0.5,[1 1])

%Serie %

G12_s=series(G1,G2)
G12_s=(G1*G2)

%Paralelo%

G12_p=parallel(G1,G2)
G12_p=G1+G2

```

```

Continuous-time transfer function.
G12_s =

      18
-----
s^3 + 3.5 s^2 + 12 s + 18

Continuous-time transfer function.
G12_p =

      2 s^2 + 12 s + 36
-----
s^3 + 3.5 s^2 + 12 s + 18

Continuous-time transfer function.
G12_p =

      2 s^2 + 12 s + 36
-----
s^3 + 3.5 s^2 + 12 s + 18

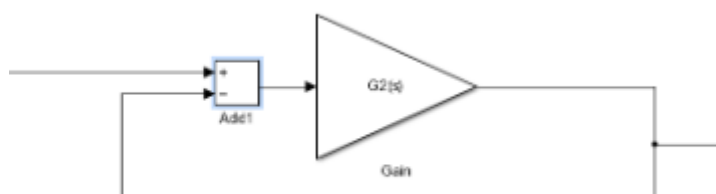
Continuous-time transfer function.

```

2.4 Instrucciones para diagrama de bloques en paralelo

III. Retroalimentación

La función de transferencia equivalente a un sistema retroalimentado se obtiene con el comando *feedback*. A este comando se le pasan, por este orden, la función de transferencias del lazo directo y la de la retroalimentación, separadas por comas. Por defecto el comando *feedback* realiza una realimentación negativa; en el caso de que la realimentación sea positiva, se debe añadir un 1 a continuación de las funciones de transferencia. En caso de tener un sistema realimentado unitariamente, tal y cual se muestra a continuación, su equivalencia será:



2.5 Sistema de Retroalimentación

Como en los casos de *series* y *parallel*, es posible obtener la función de transferencia equivalente de un sistema retroalimentado, expresado en cada caso la ecuación correspondiente. Conviene simplificar las funciones de transferencia de aquellos sistemas que se han obtenido de esta forma, es decir, como resultado de operar directamente sobre los bloques que lo forman (suma, resta, multiplicación). Para ello, el comando *minreal* devuelve el sistema simplificado como resultado de cancelar las raíces del numerador (ceros) con las del denominador (polos). En el caso de operar con los comandos *series*, *parallel* y *feedback*, MATLAB devuelve la solución simplificada.

Ejemplo 3

Como se puede observar en el resultado de las instrucciones para realizar un sistema de realimentación es de vital importancia que éste sea lo más simplificado posible en donde a través del comando `minreal` obtenemos una respuesta más precisa, para poder hacer uso de este comando realizamos los siguientes pasos:

1. Retroalimentación

$$Gr = \text{Varx} / (1 + \text{Varx})$$

2. Simplificación

$$Gr = \text{minreal}(Gr)$$

```
%Serie %
G12_s=series(G1,G2)
G12_s=(G1*G2)

%Paralelo%
G12_p=parallel(G1,G2)
G12_p=G1+G2

%Retroalimentación%
G2_r=G2/(1+G2)
G2_r=minreal(G2_r)
```

G12_p =

$$\frac{2 s^2 + 12 s + 36}{s^3 + 3.5 s^2 + 12 s + 18}$$

Continuous-time transfer function.

G2_r =

$$\frac{9 s^2 + 13.5 s + 81}{s^4 + 3 s^3 + 29.25 s^2 + 40.5 s + 162}$$

Continuous-time transfer function.

G2_r =

$$\frac{9}{s^2 + 1.5 s + 18}$$

Continuous-time transfer function.

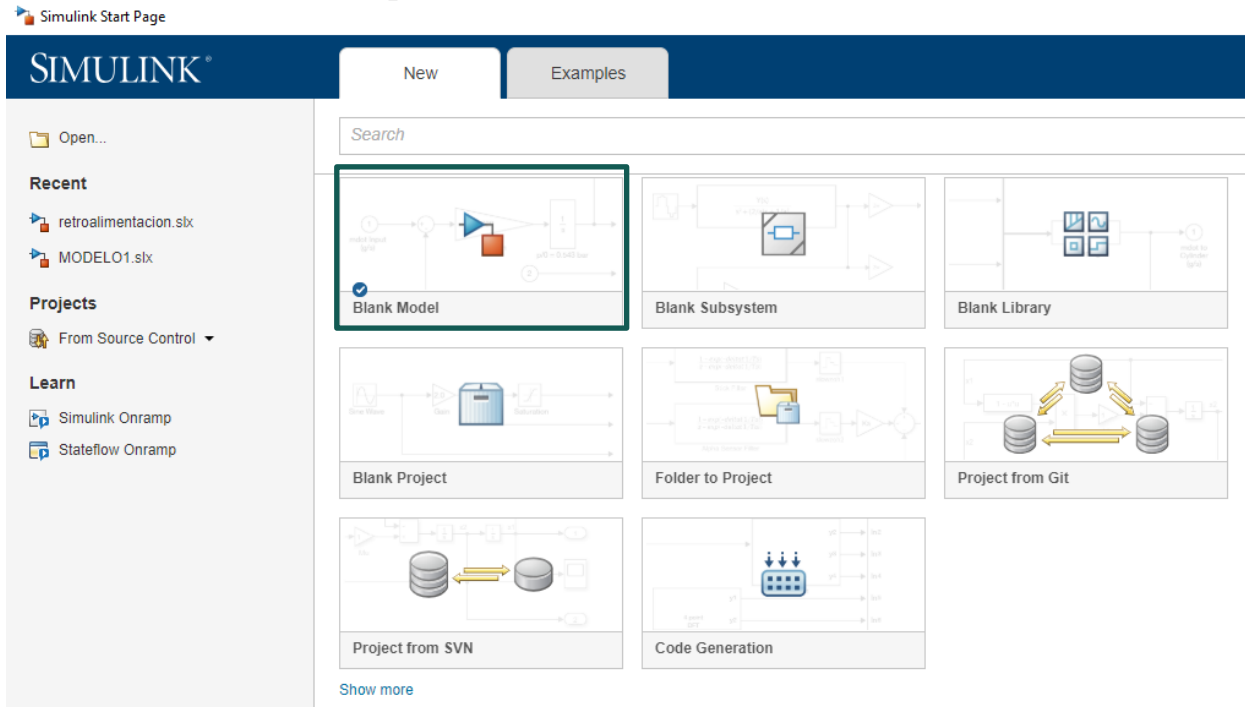
2.6 Instrucciones para Sistema de retroalimentación

3. MODELO DE SISTEMAS EN MATLAB SIMULINK

Simulink proporciona un entorno gráfico al usuario que facilita enormemente el análisis, diseño y simulación de sistemas (de control, electrónicos, etc.), al incluir una serie de rutinas que resuelven los cálculos matemáticos de fondo, junto con una sencilla interfaz para su uso. Proporciona un entorno de usuario gráfico que permite dibujar los sistemas como diagramas de bloques tal y como se haría sobre un papel.

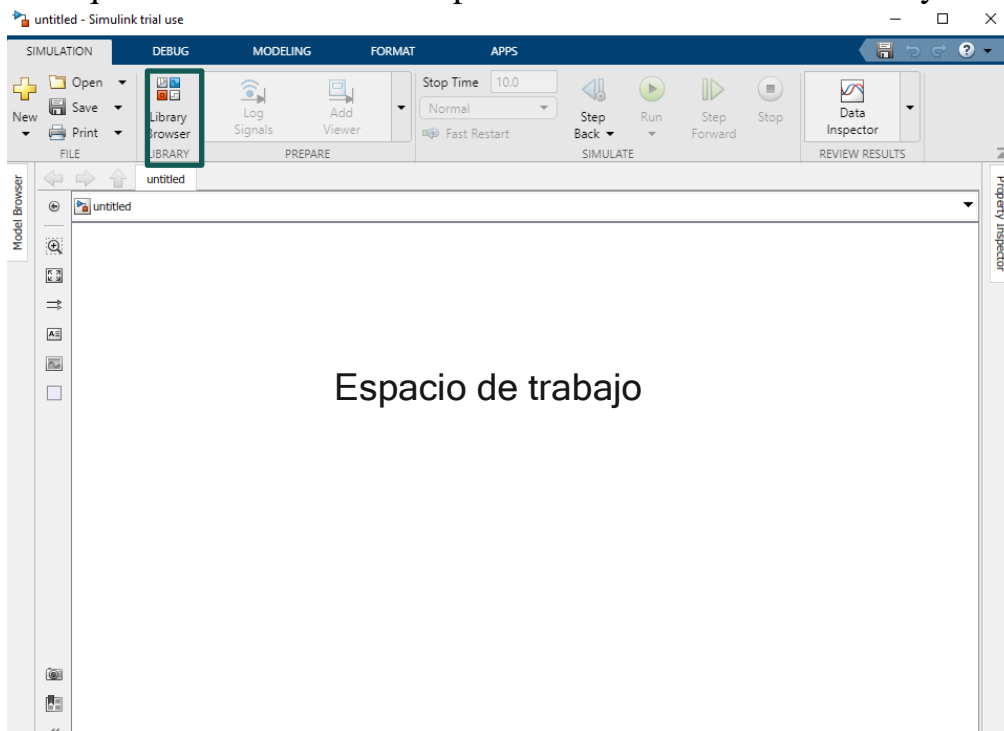
El conjunto de componentes incluidos junto al programa Simulink, incluye bibliotecas de fuentes de señal, dispositivos de presentación de datos, sistemas lineales y no lineales, conectores y funciones matemáticas. En caso de que sea necesario, se pueden crear nuevos bloques a medida por el usuario.

Para poder comenzar con el modelado de sistemas abrimos nuestro entorno gráfico a través de MATLAB, en donde visualizaremos la pantalla Simulink Start Page en donde seleccionaremos la opción de Blank Model:



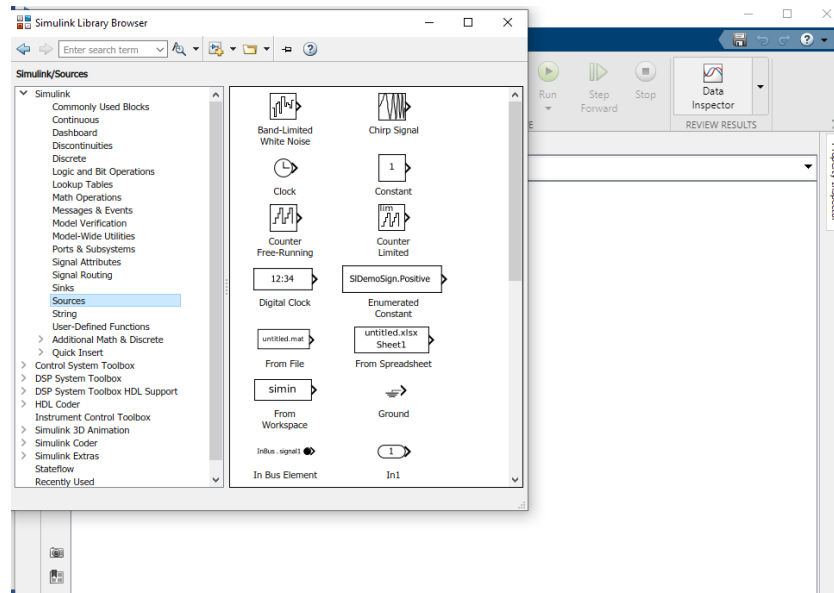
3.1 Simulink Start Page

A continuación encontraremos el espacio de trabajo en donde realizaremos los diagramas, para poder acceder a la librería de todos los elementos para el modelado de sistemas que nos ofrece Simulink presionamos en el botón Library Browser



3.2 Espacio de trabajo Simulink

Después se abrirá una ventana anexa en donde encontraremos todos los elementos para realizar el modelado dependiendo de las características del sistema en donde podemos llevarlos al espacio de trabajo a través del comando Ctrl+I o simplemente arrastrándolos, también se puede realizar la búsqueda de cada uno por su nombre en la barra superior de la pantalla de tal forma que podemos agilizar el proceso de elaboración.

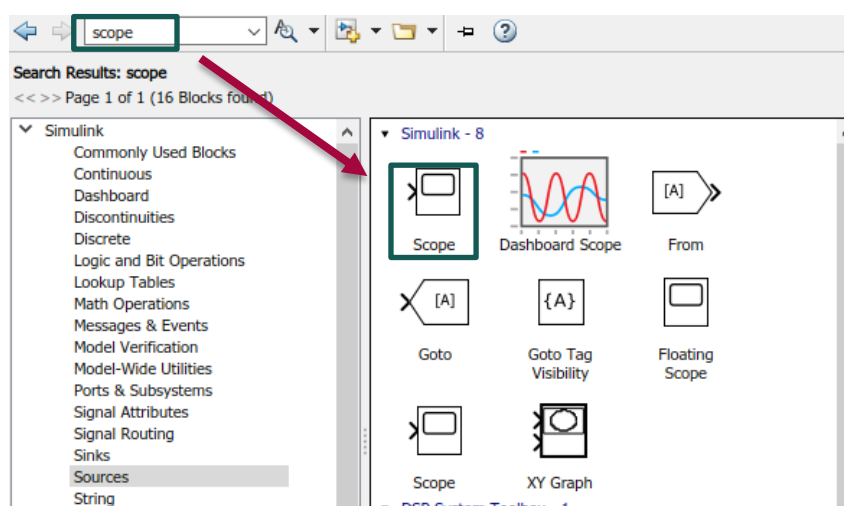


3.3 Búsqueda y anexo de elementos a espacio de trabajo

Ejemplo 1

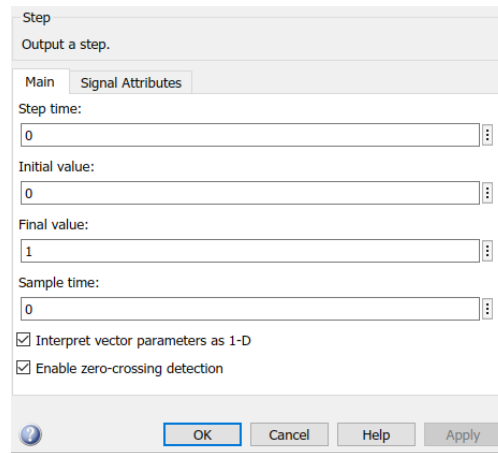
I. Diagrama de bloques con señal de entrada escalón unitario

Comenzamos realizando la búsqueda de los bloques que conformarán nuestro diagrama



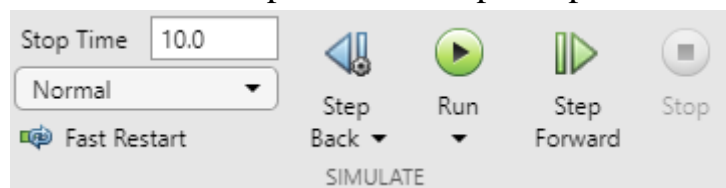
3.4 Selección de bloques por método de búsqueda para diagrama

Cuando se colocan los bloques en el espacio de trabajo los unimos con las flechas guía con las que cuenta cada uno de ellos, antes de realizar la simulación debemos de hacer la configuración del escalón.



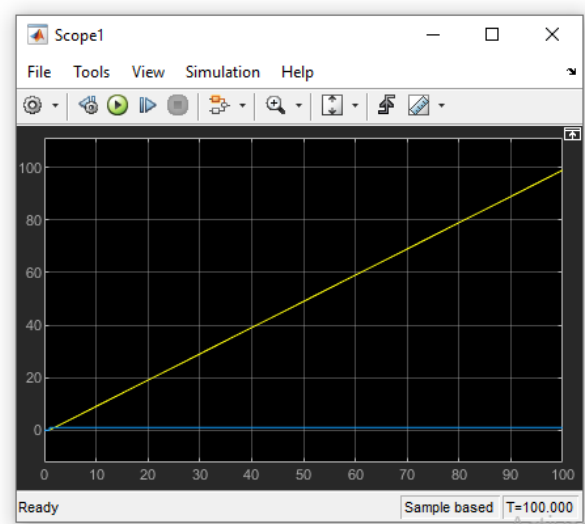
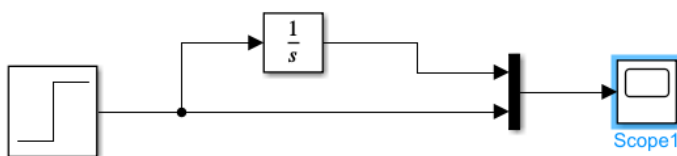
3.5 Configuración de Step

Para poder llevar a cabo la simulación presionamos el boton de RUN que se encuentra en la barra de herramientas del espacio de trabajo o Ctrl+T, podemos definir el tiempo de simulación en el recuadro de Stop Time o tiempo de paro.



3.6 Barra de herramientas para simulación

Finalmente, damos doble click en *scope* para poder visualizar la gráfica que nos arroja nuestro modelo del sistema.

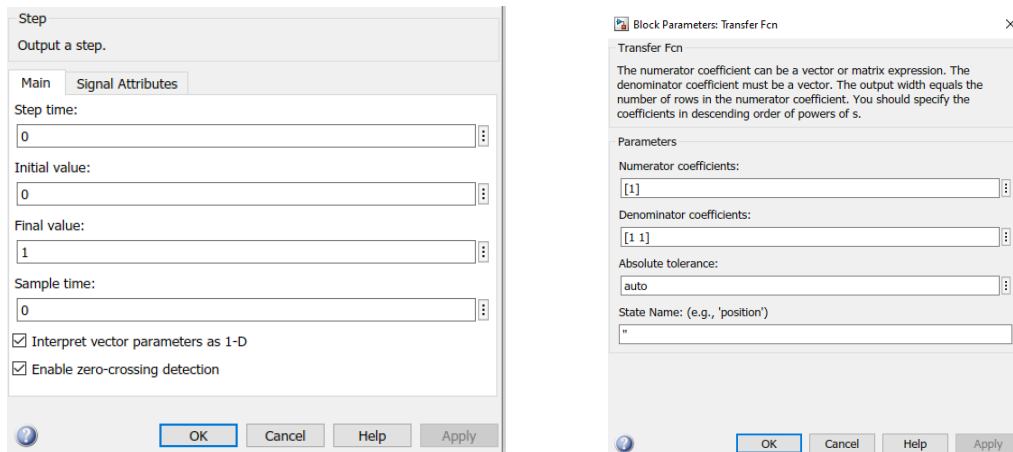


3.7 Diagrama y gráfica de bloques con señal de entrada de escalón unitario

Ejemplo 2

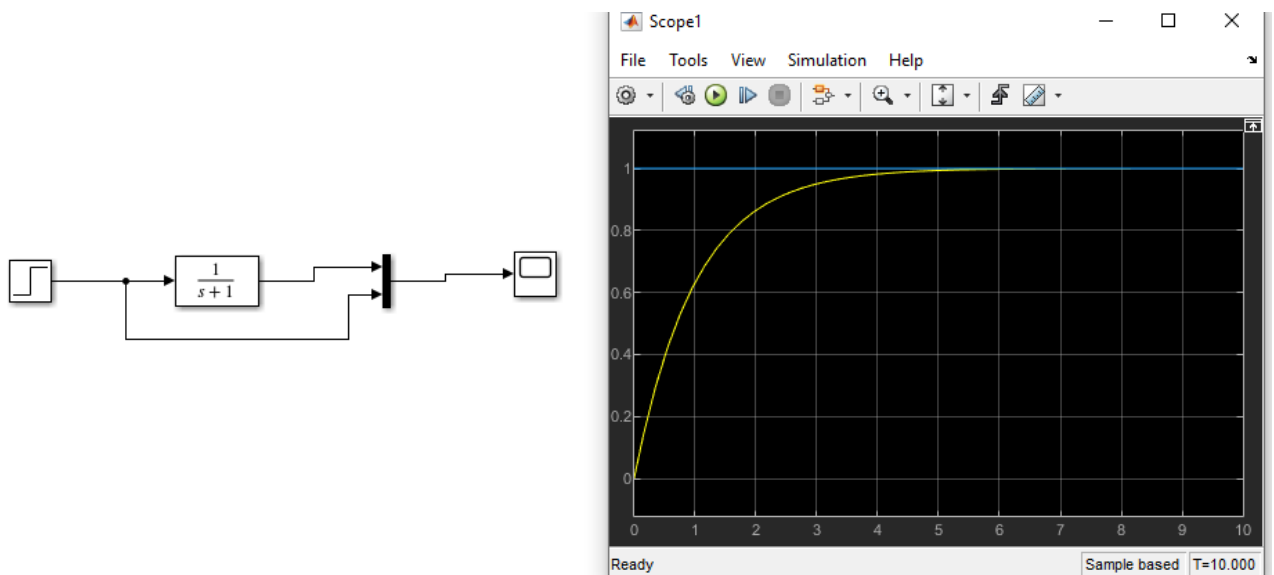
II. Función de transferencia en lazo abierto

Para utilizar funciones de transferencia en SIMULINK se debe incluir un bloque *Transfer Function*, siguiendo los parámetros antes mencionados damos doble pulsación en el elemento para activar el cuadro de diálogo en donde introduciremos los polinomios numerados y denominador siguiendo el orden de cada uno a través de un espacio.



3.8 Configuración de bloques para función de transferencia de lazo abierto

Finalmente simulamos el diagrama y pulsamos el bloque scope para obtener la gráfica de comportamiento del sistema



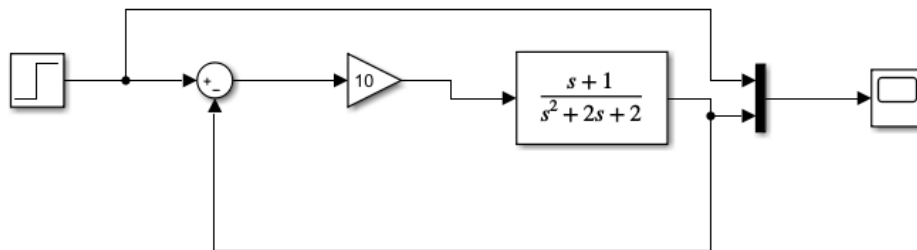
3.9 Diagrama de bloques y gráfica de lazo abierto con función de transferencia de Sistema de Primer Orden

Nota: En esta gráfica observamos como $c(t)$ tiende a 1 pero nunca lo toca debido a que cuando se le asignan valores a la constante de tiempo de la función de respuesta $c(t) = 1 - e^{-\frac{t}{T}}$

Ejemplo 3

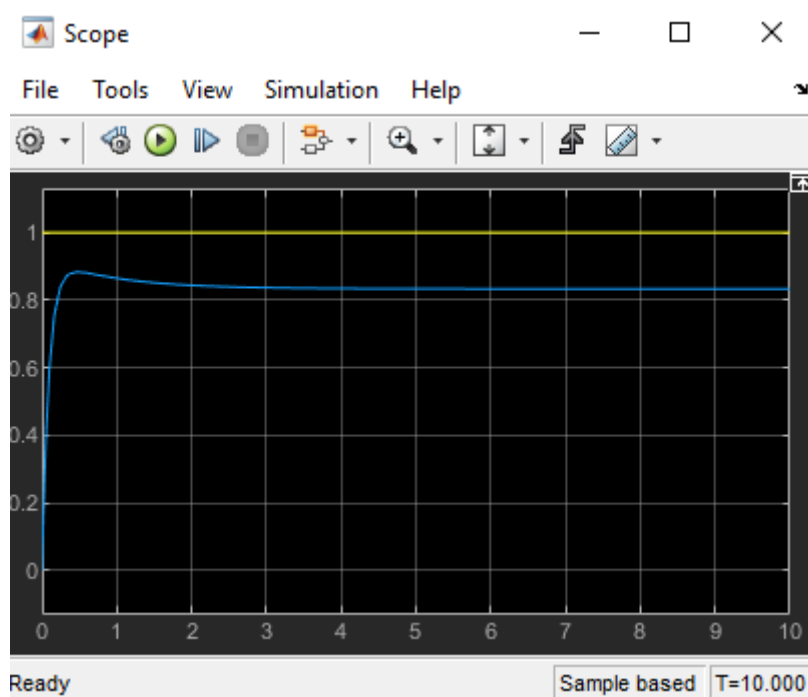
III. Función de transferencia en lazo cerrado

Para que este sistema sea de lazo cerrado debe de cumplir con la función de retroalimentación en donde nos apoyaremos del bloque *sum*. Este bloque puede variar su signo si se trata de una retroalimentación positiva y negativa. Adicionalmente se pueden introducir ganancias mediante el bloque *Gain*. En el siguiente ejemplo se muestra un sistema con todos estos elementos:



3.10 Diagrama de bloques función de transferencia de lazo cerrado

De igual forma que en el ejemplo pasado pulsamos el bloque de *scope* después de correr la simulación para obtener la gráfica de respuesta de nuestro sistema.



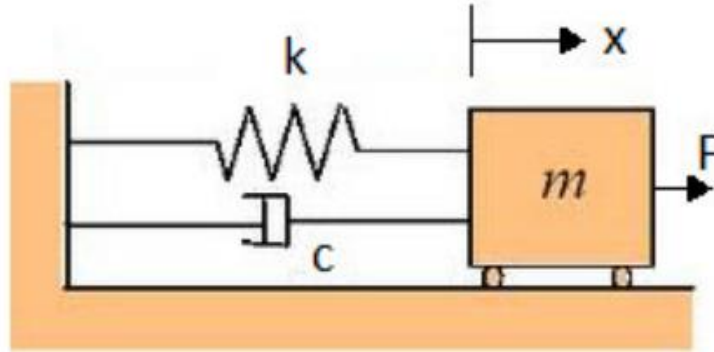
3.11 Gráfica función de transferencia de lazo cerrado

Ejemplo 4

IV. Modelo de Sistemas de Segundo Orden

Sistema Masa Muelle Amortiguador

Descripción:



4.10 Sistema Masa Muelle Amortiguador

Tomando en cuenta que el amortiguador ejerce una fuerza en dirección opuesta al movimiento con una magnitud directamente proporcional a la rapidez del objeto planteamos la siguiente ecuación

$$FA = -hv = hx'$$
$$-hx - hx' = mx'' \rightarrow mx'' + hx' + hx = 0$$

En donde mx'' pertenece a la masa, hx' al amortiguador y hx al muelle o resorte

$$x(0) = x_0 \rightarrow \text{Posición inicial}$$

$$x'(0) = V_0 \rightarrow \text{Velocidad inicial}$$

Realizando la sustitución de las variables

$$F(t) = mx(s)s^2 + cx(s)s + kx(s)$$

Simplificando

$$F(t) = x(s)(ms^2 + cs + k)$$

Valores de los
parámetros:

$$M=0.25 \text{ kg}$$

$$C=0.5 \text{ Ns/m}$$

$$K= 1 \text{ N/m}$$

$$\text{Escalón con mag} = 3$$

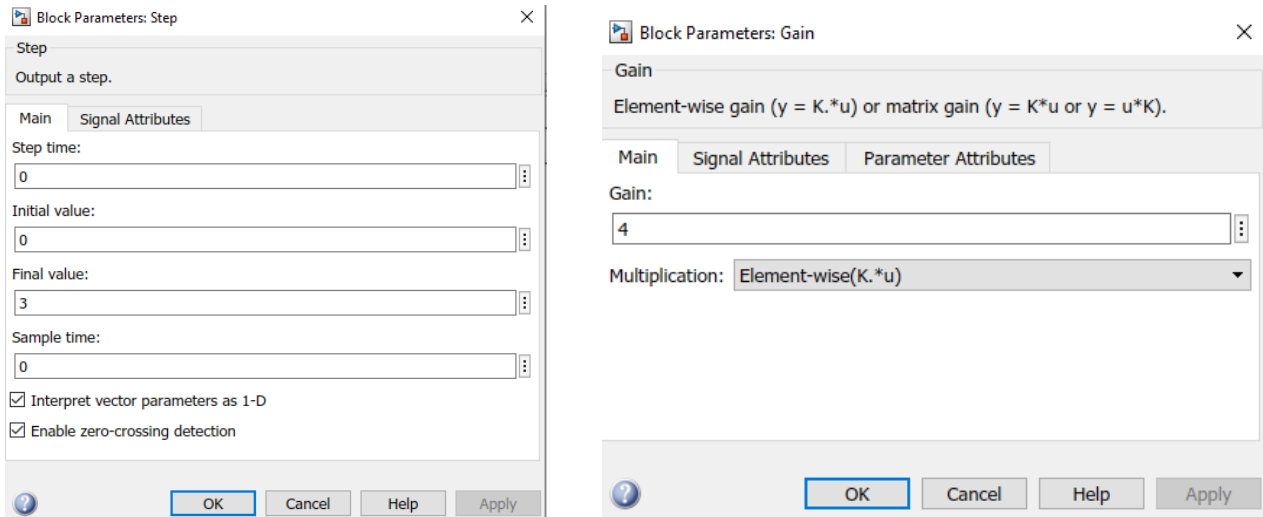
Frecuencia Natural

$$W_n = \sqrt{\frac{k}{M}} = \sqrt{\frac{1\text{N/m}}{0.25\text{kg}}} = 2\text{Hz}$$

Tasa de amortiguamiento

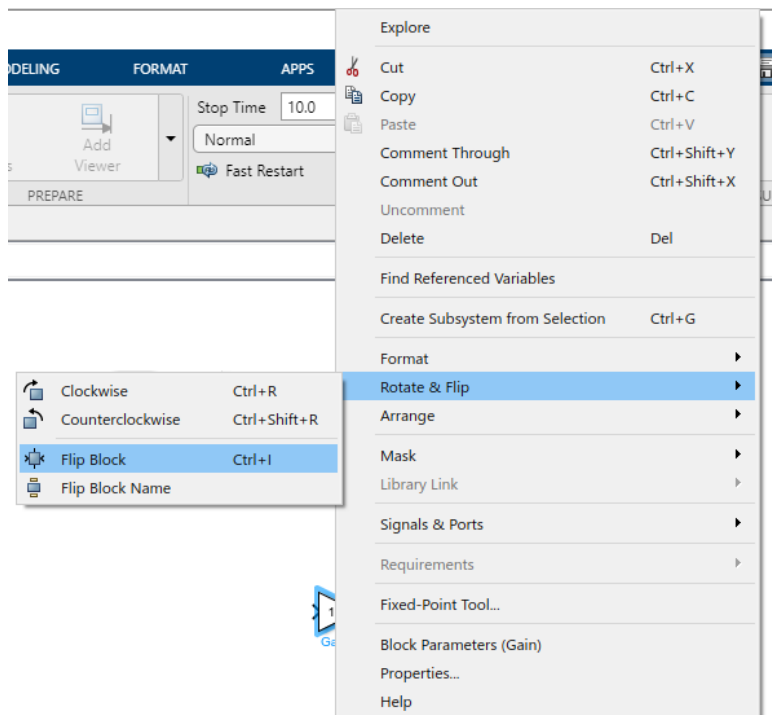
$$\varepsilon = \frac{c}{2W_nM} = \frac{0.5\text{Ns/m}}{(2)(2\text{Hz})(0.25\text{kg})} = 0.5 \rightarrow \text{Sistema subamortiguado}$$

Después de realizar los cálculos pertinentes para reconocer la respuesta del sistema pasamos a la parte del modelado en SIMULINK comenzando por la selección y configuración de los bloques para realizar el diagrama tomando en cuenta los datos proporcionados en el problema.



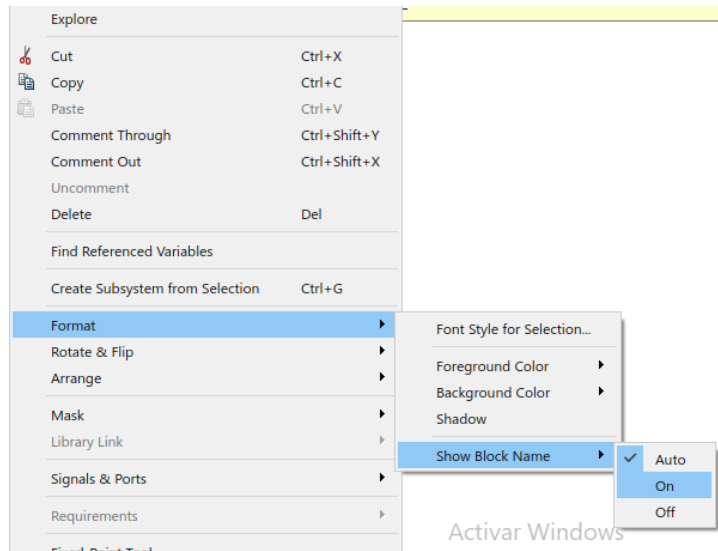
3.12 Configuración de bloques para Sistema de Segundo Orden

Para cambiar el sentido de rotación de los bloques pulsamos click derecho para desplegar le menú de opciones del elemento, seleccionamos Rotate & Flip y después Flip Block como se muestra en la iamgen 3.13



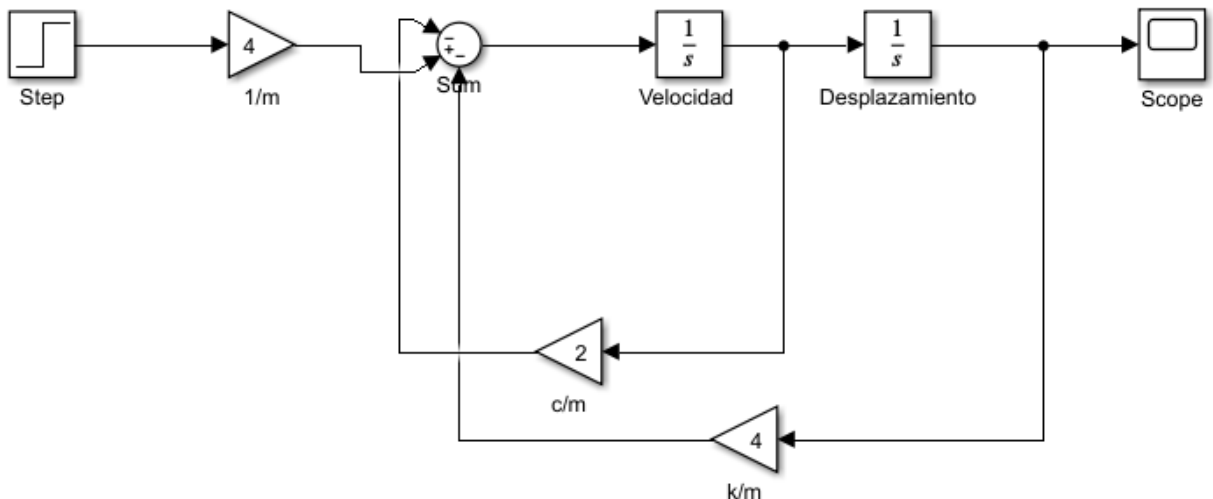
3.13 Cambio de giro de bloque

Si se desea mantener el nombre de los bloques vamos nuevamente al menú de opciones pero esta vez seleccionamos format, en seguida Show Block Name y On, en el caso de que se quiera colocar un nombre en especifico solo basta con posicionar el cursor del mouse en el nombre predeterminado del elemento y realizar la modificación.

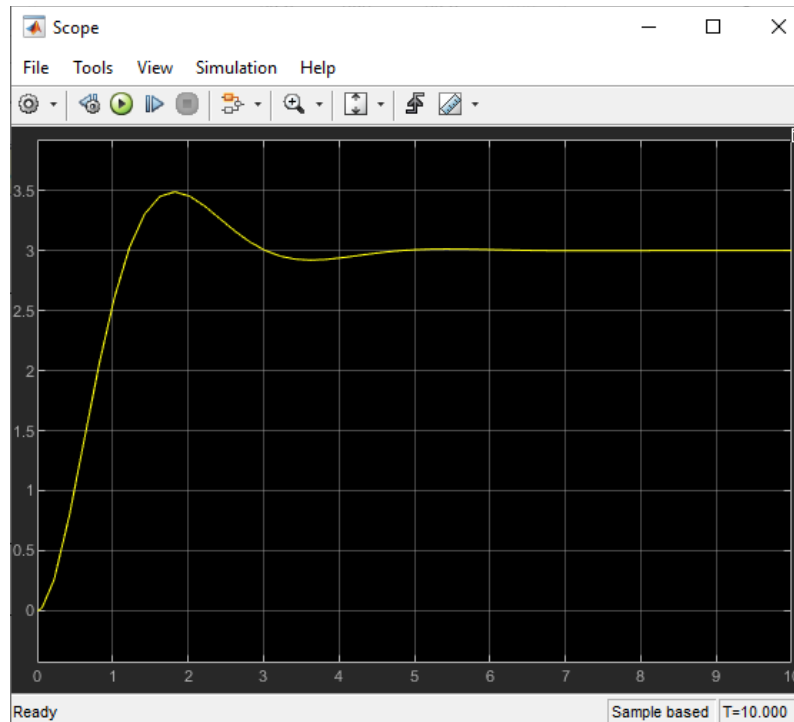


3.14 Mostrar nombre de bloques

A continuación unimos los bloques de acuerdo al orden señalado basado en la ecuación postulada anteriormente y finalmente corremos el programa visualizando en seguida la gráfica de respuesta en el tiempo del Sistema de Segundo Orden planteado.



3.15 Diagrama de Bloques Masa- Muelle- Amortiguador



3.16 Gráfica de respuesta subamortiguada de Sistema Masa- Muelle- Amortiguador

4. TRANSFORMADA Y ANTITRANSFORMADA DE LAPLACE

I. Transformada de Laplace

Es un método operativo que resuelve ecuaciones diferenciales convirtiéndolas en ecuaciones algebraicas. Permite predecir el comportamiento de un sistema sin necesidad de resolver las ecuaciones diferenciales del sistema.

Sea $f(t)$ una función en el tiempo,
La Transformada de Laplace de $f(t)$ es:

$$\mathcal{L} [f(t)] = \int_0^{\infty} f(t)e^{-st} dt$$

A continuación se muestra una tabla algunas de las funciones de uso principal y su Transformada de Laplace

Nombre	Función	Transformada de Laplace
Escalón	$f(t) = A$, para $t > 0$ y $f(t) = 0$, para $t < 0$	$\mathcal{L} [f(t)] = \int_0^{\infty} Ae^{-st} dt = \left. \frac{-Ae^{-st}}{s} \right _0^{\infty} = \frac{A}{s} = F(s)$
Rampa	$f(t) = At$, para $t > 0$ y $f(t) = 0$, para $t < 0$	$\mathcal{L} [f(t)] = \int_0^{\infty} Ate^{-st} dt = \frac{A}{s} \int_0^{\infty} e^{-st} = \frac{A}{s^2} = F(s)$
Exponencial	$f(t) = Ae^{-at}$, para $t > 0$ y $f(t) = 0$, para $t < 0$	$\mathcal{L} [f(t)] = \int_0^{\infty} Ae^{-at} e^{-st} dt = A \int_0^{\infty} e^{-(a+s)t} = \frac{A}{s+a} = F(s)$
Senoidal	$f(t) = A \text{sen}(wt)$, para $t > 0$ y $f(t) = 0$, para $t < 0$	$\mathcal{L} [f(t)] = \int_0^{\infty} A \text{sen}(wt) e^{-st} dt = A \int_0^{\infty} \text{sen}(wt) e^{-st} = \frac{Aw}{s^2 + w^2} = F(s)$
Derivación	Donde $f(0) = f(t)$ cuando $t = 0$, y $f'(0) = \frac{d}{dt} f(t)$ Cuando $t=0$	$\mathcal{L} \left[\frac{d}{dt} f(t) \right] = sF(s) - f(0)$ $\mathcal{L} \left[\frac{d^2}{dt^2} f(t) \right] = s^2 F(s) - sf(0) - f'(0)$
Integración		$\mathcal{L} \left[\int_0^t f(t) dt \right] = \frac{F(s)}{s}$

Tabla 4.1 Funciones y Transformada de Laplace

Para calcular la Transformada de Laplace en MATLAB usamos el comando `laplace(f(t))`

Ejemplo 1

Transformada de Laplace

$$F(t) = \cos(wt) + \text{sen}(wt)$$

Primero utilizamos el comando `syms` el cual se utiliza para crear objetos simbólicos que representan a variables, expresiones y matrices.

Realizamos las siguientes instrucciones para obtener la transformada de Laplace de la función propuesta anteriormente.

`syms w t`

`laplace (cos(w*t)+sin(w*t))`

`pretty (ans)`

```

syms w t
laplace (cos(w*t)+sin(w*t))
pretty(ans)

```

$$\frac{s}{s^2 + w^2} + \frac{w}{s^2 + w^2}$$

4.1 Instrucciones y respuesta a función utilizando Transformada de Laplace

Ejemplo 2

$$f(t) = 3t + 2t^2$$

`syms t`

`laplace((3*t)+2*t^2)`

`pretty (ans)`

```

syms t
laplace((3*t)+2*t^2)
pretty(ans)

```

$$\frac{3}{s^2} + \frac{4}{s^3}$$

4.2 Instrucciones y respuesta a función utilizando Transformada de Laplace

Ejemplo 3

$$f(t) = 3e^{-2t} - 2e^{5t}$$

`syms t`

`laplace(3*exp(-2*t)- 2*exp(5*t))`

`pretty (ans)`

```

syms t
laplace(3*exp(-2*t)- 2*exp(5*t))
pretty (ans)

```

$$\frac{3}{s + 2} - \frac{2}{s - 5}$$

4.3 Instrucciones y respuesta a función utilizando Transformada de Laplace

A continuación se muestra la tabla de Formulas de Transformadas de Laplace para poder realizar la comprobación de nuestros resultados:

TRANSFORMADAS DE LAPLACE	
$f(t) = \mathcal{L}^{-1}[F(s)]$	$F(s) = \mathcal{L}[f(t)]$
1. 1	$\frac{1}{s}, s > 0$
2. t^n	$\frac{n!}{s^{n+1}}, s > 0, n \geq 0$
3. \sqrt{t}	$\frac{\sqrt{\pi}}{2} \cdot s^{-3/2}, s > 0$
4. $\frac{1}{\sqrt{t}}$	$\sqrt{\pi} \cdot s^{-1/2}, s > 0$
5. e^{at}	$\frac{1}{s-a}, s > a$
6. $t^n \cdot e^{at}$	$\frac{n!}{(s-a)^{n+1}}, s > a, n \geq 0$
7. $\text{sen } at$	$\frac{a}{s^2+a^2}, s > 0$
8. $\text{cos } at$	$\frac{s}{s^2+a^2}, s > 0$
9. $t \cdot \text{sen } at$	$\frac{2as}{(s^2+a^2)^2}, s > 0$
10. $t \cdot \text{cos } at$	$\frac{s^2-a^2}{(s^2+a^2)^2}, s > 0$
11. $e^{bt} \cdot \text{sen } at$	$\frac{a}{(s-b)^2+a^2}, s > b$
12. $e^{bt} \cdot \text{cos } at$	$\frac{s-b}{(s-b)^2+a^2}, s > b$

4.4 Fórmulas de Transformadas de Laplace

II. Antitransformada de Laplace

Dada la Transformada de Laplace $F(s)$, la Transformada Inversa de Laplace es $f(t)$. La forma más general de encontrar la transformada inversa es descomponer la función $F(s)$ en fracciones parciales y luego aplicar su inversa a cada término.

$$F(s) = \frac{Q(s)}{P(s)} = \frac{r1}{s-p1} + \frac{r2}{s-p2} + \dots + K$$

Para realizar la operación en Matlab se utiliza el comando *residue*.

$$[r,p,k]=residue(num,den)$$

Ejemplo 1

Si se tiene la siguiente función de transferencia:

$$Y(s) = \frac{(s + 2)(s + 4)}{s(s + 1)(s + 3)}$$

El siguiente código de MATLAB calcula la expansión:

- num = conv ([1 2], [1 4]);
- den = conv ([1 1 0], [1 3]);
- [r,p,k] = residue (num,den);

```
num = conv ([1 2], [1 4])
den = conv ([1 1 0], [1 3])
[r,p,k] = residue(num,den)
```

4.5 Instrucciones en MATLAB

Donde:

r = [-0.1667 -1.5000 2.6667]: residuos

p = [-3 -1 0]: raíces

k = []: ganancia

```
r =
    -0.1667
    -1.5000
     2.6667

p =
    -3
    -1
     0

k =
     []
```

4.6 Respuesta a código de MATLAB

Para el caso de raíces simples y no repetidas:

$$Y(s) = \frac{C1}{s} + \frac{C2}{(s + 1)} + \frac{C3}{(s + 3)}$$

Las constantes Ci son los residuos devueltos por la función residue. Teniendo en cuenta ahora la siguiente relación tabulada de la transformada inversa:

$$F(s) = \frac{1}{s-pi} \rightarrow f(t) = e^{pit}$$

El resultado de la transformación inversa será:

$$y(t) = \frac{8}{3} 1(t) - \frac{3}{2} e^{-t} 1(t) - \frac{1}{6} e^{-3t} 1(t)$$

5. FRACCIONES PARCIALES

Las fracciones parciales son fracciones formadas por polinomios, en las que el denominador puede ser un polinomio lineal o cuadrático y, además, puede estar elevado a alguna potencia. A veces, cuando tenemos funciones racionales resulta de gran utilidad reescribir dicha función como una suma de fracciones parciales o fracciones simples.

Para convertir entre fracciones parciales y coeficientes polinomiales se utiliza el operador *residue*.

En donde la sintaxis para introducirlo a MATLAB es la siguiente:

- `[n , r]= residue ([numerador polinomial], [denominador polinomial])`

[n] representan los valor del numerador

[r] representan los valores del denominador.

MATLAB trabaja con números complejos (Número que se expresa como la suma de un número real y otro imaginario.)

Ejemplo 1

$$F(S) = \frac{4s^2 + 24s + 42}{s^3 + 8s^2 + 25s + 26} = \frac{1 - 0.5i}{s + 3 - 2i} + \frac{1 + 0.5i}{s + 3 + 2i} + \frac{2}{s + 2}$$

```
num=[4 24 42]
den=[1 8 25 26]
[r,p,k]=residue(num,den)
```

```
den = 1x4
      1    8    25    26
```

```
r = 3x1 complex
    1.0000 - 0.5000i
    1.0000 + 0.5000i
    2.0000 + 0.0000i
```

```
p = 3x1 complex
   -3.0000 + 2.0000i
   -3.0000 - 2.0000i
   -2.0000 + 0.0000i
```

```
k =
```

```
 []
```

5.1 Instrucciones y respuesta de $F(s)$ con números complejos

Ejemplo 2

$$F(x) = \frac{4x^2 + 13x - 9}{x^3 + 2x^2 - 3x} = \frac{-1}{(s-3)} + \frac{2}{(s+1)} + \frac{3}{s}$$

```
num=[4 13 -9]
den=[1 2 -3 0]
[r,p,k]=residue(num,den)
```

```
den = 1x4
      1    2   -3    0

r = 3x1
   -1.0000
    2.0000
    3.0000

p = 3x1
   -3.0000
    1.0000
         0

k =

     []
```

5.2 Instrucciones y respuesta de $F(s)$

Ejemplo 3

$$F(s) = \frac{s+2}{s^2+2s+1} = \frac{s+2}{(s+1)^2} = \frac{1}{(s+1)^2} + \frac{1}{(s+1)}$$

```
Command Window

r =

     1
     1

p =

   -1|
   -1

k =

     []
```

```
Untitled* x +
1 num = [1 2];
2 den = [1 2 1];
3 [r,p,k] = residue(num,den)
```

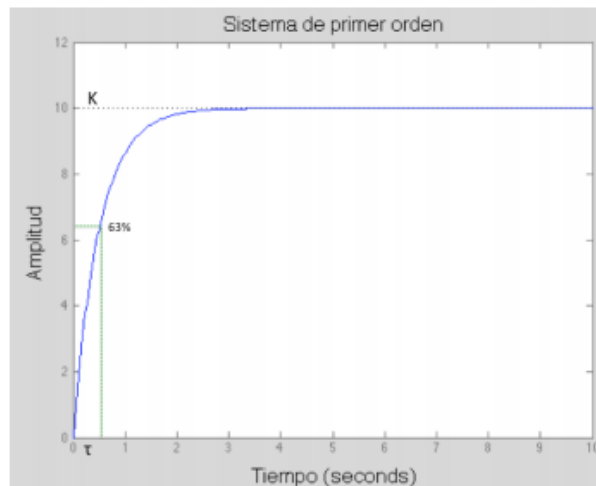
5.3 Instrucciones y respuesta de $F(s)$

6. SISTEMAS DE PRIMER ORDEN

Se denominan sistemas de primer orden a aquellos en los que en la ecuación general se reduce al formato siguiente:

$$\tau \frac{dy}{dt} + y = k u$$

La respuesta típica de estos sistemas no presenta sobreoscilación, esto quiere decir que nunca llegan al valor exacto de la consigna y por lo tanto, son sistemas relativamente lentos.



6.1 Respuesta a Sistema de Primer Orden

En el dominio de Laplace, los sistemas de primer orden están definidos por la siguiente función de transferencia:

$$G(s) = \frac{K}{1 + \tau \cdot s}$$

Donde K es la ganancia del sistema y τ es la constante de tiempo. El valor de la ganancia se calcula mediante la siguiente fórmula:

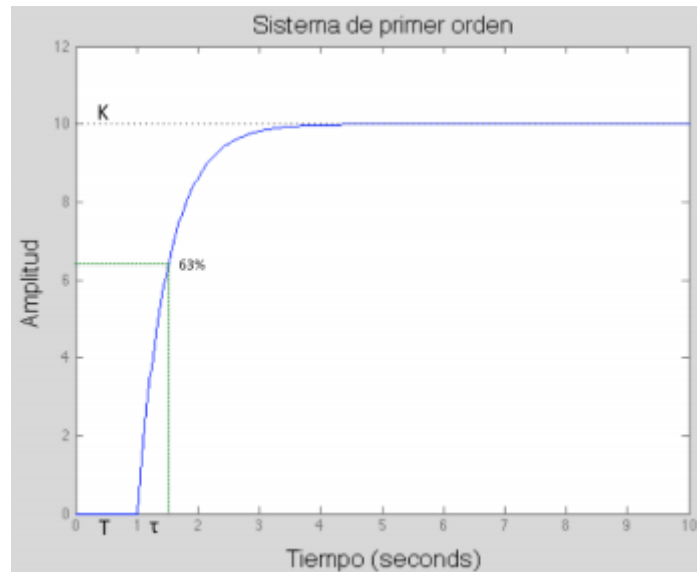
$$K = \frac{\text{Señal salida}}{\text{Señal entrada}} = \frac{\Delta y}{\Delta u}$$

El valor de la constante de tiempo se obtiene sobre la gráfica, para ello se observa el tiempo correspondiente a un valor del 63% Δy .

En ocasiones, también se trabaja con un factor denominado tiempo de establecimiento, que suele estar comprendido entre un 95–98%. Este factor determina el tiempo en el cual la respuesta se estabiliza entre los límites indicados a ese porcentaje. Un caso particular de los sistemas de primer orden, es un sistema de primer orden con retardo. En este caso, la función de transferencia sería del siguiente modo:

$$G(s) = \frac{K}{1 + \tau \cdot s} \cdot e^{-T \cdot s}$$

Donde T es el retardo en la respuesta del sistema. El aspecto de esta respuesta puede verse a continuación:



6.2 Retardo de la respuesta del sistema

Ejemplo 1

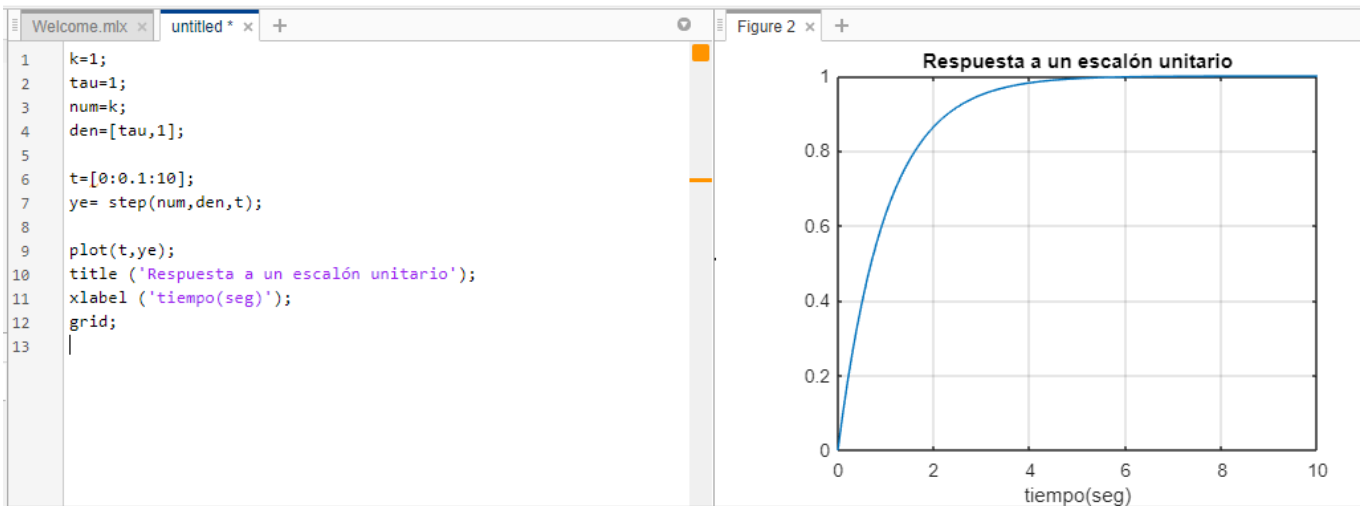
I. Representación de Función de Transferencia de forma general

La representación en forma de función de transferencia viene dada de manera general como:

$$G(s) = \frac{k}{\tau s + 1}$$

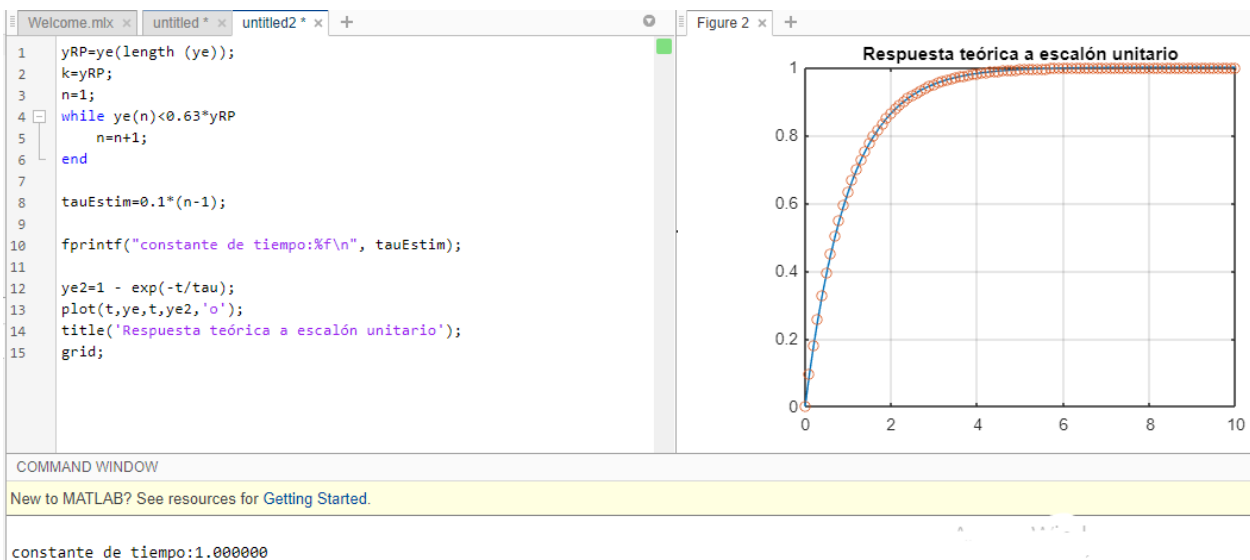
Escribimos en la ventana de *Command Window* las siguientes instrucciones en donde construiremos la función anterior dándole valor a las variables.

- Como siguiente paso definimos un intervalo de simulación para el escalón unitario de la función (*step*) en la cual, incluiremos los valores del numerador, denominador y tiempo.
- Finalmente para visualizar la gráfica de la simulación utilizamos el comando *plot* acompañado de los comandos *title* para agregar un título a la gráfica y *xlabel* el cual sirve para agregar una etiqueta al eje de las “x” si se quisiera poner en el eje de las “y” funcionaría de igual forma, solo debemos de colocar antes el eje en el que deseamos que se muestre.
- Corremos el programa con el botón de “Run” que se encuentra en la barra de herramientas o simplemente realizando la combinación de teclas Ctrl+Enter



6.3 Instrucciones y respuesta a entrada escalón unitario de un sistema de primer orden

Las dos características fundamentales de un sistema de primer orden son su ganancia estática k y su constante de tiempo τ . La constante de tiempo, el tiempo que toma la señal en alcanzar el 63% de la salida máxima. La ganancia estática es el cociente entre la amplitud de la salida y la de entrada en el régimen permanente. Estos valores se pueden comprobar directamente en la gráfica o analizando el vector de datos resultantes. Ejecutamos las siguientes instrucciones además del comando de impresión para poder observar la gráfica en donde nos muestra el tiempo que le toma al sistema alcanzar el 63%.



6.4 Constante de tiempo en un sistema de primer orden y análisis teórico vs simulación en Matlab

Ejemplo 2

Comparación de constantes k respecto al tiempo

Definimos diferentes valores de tiempo para una ganancia de 20, después ingresamos las funciones de transferencia de acuerdo a su constante, en seguida declaramos el escalón para Gs1, utilizamos el comando *hold on* para representar la gráfica añadiendo más datos y colocamos los siguientes escalones para Gs2 y Gs3, si se desea se puede agregar el comando *title* para agregar un título al gráfico mostrado en respuesta al sistema introducido.

```
Editor - Untitled*
Untitled* x +
1      t1=0.2;t2=0.6;t3=3;
2      k=20;
3      Gs1=tf(k, [t1 1])
4      Gs2=tf(k, [t2 1])
5      Gs3=tf(k, [t3 1])
6      step(Gs1,20)
7      hold on
8      step(Gs2,20)
9      step(Gs3,20)
10     grid
11     title('Comportamiento de Sistema de Primer Orden')
```

6.5 Instrucciones de ingreso para comparación de Sistemas de Primer Orden

Como respuesta a los datos ingresados podemos visualizar las Funciones de Transferencia en la ventana de Command Window

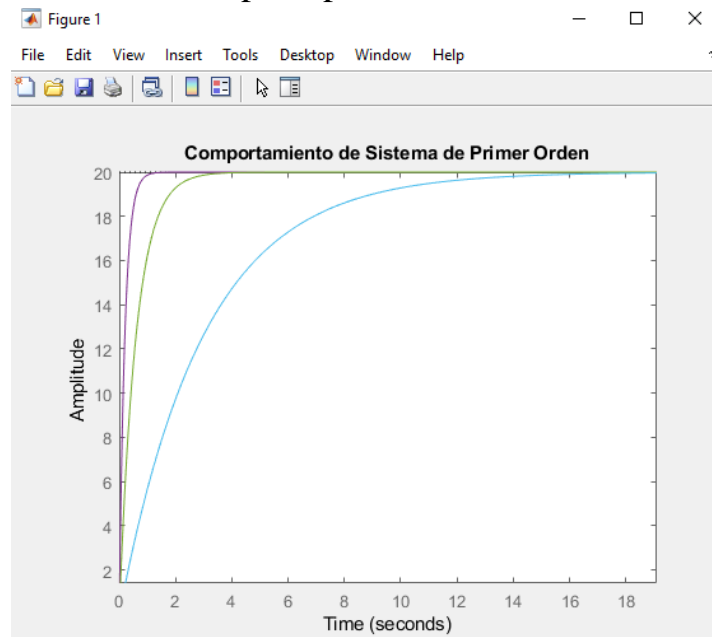
```
Gs1 =
      20
-----
0.2 s + 1
Continuous-time transfer function.

Gs2 =
      20
-----
0.6 s + 1
Continuous-time transfer function.

Gs3 =
      20
-----
3 s + 1
Continuous-time transfer function.
```

6.6 Funciones de Transferencia en Command Window

Finalmente obtenemos la gráfica de comportamiento del sistema en donde podemos observar que cuanto menor es la constante de tiempo más rápido es el sistema ya que este tiene un comportamiento de tipo exponencial.



6.7 Gráfica de comportamiento del Sistema de Comparación de Primer Orden

Ejemplo 3

III. Sistema de Primer Orden con cero nulo o derivador filtrado

Este sistema se define por la siguiente función de transferencia

$$G(s) = \frac{Y(s)}{U(s)} = \frac{Ks}{1 + \tau s}$$

Los parámetros que aparecen en la función de transferencia son la ganancia K, que en este caso no coincide con la ganancia estática $G(0)$ que es igual a 0, y la constante de tiempo τ . La ganancia K puede tener cualquier signo, mientras que τ debe ser positivo para que el sistema sea estable.

Para la función:

$$G(s) = \frac{5s}{s+2} = \frac{2,5s}{1+0,5s} \qquad y(t) = 25e^{-2t} \gamma(t)$$

Registramos las siguientes instrucciones en nuestro programa MATLAB para definir los valores del numerador y denominador de la función de transferencia así como el valor de la ganancia estática, en la siguiente imagen también se muestra el comando *title* que nos auxilia para realizar el cambio de título de la gráfica de respuesta.

```
t=1
k=2.5
Gs=tf(k, [t 0.5])
step(Gs,2.5)
grid
title('Comportamiento de Sistema de Primer Orden')
```

6.8 Instrucciones para Sistema de Primer Orden Cero nulo

De igual forma que en los ejemplos anteriores observaremos la vista preliminar de la función de transferencia que estamos evaluando.

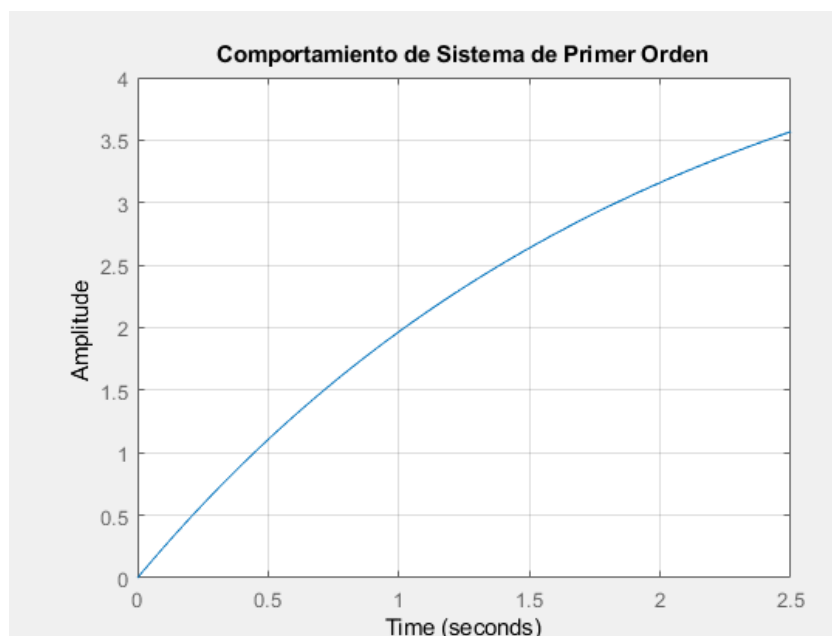
```

      2.5
-----
      s + 0.5

Continuous-time transfer function.
```

6.9 Función de transferencia en Command Window

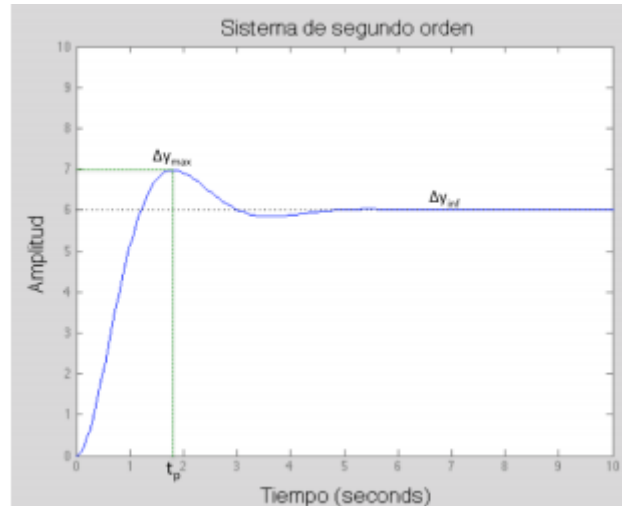
Finalmente obtenemos la gráfica de comportamiento que nos muestra en donde siempre y cuando la constante de tiempo τ sea positiva el sistema se mantendrá estable.



6.10 comportamiento de sistema de primer orden con cero nulo

7. SISTEMAS DE SEGUNDO ORDEN

La respuesta de este tipo de sistemas presenta sobreoscilación y un periodo transitorio con oscilación.



7.1 Respuesta a Sistema de Segundo Orden

La mayoría de los sistemas industriales se comportan como un sistema de este tipo, en el cual posteriormente el control pretende limitar parámetros como la sobreoscilación, del tiempo de establecimiento y error en régimen permanente.

La función de transferencia típica de un sistema de 2º orden es la siguiente:

$$G(s) = \frac{k\omega_n^2}{s^2 + 2\zeta\omega_n s + \omega_n^2}$$

Donde:

- K es la ganancia del sistema:

$$K = \frac{\Delta y}{\Delta u}$$

- ω_n es la frecuencia natural del sistema
- ξ es el factor de amortiguamiento

Estas dos últimas constantes pueden obtenerse de la sobreoscilación δ y el tiempo de pico t_p , obtenidos a partir de la respuesta del sistema ante una entrada escalón:

$$\delta = \frac{\Delta y_{\max} - \Delta y(\infty)}{\Delta y(\infty)} = e^{-\frac{\xi\pi}{\sqrt{1-\xi^2}}}$$

$$t_p = \frac{\pi}{\omega_n \sqrt{1-\xi^2}}$$

Ejemplo 1

I. Representación de Función de Transferencia de forma general

De acuerdo a la siguiente ecuación que representa a un Sistema de Segundo Orden:

$$G2(s) = \frac{0.2s^2 + 0.3s + 1}{(s + 0.5)(s^2 + 0.4s + 1)}$$

Escribimos en la ventana de comandos las instrucciones para agregar los valores al numerador y los dos denominadores de la ecuación las cuales se multiplicaran para hacer una misma variable, para ver los polos o los ceros en la función de transferencia usamos los comandos *roots (den)* o *roots (num)* respectivamente.

```
Welcome.mlx x | untitle * x | untitle2 * x | untitle3 * x | +
1 num = [.2 .3 1];
2 den1 = [1 .4 1];
3 den2 = [1 .5];
4 den = conv(den1,den2);
5
6 [ceros, polos, gan] = tf2zp (num,den)
7
```

7.2 Instrucciones para agregar valores a función de transferencia y observar polos y ceros

En respuesta MATLAB devuelve un vector conteniendo los ceros de la función de transferencia, un vector conteniendo los polos, y un escalar correspondiente a la ganancia estática.

```
COMMAND WINDOW

constante de tiempo:1.000000

ceros =

    -0.7500 + 2.1065i
    -0.7500 - 2.1065i

polos =

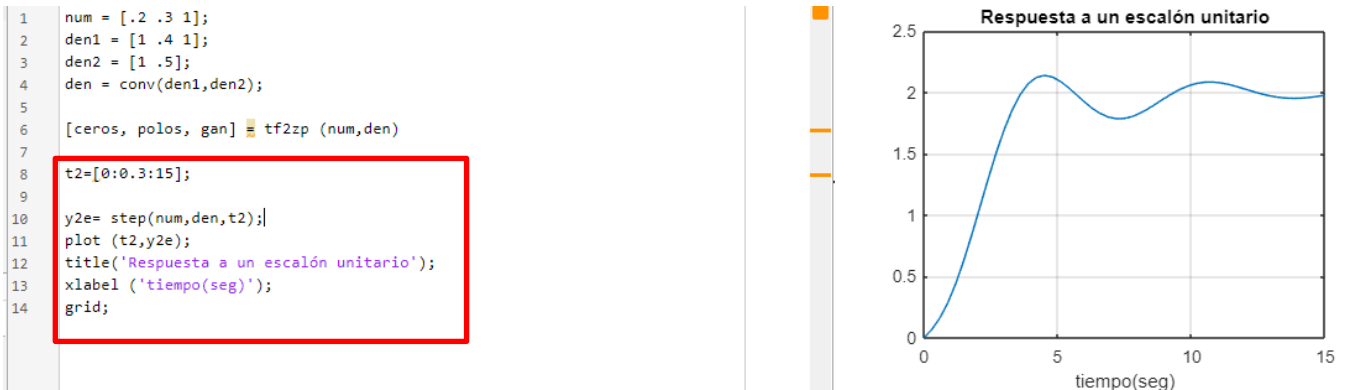
    -0.2000 + 0.9798i
    -0.2000 - 0.9798i
    -0.5000 + 0.0000i

gan =

    0.2000
```

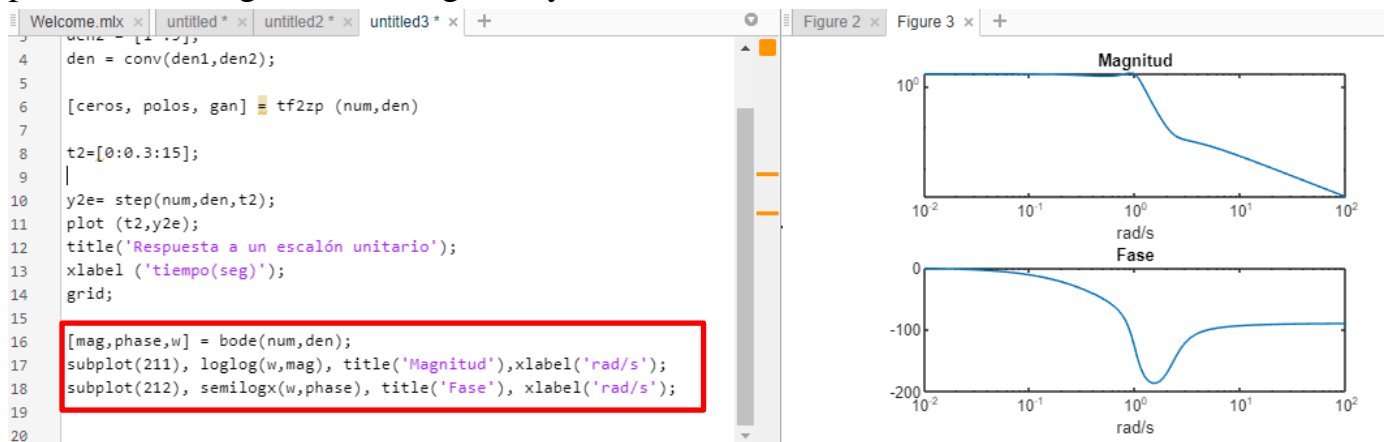
7.3 Ceros y polos de la función de transferencia

La respuesta a la entrada escalón unitario se obtiene con la función *step* para ellos definimos un intervalo de tiempo en la simulación, podemos agregar un título a la gráfica y sus etiquetas con a sus respectivos ejes “x” y “y” a través de los comandos *title* y *label*.



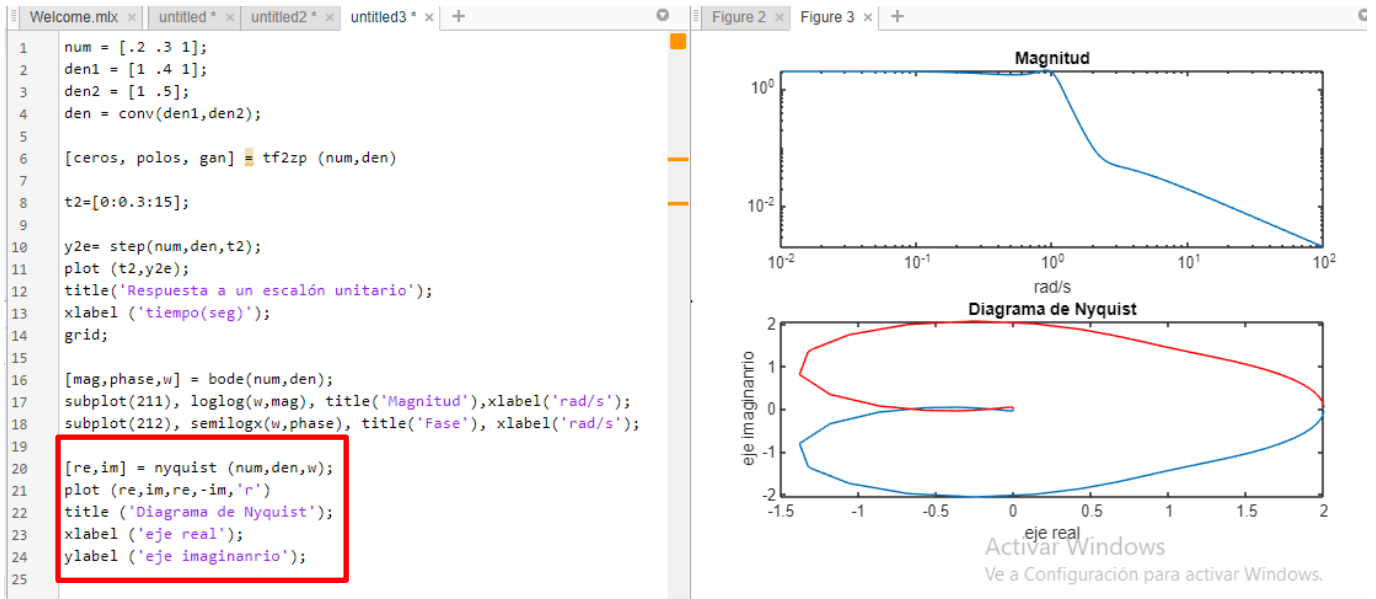
7.4 Instrucciones y gráfica para respuesta a la entrada escalón unitario

La respuesta de los sistemas en frecuencia se puede visualizar utilizando las funciones de Bode, en donde utilizamos los siguientes comandos que se muestran en la imagen para obtener la gráfica de magnitud y fase.



7.5 Comandos y gráfica para hacer usos de funciones de Bode

El comando *nysquit* calcula la parte real como la parte imaginaria de $G(j\omega)$ y realiza la presentación sino se le indican los parámetros de salida. Para obtener la gráfica solo tenemos que realizar la comparación entre la parte real y la parte imaginaria e incluir los comandos que se muestran en la siguiente imagen:



7.6 Diagrama y comandos de Nysquit

Como es bien sabido, los márgenes de estabilidad son el margen de fase y el margen de ganancia. Estos márgenes se calculan con el comando *margin* como se puede observar en la siguiente imagen.

```
26 [mg,mf,wmg,wmf] = margin(num,den)
```

COMMAND WINDOW

1.4372

mf =

8.6856

wmg =

1.2772

wmf =

1.1853

7.8 Márgenes de estabilidad

Ejemplo 2.

II. Análisis de Funcionamiento con variación de datos

En este caso se consideran nuevos valores para nuestra función de transferencia quedando:

$$G(s) = \frac{5s^2 + 5s + 2}{(s + 0.5)(s^2 + 0.4s + 1)}$$

Realizando las instrucciones con los comando como en el ejemplo 1 de este apartado obtuvimos los siguientes ceros, polos y ganancias. Los cuales nos determinan que como el sistema contiene raíces complejas conjugadas es de tipo subamortiguado en donde su constante se encuentra entre 0 y 1 en donde se puede observar que lleva más tiempo en ajustarse que la gráfica de respuesta 7.1

```
num = [5 5 2];
den1 = [1 .4 1];
den2 = [1 .5];
den= conv(den1,den2);
[ceros, polos, gan] = tf2zp(num,den);
t2=[0:0.3:30];
y2e= step(num,den,t2);
plot(t2,y2e);
grid;
```

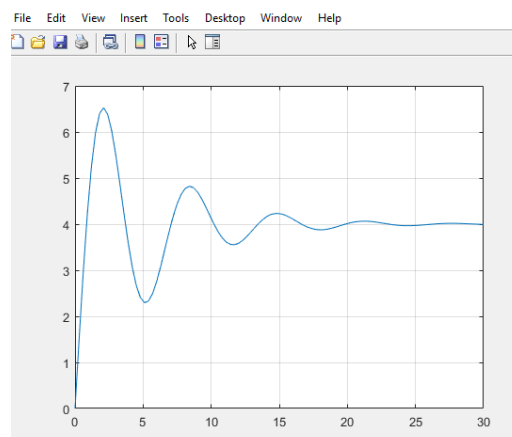
ceros =
-0.5000 + 0.3873i
-0.5000 - 0.3873i

polos =
-0.2000 + 0.9798i
-0.2000 - 0.9798i
-0.5000 + 0.0000i

gan =
5

7.9 Código para Sistema de Segundo Orden subamortiguado, ceros, polos y ganancia estática.

Si lleváramos esta gráfica a un sistema físico, podríamos interpretarla como que el sistema tiene un movimiento constante en donde existe una fuerza que se opone al movimiento, en donde la magnitud de la fuerza es tan pequeña que da característica a este entorno en donde su valor repercute en el tiempo de recuperación del sistema.



7.10 Comportamiento de Sistema de Segundo Orden subamortiguado

8. DIAGRAMAS DE BODE

El diagrama de Bode refleja gráficamente la respuesta en frecuencia de un sistema lineal invariable en el tiempo (LTI). Tanto la amplitud como la fase del sistema LTI se representan en función de la frecuencia. En el diagrama de Bode se utiliza una escala logarítmica para la frecuencia, así como para la amplitud, que se mide en decibelios (dB).

Permite lograr el rendimiento deseado del sistema de lazo cerrado mediante el trazado gráfico de la respuesta y frecuencia de lazo abierto por medio de reglas claras y fáciles de entender. Además, se pueden ver con facilidad el margen de ganancia y el margen de fase del sistema de control.

Ejemplo1

Tomando el siguiente sistema

$$H(S) = \frac{S(S + 5)}{S + 6}$$

Obtenemos los polos y ceros factorizando el 5 y 6 multiplicando S en el polo y en el cero por un uno algebraico

$$H(S) = \frac{S \left(S \frac{5}{5} + 5 \right)}{S \frac{6}{6} + 6} = \frac{5S \left(\frac{S}{5} + 1 \right)}{6 \left(\frac{S}{6} + 1 \right)}$$

Llegamos a su representación en forma logarítmica

$$20 \log H(S) = \frac{5S \left(\frac{S}{5} + 1 \right)}{6 \left(\frac{S}{6} + 1 \right)} = \frac{0.83 S \left(\frac{S}{5} + 1 \right)}{\left(\frac{S}{6} + 1 \right)}$$

Aplicamos las propiedades de los logaritmos de un cociente, en base b, que es igual a la diferencia entre los logaritmos del dividendo y del divisor en la misma base. El logaritmo de un cociente es igual al logaritmo del dividendo menos el logaritmo del divisor.

$$20H(S) = 20 \log(0.83) + 20 \log S + 20 \log \left(\frac{S}{5} + 1 \right) - 20 \log \left(\frac{S}{6} + 1 \right)$$

En donde

$$20 \log(0.83) = -1.61$$

Procedemos a realizar nuestro sistema en Matlab

Ingresando el siguiente código con el comando `margin (sys)` el cual traza la respuesta de Bode.

```
hd = tf([1 5 0],[1 6])  
[Gm, Pm, Wgm, Wpm] = margin(hd)  
margin(hd)
```

8.1 Sintaxis Func Transf Ej. 1

Visualizamos en la Ventana Comand Window nuestra función de transferencia y los márgenes de estabilidad que son el margen de fase y el margen de ganancia.

```
hd =  
  
s^2 + 5 s  
-----  
s + 6  
  
Continuous-time transfer function.
```

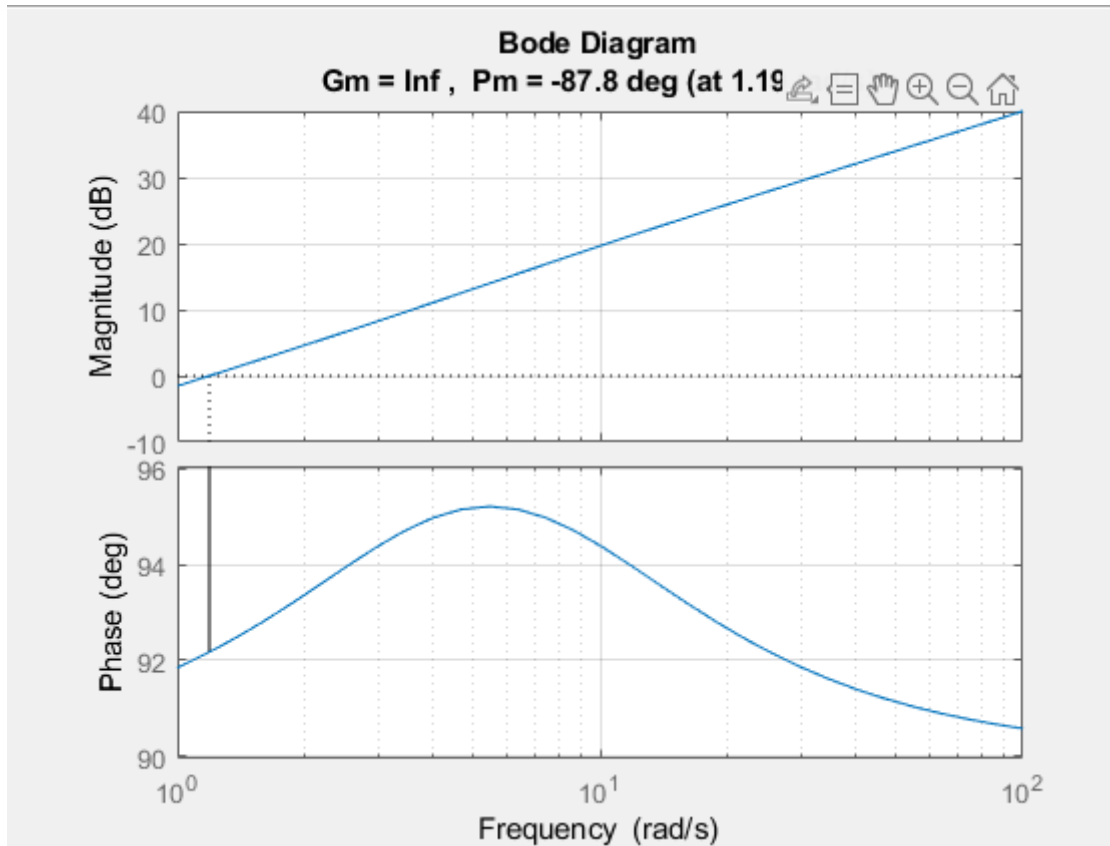
a)

```
Gm =  
  
Inf  
  
Pm =  
  
-87.8305  
  
Wgm =  
  
NaN  
  
Wpm =  
  
1.1901
```

b)

8.2 Comand Window a) Función de transferencia b) Márgenes de ganancia

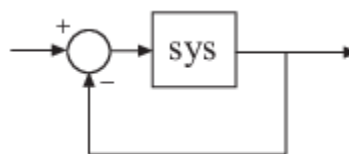
Obtenemos la siguiente gráfica:



8.3 Diagrama de Bode Función de Transferencia 1

Las líneas verticales continuas marcan el margen de ganancia y el margen de fase. Las líneas verticales discontinuas indican las ubicaciones de W_{cp} , la frecuencia donde se mide el margen de fase y W_{cg} , la frecuencia donde se mide el margen de ganancia. El título incluye la magnitud y la ubicación de la ganancia y el margen de fase.

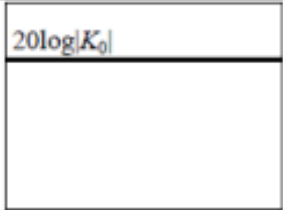
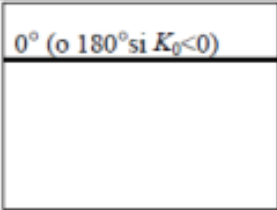
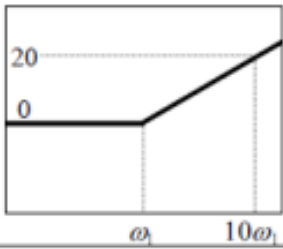
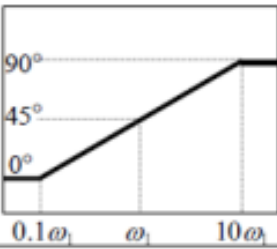
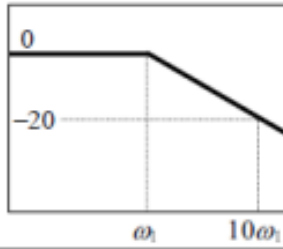
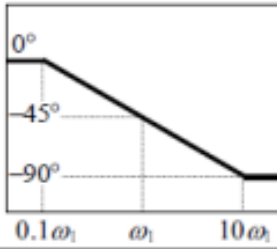
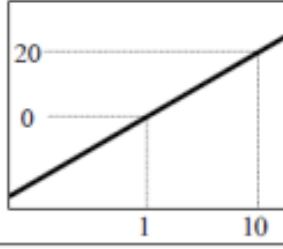
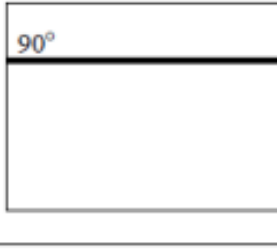
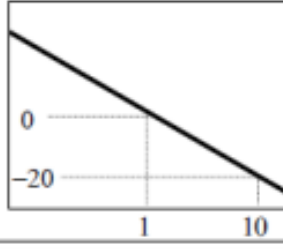
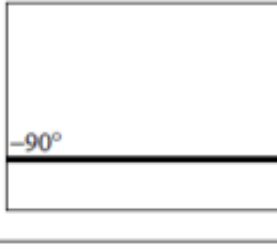
Gm y Pm de un sistema indican la estabilidad relativa del sistema de circuito cerrado formado aplicando retroalimentación negativa unitaria sys, como se muestra en el siguiente diagrama



Gm es, el margen de ganancia es $1 / g$ si g es la ganancia en la frecuencia de fase de -180° . Del mismo modo, el margen de fase es la diferencia entre la fase de la respuesta y -180° cuando la ganancia del bucle es 1.0.

La frecuencia W_{cp} a la cual la magnitud es 1.0 se llama *frecuencia de ganancia unitaria* o *frecuencia de cruce de ganancia*. Por lo general, los márgenes de ganancia de tres o más combinados con márgenes de fase entre 30° y 60° dan como resultado compensaciones razonables entre el ancho de banda y la estabilidad.

La interpretación de la gráfica está apoyada en la siguiente tabla de reglas para llevar a cabo los trazos de magnitud y fase en el diagrama de Bode:

Término	$20\log T_i(j\omega) $	$\angle T_i(j\omega)$
Constante K_0		
Cero $\frac{s}{\omega_1} + 1$		
Polo $\frac{1}{\frac{s}{\omega_1} + 1}$		
Cero en origen s		
Polo en origen $\frac{1}{s}$		

<p>Cero doble</p> $\left(\frac{s}{\omega_1} + 1\right)^2$		
<p>Polo doble</p> $\frac{1}{\left(\frac{s}{\omega_1} + 1\right)^2}$		
<p>Polos complejos</p> $\frac{1}{\left(\frac{s}{\omega_n}\right)^2 + \frac{2\zeta}{\omega_n}s + 1}$		
<p>Caso especial: polos en eje imaginario</p> $\frac{1}{\left(\frac{s}{\omega_n}\right)^2 + 1}$		

Tabla 8.1 Interpretación de trazos para diagrama de Bode

Podemos demostrar el comportamiento de nuestro sistema respecto a las frecuencias, para la magnitud tomamos en cuenta la primera parte de la resolución con ayuda de las propiedades de los logaritmos en donde partimos de $20\log(0.83) = -1.61$ sumado a 20dB ya que cumple con $S=0$ que significa que hay un cero en origen, después vamos a ir realizando la suma de las pendientes en dB/D subiendo a razón de 20dB/D, cuando tenemos $S+5$ quiere decir que tenemos un polo en 5 y subirá como pendiente a razón de 20dB/D solo si ese polo es parte del numerador, en caso contrario, que sea del denominador descenderá a razón de -20dB/D.

Para la fase es más sencillo, en este caso tenemos como primer término una constante por lo que debemos de trazar una línea recta en 0° , si ésta constante llegase a ser menor que 0 la debemos de trazar en 180° , para el segundo término tenemos un cero en el origen debido a que se trata de S así que existe una línea recta en 90° , para el tercer y cuarto elemento que corresponden a los polos deben de empezar en 0.5 y 0.6 ascendiendo o descendiendo según sea la polaridad de los mismos.

Ejemplo 2

Tomando el siguiente sistema

$$H(S) = \frac{S(S + 2)}{(S + 3)}$$

Obtenemos los polos y ceros factorizando el 5 y 6 multiplicando S en el polo y en el cero por un uno algebraico

$$H(S) = \frac{S \left(S \frac{2}{2} + 2 \right)}{S \frac{3}{3} + 3} = \frac{2S \left(\frac{S}{2} + 1 \right)}{3 \left(\frac{S}{3} + 1 \right)}$$

Llegamos a su representación en forma logarítmica

$$20 \log H(S) = \frac{2S \left(\frac{S}{2} + 1 \right)}{3 \left(\frac{S}{3} + 1 \right)} = \frac{0.6 S \left(\frac{S}{2} + 1 \right)}{\left(\frac{S}{3} + 1 \right)}$$

Aplicamos las propiedades de los logaritmos de un cociente, en base b, que es igual a la diferencia entre los logaritmos del dividendo y del divisor en la misma base. El logaritmo de un cociente es igual al logaritmo del dividendo menos el logaritmo del divisor.

$$20H(S) = 20 \log(0.66) + 20 \log S + 20 \log \left(\frac{S}{2} + 1 \right) - 20 \log \left(\frac{S}{3} + 1 \right)$$

En donde

$$20 \log(0.66) = -3.60$$

Continuando con la elaboración del Diagrama de Bode en Matlab, ingresamos nuevamente el código con la función margin cambiando las variables de nuestra función de transferencia como se muestra a continuación:

```
hd = tf([1 2 0],[1 3])
[Gm, Pm, Wgm, Wpm] = margin(hd)
margin(hd)
```

8.4 Sintaxis Func Transf Ej. 2

Visualizamos en la Ventana Comand Window nuestra función de transferencia y los márgenes de estabilidad que son el margen de fase y el margen de ganancia.

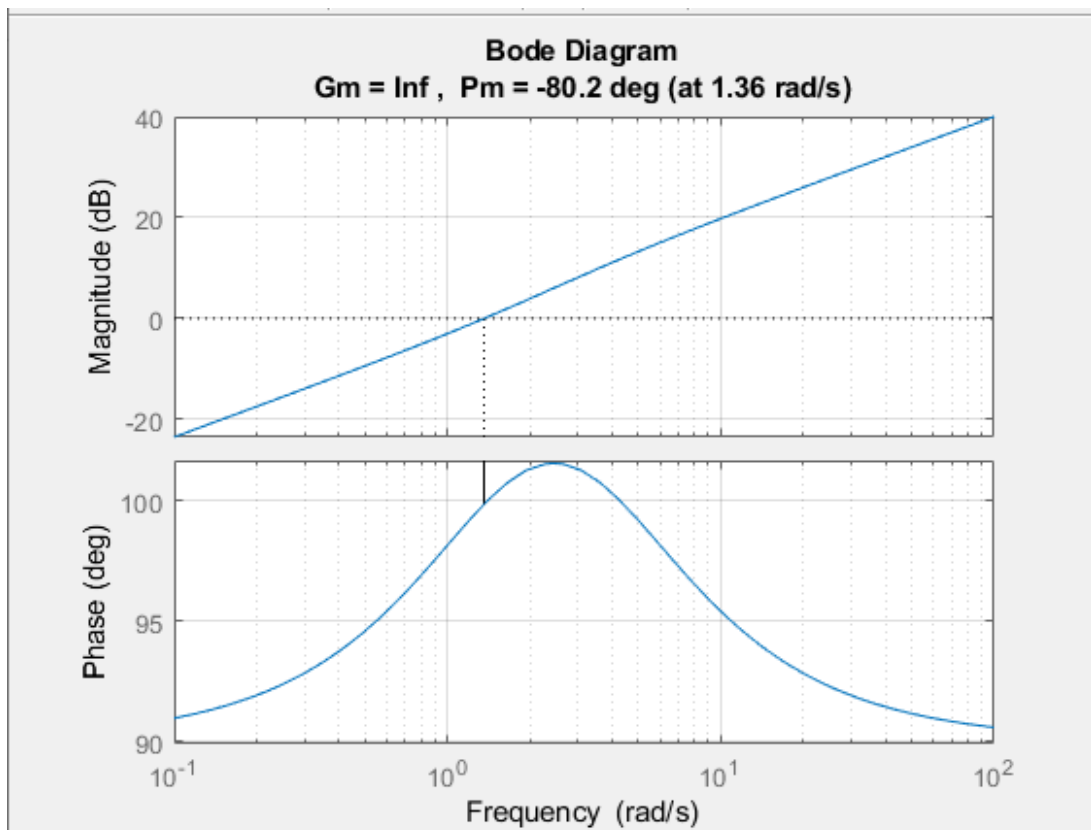
<pre> hd = s^2 + 2 s ----- s + 3 Continuous-time transfer function. </pre>	<pre> Gm = Inf Pm = -80.1645 Wgm = NaN Wpm = 1.3616 </pre>
---	--

a)

b)

8.5 Comand Window a) Función de transferencia b) Márgenes de ganancia

Obtenemos la siguiente gráfica:



8.6 Diagrama de Bode Función de Transferencia 2

Podemos demostrar el comportamiento de nuestro sistema respecto a las frecuencias, para la magnitud tomamos en cuenta la primera parte de la resolución con ayuda de las propiedades de los logaritmos en donde partimos de $20\log(0.66)=-3.60$ sumado a 20dB ya que cumple con $S=0$ que significa que hay un cero en origen, después vamos a ir realizando la suma de las pendientes en dB/D subiendo a razón de 20dB/D, cuando tenemos $S+2$ quiere decir que tenemos un polo en 2 y subirá como pendiente a razón de 20dB/D solo si ese polo es parte del numerador, en caso contrario, que sea del denominador descenderá a razón de -20dB/D.

Para la fase es más sencillo, en este caso tenemos como primer término una constante por lo que debemos de trazar una línea recta en 0° , si ésta constante llegase a ser menor que 0 la debemos de trazar en 180° , para el segundo término tenemos un cero en el origen debido a que se trata de S así que existe una línea recta en 90° , para el tercer y cuarto elemento que corresponden a los polos deben de empezar en 0.2 y 0.3 ascendiendo o descendiendo según sea la polaridad de los mismos.

Ejemplo 3

Tomando el siguiente sistema

$$H(S) = \frac{S(S + 9)}{(S + 7)}$$

Obtenemos los polos y ceros factorizando el 5 y 6 multiplicando S en el polo y en el cero por un uno algebraico

$$H(S) = \frac{S \left(S \frac{9}{9} + 9 \right)}{S \frac{7}{7} + 7} = \frac{9S \left(\frac{S}{9} + 1 \right)}{7 \left(\frac{S}{7} + 1 \right)}$$

Llegamos a su representación en forma logarítmica

$$20 \log H(S) = \frac{9S \left(\frac{S}{9} + 1 \right)}{7 \left(\frac{S}{7} + 1 \right)} = \frac{1.28 S \left(\frac{S}{9} + 1 \right)}{\left(\frac{S}{7} + 1 \right)}$$

Aplicamos las propiedades de los logaritmos de un cociente, en base b , que es igual a la diferencia entre los logaritmos del dividendo y del divisor en la misma base. El logaritmo de un cociente es igual al logaritmo del dividendo menos el logaritmo del divisor.

$$20H(S) = 20 \log(1.28) + 20 \log S + 20 \log \left(\frac{S}{9} + 1 \right) - 20 \log \left(\frac{S}{7} + 1 \right)$$

Donde:

$$20 \log(1.28) = 2.14$$

Continuando con la elaboración del Diagrama de Bode en Matlab, ingresamos nuevamente el código con la función margin cambiando las variables de nuestra función de transferencia como se muestra a continuación:

```
hd = tf([1 9 0],[1 7])
[Gm, Pm, Wgm, Wpm] = margin(hd)
margin(hd)
```

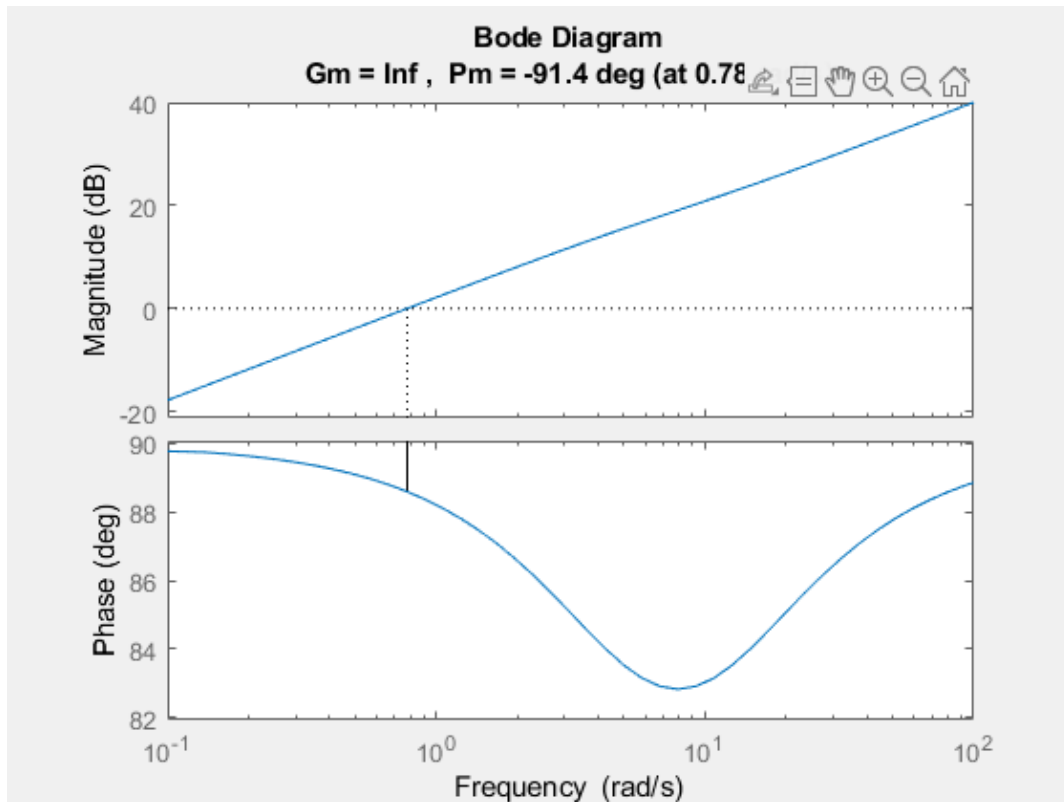
8.7 Sintaxis Func Tranf Ej. 3

Visualizamos en la Ventana Comand Window nuestra función de transferencia y los márgenes de estabilidad que son el margen de fase y el margen de ganancia.

<pre>hd = s^2 + 9 s ----- s + 7 Continuous-time transfer function.</pre>	<pre>Gm = Inf Pm = -91.4043 Wgm = NaN Wpm = 0.7797</pre>
a)	b)

8.8 Comand Window a) Función de transferencia b) Márgenes de ganancia

Obtenemos la siguiente gráfica:



8.9 Diagrama de Bode Función de Transferencia 3

Podemos demostrar el comportamiento de nuestro sistema respecto a las frecuencias, para la magnitud tomamos en cuenta la primera parte de la resolución con ayuda de las propiedades de los logaritmos en donde partimos de $20\log(1.28)=2.14$, en donde en este caso iniciaremos de forma positiva sumado a 20dB ya que cumple con $S=0$ que significa que hay un cero en origen, después vamos a ir realizando la suma de las pendientes en dB/D subiendo a razón de 20dB/D, cuando tenemos $S+9$ quiere decir que tenemos un polo en 9 y subirá como pendiente a razón de 20dB/D solo si ese polo es parte del numerador, en caso contrario, que sea del denominador descenderá a razón de -20dB/D.

Para la fase tenemos como primer término una constante por lo que debemos de trazar una línea recta en 0° , si ésta constante llegase a ser menor que 0 la debemos de trazar en 180° , para el segundo término tenemos un cero en el origen debido a que se trata de S así que existe una línea recta en 90° , para el tercer y cuarto elemento que corresponden a los polos deben de empezar en 0.9 y 0.7 ascendiendo o descendiendo según sea la polaridad de los mismos.

9. CONTROLADORES P, PI Y PID

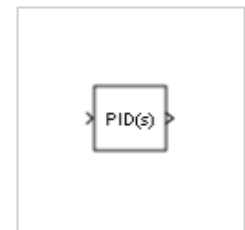
Un **sistema de control** es un conjunto de dispositivos encargados de administrar, ordenar, dirigir o regular el comportamiento de otro sistema, con el fin de reducir las probabilidades de fallo y obtener los resultados deseados.

La representación de los problemas en los sistemas de control se lleva a cabo mediante tres representaciones básicas o modelos:

1. Ecuaciones diferenciales, integrales, derivadas y otras relaciones matemáticas.
2. Diagramas en bloque.
3. Gráficas en flujo de análisis.

Los diagramas en bloque y las gráficas de flujo son representaciones gráficas que pretenden el acortamiento del proceso correctivo del sistema, sin importar si está caracterizado de manera esquemática o mediante ecuaciones matemáticas. Las ecuaciones diferenciales y otras relaciones matemáticas, se emplean cuando se requieren relaciones detalladas del sistema. Cada sistema de control se puede representar teóricamente por sus ecuaciones matemáticas. El uso de operaciones matemáticas es patente en todos los controladores de tipo Controlador proporcional (P), Controlador proporcional, Integral (PI) y Controlador proporcional, Integral y derivativo (PID), que debido a la combinación y superposición de cálculos matemáticos ayuda a controlar circuitos, montajes y sistemas industriales para así ayudar en el perfeccionamiento de los mismos.

El bloque del controlador PID implementa un controlador PID (PID, PI, PD, solo P o solo I). El bloque es idéntico al bloque Controlador PID discreto con el parámetro de **dominio de tiempo** establecido en Continuous-time.



La salida de bloque es una suma ponderada de la señal de entrada, la integral de la señal de entrada y la derivada de la señal de entrada. Los pesos son los parámetros de ganancia proporcional, integral y derivada. Un polo de primer orden filtra la acción derivada.

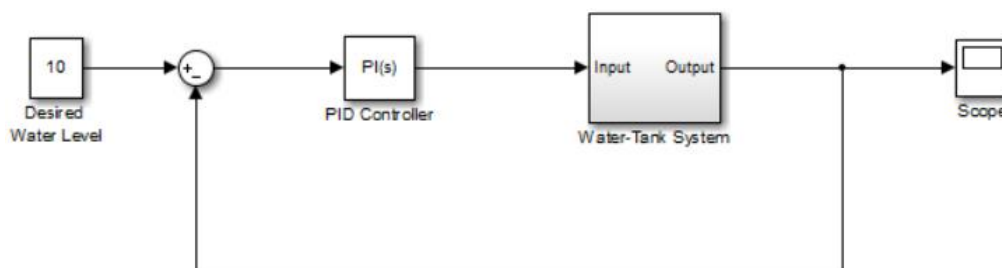
El bloque admite varios tipos y estructuras de controlador. Las opciones configurables en el bloque incluyen:

- Tipo de controlador (PID, PI, PD, solo P o solo I).
- Formulario de controlador (paralelo o ideal).
- Dominio de tiempo (continuo o discreto).
- Condiciones iniciales y reinicio del disparador.
- Límites de saturación de salida y mecanismo anti-enrollamiento incorporado.

A medida que cambia estas opciones, la estructura interna del bloque cambia al activar diferentes subsistemas variantes. Para examinar la estructura interna del bloque y sus subsistemas variantes, haga clic con el botón derecho en el bloque y seleccione **Máscara > Buscar debajo de Máscara**.

Configuración de Control

En una implementación común, el bloque controlador PID opera en la ruta de alimentación de un bucle de retroalimentación.



9.1 Diagrama de bloques con ruta de retroalimentación de Controlador PID

La entrada del bloque es típicamente una señal de error, que es la diferencia entre una señal de referencia y la salida del sistema.

Los coeficientes del controlador PID se pueden ajustar de forma manual o automática. La sintonización automática requiere el software Simulink.

Puertos

- Port_1(u) - Señal de error de entrada
escalar | vector

Cuando la señal de error es un vector, el bloque actúa por separado en cada señal, vectorizando los coeficientes PID y produciendo una señal de salida de vector de las mismas dimensiones. Puede especificar los coeficientes PID y algunos otros parámetros como vectores de las mismas dimensiones que la señal de entrada. Hacerlo es equivalente a especificar un controlador PID separado para cada entrada en la señal de entrada.

- P - Ganancia escalar proporcional
| vector

Ganancia proporcional, proporcionada desde una fuente externa al bloque. La entrada de ganancia externa es útil, por ejemplo, cuando desea asignar una parametrización PID diferente a las ganancias PID del bloque. También puede usar la entrada de ganancia externa para implementar el control PID programado por ganancia. En el control de ganancia programada, usted determina los coeficientes PID por lógica u otro cálculo en su modelo y los alimenta al bloque.

- I - Ganancia integral
escalar | vector

Ganancia integral, proporcionada desde una fuente externa al bloque. La entrada de ganancia externa es útil, por ejemplo, cuando desea asignar una parametrización PID diferente a las ganancias PID del bloque. También puede usar la entrada de ganancia externa para implementar el control PID programado por ganancia. En el control de ganancia programada, usted determina los coeficientes PID por lógica u otro cálculo en su modelo y los alimenta al bloque.

Cuando suministra ganancias externamente, también se integran las variaciones de tiempo en la ganancia integral. Este resultado ocurre debido a la forma en que se implementan las ganancias PID dentro del bloque. Para obtener detalles, consulte los **parámetros del controlador Parámetro de origen**.

- D- Ganancia derivada
escalar | vector

Ganancia derivada, proporcionada desde una fuente externa al bloque. La entrada de ganancia externa es útil, por ejemplo, cuando desea asignar una parametrización PID diferente a las ganancias PID del bloque. También puede usar la entrada de ganancia externa para implementar el control PID programado por ganancia. En el control de ganancia programada, usted determina los coeficientes PID por lógica u otro cálculo en su modelo y los alimenta al bloque.

Cuando suministra ganancias externamente, las variaciones de tiempo en la ganancia derivada también se diferencian. Este resultado ocurre debido a la forma en que se implementan las ganancias PID dentro del bloque. Para obtener detalles, consulte los **parámetros del controlador Parámetro de origen**.

- N- Coeficiente de filtro
escalar | vector

Coeficiente de filtro derivado, proporcionado desde una fuente externa al bloque. La entrada de coeficiente externo es útil, por ejemplo, cuando desea asignar una parametrización PID diferente a las ganancias PID del bloque. También puede usar la entrada externa para implementar el control PID programado por ganancia. En el control de ganancia programada, usted determina los coeficientes PID por lógica u otro cálculo en su modelo y los alimenta al bloque.

- Reset- Restablecimiento externo disparador
escalar

Disparador para restablecer el integrador y filtrar a sus condiciones iniciales. El valor del parámetro de **reinicio externo** determina si el reinicio se produce en una señal ascendente, una señal descendente o una señal de nivel. El icono del puerto indica el tipo de disparador seleccionado. Por ejemplo, la siguiente ilustración muestra un bloque PID de tiempo continuo con **restablecimiento externo** establecido en rising.

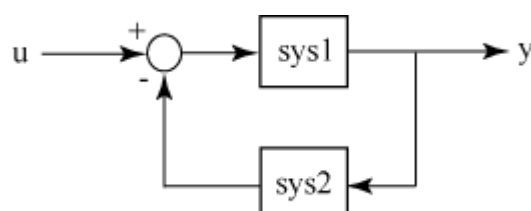


9.2 Bloque PID de tiempo continuo con restablecimiento externo

Ejemplo 1. Controlador Proporcional

Para este caso utilizaremos en la sintaxis que introduciremos en Matlab el comando *feedback* (*sys1, sys2*) que devuelve un objeto modelo *sys* para la interconexión de retroalimentación negativa de los objetos modelo *sys1, sys2* como se muestra en el siguiente diagrama de bloques.

La retroalimentación en el diseño un controlador es fundamental ya que los resultados obtenidos de una tarea o actividad son reintroducidos en el sistema con la finalidad de incidir o actuar sobre las decisiones o acciones futuras, ya sea para mantener el equilibrio en el sistema o para conducir el sistema hacia uno nuevo.



9.3 Modelo de realimentación

Introduciendo el siguiente código en nuestro software Matlab, en donde ubicaremos a “f” igual a la función de transferencia, así como el comando *step*, el cual calculará la respuesta de nuestra sistema dinámico, *kp* como variable que demuestra la ganancia proporcional y por último $C = pid(Kp, Ki, Kd, Tf)$ que creará un controlador PID de tiempo continuo con proporcional, integral, y ganancias de derivadas K_p , K_i y K_d y de primer orden del filtro derivado constante de tiempo T_f :

$$C = K_p + \frac{K_i}{s} + \frac{K_d s}{T_f s + 1}.$$

Para este caso K_i y K_d serán igual a cero.

El código para nuestro controlador se introducirá de la siguiente forma:

```

Untitled.m x g.m x +
1 - f = tf (5, [96 0.15])
2 - step (f)
3 - kp=1;
4 - c=pid (kp,0,0);
5 - sys=feedback (c*f,1);
6 - step(sys)

```

9.4 Código Controlador proporcional

Visualizamos en la Ventana Comand Window nuestra función de transferencia

```

>> Untitled

f =

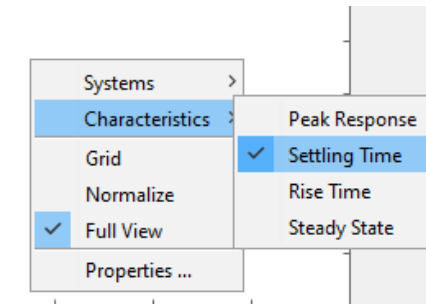
      5
-----
 96 s + 0.15

Continuous-time transfer function.

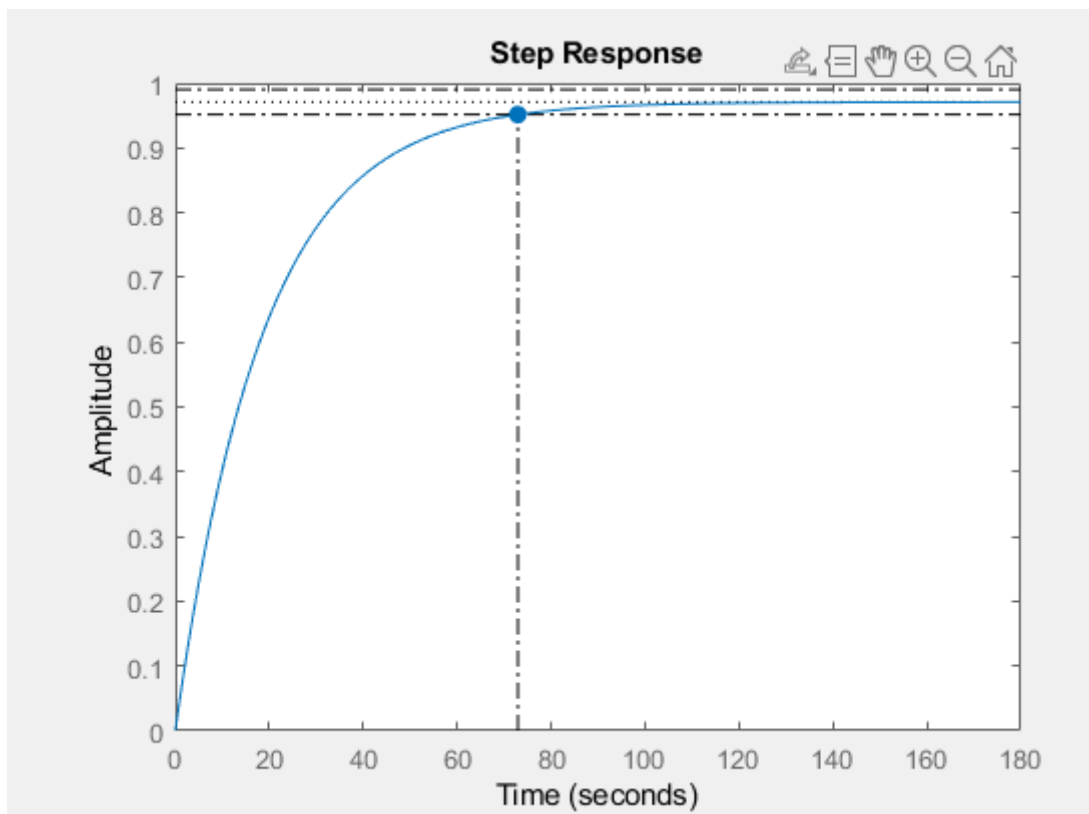
```

9.5 Función de transferencia

Obtenemos la siguiente gráfica que nos muestra la amplitud y el tiempo de estabilización del sistema utilizando la función *setting time* podemos visualizar el punto donde la onda tiene un comportamiento estable en donde para $k=1$ es de aproximadamente 72 segundos.



9.6 Función *setting Time*



9.7 Gráfica controlador P- $K_p=1$

Comprobando las características del controlador P, como que es un amplificador con ganancia ajustable, reduce el tiempo de subida, reduce el error de estado estable y acelera la respuesta de los procesos controlados modificamos k_p asignándole ahora un valor de 5 obtenemos el siguiente resultado:

```
f = tf (5, [96 0.15])
step (f)
kp=5;
c=pid (kp,0,0);
```

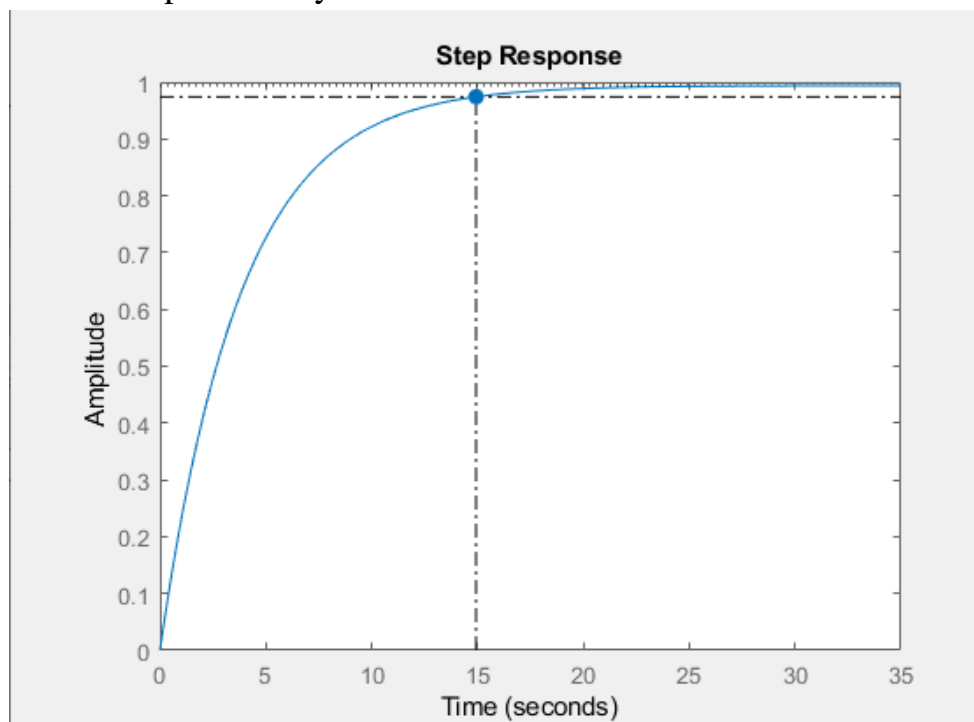
9.8 Código de Controlador PI - $K_p=5$

```
f =
      5
-----
 96 s + 0.15

Continuous-time transfer function.
```

9.9 Función de transferencia Controlador P

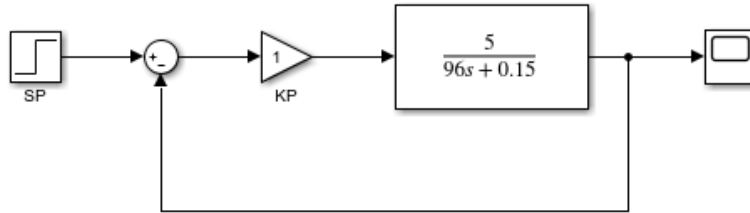
Como podemos observar en la siguiente gráfica el tiempo de respuesta se redujo de 72 a 15 segundos que es alrededor de 5 veces justo como lo introducimos en la ganancia proporcional de nuestro sistema y mientras sigamos aumentando la misma, puede reducirse aún más, sin embargo el inconveniente de este caso es que de manera práctica esta opción tiene un precio muy elevado.



9.10 Gráfica controlador P- $K_p=5$

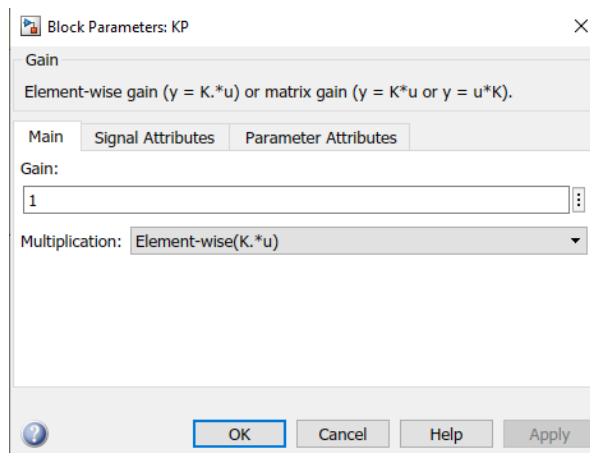
Modelado de Sistema de Control Proporcional en Simulink

Para realizar la configuración del modelo de nuestro sistema en Simulink realizamos el siguiente diagrama de bloques siguiendo las instrucciones del capítulo 3, como en este caso solo tenemos un sistema utilizaremos un método de ingreso de bloques independientes, ingresamos los *bloques step*, *sum*, *kp*, *transfer function* y *scope*.



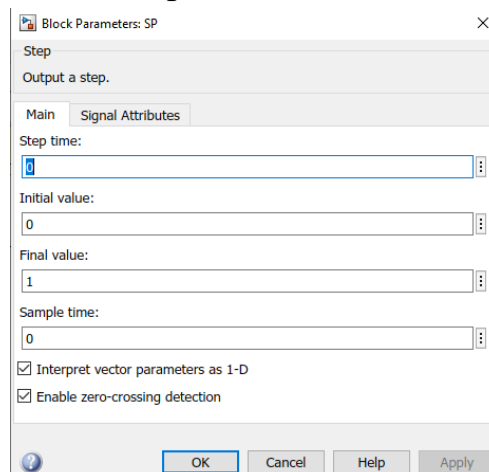
9.11 Modelado de Diagrama de Bloques Sistema de Control Proporcional

Configuramos la ganancia proporcional de nuestro bloque Kp con un valor de 1 para este primer caso.



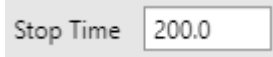
9.12 Configuración de ganancia proporcional

De igual forma configuramos el tiempo de inicio de nuestro bloque *step*



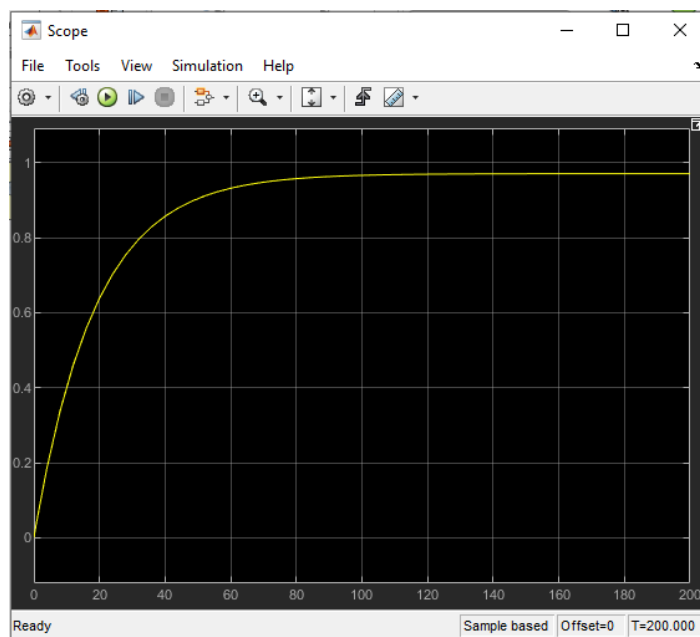
9.13 Configuración de bloque Step

De acuerdo a nuestra gráfica antes realizada en Matlab sabemos que el tiempo de estabilización de la onda es de aproximadamente 72 segundos por lo que asignamos en la barra de tiempo de paro un tiempo de 200 segundos para poder visualizar el comportamiento del sistema completo.



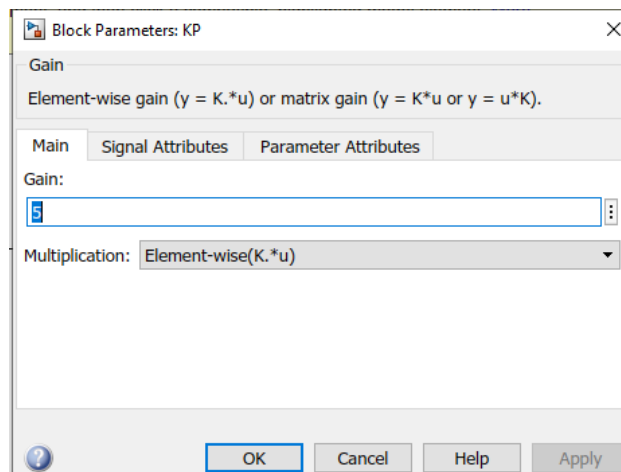
9.14 Tiempo de paro

Obtenemos la gráfica siguiente en donde confirmamos que es su comportamiento es el mismo que en la parte de Matlab



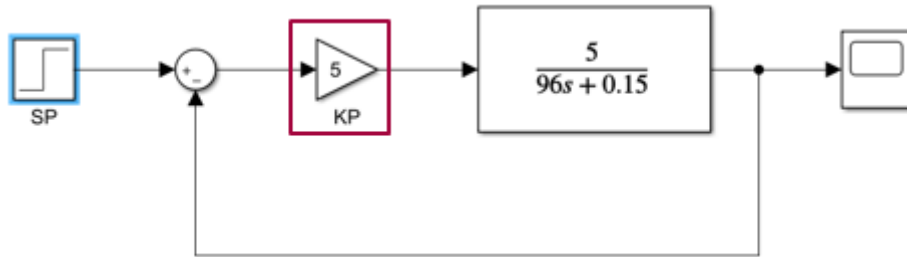
9.15 Gráfica controlador P- $K_p=1$ (Simulink)

De igual forma que en el caso anterior confirmamos las características del Controlador Proporcional aumentando la ganancia 5 veces, esto lo realizamos en el bloque Kp como se muestra a continuación:



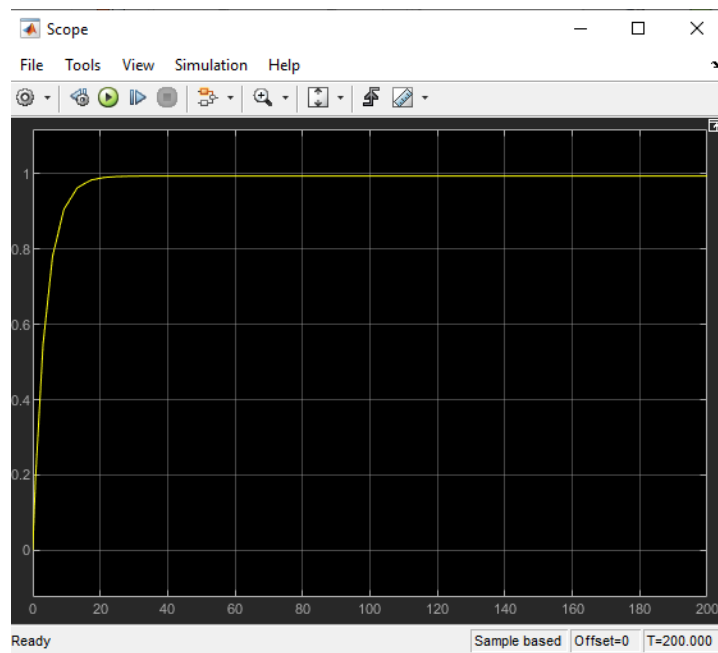
9.16 Configuración de ganancia bloque Kp

El cambio en el diagrama de bloques de nuestro sistema se puede observar de la siguiente forma:



9.17 Modelado de Diagrama de Bloques Sistema de Control Proporcional $k_p=5$

Aplicamos el mismo tiempo de paro que en caso anterior atendido a 200 segundos, podemos visualizar en la siguiente gráfica la reducción del tiempo de estabilización de la onda a 15 segundos confirmando las características antes mencionadas de nuestro controlador.



9.18 Gráfica controlador P- $K_p=5$ (Simulink)

Ejemplo 2. Controlador Proporcional Integral

Para el ejemplo del controlador PI, vamos trabajar con la función de transferencia $\frac{1}{9s^2+9s+2}$, la cual sigue el mismo modelo que se ha establecido en el ejercicio anterior. Para ello comenzaremos colocando los comandos, *clc*, *close all* y *clear all*: borrar la ventana de comandos, cerrar todas las figuras y borrar espacio de trabajo respectivamente, el único punto de estas funciones es guardar las pulsaciones de teclas, al usar Matlab con frecuencia reduce el tiempo entre operaciones y manejo de simulaciones.

Indicaremos la función de transferencia del sistema, siendo $s = tf('s')$ el comando para asignarla, para después ingresar los valores del sistema con el comando *sys*, los estableceremos de la siguiente manera: $1/(3*s+1)/(3*s+2)$, con esto obtendremos la función de transferencia ya antes mencionada.

Agregaremos el comando *step* con los valores de (1,1), el cual nos va permitir que nos muestre el estado de referencia, en el cual el sistema alcanzará la estabilidad. Además, con el comando *hold on* podremos mantener las figuras creadas sin que se borren.

Con el comando *axis* podremos establecer, los valores o límites de los ejes de nuestras gráficas en las cuales vamos a estar trabajando, que es este caso sería 20 respecto al tiempo y 1.5 para la amplitud. Agregaremos de nuevo un comando *step* con el cual vamos a pedir que nos muestre la figura del sistema. Ahora vamos a comenzar a indicar los valores para nuestro controlador, para esto vamos a establecerlo como *c1* en el cual le daremos el valor con el comando *pid* donde tendremos que ingresar entre paréntesis los valores del proporcional y del integral, separados por una coma, si se agrega un valor extra, se convertiría en un controlador PID o en el caso de solo agregar un valor, lo tomaría como un controlador P. Para este ejercicio, tenemos que el valor de $K_p = 3$ y $K_i = 0.8$.

Ingresaremos un comando de *feedback* que como anteriormente mencionamos, en el diseño un controlador es fundamental ya que los resultados obtenidos de una tarea o actividad son reintroducidos en el sistema con la finalidad de incidir o actuar sobre las decisiones o acciones futuras, en el cual vamos a indicar que se va a dar la retroalimentación del PI y del sistema como se muestra en la figura ().

Por último, vamos a ingresar un comando para poder nombrar a los elementos en la gráfica llamado *legend* el cual será escrito de manera en que se fueron ingresando cada uno de los elementos de nuestro sistema, en este caso ingresamos en primera instancia el valor de referencia, después el valor del Open Loop y por último el valor de nuestro controlador PI.


```

1 - clc
2 - close all
3 - clear all
4
5 - s = tf('s');
6 - sys = 1 / (3*s+1) / (3*s+2)
7
8 - step(1,1); hold on;
9 - axis([0 20 0 1.5]);
10 - step(sys);
11 - c1 = pid(3, 0.8)
12 - step(feedback(c1*ss(sys),1));
13 - legend('ref', 'open loop', 'pi');
14
15

```

9.19 Código Controlador PI

Continuaremos presionando el botón de iniciar para así Matlab lea nuestros comandos programados y nos mostrará la siguiente información en la tabla de *Command Window*, en donde podremos ver la función de transferencia que establecimos de manera correcta y además los valores de nuestro controlador con nuestra respectiva ecuación.

```

Command Window

sys =

      1
-----
 9 s^2 + 9 s + 2

Continuous-time transfer function.

c1 =

      Kp + Ki * ---
                s

with Kp = 3, Ki = 0.8

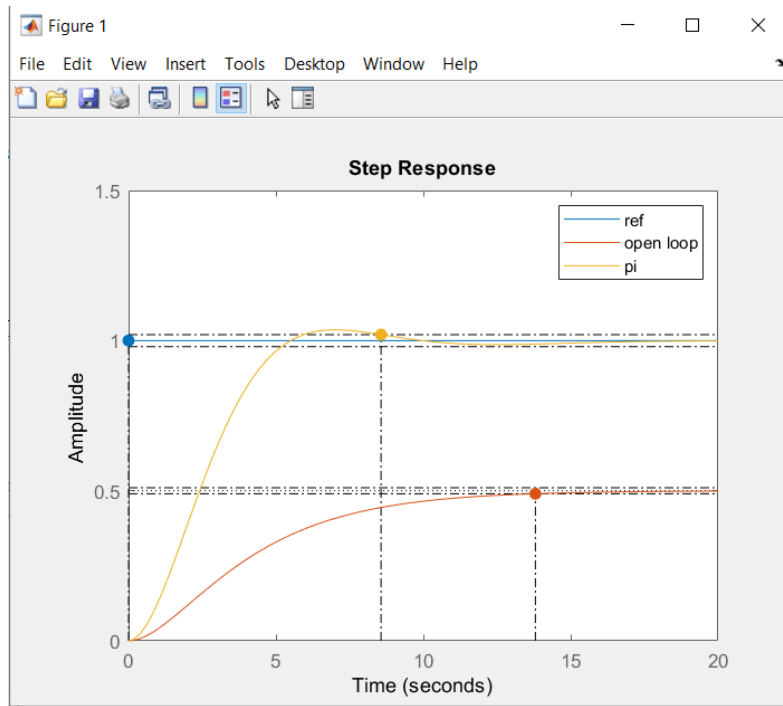
Continuous-time PI controller in parallel form.

fx >>

```

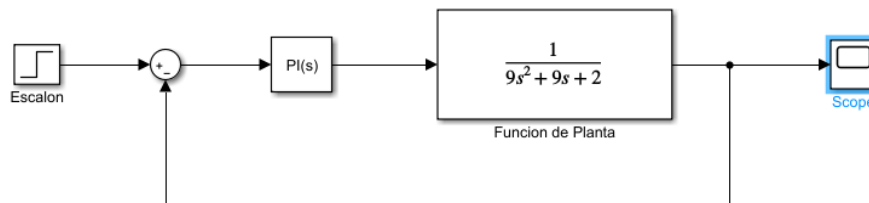
9.20 Función de transferencia controlador PI

Después de cargar estos datos, de manera automática nos va a arrojar las figuras que creamos con los comandos, esto con una gráfica con las leyendas que de igual manera establecimos, vamos a dar *clic derecho sobre la gráfica* > *Characteristics* > *Settling Time*, y nos va a mostrar el tiempo exacto que tarda en estabilizarse nuestro sistema en la referencia establecida. En este caso, se estabilizó al llegar a los 8.57 s.



9.21 Gráfica Controlador PI con estabilización a 8.57s

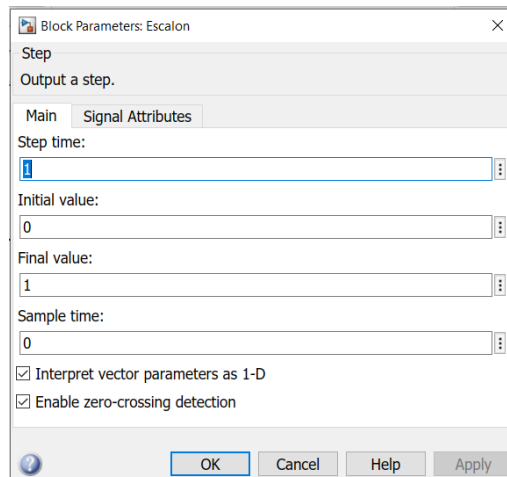
Ahora, nos iremos a la pestaña de Home y vamos a inicializar el programa de Simulink, en el cual comenzaremos a trazar nuestro diagrama de bloques, donde podremos representar de forma gráfica los sistemas y dar lugar a lo que se conoce como estructura de control.



9.22 Diagrama de Bloques PI

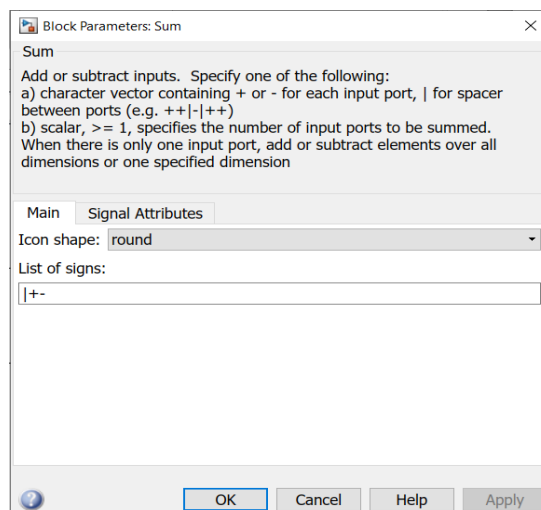
Comenzaremos a establecer los valores de cada uno de los bloques de la siguiente manera, daremos doble clic en cada uno de los bloques y dentro de la pantalla de configuración, vamos a ingresar los datos que tenemos de nuestro sistema.

Comenzaremos con el bloque de step o escalón.



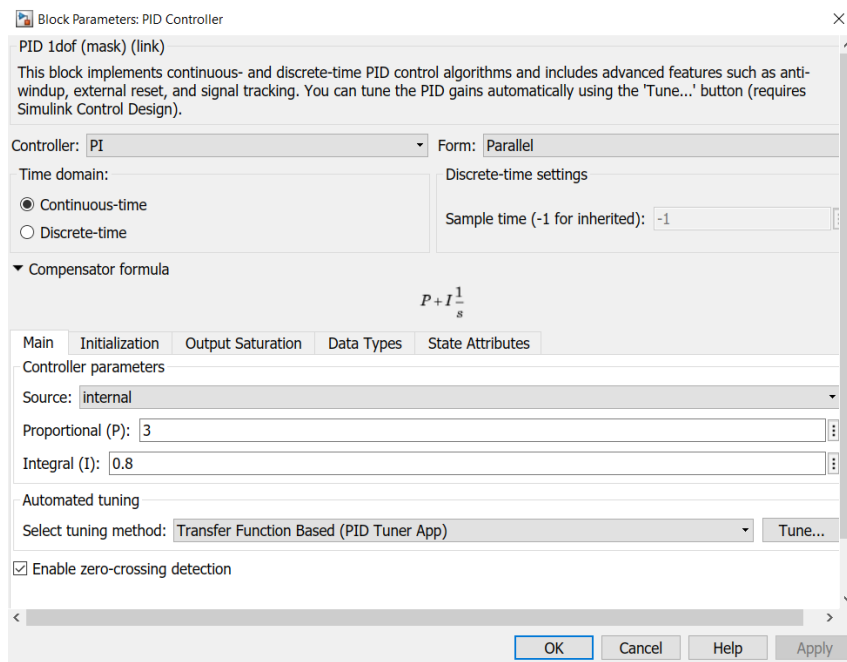
9.23 Configuración Step

Parámetros de bloque sum.



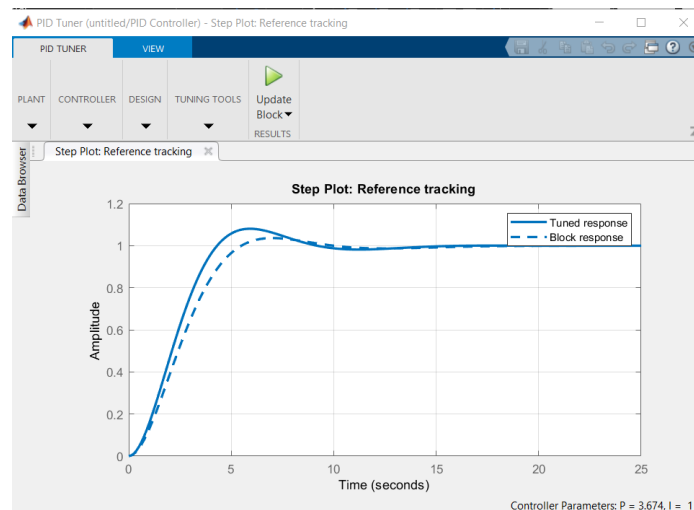
9.24 Configuración de bloque Sum

Dentro de los parámetros del bloque PI, podremos ver los valores tanto de K_p y K_i , además podremos visualizar dentro de una función de Simulink llamada *PID Tuner* presionando el botón de tuner en esta misma pantalla.



9.25 Configuración PID Tuner

En ella podremos controlar y visualizar los valores de tiempo de respuesta y comportamiento transitorio, para así poder llegar a la estabilidad de manera mas eficiente y nos ofrece además una tabla de parámetros exactos del controlador.



9.26 Gráfica de tiempo de respuesta y comportamiento trasitorio

Controller Parameters		
	Tuned	Block
P	3.6742	3
I	1	0.8
D	n/a	n/a
N	n/a	n/a

Performance and Robustness		
	Tuned	Block
Rise time	2.87 seconds	3.53 seconds
Settling time	8.36 seconds	8.57 seconds
Overshoot	8.04 %	3.6 %
Peak	1.08	1.04
Gain margin	Inf dB @ Inf rad/s	Inf dB @ Inf rad/s
Phase margin	60 deg @ 0.471 rad/s	65.4 deg @ 0.396 rad/s
Closed-loop stability	Stable	Stable

9.27 Parámetros del controlador

Continuaremos estableciendo los parámetros de los bloques, ahora con el de función de transferencia.

Block Parameters: Funcion de Planta

Transfer Fcn

The numerator coefficient can be a vector or matrix expression. The denominator coefficient must be a vector. The output width equals the number of rows in the numerator coefficient. You should specify the coefficients in descending order of powers of s.

Parameters

Numerator coefficients: [1]

Denominator coefficients: [9 9 2]

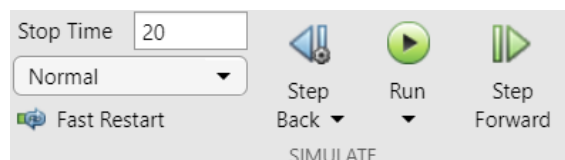
Absolute tolerance: auto

State Name: (e.g., 'position')

OK Cancel Help Apply

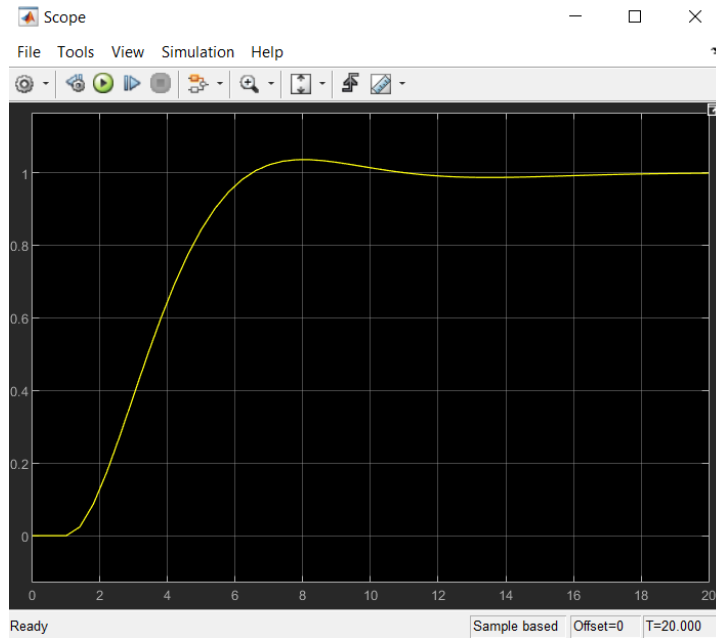
9.28 Configuración del Bloque de Parámetros de Función de Transferencia

Ingresaremos el valor del tiempo de simulación que, en este caso, tomaremos los parámetros o imites ya establecidos de 20 s.



9.29 Tiempo de Paro de simulación

Procedemos a correr la simulación y posteriormente daremos doble clic en el bloque de Scope para que podamos visualizar la gráfica, daremos clic nuevamente en correr dentro de esta ventana y nos va a arrojar la gráfica con todos los parámetros que ya anteriormente establecidos.



9.30 Gráfica final de comportamiento con parámetros establecidos

Ejemplo 3. Controlador Proporcional Integral Derivativo

Para el siguiente ejemplo del controlador PID, vamos trabajar con la misma función de transferencia $\frac{1}{9s^2+9s+2}$ del ejemplo anterior, mostrando así las diferencias entre las aplicaciones de un controlador P, PI y PID y así mostrar cómo trabaja el controlador PID respecto a los demás. Para ello comenzaremos colocando los comandos, *clc*, *close all* y *clear all*.

Indicaremos la función de transferencia del sistema, siendo $s = tf('s')$ el comando para asignarla, para después ingresar los valores del sistema con el comando *sys*, los estableceremos de la siguiente manera: $1 / (3*s+1) / (3*s+2)$, con esto obtendremos la función de transferencia ya antes mencionada.

Agregaremos el comando *step* con los valores de (1,1), el cual nos va permitir que nos muestre el estado de referencia, en el cual el sistema alcanzará la estabilidad. Además, con el comando *hold on* podremos mantener las figuras creadas sin que se borren.

Con el comando *axis* podremos establecer, los valores o límites de los ejes de nuestras gráficas en las cuales vamos a estar trabajando, que es este caso sería 20 respecto al tiempo y 1.5 para la amplitud. Agregaremos de nuevo un comando *step* con el cual

vamos a pedir que nos muestre la figura del sistema. Ahora vamos a comenzar a indicar los valores para nuestro controlador, para esto vamos a establecerlo como *c1* en el cual le daremos el valor con el comando *pid* donde tendremos que ingresar entre paréntesis los valores del proporcional, del integral y derivativo, separados por una coma, convirtiéndolo así en un controlador PID. Para este ejercicio, tenemos que el valor de $K_p = 3$, $K_i = 0.8$ y $K_d = 1.5$.

Ingresaremos un comando de *feedback* que como anteriormente mencionamos, en el diseño un controlador es fundamental ya que los resultados obtenidos de una tarea o actividad son reintroducidos en el sistema con la finalidad de incidir o actuar sobre las decisiones o acciones futuras, en el cual vamos a indicar que se va a dar la retroalimentación del PI y del sistema como se muestra en la figura ().

Por último, vamos a ingresar un comando para poder nombrar a los elementos en la gráfica llamado *legend* el cual será escrito de manera en que se fueron ingresando cada uno de los elementos de nuestro sistema, en este caso ingresamos en primera instancia el valor de referencia, después el valor del Open Loop, P, PI y por último el valor de nuestro controlador PID. Para así visualizar cada una de las figuras de los diferentes controladores y comparar los resultados de tiempo de respuesta, comportamiento transitorio y estabilización del sistema.

```
1 -   clc
2 -   close all
3 -   clear all
4
5 -   s = tf('s');
6 -   sys = 1 / (3*s+1) / (3*s+2)
7
8 -   step(1,1);
9 -   hold on;
10 -  axis([0 20 0 1.5]);
11 -  step(sys);
12 -  c = pid(3)
13 -  step(feedback(c*ss(sys),1))
14 -  c1 = pid(3, 0.8)
15 -  step(feedback(c1*ss(sys),1));
16 -  c2 = pid(3, 0.8, 1.5)
17 -  step(feedback(c2*ss(sys),1));
18 -  legend('ref', 'open loop', 'P', 'PI', 'PID');
```

9.31 Código Controlador PID

Continuaremos presionando el botón de iniciar para así Matlab lea nuestros comandos programados y nos mostrará la siguiente información en la tabla de *Command Window*, en donde podremos ver la función de transferencia que establecimos de manera correcta y además los valores de nuestro controlador con nuestra respectiva ecuación.

```

Command Window

c =

    Kp = 3

P-only controller.

c1 =

      1
    Kp + Ki * ----
              s

with Kp = 3, Ki = 0.8

Continuous-time PI controller in parallel form.

c2 =

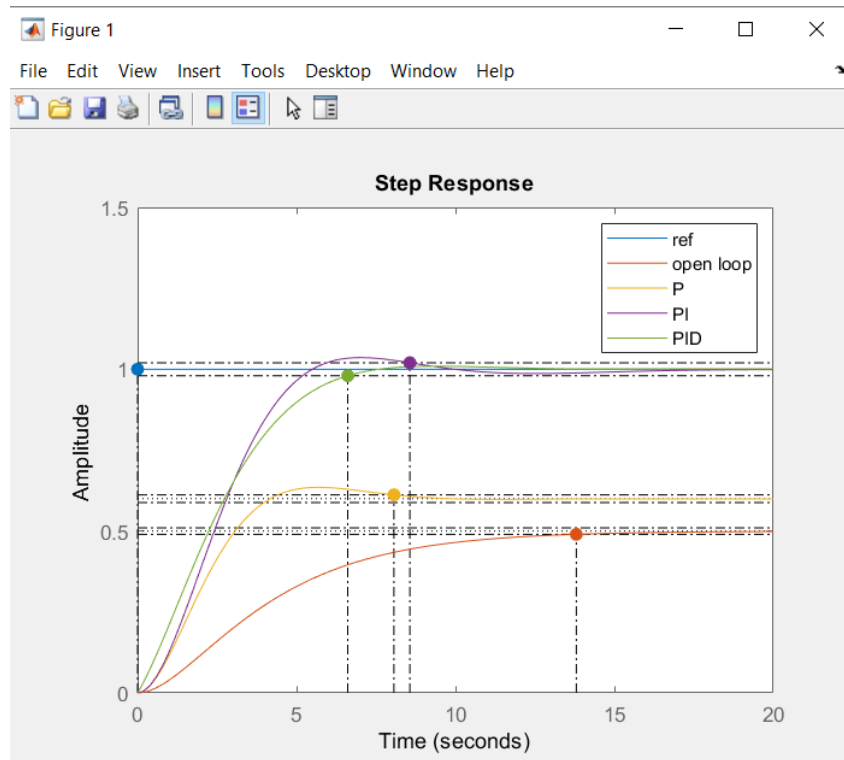
      1
    Kp + Ki * ---- + Kd * s
              s

with Kp = 3, Ki = 0.8, Kd = 1.5

```

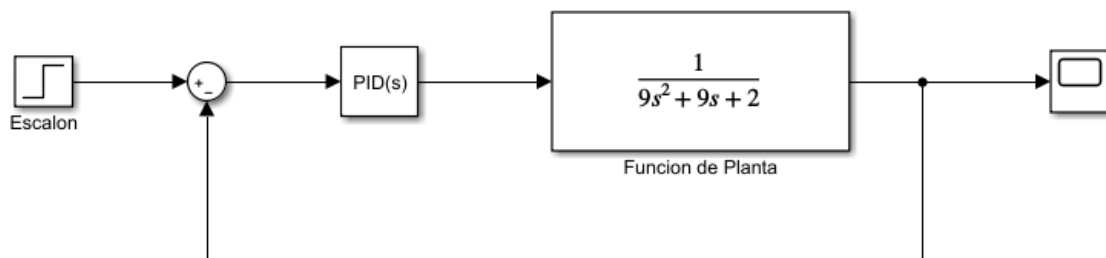
9.32 Función de transferencia controlador PID

Después de cargar estos datos, de manera automática nos va a arrojar las figuras que creamos con los comandos, esto con una gráfica con las leyendas que de igual manera establecimos, vamos a dar *clic derecho sobre la gráfica* > *Characteristics* > *Settling Time*, y nos va a mostrar el tiempo exacto que tarda en estabilizarse nuestro sistema en la referencia establecida. En este caso, se estabilizó al llegar a los 6.61 s, con lo que podemos concluir que al agregar el parámetro derivativo, se reduce de manera eficiente el tiempo de respuesta y consigue estabilizarse el sistema mejor que en los controladores P y PI.



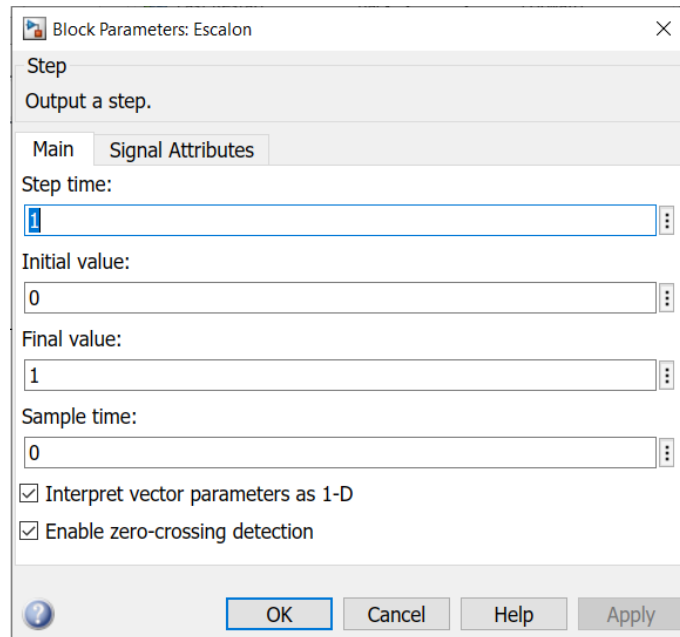
9.33 Gráfica de comportamiento controlador PID y visualización de mejoramiento en tiempo de respuesta y estabilización de forma de onda

Ahora, nos iremos a la pestaña de Home y vamos a inicializar el programa de Simulink, en el cual comenzaremos a trazar nuestro diagrama de bloques, donde podremos representar de forma gráfica los sistemas y dar lugar a lo que se conoce como estructura de control



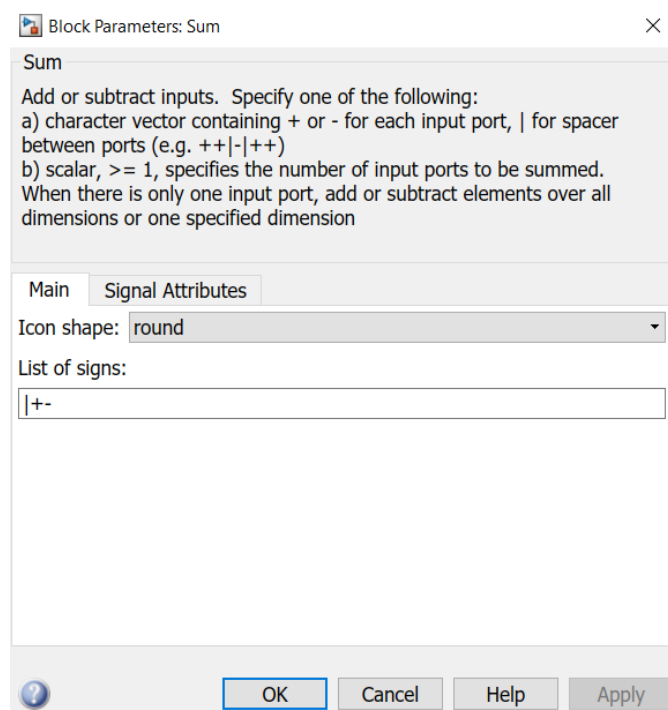
9.34 Diagrama de bloques Controlador PID

Comenzaremos a establecer los valores de cada uno de los bloques de la siguiente manera, daremos doble clic en cada uno de los bloques y dentro de la pantalla de configuración, vamos a ingresar los datos que tenemos de nuestro sistema. Comenzaremos con el bloque de step o escalón.



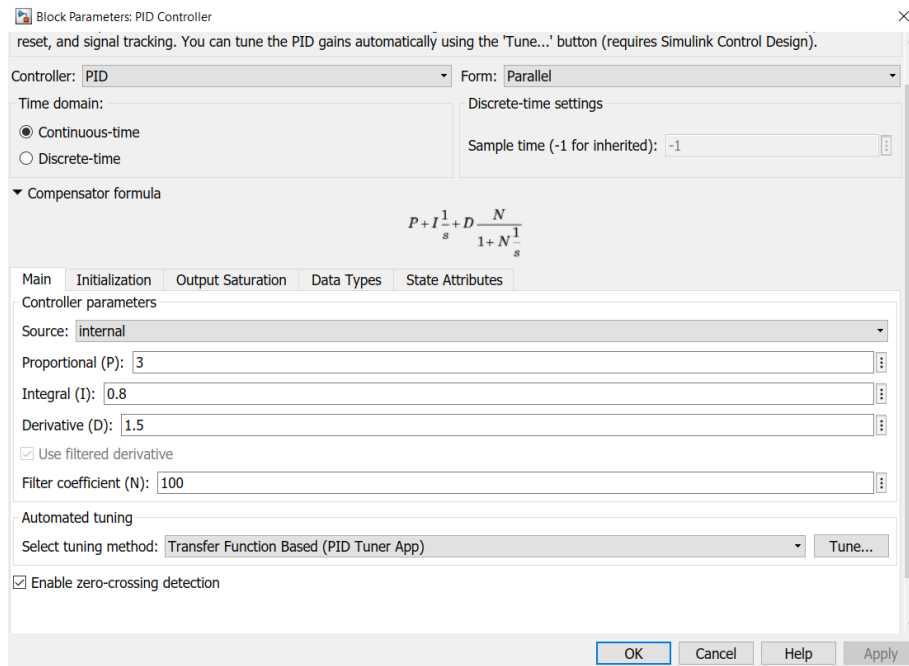
9.35 Configuración Step

Parámetros de bloque sum.



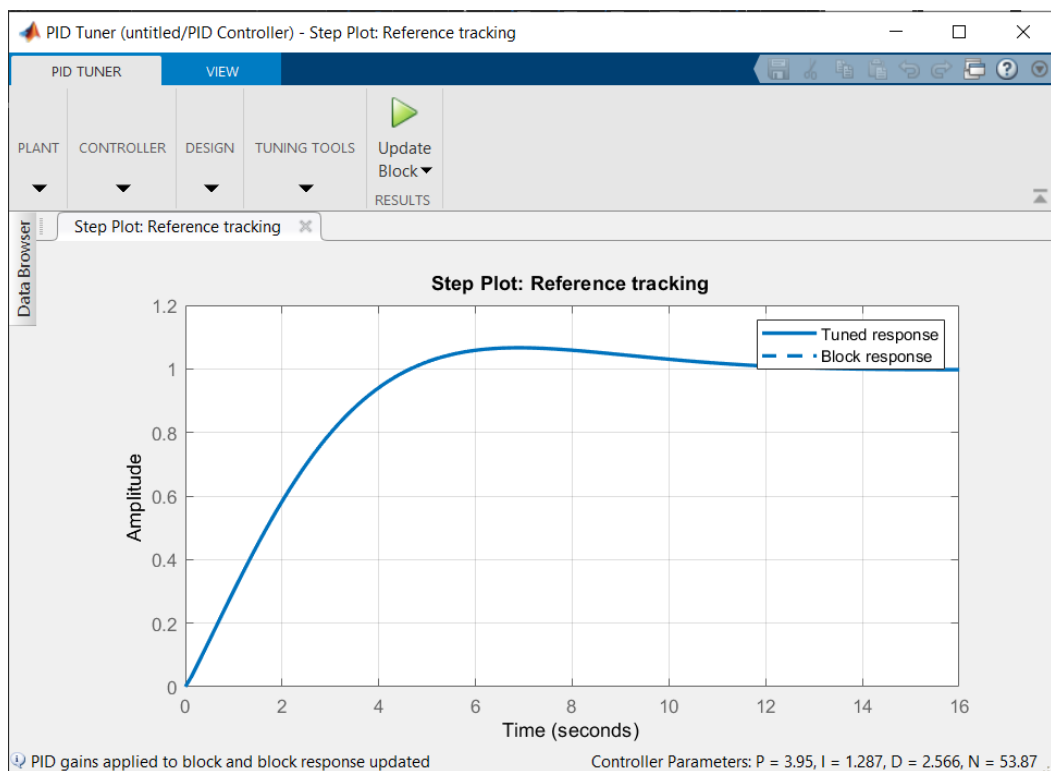
9.36 Configuración Sum

Dentro de los parámetros del bloque PI, podremos ver los valores tanto de K_p y K_i , además podremos visualizar dentro de una función de Simulink llamada *PID Tuner* presionando el botón de tuner en esta misma pantalla.



9.37 Configuración PID Tuner

En ella podremos controlar y visualizar los valores de tiempo de respuesta y comportamiento transitorio, para así poder llegar a la estabilidad de manera más eficiente y nos ofrece además una tabla de parámetros exactos del controlador.



9.38 Valores de tiempo de respuesta y comportamiento transitorio

Controller Parameters		
	Tuned	Block
P	3.9501	3.9501
I	1.287	1.287
D	2.5655	2.5655
N	53.8734	53.8734

Performance and Robustness		
	Tuned	Block
Rise time	3.31 seconds	3.31 seconds
Settling time	10.8 seconds	10.8 seconds
Overshoot	6.65 %	6.65 %
Peak	1.07	1.07
Gain margin	Inf dB @ Inf rad/s	Inf dB @ Inf rad/s
Phase margin	69 deg @ 0.471 rad/s	69 deg @ 0.471 rad/s
Closed-loop stability	Stable	Stable

9.39 Parámetros de Controlador

Continuaremos estableciendo los parámetros de los bloques, ahora con el de función de transferencia.

Block Parameters: Funcion de Planta

Transfer Fcn

The numerator coefficient can be a vector or matrix expression. The denominator coefficient must be a vector. The output width equals the number of rows in the numerator coefficient. You should specify the coefficients in descending order of powers of s.

Parameters

Numerator coefficients: [1]

Denominator coefficients: [9 9 2]

Absolute tolerance: auto

State Name: (e.g., 'position')

OK Cancel Help Apply

9.40 Configuración de parámetros de Función de Transferencia

Ingresaremos el valor del tiempo de simulación que, en este caso, tomaremos los parámetros o imites ya establecidos de 20 s.

Stop Time 20

Normal

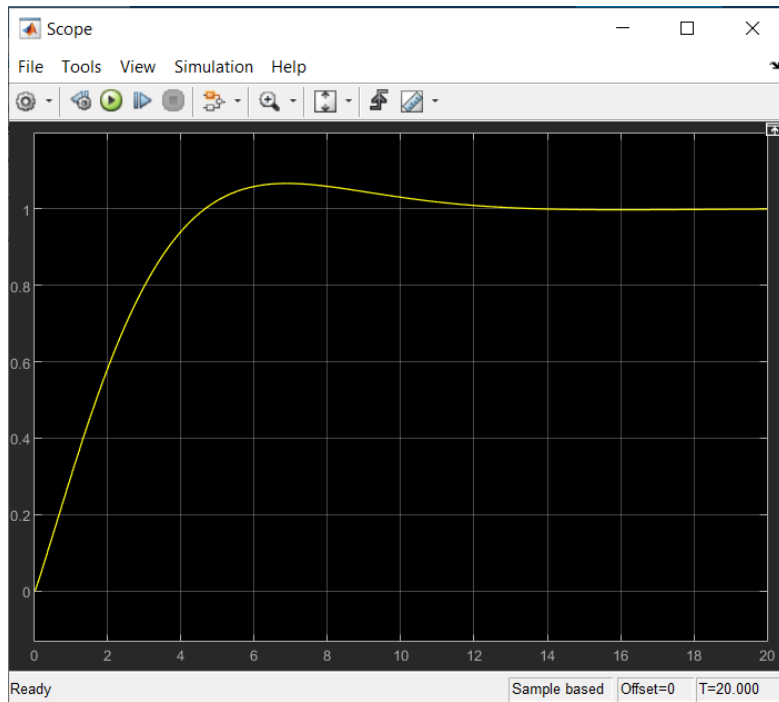
Fast Restart

Step Back Run Step Forward

SIMULATE

9.41 Tiempo de paro de simulación

Procedemos a correr la simulación y posteriormente daremos doble clic en el bloque de Scope para que podamos visualizar la gráfica, daremos clic nuevamente en correr dentro de esta ventana y nos va a arrojar la gráfica con todos los parámetros que ya anteriormente establecidos.



9.42 Gráfica final con parámetros establecidos para Controlador PID

10. ESTABILIDAD

CRITERIO DE ROUTH – HURWITZ.

Este criterio es usado en sistemas de tiempo continuo para determinar si el denominador de la función de transferencia tiene raíces en el semiplano derecho del plano s . Si este criterio es aplicado a la ecuación característica de un sistema de tiempo discreto expresado en la variable z , no puede obtenerse ninguna información sobre la estabilidad del mismo. Sin embargo, si la ecuación característica es expresada como una función de la variable de transformación bilineal w , entonces la estabilidad del sistema puede ser determinada por aplicación directa del criterio de Routh–Hurwitz, de la misma manera que se aplica a los sistemas de tiempo continuo. Esto es posible, pues a través de la transformación w el interior del círculo unitario del plano z se transforma en el semiplano izquierdo del plano w , y el exterior del círculo unitario en el semiplano derecho w .

Ejemplo:

Sea una planta de segundo orden de función de transferencia dada por:

$$G(s) = \frac{1}{s(s+1)},$$

El sistema de control de tiempo continuo (figura 4.1), es estable para todo valor positivo de la ganancia K . En efecto, si se aplica el criterio de Routh–Hurwitz a su ecuación característica:

$$F(s) = 1 + GH(s) = s^2 + s + K$$

resulta:

$$\begin{array}{r} s^2 \quad 1 \quad K \\ s^1 \quad 1 \\ s^0 \quad K \end{array}$$

El número de cambios de signo de la primer fila del arreglo de Routh–Hurwitz indica el número de raíces del polinomio característico $F(s)$ ubicadas en el semiplano derecho. En este caso no existen cambios de signo de modo que el sistema de tiempo continuo realimentado en forma unitaria es estable para todo valor positivo de la ganancia K . La función de transferencia entrada–salida para el sistema equivalente de tiempo discreto, para un período de muestreo $T = 0.1$ seg. Está dada por

$$G(z) = K \frac{0.00484 z + 0.00468}{(z-1)(z-0.905)}; \quad T = 0.1 \text{ seg}$$

Para aplicar el criterio de Routh–Hurwitz es necesario aplicar previamente la transformación bilineal

$$G(w) = K \frac{-0.00016w^2 - 0.1872w + 3.81}{3.81w^2 + 3.80w}.$$

Luego, la ecuación característica en el plano w está dada por:

$$F(w) = 1 + G(w) = (3.81 - 0.00016K) w^2 + (3.80 - 0.1872K) w + 3.81K.$$

La aplicación del criterio de Routh–Hurwitz conduce al siguiente arreglo:

$$\begin{array}{r} w^2 \quad 3.81 - 0.00016K \quad 3.81K \\ w^1 \quad 3.80 - 0.1872K \\ w^0 \quad 3.81K \end{array}$$

Para que el sistema sea estable no debe haber cambios de signos en la primera columna del arreglo, siendo necesario que la ganancia K se halle comprendida en el rango:

$$0 < K < 20.3.$$

Si se elige, en cambio, un período de muestreo mayor $T = 1$ seg., se obtiene

$$G(z) = K \frac{0.368z + 0.264}{(z-1)(z-0.368)}; \quad T = 1 \text{ seg.}$$

La expresión anterior muestra que al aumentar el período de muestreo el polo de más alta frecuencia se desplaza hacia el origen del plano z . Aplicando la transformación bilineal, la ecuación característica en el plano w resulta:

$$F(w) = (1 - 0.0381K) w^2 + (0.924 - 0.386K) w + 0.924K = 0,$$

y la aplicación del criterio de Routh–Hurwitz conduce al siguiente arreglo:

$$\begin{array}{r} w^2 \quad 1 - 0.0381K \quad 0.924K \\ w^1 \quad 0.924 - 0.386K \\ w^0 \quad 0.924K \end{array}$$

Examinando la primera columna del arreglo se observa que para que el sistema sea estable la ganancia K debe hallarse comprendida en el rango:

$$0 < K < 2.39$$

Comparando las expresiones se observa que, como era previsible, al aumentar el período de muestreo se reduce el rango de estabilidad del sistema.

Codificación en MATLAB

Para realizar los ejercicios utilizaremos el código que se muestra en la tabla 10.1, el cual será explicado parte por parte ya que de esta forma será más sencillo realizar los ejercicios propuestos en donde solamente cambiaremos el vector de entrada de la forma [an an-1 an-2... a0] tomando en cuenta que si no existe un valor en nuestra función de transferencia debe ser sustituido, por ejemplo para la función:

$$s^5 + 3s^4 + 9s^3 + 5s^1 - 3 = 0$$

El vector de entrada será:

$$[1 \ 3 \ 9 \ 0 \ 5 \ -3]$$

Nota. El criterio de estabilidad de Routh-Hurwitz identifica las condiciones cuando los polos de un polinomio se cruzan en el semiplano de la derecha y, por lo tanto, se considerarían inestables en la ingeniería de control.

A continuación se muestra el funcionamiento de cada parte del código del criterio de estabilidad de Routh-Hurwitz en MATLAB:

Código	Función
<pre>%% Initialization clear ; close all; clc</pre>	Eliminar elementos del espacio de trabajo, liberando memoria del sistema
<pre>% Taking coefficients vector and organizing the first two rows coeffVector = input('input vector of your system coefficients: \n i.e. [an an-1 an-2 ... a0] = '); coeffLength = length(coeffVector); = round(coeffLength/2);</pre>	Asignación de nombres a los coeficientes y organización de las filas a través la especificación del formato de las columnas de la tabla
<pre>% Initialize Routh-Hurwitz table with empty zero array rhTable = zeros(coeffLength,rhTableColumn);</pre>	Se ordena iniciar la tabla con una matriz de cero vacía
<pre>% Compute first row of the table rhTable(1,:) = coeffVector(1,1:2:coeffLength);</pre>	Se realiza el cálculo de la primera fila comenzando con uno hasta 2 vectores
<pre>% Check if length of coefficients vector is even or odd if (rem(coeffLength,2) ~= 0)</pre>	Comprueba con la función "IF" de comparación si la longitud de coeficientes es par o impar
<pre>% if odd, second row of table will be rhTable(2,1:rhTableColumn - 1) = coeffVector(1,2:2:coeffLength); else</pre>	Si detecta que es impar la segunda fila de la tabla codificará la siguiente indicación pero también incluye la función "else" que quiere decir que de otro modo o en casa de que no cumpla con la comparación ejecute la siguiente indicación.
<pre>% if even, second row of table will be rhTable(2,:) = coeffVector(1,2:2:coeffLength); end</pre>	Si detecta que es para la segunda fila de la tabla codificará la siguiente indicación
<pre>%% Calculate Routh-Hurwitz table's rows % Set epss as a small value epss = 0.01;</pre>	Establece un mínimo error de redondeo a través de eps, debido a esta precisión finita se pueden producir errores de redondeo al almacenar números y efectuar operaciones.
<pre>% Calculate other elements of the table for i = 3:coeffLength</pre>	Calculando otros elementos de la tabla con la función "for" que quiere decir para

	ejecutar un grupo de instrucciones en un bucle un número especificado de veces
<pre> % special case: row of all zeros if rhTable(i-1,:) == 0 order = (coeffLength - i); cnt1 = 0; cnt2 = 1; for j = 1:rhTableColumn - 1 rhTable(i-1,j) = (order - cnt1) * rhTable(i-2,cnt2); cnt2 = cnt2 + 1; cnt1 = cnt1 + 2; end end for j = 1:rhTableColumn - 1 </pre>	Si existe una fila en la que todos los números
<pre> % first element of upper row firstElemUpperRow = rhTable(i-1,1); </pre>	Primer elemento de la fila superior
<pre> % compute each element of the table rhTable(i,j) = ((rhTable(i-1,1) * rhTable(i-2,j+1)) - (rhTable(i-2,1) * rhTable(i-1,j+1))) / firstElemUpperRow; end </pre>	Ya que incluye los valores especiales, calcula todos los elementos de la tabla sin excepciones.
<pre> % special case: zero in the first column if rhTable(i,1) == 0 rhTable(i,1) = eps; end end </pre>	Para el caso especial de un cero en la primera columna ejecuta que existirá un valor muy pequeño
<pre> %% Compute number of right hand side poles(unstable poles) % Initialize unstable poles with zero unstablePoles = 0; </pre>	Inicia calculando los polos inestables del lado derecho con cero
<pre> % Check change in signs for i = 1:coeffLength - 1 if sign(rhTable(i,1)) * sign(rhTable(i+1,1)) == -1 unstablePoles = unstablePoles + 1; end end </pre>	Comprueba el cambio en el código
<pre> % Print calculated data on screen fprintf('\n Routh-Hurwitz Table:\n') rhTable </pre>	Muestra los datos calculados en la pantalla
<pre> % Print the stability result on screen if unstablePoles == 0 fprintf('~~~~~> it is a stable system! <~~~~~\n') else fprintf('~~~~~> it is an unstable system! <~~~~~\n') end </pre>	Imprime el resultado de estabilidad en la pantalla, se agregan comentarios entre paréntesis y comillas que se desean mostrar en el resultado
<pre> fprintf('\n Number of right hand side poles =%2.0f\n',unstablePoles) reply = input('Do you want roots of system be shown? Y/N ', 's'); if reply == 'y' reply == 'Y' sysRoots = roots(coeffVector); fprintf('\n Given polynomial coefficients roots :\n') sysRoots end </pre>	Crea una pregunta hacia el usuario con dos posibilidades en donde al ejecutar la letra o número que activa la función "if" muestra el resultado de la comparación, para este caso los coeficientes polinomiales dados.

Tabla 10.1 código de estabilidad de acuerdo al método de Routh-Hurwitz

Ejemplo 1

$$8s^5 + 6s^4 + 5s^3 + 3s^2 + 9s^1 + 1 = 0$$

a_0 a_1 a_2 a_3 a_4 a_5

s^5	8	5	9
s^4	6	3	1
s^3	1	7.66	
s^2	-42.96	1	
s^1	7.68		
s^0	1		

$$b_1 = \frac{a_1 a_2 - a_0 a_5}{a_1} = \frac{(6)(5) - (8)(3)}{(6)} = 1$$

$$b_2 = \frac{a_1 a_4 - a_0 a_5}{a_1} = \frac{(6)(9) - (8)(1)}{(6)} = 7.66$$

$$c_1 = \frac{b_1 a_3 - a_1 b_2}{b_1} = \frac{(1)(3) - (6)(7.66)}{(1)} = -42.96$$

$$c_2 = \frac{b_1 a_5 - a_1 b_3}{b_1} = \frac{(1)(1) - (6)(0)}{(1)} = 1$$

$$d_1 = \frac{c_1 b_2 - b_1 c_2}{c_1} = \frac{(-42.96)(7.66) - (1)(1)}{(-42.96)} = 7.68$$

Se trata de un sistema inestable, debido a que hay cambios de signo en la primera columna, entre +1 y -42.96 y -42.96 y 7.68.

```

1 clear ; close all; clc
2 % Taking coefficients vector and organizing the first two rows
3 coeffVector = ([8 6 5 3 9 1]);
4 coeffLength = length(coeffVector);
5 rhTableColumn = round(coeffLength/2);
6 % Initialize Routh-Hurwitz table with empty zero array
7 rhTable = zeros(coeffLength,rhTableColumn);
8 % Compute first row of the table
9 rhTable(1,:) = coeffVector(1,1:2:coeffLength);
10 % Check if length of coefficients vector is even or odd
11 if (rem(coeffLength,2) ~= 0)
12     % if odd, second row of table will be
13     rhTable(2,1:rhTableColumn - 1) = coeffVector(1,2:2:coeffLength);
14 else
15     % if even, second row of table will be
16     rhTable(2,:) = coeffVector(1,2:2:coeffLength);
17 end

```

10.2 Código Matlab para vector [8 6 5 3 9 1]

```

Routh-Hurwitz Table:
rhTable = 6x3
  1   2   3
1  8.0000  5.0000  9.0000
2  6.0000  3.0000  1.0000
3  1.0000  7.6667  0
4 -43.0000  1.0000  0
5  7.6899  0  0
6  1.0000  0  0

~~~~~> it is an unstable system! <~~~~~

Number of right hand side poles = 2

```

10.3 Tabla de filas pares e impares del polinomio y resultado de estado del sistema

Ejemplo 2

$$1s^5 + 7s^4 + 3s^3 + 10s^2 + 6s^1 + 8 = 0$$

$a_0 \quad a_1 \quad a_2 \quad a_3 \quad a_4 \quad a_5$

s^5	1	3	6
s^4	7	10	8
s^3	3.66	3.33	
s^2	3.66	8	
s^1	-4.73		
s^0	8		

$$b_1 = \frac{a_1 a_2 - a_0 a_3}{a_1} = \frac{(3)(7) - (1)(10)}{(3)} = 3.66$$

$$b_2 = \frac{a_1 a_4 - a_0 a_5}{a_1} = \frac{(3)(6) - (1)(8)}{(3)} = 3.33$$

$$c_1 = \frac{b_1 a_3 - a_1 b_2}{b_1} = \frac{(3.66)(10) - (7)(3.33)}{(3.66)} = 3.63$$

$$c_2 = \frac{b_1 a_5 - a_1 b_3}{b_1} = \frac{(3.66)(8) - (7)(0)}{(3.66)} = 8$$

$$d_1 = \frac{c_1 b_2 - b_1 c_2}{c_1} = \frac{(3.63)(3.33) - (3.66)(8)}{(3.63)} = -4.73$$

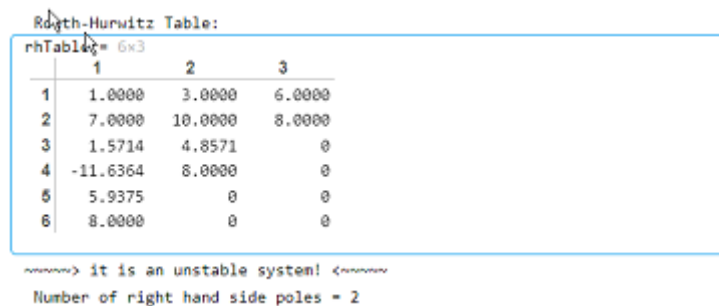
Se trata de un sistema inestable, debido a que hay cambios de signo en la primera columna, entre +3.63 y -4.73 y -4.73 y 8.

```

1 clear ; close all; clc
2 % Taking coefficients vector and organizing the first two rows
3 coeffVector = ([1 7 3 10 6 8]);
4 coeffLength = length(coeffVector);
5 rhTableColumn = round(coeffLength/2);
6 % Initialize Routh-Hurwitz table with empty zero array
7 rhTable = zeros(coeffLength,rhTableColumn);
8 % Compute first row of the table
9 rhTable(1,:) = coeffVector(1,1:2:coeffLength);
10 % Check if length of coefficients vector is even or odd
11 if (rem(coeffLength,2) ~= 0)
12     % if odd, second row of table will be
13     rhTable(2,1:rhTableColumn - 1) = coeffVector(1,2:2:coeffLength);
14 else
15     % if even, second row of table will be
16     rhTable(2,:) = coeffVector(1,2:2:coeffLength);
17 end

```

10.4 Código Matlab para vector [8 6 5 3 9 1]



10.5 Tabla de filas pares e impares del polinomio y resultado de estado del sistema

Ejemplo 3

$$1s^6 + 3s^5 + 3s^4 + 7s^3 + 9s^2 + 8s + 2 = 0$$

$a_0 \quad a_1 \quad a_2 \quad a_3 \quad a_4 \quad a_5 \quad a_6$

s^6	1	3	9	2
s^5	3	7	10	0
s^4	0.66	6.33	2	
s^3	-21.77	8.8	0	
s^2	6.59	2		
s^1	5.82			
s^0	2			

$$b_1 = \frac{a_1 a_2 - a_0 a_3}{a_1} = \frac{(3)(3) - (1)(7)}{(3)} = 0.66$$

$$b_2 = \frac{a_1 a_4 - a_0 a_5}{a_1} = \frac{(3)(9) - (1)(8)}{(3)} = 6.33$$

$$b_3 = \frac{a_1 a_6 - a_0 a_7}{a_1} = \frac{(3)(2) - (1)(0)}{(3)} = 2$$

$$c_1 = \frac{b_1 a_3 - a_1 b_2}{b_1} = \frac{(0.66)(7) - (3)(6.33)}{(0.66)} = -21.77$$

$$c_2 = \frac{b_1 a_5 - a_1 b_3}{b_1} = \frac{(0.66)(8) - (3)(0)}{(0.66)} = 8.8$$

$$c_3 = \frac{b_1 a_7 - a_1 b_4}{b_1} = \frac{(0.66)(0) - (3)(0)}{(0.66)} = 0$$

$$d_1 = \frac{c_1 b_2 - b_1 c_2}{c_1} = \frac{(-21.77)(6.33) - (0.66)(8.8)}{(-21.77)} = 6.59$$

$$d_2 = \frac{c_1 b_3 - b_1 c_3}{c_1} = \frac{(-21.77)(2) - (0.66)(0)}{(-21.77)} = 2$$

$$e_1 = 7.48$$

```

1 clear ; close all; clc
2 % Taking coefficients vector and organizing the first two rows
3 coeffVector = ([1 3 3 7 9 8 2]);
4 ceoffLength = length(coeffVector);
5 rhTableColumn = round(ceoffLength/2);
6 % Initialize Routh-Hurwitz table with empty zero array
7 rhTable = zeros(ceoffLength,rhTableColumn);
8 % Compute first row of the table
9 rhTable(1,:) = coeffVector(1,1:2:ceoffLength);
10 % Check if length of coefficients vector is even or odd
11 if (rem(ceoffLength,2) ~= 0)
12     % if odd, second row of table will be
13     rhTable(2,1:rhTableColumn - 1) = coeffVector(1,2:2:ceoffLength);
14 else
15     % if even, second row of table will be
16     rhTable(2,:) = coeffVector(1,2:2:ceoffLength);
17 end

```

10.2 Código Matlab para vector [8 6 5 3 9 1]

Routh-Hurwitz Table:

	1	2	3	4
1	1.0000	3.0000	9.0000	2.0000
2	3.0000	7.0000	8.0000	0
3	0.6667	6.3333	2.0000	0
4	-21.5000	-1.0000	0	0
5	6.3023	2.0000	0	0
6	5.8229	0	0	0
7	2.0000	0	0	0

~~~~~> it is an unstable system! <~~~~~

Number of right hand side poles = 2

10.3 Tabla de filas pares e impares del polinomio y resultado de estado del sistema

# 11. LUGAR GEOMÉTRICO DE LAS RAÍCES

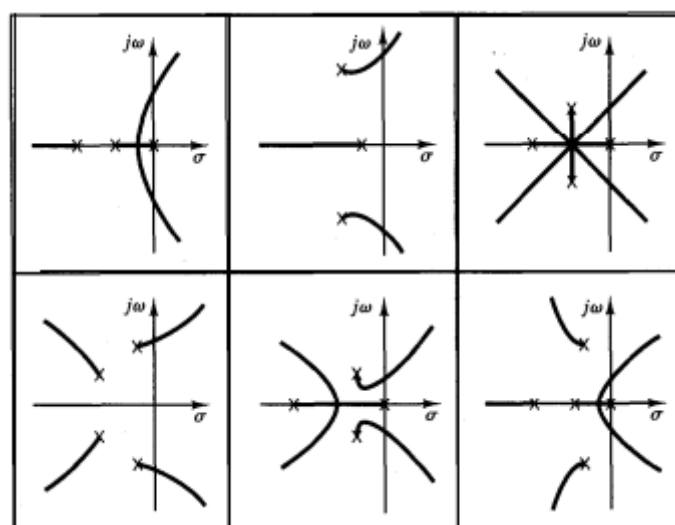
El lugar geométrico de las raíces consiste en una representación gráfica de los polos de la función de transferencia a lazo cerrado a medida que varía uno o varios parámetros del sistema.

Este método nos sirve para identificar la estabilidad de un sistema de forma gráfica, además nos muestra qué sucede con las variaciones de  $K$  y para qué puntos el sistema es estable o inestable.

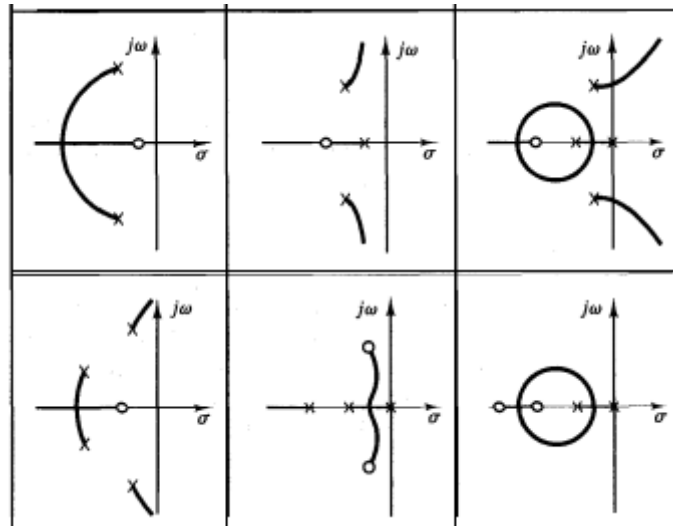
Para un sistema complejo en lazo abierto con muchos polos y ceros, puede parecer complicado construir una gráfica del lugar geométrico de las raíces, aunque en realidad no es difícil si se aplican las reglas para construir dicho lugar geométrico. Ubicando los puntos y las asíntotas específicos y calculando los ángulos de salida de los polos complejos y los ángulos de llegada a los ceros complejos, podemos construir la forma general de los lugares geométricos de las raíces sin dificultad.

## Configuraciones comunes de polos y ceros y los correspondientes lugares geométricos de las raíces.

El patrón de los lugares geométricos de las raíces sólo depende de la separación relativa de los polos y ceros en lazo abierto. Si el número de polos en lazo abierto excede el número de ceros finitos en tres o más, existe un valor de la ganancia  $K$  más allá del cual los lugares geométricos de las raíces entran en el semiplano derecho del plano  $s$  y, por tanto, el sistema puede volverse inestable. Un sistema estable debe tener todos sus polos en lazo cerrado en el semiplano izquierdo del plano  $s$ .



11.1a Configuraciones comunes de polos y ceros y los correspondientes lugares geométricos de las raíces.



11.1b Configuraciones comunes de polos y ceros y los correspondientes lugares geométricos de las raíces.

### Ejemplo 1

Tomando en cuenta la siguiente función de transferencia

$$G(s) = \frac{k(s^2 + 3s + 5)}{s(s + 5)(s + 7)(s^2 + 2.4s + 2)}$$

Asignamos una variable a cada término del denominador de manera que al aplicar el comando *conv* podamos obtener el valor completo de todo el denominador

$$a = s(s + 5) = s^2 + 5 \rightarrow [1 \ 5 \ 0]$$

$$b = s + 7 \rightarrow [1 \ 7]$$

$$c = s^2 + 2.4s + 2 \rightarrow [1 \ 2.4 \ 2]$$

$$d = \text{conv}(a, b) \quad e = \text{conv}(c, d)$$

Después calculamos los ceros de lazo abierto y cerrado a partir del comando *roots*. Finalmente utilizamos el comando *rlocus* que permite ver el lugar de las raíces de un sistema en el que el parámetro variable es la ganancia (K).

Asignamos los límites de los ejes de la gráfica apoyándonos del comando *axis* que nos permite ajustar la escala de los ejes de modo que varíe entre el mínimo y el máximo.

A continuación se muestra el código de ejecución para obtener la gráfica del lugar geométrico de las raíces así como sus ceros y polos:

```

1      %Producto de dos polinomios
2 -    a= [1 5 0];
3 -    b= [1 7];
4 -    c= [1 2.4 2];
5 -    d= conv(a,b)
6 -    e= conv(c,d)
7      %polinomio del denominador
8 -    den=[1 14.4 65.8 108 70 0]
9      %Ceros de lazo abierto
10 -   p=[1 3 5];
11 -   r=roots(p)
12      %Polos complejos conjugados en lazo abierto
13 -   q=roots(c)
14
15 -   num=[0 0 0 1 3 5]
16 -   rlocus(num,den)
17 -   t=[-10 10 -10 10]; axis(t)
18 -   grid
19 -   title('Gráfica de lugar geométrico de las raíces')

```

11.2 Código para gráfica de lugar geométrico de las raíces ejemplo 1

Ceros y polos:

```

r =
    -1.5000 + 1.6583i
    -1.5000 - 1.6583i

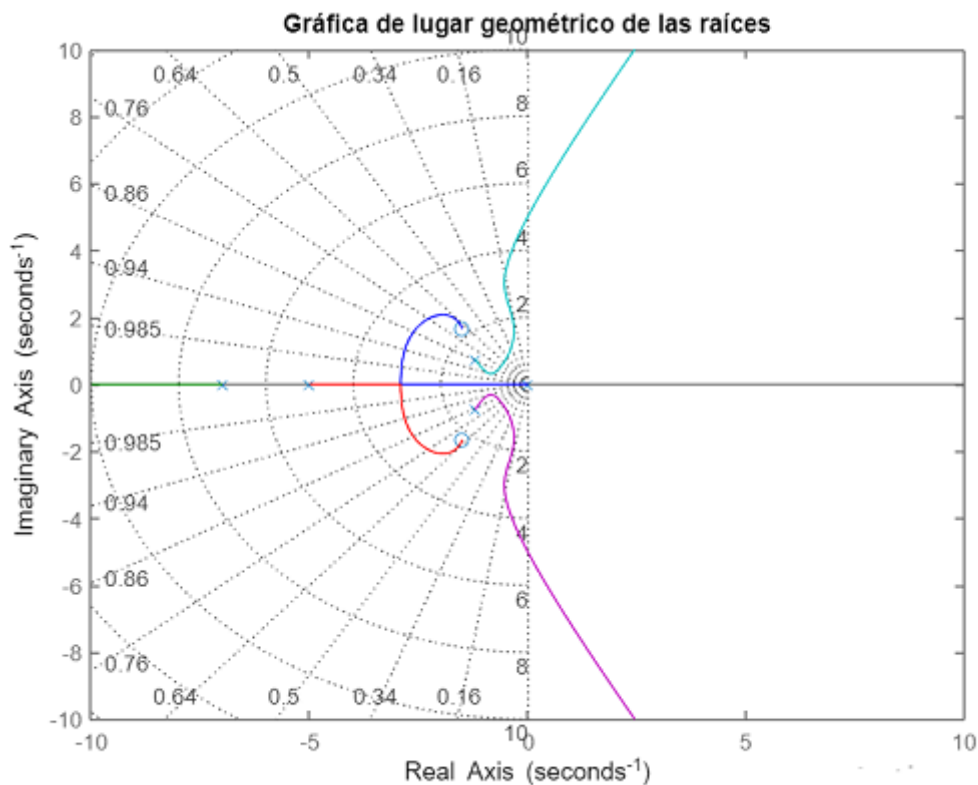
q =
    -1.2000 + 0.7483i
    -1.2000 - 0.7483i

```

11.3 Ceros y polos ejemplo 1



En nuestro software se muestra la siguiente gráfica en respuesta a las instrucciones digitalizadas:



#### 11.4 Gráfica de Lugar Geométrico de las raíces en respuesta a función de transferencia ejemplo 1

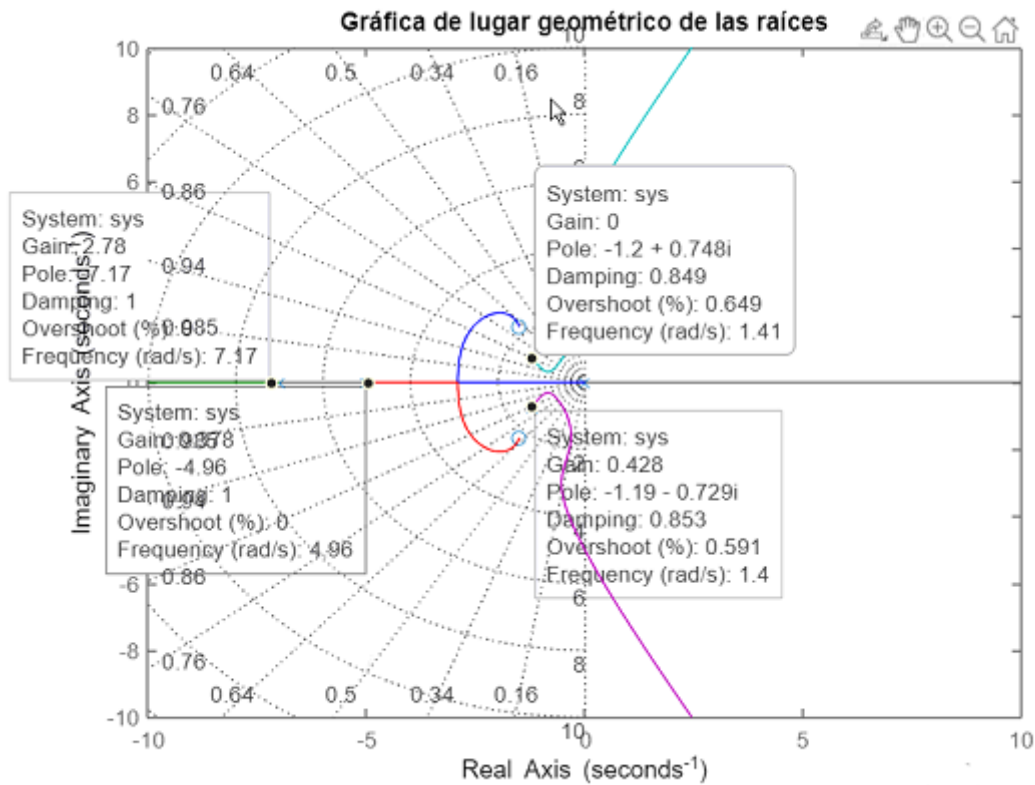
Colocamos el cursor en cada una de las raíces que nos muestra el sistema encontrando que este sistema tiene una ganancia  $K$  de 2.35 ya que en esta parte de la gráfica se van a situar los polos de bucle cerrado del sistema realimentando las posiciones indicadas. Todas las raíces tienden a cero o infinito. La estabilidad del sistema será alcanzada cuando  $k > 2.35$  entonces pasará de tener una respuesta de forma transitoria a una respuesta estacionaria.

Overshoot. Indica el valor de sobreoscilación que tendría un sistema de 2º Orden Orden "Puro" (sin ceros) subamortiguado cuyos polos fueran situados en la posición dada y su complejo conjugado.

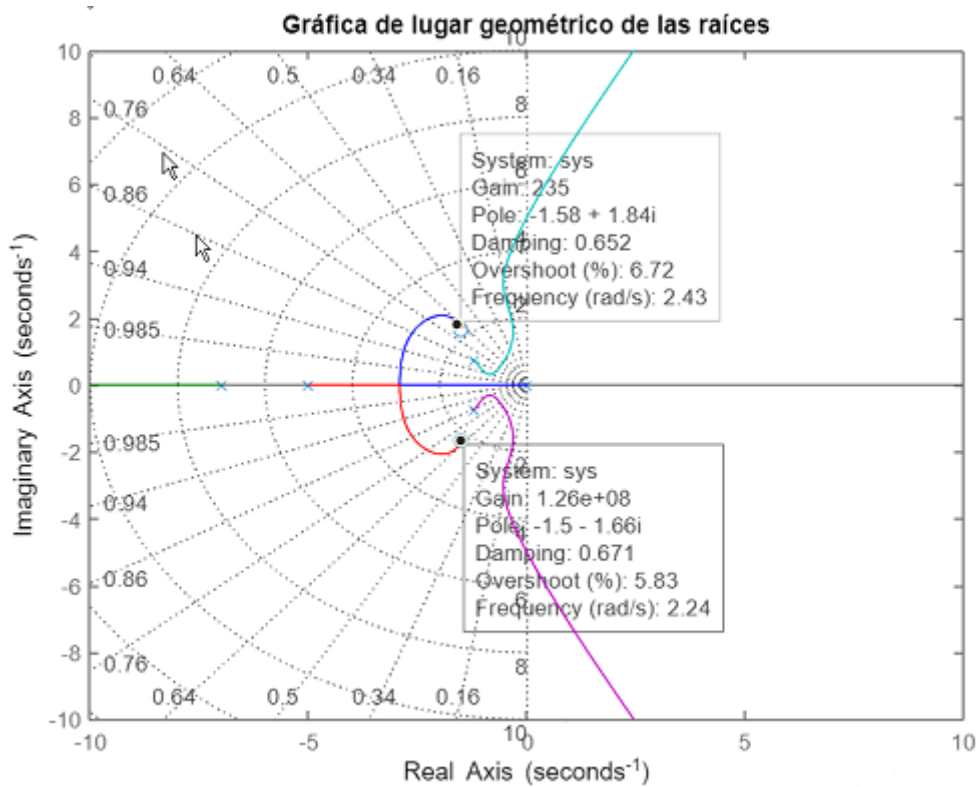
Damping. Indica el valor de amortiguamiento del sistema.

En el caso de que el sistema realimentado contenga un tercer polo para los valores de ganancia, la sobreoscilación mostrada no será la que tenga el sistema realimentado para el valor de ganancia indicado.

La cercanía de del valor real de sobreoscilación al valor de sobreoscilación mostrado, depende del grado de dominio del polo situado en la posición y de su complejo conjugado respecto al tercer polo.



11.5 Valores de las raíces, ganancia, polos, damping, overshoot y frecuencia de función de transferencia de ejemplo 1



11.6 Valores de las raíces, ganancia, polos, damping, overshoot y frecuencia de función de transferencia de ejemplo 1

## Ejemplo 2

Tomando en cuenta la siguiente función de transferencia

$$G(s) = \frac{k(s + 1)}{s^2(s + 4)}$$

Asignamos una variable al término del denominador

$$a = s^2(s + 4) = s^3 + 4s^2 \rightarrow [1 \ 4 \ 0 \ 0]$$

A continuación se muestra el código de ejecución para obtener la gráfica del lugar geométrico de las raíces así como sus ceros y polos:

```
1 %Polinomio del denominador
2 - a=[1 4 0 0];
3 %Ceros de lazo abierto
4 - p=[1 1];
5 - r=roots(p)
6 %Polos complejos conjugados en lazo abierto
7 - q=roots(a)
8
9 - num=[0 0 1 1]
10 - rlocus(num,den)
11 - t=[-4 2 -4 4]; axis(t)
12 - grid
13 - title('Gráfica de lugar geométrico de las raíces')
```

11.7 Código para gráfica de lugar geométrico de las raíces ejemplo 2

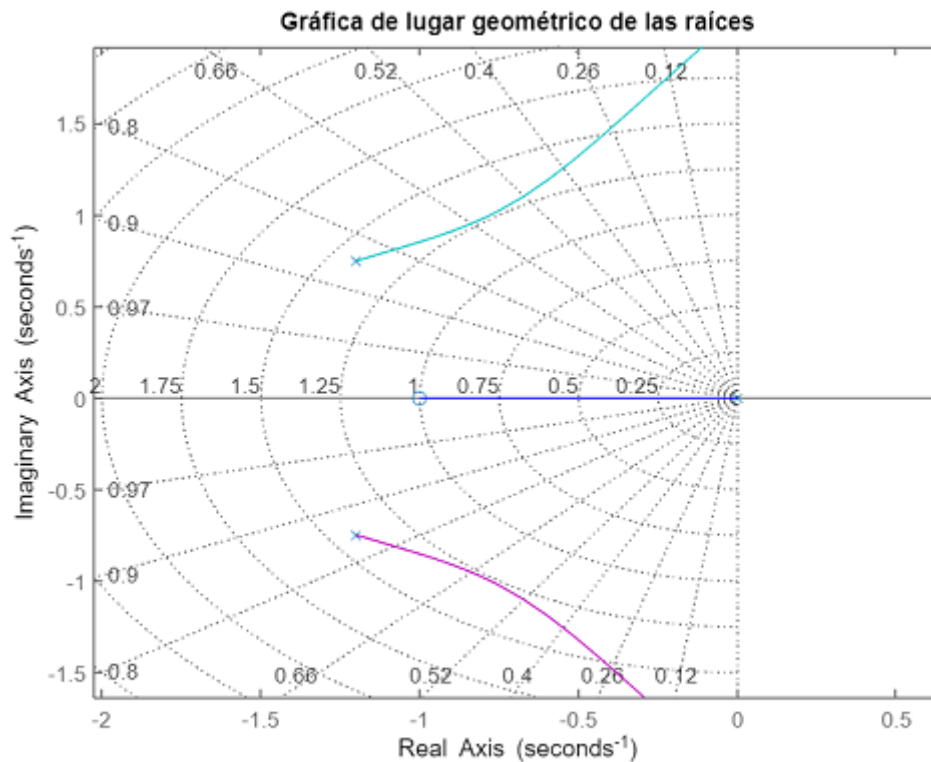
Ceros y polos:

```
r =
    -1

q =
     0
     0
    -4
```

11.8 Ceros y polos ejemplo 2

En nuestro software se muestra la siguiente gráfica en respuesta a las instrucciones digitalizadas:



### 11.9 Gráfica de Lugar Geométrico de las raíces en respuesta a función de transferencia ejemplo 2

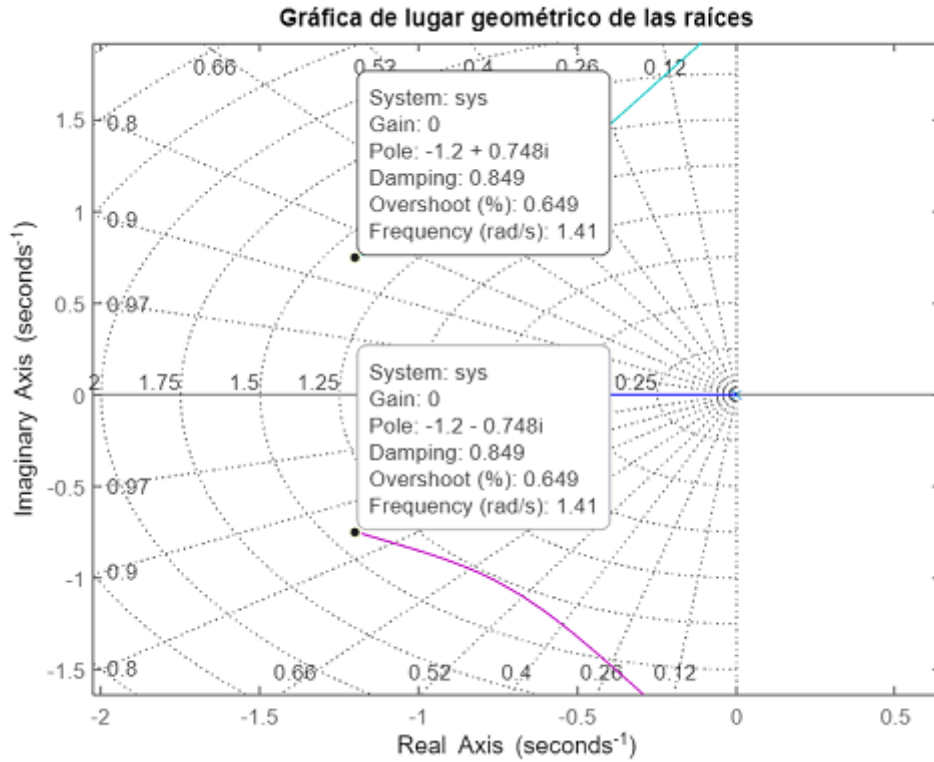
Colocamos el cursor en cada una de las raíces que nos muestra el sistema encontrando que este sistema tiene una ganancia  $K$  de 0 lo que quiere decir que nuestro sistema es completamente estable. Todas las raíces tienden a cero o infinito.

Overshoot de 0.649% Indica el valor de sobreoscilación que tendría un sistema de 2° Orden “Puro” (sin ceros) subamortiguado cuyos polos fueran situados en la posición dada y su complejo conjugado.

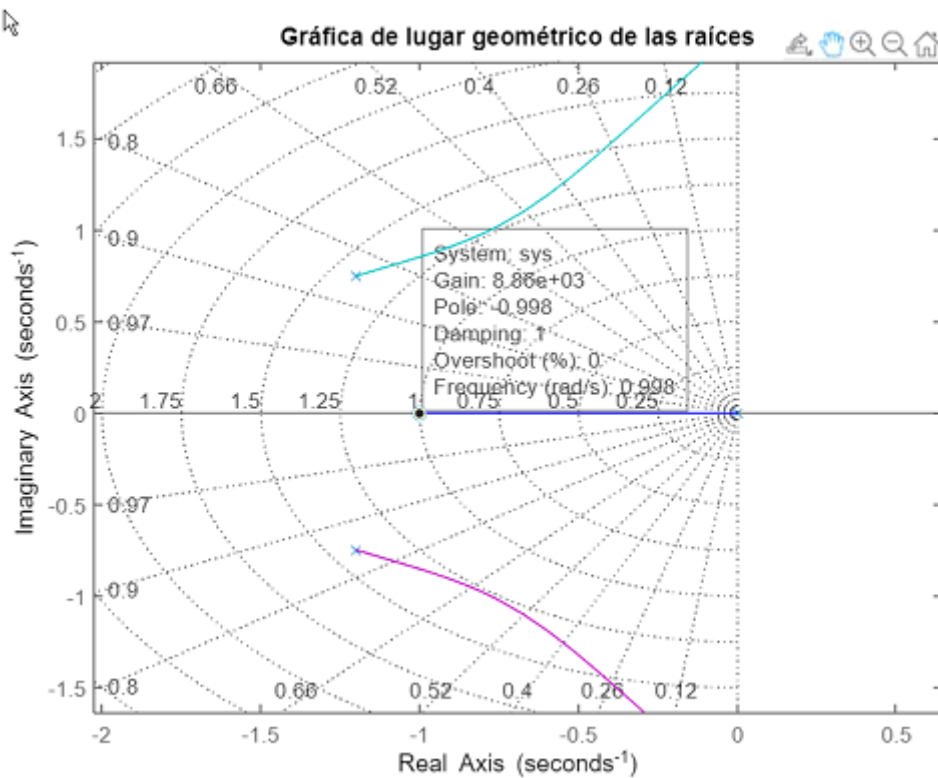
Damping. Indica el valor de amortiguamiento del sistema.

En el caso de que el sistema realimentado contenga un tercer polo para los valores de ganancia, la sobreoscilación mostrada no será la que tenga el sistema realimentado para el valor de ganancia indicado.

La cercanía de del valor real de sobreoscilación al valor de sobreoscilación mostrado, depende del grado de dominio del polo situado en la posición y de su complejo conjugado respecto al tercer polo.



11.10 Valores de las raíces, ganancia, polos, damping, overshoot y frecuencia de función de transferencia de ejemplo 2



11.11 Valores de las raíces, ganancia, polos, damping, overshoot y frecuencia de función de transferencia de ejemplo 2

### Ejemplo 3

Tomando en cuenta la siguiente función de transferencia

$$G(s) = \frac{k(s + 2)}{s(s - 2)(s^2 + 4s + 16)}$$

Asignamos una variable a cada término del denominador de manera que al aplicar el comando *conv* podamos obtener el valor completo de todo el denominador

$$a = s(s - 2) = s^2 - 2s \rightarrow [1 \ -2 \ 0]$$

$$b = s^2 + 4s + 16 \rightarrow [1 \ 4 \ 16]$$

$$c = \text{conv}(a, b)$$

A continuación se muestra el código de ejecución para obtener la gráfica del lugar geométrico de las raíces así como sus ceros y polos:

```
a=[1 -2 0];
b=[1 4 16];
c= conv(a,b);
den=[1 2 8 -32 0]
%Ceros de lazo abierto
p=[1 2];
r=roots(p)
%Polos complejos conjugados en lazo abierto
q=roots(b)

num=[0 0 0 1 2]
rlocus(num,den)
t=[-6 6 -6 6]; axis(t)
grid
title('Gráfica de lugar geométrico de las raíces')
```

11.12 Código para gráfica de lugar geométrico de las raíces ejemplo 3

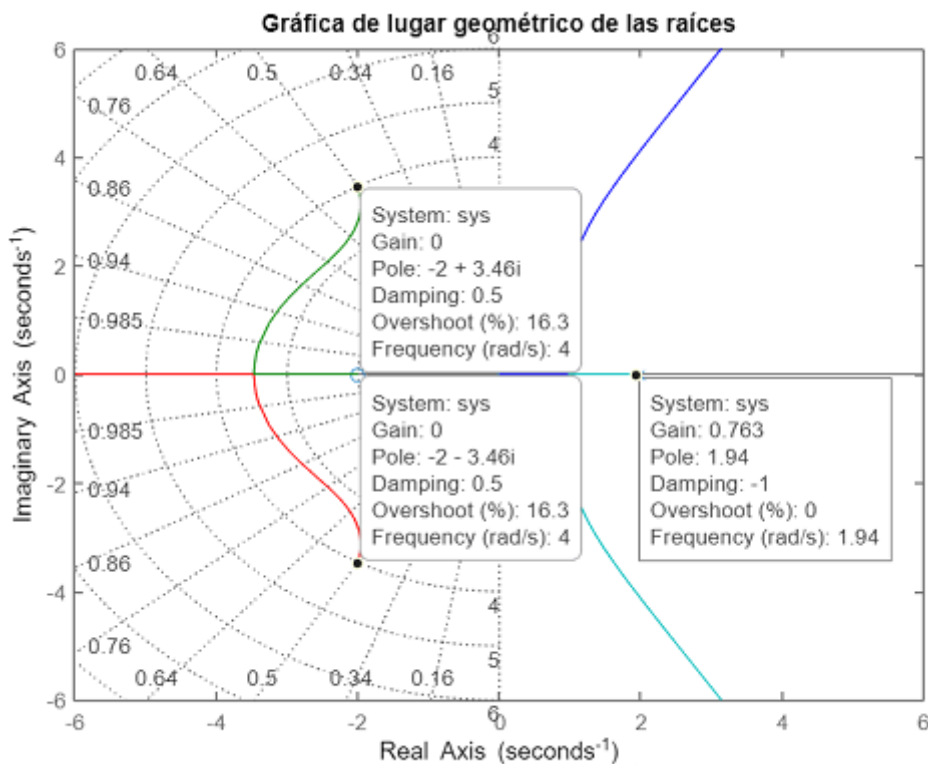
Ceros y polos:

```
r =
    -2

q =
-2.0000 + 3.4641i
-2.0000 - 3.4641i
```

11.13 Ceros y polos ejemplo 3

En nuestro software se muestra la siguiente gráfica en respuesta a las instrucciones digitalizadas:



#### 11.14 Gráfica de Lugar Geométrico de las raíces en respuesta a función de transferencia ejemplo 3

Para este caso el sistema debe tener una ganancia  $K$  mayor a 0.763 ya que en esta parte de la gráfica se observa que es inestable y para llevar al sistema la estabilidad tiene que sobrepasar ese rango, entonces pasará de tener una respuesta de forma transitoria a una respuesta estacionaria.. Todas las raíces tienden a cero o infinito.

Overshoot= 16.3% el cual indica el valor de sobreoscilación que tendría un sistema de 2° Orden “Puro” (sin ceros) subamortiguado cuyos polos fueran situados en la posición dada y su complejo conjugado.

En el caso de que el sistema realimentado contenga un tercer polo para los valores de ganancia, la sobreoscilación mostrada no será la que tenga el sistema realimentado para el valor de ganancia indicado.

Damping. Indica el valor de amortiguamiento.

La cercanía de del valor real de sobreoscilación al valor de sobreoscilación mostrado, depende del grado de dominio del polo situado en la posición y de su complejo conjugado respecto al tercer polo.

## Referencias

Andrés Ubeda. *Fundamentos de Automática*. Curso 2017/2018

Universidad de Oviedo. *Manual de uso MatLab*. Curso 2010-2011

Martín del Campo y Tinoco. *Manual de Laboratorio Teoría de Control y Robótica*. UNAM. 2020

Mecánica y control para Industriales. *Transformada de Laplace y Matlab*. Fecha de publicación: 05/03/2010. Fecha de consulta: 03/04/2020  
<https://blogs.ua.es/industriales/2010/03/05/transformada-de-laplace-y-matlab/>

Fracciones Parciales. Fecha de consulta 03/04/2020  
<http://ehernandez.mat.utfsm.cl/MAT021/pdfs/FraccionesParciales.pdf>

Sistemas de Primer Orden. Fecha de consulta 04/04/2020  
[http://www.cartagena99.com/recursos/alumnos/apuntes/Cap\\_5\\_SDx.pdf](http://www.cartagena99.com/recursos/alumnos/apuntes/Cap_5_SDx.pdf)

Ceduvirt. MatLab aplicado a ingeniería. Fecha de consulta 05/04/2020  
<https://www.ceduvirt.com/resources/CeduvirtMatlab.pdf>

Tinoco David. Apuntes Controladores. UNAM 2020. Fecha de consulta: 30/04/2020  
<https://e.edim.co/176399701/DtFniTy3j9Z1dlAM.pdf?response-content-disposition=filename%3D%22CONTROLADORES.pdf%22%3B%20filename%2A%3DU TF-8%27%27CONTROLADORES.pdf>

Tinoco David. Apuntes Respuesta en Frecuencia. UNAM 2020. Fecha de consulta: 30/04/2020  
[https://e.edim.co/176399701/GnWXnbNSwTznZYIm.pdf?response-content-disposition=filename%3D%22Respuesta\\_en\\_frecuencia.pdf%22%3B%20filename%2A%3DUTF-8%27%27Respuesta%2520en%2520frecuencia.pdf&](https://e.edim.co/176399701/GnWXnbNSwTznZYIm.pdf?response-content-disposition=filename%3D%22Respuesta_en_frecuencia.pdf%22%3B%20filename%2A%3DUTF-8%27%27Respuesta%2520en%2520frecuencia.pdf&)

PID. MathWorks. Fecha de consulta: 30/04/2020  
<https://la.mathworks.com/help/control/ref/pid.html>

Control PID de manera sencilla. MathWorks. Fecha de consulta: 30/04/2020  
[https://la.mathworks.com/videos/pid-control-made-easy-99680.html?elqsid=1588391292606&potential\\_use=Student](https://la.mathworks.com/videos/pid-control-made-easy-99680.html?elqsid=1588391292606&potential_use=Student)

Comando margin. MathWorks. Fecha de consulta: 28/04/2020  
<https://la.mathworks.com/help/control/ref/margin.html>



Ogata Katsuhiko. *Ingeniería de Control Moderna (3ª Ed)*. Person Education. México. 1998. PP. 319-357

Routh-Hurwitz stability criterion. Fecha de consulta: 16/05/2020

<https://la.mathworks.com/matlabcentral/fileexchange/17483-routh-hurwitz-stability-criterion>

Estabilidad de un Sistema de Control. Fecha de Consulta: 16/05/2020

<https://dademuch.com/2018/03/15/estabilidad-de-un-sistema-de-control/>