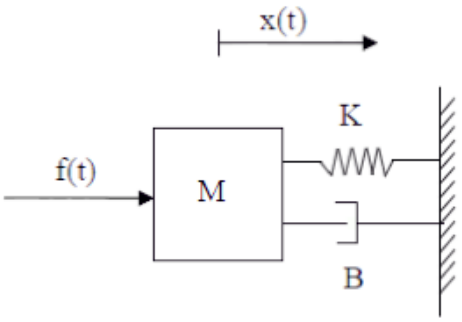
TEORÍA DE SISTEMAS

PRÁCTICA 4. FUNCIÓN DE TRANSFERENCIA

1. REPRESENTACIÓN DE UN SISTEMA MEDIANTE SU FUNCIÓN DE TRANSFERENCIA

Recordemos el sistema físico analizado en la práctica 2. Se trata de una masa **M** unida a un muelle de constante **K**, y con un rozamiento viscoso **B**, tal y como se describe en la figura:



El objetivo es ver cuál es el efecto de la fuerza aplicada ***f(t)*** al movimiento de la masa, descrito por ***x(t)***

La ecuación diferencial que rige el comportamiento de este sistema es:

$$f(t) = M \cdot \frac{d^2x(t)}{dt^2} + B \cdot \frac{dx(t)}{dt} + K \cdot x(t)$$

Para obtener la representación del sistema mediante su función de transferencia, daremos los siguientes pasos:

1. Planteamiento de la ecuación diferencial

En este caso, la ecuación diferencial es un dato del problema

2. Obtención de un punto de equilibrio

Dado que trabajaremos con variables incrementales (valor inicial = 0) será necesario determinar el punto de equilibrio sobre el que se va a trabajar.

En este caso se buscará el punto de equilibrio para una fuerza inicial:

$$f(0) = 10$$

En el punto de equilibrio las derivadas serán cero, por tanto:

$$f(0) = M \cdot 0 + B \cdot 0 + K \cdot x(0) \quad \Rightarrow \quad x(0) = \frac{f(0)}{K}$$

3. Linealización de las ecuaciones y expresión en variables incrementales

En este caso todos los términos de la ecuación son lineales; por tanto sólo hay que expresar la ecuación en variables incrementales:

$$\Delta f(t) = M \cdot \Delta\ddot{x}(t) + B \cdot \Delta\dot{x}(t) + K \cdot \Delta x(t)$$

4. Paso de las ecuaciones al dominio de Laplace

Una vez las ecuaciones expresadas en términos incrementales, el paso al dominio de Laplace es inmediato:

$$\mathcal{L}(x(t) = X(s)$$
$$\mathcal{L}(\dot{x}(t)) = s \cdot X(s)$$

En el caso que nos ocupa, la ecuación queda:

$$\mathrm{F(s)} = M \cdot s^2 \cdot X(s) + B \cdot \mathrm{s} \cdot X(s) + K \cdot \mathrm{X(s)}$$

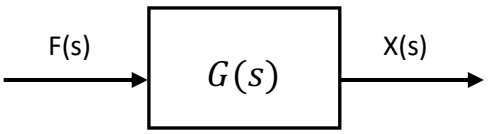
5. Función de transferencia

Una vez las ecuaciones en el dominio de Laplace, es posible obtener la función de transferencia **G(s)** o función que permite obtener la salida **X(s)** a partir de la entrada **F(s)**:

$$\mathrm{G(s)} = \frac{X(s)}{F(s)} = \frac{1}{M \cdot s^2 + B \cdot \mathrm{s} + K}$$

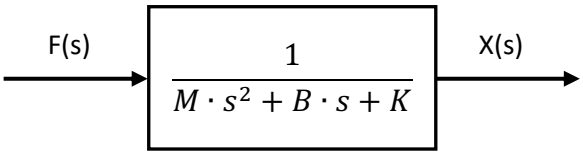
6. Diagrama de bloques

Cada función de transferencia se representa en un diagrama de bloques como el operador que multiplicado por la entrada nos ofrece la salida:



$$X(s) = G(s) \cdot F(s)$$

En nuestro caso:

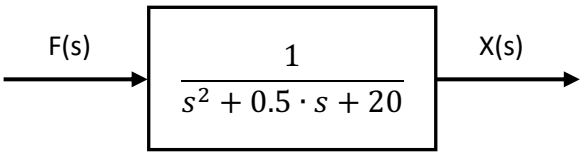


7. Simulación

Para representar una función de transferencia en Simulink se debe usar el bloque **'Transfer Fcn'** de la categoría **'Continuous'**. Este bloque permite introducir funciones de transferencia como cociente de dos polinomios. Cada polinomio se especifica por sus coeficientes en orden decreciente de potencias.

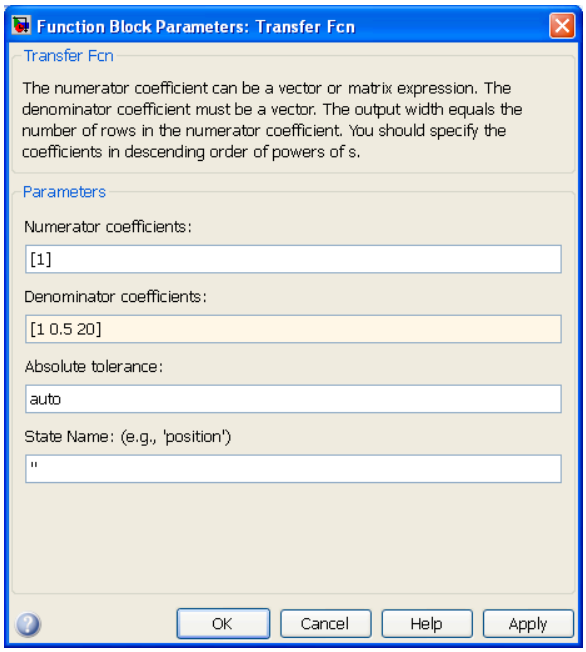
En nuestro caso, si:

- M = 1
- B = 0.5
- K =20

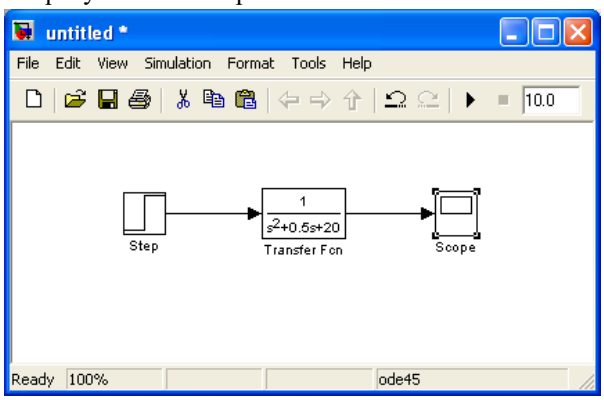


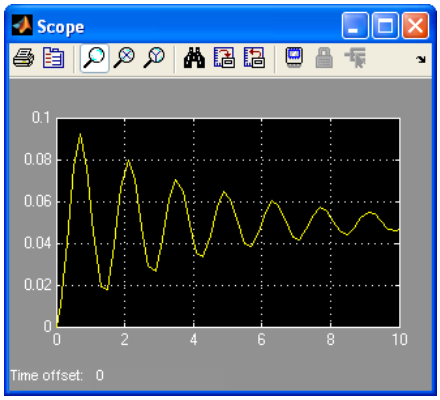
Tendremos como coeficientes del numerador: **[1]**
Y como coeficientes del denominador: **[1 0.5 20]**

Estos parámetros se introducirán como configuración del bloque **'Transfer Fcn'** tal y como muestra la figura inferior. El parámetro **'Absolute tolerance'** hace referencia al máximo error permitido en la simulación y no se modificará.

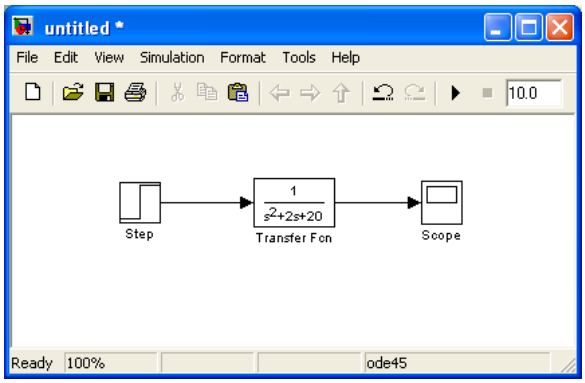


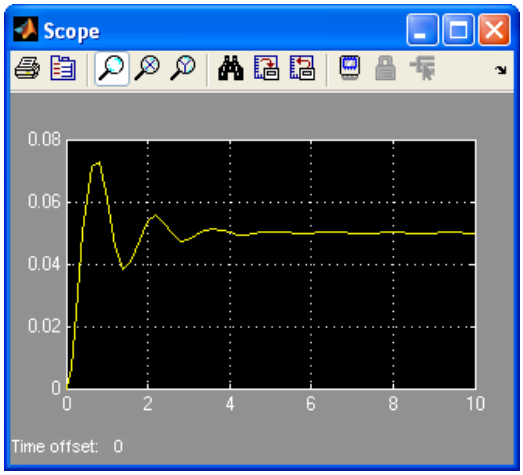
Para simular, conectaremos una señal de entrada cualquiera (escalón, senoidal, etc.) a la entrada del bloque y un osciloscopio a la salida:





A continuación podríamos probar el efecto que produce la variación de los distintos parámetros del sistema (**M, B** y **K**) sobre el comportamiento del mismo con sólo cambiar los valores de la función de transferencia.





Podemos ver, por ejemplo, como al aumentar el parámetro **B** (rozamiento) el movimiento del muelle es más amortiguado (se producen menos oscilaciones):

NOTA IMPORTANTE: Las variables con las que estamos trabajando son **incrementales:**

- Fuerza o variable de entrada: valor en el punto de equilibrio = 10N

  Aplicar un escalón de valor 1 equivale a ejercer una fuerza de 11N
  f(0) = 10N
  Δf(t) = 1N
  f(t) = f(0) + Δf(t) = 11N

- Posición o variable de salida: valor en el punto de equilibrio = 10/K = 0.5m

  Una posición que en el gráfico aparece como 0.08 equivale a una posición real de 0.58mx(0) = 0.5m
  Δx(t) = 0.08m
  x(t) = x(0) + Δx(t) = 0.58m

2. REPRESENTACIÓN EN MATLAB DE SISTEMAS CONTINUOS

Al igual que Simulink, Matlab también permite obtener la respuesta de sistemas continuos y discretos ante distintas señales de entrada.

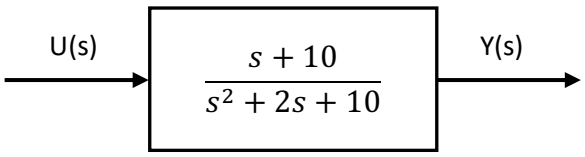
La forma de representar un sistema continuo es Matlab es mediante su función de transferencia en **s**, a través de la instrucción **tf.**

Forma de utilizar la instrucción *tf*:

```
nombre_sistema = tf (numerador, denominador)
```

- Variable que identificará al sistema numerador
- Vector con los coeficientes del polinomio del numerador
- Vector con los coeficientes del polinomio del numerador

Ejemplo:
Queremos representar el siguiente sistema continuo:



La instrucción de Matlab a utilizar será:

```
» sis1 = tf([1 10], [1 2 10])
Transfer function:
      s + 10
---------------
s^2 + 2 s + 10
```

A partir de este momento, la variable sis1 representará en Matlab el sistema correspondiente a esa función de transferencia.

3. RESPUESTA A LA SEÑAL ESCALÓN

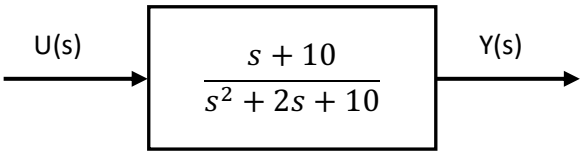
La instrucción **step** sirve para calcular la respuesta a escalón de cualquier sistema previamente definido.

Caben dos posibilidades:

- Obtener la representación gráfica de la respuesta
- Obtener los valores numéricos de la respuesta

Representación gráfica de la respuesta

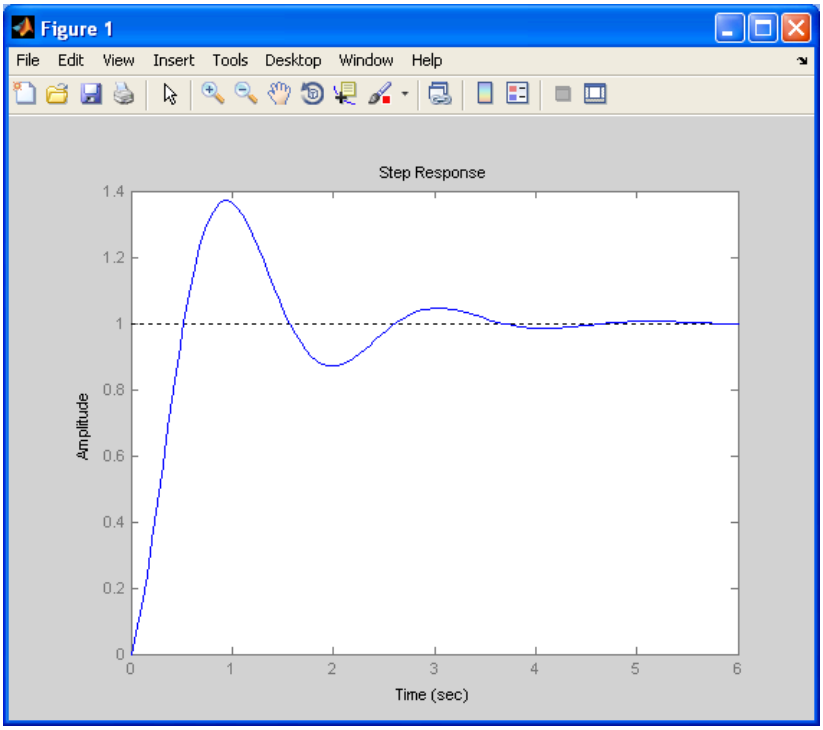
Obtendremos como ejemplo la respuesta a escalón del sistema continuo definido anteriormente:



Si suponemos que cuando creamos el sistema con la instrucción **tf** le dimos el nombre **sis1**, entonces la instrucción Matlab a teclear será la siguiente:

```
» step(sis1)
```

Y el resultado será el siguiente gráfico:



El gráfico representa el valor de la salida del sistema ante entrada escalón

Obtención de los valores numéricos de la respuesta:

En este caso lo que deseamos no es obtener el gráfico sino descargar en una variable el valor de la respuesta en cada instante de tiempo.

El formato para la instrucción es, en este caso:

```
» [y,t] = step(sis1);
```

Y el resultado será el siguiente:

- El vector **t** contendrá los instantes de tiempo para los que se ha calculado el valor de la salida
- El vector **y** contendrá los valores de la salida correspondientes a cada instante de tiempo

***NOTA**: recordemos que el punto y coma al final de la expresión sirve para que Matlab no muestre en pantalla el resultado de la operación; en otro caso habrían aparecido las ristras de valores de las variables.*

Como comprobación, podemos consultar un valor cualquiera de la variable t y de la variable y; por ejemplo el valor que hace el número 25 de todos los calculados:

```
» t(25)
ans =
      1.2590

» y(25)
ans =
      1.2285
```

Vemos que el valor 25 corresponde al instante de tiempo 1.259 segundos y que el valor de la señal de salida en ese instante de tiempo es 1.2285. Podemos comprobar estos valores aproximadamente sobre el gráfico de la respuesta que obtuvimos antes.

Disponer de los valores numéricos de los datos es útil para realizar cualquier tipo de operación matemática, como buscar el máximo, obtener el valor exacto en un instante de tiempo concreto, etc.

Forma de obtener la respuesta para un escalón no unitario

Lo visto anteriormente considera que al sistema se le aplica un escalón de valor uno. Para escalones no unitarios basta con multiplicar el sistema por el valor del escalón.

Por ejemplo, para obtener la respuesta a un escalón de 5 unidades bastará con teclear estas instrucciones:

```
» step(5*sis1)
```

o bien:

```
» [y,t] = step(5*sis1);
```

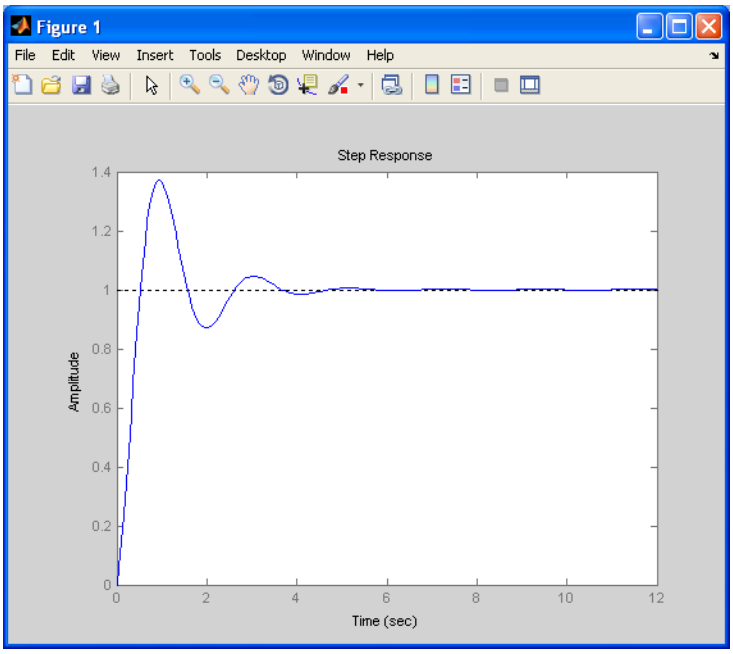
Forma de obtener la respuesta para instantes de tiempo posteriores

En el ejemplo realizado, Matlab calcula la respuesta del sistema hasta el instante t = 6 segundos (se puede comprobar sobre el gráfico). Si se desea obtener la respuesta para instantes posteriores basta con especificar un valor para el tiempo final en la instrucción step.

Por ejemplo, si qeremos obtener la respuesta ante escalón del sistema **sis1** no hasta el instante t=6 sino hasta el instante **t=12** deberíamos teclear el siguiente comando Matlab:

```
» step(sis1, 12)
```

Y el resultado sería el que mostramos en el gráfico que aparece a continuación:



4. RESPUESTA A UNA SEÑAL CUALQUIERA

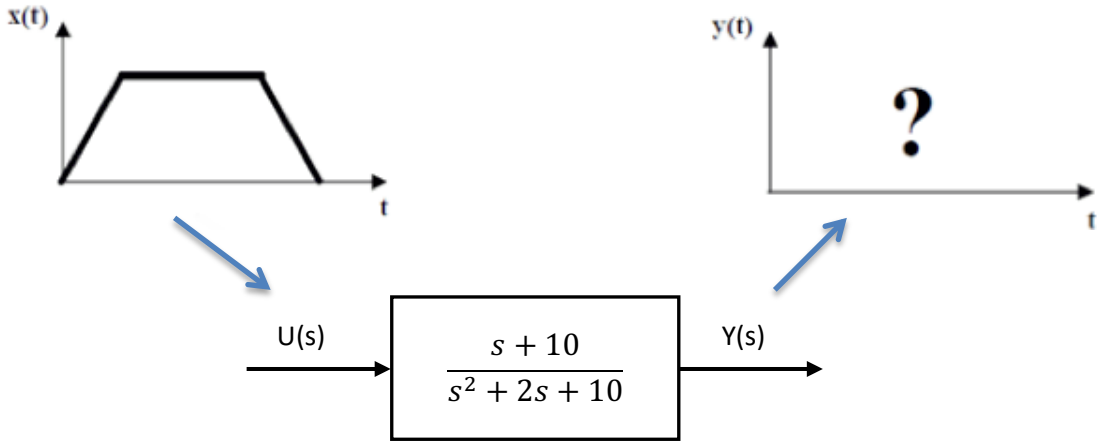
Al igual que la instrucción step nos ofrece la respuesta de un sistema a una señal escalón, también es posible obtener mediante Matlab la respuesta de un sistema ante una entrada cualquiera.

Para ello se utiliza la instrucción **lsim** que, al igual que la instrucción step, permite obtener los resultados de dos formas distintas:

- Como una representación gráfica
- Como valores numéricos

Hemos dicho que la instrucción **lsim** permite obtener la respuesta de un sistema ante una entrada cualquiera. El primer paso, antes de utilizar la instrucción, será definir la entrada a utilizar.

Por ejemplo, si deseamos conocer la respuesta del sistema sis1 definido anteriormente ante una entrada x(t) como la representada en la figura, deberemos en primer lugar definir esa señal x(t):



Una señal cualquiera se definirá mediante dos vectores:

- Un vector de instantes de tiempo
- Un vector de valores para la señal en cada uno de esos instantes de tiempo

Con el objeto de representar la señal con la mayor precisión posible, es recomendable utilizar un gran número de valores o, lo que es lo mismo, hacer que los instantes de tiempo entre cada dos valores sean lo más pequeños posible.

Formatos para la instrucción lsim:

- Si lo que se desea es la representación gráfica de la respuesta:

```
lsim (nombre_sistema, valor_entrada, t_entrada)
```

Identificador del sistema

Vector con los valores de la señal de entrada

Vector con los instantes de tiempo de la señal de entrada

- Si lo que se desea es obtener los valores numéricos de la respuesta:

```
[y,   t] = lsim (nombre_sistema, valor_entrada, t_entrada)
```

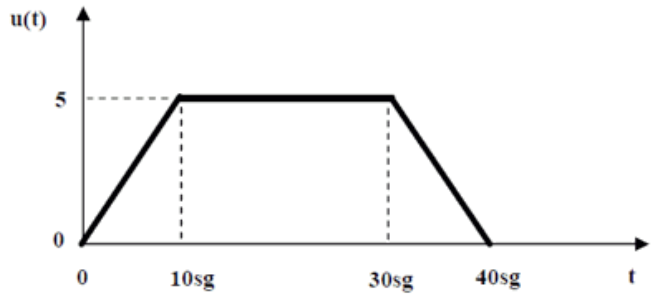
Vector de valores de la señal de salida

Vector de instantes de la señal de salida

Al igual que con la instrucción step, los vectores ***y*** y ***t*** definen la señal de salida; de este modo es posible realizar cualquier operación matemática sobre esos datos.

Ejemplo:

Obtendremos la respuesta del sistema **sis1** a la siguiente señal de entrada:



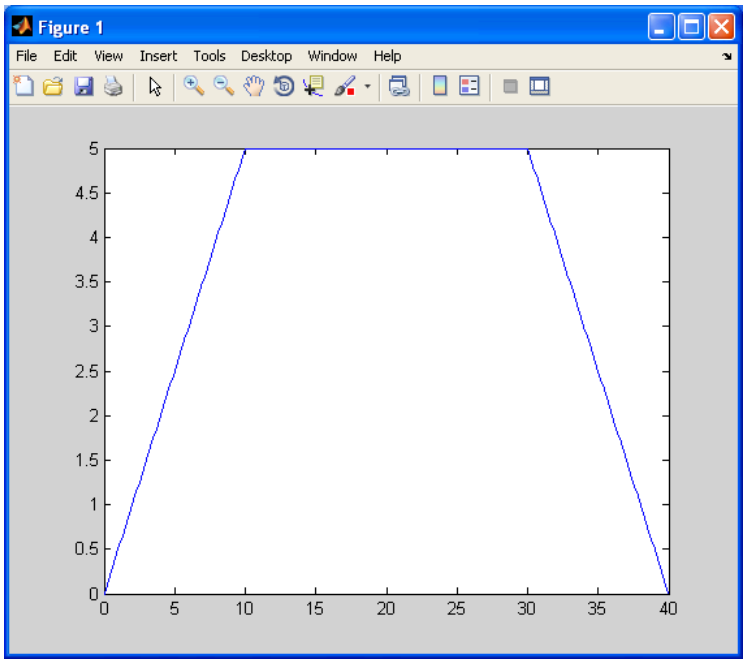
Como primer paso, debemos definir mediante dos vectores **t** (tiempo) y **u** (valores) la señal de entrada. De acuerdo con lo que se ha visto en prácticas anteriores, la definición de la señal se hará en tres tramos mediante las siguientes instrucciones de Matlab:

```
» t1 = [0:0.1:10]; % de 0 a 10 seg. a intervalos de 0.1 seg.
» u1 = 0.5*t1; % primer tramo de la señal
» t2 = [10.1:0.1:30]; % de 10.1 a 30 seg.
» u2 = 5*ones(size(t2)) % segundo tramo de la señal
» t3 = [30.1:0.1:40]; % de 30.1 a 40 seg.
» u3 = 5-0.5*(t3-30); % tercer tramo de la señal
» t = [t1, t2, t3]; % concatenación de los vectores de tiempo
» u = [u1, u2, u3]; % concatenación de los vectores de datos
```

Es conveniente comprobar que se han definido correctamente las variables **u** y **t**. Para ello el procedimiento más inmediato es utilizar la orden **plot** de Matlab, con la que podemos representar la variable **u** (señal) en función de la variable **t** (tiempo):

```
» plot(t,u)
```

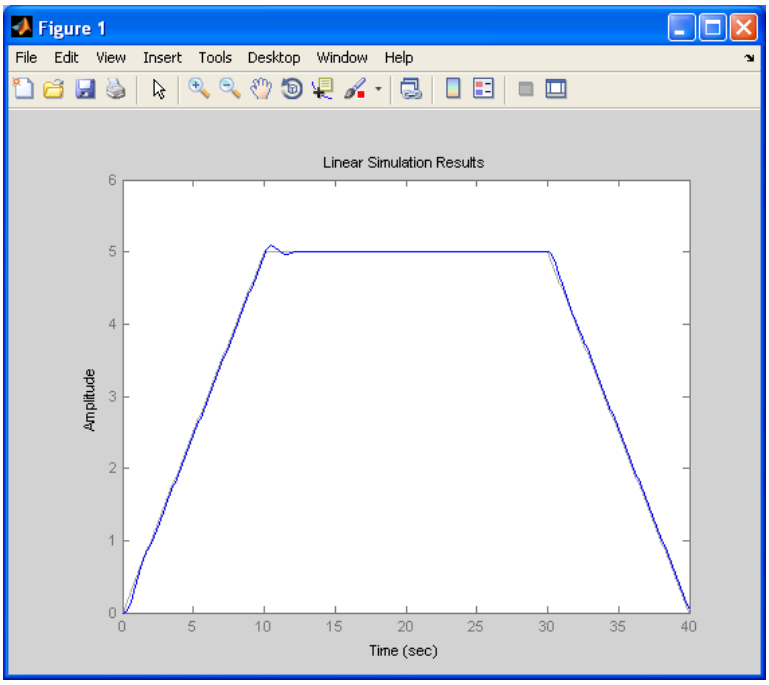
El resultado debe ser un gráfico similar al siguiente:



Una vez comprobado que los vectores se han definido correctamente, podemos lanzar la instrucción **lsim**:

```
» lsim(sis1,u,t)
```

Y el resultado será un gráfico con la respuesta que ofrece nuestro sistema **sis1** a la entrada anterior:



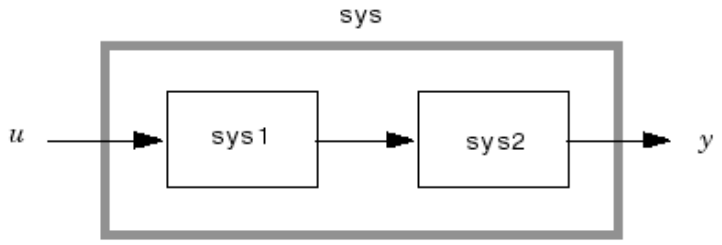
Podemos ver cómo la salida del sistema reproduce aproximadamente la entrada al mismo.

5. REDUCCIÓN DE DIAGRAMAS DE BLOQUES CON MATLAB

Matlab también permite obtener sistemas equivalentes para las principales combinaciones de bloques: en serie, en paralelo y en realimentación. De este modo, es posible reducir los diagramas de bloques.

Equivalente de dos bloques en serie:

Es el producto de los dos bloques, en Matlab se obtiene con el comando **series**:

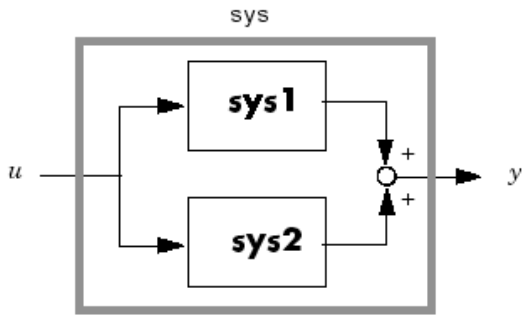


Supuestos definidos los sistemas sys1 y sys2, teclearíamos en Matlab:

```
» sistema_equivalente = series (sys1, sys2)
```

Equivalente de dos bloques en paralelo:

Es la suma de los dos bloques, en Matlab se obtiene con el comando **parallel**:

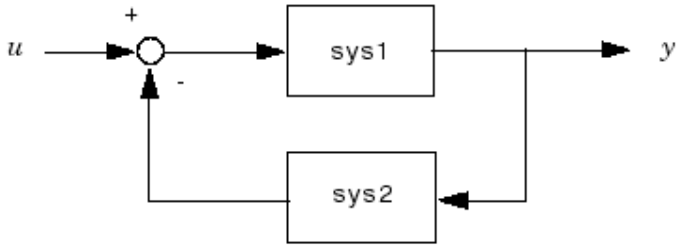


Supuestos definidos los sistemas sys1 y sys2, teclearíamos en Matlab:

```
» sistema_equivalente = parallel (sys1, sys2)
```

Equivalente para un esquema de realimentación:

Se calcula en Matlab mediante el comando **feedback**:

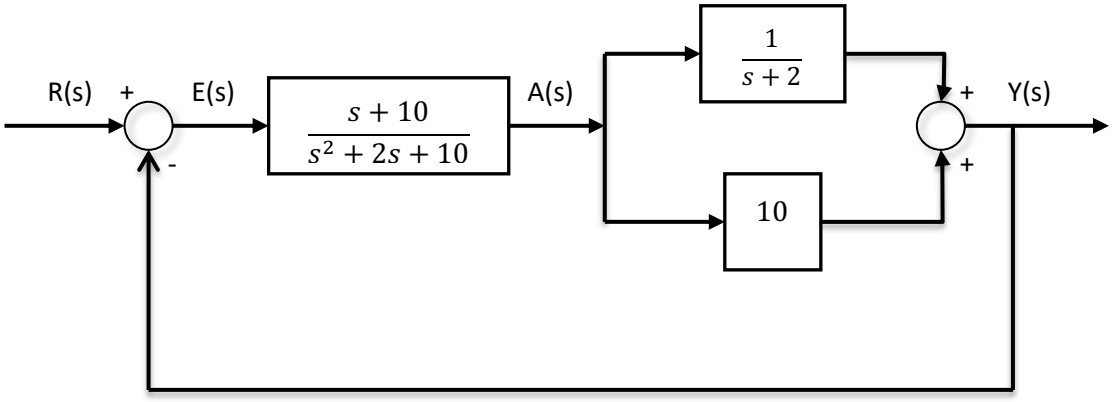


Supuestos definidos los sistemas sistema_1 y sistema_2, teclearíamos en Matlab:

```
» sistema_equivalente = feedback (sys1, sys2)
```

Ejemplo:

Dado el siguiente diagrama de bloques obtener el sistema reducido equivalente:




```
» sis1 = tf([1 10], [1 2 10])
» sis2 = tf([1], [1 2])
» sis3 = 10

» sistema = feedback( series(sis1, parallel(sis2, sis3)), 1)

Transfer function:
   10 s^2 + 121 s + 210
--------------------------
s^3 + 14 s^2 + 135 s + 230
```

6. TRANSFORMADAS Y ANTITRANSFORMADAS

Matlab permite obtener transformadas y antitransformadas de Fourier y Laplace mediante su módulo de matemática simbólica.

Procedimiento:

- Declarar una variable simbólica con la instrucción **syms**
- Obtener la transformada para una expresión definida utilizando la variable simbólica anterior

Instrucciones de Matlab correspondientes a cada una de las transformadas:

- **fourier** transformada de Fourier
- **ifourier** transformada inversa de Fourier
- **laplace** transformada de Laplace
- **ilaplace** transformada inversa de Laplace

Ejemplo:

Obtendremos la antitransformada de Laplace de la siguiente expresió $F(s) = \frac{2}{s \cdot (s+0.5)}$n:

Las instrucciones de Matlab que utilizaremos serán:

```
» syms s
» ilaplace (2/(s*(s+0.5)))
ans =
      4-4*exp(-1/2*t)
```

Por lo tanto, hemos obtenido como respuesta: $f(t) = 4 - 4 \cdot e^{-0.5t}$

Comprobación: haciendo el procedimiento inverso buscaremos la transformada de Laplace de f(t):

```
» syms t
» laplace (4-4*exp(-0.5*t))
ans =
      4/s-4/(s+1/2)
```

Se obtiene como resultado:

$$F(s) = \frac{4}{s} - \frac{4}{s+0.5} = \frac{4(s+0.5) - 4s}{s \cdot (s+0.5)} = \frac{2}{s \cdot (s+0.5)}$$

... que es la misma función F(s) de partida

Como ejercicio, se obtendrán antitransformadas de Laplace para las siguientes funciones:

$$F_1(s) = \frac{s^2 + 2s + 3}{(s+1)^3} \qquad f_1 = t^2 \cdot e^{-t} + e^{-t}$$

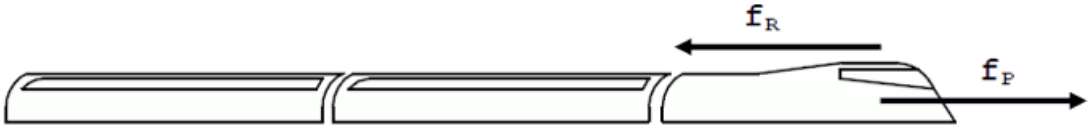
$$F_2(s) = \frac{5s + 5}{s \cdot (s^2 + 2s + 5)} \qquad f_2 = 1 - e^{-t}[\cos(2t) - 2 \cdot sen(2t)]$$

EJERCICIO 1: TREN-BALA

Se desea estudiar el comportamiento de un tren-bala a altas velocidades. Se supondrá que existen dos fuerzas antagónicas:

***f_P*:** fuerza de propulsión producida por el motor

***f_R*:** fuerza de resistencia aerodinámica que se opone al avance



Las ecuaciones de comportamiento del sistema son las siguientes:

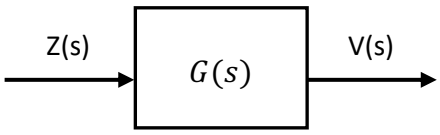
$$f_R(t) = K_R \cdot v^2(t)$$
$$f_P(t) = K_P \cdot z(t)$$
$$M \cdot \frac{dv(t)}{dt} = f_P(t) - f_R(t)$$

v(t) = velocidad del tren (m/s)
z(t) = posición palanca de mando (mm)
K_R = constante resistencia aerodinámica = **4** Ns^2/m^2
K_P = constante del propulsor = **1000** N/mm
M = masa del tren = **5000** Kg

Se pide:

- Operar como en el ejercicio anterior (muelle) hasta llegar a la función de transferencia que relaciona la velocidad del tren con la posición de la palanca de mando del motor. El sistema es **NO lineal,** por lo que se analizará el problema sobre el sistema linealizado entorno a la posición de equilibrio correspondiente a una posición de la palanca de mando = **40 mm**.

Dado que en este caso no partimos de una única ecuación sino de varias ecuaciones y por tanto de varios bloques, caben dos posibilidades: representar el diagrama de bloques en Simulink o reducir el mismo hasta obtener un único bloque y representar sólo este bloque en Simulink. En cualquier caso, la variable **Z(s)** será la entrada y la variable **V(s)** la salida:



- Utilizar como entrada un escalón de **2 unidades** (cambio brusco de 2 mm. en la palanca de mando) y comprobar cuál es el efecto en la velocidad del tren. Se deberán obtener los siguientes valores a partir de las medidas tomadas sobre el gráfico:
  - ✓ Velocidad final alcanzada por el tren.
  - ✓ Tiempo que tarda el tren en alcanzar la mitad del incremento de velocidad total.

- Repetir los cálculos del apartado anterior (escalón de 2 unidades) obteniendo la función de transferencia resultante para cada uno de estos grupos de valores:
  - ✓ **M = 10000, K_R = 4, K_P=1000**
  - ✓ **M = 5000, K_R = 16, K_P=1000**
  - ✓ ¿En qué aspectos coinciden y en qué aspectos difieren estos sistemas con respecto al primero?

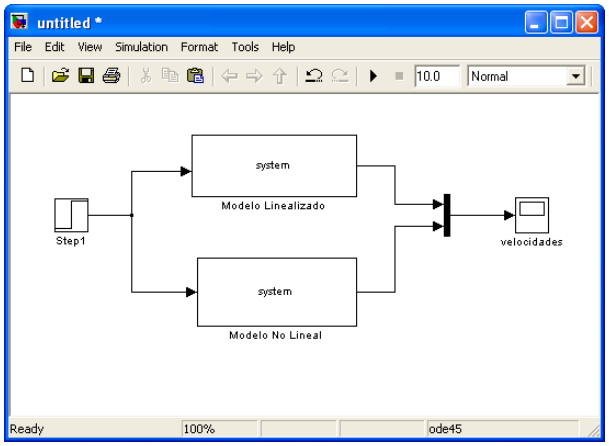
Resultados a obtener con Matlab en cada uno de los casos:

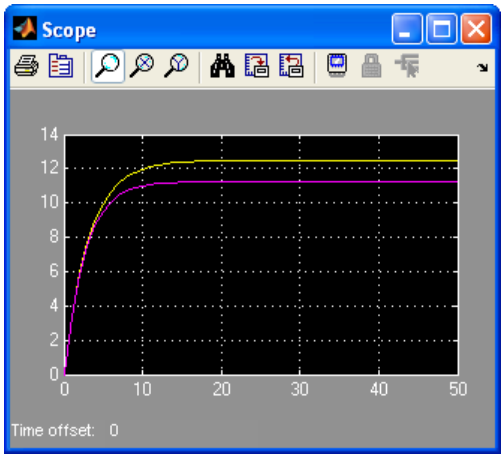
- Gráficos de la velocidad alcanzada por el tren para cada apartado. Los gráficos se obtendrán con la instrucción **plot** tras haber sido llevados a una variable **Matlab** desde el bloque **Scope**.
- Comprobación de que los resultados teóricos obtenidos para cada apartado coinciden con los de Simulink.

EJERCICIO 2: ERRORES COMETIDOS CON LA LINEALIZACIÓN

Sobre el mismo ejemplo del ejercicio anterior (tren bala) se desea conocer cuál es el error cometido en el proceso de linealización. Para ello se hará lo siguiente:

- Conservar el esquema Simulink del último apartado (M = 5000, K_R = 16, K_P = 1000).
- En la misma ventana, crear un esquema Simulink que represente el comportamiento del mismo sistema sin linealizar. En este caso no se puede hacer uso del bloque función de transferencia, y deberá representarse directamente la ecuación diferencial (como en la primera práctica de Simulink).
- Aplicar la misma entrada escalón (**amplitud 20**) a los dos sistemas y representar las dos salidas en un mismo osciloscopio; el resultado debe ser similar al que se muestra:





- Comprobar las diferencias de comportamiento obtenidas para distintos valores del escalón de entrada. ¿Por qué los errores son mayores cuanto mayor es el valor del escalón?

AYUDA:

- La función '**elevar al cuadrado**' se obtiene con el bloque '***Math function***' perteneciente a la categoría '***Math Operations***'.

- Para comparar valores correctamente, utilizaremos como variables de entrada y salida los valores incrementales de $\Delta v(t)$ y $\Delta z(t)$. Deberemos por tanto realizar la conversión a la entrada y salida en el sistema no lineal de estas variables utilizando un bloque constante.

$$v(t) = v(0) + \Delta v(t)$$
$$z(t) = z(0) + \Delta z(t)$$

- Para introducir los términos del punto de equilibrio ($v(0)$, $z(0)$) utilizaremos el bloque '***Constant***' perteneciente a la categoría '***Math Operations***'.

- Al simularse respecto a un punto de equilibrio no nulo, en el integrador del sistema no lineal debe configurarse el ***valor inicial*** $v(0)$: