Programación en Matlab

Reducción de Bloques

Ing. Eddie Ángel
Sobrado Malpartida

-

Combinación de modelos: en serie, paralelo o realimentados

Un modelo puede ser representado por un bloque con entradas y salidas ya sea conteniendo una función de transferencia o un modelo en espacio estado dentro de él. Las funciones *series*, *parallel*, y *feedback* pueden ser usadas para realizar manipulaciones básicas de bloques como se mostrará a continuación

Existen una serie de comandos relacionados con las operaciones típicas en diagramas de bloques:

[N12 , D12] = series (N1,D1,N2,D2)

Devuelve la resultante de colocar **en serie** dos funciones de transferencia. El mismo resultado podría obtenerse llamando dos veces al comando *conv*(), que recuérdese permitía multiplicar dos polinomios.

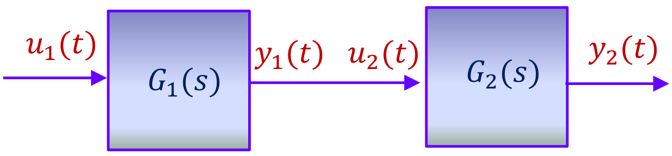


Figura 1: Conexión en serie

[N12 , D12] = parallel (N1,D1,N2,D2)

Devuelve la resultante de colocar **en paralelo** dos funciones de transferencia.

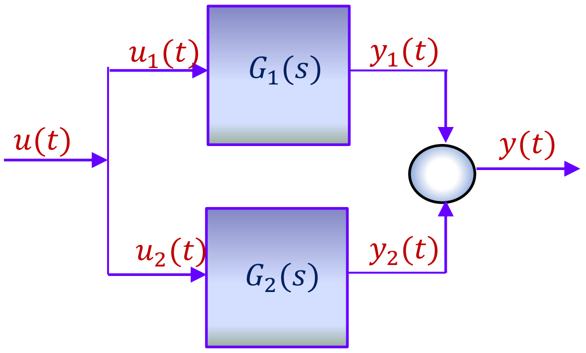


Figura 2: Conexión en paralelo

A partir de un sistema en bucle abierto, dado por el numerador y denominador N1,D1, proporciona el correspondiente en bucle cerrado, considerando que en la cadena de realimentación hay otra función de transferencia, dada por N2, D2. El ultimo parámetro indica el signo de la realimentación (−1 para realimentación negativa y 1 para positiva).

```
[Nbc,Dbc] = feedback (N1,D1,N2,D2,-1)
```

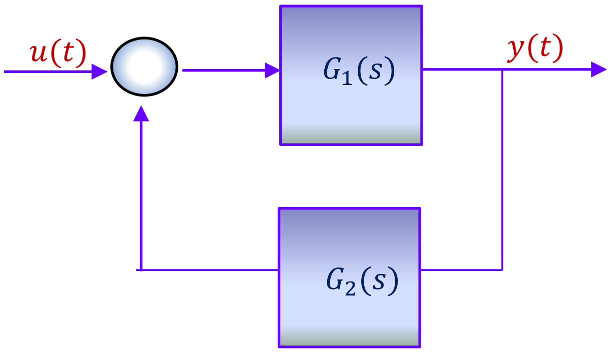


Figura 3: Conexión en realimentación

En el caso en que se pretenda obtener la función de transferencia en bucle cerrado con realimentación unitaria, puede emplearse este comando más compacto, en el que se evita tener que especificar una segunda función de transferencia.

```
[Nbc,Dbc] = cloop (N1,D1,-1)
```

Para operaciones de bloques más complejas, resulta más adecuado usar la herramienta ***SIMULINK***.

Ejercicios

Como ejemplo asuma dadas las siguientes funciones de transferencia:

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

$$H_b(s) = \frac{3}{s+4}$$

Primero creemos los dos sistemas anteriores:

```
k1=1;
k2=3;
Ha=tf(k1,[1,2]);
Hb=tf(k2,[1,4]);
```

Conectemos $H_a(s)$ y $H_b(s)$ en serie, eso significa que $H_{ser}(s) = H_a(s) * H_b(s)$

```
Hser=series(Ha,Hb)
```

```
Hser =
        3
  -------------
  s^2 + 6 s + 8
```

A continuación, $H_a(s)$ y $H_b(s)$ son conectados en paralelo, lo cual significa que

$H_{par}(s) = H_a(s) + H_b(s)$

```
Hpar=parallel(Ha,Hb)
```

```
Hpar =

     4 s + 10
  -------------
  s^2 + 6 s + 8
```

Finalmente, $H_a(s)$ y $H_b(s)$ son conectados en un lazo realimentado negativo con $H_a(s)$ en el lazo directo y $H_b(s)$ en el lazo de realimentación:

```
feedbsign=-1;
Hfeedb=feedback(Ha,Hb,feedbsign)
```

feedbsign es el signo de la realimentación, y lo establecemos como -1 o 1. La función de transferencia Hfeedb(s) está dada por:

$$Hfeedb(s) = \frac{H_a(s)}{1 - feedsign * H_a(s) * H_b(s)}$$

La respuesta de Matlab está dada por:

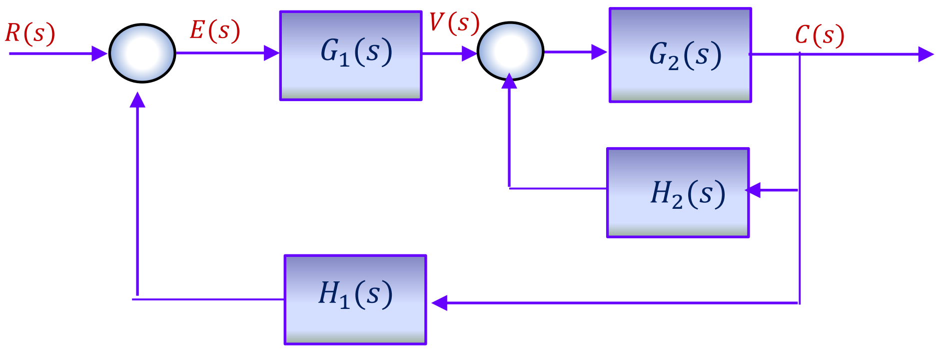
Transfer function:

```
s + 4
--------------
s^2 + 6 s + 11
```

Tarea: Reducción de bloques

Función transferencia a lazo cerrado

Supongamos que disponemos del sistema de la Figura 1



donde

$$G_1(s) = 0.4$$

$$G_2(s) = \frac{100}{s(s+2)}$$

$$H_2(s) = \frac{s}{s+20}$$

$$H_1(s) = 1$$

y pretendemos hallar la función de transferencia en lazo cerrado $\frac{C(s)}{R(s)}$

Si aplicamos reducción de bloques obtenemos:

$$G_2(s) = \frac{40s + 800}{s^3 + 22s^2 + 180s + 800}$$

La función transferencia a lazo cerrado se puede calcular de dos formas:

a. Utilizando las funciones de MATLAB *series(), parallel(), feedback()* y *cloop().*
b. Utilizando SIMULINK

En Matlab

Para calcular la función transferencia a lazo cerrado M(s) sigamos los siguientes pasos:

1. Definamos los numeradores y denominadores de las funciones transferencia de cada bloque de la siguiente forma:

```
numg1=0.4;
deng1=1;
numg2=100;
deng2=[1 2 0];
numh2=[1 0];
denh2=[1 20];
```

2. Calculemos la función transferencia de V (s) a C(s):

```
[numvc,denvc]=feedback(numg2,deng2,numh2,denh2,-1);
```

3. Ahora calculemos la función transferencia de E(s) a C(s) con:

```
[numec,denec]=series(numg1,deng1,numvc,denvc);
```

4. Por ultimo calculemos el lazo cerrado:

```
[num, den]=cloop(numec,denec,-1);

Planta=tf(num, den)
```

Lo que obtuvimos son los vectores numerador y denominador por separado, luego recordemos que para ingresarla como función de transferencia a matlab, debemos utilizar *tf()* y se obtuvo:

```
Planta =

          40 s + 800
  --------------------------
  s^3 + 22 s^2 + 180 s + 800

Continuous-time transfer function.
```