UNIVERSIDAD AUTÓNOMA DEL ESTADO DE MÉXICO

Facultad de Ingeniería: Ingeniería en Electrónica

Control I

PRÁCTICA 2 Representación de Sistemas en MatLab (2ª parte)

OBJETIVO

Que el alumno obtenga y simule algunos diagramas de bloques de algunos sistemas.

Mostrar la misma señal de salida con distintos métodos y funciones.

MATERIAL Y EQUIPO PARA UTILIZAR

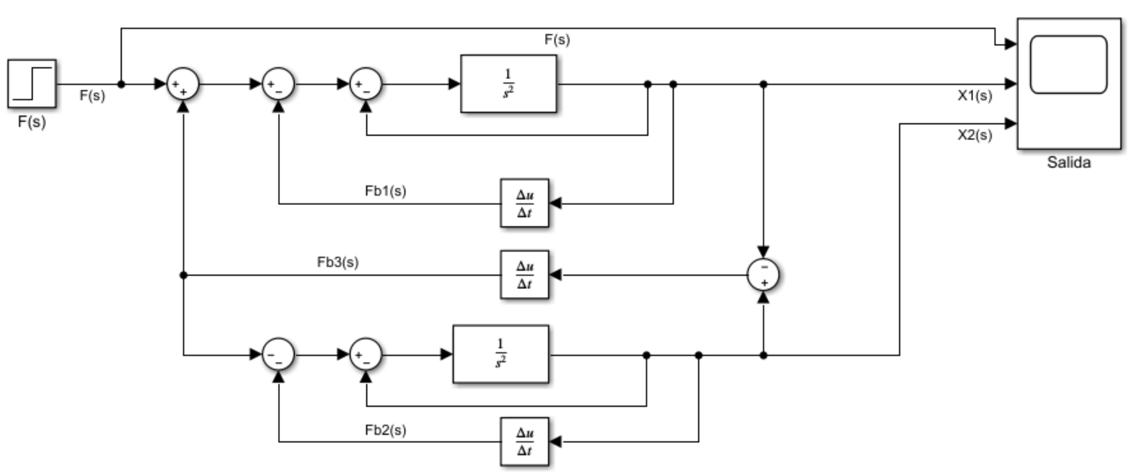
- Computadora.
- Software MatLab versión 9.3.0.7
- Sistema de almacenamiento de datos y/o impresora.

1| INTRODUCCIÓN

En los siguientes pasos de esta práctica se trabajarán las funciones de transferencia en sus formas polinomial y ceros y polos del sistema mecánico analizado en la Práctica 1, así como la respectiva ecuación diferencial y el correspondiente diagrama de bloques por componentes que describe al sistema mecánico. El enfoque principal de esta práctica es el uso del software MatLab y Simulink para la simplificación y simulación de diagramas de bloques y representaciones matemáticas del sistema mecánico; así como algunos otros sistemas descritos por diagramas de bloques para poner en practica el uso del software MatLab como una herramienta en el análisis de sistema.

2| DESARROLLO

1. Obtenga el diagrama de bloques por componentes del sistema considerado en el paso 1 de la Práctica 1.



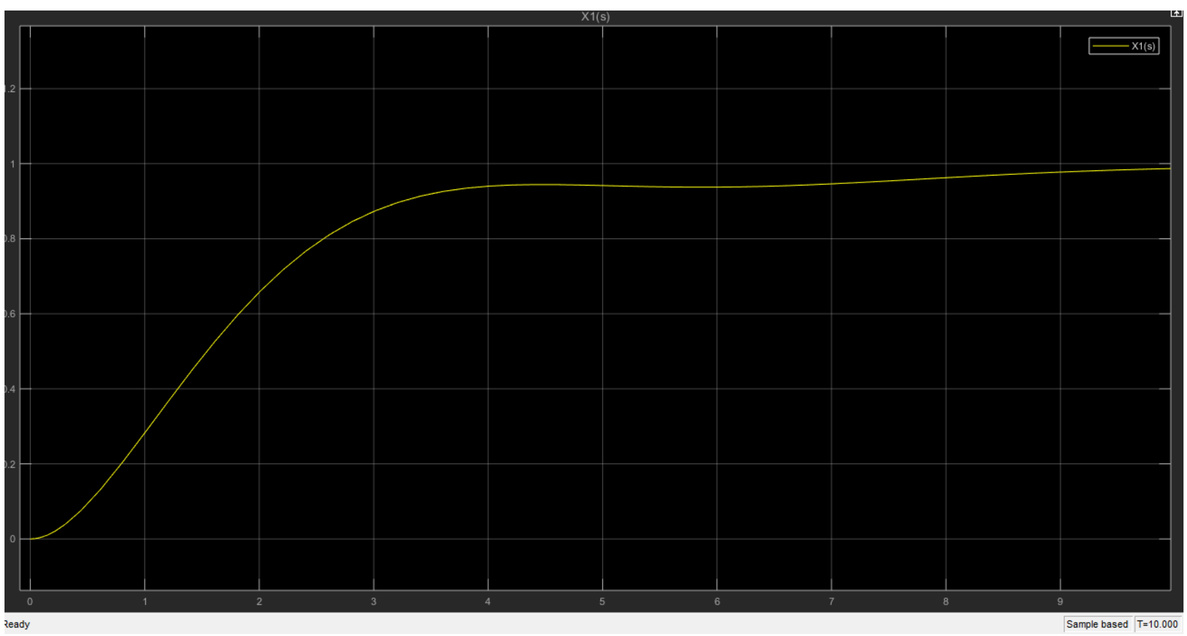
La explicación y solución a la obtención del diagrama de bloques por componentes del sistema mecánico se describen en el *Anexo 1.*

2. Simule en Simulink el diagrama por componentes obtenido en el paso 1 considerando una entrada escalón unitario.

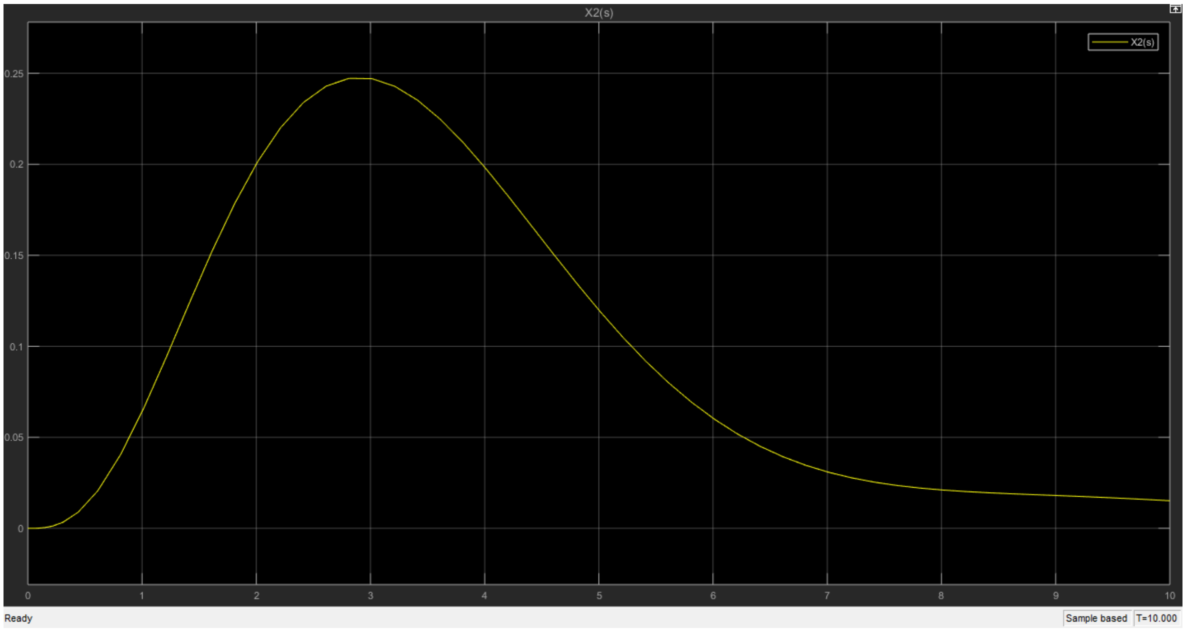
En las imágenes siguientes se muestran las gráficas correspondientes a la señal de entrada $F(t)$ escalón unitario y a las salidas $x_1(t)$ y $x_2(t)$ obtenidas en la simulación.

- Para la señal F(s):

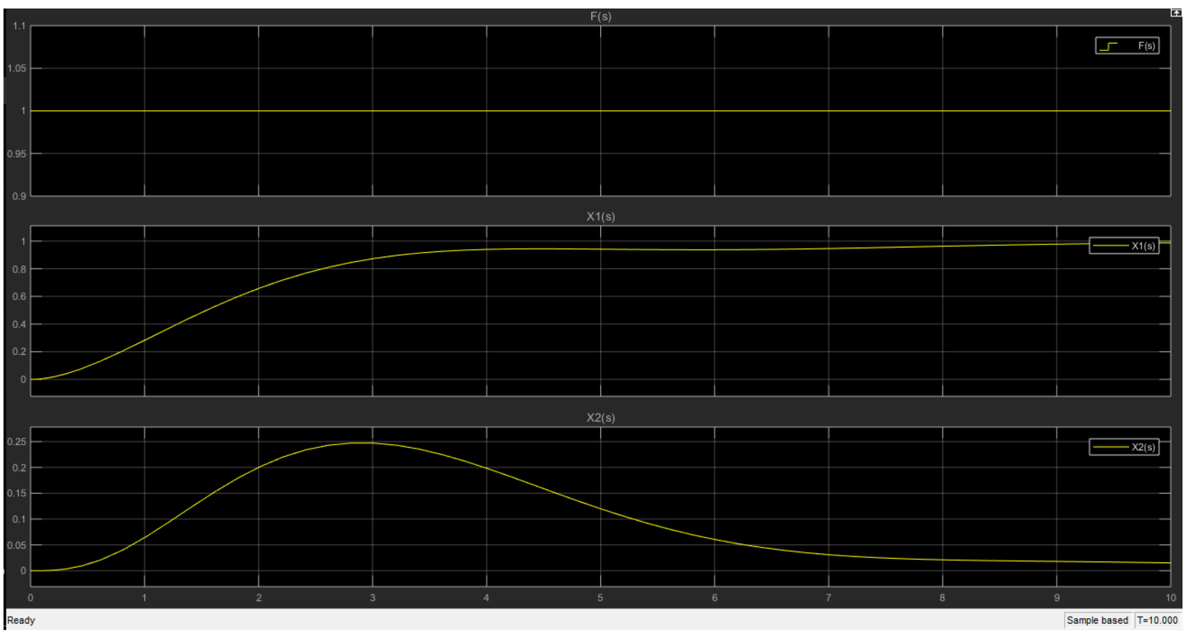
- Para la señal $x_1(s)$



- Para la señal $x_2(s)$

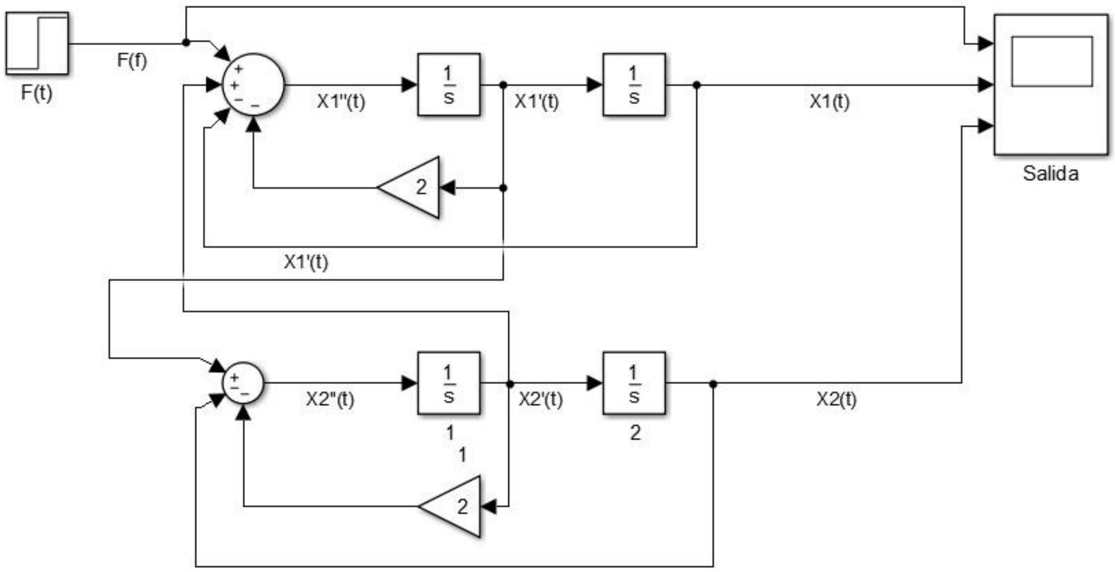


- Para F(s), $x_1(s)$ y $x_2(s)$

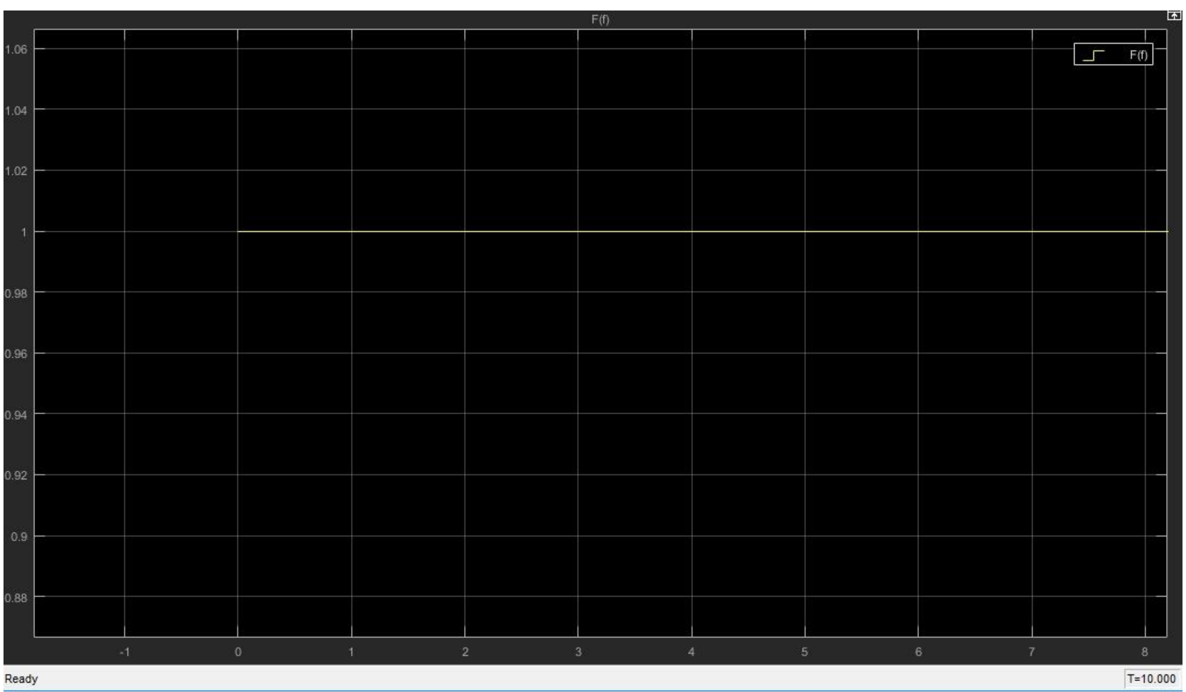


3. Simule en Simulink las representaciones obtenidas (ecuaciones diferenciales, funciones de transferencia, modelos de polos y ceros) en los pasos 2 y 5 de la Práctica 1 considerando una entrada escalón unitario.

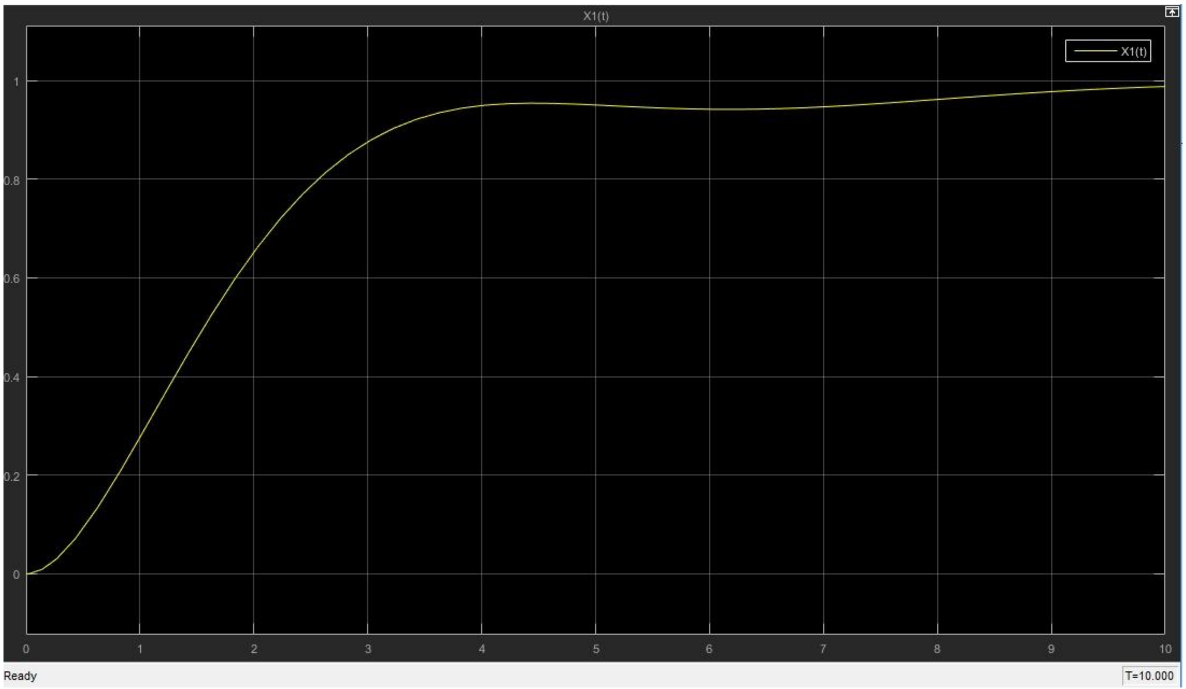
3.1 Simulación de la Ecuación Diferencial del Sistema en Simulink



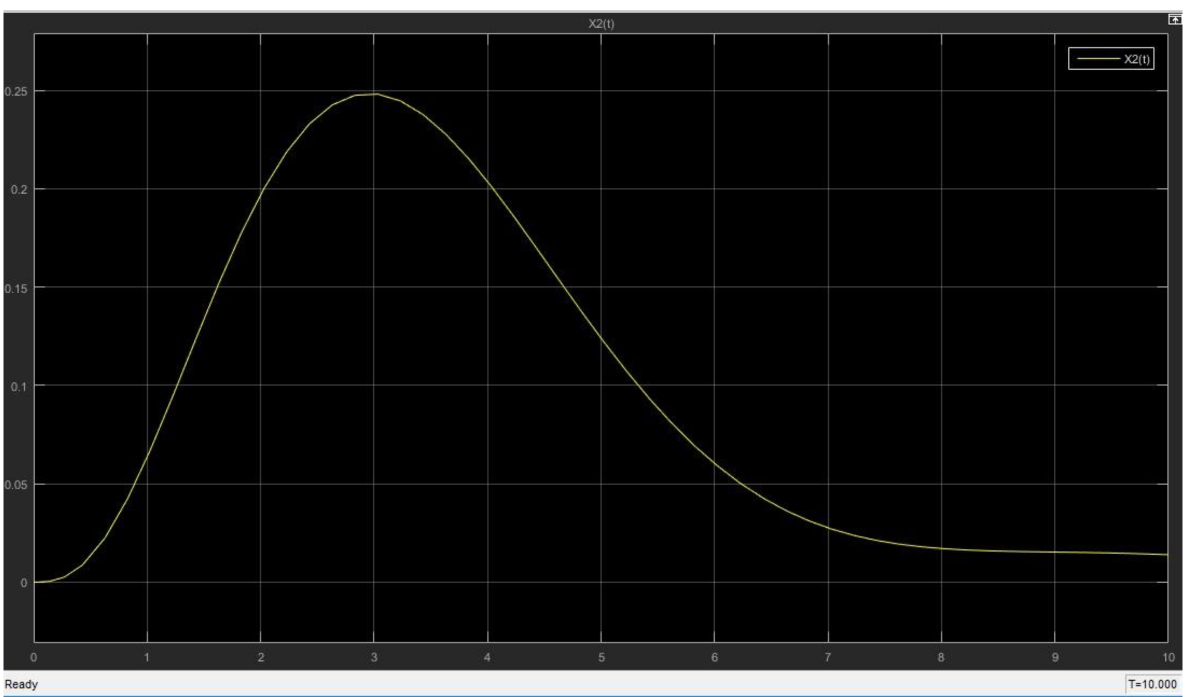
- Para la señal F(t):



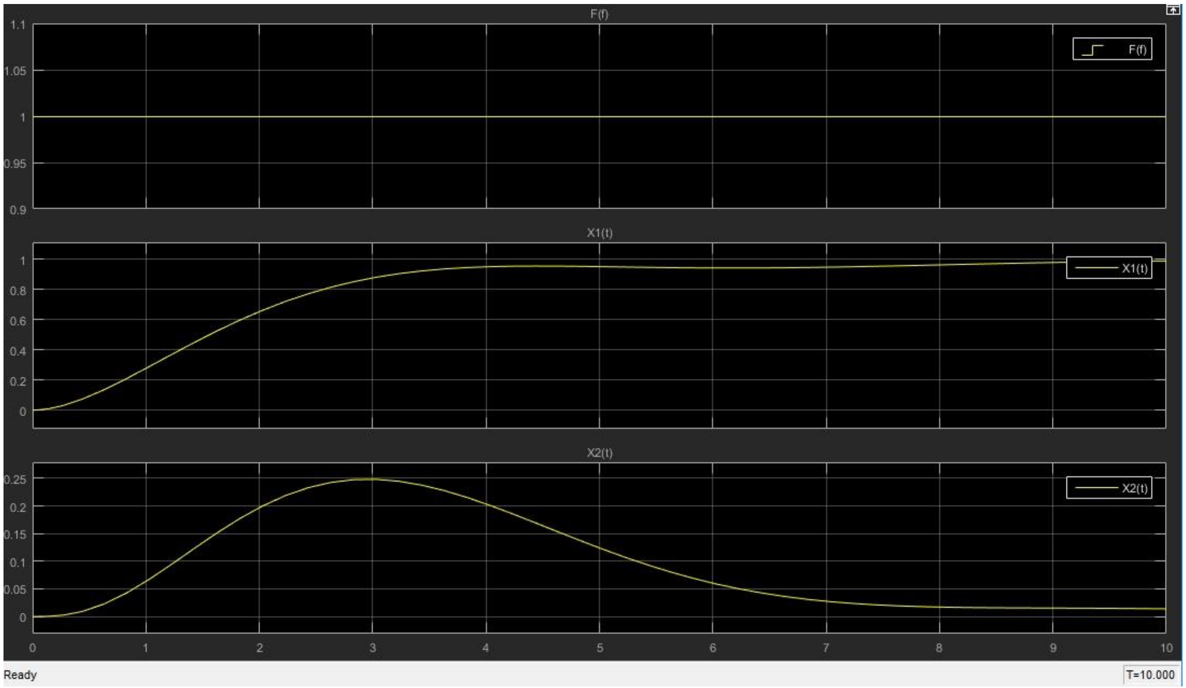
- Para la señal $x_1(t)$:



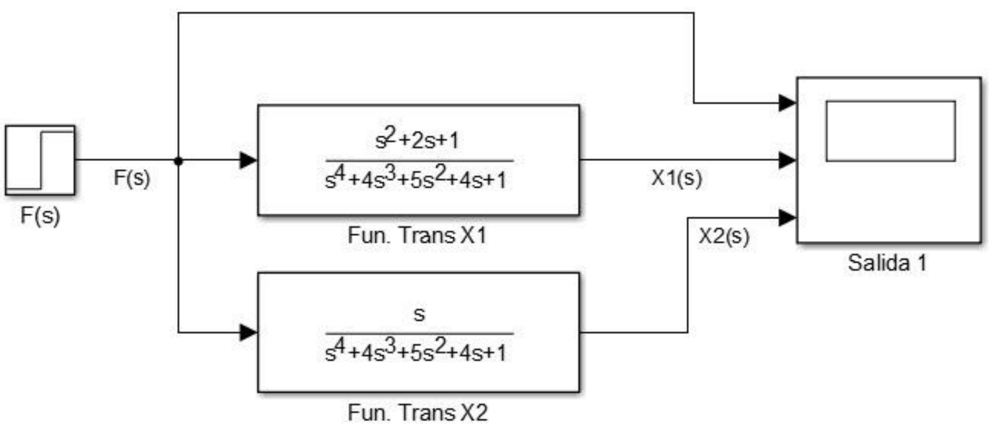
- Para la señal $x_2(t)$:



- Para la señal F(t), $x_1(t)$ y $x_2(t)$

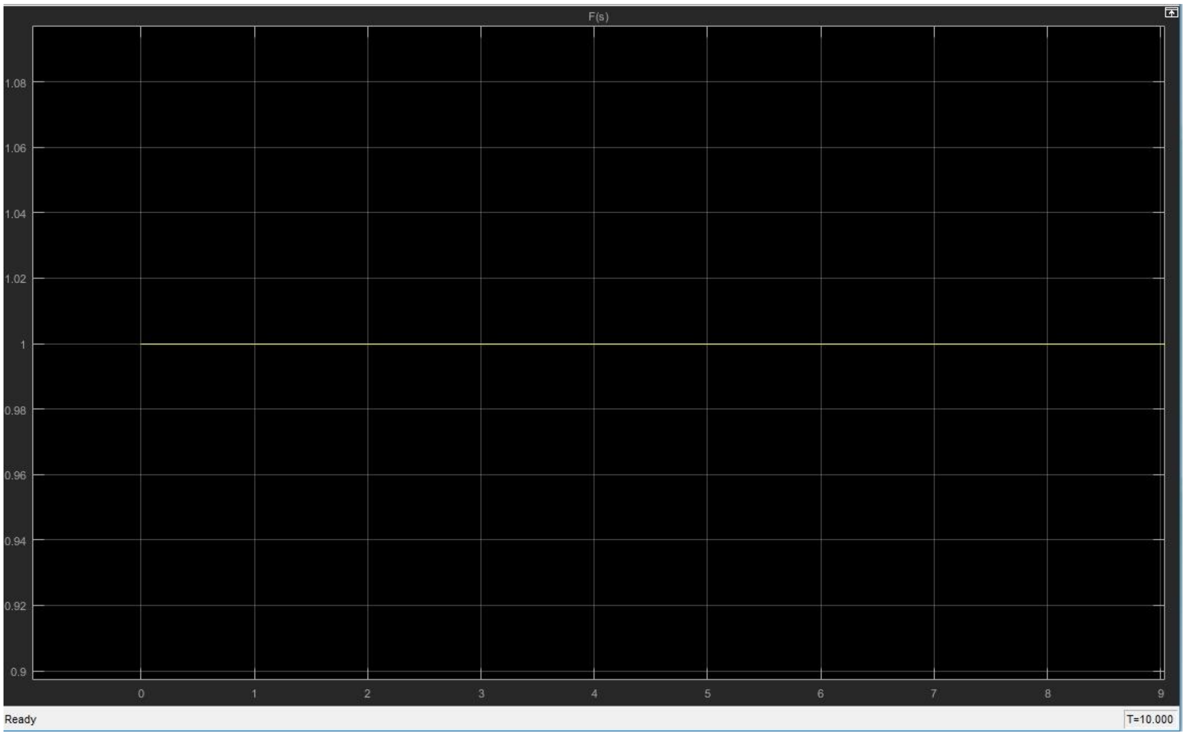


3.2 Simulación de las funciones de transferencia en forma polinomial en Simulink

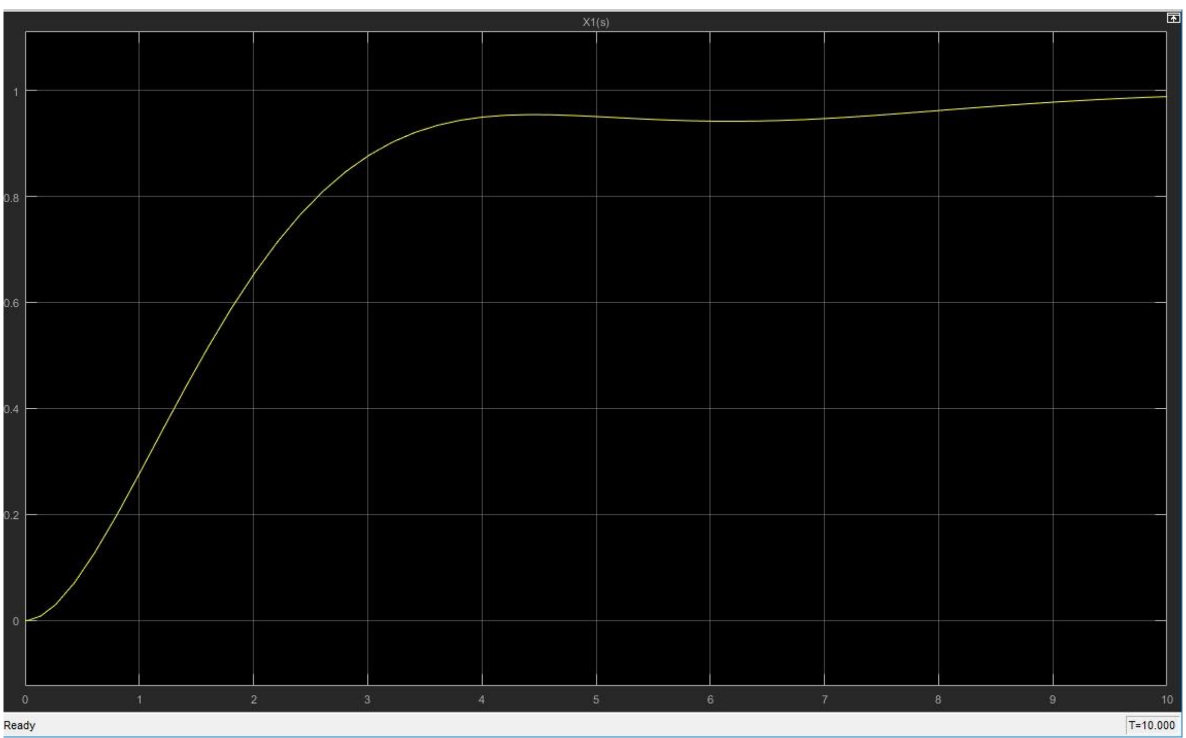


Las siguientes imágenes muestran las señales de entrada y salida que se obtuvieron a través de Simulink, con base en las funciones de transferencia del sistema.

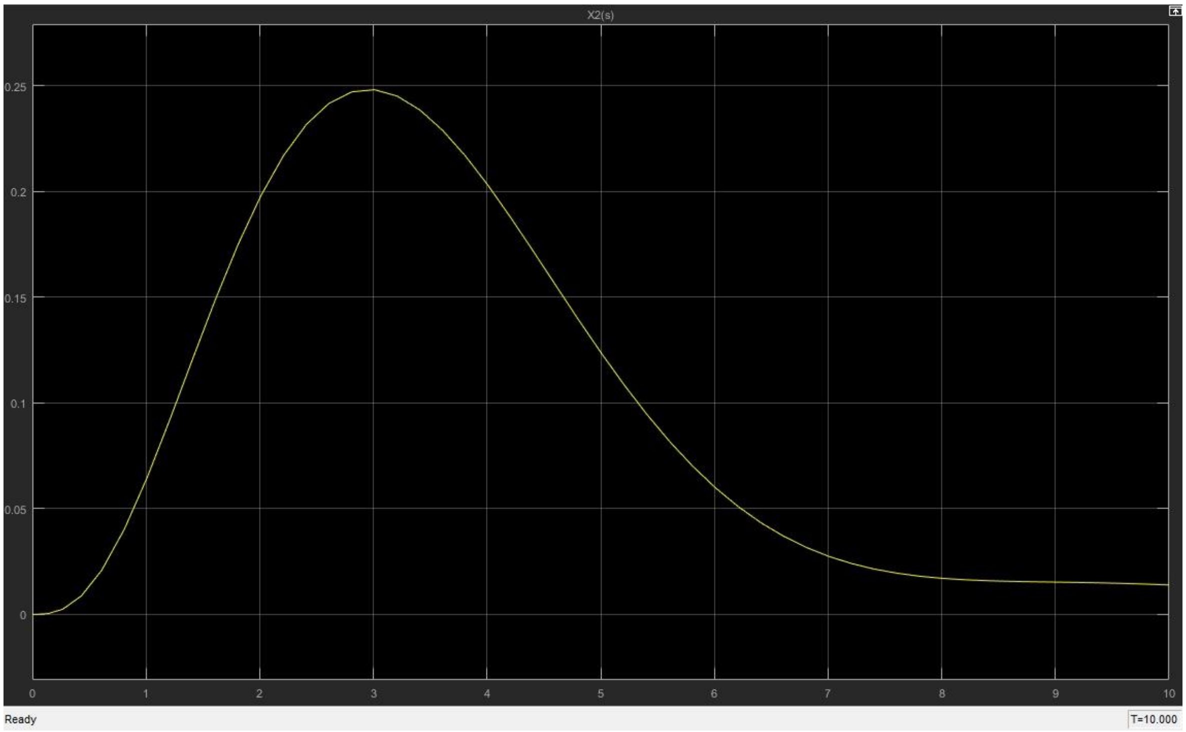
- Para la señal F(s):



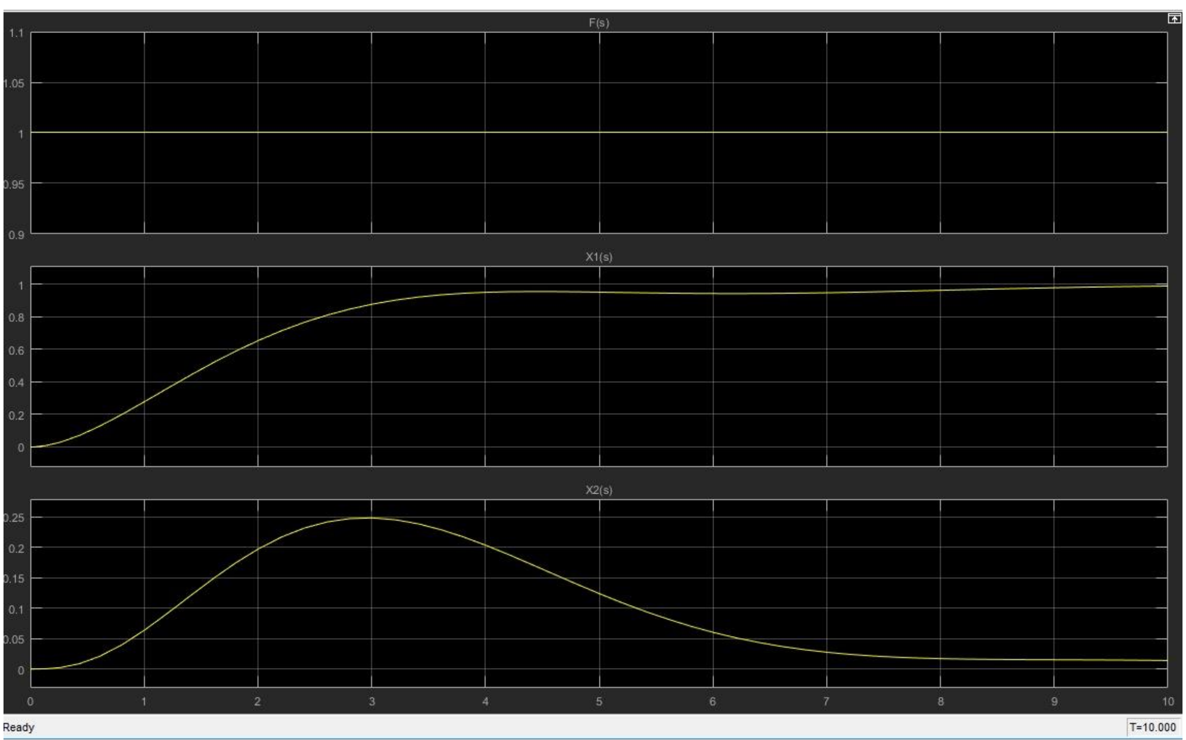
- Para la señal $x_1(s)$:



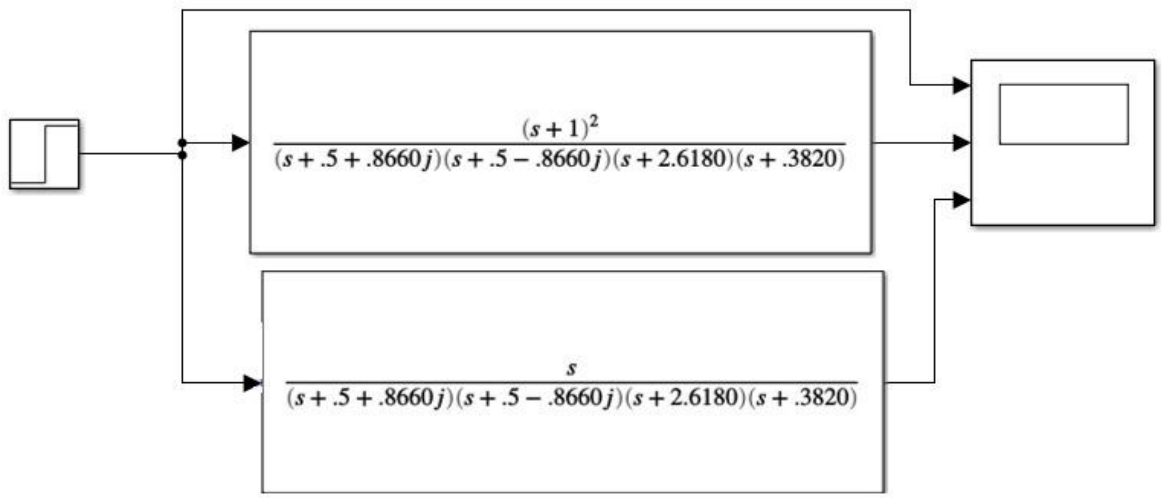
- Para la señal $x_2(s)$:



- Para la señal F(t), $x_1(s)$ y $x_2(s)$

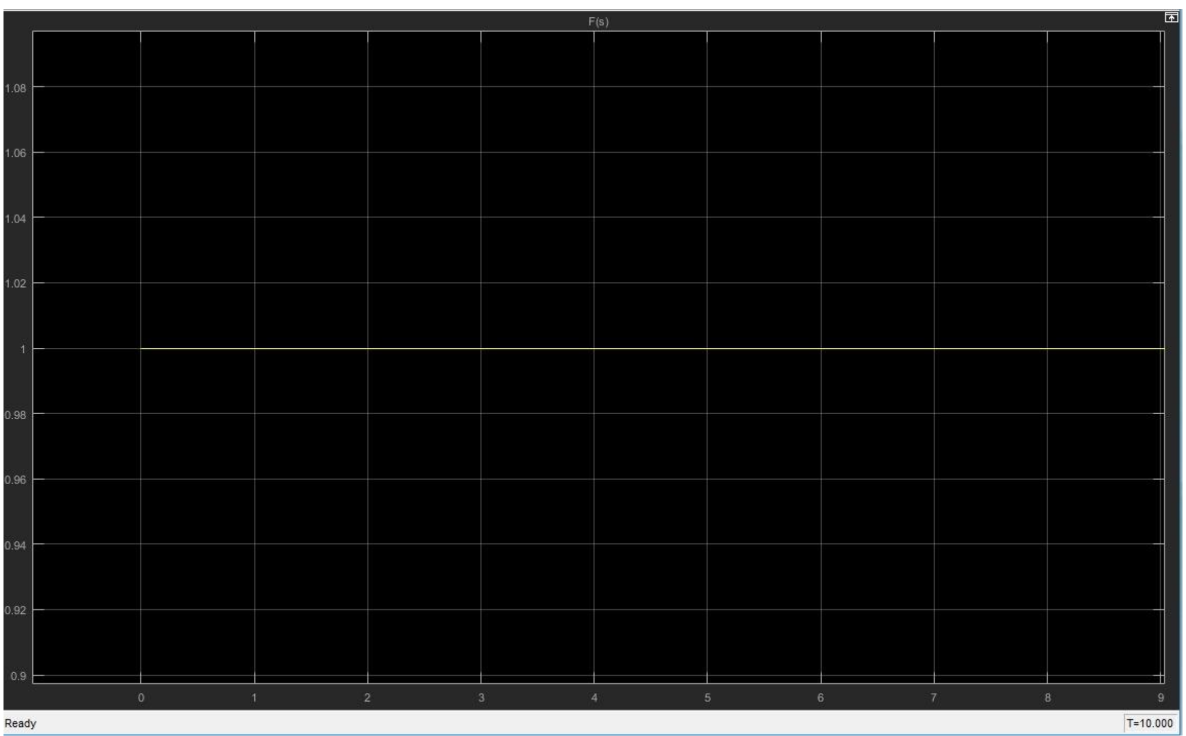


3.3 Simulación de las funciones de transferencia en forma polos y ceros en Simulink

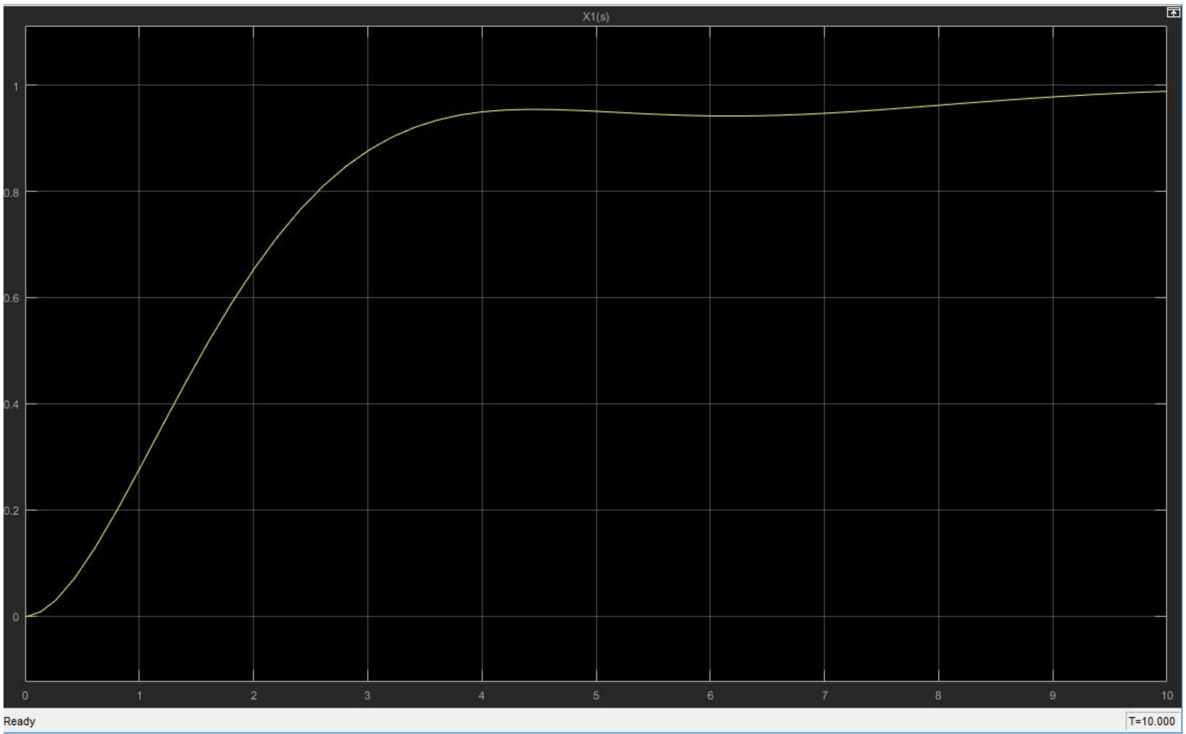


Las gráficas siguientes muestran el comportamiento de cada una de las señales sobre el sistema en forma de celos y polos:

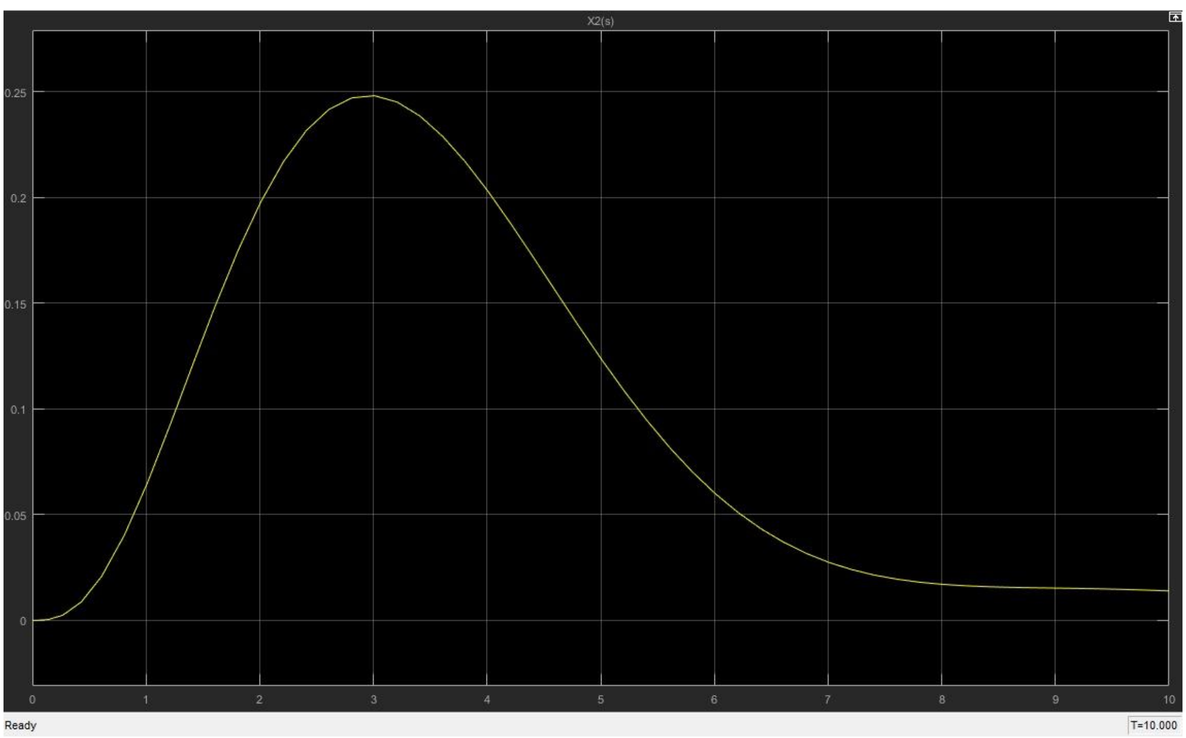
- Para la señal F(s):



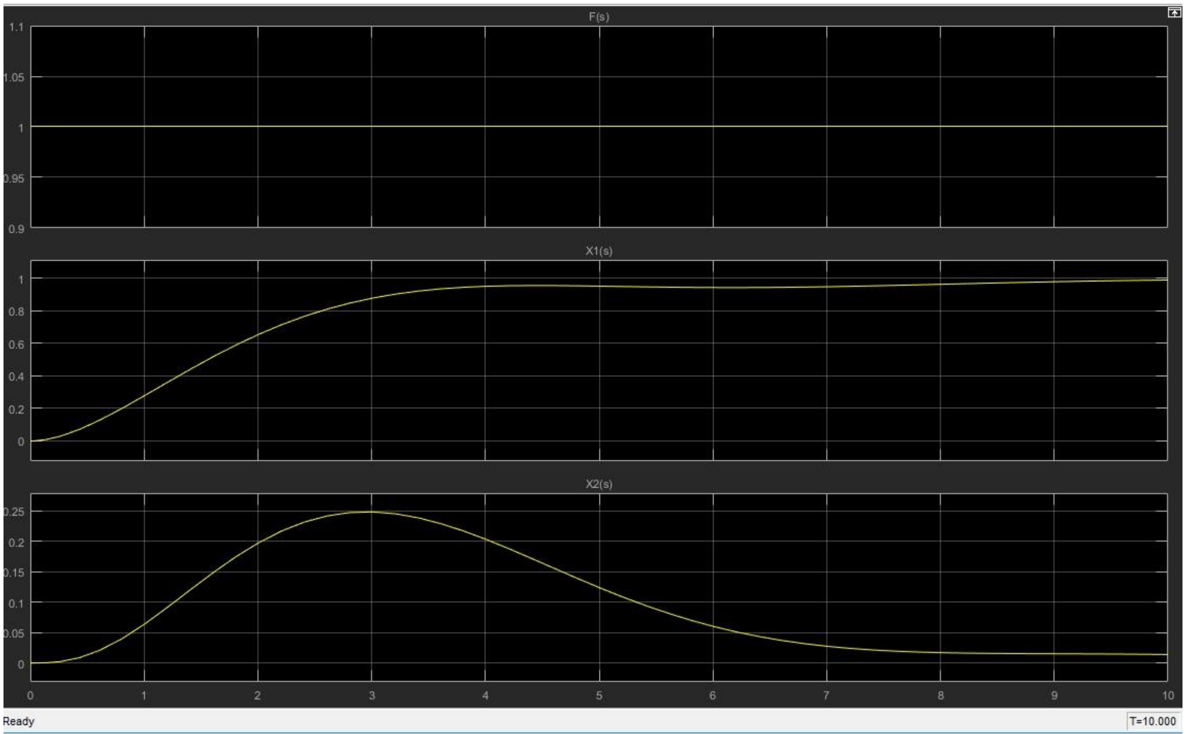
- Para la señal $x_1(s)$:



- Para la señal $x_2(s)$:



- Para la señal F(t), $x_1(s)$ y $x_2(s)$



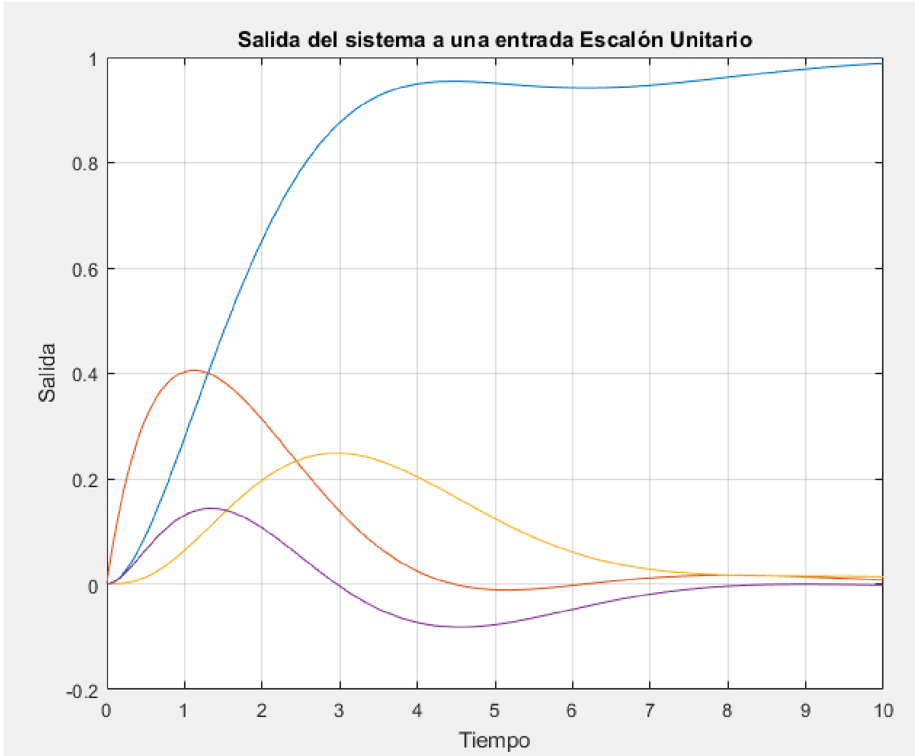
4. Obtenga en MatLab la respuesta del sistema usando las representaciones obtenidas (ecuaciones diferenciales, funciones de transferencia, modelos de polos y ceros) en los pasos 2 y 5 de la Práctica 1 considerando una entrada escalón unitario.

El primer paso para poder simular la salida del sistema a una entrada escalón unitario mediante la ecuación diferencial del sistema es utilizando el comando *"function"* del apartado *New.* La función creada se describe en el código siguiente:

```
function dz=system(t,z)
%Representación de la EDO del sistema mecánico de la Práctica 1.
% Uso de la EDO para obtener la salida del sistema.
u=1; %entrada escalón unitario.
dz=zeros(4,1);
dz(1)=z(2);
dz(2)=-1*z(1)-2*z(2)+z(4)+u;
dz(3)=z(4);
dz(4)=z(2)-z(3)-2*z(4);
end
```

Para obtener la gráfica correspondiente a la salida del sistema usamos los comandos siguientes:

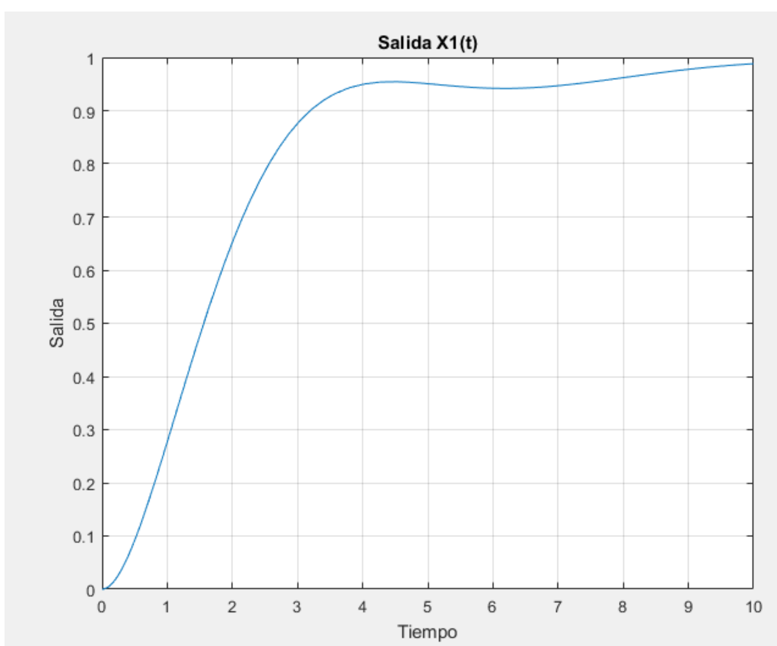
```
>> [tiempo,salida]=ode45('system',[0 10],[0 0 0 0]);
>> plot(tiempo,salida)
```

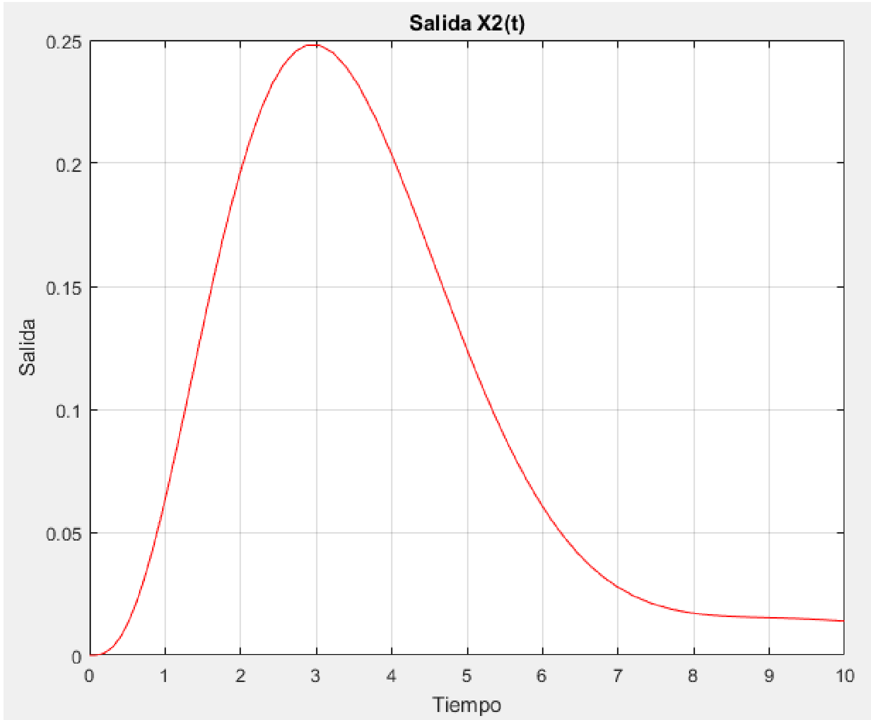


Cabe mencionar que este comando nos arroja dos salidas más que no son de nuestro interés por lo cual debemos usar los comandos siguientes para seleccionar las señales de interés $x_1(t)\ y\ x_2(t)$.

```
>> plot(tiempo,salida(:,1))
>> %Gráfica correspondiente a la salida X1(t)
>> plot(tiempo,salida(:,3))
>> %Gráfica correspondiente a la salida X2(t)
```

Usando los comandos anteriores obtenemos las gráficas siguientes





Estas dos últimas gráficas pueden ser comparadas con las gráficas obtenidas mediante la simulación de elaborada en Simulink. Como puede verse, las gráficas de $x_1(t)$ y $x_2(t)$ son las mismas que las obtenidas mediante la simulación de Simulink.

4.2 Representación de la salida del sistema usando las funciones de transferencia en forma polinomial del sistema.

Para poder identificar cada una de las funciones de transferencia de las salidas $x_1(s)$ y $x_2(s)$, se han asignado los nombres $G_1(s)$ y $G_2(s)$, respectivamente. Simplificando a $g1$ y $g2$, para facilitar la escritura en los comandos usados en Matlab. Los códigos siguientes describen a las funciones de transferencia $G_1(s)$ y $G_2(s)$.

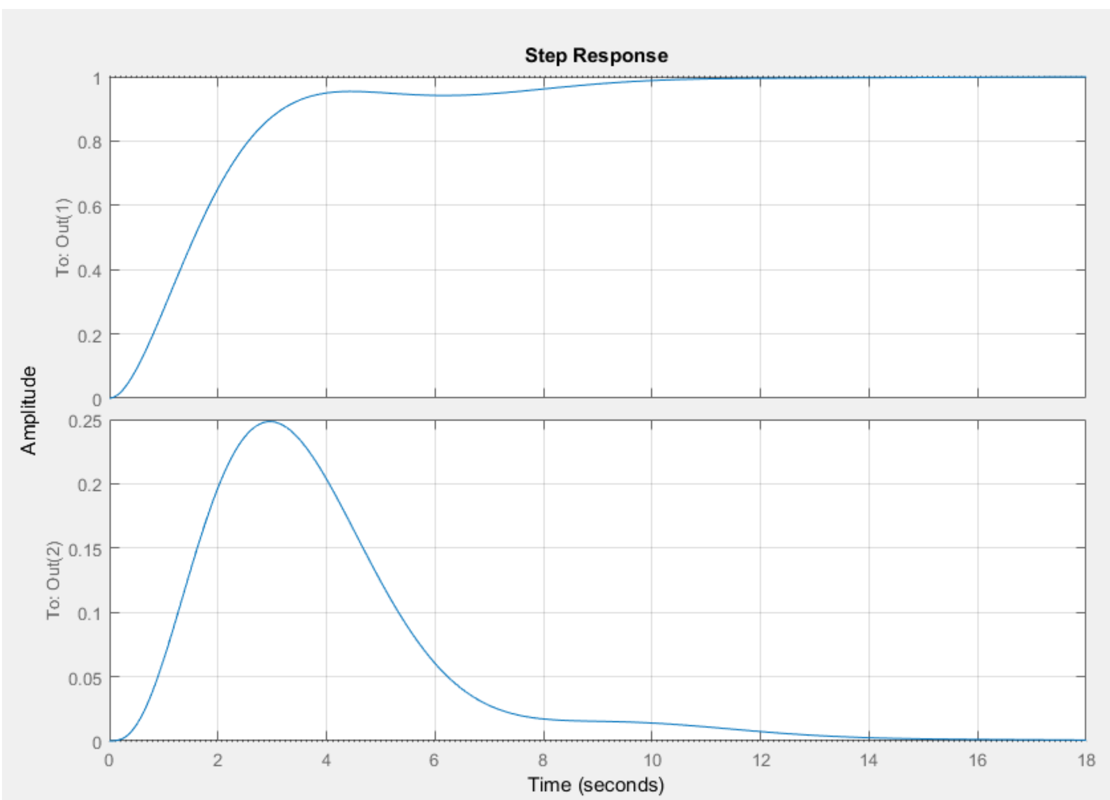
```
>> %Función de Transferencia de X1(s)
>> g1 = tf([1 2 1],[1 4 5 4 1])

g1 =

          s^2 + 2 s + 1
  -----------------------------
  s^4 + 4 s^3 + 5 s^2 + 4 s + 1

Continuous-time transfer function.

>> %Función de Transferencia de X2(s)
>> g2 = tf([1 0],[1 4 5 4 1])

g2 =

                s
  -----------------------------
  s^4 + 4 s^3 + 5 s^2 + 4 s + 1

Continuous-time transfer function.
```

Una vez asignadas las funciones de transferencia $G_1(s)\ y\ G_2(s)$ a las variables $g1\ y\ g2$, se crea una matriz G de 2x1 en la cual se colocarán los elementos correspondientes de $g1\ y\ g2$. Esta matriz nos ayudara para poder llevar a cabo la simulación.

```
>> %Creación de la Matriz G
>> G = [g1;g2]

G =

  From input to output...
                 s^2 + 2 s + 1
   1:  -----------------------------
       s^4 + 4 s^3 + 5 s^2 + 4 s + 1

                       s
   2:  -----------------------------
       s^4 + 4 s^3 + 5 s^2 + 4 s + 1

Continuous-time transfer function.
```

Para simular la salida del sistema a una entrada escalón unitario, se usa el comando *step ()*, del cual se obtiene la gráfica siguiente:

```
>> %Salida del sistema a una entrada escalón unitario
>> step (G)
>> %Out(1) es la salida correspondiente a X1(s)
>> %Out(2) es la salida correspondiente a X2(s)
```



Se puede observar que ambas gráficas corresponden a las obtenidas en los pasos anteriores para cada salida $x_1(s)$ y $x_2(s)$.

4.3 Representación de la salida del sistema usando las funciones de transferencia en forma de polos y ceros del sistema.

Aprovechando las variables creadas en el paso 4.2, las cuales corresponden a las funciones de transferencia $G_1(s)$ y $G_2(s)$ pero en forma polinomial, solo se usan los comandos *zpk ()* para obtener las funciones $G_1(s)$ y $G_2(s)$ en su forma polos y ceros. Ahora las variables que representan a las funciones de transferencia son llamadas $g1zp$ y $g2zp$.

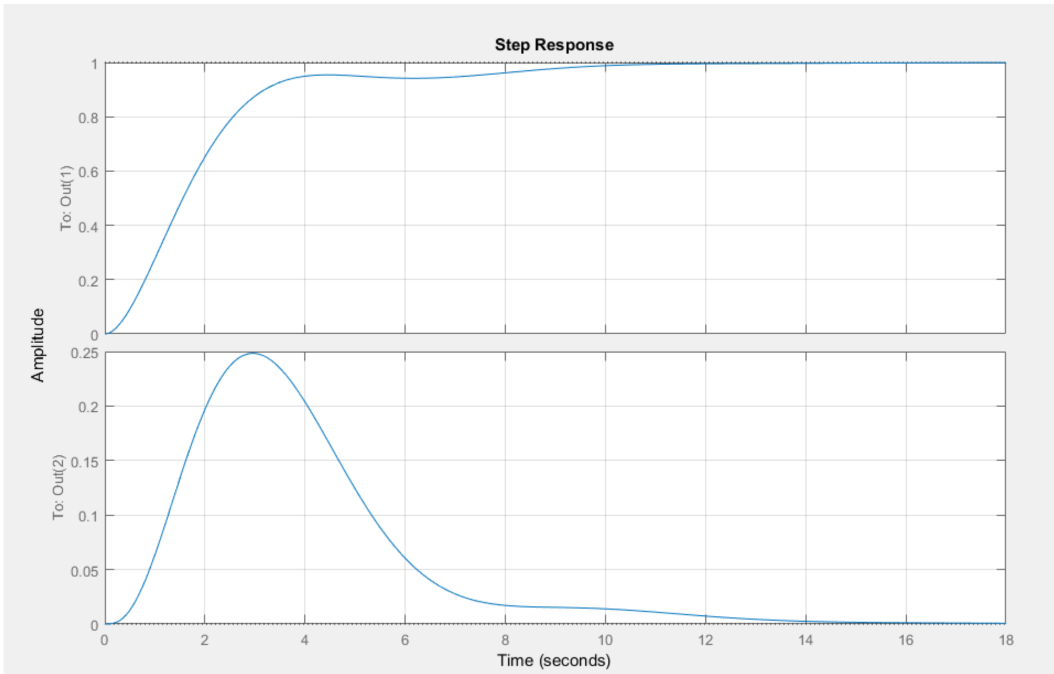
```
>> %Función de Transferencia de X1(s)
>> g1zp = zpk(g1)

g1zp =

               (s+1)^2
  --------------------------------
  (s+2.618) (s+0.382) (s^2 + s + 1)

Continuous-time zero/pole/gain model.

>> %Función de Transferencia de X2(s)
>> g2zp = zpk(g2)

g2zp =

                  s
  --------------------------------
  (s+2.618) (s+0.382) (s^2 + s + 1)

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

De igual manera que en el paso 4.2 se crea una matriz GZP en la cual se colocaran los elementos de las funciones de transferencia $g1zp$ y $g2zp$.

```
>> %Creación de la Matriz GZP
>> GZP = [g1zp;g2zp]

GZP =

  From input to output...
                    (s+1)^2
   1:  --------------------------------
       (s+2.618) (s+0.382) (s^2 + s + 1)

                       s
   2:  --------------------------------
       (s+2.618) (s+0.382) (s^2 + s + 1)

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

Para simular la salida del sistema a una entrada escalón unitario, se usa el comando *step ()*, del cual se obtiene la gráfica siguiente:

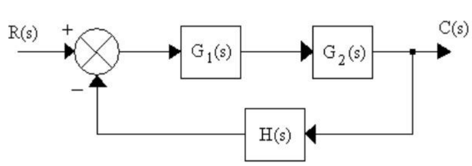
```
>> %Salida del sistema a una entrada escalón unitario
>> step(GZP)
>> %Out(1) es la salida correspondiente a X1(s)
>> %Out(2) es la salida correspondiente a X2(s)
```



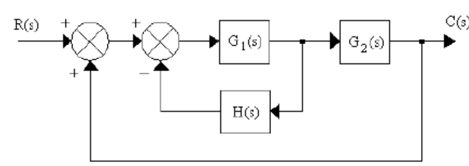
Esta última gráfica es idéntica a las obtenidas con anterioridad.

5. Simplifique analíticamente los siguientes sistemas considerando $G_1(s)\frac{4}{2s+1}$, $G_2(s)\frac{3s}{s^2+2s+1}$ y $H(s)\frac{2s+3}{s+4}$

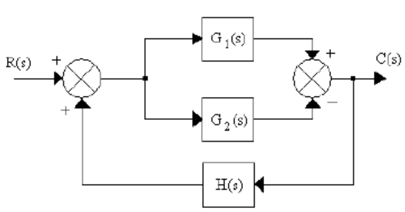
a)



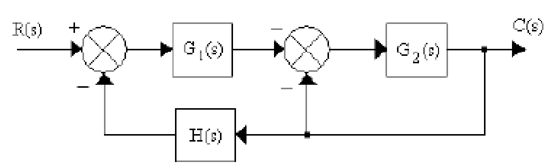
c)



b)



d)



La solución analítica de cada uno de los sistemas se presenta en el ANEXO 1.

6. Simplifique en MatLab los sistemas del paso 5.

Se introducen cada una de las funciones de transferencia en MatLab:

```
>> g1=tf([0 4],[2 1])

g1 =

     4
  -------
  2 s + 1

Continuous-time transfer function.

>> g2=tf([3 0],[1 2 1])

g2 =

       3 s
  -------------
  s^2 + 2 s + 1

Continuous-time transfer function.

>> h=tf([2 3],[1 4])

h =

  2 s + 3
  -------
   s + 4

Continuous-time transfer function.
```

- Para el inciso a:

  Al analizar el sistema se observan los bloques g1 y g2 en serie, posteriormente la retroalimentación entre el bloque resultante y el bloque h. A continuación, se muestran los pasos para cada una de las acciones que se realizaron, al final se obtiene la función de transferencia del sistema:

```
>> %Calculo en serie de g1 y g2 (g1*g2)
>> s=series(g1,g2)

s =

            12 s
  -----------------------
  2 s^3 + 5 s^2 + 4 s + 1

Continuous-time transfer function.

>> %Retroalimentacion de s y h
>> feedback(s,h)

ans =

             12 s^2 + 48 s
  ----------------------------------
  2 s^4 + 13 s^3 + 48 s^2 + 53 s + 4

Continuous-time transfer function.
```

- Para el inciso b:

Se observan los bloques g1 y g2 en paralelo y se realizan los cálculos, posteriormente se tiene la retroalimentación entre el bloque resultante y el bloque h, a continuación, se muestran los pasos que se realizaron para obtener la función de transferencia del sistema:

```
>> %Paralelo entre los bloques g1 y g2
>> p1=parallel(g1,-g2)

p1 =

     -2 s^2 + 5 s + 4
  -----------------------
  2 s^3 + 5 s^2 + 4 s + 1

Continuous-time transfer function.

>> %Retroalimentacion entre p1 y h
>> r2=feedback(p1,-h)

r2 =

     -2 s^3 - 3 s^2 + 24 s + 16
  ---------------------------------
  2 s^4 + 17 s^3 + 20 s^2 - 6 s - 8

Continuous-time transfer function.
```

- Para el inciso c:

Se calcula la retroalimentación entre le bloque g1 y h, el bloque resultante se encuentra en serie con el bloque g2 y al final, el ultimo bloque resultante realiza una retroalimentación con la unidad. A continuación, se muestra los pasos para la simplificación del sistema y la obtención de su función de transferencia

```
>> %Retroalimentacion entre el bloque g1 y h
>> r1c=feedback(g1,h)

r1c =

      4 s + 16
  ----------------
  2 s^2 + 17 s + 16

Continuous-time transfer function.

>> %Serie entre el r1c y g2
>> s1c=series(r1c,g2)

s1c =

             12 s^2 + 48 s
  -----------------------------------
  2 s^4 + 21 s^3 + 52 s^2 + 49 s + 16

Continuous-time transfer function.

>> %Retroalimentacion entre s1c y 1
>> r2c=feedback(s1c,-1)

r2c =


           12 s^2 + 48 s
  -------------------------------
  2 s^4 + 21 s^3 + 40 s^2 + s + 16

Continuous-time transfer function.
```

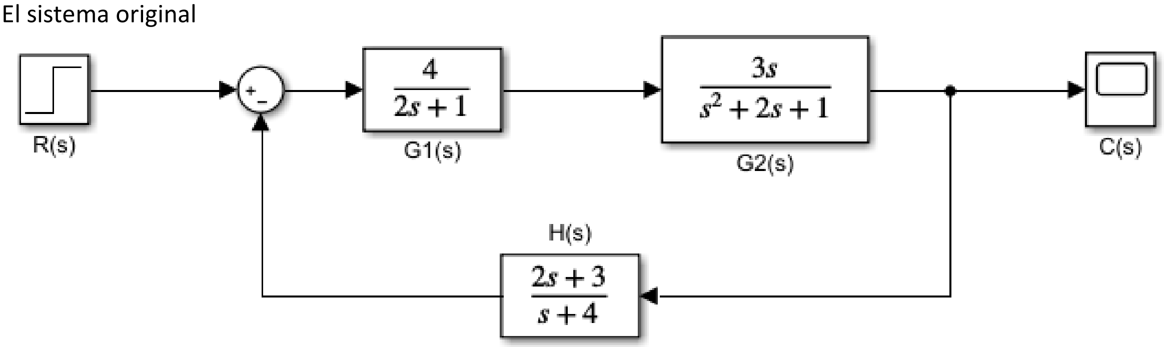
- Para el inciso d:

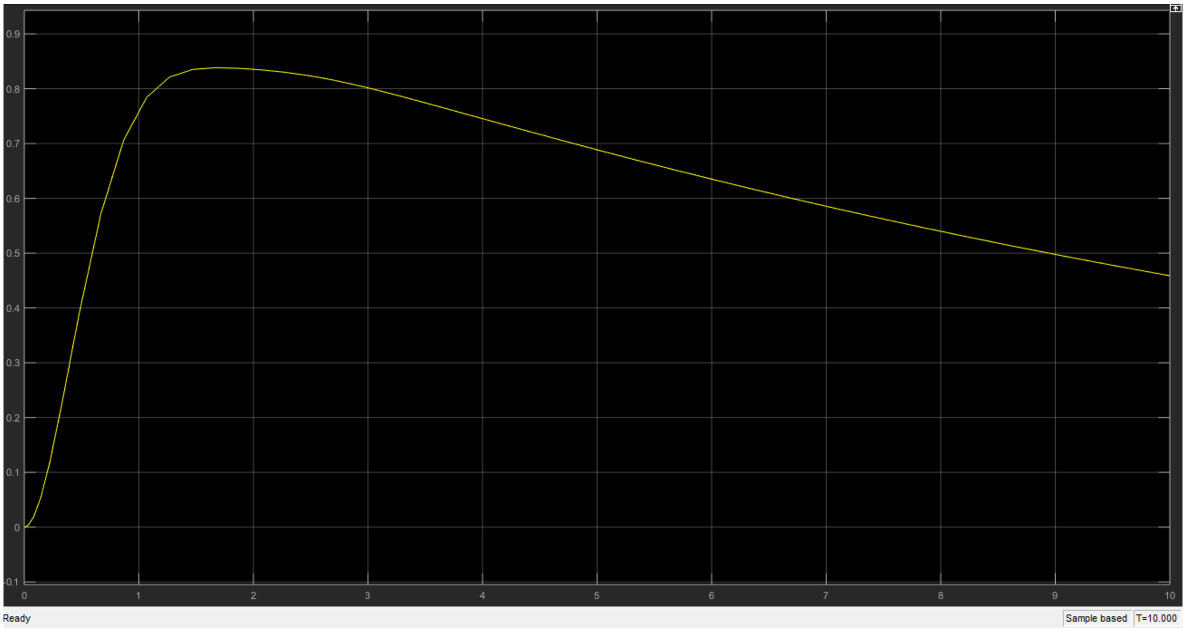
Es necesario realizar la retroalimentación del bloque g2 con la unidad, esto nos deja el bloque resultante en serie con el bloque g1 y posteriormente se realiza la retroalimentación con el ultimo bloque resultante con el bloque H. Se presentan todos los pasos para la obtención de la función de transferencia a continuación:

```
>> %Retroalimentacion entre el bloque de g2 y 1
>> r1d=feedback(g2,1)

r1d =

        3 s
  -------------
  s^2 + 5 s + 1

Continuous-time transfer function.

>> %Serie entre g1 y r1d
>> s1d=series(-g1,r1d)

s1d =

           -12 s
  ------------------------
  2 s^3 + 11 s^2 + 7 s + 1

Continuous-time transfer function.

>> %Retroalimentacion entre s1d y h
>> r2d=feedback(s1d,h)

r2d =


r2d =

           -12 s^2 - 48 s
  ---------------------------------
  2 s^4 + 19 s^3 + 27 s^2 - 7 s + 4

Continuous-time transfer function.
```

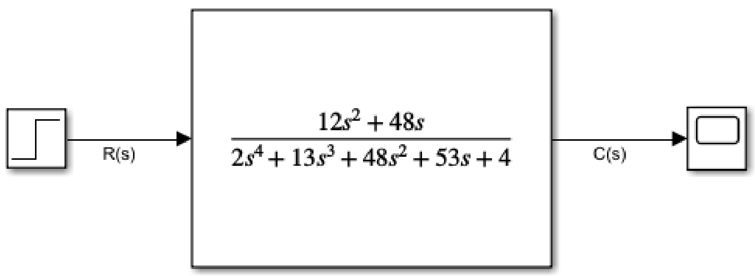
7. Simule en Simulink los sistemas originales del paso 5, y los sistemas simplificados de los pasos 5 y 6 considerando una entrada escalón unitario.

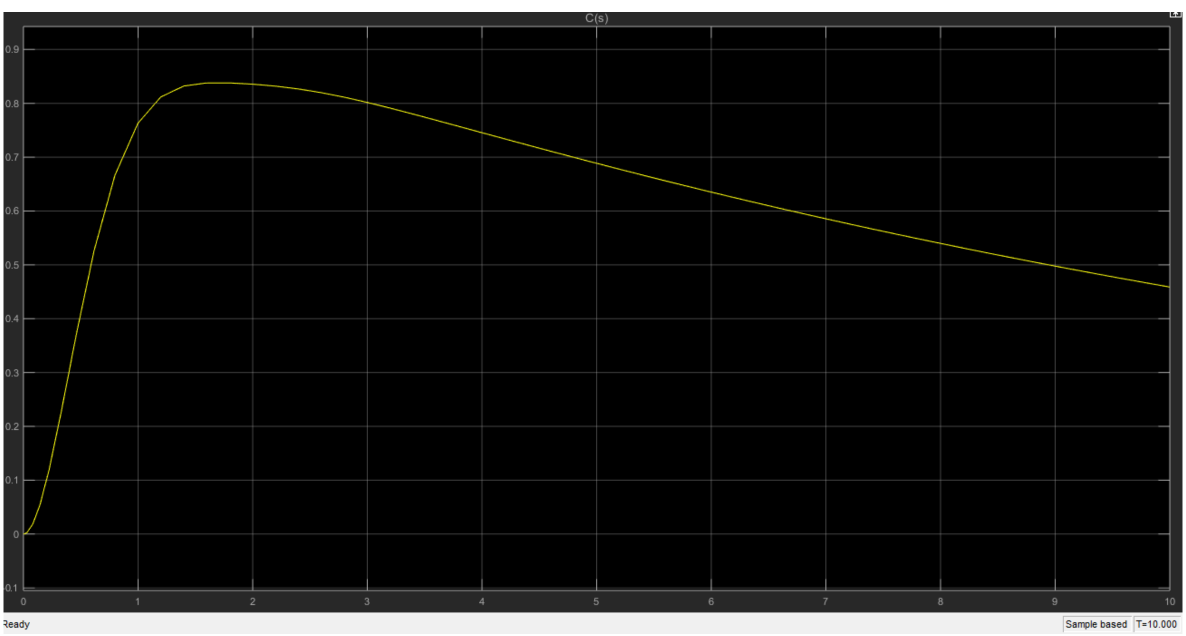
- Para el sistema a)
  El sistema original





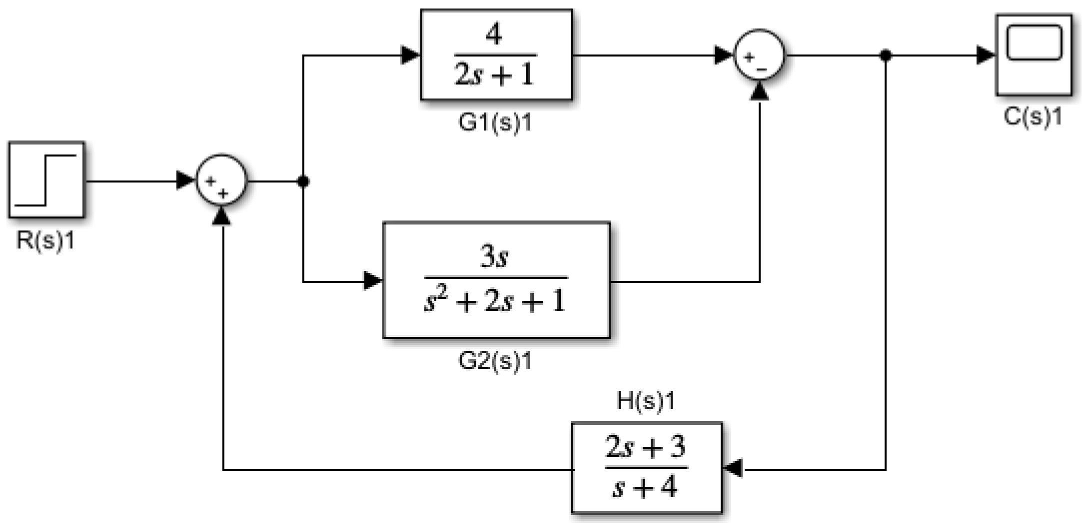
La función de transferencia del sistema, ya simplificado

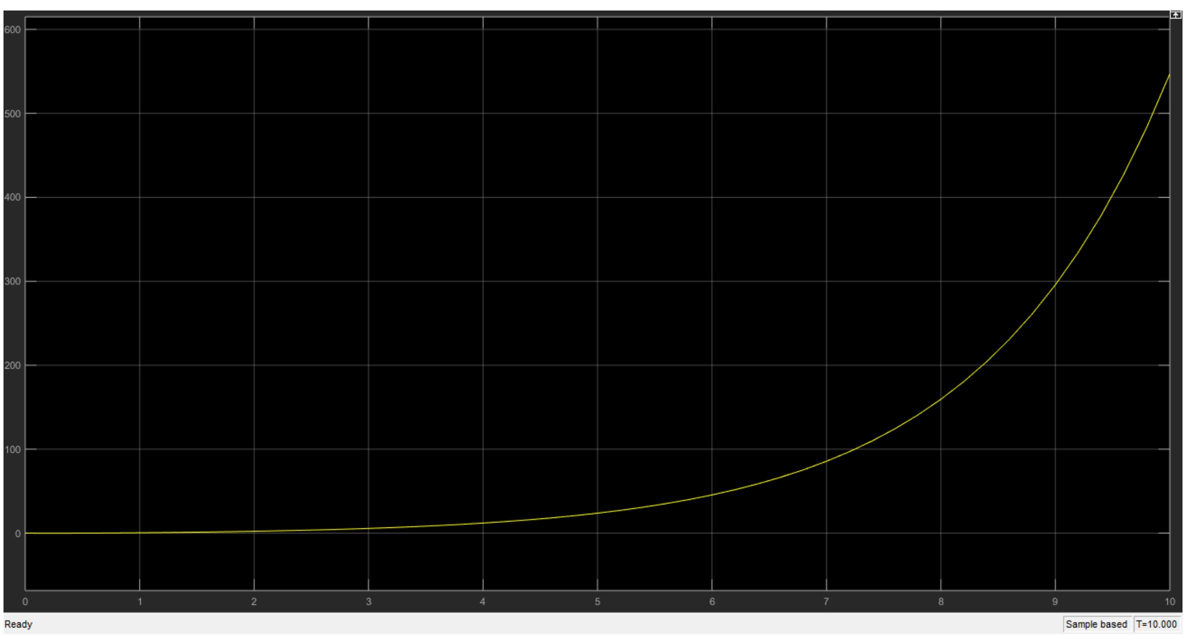




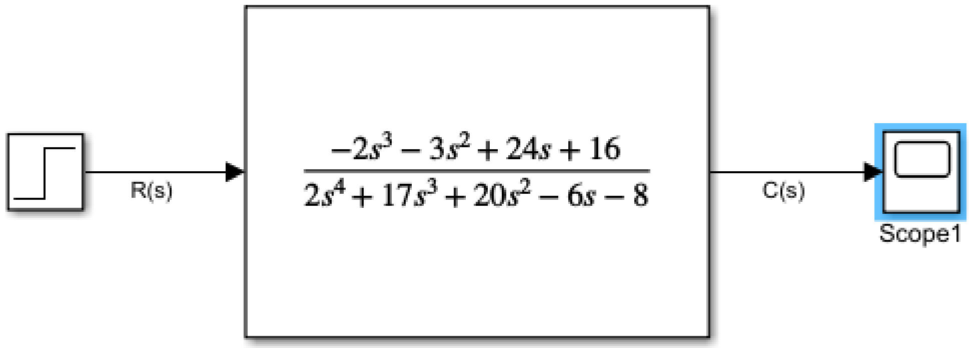
Se puede concluir que la señal que se obtiene por ambos métodos es la misma, se pueden tener dos caminos para obtener el resultado.

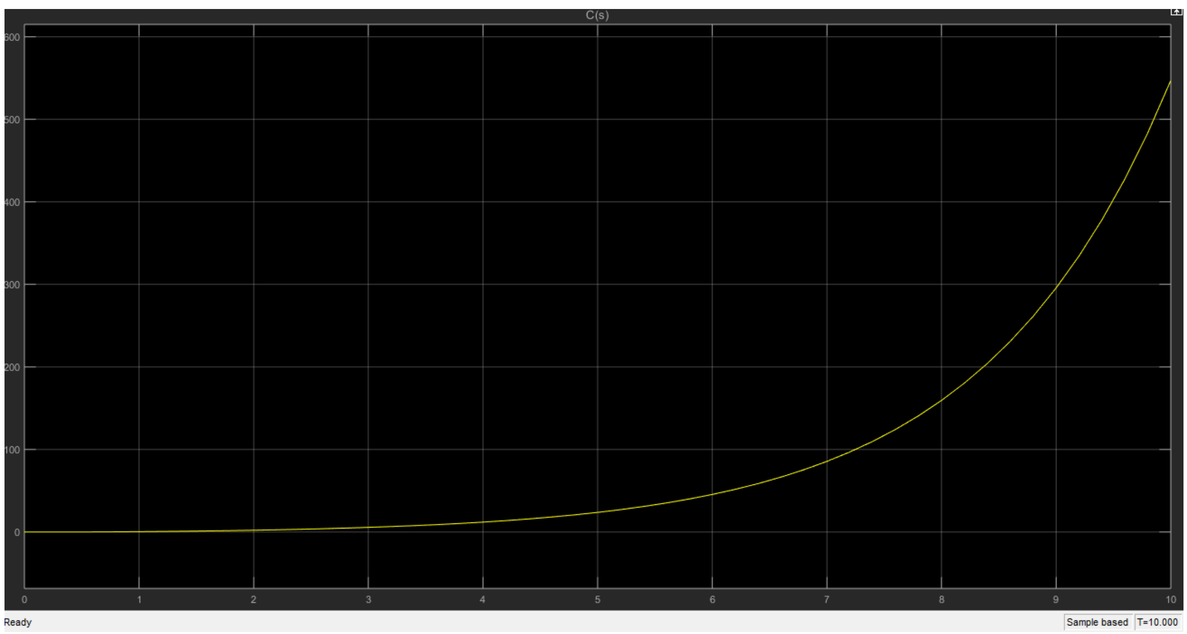
- Para el sistema b)
  El sistema original





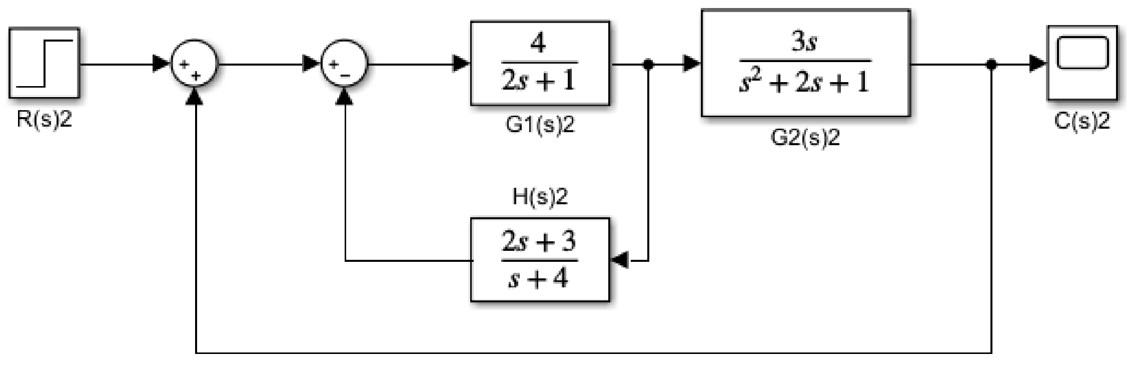
La función de transferencia del sistema, ya simplificado

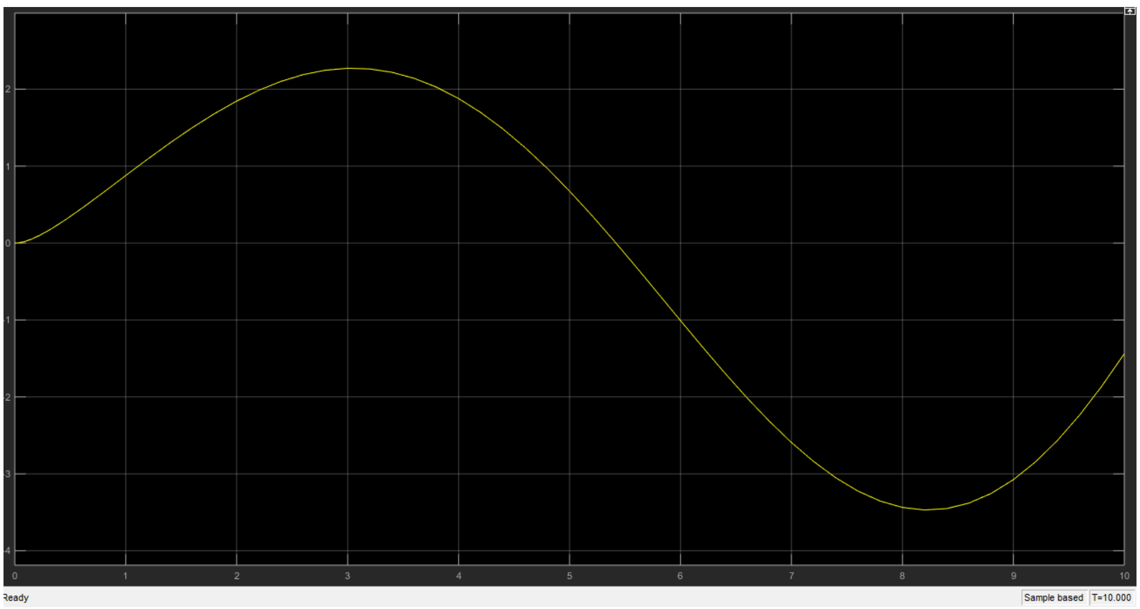




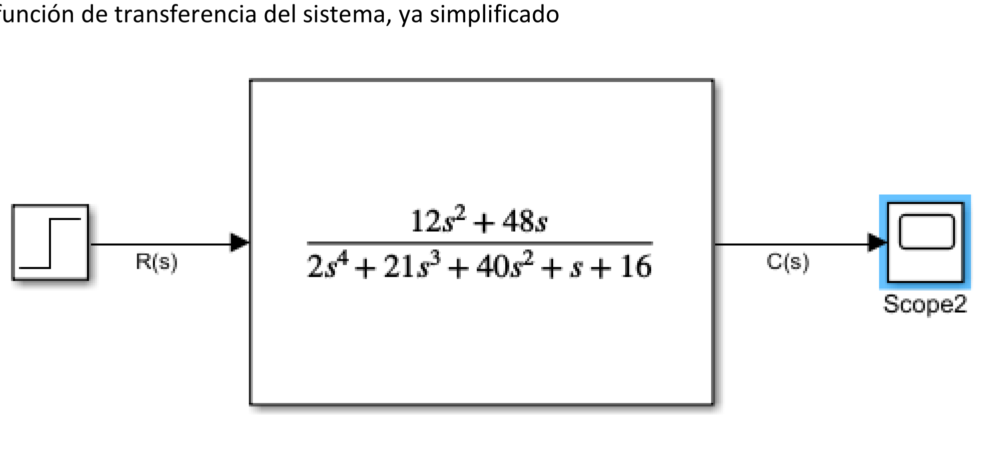
Es muy claro que, en este caso, también se contiene la misma señal controlada, ya sea trabajando a través de la simplificación del sistema o directamente sobre él.

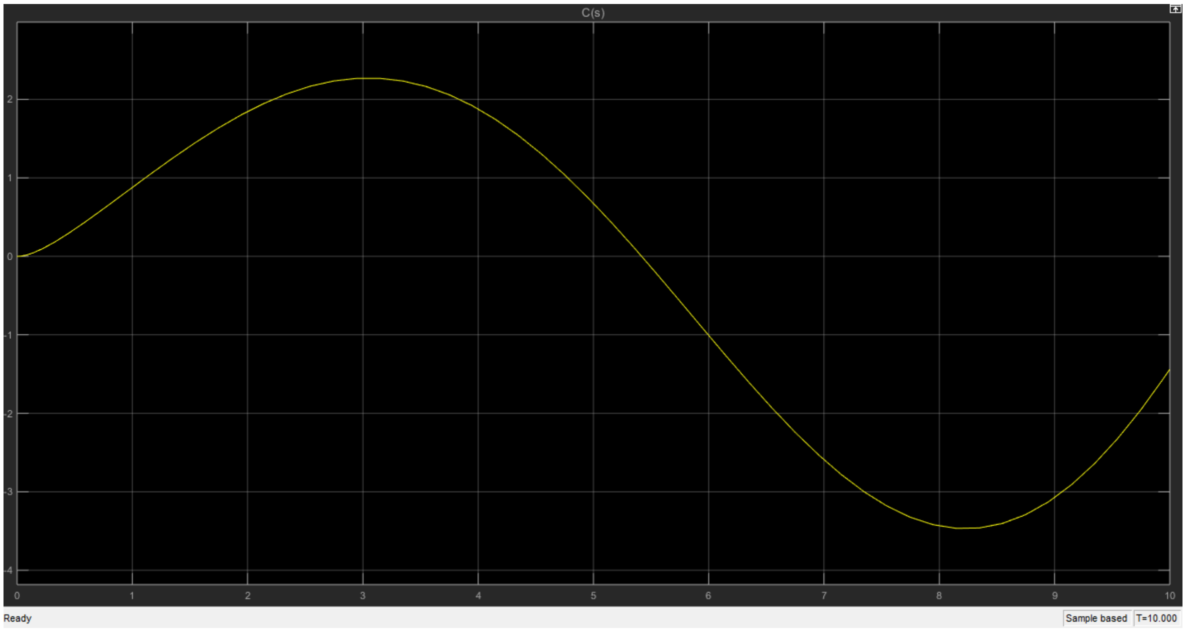
- Para el sistema c)
  El sistema original





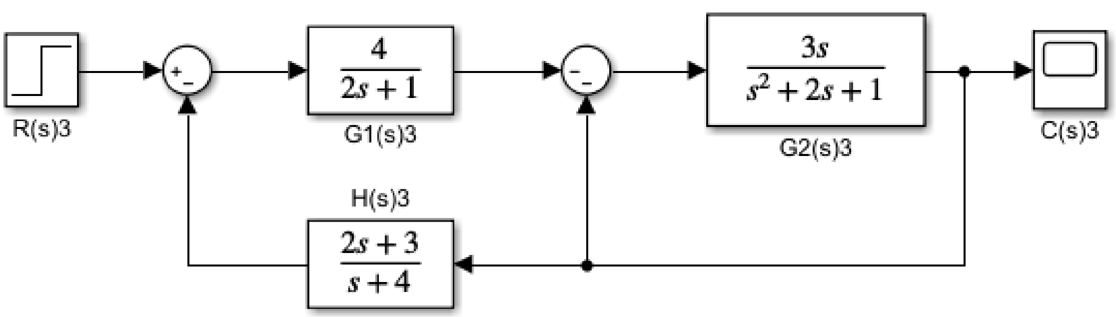
La función de transferencia del sistema, ya simplificado

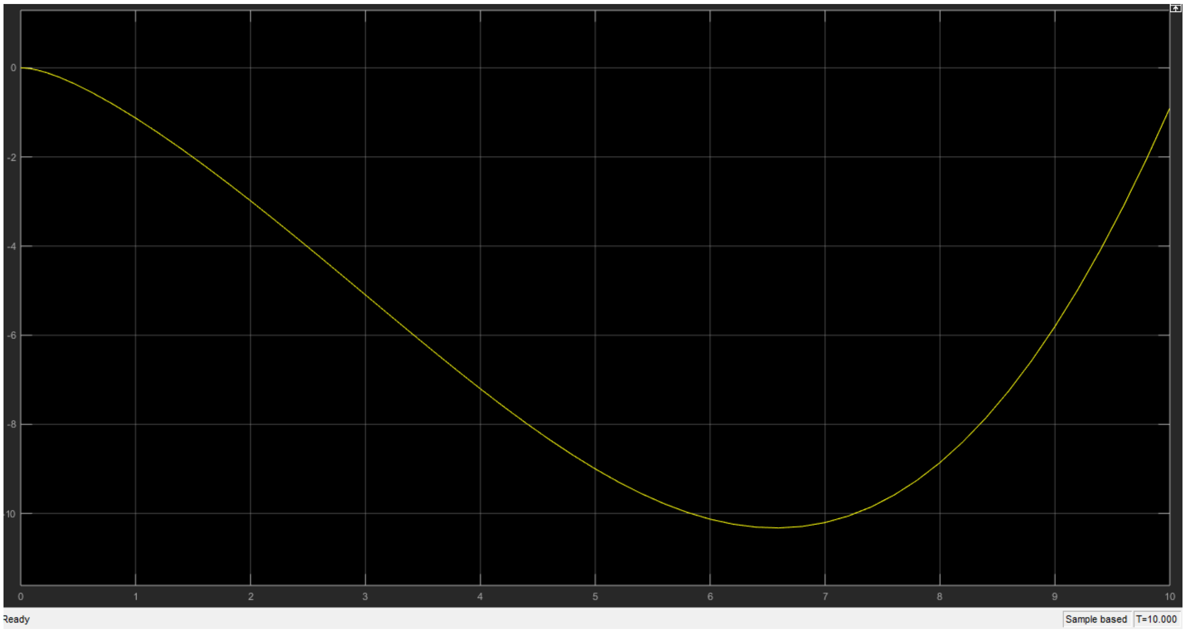




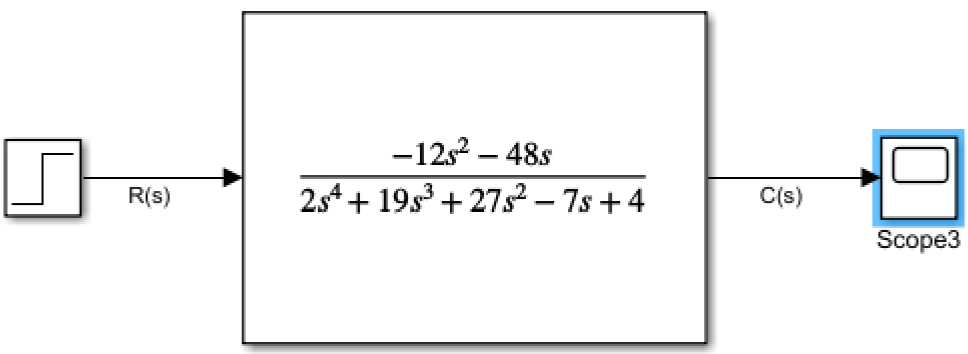
Se obtiene de igual manera la misma señal controlada por medio de los dos métodos presentados con anterioridad.

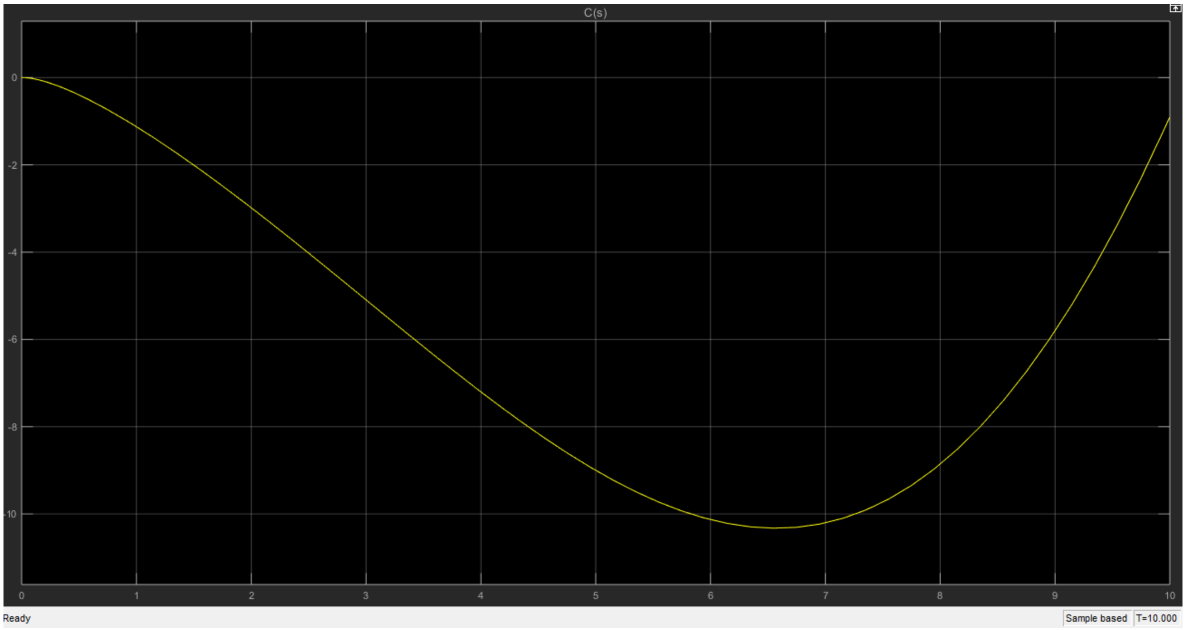
- Para el sistema d)
  El sistema original





La función de transferencia del sistema, ya simplificado





Por ultimo, en el cuarto sistema se puede concluir que se obtiene exactamente la misma señal controlada por cualquiera de los dos caminos abordados en esta práctica.

3|CONCLUSIONES

González Rodríguez Ángel: La práctica permitió adquirir un conocimiento mas profundo sobre lo que ofrece MatLab y Simulink además de las categorías que componen el toolbox "control". La seguridad que nos brinda cada uno de los programas, además la interfaz amigable y comprensible que nos ofrece Simulink para brindarnos una mejor experiencia y una seguridad en lo que refiere a sus cálculos. La práctica me permitió corregir algunos errores básicos que aún seguía arrastrando, pero el hecho de realizar la comparación y la muestra de la salida en cada uno de los casos me permite comparar y determinar que la salida es la misma de dos o mas maneras posibles, y que en la vida laboral quizá sea una de las interrogantes que nos haremos, cuantos caminos o formas hay para lograr una tarea específica, y aun mas preciso cual de todas estas es aún más rápida y confiable.

Reyes Sánchez Luis Ángel: Los puntos más importantes aprendidos en la elaboración de esta presente práctica desde mi punto de visto son: en primera parte, los softwares MatLab y Simulink son una herramienta completamente confiable para el análisis y la simplificación de los diagramas de bloques y representaciones matemáticas (ecuaciones diferenciales y funciones de transferencia) de distintos sistemas, esta es una conclusión formada a través de la comparación de distintas metodologías aplicadas a estos softwares, todas las gráficas visualizadas y obtenidas por todos los métodos aplicados en los softwares fueron prácticamente idénticas, lo que es un gran indicio para la confiabilidad del uso de estas herramientas. Por otra parte, los conceptos adquiridos sobre la teoría de control aplicada para la solución de esta práctica, así como lo aprendido en las sesiones de MatLab impartidas por el docente, fueron aplicadas en conjunto para poder entender aún más la aplicación de las representaciones matemáticas y graficas de los sistemas.

Suárez López Rodrigo: Al momento de realizar el diagrama de bloques por componentes del sistema, nos ayudó el diagrama de fuerzas de cada una de las masas, utilizando la segunda ley de newton ($\sum F = ma$), para obtener dos ecuaciones que nos indican las entradas y salidas del sistema, las otras ecuaciones se calcularon de acuerdo a cada componente del sistema, éstas ecuaciones obtenidas nos ayudaron a realizar el diagrama de bloques de sistema. Realizando la simulación de las representaciones que se obtuvieron en Simulink y en MatLab, a una entrada escalón unitario, se observa que obtenemos la misma gráfica en MatLab como en Simulink, por tanto, podemos asegurar que el resultado que se obtuvo es el correcto. De a cuerdo a las gráficas obtenidas, se observa que la salida x1(t) con el transcurso del tiempo tiende a tener el valor máximo de entrada y la salida x2(t) con el transcurso del tiempo su valor tiende a cero.

4)OBSERVACIONES

1. Al realizar las simplificaciones del diagrama de bloques analíticamente, deben utilizarse las reglas del álgebra de diagramas de bloques para al momento de sustituir las funciones G1(s), G2(s) y H(s), la obtención de la F.T sea correcta.
2. Los comandos de MatLab proporcionados en las clases de control, son una herramienta que nos ayuda a comprender de una mejor manera el funcionamiento de los sistemas y así mismo facilita la obtención de los resultados requeridos.

5|REFERENCIAS

[1] *Ingeniería de control moderna*. Ogata, K. Prentice Hall. 5ª Edición, 2010.

[2] *Ingeniería de control moderna*. Ogata, K. Prentice Hall. 4ª Edición, 2002.

[3] *Ingeniería de control moderna*. Ogata, K. Pearson Education. 3ª Edición, 1998.

[4] *Problemas de ingeniería de control utilizando MatLab*. Ogata, K. Prentice Hall. 1999.

[5] *Sistemas de control automático*. Kuo, B.C. Prentice Hall. 7ª Edición, 1996

[6] *Ingeniería de control.* Bolton, W. Alfaomega. 2ª Edición, 2001