Práctica 1: Señales y Sistemas en MATLAB		Grupo	3
		Puesto	-
Apellidos, nombre	Álvarez Ocete, José Antonio		
Apellidos, nombre	Sáez Maldonado , Francisco Javier		

El objetivo de esta práctica es presentar al alumno el modo de orientar las herramientas que ofrece MATLAB a la representación y manejo de señales y sistemas, así como verificar algunas de las propiedades de los sistemas lineales e invariantes con el tiempo.

1.1 *Introducción a Matlab*

La ayuda de Matlab es de grandísima utilidad ya que cuenta con una documentación muy detalla y con muchos ejemplos de uso. Se encuentra disponible en la parte superior derecha pulsando en el icono con el signo de interrogación.

Antes de comenzar se recomienda echar un vistazo a los siguientes apartados de la ayuda:

1.2 *Generación y manipulación básica de señales*

Para seguir este apartado escriba en la línea de comando todos los ejemplos mostrados. Utilice la ayuda de MATLAB para documentarse sobre cualquier comando que desconozca.

1.2.1 Ejercicio 1: representación de una señal en un rango dado

En general, una señal quedará representada por un vector fila o por un vector columna (es decir, por matrices con una única fila o columna). En MATLAB, todos los vectores se indexan comenzando por el `1`,

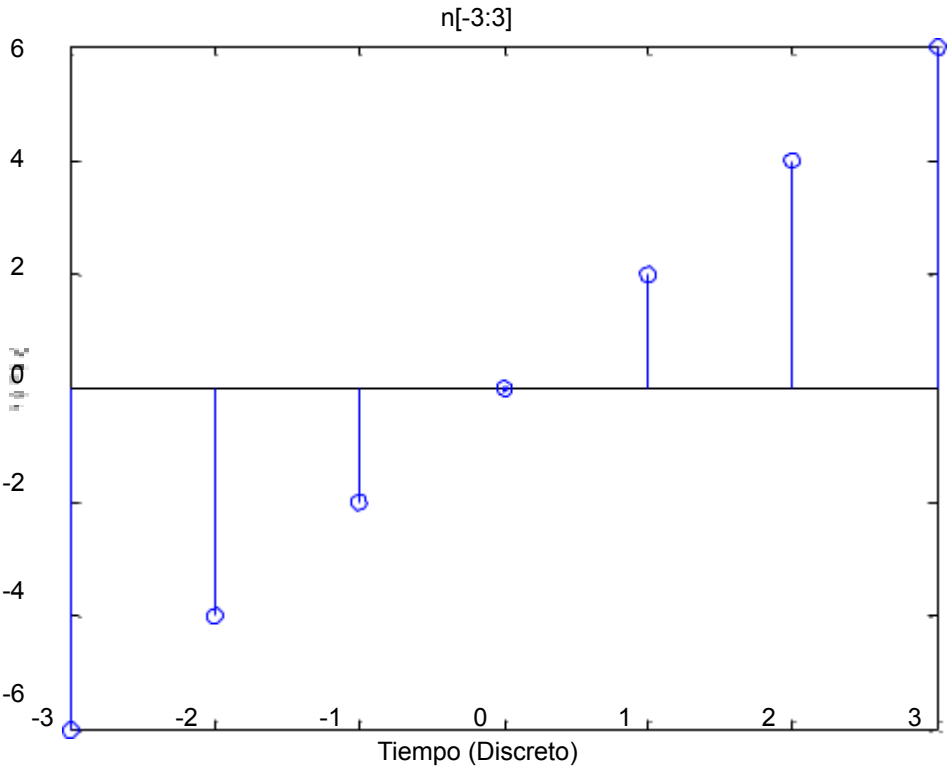
es decir, `y(1)` es el primer elemento del vector `y`. Cuando este criterio no coincida con el del problema a resolver (e.g., porque el primer valor del vector y corresponda al índice `-3`), se puede crear un vector adicional de índices. Por ejemplo, para representar la señal:

$$x[n] = \begin{cases} 2n, & -3 \le 0 \le 3 \\ 0, & resto \end{cases}$$

, puede usarse el operador '`:`' para definir un vector con los índices de `x[n]` no nulos, y luego definir el propio vector `x` de modo que contenga los valores deseados en cada uno de estos índices:

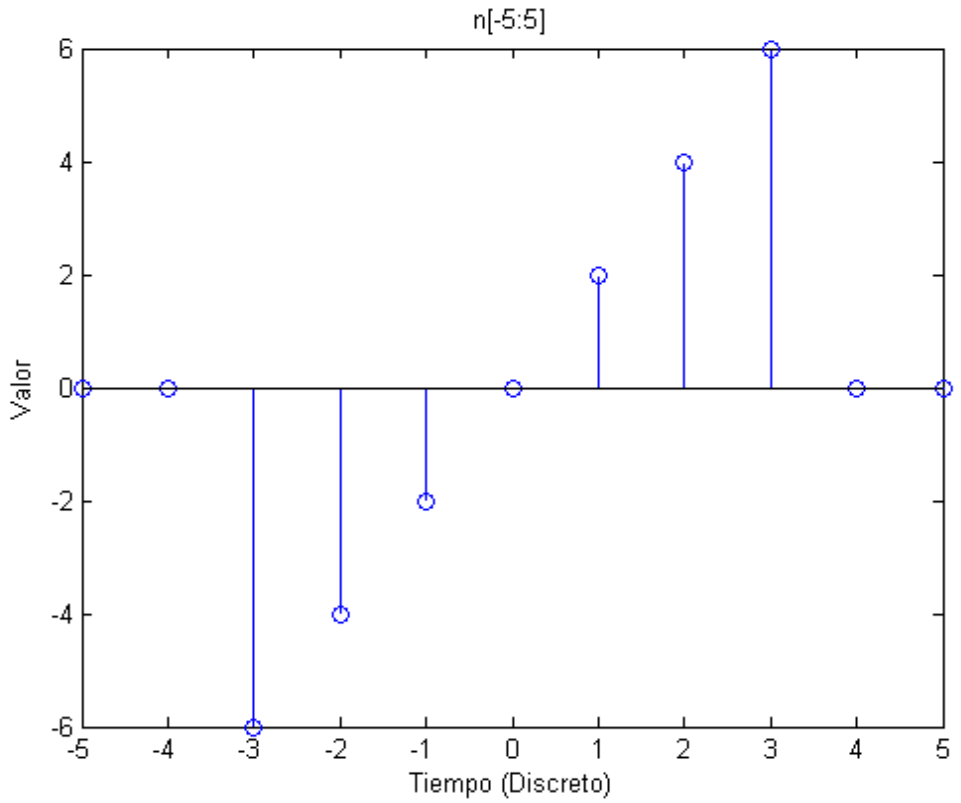
```
>> n=[-3:3];
>> x=2*n;
```

Represente esta señal escribiendo `stem(n,x)`. Para examinar la señal en un rango más amplio de índices, será necesario extender tanto el vector de índices, `n`, como la señal `x`:



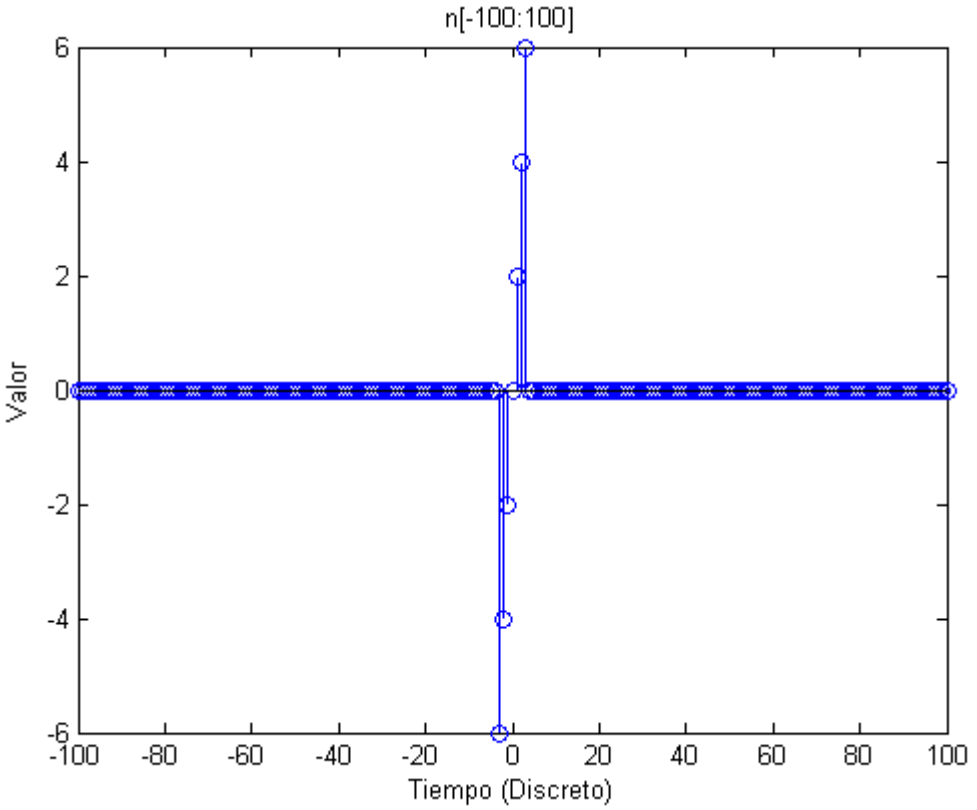
— Para representar la señal en el intervalo `[-5,5]`:

```
>> n=[-5:5];
>> x=[0 0 x 0 0];  % x tenía el valor del ejemplo anterior
```



— Para representarla en `[-100,100]`:

```
>> n=[-100:100];
>> x=[zeros(1,95) x zeros(1,95)];  % x tenía el valor del ejemplo anterior
```



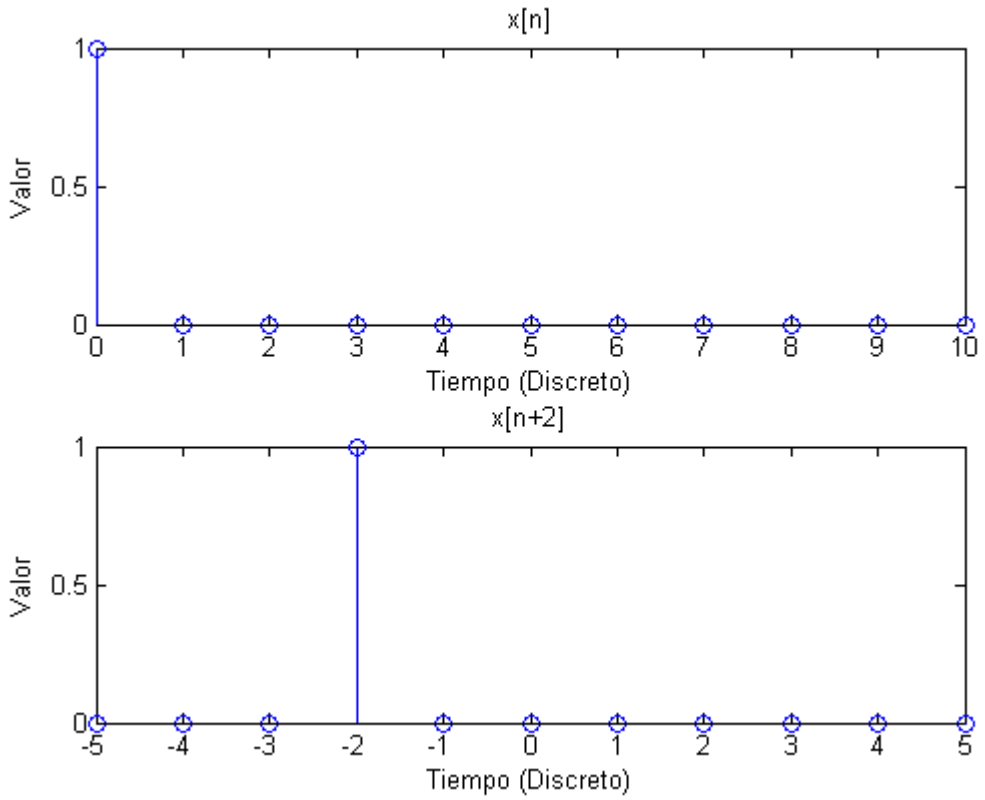
Represente cada una de estas tres señales en tres figuras distintas (vea el comando **`figure`**).

1.2.2 Ejercicio 2: representación de dos señales en un cierto rango

Sean $x_1[n] = \delta[n]$ y $x_2[n] = \delta[n+2]$ (la función $\delta[n]$ toma valor 1 para $n=0$ y valor nulo en el resto).

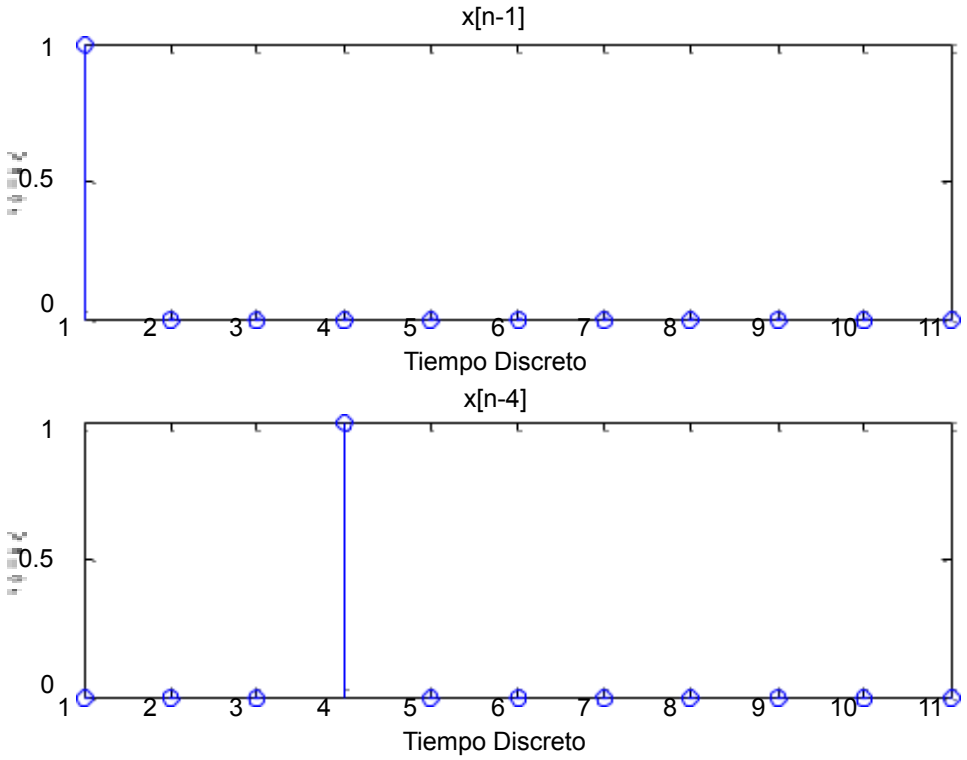
Dibuje aparte el aspecto que tienen ambas señales. Estas señales pueden definirse en MATLAB escribiendo:

```
>> nx1=[0:10];
>> x1=[1 zeros(1,10)];
>> nx2=[-5:5];
>> x2=[zeros(1,3) 1 zeros(1,7)];
```



Para representarlas, basta escribir **`stem(nx1,x1)`** y **`stem(nx2,x2)`**. Compruebe que obtiene el resultado esperado.

Represéntelas ahora directamente con **`stem(x1)`** y **`stem(x2)`**, función que en ausencia de un vector de índices asume que éste comienza en **1** y que tiene la misma longitud que la señal. Indique, en esta situación, cual es la expresión analítica de las señales que observa:



Señal representada con `stem(x1)`	δ[n-1]
Señal representada con `stem(x2)`	δ[n-4]

>> Tenga en cuenta en lo sucesivo que a la hora de representar señales, tan importante como la expresión de la señal es el vector de índices con respecto al cual se representa. <<

1.2.3 Ejercicio 3: representación de señales continuas

Una señal continua es posible representarla mediante vectores que contengan valores de dicha señal en instantes de tiempo muy cercanos entre sí. Así, si se quiere representar una señal continua en el intervalo $-5 \le t \le 5$ mediante la expresión de un valor cada *0.1* segundos, tenemos dos opciones para crear el vector de *índices* (en este caso instantes de tiempo):

```
>> t=[-5:0.1:5];
```

, o bien:

```
>> t=linspace(-5,5,101);
```

Hecho esto, para representar la señal $x(t) = sen(\pi / 4)$ basta con escribir:

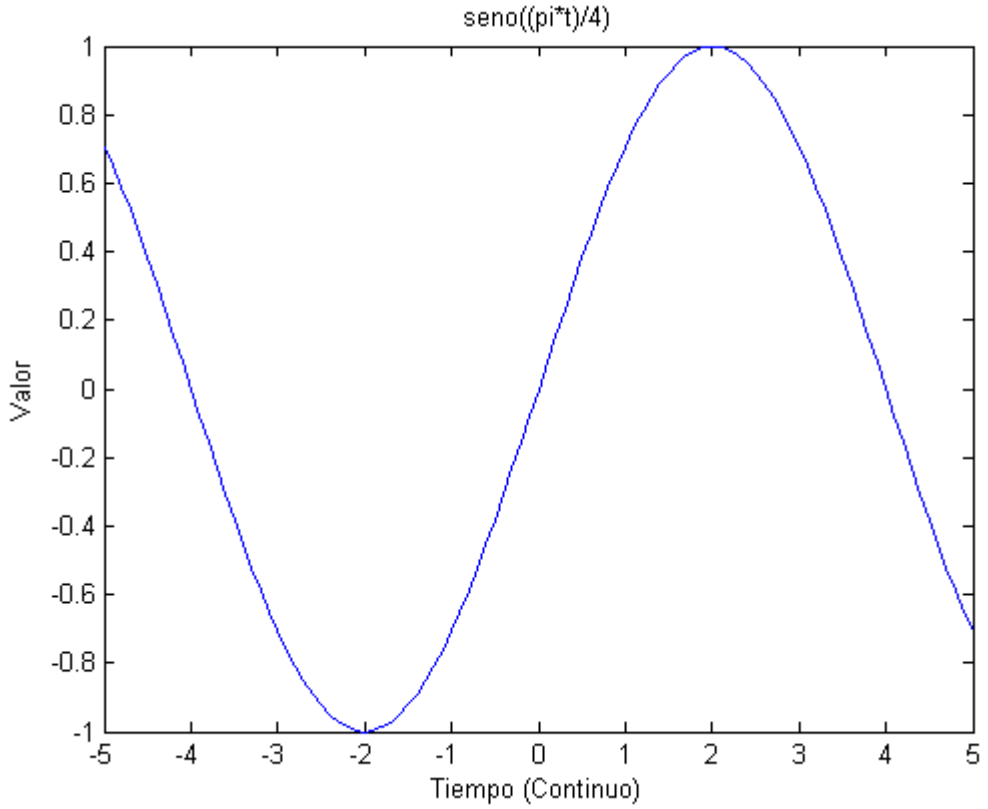
```
>> x=sin(pi*t/4);
```

>> Observe que en MATLAB, cuando el argumento de una función de este tipo (*sin*, *cos*, *exp*, etc.) es un vector, el resultado es un vector del mismo tamaño, en el que cada valor resulta de la aplicación de la función a cada valor del vector argumento. <<

Para representar gráficamente la señal, resaltando su carácter de señal continua,

utilice **`plot`** en vez de **`stem`**:

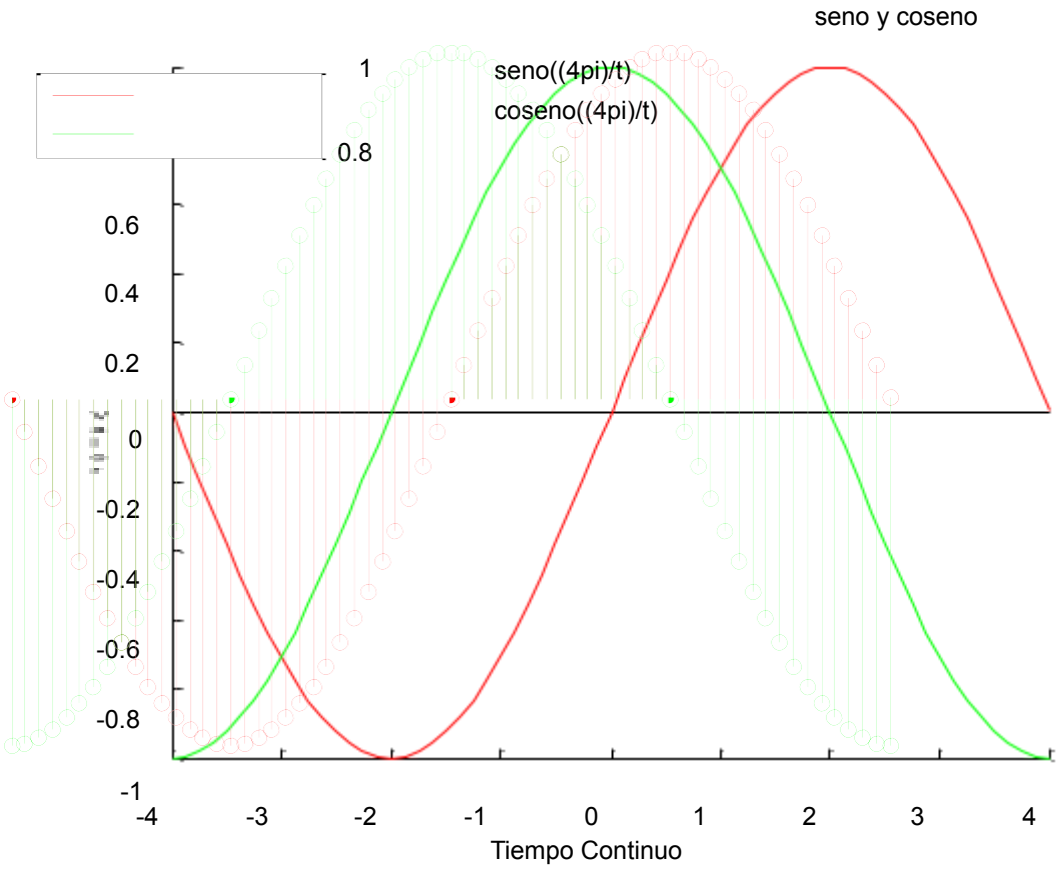
```
>> plot(t,x);
```



Represente gráficamente las señales $x_1(t) = sen(\pi t / 4)$ y $x_2(t) = \cos(\pi t / 4)$ en el intervalo $-4 \le t \le 4$

dando valores cada 1/8 de segundo. Represente ambas sobre la misma figura utilizando el comando **`plot`** y, a continuación, nuevamente sobre la misma figura, represente ambas con el comando **`stem`** (para ello utilice el comando **`hold`**). Utilice dos colores: uno para las dos representaciones de $x_1(t)$ y otro para las dos de $x_2(t)$.

>> En lo sucesivo, siempre que se quiera representar gráficamente una señal de tiempo discreto utilice el comando `stem`; análogamente, siempre que la señal sea de tiempo continuo (aunque con MATLAB se aproxime por una señal de tiempo discreto definida a intervalos regulares y *muy pequeños*) utilice el comando `plot` para resaltar este hecho y evitar cualquier confusión. <<



1.2.4 **Ejercicio 4: representación de señales complejas**

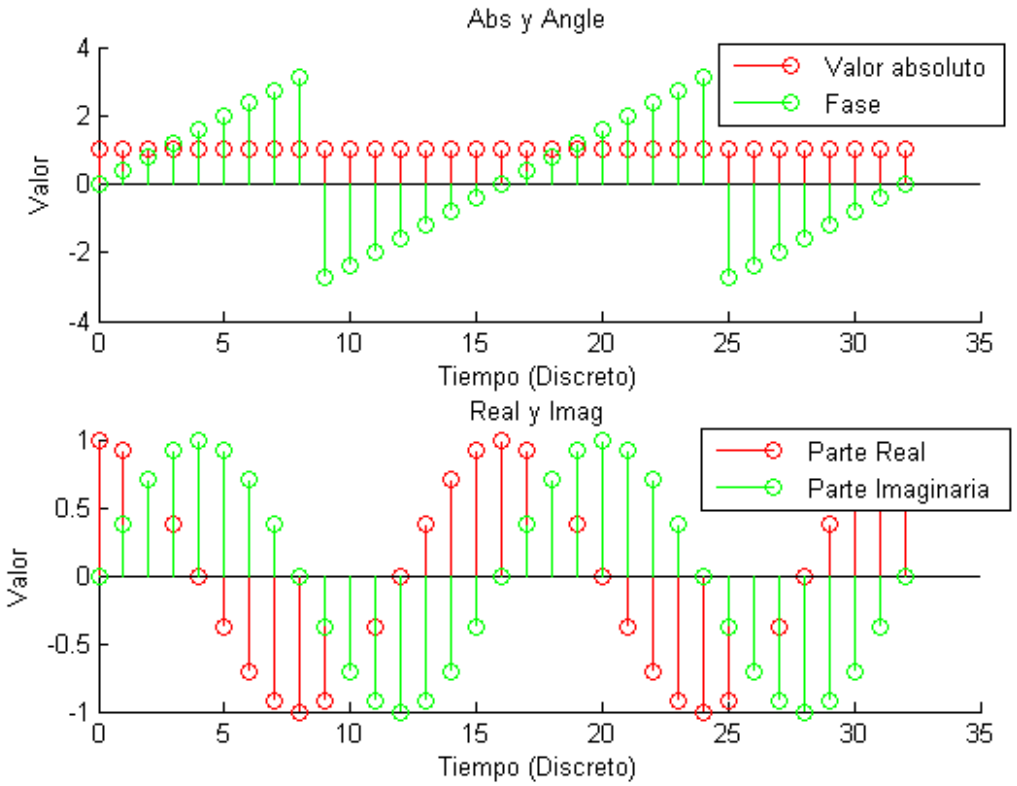
Sea ahora el caso de una exponencial compleja discreta $x[n] = e^{j(\pi/8)n}$ en el intervalo $0 \le n \le 32$:

```
>> n=[0:32];
>> x=exp(j*(pi/8)*n);
```

El vector **x** contiene una serie de *33* valores complejos de la señal $x[n]$. Represéntelos gráficamente,

haciendo uso de la función **stem**, indicando qué característica de cada valor complejo desea representar:

```
>> stem(n,real(x));
>> stem(n,imag(x));
>> stem(n,abs(x));
>> stem(n,angle(x));
```



Compruebe y recuerde que si en la función no se especifica qué característica de la señal compleja se desea representar (es decir, si escribe `stem(n,x)`), MATLAB representará, por defecto, la parte real de la señal y mostrará una advertencia en la línea de comando indicándolo.

1.2.5 Ejercicio 5: operaciones aritméticas con señales

Siempre que dos señales compartan el mismo vector de índices (es decir, que el vector que representa cada señal tenga el mismo origen de tiempos), es posible realizar directamente cierto tipo de operaciones básicas. Así, defina las señales:

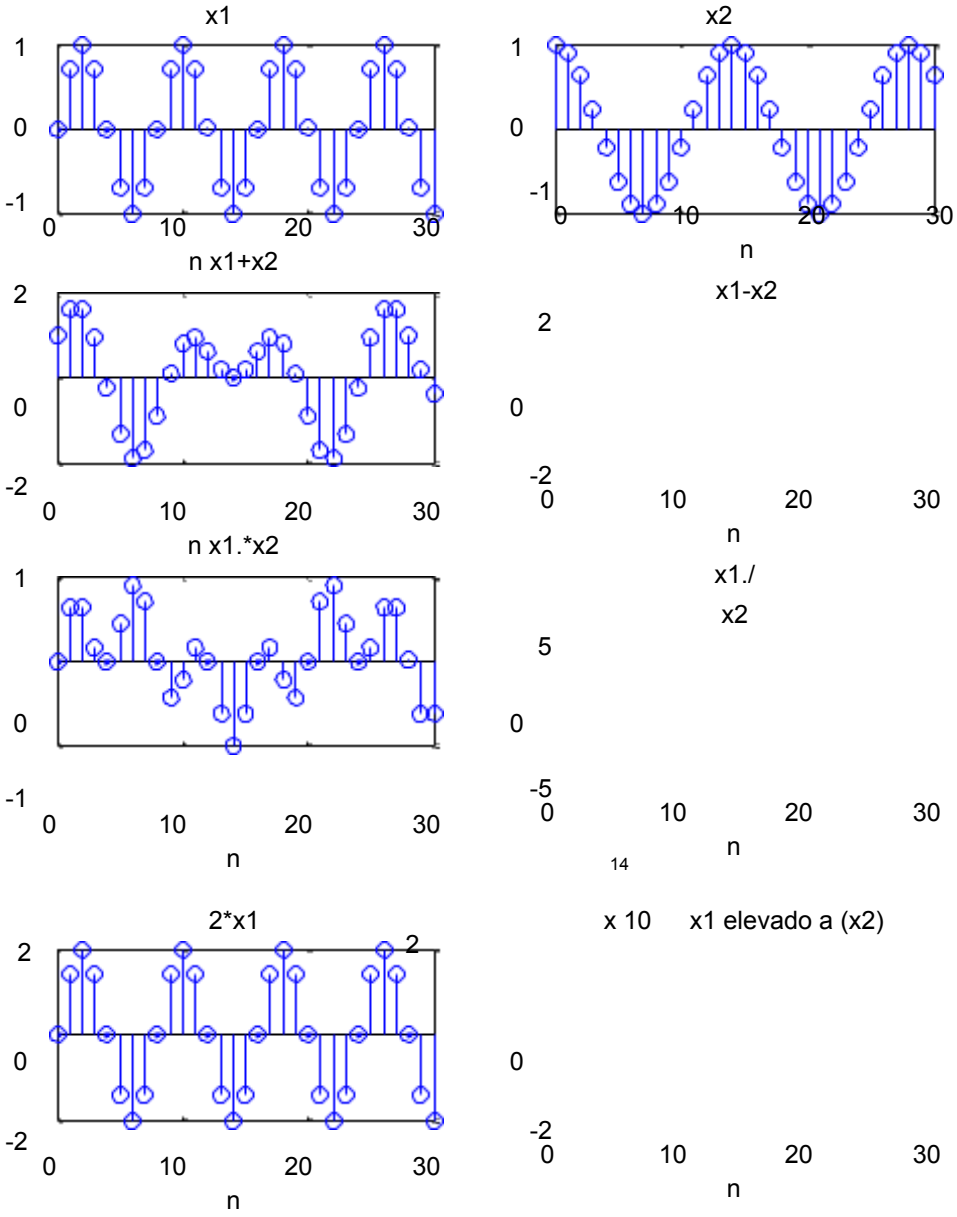
```
>> x1=sin((pi/4)*[0:30]);
>> x2=cos((pi/7)*[0:30]);
```

y efectúe las siguientes operaciones:

```
>> y1=x1+x2;
>> y2=x1-x2;
>> y3=x1.*x2;
>> y4=x1./x2;
>> y5=2*x1;
>> y6=x1.^x2;
```

Observe que en el caso de la multiplicación, división y exponenciación, es necesario preceder el operador de un punto, para indicar que la operación ha de llevarse término a término, en vez de entre matrices (e.g., el producto de matrices requiere que el segundo término tenga tantas filas como columnas tenga el primero, algo que no verifican los vectores **x1** y **x2**).

Represente las siete señales de este apartado en ocho figuras distintas.



1.2.6 Ejercicio 6: *scripts* y funciones

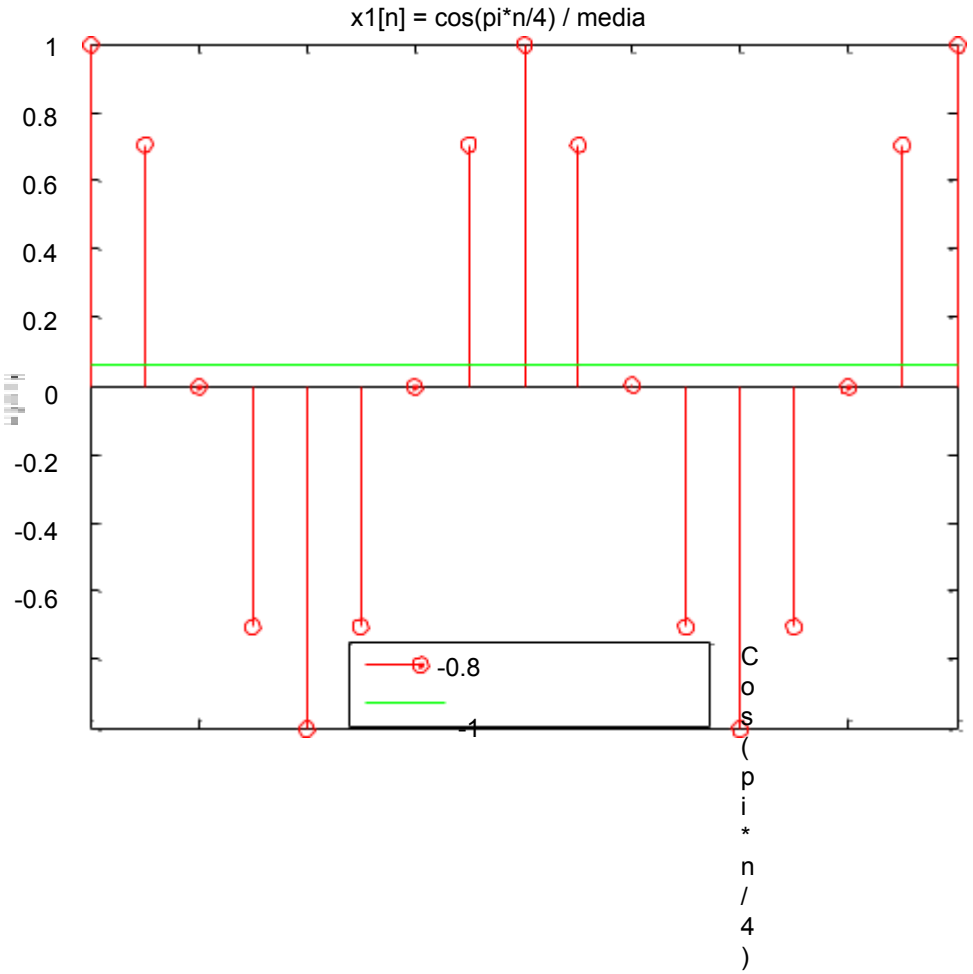
En MATLAB hay esencialmente dos tipos de ficheros con extensión '.m': *scripts* de comandos y funciones. Su uso es imprescindible de cara a organizar, depurar y guardar los ejercicios (en *scripts*) y siempre que se requiera realizar un mismo conjunto de operaciones (es decir, funciones) sobre señales diferentes.

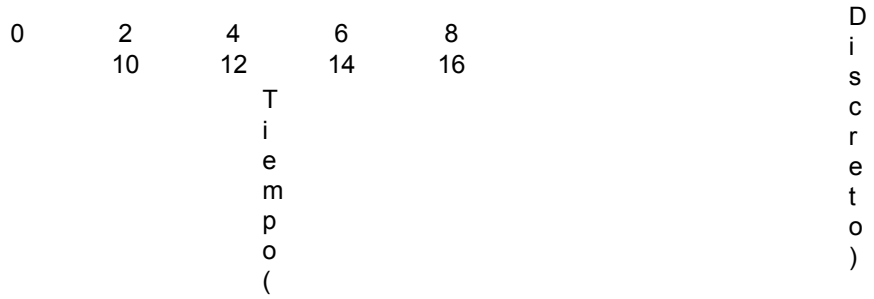
Tenga en cuenta que para poder invocar *scripts* y funciones, los ficheros '.m' que los implementan han de estar en algún lugar referenciado por el *path* de MATLAB. Utilice algún directorio de su unidad de disco privada (por defecto, **h:**\) y añádalo al *path* de MATLAB (menú '*Home/Set path…*'). Recuerde que tendrá que efectuar esta operación cada vez que reinicie su ordenador.

Replique los ejemplos que se presentan a continuación:

Genere con el editor de MATLAB el siguiente *script* (asígnele el nombre 'ejercicio_6.m'), cuyo objetivo es representar una determinada señal discreta en un intervalo dado, calcular su valor medio en el citado intervalo, y representar este valor como una función constante:

```
% ejercicio6.m
>> n = [0:16];
>> y1 = mean(x1);
>> stem(n,x1,'r')
>> title('x1[n]  = cos(pi*n/4) / media')
>> xlabel('Tiempo (Discreto)')
>> ylabel('x1[n]')
>> hold on
>> m1=y1*ones(1,17);
>> plot(n,m1,'g')
>> hold off
>> legend('Cos (pi*n/4)', 'Media (Cos (pi*n / 4))');
```



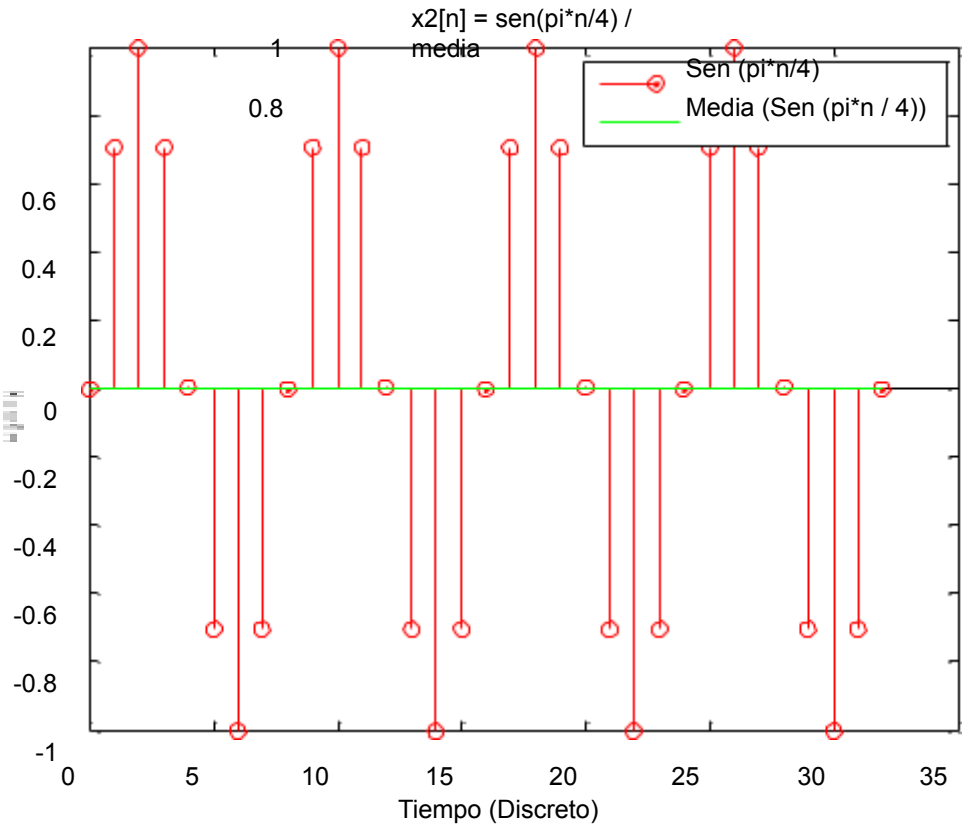


Invoque el *script* creado escribiendo **`ejercicio6`** en la línea de comandos de MATLAB.

Si ahora desea realizar la misma operación con la señal $x_2[n] = \sin(\pi n / 4)$ y en el intervalo $0 \le n \le 32$,

basta con copiar el fichero, asignarle un nuevo nombre (e.g., 'ejercicio_6b.m') y cambiar las líneas que proceda:

```
% ejercicio6b.m
>> n = [0:32];
>> x1 = sin(pi*n/4);
>> y1 = mean(x1);
>> stem(n,x1,'r')
>> title('x1[n] = sen(pi*n/4) / media')
>> xlabel('Tiempo (Discreto)')
>> ylabel('x1[n]')
>> hold on
>> m1=y1*ones(1,33);
>> plot(n,m1,'g')
>> hold off
>> legend('Sen (pi*n/4)', 'Media (Sen (pi*n / 4))');
```



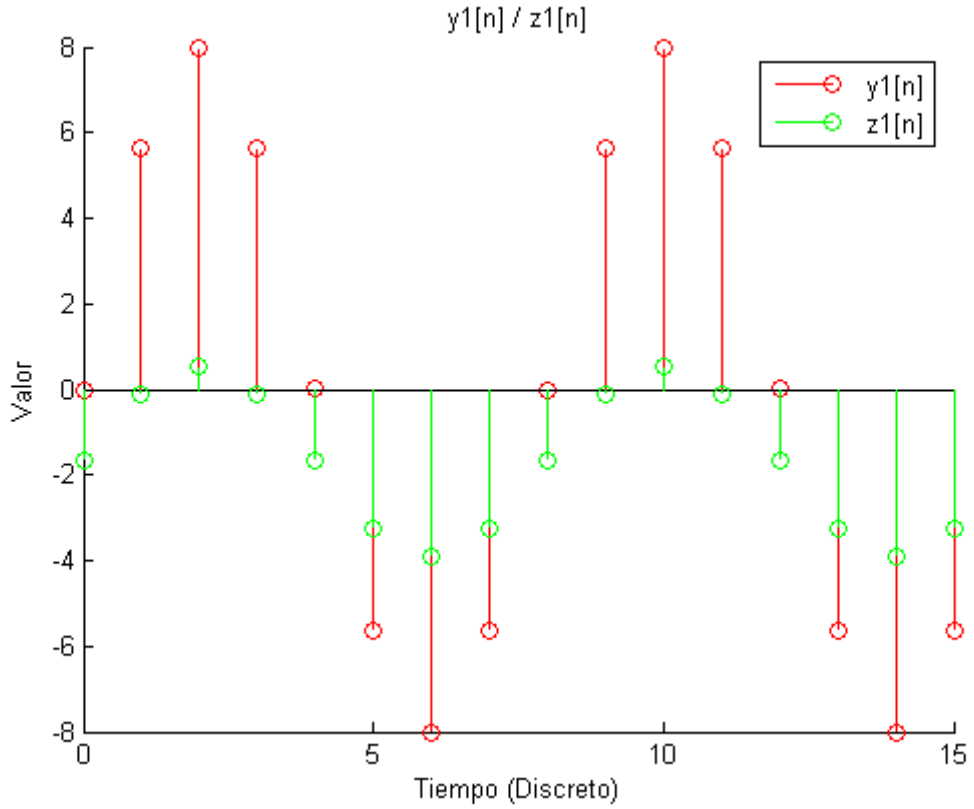
Compruebe que obtiene el resultado deseado ejecutando este segundo *script* desde la línea de comandos.

Un fichero '.m' también puede representar una función. Para ello, la primera palabra del fichero ha de ser **`function`**. El resto de la línea especifica los parámetros que acepta la función y los valores que devuelve. El siguiente ejemplo muestra una función llamada **`f_obtiene_yz`**, que toma como parámetro un vector **`x`**, y devuelve otros dos vectores, **`y`** y **`z`** (acostúmbrese a preceder el nombre de todas las funciones con las letras **`f_`**, de modo que se distingan claramente de funciones de MATLAB o de *scripts*):

```
function [y,z] = f_obtiene_yz(x)
% [y,z] = f_obtiene_yz(x) admite una señal 'x' y
% devuelve dos señales, 'y' y 'z', donde 'y' vale 2*x
% y 'z' vale (5/9)*(x-3)
y = 2.*x;
z = (5/9).*(x-3);
```

El siguiente ejemplo muestra cómo utilizar esta función desde la línea de comandos (o bien desde un *script*):

```
>> n=[0:15];
>> x1=4*sin((pi/4)*n);
>> [y1,z1]=f_obtiene_yz(x1);
>> stem(n,x1);
>> hold on;
>> stem(n,y1,'r');
>> stem(n,z1,'g');
>> hold off;
```



1.2 *Operaciones con señales*

Realice todos los ejercicios que se le solicite en ficheros (*scripts*) '.m', de modo que pueda guardar y modificar sus resultados sin necesidad de volver a teclear el código de nuevo.

1.2.1 Ejercicio 7: transformaciones de la variable independiente

Defina en un fichero la siguiente función discreta, vector **x** y del vector de índices **nx** correspondiente:

$x[n]$, en el intervalo $-3 \leq n \leq 11$, a través de un

$$x[n] = \{2,\ n = 0\ 1,\ n = 2\ \ -\ 1,$$

Represéntela gráficamente y fije las etiquetas necesarias (`xlabel`, `ylabel`, `title`) de modo que el resultado sea similar al que muestra la Fig. 1.

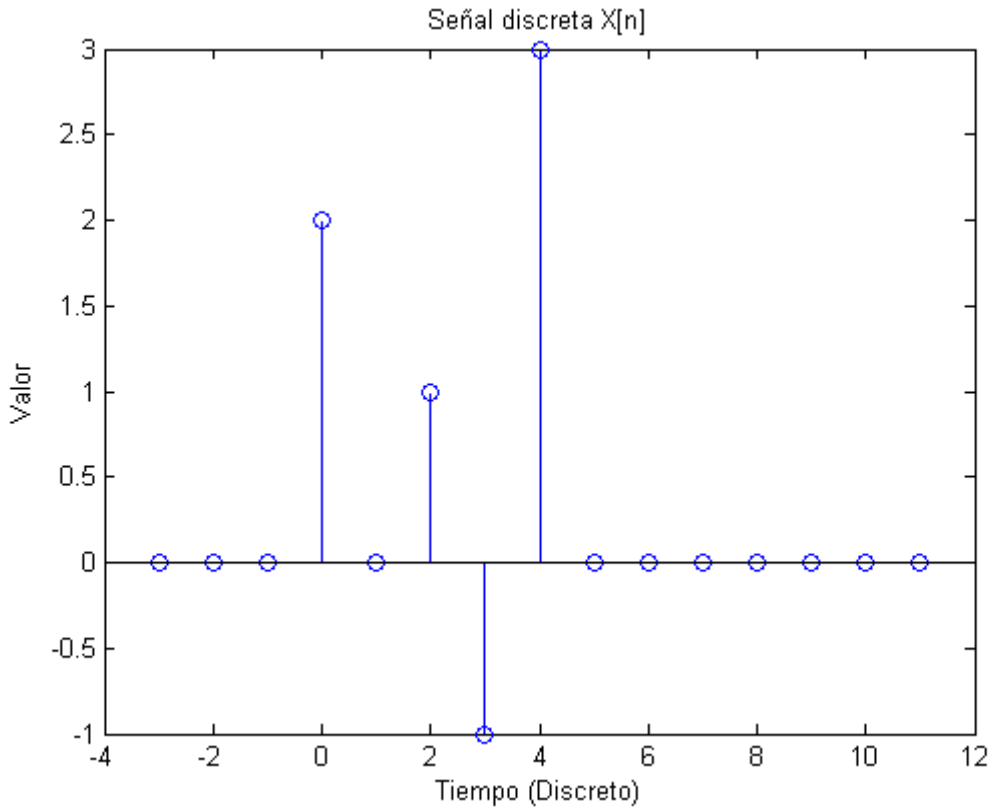


Fig.1: Representación gráfica de la señal original

Una vez definida la señal $x[n]$, defina en el mismo fichero '.m' las siguientes señales:

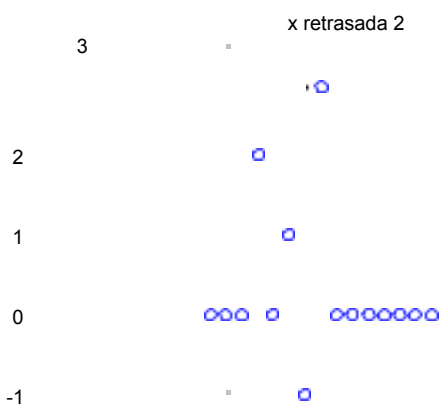
$y_1[n] = x[n-2]$ $y_2[n] = x[n+1]$ $y_3[n] = x[-n]$
$y_4[n] = x[-n+1]$

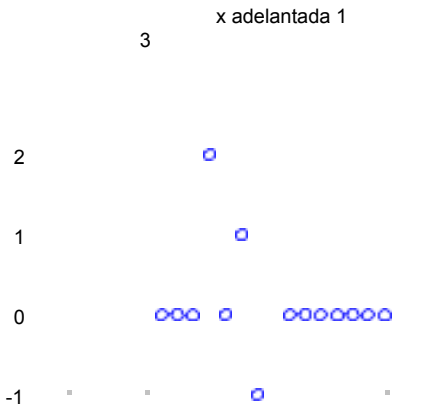
Para ello, el método que se propone, orientado a evidenciar que no cambia la señal sino la variable independiente, consiste en primero definir **`y1=x`**, **`y2=x`**, etc., y a continuación definir los correspondientes vectores índice de cada señal (**`ny1`**,…, **`ny4`**) como una transformación del vector de índices **`nx`**. Para ello dibuje aparte las cuatro señales que se le solicitan, deduzca la relación que existe entre sus respectivos vectores de índices y el vector **`nx`**, y exprese dicha relación mediante MATLAB.

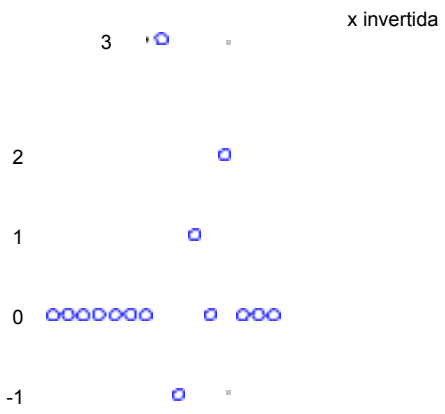
Adicionalmente, para hacer la inversa de la señal puede hacer uso de la función **`fliplr`**.

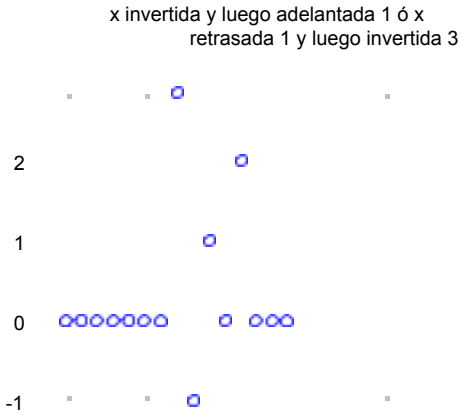
Finalice este apartado representando gráficamente la señal original ($x[n]$) y, en otra ventana, las cuatro señales resultantes de cada transformación (ver el comando **`subplot`**); titule y etiquete cada gráfico. Indique en cada uno cuál es la relación entre la señal representada $y_i[n]$ y la original $x[n]$ "adelantada tres posiciones", "invertida y luego retrasada…").

(e.g.,









Código generado:

```
n = [-3:11]
x = [zeros(1,3) 2 0 1 -1 3 zeros(1,7)]
x_ticks = linspace(-4,12,9);
y_ticks = linspace(-1,3,9);

figure(); stem(n, x);
set_ticks_and_title(x_ticks, y_ticks, 'Señal x[n] discreta');

nx1 = n + 2;
nx2 = n - 1;
nx3 = -n;
nx4 = - n + 1;

figure();
subplot(2,2,1);
stem(nx1, x);
set_ticks_and_title(x_ticks, y_ticks, 'x[n-2]');
```

```
subplot(2,2,2);
stem(nx2, x);
set_ticks_and_title(x_ticks, y_ticks, 'x[n+1]');

subplot(2,2,3);
stem(nx3, x);
set_ticks_and_title(x_ticks, y_ticks, 'x[-n]');

subplot(2,2,4);
stem(nx4, x);
set_ticks_and_title(x_ticks, y_ticks, 'x[-n+1]');

function set_ticks_and_title(x_ticks, y_ticks, my_title)
    xticks(x_ticks);
    yticks(y_ticks);
    title(my_title);
    xlabel('Tiempo (Discreto)');
    ylabel('Valor');
end
```

1.3 *Propiedades de Sistemas SLI y Convolución*

1.3.1. **Convolución**

La función de MATLAB conv calcula la suma de convolución

$$y[n] = \sum_{m=-\infty}^{\infty} h[m]x[n-m]$$

.

Para calcular la suma MATLAB requiere que $x[n]$ y $h[n]$ sean secuencias de duración finita. Si asumimos que $x[n]$ es no nula únicamente en el intervalo $n_x \le n \le n_x+N_x$ y que $h[n]$ es no nula únicamente en el intervalo $n_h \le n \le n_h+N_h$, entonces $y[n]$ es no nula únicamente en el intervalo $(n_x + n_h) \le n \le (n_x + n_h) + N_x + N_h$. Esto significa que conv sólo necesita calcular $y[n]$ para las $N_x + N_h + 1$ muestras de este intervalo.

Si x es un vector N_x-dimensional que contiene las muestras de $x[n]$ en el intervalo $n_x \le n \le n_x+N_x$, y h es un vector N_h-dimensional que contiene las muestras de $h[n]$ en el intervalo $n_h \le n \le n_h+N_h$, entonces y=conv(x,h) devuelve en y las $N_x + N_h + 1$ muestras de $y[n]$ en el intervalo $(n_x + n_h) \le n \le (n_x + n_h) + N_x + N_h$.

Hay que destacar que la función conv no devuelve los índices de las muestras de $y[n]$ almacenadas en el vector y. El usuario de la función conv es el responsable de conocer cuáles son dichos índices en función de los índices de los vectores de entrada.

1.3.2. Ejercicio 8. Propiedades Conmutativa, Asociativa y Distributiva de la Convolución, y su Aplicación a Sistemas Lineales e Invariantes.

En este ejercicio comprobará las propiedades conmutativa, distributiva y asociativa de la convolución con un conjunto específico de señales. Además examinará las implicaciones de estas propiedades en la conexión serie y paralelo de sistemas lineales e invariantes. Los problemas de este ejercicio exploran únicamente sistemas de tiempo discreto. Sin embargo, como se sabe teóricamente, las mismas propiedades son válidas también para sistemas de tiempo continuo.

a) A lo largo del ejercicio emplearemos frecuentemente las tres señales siguientes:

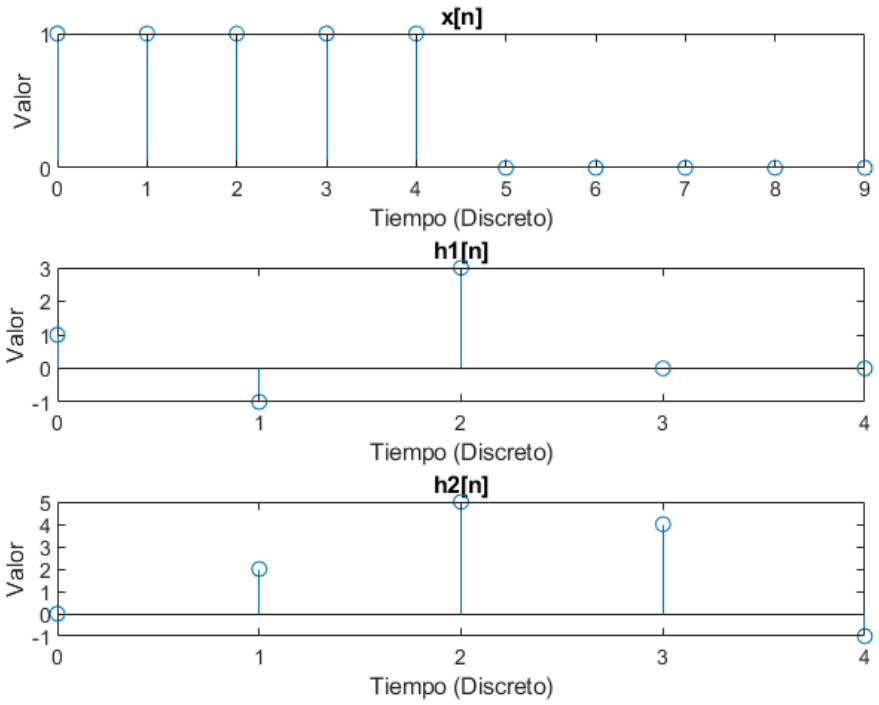
$$x_1[n] = \begin{cases} 1, & 0 \le n \le 4 \\ 0, & resto \end{cases}$$

$$h_1[n] = \begin{cases} 1, & n = 0 \\ -1, & n = 1 \\ 3, & n = 2 \\ 0 & resto \end{cases}$$

$$h_2[n] = \begin{cases} 2, & n = 1 \\ 5, & n = 2 \\ 4, & n = 3 \\ -1, & n = 4 \\ 0, & resto \end{cases}$$

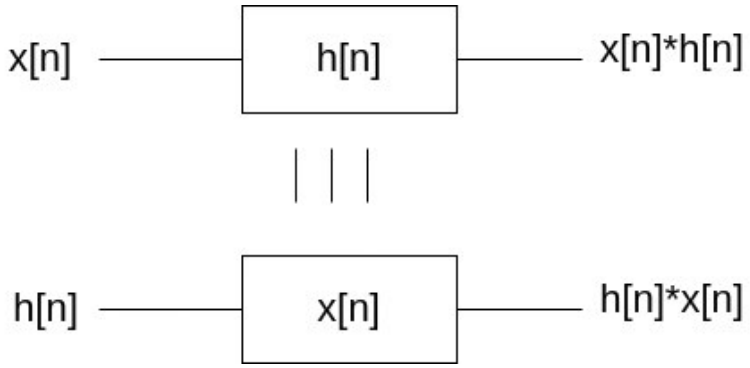
Defina el vector de MATLAB `x1` para representar la señal $x_1[n]$ en el intervalo $0 \le n \le 9$, y los vectores `h1` y `h2` para representar $h_1[n]$ y $h_2[n]$ en el intervalo $0 \le n \le 4$. Defina también los vectores `nx1`, `nh1` y `nh2` para que sean vectores de índices para las señales correspondientes. Represente las tres señales discretas con los índices correctos empleando la función `stem` y dibújelas en el espacio siguiente:

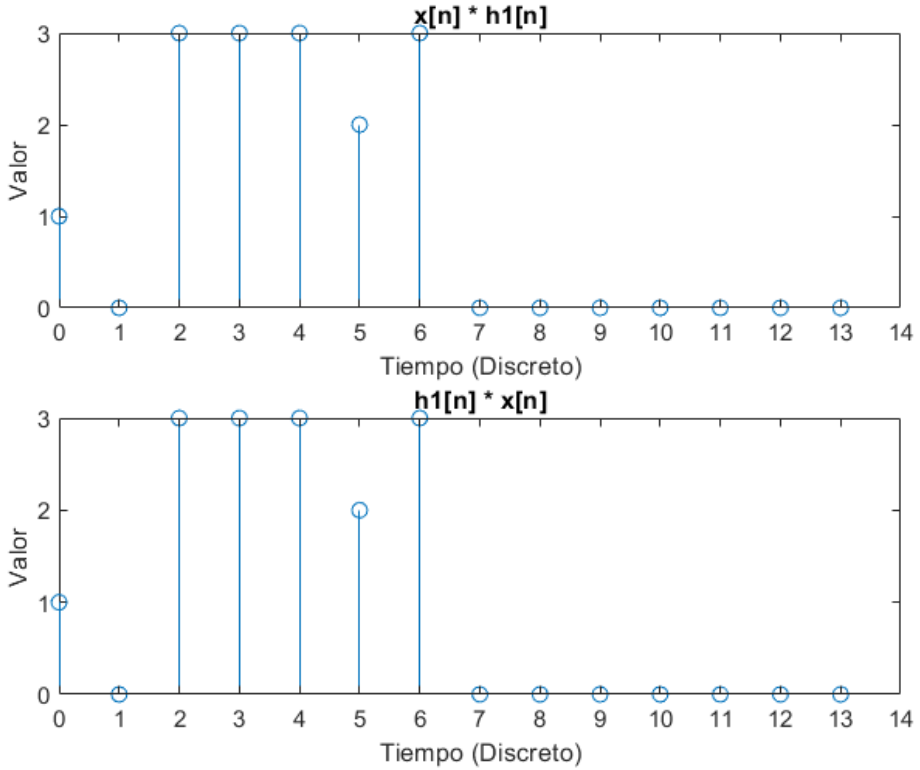
Gráficas



b) La **propiedad conmutativa** establece que el resultado de la convolución es el mismo independientemente del orden de los operandos. Esto implica que la salida de un sistema LTI con respuesta al impulso $h[n]$ cuando la entrada es $x[n]$ es igual que la salida de un sistema LTI con respuesta al impulso $x[n]$ cuando la entrada es $h[n]$. Use la función `conv` para verificar esta propiedad con los vectores `h1` y `x1`. Represente ambas salidas en el espacio siguiente. ¿Es la salida de la función `conv` la misma independientemente del orden de los argumentos de entrada?

Gráficas:





Comentarios:

Como podemos apreciar en las gráficas mostradas, el resultado de la convolución es independiente del orden de los factores. Esto es, **la convolución es conmutativa**.

c) La convolución también tiene la **propiedad distributiva** respecto de la suma. Esto quiere decir que:

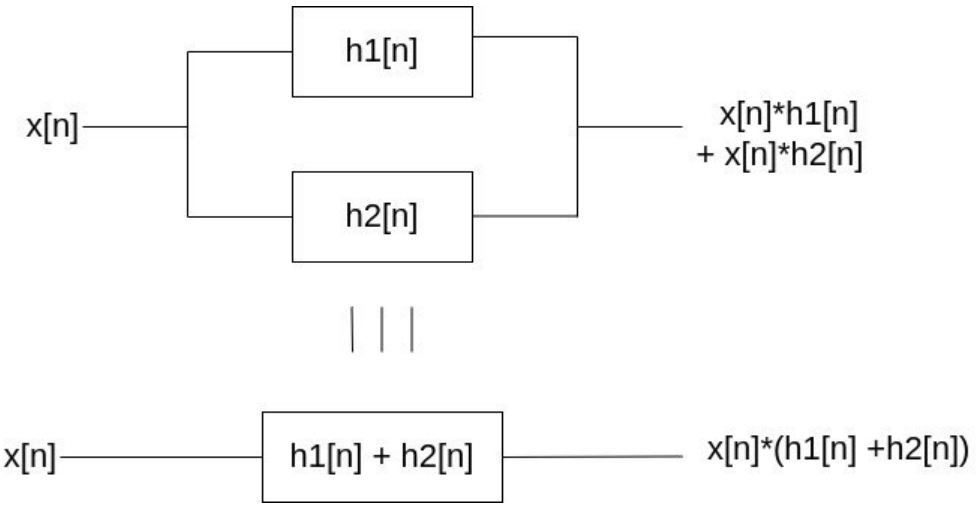
$$x[n] * (h_1[n] + h_2[n]) = x[n] * h_1[n] + x[n] * h_2[n]$$

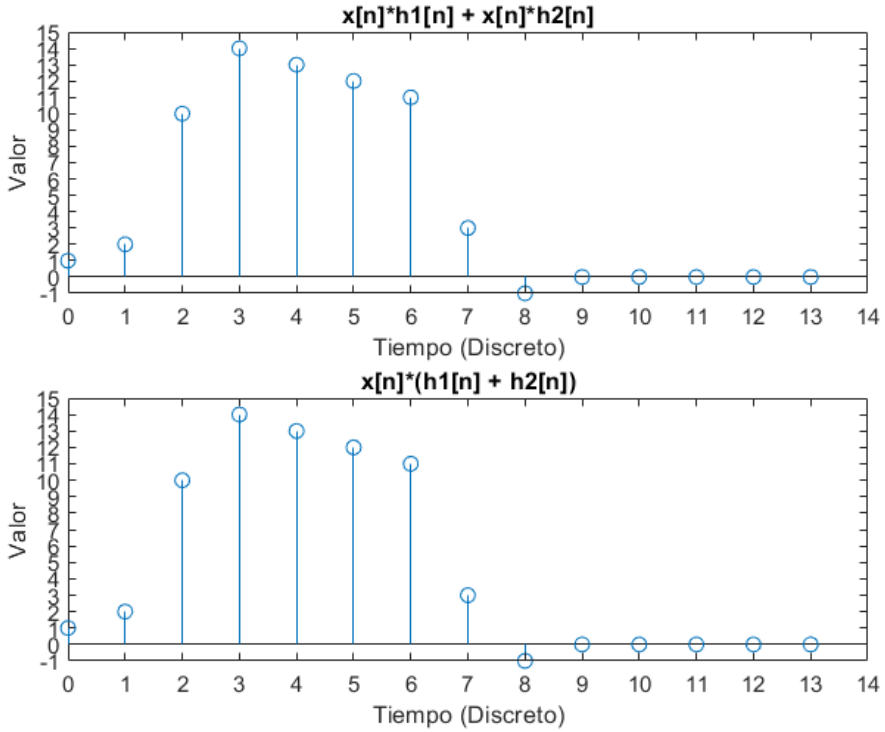
Esto implica que la salida de dos sistemas LTI conectados en paralelo es la misma que la salida de un sistema cuya respuesta al impulso es la suma de las respuestas al impulso de los sistemas conectados en paralelo. ken el espacio siguiente dos

diagramas de bloques, uno con dos sistemas conectados en paralelo y otro con el sistema único equivalente.

Compruebe la propiedad distributiva empleando los vectores `x1`, `h1` y `h2`. Para ello calcule la suma de las salidas de los dos sistemas LTI con respuestas al impulso $h_1[n]$ y $h_2[n]$ cuando $x_1[n]$ es la señal de entrada. Posteriormente calcule la salida de un sistema LTI cuya respuesta al impulso es la suma de $h_1[n]$ y $h_2[n]$ cuando $x_1[n]$ es la señal de entrada. Represente ambas salidas en el espacio adjunto y compárelas. ¿Producen estos dos métodos la misma salida?

Gráficas:





Comentarios:

Como podemos apreciar en las gráficas, **se comprueba la propiedad distributiva de la convolución**: es equivalente aplicar ambos sistemas y sumar las salidas a aplicar un único sistema con la suma de los mismos.

d) La convolución también tiene la **propiedad asociativa**, es decir

$$x[n] * \left(h_1[n] * h_2[n]\right) = \left(x[n] * h_1[n]\right) * h_2[n] = x[n] * h_1[n] * h_2[n]$$

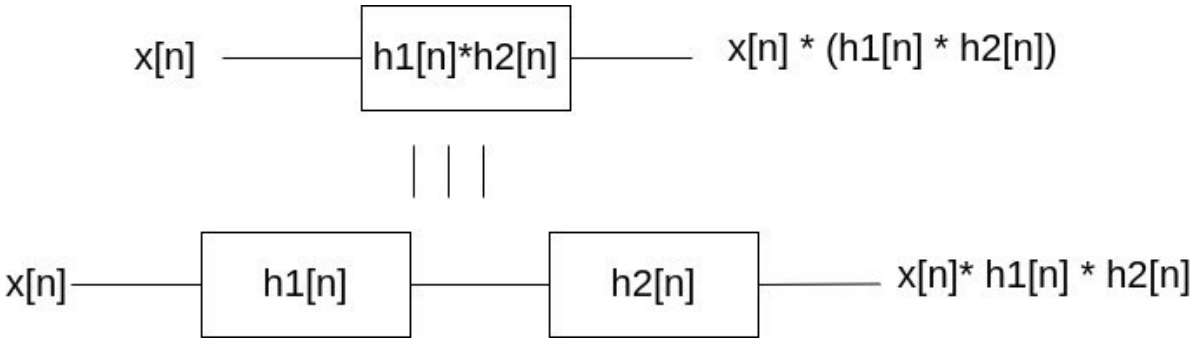
Esta propiedad implica que el resultado de procesar una señal con una serie de sistemas LTI conectados en cascada es equivalente a procesar la señal con un único sistema LTI cuya respuesta al impulso es la convolución de todas las respuestas al impulso de los sistemas LTI conectados en cascada. Dibuje en el espacio siguiente dos diagramas de bloques que ejemplifiquen esta propiedad para dos sistemas conectados en cascada.

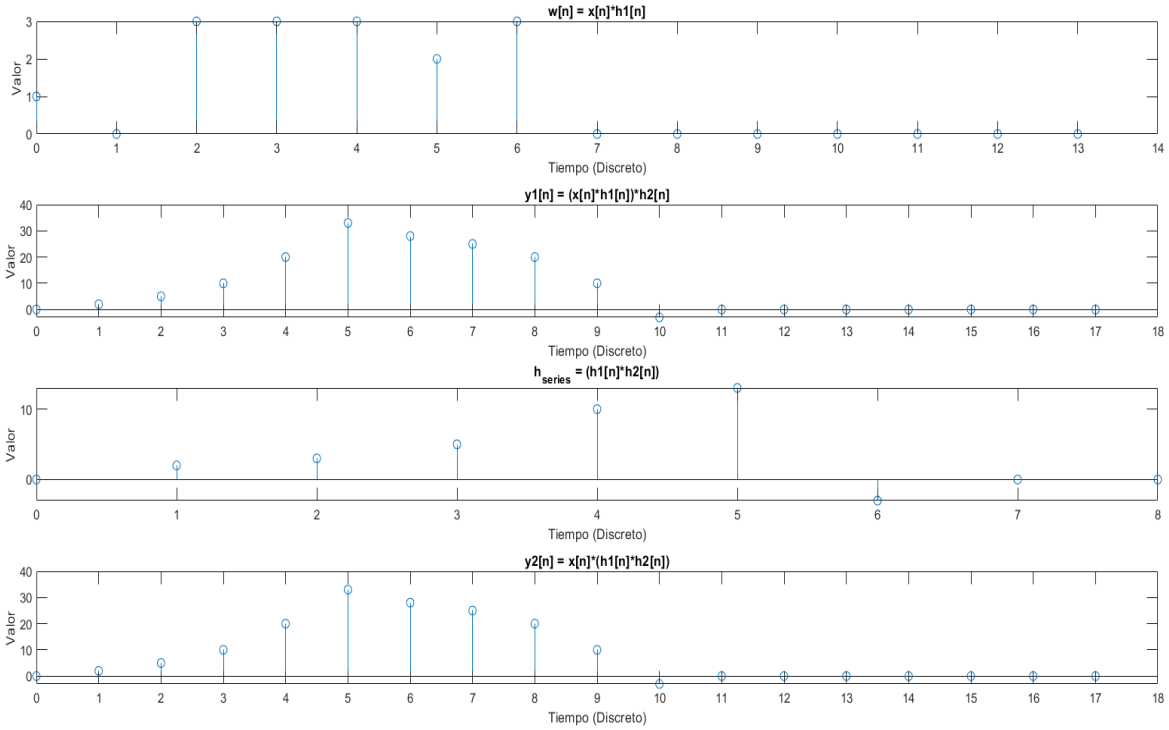
Siga los siguientes pasos para comprobar la propiedad asociativa empleando los vectores `x1`, `h1` y `h2`.

- Calcule $w[n]$ como la salida de un sistema LTI con respuesta al impulso $h_1[n]$ cuando la entrada al sistema es $x_1[n]$. Represente esta señal en el espacio siguiente.
- Calcule la salida del sistema, $y_1[n]$, conectado en cascada como la salida de un sistema LTI con respuesta al impulso $h_2[n]$ cuando la entrada al sistema es $w[n]$. Represente esta señal en el espacio siguiente.
- Calcule ahora la respuesta al impulso del sistema equivalente a la conexión en cascada de los dos sistemas $h_1[n]$ y $h_2[n]$. Denomine $h_{series}[n]$ a dicha respuesta al impulso y represéntela en espacio siguiente.
- Por último calcule la salida, $y_2[n]$, del sistema equivalente, $h_{series}[n]$, cuando la entrada al sistema es $x_1[n]$. Represente esta señal en el espacio siguiente.

Compare las dos salidas calculadas $y_1[n]$ y $y_2[n]$. ¿Obtuvo los mismos resultados al procesar $x_1[n]$ con los sistemas individuales y con el sistema equivalente?.

Gráficas:





Comentarios:

Como podemos apreciar en las gráficas anteriores, las respuestas obtenidas para y1 e y2 (en las gráficas 2 y 4 respectivamente) coinciden. **Esto comprueba empíricamente la propiedad asociativa de la convolución**.

e) Suponga que tiene dos sistemas LTI con respuestas al impulso $h_{e1} = h_1[n]$ y $h_{e2} = h_1[n-n_0]$, siendo n_0 un entero. Llamemos $y_{e1}[n]$ e $y_{e2}[n]$ a las salidas de dichos sistemas cuando la entrada es cualquier señal $x[n]$. Utilice la propiedad conmutativa para argumentar que las salidas serían las mismas si intercambiase la entrada y la respuesta al impulso de cada uno de los sistemas. Observe que, una vez que ha hecho este intercambio, los dos sistemas tienen la misma respuesta al impulso y que las entradas son versiones retardadas de la misma señal. Basándose

en esta observación y en la invarianza temporal, demuestre que $y_{e2}[n] = y_{e1}[n-n_0]$. Utilice MATLAB para comprobar esto para el caso $n_0 = 2$, y $x[n] = x_1[n]$. Comente el procedimiento seguido y dibuje las señales de salida resultantes.

Demostraciones:

Definimos:

$$y_{e1}[n] = x[n] * h_1[n]$$

$$y_{e2}[n] = x[n] * h_{e2}[n] = x[n] * h_1[n - n_0]$$

para cierto n_0 entero.

Por la propiedad de invarianza temporal sabemos que para todo k entero se cumple que

$$y_{e1}[n - k] = x[n - k] * h_1[n]$$

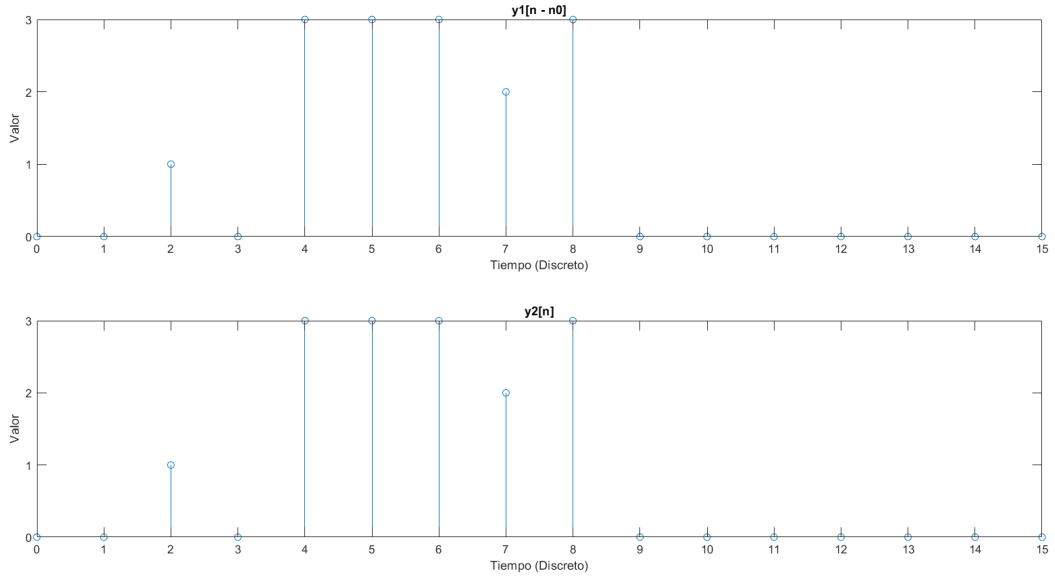
Utilizando la propiedad conmutativa vemos que:

$$y_{e1}[n] = x[n] * h_1[n] = h_1[n] * x[n]$$

Aplicando ahora la propiedad de invarianza obtenemos:

$$y_{e1}[n - k] = h_1[n - k] * x[n] = y_{e2}[n]$$

Gráficas



Comentarios:

Como cabía esperar tras la demostración realizada, las salidas coinciden.

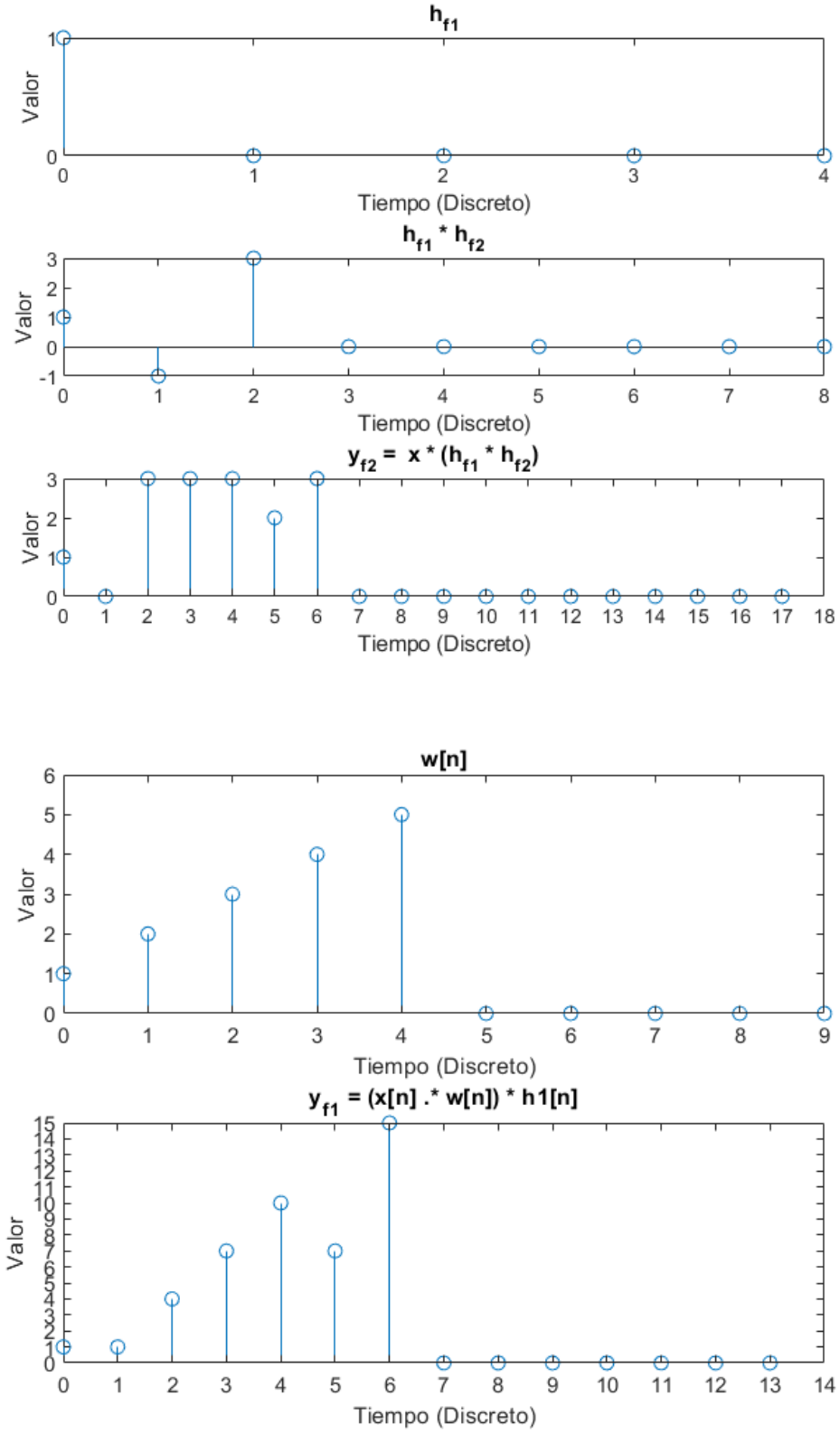
f) Considere **dos sistemas conectados en serie** que denominaremos sistema 1 y sistema 2. Suponga que el sistema 1 es un sistema sin memoria y que está caracterizado por la relación de entrada-salida $y[n] = (n+1)x[n]$, y que el sistema 2 es un sistema LTI con respuesta al impulso $h_{f2}[n] = h_1[n]$, como se definía en el apartado a). Usted desea investigar si la propiedad asociativa de la convolución se mantiene para la conexión serie de estos dos sistemas. Para ello sigue los siguiente pasos:

Calcule $w[n]$ como la salida del sistema 1 cuando la entrada al sistema es $x_1[n]$. Utilice los vectores `nx1` y `x1` con el operador de multiplicación elemento a elemento de MATLAB `.*` para definir un vector de MATLAB `w` que represente $w[n]$. Represente esta señal en el espacio siguiente.

- Utilice $w[n]$ como la entrada al sistema 2, y llame a la salida de dicho sistema $y_{f1}[n]$. Calcule `yf1` en MATLAB empleando para ello `w` y `h1`. Represente la señal obtenida en el espacio siguiente.
- Llame $h_{f1}[n]$ a la salida del sistema 1 cuando la entrada es el impulso unidad. Defina un vector `hf1` que represente esta señal sobre el intervalo $0 \leq n \leq 4$. Represente también la señal obtenida en el espacio siguiente.
- Llame $h_{series}[n] = h_{f1}[n] * h_{f2}[n]$. Calcule el vector `hseries` que represente esta señal. Represente la señal obtenida en el espacio siguiente.
- Por último calcule $y_{f2}[n]$ como la salida del sistema $h_{series}[n]$ cuando la entrada al mismo es $x_1[n]$. Calcule el vector correspondiente `yf2`. Represente la señal obtenida en el espacio siguiente.

¿Coinciden $y_{f1}[n]$ e $y_{f2}[n]$?. Si es así, ¿por qué sería esto de esperar?. Si no es así, esto significa que la interconexión en serie de los sistemas 1 y 2 no es equivalente a un sistema cuya respuesta al impulso es la convolución de la respuesta al impulso de los sistemas 1 y 2. ¿Viola esto la propiedad asociativa de la convolución discutida en el apartado d)?.

Gráficas:



Comentarios:

Como podemos apreciar en las gráficas, las señales de salida y_f1 e y_f2 no coinciden. Esto se debe a que el sistema 1 dado por la expresión $y[n] = (n+1)x[n]$ **no es LTI.** Por lo tanto, la propiedad de asociatividad no está garantizada cuando se utiliza este sistema. Es por ello que no se viola la propiedad asociativa de la convolución entre sistemas LTI.

g) Considere la **conexión en paralelo** de dos sistemas, el sistema 1 y el sistema 2. El sistema 1 es un sistema sin memoria caracterizado por la relación de entrada-salida $y[n]= x[n]^2$. El sistema 2 es un sistema LTI con respuesta al impulso $h_{g2}[n]= h_2[n]$, como se definía en el apartado a). Realice los siguientes pasos para determinar si la propiedad distributiva de la convolución se mantiene para la conexión en paralelo de los dos sistemas.

Calcule $y_{ga}[n]$ como la salida del sistema 1 cuando la entrada al sistema es $x_g[n]$ definida como un impulso unitario multiplicado por 2. Defina el vector `xg` para que represente esta señal de entrada en el intervalo $0 \leq n \leq 4$, y use dicho vector y el operador de exponenciación elemento a elemento de MATLAB `.^` para definir un vector de MATLAB `yga` que represente $y_{ga}[n]$. Represente esta señal en el espacio siguiente.

Calcule $y_{gb}[n]$ como la salida del sistema 2 cuando la entrada al sistema es $x_g[n]$, definida en el punto anterior. Defina un vector de MATLAB `ygb` que represente $y_{gb}[n]$. Represente esta señal en el espacio siguiente.

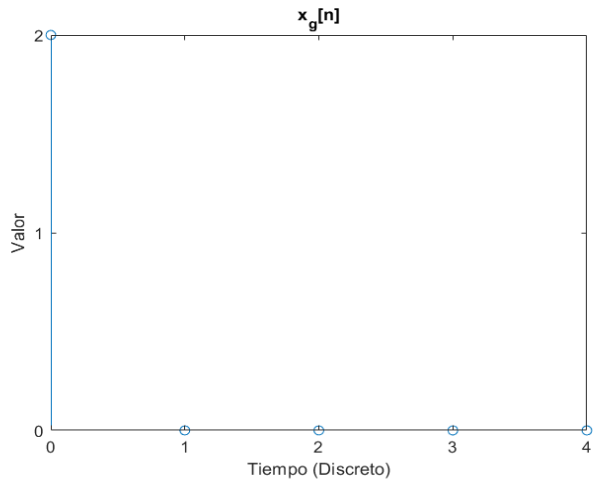
Calcule $y_{g1}[n]$ como la suma de $y_{ga}[n]$ e $y_{gb}[n]$. Defina un vector de MATLAB `yg1` que represente $y_{g1}[n]$. Observe que, dado que `yga` es más corta que `ygb`, es necesario extender el vector `yga` con más ceros antes de poder sumarlos. Represente la señal $y_{g1}[n]$ en el espacio siguiente.

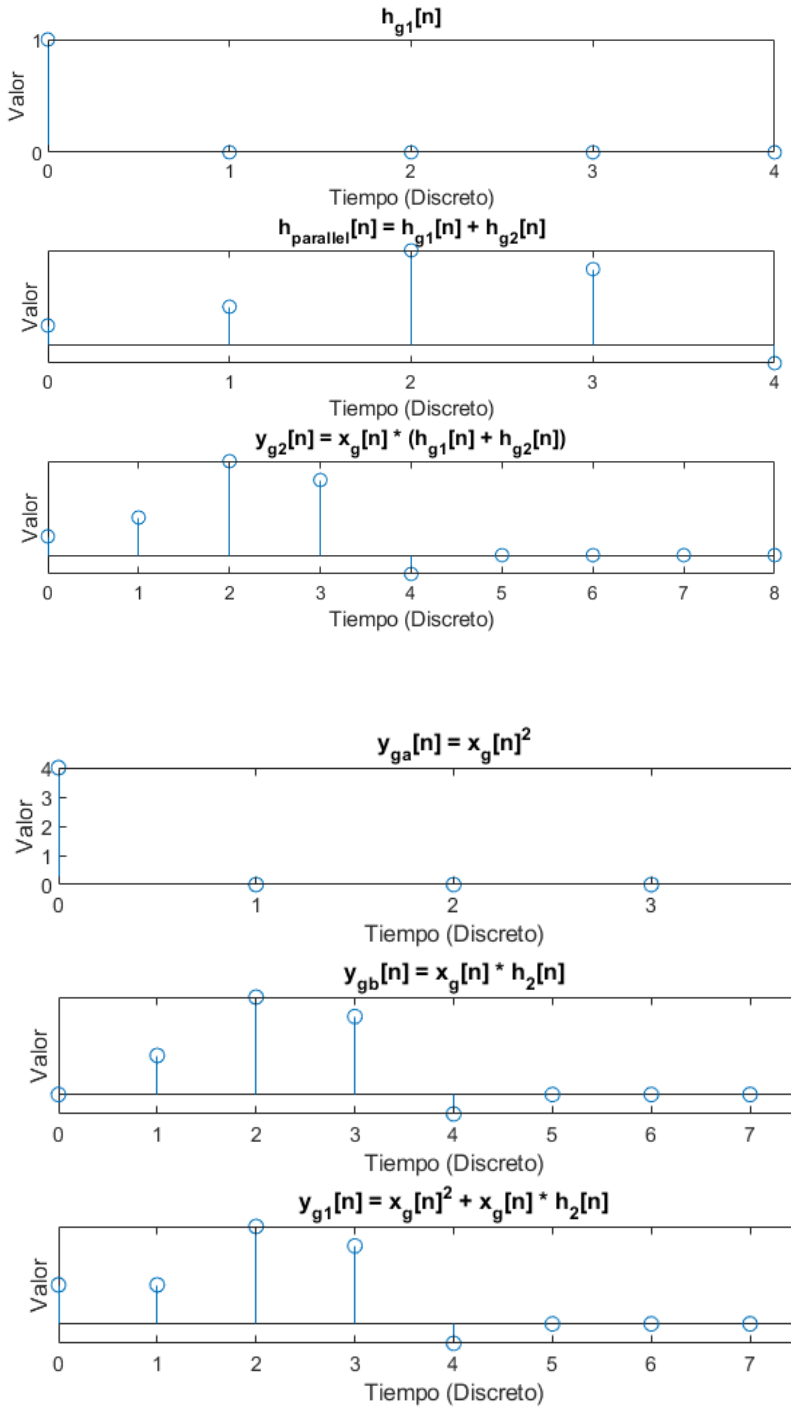
- Llame $h_{g1}[n]$ a la salida del sistema 1 cuando la entrada es el impulso unidad. Defina un vector `hg1` que represente esta señal sobre el intervalo $0 \leq n \leq 4$. Represente también la señal obtenida en el espacio siguiente.
- Llame $h_{parallel}[n] = h_{g1}[n]+h_{g2}[n]$. Calcule el vector `hparallel` que represente esta señal. Represente la señal obtenida en el espacio siguiente.

Por último calcule $y_{g2}[n]$ como la salida del sistema $h_{parallel}[n]$ cuando la entrada al mismo es $x_g[n]$. Calcule el vector correspondiente `yg2`. Represente la señal obtenida en el espacio siguiente.

¿Coinciden los vectores `yg1` e `yg2`? ¿Esperaba este resultado? Si no coinciden, ¿se ha violado la propiedad distributiva de la convolución?

Gráficas:





Comentarios:

Podemos observar como las distintas salidas no coinciden: y_g1 e y_g2 difieren en n=0. Sin embargo, esto no viola la propiedad distributiva de la convolución, pues esta se da para sistemas LTI. Sin embargo, el sistema 1 que “eleva la imagen al cuadrado” no es independiente del tiempo.