Nombre: **Jorge Augusto Balsells Orellana**

Hoja de respuesta Trabajo 2:
Series de Fourier de Tiempo Discreto

Para la revisión de la actividad debe de entregar esta hoja de respuestas en formato .docx. No entregue el guión.

Tarea 1: DTFS de señales periódicas

Rellene la siguiente tabla con la respuesta a las preguntas de esta tarea, y entregue un fichero tarea1.m con los comandos usados. Incluya todos los comandos y sus parámetros, no haga un resumen de los comandos.

Periodo fundamental de $x_1[n]$	$N = 6$
Periodo fundamental de $x_2[n]$	$N = 4$

Programa que calcula la DTFS de cada señal y las representa

```
Nota: Para la señal x1[n] (solo para ésta), una vez calculados
los coeficientes con su fórmula específica, aplicar el siguiente
comando de redondeo:

cm=0+round(cm*10^10)/10^10;

clear
clc

%Calculo de DTFS
%c = (1/N)fft(x)
%x = N*ifft(C)

%x1[n]
```

```
subplot(2,5,1);
n1=[0:5];
x1 = exp(-j*(pi/3)*n1);
N1 = 6 % pi/3 = 2*pi/N1
S = stem(n1,x1);
S.LineWidth = 1.5;
S.Marker = 'diamond';
S.MarkerFaceColor = 'red';
title("x1[n]")
xlabel("n")
ylabel("x[n]")
grid on

%Re(x1[n])
subplot(2,5,2);
c1 = (1/N1)*fft(x1);
c1=0+round(c1*10^10)/10^10
S = stem(n1,real(c1))
S.LineWidth = 1.5;
S.Marker = 'diamond';
S.MarkerFaceColor = 'red';
title("Re(C1m)")
xlabel("m")
ylabel("Cm")
grid on

%Im(x1[n])
subplot(2,5,3);
S = stem(n1,imag(c1))
S.LineWidth = 1.5;
S.Marker = 'diamond';
S.MarkerFaceColor = 'red';
title("Im(C1m)")
xlabel("m")
ylabel("Cm")
grid on

%Modulo(x1[n])
subplot(2,5,4);
S = stem(n1,abs(c1))
S.LineWidth = 1.5;
S.Marker = 'diamond';
S.MarkerFaceColor = 'red';
```

```
title("Abs(C1m)")
xlabel("m")
ylabel("Cm")
grid on

%(x1[n])
subplot(2,5,5);
S = stem(n1,angle(c1))
S.LineWidth = 1.5;
S.Marker = 'diamond';
S.MarkerFaceColor = 'red';
title("Ang(C1m)")
xlabel("m")
ylabel("Cm")
grid on

%x2[n]
subplot(2,5,6);
n2=[0:3];
x2 = [2 2 2 0];
N2 = 4 % Definido por el modulo
S = stem(n2,x2);
S.LineWidth = 1.5;
S.Marker = 'diamond';
S.MarkerFaceColor = 'red';
title("x2[n]")
xlabel("n")
ylabel("x[n]")
grid on

%Re(x2[n])
subplot(2,5,7);
c2 = (1/N2)*fft(x2);
S = stem(n2,real(c2))
S.LineWidth = 1.5;
S.Marker = 'diamond';
S.MarkerFaceColor = 'red';
title("Re(C2m)")
xlabel("m")
ylabel("Cm")
grid on

%Im(x2[n])
```

```
subplot(2,5,8);
S = stem(n2,imag(c2))
S.LineWidth = 1.5;
S.Marker = 'diamond';
S.MarkerFaceColor = 'red';
title("Im(C2m)")
xlabel("m")
ylabel("Cm")
grid on

%(x2[n])
subplot(2,5,9);
S = stem(n2,abs(c2))
S.LineWidth = 1.5;
S.Marker = 'diamond';
S.MarkerFaceColor = 'red';
title("Abs(C2m)")
xlabel("m")
ylabel("Cm")
grid on

%(x2[n])
subplot(2,5,10);
S = stem(n2,angle(c2))
S.LineWidth = 1.5;
S.Marker = 'diamond';
S.MarkerFaceColor = 'red';
title("Ang(C2m)")
xlabel("m")
ylabel("Cm")
grid on
```

Representación de las 2 señales con subplot() de 2 filas 5 columnas. Por cada señal represente la señal y sus coeficientes: parte real, imaginaria, módulo y ángulo

Gráfico aplicando redondeo:

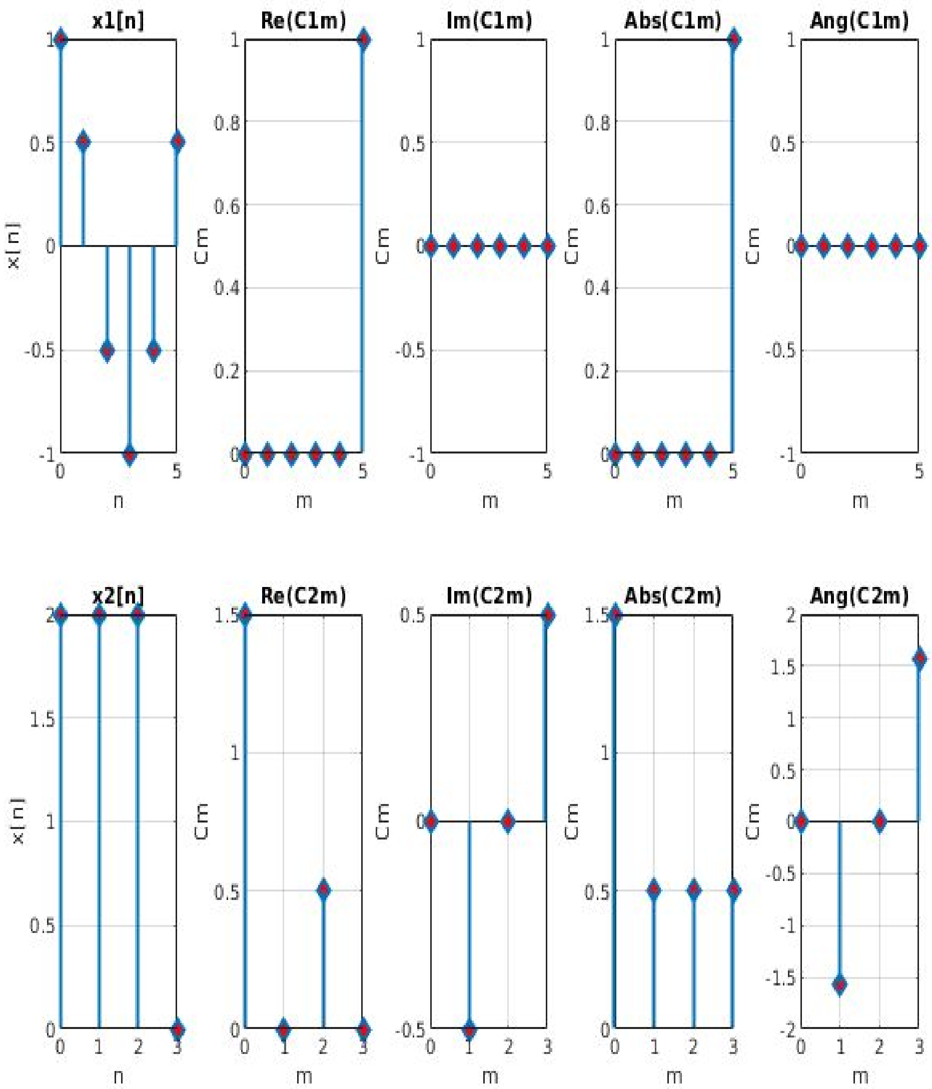
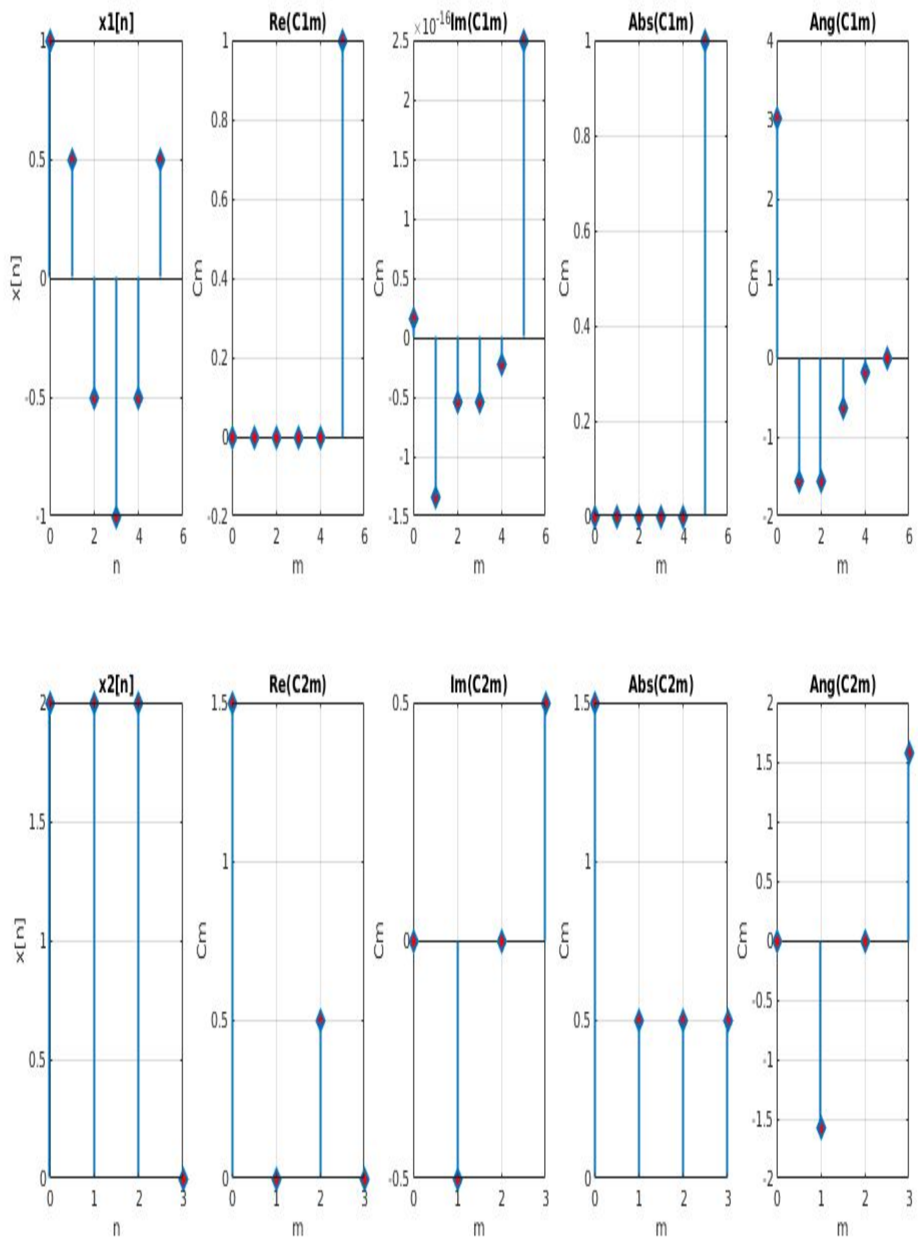


Gráfico sin aplicar redondeo:



Indique si se cumplen las propiedades de simetría del conjugado en sus coeficientes (clase 11). Si no se cumplen indique por qué. [2]

```
clear
clc

%Calculo de DTFS
%c = (1/N)fft(x)
%x = N*ifft(C)

%x1[n]
subplot(2,5,1);
n0 = [0:5];
n1=[-6:-1];
x0 = exp(-j*(pi/3)*n0);
x1 = exp(-j*(pi/3)*n1);

N1 = 6 % pi/3 = 2*pi/N1
S = stem([n1 n0],[x1 x0]);
S.LineWidth = 1.5;
S.Marker = 'diamond';
S.MarkerFaceColor = 'red';
title("x1[n]")
xlabel("n")
ylabel("x[n]")
grid on

%Re(x1[n])
subplot(2,5,2);

c0 = (1/N1)*fft(x0);
c0=0+round(c0*10^10)/10^10
c1 = (1/N1)*fft(x1);
c1=0+round(c1*10^10)/10^10

hold on
S = stem([n1 n0], [real(c1) real(c0)])
S.LineWidth = 1.5;
S.Marker = 'diamond';
S.MarkerFaceColor = 'red';
```

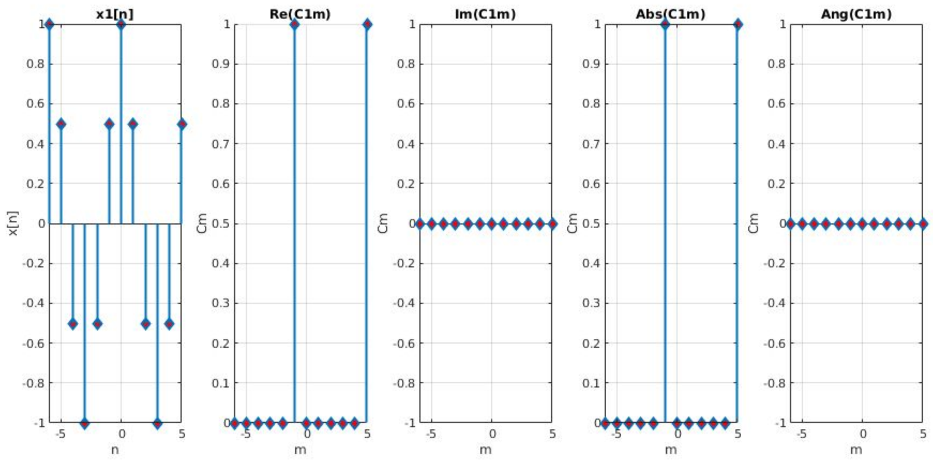
```
title("Re(C1m)")
xlabel("m")
ylabel("Cm")
grid on

%Im(x1[n])
subplot(2,5,3);
S = stem([n1 n0],[imag(c1) imag(c0)])
S.LineWidth = 1.5;
S.Marker = 'diamond';
S.MarkerFaceColor = 'red';
title("Im(C1m)")
xlabel("m")
ylabel("Cm")
grid on

%Modulo(x1[n])
subplot(2,5,4);
S = stem([n1 n0],[abs(c1) abs(c0)])
S.LineWidth = 1.5;
S.Marker = 'diamond';
S.MarkerFaceColor = 'red';
title("Abs(C1m)")
xlabel("m")
ylabel("Cm")
grid on

%(x1[n])
subplot(2,5,5);
S = stem([n1 n0],[angle(c1) angle(c0)])
S.LineWidth = 1.5;
S.Marker = 'diamond';
S.MarkerFaceColor = 'red';
title("Ang(C1m)")
xlabel("m")
ylabel("Cm")
grid on



%x2[n]
subplot(2,5,6);
n2=[0:3];
```

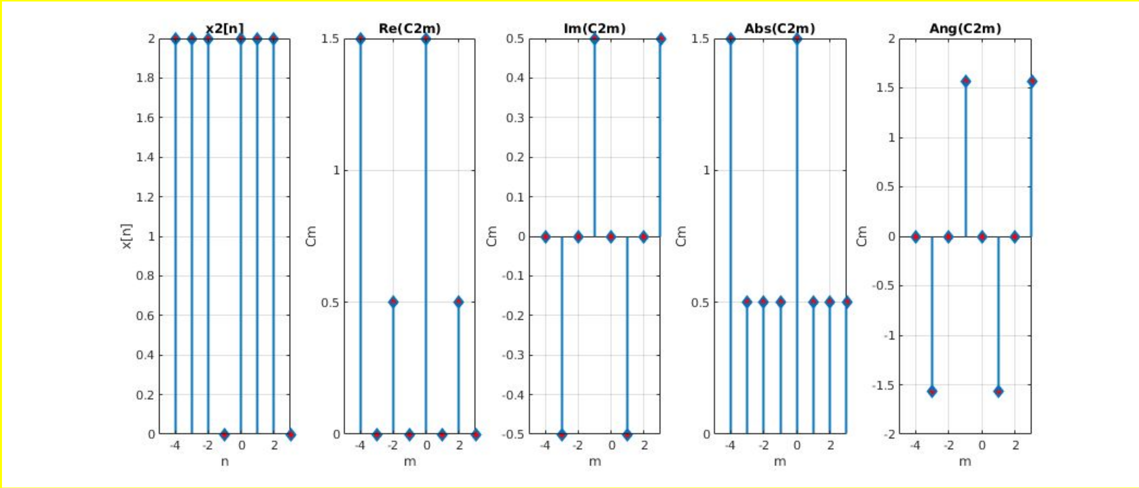
```
n3=[-4:-1]

x2 = [2 2 2 0];
x3 = [2 2 2 0];

N2 = 4 % Definido por el modulo
S = stem([n3 n2],[x3 x2]);
S.LineWidth = 1.5;
S.Marker = 'diamond';
S.MarkerFaceColor = 'red';
title("x2[n]")
xlabel("n")
ylabel("x[n]")
grid on

%Re(x2[n])
subplot(2,5,7);
c2 = (1/N2)*fft(x2);
c3 = (1/N2)*fft(x3);

S = stem([n3 n2],[real(c3) real(c2)])
S.LineWidth = 1.5;
S.Marker = 'diamond';
S.MarkerFaceColor = 'red';
title("Re(C2m)")
xlabel("m")
ylabel("Cm")
grid on

%Im(x2[n])
subplot(2,5,8);
S = stem([n3 n2],[imag(c3) imag(c2)])
S.LineWidth = 1.5;
S.Marker = 'diamond';
S.MarkerFaceColor = 'red';
title("Im(C2m)")
xlabel("m")
ylabel("Cm")
grid on

%(x2[n])
subplot(2,5,9);
S = stem([n3 n2],[abs(c3) abs(c2)])
```

```
S.LineWidth = 1.5;
S.Marker = 'diamond';
S.MarkerFaceColor = 'red';
title("Abs(C2m)")
xlabel("m")
ylabel("Cm")
grid on

%(x2[n])
subplot(2,5,10);
S = stem([n3 n2],[angle(c3) angle(c2)])
S.LineWidth = 1.5;
S.Marker = 'diamond';
S.MarkerFaceColor = 'red';
title("Ang(C2m)")
xlabel("m")
ylabel("Cm")
grid on
```

Al tratarse x1 de una función compleja, indicar si las siguientes funciones deben ser pares o impares y comprobarlo:

- $Real\{c_m\}$: **Funcion Impar**
- $Imag\{c_m\}$: **Funcion Par**
- $mod|c_m|$: **Funcion Impar**
- $Angulo\ \{c_m\}$: **Funcion Par**



Al tratarse x2 de una función real, indicar si las siguientes funciones deben ser pares o impares y comprobarlo: ● $Real\{c_m\}$: **Funcion Par** ● $Imag\{c_m\}$: **Funcion Impar** ● $mod\lvert c_m \rvert$: **Funcion Par** ● $Angulo\ \{c_m\}$: **Funcion Impar**


Tarea 2: Obtención de la señal a partir de coeficientes

Rellene la siguiente tabla con la respuesta a las preguntas de esta tarea, y entregue un fichero tarea2.m con los comandos usados.

Coeficientes a_i de la señal $x_a[n]$
Coeficientes: **a = [2, exp(-j*pi/2), 2*exp(j*i), 1, 2*exp(j*pi), expp(-j*pi/2)];** **a = [2,- 1j, -2,1, -2, - 1j] -> N=[0:1:5]**

Programa que calcula las señales temporales a partir de sus coeficientes y las representa

```
clear
clc

%Calculo de FT

n = [0:5];
a = [2, exp(-j*pi/2), 2*exp(j*pi), 1, 2*exp(j*pi), exp(-j*pi/2)]
N=6;
x = N*ifft(a);
M = mean(x);

%Graficos de señal original
subplot(1,3,1);
S = stem(n,x)
S.LineWidth = 1.5;
S.Marker = 'diamond';
S.MarkerFaceColor = 'red';
title("N*ifft(a)")
xlabel("n")
ylabel("x[n]")
grid on

subplot(1,3,2);
S = stem(n,real(x))
S.LineWidth = 1.5;
S.Marker = 'diamond';
S.MarkerFaceColor = 'red';
title("Re(x[n])")
xlabel("n")
ylabel("x[n]")
grid on

subplot(1,3,3);
S = stem(n,imag(x))
S.LineWidth = 1.5;
S.Marker = 'diamond';
S.MarkerFaceColor = 'red';
title("Im(x[n])")
```

```
xlabel("n")
ylabel("x[n]")
grid on




%Graficos de coeficientes
subplot(1,3,1);
S = stem(n,a)
S.LineWidth = 1.5;
S.Marker = 'diamond';
S.MarkerFaceColor = 'red';
title("a[n]")
xlabel("n")
ylabel("a[n]")
grid on

subplot(1,3,2);
S = stem(n,real(a))
S.LineWidth = 1.5;
S.Marker = 'diamond';
S.MarkerFaceColor = 'red';
title("Re(x[n])")
xlabel("n")
ylabel("x[n]")
grid on

subplot(1,3,3);
S = stem(n,imag(a))
S.LineWidth = 1.5;
S.Marker = 'diamond';
S.MarkerFaceColor = 'red';
title("Im(x[n])")
xlabel("n")
ylabel("x[n]")
grid on
```

Representación de señal y sus coeficientes con subplot() en 1 fila y 3 columnas

Gráfico de coeficientes:

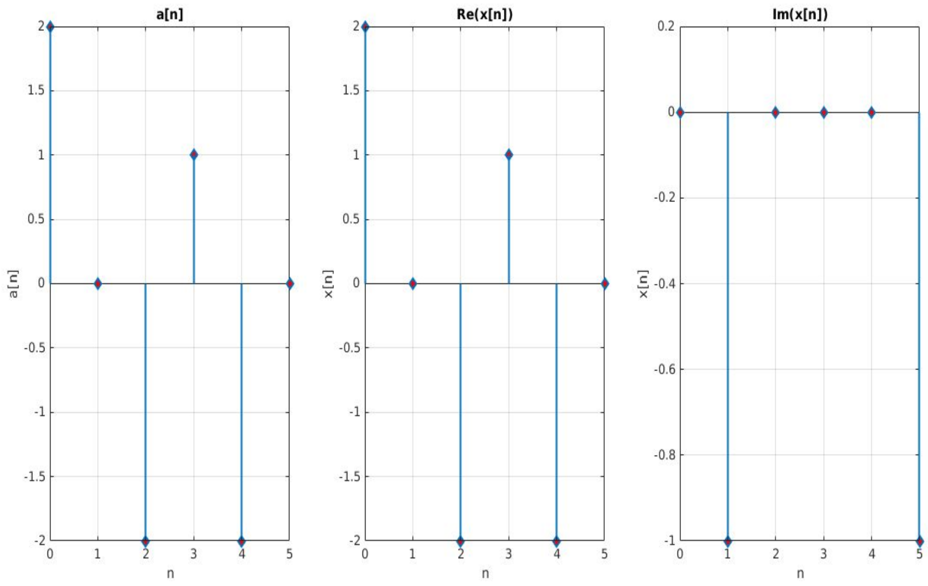
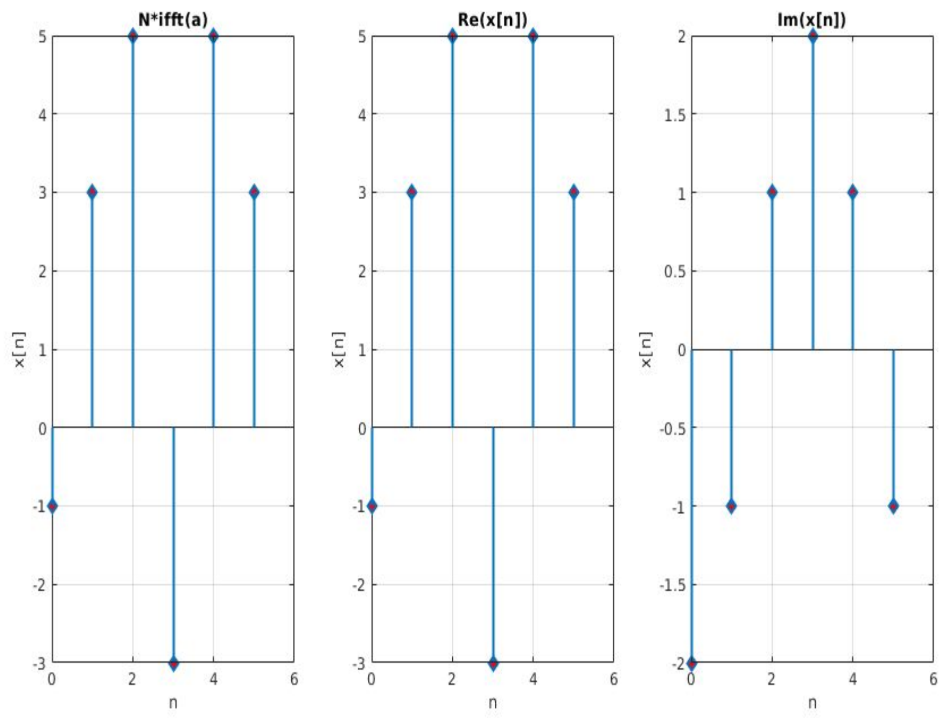
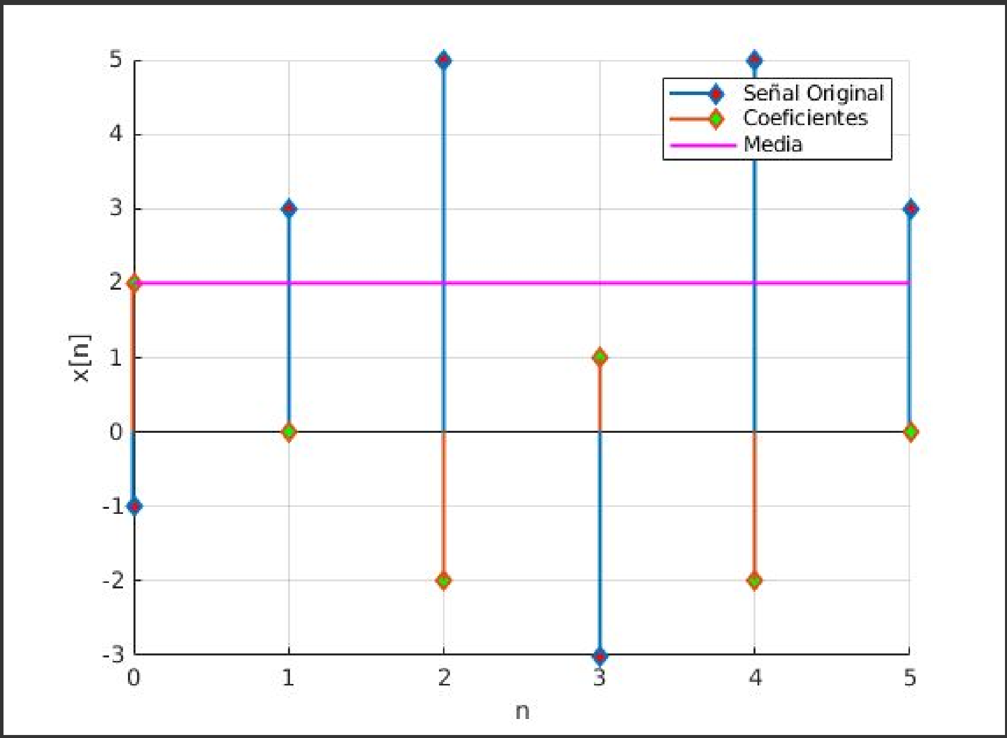


Gráfico de señal original:



Programa que comprueba si el valor medio de la señal media en el periodo corresponde al primer coeficiente. Indique cuál es el valor medio obtenido. Si no se cumple indique por qué.

```
hold on;
S = stem(n,x);
S.LineWidth = 1.5;
S.Marker = 'diamond';
S.MarkerFaceColor = 'red';
T = stem(n,a);
T.LineWidth = 1.5;
T.Marker = 'diamond';
T.MarkerFaceColor = 'green';
U = plot([0 5],[M M]);
U.LineWidth = 1.5
U.Color = 'magenta';
xlabel("n");
ylabel("x[n]");
legend("Señal Original", "Coeficientes", "Media")
grid on;
```



Programa que comprueba la relación de Parseval. Indique la potencia obtenida. Si no se cumple indiqué por qué

```
%Parseval
E1 = (1/N).*(x.^2);
E1 = sum(E1)
E2 = (a.^2);
E2 = sum(E2)
%Voltios al cuadrado

Resultado:
E1 = 11.0000 - 0.0000i
E2 = 11.0000 - 0.0000i
```

Tarea 3: Aproximación numérica de la FT

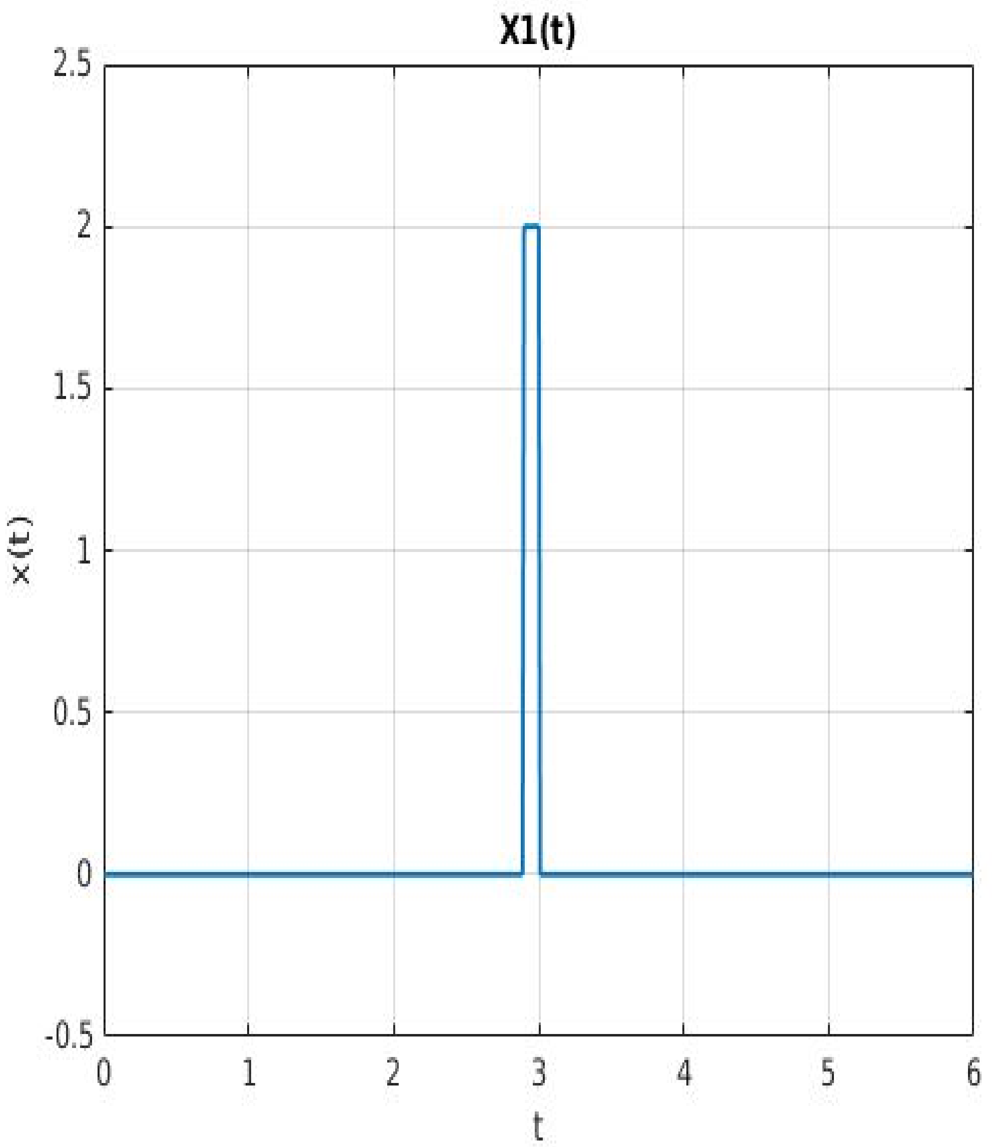
Rellene la siguiente tabla con la respuesta a las preguntas de esta tarea, y entregue un fichero `tarea3.m` con los comandos usados. Incluya todos los comandos y sus parámetros, no haga un resumen de los comandos.

Programa que representa $x_1(t)$ en el intervalo T=6 con $0 \leq t<6$ utilizando un periodo de muestreo de $\Delta t=t_s$=0.01 (`inct`)

```
clear
clc

%Aproximacion numerica de la FT
inct = 0.01;
t = [0:inct:6];
fs = 1/inct;
T = 6;
N = T/inct;

x0 = 0;
x1 = 2;
x = x0.*(t<2.9)+x1.*((t>=2.9)&(t<=3))+x0.*(t>3);
```

```
S = plot(t,x)
grid on;
axis([0 6 -0.5 2.5])
S.LineWidth = 1.5;
title("X1(t)")
xlabel("t")
ylabel("x(t)")
grid on
```

Gráfica de $x_1(t)$



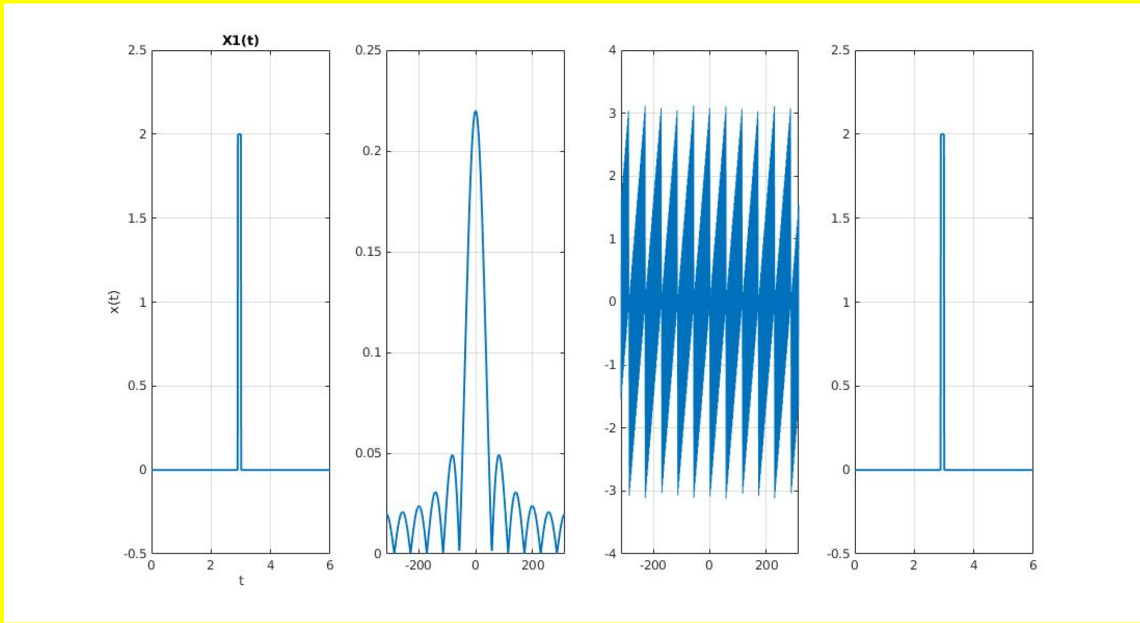
Programa que calcula numéricamente su FT muestreada $X(jw_m)$ y representa la señal en el tiempo, el módulo de la FT y la fase de la FT

```
X1 = fft(x)/fs;
wn = 2*pi*(-fs/2:fs/N:fs/2);

incw = 2*pi*fs/N;

mod = abs(fftshift(X1));
fase = angle(fftshift(X1));
```

Programa que reconstruye la señal temporal y la pinta en una cuarta columna con subplot

```
x0 = ifft(X1)/inct;
wn = (wn+fs*3);
wn = wn*T/length(wn);


% En este caso dado que la funcion estaba centrada en cero  con
la transformada de fourier, y muestreada en n muestras, se
realizó un corrimiento y la suma de la frecuencia de muestreo
multiplicada para retornar al lugar original.
```

Gráfica resultante usando `subplot` (1 filas x 4 columnas)



Tarea 4: Análisis de la FT numérica

Rellene la siguiente tabla con la respuesta a las preguntas de esta tarea, y entregue un fichero `tarea4.m` con los comandos usados.

Programa que usa `mean` para obtener el valor medio de la señal $x_1(t)$

```
clear;
clc;

T=6;

inct=0.01;
t=0:inct:T;
fs=1/inct;
```

```
N=T/inct; % Definimos el número de muestras
x = 0.*(t<2.9)+2.*((t>=2.9)&(t<=3))+0.*(t>3);

media = mean(x)


xf = (1/N).*fft(x);
xf(1)
```

¿Qué relación existe entre el valor medio y el primer coeficiente de la DFT c_0? Indique los comandos que calculan estos valores.

Usa la ecuación de análisis de la DFT:

$$X[m] = DFT\{x[n]\} = \sum_{n=0}^{N-1} x[n]e^{-j\frac{2\pi}{N}mn} \quad \text{con} \quad m = 0, 1, .., N-1$$

Comandos: Mean, xf(1), sum(x)
donde xf es el vector que devuelve la fft(x)

Valores retornados:
fft = xf =0.0367
media = 0.0366

Programa que calcula numéricamente la energía de la señal $x_1(t)$, así como su energía en el dominio de la frecuencia $X_1(jw)$.

```
% Potencia
P = trapz(wn, abs(fft(x)/fs).^2) / (2*pi);
% Energia
E = trapz(t, abs(x).^2);

disp(P)
disp(e)

P = 0.4327
E = 0.4400
```

Indique si se cumple la relación de Parseval. Si no se cumple indique por qué

Si cumple, dado que los datos obtenidos de potencia y energía son muy cercanos. Idealmente deberían ser iguales según la teoría, pero al tener diferentes métodos para llegar al mismo resultado, siempre se tienen incertezas.

Programa que estudia si se cumple que $X(jw) = X^*(-jw)$

```
clear;
clc;

T=6;

inct=0.01;
t=0:inct:T;
fs=1/inct;
N=T/inct; % Definimos el número de muestras
wn=2*pi*(-fs/2:fs/N:fs/2);
x = 0.*(t<2.9)+2.*((t>=2.9)&(t<=3))+0.*(t>3);

X1 = fft(x)/fs
X1_recortada = X1(2:601);
X1_flip = fliplr(X1_recortada);
X1_conj = conj(X1_flip);
sum(X1_recortada-X1_conj)
```

En vista de los resultados del programa anterior, ¿Se cumple que $X(jw) = X^*(-jw)$ **?**

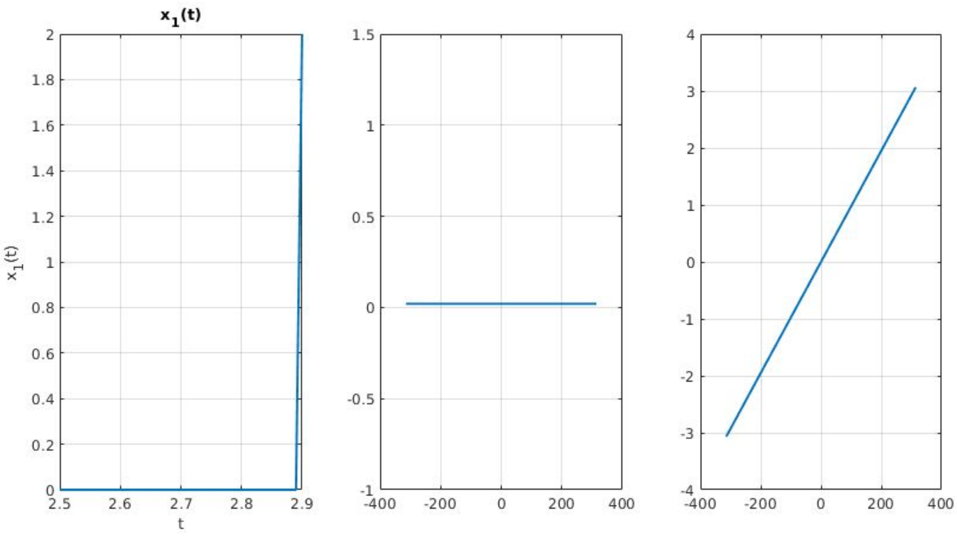
Si no se cumple indique por qué

Se cumple, dado que al ejecutar el programa anterior da como resultado 0, lo que hace que se cumpla la función X(jw) = X(-jw)

Programa anterior modificado para que calcule la señal $x_1(t)$, su módulo y su fase en un intervalo más pequeño T=0.5 con $2.5 \leq t < 3$

```
clear;
T=2.9-2.5;

inct=0.01;
t=2.5:inct:2.9;
fs=1/inct;
N=T/inct; % Definimos el número de muestras
x= 0.*(t<2.9)+2.*((t>=2.9)&(t<=3))+0.*(t>3);

X1=fft(x)/fs;
wn=2*pi*(-fs/2:fs/N:fs/2);
incw=2*pi*fs/N;

subplot(1,3,1);
S = plot(t,x)
xlabel('t');
ylabel('x_1(t)');
title('x_1(t)');
S.LineWidth = 1.5;
grid on

subplot(1,3,2);
S =plot(wn,abs(fftshift(X1)));
S.LineWidth = 1.5;
grid on

subplot(1,3,3);
S = plot(wn,angle(fftshift(X1)));
S.LineWidth = 1.5;
grid on
```

Gráfica resultante con `subplot` (1 filas x 3 columnas)



$x_1(t)$
$x_1(t)$
t
2
1.8
1.6
1.4
1.2
1
0.8
0.6
0.4
0.2
0
2.5
2.6
2.7
2.8
2.9
1.5
1
0.5
0
-0.5
-1
-400
-200
0
200
400
4
3
2
1
0
-1
-2
-3
-4
-400
-200
0
200
400