Práctica 3 de Señales y Sistemas

Señales y sistemas continuos en el dominio de la frecuencia

Francisco Javier Mercader Martínez
Rubén Gil Martínez

1. Señales periódicas continuas. Series de Fourier

Las ecuaciones que relacionan una señal periódica continua con su desarrollo en series de Fourier son

$$a_k = \frac{1}{T}\int_T x(t)e^{-jk\frac{2\pi}{T}t}\,\mathrm{d}t$$
$$x(t) = \sum_{k=-\infty}^{+\infty} a_k e^{jk\frac{2\pi}{T}t}$$

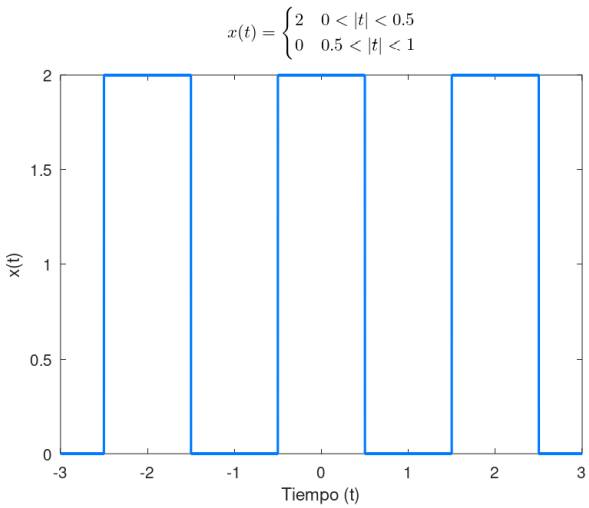
En este ejercicio se implementarán las dos ecuaciones mediante funciones en MATLAB. Para ello es necesario simular la señal continua $x(t)$, discretizándola mediante un incremento de tiempo, Δt, dando lugar a la señal continua discretizada $x(k\Delta t)$.

1. Genere en MATLAB las siguientes señales periódicas ($T = 2$), generando 3 periodos de cada una de ellas.

```
% Definir el periodo y el paso de tiempo
T = 2; dt = 0.001;
% Crear el rango de tiempo para un periodo
t = -T/2:dt:T/2-dt;
```

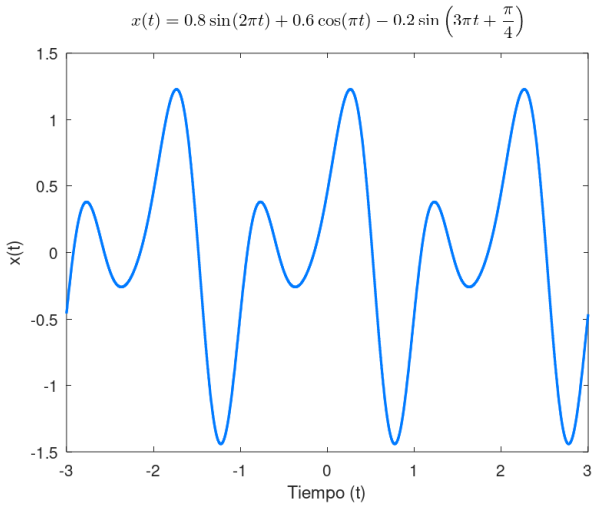
a) $x(t) = \begin{cases} 2, & 0 < |t| < 0.5 \\ 0, & 0.5 < |t| < 1 \end{cases}$

```
% a)
x = zeros(1, length(t));
ti = find(abs(t)<=0.5); x(ti) = 2;
ti = find(abs(t)> 0.5); x(ti) = 0;
figure(1), plot([t-T t t+T], [x x x], "LineWidth", 1.5, "color", "#007AFF");
xlabel('Tiempo (t)');
ylabel('x(t)');
title('$x(t)=\begin{cases}2 & 0<|t|<0.5\\0&0.5<|t|<1\end{cases}$','Interpreter','latex');
```



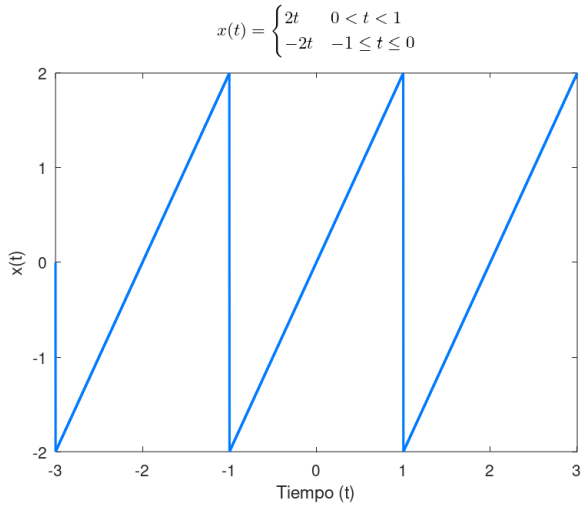
b) $x(t) = 0.8\sin(2\pi t) + 0.6\cos(\pi t) - 0.2\sin\left(3\pi t + \frac{\pi}{4}\right)$

```
% b)
x = 0.8*sin(2*pi*t) + 0.6*cos(pi*t) - 0.2*sin(3*pi*t + pi/4);
figure(2), plot([t-T t t+T], [x x x], "LineWidth", 1.5, "color", "#007AFF");
xlabel('Tiempo (t)');
ylabel('x(t)');
title('$x(t)=0.8\sin(2\pi t)+0.6\cos(\pi t)-0.2\sin\left(3\pi t+\dfrac{\pi}{4}\right)$','
    Interpreter','latex');
```



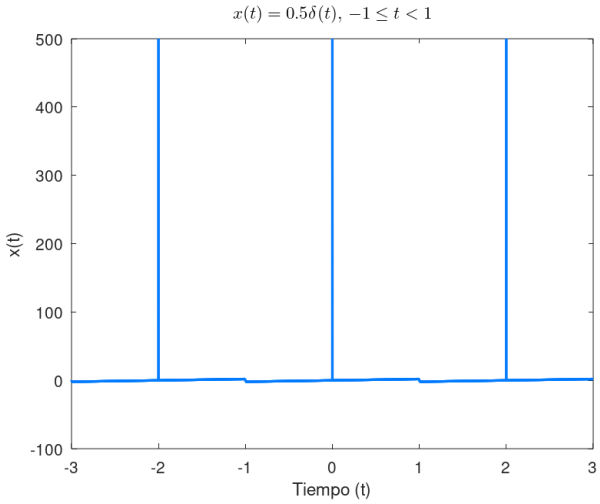
c) $x(t) = \begin{cases} 2t, & 0 < t < 1 \\ -2t, & -1 \le t \le 0 \end{cases}$

```
% (c)
x = zeros(1, length(t));
ti = find(abs(t)>=-1 & abs(t)<0); x(ti) = -2*t(ti);
ti = find(abs(t)>0 & abs(t)<1); x(ti) = 2*t(ti);
figure(3), plot([t-T t t+T], [x x x], "LineWidth", 1.5, "color", "#007AFF");
xlabel('Tiempo (t)');
ylabel('x(t)');
title('$x(t)=\begin{cases}2t & 0<t<1\\-2t&-1\le t\le0\end{cases}$','Interpreter','latex')
    ;
```



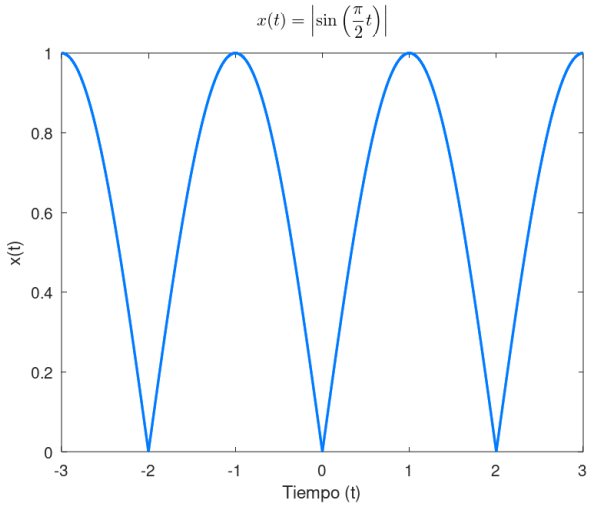
d) $x(t) = 0.5\delta(t),\ -1 \le t < 1$

```
% (d)
ti = find(abs(t)==0); x(ti) = 0.5 * (1/dt);
figure(4), plot([t-T t t+T], [x x x], "LineWidth", 1.5, "color", "#007AFF");
xlabel('Tiempo (t)');
ylabel('x(t)');
title('$x(t)=0.5\delta(t),\,-1\le t<1$', 'Interpreter', 'latex')
```



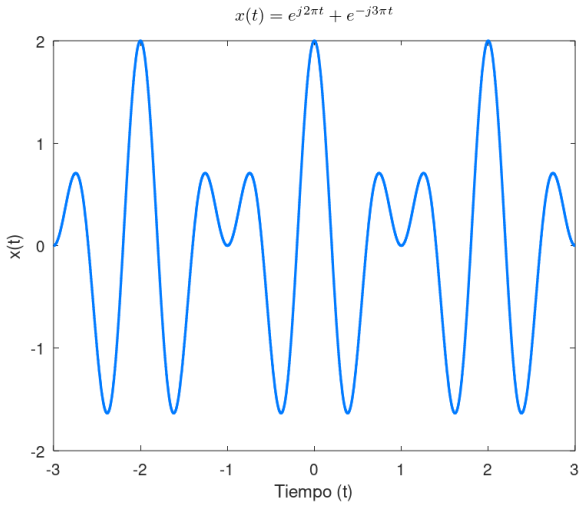
e) $x(t) = \left|\sin\left(\frac{\pi}{2}t\right)\right|$

```
% (e)
x = abs(sin(pi/2.*t));
figure(5), plot([t-T t t+T], [x x x], "LineWidth", 1.5, "color", "#007AFF");
xlabel('Tiempo (t)');
ylabel('x(t)');
title('$x(t)=\left|\sin\left(\dfrac{\pi}{2}t\right)\right|$', "Interpreter", "latex")
```



f) $x(t) = e^{j2\pi t} + e^{-j3\pi t}$

```
% (f)
x = exp(j*2*pi*t) + exp(-3*j*pi*t);
figure(6), plot([t-T t t+T], [x x x], "LineWidth", 1.5, "color", "#007AFF");
xlabel('Tiempo (t)');
ylabel('x(t)');
title('$x(t)=e^{j2\pi t}+e^{-j3\pi t}$', 'Interpreter', 'latex')
```



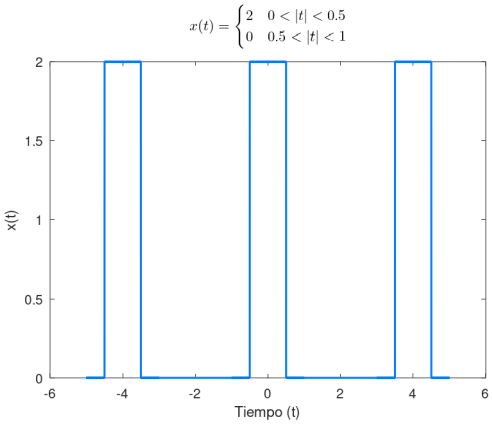
Ayuda: la señal (a) se obtiene mediante los comandos

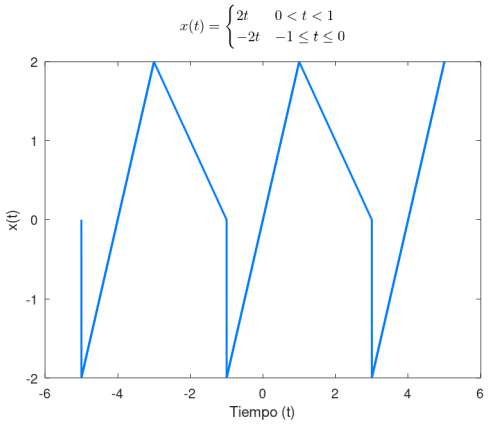
```
T = 2; dt = 0.001; t = -T/2:dt:T/2-dt;
x = zeros(1, length(t));
ti = find(abs(t)<=0.5); x(ti) = 2;
ti = find(abs(t)> 0.5); x(ti) = 0;
figure, plot([t-T t t+T], [x x x])
```

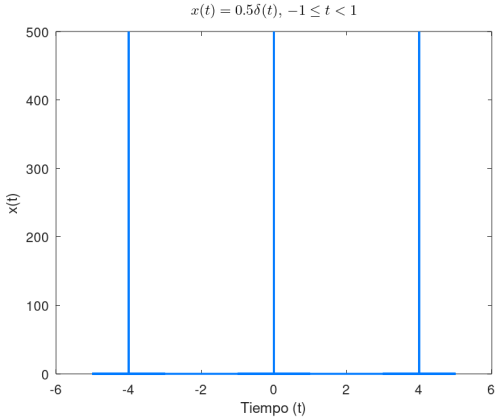
2. Repita el proceso para las señales a) c) y d), asumiendo un periodo T=4 y T=6. Razone los resultados obtenidos.

```
T = 4;dt = 0.001;
t = -T/2:dt:T/2-dt;
```

```
% (a)
x = zeros(1, length(t));
ti = find(abs(t)<=0.5); x(ti) = 2;
ti = find(abs(t)> 0.5); x(ti) = 0;
figure(7), plot([t-T t t+T], [x x x], "LineWidth", 1.5, "color", "#007AFF");
xlabel('Tiempo (t)');
ylabel('x(t)');
title('$x(t)=\begin{cases}2 & 0<|t|<0.5\\0&0.5<|t|<1\end{cases}$','Interpreter','latex');

% (c)
x = zeros(1, length(t));
ti = find(abs(t)>=-1 & abs(t)<0); x(ti) = -2*t(ti);
ti = find(abs(t)>0 & abs(t)<1); x(ti) = 2*t(ti);
figure(8), plot([t-T t t+T], [x x x], "LineWidth", 1.5, "color", "#007AFF");
xlabel('Tiempo (t)');
ylabel('x(t)');
title('$x(t)=\begin{cases}2t & 0<t<1\\-2t&-1\le t\le0\end{cases}$','Interpreter','latex');

% (d)
x = zeros(1, length(t));
ti = find(abs(t)==0); x(ti) = 0.5 * (1/dt);
figure(9), plot([t-T t t+T], [x x x], "LineWidth", 1.5, "color", "#007AFF");
xlabel('Tiempo (t)');
ylabel('x(t)');
title('$x(t)=0.5\delta(t),\,-1\le t<1$', 'Interpreter', 'latex')
```

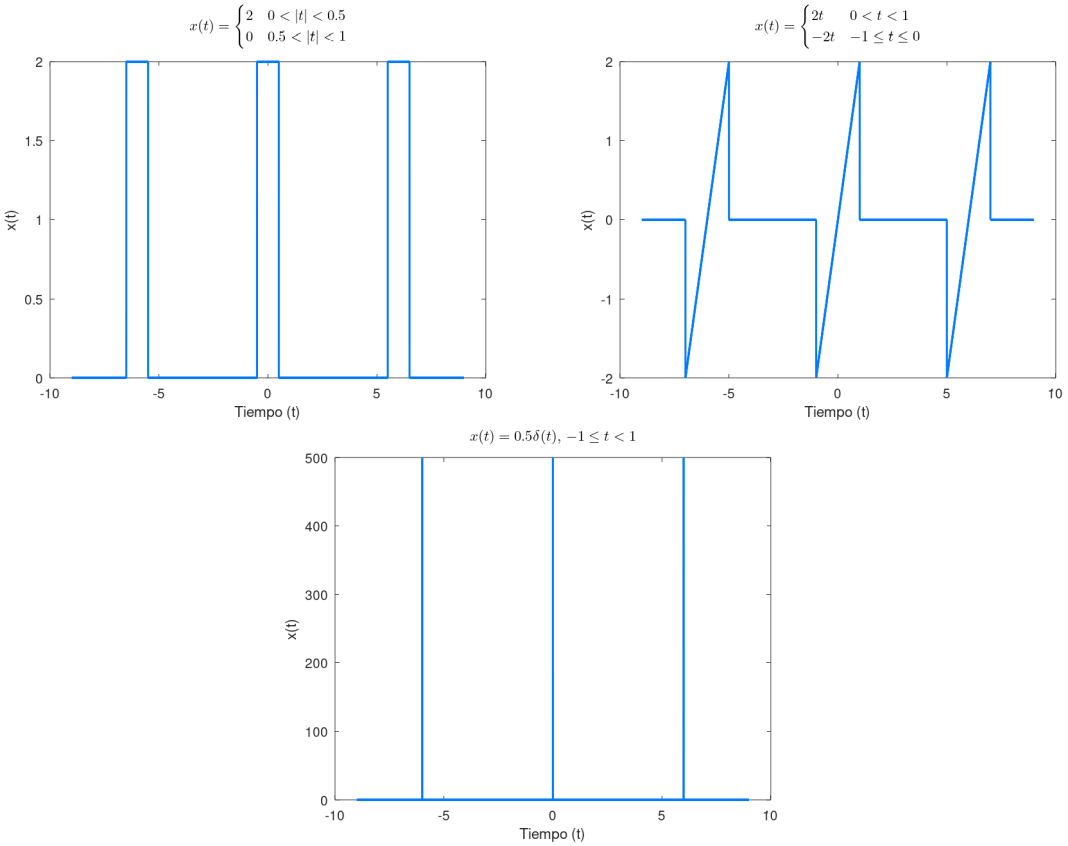






```
T = 6; dt = 0.001;
t = -T/2:dt:T/2-dt;
```

```
% (a)
x = zeros(1, length(t));
ti = find(abs(t)<=0.5); x(ti) = 2;
ti = find(abs(t)> 0.5); x(ti) = 0;
figure(10), plot([t-T t t+T], [x x x], "LineWidth", 1.5, "color", "#007AFF");
xlabel('Tiempo (t)');
ylabel('x(t)');
title('$x(t)=\begin{cases}2 & 0<|t|<0.5\\0&0.5<|t|<1\end{cases}$','Interpreter','latex');

% (c)
x = zeros(1, length(t));
ti = find(abs(t)>=-1 & abs(t)<0); x(ti) = -2*t(ti);
ti = find(abs(t)>0 & abs(t)<1); x(ti) = 2*t(ti);
figure(11), plot([t-T t t+T], [x x x], "LineWidth", 1.5, "color", "#007AFF");
xlabel('Tiempo (t)');
ylabel('x(t)');
title('$x(t)=\begin{cases}2t & 0<t<1\\-2t&-1\le t\le0\end{cases}$','Interpreter','latex');

% (d)
x = zeros(1, length(t));
ti = find(abs(t)==0); x(ti) = 0.5 * (1/dt);
figure(12), plot([t-T t t+T], [x x x], "LineWidth", 1.5, "color", "#007AFF");
xlabel('Tiempo (t)');
ylabel('x(t)');
title('$x(t)=0.5\delta(t),\,-1\le t<1$', 'Interpreter', 'latex')
```



Como se aumenta el periodo fundamental de las señales podemos observar que por la fórmula de su frecuencia angular $\left(\omega_0 = \frac{2\pi}{T}\right)$ tenemos que se expande en el tiempo porque cuanto mayor sea el periodo fundamental, la señal tardará más tiempo en repetirse.

3. Cree la siguiente función de MATLAB que obtiene los coeficientes de la serie de Fourier (DSF) de una señal periódica:

```
function ak = cfourier(x,T,N,dt)
t = -T/2:dt:T/2-dt;
ak = zeros(1,2*N+1);
for k = -N:N
    ak(k+N+1) = 1/T*sum(x.*exp(-j*k*2*pi/T*t))*dt;
end
```

donde **x** es una señal continua discretizada como las generadas anteriormente, **T** es el periodo de la señal, **N** es el orden del armónico más alto y **dt** es el paso temporal de discretación. El DSF queda almacenado en **ak** con el siguiente orden:

$$[a_{-N}a_{-N+1}a_{-N+2}\dots a_{-1}a_0a_1\dots a_{N-2}a_{N-1}a_N]$$

4. **Como estudio previo calcule analíticamente los desarrollos en series de Fourier de las señales del punto 1** y corrobore los resultados numéricos obtenidos con la función **cfourier**.

```
% Definir el periodo y el paso de tiempo
T = 2; dt = 0.001; N=5;
t = -T/2:dt:T/2-dt;
```

```
% (a)
x = zeros(1, length(t));
ti = find(abs(t)<=0.5); x(ti) = 2;
ti = find(abs(t)> 0.5); x(ti) = 0;
ak_a = cfourier(x, T, N, dt);
```

```
ak_a =
   0.1273 + 0.0000i   0.0010 - 0.0000i  -0.2122 - 0.0000i  -0.0010 + 0.0000i   0.6366 - 0.0000i
   1.0010 +      0i   0.6366 + 0.0000i  -0.0010 - 0.0000i  -0.2122 + 0.0000i   0.0010 + 0.0000i
   0.1273 - 0.0000i
```

```
% (b)
x = 0.8*sin(2*pi*t) + 0.6*cos(pi*t) - 0.2*sin(3*pi*t + pi/4);
ak_b = cfourier(x, T, N, dt);
```

```
ak_b =
   0.0000 - 0.0000i  -0.0000 - 0.0000i  -0.0707 - 0.0707i  -0.0000 + 0.4000i   0.3000 + 0.0000i
  -0.0000 +      0i   0.3000 - 0.0000i  -0.0000 - 0.4000i  -0.0707 + 0.0707i  -0.0000 + 0.0000i
   0.0000 + 0.0000i
```

```
% (c)
x = zeros(1, length(t));
ti = find(abs(t)>=-1 & abs(t)<0); x(ti) = -2*t(ti);
ti = find(abs(t)>0 & abs(t)<1); x(ti) = 2*t(ti);
ak_c = cfourier(x, T, N, dt);
```

```
ak_c =
   -0.0162 + 0.0000i   0.0000 + 0.0000i  -0.0450 + 0.0000i   0.0000 + 0.0000i  -0.4053 + 0.0000i
    1.0000 +      0i  -0.4053 - 0.0000i   0.0000 - 0.0000i  -0.0450 - 0.0000i   0.0000 - 0.0000i
   -0.0162 - 0.0000i
```

```
% (d)
x = zeros(1, length(t));
ti = find(abs(t)==0); x(ti) = 0.5 * (1/dt);
ak_d = cfourier(x, T, N, dt);
```

```
ak_d =
   0.2500   0.2500   0.2500   0.2500   0.2500   0.2500   0.2500   0.2500   0.2500   0.2500   0.2500
```

```
% (e)
x = abs(sin(pi/2.*t));
ak_e = cfourier(x, T, N, dt);
```

```
ak_e =
  -0.0064 - 0.0000i  -0.0101 - 0.0000i  -0.0182 + 0.0000i  -0.0424 - 0.0000i  -0.2122 + 0.0000i
   0.6366 +      0i  -0.2122 - 0.0000i  -0.0424 + 0.0000i  -0.0182 - 0.0000i  -0.0101 + 0.0000i
  -0.0064 + 0.0000i
```

```
% (f)
x = exp(j*2*pi*t) + exp(-3*j*pi*t);
ak_f = cfourier(x, T, N, dt);
```

```
ak_f =
   -2.3575e-16 - 7.3283e-17i   3.4063e-16 + 5.8993e-17i   1.0000e+00 - 5.6710e-17i
   -3.3348e-16 + 1.3077e-16i  -3.4729e-16 + 4.8116e-17i  -3.7292e-16 - 3.4807e-17i
   -3.3980e-16 - 1.8739e-16i   1.0000e+00 + 5.8039e-17i   3.0022e-16 + 1.5665e-18i
   -2.5388e-16 + 9.3413e-17i   3.3479e-17 + 5.4260e-17i
```

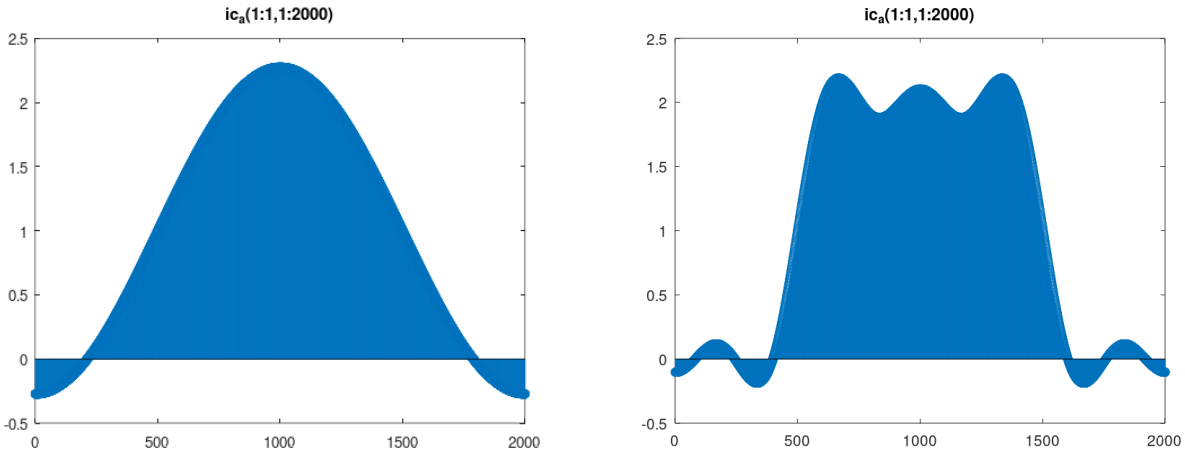
5. Cree la siguiente función de MATLAB, que calcula la señal periódica a partir de su desarrollo en series de Fourier:

```
function x = icfourier(ak,T,N,dt)
t = -T/2:dt:T/2-dt;
x = zeros(1,lenght(t));
for k = -N:N
    x = x + ak(k+N+1)*exp(j*k*2*pi/T*t);
end
```

6. Reconstruya a partir de los desarrollos mediante series de Fourier **ak** las señales a) c) d) y e) variando el máximo número de armónicos N en la reconstrucción. Comente los resultados obtenidos.

```
% (a)
ic_a = icfourier(ak_a, T, N, dt);
% (c)
ic_c = icfourier(ak_c, T, N, dt);
% (d)
ic_d = icfourier(ak_d, T, N, dt);
% (e)
ic_e = icfourier(ak_e, T, N, dt);
```

A continuación van a aparecer dos variaciones de los valores obtenidos del apartado (a) utilizando `N=1` y `N=5`, en ese orden:



Al aumentar el valor de `N` habrá más coeficientes y se obtendrá más información de la señal porque al reconstruirla usamos más coeficientes.

7. Verifique el cumplimiento del teorema de Parseval comparando los resultados **teóricos** y prácticos

$$\sum_{k=-\infty}^{\infty} |a_k|^2 = \frac{1}{T} \int_{<T>} |x(t)|^2 \, \mathrm{d}t$$

```
% Calcular la suma de los cuadrados de los coeficientes de Fourier
sum_ak2_a = sum(abs(ak_a).^2); % sum_ak2_a = 1.9351
sum_ak2_b = sum(abs(ak_b).^2); % sum_ak2_b = 0.5200
sum_ak2_c = sum(abs(ak_c).^2); % sum_ak2_c = 1.3331
sum_ak2_d = sum(abs(ak_d).^2); % sum_ak2_d = 0.6875
sum_ak2_e = sum(abs(ak_e).^2); % sum_ak2_e = 0.4999
sum_ak2_f = sum(abs(ak_f).^2); % sum_ak2_f = 2.0000
```

2. Transformada de Fourier

Las ecuaciones que relacionan una sñela con su transformada de Fourier son las siguientes:

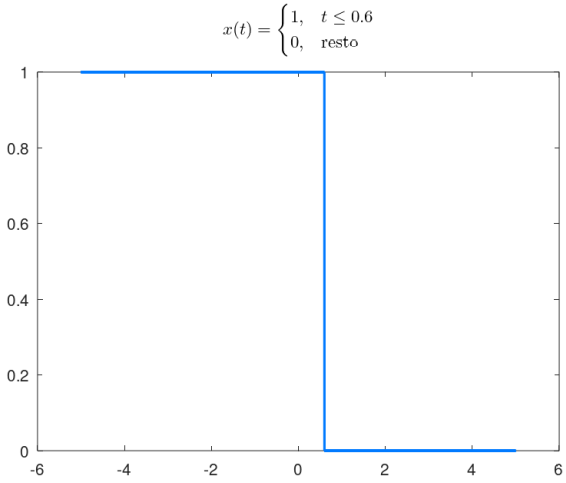
$$X(\omega) = \int_{-\infty}^{\infty} x(t) e^{-j\omega t} \, \mathrm{d}t \qquad x(t) = \frac{1}{2\pi} \int_{-\infty}^{+\infty} X(\omega) e^{j\omega t} \mathrm{d}\omega$$

Cuestiones

1. Genere en MATLAB las siguientes señales aperiódicas, tomando un intervalo de discretación de 0.002, y considerando $-5 \leq t \leq 5$.

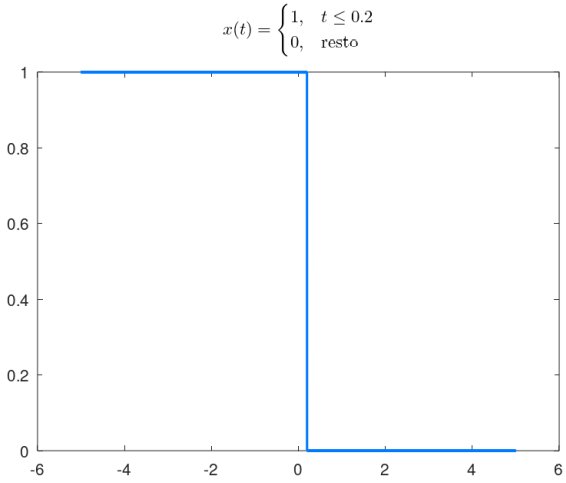
   a) $x(t) = \begin{cases} 1, & t \leq 0.6 \\ 0, & \text{resto} \end{cases}$

```
% a)
t = -5:0.002:5;
x = zeros(size(t));
x(t <= 0.6) = 1;
figure(13); plot(t, x, "LineWidth", 1.5, "color", "#007AFF");
title(' $x(t)=\begin{cases}1, & t\le0.6\\0,&\text{resto}\end{cases}$', "Interpreter", "
    latex");
```



b) $x(t) = \begin{cases} 1, & t \leq 0.2 \\ 0, & \text{resto} \end{cases}$

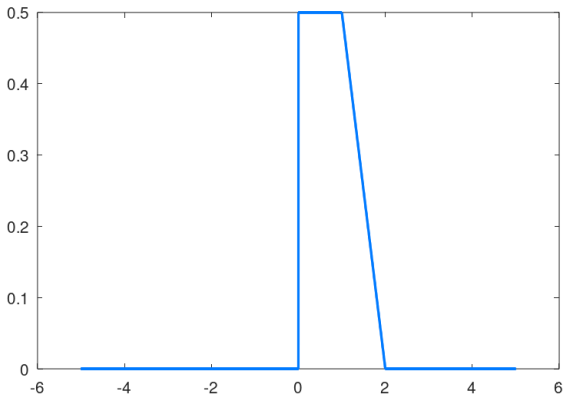
```
% b)
t = -5:0.002:5;
x = zeros(size(t));
x(t <= 0.2) = 1;
figure(14); plot(t, x, "LineWidth", 1.5, "color", "#007AFF");
title('$x(t)=\begin{cases}1, & t\le0.2\\0, & \mathrm{resto}\end{cases}$', 'Interpreter',
    'latex');
```



c) $x(t) = \begin{cases} 0.5, & 0 < t \leq 1 \\ 1 - 0.5t, & 1 < t \leq 2 \\ 0, & \text{resto} \end{cases}$

```
% c)
t = -5:0.002:5;
x = zeros(size(t));
x(0 < t & t <= 1) = 0.5;
x(1 < t & t <= 2) = 1 - 0.5 * t(1 < t & t <= 2);
figure(15); plot(t, x, "LineWidth", 1.5, "color", "#007AFF");
title('$x(t)=\begin{cases}0.5, & 0<t\le1\\1-0.5t,&1<t\le2\\0,& \mathrm{resto}\end{cases}$
    ', "Interpreter", "latex");
```

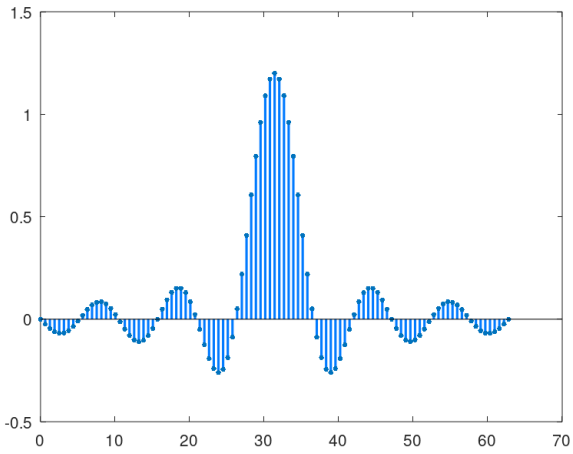
$$x(t) = \begin{cases} 0.5, & 0 < t \leq 1 \\ 1 - 0.5t, & 1 < t \leq 2 \\ 0, & \text{resto} \end{cases}$$



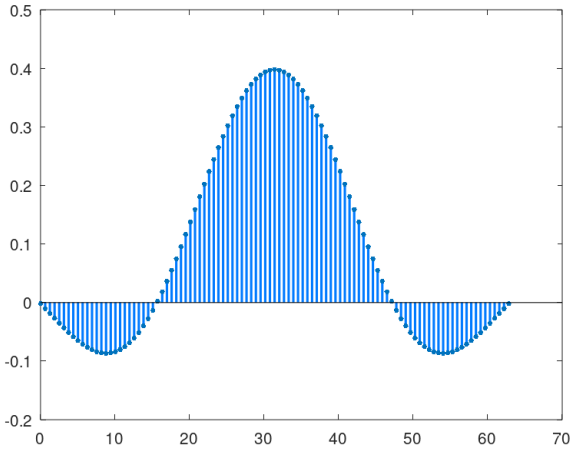
2. Suponiendo que las señales del punto 1 representaban el periodo de una señal periódica,

   i) Calcule su desarrollo en series de Fourier mediante **`ak=cfourier(x,T,N,dt)`**, y normalice por T (**`ak=T*ak`**). Al estar cada armónico está asociado a la frecuencia $k\frac{2\pi}{T}$, represente las series de Fourier normalizadas en el eje de frecuencias. Para ello emplee **`stem(w,ak,'.')`** y previamente defina el vector de frecuencias **`w=2*π*k/T`**.

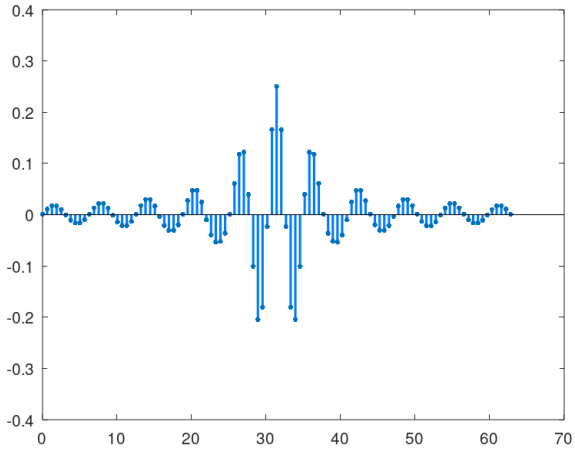
```
% a)
x = zeros(size(t));
x(t <= 0.6) = 1;
T = length(x)*0.002;
N = 50;
dt = 0.002;
ak = cfourier(x, T, N, dt);
ak = T * ak;
k = 0:length(ak)-1;
w = 2*pi*k/T;
figure(16); stem(w, ak, '.', "LineWidth", 1.5, "color", "#007AFF", "MarkerSize", 10);
```




```
% b)
x = zeros(size(t));
x(t <= 0.2) = 1;
T = length(x)*0.002;
N = 50;
dt = 0.002;
ak = cfourier(x, T, N, dt);
ak = T * ak;
k = 0:length(ak)-1;
w = 2*pi*k/T;
figure(17); stem(w, ak, '.', "LineWidth", 1.5, "color", "#007AFF", "MarkerSize", 10);
```



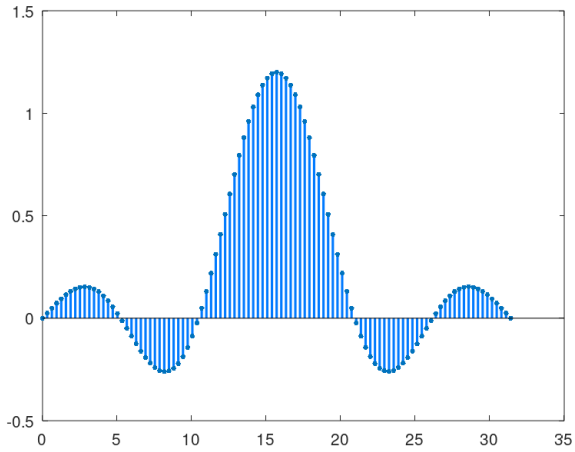

```
% c)
t = -5:0.002:5;
x = zeros(size(t));
x(0 < t & t <= 1) = 0.5;
x(1 < t & t <= 2) = 1 - 0.5 * t(1 < t & t <= 2);
T = length(x)*0.002;
N = 50;
dt = 0.002;
ak = cfourier(x, T, N, dt);
ak = T * ak;
k = 0:length(ak)-1;
w = 2*pi*k/T;
figure(18); stem(w, ak, '.', "LineWidth", 1.5, "color", "#007AFF", "MarkerSize", 10);
```



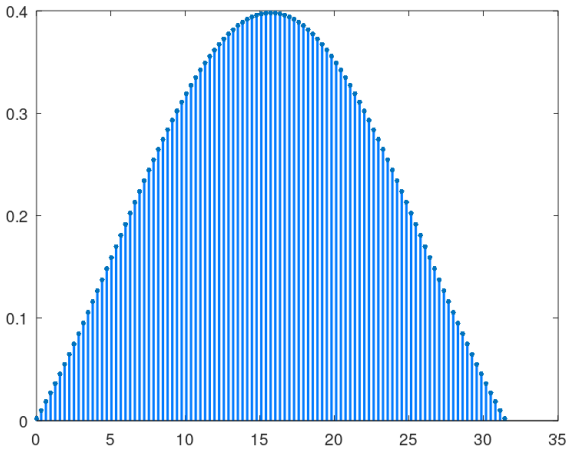
ii) Incremente la longitud del vector de la señal hacia ambos lados de la siguiente manera:

```
Ti=floor(length(x)/2);
x=[zeros(1,Ti) x zeros(1,Ti)];
```

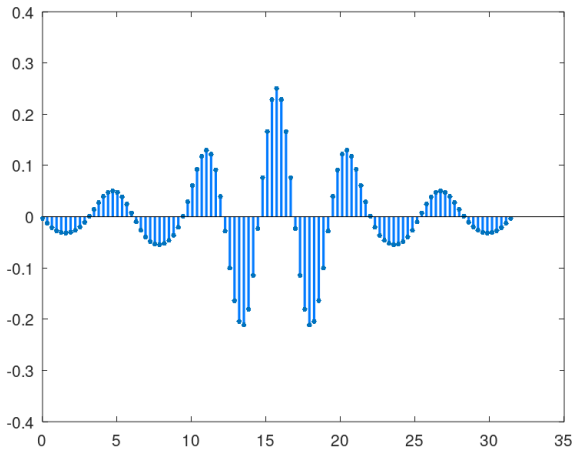
```
% a)
x = zeros(size(t));
x(t <= 0.6) = 1;
Ti = floor(length(x)/2);
x = [zeros(1, Ti) x zeros(1, Ti)];
T = length(x)*dt;
N = 50;
dt = 0.002;
ak = cfourier(x, T, N, dt);
ak = T * ak;
k = 0:length(ak)-1;
w = 2*pi*k/T;
figure(19); stem(w, ak, '.', "LineWidth", 1.5, "color", "#007AFF", "MarkerSize", 10);
```



```
% b)
x = zeros(size(t));
x(t <= 0.2) = 1;
Ti = floor(length(x)/2);
x = [zeros(1, Ti) x zeros(1, Ti)];
T = length(x)*dt;
N = 50;
dt = 0.002;
ak = cfourier(x, T, N, dt);
ak = T * ak;
k = 0:length(ak)-1;
w = 2*pi*k/T;
figure(20); stem(w, ak, '.', "LineWidth", 1.5, "color", "#007AFF", "MarkerSize", 10);
```




```
% c)
x = zeros(size(t));
x(0 < t & t <= 1) = 0.5;
x(1 < t & t <= 2) = 1 - 0.5 * t(1 < t & t <= 2);
Ti = floor(length(x)/2);
x = [zeros(1, Ti) x zeros(1, Ti)];
T = length(x)*dt;
N = 50;
dt = 0.002;
ak = cfourier(x, T, N, dt);
ak = T * ak;
k = 0:length(ak)-1;
w = 2*pi*k/T;
figure(21); stem(w, ak, '.', "LineWidth", 1.5, "color", "#007AFF", "MarkerSize", 10);
```

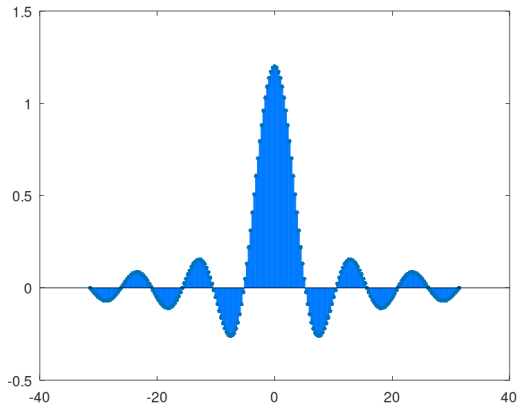


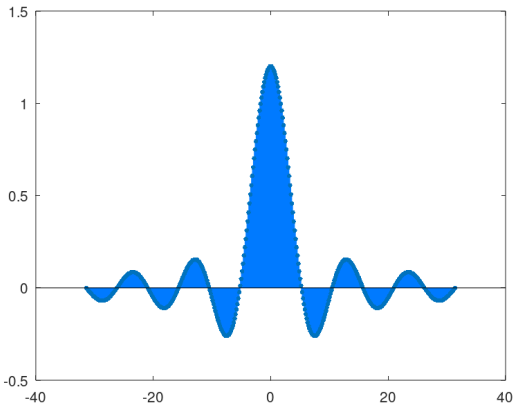
iii) Vuelva al paso **i** para obtener las nuevas series de Fourier normalizadas. Considere que:

- El orden del máximo armónico a tener en cuenta es el doble que el previo (**N=2*N**)
- El nuevo periodo obtenido es el doble del previo (**T=2*T**)

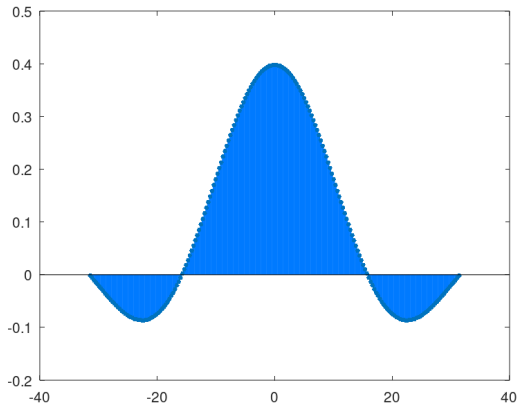
Repita este proceso (puede emplear un programa con un bucle **for**) y explique los resultados.

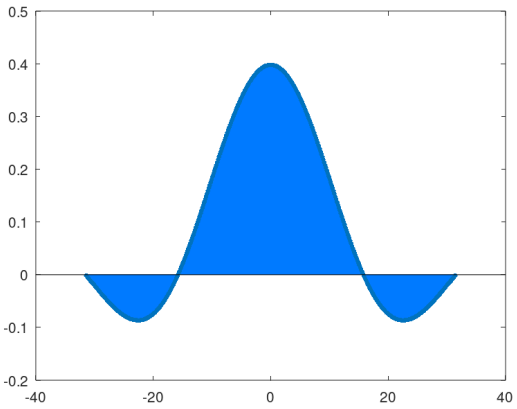
```
% a)
T = 10;
t = -T/2:0.002:T/2-0.002;
x = zeros(size(t));
ti = find(abs(t) <= 0.6)
x(ti) = 1;
N = 50;
dt = 0.002;
% Bucle for para duplicar N y T
for i = 1:2
    Ti = floor(length(x)/2);
    x = [zeros(1, Ti) x zeros(1, Ti)];
    N = 2*N;
    T = 2*T;
    k = -N:N
    % Calcular ak y normalizar
    ak = cfourier(x, T, N, dt);
    ak = T * ak;

    % Definir k y w
    w = 2*pi*k/T;

    % Crear figura y representar las series de Fourier normalizadas
    figure; stem(w, ak, '.', "LineWidth", 1.5, "color", "#007AFF", "MarkerSize", 10);
end
```



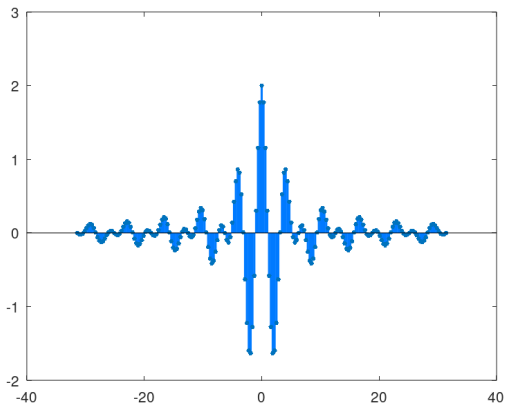


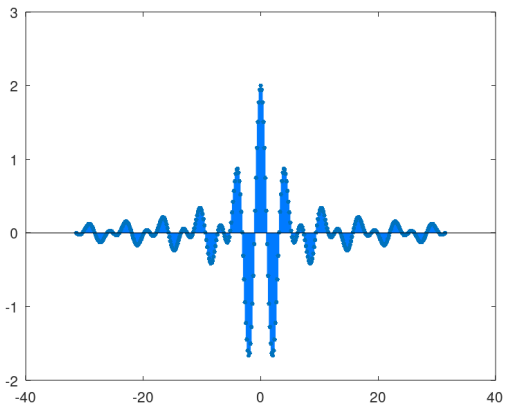

```
% b)
T = 10;
t = -T/2:0.002:T/2-0.002;
x = zeros(size(t));
ti = find(abs(t) <= 0.2)
x(ti) = 1;
N = 50;
dt = 0.002;
% Bucle for para duplicar N y T
for i = 1:2
    Ti = floor(length(x)/2);
    x = [zeros(1, Ti) x zeros(1, Ti)];
    N = 2*N;
    T = 2*T;
    k = -N:N
    % Calcular ak y normalizar
    ak = cfourier(x, T, N, dt);
    ak = T * ak;
    % Definir k y w
    w = 2*pi*k/T;
    % Crear figura y representar las series de Fourier normalizadas
    figure; stem(w, ak, '.', "LineWidth", 1.5, "color", "#007AFF", "MarkerSize", 10);
end
```





```
% c)
T = 10;
t = -T/2:0.002:T/2-0.002;
x = zeros(size(t));
ti = find(0 < abs(t) <= 0.1);
x(ti) = 0.5;
ti = find(1 < abs(t) & abs(t) <= 2);
x(ti) = 1 - 0.5 * t(ti);
N = 50;
dt = 0.002;
% Bucle for para duplicar N y T
for i = 1:2
    Ti = floor(length(x)/2);
    x = [zeros(1, Ti) x zeros(1, Ti)];
    N = 2*N;
    T = 2*T;
    k = -N:N
    % Calcular ak y normalizar
    ak = cfourier(x, T, N, dt);
    ak = T * ak;

    % Definir k y w
    w = 2*pi*k/T;

    % Crear figura y representar las series de Fourier normalizadas
    figure; stem(w, ak, '.', "LineWidth", 1.5, "color", "#007AFF", "MarkerSize", 10);
end
```





3. Implemente una función que obtenga la transformada de Fourier de una señal adaptando el programa de desarrollo en series de Fourier.

```
function [X,w] = tfourier(x,t,dw,wmax)
```

   - El intervalo de frecuencias a calcular está entre **-wmax** y **wmax** (wmax<<π/dt)
   - **dw** es la discretización en frecuencia en frecuencia
   - **w** es el vector con el eje frecuencia, su tamaño es el mismo que el de **x**.
   - **t** es el vector con el eje temporal, su tamaño es el mismo que el de **x**.

```
function [X,w] = tfourier(x,t,dw,wmax)
    % Crear el vector de frecuencias
    w = -wmax:dw:wmax;

    % Inicializar el vector de la Transformada de Fourier
    X = zeros(size(w));

    % Calcular la Transformada de Fourier para cada frecuencia
    for k = 1:length(w)
        X(k) = sum(x .* exp(-1i * w(k) * t));
    end
end
```

4. Empleando en la propiedad de la dualidad de la transformada de Fourier implemente una función que calcule la transformada inversa de Fourier haciendo uso de la función de la transformada directa.

```
function [X,w] = itfourier(x,t,dw,wmax)
```

```
function [X,t] = itfourier(x,w,dw,tmax)
    % Aplicar la Transformada de Fourier a la señal en el dominio de la frecuencia
    [x_tilde, t] = tfourier(x,w,dw,tmax);

    % Escalar la señal y tomar la parte real para obtener la señal en el dominio del tiempo
    X = real(dw/(2*pi) * x_tilde);
end
```

5. Verifique el correcto cumplimiento de las siguientes propiedades de la transformada de Fourier emplenado la función **tfourier**:

```
% Definir las señales de nuevo
t = -5:0.002:5;
a = zeros(size(t));
b = zeros(size(t));
c = zeros(size(t));
a(find(abs(t) <= 0.6)) = 1;
b(find(abs(t) <= 0.2)) = 1;
c(find(0 < abs(t) & abs(t) <= 0.1)) = 0.5;
c(find(1 < abs(t) <= 2)) = 1 - 0.5 * abs(t)(find(1 < abs(t) <= 2));
dt = 0.002
wmax = 5
% Calcular las transformadas de Fourier de las señales
[A, w] = tfourier(a, t,dt, wmax);
[B, ~] = tfourier(b, t,dt, wmax);
[C, ~] = tfourier(c, t,dt, wmax);
```

i) Propiedad de linealidad $X(\omega) + Y(\omega) \longleftrightarrow x(t) + y(t)$ (emplee las señales a y b).

```
% Propiedad de linealidad
AB = tfourier(a + b, t,dt, wmax);
assert(isequal(AB, A + B), 'No se cumple la propiedad');
```

El resultado muestra que la propiedad no se cumple

ii) Desplazamiento en el tiempo $e^{-j\omega t_0}X(\omega) \longleftrightarrow x(t-t_0)$ (emplee señal b).

```
% Propiedad de desplazamiento en el tiempo
t0 = 0.5;
B_shift = tfourier(b .* exp(-1i * w * t0), t, dt, wmax);
assert(isequal(B_shift, B .* exp(-1i * w * t0)), 'No se cumple la propiedad');
```

El resultado muestra que la propiedad no se cumple

iii) Inversión $X(-\omega) \longleftrightarrow x(-t)$ (emplee señal c).

```
% Propiedad de inversión
C_inv = tfourier(fliplr(c), t, dt, wmax);
assert(isequal(C_inv, fliplr(C)), 'No se cumple la propiedad');
```

El resultado muestra que la propiedad no se cumple

iv) Conjugación $X(\omega)^* \longleftrightarrow x(-t)^*$ (emplee señal c).

```
% Propiedad de conjugación
C_conj = tfourier(conj(c), t, dt, wmax);
assert(isequal(C_conj, conj(C)), 'No se cumple la propiedad');
```

El resultado muestra que la propiedad no se cumple

v) Real$\{X(\omega)\} \longleftrightarrow$ Parte real$\{x(t)\}$ (emplee señal c).

```
% Propiedad de parte real
C_real = tfourier(real(c), t, dt, wmax);
assert(isequal(C_real, real(C)), 'No se cumple la propiedad');
```

El resultado muestra que la propiedad no se cumple

vi) Imag$\{X(\omega)\} \longleftrightarrow$ Parte impar$\{x(t)\}$ (emplee señal c).

```
% Propiedad de parte impar
C_imag = tfourier(imag(c), t, dt, wmax);
assert(isequal(C_imag, imag(C)), 'No se cumple la propiedad');
```

El resultado muestra que la propiedad no se cumple