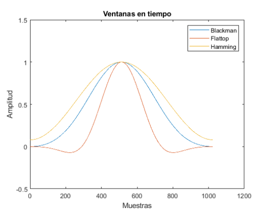
Discretización

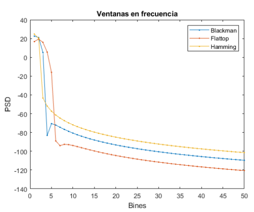
Ventanas

Diseñar 2 ventanas de N=1024 muestras con las funciones dedicadas. Plotearlas en tiempo y en frecuencia.

```
N=1024;
w_hamming=hamming(N);
w_flattop=flattopwin(N);
w_rect=rectwin(N);
w_blackman=blackman(N);
windows=table(w_blackman,w_flattop,w_hamming,'VariableNames',["Blackman" "Flattop" "Ham

%Plotear en tiempo
figure
plot(windows{:,:})
legend(windows.Properties.VariableNames)
ylim([-0.5 1.5])
xlabel("Muestras")
ylabel("Amplitud")
title("Ventanas en tiempo")
```




```

%Representación espectral
[Pxx,w]=periodogram(windows{:,:},[],N,[]);
windows_spectral=array2table(Pxx,"VariableNames",windows.Properties.VariableNames);

figure
plot(pow2db(windows_spectral{:,:}),'.-')
legend(windows.Properties.VariableNames)
xlabel("Bines")
ylabel("PSD")
title("Ventanas en frecuencia")
xlim([0 50])
```



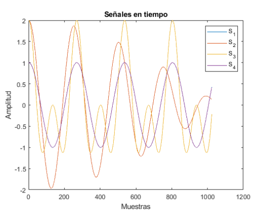
Generar las siguientes formas de onda con 1024 muestras y guardar cada una en un campo de una tabla.

1. $x(t) = cos(4t)$

2. $x(t) = cos(4t) + cos(4.5t)$
3. $x(t) = cos(4t) + cos(8t)$
4. $x(t) = cos(4t) + 0.001cos(8t)$

Plotear tempralmente y en espacil de frecuencias

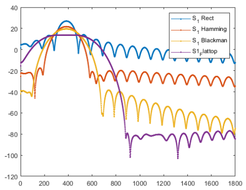
```
%Function handles
x_1= @(t) cos(4.*t);
x_2= @(t) cos(4.*t)+cos(4.5.*t);
x_3= @(t) cos(4.*t)+cos(8.*t);
x_4= @(t) cos(4.*t)+0.001.*cos(8.*t);
%Vector muestras
%t=(0:N-1)';
t=(linspace(0,6,N))';

%tabla de señales
signals=table(x_1(t),x_2(t),x_3(t),x_4(t),'VariableNames',["S_1" "S_2" "S_3" "S_4"]);

%Plotear en tiempo
figure
plot(signals{:,:})
legend(signals.Properties.VariableNames)
xlabel("Muestras")
ylabel("Amplitud")
title("Señales en tiempo")
```



Aplicar las ventanas a cada forma de onda mediante $y(t) = w(t)x(t)$ y alojarlas en distintas tablas

```
%Descomposicion espectral
%Con resolucion arbitrariamene grande
[Pxx,~]=periodogram(signals{:,:},[],100.*N,[]);
signals_rect_spectre=array2table(pow2db(Pxx),"VariableNames",signals.Properties.Variabl

[Pxx,~]=periodogram(signals{:,:}.*windows{:,"Hamming"},[],100.*N,[]);
signals_hamm_spectre=array2table(pow2db(Pxx),"VariableNames",signals.Properties.Variabl

[Pxx,~]=periodogram(signals{:,:}.*windows{:,"Blackman"},[],100.*N,[]);
signals_black_spectre=array2table(pow2db(Pxx),"VariableNames",signals.Properties.Variab

[Pxx,~]=periodogram(signals{:,:}.*windows{:,"Flattop"},[],100.*N,[]);
signals_flat_spectre=array2table(pow2db(Pxx),"VariableNames",signals.Properties.Variabl
```

```
%Plotear
figure
plot([...
    signals_rect_spectre{:,"S_1"}...
    signals_hamm_spectre{:,"S_1"}...
    signals_black_spectre{:,"S_1"}...
    signals_flat_spectre{:,"S_1"}...
    ...
    ],'.-')
xlim([0 1800])
legend("S_1 Rect","S_1 Hamming","S_1 Blackman","S1_Flattop" )
```



Hacer combinaciones de figures en donde se observen:

- Las diferencias de la señal 1 con distintas ventanas
- Las diferencias entre las señales 2-4 con distintas ventanas

Generar las siguientes formas de onda y alojarlas en una tabla

1. Sinusoidal
2. Triangular
3. Diente de sierra
4. Cuadrada

Hacer un tiledlayout (subplot) con:

- Su representación en tiempo
- Su representación en frecuencia

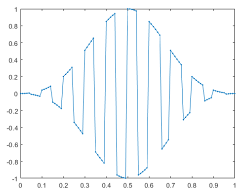
Aplicar las ventanas a las formas de onda anteriores y comparar resultados en tiempo y frecuencia.

Variar el número de muestras desde N y desde nfft del periodograma

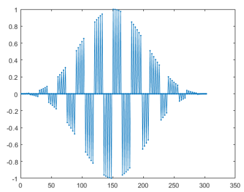
Resampling

Hacer upsampling y downsampling de las formas de onda anteriores

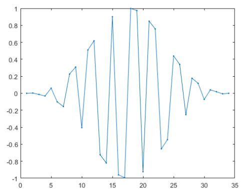
```
f = 10; %Hz
fs = 100; %Hz
Ts = 1/fs; %s
t = 0:Ts:1;

sqr_wave = square(2*pi*f.*t)';
sqr_wave_wind = sqr_wave.*blackman(numel(sqr_wave));

figure
plot(t, sqr_wave_wind, '.-')
```



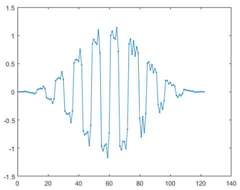
```
figure
y_up = upsample(sqr_wave_wind, 3);
plot(y_up, '.-');
```



```
figure
y_down = downsample(sqr_wave_wind, 3);
plot(y_down, '.-');
```



```
figure
y_resamp = resample(sqr_wave_wind, 6, 5);
plot(y_resamp, '.-')
```

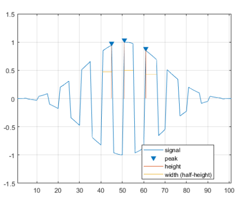


Usar distintos métodos de oversampling (interpolación) para una sección de los datos de cassini

Detector de picos

Detectar los picos de la forma de onda cuadrada y del audio del buho

```
figure
findpeaks(sqr_wave_wind, 'Annotate',"extents",...
    'WidthReference',"halfheight",...
    'MinPeakHeight', 0.8);
```




```
[y_owl, fs] = audioread("../../2022-1/Utils4SP/Datasets/Owl.wav");

figure
[pks, locs] = findpeaks(y_owl(:,1),...
    'Annotate',"extents",...
    'WidthReference',"halfheight",...
    'MinPeakHeight', 0.8)
```

```
pks = 26×1
    0.8180
    0.8439
    0.8439
    0.8180
    0.8697
    0.8439
    0.8697
    0.8697
    0.9730
    0.8439
```

```
    ⋮
locs = 26×1
       15438
       16218
       16423
       16556
       17095
       17500
       18442
       18613
       18918
       19123
    ⋮
```