LABORATORY 3
FINITE IMPULSE RESPONSE FILTERS

3.1. Introduction

A digital filter is a discrete system, used with the purpose of changing the amplitude and/or phase spectrum of a signal. The systems (filters) presented in this paper are linear and time invariant. The time domain response of a discrete LTIS is given by the convolution between the input signal $x[n]$ and the impulse rsponse of the system, denoted by $h[n]$, also called weight function:

$$y[n] = x[n] * h[n] = \sum_{k=-\infty}^{\infty} h[k]x[n-k] \tag{3.1}$$

Applying the **Z** transform to both members of relation (3.1) allows establishing a connection between the **Z** transforms of the input and output signals:

$$Y(z) = H(z)X(z) \tag{3.2}$$

where $H(z)$ is the *transfer function* of the discrete *LTIS*.

Finite impulse response filters (*FIR*) have a nonzero weight function $h[n]$ for $n \in \{0,1,\ldots,N-1\}$. N is called the *length of the impulse response*. For these filters, the filtering operation is based on the convolution relation:

$$y[n] = x[n] * h[n] = \sum_{k=0}^{N-1} h[k]x[n-k] \tag{3.3}$$

In this case the transfer function becomes:

$$H(z) = \sum_{n=0}^{N-1} h[n]z^{-n} \tag{3.4}$$

So the transfer function has no poles and is a polynomial function of order $N-1$ in z^{-1}. The coefficients of the filter are the values of the filter's impulse response.

3.2. Linear phase *FIR* filters

The *linear phase characteristic* of a *FIR* filter allows performing the filtering operation without introducing any phase distortions, which is important in the reconstruction of the signals.

The Fourier transform of the impulse response, called *frequency response* or *Fourier transform* function, is obtained by evaluating $H(z)$ from (3.4) along the unit radius circle in plane **Z**:

$$H(e^{j\omega}) = H(z)\big|_{z=e^{j\omega}} = \sum_{n=0}^{N-1} h[n]e^{-jn\omega} \tag{3.5}$$

where: $$\omega = 2\pi f = 2\pi \frac{F}{F_S} \tag{3.6}$$

FIR filters can have a *linear phase characteristic* if the weight function *h*[n] has the symmetry or antisymmetry property with respect to its central axis ($n = (N-1)/2$).

The symmetry or antisymmetry of $h[n]$ also implies a specific position of the zeros of the transfer function $H(z)$. The consequences in the **Z** plane are:

1. If z_i is a zero of $H(z)$, then $1/z_i$ is also a zero;
2. For practical applications, the coefficients $h[n]$ of the polynomial $H(z)$ are real and therefore any zero implies that its complex conjugate is also a zero.

Hence, the zeros always appear in groups of 4: two inside the unit circle and two outside it, in geometrical symmetry; the zeros on the unit circle appear in complex conjugate pairs; the real zeros appear in pairs, geometrically symmetric with respect to the circle, except the zeros $z = 1$ and $z = -1$ which are simultaneously their own inverse.

Type	*Length*	*h*[*n*]	*Mandatory zeros*	*We can design*
1	*N* odd	symmetric $h[n] = h[N-1-n]$	No constraints	LPF, HPF BPF, BSF
2	*N* even	symmetric $h[n] = h[N-1-n]$	$z = -1$	LPF, BPF
3	*N* odd	antisymmetric $h[n] = -h[N-1-n]$ $h\left[\frac{N-1}{2}\right] = 0$	$z = 1$ and $z = -1$	BPF, Hilbert transf., differentiator
4	*N* even	antisymmetric $h[n] = -h[N-1-n]$	$z = 1$	HPF, BPF, Hilbert transf., differentiator

3.3. Design of linear phase *FIR* filters

The main methods used in the design of this important category of *FIR* filters are: *the window method*, *the frequency sampling method* and *methods based on minimizing the error in the frequency domain*, using a certain error criterion.

3.3.1. The window method

***Syntax*:**

```
w = boxcar(N)
```

- returns a *column vector* `w` of length `N` which will contain the values of the rectangular window.

In order to reduce the ripples in the two bands, we can use other types of windows, which perform less abrupt truncations of the impulse response $h_\infty[n]$, compared to the rectangular window: triangular window (Bartlett), Blackman, Hamming, Hanning, Kaiser.

```
w = kaiser(N,beta)
```

- returns a column vector w of length N which will contain the values of the Kaiser window; `beta` is the value of a parameter which influences the attenuation of the secondary lobes from the Fourier transform of the window.

In MATLAB we also have the window functions `bartlett`, `blackman`, `hamming`, `hanning`, `triang`, which can be called with the general syntax:

```
w = window_name(N)
```

- returns a *column vector* `w` of length `N` which will contain the values of the window specified by `window_name`.

E1. ***Exercise*:**
Compare the rectangular window to the other types of windows from the point of view of the theoretical requirements, by running the program **`demofer.m`**.

a) For a window length **N=20** measure the width of the main lobe of the spectrum of each window and verify the values in the table below. Compare the width of the lobes of different windows' spectra.
b) Measure the attenuation in dB of the secondary lobes for each window and verify the values in the table.

c) Redo ex. a) and b) for **N=40**. How does the width of the main lobe modify when N increases? How about the secondary lobes' attenuation?

Window type	*Width of the main lobe*	*Maximum amplitude of the lateral lobes*
Rectangular	$\frac{4\pi}{N}$	–13,3
Triangular	$\frac{8\pi}{N+1}$	–26,5
Hanning	$\frac{8\pi}{N}$	–31,5
Hamming	$\frac{8\pi}{N}$	–42,7
Blackman	$\frac{12\pi}{N}$	–58,1
Kaiser with $\beta = 7{,}865$	$\frac{2\pi\beta}{N} = \frac{15{,}73\pi}{N}$	–57 (may be adjusted through β)

Let $h[n]$ be the impulse response of the digital *FIR* filter. Denoting by N the length of this causal sequence, defined on $0 \le n \le N-1$, the **Z** transform of $h[n]$ is the transfer function of the filter:

$$H(z) = \sum_{n=0}^{N-1} h[n]z^{-n} = h[0] + h[1]z^{-1} + h[2]z^{-2} + ... + h[N-1]z^{-(N-1)} \quad (3.11)$$

Keeping in mind the fact that in MATLAB the first element of a vector has index 1, the transfer function of a digital FIR filter can be implemented as:

$$H(z) = h[1] + h[2]z^{-1} + h[3]z^{-2} + ... + h[n+1]z^{-n} \quad (3.12)$$

where $h[1], h[2], h[3], ..., h[n+1]$ are the coefficients of the transfer function of the filter (samples of the impulse response) and n is the order of the filter.

***Observations*:**

- By comparing (3.11) and (3.12) we can conclude that the order n of the filter from (3.12) is equal to $N-1$ (length of the sequence minus one) from (3.11).
- The MATLAB function `fir1` computes the coefficients of the filter such that their sum equals 1, imposing that in $\omega = 0$, $H(e^{j\omega}) = \sum_n h[n] = 1$ (0 dB).
- Frequency normalization is done differently:

$$f_{MATLAB} = \frac{F}{F_S / 2} = 2\frac{F}{F_S} = 2 f_{teoretic} \tag{3.13}$$

`fir1` – ***Design of digital FIR filters of type low-pass (LPF), high-pass (HPF), band-pass (BPF) and band-stop (SBF).***

***Syntax*:**

`h = fir1(n,wn)`

- it designs, using the window method, with a Hamming window of length `n` + 1, a *low-pass filter* of order `n` with the normalized cut-off frequency `wn`; `wn` takes values between 0 and 1, where 1 corresponds to `Fs/2`; if the cut-off frequency and the sampling frequency are expressed in Hz then:
  `wn` = 2*cut-off frequency/sampling frequency;
- if `wn` is a vector with 2 elements, `wn=[w1,w2]` with `w1<w2`, it will design using the window method, with a Hamming window of length `n` + 1, a *band-pass filter* of order `n`, with the pass band between `w1` and `w2` (normalized by the previous rule).
- it will return the line vector `h` of length `n`+1 which contains the coefficients of the transfer function $H(z)$ of the filter ($\mathrm{h} = \left[h[1], h[2], h[3], ..., h[n+1]\right]$; see (3.12));

`h = fir1(n,wn,'high')`

- it designs, using the window method, with a Hamming window of length `n` + 1, a *high-pass filter* of order `n` (`n` must be even) with the cut-off frequency `Wn`

`h = fir1(n,wn,'stop')`

- `wn` is a vector with 2 elements, `wn=[w1,w2]` with `w1<w2`, it will design using the window method, with a Hamming window of length `n` + 1, a *band-stop filter* of order `n` (`n` must be even), with the stop band between `w1` and `w2`.

`h = fir1(n,wn,window_type(n+1))`

- `window_type` specifies the type of the window with which we want to design the filter; instead of `window_type(n+1)` we can have any column vector of length `n`+1 which contains the values of the samples of a sequence that will be used as window in the filter design;
- it designs, using the window method, with the specified window of length `n` + 1, a *low-pass filter* of order `n` with cut-off frequency `Wn`;

- if `wn` is a vector with 2 elements, `wn=[w1,w2]` with `w1<w2`, it will design using the window method, with the specified window of length `n` + 1, a *band-pass filter* of order `n`, with the pass band between `w1` and `w2`.

`h = fir1(n,wn,'high',tip_fereastra(n+1))`

- it designs, using the window method, with the specified window of length `n` + 1, a *high-pass filter* of order `n` (`n` must be even) with cut-off frequency `Wn`;

`h = fir1(n,wn,'stop',tip_fereastra(n+1))`

- `wn` is a vector with 2 elements, `wn=[w1,w2]` with `w1<w2`, it will design using the window method, with the specified window of length `n` + 1, a *band-stop filter* of order `n` (`n` must be even) with the stop band between `w1` and `w2`.

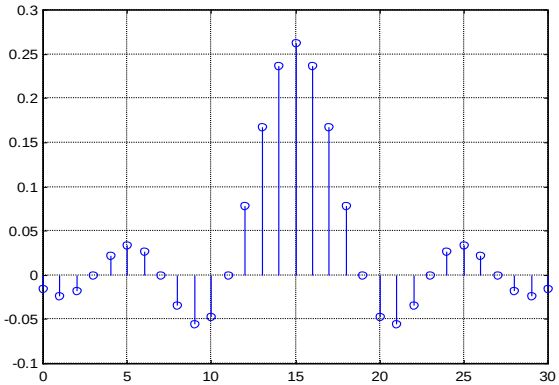
***Example*:**

Design by using the window method and a rectangular window a LPF of length $N=31$ and cut-off frequency $F_c=5\text{kHz}$. The sampling frequency is $F_S=40\text{kHz}$. Represent graphically the weight function of the filter, the positioning of the zeros in the **Z** plane and the amplitude-frequency characteristic of the transfer function (linear and in dB).

If the filter has length $N=31$ then the order of the filter (`n` from the MATLAB syntax) will be $n=N-1=30$, the window's length will be $n+1=N=31$ and the normalized cut-off frequency (`Wn` from the MATLAB syntax) computes as:

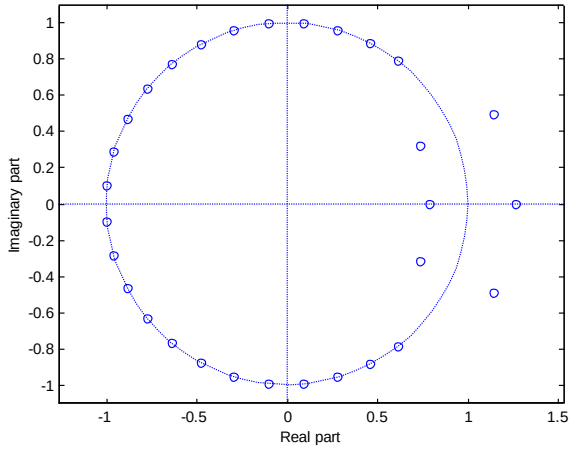
$$f_{MATLAB}=\frac{F_t}{F_S/2}=\frac{5}{40/2}=0,25$$

```
N=31; Fc=5000; Fs=40000;
wn=2*Fc/Fs;
h=fir1(N-1,wn,boxcar(N));
n=0:N-1;
figure(1),stem(n,h),grid
```

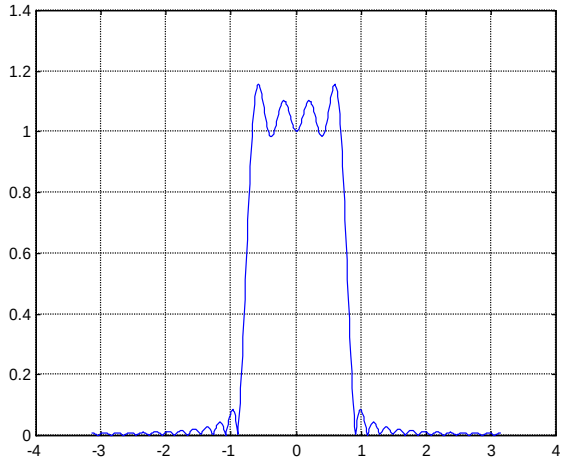


It can be observed that the designed filter is of type 1 (symmetric h and odd length).

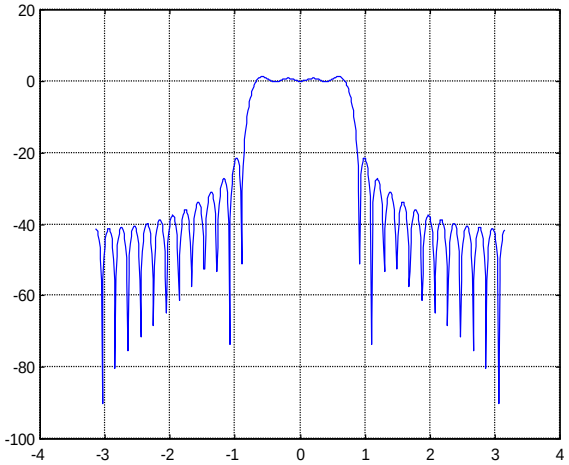
```
figure(2),zplane(h)
```




```
H=fft(h,512);
w=-pi:2*pi/512:pi-2*pi/512;
figure(3),plot(w,fftshift(abs(H))),grid
```




```
figure(4),plot(w,20*log10(fftshift(abs(H)))),grid
```



We note the presence of the ripples and their dimension near the transition area.

E2. *Exercises*:

1. a) Complete the previous example by designing another filter with the same parameters, using the Hamming window. Represent on the same graph with different colors the amplitude-frequency characteristics of the two filters. Compare the widths of the transition bands and the band-stop attenuations.
   b) Change the length of the filters to $N = 51$. What do you observe regarding the dimension of the ripples and the transition area?
   c) How much should N be such that the filter designed with the Hamming window has the same dimension for the transition band as the filter designed with a rectangular window of length $N = 31$?
2. a) Using the windows method design a high-pass filter of length $N = 31$ and cut-off frequency $F_c = 5\text{kHz}$ with a rectangular window. The sampling frequency is $F_S = 40\text{kHz}$. Mention the type of the filter. Represent graphically the filter's weight function, the zeros' positions in the **Z** plane and the amplitude-frequency characteristic for the transfer function (linear and dB representation).
   b) Redo point a) using $N = 51$. Represent on the same graph the amplitude-frequency characteristics obtained in the two cases. Discuss the results.
   c) Redo point a) using the other windows. Represent on the same graph the amplitude-frequency characteristics. Comment the results by presenting both the advantages and drawbacks of using each type of window.
3. a) Using the windows method design a band-pass filter of length $N = 46$ and cut-off frequencies $F_{c1} = 5\text{kHz}$ and $F_{c2} = 15\text{kHz}$ with a rectangular window. The sampling frequency is $F_S = 40\text{kHz}$. Mention the type of the filter. Represent graphically the weight function of the filter, the zeros' positions in the **Z** plane and the amplitude-frequency characteristic for the transfer function (both linear and in dB).
   b) Redo point a) using $N = 86$. Represent on the same graph the amplitude-frequency characteristics obtained in the two cases. Discuss the results.
   c) Redo point a) using the other windows. Represent on the same graph the amplitude-frequency characteristics. Comment the results by presenting both the advantages and drawbacks of using each type of window.
4. Redo exercise 3 for the cut-off frequencies $F_{c1} = 8\text{kHz}$ and $F_{c2} = 10\text{kHz}$. Comment the results.
5. a) Using the windows method design a band-stop filter of length $N = 45$ and cut-off frequencies $F_{c1} = 5\text{kHz}$ and $F_{c2} = 15\text{kHz}$ with a rectangular window. The sampling frequency is $F_S = 40\text{kHz}$. Mention the type of the filter.

Represent graphically the weight function of the filter, the zeros' positions in the **Z** plane and the amplitude-frequency characteristic for the transfer function (both linear and in dB).

b) Redo point a) using $N = 85$. Represent on the same graph the amplitude-frequency characteristics obtained in the two cases. Discuss the results.

c) Redo point a) using the other windows. Represent on the same graph the amplitude-frequency characteristics. Comment the results by presenting both the advantages and drawbacks of using each type of window.

6. Redo exercise 5 for the cut-off frequencies $F_{c1} = 8\text{kHz}$ and $F_{c2} = 10\text{kHz}$. Comment the results.

3.3.2. The frequency sampling method

`fir2` – Design of digital FIR filters by using the method of frequency sampling.

Syntax:

`h = fir2(n,f,m)`

- `n` is the *order of the filter* that we want to design (length minus 1);
- the vector `f` contains normalized frequencies; these values are between 0 and 1 (the first element of `f` must be 0 and the last one must be 1), where 1 corresponds to `Fs/2;` if the frequencies are expressed in Hz, then in order to compute the elements of `f` we will make the conversion:

  frequency[Hz]/(sampling frequency [Hz]/2);
- the vector `m` contains the values of the absolute value of the desired frequency response; these values correspond to the frequencies from the vector `f`; the command `plot(f,m)` displays the modulus characteristic of the desired frequency response;
- it will return the *line vector* `h` of length `n + 1` which contains the coefficients of the transfer function $H(z)$ of the filter ($\text{h} = [h[1], h[2], h[3], ..., h[n+1]]$; see (3.23));
- the design is done using a Hamming window of length `n + 1`.

`h = fir2(n,f,m,window_type(n+1))`

- `n`, `f`, `m` and `h` have the same significance as in the previous syntax;
- `window_type` specifies the type of window that we want when designing the filter; instead of `window_type(n+1)` we can have any column vector of length `n+1` which contains the values of the samples of a sequence that will be used as window in the filter design.

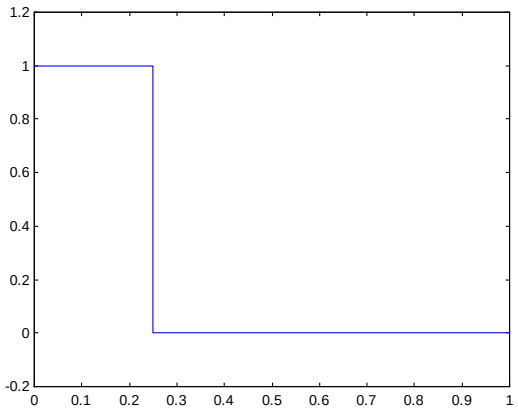
***Example*:**

Design using the frequency sampling method and a rectangular window a low-pass filter of length $N=31$ and cut-off frequency $F_c=5\text{kHz}$. The sampling frequency is $F_S=40\text{kHz}$. Represent graphically the filter's weight function, the zeros' positions in the **Z** plane and the amplitude-frequency characteristic for the transfer function (both linear and in dB).

If the filter has length $N=31$ then the filter's order (`n` from the MATLAB syntax) will be $n=N-1=30$, the window's length will be $n+1=N=31$, and the normalized cut-off frequency (`Wn` from the MATLAB syntax) is computed as:

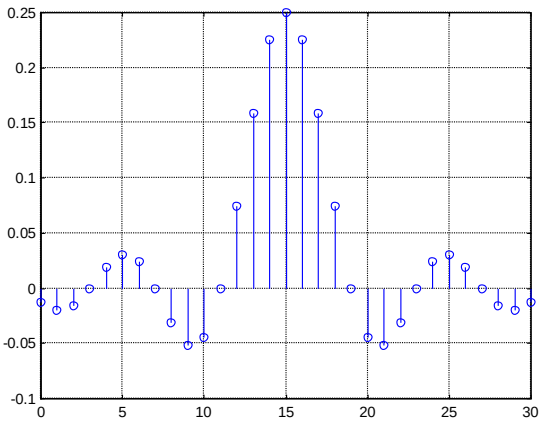
$$f_{MATLAB}=\frac{F_t}{F_S/2}=\frac{5}{40/2}=0,25$$

```
f=[0,0.25,0.25,1];
m=[1,1,0,0];
figure(1),plot(f,m)
axis([0 1 -0.2 1.2])
```



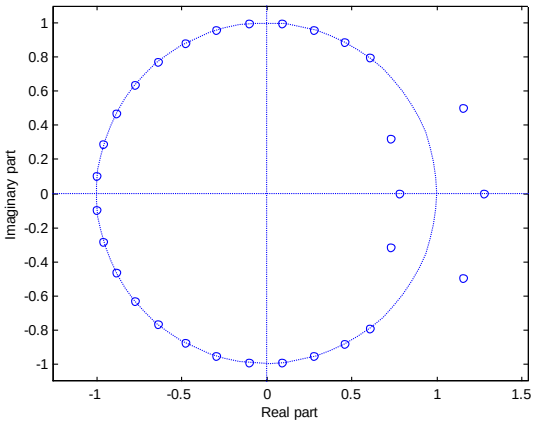
The frequency response of the desired filter (ideal low-pass filter) has been displayed; the corespondence between the values of the `f` and `m` vectors can be observed; the `axis` command has been used for a better visualization of the graph.

```
N=31;
h=fir2(N-1,f,m,boxcar(N));
n=0:N-1;
figure(2),stem(n,h),grid
```

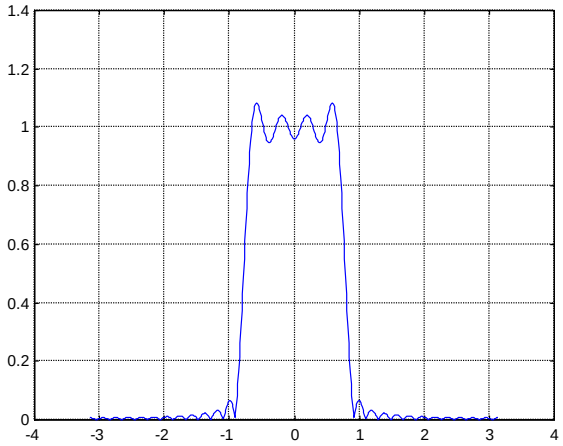


We can observe the difference regarding the values of the impulse response obtained through the design with `fir1`; this happens due to the fact that the function `fir2` does not impose the condition that in $\omega = 0$ to have $H(e^{j\omega}) = \sum_n h[n] = 1$ (0 dB).

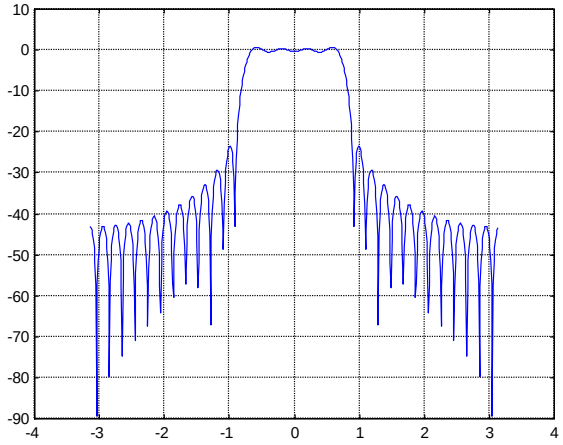
```
figure(3),zplane(h)
```




```
H=fft(h,512);
w=-pi:2*pi/512:pi-2*pi/512;
figure(4),plot(w,fftshift(abs(H))),grid
```




```
figure(5),plot(w,20*log10(fftshift(abs(H)))),grid
```



Note the presence and the dimension of the ripples near the transition area.

E3. ***Exercises*:**
Redo the exercises from `fir1` (see **E2**) using `fir2` for the filter design. Comment the results.

3.3.3. MATLAB functions used for *FIR* filters design through methods based on approximation in frequency domain

The theoretical basis is the theory of approximating the transfer functions by using a certain *error minimization criterion* between the approximated transfer function and the desired one. The error is evaluated in the frequency bands of practical interest (the effective band-pass and the effective band-stop in the case of a LPF).

`firls` – ***Design of digital FIR filters by approximation in the least squares sense of a specified frequency response.***

***Syntax*:**

`h = firls(n,f,m)`
- `n` is the *order of the filter* that we want to design (length minus 1);
- the vector `f` contains normalized frequencies; these values are between 0 and 1, where 1 corresponds to `Fs/2;` if the frequencies are expressed in Hz, then in order to compute the elements of `f` we will make the conversion:
  frequency[Hz]/(sampling frequency [Hz]/2);
- the vector `m` contains the values of the absolute value of the desired frequency response; these values correspond to the frequencies from the vector `f`; the command `plot(f,m)` displays the modulus characteristic of the desired frequency response;
- *the length of vectors* `f` *and* `m` *must be an even number*;
- it will return the line vector h of length n + 1 which contains the coefficients of the transfer function of the filter ($h = [h[1], h[2], h[3], ..., h[n+1]]$; see (3.23));

`h = firls(n,f,m,'hilbert')`
- `n`, `f`, `m` and `h` have the same significance as in the previous syntax;
- a *Hilbert transformer* will be designed.

`h = firls(n,f,m,'differentiator')`
- `n`, `f`, `m` and `h` have the same significance as in the previous syntax;
- a *digital differentiator* will be designed.

`h = firls(n,f,m,w)`

- `n`, `f`, `m` and `h` have the same significance as in the previous syntax;
- the vector `w` contains the values of the weights of the approximations in each band; its length equals half the length of the vectors `f` and `m`.

`h = firls(n,f,m,w,'hilbert')`

- `n`, `f`, `m`, `w` and `h` have the same significance as in the previous syntax;
- a *Hilbert transformer* will be designed.

`h = firls(n,f,m,w,'differentiator')`

- `n`, `f`, `m`, `w` and `h` have the same significance as in the previous syntax;
- a *digital differentiator* will be designed.

`firpm` – ***Design of digital FIR filters by Chebyshev approximation, using the Parks-McClellan algorithm (Remez changes algorithm).***

***Syntax*:**

`h = firpm(n,f,m)`

- `n` is the *order of the filter* that we want to design (length minus 1);
- the vector `f` contains normalized frequencies; these values are between 0 and 1, where 1 corresponds to `Fs/2;` if the frequencies are expressed in Hz, then in order to compute the elements of `f` we will make the conversion:
  frequency[Hz]/(sampling frequency [Hz]/2);
- the vector `m` contains the values of the absolute value of the desired frequency response; these values correspond to the frequencies from the vector `f`; the command `plot(f,m)` displays the modulus characteristic of the desired frequency response;
- *the length of vectors* `f` *and* `m` *must be an even number*;
- it will return the line vector h of length n + 1 which contains the coefficients of the transfer function of the filter ($h = [h[1], h[2], h[3], ..., h[n+1]]$; see (3.23)).

`h = firpm(n,f,m,'hilbert')`

- `n`, `f`, `m` and `h` have the same significance as in the previous syntax;
- a *Hilbert transformer* will be designed.

`h = firpm(n,f,m,'differentiator')`

- `n`, `f`, `m`, `w` and `h` have the same significance as in the previous syntax;

- a *digital differentiator* will be designed.

`h = firpm(n,f,m,w)`

- `n`, `f`, `m` and `h` have the same significance as in the previous syntax;
- the vector `w` contains the values of the weights of the approximations in each band; its length equals half the length of the vectors `f` and `m`.

`h = firpm(n,f,m,w,'hilbert')`

- `n`, `f`, `m`, `w` and `h` have the same significance as in the previous syntax;
- a *Hilbert transformer* will be designed.

`h = firpm(n,f,m,w,'differentiator')`

- `n`, `f`, `m`, `w` and `h` have the same significance as in the previous syntax;
- a *digital differentiator* will be designed.

`firpmord` – ***Determining the input parameters of the function*** **`firpm`**.

***Syntax*:**

`[n,fo,mo,w] = firpmord(f,m,dev)`

- the vector `f` contains the values of the frequencies at the ends of the interest bands (for example for a low-pass filter, the upper limit of the pass band and the lower limit of the stop band);
- the vector `m` specifies the values of the desired amplitudes for the absolute value of the frequency response of the filter, in the bands specified by the vector `f` (for example for a LPF, the vector `m` contains the values 1, corresponding to the pass band, and 0, corresponding to the stop band);
- *length of vector* `f` = 2·(*length of vector* `m`) – 2;
- the vector `dev` contains the maximum admissible values of the approximation errors between the desired amplitude (specified in the vector `m`) and the amplitude of the modulus characteristic of the frequency response of the resulting filter;
- *length of vector* `dev` = *length of vector* `m` ;
- results `n`, `fo`, `mo` and `w` are the input parameters of the function `firpm`: `h=firpm(n,fo,mo,w)`.

`[n,fo,mo,w] = firpmord(f,m,dev,Fs)`

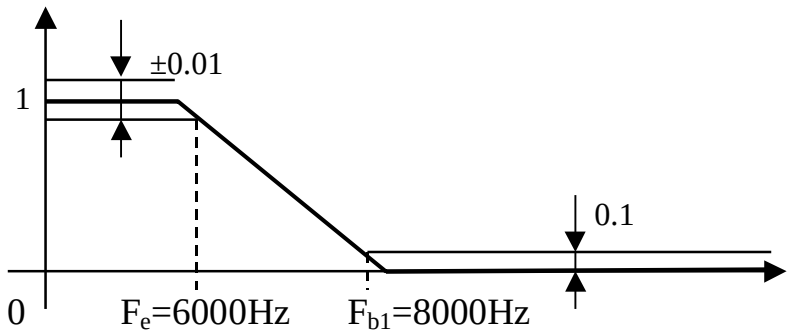
- `Fs` specifies the value of the sampling frequency; if the sampling frequency is not specified (like in the previous syntax), the value `Fs` = 2 Hz is chosen implicitly (this means that the Nyquist frequency is 1 Hz)
- the other parameters have the same significance as in the previous syntax.

***Observation*:**

In some cases the function **`firpmord`** underestimates the order of the filter. In this case, if we are not satisfied with the characteristics of the obtained filter, we can try a higher order by increasing the order by 1 or 2, using for the design `h=firpm(n+1,fo,mo,w)` or `h=firpm(n+2,fo,mo,w)`.

***Example*:**

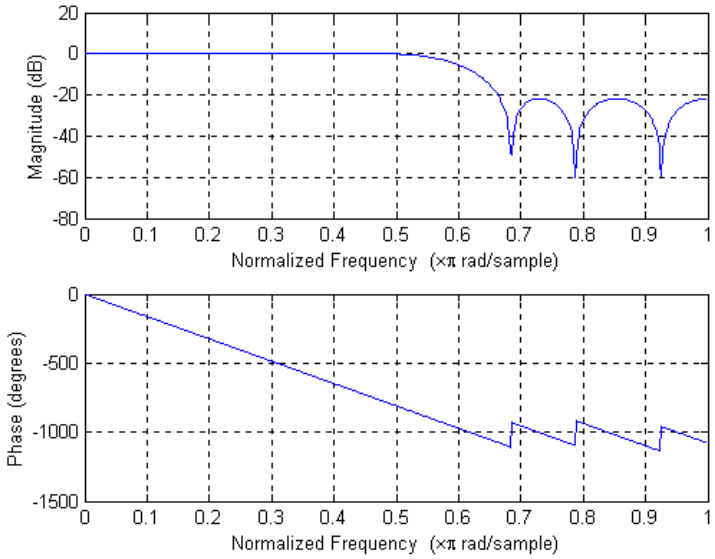
Design using the Parks-McClellan algorithm a low-pass filter having the band-pass ripple of 0.01 and the band-stop ripple of 0.1. The maximum frequency in the band-pass is of 6 kHz and the minimum frequency in the band-stop is of 8 kHz. The sampling frequency is Fs = 24 kHz.
The filter is presented in the following:



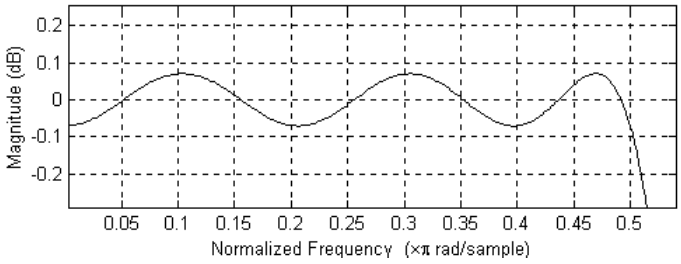

```
Fs=24000;
f=[6000,8000];
m=[1,0];
dev=[0.01,0.1];
[n,fo,mo,w]=firpmord(f,m,dev,Fs)        →  n =
                                              16
                                           fo =
                                                    0
                                              0.5000
                                              0.6667
                                              1.0000
                                           mo =
                                                1
                                                1
                                                0
                                                0
                                           w =
                                              10
```

1

```
h=firpm(n+2,fo,mo,w);
freqz(h)
```



Note that the attenuation in the band-stop is over 20dB, corresponding to the imposed design's conditions (maximum gain of 0.1 in the band-stop).
The image can be zoomed in in the band-pass area to measure the ripple's dimension and to verify the maximum imposed ripple (1±0.01 means a gain of approximately ±0.08dB).



E4. *Exercises:*

1. Design a low-pass filter of order 15 with the cut-off frequency F_c=2KHz and the sampling frequency F_s=20KHz, using the least squares approximation method (`firls` from MATLAB). The minimum frequency in the band-stop is F_b=2.2KHz. Represent the filter's coefficients and the amplitude-frequency characteristic.

2. Design using the Parks-McClellan algorithm (`firpm` in MATLAB**)** , a high-pass filter with the band-pass ripple of 0.02 and the band-stop ripple of 0.08. The maximum frequency in the band-stop is 2400Hz, and the minimum band-pass frequency is 3000Hz. The sampling frequency is 8kHz. Determine the

filter's order. Represent the filter's coefficients and the amplitude-frequency characteristic.

3. Design using `firpm` a filter to approximate the band-pass function:

$$\left|H(e^{j\omega})\right| = \begin{cases} 0, & 0 \le \omega \le 0.3\pi \\ 1, & 0.4\pi \le \omega \le 0.6\pi \\ 0 & 0.7\pi \le \omega \le \pi \end{cases}$$

The filter must have a band-pass attenuation of maximum 1 dB and the band-stop attenuation of minimum 40dB. The sampling frequency is of 24kHz. Determine the filter's order. Represent the filter's coefficients and the amplitude-frequency characteristic.

4. Redo the previous exercise using `firls`.