LABORATORY 4
INFINITE IMPULSE RESPONSE FILTERS

4.1. Introduction

Infinite impulse response filters (IIR) prove to have more benefits than FIR filters in some applications: their selective characteristics are better, with an order much smaller then the usual FIR orders. As a drawback, IIR filters can not have linear phase characteristics. Being impossible to obtain a linear phase, implies that either we approximate both the amplitude and the phase characteristics, either we approximate only the amplitude and then search for ways to correct the phase distortions. There is only one type of IIR filter that has one of the 2 characteristics – constant (the all-pass filter).

The IIR filter is described in the discrete-time domain by the finite difference equation:

$$y[n]=\sum_{i=0}^{M} b_i x[n-i]-\sum_{i=1}^{N} a_i y[n-i] \tag{4.1}$$

The transfer function derived from the previous finite difference equation is:

$$H(z)=\frac{\sum_{i=0}^{M} b_i z^{-i}}{1+\sum_{i=1}^{N} a_i z^{-i}} \tag{4.2}$$

where it was assumed that $a_0=1$.

Designing a digital IIR filter implies to determine the coefficients a_i and b_i from the expression in (4.2), in such a way that the impulse response function , $h[n]$, or the frequency response, $H(e^{j\omega})$, to approximate the target requirements in time or in frequency, demanded for implementation.

The design methods for IIR filters are divided in 2 categories:

1. *Indirect design methods* based on the design of an analog filter and transforming it into a digital filter;
2. *Direct methods* – aim to design directly a digital filter, without an analog model. These methods are based on approximation criterions in time or frequency.

4.2. Indirect design of *IIR* filters

This method starts with the design of an analog filter and then transform it into a digital filter with equivalent performances. There are 2 advantages in this approach:
a) we can exploit the knowledge and the design methods of analog filters;
b) there are transformations that preserve the selectivity properties of the analog model.

The following steps are needed:
1. Transform the requirements desired for the digital filter into requirements for an analog prototype filter;
2. Achieve the transfer function of the analog prototype, $H_a(s)$, that features the requirements from 1.
3. Transform the analog filter's transfer function into the transfer function of a digital filter, $H(z)$, with the same performances.

Going from $H_a(s)$ to $H(z)$, involves in time going from the continuous variable *t* to the discrete variable *n*, and in frequency – going from the *s* plane to the *z* plane. All such transforms need to satisfy 2 requirements:
I. *transform an analog stable filter into a digital stable filter;*
II. *preserve the selectivity characteristics (magnitude and phase, if possible) of the analog filter*.

First request means to transform the left half-part from the s plane into the interior of the unity circle in the Z domain. 2nd requirement involves a linear conversion of the $\{j\Omega\}$ axis into the unity circle contour ($z = e^{j\omega}$) in the Z plane.

There are 4 such transforms known in the literature:
1. The differential equation transform;
2. The impulse response invariance method;
3. The bilinear transform;
4. The adapted Z transform.

We will present the procedures 2 and 3 that satisfy entirely the requirement I and very good (in certain conditions) the requirement II.

4.2.1. The impulse response invariance method

This method relies on preserving the impulse response: the impulse response of the digital filter, $h[n]$, is the sampled version of the impulse response of the analog filter, $h_a(t)$. We will use the notation $\Omega = 2\pi F$ for the analog angular frequency domain and $\omega = 2\pi f$ for the normalized angular frequency domain of the digital filter. The method relies in the following steps:

1. Decompose $H_a(s)$ into partial fractions:

$$H_a(s) = \sum_{k=1}^{N} \frac{A_k}{s - s_k} \tag{4.3}$$

2. Determine the impulse response of the analog filter, $h_a(t)$:

$$h_a(t) = \sum_{k=1}^{N} A_k \ \mathcal{L}^{-1}\left\{\frac{1}{s - s_k}\right\} = \sum_{k=1}^{N} A_k e^{s_k t} u(t) \tag{4.4}$$

3. Determine the impulse response function for the digital filter:

$$h[n] = Th_a(nT) = T\sum_{k=1}^{N} A_k e^{s_k nT} u[n] \quad \text{cu} \quad u[n] = u(nT) \tag{4.5}$$

4. Compute the transfer function $H(z)$:

$$H(z) = \sum_{n=0}^{\infty} h[n] z^{-n} = \sum_{n=0}^{\infty}\left(T\sum_{k=1}^{N} A_k e^{s_k nT}\right) z^{-n} = T\sum_{k=1}^{N} A_k \left(\sum_{n=0}^{\infty} e^{s_k T} z^{-1}\right)^n = \sum_{k=1}^{N} \frac{TA_k}{1 - e^{s_k T} z^{-1}} \tag{4.6}$$

Performing those steps, the digital filter's coefficients will be obtained: a_i, b_i.

It is proved that the digital filter has the frequency response of the analog filter for the frequency interval:

$$\Omega \in \left[-\frac{\Omega_s}{2}, \frac{\Omega_s}{2}\right] \Leftrightarrow \omega \in [-\pi, \pi]$$

if and only if the following conditions are realized:

$$h[n] = Th_a(nT) \tag{4.7}$$

$$\omega = \Omega T \Leftrightarrow \Omega = \frac{\omega}{T} \tag{4.8}$$

$$H_a(j\Omega) = 0 \quad \text{la} \quad |\Omega| \geq \Omega_M \tag{4.9}$$

This method performs well for the band-limited signals (eq. 4.9). It can be used to design LPF and BPF, but it can not be used for HPF and BSF. There is a way still to design high-pass and band-stop filters using impulse invariance: it can be designed a digital low-pass filter with impulse invariance and then apply a frequency transform in the Z plane (the low-pass will be transformed into high-pass or band-pass, etc.)

4.2.2. The bilinear transform method

Using *the bilinear transformation* from the s plane to the Z plane, the transfer function preserves its algebraic form (of rational function). This transform is defined by (T represents the sampling time):

$$s = \frac{2}{T}\frac{1-z^{-1}}{1+z^{-1}} \Leftrightarrow \frac{sT}{2} = \frac{1-z^{-1}}{1+z^{-1}} \tag{4.10}$$

There are several advantages offered by this bilinear transform:
a) the aliasing errors due to the impulse invariance method are dropped, since the entire $\{j\Omega\}$ axis in *s* plane is transformed into the contour of the unity circle in the *Z* plane.
b) it transforms stable analog filters into stable digital filters.
c) it is a simple algebraic transform that can be obtained by the change in variable

$$H(z) = H_a(s)\Big|_{s=\frac{2}{T}\cdot\frac{1-z^{-1}}{1+z^{-1}}} \tag{4.11}$$

The drawback for this method is given by the nonlinear transform of the axis $\{j\Omega\}$ into the unity circle $z = e^{j\omega}$ in plane **Z**. Let's consider $s = j\Omega$ in (4.10), which imply $|z| = 1$, meaning $z = e^{j\omega}$. It results:

$$\frac{j\Omega T}{2} = \frac{1-e^{-j\omega}}{1+e^{-j\omega}} = j\tan\frac{\omega}{2} \quad \Leftrightarrow \quad \Omega = \frac{2}{T}\tan\frac{\omega}{2} \quad \Leftrightarrow \omega = 2\tan^{-1}\frac{\Omega T}{2} \tag{4.12}$$

This method can be used for analog systems with a constant frequency response. It can not be used for differentiators or Bessel filters, since the transform does not preserve the linearity of the amplitude-frequency characteristic. It can be applied for LPF, HPF, band-pass and stop-band filters, also all-pass.

4.2.3. The indirect design of digital *IIR* filters using MATLAB functions

I. Design of the analog filters

The transfer function of an analog filter of order *N* is:

$$H_a(s) = \frac{C(s)}{D(s)} = \frac{\sum_{i=0}^{M} c_i s^i}{\sum_{i=0}^{N} d_i s^i} \qquad ; \qquad N \ge M \tag{4.13}$$

The main analog prototype filters will be briefly presented in the following section: Butterworth, Cebîşev I, Cebîşev II and elliptical (Cauer) filters. Those prototypes will be used later to design digital filters.

- *Butterworth analog filters*

The frequency response of a low-pass analog Butterworth filter of order N and cutoff Ω_t is given by:

$$|H_a(j\Omega)|^2 = \frac{1}{1+\left(\frac{\Omega}{\Omega_t}\right)^{2N}} \tag{4.15}$$

It realizes a maximally flat characteristic for $\Omega = 0$. For $\Omega = \Omega_t$, no matter what order N we have, the square of the frequency response will reach 1/2. By increasing the order N, the transition band will become more and more narrow.

- *Cebîşev I analog filters*

Cebîşev I filters are polynomial filters (having only poles and no zeros). The frequency response shows an equiripple characteristic in the band-pass, while it then decreases monotonically in the stopping band. Compared to all the other polynomial filters of order N, Cebisev 1 filters have the most reduced transition band.

For an analog low-pass Cebisev 1 filter of order N and the upper limit of the passing band Ω_e, the square of the amplitude-frequency characteristic is given by:

$$|H_a(j\Omega)|^2 = \frac{1}{1+\varepsilon^2 C_N^2\left(\frac{\Omega}{\Omega_e}\right)} \tag{4.16}$$

where $C_N(x)$ is the Cebîşev polynomial of order N :

$$C_N(x) = \begin{cases} cos(N\, cos^{-1} x) \quad , & for \quad |x| \le 1 \\ cosh(N\, cosh^{-1} x), & for \quad |x| > 1 \end{cases} \tag{4.17}$$

In the stop band, the monotonic decrease of the frequency response realizes an approximation of type maximally flat of the ideal value 0. For an increasing order N, the slope is more and more abrupt. For 2 filters having the same order N, the slope is more abrupt for the one with higher ripples in the pass band. The filter's performances are completely defined by the factor ε, that specifies the size of the ripples in the pass band and the order N that drives the slope of the transition band.

- *Cebîşev II analog filters*

Cebîşev II filters realize an approximation in a Cebîşev sense of the amplitude-frequency characteristic for the stop band, the function presenting a monotonic decrease in the pass band. They are also called *inverse Cebîşev filters* because they have an equiripple behavior in the stop-band and maximally flat characteristic in the pass-band.

For an analog low-pass Cebisev 2 filter of order *N* and the lower limit of the stopping band Ω_b, the square of the amplitude-frequency characteristic is given by:

$$\left|H_a(j\Omega)\right|^2 = 1 - \frac{1}{1+\varepsilon^2 C_N^2\left(\dfrac{\Omega_b}{\Omega}\right)} = \frac{\varepsilon^2 C_N^2\left(\dfrac{\Omega_b}{\Omega}\right)}{1+\varepsilon^2 C_N^2\left(\dfrac{\Omega_b}{\Omega}\right)} \tag{4.18}$$

where $C_N(x)$ is the Cebîşev polynomial of order N.

- *Elliptic analog filters*

Elliptic filters have equiripples characteristics both in the pass-band and the stop-band. For an analog low-pass elliptical filter of order *N*, the square of the amplitude-frequency characteristic is given by:

$$\left|H_a(j\Omega)\right|^2 = \frac{1}{1+\varepsilon^2 F_N^2(\Omega)} \tag{4.19}$$

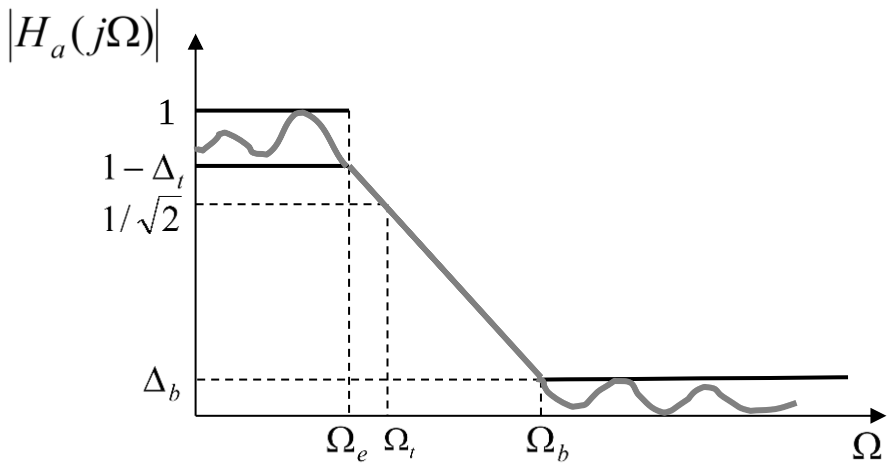
where the rational function Cebîşev $F_N(\Omega)$, first introduced by Cauer in the linear circuits theory, is given by:

$$F_N(\Omega) = \begin{cases} C_1 \displaystyle\prod_{i=1}^{\frac{N}{2}} \frac{\Omega^2 - \Omega_i^2}{\Omega^2 - \dfrac{\Omega_0^4}{\Omega_i^2}} & , pentru \quad N = par \\[4ex] C_2 \displaystyle\prod_{i=1}^{\frac{N-1}{2}} \Omega \frac{\Omega^2 - \Omega_i^2}{\Omega^2 - \dfrac{\Omega_0^4}{\Omega_i^2}} & , pentru \quad N = impar \end{cases} \tag{4.20}$$

The rational function $F_N(\Omega)$ has the same number of poles and zeros. The poles are placed symmetrically from the zeros compared to Ω_0, defined as a geometrical average of the limit frequencies for the pass-band and the stop-band: $\Omega_0^2 = \Omega_e \Omega_b$.

Elliptical filters are considered optimal, because for the same order N and the same frequency limits Ω_e and Ω_b, they realize the minimum ripples in both bands, compared to the other analog filters.

The design data for the analog filters (example – a LPF)



We consider that:

- the effective pass area is between the maximum value 1 and the minimum value $1-\Delta_t$;
- the stop area is between 0 and Δ_b;
- $\Omega_e, \Omega_b, \Omega_t$ represent (in this order) the effective pass frequency, the effective stop frequency and the theoretical cut-off frequency, expressed in rad/second.

Usually these parameters are given in dB as the maximum attenuation in the effective pass band, a_M , and the minimum attenuation in the effective stop band, a_m:

$$a_M = -20\lg(1-\Delta_t) \tag{4.21}$$

$$a_m = -20\lg(\Delta_b) \tag{4.22}$$

***Remark*:**

The MATLAB functions `buttord`, `cheb1ord`, `cheb2ord` and `ellipord` allow to determine the minimum order and the cut-off frequency for the types of analog filters presented: *Butterworth*, *Chebyshev I*, *Chebyshev II* and *elliptic*, starting from the design specifications.

***General syntax*:**

```
[n,Wn]=function_name(Wp,Ws,Rp,Rs,'s')
```

- `function_name` can be any of the functions `buttord`, `cheb1ord`, `cheb2ord` and `ellipord`;
- `Rp` represents the ripple dimension (expressed in dB) from the pass band (the maximum attenuation in the pass band) and `Rs` represents the ripple dimension (expressed in dB) from the stop band (the minimum attenuation in the stop band);
- `Wp` and `Ws` are the limit frequencies of the pass and stop bands; they are expressed in rad/second and are greater than 1; in the case of band-pass and stop-band filters, `Wp` and `Ws` are vectors with 2 elements;
- `'s'` indicates the fact that it is an analog filter;
- it will return:

1) *the minimum order* `n` (see (4.23)) of an analog filter of the corresponding type given by `function_name`, which satisfies the design conditions imposed by `Wp`, `Ws`, `Rp`, `Rs`;
2) *the cut-off frequency* `Wn` (3 dB frequency) of the same filter; in the case of band-pass and stop-band filters, `Wn` is a vector with 2 elements, because these filters have two cut-off areas.

The order `n` of the analog filter and its cut-off frequency `Wn`, which are the output arguments of the previously presented functions, will serve as input parameters for a new class of MATLAB functions – `butter`, `cheby1`, `cheby2` and `ellip` – which have as result the coefficients of the transfer function of the respective analog filter.

In MATLAB, the transfer function of an analog IIR filter of order `n` has the form:

$$H(s)=\frac{B(s)}{A(s)}=\frac{b(1)s^{n}+b(2)s^{n-1}+......+b(n)s+b(n+1)}{s^{n}+a(2)s^{n-1}+a(3)s^{n-2}+......+a(n)s+a(n+1)} \tag{4.23}$$

`butter` – ***Analog filters of type Butterworth***

***Syntax*:**

```
[b,a] = butter(n,Wn,'s')
```

- designs an *analog low-pass filter* of order `n` with cut-off frequency `Wn` (the frequency is expressed in rad/second); it returns the line vectors `b` and `a` of length `n` + 1 which contain the coefficients of the transfer function $H(s)$ of the filter (b = [$b(1), b(2), ..., b(n), b(n+1)$], a = [$1, a(2), ..., a(n), a(n+1)$]; see (4.23)); if `Wn` is a vector with two elements, `Wn`=[`w1`,`w2`], with `w1<w2`, it will design an *analog band-pass filter* of order 2`n`, with the pass band between frequencies `w1` and `w2`; `'s'` indicates the fact that it is an analog filter.

[b,a] = butter(n,Wn,'high','s')
- designs an *analog high-pass filter* of order `n` with cut-off frequency `Wn`;

[b,a] = butter(n,Wn,'stop','s')
- `Wn` is a vector with two elements, `Wn`=[`w1`,`w2`], with `w1<w2`, it will design an *analog stop-band filter* of order 2`n`, with the stop band between frequencies `w1` and `w2`.

cheby1 - ***Analog filters of type Chebyshev I***

***Syntax*:**

```
[b,a] = cheby1(n,Rp,Wn,'s')
[b,a] = cheby1(n,Rp,Wn,'high','s')
[b,a] = cheby1(n,Rp,Wn,'stop','s')
```

- the same meaning for the input and output parameters as for the syntax in the `butter` function; the input parameter `Rp` represents the ripple dimension (expressed in dB) from the pass band (the maximum attenuation from the pass band).

cheby2 - ***Analog filters of type Chebyshev II***

***Sintaxe*:**

```
[b,a] = cheby2(n,Rs,Wn,'s')
[b,a] = cheby2(n,Rs,Wn,'high','s')
[b,a] = cheby2(n,Rs,Wn,'stop','s')
```

- the same meaning for the input and output parameters as for the syntax in the `butter` function; the input parameter `Rs` represents the ripple dimension (expressed in dB) from the stop band (the minimum attenuation from the stop band).

`ellip` - ***Analog filters of type elliptic*** **(*Cauer filters*)**

***Syntax*:**

```
[b,a] = ellip(n,Rp,Rs,Wn,'s')
[b,a] = ellip(n,Rp,Rs,Wn,'high','s')
[b,a] = ellip(n,Rp,Rs,Wn,'stop','s')
```

- the same meaning for the input and output parameters as for the syntax in the `butter` function; the input parameters `Rp` and `Rs` have the same meaning as for the functions `cheby1` and `cheby2`.

Remark:

Once we found the coefficients of the transfer function of the analog filter we can consider that steps 1 and 2 from the filter design are finished (see 4.2). The third step is to transform the transfer function of the analog filter into the equivalent transfer function of the digital filter. This last step is done according to the chosen design method.

Two of the 4 existent methods were discussed here: *the impulse response invariance method* and *the bilinear transform method*. Corresponding to these two methods, in MATLAB we have the functions `impinvar` and `bilinear` which will be presented next.

II. Transforming the transfer function of the analog filter into the equivalent transfer function of the digital filter

Consider the following expressions for:

- the transfer function of an analog filter:

$$H(s)=\frac{b(1)s^m+b(2)s^{m-1}+\dots b(m)s+b(m+1)}{a(1)s^n+a(2)s^{n-1}+\dots a(n)s+a(n+1)}= \\ =k\cdot\frac{(s-z(1))(s-z(2))\dots(s-z(m))}{(s-p(1))(s-p(2))\dots(s-p(n))} \tag{4.24}$$

- the transfer function of a digital filter:

$$H(z)=\frac{bd(1)+bd(2)z^{-1}+\dots bd(m)z^{-(m-1)}+bd(m+1)z^{-m}}{ad(1)+ad(2)z^{-1}+\dots ad(n)z^{-(n-1)}+ad(n+1)z^{-n}}= \\ =kd\cdot\frac{(1-zd(1)z^{-1})(1-zd(2)z^{-1})\dots(1-zd(m)z^{-1})}{(1-pd(1)z^{-1})(1-pd(2)z^{-1})\dots(1-pd(n)z^{-1})} \tag{4.25}$$

`impinvar` - ***The impulse response invariance method***

***Remark*: Using this method we can only design LPF and BPF** (see the theoretical presentation for details)**!**

***Syntax*:**
`[bd,ad] = impinvar(b,a,Fs)`

- the vectors `b` and `a` conain the values of the coefficients for the numerator and the denominator of the transfer function of the analog filter:
  `b` = $[b(1),b(2),\dots b(m+1)]$, `a` = $[a(1),a(2),\dots a(n+1)]$; see (4.24);
- `Fs` is the sampling frequency expressed in Hz; *if not specified, it is taken implicitly* `Fs` = 1Hz;
- *The order of the numerator cannot be higher than the order of the denominator for the transfer function of the analog filter* ($m \le n$);
- It returns the line vectors `bd` and `ad` which contain the values of the coefficients for the numerator and the denominator of the transfer function of the digital filter:
  `bd`=$[bd(1),bd(2),\dots bd(m+1)]$, `ad`=$[ad(1),ad(2),\dots ad(n+1)]$(see (4.25)).

The MATLAB function `impinvar` performs the transform from analog to discrete domain considering $h[n]=h_a(nT)$ which is different from the theoretical relation (4.7). Therefore, a multiplication by T = 1/`Fs` of the obtained `bd` vector is necessary.

`bilinear` - ***The bilinear transform method***

***Syntax*:**
`[bd,ad] = bilinear(b,a,Fs)`

- the vectors `b` and `a` conain the values of the coefficients for the numerator and the denominator of the transfer function of the analog filter:
  `b` = $\left[b(1), b(2), \ldots b(m+1)\right]$, `a` = $\left[a(1), a(2), \ldots a(n+1)\right]$; see (4.24);
- `Fs` is the sampling frequency expressed in Hz;
- *The order of the numerator cannot be higher than the order of the denominator for the transfer function of the analog filter* ($m \le n$);
- It returns the line vectors `bd` and `ad` which contain the values of the coefficients for the numerator and the denominator of the transfer function of the digital filter:
  `bd`=$\left[bd(1), bd(2), \ldots bd(m+1)\right]$, `ad`=$\left[ad(1), ad(2), \ldots ad(n+1)\right]$(see (4.25)).

`[zd,pd,kd] = bilinear(z,p,k,Fs)`

- *The column vectors* `z` and `p` contain the values of the zeros and poles of the transfer function of the analog filter and the value `k` represents the gain (`z` = $\left[z(1), z(2), \ldots z(m)\right]$, `p` = $\left[p(1), p(2), \ldots p(n)\right]$; see (4.24));
- `Fs` is the sampling frequency expressed in Hz;
- *The order of the numerator cannot be higher than the order of the denominator for the transfer function of the analog filter* ($m \le n$);
- It returns the *column vectors* `zd` and `pd` which contain the values of the zeros and poles of the transfer function of the digital filter and the value `kd` which represents the gain (`zd`= $\left[zd(1), zd(2), \ldots zd(m)\right]$, `pd` = $\left[pd(1), pd(2), \ldots pd(n)\right]$; see (4.25)).

  Table 1 presents a summary of the steps performed in order to indirectly design an IIR filter.

Table 1

Step	Matlab function
I) Design the analog filter: * find `n` and `Wn`	`[n, Wn]=` $\begin{Bmatrix}\texttt{buttord}\\ \texttt{cheb1ord}\\ \texttt{cheb2ord}\\ \texttt{ellipord}\end{Bmatrix}$ `(Wp, Ws, Rp, Rs, 's')`
* design	`[b, a]=` $\begin{cases}\texttt{butter(n, Wn,'s')}\\ \texttt{cheby1(n, Rp, Wn, 's')}\\ \texttt{cheby2(n, Rs, Wn,'s')}\\ \texttt{ellip(n, Rp,Rs, Wn, 's')}\end{cases}$

II) Transform $H_a(s)$ -> $H(z)$
a) Impulse response invariance method
$\omega = \Omega T \Leftrightarrow \Omega = \frac{\omega}{T}$
or
b) Bilinear transform method
$\Omega = \frac{2}{T}\tan\frac{\omega}{2} \Leftrightarrow \omega = 2\tan^{-1}\frac{\Omega T}{2}$
	`[bd, ad]=impinvar(b,a,Fs)`
	or
	`[bd, ad]=bilinear(b,a,Fs)`

Examples**:**

1. Design using the impulse response invariance method a digital LPF of type Butterworth, knowing that:

- at the frequency $F_e = 2\text{kHz}$ the attenuation is smaller than 1 dB, i.e., $a_M = 1\text{dB}$;
- at the frequency $F_b = 3\text{kHz}$ the attenuation is greater than 20 dB, i.e., $a_m = 20\text{dB}$;
- the sampling frequency is $F_s = 20\text{kHz}$.

2. Solve the previous problem using the bilinear transform method.

Solution 1. Determine the normalized limit frequencies of the pass and stop bands:

$$\omega_e = 2\pi\frac{F_e}{F_s} = 0,2\pi \qquad \text{and} \qquad \omega_b = 2\pi\frac{F_b}{F_s} = 0,3\pi$$

Taking into account the connection between the two frequency axes (see (4.8)) compute for the analog filter:

$$\Omega_e = \frac{\omega_e}{T_s} = \omega_e F_s = 4000\pi \quad \text{radians/second}$$

$$\Omega_b = \frac{\omega_b}{T_s} = \omega_b F_s = 6000\pi \quad \text{radians/second}$$

```
Fs=20000;
we=2*pi*2/20;
wb=2*pi*3/20;
Oe=we*Fs;
Ob=wb*Fs;
```

Now we have all the input parameters for the `buttord` function, used to determine the order and the cut-off frequency of the analog filter (see its syntax):

```
[n,Wt]=buttord(Oe,Ob,1,20,'s')     →   n =
                                           8


                                       Wt =
                                           1.4144e+004
```

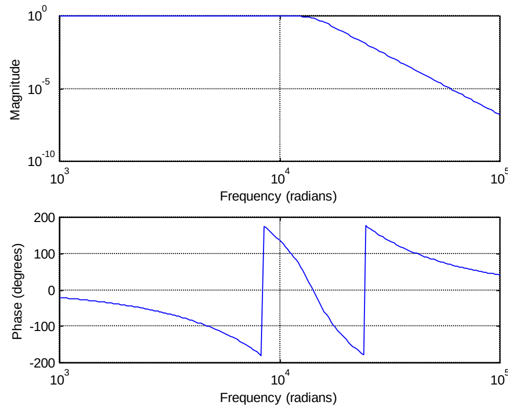
// the order of the analog filter is `n = 8` and the cut-off frequency is 14144 radians/second (it can be noticed that this value is indeed in the interval $[\Omega_e, \Omega_b]$).

The values `n` and `Wt` obtained represent the input parameters for the function `butter` which will return the coefficients of the transfer function of the analog filter (see its syntax and (4.23)):

```
[bs,as]=butter(n,Wt,'s')
bs =
  1.0e+033 *
  Columns 1 through 7
 0         0         0         0         0         0         0
  Columns 8 through 9
 0    1.6017
as =
  1.0e+033 *
  Columns 1 through 7
 0.0000   0.0000   0.0000   0.0000   0.0000   0.0000   0.0000
  Columns 8 through 9
 0.0006    1.6017
```

The amplitude-frequency and phase-frequency characteristics can be viewed using the function `freqs` (similar to `freqz`); see `help freqs`.

```
figure(1), freqs(bs,as)
```

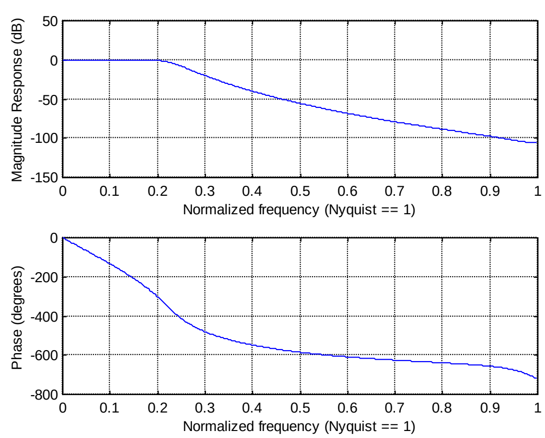


The two vectors `bs` and `as` from before will be the input arguments for the function `impinvar` which will compute the values of the coefficients of the transfer function of the desired digital filter (see its syntax and (4.24) and (4.25)):

```
[bd,ad]=impinvar(bs,as,Fs)
bd =
  Columns 1 through 4
  0.00000000000001   0.00000779825827   0.00057632592984   0.00354891171151

  Columns 5 through 8
  0.00454600814643   0.00143986570993   0.00009449298844   0.00000051475724

  Column 9
                 0
ad =
  Columns 1 through 4
  1.00000000000000  -4.47934964222873   9.26011521715720 -11.38583232119671

  Columns 5 through 8
  9.03786337833229  -4.71810555063958   1.57588822345554  -0.30701444261219

  Column 9
  0.02664905479100
```

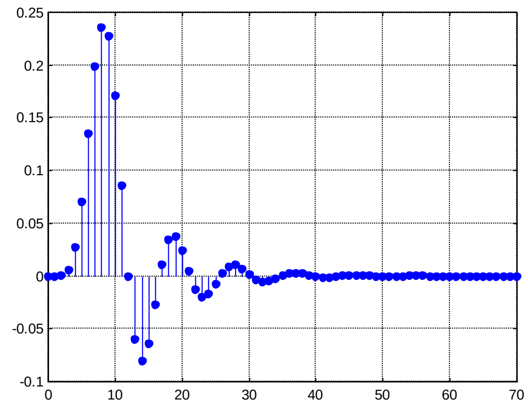
The amplitude-frequency and phase-frequency characteristics of the digital filters can be viewed using the function `freqz` (see 2.2.3. from laboratory 2):

```
figure(2), freqz(bd,ad)
```



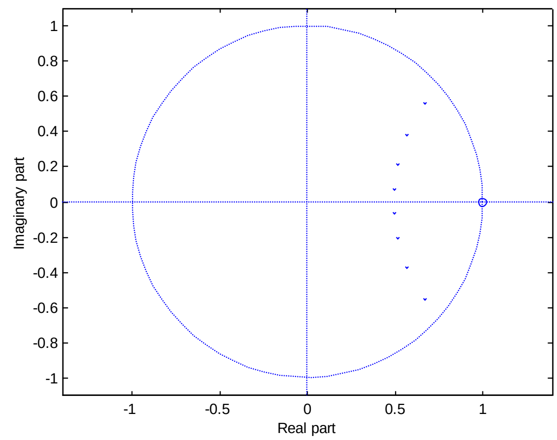
The weight function of the filter can be viewed using the command `impz` (see 2.2.1. from laboratory 2):

```
figure(3),impz(bd,ad),grid
```



In order to verify graphically the stability of the obtained digital filter (all poles must be inside the unit circle) we can use the function `zplane` (see 2.2.4. from laboratory 2), by only plotting the values of the poles of the transfer function:

```
figure(4),zplane(1,ad)
```



Solution 2. The bilinear transform method involves transforming the frequency axis according to (4.12):

$$\Omega_e = \frac{2}{T_s}\tan\frac{\omega_e}{2} = 2F_s\tan\frac{\omega_e}{2}$$

$$\Omega_b = \frac{2}{T_s}\tan\frac{\omega_b}{2} = 2F_s\tan\frac{\omega_b}{2}$$

The values for ω_e and ω_b are the same as for the previous problem.

```
Fs=20000;
we=2*pi*2/20;
wb=2*pi*3/20;
Oe=2*Fs*tan(we/2);
Ob=2*Fs*tan(wb/2);
```

The same steps as in the case of the impulse response invariance method from the previous problem are followed:

```
[n,Wt]=buttord(Oe,Ob,1,20,'s')
[bs,as]=butter(n,Wt,'s')
[bd,ad]=bilinear(bs,as,Fs)
```

It can be seen that the order of the filter is now `n` = 7, so smaller with 1 than for the impulse response invariance method.

E1. *Exercises*:

1. Design using the bilinear transform method a digital high-pass filter of type Butterworth, knowing that:
- at the frequency $F_b = 2\text{kHz}$ the attenuation is greater than 40 dB;
- at the frequency $F_e = 4\text{kHz}$ the attenuation is smaller than 1 dB;
- the sampling frequency is $F_s = 24\text{kHz}$.

Represent graphically:
- the amplitude-frequency and phase-frequency characteristics of the analog filter and of the obtained digital filter;
- the impulse response of the digital filter;
- the positioning in the **Z** plane of the poles of the digital filter transfer function.

Can we also use for the design the impulse response invariance method?

2. Solve problem 1 for the case of a Chebyshev I filter. Comment the differences.

3. Solve problem 1 for the case of a Chebyshev II filter. Comment the differences.

4. Solve problem 1 for the case of an elliptic filter. Comment the differences.

5. Consider a LPF of type Butterworth, of order n=3, having the cut-off frequency $F_C = 4\text{kHz}$. Determine the transfer function $H(z)$ of the obtained digital filter:
a) using the impulse response invariance method;
b) using the bilinear transform method.
The sampling frequency is $F_s = 40\text{kHz}$.
Represent graphically:
- the amplitude-frequency and phase-frequency characteristics of the analog filter and of the obtained digital filter;
- the impulse response of the digital filter;
- the positioning in the **Z** plane of the poles of the digital filter transfer function.

6. Solve problem 5 for the case of a Chebyshev I filter with a maximum attenuation in the pass band of 1dB. Comment the results.

7. Solve problem 5 for the case of a Chebyshev II filter with a minimum attenuation in the stop band of 40dB. Comment the results.

8. Solve problem 5 for the case of an elliptic filter with a minimum attenuation in the stop band of 40dB and a maximum attenuation in the pass band of 1dB. Comment the results.

9. Design using the impulse response invariance method a digital band-pass filter of type Butterworth, knowing that:
- at the frequencies $F_{e1} = 4\text{kHz}$ and $F_{e2} = 6\text{kHz}$ the attenuation is smaller than 1 dB;
- at the frequencies $F_{b1} = 3\text{kHz}$ and $F_{b2} = 7\text{kHz}$ the attenuation is greater than 40 dB;
- the sampling frequency is $F_s = 20\text{kHz}$.

Represent graphically:
- the amplitude-frequency and phase-frequency characteristics of the analog filter and of the obtained digital filter;
- the impulse response of the digital filter;
- the positioning in the **Z** plane of the poles of the digital filter transfer function.

Solve the problem also using the bilinear transform method.

10. Solve problem 9 for the case of a Chebyshev I filter. Comment the results.

11. Solve problem 9 for the case of a Chebyshev II filter. Comment the results.

12. Solve problem 9 for the case of an elliptic filter. Comment the results.

13. Design using the bilinear transform method a digital stop-band filter of type Butterworth, knowing that:
- at the frequencies $F_{e1} = 3\text{kHz}$ and $F_{e2} = 7\text{kHz}$ the attenuation is smaller than 1 dB;
- at the frequencies $F_{b1} = 4\text{kHz}$ and $F_{b2} = 6\text{kHz}$ the attenuation is greater than 40 dB;
- the sampling frequency is $F_s = 20\text{kHz}$.

Represent graphically:
- the amplitude-frequency and phase-frequency characteristics of the analog filter and of the obtained digital filter;
- the impulse response of the digital filter;
- the positioning in the **Z** plane of the poles of the digital filter transfer function.
- Can we also use for the design the impulse response invariance method?

14. Solve problem 13 for the case of a Chebyshev I filter. Comment the results.

15. Solve problem 13 for the case of a Chebyshev II filter. Comment the results.

16. Solve problem 13 for the case of an elliptic filter. Comment the results.

17. Design a digital LPF starting from an analog filter of type Butterwoth of order n=2 and cut-off frequency $F_t = 5\text{kHz}$. The sampling frequency is $F_s = 40\text{kHz}$. Use for the design:

a) the impulse response invariance method;

b) the bilinear transform method.

Represent graphically:
- the amplitude-frequency and phase-frequency characteristics of the analog filter and of the obtained digital filter;
- the impulse response of the digital filter;
- the positioning in the **Z** plane of the poles of the digital filter transfer function.

18. Solve problem 17 for the case of a HPF. What can you notice?

4.3. Direct design of *IIR* filters

The design methods in this category are based on a numerical optimization and allow the implementation of digital filters that approximate a certain frequency response.

The MATLAB functions `buttord`, `cheb1ord`, `cheb2ord` and `ellipord` presented before allow to determine the minimum order and the cut-off frequency for the digital filters. In MATLAB, the transfer function of a digital *IIR* filter of order `n` has the form:

$$H(z)=\frac{B(z)}{A(z)}=\frac{b(1)+b(2)z^{-1}+......+b(n)z^{-(n-1)}+b(n+1)z^{-n}}{1+a(2)z^{-1}+a(3)z^{-2}+......+a(n)z^{-(n-1)}+a(n+1)z^{-n}} \quad (4.26)$$

***General syntax*:**

`[n,Wn]=function_name(Wp,Ws,Rp,Rs)`

- `function_name` can be any of the functions `buttord`, `cheb1ord`, `cheb2ord` and `ellipord`;
- `Rp` represents the ripple dimension (expressed in dB) from the pass band (the maximum attenuation from the pass band) and `Rs` represents the ripple dimension (expressed in dB) from the stop band (the minimum attenuation from the stop band);
- `Wp` and `Ws` are the limit frequencies of the pass and stop bands; they are between 0 and 1, where 1 corresponds to half of the sampling frequency and they are computed as follows:

  Frequency[Hz] / (Fs[Hz]/2);

  In the case of band-pass and stop-band filters, `Wp` and `Ws` are vectors with 2 elements;
- it will return:
  1) *the minimum order* `n` (see (4.26)) of a digital filter of the corresponding type given by `function_name`, which satisfies the design conditions imposed by `Wp`, `Ws`, `Rp`, `Rs`;
  2) *the cut-off frequency* `Wn` (3 dB frequency) of the same filter; in the case of band-pass and stop-band filters, `Wn` is a vector with 2 elements, because these filters have two cut-off areas.

Also, the MATLAB functions `butter`, `cheby1`, `cheby2`, `ellip` allow the direct design of digital *IIR* filters (based on Chebyshev approximation directly in the digital domain). The syntaxes are similar to the ones presented in the case of

analog filters, with the difference that the input parameter `'s'` disappears (which specified that we design an analog filter) and the frequencies
`Wn` are between 0 and 1, where 1 corresponds to half of the sampling frequency, computed as before:

Frequency[Hz] / (Fs[Hz]/2).

Verify the syntaxes of these functions using the `help` command.

Table 2 presents a summary of the steps performed in order to directly design an IIR filter.

Table 2

Step	Matlab function

* find `n` and `Wn`
`Wp=2*Fe/Fs`
`Ws=2*Fb/Fs`
	`[n, Wn]=` { `buttord` / `cheb1ord` / `cheb2ord` / `ellipord` } `(Wp, Ws, Rp, Rs)`

* design	`[bd, ad]=` { `butter(n, Wn)` / `cheby1(n, Rp, Wn)` / `cheby2(n, Rs, Wn)` / `ellip(n, Rp, Rs, Wn)` }

E2. ***Exercises*:**

1. Solve problems 1-4 and 9-16 from exercises **E1** by direct design with the help of the presented MATLAB functions.