POLITECNICO
MILANO 1863

DIPARTIMENTO DI ELETTRONICA
INFORMAZIONE E BIOINGEGNERIA

# 1D Digital filters

# Filter definition in MATLAB

1. Given H(z) = B(z) / A(z):
    - [H(omega), omega] = freqz(B(z), A(z), N, 'whole'): for both FIR and IIR.
    - h(n) = filter(B(z), A(z), delta(n)): precise with FIR, only an approximation for IIR.
2. Given h(n):
    - H(k)= fft(h)

# Filter definition in MATLAB

- H(k) is defined over N samples.
- The DFT is PERIODIC:
  - In frequency domain, period = Fs, f = [0, Fs) or f = [-Fs/2, Fs/2) [Hz]
  - In angular frequency domain, period = $2\pi$Fs, $\omega$= [0, $2\pi$Fs) or

    $\omega$ =[-$\pi$Fs, $\pi$Fs) [rad/s]
  - In normalized frequency, period = 1, $\tilde{f}$= [0, 1) or $\tilde{f}$= [-0.5, 0.5)
  - In normalized angular frequency, period = $2\pi$, $\tilde{\omega}$= [0, $2\pi$) or $\tilde{\omega}$= [-$\pi$,$\pi$)

*How to relate the MATLAB result with the actual Fourier spectrum?*

→ *How to express MATLAB samples as real frequencies in Hz or normalized frequencies?*

# MATLAB metrics conversion

The N-th sample in MATLAB corresponds to the maximum frequency over one period of the periodic spectrum.

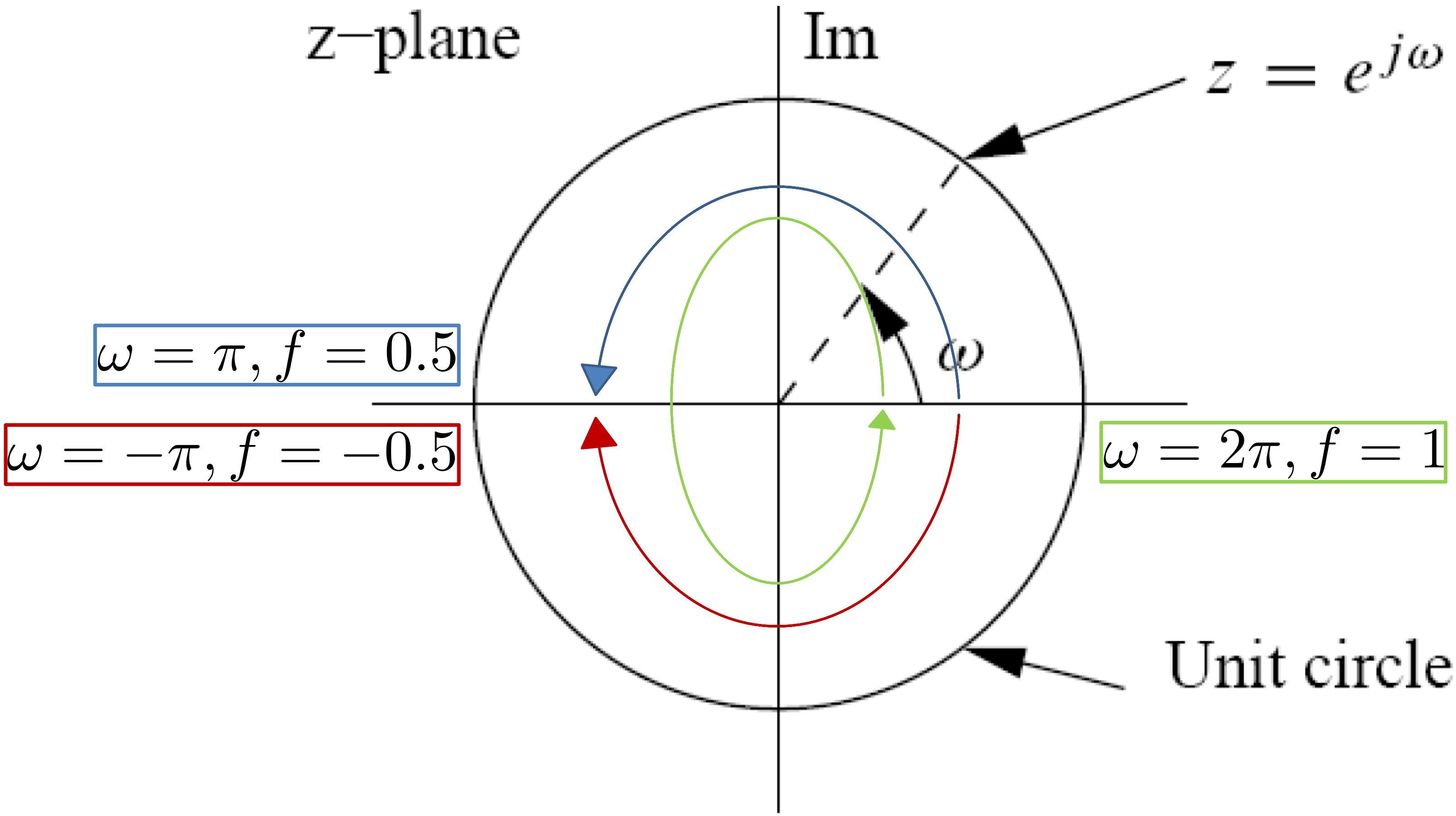
↓

Given the array of MATLAB samples: $\mathbf{n} = [0, 1, 2, ..., N-1]$

- The frequency axis [Hz] = [0, Fs) is obtained as $\mathbf{n} \cdot \frac{\mathrm{F}_s}{N}$
- The angular frequency axis [rad/s] = $[0, 2\pi\mathrm{F}_s)$ is obtained as $\mathbf{n} \cdot \frac{2\pi\mathrm{F}_s}{N}$
- The normalized frequency axis = [0, 1) is obtained as $\mathbf{n} \cdot \frac{1}{N}$
- The normalized angular frequency axis = $[0, 2\pi)$ is obtained as $\mathbf{n} \cdot \frac{2\pi}{N}$

**LEARNING BY HEART IS NOT NEEDED!** *Just think at the units of measure*

# From Z domain to normalized frequency domain




*To pass from normalized domain to real frequency domain, multiply by Fs*

# FIR vs IRR filters in MATLAB

Given a FIR filter and an input signal x(n),

the output y(n) is obtained by:

- Function 'conv' if filter is expressed in time domain
- Function 'filter' if you have H(z) or h(n)
- The product of 'fft's in frequency domain
- The product of signal 'fft' and filter 'freqz' in f domain

Given an IIR filter and an input signal x(n),

the output y(n) is obtained by:

- You cannot use 'conv'! The result will be just an approximation because the filter has infinite duration
- Function 'filter' if you have H(z)

# Zeros and poles recall

$$H(z) = \frac{\sum_{k=0}^{N} b_k z^{-k}}{\sum_{k=0}^{D} a_k z^{-k}} = z^{D-N} \frac{b_0}{a_0} \frac{\prod_{i=1}^{N}(z - z_i)}{\prod_{i=1}^{D}(z - p_i)}$$

$$= \frac{b_0}{a_0} \frac{\prod_{i=1}^{N}(1 - z_i z^{-1})}{\prod_{i=1}^{D}(1 - p_i z^{-1})}$$

- $z_i$ = roots of numerator, called 'zeros'
- $p_i$ = roots of denominator, called 'poles'

# Zeros and poles recall: *the poles*

- The poles are associated with the autoregressive part of the filter → they generate IIR filters.
- The filter amplitude response enhances frequencies which are near the poles.
- If poles are outside the unit circle and the filter is causal, the system is unstable.

# Zeros and poles recall: *the zeros*

- The zeros are associated with the moving average part of the filter → they generate FIR filters
- The filter amplitude response attenuates frequencies which are near the zeros
- Zeros influence also the phase of the filter:
  - Minimum phase zeros if $z < 1$
  - Maximum phase zeros if $z \geq 1$

# Filter design using zeros&poles

- Place poles close to the unit circle in frequencies that must be emphasized
- Place zeros according to the desired phase response
    - The closer they are to the unit circle, the higher the frequency attenuation

# Filter design using zeros&poles

Open ‘zpgui.m’

POLITECNICO
MILANO 1863

DIPARTIMENTO DI ELETTRONICA
INFORMAZIONE E BIOINGEGNERIA

# Remarkable LTI filters

# Magnitude square function

- The magnitude response of a LTI system is:

$$M(f) = |H(f)|^2 = H(f) \cdot H^*(f) = H(z) \cdot H^*(z^{-1})\Big|_{|z|=1}$$

- Given a generic rational transfer function

$$H(z) = \frac{b_0}{a_0} \frac{\prod_{i=1}^{N}(1 - z_i z^{-1})}{\prod_{i=1}^{D}(1 - p_i z^{-1})}$$

$$\Downarrow$$

$$M(z) = H(z)H^*(z^{-1}) = \frac{|b_0|^2}{|a_0|^2} \frac{\prod_{i=1}^{N}(1 - z_i z^{-1})(1 - z_i^* z)}{\prod_{i=1}^{D}(1 - p_i z^{-1})(1 - p_i^* z)}$$

# Magnitude square function

$$M(z) = H(z)H^*(z^{-1}) = \frac{|b_0|^2}{|a_0|^2} \frac{\prod_{i=1}^{N}(1 - z_i z^{-1})(1 - z_i^* z)}{\prod_{i=1}^{D}(1 - p_i z^{-1})(1 - p_i^* z)}$$

- For each zero $z_i$ of H(z), there is another zero at $\frac{1}{z_i^*}$
- For each pole $p_i$ of H(z), there is another pole at $\frac{1}{p_i^*}$
- M(z) presents poles and zeros in conjugate reciprocal pairs

# Magnitude square function

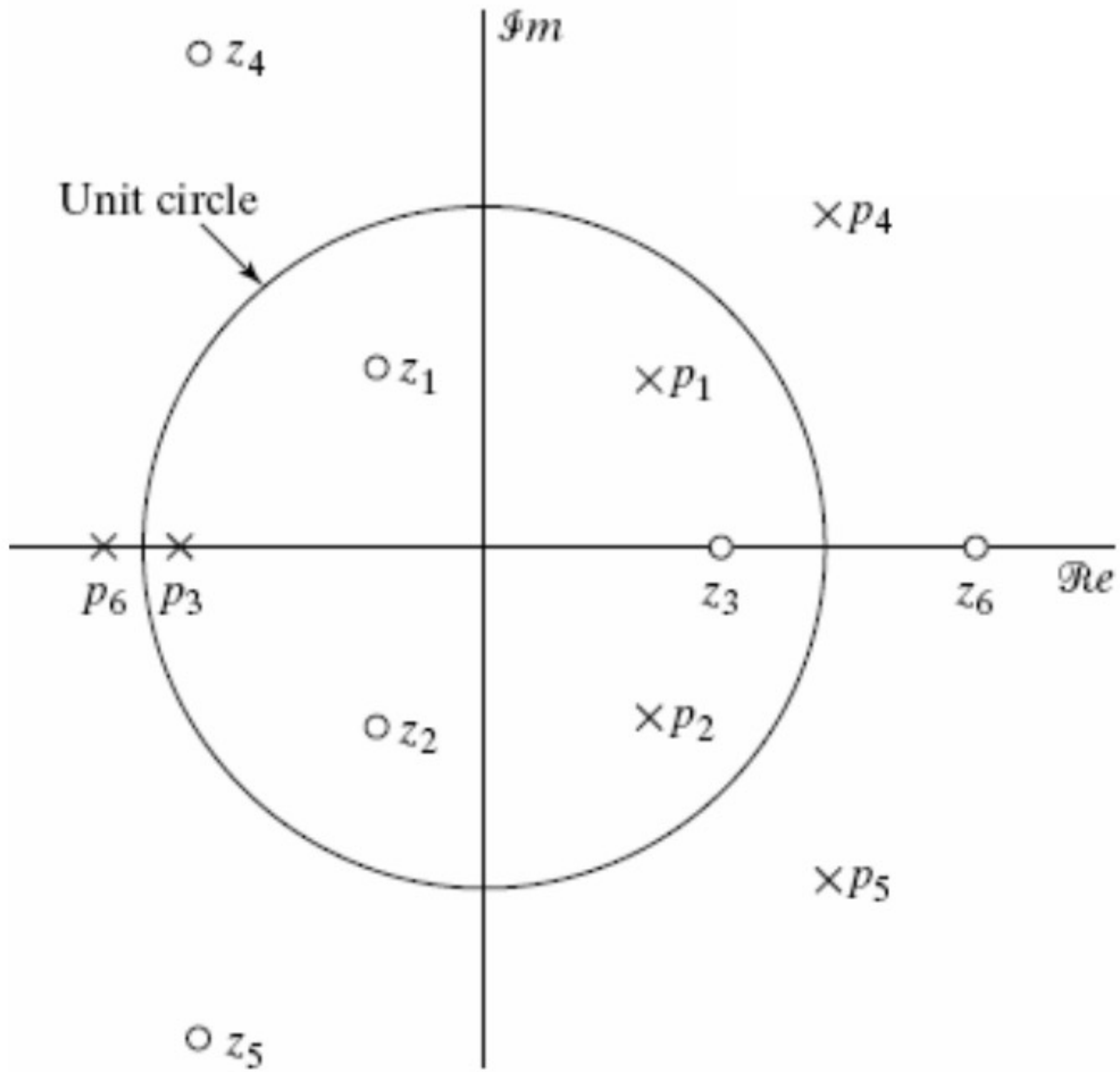
- Given a magnitude response requirement M(z) for H(z)
- Given stability and causality requirements for H(z)

- The poles of H(z) are those of M(z) inside the unit circle and are uniquely identified
- The zeros of H(z) are **not** uniquely identified
- Given a causal FIR filter H(z) of order N, it has the same magnitude response M(z) of the causal FIR filter:

$$G(z) = z^{-N} H^*(z^{-1})$$

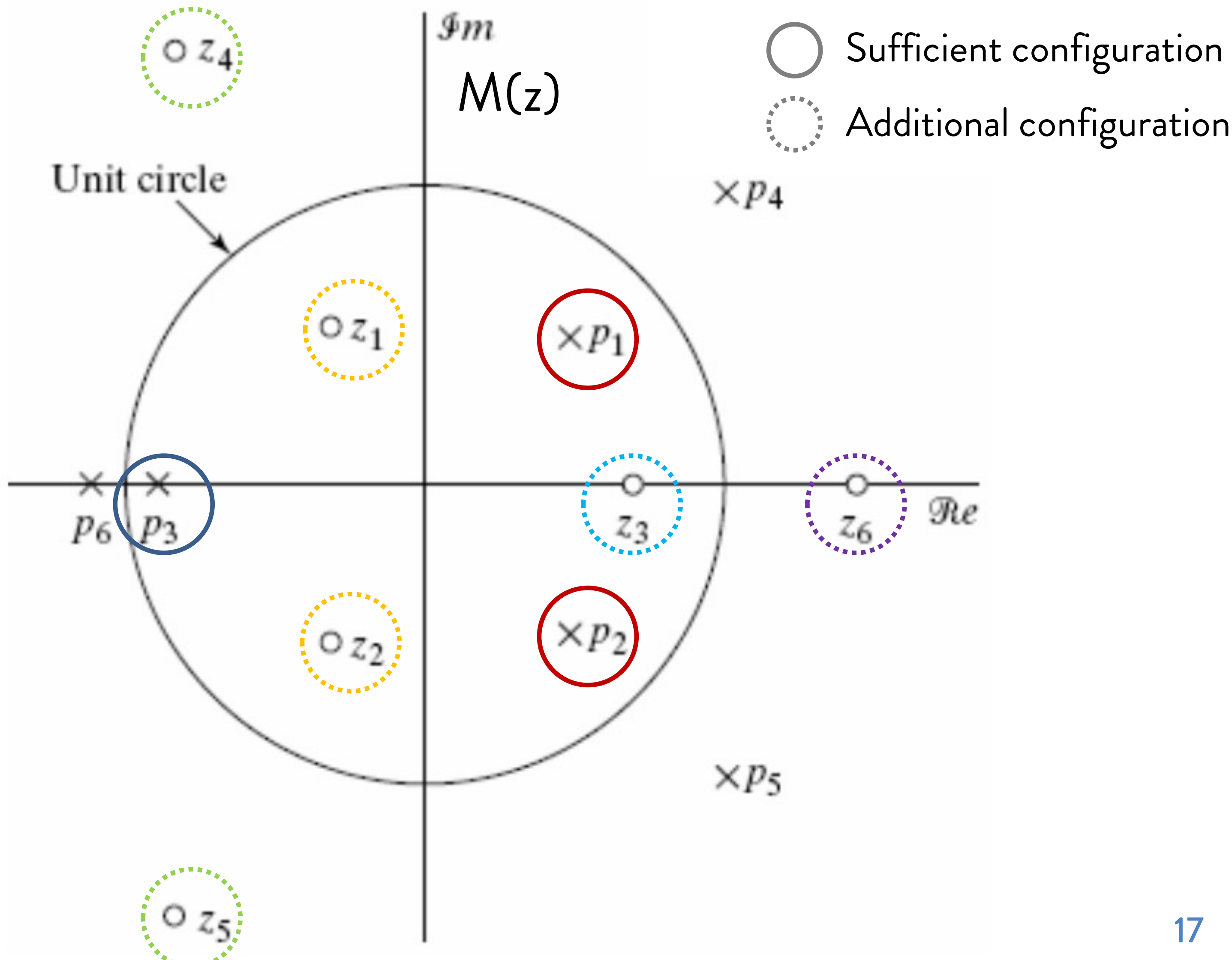
# From M(z) to H(z)

*How to get a **causal stable** system with **real** coefficients?*



# From M(z) to H(z)

*How to get a **causal stable** system with **real** coefficients?*



# Ex 20: magnitude response

- Given the filters:

$$H_1(z) = \frac{2(1 - z^{-1})(1 + 0.5z^{-1})}{(1 - 0.8e^{j\pi/4}z^{-1})(1 - 0.8e^{-j\pi/4}z^{-1})}$$

$$H_2(z) = \frac{(1 - z^{-1})(1 + 2z^{-1})}{(1 - 0.8e^{j\pi/4}z^{-1})(1 - 0.8e^{-j\pi/4}z^{-1})}$$

- Derive A(z) and B(z) for $H_1(z)$ and $H_2(z)$
- Plot the zeros and the poles in the Z-plane using 'zplane'
- Plot in the same figure the magnitude responses as a function of normalized omega in [0, 2pi), using N = 1024 samples
- How are the magnitudes related? Why?

# Allpass filters

Allpass filters are designed to have constant gain and any phase response:

$$|H_{ap}(f)| = |H_{ap}(z)|\Big|_{|z|=1} = 1$$

- Given the previous considerations, a generic causal allpass filter is:

$$H_{ap}(z) = z^{-K} e^{j\phi} \frac{A(z)}{\tilde{A}(z)}, \quad K \geq 0$$

where

$$A(z) = 1 + a_1 z^{-1} + a_2 z^{-2} + \ldots + a_N z^{-N}$$

$$\tilde{A}(z) = z^{-N} A^*(z^{-1}) = a_N^* + a_{N-1}^* z^{-1} + \ldots + a_2^* z^{2-N} + a_1^* z^{1-N} + z^{-N}$$

# Allpass filters

- Given an allpass filter:

$$H_{ap}(z) = \frac{1 + a_1 z^{-1} + a_2 z^{-2} + \ldots + a_N z^{-N}}{a_N^* + a_{N-1}^* z^{-1} + \ldots + a_2^* z^{2-N} + a_1^* z^{1-N} + z^{-N}}$$

a general form to represent an **allpass real** valued impulse response is:

$$H_{ap}(z) = c_0 \prod_{k=1}^{M_r} \frac{z^{-1} - d_k}{1 - d_k z^{-1}} \prod_{k=1}^{M_c} \frac{(z^{-1} - e_k)(z^{-1} - e_k^*)}{(1 - e_k^* z^{-1})(1 - e_k z^{-1})}$$

*Zeros and poles occur in conjugate reciprocal pairs*

# Allpass filter properties

- The cascade of two allpass filters is again an allpass filter
- Each pole of an allpass system is associated with a conjugate reciprocal zero
- The magnitude of many cascaded allpass filters is always the same

# Ex 21: allpass systems

- Write a MATLAB function ‘allpass.m’ like this:

  ‘[z_out, p_out, b_out, a_out] = allpass(b,a)’
- Inputs: b, a = numerator and denominator of H(z)
- Outputs: z_out, p_out, b_out, a_out = zeros, poles, numerator, denominator of the allpass transfer function related to H(z)
- Use the function ‘allpass’ to compute the allpass transfer function $H_{ap}(z)$ related to the causal filter $H(z) = \dfrac{1+3z^{-1}}{1-0.5z^{-1}}$
- Plot the amplitude of $H_{ap}(f)$ vs normalized frequencies in [0, 1), using N = 512 samples
- Is the filter stable? How do you expect the phase to behave?

# Minimum phase filters

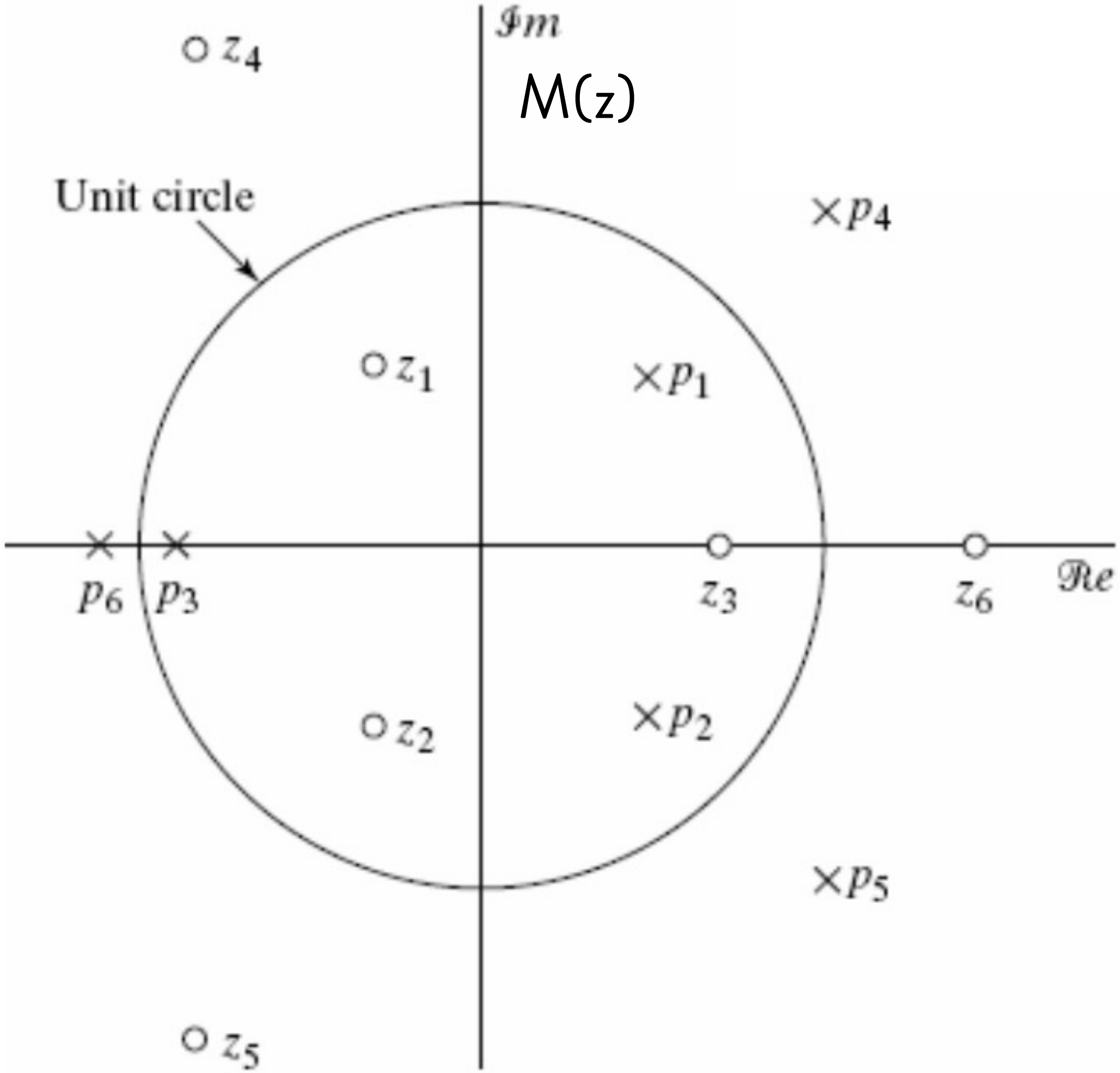
- Minimum phase filters are such that both H(z) and 1/H(z) are stable and causal

⬇

- The poles must be inside the unit circle
- The zeros must be inside the unit circle
- Given a square magnitude response M(z), there is a **unique** system whose zeros and poles are inside the unit circle and it is called minimum phase system

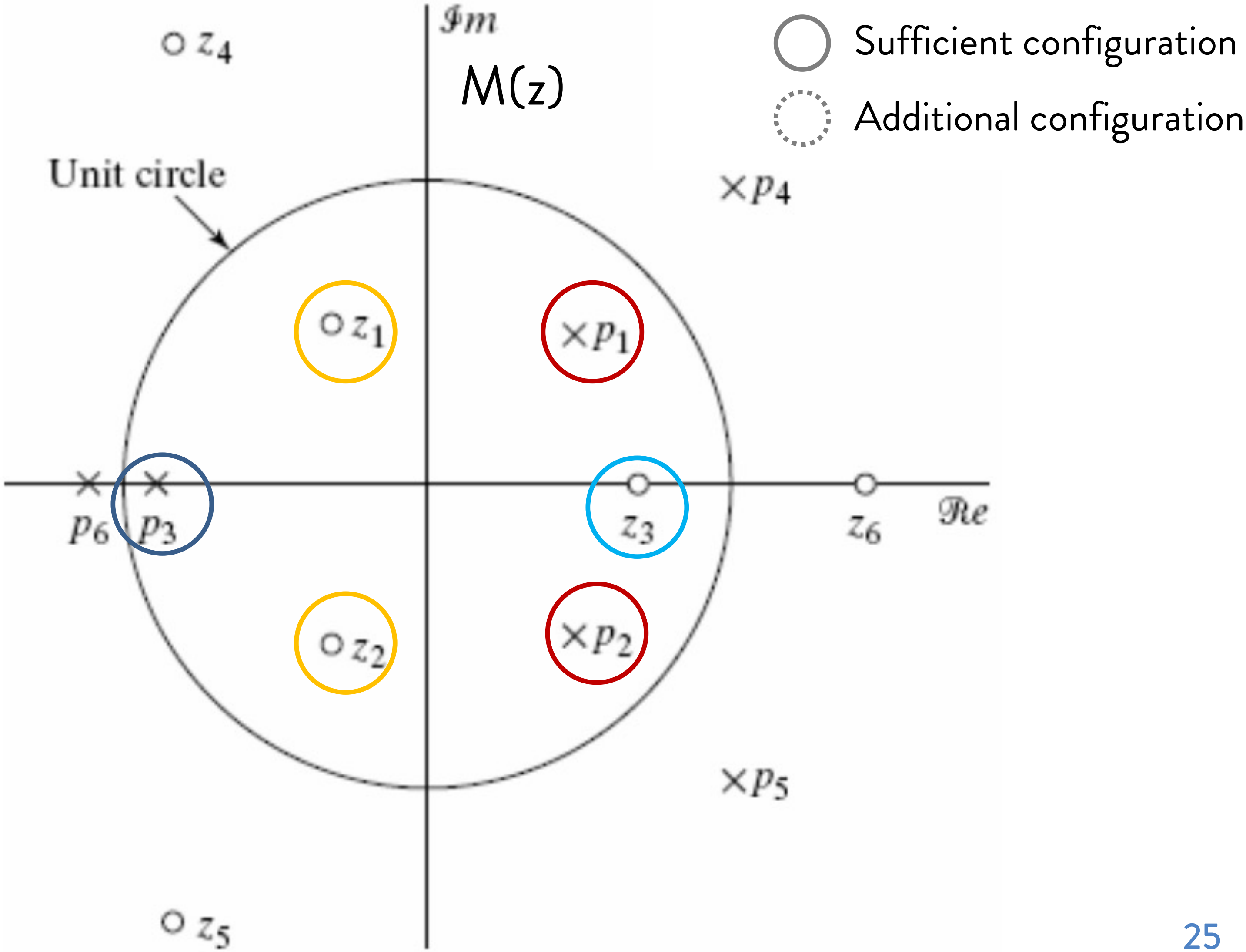
# From M(z) to H(z)

*How to get a **minimum phase** system with real coefficients?*



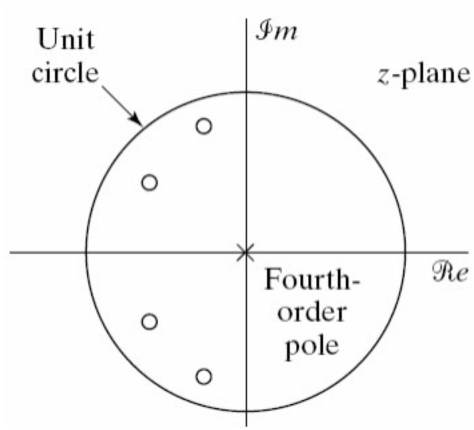
# From M(z) to H(z)

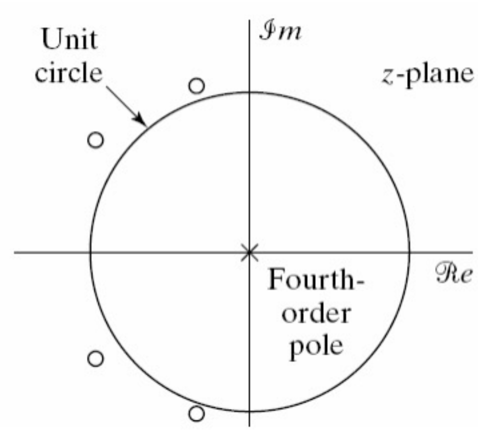
*How to get a **minimum phase** system with real coefficients?*

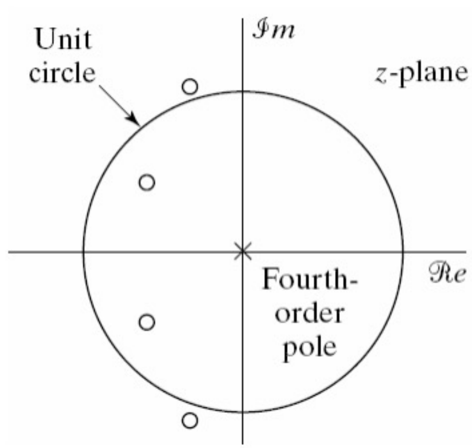


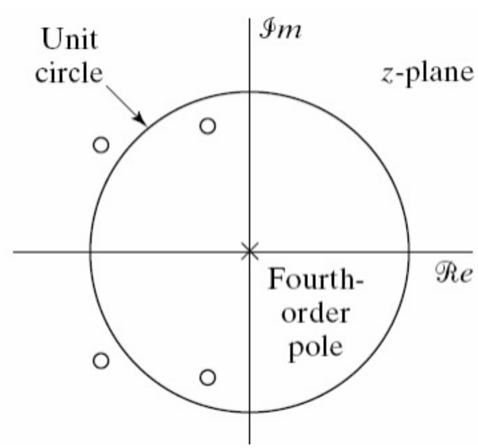
# From M(z) to H(z)

*Which is the **minimum phase** system?*



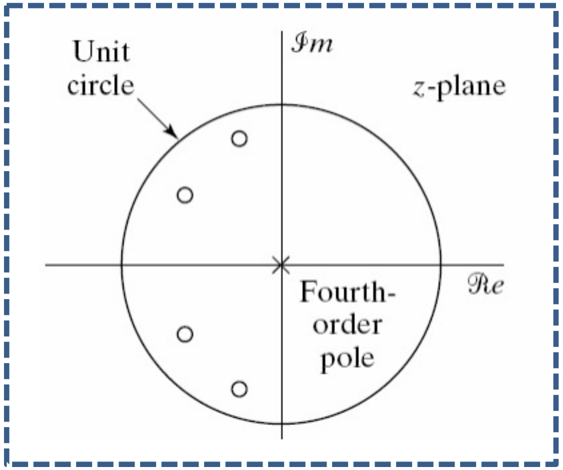



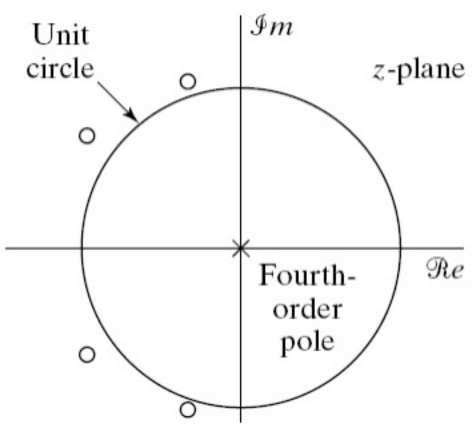



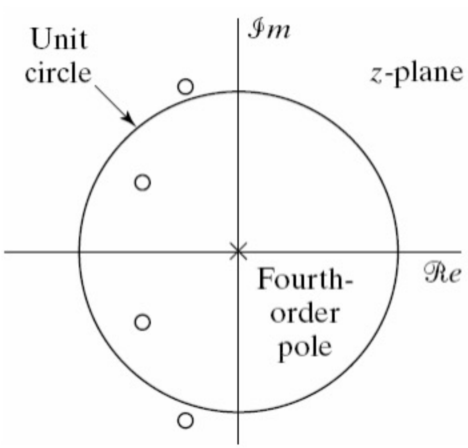



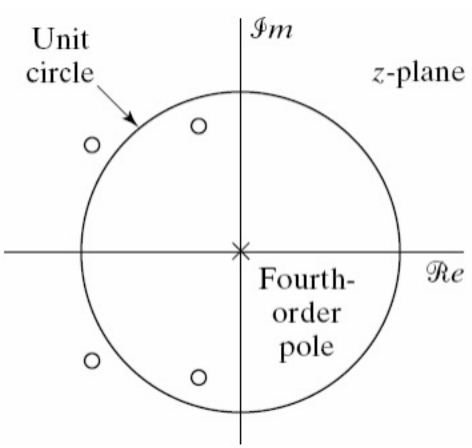
# From M(z) to H(z)

*Which is the **minimum phase** system?*



# Ex 22: minimum phase systems

- Write a MATLAB function 'typeOfFilter(b, a)' that receives as input the numerator and denominator coefficients of a causal filter H(z)=B(z)/A(z) and it returns:
  - -1 if the filter is not stable
  - 1 if the filter is stable and it is minimum phase
  - 0 if the filter is stable but it is not minimum phase
- If you test this function on a FIR filter, which can be the possible outputs?
- Test the function on $H(z) = \dfrac{1 - 2z^{-1} - 0.5z^{-3} + 0.2z^{-4}}{1 + 0.08z^{-1} + 2z^{-3}}$

# Properties of Allpass – Minimum phase filters

Any rational causal stable system can be decomposed into the multiplication of a minimum phase system and an allpass system

$$H(z) = H_{min}(z)H_{ap}(z)$$

- $H_{min}(z)$ contains:
  - the poles and zeros of H(z) that lie inside the unit circle
  - zeros that are conjugate reciprocals of the zeros of H(z) lying outside the unit circle.
- $H_{ap}(z)$ contains:
  - all the zeros of H(z) that lie outside the unit circle
  - poles which are conjugate reciprocals of the zeros of H(z) lying outside the unit circle

# Ex 23: Allpass-minimum phase conversion

- Given the filter with B(z)=[1, -1.98, 1.77, -0.17, 0.21, 0.34], A(z)=[1, 0.08, 0.40 ,0.27]
- Compute the allpass-minimum phase decomposition of H(z)
- Check the results using ‘zplane’
- Plot the amplitude of $H_{ap}(f)$ (DTFT) vs normalized angular frequencies in [0, 2pi), using N = 1024 samples
- Plot the first 50 samples of $h_{min}(n)$

# Digital filters' design

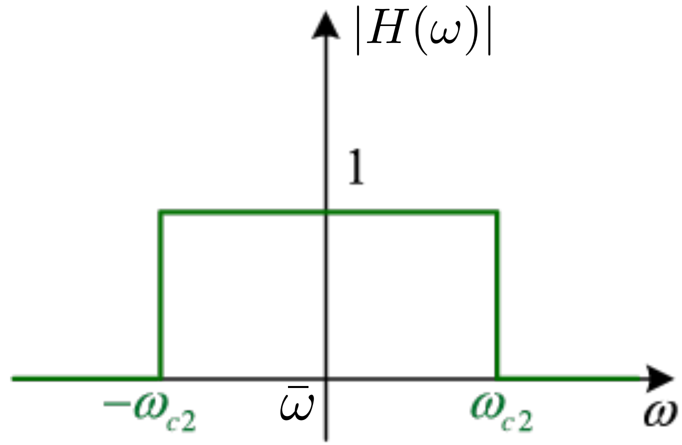
# How to design filters

1. Specify always the characteristics of the filter in frequency domain, not in time domain (e.g., lowpass, highpass, bandpass..)
2. Approximate these properties using a discrete-time system → find the filter coefficients
3. Realize the system using finite precision arithmetic

# Ideal filter

Ideal filter:

- low-pass if $\bar{\omega} = 0$
- band-pass if $0 < \bar{\omega} < \pi$
- high-pass if $\bar{\omega} = \pi$

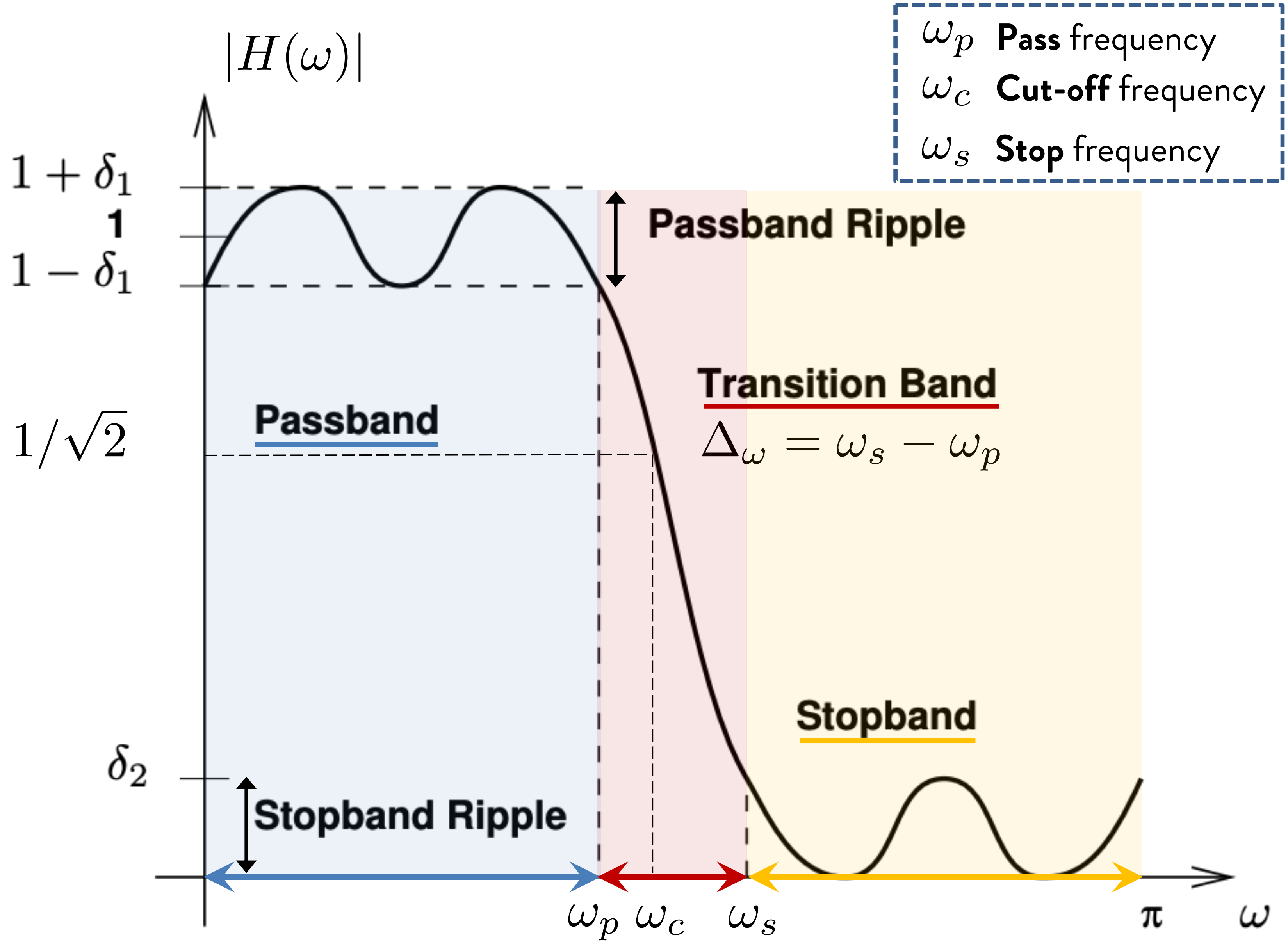



The impulse response of this filter is ≈ the sinc function.

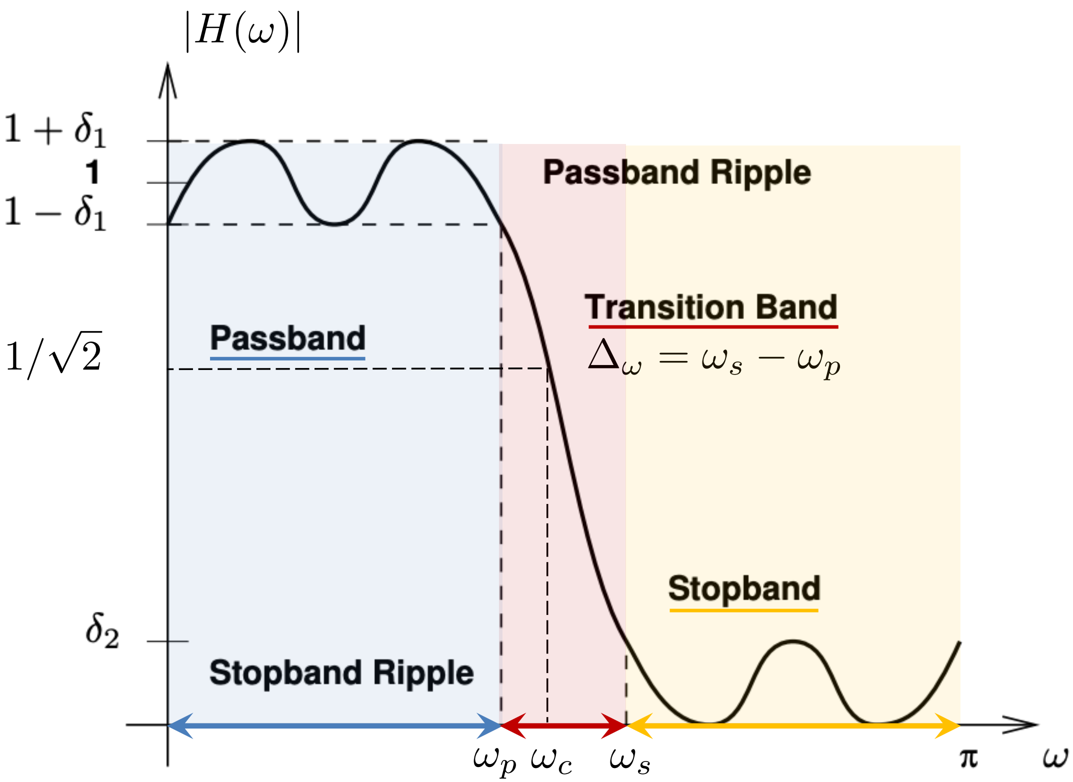
It is non-causal with an infinite delay →

Real systems can only approximate it

# Real filters



# Real filters




$\delta_1$ Peak passband ripple

$1-\delta_1$ Minimum passband gain

$\delta_2$ Peak stobband ripple

$\omega_c$ 3dB or cut-off frequency

*Towards ideal filters:*

- Peak ripple → 0
- Transition band → 0

# IIR vs FIR filters

FIR:

- Only zeros
- Always stable
- Can be linear phase
- It should be high order for best performances

IIR:

- Poles and zeros
- May be unstable
- Difficult to control phase
- Lower order (1/10-th of FIR) for high performances

# IIR vs FIR

FIR:

- Only zeros
- Always stable
- Can be linear phase
- It should be high order for best performances

IIR:

- Poles and zeros
- May be unstable
- Difficult to control phase
- Lower order (1/10-th of FIR) for high performances

We saw IIR design with poles&zeros

*How to design FIR filters?*

# FIR filter design: windowing method

The ideal filter has an infinite time duration and infinite delay.

Idea: obtain a FIR filter by truncating an infinite duration impulse response

- Given the ideal $h_i(n)$, build $h(n) = h_i(n)w(n)$
- $w(n)$ is a finite duration window

  → in frequency domain, product becomes convolution
- $H(f)$ is a blurred version of the ideal filter $H_i(f)$

# FIR filter design: windowing method

How to choose the window?

- As short as possible (in time) to minimize the cost of the FIR filter
- As narrow as possible in frequency to approach the ideal filter

# FIR filter design: windowing method

How to choose the window?

- As short as possible (in time) to minimize the cost of the FIR filter
- As narrow as possible in frequency to approach the ideal filter

# FIR filter design: windowing method

How to choose the window?

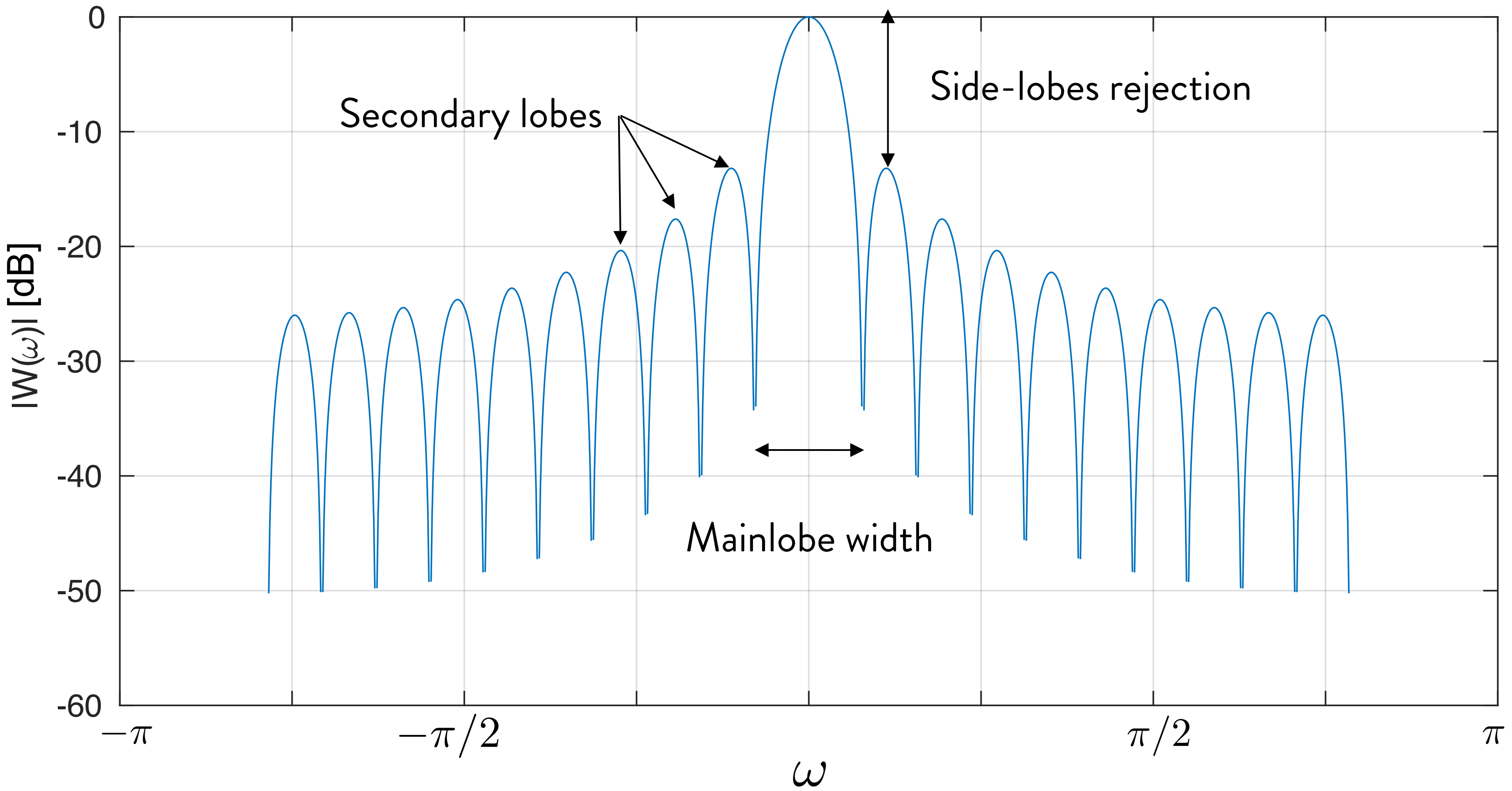
- As short as possible (in time) to minimize the cost of the FIR filter
- As narrow as possible in frequency to approach the ideal filter

Even though these requirements conflict each other, a good window is defined as the one introducing the minimum distortion.

→ Without considering the filter cost, $W(f)$ should look like a $\delta(f)$:

- its energy must be concentrated around $f = 0$
- $W(f)$ should decay fast as frequency increases

# FIR filter design: windowing method



# FIR filter design: windowing method

Every window is characterized by:

- Main-lobe width: it decreases as the window length increases
- Side-lobes rejection: ratio between the main-lobe peak and 1° secondary lobe peak [dB]
- Side-lobes roll off: asymptotic decay of the side-lobe peaks vs frequency octave [dB/octave] or frequency decade [dB/decade]

Examples of windows:

- Rectangular → 'rectwin' in MATLAB
- Hanning → 'hann' in MATLAB
- Hamming → 'hamming' in MATLAB
- Blackman → 'blackman' in MATLAB and many others…

# Window design in MATLAB

- You can use the function ‘window’ to design windows:

  ‘w = window(@window_name, Nsamples)’

- Otherwise, you can call specific functions named as the window, for instance:
  - ‘rect_w = rectwin(Nsamples)’
  - ‘hamming_w = hamming(Nsamples)’
  - ‘hann_w = hann(Nsamples)’
  - …

# FIR design in MATLAB

'fir1' is used to implement window-based FIR filter design

h = fir1(filter_order,cut-off,filter_type,window_type(filter_order+1))

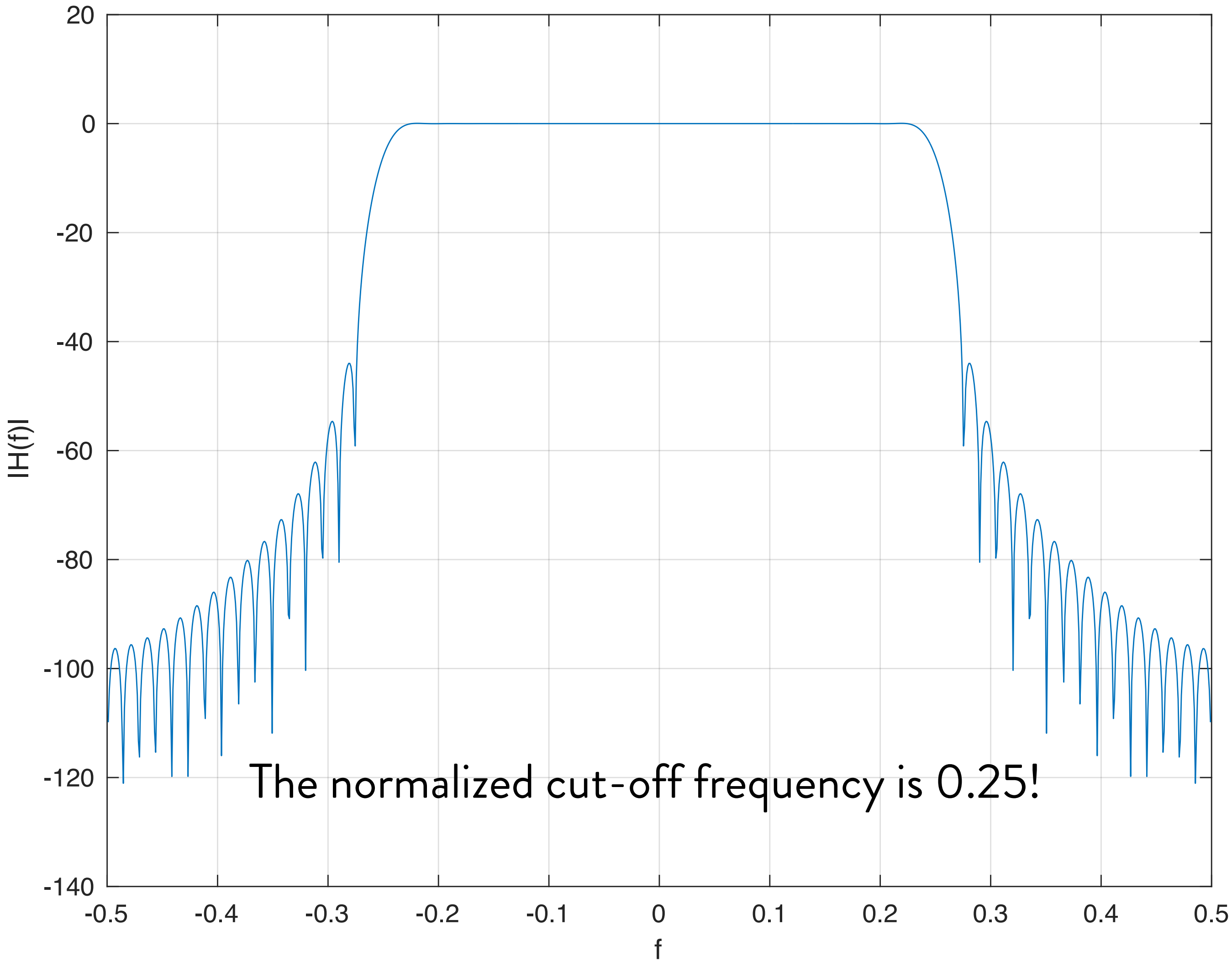
NB:

- 'cut-off' sets the 6dB point (when $|H(f)|$ is 6dB lower than the maximum peak)
- 'cut-off' for MATLAB is between 0 and 1, but 1 corresponds to half the sampling frequency!

  MATLAB Cut-off = 1 ←→ normalized frequency = 0.5
- 'filter_order' corresponds to the number of samples - 1

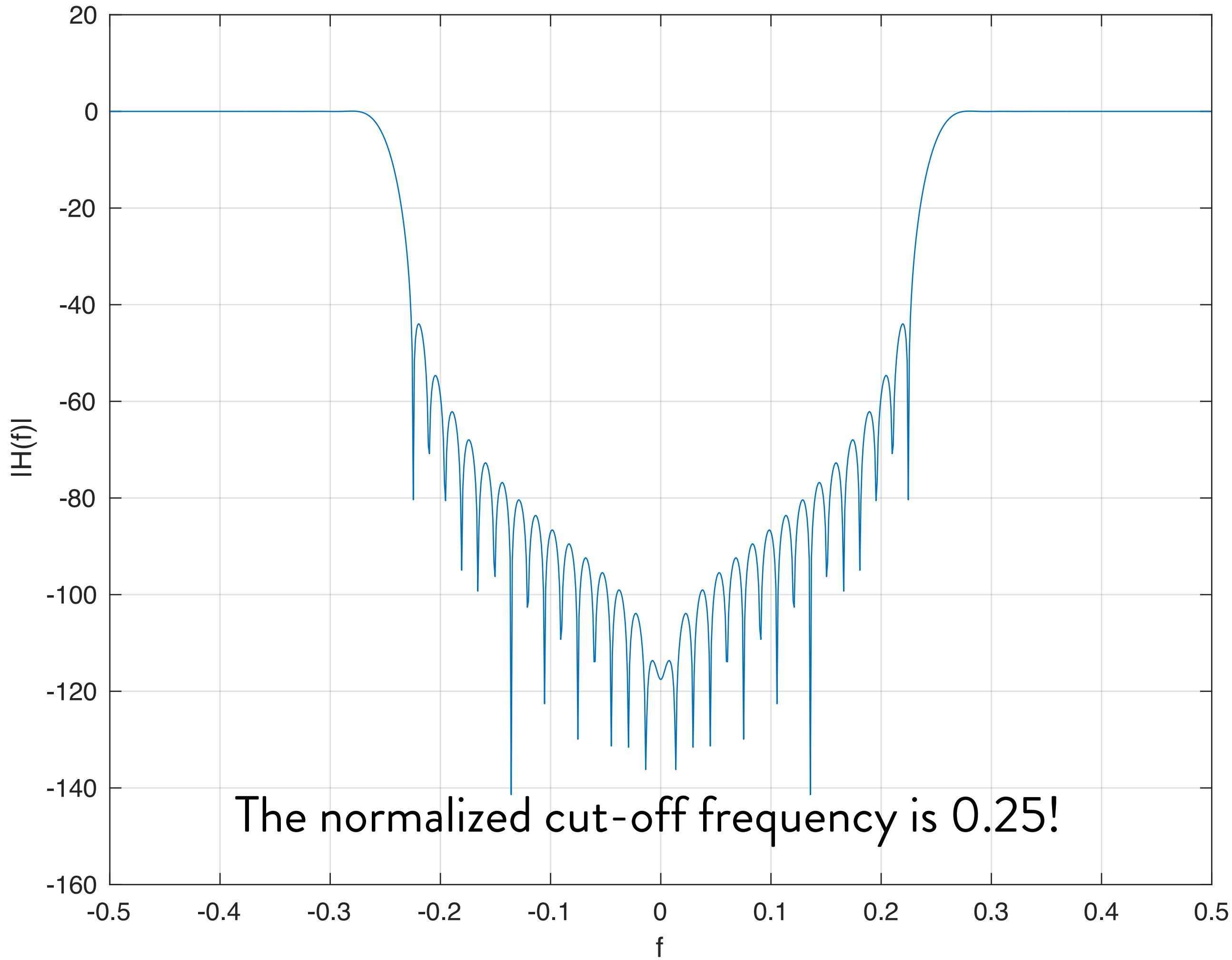
# FIR design in MATLAB:fir1

Low-pass: ‘h = fir1(66, 0.5, hann(67))’



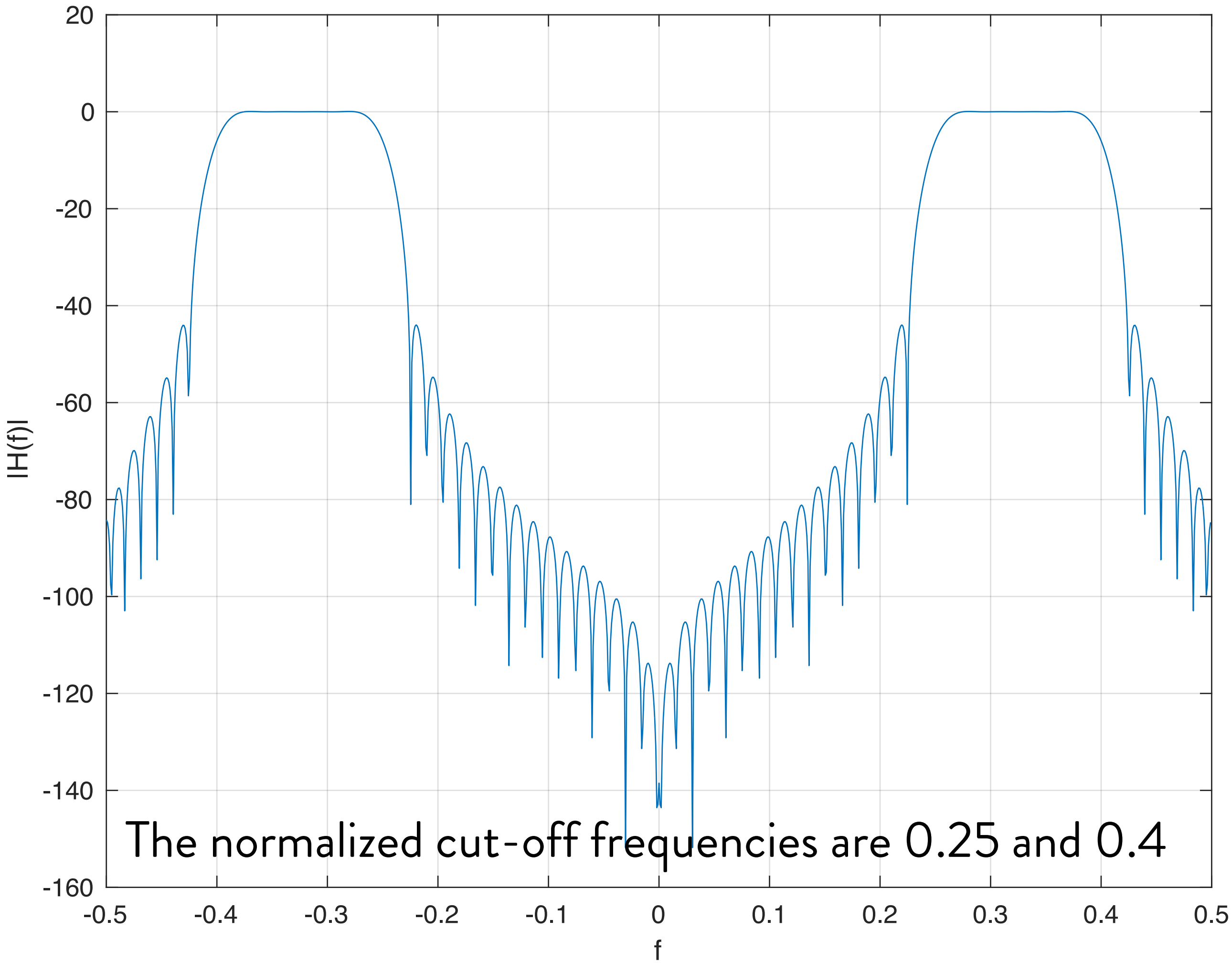
# FIR design in MATLAB:fir1

High-pass: ‘h = fir1(66, 0.5, ‘high’, hann(67))



# FIR design in MATLAB:fir1

Band-pass: ‘h = fir1(66, [0.5, 0.8], hann(67))




The normalized cut-off frequencies are 0.25 and 0.4

# Ex 24: windowing

- Given x as a cosine wave sampled at Fs = 8KHz, defined from 0 to 1 second, amplitude 1.5, frequency 1.1KHz, phase 45 deg
- Compute y as x filtered with a low-pass filter with normalized cut-off frequency of 0.4 and 64 samples
- Plot the amplitude of H(f) vs normalized frequencies in [-0.5, 0.5)
- Apply a Hanning window to select the first 512 samples of y
- Plot the amplitude of the DFT of the windowed y vs frequency in Hz, defined in [-Fs/2, Fs/2).
- If you change the cut-off frequency to 0.05, what do you expect to see in the spectrum of y?