DSP: Part II

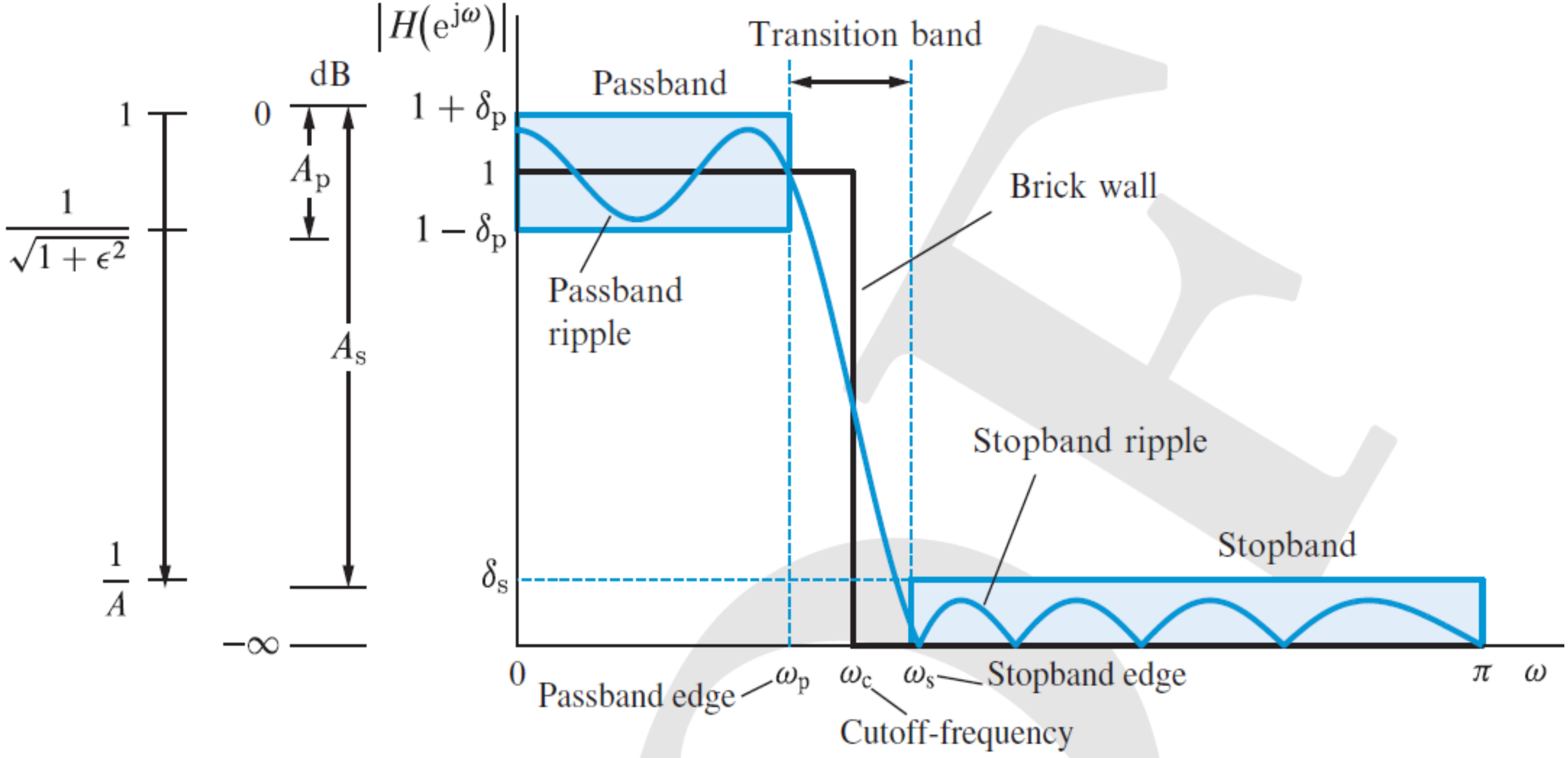
Lecture.3.1

Finite Impulse Response (FIR) Filter
(Filter design)

Methodology/Logic for FIR

1) Digital FIR filters
2) Linear Phase FIR Filters
3) Analysis: From impulse response to H(z) pole/zero plot and then to frequency response H(ω) (Reviewing Z-transform and Fourier Transform)
4) Design of FIR Filters – using the windowing technique
5) Design of FIR Filters using MATLAB.

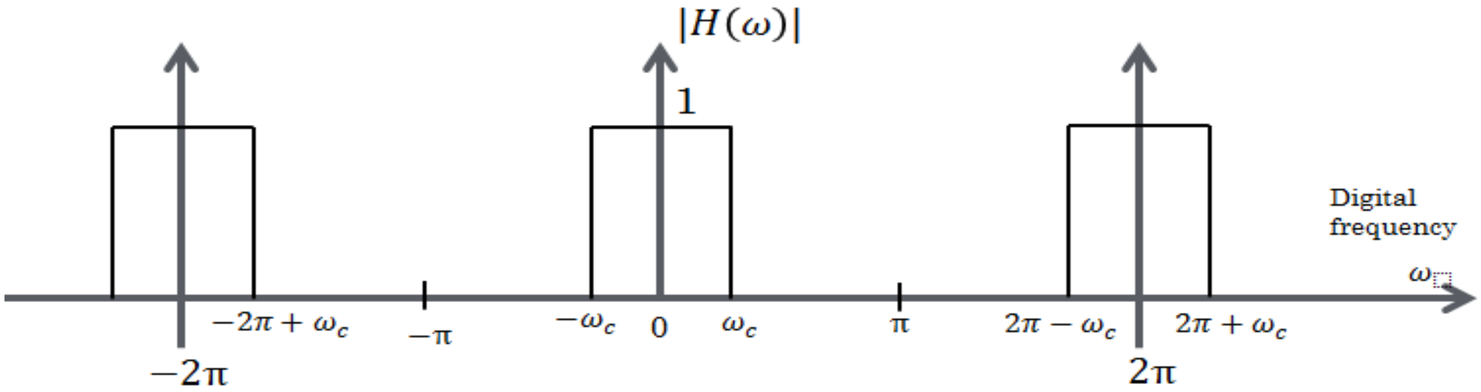
4. Specification of the filter's magnitude response



4) Design of FIR Filter – using windowing technique

FIR filter design: Inverse Fourier Transform of the target H(ω) to get h(n)

a)First begin with an ideal frequency response (e.g, low pass filter characteristic) in the digital frequency domain ω,



b)To get the impulse response (in discrete time), perform inverse Discrete Time Fourier Transform, because DTFT(h(n)) = H(ω).

- DTFT is used when the input signal is discrete, e.g, h(n) and the resultant H(ω) is periodic and continuous.
- This is what we want, impulse response h(n) must be discrete to be implemented as a digital filter.

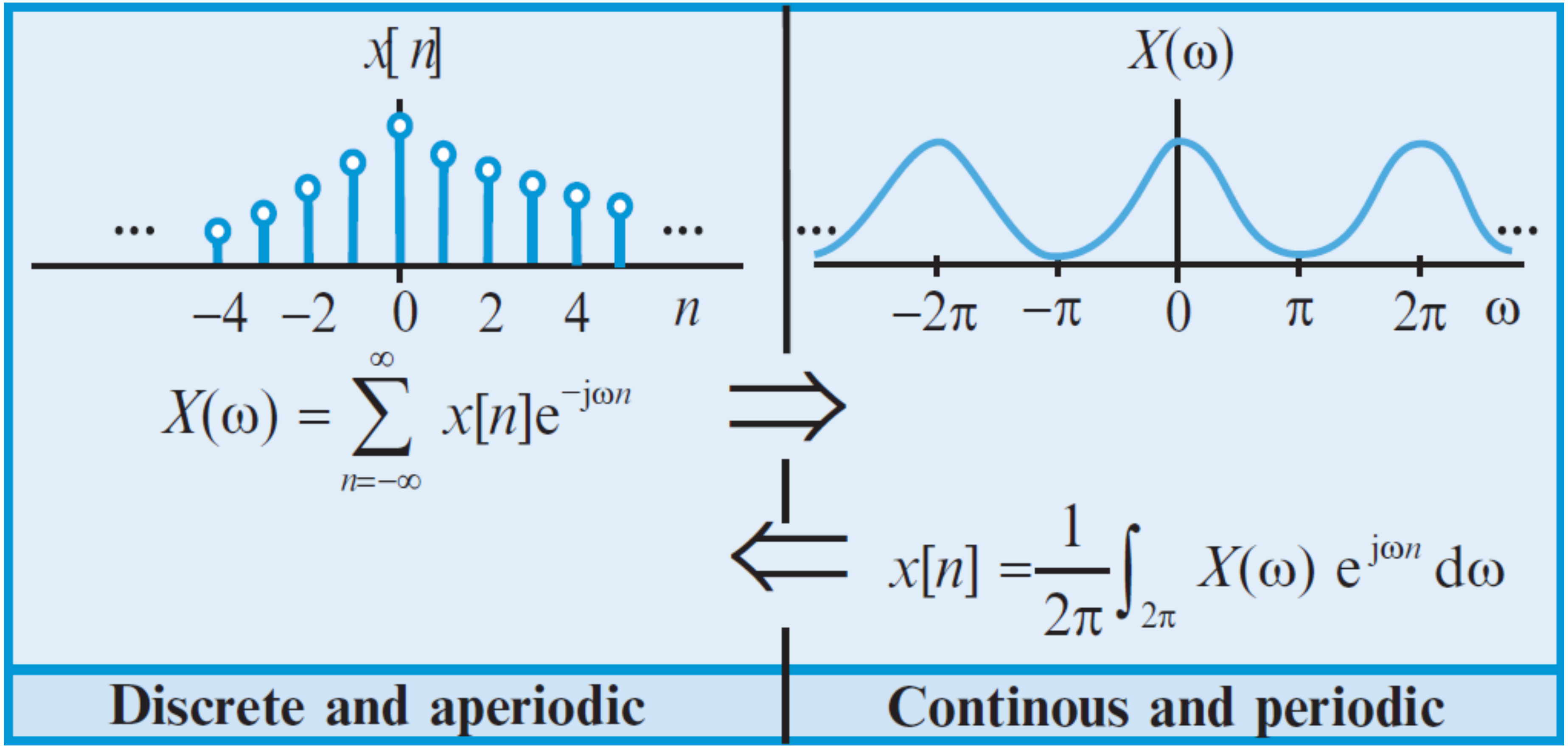
4) Design of FIR Filter – using windowing technique

Fourier Transform of discrete signal x(n):

$$X(\omega) = \sum_{n=-\infty}^{\infty} x(n) e^{-j\omega n}.$$

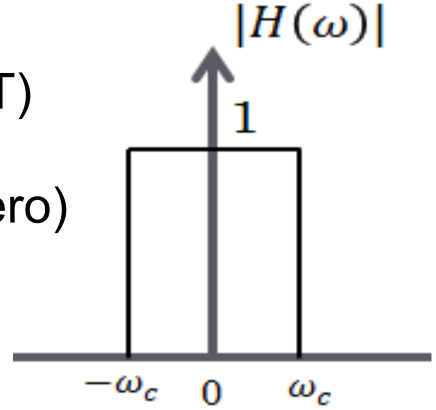
Inverse Discrete-Time Fourier transform (DTFT) of X(ω):

$$x(n) = \frac{1}{2\pi}\int_{-\pi}^{\pi} X(\omega) e^{j\omega n} d\omega.$$



4) Design of FIR Filter – using windowing technique

- Example : Design a digital filter with an ideal low pass characteristic and cutoff frequency $\omega_c = \pi/10$
- For an ideal low pass, $H(\omega) = 1$ at the passband, and 0 otherwise.



$$h(n) = \frac{1}{2\pi}\int_{\frac{-\pi}{10}}^{\frac{\pi}{10}} H(\omega)e^{+j\omega n}d\omega$$ (using IDTFT)

$$= \frac{1}{2\pi}\frac{1}{jn}\{1e^{+j\omega n}\}_{\omega=-\pi/10}^{\pi/10}$$ (if n is not zero)

$$= \frac{1}{\pi n}\left\{\frac{e^{+j\frac{\pi}{10}n} - e^{-j\frac{\pi}{10}n}}{2j}\right\}$$

Since $e^{\pm j\theta} = \cos(\theta) \pm j\sin(\theta)$, implying $e^{j\theta} - e^{-j\theta} = 2j\sin(\theta)$

Therefore $\frac{e^{+j\frac{\pi}{10}n} - e^{-j\frac{\pi}{10}n}}{2j} = \sin(\frac{\pi}{10}n)$

and $h_{ideal} = h(n) = \frac{\sin(n\,\omega_c)}{\pi n}$

4) Design of FIR Filter – using windowing technique

$$h_{ideal} = h(n) = \frac{\sin(n\,\omega_c)}{\pi n}$$

- As expected, h(n) is sinc function
- Evaluating h(n) for n =-∞ …∞ to get h(n). One problem occurs when n = 0: To solve for h(n) we use the expression directly

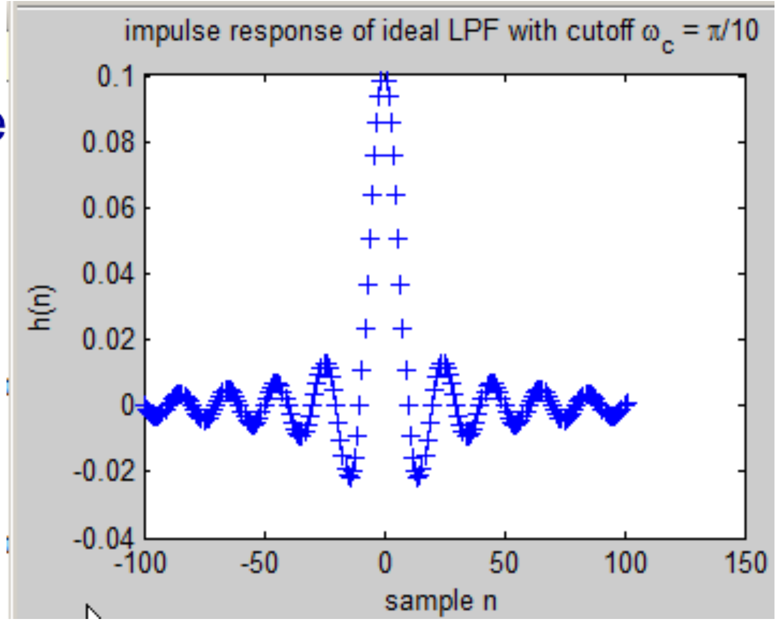
$$h(n=0) = \frac{1}{2\pi}\int_{-\omega_c}^{\omega_c} e^{j\omega n} d\omega$$

$$= \frac{1}{2\pi}\int_{-\omega_c}^{\omega_c} 1 d\omega$$

$$= \frac{2\omega_c}{2\pi} = \frac{\omega_c}{\pi}$$

4) Design of FIR Filter – using windowing technique

- What does h(n) look like? For our example



Matlab: plot_idealLPF_coefficients.m

DTFT/IDTFT: As H(ω) is continuous and periodic , h(n) is discrete and aperiodic sequence.

- Some observations:

a) h(n) is discrete and aperiodic as expected.
b) h(n) is a non-causal sequence, stretching from –infinity to +infinity
c) The value h(0) = ω_c/π. In our example, $\omega_c = \pi/10$, hence h(0) = 1/10.
d) The NULLs along the x-axis are multiples of K* $\frac{\pi}{\omega_c}$, e.g $\frac{\pi}{\omega_c} = 10$ in our example

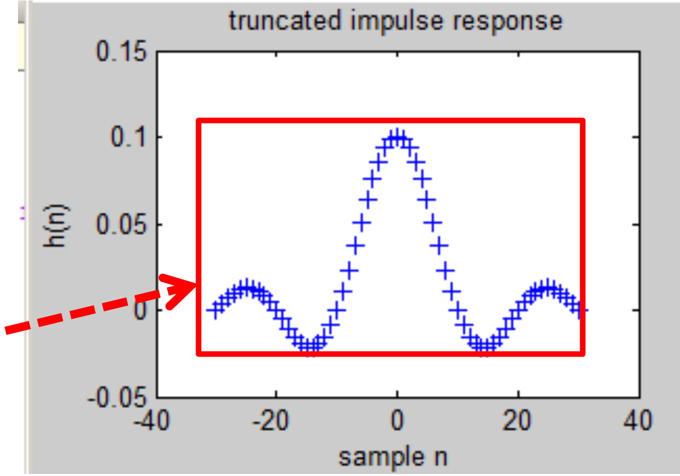
- We have a Problem! We found h(n) to be of infinite length and non-causal.
    - We cannot implement a filter with infinite length (complexity!)
    - and we want a causal filter so that it can be implemented for real time application. So how?

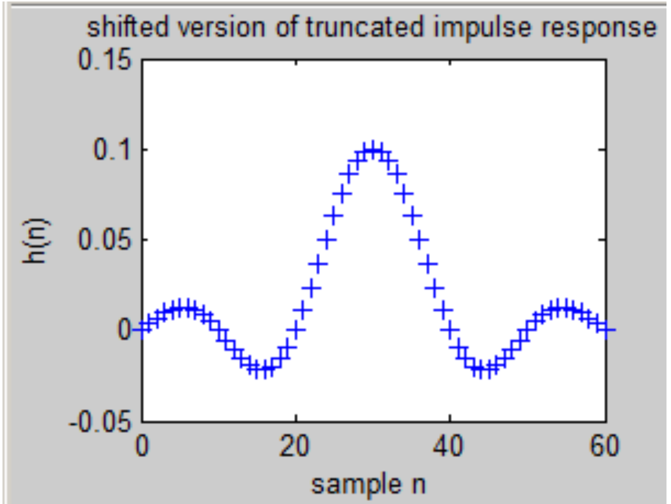
4) Design of FIR Filter – using windowing technique

- Solution:
  - truncate the filter! (e.g we take only coefficients from n=-30:30)—61 taps
  - shift the response so that h(n) = 0 for n< 0, we shift the above response (delay) by 30 samples!

    A delay version of h(n-D) simply introduces a linear phase shift.

These two operations are known as **windowing**: the truncation basically is a multiplication of a rectangle window to the original h(n)!

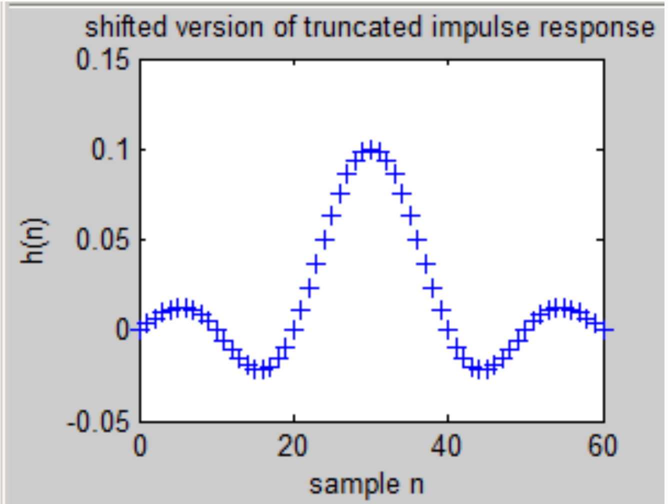


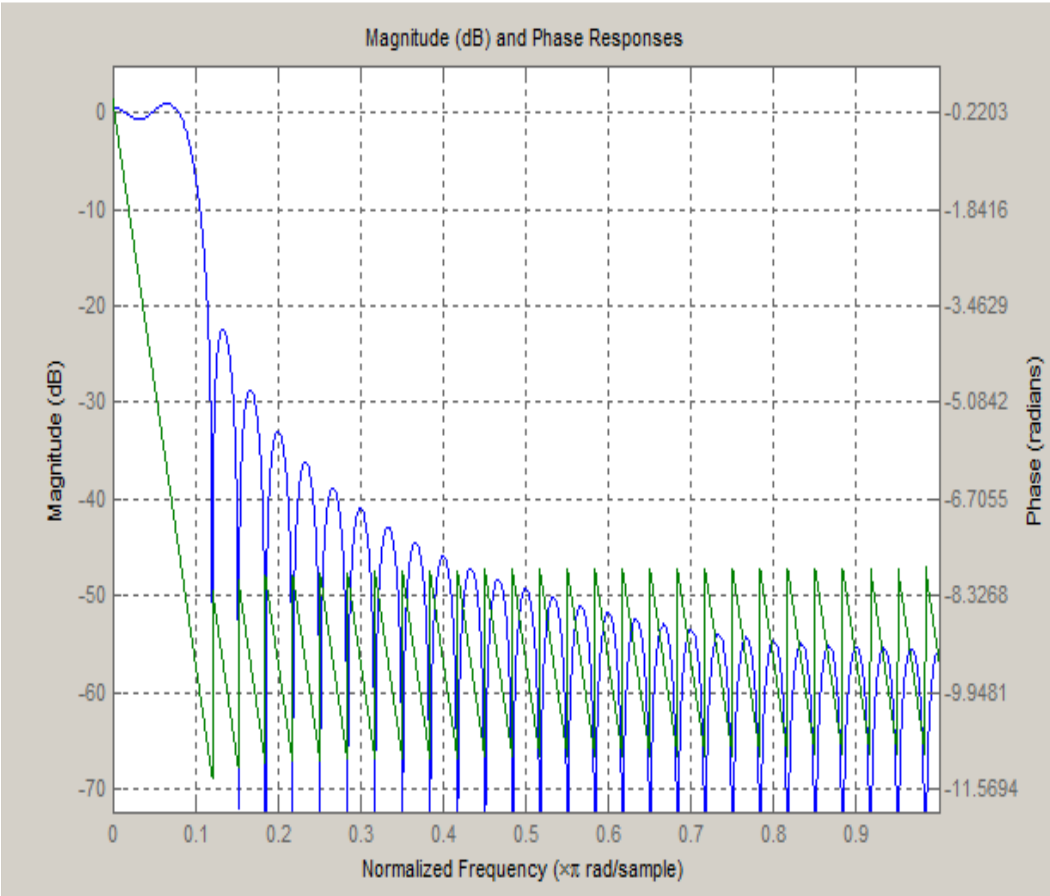


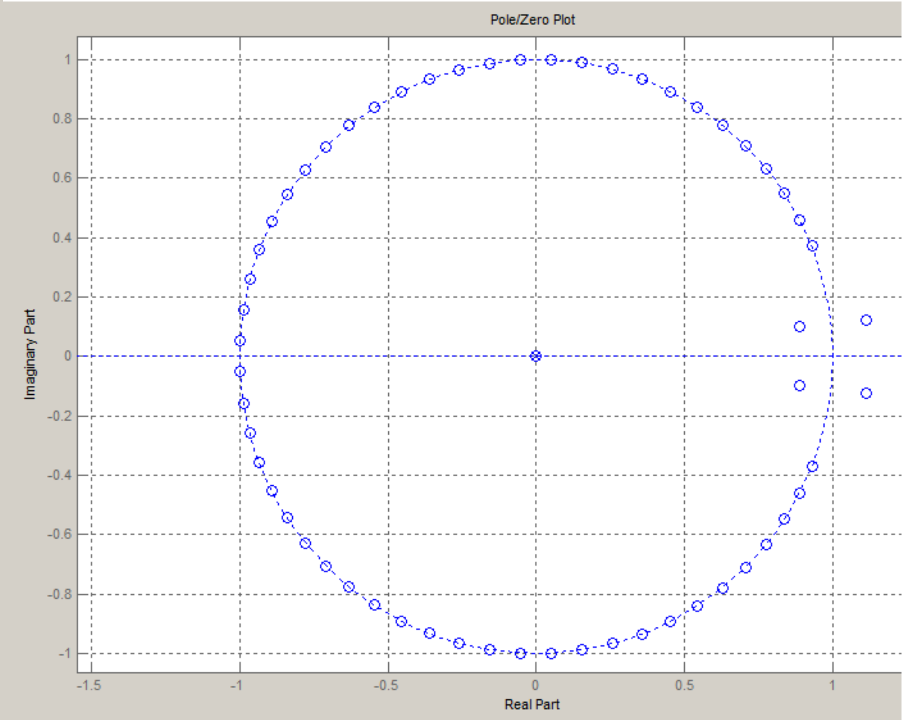
4) Design of FIR Filter – using windowing technique

The FIR filter feedforward coefficient b = $h_{final}(n)$, fvtool(b,1) to check on your design!

Intuitive to plot pole/zero and magnitude/phase response.







x-axis: it is normalizedFreq (in π)

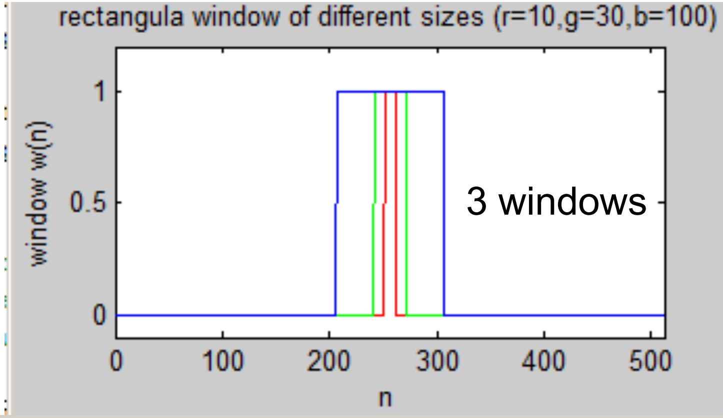
4) Effect of window: multiplication in time is convolution in frequency

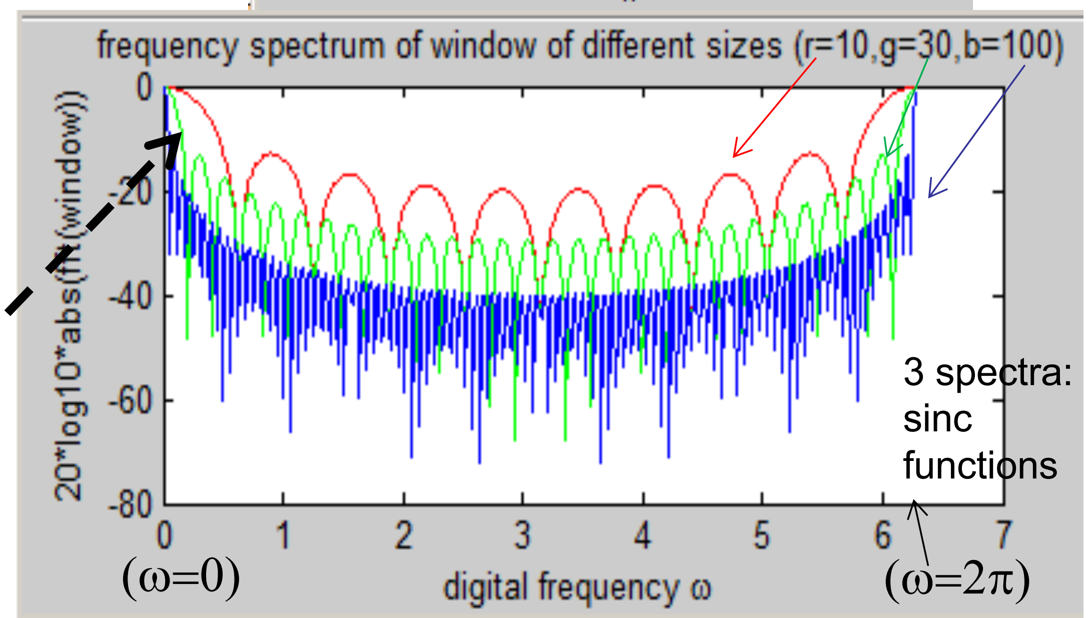
$h_{final}(n) = h_{ideal}(n).window(n)$: coefficients of the designed FIR filter!

⇕

convolution of the ideal LPF frequency response with the frequency response of the rectangular window

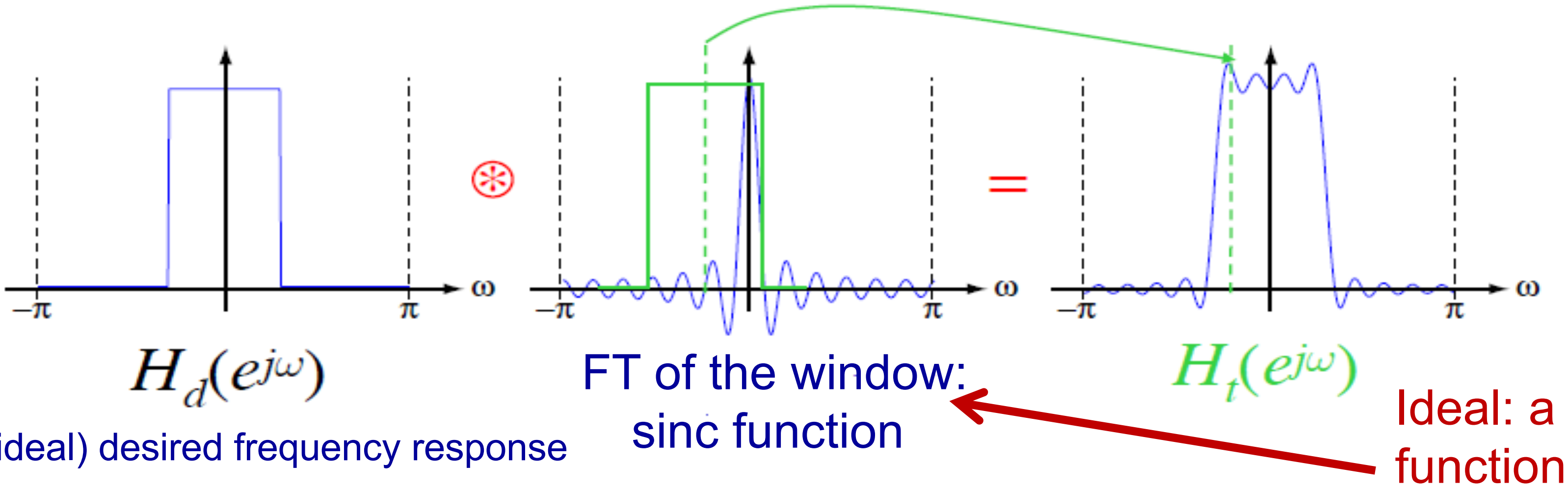
The width of the window affects the spectrum (main lobe width)





Matlab: plot_rectangleWindow_spectrum.m

4) Effect of window: multiplication in time is convolution in frequency

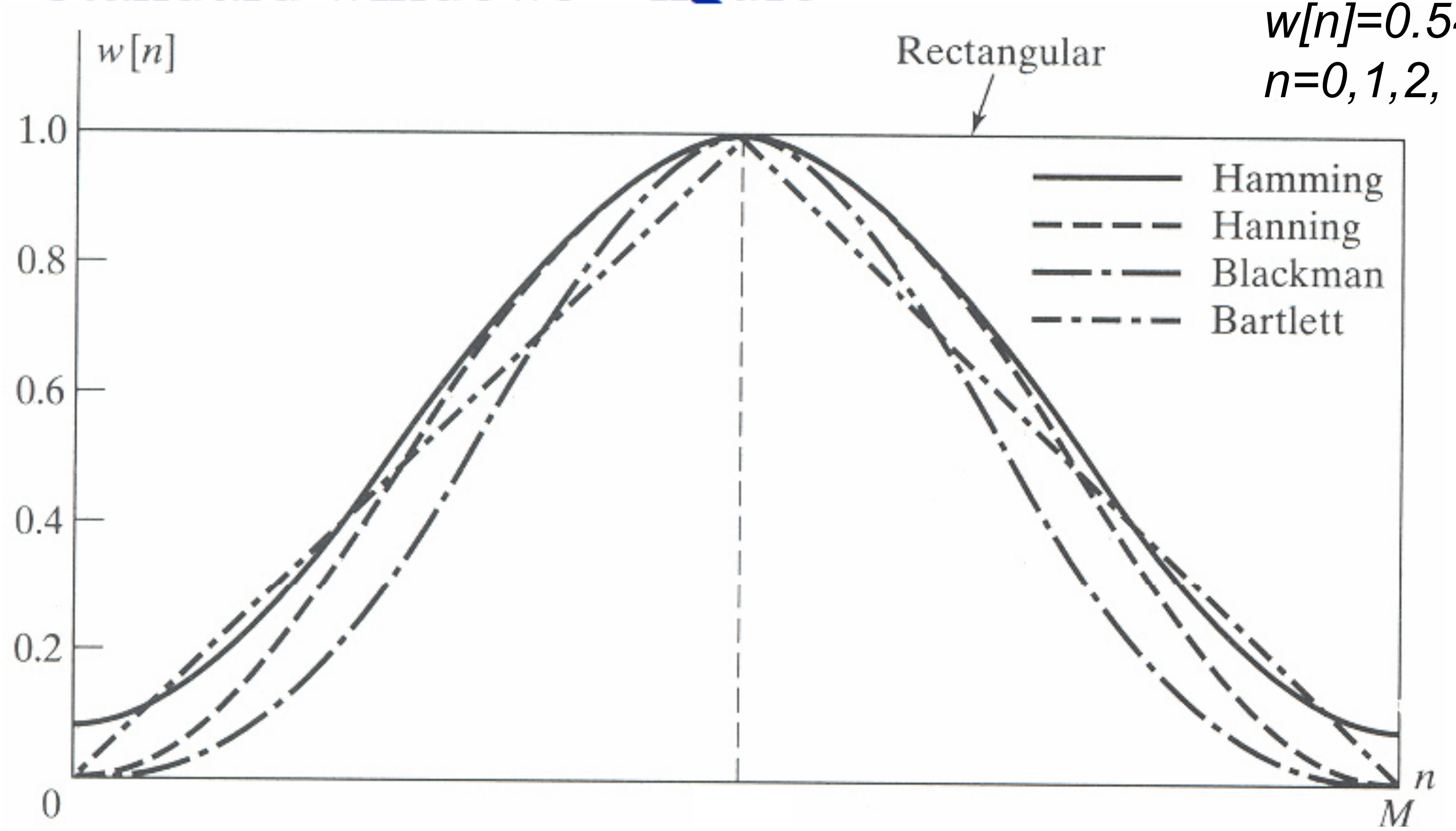


- This convolution: give rise to the ripple seen in the pass band and stop band.

See Manolakis and Ingle (Cambridge University Press, “Applied DSP”, pg 566-567 discussion)

4) Some COMMON windows

Standard windows – figure



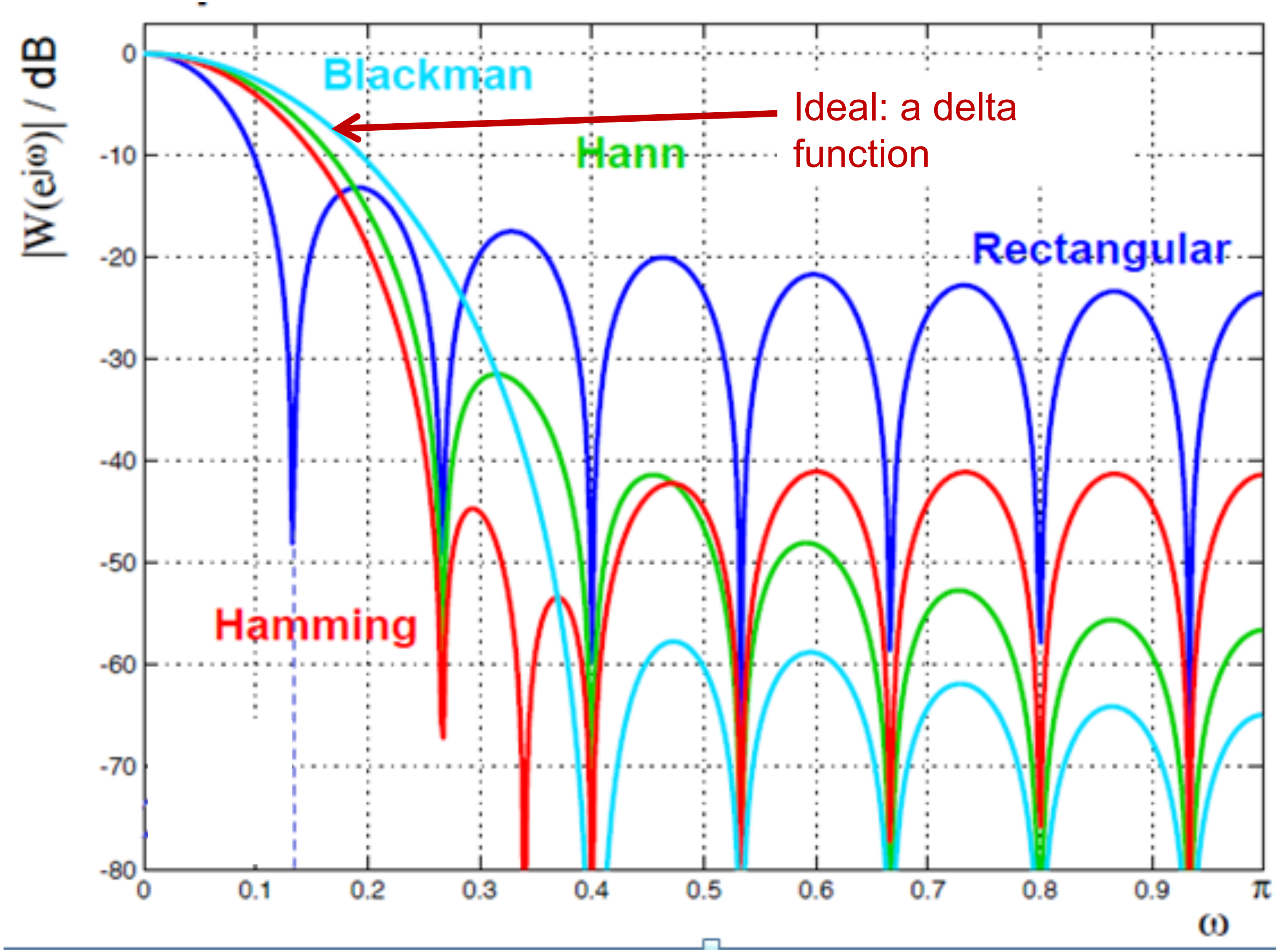
Hamming window;
$w[n]=0.54-0.46\cos(2\pi n/(N-1))$
$n=0,1,2, \ldots, N-1.$

Windows differs in its shape to allow a smoother transition in the frequency domain, hence lower its side lobes. This choice however will result in a broadening of the main lobe. We can narrow main lobe by having a larger window (i.e. more coefficients).

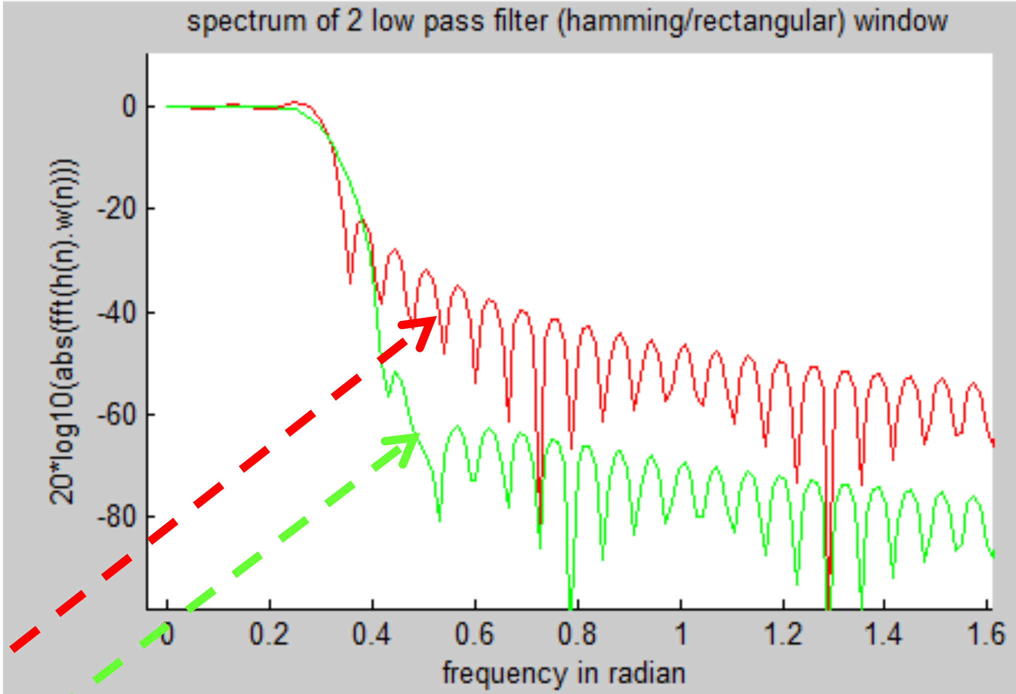
Practise with Q2, Q4(b), Tut 5.

4) Spectrum of various windows

Window shape selection affects side lobe attenuation and main lobe width

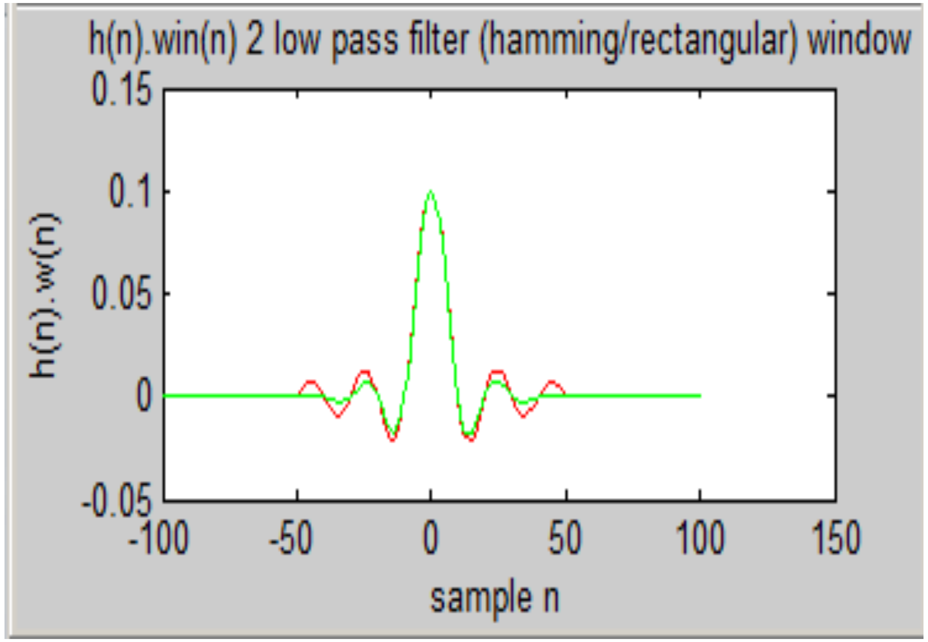


4. Comparing Hamming window and Rectangular window LPF response



The Hamming window provides a smoother transition in the time-domain (a broader transition in the frequency domain)!
However, hamming window provides better attenuation in the stop band.

Red = rectangular window
Green = hamming window
100 coefficients retained

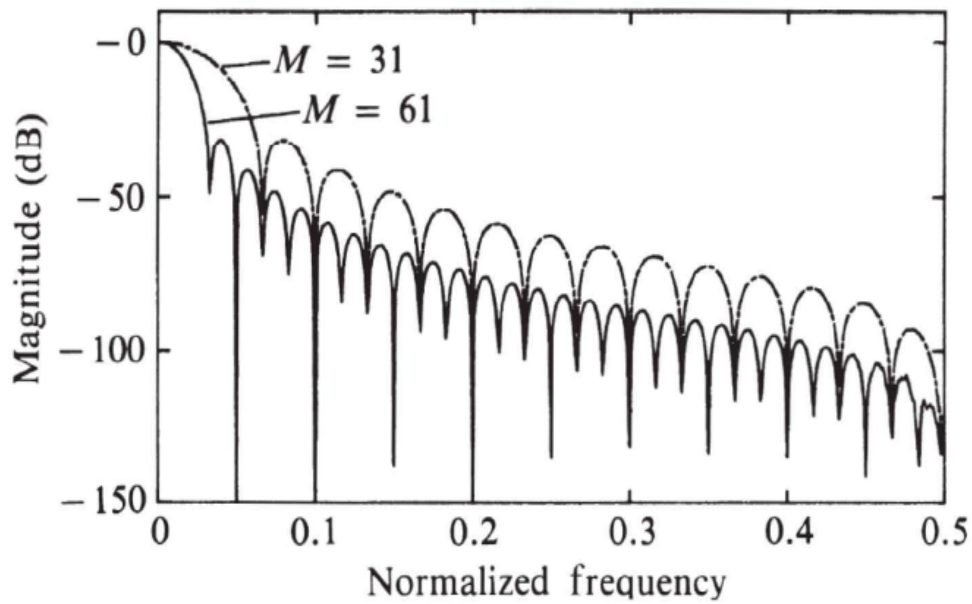


Matlab: plot_recWin_hammingWin_LPF_coefficients.m

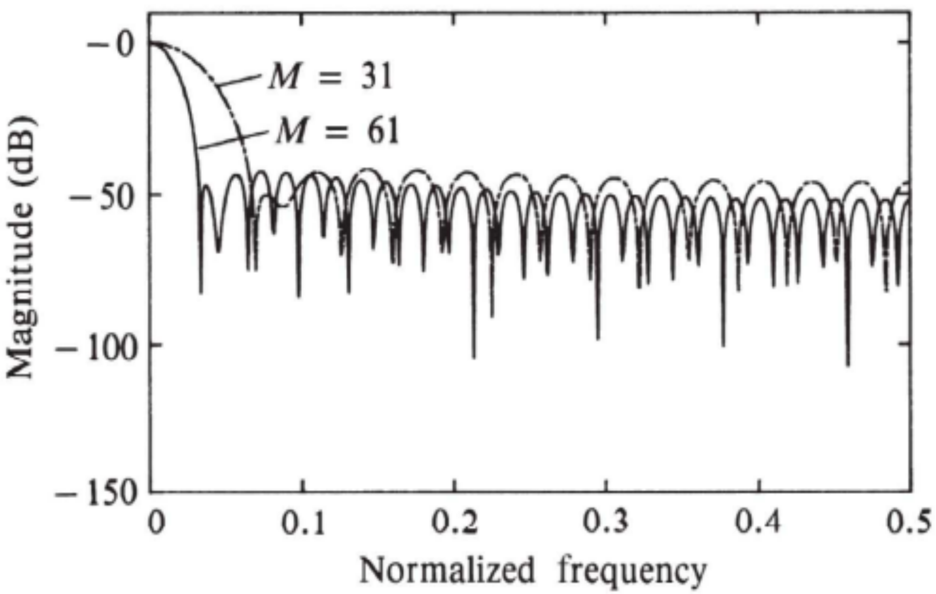
4. Window length and its increase

1) Main lobe is reduced
2) 1st side lobe height is not affected!
3) Length can be used to improve transition band at the expense of longer filter (more computation)

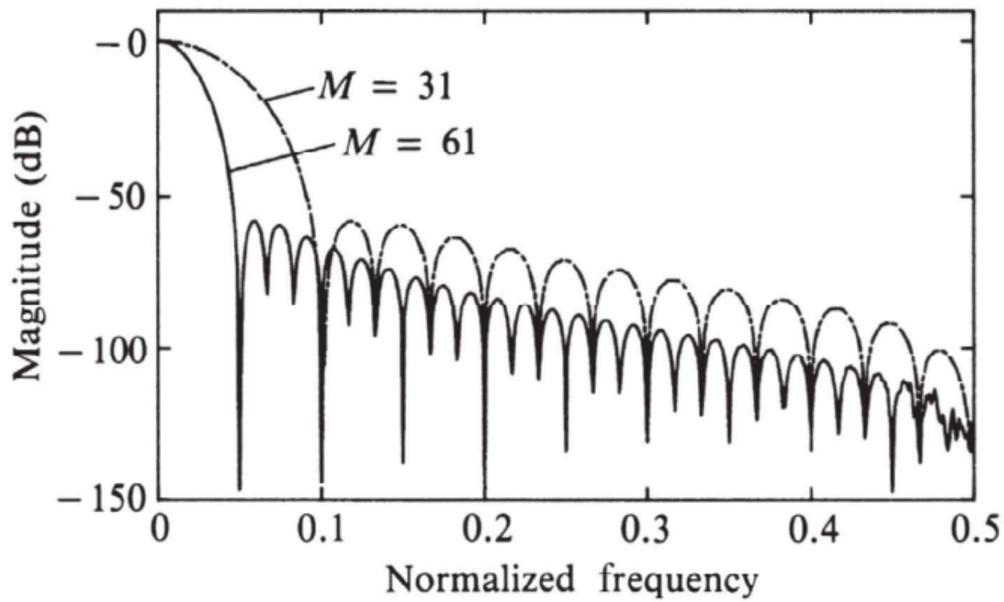
Hanning Window



Hamming Window



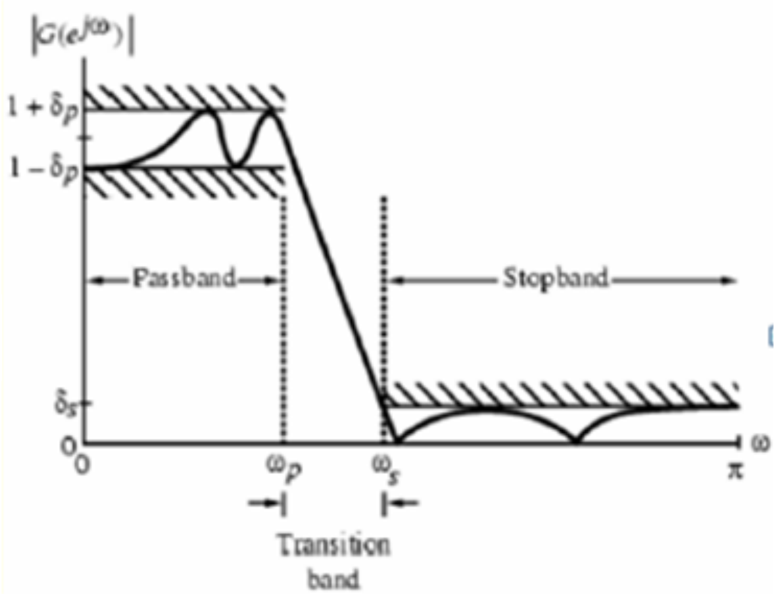
Blackman Window



4. Window length estimation

- How to select the 'suitable' order, i.e. the length
  - The lowest order that can meet the requirement
- There are several methods:
  - Kaiser, Bellanger, Hermann
- Kaiser's formula:

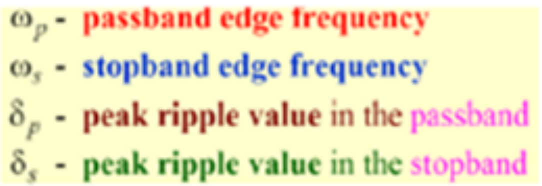
$$N \cong \frac{-20log_{10}\left(\sqrt{\delta_p\delta_s}\right)-13}{14.6(\omega_s-\omega_p)/2\pi} \text{ or } N \cong \frac{-20log_{10}\left(\sqrt{\delta_p\delta_s}\right)-13}{14.6(f_s-f_p)}$$



- Bellanger's formula:

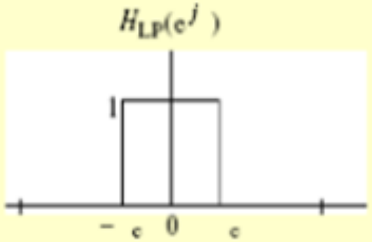
$$N \cong \frac{-2log_{10}(10\delta_p\delta_s)}{3(f_s-f_p)} - 1$$

- Hermann's formula is more complex, so we will not consider here
- Nevertheless, once the filter is designed, you should still examine the magnitude response to see if it meets the requirement. The phase response is linear.



Practise with Q1, Tut 5.

4. Coefficients of ideal filters by IDTFT

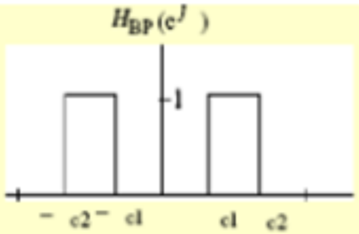


$$h_{LP}[n] = \frac{\sin \omega_c n}{\pi n}, \quad -\infty \le n \le \infty$$

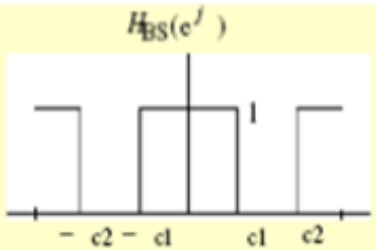
$$= \frac{\omega_c}{\pi} \quad (n = 0)$$

We have derived this!

$$h_{HP}[n] = \begin{cases} 1 - \dfrac{\omega_c}{\pi}, & n = 0 \\ -\dfrac{\sin(\omega_c n)}{\pi n}, & n \neq 0 \end{cases}$$

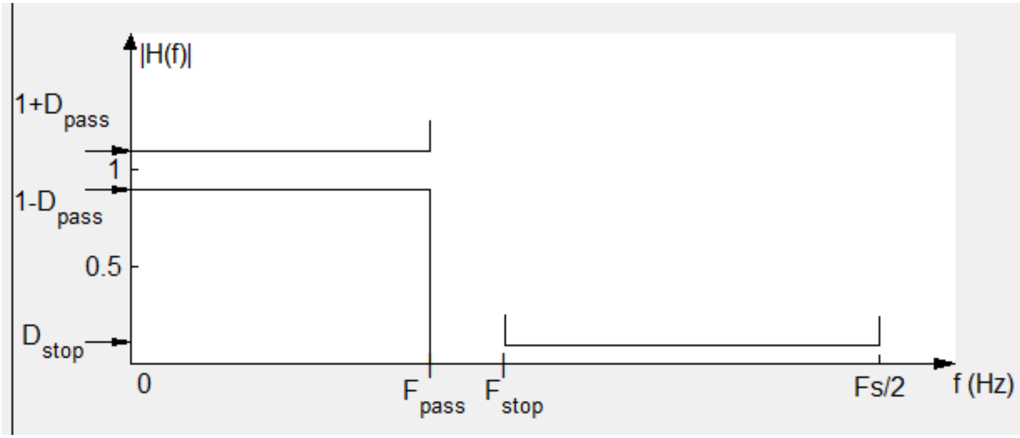


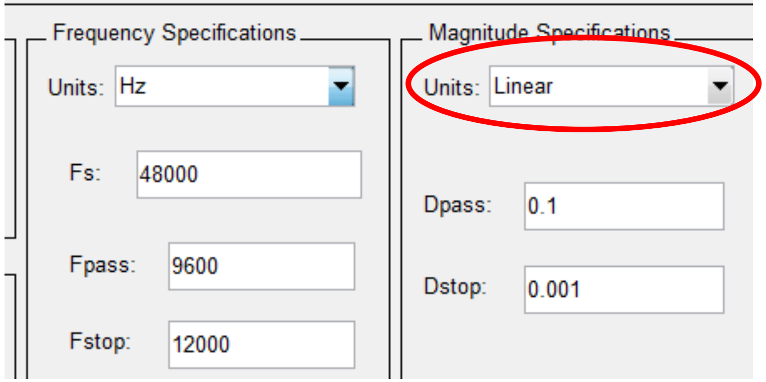
$$h_{BP}[n] = \begin{cases} \dfrac{\sin(\omega_{c2} n)}{\pi n} - \dfrac{\sin(\omega_{c1} n)}{\pi n}, & n \neq 0 \\ \dfrac{\omega_{c2}}{\pi} - \dfrac{\omega_{c1}}{\pi}, & n = 0 \end{cases}$$

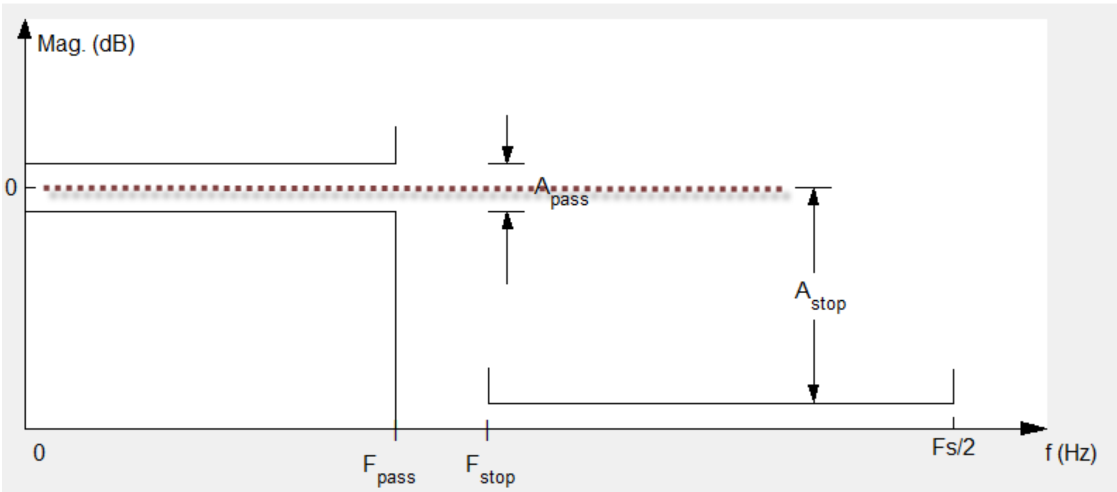


$$h_{BS}[n] = \begin{cases} 1 - \dfrac{(\omega_{c2} - \omega_{c1})}{\pi}, & n = 0 \\ \dfrac{\sin(\omega_{c1} n)}{\pi n} - \dfrac{\sin(\omega_{c2} n)}{\pi n}, & n \neq 0 \end{cases}$$

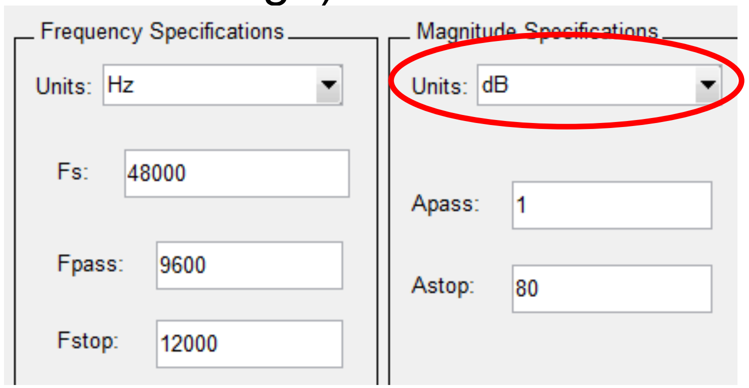
5. Specifications of the filter's magnitude response (MATLAB)





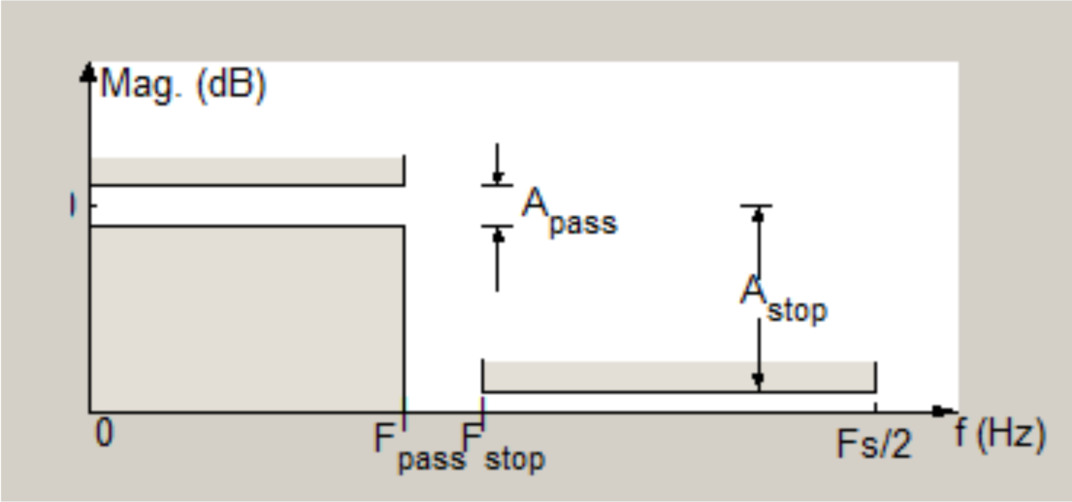


DSP: P2.2.2 (FIR, Filter Design)

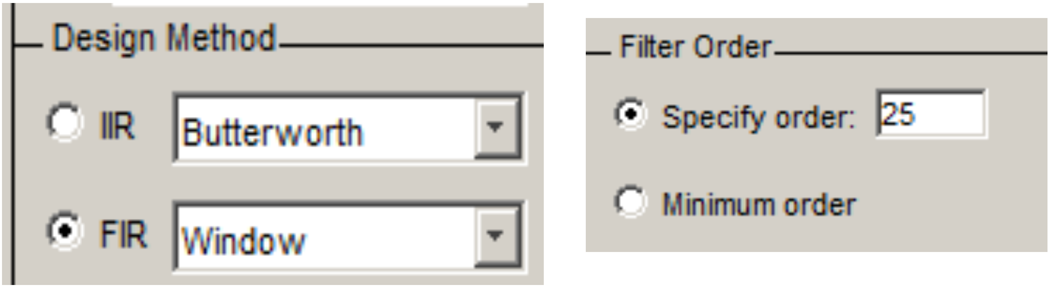


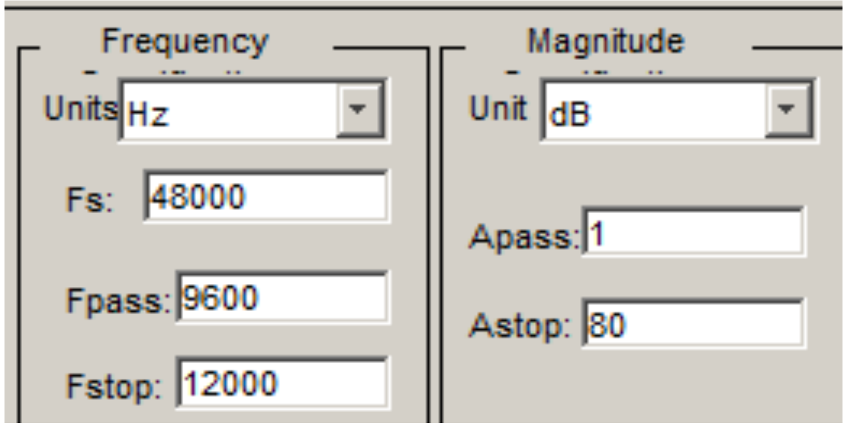
5) Design of FIR Filter using MATLAB (FDATOOL)

Matlab's fdatool (filter design analysis) tool allows us to design, and analyse digital FIR and IIR filters.



For designing, we specify the above performance specifications and design method (FIR/IIR), & Filter Order.

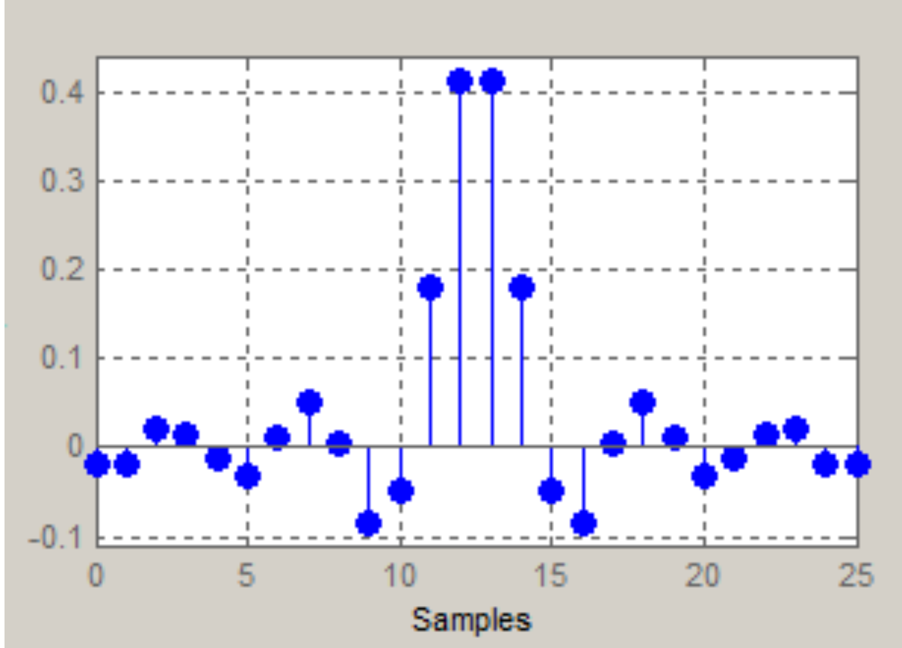




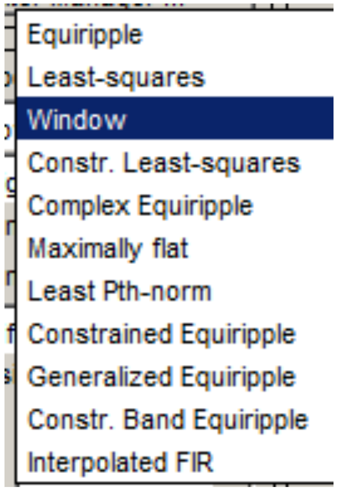
There are many other options, and you should try it out!
Since Matlab can help us design, why do we need to learn how to design it?

4) Design of FIR Filter using MATLAB.

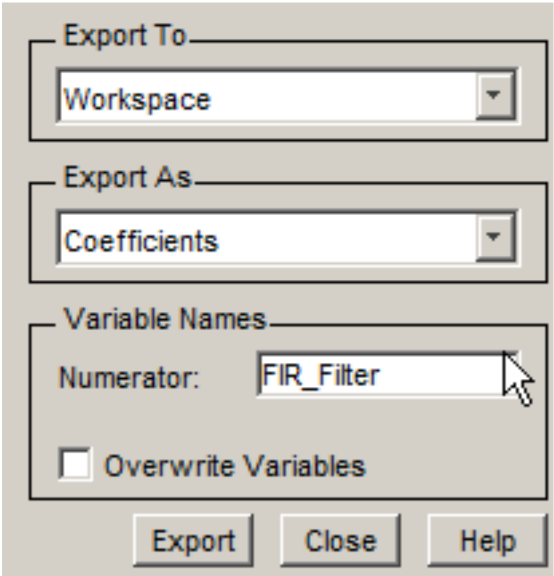
Using Matlab’s fdatool to design FIR filter:
a)You basically get the impulse response
b)Exporting the coefficients and use it in your implementation! That’s it.



Many choices for FIR Design



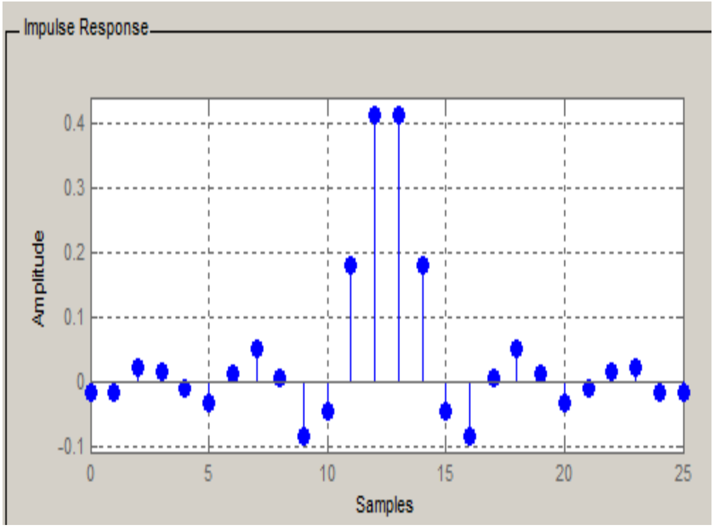
Exporting the coefficients



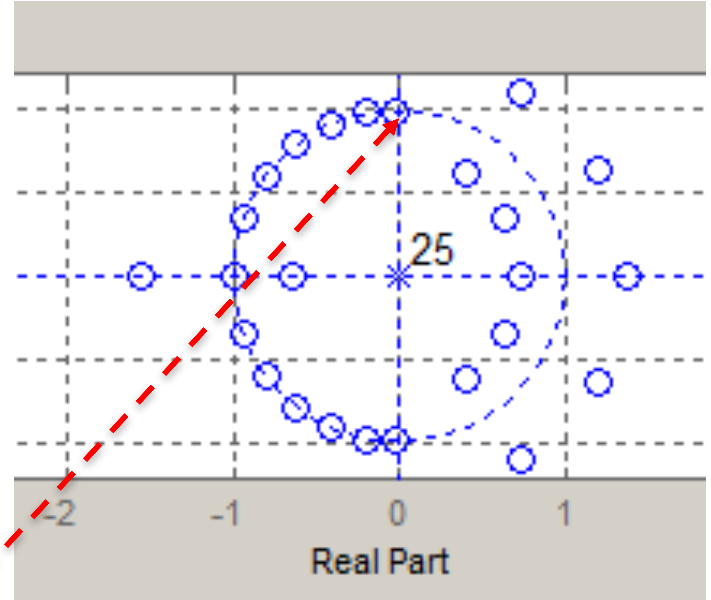

```
>> whos FIR_Filter
  Name            Size

  FIR_Filter      1x26

>> plot(FIR_Filter)
```

4) Analysis using FDATOOL

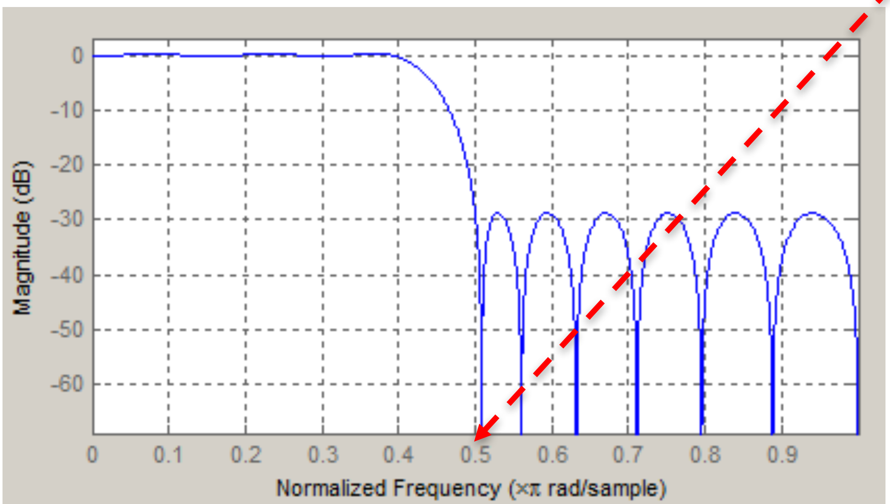
h(n)



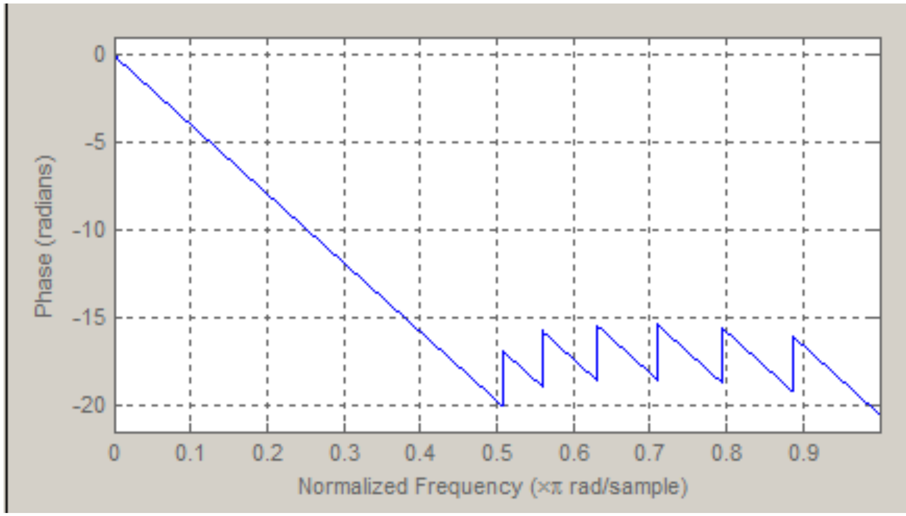
H(z)



|H(ω)|



∠H(ω)



FIR filter design summary

The windowing method for FIR Filter design.

1) Given a digital filter specification cutoff, we begin with the ideal filter spectrum and perform IDTFT to get $h_{ideal}(n)$.

2) We truncate the filter coefficients by multiplying $h_{ideal}(n)$ with a window, e.g rectangular, hamming, etc, i.e. $h_{final} = h_{ideal}(n) * your\_window(n)$

An important point to note: How does different window affect transition bandwidth and stopband attenuation?

3) The filter length can be estimated: Kaiser, and Bellanger equations.

4) The coefficients are the feedforward coefficients of FIR filter, $b = h_{final}$

5) We can check if it meets specification using Matlab: fvtool(b,1)