



Design of FIR Filters

Elena Punskeya

www-sigproc.eng.cam.ac.uk/~op205

Some material adapted from courses by
Prof. Simon Godsill, Dr. Arnaud Doucet,
Dr. Malcolm Macleod and Prof. Peter Rayner

FIR as a class of LTI Filters

Transfer function of the filter is

$$H(z) = \frac{Y(z)}{X(z)} = \frac{\sum_{k=0}^M b_k z^{-k}}{1 + \sum_{k=1}^N a_k z^{-k}}$$

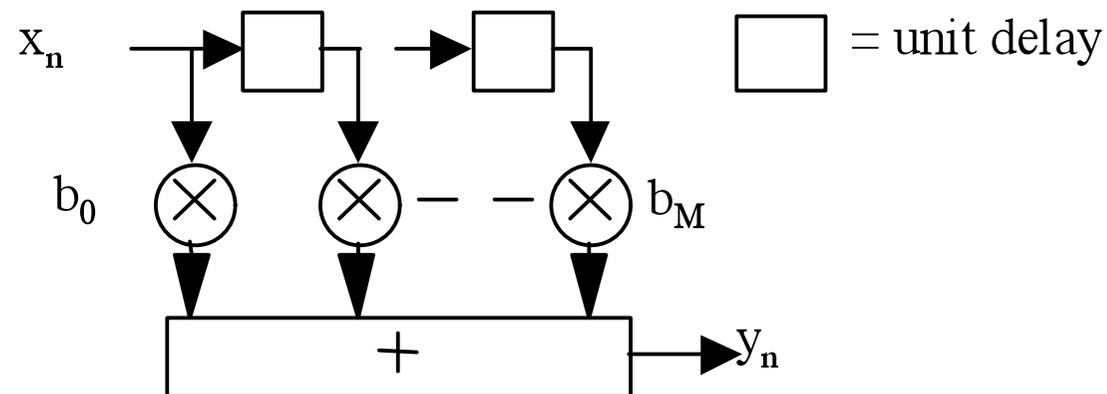
Finite Impulse Response (FIR) Filters:

$N = 0$, no feedback

FIR Filters

Let us consider an FIR filter of length M (order $N=M-1$, watch out!
order – number of delays)

$$y(n) = \sum_{k=0}^{M-1} b_k x(n-k) = \sum_{k=0}^{M-1} h(k) x(n-k)$$



FIR filters

Can immediately obtain the impulse response, with $x(n) = \delta(n)$

$$h(n) = y(n) = \sum_{k=0}^{M-1} b_k \delta(n - k) = b_n$$



The impulse response is of finite length M , as required

Note that FIR filters have only zeros (no poles). Hence known also as **all-zero** filters

FIR filters also known as **feedforward** or **non-recursive**, or **transversal**

FIR Filters

Digital FIR filters cannot be derived from analog filters – rational analog filters cannot have a finite impulse response.

Why bother?

1. They are inherently stable
2. They can be designed to have a linear phase
3. There is a great flexibility in shaping their magnitude response
4. They are easy and convenient to implement

Remember very fast implementation using FFT?

FIR Filter using the DFT

FIR filter:

$$y(n) = \sum_{k=0}^{M-1} h(k) x(n-k)$$

Since $h(n)$ and $x(n)$ are finite-duration sequences, their convolution is also finite in duration. The duration of the sequence $y(n)$ is $L + M - 1$.

Let us consider $N \geq L + M - 1$. Let us pad the sequences $h(n)$ and $x(n)$ with zeros to increase their lengths to N and perform the N -point DFT. The frequency-domain equivalent is

$$Y(k) = H(k) X(k)$$

with $k = \frac{2\pi k}{N}$.

Now N -point DFT ($Y(k)$) and then N -point IDFT ($y(n)$) can be used to compute standard convolution product and thus to perform linear filtering (given how efficient FFT is)

Linear-phase filters

The ability to have an exactly linear phase response is the one of the most important of FIR filters

$$H(\omega) = |H(\omega)| e^{j\phi(\omega)} \quad \text{where } \phi(\omega) = -\omega n_0$$

A general FIR filter does not have a linear phase response but this property is satisfied when

$$h(n) = \pm h(M-1-n), \quad n = 0, 1, \dots, M-1.$$

 four linear phase filter types

Impulse response	# coefs	$H(\omega)$	Type
$h(n) = h(M-1-n)$	Odd	$e^{-j\omega(M-1)/2} \left(h\left(\frac{M-1}{2}\right) + 2 \sum_{k=1}^{(M-3)/2} h\left(\frac{M-1}{2} - k\right) \cos(\omega k) \right)$	1
$h(n) = h(M-1-n)$	Even	$e^{-j\omega(M-1)/2} 2 \sum_{k=1}^{(M-3)/2} h\left(\frac{M}{2} - k\right) \cos\left(\omega\left(k - \frac{1}{2}\right)\right)$	2
$h(n) = -h(M-1-n)$	Odd	$e^{-j[\omega(M-1)/2 - \pi/2]} \left(2 \sum_{k=1}^{(M-1)/2} h\left(\frac{M-1}{2} - k\right) \sin(\omega k) \right)$	3
$h(n) = -h(M-1-n)$	Even	$e^{-j[\omega(M-1)/2 - \pi/2]} 2 \sum_{k=1}^{(M-1)/2} h\left(\frac{M}{2} - k\right) \sin\left(\omega\left(k - \frac{1}{2}\right)\right)$	4

Linear-phase filters – Filter types

Some observations:

- Type 1 – most versatile
- Type 2 – frequency response is always 0 at $\omega=\pi$ – not suitable as a high-pass
- Type 3 and 4 – introduce a $\pi/2$ phase shift, frequency response is always 0 at $\omega=0$ – not suitable as a high-pass

FIR Design Methods

- Impulse response truncation – the simplest design method, has undesirable frequency domain-characteristics, not very useful but intro to ...
- Windowing design method – simple and convenient but not optimal, i.e. order achieved is not minimum possible
- Optimal filter design methods

Back to Our Ideal Low-pass Filter Example

Let us consider for example a simple ideal lowpass filter defined by

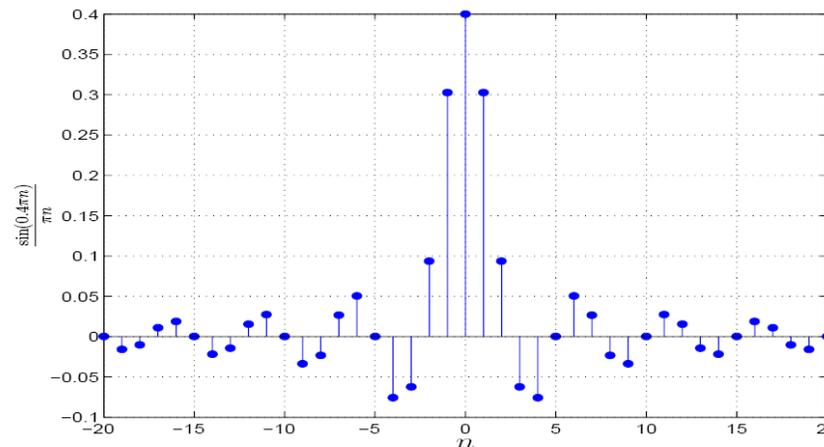
$$H_d(\omega) = \begin{cases} 1 & \text{if } |\omega| \leq \omega_c \\ 0 & \text{if } \omega_c < |\omega| < \pi. \end{cases}$$

It can be shown easily that the impulse response is given by

$$h_d(n) = \frac{\omega_c}{\pi} \frac{\sin \omega_c n}{\omega_c n}.$$

➔ Desired impulse response has a sinc shape which is non-causal and infinite in duration.

clearly
cannot
be implemented
in practice



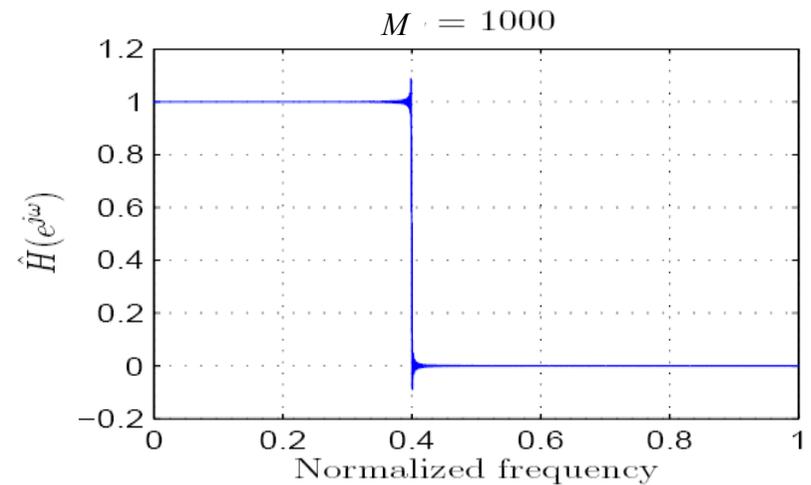
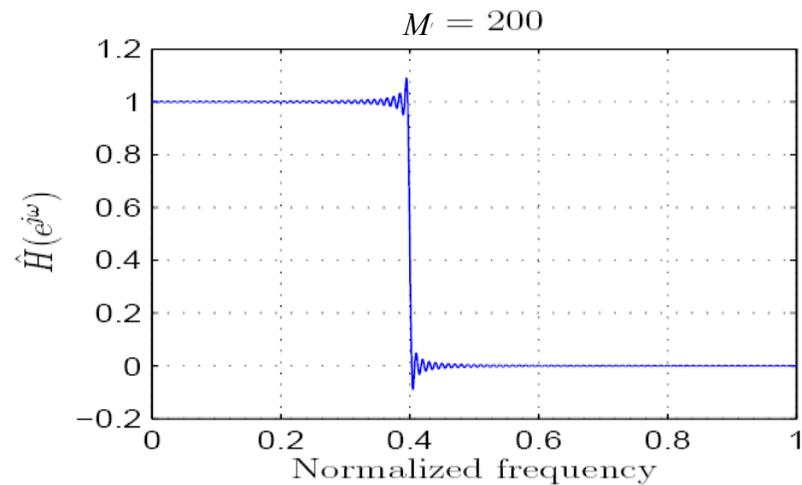
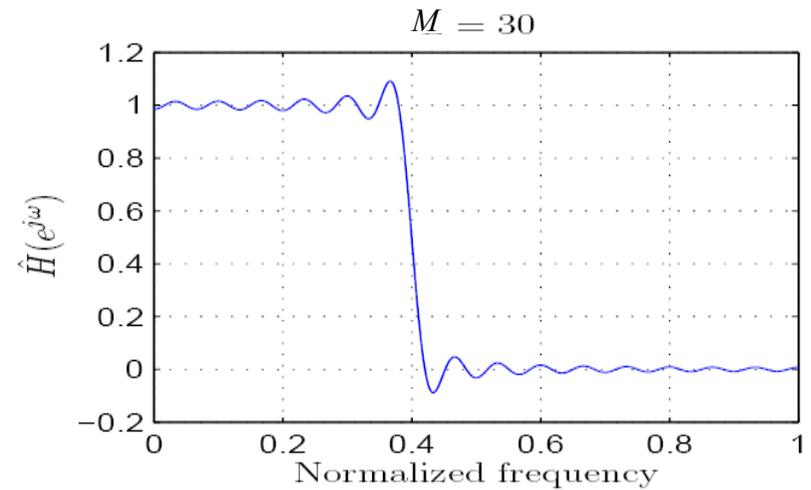
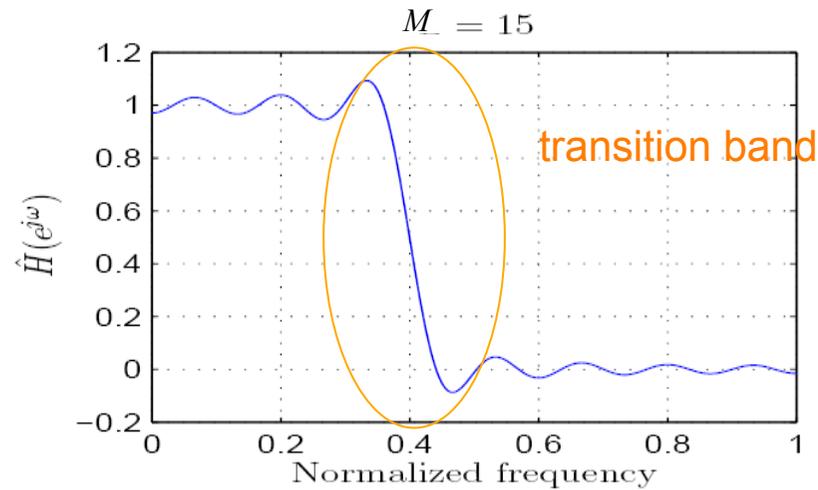
Approximation via truncation

- $h(n)$ infinite in duration \Rightarrow to compute any sample of the output, we need to know all samples of the input, both in the past and in the future!!
 \Rightarrow Unacceptable, so what if we just clip off the sinc at some large n

$$\hat{h}(n) = \frac{\sin(n\omega_c)}{\pi n} \text{ for } |n| \leq M \text{ and } 0 \text{ otherwise.}$$

- Here is what the frequency response now looks like for $\omega_c = 0.4\pi$ and different values of M .
- One observes *ripples* in both passband/stopband and transition not abrupt (leading to *transition band*).

Approximated filters obtained by truncation



Though the transition band gets narrower as $M \rightarrow \infty$, the ripple remains!

Window Design Method

To be expected ...

Truncation is just pre-multiplication by a rectangular window

$$w(n) = \begin{cases} 1 & \text{if } n = 0, 1, \dots, M - 1 \\ 0 & \text{otherwise.} \end{cases}$$

This is not very clever
– obviously one
introduces a delay

Fourier transform $H(\omega)$ of the truncated filter $h(n) = h_d(n) w(n)$ is

$$H(\omega) = \frac{1}{2\pi} \int_{-\pi}^{\pi} H_d(\lambda) W(\omega - \lambda) d\lambda$$

spectrum convolution

where $W(\omega)$ is the Fourier transform of the rectangular window

$$W(\omega) = e^{-j\omega(M-1)/2} \frac{\sin(\omega M/2)}{\sin(\omega/2)}$$

Rectangular Window Frequency Response

Fourier transform of the rectangular window

$$W(\omega) = e^{-j\omega(M-1)/2} \frac{\sin(\omega M/2)}{\sin(\omega/2)}$$

admits as magnitude and phase responses

$$|W(\omega)| = \frac{|\sin(\omega M/2)|}{|\sin(\omega/2)|}, \quad |\omega| < \pi,$$

$$\theta(\omega) = \begin{cases} -\omega(M-1)/2 & \text{when } \sin(\omega M/2) \geq 0 \\ -\omega(M-1)/2 + \pi & \text{when } \sin(\omega M/2) < 0 \end{cases}$$



i.e. $W(\omega)$ has a piecewise linear phase

Window Design Method

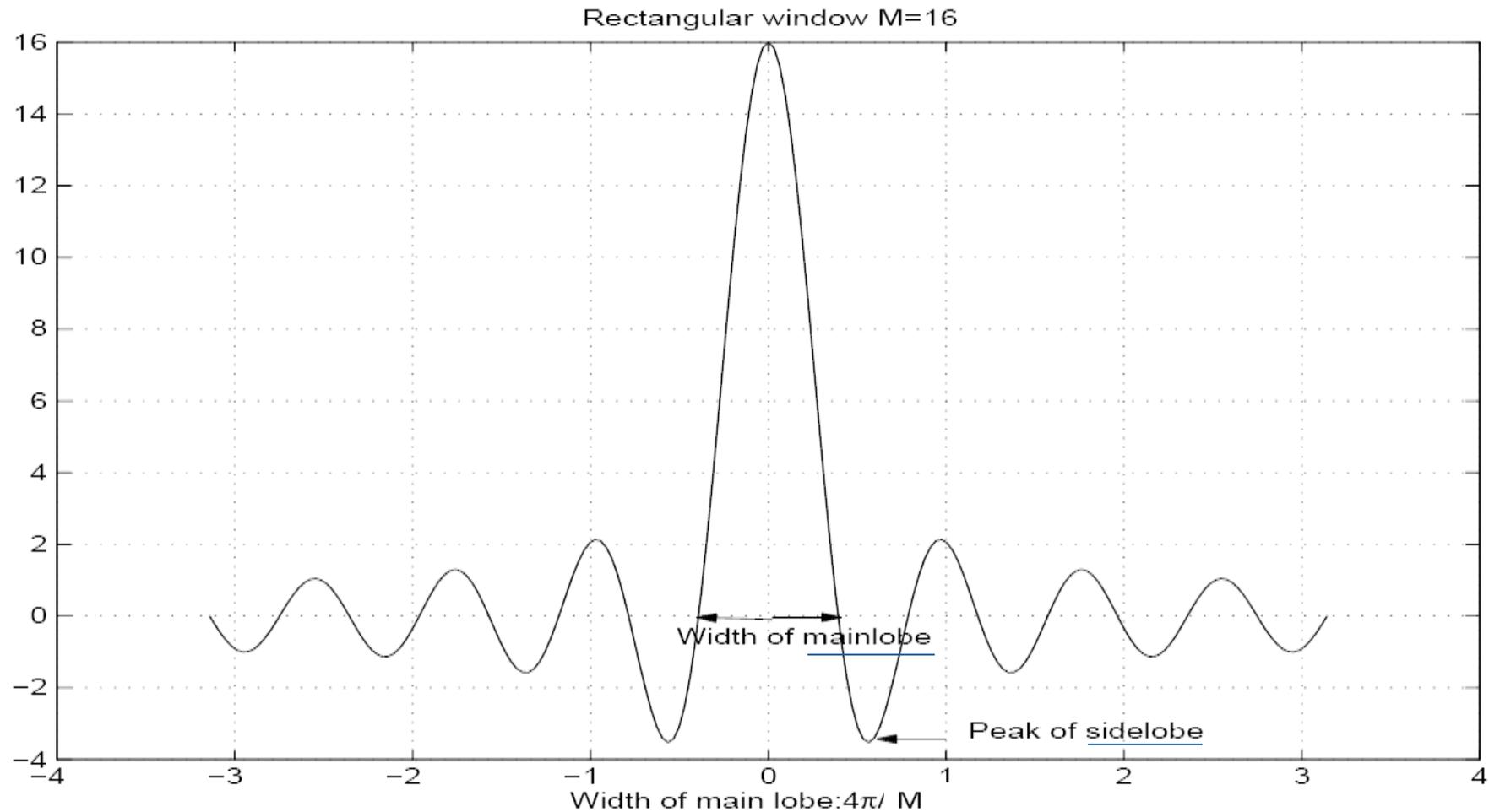
- Practically, one uses **truncated** & **delayed** impulse response

$$\tilde{h}(n) = \hat{h}\left(n - \frac{M-1}{2}\right) \text{ where } \hat{h}(n) = \frac{\omega_c \sin(\omega_c n)}{\pi \omega_c n} \mathbf{1}_{\left\{-\frac{M-1}{2}, \dots, \frac{M-1}{2}\right\}}(n)$$

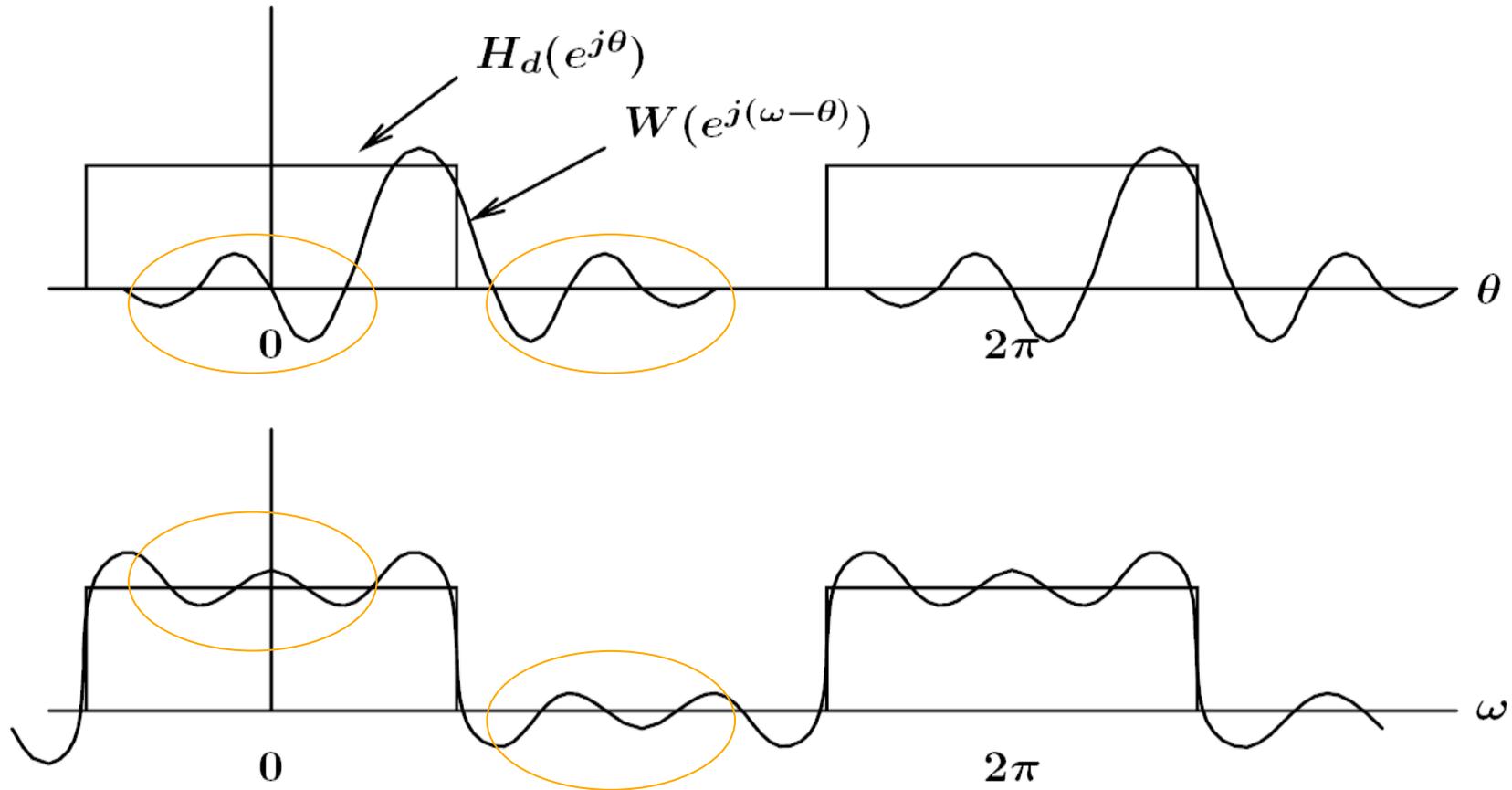
where M is the *filter length* & $N = M - 1$ is known as *filter order*.

- Delaying operation \rightarrow introduce linear phase term.
 \Rightarrow The resulting filter is causal and has a linear phase.

Magnitude of Rectangular Window Frequency Response

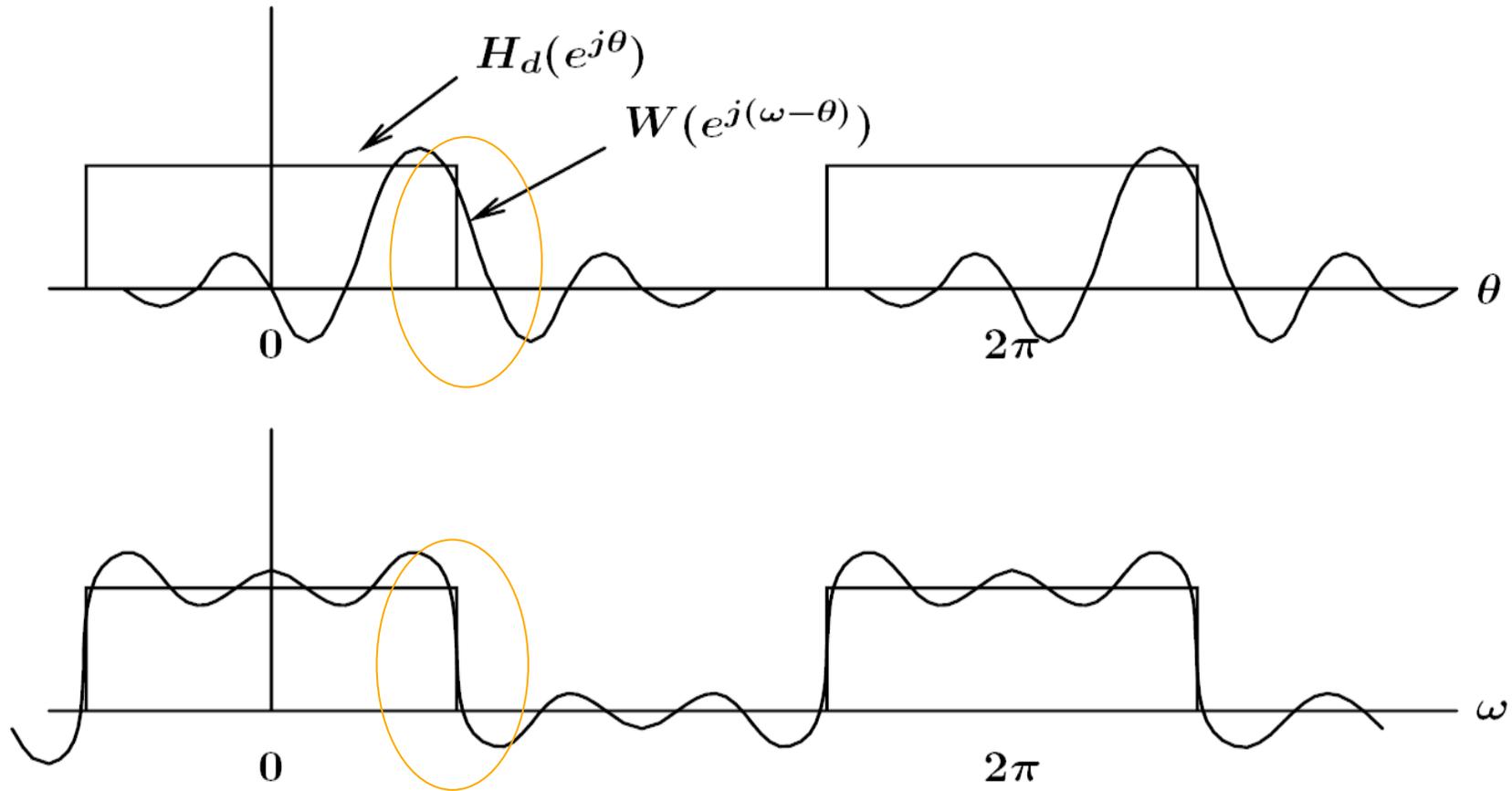


Truncated Filter



- The *higher* the areas under sidelobes, the *larger* the ripples in the passband and the stopband.

Truncated Filter



- The width of the transition region between passband and stopband in $H(\omega)$ increases with the width of the main lobe of $W(\omega)$.

Ideal Requirements

Ideally we would like to have

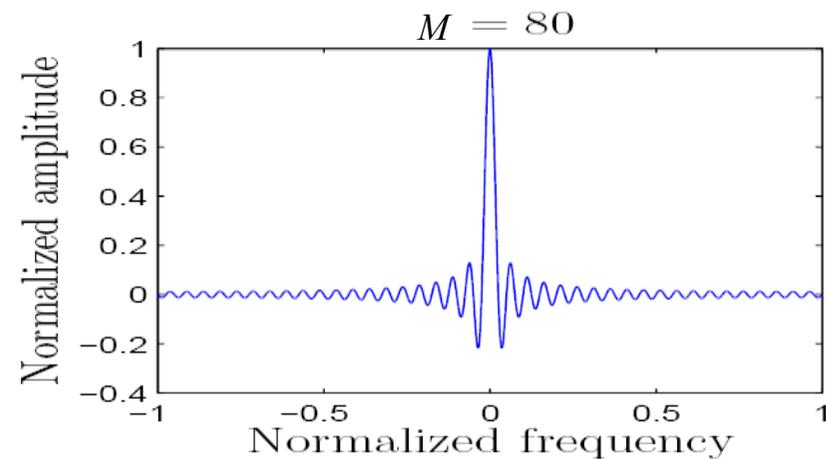
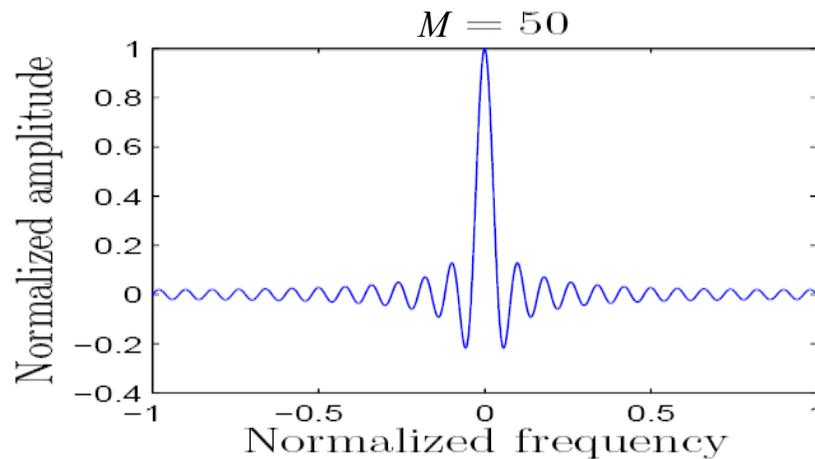
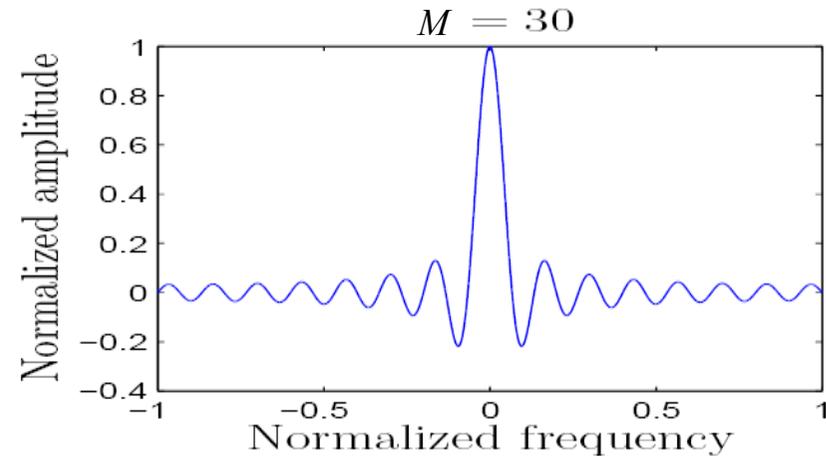
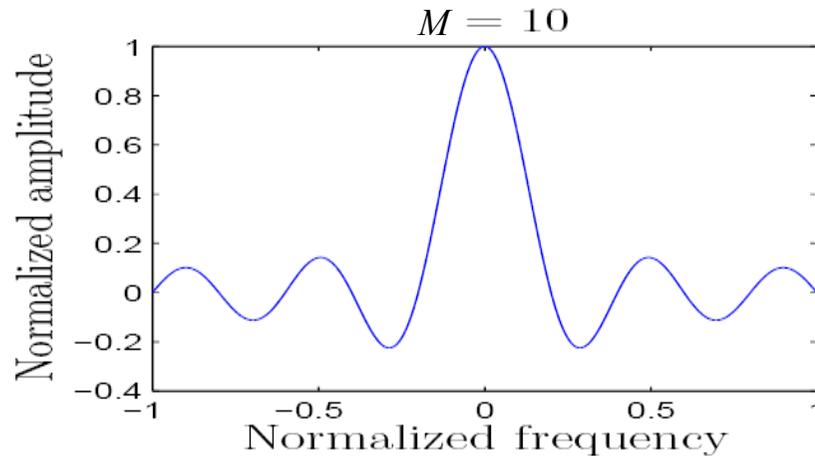
- M small – few computations
- $W(\omega)$ close to a delta Dirac mass for $H(\omega)$ to be close to $H_d(\omega)$

$$H_d(\omega) = \begin{cases} 1 & \text{if } |\omega| \leq \omega_c \\ 0 & \text{if } \omega_c < |\omega| < \pi \end{cases}$$

our ideal low-pass filter

These two requirements are conflicting!

Increasing the dimension of the window



- The width of the main lobe decreases as M increases
- The area under sidelobes remain constant as M increases

Conflicting Ideal Requirements

- The width of the transition region between passband and stopband in $H(\omega)$ increases with the width of the main lobe of $W(\omega)$.
- The ripple in the passband and stopband is dependent on the area under the side-lobes.
- For the rectangular window, the width of the main lobe decreases as M increases, but it can be shown that the area under the sidelobes remain constant.

⇒ Transition region gets smaller but the ripple remains.

⇒ PROBLEM: Sharp discontinuity of rectangular windows!

Solution to Sharp Discontinuity of Rectangular Window

Use windows with **no abrupt discontinuity** in their time-domain response and consequently **low side-lobes** in their frequency response.

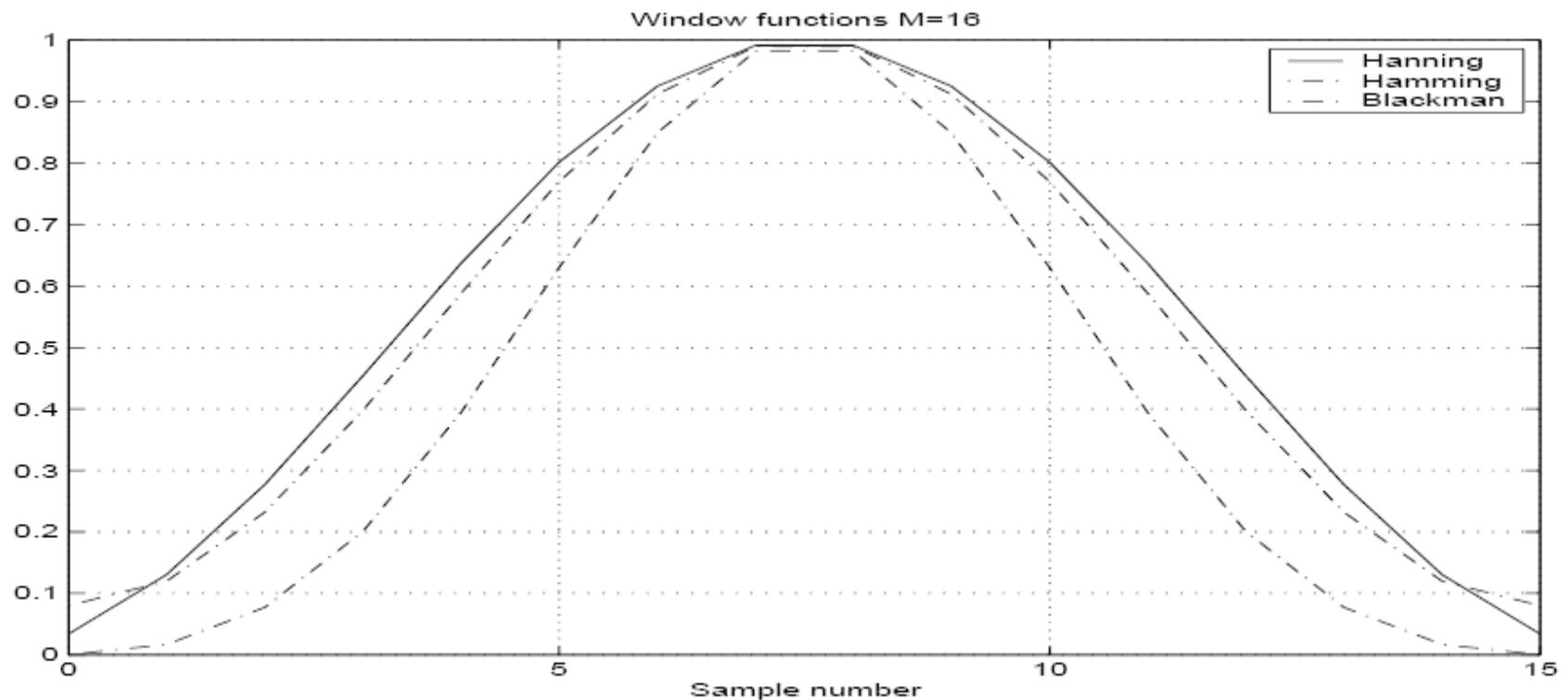
In this case, the **reduced ripple** comes at the expense of **a wider transition region** but this

However, this can be **compensated for by increasing the length** of the filter.

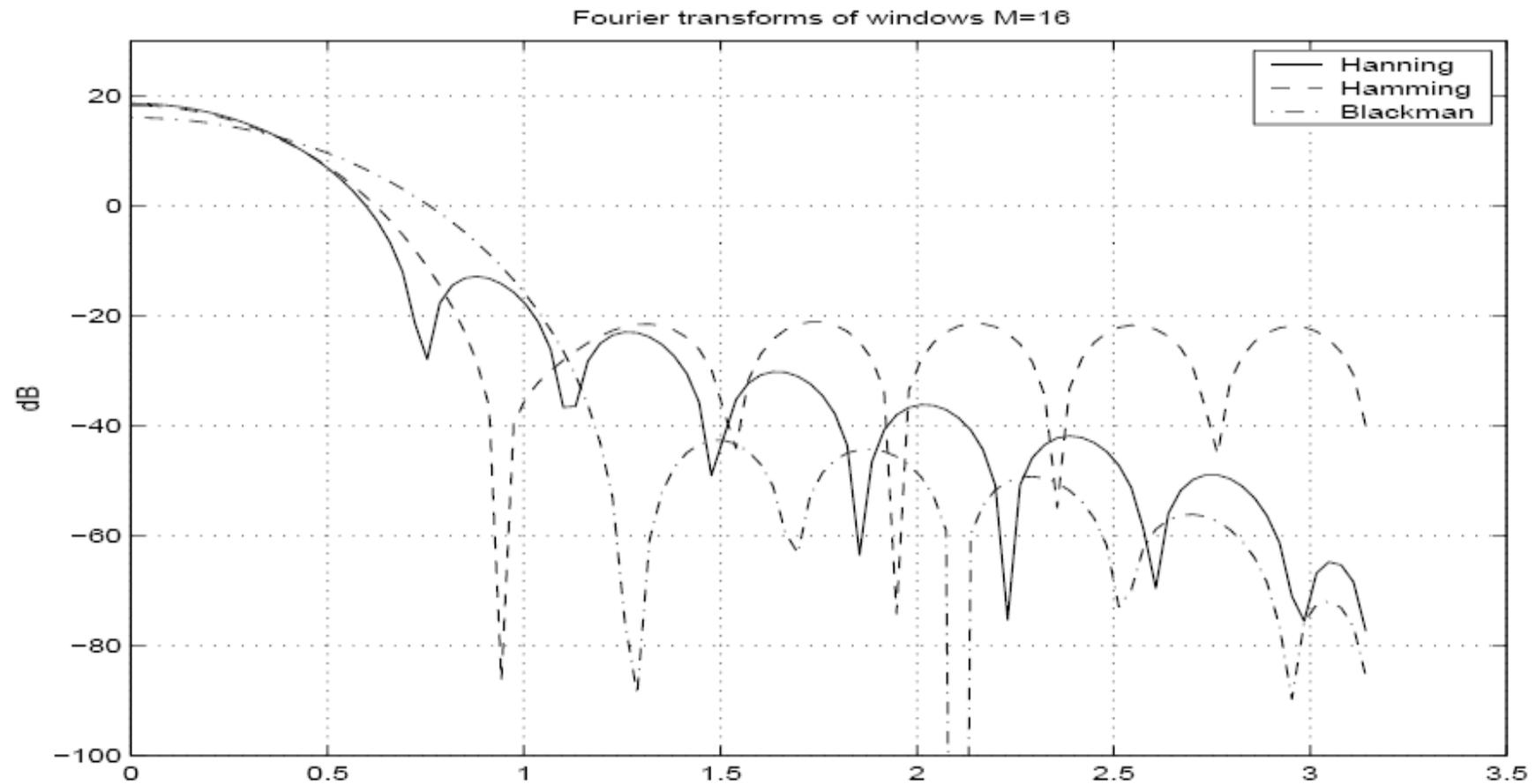
Alternative Windows –Time Domain

Many alternatives have been proposed, e.g.

- Hanning
- Hamming
- Blackman



Windows – Magnitude of Frequency Response

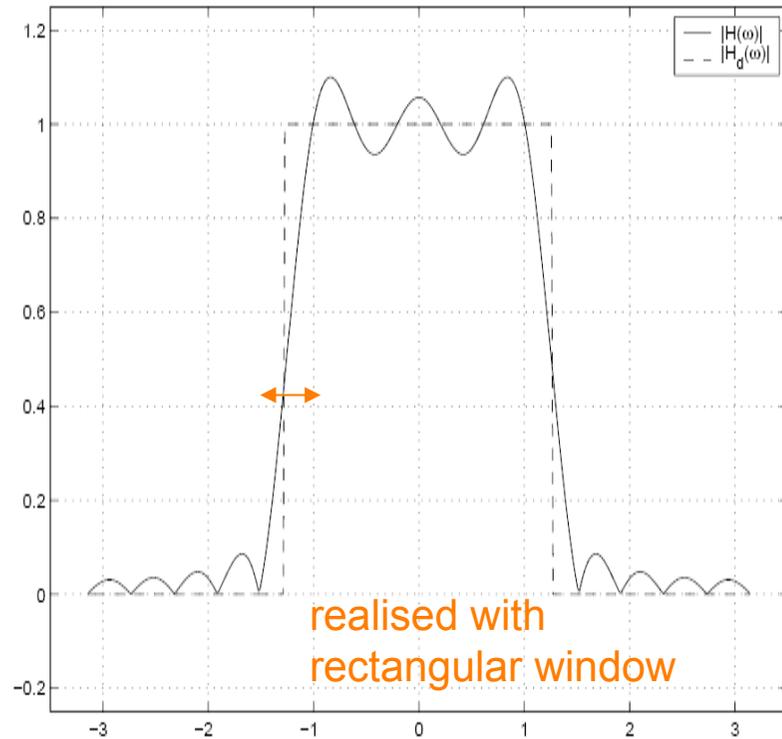


Summary of Windows Characteristics

Window's name	Mainlobe	Mainlobe/sidelobe	Peak $20\log_{10}\delta$
Rectangular	$4\pi/M$	-13dB	-21dB
Hanning	$8\pi/M$	-32dB	-44dB
Hamming	$8\pi/M$	-43dB	-53dB
Blackman	$12\pi/M$	-58dB	-74dB

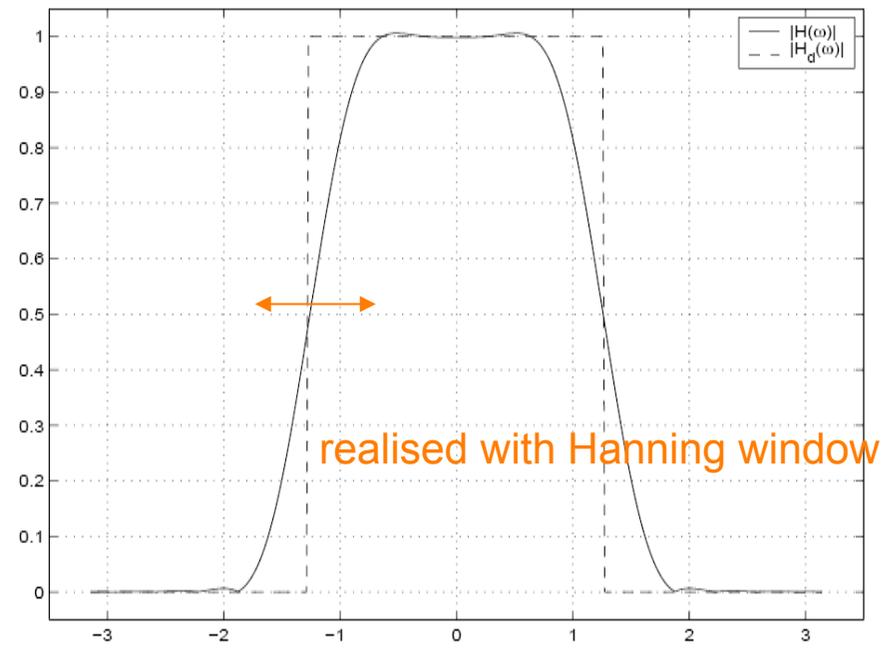
We see clearly that a wider transition region (wider main-lobe) is compensated by much lower side-lobes and thus less ripples.

Filter realised with rectangular/Hanning windows



$M=16$

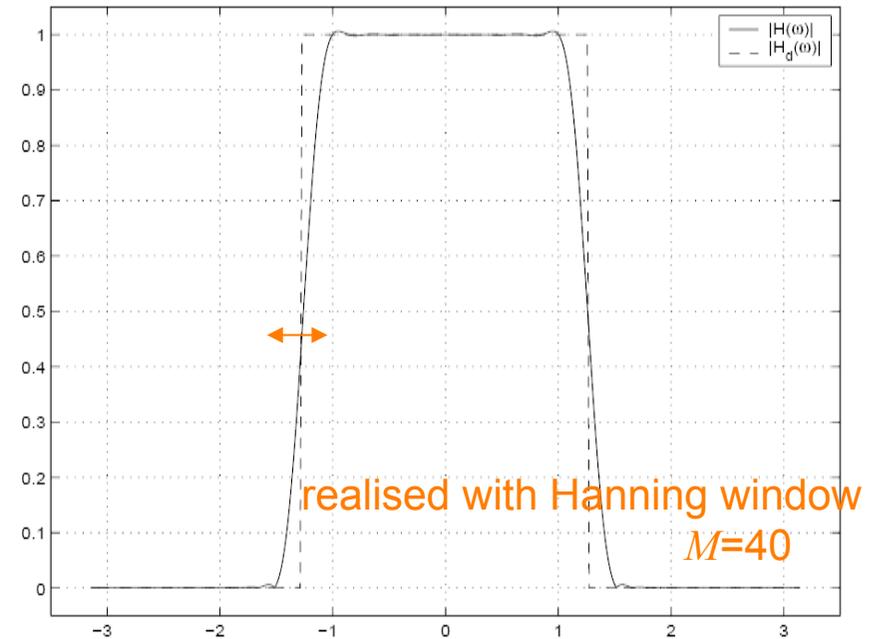
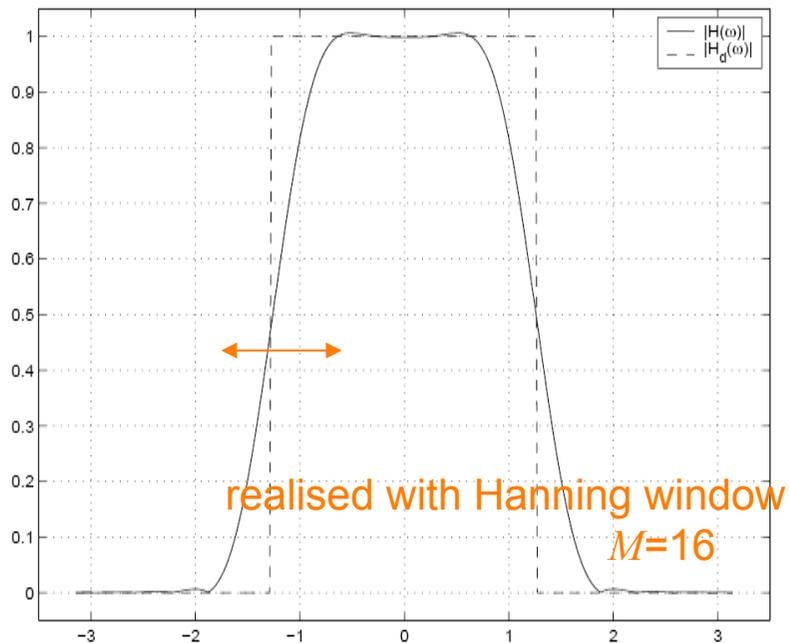
Back to our ideal filter



$M=16$

There are much less ripples for the Hanning window but that the transition width has increased

Filter realised with Hanning windows

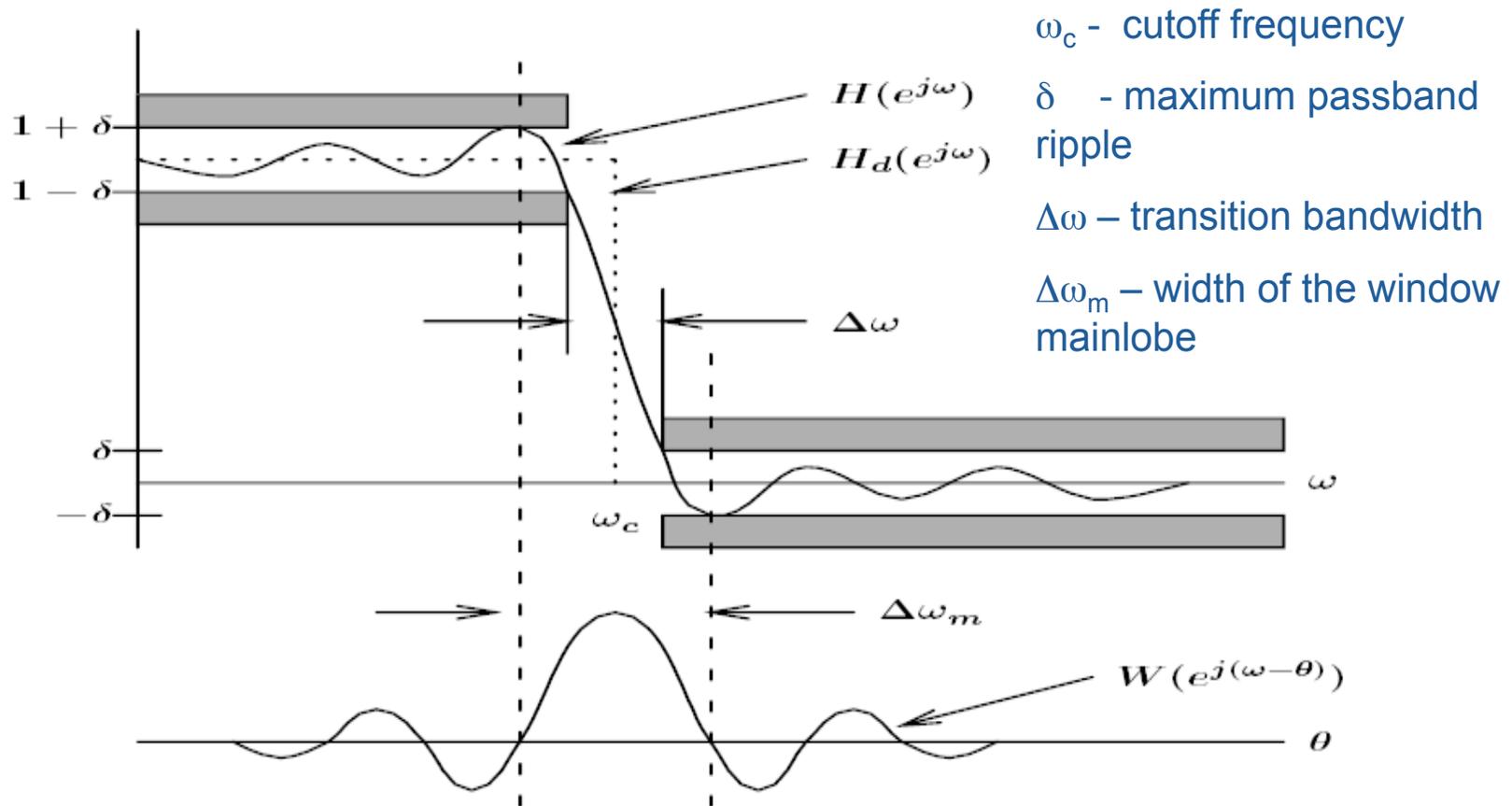


Transition width can be improved by increasing the size of the Hanning window to $M = 40$

Windows characteristics

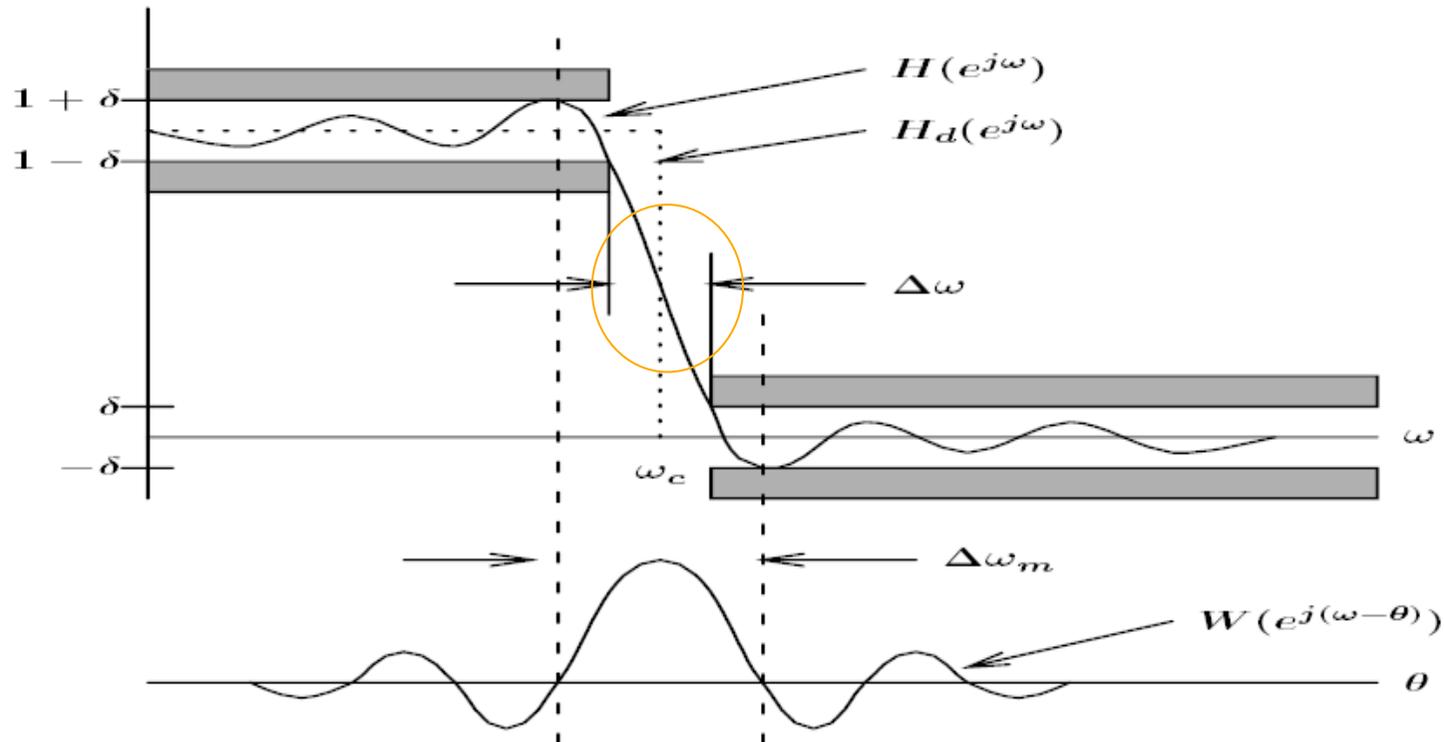
- Fundamental trade-off between main-lobe width and side-lobe amplitude
- As window *smoother*, peak *side-lobe decreases*, but the *main-lobe width increases*.
- Need to increase window length to achieve same transition bandwidth.

Specification necessary for Window Design Method



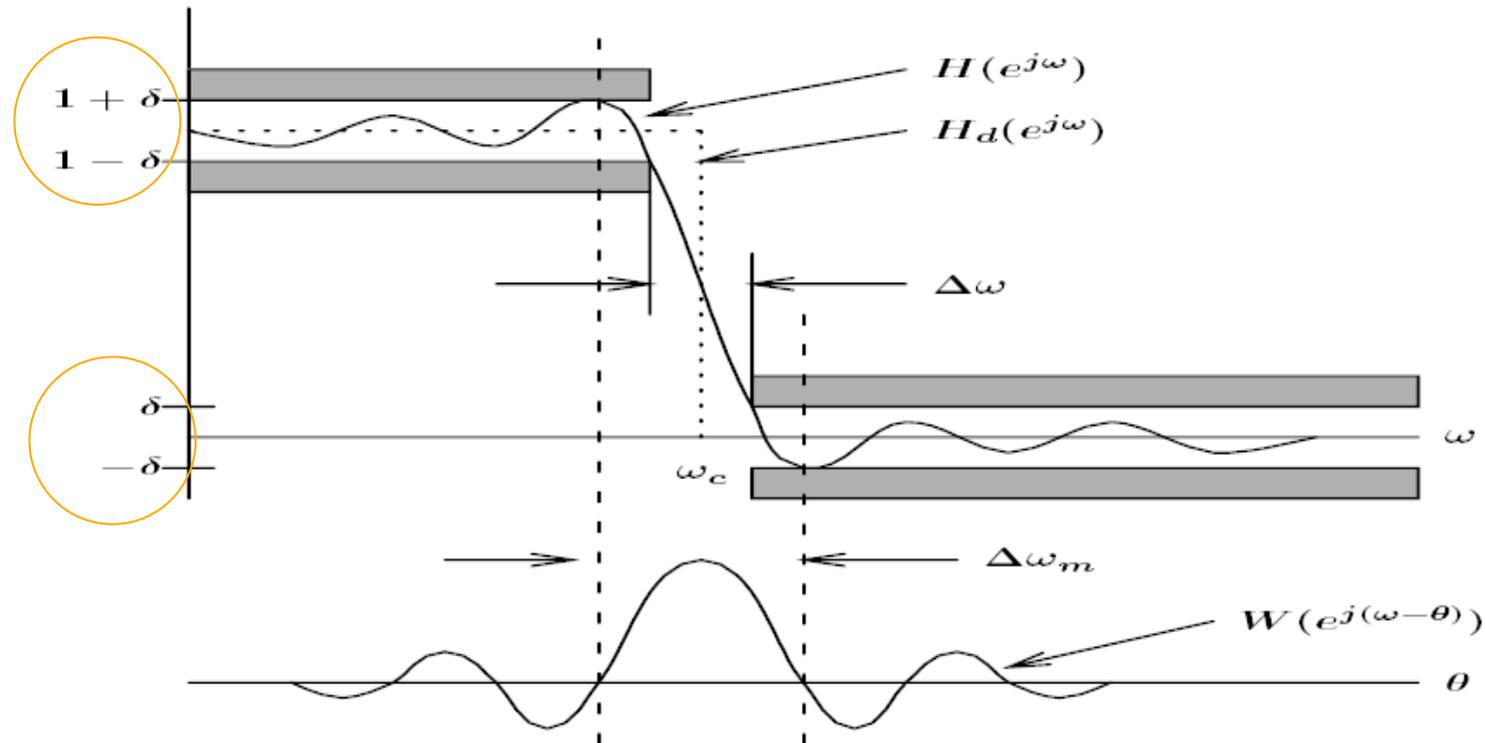
Response must not enter shaded regions

Key Property 1 of the Window Design Method



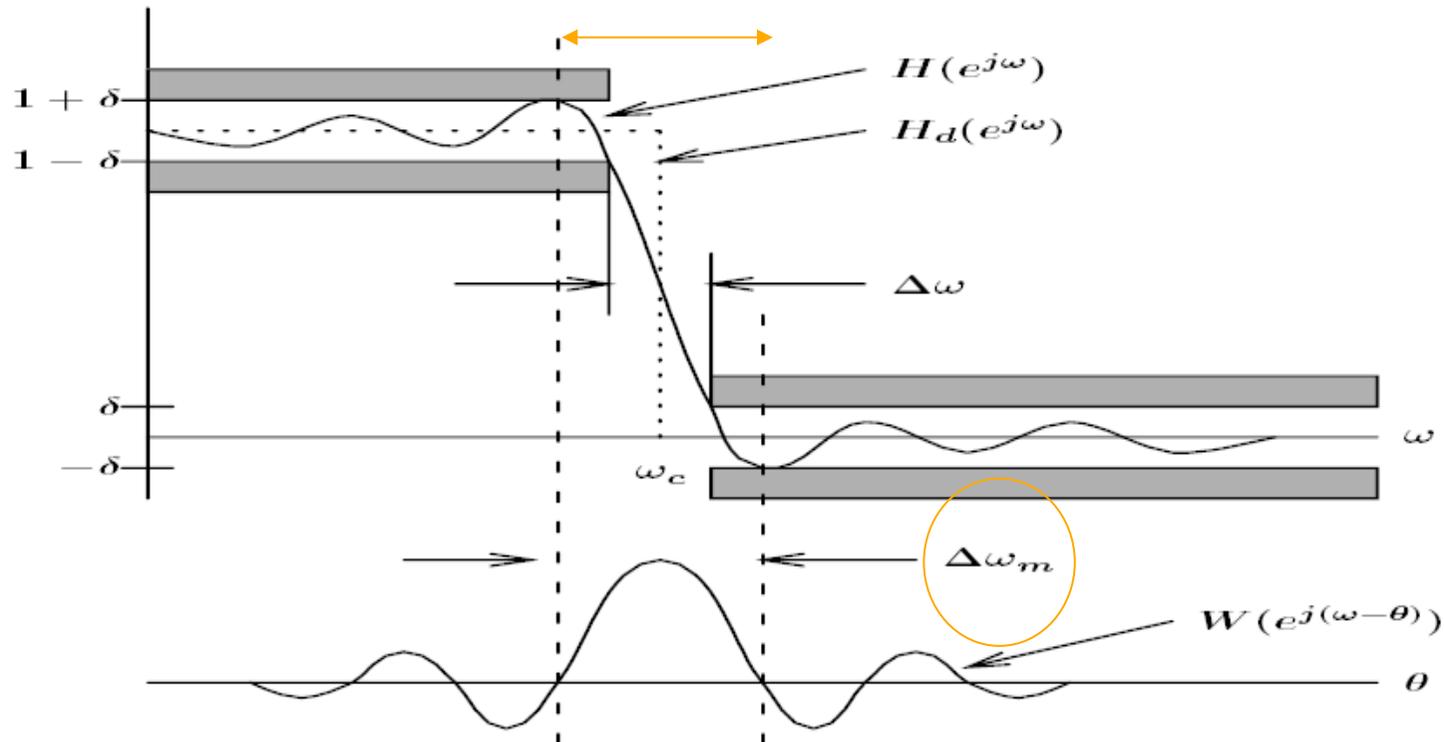
- *Property 1*: equal transition bandwidth on both sides of the ideal cutoff frequency.

Key Property 2 of the Window Design Method



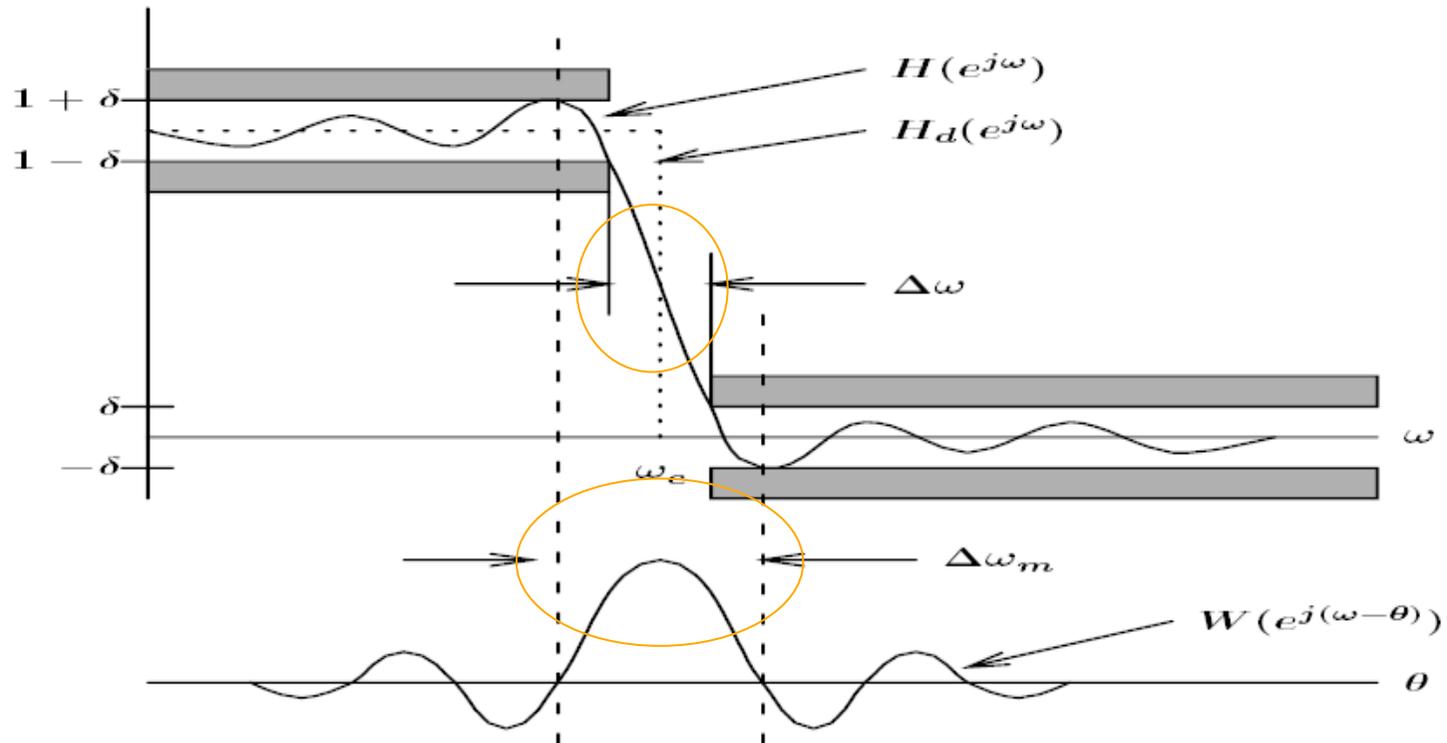
- *Property 2:* peak approximation error in the passband (δ) is equal to that in the stopband.

Key Property 3 of the Window Design Method



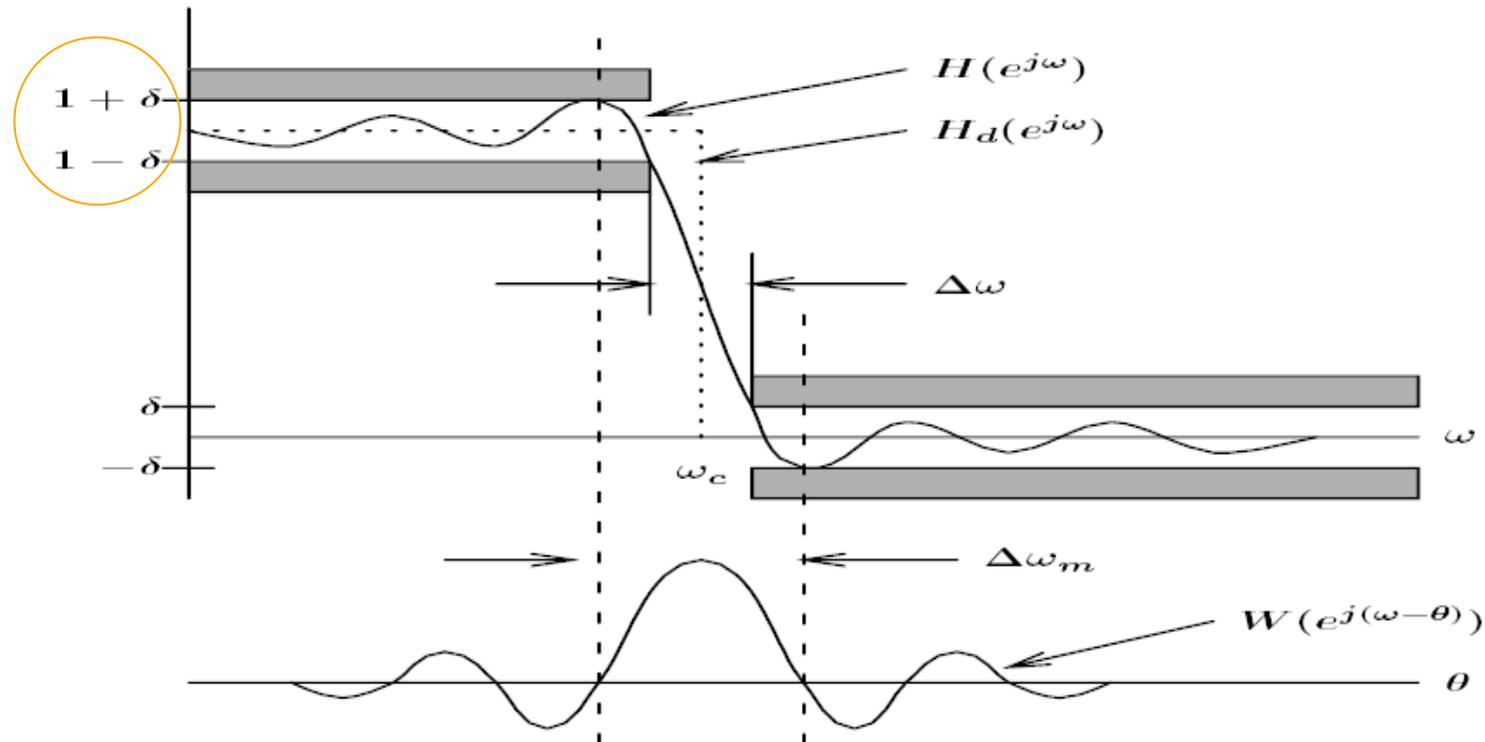
- *Property 3:* distance between approximation error peaks is approximately equal to the width of the mainlobe $\Delta\omega_m$.

Key Property 4 of the Window Design Method



- *Property 4*: the width of the mainlobe is wider than the transition bandwidth.

Key Property 5 of the Window Design Method



- *Property 5*: peak approximation error is determined by the window shape, independent of the filter order.

Passband / stopband ripples

Passband / stopband ripples are often expressed in dB:

passband ripple = $20 \log_{10} (1+\delta_p)$ dB,

or peak-to-peak passband ripple $\cong 20 \log_{10} (1+2\delta_p)$ dB;

minimum stopband attenuation = $-20 \log_{10} (\delta_s)$ dB.

Example: $\delta_p = 6\%$  peak-to-peak passband ripple $\cong 20 \log_{10} (1+2\delta_p) = 1$ dB;

$\delta_s = 0.01$  minimum stopband attenuation = $-20 \log_{10} (\delta_s) = 40$ dB.

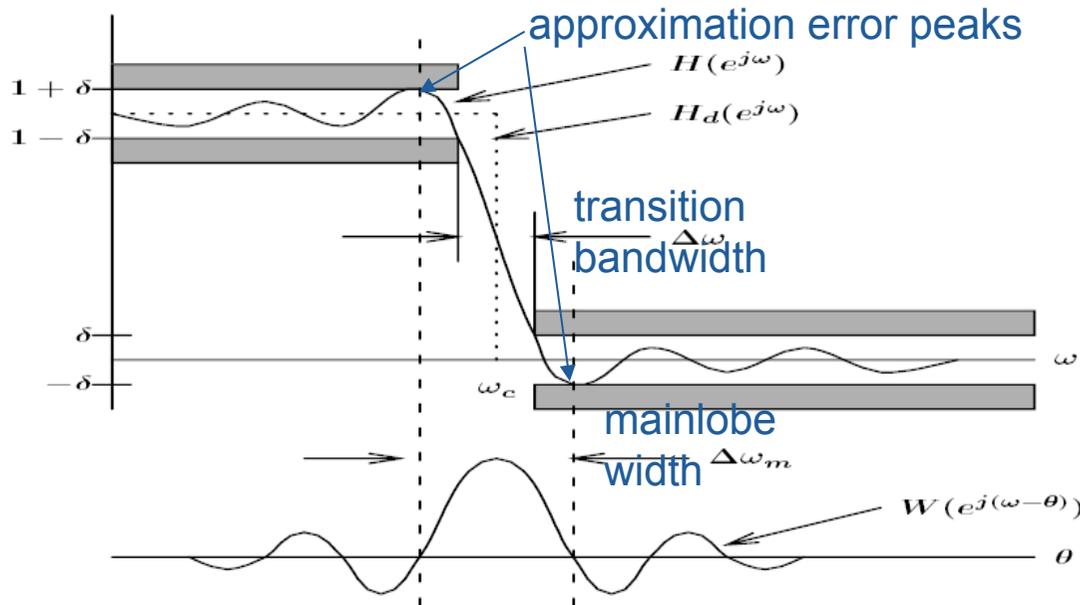
The band-edge frequencies ω_s and ω_p are often called **corner frequencies**, particularly when associated with specified gain or attenuation (e.g. gain = -3dB).

Summary of Window Design Procedure

- Ideal frequency response has infinite impulse response
- To be implemented in practice it has to be
 - truncated
 - shifted to the right (to make it causal)
- Truncation is just pre-multiplication by a rectangular window
 - the filter of a large order has a narrow transition band
 - however, sharp discontinuity results in side-lobe interference independent of the filter's order and shape **Gibbs phenomenon** 
- Windows with **no abrupt discontinuity** can be used to reduce Gibbs oscillations (e.g. Hanning, Hamming, Blackman)

Summary of the Key Properties of the Window Design Method

1. Equal transition bandwidth on both sides of the ideal cutoff frequency.
2. Equal peak approximation error in the pass-band and stop-band.
3. Distance between approximation error peaks is approximately equal to the width of the window main-lobe.
4. The width of the main-lobe is wider than the transition band.



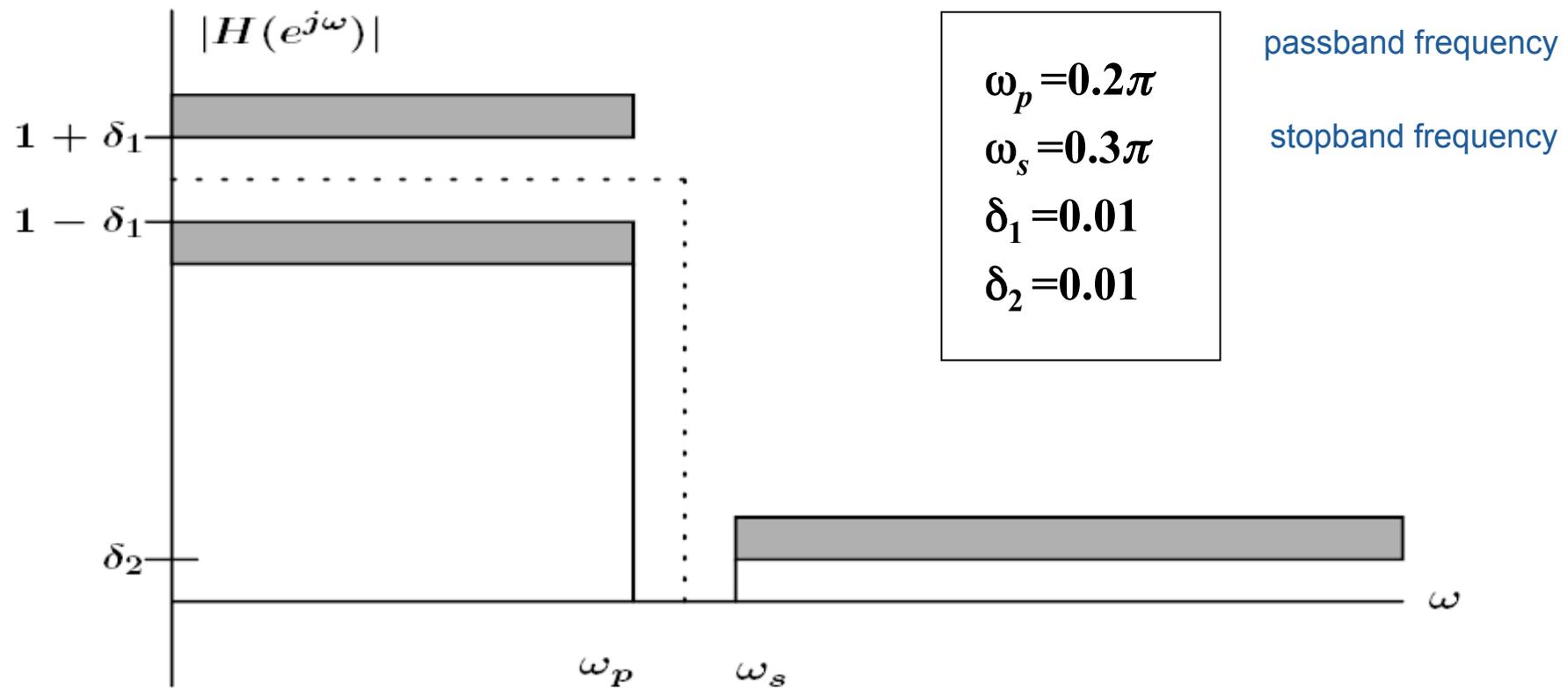
5. Peak approximation error is determined by the window shape, independent of the filter order.

Summary of the windowed FIR filter design procedure

1. Select a suitable window function
2. Specify an ideal response $H_d(\omega)$
3. Compute the coefficients of the ideal filter $h_d(n)$
4. Multiply the ideal coefficients by the window function to give the filter coefficients
5. Evaluate the frequency response of the resulting filter and iterate if necessary (typically, it means increase M if the constraints you have been given have not been satisfied)

Step by Step Windowed Filter Design Example

Design a type I low-pass filter according to the specification



Step 1. Select a suitable window function

Choosing a suitable window function can be done with the aid of published data such as

Window's name	Mainlobe	Mainlobe/sidelobe	Peak $20\log_{10}\delta$
Rectangular	$4\pi / M$	-13dB	-21dB
Hanning	$8\pi / M$	-32dB	-44dB
Hamming	$8\pi / M$	-43dB	-53dB
Blackman	$12\pi / M$	-58dB	-74dB

The required peak error spec $\delta_2 = 0.01$, i.e. $-20\log_{10}(\delta_s) = -40$ dB

➡ Hanning window

Main-lobe width $\omega_s - \omega_p = 0.3\pi - 0.2\pi = 0.1\pi$, i.e. $0.1\pi = 8\pi / M$ ➡

filter length $M \geq 80$, filter order $N \geq 79$

Type-I filter have even order ➡ $N = 80$

although for Hanning window first and last ones are 0 so only 78 in reality 107

Step 2 Specify the Ideal Response

Property 1: The band-edge frequency of the ideal response is the midpoint between ω_s and ω_p

$$\Rightarrow \omega_c = (\omega_s + \omega_p)/2 = (0.2\pi + 0.3\pi)/2 = 0.25\pi$$

$$H_d(\omega) = \begin{cases} 1 & \text{if } |\omega| \leq 0.25\pi \\ 0 & \text{if } 0.25\pi < |\omega| < \pi \end{cases}$$

our ideal low-pass filter
frequency response

Step 3 Compute the coefficients of the ideal filter

- The ideal filter coefficients h_d are given by the Inverse **Discrete time** Fourier transform of $H_d(\omega)$

$$\begin{aligned} h_d(n) &= \frac{1}{2\pi} \int_{-\pi}^{\pi} H_d(\omega) e^{j\omega n} d\omega = \frac{1}{2\pi} \int_{-\omega_c}^{\omega_c} e^{j\omega n} d\omega \\ &= \frac{\omega_c \sin \omega_c n}{\pi \omega_c n}. \end{aligned}$$

- Delayed impulse response (to make it causal)

$$\tilde{h}(n) = \hat{h}\left(n - \frac{N}{2}\right)$$

- Coefficients of the ideal filter

$$h(n) = \frac{\sin(0.5\pi(n - 40))}{\pi(n - 40)}$$

Step 3 Compute the coefficients of the ideal filter

- For our example this can be done analytically, but in general (for more complex $H_d(\omega)$ functions) it will be computed approximately using an N-point **Inverse Fast Fourier Transform (IFFT)**.
- Given a value of N (choice discussed later), create a sampled version of $H_d(\omega)$:

$$H_d(p) = H_d(2\pi p/N), \quad p=0,1,\dots,N-1.$$

[Note frequency spacing $2\pi/N$ rad/sample]

Step 3 Compute the coefficients of the ideal filter

If the Inverse FFT, and hence the filter **coefficients**, are to be purely **real-valued**, the frequency response must be **conjugate symmetric**:

$$H_d(-2\pi p/N) = H_d^*(2\pi p/N) \quad (1)$$

Since the Discrete Fourier Spectrum is also periodic, we see that

$$H_d(-2\pi p/N) = H_d(2\pi - 2\pi p/N) = H_d(2\pi(N-p)/N) \quad (2)$$

Equating (1) & (2) we must set $H_d(N-p) = H_d^*(p)$ for $p = 1, \dots, (N/2-1)$.

The Inverse FFT of $H_d^*(p)$ is an N -sample time domain function $h'(n)$.

For $h'(n)$ to be an accurate approximation of $h(n)$, N **must be made large enough to avoid time-domain aliasing** of $h(n)$, as illustrated below.

Time domain aliasing

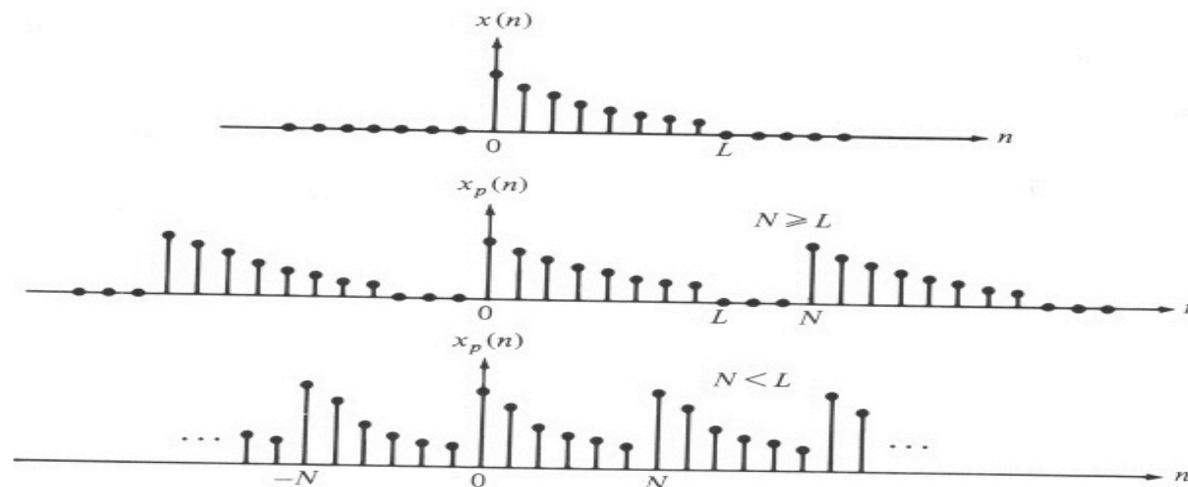
Consider FFT and IFFT

$$X_p = \sum_{n=0}^{N-1} x_n e^{-j\frac{2\pi}{N}np}, \quad p \in \{0, 1, \dots, N-1\} \quad (1)$$

$$x_n = \frac{1}{N} \sum_{p=0}^{N-1} X_p e^{j\frac{2\pi}{N}np}, \quad n \in \{0, 1, 2, \dots, N-1\} \quad (2)$$

The relationship (2) provides the reconstruction of the periodic signal x_n however it does not imply that we can recover x_n from the samples.

For sequence x_n of finite duration L this is only possible if $N \geq L$



Step 4 Multiply to obtain the filter coefficients

- Coefficients of the ideal filter

$$h(n) = \frac{\sin(0.5\pi(n-40))}{\pi(n-40)}$$

- Multiplied by a Hamming window function

$$w(n) = \begin{cases} 0.54 - 0.46 \cos(2\pi n/M), & 0 \leq n \leq M \\ 0, & \text{otherwise} \end{cases}$$

Step 5 Evaluate the Frequency Response and Iterate

The frequency response is computed as the DFT of the filter coefficient vector.

If the resulting filter does not meet the specifications, one of the following could be done

- adjust the ideal filter frequency response (for example, move the band edge) and repeat from step 2
- adjust the filter length and repeat from step 4
- change the window (and filter length) and repeat from step 4

Matlab Implementation of the Window Method

Two methods **FIR1** and **FIR2**

B=FIR2 (N , F , M)

Designs a **Nth order** FIR digital filter

F and **M** specify frequency and magnitude breakpoints for the filter such that **plot (N , F , M)** shows a plot of desired frequency

The frequencies **F** must be in increasing order between **0** and **1**, with **1** corresponding to half the sample rate.

B is the vector of length **N+1**, it is real, has linear phase and symmetric coefficients

Default window is Hamming – others can be specified

Multi-band Design

- So far, only lowpass filter: how do we design highpass, bandpass, etc. filters?
⇒ treat them as sums and differences of lowpass filters.
- *Example:* design the following highpass filter

$$H_d(\omega) = 1_{(-\pi, -\omega_c) \cup (\omega_c, \pi)}(\omega)$$

It can be rewritten as

$$H_d(\omega) = \underbrace{1_{(-\pi, \pi)}(\omega)}_{\text{LP filter cutoff } \pi} - \underbrace{1_{(-\omega_c, \omega_c)}(\omega)}_{\text{LP filter cutoff } \omega_c} \Rightarrow h(n) = \frac{\sin(\pi n)}{\pi n} - \frac{\sin(\omega_c n)}{\pi n}.$$

- Now we use window and delay this answer by $M/2$ to make it causal.

Frequency sampling method

- Drawbacks of the window design method:
 - Start with $H_d(\omega)$ and end up with approximation $H(\omega)$, difficult to predict values of $H(\omega)$ at some specific frequencies (not big problem...)
 - Computation of the IDTFT of arbitrary $H_d(\omega)$ may be difficult.
- *Simple idea:* sample the desired frequency response $H_d(\omega)$ at N frequencies unif. spaced over $[0, 2\pi)$, compute the IDFT of these N samples $\Rightarrow \{h(n)\}$.
- **Advantage:** The filter frequency response lands exactly on the specified values at the sampling points.
- **Drawback:** Difficult to control between those points (i.e. sinc-interpolated).

FIR Filter Design Using Windows

FIR filter design based on windows is simple and robust, however, it is not optimal:

- The resulting pass-band and stop-band parameters are equal even though often the specification is more strict in the stop band than in the pass band
 - ➔ unnecessary high accuracy in the pass band
- The ripple of the window is not uniform (decays as we move away from discontinuity points according to side-lobe pattern of the window)
 - ➔ by allowing more freedom in the ripple behaviour we may be able to reduce filter's order and hence its complexity

FIR Design by Optimisation: Least-Square Method

We now present a method that approximates the desired frequency response by a [linear-phase FIR](#) amplitude function according to the following optimality criterion.

The difference between the ideal filter $H_d(\omega)$ and $H(\omega)$ is measured as

$$E(\omega) = W(\omega) [H_d(\omega) - H(\omega)]$$

where $W(\omega)$ is a weighting function that allows the relative error of approximation between different bands to be defined.

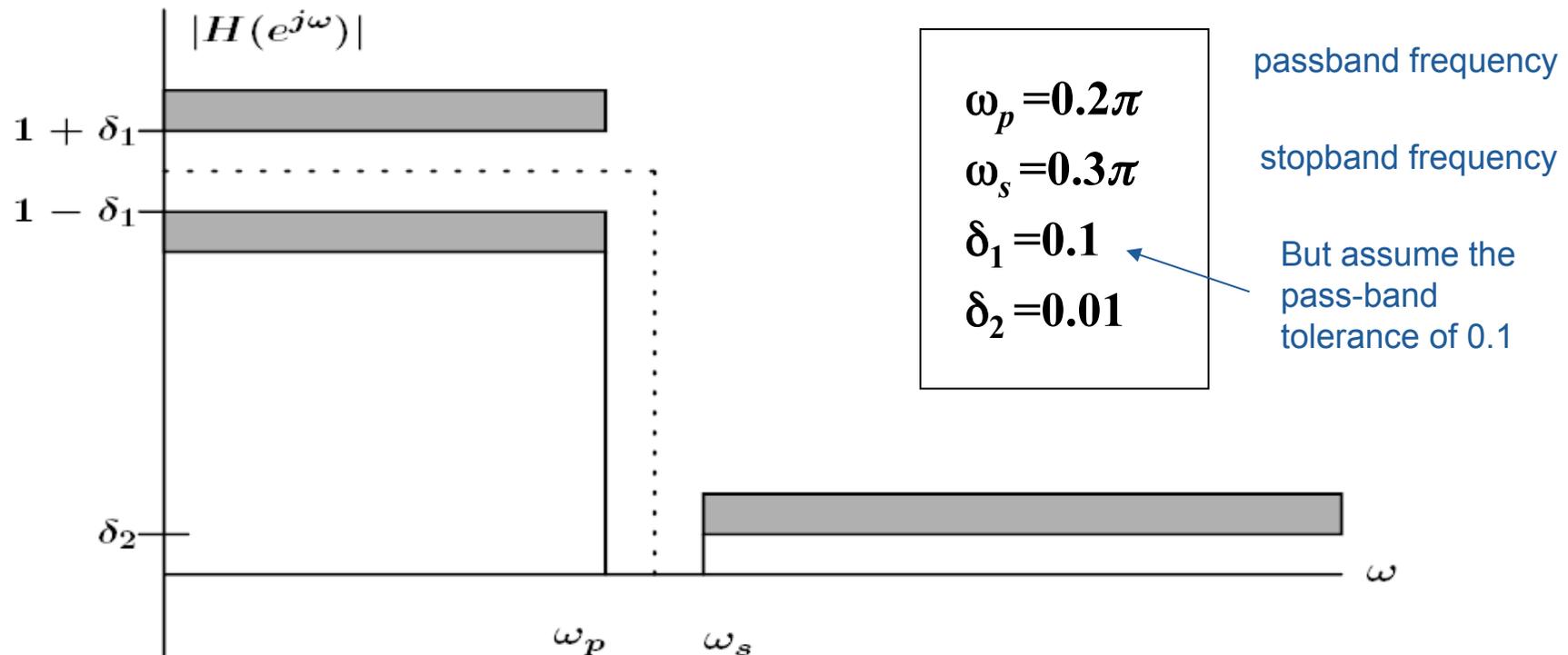
The integral of the weighted square frequency-domain error is given by

$$\varepsilon^2 = \int E^2(\omega) d\omega$$

and we assume that the order and the type of the filter are known. Under this assumptions designing the FIR filter now reduces to determining the coefficients that would minimise ε^2 .

Recall Our Example

Design a type I low-pass filter according to specification



The filter designed using window method cannot benefit from this relaxation, however, a least-square method design gives $N = 33$ (compared to $N = 80$).

Least-Square Design of FIR Filters

- Meeting the specification is not guaranteed a-priori, trial and error is often required. It might be useful to set the transition bands slightly narrower than needed, and it is often necessary to experiment with the weights
- Occasionally the resulting frequency response may be peculiar. Again, changing the weights would help to resolve the problem

Equiripple Design

The least-square criterion of minimising

$$\varepsilon^2 = \int E^2(\omega) d\omega$$

is not entirely satisfactory.

A better approach is to minimize the maximum error at each band

$$\varepsilon = \max_{\omega} |E(\omega)|$$

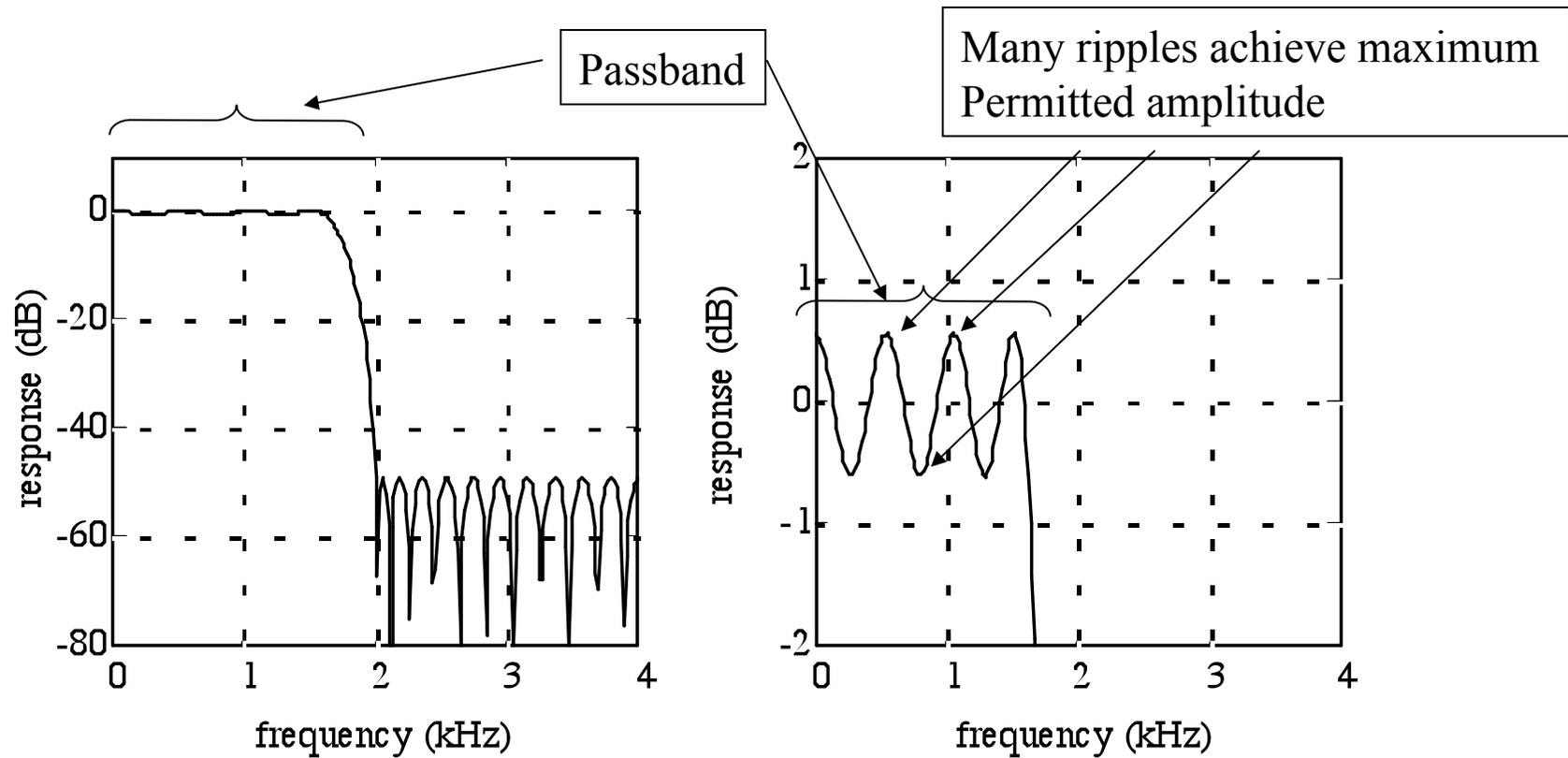
Equiripple Design

The method is optimal in a sense of minimising the maximum magnitude of the ripple in all bands of interest, the filter order is fixed

$$\min_{H_d} \max_{\omega} |E(\omega)|.$$

It can be shown that this leads to an *equiripple* filter – a filter which amplitude response oscillates uniformly between the tolerance bounds of each band

Equiripple Design



Overall and passband-only frequency response of length 37 minimax filter

Remez method

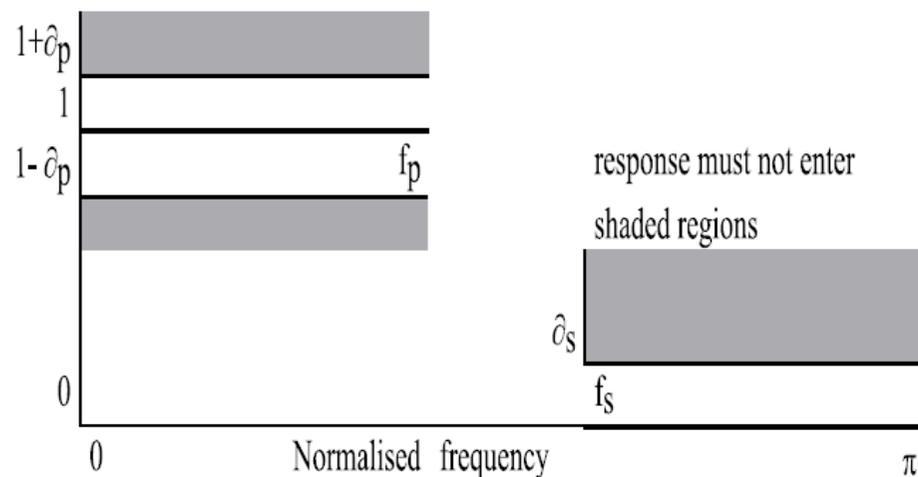
- There exists a computational procedure known as the Remez method to solve this mathematical optimization problem.
- There are also exist formulae for estimating the required filter length in the case of lowpass, bandpass and narrow transition bandwidths. However, these formulae are not always reliable so it might be necessary to iterate the procedure so as to satisfy the design constraints.

Equiripple Design: Weights

The weights can be determined in advance from a minimax specification.

For example, if a simple lowpass filter has a requirement for the passband gain to be in the range $1-\delta_p$ to $1+\delta_p$, and the stopband gain to be less than δ_s , the weightings given to the passband and stopband errors would be δ_s and δ_p respectively.

The detailed algorithm is beyond the (time!) constraints of this module.



Equiripple Design: Example

Obtain the coefficients of an FIR lowpass digital filter to meet these specifications:

passband edge frequency	1.625 kHz
passband pk-to-pk ripple	<1 dB
transition width	0.5 kHz
stopband attenuation	>50 dB
sampling frequency	8 kHz

The passband ripple corresponds to $\pm 6\%$, while the stopband attenuation is 0.32%, hence the weighting factors are set to 0.32 and 6.

Using the relevant length estimation formula gives order $N=25.8$ hence $N=26$ was chosen, i.e. length =27. This proved to be substantially too short, and it was necessary to increase the order to 36 (length 37) to meet the specifications.

Equiripple Design: Matlab

`b = remez(n, f, m)` designs an n th order FIR digital filter and returns the filter coefficients in length $n+1$ vector `b`.

Vectors `f` and `m` specify the frequency and magnitude breakpoints [*as for FIR2*].

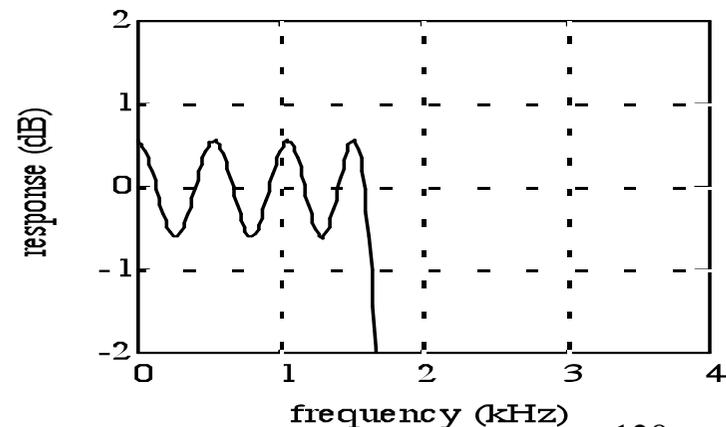
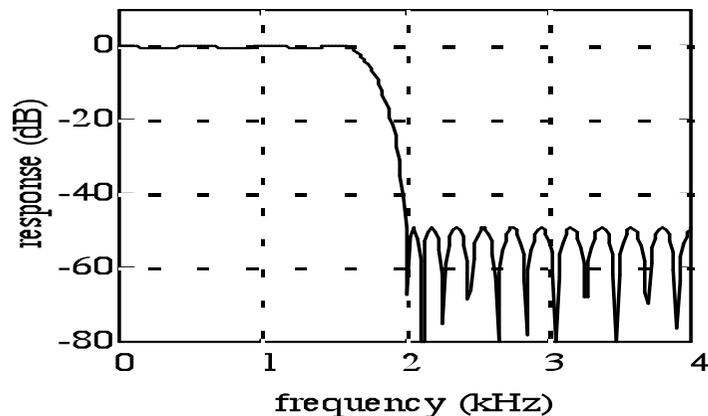
`b = remez(n, f, m, w)` uses vector `w` to specify weighting in each of the pass or stop bands in vectors `f` and `m`.

Note again the frequency normalisation, where 1.0 equals **half** the sample rate.

The call which finally met this filter specification was:

```
h = remez(36, [0 1.625 2 4]/4, [1 1 0 0], [0.32 6]);
```

The resulting frequency response is as shown previously:



The Parks-McClellan Remez exchange algorithm

The computational procedure the optimization problem is by Remez

The algorithm in common use is by Parks and McClellan.

The Parks-McClellan Remez exchange algorithm is widely available and versatile.

Important: it **designs linear phase (symmetric) filters** or **antisymmetric filters** of any of the standard types

Linear Symmetric Filters

The **frequency response** of the direct form FIR filter may be **rearranged** by **grouping** the terms involving the **first and last** coefficients, the **second and next to last**, etc.:

$$\begin{aligned}
 H(\exp(j\Omega)) &= \sum_{k=0}^M b_k \exp(-jk\Omega) \\
 &= b_0 \exp(-j0) + b_M \exp(-jM\Omega) \\
 &\quad + b_1 \exp(-j\Omega) + b_{M-1} \exp(-j(M-1)\Omega)
 \end{aligned}$$

and then taking out a common factor $\exp(-jM\Omega/2)$:

$$\begin{aligned}
 H(\exp(j\Omega)) &= \exp(-jM\Omega/2) \\
 &\quad \times \left\{ b_0 \exp(jM\Omega/2) + b(M) \exp(-jM\Omega/2) \right. \\
 &\quad \quad + b_1 \exp(j(M-2)\Omega/2) + b_{M-1} \exp(-j(M-2)\Omega/2) \\
 &\quad \quad \left. + \dots \right\}
 \end{aligned}$$

If the filter length $M+1$ is odd, then the final term in curly brackets above is the single term $b_{M/2}$, that is the centre coefficient ('tap') of the filter.

Symmetric impulse response

Symmetric impulse response: if we put $b_M = b_0$, $b_{M-1} = b_1$, etc., and note that $\exp(j\theta) + \exp(-j\theta) = 2\cos(\theta)$, the frequency response becomes

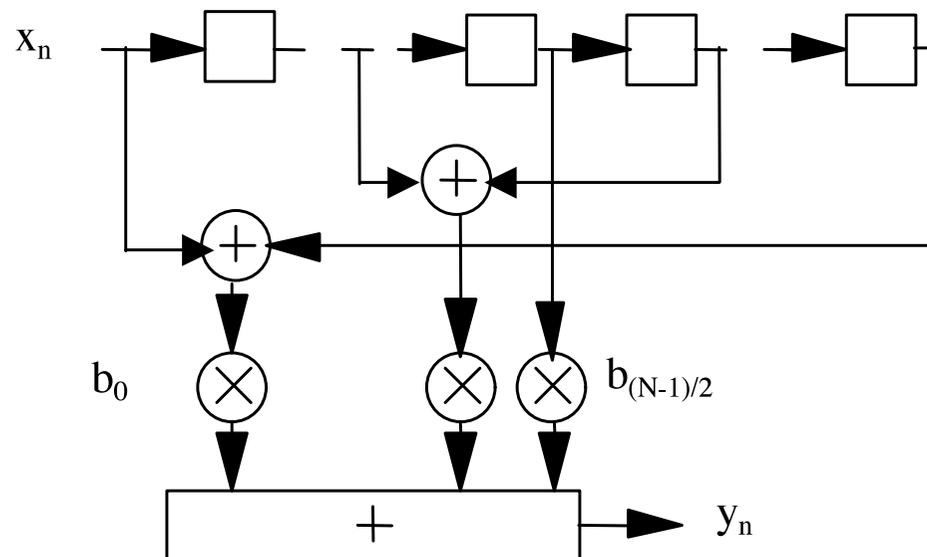
$$H(\exp(j\Omega)) = 2 \exp(-jM\Omega/2) \times \left\{ \begin{aligned} &b_0 \cos(M\Omega/2) \\ &+ b_1 \cos((M-2)\Omega/2) \\ &+ \dots \end{aligned} \right\}$$

This is a **purely real** function (sum of cosines) **multiplied by a linear phase** term, hence the **response has linear phase**, corresponding to a pure delay of $M/2$ samples, ie half the filter length.

A similar argument can be used to simplify **antisymmetric** impulse responses in terms of a sum of sine functions (such filters do not give a pure delay, although the phase still has a linear form $\pi/2 - m\Omega/2$)

Implementation of symmetric FIR filters

The symmetric FIR filters of length N can be implemented using the folded delay line structure shown below, which uses $N/2$ (or $(N+1)/2$) multipliers rather than N

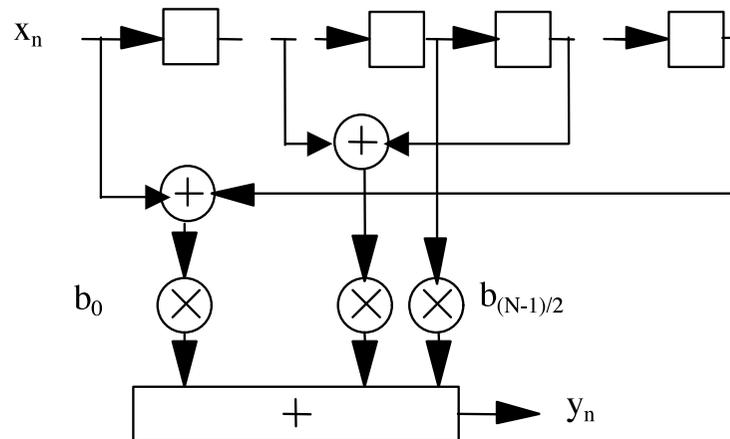


Limitations of the algorithm

Linear phase in the stopbands is never a real requirement, and in some applications strictly linear phase in the passband is not needed either.

The linear phase filters designed by this method are therefore longer than optimum non-linear phase filters.

However, symmetric FIR filters of length N can be implemented using the **folded delay line structure** shown below, which uses $N/2$ (or $(N+1)/2$) multipliers rather than N , so the longer symmetric filter may be no more computationally intensive than a shorter non-linear phase one.



Further options for FIR filter design

More general non-linear optimisation (least squared error or minimax) can of course be used to design linear or non-linear phase FIR filters to meet more general frequency and/or time domain requirements.

Matlab has suitable optimisation routines.