http://elec3004.com

Digital Filters

ELEC 3004: **Digital Linear Systems**: Signals & Controls
Dr. Surya Singh

Lecture 8

elec3004@itee.uq.edu.au
http://robotics.itee.uq.edu.au/~elec3004/

April 29, 2014

Lecture Schedule:

Week	Date	Lecture Title
1	4-Mar	Introduction & Systems Overview
	6-Mar	[Linear Dynamical Systems]
2	11-Mar	Signals as Vectors & Systems as Maps
	13-Mar	[Signals]
3	18-Mar	Sampling & Data Acquisition & Antialiasing Filters
	20-Mar	[Sampling]
4	25-Mar	System Analysis & Convolution
	27-Mar	[Convolution & FT]
5	1-Apr	Frequency Response & Filter Analysis
	3-Apr	[Filters]
6	8-Apr	Discrete Systems & Z-Transforms
	10-Apr	[Z-Transforms]
7	15-Apr	Introduction to Digital Control
	17-Apr	[Feedback]
8	29-Apr	Digital Filters
	1-May	[Digital Filters]
9	6-May	Digital Control Design
	8-May	[Digitial Control]
10	13-May	Stability of Digital Systems
	15-May	[Stability]
11	20-May	State-Space
	22-May	Controllability & Observability
12	27-May	PID Control & System Identification
	29-May	Digitial Control System Hardware
13	3-Jun	Applications in Industry & Information Theory & Communications
	5-Jun	Summary and Course Review

Linear, Discrete Dynamical Systems

Linear Difference Equations

$$u_k = f(e_0, \ldots, e_k; u_0, \ldots, u_{k-1}).$$

$$u_k = -a_1 u_{k-1} - a_2 u_{k-2} - \cdots - a_n u_{k-n} + b_0 e_k + b_1 e_{k-1} + \cdots + b_m e_{k-m}.$$

$$\begin{aligned}
\nabla u_k &= u_k - u_{k-1} && \text{(first difference)},\\
\nabla^2 u_k &= \nabla u_k - \nabla u_{k-1} && \text{(second difference)},\\
\nabla^n u_k &= \nabla^{n-1} u_k - \nabla^{n-1} u_{k-1} && (n\text{th difference}).
\end{aligned}$$

$$\begin{aligned}
u_k &= u_k,\\
u_{k-1} &= u_k - \nabla u_k,\\
u_{k-2} &= u_k - 2\nabla u_k + \nabla^2 u_k.
\end{aligned}$$

$$a_2 \nabla^2 u_k - (a_1 + 2a_2)\nabla u_k + (a_2 + a_1 + 1)u_k = b_0 e_k.$$

Assume a form of the solution

z^k :

- k: "order of difference"
- k: delay

$$Az^k = Az^{k-1} + Az^{k-2}.$$

$$1 = z^{-1} + z^{-2}$$

$$z^2 = z + 1.$$

DT Causality & BIBO Stability

- Causality:

$$h[n] = 0,\ n < 0$$

$$\rightarrow y[n] = \sum_{k=0}^{\infty} h[k]x[n-k] \quad \text{or} \quad \Rightarrow y[n] = \sum_{k=-\infty}^{n} x[k]h[n-k]$$

- Input is Causal if: $x[n] = 0,\ n < 0$
- Then output is Causal:

$$y[n] = \sum_{k=0}^{n} h[k]x[n-k] = \sum_{k=0}^{n} x[k]h[n-k]$$

- And, DT LTI is BIBO stable if:

$$\sum_{k=-\infty}^{\infty} |h[k]| < \infty$$

Impulse Response (Graphically)

Let's define the *impulse response*, $h[n]$, as the result of applying an LTI system to the unit impulse:

$$\delta[n] \longrightarrow \boxed{\text{LTI System}} \longrightarrow h[n]$$

By time invariance, we know

$$\delta[n-k] \longrightarrow \boxed{\text{LTI System}} \longrightarrow h[n-k]$$

And by linearity, we know

$$\alpha_1\delta[n-k_1] + \alpha_2\delta[n-k_2] \longrightarrow \boxed{\text{LTI System}} \longrightarrow \alpha_1 h[n-k_1] + \alpha_2 h[n-k_2]$$

$$u[n] \longrightarrow \boxed{\text{LTI System}} \longrightarrow y[n]$$

$$u[n] = \sum_{k=-\infty}^{\infty} u[k]\delta[n-k]$$

$$\boxed{y[n] = \sum_{k=-\infty}^{\infty} u[k]h[n-k]}$$

∞ matrix × ∞ vector?

How do you multiply an infinite matrix?

- First let's multiply circulant matrices…
  - A circulant matrix can be descibed completely by its first row or column

$$A = \begin{bmatrix} a_0 & a_1 & a_2 & \cdots & a_{n-1} \\ a_{n-1} & a_0 & a_1 & \cdots & a_{n-2} \\ a_{n-2} & a_{n-1} & a_0 & \cdots & a_{n-3} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ a_1 & a_2 & a_3 & \cdots & a_0 \end{bmatrix} = \begin{bmatrix} | & | & | & & | \\ h & Zh & Z^2h & \cdots & Z^{N-1}h \\ | & | & | & & | \end{bmatrix}$$

Z: Shift operator

- Multiply by u[k] ➔

$$\begin{bmatrix} | & | & | & & | \\ h & Zh & Z^2h & \cdots & Z^{N-1}h \\ | & | & | & & | \end{bmatrix} \begin{bmatrix} u[0] \\ u[1] \\ u[2] \\ \vdots \\ u[N-1] \end{bmatrix} = \sum_{k=0}^{N-1} u[k]Z^k h$$

∴ For circulant matrices, matrix multiplication reduces to a weighted combination of shifted impulse responses

Two Types of Systems

- Linear shift-invariant:

$$y = \sum_{k=0}^{N-1} u[k] Z^k h$$

Z: Shift operator

$$Z \cdot [u_0, u_1, u_2, u_3, \ldots, u_{n-1}]^T = [u_{n-1}, u_0, u_1, u_2, \ldots, u_{n-2}]^T$$

- Linear time-invariant system

$$y = \sum_{k=-\infty}^{\infty} u[k] R^k h$$

R: Unit delay operator

$$R \cdot [\ldots, u_0, u_1, u_2, u_3, \ldots]^T = [\ldots, u_{-1}, u_0, u_1, \ldots]^T$$

Impulse Response of Both Types

$$y[n] = \frac{1}{2}u[n-1] + \frac{1}{2}u[n]$$

$$\begin{aligned} y[-1] &= 0 \\ y[0] &= \frac{1}{2} \\ y[1] &= \frac{1}{2} \\ y[2] &= 0 \\ &\vdots \end{aligned}$$

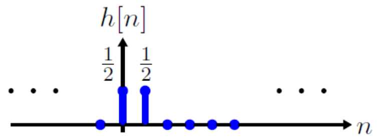
$$y[n] = \frac{1}{2}y[n-1] + u[n]$$

$$\begin{aligned} h[-1] &= 0 \\ h[0] &= 1 \\ h[1] &= \frac{1}{2} \\ h[2] &= \frac{1}{4} \\ &\vdots \end{aligned}$$

$$h[n] = \begin{cases} 0 & n < 0 \\ (\frac{1}{2})^n & n \geq 0 \end{cases}$$

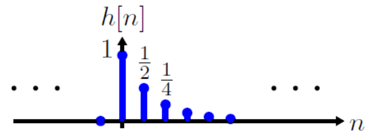
Impulse Response of Both Types

$$y[n] = \frac{1}{2}u[n-1] + \frac{1}{2}u[n]$$



'Finite impulse response" (FIR)

$$y[n] = \frac{1}{2}y[n-1] + u[n]$$



"Infinite impulse response" (IIR)

➔ Digital Filters

- Wikipedia Says:

A **digital filter** is a system that performs mathematical operations on a sampled, discrete-time signal to reduce or enhance certain aspects of that signal.

- Basically we have a transfer function or
  … a difference equation

In the Z-domain:

$$H(z) = \frac{B(z)}{A(z)} = \frac{b_0 + b_1 z^{-1} + b_2 z^{-2} + \cdots + b_N z^{-N}}{1 + a_1 z^{-1} + a_2 z^{-2} + \cdots + a_M z^{-M}}$$

- This is a recursive form with inputs (Numerator) and outputs (Denominator)
  ➔ "IIR infinite impulse response" behaviour
- If the denominator is made equal to unity (i.e. no feedback)

➔ then this becomes an FIR or finite impulse response filter.

➔ Digital Filters Types

FIR

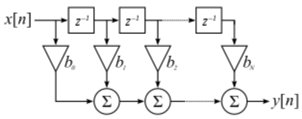
From H(z):

→ $$\begin{aligned} H(\omega) &= h_0 + h_1 e^{-\mathrm{i}\omega} + \cdots + h_{n-1} e^{-\mathrm{i}(n-1)\omega} \\ &= \sum_{t=0}^{n-1} h_t \cos t\omega - \mathrm{i} \sum_{t=0}^{n-1} h_t \sin t\omega \end{aligned}$$

→ Filter becomes a "multiply, accumulate, and delay" system:

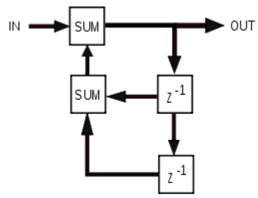
$$y(t) = \sum_{\tau=0}^{n-1} h_\tau u(t-\tau)$$

$$y[n] = b_0 x[n] + b_1 x[n-1] + \cdots + b_N x[n-N]$$



IIR

- Impulse response function that is non-zero over an infinite length of time.



FIR Properties

- Require no feedback.
- Are inherently stable.
- They can easily be designed to be linear phase by making the coefficient sequence symmetric
- Flexibility in shaping their magnitude response
- Very Fast Implementation (based around FFTs)

- The main disadvantage of FIR filters is that considerably more computation power in a general purpose processor is required compared to an IIR filter with similar sharpness or selectivity, especially when low frequency (relative to the sample rate) cutoffs are needed.

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)

➔From H(z):

$$\begin{aligned} H(\omega) &= h_0 + h_1 e^{-\mathrm{i}\omega} + \cdots + h_{n-1} e^{-\mathrm{i}(n-1)\omega} \\ &= \sum_{t=0}^{n-1} h_t \cos t\omega - \mathrm{i} \sum_{t=0}^{n-1} h_t \sin t\omega \end{aligned}$$

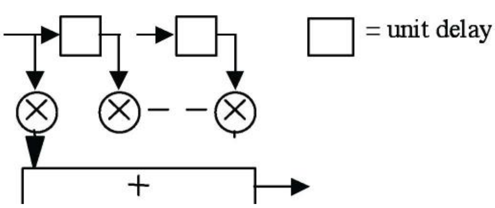
∵ H(ω) is periodic and conjugate

∴ <u>Consider ω ∈ [0, π]</u>

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 Impulse Response

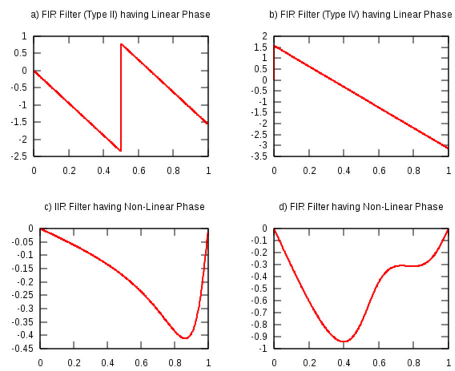
Obtain the impulse response immediately with $x(n) = \delta(n)$:

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

- The impulse response is of finite length *M* (good!)
- FIR filters have only zeros (no poles) (as they must, N=0 !!)
  - Hence known also as **all-zero** filters
- FIR filters also known as **feedforward** or **non-recursive,** or **transversal** filters

FIR & Linear Phase

- The phase response of the filter is a linear function of frequency
- Linear phase has constant group delay, all frequency components have equal delay times. ∴ No distortion due to different time delays of different frequencies

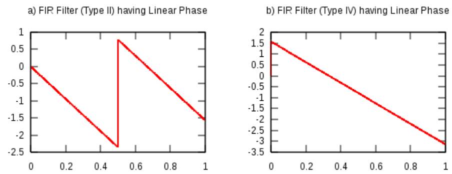


Ref: Wikipedia (Linear Phase)

- FIR Filters with:

$$\sum_{n=-\infty}^{\infty} h[n] \cdot \sin(\omega \cdot (n-\alpha) + \beta) = 0$$

FIR & Linear Phase → Four Types



Ref: Wikipedia (Linear Phase)

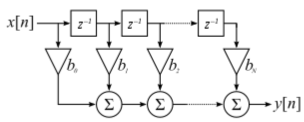
Impulse response	# coefs	$H(\omega)$	Type
$h(n) = h(M-1-n)$	Odd	$e^{-j\omega(M-1)/2}\left(h\left(\frac{M-1}{2}\right) + 2\sum_{k=1}^{(M-3)/2} h\left(\frac{M-1}{2} - k\right)\cos(\omega k)\right)$	1
$h(n) = h(M-1-n)$	Even	$e^{-j\omega(M-1)/2} 2\sum_{k=1}^{(M-3)/2} h\left(\frac{M}{2} - k\right)\cos\left(\omega\left(k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

- 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, 0 at $\omega=0$
  (not suitable as a high-pass)

FIR Filter Design

- How to get all these coefficients?

$$H(\omega) = \boxed{h_0 + h_1 e^{-i\omega} + \cdots + h_{n-1} e^{-i(n-1)\omega}}$$

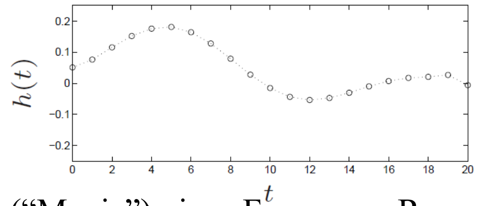


FIR Design Methods:

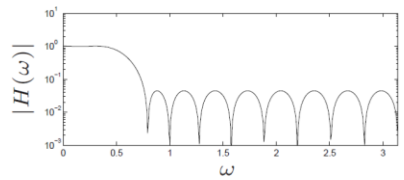
1. Impulse Response Truncation
   + Simplest
   – Undesirable frequency domain-characteristics, not very useful
2. Windowing Design Method
   + Simple
   – Not optimal (not minimum order for a given performance level)
3. Optimal filter design methods
   + "More optimal"
   – Less simple…

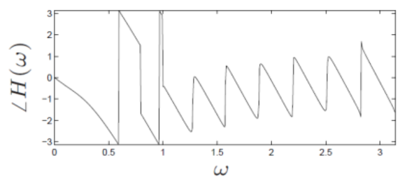
FIR Filter Design & Operation
Ex: Lowpass FIR filter

- Set Impulse response (order n = 21)
- “Determine” h(t)
  - h(t) is a 20 element vector that we’ll use to as a weighted sum



- FFT (“Magic”) gives Frequency Response & Phase





Why is this “hard”? Looking at the Low-Pass Example

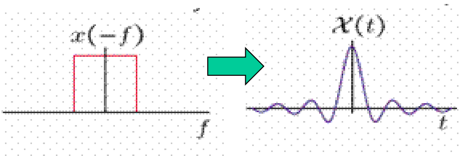
$$H_d(\omega) = \begin{cases} 1 \text{ if } |\omega| \leq \omega_c \\ 0 \text{ if } \omega_c < |\omega| < \pi \end{cases}$$

- Why is this hard?
  - Shouldn’t it be “easy” ??
    … just hit it with some FFT “magic” and then keep the bands we want and then hit it with some Inverse-FFT “supermagic”???
  - Remember we need a “system” that does this “rectangle function” in frequency
  - Let’s consider what that means…
    - It basically suggests we need an **Inverse FFT** of a **“rectangle function”**

Flashback: Fourier Series & Rectangular Functions

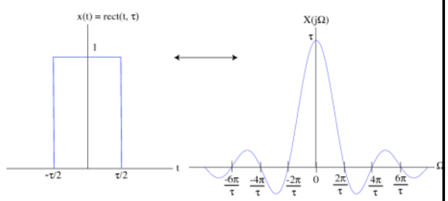
$\mathfrak{F}$: Fourier Tranform

$$\mathfrak{F}^{-1}\left\{rect\left(\frac{\omega}{2}\right)\right\} = \frac{sinc\,(t)}{\pi}$$



Ref: http://cnx.org/content/m26719/1.1/
http://www.wolframalpha.com/input/?i=IFFT%28sinc%28f%29%29

$$\mathfrak{F}\{rect\,(t)\} = sinc\left(\frac{\omega}{2}\right)$$



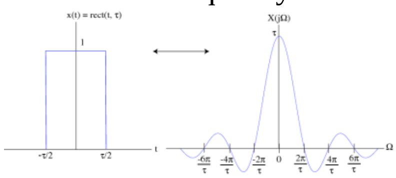
Ref: http://cnx.org/content/m32899/1.8/
http://www.thefouriertransform.com/pairs/box.php

See:

- Table 7.1 (p. 702) Entry 17 & Table 9.1 (p. 852) Entry 7

Flashback: Fourier Series & Rectangular Functions [2]

- The sinc function might look familiar
  - This is the frequency content of a square wave (box)



Ref: http://www.wolframalpha.com/input/?i=FFT%28rect%28t%29%29
http://cnx.org/content/m32899/1.8/

- This also applies to **signal reconstruction!**

➔ Whittaker–Shannon interpolation formula
  - This says that the "better way" to go from Discrete to Continuous (i.e. D to A) is not ZOH, but rather via the sinc!

$$x(t) = \sum_{n=-\infty}^{\infty} x[n] \cdot \mathrm{sinc}\left(\frac{t-nT}{T}\right)$$

∴ FIR and Low Pass Filters…

∴

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

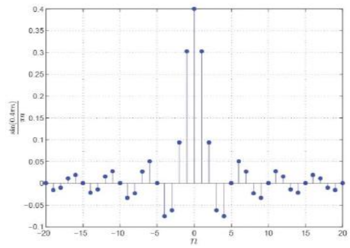
Has impulse response:

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

Thus, to filter an impulse train with an ideal low-pass filter use:

$$x(t) = \left(\sum_{n=-\infty}^{\infty} x[n] \cdot \delta(t - nT)\right) * \text{sinc}\left(\frac{t}{T}\right)$$

- **<u>However</u>**!!
  a sinc is non-causal and infinite in duration



And, this **cannot** be implemented **in practice** ☹

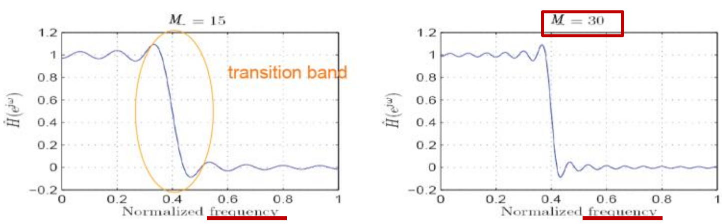
∵ we need to know all samples of the input, both in the **past** and in the **future**

Plan 0: Impulse Response Truncation

Maybe we saw this coming…

∴ Clip off the sinc at some large *n*

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



- *Ripples* in both passband/stopband
  and the transition not abrupt (i.e., a *transition band*).
- As M→∞, transition band→ 0 (as expected!)

➔ FIR Filters: Window Function Design Method

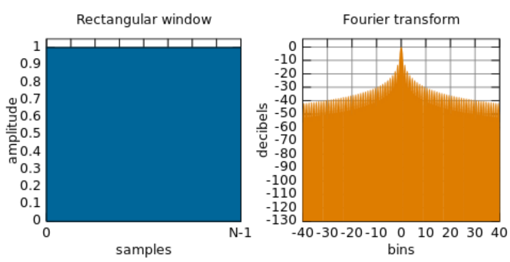
- Windowing: a generalization of the truncation idea

- There many, many "window" functions:
  - Rectangular
  - Triangular
  - Hanning
  - Hamming
  - Blackman
  - Kaiser
  - Lanczos
  - Many More … (see: http://en.wikipedia.org/wiki/Window_function)

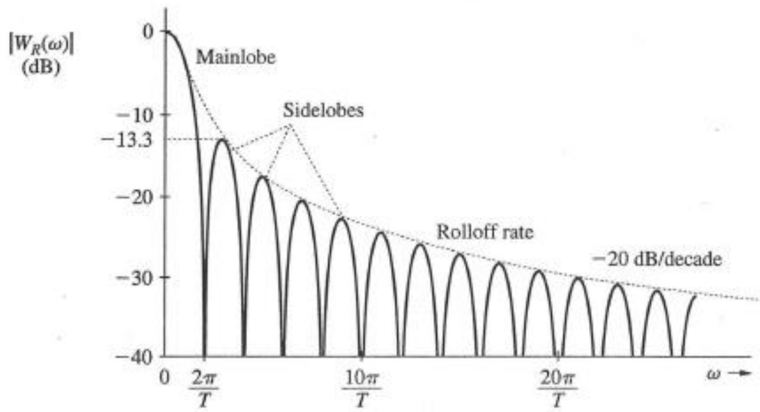
Some Window Functions [1]

1. Rectangular

$$w(n) = 1$$



Windowing and its effects/terminology

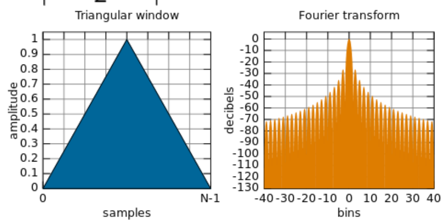


Lathi, Fig. 7.45

Some More Window Functions …

2. Triangular window

$$w(n) = 1 - \left| \frac{n - \frac{N-1}{2}}{\frac{N+1}{2}} \right|$$



- And Bartlett Windows
  - A slightly narrower variant with zero weight at both ends:

$$w(n) = 1 - \left| \frac{n - \frac{N-1}{2}}{\frac{N-1}{2}} \right|$$

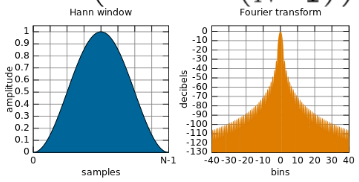
Some More Window Functions…

3. Generalized Hamming Windows

$$w(n) = \alpha - \beta \cos\left(\frac{2\pi n}{N-1}\right)$$

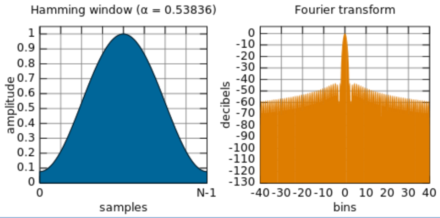
→ Hanning Window

$$\rightarrow w(n) = 0.5 \left(1 - \cos\left(\frac{2\pi n}{N-1}\right)\right)$$



→ Hamming's Window

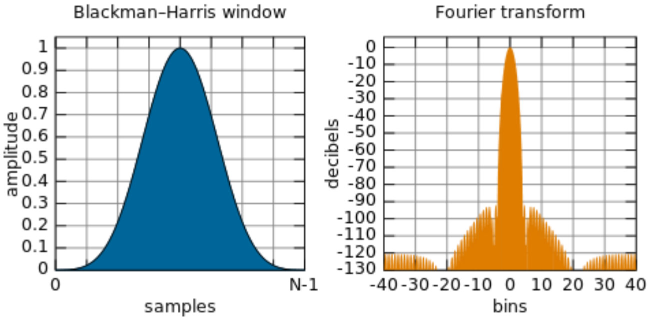
$$\rightarrow \alpha = 0.54,\ \beta = 1 - \alpha = 0.46$$



Some More Window Functions…

4. Blackman–Harris Windows
   - A generalization of the Hamming family,
   - Adds more shifted sinc functions for less side-lobe levels

$$w(n) = a_0 - a_1 \cos\left(\frac{2\pi n}{N-1}\right) + a_2 \cos\left(\frac{4\pi n}{N-1}\right) - a_3 \cos\left(\frac{6\pi n}{N-1}\right)$$

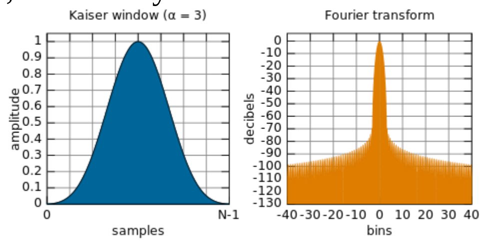


Some More Window Functions…

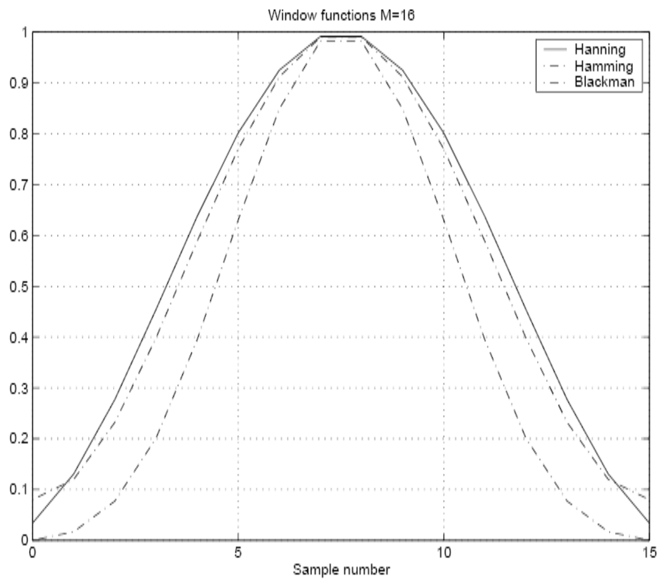
5. Kaiser window
   – A DPSS (discrete prolate spheroidal sequence)
   – Maximize the energy concentration in the main lobe

$$\rightarrow w(n) = \frac{I_0\left(\pi\alpha\sqrt{1-(\frac{2n}{N-1}-1)^2}\right)}{I_0(\pi\alpha)}$$

   – Where: I_0 is the zero-th order modified Bessel function of the first kind, and usually $\alpha = 3$.



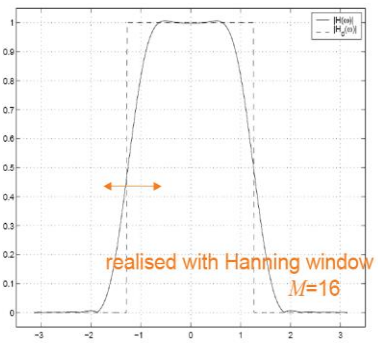
Comparison of Alternative Windows –Time Domain

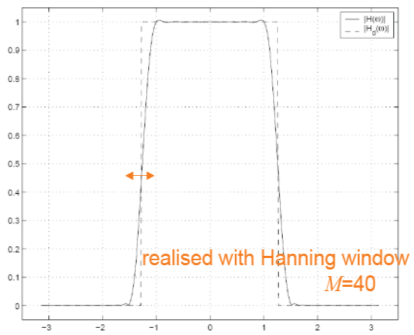


Punskaya, Slide 90

Adding Order

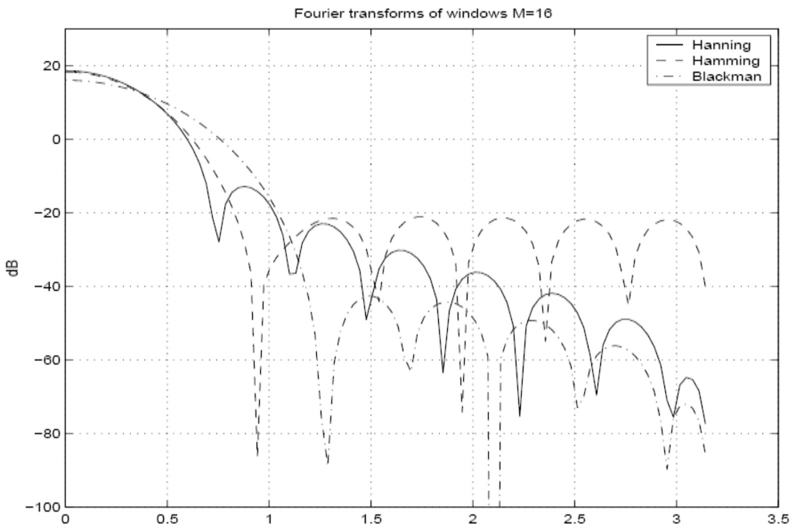
+ Transition and Smoothness
– Increased Size





Punskaya, Slide 94

Comparison of Alternative Windows Frequency Domain



Punskaya, Slide 91

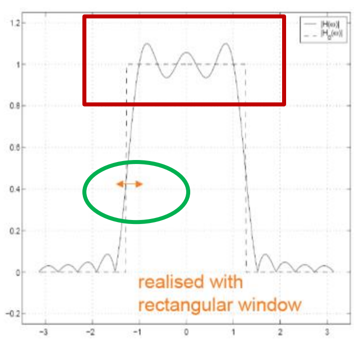
Summary Characteristics of Common Window Functions

No.	Window $w(t)$	Mainlobe Width	Rolloff Rate (dB/oct)	Peak Sidelobe level (dB)	Peak $20\log_{10}\delta$
1	Rectangular: $\text{rect}\left(\frac{t}{T}\right)$	$\frac{4\pi}{T}$	-6	-13.3	-21dB
2	Bartlett: $\Delta\left(\frac{t}{2T}\right)$	$\frac{8\pi}{T}$	-12	-26.5	
3	Hanning: $0.5\left[1+\cos\left(\frac{2\pi t}{T}\right)\right]$	$\frac{8\pi}{T}$	-18	-31.5	-44dB
4	Hamming: $0.54+0.46\cos\left(\frac{2\pi t}{T}\right)$	$\frac{8\pi}{T}$	-6	-42.7	-53dB
5	Blackman: $0.42+0.5\cos\left(\frac{2\pi t}{T}\right)+0.08\cos\left(\frac{4\pi t}{T}\right)$	$\frac{12\pi}{T}$	-18	-58.1	-74dB
6	Kaiser: $\dfrac{I_0\left[\alpha\sqrt{1-4\left(\frac{t}{T}\right)^2}\right]}{I_0(\alpha)}$ $0 \le \alpha \le 10$	$\frac{11.2\pi}{T}$	-6	-59.9 ($\alpha = 8.168$)	

Lathi, Table 7.3
Punskaya, Slide 92

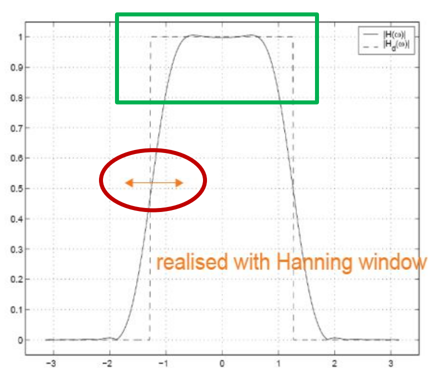
FIR: Rectangular & Hanning Windows

- Rectangular



M=16

- Hanning

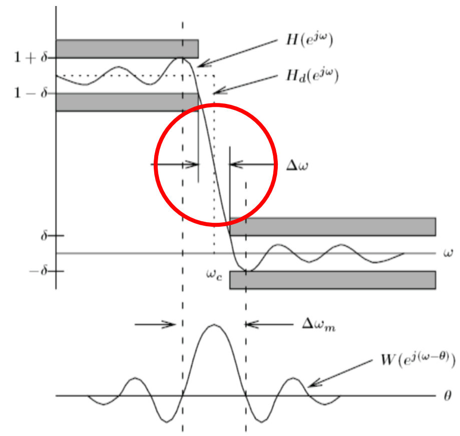


M=16

➔ Hanning: Less ripples, but wider transition band

Punskaya, Slide 93

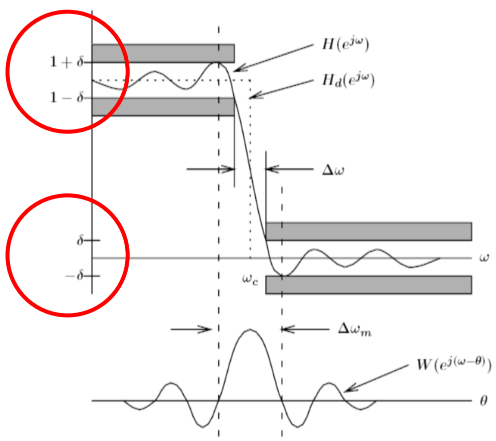
Windowed FIR Property 1:
Equal transition bandwidth



Punskaya, Slide 96

- Equal transition bandwidth on both sides of the ideal cutoff frequency

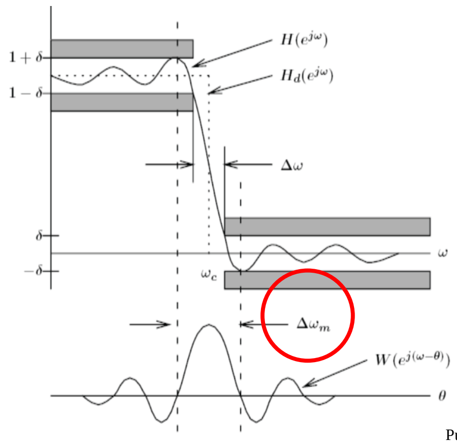
Windowed FIR Property 2:
Peak Errors same in Passband & Stopband



Punskaya, Slide 96

- Peak approximation error in the passband ($1+\delta \rightarrow 1-\delta$) is equal to that in the stopband ($\delta \rightarrow -\delta$)

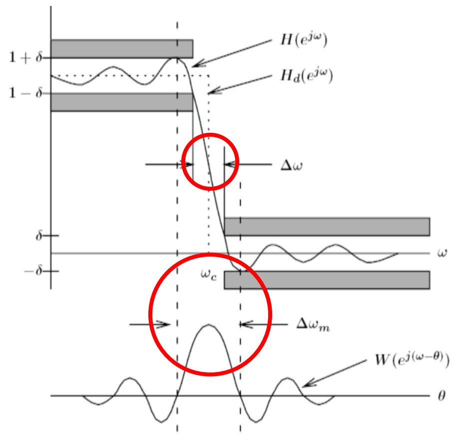
Windowed FIR Property 3: Mainlobe Width



Punskaya, Slide 99

- The distance between approximation error peaks is approximately equal to the width of the mainlobe Δw_m

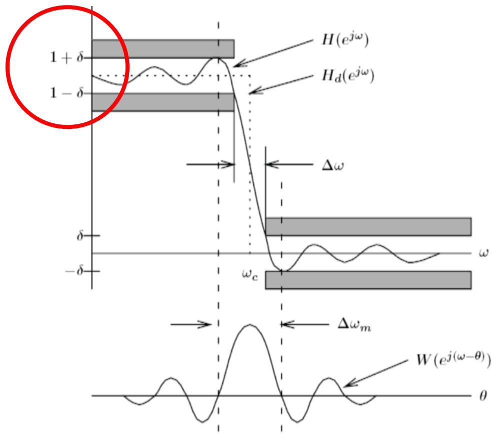
Windowed FIR Property 4: Mainlobe Width [2]



Punskaya, Slide 96

- The width of the mainlobe is wider than the transition bandwidth

Windowed FIR Property 5:
Peak Δδ is determined by the window shape



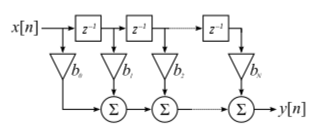
Punskaya, Slide 96

- peak approximation error is determined by the window shape, <u>independent</u> of the filter order

FIR Filter Design

- How to get all these coefficients?

$$H(\omega) = \boxed{h_0 + h_1 e^{-\mathrm{i}\omega} + \cdots + h_{n-1} e^{-\mathrm{i}(n-1)\omega}}$$

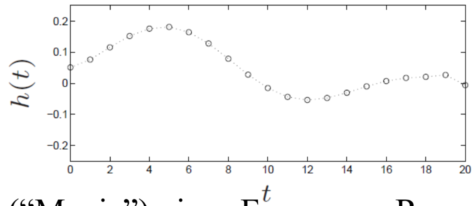


FIR Design Methods:

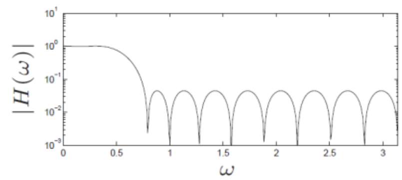
1. Impulse Response Truncation
   - \+ Simplest
   - – Undesirable frequency domain-characteristics, not very useful
2. Windowing Design Method
   - \+ Simple
   - – Not optimal (not minimum order for a given performance level)
3. Optimal filter design methods
   - \+ "More optimal"
   - – Less simple…

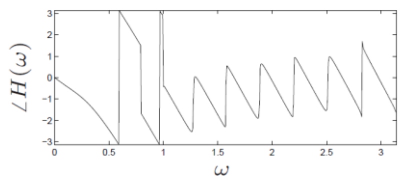
FIR Filter Design & Operation
Ex: Lowpass FIR filter

- Set Impulse response (order n = 21)
- "Determine" h(t)
  - h(t) is a 20 element vector that we'll use to as a weighted sum



- FFT ("Magic") gives Frequency Response & Phase





Why is this "hard"? Looking at the Low-Pass Example

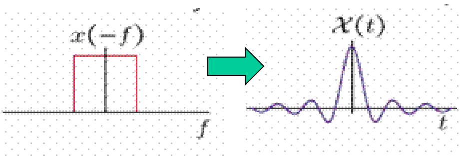
$$H_d(\omega) = \begin{cases} 1 \text{ if } |\omega| \leq \omega_c \\ 0 \text{ if } \omega_c < |\omega| < \pi \end{cases}$$

- Why is this hard?
  - Shouldn't it be "easy" ??
    … just hit it with some FFT "magic" and then keep the bands we want and then hit it with some Inverse-FFT "supermagic"???
  - Remember we need a "system" that does this "rectangle function" in frequency
  - Let's consider what that means…
    - It basically suggests we need an **Inverse FFT** of a **"rectangle function"**

Flashback: Fourier Series & Rectangular Functions

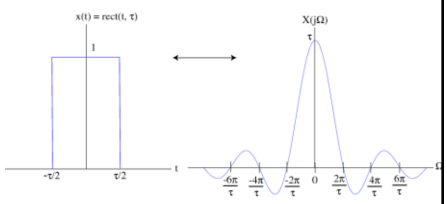
$\mathfrak{F}$: Fourier Tranform

$$\mathfrak{F}^{-1}\left\{rect\left(\frac{\omega}{2}\right)\right\} = \frac{sinc\,(t)}{\pi}$$



Ref: http://cnx.org/content/m26719/1.1/
http://www.wolframalpha.com/input/?i=IFFT%28sinc%28f%29%29

$$\mathfrak{F}\{rect\,(t)\} = sinc\left(\frac{\omega}{2}\right)$$



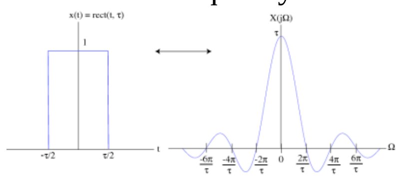
Ref: http://cnx.org/content/m32899/1.8/
http://www.thefouriertransform.com/pairs/box.php

See:

- Table 7.1 (p. 702) Entry 17 & Table 9.1 (p. 852) Entry 7

Flashback: Fourier Series & Rectangular Functions [2]

- The sinc function might look familiar
  - This is the frequency content of a square wave (box)



Ref: http://www.wolframalpha.com/input/?i=FFT%28rect%28t%29%29
http://cnx.org/content/m32899/1.8/

- This also applies to **signal reconstruction!**

➔ Whittaker–Shannon interpolation formula
  - This says that the "better way" to go from Discrete to Continuous (i.e. D to A) is not ZOH, but rather via the sinc!

$$x(t) = \sum_{n=-\infty}^{\infty} x[n] \cdot \mathrm{sinc}\left(\frac{t-nT}{T}\right)$$

∴ FIR and Low Pass Filters…

∴

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

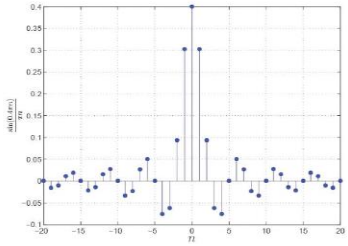
Has impulse response:

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

Thus, to filter an impulse train with an ideal low-pass filter use:

$$x(t) = \left(\sum_{n=-\infty}^{\infty} x[n] \cdot \delta(t - nT)\right) * \text{sinc}\left(\frac{t}{T}\right)$$

- **<u>However</u>**!!
  a sinc is non-causal and infinite in duration



And, this **cannot** be implemented **in practice** ☹

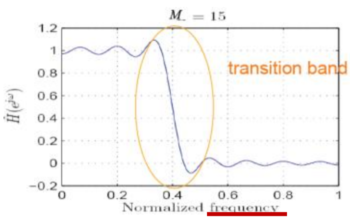
∵ we need to know all samples of the input, both in the **past** and in the **future**

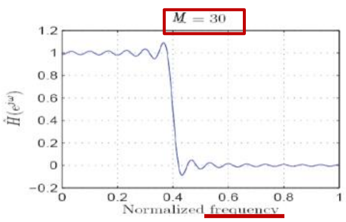
Plan 0: Impulse Response Truncation

Maybe we saw this coming…

∴ Clip off the sinc at some large *n*

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





- *Ripples* in both passband/stopband
  and the transition not abrupt (i.e., a *transition band*).
- As M→∞, transition band→ 0 (as expected!)

➔ FIR Filters: Window Function Design Method

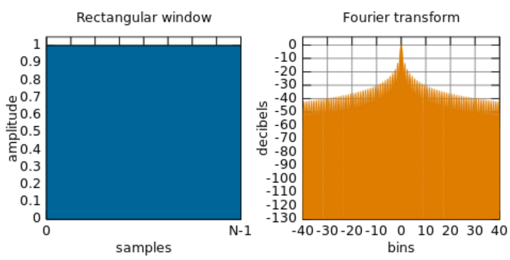
- Windowing: a generalization of the truncation idea

- There many, many “window” functions:
  - Rectangular
  - Triangular
  - Hanning
  - Hamming
  - Blackman
  - Kaiser
  - Lanczos
  - Many More … (see: http://en.wikipedia.org/wiki/Window_function)

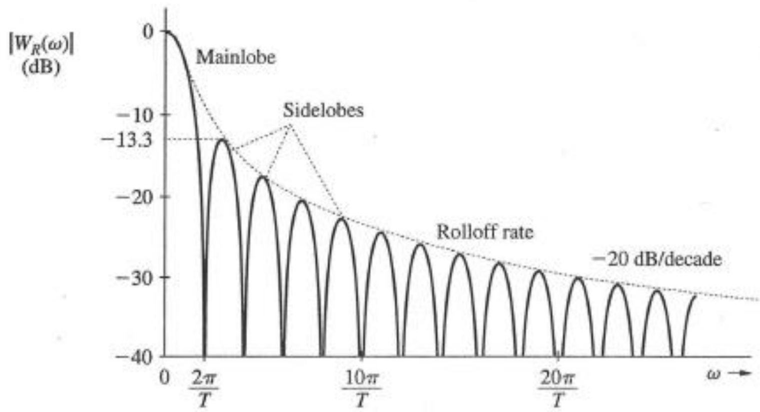
Some Window Functions [1]

1. Rectangular

$$w(n) = 1$$



Windowing and its effects/terminology

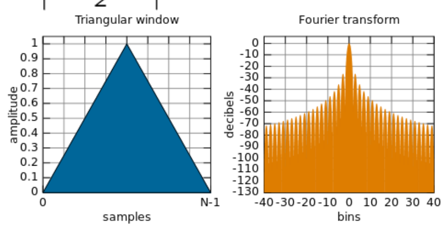


Lathi, Fig. 7.45

Some More Window Functions …

2. Triangular window

$$w(n) = 1 - \left| \frac{n - \frac{N-1}{2}}{\frac{N+1}{2}} \right|$$



- And Bartlett Windows
  - A slightly narrower variant with zero weight at both ends:

$$w(n) = 1 - \left| \frac{n - \frac{N-1}{2}}{\frac{N-1}{2}} \right|$$

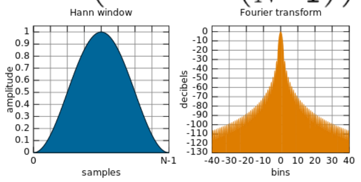
Some More Window Functions…

3. Generalized Hamming Windows

$$w(n) = \alpha - \beta \cos\left(\frac{2\pi n}{N-1}\right)$$

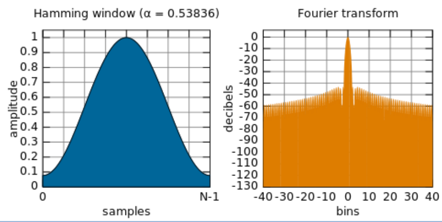
→Hanning Window

$$\rightarrow w(n) = 0.5\left(1 - \cos\left(\frac{2\pi n}{N-1}\right)\right)$$



→Hamming's Window

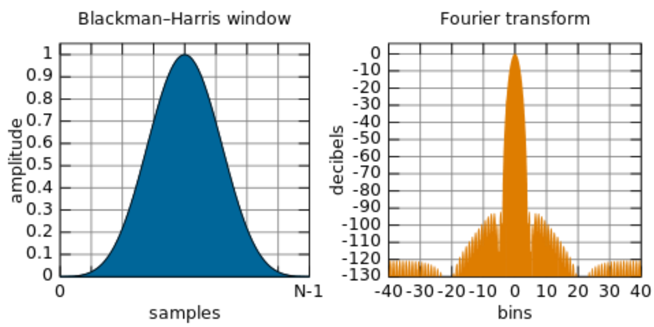
$$\rightarrow \alpha = 0.54,\ \beta = 1 - \alpha = 0.46$$



Some More Window Functions…

4. Blackman–Harris Windows
   - A generalization of the Hamming family,
   - Adds more shifted sinc functions for less side-lobe levels

$$w(n) = a_0 - a_1 \cos\left(\frac{2\pi n}{N-1}\right) + a_2 \cos\left(\frac{4\pi n}{N-1}\right) - a_3 \cos\left(\frac{6\pi n}{N-1}\right)$$

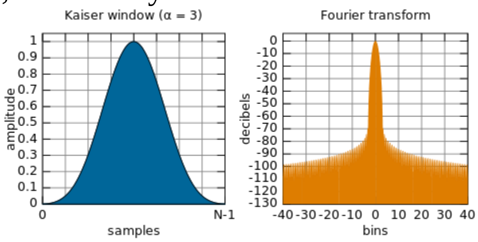


Some More Window Functions…

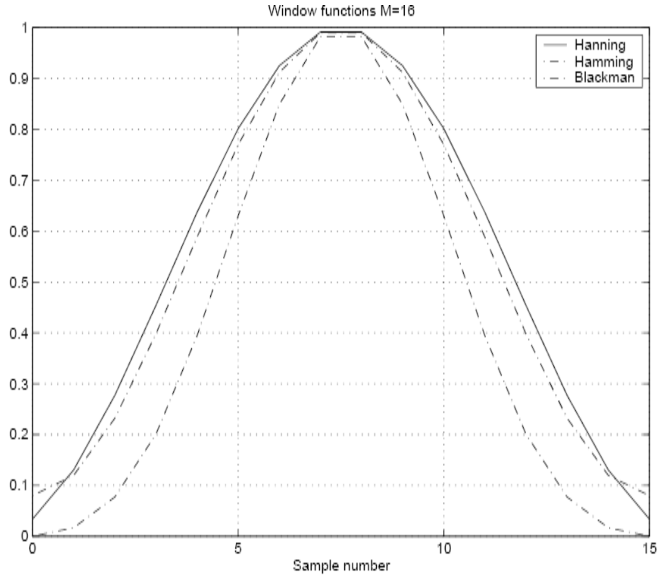
5. Kaiser window
   – A DPSS (discrete prolate spheroidal sequence)
   – Maximize the energy concentration in the main lobe

$$\rightarrow w(n) = \frac{I_0\left(\pi\alpha\sqrt{1-\left(\frac{2n}{N-1}-1\right)^2}\right)}{I_0(\pi\alpha)}$$

   – Where: I_0 is the zero-th order modified Bessel function of the first kind, and usually $\alpha = 3$.

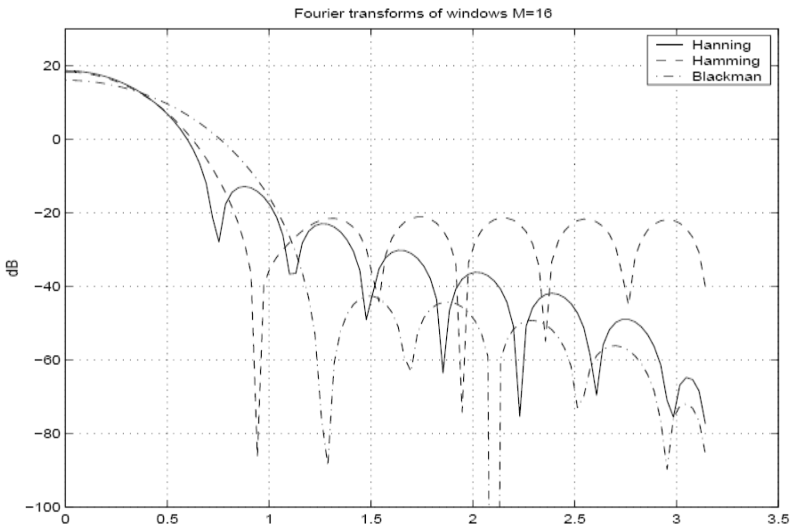


Comparison of Alternative Windows –Time Domain



Punskaya, Slide 90

Comparison of Alternative Windows
Frequency Domain



Punskaya, Slide 91

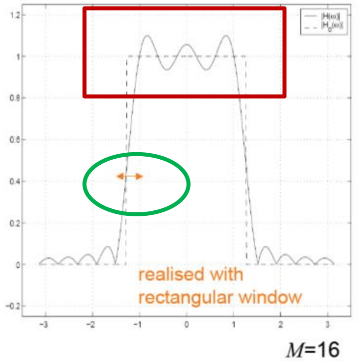
Summary Characteristics of Common Window Functions

No.	Window $w(t)$	Mainlobe Width	Rolloff Rate (dB/oct)	Peak Sidelobe level (dB)	Peak $20\log_{10}\delta$
1	Rectangular: $\mathrm{rect}\left(\frac{t}{T}\right)$	$\frac{4\pi}{T}$	-6	-13.3	-21dB
2	Bartlett: $\Delta\left(\frac{t}{2T}\right)$	$\frac{8\pi}{T}$	-12	-26.5	
3	Hanning: $0.5\left[1+\cos\left(\frac{2\pi t}{T}\right)\right]$	$\frac{8\pi}{T}$	-18	-31.5	-44dB
4	Hamming: $0.54+0.46\cos\left(\frac{2\pi t}{T}\right)$	$\frac{8\pi}{T}$	-6	-42.7	-53dB
5	Blackman: $0.42+0.5\cos\left(\frac{2\pi t}{T}\right)+0.08\cos\left(\frac{4\pi t}{T}\right)$	$\frac{12\pi}{T}$	-18	-58.1	-74dB
6	Kaiser: $\dfrac{I_0\left[\alpha\sqrt{1-4\left(\frac{t}{T}\right)^2}\right]}{I_0(\alpha)}$ $0 \le \alpha \le 10$	$\frac{11.2\pi}{T}$	-6	-59.9 $(\alpha = 8.168)$	

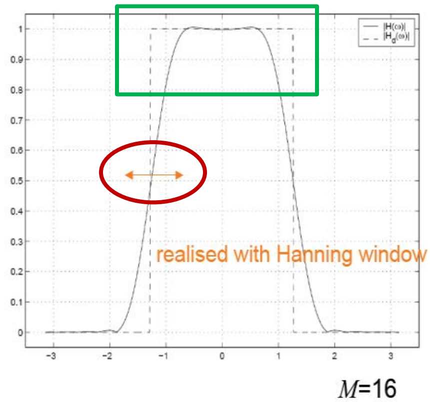
Lathi, Table 7.3
Punskaya, Slide 92

FIR: Rectangular & Hanning Windows

- Rectangular



- Hanning

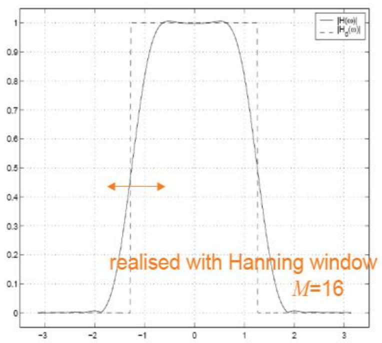


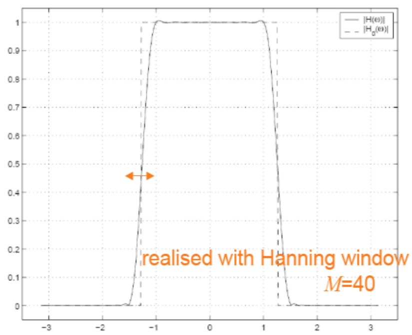
➔ Hanning: Less ripples, but wider transition band

Punskaya, Slide 93

Adding Order

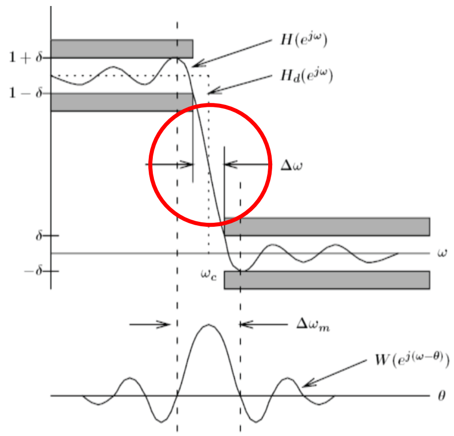
\+ Transition and Smoothness
– Increased Size





Punskaya, Slide 94

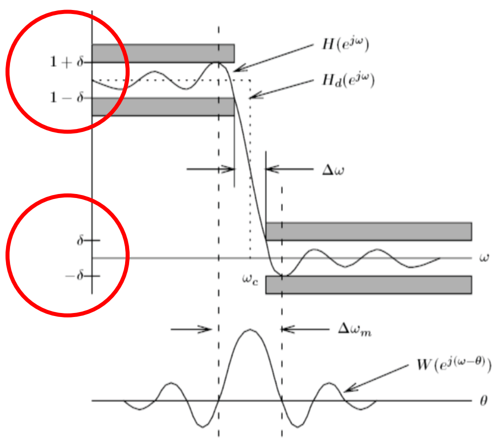
Windowed FIR Property 1:
Equal transition bandwidth



Punskaya, Slide 96

- Equal transition bandwidth on both sides of the ideal cutoff frequency

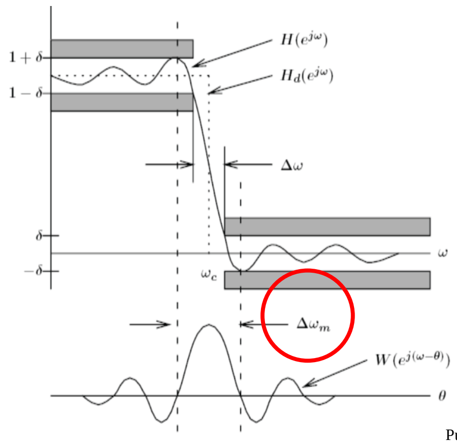
Windowed FIR Property 2:
Peak Errors same in Passband & Stopband



Punskaya, Slide 96

- Peak approximation error in the passband ($1+\delta \rightarrow 1-\delta$) is equal to that in the stopband ($\delta \rightarrow -\delta$)

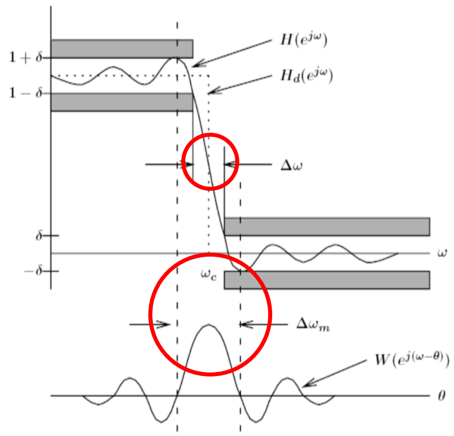
Windowed FIR Property 3: Mainlobe Width



Punskaya, Slide 99

- The distance between approximation error peaks is approximately equal to the width of the mainlobe Δw_m

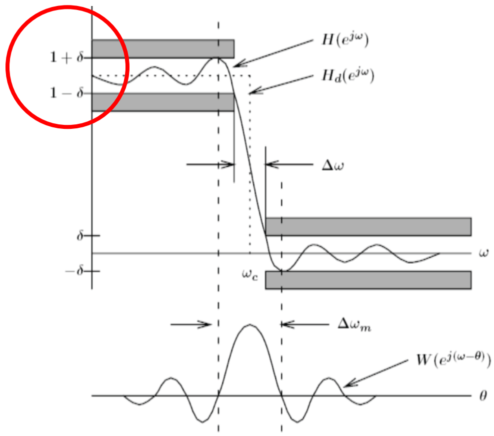
Windowed FIR Property 4: Mainlobe Width [2]



Punskaya, Slide 96

- The width of the mainlobe is wider than the transition bandwidth

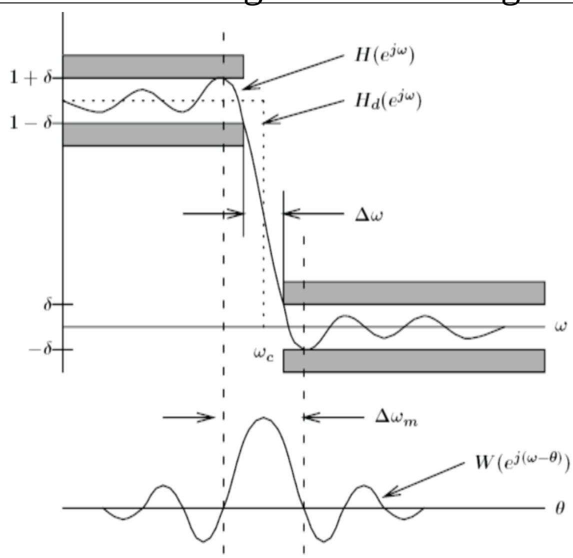
Windowed FIR Property 5: Peak Δδ is determined by the window shape



Punskaya, Slide 96

- peak approximation error is determined by the window shape, <u>independent</u> of the filter order

Window Design Method Design Terminology



Punskaya, Slide 96

Where:

- ω_c: cutoff frequency
- δ: maximum passband ripple
- Δω: transition bandwidth
- $\Delta\omega_m$: width of the window mainlobe

Passband / stopband ripples

ω_s and ω_p: Corner Frequencies

Passband / stopband ripples are often expressed in dB:

- passband ripple = 20 log_{10} $(1+\delta_p)$ dB
- peak-to-peak passband ripple ≅ 20 log_{10} $(1+2\delta_p)$ dB
- minimum stopband attenuation = -20 log_{10} (δ_s) dB

Passband / stopband ripples

ω_s and ω_p: Corner Frequencies

Passband / stopband ripples are often expressed in dB:

- passband ripple = ~~20 log_{10} $(1+\delta_p)$ dB~~ = 20 log_{10} (δ_p) dB
- peak-to-peak passband ripple ≅ ~~20 log_{10} $(1+2\delta_p)$ dB~~
  ≅ 20 log_{10} $(2\delta_p)$ dB
- minimum stopband attenuation = ~~-20 log_{10} (δ_s) dB~~
  =20 log_{10} (δ_s) dB

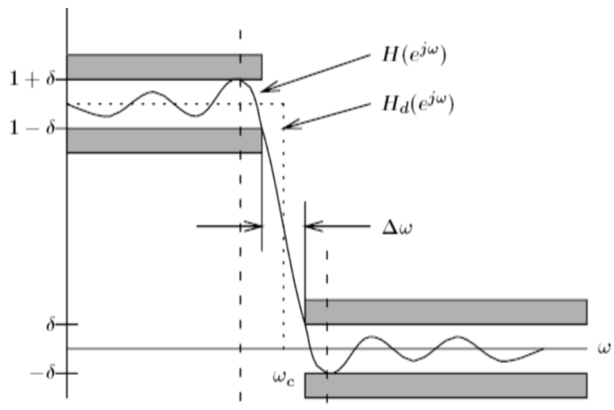
Summary of Design Procedure

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 (e.g. by increasing M if the specified constraints have not been satisfied).

Punskaya, Slide 105

Windowed Filter Design Example

- Design a type I low-pass filter with:
  - ω_p **=0.2π**
  - ω_s **=0.3π**
  - δ **=0.01**



Windowed Filter Design Example: Step 1: Select a suitable Window Function

No.	Window $w(t)$	Mainlobe Width	Rolloff Rate (dB/oct)	Peak Sidelobe level (dB)	Peak $20\log_{10}\delta$
1	Rectangular: $\mathrm{rect}\left(\frac{t}{T}\right)$	$\frac{4\pi}{T}$	-6	-13.3	-21dB
2	Bartlett: $\Delta\left(\frac{t}{2T}\right)$	$\frac{8\pi}{T}$	-12	-26.5	
3	Hanning: $0.5\left[1+\cos\left(\frac{2\pi t}{T}\right)\right]$	$\frac{8\pi}{T}$	-18	-31.5	-44dB
4	Hamming: $0.54+0.46\cos\left(\frac{2\pi t}{T}\right)$	$\frac{8\pi}{T}$	-6	-42.7	-53dB
5	Blackman: $0.42+0.5\cos\left(\frac{2\pi t}{T}\right)+0.08\cos\left(\frac{4\pi t}{T}\right)$	$\frac{12\pi}{T}$	-18	-58.1	-74dB
6	Kaiser: $\frac{I_0\left[\alpha\sqrt{1-4\left(\frac{t}{T}\right)^2}\right]}{I_0(\alpha)}$ $0 \le \alpha \le 10$	$\frac{11.2\pi}{T}$	-6	-59.9 ($\alpha = 8.168$)	

- LP with: ω_p **=0.2π**, ω_s **=0.3π**, δ **=0.01**
- δ **=0.01:** The required peak error spec: -20log10 (δ) = –40 dB

Hanning Window

- Main-lobe width:

$\omega_s - \omega_p = 0.3\pi - 0.2\pi = 0.1\pi \rightarrow 0.1\pi = 8\pi / M$

→ Filter length M ≥ 80 & Filter order N ≥ 79

- BUT, Type-I filters have even order so N = 80

Windowed Filter Design Example: Step 2: Specify the Ideal Response

- From Property 1 (Midpoint rule)

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

∴ An ideal response will be:

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

Windowed Filter Design Example: 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 $H_d(\omega)$

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

\+ Delayed impulse response (to make it causal)

$$\tilde{h}(n) = \hat{h}\left(n - \frac{N-1}{2}\right)$$

- Coefficients of the ideal filter (via equation or IFFT):

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

Windowed Filter Design Example: Step 4: Multiply to obtain the filter coefficients

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

- Multiply by a Hamming window function for the passband:

$$w(n) = 0.54 - 0.46\ \cos\left(\frac{2\pi n}{M}\right)$$

Windowed Filter Design Example: 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, **then**:
  - Adjust the ideal filter frequency response (for example, move the band edge) and repeat (step 2)
  - Adjust the filter length and repeat (step 4)
  - change the window (& filter length) (step 4)

- And/Or consult with Matlab:
  - `FIR1` and `FIR2`
  - `B=FIR2(N,F,M):` Designs a Nth order FIR digital filter with

Windowed Filter Design Example: Consulting Matlab:

- `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
  - Frequencies `F` must be in increasing order between 0 and Fs/2, with Fs corresponding to 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

In Conclusion

- FIR Filters are digital (can not be implemented in analog) and exploit the difference and delay operators

- A window based design builds on the notion of a truncation of the “ideal” box-car or rectangular low-pass filter in the Frequency domain (which is a sinc function in the time domain)

- Other Design Methods exist:
  - Least-Square Design
  - Equiripple Design
  - Remez method
  - The Parks-McClellan Remez algorithm
  - Optimisation routines …