ECE 6560
Multirate Signal Processing
Chapter 3

Dr. Bradley J. Bazuin
Western Michigan University
College of Engineering and Applied Sciences
Department of Electrical and Computer Engineering
1903 W. Michigan Ave.
Kalamazoo MI, 49008-5329
# Chapter 3: Digital Filters

### 3.1 Filter Specifications

- Freq. domain rect function prototype filter
- Coherent gain

### 3.2 Windowing

- FFT windowing with “filter functions”

### 3.3 The Remez Algorithm

- Optimal digital FIR filter generation

<table>
<thead>
<tr>
<th>Subsection</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.3.1 Equiripple Vs, 1/f Ripple Designs</td>
<td>60</td>
</tr>
<tr>
<td>3.3.2 Acceptable In-band Ripple Levels</td>
<td>66</td>
</tr>
</tbody>
</table>
Filtering Needs

- Multirate processing involved: filter-decimation and interpolation-filter operations

- The definition of filters appropriate for the signals being processed is necessary for the signal processing being performed.
  - The desired or required passband bandwidths for the signals of interest must be known.
  - The filter structure is also an important consideration.

Filter passbands, transition bands, stopbands, and other shape characteristics must be defined and understood!
Filter Notes

• The “Filter Notes” slides provide standard definitions, MATLAB programming for classic “analog” filters, MATLAB programming for classic analog filters as digital IIR filters, and a examples of MATLAB generating the filters.

• The “FIR Filter DSP Notes” provide insight into digital filters, properties of the four types of FIR filters, translated and complimentary filters, and a brief IIR filter discussion.
  – Things to know: Summary pages, Type 1 (odd # coef., real symmetric) and type 2 FIR (even # coef., real symmetric)
## FIR Filter Types

<table>
<thead>
<tr>
<th>Filter Type</th>
<th># Coef., Symmetry</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type 1 Filter:</td>
<td>Odd, Symmetric</td>
<td>It can do all filters. (the best option)</td>
</tr>
<tr>
<td>Type 2 Filter:</td>
<td>Even, Symmetric</td>
<td>It can not be a highpass filter.</td>
</tr>
<tr>
<td>Type 3 Filter:</td>
<td>Odd, Anti-Symmetric</td>
<td>It can only be a bandpass filter.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(worthless?!)</td>
</tr>
<tr>
<td>Type 4 Filter:</td>
<td>Even, Anti-Symmetric</td>
<td>It can not be a lowpass filter.</td>
</tr>
</tbody>
</table>

**Note:** Type 2 and type 4 are often related by \((-1)^n\)
(changing symmetric to anti-symmetric)
Rectangular Low Pass Filter

\[ H(f) = \text{rect} \left( \frac{f}{2 \cdot f_1} \right) \]

\[ h(t) = 2 \cdot f_1 \cdot \frac{\sin(2\pi \cdot f_1 \cdot t)}{2\pi \cdot f_1 \cdot t} \]

\[ h(t) = 2 \cdot f_1 \cdot \text{sinc}(2 \cdot f_1 \cdot t) \]

\[ h(t) = 2 \cdot f_1 \cdot \frac{\sin \left(2\pi \cdot \left( \frac{2 \cdot f_1}{2} \right) \cdot t \right)}{2\pi \left( \frac{2 \cdot f_1}{2} \right) \cdot t} \]

\[ h(t) = 2 \cdot f_1 \cdot \frac{\sin \left( \frac{2 \cdot f_1}{2} \cdot t \right)}{\pi} \]

**First zero:** \[ 2\pi \left( \frac{2 \cdot f_1}{2} \right) \cdot t_{1st\text{-}zero} = \pi \]

\[ t_{1st\text{-}zero} = \frac{1}{2 \cdot f_1} \]

**Figure 3.1** Frequency Response Prototype Low-pass Filter
Preserving Filter Gain in Sampling

- We want to normalize the integral of the continuous time impulse response (unity gain DC consideration).

\[ \int_{t=-\infty}^{\infty} h(t) \cdot dt = H(0) = K \cdot \sum_{n=-\infty}^{\infty} h(n \cdot T) \]

- Based on Poisson’s sum formula (not proven here)

\[ K = T = \frac{1}{f_s} \]

- Therefore the filter coefficients should be

\[ h(n) = 2 \cdot \frac{f_1}{f_s} \cdot \text{sinc}(2 \cdot f_1 \cdot t) \]
Sampled Impulse Response

\[ h(n) = \frac{1}{f_S} \left. h(t) \right|_{t=n\frac{1}{f_S}} \]

\[ h(n) = \frac{2f_1}{f_S} \frac{\sin(2\pi f_1 n)}{(2\pi f_1 n)} \]

\[ = \frac{2f_1}{f_S} \frac{\sin(n\theta_1)}{(n\theta_1)} \text{, where } \theta_1 = 2\pi \frac{f_1}{f_S} \]

\[ h(t) \cdot \frac{1}{f_S} \delta(t - \frac{n}{f_S}) = 2 \cdot \frac{f_1}{f_S} \cdot \frac{\sin(2\pi f_1 t)}{2\pi f_1 t} \cdot \delta(t - \frac{n}{f_S}) \]

\[ h\left( \frac{n}{f_s} \right) = 2 \cdot \frac{f_1}{f_S} \cdot \frac{\sin \left( \frac{2\pi f_1 n}{f_s} \right)}{2\pi \cdot \frac{f_1}{f_s} \cdot \frac{n}{f_s}} \text{ for } 2\pi \cdot \frac{f_1}{f_s} = \theta_1 \]

\[ h\left( \frac{n}{f_s} \right) = h_s(n) = \frac{\theta_1}{\pi} \cdot \sin \left( \frac{n \cdot \theta_1}{n \cdot \theta_1} \right) \]
Sampled Response

Figure 3.3  Sampled Data Impulse Response of Prototype Filter

How many discrete sample are there between n=0 and the fist null?

\[ 2\pi \cdot \frac{f_1}{f_s} \cdot n = \theta_1 \cdot n = \pi \]

\[ n = \frac{1}{2} \cdot \frac{f_s}{f_1} \]

Note that a first order estimate of a low pass filter cutoff frequency can be made by counting the sinc function discrete points between 0 and the first null.

\[ f_1 = \frac{1}{2} \cdot \frac{f_s}{n} \]
**Sampled Response**

The number of samples is seen to be $f_s/(2f_1)$, the ratio of sample rate $f_s$ to the **two-sided bandwidth** $f_1$, which for our specific example is 5.

- Therefore the two-sided bandwidth is $1/5^{th}$ the sample rate!
- The single sided bandwidth is $1/10^{th}$ the sample rate!

The relative bandwidth can be estimated by counting ….

---

**Figure 3.3** Sampled Data Impulse Response of Prototype Filter

The number of samples is seen to be $f_s/(2f_1)$, the ratio of sample rate $f_s$ to the **two-sided bandwidth** $f_1$, which for our specific example is 5.

- Therefore the two-sided bandwidth is $1/5^{th}$ the sample rate!
- The single sided bandwidth is $1/10^{th}$ the sample rate!

The relative bandwidth can be estimated by counting ….
Spectrum

The sampling process, of course, causes the spectra to be periodically extended with spectral replicates at all multiples of the sample rate.

The expression for the spectrum of the sampled data impulse response is shown in where the sampled data frequency variable $\omega T_s$ is denoted by $\theta$ with units of radians/sample. In this coordinate system, the spectrum is periodic in $2\pi$.

$$H(\theta) = \sum_{n=-\infty}^{+\infty} h(n)e^{-jn\theta}$$

$$H(\theta) = \sum_{n=-\infty}^{\infty} h(n)\cdot\exp(-j\cdot n\cdot \theta)$$
Continuous vs. Discrete

• The previous analysis looked at the continuous-time continuous-frequency Fourier Transform and the discrete-time continuous-frequency Fourier Transform.

• What about the inverse transform of the discrete-time discrete-frequency Fourier Transform?
Ideal Finite Spectrum Discrete Filter

\[ h(n) = \frac{1}{N} \cdot \sum_{k=-N/2}^{N/2-1} \text{rect} \left( \frac{k}{2 \cdot k_1} \right) \cdot W^{-n \cdot k} \quad k_1 < \frac{N}{2} \]

\[ h(n) = \frac{1}{N} \cdot \sum_{k=-k_1}^{k_1} \exp \left( j \cdot 2\pi \cdot \frac{n \cdot k}{N} \right) \]

\[ k' = k + k_1 \]

\[ h(n) = \frac{1}{N} \cdot \exp \left( j \cdot 2\pi \cdot \frac{-n \cdot k_1}{N} \right) \cdot \sum_{k=0}^{2k_1} \exp \left( j \cdot 2\pi \cdot \frac{n \cdot k}{N} \right) \rightarrow \text{Infinite sum math trick} \]

\[ h(n) = \frac{1}{N} \cdot \exp \left( j \cdot 2\pi \cdot \frac{-n \cdot k_1}{N} \right) \cdot \left[ 1 - \exp \left( j \cdot 2\pi \cdot \frac{n \cdot (2 \cdot k_1 + 1)}{N} \right) \right] \cdot \left[ \frac{1}{1 - \exp \left( j \cdot 2\pi \cdot \frac{n}{N} \right)} \right] \]

\[ h(n) = \frac{1}{N} \cdot \frac{2 \cdot j \cdot \sin \left( \frac{2\pi \cdot n \cdot (k_1 + 0.5)}{N} \right)}{2 \cdot j \cdot \sin \left( \frac{2\pi \cdot n \cdot (0.5)}{N} \right)} = \frac{1}{N} \cdot \frac{\sin \left( \frac{2\pi \cdot n \cdot (2 \cdot k_1 + 1)}{2 \cdot N} \right)}{\sin \left( \frac{2\pi \cdot n}{2 \cdot N} \right)} \]
Sample Replicated Sine Functions

\[ h(n) = \frac{1}{N} \cdot \left[ \frac{2 \cdot j \cdot \sin\left(\frac{2\pi \cdot n \cdot (k + 0.5)}{N}\right)}{2 \cdot j \cdot \sin\left(\frac{2\pi \cdot n \cdot (0.5)}{N}\right)} \right] = \frac{1}{N} \cdot \left[ \frac{\sin\left(\frac{2\pi \cdot n \cdot (2 \cdot k + 1)}{2 \cdot N}\right)}{\sin\left(\frac{2\pi \cdot n}{2 \cdot N}\right)} \right] \]

for \( \frac{2\pi}{N} = \theta \)

\[ h(n) = \frac{1}{N} \cdot \left[ \frac{\sin\left(n \cdot (2 \cdot k + 1) \cdot \left(\theta_2\right)\right)}{\sin\left(n \cdot \theta_2\right)} \right] \]

For those who prefer sinc functions

\[ h(n) = \frac{1}{N} \cdot \left[ \frac{\text{sinc}\left(\frac{2 \cdot n \cdot (2 \cdot k + 1)}{2 \cdot N}\right)}{\text{sinc}\left(\frac{2 \cdot n}{2 \cdot N}\right)} \right] \cdot \frac{2 \cdot n \cdot (2 \cdot k + 1)}{2 \cdot N} \cdot \frac{2 \cdot N}{2 \cdot n} = \frac{(2 \cdot k + 1)}{N} \cdot \left[ \frac{\text{sinc}\left(\frac{2 \cdot k + 1}{N}\right)}{\text{sinc}\left(\frac{n}{N}\right)} \right] \]
Comparing results

DT-CF Fourier Transform

\[
h\left(\frac{n}{f_s}\right) = 2 \cdot \frac{f_i}{f_s} \cdot \frac{\sin\left(2\pi \cdot \frac{f_i \cdot n}{f_s}\right)}{2\pi \cdot \frac{f_i \cdot n}{f_s}} = 2 \cdot \frac{f_i}{f_s} \cdot \text{sinc}\left(2 \cdot \frac{f_i \cdot n}{f_s}\right)
\]

\[
\frac{f_s}{\Delta f} \Rightarrow N \quad f_i \Rightarrow \left(k_1 + \frac{1}{2}\right) \cdot \Delta f
\]

\[
h(n) = \frac{(2 \cdot k_1 + 1)}{N} \cdot \text{sinc}\left(\frac{(2 \cdot k_1 + 1) \cdot n}{N}\right)
\]

DT-DF Fourier Transform

\[
h(n) = \frac{(2 \cdot k_1 + 1)}{N} \cdot \left[\frac{\text{sinc}\left(\frac{(2 \cdot k_1 + 1) \cdot n}{N}\right)}{\text{sinc}\left(\frac{n}{N}\right)}\right]
\]

The numerator is the same, while the denominator is due to discretizing the frequency space.
Generating Digital Filters

For filters with minimal passband ripple and narrow transition bands:

1. Generate an analog filter and then transform it into a digital filter. (Not used in this textbook/class)

2. Use a windowed sinc function. (notes follow)
   - Assumes an ideal frequency band filter is convolved by a frequency domain window function.
   - In the time domain, the infinite sinc samples are truncated in length and shaped by a time window.

3. Use a digital filter generating algorithm
   - Parks-McClellan is the most popular (remez Algorithm)
   - Other iterative and non-iterative digital filter algorithms exist.
Windows vs. Filters (1)

• Windows and filters are sets of coefficients that modify the spectrum of the signal being processed.

• A filter is applied to a signal in time (continuous or sampled) to modify the spectral content.

• A window is applied to an entity with a defined number of samples (a sample set) to enhance spectral components.
  – Before applying an FFT, the data is windowed to shape the spectral bins
  – The infinite number of discrete filter coefficients had a window applied to create a finite impulse response (e.g. a rectangular windowed)

Fourier Pair Concerns

Finite Time ⇔ Infinite Frequency
Infinite Time ⇔ Finite Frequency
Windows vs. Filters (2)

- **Filter Application**
  - Convolution in the time domain
  - Multiplication in the frequency domain

\[
y(n) = x(n) * h(n) = \sum_{k=-\infty}^{\infty} h(k) \cdot x(n-k) = \sum_{k=-\infty}^{\infty} x(k) \cdot h(n-k)
\]

\[
Y(f) = X(f) \cdot H(f)
\]

- **Window Application**
  - Multiplication in the time domain
  - Convolution in the frequency domain

\[
y(n) = x(n) \cdot h(n)
\]

\[
Y(f) = X(f) * H(f)
\]
Windowing a Filter
(Convolution in the frequency domain)

**DTCF Derivation**

\[ h_w(n) = w(n) \cdot h(n) \]

\[ H_w(\theta) = \sum_{n=-\infty}^{+\infty} h(n)w(n)e^{-jn\theta} = \sum_{n=-N/2}^{+N/2} h(n)w(n)e^{-jn\theta} \]

\[ H_w(\theta) = \frac{1}{2\pi} \int_{-\pi}^{+\pi} H(\phi)W(\phi - \theta)\,d\phi \]

**DTDF Derivation**

\[ H_w(k) = \sum_{n=0}^{N-1} w(n)\cdot h(n)\cdot W_N^{-nk} \]

\[ H_w(k) = \frac{1}{N} \cdot \sum_{l=0}^{N} \left[ W(l)\cdot W_N^{-nl} \right] \cdot h(n)\cdot W_N^{nk} \]

\[ H_w(k) = \sum_{l=0}^{N-1} \frac{1}{N} \cdot W(l)\cdot \sum_{n=0}^{N-1} W_N^{-nl} \cdot h(n)\cdot W_N^{nk} \]

\[ H_w(k) = \sum_{l=0}^{N-1} \frac{1}{N} \cdot W(l)\cdot \sum_{n=0}^{N-1} h(n)\cdot W_N^{-n(k-l)} \]

\[ H_w(k) = \frac{1}{N} \cdot \sum_{l=0}^{N-1} W(l)\cdot \langle H(k-l) \rangle_N \]

“Circular convolution”

Symmetric Rectangular Window
(Non-Causal Spectral Definition)

Our first candidate for a window is the symmetric rectangle, sometimes called the default window (odd length, $M_{even} + 1$).

$$W(k) = \sum_{n=-N/2}^{N/2-1} \text{rect}\left(\frac{n}{2M}\right) \cdot W_n^{n,k}$$

$$W(k) = \sum_{n=-M/2}^{M/2} \exp\left(-j \cdot 2\pi \cdot \frac{n \cdot k}{N}\right) = \exp\left(j \cdot 2\pi \cdot \frac{M/2 \cdot k}{N}\right) \cdot \sum_{n=0}^{M} \exp\left(-j \cdot 2\pi \cdot \frac{n \cdot k}{N}\right)$$

$$W(k) = \exp\left(j \cdot 2\pi \cdot \frac{M/2 \cdot k}{N}\right) \cdot \left[1 - \exp\left(-j \cdot 2\pi \cdot \frac{(M+1) \cdot k}{N}\right)\right] \cdot \frac{1}{1 - \exp\left(-j \cdot 2\pi \cdot \frac{k}{N}\right)}$$

For $\frac{2\pi \cdot k}{N} = \theta_k$

$$W(k) = \frac{\sin\left(2\pi \cdot \frac{(M/2 + 1/2) \cdot k}{N}\right)}{\sin\left(2\pi \cdot \frac{1/2 \cdot k}{N}\right)} = \frac{\sin\left(2\pi \cdot \frac{(M+1) \cdot k}{2N}\right)}{\sin\left(2\pi \cdot \frac{k}{2N}\right)} = \frac{\sin\left(M + 1 \cdot \frac{1}{2} \cdot \theta_k\right)}{\sin\left(\frac{1}{2} \cdot \theta_k\right)}$$
Rectangular Window
(Causal Discrete Time Definition)

Our next contender for a window is another rectangle, also thought of as a default window (even length window, $M_{\text{even}}$).

$$W(k) = \sum_{n=0}^{N-1} w(n) \cdot W_n^{n,k}, \text{ where } w(n) = 1 \text{ for } n = 0 \text{ to } M - 1$$

$$W(k) = \sum_{n=0}^{M-1} W_n^{n,k} = \left(1 - \exp\left(-j \cdot 2\pi \cdot \frac{M \cdot k}{N}\right)\right)$$

$$W(k) = \exp\left(-j \cdot 2\pi \cdot \frac{(M-1) \cdot k}{2N}\right) \cdot \left(\frac{\sin\left(2\pi \cdot \frac{M \cdot k}{2N}\right)}{\sin\left(\frac{2\pi \cdot k}{2N}\right)}\right)$$

for $\frac{2\pi \cdot k}{N} = \theta_k$  

$$W(\theta_k) = \exp\left(-j \cdot (M-1) \cdot \frac{\theta_k}{2}\right) \cdot \left(\frac{\sin\left(M \cdot \frac{\theta_k}{2}\right)}{\sin\left(\frac{\theta_k}{2}\right)}\right)$$
Odd vs. Even Rectangular Window

M Odd (Non-causal)

\[
W(k) = \frac{\sin\left(2\pi \cdot \frac{(M+1) \cdot k}{2 \cdot N}\right)}{\sin\left(2\pi \cdot \frac{k}{2 \cdot N}\right)}
\]

\[
W(\theta_k) = \frac{\sin\left(\frac{M+1}{2} \cdot \theta_k\right)}{\sin\left(\frac{1}{2} \cdot \theta_k\right)}
\]

M Even (Causal)

\[
W(k) = \exp\left(-j \cdot 2\pi \cdot \frac{(M-1) \cdot k}{2N}\right) \cdot \frac{\sin\left(2\pi \cdot \frac{M \cdot k}{2N}\right)}{\sin\left(2\pi \cdot \frac{k}{2N}\right)}
\]

\[
W(\theta_k) = \exp\left(-j \cdot \frac{(M-1)}{2} \cdot \theta_k\right) \cdot \frac{\sin\left(\frac{M}{2} \cdot \theta_k\right)}{\sin\left(\frac{1}{2} \cdot \theta_k\right)}
\]

Note magnitude difference in even vs. odd lengths

Phase due to time delay

ECE 6560

Symmetric Rectangular Window

From the text: The sampled rectangle weighting function has a spectrum described by the Dirichlet kernel as shown in (3.9).

\[ W_{RECT}(\theta) = \sum_{n=-N/2}^{+N/2} 1 \cdot e^{-in\theta} = \frac{\sin(N\frac{\theta}{2})}{\sin(\frac{\theta}{2})} \]

This is called the Dirichlet kernel. It is seen to be the periodic extension of the \( \frac{\sin(2 \pi fT_{support}/2)}{2 \pi fT_{support}/2} \) spectrum, the transform of a continuous time rectangle function.
Rectangular Window

Figure 3.4 Spectrum of 100-point Rectangle Window with Zoom to Main Lobe

“even length”
Rectangular Window Low Pass Filter

- Applicable to all discretely sampled and then truncated (rect window) in length filter implementations!

The frequency domain convolution of the filter by a “window” function.
  - ripple
  - transition

Figure 3.5 Spectrum of Rectangle Windowed Prototype Filter Obtained as Convolution Between Spectrum of Prototype Filter and Spectrum of Rectangle Window
Rectangular Windowed
“Perfect” Low Pass Filter

Figure 3.6 Log Display of Spectrum to Emphasize High Levels of Out-of-band Side
Lobe Response of Rectangle Windowed Prototype Filter

Are There Better Windows?

- Something “smoother” that doesn’t have the ripples of a sinc function!
- Changes will likely make the “main-lobe” wider.
  - Reducing the high frequency content means more low frequency spectral content.
  - Smoother means less zero crossing ripple and smaller “side lobes” due to the sinc ripples.

- Idea: use one cycle of a cosine wave … with an appropriate “DC” offset
Figure 3.7 Raised Cosine Window and Transform Illustrating Sum of Translated and Scaled Dirichlet Kernels
Raised Cosine Window

• This can be thought of as:
  – A DC bias $\rightarrow$ transforms to a delta function in the frequency domain
  – A continuous cosine wave $\rightarrow$ delta functions at the positive and negative frequencies
  – A rectangular window about the zero time $\rightarrow$ sinc in frequency

• Fourier transform:
  – The DC and cosine are added
  – The sinc is convolved
  – Since the “sinc lobes” are at the delta function spacing, the signals add coherently near DC (widening the main lobe) and non-coherently at higher frequencies (reducing the ripples)
Raised Cosine Window

Figure 3.7 Raised Cosine Window and Transform Illustrating Sum of Translated and Scaled Dirichlet Kernels

Are There Better Windows?

• fred harris wrote one of the original window papers …
• For a paper describing window performance, see:
    http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=1455106
  – see Table 1 in the paper
# Window Functions

## Table 3-1 Windows Formed as Weighted Sum of Cosines

<table>
<thead>
<tr>
<th>Window Name</th>
<th>Weights</th>
<th>Maximum SideLobe</th>
<th>Main-Lobe Width Peak-to-First Zero</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rectangle</td>
<td>$a_0 = 1.0$</td>
<td>-13.5 dB</td>
<td>1</td>
</tr>
<tr>
<td>Hann</td>
<td>$a_0 = 0.5$</td>
<td>-32 dB</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>$a_1 = -0.5$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Hamming</td>
<td>$a_0 = 0.54$</td>
<td>-43 dB</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>$a_1 = -0.46$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Blackman (Approximation)</td>
<td>$a_0 = 0.42$</td>
<td>-58 dB</td>
<td>3</td>
</tr>
<tr>
<td></td>
<td>$a_1 = -0.50$</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>$a_2 = 0.08$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Blackman (Exact)</td>
<td>$a_0 = 0.42659$</td>
<td>-68 dB</td>
<td>3</td>
</tr>
<tr>
<td></td>
<td>$a_1 = -0.49656$</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>$a_2 = 0.07685$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Blackman-harris (3-Term)</td>
<td>$a_0 = 0.42323$</td>
<td>-72 dB</td>
<td>3</td>
</tr>
<tr>
<td></td>
<td>$a_1 = -0.49755$</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>$a_2 = 0.07922$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Blackman-harris (4-Term)</td>
<td>$a_0 = 0.35875$</td>
<td>-92 dB</td>
<td>4</td>
</tr>
<tr>
<td></td>
<td>$a_1 = -0.48829$</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>$a_2 = 0.14128$</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>$a_3 = -0.01168$</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Window Functions

- See MATLAB code example: WindowTest.m
  - The FFT of the windows … zero padded
- See MATLAB code example: WindowTest_FFT.m
  - Sin wave window FFT operation. Different FFT lengths.
  - Comparing “bin centered” and “between bin” outputs.

- See windowed filter example: WindowTest2.m
  - Truncated time domain sinc function … approximation of ideal filter
  - Windowed sinc functions with various time-domain windows.
Empirical Trade-off: Main-lobe vs. Side-lobe

Scanning Table 3-1 we can estimate the rate at which we can trade side-lobe levels for main-lobe width. This rate is approximately -22 dB/spectral bin so that in order to obtain – 60 dB sidelobes, we have to increase the main-lobe bandwidth to 2.7 \( f_s/N \). Remembering that the window’s two-sided main-lobe width is an upper bound to the filter’s transition bandwidth, we can estimate the transition bandwidth of a filter required to obtain a specified side-lobe level.

Equation (3.10) presents an empirically derived approximate relationship valid for window-based design while (3.11) rearranges (3.10) to obtain an estimate of the filter length required to meet a set of filter specifications.

\[
\Delta f_{\text{MINIMUM}} = \frac{f_s}{N}
\]

\[
\Delta f = \frac{f_s}{N} K(\text{Atten}) \approx \frac{f_s}{N} \frac{\text{Atten}(dB)-8}{14}
\]

\[
N \equiv \frac{f_s}{\Delta f} \frac{\text{Atten}(dB)-8}{14}
\]
Kaiser-Bessel Window

The primary reason we examined windows and their spectral description as weighted Dirichlet kernels was to develop a sense of how we trade window main-lobe width, for window side-lobe levels and in turn filter transition bandwidth and side-lobe levels.

Some windows perform this exchange of bandwidth for side-lobe level very efficiently while others do not.

The Kaiser-Bessel window is very effective while the triangle (or Fejer) window is not. The Kaiser-Bessel window is in fact a family of windows parameterized over $\beta$, the time-bandwidth product of the window. The main-lobe width increases with $\beta$ while the peak side-lobe level decreases with $\beta$. 
Kaiser-Bessel Window

\[ w(n) = \frac{I_0(\pi \beta \sqrt{1.0 - \left( \frac{n}{N/2} \right)^2})}{I_0(\pi \beta)} : -N/2 \leq n < N/2 \]

where \( I_0(x) = \sum_{k=0}^{\infty} \left( \frac{(x/2)^k}{k!} \right)^2 \)

\[ W(\theta) = \frac{N}{I_0(\pi \beta)} \frac{\sinh \left( \sqrt{\pi \beta} \sqrt{\left( \frac{\pi \beta}{2} - (N\theta/2)^2 \right)} \right)}{\sqrt{\left( \frac{\pi \beta}{2} - (N\theta/2)^2 \right)}} \]
Kaiser-Bessel Window Example

• See Matlab: KaiserBesselTest.m
  – Beta varied from $\pi/2$ to $4\pi$ in steps of $\pi/2$
  – FFT of resulting window
  – Main lobe width and magnitude of 1st sidelobe estimated
  – Note: a windowed filter is expected to have
    (1) lower sidelobe levels than the window and
    (2) a wider mainlobe than the window.

• See Matlab: KaiserBesselTest2.m
  – Sinc of width “127” times KaiserBessel
  – Beta varied from $\pi/2$ to $4\pi$ in steps of $\pi/2$
Example 3.1 Window Design of Low-Pass FIR Filter

Design a FIR filter with the following specifications:

Sample Rate \hspace{1cm} 100 kHz
Pass-Band Band Edge \hspace{1cm} ±10 kHz
Stop-Band Band Edge \hspace{1cm} ±15 kHz
Minimum Attenuation \hspace{1cm} 60 dB

Estimate Number of Taps, try and iterate!

\[
N \equiv (f_s/\Delta f) \times (\text{Atten(dB)}-8)/14
= (100/5) \times (60-8)/14 = 75 \text{ taps}
\]
Example 3.1

Figure 3.9 Time and Frequency Response of FIR Filter Windowed with Kaiser-Bessel: First Designed for 6-dB Band Edge, Second for Pass-band and Stop-band Edges
Example 3.1 MATLAB

- Chap3_4.m script
Kaiserord and Demo

• \([N, Wn, BTA, FILTYPE] = \text{kaiserord}(F, A, DEV, Fs)\)
  – generate parameters, including beta
  – uses fir1.m Matlab’s windowed filter generator function
  – see http://www.mathworks.com/help/signal/ref/kaiserord.html

• \(h1 = \text{fir1}(N, Wn, FILTYPE, \text{kaiser}(N+1, BTA), 'noscale')\)
  – generate a windowed filter using the defined window.

• see Kaiser_FilterGen.m
  – Previous example results in Nord = 73, Beat = 5.6533
  – Rather close to the text values computed/used.
The Parks-McClellan Filter
“remez exchange algorithm”

Filter Specification Parameters

- $f_s$: Sample Rate
- $f_1$: Frequency at End of Pass-Band
- $f_2$: Frequency at Start of Stop-Band
- $\delta_1$: Maximum Pass-Band Ripple
- $\delta_2$: Maximum Stop-Band Ripple

![Diagram of filter specification parameters](image)

**Figure 3.10** Parameters Required to Specify Sampled Data Low-Pass Filter

\[ N = function(f_s, f_1, f_2, \delta_1, \delta_2) \]  \hspace{1cm} (3.14)
FIRPM vs. REMEZ

- MATLAB has changed algorithm names
  - Old name remez – The Parks-McClellan FIR filter generation algorithm is based on the “remez exchange algorithm”.
  
  Therefore, the old implementation was called remez.
  
  - New name FIRPM – FIR “Parks-McClellan” filter generations.
  
  It works almost identically to remez.
  It has a full compliment of order estimators that go with it (like Butterworth Filters)
FIRPM Design Domain

- The window design of a FIR filter occurs in the time domain as the point-by-point product of a prototype impulse response with the smooth window sequence. The quality of the resultant design is verified by examining the transform of the windowed impulse response.

- By contrast, the equiripple design is performed entirely in the frequency domain by an iterative adjustment of the location of sampled spectral values to obtain a Tchebyschev approximation to a desired spectrum.

- The desired, or target, spectrum has accompanying tolerance bands that define acceptable deviations from the target spectrum in distinct spectral regions.
Equiripple Frequency Domain Design

Figure 3.11 Frequency Response of Equiripple Filter and Error Frequency Profile
Weighted Error Filter Design

• When the FIR filter has 2M+1 even symmetric coefficients, its spectrum can be expanded as a trigonometric polynomial in θ as shown in (3.15).

\[ H(\theta) = \sum_{n=0}^{M} a(n) \cos(n\theta) \]

• Defining a positive valued weighting function \( W(\theta) \) and the target function \( T(\theta) \), we can define the weighted error function \( E(\theta) \) as shown in (3.16).

\[ E(\theta) = W(\theta) [H(\theta) - T(\theta)] \]
Derivation of REMEZ

- The Mth order polynomial $H(\theta)$ is defined by $M+1$ coefficients $a(n)$ or equivalently by the $M-1$ local extrema $H(\theta_k)$ and the 2-boundary values at $H(\theta \text{ pass})$ and $H(\theta \text{ stop})$. The problem is that we don’t know the locations of the local extrema. The Remez multiple exchange algorithm rapidly locates these extremal positions by iterating from an initial guess of their positions to their actual positions.

- A ubiquitous design algorithm written by McClellan, Parks, and Rabiner expanded on the original Parks and McClellan design and has become the standard implementation of the Remez algorithm. It is embedded in most FIR filter design routines.

- Read the text for an explanation …
More Parks-McClellan References

• The interested reader should read the material presented in:
  – IEEE Papers
  – Books
    • Digital Filter Design by Parks and Burrus
Parameters

- **Sample Rate**
  - Satisfy the nyquist or desired sample rate criteria
- **Passband**
  - The highest frequency of interest
- **Stopband**
  - The frequency at which the stopband attenuation must be attained
- **Passband Ripple (typically 1% to 5%)**
  - The ripple allowed in the filter passband (note ripple may appear as a modulation distortion of the signal)
- **Stopband Ripple (typically –60 to –80 dB)**
  - The maximum sidelobes allowed in the stopband
Parameter Selection

- Relaxing constraints typically allows fewer FIR filter coefficients to be required
  - Wider percentage passbands (fpb/fs)
  - Wider transition bands (fstop-fpb)
  - More passband ripple
  - More stopband ripple (less stopband attenuation)
- Order estimators (Chap3_5b.m)
  - remezord_harris
  - remez_est
  - remezord2
  - firpmord (formerly remezord)
Example 3.2 Remez Design of Low-Pass FIR Filter
Design a low-pass FIR filter with the Remez algorithm that meets the same specifications as the window design Example 3.1. The expanded filter specifications are:

- Sample rate: 100 kHz
- Pass Band: ±10 kHz
- Stop Band: ±15 kHz
- Min. Atten.: 60 dB
- Pass-band Ripple: 0.1 dB (1.2%)

Chap3_6.m

h2=remez(53,[0 fpass fstop (fsample/2)]/(fsample/2),[1 1 0 0],[1 10]);
Example 3.2

Figure 3.17 Time and Frequency Response of FIR Filter Designed with the Remez Algorithm

Design Characteristic

- Uniform passband ripple
- Uniform stopband ripple

- Question
  - Do we really want a uniform stopband ripple if we are going to decimate the output?!
  
  - A filter spectrum has a decay rate related to the order of any discontinuities in the time domain signal and the signal derivatives.
    - 0\textsuperscript{th} order – equiripple
    - 1\textsuperscript{st} order – 1/f
    - 2\textsuperscript{nd} order – 1/f^2 … etc.
The REMEZ Discontinuity: Equiripple

Figure 3.18 Remez Impulse Response, Showing Detail of End Point, Close-up of End Point, and Spectra of Original Filter and of Filter with Clipped End Point

Achieving 1/f Ripple

- Find and remove the discontinuity …
  - Modify the discontinuous coefficients
  - “Window” REMEZ output
  - Zero the outer coefficients (make the filter +2 larger)

- Chap3_7.m

- Modified Filter
  - 1/f behavior
  - Side lobe increase
  - Passband ripple increase
Simple Noise Power Example

- Chap3_8
firpm Example Template

- see PMcc_FilterGen.m

- Also see: KaiserVsfirpm.m
MATLAB

• Related functions
  – kaiserord
  – firpmord
  – fir1
  – firpm

• Examples
  – Kaiser_FilterGen.m
  – PMcC_FilterGen.m
  – KaiserVsfirpm.m
Quantization

- All real-world FIR filters are quantized to the available bit precision
- Therefore, all real filters appears as a combination of the desired filter $h(n)$ minus the quantization error

\[
h_{\text{SCALED}} = h / \max(h)
\]

\[
h_{\text{QUANT}} = \text{round}(h_{\text{SCALED}} \ast 2^{(\text{bits}-1)}) / 2^{(\text{bits}-1)}
\]

\[
h_{\text{FIR}}(n) = h(n) - h_{\text{Qerror}}(n)
\]

- Quantization can (will) cause modifications to the desired filter response!
Quantization Error

Table 3.2 Integrated Side-Lobe Levels for Equiripple and 1/f Side-Lobe FIR Filters, Unquantized and 10-bit Quantized Versions

<table>
<thead>
<tr>
<th></th>
<th>Equiripple Side Lobes</th>
<th>1/f Side Lobes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Unquantized</td>
<td>-36.1 dB</td>
<td>-49.3 dB</td>
</tr>
<tr>
<td>Quantized (10-bits)</td>
<td>-35.4 dB</td>
<td>-45.1 dB</td>
</tr>
</tbody>
</table>

Figure 3.20 Frequency Response of Reference Filters with Equiripple and 1/f Side Lobes and of 10-bit Quantized Version of Same Filters
Stairstep Weight Function

```matlab
ff=[0 0.6 3.4 5.0 5.1 7.5 7.6 10.0 10.1 15.0 60.1 64.0];
ff=ff/64.0;
aa=[1 1 0 0 0 0 0 0 0 0 0 0];
ww=[ 1 1.5 3 4.5 6 21 ];
hh=remez(154, ff, aa, ww);
```

![Graphs showing the stairstep weight function](image)

**Figure 3.22** Remez Filter Time and Frequency Response Designed with Linear Frequency and with Staircase Frequency Weight Function

Tools for 1/f Improvement

• Harris’ “myrfr” matlab script
• Chap3_9.m
• Chap3_10.m
• Harris web site files
In-Band Ripple or
Why Passband Ripple is Bad

![Filter Diagram](image)

**Figure 3.24** Input and Output of a Linear Filter

\[ y(t) = A x(t - \tau) \]

\[ Y(\omega) = A X(\omega) e^{-j\omega \tau} = X(\omega) A e^{-j\omega \tau} \]

\[ H(\omega) = A e^{-j\omega \tau} \]

- The desired signal output is a delayed version of the input as shown
- To accomplish this, the filter passband requires the spectral characteristics shown (note: this refers to the passband, not the entire filter!)
Passband Ripple

• With passband ripple, the equivalent spectral results are:

\[ Y(\omega) = X(\omega)H(\omega) \]

\[ = X(\omega) [1 + \epsilon \cos(\omega T_p)] e^{-j \omega T_D} \]

The equivalent ripple in frequency
Time Domain Ripple

• Developing the equivalent representation

\[ Y(\omega) = X(\omega)H(\omega) \]
\[ = X(\omega) [1 + \epsilon \cos(\omega T_p)] e^{-j\omega T_D} \]
\[ = X(\omega)e^{-j\omega T_D} + \epsilon X(\omega)\cos(\omega T_p)e^{-j\omega T_D} \]
\[ = X(\omega)e^{-j\omega T_D} + \frac{\epsilon}{2} X(\omega)e^{-j\omega(T_D+T_p)} + \frac{\epsilon}{2} X(\omega)e^{-j\omega(T_D-T_p)} \]

• Estimating the inverse transform

\[ y(t) = x(t - T_D) + \frac{\epsilon}{2} x(t - (T_D + T_p)) + \frac{\epsilon}{2} x(t - (T_D - T_p)) \]

• The result shows the desired SOI and two additional terms, paired “echoes”, a pre-echo and a post-echo!
Time Domain Echoes

**Figure 3.26** Time Signals at Input and Output of Filter With Equiripple Spectral Response
Text Example: Magnitude Ripple

Figure 3.27 Impulse and Frequency Response of a FIR Filter. Detail of Passband Shows ±1 dB In-band Ripple. Spectrum of Input Signal is Fully Contained in Filter Passband

Magnitude Ripple (2)

Figure 3.28 Time-aligned Input and Output Signal Waveforms and Difference of Two Waveforms to Show Paired Echoes

Passband with Phase Ripple

Figure 3.29 Impulse, Magnitude Response, and Group Delay Response of an IIR Filter and of a Phase Equalized Version of Same Filter. Detail of Equalized Phase Shows ±2.2 degree In-Band Ripple.
Phase Ripple (2)

Figure 3.30 Time-Aligned Input and Output Waveforms from Nonuniform Phase IIR Filter with the Difference of Two Waveforms to Show Paired Echoes
Phase Equalized

**Figure 3.31** Time-Aligned Input and Output Waveforms from Phase Equalized IIR Filter with the Difference of Two Waveforms to Show Paired Echoes