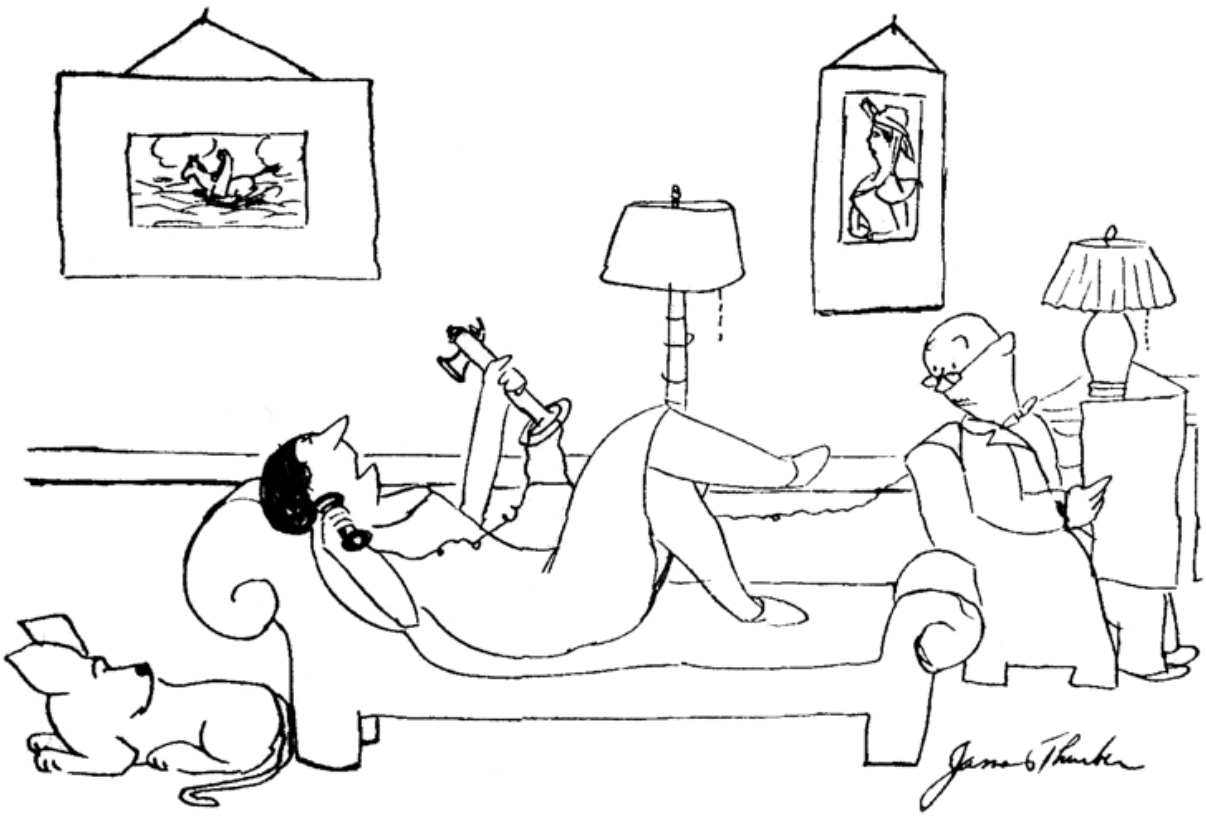


"Well, if I called the wrong number, why did you answer the phone?"

EE 311 **February 20, 2019**
Lecture 12

Chapter 5

****

SKIP KAISER FILTER

****

5.3 Advanced Window Functions

Kaiser Window

The Kaiser window function is different from the previous window functions in that it provides a window whose constants can be altered to achieve a given ripple specification.

The Kaiser window is formed from Bessel functions and the equation is given by:

$$w_K[n] = \frac{I_0(\beta)}{I_0(\alpha)}$$

where $I_0(\beta)$ and $I_0(\alpha)$ are zero order modified Bessel functions of the first kind. The constant α is used to adjust the side lobe attenuation and is related to β by the equation:

$$\beta = \alpha\sqrt{1-\left(\frac{2n}{L-1}\right)^2}$$

Note that β is a function of α and n.

We can compute the Bessel functions from a power series:

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

Alternatively, MATLAB® has a Kaiser window generator which accepts the filter length and the parameter α as arguments.

Figure 5.20 shows a typical low pass filter with the symbols and definitions for specifying the filter parameters. Using this figure as a guide we make the following definitions:

ε - the ripple or the amount of overshoot and undershoot that can be tolerated in a given band.

ε_p - pass band ripple. For a low pass filter whose pass band gain has been normalized to one the maximum gain in the pass band is $1+\varepsilon$ and the minimum gain in the pass band is $1-\varepsilon$.

ε_{st} - stop band ripple. For a low pass filter the maximum gain in the stop band is ε_{st}.

A_p - Attenuation in the pass band. This parameter is typically given in decibels and is related to the pass band ripple by

$$A_p = -20\log_{10}[(1+\varepsilon_p)/(1-\varepsilon_p)].$$

A_{st} - Attenuation in the stop band. This parameter is typically given in decibels and is related to the stop band ripple by $A_{st} = -20\log_{10}(\varepsilon_{st})$.

f_c - the cut off frequency of the filter. This is the frequency at which the gain is 0.5 for a low pass filter whose pass band gain has been normalized to one.
f_p - The uppermost frequency of the pass band for a low pass filter.
f_{st} - the lowermost frequency of the stop band edge for a low pass filter.

With these definitions in mind we can outline the design procedure for an FIR filter using a Kaiser window.

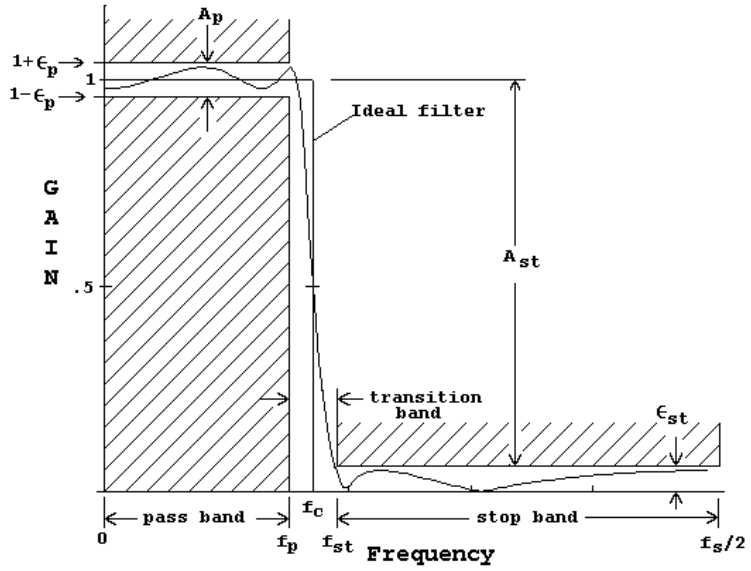


Figure 5.20
This figure gives the variables for a low pass filter design using the Kaiser window.

1. Calculate the attenuation from the filter specifications. The attenuation can be found from the following relationships (A is in decibels):

$$A = \max(A_p, A_{st}) \text{ or,}$$
$$\varepsilon = \min(\varepsilon_p, \varepsilon_{st})$$
$$A = -20\log_{10}(\varepsilon)$$

2. Calculate α from the following equations[1] :

$$\alpha = \begin{cases} .1102(A-8.7) & A \geq 50 \\ .5842(A-21)^{.4} + .07886(A-21) & 21 < A < 50 \\ 0 & A \leq 21 \end{cases}$$

3. Calculate the filter length L from the following equations:

$$L = 1 + \frac{K_A f_s}{f_{st} - f_p} \text{ where, } K_A = \begin{cases} \dfrac{A-7.95}{14.36} & A > 21 \\ .922 & otherwise \end{cases}$$

The value of L should be rounded up to the next odd integer.

4. Calculate the Kaiser window function using the following relationships:

$$w_K[n] = \frac{I_0(\beta)}{I_0(\alpha)}$$

$$\beta = \alpha\sqrt{1-\left(\frac{2n}{L-1}\right)^2}$$

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

Alternatively, we can use MATLAB® to calculate the Kaiser window for us using the single command line:

`kaiser(L,α)`

5. Find the impulse response function for an ideal filter that meets the specifications. This is done by using the Fourier series. For an ideal low pass filter of odd length the filter coefficients are given by:

$$C_k = \frac{1}{k\pi}\sin(2\pi k f_c / f_s)$$

6. To make the filter causal and finite we multiply the impulse response coefficients by the corresponding window coefficients to get the final coefficients for the filter. MATLAB® has a function `fir1` which can be used to apply a Kaiser window to a filter created using the Fourier series.

Figure 5.21
Design procedure for creating an FIR filter with a Kaiser window.

The shape of the Kaiser window is dependent on the set of parameters chosen for the filter being designed. For this particular example the time and frequency domain responses are shown in Figure 5.23.

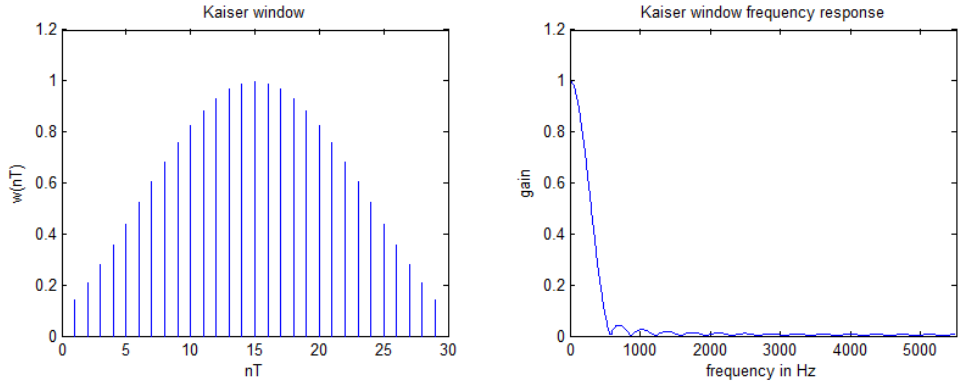


Figure 5.23
A Kaiser window. The impulse response is shown on the left and the frequency plot is on the right.

EE 311 **February 20, 2012**

Kaiser/Hamming in class

SOLUTION

Design a windowed FIR filter to meet the following specification using a Kaiser window. Compare the frequency response to that of a Hamming windowed filter of the same order.

Pass band	0 to 2500
Stop band	3400 to fs/2
Pass band Ripple	.02
Stop band Ripple	0.03
Sample Frequency	11025

```
%KaiserHamming.m
fs = 11025;
fpass = 2500;
fstop = 3400;
fc = fpass + (fstop-fpass)/2;
rp = .01;rs = .03;
%Kaiser Window
eps = min(rp, rs);
A = -20*log10(eps);
Ka = (A - 7.95)/14.36;
alpha = .5842*(A - 21)^.4 + .07886*(A - 21);
L = ceil(1 + Ka*fs/(fstop - fpass));
num = fir1(L-1, fc/(fs/2), kaiser(L, alpha));
[Hk f] = freqz(num, 1, 1024, fs);
figure(1);clf;
plot(f, abs(Hk));
%Hamming window
N = L - 1;
numH = fir1(N, fc/(fs/2), hamming(N+1));
[Hh f] = freqz(numH, 1, 1024, fs);
hold on;
plot(f, abs(Hh), 'r');
```

5.4 Frequency Sampling FIR filters

The frequency sampling method relies on the discrete Fourier transform (DFT). It begins with an ideal frequency response for a filter which is sampled at regular intervals. We can apply the inverse DFT (IDFT) to these samples to get the impulse response and thereby arrive at a transfer function. Some restrictions apply since we want to have a filter that has real coefficients and a linear phase response.

The equation for the IDFT is:

$h[n] = \frac{1}{L}\sum_{k=0}^{L-1} H(k) e^{j\frac{2\pi kn}{L}}$ where L is the length of the filter.

The equation for the z transform is:

$H(z) = \sum_{n=0}^{L-1} h[n] z^{-n}$

If we take the values of $H(k)$ to be the frequency samples, we can put them in the equation for the IDFT and substitute the calculated value of $h[n]$ into the equation of the z transform to get a transfer function. This gives the following result:

$$H(z) = \sum_{n=0}^{L-1}\left[\frac{1}{L}\sum_{k=0}^{L-1} H(k) e^{j\frac{2\pi kn}{L}}\right] z^{-n}$$

Rearranging the order of summation gives:

$$H(z) = \sum_{k=0}^{L-1}\frac{H(k)}{L}\sum_{n=0}^{L-1} e^{j\frac{2\pi kn}{L}} z^{-n} = \sum_{k=0}^{L-1}\frac{H(k)}{L}\sum_{n=0}^{L-1} W_k^n z^{-n}$$

where $W_k = e^{j2\pi k/L}$. This equation can be simplified to

$$H(z) = \sum_{k=0}^{L-1}\frac{H(k)}{L}\cdot\frac{z^L - 1}{z^{L-1}(z - W_k)} \qquad \textbf{(5.10)}$$

This transfer function appears to be that of a recursive filter since it has poles at locations other than the origin. However, notice that the numerator term $z^L - 1$ can be factored since its roots are the L roots of unity – namely $e^{j2\pi k/L} = W_k$. Thus the zeros in the transfer function cancel the poles that are not at the origin. Since *H(z)* is written as a sum, it is necessary to expand the sum and create a common denominator before canceling the poles and zeros. If we do this operation the result is the following transfer function:

$$H(z) = \frac{1}{Lz^{L-1}}\sum_{k=0}^{L-1} H(k)\cdot\left[\prod_{\substack{i=0\\ i\neq k}}^{L-1}(z - e^{j\frac{2\pi i}{L}})\right] \qquad \textbf{(5.11)}$$

The design process for frequency sampling filters can be summarized as follows:

1. From the frequency response plot for an ideal filter choose L equally spaced samples. These are the $H(k)$.

2. Use $h[n]=\frac{1}{L}\sum_{k=0}^{L-1}H(k)e^{j\frac{2\pi kn}{L}}$ to find the impulse response of the filter.

3. Use equation 5.11 to find the transfer function for the FIR filter.

You cannot arbitrarily choose the frequency sampling points if you want linear phase and real coefficients. See the text for how these conditions modify the transfer function. After some manipulation the impulse response can be written in terms of trigonometric functions as follows:

Symmetry	***L***	**Impulse Response**	**Restriction**
Symmetric	Odd	$h[n]=\frac{1}{L}\left(A(0)+2\sum_{k=1}^{\frac{L-1}{2}}(-1)^k A(k)\cos\frac{(2n+1)\pi k}{L}\right)$	none
Symmetric	Even	$h[n]=\frac{1}{L}\left(A(0)+2\sum_{k=1}^{\frac{L}{2}-1}(-1)^k A(k)\cos\frac{(2n+1)\pi k}{L}\right)$	$A(L/2)=0$
Anti-symmetric	Odd	$h[n]=\frac{2}{L}\sum_{k=1}^{\frac{L-1}{2}}(-1)^{k+1} A(k)\sin\frac{(2n+1)\pi k}{L}$	$A(0)=0$
Anti-symmetric	Even	$h[n]=\frac{1}{L}\left((-1)^{\frac{L}{2}+n}A(L/2)+2\sum_{k=1}^{\frac{L}{2}-1}(-1)^k A(k)\sin\frac{(2n+1)\pi k}{L}\right)$	$A(0)=0$

Table 5.7
Impulse response functions for frequency sampling filters.

The equations above are implemented in MATLAB® in the function fir2.

Example 5.10
Use MATLAB® and frequency sampling to design two low pass filters. Both filters should have a length of 21 with a sample frequency of 11,025 Hz. For the first filter allow the transition band to be of zero width with a pass band from 0 Hz to 2,500 Hz (stop band from 2,500 Hz to $f_s/2$ Hz). For the second filter allow the transition band to be 750 Hz wide so that the stop band goes from 2,500 Hz to 3,250 Hz.

Solution:
MATLAB® implements the frequency sampling filter equations in a function called `fir2`, which allows a user to add a window to the filter so designed. For this example we will use a rectangular window. The frequency samples are obtained by sampling an idealized filter based on the number of terms required. The idealized filter is specified by two vectors which form a set of points defining line segments in the idealized filter. The code below creates the two filters and plots their magnitude response along with the ideal filter which was used in the frequency sampling method.

```
fs = 11025;
```

```
L = 21;N = L-1;
fpass = 2500;fstop = 2500;
F = [0 fpass/(fs/2) fstop/(fs/2) 1]; %normalized to fs/2
A = [1 1 0 0];
num = fir2(N, F, A, rectwin(L));
figure(1);
[H f] = freqz(num, 1, 1024, fs);
plot(f, abs(H));                    %first filter
hold on;
plot(F*fs/2, A);                    %Idealized filter
%
fpass = 2500;
fstop = 3250;
F = [0 fpass/(fs/2) fstop/(fs/2) 1];
A = [1 1 0 0];
num = fir2(N, F, A, rectwin(L));
[H f] = freqz(num, 1, 1024, fs);
plot(f, abs(H));                    %second filter
hold on;
plot(F*fs/2, A);                    %Idealized filter
```

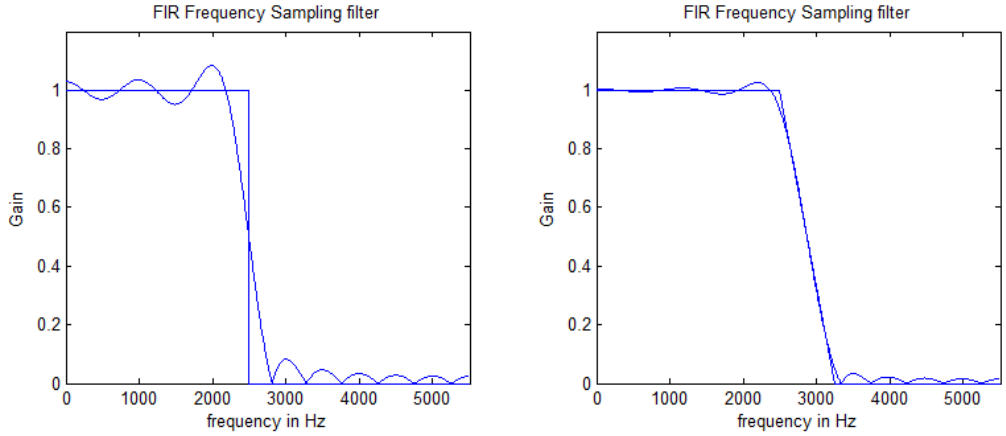


Figure 5.27
The leftmost figure shows a 20th order FIR filter created from an ideal low pass filter. On the right, the filter is also of 20th order but the addition of the transition band had reduced the ripple.

Frequency sampling provides the designer with a tradeoff between the amount of ripple and the width of the transition band which is similar to the tradeoff in the window design technique.

In general, the design equations given in Table 5.7 can be thought of as a set of equations which fit a curve through a set of frequency sample points.

EE 311 **February 19, 2018**
Frequency Sampling

Use MATLAB® and frequency sampling to design three low pass filters. All three filters should have a length of 31 with a sample frequency of 11,025 Hz. For the first filter allow the transition band to be of zero width with a pass band from 0 Hz to 2,700 Hz (stop band from 2,700 Hz to $f_s/2$ Hz). For the second filter allow the transition band to be 500 Hz wide so that the stop band goes from 3,200 Hz to fs/2 Hz. For the third filter allow the transition band to be 1000 Hz wide so that the stop band goes from 3,700 Hz to fs/2. Plot the magnitude of all three filters on the same axis with different colors.

Solution:
MATLAB® implements the frequency sampling filter equations in a function called `fir2`,

```
%FreqSampling.m
fs = 11025;
L = 21;N = L-1;
fpass = 2700;fstop = 2700;
F = [0 fpass/(fs/2) fstop/(fs/2) 1]; %normalized to fs/2
A = [1 1 0 0];
num = fir2(N, F, A, rectwin(L));
figure(1);
[H f] = freqz(num, 1, 1024, fs);
plot(f, abs(H), 'b');                    %first filter
hold on;
plot(F*fs/2, A, 'k');                    %Idealized filter
%
fpass = 2700;
fstop = 3200;
F = [0 fpass/(fs/2) fstop/(fs/2) 1];
A = [1 1 0 0];
num = fir2(N, F, A, rectwin(L));
[H f] = freqz(num, 1, 1024, fs);
plot(f, abs(H), 'r');                    %second filter
%
fpass = 2700;
fstop = 3700;
F = [0 fpass/(fs/2) fstop/(fs/2) 1];
A = [1 1 0 0];
num = fir2(N, F, A, rectwin(L));
[H f] = freqz(num, 1, 1024, fs);
plot(f, abs(H), 'g');                    %third filter
```

5.5 The Parks-McClellan Design Technique for FIR filters

The Parks-McClellan method of designing linear phase FIR filters overcomes these faults and allows a filter designer to produce a filter that is in some sense optimal [7][8]. The term *optimal* means that a filter is produced which meets some design criteria and it is the lowest order linear phase FIR filter that can meet those criteria. A low order filter is important since it implies fewer calculations in the implementation process. This may lead to less hardware or the possibility of a higher sampling rate.

The simplest definition for the error is to take the difference between the filter's frequency response and the ideal filter. This difference will be both positive and negative. Since the sign of the error is not important to us we will take the absolute value of the difference.

$$E(\Omega) = W(\Omega)\left|H_D(e^{j\Omega}) - H(e^{j\Omega})\right|$$

In this equation $H_D(e^{j\Omega})$ is the ideal filter which, for a low pass filter is normally 1 in the pass band and 0 in the stop band. The filter we are designing is given by $H(e^{j\Omega})$. $W(\Omega)$ is the weighting function and is a function of the frequency. We wish to find a filter such that the maximum value of this weighted error is minimized over a specified band of frequencies. This error is sometimes referred to as the *minimax* or the *Chebyschev* error criterion.

The *z*-transform for an FIR filter can be written as

$H(z) = \sum_{n'=0}^{M} h[n']z^{-n'}$, where M is the filter order.

If this is a linear phase filter, $h[n']$ is symmetric. If we take M to be even and shift $H(z)$ in time by $M/2$ we can write

$$H(z) = \sum_{n'=-M/2}^{M/2} h[n'+M/2]z^{-n'-M/2}$$

From this equation we factor out $z^{-M/2}$ to get

$$H(z) = z^{-M/2}\sum_{n'=-M/2}^{M/2} h[n'+M/2]z^{-n'}$$

Change variables by letting $n = n' + M/2$ to get

$$H(z) = z^{-M/2}\sum_{n=0}^{M} h[n]z^{-n+M/2}$$

Since $h[n]$ is symmetric (5.8) we have

$$H(z) = z^{-M/2}\left\{h[0](z^{M/2} + z^{-M/2}) + h[1](z^{M/2-1} + z^{-(M/2-1)}) + \cdots + h[M/2-1](z^{-1} + z^{+1}) + h[M/2]\right\}$$

If we look at the frequency response of this function we must substitute $e^{j\omega}$ for z. In general, from Euler's relation,

$(e^{jx} + e^{-jx}) = 2\cos(x)$.

We can therefore write the frequency response as a sum of cosines.

$$H(e^{j\omega}) = h[M/2]e^{j\Omega M/2} + 2\sum_{n=0}^{M/2} h[M/2]\cos([M/2-n]\Omega) \qquad \textbf{(5.14)}$$

Without any loss in generality we can take $h[n]$ to be symmetric and centered about the origin rather than the $M/2$ point. If we do this equation 5.14 becomes

$$H(e^{j\Omega}) = h[0] + 2\sum_{n=1}^{M/2} h[n]\cos(n\Omega)$$

This equation can be rewritten as

$$H(e^{j\Omega}) = \sum_{n=0}^{M/2} c_n \cos(n\Omega) \qquad \textbf{5.15}$$

Where $c_0 = h[0]$ and $c_n = 2h[n]$ for $n = 1$ to $M/2$

A Chebyshev polynomial, which is discussed in more detail in Chapter 6 when we talk about Chebyshev IIR filters, is defined as

$V_N(x) = \cos(N\cos^{-1}(x))$ = nth order polynomial in x.

If we let $x = \cos(\Omega)$ the Chebyshev polynomial becomes

$$V_N(\cos(\Omega)) = \cos(N\Omega)$$

Equation 5.15 can be written as $H(e^{j\Omega}) = \sum_{n=0}^{M/2} c_n \cos(n\Omega)$

$$\sum_{n=0}^{M/2} c_n \cos(n\Omega) = \sum_{n=0}^{M/2} c_n \left[\sum_{k=0}^{n} \beta_{nk} \cos^k \Omega\right] = \sum_{n=0}^{M/2} c_n' \cos^n \Omega$$

$H(e^{j\Omega}) = \sum_{n=0}^{M/2} c_n x^n = P(x)$ where $x = \cos(\Omega)$,

Thus we have shown that a low pass type 1 filter with linear phase and of even order can be represented by a trigonometric polynomial. Type 2, 3, and 4 filters can be similarly expressed.

The Alternation theorem [2] provides the basis for optimizing the filter so as to minimize $E(\Omega)$ for a given filter order. From the Alternation theorem we can state:

If $H(e^{j\Omega})$ has at least $M/2 + 2$ *extremal frequencies* $\Omega_1 < \Omega_2 < \ldots < \Omega_{M/2+2}$ such that:

A) The weighted error alternates in sign: $E(\Omega_1) = -E(\Omega_2) = E(\Omega_3) = -E(\Omega_4) = \ldots$

B) The maximum error $|E_{\max}| = |E(\Omega_1)| = |E(\Omega_2)| = \cdots = |E(\Omega_k)|$

then $H(e^{j\Omega})$ is the unique best approximation for the ideal filter in minimizing the maximum weighted error in all of the frequency bands.

From the Alternation theorem we can write

$$E(\Omega_i) = W(\Omega_i)\left|H_D(e^{j\Omega_i}) - H(e^{j\Omega_i})\right| = \pm|E_{\max}|$$

Where Ω_i is one of the extremal frequencies.

Parks and McClellan found that the error can be written as

$$E(\Omega_i) = \frac{\sum_{i=1}^{M/2+2} H_D(e^{j\Omega_i})}{\sum_{i=1}^{M/2+2} b_i(-1)^{i+1}/W(\Omega_i)} \qquad \text{where } b_i = \prod_{\substack{n=1 \\ n\neq i}}^{M/2+2} \frac{1}{\cos(\Omega_i) - \cos(\Omega_n)}$$

Using the above information Parks and McClellan developed an algorithm for finding the optimum filter. They began by making an initial guess at where the $M/2+2$ extremal frequencies were located for a given filter. Lagrange interpolation (see Example 5.11) using these frequencies produced a polynomial that passed through the estimated extremal frequency points. With this done, they have a continuous polynomial that passes through all of the *estimated* extremal frequency points. The extremal points in this polynomial however, will not generally be the same as the estimated extremal points. If they are not, the estimated extremal points are exchanged for the extremal points in the interpolated polynomial (see Figure 5.28). With this new set of extremal points a new polynomial can be interpreted. This process converges and ends when the estimated extremal points match those of the new polynomial.

The process of exchanging the estimated extremal frequencies for those in the newly interpolated polynomial is called the Remez exchange algorithm.

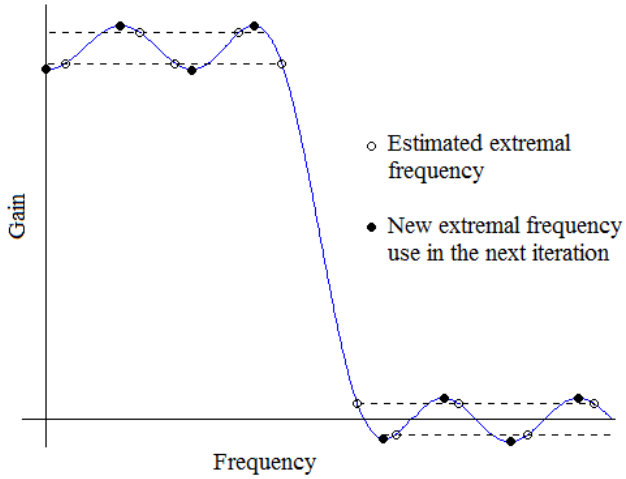


Figure 5.28
Open circles indicate initial estimates for extremal frequencies. Dark circles indicate the extremal frequency actual locations on the interpolated polynomial. In the Remez exchange algorithm, the dark circles are used as the extremal frequency estimates in the next iteration.

In summary, Parks and McClellan have developed a method of designing linear phase FIR filters which are minimax optimal, based on the alternation theorem and making use of the Remez exchange algorithm. They published a FORTRAN program for their method in 1972 and it has enjoyed wide use. The program requires that the user input the band edges, the ripple, the weighting factor, and the value of the gain in the bands. Empirical formulas are used to estimate the minimum filter order. The following equation can be used to estimate the order [4].

$$N = \frac{-20\log_{10}(\sqrt{\varepsilon_1\varepsilon_2}) - 13}{14.6\Delta f \,/\, f_s}$$

where ε_1 and ε_2 are the ripple numbers for the pass and stop bands respectively, Δf is the transition band width, and f_s is the sample frequency.

Alternatively, the filter can be designed with a very rough estimate of the order. The filter order can then be increased or decreased to find the minimum order which meets the specifications. The Parks-McClellan method has been implemented in MATLAB® and has a function name of `firpm`. The function `firpmord` estimates the order and incorporates a weighting function. The example below illustrates the use of these functions.

Example 5.12
Use the Parks-McClellan method to design an optimal low pass FIR linear phase filter that meets the following specifications:

Sample frequency:	11,025Hz
Pass band :	0 - 2300Hz
Pass band ripple:	.03
Stop band:	3500 Hz – fs/2
Stop band ripple	.02
Weighting factor:	1

Solution:
Using MATLAB® we enter the following lines:

```
fs = 11025;
fpass = 2300;fstop = 3000;
rpass = 0.03;rstop = 0.02;
F = ([fpass fstop]);        %define the frequency bands
M = ([1 0]);                %gain within the bands
Err = [rpass rstop];        %Ripple in the bands
%W is the weighting function within each band
[N F A W] = firpmord(F, M, Err, fs);
num = firpm(N, F, A, W);
figure(1)
[H f] = freqz(num, 1, 1024, fs);
plot(f, abs(H));
figure(2)
zplane(num, 1);             %Pole/zero plot
```

In these equations, MATLAB® requires that F be a vector containing the normalized band edges and M is a vector containing the values of the magnitude at the band edges for the ideal filter. The frequencies in F need not be normalized if the sample frequency, f_s is included in `firpmord`. N is the estimated filter order.

The answer comes back in a vector `num`, which contains the numerator coefficients for the filter. Since the order *N*, is an estimate of the order needed it is prudent to check the filter's frequency response to verify that ripple specifications have been met. In some cases, it is necessary to increase the order by a small amount. The results are shown in Figure 5.30 and 5.31.

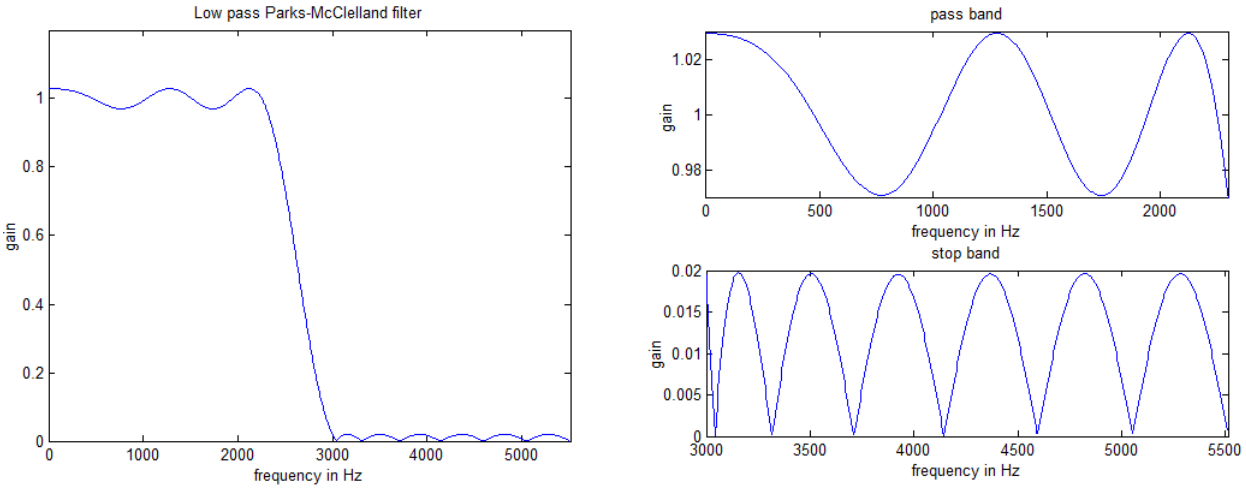


Figure 5.30
The magnitude plot for a Parks-McClellan filter meeting the specifications for this example.

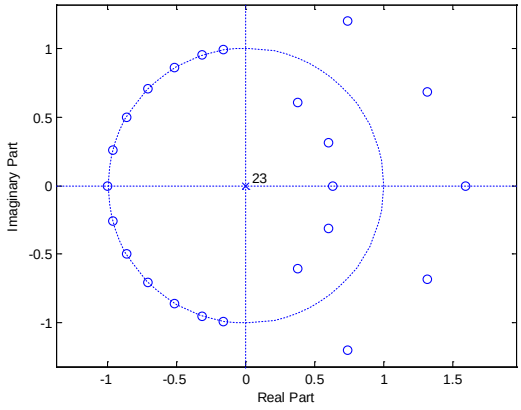


Figure 5.31
The z-plane pole/zero plot for the 23rd order low pass FIR filter.

Example 5.13
Use the Parks-McClellan method to design an optimal band stop FIR linear phase filter that meets the following specifications:

Sample frequency:	44,100Hz
Pass band 1 :	0 - 3600Hz
Pass band 1 ripple:	.03
Stop band:	5820 - 9600Hz
Stop band ripple:	.02
Pass band 2	7600 Hz - fs/2
Pass band 2 ripple:	.03

Solution:
The MATLAB® code is very similar to that of Example 5.12 except for the first few lines. The new code and results are shown in Figure 5.32 and 5.33.

```
fs = 44100;
fps1 = 3600;fps2 = 9600;
fst1 = 5820;fst2 = 7600;
rpass = 0.01;rstop = 0.01;
F = ([fps1 fst1 fst2 fps2]);
M = ([1 0 1]);
Dev = [rpass rstop rpass];
[N F A W] = firpmord(F, M, Dev,
fs);
num = firpm(N, F, A, W);
```

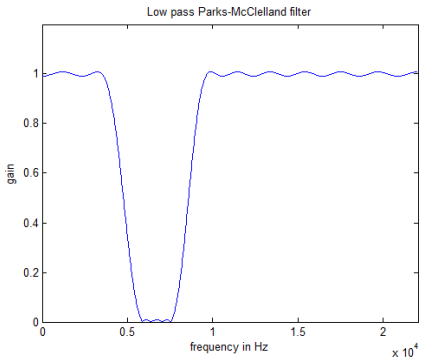
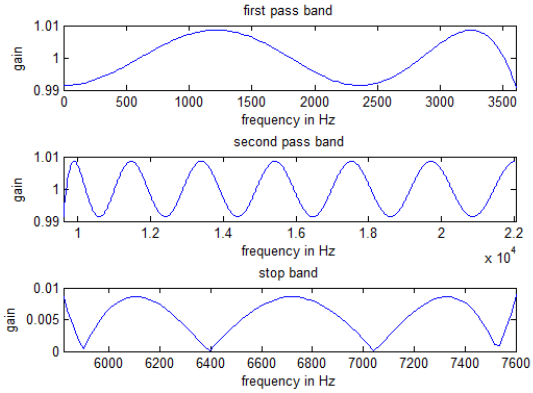


Figure 5.32
An equiripple band pass filter designed using Parks-McClellan algorithm.



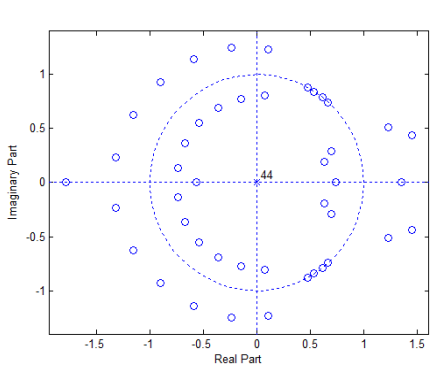


Figure 5.33
Left is the ripple in the pass and stop bands showing that the filter meets specification. Right is the pole/zero plot for the 44th order FIR filter.

EE 311 **February 21, 2018**
ParksMcClellan/Hamming in class

Design a windowed FIR filter to meet the following specification using the Parks-McClellan design method. Compare the frequency response to that of a Hamming windowed filter of the same order.

Pass band	0 to 2500
Stop band	3400 to fs/2
Pass band Ripple	.02
Stop band Ripple	0.03
Sample Frequency	11025

```
%PMHamming.m
fs = 11025;
fpass = 2500;
fstop = 3400;
fc = fpass + (fstop-fpass)/2;
rp = .01;rs = .03;
%Parks-McClellan
F = ([fpass fstop]);        %define the frequency bands
M = ([1 0]);                %gain within the bands
Err = [rp rs];              %Ripple in the bands
%W is the weighting function within each band
[N F A W] = firpmord(F, M, Err, fs);
N = N + 2;    %*** ADJUST ORDER ***
num = firpm(N, F, A, W);
figure(1);clf;
[H f] = freqz(num, 1, 1024, fs);
plot(f, abs(H));
%Hamming window
numH = fir1(N, fc/(fs/2), hamming(N+1));
[Hh f] = freqz(numH, 1, 1024, fs);
hold on;
plot(f, abs(Hh), 'r');
%Blow-ups
figure(2);clf;
subplot(2, 1, 1);
plot(f, abs(H));
hold on;
plot(f, abs(Hh), 'r');
axis([0 fpass 1-rp 1+rp]);
subplot(2, 1, 2);
plot(f, abs(H));
hold on;
plot(f, abs(Hh), 'r');
axis([fstop fs/2 0 rs]);
```