Design of Minimum-Phase FIR Filters

- Linear-phase FIR filters with narrow transition bands are of very high order, and as a result have a very long group delay that is about half the filter order
- By relaxing the linear-phase requirement, it is possible to design an FIR filter of lower order thus reducing the overall group delay and the computational cost

Design of Minimum-Phase FIR Filters

- A very simple method of minimum-phase FIR filter is described next
- Consider an arbitrary FIR transfer function of degree N:

$$H(z) = \sum_{n=0}^{N} h[n]z^{-n} = h[0]\prod_{k=1}^{N}(1-\xi_k z^{-1})$$

Design of Minimum-Phase FIR Filters

- The mirror-image polynomial to $H(z)$ is given by

$$\hat{H}(z) = z^{-N} H(z^{-1})$$
$$= \sum_{n=0}^{N} h[N-n] z^{-n} = h[N] \prod_{k=1}^{N} (1 - z^{-1}/\xi_k)$$

- The zeros of $\hat{H}(z)$ are thus at $z = 1/\xi_k$, i.e., are reciprocal to the zeros of $H(z)$ at $z = \xi_k$

Design of Minimum-Phase FIR Filters

- As a result,

$$G(z) = H(z)\hat{H}(z) = z^{-N} H(z) H(z^{-1})$$

has zeros exhibiting mirror-image symmetry in the z-plane and is thus a Type 1 linear-phase transfer function of order $2N$

- Moreover, if $H(z)$ has a zero on the unit circle, $\hat{H}(z)$ will also have a zero on the unit circle at the conjugate reciprocal position

Design of Minimum-Phase FIR Filters

- Thus, unit circle zeros of $G(z)$ occur in pairs
- On the unit circle we have

$$\left|H(e^{j\omega})\right|^2 = \breve{G}(\omega)$$

$$\Rightarrow \quad \breve{G}(\omega) \geq 0$$

- Moreover, the amplitude response $\breve{G}(\omega)$ has double zeros in the frequency range $[0, \pi]$

Design of Minimum-Phase FIR Filters

- **Design Procedure –**
- **Step 1**: Design a Type 1 linear-phase transfer function $F(z)$ of degree $2N$ satisfying the specifications:

$$1-\delta_p^{(F)} \le \breve{F}(\omega) \le 1+\delta_p^{(F)} \quad \text{for} \quad \omega \in [0, \omega_p]$$

$$-\delta_s^{(F)} \le \breve{F}(\omega) \le \delta_p^{(F)} \quad \text{for} \quad \omega \in [\omega_s, \pi]$$

- Note that $F(z)$ has single unit circle zeros

Design of Minimum-Phase FIR Filters

- **Step 2**: Determine the linear-phase transfer function

$$G(z) = \delta_s^{(F)} z^{-N} + F(z)$$

- Its amplitude response satisfies

$$1 + \delta_s^{(F)} - \delta_p^{(F)} \le \breve{G}(\omega) \le 1 + \delta_s^{(F)} + \delta_p^{(F)} \quad \text{for} \quad \omega \in [0, \omega_p]$$

$$0 \le \breve{G}(\omega) \le 2\delta_s^{(F)} \quad \text{for} \quad \omega \in [\omega_s, \pi]$$

Design of Minimum-Phase FIR Filters

- Note that $G(z)$ has double zeros on the unit circle and all other zeros are situated with a mirror-image symmetry
- Hence, it can be expressed in the form

$$G(z) = z^{-N} H_m(z) H_m(z^{-1})$$

where $H_m(z)$ is a minimum-phase transfer function containing all zeros of $G(z)$ that are inside the unit circle and one each of the unit circle double zeros

Design of Minimum-Phase FIR Filters

- **Step 3**: Determine $H_m(z)$ from $G(z)$ by applying a spectral factorization
- The passband ripple $\delta_p^{(F)}$ and the stopband ripple $\delta_s^{(F)}$ of $F(z)$ must be chosen to ensure that the specified passband ripple δ_p and the stopband ripple δ_s of $H_m(z)$ are satisfied

Design of Minimum-Phase FIR Filters

- It can be shown

$$\delta_p^{(F)} = \sqrt{1 + \frac{\delta_p}{1+\delta_s}} - 1, \quad \delta_s^{(F)} = \sqrt{\frac{2\delta_s}{1+\delta_s}}$$

- An estimate of the order N of $H_m(z)$ can be found by first estimating the order of $F(z)$ and then dividing it by 2
- If the estimated order of $F(z)$ is an odd integer, it should be increased by 1

FIR Digital Filter Design Using MATLAB

- Order Estimation -
- Kaiser's Formula:

$$N \cong \frac{-20\log_{10}(\sqrt{\delta_p \delta_s})}{14.6(\omega_s - \omega_p)/2\pi}$$

- Note: Filter order N is inversely proportional to transition band width $(\omega_s - \omega_p)$ and does not depend on actual location of transition band

FIR Digital Filter Design Using MATLAB

- Hermann-Rabiner-Chan's Formula:

$$N \cong \frac{D_\infty(\delta_p,\delta_s) - F(\delta_p,\delta_s)[(\omega_s-\omega_p)/2\pi]^2}{(\omega_s-\omega_p)/2\pi}$$

where

$$D_\infty(\delta_p,\delta_s) = [a_1(\log_{10}\delta_p)^2 + a_2(\log_{10}\delta_p) + a_3]\log_{10}\delta_s + [a_4(\log_{10}\delta_p)^2 + a_5(\log_{10}\delta_p) + a_6]$$

$$F(\delta_p,\delta_s) = b_1 + b_2[\log_{10}\delta_p - \log_{10}\delta_s]$$

with $a_1 = 0.005309,\ a_2 = 0.07114,\ a_3 = -0.4761$

$$a_4 = 0.00266,\ a_5 = 0.5941,\ a_6 = 0.4278$$

$$b_1 = 11.01217,\quad b_2 = 0.51244$$

FIR Digital Filter Design Using MATLAB

- Formula valid for $\delta_p \geq \delta_s$
- For $\delta_p < \delta_s$, formula to be used is obtained by interchanging δ_p and δ_s
- Both formulas provide only an estimate of the required filter order N
- Frequency response of FIR filter designed using this estimated order may or may not meet the given specifications
- If specifications are not met, increase filter order until they are met

FIR Digital Filter Design Using MATLAB

- MATLAB code fragments for estimating filter order using Kaiser's formula

```
num = - 20*log10(sqrt(dp*ds)) - 13;
den = 14.6*(Fs - Fp)/FT;
N = ceil(num/den);
```

- M-file `firpmord` implements Hermann-Rabiner-Chan's order estimation formula

FIR Digital Filter Design Using MATLAB

- For FIR filter design using the Kaiser window, window order is estimated using the M-file `kaiserord`
- The M-file `kaiserord` can in some cases generate a value of N which is either greater or smaller than the required minimum order
- If filter designed using the estimated order N does not meet the specifications, N should either be gradually increased or decreased until the specifications are met

Equiripple FIR Digital Filter Design Using MATLAB

- The M-file `firpm` can be used to design an equiripple FIR filter using the Parks-McClellan algorithm
- Example - Design an equiripple FIR filter with the specifications: $F_p = 0.8$ kHz, $F_s = 1$ kHz, $F_T = 4$ kHz, $\alpha_p = 0.5$ dB, $\alpha_s = 40$ dB
- Here, $\delta_p = 0.0559$ and $\delta_s = 0.01$

Equiripple FIR Digital Filter Design Using MATLAB

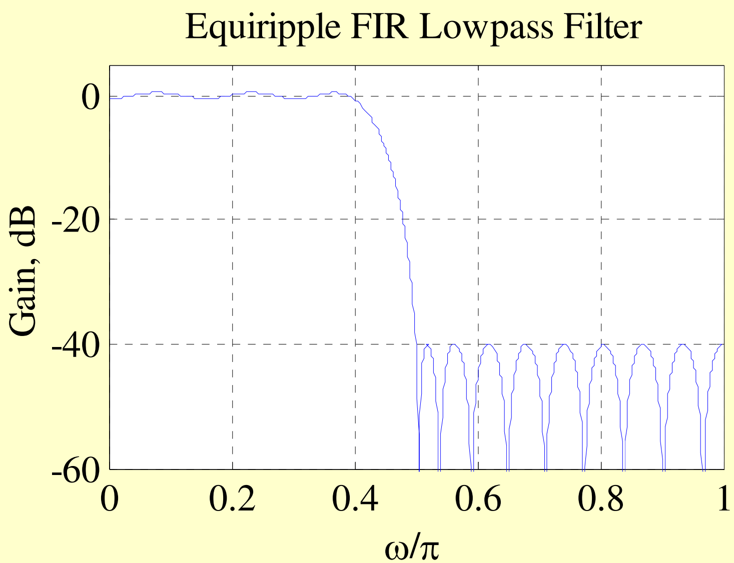
- MATLAB code fragments used are

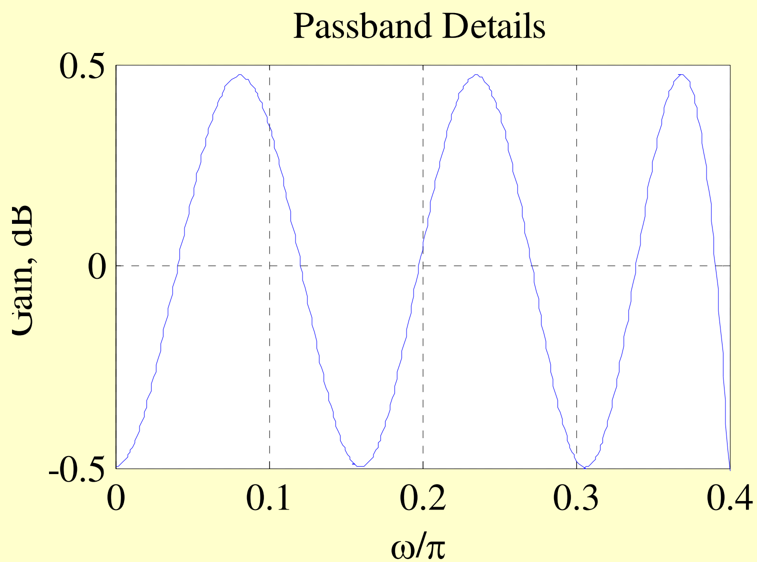
```
[N, fpts, mag, wt] =
firpmord(fedge, mval, dev, FT);
b = firpm(N, fpts, mag, wt);
```

where `fedge = [800   1000]`,
`mval = [1 0]`, `dev =[0.0559  0.01]`,
and `FT = 4000`

Equiripple FIR Digital Filter Design Using MATLAB

- The computed gain response with the filter order obtained ($N = 28$) does not meet the specifications ($\alpha_p = 0.6\,\text{dB}, \alpha_s = 38.7\,\text{dB}$)
- Specifications are met with $N = 30$





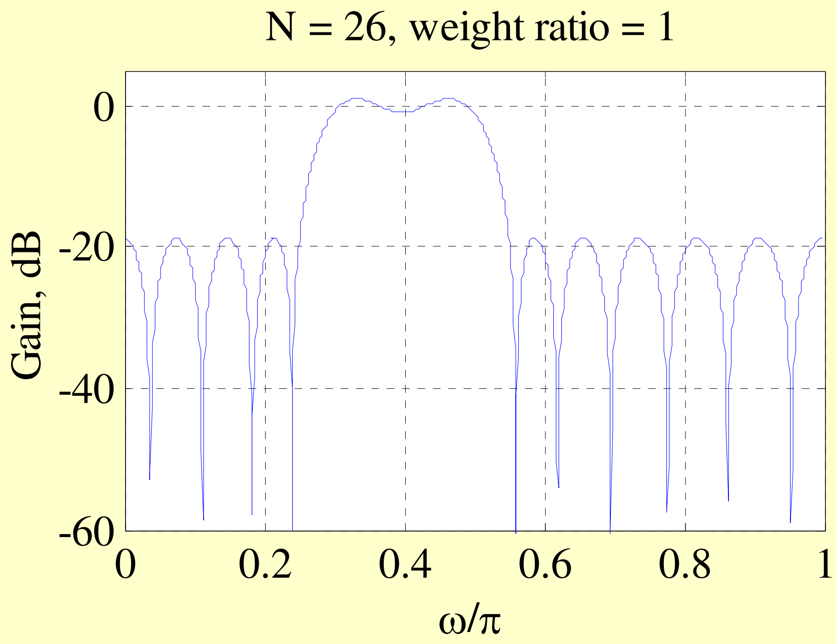
Equiripple FIR Digital Filter Design Using MATLAB

- Example - Design a linear-phase FIR bandpass filter of order 26 with a passband from 0.3 to 0.5, and stopbands from 0 to 0.25 and from 0.55 to 1
- The pertinent input data here are

```
N = 26
fpts = [0  0.25  0.3  0.5  0.55 1]
mag = [0  0  1  1  0  0]
wt = [1  1  1]
```

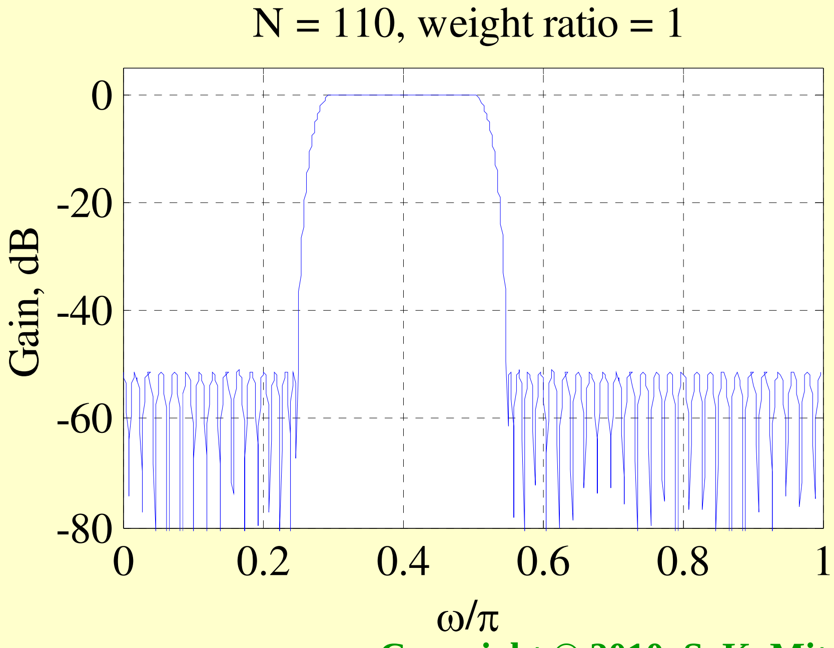
Equiripple FIR Digital Filter Design Using MATLAB

- Computed gain response shown below where $\alpha_p = 1$ dB, $\alpha_s = 18.7$ dB



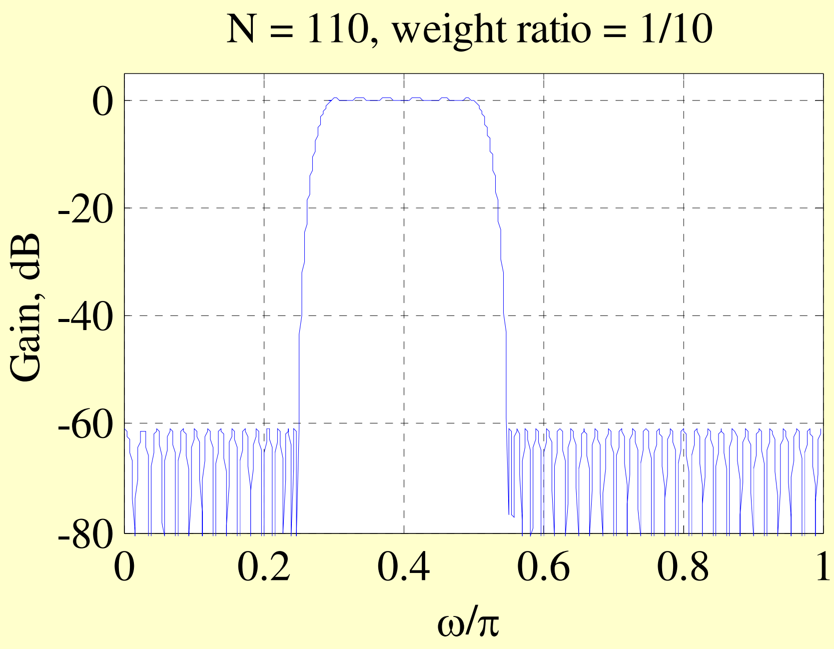
Equiripple FIR Digital Filter Design Using MATLAB

- We redesign the filter with order increased to 110
- Computed gain response shown below where $\alpha_p = 0.024$ dB, $\alpha_s = 51.2$ dB
- Note: Increase in order improves gain response at the expense of increased computational complexity



Equiripple FIR Digital Filter Design Using MATLAB

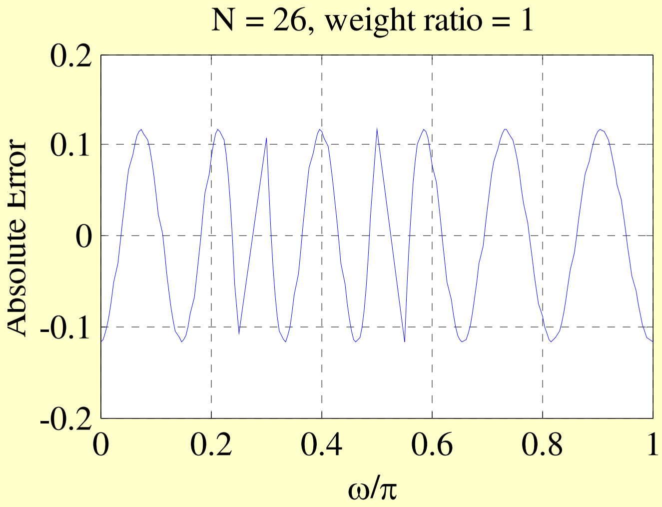
- α_s can be increased at the expenses of a larger α_p by decreasing the relative weight ratio $W(\omega) = \delta_p / \delta_s$
- Gain response of bandpass filter of order 110 obtained with a weight vector [1 0.1 1]



- Now $\alpha_p = 0.076$ dB, $\alpha_s = 60.86$ dB

Equiripple FIR Digital Filter Design Using MATLAB

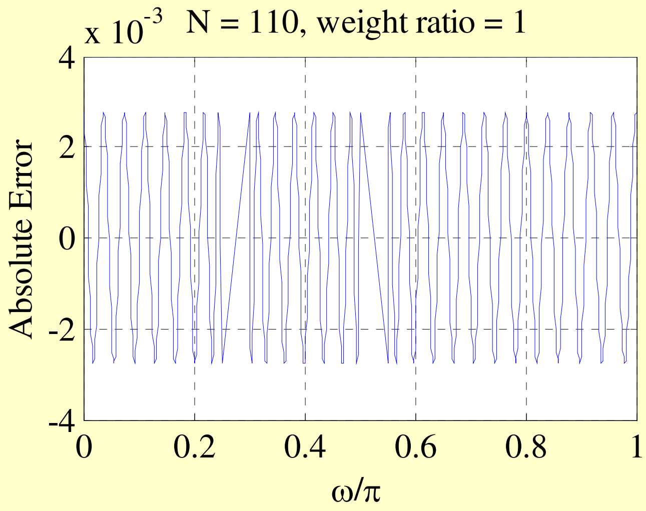
- Plots of absolute error for 1st design
- Absolute error has same peak value in all bands

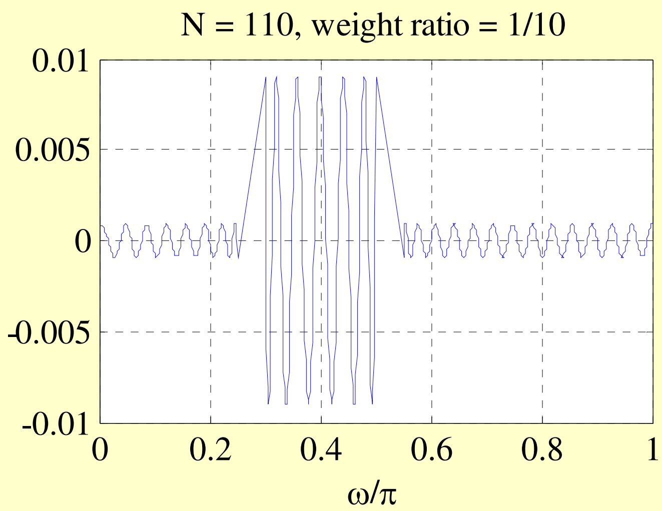


- As $L = 13$, and there are 4 band edges, there can be at most $L - 1 + 6 = 18$ extrema
- Error plot exhibits 17 extrema

Equiripple FIR Digital Filter Design Using MATLAB

- Absolute error has same peak value in all bands for the 2nd design
- Absolute error in passband of 3rd design is 10 times the error in the stopbands





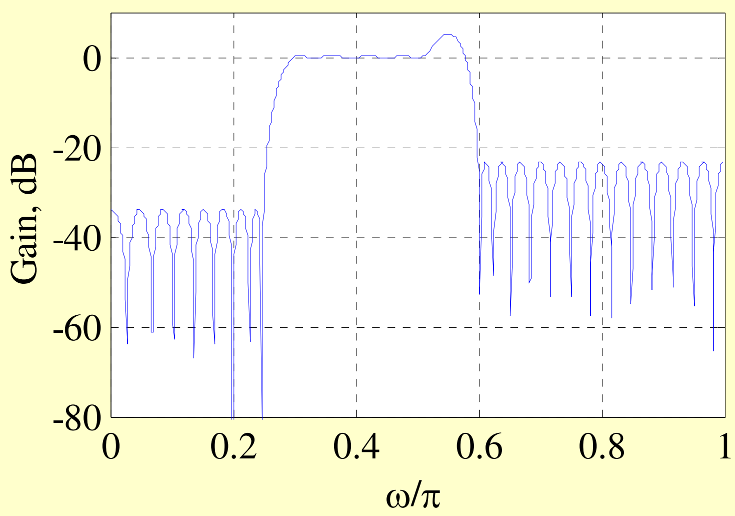
Equiripple FIR Digital Filter Design Using MATLAB

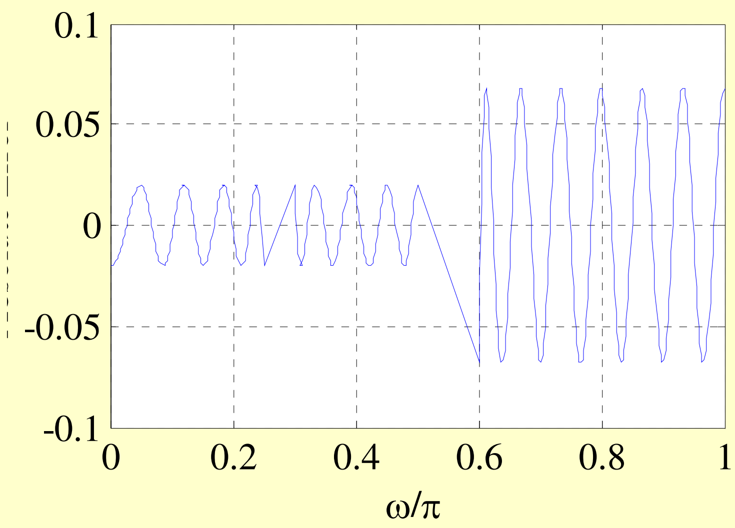
- Example - Design a linear-phase FIR bandpass filter of order 60 with a passband from 0.3 to 0.5, and stopbands from 0 to 0.25 and from 0.6 to 1 with unequal weights
- The pertinent input data here are

```
N = 60
fpts = [0  0.25  0.3  0.5  0.6 1]
mag = [0  0  1  1  0  0]
wt = [1  1  0.3]
```

Equiripple FIR Digital Filter Design Using MATLAB

- Plots of gain response and absolute error shown below



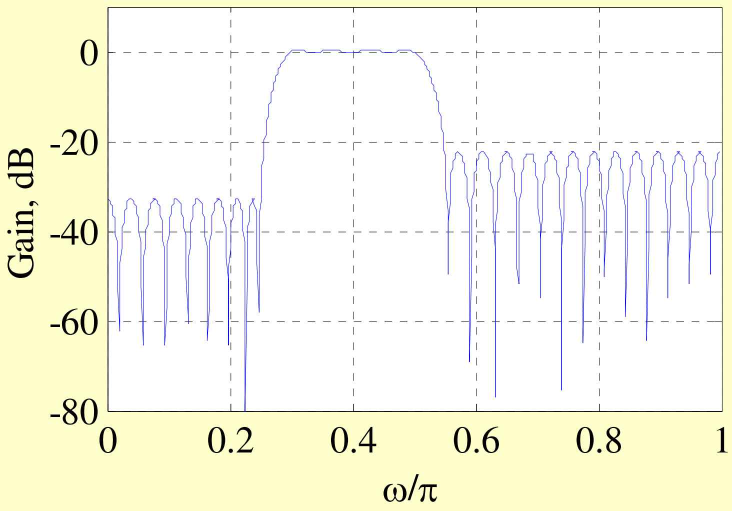


Equiripple FIR Digital Filter Design Using MATLAB

- Response in the second transition band shows a peak with a value higher than that in passband
- Result does not contradict alternation theorem
- As $N = 60$, $M = 30$, and hence, there must be at least $M + 2 = 32$ extremal frequencies
- Plot of absolute error shows the presence of 32 extremal frequencies

Equiripple FIR Digital Filter Design Using MATLAB

- If gain response of filter designed exhibits a nonmonotonic behavior, it is recommended that either the filter order or the bandedges or the weighting function be adjusted until a satisfactory gain response has been obtained
- Gain plot obtained by moving the second stopband edge to 0.55



Equiripple FIR Differentiator Design Using MATLAB

- A lowpass differentiator has a bandlimited frequency response

$$H_{DIF}(e^{j\omega}) = \begin{cases} j\omega, & 0 \le |\omega| \le \omega_p \\ 0, & \omega_s \le |\omega| \le \pi \end{cases}$$

where $0 \le |\omega| \le \omega_p$ represents the passband and $\omega_s \le |\omega| \le \pi$ represents the stopband

- For the design phase we choose

$$W(\omega) = 1/\omega, \quad D(\omega) = 1, \qquad 0 \le |\omega| \le \omega_p$$

Equiripple FIR Differentiator Design Using MATLAB

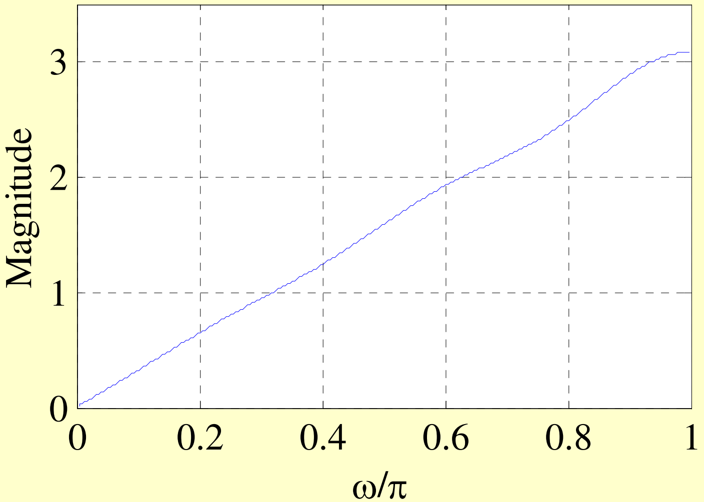
- The M-file `firpmord` cannot be used to estimate the order of an FIR differentiator
- Example - Design a full-band ($\omega_p = \pi$) differentiator of order 11
- Code fragment to use

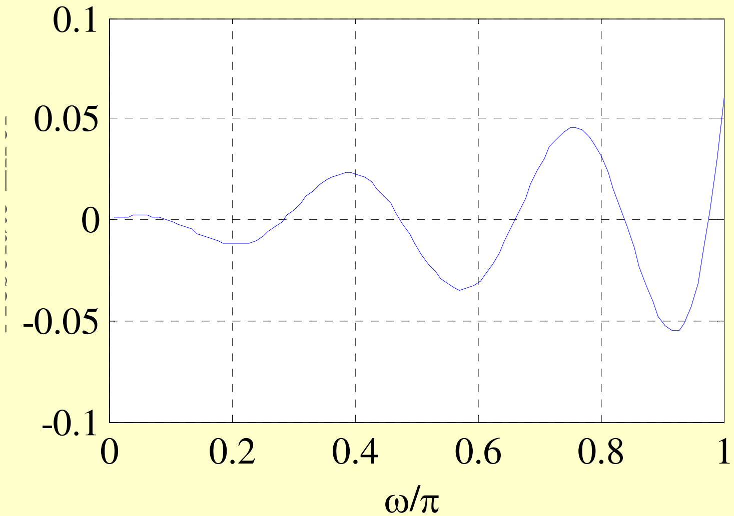
```
b =
firpm(N,fpts,mag,'differentiator');
```

where
```
N = 11
fpts = [0   1]
mag = [0   pi]
```

Equiripple FIR Differentiator Design Using MATLAB

- Plots of magnitude response and absolute error





- Absolute error increases with ω as the algorithm results in an equiripple error of the function $[\frac{A(\omega)}{\omega} - 1]$

Equiripple FIR Differentiator Design Using MATLAB

- Example - Design a lowpass differentiator of order 50 with $\omega_p = 0.4\pi$ and $\omega_s = 0.45\pi$
- Code fragment to use

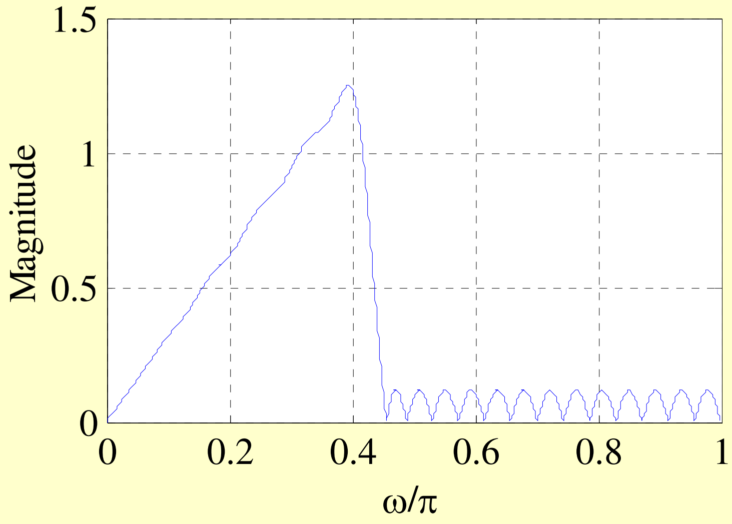
```
b =
firpm(N,fpts,mag'differentiator');
```

where

```
N = 50
fpts = [0    0.4    0.45    1]
mag = [0    0.4*pi    0    0]
```

Equiripple FIR Differentiator Design Using MATLAB

- Plot of the magnitude response of the lowpass differentiator



Equiripple FIR Hilbert Transformer Design Using MATLAB

- Desired amplitude response of a bandpass Hilbert transformer is

$$D(\omega) = 1, \quad \omega_L \le |\omega| \le \omega_H$$

with weighting function

$$W(\omega) = 1, \quad \omega_L \le |\omega| \le \omega_H$$

- Impulse response of an ideal Hilbert transformer satisfies the condition

$$h_{HT}[n] = 0, \qquad \text{for } n \text{ even}$$

which can be met by a Type 3 FIR filter

Equiripple FIR Hilbert Transformer Design Using MATLAB

- Example - Design a linear-phase bandpass FIR Hilbert transformer of order 20 with $\omega_L = 0.1\pi$, $\omega_H = 0.9\pi$
- Code fragment to use

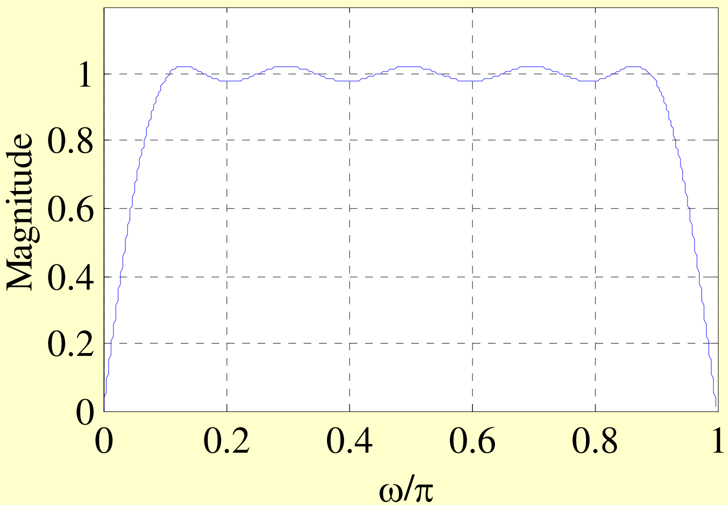
```
b = firpm(N,fpts,mag,'Hilbert');
```

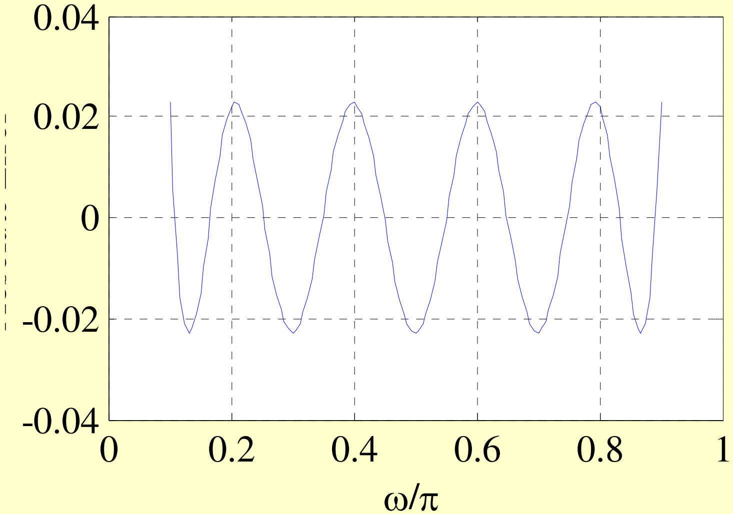
where

```
N = 20
fpts = [0.1   0.9]
mag = [1   1]
```

Equiripple FIR Hilbert Transformer Design Using MATLAB

- Plots of magnitude response and absolute error





Window-Based FIR Filter Design Using MATLAB

- Window Generation - Code fragments to use

  w = blackman(L);

  w = hamming(L);

  w = hanning(L);

  w = chebwin(L, Rs);

  w = kaiser(L, beta);

  where window length L is odd

Window-Based FIR Filter Design Using MATLAB

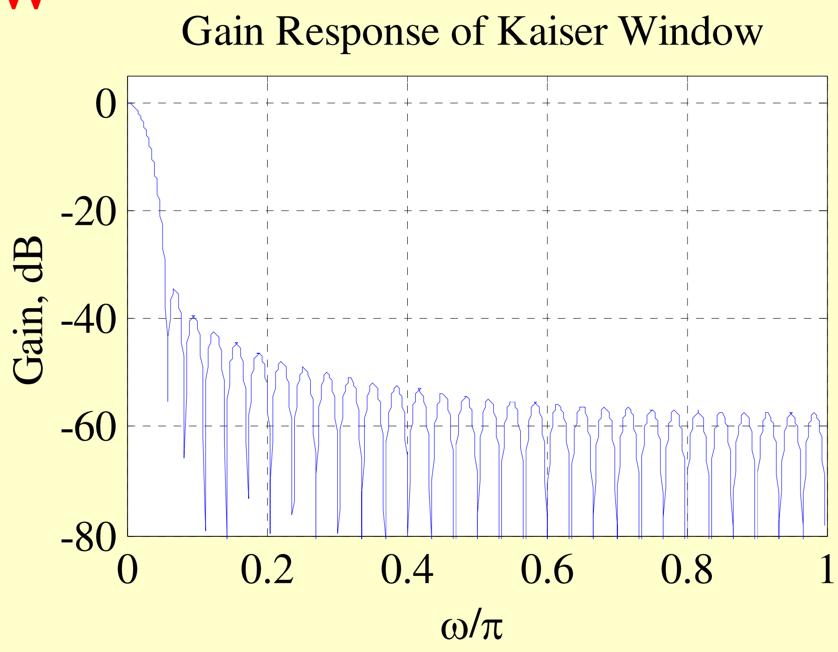
- Example - Kaiser window design for use in a lowpass FIR filter design
- Specifications of lowpass filter: $\omega_p = 0.3\pi$, $\omega_s = 0.4\pi$, $\alpha_s = 50$ dB $\Rightarrow \delta_s = 0.003162$
- Code fragments to use

```
[N, Wn, beta, ftype] = kaiserord(fpts, mag,dev);
w = kaiser(N+1, beta);
```

where
```
fpts = [0.3   0.4]
mag = [1   0]
dev = [0.003162   0.003162]
```

Window-Based FIR Filter Design Using MATLAB

- Plot of the gain response of the Kaiser window



Window-Based FIR Filter Design Using MATLAB

- M-files available are fir1 and fir2
- fir1 is used to design conventional lowpass, highpass, bandpass, bandstop and multiband FIR filters
- fir2 is used to design FIR filters with arbitrarily shaped magnitude response
- In fir1, Hamming window is used as a default if no window is specified

Window-Based FIR Filter Design Using MATLAB

- Example - Design using a Kaiser window a lowpass FIR filter with the specifications:
  $\omega_p = 0.3\pi$, $\omega_s = 0.4\pi$, $\delta_s = 0.003162$
- Code fragments to use

```
[N, Wn, beta, ftype] = kaiserord(fpts, mag, dev);
b = fir1(N, Wn, kaiser(N+1, beta));
```

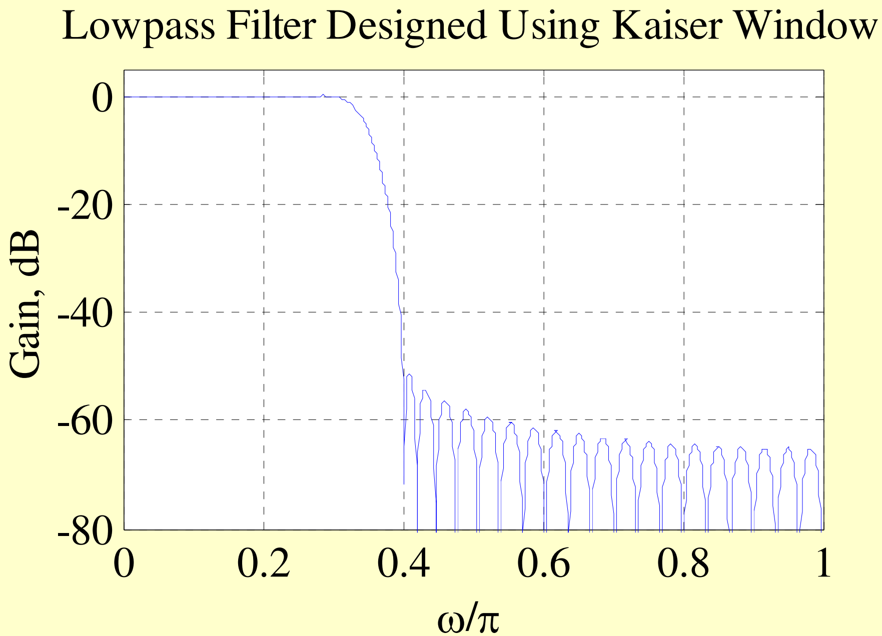
where fpts = [0.3   0.4]

mag = [1   0]

dev = [0.003162   0.003162]

Window-Based FIR Filter Design Using MATLAB

- Plot of gain response



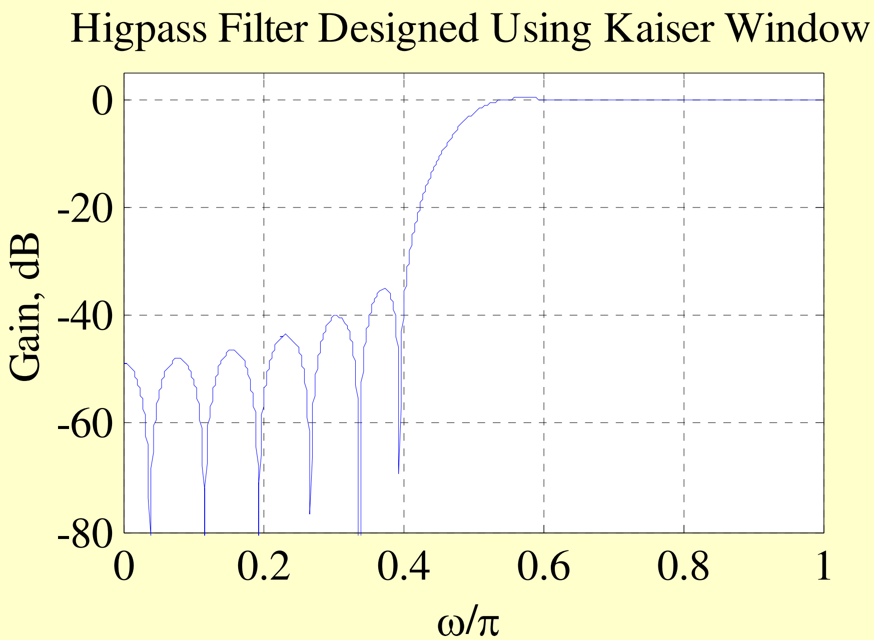
Window-Based FIR Filter Design Using MATLAB

- Example - Design using a Kaiser window a highpass FIR filter with the specifications:
  $\omega_p = 0.55\pi$, $\omega_s = 0.4\pi$, $\delta_s = 0.02$
- Code fragments to use
- [N, Wn, beta, ftype] = kaiserord(fpts, mag, dev);
  b = fir1(N, Wn, 'ftype', kaiser(N+1, beta));

  where fpts = [0.4 0.55]
  mag = [0 1]
  dev = [0.02 0.02]

Window-Based FIR Filter Design Using MATLAB

- Plot of gain response



Window-Based FIR Filter Design Using MATLAB

- Example - Design using a Hamming window an FIR filter of order 100 with three different constant magnitude levels: 0.3 in the frequency range [0, 0.28], 1.0 in the frequency range [0.3, 0.5], and 0.7 in the frequency range [0.52, 1.0]

Window-Based FIR Filter Design Using MATLAB

- Code fragment to use

```
b = fir2(100, fpts, mval);
```

where

```
fpts = [0  0.28  0.3  0.5  0.52  1];
mval = [0.3  0.3  1.0  1.0  0.7  0.7];
```