Computer-Aided Design of Digital Filters

- The IIR and FIR filter design techniques discussed so far can be easily implemented on a computer
- In addition, there are a number of filter design algorithms that rely on some type of optimization techniques that are used to minimize the error between the desired frequency response and that of the computer-generated filter

Computer-Aided Design of Digital Filters

- Basic idea behind the computer-based iterative technique
- Let $H(e^{j\omega})$ denote the frequency response of the digital filter $H(z)$ to be designed approximating the desired frequency response $D(e^{j\omega})$, given as a piecewise linear function of ω, in some sense

Computer-Aided Design of Digital Filters

- Objective - Determine iteratively the coefficients of $H(z)$ so that the difference between between $H(e^{j\omega})$ and $D(e^{j\omega})$ over closed subintervals of $0 \le \omega \le \pi$ is minimized
- This difference usually specified as a weighted error function

$$\mathcal{E}(\omega) = W(e^{j\omega})[H(e^{j\omega}) - D(e^{j\omega})]$$

where $W(e^{j\omega})$ is some user-specified weighting function

Computer-Aided Design of Digital Filters

- **Chebyshev** or **minimax criterion** - Minimizes the peak absolute value of the weighted error:

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

where R is the set of disjoint frequency bands in the range $0 \le \omega \le \pi$, on which $D(e^{j\omega})$ is defined
- For example, for a lowpass filter design, R is the disjoint union of $[0, \omega_p]$ and $[\omega_s, \pi]$

Computer-Aided Design of Digital Filters

- **Least-*p* Criterion** - Minimize

$$\varepsilon = \int_{\omega \in R} \left| W(e^{j\omega})[H(e^{j\omega}) - D(e^{j\omega})] \right|^p d\omega$$

over the specified frequency range R with p a positive integer
- $p = 2$ yields the **least-squares criterion**
- As $p \to \infty$, the least p-th solution approaches the minimax solution

Computer-Aided Design of Digital Filters

- **Least-*p* Criterion** - In practice, the p-th power error measure is approximated as

$$\varepsilon = \sum_{i=1}^{K} \{W(e^{j\omega_i})[H(e^{j\omega_i}) - D(e^{j\omega_i})]\}^p$$

where ω_i, $1 \le i \le K$, is a suitably chosen dense grid of digital angular frequencies
- For linear-phase FIR filter design, $H(e^{j\omega})$ and $D(e^{j\omega})$ are zero-phase frequency responses
- For IIR filter design, $H(e^{j\omega})$ and $D(e^{j\omega})$ are magnitude functions

Design of Equiripple Linear-Phase FIR Filters

- The linear-phase FIR filter obtained by minimizing the peak absolute value of

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

is usually called the **equiripple FIR filter**

- After ε is minimized, the weighted error function $\mathcal{E}(\omega)$ exhibits an equiripple behavior in the frequency range R

Design of Equiripple Linear-Phase FIR Filters

- The general form of frequency response of a causal linear-phase FIR filter of length $2M+1$:

$$H(e^{j\omega}) = e^{-jM\omega} e^{j\beta} \breve{H}(\omega)$$

where the amplitude response $\breve{H}(\omega)$ is a real function of ω

- Weighted error function is given by

$$\mathcal{E}(\omega) = W(\omega)[\breve{H}(\omega) - D(\omega)]$$

where $D(\omega)$ is the desired amplitude response and $W(\omega)$ is a positive weighting function

Design of Equiripple Linear-Phase FIR Filters

- **Parks-McClellan Algorithm** - Based on iteratively adjusting the coefficients of $\breve{H}(\omega)$ until the peak absolute value of $\mathcal{E}(\omega)$ is minimized
- If peak absolute value of $\mathcal{E}(\omega)$ in a band $\omega_a \le \omega \le \omega_b$ is ε_o, then the absolute error satisfies

$$|\breve{H}(\omega) - D(\omega)| \le \frac{\varepsilon_o}{|W(\omega)|}, \quad \omega_a \le \omega \le \omega_b$$

Design of Equiripple Linear-Phase FIR Filters

- For filter design,

$$D(\omega) = \begin{cases} 1, & \text{in the passband} \\ 0, & \text{in the stopband} \end{cases}$$

- $\breve{H}(\omega)$ is required to satisfy the above desired response with a ripple of $\pm\delta_p$ in the passband and a ripple of δ_s in the stopband

Design of Equiripple Linear-Phase FIR Filters

- Thus, weighting function can be chosen either as

$$W(\omega) = \begin{cases} 1, & \text{in the passband} \\ \delta_p / \delta_s, & \text{in the stopband} \end{cases}$$

or

$$W(\omega) = \begin{cases} \delta_s / \delta_p, & \text{in the passband} \\ 1, & \text{in the stopband} \end{cases}$$

Design of Equiripple Linear-Phase FIR Filters

- **Type 1 FIR Filter** - $\breve{H}(\omega) = \sum_{k=0}^{M} a[k]\cos(\omega k)$

where

$$a[0] = h[M],\ a[k] = 2h[M-k],\ 1 \le k \le M$$

- **Type 2 FIR filter** -

$$\breve{H}(\omega) = \sum_{k=1}^{(2M+1)/2} b[k]\cos\left(\omega(k - \tfrac{1}{2})\right)$$

where

$$b[k] = 2h[\tfrac{2M+1}{2} - k],\ 1 \le k \le \tfrac{2M+1}{2}$$

Design of Equiripple Linear-Phase FIR Filters

- **Type 3 FIR Filter** - $\breve{H}(\omega) = \sum_{k=1}^{M} c[k]\sin(\omega k)$

  where

  $$c[k] = 2h[M-k], \;\; 1 \le k \le M$$

- **Type 4 FIR Filter** -

  $$\breve{H}(\omega) = \sum_{k=1}^{(2M+1)/2} d[k]\sin\left(\omega(k-\tfrac{1}{2})\right)$$

  where

  $$d[k] = 2h[\tfrac{2M+1}{2}-k], \quad 1 \le k \le \tfrac{2M+1}{2}$$

Design of Equiripple Linear-Phase FIR Filters

- Amplitude response for all 4 types of linear-phase FIR filters can be expressed as

  $$\breve{H}(\omega) = Q(\omega)A(\omega)$$

  where

  $$Q(\omega) = \begin{cases} 1, & \text{for Type 1} \\ \cos(\omega/2), & \text{for Type 2} \\ \sin(\omega), & \text{for Type 3} \\ \sin(\omega/2), & \text{for Type 4} \end{cases}$$

Design of Equiripple Linear-Phase FIR Filters

and

$$A(\omega) = \sum_{k=0}^{L} \tilde{a}[k]\cos(\omega k)$$

where

$$\tilde{a}[k] = \begin{cases} a[k], & \text{for Type 1} \\ \tilde{b}[k], & \text{for Type 2} \\ \tilde{c}[k], & \text{for Type 3} \\ \tilde{d}[k], & \text{for Type 4} \end{cases}$$

Design of Equiripple Linear-Phase FIR Filters

with

$$L = \begin{cases} M, & \text{for Type 1} \\ \frac{2M-1}{2}, & \text{for Type 2} \\ M-1, & \text{for Type 3} \\ \frac{2M-1}{2}, & \text{for Type 4} \end{cases}$$

$\tilde{b}[k]$, $\tilde{c}[k]$, and $\tilde{d}[k]$, are related to $b[k]$, $c[k]$, and $d[k]$, respectively

Design of Equiripple Linear-Phase FIR Filters

- Modified form of weighted error function

  $$\mathcal{E}(\omega) = W(\omega)[Q(\omega)A(\omega) - D(\omega)]$$
  $$= W(\omega)Q(\omega)[A(\omega) - \tfrac{D(\omega)}{Q(\omega)}]$$
  $$= \tilde{W}(\omega)[A(\omega) - \tilde{D}(\omega)]$$

  where we have used the notation

  $$\tilde{W}(\omega) = W(\omega)Q(\omega)$$
  $$\tilde{D}(\omega) = D(\omega)/Q(\omega)$$

Design of Equiripple Linear-Phase FIR Filters

- Optimization Problem - Determine $\tilde{a}[k]$ which minimize the peak absolute value ε of

  $$\mathcal{E}(\omega) = \tilde{W}(\omega)[\sum_{k=0}^{L} \tilde{a}[k]\cos(\omega k) - \tilde{D}(\omega)]$$

  over the specified frequency bands $\omega \in R$

- After $\tilde{a}[k]$ has been determined, corresponding coefficients of the original $A(e^{j\omega})$ are computed from which $h[n]$ are determined

Design of Equiripple Linear-Phase FIR Filters

- Alternation Theorem - $A(\omega)$ is the best unique approximation of $D(\omega)$ obtained by minimizing peak absolute value ε of
$$\mathcal{E}(\omega) = W(\omega)[Q(\omega)A(\omega) - D(\omega)]$$
if and only if there exist at least L+2 extremal frequencies, $\{\omega_i\}$, $0 \le i \le L+1$, in a closed subset R of the frequency range $0 \le \omega \le \pi$ such that $\omega_0 < \omega_1 < \cdots < \omega_L < \omega_{L+1}$ and $\mathcal{E}(\omega_i) = -\mathcal{E}(\omega_{i+1})$, $|\mathcal{E}(\omega_i)| = \varepsilon$ for all i

Design of Equiripple Linear-Phase FIR Filters

- Consider a Type 1 FIR filter with an amplitude response $A(\omega)$ whose approximation error $\mathcal{E}(\omega)$ satisfies the Alternation Theorem
- Peaks of $\mathcal{E}(\omega)$ are at $\omega = \omega_i$, $0 \le i \le L+1$ where $d\mathcal{E}(\omega)/d\omega = 0$
- Since in the passband and stopband, $\widetilde{W}(\omega)$ and $\widetilde{D}(\omega)$ are piecewise constant,
$$\frac{d\mathcal{E}(\omega)}{d\omega} = \frac{dA(\omega)}{d\omega} = 0$$

Design of Equiripple Linear-Phase FIR Filters

- Using $\cos(\omega k) = T_k(\cos\omega)$, where $T_k(x)$ is the k-th order Chebyshev polynomial
$$T_k(x) = \cos(k\cos^{-1} x)$$
- $A(\omega)$ can be expressed as
$$A(\omega) = \textstyle\sum_{k=0}^{L} \alpha[k](\cos\omega)^k$$
which is an Lth-order polynomial in $\cos\omega$
- Hence, $A(\omega)$ can have at most $L-1$ local minima and maxima inside specified passband and stopband

Design of Equiripple Linear-Phase FIR Filters

- At bandedges, $\omega = \omega_p$ and $\omega = \omega_s$, $|\mathcal{E}(\omega)|$ is a maximum, and hence $A(\omega)$ has extrema at these points
- $A(\omega)$ can have extrema at $\omega = 0$ and $\omega = \pi$
- Therefore, there are at most L+3 extremal frequencies of $\mathcal{E}(\omega)$
- For linear-phase FIR filters with K specified bandedges, there can be at most L+K+1 extremal frequencies

Design of Equiripple Linear-Phase FIR Filters

- The set of equations
$$\widetilde{W}(\omega_i)[A(\omega_i) - \widetilde{D}(\omega_i)] = (-1)^i \varepsilon, \quad 0 \le i \le L+1$$
is written in a matrix form
$$\begin{bmatrix} 1 & \cos(\omega_0) & \cdots & \cos(L\omega_0) & -1/\widetilde{W}(\omega_0) \\ 1 & \cos(\omega_1) & \cdots & \cos(L\omega_1) & 1/\widetilde{W}(\omega_1) \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 1 & \cos(\omega_L) & \cdots & \cos(L\omega_L) & (-1)^{L-1}/\widetilde{W}(\omega_L) \\ 1 & \cos(\omega_{L+1}) & \cdots & \cos(L\omega_{L+1}) & (-1)^{L}/\widetilde{W}(\omega_{L+1}) \end{bmatrix} \begin{bmatrix} \widetilde{a}[0] \\ \widetilde{a}[1] \\ \vdots \\ \widetilde{a}[L] \\ \varepsilon \end{bmatrix} = \begin{bmatrix} \widetilde{D}(\omega_0) \\ \widetilde{D}(\omega_1) \\ \vdots \\ \widetilde{D}(\omega_L) \\ \widetilde{D}(\omega_{L+1}) \end{bmatrix}$$

Design of Equiripple Linear-Phase FIR Filters

- The matrix equation can be solved for the unknowns $\widetilde{a}[i]$ and ε if the locations of the L+2 extremal frequencies are known a priori
- The **Remez exchange algorithm** is used to determine the locations of the extremal frequencies

Remez Exchange Algorithm

- Step 1: A set of initial values of extremal frequencies are either chosen or are available from completion of previous stage
- Step 2: Value of ε is computed using

$$\varepsilon = \frac{c_0\tilde{D}(\omega_0)+c_1\tilde{D}(\omega_1)+\cdots+c_{L+1}\tilde{D}(\omega_{L+1})}{\dfrac{c_0}{\tilde{W}(\omega_0)}-\dfrac{c_1}{\tilde{W}(\omega_1)}+\cdots+\dfrac{(-1)^{L+1}c_{L+1}}{\tilde{W}(\omega_{L+1})}}$$

where $c_n = \prod_{\substack{i=0\\ i\neq n}}^{L+1} \dfrac{1}{\cos(\omega_n)-\cos(\omega_i)}$

Remez Exchange Algorithm

- Step 3: Values of $A(\omega)$ at $\omega=\omega_i$ are then computed using

$$A(\omega_i)=\frac{(-1)^i\varepsilon}{\tilde{W}(\omega_i)}+\tilde{D}(\omega_i), \quad 0\le i\le L+1$$

- Step 4: The polynomial $A(\omega)$ is determined by interpolating the above values at the L+2 extremal frequencies using the Lagrange interpolation formula

Remez Exchange Algorithm

- Step 4: The new error function

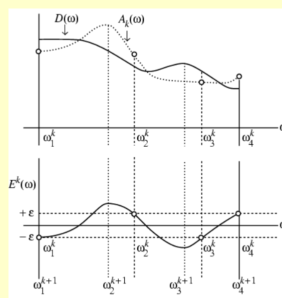
$$\mathcal{E}(\omega)=\tilde{W}(\omega)[A(\omega)-\tilde{D}(\omega)]$$

is computed at a dense set S ($S \ge L$) of frequencies. In practice $S = 16L$ is adequate. Determine the L+2 new extremal frequencies from the values of $\mathcal{E}(\omega)$ evaluated at the dense set of frequencies.

- Step 5: If the peak values ε of $\mathcal{E}(\omega)$ are equal in magnitude, algorithm has converged. Otherwise, go back to Step 2.

Remez Exchange Algorithm

- Illustration of algorithm



Iteration process is stopped if the difference between the values of the peak absolute errors between two consecutive stages is less than a preset value, e.g., 10^{-6}

Remez Exchange Algorithm

- Example - Approximate the desired function $D(x)=1.1x^2-0.1$ defined for the range $0\le x\le 2$ by a linear function a_1x+a_0 by minimizing the peak value of the absolute error

$$\max_{x\in[0,2]} \left|1.1x^2-0.1-a_0-a_1x\right|$$

- Stage 1:

Choose arbitrarily the initial extremal points

$$x_1=0,\ x_2=0.5,\ x_3=1.5$$

Remez Exchange Algorithm

- Solve the three linear equations

$$a_0+a_1x_\ell-(-1)^\ell\varepsilon=D(x_\ell), \quad \ell=1,2,3$$

i.e.,

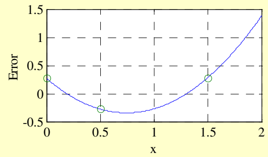
$$\begin{bmatrix}1 & 0 & 1\\ 1 & 0.5 & -1\\ 1 & 1.5 & 1\end{bmatrix}\begin{bmatrix}a_0\\ a_1\\ \varepsilon\end{bmatrix}=\begin{bmatrix}-0.1\\ 0.175\\ 2.375\end{bmatrix}$$

for the given extremal points yielding

$$a_0=-0.375,\ a_1=1.65,\ \varepsilon=0.275$$

Remez Exchange Algorithm

- Plot of $\mathcal{E}_1(x) = 1.1x^2 - 1.65x + 0.275$ along with values of error at chosen extremal points shown below



- Note: Errors are equal in magnitude and alternate in sign

Remez Exchange Algorithm

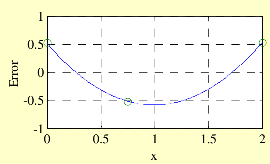
- Stage 2:
- Choose extremal points where $\mathcal{E}_1(x)$ assumes its maximum absolute values
- These are $x_1 = 0,\ x_2 = 0.75,\ x_3 = 2$
- New values of unknowns are obtained by solving

$$\begin{bmatrix} 1 & 0 & 1 \\ 1 & 0.75 & -1 \\ 1 & 2 & 1 \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ \varepsilon \end{bmatrix} = \begin{bmatrix} -0.1 \\ 0.5188 \\ 4.3 \end{bmatrix}$$

yielding $a_0 = -0.6156,\ a_1 = 2.2,\ \varepsilon = 0.5156$

Remez Exchange Algorithm

- Plot of $\mathcal{E}_2(x) = 1.1x^2 - 2.2x + 0.5156$ along with values of error at chosen extremal points shown below



Remez Exchange Algorithm

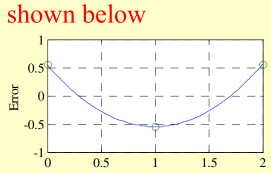
- Stage 3:
- Choose extremal points where $\mathcal{E}_2(x)$ assumes its maximum absolute values
- These are $x_1 = 0,\ x_2 = 1,\ x_3 = 2$
- New values of unknowns are obtained by solving

$$\begin{bmatrix} 1 & 0 & 1 \\ 1 & 1 & -1 \\ 1 & 2 & 1 \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ \varepsilon \end{bmatrix} = \begin{bmatrix} -0.1 \\ 1.0 \\ 4.3 \end{bmatrix}$$

yielding $a_0 = -0.65,\ a_1 = 2.2,\ \varepsilon = 0.55$

Remez Exchange Algorithm

- Plot of $\mathcal{E}_3(x) = 1.1x^2 - 2.2x + 0.55$ along with values of error at chosen extremal points shown below



- Algorithm has converged as ε is also the maximum value of the absolute error

IIR Digital Filter Design Using MATLAB

- Order Estimation -
- For IIR filter design using bilinear transformation, MATLAB statements to determine the order and bandedge are:

[N, Wn] = buttord(Wp, Ws, Rp, Rs);
[N, Wn] = cheb1ord(Wp, Ws, Rp, Rs);
[N, Wn] = cheb2ord(Wp, Ws, Rp, Rs);
[N, Wn] = ellipord(Wp, Ws, Rp, Rs);

IIR Digital Filter Design Using MATLAB

- Example - Determine the minimum order of a Type 2 Chebyshev digital highpass filter with the following specifications:

  $F_p = 1$ kHz, $F_p = 1$ kHz, $F_T = 4$ kHz,

  $\alpha_p = 1$ dB, $\alpha_s = 40$ dB
- Here, $\mathrm{Wp} = 2 \times 1/4 = 0.5$, $\mathrm{Ws} = 2 \times 0.6/4 = 0.3$
- Using the statement

  [N, Wn] = cheb2ord(0.5, 0.3, 1, 40);

  we get N = 5 and Wn = 0.3224

IIR Digital Filter Design Using MATLAB

- Filter Design -
- For IIR filter design using bilinear transformation, MATLAB statements to use are:

  [b, a] = butter(N, Wn)

  [b, a] = cheby1(N, Rp, Wn)

  [b, a] = cheby2(N, Rs, Wn)

  [b, a] = ellip(N, Rp, Rs, Wn)

IIR Digital Filter Design Using MATLAB

- The form of transfer function obtained is

$$G(z) = \frac{B(z)}{A(z)} = \frac{b(1) + b(2)z^{-1} + \cdots + b(N+1)z^{-N}}{1 + a(2)z^{-1} + \cdots + a(N+1)z^{-N}}$$

- The frequency response can be computed using the M-file freqz(b, a, w) where w is a set of specified angular frequencies
- It generates a set of complex frequency response samples from which magnitude and/or phase response samples cn be computed

IIR Digital Filter Design Using MATLAB

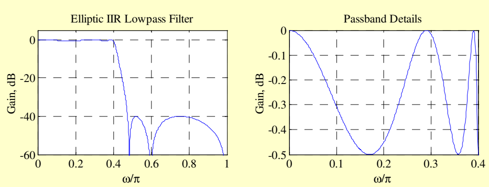
- Example - Design an elliptic IIR lowpass filter with the specifications: $F_p = 0.8$ kHz, $F_s = 1$ kHz, $F_T = 4$ kHz, $\alpha_p = 0.5$ dB, $\alpha_s = 40$ dB
- Here, $\omega_p = 2\pi F_p / F_T = 0.4\pi$, $\omega_s = 2\pi F_s / F_T = 0.5\pi$
- Code fragments used are:

  [N,Wn] = ellipord(0.4, 0.5, 0.5, 40);

  [b, a] = ellip(N, 0.5, 40, Wn);

IIR Digital Filter Design Using MATLAB

- Gain response plot is shown below:



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 remezord 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 estimted 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 remez can be used to design an equiripple FIR filter using the Parks-McClellan algorithm $\delta_s = 0.01$
- 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] =
    remezord(fedge, mval, dev, FT);
b = remez(N, fpts, mag, wt);
```

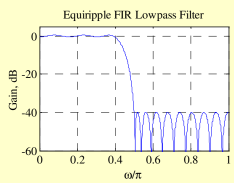
where fedge = [800   1000],

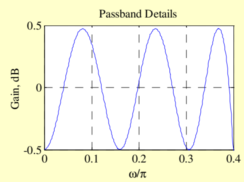
mval = [1   0], dev = [0.0559   0.01], and

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 ($\alpha_p = 0.6$ dB, $\alpha_s = 38.7$ 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

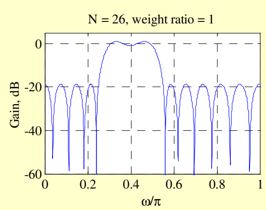
  fpts = [0 0.25 0.3 0.5 0.55 1]

  mag = [0 0 1 1 0 0]

  wt = [1 1 1]

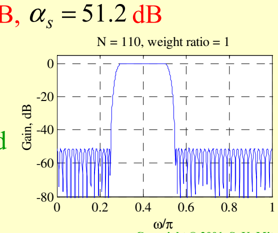
Equiripple FIR Digital Filter Design Using MATLAB

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



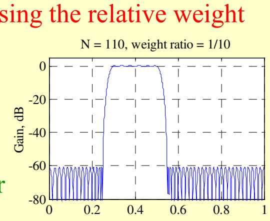
Equiripple FIR Digital Filter Design Using MATLAB

- 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

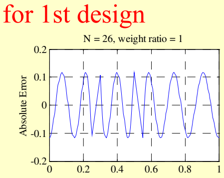
- α_s can be increased at the expenses of a larger α_p by decreasing the relative weight ratio $W(\omega) = \delta_p / \delta_s$
- Gain response of bandpass filter of order 110 obtained with a weight vector [1 0.1 1]



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

Equiripple FIR Digital Filter Design Using MATLAB

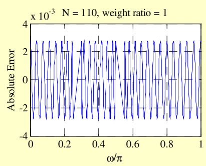
- Plots of absolute error for 1st design
- Absolute error has same peak value in all bands

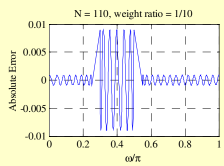


- As $L = 13$, and there are 4 band edges, there can be at most $L - 1 + 6 = 18$ extrema
- Error plot exhibits 17 extrema

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

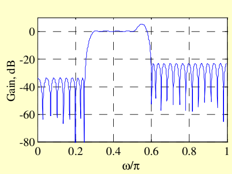
  fpts = [0 0.25 0.3 0.5 0.6 1]

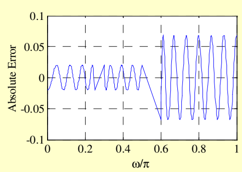
  mag = [0 0 1 1 0 0]

  wt = [1 1 0.3]

Equiripple FIR Digital Filter Design Using MATLAB

- Plots of gain response and absolute error shown below



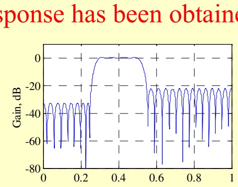


Equiripple FIR Digital Filter Design Using MATLAB

- 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

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

Equiripple FIR Differentiator Design Using MATLAB

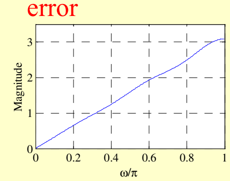
- The M-file remezord cannot be used to estimate the order of an FIR differentiator
- Example - Design a full-band ($\omega_p = \pi$) differentiator of order 11
- Code fragment to use

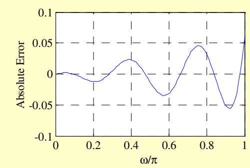
b = remez(N, fpts, mag, 'differentiator');

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

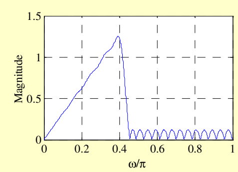
b = remez(N, fpts, mag, 'differentiator');

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

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

- Impulse response of an ideal Hilbert transformer satisfies the condition

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

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

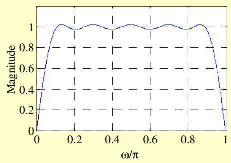
b = remez(N, fpts, mag, 'Hilbert');

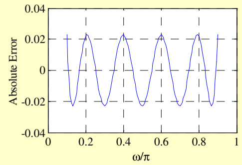
where

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);

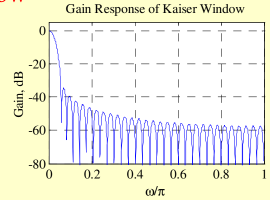
where fpts = [0.3 0.4]

mag = [0 1]

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));

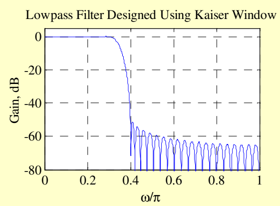
where fpts = [0.3 0.4]

mag = [1 0]

dev = [0.003162 0.003162]

Window-Based FIR Filter Design Using MATLAB

- Plot of gain response



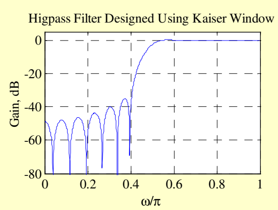
Window-Based FIR Filter Design Using MATLAB

- 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];