

ESE 531: Digital Signal Processing

Lec 17: March 22, 2018

Optimal Filter Design



Optimal Filter Design

- ❑ Window method
 - Design Filters heuristically using windowed sinc functions
 - Choose order and window type
 - Check DTFT to see if filter specs are met
- ❑ Optimal design
 - Design a filter $h[n]$ with $H(e^{j\omega})$
 - Approximate $H_d(e^{j\omega})$ with some optimality criteria - or satisfies specs.

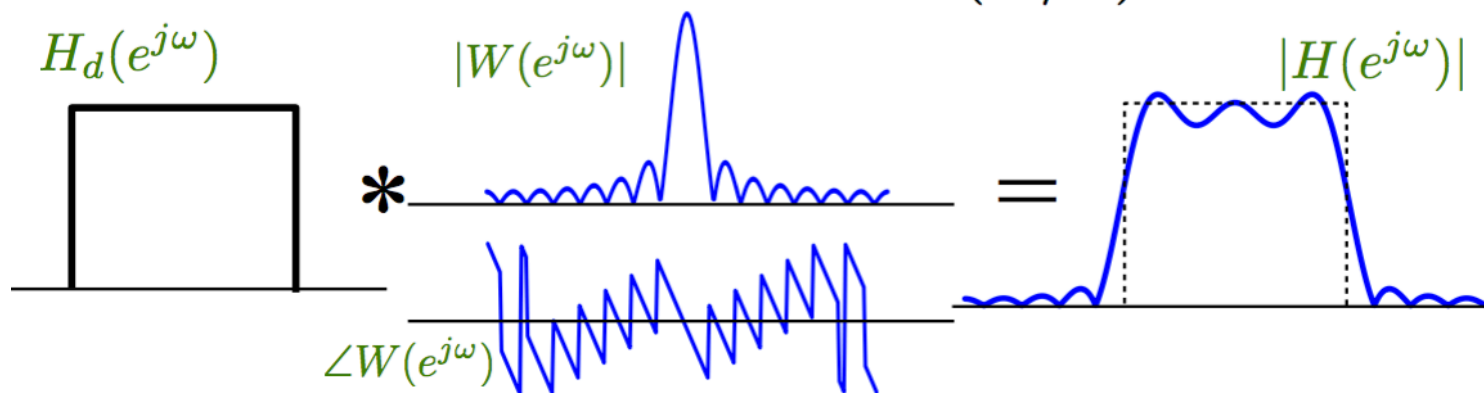
FIR Design by Windowing

- We already saw this before,

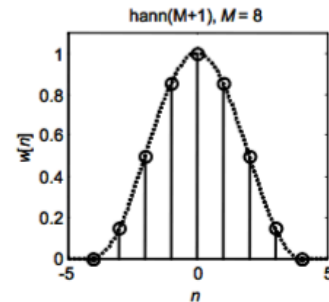
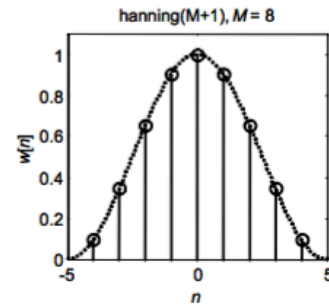
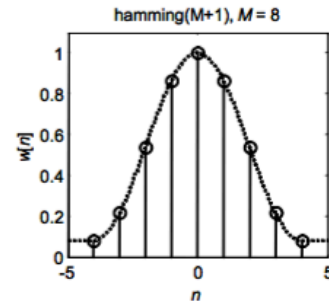
$$H(e^{j\omega}) = H_d(e^{j\omega}) * W(e^{j\omega})$$

- For Boxcar (rectangular) window

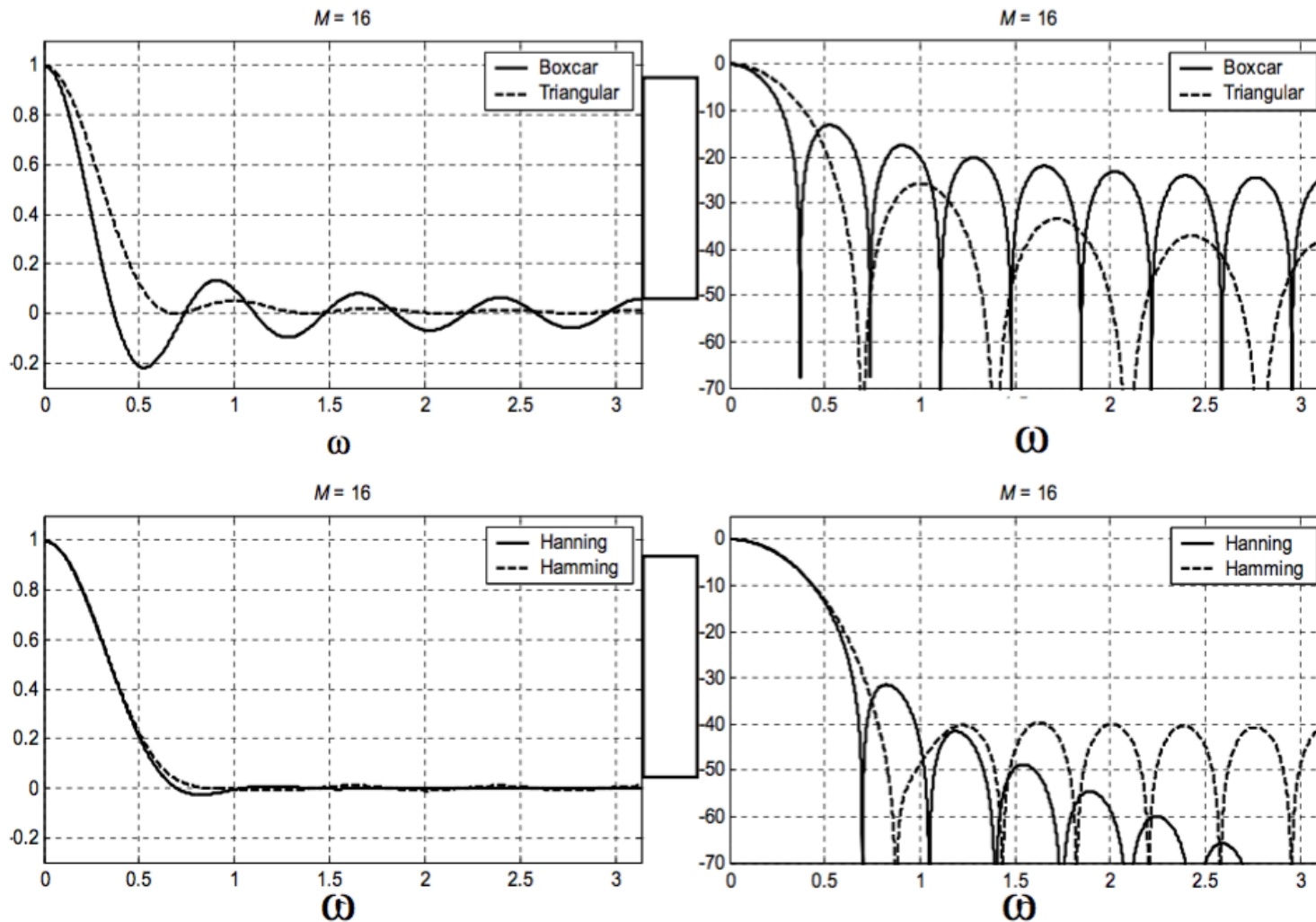
$$W(e^{j\omega}) = e^{-j\omega \frac{M}{2}} \frac{\sin(\omega(M+1)/2)}{\sin(\omega/2)}$$



Tapered Windows

Name(s)	Definition	MATLAB Command	Graph ($M = 8$)
Hann	$w[n] = \begin{cases} \frac{1}{2} \left[1 + \cos\left(\frac{\pi n}{M/2}\right) \right] & n \leq M/2 \\ 0 & n > M/2 \end{cases}$	<code>hann(M+1)</code>	
Hanning	$w[n] = \begin{cases} \frac{1}{2} \left[1 + \cos\left(\frac{\pi n}{M/2 + 1}\right) \right] & n \leq M/2 \\ 0 & n > M/2 \end{cases}$	<code>hanning(M+1)</code>	
Hamming	$w[n] = \begin{cases} 0.54 + 0.46 \cos\left(\frac{\pi n}{M/2}\right) & n \leq M/2 \\ 0 & n > M/2 \end{cases}$	<code>hamming(M+1)</code>	

Tradeoff – Ripple vs. Transition Width



Alternative Design through FFT

- ❑ To design order M filter:
- ❑ Over-Sample/discretize the frequency response at P points where $P \gg M$ ($P=15M$ is good)

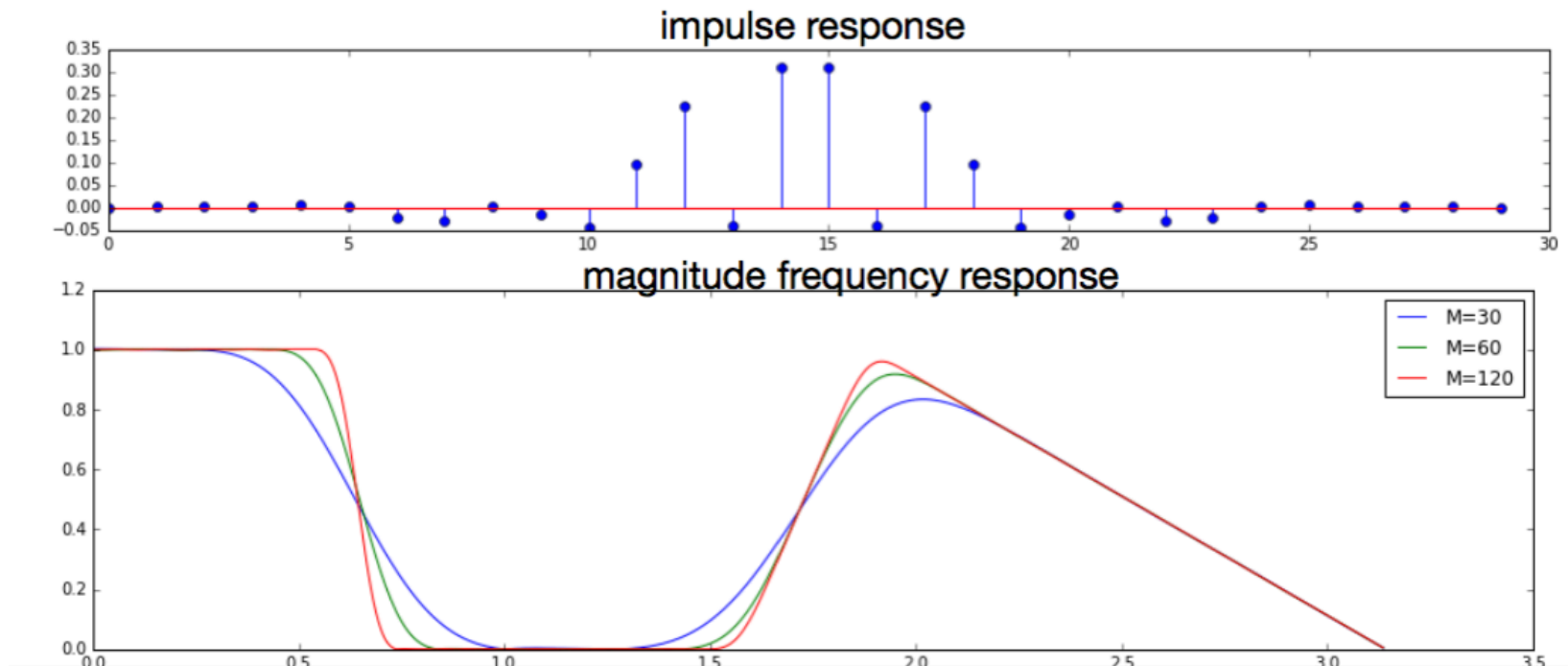
$$H_1(e^{j\omega_k}) = H_d(e^{j\omega_k})e^{-j\omega_k \frac{M}{2}}$$

- ❑ Sampled at: $\omega_k = k \frac{2\pi}{P} \quad |k = [0, \dots, P-1]$
- ❑ Compute $h_1[n] = \text{IDFT}_P(H_1[k])$
- ❑ Apply M+1 length window:

$$h_w[n] = w[n]h_1[n]$$

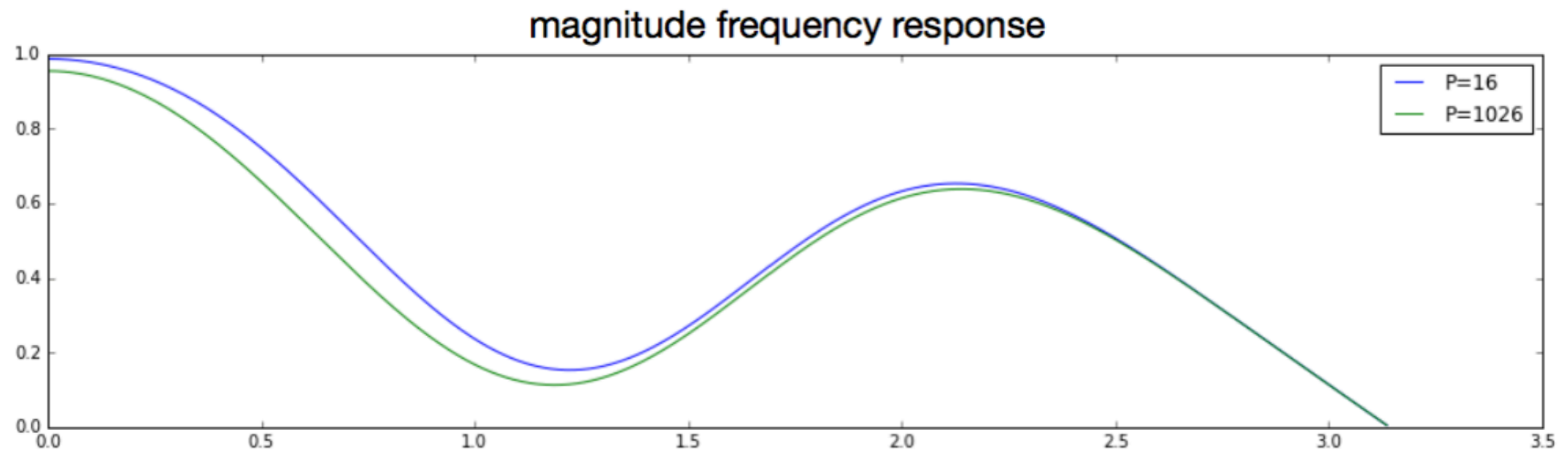
Example

- `signal.firwin2(M+1,omega_vec/pi, amp_vec)`
- `taps1 = signal.firwin2(30, [0.0,0.2,0.21,0.5, 0.6, 1.0], [1.0, 1.0, 0.0,0.0,1.0,0.0])`



Example

- For $M+1=14$
 - $P = 16$ and $P = 1026$

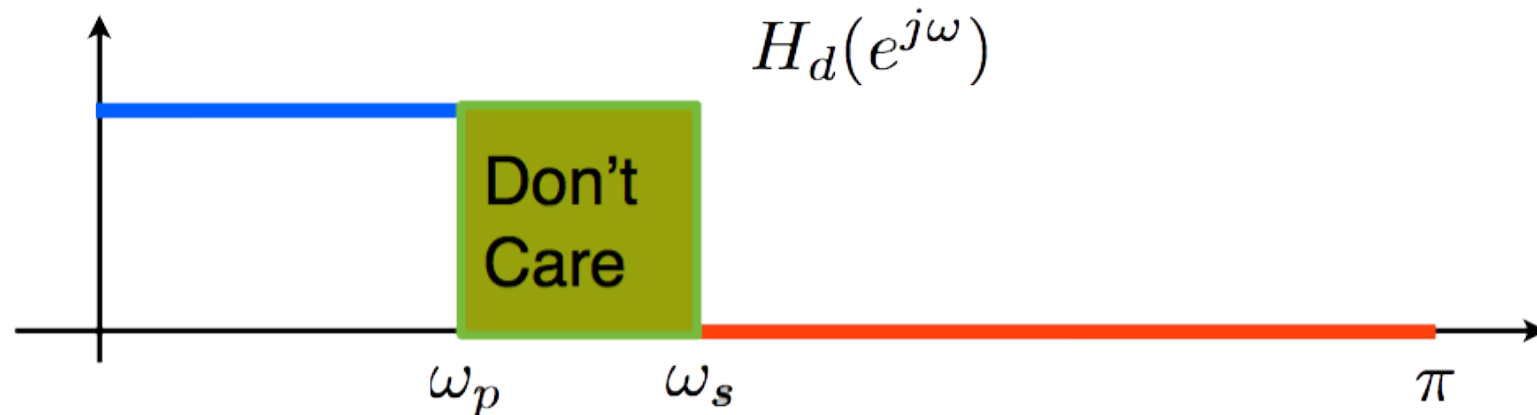




Optimal Filter Design

- ❑ Window method
 - Design Filters heuristically using windowed sinc functions
 - Choose order and window type
 - Check DTFT to see if filter specs are met
- ❑ Optimal design
 - Design a filter $h[n]$ with $H(e^{j\omega})$
 - Approximate $H_d(e^{j\omega})$ with some optimality criteria - or satisfies specs.

Optimality



- Least Squares:

$$\text{minimize} \int_{\omega \in \text{care}} |H(e^{j\omega}) - H_d(e^{j\omega})|^2 d\omega$$

- Variation: Weighted Least Squares:

$$\text{minimize} \int_{-\pi}^{\pi} W(\omega) |H(e^{j\omega}) - H_d(e^{j\omega})|^2 d\omega$$



Optimality

□ Chebychev Design (min-max)

$$\text{minimize}_{\omega \in \text{care}} \quad \max |H(e^{j\omega}) - H_d(e^{j\omega})|$$

- Parks-McClellan algorithm - equi-ripple
- Also known as Remez exchange algorithms (signal.remez)
- Can also use convex optimization



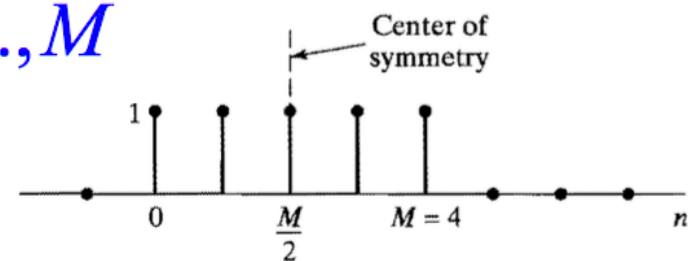
Parks-McClellan

- ❑ Assume designing a Type I GLP FIR filter:

FIR GLP: Type I

Type I Even Symmetry, M even

$$h[n] = h[M - n], \quad n = 0, 1, \dots, M$$



$$\text{Then } H(e^{j\omega}) = \sum_{n=0}^M h[n] e^{-j\omega n} = \underbrace{A(\omega)}_{\text{Real, Even}} e^{-j\omega M/2}$$

← integer delay

$$H(e^{j\omega}) = e^{-j\omega M/2} \left(\sum_{k=0}^{M/2} a[k] \cos \omega k \right)$$

$$a[0] = h[M/2],$$

$$a[k] = 2h[(M/2) - k], \quad k = 1, 2, \dots, M/2.$$



Parks-McClellan

- Assume designing a Type I GLP FIR filter:

$$H(e^{j\omega}) = A_e(e^{j\omega})e^{-j\omega M/2}.$$

$$A_e(e^{j\omega}) = h_e[0] + \sum_{n=1}^L 2h_e[n] \cos(\omega n).$$



Parks-McClellan

- Reformulate

$$A_e(e^{j\omega}) = h_e[0] + \sum_{n=1}^L 2h_e[n] \cos(\omega n).$$

- To fitting a polynomial

$$A_e(e^{j\omega}) = \sum_{k=0}^L a_k (\cos \omega)^k,$$

$$A_e(e^{j\omega}) = P(x)|_{x=\cos \omega},$$

$$P(x) = \sum_{k=0}^L a_k x^k.$$



Parks-McClellan

- Define approximation error function

$$E(\omega) = W(\omega)[H_d(e^{j\omega}) - A_e(e^{j\omega})],$$

- Apply min-max or Chebyshev criteria

$$\min_{\{h_e[n]: 0 \leq n \leq L\}} \left(\max_{\omega \in F} |E(\omega)| \right),$$



Parks-McClellan – Alternation Theorem

Alternation Theorem: Let F_P denote the closed subset consisting of the disjoint union of closed subsets of the real axis x . Furthermore,

$$P(x) = \sum_{k=0}^r a_k x^k$$

is an r^{th} -order polynomial, and $D_P(x)$ denotes a given desired function of x that is continuous on F_P ; $W_P(x)$ is a positive function, continuous on F_P , and

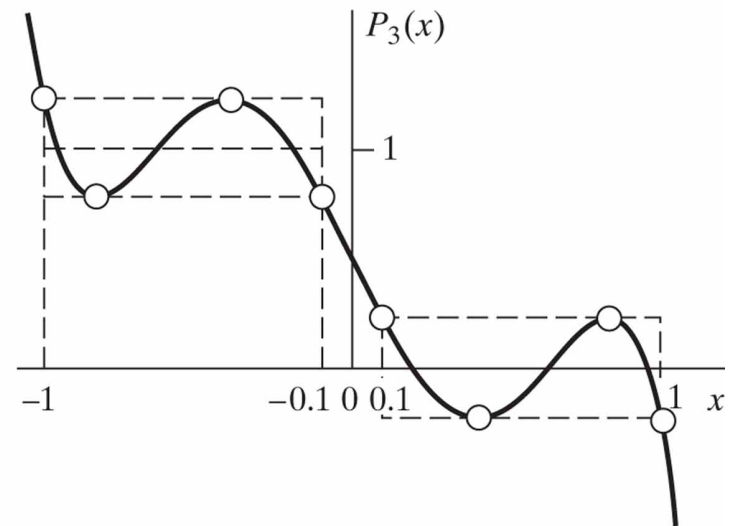
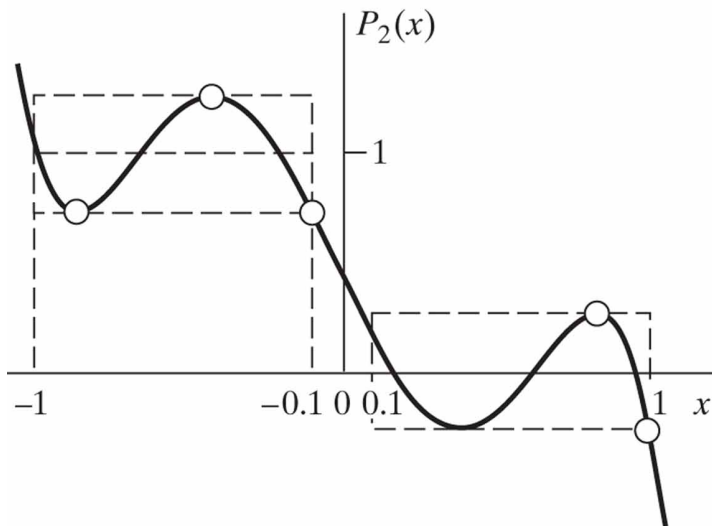
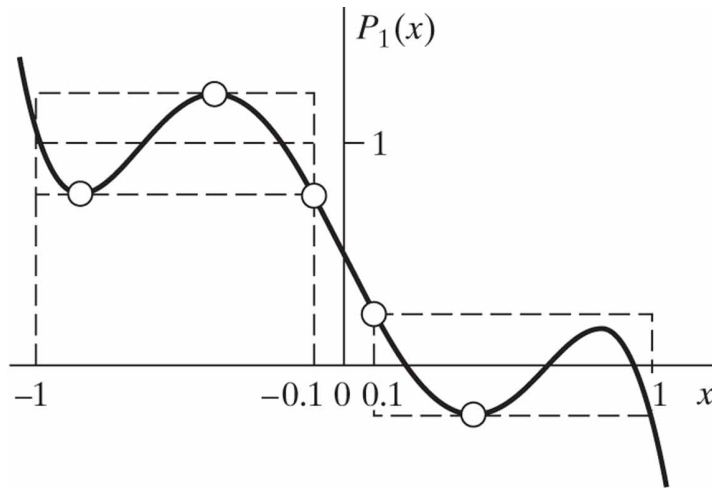
$$E_P(x) = W_P(x)[D_P(x) - P(x)]$$

is the weighted error. The maximum error is defined as

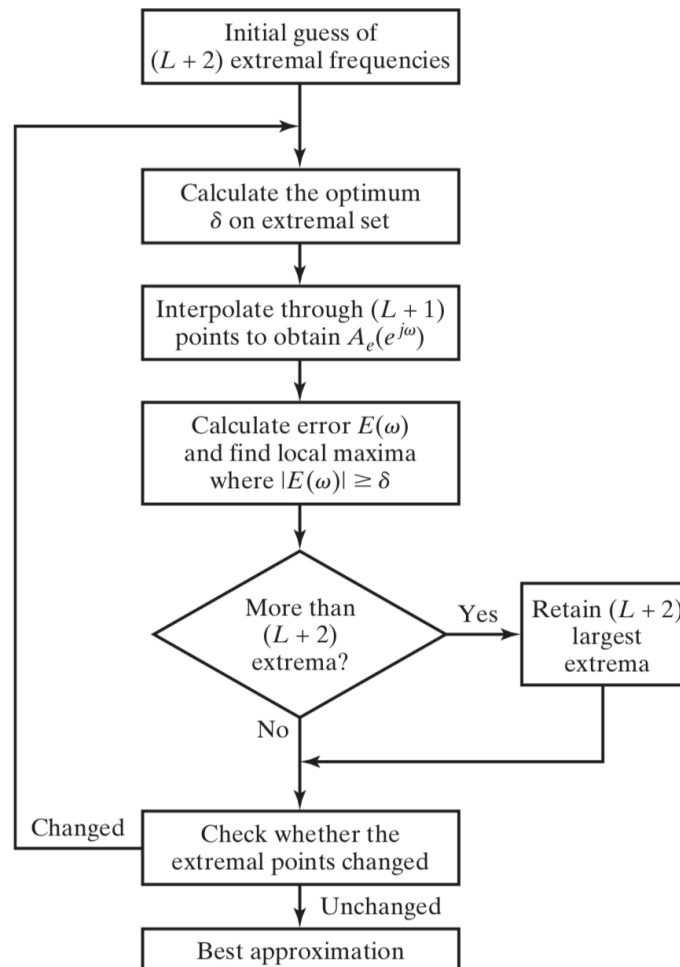
$$\|E\| = \max_{x \in F_P} |E_P(x)|.$$

A necessary and sufficient condition that $P(x)$ be the unique r^{th} -order polynomial that minimizes $\|E\|$ is that $E_P(x)$ exhibit *at least* $(r + 2)$ alternations; i.e., there must exist at least $(r + 2)$ values x_i in F_P such that $x_1 < x_2 < \cdots < x_{r+2}$ and such that $E_P(x_i) = -E_P(x_{i+1}) = \pm \|E\|$ for $i = 1, 2, \dots, (r + 1)$.

Alternation Theorem Example – 5th order



Parks–McClellan algorithm





Mathematical Optimization

(mathematical) optimization problem

$$\begin{array}{ll}\text{minimize} & f_0(x) \\ \text{subject to} & f_i(x) \leq b_i, \quad i = 1, \dots, m\end{array}$$

- $x = (x_1, \dots, x_n)$: optimization variables
- $f_0 : \mathbf{R}^n \rightarrow \mathbf{R}$: objective function
- $f_i : \mathbf{R}^n \rightarrow \mathbf{R}, i = 1, \dots, m$: constraint functions

optimal solution x^* has smallest value of f_0 among all vectors that satisfy the constraints



Examples

portfolio optimization

- variables: amounts invested in different assets
- constraints: budget, max./min. investment per asset, minimum return
- objective: overall risk or return variance

device sizing in electronic circuits

- variables: device widths and lengths
- constraints: manufacturing limits, timing requirements, maximum area
- objective: power consumption

data fitting

- variables: model parameters
- constraints: prior information, parameter limits
- objective: measure of misfit or prediction error



Solving Optimization Problems

general optimization problem

- very difficult to solve
- methods involve some compromise, *e.g.*, very long computation time, or not always finding the solution

exceptions: certain problem classes can be solved efficiently and reliably

- least-squares problems
- linear programming problems
- convex optimization problems



Linear Programming

$$\begin{array}{ll}\text{minimize} & c^T x \\ \text{subject to} & a_i^T x \leq b_i, \quad i = 1, \dots, m\end{array}$$

solving linear programs

- no analytical formula for solution
- reliable and efficient algorithms and software
- computation time proportional to n^2m if $m \geq n$; less with structure
- a mature technology

using linear programming

- not as easy to recognize as least-squares problems
- a few standard tricks used to convert problems into linear programs
(*e.g.*, problems involving ℓ_1 - or ℓ_∞ -norms, piecewise-linear functions)



Least-Squares Optimization

$$\text{minimize } \|Ax - b\|_2^2$$

solving least-squares problems

- analytical solution: $x^* = (A^T A)^{-1} A^T b$
- reliable and efficient algorithms and software
- computation time proportional to $n^2 k$ ($A \in \mathbf{R}^{k \times n}$); less if structured
- a mature technology

using least-squares

- least-squares problems are easy to recognize
- a few standard techniques increase flexibility (*e.g.*, including weights, adding regularization terms)



Convex Optimization

$$\begin{array}{ll}\text{minimize} & f_0(x) \\ \text{subject to} & f_i(x) \leq b_i, \quad i = 1, \dots, m\end{array}$$

- objective and constraint functions are convex:

$$f_i(\alpha x + \beta y) \leq \alpha f_i(x) + \beta f_i(y)$$

$$\text{if } \alpha + \beta = 1, \alpha \geq 0, \beta \geq 0$$

- includes least-squares problems and linear programs as special cases



Design Through Optimization

- ❑ Idea: Sample/discretize the frequency response

$$H(e^{j\omega}) \Rightarrow H(e^{j\omega_k})$$

- ❑ Sample points are fixed $\omega_k = k \frac{\pi}{P}$

$$-\pi \leq \omega_1 < \dots < \omega_p \leq \pi$$

- ❑ $M+1$ is the filter order
- ❑ $P \gg M + 1$ (rule of thumb $P=15M$)
- ❑ Yields a (good) approximation of the original problem



Example: Least Squares

- ❑ Target: Design $M+1 = 2N+1$ filter
- ❑ First design non-causal $\tilde{H}(e^{j\omega})$ and hence $\tilde{h}[n]$



Example: Least Squares

- ❑ Target: Design $M+1 = 2N+1$ filter
- ❑ First design non-causal $\tilde{H}(e^{j\omega})$ and hence $\tilde{h}[n]$
- ❑ Then, shift to make causal

$$h[n] = \tilde{h}[n - M/2]$$

$$H(e^{j\omega}) = e^{-j\frac{M}{2}} \tilde{H}(e^{j\omega})$$



Example: Least Squares

$$\tilde{h} = [\tilde{h}[-N], \tilde{h}[-N+1], \dots, \tilde{h}[N]]^T$$

$$b = [H_d(e^{j\omega_1}), \dots, H_d(e^{j\omega_P})]^T$$

$$A = \begin{bmatrix} e^{-j\omega_1(-N)} & \dots & e^{-j\omega_1(+N)} \\ \vdots & & \\ e^{-j\omega_P(-N)} & \dots & e^{-j\omega_P(+N)} \end{bmatrix}$$

$$\operatorname{argmin}_{\tilde{h}} ||A\tilde{h} - b||^2$$



Least-Squares

$$\operatorname{argmin}_{\tilde{h}} \quad ||A\tilde{h} - b||^2$$

Solution:

$$\tilde{h} = (A^* A)^{-1} A^* b$$

- ❑ Result will generally be non-symmetric and complex valued.
- ❑ However, if $\tilde{H}(e^{j\omega})$ is real, $\tilde{h}[n]$ should have symmetry!

Design of Linear-Phase L.P Filter

- Suppose:
 - $\tilde{H}(e^{j\omega})$ is real-symmetric
 - M is even ($M+1$ taps)
- Then:
 - $\tilde{h}[n]$ is real-symmetric around midpoint
- So:

$$\begin{aligned}\tilde{H}(e^{j\omega}) &= \tilde{h}[0] + \tilde{h}[1]e^{-j\omega} + \tilde{h}[-1]e^{+j\omega} \\ &\quad + \tilde{h}[2]e^{-j2\omega} + \tilde{h}[-2]e^{+j2\omega} \dots \\ &= \tilde{h}[0] + 2\cos(\omega)\tilde{h}[1] + 2\cos(2\omega)\tilde{h}[2] + \dots\end{aligned}$$

Reminder: FIR GLP: Type I – Example, $M=4$

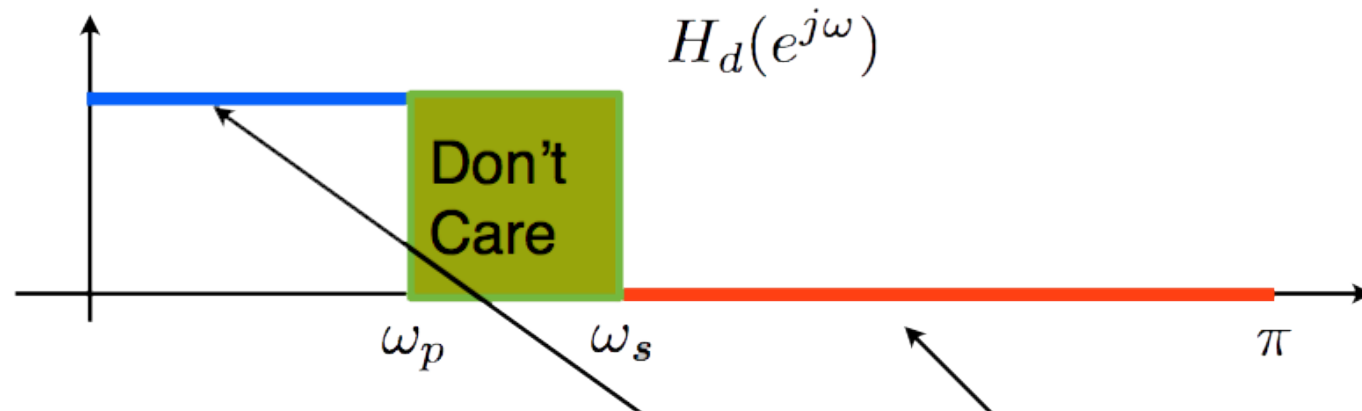
Type I Even Symmetry, M even

$$h[n] = h[M - n], \quad n = 0, 1, \dots, M$$

$$\text{Then } H(e^{j\omega}) = \sum_{n=0}^M h[n] e^{-j\omega n} = \underbrace{A(\omega)}_{\text{Real, Even}} e^{-j\omega M/2} \quad \leftarrow \text{integer delay}$$

$$\begin{aligned} H(e^{j\omega}) &= h[0] + h[1]e^{-j\omega} + h[2]e^{-j2\omega} + h[3]e^{-j3\omega} + h[4]e^{-j4\omega} \\ &= e^{-j2\omega} \left[h[0]e^{j2\omega} + h[1]e^{j\omega} + h[2] + h[1]e^{-j\omega} + h[0]e^{-j2\omega} \right] \\ &= \underbrace{\left[2h[0]\cos(2\omega) + 2h[1]\cos(\omega) + h[2] \right]}_{A(\omega) \text{ (even)}} e^{-j2\omega} \end{aligned}$$

Least-Squares Linear Phase Filter



Given M , ω_P , ω_s find the best LS filter:

$$A = \begin{bmatrix} 1 & \cdots & 2 \cos\left(\frac{M}{2}\omega_1\right) \\ \vdots & & \\ 1 & \cdots & 2 \cos\left(\frac{M}{2}\omega_p\right) \\ 1 & \cdots & 2 \cos\left(\frac{M}{2}\omega_s\right) \\ \vdots & & \\ 1 & \cdots & 2 \cos\left(\frac{M}{2}\omega_P\right) \end{bmatrix}$$

$$b = [1, 1, \dots, 1, 0, 0, \dots, 0]^T$$

Least-Squares Linear Phase Filter

Given M , ω_P , ω_s find the best LS filter:

$$A = \begin{bmatrix} 1 & \cdots & 2 \cos(\frac{M}{2}\omega_1) \\ \vdots & & \\ 1 & \cdots & 2 \cos(\frac{M}{2}\omega_p) \\ 1 & \cdots & 2 \cos(\frac{M}{2}\omega_s) \\ \vdots & & \\ 1 & \cdots & 2 \cos(\frac{M}{2}\omega_P) \end{bmatrix} \quad b = [1, 1, \dots, 1, 0, 0, \dots, 0]^T$$

$$\tilde{h}_+ = [\tilde{h}[0], \dots, \tilde{h}[\frac{M}{2}]]^T = (A^* A)^{-1} A^* b$$

$$\tilde{h} = \begin{cases} \tilde{h}_+[n] & n \geq 0 \\ \tilde{h}_+[-n] & n < 0 \end{cases}$$

$$h[n] = \tilde{h}[n - M/2]$$



Extension:

- ❑ LS has no preference for pass band or stop band
- ❑ Use weighting of LS to change ratio

want to solve the discrete version of:

$$\text{minimize } \int_{-\pi}^{\pi} W(\omega) |H(e^{j\omega}) - H_d(e^{j\omega})|^2 d\omega$$

where $W(\omega)$ is δ_p in the pass band and δ_s in stop band

Similarly: $W(\omega)$ is 1 in the pass band and δ_p/δ_s in stop band

Weighted Least-Squares

$$\operatorname{argmin}_{\tilde{h}_+} (A\tilde{h}_+ - b)^* W^2 (A\tilde{h}_+ - b)$$

Solution:

$$\tilde{h}_+ = (A^* W^2 A)^{-1} W^2 A^* b$$

$$W = \begin{bmatrix} 1 & & & & & 0 \\ & 1 & & & & \\ & & \dots & & & \\ & & & \frac{\delta_p}{\delta_s} & & \\ & & & & \dots & \\ 0 & & & & & \frac{\delta_p}{\delta_s} \end{bmatrix}$$



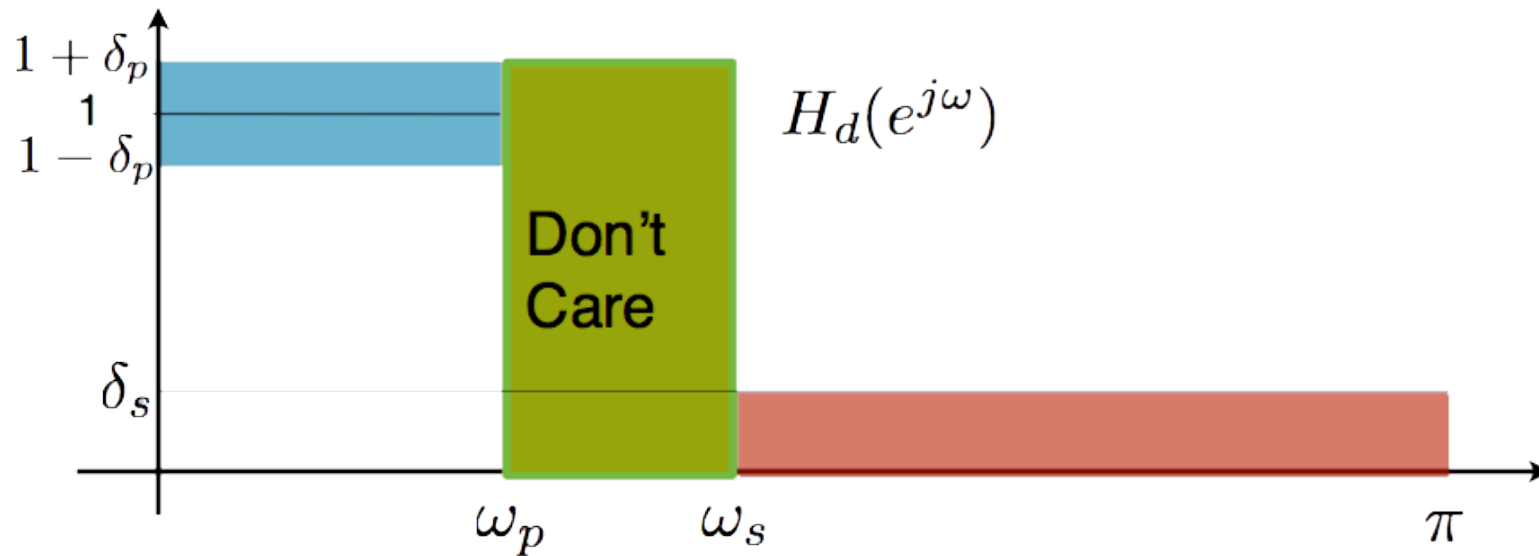
Optimality

❑ Chebychev Design (min-max)

$$\text{minimize}_{\omega \in \text{care}} \quad \max |H(e^{j\omega}) - H_d(e^{j\omega})|$$

- Parks-McClellan algorithm - equiripple
- Also known as Remez exchange algorithms (signal.remez)
- Can also use optimization

Specifications



- ❑ Filter specifications are given in terms of boundaries



Min-Max Filter Design

□ Constraints:

- min-max pass-band ripple

$$1 - \delta_p \leq |H(e^{j\omega})| \leq 1 + \delta_p, \quad 0 \leq \omega \leq \omega_p$$

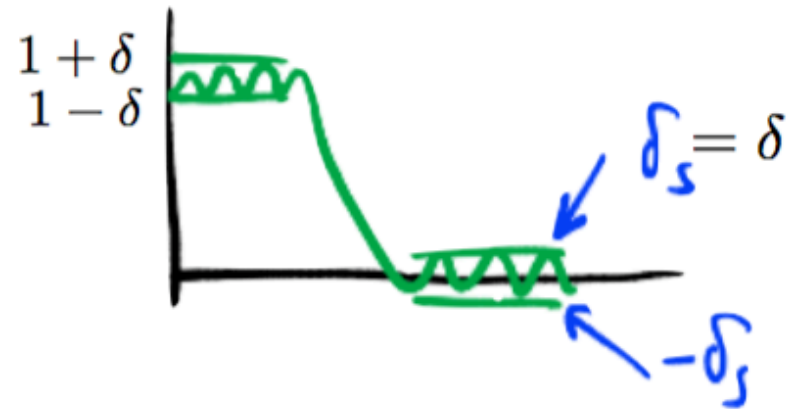
- min-max stop-band ripple

$$|H(e^{j\omega})| \leq \delta_s, \quad \omega_s \leq \omega \leq \pi$$

Min-Max Ripple Design

□ Recall, $\tilde{H}(e^{j\omega})$ is symmetric and real

□ Given ω_p , ω_s , M , find δ , $\tilde{h}_+$



minimize δ

Subject to :

$$\begin{aligned} 1 - \delta &\leq \tilde{H}(e^{j\omega_k}) \leq 1 + \delta & 0 \leq \omega_k \leq \omega_p \\ -\delta &\leq \tilde{H}(e^{j\omega_k}) \leq \delta & \omega_s \leq \omega_k \leq \pi \\ \delta &> 0 \end{aligned}$$

□ Formulation is a linear program with solution δ , $\tilde{h}_+$

□ A well studied class of problems with good solvers

Min-Max Ripple via LP

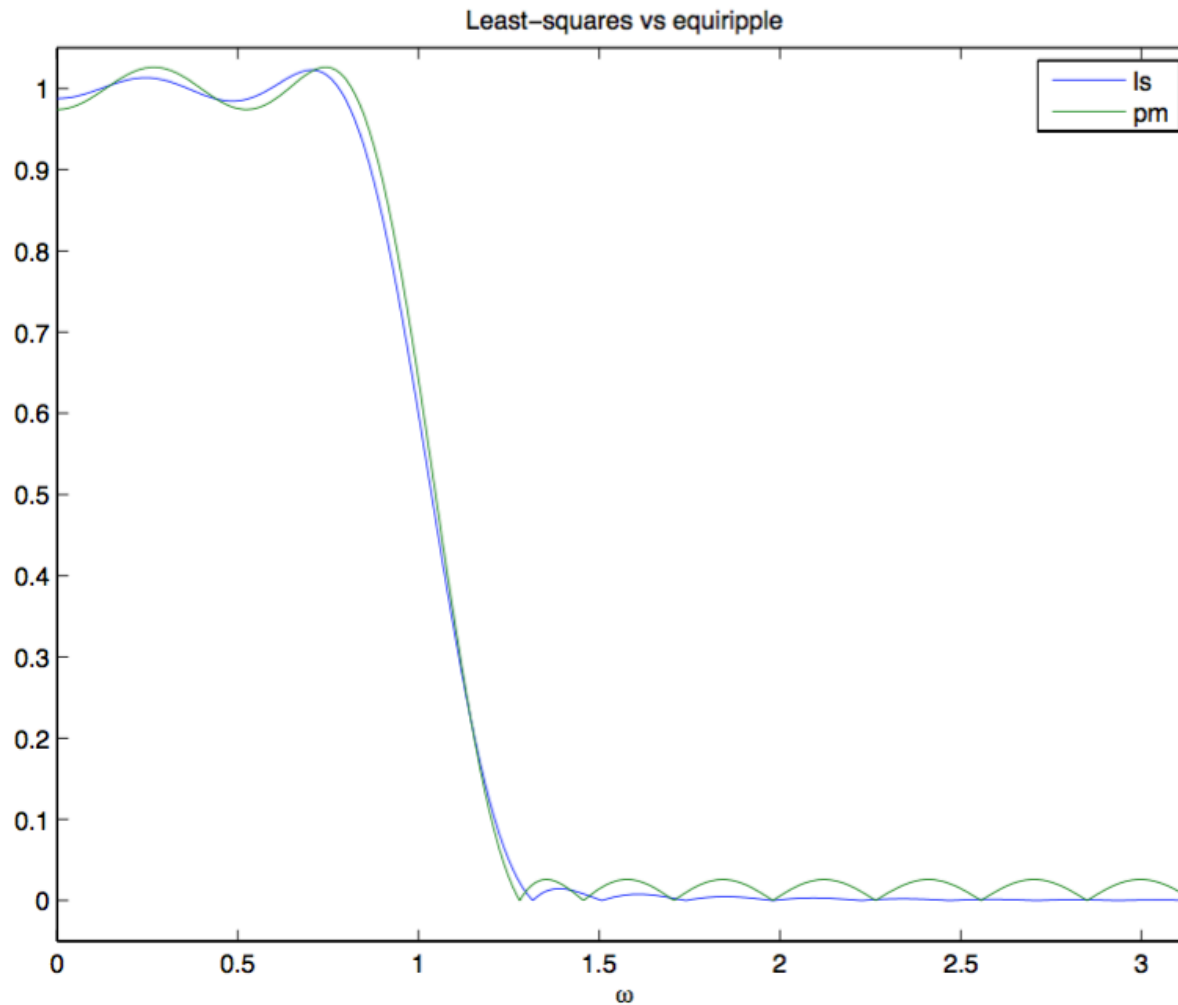
minimize δ
subject to :

$$\begin{aligned} 1 - \delta &\preceq A_p \tilde{h}_+ \preceq 1 + \delta \\ -\delta &\preceq A_s \tilde{h}_+ \preceq \delta \\ \delta &> 0 \end{aligned}$$

$$A_p = \begin{bmatrix} 1 & 2 \cos(\omega_1) & \cdots & 2 \cos(\frac{M}{2} \omega_1) \\ \vdots & & & \\ 1 & 2 \cos(\omega_p) & \cdots & 2 \cos(\frac{M}{2} \omega_p) \end{bmatrix}$$
$$A_s = \begin{bmatrix} 1 & 2 \cos(\omega_s) & \cdots & 2 \cos(\frac{M}{2} \omega_s) \\ \vdots & & & \\ 1 & 2 \cos(\omega_P) & \cdots & 2 \cos(\frac{M}{2} \omega_P) \end{bmatrix}$$

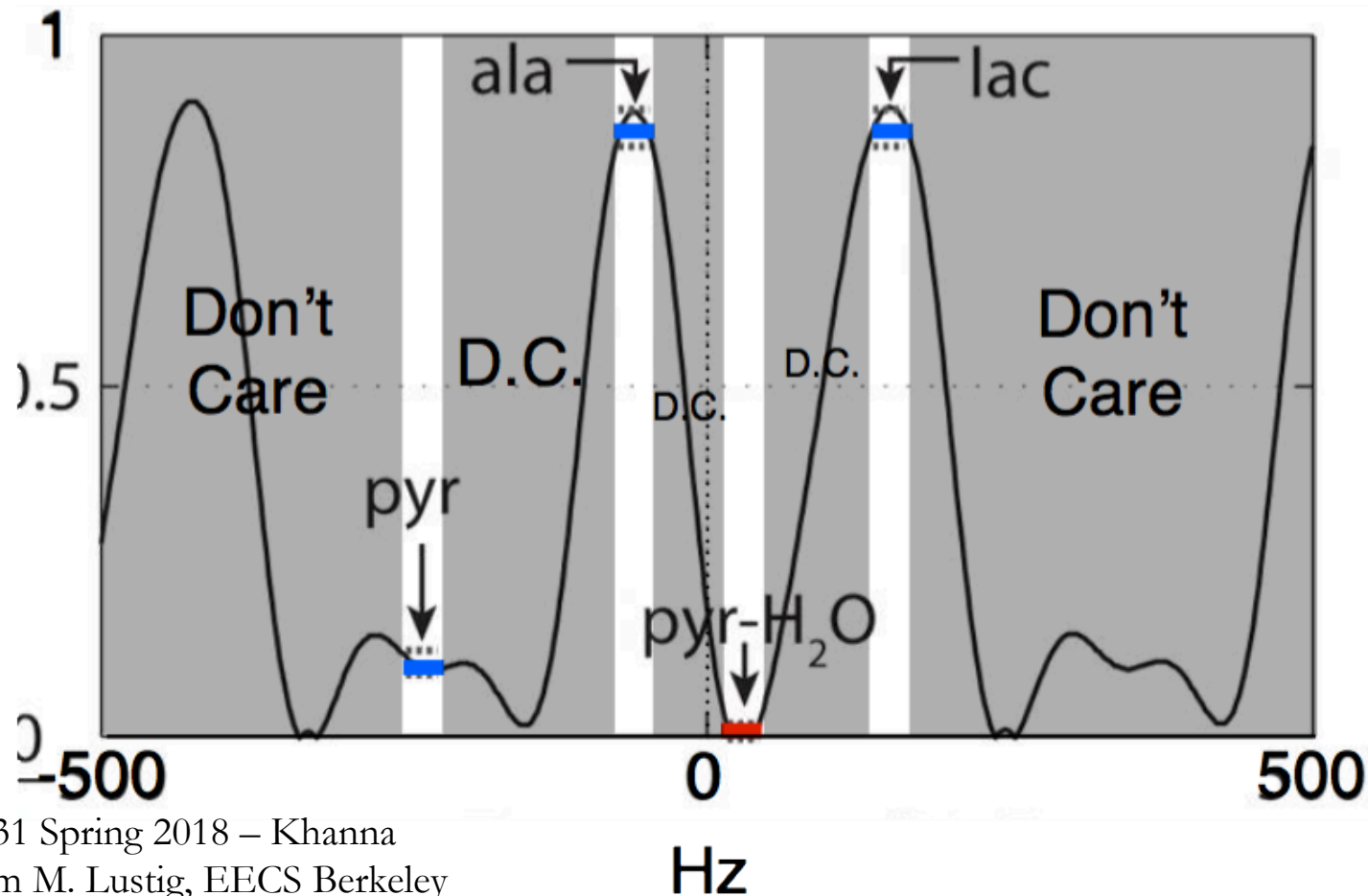
capital P

Least-Squares v.s. Min-Max



Example of Complex Filter

- ❑ Larson et. al, “Multiband Excitation Pulses for Hyperpolarized ^{13}C Dynamic Chemical Shift Imaging” JMR 2008;194(1):121-127
- ❑ Need to design 11 taps filter with following frequency response:





Convex Optimization

- ❑ Many tools and Solvers
- ❑ Tools:
 - CVX (Matlab) <http://cvxr.com/cvx/>
 - CVXOPT, CVXMOD (Python)
- ❑ Engines:
 - Sedumi (Free)
 - MOSEK (commercial)

Using CVX (in Matlab)

```
M = 16;
wp = 0.5*pi;
ws = 0.6*pi;
MM = 15*M;
w = linspace(0,pi,MM);
```

```
idxp = find(w <= wp);
idxs = find(w >= ws);
```

```
Ap = [ones(length(idxp),1) 2*cos(kron(w(idxp)',
[1:M/2]))];
As = [ones(length(idxs),1) 2*cos(kron(w(idxs)',
[1:M/2]))];
```

% optimization

```
cvx_begin
    variable hh(M/2+1,1);
    variable d(1,1);
```

```
    minimize(d)
```

```
    subject to
```

```
        Ap*hh <= 1+d;
```

```
        Ap*hh >= 1-d;
```

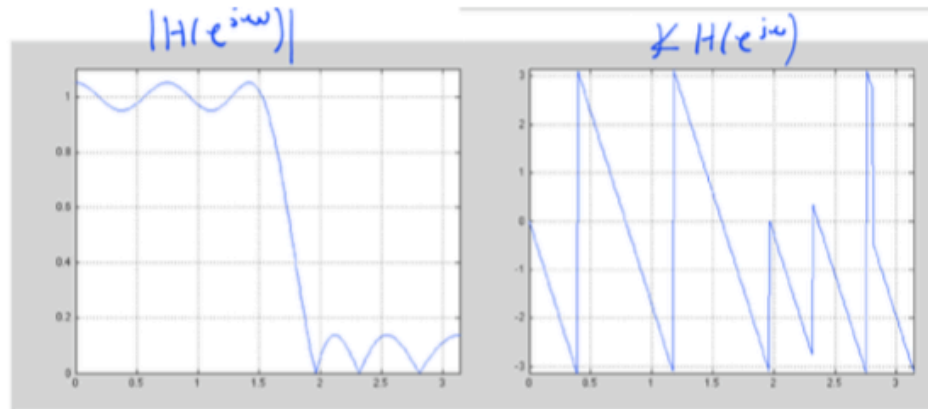
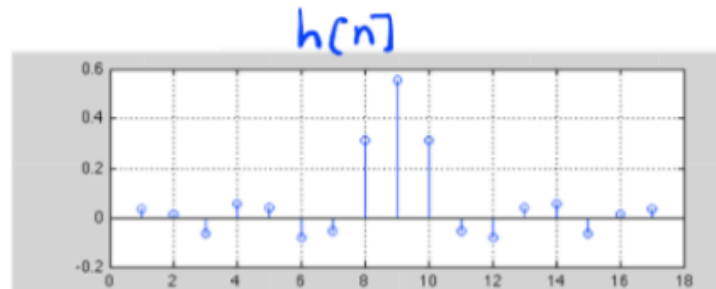
```
        As*hh < d;
```

```
        As*hh > -d;
```

```
        ds > 0;
```

```
cvx_end
```

```
h = [hh(end:-1:1); hh(2:end)];
```





Admin

- ❑ HW 7 due tomorrow @ midnight