ESE 531: Digital Signal Processing

Lec 21: April 14, 2020

Adaptive Filters

Lecture Outline

- Circular convolution as linear convolution with aliasing
- Adaptive Filters

Circular Convolution

Linear Convolution with aliasing!

Circular Convolution

- Circular Convolution:

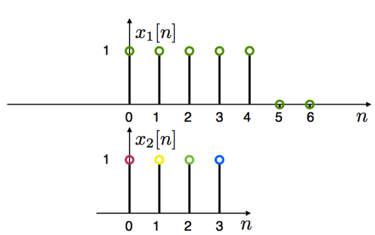
$$x_1[n] \textcircled{N} x_2[n] \triangleq \sum_{m=0}^{N-1} x_1[m] x_2[((n-m))_N]$$

For two signals of length N

Note: Circular convolution is commutative

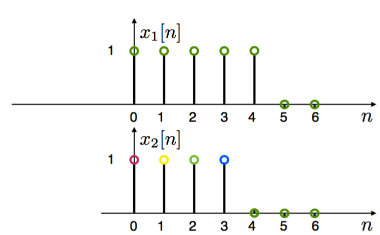
$$x_2[n] \textcircled{N} x_1[n] = x_1[n] \textcircled{N} x_2[n]$$

Compute Circular Convolution Sum



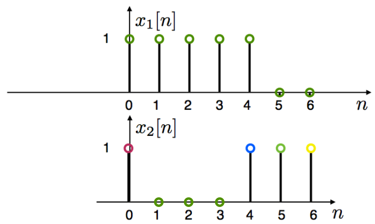
$$x_1[n] \textcircled{N} x_2[n] \triangleq \sum_{m=0}^{N-1} x_1[m] x_2[((n-m))_N]$$

Compute Circular Convolution Sum



$$x_1[n] \textcircled{N} x_2[n] \triangleq \sum_{m=0}^{N-1} x_1[m] x_2[((n-m))_N]$$

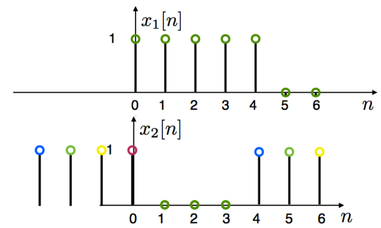
Compute Circular Convolution Sum



$$x_1[n] \circledN x_2[n] \triangleq \sum_{m=0}^{N-1} x_1[m]x_2[((n-m))_N]$$

Compute Circular Convolution Sum

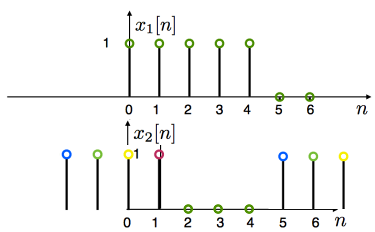
y[0]=2



$$x_1[n] \circledN x_2[n] \triangleq \sum_{m=0}^{N-1} x_1[m]x_2[((n-m))_N]$$

Compute Circular Convolution Sum

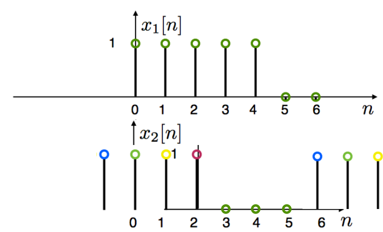
y[0]=2
y[1]=2



$$x_1[n] \circledN x_2[n] \triangleq \sum_{m=0}^{N-1} x_1[m]x_2[((n-m))_N]$$

Compute Circular Convolution Sum

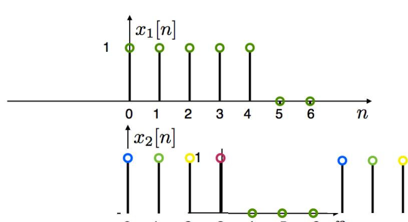
y[0]=2
y[1]=2
y[2]=3



$$x_1[n] \circledN x_2[n] \triangleq \sum_{m=0}^{N-1} x_1[m]x_2[((n-m))_N]$$

Compute Circular Convolution Sum

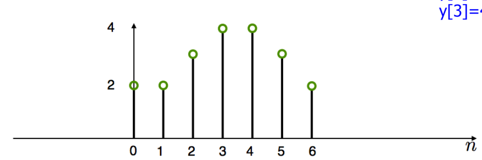
y[0]=2
y[1]=2
y[2]=3
y[3]=4



$$x_1[n] \circledN x_2[n] \triangleq \sum_{m=0}^{N-1} x_1[m]x_2[((n-m))_N]$$

Result

y[0]=2
y[1]=2
y[2]=3
y[3]=4



$$x_1[n] \circledN x_2[n] \triangleq \sum_{m=0}^{N-1} x_1[m]x_2[((n-m))_N]$$

Linear Convolution

- We start with two non-periodic sequences:

$$x[n] \quad 0 \le n \le L-1$$
$$h[n] \quad 0 \le n \le P-1$$

  - E.g. x[n] is a signal and h[n] a filter's impulse response
- We want to compute the linear convolution:

$$y[n] = x[n] * h[n] = \sum_{m=0}^{L-1} x[m]h[n-m]$$

  - y[n] is nonzero for $0 \le n \le L+P-2$ with length M=L+P-1

Requires LP multiplications

Linear Convolution via Circular Convolution

- Zero-pad x[n] by P-1 zeros

$$x_{zp}[n] = \begin{cases} x[n] & 0 \le n \le L-1 \\ 0 & L \le n \le L+P-2 \end{cases}$$

- Zero-pad h[n] by L-1 zeros

$$h_{zp}[n] = \begin{cases} h[n] & 0 \le n \le P-1 \\ 0 & P \le n \le L+P-2 \end{cases}$$

- Now, both sequences are length M=L+P-1

Circular Conv. via Linear Conv. w/ Aliasing

- If the DTFT $X(e^{j\omega})$ of a sequence x[n] is sampled at N frequencies $\omega_k = 2\pi k/N$, then the resulting sequence X[k] corresponds to the periodic sequence

$$\tilde{x}[n] = \sum_{r=-\infty}^{\infty} x[n-rN].$$

Circular Conv. via Linear Conv. w/ Aliasing

- If the DTFT $X(e^{j\omega})$ of a sequence x[n] is sampled at N frequencies $\omega_k = 2\pi k/N$, then the resulting sequence X[k] corresponds to the periodic sequence

$$\tilde{x}[n] = \sum_{r=-\infty}^{\infty} x[n-rN].$$

- And $X[k] = \begin{cases} X(e^{j(2\pi k/N)}), & 0 \le k \le N-1, \\ 0, & \text{otherwise,} \end{cases}$ is the DFT of one period given as

$$x_p[n] = \begin{cases} \tilde{x}[n], & 0 \le n \le N-1, \\ 0, & \text{otherwise.} \end{cases}$$

Circular Conv. via Linear Conv. w/ Aliasing

$$x_p[n] = \begin{cases} \tilde{x}[n], & 0 \le n \le N-1, \\ 0, & \text{otherwise.} \end{cases}$$

- If x[n] has length less than or equal to N, then $x_p[n]=x[n]$

Circular Conv. via Linear Conv. w/ Aliasing

$$x_p[n] = \begin{cases} \tilde{x}[n], & 0 \le n \le N-1, \\ 0, & \text{otherwise.} \end{cases}$$

- If x[n] has length less than or equal to N, then $x_p[n]=x[n]$
- However if the length of x[n] is greater than N, this might not be true and we get aliasing in time
  - N-point convolution results in N-point sequence

Circular Conv. via Linear Conv. w/ Aliasing

- Given two N-point sequences ($x_1[n]$ and $x_2[n]$) and their N-point DFTs ($X_1[k]$ and $X_2[k]$)
- The N-point DFT of $x_3[n]=x_1[n]*x_2[n]$ is defined as

$$X_3[k] = X_3(e^{j(2\pi k/N)})$$

Circular Conv. via Linear Conv. w/ Aliasing

- Given two N-point sequences ($x_1[n]$ and $x_2[n]$) and their N-point DFTs ($X_1[k]$ and $X_2[k]$)
- The N-point DFT of $x_3[n]=x_1[n]*x_2[n]$ is defined as

$$X_3[k] = X_3(e^{j(2\pi k/N)})$$

- And $X_3[k]=X_1[k]\bullet X_2[k]$, where the inverse DFT of $X_3[k]$ is

$$x_{3p}[n] = \begin{cases} \tilde{x}_3[n] & 0 \le n \le N-1 \\ 0 & \text{else} \end{cases}$$

Circular Conv. via Linear Conv. w/ Aliasing

- Given two N-point sequences ($x_1[n]$ and $x_2[n]$) and their N-point DFTs ($X_1[k]$ and $X_2[k]$)
- The N-point DFT of $x_3[n]=x_1[n]*x_2[n]$ is defined as

$$X_3[k] = X_3(e^{j(2\pi k/N)})$$

- And $X_3[k]=X_1[k]\bullet X_2[k]$, where the inverse DFT of $X_3[k]$ is

$$x_{3p}[n] = \begin{cases} \sum_{r=-\infty}^{\infty} x_3[n-rN], & 0 \le n \le N-1, \\ 0, & \text{otherwise}, \end{cases}$$

Circular Conv. as Linear Conv. w/ Aliasing

$$x_{3p}[n] = \begin{cases} \sum_{r=-\infty}^{\infty} x_3[n-rN], & 0 \le n \le N-1, \\ 0, & \text{otherwise}, \end{cases}$$

- Thus

$$x_{3p}[n] = \begin{cases} \sum_{r=-\infty}^{\infty} x_1[n-rN]*x_2[n-rN] & 0 \le n \le N-1 \\ 0 & \text{else} \end{cases}$$

Circular Conv. as Linear Conv. w/ Aliasing

$$x_{3p}[n] = \begin{cases} \sum_{r=-\infty}^{\infty} x_3[n-rN], & 0 \le n \le N-1, \\ 0, & \text{otherwise}, \end{cases}$$

- Thus

$$x_{3p}[n] = \begin{cases} \sum_{r=-\infty}^{\infty} x_1[n-rN]*x_2[n-rN] & 0 \le n \le N-1 \\ 0 & \text{else} \end{cases}$$

$$x_{3p}[n] = x_1[n] \circledN x_2[n]$$

Circular Conv. as Linear Conv. w/ Aliasing

$$x_{3p}[n] = \begin{cases} \sum_{r=-\infty}^{\infty} x_3[n-rN], & 0 \le n \le N-1, \\ 0, & \text{otherwise}, \end{cases}$$

- Thus

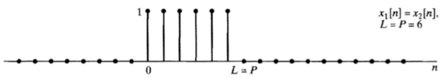
$$x_{3p}[n] = \begin{cases} \sum_{r=-\infty}^{\infty} x_1[n-rN]*x_2[n-rN] & 0 \le n \le N-1 \\ 0 & \text{else} \end{cases}$$

$$x_{3p}[n] = x_1[n] \circledN x_2[n]$$

- The N-point circular convolution is the sum of linear convolutions shifted in time by N

Example 1:

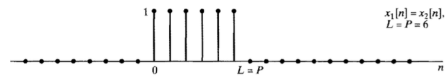
- Let



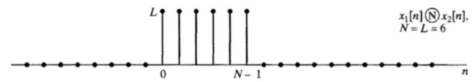
- The N=L=6-point circular convolution results in

Example 1:

- Let

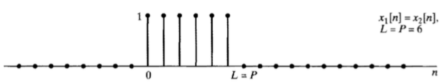


- The N=L=6-point circular convolution results in



Example 1:

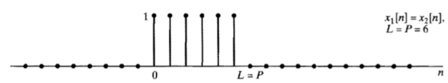
- Let



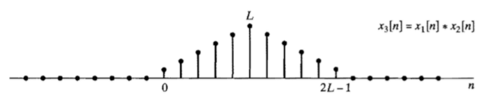
- The linear convolution results in

Example 1:

- Let

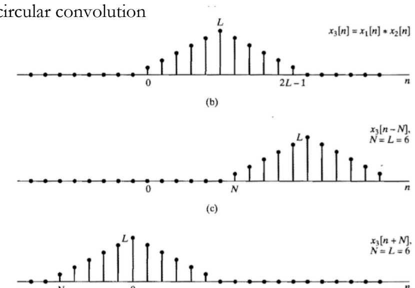


- The linear convolution results in



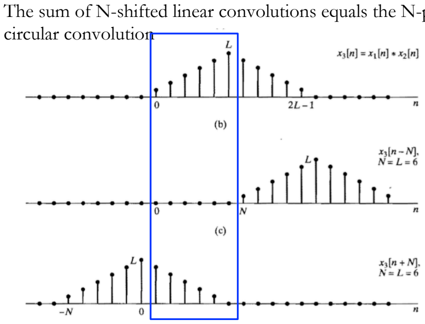
Example 1:

- The sum of N-shifted linear convolutions equals the N-point circular convolution



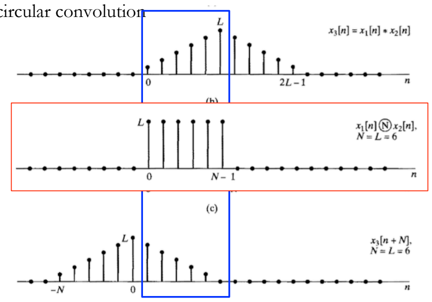
Example 1:

- The sum of N-shifted linear convolutions equals the N-point circular convolution



Example 1:

- The sum of N-shifted linear convolutions equals the N-point circular convolution

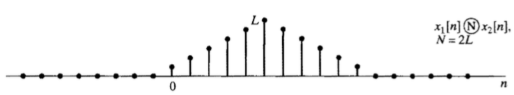


Example 1:

- If I want the circular convolution and linear convolution to be the same, what do I do?

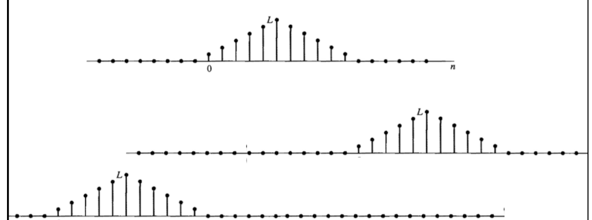
Example 1:

- If I want the circular convolution and linear convolution to be the same, what do I do?
  - Take the N=2L-point circular convolution



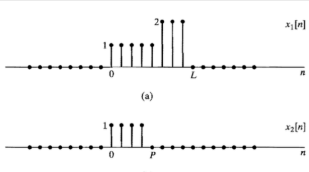
Example 1:

- If I want the circular convolution and linear convolution to be the same, what do I do?
  - Take the N=2L-point circular convolution



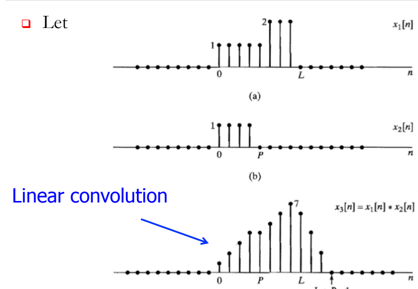
Example 2:

- Let



Example 2:

- Let

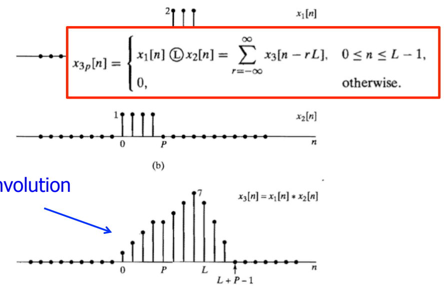


- What does the L-point circular convolution look like?

Example 2:

- Let

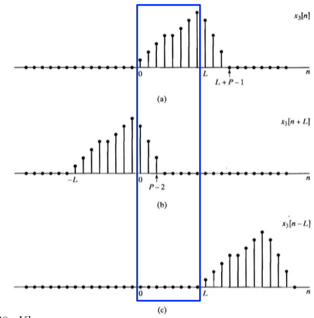
$$x_{3p}[n] = \begin{cases} x_1[n] \; Ⓛ \; x_2[n] = \sum_{r=-\infty}^{\infty} x_3[n - rL], & 0 \le n \le L-1, \\ 0, & \text{otherwise.} \end{cases}$$



- What does the L-point circular convolution look like?

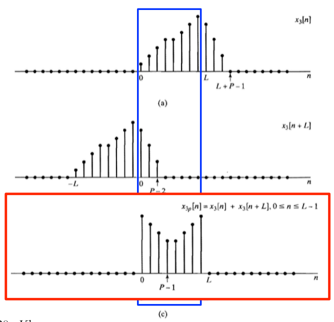
Example 2:

- The L-shifted linear convolutions



Example 2:

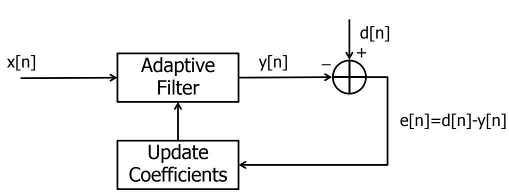
- The L-shifted linear convolutions



Adaptive Filters

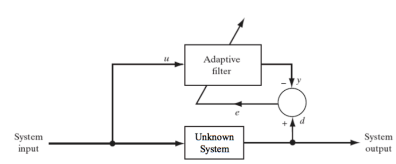
Adaptive Filters

- An adaptive filter is an adjustable filter that processes in time
  - It adapts…



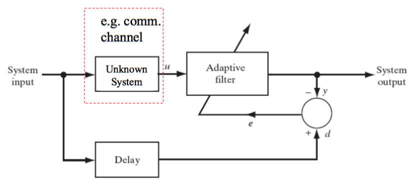
Adaptive Filter Applications

- System Identification



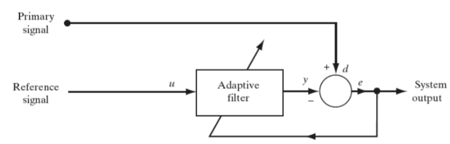
Adaptive Filter Applications

- Identification of inverse system



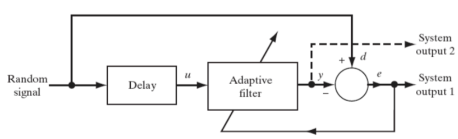
Adaptive Filter Applications

- Adaptive Interference Cancellation



Adaptive Filter Applications

- Adaptive Prediction

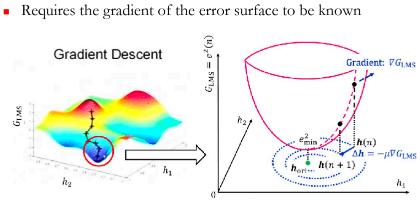


Stochastic Gradient Approach

- Most commonly used type of Adaptive Filters
- Define cost function as mean-squared error
  - Difference between filter output and desired response
- Based on the method of steepest descent
  - Move towards the minimum on the error surface to get to minimum
  - Requires the gradient of the error surface to be known

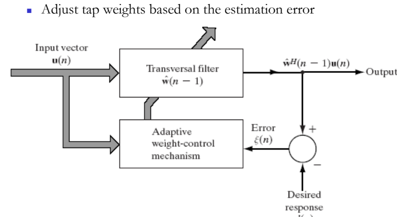
Stochastic Gradient Approach

- Most commonly used type of Adaptive Filters
- Define cost function as mean-squared error
  - Difference between filter output and desired response
- Based on the method of steepest descent
  - Move towards the minimum on the error surface to get to minimum
  - Requires the gradient of the error surface to be known



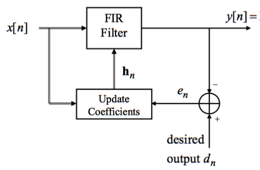
Least-Mean-Square (LMS) Algorithm

- The LMS Algorithm consists of two basic processes
  - Filtering process
    - Calculate the output of FIR filter by convolving input and taps
    - Calculate estimation error by comparing the output to desired signal
  - Adaptation process
    - Adjust tap weights based on the estimation error



(a)

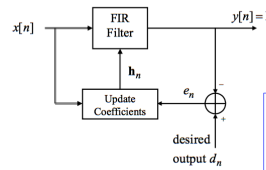
Adaptive FIR Filter: LMS



$$\mathbf{x}_n = (x[n], x[n-1]..., x[n-M])^T$$

$$\mathbf{h}_n = (h_n[0], h_n[1], ..., h_n[M])^T$$

Adaptive FIR Filter: LMS

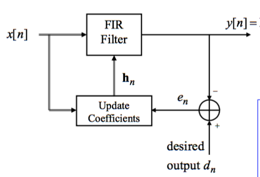


$$\mathbf{x}_n = (x[n], x[n-1]..., x[n-M])^T$$

$$\mathbf{h}_n = (h_n[0], h_n[1], ..., h_n[M])^T$$

$$e_n^2 = (d[n] - y[n])^2 = (d[n] - \mathbf{h}_n^T \mathbf{x}_n)^2$$

Adaptive FIR Filter: LMS



$$\mathbf{x}_n = (x[n], x[n-1]..., x[n-M])^T$$

$$\mathbf{h}_n = (h_n[0], h_n[1], ..., h_n[M])^T$$

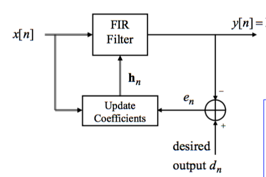
$$e_n^2 = (d[n] - y[n])^2 = (d[n] - \mathbf{h}_n^T \mathbf{x}_n)^2$$

$$\frac{d\,e_n^2}{d\,\mathbf{h}_n} = -2(d[n] - \mathbf{h}_n^T \mathbf{x}_n)\mathbf{x}_n = -2e_n \mathbf{x}_n$$

$$\vec{a}^T\vec{b} = \begin{bmatrix} a_1 & a_2 & a_3 & a_4 \end{bmatrix} \begin{bmatrix} b_1 \\ b_2 \\ b_3 \\ b_4 \end{bmatrix} = a_1b_1 + a_2b_2 + a_3b_3 + a_4b_4$$

$$\frac{\partial(a^Tb)}{\partial\vec{a}} = \begin{bmatrix} \frac{\partial(a^Tb)}{\partial a_1} & \frac{\partial(a^Tb)}{\partial a_2} & \frac{\partial(a^Tb)}{\partial a_3} & \frac{\partial(a^Tb)}{\partial a_4} \end{bmatrix}^T = \begin{bmatrix} b_1 \\ b_2 \\ b_3 \\ b_4 \end{bmatrix} = \vec{b}$$

Adaptive FIR Filter: LMS



$$\mathbf{x}_n = (x[n], x[n-1]..., x[n-M])^T$$

$$\mathbf{h}_n = (h_n[0], h_n[1], ..., h_n[M])^T$$

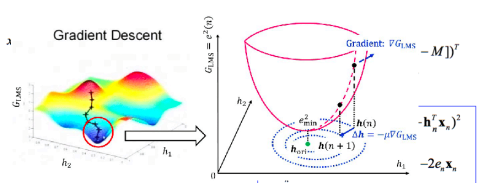
$$e_n^2 = (d[n] - y[n])^2 = (d[n] - \mathbf{h}_n^T \mathbf{x}_n)^2$$

$$\frac{d\,e_n^2}{d\,\mathbf{h}_n} = -2(d[n] - \mathbf{h}_n^T \mathbf{x}_n)\mathbf{x}_n = -2e_n \mathbf{x}_n$$

Coefficient Update: Move in direction *opposite* to sign of gradient, proportional to magnitude of gradient $\mathbf{h}_{n+1} = \mathbf{h}_n + 2\mu e_n \mathbf{x}_n$

Stochastic Gradient Algorithm

Adaptive FIR Filter: LMS

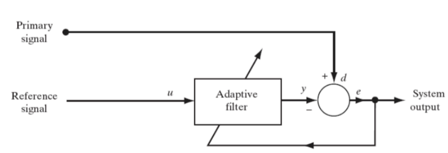


Coefficient Update: Move in direction *opposite* to sign of gradient, proportional to magnitude of gradient $\mathbf{h}_{n+1} = \mathbf{h}_n + 2\mu e_n \mathbf{x}_n$

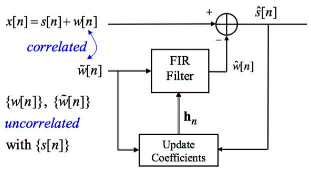
Stochastic Gradient Algorithm

Adaptive Filter Applications

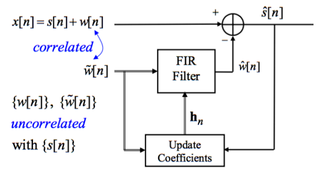
- Adaptive Interference Cancellation



Adaptive Interference Cancellation

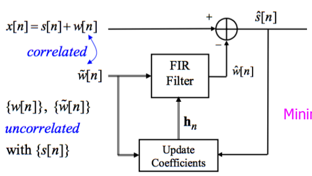


Adaptive Interference Cancellation



$$\hat{s}[n]=s[n]+w[n]-\hat{w}[n]$$
$$=s[n]+w[n]-h_n^T\tilde{w}_n$$

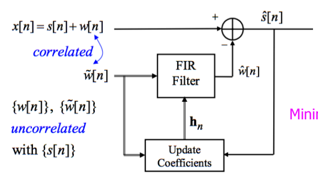
Adaptive Interference Cancellation



$$\hat{s}[n]=s[n]+w[n]-\hat{w}[n]$$
$$=s[n]+w[n]-h_n^T\tilde{w}_n$$

Minimizing $(\hat{s}[n])^2$ removes noise w[n]

Adaptive Interference Cancellation

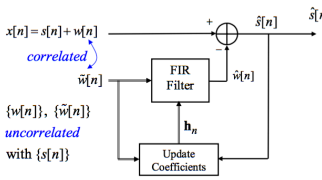


$$\hat{s}[n]=s[n]+w[n]-\hat{w}[n]$$
$$=s[n]+w[n]-h_n^T\tilde{w}_n$$

Minimizing $(\hat{s}[n])^2$ removes noise w[n]

$$\left(\hat{s}[n]\right)^2=\left(s[n]+w[n]-h_n^T\tilde{w}_n\right)^2$$
$$\frac{\partial\hat{s}^2[n]}{\partial h_n}=2\left(s[n]+w[n]-h_n^T\tilde{w}_n\right)(-\tilde{w}_n)$$
$$=2\hat{s}[n](-\tilde{w}_n)=-2\hat{s}[n]\tilde{w}_n$$

Adaptive Interference Cancellation



$$\hat{s}[n]=x[n]-\mathbf{h}_n^T\tilde{\mathbf{w}}_n$$
$$=s[n]+\{w[n]-\mathbf{h}_n^T\tilde{\mathbf{w}}_n\}$$

uncorrelated

Minimize *instantaneous power* $\hat{s}^2[n]$ for best interference cancellation

$$\frac{d\left(\hat{s}[n]\right)^2}{d\,\mathbf{h}_n}=-2\hat{s}[n]\tilde{\mathbf{w}}_n$$

$$\boxed{\mathbf{h}_{n+1}=\mathbf{h}_n+2\mu\,\hat{s}[n]\tilde{\mathbf{w}}_n}$$

Stability of LMS

- The LMS algorithm is convergent in the mean square if and only if the step-size parameter satisfy

$$0<\mu<\frac{2}{\lambda_{max}}$$

- Here λ_{max} is the largest eigenvalue of the correlation matrix of the input data
- More practical test for stability is

$$0<\mu<\frac{2}{\text{input signal power}}$$

- Larger values for step size
  - Increases adaptation rate (faster adaptation)
  - Increases residual mean-squared error

Big Ideas

- Linear vs. Circular Convolution
  - Use circular convolution (i.e DFT) to perform fast linear convolution
    - Overlap-Add, Overlap-Save
  - Circular convolution is linear convolution with aliasing
- Adaptive Filters
  - Use LMS algorithm to update filter coefficients
  - applications like system ID, channel equalization, and signal prediction

Admin

- Project
  - Out now
  - Work in pairs
  - Due 4/28
- Final Exam – Tentative 5/7
  - Administered in Canvas
    - 2hr timed exam in 8hr window
  - Open notes
  - Random set of questions
  - Covers lec 1-20