Lecture 9: Exam 1 Review

Mark Hasegawa-Johnson

ECE 417: Multimedia Signal Processing, Fall 2021

Outline

Topics

1. HW1: Signal Processing Review
2. MP1: LPC
3. HW2: Linear Algebra
4. MP2: Image Processing & Optical Flow

Outline

DTFT

$$X(\omega) = \sum_{n=-\infty}^{\infty} x[n] e^{-j\omega n}$$

$$x[n] = \frac{1}{2\pi} \int_{-\pi}^{\pi} X(\omega) e^{j\omega n}$$

Frequency Response

$$Y(\omega) = H(\omega)X(\omega)$$

$$y[n] = h[n] * x[n]$$

Z Transform

$$\sum_{m=0}^{M-1} b_m x[n-m] = \sum_{k=0}^{N-1} a_k y[n-k]$$

$$H(z) = \frac{\sum_{m=0}^{M-1} b_m z^{-m}}{\sum_{k=0}^{N-1} a_k z^{-k}}$$

Frequency Response

$$H(\omega) = H(z = e^{j\omega})$$

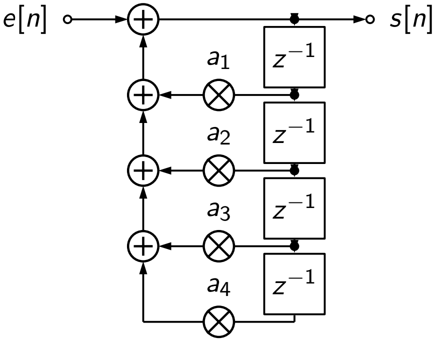
Outline

All-Pole Filter

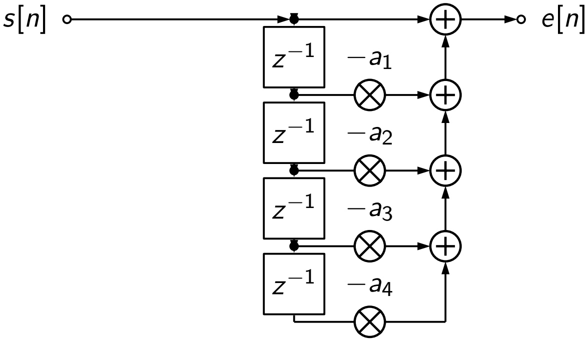
$$H(z) = \frac{1}{1-\sum_{k=1}^{N} a_k z^{-k}} = \frac{1}{\prod_{k=1}^{N}\left(1-p_k z^{-1}\right)} = \sum_{k=1}^{N} \frac{C_k}{1-p_k z^{-1}}$$

$$h[n] = \sum_{k=1}^{N} C_k p_k^n u[n]$$

Linear Predictive Synthesis Filter



Linear Predictive Analysis Filter



Finding the Linear Predictive Coefficients

Formulate the problem like this: we want to find a_k in order to minimize:

$$\mathcal{E} = \sum_{n=-\infty}^{\infty} e^2[n] = \sum_{n=-\infty}^{\infty} \left(s[n] - \sum_{m=1}^{p} a_m s[n-m] \right)^2$$

If we set $d\mathcal{E}/da_k = 0$, we get

$$0 = \sum_{n=-\infty}^{\infty} \left(s[n] - \sum_{m=1}^{p} a_m s[n-m] \right) s[n-k] = \sum_{n=-\infty}^{\infty} e[n]s[n-k]$$

which we sometimes write as $e[n] \perp s[n-k]$

Autocorrelation

In order to write the solution more easily, let's define something called the "autocorrelation," $R[m]$:

$$R[m] = \sum_{n=-\infty}^{\infty} s[n]s[n-m]$$

In terms of the autocorrelation, the orthogonality equations are

$$0 = R[k] - \sum_{m=1}^{p} a_m R[k-m] \quad \forall\ 1 \le k \le p$$

which can be re-arranged as

$$R[k] = \sum_{m=1}^{p} a_m R[k-m] \quad \forall\ 1 \le k \le p$$

Matrices

Since we have p linear equations in p unknowns, let's create matrices:

$$\vec{\gamma} = \begin{bmatrix} R[1] \\ R[2] \\ \vdots \\ R[p] \end{bmatrix}, \quad R = \begin{bmatrix} R[0] & R[1] & \cdots & R[p-1] \\ R[1] & R[0] & \cdots & R[p-2] \\ \vdots & \vdots & \ddots & \vdots \\ R[p-1] & R[p-2] & \cdots & R[0] \end{bmatrix}.$$

Then the normal equations become

$$\vec{\gamma} = R\vec{a}$$

and their solution is

$$\vec{a} = R^{-1}\vec{\gamma}$$

Outline

Linear Algebra Review

- A linear transform, A, maps vectors in space $\vec{x}$ to vectors in space $\vec{y}$.
- The determinant, $|A|$, tells you how the volume of the unit sphere is scaled by the linear transform.
- Every $D \times D$ linear transform has D eigenvalues, which are the roots of the equation $|A - \lambda I| = 0$.
- Left and right eigenvectors of a matrix are either orthogonal ($\vec{u}_i^T \vec{v}_j = 0$) or share the same eigenvalue ($\kappa_i = \lambda_j$).
- For a symmetric matrix, the left and right eigenvectors are the same. If the eigenvalues are distinct and real, then:

$$A = V\Lambda V^T, \quad \Lambda = V^T A V, \quad VV^T = V^T V = I$$

Pseudo-Inverse

If A is a tall thin matrix, then there is usually no vector $\vec{v}$ that solves $\vec{b} = A\vec{v}$, but $\vec{v} = A^\dagger\vec{b}$ is the vector that comes closest, in the sense that

$$A^\dagger\vec{b} = \text{argmin}_{\vec{v}}\|\vec{b} - A\vec{v}\|^2$$

If we differentiate the norm, and set the derivative to zero, we get

$$A^\dagger = (A^T A)^{-1} A^T$$

Outline

What is a Multidimensional Signal?

A multidimensional signal is one that can be indexed in many directions. For example, a typical video that you would play on your laptop is a 4-dimensional signal, $x[k, t, r, c]$:

- k indexes color ($k = 0$ for red, $k = 1$ for green, $k = 2$ for blue)
- t is the frame index
- r is the row index
- c is the column index

If there are 3 colors, 30 frames/second, 480 rows and 640 columns, with one byte per pixel, then that's $3 \times 30 \times 480 \times 640 = 27684000$ bytes/sec.

Multidimensional Convolution

Any linear, shift-invariant system can be implemented as a convolution. 2D convolution is defined as

$$\begin{aligned} y[n_1,n_2] &= x[n_1,n_2] * h[n_1,n_2] \\ &= \sum_{m_1=-\infty}^{\infty} \sum_{m_2=-\infty}^{\infty} x[m_1,m_2]h[n_1-m_1,n_2-m_2] \end{aligned}$$

The Fourier transform of convoluton is multiplication:

$$y[\vec{n}] = x[\vec{n}] * h[\vec{n}] \Leftrightarrow Y(\vec{\omega}) = H(\vec{\omega})X(\vec{\omega})$$

Separable Filters

A filter $h[n_1, n_2]$ is called "separable" if it can be written as

$$h[n_1, n_2] = h_1[n_1]h_2[n_2]$$

If a filter is separable, then the computational cost of convolution can be reduced by using separable convolution:

$$x[n_1, n_2] * h[n_1, n_2] = h_1[n_1] *_1 \left(h_2[n_2] *_2 x[n_1, n_2]\right)$$

Example: Image gradient

For example, we can compute image gradient using the filter

$$h[n] = 0.5\delta[n+1] - 0.5\delta[n-1]$$

then

$$\frac{\partial f}{\partial n_1} \approx h[n_1] *_1 f[n_1, n_2]$$
$$\frac{\partial f}{\partial n_2} \approx h[n_2] *_2 f[n_1, n_2]$$

Outline

Optical Flow

Definition: **optical flow** is the vector field $\vec{v}(t, r, c)$ specifying the current apparent velocity of the pixel at position (r, c). It depends on motion of (1) the object observed, and (2) the observer.

Then the optical flow equation is:

$$-\frac{\partial f}{\partial t} = (\nabla f)^T \vec{v}$$

The Lucas-Kanade Algorithm

The Lucas-Kanade algorithm solves the equation

$$\vec{b} = A\vec{v}$$

where

$$\vec{b} = -\left[\begin{array}{c} \frac{\partial f[t,r,c]}{\partial t} \\ \vdots \\ \frac{\partial f[t,r+H-1,c+W-1]}{\partial t} \end{array}\right], \quad \vec{v} = \left[\begin{array}{c} v_c[t,r,c] \\ v_r[t,r,c] \end{array}\right]$$

$$A = \left[\begin{array}{cc} \frac{\partial f[t,r,c]}{\partial c} & \frac{\partial f[t,r,c]}{\partial r} \\ \vdots & \\ \frac{\partial f[t,r+H-1,c+W-1]}{\partial c} & \frac{\partial f[t,r+H-1,c+W-1]}{\partial r} \end{array}\right]$$

Outline

Summary

1. HW1: Signal Processing Review
2. MP1: LPC
3. HW2: Linear Algebra
4. MP2: Image Processing & Optical Flow