Speech Signal Representations

Part 2: Speech Signal Processing

Hsin-min Wang

References:
1 X. Huang et al., Spoken Language Processing, Chapters 5-6
2 J. R. Deller et al., Discrete-Time Processing of Speech Signals, Chapters 4-6
3 J. W. Picone, “Signal modeling techniques in speech recognition,” proceedings of the IEEE, September 1993, pp. 1215-1247
Speech Recognition - Acoustic Processing

Speech Waveform

Framing

Signal Processing

Feature vector sequence

\[ o_1 o_2 o_3 o_4 \ldots o_t \ldots \]

\[ s_1 s_2 s_3 s_4 \ldots s_t \ldots \]

Hidden Markov Model

\[ S^* = \arg \max_S P(O \mid S) \]

\[ W^* = \arg \max_W P(O \mid W) \]

\[ a_{ij} = P(s_t = j \mid s_{t-1} = i) \]

\[ b_i(o_t) = P(o_t \mid s_t = i) \]

\[ = \sum_{k=1}^{M} c_{ik} N(o_t; \mu_{ik}; \Sigma_{ik}) \]

\[ S = \arg \max_{s} \{ P(s) \} \]

\[ O = \{ o_t \}_{t=1}^{T} \]

\[ W = \{ w_t \}_{t=1}^{T} \]
Source-Filter Model

Source-Filter model: decomposition of speech signals

- A source passed through a *linear-time-varying* filter
- **Source** (excitation): the air flow at the vocal cords (聲帶)
- **Filter**: the resonances (共鳴) of the vocal tract (聲道) which change over time
- Once the filter has been estimated, the source can be obtained by passing the speech signal through the inverse filter

\[ e[n] \xrightarrow{} h[n] \xrightarrow{} x[n] \]
Source-Filter Model (cont.)

Phoneme classification is mostly dependent on the characteristics of the filter

- **Speech recognizers** estimate the filter characteristics and ignore the source
  - **Speech production model**: *linear prediction coding* and *cepstral analysis*
  - **Speech perception model**: *mel-frequency cepstrum*

- **Speech synthesis techniques** use a source-filter model because it allows flexibility in altering the pitch and the filter

- **Speech coders** use a source-filter model because it allows a low bit rate
Characteristics of the Source-Filter Model

The characteristics of the vocal tract define the uttered phoneme

- Such characteristics are evidenced in the frequency domain by the location of the formants, i.e., the peaks given by resonances of the vocal tract
Main Considerations in Feature Extraction

- **Perceptually Meaningful**
  - Parameters represent salient aspects of the speech signal
  - Parameters are analogous to those used by human auditory system (perceptually meaningful)

- **Robust Parameters**
  - Parameters are robust to variations in environments such as the channels, speakers, and transducers

- **Time-Dynamic Parameters**
  - Parameters can capture spectral dynamics, or changes of the spectrum with time (temporal correlation)
Typical Procedures for Feature Extraction

Speech Signal

A/D Conversion → Pre-emphasis → Framing and Windowing → Spectral Shaping

Cepstral Processing → Measurements

Fourier Transform Filter Bank or Linear Prediction (LP) → Spectral Analysis

Conditioned Signal

Parameters

Parametric Transform
Spectral Shaping

- **A/D Conversion**
  - Conversion of the signal from a sound pressure wave to a digital signal
  - Sampling

- **Digital Filtering (Pre-emphasis)**
  - Emphasizing important frequency components in the signal

- **Framing and Windowing**
  - Short-time processing

*Figure 5.2 Analog signal and its corresponding digital signal.*
A/D Conversion

Undesired side effects of A/D conversion
- Line frequency noise (50/60-Hz hum)
- Loss of low- and high-frequency information
- Nonlinear input-output distortion
- Example:
  - Frequency response of a typical telephone grade A/D converter
  - The sharp attenuation of low frequency and high frequency response causes problem for subsequent parametric spectral analysis algorithms

The most popular sampling frequency
- Telecommunication: 8kHz
- Non-telecommunication: 10~16kHz

Fig. 3. The frequency response of a typical telephone grade A/D converter is shown.
## Sampling Frequency vs. Recognition Accuracy

Table 9.1 Relative error rate reduction with different sampling rates. The reduction is relative to that of the preceding row.

<table>
<thead>
<tr>
<th>Sampling Rate</th>
<th>Relative Error-Rate Reduction</th>
</tr>
</thead>
<tbody>
<tr>
<td>8 kHz</td>
<td>Baseline</td>
</tr>
<tr>
<td>11 kHz</td>
<td>+10%</td>
</tr>
<tr>
<td>16 kHz</td>
<td>+10%</td>
</tr>
<tr>
<td>22 kHz</td>
<td>+0%</td>
</tr>
</tbody>
</table>
Pre-emphasis
Pre-emphasis

The pre-emphasis filter

- A FIR high-pass filter
- A first-order finite impulse response filter is widely used

\[ H_{\text{pre}}(z) = \sum_{k=0}^{N_{\text{pre}}} a_{\text{pre}}(k)z^{-k} \]

\[ H_{\text{pre}}(z) = 1 - a_{\text{pre}}z^{-1} \]

- \( a_{\text{pre}} \): values close to 1.0 that can be efficiently implemented in fixed point hardware, such as \(-1\) or \(-1/16\), are most common
- Boost the signal spectrum approximately 20 dB per decade

Speech signal \( x[n] \)

\( x'[n] = x[n] - ax[n-1] \)

\( H(z) = 1 - a \cdot z^{-1}, \quad 0 < a \leq 1 \)
Why Pre-emphasis?

- **Reason 1: Eliminate the glottal formants**
  - The component of the glottal signal can be modeled by a simple two-real-pole filter whose poles are near $z=1$
  - The lip radiation characteristic, with its zero near $z=1$, tends to cancel the spectral effects of one of the glottal pole
  - By introducing a second zero near $z=1$ (pre-emphasis), we can eliminate effectively the larynx and lips spectral contributions

  $\Rightarrow$ **Analysis can be asserted to be seeking the parameters corresponding to the vocal tract only**

\[
\begin{align*}
   u[n] & \quad \rightarrow \quad G[z] = \frac{1}{1-h_z z^{-1}} \cdot \frac{1}{1-h_z z^{-1}} \quad \rightarrow \quad u_G[n] \\
   & \quad \quad \text{glottal signal} \\
   & \quad \quad \text{vocal tract} \\
   & \quad \rightarrow \quad H(z) \rightarrow \quad 1-cz^{-1} \quad \rightarrow \quad x[n] \\
   & \quad \quad \text{lip}
\end{align*}
\]
Why Pre-emphasis? (cont.)

- **Reason 2: Prevent Numerical Instability**
  - If the speech signal is dominated by low frequencies, it is highly predictable and a large LP model will result in an ill-conditioned autocorrelation matrix

- **Reason 3:**
  - Voiced sections of the speech signal naturally have a negative spectral slope (attenuation) of approximately 20 dB per decade due to physiological characteristics of the speech production system
  - High frequency formants have small amplitude with respect to low frequency formants. A pre-emphasis of high frequencies is therefore required to obtain similar amplitude for all formants
Why Pre-emphasis? (cont.)

- **Reason 4:**
  - Hearing is more sensitive above the 1 kHz region of the spectrum
  - The pre-emphasis filter amplifies this most perceptually important area of the spectrum

![Diagram showing the sound pressure level (SPL) level in dB as a function of frequency.]
Framing and Windowing
Short-Time Fourier Analysis

- **Spectral Analysis**

- **Spectrogram Representation**
  - A spectrogram of a time signal is a two-dimension representation that displays *time* in its horizontal axis and *frequency* in its vertical axis.
  - A gray scale is typically used to indicate the energy at each point (t,f)
    - “white”: low energy
    - “black”: high energy
Framing and Windowing

- Short-time-analysis by framing: decompose the speech signal into a series of overlapping frames
  - Traditional methods for spectral evaluation are reliable in the case of a stationary signal (i.e., a signal whose statistical characteristics are invariant with respect to time)
    - The frame has to be short enough for the behavior (periodicity or noise-like appearance) of the signal to be approximately constant or assumed stationary
      - the signal characteristics (whether periodicity or noise-like appearance) are uniform in that region

- Terminology
  - **Frame Duration** (N): the length of time over which a set of parameters is valid, typically on the order of 20 ~ 30 ms
  - **Frame Period** (L): the length of time between successive parameter calculations (Target Rate)
  - **Frame Rate**: the number of frames computed per second
Framing and Windowing (cont.)

- Given a speech signal $x[n]$, we define the short-time signal $x_m[n]$ of frame $m$ as the product of $x[n]$ by a window function $w_m[n]$

$$x_m[n] = x[n]w_m[n]$$

- $w_m[n] = w[m-n]$ where $w[n]=0$ for $|n|>N/2$
  - In practice, the window length $N$ is on the order of 20 to 30

- The short-time Fourier representation for frame $m$ is defined as

$$X_m(e^{jw}) = \sum_{n=-\infty}^{\infty} x_m[n]e^{-jwn} = \sum_{n=-\infty}^{\infty} w[m-n]x[n]e^{-jwn}$$
Framing and Windowing (cont.)

- **Rectangular window**
  - \( w[n] = 1 \) for \( 0 \leq n \leq N-1 \)
  - Just extract the frame part of signal without further processing
  - Its frequency response has high side lobes

- **Main lobe**: spreads out in a wider frequency range the narrow band power of the signal, and thus reduces the local frequency resolution

- **Side lobe**: swaps energy from different and distant frequencies of \( x_m[n] \), which is called *spectral leakage*
Framing and Windowing (cont.)

\[ x[n] = \sum_{k=-\infty}^{\infty} \delta[n - kP] \quad \leftrightarrow \quad X(e^{jw}) = \frac{2\pi}{P} \sum_{k=0}^{N-1} \delta(w - 2\pi k / P) \]

Hamming window of length \( N \)

Main lobe width = \( 4 \pi / N \)
\[ \rightarrow N \geq 2P \]
Framing and Windowing (cont.)

The Hamming window offers less spectral leakage than the rectangular window.

The rectangular window provides better time resolution than the Hamming window.

Rectangular windows are rarely used for speech analysis despite their better time resolution.
Framing and Windowing (cont.)

- We want to select a window satisfy
  - the main lobe is as narrow as possible in its width
  - the side lobe is as low as possible in its magnitude
  
  *However, this is a trade-off!*

- In practice, the windows lengths are on the order of 20 to 30 ms
  - This choice is a compromise between stationarity assumption and the frequency resolution
Framing and Windowing (cont.)

- The Hamming window is most widely used

\[ w[n] = \begin{cases} 
0.54 - 0.46 \cos\left(\frac{2\pi n}{N-1}\right), & n = 0,1,\ldots, N-1 \\
0 & \text{otherwise}
\end{cases} \]
Framing and Windowing (cont.)

- Male Voiced Speech

![Figure 6.3](image)

**Figure 6.3** Short-time spectrum of male voiced speech (vowel /ah/ with local pitch of 110Hz): (a) time signal, spectra obtained with (b) 30 ms rectangular window and (c) 15 ms rectangular window, (d) 30 ms Hamming window, (e) 15 ms Hamming window. The window lobes are not visible in (e), since the window is shorter than 2 times the pitch period. Note the spectral leakage present in (b).
Framing and Windowing (cont.)

- Female Voiced Speech

Figure 6.4 Short-time spectrum of female voiced speech (vowel /aa/ with local pitch of 200Hz): (a) time signal, spectra obtained with (b) 30 ms rectangular window and (c) 15 ms rectangular window, (d) 30 ms Hamming window, (e) 15 ms Hamming window. In all cases the window lobes are visible, since the window is longer than 2 times the pitch period. Note the spectral leakage present in (b) and (c).
Framing and Windowing (cont.)

- Unvoiced Speech

No regularity is observed

Figure 6.5 Short-time spectrum of unvoiced speech: (a) time signal, (b) 30 ms rectangular window, (c) 15 ms rectangular window, (d) 30 ms Hamming window, (e) 15 ms Hamming window.
Linear Predictive Coding
Linear Predictive Coding

The theory of linear predictive coding (LPC), as applied to speech, has been well understood for many years

- LPC provides a good model of the speech signal
  - This is especially true for the quasi steady state regions of speech in which the all-pole model of LPC provides a good approximation to the vocal tract spectral envelope
  - During the unvoiced and transient regions of speech, the LPC model is less effective than for the voiced regions, but it still provides an acceptably useful model for speech recognition purpose
- The way in which LPC is applied to the analysis of speech signals leads to a reasonable source-vocal tract separation
  - A good representation of the vocal tract characteristics becomes possible
- LPC is an analytically tractable model
  - The method of LPC is mathematically precise and is simple and straightforward to implement in either software or hardware
- The LPC model works well in speech recognition
  - LPC front-end processing has been used in a large number of recognizers
The LPC Model

- The basic idea behind the LPC model is that a given speech sample at time \( n \), \( x[n] \), can be approximated as a linear combination of the past \( p \) speech samples, such that

\[
x[n] \approx a_1 x[n-1] + a_2 x[n-2] + \cdots + a_p x[n-p]
\]

where the coefficients \( a_1, a_2, \ldots, a_p \) are assumed constant over the speech analysis frame.

- By including an excitation term, \( Gu[n] \), \( x[n] \) can be expressed as

\[
x[n] = \sum_{k=1}^{p} a_k x[n-k] + Gu[n]
\]

- By expressing in the \( z \)-domain, we get

\[
X(z) = \sum_{k=1}^{p} a_k z^{-k} X(z) + GU(z)
\]

- The transfer function is

\[
H(z) = \frac{X(z)}{GU(z)} = \frac{1}{1 - \sum_{k=1}^{p} a_k z^{-k}} = \frac{1}{A(z)}
\]

- An all-pole filter with a sufficient number of poles is a good approximation to model the vocal tract (filter) for speech signals.
The LPC Model (cont.)

Figure 3.28 Speech synthesis model based on LPC model.
The LPC Model (cont.)

- Based on the LPC model, the exact relation between \( x[n] \) and \( u[n] \) is

\[
x[n] = \sum_{k=1}^{p} a_k x[n-k] + Gu[n]
\]

- We approximate \( x[n] \) as the linear combination of past speech samples

\[
\tilde{x}[n] = \sum_{k=1}^{p} a_k x[n-k]
\]

- The prediction error, \( e[n] \), is defined as

\[
e[n] = x[n] - \tilde{x}[n] = x[n] - \sum_{k=1}^{p} a_k x[n-k]
\]

- When \( x[n] \) is actually generated by a LPC model, then the prediction error \( e[n] \) will equal to \( Gu[n] \)

- The basic problem of linear prediction analysis is to determine the set of predictor coefficients, \( \{a_k\} \), directly from the speech signal

  - Since the spectral characteristics of speech vary over time, the basic approach is to find a set of predictor coefficients that minimize the mean-square prediction error over a short segment of the speech waveform
To estimate the LPC coefficients from a set of speech samples, we use the short-term analysis technique

\[ E_m = \sum_n e_m^2[n] = \sum_n (x_m[n] - \tilde{x}_m[n])^2 \]

Framing/Windowing,
The short-term prediction error for a specific frame \(m\)

We can estimate the LPC coefficients as those that Minimize the total prediction error

\[ \sum_n \left( x_m[n] - \sum_{j=1}^p a_j x_m[n-j] \right)^2 \]

Take the derivative

\[ \frac{\partial E_m}{\partial a_i} = \frac{\partial}{\partial a_i} \left[ \sum_n \left( x_m[n] - \sum_{j=1}^p a_j x_m[n-j] \right)^2 \right] = 0, \quad 1 \leq i \leq p \]

\[ \sum_n \left( x_m[n] - \sum_{j=1}^p a_j x_m[n-j] \right) x_m[n-i] = 0, \quad 1 \leq i \leq p \]

\[ \langle e_m, x_m^i \rangle = \sum_n e_m[n] x_m[n-i] = 0, \quad 1 \leq i \leq p \]

**orthogonality principle:** The error vector is orthogonal to the past vectors
LPC – the Yule-Walker Equations

\[ \sum_{n} \left[ \left( x_{m}[n] - \sum_{j=1}^{p} a_{j} x_{m}[n-j] \right) x_{m}[n-i] \right] = 0, \ 1 \leq i \leq p \]

\[ \Rightarrow \sum_{n} \left[ \sum_{j=1}^{p} \left( a_{j} x_{m}[n-i] x_{m}[n-j] \right) \right] = \sum_{n} (x_{m}[n-i] x_{m}[n]), \ 1 \leq i \leq p \]

\[ \Rightarrow \sum_{j=1}^{p} \left[ a_{j} \sum_{n} (x_{m}[n-i] x_{m}[n-j]) \right] = \sum_{n} (x_{m}[n-i] x_{m}[n]), \ 1 \leq i \leq p \]

Define correlation coefficients:

\[ \phi_{m}[i, j] = \sum_{n} (x_{m}[n-i] x_{m}[n-j]) \]

\[ \Rightarrow \sum_{j=1}^{p} a_{j} \phi_{m}[i, j] = \phi_{m}[i, 0], \ 1 \leq i \leq p \]

The Yule-Walker equations: Solution of the set of \( p \) linear equations results in the \( p \) LPC coefficients that minimize the prediction error.
LPC – the Minimum Mean-Squared Error

\[ E_m = \sum_n e_n^2 = \sum_n (x_m[n] - \bar{x}_m[n])^2 = \sum_n \left( x_m[n] - \sum_{j=1}^{p} a_j x_m[n - j] \right)^2 \]

\[ = \sum_n x_m^2[n] - 2 \sum_n \left( x_m[n] \sum_{j=1}^{p} a_j x_m[n - j] \right) + \sum_n \left( \sum_{j=1}^{p} a_j x_m[n - j] \sum_{k=1}^{p} a_k x_m[n - k] \right) \]

\[ \sum_n \left( \sum_{j=1}^{p} a_j x_m[n - j] \sum_{k=1}^{p} a_k x_m[n - k] \right) \]

\[ = \sum_{j=1}^{p} a_j \left\{ \sum_{k=1}^{p} a_k \sum_n (x_m[n - j] x_m[n - k]) \right\} \]

\[ = \sum_{j=1}^{p} a_j \sum_n x_m[n - j] x_m[n] \]

\[ E_m = \sum_n x_m^2[n] - \sum_{j=1}^{p} a_j \sum_n (x_m[n] x_m[n - j]) = \phi_m[0,0] - \sum_{j=1}^{p} a_j \phi_m[0, j] \]
LPC – Solution of the LPC Equations

\[ x_m[n] = \sum_{k=1}^{p} a_k x_m[n-k] + e_m[n], \quad 0 \leq n \leq N-1 \]

\[
\begin{bmatrix}
  x_m[-1] & x_m[-2] & \ldots & x_m[-p] \\
  x_m[1-1] & x_m[1-2] & \ldots & x_m[1-p] \\
  \vdots & \vdots & \ddots & \vdots \\
  x_m[N-1-1] & x_m[N-1-2] & \ldots & x_m[N-1-p]
\end{bmatrix}
\begin{bmatrix}
  a_1 \\
  a_2 \\
  \vdots \\
  a_p
\end{bmatrix}
= \begin{bmatrix}
  e_m[0] \\
  e_m[1] \\
  \vdots \\
  e_m[N-1]
\end{bmatrix} + \begin{bmatrix}
  x_m[0] \\
  x_m[1] \\
  \vdots \\
  x_m[N-1]
\end{bmatrix}
\]

\[ X = \left[ x_m^1 \ x_m^2 \ldots \ x_m^P \right] \]

\[ Xa + e_m = x_m \]

\[ E_m = \sum_n e_m^2[n] \text{ is minimal if } X^T e_m = 0 \]

\[ \Rightarrow X^T (x_m - Xa) = 0 \Rightarrow X^T Xa = X^T x_m \Rightarrow a = \left( X^T X \right)^{-1} X^T x_m \]

The solution can be achieved with any standard matrix inverse package. Because of the special form of the matrix here, some efficient solutions are possible; e.g. the **autocorrelation method**, the **covariance method**, and the lattice method.
LPC – *the Covariance Method*

One way to solve for the LPC coefficients is to fix the interval over which the mean-squared error is computed to the range $0 \leq n \leq N-1$ and to use the unweighted speech directly; i.e.,

$$E_m = \sum_{m=0}^{N-1} e_m^2[n] = \sum_{m=0}^{N-1} (x_m[n] - \tilde{x}_m[n])^2$$

![Diagram showing the process of shifting and comparing the speech signal](image-url)
\[ \phi_m[i, j] = \sum_{n=0}^{N-1} x_m[n - i]x_m[n - j] \]
\[ = \sum_{n=0}^{N-1} x_m[n - i]x_m[n - j] \]
\[ = \sum_{n=-i}^{N-1} x_m[n]x_m[n + (i - j)] \]
\[ = \sum_{n=-j}^{N-1} x_m[n]x_m[n + (j - i)] \]
\[ = \phi_m[j, i] \]
LPC — the Covariance Method (cont.)

\[ \sum_{j=1}^{P} a_j \phi_m[i, j] = \phi_m[i, 0], \quad i = 1, 2, \ldots, p \]

and \( \phi_m[i, j] = \phi_m[j, i] \)

\[
\begin{bmatrix}
\phi_m[1,1] & \phi_m[1,2] & \phi_m[1,3] & \ldots & \phi_m[1,p] \\
\phi_m[2,1] & \phi_m[2,2] & \phi_m[2,3] & \ldots & \phi_m[2,p] \\
\phi_m[3,1] & \phi_m[3,2] & \phi_m[3,3] & \ldots & \phi_m[3,p] \\
\vdots & \vdots & \vdots & \ddots & \vdots \\
\phi_m[p,1] & \phi_m[p,2] & \phi_m[p,3] & \ldots & \phi_m[p,p]
\end{bmatrix}
\begin{bmatrix}
a_1 \\
a_2 \\
a_3 \\
\vdots \\
a_p
\end{bmatrix}
=
\begin{bmatrix}
\phi_m[1,0] \\
\phi_m[2,0] \\
\phi_m[3,0] \\
\vdots \\
\phi_m[p,0]
\end{bmatrix}
\]

\[ \Rightarrow \Phi a = \psi \quad \Phi: \text{symmetric and positive definite} \]

\[ x^T \Phi x > 0 \text{ for all nonzero vectors } x \in \mathbb{R}^p \]
LPC – *the Covariance Method* (cont.)

\[ \Phi a = \psi \]

The matrix \( \Phi \) is expressed as \( \Phi = V D V^t \)

where \( V \) is a lower triangular matrix (whose main diagonal elements are 1's), and \( D \) is a diagonal matrix.

\[
\begin{bmatrix}
\Phi_{11} & \Phi_{21} & \Phi_{31} \\
\Phi_{21} & \Phi_{22} & \Phi_{32} \\
\Phi_{31} & \Phi_{32} & \Phi_{33}
\end{bmatrix}
= \begin{bmatrix}
1 & 0 & 0 \\
V_{21} & 1 & 0 \\
V_{31} & V_{32} & 1
\end{bmatrix}
\times
\begin{bmatrix}
d_1 & 0 & 0 \\
0 & d_2 & 0 \\
0 & 0 & d_3
\end{bmatrix}
\times
\begin{bmatrix}
1 & V_{21} & V_{31} \\
0 & 1 & V_{32} \\
0 & 0 & 1
\end{bmatrix}
\]

\[
= \begin{bmatrix}
d_1 & 0 & 0 \\
d_1 V_{21} & d_2 & 0 \\
d_1 V_{31} & d_2 V_{32} & d_3
\end{bmatrix}
\times
\begin{bmatrix}
1 & V_{21} & V_{31} \\
0 & 1 & V_{32} \\
0 & 0 & 1
\end{bmatrix}
\]

\[
= \begin{bmatrix}
d_1 \\
0 \\
0
\end{bmatrix}
\begin{bmatrix}
d_1 V_{21} \\
0 \\
0
\end{bmatrix}
+ \begin{bmatrix}
d_2 \\
0 \\
0
\end{bmatrix}
\begin{bmatrix}
d_1 V_{21} V_{31} + d_2 \\
d_1 V_{31} V_{31} + d_2 V_{32} \\
d_1 V_{31} V_{31} + d_2 V_{32} + d_3
\end{bmatrix}
\]
LPC – *the Covariance Method* (cont.)

\[ \Phi a = \psi \]

The matrix \( \Phi \) is expressed as \( \Phi = VDV^t \)
where \( V \) is a lower triangular matrix (whose main diagonal elements are 1's), and \( D \) is a diagonal matrix.

So each element of \( \Phi \) can be expressed as
\[ \phi[i, j] = \sum_{k=1}^{j} V_{ik} d_k V_{jk} \quad 1 \leq j < i \]
or alternatively
\[ V_{ij} d_j = \phi[i, j] - \sum_{k=1}^{j-1} V_{ik} d_k V_{jk} \quad 1 \leq j < i \]
Eq2

and for the diagonal elements
\[ \phi[i, i] = \sum_{k=1}^{i} V_{ik} d_k V_{ik} \]

or alternatively
\[ d_i = \phi[i, i] - \sum_{k=1}^{i-1} V_{ik}^2 d_k \quad i \geq 2 \quad \text{with} \quad d_1 = \phi[1,1] \]
Eq3

The Cholesky decomposition starts with Eq1 then alternates between Eq 2 and Eq3 to solve \( V \) and \( D \)
LPC – *the Covariance Method* (cont.)

\[ \Phi \mathbf{a} = \mathbf{\psi} \text{ where } \Phi = \mathbf{VDV}^t \]

Once \( \mathbf{V} \) and \( \mathbf{D} \) have been determined \( \Rightarrow \mathbf{VDV}^t \mathbf{a} = \mathbf{\psi} \)

\( \Rightarrow \mathbf{VY} = \mathbf{\psi} \), where \( \mathbf{Y} = \mathbf{DV}^t \mathbf{a} \) or alternatively \( \mathbf{V}^t \mathbf{a} = \mathbf{D}^{-1} \mathbf{Y} \)

Given matrix \( \mathbf{V} \), \( \mathbf{Y} \) can be solved recursively as

\[ Y_i = \psi_i - \sum_{j=1}^{i-1} V_{ij} Y_j, \ 2 \leq i \leq p, \text{ with the initial condition } Y_1 = \psi_1 \]

Having determined \( \mathbf{Y} \), \( \mathbf{a} \) can be solved recursively as

\[ a_i = Y_i / d_i - \sum_{j=i+1}^{p} V_{ji} a_j, \ 1 \leq i < p, \text{ with the initial condition } a_p = Y_p / d_p \]

where the index \( i \) proceeds backwards
LPC – the Autocorrelation Method

Assume $x_m[n]$ identically zero outside the interval $0 \leq n \leq N-1$

$$x_m[n] = \begin{cases} x[n + mL]w[n], & 0 \leq n \leq N - 1 \\ 0, & \text{otherwise} \end{cases}$$

$L$: Frame Period, the length of time between successive frames

$\tilde{x}_m[n] = x[n + mL]$  

windowing

$$x_m[n] = \tilde{x}_m[n]w[n]$$
LPC – the Autocorrelation Method (cont.)

The mean-squared error is

\[ E_m = \sum_{m=0}^{N-1+p} e_m^2[n] = \sum_{m=0}^{N-1+p} (x_m[n] - \tilde{x}_m[n])^2 \]

\[
\Rightarrow \sum_{j=1}^{p} a_j \phi_m[i, j] = \phi_m[i, 0], \quad 1 \leq i \leq p
\]

\[
\phi_m[i, j] = \sum_{n=0}^{N+p-1} x_m[n-i]x_m[n-j]
\]

\[
= \sum_{n=i}^{N-1+j} x_m[n-i]x_m[n-j]
\]

\[
= \sum_{n=0}^{N-1-(i-j)} x_m[n]x_m[n+(i-j)]
\]
Define the autocorrelation function of \( x_m[n] \) as

\[
R_m[k] = \sum_{n=0}^{N-1-k} x_m[n] x_m[n+k]
\]

\[
\phi_m[i, j] = R[i - j]
\]

\[
\phi_m[i, j] = \sum_{n=0}^{N-1-(i-j)} x_m[n] x_m[n + (i-j)]
\]

Then

\[
\sum_{j=1}^{p} a_j \phi_m[i, j] = \phi_m[i, 0], \quad i = 1, 2, \ldots, p
\]

\[
R_m[k] = R_m[-k]
\]

\[
\Rightarrow \sum_{j=1}^{p} a_j R_m[i - j] = R_m[i], \quad i = 1, 2, \ldots, p
\]

\[
\begin{bmatrix}
R_m[0] & R_m[1] & R_m[2] & \cdots & R_m[p-1] \\
R_m[1] & R_m[0] & R_m[1] & \cdots & R_m[p-2] \\
\vdots & \vdots & \vdots & \ddots & \vdots \\
R_m[p-1] & R_m[p-2] & R_m[p-3] & \cdots & R_m[0]
\end{bmatrix}
\begin{bmatrix}
a_1 \\
a_2 \\
a_3 \\
\vdots \\
a_p
\end{bmatrix}
= 
\begin{bmatrix}
R_m[1] \\
R_m[2] \\
R_m[3] \\
\vdots \\
R_m[p]
\end{bmatrix}
\]

A Toeplitz Matrix: symmetric and all the elements in the diagonals are identical
Levinson-Durbin Recursion

- 1. Initialization \( E(0) = R_m[0] \)

- 2. Iteration. For \( i=1, \ldots, p \) do the following recursion

\[
k(i) = \frac{R_m[i] - \sum_{j=1}^{i-1} a_j(i-1)R_m[i-j]}{E(i-1)}
\]

\[
a_i(i) = k(i)
\]

\[
a_j(i) = a_j(i-1) - k(i)a_{i-j}(i-1), \quad \text{for} \quad 1 \leq j \leq i-1
\]

\[
E(i) = (1 - [k(i)]^2)E(i-1)
\]

- 3. Final Solution:

\[
a_j = a_j(p) \quad \text{for} \quad 1 \leq j \leq p
\]

Presentation topic:
The derivation of the Levinson-Durbin recursion and lattice formulation.
LPC – the Autocorrelation Method (cont.)

\[ E(0) = R_m[0] \]

\[ R_m[0]a_1(1) = R_m[1] \Rightarrow a_1(1) = \frac{R_m[1]}{R_m[0]}, \quad E(1) = R_m[0] - \frac{R_m[1]}{R_m[0]} \times R_m[1] \]

\[ \begin{bmatrix} R_m[0] & R_m[1] \\ R_m[1] & R_m[0] \end{bmatrix} \begin{bmatrix} a_1(2) \\ a_2(2) \end{bmatrix} = \begin{bmatrix} R_m[1] \\ R_m[2] \end{bmatrix} \]

\[ a_2(2) = \frac{R_m[0]R_m[2] - R_m[1]^2}{R_m[0]^2 - R_m[1]^2} = \frac{R_m[2] - \frac{R_m[1]^2}{R_m[0]}}{R_m[0]^2 - R_m[1]^2} = \frac{R_m[2] - a_1(1) \times R_m[1]}{E(1)} = k(2) \]

\[ a_1(2) = \frac{R_m[1]R_m[0] - R_m[1]R_m[2]}{R_m[0]^2 - R_m[1]^2} \]

\[ a_1(1) - k(2) \times a_1(1) = \frac{R_m[1]}{R_m[0]} - \frac{R_m[0]R_m[2] - R_m[1]^2}{R_m[0]^2 - R_m[1]^2} \times \frac{R_m[1]}{R_m[0]} = \frac{R_m[1]R_m[0] - R_m[1]R_m[2]}{R_m[0]^2 - R_m[1]^2} = a_1(2) \]

\[ E(2) = R_m[0] - a_1(2)R_m[1] - a_2(2)R_m[2] \]

\[ E(2) = \left(1 - [k(2)]^2\right)E(1) = \left(1 - a_2(2)\left(\frac{R_m[0]R_m[2] - R_m[1]^2}{R_m[0]^2 - R_m[1]^2}\right)\right) \left(R_m[0] - \frac{R_m[1]}{R_m[0]} \times R_m[1]\right) \]

\[ = R_m[0] - \frac{R_m[1]^2}{R_m[0]} - a_2(2) \left(\frac{R_m[0]R_m[2] - R_m[1]^2}{R_m[0]^2 - R_m[1]^2}\right) \left(\frac{R_m[0]^2 - R_m[1]^2}{R_m[0]}ight) \]

\[ = R_m[0] - a_2(2)R_m[2] - \left(\frac{R_m[1]^2}{R_m[0]} + \frac{R_m[0]R_m[2] - R_m[1]^2}{R_m[0]^2 - R_m[1]^2}\right) \left(\frac{R_m[1]^2}{R_m[0]}\right) = R_m[0] - a_2(2)R_m[2] - a_1(2)R_m[1] \]
Spectral Analysis via LPC

- LPC spectrum matches more closely the peaks than the valleys

\[ E_m = \sum_{n=0}^{N-1+p} e_n^2[n] = \frac{1}{2\pi} \int_{-\pi}^{\pi} \left| E_m(e^{j\omega}) \right|^2 d\omega = G^2 = \frac{1}{2\pi} \int_{-\pi}^{\pi} \left| \frac{X_m(e^{j\omega})}{H(e^{j\omega})} \right|^2 d\omega \]

\[ H'(e^{j\omega}) = G \cdot H(e^{j\omega}) \]

The higher \( p \), the more details of the spectrum are preserved.

Figure 6.20 LPC spectrum of the /ahl/ phoneme in the word lives of Figure 6.3. Used here are a 30-ms Hamming window and the autocorrelation method with \( p = 14 \). The short-time spectrum is also shown.

- Because the regions where \( \left| X_m(e^{j\omega}) \right| > \left| H(e^{j\omega}) \right| \) contribute more to the error than those where \( \left| H(e^{j\omega}) \right| > \left| X_m(e^{j\omega}) \right| \)

Figure 6.21 LPC spectra of Figure 6.20 for various values of the predictor order \( p \).
If $p$ is large enough, we can approximate the signal spectrum with arbitrarily small error.
The Prediction Error

- The prediction error signal is also called the *excitation* or *residual* signal
  \[ e[n] = x[n] - \sum_{k=1}^{p} a_k x[n-k] \]

- For unvoiced speech we expect the residual to be approximately white noise
  - In practice, this approximation is quite good

- For voiced speech we expect the residual to approximate an impulse train
  - In practice, this is not the case

**Figure 6.22** LPC prediction error signals for several vowels.
The Prediction Error (cont.)

- **How do we choose \( p \)?**
  - Larger values of \( p \) lead to lower prediction errors
  - Unvoiced speech has higher error than voiced speech because the LPC model is more accurate for voiced speech

- If we use a large value of \( p \), we are fitting the individual harmonics; thus the LPC filter is modeling the source, and the separation between source and filter is not going to be so good

*Figure 6.23* Variation of the normalized prediction error with the number of prediction coefficients \( p \) for the voiced segment of Figure 6.3 and the unvoiced speech of Figure 6.5. The autocorrelation method was used with a 30 ms Hamming window, and a sampling rate of 8 kHz.

Rule of Thumb: a pole per kHz plus 2-4 poles to model the radiation and glottal effects.
Cepstral Processing
The real cepstrum of a digital signal $x[n]$ is defined as
\[
c[n] = \frac{1}{2\pi} \int_{-\pi}^{\pi} \ln |X(e^{jw})|e^{jwn} \, dw
\]

The complex cepstrum of $x[n]$ is defined as
\[
\hat{x}[n] = \frac{1}{2\pi} \int_{-\pi}^{\pi} \ln X(e^{jw})e^{jwn} \, dw, \text{ where } \hat{X}(e^{jw}) = \ln X(e^{jw}) = \ln |X(e^{jw})| + j\angle X(e^{jw})
\]

Both real and complex cepstrum are homomorphic transformations.

If the signal $x[n]$ is real, both $c[n]$ and $\hat{x}[n]$ are real signals.

The term *cepstrum* is coined by reversing the first syllable of the word *spectrum*, given that it is obtained by taking the inverse Fourier transform of the log spectrum.

The term *quefrency* is defined to represent the independent variable $n$ in $c[n]$. The *quefrency* has dimension of time.
LPC-Cepstrum

Given the LPC filter

\[ H(z) = \frac{G}{1 - \sum_{k=1}^{p} a_k z^{-k}} = \frac{G}{\prod_{k=1}^{p} (1 - b_k z^{-1})} \]

\[ E(z)H(z) = X(z) \]

Take the logarithm

\[ \hat{H}(z) = \ln G - \sum_{k=1}^{p} \ln (1 - b_k z^{-1}) = \sum_{k=-\infty}^{\infty} \hat{h}[k] z^{-k} \]

\[ \hat{h}[k] : \text{complex cepstrum} \]

\[ \ln(1-x) = -\sum_{n=1}^{\infty} \frac{x^n}{n} \]

\[ \hat{H}(z) = \ln G - \sum_{k=1}^{p} \sum_{n=1}^{\infty} \frac{b_k^nz^{-n}}{n} = \ln G + \sum_{n=1}^{\infty} \sum_{k=1}^{p} \frac{b_k^nz^{-n}}{n} = \sum_{n=-\infty}^{\infty} \hat{h}[n] z^{-n} \]

Equate terms in \( z^{-n} \)

\[ \hat{h}[n] = \begin{cases} 
0 & n < 0 \\
\ln G & n = 0 \\
\sum_{k=1}^{p} \frac{b_k^n}{n} & n > 0 
\end{cases} \]

While there are a finite number of LPC coefficients, the number of cepstrum coefficients is infinite

Typically, a finite number of cepstrum coefficients are sufficient to approximate it
LPC-Cepstrum (cont.)

Given the LPC filter

\[ H(z) = \frac{G}{1 - \sum_{k=1}^{p} a_k z^{-k}} \]

Take the logarithm

\[ \hat{H}(z) = \ln G - \ln \left(1 - \sum_{l=1}^{p} a_l z^{-l}\right) = \sum_{k=-\infty}^{\infty} \hat{h}[k] z^{-k} \]

Take the derivative of both sides with respect to \( z \)

\[-\sum_{n=1}^{p} n a_n z^{-n-1} \frac{1}{1 - \sum_{l=1}^{p} a_l z^{-l}} = -\sum_{k=-\infty}^{\infty} k \hat{h}[k] z^{-k-1} \]

Multiply both sides by \( -z(1 - \sum_{l=1}^{p} a_l z^{-l}) \)

\[ \sum_{n=1}^{p} n a_n z^{-n} = \sum_{n=-\infty}^{\infty} n \hat{h}[n] z^{-n} - \sum_{l=1}^{p} \sum_{k=-\infty}^{\infty} k \hat{h}[k] a_l z^{-l-1} \]

Replace \( l = n-k \)

\[ 1 \leq n-k \leq p \]

\[ \sum_{n=1}^{p} n a_n z^{-n} = \sum_{n=-\infty}^{\infty} n \hat{h}[n] z^{-n} - \sum_{n=-\infty}^{\infty} \sum_{k=n-p}^{n-1} k \hat{h}[k] a_{n-k} z^{-n} \]

Equate terms in \( z^{-n} \)

\[ n a_n = n \hat{h}[n] - \sum_{k=1}^{n-1} k \hat{h}[k] a_{n-k}, \quad 0 < n \leq p \]

\[ 0 = n \hat{h}[n] - \sum_{k=n-p}^{n-1} k \hat{h}[k] a_{n-k}, \quad n > p \]
LPC-Cepstrum (cont.)

\[ na_n = n \hat{h}[n] - \sum_{k=1}^{n-1} k \hat{h}[k] a_{n-k}, \quad 0 < n \leq p \]

\[ 0 = n \hat{h}[n] - \sum_{k=n-p}^{n-1} k \hat{h}[k] a_{n-k}, \quad n > p \]

LPC-cepstrum

\[ \hat{h}[n] = \begin{cases} 
0 & n < 0 \\
\ln G & n = 0 \\
a_n + \sum_{k=1}^{n-1} \left( \frac{k}{n} \right) \hat{h}[k] a_{n-k} & 0 < n \leq p \\
\sum_{k=1}^{n-1} \left( \frac{k}{n} \right) \hat{h}[k] a_{n-k} & n > p 
\end{cases} \]

\[ \hat{H}(z) = \ln G - \ln \left( 1 - \sum_{i=1}^{p} a_i z^{-i} \right) \]

\[ = \ln G + \sum_{n=1}^{\infty} \frac{\left( \sum_{i=1}^{p} a_i z^{-i} \right)^n}{n} \]

\[ = \sum_{k=-\infty}^{\infty} \hat{h}[k] z^{-k} \]

\[ \hat{h}[0] = \ln G \]

The LPC coefficients can be computed recursively from the LPC coefficients.

\[ \ln(1-x) = -\sum_{n=1}^{\infty} \frac{x^n}{n} \]
Homomorphic Transform and Cepstral Processing

- A homomorphic transform $D(\cdot)$ is a transform that converts a convolution into a sum

$$x[n] = e[n] * h[n] \quad \hat{x}[n] = D(x[n]) = \hat{e}[n] + \hat{h}[n]$$

$x(n) = e(n) * h(n) \Rightarrow X(\omega) = E(\omega)H(\omega)$
$\Rightarrow |X(\omega)| = |E(\omega)||H(\omega)| \Rightarrow \log |X(\omega)| = \log |E(\omega)| + \log |H(\omega)|$

- Cepstrum is regarded as a homomorphic function that allows us to separate the source from the filter

  - We can find a value $N$ such that the cepstrum of the filter
    $$\hat{h}[n] \approx 0 \quad \text{for} \quad n \geq N$$
    and that the cepstrum of the excitation
    $$\hat{e}[n] \approx 0 \quad \text{for} \quad n < N$$

  ![Homomorphic filtering](image)

  **Figure 6.24** Homomorphic filtering to recover the filter’s response from a periodic signal. We have used the homomorphic filter of Eq. (6.102).
If we use a large value of $p$, the separation between source and filter is not going to be so good.

Why?
Mel-Frequency Cepstrum Coefficients (MFCC)
Mel-Frequency Cepstrum Coefficients (MFCC)

- Most widely used in the speech recognition
- Has generally obtained a better accuracy and a minor computational complexity
For each frame of signal (N points, e.g., N=512)
- The Discrete Fourier Transform (DFT) is first performed to obtain its spectrum (N points, for example N=512)
- A filterbank with M filters designed according to Mel scale is then applied, and each filter output is the sum of its filtered spectral components (M output values, for example M=24)

\[
X_a[k] = \sum_{n=0}^{N-1} \tilde{x}[n] e^{-j2\pi kn/N}
\]

\[
\tilde{x}[n] = t[n] \cdot \tilde{f}[n]
\]

\[
x[n] = A \cdot \sin(\omega_0 t) + \sum_{m=1}^{M-1} A_m \sin(\omega_m t + \phi_m)
\]
DFT and Mel-filterbank Processing (cont.)

\[ B(f) = 1125(1 + f / 700) \quad \text{↔} \quad B^{-1}(b) = 700(\exp(b / 1125) - 1) \]

\[
H_{m-1}[k^*] = \frac{f[m] - k^*}{f[m] - f[m-1]}
\]

\[
H_m[k^*] = \frac{k^* - f[m-1]}{f[m] - f[m-1]}
\]
DFT and Mel-filterbank Processing (cont.)

- We then compute the log-energy at the output of each filter as

\[ S[m] = \ln \left( \sum_{k=0}^{N-1} \left| X_a[k] \right|^2 H_m[k] \right), \quad 0 < m \leq M \]

- The mel-frequency cepstrum is then the discrete cosine transform of the \( M \) filter outputs

\[ c[n] = \sum_{m=1}^{M} S[m] \cos(\pi n (m - 1 / 2) / M), \quad 0 \leq n < M \]

- \( M \) varies for different implementations from 24 to 40
- For speech recognition, typically only the first 13 cepstrum coefficients are used
Motivations for Mel-filterbank Analysis

- The human ear resolves frequencies non-linearly across the audio spectrum and empirical evidence suggests that designing a front-end to operate in a similar non-linear manner improves recognition performance.

- The position of maximum displacement along the basilar membrane for stimuli such as pure tone is proportional to the logarithm of the frequency of the tone.

- Frequencies of a complex sound within a certain bandwidth of some nominal frequency cannot be individually identified.
  - When one of the components of this sound falls outside this bandwidth, it can be individually distinguished.
  - This bandwidth is referred to as the critical bandwidth.
  - A critical bandwidth is nominally 10% to 20% of the center frequency of the sound.
Motivations for Mel-filterbank Analysis (cont.)

For speech recognition purpose:

- Filters are non-uniformly spaced along the frequency axis
- The part of the spectrum below 1kHz is processed by more filter banks
  - This part contains more information on the vocal tract such as the first formant
- Non-linear frequency analysis is also used to achieve frequency/time resolution
  - Narrow band-pass filters at low frequencies enables harmonics to be detected
  - Wide bandwidth at higher frequencies allows for higher temporal resolution of bursts
Why Log Energy Computation?

- Using the magnitude (power) only to discard phase information
  - Phase information is useless in speech recognition
    - Replacing the phase part of the original speech signal with continuous random phase won’t be perceived by human ear

- Using the logarithmic operation to compress the component amplitudes at every frequency
  - The characteristic of the human hearing system
  - The compression makes feature extraction less sensitive to dynamic variations
  - To separate the excitation (source) produced by the vocal cords and the filter that represents the vocal tract (homomorphic systems)
Why Inverse DFT?

Discrete Cosine Transform (DCT)

- Since the log-power spectrum is real and symmetric, the inverse DFT reduces to a Discrete Cosine Transform (DCT). The DCT has the property to produce more highly uncorrelated features

\[ c_l[n] = \sqrt{\frac{2}{M}} \sum_{m=1}^{M} S_l[m] \cos \left( \frac{n\pi}{M} \left( m - \frac{1}{2} \right) \right), \quad n = 0, 1, ..., L < M \]

- Cepstral coefficients are more compact since they are sorted in variance order
  - Can be truncated to retain the highest energy coefficients, which represents an implicit liftering operation with a rectangular window
- Successfully separates the vocal tract and the excitation
  - The envelope of the vocal tract changes slowly, and thus at low quefrencies (lower order cepstrum), while the periodic excitation are at high quefrencies (higher order cepstrum)
The performance of a speech recognition system can be greatly enhanced by adding time derivatives to the basic static parameters. The derivatives are calculated as follows:

\[
\Delta c_i[n] = \frac{\sum_{p=1}^{p} p (c_{i+p}[n] - c_{i-p}[n])}{2 \sum_{p=1}^{p} p^2}
\]

\[
\Delta^2 c_i[n] = \frac{\sum_{p=1}^{p} p (\Delta c_{i+p}[n] - \Delta c_{i-p}[n])}{2 \sum_{p=1}^{p} p^2}
\]
MFCC vs. LPC Cepstral Coefficients

- MFCC outperforms LPC Cepstral coefficients
  - Perceptually motivated mel-scale representation indeed helps recognition

<table>
<thead>
<tr>
<th>Feature Set</th>
<th>Relative Error Reduction</th>
</tr>
</thead>
<tbody>
<tr>
<td>13th-order LPC cepstrum coefficients</td>
<td>Baseline</td>
</tr>
<tr>
<td>13th-order MFCC</td>
<td>+10%</td>
</tr>
<tr>
<td>16th-order MFCC</td>
<td>+0%</td>
</tr>
<tr>
<td>+1st- and 2nd-order dynamic features</td>
<td>+20%</td>
</tr>
<tr>
<td>+3rd-order dynamic features</td>
<td>+0%</td>
</tr>
</tbody>
</table>

- Higher-order MFCC does not further reduce the error rate in comparison with the 13-order MFCC
- Another perceptually motivated features such as first- and second-order delta features can significantly reduce the recognition errors
Vector Quantization (VQ)
Vector Quantization (VQ)

- The results of either the filter-bank analysis or the LPC analysis are a series of characteristic vectors of the time-varying spectral characteristics of the speech signal.
- Vector quantization: an arbitrary spectral vector is represented as the index of the codebook vector that best matches the input vector.
- Advantages:
  - Reduced storage for spectral analysis information.
  - Reduced computation for determining similarity of spectral analysis vectors.
  - Discrete representation of speech sounds.
- Disadvantages:
  - An inherent spectral distortion in representing the actual analysis vector.
  - The storage required for codebook vectors is often nontrivial.
VQ – Training and Classification

Figure 3.40  Block diagram of the basic VQ training and classification structure.
The way in which a set of \( L \) training vectors can be clustered into a set of \( M \) codebook vectors is the following (the K-means algorithm):

1. **Initialization**: Arbitrarily choose \( M \) vectors (initially out of the training set of \( L \) vectors) as the initial set of code words in the codebook.

2. **Nearest-Neighbor Search**: For each training vector, find the code word in the current codebook that is closest (in terms of spectral distance), and assign that vector to the corresponding cell (associated with the closest code word). We often use the Euclidean distance for spectral distance measure.

3. **Centroid Update**: Update the code word in each cell using the centroid of the training vectors assigned to that cell.

4. **Iteration**: Repeat steps 2 and 3 until the average distance falls below a preset threshold.
VQ – The Binary Split Algorithm

1. Design a 1-vector codebook; this is the centroid of the entire set of the training vectors (no iteration is required in this step)

2. Double the size of the codebook by splitting each code word $y_n$ in current codebook according to the rule
   $$y^+_n = y_n (1 + \varepsilon)$$
   $$y^-_n = y_n (1 - \varepsilon)$$
   $\varepsilon$: a small positive value, say 0.001

3. Use the K-means iterative algorithm to get the best set of centroids for the split codebook

4. Iterate steps 2 and 3 until a codebook of size $M$ is obtained

Figure 3.42 Flow diagram of binary split codebook generation algorithm.
Pitch Determination
The Role of Pitch

- Pitch determination is very important for many speech processing algorithms
  - Speech synthesis methods require pitch tracking on the desired speech segment if prosody modification is to be done
  - Chinese speech recognition systems use pitch tracking for tone recognition

- LPC and cepstrum represent the filter while pitch represents the source

- Pitch determination algorithms also use short-time analysis techniques
  - For every frame $x_m$, we get a score $f(T \mid x_m)$ that is a function of the candidate pitch periods $T$
  - The pitch determination algorithms determine the optimal pitch according to $T_m = \arg \max_T f(T \mid x_m)$
Pitch Determination
– the Autocorrelation Method

A commonly used method to estimate pitch is based on detecting the highest value of the autocorrelation function in the region of interest.

Given a sinusoidal random process \( x[n] = \cos(w_0 n + \varphi) \)

The autocorrelation is

\[
R[m] = E \{ x^*[n]x[n+m] \} = \int_{-\pi}^{\pi} \cos(w_0n + \varphi) \cos(w_0(n+m) + \varphi) \frac{1}{2\pi} d\varphi
\]

\[
= \frac{1}{2} \int_{-\pi}^{\pi} \left[ \cos(w_0(2n+m) + 2\varphi) + \cos(w_0m) \right] \frac{1}{2\pi} d\varphi
\]

\[
= \frac{1}{2} \cos(w_0m), \quad \text{which has maxima for } m=lT_0, \quad \text{the pitch period and its harmonics}
\]

Any WSS (wide-sense stationary) periodic process \( x[n] \) with period \( T_0 \) also has an autocorrelation \( R[m] \) which exhibits its maxima at \( m=lT_0 \)
Pitch Determination
– the Autocorrelation Method (cont.)

In practice, we need to obtain an estimate \( \hat{R}[m] \) from knowledge of only \( N \) samples.

Use a window \( w[n] \) of length \( N \) on \( x[n] \), the empirical autocorrelation function is given by

\[
\hat{R}[m] = \frac{1}{N} \sum_{n=0}^{N-1}{w[n]x[n]w[n+|m|]x[n+|m|]}
\]

\[
E\left\{ \hat{R}[m]\right\} = \int_{x} \frac{1}{N} \sum_{n=0}^{N-1}{w[n]x[n]w[n+|m|]x[n+|m|]} f_x dx
\]

\[
= \frac{1}{N} \sum_{n=0}^{N-1}{w[n]w[n+|m|]} \int_{x} x[n]x[n+|m|] f_x dx
\]

\[
= R[m](w[m] * w[-m])
\]

For the case of a sinusoidal random process with a rectangular window of length \( N \)

\[
E\left\{ \hat{R}[m]\right\} = \left(1 - \frac{|m|}{N}\right) \frac{\cos\left(\frac{w_0 m}{2}\right)}{2} \quad |m| < N
\]
Pitch Determination
– *the Autocorrelation Method* (cont.)

**Figure 6.31** Waveform and unsmoothed pitch track with the autocorrelation method. A frame shift of 10 ms, a Hamming window of 30 ms, and a sampling rate of 8 kHz were used. Notice that two frames in the voiced region have an incorrect pitch. The pitch values in the unvoiced regions are essentially random.

**Figure 6.32** Autocorrelation function for frame 40 in Figure 6.31. The maximum occurs at 8 samples. A sampling frequency of 8 kHz and window shift of 10 ms are used. The top figure is using a window length of 30 ms, whereas the bottom one is using 50 ms. Notice the quasi periodicity in the autocorrelation function.
Output of LTI Systems

\[ x[n] = \sum_{k=-\infty}^{\infty} \delta[n - kP] \]

\[ h[n] = \sum_{k=-\infty}^{\infty} a^n u[n], \quad |a| < 1 \]

**DTFT**

\[ X(\omega) = \sum_{k=-\infty}^{\infty} \frac{2\pi}{P} \delta\left( \omega - \frac{2\pi}{P} k \right) \]

\[ H(\omega) = \frac{1}{1 - ae^{-j\omega}} \]

\[ x[n] * h[n] \Leftrightarrow H(e^{j\omega})X(e^{j\omega}) \]

\[ Y(\omega) = H(\omega)X(\omega) \]

\[ = \frac{1}{1 - ae^{-j\omega}} \sum_{k=-\infty}^{\infty} \frac{2\pi}{P} \delta\left( \omega - \frac{2\pi}{P} k \right) \]

\[ = \frac{2\pi}{P} \sum_{k=-\infty}^{\infty} \frac{1}{1 - ae^{-j\frac{2\pi}{P} k}} \delta\left( \omega - \frac{2\pi}{P} k \right) \]
The Sampling Theorem

\[ P(j\Omega) = \frac{2\pi}{T} \sum_{k=-\infty}^{\infty} \delta(\Omega - k\Omega_s) \]

\[ X_p(j\Omega) = \frac{1}{2\pi} X(j\Omega) * P(j\Omega) \]

\[ X_p(j\Omega) = \frac{1}{T} \sum_{k=-\infty}^{\infty} X(j(\Omega - k\Omega_s)) \]

\[ \Omega_N < \frac{1}{2}\Omega_s = \frac{\pi}{T} \]

\[ F_s / 2 : \text{the Nyquist frequency} \]

\[ \Omega_N > \frac{1}{2}\Omega_s = \frac{\pi}{T} \]

aliasing distortion
The Rectangular Window

\[ h_\pi[n] = u[n] - u[n - N] \]

\[ H_\pi(z) = \sum_{n=0}^{N-1} z^{-n} = \frac{1 - z^{-N}}{1 - z^{-1}} \]

\[ H_\pi(e^{jw}) = \frac{1 - e^{-jwN}}{1 - e^{-jw}} = e^{-jw/2} \left( e^{jw/2} - e^{-jw/2} \right) e^{-jw/2} = \frac{\sin(wN/2)}{\sin(w/2)} e^{-jw(N-1)/2} = A(w)e^{-jw(N-1)/2} \]

\[ A(w) = 0, \quad \text{for } w_k = \frac{2\pi k}{N} \quad k \neq \{0, \pm N, \pm 2N, \ldots\} \]

Figure 5.19 Frequency response (magnitude in dB) of the rectangular window with \( N = 50 \), which is a digital sinc function.
Fourier Transforms of the Impulse Train

\[ p_N[n] = \sum_{k=-\infty}^{\infty} \delta[n - kN] \]

The impulse train is periodic with period \( N \)

\[ p_N[k] = \frac{1}{N} \sum_{n=0}^{N-1} p_N[n] e^{-j\frac{2\pi nk}{N}} = 1 \]

Alternate expression of an impulse train

\[ \therefore \quad e^{jw_0 n} \leftrightarrow \frac{1}{2\pi} \int 2\pi \delta(w - w_0) e^{jwn} \, dw = e^{jw_0 n} \]

The Fourier transform of an impulse train signal is also an impulse train

\[ P_N(e^{jw}) = \frac{2\pi}{N} \sum_{k=0}^{N-1} \delta(w - 2\pi k / N) \]
Fourier Transforms of Periodic Signals –
\textit{general periodic signals}

Given a periodic signal \( x_N[n] \) with period \( N \), let
\[
x[n] = \begin{cases} x_N[n] & 0 \leq n < N \\ 0 & \text{otherwise} \end{cases}
\]

Then
\[
x_N[n] = \sum_{k=-\infty}^{\infty} x[n - kN] = x[n] * \sum_{k=-\infty}^{\infty} \delta[n - kN] = x[n] * p_N[n]
\]

\[
X_N(e^{jw}) = X(e^{jw}) \left( \frac{2\pi}{N} \sum_{k=0}^{N-1} \delta(w - 2\pi k / N) \right) = \frac{2\pi}{N} \sum_{k=0}^{N-1} X(e^{j2\pi k / N}) \delta(w - 2\pi k / N)
\]

\textbf{Figure 5.14} Relationships between finite and periodic signals and their Fourier transforms. On one hand, \( x[n] \) is a length \( N \) discrete signal whose transform \( X(e^{j\omega}) \) is continuous and periodic with period \( 2\pi \). On the other hand, \( x_N[n] \) is a periodic signal with period \( N \) whose transform \( X_N(e^{j\omega}) \) is discrete and periodic.