An Introductory Course on Speech Processing

COMPLEMENTS

http://elearning.fpms.ac.be/public/1005_08/
http://tcts.fpms.ac.be/cours/1005-08/speech/

T. Dutoit
Thierry.Dutoit@fpms.ac.be
http://tcts.fpms.ac.be/~dutoit

TCTS Lab
Faculté Polytechnique de Mons
Belgium

PART I
Introduction to speech
Spectral snapshot

- Short-term fast Fourier transform (ST-FFT)
  \[ x(n) \leftrightarrow X(k) = \sum_{n=0}^{N-1} x(n)w(n-i)\cdot e^{-\frac{j2\pi nk}{N}} \quad \text{for } k = 0..N-1 \]
- Narrow-band (50ms) to wide-band (5ms)

Spectrogram (wide-band)
PART II
(short-term) Modeling of the speech signal
Contents

- Production models
  - Autoregressive modeling of speech
  - Autoregressive modeling of stationary random signals
    - AR model - Estimation algorithms
    - Autoregressive estimation of speech
    - Extensions of the AR model
- Transform-based models (phenomenological)
  - Homomorphic (or cepstral) transform
    - Definition - Properties
    - Relationship with AR modeling
- F0 estimation

Autoregressive modeling of stationary random signals

- Each sample $x(n)$ is the sum of a completely (linearly) predictable component, and an « innovation » component $e(n)$
  - $\Rightarrow$ Autoregressive or linear prediction (LP) model

$$
X(z) = \frac{E(z)}{\sum_{i=1}^{p} a_i z^{-i}}
$$

$$
x(n) = e(n) + \sum_{i=1}^{p} -a_i x(n-i)
$$

$$
e(n) = \sum_{i=0}^{p} a_i x(n-i) \quad (a_0 = 1)
$$

- $e(n)$ can be computed from $a_i$ ($i=1...p$) and $x(n)$
- $e(n)$ is the output of the inverse filter
**YULE-WALKER equations**

\[ \sigma_e^2 = E[e^2(n)] \]

\[ = E \left[ \sum_{i=1}^{p} a_i x(n-i) \sum_{j=0}^{p} a_j x(n-j) \right] \]

\[ = \sum_{i,j=0}^{p} a_i a_j E[x(n-i)x(n-j)] \]

\[ = \sum_{i,j=0}^{p} a_i a_j \phi_e(i - j) \]

\[ \frac{\partial \sigma_e^2}{\partial a_i} = 0 \quad (i = 1 \ldots p) \]

\[ 2 \sum_{j=1}^{p} \phi_e(i-j) a_j = 0 \]

\[ \sum_{j=1}^{p} \phi_e(i-j) a_j = -\phi_e(i) \quad (i = 1 \ldots p) \]

- Problem: find \( a_i \) \((i=1\ldots p)\) from \( x(n) \), not knowing \( e(n) \)
- Minimize the variance \( \sigma_e^2 \) of \( e(n) \)

- **YULE-WALKER equations**

**YULE-WALKER equations**

\[ \sum_{j=1}^{p} \phi_e(i-j) a_j = -\phi_e(i) \quad (i = 1 \ldots p) \]

\[ \begin{bmatrix} \phi_e(0) & \phi_e(1) & \ldots & \phi_e(p-1) \\ \phi_e(1) & \phi_e(0) & \ldots & \phi_e(p-2) \\ \vdots & \vdots & \ddots & \vdots \\ \phi_e(p-1) & \phi_e(p-2) & \ldots & \phi_e(0) \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \\ \vdots \\ a_p \end{bmatrix} = \begin{bmatrix} \phi_e(1) \\ \phi_e(2) \\ \vdots \\ \phi_e(p) \end{bmatrix} \]

\[ \Phi_e^{-1} a = -\Phi_e^b \]

- \( p \) linear equations with \( p \) unknowns \( \Rightarrow O(p^3) \)?
- Toeplitz matrix \( \Rightarrow O(p^2) \): recursively on the prediction order

**LEVINSON**

**SCHUR(-LEROUX-GUEGEN)**
LEVINSON algorithm

\[
\begin{bmatrix}
\phi_{0} & \phi_{1} & \cdots & \phi_{m-1} & \phi_{m} \\
\phi_{1} & \phi_{0} & \cdots & \phi_{m-2} & \phi_{m-1} \\
\vdots & \vdots & \ddots & \vdots & \vdots \\
\phi_{m-1} & \phi_{m-2} & \cdots & \phi_{0} & \phi_{1}
\end{bmatrix}
\begin{bmatrix}
a_{0} \\
a_{1} \\
\vdots \\
a_{m}
\end{bmatrix}
= 
\begin{bmatrix}
\phi_{1} \\
\phi_{2} \\
\vdots \\
\phi_{m+1}
\end{bmatrix}
\]

Find solution for order \(m+1\) from sol. for \(m\)?

? ```
\[
\begin{bmatrix}
\phi_0(0) & \phi_0(1) & \cdots & \phi_0(m-1) & \phi_0(m) \\
\phi_1(0) & \phi_1(1) & \cdots & \phi_1(m-2) & \phi_1(m-1) \\
\vdots & \vdots & \ddots & \vdots & \vdots \\
\phi_{(m-1)}(0) & \phi_{(m-1)}(1) & \cdots & \phi_{(m-1)}(m-2) & \phi_{(m-1)}(m-1) \\
\phi_{(m)}(0) & \phi_{(m)}(1) & \cdots & \phi_{(m)}(m-2) & \phi_{(m)}(m-1) \\
\end{bmatrix}
\begin{bmatrix}
a_0^m \\
a_1^m \\
\vdots \\
a_{(m-1)}^m \\
a_{(m)}^m \\
\end{bmatrix}
= 
\begin{bmatrix}
\phi_0(1) \\
\phi_1(2) \\
\vdots \\
\phi_{(m)}(m) \\
\phi_{(m+1) + \mu} \\
\end{bmatrix}
\]

(1)

\[
\begin{bmatrix}
\phi_0(0) & \phi_0(1) & \cdots & \phi_0(m-1) & \phi_0(m) \\
\phi_1(0) & \phi_1(1) & \cdots & \phi_1(m-2) & \phi_1(m-1) \\
\vdots & \vdots & \ddots & \vdots & \vdots \\
\phi_{(m-1)}(0) & \phi_{(m-1)}(1) & \cdots & \phi_{(m-1)}(m-2) & \phi_{(m-1)}(m-1) \\
\phi_{(m)}(0) & \phi_{(m)}(1) & \cdots & \phi_{(m)}(m-2) & \phi_{(m)}(m-1) \\
\end{bmatrix}
\begin{bmatrix}
a_0^{m+1} \\
a_1^{m+1} \\
\vdots \\
a_{(m-1)}^{m+1} \\
a_{(m)}^{m+1} \\
\end{bmatrix}
= 
\begin{bmatrix}
\phi_0(1) \\
\phi_1(2) \\
\vdots \\
\phi_{(m)}(m) \\
\phi_{(m+1) + \mu} \\
\end{bmatrix}
\]

(2)

\[
\begin{bmatrix}
\phi_0(0) & \phi_0(1) & \cdots & \phi_0(m-1) & \phi_0(m) \\
\phi_1(0) & \phi_1(1) & \cdots & \phi_1(m-2) & \phi_1(m-1) \\
\vdots & \vdots & \ddots & \vdots & \vdots \\
\phi_{(m-1)}(0) & \phi_{(m-1)}(1) & \cdots & \phi_{(m-1)}(m-2) & \phi_{(m-1)}(m-1) \\
\phi_{(m)}(0) & \phi_{(m)}(1) & \cdots & \phi_{(m)}(m-2) & \phi_{(m)}(m-1) \\
\end{bmatrix}
\begin{bmatrix}
a_0^{m+1} \\
a_1^{m+1} \\
\vdots \\
a_{(m-1)}^{m+1} \\
a_{(m)}^{m+1} \\
\end{bmatrix}
= 
\begin{bmatrix}
\phi_0(1) \\
\phi_1(2) \\
\vdots \\
\phi_{(m)}(m) \\
\phi_{(m+1) + \mu} \\
\end{bmatrix}
\]

(3)

\[
\phi_i(m + 1) + \mu_m + k_{m+1} \alpha_m = \phi_i(m + 1) \quad k_{m+1} = \frac{-H_m}{\alpha_m}
\]
**LEVINSON algorithm**

- **Initialization**
  
  \[
  a_m(0) = 1, \quad m = 1, 2, \ldots, p \quad \alpha_0 = \phi_1(0) = \sigma_x^2
  \]

- **Recursion on** \( m=0\ldots p-1 \)
  
  for \( m = 0, 1, \ldots, p-1 \)
  
  \[
  k_{m+1} = -(1 / \alpha_m) \sum_{i=0}^{m-1} a_{m-1}(i) \cdot \phi_i(m-i)
  \]
  
  for \( i=1, 2, \ldots, m-1 \)
  
  \[
  a_m(i) = a_{m-1}(i) + k_m \cdot a_{m-1}(m-i)
  \]
  
  \[
  a_{m+1}(m) = k_{m+1}
  \]
  
  \[
  \alpha_{m+1} = \alpha_m (1 - k_{m+1}^2)
  \]

**\( k_i \): PARCOR coefficients**

- Naturally appear in the LEVINSON recursion
- Can be **physically** interpreted as area ratios of acoustic tubes in series
- Equivalent to prediction coefficients (there is a recursion formula for \( \{a_i\} \rightarrow \{k_i\} \rightarrow \{a_i\} \))
- Transfer function of \( 1/A(z) \) much less sensitive to \( \Delta k_i \) then to \( \Delta a_i \)
- Good **interpolation** properties
- Correspond to the **lattice filter structure** for \( 1/A(z) \)
- \( 1/A(z) \) stable if \(-1 < k_i < +1\)
Interpolating PARCORs

How good are the $a_i$’s or $k_i$’s?

Levinson (or Schur) algorithm

$\frac{1}{A_p(z)}$

$x_{ad}(n)$

$x \equiv X_{AR}$
How good are the $a_i$’s or $k_i$’s?

$$\phi_x(k) \equiv \phi_{AR}(k)$$

for $k = 0...p$

In practice

- (Pre-accentuation: filter $P(z) = 1 - \mu z^{-1}$)
- (Hamming) Weighting window
  - prevent $e(n)$ from naturally taking high values at the beginning and at the end of the frame
  - let each set of prediction coefficients be more representative of the 10 central ms

$$w(n) = 0.54 + 0.46 \cos(2\pi \frac{n}{N})$$

- Computational load

<table>
<thead>
<tr>
<th></th>
<th>$N=300, p=10$</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Weighting</strong></td>
<td>N 300</td>
</tr>
<tr>
<td><strong>Autocorrelation</strong></td>
<td>$(N - p / 2) (p + 1)$ 3245</td>
</tr>
<tr>
<td><strong>Schur</strong></td>
<td>p $(p + 1)$ 110</td>
</tr>
</tbody>
</table>
Contents

- Production models
  - Autoregressive modeling of speech
  - Autoregressive modeling of stationary random signals
    - AR model - Estimation algorithms
  - Autoregressive estimation of speech
  - Extensions of the AR model
- Transform-based models (phenomenological)
  - Homomorphic (or cepstral) transform
    - Definition - Properties
    - Relationship with AR modeling
- F0 estimation

Homomorphic transform

- Complex cepstrum $x(n)$

\[
x(n) = u(n) \ast h(n) \rightarrow X(z) = U(z)H(z) \rightarrow x(n) = u(n) + h(n)
\]

- Convolution becomes summation:

- If $X(z)$ is rational, then $x(n)$ falls faster than $1/n$
Real cepstrum

- \( c_x(n) \) is the \( y^{-1} \) of the amplitude spectrum of \( x(n) \) in neper (\( \approx \) in dB)

- \( c_x(n) \) can be computed with FFT\(^{-1} \), provided \( N_{\text{FFT}} \) is big enough (to avoid undersampling of \( X(\omega) \))

- NB: cepstrum, quefrency, liftering :-)

Applications to speech

- If \( x(n) = h(n) \) (vocal tract)

\[
\begin{align*}
  c_x(n) & = h(n) \\
  & + c_h(n)
\end{align*}
\]

Use?
- *Measure* \( T_0 \): easier on \( c_x(n) \) than on \( x(n) \)
- *Measure the spectral envelope of* \( x(n) \):
  Isolate \( c_h(n) \to h(n) \)
Applications to speech

Cepstral Mean Subtraction

If \( x(n) = \text{waveform} \ast h(n) \) (tel. line)

\[
c_x(n) = c_{\text{speech}}(n) + c_{\text{line}}(n)
\]

\[
E[c_x(n)] = E[c_{\text{speech}}(n)] + E[c_{\text{line}}(n)]
\]

\[
= E[c_{\text{line}}(n)] + k \text{ (estimator of } c_{\text{line}}(n))
\]

\[
c_{\text{CMS}}(n) = c_x(n) - E[c_x(n)]
\]

\( c_{\text{CMS}}(n) \) is independent from the channel

\textit{Used in speech recognition}

Relationship with AR model

The first values of the cepstrum are characteristic of the transfer function of the vocal tract ⇒ must be related to \( \{a_i\} \)

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

\[
-A'(z) / A(z) = \sum_{n=1}^{\infty} n x(n) z^{-(n+1)}
\]

\[
-\sum_{i=1}^{n} i a_i z^{-(i+1)} = \left[ \sum_{j=0}^{\infty} a_j z^{-j} \right] \left[ \sum_{n=1}^{\infty} n x(n) z^{-(n+1)} \right] \ldots
\]
Contents

- Production models
  - Autoregressive modeling of speech
  - Autoregressive modeling of stationary random signals
    - AR model - Estimation algorithms
  - Autoregressive estimation of speech
  - Extensions of the AR model
- Transform-based models (phenomenological)
  - Homomorphic (or cepstral) transform
    - Definition - Properties
    - Relationship with AR modeling
- F0 estimation

F0 estimation

- Short-term estimation of F0 on an analysis frame (at least 2 times the local pitch period); produces candidates for F0, with score
- Long-term post-processing, making the ultimate choice among candidates (on a 3 frames-basis, usually), as a function of their score and taking into account constraints and correcting estimation biases
Autocorrelation-based F0 estimation

- Periodicity of $x$ implies a max in $r_x(k)$ for $kT_s \approx T_0$
  
  \[ r(k) = \sum_{n=-\infty}^{\infty} x(n)x(n+k) \quad \text{for } k = K_{\text{min}}, \ldots, K_{\text{max}} \]

- Computational load $\gg$>
- F0 doubling for females, because of $F_1 \approx H_2$
  \[ \Rightarrow \text{use threshold}(k) \]

Simplified Inverse Filtering Technique (SIFT)

- Finds the max of the autocorrelation of the error signal ($\Rightarrow F_1 \approx H_2 : \text{flat envelope!}$)
- Decimation-interpolation for lower computational cost
Cepstrum-based F0 estimation

- Cepstrum is $\text{FFT}^{-1}(\log\text{Spectrum})$
- Log spectrum is flatter than Spectrum
- Cepstrum is more pulse-like than signal
Estimating F0 with a comb filter

- Idea: filter signal with comb filter whose F0 varies from 70 to 500 Hz, and measure the energy of the output: $F_0 = F(\min(\text{energy}))$

- Usually performed in the frequency domain directly: $F_0 = F(\min(<\text{FFT}(x)^2, H^2(\text{filter})>))$

Post-processing

- Based on Dynamic Programming:
  From the possible F0 values for successive frames, find the sequence which minimizes a cost function

- Logical filtering
  - ex: one V frame in an island of UV frames $\Rightarrow$ UV
Conclusion

- **LP** model is good at modeling the short-term **spectral envelope** (id. for cepstrum)
- **MP-LPC, CELP, Hybrid H/N** add a better modeling of the contribution of the **excitation** component (the *fine* spectral structure)
- **NB :** these were models of the acoustic component of speech only...

Part III
Speech Coding
**Lower bound?**

- Speech = content, speaker, para-linguistic information
- **Hyp:** Speech information \( \approx \) sequence of phonemes

Bit rate?
- 10 phonemes/sec
- 30-50 phonemes for a language \( = 32 = 2^5 \)
- Min. bit rate\( = 5 \times 10 = 50 \text{ bps!} \)
- To be compared with the **33.6 Kbps** allowed on an analog telephone line (and **64 Kbps** on a digital line)!

---

**Coding**

\[ x(n) \rightarrow \text{Speech Coding} \rightarrow \text{Storage} \rightarrow \text{Speech Decoding} \rightarrow y(n) \]

Transmission line

\[ e(n) = x(n) - y(n) \]

\[ SNR(dB) = 10 \log(\sigma_s^2 / \sigma_e^2) \]

\[ SNRSEG(dB) = E[SNR(dB)] \]

- Minimize storage/transmission bit-rate while keeping high SNRSEG
- NB: in practice, subjective tests are also used (MOS: mean opinion scores)
### Coding, Quantizing

<table>
<thead>
<tr>
<th>Coding Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>No coding</td>
<td>Uniform quantization, Non-uniform quantization, Un. quantization with compression-expansion, Adaptive quantization</td>
</tr>
<tr>
<td>Time-domain coding</td>
<td>Prediction, Quantization</td>
</tr>
<tr>
<td>Parametric coding</td>
<td>Model, Quantization</td>
</tr>
</tbody>
</table>

### Contents

- **No coding**
  - Uniform quantization
  - Non-uniform quantization
  - Un. quantization with compression-expansion
  - Adaptive quantization
- **Time-domain speech coding**
  - DPCM
  - ADPCM
- **Parametric coding**
  - LPC (scalar quantization/vector quantization)
  - Hybrid H/N coding (MBE)
  - MP-LPC, CELP
Uniform quantization

- $b =$ number of bits
  - $x_s =$ saturation level
  - $\Gamma =$ load factor = $x_s / \sigma_x$ (ex: Gaussian $\Gamma=3$)
- If $x$ is guassian and the quantization step is small enough, than in the non-saturated range of $x$:
  \[
  RSB = 10 \log (3\sigma_x^2 / x_s^2 \cdot 2^{2b}) \approx 6.02b + 4.77 - 20 \log \Gamma
  \]
- For a given $\Gamma$, **every bit counts for 6 DBs**
Uniform quantization

Non-uniform quantization

- Bit assignment should follow the p.d.f. of $x$
- Lloyd-Max optimization:
  - start with uniform quantization
  - compute centers of gravity
  - new borders (middle of centers of gravity)

\[
10\log(\frac{\sigma_x^2}{\sigma_t^2}) = -20\log(\Gamma)
\]
Uniform quantization with compression-expansion

- Equivalent to non-uniform quantization

\[
\begin{align*}
F(x) &= \frac{A|x|}{1 + \ln A} \cdot \text{sgn}(x) \\
&= x_{\text{max}} \cdot \frac{1 + \ln \left[ A|x|/x_{\text{max}} \right]}{1 + \ln A} \cdot \text{sgn}(x) 1/ A \leq |x|/x_{\text{max}} \leq 1
\end{align*}
\]

- Europe : A law

\[
A = 87.56  \quad b = 8
\]

A Law
A Law

- \(x \gg \) \( RSB = 6.02 \cdot b + 4.77 - 20 \log (1 + \ln A) = 38.15 \text{dB} \)

- \(x \ll \) \( RSB = 6.02 \cdot b + 4.77 + 20 \cdot \log \left( \frac{A}{1 + \ln A} \right) - 20 \cdot \log \Gamma \)

20 \log(16) = 24 = 4 \text{ bits!}

**G.711 norm at 64 kbps**
(8bits at 8kHz)

Adaptive quantization

- Adapt the quantization step to the local variance of the signal (AQF-AQB)

(1) Adaptive quantization feedforward AQF

(2) Adaptive quantization backward AQB
Contents

- No coding
  - Uniform quantization
  - Non-uniform quantization
  - Un. quantization with compression-expansion
  - Adaptive quantization

- Time-domain speech coding
  - DPCM
  - ADPCM

- Parametric coding
  - LPC (scalar quantization/vector quantization)
  - Hybrid H/N coding (MBE)
  - MP-LPC, CELP

DPCM: differential pulse modulation coding

- Speech is redundant: each sample can be somehow predicted from previous samples
  \[ d(n) = x(n) - x(n-1) \]
  \[ \sigma_d^2 = 2\sigma_x^2 - 2\phi_x(1) = 2\sigma_x^2[1 - \rho_x(1)] \]

- If the autocorrelation is >0.5, than the variance of \( d \) is smaller than that of \( x \), and:
  \[ \Delta \text{RSB} = 20\log \left( \frac{\sigma_x}{\sigma_d} \right) = 10\log \frac{1}{2[1 - \rho_x(1)]} \]
DPCM: more generally

\[ d(n) = x(n) - \sum_{i=1}^{p} a_i x(n-i) \]

\[ P(z) = \sum_{i=1}^{p} a_i z^{-i} \]

Avoiding the error drift
Prediction gain

\[ \Delta SNR \text{ (dB)} \]

Adaptive DPCM: ADPCM

- DPCM with adaptive prediction (ex: APF)
**Adaptive DPCM: ADPCM**

- DPCM with adaptive quantization (ex: AQB)

- CCITT G721-723-726-727 (32 kb/s) are AQB-APB
- ex: 16 kbps

**A comparison at 24 kbps**

![Graph showing SNR (dB) and Gain over Log PCM (dB) for different quantizer types and predictors.](image)
Contents

- No coding
  - Uniform quantization
  - Non-uniform quantization
  - Un. quantization with compression-expansion
  - Adaptive quantization
- Time-domain speech coding
  - DPCM
  - ADPCM
- Parametric coding
  - LPC (scalar quantization/vector quantization)
  - Hybrid H/N coding (MBE)
  - MP-LPC, CELP

Linear Prediction Coding

Sender

\[ x(n) \]

\[ \text{Schur} \]

\[ F_0 \]

Quantization of \( k_i T_0 \) and \( V/UV \)

NATO LPC10: 2400 bps

Receiver

\[ \frac{1}{A(z)} \]

\[ \hat{x}(n) \]
LPC + Vector Quantization

- Idea: pre-establish a (finite) list of \{k_i\} so as to transmit the index of a \{k_i\} in this list, instead of the \{k_i\} itself
- More generally: separate the continuum of \(x = [x_1, x_1, \ldots, x_p]^T\) into a finite set of classes, represented by a centroid \(C_i\)
- For \(x = [x_1, x_1, \ldots, x_p]^T\) send to index of the closest centroid
- \textit{Distance?} Euclidian, Itakura, etc.
- \textit{Centroids?} Generalized Lloyd-Max

LPC + Vector Quantization

- Generalized Lloyd-Max optimization:
  - start with random choice for centroids ●
  - assign each vector to its closest centroid
  - new centroids = centers of gravity ★ of classes

Typ: 800 bps
Used in spacial communications
Contents

- No coding
  - Uniform quantization
  - Non-uniform quantization
  - Un. quantization with compression-expansion
  - Adaptive quantization
- Time-domain speech coding
  - DPCM
  - ADPCM
- Parametric coding
  - LPC (scalar quantization/vector quantization)
  - **Hybrid H/N coding (MBE)**
  - MP-LPC, CELP

**Multi-Band Excited (MBE) coder**

- Use a hybrid H/N model, and send the H and N components with the least possible number of bits
- Send amplitude of harmonics? No: send spectral envelope (LPC)
- Phases? No assume min. phase
- Noise? Send noise levels / freq band

![Diagram of Multi-Band Excited (MBE) coder](attachment:image.png)
Multi-Band Excited (MBE) coder

Used in marine communications (4.8 kbps)

MP-LPC coder

GSM (RPE): 13000 bps
Issues in CELP coding

- Optimize the excitation dictionary?
- Computational load >>
  ⇒ use specially designed (tree-based) dictionaries
- For very low bit rate, use a predictor on the excitation itself:
  \[ \hat{e}(n) = \hat{e}_c(n) + b\hat{e}(n - P) \]
- Latest versions reach the ADPCM 32 kbps quality with 4.8 kbps only.
PART IV

Automatic Speech Recognition (ASR)
Contents

- Introduction

**Feature extraction**
- Instance-based approach (DTW)
- Model-based approach (HMM, HMM/ANN)
  - Acoustic model
  - Phonetic model
  - Language model

Speech models for ASR

- Ideal properties of parameters:
  - Invariant across the speakers for the same sounds
  - Good discrimination between speech sounds
  - Robust to noise

- Types of features used in practice:
  - LPC-based features: **cepstrum coefficients**
  - Frequency warped spectral features: idem + use a non-linear frequency axis to mimic the human auditory system. (e.g. **PLP analysis, MEL-cepstrum**).
  - Auditory features: outputs of auditory models of the cochlea and auditory nerves
**CMS-LPC coefficients**

- **Speech Signal**
  - High-pass filter: \(1/(1-a.z^{-1})\)
  - Spectral shaping
  - (Hamming) Windowing
  - Autocorrelation
  - Schur algorithm + \(k_i \rightarrow a_i\)
- **Cepstral coefficients**
  - \(a_i \rightarrow c_i\) recursion
  - Lifting
  - Cepstral mean substraction
- **CMS - LPC cepstrum coefficients** (every 10 ms)

**MFCC - PLP**

- **Mel-Frequency Cepstrum Coefficients**
  - Apply frequency warping on the spectrum of \(x(n)\)
  - Bark scale: based on auditory critical bands (non linearly spaced in Hz; linearly spaced in Barks)
  - Apply LP model on the result

- **Perceptual Linear Prediction coefficients**
  - \(x(n) \rightarrow x(\omega)\) Hz→Mel
  - \(\ln | | \rightarrow \overset{\gamma-1}{\overset{\gamma}{\rightarrow}} \text{mfcc}_c(n)\)
Enriching the feature set

- $c_x(n)$ can be anything for silence frames
  $\Rightarrow$ add **energy** $\sigma_x$ of $x(n)$
- Add $F_0$? tried, without success
- Add **delta features** for better accounting
  for (frame-to-frame) spectral dynamics:
  $\Delta c_x^i(n) = c_x^i(n) - c_x^{i-1}(n)$
  $\Delta \sigma_x^i(n) = \sigma_x^i(n) - \sigma_x^{i-1}(n)$

Contents

- Introduction
- Feature extraction
  - **Instance-based approach (DTW)**
  - Model-based approach (HMM, HMM/ANN)
    - Acoustic model
    - Phonetic model
    - Language model
Global distance $D(X, Y^k)$?

- **Non linear** time warping

![Graph showing non linear time warping](image)

- Best path?

Dynamic Time Warping (DTW)

- $D(n,j) = \text{accumulated distance from (1,1) to (n,j)}$
- $D(n,j) = \min \text{ of}$
  - $d(n,j) + D(n-1,j-1)$
  - $d(n,j) + D(n,j-1)$
  - $d(n,j) + D(n-1,j)$
- $D(X, Y^k) = D(N, J(k))$
Dynamic Time Warping (DTW)

- Possible paths are constrained
  - by allowable steps; ex:
    - by additional global constraints

- Penalties can be added to steps, so as to avoid too much deviation from diagonal

Contents

- Introduction
- Feature extraction
- Instance-based approach (DTW)
- Model-based approach (HMM, HMM/ANN)
  - Acoustic model
    - Phonetic model
    - Language model
Training HMMs

- Data: «brbrrbgbrbbgbggbrbgbrggbrbgg... »
- Emission and transition probabilities? If states were known: counting
- **EM** (expectation-maximization) algorithm
  - Initialize Probs. (first guess if possible): \( M^0 \)
  - Decode the data with \( M^0 \rightarrow \) states
  - Re-estimate Probs. by counting: \( M^1 \) until \( M^j \approx M^{j+1} \)

HMMs for ASR

- Observation: \( x_n \) \( \rightarrow P(x_n|\text{state}) = \text{continuous} \)
  Cannot be estimated by counting
  Estimated via the p.d.f. of a distribution
  - ex: Gaussian
  - Multi-Gaussian

- In practice, each phoneme is modeled as 3 states (*Bakis* model)
HMM/ANN hybrids for ASR

States: $1 \ 2 \ 3 \ \ldots \ K$

Frames: $x_{n-4} \ x_{n-3} \ \ldots \ x_n \ \ldots \ x_{1n+3} \ x_{n+4}$

(The perceptron)

\[ g(x) = \frac{1}{1 + e^{-x}} \]
HMM/ANN hybrids for ASR

- Multi-Layer Perceptrons (MLP) can be used for estimating posterior probabilities:
  \[ P(\text{state}|x_n) \]
  i.e. they can approach Bayesian classification!
  [Bourlard & Wellekens 90]

Contents

- Introduction
- Feature extraction
- Instance-based approach (DTW)
- Model-based approach (HMM, HMM/ANN)
  - Acoustic model
  - Phonetic model
  - Language model
**n-gram decoding**

- \( \max P(M_j|X) = \max [P(X|M_j) \cdot P(M_j)] \)
- It has been assumed that \( M_j \) was known, while \( M_j \) should obviously depend on \( X \) itself! (try to recognize the most probable sentence first)
- **Depth-first search:**
  - at each frame \( i \), the decoder stores on a stack the list of most likely word sequences up to frame \( i \)
  - these word sequences are tested first for the estimation of the acoustic score for frame \( i+1 \)

---

**PART V**

*Text-to-Speech Synthesis (TTS)*
Contents

- Introduction
- Acoustic speech synthesis (DSP)
  - Model-based (rule-based) approach
  - **Instance-based (concatenative) approach**
- From text to phonemes and prosody (NLP)
  - Preprocessing
  - Morpho-syntactic analysis
  - Phonetization
  - Prosody Generation

2. Diphone concatenation (1977)
2. Diphone concatenation (1977)

Joe Olive’s LPC synthesizer (1977)
Christian Hamon’s PSOLA (1988)

- Based on the same Poisson’s sum formula as PSOLA, but using edited diphones
- Similar overall quality as PSOLA
- Same computational load
- Completely automatic!
  ⇒ can be used to create lots of compatible synthesizers

Ma voix...

T. Dutoit’s MBROLA (1993)

- J’ai été conçu...
- Ma voix...

Unit selection-based synthesis

How to get the best sequence of units for a given utterance? **Viterbi search**

- **Target cost?**
  
  How to predict which units will sound as they would naturally connected? (should be perceptual)

- **Concatenation cost?**

  How to predict which sequences of units will sound naturally connected? (should be perceptual)
3. Automatic unit selection

**TARGET**

sent: “To be…”
phonet: _ t U b i: ...
stress: ^ ...
tone: H ...
dur: 210 40 55 80 198 ...
f0: ...

**Very Large corpus**

sent: “… to bear.”
phonet: t U b E@ ...
stress: ^ ...
tone: L ...
dur: 150 50 85 90 150 ...
f0: Formants:

**TARGET**

sent: “… to bear.”
phonet: t U b E@ ...
stress: ^ ...
tone: L ...
dur: 150 50 85 90 150 ...
f0: Formants:

**Very Large corpus**

sent: “… to bear.”
phonet: t U b E@ ...
stress: ^ ...
tone: L ...
dur: 150 50 85 90 150 ...
f0: Formants:

Unit i

Unit i+1

Concatenation cost cc(i,i+1)

Target cost tc(j,i)

Formants:
3. Automatic unit selection

Very Large corpus

sent: "... to bear."
phonet: t U b E@ ...
stress: ^ ...
tone: l L ...
dur: 150 50 85 90 150 ...
f0:

Formants:

Unit i-1

Unit i

Concatenation cost \(cc(i-1, i)\) = 0 in case of successive units

Contents

- Introduction
- Acoustic speech synthesis (DSP)
  - Model-based (rule-based) approach
  - Instance-based (concatenative) approach
- From text to phonemes and prosody (NLP)
  - Preprocessing
  - Morpho-syntactic analysis
  - Phonetization
  - Prosody Generation
4. Syntactic-Prosodic Phrasing

- CART tree

\[ st = \text{time to end of sentence} \]
\[ j_3 = \text{tag of word on the right} \]
\[ j_2 = \text{tag of word on the left} \]
\[ tr = \text{utterance rate (in words/second)} \]

Classification and Regression trees (CARTs)

Predict Color(n) \(\leftarrow\) Shapes, Sizes \((n,n-1,n+1,n-2,n+2,\ldots)\) ?

SOL: if \(\text{Shape}(n-1) = \text{Shape}(n)\) \(\Rightarrow\) Color(n) = White
else Color(n) = Black

This can be seen as a classification problem:

<table>
<thead>
<tr>
<th>C(n)</th>
<th>S(n)</th>
<th>SH(n)</th>
<th>S(n-1)</th>
<th>SH(n-1)</th>
<th>S(n+1)</th>
<th>SH(n+1)</th>
<th>...</th>
</tr>
</thead>
<tbody>
<tr>
<td>B</td>
<td>S</td>
<td>S</td>
<td>-</td>
<td>-</td>
<td>S</td>
<td>C</td>
<td>...</td>
</tr>
<tr>
<td>B</td>
<td>S</td>
<td>C</td>
<td>S</td>
<td>S</td>
<td>B</td>
<td>C</td>
<td>...</td>
</tr>
<tr>
<td>W</td>
<td>B</td>
<td>C</td>
<td>S</td>
<td>C</td>
<td>S</td>
<td>C</td>
<td>...</td>
</tr>
<tr>
<td>W</td>
<td>S</td>
<td>C</td>
<td>B</td>
<td>C</td>
<td>B</td>
<td>T</td>
<td>...</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>
Classification and Regression trees (CARTs)

« impure » set

Yes

Question which splits into best « purified » sets

No

« pure » sets

« Purity » of a set?
= « Entropy » (bits)
= -P(Black) log₂[P(Black)] -P(White) log₂[P(White)]

Entropy = -(1/2 x -1) -(1/2 x -1)
=1 bit

Total entropy after split = 0+0=0

Entropy = -(1 x 0) -(0 x ...)
=0 bit

Entropy = -(1 x 0) -(0 x ...)
=0 bit