2. Isolated word speech recognition & HMMs

Task: Recognition of isolated spoken words from a given vocabulary

Problems:
- variable speed
- speaker independence
- prosody, etc.

How do we (mammals) hear?

audio signal → tympanic membrane → ossicles →
cochlea: basilar membrane (scala media), inner & outer hair cells →
auditory cortex

A. Signal pre-processing

- Sample the pressure-time function $f(t)$ and digitise it
  highest frequency in speech signal < 10 KHz
  → Nyquist theorem → sample with 20 KHz

- Apply digital Fourier transform with sliding window

$$c(\omega,t) = \int_{-\infty}^{\infty} W(t-t') f(t') e^{i\omega t'} dt'$$

simplest window function $W(t) = \begin{cases} 1 & \text{if } |t| < b \\ 0 & \text{otherwise} \end{cases}$

width b: lowest freq. vs. time resolution

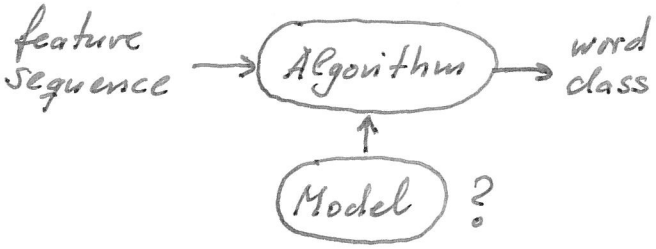
- Energy in spectra (logarithmic, dB)

$$S(\omega,t) = 20 \log_{10} \|c(\omega,t)\|$$

discretise domain of ω into ~20 frequency channels

- Possibly cluster spectral vectors
  pro: small number of feature vectors
  con: dominance of stationary parts

B. Dynamic time warping & word recognition



Model: a set of prototypes (i.e. feature sequences) per class

Algorithm: nearest neighbour classifier

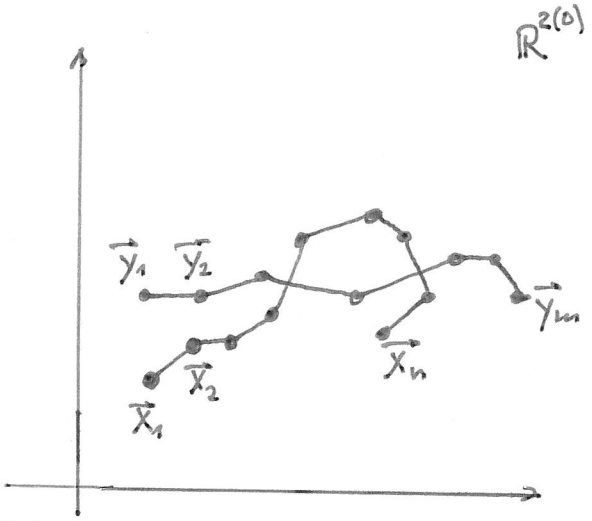
We need a distance measure for sequences of feature vectors

prototype $x = (\vec{x}_1, \vec{x}_2, .., \vec{x}_n)$, signal $y = (\vec{y}_1, \vec{y}_2, .., \vec{y}_m)$

$\vec{x}_i, \vec{y}_j \in \mathbb{R}^{20}$. Distance $D(x,y) = ?$

Monotonous matching (aka time warping)

$\tau = ((i_1 j_1), (i_2 j_2), ..., (i_e j_e)) \in T$ if



(1) $(i_1 j_1) = (1,1)$, $(i_e j_e) = (n,m)$

(2) $i_{k-1} \leq i_k \leq i_{k-1} + 1$

similarly for j-s

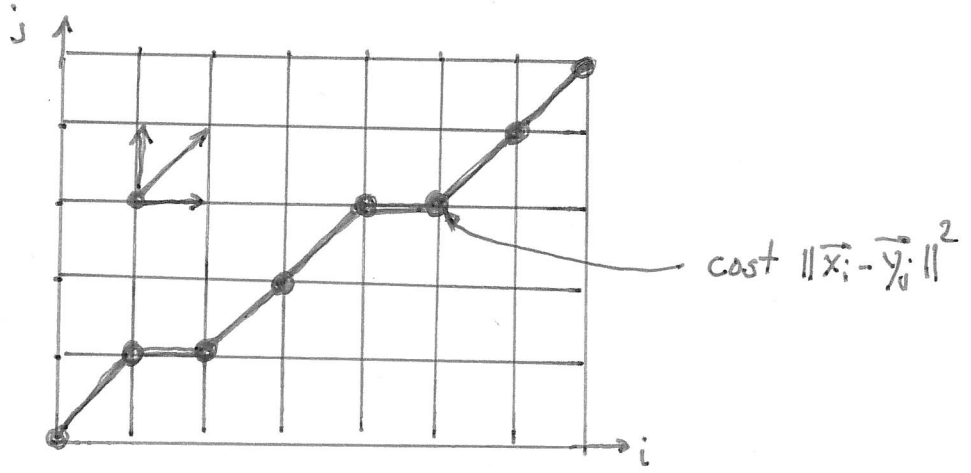
Distance for a fixed matching $\tau \in T$

$$D(x,y;\tau) = \sum_{k=1}^{|\tau|} \| \vec{x}_{i_k} - \vec{y}_{j_k} \|^2$$

Distance

$$D(x,y) = \min_{\tau \in T} D(x,y;\tau)$$

How to compute it efficiently?



i.e. shortest path, here by dynamic programming, complexity $O(nm)$

Discussion model & algorithm

- Inference has high time complexity $O(n^2 p)$, where p is the total number of prototypes
- learning: how to choose optimal prototypes?

Better: Model each word (class) by an HMM

$X = (\vec{x}_1, \dots, \vec{x}_n)$ – sequence of features

$S = (s_1, \dots, s_n)$ – sequence of hidden states (e.g. phonemes)

$$p(X,S) = p(s_1) \prod_{i=2}^{n} p(s_i | s_{i-1}) \prod_{i=1}^{n} p(\vec{x}_i | s_i)$$

- fast inference (linear in n)
- feasible learning of model parameters