


Introduction to audio signal processing



Part 2

Chapter 3: Audio feature extraction techniques

Chapter 4 : Recognition Procedures

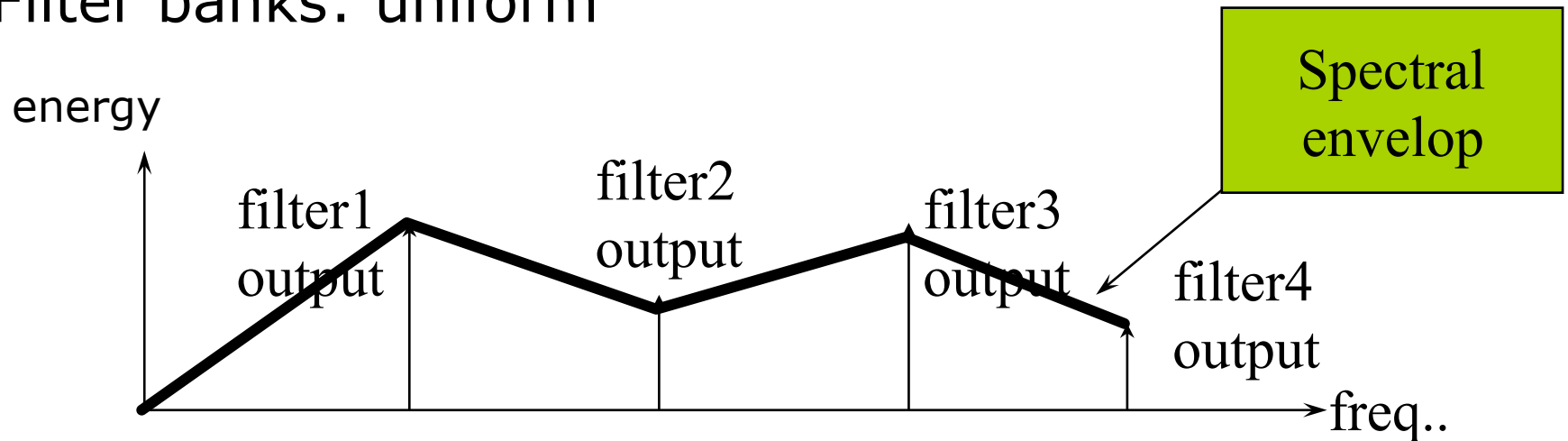
Chapter 3: Audio feature extraction techniques

- ▣ 3.1 Filtering
- ▣ 3.2 Linear predictive coding
- ▣ 3.3 Vector Quantization (VQ)

Chapter 3 : Speech data analysis techniques

□ Ways to find the spectral envelope

■ Filter banks: uniform



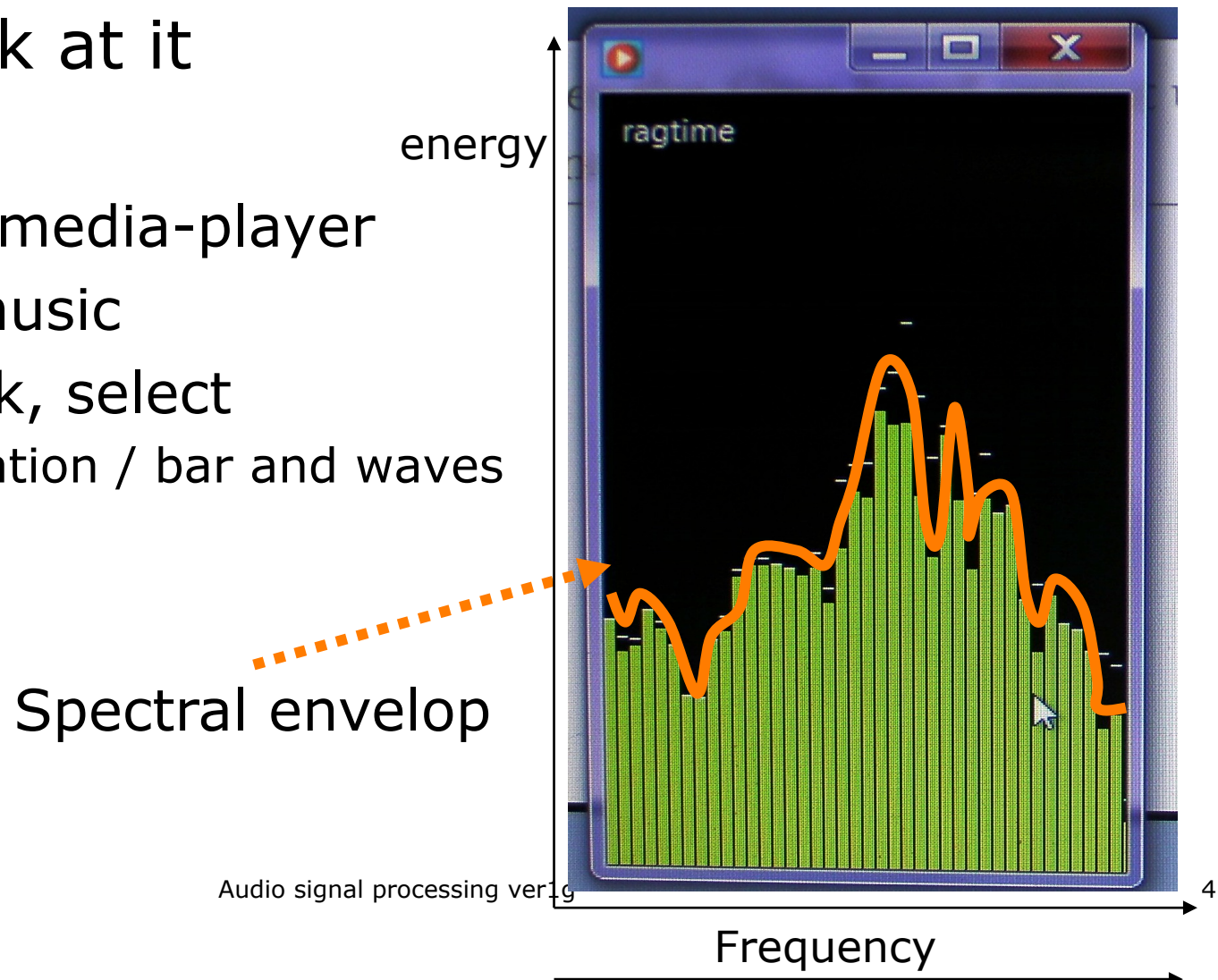
■ Filter banks can also be non-uniform

■ LPC and Cepstral LPC parameters

□ Vector quantization method to represent data more efficiently

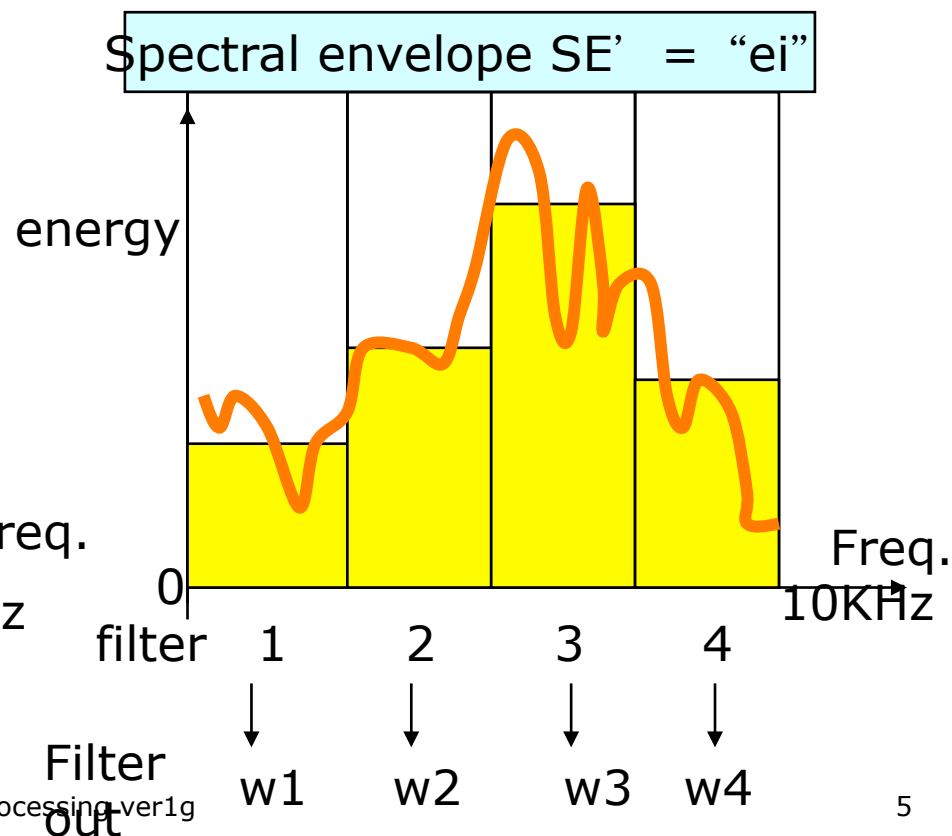
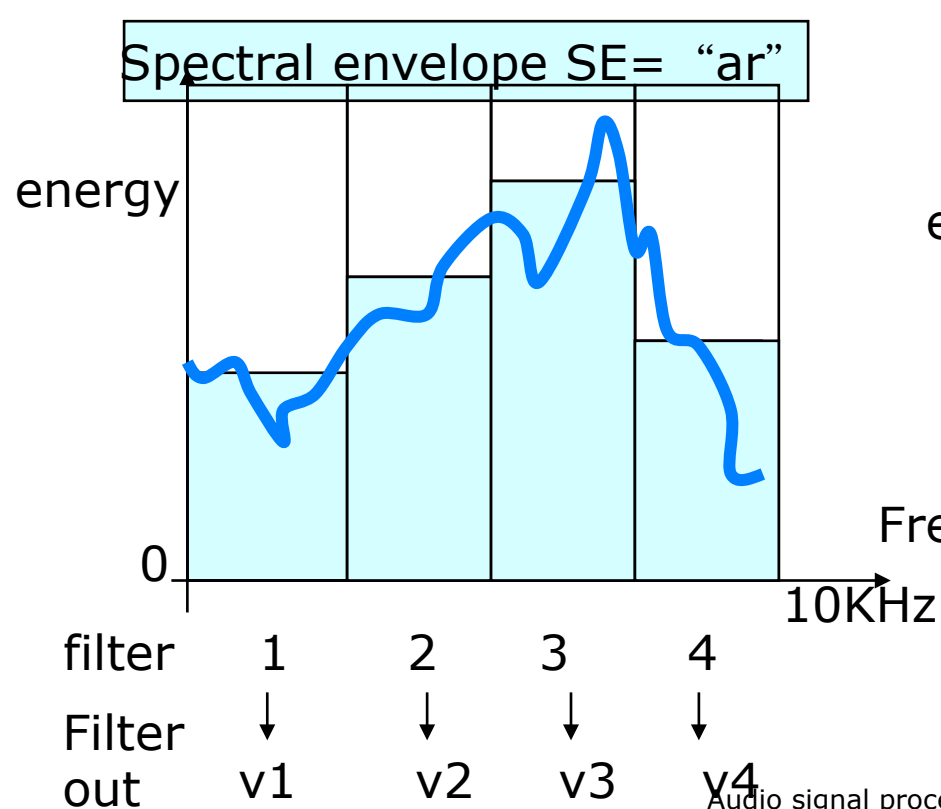
You can see the filter band output using windows-media-player for a frame

- Try to look at it
- Run
 - windows-media-player
 - To play music
 - Right-click, select
 - Visualization / bar and waves



Speech recognition idea using 4 linear filters, each bandwidth is 2.5KHz

- Two sounds with two spectral envelopes SE, SE'
- E.g. SE="ar", SE'="ei"

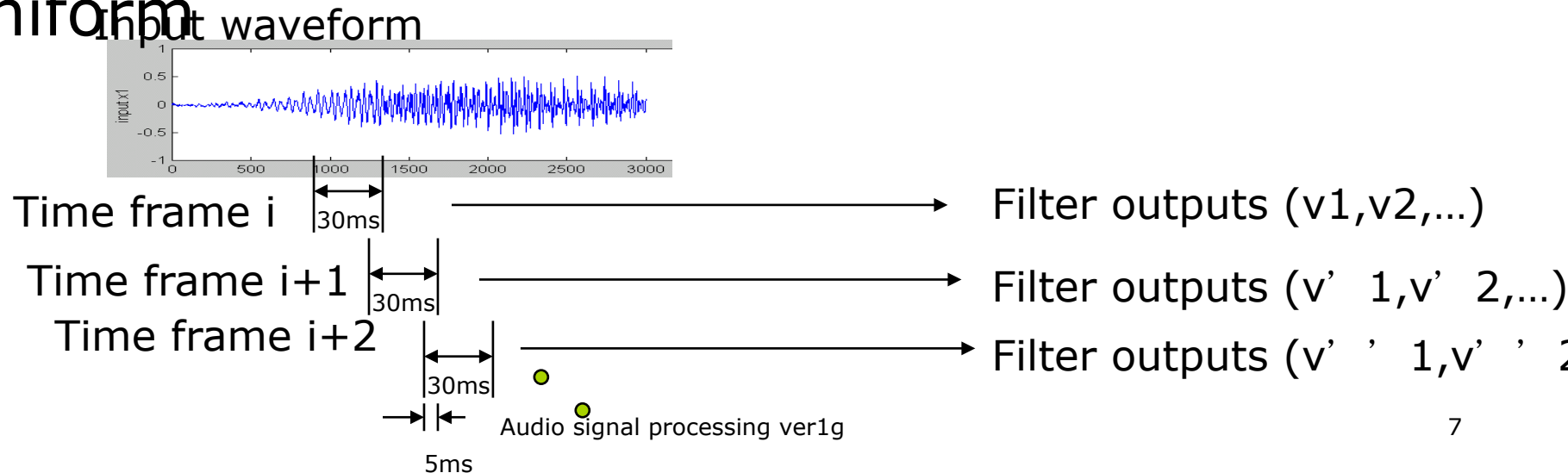


Difference between two sounds (or spectral envelopes SE SE')

- Difference between two sounds
- E.g. $SE = \text{"ar"}$, $SE' = \text{"ei"}$
- A simple measure is
- $\text{Dist} = |v_1 - w_1| + |v_2 - w_2| + |v_3 - w_3| + |v_4 - w_4|$
- Where $|x| = \text{magnitude of } x$

3.1 Filtering method

- For each frame (10 - 30 ms) a set of filter outputs will be calculated. (frame overlap 5ms)
- There are many different methods for setting the filter bandwidths -- uniform or non-uniform



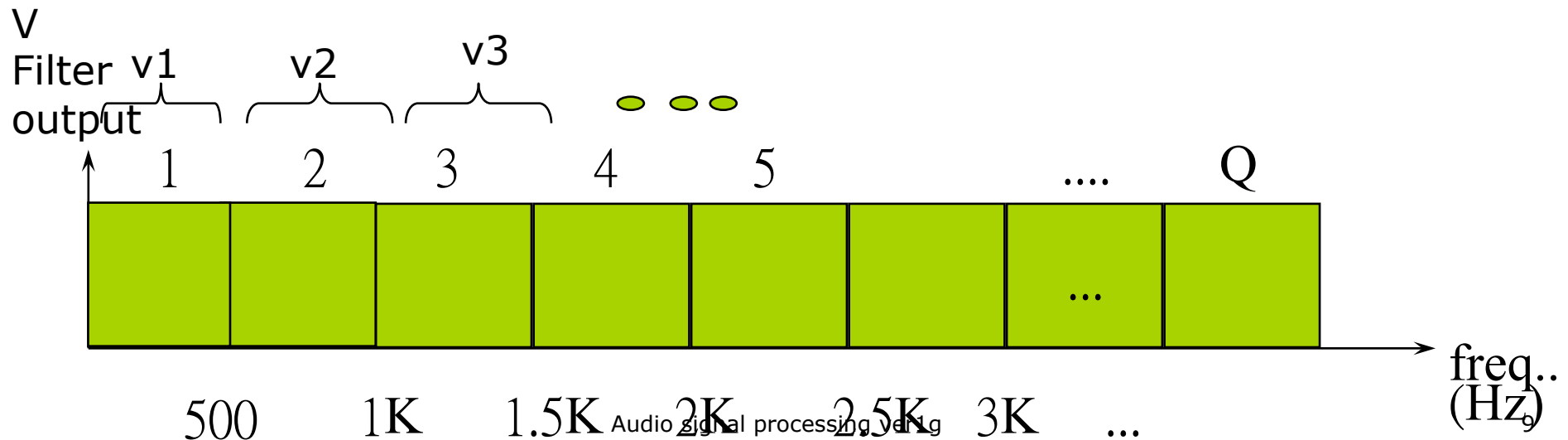
How to determine filter band ranges

- The pervious example of using 4 linear filters is too simple and primitive.
- We will discuss
 - Uniform filter banks
 - Log frequency banks
 - Mel filter bands

Uniform Filter Banks

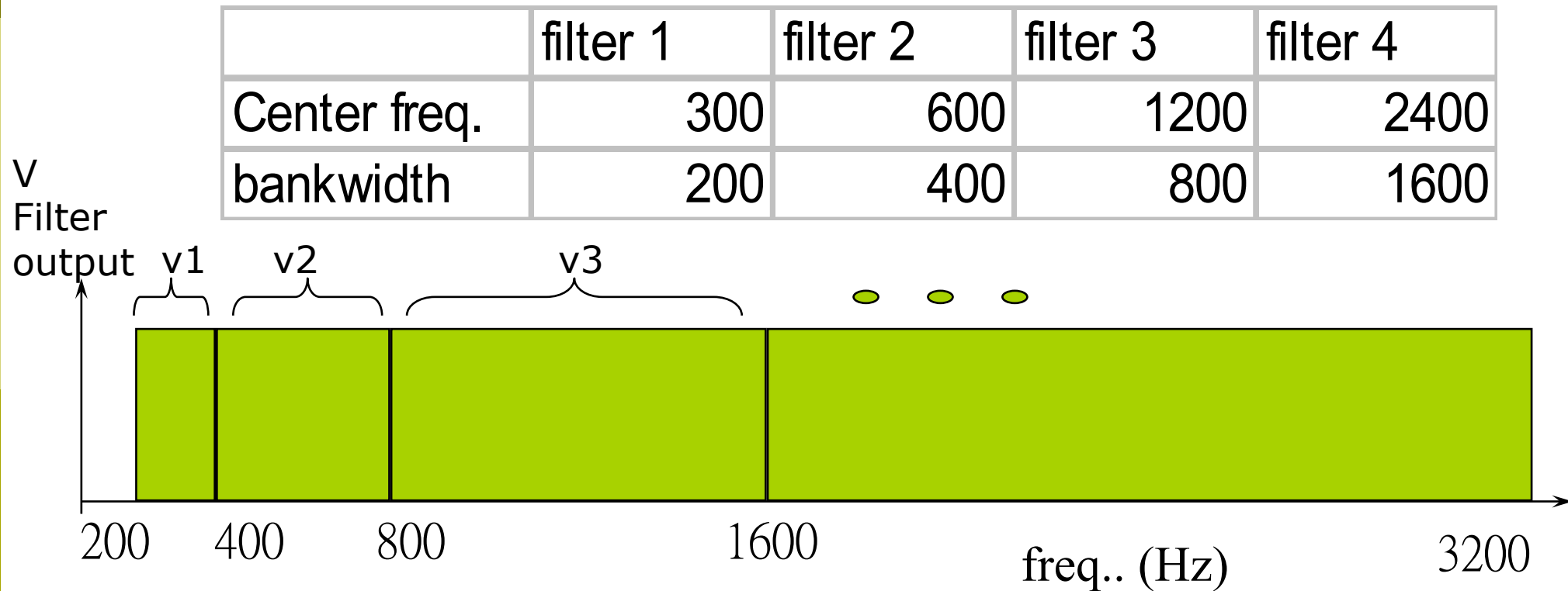
□ Uniform filter banks

- bandwidth $B = \text{Sampling Freq... } (F_s) / \text{no. of banks } (N)$
- For example $F_s = 10\text{Kz}$, $N = 20$ then $B = 500\text{Hz}$
- Simple to implement but not too useful



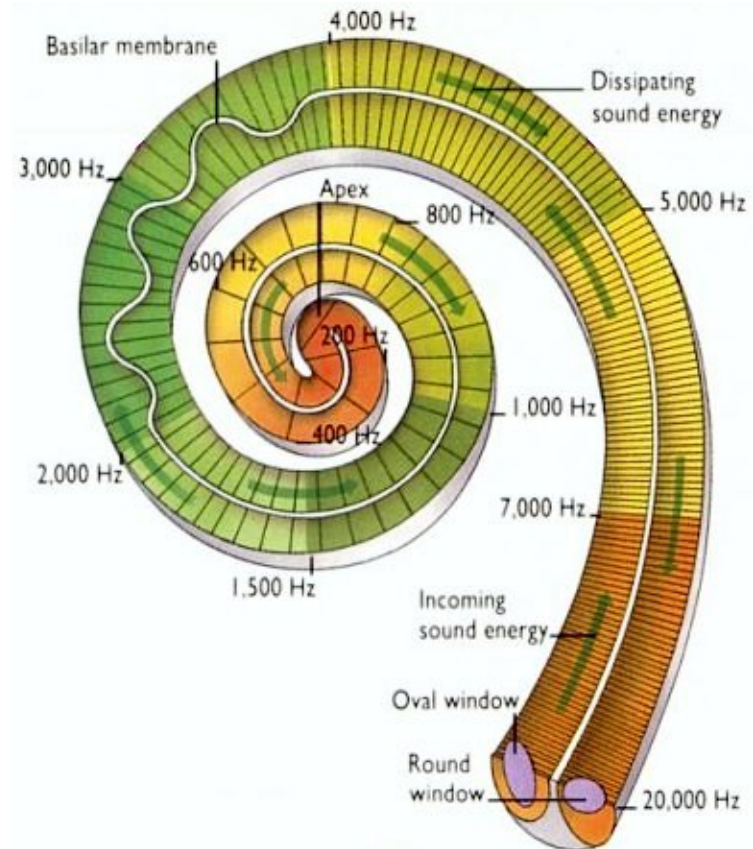
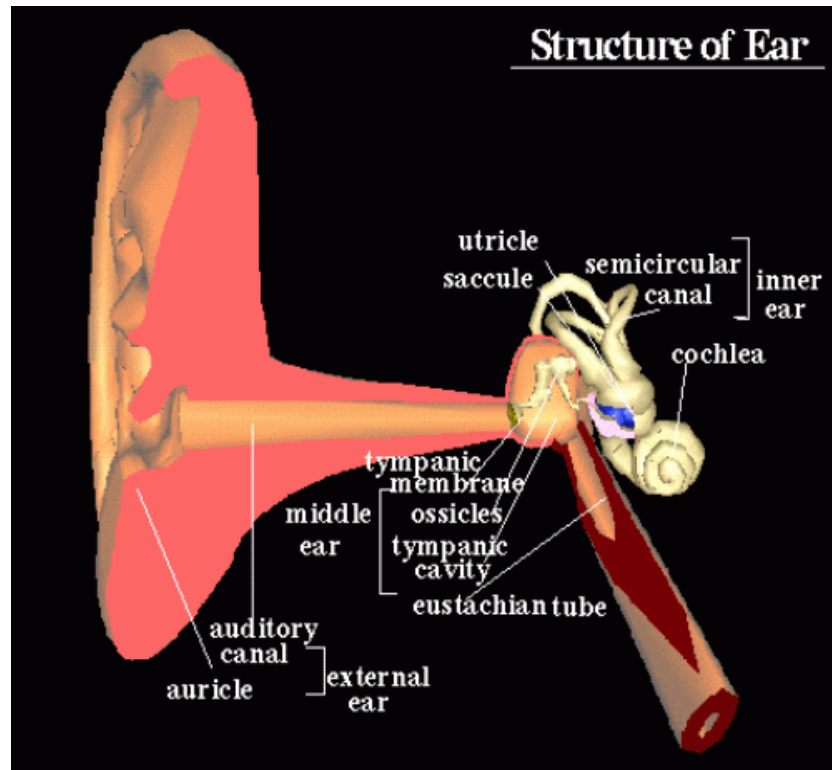
Non-uniform filter banks: Log frequency

- Log. Freq... scale : close to human ear



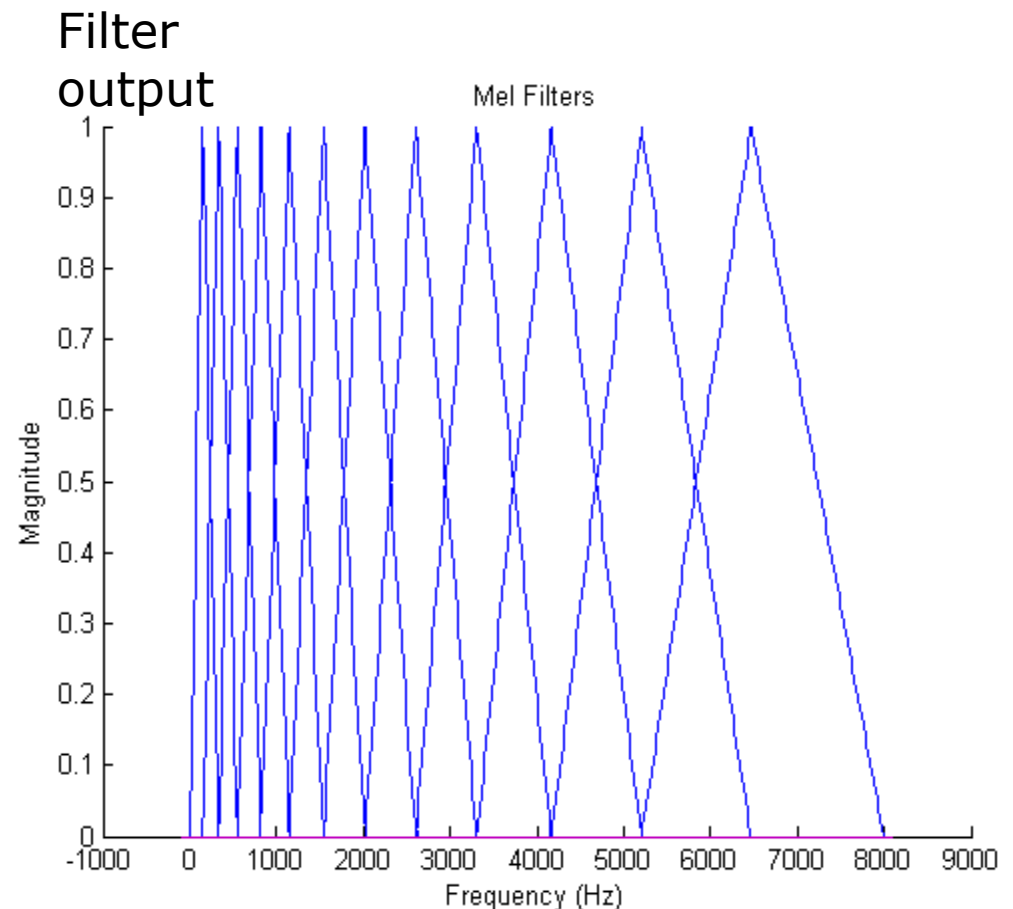
Inner ear and the cochlea (human also has filter bands)

□ Ear and cochlea



Mel filter bands (found by psychological and instrumentation experiments)

- Freq. lower than 1 KHz has narrower bands (and in linear scale)
- Higher frequencies have larger bands (and in log scale)
- More filter below 1KHz
- Less filters above 1KHz

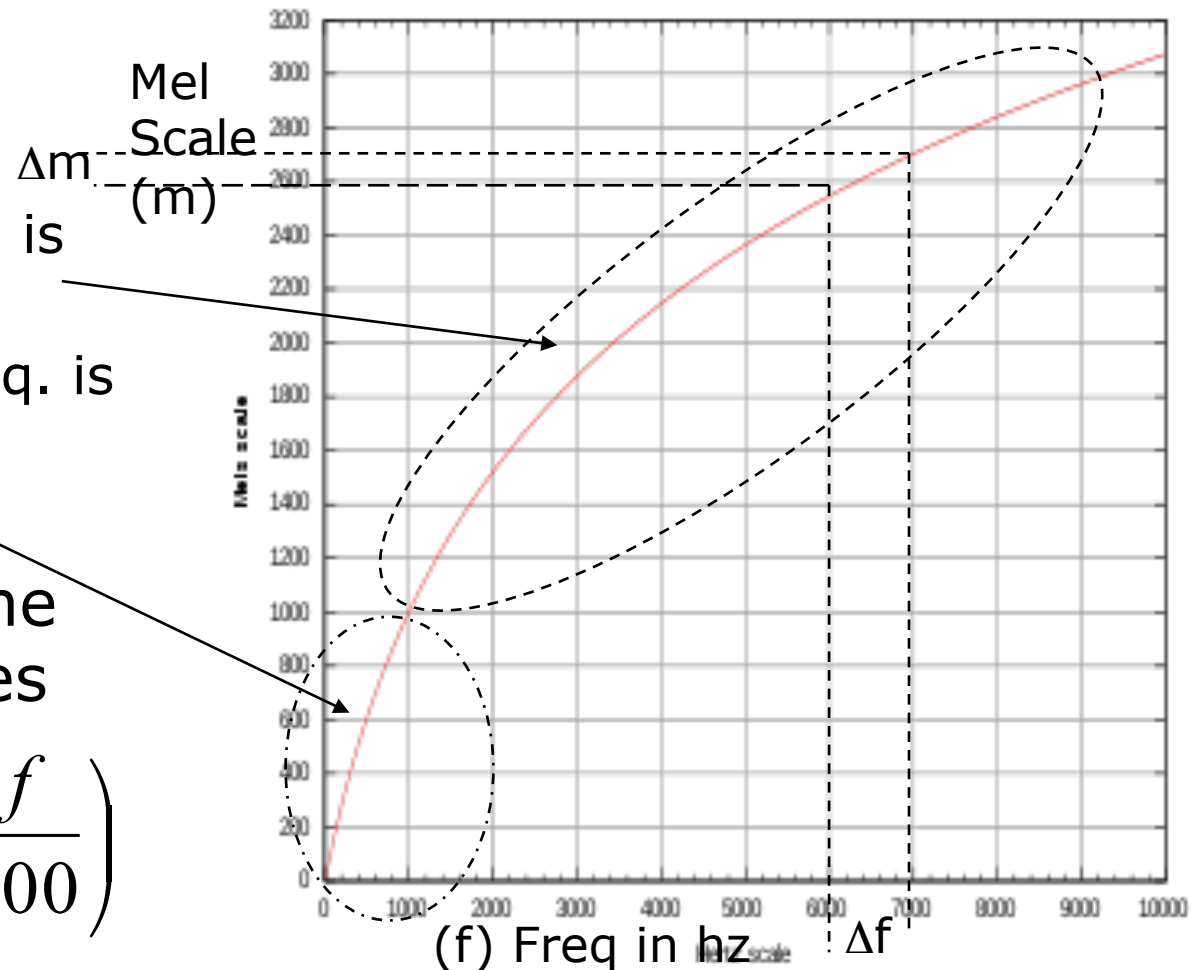


Critical band scale: Mel scale

- Based on perceptual studies
 - Log. scale when freq. is above 1KHz
 - Linear scale when freq. is below 1KHz

- popular scales are the "Mel" or "Bark" scales

$$m = 2595 \log_{10} \left(1 + \frac{f}{700} \right)$$



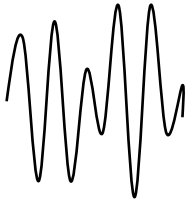
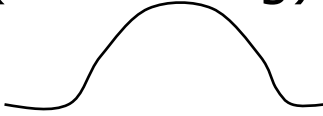
How to implement filter bands



Linear Predictive coding LPC
methods

3.2 Feature extraction data flow

- The LPC (Liner predictive coding) method based method

- Signal 
-
- preprocess -> autocorrelation-> LPC ----> cepstral coef
- (pre-emphasis) r_0, r_1, \dots, r_p a_1, \dots, a_p c_1, \dots, c_p
- (windowing)  (Durbin alog.)

Pre-emphasis

- *" The high concentration of energy in the low frequency range observed for most speech spectra is considered a nuisance because it makes less relevant the energy of the signal at middle and high frequencies in many speech analysis algorithms."*
- From Vergin, R. etal. ,""Compensated mel frequency cepstrum coefficients ", IEEE, ICASSP-96. 1996 .

Pre-emphasis -- high pass filtering
(the effect is to suppress low frequency)

- To reduce noise, average transmission conditions and to average signal spectrum.

$$\tilde{S}(n) = S(n) - \tilde{a}S(n-1)$$

$$0.9 \leq \tilde{a} \leq 1.0, \text{ typically } \tilde{a} = 0.95$$

For $S(0), S(1), S(2), \dots$,

the value $\tilde{S}(0)$ does not exist and is never used.

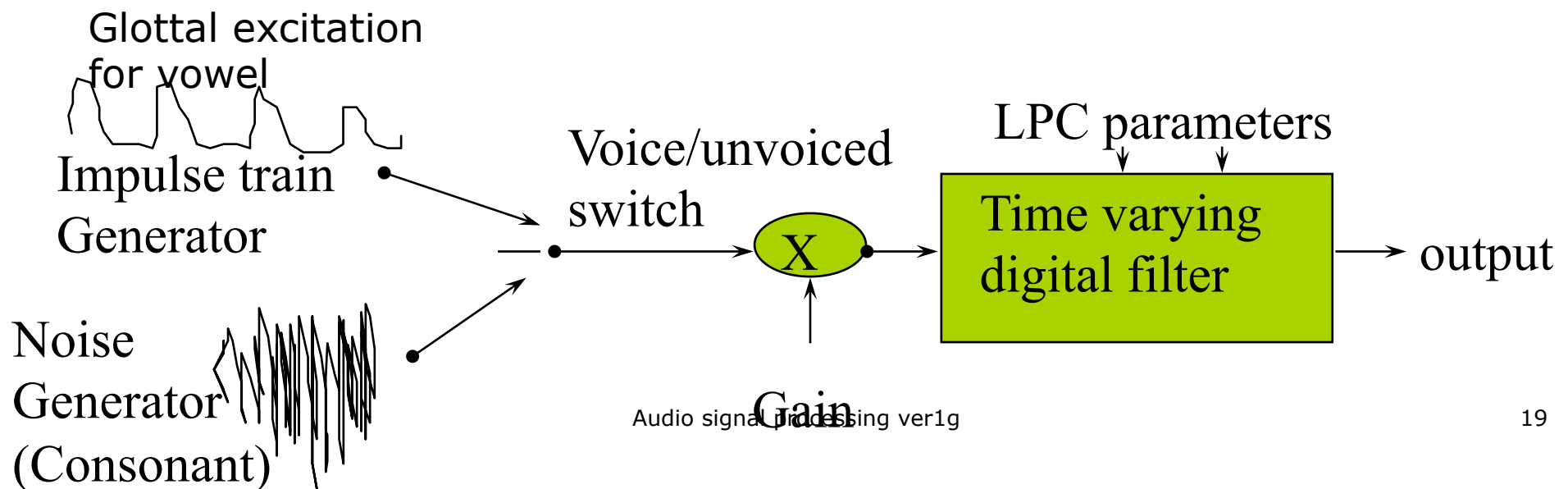
3.2 The Linear Predictive Coding LPC method

- Linear Predictive Coding LPC method
 - Time domain
 - Easy to implement
 - Archive data compression

First let's look at the LPC speech production model

□ Speech synthesis model:

- Impulse train generator governed by pitch period--glottis
- Random noise generator for consonant.
- Vocal tract parameters = LPC parameters

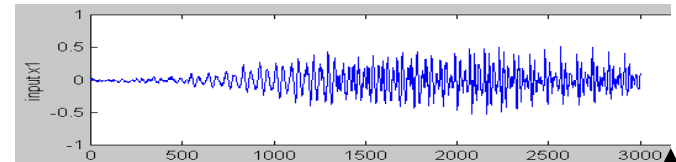
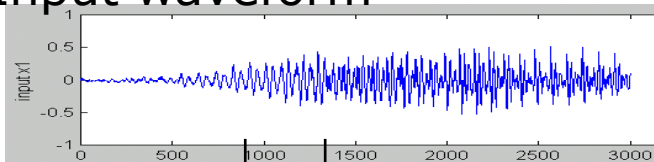


For vowels (voiced sound), use LPC to represent the signal

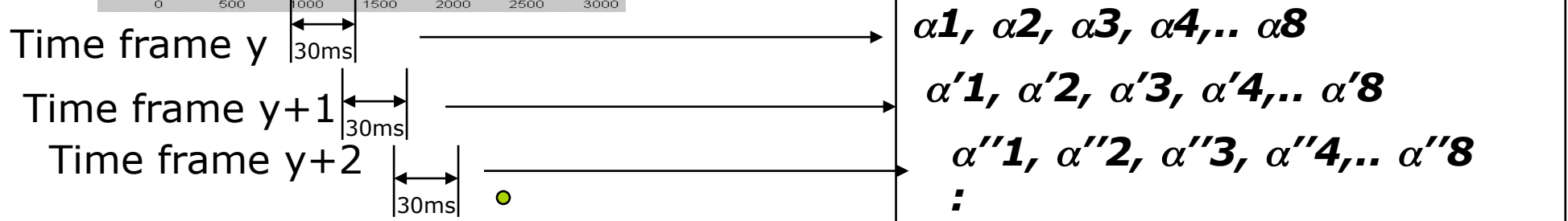
- The concept is to find a set of parameters ie. $\alpha_1, \alpha_2, \alpha_3, \alpha_4, \dots, \alpha_{p=8}$ to represent the same waveform (typical values of $p=8 \rightarrow 13$)

For example

Input waveform



Can reconstruct the waveform from these LPC codes



Each time frame $y=512$ samples

$(S_0, S_1, S_2, \dots, S_n, S_{N-1}=511)$

512 floating points

Each set has 8 floating points

Concept: we want to find a set of a_1, a_2, \dots, a_8 , so when applied to all S_n in this frame ($n=0, 1, \dots, N-1$), the total error E ($n=0 \rightarrow N-1$) is minimum



Predicted \tilde{s}_n at n using past history

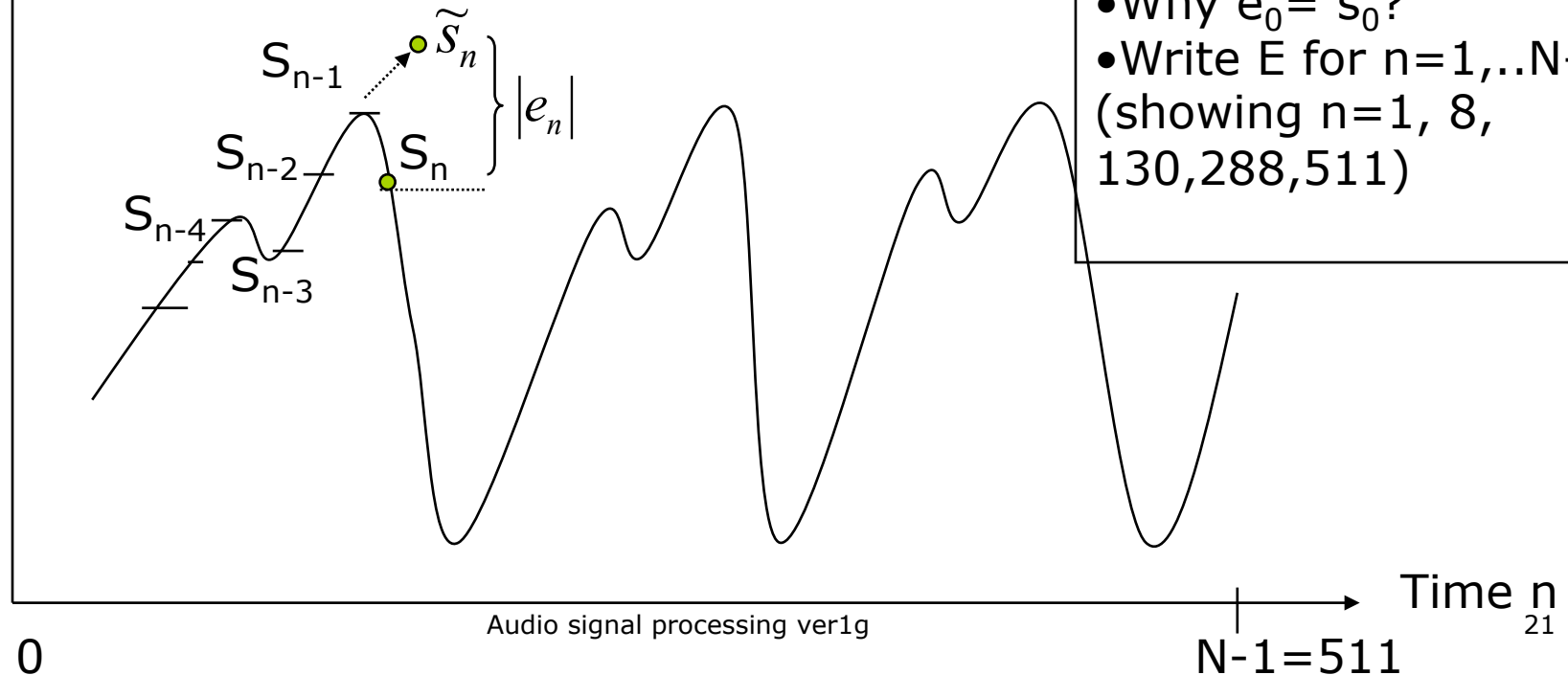
$$\tilde{s}_n = a_1 s_{n-1} + a_2 s_{n-2} + a_3 s_{n-3} + \dots + a_p s_{n-p}$$

predicted error at $n = e_n = s_n - \tilde{s}_n$

so the whole segment $n = 0$ to $N - 1$

$$E = \sum_{n=0}^{n=N-1} (e_n)^2$$

S
Signal level



Exercise 5

- Write the error function e_n at $N=130$, draw it on the graph
- Write the error function at $N=288$
- Why $e_0 = s_0$?
- Write E for $n=1, \dots, N-1$, (showing $n=1, 8, 130, 288, 511$)

Answers

- Write error function at $N=130$, draw e_n on the graph

$$e_{130} = s_{130} - \tilde{s}_{130} = s_{130} - (a_1 s_{129} + a_2 s_{128} + a_3 s_{127} + \dots + a_p s_{130-8=122})$$

- Write the error function at $N=288$

$$e_{288} = s_{288} - \tilde{s}_{288} = s_{288} - (a_1 s_{287} + a_2 s_{286} + a_3 s_{285} + \dots + a_p s_{288-8=280})$$

- Why $e_1 = 0$?

- Answer: Because $s_{-1}, s_{-2}, \dots, s_{-8}$ are outside the frame and are considered as 0. The effect to the overall solution is very small.

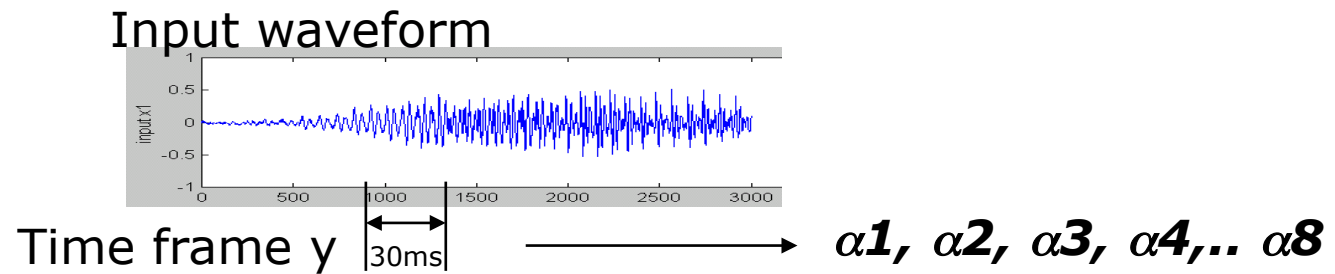
- Write E for $n=1, \dots, N-1$, (showing $n=1, 8, 130, 288, 511$)

$$E = [s_0 - \tilde{s}_0]^2 + [s_1 - \tilde{s}_1]^2 + \dots + [s_8 - \tilde{s}_8]^2 + \dots + [s_{130} - \tilde{s}_{130}]^2 + \dots + [s_{288} - \tilde{s}_{288}]^2 + \dots + [s_{511} - \tilde{s}_{511}]^2$$

LPC idea and procedure

- ❑ The idea: from all samples $s_0, s_1, s_2, \dots, s_{N-1=511}$, we want to find a_p ($p=1, 2, \dots, 8$), so that E is a minimum. The periodicity of the input signal provides information for finding the result.
- ❑ Procedures
 - For a speech signal, we first get the signal frame of size $N=512$ by windowing (will discuss later).
 - Sampling at 25.6KHz, it is equal to a period of 20ms.
 - The signal frame is $(s_0, s_1, s_2, \dots, s_{N-1=511})$.
 - Ignore the effect of outside elements by setting them to zero, i.e. $s_{-\infty} \dots s_{-2} = s_{-1} = s_{512} = s_{513} = \dots = s_{\infty} = 0$ etc.
 - We want to calculate LPC parameters of order $p=8$, i.e. $\alpha_1, \alpha_2, \alpha_3, \alpha_4, \dots, \alpha_{p=8}$.

For each
30ms time
frame



The predicted value s_n is denoted by \tilde{s}_n

$$\tilde{s}_n = a_1 s_{n-1} + a_2 s_{n-2} + a_3 s_{n-3} + \dots + a_p s_{n-p} \dots (1)$$

prediction error

$$e_n = s_n - \tilde{s}_n, \text{ from (1)}$$

$$e_n = s_n - (a_1 s_{n-1} + a_2 s_{n-2} + a_3 s_{n-3} + \dots + a_p s_{n-p})$$

$$e_n = s_n - \sum_{i=1}^{i=p} a_i s_{n-i},$$

so the whole frame $n = 0$ to $N-1$

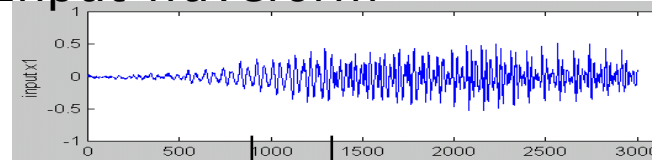
$$E = \sum_{n=0}^{n=N-1} (e_n)^2 = \sum_{n=0}^{n=N-1} \left(s_n - \sum_{i=1}^{i=p} a_i s_{n-i} \right)^2$$

To find $a_{i=1,2,\dots,p}$ that generate E_{\min} , solve $\frac{\partial E}{\partial a_i} = 0$ for all $i = 1, 2, \dots, p$

Solve for

$a_{1,2,\dots,p}$

Input waveform



Time frame y $\xrightarrow{30\text{ms}}$ $\alpha 1, \alpha 2, \alpha 3, \alpha 4, \dots \alpha 8$

□ To find $a_{i=1,2,\dots,p}$, that generate E_{\min} , solve $\frac{\partial E}{\partial a_i} = 0$ for all $i = 1, 2, \dots, p$

After some manipulations we have

Derivations can be found at
http://www.cslu.ogi.edu/people/hosom/cs552/lecture07_features.ppt

$$\begin{bmatrix} r_0 & r_1 & r_2 & \dots & r_{p-1} \\ r_1 & r_0 & r_1 & \dots & r_{p-2} \\ r_2 & r_1 & r_0 & \dots & : \\ : & : & : & \dots & : \\ r_{p-1} & r_{p-2} & r_{p-3} & \dots & r_0 \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \\ : \\ : \\ a_p \end{bmatrix} = \begin{bmatrix} r_1 \\ r_2 \\ : \\ : \\ r_p \end{bmatrix} \quad \text{---(2)}$$

Use Durbin's equation to solve this

$$r_0 = \sum_{n=0}^{N-1-0} (s_n \cdot s_n), \quad r_i = \sum_{n=0}^{N-1-i} (s_n \cdot s_{n+i}) = \text{auto-correlation functions}$$

If we know $r_0, r_1, r_2, \dots, r_p$, we can find out a_1, a_2, \dots, a_p by the set of equations in (2)

The example

- For each time frame (25 ms), data is valid only inside the window.
- 20.48 KHZ sampling, a window frame (25ms) has 512 samples (N)
- Require 8-order LPC, $i=1,2,3,..8$
- calculate using $r_0, r_1, r_2,.. r_8$, using the above formulas, then get LPC parameters $a_1, a_2,.. a_8$ by the Durbin recursive Procedure.

Steps for each time frame to find a set of LPC

- (step1) $N = \text{WINDOW} = 512$, the speech signal is s_0, s_1, \dots, s_{511}
- (step2) Order of LPC is 8, so r_0, r_1, \dots, r_8 required are:

$$r_0 = s_0s_0 + s_1s_1 + s_2s_2 + s_3s_3 + s_4s_4 + \dots + s_{511}s_{511}$$

$$r_1 = s_0s_1 + s_1s_2 + s_2s_3 + s_3s_4 + s_4s_5 + \dots + s_{510}s_{511}$$

$$r_2 = s_0s_2 + s_1s_3 + s_2s_4 + s_3s_5 + s_4s_6 + \dots + s_{509}s_{511}$$

$$r_3 = s_0s_3 + s_1s_4 + s_2s_5 + s_3s_6 + s_4s_7 + \dots + s_{508}s_{511}$$

\Downarrow

$$r_7 = s_0s_7 + s_1s_8 + s_2s_9 + s_3s_{10} + s_4s_{11} + \dots + s_{504}s_{511}$$

$$r_8 = s_0s_8 + s_1s_9 + s_2s_{10} + s_3s_{11} + s_4s_{12} + \dots + s_{503}s_{511}$$

- (step3) Solve the set of linear equations (previous slide)

Program segmentation algorithm for auto-correlation

- ❑ WINDOW=size of the frame; coeff = autocorrelation matrix; sig = input, ORDER = lpc order
- ❑ void autocorrelation(float *sig, float *coeff)
- ❑ {int i,j;
- ❑ for (i=0;i<=ORDER;i++)
- ❑ {
- ❑ coeff[i]=0.0;
- ❑ for (j=i;j<WINDOW;j++)
- ❑ coeff[i]+= sig[j]*sig[j-i];
- ❑ }
- ❑ }

To calculate LPC $a[]$ from auto-correlation matrix *coef using Durbin's Method (solve equation 2)

$$\begin{bmatrix} r_0 & r_1 & r_2 & \dots & r_{p-1} \\ r_1 & r_0 & r_1 & \dots & r_{p-2} \\ r_2 & r_1 & r_0 & \dots & \vdots \\ \vdots & \vdots & \vdots & \dots & \vdots \\ r_{p-1} & r_{p-2} & r_{p-3} & \dots & r_0 \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \\ \vdots \\ \vdots \\ a_p \end{bmatrix} = \begin{bmatrix} r_1 \\ r_2 \\ \vdots \\ \vdots \\ r_p \end{bmatrix} \quad \text{---(2)}$$

- ❑ void lpc_coeff(float *coeff)
- ❑ {int i, j; float sum,E,K,a[ORDER+1][ORDER+1];
- ❑ if(coeff[0]==0.0) coeff[0]=1.0E-30;
- ❑ E=coeff[0];
- ❑ for (i=1;i<=ORDER;i++)
- ❑ { sum=0.0;
- ❑ for (j=1;j<i;j++) sum+= a[j][i-1]*coeff[i-j];
- ❑ K=(coeff[i]-sum)/E; a[i][i]=K; E*=(1-K*K);
- ❑ for (j=1;j<i;j++) a[j][i]=a[j][i-1]-K*a[i-j][i-1];
- ❑ }
- ❑ for (i=1;i<=ORDER;i++) coeff[i]=a[i][ORDER];}

Cepstrum



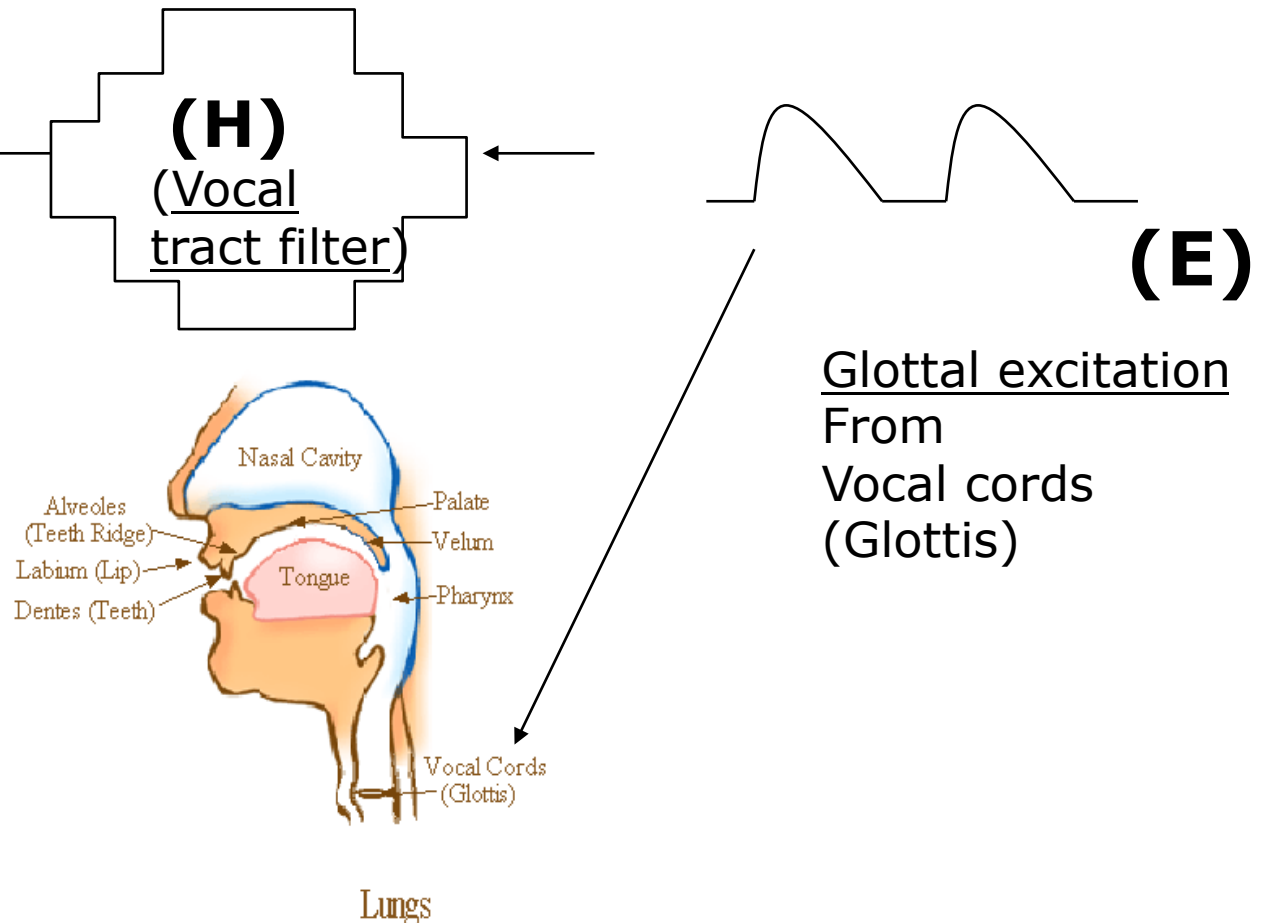
A new word by reversing the first 4 letters of spectrum → cepstrum.
It is the spectrum of a spectrum of a signal

Glottis and Cepstrum

Speech wave (X) = Excitation (E) . Filter (H)

- Output **(S)**
So voice has a strong glottis
Excitation
Frequency content

In Cepstrum
We can easily
identify and
remove the glottal
excitation



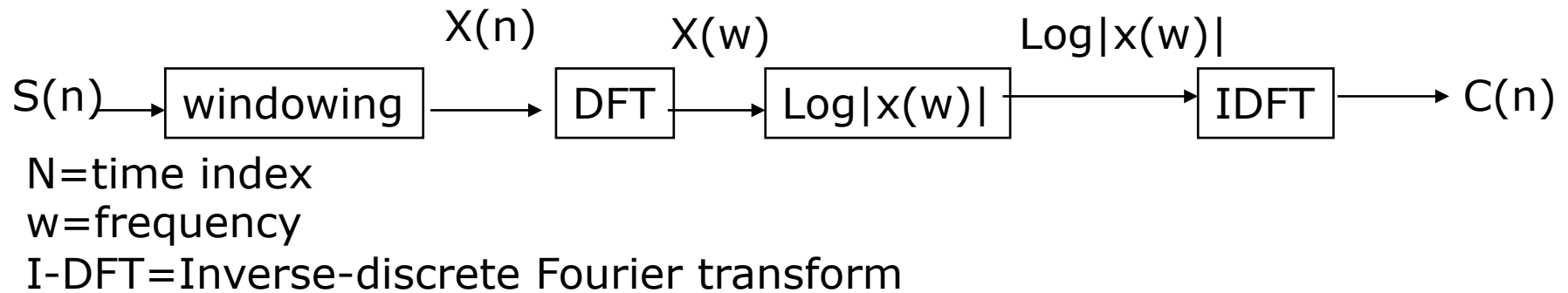
<http://home.hib.no/al/engelsk/seksjon/SOFF-MASTER/ill061.gif>

Cepstral analysis

- Signal(s)=convolution(*) of
 - glottal excitation (e) and vocal_tract_filter (h)
 - $s(n)=e(n)*h(n)$, n is time index
- After Fourier transform FT: $FT\{s(n)\}=FT\{e(n)*h(n)\}$
 - Convolution(*) becomes multiplication (.)
 - $n(\text{time}) \rightarrow w(\text{frequency})$,
- $S(w) = E(w).H(w)$
- Find Magnitude of the spectrum
- $|S(w)| = |E(w)|.|H(w)|$
- $\log_{10} |S(w)| = \log_{10}\{|E(w)|\} + \log_{10}\{|H(w)|\}$

Cepstrum

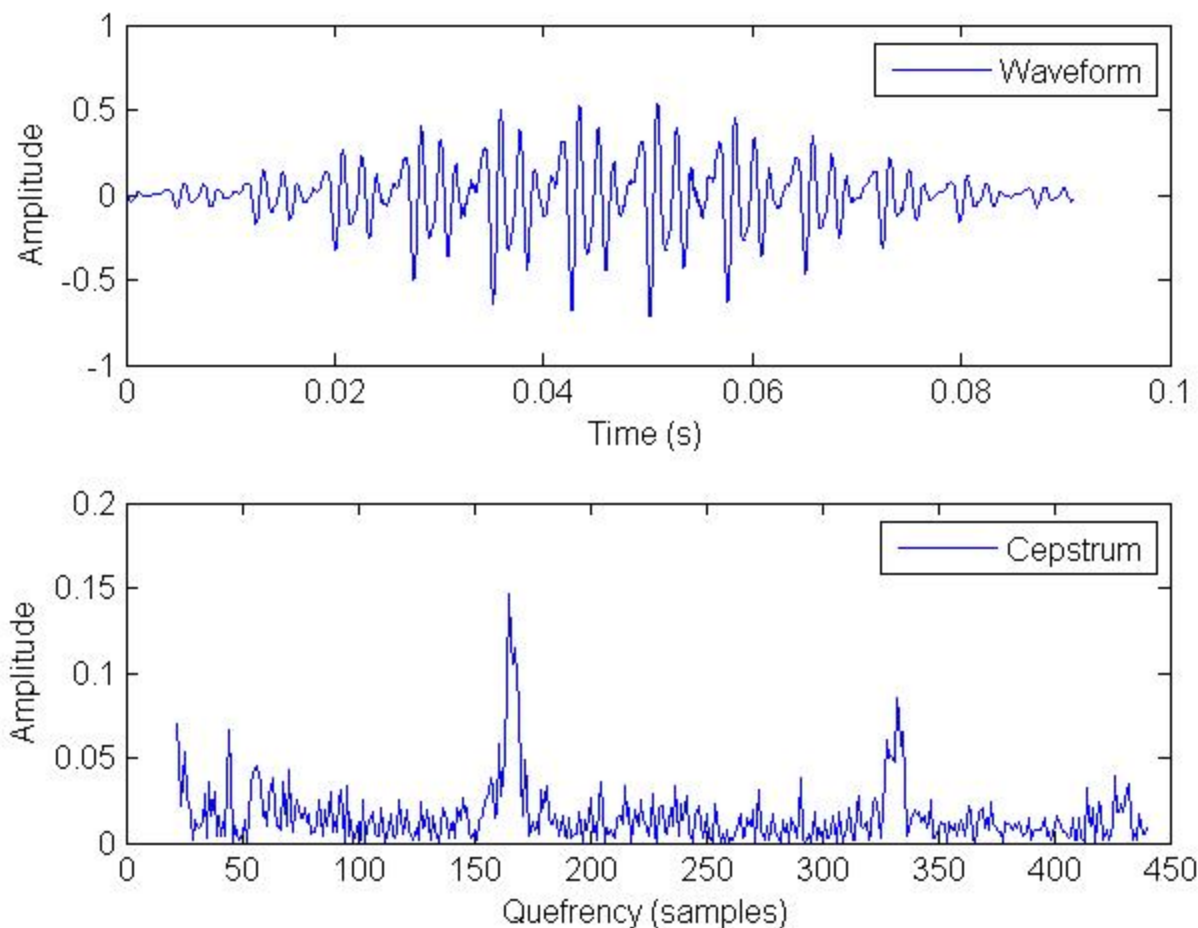
- $C(n) = \text{IDFT}[\log_{10} |S(w)|] =$
- $\text{IDFT}[\log_{10}\{|E(w)|\} + \log_{10}\{|H(w)|\}]$



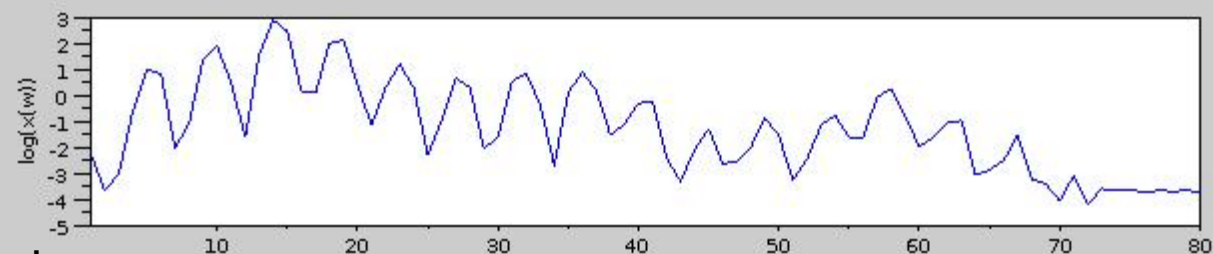
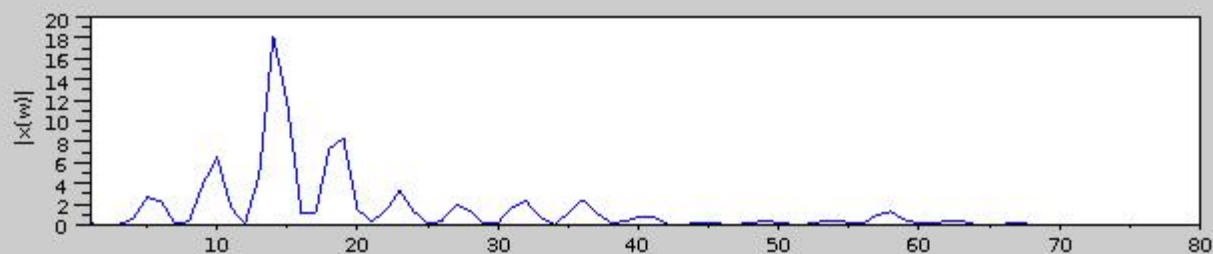
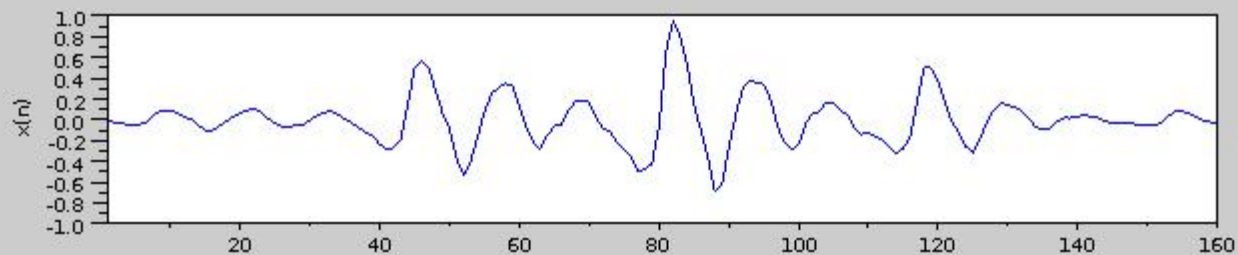
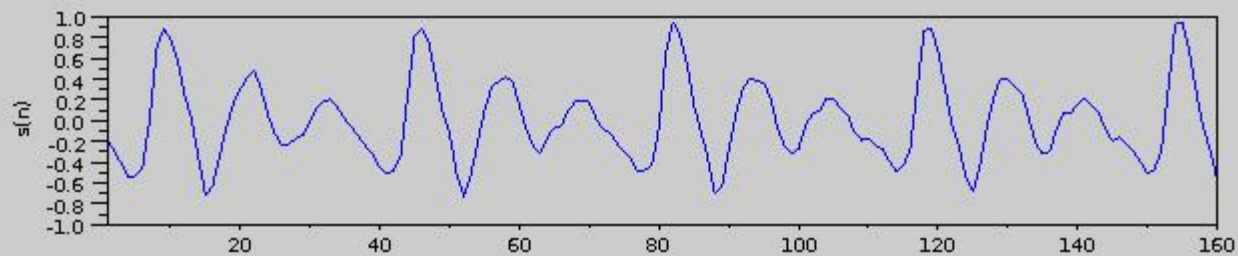
- In $c(n)$, you can see $E(n)$ and $H(n)$ at two different positions
- Application: useful for (i) glottal excitation (ii) vocal tract filter analysis

Example of cepstrum using spCepstrumDemo.m on sor1.wav

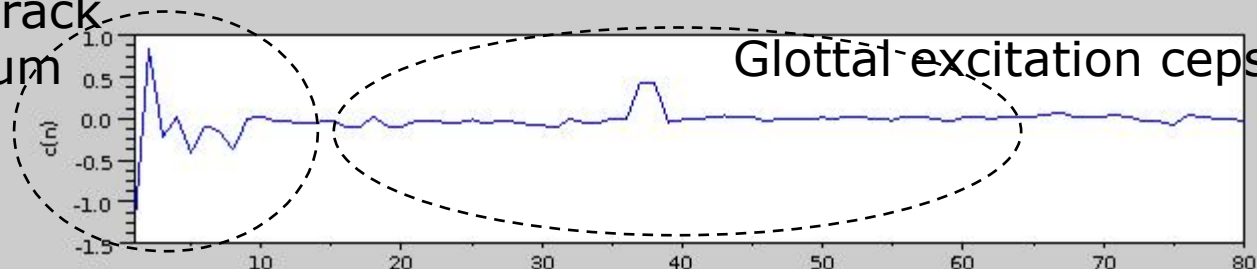
'sor1.wav' = sampling frequency 22.05KHz



Examples



Vocal track
cepstrum



Glottal excitation cepstrum

Liftering

□ Low time liftering:

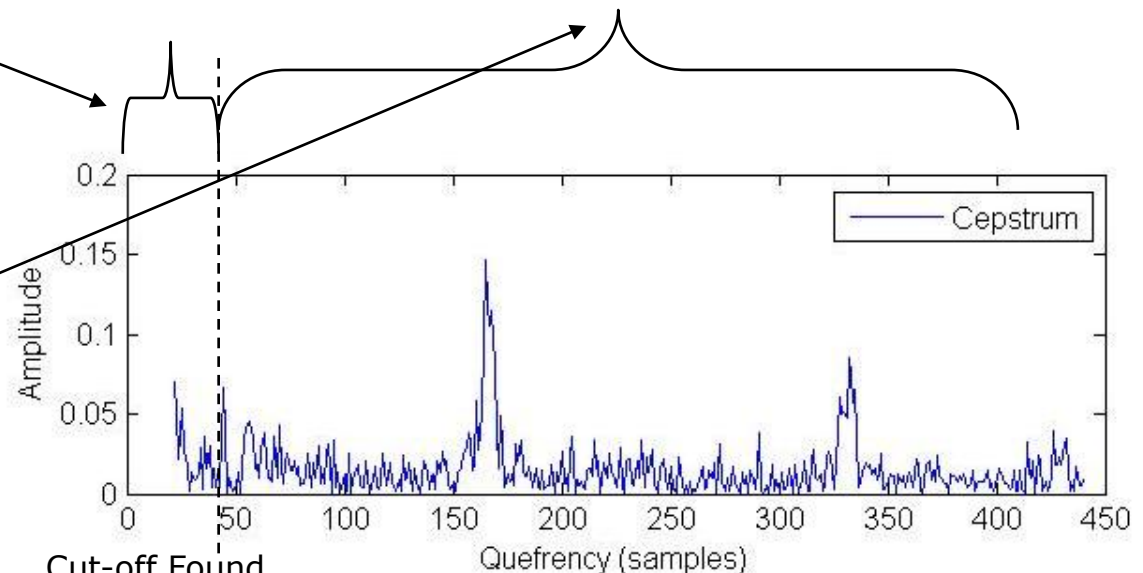
- Magnify (or Inspect) the low time to find the vocal tract cepstrum

Vocal tract
Cepstrum
Used for
Speech
recognition

Glottal excitation
Cepstrum, useless for
speech recognition,

□ High time liftering:

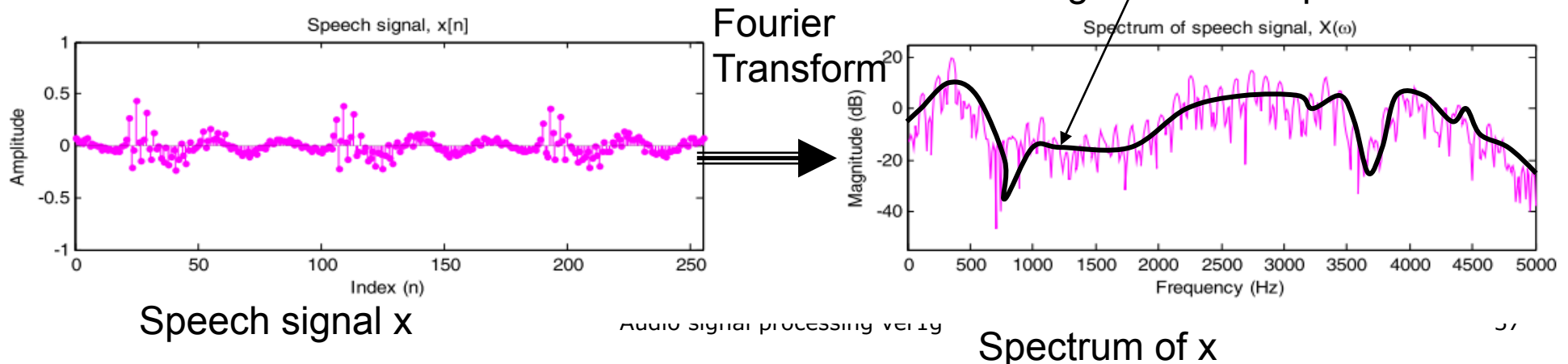
- Magnify (or Inspect) the high time to find the glottal excitation cepstrum (remove this part for speech recognition).



Reasons for liftering Cepstrum of speech

- Why we need this?
 - Answer: remove the ripples
 - of the spectrum caused by
 - glottal excitation.

Too many ripples in the spectrum caused by vocal cord vibrations. But we are more interested in the speech envelope for recognition and reproduction



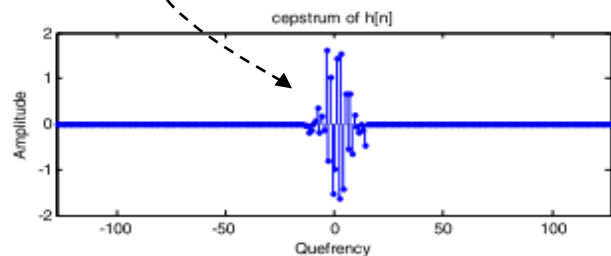
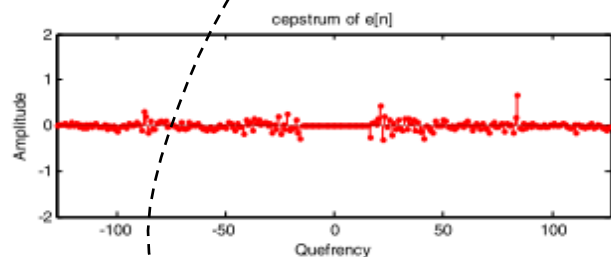
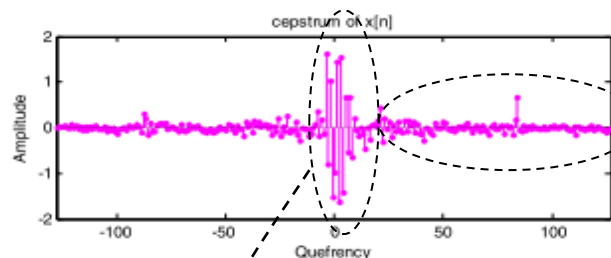
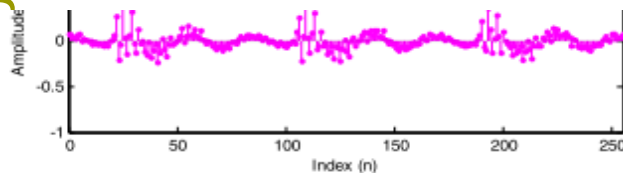
Liftering method: Select the high time and low time liftering

Signal X

Cepstrum

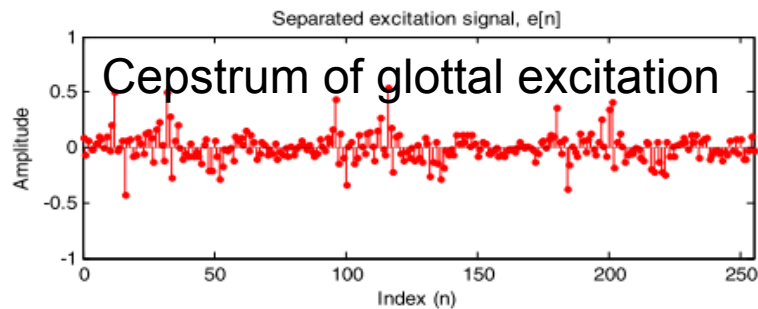
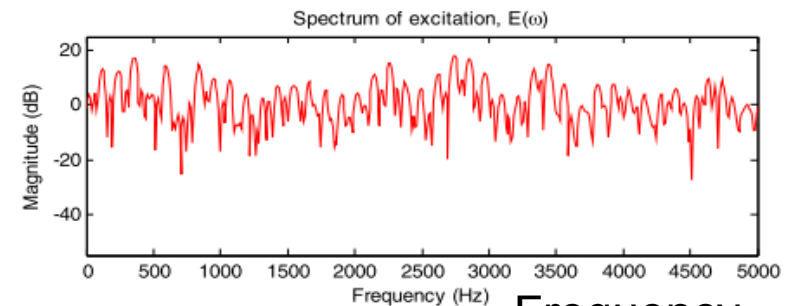
Select high
time, C_{high}

Select low
time
 C_{low}



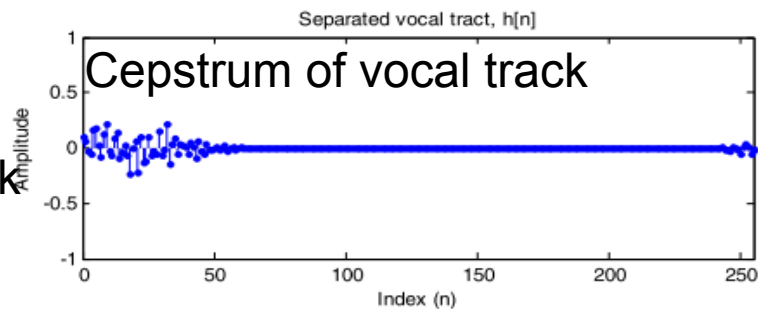
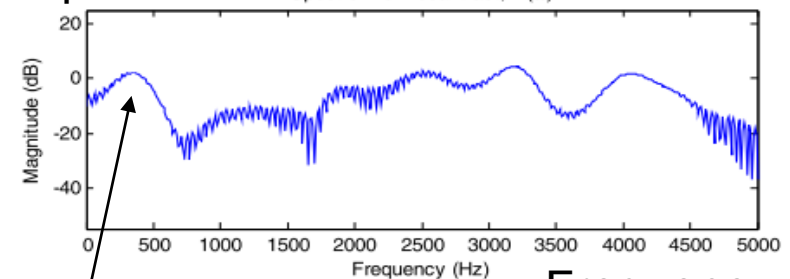
Recover Glottal excitation and vocal track spectrum

Spectrum of glottal excitation



C_high
For
Glottal
excitation

Spectrum of vocal track filter



C_high
For
Vocal track

quefrequency (sample index)

This peak may be the pitch period:
This smoothed vocal track spectrum can
be used to find pitch

For more information see :

<http://isdl.ee.washington.edu/people/stevenschimmel/sphsc503/files/notes10.pdf>

Exercise 6

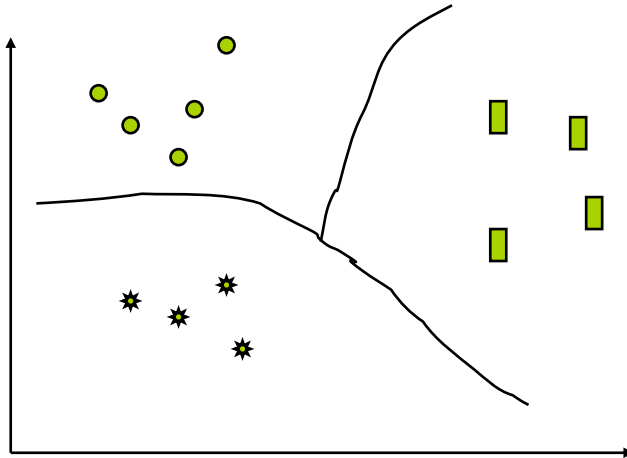
- A speech waveform S has the values $s_0, s_1, s_2, s_3, s_4, s_5, s_6, s_7, s_8 = [1, 3, 2, 1, 4, 1, 2, 4, 3]$. The frame size is 4.
 - Find the pre-emphasized wave if $\tilde{a} =$ is 0.98.
 - Find auto-correlation parameter r_0, r_1, r_2 .
 - If we use LPC order 2 for our feature extraction system, find LPC coefficients a_1, a_2 .
 - If the number of overlapping samples for two frames is 2, find the LPC coefficients of the second frame.

3.3 Vector Quantization (VQ)

- ❑ Vector quantization is a data compression method
 - raw speech 10KHz/8-bit data for a 30ms frame is 300 bytes
 - 10th order LPC = 10 floating numbers = 40 bytes
 - after VQ it can be as small as one byte.
- ❑ Used in tele-communication systems.
- ❑ Enhance recognition systems since less data is involved.

Use of Vector quantization for Further compression

- ❑ LPC=10, is a data in a 10 dimensional space
- ❑ after VQ it can be as small as one byte.
- ❑ Example, in LPC2 (2 D space)



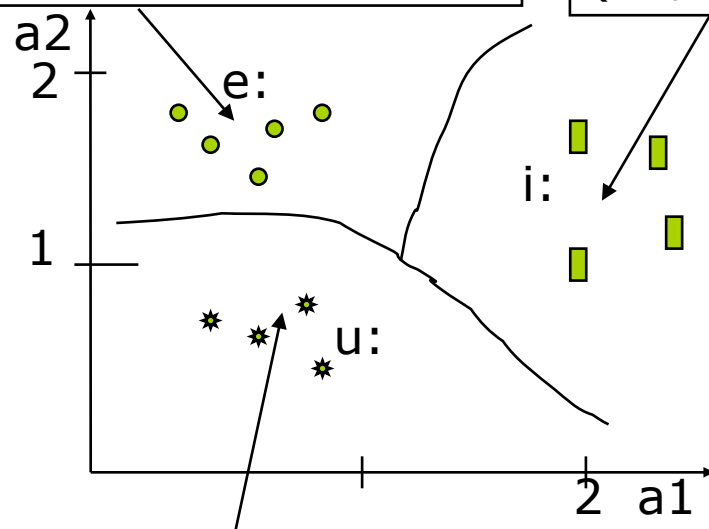
3.3 Vector Quantization (VQ)

A simple example, 2nd order LPC, LPC2

- We can classify speech sound segments by Vector quantization
- Make a table

The standard sound is the centroid of all samples of e:
(a1,a2)=(0.5,1.5)

The standard sound is the centroid of all samples of i:
(a1,a2)=(2,1.3)



code		a1	A2
1	e:	0.5	1.5
2	i:	2	1.3
3	u:	0.7	0.8

Using this table, 2 bits are enough to encode each sound

Feature space and sounds are classified into three different types
e:, i: , u:

The standard sound is the centroid of all samples of u:, (a1,a2)=(0.7,0.8)

Another example LPC8

- 256 different sounds encoded by the table (one segment which has 512 samples is represented by one byte)
- Use many samples to find the centroid of that sound, i. e.: , or i:
- Each row is the centroid of that sound in LPC8.
- In telecomm., the transmitter only transmits the code (1 segment using 1 byte), the receiver reconstructs the sound using that code and the table. The table is only transmitted once.

One segment (512 samples) compressed into 1 byte
 transmitter -----> receiver

Code (1 byte)	a1	a2	a3	a4	a5	a6	a7	a8
0=(e:)	1.2	8.4	3.3	0.2
1=(i:)
2=(u:)								
:								
255

VQ techniques, M code-book vectors from L training vectors

- K-means clustering algorithm
 - Arbitrarily choose M vectors
 - Nearest Neighbor search
 - Centroid update and reassignment, back to above statement until error is minimum.
- Binary split with K-means clustering algorithm, this method is more efficient.

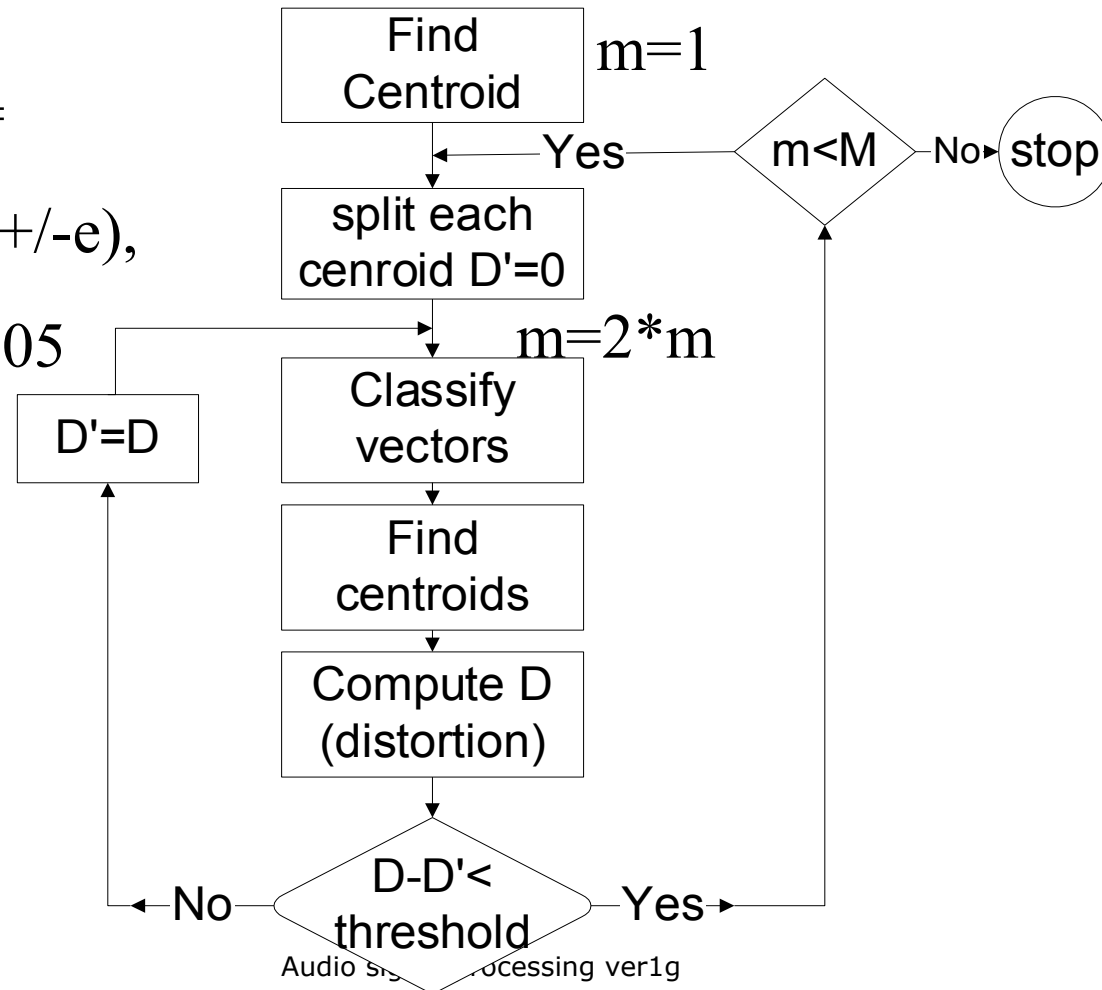
Binary split code-book: (assume you use all available samples in building the centroids at all stages of calculations)

Split function:

$\text{new_centroid} =$

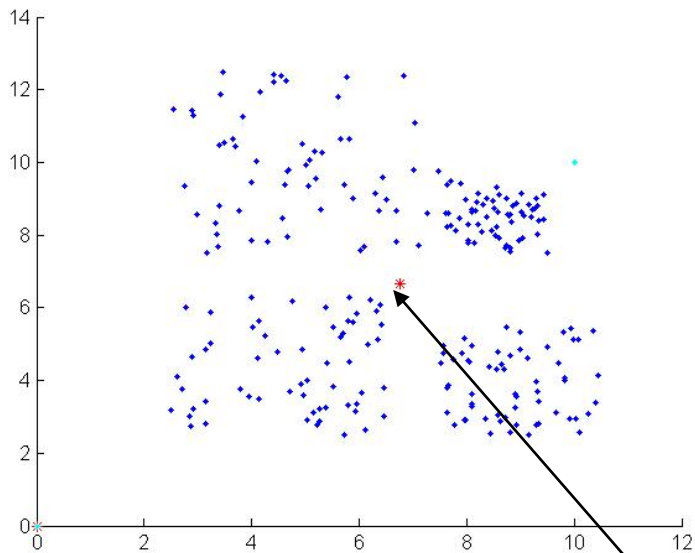
$\text{old_centroid}(1 \pm e),$

for $0.01 \leq e \leq 0.05$

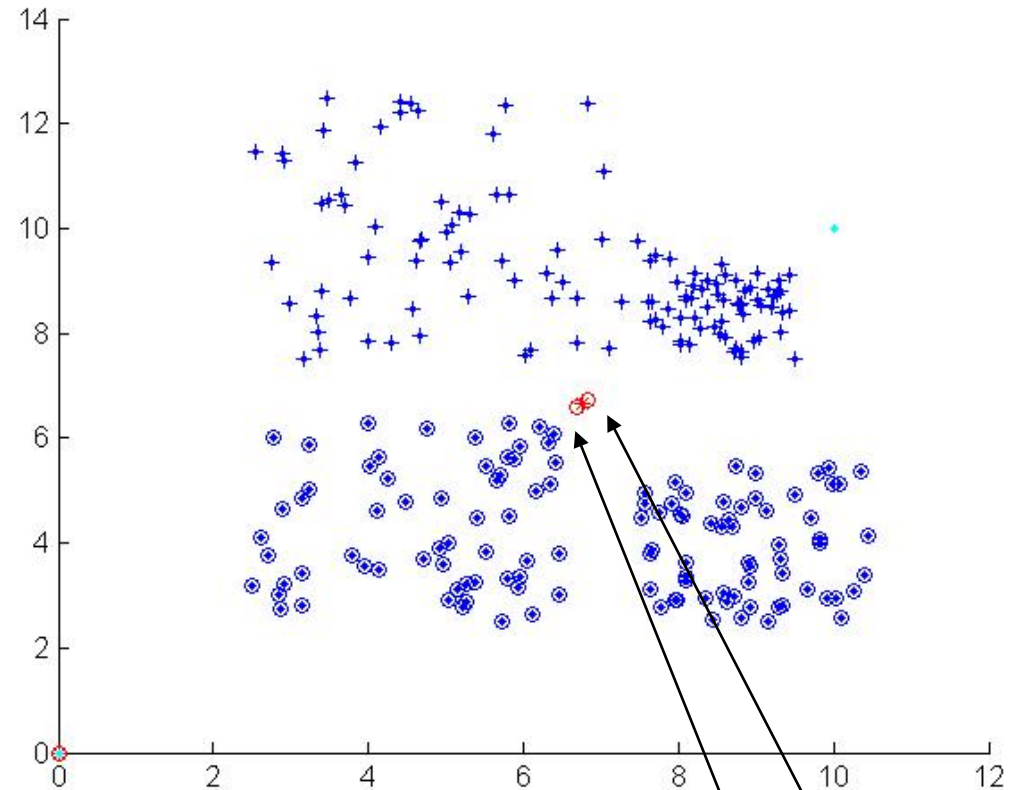


Example: VQ : 240 samples use VQ to split to 4 classes

□ Steps

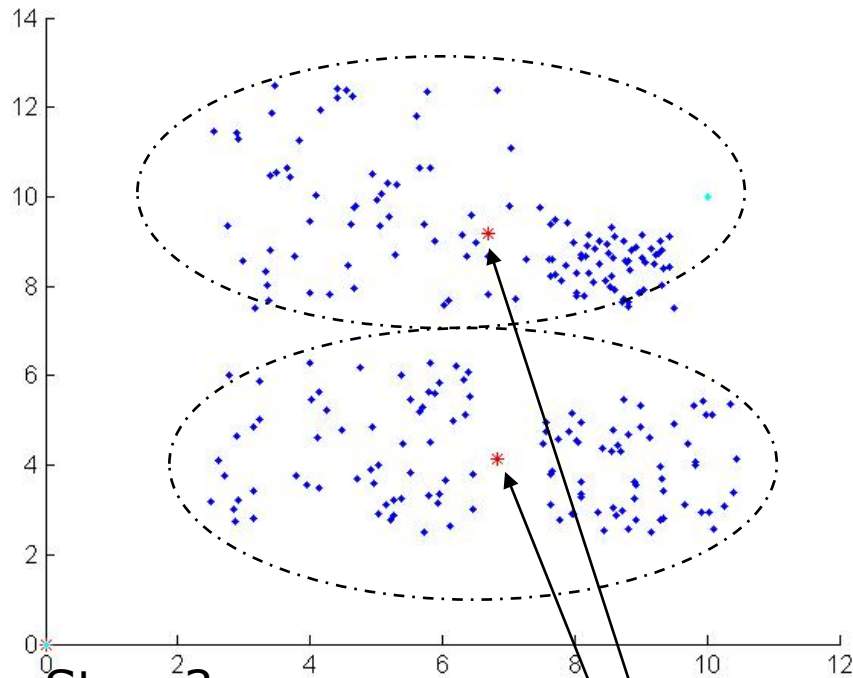


Step1: all data find centroid C
 $C1 = C(1+e)$
 $C2 = C(1-e)$



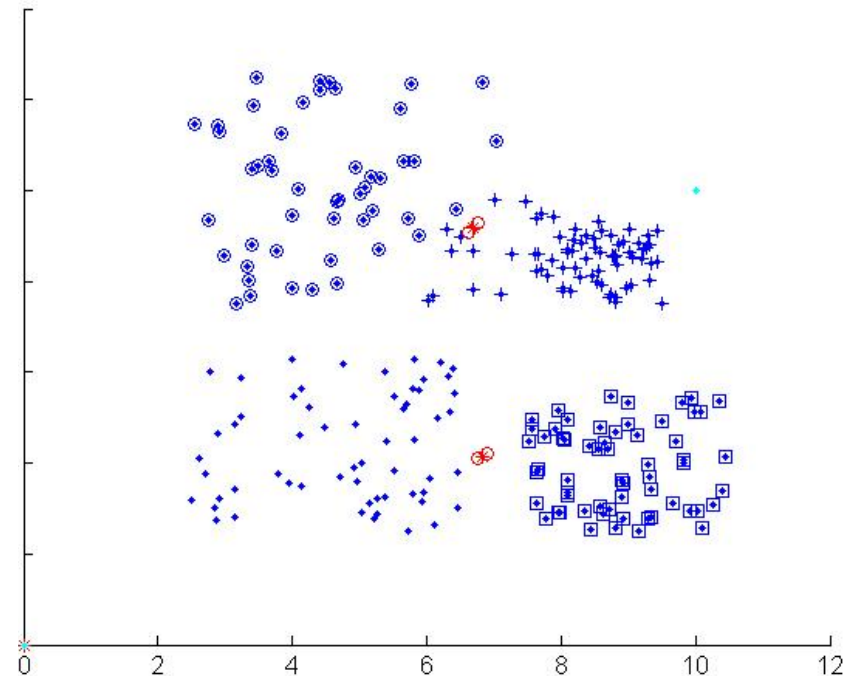
Step2:
• split the centroid into two $C1, C2$
• Regroup data into two classes according to the two new centroids $C1, C2$

continue



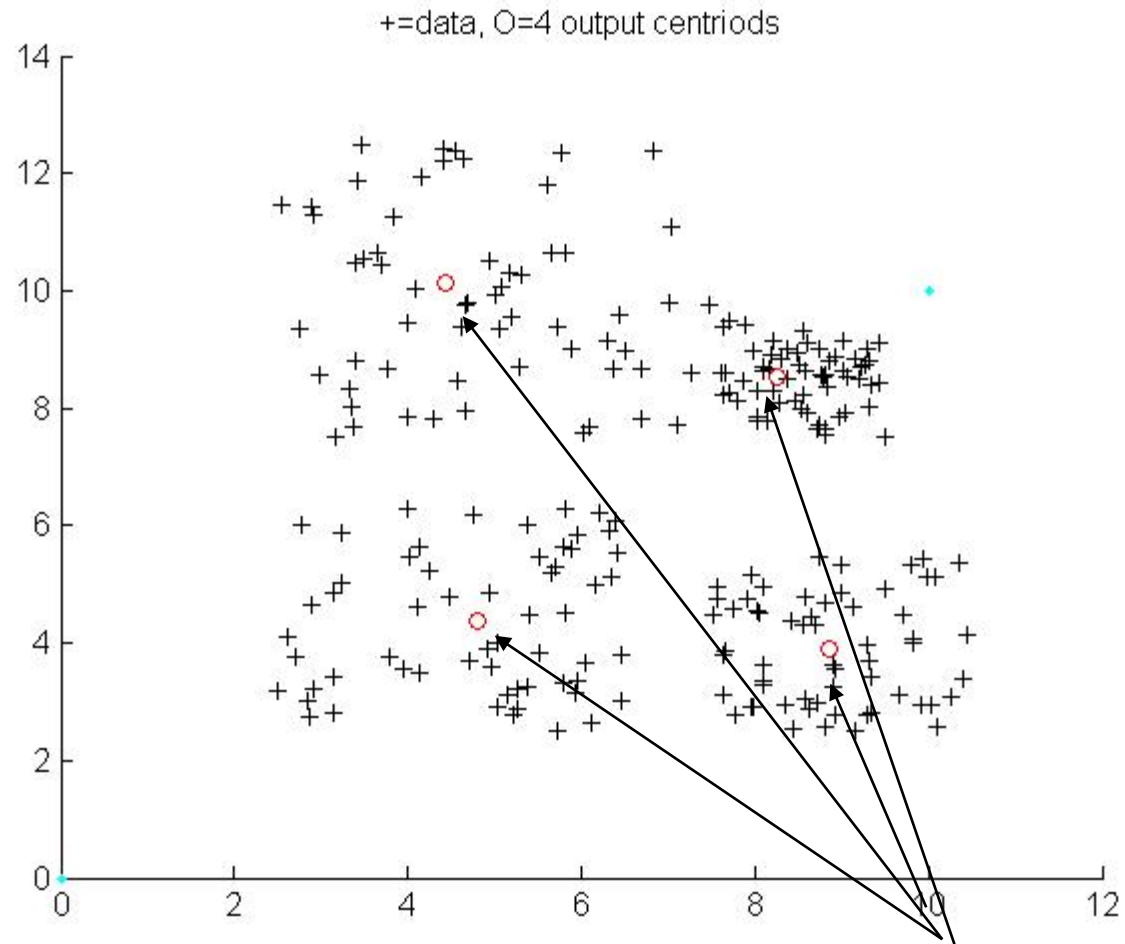
Stage 3:

- Update the 2 centroids according to the two splitted groups
- Each group find a new centroid.



Stage 4: split the 2 centroids again to become 4 centroids

Final result



Stage 5: regroup and update the 4 new centroids, done. 49

Tutorials for VQ

- Given 4 speech frames, each is described by a 2-D vector (x,y) as below.
- $P_1=(1.2,8.8); P_2=(1.8,6.9); P_3=(7.2,1.5); P_4=(9.1,0.3)$
- Find the code-book of size two using K-means method. (Answer see Appendix A.1)
- Write Pseudo code (or a C program segment) to build the code book of size 4 from 100 2D-vectors, the input vectors (x,y) are stored in `int x[100]` and `int y[100]`.

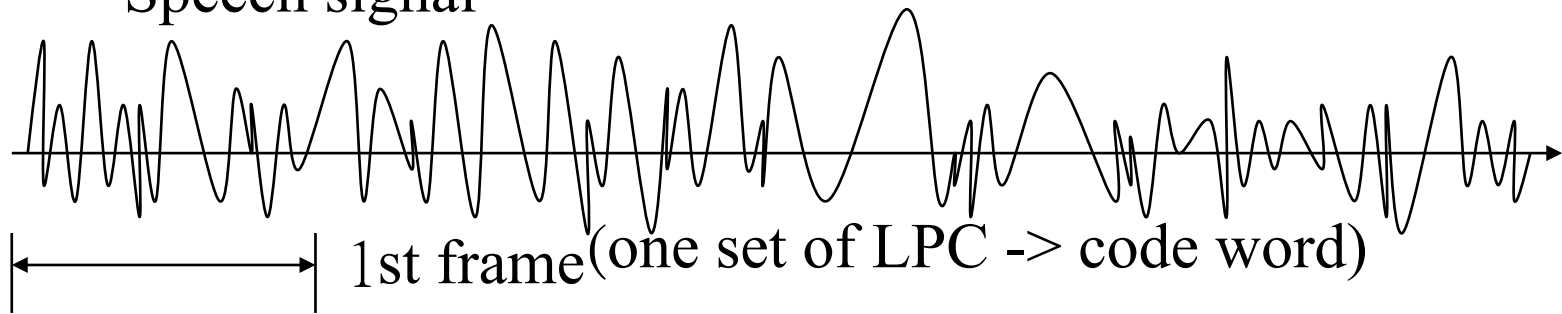
Exercise 7

- Given 4 speech frames, each is described by a 2-D vector (x,y) as below.
- $P1=(1.2,8.8)$; $P2=(1.8,6.9)$; $P3=(7.2,1.5)$; $P4=(9.1,0.3)$.
 - Use K-means method to find the two centroids.
 - Use Binary split K-means method to find the two centroids. Assume you use all available samples in building the centroids at all stages of calculations
 - A raw speech signal is sampled at 10KHz/8-bit. Estimate compression ratio ($=\text{raw data storage}/\text{compressed data storage}$) if LPC-order is 10 and frame size is 25ms with no overlapping samples.

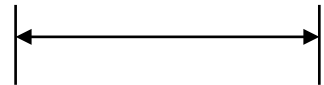
Example of speech signal analysis



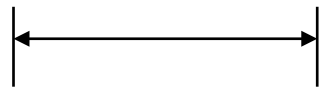
Speech signal



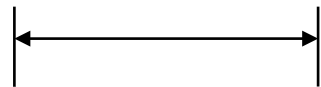
One frame
=N
=512 samples



2nd frame (one set of LPC -> code word)



3rd frame (one set of LPC -> code word)



4th frame (one set of LPC -> code word)



5th frame

Separated
by n samples



Chapter 4 : Recognition Procedures

- Recognition procedure
- Dynamic programming
- HMM

Chapter 4 : Recognition Procedures

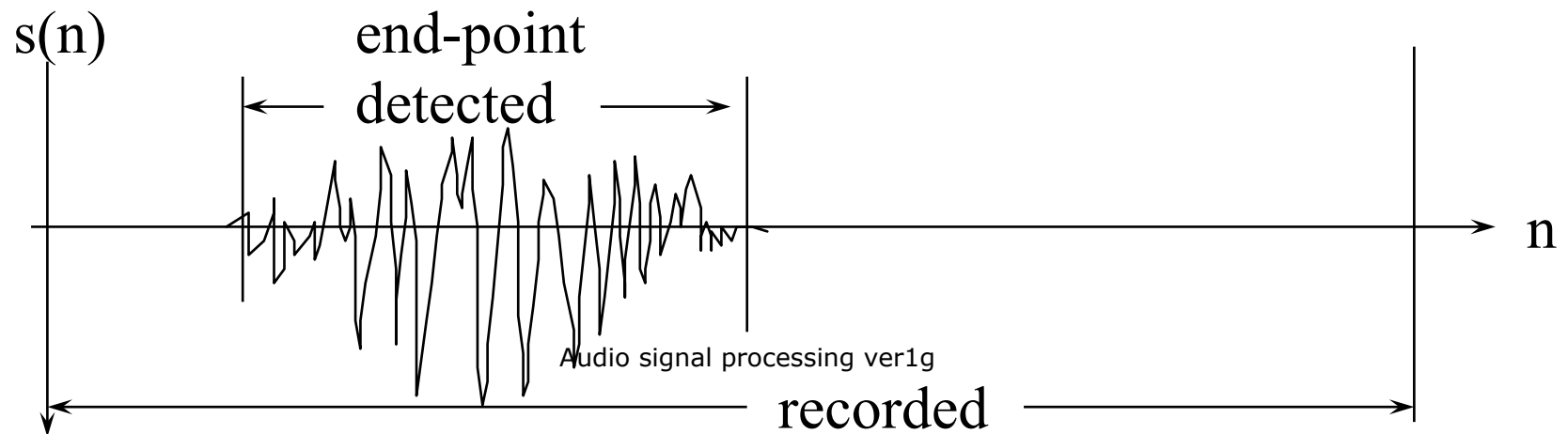
- Preprocessing for recognition
 - endpoint detection
 - Pre-emphasis
 - windowing
 - distortion measure methods
- Comparison methods
 - Vector quantization
 - Dynamic programming
 - Hidden Markov Model

LPC processor for a 10-word isolated speech recognition system

- ❑ End-point detection
- ❑ Pre-emphasis -- high pass filtering
- ❑ Frame blocking and Windowing
- ❑ Auto-correlation analysis
- ❑ LPC analysis,
- ❑ Cepstral coefficients,
- ❑ Weighting
- ❑ Temporal cepstral derivation

End point detection

- To determine the start and end points of the speech sound
 - It is not always easy since the energy of the starting energy is always low.
 - Determined by energy & zero crossing rate



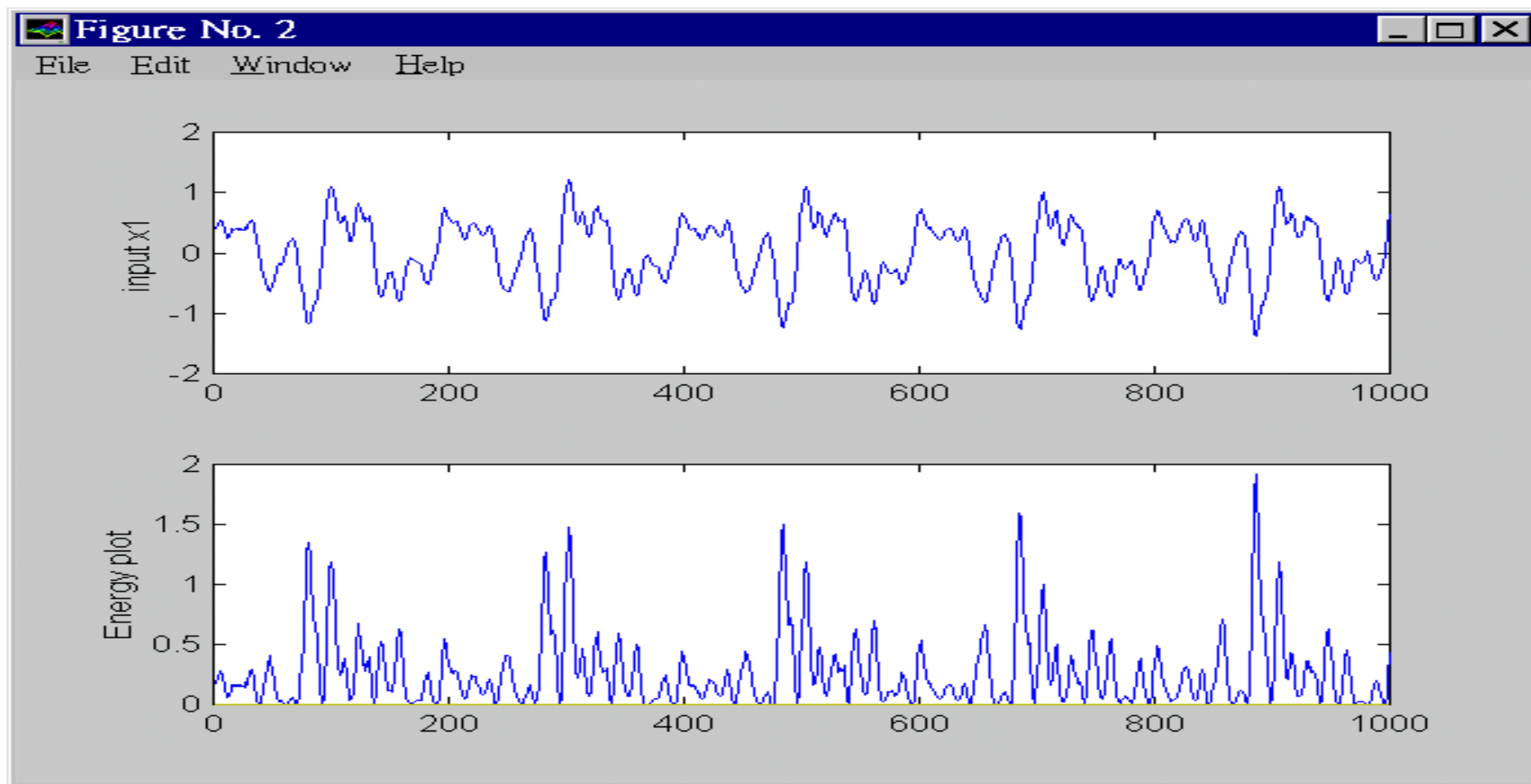
A simple End point detection algorithm

- At the beginning the energy level is low.
- If the energy level and zero-crossing rate of 3 successive frames is high it is a starting point.
- After the starting point if the energy and zero-crossing rate for 5 successive frames are low it is the end point.

Energy calculation

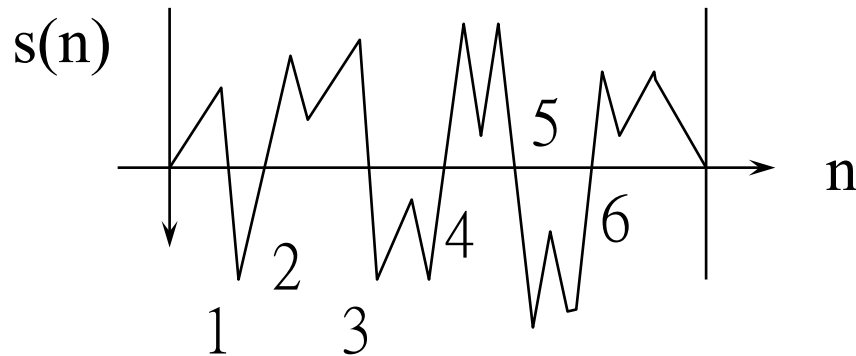
- $E(n) = s(n).s(n)$
- For a frame of size N ,
- The program to calculate the energy level:
 for($n=0;n<N;n++$)
- {
- Energy(n)= $s(n) s(n)$;
- }

Energy plot



Zero crossing calculation

- A zero-crossing point is obtained when
 - $\text{sign}[s(n)] \neq \text{sign}[s(n-1)]$
- The zero-crossing points of $s(n)$ are 6



Pre-emphasis -- high pass filtering

- To reduce noise, average transmission conditions and to average signal spectrum.

$$\tilde{S}(n) = S(n) - \tilde{a}S(n-1)$$

$$0.9 \leq \tilde{a} \leq 1.0, \text{ typically } \tilde{a} = 0.95$$

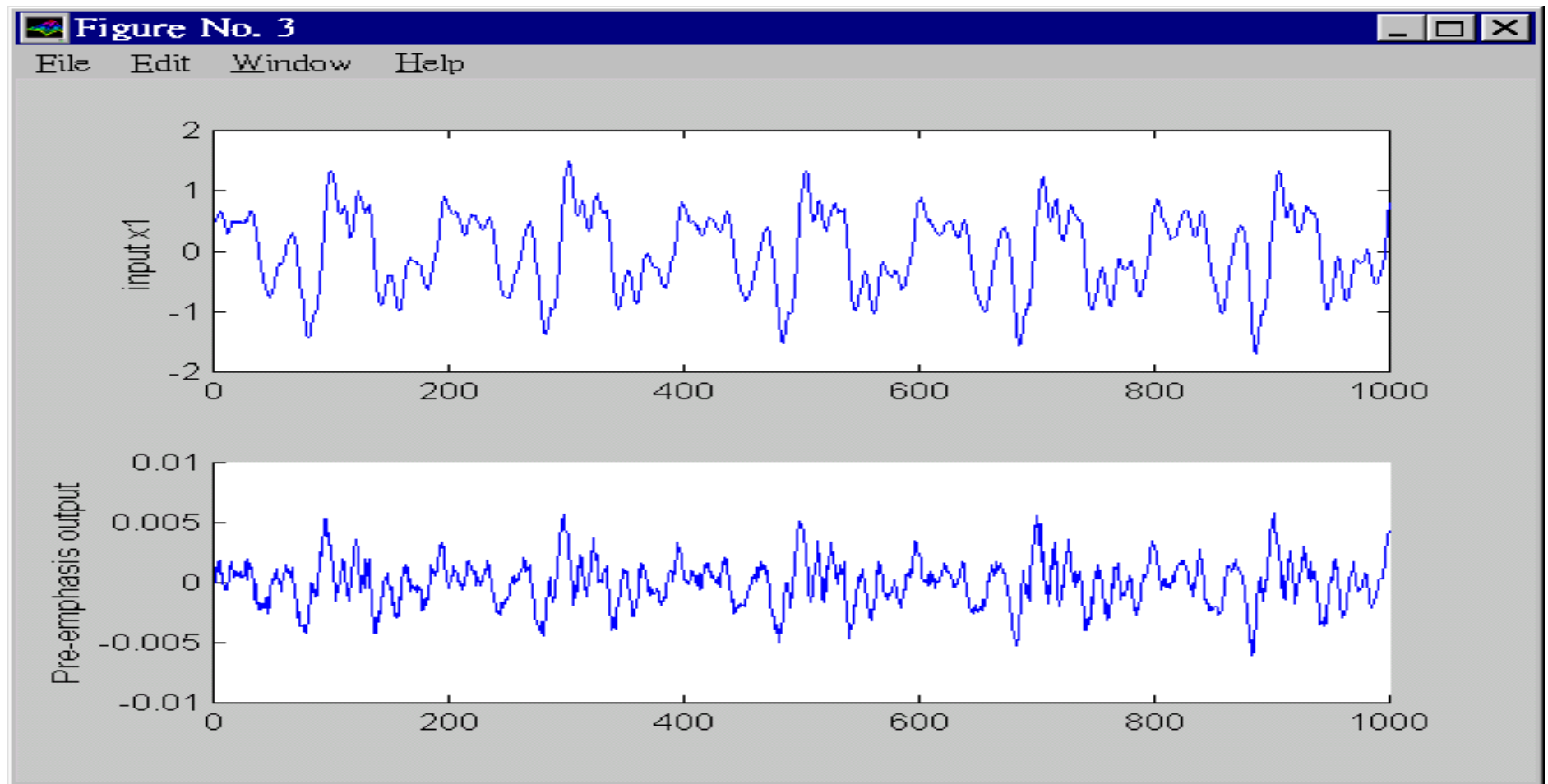
For $S(0), S(1), S(2), \dots$, the value $\tilde{S}(0)$ does not exist and is never used.

- Tutorial: write a program segment to perform pre-emphasis to a speech frame stored in an array `int s[1000]`.

Pre-emphasis program segment

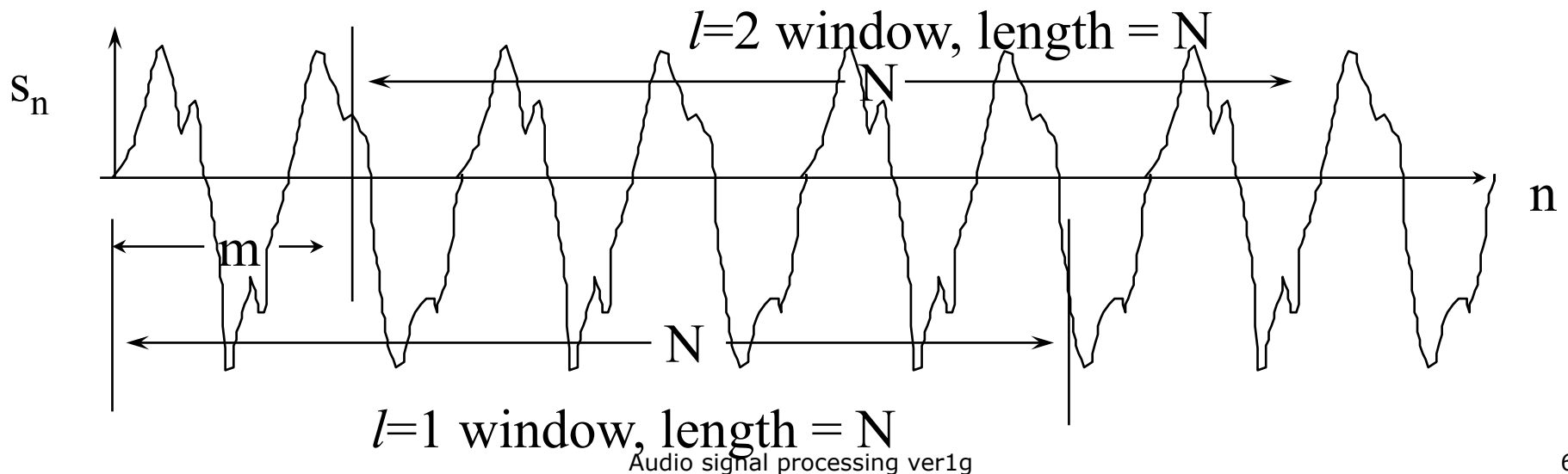
```
□ input=sig1, output=sig2
□ void pre_emphasize(char far *sig1, float
    *sig2)
□ {
□   int j;
□   sig2[0]=(float)sig1[0];
□   for (j=1;j<WINDOW;j++)
□     sig2[j]=(float)sig1[j] -
        0.95*(float)sig1[j-1];
□ }
```

Pre-emphasis



Frame blocking and Windowing

- To choose the frame size (N samples) and adjacent frames separated by m samples.
- I.e.. a 16KHz sampling signal, a 10ms window has $N=160$ samples, $m=40$ samples.



Windowing

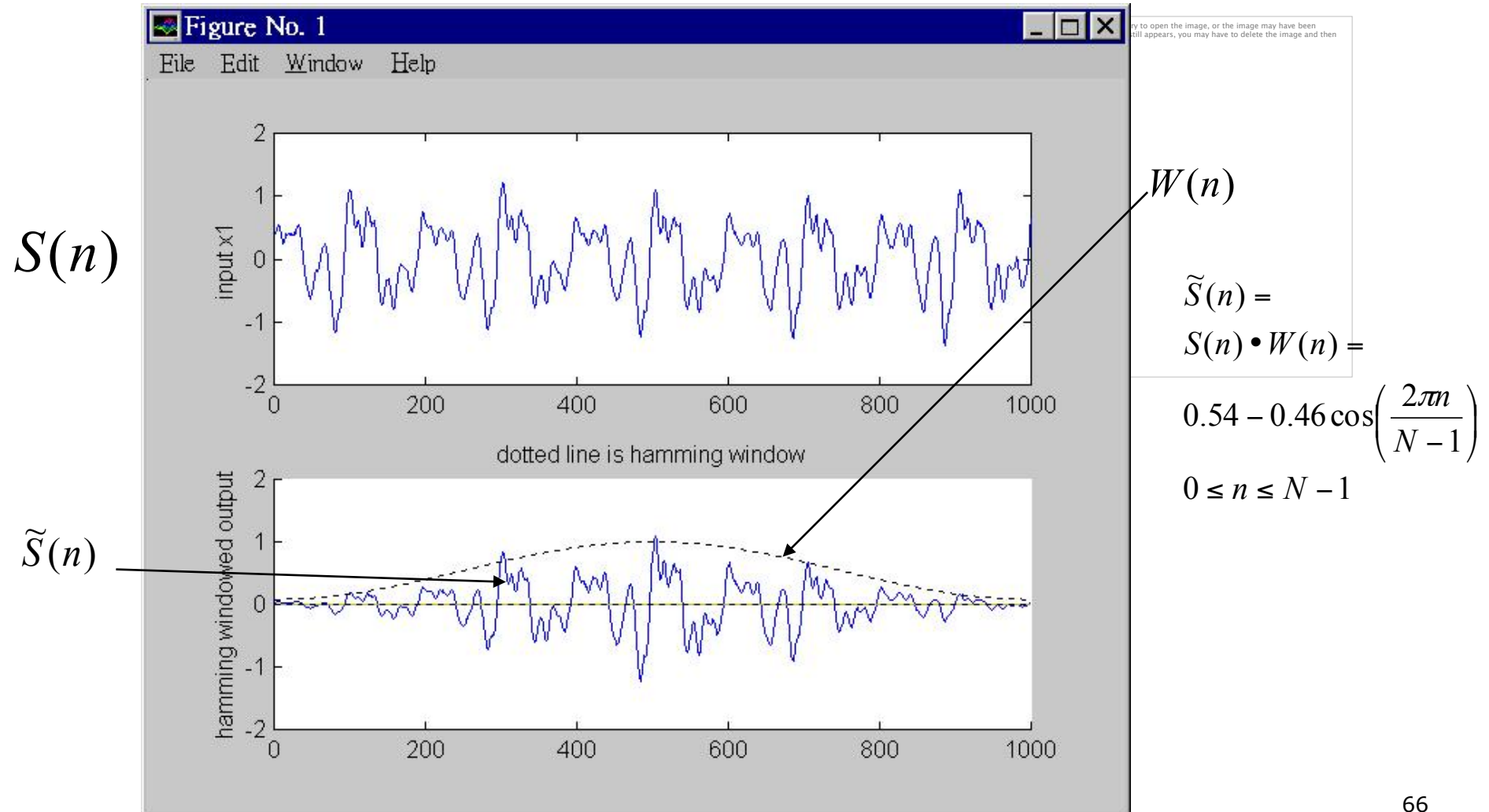
- ❑ To smooth out the discontinuities at the beginning and end.
- ❑ Hamming or Hanning windows can be used.
- ❑ Hamming window

$$\tilde{S}(n) = S(n) \cdot W(n) = 0.54 - 0.46 \cos\left(\frac{2\pi n}{N-1}\right)$$

$$0 \leq n \leq N-1$$

- ❑ Tutorial: write a program segment to find the result of passing a speech frame, stored in an array `int s[1000]`, into the Hamming window.

Effect of Hamming window



Matlab code segment

```
□ x1=wavread('violin3.wav');  
□ for i=1:N  
□  
□     hamming_window(i)=  
□         abs(0.54-0.46*cos(i*(2*pi/N)));  
□  
□     y1(i)=hamming_window(i)*x1(i);  
□ end
```

Cepstrum Vs spectrum

- the spectrum is sensitive to glottal excitation (E). But we are only interested in the filter H
- In frequency domain
 - Speech wave (X) = Excitation (E) . Filter (H)
 - $\text{Log}(X) = \text{Log}(E) + \text{Log}(H)$
- Cepstrum = Fourier transform of log of the signal's power spectrum
- In Cepstrum, the $\text{Log}(E)$ term can easily be isolated and removed.

Auto-correlation analysis

- Auto-correlation of every frame ($l = 1, 2, \dots$) of a windowed signal is calculated.
- If the required output is p -th ordered LPC
- Auto-correlation for the l -th frame is

$$r_l(m) = \sum_{n=0}^{N-1-m} \tilde{S}_l \cdot \tilde{S}_l(n+m)$$

$$m = 0, 1, \dots, p$$

LPC to Cepstral coefficients conversion

- Cepstral coefficient is more accurate in describing the characteristic of speech signal
- Normally cepstral coefficients of order $1 \leq m \leq p$ are enough to describe the speech signal.
- Calculate $c_1, c_2, c_3, \dots, c_p$ from $a_1, a_2, a_3, \dots, a_p$

$$c_0 = r_0$$

$$c_m = a_m + \sum_{k=1}^{m-1} \left[\frac{k}{m} \right] c_k a_{m-k}, \quad 1 \leq m \leq p$$

$$c_m = \sum_{k=m-p}^{m-1} \frac{k}{m} c_k a_{m-k}, \quad m > p (\text{if needed})$$

Distortion measure - difference between two signals

- measure how different two signals is:
- Cepstral distances between a frame (described by cepstral coeffs $(c_1, c_2 \dots c_p)$ and the other frame $(c'_1, c'_2 \dots c'_p)$ is
- Weighted Cepstral distances to give different weighting to different cepstral coefficients(more accurate)

$$d^2 = \sum_{n=1}^p (c_n - c'_n)^2$$

$$\sum_{n=1}^p w(n) \cdot (c_n - c'_n)^2$$

Matching method: Dynamic programming DP

- ❑ Correlation is a simply method for pattern matching BUT:
- ❑ The most difficult problem in speech recognition is time alignment. No two speech sounds are exactly the same even produced by the same person.
- ❑ Align the speech features by an elastic matching method -- DP.

Exercise

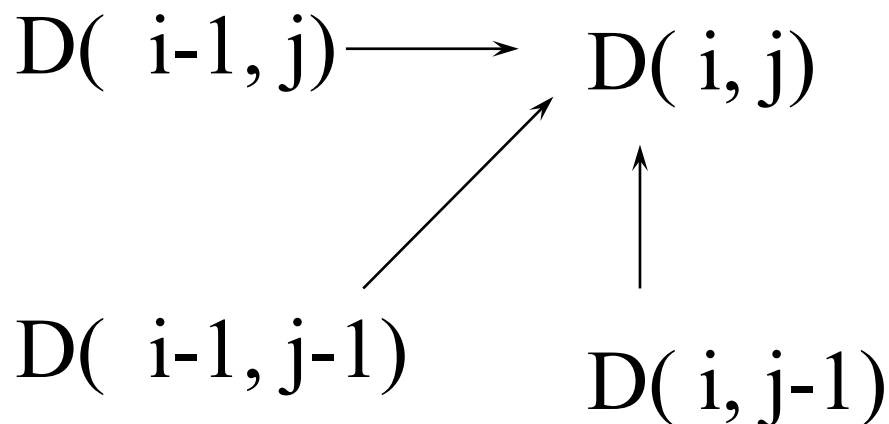
Small Vocabulary (10 words) DP speech recognition system

- Train the system, each word has a reference vector
- For an unknown input, compare with each reference using DP, and the one with the minimum distance is the result.
- Training is easy but recognition takes longer time.

Dynamic programming algo.

- Step 1: calculate the distortion matrix $dist()$
- Step 2: calculate the accumulated matrix
 - by using

$$D(i, j) = dist(i, j) + \min \begin{cases} D(i-1, j-1), \\ D(i-1, j), \\ D(i, j-1) \end{cases}$$



Example in

DP(LEA,
Trends in speech recognition.)

- Step 1 :
- distortion matrix
- Reference

R	9	6	2	2
O	8	1	8	6
O	8	1	8	6
F	2	8	7	7
F	1	7	7	3
	F	O	R	R

- Step 2:

accumulated
score matrix (D)

- Reference

unknown input

R	28	11	<u>7</u>	<u>9</u>
O	19	<u>5</u>	12	18
O	11	<u>4</u>	12	18
F	<u>3</u>	9	15	22
F	<u>1</u>	8	15	18
	F	O	R	R

i-axis

j-axis

Audio signal processing ver 1.9

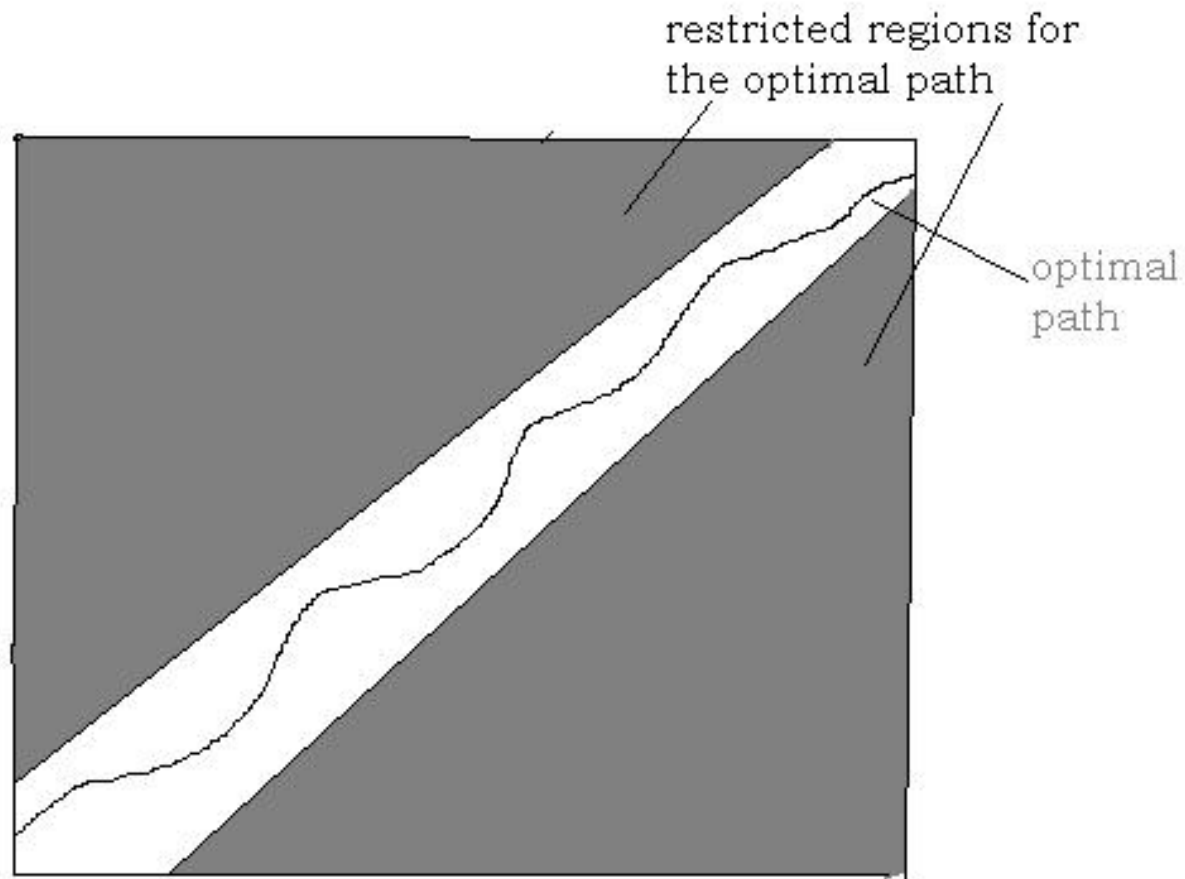
To find the optimal path in the accumulated matrix

- Starting from the top row and right most column, find the lowest cost $D(i,j)_t$: it is found to be the cell at $(i,j)=(3,5)$, $D(3,5)=7$ in the top row.
- From the lowest cost position $p(i,j)_t$, find the next position $(i,j)_{t-1} = \text{argument_min}_{i,j}\{D(i-1,j), D(i-1,j-1), D(i,j-1)\}$.
- E.g. $p(i,j)_{t-1} = \text{argument_min}_{i,j}\{9,11,4\} = (3-0,5-1) = (3,4)$ that contains 4 is selected.
- Repeat above until the path reaches the right most column or the lowest row.
- Note: $\text{argument_min}_{i,j}\{\text{cell1}, \text{cell2}, \text{cell3}\}$ means the argument i,j of the cell with the lowest value is selected.

Optimal path

- It should be from any element in the top row or right most column to any element in the bottom row or left most column.
- The reason is noise may be corrupting elements at the beginning or the end of the input sequence.
- However, in fact, in actual processing the path should be restrained near the 45 degree diagonal (from bottom left to top right), see the attached diagram, the path cannot pass the restricted regions. The user can set this regions manually. That is a way to prohibit unrecognizable matches. See next page.

Optimal path and restricted regions.



Example of an isolated 10-word recognition system

- ❑ A word (1 second) is recorded 5 times to train the system, so there are 5×10 templates.
- ❑ Sampling freq.. = 16KHz, 16-bit, so each sample has 16,000 integers.
- ❑ Each frame is 20ms, overlapping 50%, so there are 100 frames in 1 word=1 second .
- ❑ For 12-ordered LPC, each frame generates 12 LPC floating point numbers,, hence 12 cepstral coefficients C_1, C_2, \dots, C_{12} .

-
- ❑ So there are $5 \times 10 \text{ samples} = 5 \times 10 \times 100$ frames
 - ❑ Each frame is described by a vector of 12-th dimensions (12 cepstral coefficients = 12 floating point numbers)
 - ❑ Put all frames to train a cook-book of size 64. So each frame can be represented by an index ranged from 1 to 64
 - ❑ Use DP to compare an input with each of the templates and obtain the result which has the minimum distortion.

Exercise for DP

- The VQ-LPC codes of the speech sounds of 'YES' and 'NO' and an unknown 'input' are shown. Is the 'input' = 'Yes' or 'NO'? (ans: is 'Yes') distortion

$$distortion(dist) = \sum_{-\infty}^{\infty} (x - x')^2$$

YES'	2	4	6	9	3	4	5	8	1
NO'	7	6	2	4	7	6	10	4	5
Input	3	5	5	8	4	2	3	7	2

YES'	2	4	6	9	3	4	5	8	1
NO'	7	6	2	4	7	6	10	4	5
Input	3	5	5	8	4	2	3	7	2



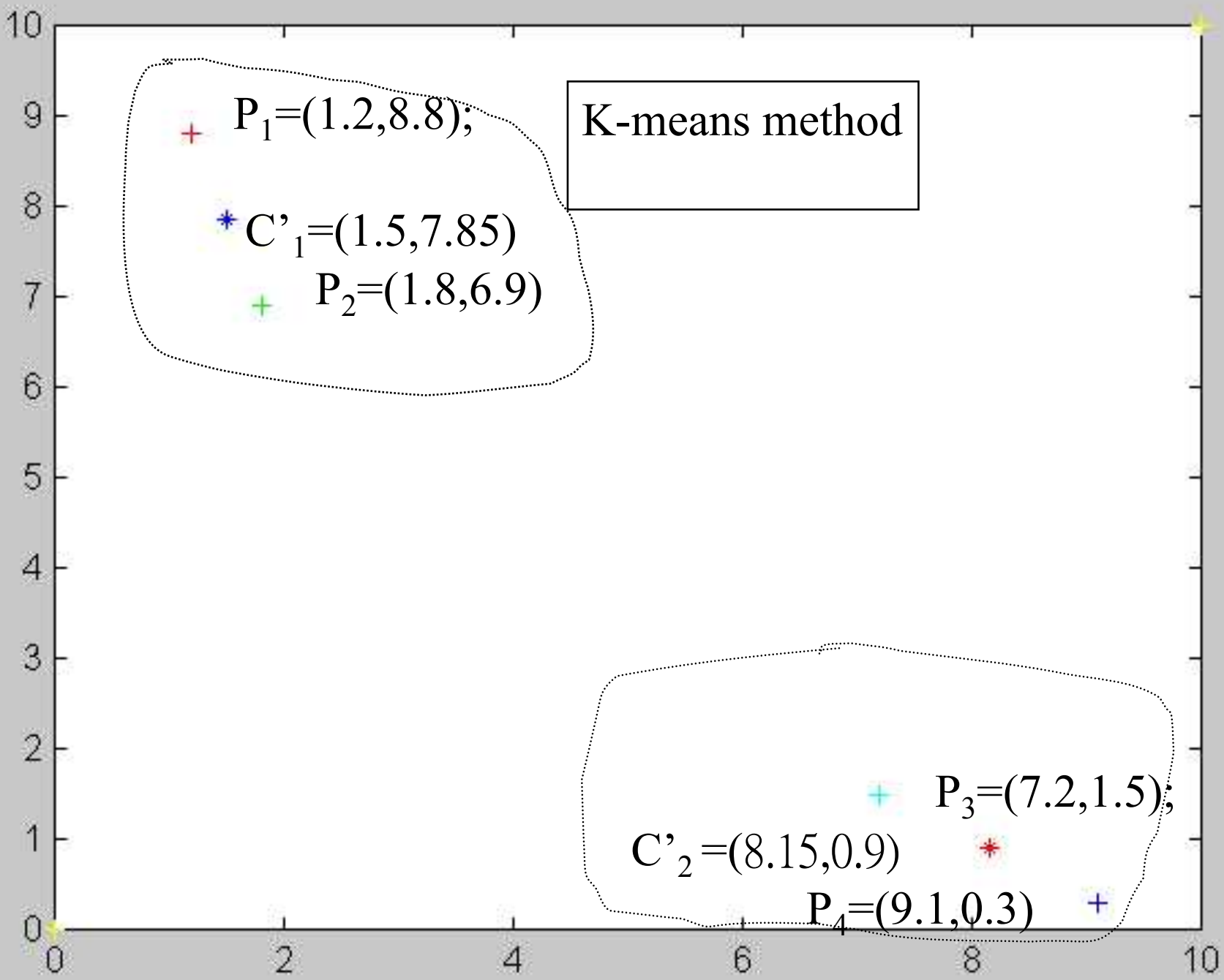
		Distortion matrix for YES									
1	4										
8	25										
5	4										
4	1										
3	0										
9	36										
6	9										
4	1										
2	1										
	3	5	5	8	4	2	3	7	2		
		Accumulation matrix for YES									
1	81										
8	77										
5	52										
4	48										
3	47										
9	47										
6	11										
4	2										
2	1										
	3	5	5	8	4	2	3	7	2		

Conclusion

- ❑ Speech processing is important in communication and AI systems to build more user friendly interfaces.
- ❑ Already successful in clean (not noisy) environment.
- ❑ But it is still a long way before comparable to human performance.

Appendix A.1: *K-means method to find the two centroids*

-
- $P_1=(1.2,8.8); P_2=(1.8,6.9); P_3=(7.2,1.5); P_4=(9.1,0.3)$
 - Arbitrarily choose P_1 and P_4 as the 2 centroids. So $C_1=(1.2,8.8); C_2=(9.1,0.3)$.
 - Nearest neighbor search; find closest centroid
 - $P_1 \rightarrow C_1; P_2 \rightarrow C_1; P_3 \rightarrow C_2; P_4 \rightarrow C_2$
 - Update centroids
 - $C'_1 = \text{Mean}(P_1, P_2) = (1.5, 7.85);$
 $C'_2 = \text{Mean}(P_3, P_4) = (8.15, 0.9).$
 - Nearest neighbor search again. No further changes, so VQ vectors $= (1.5, 7.85)$ and $(8.15, 0.9)$
 - Draw the diagrams to show the steps.

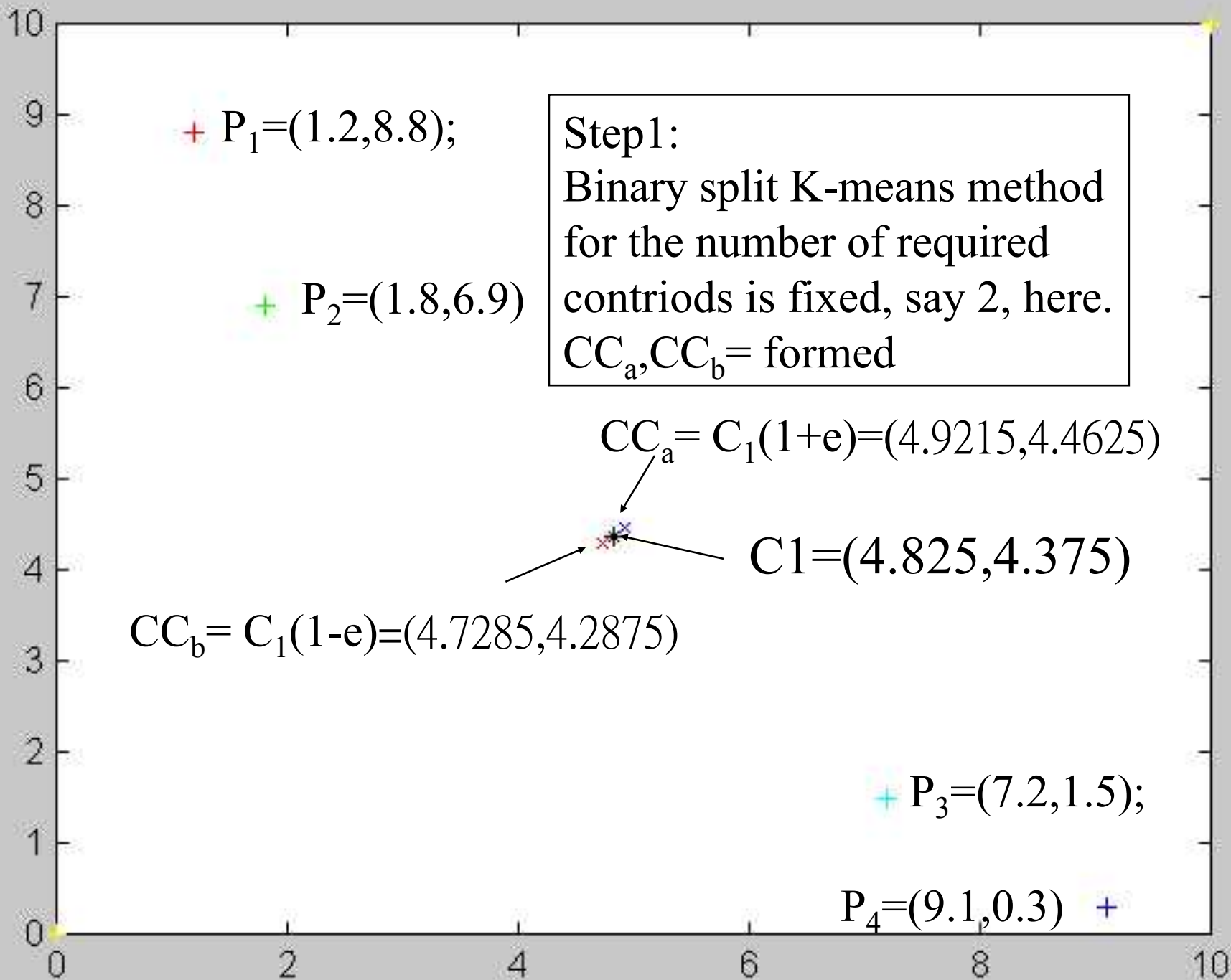


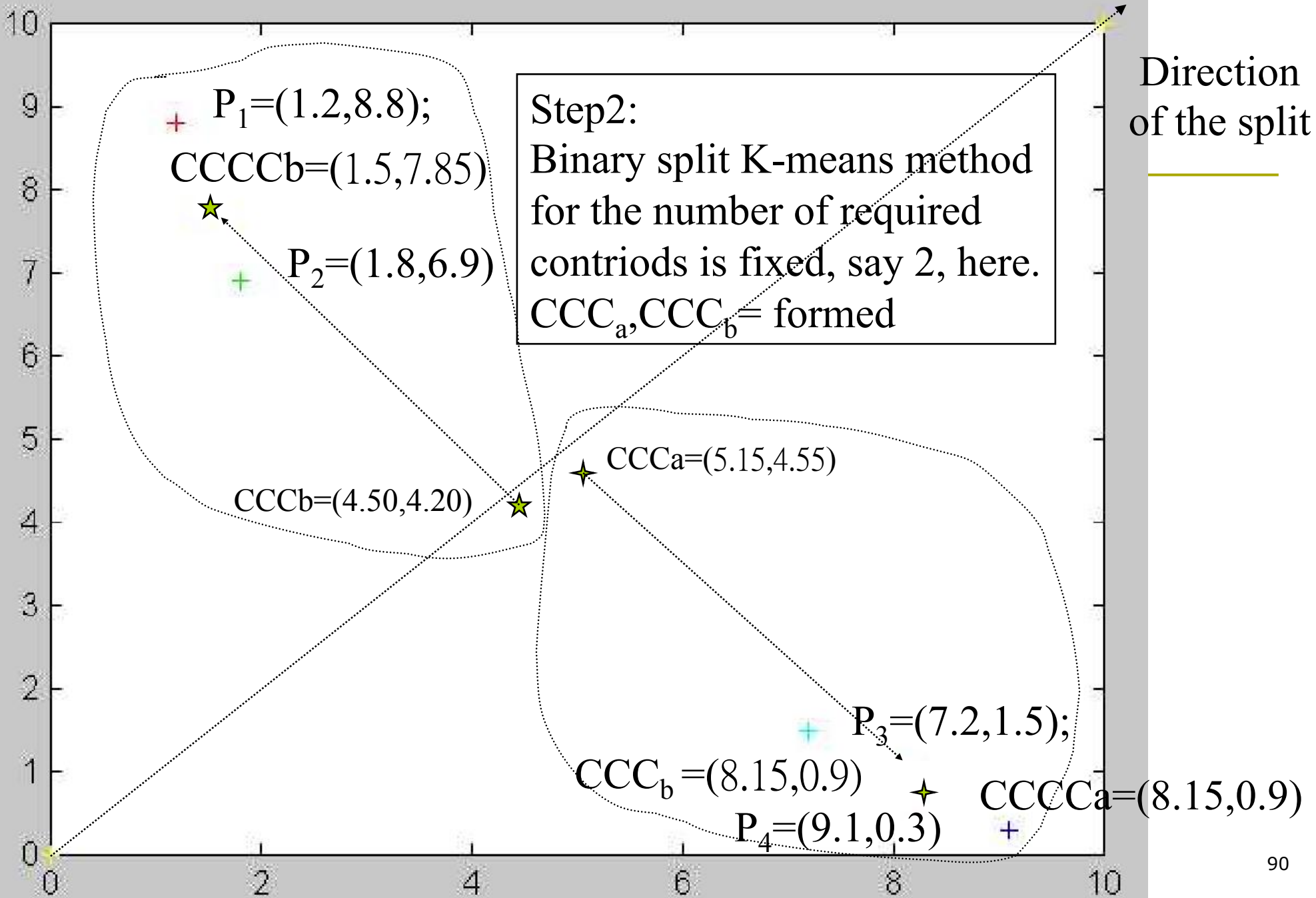
Appendix A.2: Binary split K-means method for the number of required contriods is fixed (see binary_split2_a2.m.txt) (assume you use all available samples in building the centroids at all stages of calculations)

- $P_1=(1.2,8.8);P_2=(1.8,6.9);P_3=(7.2,1.5);P_4=(9.1,0.3)$
- first centroid $C_1=((1.2+1.8+7.2+9.1)/4, 8.8+6.9+1.5+0.3)/4) = (4.825,4.375)$
- Use $e=0.02$ find the two new centroids
- Step1: $CC_a = C_1(1+e)=(4.825 \times 1.02, 4.375 \times 1.02)=(4.9215, 4.4625)$
- $CC_b = C_1(1-e)=(4.825 \times 0.98, 4.375 \times 0.98)=(4.7285, 4.2875)$
 $CC_a=(4.9215, 4.4625)$
- $CC_b=(4.7285, 4.2875)$
- The function $\text{dist}(P_i, CC_x) = \text{Euclidean distance between } P_i \text{ and } CC_x$
-

points	dist to CC_a	$-1 * \text{dist to } CC_b$	=diff	Group to
P1	5.7152	-5.7283	= -0.0131	CC_a
P2	3.9605	-3.9244	= 0.036	CC_b
P3	3.7374	-3.7254	= 0.012	CC_b
P4	5.8980	-5.9169	= -0.019	CC_a

- Nearest neighbor search to form two groups. Find the centroid for each group using K-means method. Then split again and find new 2 centroids. P1,P4 -> CCa group; P2,P3 -> CCb group
- Step2: $CCCa = \text{mean}(P1, P4)$, $CCCb = \text{mean}(P3, P2)$;
- $CCCa = (5.15, 4.55)$
- $CCCb = (4.50, 4.20)$
- Run K-means again based on two centroids CCCa,CCCb for the whole pool -- P1,P2,P3,P4.
- | points | dist to CCCa | -dist to CCCb | =diff2 | Group to |
|--------|--------------|---------------|-----------|----------|
| P1 | 5.8022 | -5.6613 | = 0.1409 | CCCb |
| P2 | 4.0921 | -3.8148 | = 0.2737 | CCCb |
| P3 | 3.6749 | -3.8184 | = -0.1435 | CCCa |
| P4 | 5.8022 | -6.0308 | = -0.2286 | CCCa |
- Regrouping we get the final result
- $CCCCa = (P3+P4)/2 = (8.15, 0.9)$; $CCCCb = (P1+P2)/2 = (1.5, 7.85)$





Appendix A.3. Cepstrum Vs spectrum

- the spectrum is sensitive to glottal excitation (E). But we only interested in the filter H
- In frequency domain
 - Speech wave (X) = Excitation (E) . Filter (H)
 - $\text{Log}(X) = \text{Log}(E) + \text{Log}(H)$
- Cepstrum = Fourier transform of log of the signal's power spectrum
- In Cepstrum, the $\text{Log}(E)$ term can easily be isolated and removed.

Appendix A4: LPC analysis for a frame based on the auto-correlation values $r(0), \dots, r(p)$, and use the Durbin's method (See P.115 [Rabiner 93])

- LPC parameters a_1, a_2, \dots, a_p can be obtained by setting for $i=0$ to $i=p$ to the formulas

$$E^{(0)} = r(0)$$

$$k_i = \frac{\left\{ r(i) - \sum_{j=1}^i a_j^{(i-1)} r(|i-j|) \right\}}{E^{(i-1)}}, 1 \leq i \leq p$$

$$a_i^{(i)} = k_i$$

$$a_j^{(i)} = a_j^{(i-1)} - k_i a_{i-j}^{(i-1)}$$

$$E^{(i)} = (1 - k_i^2) E^{(i-1)}$$

Finally $LPC_coefficients = a_m^p$

Program to Convert LPC coeffs. to Cepstral coeffs.

```
❑ void cepstrum_coeff(float *coeff)
❑ {int i,n,k; float sum,h[ORDER+1];
❑   h[0]=coeff[0],h[1]=coeff[1];
❑   for (n=2;n<=ORDER;n++){ sum=0.0;
❑       for (k=1;k<n;k++)
❑         sum+= (float)k/(float)n*h[k]*coeff[n-k];
❑       h[n]=coeff[n]+sum;}
❑   for (i=1;i<=ORDER;i++)
❑     coeff[i-1]=h[i]*(1+ORDER/2*sin(PI_10*i));}
```

Define Cepstrum: also called the spectrum of a spectrum

- ❑ *"The power cepstrum (of a signal) is the squared magnitude of the Fourier transform (FT) of the logarithm of the squared magnitude of the Fourier transform of a signal"* From Norton, Michael; Karczub, Denis (2003). Fundamentals of Noise and Vibration Analysis for Engineers. Cambridge University Press
- ❑ Algorithm: signal \rightarrow FT \rightarrow abs() \rightarrow square \rightarrow \log_{10} \rightarrow FT \rightarrow abs() \rightarrow square \rightarrow power cepstrum

$$\text{cepstrum}(f(t)) = \left| FT \left\{ \log_{10} \left(\left| FT \{ f(t) \} \right|^2 \right) \right\} \right|^2$$

<http://mi.eng.cam.ac.uk/~ajr/SA95/node33.html>

<http://en.wikipedia.org/wiki/Cepstrum>