An investigation of temporal decomposition of speech parameters for automatic segmentation of speech

Rapport no. 459

An Investigation of Temporal Decomposition of Speech Parameters for Automatic Segmentation of Speech

E.A. Zuk

An Investigation of Temporal Decomposition of Speech Parameters for Automatic Segmentation of Speech

Instituut voor Perceptie Onderzoek (IPO)

Eindhoven, The Netherlands.

By Edward Zuk
Supervisor Ir. A. C. G. de Jager
Coach Ir. L. F. Willems
Chief Professor Prof. Dr. S. G. Nooteboom
Professor Prof. Ir. R. P. Offereins
Date 29-4-1984

Speech Segmentation.

CONTENTS

Speech Segmentation.

Summary.

This report describes an investigation of a procedure for automatic segmention. The original procedure, as published by B.S. Atal, was written to reduce the data rate when transmitting speech. The method describes the speech by a small number of slowly varying and compact phi functions. Atal implies that these functions each correspond to an articulatory gesture made in producing an utterance. The current task is to modify this procedure, not to minimise bit rates, but to find phonemes. The report firstly suggests reasons why this should be possible. It then explains the original procedure, emphasis being placed on its implementation. Next the report describes a number of problems dicovered when trying to use the procedure to locate phonemes. Possible reasons for the problems are discussed together with some solutions, or attempted solutions. Finally alternative approaches to the main problems are suggested.

Speech Segmentation.

Introduction.

At the Institute for Perception Research (IPO) Eindhoven, synthetic speech is made by joining not phonemes but diphones. A diphone is a segment consisting of the stable part of one phoneme and the transition until the stable part of the next phoneme. The method was implemented for Dutch. With about 40 phonemes in the Dutch language, over 1200 phoneme pairs had to be extracted by hand.

B.S. Atal of the Bell Laboratories recently published a paper entitled: " Efficient coding of L.P.C. parameters by temporal decomposition." [1] In it he describes a method to decrease the bit rate for transmitting speech. In this method he tries to associate so-called phi functions with phonemes. This project attempted to use this method not to obtain a reduction in bit rate, but to locate phonemes automatically. With this method the tedious task of hand segmentation of diphones can be automated. This would be a useful tool for creating a speech database for other languages.

Speech Segmentation.

1 Speech.

The overall aim of the analysis procedure is to break up speech by marking the phonetic boundaries. This is by no means an easy task. Even defining what the boundaries should be is difficult. Consider the dutch diphtong "ei" ; is it one phonetic sound or has it a boundary somewhere in the transition of "e" to "i"?

First let us consider how we speak. Speech is produced by a physical system, namely our vocal cords and the vocal tract. We produce different sounds by changing the shape of our vocal tract. Each particular sound or PHONEME has its own target position. Speech is made up of articulatory movements towards the target positions of each of the sounds in an utterance. For example when we say the word "bag" there are three target positions. First we have a bilabial place of aticulation where the lips close to produce the "b" sound. Next the vocal tract opens as we pronounce the vowel. Finally the back of the tongue closes against the soft palate for the velar consonant "g".

Figure 1 shows the waveform of a small segment of speech 150 mSecs long. It is the final "be" from the nonsense word "beboobe" Notice how the waveform quickly gains amplitude. This is where the mouth opens for the plosive "b". From this data we can reconstruct the shape of the vocal tract. See figure 1a. The interval between each diagram of the vocal tract is 100 mSecs. We can see in the figure firstly, the mouth open from the previous consonant, then the mouth closing for the plosive. Next there is a quick opening of the mouth where the pressure built up during the closure is released and finally the tract positioning itself for the next vowel.

SECTOR 1, STARTING FRAME 65, 15 FRAMES, CONTEXT 100

FIG 1 "-be"

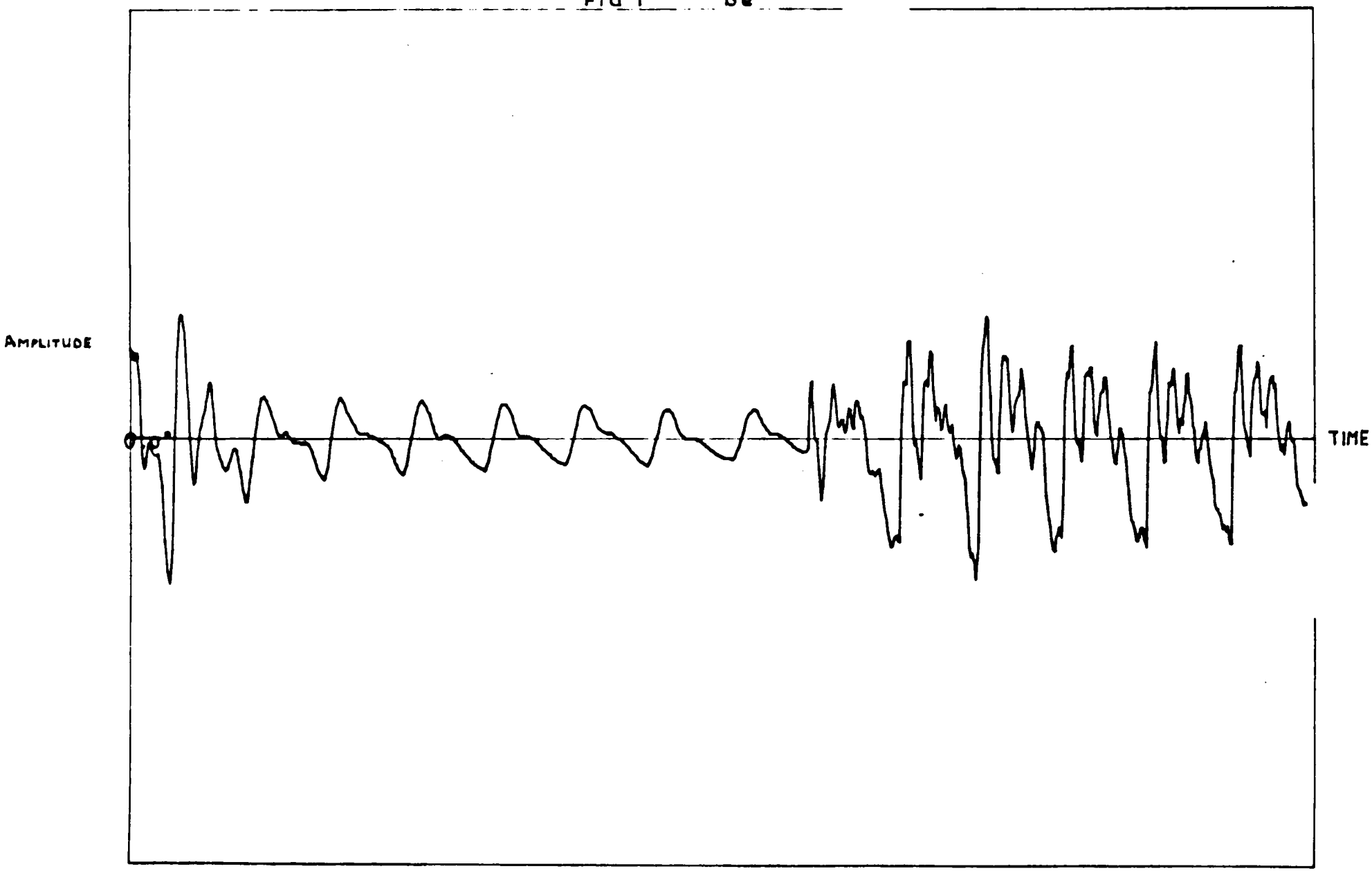
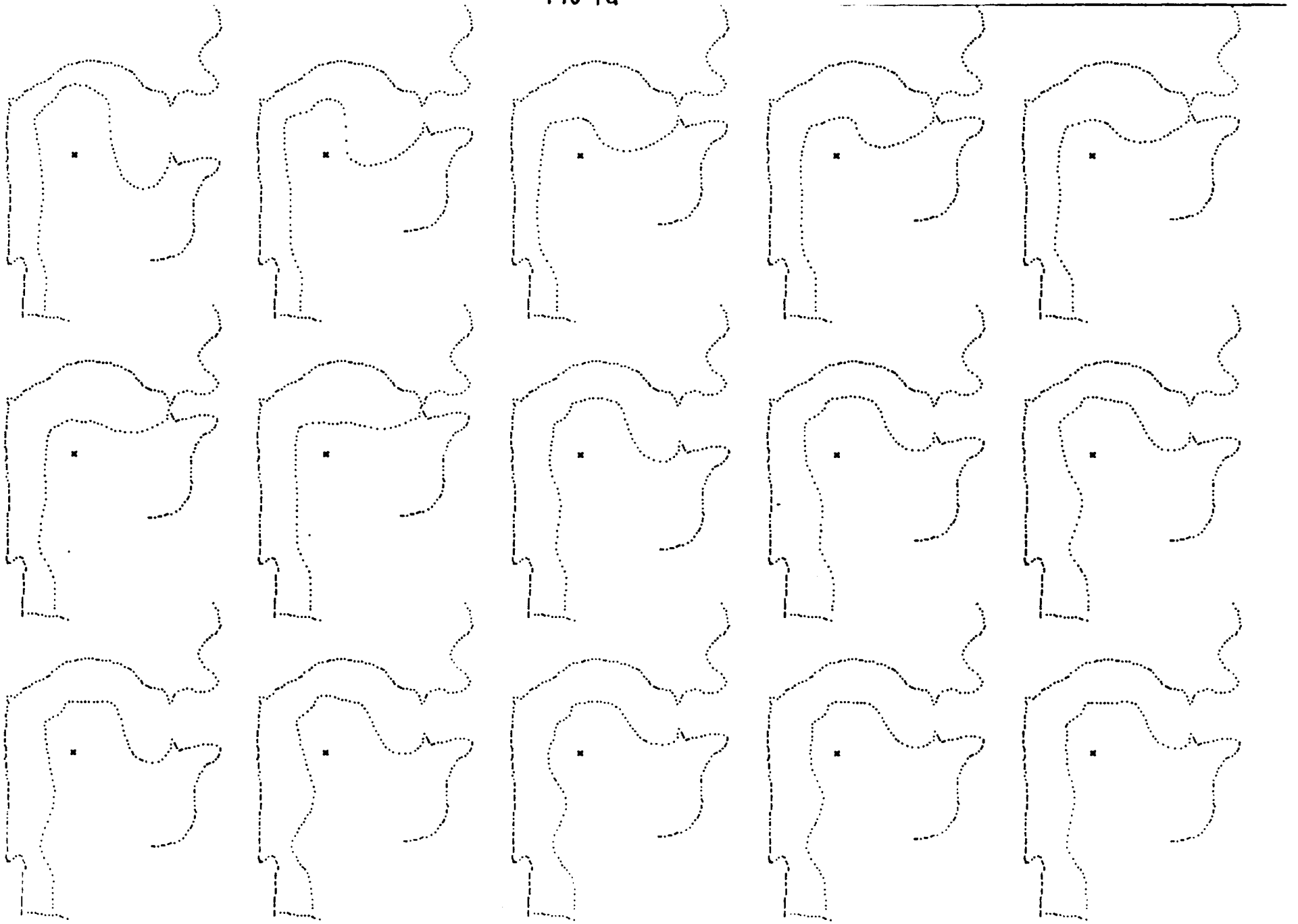


FIG 1a



Speech Segmentation.

Because speech is a physical system it has an inherent inertia. We cannot change the shape of our vocal tract instantaneously. There are transitions. In the interval between two phonemes the shape of the vocal tract corresponds to neither of the two phoneme's target positions. In fact, especially for fast speech, we may never reach the target positions. The tract starts to shape itself for a sound, but before it is stable it starts to approach the position for the next phoneme. A diphone takes into account the characteristic region between two phonemes.

Normal sampling does not take advantage of the transitions in speech. In a transistion region we may require about 10 mSecs per frame for good resynthesis, while in the stable part of a sound, 100 mSecs per frame might be sufficient. Atal's bit rate reduction method tries to take advantage of this. The question is whether the method can also be used for segmentation.

In order to detect phonemes the best approach would be to detect the transitions towards and away from the phonemes. Basically this is what the method tries to achieve.

For a piece of speech data to be analysed, (an utterance), we wish to associate a so-called phi function for every speech event in that utterance. A speech event could be defined as a distinct component of the speech. For example, a silence section, a pure vowel, or a nasal consonant would all be distinct speech events. The associated phi function should be zero until the transition for that speech event begins. At this point the phi function should track the transitions towards and away from that speech event, and then, remain zero. Obviously, if we can get such phi functions, the segmentation of speech would be rather simple. If speech can be described by a number of transition functions, then this should be possible.

2 Choice of Speech Parameters.

Before we can start analysing speech we must have it in a form easy to manipulate. If you take a look at the waveform of the speech, you will very quickly realize that it is hard to handle the raw data. In the original method as proposed by Atal "log area parameters" are used as a parametric description of speech. Throughout my investigations I have been using log area parameters as well. In general the analysis method should work on any type of parameters with maybe small modifications. In spite of this I feel that the log area parameters are best suited for the analysis.

2.1 What are the Log Area Parameters ?

Log area parameters are determined from a LPC analysis of speech [2]. The speech output is modelled by an electrical source and an all pole filter.

The electrical source is either

1) white noise generator for unvoiced speech segments, or

2) an oscillator for voiced speech. The frequency of oscillation is set equal to the pitch.

The filter coefficients are chosen so as to minimize, in the particular time frame, the mean square error between the synthesized and actual speech. This is done by a Linear Predictive Coding method [3]. The filter can be described in a number of ways. The most common method is to describe the impulse response or as a cascade of second order filters. There is a variety of ways of describing a 2-nd order filter. Two obvious ways are, the coefficients of the polynomial of the filter in the frequency domain, or the resonant frequency and bandwidth (Q factor). There are other ways. Each set of parameters is related by some specific transformation. One such set of parameters is the set of log area coefficients [4].

2.1.1 Interpretation of Log Area Parameters.

Log area parameters describe an acoustic model of speech. The speech output can be considered as arising from a sound source in a sound cavity. The source is analogous to the electrical case:

1) a noise source for unvoiced speech, and

2) an oscillating source with a repetition frequency equal to the pitch frequency for voiced speech.

For simplicity the cavity is considered as divided into a number of equally spaced sections. Each section has its own cross sectional area. The log of the ratio of areas of neighbouring sections define the log area parameters. The log area parameters hence can be interpreted as describing a simplified model of our vocal tract. This is shown on fig. 1. The last log area parameter describes the ratio of areas in the throat near the vocal cords, while the first at the lips. Figure 1 shows the mouth changing position as it articulates a word.

Speech Segmentation.

Figure 2 shows for the speech segment of figure 1, the log area parameters changing with time. Each set of log area parameters correspods to a interval of 10 mSecs. The plots are made by smoothing the 10 log area parameters over "distance", the position in the acoustic tube. This set of log area parameters was used to derive the shape of the vocal tract in fig. 1a.

STARTING FRAME = 85, NUMBER OF FRAMES = 15

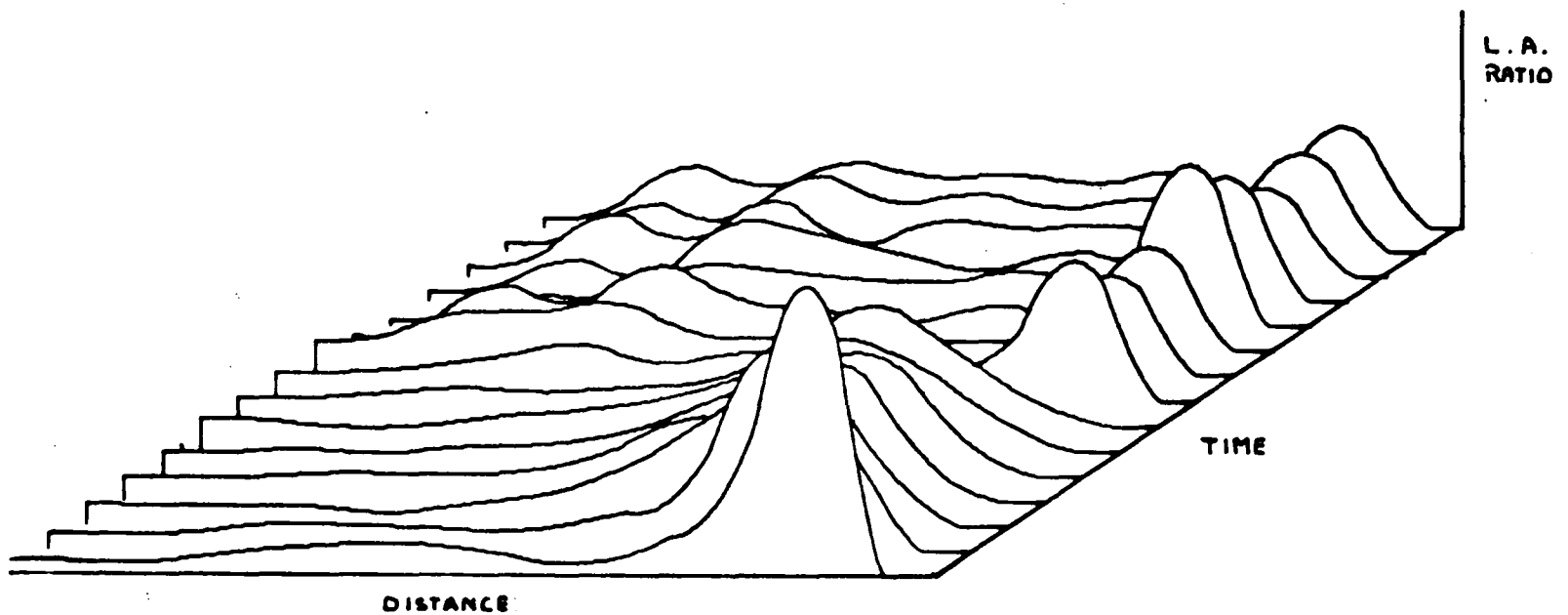


FIG.2

Advantages of the Log Area Parameters.

1) They are easily interpreted as the shape of the vocal tract. Changes in a log area parameter means a movement in the vocal tract. This makes this set of parameters well suited for our analysis.

2) The parameters all have the same meaning. Unlike resonant frequency and bandwidth, they can be treated as similar quantities. This means we don't have to worry about the effects of adding a bandwidth to a frequency!

3) They have similar statistics. The mean and standard deviation are approximately equal for each parameter. This allows them to be treated simply in error calculations and minimizations. If on the other hand the means were not equal, then the parameter with the highest mean would be weighted the most for the error minimization. Unless of course, the method took this into account.

4) The parameters change rather slowly in time. This allows for greater intervals between samples. As a consequence it saves computer time.

5) They are uniformally distributed over their range of values. This minimizes the dynamic range and errors introduced due to truncation.

A disadvantage of the log area parameters, or LPC parameters in general is that, unlike our vocal tract which has antiresonances as well as resonances, the LPC parameters model only resonances.

My investigations are based on the temporal decomposition model for log area parameters as described by B.S. Atal in the Proceedings of the ICASSP 1983 [1]. (This paper is supplied as an appendix to this report). The first problem is understanding the method. While some of the theory for this method is described in that paper, a lot of the practical details are missing. I present here my interpretation of the procedure together with an outline of how this is implemented.

3 The Data

A segment of speech is first recorded on a reel to reel magnetic tape. Next we digitize the speech at a sampling rate of 10kHz. The speech spectrum is assumed to have a highest significant frequency component of 5kHz. The analog to digital converter has a resolution of 12 bits. Each sample is an integer between -2048 and +2047.

The raw quantized data is passed to the program "AALA". Here the speech is firstly converted to LPC parameters. For the analysis a window of duration 25 mSec (250 samples) is used. Shifting the window 100 samples at a time, gives frames of LPC parameters every 10 mSec. The speech is now decribed by a tenth order filter through linear prediction. Other parameters extracted are the voiced or a unvoiced source, gain, and, for a voiced source, its pitch.

$$S(t) = X(t).G(t).O(t)$$

where
- $S(t)$ is the speech
- $X(t)$ is the excitation and can be either the voiced or unvoiced source
- $G(t)$ is the gain
- $O(t)$ is the filter

In our case of tenth order analysis

$$O(t) = \sum_{j=1}^{10} a_j z^{-1} \qquad (a_j \text{ being the LPC parameters})$$

The AALA program then converts the LPC parameters to log area parameters. [4]

4 The Analysis

The aim of the analysis is to descibe the data by phi functions, one phi function for each speech event.

The speech data is represented by a pxN matrix, Y, where

p is the number of L.A. parameters (Log Area)
N is the number of frames of L.A. parameters in the utterance. (The time dimension).

In my case p is set to 10

We can convert the L.A. parameters to other abstract parameters by a linear transformation. This can be represented as :

$$Y = A.\varphi \qquad (1)$$

where

A is a pxm transformation matrix
m is the number of new parameters
φ is the new data matrix consisting of
N frames of
m new parameters.

Note that that there are no time transformations.

The speech, in other words, is desribed by some linear combination of basis functions. In our case we want the basis functions to have special properties, namely to be phi functions. The number of parameter (m) is not known. This depends on the number of speech events present in the utterance.

From 1 we can write

$$\varphi = (A^T A)^{-1} A^T Y \quad \text{iff } (A^T A)^{-1} \text{ exists} \qquad (2)$$

For a speech segment of 1 to 2 seconds duration, it will contain about 20 to 30 speech events. So we can write

$$Y_j = \sum_{i=1}^{30} A_{ij}\, \varphi_i \quad \text{for } j = 1 \text{ to } p \qquad (3)$$

The rank of Y however, can be no greater than 10 (The number of log area parameters). As a result we cannot find 30 phi functions with this one utterance.

Instead we take a smaller time interval of data, say 300 mSec, where we expect no more than 7 or 8 speech events. In this case the equation for the phi functions is valid. The requirement can be restated as follows.

The rank of the Y matrix must be greater than the number of speech events (m).

To satisfy this requirement it is estimated that the length of speech should be about

200 to 400 mSec.

4.1 Rank of the Speech Matrix.

From equation 2 we note that we can write

$$\varphi_k(n) = \sum_{i=1}^{p} W_{ki}\ Y_i(n) \quad 1 \leq k \leq m,\ 1 \leq n \leq N \qquad (4)$$

So the phi functions are simply a linear combination of the actual parameters.

The number of phi functions should be the same as the the rank of Y. This is because if we want the phi functions to reconstruct the data, then the rank for a segment of data can be no greater than the number of phi functions in that segment.

In principle the number of phi functions is fixed by the number of speech events. There will not be 10 phi functions for every speech segment we consider. The problem is to determine the rank of the speech matrix. This is accomplished by the singular value decomposition of Y:

$$Y^T = U\ D\ V^T \qquad (5)$$

Y^T = the speech matrix transformed
U_T = the left hand singular vectors
V^T = the right hand singular vectors
D = an array of eigenvalues

The decomposition transforms the log area parameters, Y to orthogonal ones, U. The amount of information in each of the orthogonal components is specified by the eigenvalue of that dimension.

If we take all the eigenvalues as significant, we maintain all the information. However, taking only a limited number of eigenvalues as significant and the rest as zero, we throw away some of the information. The eigenvalue is a measure of the mean square error we would introduce by assuming that eigenvalue is zero. [5] Take only the m highest eigenvalues. The error introduced is

$$e^2 = \sum_{i=m+1}^{p} \lambda_i \qquad (6)$$

For the log area parameters we can tolerate a 5 percent error. This percentage is estimated from listening tests with resynthesised speech. It is possible to reconstruct the speech after the complete decomposition has taken place with a

5% feature loss. The resulting distortion is small. This percentage can and will be varied later.

We take as significant the eigenvalues that add to 95% of the total sum of the eigenvalues. The other eigenvalues are set to zero. Consequently some of the othogonal vectors in U are multiplied by zero. This constitutes a decrease in dimensionality, hence a decrease in rank. The important information stays.

For example if the 10 eigenvalues are:

10, 8, 7, 6, 3, 1, 0.5, 0.4, 0.2, 0.1

$\sum_{i=1}^{10} \lambda_i = 36.2$

95 % of 36.2 = 34.39

for a 5% error take the first 6 eigenvalues as siginficant

$\sum_{i=7}^{10} \lambda_i = 1.2$

or a 1.2/36.2x100 = 3.3% error.

We consider as relevant data, only the first 6 orthogonal functions in U in equation 5. The problem is now reduced to have a rank of 6.

4.2 Determining the Phi Functions.

The difficulty with determining phi functions is to decribe them mathematically. What we do instead is to describe the properties of them.

If the basis of speech are the articulatory movements, these should be reflected in the L.A. parameters. It is obvious that from equations 4 and 5 we can write

$$\varphi_k(n) = \sum_{i=1}^{m} b_{ki} U_i(n) \qquad (7)$$

What we have to calculate are the "b" coefficients.

The conditions under which we calculate the b coefficients must reflect the property of the phi function. Speech events last for a short time only. Consequently phi functions should have short durations. The aim is to find a linear transformation so as to obtain such functions. We reflect this property by a measure of distance:

$$\theta(1) = [\Sigma_n (n-1)^2 \varphi^2(n) / \Sigma_n \varphi^2(n)]^{1/2} \qquad (8)$$

This function tells us how concentrated the phi function is about the sample 1. The smaller that $\theta(1)$ is, the better. Hence we try to minimize $\theta(1)$ with respect to the unknown coefficients b.

The requirement becomes:

$$\frac{\delta\theta(1)}{\delta b_{ki}} = 0 \quad 1 \leq i \leq m \text{ for each } \varphi_k \qquad (9)$$

This equation reduces to an eigenvalue problem:

$$R\,b = \lambda b \qquad (10)$$

where

R is a mxm matrix and

$$R_{ij} = \sum_n (n-1)^2 U_i(n) U_k(n)$$

b is the eigenvector corresponding to the unknown coefficients

λ is the eigenvalue and it is equal to the minimum value of θ^2

The eigenvalue problem has m solutions and the solution to the minimisation problem is the smallest eigenvalue.

4.3 Locating Phi Functions.

Although we have now defined a criterion for generating phi functions, we do not know

1) How many phi functions exist
2) where they are located.

Consider equation 8, notice that the expression should be minimum if l is somewhere around the middle of the phi function. At this location the derived phi function should be close to the actual phi function. If l is located somewhere between two phi functions then these two will interact producing a resultant. Although the function produces the minimum value of equation 8, it is not any of the desired phi functions. If l is located near the centre of the phi function, any other phi functions interacting should only have small effects, because being away from l, they are heavily weighted.

We define the location of a phi function by its centre of mass

$$L = \sum_n n.\varphi^2(n) / \sum_n \varphi^2(n) \qquad (11)$$

The original procedure for locating phi functions is as follows.

1) Assume there is a phi function located about the first frame, l=1.

2) Select a speech segment 30 mSec long, 15 mSec each side of the point l Problems with extending the data to always be able to have such a segment will be discussed later.

3) Use equations 10 and 7 to calculate the phi functions. Consider only the phi function associated with the smallest eigenvalue of equation 10. This corresponds to the function concentrated most about l.

4) Calculate the phi function's location: L. Define a new function, the location function.

$$\nu(l) = L - l \qquad (12)$$

5) l=l+l, repeat stages 2 to 4 for all the frames.

Figure 3 shows the phi functions as they are calculated for the word "beboobe". Notice how the phi functions are similar in regions, but then move into a transition section before the next "stable" phi region. The "oo" is broken in two sections, first the pure "o" sound followed by the english "w" sound. This occurs because the vowel has a slight diphthong charateristic.

Figure 4a shows the location function $\nu(l)$. This function

1) Is positive when l is less than the location of the phi function.

2) Is negative when l is greater than the location of the phi function.

3) Changes rapidly from negative to positive when l is near the transition regions of the phi functions.

Equation 12 can be used to find where and how many phi functions exist. At every positive to negative zero crossing of $\nu(l)$ we say that a phi function is located, at that l.

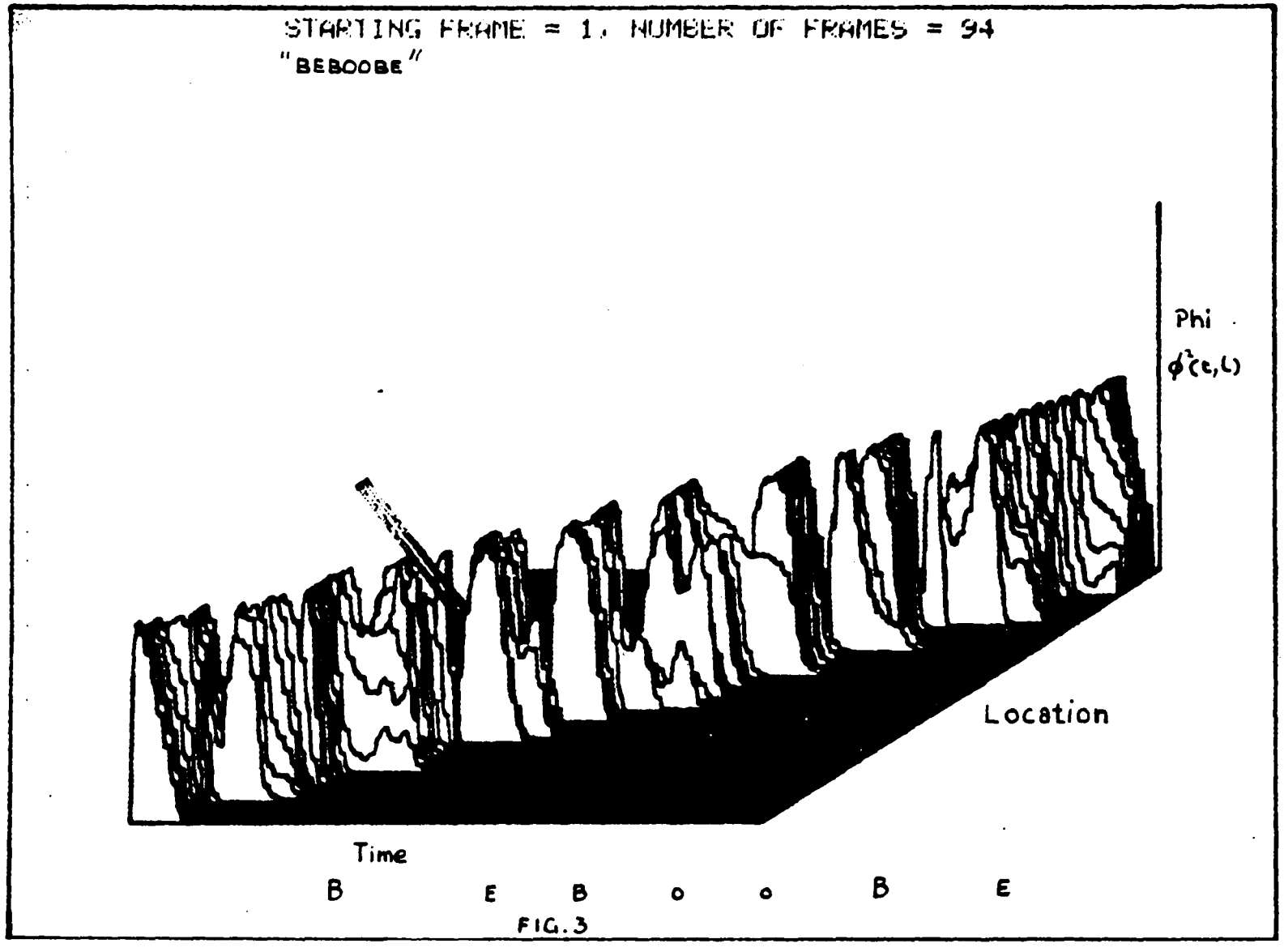

STARTING FRAME = 1, NUMBER OF FRAMES = 94
"BEBOOBE"
Phi
$\phi^2(t,L)$
Location
Time
B E B o o B E

FIG. 3

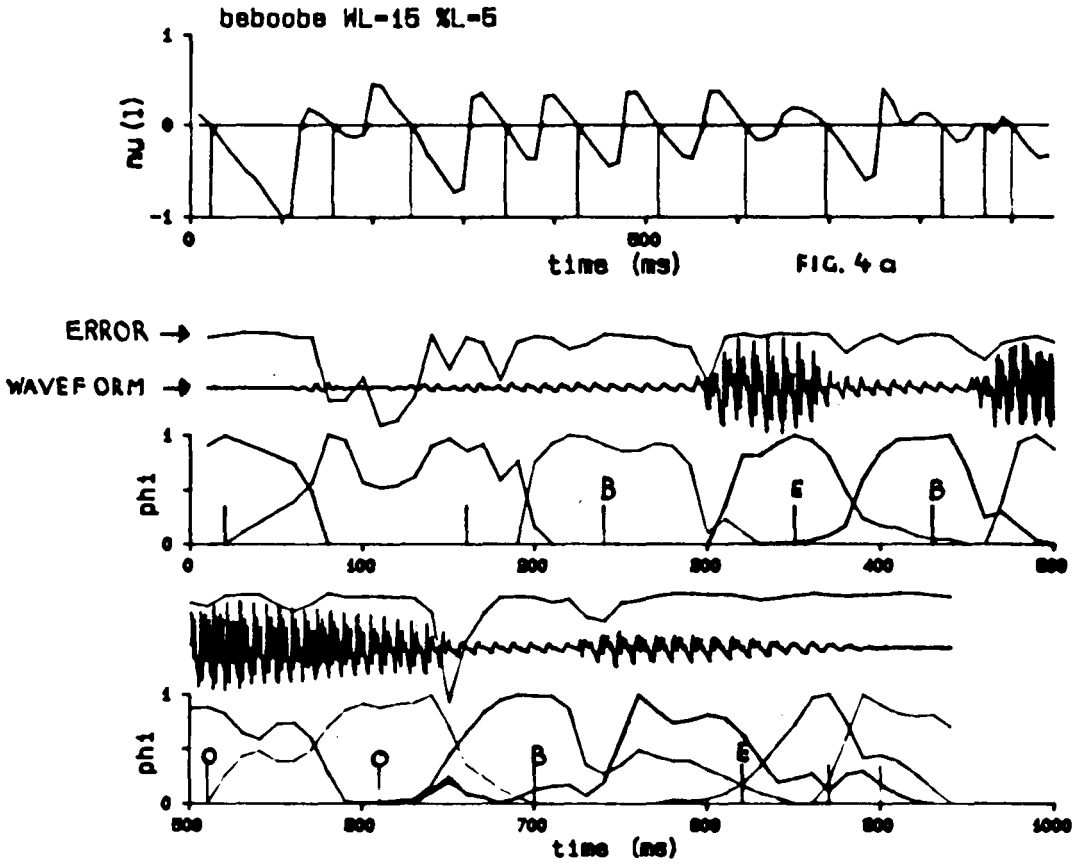

beboobe WL=15 %L=5
1
0
-1
0
800
time (ms)
ERROR
WAVEFORM
phi
1
0
0
100
200
300
400
500
B
E
B
phi
1
0
500
600
700
800
900
1000
O
O
B
E
time (ms)

FIG. 4 a

FIG. 4

4.4 A Better estimate of the Phi Functions.

The procedure as described by Atal recommends two methods for obtaining better estimates of the phi functions.

4.4.1 Recomputation.

The phi functions are recomputed at every location point found. Instead of using a window of 300 mSec, this time the window is adjusted to fit 5 phi functions. That is a window that covers the phi function at the location and two others at both sides. In this segment, the L.A. parameters will be a linear combination of 5 phi functions. To reflect this fact we set the rank of the Y matrix to 5. The first 5 orthogonal functions from the singular value decomposition will be taken as significant.

For the first two and the last two phi functions a smaller window length is used. In these cases only 3 phi functions are included in the data segment. The rank is set to 3.

4.4.2 An iterative Refinement Procedure.

The next stage is to fit the calculated phi functions to the original data. So doing should give a much better estimate of the phi functions. The phi functions are calculated at positions which are not necessarily optimal. Even so the method of minimization produces distortions as it cannot correctly separate them .

The iterative process comprises of two steps.

Step 1) With the calculated phi functions, find the A matrix to minimize the mean square error defined as:

$$E = \sum_{n}[Y_i(n) - \sum_{k=1}^{m} a_{ik}\varphi_k(n)]^2 \qquad (13)$$

$1 \leq i \leq p$, $1 \leq k \leq$ # of phi functions.

We equate the derivative of E w.r.t. A (the unknown coefficients) to zero. This simplifies to a set of simultaneous equations.

$$\sum_{k} a_{ik} \sum_{n} \varphi_k(n)\varphi_r(n) = \sum_{n} Y_i(n)\,\varphi_r(n) \qquad (14)$$

$1 \leq r \leq$ # of phi functions, $1 \leq i \leq p$

where p is the number of log area parameters.

Step 2) With the calculated A matrix, recalculate the phi functions to minimize the mean square error again. This time we take the derivative w.r.t. the phi function and equate that to zero. The resulting expression for the phi function becomes:

$$\varphi_r(n) = \left[\sum_{i=1}^{p} Y_i(n) a_{ir} - \sum_{k \neq r} \varphi_k(n) \sum_{i=1}^{p} a_{ir} a_{ik} \right] / \sum_{i=1}^{p} a_{ir}^2 \qquad (15)$$

This process is repeated until the error drop is smaller than some threshold. Figure 4 shows the results of one run on the utterance "beboobe". Once again we can see that the "oo" is in two overlapping sections. The segment for the closed mouth section of the first "b" is unfortunately also in two sections. Most probably because it is such a long section. Compare it to the second "b".

5 Investigating the Procedure.

Originally it was thought that the method could be put directly to segmenting speech. It did not take much investigation to show that this was far from the truth. As the procedure stands, it has a lot of shortcomings. The main criticism is the procedure's sensitivity to various parameters. It was found that for a good combination of parameters the results were also good. The problem lies in the fact that every speech segment needed a different set of parameters. This is not very good for automatic segmentation.

The first stage of the investigation was to establish the effects of a few simple parameters on the process. These parameters were

1) The original window size for analysis

2) The amount of error tolerated for the singular value decomposition.

3) The number of log area parameters.

The third parameter was found to be fairly unimportant. Similar results were obtained for 8 or more parameters. This suggests that most of the articulatory information is in the first 8 or so L.A. parameters.

5.1 Varying the Parameters.

The procedure was first run on the speech segment "mama". The aim was to see how the phi functions change with some parameter variations. It was found that the effects were dramatic. A simple change in one parameters could change not only the location of phi functions but also the number.

The two parameters under investigation are:

1) The length of the speech used to analyse phi functions. This is called the window length. It is the length in frames of a rectangular window which multiplies our utterance to give the short speech segment for analysis.

2) The amount of information allowed to be cut by reducing the dimensionality of the speech. This is the % feature loss. It says how much of the total information is lost.

The best way to judge the effects of these parameters is by the ν function. This is plotted in figure 5 for the values:

Window length = 100, 200, 300, 400 mSec
% feature loss = 2, 5, 10 %

As can be seen the locations and number of phi functions are different for every set of parameters. (Locations are indicated by vertical lines on the graphs.)

MAMA

% FEATURE LOSS

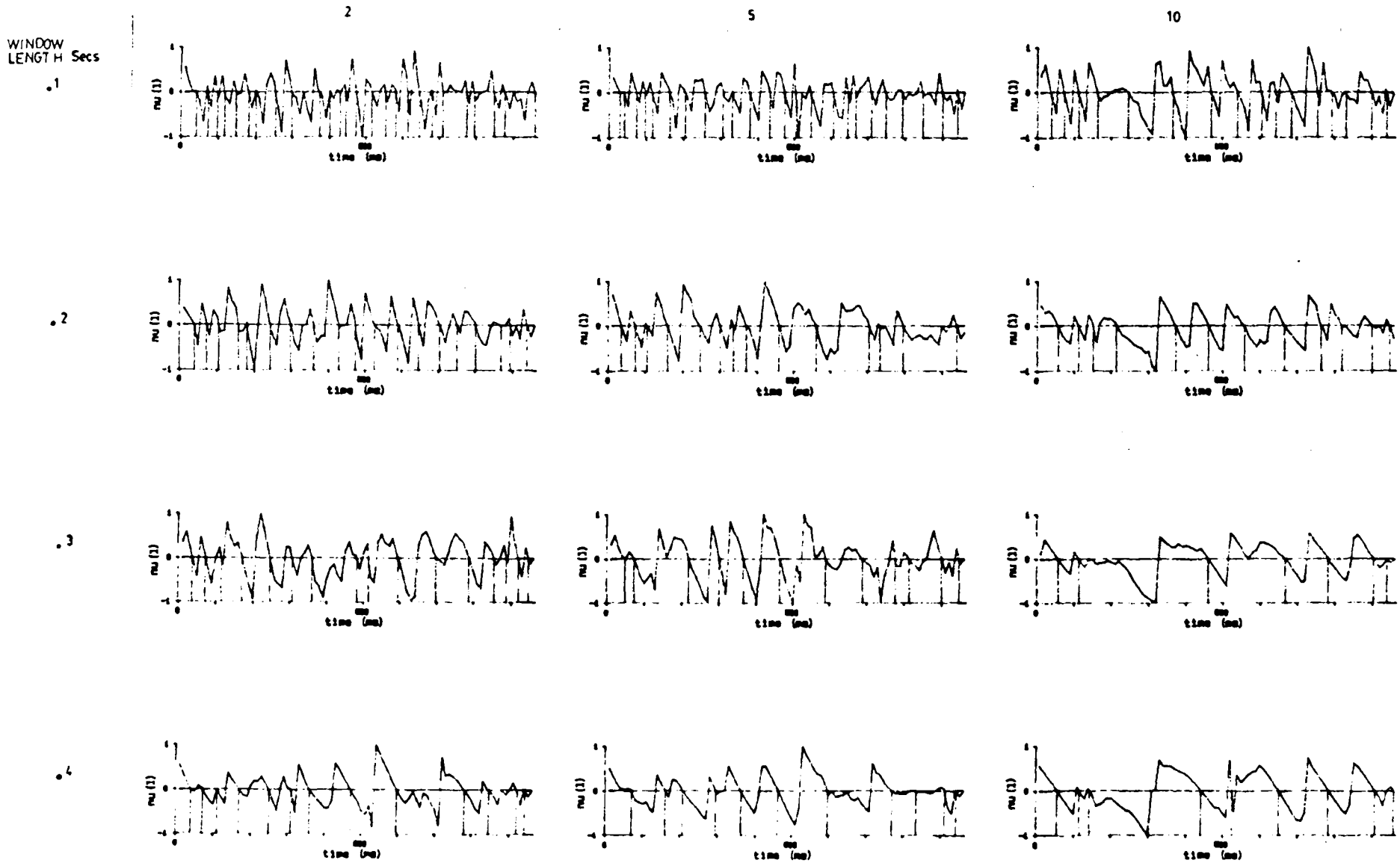


FIG. 5

In terms of segmentation this is disastrous. It basically means that as the program stands, the parameters decide on the segmentation, not necessarily the data. One can hardly hope to obtain any relation between phi functions and actual articulation this way.

At this point a closer investigation is required.

The first step is to study the procedure knowing what the outcome should be.

5.2 Investigations with Resynthesised Data.

To see how the program behaves, data constructed from phi functions was used. The output of an analysis run was converted back to log area parameters. From the phi functions and the A matrix, the L.A. parameters are obtained from equation 1.

These resynthesised L.A. parameters were analysed as normal data. The same parameter analysis as before leads to fig 6.

Now the location function is more consistent when considering parameter changes. For window lengths of 200 to 300 mSecs and % feature loss of 2 to 5%, the location functions are almost identical to each other.

From this some conclusions can be drawn.

1) If the % feature loss is high then some phi functions are not found.

2) The window length must be able to fit the biggest phi finction.

3) The window length must remain small enough to maintain the rank greater than the number of phi functions in the window.

A plot of the results for

window length = 300 mSecs
%feature loss = 5%

is shown in figure 7. We see that the phi functions are almost identical to the originals. The phi functions for the resynthesised data tend to be more spread out. We can explain this by the feature loss. The phi functions are recalculated for the data with reduced rank, and so some data loss. In actual data the feature loss would involve mostly noise. There is no such noise in the resynthesised data! The iterative refinement uses the original data. By distorting the functions the refinement procedure compensates for the feature loss. As would be predicted, for 0 %feature loss there is no distortion.

MAMA RESYNTHESISED

% FEATURE LOSS

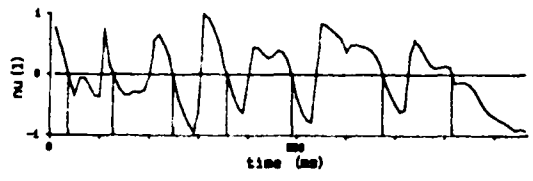
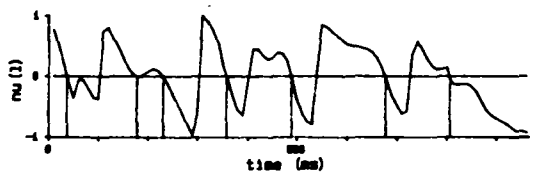
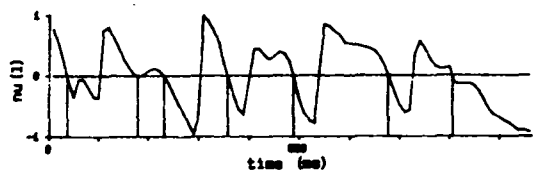
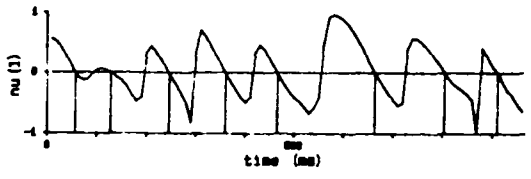
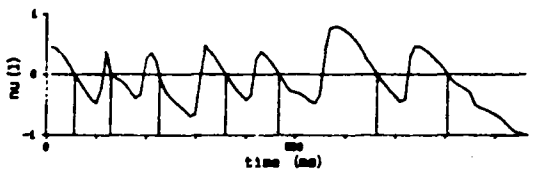
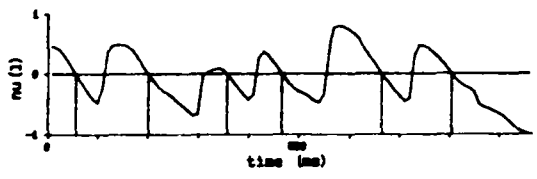
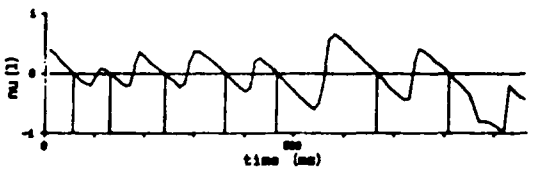
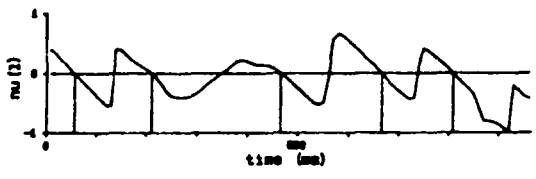
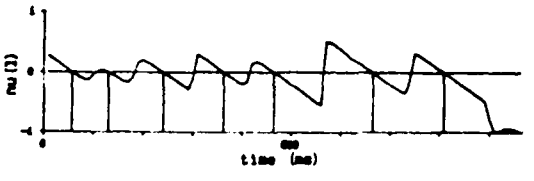
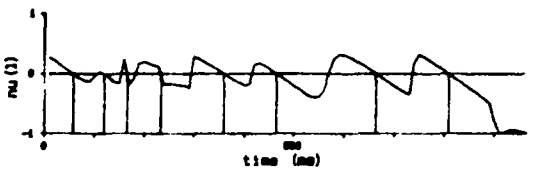
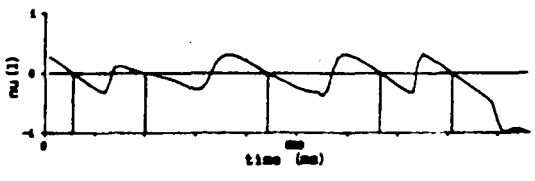
WINDOW LENGTH Secs	2	5	10
.1			
.2			
.3			
.4			

FIG. 6

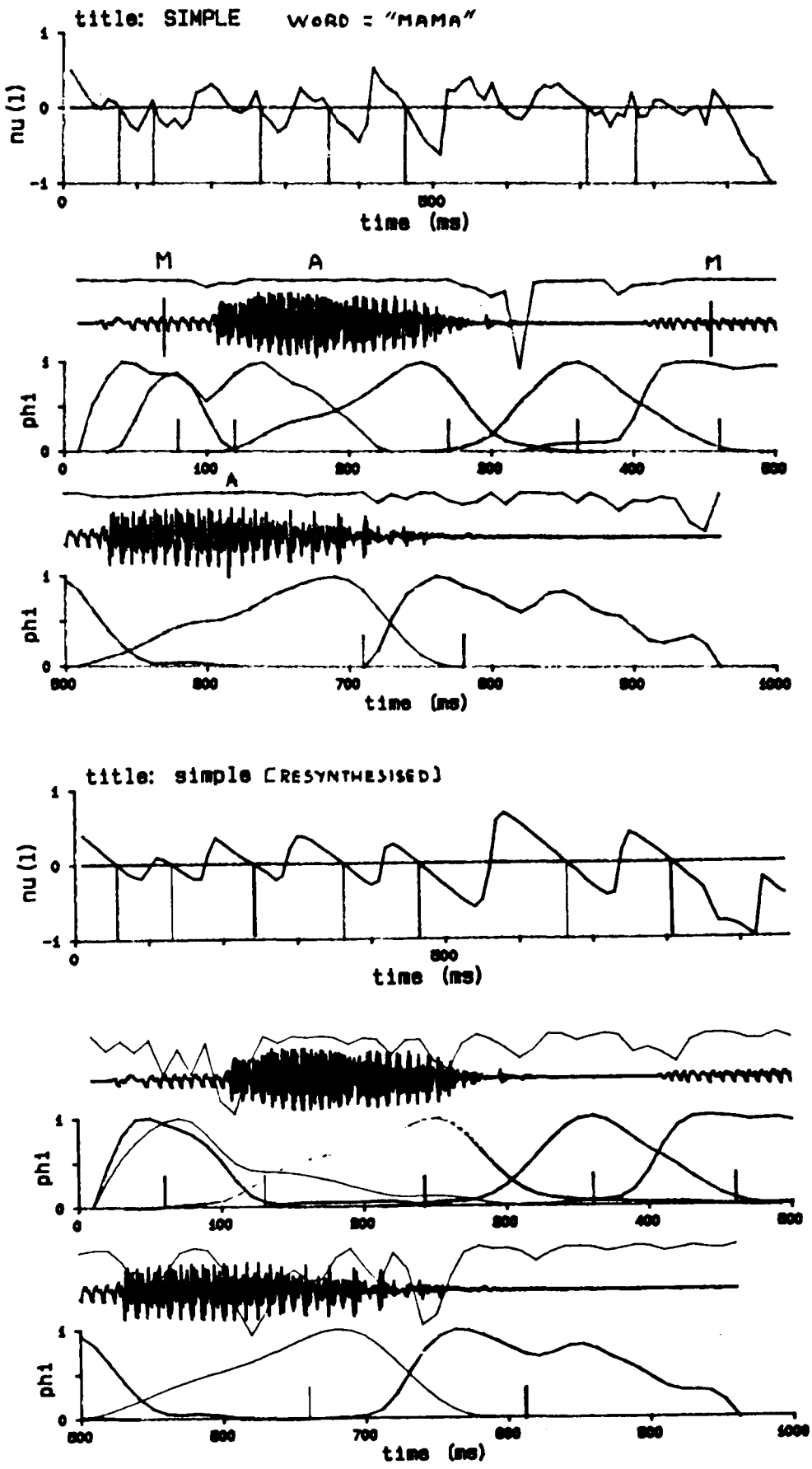

title: SIMPLE WORD = "MAMA"
nu(1)
time (ms)
M
A
M
phi
A
phi
time (ms)
title: simple [RESYNTHESISED]
nu(1)
time (ms)
phi
phi
time (ms)

FIG. 7

5.2.1 Procedure Modification with Resynthesised Data.

While still using the resynthesised data a few changes to the procedure were tried.

5.2.2 Utilizing the Same Windowed Data for a Few Values of l

The singular value decomposition is a necessary but expensive process. I tested whether it was possible to have the window shifted by quarter of its length, rather than by steps of one frame. One set of windowed data, and hence one singular value decomposition is used for a few values of l. The location of l is always near the centre of the window.

Although this proved to save computation time, the method decrease the range in which the parameters can vary. The plot of the location function for various parameters is on figure 8. For most parameter settings extra phi functions are found. The location of some of the phi functions has also changed. I do not believe the saving in time is justified by the deterioration in results.

MAMA RESYNTHESISED : ONE WINDOW FEW ℓ

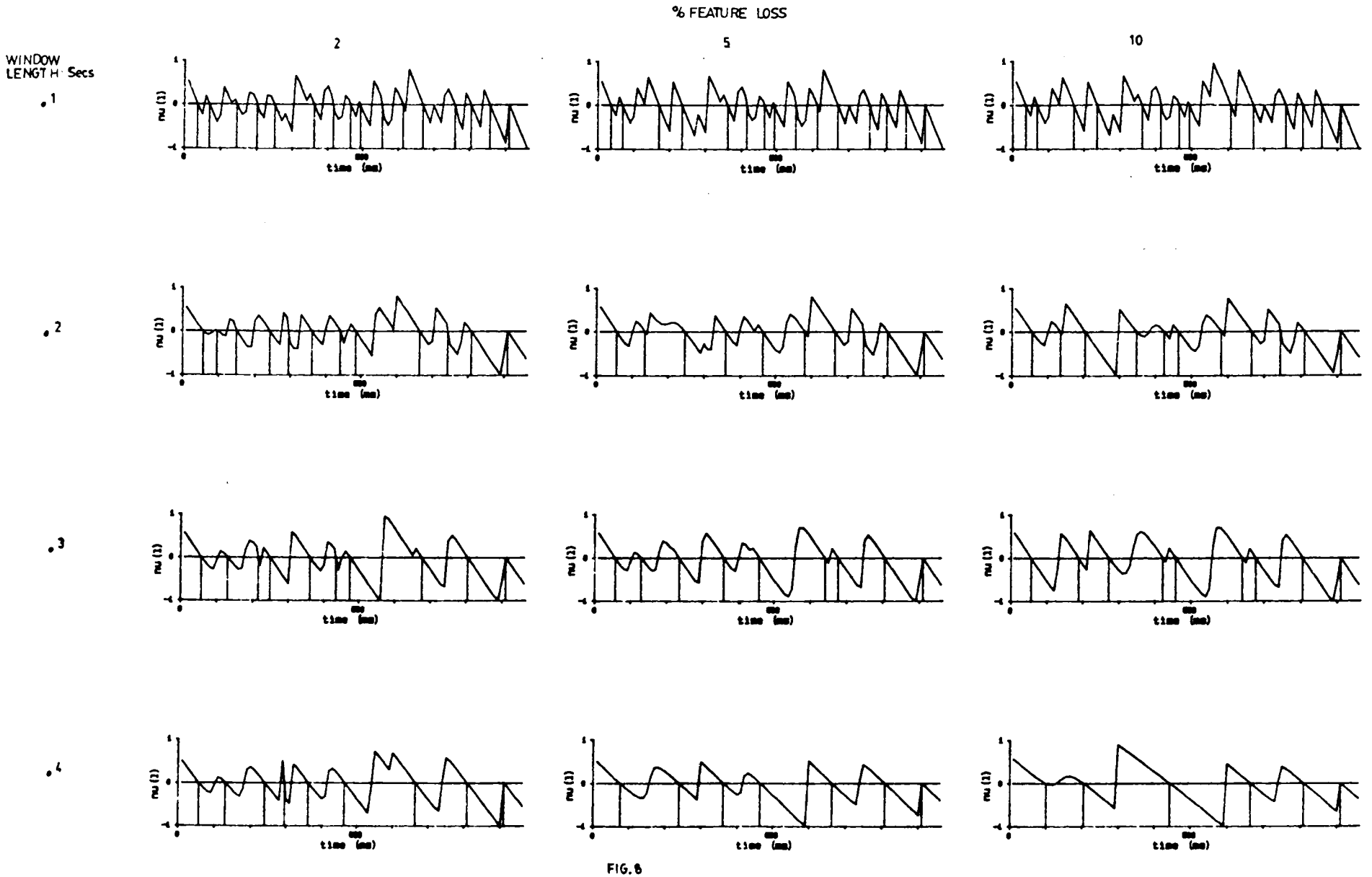


FIG. 8

5.2.3 Modifying the Location Function.

We wish to find the position in the phi function for which $\theta(1)$ is minimum.

Equation 12 can be rewritten as

$$\nu(1) = \sum_n (n-1)\varphi^2(n) / \sum_n \varphi^2(n)$$

A better function could be

$$\nu'(1) = \sum_n \mathrm{sgn}(n-1)(n-1)^2\varphi^2(n) / \sum_n \varphi^2(n) \qquad (16)$$

where

$$\mathrm{sgn}(x) = \begin{cases} -1 & \text{for } x < 0 \\ 0 & \text{for } x = 0 \\ 1 & \text{for } x > 0 \end{cases}$$

Intuitively the $(n-1)^2$ factor should be an improvement because this finds the point about which the function has symmetrical weighting. The results for this new location function with different parameters is shown on figure 9. This shows no improvement over the old location function. The number and location of phi functions is the same as before, but the dynamic range of the function is greater. This modification is not recommended.

MAMA RESYNTHESISED MODIFIED LOCATION FUNCTION

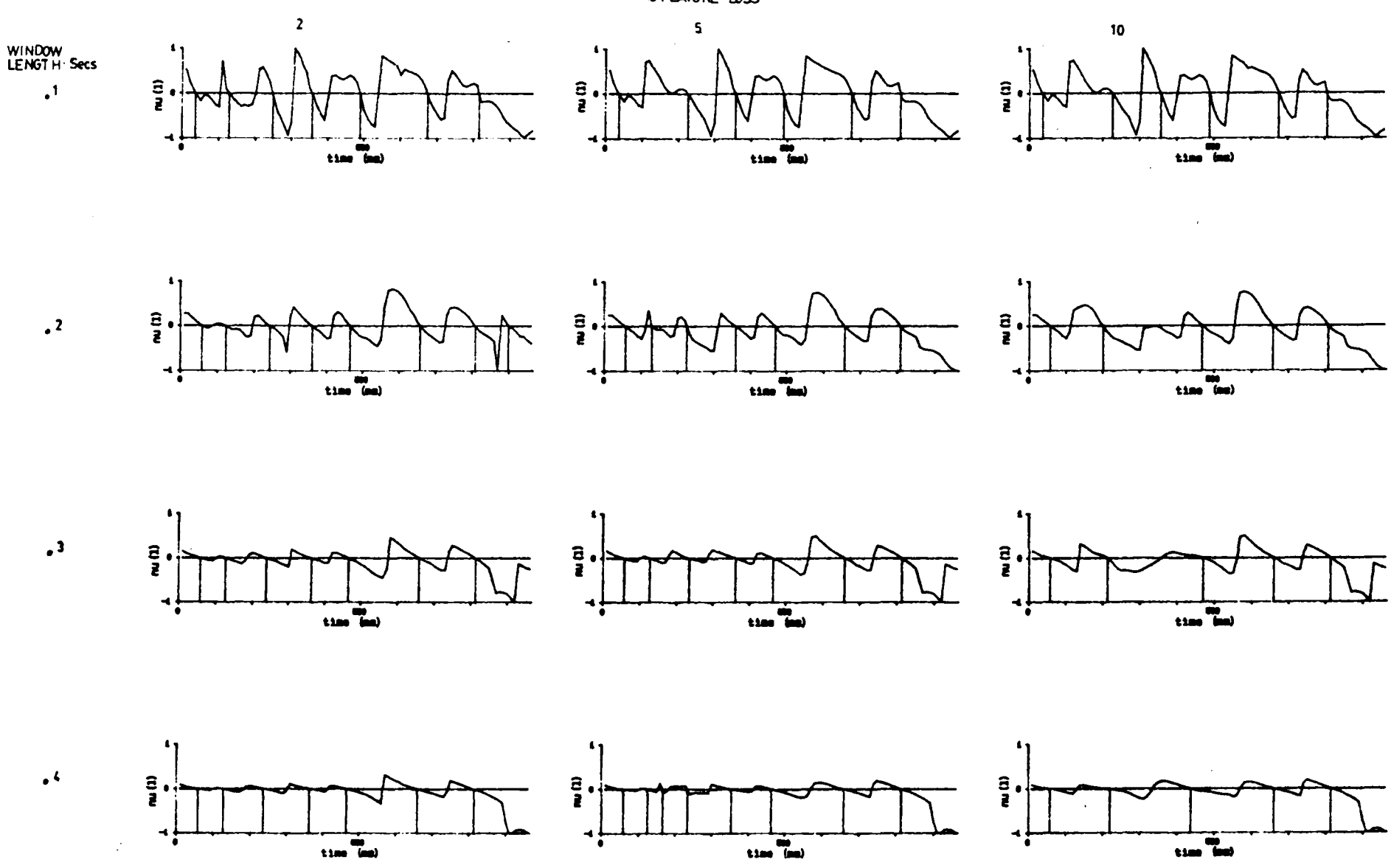


FIG. 9

5.3 Modifying the Data.

From the previous section it appears that the data may not have the information required for segmentation. Alternatively the information may be hidden.

In order to determine this a number of parameters of the data were changed. These included

1) The window length for a frame of L.P.C. analysis

2) The number of log area parameters.

I found that these had minimal effects. If the window length was too long phonemes disappeared. They were averaged out over the surroundings. Similarly if the number of L.A. parameters decreased then too much information was lost and phonemes could not be located. Too short window lengths produced spurious phi functions, as did a large number of log area parameters.

I did notice that the most unstable regions in the data were the instances when the energy was the lowest. For example the silent part of a plosive. In these cases the data is pure noise, not related at all to the rest of the data. The L.A. parameters, descibing only the shape of the acoustic tube, have no amplitude information. Noise data leads to totally random parameters which are not easily distinguished from valid ones.

During the iterative refinement the error is minimized over the whole interval. Even where the data is unreliable. The other problem is that there are numerous phi functions produced to describe the "noise" data. Generally these phi functions can be considered as "out of character" as the data is not reliable. The phi functions interact in the production and location of other phi functions. "Noise" phi functions can therefore distort other phi functions.

5.3.1 Modified Parameters.

The first attempt to solve the problems with low energy was to multiply the log area parameters with the log of the energy in the frame. The idea is that the low energy regions should contribute only one phi function, the $\theta(1)$ function being smallest for this section. Secondly when considering equations 13 and 14 in the iterative refinement, the low energy sections will have a small effect. The L.A. parameters are small in the noise sections so they should not contribute greatly to the error. Incorparating the energy to the data also adds information that should help with segmentation.

The results did not however support these considerations. There was no visible improvement. The phi functions obtained where just distorted versions of phi functions in a normal analysis

5.3.2 Interpolating the Data

An alternative solution is to interpolate low energy L.A. parameter sections between two high energy points, which should provide reliable data. The threshold between reliable and unreliable data was chosen as the point at which the mean energy in a frame equals 500. The waveform is normalised to have values

between -2047 and 2048, while the energy is calculated as the sum of the squares of the sample values.

Any frame with with an energy level lower than 500 is thrown away. Instead the values for these points are linearly interpolated. The interpolation uses the two nearest calculated points either side of the low energy points.

Results show that this stabilizes the data somewhat. The problem now is that these L.A. frames do not match the rest of the data. For the low energy regions the phi functions have characteristic straight lines. This contrasts to the other phi functions.

5.3.3 Smoothing the Log Area Parameters

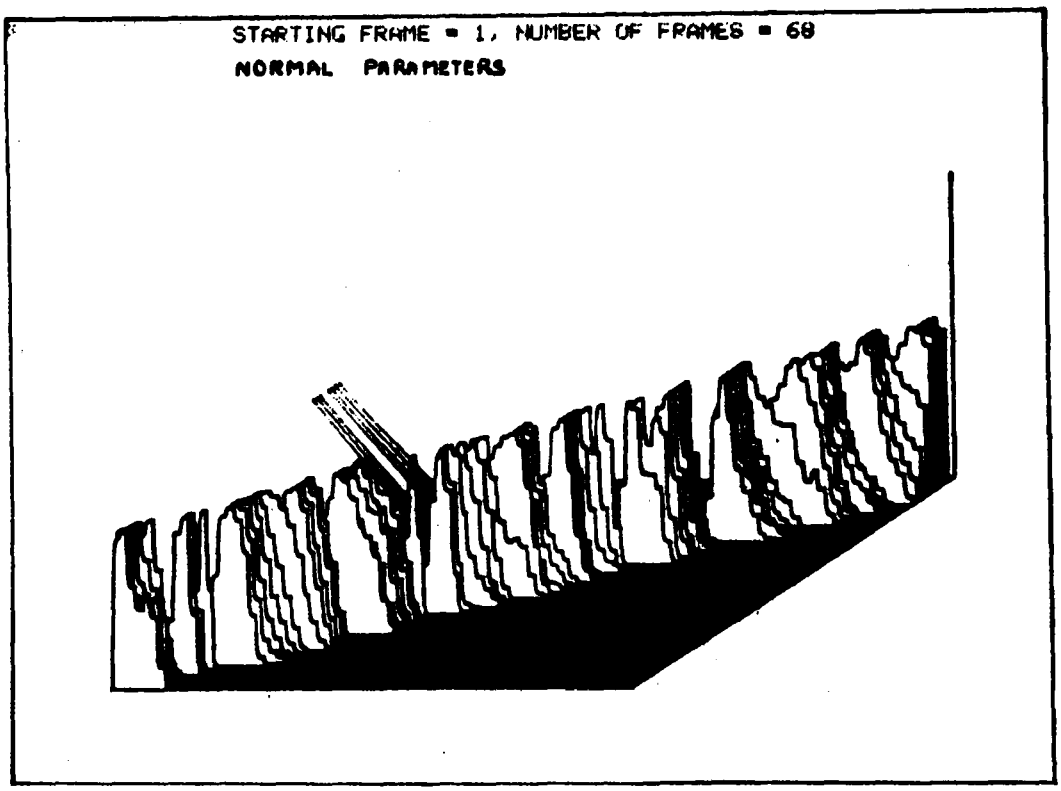
Another improvement for the data is to pass it through a low pass filter. A three point moving average filter was used. The filter is non-causal with a symmetric impulse response about 0. This introduces no phase delay and ensures the L.A. parameters are synchronous with the original data.

The impulse response is given by:

$$h(n) = \alpha\delta(n+1)+(1-2\alpha)\delta(n)+\alpha\delta(n-1) \qquad (17)$$

The optimum value of α is about 0.15.

A comparision of the results with normal and enhanced L.A. paramters is shown in figure 10. Clearly the phi functions belonging to phonemes are much easier selected from the enhanced parameters.


STARTING FRAME = 1, NUMBER OF FRAMES = 68
NORMAL PARAMETERS

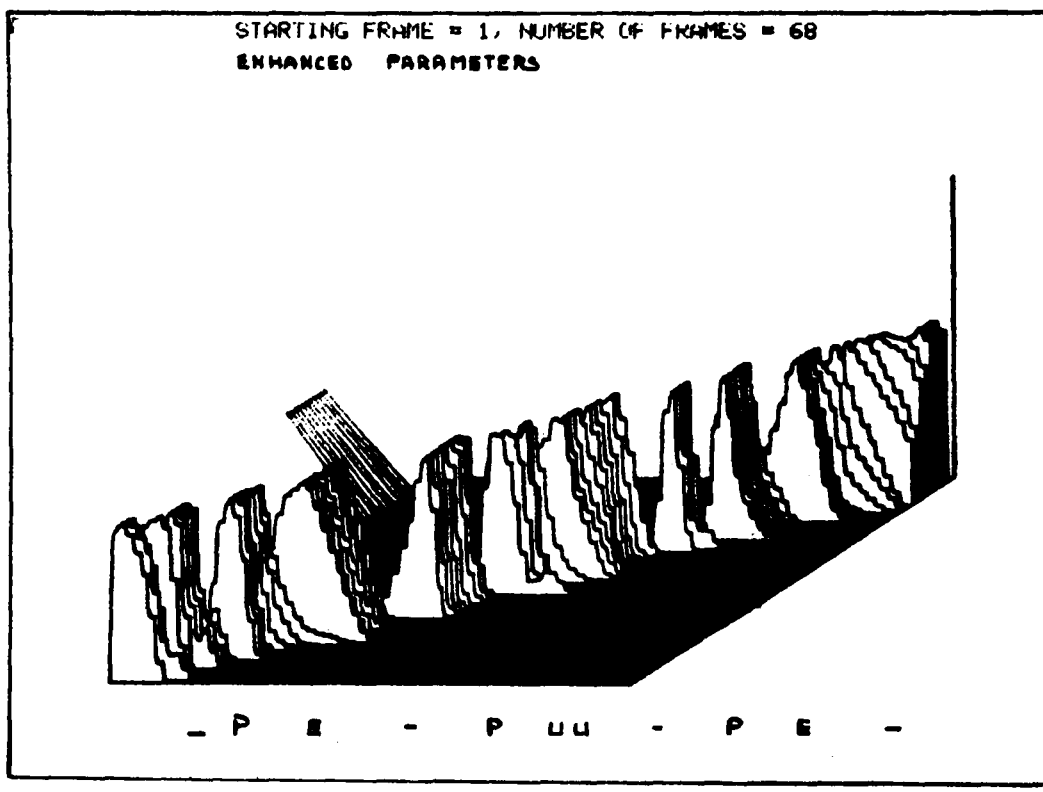

STARTING FRAME = 1, NUMBER OF FRAMES = 68
ENHANCED PARAMETERS

FIG. 10

5.4 Investigation of the Phi Functions

To investigate the actual phi functions found, a program was written to produce a three dimensional plot of the phi functions as a function of 1. Figure 3 shows one such plot. On the horizontal axis we have time. Vertically it is the magnitude of the phi function. The axis into the page is 1, the position over which the window is centred. For simplicity the phi functions are smoothed and truncated to their main lobes.

A few observations and modifications to the procedure resulted from the 3 dimension plot of phi functions.

Sidelobes. It was realized that the phi functions found usually have side lobes. However we are only interested in and use the main lobe of the phi function. This has a few consequences. The side lobes have an effect on the location function. Sometimes the side lobes vary erratically and so does the location function.

Another problem is that, because there are side lobes present, the main lobe is not the true phi function. Usually this effect is minimal. The presence of side lobes, we can say, does not occur in the desired phi function. This is because phi functions are to represent one articulatory movement. A side lobe can only be interpreted as two movements, in opposite directions.

The effect is most noticeable in transitional regions. The calculate phi function is the result of two or more equally contributing "desired" phi functions. In these cases a lot of the energy of the phi function is in the side lobes.

Solutions. To minimize the effect that side lobes have on the location function, the phi functions are truncated to the main lobe before evaluation of the location function. The main lobe is the section of the phi function that has the same sign as that at the point "1". Because the main lobe in most cases is stable the resulting location function should not be so erratic.

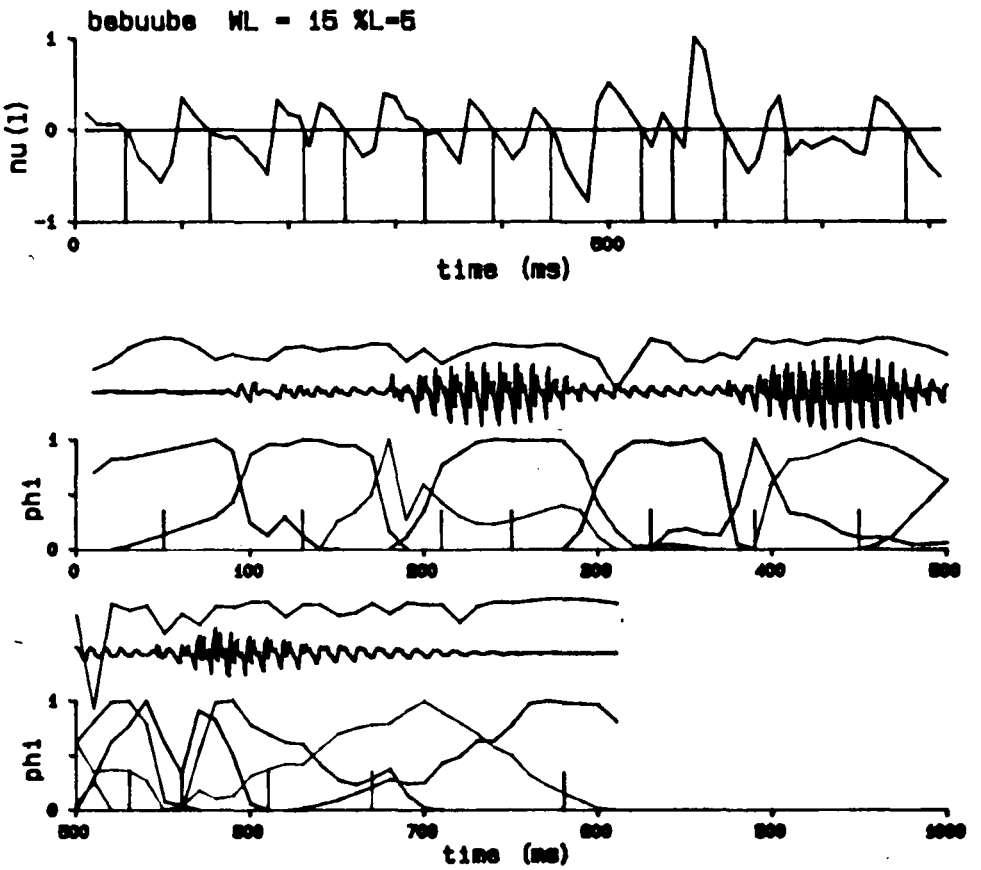
The only successful way I have found to deal with phi functions in the transition regions, is to ignore them. From equation 8 we notice that resulting phi functions are normalized by

$$\sum_{n} \varphi^2(n) = 1 \qquad (18)$$

If after truncation, the "energy" of the phi functions has a value of 0.7 or less, then the phi function is ignored. Under these circumstances we assume that the phi function calculated is a result of two or more actual phi functions interacting.

The results for these modification are shown in figure 11 for the utterance "bebuube". There are a few things to note. Firstly unlike "beboobe", some of the plosives are in two sections. A silence where the mouth is closed followed by

the rapid opening of the mouth. The location function is smoother in the second case. Here the last "e" is represented by one phi function as desired and the mess between 500 and 600 mSecs has been cleaned up.


bebuube WL = 15 %L=5
nu(1)
time (ms)
phi
phi
time (ms)

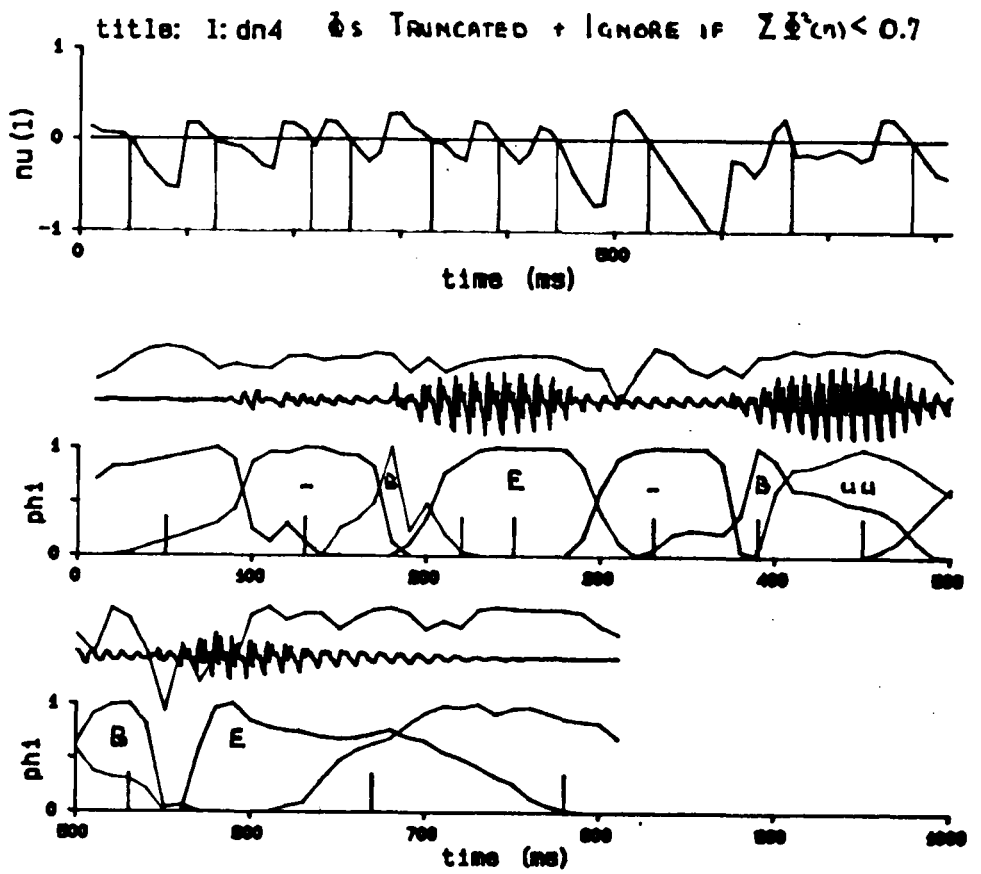

title: I:dn4 Φs Truncated + Ignore if Σ Φ²(n)< 0.7
nu(1)
time (ms)
phi
B
E
B
E
phi
time (ms)

FIG 11

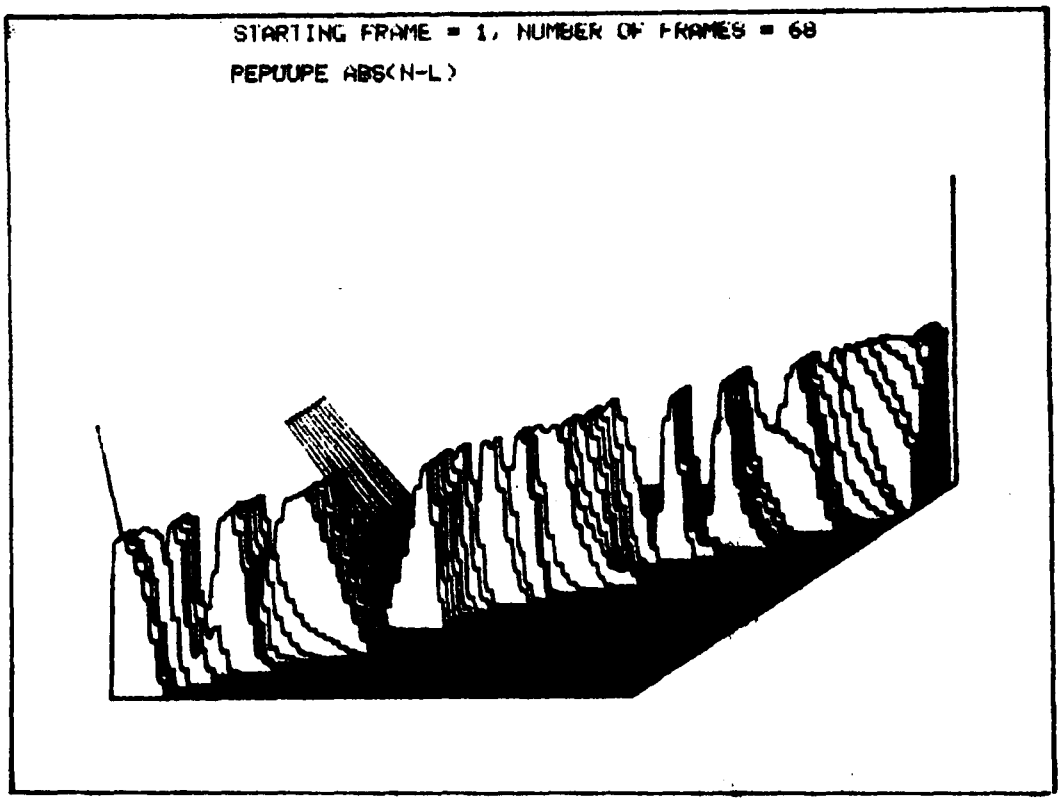
5.5 Further observations.

A look at figure 3 shows that there are distinct regions of phi functions. In these regions, all the phi functions are similar. Between these regions the phi functions either change abruptly or smoothly. An abrupt change is prefered, as from these it is easier to identify phi function boundaries.

In an attempt to obtain more abrupt changes, equation 8, the measure of distance, was modified.

The $(n-1)^2$ factor was changed to $(n-1)^4$ and to ABS(n-1)

The resulting phi functions are plotted in figure 12 for the word "beboobe". Compare this with figure 10. No significant advantage is seen in either of the plots. For the ABS(n-1) factor the changes are less abrupt, but in the case of $(n-1)^4$, there is not much improvement.


STARTING FRAME = 1, NUMBER OF FRAMES = 68
PEPUUPE ABS(N-L)

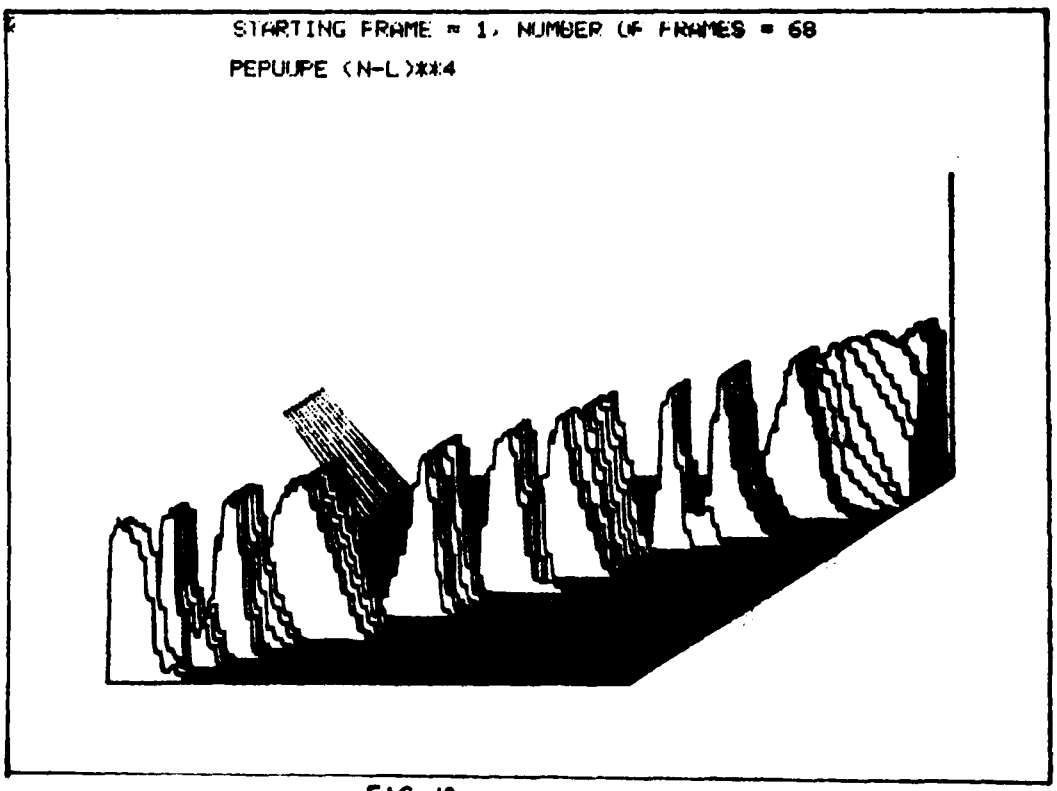

STARTING FRAME = 1, NUMBER OF FRAMES = 68
PEPUUPE (N-L)**4

FIG 12

5.5.1 Begin and End of Utterances

Normally for phi analysis we use a symmetrical rectangular window. At the beginning and end of segments this isn't possible. The original program shortened the window on the side where the data was missing. The window about frame 2 would start at frame 1 and end at sample 17 (A normally 300 mSec window with 10 mSec between frames). The result is a shorter unsymmetrical window.

We would obtain identical results when using the skewed window as when we use a symmetrical window with the data zero before frame 1 and after the last frame. A bad effect is that the first and last phi functions may not be found. This happens because the phi function as calculated is actually zero over some section. Before it is properly located the next phi function becomes predominant in the measure of distance function (equation 8).

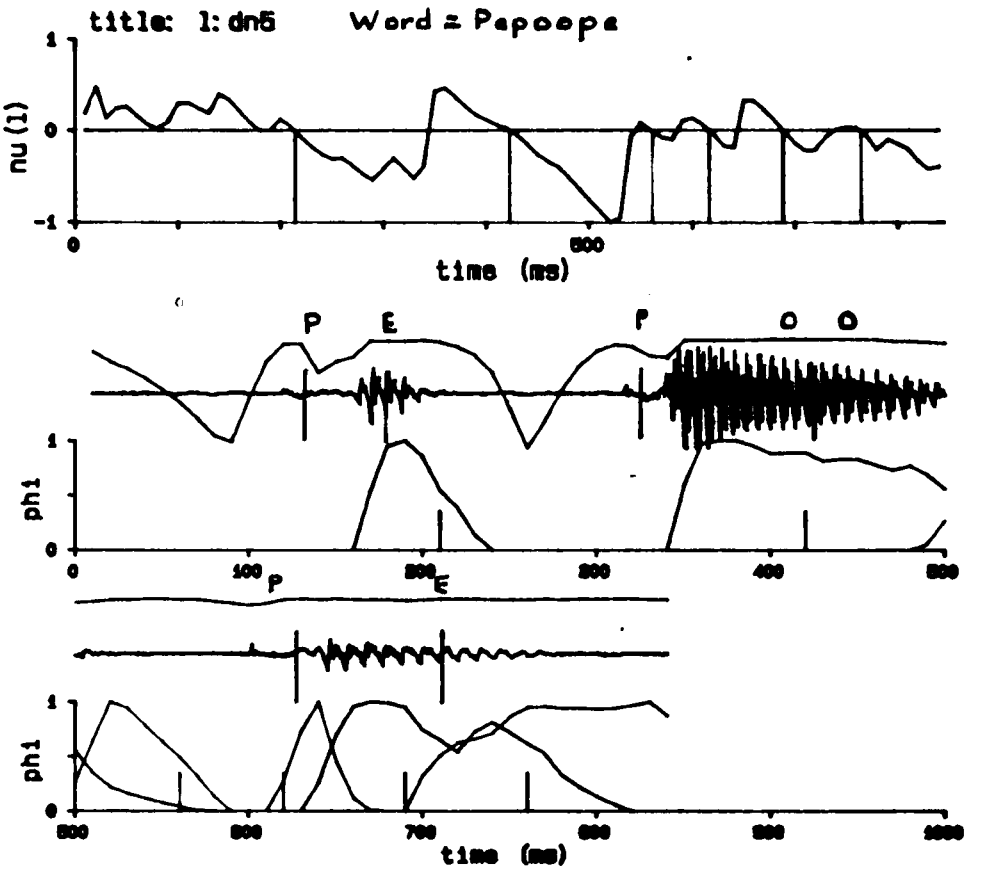
Several methods were tried to compensate this effect. Each of them involved extending the data in some way.

1) Reflect the data about frame 1.

2) Reflect the data and multiply it by a decaying function.

3) Extend the first sample by two frames.

Peculiarly enough the 3rd method seems to work best. A comparison between extended data and normal analysis is seen in figure 13. This in general is a bad run for it misses a lot of phonemes. The important thing to observe is that with the extended data the first phi function is located. Fig. 14 gives some idea why this is a difficult segment of data. The plot of all phi functions shows only a few regions where the phi functions are stable. However the first phi function is clearly visible, but not found until the data is extended. On the resulting analysis run, the first phi function is highly distorted. This arises from the iterative refinement's attempt to minimize the error.


title: 1:dn5 Word = Papoopa
nu(1)
time (ms)
P E P O O
ph1
P
E
time (ms)

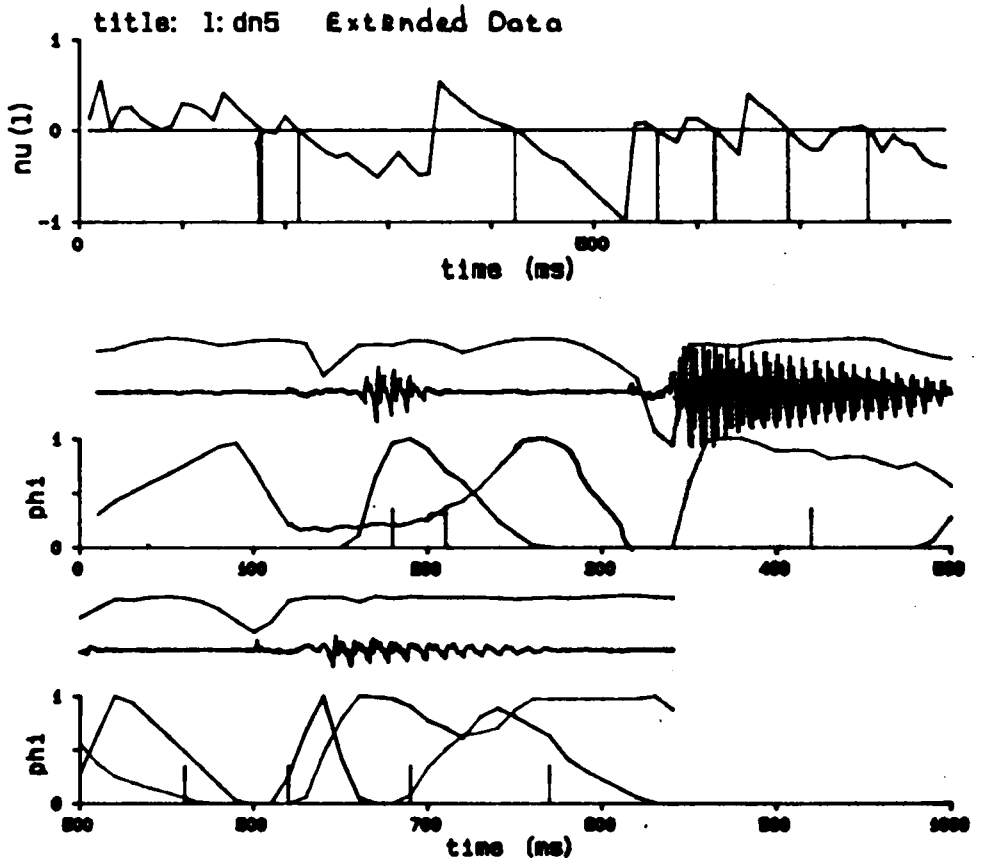

title: 1:dn5 Extended Data
nu(1)
time (ms)
ph1
time (ms)

FIG. 13

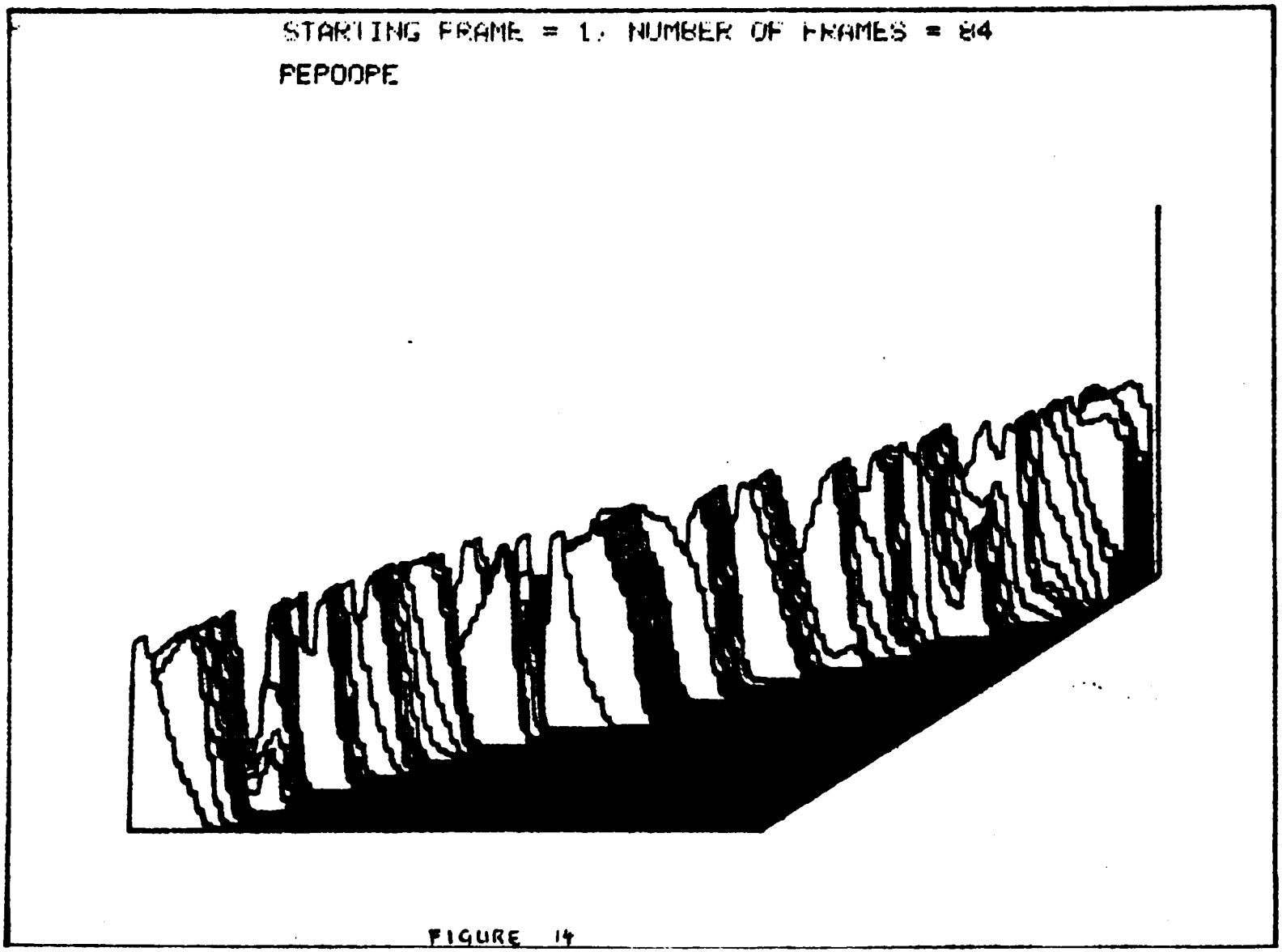


FIGURE 14

5.5.2 Other Problems

For a the utterance "pepaape" some more shortcomings of the system were noticed. Figure 15 shows the 3 dimensional plot of all phi functions, while figure 16 shows the result of the complete analysis. Notice that for the first "P" and the vowel phoneme there is no phi function found. Clearly there are phi functions for this phoneme in the plot of all phi functions.

A possible explaination is that the sidelobes have a strong effect on the measure of distance. The effects at the edge of the window may have a too big influence on the procedure to find phi functions. The resulting phi functions are primarily due to decreasing the effects away from the location point.

To counter this a Hamming window, instead of the rectangular window, was employed to select small segments of the utterance. The results of using the Hamming window are seen by the segmentation plot in figure 16. This time all the phi functions are present.

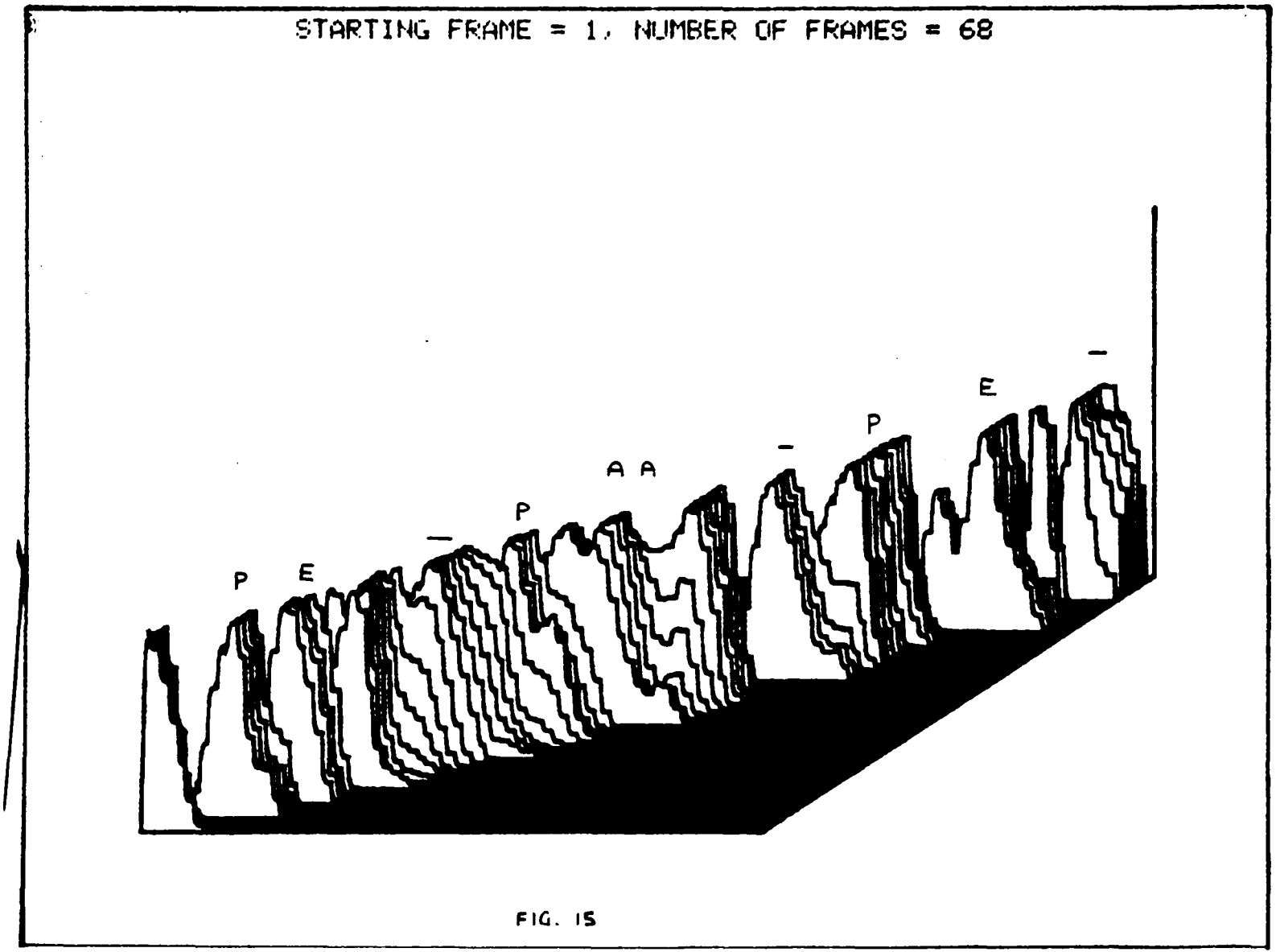


FIG. 15

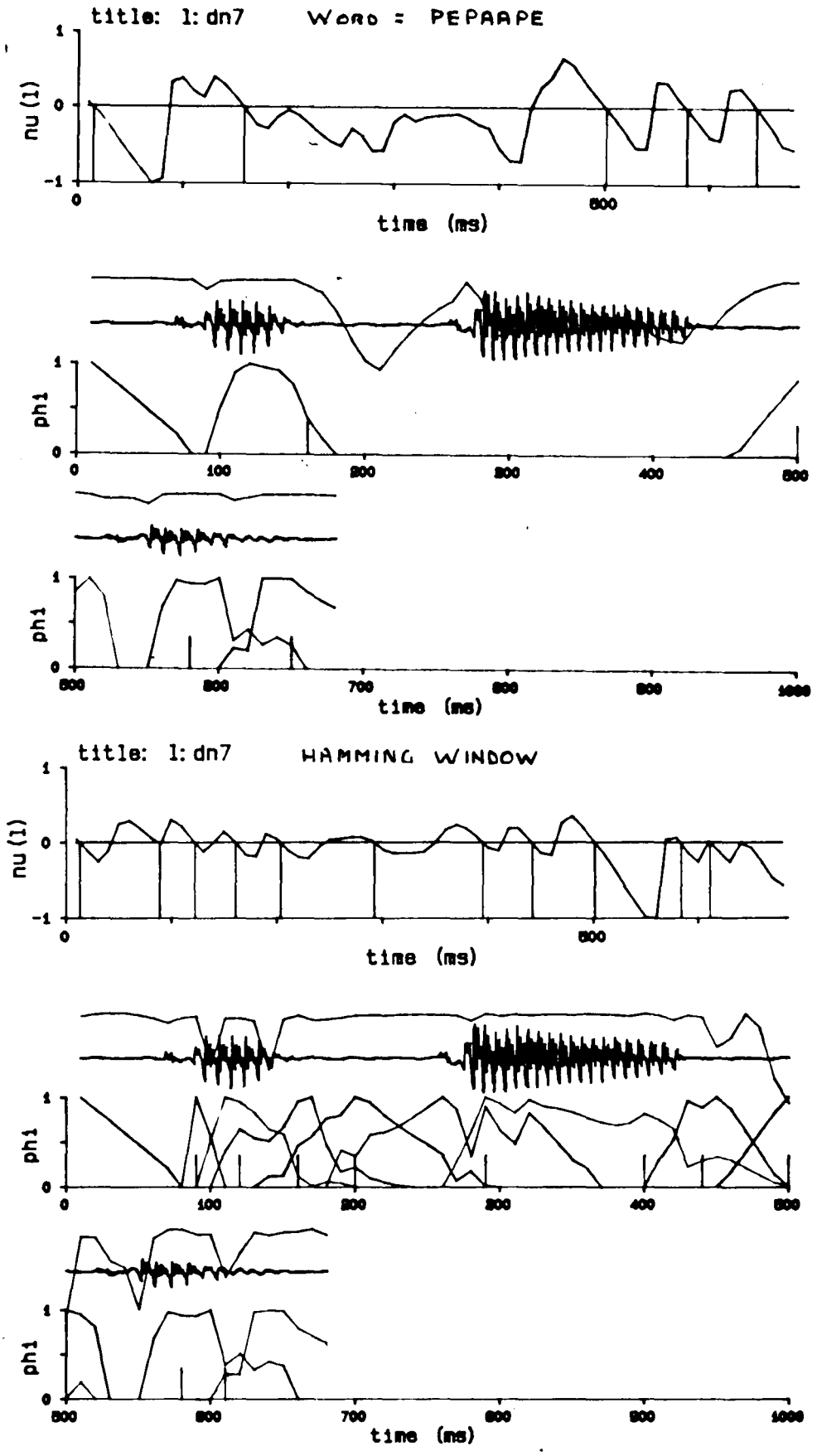

title: 1: dn7 Word = PEPAAPE
nu(1)
time (ms)
phi
title: 1: dn7 HAMMING WINDOW
nu(1)
time (ms)
phi
time (ms)

FIG. 16

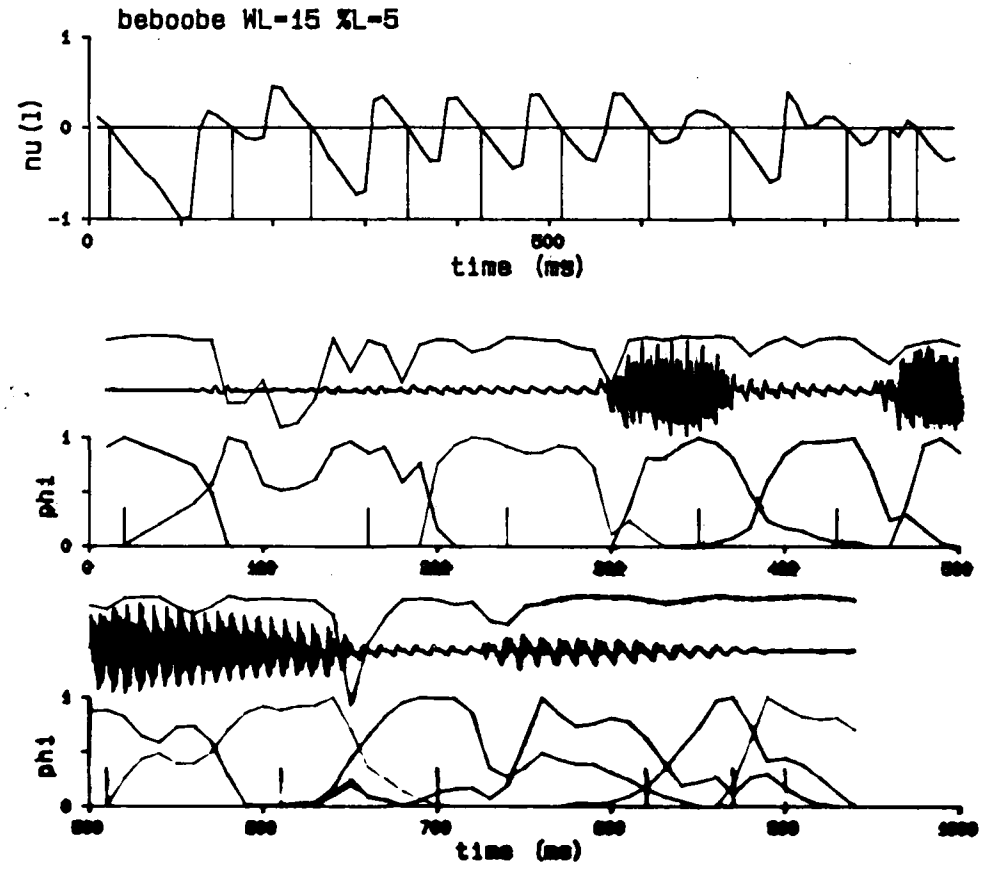
Yet another problem was noticed. Even though a phi function was found in the location procedure, the function never manifested itself in the output. The reason has to do with the system's sensitivity to the window length. In using the second window length of five functions, the phi function can change altogether. The solution is simple. Use the phi functions found using the fixed window length. This may cause more steps to be taken in the iterative refinement. The iterative refinement process seems powerful enough to handle less than optimal starting functions and still produce decent results.
Computation time used by the extra steps in the iterative refinement is compensated by not recalculating the phi functions.

5.5.3 Extra Phi functions.

Although the Hamming window does solve a few problems it also creates some. The Hamming window causes spurious phi functions to be located. (fig 14) Even when the Hamming window is not used, spurious phi functions might be found. A few techniques were tried to eliminate the spurious phi functions.

5.5.3.1 Smooth the Location Function.

The same moving average low pass filter as used for smoothing the log area data, was applied to smooth the location function. This compensated for small fluctuations in the found phi functions. By smoothing the location function, truncating phis before evaluation of the phi functions and ignoring phi functions with high side lobe energy, a lot of the spurious phi functions are eliminated. See figure 17.


beboobe WL=15 %L=5
nu(1)
time (ms)
phi
phi
time (ms)

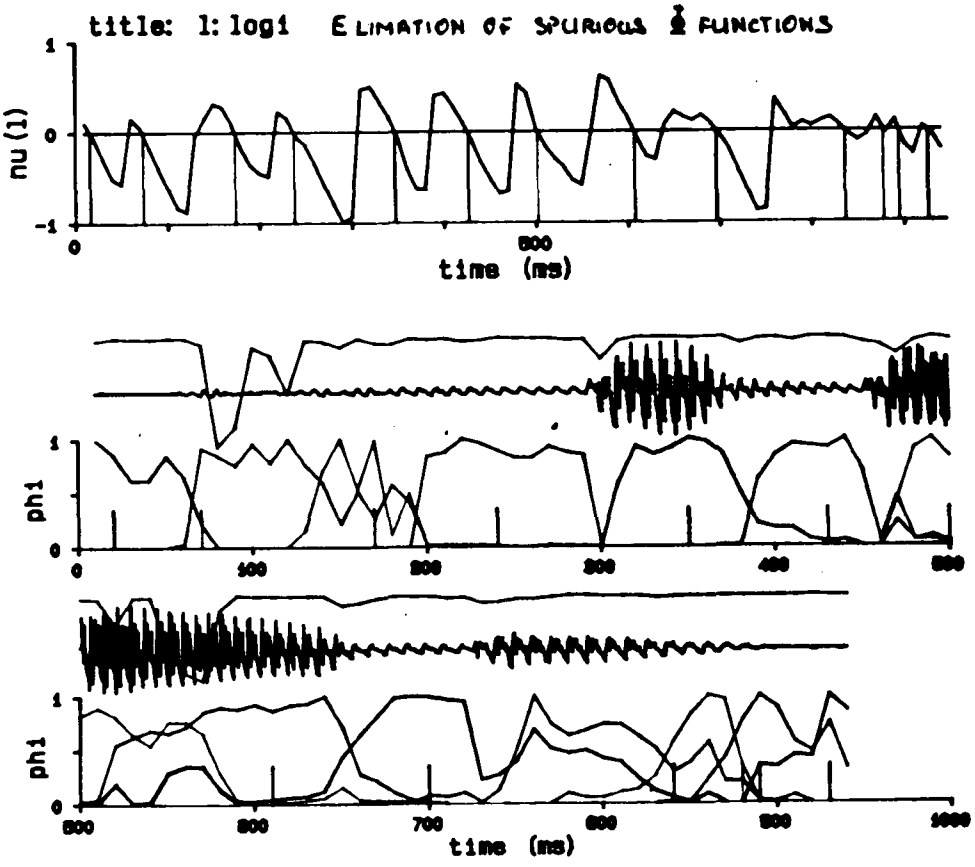

title: 1: log1 ELIMATION OF SPURIOUS ℓ FUNCTIONS
nu(1)
time (ms)
phi
phi
time (ms)

FIG 17

5.5.3.2 Elimination Based on Inproducts.

The inproduct between two phi functions is defind by

$$I_{1,2} = \sum_{n} \varphi_1(n) . \varphi_2(n) \qquad (19)$$

We first normalize the phi function to have a peak value of 1. Next we calculate the inproducts

$$I_{1,1}; \; I_{1,2} \; ; \; I_{2,2} \; .$$

The ratios

$$I_{1,2}/I_{1,1} \text{ and } I_{1,2}/I_{2,2} \qquad (20)$$

tell us how much of phi function 1 or 2 respectively is contained in the other phi function. We consider the greater of the two ratios. If the ratio is greater then a threshold, that phi function is eliminated. The normal threshold used is 0.85 . A higher ratio indicates that one phi function virtually duplicates the other phi function. This process works well but has one drawback. If two valid phi functions (those representing phonemes) are closely overlapping, one will be eliminated. This procedure is used for each phi function pair before every iterative refinement step..

5.5.3.3 Elimination on Error Calculations

I noticed that, when phi functions are eliminated, the error in the next iterative refinement step does not increase by a large amount. Sometimes it even decreases. This suggested two other methods to find spurious phi function.

After the iterative refinement has finished, throw away each phi function in turn. Perform an iterative refinement step after each phi function is thrown away. If the error still decreases, eliminate that phi function. When there is an increase in error, incorporate that phi function as a valid one. This procedure however was not succesful.

The second method considers three consecutive phi functions in time. We perform a step of the iterative refinement for the data covered by the middle phi function, with the three phi functions. The step is then repeated with only two outer phi functions. If the error is lower or only slightly higher the middle phi function is eliminated.

Both of these methods don't work. They eliminate phi functions corresponding to phonemes, and maintain those which should be considered spurious.

5.6 Other methods to find phi functions.

Even with all the modifications outlined, the procedure does not work well. Parameters can be set for utterances. These parameters have different optimal values for different utterances.

Some other methods for locating phi functions were tried.

5.6.1 Derivative of Log Area Functions.

Instead of the location function the sum of the first differences of the log area parameters was used. Phi functions tend to be found where the log area parameters all have peaks or throughs. Hence

$$\nu'(1) = \sum_{i=1}^{p} Y_i(1)-Y_i(1-1) \qquad (21)$$

This proved to be a very erratic function. Figure 18 shows a smoothed version. Even here the function is too wild to obtain useful information from it. Compare this to figure 17.

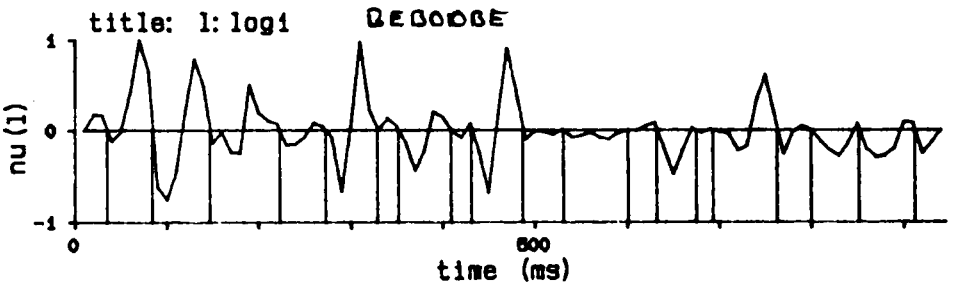


FIG 18

5.6.2 Phi detection by comparing Consecutive Phis.

In this method we base our detection of phi functions on the inproducts of consecutive phi functions. After a phi function is calculated, at each value of 1, the functions are truncated to the main lobe. They are still normalized as per equation 18. The following inproducts are calculated, for this and the previous phi function.

$$I_{1,2};\ I_{2,2};\ S_{1,2} = I_{1.2}/\sqrt{I_{1,1}\cdot I_{2,2}} \qquad (22)$$

The idea is that when we have a phi function that exists in the data, the neighbouring phi functions are similar. We try to group phi functions that have varied slowly. At the boundaries between phonemes the phi functions should change quickly, as a result we expect $S_{1,2}$ to be low. When we are near the centre of a phi function $S_{1,2}$ should be high. In general this seems to hold true. The phi function with the highest $S_{1,2}$ of a stable group is chosen as the best representative of that group of phi functions. The problem now becomes interpreting the data. A few methods were tried to separate the stable and transition regions.

1) By two threshold values. Once $S_{1,2}$ goes above the first it is considered to be in a stable region. When $S_{1,2}$ goes below the second it is considered to be in the transition region until it goes above

the first again.

2) As the first method, but this time the thresholds are differential. That is the change in $S_{1,2}$ has to go above or below a threshold.

3) As the first method but now the thresholds are not fixed. The thresholds are calculated as a certain amount above or below the maximum or minimum value of $S_{1,1}$ reached in that group.

Of the three methods the third seems to work the best. For a trial of 18 utterances I found that the optimal threshold values depended on the data. In some cases no values of the thresholds would locate all the phi functions. This is as the lower threshold had to be higher than the high threshold to locate all phi functions. The third method was motivated by this fact. Phi functions found using the normal location procedure could not always be found with this method.

5.6.3 Proposed Location Procedure.

The last location procedure motivated me to develop another method for locating phi functions. "Inproducts" are still used, but in a different way. The algorithm is as follows:

1) Calculate the phi functions about the frame l=1. Take this to be the first phi function. Calculate $I_{1,1}$.

2) Calculate the phi function about the next location. (l=l+1). Calculate $I_{2,2}$.

3) If $I_{2,2}$ is less than 0.7 ignore that phi function and goto step 7.

4) Calculate $I_{1,2}$.

5) If $I_{1,2}$ is greater than 0.7 then assume that the phi functions belong to the same group. Decide which phi function is better and maintain that one as the representative of that group. Throw away the other.

6) If $I_{1,2}$ is less than 0.7 then the phi functions belong to different phonemes. Save the first phi function as the phi function for the first phoneme. Keep the second as an initial phi function for the second phoneme. This function is used for comparison in step 3.

7) Goto step 2 for all frames.

5.6.3.1 Deciding the better phi function.

Decisions for which phi function is better is based on a score. The score is based on the measure of distance function (Equation 8). We normalize the minimum value of $\theta^2(1)$ so not to penalize wide phi functions. Normalization is based on giving any two rectangular phi functions, symmetrical about 1, the same score. We calculate the score thus:

$$S(1) = \theta^2_{mim}/((1-IB0.(1-IB-1)+(IT-1)(IT-1-1) \quad (23)$$

Where IB is the first non zero frame of the phi function
and IT is the last non zero frame of the phi function.

This method was implemented. Time however did not permit me to fully investigate this method. At first look it does look promising. The method tracks stable phis and selects the phi function, as our criterion requires, that is the most compact about 1.

5.7 Iterative Refinement Procedure.

Most of my work has concentrated on locating phi functions, and the associated problems. The iterative refinement procedure may need improvement as well.

The iterative refinement procedure suffers from two problems:

1) It tends to expand phi functions.

2) It often happens that the iterative refinement causes the error to start increasing.

The fact that the iterative refinement is a two step process, minimizes the extent at which the phi functions expand.

The increase in error is often sharp and sudden. The first three steps may cause gradual decrease in error, but the fourth, a significant increase. Unfortunately there was not enough time for me to consider these problems.

6 Performance of the Analysis Method.

To test how "well" the procedure works, the program analysed 18 utterances. The utterances were chosen represent various classes of consonants and important vowels. The utterance are all dutch nonsense words and are:

beboobe bebeebe bebaabe bebuube	voiced plosives
pepoope pepeepe pepaape pepuupe	unvoiced plosives
nenoone neneene nenaane nenuune	nasal consonants
sesoose seseese sesaase sesuuse	unvoiced fricatives
keklaake seslaase	combinational consonants

The performance with the standard location function works best for:

Window length = 250 mSecs
%feature loss = 5%

For plosives it was found that a smaller window length usually works better, while for other consonants, longer window lengths can be advantageous. The same combination of parameters works well with the last location procedure discussed.

The main problem areas seem to be plosives. Usually the plosive after the pause is too small for detection. It is sometimes smoothed out by the LPC analysis. Other times the plosive appears as two parts: the silence and the actual plosive. Fricatives being comprised of coloured noise are also hard to analyse. As the signal is basically stochastic in nature the L.A. parameters are subject to fluctuations. This often manifests itself as more than one phi function being required to descibe the consonant.

Vowels are usually the easiest to detect. There are problems with long vowels. During the vowel the log area parameters may start to change. This usually introduces an extra phi function. Alternatively a vowel may have a diphthong nature, and once again be represented by two phi functions.

7 Conclusions.

Automatic segmentation of speech by this method is still some way off. Numerous problems have been detected, the solutions still not fully found.

The question remains, can this method segment speech? Indications are that it can. There are too many phi functions found that correspond to phonemes, to say definitely no. The problems associated in obtaining these phi functions are numerous. It may be that a complete rethinking of the method is required before automatic segmentation is achieved. Certainly the analysis with the resynthesized data shows that if the data is made up of a series of articulatory movements, then we can obtain segmentation with this method.

The method works well for speech coding. There are more phi functions than the rank of speech matrix. This ensures that the iterative refinement can always minimize the error to obtain good speech resynthesis. Detecting phoneme boundaries is not so important for speech coding. It is harder to match phi functions and phonemes, than to obtain phi fuctions to code speech.

A Word of Thanks.

I would like to thank the following people for their assistance with this project & report: L.Willems, S. Marcus, and J van Hemert. They gave a lot of their time in discussion and helped develop some ideas. A special thanks to G. Doodeman as well for his help in producing this report.

References

[1] B.S. Atal, "Efficient Coding of LPC parameters by Temporal Decomposition." Proceedings of the ACASSP 83, Boston, pp 81-84. See appendix a.

[2] J.D. Markel and A.H. Grey Jr., Linear Prediction of Speech. New York: Springer-Verlag, 1976.

[3] J 't Hart, S.G. Nooteboom, L.L.M. Vogten and L.F. Willems, "Manipulation of Speech Sounds." Philips Tech. Rev. 40, 134-145 1982,No. 5.

[4] R. Viswanathan and J. Makhoul, "Quantization Properties of Transmission Parameters in Linear Predictive Systems." IEEE Trans. Acoust. Speech, Signal Processing Vol ASSP-23 June 1975.

[5] Fukunaga, Introduction to Statistical Pattern Recognition. New York: Academic Press, 1972.

A B. S. Atal's Paper

EFFICIENT CODING OF LPC PARAMETERS BY TEMPORAL DECOMPOSITION

Bishnu S. Atal

Bell Laboratories

Murray Hill, New Jersey 07974

ABSTRACT

This paper describes a method for efficient coding of LPC log area parameters. It is now well recognized that sample-by-sample quantization of LPC parameters is not very efficient in minimizing the bit rate needed to code these parameters. Recent methods for reducing the bit rate have used vector and segment quantization methods. Much of the past work in this area has focussed on efficient coding of LPC parameters in the context of vocoders which put a ceiling on achievable speech quality. The results from these studies cannot be directly applied to synthesis of high quality speech. This paper describes a different approach to efficient coding of log area parameters. Our aim is to determine the extent to which the bit rate of LPC parameters can be reduced without sacrificing speech quality. Speech events occur generally at non-uniformly spaced time intervals. Moreover, some speech events are slow while others are fast. Uniform sampling of speech parameters is thus not efficient. We describe a non-uniform sampling and interpolation procedure for efficient coding of log area parameters. A temporal decomposition technique is used to represent the continuous variation of these parameters as a linearly-weighted sum of a number of discrete elementary components. The location and length of each component is automatically adapted to speech events. We find that each elementary component can be coded as a very low information rate signal.

INTRODUCTION

A long standing goal of speech research has been to develop a simple and efficient description of speech events. Such a description is important for many practical applications, such as speech coding, speech synthesis, and speech recognition. For example, in speech coding, our aim is to represent the speech wave by a small number of time-varying parameters which are capable of regenerating speech at low bit rates without significant distortion. Speech wave has a bandwidth of about 4 kHz. Speech parameters, such as a log area parameter determined by LPC analysis [1-4], can be limited in bandwidth to about 50 Hz without introducing any additional distortion due to band limiting [3,4]. The total bandwidth for 12 log area parameters is therefore 600 Hz, which is considerably lower than 4000 Hz required for the speech signal. A major source of redundancy in LPC area parameters arises from the correlations between successive time frames. These correlations are caused by a number of factors involved in human speech production. Most obvious of these is the smooth movement of different articulators in the vocal tract.

A common method of coding log area parameters is time sampling followed by scalar or vector quantization [5,6]. If each parameter is band limited to 50 Hz, it can be sampled at 100 Hz without loss of information. Scalar quantization of each frame of log area parameters typically requires 48 bits which yields a bit rate of 4800 bits/sec. What can be done to reduce this bit rate? One possibility is to reduce the bandwidth of each parameter even more. For example, if the bandwidth is lowered to 25 Hz, the parameters can be sampled at 50 Hz yielding a bit rate of 2400 bits/sec. However, a bandwidth of 25 Hz is usually too small to represent fast variations of short transient sounds accurately.

Speech events occur generally at non-uniformly spaced time intervals. Moreover, articulatory movements for some speech sounds are fairly slow while for others they are relatively fast. Uniform sampling of speech parameters is thus not efficient. With uniform sampling, one is forced to use a small sampling interval to be able to represent the fastest speech event accurately. Non-uniform sampling of speech parameter variations is in general more efficient because the sampling interval can be adapted to the nature of speech events. Since speech sounds are produced in human speech at an average rate of approximately between 10 and 15 sounds/sec, it should be sufficient to specify the acoustic parameters at an average rate of less than 15 frames/sec. In this paper, we present a procedure to break up the continuous variation of log area parameters into discrete units of variable lengths located at non-uniformly spaced time intervals. Coding efficiency is achieved by coding these units rather than the parameters themselves.

TEMPORAL DECOMPOSITION MODEL FOR LOG AREA PARAMETERS

Consider the variation of log area parameters as a function of time. Let $y_i(n)$ be the ith log area parameter at the nth sampling instant. It is assumed that the parameters have been sampled at closely spaced time intervals small enough to represent accurately even the fastest speech events. The sampling interval is typically 1 to 2 msec. The index i varies from 1 to p where p is the total number of area parameters determined by LPC analysis. The value of p is typically 16 for speech sampled at 8 kHz. The index n varies from 1 to N where $n=1$ is the first sample in the utterance and $n=N$ is the last sample in the utterance. Figure 1 shows the first 8 log area parameters for the utterance "Joe brought a young girl" spoken by a male speaker. The rms amplitude is also shown on the figure.

We represent $y_i(n)$ as

$$\hat{y}_i(n) = \sum_{k=1}^{m} a_{ik}\phi_k(n), \quad 1 \leqslant n \leqslant N, \quad 1 \leqslant i \leqslant p, \tag{1}$$

where $\hat{y}_i(n)$ is the approximation of $y_i(n)$ produced by the model, $\phi_k(n)$ is the kth interpolation function at the sampling instant n, and a_{ik} is the contribution of the kth interpolation function to the ith area parameter. The value of m corresponds roughly to the number of speech (and silence) events in the speech utterance in the time interval $n=1$ to $n=N$.

Equation (1) can be expressed in matrix notations as

$$Y = A\,\Phi \tag{2}$$

where Y is a $p \times N$ matrix whose (i,n) element (ith row and nth column) is $y_i(n)$, A is a $p \times m$ matrix whose (i,k) element is a_{ik}, and Φ is a $m \times N$ matrix whose (k,n) element is $\phi_k(n)$. We wish to determine matrices A and Φ so that the bit rate required to represent them is minimum.

ICASSP 83, BOSTON CH1841-6/83/0000-0081 $1.00 © 1983 IEEE

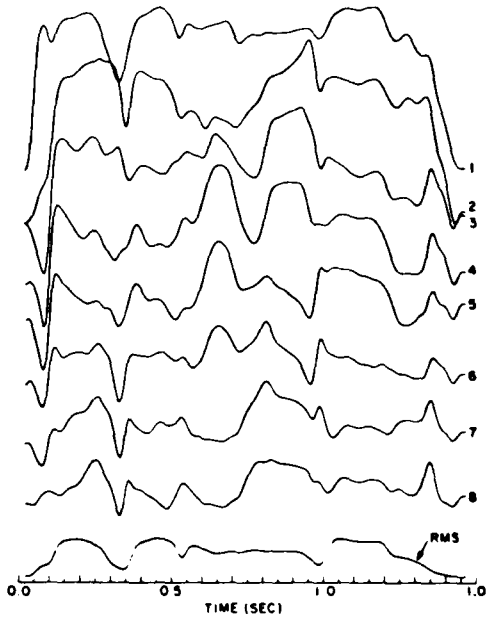


Fig. 1. Plot of first 8 log area parameters and rms amplitude as a function of time for a sentence-length utterance, "Joe brought a young girl", spoken by a male speaker.

We will assume that the functions $\phi_k(n)$ are ordered with respect to their locations in time. That is, the function $\phi_2(n)$ occurs later than the function $\phi_1(n)$ and so on. Each $\phi_k(n)$ is supposed to correspond to a particular speech event. Since a speech event lasts for a short time, each $\phi_k(n)$ should be non zero only over a small range of values of n. A typical $\phi(n)$ is sketched in Fig. 2. For efficient coding, the matrix Φ should be a sparse matrix.

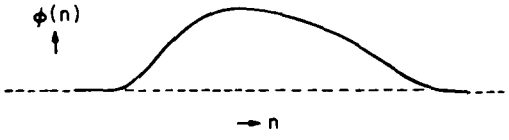


Fig. 2. Idealized sketch of a typical interpolation function.

We illustrate the above point in the example shown in Fig. 3. We show there three functions of time $y_1(n)$, $y_2(n)$, and $y_3(n)$. These functions were constructed by combining the three functions of time $\phi_1(n)$, $\phi_2(n)$, and $\phi_3(n)$, shown in Fig. 3(b), using three different sets of coefficients a_{ik} in Eq. (1). Thus, all of the $y(n)$ of Fig. 3 follow Eq. (1) exactly. Since each $\phi(n)$ is limited to a much shorter interval in comparison to any one of the $y(n)$ and the bandwidth of each $y(n)$ is the maximum bandwidth of any one of the $\phi(n)$, it is obvious that direct coding of $y(n)$ will take more bits than the coding of the $\phi(n)$ and the coefficients used to combine $\phi(n)$ to form $y(n)$.

As mentioned earlier, the value of m in Eq. (1) is related to the duration of the speech segment and the number of sounds the speech segment contains. In general, m is proportional to N. Consider a short segment of speech such that the rank of the matrix $Y \geqslant m$. The maximum rank of the matrix Y is p, no matter how long the speech segment. Previous work suggests that the rank of Y is about 10 even for very long utterances. To satisfy the requirement that rank of the matrix $Y \geqslant m$, the duration of speech segment should be

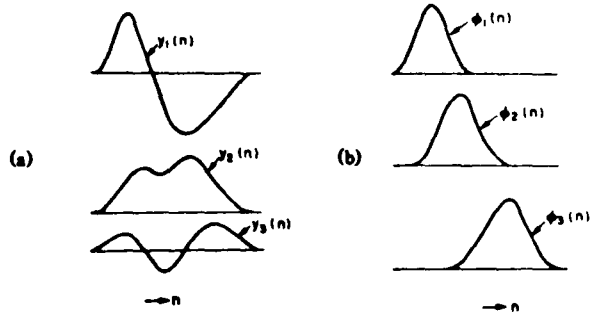


Fig. 3. (a) Three different linear combinations of the basis functions shown on the right and (b) the basis functions.

approximately 0.2 to 0.3 sec. Whenever the rank of $Y \geqslant m$, Eq. (2) can be inverted to yield

$$\Phi = (A'A)^{-1}A'Y, \tag{3}$$

which implies that

$$\phi_k(n) = \sum_{i=1}^{p} w_{ki} y_i(n), \quad 1 \leqslant k \leqslant m, \quad 1 \leqslant n \leqslant N, \tag{4}$$

for some choice of the weights w_{ki}. That is, each interpolation function ϕ is a linear combination of the y's.

The problems related to determining the rank of Y are easily resolved by looking at the eigenvalues obtained from the singular-value decomposition of Y. We represent Y as

$$Y' = U\,D\,V', \tag{5}$$

where U is a $N \times p$ orthogonal matrix, V is a $p \times p$ orthogonal matrix, D is a diagonal matrix of eigenvalues, and the superscript t on a matrix means its transpose. Typical values of the first ten eigenvalues, for a short speech segment 0.25 sec in duration, are 0.83, 0.52, 0.16, 0.13, 0.06, 0.03, 0.03, 0.02, 0.01, and 0.01, respectively. Assuming that an error of 0.05 in log areas is insignificant, the rank of Y is 5. We then set m to 5.

It is obvious from Eqs. (4) and (5) that an interpolation function $\phi_k(n)$ can also be represented as

$$\phi_k(n) = \sum_{i=1}^{m} b_{ki} u_i(n), \tag{6}$$

where $u_i(n)$ is the element in the nth row and the ith column of the matrix U and b_{ki} are a set of amplitude coefficients.

DETERMINATION OF INTERPOLATING FUNCTIONS

We define a measure of distance of $\phi(n)$ from the sample $n=l$ as

$$\theta(l) = [\sum_n (n-l)^2 \phi^2(n) \,/\, \sum_n \phi^2 n]^{1/2}, \tag{7}$$

where the sum over the index n extends over the duration of the speech segment. The optimum $\phi(n)$ is chosen so as to minimize the distance function $\theta(l)$.

Minimization of $\theta(l)$

Since the problem of minimizing $\theta(l)$ is equivalent to the problem of minimizing $\ln \theta(l)$, we set the derivatives of $\ln \theta(l)$ with respect to the unknown amplitude coefficients b_{ki} of Eq. (6) equal to zero. We then obtain

$$\sum_n (n-l)^2 \frac{\partial}{\partial b_r} \phi^2(n) = \lambda \sum_n \frac{\partial}{\partial b_r} \phi^2(n), \quad 1 \leqslant r \leqslant m, \tag{8}$$

where

$$\lambda = [\sum_n (n-l)^2\phi^2(n) \;/\; \sum_n \phi^2(n)] = \theta^2_{min} \qquad (9)$$

From Eq. (6), we can write

$$\phi^2(n) = \sum_{i=1}^{m}\sum_{j=1}^{m} b_i b_j u_i(n) u_j(n), \qquad (10)$$

where the subscript k has been dropped. Then,

$$\frac{\partial}{\partial b_r}\phi^2(n) = 2\sum_{i=1}^{m} b_i u_i(n) u_r(n), \quad 1 \leq r \leq m. \qquad (11)$$

On combining Eqs. (8) and (11), one obtains

$$\sum_{i=1}^{m} b_i \sum_n (n-l)^2 u_i(n) u_r(n) = \lambda \sum_{i=1}^{m} b_i \sum_n u_i(n) u_r(n) = \lambda b_r. \qquad (12)$$

Equation (12) can be expressed in matrix notations as

$$R\mathbf{b} = \lambda \mathbf{b}, \qquad (13)$$

where the element in the ith row and rth column of the matrix R is given by

$$R_{ir} = \sum_n (n-l)^2 u_i(n) u_r(n). \qquad (14)$$

Equation (13) has exactly m solutions. If all the λ's are different, the solution corresponding to the smallest λ provides the correct $\mathbf{b}$. In case they are not, the minimum value of λ determines the optimum $\mathbf{b}$; although the choice of optimum $\mathbf{b}$ is not unique. The nearest $\phi(n)$ is determined from the coefficients b_i's by using Eq. (6). The location of the nearest $\phi(n)$ is given by

$$\nu(l) = [\sum_n (n-l)\phi^2(n) \;/\; \sum_n \phi^2(n)]. \qquad (15)$$

The function $\nu(l)$ crosses the $\nu(l)=0$ axis from the positive side at each sampling instant l which equals the location of one of the $\phi_k(n)$ for some k.

Better estimates of $\phi(n)$'s are obtained by repeating the minimization for all values of l for which $\nu(l)=0$, and using a time interval which contains exactly 5 speech events ($m = 5$). This indeed is always possible except at the beginning or at the end of an utterance which begins or ends with a silence. A lower value of m is used in these shorter segments. The first and last $\phi(n)$'s correspond to "silence" segments.

Determination of amplitude coefficients a_{ik}

The amplitude coefficients a_{ik} of Eq. (1) are determined by minimizing the mean-squared error E defined by

$$E = \sum_n [y_i(n) - \sum_{k=1}^{M} a_{ik}\,\phi_k(n)]^2, \qquad (16)$$

where M represents the total number of speech events within the range of index n over which the sum is carried out. On setting the partial derivatives of E with respect to the coefficients a_{ik} equal to zero, we obtain a set of simultaneous linear equations

$$\sum_{k=1}^{M} a_{ik} \sum_n \phi_k(n)\phi_r(n) = \sum_n y_i(n)\phi_r(n), \quad 1 \leq r \leq M, \quad 1 \leq i \leq p, \qquad (17)$$

which can be solved for the unknown coefficients a_{ik}.

Iterative Refinement of $\phi_k(n)$'s and a_{ik}'s

Figure 4 shows a plot (solid line) of the interpolation functions $\phi_k(n)$, obtained from the above procedure, for the example illustrated in Fig. 3. The actual functions $\phi_k(n)$ are also shown as dashed curve on the same plot. The agreement between the two is close except for the presence of a number of small ripples and the narrowing of the major lobe. The mean-squared criterion used for the distance function shown in Eq. (7) is a contributing factor for these differences. We discuss here an iterative refinement procedure for obtaining better estimates of $\phi_k(n)$ and a_{ik}. For a given set of

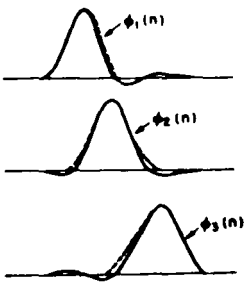


Fig. 4. Plots of the interpolation functions obtained by temporal decomposition of the curves shown in Fig. 3(a). The dashed curves are the actual basis functions illustrated in Fig. 3(b).

a_{ik}, we determine $\phi_k(n)$ to minimize the error E given in Eq. (16). This is done by setting the partial derivatives of E with respect to $\phi_k(n)$ equal to zero. One then obtains

$$\frac{\partial E}{\partial \phi_r(n)} = 2\sum_{i=1}^{p}[y_i(n) - \sum_{k=1}^{M} a_{ik}\phi_k(n)]a_{ir} = 0, \quad 1 \leq r \leq M, \qquad (18)$$

which further simplifies to

$$\phi_r(n) = [\sum_{i=1}^{p} y_i(n) a_{ir} - \sum_{k \neq r} \phi_k(n) \sum_{i=1}^{p} a_{ik} a_{ir}] \;/\; [\sum_{i=1}^{p} a^2_{ir}]. \qquad (19)$$

Since the the coding of minor lobes of $\phi(n)$ can use a significant number of bits, we retain only the major lobe of the interpolation functions and set the functions equal to zero every where else. The resultant $\phi_k(n)$ are used again in Eq. (17) to obtain an even better estimate of a_{ik}. The procedure is repeated until the decrease in error E, as defined in Eq. (16), falls below a predetermined threshold value. Four iterations are usually sufficient to converge both a_{ik} and $\phi_k(n)$ to stable set of values.

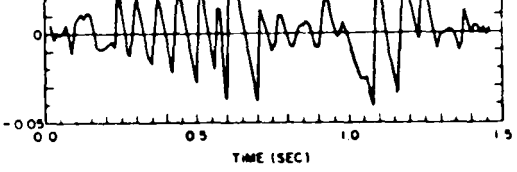


Fig. 5. Plot of the timing function $\nu(l)$ for the utterance "Joe brought a young girl" shown in Fig. 1.

RESULTS

The above procedure was carried out on several sentences spoken both by male and female speakers. We present results here for one sentence "Joe brought a young girl" spoken by a male speaker. The timing function $\nu(l)$ defined in Eq. (15) is illustrated in Fig. 5. A new value of $\nu(l)$ was computed once every 10 msec. Each zero crossing from positive to negative values indicates the location of a speech event. The zero crossings going from negative side to positive side signify a rapid shift from one $\phi(n)$ to the next. This shift is very sharp as expected. The function $\nu(l)$ has a total of 23 negative-going zero crossings. The interpolation functions $\phi(n)$ located at these time instants are shown in Fig. 6 together with the corresponding speech waveforms. As expected, the interpolation functions for short transient sounds last over a short time interval while the interpolation functions for relatively stationary vowel sounds last over a much longer time interval.

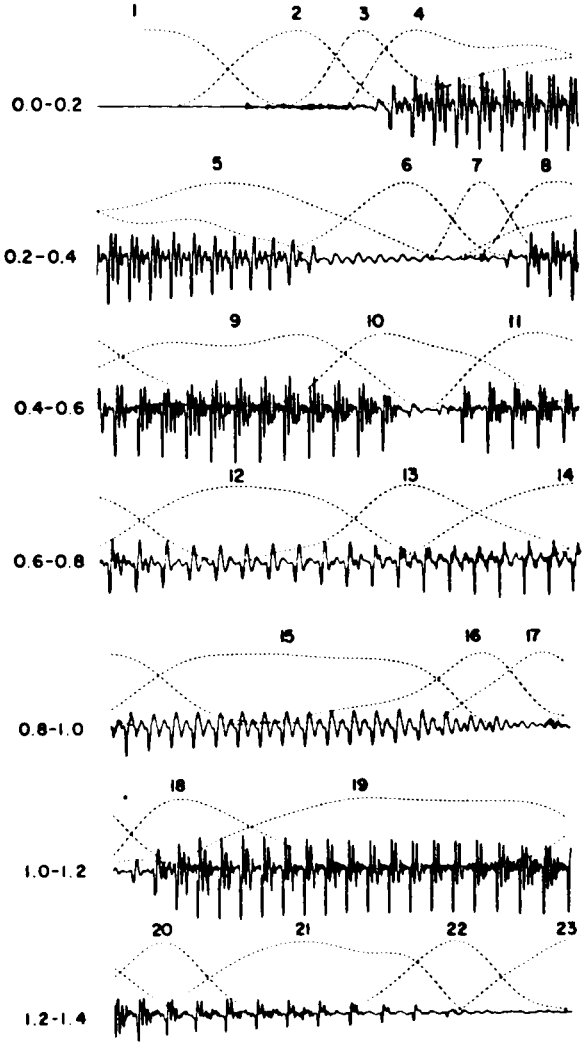


Fig. 6. Plots of speech waveform for the utterance "Joe brought a young girl" and the various interpolation functions determined by the temporal decomposition technique. The time intervals (in secs) for the different segments are marked in the left margin in each case.

Figure 7 shows the first 8 log area parameters and the rms value as a function of time for the utterance shown in Fig. 6. The solid curve shows the original areas determined by LPC analysis of the speech wave. The dashed curve shows the approximation of each $y_i(n)$ by the additive model defined in Eq. (1). The results for the remaining 8 log areas are similar. As can be seen, the agreement between the model and the actual results is very good.

Bit Rate Required to Encode Area Parameters

The interpolating functions in general vary smoothly as a function of time. We have determined the bandwidth of each interpolating function from its amplitude spectrum. An effective bandwidth for each spectrum can be defined as the frequency at which the amplitude spectrum falls to 1/20 of its value at d.c. We find that an average of 4 samples per $\phi_k(n)$ are needed to sample the function at the Nyquist rate.

It is sufficient to encode each sample of $\phi_k(n)$ at 4 bits/sample

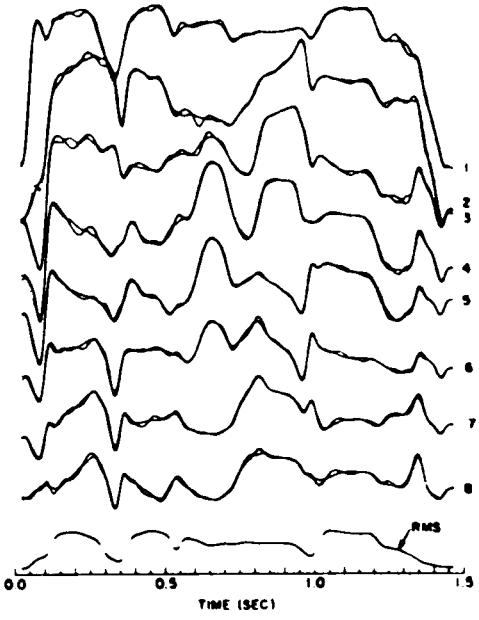


Fig. 7. Plot comparing the model-generated area parameters (dashed curve) with the actual areas (solid curve) of Fig. 1.

to keep the error in the log area parameters to be less than 0.10. Thus the total number of bits required to encode each $\phi_k(n)$ is 16 bits.

For each k, the coefficients a_{ik} need to be coded with the same accuracy as a single frame of log area parameters. With scalar quantization, we find that 48 bits/frame are sufficient [4]. Recent work on vector quantization suggests that number of bits/frame can be reduced even further [6].

The total information rate for encoding of log area parameters depends upon the number of speech events (or sounds) spoken per second. For slow speaking rate, this number is about 10. Assuming 5 bits to represent the location of each ϕ, the bit rate for coding both a_{ik} and $\phi_k(n)$ will then be $(48 + 16 + 5) = 690$ bits/sec. The bit rate would increase to 1035 bits/sec for a speaking rate of 15 sounds per sec.

REFERENCES

[1] J. D. Markel and A. H. Gray, Jr., *Linear Prediction of Speech*. New York: Springer-Verlag, 1976.

[2] L. R. Rabiner and R. W. Schafer, *Digital Processing of Speech Signals*, Englewood Cliffs: Prentice Hall, 1973.

[3] B. S. Atal and S. L. Hanauer, "Speech Analysis and Synthesis by Linear Prediction," *J. Acoust. Soc. Amer.* vol.50, pp. 637-655, Aug. 1971.

[4] B. S. Atal, "Predictive Coding of Speech at Low Bit Rates," *IEEE Trans. Commun.* , vol. COM-30, pp. 500-614, April 1982.

[5] R. Viswanathan and J. Makhoul, "Quantization Properties of Transmission Parameters in Linear Predictive Systems," *IEEE Trans. Acoust. Speech, Signal Processing*, vol. ASSP-23, pp. 309-321, June 1975.

[6] D. Y. Wong and B. H. Juang, "Voice Coding at 800 BPS and Lower Data Rates with LPC Vector Quantization," *Proc. Int. Conf. on Acoustics, Speech and Signal Processing*. Paris, France, 1982, pp. 606-609.

2.6