

AD-A125 360

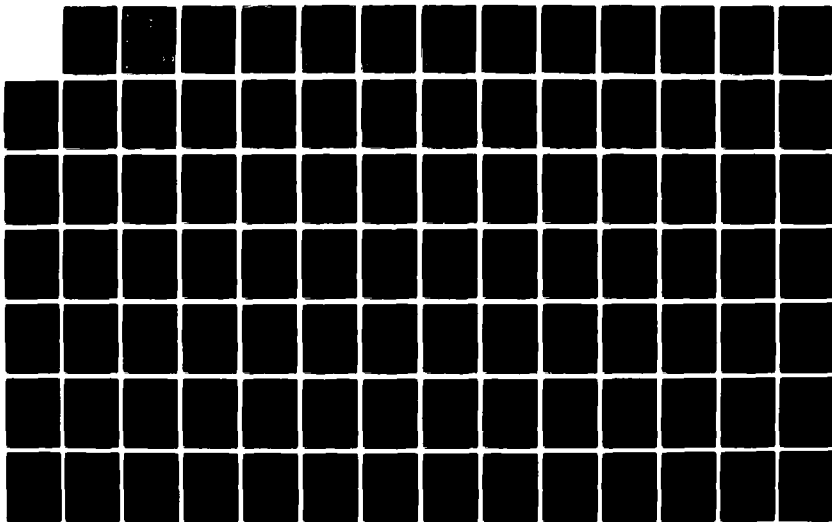
COMPUTER RECOGNITION OF PHONETS IN SPEECH(U) AIR FORCE  
INST OF TECH WRIGHT-PATTERSON AFB OH SCHOOL OF  
ENGINEERING D L MARTIN DEC 82 AFIT/GE/EE/82D-46

1/3

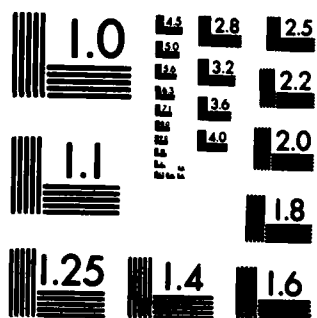
UNCLASSIFIED

F/G 9/2

NL



M-2.



MICROCOPY RESOLUTION TEST CHART  
NATIONAL BUREAU OF STANDARDS-1963-A

AD A 1 25360



Computer Recognition  
of  
Phonets in Speech

Thesis

AFIT/GE/EE/82D-46 Dan Martin  
Captain, USAF

This document has been approved  
for public release and sale; its  
distribution is unlimited.

DEPARTMENT OF THE AIR FORCE  
AIR UNIVERSITY (ATC)

**AIR FORCE INSTITUTE OF TECHNOLOGY**

Wright-Patterson Air Force Base, Ohio

DTIC  
ELECTE  
FEB 24 1983

S D  
E

DTIC FILE COPY

02 023 114

Unclassified

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

| REPORT DOCUMENTATION PAGE                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |                                    | READ INSTRUCTIONS<br>BEFORE COMPLETING FORM                 |
|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|------------------------------------|-------------------------------------------------------------|
| 1. REPORT NUMBER<br>AFIT/GE/EE/82D-46                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    | 2. GOVT ACCESSION NO.<br>AD-A12536 | 3. RECIPIENT'S CATALOG NUMBER                               |
| 4. TITLE (and Subtitle)<br>Computer Recognition of Phonets In Speech                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |                                    | 5. TYPE OF REPORT & PERIOD COVERED<br>MS Thesis             |
| 7. AUTHOR(s)<br>Daniel L. Martin, Captain, USAF                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |                                    | 6. PERFORMING ORG. REPORT NUMBER                            |
| 9. PERFORMING ORGANIZATION NAME AND ADDRESS<br>Air Force Institute of Technology (AFIT/EN)<br>Wright-Patterson AFB OH 45433                                                                                                                                                                                                                                                                                                                                                                                                                                                              |                                    | 8. CONTRACT OR GRANT NUMBER(s)                              |
| 11. CONTROLLING OFFICE NAME AND ADDRESS                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |                                    | 10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS |
| 14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office)                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |                                    | 12. REPORT DATE<br>December 1982                            |
|                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |                                    | 13. NUMBER OF PAGES<br>226                                  |
|                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |                                    | 15. SECURITY CLASS. (of this report)<br>Unclassified        |
|                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |                                    | 15a. DECLASSIFICATION/DOWNGRADING SCHEDULE                  |
| 16. DISTRIBUTION STATEMENT (of this Report)<br><br>Approved for public release; distribution unlimited.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |                                    |                                                             |
| 17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |                                    |                                                             |
| 18. SUPPLEMENTARY NOTES<br><br>Approved for public release: LAW AFB 100-17.<br>Lynn E. WOLVER<br>Dean for Research and Professional Development<br>Air Force Institute of Technology (ATC)<br>Wright-Patterson AFB OH 45433<br><br>4 JAN 1983                                                                                                                                                                                                                                                                                                                                            |                                    |                                                             |
| 19. KEY WORDS (Continue on reverse side if necessary and identify by block number)<br>- phoneme recognition by prototype matching<br>- speech synthesis by phoneme sounds<br>- phoneme recognition by average magnitude differencing<br>- continuous speech recognition<br>(continued on reverse side)                                                                                                                                                                                                                                                                                   |                                    |                                                             |
| 20. ABSTRACT (Continue on reverse side if necessary and identify by block number)<br>→ This project generated phonetic units, termed "phonets," from digitized speech files. The time file was converted to feature space using the Fourier Transform, and phonet occurrences were detected using Minkowski One and Two distance measures. Phonet matches were detected and ranked for each phonet compared against a template file. Phonet short-time energy was included in the output files. An algorithm was developed to partition feature space and its performance was evaluated. |                                    |                                                             |

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

Unclassified

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

19. (Cont'd)

- phoneme recognition by spectrogram matching
- analysis of phoneme sounds in continuous speech

Unclassified

SECURITY CLASSIFICATION OF THIS PAGE(When Data Entered)

GE/EE/82D-46

Computer Recognition  
of  
Phonets in Speech

Thesis

AFIT/GE/EE/82D-46 Dan Martin  
Captain, USAF

Approved for public release; distribution unlimited.

DTIC  
ELECTE  
FEB 24 1983  
E

GE/EE/82D-46

**Computer Recognition of Phonets  
In Speech**

**Thesis**

**Presented to the Faculty of the School of Engineering  
of the Air Force Institute of Technology**

**Air University**

**In Partial Fulfillment of the  
Requirements for the Degree of  
Master of Science**



|                    |                                     |
|--------------------|-------------------------------------|
| Accession For      |                                     |
| NTIS GRA&I         | <input checked="" type="checkbox"/> |
| DTIC TAB           | <input type="checkbox"/>            |
| Unannounced        | <input type="checkbox"/>            |
| Justification      |                                     |
| By _____           |                                     |
| Distribution/      |                                     |
| Availability Codes |                                     |
| Dist               | Avail and/or<br>Special             |
| A                  |                                     |

By

**Daniel L. Martin, B.S.E.E., B.S.M.A.**

**Captain**

**USAF**

**Graduate Electrical Engineering**

**December 1982**

**Approved for public release; distribution unlimited.**

# I

## ACKNOWLEDGEMENTS

I wish to express my appreciation and gratitude to Dr. Mathew Kabrisky, my thesis advisor, for his help and encouragement through the course of this effort. I also want to thank Major Larry Kizer and Dr. Peter Maybeck, members of my thesis committee, for their assistance and constructive criticism.

## PREFACE

The purpose of this project was to generate and detect features in connected speech. It is a part of the larger speech recognition problem.

Phonetic units, which we called observations and phonets, were generated from connected speech. The time file was converted to feature space using the Fourier Transform. Phonet occurrences were detected using Minkowski One and Two distance measures. Phonet matches were ranked in order from minimum to maximum distance between each input phonet and each phonet in the template file.

An algorithm was developed to partition feature space into classes of phonetic units. The central tendency of each class was considered to be a template against which observations were to be compared. The algorithm is adaptive in that a supervisory capability is provided to decide, on the basis of detection results and class variability, if class descriptors should be modified. Algorithm performance in additive, white, gaussian noise was evaluated and decision parameters were related to probability of classification error.

## TABLE OF CONTENTS

|                                                        | <u>Page</u> |
|--------------------------------------------------------|-------------|
| Acknowledgements . . . . .                             | iii         |
| Preface . . . . .                                      | iv          |
| List of Figures . . . . .                              | vi          |
| List of Tables . . . . .                               | viii        |
| Abstract . . . . .                                     | ix          |
| I. Introduction . . . . .                              | 1           |
| Background . . . . .                                   | 1           |
| Scope . . . . .                                        | 4           |
| II. Techniques . . . . .                               | 6           |
| Time-Domain Processing . . . . .                       | 6           |
| Observation Generation . . . . .                       | 8           |
| Phonet Generation . . . . .                            | 17          |
| Distance Computation . . . . .                         | 17          |
| III. Results . . . . .                                 | 19          |
| The Acoustic Analyzer (AA) . . . . .                   | 19          |
| Parameter Variation . . . . .                          | 20          |
| Phoneme Length . . . . .                               | 21          |
| FFT Window Size . . . . .                              | 31          |
| Sampling Rate, Window Shape, and Emphasis . . . . .    | 39          |
| Thresholding . . . . .                                 | 47          |
| Cluster Algorithm . . . . .                            | 92          |
| IV. Conclusions . . . . .                              | 109         |
| V. Recommendations . . . . .                           | 110         |
| VI. Bibliography . . . . .                             | 112         |
| VII. Appendix A: Spectrogram of File CT56.SP . . . . . | 114         |
| VIII. Appendix B: Software Documentation . . . . .     | 116         |
| IX. Vita . . . . .                                     |             |

## LIST OF FIGURES

| <u>Figure</u> | <u>Title</u>                                                                                                                                                       | <u>Page</u> |
|---------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------|
| 1             | Spectrum Emphasis Filter                                                                                                                                           | 9           |
| 2             | Linear System Representation of Spectral Computer in Acoustic Analyzer                                                                                             | 10          |
| 3             | Plot of Spectrum of Tones at 200Hz, 1000Hz and 2500Hz (50 units)                                                                                                   | 12          |
| 4             | Plot of Spectrum of Tones at 200Hz, 1000Hz and 2500Hz (200 units)                                                                                                  | 13          |
| 5             | Plot of Spectrum of Tones at 200Hz, 1000Hz and 2500Hz (50 units)                                                                                                   | 15          |
| 6             | Plot of Spectrum of Tones at 200Hz, 1000Hz and 2500Hz (200 units)                                                                                                  | 16          |
| 7             | Memory Map for DSTA and DSTN                                                                                                                                       | 126         |
| 8-15          | Distance Plots of Observation Numbers 41, 45, 51, 55, 61, 71, 81 and 91                                                                                            | 23-30       |
| 16-22         | Spectral Plots of Observation Numbers 45, 47, 86, 87, 88, 89 and 90                                                                                                | 32-38       |
| 23-28         | Distance Plots of Observation Numbers 85, 90, 115, 120, 125 and 130                                                                                                | 40-45       |
| 29-56         | Spectral Plots of Observation Numbers 21, 24, 31, 34, 70, 75, 80, 85, 87, 88, 90, 95, 100, 110, 120, 130, 135, 140, 145, 150, 160, 170, 180, 190, 200, 201 and 202 | 50-77       |
| 57-68         | Distance Plots of Observation Numbers 21, 31, 81, 91, 101, 131, 171, 181, 191, 195, 200, and 202                                                                   | 80-91       |
| 69            | Model of an Adaptive Recognition Scheme Adapted From Reference 6                                                                                                   | 93          |
| 70            | Model of the Supervisory Detection Process                                                                                                                         | 102         |
| 71            | Structure of the Acoustic Analyzer                                                                                                                                 | 118         |

| <u>Figure</u> | <u>Title</u>                           | <u>Page</u> |
|---------------|----------------------------------------|-------------|
| 72            | Flowchart of DRSQ                      | 131         |
| 73            | Flowchart of S128 and S64              | 148         |
| 74            | Flowchart of DSTA and DSTN             | 165-166     |
| 75            | Flowchart of CHOOS                     | 181         |
| 76            | Example of PLTA                        | 201         |
| 77            | Spectral Plot of Observation Number 45 | 208         |
| 78            | Example of PLTN                        | 214         |
| 79            | Example of PLTT                        |             |

## LIST OF TABLES

| <u>Table</u> |                                                                              | <u>Page</u> |
|--------------|------------------------------------------------------------------------------|-------------|
| 1            | Header Assignments for Spectral File<br>Computed by DRSQ Calling S128 or S64 | 134         |
| 2            | Header Assignments for Test Option                                           | 135         |
| 3            | Header Assignments for Spectral File<br>Computed by DRSQ Calling S128 or S64 | 152         |
| 4            | DSTA Output File                                                             | 162         |
| 5            | DSTN Output File                                                             | 163         |
| 6            | Distance File Header                                                         | 167         |

# ABSTRACT

## "Phonet Detection in Connected Speech"

This project generated phonetic units, termed "phonets," from digitized speech files. The time file was converted to feature space using the Fourier Transform, and phonet occurrences were detected using Minkowski One and Two distance measures. Phonet matches were detected and ranked for each phonet compared against a template file. Phonet short-time energy was included in the output files. An algorithm was developed to partition feature space and its performance was evaluated.

## I. INTRODUCTION.

### BACKGROUND:

This project concerns the generation and detection of features in connected speech. It is a part of a larger effort directed towards connected speech recognition.

We have divided the speech recognition problem into six parts:

- (1) Digitization of the acoustic signal,
- (2) Feature extraction,
- (3) Partitioning of the feature space into classes,
- (4) Feature detection,
- (5) Word recognition from detected features, and
- (6) Concept recognition from recognized words.

Falkey (Ref 2) and Seelandt (Ref 1) have worked on Parts I, II, and IV of the problem. This project extends Seelandt's work on feature detection and offers an algorithm for partitioning feature space into classes. Montgomery (Ref 10) has worked on Part V of this problem and Part VI has yet to be addressed.

In the remainder of this chapter, we mention works in the literature which influenced the development of this project and state its scope.

Felkey was able to generate spectrograms from digitized speech (Ref 2). This capability was extended by Seelandt (Ref 1) into a Speech Sound Analysis Machine (SSAM) which gave an operator the ability to listen to speech segments chosen from a spectrogram displayed on a Tektronix 4010-1 Graphics Terminal. Seelandt was able to use his machine to choose phonetic units which he thought might be detectable in connected speech. He was able to construct template files of these units and then to detect similar units in connected speech. Detection was accomplished by calculating a Minkowski metric of order one (Ref 1 and 9) between each phonetic unit and spectrum of consecutive time slices from the input speech. The template which matched the speech segment in the sense of minimum distance was deemed the detected phoneme.

Felkey and Seelandt both applied the Fourier Transform and extracted features in the spectral domain. Other transformations from measurement space can be applied. DeSouza and Thomson derived likelihood ratio statistics for the case in which two estimated Linear Predictive Coding (LPC) vectors are being compared (Ref 3). They also investigated the distribution and sensitivity of other LPC distance measures.

Kassam has compiled an alphabetical listing of papers from the engineering literature on nonparametric detection theory and application (Ref 4). Nonparametric detectors

12

are based on statistical hypothesis testing principles for situations where parametric statistical models cannot be specified for the observation under the null hypothesis. The usual performance characteristic is minimum Type I error probability or false-alarm rate. Kassam also states that robust procedures, where not only the false-alarm probability but also the power of the test is considered in defining a performance criterion, may be applicable to situations where parametric models cannot be assumed.

Lainiotis (Ref 5) has studied adaptive systems for detection and pattern recognition as well as for feature extraction. He states that for the supervised learning case, optimal adaptive systems are realizable in a partitioned form which consists of a linear, nonadaptive part made up of a bank of Kalman filters, and a nonlinear, adaptive part made up of probability computers. He offers relatively tight upper and lower bounds to the probability of error in the feature extraction problem in terms of functionals such as the Bhattacharya coefficient.

Work done at Johns Hopkins University points out that the Fast Fourier Transform (FFT) is an extremely robust procedure for large time-band width signals (Ref 7). The report comments that the FFT is robust enough to give good performance on an error probability basis on random input time series with probability distributions so extreme as to be unlikely to occur in nature. The price for this

13

robustness is reduced power; that is, the probability of making a correct binary decision conditioned on knowledge of the signal sent, according to the report.

We decided to continue feature extraction in the spectral domain for the following reasons:

(1) We could build directly on the work already accomplished by Seelandt and Felkey.

(2) We could apply the already available Eclipse AP/130 Array Processor to the task of transforming the speech time series to the spectral domain.

(3) While no one transformation into feature space that we found documented in the literature was clearly a best choice for our purpose, we deemed it wise to sacrifice power for the robustness of the FFT.

#### SCOPE:

The goals of this project are:

(1) To generate phonetic units, which we term phonets or observations, from speech files placed on disk.

(2) To detect occurrence of members of one set of phonetic units, referred to as phonets, in a contiguous file of observations generated from speech.

(3) To apply the Eclipse AP/130 Array Processor to tasks one and two for speed.

(4) To build a sufficient degree of flexibility into the phonet generator and distance computer to allow

14  
factors, such as FFT window point size and type, filtering, and distance rule, to be varied and their effect investigated.

(5) To identify a means for generating an adequate phonet set for our purposes.  
Tasks one through four are accomplished by development of the Acoustic Analyzer: an interactive software package which accomplishes those tasks. Task five is accomplished by formulation of an algorithm for partitioning feature space into template classes. Additionally, some characteristics of phonets and their interaction with observations are illustrated.

This work was not concerned with acquiring speech or generating time files containing speech. It takes the techniques Seelandt (Ref 1:9-12) used as given and relies on speech files resident on disk.

15

## II. TECHNIQUES.

### TIME DOMAIN PROCESSING:

In this chapter, we discuss techniques used to implement the Acoustic Analyzer. In this section, we discuss time domain processing of the digitized speech. In the following sections, we describe the generation of what we call observations and phonets: spectral domain representations of the fundamental phonetic units, and the distance computation between observations and phonets.

Speech is digitized at an 8KHz sampling rate and stored on disk in the manner described by Seelandt (Ref 1: 9-12) before being input to the Acoustic Analyzer. Given this input file, the Acoustic Analyzer converts speech to observations and then compares them to a template file of phonets. Time domain processing is accomplished in the segment of the Acoustic Analyzer which converts speech to observations and consists of the application of a window to the time file. The operator has control of the following window parameters, which are explained below:

- (1) Window type.
- (2) Window size.
- (3) Overlap of the 128 point window.

Two types of windows are available: a Hamming and a rectangular window. The Hamming window option was included because: (1) Seelandt used it, and (2) spectral distortion caused by the rectangular window is an undesired complication. It was desirable to include the capability to select a number of different window sizes to provide flexibility in future investigations. It is clear that a choice of window size involves a trade-off between spectral resolution and time resolution (Ref 8:260). It appears that several window sizes will need to be tried and one best chosen on the basis of recognition results. Seelandt used a 64 point window at a sample rate of 8KHz for time slices 8MSEC long (Ref 1:120). Rabiner and Schafer discuss the window effects with regard to using time slice energy as an indicator of voiced or unvoiced speech (Ref 8:122). They recommend using a window duration of from 10-20MSEC. At the 8KHz sampling rate used by Seelandt to digitize the speech files (Ref 1:10), a 10-20MSEC window is from 80-160 points wide. Also, Rabiner and Schafer show that stops and other short-lived temporal features of speech last roughly 10MSEC (Ref 8:38-60). For these two reasons, we consider the 128 point window as our primary choice, or point-of-departure for further investigations into the question of optimum window size. Because of concern that a Hamming window shape coupled with 16MSEC window duration

might miss some temporal detail, we included the capability to cause the window to increment along the speech file in 64 point increments overlapping by 64 points. This option is not available for the other window sizes and does not significantly add to processing time.

#### OBSERVATION GENERATION:

An observation is, basically, the magnitude of the Fast Fourier Transform (FFT) of a segment of an input speech file. Spectral shaping, described later, is applied to the spectral components. It is this shaped spectrum that is compared to the phonets in the template file; a phonet being an observation assigned to the template file.

Short-time energy of the speech signal is a parameter that reflects amplitude variations between unvoiced and voiced speech segments (Ref 8:120). Rabiner and Schafer define the short-time energy as:

$$E_n = \sum_{m=-\infty}^{\infty} [x(m) w(n-m)]^2 \quad (1)$$

with:

$E_n$  = short-time energy.

$\{x(m)\}$  = speech time sequence.

$\{w(n-m)\}$  = window time sequence of finite length.

They define the time dependent Fourier Transform as (Ref 8:251):

$$X_n(e^{j\omega}) = \sum_{m=-\infty}^{\infty} w(n-m) x(m) e^{-j\omega m} \quad (2)$$

with:

$$\{X_n(e^{j\omega})\} = \text{time dependent Fourier Transform.}$$

Our Acoustic Analyzer computes a value which we call the observation energy, defined as:

$$E_n^1 = \sum_{m=-\infty}^{\infty} |X_m(e^{j\omega}) p(m)|^2 \quad (3)$$

where:

$E_n^1$  = observation energy.

$\{p(m)\}$  = spectral emphasis sequence.

The emphasis sequence,  $\{p(m)\}$ , is equivalent to a linear, shift invariant filter with a spectral response shown in Figure 1.

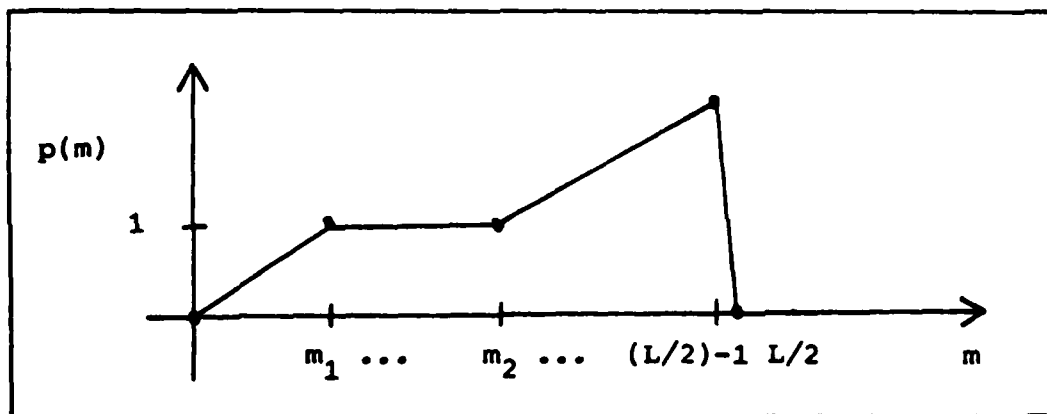


FIGURE 1. Spectrum Emphasis Filter

For an  $L$ - point window,  $p(m) = 0$  for  $m = 0$  and for  $m = L/2$ . Low frequency components are de-emphasized at a

selectable rate up to a selectable corner frequency,  $m_1$ . High frequency components are emphasized at a selectable rate and corner frequency,  $m_2$ . Emphasis rates are selectable in units of dB/Octave. Note that the zero and L/2 points are zero. Assuming negligible energy in the speech signal at 4KHz, and assuming a DC block, the observations computed by our Acoustic Analyzer are the output of the linear system as shown in Figure 2.

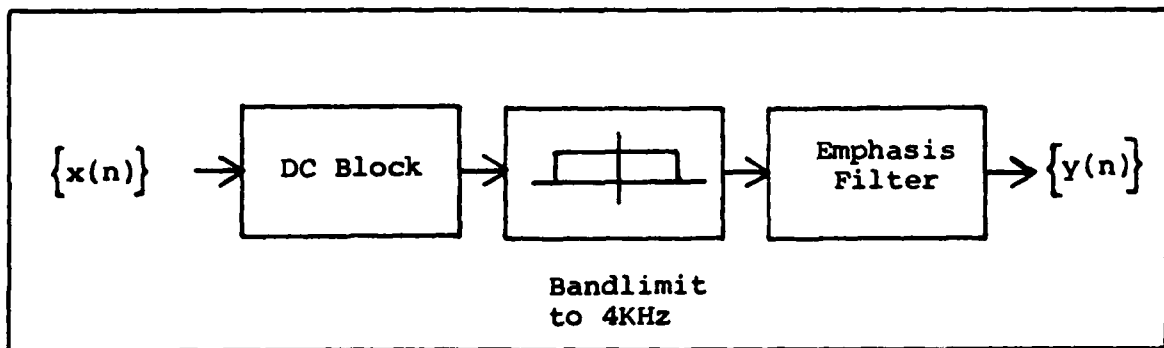


FIGURE 2. Linear System Representation of Spectral Computer In Acoustic Analyzer.  $\{X(n)\}$  is the speech input.

By Parseval's theorem (Ref 9:36), the energy in the output sequence,  $E_n^1$ , is related to the energy in the input speech,  $E_n$ , under assumptions that there is: (1) no DC value to the speech input, and (2) no speech energy at or above 4KHz. Hence, the observation energy,  $E_n^1$ , computed by the Acoustic Analyzer, can be used as an indication of voiced/unvoiced speech in the same manner as the short-time energy. This value can be used by a word recognition

machine to aid in that process. The output file at this point contains, for each observation, the observation energy as the first component and spectral components one through  $L/2 - 1$  in the second through  $L/2$  component positions.

For the same reason Seelandt did (Ref 1:121), we normalized the energy in each observation to a constant value. This provided a measure of automatic gain control, in that if the only difference between two speech segments was their energy content, the components in observations produced from them would be of approximately the same amplitude. Thus, variability caused by different speech amplitudes would be reduced. To normalize the observation energy, we scaled each spectral component by the factor  $10^4 / \sqrt{E_n^1}$ .

We observed the effectiveness of obtaining automatic gain control through energy normalizing by including the capability to scale the spectral components by the factor  $10^4 / E_n^1$ , rather than energy normalizing. We generated two time files containing tones at 200Hz, 1KHz, and 2.5KHz, each at an amplitude of 50 units in File A and 200 units in File B. Figure 3 is an observation from File A and Figure 4 is from File B; both energy normalized. In these and the next two figures, the value plotted in position zero along the horizontal axis is the energy value,  $\sqrt{E_n^1}$ . One can see that the amplitudes of the spectral components

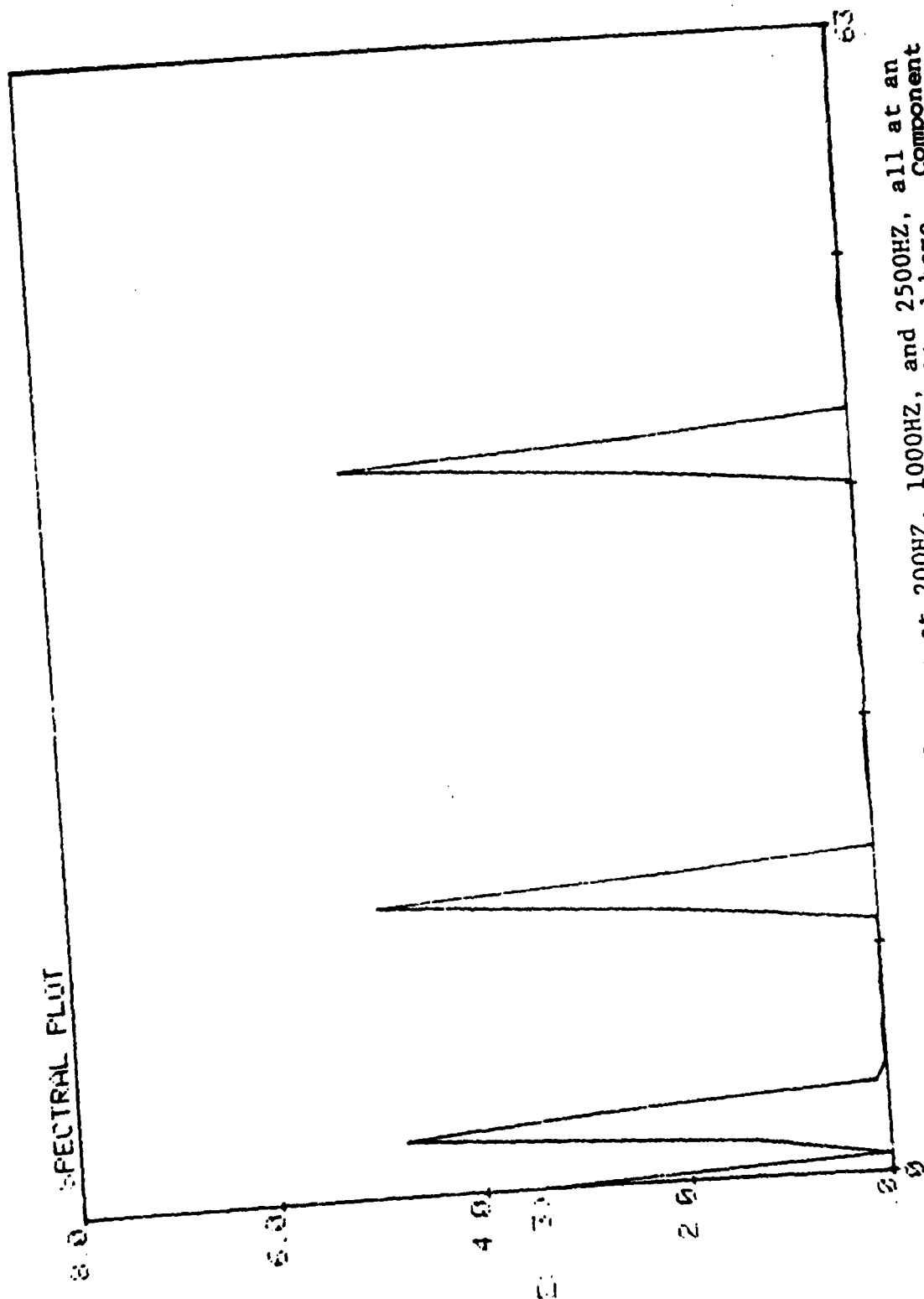


Figure 3 Plot of spectrum of tones at 200HZ, 1000HZ, and 2500HZ, all at an amplitude of 50 units. Observation is normalized here. Component zero is  $\sqrt{E_n}$ , where  $E_n$  is observation energy.

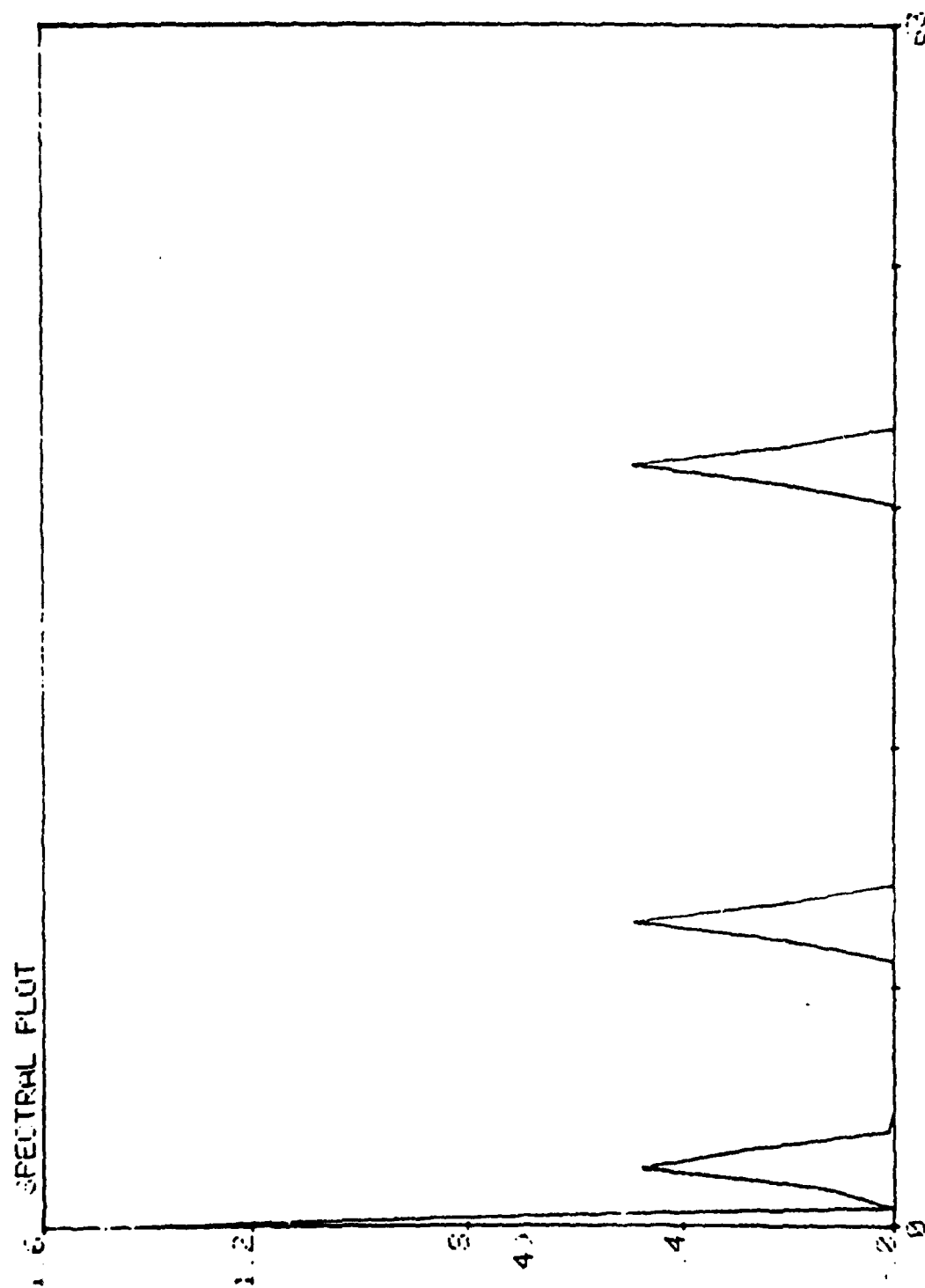


Figure 4 Plot of spectrum of tones at 200HZ, 1000HZ, and 2500HZ; all at an amplitude of 200 units. Observation is normalized here. Component zero is  $\sqrt{E_n^1}$ , where  $E_n^1$  is observation energy.

are nearly the same in the two observations. The amplitudes of the spectral components differ markedly between the next two figures which were scaled by the factor  $10^4/E_n$ .

Figure 5 was computed from File A which contained tones at an amplitude of 200 units and Figure 6 was computed from File B which contained tones at an amplitude of 50 units. One can also see that energy normalization removes from the observation spectrum all of the variability caused by speech amplitude differences.

Energy normalization has, at least, one other consequence: it limits the maximum Euclidean distance between any two observations. To see this, suppose the observations have been energy normalized by having their components scaled by the factor  $K/\sqrt{E}$ , where  $E$  is the energy in observation being normalized. We write each observation as a vector and note that the energy in the difference between any two observations is:

$$|\underline{x}_1 - \underline{x}_2|^2 \leq |\underline{x}_1|^2 + |\underline{x}_2|^2 \quad (4)$$

Now the energy in each observation is  $K^2$  so the Euclidean distance is:

$$|\underline{x}_1 - \underline{x}_2| \leq \sqrt{2} K \quad (5)$$

So the Euclidean distance is upper bounded by  $\sqrt{2}$  times the scale factor constant,  $K$ .

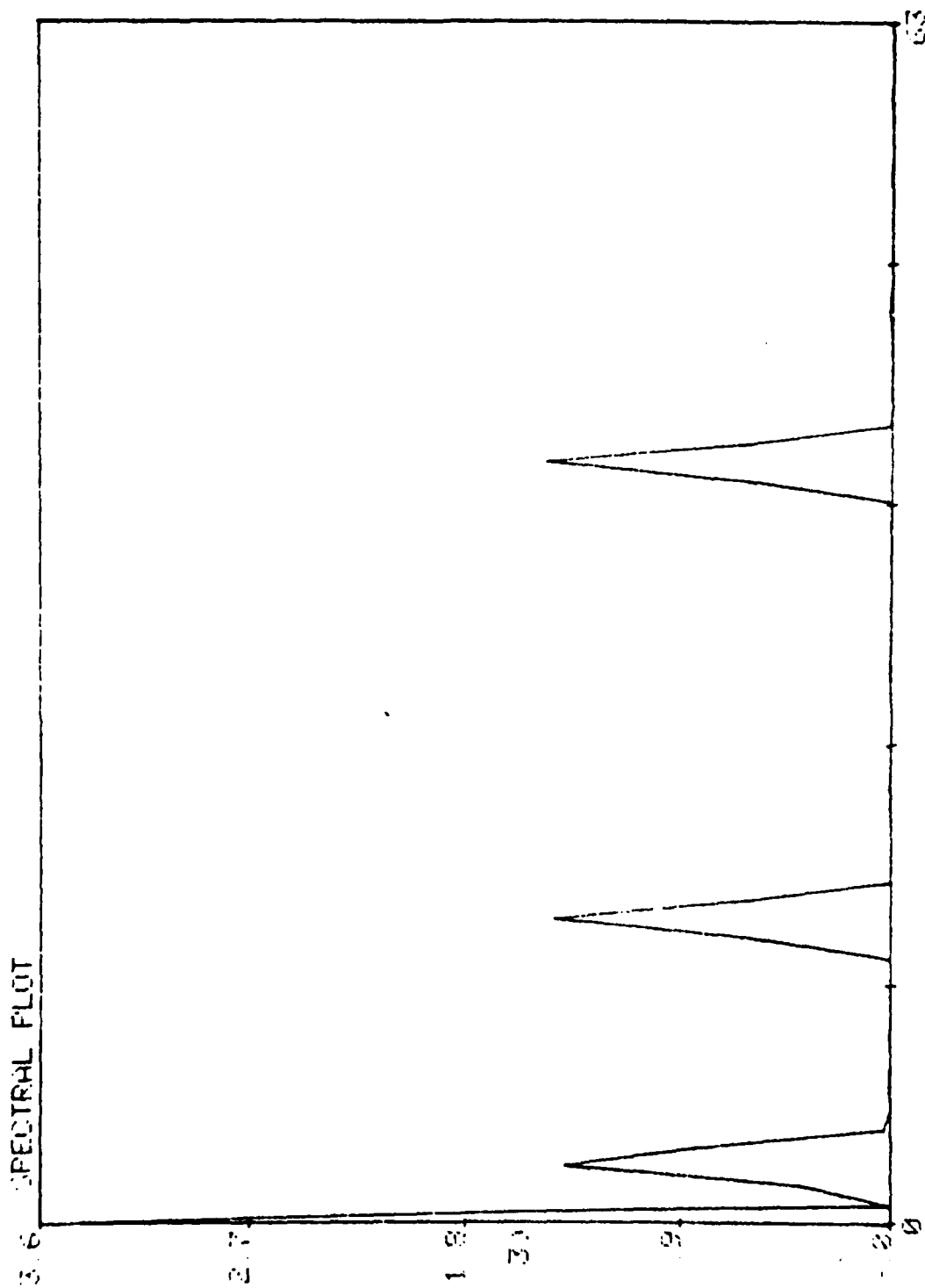


Figure 5 Plot of spectrum of tones at 200HZ, 1000HZ, and 2500HZ, all at an amplitude of 50 units. Observation is divided by its energy. Component zero is  $\sqrt{E_1}$ , where  $E_1$  is observation energy.

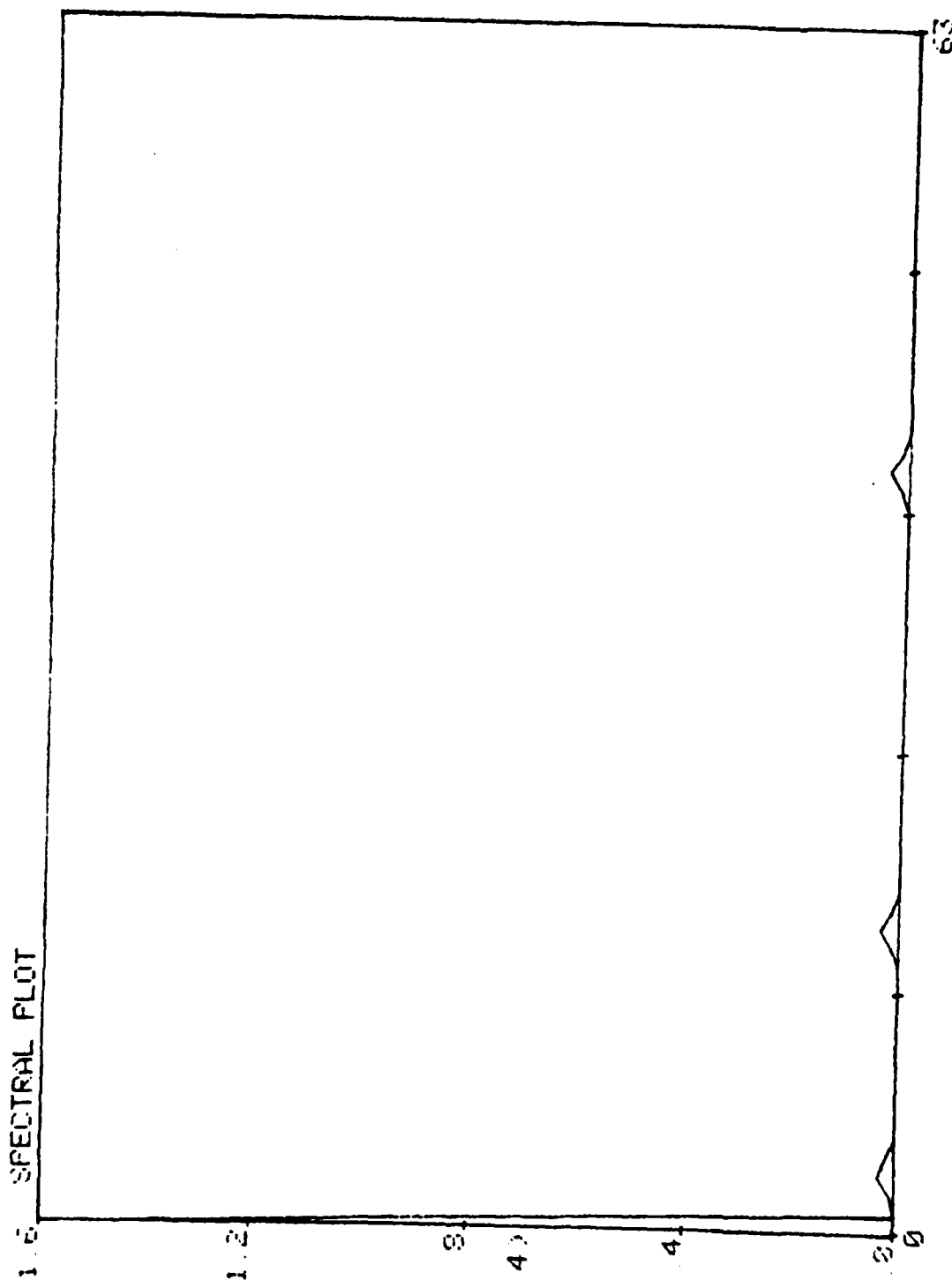


Figure 6 Plot of spectrum of tones at 200HZ, 1000HZ, and 2500HZ, all at an amplitude of 200 units. Observation is divided by its energy. Component zero is  $\sqrt{E_1}$ , where  $E_1$  is observation energy.

#### PHONET GENERATION:

Phonets are selected from a set of observations computed by the Acoustic Analyzer (AA) using the spectrum option. We presently have no systematic means for selecting an adequate set of phonets. Such a set should span the observation space in that any observation can be seen to resemble at least one phonet in a detectable way. It should also be discriminatory in that any observation should resemble only one phonet better than all others to a detectable degree. One last property that an adequate phonet set should have is that of compactness, in that the number of phonets should not be so great as to impose prohibitive computational burdens. Given any phonet set, determining if it has these three basic properties, requires consideration of the distance rule to be used.

#### DISTANCE COMPUTATION:

Seelandt used the Minkowski One (M1) distance and a smallest distance determination (Ref 1:53-57) to choose five closest phonetic units to each observation run through his distance computer. He used this measure because it was convenient to compute. Our Acoustic Analyzer includes this distance measure as one option. We also included the Euclidean, or Minkowski Two (M2) distance (Ref 9:232). One option of our Acoustic Analyzer is a distance computer which uses either the M1 or M2 measure to calculate a

symmetric distance matrix,  $\varphi$ . The element  $\varphi(i, j)$  is the distance between the  $i^{\text{th}}$  observation and the  $j^{\text{th}}$  phonet. The other distance option in our AA uses either distance measure to choose a selectable number of closest phonets to each observation. In the file for each observation is the observation number as the first element, the observation energy as the second element, an ordered pair of (phonet number, corresponding maximum distance), in the next two elements and the selected number of ordered pairs (phonet number, corresponding minimum distance). These last ordered pairs are themselves ordered minimum distance choice-to-maximum. Files for consecutive observations are contiguously arranged in the output file.

We chose to implement the M2 distance measure because it is the Euclidean distance in a geometrical sense in the absence of noise. We implemented the M1 rule to provide the capability to compare results obtained using the Acoustic Analyzer with those using Seelandt's M1 implementation. It is acknowledged that other distance measures and more elaborate decision rules will be needed to achieve a specified error probability in noise. This implementation assumes a completely deterministic system without noise of any kind. Noise is introduced into the analysis of the feature space cluster algorithm described in the next chapter.

### III. RESULTS.

#### THE ACOUSTIC ANALYZER (AA):

One objective of this project was to implement a machine like Seelandt's (Ref 1) Speech Sound Analysis Machine (SSAM) on the Eclipse AP/130 Array Processor, taking advantage of its fast computational capability. This work resulted in delivery of a fast, flexible software package with the required features:

- (1) A flexible spectral computing option which transforms time slices of speech into observations.
- (2) A flexible distance computer which calculates a symmetric distance matrix between each observation and phonet.
- (3) Another distance computer which chooses a selectable number of best matches from a distance matrix.
- (4) A graphics capability which allows one to view plots of speech files, observation files, phonet files, and distance files.

The ability to select parameters such as those which define the FFT time window, define processing to be done on the transformed spectral units, and characterize the detection scheme, has been included in our Acoustic Analyzer. We chose to implement a 128 point, overlapped FFT window.

In the discussion on observation generation, we saw that normalizing the energy in each observation removed much of the variability from speech amplitude variation. In the next section, we will illustrate effects of other parameter selections.

#### PARAMETER VARIATION:

In this section, we illustrate observations and distances computed using several parameter settings. We use only one speech file, CT56.OB, which is an adult male saying the words "five" and "six." It is beyond the scope of this project to illustrate parameter variation effects for a large number of speech files; but, there is reason to believe that such effects would be similar to those illustrated here.

Seelandt (Ref 1) identified the following parameters as requiring study in order to determine optimum settings:

(1) The number of consecutive time slices grouped together to represent a phoneme. Seelandt's phonetic units were five time slices long.

(2) The FFT window size. Seelandt used a 64 point window.

(3) Rate at which the speech time signal is sampled when digitized. Seelandt used speech sampled at 8KHz.

(4) Window shape: rectangular, Hamming, or other. Seelandt used the Hamming window.

(5) Spectral shaping: pre-emphasis to emphasize frequency components above a corner frequency, and de-emphasis to attenuate frequency components below a corner frequency.

(6) Spectral domain thresholding. Seelandt normalized the energy in the spectrum in each time slice when that energy was above a threshold. He attenuated the spectrum in time slices which contained less energy than the threshold.

#### PHONEME LENGTH:

We decided to treat phonetic units one time slice long, rather than form five time slice phonemes, for the following reasons. First, it is more convenient to process single unit phonets than multiple unit phonemes. Second, the dynamic behavior of speech segments would be visible with more resolution in a plot of distance-versus-phonet number. Third, until the dynamic behavior of adjacent short-time speech units is better characterized, we believed that it would not be prudent to average their effect in a distance measure. We reasoned that it would be better to preserve the time slice-by-time slice variation. Then when short-time speech dynamics are better understood, one can simply reformat spectral and distance files generated by our Acoustic Analyzer to accommodate any desired phoneme length.

Computing distances on single time slice phonets, we found that adjacent time slices were similar in the midst of a vowel and not similar in unvoiced fricatives. From a spectrogram of file CT56.OB, we knew that the speaker was uttering the word "five" during time slices 40-80 and the "si" part of "six" during time slices 90-100. The spectrogram is provided in Appendix A. We used a 128 point, non-overlapped Hamming window with 10dB/Octave de-emphasis ending at 300Hz and 10dB/Octave pre-emphasis starting at 500Hz. We computed the M2 distance between the file and itself for time slices 41-102; that is, we ran observations 41-102 against phonets 41-102 of the same file. Plots of observations 41, 51, 61, 71, 81, and 91 against phonets 41-102 are shown in Figures 8, 10, 12, 13, 14, and 15, respectively. The similarity of adjacent observations in the vowel regions in Figures 10 and 12 are apparent, as is the dissimilarity of adjacent observations elsewhere. This effect is illustrated on an expanded scale in Figures 9 and 11, which show distances between the observations 45 and 55, respectively, plotted against phonets 41-70.

We have seen this effect every time we have looked for it. Adjacent time slices resemble each other in both M2 and M1 space much more in regions of vowels when a spectrogram shows clear formant structure than in regions where a spectrogram shows little formant structure. This formant structure is visible in plots of the observations.

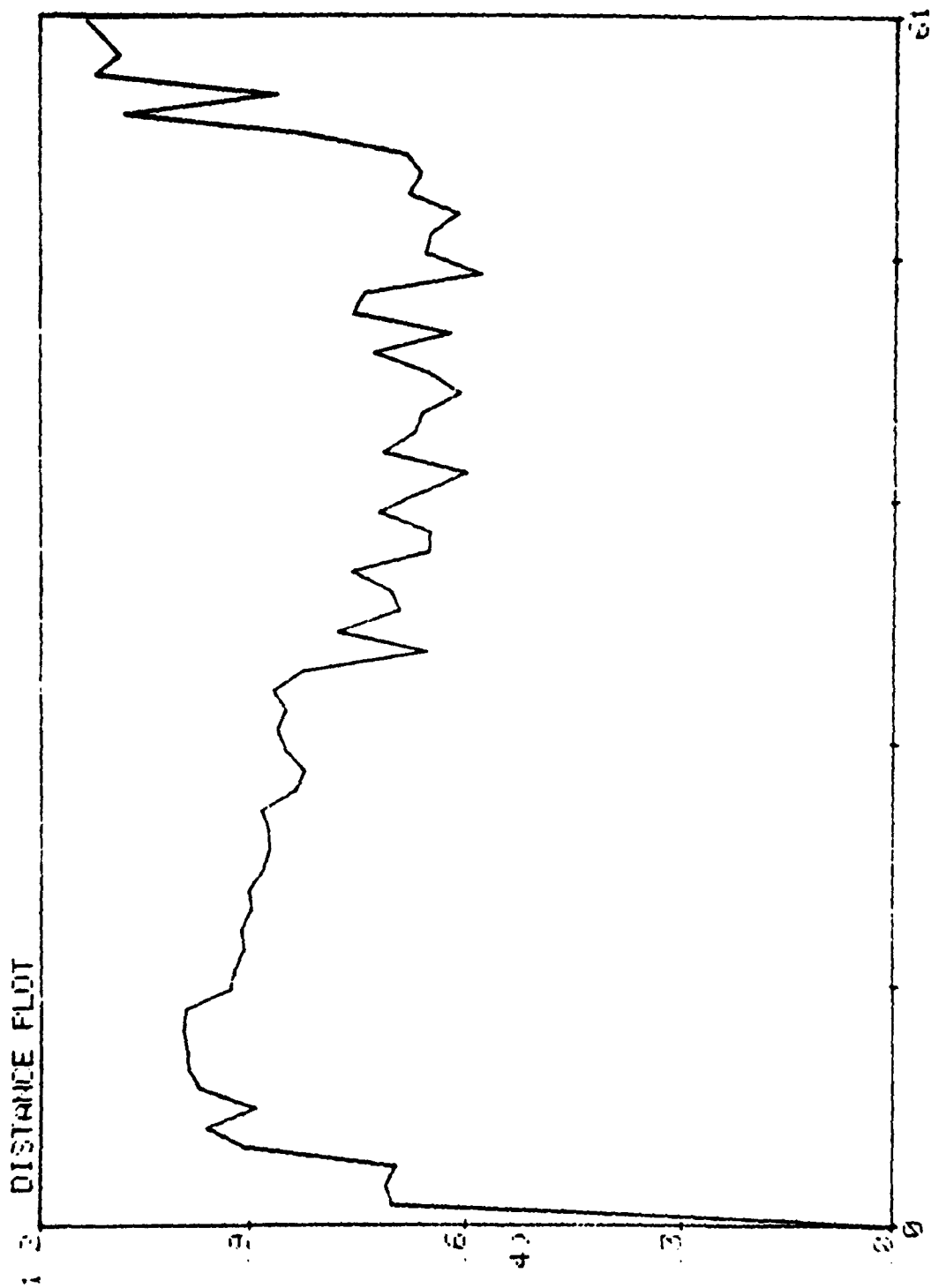


FIGURE 8. Distance plot of observation number 41 against 41 through 102. M2 rule, 128 point, nonoverlapped hamming window, energy normalized.

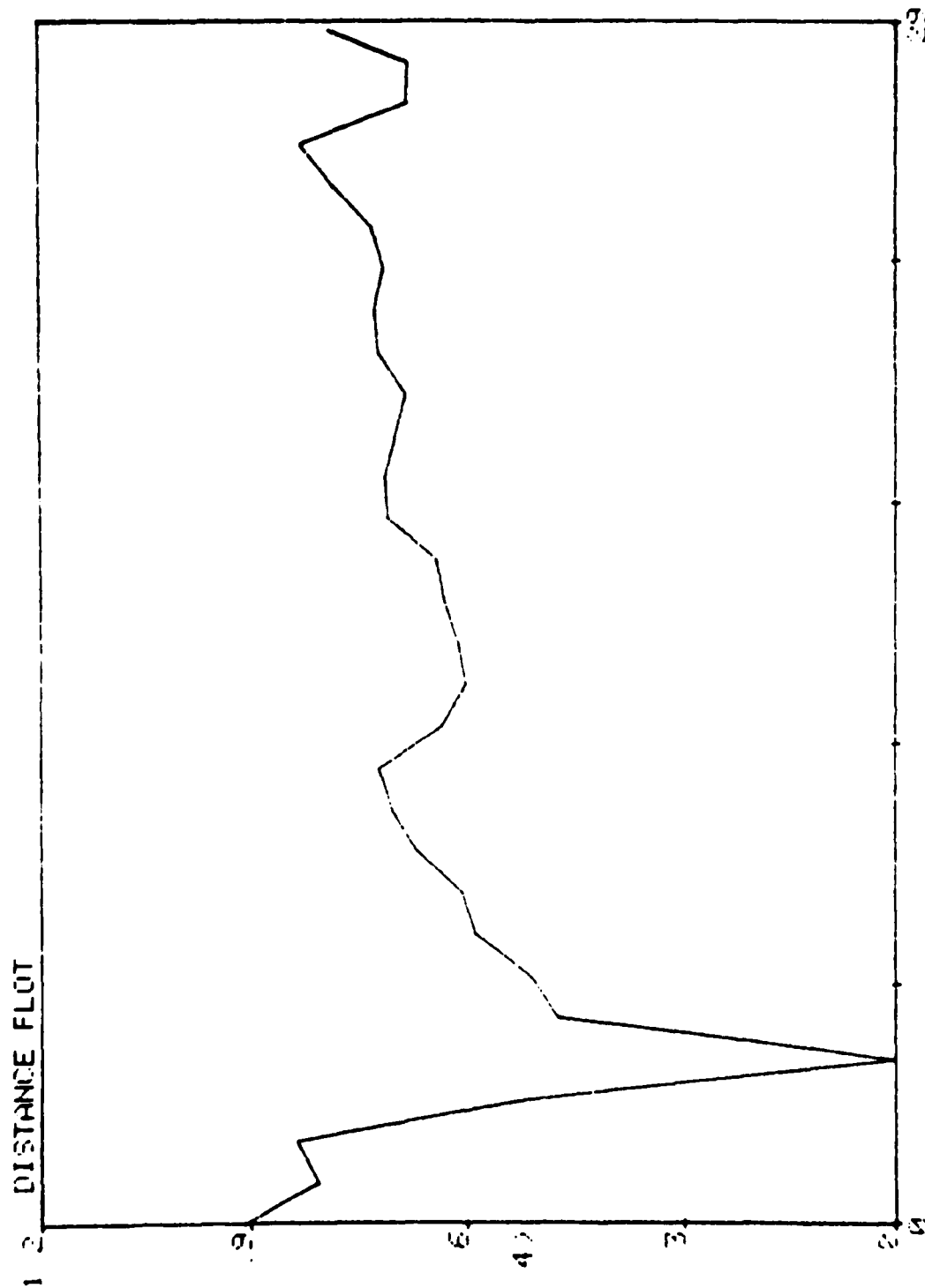


FIGURE 9. Distance plot of observation number 45 against 41 through 70. M2 rule, 128 point, nonoverlapped Hamming window, energy normalized.

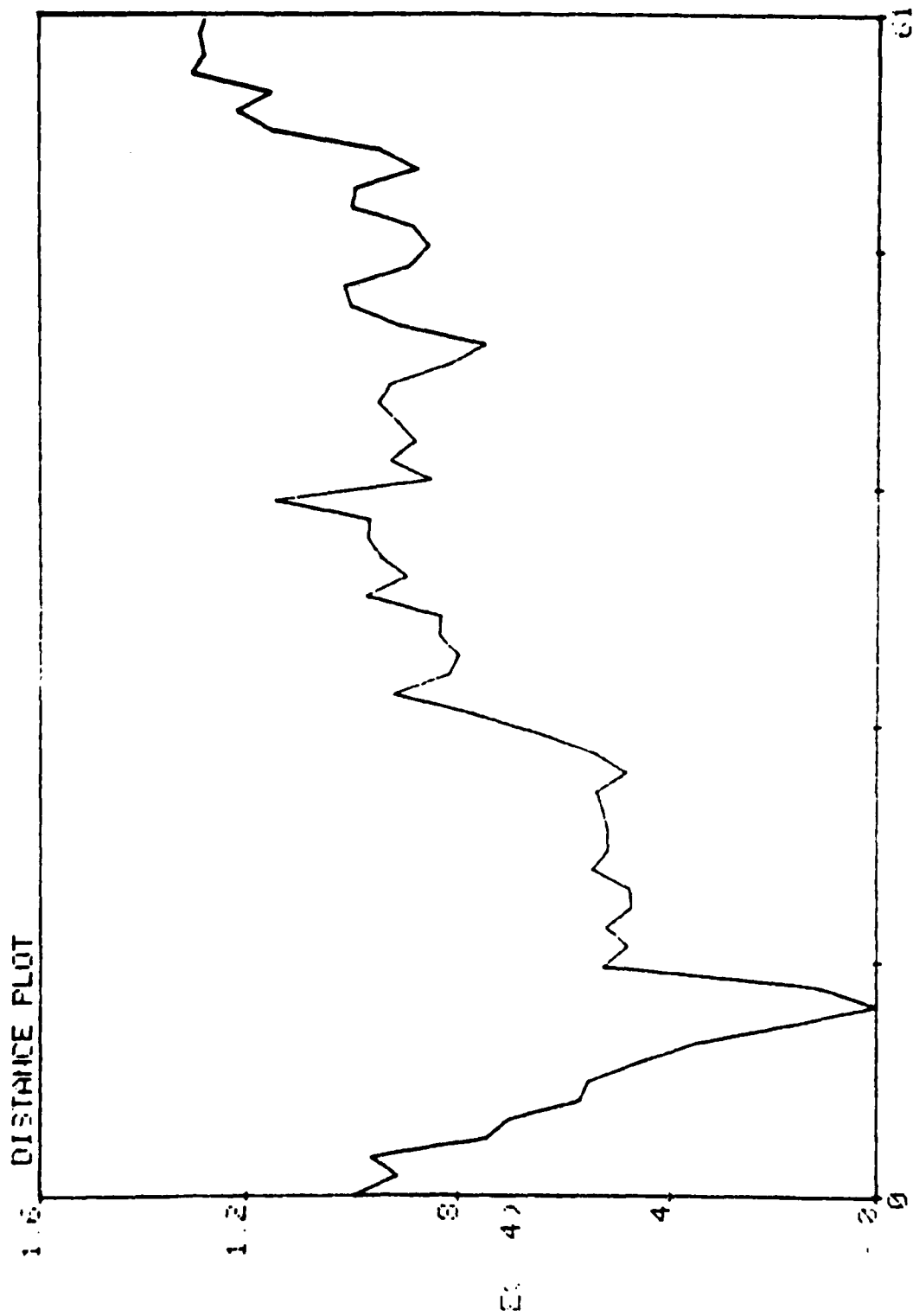


FIGURE 10. Distance plot of observation number 51 against 41 through 102. M2 rule, 128 point, nonoverlapped Hamming window, energy normalized.

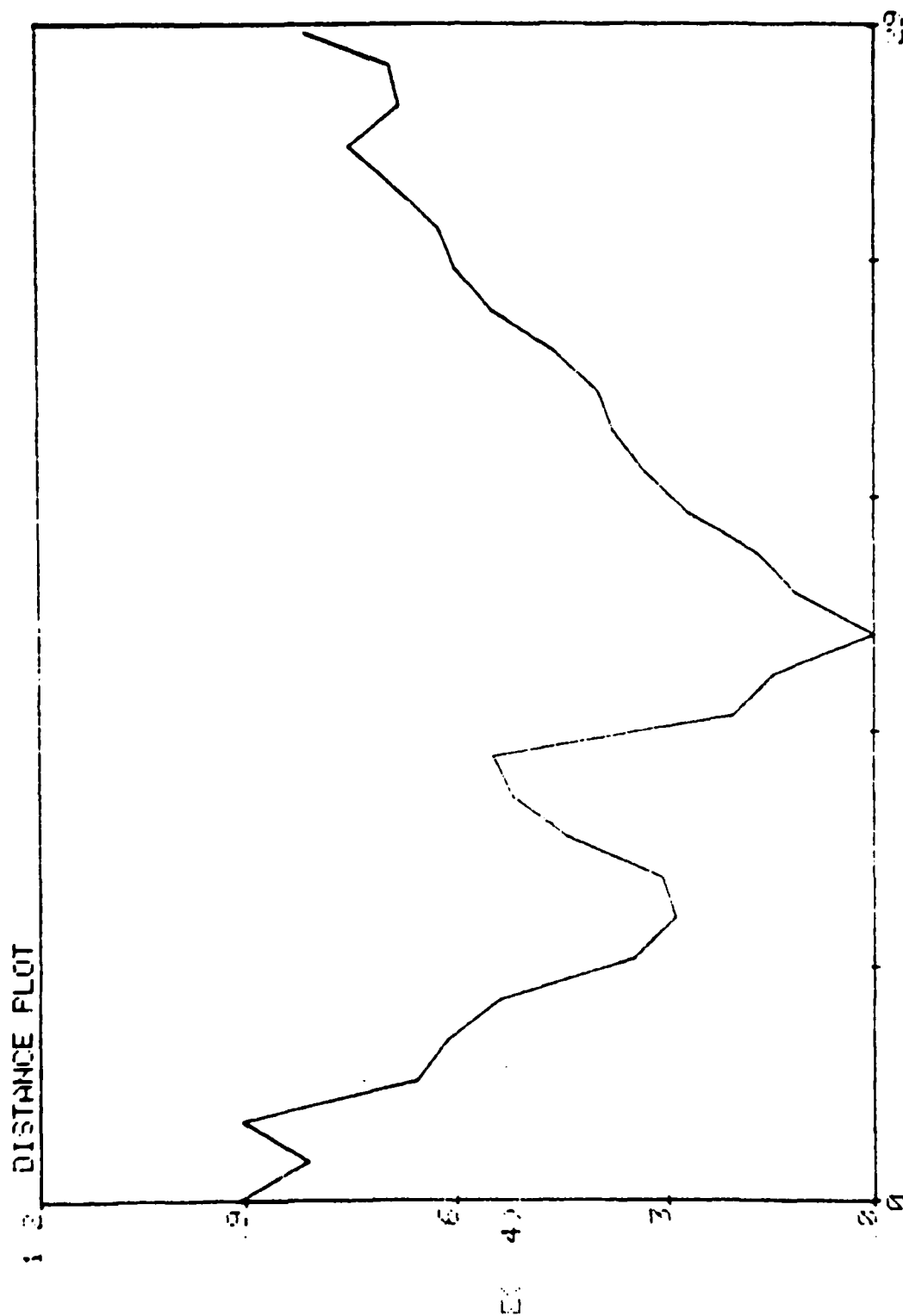


FIGURE 11. Distance plot of observation 55 against 41 through 70. M2 rule, 128 point, nonoverlapped Hamming window, energy normalized.

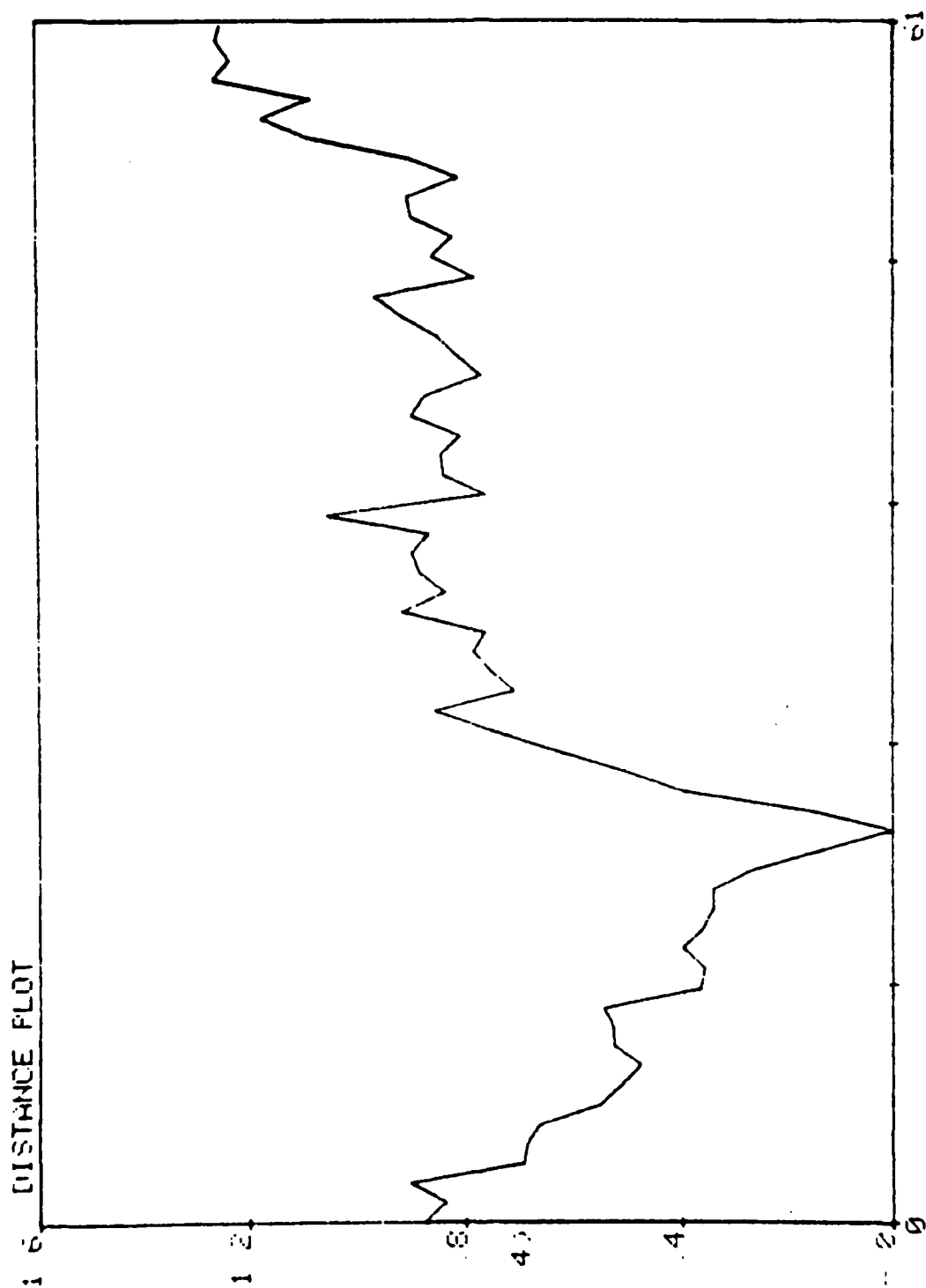


FIGURE 12. Distance plot of observation 61 against 41 through 102.  
M2 rule, 128 point, nonoverlapped Hamming window, energy  
normalized.

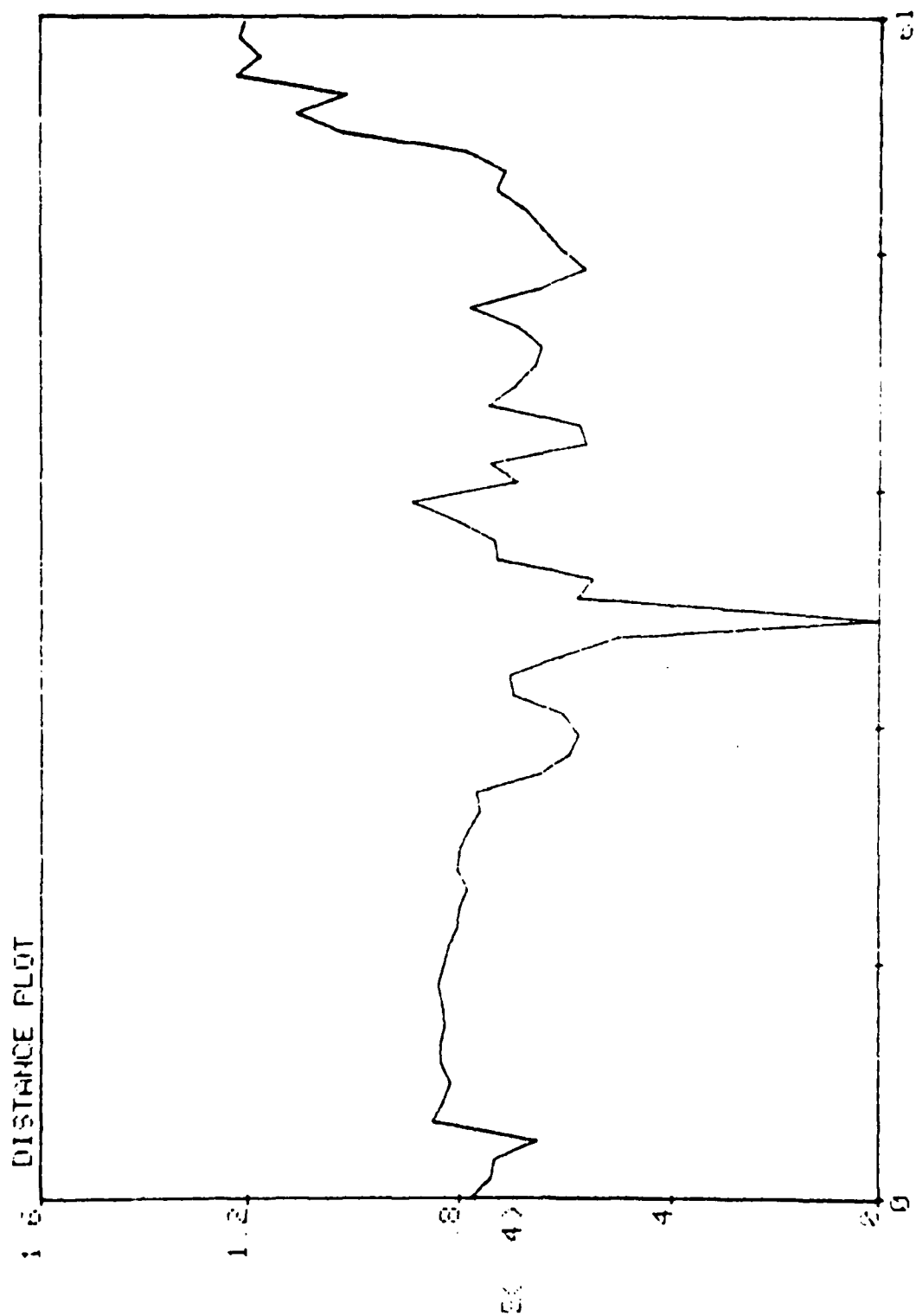


FIGURE 13. Distance plot of observation 71 against 41 through 102. M2 rule, 128 point, nonoverlapped Hamming window, energy normalized.

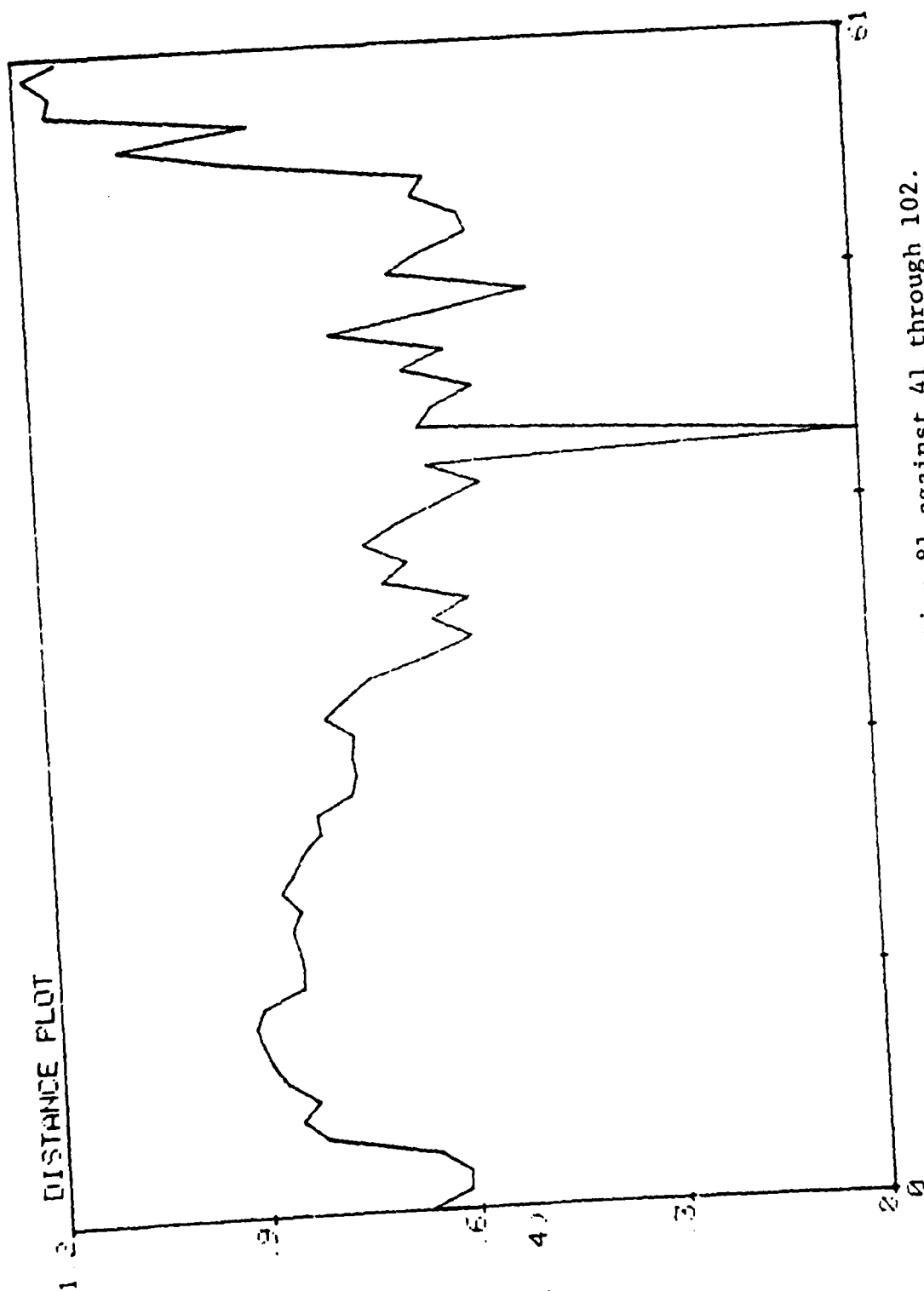


FIGURE 14. Distance plot of observation 81 against 41 through 102.  
M2 rule, 128 point, nonoverlapped Hamming window, energy  
normalized.

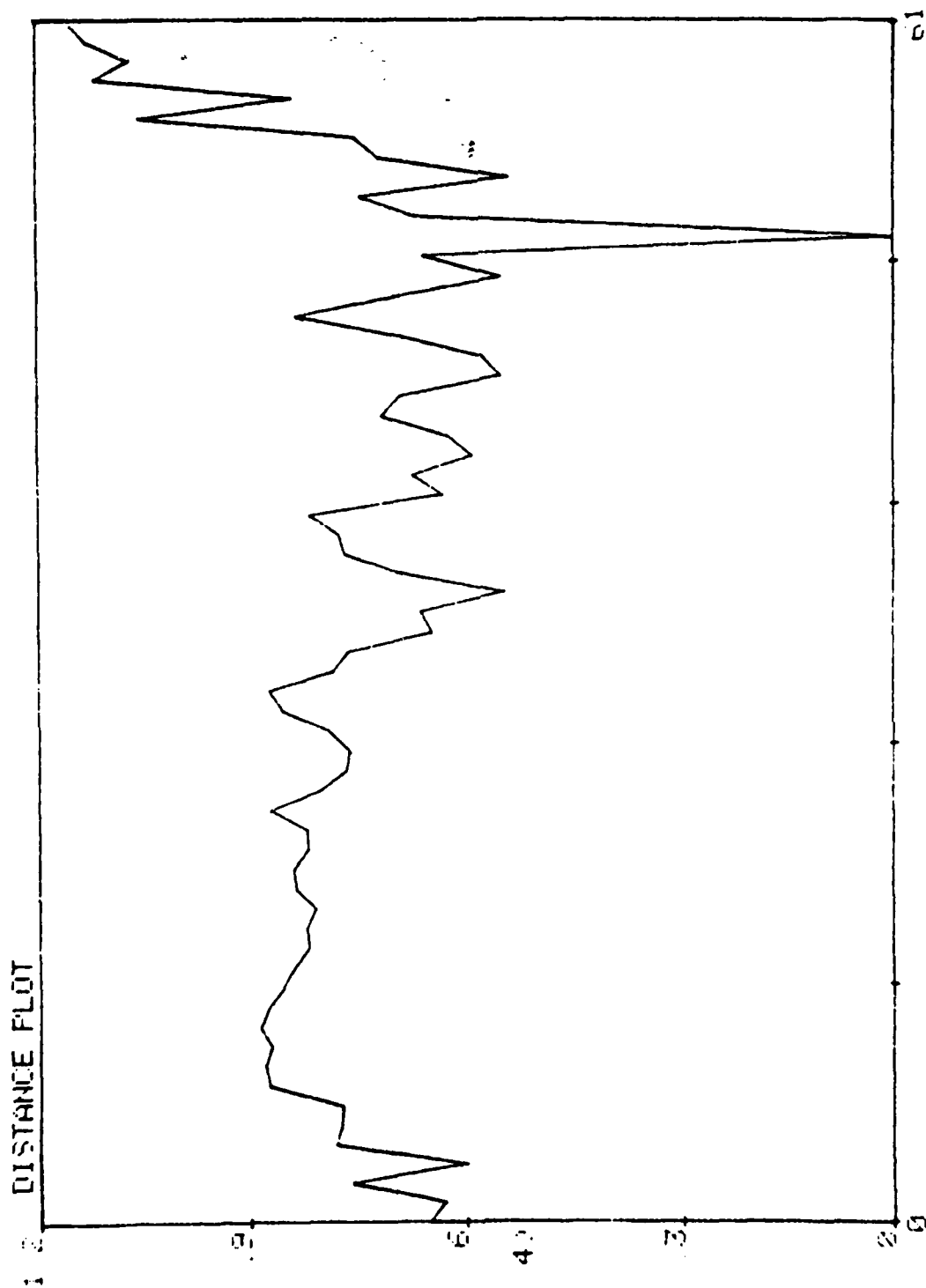


FIGURE 15. Distance plot of observation 91 against 41 through 102. M2 rule, 128 point, nonoverlapped Hamming window, energy normalized.

Figures 16 and 17 are plotted using the spectral plot option of our Acoustic Analyzer, and are observations 45 and 47 from the observation file above. Though these observations are separated by 32MSEC, their resemblance is clear.

FFT WINDOW SIZE:

We have discussed our choice of FFT window size in Section II of this report. A 128 point window provides more frequency resolution than does a 64 point window, and leads to useful behavior of short-time energy with regard to voiced/unvoiced speech (Ref 8). We also found when we computed distances between an observation file and itself in voiced vowel regions of pronounced formant structure, that adjacent observations had similar formant structure using a 64 point window as with a window size of 128 points. To illustrate, observations were computed using a 64 point Hamming window. Observations were de-emphasized at 10dB/Octave below 300Hz and pre-emphasized at 10dB/Octave above 500Hz. The formant structure can be seen to develop in the sequence of plots of observation number 86, 87, 88, 89, and 90; Figures 18, 19, 20, 21, and 22, respectively. These observations are in the same speech segment as Figures 8-17; there are twice the number of observations in a time segment, here, at 64 points per time slice. The M2 distance measure can be seen to develop broad valleys in

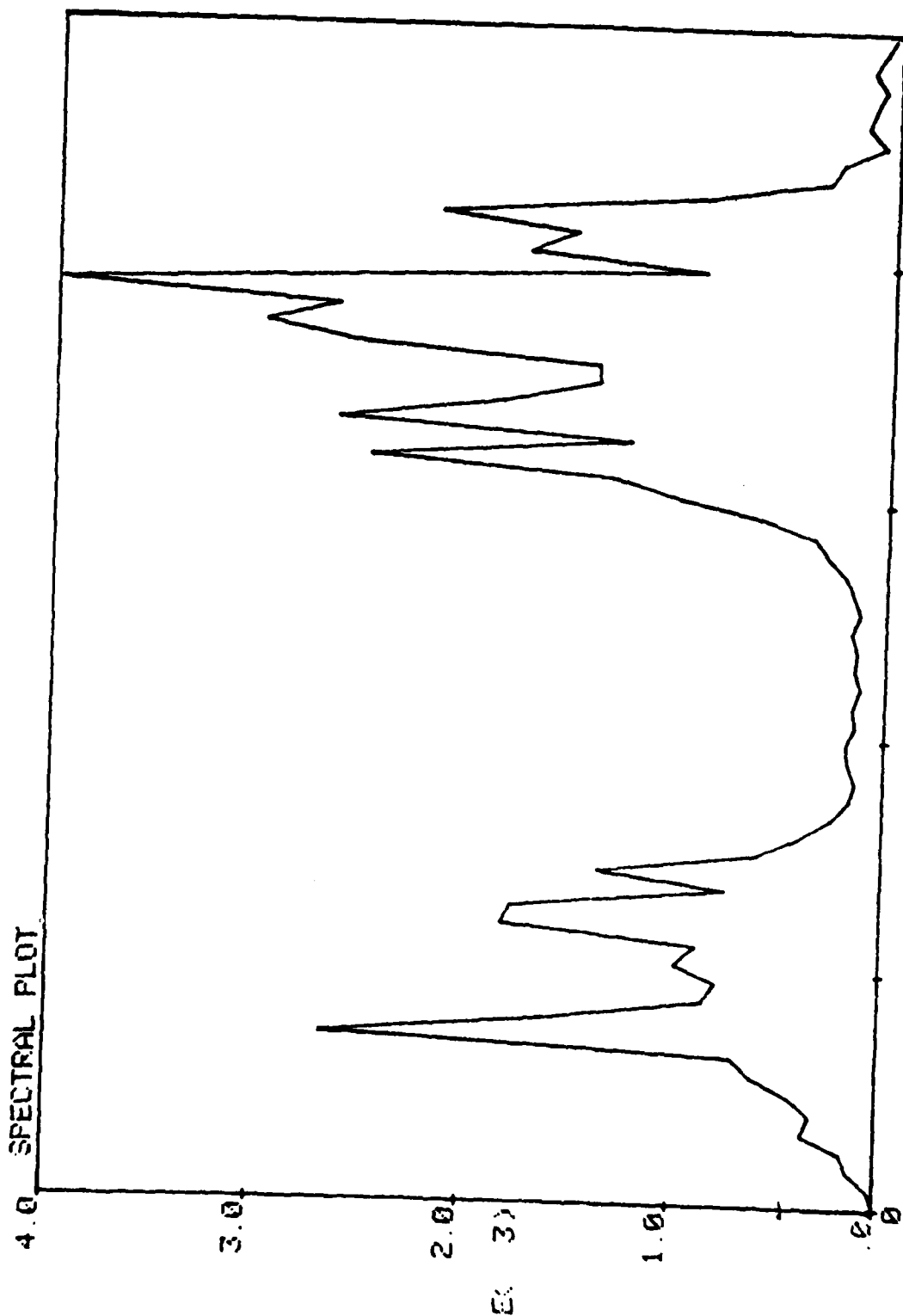


FIGURE 16. Spectral plot of observation number 45. One-hundred twenty-eight (128) point, nonoverlapped Hamming window, energy normalized.

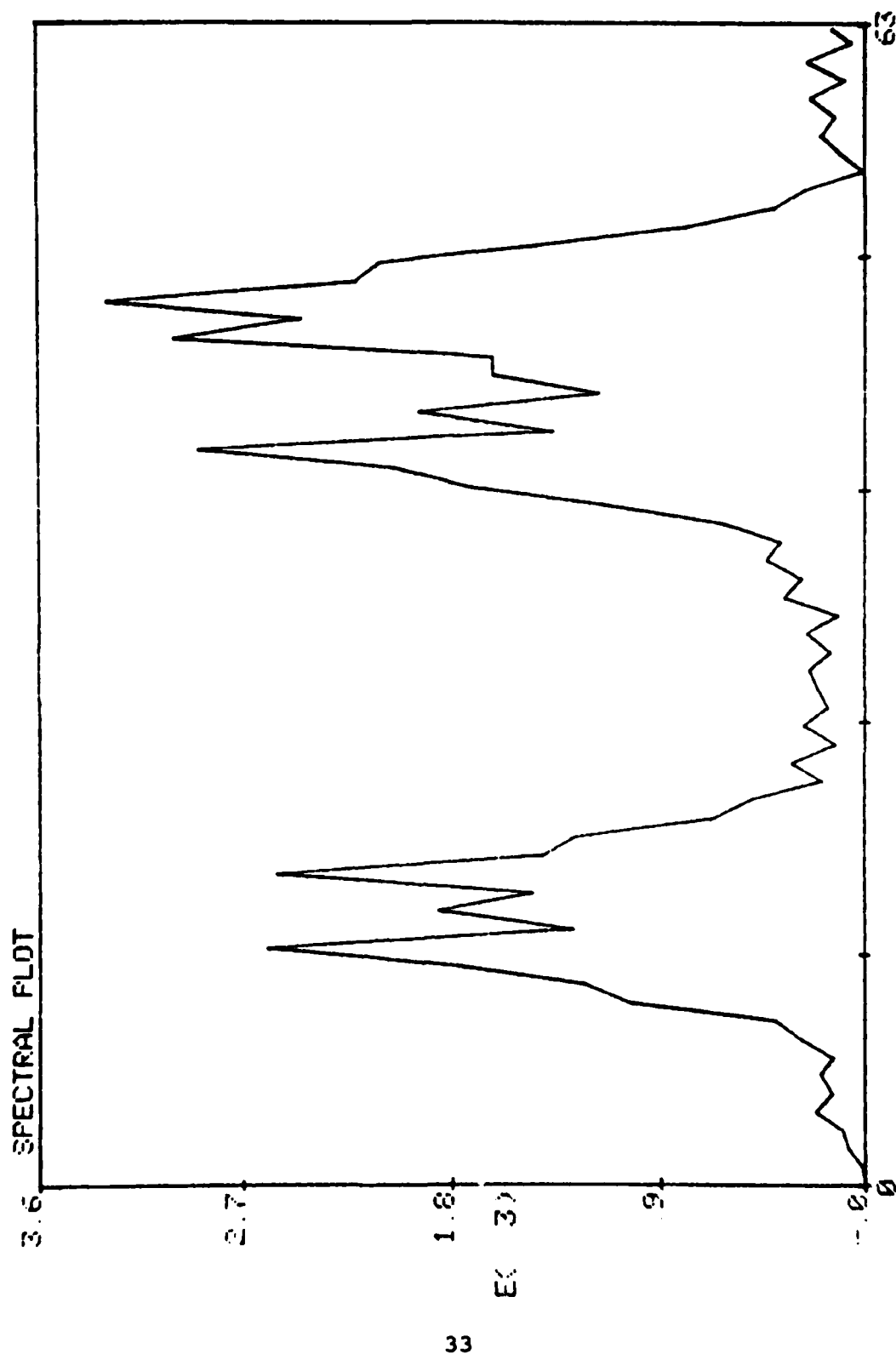


FIGURE 17. Spectral plot of observation number 47. One-hundred twenty-eight (128) point, nonoverlapped Hamming window, energy normalized.

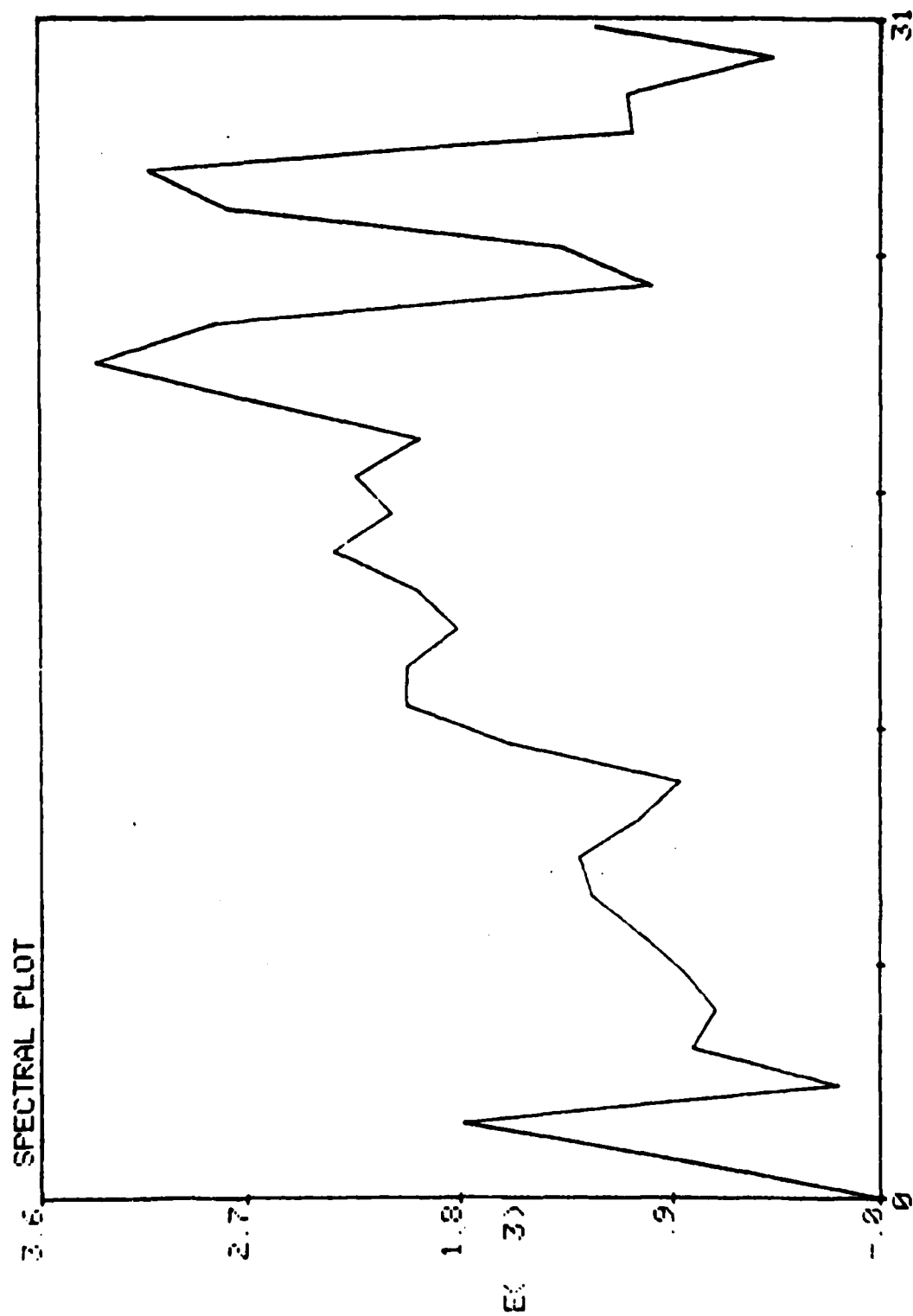


FIGURE 18. Spectral plot of observation number 86. Sixty-four (64) point Hamming window, energy normalized.

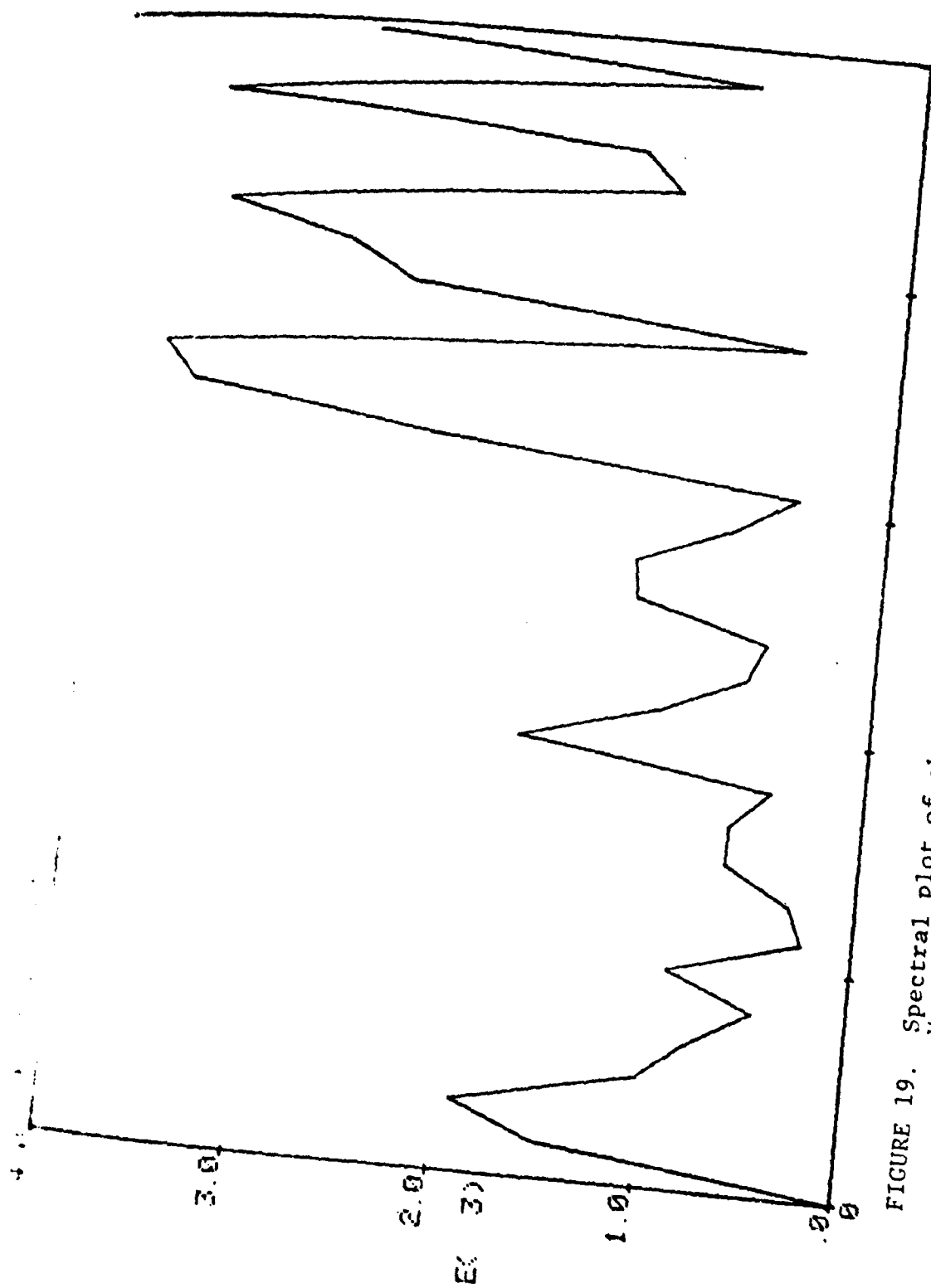


FIGURE 19. Spectral plot of observation number 87. Hamming window, energy normalized.

31

Sixty-four (64) point

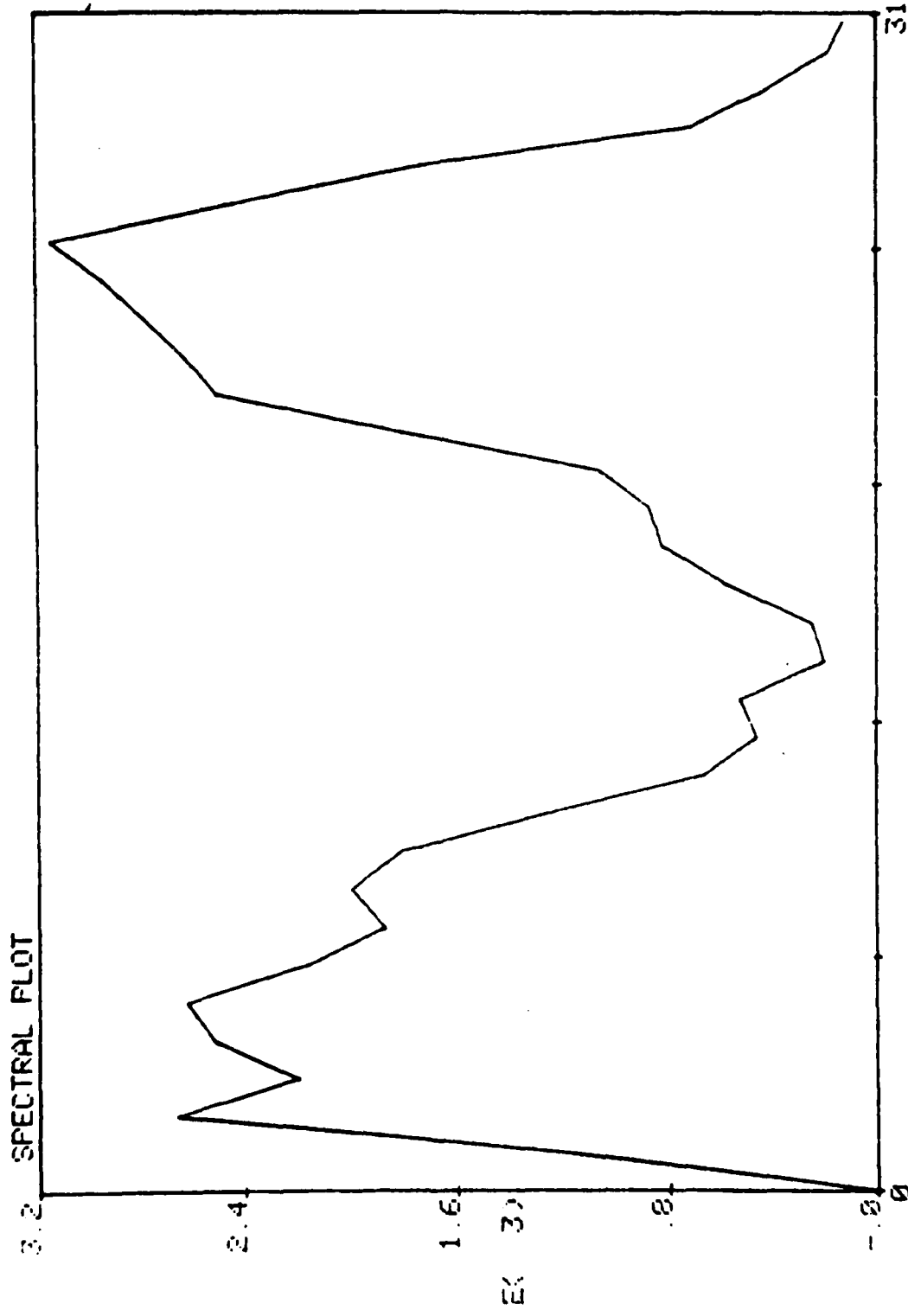


FIGURE 20. Spectral plot of observation number 88. Sixty-four (64) point Hamming window, energy normalized.

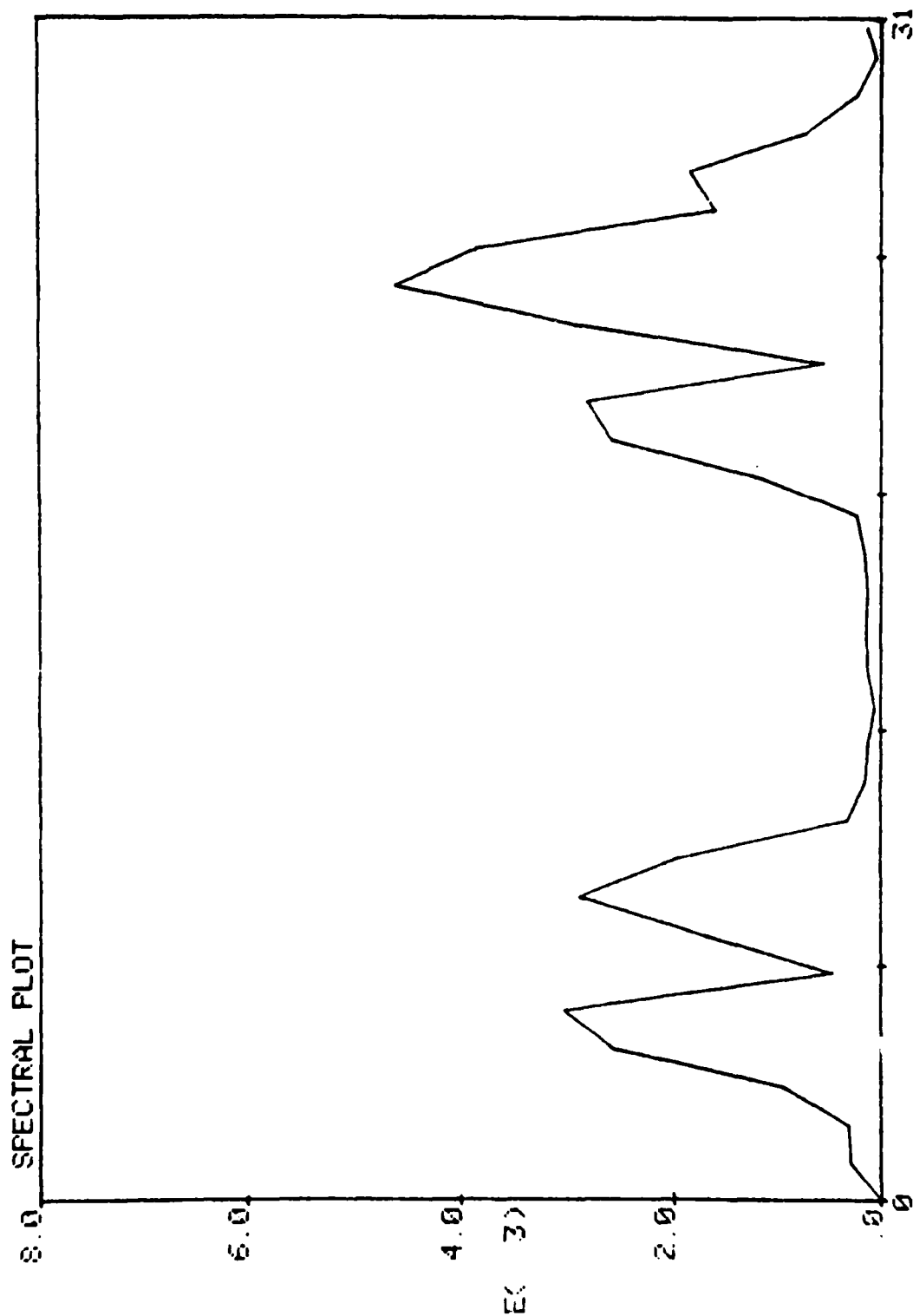


FIGURE 21. Spectral plot of observation number 89. Sixty-four (64) point Hamming window, energy normalized.

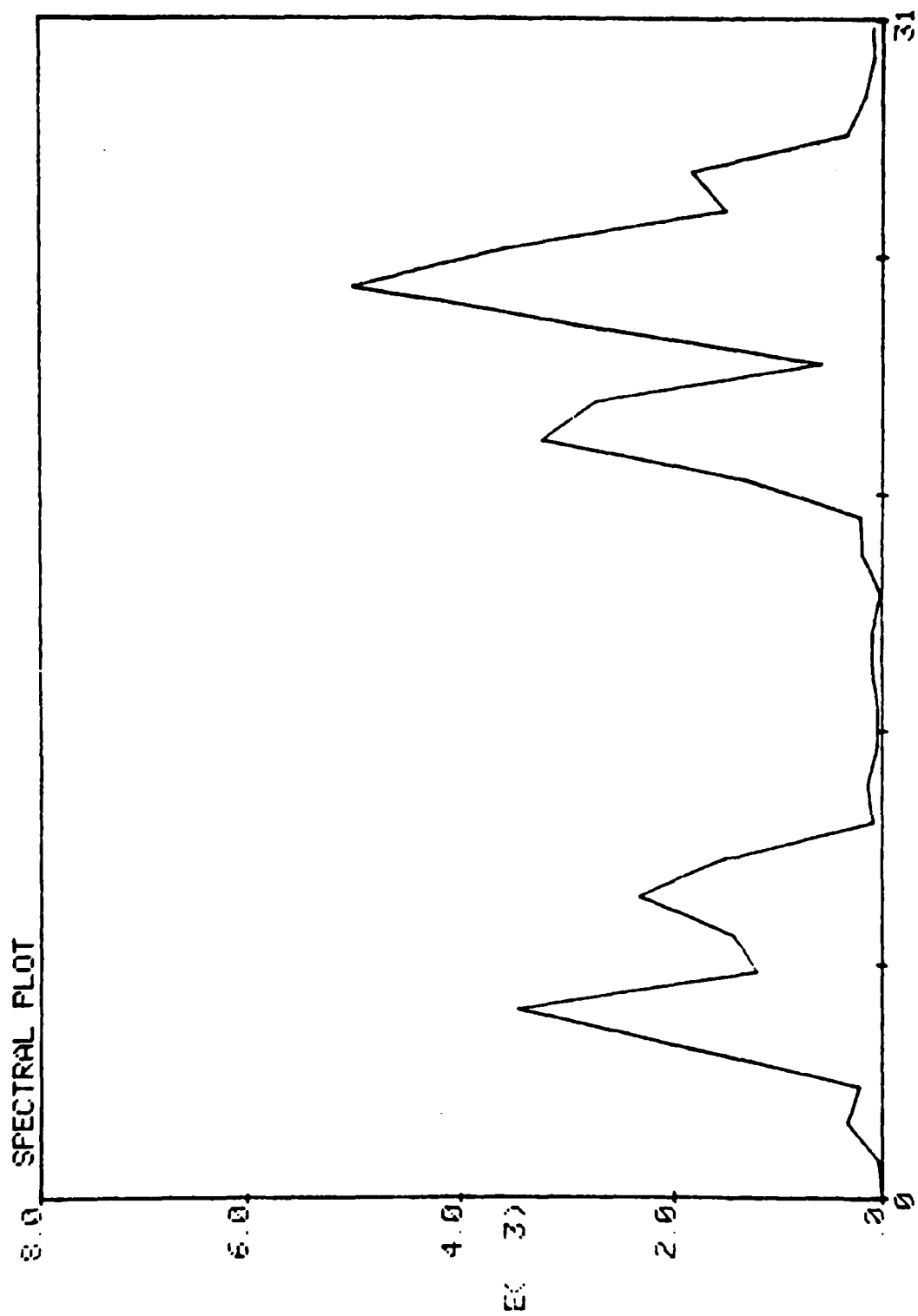


FIGURE 22. Spectral plot of observation number 90. Sixty-four (64) point Hamming window, energy normalized.

regions of most pronounced formant structure in Figures 23, 24, 25, 26, 27, and 28. These are plots of observations number 85, 90, 115, 120, 125, and 130, respectively, all against phonet numbers 81-144.

SAMPLING RATE, WINDOW SHAPE, AND EMPHASIS:

We used speech files digitized at an 8KHz sampling rate and 12 bit quantization because it was the highest sampling rate available. It is well known, of course, that speech sampled at this rate is perfectly intelligible when reconstructed. Seelandt describes the digitizing apparatus (Ref 1). Here we discuss the advisability of changing that rate. We also discuss window shape.

The capability of our Acoustic Analyzer to calculate FFTs of window size 512 or 1024 points allows one to double or quadruple the sampling rate while maintaining the behavior of short-time energy discussed by Rabiner and Schafer (Ref 8) and in Section II of this report. Using the Array Processor, the computational burden of large windows is light. The advantages would not only be an increase in spectral resolution, but also a decrease in spectral distortion due to the longer window. The effects of window shape become more important as pre-emphasis is applied to high frequencies, especially so when a large portion of the energy in speech is concentrated in the lower frequency range. In this case, the sidelobes from the low frequency

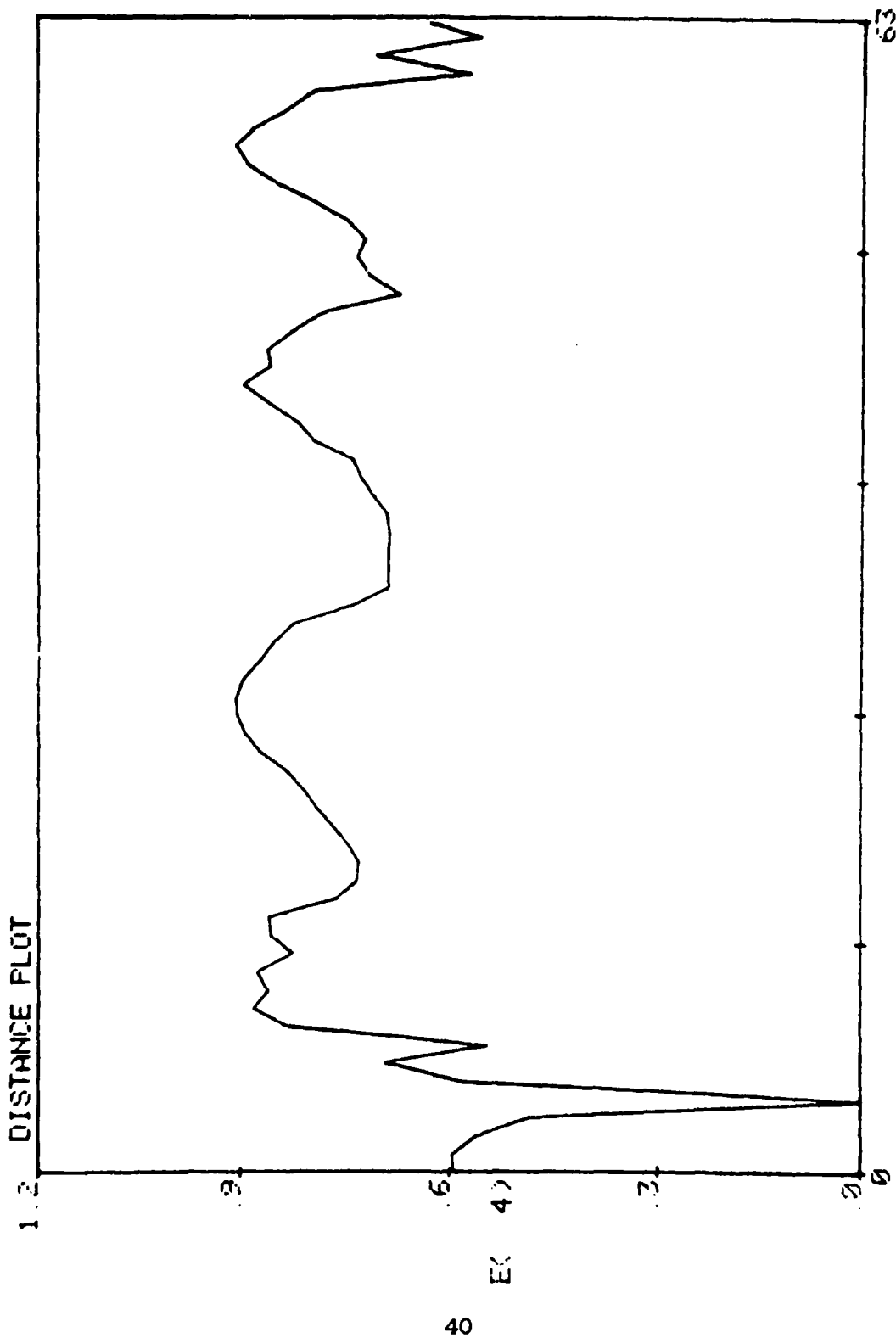


FIGURE 23. Distance plot of observation 85 against 81 through 144.  
M2 rule, 64 point Hamming window, energy normalized.

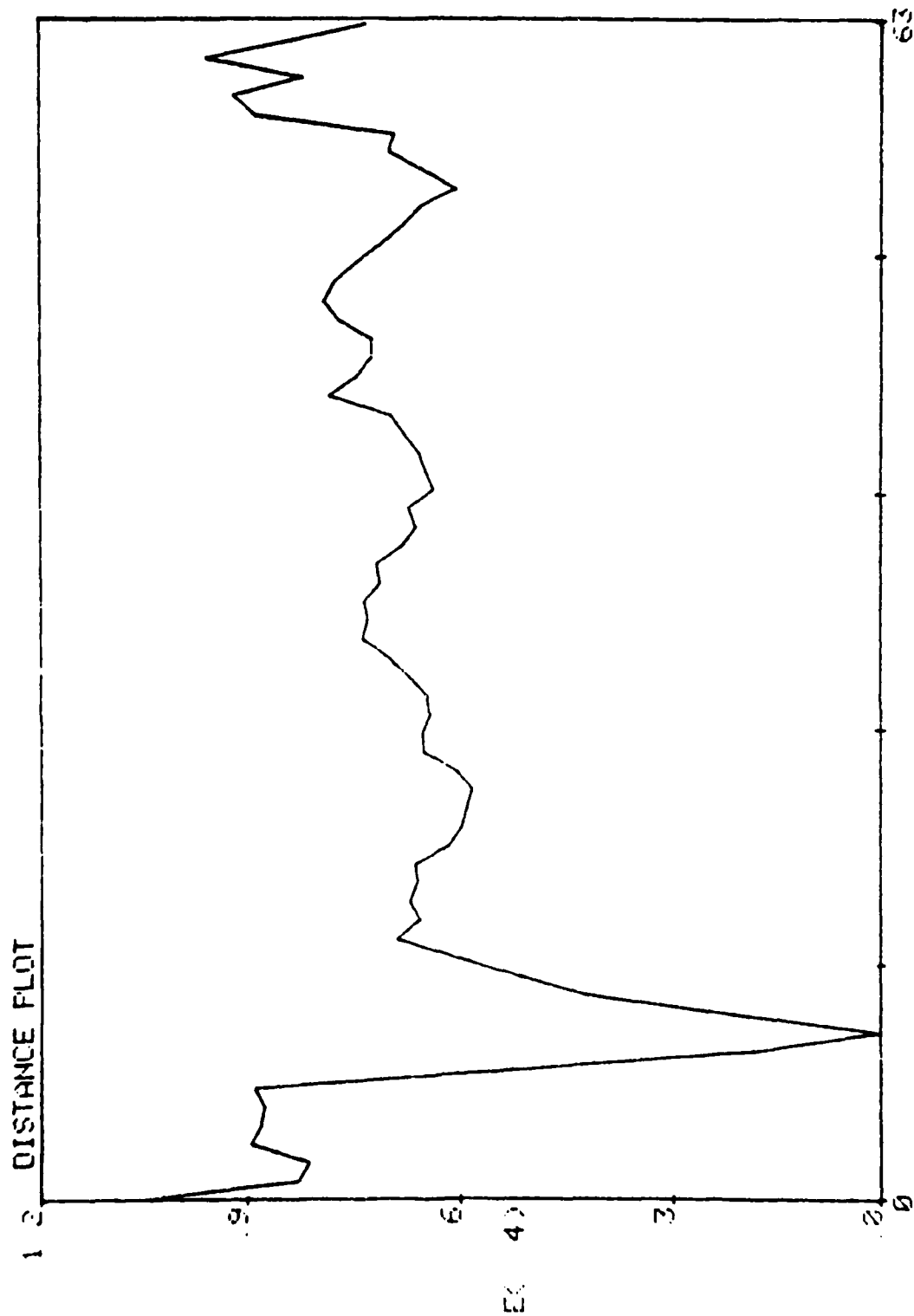


FIGURE 24. Distance plot of observation 90 against 81 through 144. M2 rule, 64 point Hamming window, energy normalized.

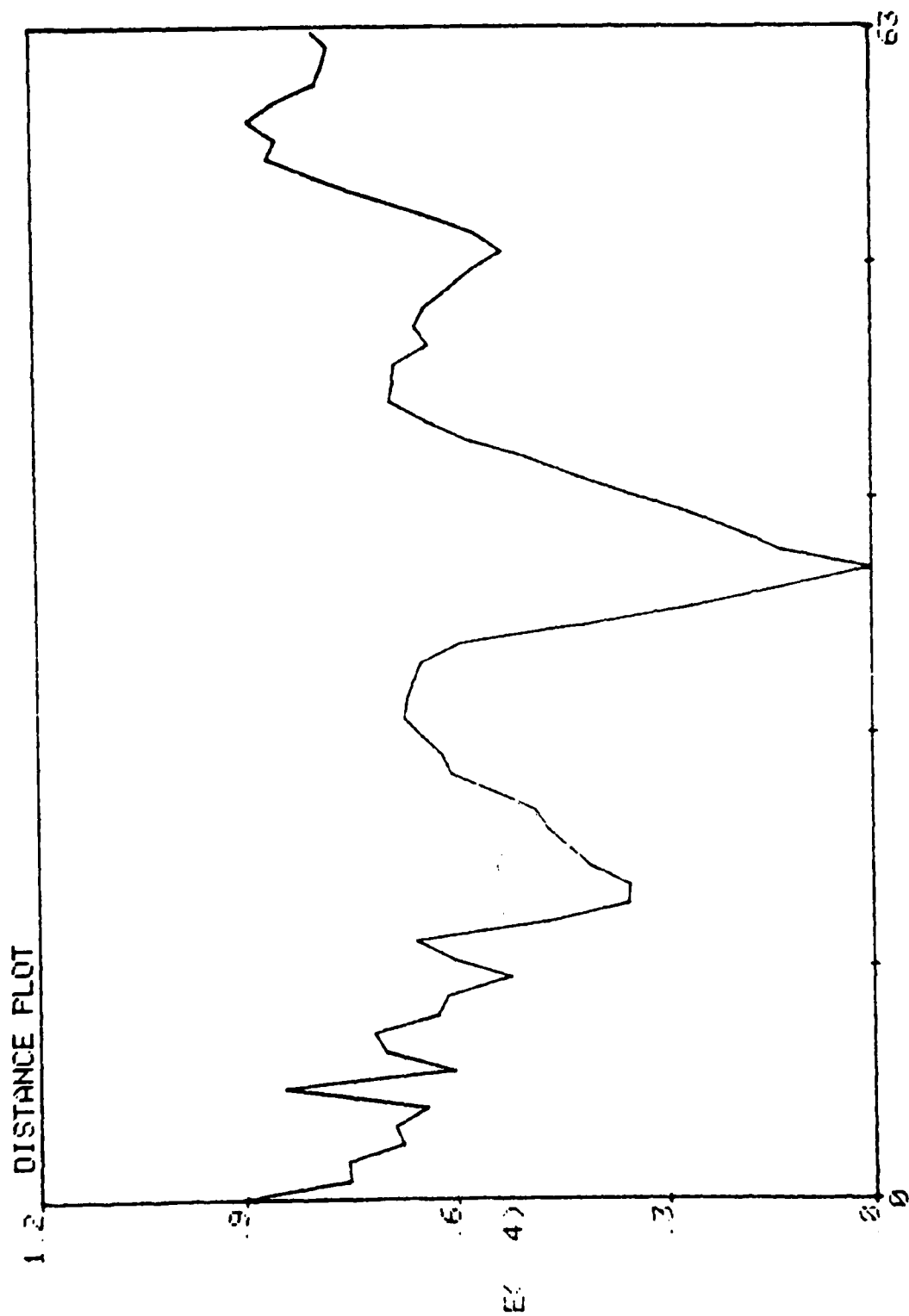


FIGURE 25. Distance plot of observation 115 against 81 through 144. M2 rule, 64 point Hamming window, energy normalized.

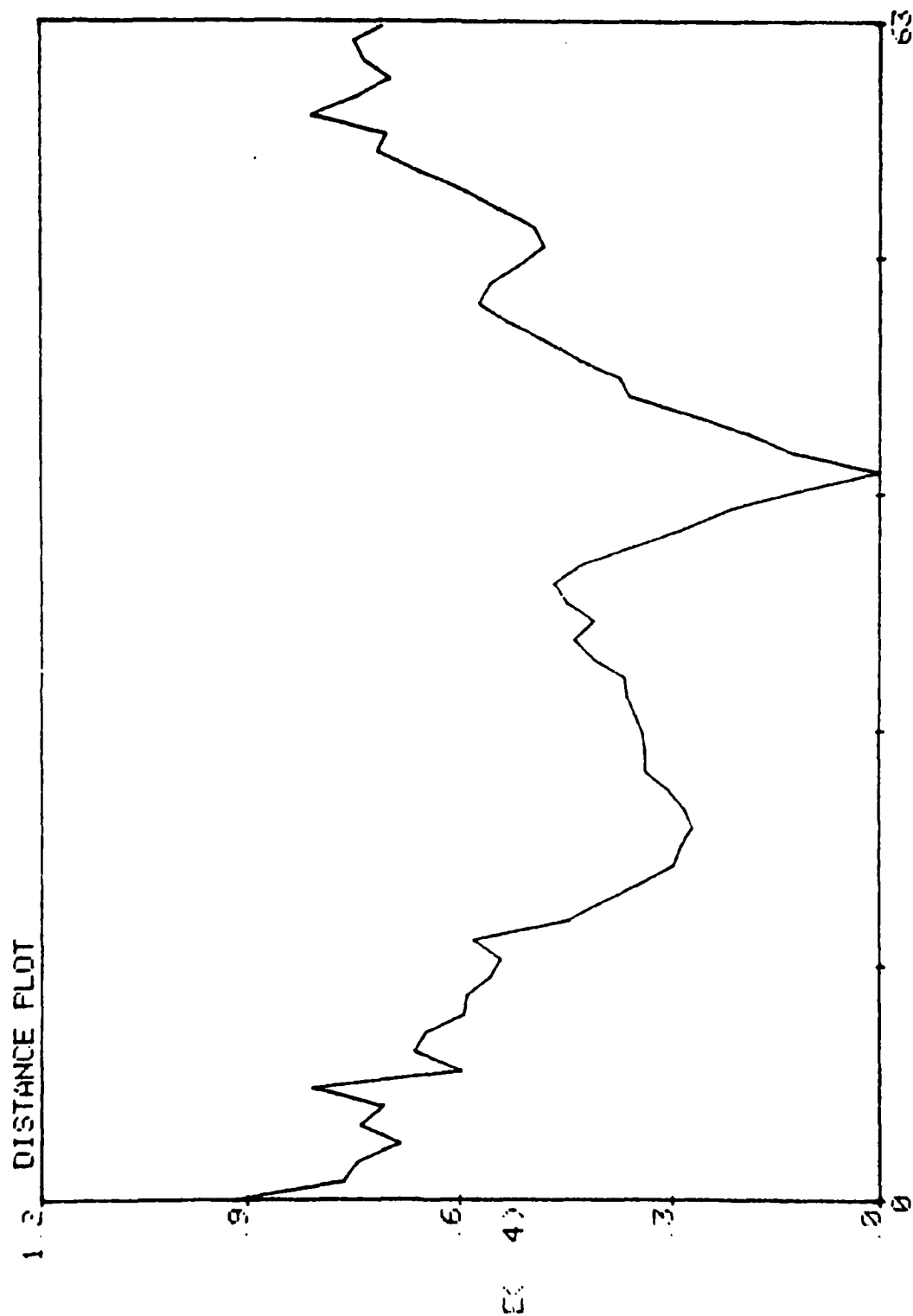


FIGURE 26. Distance plot of observation number 120 against 81 through 144. M2 rule, 64 point Hamming window, energy normalized.

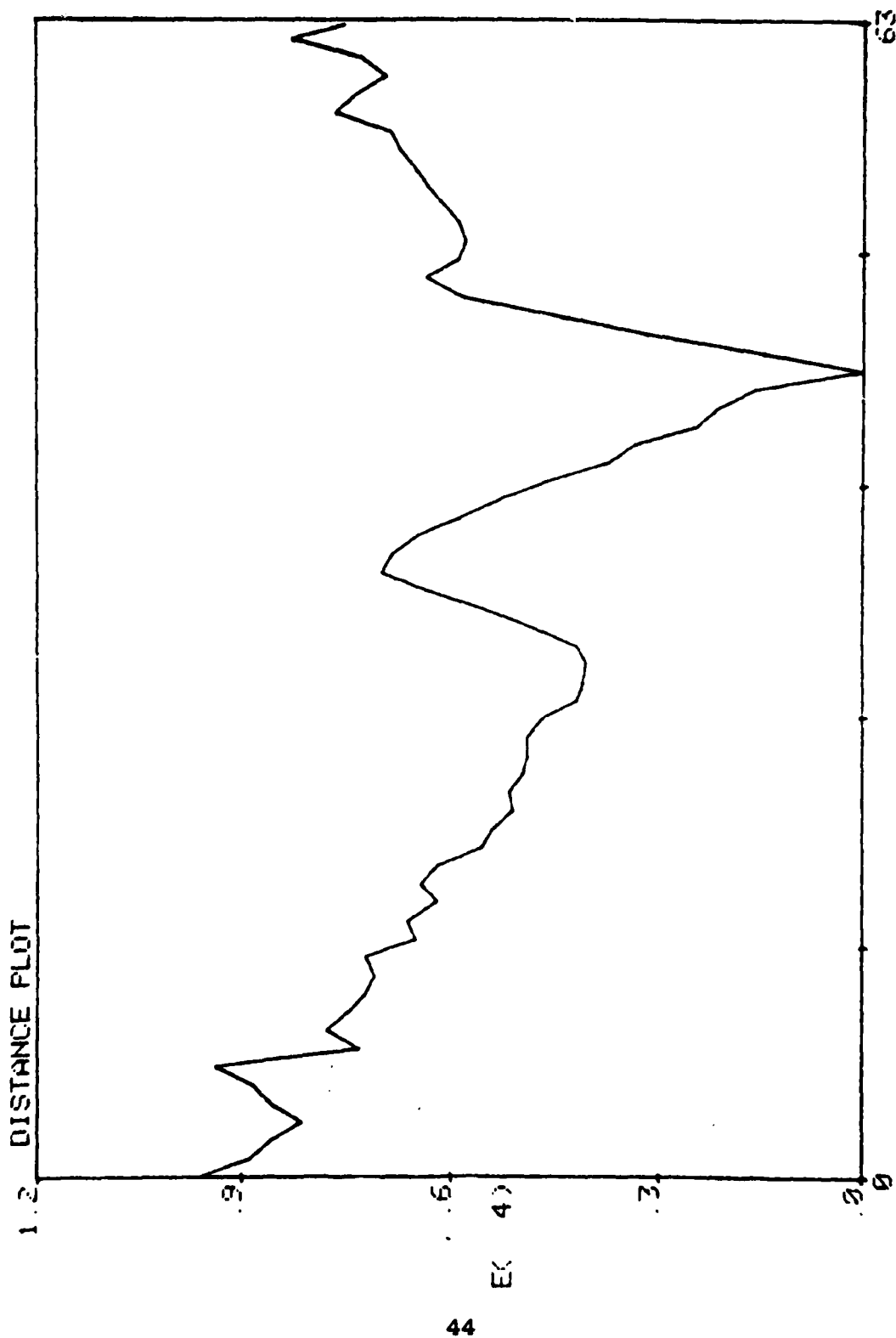


FIGURE 27. Distance plot of observation 125 against 81 through 144. M2 rule, 64 point Hamming window, energy normalized.

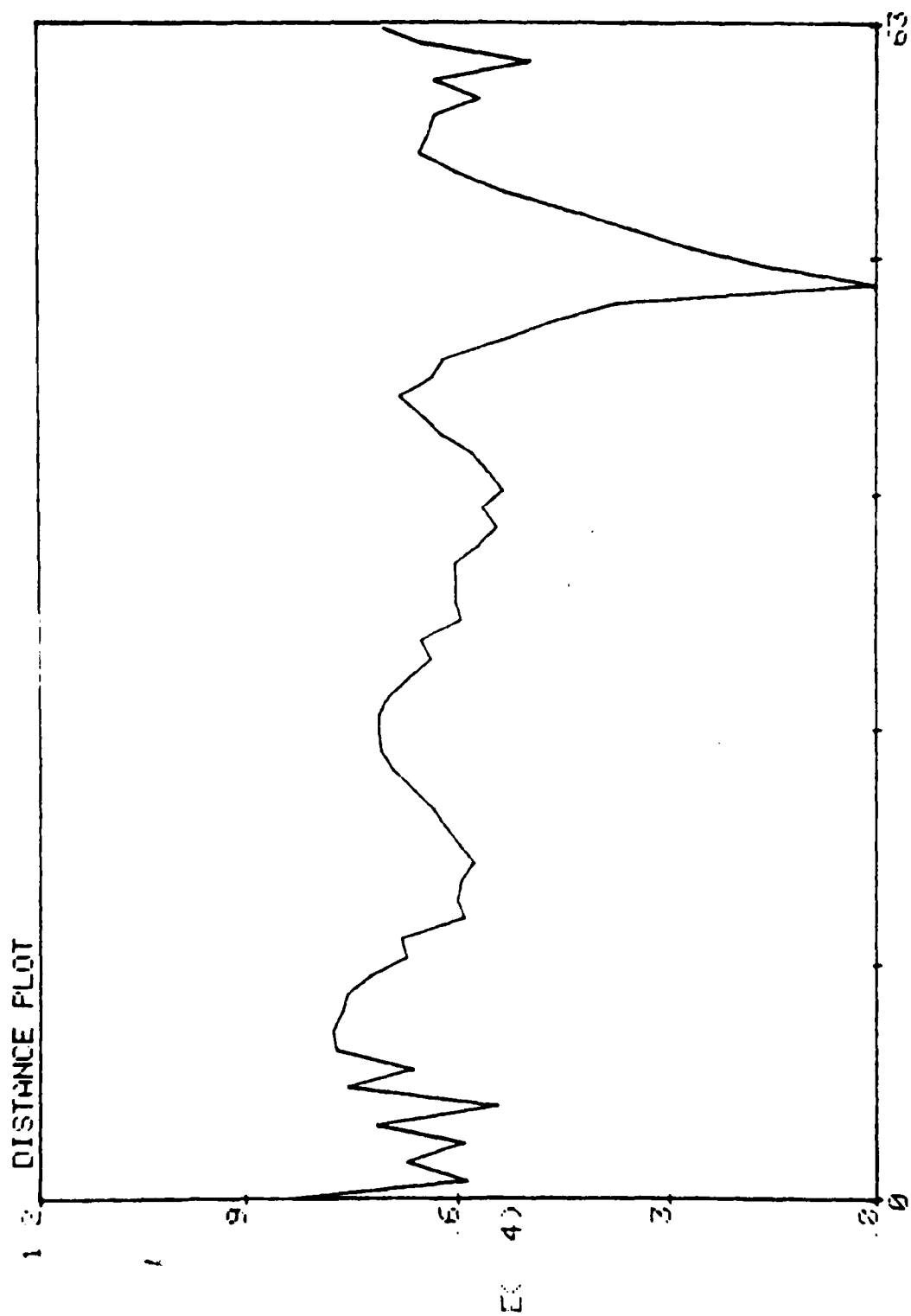


FIGURE 28. Distance plot of observation 130 against 81 through 144. M2 rule, 64 point Hamming window, energy normalized.

energy are emphasized. Sivian showed that formant amplitude rolled-off at approximately 10dB per Octave relative to the fundamental (Ref 11). Kuligowski has shown the same (Ref 12:20, 21). If pre-emphasis is applied to the speech spectrum to compensate for this roll-off, then sidelobes of the window spectrum excited by large amplitude fundamental and first formant components have a 20dB per Octave advantage over the high formants: 10dB per Octave because the formant is down that far and 10dB per Octave because the sidelobe is emphasized that much. De-emphasis of the fundamental in the spectral domain does not affect these sidelobes because they are generated in the transformation and de-emphasis is applied afterwards. They can be lowered by:

- (1) High-pass filtering before the FFT to decrease the energy in the fundamental, and hence in the sidelobes.

- (2) Using the longest possible window size and least distorting window shape while maintaining the 10MSEC-to-20MSEC window duration for short-time energy.

The first solution is difficult to apply. Kuligowski has shown (Ref 12:20, 21) that the location of the fundamental varies by as much as an octave over the vowels he listed, and depending on speaker sex and age. Further, for several vowel-speaker combinations, the fundamental of one vowel-speaker combination is at the same spectral location as the first formant at other vowel-speaker combinations. An adaptive filter before the FFT would be necessary to

attenuate the fundamental prior to transformation without attenuating the first formant. The second solution can be implemented with enough memory and a fast enough analog-to-digital converter. At a 1024 point window size, a sampling rate of 68KHz would provide a window duration of 15MSEC.

As has been previously noted, the FFT is a robust feature (Ref 7). It may be sufficiently robust for an 8KHz sampling rate and a 128 point Hamming window with the observation pre-emphasized at 10dB per Octave. The measure of sufficiency is in the way the features cluster, and an adequate feature-space partitioning algorithm for this problem has not yet been implemented.

It is clear that the fundamental should be de-emphasized in the observation spectrum. It is a high energy feature which is highly speaker dependent. The problem with de-emphasizing it is that for some vowel-speaker combinations, it is located at or near the first formant of other vowel-speaker combinations (Ref 12:20, 21). We chose to de-emphasize at 10dB/Octave ending at a corner frequency of 300Hz as a compromise between allowing too much fundamental energy to remain in the observation, and reducing the amount of energy in the important first formant.

#### THRESHOLDING:

The last parameter Seelandt identified for further study is a threshold below which phonetic units should be

attenuated when their energy content is low. When the energy in a time slice did not exceed a preset threshold, the spectral components were attenuated rather than energy normalized. Seelandt did this to prevent vectors which consist predominately of background noise from entering into the decision process (Ref 1:121, 122).

We found that unvoiced fricative sounds, especially those in sounds involving the letter "f," are low energy sounds and hard to distinguish from noise in a spectrogram. Phonet plots did reveal structure in these sounds that might distinguish them from white noise (flat spectrum), or other background noise models. Distance measures computed to phonets which represent background noise, coupled with knowledge of phonet and observation energy, may permit a phonet-to-word translation machine, such as Montgomery's (Ref 10), to distinguish between unvoiced fricative sounds and noise, and may aid in detecting word boundaries as well. Figures 29-55 illustrate observations taken from regions of the speech file CT56.SP known to contain noise as well as the words "five" and "six." The plots are in dB versus spectral component number. Component number zero is the observation energy, also in dB. Figures 29-34 are from a region where there was no speaker sound. Compare those figures with Figures 35-42 which are from a region of the file where the speaker begins the "five" sound. It is difficult to detect where the "f" sound begins, either by

studying the spectral distribution or by noting the energy. One can see that as the characteristic formant structure associated with the vowel "i" sound develops, the energy increases from roughly 63dB to roughly 100dB in the "i" sound. But it does not appear that energy content is a useful means for discriminating between noise and the "f" sound.

Figures 43-50 were taken from the word "five" from well into the vowel sound "i" through the "v" sound on into a pause between the words "five" and "six." One can see the spectral distribution change from the "i" formant structure through a "v" sound structure of relatively flat spectrum but moderate energy, to a nonspeaker background noise of flat spectrum and low energy. Figures 51-56 show a sequence of observations from the speakerless background region between the words to well into the "s" in "six." Note that as the characteristic upward slope (left-to-right) of the spectral distribution of the "s" sound develops, the energy increases roughly 25dB. One can see in this overall sequence of figures that energy is a poor discriminant between noise and some unvoiced sounds. We conclude that energy is an unreliable discriminant between background noise and unvoiced speech.

The question of how unvoiced sounds may be detected in noise needs to be addressed. It is beyond the scope of this project to do so because it remains to be seen if

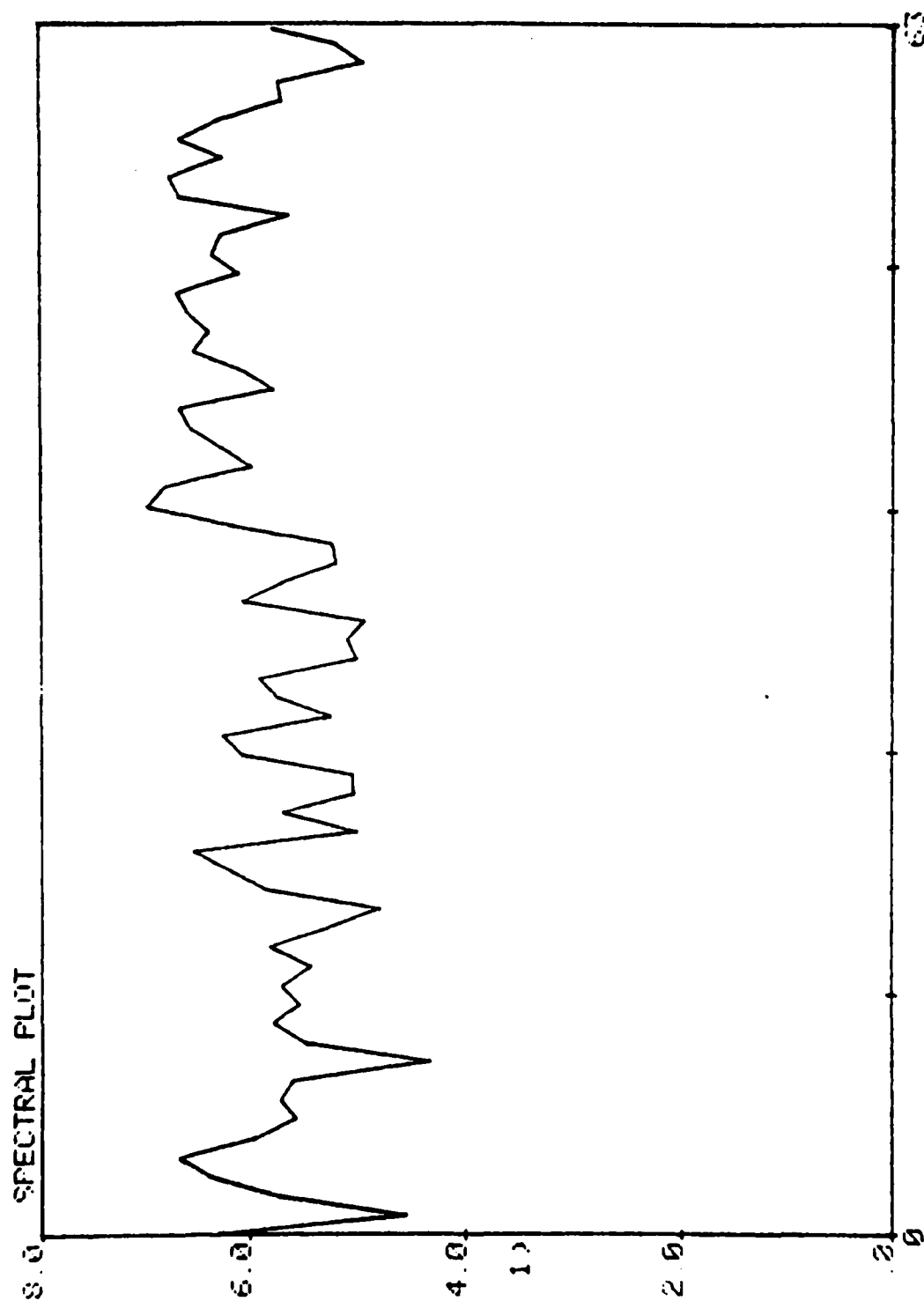


FIGURE 29. Spectral plot observation number 21, 128 point overlapping Hamming window, 10dB de-emphasis to 300HZ, 10dB pre-emphasis from 500HZ, energy normalized.

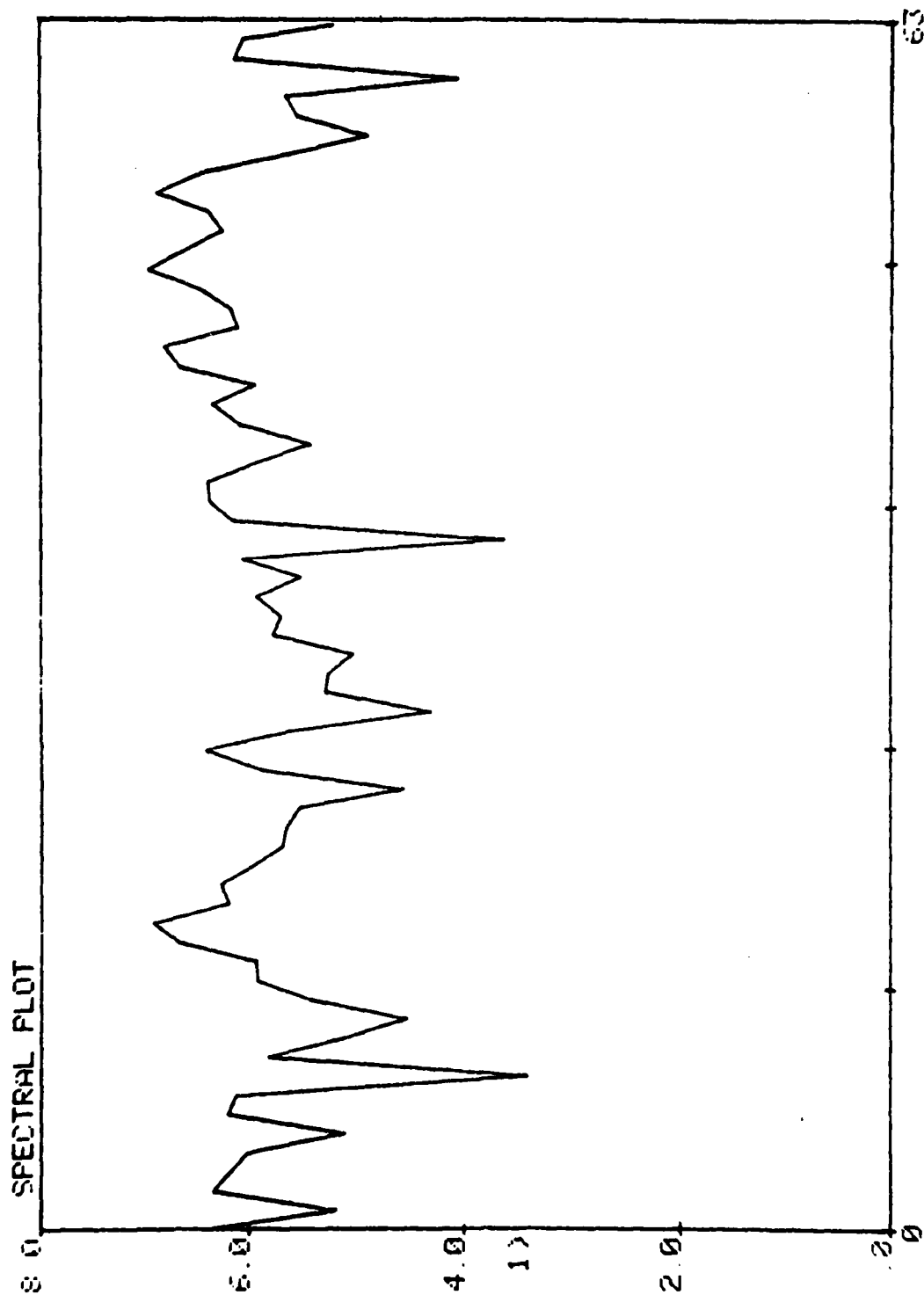


FIGURE 30. Spectral plot observation number 24, 128 point overlapping Hamming window, 10dB de-emphasis to 300HZ, 10dB pre-emphasis from 500HZ, energy normalized.

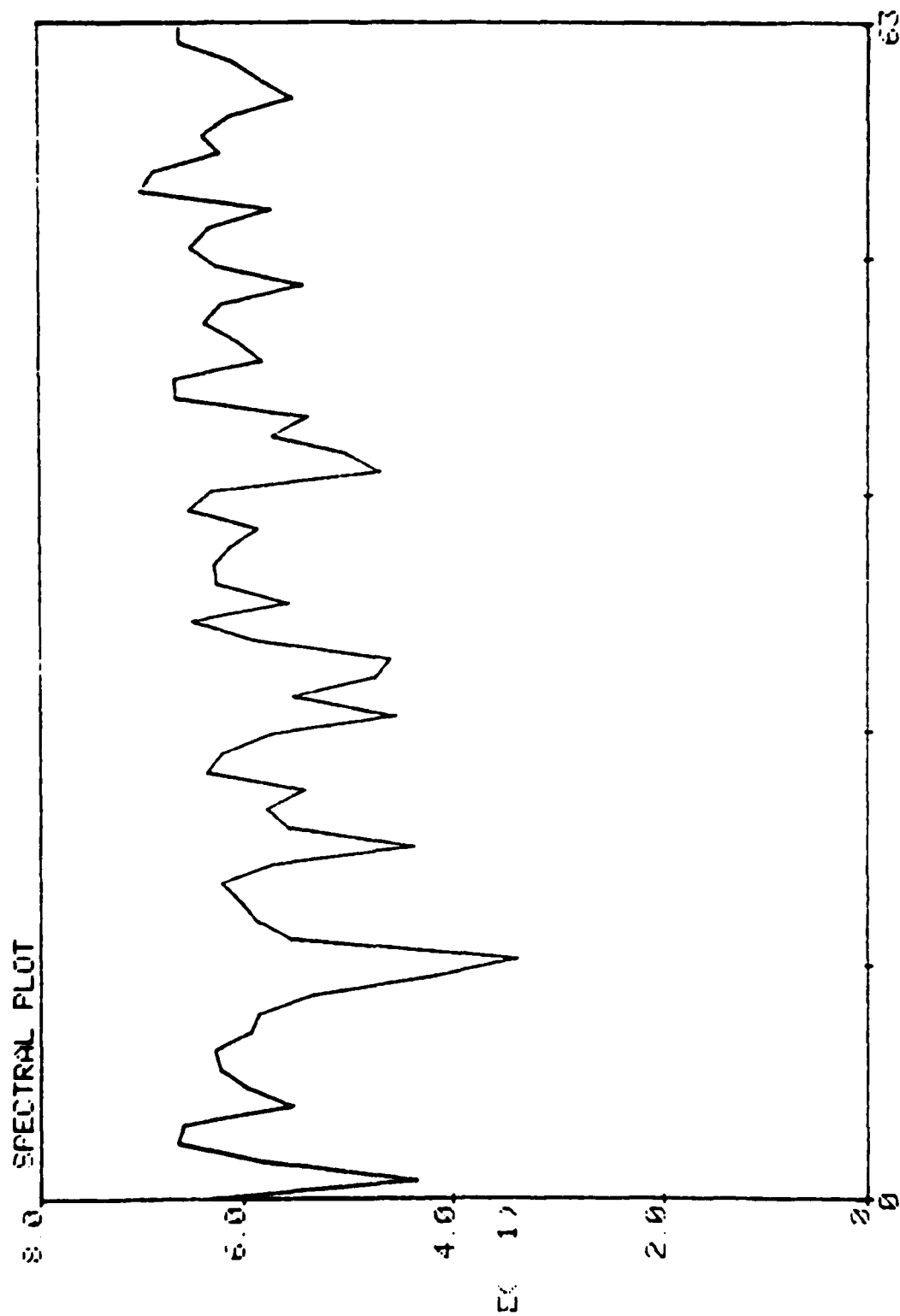


FIGURE 31. Spectral plot observation number 31, 128 point overlapping Hamming window, 10dB de-emphasis to 300HZ, 10dB pre-emphasis from 500HZ, energy normalized.

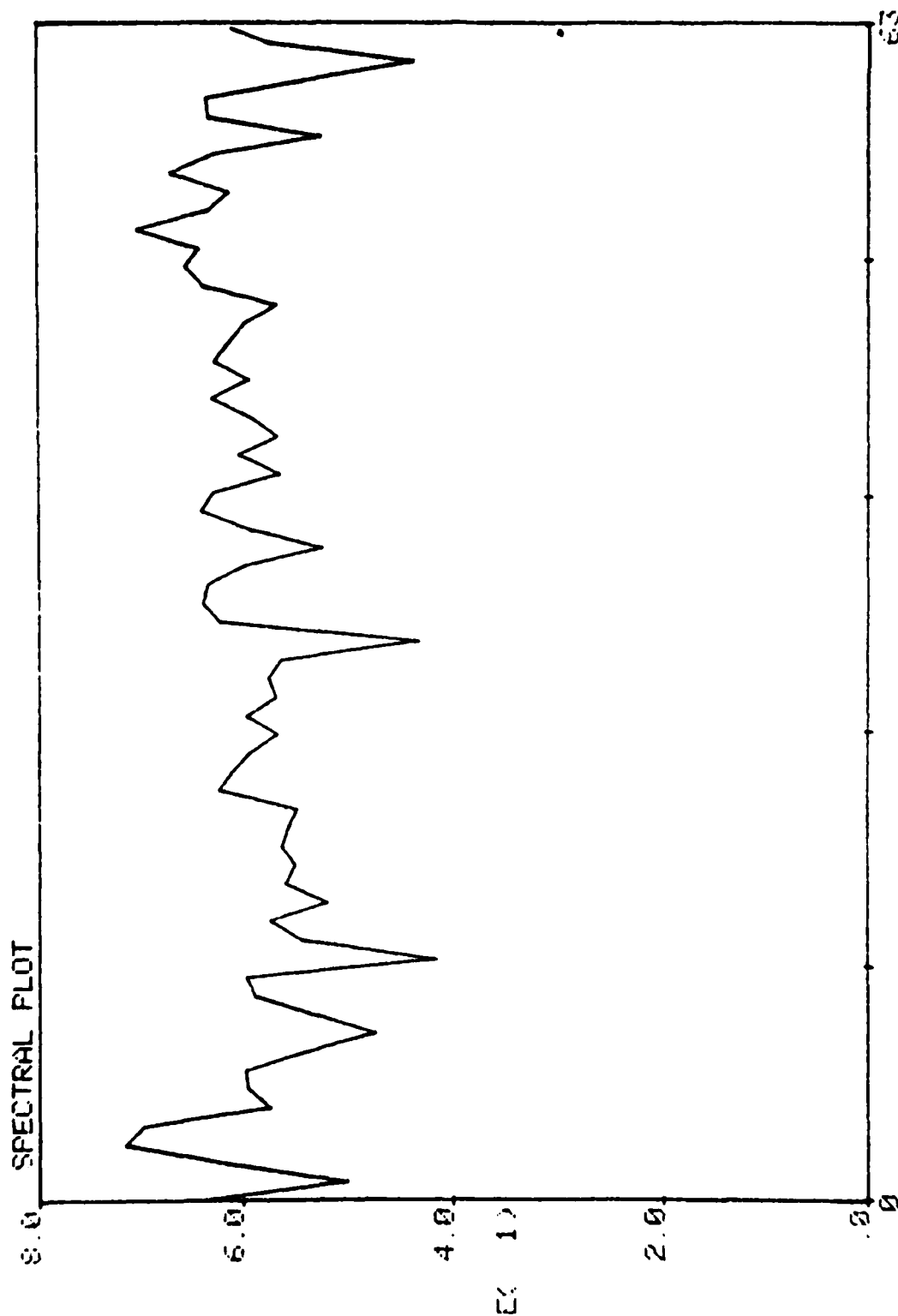


FIGURE 32. Spectral plot observation number 34, 128 point overlapping Hamming window, 10dB de-emphasis to 300HZ, 10dB pre-emphasis from 500HZ, energy normalized.

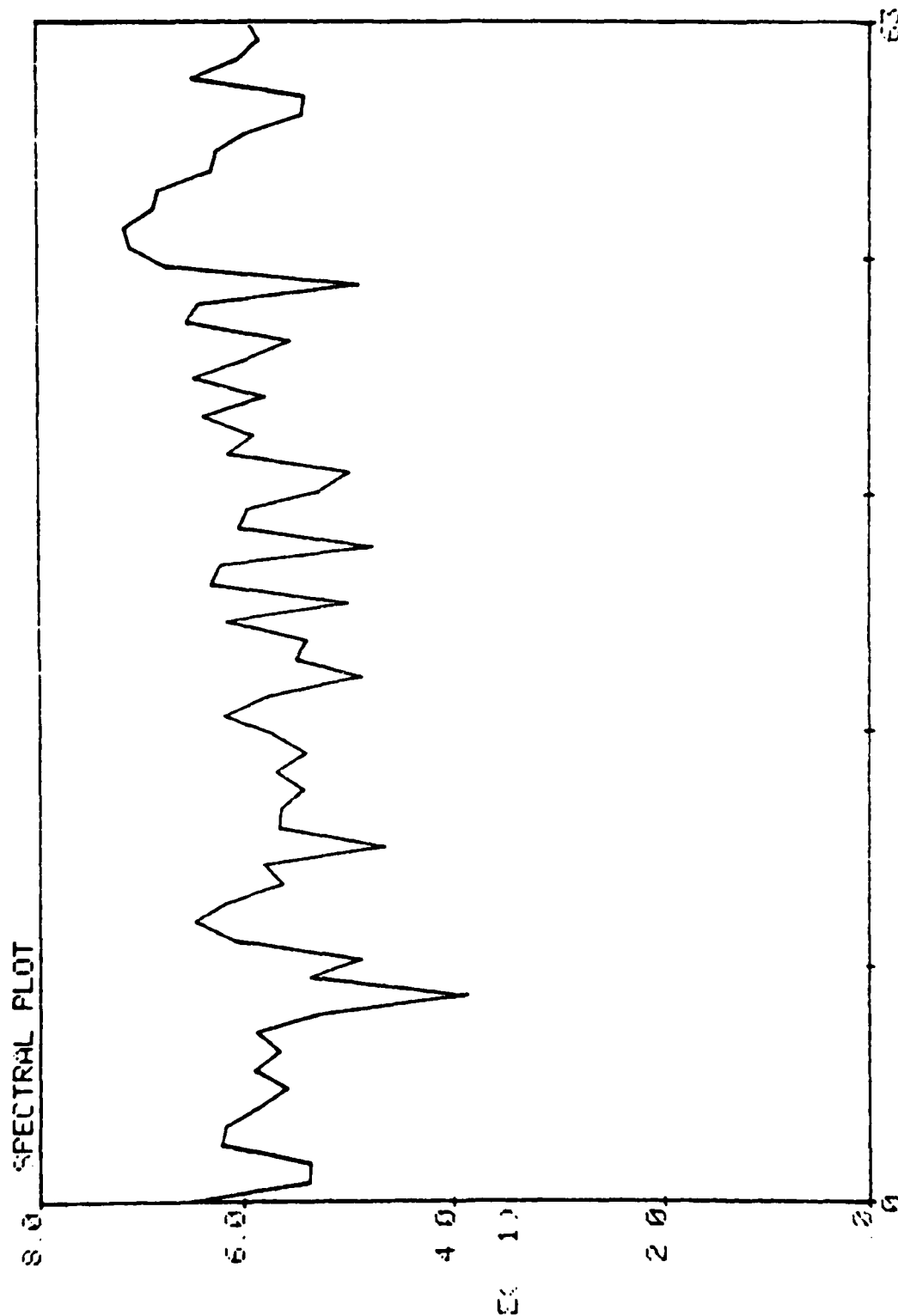


FIGURE 33. Spectral plot observation number 70, 128 point overlapping Hamming window, 10dB de-emphasis to 300HZ, 10dB pre-emphasis from 500HZ, energy normalized.

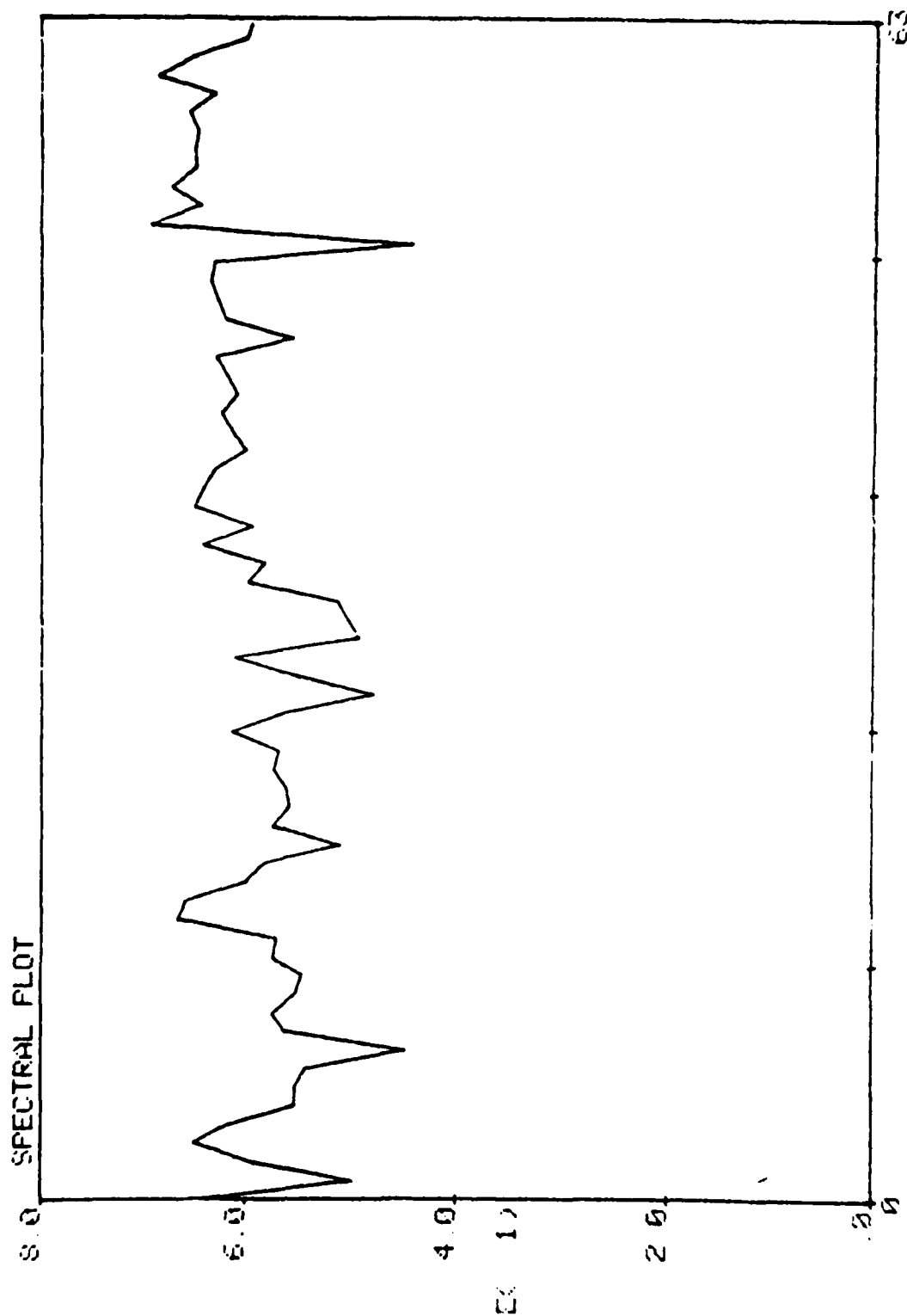


FIGURE 34. Spectral plot observation number 75, 128 point overlapping Hamming window, 10dB de-emphasis to 300HZ, 10dB pre-emphasis from 500HZ, energy normalized.

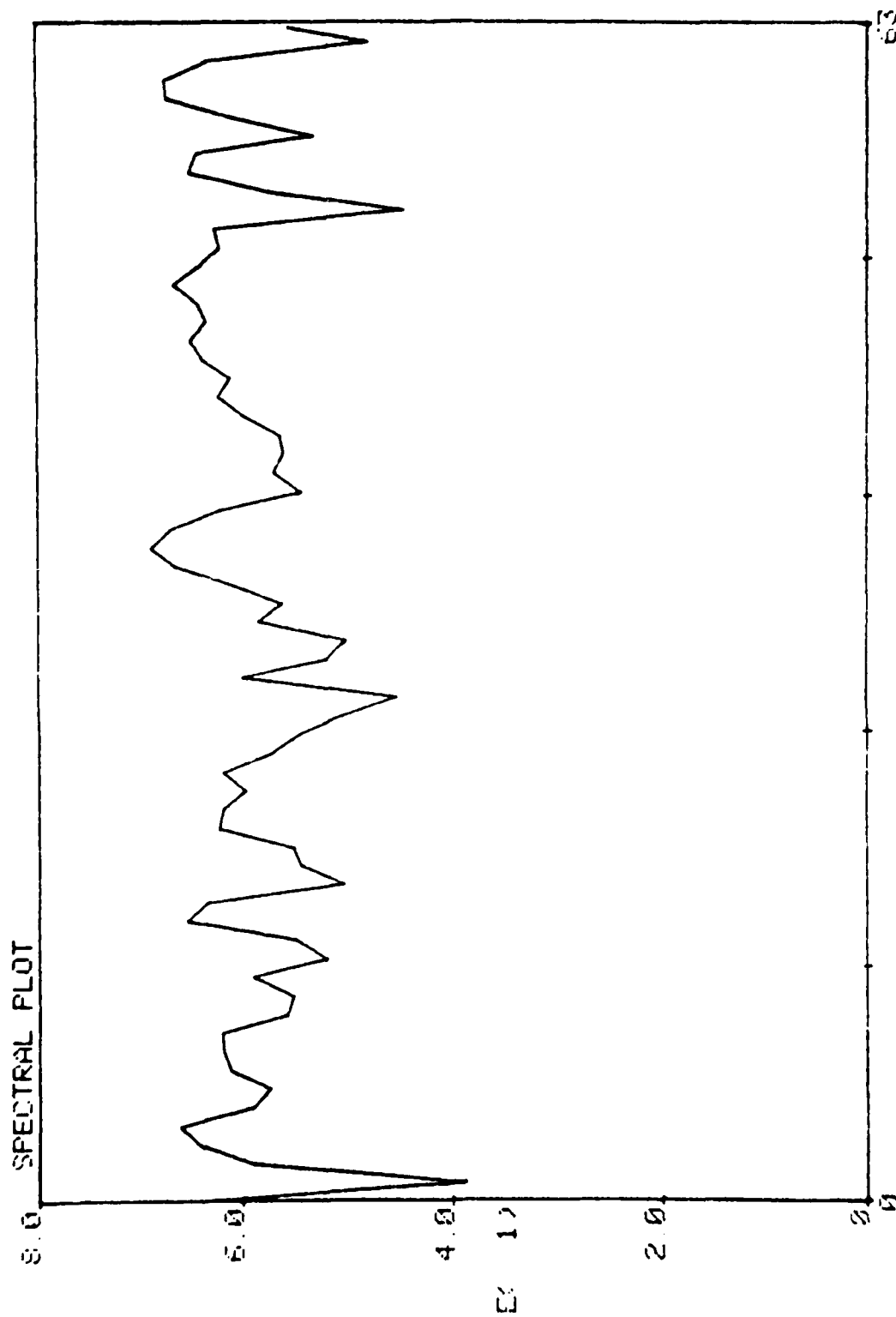


FIGURE 35. Spectral plot observation number 80, 128 point overlapping Hamming window, 10dB de-emphasis to 300Hz, 10dB pre-emphasis from 500Hz, energy normalized.

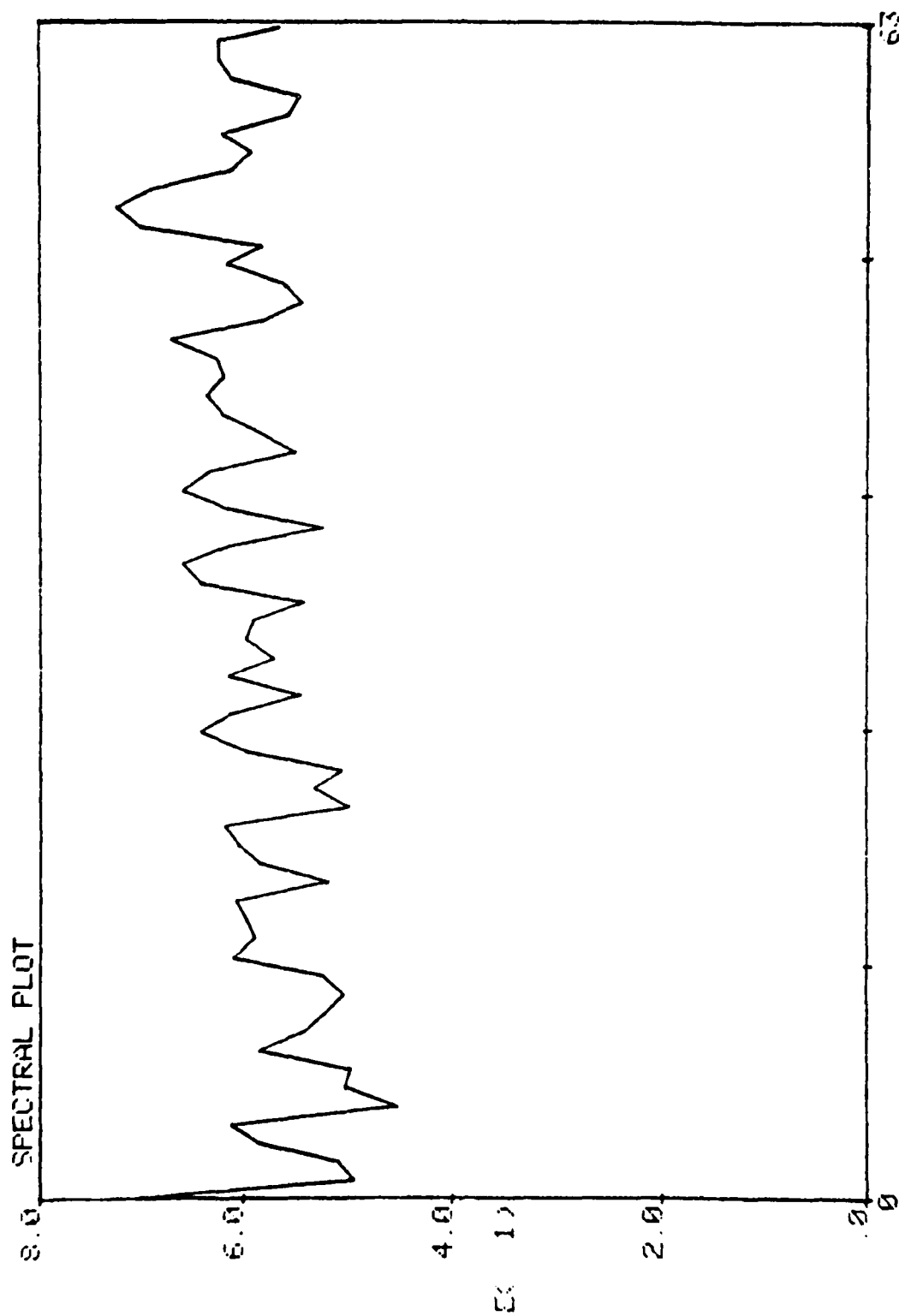


FIGURE 36. Spectral plot observation number 85, 128 point overlapping Hamming window, 10dB de-emphasis to 300Hz, 10dB pre-emphasis from 500Hz, energy normalized.

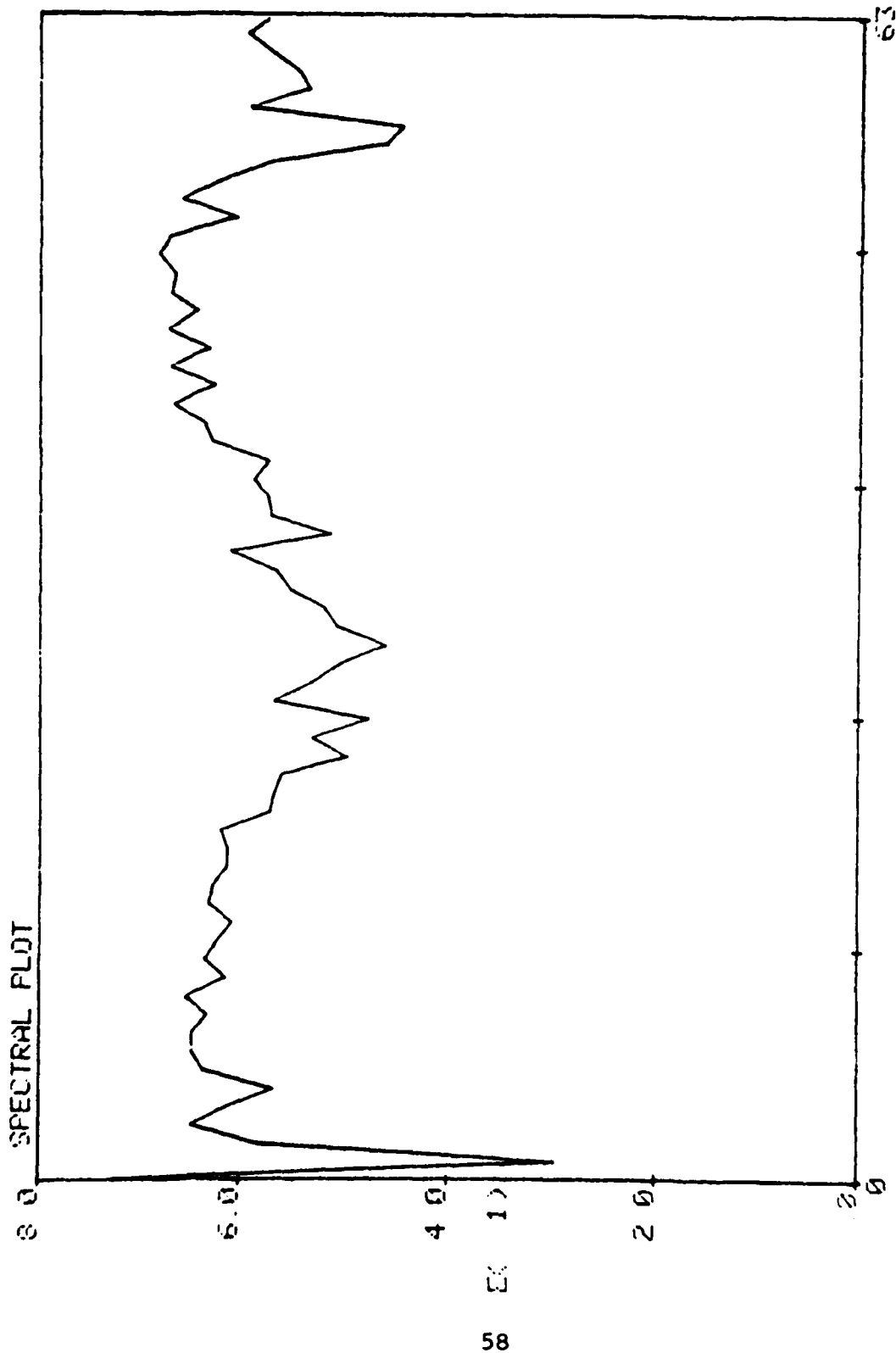


FIGURE 37. Spectral plot observation number 87, 128 point overlapping Hamming window, 10dB dc-emphasis to 300HZ, 10dB pre-emphasis from 500HZ, energy normalized.

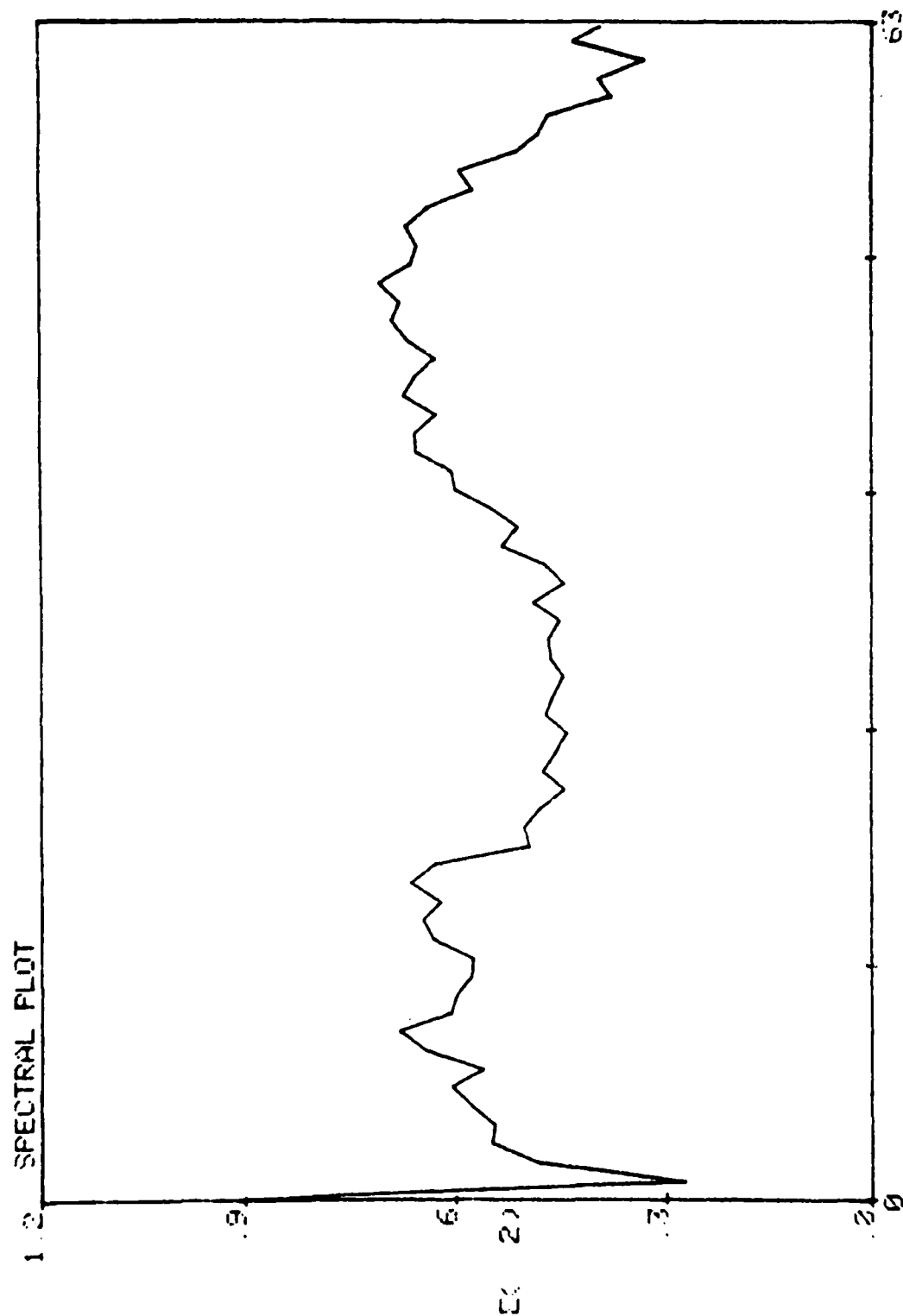


FIGURE 38. Spectral plot observation number 88, 128 point overlapping Hamming window, 10dB de-emphasis to 300HZ, 10dB pre-emphasis from 500HZ, energy normalized.

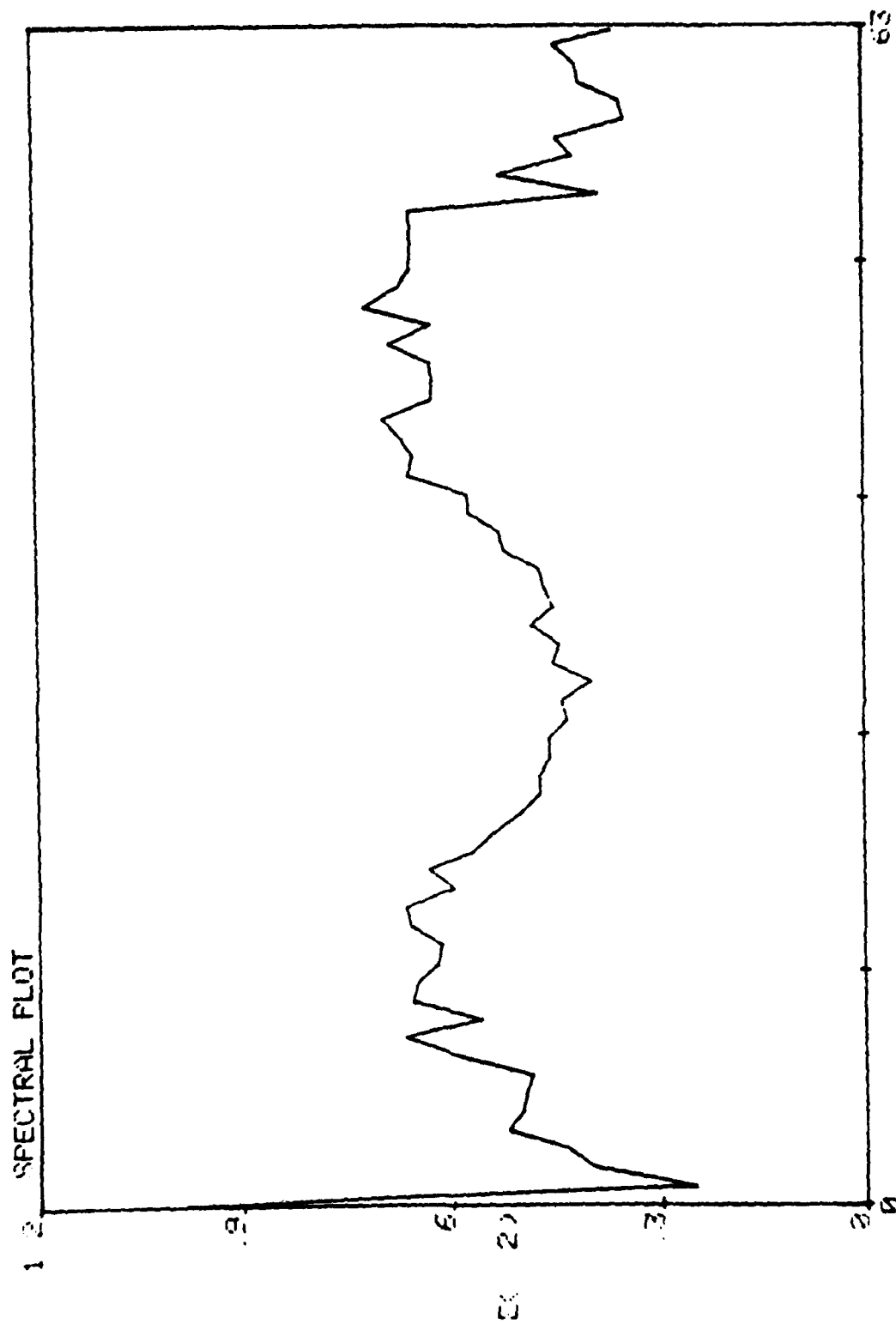


FIGURE 39. Spectral plot observation number 90, 128 point overlapping Hamming window, 10dB de-emphasis to 300HZ, 10dB pre-emphasis from 500HZ, energy normalized.

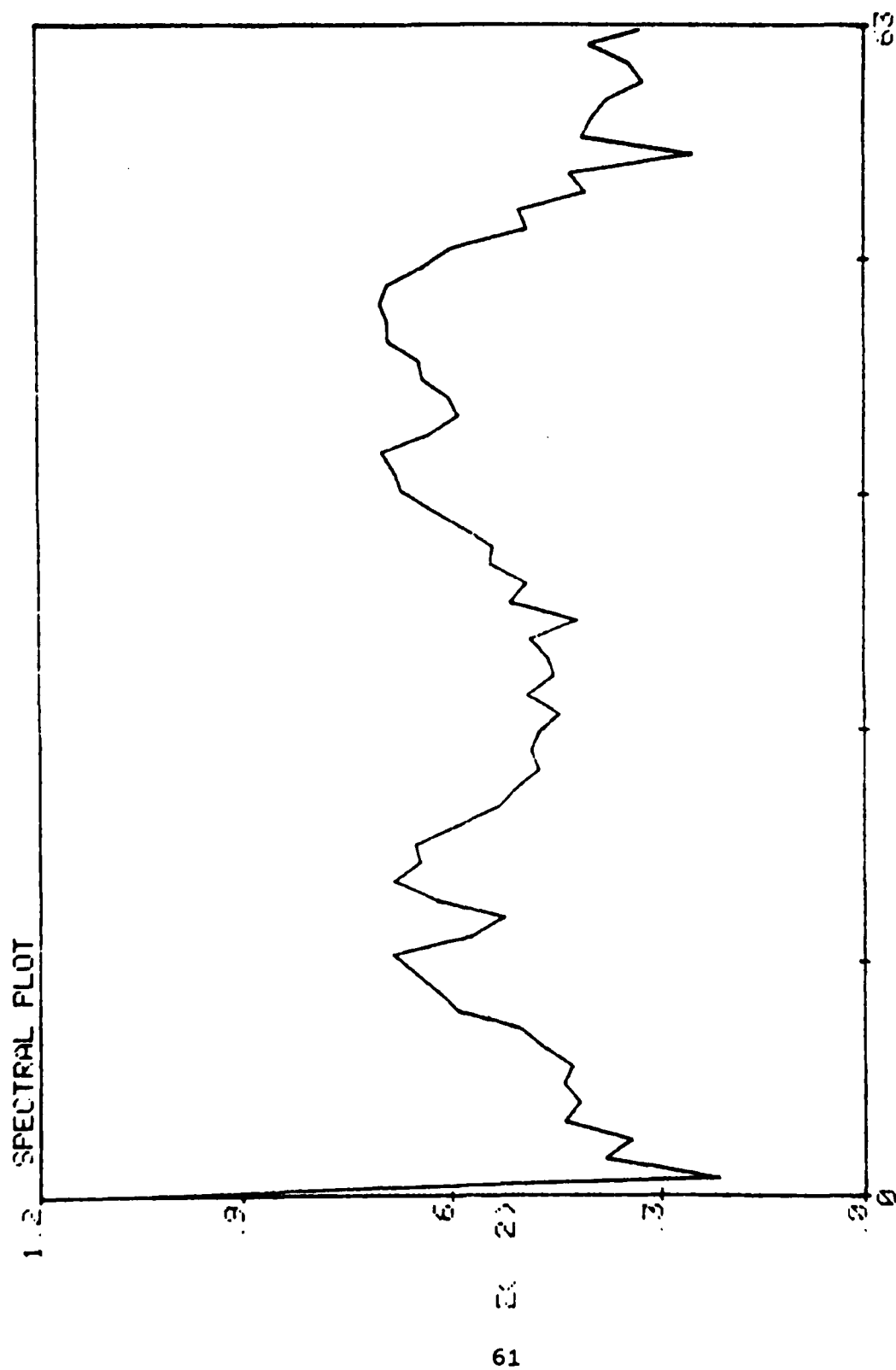


FIGURE 40. Spectral plot observation number 95, 128 point overlapping Hamming window, 10dB de-emphasis to 300HZ, 10dB pre-emphasis from 500HZ, energy normalized.

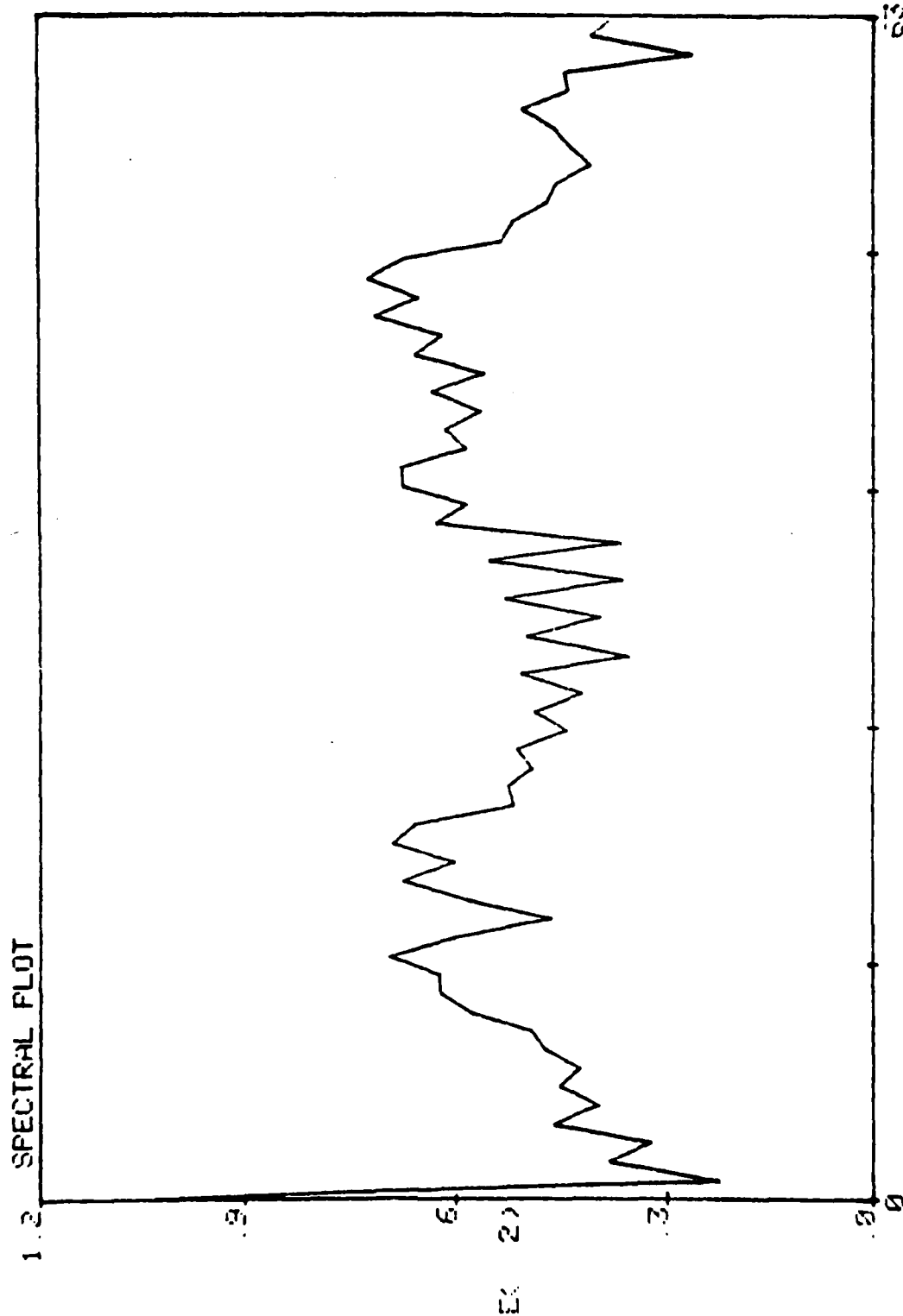


FIGURE 41. Spectral plot observation number 100, 128 point overlapping Hamming window, 10dB de-emphasis to 300HZ, 10dB pre-emphasis from 500HZ, energy normalized.

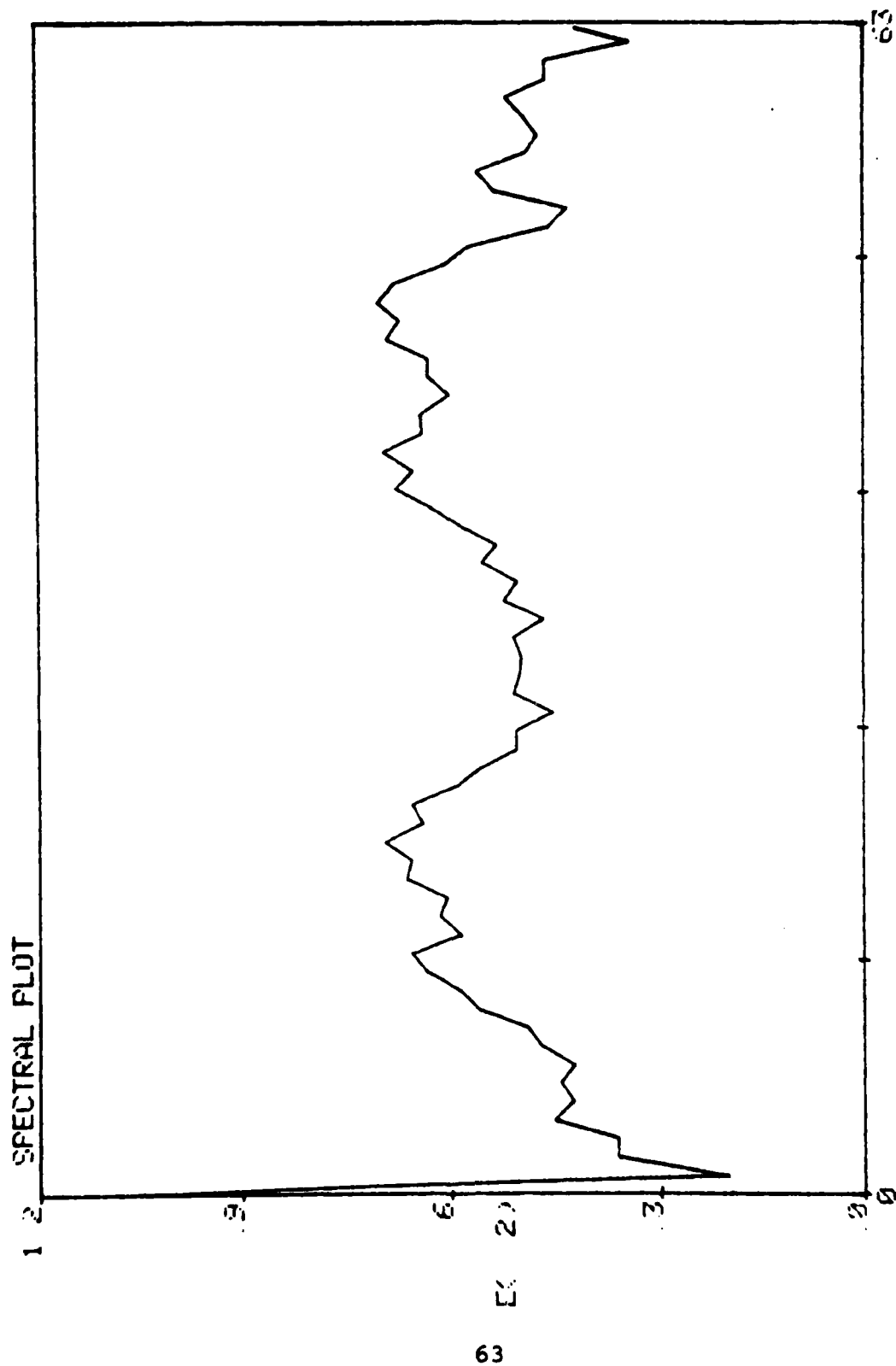


FIGURE 42. Spectral plot observation number 110, 128 point overlapping Hamming window, 10dB de-emphasis to 300HZ, 10dB pre-emphasis from 500HZ, energy normalized.

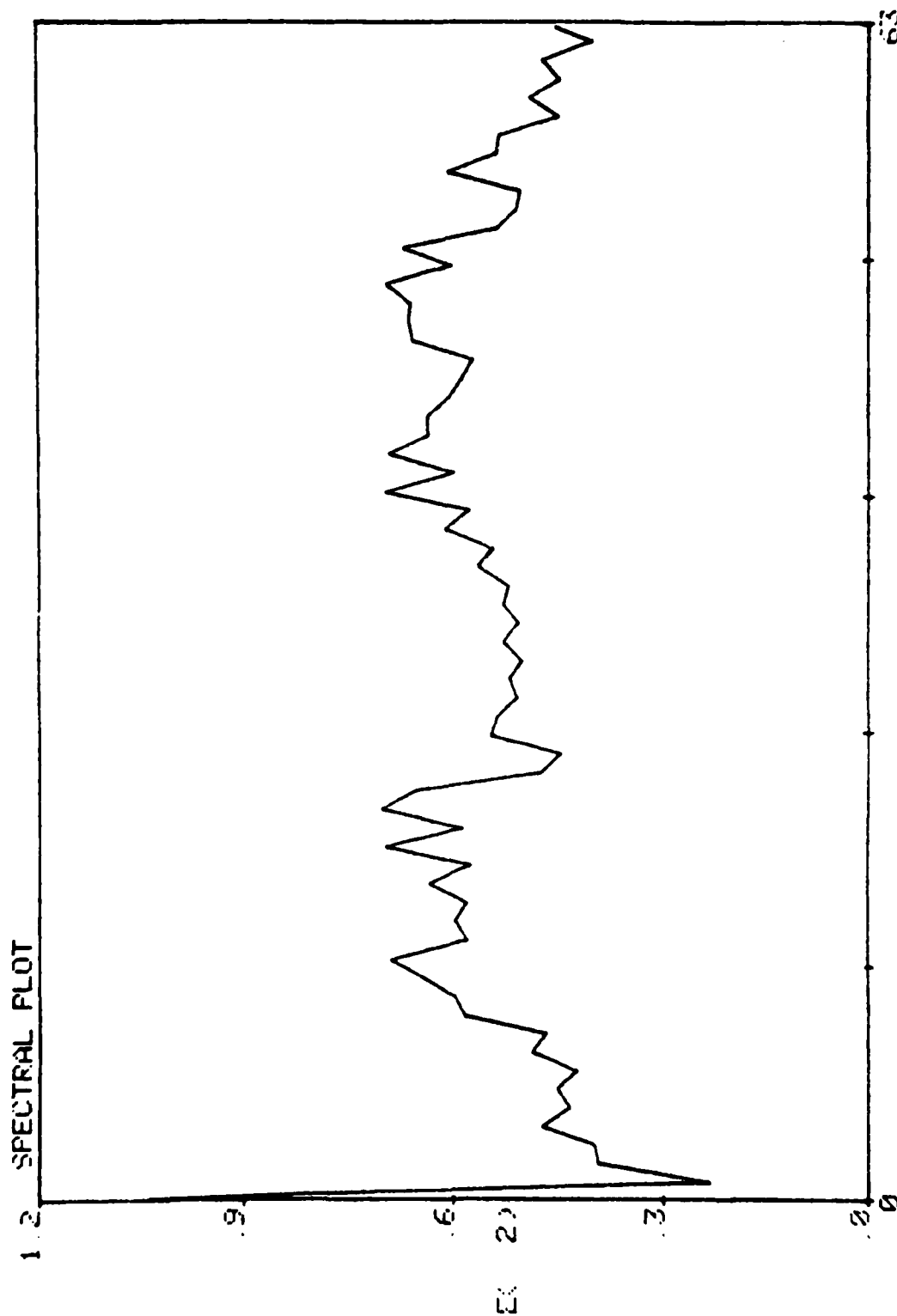


FIGURE 43. Spectral plot observation number 120, 128 point overlapping Hamming window, 10dB de-emphasis to 300Hz, 10dB pre-emphasis from 500Hz, energy normalized.

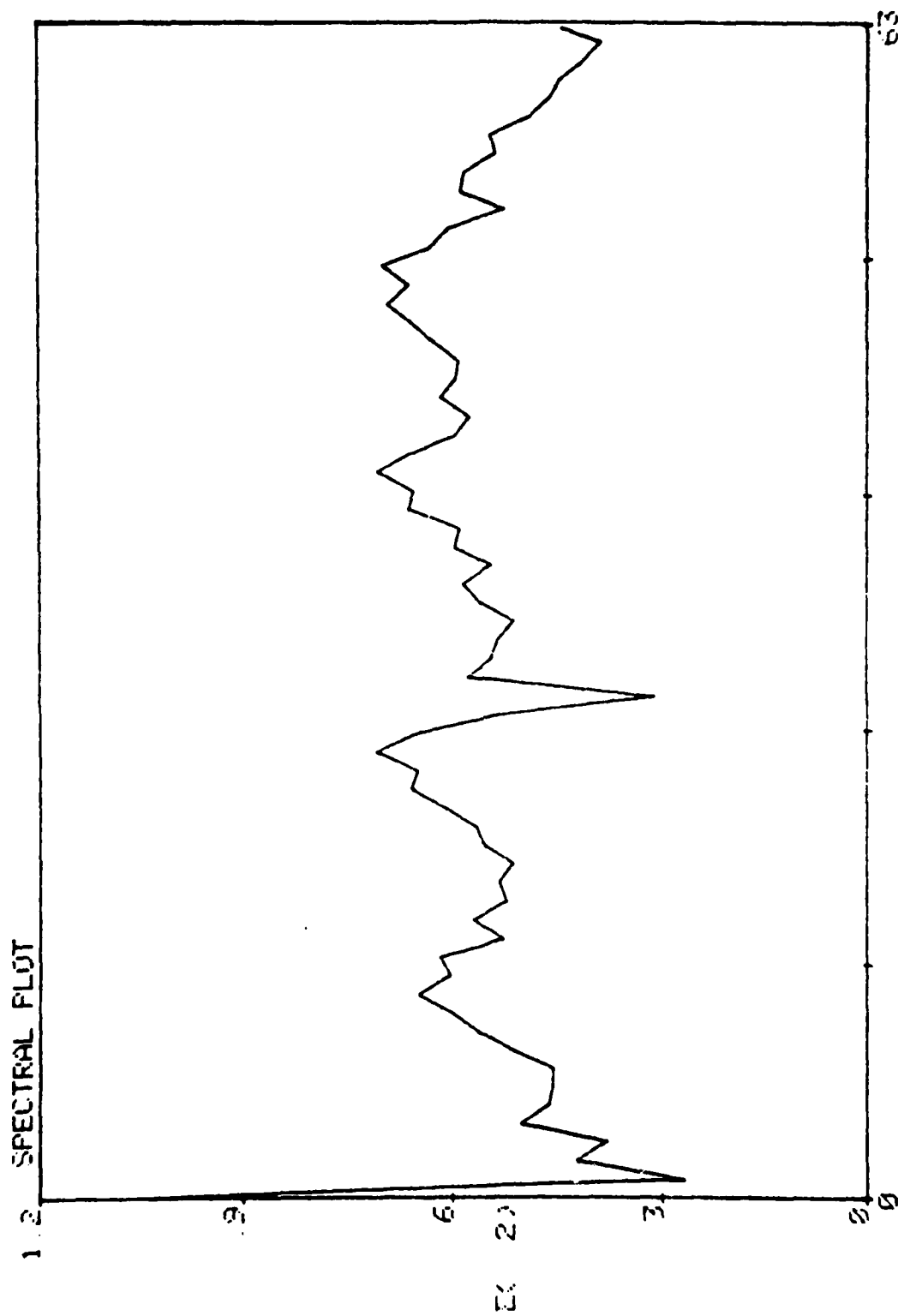


FIGURE 44. Spectral plot observation number 130, 128 point overlapping Hamming window, 10dB de-emphasis to 300HZ, 10dB pre-emphasis from 500HZ, energy normalized.

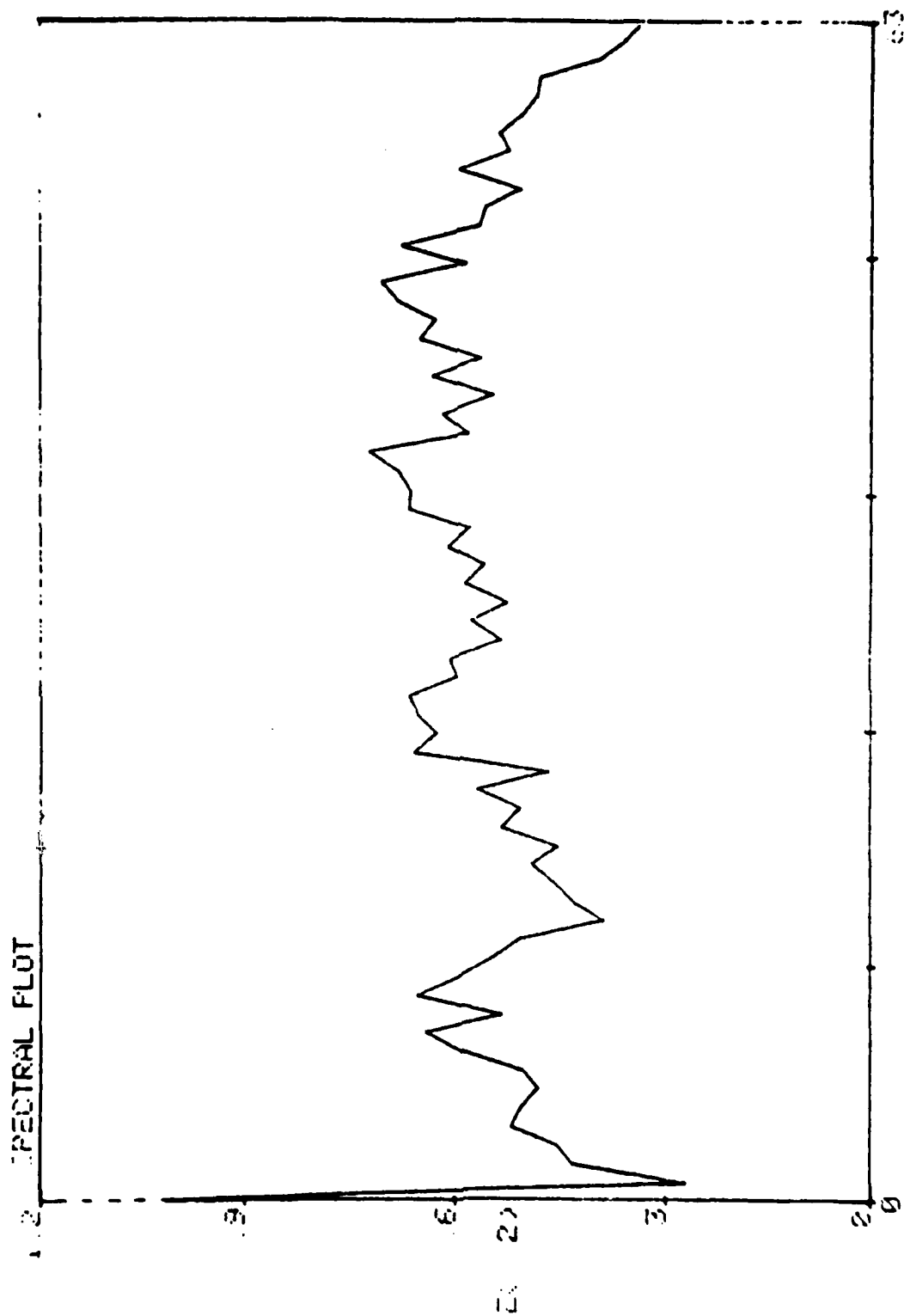


FIGURE 45. Spectral plot observation number 135, 128 point overlapping Hamming window, 10dB de-emphasis to 300HZ, 10dB pre-emphasis from 500HZ, energy normalized.

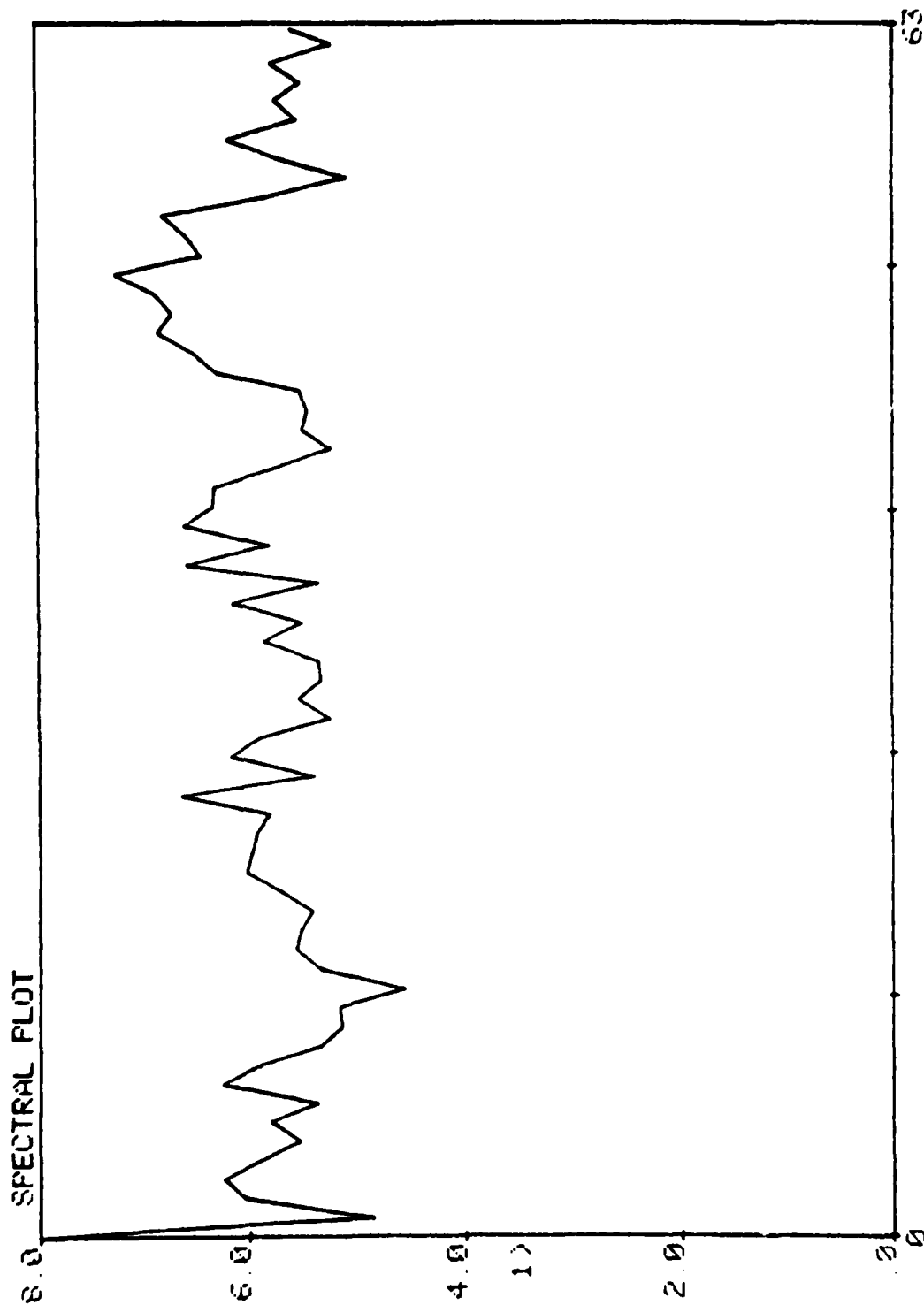


FIGURE 46. Spectral plot observation number 140, 128 point overlapping Hamming window, 10dB de-emphasis to 300HZ, 10dB pre-emphasis from 500HZ, energy normalized.

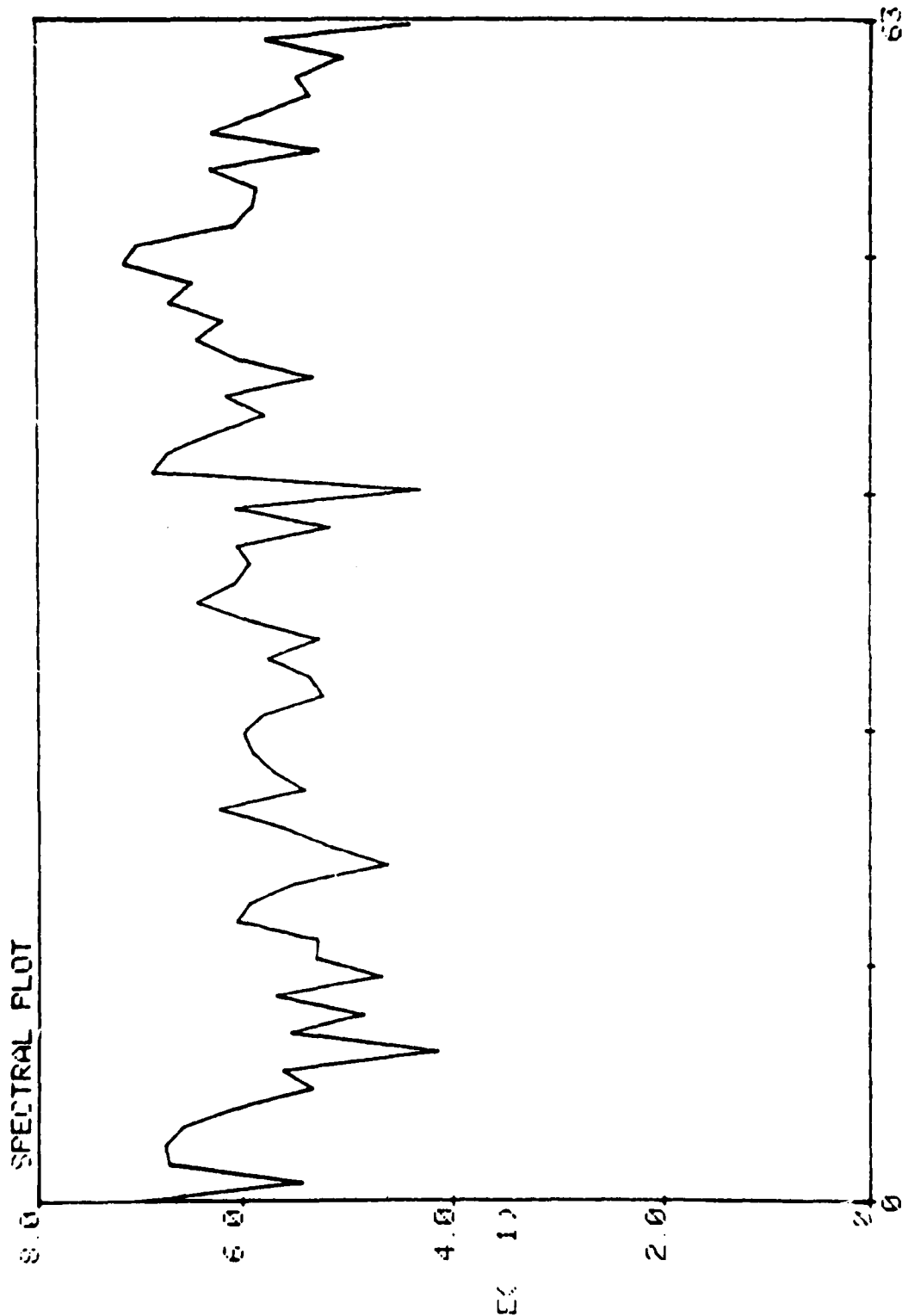


FIGURE 47. Spectral plot observation number 145, 128 point overlapping Hamming window, 10dB de-emphasis to 500Hz, 10dB pre-emphasis from 500Hz, energy normalized.

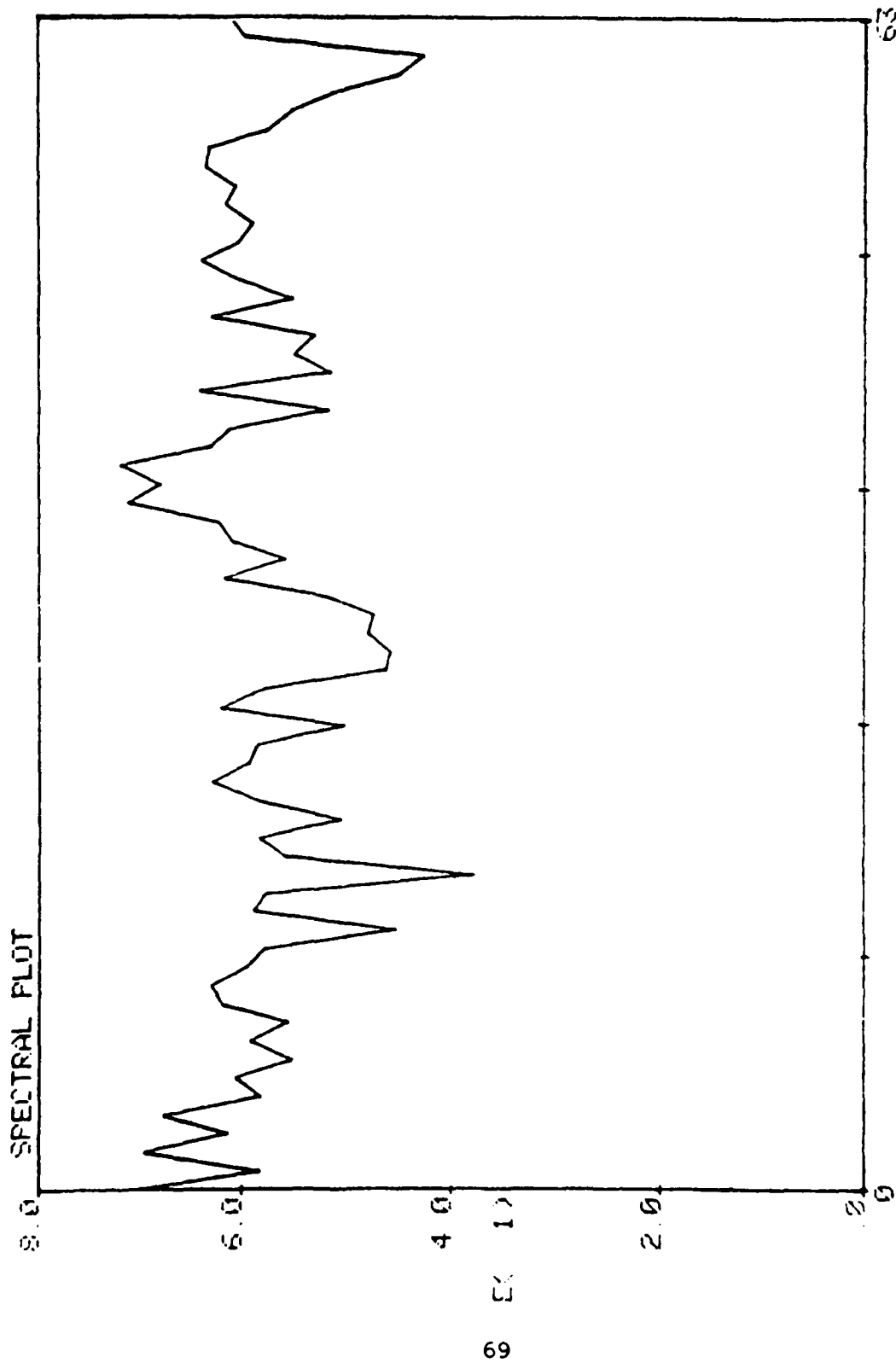


FIGURE 48. Spectral plot observation number 150, 128 point overlapping Hamming window, 10dB de-emphasis to 300HZ, 10dB pre-emphasis from 500HZ, energy normalized.

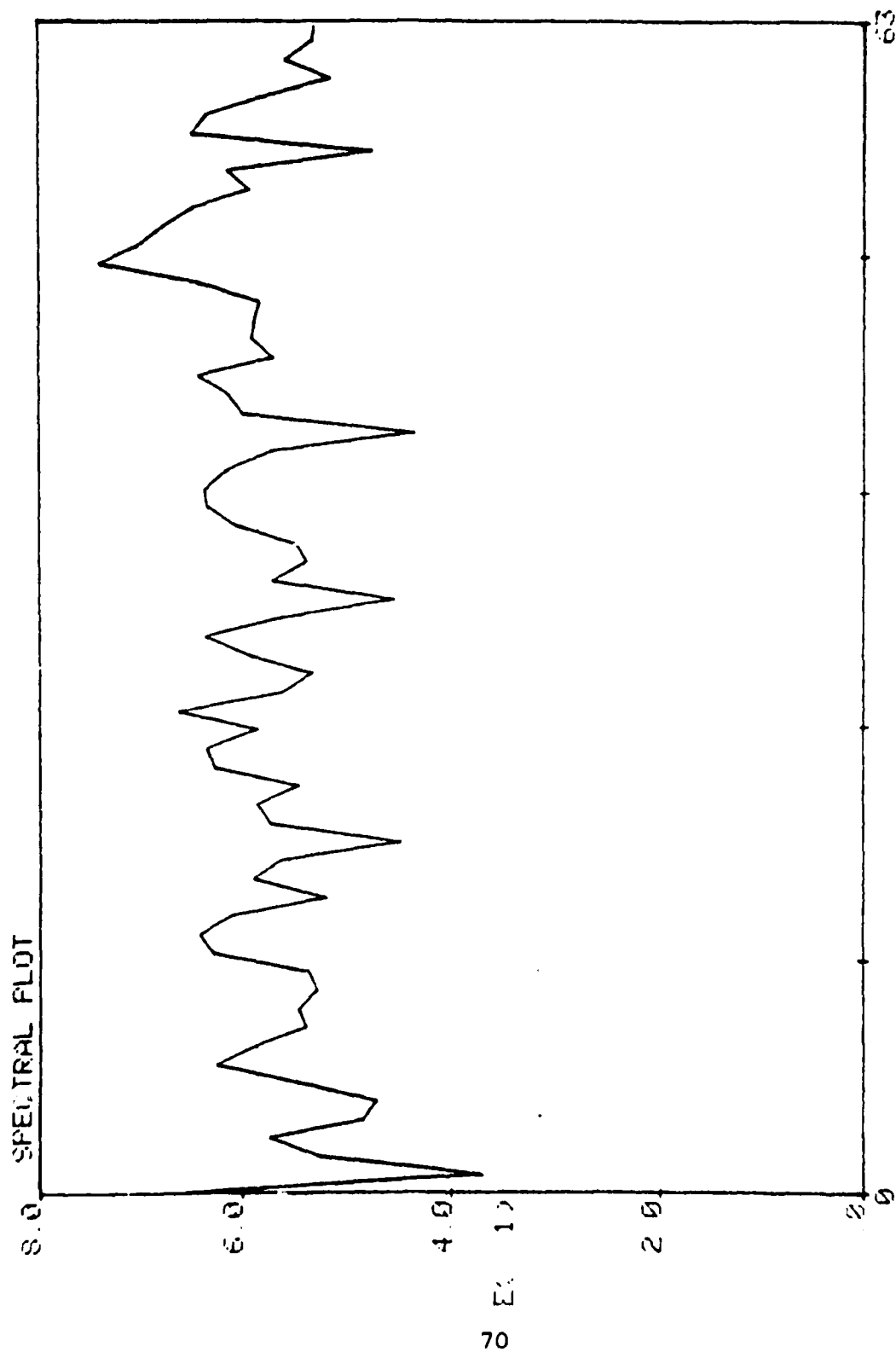


FIGURE 49. Spectral plot observation number 160, 128 point overlapping Hamming window, 10dB de-emphasis to 300Hz, 10dB pre-emphasis from 500Hz, energy normalized.

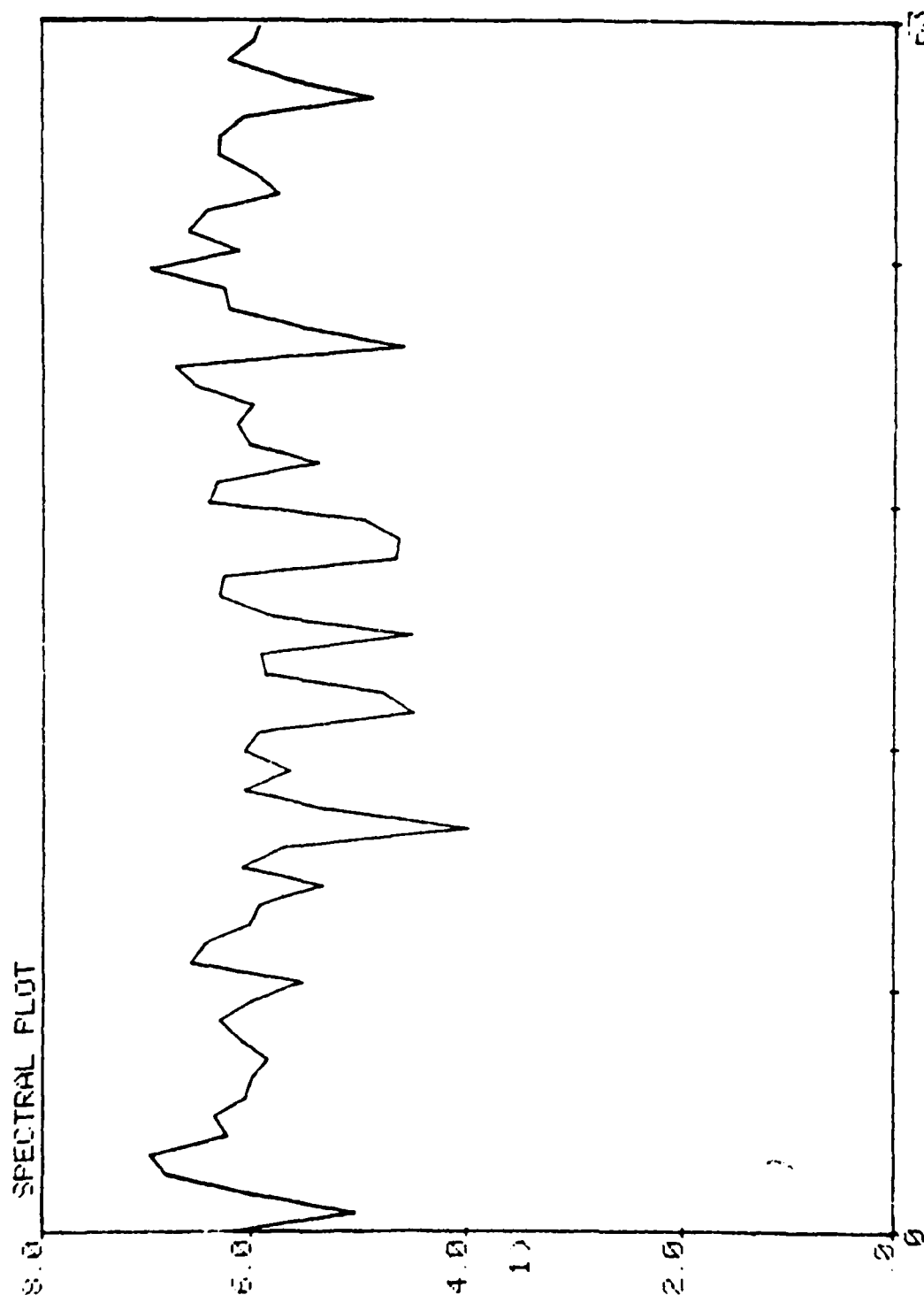


FIGURE 50. Spectral plot observation number 170, 128 point overlapping Hamming window, 10dB de-emphasis to 300HZ, 10dB pre-emphasis from 500HZ, energy normalized.

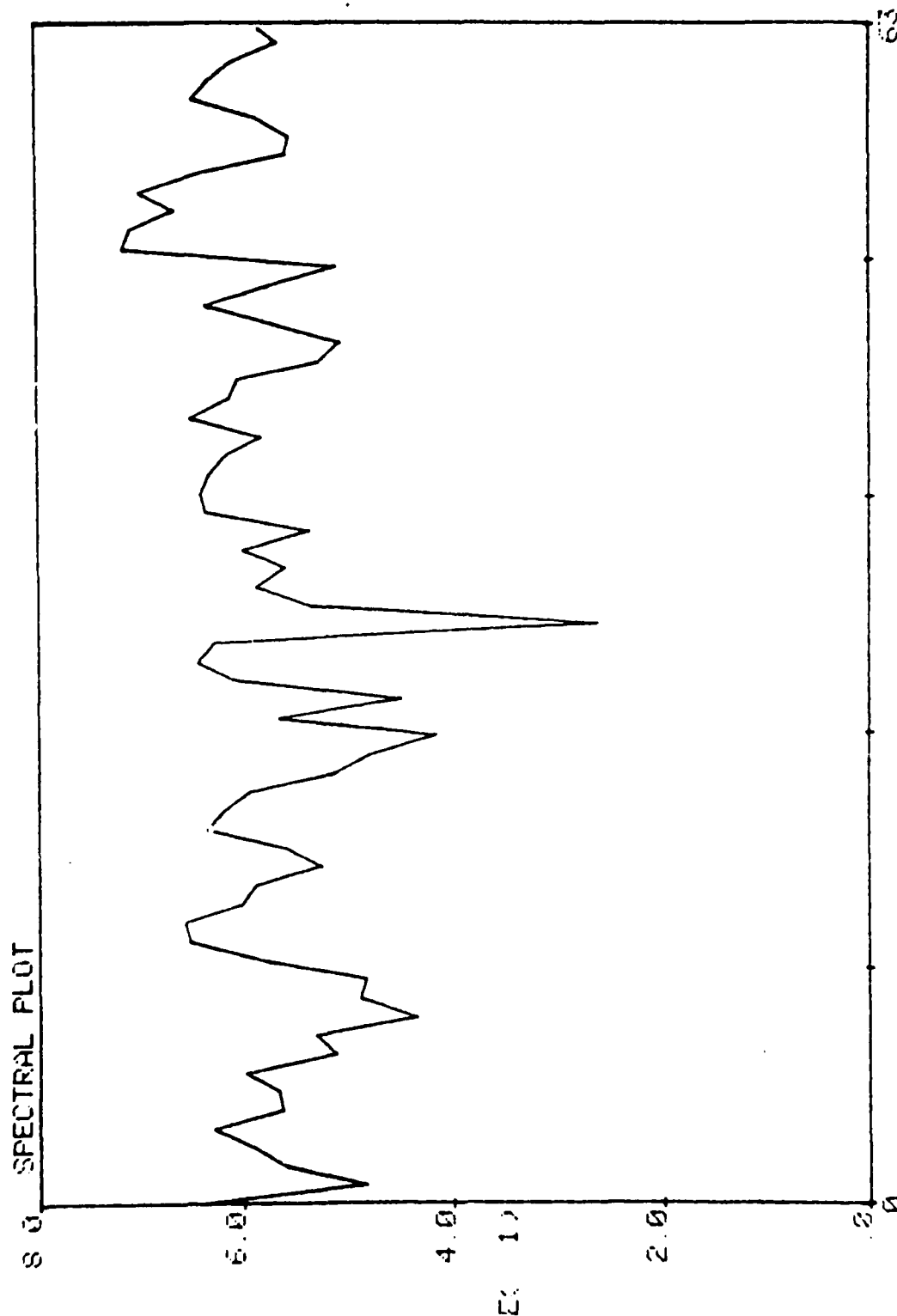


FIGURE 51. Spectral plot observation number 180, 128 point overlapping Hamming window, 10dB de-emphasis to 300HZ, 10dB pre-emphasis from 500HZ, energy normalized.

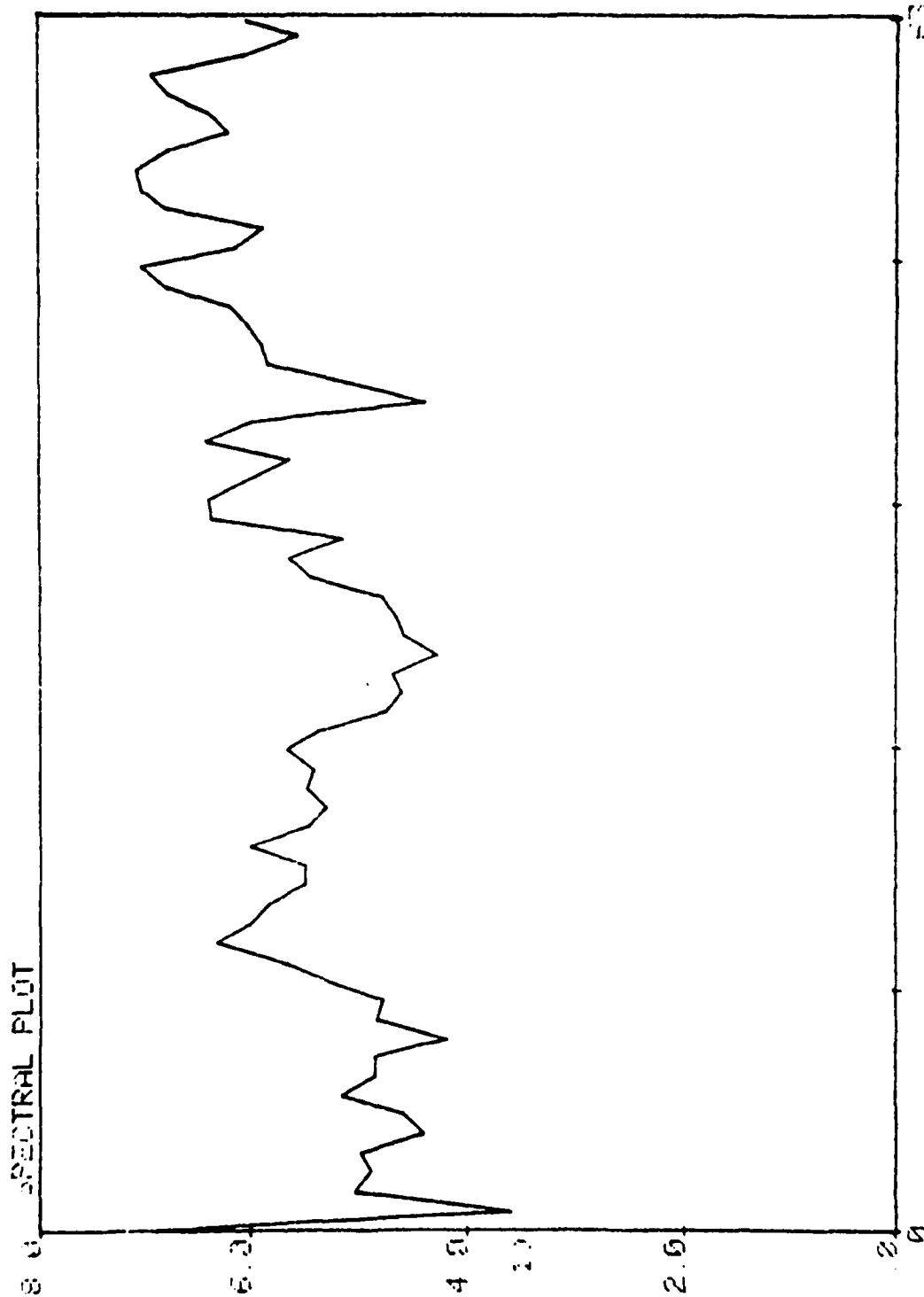


FIGURE 52. Spectral plot observation number 190, 128 point overlapping Hamming window, 10dB de-emphasis to 300HZ, 10dB pre-emphasis from 500HZ, energy normalized.

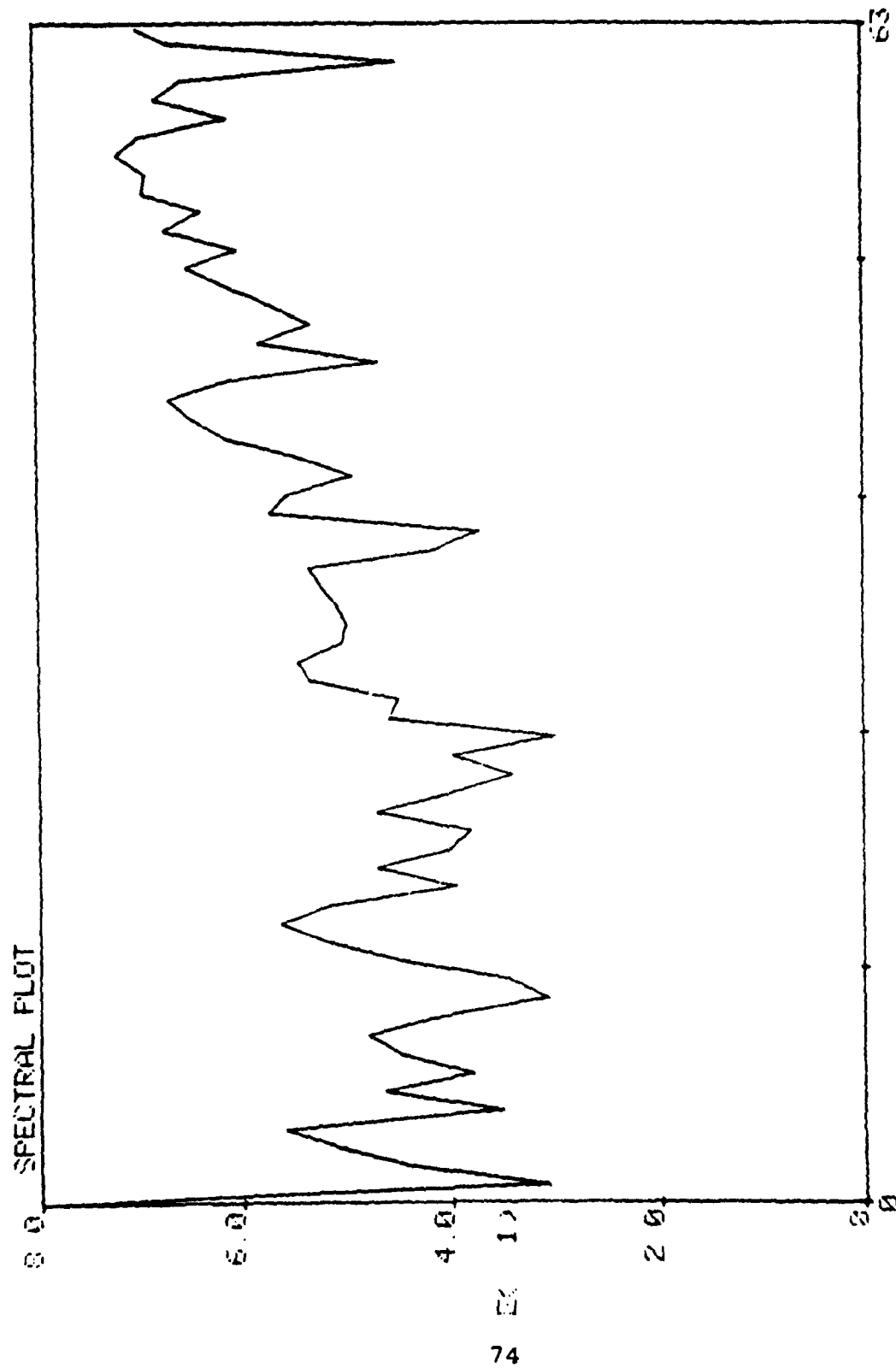


FIGURE 53. Spectral plot observation number 195, 128 point overlapping Hamming window, 10dB de-emphasis to 300Hz, 10dB pre-emphasis from 500Hz, energy normalized.

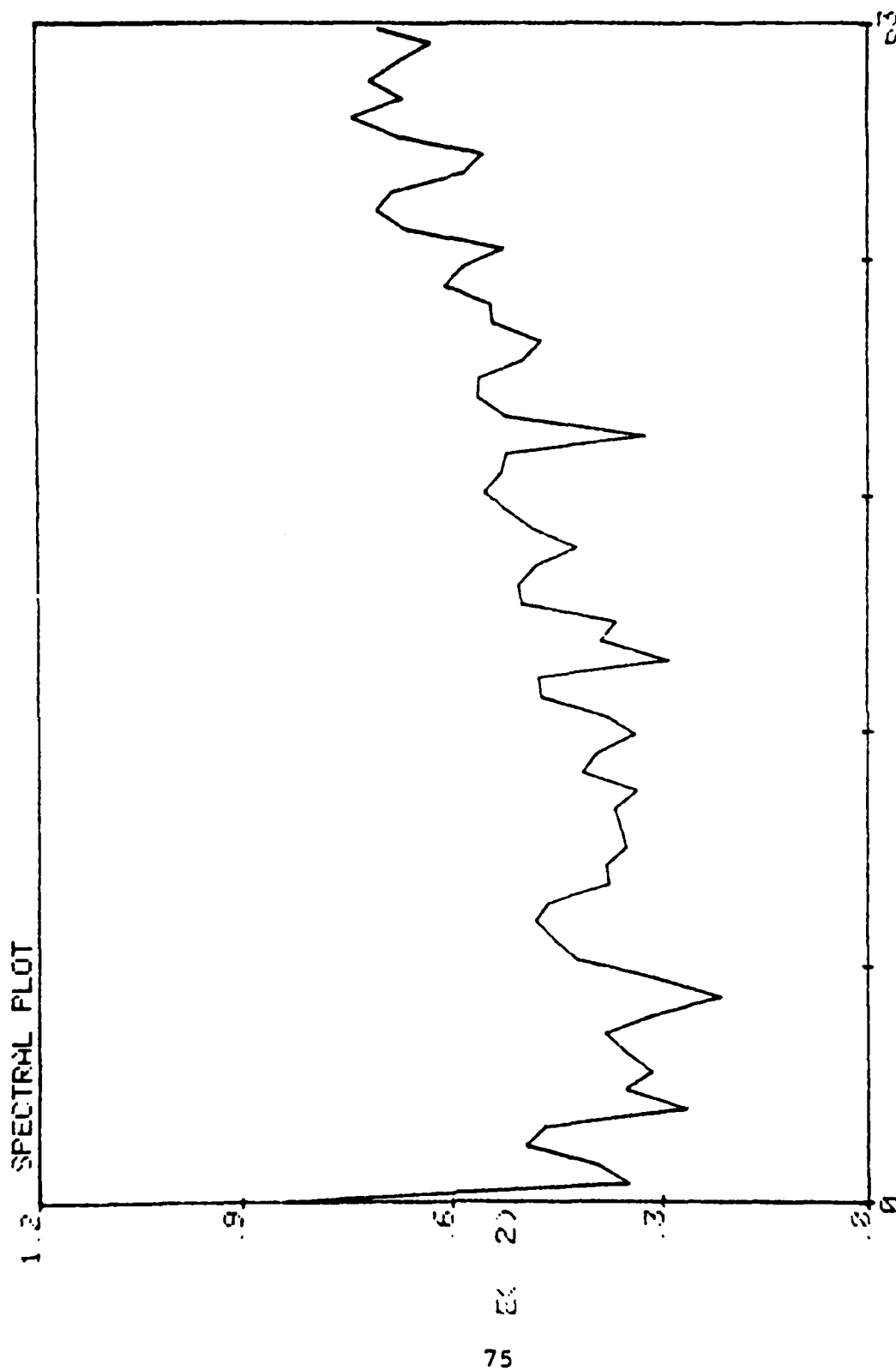


FIGURE 54. Spectral plot observation number 200, 128 point overlapping Hamming window, 10dB de-emphasis to 300Hz, 10dB pre-emphasis from 500Hz, energy normalized.

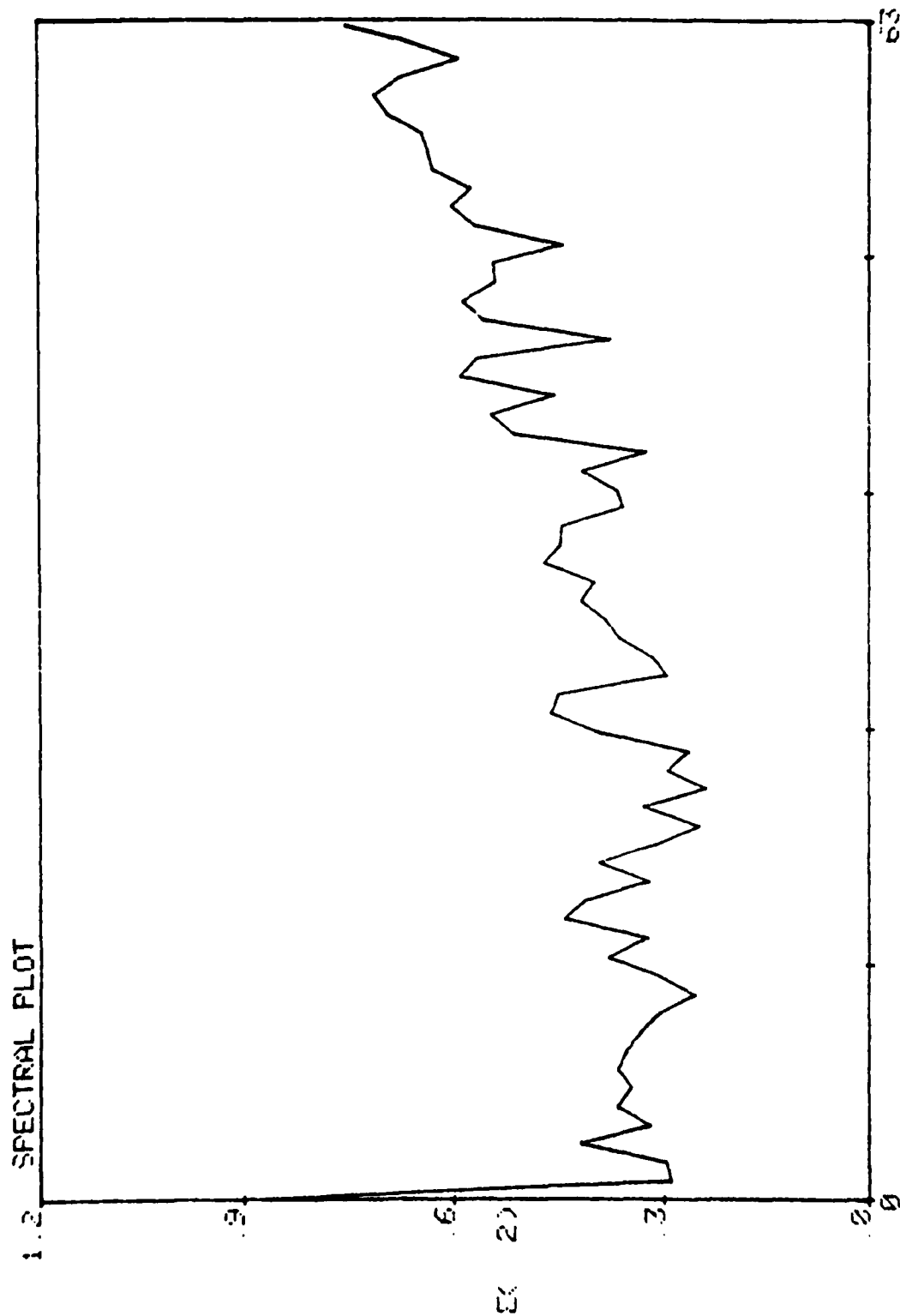


FIGURE 55. Spectral plot observation number 201, 128 point overlapping Hamming window, 10dB de-emphasis to 300HZ, 10dB pre-emphasis from 500HZ. energy normalized.

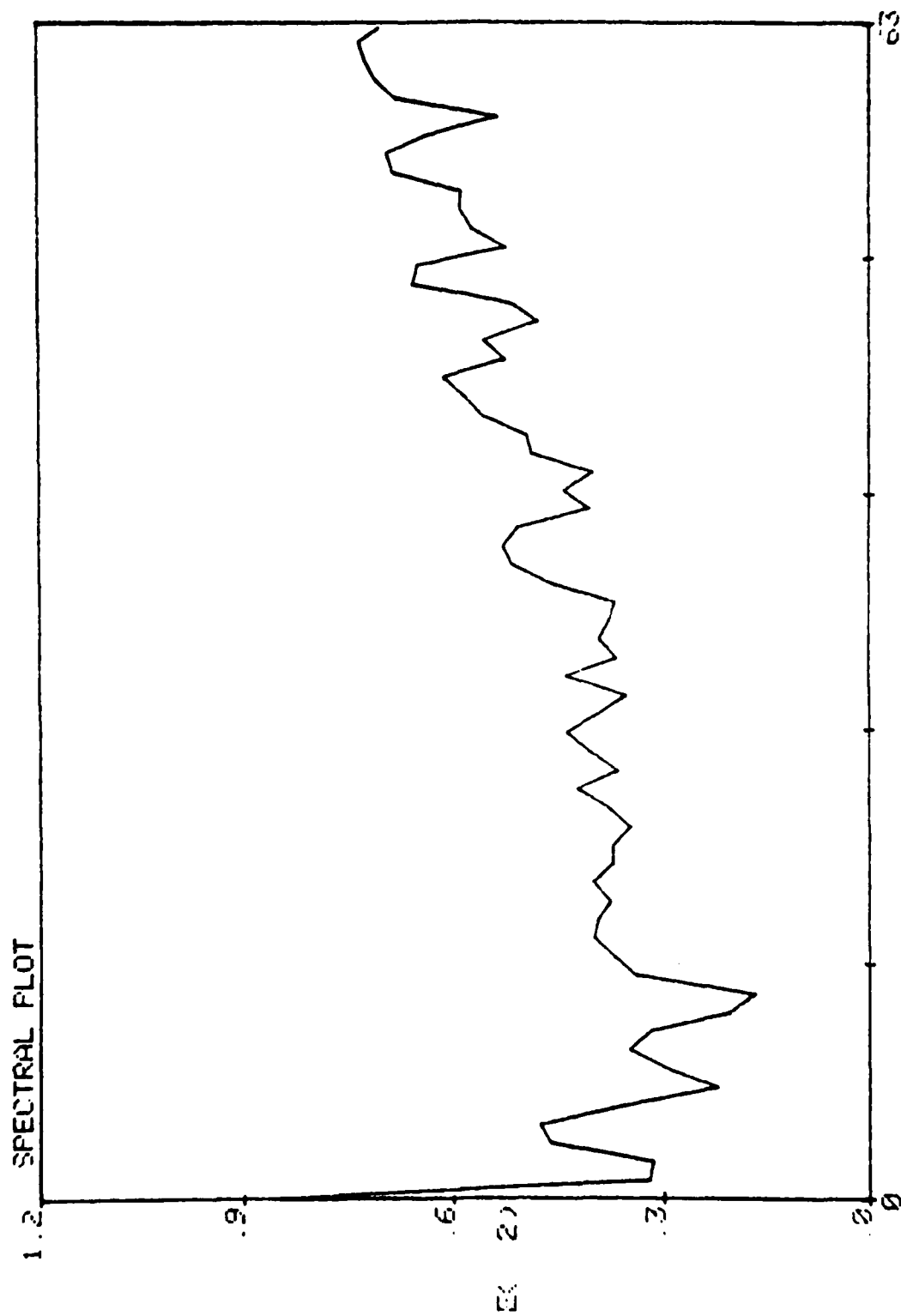


FIGURE 56. Spectral plot observation number 202, 128 point overlapping Hamming window, 10dB de-emphasis to 300HZ, 10dB pre-emphasis from 500HZ, energy normalized.

feature space will cluster for unvoiced sounds and noise sources. It can be argued, however, that success of a scheme for detecting phonetic units in noise will depend not so much on an energy threshold, but on how well phonets representing noise and speech separate, and on noise cancellation techniques employed. Several clustering algorithms (Ref 13) will need to be studied, and perhaps implemented, to quantify phonet detection in noise. A cluster algorithm is presented in the next section. But it seems too early, at this point, to settle on a threshold criterion such as Seelandt's.

A reliable discriminant between unvoiced speech and noise may be one based on clustering properties of observations about selected phonets. We would not expect distances from a white noise background model to cluster, but nonwhite noise may.

The distribution of similar observations in several speech segments is plotted in Figures 57-68. In each plot, M2 distances are plotted from a particular observation to each observation in the speech file. The smallest five and the largest distances are plotted along with the energy of the particular observation. The position along the horizontal axis at which a distance is plotted corresponds to the number of the observation to which that distance was computed. One of the minimum distances will be zero corresponding to a perfect match. The energy value is plotted in position zero.

Figures 57 and 58 plot the behavior of two observations from the nonspeaker region before the word "five" and illustrate the lack of clustering characteristics of low energy speakerless regions. Figures 63-65 plot distance choices from observations in the speakerless segment between the words "five" and "six" and again show the typical lack of clustering.

Figures 59 and 60 plot distance choices for observations in the "f" segment and in the "fi" segment, respectively. Observations from the "f" sound of this speaker did not cluster well; but, as the voiced "i" sound came in, distance choices clustered much tighter. Characteristic of the tight cluster in high energy vowel sounds is the plot in Figure 62. We were surprised to find that distance choices from observations in "s" segments clustered tightly. Figures 66, 67, and 68 plot choices from observations in the "s" sound in the word "six." From a spectrogram of the speech segment, we could see that these observations precede a strong "i" influence, and spectral plots from the region, Figures 54, 55, and 56, do not show the characteristic "i" structure. These are "s" sounds of moderate energy content which cluster.

Thus far in this chapter, we have presented the Acoustic Analyzer and illustrated observations and distances computed using values we have assigned to six parameters. We have discussed these parameters in both

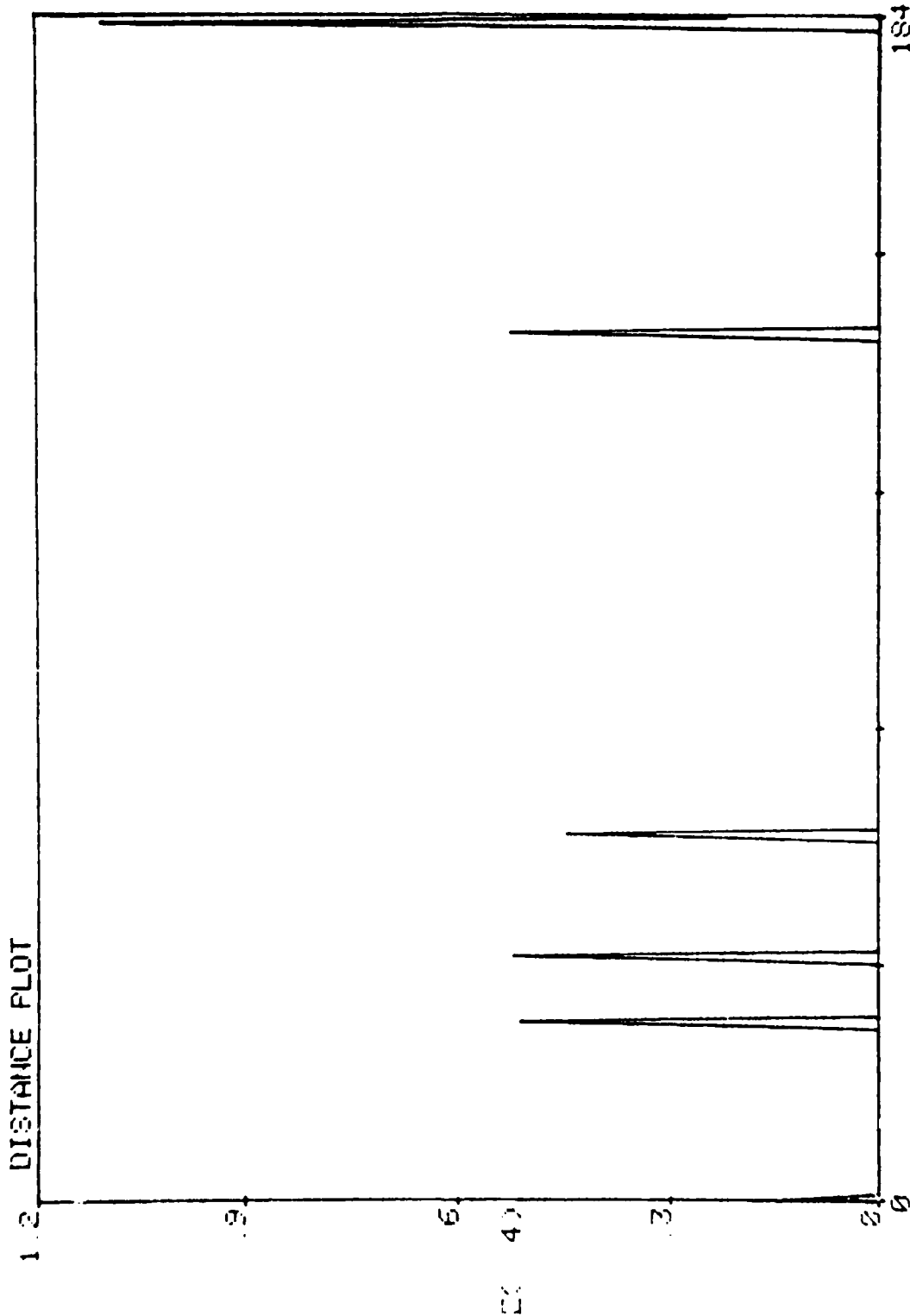


FIGURE 57. Five-Best Distances. Plot of observation energy in ordinate position zero, maximum and five-smallest distances from observation number 21, N2 rule.

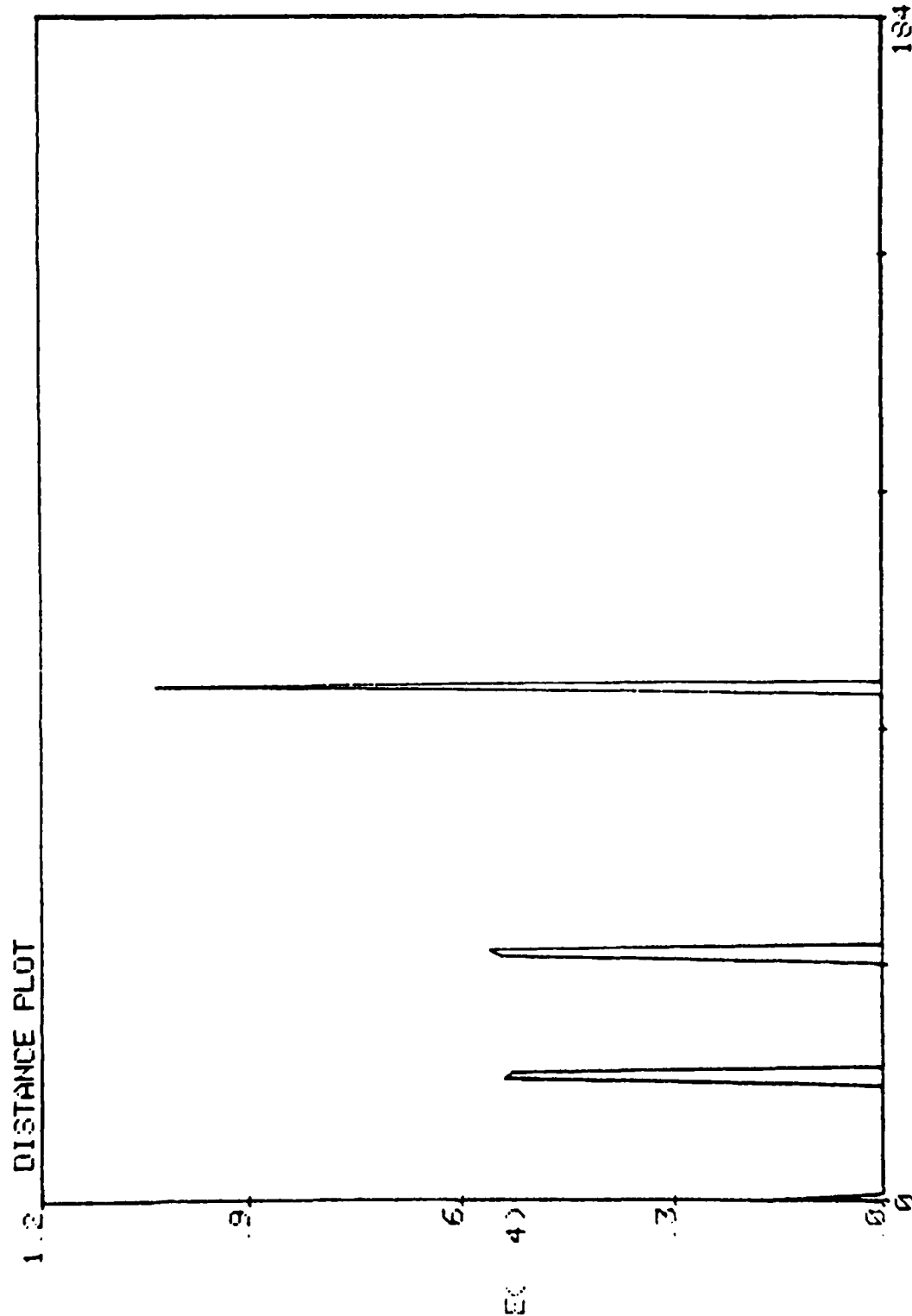


FIGURE 58. Five-Best Distances. Plot of observation energy in ordinate position zero, maximum and five-smallest distances from observation number 31, M2 rule.

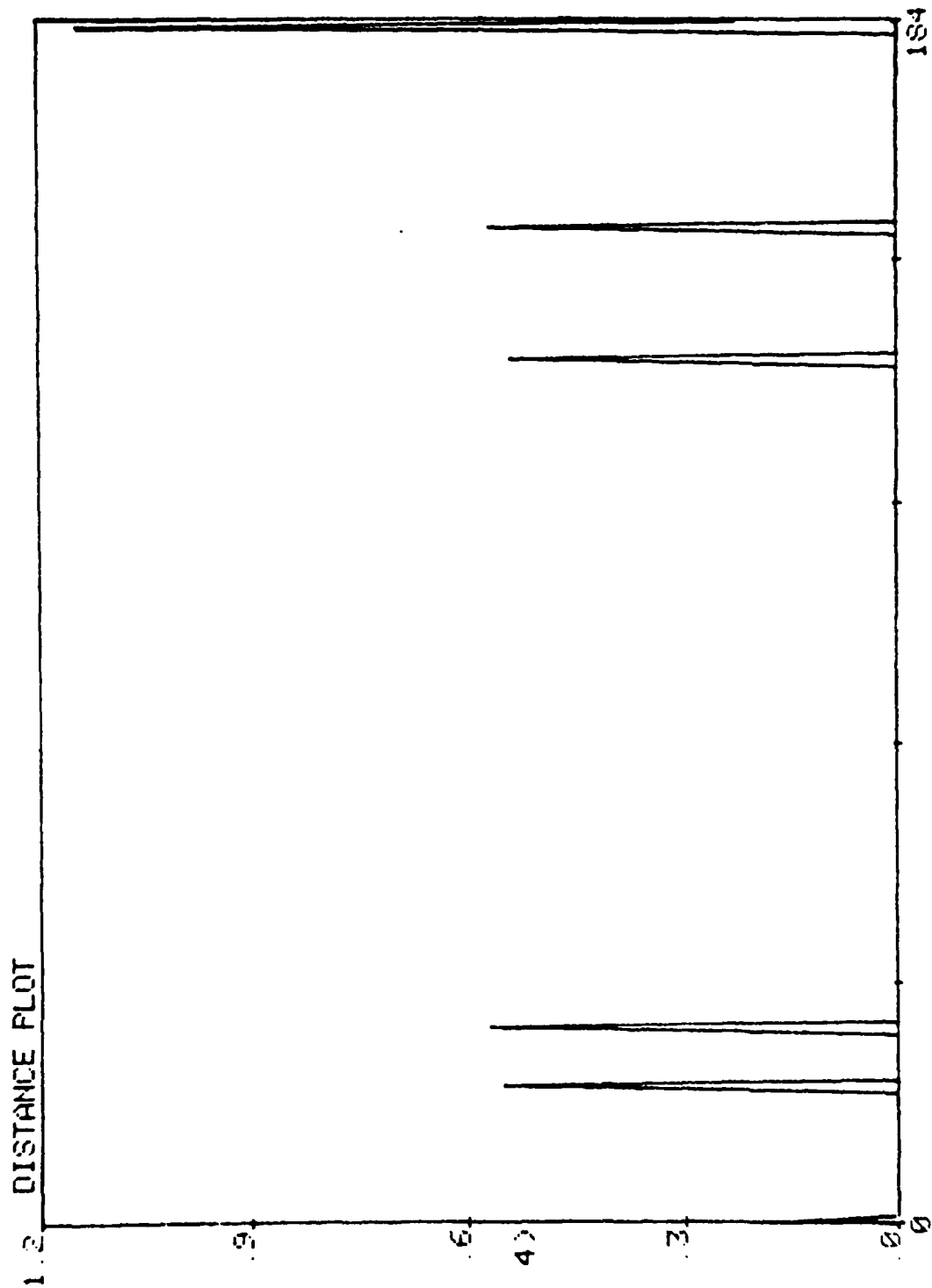


FIGURE 59. Five-Best Distances. Plot of observation energy in ordinate position zero, maximum and five-smallest distances from observation number 81 , N2 rule.

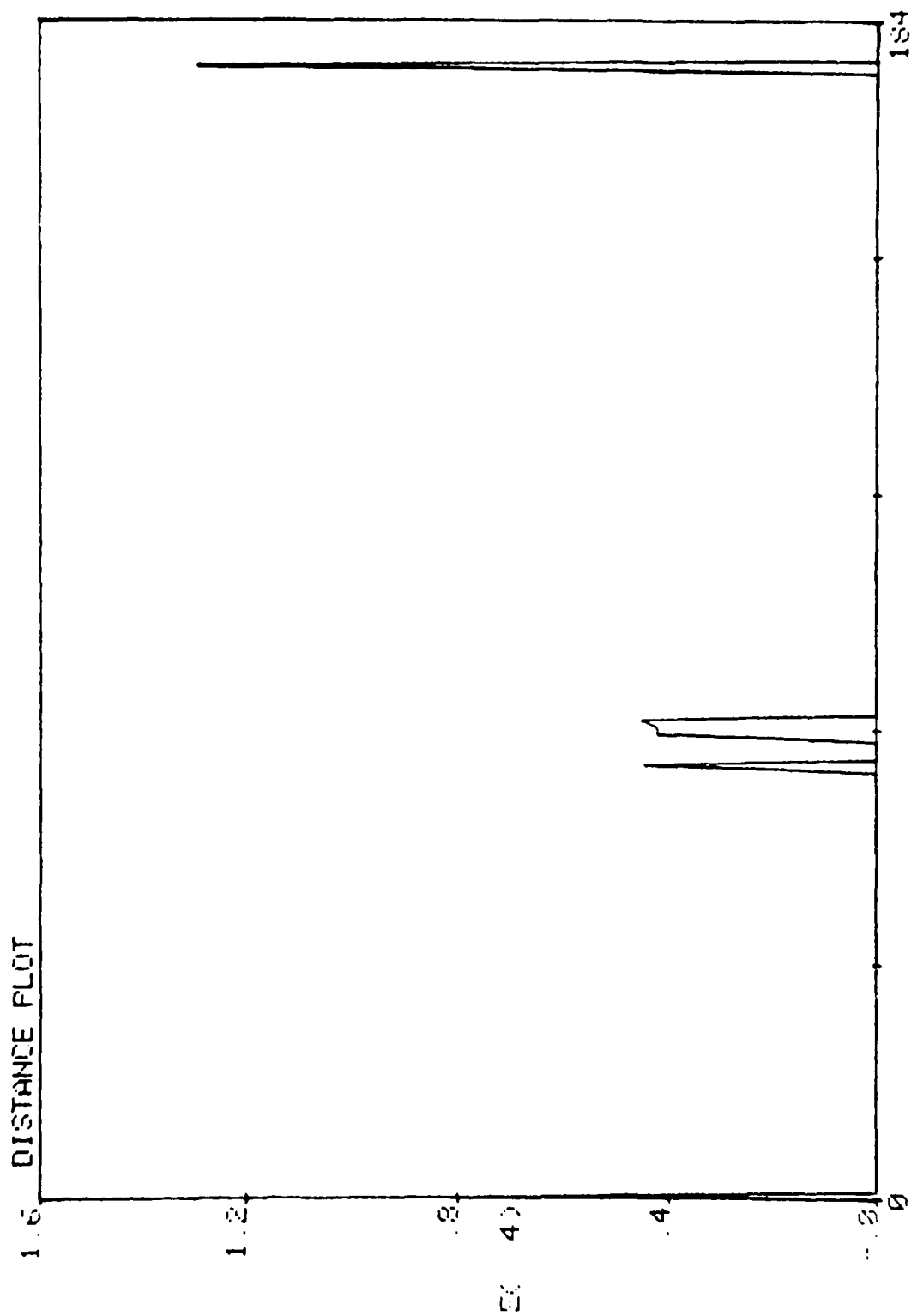


FIGURE 60. Five-Best Distances. Plot of observation energy in ordinate position zero, maximum and five-smallest distances from observation number 91, M2 rule.

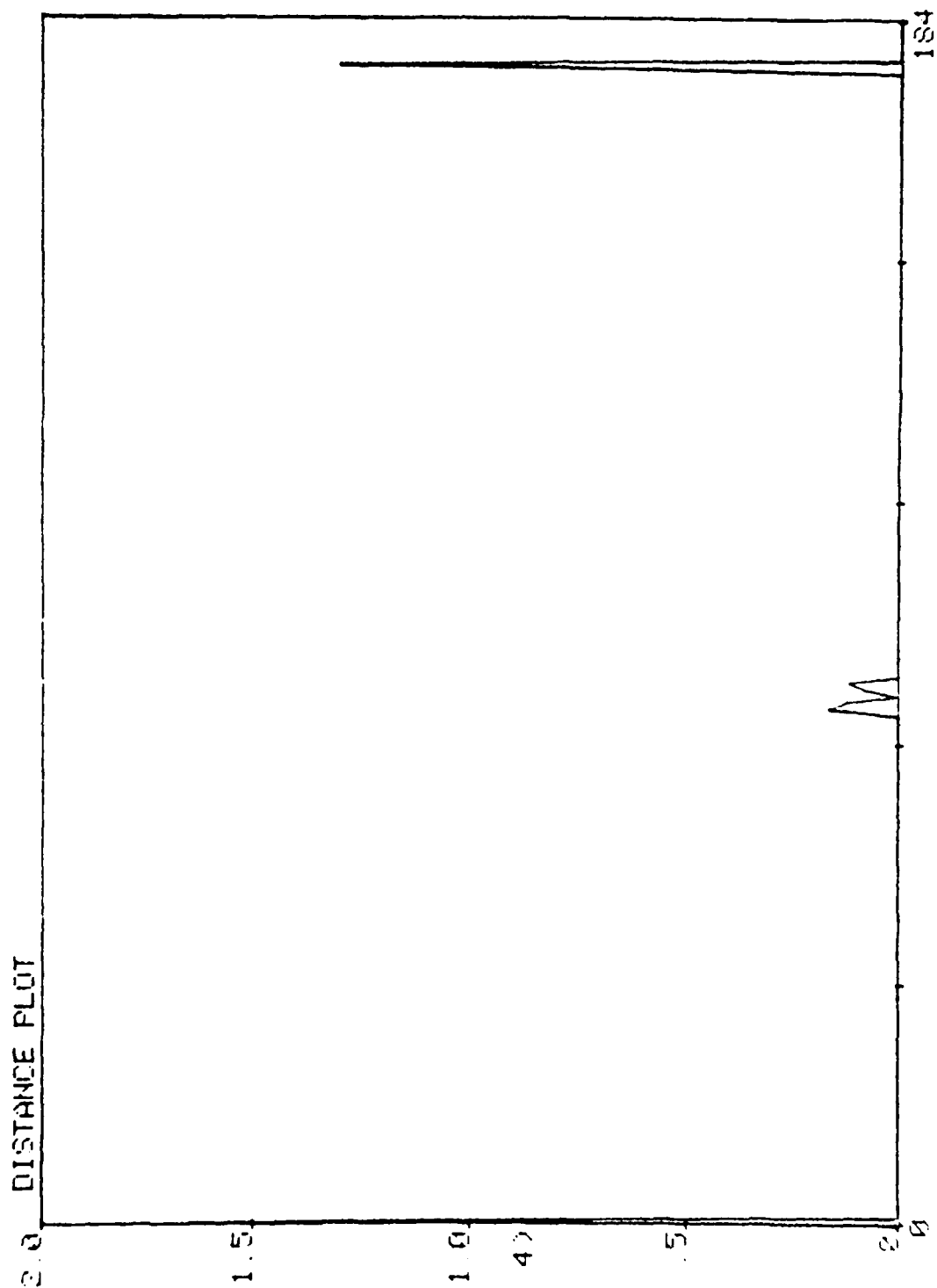


FIGURE 61. Five-Best Distances. Plot of observation energy in ordinate position zero, maximum and five-smallest distances from observation number 101, M2 rule.

AD-A125 360

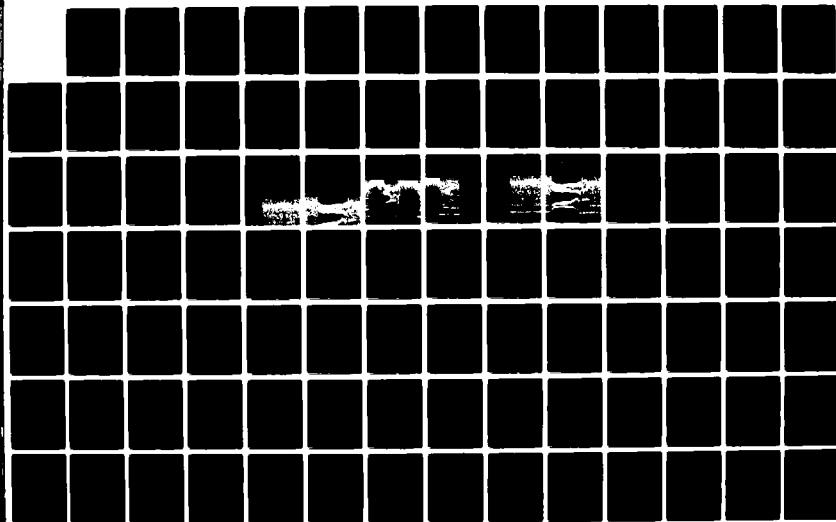
COMPUTER RECOGNITION OF PHONETS IN SPEECH (U) AIR FORCE  
INST OF TECH WRIGHT-PATTERSON AFB OH SCHOOL OF  
ENGINEERING D L MARTIN DEC 82 AFIT/GE/EE/82D-46

23

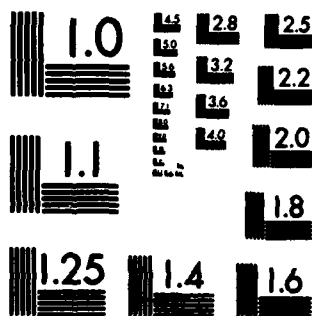
UNCLASSIFIED

F/G 9/2

NL



M-2



MICROCOPY RESOLUTION TEST CHART  
NATIONAL BUREAU OF STANDARDS-1963-A

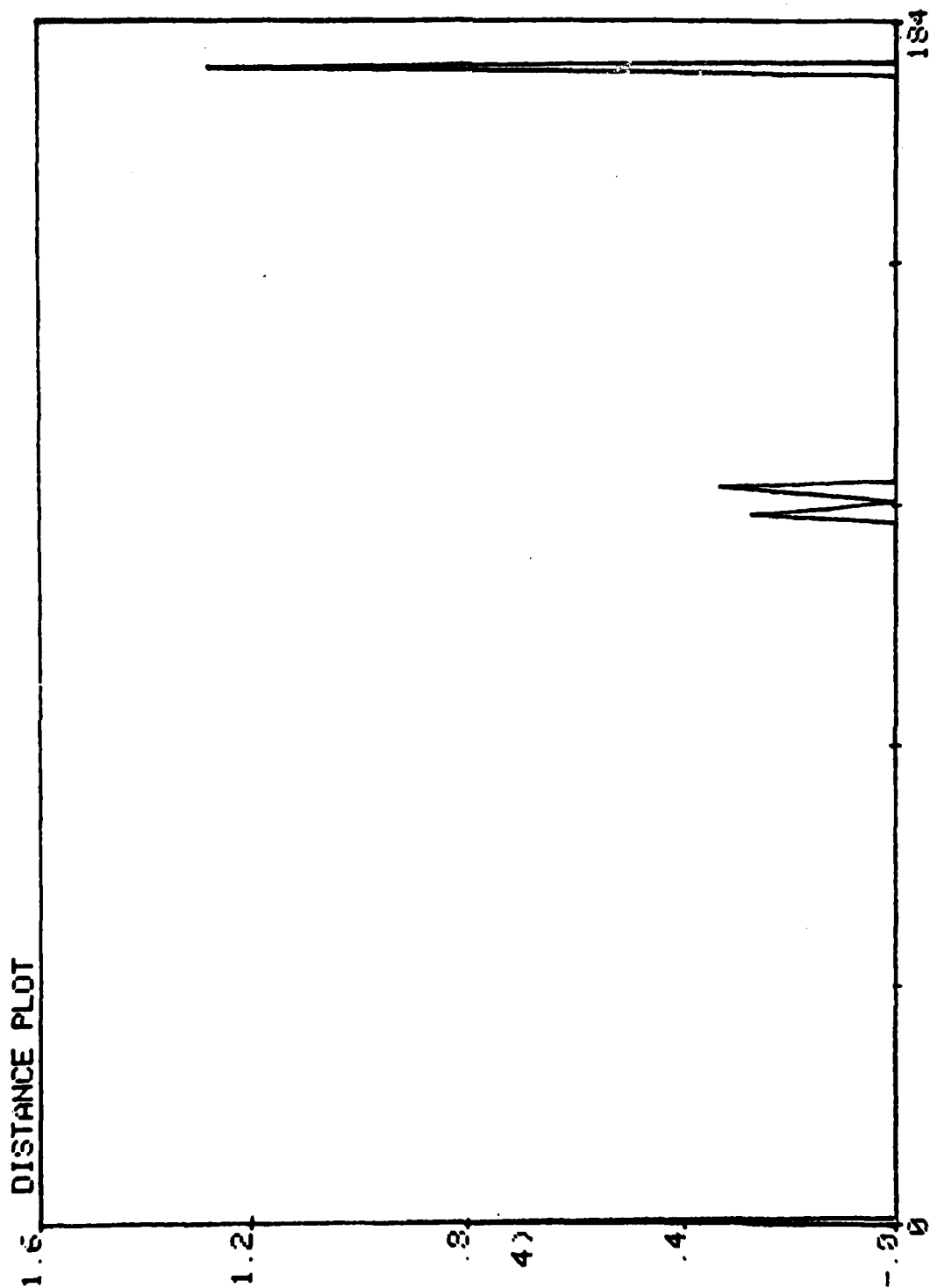


FIGURE 62. Five-Best Distances. Plot of observation energy in ordinate position zero, maximum and five-smallest distances from observation number 131, M2 rule.

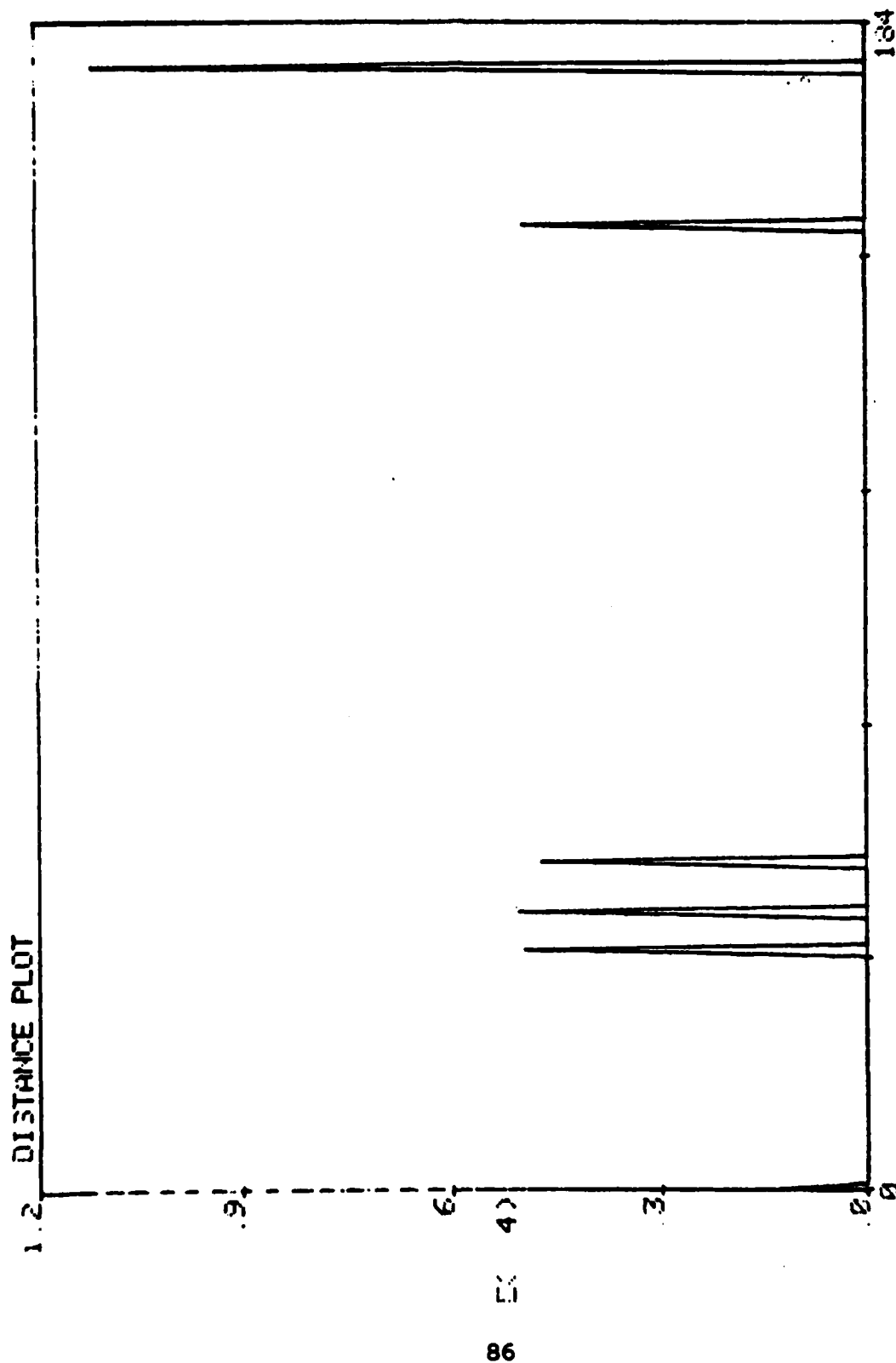


FIGURE 63. Five-Best Distances. Plot of observation energy in ordinate position zero, maximum and five-smallest distances from observation number 171 , M2 rule.

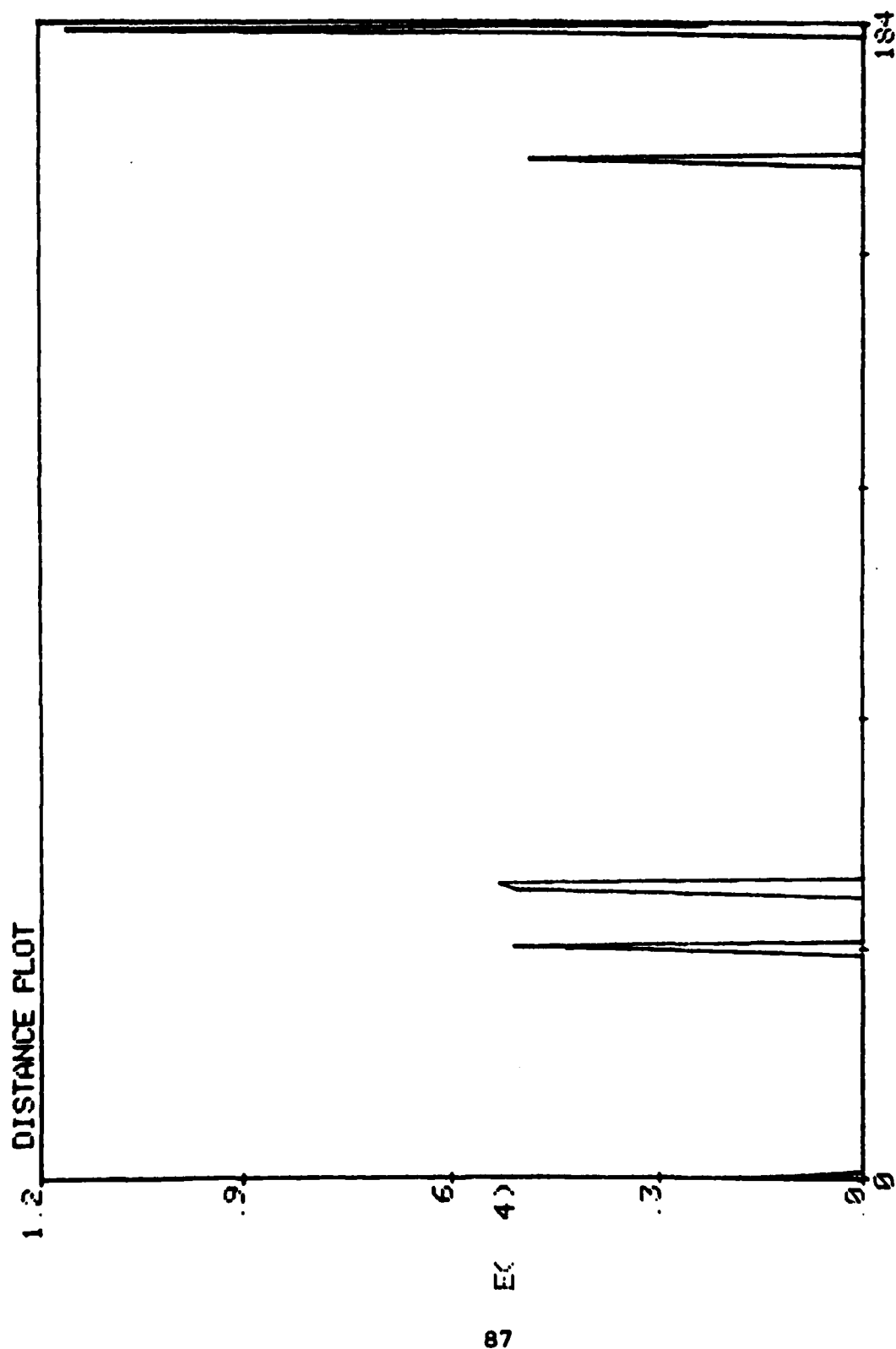
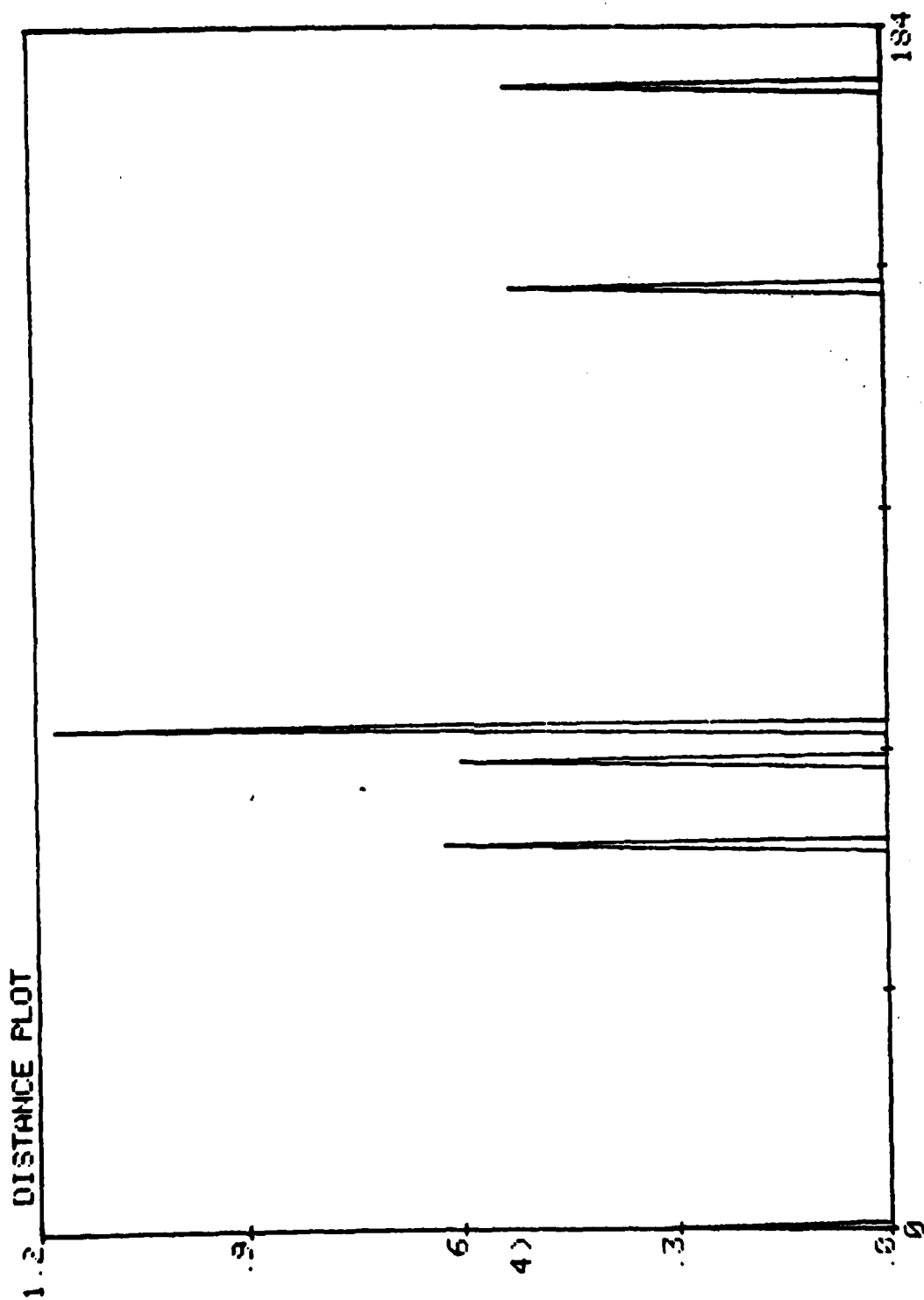


FIGURE 64. Five-Best Distances. Plot of observation energy in ordinate position zero, maximum and five-smallest distances from observation number 181, M2 rule.



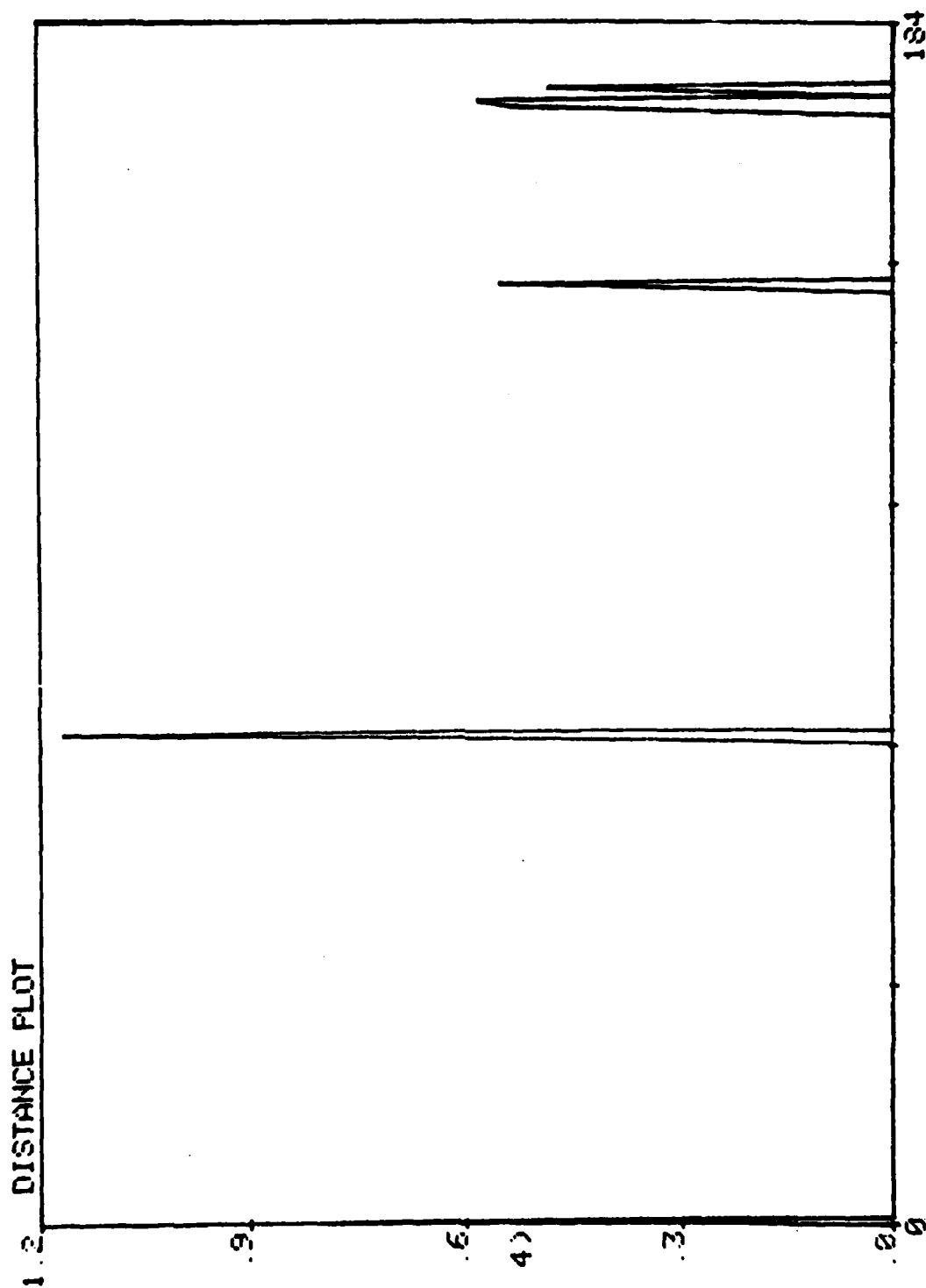


FIGURE 66. Five-Best Distances. Plot of observation energy in ordinate position zero, maximum and five-smallest distances from observation number 195, M2 rule.

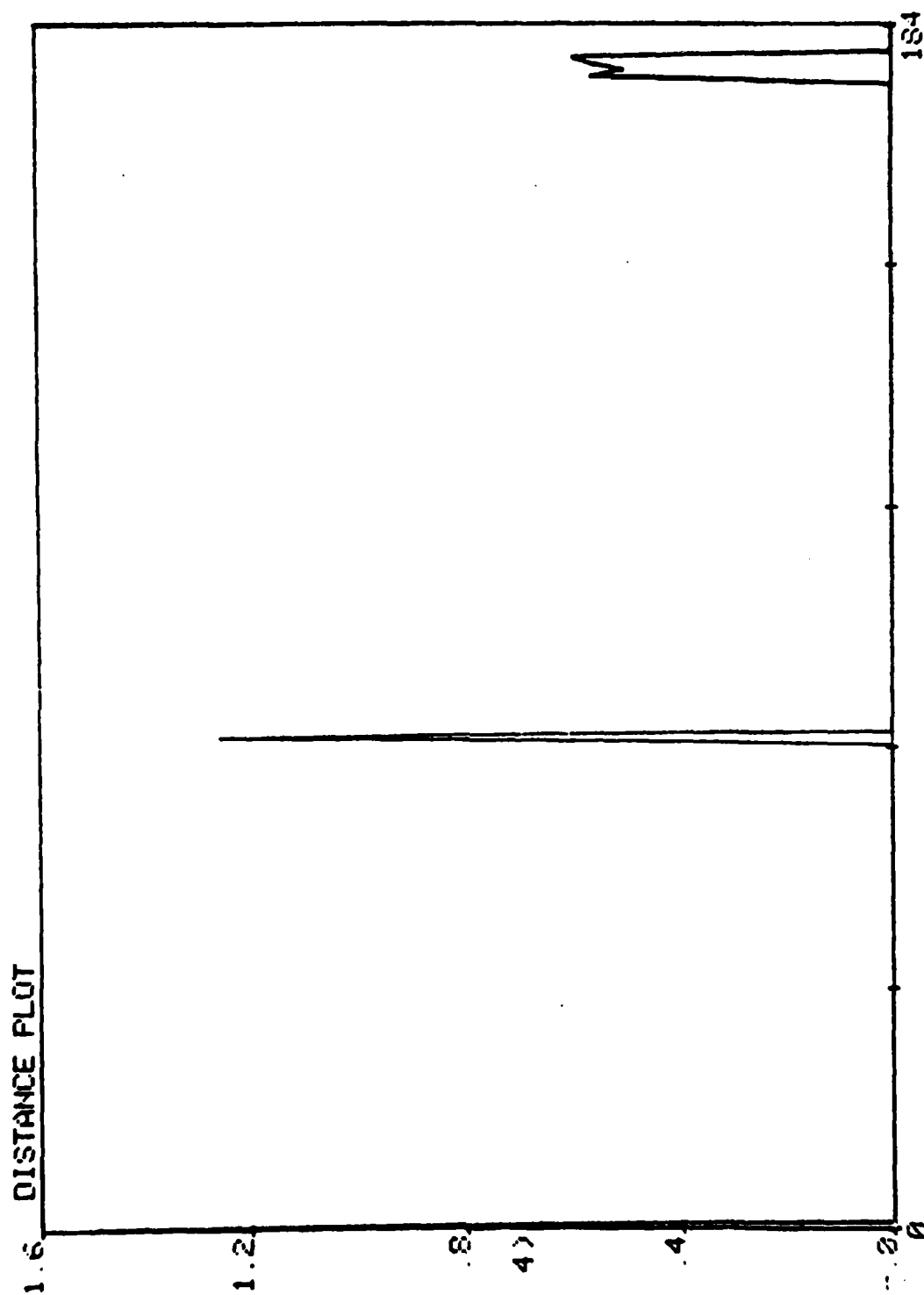


FIGURE 67. Five-Best Distances. Plot of observation energy in ordinate position zero, maximum and five-smallest distances from observation number 200, M2 rule.

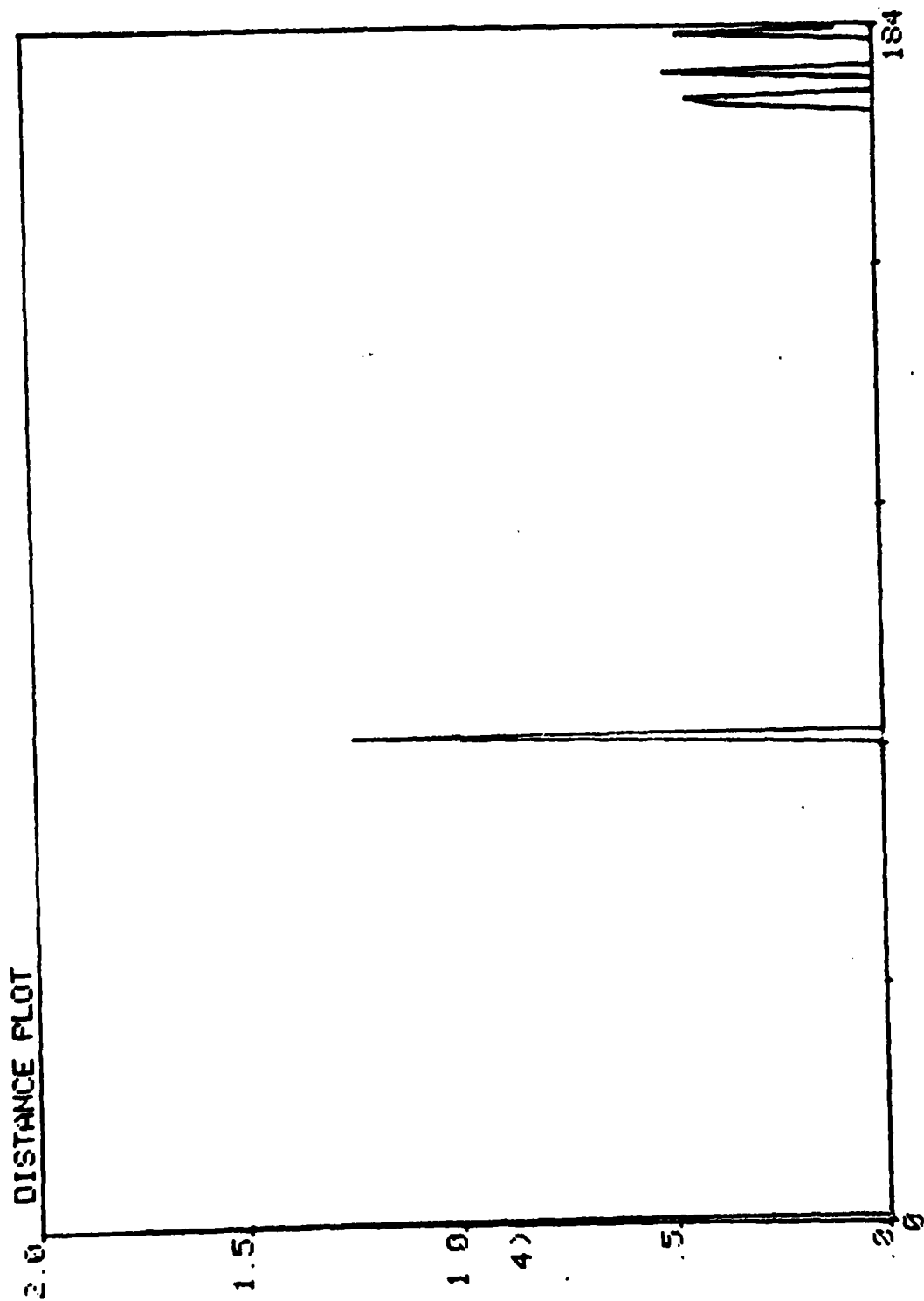


FIGURE 68. Five-Best Distances. Plot of observation energy in ordinate position zero, maximum and five-smallest distances from observation number 202, M2 rule.

100  
this and the previous chapter. We saw that adjacent observations may lie approximately the same distance from neighboring observations in segments of speech which contain high-to-moderate energy and pronounced formant structure. We saw this behavior for voiced vowels and for the "s" sound. We conclude from this observed behavior that observations generated from speech may cluster (Ref 13) in feature space. That is, it should be possible to classify observations without a priori knowledge based on a distance measure computed between them. In the next section, we describe an algorithm designed to do just that.

#### CLUSTER ALGORITHM:

In the previous section, we illustrated observations and their behavior under a distance measure. In this section, we describe an adaptive algorithm designed to cluster feature space into classes of observations. Classification is made on the basis of a decision rule without the benefit of a priori knowledge. The algorithm is adaptive in that a supervisor controls the modification of class descriptions in response to the classification results.

Pal, et. al., described an adaptive model (Ref 6) for computer recognition of vowel sounds using the first three formants as features. Their method uses a training procedure for self-supervised learning and a maximum value of fuzzy membership function as the basis of recognition. In

the remainder of this section, we will present an algorithm based on the above model, but modified to partition the space of observations in noise into clusters.

The model for our cluster algorithm is shown in Figure 69. It is redrawn from Reference 6. It uses a classifier which calculates a sufficient statistic between a set of phonets,  $R$ , and an input observation,  $x$ . The observation is assigned to the phonet class if the sufficient statistic is small enough; otherwise, the observation defines a new class.

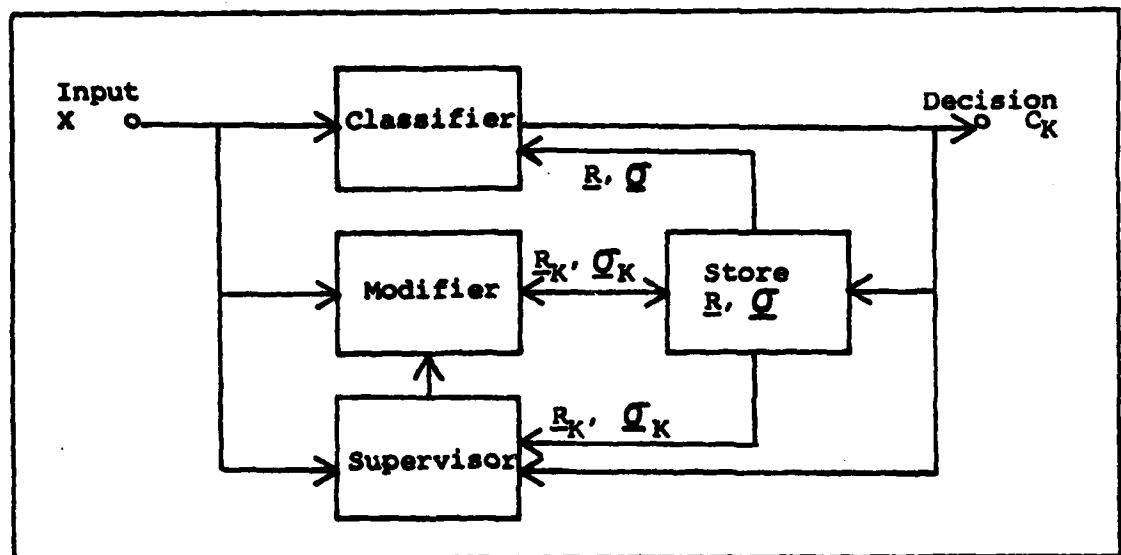


FIGURE 69. Model of an adaptive recognition scheme adapted from reference 6. The classifier assigns the input vector to the class to which it conforms most closely. The supervisor decides if the class descriptors should be modified.

102

Each sample,  $X_i$ , in the sample space,  $\Omega$ , is an observation computed by our Acoustic Analyzer. We assume that these samples have a central tendency about which they cluster. This central tendency is approximated by the ensemble average of all observations which cluster tightly about it.

The sample space is partitioned into a sequence of classes  $\{A_i\}$ . Each class is described by its central tendency, which we call a phonet; by its variance, which is the second central moment of all observations which cluster tightly about the central tendency; by an inner zone parameter which quantifies the adjective "tight" as applied to a cluster; and by an outer zone parameter which determines the outer boundary of the class. These class descriptors are calculated iteratively as the sample space is partitioned.

In this model, the supervisor assumes that observations have a central tendency about which they cluster. It also assumes that if an observation lies close enough to a central tendency, then the probability of misclassification is tolerable.

Accordingly, an annulus is constructed about each central tendency and each observation is compared to the boundaries of this annulus. If the observation is close enough to the central tendency to fall within the inner radius, then it is permitted to modify the description of

that class. If it does not fall within the outer radius of any annulus, then a new class is created to which this observation is assigned.

In operation, a tentative classification of the  $i^{\text{th}}$  observation,  $\underline{x}_i$ , to the  $j^{\text{th}}$  class,  $A_j$ , is based on the minimum value of the sufficient statistic  $D(i, j)$ ; that is, tentatively assign  $\underline{x}_i$  to class  $A_j$  if  $D(i, j) = \min_K D(i, K)$ .

The classification is made permanent on the basis of the two zone parameters,  $\lambda_1$  and  $\lambda_2$ , according to the following rule:

- (1) If  $D(i, j) \leq \lambda_1$ , then make the tentative classification permanent and modify the class descriptors.
- (2) If  $\lambda_1 < D(i, j) \leq \lambda_2$ , then make the tentative classification permanent but do not modify the class descriptors.
- (3) If  $\lambda_2 < D(i, j)$ , then delete the tentative classification, create a new class, and assign  $\underline{x}_i$  as the phonetic descriptor of the class.

These zone parameters  $\lambda_1$  and  $\lambda_2$  are called the inner and outer zone parameters, respectively. They control the classification process and are related to detection performance.

The zone parameters can be set in a number of ways. They could be set arbitrarily and varied until optimum values are found experimentally. Or they could be related

probability of misclassification and noise energy  
speech signal.

The other two class descriptors, the phonet and  
ice vectors, can be calculated iteratively. Suppose  
ence of observations  $\{\underline{x}_i\}$ . Let  $\underline{x}_i \cdot \underline{x}_j$  denote the  
plication of these two vectors component-by-component;  
is, let:

$$\underline{x}_i \cdot \underline{x}_j = (x_{i1} x_{j1}, x_{i2} x_{j2}, \dots, x_{iN} x_{jN}) \quad (7)$$

et  $\underline{x}_i / \underline{x}_j$  denote the division of these two vectors  
nent-by-component. That is, let:

$$\underline{x}_i / \underline{x}_j = (x_{i1}/x_{j1}, x_{i2}/x_{j2}, \dots, x_{iN}/x_{jN}) \quad (8)$$

Given the first  $t$  observations of the sequence  $\{\underline{x}_i\}$ ,  
first noncentral moment is:

$$\underline{R}_t = \frac{1}{t} \sum_{i=1}^t \underline{x}_i, \quad (9)$$

cond noncentral moment is:

$$\underline{C}_t = \sum_{i=1}^t \underline{x}_i \cdot \underline{x}_i, \quad (10)$$

he second central moment is:

$$\underline{S}_t = \frac{1}{t} \underline{C}_t - (\underline{R}_t \cdot \underline{R}_t). \quad (11)$$

ssume another observation is to alter these class  
iptors. Then:

$$\underline{R}_{t+1} = \frac{t}{t+1} \underline{R}_t + \frac{1}{t+1} \underline{x}_{t+1} \quad (12)$$

$$\underline{C}_{t+1} = \underline{C}_t + (\underline{x}_{t+1} \cdot \underline{x}_{t+1}) \quad (13)$$

$$\underline{S}_{t+1} = \frac{1}{t+1} \underline{C}_{t+1} - (\underline{R}_{t+1} \cdot \underline{R}_{t+1}). \quad (14)$$

These equations provide the means for rapid, iterative calculation of the class descriptors using the Array Processor.

The algorithm given by Pal, et. al. (Ref 6), did not specify how to partition the feature space into classes. It assumed class partitions and their descriptors as initial conditions. That is, an initial set of class descriptors was resident in memory at the start of their algorithm. The algorithm given below in step-by-step format does not assume that the sample space is partitioned initially. Rather, it partitions feature space into classes whose boundaries are determined by noise in the system, the sufficient statistic, and class membership variability.

| <u>STEP</u> | <u>ACTION</u>                                                                                                                                                                                                                                                                               |
|-------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 0           | Choose a variance vector, $\underline{V}_0$ , and zone parameters $\lambda_1$ and $\lambda_2$ . One way to do this is to let $V_N$ be the noise variance and $\underline{V}_0 = \lambda_1 \underline{V}_N$ . Then set $\lambda_2 = \lambda_1$ by some multiple. Assume zero-mean noise.     |
| 1           | Read the first observation, $\underline{X}_1$ .<br>Assign $\underline{X}_1$ to class $A_1$ .<br>Describe $A_1$ by:<br>$N_1 = 1$ , $\underline{R}_1 = \underline{X}_1$ , $\underline{C}_1 = \underline{V}_0 + (\underline{X}_1 \cdot \underline{X}_1)$ , $\underline{V}_1 = \underline{V}_0$ |
| 2           | Read the second observation, $\underline{X}_2$ .<br>Compute the supervisory parameter between $\underline{X}_2$ and $\underline{R}_1$ , that is, compute $D(2,1)$ .                                                                                                                         |

STEP

ACTION

If  $D(2,1) \leq \lambda_1$   
Then: assign  $\underline{X}_2$  to  $A_1$   
      modify description of  $A_1$   
else: assign  $\underline{X}_2$  to  $A_2$   
      describe  $A_2$

3

Read the third observation,  $\underline{X}_3$ .  
Choose  $\text{Min}_K D(3, K)$  for  $K = 1, 2$   
Say the minimum is at  $K = 1$ .  
If  $D(3, 1) \leq \lambda_1$   
  Then: assign  $\underline{X}_3$  to class  $A_1$   
        modify description of  $A_1$   
  else: assign  $\underline{X}_3$  to class  $A_3$   
        describe  $A_3$

.

.

.

M

Read  $m^{\text{th}}$  observation  $\underline{X}_m$ .  
Choose  $\text{Min}_K D(M, K)$  for  $K = 1, 2, \dots, n$   
Say the minimum is at  $K = 1$ .  
If  $D(M, 1) \leq \lambda_1$   
  Then: assign  $\underline{X}_m$  to class  $A_1$   
        modify description of  $A_1$   
  else: assign  $\underline{X}_m$  to class  $A_{n+1}$   
        describe class  $A_{n+1}$

.

.

.

107  
Continue in this way until  $M_u$  classes have been described.  $M_u$  is the largest number of classes tolerable and  $M_1$  is the smallest number of classes tolerable.

At this point,  $M_u$  classes have been described. Each class was created because the observation currently under consideration was not described adequately by any of the classes then in existence. Adequacy was defined in terms of class variance. Now, at most,  $M_u$  classes are to be permitted and one cannot be sure at this point that future observations will be described adequately by the current set of  $M_u$  classes. If a future observation is not described adequately, a new class will need to be created. Since processing limitations will intrinsically set some upper limit on  $M_u$ , the number of classes must be reduced to  $M_1$ , some number less than  $M_u$ . This reduction is to be accomplished in such a way as to increase class variance for some of the classes retained so that each class describes adequately more of the feature space. This is done by combining the most similar classes in pairs; that is, by treating the phonets as observations and computing the distance measure on the phonet set.

| <u>STEP</u> | <u>ACTION</u>                                                                 |
|-------------|-------------------------------------------------------------------------------|
| .           |                                                                               |
| .           |                                                                               |
| .           |                                                                               |
| M+1         | Read the first phonet $R_1$ .<br>Compute the distance between $R_1$ and every |

STEPACTION

other phonet, that is,  $D(1, K)$  for  $K = 2, \dots, M_u$ . Choose  $\min_{K \neq 1} D(1, K)$ .

Say the minimum is at  $K = 1$ .

If  $D(1, 1) \leq \lambda_1$

Then: modify description of class  $A_1$  using  $R_1$  as an observation

else: do not modify  $A_1$  descriptors

Delete class  $A_1$ .

M+2

Read next phonet,  $R_2$ .

Choose  $\min_{K \neq 2} D(2, K)$  for  $K = 3, \dots, M_u$

Say minimum at  $K = 1$ .

If  $D(2, 1) \leq \lambda_1$

Then: modify  $A_1$  descriptors using  $R_2$

else: do not modify  $A_1$ .

Delete class  $A_2$ .

•  
•  
•

M+r

Read phonet  $R_r$

Choose  $\min_{K \neq r} D(r, K)$  for  $K = r+1, \dots, M_u$

Say minimum at  $K = 1$

If  $D(r, 1) \leq \lambda_1$

Then: modify description of  $A_1$  using  $R_r$

else: do not modify  $A_1$

Delete  $A_r$ .

•  
•  
•

Continue on in this way until the number of classes is reduced to  $M_1$ . Renumber the classes  $A_1$  through  $A_{M_1}$ .

Once the phonet space is reduced to  $M_1$  classes, then classification proceeds using the rule  $D(i, j)$  as was done above when the phonet vectors were classified. Now, however, the outer zone parameter is applied to the detection problem to determine if a new class should be added to the set  $\{A_i\}$ .

| <u>STEP</u> | <u>ACTION</u>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
|-------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| .           |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
| .           |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
| .           |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
| t           | <p>Read observation <math>\underline{x}_n</math>.</p> <p>Choose <math>\min_K D(n, K)</math> <math>K = 1, \dots, M &lt; M_u</math></p> <p>Say minimum at <math>K = 1</math>.</p> <p>If <math>D(n, 1) &lt; \lambda_1</math></p> <p>Then: assign <math>\underline{x}_n</math> to class <math>A_1</math></p> <p>modify class descriptors for <math>A_1</math></p> <p>else: If <math>D(n, 1) \leq \lambda_2</math></p> <p>then: assign <math>\underline{x}_n</math> to class <math>A_1</math></p> <p>do not modify descriptors</p> <p>else: assign <math>\underline{x}_n</math> to <math>A_{M+1}</math></p> <p>describe class <math>A_{M+1}</math></p> |
| .           |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
| .           |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
| .           |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |

Continue in this way until  $M+1 = M_u$ . When  $M+1 = M_u$ , then reduce the phonet space until there are  $M_1$  classes. Then classify more observations...

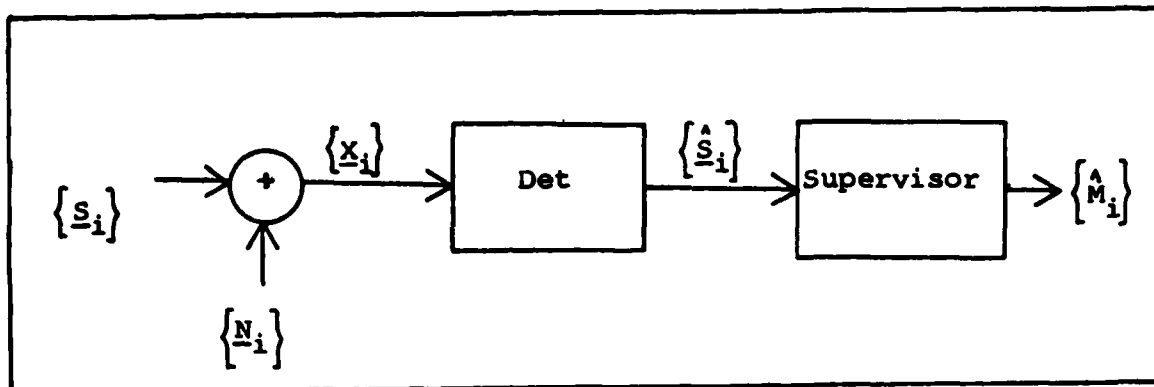


FIGURE 70. Model of the Supervisory Detection Process.

A model of the supervisory detection process is shown in Figure 70. The additive noise vector is assumed to have uncorrelated components, each distributed with zero-mean and variance  $V_i$ . Thus  $\underline{X}_i = \underline{S}_i + \underline{N}_i$ , its mean is  $E[\underline{X}_i] = \underline{\bar{X}}_i = \underline{S}_i$ , and its variance is:

$$V = \begin{pmatrix} V_1 & & 0 \\ & V_2 & \\ & & \ddots \\ 0 & & & \ddots \\ & & & & V_m \end{pmatrix} \quad (15)$$

Wozencraft and Jacobs have shown that the optimum decision rule is to choose the class,  $A_i$ , to which the

observation,  $\underline{X}_i$ , most probably belongs (Ref 14:212-214).

That is, choose

$$A_j: \text{Max}_i P\{\underline{S}_i\} P_N(\underline{X}|\underline{S}_i) \quad (16)$$

where the set of a priori probabilities are  $\{P\{\underline{S}_i\}\}$  and  $P_N(\underline{X}|\underline{S}_i)$  is assumed to be the joint - M gaussian density of  $\underline{X}$  conditioned on  $\underline{S}_i$ . Melsa and Cohn give the density with mean  $\underline{S}_i$  and variance  $V$  (Ref 15:69), and develop the multiple decision problem (Ref 15:113-116). We parallel their development.

The joint - M gaussian density is:

$$P_N(\underline{X}|\underline{S}_i) = \frac{\exp\left\{-\frac{1}{2}(\underline{X} - \underline{S}_i)^T V^{-1}(\underline{X} - \underline{S}_i)\right\}}{\sqrt{\prod_{j=1}^M 2\pi v_j}} \quad (17)$$

The optimum decision rule, equation 16, is met when the logarithm of its terms is maximum, so we express our decision rule as:

$$A_j: \text{Max}_i U_i - \frac{1}{2}(\underline{X} - \underline{S}_i)^T V^{-1}(\underline{X} - \underline{S}_i) \quad (18)$$

where:

$$U_i = \ln P\{\underline{S}_i\} - \frac{1}{2} \sum_{j=1}^M \ln(2\pi v_j) \quad (19)$$

Assuming equally likely a priori probabilities, then

$U_i = U_j \forall i, j$  and our decision rule reduces to:

$$A_j: \text{Min}_i (\underline{X}_i - \underline{S}_i)^T V^{-1}(\underline{X} - \underline{S}_i) \quad (20)$$

We define the sufficient statistic:

$$D(i, j) = (\underline{X}_i - \underline{S}_j)^T V^{-1} (\underline{X}_i - \underline{S}_j) \quad (21)$$

and write the decision rule as:

$$A_j: \min_K D(i, K). \quad (22)$$

Our algorithm approximates  $\underline{S}_i$  as the ensemble average of all observations which it is sufficiently certain belongs to the class  $A_i$ , and the variance  $V_i$  as the ensemble variance of those observations.

The probability of error is the probability that there exists at least one class,  $A_K$ , given  $\underline{S}_j$ , for which  $D(i, K) < D(i, j)$ . That is, we let  $E_K$  denote the event:

$$E_K = \{W: D(i, K) < D(i, j) | \underline{S}_j\} \quad (23)$$

where the statistics are, themselves, random variables and their dependence on the sample point,  $W$ , is suppressed for convenience.

We apply the union bound as given by Melsa and Cohn (Ref 15:115) to upper bound the probability of error. Say there are  $L$  classes. Then we say:

$$P\{E | \underline{S}_j\} < \sum_{\substack{K=1 \\ K \neq j}}^L P\{E_K | \underline{S}_j\}. \quad (24)$$

The task is to find  $P\{E_K | \underline{S}_j\} \forall K \neq j, K \leq L$ .

The event  $E_K$  occurs whenever, given  $\underline{S}_j$ ,

$$D(i, K) < D(i, j) \quad (25)$$

$$(\underline{X}_i - \underline{S}_K)^T V^{-1} (\underline{X}_i - \underline{S}_K) < (\underline{X}_i - \underline{S}_j)^T V^{-1} (\underline{X}_i - \underline{S}_j) \quad (26)$$

$$\begin{aligned}
& \underline{x}_i^T \underline{v}^{-1} \underline{x}_i - \underline{s}_K^T \underline{v}^{-1} \underline{x}_i - \underline{x}_i^T \underline{v}^{-1} \underline{s}_K + \underline{s}_K^T \underline{v}^{-1} \underline{s}_K \\
& < \underline{x}_i^T \underline{v}^{-1} \underline{x}_i - \underline{s}_j^T \underline{v}^{-1} \underline{x}_i - \underline{x}_i^T \underline{v}^{-1} \underline{s}_j + \underline{s}_j^T \underline{v}^{-1} \underline{s}_j \\
& \underline{s}_K^T \underline{v}^{-1} \underline{s}_K - \underline{s}_j^T \underline{v}^{-1} \underline{s}_j < 2 \underline{x}_i^T \underline{v}^{-1} (\underline{s}_K - \underline{s}_j) \\
& 2 \underline{x}_i^T \underline{v}^{-1} (\underline{s}_K - \underline{s}_j) > \underline{s}_K^T \underline{v}^{-1} \underline{s}_K - \underline{s}_j^T \underline{v}^{-1} \underline{s}_j.
\end{aligned} \tag{27}$$

Using  $\underline{x}_j = \underline{s}_j + \underline{n}_j$ , we have:

$$\begin{aligned}
& 2(\underline{s}_j + \underline{n}_j)^T \underline{v}^{-1} (\underline{s}_K - \underline{s}_j) > \underline{s}_K^T \underline{v}^{-1} \underline{s}_K - \underline{s}_j^T \underline{v}^{-1} \underline{s}_j \\
& 2 \underline{n}_j^T \underline{v}^{-1} (\underline{s}_K - \underline{s}_j) > \underline{s}_K^T \underline{v}^{-1} \underline{s}_K - 2 \underline{s}_j^T \underline{v}^{-1} (\underline{s}_K - \underline{s}_j) - \underline{s}_j^T \underline{v}^{-1} \underline{s}_j \\
& 2 \underline{n}_j^T \underline{v}^{-1} (\underline{s}_K - \underline{s}_j) > \underline{s}_K^T \underline{v}^{-1} \underline{s}_K - 2 \underline{s}_j^T \underline{v}^{-1} \underline{s}_K + \underline{s}_j^T \underline{v}^{-1} \underline{s}_j
\end{aligned} \tag{28}$$

Denote the left side of equation 28 by the random variable  $Y_K$  and the right side by the nonrandom variable  $Z_K$ . The mean of  $Y_K$  is zero and its variance is:

$$\text{Var}\{Y_K\} = E[Y_K^2] \tag{29}$$

$$= 2 E \left[ \left( \sum_{i=1}^M \frac{n_{ij} (s_{ik} - s_{ij})}{v_i} \right)^2 \right] \tag{30}$$

$$= 2 E \left[ \sum_{i=1}^M \frac{n_{ij} (s_{ik} - s_{ij})}{v_i} \left( \sum_{l=1}^M \frac{n_{lj} (s_{lk} - s_{lj})}{v_l} \right) \right] \tag{31}$$

Because the noise components are uncorrelated, equation 31 reduces to:

$$\text{Var}\{Y_K\} = 2 \sum_{i=1}^M (s_{ik} - s_{ij})^2 \tag{32}$$

So we write equation 28 in terms of the zero mean, unit variance random variable:

$$a_K > \frac{Z_K}{2 |\underline{s}_K - \underline{s}_j|^2} = Z_{K1} \tag{33}$$

where  $|\underline{s}_K - \underline{s}_j|$  is the Euclidean norm between the phonets of the pair of classes  $A_K$  and  $A_j$ , the phonet being approximated by the mean descriptor. The random variable  $a_K$  is distributed gaussian with zero-mean and unit variance. So its density is:

$$P_a(a_K) = \frac{1}{\sqrt{2\pi}} \exp \left\{ -\frac{1}{2} a_K^2 \right\} \quad (34)$$

Then:

$$P\{E_K | \underline{s}_j\} = \frac{1}{\sqrt{2\pi}} \int_{Z_{K1}}^{\infty} \exp \left\{ -\frac{1}{2} a_K^2 \right\} da_K \quad (35)$$

$$= Q(Z_{K1}) \quad (36)$$

The error probability conditioned on  $\underline{s}_j$  is, from equation 24,

$$P\{E | \underline{s}_j\} < \sum_{\substack{K=1 \\ K \neq j}}^L Q(Z_{K1}) \quad (37)$$

To bound the overall average error probability, equation 37 is averaged over all classes, so:

$$P\{E\} < \sum_{l=1}^L \frac{1}{L} \sum_{\substack{K=1 \\ K \neq j}}^L Q(Z_{K1}) \quad (38)$$

This average can be further upper-bounded in terms of the minimum  $Z_{K1}$ . That is, denote:

$$Z_{\min} = \min_1 \left[ \min_{K \neq 1} Z_{K1} \right]. \quad (39)$$

Then:

$$P\{E\} < (L - 1) Q(Z_{\min}). \quad (40)$$

The Q-function can be approximated (Ref 15:258) by:

$$Q(\chi) \approx \frac{1}{\sqrt{2\pi}} \frac{2 \exp \left\{ -\frac{1}{2} \chi^2 \right\}}{\chi + \sqrt{\chi^2 + \gamma/\pi}} \quad (41)$$

so:

$$P \{E\} \approx \frac{2 (L - 1) \exp \left\{ -\frac{1}{2} z_{\min}^2 \right\}}{\sqrt{2\pi} (z_{\min} + \sqrt{z_{\min}^2 + \gamma/\pi})} \quad (42)$$

One might relate the probability of classification error to the maximum number of classes,  $M_u$ , and the outer supervisory parameter in the following way. One could consider the minimum value of  $Z_k$ , the right side of equation 28, to be the signal-to-noise ratio in detecting the closest two classes. The signal-to-noise ratio at the input to the detector could be measured. It would be:

$$SNR = \frac{\min_{i \neq j} |\underline{s}_i - \underline{s}_j|^2}{\text{Var} \{ \underline{N} \}} \quad (43)$$

For a specified  $P \{E\}$  in the detector, then,

$$z_{\min} = Q^{-1} \left( \frac{P \{E\}}{(M_u - 1)} \right) \quad (44)$$

from equation 40 can be thought of as the minimum distance normalized to the input variance at the detector. Scaling this distance by the SNR at the detector input, one gets the outer supervisor parameter from:

$$\lambda_u^2 z_{\min} = SNR. \quad (45)$$

Once the feature space is partitioned into classes of phonetic units, then those phonetic units can be recognized

in speech. Our Acoustic Analyzer, Seelandt's Speech Sound Analysis Machine (Ref 1) or some other distance computer, can be used to accomplish the recognition. The constraint, of course, is that the phonetic units used to partition feature space must be the same as those generated from speech by the distance computer.

In this chapter, we have presented the results of this project. We presented the Acoustic Analyzer and illustrated the behavior of observations under a minimum distance decision criterion. Then Conclusions and Recommendations are presented in Chapters IV and V, respectively.

#### IV. CONCLUSIONS.

We have implemented Seelandt's Speech Sound Analysis Machine (SSAM) on the Eclipse S250 Array Processor. We were able to generate phonetic units, which we called observations and phonets, from speech files. We calculated distances between observations and phonets, and we were able to output the best phonet choices along with other parameters for input to Montgomery's word recognizer (Ref 10). We also provided a graphics capability for plotting phonetic units, distances, and time files.

We also reviewed the parameters Seelandt identified for further study (Ref 1) and provided a basis for choosing parameter values. With the values we chose, we observed that adjacent phonetic units in voiced vowel sounds resemble each other in both distance rules. We found that while adjacent phonetic units in speaker-absent background noise and in "r" sounds did not resemble each other, they did in "s" sounds. We conclude that there seems to be a sufficient tendency for phonetic units to cluster to warrant further work along these lines.

We also adapted an algorithm for self-supervised vowel recognition, which used the first three formants as features in the absence of noise to our problem of phonetic unit recognition. The algorithm is explained and we showed how the S250 Array Processor and our Acoustic Analyzer could be used to implement it.

## RECOMMENDATIONS.

We were unable to accomplish much important work that is to be done before a phonet detector built along the lines of our Acoustic Analyzer can be optimized. One major effort is needed in implementing an algorithm for partitioning the feature space. The algorithm by Pal, et. al. (Ref 10) that we considered in this project is one of many candidates. While we feel it has merit; we found many others also warrant careful consideration. The recent book by Rabinovitch and Kittler (Ref 9) discusses the topic in detail. The text by Fukunaga (Ref 13) also treats the subject and presents algorithms which could be applied to this problem. An algorithm should be implemented.

Once a method for partitioning feature space is implemented, the parameters Seelandt discussed and that were tested in this project, should be investigated experimentally. That is, their effect on recognition of phonetic features chosen by a feature space partitioning algorithm should be determined.

Several features need to be added to our Acoustic Analyzer. For one, the capability to overlap FFT windows of sizes other than just 128 points needs to be incorporated. Routine S64, which allows the necessary window sizes but does not permit window overlapping, can serve as a departure

point for a general purpose FFT routine that incorporates the overlap feature, additional window shapes, and more sophisticated spectrum shaping options.

Also needed is the capability for the distance routines, DSTA and DSTN, to do a cross-correlation over shifts of only a few components between observation and phonet spectrum. In this way, the number of phonet clusters in feature space may be reduced if several phonet clusters are only shifted versions of others. Detection could proceed conditionally on frequency shift, and then choices between the shifts could be made. But before this is implemented, it may be valuable to implement first a partitioning algorithm. Once the feature space is partitioned, the cluster characteristics may suggest whether spectral shifts may offer improvement, how far to shift, and how much improvement. Answers to these questions depend heavily on the way that feature space is partitioned.

## BIBLIOGRAPHY

1. Seelandt, Karl G. Computer Analysis and Recognition of Phoneme Sounds in Connected Speech, MS Thesis GE/EE/81D-53, School of Engineering, Air Force Institute of Technology, Wright-Patterson AFB, Ohio, December 1981.
2. Felkey, Mark A. Automatic Recognition of Phonemes in Continuous Speech, MS Thesis GE/EE/80D-20, School of Engineering, Air Force Institute of Technology, Wright-Patterson AFB, Ohio, December 1980.
3. De Souza, Peter and Thomson, Peter J. "LPC Distance Measures and Statistical Tests with Particular Reference to the Likelihood Ratio," IEEE Transactions on Acoustics, Speech, and Signal Processing, Vol ASSP-30, No. 2, April 1982.
4. Kassam, Salem A. A Bibliography on Nonparametric Detection, AD A080187, prepared for the Air Force Office of Scientific Research, AFOSR-TR-80-0025, 1979.
5. Laniotis, D. G. Adaptive Systems, AD 776015, prepared for the Air Force Office of Scientific Research, AFOSR-TR-74-0309, February 1974.
6. Pal, S. K., Datta, A. K., and Dutta Majumder, D. "A Self-Supervised Vowel Recognition System," Pattern Recognition, Pergamon Press Ltd, Great Britain, Vol 12, 27-34.
7. Nonparametric/Robust Methods and Non-Gaussian Models in Communication Theory, Johns Hopkins University, AD-763407, prepared for the Office of Naval Research, 1971.
8. Rabiner, L. R. and Schafer, R. W. Digital Processing of Speech Signals, Prentice Hall, 1978.
9. Devijver, R. P. and Kittler, J. Pattern Recognition: A Statistical Approach, Prentice Hall, 1982.
10. Montgomery, Gerald. Isolated Word Recognition Using Fuzzy Set Theory, MS Thesis GE/EE/82D-74, School of Engineering, Air Force Institute of Technology, Wright-Patterson AFB, Ohio, December 1982.
11. Sivian, L. J. "Speech and its Power Measurement," the Bell System Technical Journal, Vol 8, pp. 646-661, 1929.

12. Kuligowski, T. J. An Automatic Speech Teaching Responder, MS Thesis, United States Naval Postgraduate School, AD 851813, June, 1968.
13. Fukunaga, K. Introduction to Statistical Pattern Recognition, Academic Press Inc., 1972.
14. Wozencraft, J. M., and Jacobs, I. M. Principles of Communication Engineering, John Wiley and Sons, New York, 1965.
15. Melsa, J. L., and Cohn, D. L. Decision and Estimation Theory, McGraw-Hill, Inc., 1978.

APPENDIX A  
SPECTROGRAM OF FILE CT56.SP

ED \*\*\*  
ENTENCE SPOKEN: FIVE SIX  
TIME: 11 49 21  
LAST TIME SLICE = 324

0Z OVERLAP  
CT ABOVE 500HZ.  
T BELOW 300HZ.

FREQUENCY (HZ)

2. K

3. K

4. K

3K

93K

183K

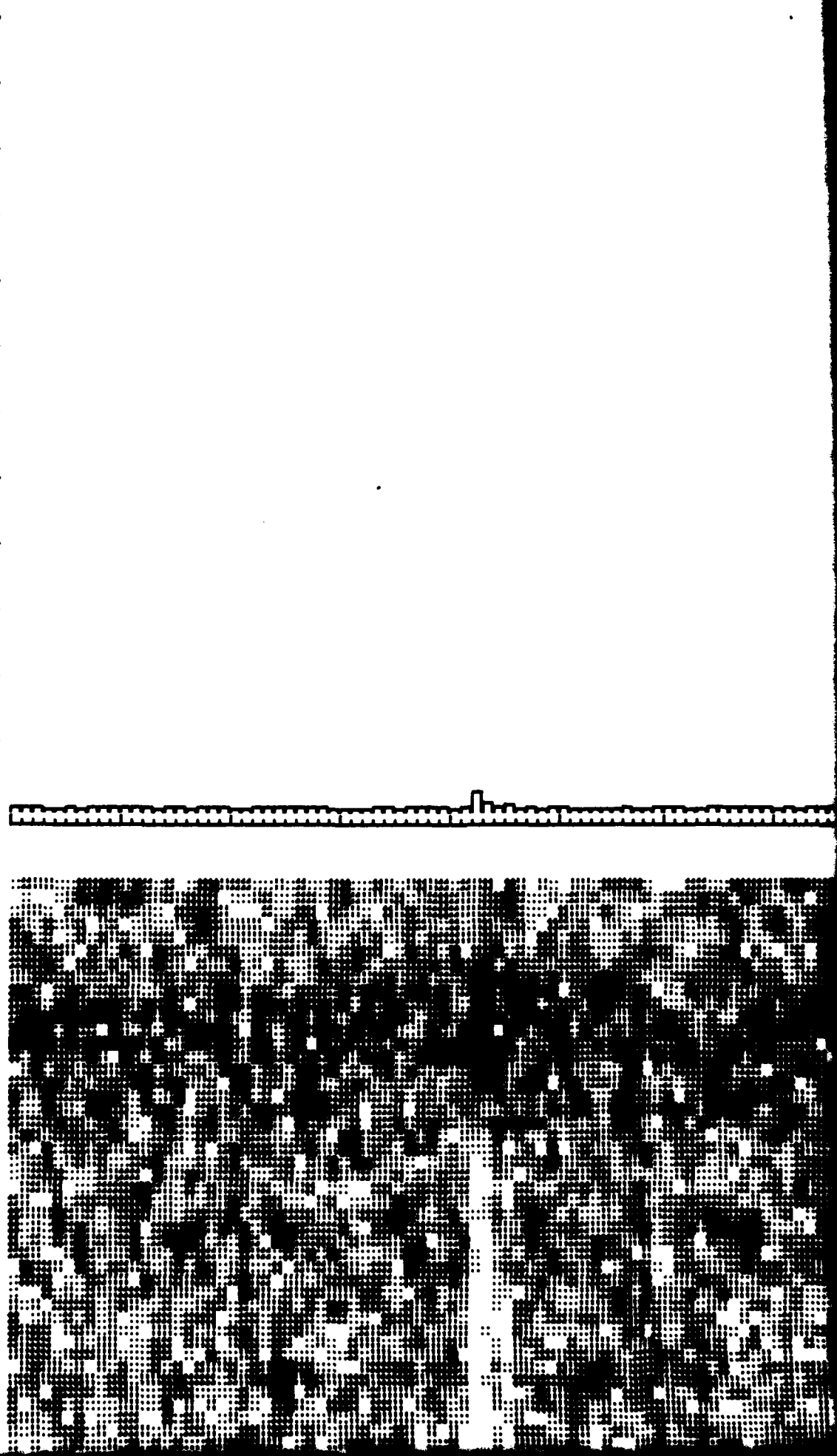
ENERGY

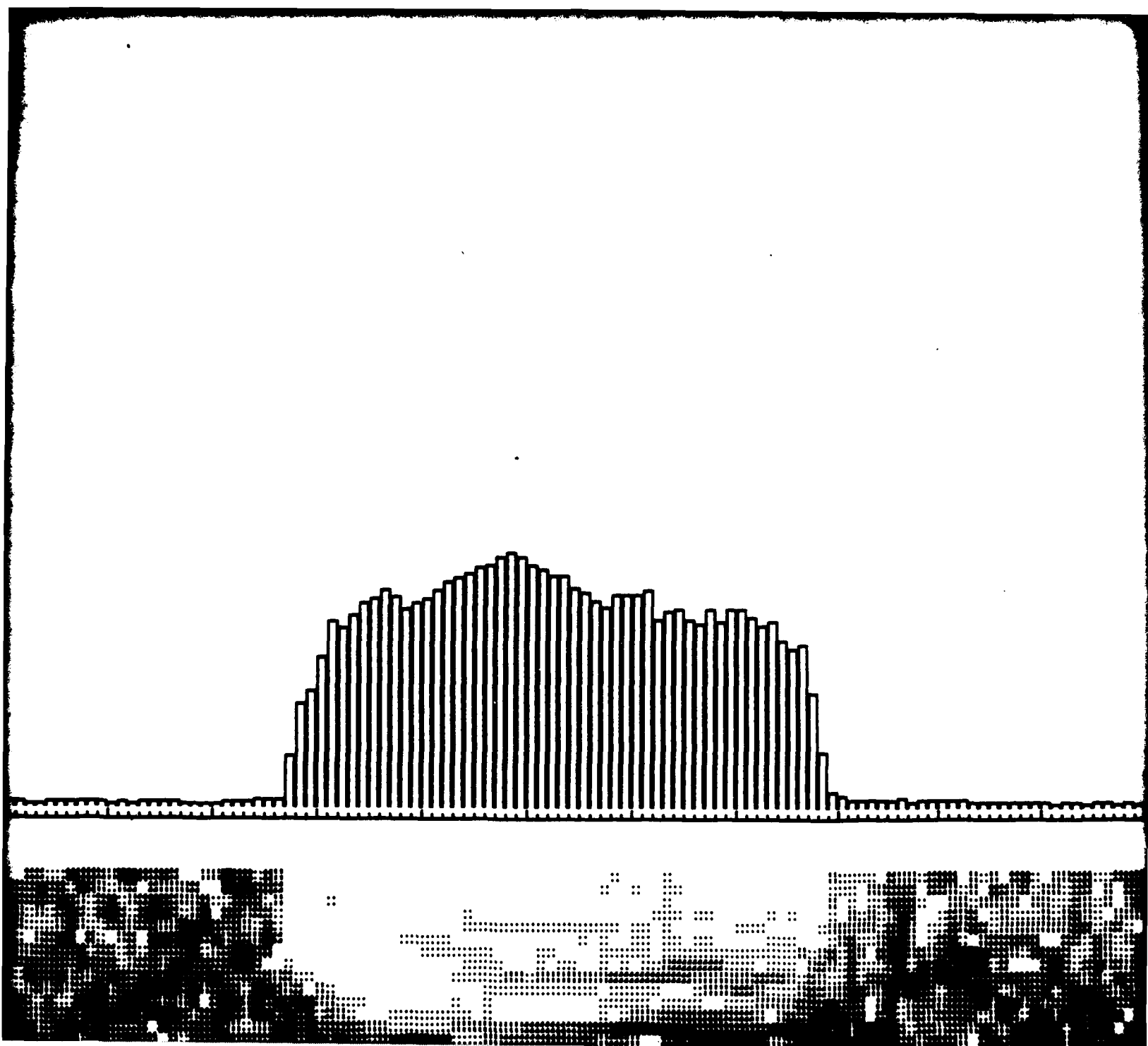
273K

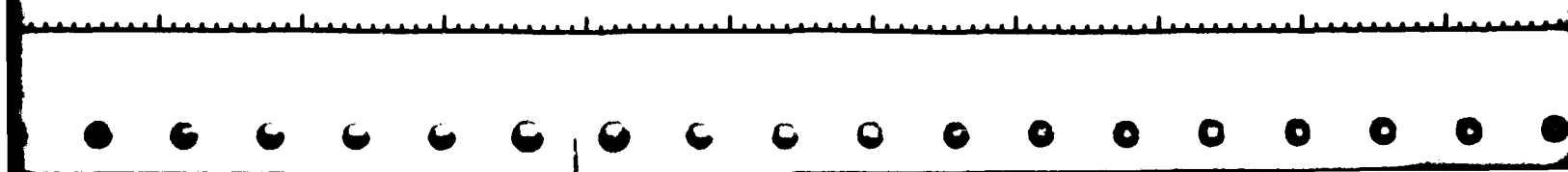
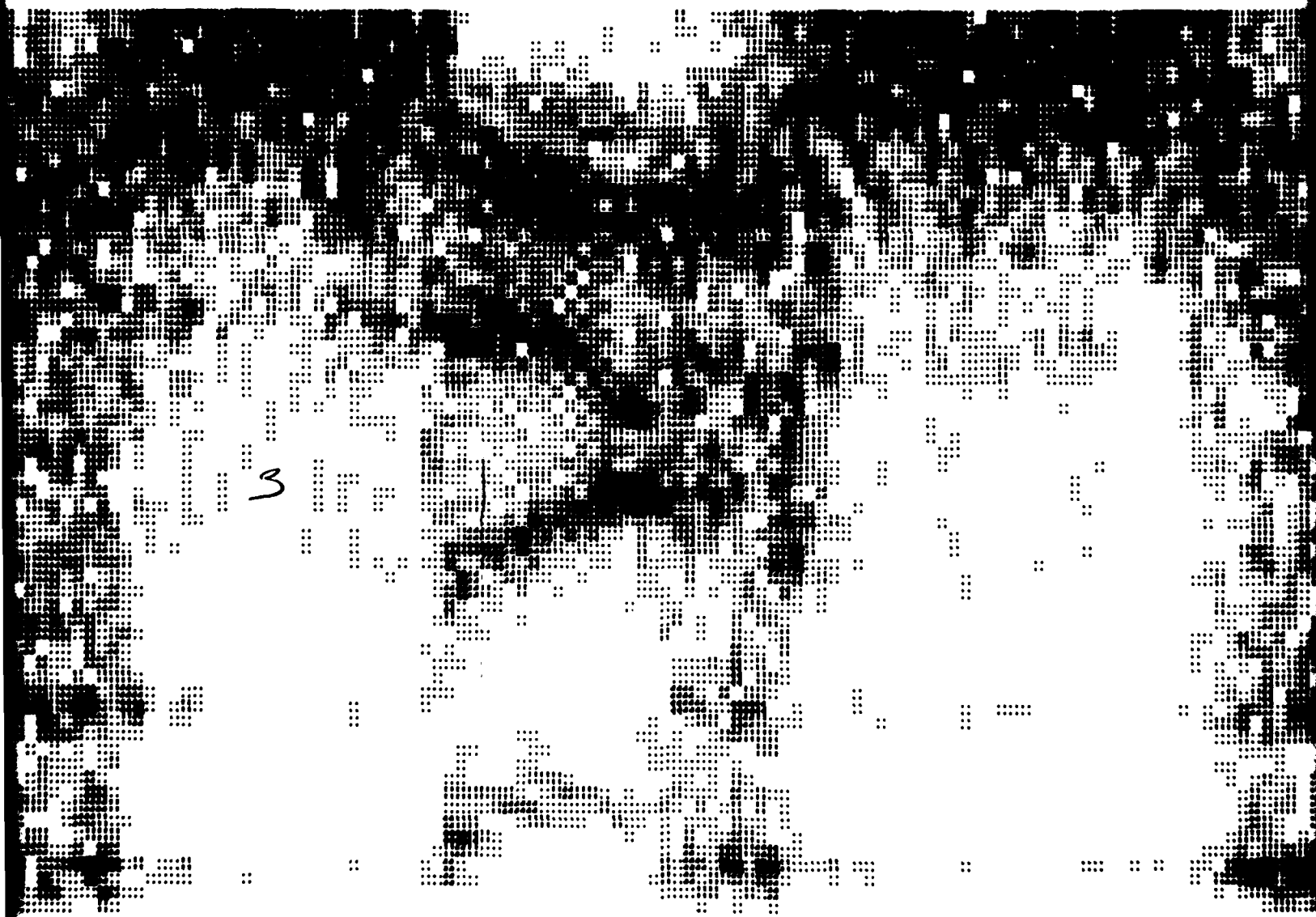
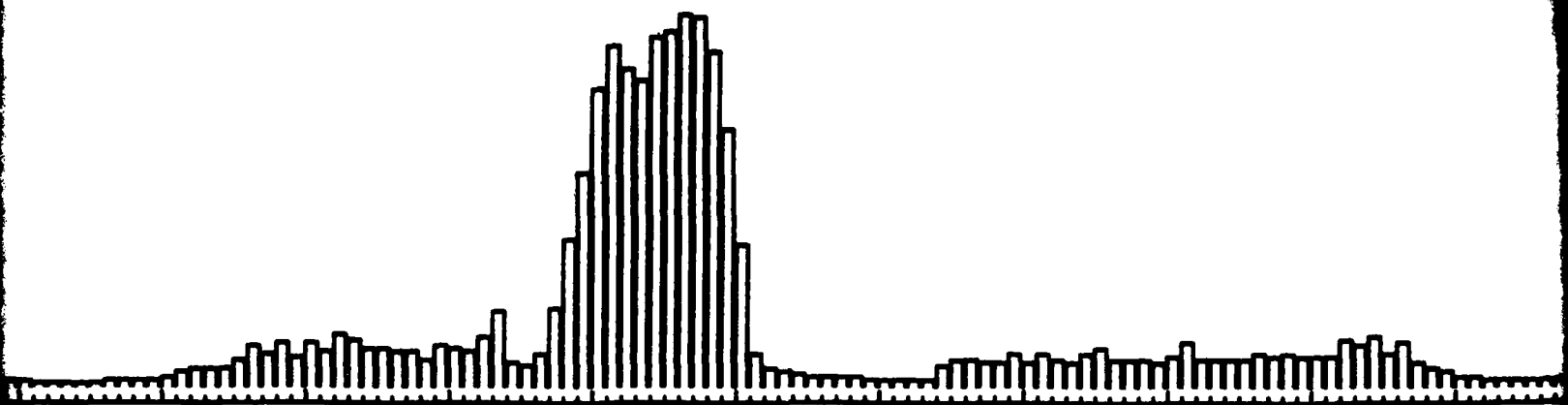
363K

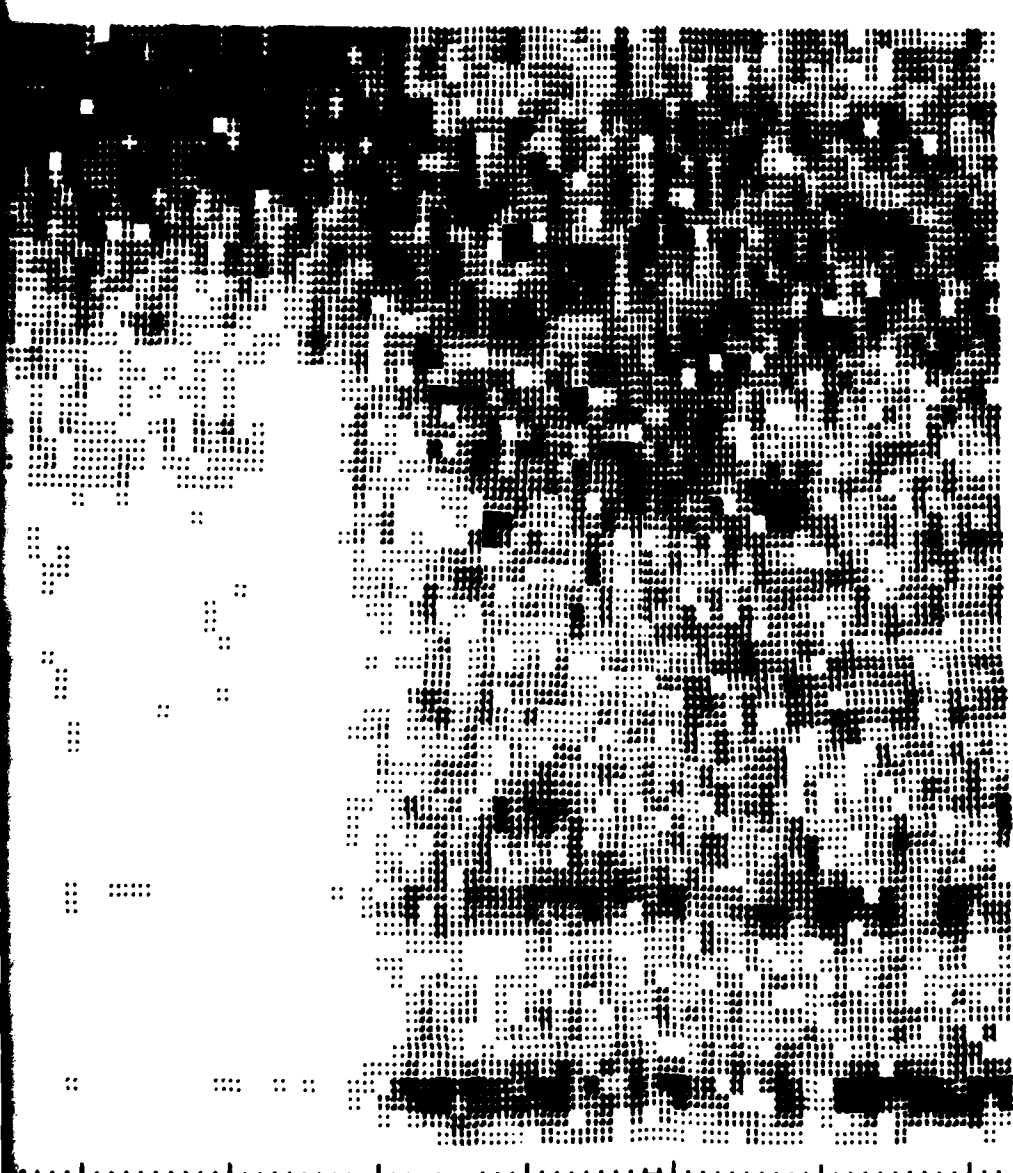
453K

543K

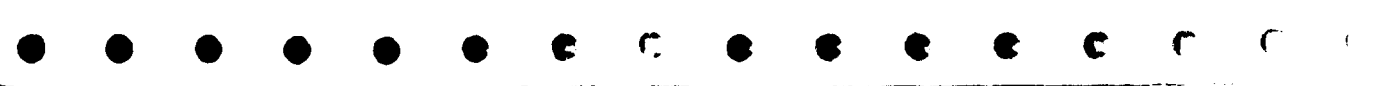








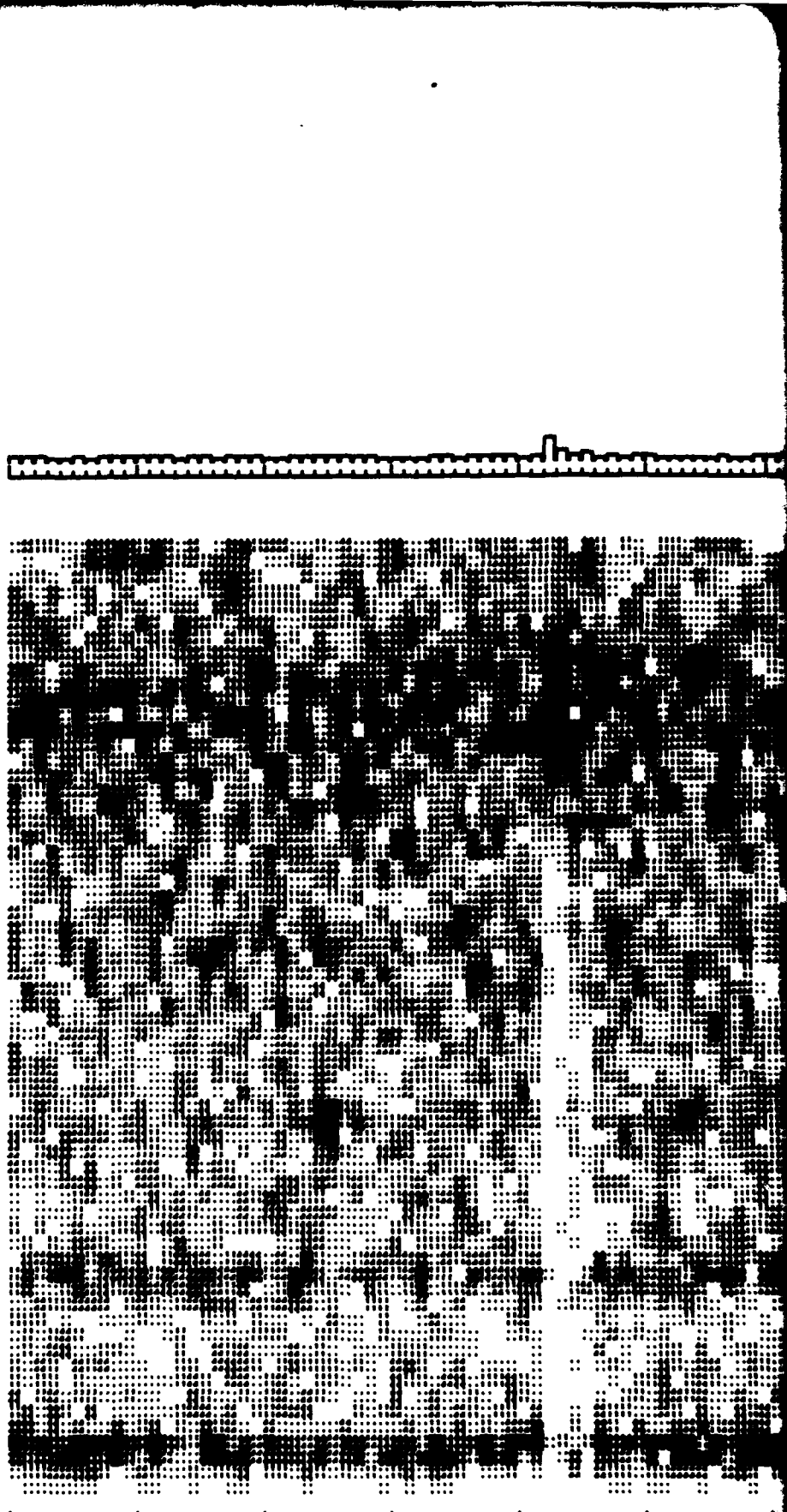
4

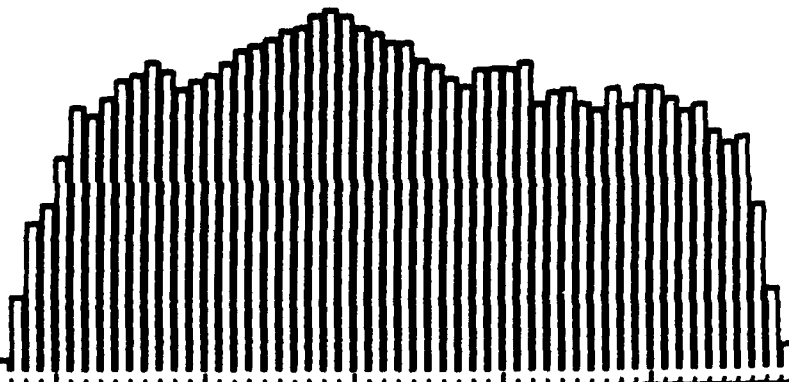


5  
\*\*\* HANNING WINDOW USED \*\*\*  
FILE: CT36.08 SENTENCE SPOKEN: FIVE SIX  
DATE: 11 29 1982 TIME: 11 49 21  
FIRST TIME SLICE = 1 LAST TIME SLICE = 324

DFT SIZE = 128 50% OVERLAP  
PREEMPHASIS = 10DB/OCT ABOVE 500HZ.  
DEEMPHASIS = 10DB/OCT BELOW 300HZ.

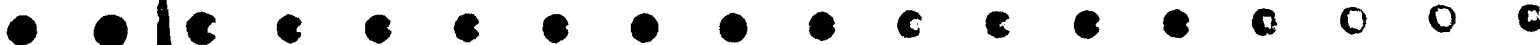
DC 1. K 2. K 3. K 4. K 93K 183K ENERGY 27





2

6



APPENDIX B  
SOFTWARE DOCUMENTATION

### STRUCTURE OF THE ACOUSTIC ANALYZER:

The Acoustic Analyzer consists of one executive routine, 15 subroutines, and calls to two library files. The executive routine, DRVR.SV, and subroutines CHKO, FCTR, GFDB, along with subroutines from the library files, are loaded into one save file, DRVR.SV. An overlay node is created in the save file into which segments of the overlay file, DRVR.OL, are loaded when called. The macro file which calls the loader utility is included in this documentation.

Figure 71 illustrates the software structure. As called, routines DRSQ, DSTN, DSTA, or PLTO are loaded into the executive routine at the overlay node, along with their supporting subroutines. The other routines: CHKO, FCTR, GFDB, and the libraries are accessible to the contents of the overlay node.

Below is a list of each routine in the Acoustic Analyzer, along with a brief statement of its purpose:

- (1) DRVR is the executive. It allows the operator to select either the spectral option, either of the two distance options, or the plot options.
- (2) DRSQ calls either S128 or S64 to compute the Fast Fourier Transform (FFT) of an input file.
- (3) S128 computes FFTs of an input file using a 128 point window which overlaps adjacent windows by 64 points.

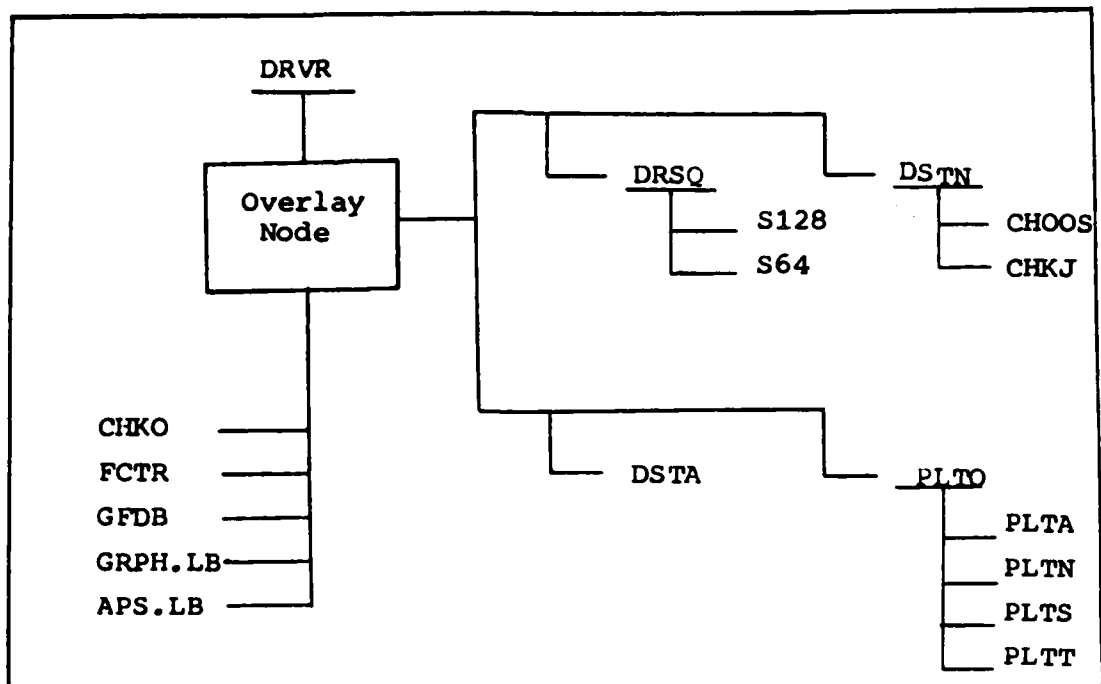


FIGURE 71. Structure of the Acoustic Analyzer. Options available from DRVR are program segments resident on disk and loaded into the overlay node on call.

(4) S64 computes FFTs of an input file using a nonoverlapped window of any length which is an integral multiple of two in the range 16-1024 points, inclusive.

(5) DSTA computes distances between observations and phonets, and stores the symmetric distance matrix on disk.

(6) DSTN computes the distance matrix that DSTA computes, but chooses a selected number of best template matches for output to disk.

(7) CHOOS is the subroutine that DSTN calls to find best matches between observations and phonets.

(8) CHKJ is called by DSTN to output data when distance-file elements are accumulated.

(9) PLTO allows the operator to call options h either display spectrum, display distances output STA, display distances output by DSTN, or display the ents of integer disk files.

(10) PLTA displays distance files computed by

.

(11) PLTS displays spectral files computed by

.

(12) PLTN displays distance files computed by

.

(13) PLTT displays any segment of 512 integers  
ewer from a disk file.

(14) CHKO is called by DSTA and DSTN to type a  
ion statement to screen if a file unit number is in use.

(15) FCTR is called by GFDB to factor an integer  
a linear combination of four terms.

(16) GFDB is called by PLTA, PLTS, PLTN, and PLTT  
compute the location of a specified data point in a file.

FILE: LDDVR.MC  
LANGUAGE: Command Line Interpreter  
DATE: September 21, 1982  
AUTHOR: D. Martin  
SUBJECT: Acoustic Analysis  
CALLING SEQUENCE: LDDVR  
DATE OF LAST REVISION: September 21, 1982

PURPOSE:

This macro file loads program DRVR.SV. An overlay node is established and an overlay file, DRVR.OL, and load map, DRVR.LM, are created.

DESCRIPTION:

Location: DP4:BRATCHET

|       |          |             |
|-------|----------|-------------|
| Size: | LDDVR.MC | 155 bytes   |
|       | DRVR.SV  | 51200 bytes |
|       | DRVR.FR  | 2613 bytes  |
|       | DRVR.RB  | 3054 bytes  |
|       | DRVR.OL  | 40960 bytes |
|       | DRVR.LM  | 9491 bytes  |

PROGRAM USE:

This macro uses the libraries GRPH.LB and APS.LB; a link to these are required. It loads the main program DRVR and the subroutines DRSQ, S128, S64, DSTA, DSTN, PLTA, PLTO, PLTS, PLTT, PLTN, CHKO, CHKJ, CHOOS, FCTR, and GFDB.

DELETE/V DRVR.LM  
RLBR/C/R/E/P 2000/M DRVR [DRSQ S128 S64,DSTA,DSTM CHOOS CHXJ,\*  
PLTO PLTA PLTS PLTN PLTT] CHKO FCTR GFDB \*  
DRVR.LM/L GRPH.LB APS.LB @FLIB

FILE: DRVR  
LANGUAGE: FORTRAN 5  
DATE: September 21, 1982  
AUTHOR: D. Martin  
SUBJECT: Acoustic Analysis  
CALLING SEQUENCE: DRVR  
DATE OF LAST REVISION: September 21, 1982

PURPOSE:

This program drives the acoustic analysis package developed by Captain Dan Martin for his thesis. When this program is executed, a menu is displayed from which the operator chooses an analysis option: to get spectrum, to get distances, to get best distances, or to plot results.

DESCRIPTION:

Location: DP4:BRATCHET

|       |         |             |
|-------|---------|-------------|
| Size: | DRVR.SV | 51200 bytes |
|       | DRVR.FR | 2613 bytes  |
|       | DRVR.RB | 3054 bytes  |
|       | DRVR.OL | 40960 bytes |
|       | DRVR.LM | 9491 bytes  |

PROGRAM USE:

CAUTION. The Array Processor must be properly initialized prior to any attempt to run this routine. A macro file, LDAPS.MC, in DP4:BRATCHET will accomplish this initialization. Also, at least 14 blocks of extended memory must be allocated to the ground on which this routine is run. Further, DRVR.OL, GRPH.LB, and APS.LB

need be accessible to the directory in which this routine is run.

This program is not called by any routine and calls routines DRSQ, DSTA, DSTN, and PLTO. When executed, this program presents a menu from which the operator selects one of the following options:

- (1) To compute observation spectrum from a speech file.
- (2) To compute distances between observations and phonets. Phonets are observations set aside to represent recognizable units of speech.
- (3) To compute distances between observations and phonets as in the previous option, but to choose a selected number of best matches based on those distances.
- (4) To plot files on the Tektronix 4010-1 Graphics Terminal.

LIST OF VARIABLES:

| <u>VARIABLE</u> | <u>TYPE</u> | <u>PURPOSE</u>                                                                                  |
|-----------------|-------------|-------------------------------------------------------------------------------------------------|
| XMEM            | Integer     | Number of available extended memory blocks.                                                     |
| INITMEM         | Integer     | Number of extended memory blocks over and above those used in the logical address space window. |
| SFREQ           | Real        | Sampling rate for DRSQ.                                                                         |
| DTARRAY         | Real Array  | Vector in array processor memory.                                                               |
| SKIP            | Integer     | Switch to select options.                                                                       |

### SWITCH SETTINGS:

| <u>SWITCH</u> |   | <u>SETTING</u>            |
|---------------|---|---------------------------|
| SKIP          | = | 1 to compute observations |
|               | = | 2 to compute distances    |
|               | = | 3 to select best matches  |
|               | = | 4 to plot files           |
|               | = | 5 to stop                 |

The remainder of this section is devoted to a discussion of: (1) our use of the S250 Array Processor, (2) extended memory management, and (3) modification of the Acoustic Analyzer (AA).

The Eclipse S250 Array Processor combines an Eclipse S250 CPU with a floating-point Array Processor. This allows high-speed computation on real and complex vectors. We used FORTRAN 5 as the application programming language and Data General's Array Processor Software (APS) running under Data General's mapped RDOS operating system. The AP/S250 contains both an independent pipelined floating-point multiplier and add/subtract compare units that operate simultaneously. Several multiply or add/subtract operations overlap during given time periods.

AP memory consists of an AP-mounted, 64 bit by 1K (8kByte), 20MSEC bipolar RAM. This memory is dual-ported which allowed us to address it by two methods:

- (1) We addressed APM as part of main memory in which case we mapped APM into our 32K word logical

address space. In this way, data transfer between APM and logical address space where unnecessary.

(2) We addressed APM as local memory by using AP instructions.

We used the two categories of APS routines: support routines and interface routines. With the support routines, we handled the following tasks:

- (1) AP initialization.
- (2) Setting up control blocks and default values.
- (3) AP memory management.
- (4) Miscellaneous utility tasks.

With the interface routines, we performed specific Array Processing operations, such as to multiply two arrays, for example. We used first level interface routines, referred to as V-routines. Before executing a V-routine, we defined all data and control words required by the APS for that V-routine. This setup is accomplished by calling support routines as specified in the V-routine description. Besides control block words, we used a number of special APS parameters to make our application program independent of actual main memory addresses or APM offsets.

We managed memory in our Acoustic Analyzer (AA) using memory mapping and static memory allocation. We chose to declare a nine-memory block window in logical address space into which we would map APM and other arrays which we used as ports to extended memory. These port arrays: IDSP,

IDOB in DRSQ, and DIST, FON, and OBS in DSTA and DSTN, were mapped through extended memory to access data with a minimum of disk accesses. This scheme is illustrated in Figure 7 for DSTA and DSTN. In extended memory, the first four memory blocks are dedicated to the AP; this fact cannot be altered by the application program. Our window in logical address space consists of the four real arrays: WKOBS, WKFON, OBS, and FON; and the one integer array DIST. Each array is 1024 elements long. Array Processor memory is mapped into WKOBS and WKFON. Arrays OBS, FON, and DIST are mapped through extended memory to access observation spectrum, phonet spectrum, or to store distances. From these arrays, data are parcelled to WKOBS and WKFON for processing. The scheme for DRSQ is the same except that the entire window is not used.

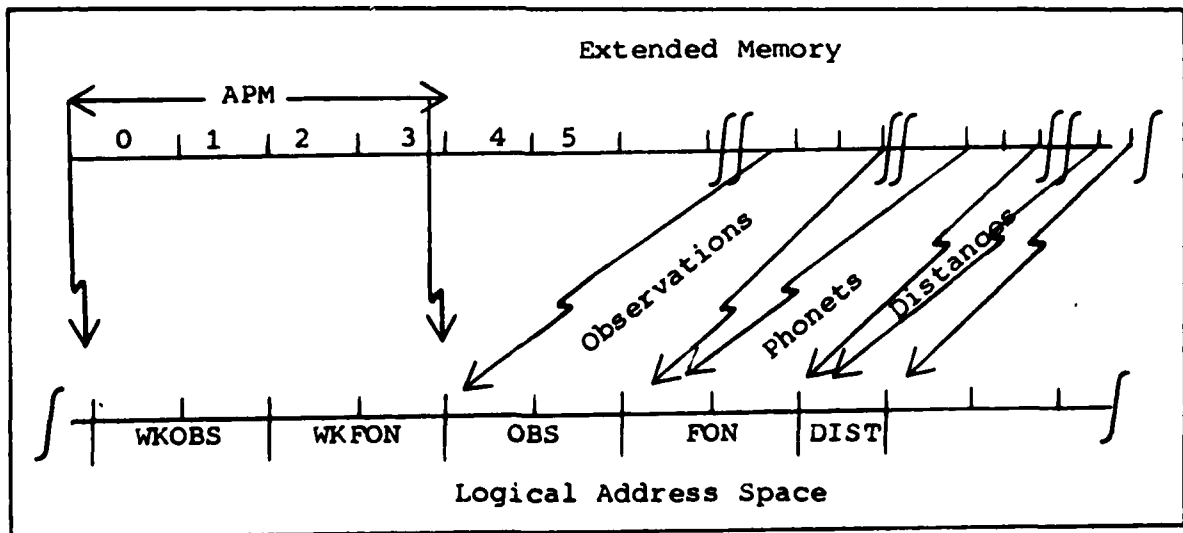


FIGURE 7. Memory Map for DSTA and DSTN

We have modified the Acoustic Analyzer for use by two students also doing their projects in the Speech Laboratory. We allowed them to run the spectral option and the distance option non-interactively by setting the switches in copies of the object code to their specifications. They now have access to the original interactive package with graphics capability as well as a tailored, non-interactive package capable of being run by MACRO files under RDOS. They use the output of the tailored Acoustic Analyzer as input to Montgomery's word recognition machine (Ref 10) which is also implemented on the Eclipse S/250 in the Speech Laboratory.

RELATED PROGRAMS:

PLTA, PLTS, PLTN, PLTO, PLTT, DRSQ, S128, S64, DSTA, DSTN, CHKJ, CHKO, CHOOS, FCTR, GFDB, GFPH2.

```

C***** ROUTINE DRV.R FR  NOTE: THIS ROUTINE IS FOR FORTRAN 5 ! !
C  THIS ROUTINE DRIVES DRSD,S128,S64,DSTA,DSTN,CH008
C  CHKJ,PLTO,PLTA,PLTS,PLTN,PLTT,CHKO,FCTR,CFDB.
C  BY:  CAPT DAN MARTIN
C  DATE:  9/27/82
C  SUBJ:  ACOUSTIC ANALYSER
C  THIS ROUTINE DRIVES THE ACOUSTIC ANALYSIS PACKAGE
C  I DEVELOPED.  THE OPERATOR IS PROMPTED TO SPECIFY:
C      1)  AMOUNT OF EXTENDED MEMORY TO USE,
C      2)  TASK TO DO
C          #GET SPECTRUM
C          #GET DISTANCES
C          #GET N-BEST DISTANCES
C          #GET PLOTS
C          #STOP
C  FOR DETAILS, SEE THE USERS MANUAL OR MY THESIS.

```

```

INCLUDE "ARRAYP:F5APS.FR"
EXTERNAL ODRSD,ODSTA,ODSTN,OPLTD
PARAMETER LM=512
PARAMETER NDP=1024
PARAMETER HLM=LM/2
REAL DTARAY,B,PREM,WIND,SFREQ,IDOB
INTEGER FILE1,IOPIN,IPRE,IDB,IHAM,FLEN,SKIP,INITMEN
INTEGER ISTART,ILAST,COUNT,FILE2,SMOOTH,WRTD,TEST,FLAG
INTEGER NBLKS,IFRQ(10),IAMP(10),XMEN,WRTDA,CB1,DUMMY
INTEGER CNTBR,IDSP,SXMEN,GSPCT,IVAL
COMMON /APM/ DTARAY(NDP),B(NDP)
COMMON / VALS / IDSP(NDP),IDOB(NDP),IHAM,IPRE,WIND(LM)
COMMON / VALS / PREM(LM),INITMEN,SXMEN,DUMMY(306)
COMMON / VALT / SKIP,COUNT,SMOOTH,WRTD,NBLKS,XMEN,SFREQ,FLEN
COMMON / VALT / FILE1(13),FILE2(13),FLAG,CNTBR,ISXMEN
COMMON / VALT / JAOUT,LASTCD
COMMON / VALV / CB1(0:CBMAX)
COMMON / VALU / IVAL(2058)

30  CALL VMEN(XMEN,IER)
    INITMEN=XMEN-9
    XMEN=XMEN-1
35  TYPE"YOU HAVE",XMEN," 1K BLOCKS OF EXT MEMORY ACCESSABLE"
40  ACCEPT"***HOW MANY EXTENDED MEMORY 1K BLOCKS WILL YOU USE? ",IXMEN
    IF(IXMEN.LE.XMEN)GOTO 45
    TYPE"YOU DON'T HAVE THAT MUCH MEMORY AVAILABLE."
    GOTO 35
45  XMEN=IXMEN-10
    IF(XMEN.GT.0)GOTO 50
    IXMEN=1-XMEN
    TYPE"***NEED SPECIFY AT LEAST",IXMEN," MORE BLOCKS!"
    GOTO 30
50  XMEN=IXMEN-9 ;LOSE 9 BLKS TO THE WINDOW
    ISXMEN=XMEN
    SFREQ=8000.0

```

```

CALL RESET
CALL OVOPN(20,"DRVR.OL",IER)
CALL CHECK(IER)

C***** INITIALIZE ARRAY PROCESSOR AND MEMORY MAP
CALL APINIT(NIL,DTARAY,9,INITHEM,IER)
CALL APHAP(DTARAY,0,4,IER)

80  TYPE"***WHAT WILL YOU HAVE ME DO?"
    TYPE"***CHOOSE OPTION***"
    TYPE" "
    TYPE"1 TO GET SPECTRUM."
    TYPE"2 TO GET DISTANCES."
    TYPE"3 TO GET N-BEST DISTANCES."
    TYPE"4 TO GET PLOTS."
    TYPE"5 TO STOP."
    ACCEPT"***ENTER OPTION DESIRED: ",SKIP

100  IF(SKIP.NE.1)GOTO 110
     CALL OVLOD(ODRSQ,-1,IER)
     CALL CHECK(IER)
     CALL DRSQ
     GOTO 80

110  IF(SKIP.NE.2)GOTO 120
     CALL OVLOD(ODSTA,-1,IER)
     CALL CHECK(IER)
     CALL DSTA
     GOTO 80

120  IF(SKIP.NE.3)GOTO 130
     CALL OVLOD(ODSTN,-1,IER)
     CALL CHECK(IER)
     CALL DSTN
     GOTO 80

130  IF(SKIP.NE.4)GOTO 140
     CALL OVLOD(OPLTO,-1,IER)
     CALL CHECK(IER)
     CALL PLTO
     GOTO 80

140  IF(SKIP.EQ.5)GOTO 1000
     TYPE"ERROR: COULDN'T FIND YOUR OPTION."
     TYPE"YOU SELECTED OPTION: ",SKIP
     GOTO 80

1000 CALL RESET
     STOP
     END

```

FILE: DRSQ  
LANGUAGE: FORTRAN 5  
DATE: September 21, 1982  
AUTHOR: D. Martin  
SUBJECT: Acoustic Analysis  
CALLING SEQUENCE: DRSQ  
DATE OF LAST REVISION: September 21, 1982

PURPOSE:

This program calls subroutines S128 and S64 to compute the Fast Fourier Transform (FFT) of an input file. It uses the Eclipse AP/S250 Array Processor. Although flexible, these routines were designed to transform speech files produced by the Cromemco analog-to-digital system.

DESCRIPTION:

Location: DP4:BRATCHET

|       |         |             |
|-------|---------|-------------|
| Size: | DRSQ.FR | 11400 bytes |
|       | DRSQ.RB | 15840 bytes |

PROGRAM USE:

This program is called by DRVR and calls S128, S64, and CHKO. This is a flexible routine in that several options are available with regard to windowing of input files, emphasis and scaling of output files. A block diagram is included at Figure 72.

When this routine is called, it prompts the operator for the following values and options:

- (1) To pre-emphasize high frequencies or not?

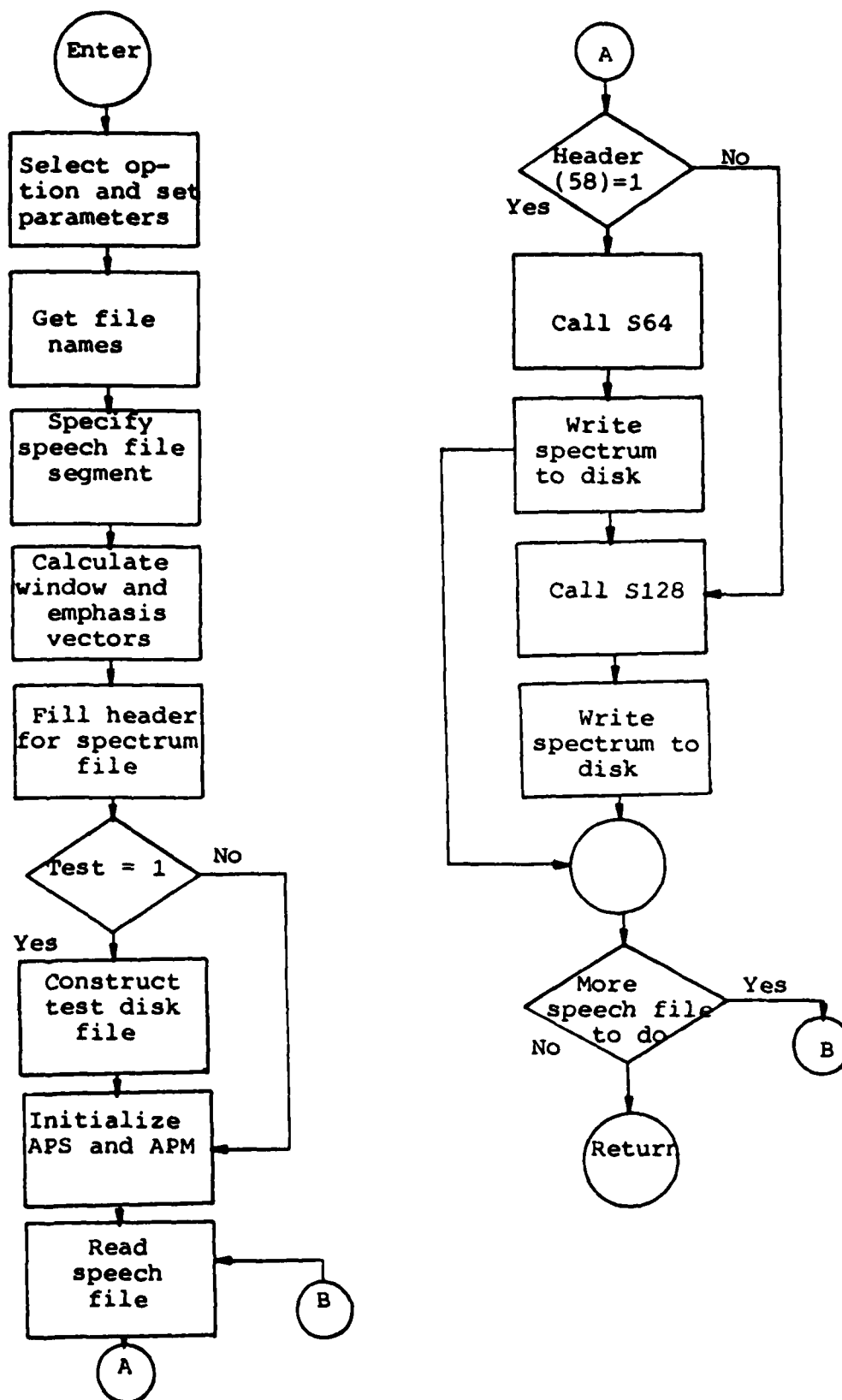


FIGURE 72. Flowchart of DRSQ

(2) If pre-emphasis is selected, at what rate in dB/Octave, and corner frequency?

(3) To de-emphasize low frequencies or not?

(4) If de-emphasis is selected, at what rate in dB/Octave, and corner frequency?

(5) What FFT window length? Any multiple of two between 16 and 1024, inclusive, can be selected.

(6) Choose Hamming or rectangular window. If a rectangular window is selected, no window vector is computed. If a Hamming window is selected, the window vector is calculated prior to the S128 or S64 call.

(7) If a window length of 128 points is selected, the program will prompt for a specification to overlap or not. If overlap is selected, the window will be moved along the input file in 64 point increments.

(8) To either normalize the observation energy to preset value or to divide each spectral component by the observation energy. The observation energy is the sum of the squares of each spectral component: one through  $N/2-1$ .

(9) Specify whether to write spectrum to disk only, to the terminal as well, or to the line printer as well.

(10) Specify if a test disk file is to be created and processed, or a speech file is to be read from disk and processed. DRSQ can generate a disk file with a signal

is a sum of up to ten tones at as many amplitudes. Operator is prompted for these values if the test option selected.

(11) Specify speech file name.

(12) Specify a disk file name in which to store observations. If this file does not exist, it will be created.

(13) Specify the portion of the speech file to process. This is done by specifying the first and last blocks to process.

A header is prepared and written to disk block zero. Block 1 contains the header assignments for the nontest option, Table 2 for the test option.

Extended memory blocks zero through eight are unavailable for use; they are used by the window declared in local address space. The speech file is read into all available extended memory blocks and array IDSP is mapped through extended memory accessing data. These data are moved from IDSP to Array Processor memory in FFT times. After the FFT computation, the real spectrum is transferred through real array IDOB to extended memory. If extended memory contents have been replaced with observations, its contents are written to disk.

| HEADER ELEMENT | CONTENT                                       |
|----------------|-----------------------------------------------|
| 1-13           | Observation file name.                        |
| 14-26          | Speech file name.                             |
| 27             | IPRE, pre-emphasis switch.                    |
| 28             | IDB, pre-emphasis rate (dB/Octave).           |
| 29             | FREQ, frequency at which pre-emphasis starts. |
| 30             | FLEN, FFT window size.                        |
| 31             | IHAM, window switch.                          |
| 32             | Unused (set to zero).                         |
| 33             | TEST, test option switch.                     |
| 34-54          | Unused (set to zero).                         |
| 55             | COUNT, number of first time slice to do.      |
| 56             | LTSTDO, number of last time slice to do.      |
| 57             | HL, number of elements, per observation.      |
| 58             | Overlapping switch.                           |
| 59             | Number of disk blocks in observation file.    |
| 60             | IDPRE, de-emphasis switch.                    |
| 61             | DDP, de-emphasis rate (dB/Octave).            |
| 62             | DFREQ, frequency at which de-emphasis ends.   |
| 63-256         | Unused (set to zero).                         |

TABLE 1. Header Assignments for Spectral File Computed by DRSQ Calling S128 or S64.

Test file header - but the header isn't written to disk.

| HEADER ELEMENT | CONTENT                                                     |
|----------------|-------------------------------------------------------------|
| 1-13           | Test file name.                                             |
| 14-26          | Observation file name.                                      |
| 27             | Switch: 1 = pre-emphasize/0 =<br>don't pre-emphasize.       |
| 28             | Pre-emphasis slope.                                         |
| 29             | Pre-emphasis corner frequency.                              |
| 30             | Number of time points-per-FFT.                              |
| 31             | Switch: 1 = Hamming window/0 =<br>rectangular window.       |
| 32             | Unused.                                                     |
| 33             | Switch: 1 = create test file/0 =<br>don't create test file. |
| 34             | Number of tones in test file.                               |
| 35-44          | Tone frequencies (HZ).                                      |
| 45-54          | Tone amplitudes.                                            |
| 55-57          | Unused.                                                     |
| 58             | Switch: 1 = overlapping time<br>slice/0 = nonoverlapping.   |
| 59-256         | Unused.                                                     |

TABLE 2. Header Assignments for Test Option.

LIST OF VARIABLES:

| <u>VARIABLE</u> | <u>TYPE</u>   | <u>PURPOSE</u>                           |
|-----------------|---------------|------------------------------------------|
| XMEM            | Integer       | Number of extended memory blocks to use. |
| IOPTION         | Integer       | Switch to set parameter options.         |
| IPRE            | Integer       | Pre-emphasis switch.                     |
| IDB             | Integer       | dB per Octave pre-emphasis.              |
| FREQ            | Integer       | Starting pre-emphasis frequency.         |
| IDPRE           | Integer       | De-emphasis switch.                      |
| DDB             | Integer       | dB per Octave de-emphasis.               |
| DFREQ           | Integer       | Ending de-emphasis frequency.            |
| FLEN            | Integer       | Number of points in FFT window.          |
| IHAM            | Integer       | Switch to choose window.                 |
| HEADER          | Integer Array | Holds header values.                     |
| NORM            | Integer       | Switch to select energy normalization.   |
| WRTD            | Integer       | Switch to select output option.          |
| TEST            | Integer       | Switch to select input option.           |
| FILE1           | Integer Array | To hold input file name.                 |
| FILE2           | Integer Array | To hold output file name.                |
| ISTART          | Integer       | First input disk block.                  |
| ILAST           | Integer       | Last input disk block.                   |
| LTSTDO          | Integer       | Last time slice to do.                   |

| <u>VARIABLE</u> | <u>TYPE</u>   | <u>PURPOSE</u>                             |
|-----------------|---------------|--------------------------------------------|
| COUNT           | Integer       | First time slice to do.                    |
| WIND            | Real Array    | Holds window values.                       |
| PREM            | Real Array    | Holds emphasis values.                     |
| FREQ1           | Integer       | First component to pre-emphasize.          |
| DFREQ           | Integer       | Last component to de-emphasize.            |
| IDSP            | Integer Array | Holds time file.                           |
| DTARAY          | Real Array    | First Array Processor Memory (APM) vector. |
| COUNTDOWN       | Integer       | Counts time slices done.                   |
| WRTDA           | Integer       | Output option switch.                      |
| NBLKS           | Integer       | Number of disk blocks to do.               |
| LASTCD          | Integer       | Counts time slices to do.                  |

SWITCH SETTINGS:

| <u>SWITCH</u> | <u>SETTING</u>                                                                                                                                   |
|---------------|--------------------------------------------------------------------------------------------------------------------------------------------------|
| IDPRE         | = 1 to de-emphasize<br>= 0 to not de-emphasize                                                                                                   |
| IHAM          | = 1 for Hamming window<br>= 0 for rectangular window                                                                                             |
| HEADER(58)    | = 1 to overlap time slices<br>= 0 to not overlap                                                                                                 |
| NORM          | = 1 to normalize energy in time slice<br>= 0 to divide spectrum by time slice energy                                                             |
| WRTD          | = 0 to write output to disk only<br>= 10 to write spectrum to terminal as well as disk<br>= 12 to write spectrum to line printer as well as disk |

| <u>SWITCH</u> | <u>SETTING</u>                                                    |
|---------------|-------------------------------------------------------------------|
| TEST          | = 1 to create an integer disk file of tones<br>to simulate speech |
|               | = 0 to process speech                                             |
| IOPTION       | = 0 for ability to set parameters                                 |
|               | = 20-27 are option choices                                        |
| IPRE          | = 1 to pre-emphasize                                              |
|               | = 0 to not pre-emphasize                                          |
| SKIP          | = 0 to not execute APS and APM initialization                     |
|               | = 1 never                                                         |

RELATED PROGRAMS:

PLTA, PLTN, PLTS, PLTO, DSTA, DSTN, CHOOS, CHKJ, CHKO,  
S128, S64, GFDB, GRPH2, FCTR, PLTT, DRVR.

```

C***** SUBROUTINE DRSQ NOTE: THIS SUBROUTINE IS FOR FORTRAN 5 !!
C THIS SUBROUTINE DRIVES S128, S64. IT IS CALLED BY DRVR.SV.
C BY: CAPT DAN MARTIN
C DATE: 9/21/82
C SUBJ: ACOUSTIC ANALYSIS
C THIS ROUTINE SETS UP NECESSARY FILES AND SWITCHES FOR
C SUBROUTINES S128 AND S64, WHICH COMPUTE FOURIER TRANSFORMS
C OF SPEECH FILES. THE OPERATOR IS PROMPTED FOR NECESSARY
C INFORMATION.
C DRSQ DOES THE FOLLOWING:
C 1) GETS PARAMETERS AND OPTIONS
C 2) GETS SPEECH FILE
C 3) GETS OBSERVATION (SPECTRAL) FILE
C 4) GETS SPEECH FILE SEGMENT, SPECIFIED AS
C 1ST AND LAST DISK BLOCKS
C 5) COMPUTES WINDOW VECTOR
C 6) COMPUTES EMPHASIS VECTOR
C 7) CONSTRUCTS SPECTRAL FILE HEADER
C 8) CONSTRUCTS TEST SPEECH FILE IF THAT OPTION
C WAS SELECTED
C 9) SETS UP WINDOW AND MEMORY MAP FOR
C ARRAY PROCESSOR
C 10) READS SPEECH FILE INTO EXTENDED MEMORY
C 11) CALLS EITHER S128 OR S64 TO GET SPECTRUM
C 12) WRITES SPECTRUM FROM EXTENDED MEMORY TO
C THE OBSERVATION FILE
C 13) READS MORE SPEECH IF THERE IS MORE TO DO
C 14) RETURNS TO DRVR
C FOR MORE INFORMATION, SEE THE USERS MANUAL OR MY THESIS.

```

```

SUBROUTINE DRSQ
OVERLAY ODRSQ

```

```

INCLUDE "ARRAYP:FSAPS.FR".
PARAMETER LM=512
PARAMETER NDP=1024
PARAMETER HLM=LM/2
REAL DTARAY,B,PREM,WIND,SFREQ,IDOB
INTEGER FILE1,IOPIN,IPRE,IDPRE,IDB,IHAN,FLEN,SKIP,HEADER(256),IDSP
INTEGER ISTART,ILAST,COUNT,FILE2,SMOOTH,WRTD,TEST,DUMMY,FREQ
INTEGER NBLKS,IFRQ(10),IAMP(10),XMEN,WRTDA,FLAG,CB1,CNTBR
INTEGER INITHEN, SXMEN,GSPCT,LTSTDO,COUNTDOWN,NORM,DFREQ,DOB
COMMON /APM/ DTARAY(NDP),B(NDP)
COMMON / VALS / IDSP(NDP),IDOB(NDP),IHAN,IPRE,WIND(LM)
COMMON / VALS / PREM(LM),INITHEN,SXMEN,DUMMY(304),COUNTDOWN
COMMON / VALS / LTSTDO
COMMON / VALT / SKIP,COUNT,SMOOTH,WRTD,NBLKS,XMEN,SFREQ,FLEN
COMMON / VALT / FILE1(13),FILE2(13),FLAG,CNTBR,ISXMEN
COMMON / VALT / JAOUT,LASTCD
COMMON / VALV / CB1(0:CBMAX)
COMMON / VALU / NORM

```

```

C***** Define variables used:
4 FLAG=0

```

```

XMEM=ISXMEM
C***** GET SPECTRAL OPTIONS
5 TYPE"      Several sets of defaults exist for the"
  TYPE"      following parameters: "
  TYPE"      1) PREEMPHASIS (YES/NO)"
  TYPE"      2) PREEMPHASIS SLOPE (dB/octave)"
  TYPE"      3) PREEMPHASIS STARTING FREQUENCY"
  TYPE"      4) DEEMPHASIS (YES/NO)"
  TYPE"      5) DEEMPHASIS SLOPE"
  TYPE"      6) LAST FREQUENCY"
  TYPE"      7) FFT VECTOR LENGTH"
  TYPE"      8) WINDOW TYPE"
  TYPE"      They are: "

```

```

TYPE"      OPTION NUMBER"

```

| TYPE"        | #21 | #22 | #23 | #24 | #25 | #26"  |
|--------------|-----|-----|-----|-----|-----|-------|
| TYPE"PREEMPH | Y   | N   | Y   | N   | Y   | NO"   |
| TYPE"SLOPE   | 10  | NA  | 10  | NA  | 10  | NA"   |
| TYPE"START   | 500 | NA  | 500 | NA  | 500 | NA"   |
| TYPE"DEEMPH  | Y   | N   | Y   | N   | N   | N"    |
| TYPE"SLOPE   | 10  | NA  | 10  | NA  | NA  | NA"   |
| TYPE"LAST    | 300 | NA  | 300 | NA  | NA  | NA"   |
| TYPE"FFT LEN | 64  | 64  | 128 | 128 | 128 | 64"   |
| TYPE"WINDOW  | HM  | HM  | HM  | HM  | HM  | RECT" |

```

TYPE"      If you choose to enter values for these parameters from"
TYPE"      the keyboard, enter numeral zero. If you choose a default"
ACCEPT"      option, enter the option desired: ",IOPTN

```

```

IF(IOPTN.GT.20.AND.IOPTN.LT.27)GOTO 2050

```

```

ACCEPT"PREEMPHASIZE HIGH FREQ ? (1=YES/0=NO): ",IPRE
IF(IPRE.EQ.1)ACCEPT"ENTER dB/OCTAVE: ",IDB
IF(IPRE.EQ.1)ACCEPT"ENTER STARTING FREQ: ",FREQ
ACCEPT"DEEMPHASIZE LOW FREQ? (1=YES/0=NO): ",IDPRE
IF(IDPRE.EQ.1)ACCEPT"ENTER DB/OCTAVE: ",ddb
IF(IDPRE.EQ.1)ACCEPT"ENTER LAST FREQ: ",dfreq
TYPE"ENTER FFT VECTOR LENGTH (SAMPLES/VECTOR). LENGTH MUST"
ACCEPT"BE A MULTIPLE OF 2 AND IN THE RANGE [16,1024]: ",FLEN
TYPE"WHAT TYPE OF WINDOW IS DESIRED?"
ACCEPT"HAMMING / RECTANGULAR (1/0): ",IHAM

```

```

GOTO 50
2050 IF(IOPTN.EQ.21)GOTO 21
  IF(IOPTN.EQ.22)GOTO 22
  IF(IOPTN.EQ.23)GOTO 23
  IF(IOPTN.EQ.24)GOTO 24
  IF(IOPTN.EQ.25)GOTO 25
  IF(IOPTN.EQ.26)GOTO 26
GOTO 50

```

```

21 IPRE=1 ;preemphasize

```

```

IDB=10                                ;dB/octave
FREQ=500                              ;start preemphasis here
IDPRE=1
DDB=10
DFREQ=300
FLEN=64
IHAM=1
GOTO 50
22  IPRE=0
    IDB=0
    FREQ=500
    IDPRE=0
    DDB=0
    DFREQ=0
    FLEN=64
    IHAM=1
    GOTO 50
23  IPRE=1
    IDB=10
    FREQ=500
    IDPRE=1
    DDB=10
    DFREQ=300
    FLEN=128
    IHAM=1
    GOTO 50
24  IPRE=0
    IDB=0
    FREQ=500
    IDPRE=0
    DDB=0
    DFREQ=0
    FLEN=128
    IHAM=1
    GOTO 50
25  IPRE=1
    IDB=10
    FREQ=500
    IDPRE=0
    DDB=0
    DFREQ=0
    FLEN=128
    IHAM=1
    GOTO 50
26  IPRE=0
    IDB=0
    FREQ=500
    IDPRE=0
    DDB=10
    DFREQ=300
    FLEN=64
    IHAM=0
    GOTO 50

```

```

C***** SET PARAMETERS
50    CONTINUE
      DO 415 J5=1,256
415    HEADER(J5)=0
      IF(FLEN.NE.128)GOTO 52
      TYPE"***ENTER '1' FOR OVERLAPPING"
      ACCEPT"***      '0' FOR NON-OVERLAPPING: ",HEADER(56)
52    CONTINUE
      TYPE"***ENTER '1' TO NORMALIZE ENERGY IN FILE TO UNITY"
      ACCEPT"      OR '0' TO DIVIDE SPECTRUM BY FILE ENERGY: ",NORM
      PARAMETER L=FLEN
      PARAMETER HL=L/2
C***** GET OUTPUT OPTION
      TYPE"ENTER OUTPUT OPTION: "
      TYPE"*0 TO WRITE SPECTRUM TO DISK (AND NO OTHER OUTPUT),"
      TYPE"*10 TO WRITE SPECTRUM TO SCREEN AS WELL AS DISK,"
      ACCEPT"*12 TO PRINT SPECTRUM AS WELL AS WRITE TO DISK. ",WRID
C***** GET TEST OPTION
      TYPE"GET TIME-FILE TYPE"
      TYPE"***ENTER A '1' TO CREATE A DISK FILE TO SIMULATE SPEECH: "
      ACCEPT"***ENTER A '0' TO CRUNCH SPEECH: ",TEST
      IF(TEST.EQ.0)GOTO 185
      ACCEPT"ENTER TEST DISK FILENAME: "
      READ(11,10)FILE1(1)
8    CALL OPEN(5,FILE1,2,IER)
      IF(IER.NE.1)TYPE"ERROR ON OPEN OF TEST DISK FILE, IER= ",IER
      CALL CHK0(IER)          ;SEE IF UNIT = IN USE
      IF(IER.NE.13)GOTO 9
      TYPE"FILE DOES NOT EXIST. WILL CREATE IT FOR YOU."
      CALL CFILW(FILE1,2,IER1)
      IF(IER1.NE.1)TYPE"ERROR ON FILE CREATION. IER1= ",IER1
      CALL CHECK(IER1)
      GOTO 8
9    CONTINUE
      GOTO 190
C***** GET SPEECH FILENAME (IF NOT TEST)
185    ACCEPT "ENTER NAME OF SPEECHFILE: "
      READ(11,10)FILE1(1)
      CALL OPEN(5,FILE1,1,IER1)
      IF(IER1.NE.1)TYPE"ERROR ON OPEN OF SPEECHFILE, IER1= ",IER1
      CALL CHK0(IER1)          ;SEE IF UNIT = OPEN
190    CONTINUE
C***** GET OBSERVATION FILENAME
      ACCEPT"ENTER NAME OF OUTPUT DISK FILE: "
      READ(11,10)FILE2(1)
10    FORMAT(S13)
12    CALL OPEN(4,FILE2,2,IER)
      IF(IER.NE.1)TYPE"ERROR ON OPEN OF OBSERVATION FILE, IER= ",IER
      CALL CHK0(IER)          ;SEE IF UNIT = OPEN
      IF(IER.NE.13)GOTO 15
      TYPE"FILE DOES NOT EXIST. WILL CREATE IT FOR YOU."
      CALL CFILW(FILE2,2,IER1)
      IF(IER1.NE.1)TYPE"ERROR ON FILE CREATION. IER1= ",IER1
      CALL CHECK(IER1)

```

```

      GOTO 12
15      CONTINUE
C***** SPECIFY PORTION OF SPEECH FILE TO GET SPECTRUM OF, OR LENGTH
C      OF TEST INPUT FILE

      TYPE"***SPECIFY SPEECH FILE SEGMENT "
      TYPE"(MULTIPLE OF 4 DISK BLOCKS IF TEST FILE)"
      IF(TEST.EQ.0)GOTO 18
      TYPE"TEST TIME-DOMAIN FILE WILL START AT BLK #0"
      ISTART=0
      GOTO 19
18      ACCEPT"***ENTER FIRST BLOCK TO BE READ: ",ISTART
19      ACCEPT"***ENTER LAST BLOCK: ",ILAST
      IF(HEADER(58).EQ.0)LTSTDQ=256*(1+ILAST)/L
                                     ;LAST TIME-SLICE TO DO
      IF(HEADER(58).EQ.0)COUNT=ISTART*256/L+1
                                     ;1ST TIME-SLICE TO DO
      IF(HEADER(58).NE.0)LTSTDQ=256*(1+ILAST)/HL
                                     ;LAST TIME-SLICE TO DO
      IF(HEADER(58).NE.0)COUNT=ISTART*256/HL+1
                                     ;1ST TIME-SLICE TO DO

C***** COMPUTE WINDOW VECTOR
      TYPE"COMPUTING WINDOW VECTOR (IF REQUESTED)"
      IF(IHAM.NE.1)GOTO 210
      DO 200 J=1,L
      WIND(J)= 0.54 - (0.46*COS(6.2831853072*(J-1)/L))
200      CONTINUE

C***** CALCULATE EMPHASIS VECTOR
210      TYPE"CALCULATING EMPHASIS VECTOR (IF REQUESTED)"
      IF(IPRE.NE.1.AND.IDPRE.NE.1)GOTO 408
      DO 405 IN=1,HL
405      PREM(IN)=1.0
      IF(IPRE.NE.1)GOTO 295 ;TO NOT PREEMPHASIZE
      FREQ1=(FREQ/SFREQ)*L+1 ;FIRST FREQ PREEMPH
      DO 400 IN=FREQ1,HL
400      PREM(IN)=(IN/FREQ1)**(0.1660964*2>IDB)
295      CONTINUE

      IF(IDPRE.NE.1)GOTO 350 ;TO NOT DEEMPHASIZE
      FREQ1=(DFREQ/SFREQ)*L ;LAST FREQ
      DO 300 IN=1,FREQ1
300      PREM(IN)=(IN/FREQ1)**(0.1660964*2*DDB)
350      CONTINUE
408      CONTINUE
C***** CONSTRUCT HEADER FOR OUTPUT FILE
      DO 410 J5=1,13
410      HEADER(J5)=FILE2(J5)
      DO 412 J5=14,26
      J4=J5-13
412      HEADER(J5)=FILE1(J4)
      HEADER(27)=IPRE
      HEADER(28)=IDB

```

```

HEADER(29)=FREQ
HEADER(30)=FLEN
HEADER(31)=IHAM
HEADER(33)=TEST
HEADER(55)=COUNT
HEADER(56)=LTSTDO
HEADER(57)=HL
HEADER(60)=IDPRE
HEADER(61)=DDB
HEADER(62)=DFREQ
IF(IDPRE.EQ.1)IPRE=1
WRTDA=WRTD
IF(WRTD.EQ.0)WRTD=10
WRITE(WRTD,950)HEADER(1)
WRITE(WRTD,952)HEADER(14)
WRITE(WRTD,954)
WRITE(WRTD,956)(HEADER(J5),J5=27,31)
WRITE(WRTD,955)
WRITE(WRTD,956)(HEADER(J5),J5=60,62)
WRITE(WRTD,958)HEADER(33)
416 IF(TEST.EQ.1)GOTO 430
420 GOTO 500
C***** FILL TEST DISK FILE
430 IS=1
DO 432 JJ1=1,10
IFRQ(JJ1)=0
432 IAMP(JJ1)=0
ACCEPT"*ENTER 1ST FREQ IN HZ: ",IFRQ(IS)
445 ACCEPT"*ENTER AMPLITUDE: ",IAMP(IS)
ACCEPT"******WANT ANOTHER TONE? (1=YES/0=NO): ",IT
IF(IT.EQ.0)GOTO 455
IS=IS+1
IF(IS.GT.10)GOTO 455
ACCEPT"*ENTER ANOTHER FREQ IN HZ: ",IFRQ(IS)
GOTO 445
455 IF(IS.GT.10)IS=10
HEADER(34)=IS
DO 456 J5=35,44
J4=J5-34
456 HEADER(J5)=IFRQ(J4)
DO 457 J5=45,54
J4=J5-44
457 HEADER(J5)=IAMP(J4)
WRITE(WRTD,962)HEADER(34)
WRITE(WRTD,960)(HEADER(J5),J5=35,HEADER(34)+34)
WRITE(WRTD,960)(HEADER(J5),J5=45,HEADER(34)+44)
459 DO 495 JI=0,ILAST,4
DO 460 IT=1,NDP
460 IDSP(IT)=0.0
DO 470 IQ=1,IS
DO 450 IT=1,NDP
450 IDSP(IT)=IAMP(IQ)*COS(6.2831853072*IFRQ(IQ)*(IT-1)/SFREQ)+IDSP(IT)
470 CONTINUE
CALL WRBLX(5,JI,IDSP,4,IER)

```

```

495      CONTINUE

500      CALL FCTIME(IHOUR0,IMINO,ISECO)

      SKIP=0
C***** INITIALIZE APS AND APH MAP FOR ERDB
      IF(SKIP.EQ.0)GOTO 502
      SKIP=0
      CALL APINIT(NIL,DTARAY,9,INITMEN,IER)
      CALL APHMAP(DTARAY,0,4,IER)
502      JI=1                      ;DISK BLOCK TO WRITE TO
      JIH=1                      ;DISK BLOCK TO WRITE TO
503      IF(HEADER(58).EQ.0)XMEM=XMEM*4 ;GET # AVAIL QTR BLKS EXT MEN
      IF(HEADER(58).NE.0)XMEM=XMEM*2
504      COUNTDOWN=HEADER(56)-HEADER(55)+1
      TYPE"COUNTDOWN=",COUNTDOWN
      IF(WRTDA.EQ.0)WRTD=0
505      NBLKS=ILAST-ISTART+1
      IF(NBLKS.LE.XMEM)GOTO 510
      NBLKS=XMEM ;# QTR BLKS CAN DO
510      IF(HEADER(58).NE.0)GOTO 512
      COUNT=256*ISTART/L ;INITIALIZE TIME SLICE COUNT
      CALL ERDB(5,ISTART,36,NBLKS,ICNT,IER)
      GOTO 530
512      COUNT=256*ISTART/HL
      CALL ERDB(5,ISTART,36+XMEM,NBLKS,ICNT,IER)
530      CONTINUE
      IF(IER.NE.9)GOTO 850
      TYPE"READ END OF FILE! SUCCESSFULLY TRANSFERRED",ICNT," QTR BLOCKS."
      NBLKS=ICNT
      ILAST=ISTART+NBLKS-1
      TYPE"PROCEEDING WITH ",NBLKS," QTR BLOCKS TRANSFERRED."
      IER=1
850      IF(IER.EQ.1)GOTO 860
      TYPE"ERROR ON ERDB,IER= ",IER
      GOTO 1000
860      ISTART=ISTART+NBLKS
2000     FORMAT(5G12.4)
      IF(FLEN.EQ.128.AND.HEADER(58).NE.0)GOTO 890
      LASTCD=COUNTDOWN-NBLKS*256/L
      CALL S64
      CALL EWRB(4,JIH,36,NBLKS,ICNT,IER)
      GOTO 904
890      LASTCD=COUNTDOWN-NBLKS*256/HL
      CALL S128
902      CALL EWRB(4,JI,36,NBLKS*2,ICNT,IER)
904      IF(IER.NE.1)TYPE"ERROR ON EWRB, IER= ",IER
      IF(COUNTDOWN.LE.0.AND.IER.EQ.9)NBLKS=ICNT
905      JI=JI+NBLKS*2
      JIH=JIH+NBLKS
      TYPE"***COUNTDOWN=",COUNTDOWN
      IF(COUNTDOWN.LE.0)GOTO 911
910      IF(ISTART.LE.ILAST)GOTO 505 ;MORE SPEECHFILE TO DO

```

```

911      CONTINUE
        IF(FLEN.EQ.128.AND.HEADER(58).NE.0)GOTO 912
        HEADER(59)=JIH           ;# QB'S IN FILE
        GOTO 915
912      HEADER(59)=JI
915      CALL WRBLK(4,0,HEADER,1,IER)           ;WRITE HEADER TO 1ST QBLX
        CALL RESET
        WRTD=WRTDA
        CALL FCTIME(IHOUR,IMIN,ISEC)
        ISECT=ISEC-ISECO+60*(IMIN-IMINO)+3600*(IHOUR-IHOUR0)
        TYPE"TIME TO GET SPECTRUM IS",ISECT," SECONDS"

950      FORMAT("**** THIS FILE'S NAME IS: ",S13)
952      FORMAT(" TIME DOMAIN FILE NAME: ",S13)
954      FORMAT(7X,"IPRE",6X,"IDB",6X,"FREQ",6X,"FLEN",6X,"IHAN")
955      FORMAT(7X,"IDPRE",7X,"DDB",6X,"DFREQ")
956      FORMAT(6I10)
958      FORMAT(" ---- TEST = ",I4" ---")
960      FORMAT(" ",10I7)
962      FORMAT(2X,I4," TONES IN TEST FILE. HERE ARE FREQS AND AMPLITUDES")
1000     CALL OVEXIT(ODRSQ,IER)
        CALL CHECK(IER)
        RETURN
        END

```

E: S128 and S64  
LANGUAGE: FORTRAN 5  
E: September 21, 1982  
AUTHOR: D. Martin  
SUBJECT: Fourier Transform  
PROGRAMMING SEQUENCE: S128 or S64  
DATE OF LAST REVISION: September 21, 1982

DESCRIPTION:

Subroutine S128, called by DRSQ, computes overlapping point Fast Fourier Transforms (FFTs) on an integer array residing in extended memory. Subroutine S64 does not overlap the input window and will do FFTs of window sizes which are multiples of two in the range 16-1024, inclusive.

DESCRIPTION:

Location: DP4:BRATCHET

|       |         |            |
|-------|---------|------------|
| Size: | S128.FR | 4866 bytes |
|       | S128.RB | 5378 bytes |
|       | S64.FR  | 3910 bytes |
|       | S64.RB  | 4276 bytes |

GRAM USE:

These programs are called by DRSQ and call no subroutines. They use the Array Processor and extended memory to maximize the speed with which they calculate FFTs. These routines are similar and the flowchart, Figure 73, applies to both. The two differences between them are: (1) the window size of S128 is fixed at 128 points while that of S64 is selectable, (2) the window of S128 is moved along the

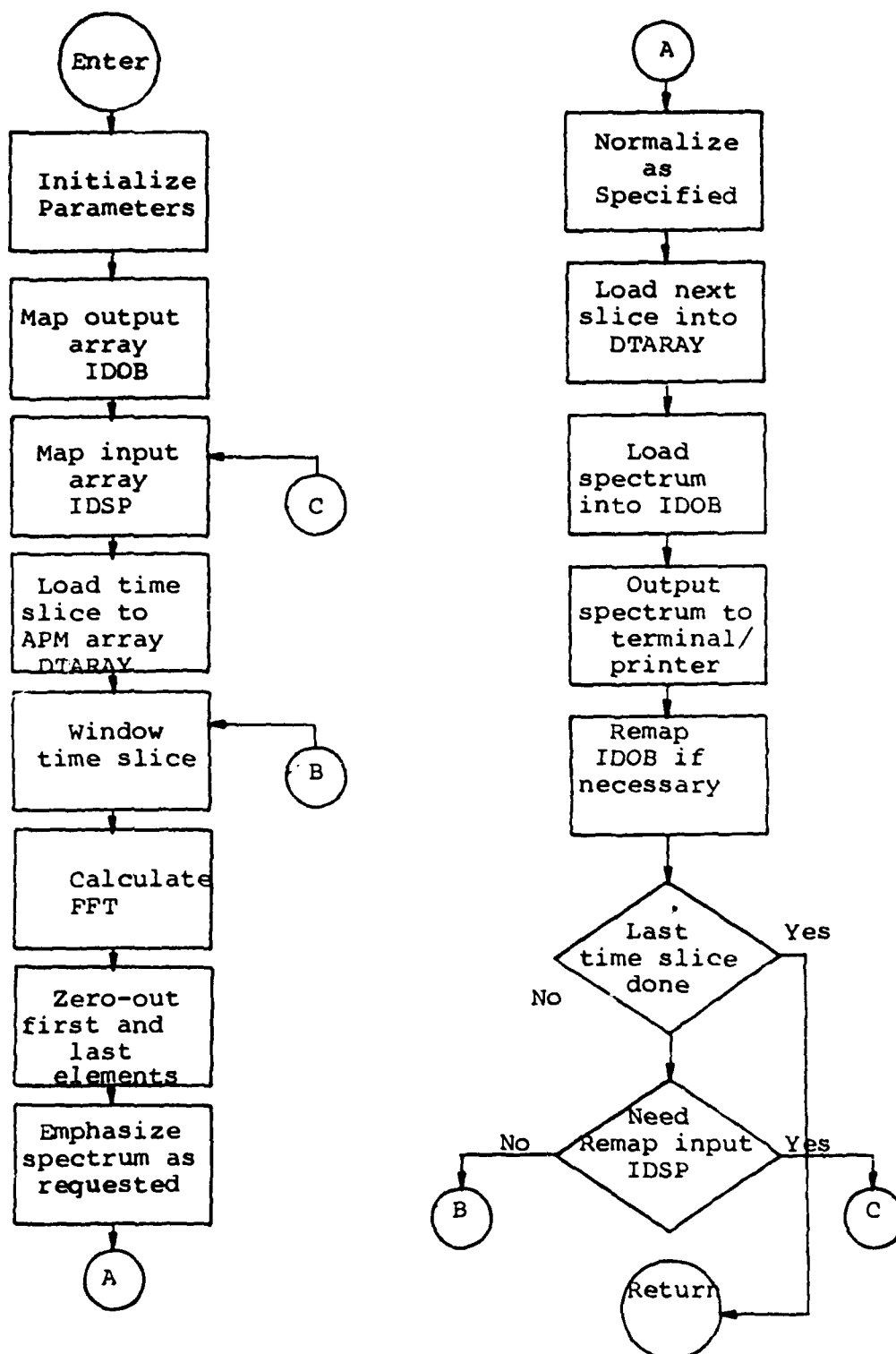


FIGURE 73. Flowchart of S128 and S64

input file in increments of 64 points at each FFT calculation while the window in S64 is moved along the input file in increments of its window size.

These differences require that extended memory be managed differently in the two routines. For S64, an input window of  $N$  integer points returns  $N/2$  real components which replace the input time slice-by-time slice. The spectrum is stored on disk in contiguous units called observations. The first element is the observation energy,  $E$ , computed as the sum of the squares of spectral components one through  $N/2-1$ . The Array Processor FFT routine returns  $N/2+1$  components; but prior to the energy computation, the zero and  $N/2$  components are set to zero. Each observation, then, consists of an energy value as its first element, and spectral components one through  $N/2-1$  in the remaining  $N/2-1$  elements. The  $K^{\text{th}}$  element of the observation is the  $K^{\text{th}}$  spectral component of the  $N$ - point time slice.

The observations are written to disk starting at disk block number one. Each  $N$ - point time slice requires  $N$  words of memory. Taking advantage of this, S64 stores the spectrum back into the storage locations previously occupied by the window. In this sense, the FFTs are computed in-place.

But because S128 overlaps its window by 64 points each FFT calculation, the in-place scheme will not work. The scheme S128 uses has DRSQ read the input file into the

upper half of available extended memory. Then as the input window is moved along the upper half, the output is stored in the lower half of extended memory, starting at the first available location. The input and output storage locations overlap only at the last locations at the top of extended memory.

Both routines transform an integer input file, such as that produced by the Cromemco analog-to-digital conversion system in the Speech Laboratory, into a real disk file. Disk block number zero is a header containing information which describes and identifies the file. This header is the same for both S128 and S64, and its assignments are listed in Table 3.

LIST OF VARIABLES:

| <u>VARIABLE</u> | <u>TYPE</u>   | <u>PURPOSE</u>                              |
|-----------------|---------------|---------------------------------------------|
| JAOUT           | Integer       | First element of output vector.             |
| DUMMY(1)        | Integer       | Last memory block read from.                |
| J6              | Integer       | First memory block read from.               |
| J6S             | Integer       | Signals first call.                         |
| XMEM            | Integer       | Extended memory available for input.        |
| DTARAY          | Real Array    | Array Processor memory.                     |
| IDSP            | Integer Array | Input array mapped through extended memory. |
| WIND            | Real Array    | Contains window to be applied to input.     |

| <u>VARIABLE</u> | <u>TYPE</u> | <u>PURPOSE</u>                                |
|-----------------|-------------|-----------------------------------------------|
| PREM            | Real Array  | Contains emphasis vector applied to spectrum. |
| NORM            | Integer     | Switch to select energy normalization.        |
| IPRE            | Integer     | Switch to select emphasis.                    |
| IHAM            | Integer     | Switch to select window type.                 |
| IDOB            | Real Array  | Output array mapped through extended memory.  |

SWITCH SETTINGS:

| <u>SWITCH</u> | <u>SETTING</u>                                                                                                                |
|---------------|-------------------------------------------------------------------------------------------------------------------------------|
| IHAM          | = 1 for Hamming window<br>= 0 for rectangular window                                                                          |
| IPRE          | = 1 to emphasize spectrum<br>= 0 to not                                                                                       |
| NORM          | = 1 to normalize spectral components to observation energy<br>= 0 to divide each spectral component by the observation energy |

RELATED PROGRAMS:

PLTA, PLTS, PLTN, PLTT, PLTO, DRVR, DSTA, DSTN, DRSQ,  
CHKJ, CHKO, CHOOS, FCTR, GFDB, GRPH2.

| HEADER ELEMENT | CONTENT                                       |
|----------------|-----------------------------------------------|
| 1-13           | Observation file name.                        |
| 14-26          | Speech file name.                             |
| 27             | IPRE, pre-emphasis switch.                    |
| 28             | IDB, pre-emphasis rate (dB/Octave).           |
| 29             | FREQ, frequency at which pre-emphasis starts. |
| 30             | FLEN, FFT window size.                        |
| 31             | IHAM, window switch.                          |
| 32             | Unused (set to zero).                         |
| 33             | TEST, test option switch.                     |
| 34-54          | Unused (set to zero).                         |
| 55             | COUNT, number of first time slice to do.      |
| 56             | LTSTD0, number of last time slice to do.      |
| 57             | HL, number of elements, per observation.      |
| 58             | Overlapping switch.                           |
| 59             | Number of disk blocks in observation file.    |
| 60             | IDPRE, de-emphasis switch.                    |
| 61             | DDP, de-emphasis rate (dB/Octave).            |
| 62             | DFREQ, frequency at which de-emphasis ends.   |
| 63-256         | Unused (set to zero).                         |

TABLE 3. Header Assignments for Spectral File Computed by DRSQ Calling S128 or S64.

```

C***** SUBROUTINE S128 NOTE: THIS ROUTINE IS FOR FORTRAN 5 !!
C      THIS SUBROUTINE CALLS ARRAY PROCESSOR ROUTINES TO CALCULATE 128
C      POINT FFT'S. IT OVERLAPS THE TIME WINDOWS BY 64 POINTS.
C      BY: CAPT DAN MARTIN
C      DATE: 9/21/82
C      SUBP: ACOUSTIC ANALYSIS
C      THIS ROUTINE IS CALLED BY DRSD WHICH SETS UP THE NECESSARY
C      FILE INPUT/OUTPUT FILES AND OPTIONS. THE INPUT TO THIS ROUTINE
C      RESIDES IN EXTENDED MEMORY AND THE OUTPUT IS DEPOSITED THERE.
C      S128 DOES THE FOLLOWING:
C          1) INITIALIZES PARAMETERS
C          2) MAPS OUTPUT ARRAY IDOB
C          3) MAPS INPUT ARRAY IDSP
C          4) LOADS TIME SLICE INTO DTARAY
C          5) WINDOWS TIME SLICE
C          6) CALCULATES FFT
C          7) ZEROS-OUT 1ST AND LAST ELEMENTS
C          8) EMPHASIZES SPECTRUM
C          9) NORMALIZES SPECTRUM
C          10) LOADS SPECTRUM INTO IDOB
C          11) REMAPS IDOB IF NECESSARY
C          12) REMAPS IDSP IF NECESSARY
C      FOR MORE INFORMATION, SEE THE USERS MANUAL OR MY THESIS.
C      SUBROUTINE S128
C      INCLUDE "ARRAYP:F5APS.FR" ; APS PARAMETER FILE

```

```

C***** Set up variables to be used.
PARAMETER LM=64
PARAMETER TLM=128
PARAMETER NDP=1024
PARAMETER MAXLM=512
REAL DTARAY,B,PREM,WIND,SFREQ
REAL SUNE,FACTOR,IDOB
INTEGER CB1,DUMMY,FLEN,FILE1,FILE2,FLAG,CNTBR,IDSP
INTEGER IHAM,IPRE,SKIP,COUNT,SMOOTH,WRTD,NBLKS,XMEM
INTEGER INITMEN, SXMEM, GSPCT, LTSTD0, COUNTDOWN, NORM

COMMON /APM/ DTARAY(NDP),B(NDP) ;AP MEMORY
COMMON / VALS / IDSP(NDP),IDOB(NDP),IHAM,IPRE,WIND(MAXLM)
COMMON / VALS / PREM(MAXLM),INITMEN,SXMEM,DUMMY(304),COUNTDOWN
COMMON / VALS / LTSTD0
COMMON / VALT / SKIP,COUNT,SMOOTH,WRTD,NBLKS,XMEM,SFREQ,FLEN
COMMON / VALT / FILE1(13),FILE2(13),FLAG,CNTBR,ISXMEM
COMMON / VALT / JAOUT, LASTCD
COMMON / VALU / CB1(0:CBMAX)
COMMON / VALU / NORM

```

```

C***** INITIALIZE APS AND APM MAP
IF(SKIP.EQ.0)GOTO 250
CALL APINIT(NIL,DTARAY,9,INITMEN,IER)
CALL APMAP(DTARAY,0,4,IER)
250 CONTINUE
JAOUT=1 ;1ST ELEMENT OF OUTPUT VECTOR

```

```

DUMMY(1)=9+(XMEM+NBLKS)/4          ;LAST MEM BLK READ FROM
J6=XMEM/4+9                        ;1ST MEM BLK READ FROM
J6S=J6                             ;SIGNALS 1ST ENTRY
J5=9
CALL APMAP(IDOB,J5,-2,IER)          ;HAP OUTPUT ARRAY
260  CONTINUE
    CALL APMAP(IDSP,J6,-1,IER)      ;HAP INPUT ARRAY
C***** ITERATE TO LINE 700
C***** THE FIRST ELEMENT OF DATA DOESN'T RESIDE ON A 1K BOUNDARY,
C      NECESSARILY, IN THE FIRST MAP. TO ACCOUNT
C      FOR THAT POSSIBILITY:
    IF(J6.NE.J6S)J1=1              ;GET 1ST ELEMENT OF IDSP
    IF(J6.EQ.J6S)J1=1+NBP*(FLOAT(XMEM)/4-IFIX(XMEM/4))
    J6=J6+1
C***** TRANSFER 128 PNTS FROM IDSP
280  DO 300 J2=J1,J1+TLM-1
    J3=J2-(J1-1)
300  DTARAY(J3)=IDSP(J2)

C***** WINDOW VECTOR DTARAY
305  IF(IHAM.NE.1)GOTO 310
    CALL APSETL(TLM,IER)
    CALL CBSET(CB1,CBAXR,B,CBAAMN,WIND,IER)
    CALL VLDR(CB1)                  ;LOAD WINDOW VECTOR
    CALL CBSET(CB1,CBAXR,DTARAY,CBAYR,B,CBAZR,DTARAY,IER)
    CALL VMRA(CB1)                  ;MULTIPLY VECTORS

310  CONTINUE
C***** CALCULATE FFT OF VECTOR DTARAY

    CALL APSETL(LM,IER)

    CALL CBSET(CB1,CBAXC,DTARAY,CBCW,CWDFT,IER)
    CALL VFFTC(CB1)
    CALL VBRC(CB1)
    CALL VFFTR(CB1)
C***** COMPLEX DTARAY CONTAINS 1ST LM+1 SPECTRUM COEFFICIENTS
C      NEED ZERO-OUT 1ST ELEMENT, I.E., ZERO-HERTZ TERM
C      ALSO NEED ZERO-OUT 65TH ELEMENT, I.E., 4KHZ TERM
    DTARAY(1)=0.0
    DTARAY(2)=0.0
    CALL CBSET(CB1,CBCW,CWSTD,CBAZR,DTARAY,IER)
    CALL VSMA(CB1)

400  IF(IPRE.NE.1)GOTO 406
    CALL CBSET(CB1,CBAXR,B,CBAAMN,PREM,IER)
    CALL VLDR(CB1)                  ;LOAD EMPHASIS VECTOR
    CALL CBSET(CB1,CBAZR,DTARAY,CBAYR,DTARAY,IER)
    CALL VMRA(CB1)                  ;MULTIPLY VECTORS
C***** GET ENERGY IN SPEECHFILE

```

```

406    CALL CBSET(CB1,CBAAMN,SUNE,CBAXR,DTARAY,IER)
      CALL VSER(CB1)                ;SUM VECTOR ELEMENTS
C***** SQR T SPECTRAL COMPONENTS
      DO 410 J4=1,LM
410    DTARAY(J4)=SQRT(DTARAY(J4))
C***** NORMALIZE ENERGY IN SPECTRAL FILE TO A CONSTANT
C      OR DO SOMETHING ELSE
      IF(NORM.EQ.0)GOTO 418
      FACTOR=10000/SQRT(SUNE)
      GOTO 420
418    FACTOR=10000000/SUNE
420    CONTINUE
      CALL CBSET(CB1,CBSCR,FACTOR,CBAZR,B,IER)
      CALL VMRS(CB1)                ;MULTIPLY VECTOR BY SCALER
C***** LOAD DTARAY WITH NEXT 128 PNTS
      J1=J1+LM                    ;MOVE OVER 64 PNTS
      IF(J1.EQ.1025)GOTO 429        ;NEED TO REMAP TO NEXT EXT MEM BLK
      IF(J1.EQ.961)GOTO 422         ;ONLY 64 PNTS LEFT
      J3=0
      DO 421 J2=J1,J1+127
      J3=J3+1
421    DTARAY(J3)=IDSP(J2)
      GOTO 429
422    J3=0
      DO 423 J2=J1,1024            ;LOAD LAST 64 PNTS OF CURRENT
      J3=J3+1                      ;MEMORY BLOCK
423    DTARAY(J3)=IDSP(J2)
      IF(J6.GT.DUMMY(1))GOTO 425    ;LAST MEM BLOCK
      CALL APMAP(IDSP,J6,-1,IER)
      DO 424 J2=65,128            ;LOAD 1ST 64 PNTS OF NEXT
      J3=J2-64                    ;MEMORY BLOCK
424    DTARAY(J2)=IDSP(J3)
      CALL APMAP(IDSP,J6-1,-1,IER)
      GOTO 429
425    DO 427 J3=65,128            ;NEED FUDGE LAST 64 PNTS
427    DTARAY(J3)=DTARAY(64)*EXP(64-J3)
C***** LOAD NORMALIZED SPECTRUM INTO IDOB
429    COUNT=COUNT+1
      COUNTDOWN=COUNTDOWN-1
      IDOB(JAOUT)=SQRT(SUNE)
      J3=1
      DO 430 J2=JAOUT+1,JAOUT+63  ;LOADING 64 PNTS
      J3=J3+1
430    IDOB(J2)=B(J3)
      JAOUT=JAOUT+LM
      IF(JAOUT.NE.1025)GOTO 435
      JAOUT=1
      J5=J5+2
      CALL APMAP(IDOB,J5,-2,IER)
435    CONTINUE
440    FORMAT(5G12.4)
      IF(COUNTDOWN.LE.LASTCD)GOTO 1000
700    CONTINUE
750    IF(J1.LE.961)GOTO 305

```

800 GOTO240

1000 RETURN  
END

```

C***** SUBROUTINE S64 NOTE: THIS ROUTINE IS FOR FORTRAN 5 !!
C THIS SUBROUTINE CALLS ARRAY PROCESSOR ROUTINES TO CALCULATE
C FFT'S. THE WINDOW SIZE CAN BE ANY MULTIPLE OF 2 IN THE RANGE
C [16,1024]. ALTHOUGH DESIGNED FOR SPEECH ANALYSIS, THIS
C ROUTINE IS FLEXIBLE.
C BY: CAPT DAN MARTIN
C DATE: 9/21/82
C SUBJ. ACOUSTIC ANALYSIS
C THIS ROUTINE IS CALLED BY DRSG WHICH SETS UP THE NECESSARY FILE
C INPUT/OUTPUT OPTIONS. THE INPUT TO THIS ROUTINE RESIDES IN
C EXTENDED MEMORY AND THE OUTPUT IS DEPOSITED THERE.
C S64 DOES THE FOLLOWING:
C 1) INITIALIZES PARAMETERS
C 2) MAPS OUTPUT ARRAY IDOB
C 3) MAPS INPUT ARRAY IDSP
C 4) LOADS TIME SLICE INTO DTARAY
C 5) WINDOWS TIME SLICE
C 6) CALCULATES FFT
C 7) ZEROS-OUT 1ST AND LAST ELEMENTS
C 8) EMPHASIZES SPECTRUM
C 9) NORMALIZES SPECTRUM
C 10) LOADS SPECTRUM INTO IDOB
C 11) REMAPS IDOB IF NECESSARY
C 12) REMAPS IDSP IF NECESSARY
C FOR MORE INFORMATION, SEE MY THESIS OR THE USERS MANUAL.

```

```

SUBROUTINE S64
INCLUDE "ARRAYP:F5APS.FR" ; APS PARAMETER FILE

```

```

C***** Set up variables to be used.
PARAMETER LM=FLEN
PARAMETER HLM=LM/2
PARAMETER KLM=HLM-1
PARAMETER NDP=1024
PARAMETER MAXLM=512
REAL DTARAY,B,PREM,WIND,SFREQ
REAL SUME,FACTOR,IDOB
INTEGER CB1,FLEN,FILE1,FILE2,FLAG,CNTBR,IDSP,INITMEN,IXMEN
INTEGER IHAM,IPRE,SKIP,COUNT,SMOOTH,WRTD,NBLKS,XMEN,DUMMY
INTEGER GSPCT,LTSTD0,COUNTDOWN,NORM
COMMON /APM/ DTARAY(NDP),B(NDP) ;AP MEMORY
COMMON / VALS / IDSP(NDP),IDOB(NDP),IHAM,IPRE,WIND(MAXLM)
COMMON / VALS / PREM(MAXLM),INITMEN,IXMEN,DUMMY(304),COUNTDOWN
COMMON / VALS / LTSTD0
COMMON / VALT / SKIP,COUNT,SMOOTH,WRTD,NBLKS,XMEN,SFREQ,FLEN
COMMON / VALT / FILE1(13),FILE2(13),FLAG,CNTBR,ISXMEN
COMMON / VALT / JAOUT,LASTCD
COMMON / VALV / CB1(0:CBMAX)
COMMON / VALU / NORM

C***** INITIALIZE APS AND APM MAP
IF(SKIP.EQ.0)GOTO 250
CALL APINIT(NIL,DTARAY,9,INITMEN,IER)

```

```

                CALL APHAP(DTARAY,0,4,IER)
250      J5=9                      ;MEM BLOCK
260      CONTINUE
                CALL APHAP(IDSP,J5,-1,IER)          ;INPUT ARRAY
                CALL APHAP(IDOB,J5,-1,IER)          ;OUTPUT ARRAY

                JAOUT=1                      ;POINTER IN OUTPUT VECTOR
C***** ITERATE DOING FFT'S BY TIME-SLICE
                DO 700 J1=1,NBP,LM
C      INPUT SEQUENCE IS INTEGER, OUTPUT IS REAL.  SO OUTPUT
C      TAKES PLACE OF INPUT ELEMENT FOR ELEMENT
                DO 300 J2=J1,J1+LM-1
                  J3=J2-(J1-1)
300      DTARAY(J3)=IDSP(J2)

C***** WINDOW VECTOR DTARAY
                IF(INAM.NE.1)GOTO 310

                CALL APSETL(LM,IER)
                CALL CBSET(CB1,CBAXR,B,CBAAMH,WIND,IER)
                CALL VLDR(CB1)                  ;LOAD WINDOW VECTOR
                CALL CBSET(CB1,CBAXR,DTARAY,CBAYR,B,CBAZR,DTARAY,IER)
                CALL VHRA(CB1)                  ;MULTIPLY VECTORS

310      CONTINUE
C***** CALCULATE FFT OF VECTOR DTARAY

                CALL APSETL(HLM,IER)

                CALL CBSET(CB1,CBAXC,DTARAY,CBCW,CWDFT,IER)
                CALL VFFTC(CB1)
                CALL VBRC(CB1)
                CALL VFFTR(CB1)
C***** DTARAY CONTAINS 1ST HLM+1 SPECTRUM COEFFICIENTS
C      NEED ZERO-OUT 1ST ELEMENT, I.E., ZERO-HERTZ TERM
C      ALSO NEED ZERO-OUT 33RD ELEMENT, I.E., 4KHZ TERM
                DTARAY(1)=0.0
                DTARAY(2)=0.0
                CALL CBSET(CB1,CBCW,CWSTD,CBAZR,DTARAY,IER)
                CALL VSMA(CB1)                  ;GET SQUARE MAGNITUDE

400      IF(IPRE.NE.1)GOTO 406
                CALL CBSET(CB1,CBAXR,B,CBAAMH,PREM,IER)
                CALL VLDR(CB1)                  ;LOAD ARRAY
                CALL CBSET(CB1,CBAZR,DTARAY,CBAYR,DTARAY,IER)
                CALL VHRA(CB1)                  ;MULTIPLY VECTORS

406      CALL CBSET(CB1,CBAAMH,SUM,CBAXR,DTARAY,IER)
                CALL VSR(CB1)                  ;SUM ELEMENTS
C***** SORT SPECTRAL COMPONENTS
                DO 420 J4=2,HLM

```

```

420      DTARAY(J4)=SQRT(DTARAY(J4))
C***** NORMALIZE ENERGY IN SPECTRAL FILE TO A CONSTANT
C      OR DO SOMETHING ELSE
      IF(NORM.EQ.0)GOTO 422
      FACTOR=10000/SQRT(SUME)
      GOTO 425
422      FACTOR=100000000/SUNE
425      CONTINUE
      CALL CBSET(CB1,CBSCR,FACTOR,CBAZR,DTARAY,IER)
      CALL VHR5(CB1) ;MULTIPLY BY SCALER
C***** LOAD NORMALIZED SPECTRUM INTO IDOB
      COUNTDOWN=COUNTDOWN-1
      COUNT=COUNT+1
      IDOB(JAOUT)=SQRT(SUME) ;LOAD ENERGY
      J3=1
      DO 430 J2=JAOUT+1,JAOUT+KLM ;LOADING ONLY HLM POINTS
      J3=J3+1
430      IDOB(J2)=DTARAY(J3)
440      FORMAT(5G12.4)

      IF(COUNTDOWN.LE.LASTCD)GOTO 1000
700      JAOUT=JAOUT+HLM
800      J5=J5+1
      GOTO260

1000     RETURN
        END

```

FILE: DSTA and DSTN  
LANGUAGE: FORTRAN 5  
DATE: September 21, 1982  
AUTHOR: D. Martin  
SUBJECT: Acoustic Analysis  
CALLING SEQUENCE: DSTA or DSTN  
DATE OF LAST REVISION: September 21, 1982

PURPOSE:

These programs compute distances between observations and phonets. DSTA fills the output disk file with all the distances between the specified observations and phonets. DSTN fills the output disk file with one unit for each specified observation. Each unit contains the observation number, observation energy, worst phonet match, maximum distance, and a selected number of ordered pairs: (phonet number, next best distance).

DESCRIPTION:

Location: DP4:BRATCHET

|       |         |             |
|-------|---------|-------------|
| Size: | DSTN.FR | 12912 bytes |
|       | DSTN.RB | 17402 bytes |
|       | DSTA.FR | 12699 bytes |
|       | DSTA.RB | 17488 bytes |

PROGRAM USE:

These programs are called by DRV R and call CHOOS, GFDB, CHKO, and CHKJ. They use the Array Processor to compute distances between observations and phonets from files calculated by DRSQ. The distinction between observation

le and phonet file is arbitrary in format; any file can run against itself. The distinction in the way these routines handle files is that distances to all phonets are computed for each observation.

The main difference between these routines is that the DSTA outputs every computed distance, DSTN selects the best matches and outputs the observation number, energy, and worst match as well. Tables 4 and 5 contain the output file formats.

Two distance rules are selectable: M1 and M2. M1 is Minkowski One distance:

$$D_{ij} = K \sum_{l=1}^N |X_{il} - Y_{jl}|$$

where:

$D_{ij}$  = distance between the  $i^{\text{th}}$  observation and the  $j^{\text{th}}$  phonet.

$K$  = constant scale factor.

$X_{il}$  =  $l^{\text{th}}$  component of the  $i^{\text{th}}$  observation.

$Y_{jl}$  =  $l^{\text{th}}$  component of the  $j^{\text{th}}$  phonet.

M2 is the Minkowski Two distance:

$$D_{ij} = K \sqrt{\sum_{l=1}^N (X_{il} - Y_{jl})^2}$$

where:

$D_{ij}$  = distance between the  $i^{\text{th}}$  observation and the  $j^{\text{th}}$  phonet.

$K$  = constant scale factor.

$X_{il}$  =  $l^{\text{th}}$  component of the  $i^{\text{th}}$  observation.

| ELEMENT NUMBER | ASSIGNMENT    |
|----------------|---------------|
| 1              | $D(i, j)$     |
| 2              | $D(i, j+1)$   |
| 3              | $D(i, j+2)$   |
| .              | .             |
| .              | .             |
| .              | .             |
| $K-j+1$        | $D(i, K)$     |
| $K-j+2$        | $D(i+1, j)$   |
| $K-j+3$        | $D(i+1, j+1)$ |
| .              | .             |
| .              | .             |
| .              | .             |
| $2K-j+1$       | $D(i+1, K)$   |
| .              | .             |
| .              | .             |
| .              | .             |
| $(1+1)K-j+1$   | $D(i+1, K)$   |

where:  $i$  = starting observation number specified.

$j$  = starting phonet number specified.

$i+1$  = last observation number specified.

$K$  = last phonet specified.

TABLE 4. DSTA Output File

| ELEMENT NUMBER | ASSIGNMENT                                                       |
|----------------|------------------------------------------------------------------|
| 1              | Number of first observation specified.                           |
| 2              | Energy of first observation specified.                           |
| 3              | Number of worst phonet choice.                                   |
| 4              | Largest distance.                                                |
| 5              | Number of best phonet choice.                                    |
| 6              | Minimum distance.                                                |
| 7              | Number of next best phonet choice.                               |
| 8              | Next minimum distance.                                           |
| .              | .                                                                |
| .              | .                                                                |
| .              | .                                                                |
| 4+2N           | N <sup>th</sup> minimum distance.                                |
| 4+2N+1         | Number of next observation specified.                            |
| .              | .                                                                |
| .              | .                                                                |
| .              | .                                                                |
| K(4+2N)        | N <sup>th</sup> minimum distance to K <sup>th</sup> observation. |

TABLE 5. DSTN Output File

$y_{jl} = 1^{\text{th}}$  component of the  $j^{\text{th}}$  phonet

The observation energy is not included in either distance computation. Also, the distance is checked for overflow before being truncated to an integer and written to disk. If the distance is greater than 32767, it is hard limited to that value. The scale factor, K in the computations, is adjustable to obtain a satisfactory range of distance measures. The flowchart in Figure 74 applies to both DSTA and DSTN. Distance file header assignments are in Table 6.

LIST OF VARIABLES:

| <u>VARIABLE</u> | <u>TYPE</u>   | <u>PURPOSE</u>                           |
|-----------------|---------------|------------------------------------------|
| XMEM            | Integer       | Number of extended memory blocks to use. |
| OUTD            | Integer       | Output option switch.                    |
| FILE2           | Integer Array | Holds observation file name.             |
| FILE1           | Integer Array | Holds phonet file name.                  |
| FILE3           | Integer Array | Holds output distance file name.         |
| STS1            | Integer       | First observation to do.                 |
| STSL            | Integer       | Last observation to do.                  |
| PTS1            | Integer       | First phonet to do.                      |
| PTSL            | Integer       | Last phonet to do.                       |
| HEADER          | Integer Array | Holds header.                            |
| NSTSTB          | Integer       | Number of observations to do.            |
| NPSTS           | Integer       | Number of words in observation.          |

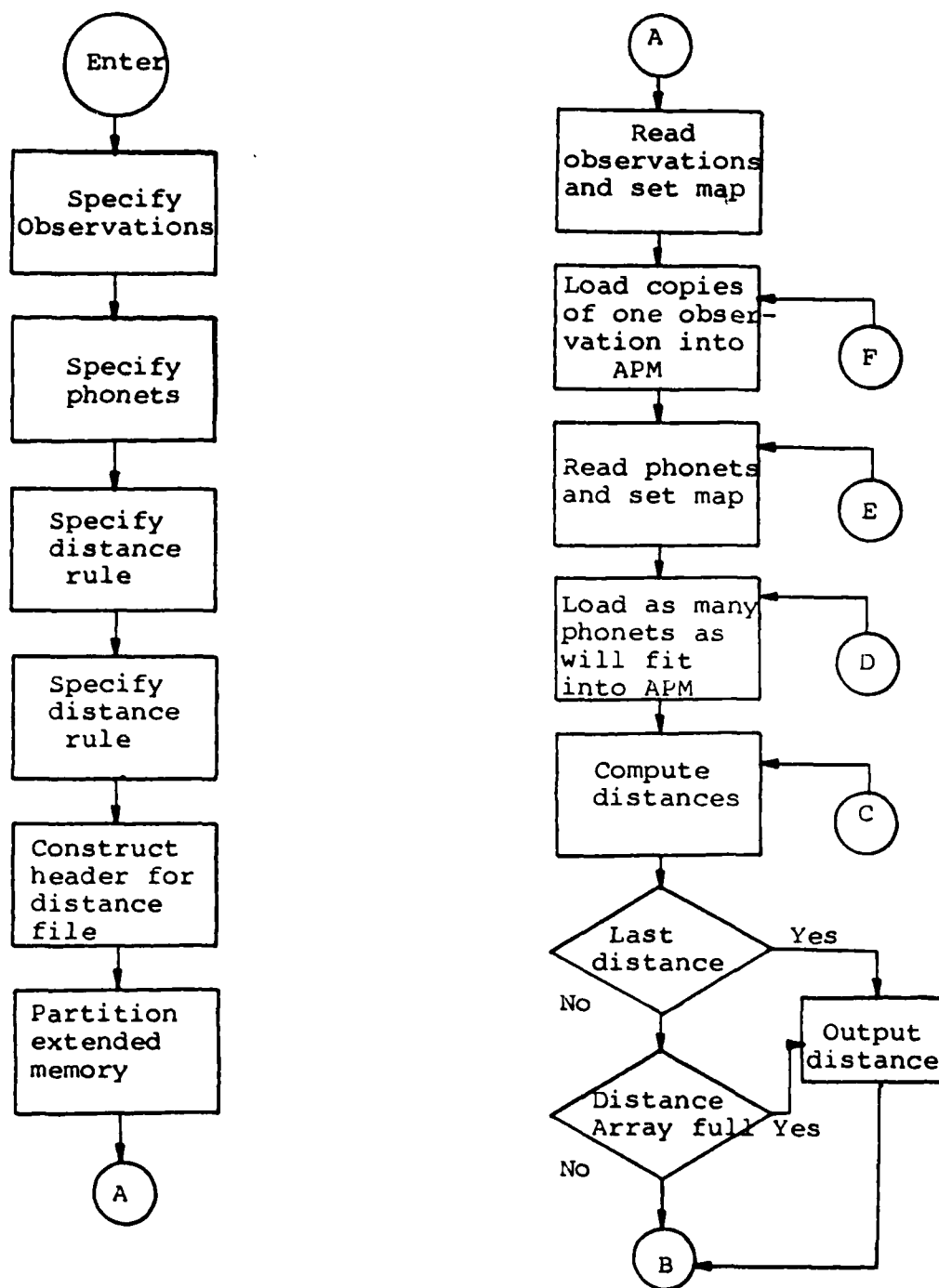


FIGURE 74. Flowchart of DSTA and DSTN

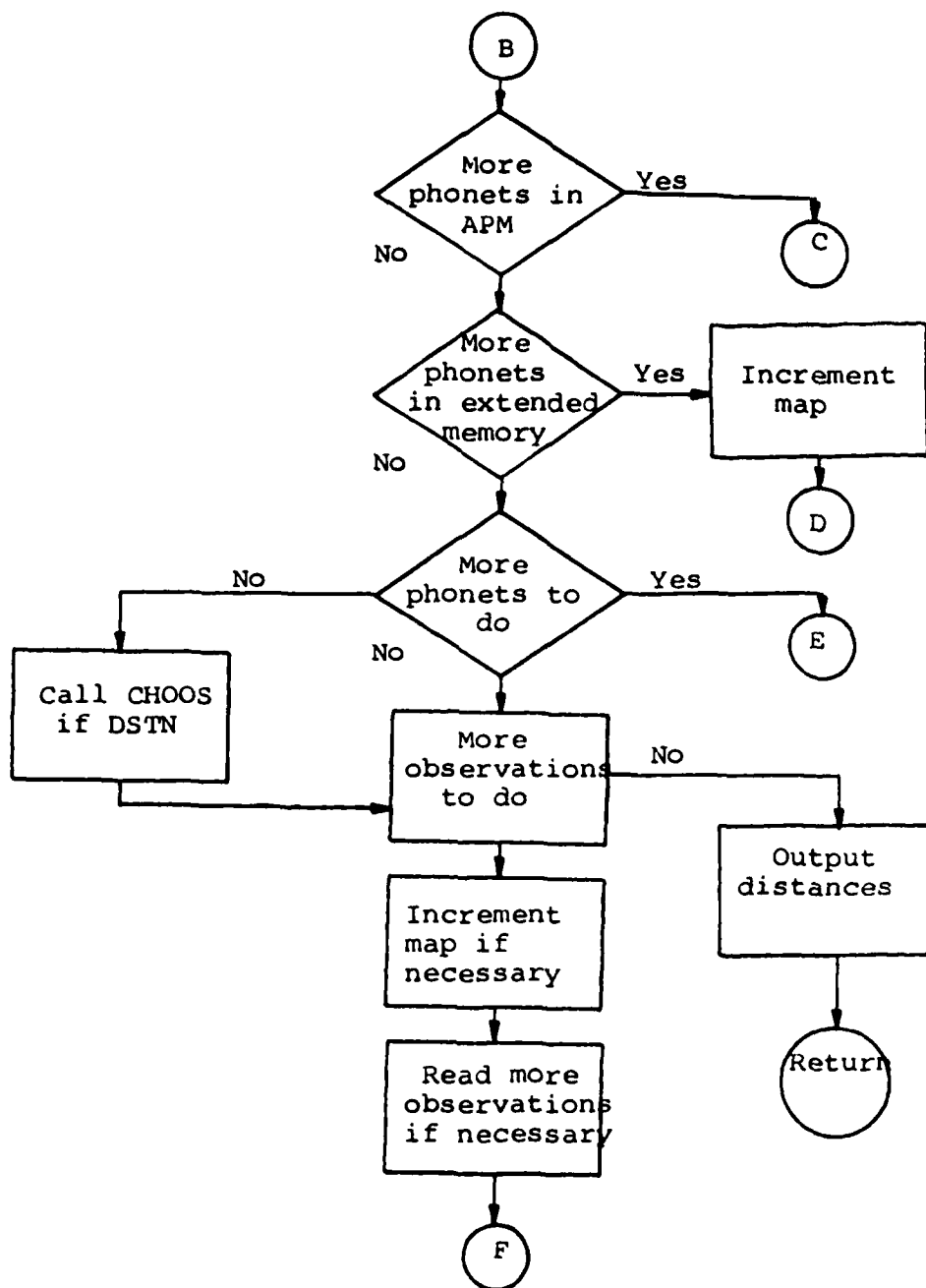


FIGURE 74 CONTINUED. Flowchart of DSTA and DSTN

| ELEMENT NUMBER | CONTENT                                                           |
|----------------|-------------------------------------------------------------------|
| 1-13           | Distance file name.                                               |
| 14-26          | Observation file name.                                            |
| 27-39          | Phonet file name.                                                 |
| 40             | Number of first observation time slice to do.                     |
| 41             | Number of last observation time slice to do.                      |
| 42             | Number of first phonet time slice to do.                          |
| 43             | Number of last phonet time slice to do.                           |
| 44             | Number of disk block that holds first observation to do.          |
| 45             | Number of disk block that holds first phonet to do.               |
| 46             | Switch: 4 = observation and phonet files identical/0 = different. |
| 47             | NCHOICES.                                                         |
| 48             | Number of observation time slices to do.                          |
| 49             | Number of phonet time slices to do.                               |
| 50             | Number of elements per time slice.                                |
| 51             | Number of extended memory block used less those in window.        |
| 52-256         | Unused.                                                           |

TABLE 6. Distance File Header

| <u>VARIABLE</u> | <u>TYPE</u> | <u>PURPOSE</u>                                            |
|-----------------|-------------|-----------------------------------------------------------|
| IBO             | Integer     | Holds the first observation number.                       |
| FSPFSQ          | Integer     | Switch which selects phonet file.                         |
| NPTSTB          | Integer     | Number of phonets to do.                                  |
| NLDTB           | Integer     | Number of distances to compute.                           |
| JCTLD           | Integer     | Counts distances computed.                                |
| IBP             | Integer     | Holds the first phonet number.                            |
| FDSTR           | Integer     | Switch which selects distance rule.                       |
| NPFS            | Integer     | Number of required memory blocks for observations.        |
| NPFP            | Integer     | Number of required memory blocks for phonets.             |
| NBFPH           | Integer     | Number of extended memory blocks to use for phonets.      |
| NBFOBS          | Integer     | Number of extended memory blocks to use for observations. |
| NBFDIST         | Integer     | Number of extended memory blocks to use for distances.    |
| OBSMAPCNT       | Integer     | First extended memory block used for observations.        |
| FONMAPCNT       | Integer     | First extended memory block used for phonets.             |
| DISTMAPCNT      | Integer     | Extended memory block used for distances.                 |
| FONBANGCNT      | Integer     | Counts phonets.                                           |
| OBSLEFT         | Integer     | Specifies observation points left.                        |

| <u>VARIABLE</u> | <u>TYPE</u>   | <u>PURPOSE</u>                             |
|-----------------|---------------|--------------------------------------------|
| FONLEFT         | Integer       | Specifies phonet points left.              |
| NPPTS           | Integer       | Number of words in phonet.                 |
| WKOBS           | Real Array    | Array Processor memory array.              |
| WKFON           | Real Array    | Array Processor memory array.              |
| DIST            | Integer Array | Output distance file.                      |
| JDISTNBR        | Integer       | Counts phonets in WKFON.                   |
| JOUTDIST        | Integer       | Counts distances in array DIST.            |
| NDBFDIST        | Integer       | Holds disk block to write distances to.    |
| NQBFOBS         | Integer       | Number of quarter blocks for observations. |
| NQBFPH          | Integer       | Number of quarter blocks for phonets.      |
| CNTOBS          | Integer       | First disk block of observations to get.   |
| IFBC            | Integer       | First data point in disk block.            |
| ICNTOBS         | Integer       | Counts observation disk blocks read.       |
| CNTFON          | Integer       | First disk block of phonets to get.        |
| IFBD            | Integer       | First data point in disk block.            |
| ICNTFON         | Integer       | Counts phonet disk blocks read.            |
| NFONLD          | Integer       | Number of phonets in next map.             |
| NPLTB           | Integer       | Number of phonets left to do.              |

| <u>VARIABLE</u> | <u>TYPE</u> | <u>PURPOSE</u>                                      |
|-----------------|-------------|-----------------------------------------------------|
| FONLEFT         | Integer     | Number of points in WKFON to fill.                  |
| JOUTBLKS        | Integer     | Number of distance quarter blocks to write to disk. |
| JOUTDIST        | Integer     | Counts distances.                                   |
| FON             | Real Array  | Mapped through extended memory to get phonets.      |
| OBS             | Real Array  | Mapped through extended memory to get observations. |

SWITCH SETTINGS:

| <u>SWITCH</u> | <u>SETTINGS</u>                                                                               |
|---------------|-----------------------------------------------------------------------------------------------|
| OUTD          | = 0 for disk output only<br>= 10 for screen output as well<br>= 12 for printed output as well |
| FSPFSQ        | = 4 to run the observation file against itself<br>= 3 to use separate phonet file             |
| FDSTR         | = 1 for M1 distance rule<br>= 2 for M2 distance rule                                          |

RELATED PROGRAMS:

PLTA, PLTO, FLTS, PLTT, PLTN, DRVR, S64, S128, CHOOS,  
FCTR, GFDB, CHKO, CHKJ, GRPH2.

```

C***** ROUTINE DSTN  NOTE:  THIS ROUTINE IS FOR FORTRAN 5 ! !
C      THIS SUBROUTINE CALLS ARRAY PROCESSOR ROUTINES AND USES EXTENDED
C      MEMORY (14 BLOCKS OR MORE) TO COMPUTE A SPECIFIED NUMBER OF BEST
C      DISTANCES, BEST BEING SMALLEST, BETWEEN SPECIFIED OBSERVATIONS
C      AND SPECIFIED PHONETS.
C      BY:  CAPT DAN MARTIN
C      DATE: 9/21/82
C      SUBJ: ACOUSTIC ANALYSIS
C      THIS ROUTINE IS CALLED BY DRVH AND CALLS ROUTINES CHOOS, CHKO, CHKJ,
C      AND GFDB.  ONE OF TWO DISTANCE RULES, M1 OR M2, CAN BE SELECTED.
C      INPUT FILES ARE SPECTRAL FILES OUTPUT BY DRSQ.  THE OUTPUT FILE IS
C      AN INTEGER FILE OF CONSECUTIVE UNITS, EACH UNIT BEING 4+2N POINTS
C      LONG, N BEING THE NUMBER OF BEST PHONET CHOICES SPECIFIED.  IN EACH
C      UNIT, THE FIRST ELEMENT IS THE OBSERVATION NUMBER, THE SECOND
C      ELEMENT IS THE OBSERVATION ENERGY, THE THIRD AND FOURTH ARE AN
C      ORDERED PAIR: (PHONET NUMBER, MAX DISTANCE), AND THE REMAINING
C      POINTS ARE N-ORDERED PAIRS, ORDERED BEST-TO-WORST: (PHONET NUMBER,
C      DISTANCE).
C      THIS ROUTINE ACCOMPLISHES THE FOLLOWING TASKS:
C          1) GET OBSERVATION FILE
C          2) SPECIFY OBSERVATIONS-FIRST AND LAST-TO DO
C          3) GET PHONET FILE
C          4) SPECIFY PHONETS-FIRST AND LAST-TO DO
C          5) GET DISTANCE RULE
C          6) GET FILE NAME FOR OUTPUT DISTANCES
C          7) CONSTRUCT OUTPUT HEADER
C          8) PARTITION EXTENDED MEMORY BETWEEN DISTANCES,
C             OBSERVATIONS, AND PHONETS
C          9) READ OBSERVATIONS INTO MEMORY
C          10) GET ONE OBSERVATION
C          11) READ PHONETS INTO MEMORY
C          12) GET PHONETS TO DO
C          13) GET ALL DISTANCES FOR CURRENT PHONET
C          14) CALL CHOOS TO FILL OUTPUT FILE
C          15) GET NEXT OBSERVATION
C          16) GO TO STEP 11 UNTIL ALL OBSERVATIONS ARE DONE
C          17) OUTPUT DISTANCES ACCUMULATED
C      FOR MORE INFORMATION, SEE THE USERS MANUAL OR MY THESIS.

```

```

SUBROUTINE DSTN
OVERLAY ODSN
INCLUDE"ARRAY:FSAPS.FR"
PARAMETER NDP=1024
PARAMETER NDPN1=1023
PARAMETER QTRBLK=256
REAL WKFON,WKOBS,NPFSSR,NPFPR,FACTOR,FON,OBS,HOLDIST
REAL MAXDIST,MINDIST
INTEGER FSPFN,FILE2,HEADER,STS1,STSL,NSTSTB,NPSTS,NPFS
INTEGER DIST,FSPFSQ,FILE1,PTS1,PTSL,JOUTDIST
INTEGER NPTSTB,NPPTS,NPFP,FUSTR,FILE3,OUTD,FLAG,CW1,SKIP
INTEGER XMEN,NBFDIST,NBFOBS,NBFPH,CNTOBS,CNTFON,OBSMAPCNT
INTEGER FONMAPCNT,FONBANGCNT,OBSLEFT,NQBFOBS,NQBFPH,NQBFDIST
INTEGER DISTMAPCNT,OBSBANGCNT,FONLEFT,ICNTOBS,IFBC
INTEGER NPLTB,JFONMAPCNT,JOBSMAPCNT,DISTMAPCNTMAX,NDBFDIST

```

INTEGER JDISTNBR,JCNTOBS,JCNTFON,ICNTFON,NFONLD,INITMEM

COMMON / APM / WKOBS(NDP),WKFON(NDP)  
COMMON / VALS / OBS(NDP),FON(NDP),DIST(NDP),OBSLEFT,FSPFN  
COMMON / VALS / INITMEM,FILE3(13),HEADER(QTRBLK),NPFSR,NPFP  
COMMON / VALS / PTS1,PTSL,NPTSTB,NPPTS,NPFP,FDSTR,OUTD,NBFDIST  
COMMON / VALS / NBFOBS,NBFPFH,CNTOBS,CNTFON,OBSMAPCNT,FONMAPCNT  
COMMON / VALS / FONBANGCNT,NQBFOBS,NQBFPFH,NQBFDIST,DISTMAPCNT  
COMMON / VALS / ICNTOBS,ICNTFON,IFBC,JFONMAPCNT,NFONLD  
COMMON / VALS / OBSBANGCNT,FONLEFT,NPLTB,JOBSMAPCNT,NBFDIST  
COMMON / VALS / DISTMAPCNTMAX,JOUIDIST,JDISTNBR,JCNTOBS,JCNTFON  
COMMON / VALT / SKIP,STSL,NSTSTB,NPSTS,NPFS,XMEM,FACTOR,FSPFSQ  
COMMON / VALT / FILE1(13),FILE2(13),FLAG,STS1,ISXMEM  
COMMON / VALV / CB1(0:CBMAX)  
COMMON / VALU / HOLDIST(NDP),MAXDIST,MINDIST,MAXDISTLOC,MINDISTLOC  
COMMON / VALU / NCHOICES,NBRPHONES

TYPE"\*\*\*\*YOU ARE NOW IN ROUTINE DSTN.FR\*\*\*\*"

50 TYPE"\*YOU HAVE",ISXMEM," 1K BLOCKS OF EXT MEMORY ACCESSABLE"  
XMEM=ISXMEM  
3 TYPE"\*ENTER OUTPUT DEVICE \*"  
ACCEPT"(0=NONE/10=TERMINAL/12=PRINTER): ",OUTD  
IF(OUTD.EQ.0.OR.OUTD.EQ.10.OR.OUTD.EQ.12)GOTO 5  
TYPE"ERROR: INVALID VALUE FOR OUTD: ",OUTD  
GOTO 3

C\*\*\*\*\* GET OBSERVATION FILE NAME

5 ACCEPT"\*ENTER NAME OF FILE THAT HOLDS OBSERVATION TIME-SLICES: "  
READ(11,10)FILE2(1)  
10 FORMAT(S13)  
12 CALL OPEN(4,FILE2,2,IER)  
IF(IER.NE.1)TYPE"ERROR ON OPEN OF SPECTRAL FILE, IER= ",IER  
GOTO 18  
15 CONTINUE

C\*\*\*\*\* GET NUMBERS OF 1ST & LAST TIME-SLICES OF SPECTRUM

18 CALL RDBLK(4,0,HEADER,1,IER)  
IF(OUTD.EQ.0)GOTO 19  
WRITE(OUTD,9001)(HEADER(15),15=1,13)  
19 TYPE"1ST SPECTRAL TIME-SLICE \* IS:",HEADER(55)  
TYPE"LAST SPECTRAL TIME-SLICE \* IS:",HEADER(56)  
20 ACCEPT"\*ENTER 1ST SPECTRAL TIME-SLICE \* TO BANG: ",STS1  
IF(STS1.GE.HEADER(55))GOTO 22  
TYPE"ERROR: 1ST TIME-SLICE TOO SMALL! MUST BE GREATER THAN OR"  
TYPE"EQUAL TO",HEADER(55)  
GOTO 20  
22 ACCEPT"\*ENTER LAST SPECTRAL TIME-SLICE \* TO BANG: ",STSL  
IF(STSL.LE.HEADER(56))GOTO 23  
TYPE"ERROR: LAST TIME-SLICE TOO LARGE! MUST BE LESS THAN OR"  
TYPE"EQUAL TO",HEADER(56)  
GOTO 22  
23 IF(STS1.LT.STSL)GOTO 24  
TYPE"ERROR: 1ST TIME-SLICE LARGER THAN LAST!"

```

      GOTO 20
C***** GET # OF TS'S TO BANG AND AMOUNT OF MEMORY REQUIRED
24  NSTSTB=STSL-STSL+1      ;# TS'S TO BANG
    NPSTS=HEADER(57)*2      ;# WORDS PER TS
    IBO=HEADER(55)
C***** GET PHONET FILE
    TYPE"*DO YOU WANT TO BANG THE SPECTRAL FILE AGAINST ITSELF?"
    ACCEPT"ENTER (4=YES/3=NO): ",FSPFSQ
    IF(FSPFSQ.EQ.4)GOTO 37
35  ACCEPT"***ENTER PHONET FILENAME: "
    READ(11,10)FILE1(1)
36  CALL OPEN(3,FILE1,2,IER)
    CALL CHECK(IER)
    CALL RDBLK(3,0,HEADER,1,IER)
    CALL CHECK(IER)
    IF(OUTD.EQ.0)GOTO 39
    WRITE(OUTD,9001)(HEADER(I5),I5=1,13)
    GOTO 39
C***** GET THE 1ST & LAST PHONETS
37  DO 38 J1=1,13
38  FILE1(J1)=FILE2(J1)
39  TYPE"1ST PHONET TS* IS:",HEADER(55)
    TYPE"LAST PHONET TS* IS: ",HEADER(56)
41  ACCEPT"***ENTER 1ST PHONET TS TO DO: ",PTS1
    IF(PTS1.GE.HEADER(55))GOTO 42
    TYPE"ERROR: 1ST TIME-SLICE IS TOO SMALL! MUST BE GREATER"
    TYPE"THAN OR EQUAL TO",HEADER(55)
    GOTO 41
42  ACCEPT"***ENTER LAST PHONET TS TO DO: ",PTSL
    IF(PTSL.LE.HEADER(56))GOTO 43
    TYPE"ERROR: LAST TIME-SLICE TOO BIG! MUST BE SMALLER THAN"
    TYPE"OR EQUAL TO",HEADER(56)
    GOTO 42
43  IF(PTS1.LE.PTSL)GOTO 44
    TYPE"ERROR: 1ST TIME-SLICE LARGER THAN LAST!"
    GOTO 41
44  ACCEPT" ENTER NUMBER OF DISTANCE CHOICES TO GET: ",NCHOICES
C***** GET NUMBER OF PHONET TS'S TO BANG AND AMOUNT OF MEMORY
C    THAT WILL TAKE
    NPTSTB=PTSL-PTS1+1      ;# TS'S TO BANG
    NLDTD=NSTSTB*(4+2*NCHOICES) ;# DISTANCES TO GET
    IBP=HEADER(55)
C***** VERIFY # PNTS IN PHONET TIME-SLICES MATCH # PNTS IN SPECTRAL TS'S
    NPPTS=HEADER(57)*2
    IF(NPPTS.EQ.NPSTS)GOTO 60
    TYPE"ERROR: # PNTS IN PHONETIC TIME-SLICE MUST MATCH # PNTS"
    TYPE"IN SPECTRAL TIME-SLICE. AS SPECIFIED,"
    TYPE"* PNTS IN PTS=",NPPTS
    TYPE"* PNTS IN STS=",NPSTS
    GOTO 35
60  CONTINUE
C***** GET DISTANCE RULE
    ACCEPT"*SELECT DISTANCE RULE (1=M1/2=M2): ",FDSTR
C***** GET DISTANCE FILE

```

```

C***** GET NAME OF FILE TO STASH DISTANCES INTO
65 ACCEPT"***ENTER NAME OF FILE TO RECIEVE DISTANCES: "
   READ(11,10)FILE3(1)
70 CALL OPEN(2,FILE3,2,IER)
   IF(IER.NE.1)TYPE"ERROR ON OPEN OF DISTANCE FILE, IER= ",IER
   IF(IER.NE.13)GOTO 75
   TYPE"FILE DOES NOT EXIST. WILL CREATE IT FOR YOU."
   CALL CFILW(FILE3,2,IER1)
   IF(IER1.NE.1)TYPE"ERROR ON FILE CREATION. IER1= ",IER1
   GOTO 70
75 CONTINUE
C***** STORE FILENAMES IN HEADER OF DISTANCE FILE
   DO 100 J1=1,QTRBLK
100  HEADER(J1)=0.0
   DO 110 J1=1,13
   HEADER(J1)=FILE3(J1)
   J2=J1+13
   HEADER (J2)=FILE2(J1)
   J3=J1+26
110  HEADER(J3)=FILE1(J1)
   HEADER(40)=STS1
   HEADER(41)=STSL
   HEADER(42)=PTS1
   HEADER(43)=PTSL
   HEADER(46)=FSPFSQ ;SWITCH INDICATES IF OBS AND PHONS ARE SAME
   HEADER(47)=NCHOICES ;# DISTANCE CHOICES
   HEADER(48)=NSTSTB ;# OBS TIME SLICES
   HEADER(49)=NPTSTB ;# PHON TIME SLICES
   HEADER(50)=NPPTS ;# ELEMENTS PER TIME-SLICE
C***** DIVVY-UP EXTENDED MEMORY BETWEEN OBSERVATION SPECTRUM,
C PHONET SPECTRUM, AND THE DISTANCE FILES.
111 NPFS=2+2*IFIX(FLOAT(NSTSTB)*NPPTS/(2*NDP)) ; MEM BLKS FOR OBS FILE
   NPFP=2+2*IFIX(FLOAT(NPTSTB)*NPPTS/(2*NDP)) ; MEM BLKS FOR PHON FILE
114 CONTINUE
116 J1=IFIX((XMEM-3)/2)*2 ;NEED EVEN # MEM BLKS
   IF(NPFP.LT.J1)NBFPH=NPFP
   IF(NPFP.GE.J1)NBFPH=J1
   J1=IFIX((XMEM-1-NBFPH)/2)*2
   IF(NPFS.LT.J1)NBFOBS=NPFS
   IF(NPFS.GE.J1)NBFOBS=J1
   MBFDIST=1
   IF(NBFOBS.GT.0)GOTO 118
   TYPE"ERROR: INSUFFICIENT EXTENDED MEMORY AVAILABLE."
   TYPE"NEED AT LEAST 14 1K BLOCKS."
   CALL VMEM(XMEM,IER)
   TYPE"YOU HAVE",XMEM," 1K BLOCKS OF EXTENDED MEMORY AVAILABLE."
   ACCEPT"HOW MANY DO YOU WANT? ",XMEM
   XMEM=XMEM-9
   GOTO 114
118 CONTINUE
   HEADER(51)=XMEM
C***** INITIALIZE TO 1ST AVAILABLE EXT MEM BLK
   OBSMAPCNT=9 ;1ST EXT MEM BLK USED FOR OBSERVATIONS
   FONMAPCNT=OBSMAPCNT+NBFOBS ;1ST EXT MEM BLK

```

```

***** ROUTINE DSTA  NOTE:  THIS ROUTINE IS FOR FORTRAN 5 ! !
THIS SUBROUTINE CALLS ARRAY PROCESSOR ROUTINES AND USES EXTENDED
MEMORY (14 MEMORY BLOCKS OR MORE) TO COMPUTE ALL DISTANCES
BETWEEN EACH OBSERVATION SPECIFIED AND EACH PHONET SPECIFIED.
BY:  CAPT DAN MARTIN
DATE:  9/21/82
SUBJ:  ACOUSTIC ANALYSIS
THIS ROUTINE IS CALLED BY DRVH AND CALLS CHXO AND GFDB.  INPUT
FILES ARE SPECTRAL FILES OUTPUT BY DRSQ.  THE OUTPUT FILE IS AN
INTEGER FILE OF CONSECUTIVE UNITS.  EACH UNIT CONTAINS DISTANCES
BETWEEN AN OBSERVATION AND EACH PHONET, THE K-TH ELEMENT OF THE
J-TH UNIT IS THE DISTANCE BETWEEN THE J-TH OBSERVATION AND THE
K-TH PHONET.
THIS ROUTINE ACCOMPLISHES THE FOLLOWING TASKS:
1)  GET OBSERVATION FILE
2)  SPECIFY OBSERVATIONS-FIRST AND LAST-TO DO
3)  GET PHONET FILE
4)  SPECIFY PHONETS-FIRST AND LAST-TO DO
5)  GET DISTANCE RULE
6)  GET FILE NAME FOR OUTPUT DISTANCES
7)  CONSTRUCT OUTPUT HEADER
8)  PARTITION EXTENDED MEMORY BETWEEN DISTANCES,
OBSERVATIONS, AND PHONETS
9)  READ OBSERVATIONS INTO MEMORY
10) GET ONE OBSERVATION
11) READ PHONETS INTO MEMORY
12) GET PHONETS TO DO
13) GET A DISTANCE
14) OUTPUT DISTANCES WHEN 1024 HAVE BEEN ACCUMULATED
15) GET MORE PHONETS UNTIL ALL ARE DONE
16) GET NEXT OBSERVATION
17) GO TO STEP 11 UNTIL ALL OBSERVATIONS ARE DONE
18) OUTPUT DISTANCES ACCUMULATED
FOR MORE INFORMATION, SEE THE USERS MANUAL OR MY THESIS.

```

```

SUBROUTINE DSTA
OVERLAY ODSA
INCLUDE"AKRAYP:F5APS.FR"
PARAMETER NDP=1024
PARAMETER NDPM1=1023
PARAMETER UTRBLK=256
REAL WKFON,WKOB,MLDID,JCTLD,NPFSR,NPFFR,FACTOR,FON,OBS
INTEGER FSPFN,FILE2,HEADER,STS1,STSL,NSTSTB,NPSTS,NPFS
INTEGER DIST,FSPFSO,FILE1,PTS1,PTSL,JOUTDIST
INTEGER NPTSTB,NPPTS,NPFP,FDSTR,FILE3,OUTD,FLAG,CB1,SKIP
INTEGER XMEM,NBFDIST,NBFOBS,NBFFPH,CNTOBS,CNTFON,OBSMAPCNT
INTEGER FONMAPCNT,FONBANGCNT,OBSLEFT,NQBFOBS,NQBFFPH,NQBFDIST
INTEGER DISTMAPCNT,OBSBANGCNT,FONLEFT,ICNTOBS,IFBC,GSPCT
INTEGER NPLTB,JFONMAPCNT,JOBSMAPCNT,DISTMAPCNTMAX,NBFDIST
INTEGER JDISTNBR,JCNTOBS,JCNTFON,ICNTFON,NFONLD,INITMEM

```

```

COMMON / APM / WKOB(NDP),WKFON(NDP)
COMMON / VALS / OBS(NDP),FON(NDP),DIST(NDP),OBSLEFT,FSPFN

```

AD-A125 360

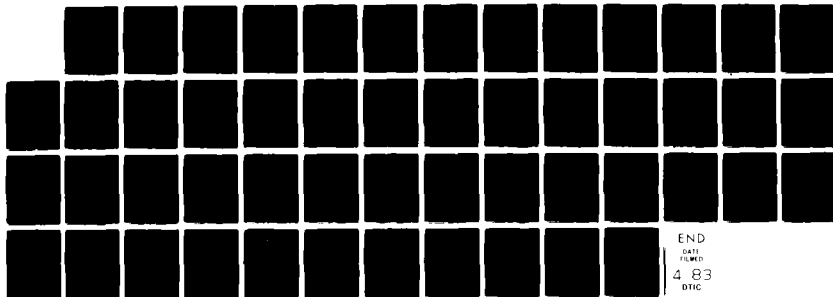
COMPUTER RECOGNITION OF PHONETS IN SPEECH(U) AIR FORCE  
INST OF TECH WRIGHT-PATTERSON AFB OH SCHOOL OF  
ENGINEERING D L MARTIN DEC 82 AFIT/GE/EE/82D-46

3/3

UNCLASSIFIED

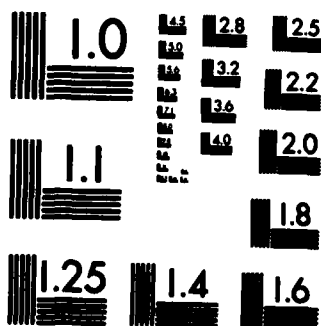
F/G 9/2

NL



END  
DATE  
FILMED  
4 83  
DTIC

M-2



MICROCOPY RESOLUTION TEST CHART  
NATIONAL BUREAU OF STANDARDS-1963-A

```

COMMON / VALS / INITMEN,FILE3(13),HEADER(QTRBLK),NPFSR,NPFPF
COMMON / VALS / PTS1,PTSL,NPTSTB,NPPTS,NPFP,PDSTR,OUTD,NBFDIST
COMMON / VALS / NBFOBS,NBFPF,CNTOBS,CNTPON,OBSDAPCNT,FONMAPCNT
COMMON / VALS / FONDANGCNT,NBFOBS,NBFPF,NBFDIST,DISTAPCNT
COMMON / VALS / ICNTOBS,ICNTPON,IFBC,JFONMAPCNT,NFONLD
COMMON / VALS / ONSDANGCNT,FONLEFT,NPLTB,JOBSMAPCNT,NBFDIST
COMMON / VALS / DISTAPCNTMAX,JOUTDIST,JDISTNBR,JCNTOBS,JCNTPON
COMMON / VALT / SKIP,STSL,NSTSTB,NPSTS,NPFS,XMEN,FACTOR,PSPPSQ
COMMON / VALT / FILE1(13),FILE2(13),FLAG,STS1,ISXMEN
COMMON / VALV / CB1(0:CBMAX)

```

TYPE"\*\*\*\*YOU ARE NOW IN ROUTINE DSTA.FR\*\*\*\*"

50 TYPE"YOU HAVE",ISXMEN," 1K BLOCKS OF EXT MEMORY ACCESSABLE"  
XMEN=ISXMEN

3 TYPE"ENTER OUTPUT DEVICE "  
ACCEPT"(0=NONE/10=TERMINAL/12=PRINTER): ",OUTD  
IF(OUTD.EQ.0.OR.OUTD.EQ.10.OR.OUTD.EQ.12)GOTO 5  
TYPE"ERROR: INVALID VALUE FOR OUTD: ",OUTD  
GOTO 3

C\*\*\*\*\* GET OBSERVATION FILE NAME

5 ACCEPT"ENTER NAME OF FILE THAT HOLDS OBSERVATION TIME-SLICES: "  
READ(11,10)FILE2(1)  
10 FORMAT(S13)  
12 CALL OPEN(4,FILE2,2,IER)  
IF(IER.NE.1)TYPE"ERROR ON OPEN OF SPECTRAL FILE, IER= ",IER  
CALL CHKO(IER) ;IS UNIT # IN USE?  
GOTO 18  
15 CONTINUE

C\*\*\*\*\* GET NUMBERS OF 1ST & LAST TIME-SLICES OF SPECTRUM

18 CALL RDBLK(4,0,HEADER,1,IER)  
CALL CHECK(IER)  
IF(OUTD.NE.0)WRITE(OUTD,9001)(HEADER(I5),I5=1,13)  
19 TYPE"1ST SPECTRAL TIME-SLICE # IS:",HEADER(55)  
TYPE"LAST SPECTRAL TIME-SLICE # IS:",HEADER(56)  
20 ACCEPT"ENTER 1ST SPECTRAL TIME-SLICE # TO BANG: ",STS1  
IF(STS1.GE.HEADER(55))GOTO 22  
TYPE"ERROR: 1ST TIME-SLICE TOO SMALL! MUST BE GREATER THAN OR"  
TYPE"EQUAL TO",HEADER(55)  
GOTO 20  
22 ACCEPT"ENTER LAST SPECTRAL TIME-SLICE # TO BANG: ",STSL  
IF(STSL.LE.HEADER(56))GOTO 23  
TYPE"ERROR: LAST TIME-SLICE TOO LARGE! MUST BE LESS THAN OR"  
TYPE"EQUAL TO",HEADER(56)  
GOTO 22  
23 IF(STS1.LE.STSL)GOTO 24  
TYPE"ERROR: 1ST TIME-SLICE LARGER THAN LAST!"  
GOTO 20

C\*\*\*\*\* GET # OF TS'S TO BANG AND AMOUNT OF MEMORY REQUIRED

24 NSTSTB=STSL-STSL+1 ;# TS'S TO BANG  
NPSTS=2\*HEADER(57) ;# WORDS PER TS

```

      IBO=HEADER(55)
C***** GET PHONET FILE
      TYPE="DO YOU WANT TO BANG THE SPECTRAL FILE AGAINST ITSELF?"
      ACCEPT"ENTER (4=YES/3=NO): ",FSPFSQ
      IF(FSPFSQ.EQ.4)GOTO 37
35      ACCEPT"***ENTER PHONET FILENAME: "
      READ(11,10)FILE1(1)
36      CALL OPEN(3,FILE1,2,IER)
      CALL CHECK(IER)
      CALL RDBLK(3,0,HEADER,1,IER)
      CALL CHECK(IER)
      IF(OUTD.NE.0)WRITE(OUTD,9001)(HEADER(15),15=1,13)
      GOTO 39
C***** GET THE 1ST & LAST PHONETS
37      DO 38 J1=1,13
38      FILE1(J1)=FILE2(J1)
39      TYPE"1ST PHONET TS= IS:",HEADER(55)
      TYPE"LAST PHONET TS= IS: ",HEADER(56)
41      ACCEPT"***ENTER 1ST PHONET TS TO DO: ",PTS1
      IF(PTS1.GE.HEADER(55))GOTO 42
      TYPE"ERROR: 1ST TIME-SLICE IS TOO SMALL! MUST BE GREATER"
      TYPE"THAN OR EQUAL TO",HEADER(55)
      GOTO 41
42      ACCEPT"***ENTER LAST PHONET TS TO DO: ",PTSL
      IF(PTSL.LE.HEADER(56))GOTO 43
      TYPE"ERROR: LAST TIME-SLICE TOO BIG! MUST BE SMALLER THAN"
      TYPE"OR EQUAL TO",HEADER(56)
      GOTO 42
43      IF(PTS1.LE.PTSL)GOTO 44
      TYPE"ERROR: 1ST TIME-SLICE LARGER THAN LAST!"
      GOTO 41
C***** GET NUMBER OF PHONET TS'S TO BANG AND AMOUNT OF MEMORY
      C
      THAT WILL TAKE
44      NPTSTB=PTSL-PTS1+1          ;# TS'S TO BANG
      NLDTD=NSTSTB-NPTSTB          ;# DISTANCE? TO COMPUTE
      JCTLD=0                      ;SET COUNTER FOR # DISTANCES
      IBP=HEADER(55)
C***** VERIFY # PNTS IN PHONET TIME-SLICES MATCH # PNTS IN SPECTRAL TS'S
      NPPTS=2*HEADER(57)
      IF(NPPTS.EQ.NPSTS)GOTO 60
      TYPE"ERROR: # PNTS IN PHONETIC TIME-SLICE MUST MATCH # PNTS"
      TYPE"IN SPECTRAL TIME-SLICE. AS SPECIFIED,"
      TYPE"# PNTS IN PTS=",NPPTS
      TYPE"# PNTS IN STS=",NPSTS
      GOTO 35
60      CONTINUE
C***** GET DISTANCE RULE
      ACCEPT"SELECT DISTANCE RULE (1=M1/2=M2): ",PDSTR
C***** GET DISTANCE FILE
C***** GET NAME OF FILE TO STASH DISTANCES INTO
65      ACCEPT"***ENTER NAME OF FILE TO RECIEVE DISTANCES: "
      READ(11,10)FILE3(1)
70      CALL OPEN(2,FILE3,2,IER)
      IF(IER.NE.1)TYPE"ERROR ON OPEN OF DISTANCE FILE, IER= ",IER

```

```

CALL CHK0(IER)
IF(IER.NE.13)GOTO 75
TYPE"FILE DOES NOT EXIST. WILL CREATE IT FOR YOU."
CALL CPILW(FILE3,2,IER1)
IF(IER1.NE.1)TYPE"ERROR ON FILE CREATION. IER1= ",IER1
GOTO 70
75  CONTINUE
C***** STORE FILENAMES IN HEADER OF DISTANCE FILE
DO 100 J1=1,QTRBLK
100  HEADER(J1)=0.0
      DO 110 J1=1,13
      HEADER(J1)=FILE3(J1)
      J2=J1+13
      HEADER(J2)=FILE2(J1)
      J3=J1+26
110  HEADER(J3)=FILE1(J1)
      HEADER(40)=STS1
      HEADER(41)=STSL
      HEADER(42)=PTS1
      HEADER(43)=PTSL
      HEADER(46)=FSPFSQ      ;SWICH INDICATING IF OBS AND PHONS ARE SAME
      HEADER(48)=NSTSTB      ;# OBS TIME SLICES
      HEADER(49)=NPTSTB      ;# PHON TIME SLICES
      HEADER(50)=NPPTS       ;# WORDS PER TIME-SLICE
C***** DIVVY-UP EXTENDED MEMORY BETWEEN OBSERVATION SPECTRUM,
C      PHONET SPECTRUM, AND THE DISTANCE FILES.
111  NPFS=2+2*IFIX(FLOAT(NSTSTB)*NPPTS/(2*NDP)) ;# MEM BLKS FOR OBS FILE
      NPFP=2+2*IFIX(FLOAT(NPTSTB)*NPPTS/(2*NDP)) ;# MEM BLKS FOR PHON FILE
114  CONTINUE
116  J1=IFIX((XMEN-3)/2)*2      ;NEED EVEN # MEM BLKS
      IF(NPFP.LT.J1)NBFPH=NPFP
      IF(NPFP.GE.J1)NBFPH=J1
      J1=IFIX((XMEN-1-NBFPH)/2)*2
      IF(NPFS.LT.J1)NBFOBS=NPFS
      IF(NPFS.GE.J1)NBFOBS=J1
      NBFDIST=1
      IF(NBFOBS.GT.0)GOTO 118
      TYPE"ERROR: INSUFFICIENT EXTENDED MEMORY AVAILABLE."
      TYPE"NEED AT LEAST 14 1K BLOCKS."
      CALL VMEN(XMEN,IER)
      TYPE"YOU HAVE",XMEN," 1K BLOCKS OF EXTENDED MEMORY AVAILABLE."
      ACCEPT"=HOW MANY DO YOU WANT? ",XMEN
      XMEN=XMEN-9
      GOTO 114
118  CONTINUE
      HEADER(51)=XMEN
C***** INITIALIZE TO 1ST AVAILABLE EXT MEM BLK
      OBSMAPCNT=9      ;1ST EXT MEM BLK USED FOR OBSERVATIONS
      FONMAPCNT=OBSMAPCNT+NBFOBS      ;1ST EXT MEM BLK
                                      ;USED FOR PHONETS
      DISTMAPCNT=FONMAPCNT+NBFPH      ;1ST EXT MEM BLK
                                      ;USED FOR DISTANCES
      FONDANGCNT=0
      OBSLEFT=NDP

```

FILE: CHOOS  
LANGUAGE: FORTRAN 5  
DATE: September 21, 1982  
AUTHOR: D. Martin  
SUBJECT: Acoustic Analysis  
CALLING SEQUENCE: CHOOS  
DATE OF LAST REVISION: September 21, 1982

PURPOSE:

This program uses the Array Processor to choose the N-best phonet choices from a file of up to 512 distance measures. The number of choices, N, is selected in the routine which calls this, DSTN.

DESCRIPTION:

Location: DP4:BRATCHET

|       |          |            |
|-------|----------|------------|
| Size: | CHOOS.FR | 3994 bytes |
|       | CHOOS.RB | 2870 bytes |

PROGRAM USE:

This routine calls CHKJ and is called by DSTN. It uses the Array Processor to choose a selectable number of best distances, best being minimum. It fills the output file with the observation number, the observation energy, the phonet number of the maximum distance, the maximum distance, and consecutive ordered pairs for the choices: the first number being the phonet number, the second is the distance; the first pair is the best choice and the N<sup>th</sup> pair is the N<sup>th</sup> choice. Each file segment of 4+2N contains these

assignments in the order given here. The output file is an integer file.

Just prior to assignment, the energy value and each distance is checked, while real, for integer overflow. Any value larger than 32767 is assigned the value 32767, which is the largest integer the Eclipse S/250 recognizes. The number of choices is selected in routine DSTN.

A flowchart of this routine is in Figure 75. When called, the distances are loaded into Array Processor memory. After the energy and observation number are loaded into the output array DIST, the Array Processor is used to first get the input distance file maximum and corresponding phonet number. Then the routine enters a loop which successively gets the minimum distance and phonet number, replaces the minimum distance with the max, then gets the next minimum. This continues until the selected number of choices is obtained.

LIST OF VARIABLES:

| <u>VARIABLE</u> | <u>TYPE</u> | <u>PURPOSE</u>                |
|-----------------|-------------|-------------------------------|
| NPFSR           | Integer     | Energy in observation.        |
| NPTSTB          | Integer     | Number of phonets.            |
| HOLDIST         | Real Array  | Holds distances.              |
| WKOBS           | Real Array  | Holds distances in APM.       |
| JOUTDIST        | Integer     | Counts output array elements. |
| OBSBANGCNT      | Integer     | Counts observations done.     |

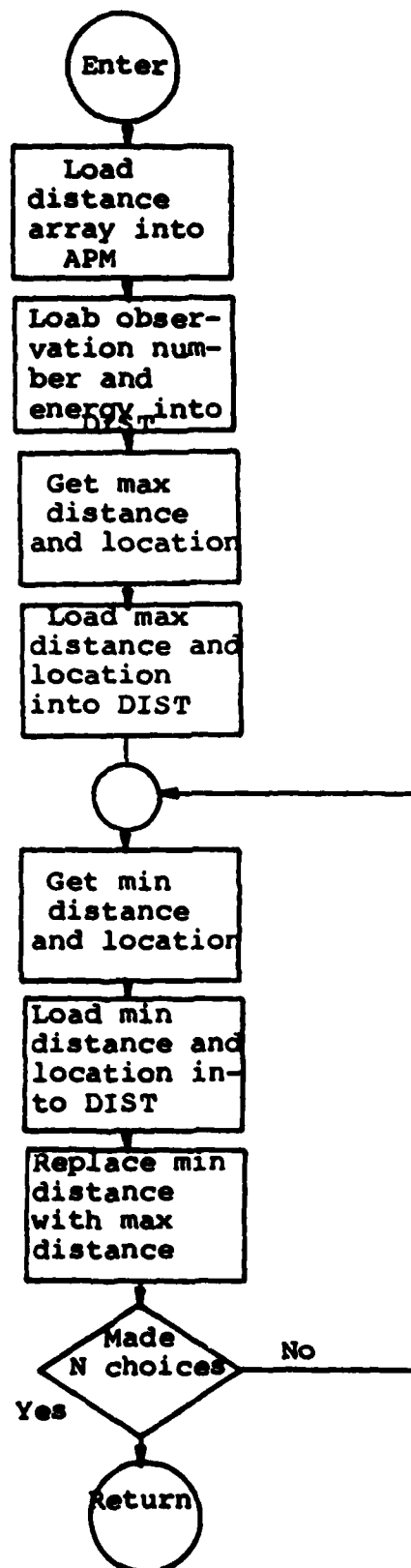


FIGURE 75. Flowchart of CHOOS

| <u>VARIABLE</u> | <u>TYPE</u> | <u>PURPOSE</u>                               |
|-----------------|-------------|----------------------------------------------|
| DIST            | Integer     | Output array.                                |
| MAXDIST         | Integer     | Maximum distance in<br>HOLDIST.              |
| MAXDISTLOC      | Integer     | Phonet number correspond-<br>ing to MAXDIST. |
| MINDIST         | Integer     | Minimum distance in<br>HOLDIST.              |
| MINDISTLOC      | Integer     | Phonet number correspond-<br>ing to MINDIST. |

SWITCH SETTINGS: None.

RELATED PROGRAMS:

PLTO, PLTA, PLTS, PLTN, PLTT, GRPH2, DRVR, DSTA, DSTN,  
S128, S64, CHKO, CHKJ, FCTR, GFDB.

```

C***** ROUTINE CHOOS. THIS ROUTINE IS FOR FORTRAN 5 ! !
C THIS ROUTINE USES THE ARRAY PROCESSOR TO CHOOSE THE N-BEST PHONETS
C FROM AN ARRAY OF UP TO 512 DISTANCE MEASURES. THE NUMBER OF
C CHOICES, N, IS SELECTED IN THE ROUTINE WHICH CALLS THIS ONE, DSTN.
C BY: CAPT DAN MARTIN
C DATE: 9/21/82
C SUBJ: ACOUSTIC ANALYSIS
C THIS ROUTINE CALLS CHXJ AND IS CALLED BY DSTN. IT FILLS AN OUTPUT
C FILE WITH THE OBSERVATION NUMBER, THE OBSERVATION ENERGY, THE
C PHONET NUMBER OF THE MAXIMUM DISTANCE, THE MAXIMUM DISTANCE,
C AND CONSECUTIVE ORDERED PAIRS FOR EACH CHOICE: (PHONET NUMBER,
C NEXT SMALLEST DISTANCE). THE OUTPUT FILE CONSISTS OF A HEADER
C OF ONE DISK BLOCK AND CONTIGUOUS UNITS OF 4+2N POINTS IN LENGTH,
C N BEING THE NUMBER OF CHOICES MADE. EACH UNIT CONTAINS THE ABOVE
C LISTED VALUES FOR EACH OBSERVATION.
C THIS ROUTINE ACCOMPLISHES THE FOLLOWING TASKS:
C 1) LOAD DISTANCE ARRAY INTO ARRAY PROCESSOR MEMORY
C 2) LOAD OBSERVATION NUMBER AND ENERGY INTO OUTPUT
C ARRAY DIST
C 3) GET MAX DISTANCE AND LOCATION
C 4) LOAD MAX DISTANCE AND LOCATION INTO DIST
C 5) GET MIN DISTANCE AND LOCATION
C 6) LOAD MIN DISTANCE AND LOCATION INTO DIST
C 7) REPLACE MIN DISTANCE WITH MAX DISTANCE
C 8) GO TO TASK 5 UNTIL ALL CHOICES ARE MADE
C FOR MORE INFORMATION SEE THE USERS MANUAL OR MY THESIS.

```

SUBROUTINE CHOOS

OVERLAY OCHOO

INCLUDE "ARRAYP:FSAPS.FR"

PARAMETER NDP=1024

PARAMETER NDPH1=1023

PARAMETER QTRDLK=256

REAL WKFON, WKODS, NPFSR, NPFP, FACTOR, FON, OBS, HOLDIST

REAL MAXDIST, MINDIST

INTEGER FSPFN, FILE2, HEADER, STS1, STSL, NSTSTB, NPSTS, NPFS

INTEGER DIST, FSPFSQ, FILE1, PTS1, PTSL, JOUTDIST

INTEGER NPTSTB, NPPTS, NPFP, FOSTR, FILE3, OUTD, FLAG, CD1, SKIP

INTEGER XNEN, NBFDIST, NBFODS, NBFPH, CNTODS, CNTFON, ODSHAPCNT

INTEGER FONHAPCNT, FONDANGCNT, ODSLEFT, NBFODS, NBFPH, NBFDIST

INTEGER DISTRHAPCNT, ODSBANGCNT, FONLEFT, ICNTODS, IFDC, CSPCT

INTEGER NPLTB, JFONHAPCNT, JOBSHAPCNT, DISTRHAPCNTMAX, NBFDIST

INTEGER JDISTNDR, JCNTODS, JCNTFON, ICNTFON, NFONLD, INITHEN

INTEGER MAXDISTLOC, MINDISTLOC, NCHOICES, NBRPHONES

COMMON / APH / WKODS(NDP), WKFON(NDP)

COMMON / VALS / OBS(NDP), FON(NDP), DIST(NDP), ODSLEFT, FSPFN

COMMON / VALS / INITHEN, FILE3(13), HEADER(QTRDLK), NPFSR, NPFP

COMMON / VALS / PTS1, PTSL, NPTSTB, NPPTS, NPFP, FOSTR, OUTD, NBFDIST

COMMON / VALS / NBFODS, NBFPH, CNTODS, CNTFON, ODSHAPCNT, FONHAPCNT

COMMON / VALS / FONDANGCNT, NBFODS, NBFPH, NBFDIST, DISTRHAPCNT

COMMON / VALS / ICNTODS, ICNTFON, IFDC, JFONHAPCNT, NFONLD

COMMON / VALS / ODSBANGCNT, FONLEFT, NPLTB, JOBSHAPCNT, NBFDIST

COMMON / VALS / DISTRHAPCNTMAX, JOUTDIST, JDISTNDR, JCNTODS, JCNTFON

COMMON / VALT / SKIP, STSL, NSTSTB, NPSTS, NPFS, XNEN, FACTOR, FSPFSQ

```

COMMON / VALT / FILE1(13),FILE2(13),FLAG,STS1
COMMON / VALU / CB1(0:CBMAX)
COMMON / VALU / HOLDIST(MDP),MAXDIST,MINDIST,MAXDISTLOC,MINDISTLOC
COMMON / VALU / NCHOICES,NBRPHONES

```

```

      SKIP=0
C***** INITIALIZE APS AND APH MAP FOR ERDB
      IF(SKIP.EQ.0)GOTO 502
      SKIP=0
      CALL APINIT(NIL,DTARAY,9,INITMEN,IER)
      CALL APMAP(DTARAY,0,4,IER)
502    CONTINUE
C***** LOAD ARRAY HOLDIST INTO WKOBS
      CALL APSETL(NPTSTB,IER)
      CALL CBSET(CB1,CBAAMH,HOLDIST,CBAXR,WKOBS,IER)
      CALL VLDR(CB1)
C***** LOAD OBSERVATION * INTO OUTPUT ARRAY DIST
      DIST(JOUTDIST)=OBSBANGCNT-1
      CALL CHKJ                      ;SEE IF DIST IS FULL.
      JOUTDIST=JOUTDIST+1
C***** LOAD OBSERVATION ENERGY NEXT
      NPFSR=NPFSR/100                ;SCALE ENERGY
      IF(NPFSR.GT.32767)NPFSR=32767
      DIST(JOUTDIST)=NPFSR
      CALL CHKJ
      JOUTDIST=JOUTDIST+1
C***** GET MAX DISTANCE AND ITS LOCATION
      CALL CBSET(CB1,CBAXR,WKOBS,CBAAMH,MAXDIST,CBAAMH,MAXDISTLOC,IER)
      CALL VNXR(CB1)
      MAXDISTLOC=MAXDISTLOC+1
      DIST(JOUTDIST)=MAXDISTLOC
      CALL CHKJ
      JOUTDIST=JOUTDIST+1
      DIST(JOUTDIST)=MAXDIST
      CALL CHKJ
      JOUTDIST=JOUTDIST+1
C***** GET MINIMUM DISTANCES AND THEIR LOCATIONS, I.E., PHONET #S
      DO 2000 J2=1,NCHOICES
      CALL CBSET(CB1,CBAXR,WKOBS,CBAAMH,MINDIST,CBAAMH,MINDISTLOC,IER)
      CALL VNNR(CB1)                ;GET MINIMUM
      DIST(JOUTDIST)=MINDISTLOC+1    ;LOCATION OF MINIMUM
      WKOBS(DIST(JOUTDIST))=MAXDIST
      CALL CHKJ
      JOUTDIST=JOUTDIST+1
      DIST(JOUTDIST)=MINDIST
      CALL CHKJ
2000   JOUTDIST=JOUTDIST+1

      RETURN
      END

```

|                        |                               |
|------------------------|-------------------------------|
| FILE:                  | FCTR                          |
| LANGUAGE:              | FORTRAN 5                     |
| AUTHOR:                | D. Martin                     |
| DATE:                  | September 21, 1982            |
| SUBJECT:               | Acoustic Analysis             |
| CALLING SEQUENCE:      | FCTR (W, M, H, Q, A, B, C, D) |
| DATE OF LAST REVISION: | September 21, 1982            |

**PURPOSE:**

This subroutine is called by GFDB and calls no sub-routines. It factors an integer into a linear combination of four terms. GFDB uses it to determine the unit (disk block, etc.) and starting data point in that unit at which specified data reside. For example, if the  $N^{\text{th}}$  element of an integer disk file is to be accessed, then GFDB will call FCTR to specify the disk block and location in that block where that  $N^{\text{th}}$  element resides.

**DESCRIPTION:**

Location: DP4:BRATCHET

|       |         |           |
|-------|---------|-----------|
| Size: | FCTR.FR | 605 bytes |
|       | FCTR.RB | 218 bytes |

**ARGUMENT STRUCTURE:**

| <u>ARGUMENTS</u> | <u>TYPE</u> | <u>PURPOSE</u>          |
|------------------|-------------|-------------------------|
| W                | Integer     | Integer to be factored. |
| M                | Integer     | Factor.                 |
| A                | Integer     | Coefficient of M.       |
| H                | Integer     | Factor.                 |

| <u>ARGUMENTS</u> | <u>TYPE</u> | <u>PURPOSE</u>      |
|------------------|-------------|---------------------|
| B                | Integer     | Coefficient of H.   |
| Q                | Integer     | Factor.             |
| C                | Integer     | Coefficient of Q.   |
| D                | Integer     | Coefficient of one. |

PROGRAM USE:

Values of W, M, H, and Q are passed to FCTR. FCTR returns values of A, B, C, and D for which:

$$W = AM + BH + CQ + D.$$

RELATED PROGRAMS:

PLTO, PLTS, PLTA, PLTN, PLTT, DRVR, DRSQ, DSTA, DSTN, S64, S128, GFDB, GRPH2, CHKO, CHKJ.

```

C***** ROUTINE FCTR.  THIS ROUTINE FOR FORTRAN 5 ! !
C      THIS SUBROUTINE IS CALLED BY CFDB AND CALLS NO SUBROUTINES.  IT
C      FACTORS AN INTEGER INTO A LINEAR COMBINATION OF FOUR 1-ST ORDER
C      TERMS.  CFDB USES IT TO DETERMINE THE DISK BLOCK AND STARTING
C      ELEMENT IN A DISK FILE GIVEN A SPECIFIED TIME SLICE OR SPECTRAL
C      SLICE NUMBER.
C      BY:  CAPT DAN MARTIN
C      DATE:  9/21/82
C      SUBJ:  ACOUSTIC ANALYSIS
C      SEE THE USERS MANUAL OR MY THESIS FOR MORE INFORMATION.
C      SUBROUTINE FCTR(W,M,H,Q,A,B,C,D)
C      INTEGER W,M,H,Q,A,B,C,D,X,Y
C*****  W=A*M+B*H+C*Q+D
C          A=W/M
C          X=W-A*M
C          B=X/H
C          Y=X-B*H
C          C=Y/Q
C          D=Y-C*Q
C      RETURN
C      END

```

|                        |                    |
|------------------------|--------------------|
| FILE:                  | GFDB               |
| LANGUAGE:              | FORTRAN 5          |
| DATE:                  | September 21, 1982 |
| AUTHOR:                | D. Martin          |
| SUBJECT:               | Acoustic Analysis  |
| CALLING SEQUENCE:      | GFDB (IA, IB)      |
| DATE OF LAST REVISION: | September 21, 1982 |

PURPOSE:

This routine is called by PLTA, PLTS, PLTN, PLTT, DSTA, and DSTN. It calls FCTR to compute the location of a specified data point.

DESCRIPTION:

Location: DP4:BRATCHET

|       |         |           |
|-------|---------|-----------|
| Size: | GFDB.FR | 644 bytes |
|       | GFDB.RB | 278 bytes |

ARGUMENT STRUCTURE:

| <u>ARGUMENT</u> | <u>TYPE</u> | <u>PURPOSE</u>                                                        |
|-----------------|-------------|-----------------------------------------------------------------------|
| IA              | Integer     | Passes file slice number.<br>Returns first disk block.                |
| IB              | Integer     | Passes number of integer words per slice.<br>Returns first data word. |

PROGRAM USE:

The plot routines PLTA, PLTS, PLTN, PLTT, DSTA, and DSTN use this routine to locate specified data.

```

C***** ROUTINE CFDB.  THIS ROUTINE IS FOR FORTRAN 5 ! !
C  THIS ROUTINE GETS THE 1ST DISK BLOCK (256 INTEGER
C  WORDS) AND THE 1ST ELEMENT OF THE ARRAY WHICH WAS LOADED
C  AT THAT DISK BLOCK.  THE FIRST ELEMENT OF THE ARRAY
C  IS NUMBER ONE.
C  BY:  CAPT DAN MARTIN
C  DATE:  9/21/82
C  SUBJ:  ACOUSTIC ANALYSIS
C  WHEN PASSED:  IA=FILE SLICE NUMBER
C                IB=NUMBER OF INTEGER WORDS PER SLICE
C  WHEN RETURNED:  IA=1ST DISK BLOCK
C                  IB=1ST DATA WORD
C  FOR MORE INFORMATION, SEE THE USERS MANUAL OR MY THESIS.

```

```

SUBROUTINE CFDB(IA,IB)
INTEGER W,A,B,C,D

```

```

W=IB*(IA-1)
CALL FCTR(W,1024,512,256,A,B,C,D)
IA=4*A+2*B+C
IB=D+1
RETURN
END

```

FILE: CHKJ  
LANGUAGE: FORTRAN 5  
DATE: September 21, 1982  
AUTHOR: D. Martin  
SUBJECT: Acoustic Analysis  
CALLING SEQUENCE: CHKJ  
DATE OF LAST REVISION: September 21, 1982

PURPOSE:

This routine is called by DSTN and calls no subroutines. It checks the number of distances accumulated, writes them if 1024 distances are accumulated, resets the counter, and increments a pointer to the next disk block to write.

DESCRIPTION:

Location: DP4:BRATCHET

|       |         |            |
|-------|---------|------------|
| Size: | CHKJ.FR | 2324 bytes |
|       | CHKJ.RB | 1012 bytes |

LIST OF VARIABLES:

| <u>VARIABLE</u> | <u>TYPE</u>   | <u>PURPOSE</u>                   |
|-----------------|---------------|----------------------------------|
| JOUTDIST        | Integer       | Number of distances accumulated. |
| DIST            | Integer Array | Holds distances accumulated.     |
| NDBFDIST        | Integer       | Disk block to write output to.   |

PROGRAM USE:

This routine is designed for use by DSTN, to be called each time JOUTDIST is incremented.

RELATED PROGRAMS:

PLTO, PLTA, PLTS, PLTN, PLTT, FCTR, GRPH2, CHOOS, DRVR,  
DRSQ, S128, S64, DSTA, DSTN, CKHJ, CHKO.

```

C***** ROUTINE CHKJ. THIS ROUTINE FOR FORTRAN 5 ! !
C THIS ROUTINE IS CALLED BY DSTN AND CALLS NO SUBROUTINES. IT
C CHECKS THE NUMBER OF DISTANCES ACCUMULATED, WRITES THEN IF
C 1024 HAVE BEEN ACCUMULATED, RESETS THE COUNTER, AND INCREMENTS
C A POINTER TO THE NEXT DISK BLOCK TO WRITE TO.
C BY: CAPT DAN MARTIN
C DATE: 9/21/82
C SUBJ: ACOUSTIC ANALYSIS
C THIS ROUTINE IS DESIGNED FOR USE BY DSTN TO BE CALLED EACH TIME
C JOUTDIST IS INCREMENTED. FOR MORE INFORMATION, SEE THE
C USERS MANUAL FOR DSTN OR THIS ROUTINE, OR SEE MY THESIS.

```

```

SUBROUTINE CHKJ
OVERLAY DCHKJ
PARAMETER NDP=1024
PARAMETER NDP1=1023
PARAMETER QTRBLK=254
REAL WKFON,WKOBDS,NPFSR,NPFFR,FACTOR,FON,OBS,HOLDIST
REAL MAXDIST,MINDIST
INTEGER FSPFH,FILE2,HEADER,STS1,STSL,NSTSTB,NPSTS,NPFS
INTEGER DIST,FSPFSQ,FILE1,PTS1,PTSL,JOUTDIST
INTEGER NPTSTB,NPPTS,NPFP,FDSTR,FILE3,OUTD,FLAG,CB1,SKIP
INTEGER XMEM,NBFDIST,NBFOBS,NBFBPH,CNTOBS,CNTFON,OBShAPCNT
INTEGER FONHAPCNT,FONBANGCNT,OBSLEFT,NQBFOBS,NQBFBPH,NQBFDIST
INTEGER DISTHAPCNT,OBShANGCNT,FONLEFT,ICNTOBS,IFBC,CSPCT
INTEGER NPLTB,JFONHAPCNT,JOShAPCNT,DISTHAPCNTMAX,NBFDIST
INTEGER JDISTNBR,JCNTOBS,JCNTFON,ICNTFON,NFONLD,INITHEM
INTEGER MAXDISTLOC,MINDISTLOC,NCHOICES,NBRPHONES

COMMON / APM / WKOBDS(NDP),WKFON(NDP)
COMMON / VALS / OBS(NDP),FON(NDP),DIST(NDP),OBSLEFT,FSPFH
COMMON / VALS / INITHEM,FILE3(13),HEADER(QTRBLK),NPFSR,NPFFR
COMMON / VALS / PTS1,PTSL,NPTSTB,NPPTS,NPFP,FDSTR,OUTD,NBFDIST
COMMON / VALS / NBFOBS,NBFBPH,CNTOBS,CNTFON,OBShAPCNT,FONHAPCNT
COMMON / VALS / FONBANGCNT,NQBFOBS,NQBFBPH,NQBFDIST,DISTHAPCNT
COMMON / VALS / ICNTOBS,ICNTFON,IFBC,JFONHAPCNT,NFONLD
COMMON / VALS / OBShANGCNT,FONLEFT,NPLTB,JOShAPCNT,NBFDIST
COMMON / VALS / DISTHAPCNTMAX,JOUTDIST,JDISTNBR,JCNTOBS,JCNTFON
COMMON / VALT / SKIP,STSL,NSTSTB,NPSTS,NPFS,XMEM,FACTOR,FSPFSQ
COMMON / VALT / FILE1(13),FILE2(13),FLAG,STS1
COMMON / VALU / HOLDIST(NDP),MAXDIST,MINDIST,MAXDISTLOC,MINDISTLOC
COMMON / VALU / NCHOICES,NBRPHONES

```

```

IF(JOUTDIST.LE.NDP)GOTO 365 ;DIST ARRAY NOT FULL YET
JOUTDIST=1 ;NEED WRITE TO DISK IF FULL
IF(OUTD.NE.0)WRITE(OUTD,9000)(DIST(J2),J2=1,NDP)
C***** WRITE 4 QTR BLKS OF DISTANCES
CALL EWRB(2,NBFDIST,36+NQBFOBS+NQBFBPH,4,IER)
IF(IER.NE.1)TYPE"ERROR ON EWRB OF DISTANCES! IER=",IER
NBFDIST=NBFDIST+4
365 CONTINUE
9000 FORMAT(5G12.4)
RETURN

```

END

FILE: PLTO  
LANGUAGE: FORTRAN 5  
DATE: September 21, 1982  
AUTHOR: D. Martin  
SUBJECT: Graphics  
CALLING SEQUENCE: PLTO  
DATE OF LAST REVISION: September 21, 1982

PURPOSE:

This routine calls one of four plot routines selected from a displayed menu. The graph is displayed on a Tektronix 4010-1 Graphics Terminal using the routine GRPH2 by G. Shaw as modified by L. Kizer and D. Zambon.

DESCRIPTION:

Location: DP4:BRATCHET

|       |         |            |
|-------|---------|------------|
| Size: | PLTO.FR | 1491 bytes |
|       | PLTO.RB | 1650 bytes |

PROGRAM USE:

This program is called by DRVR and calls PLTS, PLTA, PLTN, and PLTT. When called, this routine presents a menu from which the operator is to choose one of five options, one being to return to DRVR. If the option to plot spectrum is chosen, PLTS is called to plot the spectrum of a time slice from a disk file computed by DRSQ. If the option to plot all distances is selected, PLTA is called to plot distances between one observation and selected phonets from a disk file computed by DSTA. If the option to plot N-best distances is selected, PLTN is called to plot observation

energy and selected distances from a disk file computed by DSTN. Finally, if the option to plot an integer file is selected, then one unit of up to 512 elements from any integer disk file is plotted.

LIST OF VARIABLES:

| <u>VARIABLE</u> | <u>TYPE</u> | <u>PURPOSE</u>                                                      |
|-----------------|-------------|---------------------------------------------------------------------|
| OUTD            | Integer     | Currently unused, but can be used as unit number for screen output. |
| SKIP            | Integer     | Switch to select plot option.                                       |

SWITCH SETTINGS:

| <u>SWITCH</u> | <u>SETTINGS</u>                                                                   |
|---------------|-----------------------------------------------------------------------------------|
| SKIP          | = 6 to call PLTS to plot spectrum                                                 |
|               | = 7 to call PLTA to plot distances                                                |
|               | = 8 to call PLTN to plot N-best distances, worst distance, and observation energy |
|               | = 9 to call PLTT to plot a segment of an integer file                             |
|               | = 10 to return to DRVR                                                            |

RELATED PROGRAMS:

PLTA, PLTS, PLTN, PLTT, DRVR, DSTA, DSTN, DRSQ, CHKJ, CHKO, CHOOS, FCTR, GFDB, S128, S64, GRPH2.

```

C***** ROUTINE PLTO. THIS ROUTINE IS FOR FORTRAN 5 ! !
C THIS ROUTINE CALLS PLTA, PLTS, PLTN, AND PLTT , AND IS CALLED
C BY BRVR. IT SELECTS THE APPROPRIATE PLOT ROUTINE TO PLOT THE
C DATA AS REQUESTED.
C BY: CAPT DAN MARTIN
C DATE: 9/21/82
C SUBJ: ACOUSTIC ANALYSIS
C FOR MORE INFORMATION SEE THE USERS MANUAL OR MY THESIS.
      SUBROUTINE PLTO
      OVERLAY OPLTO
      INCLUDE "ARRAYP:F5APS.FR"
      PARAMETER MNDP=1024
      PARAMETER QTRBLK=256
      REAL DATA,XMIN,XMAX
      INTEGER CSPCT,FILE2,HEADER,FSPFH,OUTD,ODSNBR,PHON1,PHONL
      INTEGER DIST,FONNBR,FRSTDP,FRSTDBLK,ICNT,D1,IER,SKIP

      COMMON / VALS / DATA(MNDP),DIST(1024),ODSNBR,FSPFH,OUTD,PHON1
      COMMON / VALS / PHONL,ICNT,IER,FONNBR,FRSTDP,FRSTDBLK,FILE2(13)
      COMMON / VALS / HEADER(QTRBLK),SKIP
      COMMON / VALT / D1(37),ISXMEM

      TYPE"****YOU ARE NOW IN ROUTINE PLTO.FR****"
      OUTD=10

40      TYPE"==WHAT WILL YOU HAVE ME PLOT?"
      TYPE"****CHOOSE OPTION****"
      TYPE"="
      TYPE"6 TO PLOT SPECTRUM."
      TYPE"7 TO PLOT ALL DISTANCES."
      TYPE"8 TO PLOT N-BEST DISTANCES."
      TYPE"9 TO PLOT INTEGER FILE."
      TYPE"10 TO RETURN TO THE PROGRAM WHICH CALLED THIS ONE."
      ACCEPT"==ENTER DESIRED OPTION: ",SKIP

100     IF(SKIP.NE.6)GOTO 110
        CALL PLTS
        GO TO 40
110     IF(SKIP.NE.7)GOTO 120
        CALL PLTA
        GOTO 40
120     IF(SKIP.NE.8)GOTO 130
        CALL PLTN
        GOTO 40
130     IF(SKIP.NE.9)GOTO 140
        CALL PLTT
        GOTO 40
140     IF(SKIP.EQ.10)GOTO 1000

        TYPE"ERROR: COULDN'T FIND YOUR OPTION."
        TYPE"YOU SELECTED OPTION: ",SKIP
        GOTO 40
1000    CALL RESET
        CALL OVEXIT(OPLTO,IER)

```

CALL CHECK(1ER)  
RETURN  
END

|                        |                    |
|------------------------|--------------------|
| FILE:                  | PLTA               |
| LANGUAGE:              | FORTRAN 5          |
| DATE:                  | September 21, 1982 |
| AUTHOR:                | D. Martin          |
| SUBJECT:               | Graphics           |
| CALLING SEQUENCE:      | PLTA               |
| DATE OF LAST REVISION: | September 21, 1982 |

PURPOSE:

This routine plots distances from a disk file computed by DSTA. Distances between an observation and all phonets specified by the operator are displayed on a Tektronix 4010-1 Graphics terminal using the plot routine GRPH2 by G. Shaw as modified by L. Kizer and D. Zambon.

DESCRIPTION:

Location: DP4:BRATCHET

|       |         |            |
|-------|---------|------------|
| Size: | PLTA.FR | 4371 bytes |
|       | PLTA.RB | 6712 bytes |

PROGRAM USE:

This program is called by PLTO and calls GFDB and GRPH2. It plots distances from a disk file between a specified observation and the first and last phonets specified.

The operator is first prompted for an option to print numerical values of data to be plotted. The data values plotted can be displayed on the operator's terminal, written to the line printer, or not displayed at all. Next, the

operator is prompted for the name of the disk file holding the data to be plotted. The file header is read and identifying information from it displayed for the operator. Then the desired observation number, and the numbers of the first and last phonet to be plotted, are requested. When signalled to proceed, the program will display the plot.

LIST OF VARIABLES:

| <u>VARIABLE</u> | <u>TYPE</u>   | <u>PURPOSE</u>                  |
|-----------------|---------------|---------------------------------|
| OUT             | Integer       | Switch for data output option.  |
| OUTD            | Integer       | Channel number.                 |
| FILE3           | Integer Array | Holds disk file name.           |
| OBSNBR          | Integer       | Observation number.             |
| PHON1           | Integer       | First phonet.                   |
| PHONL           | Integer       | Last phonet.                    |
| FONNBR          | Integer       | Number of distances to plot.    |
| FRSTDBLK        | Integer       | First disk block to read.       |
| FRSTDP          | Integer       | First data point in disk block. |
| DATA            | Real Array    | Holds data to plot.             |
| DIST            | Integer Array | Data read from disk.            |

SWITCH SETTINGS:

| <u>SWITCH</u> | <u>SETTING</u>                         |
|---------------|----------------------------------------|
| OUT           | = 10 for screen output of plotted data |
|               | = 12 for line printer output           |
|               | = 0 for no output of numbers.          |

RELATED PROGRAMS:

PLTO, PLTS, PLTN, PLTT, DRSQ, S128, S64, DSTA, DSTN,  
CHOOS, CHKJ, CHKO, FCTR, GFDB, DRVR, GRPH2.

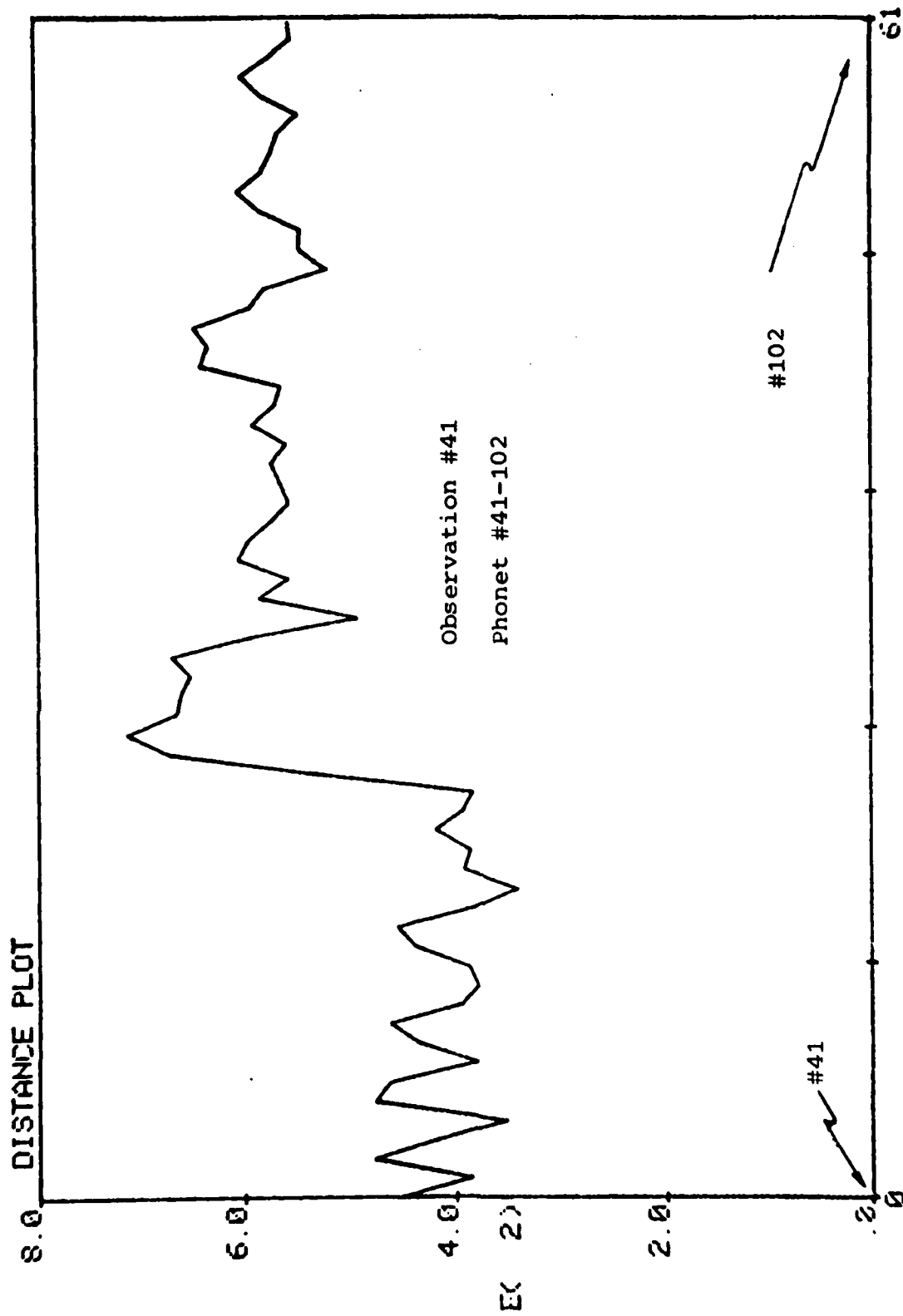


FIGURE 76. Example of PLTA

```

C***** ROUTINE PLTA. THIS ROUTINE IS FOR FORTRAN 5 ! !
C THIS ROUTINE IS CALLED BY PLTO AND CALLS THE PLOT ROUTINES GRPH2
C AND CFDD. GRPH2 WAS WRITTEN BY G.SHAU AND MODIFIED BY L.KIZER
C AND D.ZAMBON. IT IS IN THE SPEECH LAB PROGRAM LIBRARY.
C THIS ROUTINE PLOTS DISTANCES COMPUTED BY DSTA. IT WILL PLOT
C DISTANCES BETWEEN A SPECIFIED OBSERVATION AND SPECIFIED PHONETS.
C THE USER IS PROMPTED FOR THIS INFORMATION AS WELL AS FOR THE
C FILE NAME.
C BY: CAPT DAN MARTIN
C DATE: 9/21/82
C SUBJ: ACOUSTIC ANALYSIS
C PLTA ACCOMPLISHES THE FOLLOWING TASKS:
C 1) GET DISTANCE FILE
C 2) DISPLAY INFORMATION FROM HEADER TO IDENTIFY THE FILE
C 3) GET OBSERVATION NUMBER AND PHONET NUMBERS
C 4) LOCATE DATA IN DISK FILE
C 5) PREPARE DATA FOR PLOT ROUTINE GRPH2
C 6) CALL THE PLOT ROUTINE
C FOR MORE INFORMATION SEE THE USERS MANUAL OR MY THESIS.

```

```

SUBROUTINE PLTA
OVERLAY OPLTA
INCLUDE"ARRAYP:FSAPS.FR"
PARAMETER HNDP=512
PARAMETER QTRBLK=256
REAL DATA,XMIN,XMAX
INTEGER CSPCT,FILE3,HEADER,FSPFN,OUTD,OBSENR,PHON1,PHONL
INTEGER DIST,FONNBR,FRSTDP,FRSTDBLK,ICNT,D1,IER,OUT

```

```

COMMON / VALS / DATA(HNDP),DIST(1024),OBSENR,FSPFN,OUTD,PHON1
COMMON / VALS / PHONL,ICNT,IER,FONNBR,FRSTDP,FRSTDBLK,FILE3(13)
COMMON / VALS / HEADER(QTRBLK)
COMMON / VALT / D1(37)

```

```

30 TYPE"****YOU ARE NOW IN ROUTINE PLTD.FR****"
TYPE"DO YOU WANT OUTPUT OF DATA PLOTTED?"
TYPE"ENTER: 10 FOR SCREEN OUPUT OF NUMBERS"
TYPE" 12 FOR PRINTED NUMERICAL OUTPUT"
ACCEPT" 0 FOR NO NUMERICAL OUTPUT: ",OUT
IF(OUT.EQ.0.OR.OUT.EQ.10.OR.OUT.EQ.12)GOTO 40
TYPE"ERROR: ILLEGAL INPUT."
GOTO 30
40 OUTD=10

```

```

C***** GET DISTANCE FILE NAME
50 ACCEPT"ENTER NAME OF FILE HOLDING DISTANCES: "
READ(11,100)FILE3(1)
100 FORMAT(S13)
120 CALL CLOSE(2,IER)
CALL OPEN(2,FILE3,2,IER)
CALL CHECK(1ER)
IF(1ER.NE.1)TYPE"ERROR ON OPEN OF SPECTRAL FILE, 1ER: ",1ER
GOTO 180
150 CONTINUE

```

```

C***** GET NUMBERS OF 1ST & LAST TIME-SLICES OF SPECTRUM
180  CALL RDBLK(2,0,HEADER,1,IER)
    CALL CHECK(IER)
    WRITE(OUTD,9110)
    WRITE(OUTD,9100)(HEADER(15),15=1,65)
    WRITE(OUTD,9000)(HEADER(15),15=1,13)
    WRITE(OUTD,9001)(HEADER(15),15=14,26)
    WRITE(OUTD,9004)HEADER(40),HEADER(41)
    WRITE(OUTD,9002)(HEADER(15),15=27,39)
    WRITE(OUTD,9005)HEADER(42),HEADER(43)
    ACCEPT"IS THIS THE FILE YOU'RE AFTER? (1=YES/0=NO): ",FSPFN
    IF(FSPFN.EQ.0)GOTO 50
190  ACCEPT"ENTER THE OBSERVATION NUMBER: ",OBSNBR
    IF(OBSNBR.GE.HEADER(40).AND.OBSNBR.LE.HEADER(41))GOTO 200
    TYPE"ERROR: THIS NUMBER MUST LIE IN THE RANGE"
    TYPE"[",HEADER(40),",",HEADER(41),"]"
    GOTO 190
200  ACCEPT"ENTER THE FIRST PHONET NUMBER: ",PHON1
    IF(PHON1.GE.HEADER(42).AND.PHON1.LE.HEADER(43))GOTO 220
    TYPE"ERROR: THIS NUMBER MUST LIE IN THE RANGE"
    TYPE "[",HEADER(42),",",HEADER(43),"]"
    GOTO 200
220  ACCEPT"ENTER THE LAST PHONET NUMBER: ",PHONL
    IF(PHONL.GE.PHON1.AND.PHONL.LE.HEADER(43))GOTO 240
    TYPE"ERROR: THIS NUMBER MUST BE IN THE RANGE"
    TYPE"[",PHON1,",",HEADER(43),"]"
    GOTO 200
240  FONNBR=PHONL-PHON1+1
    IF(FONNBR.LE.HNDP)GOTO 260
    TYPE"ERROR: YOU REQUESTED MORE THAN 512 DISTANCES."
    GOTO 200
260  CONTINUE
    FONNBR=HEADER(43)-HEADER(42)+1
C***** GET DISK BLOCK AND 1ST DATA POINT
    IA=OBSNBR-HEADER(40)+1
    IB=FONNBR
    CALL CFDB(IA,IB)
    FRSTDBLK=IA+1
    FRSTDP=IB
    CALL RDBLK(2,FRSTDBLK,DIST,3,ICNT,IER)
    IF(IER.NE.9)GOTO 300
    TYPE"READ EOF! SUCCESSFULLY TRANSFERRED",ICNT," QTR BLKS"
    TYPE"PROCEEDING WITH ",ICNT,"QTR BLOCKS TRANSFERRED."
    IER=1
300  CALL CHECK(IER)
    CALL CLOSE(2,IER)
    CALL CHECK(IER)

    IF(OUT.EQ.0)GOTO 310
    IF(OUT.EQ.12)OUTD=12
C***** MOVE THE DATA UP-FRONT AND INTO THE ARRAY TO PLOT
310  IL=PHON1-HEADER(42)+1
    IU=PHONL-HEADER(42)+1
    FRSTDP=FRSTDP+IL-1

```

```

DO 320 J1=1,IU-IL+1
320 DATA(J1)=DIST(J1+PRSTD-1)
   IF(OUT.EQ.0)GOTO 322
   WRITE(OUTD,9100)(DATA(I5),I5=1,IU-IL+1)
322 ACCEPT"OK TO CONTINUE? (ENTER ANY NUMBER) ",I5
   IF(I5.NE.0)GOTO 324
324 CONTINUE
   FOMNDR=PHONL-PHON1+1 .
   CALL CRPH2(" DISTANCE PLOT",1,DATA,U,FOMNDR,0,XMIN,XMAX,0)
   ACCEPT"OK TO CONTINUE? (ENTER ANY NUMBER) ",I5
   IF(I5.EQ.0)GOTO 350
350 CONTINUE
9100 FORMAT(5G12.4)
9110 FORMAT(T5,"HERE'S THE FIRST 65 ELEMENTS OF THE HEADER:")
9000 FORMAT(T3,"DISTANCE FILE NAME IS: ",I3S2)
9001 FORMAT(T5,"OBSERVATION FILE NAME WAS: ",I3S2)
9002 FORMAT(T5,"PHONET FILE NAME WAS: ",I3S2)
9004 FORMAT(T2,"FIRST OBSERVATION TS# = ",I5," AND LAST = ",I6)
9005 FORMAT(T2,"FIRST PHONET TS# = ",I5," AND LAST = ",I6)

RETURN
END

```

FILE: PLTS  
LANGUAGE: FORTRAN 5  
DATE: September 21, 1982  
AUTHOR: D. Martin  
SUBJECT: Graphics  
CALLING SEQUENCE: PLTS  
DATE OF LAST REVISION: September 21, 1982

PURPOSE:

This routine plots spectrum from a disk file computed by DRSQ. The plot is displayed on a Tektronix 4010-1 Graphics Terminal using the routine GRPH2 by G. Shaw as modified by L. Kizer and D. Zambon.

DESCRIPTION:

Location: DP4:BRATCHET

|       |         |            |
|-------|---------|------------|
| Size: | PLTS.FR | 3785 bytes |
|       | PLTS.RB | 5302 bytes |

PROGRAM USE:

This program is called by PLTO and calls GFDB and GRPH2. It plots spectrum from a disk file computed by DRSQ.

The operator is first prompted for an option to print numerical values of data to be plotted. The data values plotted can be displayed on the operator's terminal, written to the line printer, or not displayed at all. Next, the operator is prompted for the name of the disk file holding the data to be plotted. The file header is read and

identifying information from it is displayed. The time slice number corresponding to the desired spectrum is requested and the dB or magnitude option prompted. When the program is signalled by the operator to proceed, it will display the plot. The value of the first component will be the observation energy.

LIST OF VARIABLES:

| <u>VARIABLE</u> | <u>TYPE</u>   | <u>PURPOSE</u>                        |
|-----------------|---------------|---------------------------------------|
| OUT             | Integer       | Switch for data output option.        |
| OUTD            | Integer       | Channel number.                       |
| FILE2           | Integer Array | Holds disk file name.                 |
| OBSNBR          | Integer       | Time slice number.                    |
| FONNBR          | Integer       | Number of spectral components.        |
| FRSTDBLK        | Integer       | First disk block to read.             |
| FRSTDP          | Integer       | First data point in disk block.       |
| HEADER(57)      | Integer       | Number of spectral components.        |
| DATA            | Real Array    | Data to be plotted - passed to GRPH2. |
| DIST            | Real Array    | Data read from disk.                  |
| IDP             | Integer       | Switch for dB or magnitude plot.      |

SWITCH SETTINGS:

| <u>SWITCH</u> | <u>SETTING</u>                         |
|---------------|----------------------------------------|
| OUT           | = 10 for screen output of plotted data |
|               | = 12 for line printer output           |
|               | = 0 for no output of numbers           |

SWITCH

SETTING

IDP        =    1 for dB versus component number  
           =    0 for magnitude versus component number

RELATED PROGRAMS:

PLTA, PLTO, PLTN, PLTT, DRVR, DRSQ, DSTA, DSTN, CHKJ,  
CHKO, CHOOS, FCTR, GFDB, S128, S64, GRPH2.

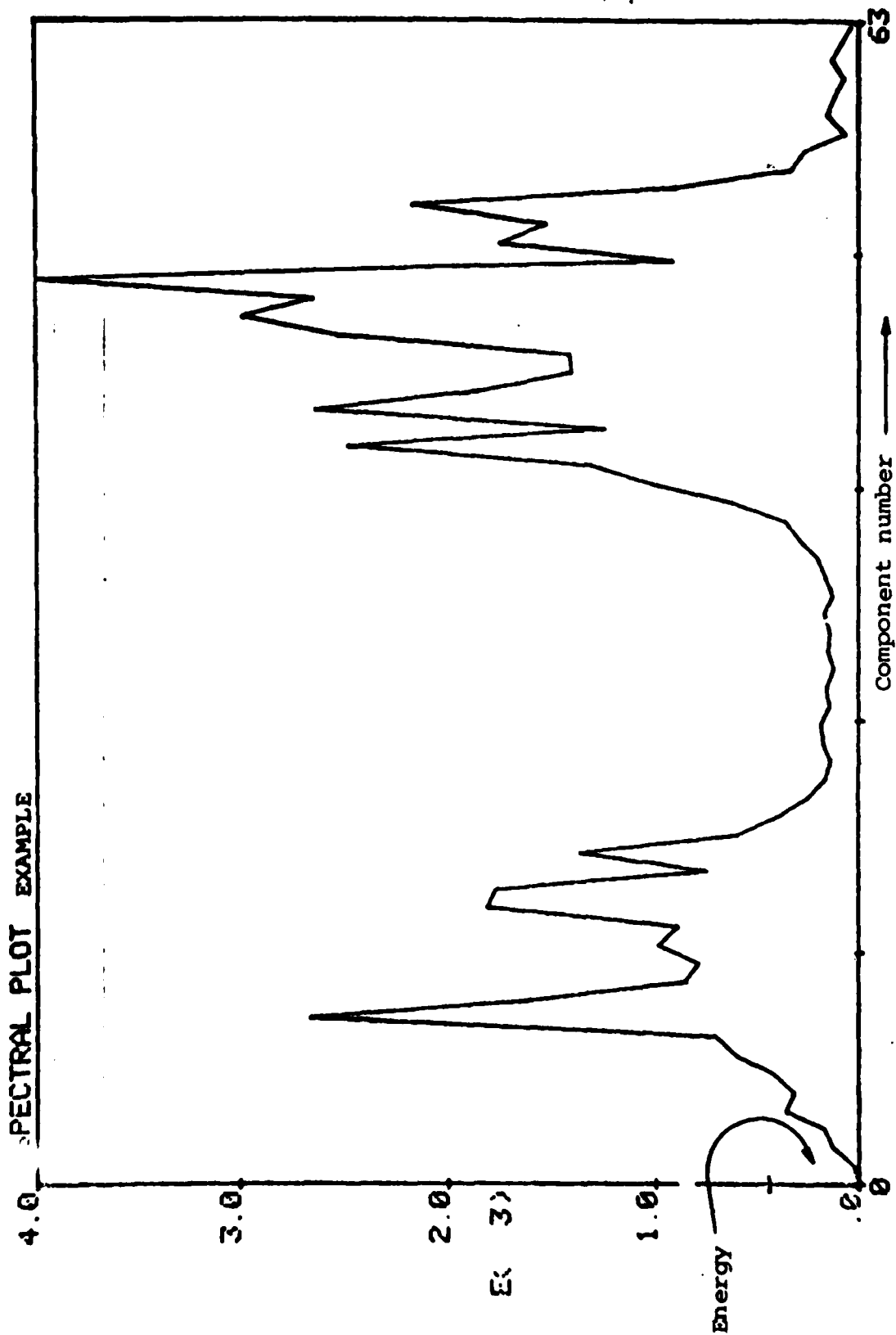


FIGURE 77. Spectral plot of observation number 45. One-hundred twenty-eight (128) point, nonoverlapped Hamming window, energy normalized.

```

C***** ROUTINE PLTS. THIS ROUTINE IS FOR FORTRAN 5 1 1
C THIS ROUTINE IS CALLED BY PLTO AND CALLS THE PLOT ROUTINES GRPH2
C AND GPBD. GRPH2 WAS WRITTEN BY C.SHAW AND MODIFIED BY L.KIZER
C AND D.ZAMBON. IT IS IN THE SPEECH LAB PROGRAM LIBRARY.
C THIS ROUTINE PLOTS SPECTRUM OF TIME SLICES COMPUTED BY DRSS.
C THE USER IS PROMPTED FOR THE TIME SLICE NUMBER, FILE NAME,
C AND OTHER NECESSARY INFORMATION.
C BY: CAPT DAN MARTIN
C DATE: 9/21/82
C SUBJ: ACOUSTIC ANALYSIS
C PLTA ACCOMPLISHES THE FOLLOWING TASKS:
C 1) GET SPECTRAL FILE
C 2) DISPLAY INFORMATION FROM HEADER TO IDENTIFY THE FILE
C 3) GET OBSERVATION NUMBER (TIME SLICE NUMBER)
C 4) LOCATE DATA IN DISK FILE
C 5) PREPARE DATA FOR PLOT ROUTINE GRPH2
C 6) CALL THE PLOT ROUTINE
C FOR MORE INFORMATION SEE THE USERS MANUAL OR MY THESIS.

```

```

SUBROUTINE PLTS
OVERLAY OPLTS
INCLUDE"ARRAYP:F5APS.FR"
PARAMETER HNDB=512
PARAMETER QTRBLK=256
REAL DATA,XMIN,XMAX,DIST
INTEGER CSPCT,FILE2,HEADER,FSPFN,OUTD,OBSDNR,PHON1,PHONL
INTEGER FOMNDR,FRSTD,FRSTDBLK,ICNT,D1,IER,OUT

COMMON / VALS / DATA(HNDB),DIST(512),OBSDNR,FSPFN,OUTD,PHON1
COMMON / VALS / PHONL,ICNT,IER,FOMNDR,FRSTD,FRSTDBLK,FILE2(13)
COMMON / VALS / HEADER(QTRBLK)
COMMON / VALT / D1(37)

```

```

30 TYPE"****YOU ARE NOW IN ROUTINE PLTS.FR****"
TYPE"DO YOU WANT OUTPUT OF DATA PLOTTED?"
TYPE"ENTER: 10 FOR SCREEN OUTPUT OF NUMBERS"
TYPE" 12 FOR PRINTED NUMERICAL OUTPUT"
ACCEPT" 0 FOR NO NUMBERS: ",OUT
IF(OUT.EQ.0.OR.OUT.EQ.10.OR.OUT.EQ.12)GOTO 40
TYPE"ERROR: ILLEGAL INPUT."
GOTO 30
40 OUTD=10

```

```

C***** GET SPECTRAL FILE NAME
50 ACCEPT"ENTER NAME OF FILE HOLDING SPECTRUM: "
READ(11,100)FILE2(1)
100 FORMAT(S13)
120 CALL CLOSE(4,IER)
CALL OPEN(4,FILE2,2,IER)
CALL CHECK(IER)
IF(IER.NE.1)TYPE"ERROR ON OPEN OF SPECTRAL FILE, IER= ",IER
GOTO 180
150 CONTINUE
C***** GET NUMBERS OF 1ST & LAST TIME-SLICES OF SPECTRUM

```

```

180  CALL RDBLK(4,0,HEADER,1,IER)
    CALL CHECK(IER)
    WRITE(OUTD,9110)
    WRITE(OUTD,9100)(HEADER(15),15=1,65)
    WRITE(OUTD,9000)(HEADER(15),15=14,26)
    WRITE(OUTD,9001)(HEADER(15),15=1,13)
    WRITE(OUTD,9004)HEADER(55),HEADER(56)
    ACCEPT"IS THIS THE FILE YOU'RE AFTER? (1=YES/0=NO): ",FSPFM
    IF(FSPFM.EQ.0)GOTO 50
190  ACCEPT"ENTER THE TIME SLICE NUMBER: ",OBSNBR
    IF(OBSNBR.GE.HEADER(55).AND.OBSNBR.LE.HEADER(56))GOTO 200
    TYPE"ERROR: THIS NUMBER MUST LIE IN THE RANGE"
    TYPE"C",HEADER(55),"",HEADER(56),"J"
    GOTO 190
200  CONTINUE
240  FOMNBR=HEADER(57)
    IF(FOMNBR.LE.HMNDP)GOTO 260
    TYPE"ERROR: YOU REQUESTED MORE THAN 512 SPECTRAL COMPONENTS."
    GOTO 200
260  CONTINUE
C***** DISK BLOCK AND ARRAY LOCATION OF 1ST DATA POINT IS
    IA=OBSNBR-HEADER(55)+1
    IB=HEADER(57)*2
    CALL CFDB(IA,IB)
    FRSTDBLK=1+IA
    FRSTDP=(IB-1)/2+1
    CALL RDBLK(4,FRSTDBLK,DIST,4,ICNT,IER)
    IF(IER.NE.9)GOTO 300
    TYPE"READ EOF! SUCCESSFULLY TRANSFERRED",ICNT," QTR BLKS"
    TYPE"PROCEEDING WITH ",ICNT,"QTR BLOCKS TRANSFERRED."
    IER=1
300  CALL CHECK(IER)
    CALL CLOSE(4,IER)
    CALL CHECK(IER)

    IF(OUT.EQ.12)OUTD=12
C***** MOVE THE DATA UP-FRONT AND INTO THE ARRAY TO PLOT
    ACCEPT"ENTER 1 FOR DB PLOT/ 0 FOR MAGNITUDE: ",IDP
    IF(IDP.EQ.1)GOTO 310
    DO 305 J1=1,HEADER(57)
305  DATA(J1)=DIST(J1+FRSTDP-1)
    GOTO 320
310  DO 315 J1=1,HEADER(57)
315  DATA(J1)=20*ALOG10(DIST(J1+FRSTDP-1))
320  IF(OUT.EQ.0)GOTO 321

    WRITE(OUTD,9100)(DATA(15),15=1,FOMNBR)
321  ACCEPT"OK TO CONTINUE? (ENTER ANY NUMBER) ",15
322  IF(15.EQ.0)GOTO 324
324  CONTINUE
    CALL GRPH2(" SPECTRAL PLOT",1,DATA,U,FOMNBR,0,XMIN,XMAX,0)
    ACCEPT"OK TO CONTINUE? (ENTER ANY NUMBER) ",15
    IF(15.EQ.0)GOTO 350
350  CONTINUE

```

9100 FORMAT(5G12.4)  
9110 FORMAT(T5,"HERE'S THE FIRST 45 ELEMENTS OF THE HEADER:")  
9000 FORMAT(T3,"SPEECH FILE NAME IS: ",13S2)  
9001 FORMAT(T5,"SPECTRAL FILE NAME WAS: ",13S2)  
9004 FORMAT(T2,"FIRST SPECTRAL TS= ",15," AND LAST = ",16)

RETURN  
END

|                        |                    |
|------------------------|--------------------|
| FILE:                  | PLTN               |
| LANGUAGE:              | FORTRAN 5          |
| DATE:                  | September 21, 1982 |
| AUTHOR:                | D. Martin          |
| SUBJECT:               | Graphics           |
| CALLING SEQUENCE:      | PLTN               |
| DATE OF LAST REVISION: | September 21, 1982 |

PURPOSE:

This routine plots distances from a disk file computed by DSTN. The plot is displayed on a Tektronix 4010-1 Graphics Terminal using the routine GRPH2 by G. Shaw as modified by L. Kizer and D. Zambon.

DESCRIPTION:

Location: DP4:BRATCHET

|       |         |            |
|-------|---------|------------|
| Size: | PLTN.FR | 4132 bytes |
|       | PLTN.RB | 5970 bytes |

PROGRAM USE:

This program is called by PLTO and calls GFDB and GRPH2. For a specified observation, this routine plots the observation energy as the first data point and the distances at locations corresponding to their phonet numbers. In this way, a sort of scatter plot is obtained.

The operator is first prompted for an option to print numerical values of the data to be plotted. The values plotted can be displayed on the operator's terminal, written to the line printer, or not displayed at all. Next, the

operator is prompted for the name of the disk file holding the data to be plotted. The file header is read and identifying information displayed. Then, the observation number is requested. The program will display the plot when signalled by the operator to proceed.

#### LIST OF VARIABLES:

| <u>VARIABLE</u> | <u>TYPE</u>   | <u>PURPOSE</u>                                      |
|-----------------|---------------|-----------------------------------------------------|
| OUT             | Integer       | Switch for data output option.                      |
| OUTD            | Integer       | Channel number.                                     |
| FILE3           | Integer Array | Holds disk file name.                               |
| OBSNBR          | Integer       | Observation number.                                 |
| FONNBR          | Integer       | Number of data points (one plus number of phonets). |
| FRSTDBLK        | Integer       | First disk block to read.                           |
| FRSTDP          | Integer       | First data point in disk block.                     |
| DATA            | Real Array    | Holds data to be plotted.                           |
| DIST            | Integer Array | Data read from disk.                                |

#### SWITCH SETTINGS:

| <u>SWITCH</u> | <u>SETTING</u>                         |
|---------------|----------------------------------------|
| OUT           | = 10 for screen output of plotted data |
|               | = 12 for line printer output           |
|               | = 0 for no output of numbers           |

#### RELATED PROGRAMS:

PLTO, PLTS, PLTA, PLTT, DRVR, DRSQ, S64, S128, DSTA, DSTN, CHOOS, CHKJ, CHKO, FCTR, GFDB, GRPH2.

# DISTANCE PLOT

Observation #41  
Phonet #41-102

Energy

Worst distance

10 best (minimum)  
distance choices

0 41

62

-102

FIGURE 78. Example of PLTN.

```

C***** ROUTINE PLTN. THIS ROUTINE IS FOR FORTRAN 5 ! !
C THIS ROUTINE IS CALLED BY PLTO AND CALLS THE PLOT ROUTINES
C GRPH2 AND CFBD. GRPH2 WAS WRITTEN BY G.SHAU AND MODIFIED BY
C L.KIZER AND D.ZAMBON. IT IS IN THE SPEECH LAB PROGRAM LIBRARY.
C PLTN PLOTS DISTANCES COMPUTED BY DSTN. FOR A SPECIFIED
C OBSERVATION, IT WILL PLOT THE OBSERVATION ENERGY AS THE FIRST
C DATA POINT, AND THE DISTANCES AT LOCATIONS CORRESPONDING TO
C THIER PHONET NUMBERS. IN THIS WAY A SORT OF SCATTER PLOT OF
C N-BEST DISTANCES IS PLOTTED.
C BY: CAPT DAN MARTIN
C DATE: 9/21/82
C SUBJ: ACOUSTIC ANALYSIS
C PLTN ACCOMPLISHES THE FOLLOWING TASKS:
C 1) GET DISTANCE FILE
C 2) DISPLAY INFORMATION FROM HEADER TO IDENTIFY THE FILE
C 3) GET OBSERVATION NUMBER AND PHONET NUMBERS
C 4) LOCATE DATA IN DISK FILE
C 5) PREPARE DATA FOR PLOT ROUTINE
C 6) CALL THE PLOT ROUTINE
C FOR MORE INFORMATION SEE MY THESIS OR THE USERS MANUAL.

```

```

SUBROUTINE PLTN
OVERLAY OPLTN
INCLUDE"ARRAYP:F5APS.FR"
PARAMETER HNDDP=512
PARAMETER QTRBLK=256
REAL DATA,XMIN,XMAX
INTEGER CSPCT,FILE3,HEADER,FSPFN,OUTD,OBSNBR,PHON1,PHONL
INTEGER DIST,FONNBR,FRSTD,FRSTDBLK,ICNT,D1,IER,D2,OUT

```

```

COMMON / VALS / DATA(HNDDP),DIST(1024),OBSNBR,FSPFN,OUTD,PHON1
COMMON / VALS / PHONL,ICNT,IER,FONNBR,FRSTD,FRSTDBLK,FILE3(13)
COMMON / VALS / HEADER(QTRBLK)
COMMON / VALT / D1(37)
COMMON / VALU / D2(2054),NCHOICES,NBRPHONES

```

```

30 TYPE"****YOU ARE NOW IN ROUTINE PLTD.FR****"
TYPE"DO YOU WANT OUTPUT OF DATA PLOTTED?"
TYPE"ENTER: 10 FOR SCREEN OUTPUT OF NUMBERS"
TYPE" 12 FOR PRINTED NUMERICAL OUTPUT"
ACCEPT" 0 FOR NO NUMBERS: ",OUT
IF(OUT.EQ.0.OR.OUT.EQ.10.OR.OUT.EQ.12)GOTO 40
TYPE"ERROR: ILLEGAL INPUT."
GOTO 30
40 OUTD=10

```

```

C***** GET DISTANCE FILE NAME
50 ACCEPT"ENTER NAME OF FILE HOLDING DISTANCES: "
READ(11,100)FILE3(1)
100 FORMAT(S13)
120 CALL CLOSE(2,IER)
CALL OPEN(2,FILE3,2,IER)
CALL CHECK(IER)
C***** GET NUMBERS OF 1ST & LAST TIME-SLICES OF SPECTRUM

```

```

180  CALL RDBLK(2,0,HEADER,1,IER)
      CALL CHECK(IER)
      WRITE(OUTD,9110)
      WRITE(OUTD,9100)(HEADER(15),15=1,65)
      WRITE(OUTD,9000)(HEADER(15),15=1,13)
      WRITE(OUTD,9001)(HEADER(15),15=14,26)
      WRITE(OUTD,9004)HEADER(40),HEADER(41)
      WRITE(OUTD,9002)(HEADER(15),15=27,39)
      WRITE(OUTD,9005)HEADER(42),HEADER(43)
      ACCEPT"IS THIS THE FILE YOU'RE AFTER? (1=YES/0=NO): ",FSPFN
      IF(FSPFN.EQ.0)GOTO 50
190  ACCEPT"ENTER THE OBSERVATION NUMBER: ",OBSNBR
      IF(OBSNBR.GE.HEADER(40).AND.OBSNBR.LE.HEADER(41))GOTO 200
      TYPE"ERROR: THIS NUMBER MUST LIE IN THE RANGE"
      TYPE"[",HEADER(40),",",HEADER(41),"]"
      GOTO 190
200  CONTINUE
240  FOMNBR=HEADER(43)-HEADER(42)+2
      IF(FOMNBR.LE.HNDP)GOTO 260
      TYPE"ERROR: YOU REQUESTED MORE THAN 512 DISTANCES."
      GOTO 1000
260  CONTINUE
C***** GET DISK BLOCK AND 1ST DATA POINT IS
      IA=OBSNBR-HEADER(40)+1
      IB=4+2*HEADER(47)
      CALL GFDB(IA,IB)
      FRSTDBLK=IA+1
      FRSTDP=IB
      CALL RDBLK(2,FRSTDBLK,DIST,3,ICNT,IER)
      IF(IER.NE.9)GOTO 300
      TYPE"READ EOF! SUCCESSFULLY TRANSFERRED",ICNT," QTR BLKS"
      TYPE"PROCEEDING WITH ",ICNT,"QTR BLOCKS TRANSFERRED."
      IER=1
300  CALL CHECK(IER)

      IF(OUT.EQ.12)OUTD=12

C***** ZERO-OUT ARRAY DATA
      DO 400 JJ=1,FOMNBR+1
400  DATA(JJ)=0.0
C***** FIRST DATA POINT IS OBSERVATION ENERGY
      DATA(1)=DIST(FRSTDP+1)
      TYPE"OBSERVATION NUMBER=",DIST(FRSTDP)
      ACCEPT"IS THIS THE ONE YOU WANT? (1=YES/0=NO): ",FSPFN
      IF(FSPFN.EQ.0)GOTO 120
C***** THEN FILL DATA WITH DISTANCES
      DO 410 JJ=FRSTDP+2,3+FRSTDP+2*HEADER(47),2
410  DATA(DIST(JJ)+1)=DIST(JJ+1)
      IF(OUT.NE.0)WRITE(OUTD,9100)(DATA(15),15=1,HEADER(43)-HEADER(42)+1)
310  ACCEPT"OK TO CONTINUE? (ENTER ANY NUMBER) ",15
      IF(15.EQ.0)GOTO 312
312  CONTINUE
      CALL GRPH2(" DISTANCE PLOT",1,DATA,U,FOMNBR,0,XMIN,XMAX,0)
      ACCEPT"OK TO CONTINUE? (ENTER ANY NUMBER) ",15

```

```

      IF(I5.EQ.0)GOTO 350
350   CONTINUE
      CALL RESET
9100   FORMAT(5G12.4)
9110   FORMAT(T5,"HERE'S THE FIRST 65 ELEMENTS OF THE HEADER:")
9000   FORMAT(T3,"DISTANCE FILE NAME IS: ",13S2)
9001   FORMAT(T5,"OBSERVATION FILE NAME WAS: ",13S2)
9002   FORMAT(T5,"PHONET FILE NAME WAS: ",13S2)
9004   FORMAT(T2,"FIRST OBSERVATION TS# = ",15," AND LAST = ",16)
9005   FORMAT(T2,"FIRST PHONET TS# = ",15," AND LAST = ",16)

1000   RETURN
      END

```

FILE: PLTT  
LANGUAGE: FORTRAN 5  
DATE: September 21, 1982  
AUTHOR: D. Martin  
SUBJECT: Graphics  
CALLING SEQUENCE: PLTT  
DATE OF LAST REVISION: September 21, 1982

PURPOSE:

This routine plots an integer disk file in units of from one to 512 integer words on a Tektronix 4010-1 Graphics Terminal using the routine GRPH2 by G. Shaw as modified by L. Kizer and D. Zambon.

DESCRIPTION:

Location: DP4:BRATCHET

|       |         |            |
|-------|---------|------------|
| Size: | PLTT.FR | 2840 bytes |
|       | PLTT.RB | 3290 bytes |

PROGRAM USE:

This program is called by PLTO and calls GFDB and GRPH2. It will plot segments of an integer disk file; the segment length can be from one to 512 integer words. The routine was designed to be used to plot segments of speech files and to plot distances between several consecutive observations and phonets. The operator is prompted for the disk file name, the segment number, and the segment size. To illustrate, say the header of a file computed by either DRSQ, DSTA, or DSTN is to be examined. Using this

routine, the operator could specify the first segment of 256 words along with a print option to examine the header. The plot could be discarded. On the other hand, suppose a curious segment of a speech file is to be examined. One determines which disk block contains the curious segment, and uses PLTT to plot that disk block.

#### LIST OF VARIABLES:

| <u>VARIABLE</u> | <u>TYPE</u>   | <u>PURPOSE</u>                  |
|-----------------|---------------|---------------------------------|
| OUT             | Integer       | Switch for data output option.  |
| OUTD            | Integer       | Unit number.                    |
| FILE3           | Integer Array | Holds disk file name.           |
| OBSNBR          | Integer       | Disk file segment.              |
| PHONL           | Integer       | Disk file segment length.       |
| FRSTDBLK        | Integer       | First disk block to read.       |
| FRSTDPT         | Integer       | First data point in disk block. |
| DATA            | Real Array    | Holds data to be plotted.       |
| DIST            | Integer Array | Data read from disk.            |

#### SWITCH SETTINGS:

| <u>SWITCH</u> | <u>SETTING</u>                                  |
|---------------|-------------------------------------------------|
| OUT           | = 10 for screen output of numbers to be plotted |
|               | = 12 for line printer output                    |
|               | = 0 for no numerical output                     |

#### RELATED PROGRAMS:

PLTO, PLTA, PLTN, PLTS, DRVR, DRSQ, S128, S64, DSTA, DSTN, CHOOS, CHKJ, CHKO, FCTR, GFDB, GRPH2.

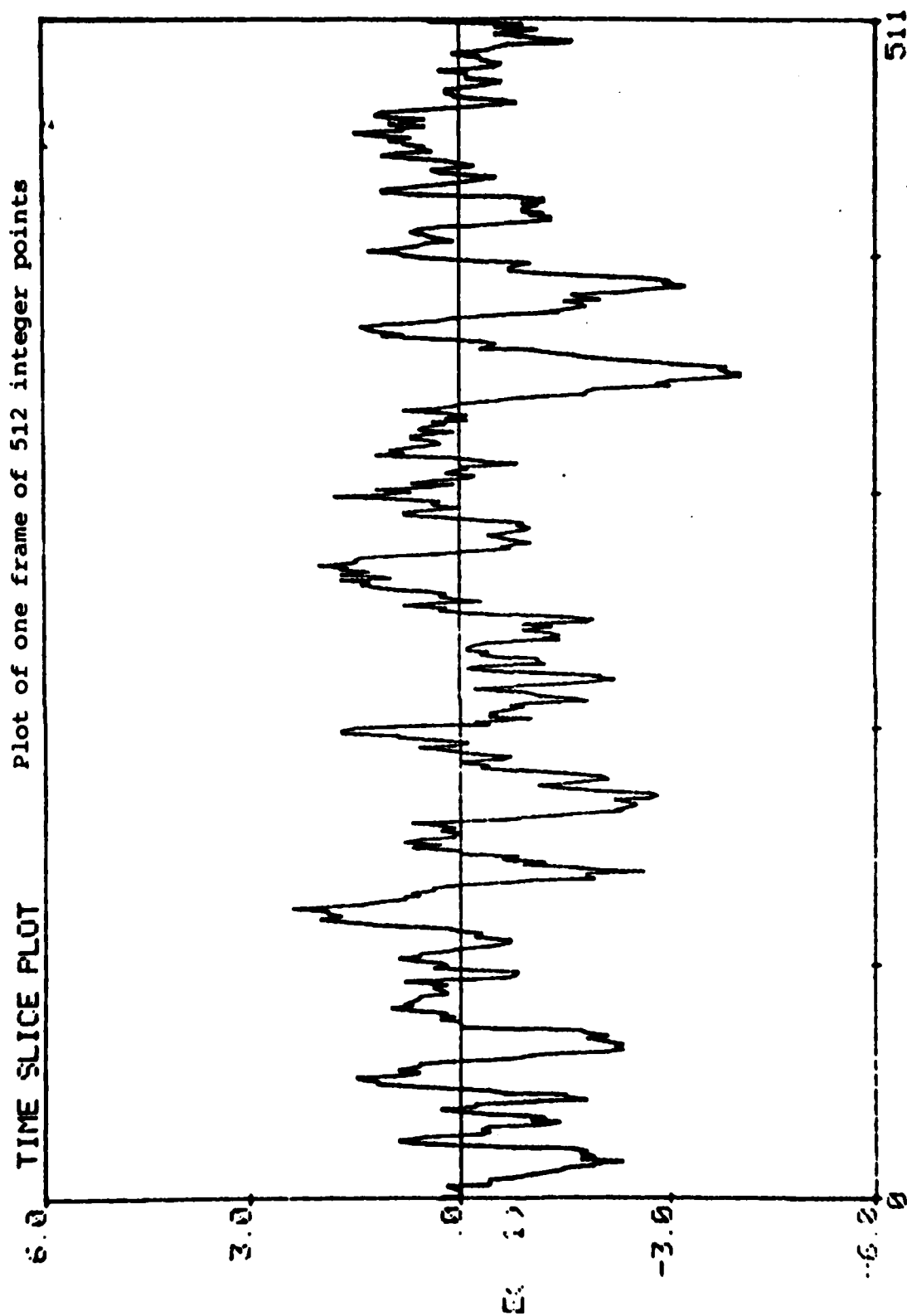


FIGURE 79. Example of PLTT

```

C***** ROUTINE PLTT. THIS ROUTINE IS FOR FORTRAN 5 I I
C THIS ROUTINE IS CALLED BY PLTO AND CALLS THE PLOT ROUTINES GRPH2
C AND CFDD. GRPH2 WAS WRITTEN BY G.SHAW AND MODIFIED BY L.KIZER
C AND D.ZANBOM. IT IS IN THE SPEECH LAB PROGRAM LIBRARY.
C THIS ROUTINE PLOTS INTEGER DISK FILES. IT WILL PLOT SEGMENTS
C OF THE FILE; EACH SEGMENT IS SPECIFIED BY ITS NUMBER AND POINT
C LENGTH. THE POINT LENGTH CAN BE FROM 1 TO 512 INTEGER WORDS.
C THE USER IS PROMPTED FOR THIS INFORMATION AS WELL AS FOR THE
C FILE NAME.
C BY: CAPT DAN MARTIN
C DATE: 9/21/82
C SUBJ: ACOUSTIC ANALYSIS
C PLTA ACCOMPLISHES THE FOLLOWING TASKS:
C 1) GET FILE
C 3) GET UNIT NUMBER AND UNIT SIZE
C 4) LOCATE DATA IN DISK FILE
C 5) PREPARE DATA FOR PLOT ROUTINE GRPH2
C 6) CALL THE PLOT ROUTINE
C FOR MORE INFORMATION SEE THE USERS MANUAL OR MY THESIS.

```

```

SUBROUTINE PLTT
OVERLAY DPLTT
INCLUDE"ARRAYP:F5APS.FR"
PARAMETER HNBP=512
PARAMETER QTRBLK=256
REAL DATA,XMIN,XMAX
INTEGER CSPCT,FILE3,HEADER,FSPFN,OUTD,OBSNBR,PHON1,PHONL
INTEGER DIST,FONNBR,FRSTDP,FRSTDBLK,ICNT,D1,IER,OUT

```

```

COMMON / VALS / DATA(HNBP),DIST(1024),OBSNBR,FSPFN,OUTD,PHON1
COMMON / VALS / PHONL,ICNT,IER,FONNBR,FRSTDP,FRSTDBLK,FILE3(13)
COMMON / VALS / HEADER(QTRBLK)
COMMON / VALT / D1(37)

```

```

30  TYPE"****YOU ARE NOW IN ROUTINE PLTT.FR****"
    TYPE"DO YOU WANT OUTPUT OF DATA PLOTTED?"
    TYPE"ENTER: 10 FOR SCREEN OUPUT OF NUMBERS"
    TYPE"          12 FOR PRINTED NUMERICAL OUTPUT"
    ACCEPT"          0 FOR NO NUMERICAL OUTPUT: ",OUT
    IF(OUT.EQ.0.OR.OUT.EQ.10.OR.OUT.EQ.12)GOTO 40
    TYPE"ERROR: ILLEGAL INPUT."
    GOTO 30
40  OUTD=10

```

```

60  CONTINUE
C***** GET INTEGER FILE NAME
ACCEPT"ENTER NAME OF FILE HOLDING INTEGER DATA: "
READ(11,100)FILE3(1)
100  FORMAT(813)
120  CALL OPEN(5,FILE3,1,IER)
    CALL CHECK(1ER)
190  ACCEPT"ENTER THE TIME SLICE NUMBER: ",OBSNBR
220  ACCEPT"ENTER THE NUMBER OF POINTS IN TIME SLICE: ",PHONL
    IF(PHONL.LE.HNBP)GOTO 260

```

```

        TYPE"ERROR: YOU REQUESTED MORE THAN 512 POINTS."
        GOTO 220
260    CONTINUE
C***** DISK BLOCK THAT HOLDS 1ST DATA POINT IS
        IA=ODSNDR
        IB=PHONL
        CALL CFDB(IA,IB)
        FRSTDBLK=IA
        FRSTDPIB
        CALL RDBLK(5,FRSTDBLK,DIST,4,ICNT,IER)
        IF(IER.NE.9)GOTO 300
        TYPE"READ EOF! SUCCESSFULLY TRANSFERRED",ICNT," QTR BLKS"
        TYPE"PROCEEDING WITH ",ICNT,"QTR BLOCKS TRANSFERRED."
        IER=1
300    CALL CHECK(IER)
        CALL CLOSE(5,IER)
        CALL CHECK(IER)

        IF(OUT.EQ.0)GOTO 310
        IF(OUT.EQ.12)OUTD=12
C***** MOVE THE DATA UP-FRONT AND INTO THE ARRAY TO PLOT
310    IU=PHONL
        DO 320 J1=1,IU
320    DATA(J1)=DIST(J1+FRSTDPIB-1)
        IF(OUT.EQ.0)GOTO 322
        TYPE"OUTD=",OUTD
        WRITE(OUTD,9100)(DATA(I5),I5=1,IU)
322    ACCEPT"OK TO CONTINUE? (ENTER ANY NUMBER) ",I5
        IF(I5.NE.0)GOTO 324
324    CONTINUE
        CALL GRPH2(" TIME SLICE PLOT",1,DATA,U,IU,0,XMIN,XMAX,0)
        ACCEPT"OK TO CONTINUE? (ENTER ANY NUMBER) ",I5
        IF(I5.EQ.0)GOTO 350
350    CONTINUE
9100    FORMAT(5G12.4)

        RETURN
        END

```

FILE: CHKO  
LANGUAGE: FORTRAN 5  
DATE: September 21, 1982  
AUTHOR: D. Martin  
SUBJECT: Acoustic Analysis  
CALLING SEQUENCE: CHKO (IER)  
DATE OF LAST REVISION: September 21, 1982

PURPOSE:

This routine is called by DSTA and DSTN, and calls no subroutine. It checks the value of IER returned from an OPEN file attempt. It returns a caution statement if the unit number is in use.

DESCRIPTION:

Location: DP4:BRATCHET

|       |         |           |
|-------|---------|-----------|
| Size: | CHKO.FR | 586 bytes |
|       | CHKO.RB | 288 bytes |

LIST OF VARIABLES: None.

ARGUMENT STRUCTURE:

| <u>ARGUMENT</u> | <u>TYPE</u> | <u>PURPOSE</u>                          |
|-----------------|-------------|-----------------------------------------|
| IER             | Integer     | Holds error code from OPEN system call. |

PROGRAM USE:

To be called by DSTA and DSTN to provide caution statement.

RELATED PROGRAMS:

PLTO, PLTA, PLTS, PLTT, PLTN, FCTR, GRPH2, GRDB, DRVR,  
DRSQ, DSTA, DSTN, CHKJ, CHOOS, S128, S64.

```

C***** ROUTINE CHK0.  THIS ROUTINE IS FOR FORTRAN 5 ! !
C      THIS ROUTINE IS CALLED BY DSTA AND DSTN, AND CALLS NO SUBROUTINES.
C      IT CHECKS THE VALUE OF IER RETURNED FROM AN OPEN FILE ATTEMPT.
C      IT RETURNS A CAUTION STATEMENT TO THE SCREEN IF THE UNIT NUMBER
C      IS IN USE.
C      BY:  CAPT DAN MARTIN
C      DATE: 9/21/82
C      SUBJ: ACOUSTIC ANALYSIS
C      FOR MORE INFORMATION SEE THE USERS MANUAL FOR DSTA, DSTN, OR THIS
C      ROUTINE.  OR SEE MY THESIS.

```

```

      SUBROUTINE CHK0(IER)
      OVERLAY OCHK0
      INTEGER IER
      IF(IER.NE.3096)GOTO 100
      IER=1
      TYPE"*****CAUTION:  UNIT NUMBER IS IN USE!*****"
100    RETURN
      END

```

## VITA

Dan Martin was born on 24 March 1952 in Wheeling, West Virginia. He graduated from Walnut Ridge High School in Columbus, Ohio, in 1970 and enlisted in the U.S. Air Force that same year. While assigned to the Air Force Flight Dynamics Laboratory, Wright-Patterson AFB, Ohio, he attended Wright State University in Fairborn, Ohio. He majored in Applied Mathematics and upon graduation in December 1976, he received his B.S. degree. He was commissioned upon his graduation from Officers Training School at Lackland AFB, Texas, on 24 March 1977. He attended Communications-Electronics Officers Course, Keesler AFB, Mississippi from April 1977 to December 1977. He was then assigned to the USAF Frequency Management Office, Bolling AFB, Washington, D.C., and served as Radio Frequency Engineer. In August 1980, he was assigned to the School of Engineering, Air Force Institute of Technology, to pursue a B.S.E.E. degree. Upon his graduation in March 1982, Captain Martin was selected to remain at the Air Force Institute of Technology to pursue an M.S.E.E. with emphasis in Communications and Electromagnetic Field Theory. He is a member of Tau Beta Pi and Eta Kappa Nu, and the father of one son in whom he takes great pride. The author's permanent address is:

1236 Huntly Drive  
Columbus, Ohio 43227