TM-4652/200/00

DESCRIPTION AND ANALYSIS OF THE VICENS-REDDY PREPROCESSING AND SEGMENTATION ALGORITHMS

4 December 1970

SERIES	BASE NO. / VOL
TM	4652 /200/00

TECHNICAL MEMORANDUM

(TM Series)

The work reported herein was supported by the Advanced Research Projects Agency of the Department of Defense under Contract DAHC15-67-C-0149, ARPA Order No. 1327, Amendment No. 3, Program Code No. 1D30, and 1P10.

DESCRIPTION AND ANALYSIS OF THE VICENS-REDDY PREPROCESSING AND SEGMENTATION ALGORITHMS

by

Iris Kameny

H. Barry Ritea

4 December 1970

SYSTEM DEVELOPMENT CORPORATION
2500 COLORADO AVE.
SANTA MONICA
CALIFORNIA
90406

(Page ii blank)

ABSTRACT

This document provides a detailed description and analysis of the preprocessing and segmentation procedures used in the Vicens-Reddy speech recognition system.

TABLE OF CONTENTS

(Page vi blank)

LIST OF TABLES

LIST OF FIGURES

1. INTRODUCTION

This document provides a detailed description of the preprocessing and segmentation procedures used in the Vicens-Reddy speech recognition system [1]. In addition, explanations are given for a large number of previously unexplained heuristics used in the program. In particular, special attention is called to the discussions of the functions CLOS1 and PROXIM.

We wish to thank Gary Goodman of the Stanford Artificial Intelligence Project for his help in furnishing processed speech samples and program listings to aid in the implementation and checkout of the system on the IBM 360/67.

2. PREPROCESSING

Preprocessing consists of dividing the speech flow into minimal segments and determining the zero-crossing count and the maximum amplitude for each of three different frequency bands within the speech spectrum. The frequency bands are: 150 Hz - 900 Hz, 900 Hz - 2200 Hz, and 2200 Hz - 5000 Hz. As vowels contain, in general, more reliable information than other phonemes, the choice of the cutoff values was dictated by known parameter values for the vowels as developed by Peterson and Barney [2].

2.1 AMPLITUDES

Let an arbitrary wave within a minimal segment be represented by a discrete function f_i, whose values are the ordinates of this wave at equidistant points. The amplitude of the wave on the minimal segment is then defined to be*

$$\max_{1 \le i \le n} \{f_i\} - \min_{1 \le i \le n} \{f_i\}$$

The amplitude is thus measured from the lowest peak to the highest peak, where the possible range is from 0 to -10 volts.

Let A1, A2, and A3 denote the amplitudes of the wave on a minimal segment in the frequency bands 150 - 900 Hz, 900 - 2200 Hz, and 2200 - 5000 Hz, respectively.

---

*For sine waves, this expression is proportional to the square root of the average power of the signal over the minimal segment (see Appendix).

The following table illustrates the A-to-D conversion for amplitudes:

Table 1. A to-D Conversion of Amplitudes

Actual Amplitude (A1,A2,A3)	Volts into A/D Converter	Raw Binary from A/D Converter (2's Complement)		Output of A/D Conversion Routine		
				Binary	Octal	Decimal
0	0	100000	000000	000000	0	0
	-4.9	111111	111111	011111	37	31
1/2 of maximum possible amplitude	-5	000000	000000	100000	40	32
maximum possible amplitude	-10	011111	111111	111111	77	63

2.2 ZERO-CROSSINGS

The zero-crossings of the wave on the minimal segment are the number of sign changes of $f_1, \ldots f_n$. Let Z1, Z2, and Z3 denote the zero-crossing counts for the three frequency bands 150 - 900 Hz, 900 - 2200 Hz, and 2200 - 5000 Hz, respectively. The zero-crossing count is normalized into 7 bits.

The following table illustrates the A-to-D conversion of the zero-crossings:

Table 2. A-to-D Conversion of Zero-Crossings

Actual Hz	Zero-Crossings in 10 Msec.	Volts into A/D Converter	Raw Binary from A/D Converter (2's Complement)	Output of A/D Conversion Routine	
				Octal	Decimal
0	0	0	100000 000000	0	0
		-4.9	111111 111111		
2500	50	-5	000000 000000	62	50
5000	100	-10	011111 111111	144	100

Each zero-crossing unit represents 50 Hz. The range is from 0 - 100 corresponding to 0 Hz - 5000 Hz. The input hardware is adjusted for an acceptance level of 0.03 V on the original signal, i.e., the zero-crossings are counted only if the amplitude of the original signal is higher than 0.03 V.

2.3 THE PREPROCESSING PROGRAM

Originally, the Vicens-Reddy program had a main driver that called a subroutine KLUDGE to run the recorder, effect the A-to-D conversion (as discussed in Sections 2.1 and 2.2), and build the Q-matrix.

The KLUDGE subroutine computes the two variables:

MAXAMP = maximum amplitude of all A1, A2, and A3 amplitudes over the whole speech sample, and

MAXAM1 = maximum of all A1 amplitudes over the whole speech sample,*

and, depending upon their relative values, the following actions were taken:

Table 3. "KLUDGE" Table for MAIN Package

MAXAMP	MAXAM1	RESULT
≤58	≤58	Message returned to user: "Get a bit closer, please."
>58	>62	No normalization
>58	≤62	Normalization as follows: $A1 = \frac{A1 \cdot 64}{MAXAM1}$ $A2 = \frac{A2 \cdot 64}{MAXAM1}$ For all A1, A2, and A3 in all rows of Q-matrix $A3 = \frac{A3 \cdot 64}{MAXAM1}$ (If the resulting A2 or A3 was >127, it was reset = 127.)

*This implies that MAXAM1 ≤ MAXAMP.

A new program RECORD was later written which contains a modified subroutine KLUDGE also called KLUDGE. One of the reasons for RECORD was to free the user from immediately having to speak after the recorder was started. It incorporates tests for determining when the start of speech occurs and terminates when there is no noticeable input for more than 320 msec. RECORD calls KLUDGE to process each live message being recorded. KLUDGE detects the maximum A1 amplitude, and if it is less than 50, KLUDGE returns to the RECORD "weak signal" return. The return is presently handled as if the signal were stronger and goes on to the smoothing process. If KLUDGE detects a maximum A1 amplitude greater than 50, which indicates a possible clipping of the signal, it calls CLOS1 to compute the first closeness values (discussed below), returns to RECORD, indicating that clipping has occurred, and goes on to smoothing.

KLUDGE determines the A1, Z1, A2, Z2, A3, and Z3 values by calling program CHKMIC which runs under the Stanford AI Spacewar mode. CHKMIC is run every 1/60th of a second and processes a maximum of 3 minimal segments each time it runs. The start of the input is determined by computing the sum

$$A1 + Z1 + A2 + Z2 + A3 + Z3/4$$

for successive minimal segments and comparing it to 10. When the sum is equal to or greater than 10, the input buffer is backed up 7 segments and that is the start of the first Q-matrix minimal segment. When CHKMIC detects a group of more than 32 minimal segments (320 msec) without an individual sum exceeding 9, it stops recording. If the number of acceptable Q-matrix segments is less than or equal to 20, it considers the speech sample too small and goes back to a subroutine RETRY to re-initiate the recording. Otherwise, CHKMIC finds the maximum A1 amplitude after doing the A-D convercion.

RECORD smoothes A1, Z1, A2, Z2, A3, and Z3 as follows: For each of these six variables, three adjacent values (called FIRST, SECOND, and THIRD in the program) are considered. If

$$(SECOND-FIRST)\cdot(THIRD-SECOND) < 0$$

then the second value is smoothed as follows:

$$SECOND = \frac{FIRST+SECOND+THIRD}{3} + .5$$

On the other hand, if

(SECOND-FIRST)·(THIRD-SECOND) ≥0

then no smoothing occurs.

The maximum amplitude for each frequency band is then defined as follows:

MAXA1 = maximum amplitude in range 150 - 900 Hz

MAXA3 = maximum amplitude in range 900 - 2200 Hz

MAXA5 = maximum amplitude in range 2200 - 5000 Hz

Let

M = max {MAXA1, MAXA3, MAXA5}

Depending upon the value of M, one of the actions shown in Table 4 is taken:

Table 4. Modified "KLUDGE" Table

Range of M	Result
$M < 30$	Compute $FMAXA = \max\{MAXA1, MAXA3\}$ and $ATFAC = \frac{1}{2}\,IFIX\left(4 \log \frac{\bar{A} + \frac{3}{2}}{FMAXA}\right) + \frac{1}{2}$, where IFIX(V) = integer part of V, and $\bar{A}$ = average of A = 31. User receives the response "Raise the volume by about ⟨ATFAC⟩."
$30 \leq M < 62$	Normalization by RECORD as follows: $A1 = \frac{A1 \cdot 63}{MAXA1}$, $A2 = \frac{A2 \cdot 63}{MAXA1}$, $A3 = \frac{A3 \cdot 63}{MAXA1}$.
$M \geq 62$	User receives the response "Signal is clipping" and RECORD normalizes the amplitudes as follows: $A1 = \frac{A1 \cdot 63}{MAXA1}$, $A2 = \frac{A2 \cdot 63}{MAXA1}$, $A3 = \frac{A3 \cdot 63}{MAXA1}$.

The old normalizing routine used with MAIN and its accompanying KLUDGE checked the A2 and A3 values to make sure they did not exceed 7 bits (i.e., 127) after normalizing. The new normalization routine used by RECORD and its accompanying KLUDGE does not make this check after normalizing. It appears possible (but highly unlikely) that if any one of the ratios $\frac{A1}{MAXA1}$, $\frac{A2}{MAXA1}$, $\frac{A3}{MAXA1}$ is >2, then the renormalized A2 and A3 values would exceed 7 bits.

3. SEGMENTATION

3.1 THE Q-MATRIX

The resulting digitized data are arranged in an array, called the Q-matrix, comprised of 7 columns and 500 rows. Each row represents a minimal segment (10 msec) of speech data; row i of Q will be referred to as Q(i). A maximum of 5000 msec = 5 sec of speech is thus allowable. Let $Q = (q_{ij})$, where $i = 1, \ldots, 500$ and $j = 1, \ldots, 7$. Then the 7 columns of Q are defined as follows:

$q_{i,1}$ = A1Q(i) = amplitude in the range 150 Hz - 900 Hz for Q(i)
$q_{i,2}$ = Z1Q(i) = zero-crossings in the range 150 Hz - 900 Hz for Q(i)
$q_{i,3}$ = A2Q(i) = amplitude in the range 900 Hz - 2200 Hz for Q(i)
$q_{i,4}$ = Z2Q(i) = zero-crossings in the range 900 Hz - 2200 Hz for Q(i)
$q_{i,5}$ = A3Q(i) = amplitude in the range 2200 Hz - 5000 Hz for Q(i)
$q_{i,6}$ = Z3Q(i) = zero-crossings in the range 2200 Hz - 5000 Hz for Q(i)
$q_{i,7}$ = CLOVAL(i) = closeness value (to be defined below)

The matrix Q thus has the following appearance:

Table 5. The Q-Matrix

Data for Q(1):	A1Q(1)	Z1Q(1)	A2Q(1)	Z2Q(1)	A3Q(1)	Z3Q(1)	CLOVAL(1)
Data for Q(2):	A1Q(2)	Z1Q(2)	A2Q(2)	Z2Q(2)	A3Q(2)	Z3Q(2)	CLOVAL(2)
.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.
Data for Q(500):	A1Q(500)	Z1Q(500)	A2Q(500)	Z2Q(500)	A3Q(500)	Z3Q(500)	CLOVAL(500)

By convention, CLOVAL(1) = CLOVAL(500) = 0.

The first computation of CLOVAL(i) occurs by the main program which calls subroutine CLOS1 (described in Section 3.6.1) before the actual segmentation subroutine is entered. The first computation of CLOVAL(i) is based on computing the closeness between Q(i-1) and Q(i+1) for all segments in the Q-matrix.*

*This appears to be at odds with statements in [1] such as "The Purpose of the primary segmentation procedure is to group together similar adjacent minimal segments which are produced by the preprocessing procedure." However, Reddy [3] (an interesting discussion of closeness and tolerance interval on pages 43-47) has said: "It might so happen that the choice of a minimal segment is such that it falls between two peaks of the speech wave (i.e., between two pitch periods). One way to correct this would be to choose a minimal segment interval so that it will include at least one pitch period whenever voicing is present. However, this will decrease the precision with which segmentation may be achieved. This difficulty may also occur due to the irregularities of the vocal apparatus. It is corrected by using the rule that if two segments are about the same intensity level and have about the same number of zero crossings and if they are sufficiently close (20 ms) to each other, although not adjacent, they can be grouped to form one segment."

3.2 PRIMARY SEGMENTATION

Primary segmentation groups together similar minimal segments (on the basis of the closeness values) and labels them "sustained" or "transitional." Sustained segments consist of a string of minimal adjacent segments having a positive closeness value and include as their first minimal segment the previously adjacent minimal segment with a negative closeness value. If the first Q-matrix minimal segment begins with a positive closeness value (which is almost always the case), the first sustained segment will consist of only adjacent segments with positive closeness values. All other minimal segments not a part of sustained segments become grouped into transitional segments.

The rationale for this method of grouping is the way in which the first closeness value is computed. A negative closeness for Q(i) indicates a lack of closeness between Q(i-1) and Q(i+1). If the negative closeness of Q(i) is followed by a positive closeness for Q(i+1), this indicates a closeness between Q(i) and Q(i+2). The beginning of the sustained segment is taken to be at Q(i). Also, the last Q(i) in a string of adjacent segments with positive closeness values indicates a closeness between Q(i-1) and Q(i+1). The negative closeness value for Q(i+1) indicates a lack of closeness between Q(i) and Q(i+2). The end of the sustained segment is taken to be Q(i). The combining rules are summarized in the following two diagrams:

	Segment #	Closeness Value
transitional segment	.	.
	.	.
	.	.
	i-1	-
sustained segment	i	-
	i+1	+
	i+2	+
	.	.
	.	.
	.	.

	Segment #	Closeness Value
sustained segment	.	.
	.	.
	.	.
	i-1	+
	i	+
transitional segment	i+1	-
	i+2	-
	.	.
	.	.
	.	.

3.3 THE P-MATRIX

The composite segments resulting from the above primary segmentation process lead to the construction of a new array, called the P-matrix, which contains 25 columns and 200 rows. Each row of the P-matrix contains data relating to a segment either minimal or larger; row i of P will be referred to as P(i). An arbitrary row of P will be called a P-segment. Let $P = (p_{ij})$, where $i = 1, \ldots, 200$ and $j = 1, \ldots, 25$. Then the 25 columns of P are described as follows:

$p_{i,1}$ = SBG(i): The variable SBG(i) + 1 points to the beginning minimal segment in the Q-matrix that identifies the start of the larger P-matrix segment.

$p_{i,2} = \left\{ \begin{matrix} \text{SND(i)} \\ \text{TYPE(i)} \end{matrix} \right\}$: This column is used for successive storage of the two variables as shown; initially, $p_{i,2}$ is filled with SND(i) as used in subroutine SEGMENT to point to the ending minimal segment in the Q-matrix that identifies the end of the larger P-matrix segment. Later, SND(i) is replaced by TYPE(i), which is used in the recognition subroutine to define vowel, burst, consonant, stop, etc.

$p_{i,3}$ = DUR(i): Number of minimal segments in the P-matrix segment; and also, because each minimal segment is 10 ms in length, DUR(i) is the time duration of the segment in 10 ms units.

$p_{i,4}$ = AlMN(i): The minimum amplitude in the range 150 Hz - 900 Hz of all minimal segments making up the larger P-matrix segment.

$P_{i,5}$ = A1(i): The average amplitude in the range 150 Hz - 900 Hz of all minimal segments making up the larger P-matrix segment, computed as follows:
Let

J1B = SBG(i) + 1 + BETA,

where BETA = 0 for the initial construction of the P-matrix,* and let

J1E = SND(i) - BETA.

Then

$$A1(i) = \frac{\sum_{j=J1B}^{J1E} A1Q(j) + \frac{J1E - J1B}{2}}{J1E - J1B + 1}.$$

$P_{i,6}$ = A1MX(i): The maximum amplitude in the range 150 Hz - 900 Hz of all minimal segments making up the larger P-matrix segment.

$P_{i,7}$ = Z1MN(i): The minimum zero-crossing count in the range 150 Hz - 900 Hz of all minimal segments making up the larger P-matrix segment.

$P_{i,8}$ = Z1(i): The average zero-crossing count in the range 150 Hz - 900 Hz of all minimal segments making up the larger P-matrix segment, computed as follows:
Let J1B and J1E be defined as above. Then

$$Z1(i) = \frac{\sum_{j=J1B}^{J1E} Z1Q(j) + \frac{J1E - J1B}{2}}{J1E - J1B + 1}.$$

$P_{i,9}$ = Z1MX(i): The maximum zero-crossing count in the range 150 Hz - 900 Hz of all minimal segments making up the larger P-matrix segment.

$P_{i,10}$ = A2MN(i): The minimum amplitude in the range 900 Hz - 2200 Hz of all minimal segments making up the larger P-matrix segment.

*Later modifications of the P-matrix will require other (possibly non-zero) values of BETA, and these calculations will be described in a later section on the CREAT2 and CRFAT4 subroutines.

$P_{i,11}$ = A2(i): The average amplitude in the range 900 Hz - 2200 Hz of all minimal segments making up the larger P-matrix segment; A2(i) is computed in the same manner as A1(i), with A2Q(j) replacing A1Q(j).

$P_{i,12}$ = A2MX(i): The maximum amplitude in the range 900 Hz - 2200 Hz of all minimal segments making up the larger P-matrix segment.

$P_{i,13}$ = Z2MN(i): The minimum zero-crossing count in the range 900 Hz - 2200 Hz of all minimal segments making up the larger P-matrix segment.

$P_{i,14}$ = Z2(i): The average zero-crossing count in the range 900 Hz - 2200 Hz of all minimal segments making up the larger P-matrix segment; Z2(i) is computed in the same manner as Z1(i), with Z2Q(j) replacing Z1Q(j).

$P_{i,15}$ = Z2MX(i): The maximum zero-crossing count in the range 900 Hz - 2200 Hz of all minimal segments making up the larger P-matrix segment.

$P_{i,16}$ = A3MN(i): The minimum amplitude in the range 2200 Hz - 5000 Hz of all minimal segments making up the larger P-matrix segment.

$P_{i,17}$ = A3(i): The average amplitude in the range 2200 Hz - 5000 Hz of all minimal segments making up the larger P-matrix segment; A3(i) is computed in the same manner as A1(i), with A3Q(j) replacing A1Q(j).

$P_{i,18}$ = A3MX(i): The maximum amplitude in the range 2200 Hz - 5000 Hz of all minimal segments making up the larger P-matrix segment.

$P_{i,19}$ = Z3MN(i): The minimum zero-crossing count in the range 2200 Hz - 5000 Hz of all minimal segments making up the larger P-matrix segment.

$P_{i,20}$ = Z3(i): The average zero-crossing count in the range 2200 Hz - 5000 Hz of all minimal segments making up the larger P-matrix segment; Z3(i) is computed in the same manner as Z1(i), with Z3Q(j) replacing Z1Q(j).

$P_{i,21}$ = Z3MX(i): The maximum zero-crossing count in the range 2200 Hz - 5000 Hz of all minimal segments making up the larger P-matrix segment.

$P_{i,22} = \begin{Bmatrix} MK(i) \\ CLO(i) \end{Bmatrix}$: This column is used for successive storage of the two variables as shown; initially, $P_{i,22}$ is filled with MK(i), which is a logical marker in looking for a parameter with large variation within a P-segment. MK(i) = .TRUE. if there is a variable parameter such that one of the Q-segments within the P-segment has a closeness value (CLOVAL) ≤ 7. Later, MK(i) is replaced by CLO(i), a measure of the closeness between P-segments as calculated by function PROXIM (see Section 3.6.2).

$P_{i,23}$ = SXT(i): SXT(i) = 0 if the segment is not a local minimum or maximum as determined by pseudo-subroutine MINMAX, SXT(i) = 1 if the segment is a local maximum, SXT(i) = -1 if the segment is a local minimum.

$P_{i,24}$ = BPT(i): The logical pointer for the physical row P(i).

$P_{i,25}$ = NAT(i): The description of the segment, i.e., SUST (sustained) or TR (transitional).

The initial configuration of the P-matrix is shown in Table 6.

Table 6. Initial Configuration of the P-Matrix

Data for P(1):	SBG(1)	SND(1)	DUR(1)	A1MN(1)	A1(1)	A1MX(1)	Z1MN(1)	Z1(1)	Z1MX(1)	A2MN(1)	A2(1
Data for P(2):	SBG(2)	SND(2)	DUR(2)	A1MN(2)	A1(2)	A1MX(2)	Z1MN(2)	Z1(2)	Z1MX(2)	A2MN(2)	A2(2
.	.	.	.	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.	.	.	.
Data for P(200):	SBG(200)	SND(200)	DUR(200)	A1MN(200)	A1(200)	A1MX(200)	Z1MN(200)	Z1(200)	Z1MX(200)	A2MN(200)	A2(20

(1)	Z2MN(1)	Z2(1)	Z2MX(1)	A3MN(1)	A3(1)	A3MX(1)	Z3MN(1)	Z3(1)	Z3MX(1)	MK(1)	SXT(1)	BPT(1)	NAT(1)
(2)	Z2MN(2)	Z2(2)	Z2MX(2)	A3MN(2)	A3(2)	A3MX(2)	Z3MN(2)	Z3(2)	Z3MX(2)	MK(2)	SXT(2)	BPT(2)	NAT(2)
	.	.	.	.	.	.	.	.	.	.	.	.	.
	.	.	.	.	.	.	.	.	.	.	.	.	.
	.	.	.	.	.	.	.	.	.	.	.	.	.
200)	Z2MN(200)	Z2(200)	Z2MX(200)	A3MN(200)	A3(200)	A3MX(200)	Z3MN(200)	Z3(200)	Z3MX(200)	MK(200)	SXT(200)	BPT(200)	NAT(200)

3.4 SECONDARY SEGMENTATION

Once the P-matrix has been constructed, it is subjected to various modifications. These changes include the suppression of so-called "noisy" segments and a refinement of the segmentation of the speech sample by checking intra-segment variation.

3.4.1 Suppression of Noisy Segments

The secondary segmentation routine suppresses the noisy segments at the beginning and end of the speech utterance. Noisy segments are defined as being those adjacent segments from the beginning minimal segment forward for which A1MX(i) < 8 or Z3MX(i) ≤ 40; or those adjacent segments from the ending minimal segment backwards for which A1MX(i) < 8 or Z3MX(i) ≤ 40.

3.4.2 Intra-Segment Variation

All sustained segments are checked for internal variations and broken down into smaller segments if necessary. This is done by using the function VARFUN (Section 3.7) to check the variation within a row of the P-matrix and to flag the most variable parameter, if any. New closeness indices are then computed for the Q-matrix segments making up the P-matrix segment with an increased weight for the most variable parameter. To accomplish this, function CLOS1 is called to compute the value of CLOVAL(j), which is computed as the closeness value between Q(j-2) and Q(j+1), beginning with j = SBG(i) + 1, and ending with j = SND(i); however, if j-2 < 1, we set j-2 = 1 and if j+1 > SIZEP (the number of segments in the P-matrix), we set j+1 = SIZEP.

If there exists a segment Q(j) within P(i) for j = SBG(i) + 2* to j = SND(i) for which CLOVAL(j) ≤ 7, P(i) is subdivided. Any newly created P-matrix sustained segments are then checked for internal variation again, and the above process is repeated until CLOVAL(j) > 7 for all Q(j) within P(i) from j = SBG(i) + 2 to j = SND(i).

---

*Although the previous calculations treat Q(j) beginning with j = SBG(i) + 1, this set of tests begins with j = SBG(i) + 2.

If, after processing all P(i) from i = 1 to i = SIZEP in the above manner, it has been determined that at least one P(i) has internal variation, the entire P-matrix is recompacted into new P-segments, and the program again starts checking all sustained P-segments for internal parameter variation, etc. The process ends when the entire P-matrix is searched for sustained segments with internal variation and none are found.

The P-matrix is then sorted into sequential order; noisy segments at the beginning and end are suppressed as above; the P-matrix is compacted and pointers are assigned as follows:

INUSE(i): Points to the physical P-matrix row number (see Section 3.9)

BPT(i): Points to the logical P-matrix row number (see Section 3.9)

Finally, subroutine MINMAX is applied to each segment to determine if it is a local minimum or maximum, as defined below.

3.4.3 Pseudo-Subroutine* MINMAX and the Creation of New Sustained Segments

In order to identify P-segments which contain a local maximum or a local minimum, we proceed as follows:

Let j = INUSE(i) as given above, and define

$$SCRAT(i) = 2 \cdot A1(j) + A2(j) + \frac{A3(j) + 1}{2} + DUR(j)$$

for all i beginning with i = 1 and ending with i = SIZEP.

SCRAT(i) can be interpreted as a smoothing process on the values A1(j), A2(j), and A3(j). The weights 2, 1, and $\frac{1}{2}$ can be justified by observing that A1(j) is defined over the frequency range 150 - 900 Hz, A2(j) is defined over 900 - 2200 Hz, and A3(j) over 2200 - 5000 Hz. The lengths of these three intervals are 750, 1300, and 2800, respectively, and are thus in the approximate ratio $\frac{1}{2}$: 1 : 2 with one another. Therefore, in order to weight A1(j), A2(j), and A3(j) fairly, we must multiply A1(j) by 2, A2(j) by 1, and A3(j) by $\frac{1}{2}$. To

*A pseudo-subroutine is an internal subroutine used in a larger program and is entered and exited by assigning return addresses from one routine to the next.

make SCRAT(i) duration-dependent, the term DUR(j) is added. The "+1" appearing in the third term of SCRAT(i) is a rounding factor. Figure 1 illustrates the computation of SCRAT(i), which can be thought of as a crude approximation to the power spectrum of the data for the sustained segment.

Now if

$$SCRAT(i) \leq SCRAT(i-1) - 10 \text{ and } SCRAT(i) \leq SCRAT(i+1) - 10,$$

then segment j = INUSE(i) is said to be a local minimum, and we set

$$SXT(j) = -1.$$

Alternatively, if

$$SCRAT(i) \geq SCRAT(i-1) + 10 \text{ and } SCRAT(i) \geq SCRAT(i+1) + 10,$$

then segment j = INUSE(i) is said to be a local maximum, and we set

$$SXT(j) = 1.$$

If neither set of inequalities is satisfied, it may still be possible to define a local extremum. Suppose, for example, that there exists an integer i such that

$$SCRAT(i) \leq SCRAT(i-1) - 10$$

and

$$|SCRAT(k) - SCRAT(k+1)| < 10$$

for k = i, i+1, ... , i+n-1 for some integer n. If we now have that

$$SCRAT(i+n) \leq SCRAT\ (i+n+1) - 10,$$

then the local minimum is spread over the segments i, i+1, ... , i+n. In this case, the segment of longest duration is said to be the local minimum. An analogous set of tests leads to the development of a local maximum spread over several segments. If no local extremum is found for segment j, we set

$$SXT(j) = 0.$$

The P-matrix segments are examined and any transitional segment containing a local minimum or maximum with a duration = 1 (i.e., a 10 msec minimal segment)

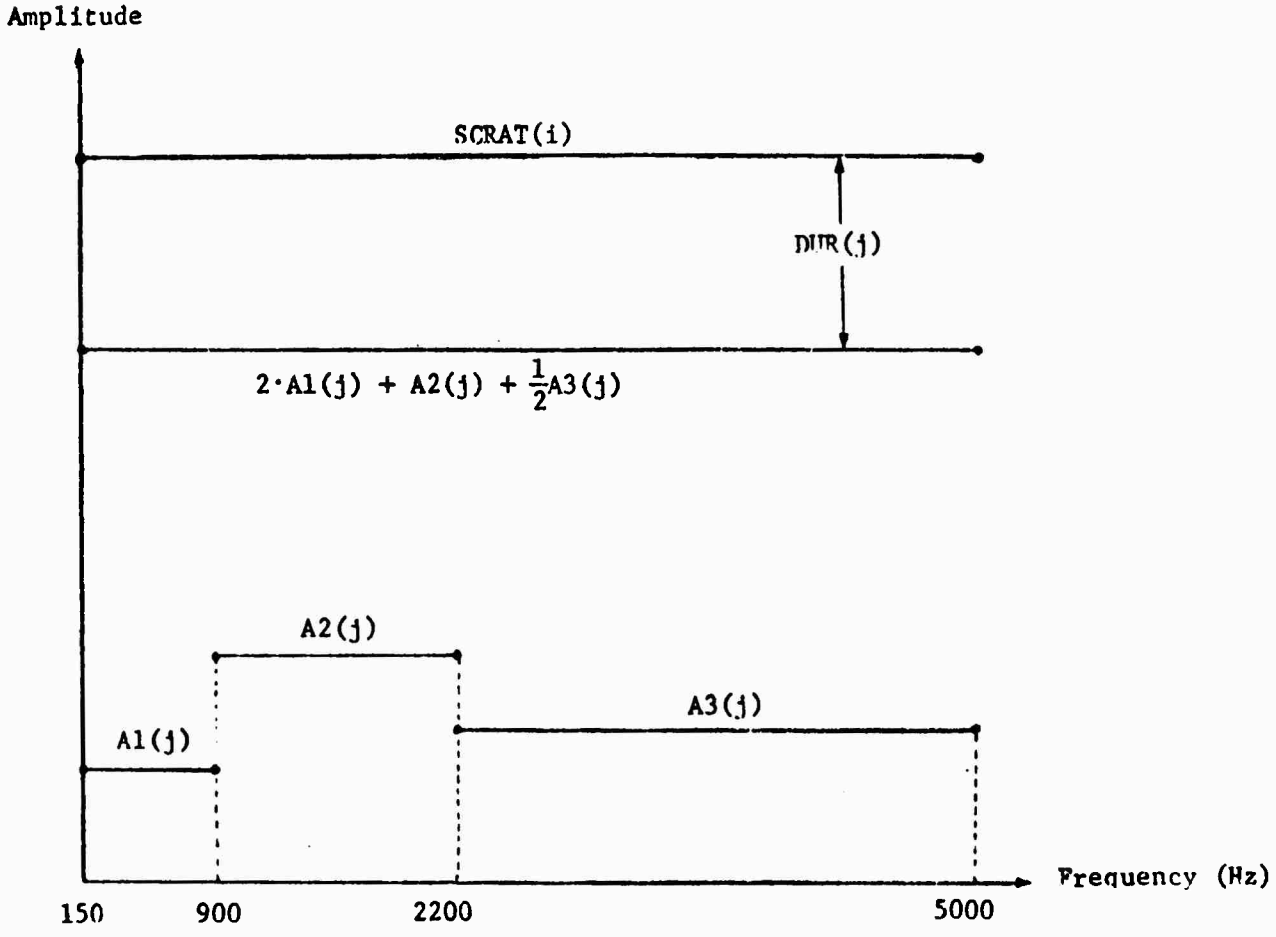


Figure 1. Computation of SCRAT(i)

is renamed "sustained." Any transitional segment containing a local minimum or maximum with a duration greater than 1 (i.e., more than a 10 msec minimal segment) is broken up into more than one segment. If the transitional P-segment has a local maximum, then the Q-segment within the P-segment having the largest amplitude in the frequency range 150 - 900 Hz is labeled "sustained" and flagged as a local maximum. On the other hand, if the transitional segment was a local minimum, then the Q-segment within the P-segment having the smallest amplitude in the frequency range 150 - 900 Hz is labeled "sustained" and flagged as a local minimum. The rest of the transitional segment is then labeled "transitional" with no local extrema. Note that if the minimal segment containing the extremum is bounded by other minimal segments within the larger transitional P-segment, the larger transitional P-segment is broken into three segments: a transitional segment preceding the minimal sustained segment, the minimal sustained segment and a transitional segment following the minimal sustained segment.*

---

*Justification for this process is given by the following ([1], p. 66):

"Local maxima and minima of the amplitude of the waveform are phonemically significant (they usually represent significant vowels and consonants). When a phoneme is articulated for a very short period of time, it has a rapidly varying on-glide and off-glide. When closeness indices are computed for this portion of the sound, one may find that no two adjacent segments satisfy our definition of being close. Thus, they may end up being part of a longer transitional segment. A special effort is made to detect and recover such extrema by searching the transitional segments. In this case, the original transitional segment is replaced by two or more segments, the local extremum being a 10 ms sustained segment."

Consideration is also given to one other type of segment contained within a transitional segment; this is a very short burst segment, characterized by the following three conditions:

(1) Z3Q(i) ≥ 45 or Z3Q(i) + Z2Q(i) ≥ 50,

and

(2) A3Q(i) ≥ A1Q(i),

and

(3) A1Q(i) < 6.

Such segments are made into sustained P-segments. The rest of the P-segment to which they belonged becomes either one or two transitional P-segments, depending on where the short burst occurred within the larger segment: if the short burst occurred after the first minimal segment and before the last, then the short burst will be a sustained segment bounded by transitional segments. This process is repeated until there are no more transitional segments containing a local extremum or a short burst.

Adjacent transitional segments are then grouped together to form one transitional segment; the P-matrix is recompacted; and then for each transitional P-segment of duration ≥ 5, the closeness values (CLOVAL(i)) for its minimal Q-segments are recomputed, where CLOVAL(i) is a measure of the closeness between Q(i-1) and Q(i+1). A loop is then established to additively increase the closeness values of all minimal segments by 1 to a maximum of 6 in order to be able to identify a sustained segment out of this larger transitional segment. If a non-negative closeness value is found, the larger transitional P-segment of duration ≥ 5 is resegmented in order that the minimal segments having non-negative closeness values can be made into a sustained segment.

The P-matrix is then recompacted, resorted, and the local minima and maxima are recomputed. The new P-matrix is then examined for any transitional segments with a local minimum or maximum. If one is found, the program recurses to

examine all transitional segments for local extrema, short bursts, etc. When the P-matrix contains no transitional segments with local extrema, the combining process is begun.

3.5 COMBINING

The purpose of the combining process is to group together acoustically similar P-segments. This task is performed by first treating the transitional segments and then combining sustained segments which are similar in the sense of the definition given below. In general, the transitional segments are considered to be null-segments as defined in [4], pp. 337-342. It is assumed that they do not contain any pertinent information, and a special effort is made to reduce the number of such segments as much as possible. This is done by extending the sustained segments onto the transitional segments if their parameters satisfy the tests given below.

For any given P-segment, say P(i), a set of lower bounds INLIM(n) (n = 1, ... , 6) and upper bounds SUPLIM(n) (n = 1, ... , 6) are computed as follows, using the average amplitudes A1(i), A2(i), and A3(i) and average zero-crossings Z1(i), Z2(i), and Z3(i) of P(i):

n	INLIM(n)	SUPLIM(n)
1	$A1(i) - \left(\frac{A1(i)}{8} + 3\right)$	$A1(i) + \left(\frac{A1(i)}{8} + 3\right)$
2	$Z1(i) - \left(\frac{Z1(i)}{10} + 1\right)$	$Z1(i) + \left(\frac{Z1(i)}{10} + 1\right)$
3	$A2(i) - \left(\frac{A2(i)}{8} + 3\right)$	$A2(i) + \left(\frac{A2(i)}{8} + 3\right)$
4	$Z2(i) - \left(\frac{Z2(i)}{10} + 2\right)$	$Z2(i) + \left(\frac{Z2(i)}{10} + 2\right)$
5	$A3(i) - \left(\frac{A3(i)}{8} + 3\right)$	$A3(i) + \left(\frac{A3(i)}{8} + 3\right)$
6	$Z3(i) - \left(\frac{Z3(i)}{10} + 4\right)$	$Z3(i) + \left(\frac{Z3(i)}{10} + 4\right)$

3.5.1 Extending Sustained Segments onto Adjacent Transitional Segments

For every transitional P-segment (say P(j)), except for the first P-segment, the bounds INLIM(n) and SUPLIM(n) (n = 1, ..., 6) are computed for the previous sustained P-segment (call it P(i)). Beginning with the first minimal Q-segment of P(j) and continuing until the ending minimal Q-segment, the following tests are performed on the parameters A1Q( ), Z1Q( ), A2Q( ), Z2Q( ), A3Q( ), and Z3Q( ):

INLIM(1) ≤ A1Q( ) ≤ SUPLIM(1)
INLIM(2) ≤ Z1Q( ) ≤ SUPLIM(2)
INLIM(3) ≤ A2Q( ) ≤ SUPLIM(3)
INLIM(4) ≤ Z2Q( ) ≤ SUPLIM(4)
INLIM(5) ≤ A3Q( ) ≤ SUPLIM(5)
INLIM(6) ≤ Z3Q( ) ≤ SUPLIM(6)

If more than one parameter is outside the bounds, P(i) is extended to include all Q-segments of P(j) up to the Q-segment containing more than one parameter outside the bounds by using subroutine CREAT4 (described below). P(j) then becomes a transitional segment beginning with the first Q-segment that had more than one parameter not falling within the bounds computed for P(i).

The same procedure is now applied to the sustained P-segment following P(j): for the above transitional P(j), the bounds INLIM(n) and SUPLIM(n) (n = 1, ..., 6) are computed for the following sustained P-segment (call it P(k)). Beginning with the last minimal Q-segment of P(j) and continuing backward to the first minimal Q-segment, the parameters A1Q( ), Z1Q( ), A2Q( ), Z2Q( ), A3Q( ), and Z3Q( ) are tested to see if they lie within the corresponding bounds of P(k). If more than one parameter is outside the bounds, P(k) is extended backwards to include all minimal Q-segments of P(j) back to the Q-segment containing more than one parameter outside the bounds. P(j) then becomes a transitional segment ending with the first Q-segment that has more than one parameter not falling within the bounds computed for P(k).

The P-matrix is then recompacted, sorted, and local minima and maxima for the new segments are identified using pseudo-subroutine MINMAX.

In order to combine P-segments, we determine whether or not two P-segments are similar by using function PROXIM (see Section 3.6.2). The parameters used in the closeness index computation are now the average parameters for the P-segments, viz., A1(i), Z1(i), A2(i), Z2(i), A3(i), and Z3(i).

For every P-segment except:

(1) those containing a local extremum and consisting of one minimal Q-segment

and

(2) those containing no local extremum and consisting of either one, two, three or four minimal Q-segments,

the closeness between P(i) and P(i+1) is computed by function PROXIM. The result is stored in CL(i+1). For cases (1) and (2) above, however, the closeness value is computed between P(i+1) and a "pseudo-segment" based upon P(i). This "pseudo-segment" is created by subroutine CREAT4 and is computed in row number SIZEPM, the last row of the P-matrix used as a "scratch" area. CREAT4 is called after setting

SEND = SND(i), SBEG = SND(i) - 5 and LENSEG = 5.

CREAT4 then computes

$$\text{BETA} = \frac{\text{LENSEG} - 2}{3}$$

which in this case is = 1, and then calculates the following parameters for row m = SIZEPM:

A1MN(m) , A1(m) , A1MX(m)
Z1MN(m) , Z1(m) , Z1MX(m)
A2MN(m) , A2(m) , A2MX(m)
Z2MN(m) , Z2(m) , Z2MX(m)
A3MN(m) , Z3(m) , A3MX(m)
Z3MN(m) , Z3(m) , Z3MX(m)

In the case in which the P-segment consists of one minimal Q-segment, the above parameters are computed from the three minimal Q-segments immediately preceding the P-segment. If the P-segment consists of two Q-segments, the parameters are computed from the first Q-segment of the P-segment plus the preceding two Q-segments. For the case of three Q-segments, the calculations are based upon the first two Q-segments of the P-segment and the preceding Q-segment. Finally, for four Q-segments, the first three Q-segments of the P-segment are used.

After CLO(i+1) is computed, a check is made to see if the duration of the combination of P(i) and P(i+1) is greater than 300 msec, and if so, we set CLO(i+1) = - 30. Otherwise, CLO(i+1) is left equal to its previously computed value.

3.5.2 Combining Segments

Let i = 2. An attempt is now made to combine P(i) with the P-segment immediately preceding it, which might not be P(i-1) if P(i-1) has previously been combined with some other P-segment. When we have found this immediately preceding P-segment, we let I1 = the P-matrix row number of this segment. Let I2 = i and I3 = i+1. If i = SIZEP, let I3 = 0.

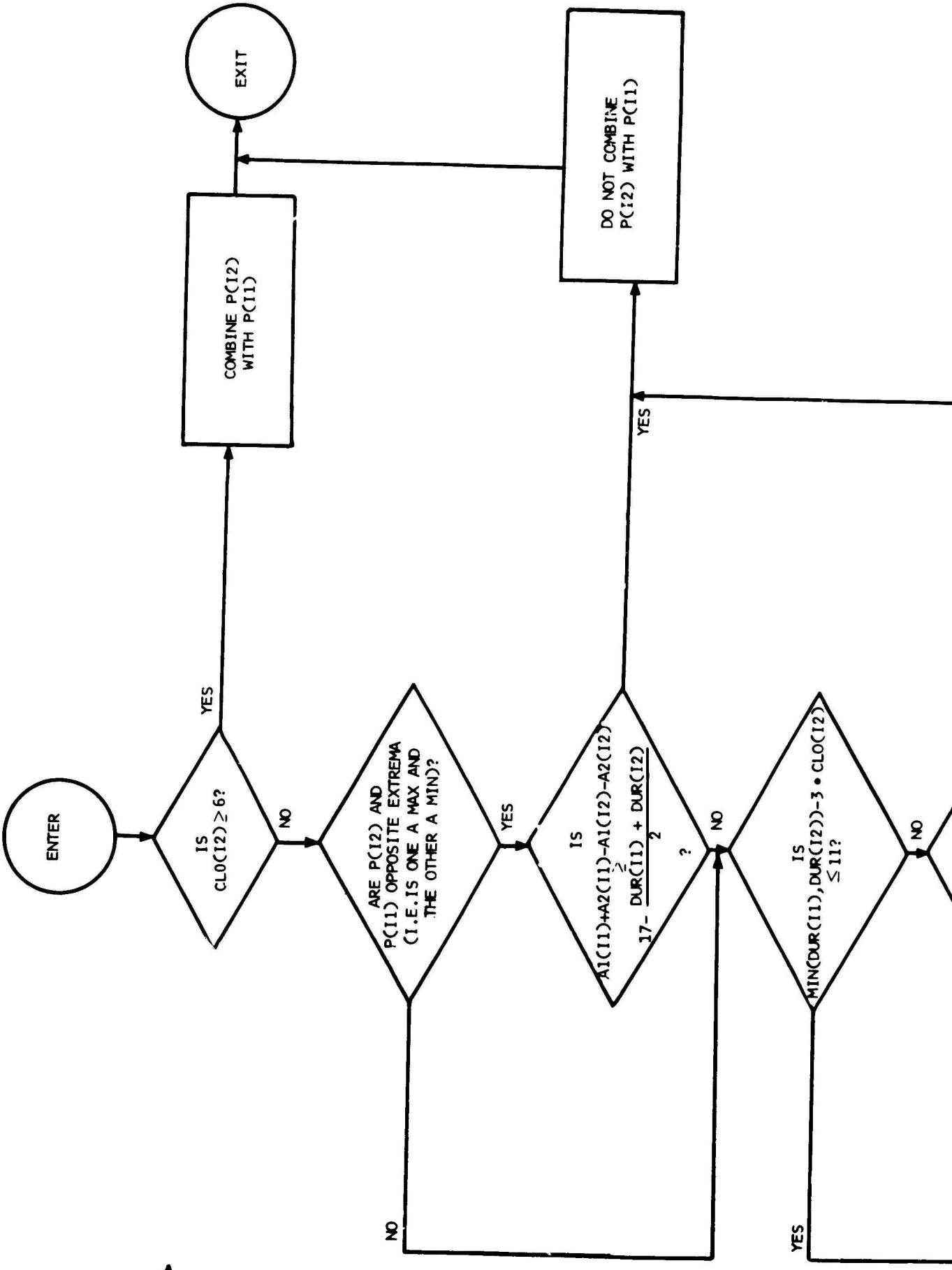
Because of the complexity involved in the combining process, the heuristics used to decide whether to combine P(I2) with P(I1) are illustrated by the flow chart in Figure 2. It is important to be able to interpret these heuristics in light of various mathematical techniques.

Consider, for example, the inequality

$$\left|A1(I1) + A2(I1) - A1(I2) - A2(I2)\right| \geq 17 - \frac{DUR(I1) + DUR(I2)}{2}$$

Rewrite this in the form

$$\left| \frac{A1(I1) - A1(I2)}{\frac{1}{2}(DUR(I1) + DUR(I2))} + \frac{A2(I1) - A2(I2)}{\frac{1}{2}(DUR(I1) + DUR(I2))} \right| \geq \frac{1}{4}\left\{ \frac{34}{\frac{1}{2}(DUR(I1) + DUR(I2))} + \frac{34}{\frac{1}{2}(DUR(I1) + DUR(I2))} \right\} - 1$$


ENTER
IS CLO(I2) ≥ 6?
YES
NO
ARE P(I2) AND P(I1) OPPOSITE EXTREMA (I.E. IS ONE A MAX AND THE OTHER A MIN)?
YES
NO
IS A1(I1)+A2(I1)-A1(I2)-A2(I2) ≥ 17 - DUR(I1) + DUR(I2) / 2 ?
YES
NO
IS MIN(DUR(I1), DUR(I2))-3 • CLO(I2) ≤ 11?
YES
NO
COMBINE P(I2) WITH P(I1)
DO NOT COMBINE P(I2) WITH P(I1)
EXIT

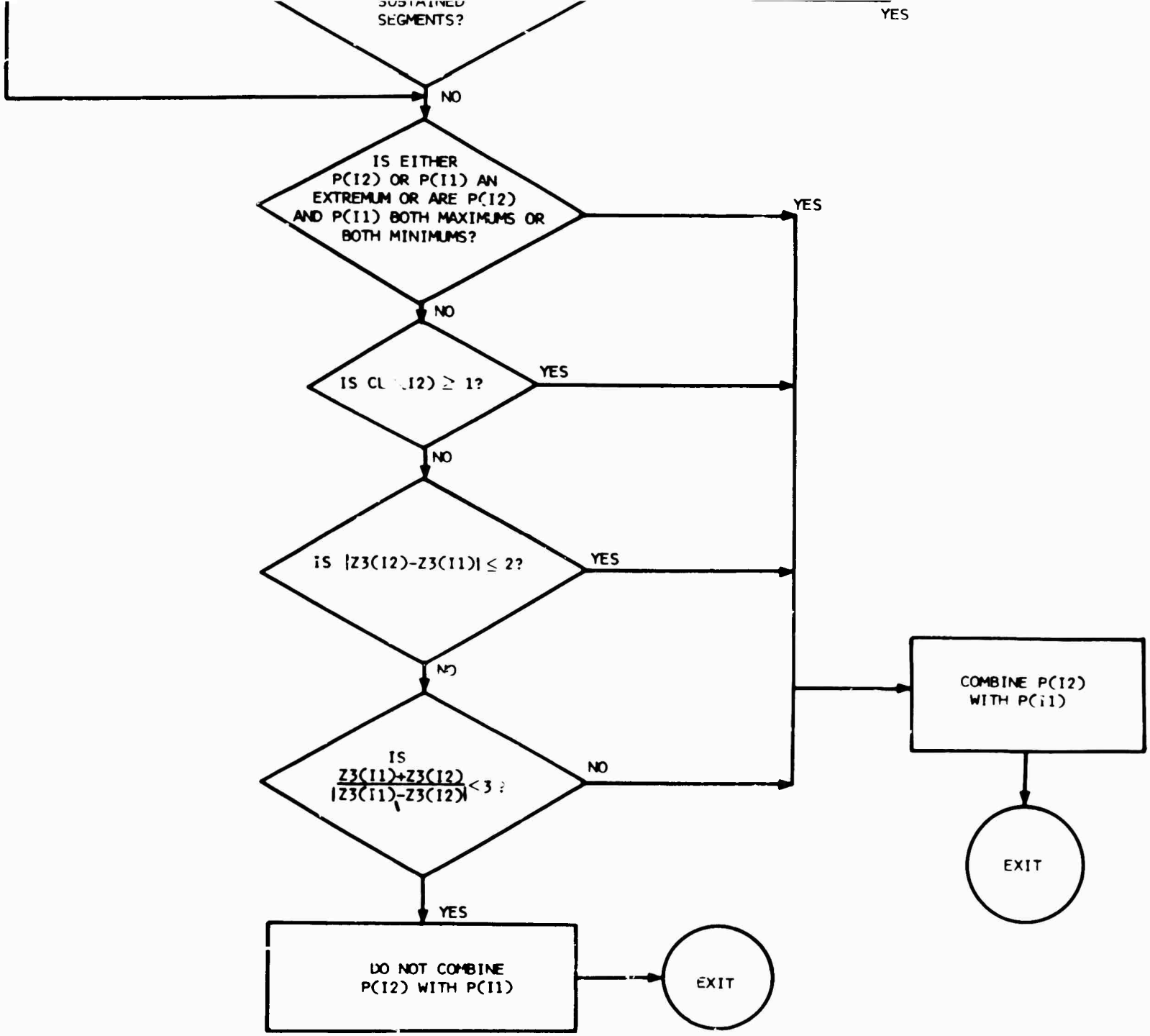


Figure 2. Flow Chart of the Refinement Subroutine

The term

$$\frac{A1(I1) - A1(I2)}{\frac{1}{2}(DUR(I1) - DUR(I2))}$$

measures the rate of change of A1(i) as we proceed from segment P(I1) to segment P(I2). A similar statement applies to the term

$$\frac{A2(I1) - A2(I2)}{\frac{1}{2}(DUR(I1) + DUR(I2))}.$$

To identify the significance of the right-hand side of the inequality, we proceed as follows: consider A1(i) and A2(i) as discrete random variables selected from a uniform distribution. Each is assumed to lie in the range from 4 to 63 inclusive, since this is the range used in CLOS1. The expected value of either A1(i) or A2(i) in this range is then

$$\frac{4 + \cdots + 63}{63 - 4 + 1} = 33.5 = 34 \text{ (rounded)}.$$

Each term

$$\frac{34}{\frac{1}{2}(DUR(I1) + DUR(I2))}$$

thus represents an average rate of change that can be expected in either A1(i) or A2(i). The inequality can now be interpreted as follows: if P(I1) and P(I2) are opposite extrema and if the sum of the rate of change in A1(i) and that of A2(i) is greater than or equal to 25% of the sum of the average rates of change that can be expected in A1(i) and A2(i), then P(I1) and P(I2) are not combined.

If P(I1) is not combined with P(I2), then the processing of the P-segments continues until P(SIZEP) has been tested.

If P(I2) is to be combined with P(I1), then pseudo-subroutine CREAT4 is used to construct a new P(I1) where the beginning of P(I1) is defined by SBG(I1) + 1 and the end by SND(I2). The LENSEG (length of P(I1)) is given by DUR(I1) + DUR(I2) and TYPE (I1) = SUST unless both P(I1) and P(I2) were "TR," in which case TYPE(I1) = TR.

I2 is then made available for reassignment as a row number by being released from the INUSE table. The P-matrix is recompacted. The P-segment immediately preceding P(I1) will be called P(I0). P(I0) is examined and if P(I0) is not a local minimum or maximum and DUR(I0) is < 5 or if P(I0) is a local minimum or maximum and DUR(I0) = 1, then a pseudo-segment is created by CREAT4 in P(200) such that LENSEG = 5, SEND = SND(I0) and SBEG = SEND -5. A new closeness value, CLO(I1), is then computed between P(I0) (or P(200)) and P(I1). If the duration of P(I0) plus that of P(I1) is greater than 300 msec then CLO(I1) = -30. The same procedure is followed for P(I1) and P(I3). The pseudo-subroutine MINMAX is called to find the new extrema, i is reset equal to 2, and the combining process starts all over again.

3.5.3 The Creation of Beginning and Ending Segments

Recall that earlier the "noisy" segments at the beginning and end of the sample were suppressed. An attempt is now made to create a beginning segment and an ending segment based upon the suppressed data still resident in the O-matrix. Again, because of the complexity of this process, the logic is depicted in the flow chart of Figure 3.

3.5.4 Further Suppression of Transitional Segments

For all transitional P(i) for i = 1, ..., SIZEP, we calculate:

$$I1 = INUSE(i-1)$$

$$I2 = INUSE(i)$$

$$I3 = INUSE(i+1)$$

$$INUSE(i) = 0$$

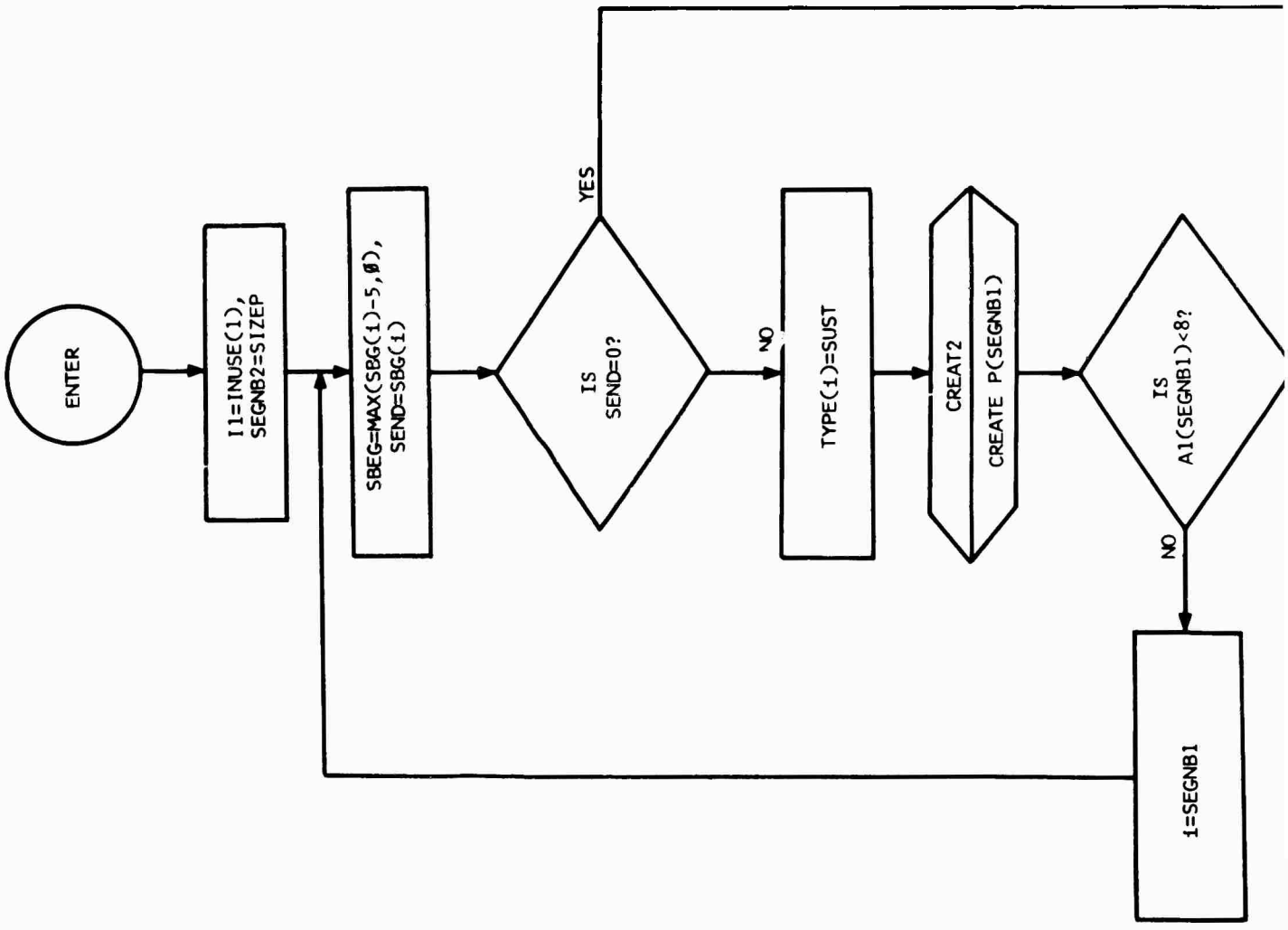
$$DUR(I1) = DUR(I1) + \frac{DUR(I2)}{2}$$

$$DUR(I3) = DUR(I3) + \frac{DUR(I2)}{2}$$

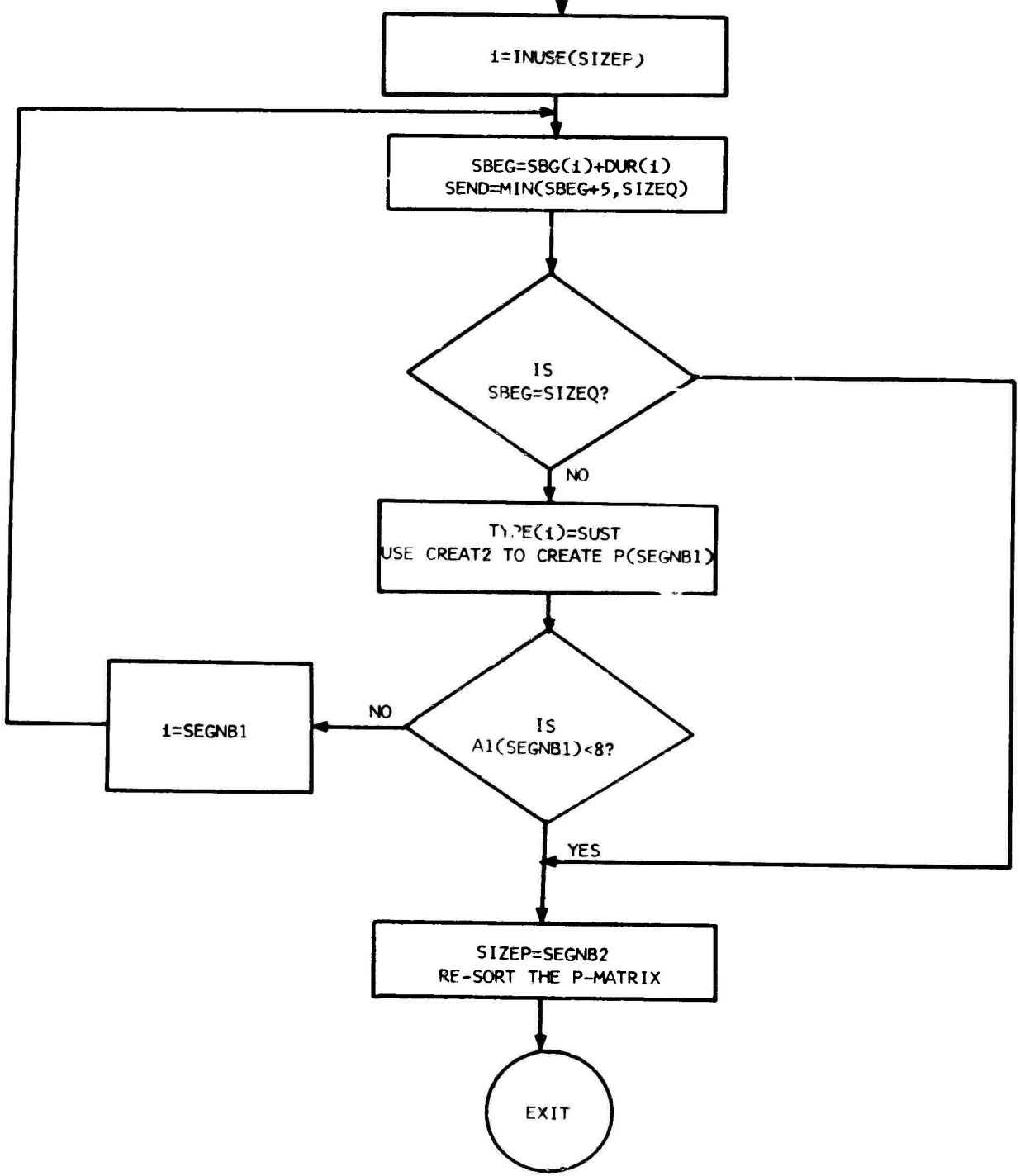
$$SBG(I3) = SBG(I3) - \frac{DUR(I2)}{2}$$

$$SND(I1) = SND(I1) + \frac{DUR(I2)}{2}$$

(Page 32 blank)



A


i=INUSE(SIZEP)
SBEG=SBG(i)+DUR(i)
SEND=MIN(SBEG+5,SIZEQ)
IS
SBEG=SIZEQ?
NO
TYPE(i)=SUST
USE CREAT2 TO CREATE P(SEGNB1)
IS
A1(SEGNB1)<8?
NO
i=SEGNB1
YES
SIZEP=SEGNB2
RE-SORT THE P-MATRIX
EXIT

This results in extending each transitional segment halfway onto the preceding and following sustained segments. Of course, if a transitional segment happens to consist of an odd number of minimal segments, the fate of the minimal segment in the center is questionable.*

The P-matrix is then compacted, and an entirely new set of transitional segments is artificially created as follows: for all values of i between 2 and SIZEP - 1, we isolate all P(i) for which either (1) DUR(i) ≤ 3 or (2) P(i) is "SUST." For each i, a variable TRINDX (which will represent a count of parameters with transitional characteristics) is initialized to zero. For j = 5, 8, 11, 14, 17, and 20 (the column numbers of the average parameter values A1, Z1, A2, Z2, A3, and Z3 of the P-matrix), we proceed as follows.

If

$$| P(I2,j) - P(I1,j) | < \max\left(1, \frac{P(I2,j) + P(I1,j)}{40}\right)$$

then we set TEMP1 = 0. If, however,

$$| P(I2,j) - P(I1,j) | \geq \max\left(1, \frac{P(I2,j) + P(I1,j)}{40}\right)$$

then we set

$$TEMP1 = P(I2,j) - P(I1,j).$$

Similarly, if

$$| P(I2,j) - P(I3,j) | < \max\left(1, \frac{P(I2,j) + P(I3,j)}{40}\right)$$

then we set TEMP2 = 0. If, however,

$$| P(I2,j) - P(I3,j) | \geq \max\left(1, \frac{P(I2,j) + P(I3,j)}{40}\right)$$

then we set

$$TEMP2 = P(I2,j) - P(I3,j).$$

*In the SDC version of the program, if a transitional segment is composed of an odd number n of minimal segments, then $\frac{n-1}{2}$ segments are extended onto the preceding sustained segments, and $\frac{n+1}{2}$ segments are extended onto the following sustained segment.

Now if

TEMP1 · TEMP2 ≤ 0

then we put*

TRINDX = TRINDX +1

At the completion of the j-loop above, TRINDX will have a value between 0 and 6. P(I2) will be labeled TR if either of the following two tests is satisfied:

(1) If P(I2) is not an extremum and 2 · DUR(I2) - TRINDX ≤ 2,

(2) If P(I2) is an extremum and DUR(I2) < 3 and TRINDX ≥ 5.

We now recompute closeness values for i = 2, ···, SIZEP so that

I1 = INUSE(i-1)

I2 = INUSE(i)

and CLO(I2) = closeness between P(I1) and P(I2).

If the last two logical P-segments have a duration greater than 300 msec,we put

CLO(SIZEP) = -30.

For each transitional P(i) for i = 2, ···, SIZEP, we set

I1 = INUSE(i-1)

I2 = INUSE(i)

I3 = INUSE(i+1)

If i = SIZEP, then put I3 = 0. If

(a) CLO(I1) < CLO(I2) and CLO(I2) ≥ - 8

and if either P(I3) is sustained or if it is transitional with I1 = 0, or if:

(b) - 8 ≤ CLO(I3) ≤ CLO(I2)

*If the slope from P(I1) to P(I2) is opposite the slope from P(I2) to P(I3), we say P(I2) has transitional characteristics, and TRINDX is incremented by 1.

and if P(I1) is transitional and P(I3) is sustained, then we set INUSE(I2) = 0 and calculate the following new average parameters:

$$A1(I3) = \frac{A1(I2)\cdot DUR(I2) + A1(I3)\cdot DUR(I3)}{DUR(I2) + DUR(I3)}$$

$$Z1(I3) = \frac{Z1(I2)\cdot DUR(I2) + Z1(I3)\cdot DUR(I3)}{DUR(I2) + DUR(I3)}$$

where A2(I3) and A3(I3) are calculated exactly as A1(I3), but with A2( ) replacing A1( ), etc., and Z2(I3) and Z3(I3) are calculated similarly. These new average values are the old values appropriately weighted to reflect their durations in the P-segments.

Also calculate:

$$DUR(I3) = DUR(I2) + DUR(I3)$$

and

$$SBG(I3) = SBG(I3) - DUR(I2).$$

A new closeness value CLO(I3) is now computed between P(I1) and P(I3), since P(I2) has been combined with P(I3). However, if

$$DUR(I1) + DUR(I3) > 30,$$

then we set CLO(I3) = - 30.

If neither (a) nor (b) above is satisfied, but:

(c) $CLO(I3) \leq CLO(I2)$ and $CLO(I2) \geq -8$,

and if P(I1) is a sustained segment, then we set INUSE(I2) = 0 and calculate values of A1(I1), Z1(I1), A2(I1), Z2(I1), A3(I1), and Z3(I1) in the same manner as above but with I1 replacing I3. Also, we compute

$$DUR(I1) = DUR(I1) + DUR(I2)$$

and a new closeness value CLO(I3) between P(I1) and P(I3). However, if

$$DUR(I1) + DUR(I3) > 30,$$

then we set CLO(I3) = - 30. If neither (a) nor (b) nor (c) is satisfied, we do not combine P(I2) with either P(I1) or P(I3). If any P-segment has been combined, then the P-matrix is recompacted and we continue the above procedure until no

more P-segments can be combined. New relative extrema are computed using pseudo-subroutine MINMAX; pseudo-subroutine REORD (Section 3.8.5) is used to reorder the P-matrix, SIZEP = SIZEP +1 and segmentation is complete.

3.6 CLOSENESS FUNCTIONS

3.6.1 Function CLOS1

Function CLOS1 is a basic routine used to compute closeness values (defined below) between rows of the Q-matrix. The input parameters to CLOS1 are:

SEGNB1 = a row number of one particular row of the Q-matrix

SEGNB2 = a row number of another row of the Q-matrix to which row SEGNB1 is to be compared

ALFA = a "flag" of the most variable parameter as determined by function VARFUN; ALFA = 1 if A1Q( ) is most variable
ALFA = 2 if Z1Q( ) is most variable
ALFA = 3 if A2Q( ) is most variable
ALFA = 4 if Z2Q( ) is most variable
ALFA = 5 if A3Q( ) is most variable
ALFA = 6 if Z3Q( ) is most variable
ALFA = 0 if none of A1Q( ), Z1Q( ), A2Q( ), Z2Q( ), A3Q( ), or Z3Q( ) is most variable.

TYPC = 1 if CLOS1 is to compute a closeness value between rows SEGNB1 and SEGNB2, and
= 2 if CLOS1 is to compute closeness values for all rows in the Q-matrix beginning with SEGNB1 and ending with SIZEQ. CLOS1 computes a measure of closeness between row i-1 and row i+1 and stores the result in row i.

In the following discussion, the characters "S1" and "S2" will be used in place of "SEGNB1" and "SEGNB2."

CLOS1 begins by isolating all segments which characterize a fricative type "S." These segments are defined by either one of the following two properties:

(1) A3Q(S1) ≥ A1Q(S1)
and
A3Q(S2) ≥ A1Q(S2)
and
Z3Q(S1) ≥ 60
and
Z3Q(S2) ≥ 60

or

(2) Z3Q(S1) ≥ 45
and
Z3Q(S2) ≥ 45
and
A1Q(S1) ≥ 6
and
A1Q(S2) ≥ 6.

In the previous version of the program (containing MAIN), the fricative test (2) above contained the inequalities

A1Q(S1) ≤ 6 and A1Q(S2) < 6,

opposite to those above. The latter two inequalities appear to give a more reasonable characterization of a fricative type "S," since such a fricative is typically of low A1 amplitude and high Z3 frequency. The reason for the change is unknown.

If a fricative type "S" has been detected, we set CLOS1 = 8 and return to the segmentation algorithm. Otherwise, we proceed by establishing the following table of parameters.

Table 7. Parameters for CLOS1

Parameter	Parameter Number (j)	LIM(j)	TEMLIM(j)	RATLIM(j)	WEIGHT(j)
A1Q( )	1	4	0	0.6	15.0
Z1Q( )	2	2	2	0.3	30.0
A2Q( )	3	4	0	0.6	15.0
Z2Q( )	4	4	14	0.3	30.0
A3Q( )	5	4	0	0.6	15.0
Z3Q( )	6	10	30	0.5	30.0

CLOS1 is initialized to zero, a logical flag (called PRFLAG) is initially set to FALSE, indicating that the segments are similar. Let j = 1 and compute

(1) TEMP1 = max {A1Q(S1), LIM(j)},

and

(2) TEMP2 = max {A1Q(S2), LIM(j)}.

If ALFA = j, then FACT = 1.25; otherwise, FACT = 1.0. Now if

$$| \text{TEMP1} - \text{TEMP2} | \leq \text{LIM}(j),$$

we put

$$\text{CLOS1} = \text{CLOS1} + 2.0.$$

If, however,

$$| \text{TEMP1} - \text{TEMP2} | > \text{LIM}(j),$$

then we compute

(3) $$\text{RATIO} = \frac{| \text{TEMP1} - \text{TEMP2} |}{4 \cdot \sqrt{\text{TEMP1} + \text{TEMP2}}}.$$

If

$$\max \{TEMP1, TEMP2\} \leq TEMLIM(j),$$

we set

$$RATIO = (.7) \cdot RATIO;$$

otherwise leave RATIO alone. If

$$RATIO > RATLIM(j),$$

set PRFLAG = TRUE (meaning that the segments are nonsimilar). In either event, whether

$$RATIO > RATLIM(j) \text{ or } RATIO \leq RATLIM(j),$$

we compute

$$CLOS1 = CLOS1 + \left(\frac{5}{2} - FACT \cdot WEIGHT(j) \cdot RATIO \right).$$

Now step j = j + 1, and return to equations (1) and (2) above with Z1Q(S1) replacing A1Q(S1) and Z1Q(S2) replacing A1Q(S2). CLOS1 is again updated, j = j + 1 again, and we continue with A2Q(S1) and A2Q(S2), etc., until we reach j > 6.

If PRFLAG indicates that the segments are nonsimilar, and if

$$CLOS1 > -4,$$

we set

$$CLOS1 = -4.$$

Finally, two segments are considered "close" if the final value of CLOS1 is > 0.

No interpretation currently exists for either the choice of parameter values or the particular equations used in this routine. In an attempt to provide an understanding, we offer the following:

To begin with, consider the column of values RATLIM(j) for j = 1,...,6. These values serve as threshold parameters for the function

$$RATIO = \frac{| TEMP1 - TEMP2 |}{4 \cdot \sqrt{TEMP1 + TEMP2}}.$$

It can be shown that the values of RATLIM(j) are the averages of RATIO when TEMP1 and TEMP2 are varied over the six ranges shown in Table 8.

Table 8. Comparison of Averages of RATIO and Values of RATLIM(j)

Parameter	Range of TEMP1	Range of TEMP2	Average of RATIO	RATLIM(j)
A1Q( )	4-63	4-63	.63	.6
Z1Q( )	3-18	3-18	.30	.3
A2Q( )	4-63	4-63	.63	.6
Z2Q( )	18-44	18-44	.29	.3
A3Q( )	4-63	4-63	.63	.6
Z3Q( )	44-100	44-100	.40	.5

Thus, for each of the six variables, RATIO compared with its average value.

The function used to generate the average for A1Q( ), A2Q( ) and A3Q( ) is

$$\frac{1}{(63-4+1)^2} \sum_{TEMP1=4}^{63} \sum_{TEMP2=4}^{63} \frac{|\,TEMP1 - TEMP2\,|}{4 \cdot \sqrt{TEMP1 + TEMP2}} .$$

The averages for Z1Q( ), Z2Q( ) and Z2Q( ) are generated similarly. Note that the ranges

3-18, 18-44, 44-100

are the frequency ranges of the three front-end hardware filters.

The first five values of the average of RATIO agree well with RATLIM(j); however, we have disagreement with RATLIM(6). We assert that the correct value for RATLIM(6) is 0.4 and that certain values of WEIGHT(j) are incorrect as given. To show this, we note that the expression for RATIO appears in [1] and in one version of the program as

$$(4) \quad RATIO = \frac{|\,TEMP1 - TEMP2\,|}{\sqrt{TEMP1 + TEMP2}} .$$

The given associated values for RATLIM(j), WEIGHT(j) and the computed averages of RATIO are:

Table 9. Data for Modified RATIO Function

j	RATLIM(j)	WEIGHT(j)	Average of RATIO	
			(computed)	(rounded)
1	2.5	4.0	2.53	2.5
2	1.2	7.5	1.19	1.2
3	2.5	4.0	2.53	2.5
4	1.2	7.5	1.15	1.2
5	2.5	4.0	2.53	2.5
6	2.0	7.5	1.59	1.6

Now, to calculate CLOS1, we must compute the expression

$$\frac{5}{2} - \text{FACT} \cdot \text{WEIGHT}(j) \cdot \text{RATIO}$$

alternately for amplitudes and zero-crossings and add the results together. The numbers WEIGHT(j) ensure that the amplitudes and zero-crossings are weighted fairly; they may be justified by observing that the sum of the average values for the amplitudes is

$$2.5 + 2.5 + 2.5 = 7.5,$$

and for zero-crossings,

$$1.2 + 1.2 + 1.6 = 4.0.$$

The assignment of proper weights is now obvious. The appropriate value of RATLIM(6) for RATIO in (4) is

$$\text{RATLIM}(6) = 1.6.$$

A corrected set of values of RATLIM(j) and WEIGHT(j) to be used in conjunction with RATIO in (3) can now be given as follows:

Table 10. Corrected Values of RATLIM(j) and WEIGHT(j)

j	RATLIM(j)	WEIGHT(j)
1	.625	16.0
2	.300	30.0
3	.625	16.0
4	.300	30.0
5	.625	16.0
6	.400	30.0

There is a further justification for the selection of RATLIM(6) = .4. If we compute the percentage of cases in which

RATIO ≤ RATLIM(j),

and thus call the segments close, we get the following table:

Table 11. Analysis of RATLIM(j) Values

Range of TEMP1	Range of TEMP2	RATLIM(j)	%
4-63	4-63	.625	54
3-18	3-18	.300	55
4-63	4-63	.625	54
18-44	18-44	.300	57
4-63	4-63	.625	54
44-100	44-100	.400	55
44-100	44-100	.500	66

In other words, using RATLIM(6) = .5 as given results in 66% of the values of RATIO being less than or equal to RATLIM(6). The more reasonable value appears again to be RATLIM(6) = .4.

We will now provide justification for the use of the formula

$$\text{CLOS1} = \text{CLOS1} + \left(\frac{5}{2} - \text{FACT}\cdot\text{WEIGHT(j)}\cdot\text{RATIO}\right).$$

For each individual parameter (A1Q( ), Z1Q( ), etc.), we calculate

(5) $$\frac{5}{2} - \text{FACT}\cdot\text{WEIGHT(j)}\cdot\text{RATIO}.$$

For amplitudes, WEIGHT(j) = 16.0, and so the expression in (5) (assuming FACT = 1.0) is > 0 if

$$\text{RATIO} < \frac{5}{2\cdot\text{WEIGHT(j)}} = \frac{5}{32} = (.25)(.625).$$

In order for corresponding amplitudes for two segments to be considered close, RATIO must therefore be within 25% of RATLIM(j).

For zero-crossings, WEIGHT(j) = 30.0. The average value of RATLIM(j) for all three zero-crossing ranges is

$$\frac{.300 + .300 + .400}{3} = \frac{1}{3}.$$

The expression in (5) (assuming FACT = 1.0) is > 0 if

$$\text{RATIO} < \frac{5}{2\cdot\text{WEIGHT(j)}} = \frac{5}{60} = (.25)(.333\cdots).$$

In order for corresponding zero-crossings for two segments to be considered close, RATIO must therefore be within 25% of the average value of RATLIM(j) for all three zero-crossing ranges.

3.6.2 Function PROXIM

Function PROXIM computes closeness values between segments of the P-matrix in a similar manner as CLOS1 calculates closeness values between segments of the Q-matrix. The input parameters to PROXIM are:

SEGNB1 = a row number of one particular row of the P-matrix,

SEGNB2 = a row number of another row of the P-matrix to which row SEGNB1 is to be compared,

ALFA = a 'flag" of the most variable parameter as determined by function VARFUN; ALFA = 1 if A1( ) is most variable
= 2 if Z1( ) is most variable
= 3 if A2( ) is most variable
= 4 if Z2( ) is most variable
= 5 if A3( ) is most variable
= 6 if Z3( ) is most variable
= 0 if none of A1( ), Z1( ), A2( ), Z2( ), A3( ) or Z3( ) is most variable.

PROXIM performs exactly the same operations as CLOS1, but bases the computations on the average parameters A1( ), Z1( ), A2( ), Z2( ), A3( ) and Z3( ), and uses a different set of weights WEIGHT(j) and a different set of values of RATLIM(j). Some confusion exists as to the correct values for WEIGHT(j) and RATLIM(j). At least three different sets of values are in existence: one set is given in [1], another appears in an early version of the program (that in which MAIN is used for recording), and a third is given in a later version (that in which RECORD replaces MAIN). Table 12 lists these three sets.

Table 12. Comparison of Various WEIGHT(j) and RATLIM(j) Values for PROXIM

[1]		Version with MAIN		Version with RECORD	
WEIGHT(j)	RATLIM(j)	WEIGHT(j)	RATLIM(j)	WEIGHT(j)	RATLIM(j)
4.0	2.0	4.0	2.0	15.0	.5
6.0	1.0	6.0	1.0	25.0	.25
4.0	2.0	4.0	2.5	15.0	.5
5.0	1.0	6.0	4.0	20.0	.25
4.0	2.0	4.0	2.5	15.0	.5
5.0	1.6	6.0	1.6	20.0	.4

The RATIO function used in the version of the program with MAIN is*

$$\text{RATIO} = \frac{|\text{TEMP1} - \text{TEMP2}|}{\sqrt{\text{TEMP1} + \text{TEMP2}}},$$

and in the version with RECORD,

$$\text{RATIO} = \frac{|\text{TEMP1} - \text{TEMP2}|}{4\sqrt{\text{TEMP1} + \text{TEMP2}}}.$$

The SDC version of the segmentation program was checked out by taking Q-matrices which were generated by the Stanford program and comparing the P-matrices created by the two programs. The results agreed with the Stanford version with MAIN, but the results did not agree with the version with RECORD. The sets of values of WEIGHT(j) and RATLIM(j) were then modified by multiplying the values of WEIGHT(j) in the MAIN version by 4.0 and dividing the related values of RATLIM(j) by 4.0. The SDC version with these modified values did check out with the Stanford RECORD version. These values are:

WEIGHT(j)	RATLIM(j)
16.0	.5
24.0	.25
16.0	.6
24.0	.25
16.0	.6
24.0	.4

Note that the RECORD values for RATLIM(j) are 1/4 of the RATLIM(j) given by [1], rather than 1/4 of the MAIN RATLIM(j) values. The RECORD values for WEIGHT(j) appear to have been generated in a similar manner by multiplying WEIGHT(j) in [1] by 4.0, but there is not an exact correlation. Since the SDC version with the modified values agrees with the Stanford RECORD version,

*TEMP1 and TEMP2 are defined above in the section on CLOS1 and have the same meaning in PROXIM.

we assume that the values of WEIGHT(j) and RATLIM(j) in the Stanford RECORD version were modified before test results were generated for SDC at Stanford.

However, based upon the previous analysis of the values of WEIGHT(j) and RATLIM(j) in function CLOS1, it is felt that none of the previous sets of WEIGHT(j) and RATLIM(j) for PROXIM are correct. In fact, it appears that the RATLIM(j) values for PROXIM were originally intended to be 80% of those for CLOS1. Based upon this assumption, we assert that the correct set of RATLIM(j) and WEIGHT(j) is:

RATLIM(j)	WEIGHT(j)
.50	12.8
.24	24.0
.50	12.8
.24	24.0
.50	12.8
.32	24.0

The use of this last set of values in PROXIM results in a relaxation of the closeness criteria for P-segments as originally intended in [1]. Indeed, in [1], p. 67, we are told "... as we are dealing with average parameters (i.e., comparing corresponding values of A1( ), Z1( ), A2( ), Z2( ), A3( ) and Z3( ) of two P-segments), the weights are decreased to make the procedure less sensitive to smaller variations." It is instructive to quantify this last statement and see how much less sensitive the procedure becomes with the decreased weights. For corresponding average amplitudes of two P-segments to be considered close, we require that

$$\frac{5}{2} - \text{FACT} \cdot \text{WEIGHT(j)} \cdot \text{RATIO} > 0.$$

Take FACT = 1.0. Then since WEIGHT(j) = 12.8 for amplitudes, the requirement becomes

$$\text{RATIO} < \frac{5}{2 \cdot \text{WEIGHT(j)}} = \frac{5}{25.6} = (.39)(.50),$$

and RATIO must therefore be within 39% of RATLIM(j), as opposed to 25% in CLOS1.

For zero-crossings, WEIGHT(j) = 24.0. The average value of RATLIM(j) for all three zero-crossing ranges is

$$\frac{.24 + .24 + .32}{3} = \frac{.8}{3} = \frac{4}{15}.$$

We then have that in this case,

$$\frac{5}{2} - \text{FACT} \cdot \text{WEIGHT(j)} \cdot \text{RATIO} > 0$$

if

$$\text{RATIO} < \frac{5}{2 \cdot \text{WEIGHT(j)}} = \frac{5}{48} = (.39)\left(\frac{4}{15}\right).$$

In order for corresponding zero-crossings for two P-segments to be considered close, RATIO must therefore be within 39% of the average value of RATLIM(j) for all three zero-crossing ranges.

3.7 FUNCTION VARFUN

The purpose of VARFUN is to flag the most variable of the parameters A1(i), Z1(i), A2(i), Z2(i), A3(i), Z3(i) for a given P-segment P(i) in the sense of the definition given below. The sole input variable to VARFUN is the row

number i of the given P-segment. VARFUN returns with one of the following seven values:

$$\text{VARFUN} = \begin{cases} 1 \text{ indicates that A1(i) is most variable} \\ 2 \text{ indicates that Z1(i) is most variable} \\ 3 \text{ indicates that A2(i) is most variable} \\ 4 \text{ indicates that Z2(i) is most variable} \\ 5 \text{ indicates that A3(i) is most variable} \\ 6 \text{ indicates that Z3(i) is most variable} \\ 0 \text{ indicates that none of A1(i), Z1(i), A2(i), Z2(i), A3(i), or Z3(i) is most variable.} \end{cases}$$

VARFUN begins by trying to isolate a fricative type "S", characterized as follows:*

$$Z3(i) \geq 40 \text{ and } A1MX(i) \leq \text{ , and } A3MX(i) \geq A1MX(i).$$

If, in addition to the above three tests, we also have that

$$Z3MN(i) < 30,$$

then Z3(i) is called most variable and we set

$$\text{VARFUN} = 6$$

and return. However, if

$$Z3MN(i) \geq 30,$$

we set

$$\text{VARFUN} = 0$$

and return.

If a fricative type "S" cannot be identified, we proceed by defining a function called VARLIM dependent upon the duration DUR(i) of the segment:

$$\text{VARLIM} = \begin{cases} 3 \text{ if } DUR(i) < 6, \\ 4 - \frac{DUR(i)}{6} \text{ if } 6 \leq DUR(i) < 12, \\ 2 + \frac{DUR(i) - 12}{10} \text{ if } DUR(i) \geq 12. \end{cases}$$

*Note that this test for a fricative type "S" differs from that used in CLOS1.

The test for variability results from calculations performed serially on the six parameters A1(i), Z1(i), A2(i), Z2(i), A3(i), and Z3(i). Table 13 contains the values of the variables XLIM1(j) and FACT(j) used in the computations:

Table 13. Parameters for VARFUN

j	XLIM1(j)	FACT(j)
1	6.0	1.75
2	2.0	2.0
3	4.0	1.75
4	4.0	2.0
5	4.0	1.25
6	10.0	1.25

The calculations are the same for each of the six parameters, with appropriate values of XLIM1(j) and FACT(j) used; these computations will be illustrated by those for A1(i):

Define

$$V1 = \max \{A1MN(i), XLIM1(1)\}$$

and

$$V2 = \max \{A1MX(i), XLIM1(1)\}.$$

If

$$| V1 - V2 | \leq XLIM1(1),$$

then the parameter A1(i) is considered to be not variable, and we begin with the same tests for Z1(i) and continue until we find V1 and V2 and a j such that

$$| V1 - V2 | > XLIM1(j).$$

If no such V1, V2, and j can be found, VARFUN = 0, and we return. Otherwise, set

$$\text{FACT1} = \begin{cases} .75 \text{ if } V1 + V2 < 10.0, \\ 1.0 \text{ otherwise}. \end{cases}$$

If

$$\frac{V1 + V2}{|V1 - V2|} \geq \text{VARLIM} \cdot \text{FACT1} \cdot \text{FACT}(1),$$

then the parameter Al(i) is considered to be not variable. Otherwise, the ratio

$$\frac{|V1 - V2| \cdot \text{VARLIM} \cdot \text{FACT1} \cdot \text{FACT}(1)}{V1 + V2}$$

is saved, along with the parameter number (1 for Al(i)). After performing these calculations for all six parameters, we find the largest of the stored ratios, and VARFUN is set equal to the corresponding parameter number.

Extensive computations were performed in attempting to identify the origin of the values FACT(j) and XLIM1(j). These calculations consisted of averaging the function

$$\frac{V1 + V2}{|V1 - V2|}$$

over various ranges of V1 and V2, and combining the averages with various values of VARLIM for different durations. No correlations have yet been found.

3.8 PSEUDO-SUBROUTINES USED BY THE SEGMENTATION PROGRAM

The following routines are used by the segmentation program as internal subroutines by assigning return addresses from one routine to the next and eventually back to segmentation.

3.8.1 Pseudo-Subroutine SEARCH

The purpose of SEARCH is to find and create P-matrix segments from the Q-matrix segments beginning with Q(IS1) and continuing and including Q(IS2).* The segments are created on the basis of the closeness values of the Q-segments. Sustained segments consist of a string of minimal adjacent segments having a positive closeness value and include as their first minimal segment the previously adjacent minimal segment with a negative closeness value as discussed earlier. All other minimal segments not a part of sustained segments become grouped into transitional segments. SEARCH calls the CREAT1 pseudo-subroutine to build the P-matrix row. The exit from SEARCH is to the return address RETSEA.

3.8.2 Psuedo-subroutines CREAT1, CREAT2, and CREAT4

The CREAT pseudo-subroutines are used to compute the values for a P-matrix row.

3.8.2.1 CREAT1

The inputs to CREAT1 are:

SEND = the number of the ending Q-segment of the P-matrix row,
LENSEG = the number of Q-matrix segments in the P-matrix row, and
TYPE = either TR or SUST.

Pseudo-subroutine SEARCH defines a set of Q-segments to be combined into one P-segment on the basis of the closeness values between these Q-segments. SEARCH then calls CREAT1, which allocates the appropriate P-matrix row by assigning a P-matrix row number to SEGNB1 from the AVALBL table and incrementing

* IS1 and IS2 are defined in the segmentation routine before entering SEARCH; IS1 is the initial Q-matrix segment number and IS2 is the final Q-matrix segment number which define the boundaries of the Q-matrix segments to be combined into P-matrix segments.

SEGNB2 (which is the logical row number of the P-segment). It must be noted, however, that SEGNB2 is initialized to zero by the SEGMENT subroutine prior to the first call to SEARCH. Then compute SBEG = SEND - LENSEG.

The minimum, average, and maximum values for each of the parameters A1Q( ), Z1Q( ), A2Q( ), Z2Q( ), A3Q( ), and Z3Q( ) are computed from Q(SBEG +1) through Q(SEND), as discussed earlier in the construction of the P-matrix.

The following variables are then set:

NAT(SEGNB1) = TYPE (either "TR" or "SUST")

INUSE(SEGNB2) = SEGNB1 (the physical P-matrix row number)

BPT(SEGNB1) = SEGNB2 (the logical P-matrix row number)

SBG(SEGNB1) = SBEG

SND(SEGNB1) = SEND

DUR(SEGNB1) = LENSEG

CREAT1 exits through the RETCRE return address.

3.8.2.2 CREAT2

The inputs to CREAT2 are:

SEND = the number of the ending Q-segment of the P-matrix row

SBEG = the number of the beginning (Q-segment) -1 of the P-matrix row

SEGNB1 = the number of the P-matrix row for which the P-matrix parameters are to be recompiled

SEGNB2 = the logical row number of the P-segment

TYPE = either "TR" or "SUST."

CREAT2 begins by computing LENSEG = SEND-SBEG and BETA = $\frac{LENSEG-2}{3}$.

This means that:

BETA = 0 if $0 \leq LENSEG \leq 4$

BETA = 1 if $5 \leq LENSEG \leq 7$

BETA = 2 if $8 \leq LENSEG \leq 10$

3.8.1 Pseudo-Subroutine SEARCH

The purpose of SEARCH is to find and create P-matrix segments from the Q-matrix segments beginning with Q(IS1) and continuing and including Q(IS2).* The segments are created on the basis of the closeness values of the Q-segments. Sustained segments consist of a string of minimal adjacent segments having a positive closeness value and include as their first minimal segment the previously adjacent minimal segment with a negative closeness value as discussed earlier. All other minimal segments not a part of sustained segments become grouped into transitional segments. SEARCH calls the CREAT1 pseudo-subroutine to build the P-matrix row. The exit from SEARCH is to the return address RETSEA.

3.8.2 Psuedo-subroutines CREAT1, CREAT2, and CREAT4

The CREAT pseudo-subroutines are used to compute the values for a P-matrix row.

3.8.2.1 CREAT1

The inputs to CREAT1 are:

SEND = the number of the ending Q-segment of the P-matrix row,
LENSEG = the number of Q-matrix segments in the P-matrix row, and
TYPE = either TR or SUST.

Pseudo-subroutine SEARCH defines a set of Q-segments to be combined into one P-segment on the basis of the closeness values between these Q-segments. SEARCH then calls CREAT1, which allocates the appropriate P-matrix row by assigning a P-matrix row number to SEGNB1 from the AVALBL table and incrementing

---

* IS1 and IS2 are defined in the segmentation routine before entering SEARCH; IS1 is the initial Q-matrix segment number and IS2 is the final Q-matrix segment number which define the boundaries of the Q-matrix segments to be combined into P-matrix segments.

SEGNB2 (which is the logical row number of the P-segment). It must be noted, however, that SEGNB2 is initialized to zero by the SEGMENT subroutine prior to the first call to SEARCH. Then compute SBEG = SEND - LENSEG.

The minimum, average, and maximum values for each of the parameters A1Q( ), Z1Q( ), A2Q( ), Z2Q( ), A3Q( ), and Z3Q( ) are computed from Q(SBEG +1) through Q(SEND), as discussed earlier in the construction of the P-matrix.

The following variables are then set:

NAT(SEGNB1) = TYPE (either "TR" or "SUST")

INJSE(SEGNB2) = SEGNB1 (the physical P-matrix row number)

BPT(SEGNB1) = SEGNB2 (the logical P-matrix row number)

SBG(SEGNB1) = SBEG

SND(SEGNB1) = SEND

DUR(SEGNB1) = LENSEG

CREAT1 exits through the RETCRE return address.

3.8.2.2 CREAT2

The inputs to CREAT2 are:

SEND = the number of the ending Q-segment of the P-matrix row

SBEG = the number of the beginning (Q-segment) -1 of the P-matrix row

SEGNB1 = the number of the P-matrix row for which the P-matrix parameters are to be recompiled

SEGNB2 = the logical row number of the P-segment

TYPE = either "TR" or "SUST."

CREAT2 begins by computing LENSEG = SEND-SBEG and BETA = $\frac{\text{LENSEG}-2}{3}$.

This means that:

BETA = 0 if $0 \leq \text{LENSEG} \leq 4$

BETA = 1 if $5 \leq \text{LENSEG} \leq 7$

BETA = 2 if $8 \leq \text{LENSEG} \leq 10$

BETA = 3 if 11 ≤ LENSEG ≤ 13

BETA = 4 if 14 ≤ LENSEG ≤ 16

etc.

CREAT2 allocates the P-matrix row by assigning a P-matrix row number to SEGNB1 from the AVALBL table and incrementing SEGNB2.

The minimum, average, and maximum values for each of the parameters A1Q( ), Z1Q( ), A2Q( ), Z2Q( ), A3Q( ), and Z3Q( ) are computed from segment Q(SBEG+1+BETA) to Q(SEND-BETA). Thus, the parameters are computed on the basis of values in the inner Q-segments and not for the total range of the Q-segments.

The rationale behind this has not been documented, but we guess one of the reasons to be a by-product of considering transitional segments to be null segments and extending them onto surrounding sustained segments. The duration of the new sustained segment consists of the duration of the transitional segment plus that of the old sustained segment. All of the parameter values of the transitional segment, if included in the computed representative values for the new sustained P-segment, might degrade the purer P-segment values computed only on the basis of the inner segment. Thus, in the case of P-matrix rows computed by CREAT2 or CREAT4, the average parameter values do not represent the average over the duration of the entire segment. We then compute:

NAT(SEGNB1) = TYP' (either "TR" or "SUST")

INUSE(SEGNB2) = SEGNB1 (the physical P-matrix row number)

BPT(SEGNB1) = SEGNB2 (the logical P-matrix row number)

SBG(SEGNB1) = SBEG

SND(SEGNB1) = SEND

DUR(SEGNB1) = LENSEG

CREAT2 exits through the RETCRE return address.

3.8.2.3 CREAT4

The inputs to CREAT4 are:

SEND = the number of the ending Q-segment of the P-matrix row

SBEG = the number of the beginning (Q-segment) -1 of the P-matrix row

SEGNB1 = the number of the P-matrix row for which the P-matrix parameters are to be computed

SEGNB2 = the logical row number of the P-segment

TYPE = either "TR" or "SUST"

LENSEG = either the length of the segment (in terms of minimal segments) or the segmentation subroutine sets LENSEG = 5 for segments less than 50 msec long.

CREAT4 begins by computing BETA = $\frac{LENSEG-2}{3}$ and then proceeds to compute the minimum, average, and maximum parameter values as per CREAT2.

3.8.3 Pseudo-Subroutine COMPAC

The pseudo-subroutine COMPAC is used to consolidate or pack the INUSE( ) table and to reassign BPT( ) values. No further explanation will be given other than to note that BPT(INUSE(i)) is the logical row number for the physical INUSE(i) P-segment.

3.8.4 Pseudo-Subroutine SORT

The pseudo-subroutine SORT sorts the INUSE( ) table into logical P-segment order (i.e., INUSE(1) points to the P-segment having the lowest beginning Q-segment number, INUSE(2) points to the P-segment having the next higher beginning Q-segment number, etc. The BPT( ) table is reset so that the BPT( ) entry corresponding to the physical P-row number for the last logical P-row points to the logical row SIZEP.

3.8.5 Pseudo-Subroutine REORD

The pseudo-subroutine REORD actually moves the physical P-matrix rows into the proper logical order, i.e., physical row 1 is logical row 1, etc.

3.8.6 Pseudo-Subroutine MINMAX

The pseudo-subroutine MINMAX determines which P-segments are local maxima or minima. A complete discussion of the MINMAX function can be found in Section 3.4.3 on secondary segmentation.

3.8.7 Pseudo-Subroutine SUPNOI

The pseudo-subroutine SUPNOI is used to suppress the noisy segments at the beginning and the end of the speech sample. Noisy segments are defined as being those adjacent segments from the beginning minimal segment forward for which AIMX(i) < 8 or Z3MX(i) ≤ 40; or those adjacent segments from the ending minimal segment backwards for which AlMX(i) < 8 or Z3MX(i) ≤ 40.

3.9 CONTROL OF THE P-MATRIX*

Ideally the P-matrix consists of physical rows that are in logical segment order. This is true for primary segmentation. However, whenever a segment is broken into multiple segments or two segments are combined into one segment, the P-matrix would have to be physically sorted to maintain a physical order corresponding to its logical order. A logical table pointing to physical rows is used to conserve time.

During primary segmentation, we set SEGNB2 = 0 (the logical row number or the count of the segments in the P-matrix). The AVALBL table is used to assign physical row numbers and is initialized as follows:

AVALBL(1) = 2
AVALBL(2) = 2
AVALBL(3) = 3
AVALBL(i) = i
. .
. .
. .
AVALBL(202) = 202.

*This section will be of interest mainly to programmers.

AVALBL(1) points to the current AVALBL( ) slot (i.e., AVALBL(AVALBL(1)) which contains the physical P-matrix row number to assign to the logical P-matrix row. When pseudo-subroutine SEARCH calls pseudo-subroutine CREAT1 to create a logical row, SEGNB1 (which is the physical row number) is set equal to AVALBL(AVALBL(1)), and AVALBL(1) is incremented by one. SEGNB2 (which is the logical row number) is incremented by one also. CREAT1 sets up the P-matrix row and the INUSE( ) indicator.

The controls of the subscripts of the rows are established as follows: The physical row BPT(SEGNB1) points to the logical row number SEGNB2, and the logical row INUSE(SEGNB2) points to the physical row number SEGNB1. At the completion of primary segmentation, SIZEP = SEGNB2, the count of the number of logical rows in the P-matrix.

In order to combine two segments P(i) and P(i+1), the following is done:

```
AVALBL(1) = AVALBL(1)-1
AVALBL(AVALBL(1)) = i+1
INUSE(i+1) = 0.
```

In order to break up one segment P(i) into two segments, the following is done:

```
AVALBL(1) = AVALBL(1)-1
AVALBL(AVALBL(1)) = i
INUSE(i) = 0.
```

(Page 58 blank)

SEARCH AND CREAT are then called to make up two new segments on the basis of closeness values computed by function CLOS1 or PROXIM. The following calculations are then performed:

```
INUSE(SIZEP) = AVALBL(AVALBL(1))
BPT(AVALBL(AVALBL(1))) = SIZEP
AVALBL(1) = AVALBL(1)+1
INUSE(SIZEP+1) = AVALBL(AVALBL(1))
BPT(AVALBL(AVALBL(1))) = SIZEP+1
AVALBL(1) = AVALBL(1)+1
```

The COMPAC pseudo-subroutine is called to compress the INUSE( ) table. It eliminates the zero INUSE( ) entries. The SORT pseudo-subroutine is called to sort the INUSE( ) table into logical order. The REORD pseudo-subroutine is called to actually move P-matrix rows into logical order and reassign INUSE( ) values. After reordering, P(2) is logical row 1, P(3) is logical row 2, etc.

APPENDIX

Let $\{f_i\}$ for $i = 1, \ldots, n$ be a discrete function representing an arbitrary wave within a minimal segment. The amplitude of the wave on the minimal segment was defined to be

$$\max_{1 \le i \le n} \{f_i\} - \min_{1 \le i \le n} \{f_i\}.$$

We shall show that for sine waves, this expression is proportional to the square root of the average power of the signal over the minimal segment.

The average power of a continuous signal $f(t)$ is defined in [5] to be

$$\bar{f}^2(t) = \lim_{T \to \infty} \frac{1}{2T} \int_{-T}^{T} |f(t)|^2 \, dt.$$

In the case when $f(t)$ is discrete, the interval $[-T,T]$ can be decomposed into subintervals:

$$-T = t_{-n} < t_{-n+1} < \ldots < t_o < \ldots < t_{n-1} < t_n = T,$$

such that

$$f(t) = f_j \quad \text{for} \quad t_j < t \le t_{j+1} \quad (j = -n, \ldots, -1)$$

and

$$f(t) = f_{j+1} \text{ for } t_j < t \le t_{j+1} \quad (j = 0, 1, \ldots, n-1).$$

If the points $\{t_j\}$ are evenly spaced, say

$$t_j = t_o + jh \quad (j = -n, \ldots, 0, \ldots, n),$$

where $h > 0$, then

$$\frac{1}{2T} = \frac{1}{t_n - t_{-n}} = \frac{1}{t_o + nh - (t_o - nh)} = \frac{1}{2nh}.$$

Therefore,

$$\frac{1}{2T}\int_{-T}^{T} |f(t)|^2 dt = \frac{1}{2nh} \sum_{\substack{j=-n \\ j\neq 0}}^{n} |f_j|^2 \cdot h = \frac{1}{2n} \sum_{\substack{j=-n \\ j\neq 0}}^{n} |f_j|^2,$$

and so

$$\bar{f}^2(t) = \lim_{n\to\infty} \frac{1}{2n} \sum_{\substack{j=-n \\ j\neq 0}}^{n} |f_j|^2.$$

In the case when the discrete signal represents a sine wave, we may take

$$f_j = A \sin \omega t_j \quad \text{and} \quad t_o = 0.$$

Assume that $t_n - t_{-n} > 2\pi/\omega$, so that $\{f_j\}$ contains at least one full cycle between t_{-n} and t_n. Then

$$f_{-j} = A \sin \omega t_{-j} = A \sin \omega(-jh) = -A \sin \omega jh = -A \sin \omega t_j = -f_j.$$

Hence,

$$\bar{f}^2(t) = \lim_{n\to\infty} \frac{1}{2n} \sum_{\substack{j=-n \\ j\neq 0}}^{n} |f_j|^2 = \lim_{n\to\infty} \frac{1}{n} \sum_{j=1}^{n} |f_j|^2.$$

To evaluate this last sum, note that

$$|f_j|^2 = A^2 \sin^2 \omega jh$$

$$= A^2\left(\frac{1-\cos 2\omega jh}{2}\right)$$

$$= \frac{A^2}{2} - \frac{A^2}{2} \operatorname{Re}\, e^{2\omega jhi}, \qquad (i = \sqrt{-1}).$$

Therefore,

$$\sum_{j=1}^{n} |f_j|^2 = n\frac{A^2}{2} - \frac{A^2}{2} \operatorname{Re} \sum_{j=1}^{n} e^{2\omega jhi}$$

$$= n\frac{A^2}{2} - \frac{A^2}{2} \operatorname{Re}\left\{e^{2\omega hi}\,\frac{1-e^{2n\omega hi}}{1-e^{2\omega hi}}\right\}$$

$$= n\frac{A^2}{2} - \frac{A^2}{2} \operatorname{Re}\left\{e^{2\omega hi}\,\frac{(1-e^{2n\omega hi})(1-e^{-2\omega hi})}{2-2\cos 2\omega h}\right\}$$

Let

$$\epsilon = \frac{A^2}{2} \operatorname{Re}\left\{e^{2\omega hi}\,\frac{(1-e^{2n\omega hi})(1-e^{-2\omega hi})}{2-2\cos 2\omega h}\right\}.$$

Then

$$|\epsilon| \le \frac{A^2}{2} \cdot \frac{4}{2-2\cos 2\omega h} = \frac{A^2}{1-\cos 2\omega h} = \text{const}.$$

Thus

$$\frac{1}{n}\sum_{j=1}^{n} |f_j|^2 = \frac{A^2}{2} - \frac{\epsilon}{n},$$

and so

$$\bar{f}^2(t) = \lim_{n\to\infty}\left\{\frac{A^2}{2} - \frac{\epsilon}{n}\right\} = \frac{A^2}{2},$$

and

$$A = \sqrt{2\bar{f}^2(t)}.$$

But since

$$\max_{1\le i\le n}\left\{f_i\right\} - \min_{1\le i\le n}\left\{f_i\right\} = A - (-A) = 2A,$$

we get that

$$\max_{1\le i\le n}\left\{f_i\right\} - \min_{1\le i\le n}\left\{f_i\right\} = 2\sqrt{2}\sqrt{\bar{f}^2(t)}.$$

REFERENCES

[1] VICENS, P., Aspects of Speech Recognition by Computer, Stanford University AI Memo No. 85 (CS 127), 1969.

[2] PETERSON, G. E. and BARNEY, H. L., Control Methods Used in a Study of the Vowels, J. Acoust. Soc. Am., Vol. 24 (1952), pp. 175-184.

[3] REDDY, D. R., An Approach to Computer Speech Recognition by Direct Analysis of the Speech Wave, Stanford University AI Memo No. 43 (CS 49), 1966.

[4] REDDY, D. R., Computer Recognition of Connected Speech, J. Acoust. Soc. Am., Vol. 42 (1967), pp. 329-347.

[5] PAPOULIS, A., The Fourier Integral and Its Applications, New York: McGraw-Hill, 1962.

UNCLASSIFIED

Security Classification

DOCUMENT CONTROL DATA - R & D

(Security classification of title, body of abstract and indexing annotation must be entered when the overall report is classified)

1. ORIGINATING ACTIVITY *(Corporate author)*
System Development Corporation
Santa Monica, California

2a. REPORT SECURITY CLASSIFICATION
Unclassified

2b. GROUP

3. REPORT TITLE
Description and Analysis of the Vicens-Reddy Preprocessing and Segmentation Algorithms

4. DESCRIPTIVE NOTES *(Type of report and inclusive dates)*
Technical Report -- July 1970 - December 1970

5. AUTHOR(S) *(First name, middle initial, last name)*
Iris Kameny
H. Barry Ritea

6. REPORT DATE	7a. TOTAL NO. OF PAGES	7b. NO. OF REFS
4 December 1970	63	5

8a. CONTRACT OR GRANT NO.	9a. ORIGINATOR'S REPORT NUMBER(S)
DAHC15-67-C-0149 b. PROJECT NO. ARPA Order #1327, Amendment #3, Program c. Code #1D30, and 1P10 d.	TM-4652/200/00 9b. OTHER REPORT NO(S) *(Any other numbers that may be assigned this report)*

10. DISTRIBUTION STATEMENT
Distribution of this document is unlimited.

11. SUPPLEMENTARY NOTES	12. SPONSORING MILITARY ACTIVITY

13. ABSTRACT

This document provides a detailed description and analysis of the preprocessing and segmentation procedures used in the Vicens-Reddy speech recognition system.

DD FORM 1 NOV 68 1473

UNCLASSIFIED

Security Classification

14 KEY WORDS	LINK A		LINK B		LINK C	
	ROLE	WT	ROLE	WT	ROLE	WT
Speech Analysis						
Segmentation of Voice Input						
Voice Input						