Communications Research Centre

A FORTRAN PROGRAM FOR TIME ANALYSIS OF VOCAL INTERACTION (TAVI)

by

PIERRE LEWIS AND WILLIAM C. TREURNIET

CRC TECHNICAL NOTE NO. 692

Department of Communications
Ministère des Communications

OTTAWA, JUNE 1978

COMMUNICATIONS RESEARCH CENTRE

DEPARTMENT OF COMMUNICATIONS
CANADA

A FORTRAN PROGRAM FOR TIME ANALYSIS OF VOCAL INTERACTION (TAVI)

by

Pierre Lewis and William C. Treurniet

(Technology and Systems Branch)

CRC TECHNICAL NOTE NO. 692

June 1978
OTTAWA

TABLE OF CONTENTS

A FORTRAN PROGRAM FOR TIME ANALYSIS OF VOCAL INTERACTION (TAVI)

by

Pierre Lewis and William C. Treurniet

ABSTRACT

The method of time analysis of vocal interaction (TAVI) was developed to facilitate automated analysis of temporal patterns of interaction between groups for up to eight nodes. Further documentation on the methodology may be found in CRC Technical Note 684. The present paper describes the organization of the Fortran computer program that computes the relevant variables from a data base created from sampling each channel at a rate of 10 Hz for the presence of a speech signal. In general, vocal interaction among up to eight locations can be analyzed. A specific application is the study of interactions via satellite. In this instance, these data are analyzed from the point of view of each node after correcting to compensate for the satellite circuit delay.

1. INTRODUCTION

The Fortran computer program described in this document is a component of the method of Time Analysis of Vocal Interaction (TAVI). The development of the TAVI method is described in detail in Phillips, Treurniet, Tigges, and Lewis (1977). Briefly, it was developed to facilitate the study of temporal patterns of interaction among groups of people at two to eight locations. Interactions over circuits with long propagation times (e.g., satellite circuits) can also be examined from each location's point of view. The information provided here will be useful to anyone wishing to modify the program or understand its development. The source listing for the TAVI program is included in Appendix M.

1.1 INPUT AND ENVIRONMENT

This section describes how the input data are obtained. The audio from each location or node of a vocal interaction is recorded on a single channel of a multichannel tape recorder. There may be more than one speaker or person at a given location who cannot be electronically separated for various reasons (such as sharing a common microphone or production of acoustical crosstalk). The TAVI program treats a multi-speaker location as a single speaker, corresponding to a single recorded channel. The envelopes of the recorded speech on the various channels are then sampled by a minicomputer at a specifiable rate. This rate is usually 10 Hz, but some other rate may be more suitable if the data are to be manipulated to compensate for propagation delay. The sampling program assumes there is speech on a channel if the envelope's voltage is above a given threshold. The output of this stage is called speech on-off data and is the basic data input to the TAVI analysis program that is described in this note. The format of these data is described in Appendix G.

In this note, the words location, node, channel, speaker, person are essentially synonymous. The actual word used depends on the context. Their relationships can be summarized as follows: there may be one or more persons or speakers at a given location or node whose voices are recorded on a single channel. As mentioned above, all channels are treated by TAVI as if one person were present at a given location even though more than one may actually be present.

1.2 DEFINITIONS OF COMPUTED VARIABLES

The speech from more than one person at a node is handled as if only one person were speaking from that node. Therefore, in the following definitions of the variables computed by the program, the assumption is made that there is one person at each node.

1) FLOOR TIME: whenever there is only one person active, that person then has the floor. He loses the floor when he is not talking unless no one is talking. If, when the original speaker loses the floor, there is more than one other speaker, no one gets the floor until there is again only one person talking. Only one person has the floor at a time. Non-floor time for a speaker is the time he does not have the floor including his unsuccessful interruptions. For each speaker, floor time and non-floor time equal total time. For the group of 2 to 8 persons, sums of floor times may not equal total time because there may be periods when no one is assigned the floor.

2) A FLOOR CYCLE is measured from the time a speaker gains the floor until he gains the floor again after another speaker has had possession of the floor. This period includes a period of floor time followed by a period of non-floor time.

3) A SPEECH is the "on" time for a speaker between "off" times. Speeches may be within floor times or outside floor times; in the latter case they are also interruptions unless no one has the floor.

4) "OFF" TIME for a speaker is divided into two parts. PAUSES are "off" times within floor times, excluding the pause at the end of a speech before a speaker loses the floor. This pause will be called a response time. SILENCES are those "off" times between floor times. A silence is not synonymous with a period of non-floor time because the latter can be broken by unsuccessful interruptions.

5) INTERRUPTIONS are divided into UNSUCCESSFUL and SUCCESSFUL interruptions. An unsuccessful interruption occurs when a second speaker begins a speech while another person is speaking and ends before the first speaker stops speaking. A successful interruption occurs when a person begins a speech while another person is speaking and ends either at the same time or after the first speaker stops speaking.

6) A RESPONSE is a change of speakers. A response time is the time from the end of a speech by the first speaker until the beginning of a speech by another speaker.

7) CHALLENGE time is measured from the time a speaker gains the floor until the beginning of the first interruption.

8) HESITANCIES (H) measure the pause/speech ratio within a person's speech. For each pause (P) (i.e. within floor time), $H=P/(S+P)$ where S is the preceding speech. It is expressed as a percentage.

1.3 OUTPUT OF COMPUTED VARIABLES

The program first outputs general variables; namely, total time, total no-speaker, one-speaker and multi-speaker times. Then, for each node, the program outputs total speech and non-speech times, total floor time, percent speech and percent floor time. It also outputs the mean, standard deviation and frequency for 15 different variables which are measures of various properties of the interaction as viewed from the given locations. These 15 variables are derived from the above definitions and are listed in Appendix C.

The program outputs the results to the printer and to a disc file in a format suitable for further analysis. This output is organized by input channel and is described in Appendix F.

1.4 OVERVIEW OF PROGRAM DESIGN

The program is divided into three main parts:

a) the driver obtains the digitized speech data a sample at a time, and sends each channel's data through a shift register in which smoothing takes place. Then it calls a sample analysis routine for every sample, after recognizing speech and floor transitions.

b) the sample analysis routine computes various measures when a transition occurs. These measures are calculated from stored information about last transitions and other information stored in local variables. It also computes measures such as total speech time.

c) the cumulative statistics routine is called every time a value is produced by the sample analysis routine. It updates running sums for the computation of means and variances. This routine also includes the output routine.

2. DRIVER

The following discussion is accompanied by the flow chart in Appendix J. Additionally, the data format is described in Appendix G and the relevant variable names are listed in Appendix A.

The driver obtains speech on-off data a sample at a time (assumed to have been obtained initially every 100 milliseconds). The data for each channel are put through a shift register which is shifted by one position for every sample obtained. Data are smoothed while passing through the shift register. This part of the program also recognizes speech off-on and on-off transitions, assigns the floor and recognizes floor transitions. It calls the sample analysis routine (VSAMPLE), described in Section 3, for every sample.

2.1 SHIFT REGISTER

The shift register concept is used because it allows continuous smoothing of the incoming raw data without having to store all of the raw data in core memory. Figure 1 will help in understanding the operation and concept of the shift register.

The data contained in the shift registers have the following meaning:

-1 beginning of run

-2 end of run

0 speech off

1 speech on

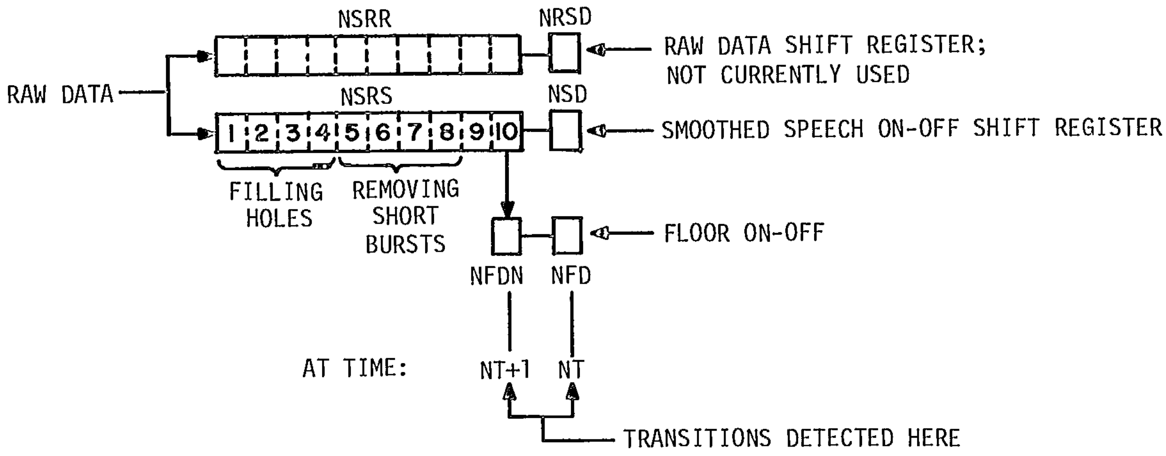


Figure 1. Schematization of Shift Register.

Smoothing will currently fill up to three zeros surrounded by ones and eliminate up to three consecutive ones in isolation (short burst). The smoothing can be described as follows, where position numbers refer to positions within the shift register. If position 1 is on (i.e., contains a 1), then check positions 3 to 5; if any one is on, turn on positions from position 2 to that position minus 1. Now from position 4 onwards the shift register contains data with short pauses removed. If position 4 is off, check positions 6 to 8; if any one is off, clear from position 5 to that position minus 1 in order to remove the short burst. The data from position 7 onwards are now properly smoothed.

2.2 FLOOR ASSIGNMENT

The floor is assigned at the time the data are in position 10 of the shift register as follows: if one person is talking, he or she has the floor; if no one is talking, the first floor remains assigned to the last one who had it; if more than one person is talking, he who has the floor keeps it only as long as he keeps talking. That is, floor assignment does not change until someone else is the only one talking.

2.3 TRANSITION RECOGNITION

Off-on and on-off transitions indicating the beginnings and ends of speech or floor times, are recognized when a change occurs between the current state in NSD and the next state (position 10 of the shift register) or the current state in NFD and the next state in NFDN. The first indicates that speech has begun or ended, while the second indicates that a floor period has begun or ended. The transitions are noted in one of the following arrays:

VSON	beginning of speech
VSOFF	end of speech
VFON	beginning of floor
VFOFF	end of floor

Then the sample analysis routine is called.

3. SAMPLE ANALYSIS ROUTINE

In the following discussion, refer to the flow chart in Appendix K and the variable list in Appendix B.

The sample analysis routine uses information on transitions (obtained from the driver) and data stored within the subroutine to compute values for the desired variables. Data stored within the subroutine include time of the last transitions and second last transitions as well as other required information. The subroutine also computes the various overall measures.

3.1 USE OF TRANSITIONS

The program is based conceptually on the idea that most variables can be computed at transition times from knowledge of past transitions. A past transition is stored in vectors NS and NF (for speech and floor transitions, respectively) as a positive time at which the transition occurred for an off-on transition, and a negative time for an on-off transition. The indices of the vectors refer to the appropriate channel number. For example, a non-floor time value is obtained at the time of a floor off-on transition as NT-ABS(NF(N)) where N is the index (1 to 8) of the speaker concerned. This is equivalent to NT+NF(N) since NF(N) must be negative, the last floor transition being on-off. Similarly, the time to first challenge is the time (NT) at which a speaker begins to interrupt minus the last off-on floor transition of the person who has the floor. Floor cycles and hesitancies require local storage because they both involve one cycle (2 transitions). Also, at the time an interruption occurs, it is not yet known if it will or will not be successful. Hence, the time the interruption starts is stored in a special array. When one of the two speakers involved stops speaking, the type of interruption will be known as well as its duration. How the various other variables are obtained can be deduced from the flow chart in Appendix K.

The sample analysis routine also keeps track of how much time each node spends talking, as well as the total floor time for each node. The total time when no one, one speaker, and more than one speaker are talking is also accumulated.

VLSAMPLE is a special entry point for proper conclusion at the end of data. Again the flow chart says it well.

4. CUMULTIVE STATISTICS AND OUTPUT ROUTINE (INCLUDING VAR2)

This subroutine contains several entry points which allow initialization of running sums and other variables, updating of running sums, and output of all information currently accumulated in running sums or other storage locations.

4.1 CUMULATIVE STATISTICS

Subroutine VINIT initializes the running sums and other storage locations. It is called at the beginning of an analysis, and if subsection analyses are requested, at the beginning of subsection analyses. Subsections are non-overlapping, consecutive intervals of time.

Subroutine VAR1 is called every time a value is obtained for a given variable by the Sample Analysis Routine. It adds the value of the variable to the appropriate running sums. It is used for one-dimensional variables (involving one speaker) such as floor times, "interruptions by",...

Subroutine VAR2 is called for two-dimensional variables such as "interruptions by N to NB". For this example, VAR2 calls VAR1 once for "interruptions by" for N and once for "interruptions to" for NB.

Appendix C gives a list of variables computed by the program. Appendix I describes the generation of additional statistics based on the computation of correlations among the variables. This part of the program also provides for accumulation of data as a function of time. These frequency distributions may be output and used to plot histograms.

4.2 OUTPUT

Output is produced when subroutine VSTAT is called. First the general variables such as total time, one-speaker time, etc., are printed. Then total speech, silence and floor times, and percent speech for each speaker are printed, followed by the one-dimensional variables and the two-dimensional variables.

All the one-dimensional variables plus total time, time when no one, one speaker and two speakers are speaking, total speech, silence and floor times, and percent speech and percent floor time for every speaker may be punched or output to a data file. Appendix F describes this output.

4.3 SUBSECTION ANALYSIS

As indicated previously, subsections are non-overlapping, consecutive intervals of time for which separate printed outputs may be requested. When subsection analyses occur, the statistics are dumped as they have accumulated. No break-up of the input data occurs. Hence, a floor time which starts in one subsection and ends in the next will be correctly included in the statistics of the latter. Requesting subsection analysis also affects output to the summary data file. That is, statistics for each subsection are retained in the file.

5. DELAYED INTERACTIONS

This section describes the special case of the analysis of recorded data from delayed vocal interactions as, for example, over a satellite link. The analysis is performed from each node's point of view, shifting the data such that they correspond to the audio at that node.

The special processing required is confined to the subroutine DUNSPEEL. This subroutine has an entry point (GWRD) which supplies data in the format described in Appendix G. The routine deblocks data from a disc file where data-words are stored in half words in user blocked records. The subroutine contains, in addition, logic to appropriately delay the data from each node if required. It also includes an initializing part which obtains information on the delays, if applicable, to prepare for the analysis. A flow chart of the subroutine is included in Appendix L, a list of relevant variables are described in Appendix H, and the source listing for subprogram DUNSPEEL may be found in Appendix N.

5.1 DELAY COMPUTATIONS

The amount of delay to apply to the data from node j when analyzing from the point of view of node i is simply the time it takes for the audio to go from node j to node i minus the time it takes to go from node j to the tape recorder (delay already present on the recording). This computation often results in negative delays. Therefore, for simplicity, delays are transformed by adding the absolute value of the largest negative delay to all delays.

The entry point DINIT is called at the beginning of each analysis from the point of view of each node. The first time it is called after finding out if there is need for special processing of delayed data, the subroutine queries the user for a delay array (delay from node to node) and a delay vector (delay from node to tape recorder). The subroutine also obtains the input rate (rate at which data on the tape were digitized). This was chosen to be 33 msecs to allow more accurate shifting of data from the Communications Technology Satellite (CTS) experiments. The CTS produces a one-way delay of 264 msec which does not divide well by TAVI's basic rate of 100 msecs. The actual delays are then computed and transformed as described above. The maximum delay (after transformation) is divided by the input rate in msecs to find out how many zero samples will be needed at the end of the file so that no data are lost. This number also gives the dynamic length of the shift register used to compensate for the propagation delay. The actual delay array is converted from msecs to number of samples of delay by dividing by the input rate in msecs.

Before every analysis, control variables are reset and the shift register cleared. Analysis is terminated at this point when all nodes have been considered.

Entry point DOUT is called, for each unique point of view, to print a title indicating for which node, or nodes, the point of view is relevant.

5.2 OBTAINING A SAMPLE - GWRD

The following section applies when analysis of delayed interactions is requested. Otherwise, the data are unspooled and passed directly to TAVI. For each sample requested by TAVI via entry point GWRD, n samples are obtained from the data file and put through a shift register. Here, n equals 100 msecs divided by the input rate in msecs. The sample in the first position of the shift register corresponds to a delay of zero. Delayed data are found further down the shift register. The word of data returned to TAVI is made up of bits (one bit per node - see Appendix G) obtained from appropriately delayed samples for each channel..

Special words in the input data are sent directly to TAVI except "end of file". These may be, at present, JID (job identification) or marker words. The "end of file" is not sent to TAVI until a sufficient number of zero input samples have been fed into the shift register to put all data past the last used position of the shift register.

6. OPERATION

On a general purpose computer, unit numbers need to be assigned to control input and output operations. Running the program on a Xerox Sigma 9 computer, where it is currently implemented, requires the assignment of Data Control Blocks as follows:

!SET	F:5	/file	(input file)
!SET	F:106	/sfile	(summary file) (if omitted, punched output will result)
!SET	F:108	LP	(printed output)

Queries will appear on the console for job title, number of speakers, input channels to consider, length of subsection (type carriage return only if subsections are not required), and whether or not histograms are required. A prompt requesting the number of concatenated files will also appear. This parameter enables the program to skip user supplied end-of-file markers in a data file. Such a file may be composed of a number of smaller files created from successive sections of an input audio tape. If no files were concatenated, "0" or "1" is the appropriate response to the query. Every 3000 data samples a message of the form NT=(time in minutes) will appear on the console to give assurance that the program is running.

7. REFERENCES

Feldstein, S. and Welkowitz, J., *A Chronography of Conversations: In Defense of an Objective Approach* in Nonverbal Behavior and Communications, Siegman, A.W. and Feldstein S. (Eds.), Hillsdale, New Jersey: Lawrence Erlbaum Associates Inc., (in press).

Phillips, D.A., Treurniet, W.C., Tigges, W.S., and Lewis, P., *Time Analysis of Vocal Interaction: A Report on Methodology*, CRC Technical Note 684, 1977.

APPENDIX A

DRIVER VARIABLES

IEOF	end of file switch (initially zero)
NNF	number of concatenated data files
IM	mask for decoding IRD
IRD	16 bit word (standard format, obtained via "DUNSPEEL"*)
JID	job identification (not currently used)
LMK	time of last recognized marker; program ignores markers within 20 seconds of last recognized marker.
N	speaker index
NF	speaker who currently has floor
NFD (N)	floor data
NFDN(N)	next floor data (corresponds to position 10 of shift register)
NFT	temporary guess of NF
NODE(N)	raw data channel corresponding to TAVI output channel label (default is that they are the same)
NPT	number of speakers currently talking
NS	number of speakers (2 to 8)
NSD(N)	smoothed speech data
NSRS(N,10)	smoothed data shift register
NSS	number of raw data channels — normally equal to NS
NSUB	subsection divider
NT	current time in 10th of seconds
VSOFF, VSON, VFOFF, VFON(N)	transitions, indicated by a 1

* *DUNSPEEL is a subroutine which unpacks the input data in a file, returning one sample in a standard word (Appendix G)*

APPENDIX B

SAMPLE ANALYSIS VARIABLES

ITF(N)	total floor time for node N
ITS(N)	total speech time for node N
KK	is non zero when the last person to have gained the floor has not yet been challenged
N	main speaker being considered (for 2-D, "by" speaker)
NB	auxiliary speaker (for 2-D, "to" speaker)
NF(N)	time of last floor transition (+NT)
NFC(N)	beginning of floor cycle
NH(N)	beginning of speech (for hesitancy measure)
NOS	number of speakers talking
NS(N)	time of last speech transition (+NT)
NST	number of speakers (NS elsewhere)
NSI(N,NB)	start of interruption
NS0	no-speaker time
NS1	one-speaker time
NS2	two-speaker time
NT	current time
VSOFF(N), etc.	as in Appendix A

APPENDIX C

CUMULATIVE STATISTICS VARIABLES

IHVAR(IV,N,IH)	histogram counts (1<IH<81)
IHRES(IV)	histogram resolution
IN(IV,N)	1-D count
VN(IV,N)	1-D count normalized
IN2(IV,N,NB)	2-D count
ISVAR(IV,N)	1-D sum
ISVAR2(IV,N,NB)	2-D sum
ISSVAR(IV,N)	1-D sum of squares
ISSVAR2(IV,N,NB)	2-D sum of squares
ISW	histogram printout switch
NTT	total time in tics
VNT	total time in seconds
VOUT(15)	output buffer
TFO(N)	total floor time of others
NFO (N)	total number of floor times of others

IV meanings

1-D variables

1	pauses
2	silences
3	speeches
4	floor cycles
5	floor times
6	non-floor times
7	hesitancies
8	unsuccessful interruptions by
9	unsuccessful interruptions to
10	successful interruptions by
11	successful interruptions to
12	responses by
13	response to
14	challenges by
15	challenges to

2-D variables

1	unsuccessful interruptions
2	successful interruptions
3	responses
4	challenges

APPENDIX D

LABELLED COMMONS

/ASRUN/	NS (NST in VSAMPLE), NT, VSON, VSOFF, VFON, VFOFF
/ASVAR/	ITS, ITF, NS0, NS1, NS2
/ASDAT/	IN2, ISVAR2, ISSVAR2
/TITLE/	TITLE, NODE

APPENDIX E

ENTRY POINTS

MAIN	driver
VSAMPLE	sample analysis routine
VSINIT	initialization; called once at beginning
VLSAMPLE	last sample analysis; called once at end
VINIT	cumulative statistics initialization
VAR1	one-D variables
VSTAT	print, punch
VAR2	two-D variables (calls VAR1 twice)

APPENDIX F

PUNCH OUTPUT

Six cards are punched per speaker (or group if more than one person is at a node) with forty variables on the first five cards. The order of the first ten of these variables are as follows:

1) speaker number (N)

2) total time

3) total time for no speech

4) total time for one speaker speaking

5) total time for two or more speakers speaking

6) total speech time for speaker N

7) total non-speech time for speaker N

8) total floor time for speaker N

9) percent speech for speaker N

10) percent floor time for speaker N

The next 30 variables are the means and frequencies of all 15 one-dimensional variables in the same order as shown in Appendix C. However, the frequencies have been normalized in the following ways:

a) Variables 2 to 6 are divided by total session time

b) Variable 8 is divided by the sum of the total durations of floor times of all other speakers

c) Variables 1 and 9 are divided by the total floor time duration for speaker N

d) Variables 10, 12, and 14 are divided by the sum of the number of floor times of all other speakers

e) Variables 11, 13, and 15 are divided by the number of floor times of speaker N.

Finally, the sixth card contains the unnormalized frequencies of the preceeding 15 variables in the order given in Appendix C.

APPENDIX G

STANDARD FORMAT

The following figure describes how the raw data are organized on disk or magnetic tape before it is read by TAVI. Each sample is held in a 16 bit word and each record is 500 words (1000 bytes) long including the last one. The data is decoded from the file into standard Sigma-9 words by the subroutine DUNSPEEL. A word is returned upon each call.

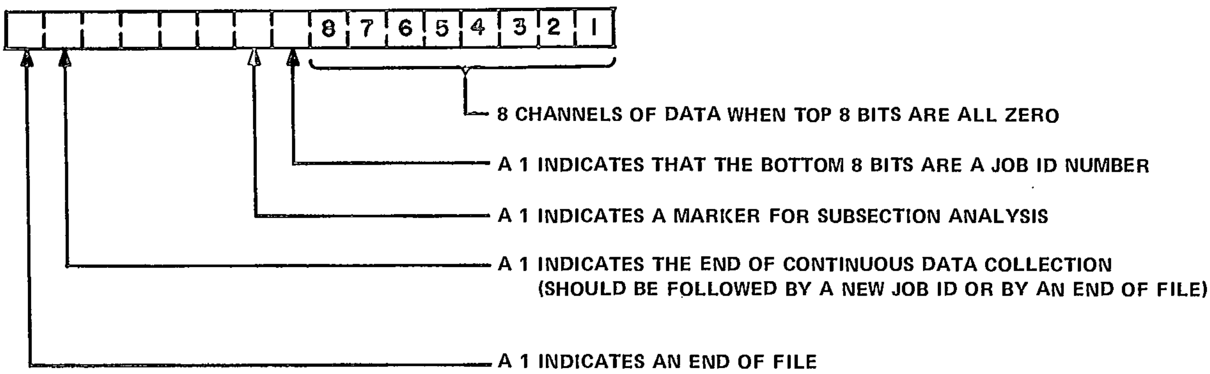


Figure G-1. Schematization of Raw Data Storage

APPENDIX H

DUNSPEEL VARIABLES

DEL(I,J)	actual delay in number of samples applied to J when analyzing for I
IA	a sample (as per appendix G)
IANA	node being considered in current analysis
IDEL(I,J)	delay in msecs from I to J; later, actual delay in msecs applied to I when analyzing for J
IEOF	number of zero samples needed at end of file
IL	dynamic length of shift register
IM	mask for bit manipulations of samples
INR	input rate (typically 33 msecs)
ISN	input sample number (fixed point) of front of shift register
ISW	when 0, delay information not yet entered
	when 9999, analysis from current point of view is equivalent to another already performed (used as switch to skip analysis)
ITR(I)	delay in msecs from I to tape
K(250),KK(500)	arrays for input data debuffering
KSW	index to next sample in KK; when 0, input buffer is empty
LINE(104)	shift register
NS	number of speakers (nodes)
OUTR	output rate (currently set at 100 msecs)
SN	input sample number (floating point) to be sent to TAVI next
SINC	increment to SN (floating point) equal to number of input samples per 100 msecs.

APPENDIX I

CORRELATIONS AMONG VARIABLES

This appendix describes modifications to the TAVI program to compute correlations between various variables. Similar correlations have been called congruence values by other workers in the field (e.g., Feldstein and Welkowitz, in press), and may indicate inter-nodal dependencies among the speech patterns. The computations are handled in a subroutine with entry points CINIT, CSTAT, COR1F, COR1, COR2F, COR2, which are analogous to VINIT, VSTAT, VAR1, VAR2. One major difference is that correlations require two values and, in the present situation, these appear at different times (e.g., while correlating the successive durations of floor times for two different nodes). Hence, one entry point is used to submit the first value, and the other one to submit the second value and compute running sums. This method of collecting values greatly simplifies modifications to TAVI. Values for correlations are obtained in a way similar to other variables.

CORRELATION SUBROUTINE

Entry point CINIT initialises all running sums, and is called at the same time as VINIT is called. The first time it is called, it queries the user on whether or not correlations are wanted.

Entry point CSTAT causes output of correlation coefficients, Student t values and frequencies. First the two dimensional (transition) correlations are printed, then the one-dimensional correlations are printed along with a zero-dimensional correlation coefficient for each variable.

The first value of a pair of values to be correlated is submitted via COR2F (two-dimensional correlations) or COR1F (one-dimensional). The second value is submitted via COR2 or COR1. When the second value is submitted, the routine first checks for the presence of a first value (which would have been stored in an array by COR2F or COR1F) and for meaningful speaker indices (both non zero, different if two-dimensional or same if one-dimensional). If all is as it should be, the values are incorporated into the running sums. In the case of two-dimensional variables, these values are also incorporated into appropriate one-dimensional running sums (as with VAR2 calling VAR1 twice).

Variables

IN, IX, IY, IXX, IYY, IXY(IV,N)	one-dimensional running sums
ISN, ISX, ISY, ISXX, ISYY ISXY(IV,N,NB)	two-dimensional running sums
INP, IXF(IV)	one-dimensional first speaker and value
ISNP, ISXF(IV)	two-dimensional first speaker and value
IS, SX, SY, SXX, SYY, SXY	zero-dimensional running sums
ISW	correlation computation switch
NS	number of speakers (nodes)
R	correlation coefficient
VOUT, T, IDF(n)	preparation of output data

CHANGES TO VSAMPLE

In order to compute the correlations, changes to subroutine VSAMPLE (Appendix K) were required, and are described as follows.

At the time of hesitancy computations, appropriate values are stored into running sums for mean speeches, pauses and hesitancies.

At the time of floor time computations, the floor time value is sent twice, once as the last of the current pair, and once as the first of the next pair. A similar procedure is followed for floor cycles.

At the time a speaker loses the floor, mean speeches, pauses and hesitancies are computed (from running sums) and sent twice as described in the previous paragraph. In addition, mean pauses is sent as the first value of a pause-response pair and the second value of a response-pause pair.

The time a response is recognized is the wrong time to send values for a pause-response or a response-pause pair because the mean pauses relating to the preceeding floor time will be computed later (at the time of VFOFF transition analysis). Hence, the values and speaker numbers are stored in NTD, ND, NBD and are sent after VFOFF processing as the first value of a response-pause pair and the second value of a pause-response pair.

When VLSAMPLE is called at the end of the input data, a second value of floor time is sent. Initialisation for IRS, IRN and ND is added under VSINIT.

IV meanings for correlations

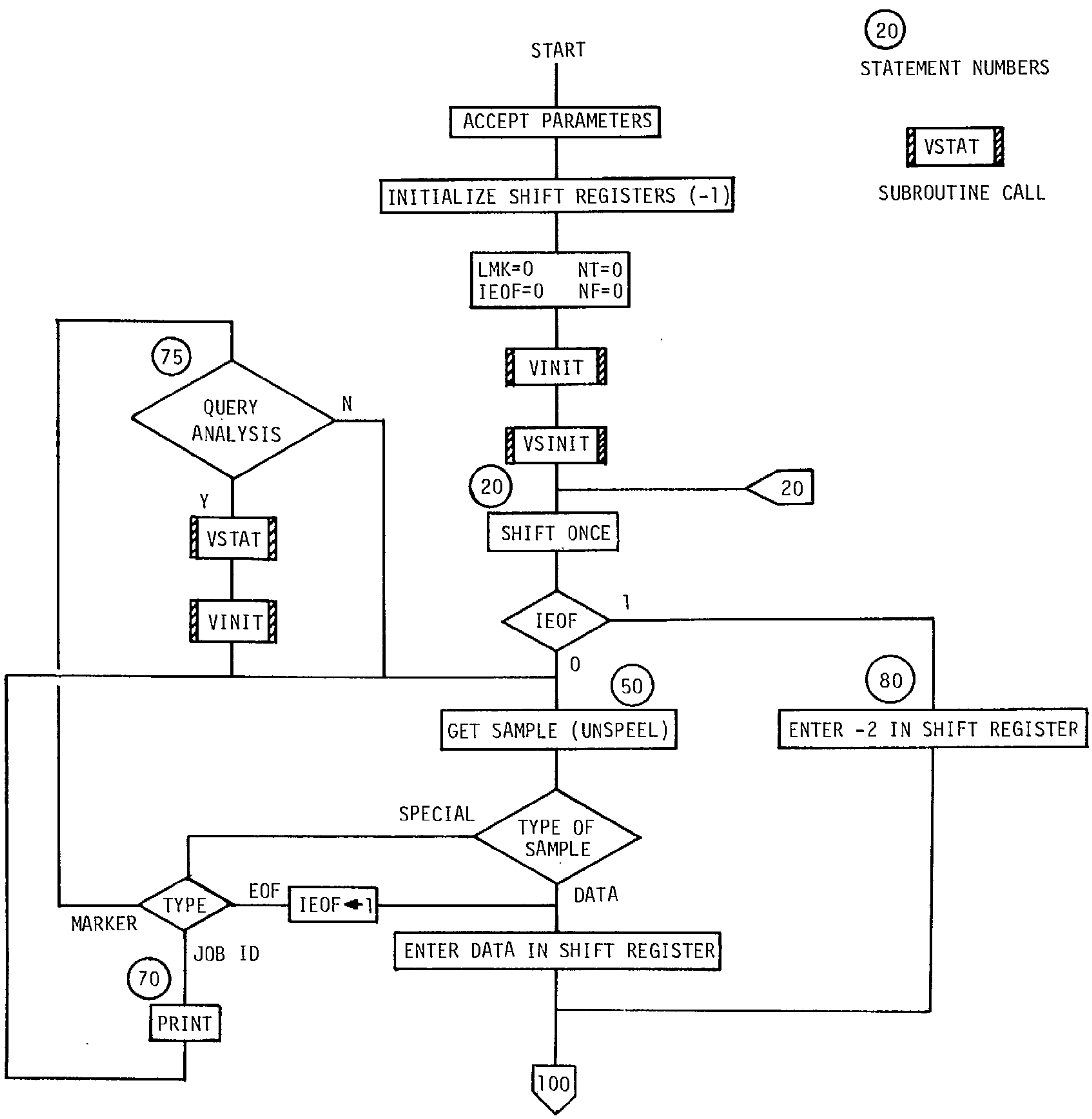
2–D	1–D	meaning
1	1,2	pauses by.. with following response by..
2	3,4	response to.. with following pauses by..
3	5,6	speeches
4	7,8	floor cycles
5	9,10	floor times
6	11,12	pauses
7	13,14	hesitancies

New variables in VSAMPLE

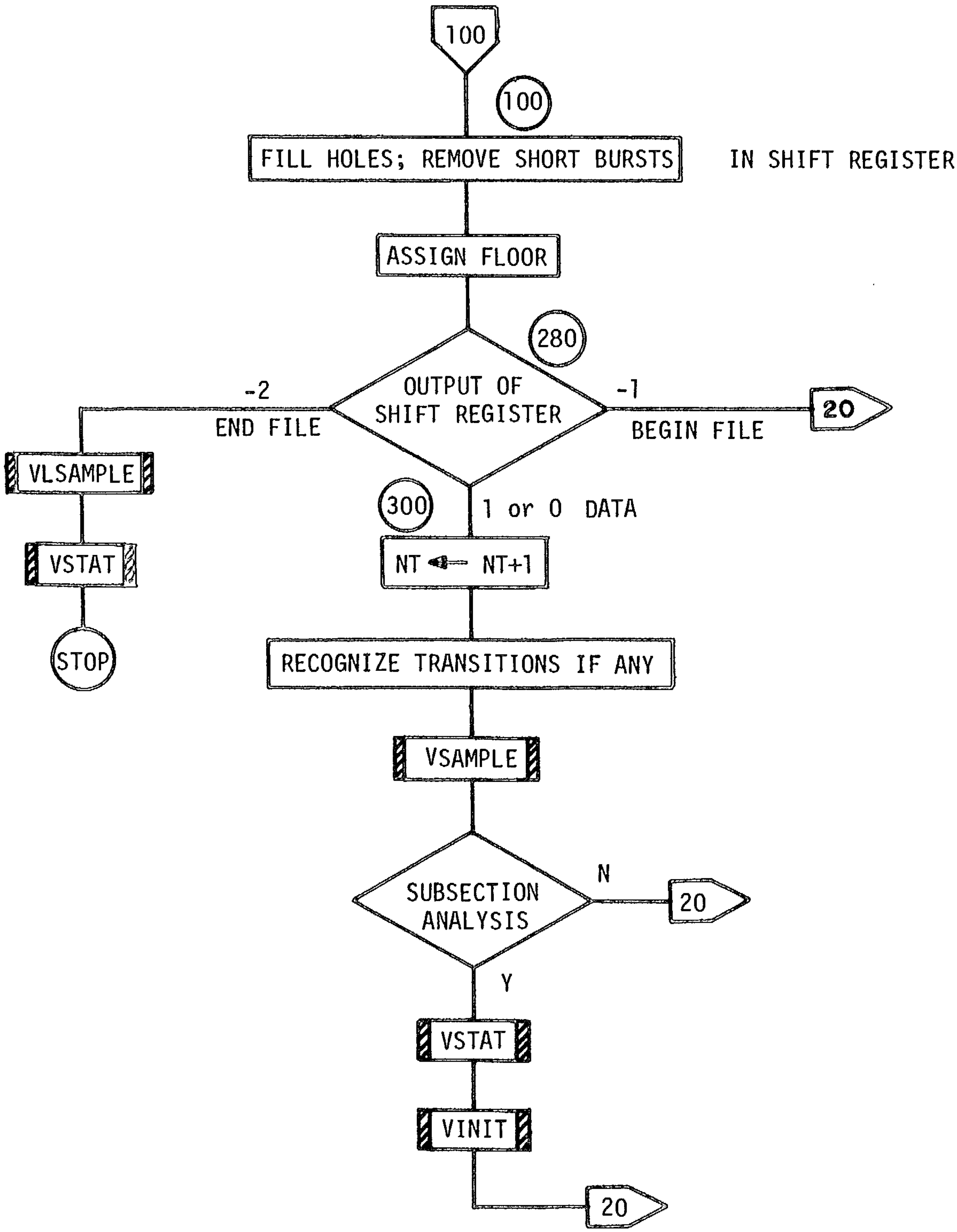
IRN	frequency for mean pauses,....
IRS (3)	running sums for mean pauses, speeches, hesitancies
NTD, ND, NBD	response storage (none if ND=0)

APPENDIX J

DRIVER FLOW CHART

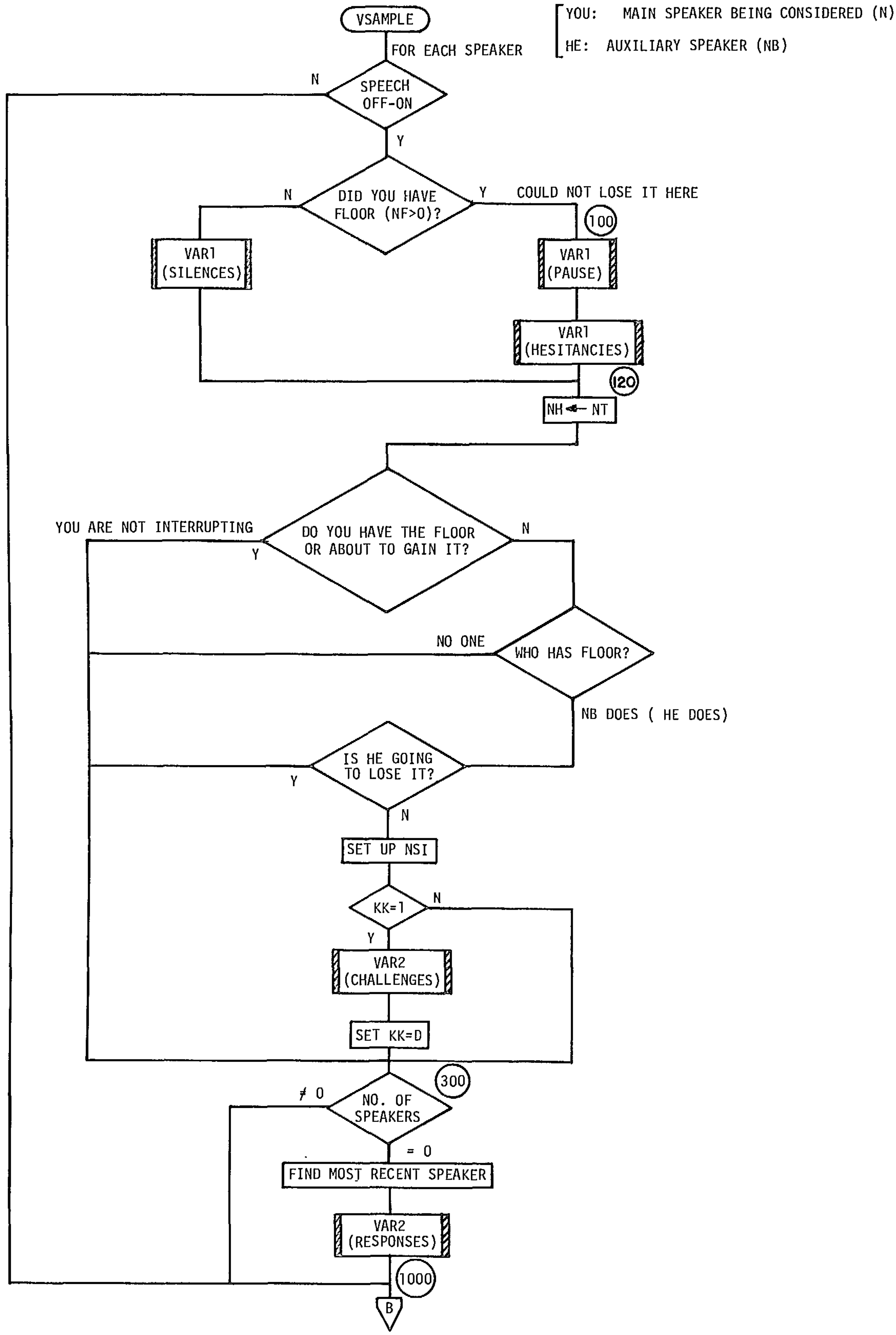


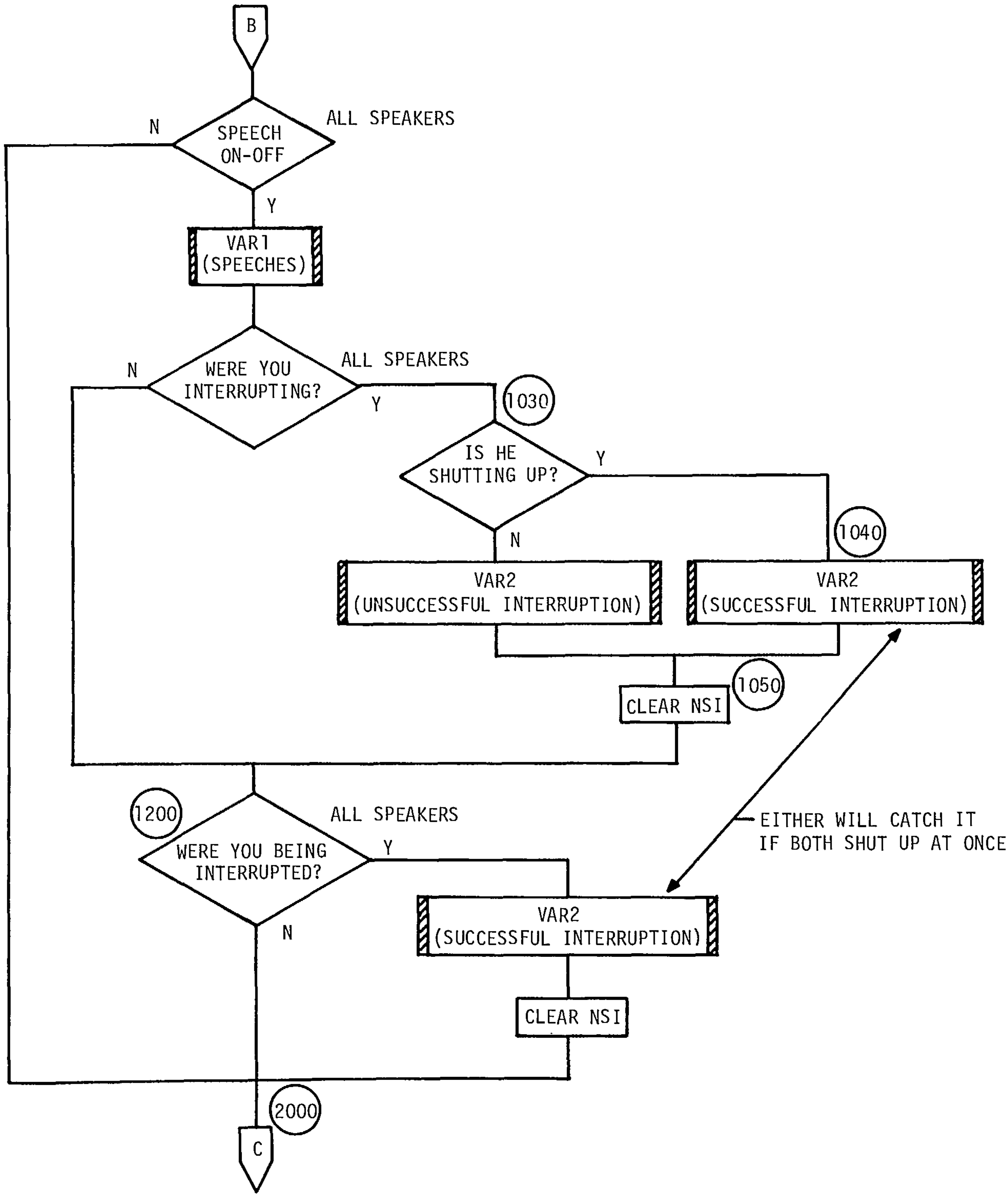
20
STATEMENT NUMBERS
VSTAT
SUBROUTINE CALL
START
ACCEPT PARAMETERS
INITIALIZE SHIFT REGISTERS (-1)
LMK=0 NT=0
IEOF=0 NF=0
VINIT
VSINIT
20
SHIFT ONCE
20
IEOF
1
0
50
80
GET SAMPLE (UNSPEEL)
ENTER -2 IN SHIFT REGISTER
75
QUERY ANALYSIS
N
Y
VSTAT
VINIT
SPECIAL
TYPE OF SAMPLE
DATA
TYPE
EOF
IEOF ← 1
MARKER
JOB ID
70
PRINT
ENTER DATA IN SHIFT REGISTER
100

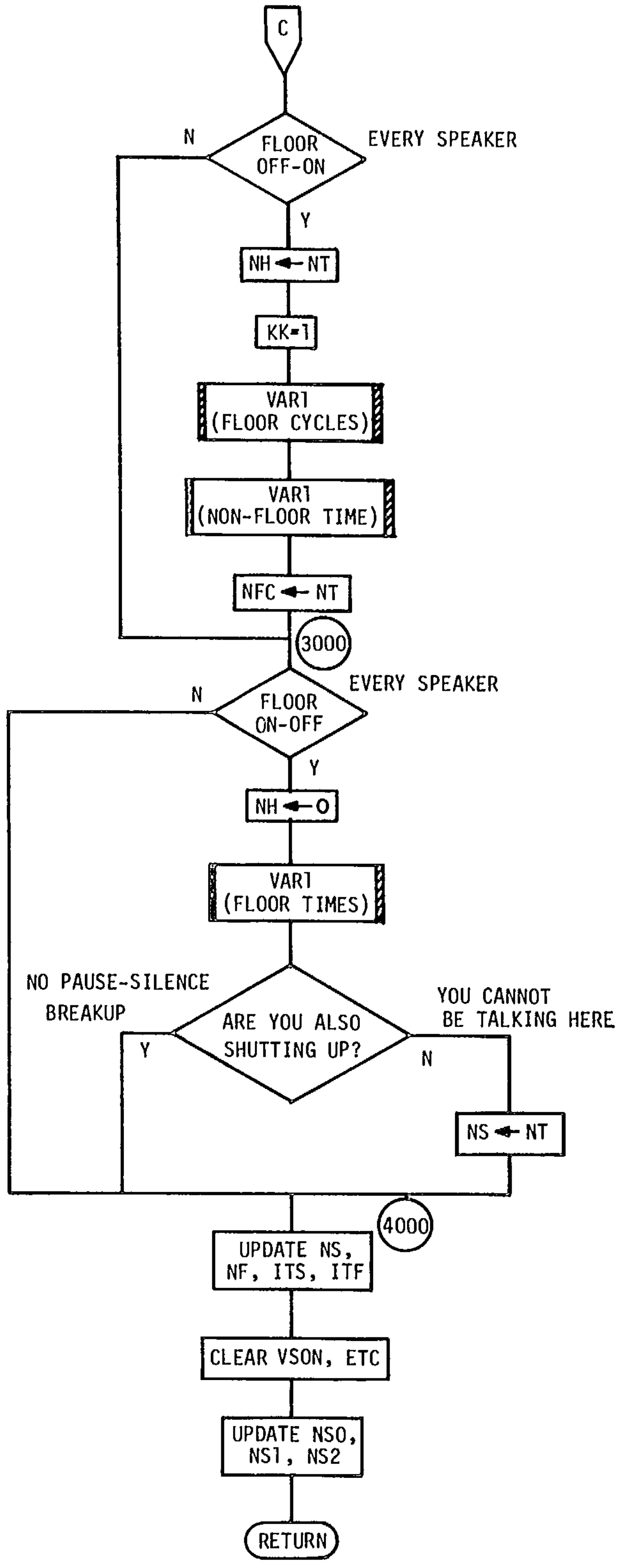

100
100
FILL HOLES; REMOVE SHORT BURSTS
IN SHIFT REGISTER
ASSIGN FLOOR
280
OUTPUT OF SHIFT REGISTER
-2
END FILE
-1
BEGIN FILE
20
VLSAMPLE
VSTAT
STOP
300
1 or 0 DATA
NT ← NT+1
RECOGNIZE TRANSITIONS IF ANY
VSAMPLE
SUBSECTION ANALYSIS
N
20
Y
VSTAT
VINIT
20

APPENDIX K

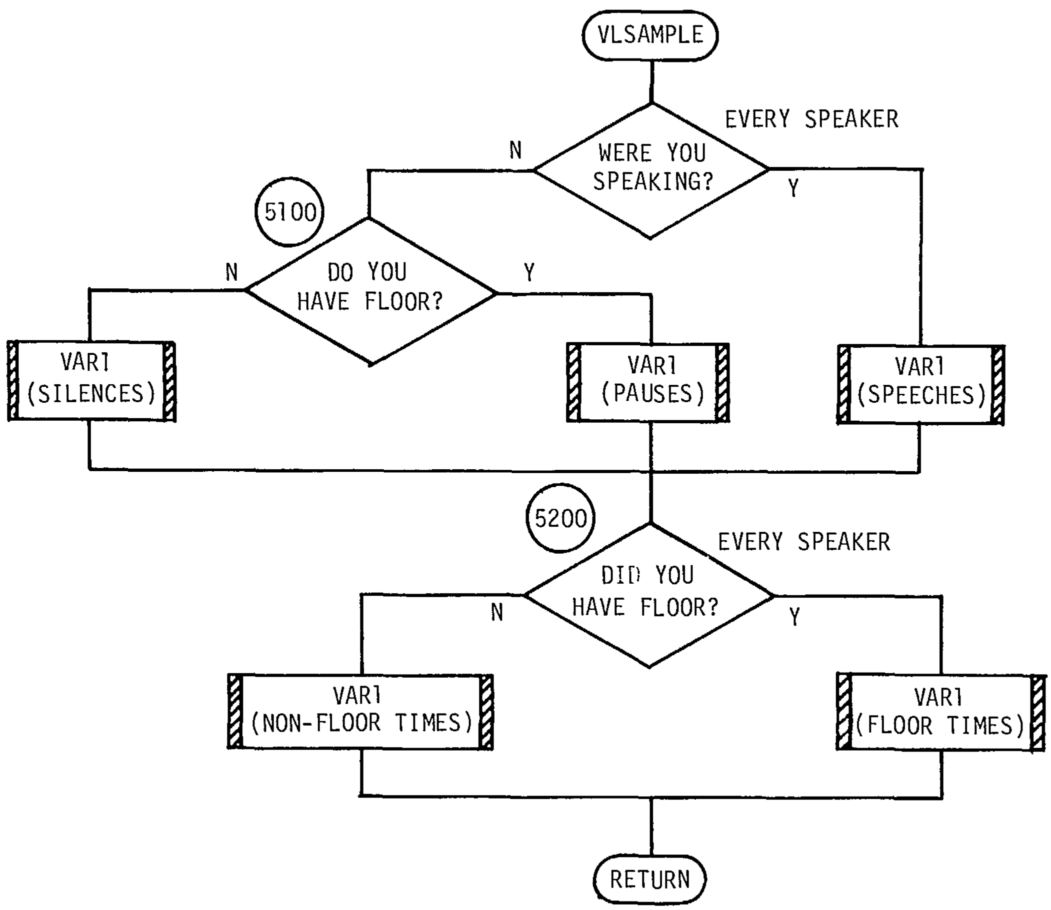
VSAMPLE, VLSAMPLE FLOW CHARTS
(SAMPLE ANALYSIS ROUTINE)


VSAMPLE
FOR EACH SPEAKER
YOU: MAIN SPEAKER BEING CONSIDERED (N)
HE: AUXILIARY SPEAKER (NB)
N
SPEECH OFF-ON
Y
N
DID YOU HAVE FLOOR (NF>0)?
Y
COULD NOT LOSE IT HERE
100
VAR1 (SILENCES)
VAR1 (PAUSE)
VAR1 (HESITANCIES)
120
NH ← NT
YOU ARE NOT INTERRUPTING
Y
DO YOU HAVE THE FLOOR OR ABOUT TO GAIN IT?
N
NO ONE
WHO HAS FLOOR?
NB DOES (HE DOES)
IS HE GOING TO LOSE IT?
Y
N
SET UP NSI
KK=1
N
Y
VAR2 (CHALLENGES)
SET KK=D
300
≠ 0
NO. OF SPEAKERS
= 0
FIND MOST RECENT SPEAKER
VAR2 (RESPONSES)
1000
B


B
SPEECH ON-OFF
ALL SPEAKERS
N
Y
VAR1 (SPEECHES)
WERE YOU INTERRUPTING?
ALL SPEAKERS
N
Y
1030
IS HE SHUTTING UP?
Y
N
1040
VAR2 (UNSUCCESSFUL INTERRUPTION)
VAR2 (SUCCESSFUL INTERRUPTION)
1050
CLEAR NSI
1200
WERE YOU BEING INTERRUPTED?
ALL SPEAKERS
Y
N
EITHER WILL CATCH IT
IF BOTH SHUT UP AT ONCE
VAR2 (SUCCESSFUL INTERRUPTION)
CLEAR NSI
2000
C

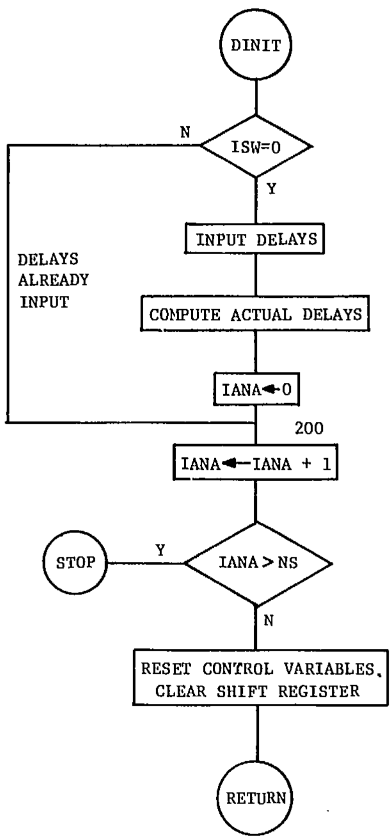


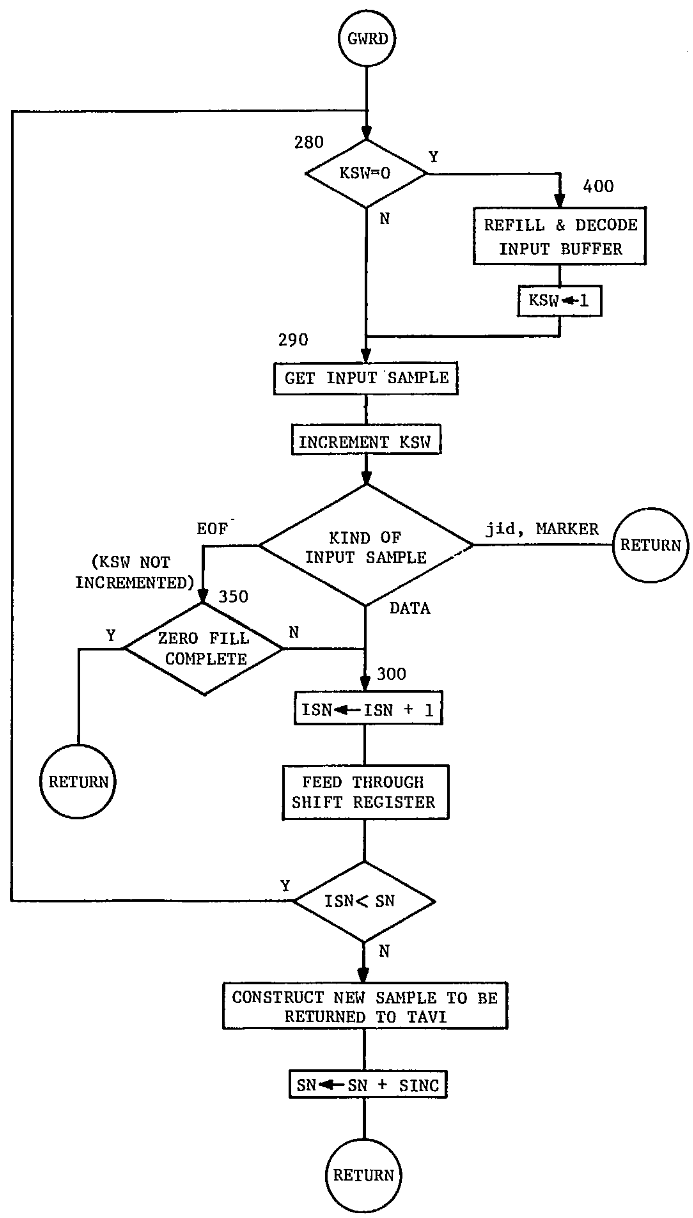
C
FLOOR OFF-ON
EVERY SPEAKER
N
Y
NH ← NT
KK=1
VAR1 (FLOOR CYCLES)
VAR1 (NON-FLOOR TIME)
NFC ← NT
3000
FLOOR ON-OFF
EVERY SPEAKER
N
Y
NH ← 0
VAR1 (FLOOR TIMES)
NO PAUSE-SILENCE BREAKUP
ARE YOU ALSO SHUTTING UP?
YOU CANNOT BE TALKING HERE
Y
N
NS ← NT
4000
UPDATE NS, NF, ITS, ITF
CLEAR VSON, ETC
UPDATE NS0, NS1, NS2
RETURN


VLSAMPLE
EVERY SPEAKER
N
WERE YOU SPEAKING?
Y
5100
N
DO YOU HAVE FLOOR?
Y
VAR1 (SILENCES)
VAR1 (PAUSES)
VAR1 (SPEECHES)
5200
EVERY SPEAKER
DID YOU HAVE FLOOR?
N
Y
VAR1 (NON-FLOOR TIMES)
VAR1 (FLOOR TIMES)
RETURN

APPENDIX L

DUNSPEEL FLOW CHART


DINIT
ISW=0
N
Y
INPUT DELAYS
DELAYS ALREADY INPUT
COMPUTE ACTUAL DELAYS
IANA←0
200
IANA←IANA + 1
STOP
Y
IANA > NS
N
RESET CONTROL VARIABLES, CLEAR SHIFT REGISTER
RETURN


GWRD
280
KSW=0
Y
400
N
REFILL & DECODE INPUT BUFFER
KSW←1
290
GET INPUT SAMPLE
INCREMENT KSW
EOF
KIND OF INPUT SAMPLE
jid, MARKER
RETURN
(KSW NOT INCREMENTED)
350
DATA
Y
ZERO FILL COMPLETE
N
300
ISN←ISN + 1
RETURN
FEED THROUGH SHIFT REGISTER
Y
ISN< SN
N
CONSTRUCT NEW SAMPLE TO BE RETURNED TO TAVI
SN←SN + SINC
RETURN

APPENDIX M

TAVI SOURCE LISTING

```
C
CCCC   DRIVER
C
       DIMENSION   NSRS(8,10),NFDN(8)
       INTEGER VSON(8),VSOFF(8),VFON(8),VFOFF(8)
       DIMENSION NSD(8) ,NFD(8)
       COMMON /ASRUN/ NS,NT,VSON,VSOFF,VFON,VFOFF
       DIMENSION TITLE(15)
       DIMENSION NODE(8),NODE2(8)
       COMMON /TITLE/ TITLE, NODE
       COMMON /DSP/ NSS
       DO 17 I=1,8
    17 NODE(I)=I
CCCC   INITIALIZE
       TYPE 6
     6 FORMAT(' TYPE TITLE - 60 CHARS')
       ACCEPT 7,TITLE
     7 FORMAT(15A4)
       PRINT 5,TITLE
     5 FORMAT('1',20X,15A4)
       TYPE 8
     8 FORMAT(' TYPE NUMBER OF CONCATENATED FILES - NN')
       ACCEPT 4,NNF
       IF(NNF.EQ.0)NNF=1
       TYPE 1
     1 FORMAT(' TYPE TOTAL NO OF CHANNELS - N')
       ACCEPT 2,NS
       NSS=NS
     2 FORMAT(I1)
X      TYPE 21
X   21 FORMAT(' TYPE CHANNELS TO CONSIDER - 12345678')
X      ACCEPT 22,NODE2
X   22 FORMAT(8I1)
X      IF (NODE2(1).EQ.0)GO TO 27
X      NS=0
X      DO 24 I=1,NSS
X      NODE(I)=NODE2(I)
X      IF(NODE(I).NE.0)NS=NS+1
X   24 CONTINUE
X   27 CONTINUE
X      DO 23 I=1,NS
X   23 NODE2(I)=2**(NODE(I)-1)
       TYPE 3
     3 FORMAT(' SUBSECTION LENGTH MIN - NN')
       ACCEPT 4,NSUB
     4 FORMAT(I2)
       IF(NSUB .EQ.0)NSUB=10000
       NSUB=NSUB*600
X      TYPE 12
X   12 FORMAT(' ANALYSIS LIMITS IN SECS - NNNN NNNN')
X      ACCEPT 13,LIML,LIMH
X   13 FORMAT(I4,I5)
```

```
X      IF(LIMH.LE.Ø)LIMH=9999
X      PRINT 14,LIML,LIMH
X   14 FORMAT(' ANALYSIS FROM',I5,' TO',I5,' SECONDS')
X      LOW=Ø
X      LIML=LIML*1Ø
X      LIMH=LIMH*1Ø-LIML-11
    15 CALL DINIT
       DO 1Ø N=1,NS
       NFDN(N)=Ø
       NSRS(N,1)=Ø
       DO 1Ø I=2,1Ø
    1Ø NSRS(N,I)=-1
       NT=Ø
       LMK=Ø
       CALL VINIT
       CALL CINIT
       CALL VSINIT
       NFILE=Ø
       IEOF=Ø
       NF=Ø
CCCC   DRIVER (TOP OF)
    2Ø DO 4Ø N=1,NS
       NFD(N)=NFDN(N)
       NSD(N)=NSRS(N,1Ø)
       DO 3Ø I=1Ø,2,-1
    3Ø NSRS(N,I)=NSRS(N,I-1)
    4Ø CONTINUE
       IF (IEOF .EQ. 1)GO TO 8Ø
CCCC   GET A SAMPLE (STANDARD FORMAT) VIA DSPEEL
    5Ø CALL GWRD(IRD)
       IF(IRD.EQ.512)GO TO 75
       IF(IRD.GE.256 .AND. IRD.LT.512)GO TO 7Ø
       IF(IRD.EQ.32768)NFILE=NFILE+1
       IF(NFILE.EQ.NNF)IEOF=1
       IF(IRD.EQ.32768.AND.NFILE.NE.NNF)GO TO 5Ø
C      IF(IRD.EQ.16384)GO TO 75
       IF(IRD.EQ.16384)GO TO 5Ø
X      LOW=LOW+1
X      IF(LOW.LT.LIML)GO TO 5Ø
X      IF(NT.GT.LIMH)IEOF=1
X      GO TO 63
    55 IM=1
       DO 6Ø N=1,NS
        NSRS(N,1)=Ø
       IF (IAND(IRD,IM).NE.Ø) NSRS(N,1)=1
    6Ø IM=IM+IM
       GO TO 1ØØ
X   63 DO 65 N=1,NS
X       NSRS(N,1)=Ø
X      IF(IAND(IRD,NODE2(N)).NE.Ø) NSRS(N,1)=1
X   65 CONTINUE
X      GO TO 1ØØ
```

```
CCCC  JID
   7Ø JID=IRD-256
      OUTPUT JID
      GO TO 5Ø
CCCC  MARKER
   75 IF(NT-LMK .LT. 2ØØ)GO TO 5Ø
      TYPE 76,NT
   76 FORMAT(' MARKER AT',I6,', SUBSECTION ANALYSIS? 1 - YES')
      ACCEPT 2,ISW
      IF (ISW .NE. 1)GO TO 5Ø
      CALL VSTAT
      CALL CSTAT
      CALL VINIT
      CALL CINIT
      LMK=NT
      GO TO 5Ø
CCCC  END OF RUN FILL
   8Ø DO 9Ø N=1,NS
   9Ø  NSRS(N,1)=-2
CCCC  FILL HOLES
  1ØØ DO 2ØØ N=1,NS
      IF(NSRS(N,1).NE.1) GO TO 15Ø
      DO 11Ø IL=3,5
  11Ø IF(NSRS(N,IL).EQ.1)GO TO 12Ø
      GO TO 15Ø
  12Ø DO 13Ø I=2,IL-1
  13Ø NSRS(N,I)=1
CCCC  REMOVE SHORT BURSTS
  15Ø IF(NSRS(N,4).NE.Ø)GO TO 2ØØ
      DO 16Ø IL=6,8
  16Ø IF(NSRS(N,IL).EQ.Ø)GO TO 17Ø
      GO TO 2ØØ
  17Ø DO 18Ø I=5,IL-1
  18Ø NSRS(N,I)=Ø
  2ØØ CONTINUE
CCCC  ASSIGN THE FLOOR
      NPT=Ø
      DO 21Ø N=1,NS
      IF(NSRS(N,1Ø).EQ.1)NPT=NPT+1
  21Ø IF(NSRS(N,1Ø).EQ.1)NFT=N
      IF(NPT.EQ.Ø)GO TO 28Ø
      IF(NPT.GT.1)GO TO 24Ø
      IF(NFT.EQ.NF)GO TO 28Ø
      IF(NF.NE.Ø)NFDN(NF)=Ø
      NFDN(NFT)=1
      NF=NFT
      GO TO 28Ø
  24Ø IF(NF.EQ.Ø)GO TO 28Ø
      IF(NSRS(NF,1Ø).NE.1)NFDN(NF)=Ø
      IF(NFDN(NF).EQ.Ø)NF=Ø
  28Ø IF(NSD(1).EQ.-1)GO  TO 2Ø
      IF(NSRS(1,1Ø).EQ.-2)GO TO 1ØØØ
```

```
  300 NT=NT+1
      IF(3000*(NT/3000).EQ.NT)TYPE 301,NT/600
  301 FORMAT(' NT=',I6)
CCCC  TRANSITIONS AND CALL TO VSAMPLE
  310 DO 340 N=1,NS
      IF(NSD(N).EQ.NSRS(N,10))GO TO 320
      IF(NSD(N).EQ.1) VSOFF(N)=1
      IF(NSD(N).EQ.0) VSON(N)=1
  320 IF(NFD(N).EQ.NFDN(N))GO TO 330
      IF(NFD(N).EQ.1) VFOFF(N)=1
      IF(NFD(N).EQ.0) VFON(N)=1
  330 CONTINUE
  340 CONTINUE
      CALL VSAMPLE
      IF(NSUB*(NT/NSUB).NE.NT)GO TO 20
      CALL VSTAT
      CALL CSTAT
      CALL VINIT
      CALL CINIT
      GO TO 20
CCCC  END OF RUN
 1000 CALL VLSAMPLE
      CALL VSTAT
      CALL CSTAT
      GO TO 15
      END
C
CCCC  EVERY SAMPLE ANALYSIS ROUTINE
C
      SUBROUTINE VSAMPLE
      DIMENSION IRS(3)
      DIMENSION NF(8),NS(8),NFC(8),NH(8),NSI(8,8),ITS(8),
     #ITF(8)
      INTEGER VSON(8),VSOFF(8),VFON(8),VFOFF(8)
      COMMON /ASRUN/ NST,NT,VSON,VSOFF,VFON,VFOFF
      COMMON /ASVAR/ ITS,ITF,NS0,NS1,NS2
CCCC  VSON PROCESSING
      DO 1000 N=1,NST
      IF(VSON(N).EQ.0)GO TO 1000
      IF(NF(N).GT.0)GO TO 100
      CALL VAR1(2,N,NT+NS(N))
      GO TO 120
  100 CALL VAR1(1,N,NT+NS(N))
      IF(NH(N).EQ.0)GO TO 120
      IHV=1000.*(NT+NS(N))/(0.+NT-NH(N))+.5
      IRN=IRN+1
      IRS(1)=IRS(1)+NT+NS(N)
      IRS(2)=IRS(2)-NS(N)-NH(N)
      IRS(3)=IRS(3)+IHV
      CALL VAR1(7,N,IHV)
  120 NH(N)=NT
      IF(VFON(N).NE.0 .OR. NF(N).GT.0)GO TO 300
```

```
      DO 130 NB=1,NST
  130 IF(NF(NB).GT.0)GO TO 140
      GO TO 300
  140 IF (VFOFF(NB).NE.0)GO TO 300
      NSI(N,NB)=NT
      IF(KK.EQ.0)GO TO 300
      CALL VAR2(4,N,NB,NT-NF(NB))
      KK=0
  300 NOS=0
      DO 310 NB=1,NST
  310 IF(NS(NB).GT.0)NOS=NOS+1
      IF(NOS.NE.0)GO TO 1000
      MAX=0
      DO 320 NB=1,NST
  320 IF(-NS(NB).GT.MAX)MAX=-NS(NB)
      IF (MAX .EQ.0)GO TO 1000
      DO 330 NB=1,NST
      IF (N.EQ.NB)GO TO 330
      IF(-NS(NB).NE.MAX)GO TO 330
      ND=N
      NBD=NB
      NTD=NT-MAX
      CALL VAR2(3,N,NB,NT-MAX)
  330 CONTINUE
 1000 CONTINUE
CCCC  VSOFF PROCESSING
      DO 2000 N=1,NST
      IF(VSOFF(N).EQ.0)GO TO 2000
      CALL VAR1(3,N,NT-NS(N))
      DO 1070 NB=1,NST
      IF(NSI(N,NB).LE.0)GO TO 1070
      IF(VSOFF(NB).NE.0)GO TO 1040
      CALL VAR2(1,N,NB,NT-NS(N))
      GO TO  1050
 1040 CALL VAR2(2,N,NB,NT-NS(N))
 1050 NSI(N,NB)=0
 1070 CONTINUE
 1200 DO 1270 NB=1,NST
      IF(NSI(NB,N).LE.0)GO TO 1270
      CALL VAR2(2,NB,N,NT-NS(NB))
      NSI(NB,N)=0
 1270 CONTINUE
 2000 CONTINUE
CCCC  VFON PROCESSING
      DO 3000 N=1,NST
      IF(VFON(N).EQ.0)GO TO 3000
      KK=1
      NH(N)=NT
      IF(NFC(N).EQ.0)GO TO 2100
      CALL VAR1(4,N,NT-NFC(N))
      CALL COR2(4,N,NT-NFC(N))
      CALL COR2F(4,N,NT-NFC(N))
```

```
 2100 CONTINUE
      NFC(N)=NT
      CALL VAR1(6,N,NT+NF(N))
 3000 CONTINUE
CCCC  VFOFF PROCESSING
      DO 4000 N=1,NST
      IF(VFOFF(N).EQ.0)GO TO 4000
      NH(N)=0
      CALL VAR1(5,N,NT-NF(N))
      CALL COR2(5,N,NT-NF(N))
      CALL COR2F(5,N,NT-NF(N))
      IF(VSOFF(N).NE.0)GO TO 3500
      NS(N)=-NT
 3500 NB=N
      IF(IRN.EQ.0)NB=0
      CALL COR2(6,NB,IRS(1)/IRN)
      CALL COR2F(6,NB,IRS(1)/IRN)
      CALL COR2(3,NB,IRS(2)/IRN)
      CALL COR2F(3,NB,IRS(2)/IRN)
      CALL COR2(7,NB,IRS(3)/IRN)
      CALL COR2F(7,NB,IRS(3)/IRN)
      CALL COR2(2,NB,IRS(1)/IRN)
      CALL COR2F(1,NB,IRS(1)/IRN)
      IRN=0
      DO 3510 IK=1,3
 3510 IRS(IK)=0
 4000 CONTINUE
      IF(ND.EQ.0)GO  TO 4010
      CALL COR2F(2,NBD,NTD)
      CALL COR2(1,ND,NTD)
      ND=0
 4010 CONTINUE
CCCC  EVERY SAMPLE PROCESSING AND SWITCHES UPDATES
      NOS=0
      DO 4100 N=1,NST
      IF(VSOFF(N).NE.0)NS(N)=-NT
      IF(VSON(N).NE.0)NS(N)=NT
      IF(VFOFF(N).NE.0)NF(N)=-NT
      IF(VFON(N).NE.0)NF(N)=NT
      IF(NS(N).GT.0)NOS=NOS+1
      IF(NS(N).GT.0)ITS(N)=ITS(N)+1
      IF(NF(N).GT.0)ITF(N)=ITF(N)+1
 4100 VSOFF(N)=VSON(N)=VFOFF(N)=VFON(N)=0
      IF(NOS.EQ.0)NS0=NS0+1
      IF(NOS.EQ.1)NS1=NS1+1
      IF(NOS.GT.1)NS2=NS2+1
      RETURN
CCCC  INITIALIZE SWITCHES AND LOCAL VARS
      ENTRY VSINIT
      KK=0
      DO 5000 N=1,NST
      VSON(N)=VSOFF(N)=VFON(N)=VFOFF(N)=0
```

```
      NF(N)=NS(N)=NFC(N)=NH(N)=Ø
      DO 5ØØØ NB=1,NST
 5ØØØ NSI(N,NB)=Ø
      DO 5Ø1Ø IK=1,3
 5Ø1Ø IRS(IK)=Ø
      IRN=Ø
      ND=Ø
      RETURN
CCCC  LAST SAMPLE
      ENTRY VLSAMPLE
      DO 6ØØØ N=1,NST
      IF(NS(N).LE.Ø)GO TO 51ØØ
      CALL VAR1(3,N,NT-NS(N))
      GO TO 52ØØ
 51ØØ IF(NF(N).GT.Ø)CALL VAR1(1,N,NT+NS(N))
      IF(NF(N).LE.Ø)CALL VAR1(2,N,NT+NS(N))
 52ØØ IF(NF(N).GT.Ø)CALL VAR1(5,N,NT-NF(N))
      IF(NF(N).GT.Ø)CALL COR2(5,N,NT-NF(N))
      IF(NF(N).LE.Ø)CALL VAR1(6,N,NT+NF(N))
 6ØØØ CONTINUE
      RETURN
      END
C
CCCC  CUMULATIVE STATS AND OUTPUT ROUTINES
C
      SUBROUTINE VINIT
      DIMENSION TITLE(15)
      DIMENSION NODE(8)
      COMMON /TITLE/ TITLE, NODE
      INTEGER TFO(8)
      DIMENSION IN(15,8),ISVAR(15,8),ISSVAR(15,8),NFO(8)
      DIMENSION IHVAR(15,8,81),IHRES(15),VN(15,8)
      DIMENSION IN2(4,8,8),ISVAR2(4,8,8),ISSVAR2(4,8,8)
      DIMENSION ITS(8),ITF(8),VOUT(15)
      DATA IHRES/2,1Ø,2,2Ø,2Ø,2Ø,2Ø,1,1,1,1,2,1,5,5/
      DATA ISW/Ø/
      COMMON /ASVAR/ ITS,ITF,NSØ,NS1,NS2
      COMMON /ASDAT/ IN2,ISVAR2,ISSVAR2
      COMMON /ASRUN/ NS,NT
CCCC  INITIALIZE RUNNING STATS STORAGE ARRAYS
      DO 4Ø N=1,NS
      DO 2Ø I=1,15
      DO 1Ø J=1,81
   1Ø IHVAR(I,N,J)=Ø
      ISVAR(I,N)=Ø
      ISSVAR(I,N)=Ø
   2Ø IN(I,N)=Ø
      ITS(N)=Ø
      ITF(N)=Ø
      DO 3Ø I=1,4
      DO 3Ø NB=1,NS
      ISVAR2(I,N,NB)=Ø
```

```
      ISSVAR2(I,N,NB)=Ø
   3Ø IN2(I,N,NB)=Ø
   4Ø CONTINUE
      NSØ=NS1=NS2=Ø
   32 FORMAT(I1)
      IF(ISW.GE.Ø)RETURN
      TYPE 51
   51 FORMAT(' HISTOGRAMS? 1 - YES')
      ACCEPT 32,ISW
      RETURN
CCCC  RUNNING STATS OF ONE DIMENSION VARS
      ENTRY VAR1(IV,NP,IVAL)
C     IF(IV.LT.8)PRINT 991,IV,NP,IVAL
C 991 FORMAT(' VAR1 ',3I8)
      IN(IV,NP)=IN(IV,NP)+1
      ISVAR(IV,NP)=ISVAR(IV,NP)+IVAL
      ISSVAR(IV,NP)=ISSVAR(IV,NP)+IVAL*IVAL
      IF(ISW.EQ.Ø)RETURN
      IH=IVAL/IHRES(IV)+1
      IF(IH.GT.81)IH=81
      IHVAR(IV,NP,IH)=IHVAR(IV,NP,IH)+1
      RETURN
CCCC  PRINTING
      ENTRY VSTAT
   8Ø IF(ISW.NE.1)GO TO 2ØØ
      DO 1ØØ I=1,15
      PRINT 99
      IF(I.EQ.Ø1)PRINT 1Ø1
      IF(I.EQ.Ø2)PRINT 1Ø2
      IF(I.EQ.Ø3)PRINT 1Ø3
      IF(I.EQ.Ø4)PRINT 1Ø4
      IF(I.EQ.Ø5)PRINT 1Ø5
      IF(I.EQ.Ø6)PRINT 1Ø6
      IF(I.EQ.Ø7)PRINT 1Ø7
      IF(I.EQ.Ø8)PRINT 1Ø8
      IF(I.EQ.Ø9)PRINT 1Ø9
      IF(I.EQ.1Ø)PRINT 11Ø
      IF(I.EQ.11)PRINT 111
      IF(I.EQ.12)PRINT 112
      IF(I.EQ.13)PRINT 113
      IF(I.EQ.14)PRINT 114
      IF(I.EQ.15)PRINT 115
      VR=Ø
      DO 9Ø J=1,81
      IF(I.EQ.7 .AND. J.GE.52)GO TO 1ØØ
      PRINT 91, VR,(IHVAR(I,N,J),N=1,NS)
   9Ø VR=VR+IHRES(I)*.1
   91 FORMAT(5X,F7.1,8X,8I8)
   99 FORMAT('1')
  1ØØ CONTINUE
  1Ø1 FORMAT(' PAUSES')
  1Ø2 FORMAT(' SILENCES')
```

```
103 FORMAT(' SPEECHES')
104 FORMAT(' FLOOR CYCLES')
105 FORMAT(' FLOOR TIMES')
106 FORMAT(' NON FLOOR TIMES')
107 FORMAT(' HESITANCIES')
108 FORMAT(' UNSUCCESSFUL INTERRUPTS BY')
109 FORMAT(' UNSUCCESSFUL INTERRUPTS TO')
110 FORMAT(' SUCCESSFUL INTERRUPTS BY')
111 FORMAT(' SUCCESSFUL INTERRUPTS TO')
112 FORMAT(' RESPONSES BY')
113 FORMAT(' RESPONSES TO')
114 FORMAT(' CHALLENGES BY')
115 FORMAT(' CHALLENGES TO')
200 VNT=NT*.1
201 FORMAT('1',20X,15A4//40X,'ANALYSIS FROM'
   # ,F8.1,' TO',F8.1,' SECONDS')
    NTT=NS0+NS1+NS2
    VNTT=VNT-NTT*.1
    PRINT 201,TITLE,VNTT,VNT
    CALL DOUT
    VNT=NTT*.1
    VNS0=NS0*.1
    VNS1=NS1*.1
    VNS2=NS2*.1
    PRINT 202, VNT,VNS0,VNS1,VNS2
202 FORMAT(' TOTAL TIME =',F7.1//
   # ' NO SPEAKERS =',F7.1/
   # ' ONE SPEAKER =',F7.1/
   # ' TWO OR MORE SPEAKERS =',F7.1)
    PRINT 209,(NODE(N),N=1,NS)
209 FORMAT(//5X,'SPEAKER',8X,8I8)
    PRINT 203
203 FORMAT(//' TOTAL SPEECH TIME')
    DO 210 N=1,NS
210 VOUT(N)=ITS(N)*.1
    PRINT 204,(VOUT(N),N=1,NS)
204 FORMAT(20X,8F8.1)
    PRINT 206
206 FORMAT (' TOTAL NON-SPEECH TIME')
    DO 215 N=1,NS
215 VOUT(N)=(NTT-ITS(N))*.1
    PRINT 204, (VOUT(N),N=1,NS)
    PRINT 205
205 FORMAT(' TOTAL FLOOR TIME')
    DO 220 N=1,NS
220 VOUT(N)=ITF(N)*.1
    PRINT 204,(VOUT(N),N=1,NS)
    PRINT 208
208 FORMAT(' PERCENT SPEECH')
    DO 230 N=1,NS
230 VOUT(N)=ITS(N)*100./NTT
    PRINT 207, (VOUT(N),N=1,NS)
```

```
207 FORMAT(2Ø X,8F8.2/)
    PRINT 27Ø
27Ø FORMAT(' PERCENT FLOOR TIME')
    DO 271 N=1,NS
271 VOUT(N)=ITF(N)*1ØØ./NTT
    PRINT 2Ø7,(VOUT(N),N=1,NS)
    DO 3ØØ I=1,15
    IF(I.EQ.Ø1)PRINT 1Ø1
    IF(I.EQ.Ø2)PRINT 1Ø2
    IF(I.EQ.Ø3)PRINT 1Ø3
    IF(I.EQ.Ø4)PRINT 1Ø4
    IF(I.EQ.Ø5)PRINT 1Ø5
    IF(I.EQ.Ø6)PRINT 1Ø6
    IF(I.EQ.Ø7)PRINT 1Ø7
    IF(I.EQ.Ø8)PRINT 1Ø8
    IF(I.EQ.Ø9)PRINT 1Ø9
    IF(I.EQ.1Ø)PRINT 11Ø
    IF(I.EQ.11)PRINT 111
    IF(I.EQ.12)PRINT 112
    IF(I.EQ.13)PRINT 113
    IF(I.EQ.14)PRINT 114
    IF(I.EQ.15)PRINT 115
    DO 25Ø N=1,NS
    NN=IN(I,N)
    VOUT(N)=Ø
    IF(NN.GT.Ø)VOUT(N)=ISVAR(I,N)*.1/NN
25Ø CONTINUE
    PRINT 251,(VOUT(N),N=1,NS)
251 FORMAT(5X,'MEAN',11X,8F8.2)
    DO 26Ø N=1,NS
    NN=IN(I,N)
    VOUT(N)=Ø
    IF(NN.GT.1)VOUT(N)=SQRT((1.*NN*ISSVAR(I,N)-
   # (ISVAR(I,N)*1.)**2)/(NN**2-NN))*.1
26Ø CONTINUE
    PRINT 261,(VOUT(N),N=1,NS)
261 FORMAT(5X,'STD DEV',8X,8F8.2)
    PRINT 262,(IN(I,N),N=1,NS)
262 FORMAT(5X,'N',14X,8I8)
3ØØ CONTINUE
    PRINT 3Ø5
    DO 5ØØ I=1,4
    DO 4ØØ N=1,NS
    IF(I.EQ.1)PRINT 3Ø1,NODE(N)
    IF(I.EQ.2)PRINT 3Ø2,NODE(N)
    IF(I.EQ.3)PRINT 3Ø3,NODE(N)
    IF(I.EQ.4)PRINT 3Ø4,NODE(N)
3Ø1 FORMAT(' UNSUCCESSFUL INTERRUPTS BY',I2,' TO ...')
3Ø2 FORMAT(' SUCCESSFUL INTERRUPTS BY',I2,' TO ...')
3Ø3 FORMAT(' RESPONSES BY',I2,' TO ...')
3Ø4 FORMAT(' CHALLENGES BY' ,I2,' TO ...')
3Ø5 FORMAT('1TWO-DIMENSIONAL VARIABLES'//)
```

```
      DO 350 NB=1,NS
      NN=IN2(I,N,NB)
      VOUT(NB)=0
      IF(NN.EQ.0)GO TO 350
      VOUT(NB)=ISVAR2(I,N,NB)*.1/NN
  350 CONTINUE
      PRINT 251,(VOUT(NB),NB=1,NS)
      DO 360 NB=1,NS
      NN=IN2(I,N,NB)
      VOUT(NB)=0
      IF(NN .LE.1)GO TO 360
      VOUT(NB)=SQRT((1.*NN*ISSVAR2(I,N,NB)-
     # (ISVAR2(I,N,NB)*1.)**2)/(NN**2-NN))*.1
  360 CONTINUE
      PRINT 261,(VOUT(NB),NB=1,NS)
      PRINT 262,(IN2(I,N,NB),NB=1,NS)
  400 CONTINUE
  500 CONTINUE
CCCC  PUNCHING
      DO 700 N=1,NS
      VOUT(1)=VNT
      VOUT(2)=VNS0
      VOUT(3)=VNS1
      VOUT(4)=VNS2
      VOUT(5)=ITS(N)*.1
      VOUT(7)=ITF(N)*.1
      VOUT(8)=(VOUT(7)/VNT)*100.
      VOUT(9)=ITS(N)*100./NTT
      VOUT(6)=(NTT-ITS(N))*.1
      VOUT8=VOUT(8)
      VOUT9=VOUT(9)
      PUNCH 601,NODE(N),(VOUT(I),I=1,7)
  601 FORMAT(I2,9X,7F9.2)
      DO 620 I=1,15
      VOUT(I)=0
      IF(IN(I,N).NE.0)VOUT(I)=ISVAR(I,N)*.1/IN(I,N)
  620 CONTINUE
      VN(1,N)=(IN(1,N)/ITF(N))*1000.
      DO 701 K=2,6
      VN(K,N)=(IN(K,N)/VNT)*1000.
  701 CONTINUE
      VN(7,N)=IN(7,N)
      DO 703 I=1,NS
      TFO(I)=0
      NFO(I)=0
      DO 704 K=1,NS
      IF(K.EQ.I)GO TO 704
      TFO(I)=TFO(I)+ITF(K)
  704 CONTINUE
      DO 707 K=1,NS
      IF(K.EQ.I)GO TO 707
      NFO(I)=NFO(I)+IN(5,K)
```

```
  707 CONTINUE
  703 CONTINUE
      VN(8,N)=(IN(8,N)/TFO(N))*10000.
      DO 705 K=10,14,2
      VN(K,N)=(IN(K,N)/NFO(N))*1000.
  705 CONTINUE
      IF(ITF(N).EQ.0)GO TO 709
      VN(9,N)=(IN(9,N)/ITF(N))*10000.
      GO TO 710
  709 VN(K,N)=0.
  710 CONTINUE
      DO 706 K=11,15,2
      IF(IN(5,N).EQ.0)GO TO 708
      VN(K,N)=(IN(K,N)/IN(5,N))*1000.
      GO TO 706
  708 VN(K,N)=0.
  706 CONTINUE
      PUNCH 602,VOUT9,VOUT8,(VOUT(I),VN(I,N),I=1,15)
  602 FORMAT(2X,8F9.2)
      PUNCH 717,(IN(I,N),I=1,15)
  717 FORMAT(5X,15I5)
  700 CONTINUE
      RETURN
      END
CCCC RUNNING STATS ON TWO DIMENSION VARS
      SUBROUTINE VAR2(IV,NP,NQ,IVAL)
C     PRINT 991,IV,NP,NQ,IVAL
C 991 FORMAT(' VAR2 ',4I8)
      DIMENSION IN2(4,8,8),ISVAR2(4,8,8),ISSVAR2(4,8,8)
      COMMON /ASDAT/ IN2,ISVAR2,ISSVAR2
      IF(NP.EQ.NQ) RETURN
      IN2(IV,NP,NQ)=IN2(IV,NP,NQ)+1
      ISVAR2(IV,NP,NQ)=ISVAR2(IV,NP,NQ)+IVAL
      ISSVAR2(IV,NP,NQ)=ISSVAR2(IV,NP,NQ)+IVAL*IVAL
      CALL VAR1(6+2*IV,NP,IVAL)
      CALL VAR1(7+2*IV,NO,IVAL)
      RETURN
      END
C
CCCC RUNNING STATS ON CORRELATIONS
C
      SUBROUTINE CINIT
      DIMENSION TITLE(15),NODE(8)
      COMMON /TITLE/ TITLE,NODE
      DATA ISW /-1/
      DIMENSION ISN(7,8,8),ISX(7,8,8),ISY(7,8,8),ISXX(7,8,8)
      DIMENSION ISYY(7,8,8),ISXY(7,8,8)
      DIMENSION IN(14,8),IX(14,8),IY(14,8),IXX(14,8),IYY(14,8),IXY(14,8)
      DIMENSION ISNP(7),ISXF(7),INP(14),IXF(14)
      COMMON /ASRUN/ NS
      DIMENSION VOUT(8),T(8),IDF(8)
      DO 30 I=1,7
```

```
      DO 2Ø J=1,NS
      DO 2Ø K=1,NS
      ISN(I,J,K)=ISX(I,J,K)=ISY(I,J,K)=Ø
   2Ø ISXX(I,J,K)=ISYY(I,J,K)=ISXY(I,J,K)=Ø
   3Ø ISNP(I)=ISXF(I)=Ø
      DO 5Ø I=1,14
      DO 4Ø J=1,NS
   4Ø IN(I,J)=IX(I,J)=IY(I,J)=IXX(I,J)=IYY(I,J)=IXY(I,J)=Ø
   5Ø INP(I)=IXF(I)=Ø
      IF(ISW.NE.-1)RETURN
      TYPE 51
   51 FORMAT (' CORRELATIONS? 1 - YES')
      ACCEPT 52, ISW
   52 FORMAT(I1)
      RETURN
      ENTRY COR2F(IV,NP,IVA)
C     PRINT 991, IV,NP,IVA
C 991 FORMAT(45X,' COR2F',3I8)
      ISNP(IV)=NP
      ISXF(IV)=IVA
      RETURN
      ENTRY COR2(IV,NQ,IVB)
C     PRINT 992,IV,NQ,IVB
C 992 FORMAT(45X,' COR2 ',3I8)
      IF(ISW.EQ.Ø)RETURN
      N=ISNP(IV)
      ISNP(IV)=Ø
      IF(N.EQ.Ø.OR.NQ.EQ.Ø)RETURN
      IF(N.EQ.NQ)RETURN
      JX=ISXF(IV)
      ISN(IV,N,NQ)=ISN(IV,N,NQ)+1
      ISX(IV,N,NQ)=ISX(IV,N,NQ)+JX
      ISY(IV,N,NQ)=ISY(IV,N,NQ)+IVB
      ISXX(IV,N,NQ)=ISXX(IV,N,NQ)+JX*JX
      ISYY(IV,N,NQ)=ISYY(IV,N,NQ)+IVB*IVB
      ISXY(IV,N,NQ)=ISXY(IV,N,NQ)+JX*IVB
      I=IV*2-1
      IN(I,N)=IN(I,N)+1
      IX(I,N)=IX(I,N)+JX
      IY(I,N)=IY(I,N)+IVB
      IXX(I,N)=IXX(I,N)+JX*JX
      IYY(I,N)=IYY(I,N)+IVB*IVB
      IXY(I,N)=IXY(I,N)+JX*IVB
      I=IV*2
      IN(I,NQ)=IN(I,NQ)+1
      IX(I,NQ)=IX(I,NQ)+JX
      IY(I,NQ)=IY(I,NQ)+IVB
      IXX(I,NQ)=IXX(I,NQ)+JX*JX
      IYY(I,NQ)=IYY(I,NQ)+IVB*IVB
      IXY(I,NQ)=IXY(I,NQ)+JX*IVB
      RETURN
      ENTRY COR1F(IV,NP,IVA)
```

```
      INP(IV)=NP
      IXF(IV)=IVA
      RETURN
      ENTRY COR1(IV,NQ,IVB)
      IF(ISW.EQ.Ø)RETURN
      N=INP(IV)
      INP(IV)=Ø
      IF(N.EQ.Ø)RETURN
      IF(N.NE.NQ)RETURN
      JX=IXF(IV)
      IN(IV,N)=IN(IV,N)+1
      IX(IV,N)=IX(IV,N)+JX
      IY(IV,N)=IY(IV,N)+IVB
      IXX(IV,N)=IXX(IV,N)+JX*JX
      IYY(IV,N)=IYY(IV,N)+IVB*IVB
      IXY(IV,N)=IXY(IV,N)+JX*IVB
      RETURN
CCCC PRINTING
      ENTRY CSTAT
      IF (ISW.LE.Ø)RETURN
  1Ø1 FORMAT(' PAUSES BY ...  WITH FOLLOWING RESPONSES')
  1Ø2 FORMAT(' RESPONSES BY ...  WITH PRECEDING PAUSES')
  1Ø3 FORMAT(' RESPONSES TO ...  WITH FOLLOWING PAUSES')
  1Ø4 FORMAT(' PAUSES BY ...  WITH PRECEDING RESPONSES')
  1Ø5 FORMAT(' SPEECHES BY ...  WITH FOLLOWING SPEECHES')
  1Ø6 FORMAT(' SPEECHES BY ...  WITH PRECEDING SPEECHES')
  1Ø7 FORMAT(' FLOOR CYCLES BY ...  WITH FOLLOWING FLOOR CYCLES')
  1Ø8 FORMAT(' FLOOR CYCLES BY ...  WITH PRECEDING FLOOR CYCLES')
  1Ø9 FORMAT(' FLOOR TIMES BY ...  WITH FOLLOWING FLOOR TIMES')
  11Ø FORMAT(' FLOOR TIMES BY ...  WITH PRECEDING FLOOR TIMES')
  111 FORMAT(' PAUSES BY ...  WITH FOLLOWING PAUSES')
  112 FORMAT(' PAUSES BY ...  WITH PRECEDING PAUSES')
  113 FORMAT(' HESITANCIES BY ...  WITH FOLLOWING HESITANCIES')
  114 FORMAT(' HESITANCIES BY ...  WITH PRECEDING HESITANCIES')
  2Ø1 FORMAT(' PAUSES BY',I2,' WITH FOLLOWING RESPONSES BY ...')
  2Ø2 FORMAT(' RESPONSES TO',I2,' WITH FOLLOWING PAUSES BY ...')
  2Ø3 FORMAT(' SPEECHES BY',I2,' WITH FOLLOWING SPEECHES BY ...')
  2Ø4 FORMAT(' FLOOR CYCLES BY',I2,
     # ' WITH FOLLOWING FLOOR CYCLES BY ...')
  2Ø5 FORMAT(' FLOOR TIMES BY',I2,' WITH FOLLOWING FLOOR TIMES BY ...')
  2Ø6 FORMAT(' PAUSES BY',I2,' WITH FOLLOWING PAUSES BY ...')
  2Ø7 FORMAT(' HESITANCIES BY',I2,' WITH FOLLOWING HESITANCIES BY ...')
      PRINT 61
   61 FORMAT('1TRANSITION CORRELATIONS'//)
      DO 5ØØ I=1,7
      DO 4ØØ N=1,NS
      IF(I.EQ.1)PRINT 2Ø1,NODE(N)
      IF(I.EQ.2)PRINT 2Ø2,NODE(N)
      IF(I.EQ.3)PRINT 2Ø3,NODE(N)
      IF(I.EQ.4)PRINT 2Ø4,NODE(N)
      IF(I.EQ.5)PRINT 2Ø5,NODE(N)
      IF(I.EQ.6)PRINT 2Ø6,NODE(N)
```

```
      IF(I.EQ.7)PRINT 207,NODE(N)
      DO 350 NB=1,NS
      VOUT(NB)=0
      T(NB)=0
      NN=ISN(I,N,NB)
      IDF(NB)=NN
      IF(NN.LE.2) GO TO 350
      R=0.+NN*ISXY(I,N,NB)-ISX(I,N,NB)*ISY(I,N,NB)
      R=R/SQRT((NN*1.*ISXX(I,N,NB)-(0.+ISX(I,N,NB))**2)*
     # (NN*1.*ISYY(I,N,NB)-(0.+ISY(I,N,NB))**2))
      VOUT(NB)=R
      IF(R.GE.1.)R=.99999
      IF(R.LE.-1.)R=-.99999
      T(NB)=ABS(R*SQRT((NN-2)/(1.-R*R)))
  350 CONTINUE
      PRINT 351,(VOUT(NB),NB=1,NS)
  351 FORMAT(5X,'CORR',11X,9F8.4)
      PRINT 352,(T(NB),NB=1,NS)
  352 FORMAT(5X,'T',14X,9F8.3)
      PRINT 353,(IDF(NB),NB=1,NS)
  353 FORMAT(5X,'N ',13X,9I8)
  400 CONTINUE
  500 CONTINUE
      PRINT 71
   71 FORMAT ('1CORRELATIONS'//)
      DO 700 I=1,14
      IF(I.EQ.01) PRINT 101
      IF(I.EQ.2)PRINT 102
      IF (I.EQ.3) PRINT 103
      IF(I.EQ.4)PRINT 104
      IF(I.EQ.5)PRINT 105
      IF(I.EQ.6)PRINT 106
      IF(I.EQ.7)PRINT 107
      IF(I.EQ.8)PRINT 108
      IF(I.EQ.9)PRINT 109
      IF(I.EQ.10)PRINT 110
      IF(I.EQ.11)PRINT 111
      IF(I.EQ.12)PRINT 112
      IF(I.EQ.13)PRINT 113
      IF(I.EQ.14)PRINT 114
      IS=0
      SX=SY=SXX=SXY=SYY=0.
      DO 550 N=1,NS
      IS=IS+IN(I,N)
      SX=SX+IX(I,N)
      SY=SY+IY(I,N)
      SXX=SXX+IXX(I,N)
      SXY=SXY+IXY(I,N)
      SYY=SYY+IYY(I,N)
      VOUT(N)=0
      T(N)=0
      NN=IN(I,N)
```

```
      IDF(N)=NN
      IF(NN.LE.2)GO TO 550
      R=0.+NN*IXY(I,N)-IX(I,N)*IY(I,N)
      R=R/SQRT((NN*1.*IXX(I,N)-(0.+IX(I,N))**2)*
     # (NN*1.*IYY(I,N)-(0.+IY(I,N))**2))
      VOUT(N)=R
      IF(R.GE.1.)R=.99999
      IF(R.LE.-1.)R=-.99999
      T(N)=ABS(R*SQRT((NN-2)/(1.-R*R)))
  550 CONTINUE
      IF((I/2)*2 .NE. I)GO TO 600
      IF(IS.LE.2)GO TO 600
      SR=(IS*SXY-SX*SY)/SQRT((IS*SXX-SX*SX)*(IS*SYY-SY*SY))
      ST=ABS(SR*SQRT((IS-2)/(1.-SR*SR)))
      PRINT 351,(VOUT(N),N=1,NS),SR
      PRINT 352,(T(N),N=1,NS),ST
      PRINT 353,(IDF(N),N=1,NS),IS
      GO TO 700
  600 PRINT 351,(VOUT(N),N=1,NS)
      PRINT 352,(T(N),N=1,NS)
      PRINT 353,(IDF(N),N=1,NS)
  700 CONTINUE
      RETURN
      END
```

APPENDIX N

DUNSPEEL SOURCE LISTING

```
C
CCCC DUNSPEEL
C
      SUBROUTINE DINIT
      INTEGER OUTR
      DATA ISW/-1/
      COMMON /DSP/ NS
      DIMENSION TITLE(15),NODE(8)
      COMMON /TITLE/ TITLE,NODE
      DOUBLE PRECISION SN,SINC
      DIMENSION LINE(104),IDEL(8,8),DEL(8,8)
      DIMENSION K(250),KK(500),ITR(8)
CCCC INPUT OF DELAY ARRAYS
      IF (ISW.EQ.1)GO TO 200
      IF(ISW.EQ.0)STOP  END OF ANALYSIS
      TYPE 17
   17 FORMAT(' TYPE 1 FOR DELAY DATA')
      ACCEPT 19,ISW
   19 FORMAT(I1)
      INR=33
      IEOF=0
      IF(ISW.EQ.0)GO TO 210
C     TYPE 1
C   1 FORMAT(' INPUT RATE MSEC - XXX')
C     ACCEPT 11,INR
   11 FORMAT(I3,7I4)
      OUTR=100
      DO 20 I=1,NS
      TYPE 2,I
    2 FORMAT(' DELAYS FROM',I2,' TO .. IN MSEC - XXX XXX ..')
   20 ACCEPT 11,(IDEL(I,J),J=1,NS)
      TYPE 4
    4 FORMAT(' DELAYS TO TAPE FROM .. IN MSEC - XXX XXX ..')
      ACCEPT 11,ITR
      PRINT 5,INR,OUTR
    5 FORMAT(' INPUT RATE = ',I3,5X,'OUTPUT RATE = ',I3)
      PRINT 6
    6 FORMAT(/' DELAY ARRAY')
      DO  30 I=1,NS
   30 PRINT 7,I,(IDEL(I,J),J=1,NS)
    7 FORMAT(5X,'FROM',I2,' TO ..',3X,8I8)
      PRINT 8,(ITR(I),I=1,NS)
    8 FORMAT(5X,'DELAY TO TAPE',2X,8I8)
CCCC COMPUTE ACTUAL DELAYS FOR J TO ANALYSE FROM I'S PT OF VIEW
      MIN=0
      DO 80 I=1,NS
      DO 80 J=1,NS
      IDEL(J,I)=IDEL(J,I)-ITR(J)
   80 IF(MIN .GT. IDEL(J,I))MIN=IDEL(J,I)
      MAX=0
      DO 90 I=1,NS
      DO 90 J=1,NS
```

```
      IDEL(J,I)=IDEL(J,I)-MIN
      IF(MAX.LT.IDEL(J,I))MAX=IDEL(J,I)
   90 DEL(I,J)=IDEL(J,I)*1./INR
      PRINT 9
    9 FORMAT(' ACTUAL DELAYS APPLIED TO DATA')
      DO 100 I=1,NS
  100 PRINT 3,I,(IDEL(J,I),J=1,NS)
    3 FORMAT(5X,I1,'''S PT OF VIEW ',8I8)
      DO 120 I=1,NS
      DO 110 J=1,NS
      IF(NODE(J).EQ.I)GO TO 120
  110 CONTINUE
      ITR(I)=9999
  120 CONTINUE
      ISW=1
      IANA=0
      SINC=OUTR*1./INR
CCCC SETTING UP FOR ONE RUN
  200 IANA=IANA+1
      IF(IANA.GT.NS)STOP 'END OF ANALYSIS'
      IF(ITR(IANA).EQ.9999)GO TO 200
      SN=.5
      ISN=0
      KSW=0
      REWIND 5
      DO 230 I=1,104
  230 LINE(I)=0
      IEOF=MAX*1./INR+.5
      IL=IEOF+2
  210 KSW=0
      RETURN
      ENTRY DOUT
      IF(ISW.NE.1)RETURN
      PRINT 12,IANA
   12 FORMAT(/40X,'ANALYSIS FROM POINT OF VIEW OF',I2)
   13 FORMAT(40X,'         FROM POINT OF VIEW OF',I2)
      IF (IANA.EQ.NS)RETURN
      DO 260 J=IANA+1,NS
      DO 240 I=1,NS
  240 IF(IDEL(I,IANA).NE.IDEL(I,J))GO TO 250
      PRINT 13,J
      ITR(J)=9999
  250 CONTINUE
  260 CONTINUE
      RETURN
CCCC OBTAINING A SAMPLE
      ENTRY GWRD(IA)
  280 IF(KSW.EQ.0)GO TO 400
  290 IA=KK(KSW)
      IF(IA.EQ.32768)GO TO 350
      KSW=KSW+1
      IF(KSW.GE.501)KSW=0
```

```
      IF(ISW.EQ.Ø)RETURN
      IF(IA.GE.256)RETURN
300   ISN=ISN+1
      DO 31Ø I=IL,2,-1
31Ø   LINE(I)=LINE(I-1)
      LINE(1)=IA
      IF(ISN.LT.SN)GO TO 28Ø
      IA=Ø
      IM=1
      DO 32Ø I=1,NS
      J=SN+DEL(IANA,I)
      J=J-ISN+2
      IA=IA+IAND(LINE(J),IM)
32Ø   IM=IM+IM
      SN=SN+SINC
      RETURN
35Ø   IF(IEOF.LE.Ø)KSW=Ø
      IF(ISW.EQ.Ø)RETURN
      IF(IEOF.LE.Ø)RETURN
      IEOF=IEOF-1
      IA=Ø
      GO TO 3ØØ
4ØØ   CALL `BUFFERIN(5,1,K,25Ø,IC)
      IF(IC.NE.2)OUTPUT IC
      II=1
      DO 41Ø I=1,25Ø
      KK(II)=K(I)/65536
      IF(KK(II).EQ.-32768)KK(II)=32768
      II=II+1
      KK(II)=K(I)-KK(II-1)*65536
41Ø   II=II+1
      KSW=1
      GO TO 29Ø
      END
```

CRC DOCUMENT CONTROL DATA

1. **ORIGINATOR:** Department of Communications/Communications Research Centre

2. **DOCUMENT NO:** CRC Technical Note No. 692

3. **DOCUMENT DATE:** June 1978

4. **DOCUMENT TITLE:** A Fortran Program for Time Analysis of Vocal Interaction (TAVI)

5. **AUTHOR(s):** Pierre Lewis and William C. Treurniet

6. **KEYWORDS:**
   (1) Program
   (2) Vocal
   (3) Interaction

7. **SUBJECT CATEGORY (FIELD & GROUP: COSATI)**

   05 Behavioral and Social Sciences

   05 10 Psychology

8. **ABSTRACT:** The method of time analysis of vocal interaction (TAVI) was developed to facilitate automated analysis of temporal patterns of interaction between groups for up to eight nodes. Further documentation on the methodology may be found in CRC Technical Note 684. The present paper describes the organization of the Fortran computer program that computes the relevant variables from a data base created from sampling each channel at a rate of 10 Hz for the presence of a speech signal. In general, vocal interaction among up to eight locations can be analyzed. A specific application is the study of interactions via satellite. In this instance, these data are analyzed from the point of view of each node after correcting to compensate for the satellite circuit delay.

9. **CITATION:**

Government of Canada
Gouvernement du Canada