NASA-CR-135294
PWA-5554-4

METHOD OF FAN SOUND MODE STRUCTURE DETERMINATION
COMPUTER PROGRAM USER'S MANUAL
MICROPHONE LOCATION PROGRAM

by

G. F. Pickett, R. A. Wells and R. A. Love

August 1977

PRATT & WHITNEY AIRCRAFT GROUP
Commercial Products Division

(NASA-CR-135294) METHOD OF FAN SOUND MODE STRUCTURE DETERMINATION COMPUTER PROGRAM USER'S MANUAL: MICROPHONE LOCATION PROGRAM (Pratt and Whitney Aircraft Group) 72 p HC A04/MF A01 CSCL 21E G3/07 N78-17065 Unclas 05935

prepared for

NATIONAL AERONAUTICS AND SPACE ADMINISTRATION

NASA-Lewis Research Center
Contract NAS3-20047

1. Report No.	2. Government Accession No.	3. Recipient's Catalog No.
NASA CR-135294		

4. Title and Subtitle	5. Report Date
Method of Fan Sound Mode Structure Determination Computer Program User's Manual Microphone Location Program	August 1977
	6. Performing Organization Code
7. Author(s)	8. Performing Organization Report No.
G. F. Pickett, R. A. Wells and R. A. Love	PWA 5554-4
9. Performing Organization Name and Address	10. Work Unit No.
Pratt & Whitney Aircraft Group Commercial Products Division United Technologies Corporation East Hartford, Connecticut 06108	11. Contract or Grant No. NAS3-20047
	13. Type of Report and Period Covered
12. Sponsoring Agency Name and Address	Contract Report
National Aeronautics and Space Administration Washington, D. C. 20546	14. Sponsoring Agency Code

15. Supplementary Notes

16. Abstract

This computer user's manual describes the operation and the essential features of the Microphone Location Program, one of the two programs developed under the Method of Fan Sound Mode Structure Determination Program, NAS3-20047. Jointly, the two programs are used to determine the coherent modal structure in annular geometries. The purpose of the Microphone Location Program is to determine microphone locations that ensure accurate and stable results from the equation system used to calculate modal structures in the second of the two programs. As part of the computational procedure for the Microphone Location Program, a first-order measure of the stability of the equation system is indicated by a matrix "conditioning" number.

17. Key Words (Suggested by Author(s))	18. Distribution Statement
fan sound mode structures amplitude phase coherent sound modes extraneous sound modes	Unclassified – Unlimited

19. Security Classif. (of this report)	20. Security Classif. (of this page)	21. No. of Pages	22. Price*
Unclassified	Unclassified	74	

* For sale by the National Technical Information Service, Springfield, Virginia 22151

TABLE OF CONTENTS

LIST OF ILLUSTRATIONS

LIST OF TABLES

METHOD OF FAN SOUND MODE STRUCTURE DETERMINATION
COMPUTER PROGRAM USER'S MANUAL
MICROPHONE LOCATION PROGRAM

by

G. F. Pickett, R. A. Wells and R. A. Love
Pratt & Whitney Aircraft Group

1.0 SUMMARY

This computer user's manual describes the operation and the essential features of the Microphone Location Program, one of the two programs developed under the Method of Fan Sound Mode Structure Determination Program, NAS3-20047. Jointly, the two programs are used to determine the coherent modal structure in annular geometries. The purpose of the Microphone Location Program is to determine microphone locations that ensure accurate and stable results from the equation system used to calculate modal structures in the second of the two programs. As part of the computational procedure for the Microphone Location Program, a first-order measure of the stability of the equation system is indicated by a matrix "conditioning" number.

2.0 INTRODUCTION

New fan designs for modern high bypass ratio commercial engines utilize blade-vane interaction theory to the extent possible for controlling the propagation of interaction noise. Currently, this theory defines the modes that can propagate, but has not been developed to the extent that it can reliably predict the strengths of the propagating modes.

Further noise reduction could be achieved if the propagating modal structure were quantified. Once the modal structure were defined, an analytical system for acoustic-treatment design could be utilized to optimize treatment for a given modal structure, to produce more efficient schemes. In addition, the modal structure could be employed to verify developing theories of fan noise generation. To provide this capability by means of measured data, the Method of Fan Sound Mode Structure Determination Program (NAS3-20047) was undertaken. The method would be utilized until a valid fan noise generation model on a modal basis becomes available.

The theory upon which fan spinning mode theory is founded was presented in 1961 by Tyler and Sofrin (ref. 1), following extensive analytical and experimental studies. Later, Sofrin and McCann (ref. 2) derived the general form of a coherent acoustic wave in an infinitely long cylindrical duct which extended the theory to include effects of axial flow. This equation expresses the coherent acoustic pressure at locations in the duct as a function of the amplitude and phase of the propagating modes comprising the sound field. These purely coherent signals, which are due to the contributions of the constituent modes, are extracted from the overall signal by enhancement techniques adapted at Pratt & Whitney Aircraft — the advantages of utilizing signal enhancement is discussed by Posey in reference 3.

Both the analytical expression derived for a general coherent acoustic wave and a signal enhancement technique form the basis for developing a method to determine fan sound mode structures. The method, in principle, is capable of determining the amplitude and phase of all modes that can propagate at a given frequency. In practice, the number of modes that can be determined is limited by the storage capacity and the running time of the computer and by measurement and location accuracy.

The method for determining fan sound mode structure (ref. 4) requires two computer programs: a Microphone Location Program (MLP) and a Modal Calculation Program (MCP). This User's Manual describes the MLP; the MCP is presented in a companion Manual.

The MLP identifies microphone locations in the duct for measuring acoustic pressures for input to the MCP that will insure a numerically stable solution. The MCP calculates modal structures from acoustic pressure measurements and calculates coefficients that can be used to determine the sensitivity of the modal calculation procedure to first-order errors in acoustic pressure measurements and microphone placement.

In the following sections, the algorithm for microphone location and the program elements such as subroutines, functional elements, and principal element interrelationships are discussed. A description of the input parameters is included. The output format is also described and illustrated by a sample case. Finally, a listing of the program code is provided in Appendix B.

3.0 PROGRAM DESCRIPTION

3.1 ALGORITHM

The Microphone Location Program is an algorithm which systematically determines microphone locations that satisfy three criteria

1) The number of selected locations equals the number of modes.

2) The selected locations maximize the determinant associated with an equation system (ref. 4) characterized by perturbing one location while the previous locations are fixed.

3) The locations are restricted to the duct wall until the specific modes become practically indistinquishable then radial probe locations are used.

The procedure for selecting the microphone locations was developed using both a stoichiometric search procedure and a microphone acceptance procedure. Output from these procedures are the microphone locations that provide accurate and stable solutions when using matrix inversion techniques. This procedure is described in reference 4 with respect to the overall program objectives. A description of the program algorithm is provided below.

The input to the program consists of the sound field in the duct comprising N acoustic duct modes, the geometric parameters (e. g., duct radius, hub tip ratio), the test parameters (e. g., frequency, axial Mach number, speed of sound), and a region in the duct where microphones can be placed with a reference microphone location specified on the duct wall. The

N modes are then ordered such that the circumferential mode indices are arranged largest to smallest and the corresponding radial indicies are ordered from largest to smallest within each circumferential mode index.

The microphone location procedure is initiated by introducing both the first mode in the specified order and the above reference microphone location into the equation system.

The second mode in the sequential order is then introduced into the equation system and the corresponding second location is determined using a stochastic search procedure. This procedure selects a location that maximizes the absolute magnitude of the determinant associated with the resulting 2 X 2 matrix equation system. The selected location is obtained by an iterative process in the two coordinates (X,θ) on the duct wall using randomly generated locations in the region where microphones can be placed.

The first attempt generates microphone coordinates from a Gaussian distribution characterized by a mean at the middle of the search region with a standard deviation equal to half the length of the search region. Five hundred candidate microphone locations are generated within this region, and the corresponding determinant of the equation system is calculated. Only the thirty microphone locations that yield the largest magnitude of the determinant and the value of the determinant are retained. The mean and standard deviation of these thirty values are calculated and then used to restrict the next attempt to determine a suitable location.

Five hundred microphone locations are now randomly generated for a second time, but corresponding to the new statistical information. This process of redefining the statistical parameters and randomly generating five hundred locations is continued until the range of the thirty values corresponding to each microphone coordinate has converged to within a specified tolerance or until fifty iterations have been completed.

At this time, the selected microphone location — which corresponds to the largest value of the determinant calculated from the resulting 2 X 2 matrix equation system — is examined by the microphone location acceptance procedure. A conditioning number (ref. 5) is calculated as the ratio of the extreme eigen values associated with the equation system. The conditioning number is a first-order measure of the sensitivity of the equation system to small perturbations in microphone location or pressure measurement. If the conditioning number is less than or equal to a specified value (e. g., one hundred), which is input to the program, then the equation system is considered to be well conditioned in a mathematical sense. In this case, the microphone location is acceptable, and the third mode in the ordered sequence is introduced in the resulting 3 X 3 equation system. The stoichiometric search procedure is then initiated to locate a third microphone on the duct wall, using a two parameter (X,θ) search procedure.

When the conditioning number is greater than the specified value, two possibilities exist. In the first possibility, the stochastic search procedure is reinstated to obtain a suitable microphone location on a radial probe. This procedure uses a two parameter (r, θ) search, assuming a radial probe at the furthest location from the fan within the axial region. Subsequent microphones are placed on this probe using a one-parameter (r) search until the conditioning number is greater than the specified value or the maximum number of microphones per probe is satisfied. In either case, a two parameter (r, θ) search is used to select an alternate location assuming a new radial probe. The second possibility is when the selected microphone location is the first to be placed on a probe, using the above two-parameter (r, θ) search, and the conditioning number is still greater than the specified value. In this case, the location is accepted and no further microphones are permitted on the probe. Radial probes are used to locate microphones for either possibility until the maximum number of probes as specified by input is reached. When this occurs, a message is output to the effect and the program execution is terminated.

As an option, the Microphone Location Program also can be used to assess the suitability of an existing set of microphone locations to provide satisfactory input data to the Modal Calculation Program. The previously discussed stochostic search procedure and acceptance procedure are not utilized in this option. The suitability is assessed by means of the value of the conditioning number calculated from the eigen values of the equation system. If the number is less than a specified level, the equation system is considered to be well conditioned in a mathematical sense.

Experience gained from running analytical test cases has shown that a good upper bound for the conditioning number is on the order of ten. For this conditioning number, at least three significant figures are retained in the matrix inversion algorithm. To expedite the intermediate steps of locating each successive microphone, the value that is internally initialized at execution of the program in lieu of any input value has been chosen to be 100. This value can be altered by input prior to the data case if more or less accuracy is desired. However, to ensure stability, the final conditioning number should be about ten.

3.2 PROGRAM OVERVIEW

The Microphone Location Program comprises six program sections which are utilized in part or whole to accomplish the objectives of the two possible modes of operation. These six sections are:

1) Input - The input of all data is by the NAMELIST specification, and the internal parameters are initialized for program execution.

2) Characteristic Number Calculation - The characteristic numbers $k'^{\sigma}_{m\mu}$ and $Q^{\sigma}_{m\mu}$ are calculated using the procedure in Appendix A.

3) Microphone Location Determination An iterative procedure is defined for successively locating each microphone. The algorithm determines locations that maximize the determinant of an equation system characterized by perturbing one location while the previous locations are fixed. This procedure was described in Section 3.1.

4) Conditioning Number Calculation - A conditioning number (ref. 5) is calculated for the acoustic wave equation matrix after determination of each microphone location. The conditioning number is defined to be the ratio of the extreme eigen values of the matrix.

5) Microphone Location Acceptance - Acceptance of a microphone location is based on the magnitude of the conditioning number. The location procedure is reinstated when the

conditioning number is greater than a criterium specified as input. A value of 100 is internally initialized if a value is not supplied by the user. A description of the microphone acceptance procedure was presented in Section 3.1.

6) Output - All results calculated by the program are printed.

The interrelationships between the six program sections and their utility for each option is illustrated in Figure 1. As input, both options require a specific mode group, inlet geometry, and test condition to calculate characteristic numbers. One option, A, requires additional input in the form of stochastic search parameters to bound the geometry where microphone locations can be placed. Additionally, option A examines the conditioning number to determine whether the equation system associated with the introduction of each successive mode-microphone location is well conditioned in a mathematical sense. The other option B, requires that the microphone locations be specified as input either arbitrarily or as a set of existing locations. In both options, the conditioning number is calculated for the acoustic wave equation system. This number is utilized by the computer program in option A to determine whether each microphone location is acceptable for acquiring experimental data. Option B supplies this value to the user to evaluate an existing set of locations. The results from both options are printed by the output section.

3.3 PROGRAM SUBROUTINES AND FUNCTIONS DESCRIPTION

The subroutines and functions used in the six program sections presented in Section 3.2 are listed below. The purpose of each subroutine or function is described. Also, as appropriate, a principle element diagram of the more complicated sections are presented and discussed.

Input Section

The input of data to the computer program is by the NAMELIST format. This form of input is described in Section 3.4.1, and are set by specifying both the input variable name and its value. In Section 3.4.2, there is a listing of the input variable names with a corresponding description of their purpose for operation of the computer program. All input are read into the program by the following subroutine:

INPUT - This subroutine inputs data for each case and sets up the necessary internal parameters.

Characteristic Number Calculation Section

Expressions are derived in Appendix A for solving two simultaneous equations that yield the characteristic numbers $K'^{o}_{m\mu}$ and $Q^{o}_{m\mu}$. A principle element diagram is provided in Figure 2 to illustrate the functional elements that lead to a determination of these numbers. Initially, the order of the Bessel function Jm and Ym are determined from the circumferential order of a particular mode. The Jm and Ym Bessel functions are then evaluated, as appropriate depending on the value of the duct hub tip ratio yielding the value of $K'^{o}_{m\mu}$ and $Q^{o}_{m\mu}$ by solving the simultaneous equations comprising these Bessel functions.

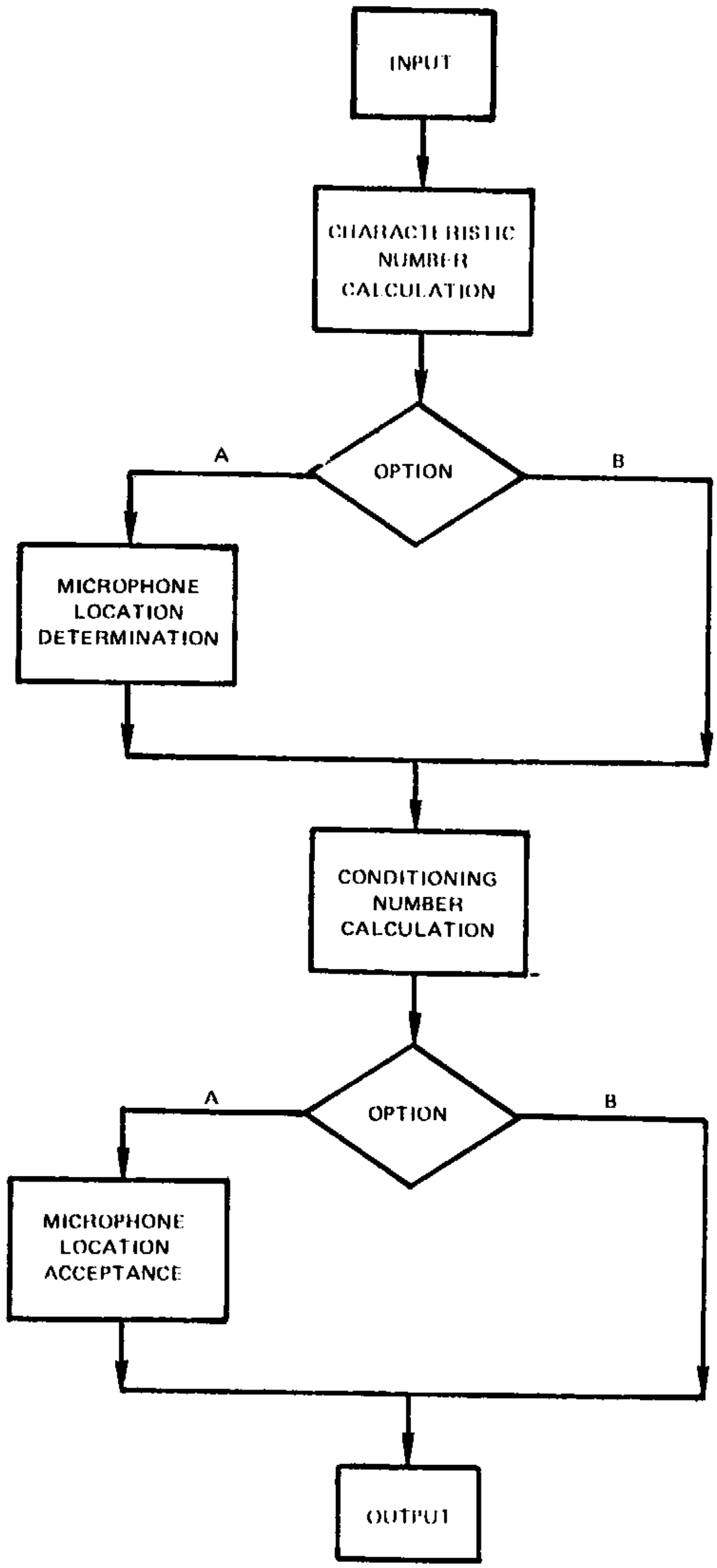

Figure 1 Program Overview

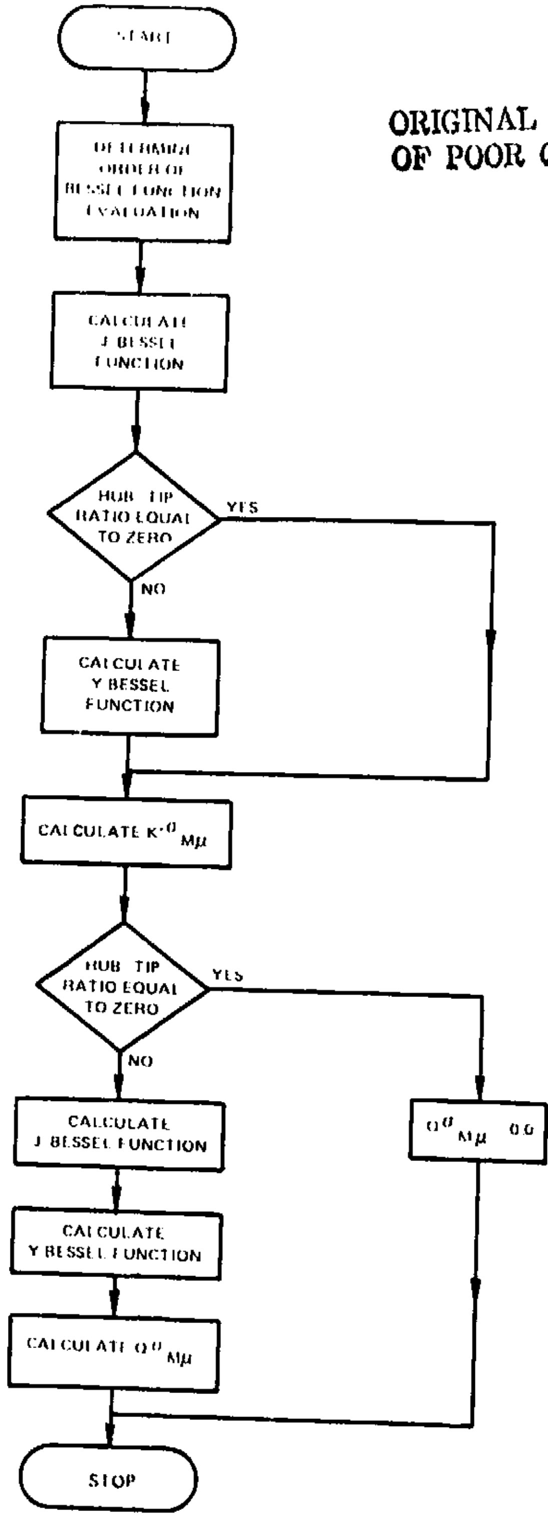

Figure 2   Principal-Element Diagram   Characteristic Number Calculation Section

The subroutines and functions that are utilized in this section are:

KQCAL - This subroutine calculates the characteristic numbers $k'^{\sigma}_{m\mu}$ and $Q^{\sigma}_{m\mu}$

KMUCAL - This subroutine is used by KQCAL to calculate the characteristic number $k'^{\sigma}_{m\mu}$.

FMUCAL - This subroutine calculates characteristic F-function values for a particular radial value $r' = r/b$.

FALZIP - This function solves for a root of a given function using a combination of false position and bisection techniques.

BESL1 - This function is used by KMUCAL to calculate values of $k'^{\sigma}_{m\mu}$ for the equation which defines the system of differential equations.

$$\frac{d}{dr'}[J_m(k'^{\sigma}_{m\mu})] + Q^{\sigma}_{m\mu}\frac{d}{dr'}\left[Y_m(k'^{\sigma}_{m\mu})\right] = 0$$

$$\frac{d}{dr'}[J_m(\sigma k'^{\sigma}_{m\mu})] + Q^{\sigma}_{m\mu}\frac{d}{dr'}\left[Y_m(\sigma k'^{\sigma}_{m\mu})\right] = 0$$

for a hub-tip ratio not equal to zero.

BESL2 - This function is used by KMUCAL to calculate values $k'^{\sigma}_{m\mu}$ for the equation which defines the above system of differential equations for a hub-tip ratio equal to zero.

BESJ - This subroutine calculates values of the Bessel function of the first kind.

BESY - This subroutine calculates values of the Bessel function of the second kind.

Microphone Location Determination Section

The algorithm for determining microphone locations was discussed in Section 3.1 and illustrated by a principle-element diagram in Figure 3. Initially, statistical information is calculated for the geometric search range by determining the mean and standard deviations of the axial, circumferential, and radial axis where microphones can be placed. The next successive microphone location is determined by restricting all previously determined locations. This location is obtained by generating five hundred locations and retaining only the values that yield the thirty largest determinates of the equation system. If the tolerance of the thirty values associated with each microphone coordinate is within the specific value, a suitable microphone location is determined. When the tolerance is greater than the specified value, the search is reinstated for at least fifty times by adjusting the statistical information to restrict the next search in the geometry associated with the largest determinates. The subroutines and functions that are utilized in this algorithm are:

STOCH - This subroutine performs a stochastic search to determine microphone locations.

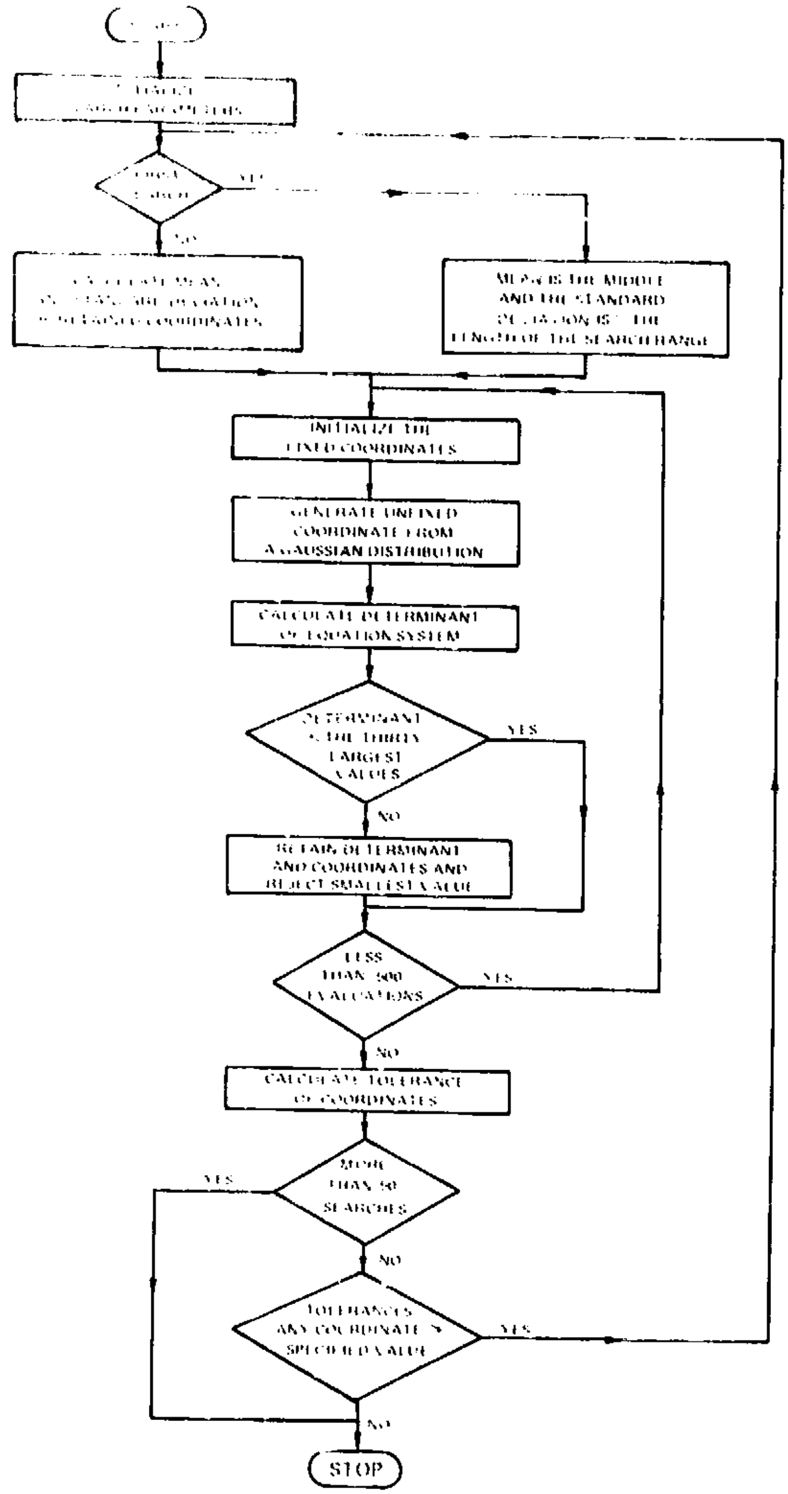

Figure 3 Principal-Element Diagram Microphone Location Determination Section

INITIAL - This subroutine is used by STOCH to initialize the necessary parameters for the stochastic search.

STAT - This subroutine is used by STOCH to generate the various statistics required for the stochastic search.

RANDOM - This subroutine is used by STOCH to generate the random values associated with the microphone locations used in an evaluation.

STVAR - This subroutine is used by RANDOM to provide random variables for either a normal or an exponential distribution.

RAND - This subroutine is used by STVAR to calculate random numbers.

DETCAL - This function is used by STOCH to calculate the determinant of the matrix equation system.

UPDATE - This subroutine is used by STOCH to update the table of determinants with the current value of the matrix determinant.

PRINTT - This subroutine is used by STOCH to print the results of the stochastic search.

CONVRG - This subroutine is used by STOCH to test for the convergence of the microphone coordinates.

Conditioning Number Calculation Section

The conditioning number (ref. 5) is determined as the functional elements illustrate in Figure 4. Initially, the eigen values of the equation system are calculated and ordered numerically. In this way, the ratio of the maximum and minimum eigen values can be computed. The subroutines that calculate the eigen values are listed below.

EIGCC, EBALAC, EHESSC, ELRH1C, ELRH2C, EBBCKC, UERTEST - These subroutines calculate the eigen values and eigen vectors of the equation system from which the condition number is calculated. This package is the property of International Mathematical and Statistical Libraries, Inc. of Houston, Texas.

Microphone Location Acceptance Section

In Section 3.1 was a discussion of the procedure for determining whether the selected microphone locations insure a numerically stable solution when used in the MPC. A principal element diagram is provided in Figure 5 to illustrate the acceptance procedure. The logic enclosed by dashed lines represents the acceptance procedure. The interrelationships between this procedure and other program sections is illustrated by the logic lines extending outside of the dashed lines. Initially, the conditioning number is compared to the acceptance value. If the conditioning number is acceptable, the microphone location procedure is reinitiated for the next mode-microphone combination. If the conditioning number is greater than the

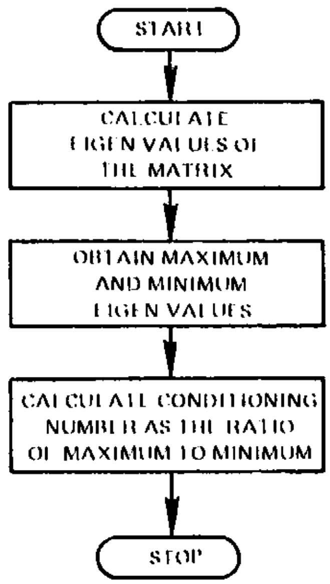

Figure 4 Principal-Element Diagram - Sensitivity Coefficient Calculation

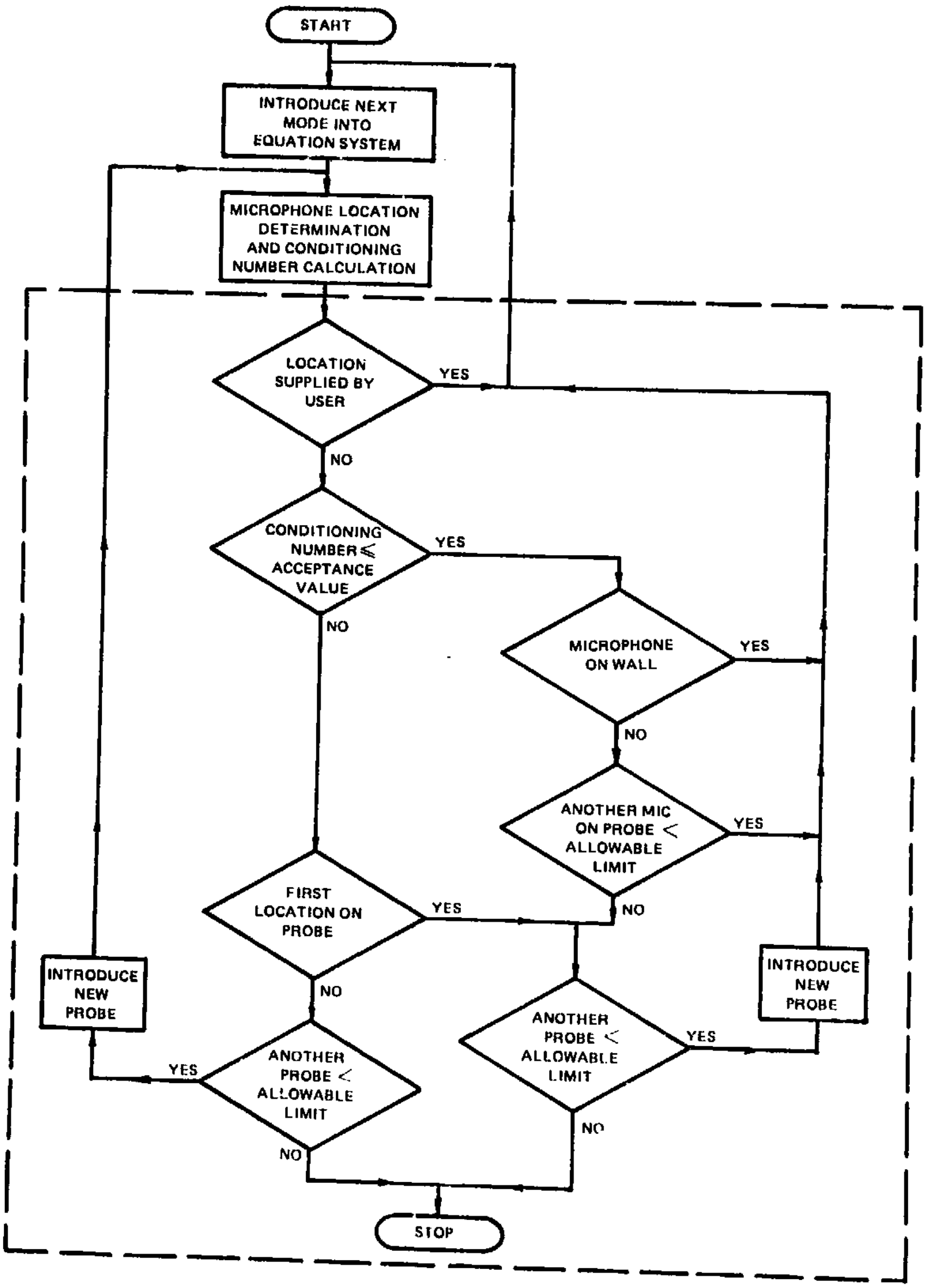

Figure 5 Principal Element Diagram Microphone Location Section

acceptance value, the microphone location procedure is reinstated to select an alternate location on a probe. Additionally, the allowable limit for microphones per probe and number of probes are checked. Since this procedure is performed in the MAIN, there are no subroutines or functions to list.

Output Section

The output format and variables from operating the MLP are discussed in Section 3.5.1. A sample case is presented in Section 3.5.2 to illustrate the execution of a case comprising three propagating modes. Both sections address the two possible modes of operation that are executable with this program. Results from the computational procedure are printed by the subroutine listed below after all angles are converted to within the range 0° to 360°.

PRINT   This subroutine prints input and output values.

ANGPOS   This subroutine converts negative angles to positive angles in the range 0° to 360° for printing.

3.4 INPUT DESCRIPTION

3.4.1 Input Format

The NAMELIST format is used to input data into the Microphone Location Program and consists of a list of parameter names grouped under an identifying name: &INDATA. The parameter names correspond to variables — single variables and matrix elements — used in the program. These variables are set by specifying both the parameter name and its value. A feature of this type of input is that all associated parameters need not be specified. Any parameter not specified in the input retains its value from the preceeding case or the default value if the input is for the first case.

NAMELIST input for each case is identified by the characteris &INDATA in Columns 2-7 of the first input card. Beginning in Column 9, parameters may be set using the format:

Parameter Name = Constant

The constant may be either a real or integer value and must be followed immediately by a comma. Parameter names, assigned values, or necessary commas must not extend beyond Column 72; and names or values cannot be continued on a subsequent card. Embedded blanks are not permitted in either the parameter name or constant value. Parameter names and their associated values may be specified in any order. The characters &END signify the end of the input for a particular case. If additional cards are required, parameter names must begin in Column 2.

A sample of this form of input for locating three microphones is presented in Figure 6.

PRATT & WHITNEY AIRCRAFT Division of UNITED TECHNOLOGIES

UTILITY CODING FORM

PWA FORM 3412 REV. 6-70

ANALYST | ADDRESS | PHONE EXT | TITLE | SHEET ___ OF ___ | JOB OR PROGRAM NO.

```
&INDATA NLOC=3, IIM(1)=1,
[illegible]=-4, -4, -4,
[illegible] 2., 2.,
[illegible]
[illegible] FRQ=[illegible], SPEED=[illegible],
[illegible], XMAX=25., TMIN=0., TMAX=[illegible],
[illegible] ELEM=11.5,
[illegible] MPROB=[illegible]
[illegible]=0. ZM(1)=25., TAM(1)=0.,
[illegible] &END
&INDATA NO=3,
XM(1)=0., 10., 20.,
ZM(1)=25., 25., 25.,
THM(1)=[illegible], 0., 0. &END
/*
//
```

Figure 6   Sample of Input Form for Locating Three Microphones

3.4.2 Input Parameters

A sign convention was adopted for assigning positive or negative values to the input parameters. Any input parameter not addressed in this discussion is a positive value. The sign convention is formulated with respect to a cylindrical coordinate system that is consistent with the derivation of the coherent acoustic wave propagation model. Its unit vectors are designated by the directions: axial - x, circumferential - θ, and radial - r.

A constant radius, annular duct is aligned with respect to this coordinate system so that the positive axial unit vector projects in a direction opposite to the flow. Thus, the Mach number of a uniform axial flow is always designated by a negative value, denoting the axial flow rate in the negative axial direction. A positive circumferential unit vector projects in the direction that the rotor spins with negative circumferential values related to the counterrotating rotor direction. Finally, the radial axis projects perpendicular to the centerline of the duct; thus radial values are always positive.

Each mode is characterized by three parameters which represent the circumferential and radial pressure distribution and its propagation direction. A specific mode is uniquely defined by the parenthetical notation (M, μ). The M defines a periodic circumferential pressure distribution with M number of lobes. Positive integers represent a corrotating M-circumferential lobe pattern with respect to the rotor direction and negative M integers refer to counterrotating modes. The radial mode index μ corresponds to the radial pressure distribution. These values are always non-negative integer numbers, with high integer values indicating large pressure variations with respect to the radius.

The modal propagation direction in an inlet or discharge duct can be either an incident wave propagating from the fan or a reflected wave propagating towards the fan. Wave propagation in a moving medium is similarly affected by the flow rate for modes that are propagating with or against the flow direction. Hence, the input variable IDIR designates wave propagation with respect to the flow direction. Positive values denote waves propagating in the opposite direction with respect to the flow, such as incident waves in the inlet duct and reflected waves in the discharge duct. Modes that propagate in the same direction as the flow are designated by a negative value for the input parameter IDIR.

The assigning of values to the input parameters will now be considered.

Since a determinative equation system is required, the number of microphone locations to be calculated is equal to the number of modes indicated by the user. To initiate the selection procedure, the input must define at least one microphone location at a reference point. In order to reduce the experimental commitment by avoiding probe-mounted microphones, the referenced location should be on the duct wall. When option B is utilized, a number of fixed microphone coordinates equal to the number specified by the input parameter ND are required.

A description of the input variables is provided in Tables I, II, and III: Table I General Input Parameters; Table II Test Geometry and Condition Input Parameters; Table III Stochastic Search Input Parameters. Under the column heading "Variable Type": the letter "R" indicates that the number is real and contains a decimal point; the letter "I" indicates the number is an integer and does not have a decimal point. "Default Values" are also delineated and indicate the value of the parameter that is internally initialized prior to the program execution. Parameters not specified in the input for the first case retain this value. Although the default values are expressed in units of the English System, the computer program can be executed with data in any consistent system of units.

TABLE I GENERAL INPUT PARAMETERS

Input Name	Variable Type	Default Value	Description
NLOC	I	2	Number of microphones or modes. (Less than or equal to fifty).
IEMU	I	0	Print indicator for characteristic E-function value. 0 = No Print 1 = Print
M(1)	I	2	Circumferential mode index.
M(2)		2	(Input NLOC values)
M(3)		0	
.		.	
.		.	
.		.	
M(50)		0	
MUS(1)	I	0	Radial mode index.
MUS(2)		1	(Input NLOC values)
MUS(3)		0	
.		.	
.		.	
.		.	
MUS(50)		0	
IDIR(1)	I	1	Mode propagation direction indicator.
IDIR(2)		1	(Input NLOC values)
IDIR(3)		0	
.		.	1 = opposite flow direction
.		.	-1 = with flow direction
.		.	
IDIR(50)		0	

TABLE II TEST GEOMETRY AND CONDITION INPUT PARAMETERS

Input Name	Variable Type	Default [a] Value	Description
HTR	R	0.438	Hub-tip ratio
OR	R	5.0	Outer radius of duct
FMX	R	0.07	Axial Mach number (always positive)
FRQ	R	3100.	Test frequency (Hertz)
SPEED	R	13566.	Speed of sound
ND	I	1	Number of initially fixed microphone locations (within range one to NLOC)
XM(1)	R	0.0	Axial coordinates of the fixed microphone locations.
XM(2)		0.0	(Input ND values)
XM(3)		0.0	
.		.	
.		.	
.		.	
XM(50)		0.0	
RM(1)	R	5.0	Radial coordinates of the fixed microphone locations.
RM(2)		0.0	(Input ND values)
RM(3)		0.0	
.		.	
.		.	
.		.	
RM(50)		0.0	
THM(1)	R	0.0	Circumferential coordinates of the fixed microphone locations. (degrees)
THM(2)		0.0	(Input ND values)
THM(3)		0.0	
.		.	
.		.	
.		.	
THM(50)		0.0	

Note: (a) Default values shown in table are in units of the English System. The program, however, is designed to execute with data in any consistent system of units.

TABLE III STOCHASTIC SEARCH INPUT PARAMETERS

Input Name	Variable Type	Default Value[a]	Description
IPT	I	0	Print indicator for table of determinants and microphone coordinates. 0 = No print 1 = Print Final Search 2 = Print every Search
NSRH	I	50	Maximum number of searches.
NVAL	I	500	Number of evaluations per search.
IXPNT	I	-2	Exponent of convergence tolerance. (e.g., Tolerance = 10^{IXPNT})
CONDNO	R	100.	Value of the conditioning number for microphone location acceptance.
NPROB	I	1	Maximum number of probes in duct.
MPROB	I	1	Maximum number of microphones per probe.
XMIN	R	0.0	Minimum axial location for microphone placement.
XMAX	R	12.0	Maximum axial location for microphone placement.
TMIN	R	0.0	Minimum angular location for microphone placement (degrees).
TMAX	R	360.	Maximum angular location for microphone placement (degrees).
DELX	R	1.0	Minimum axial distance between microphone locations.
DELR	R	1.0	Minimum radial distance between microphone locations.
DELTH	R	11.5	Minimum angular distance between microphone locations. (degrees)

Note: (a) Default values shown in table are in units of the English System. The program, however, is designed to execute with data in any consistent system of units.

3.5 OUTPUT DESCRIPTION

3.5.1 Output Format

The output from the Microphone Location Program is organized into four sections: Stochastic Search Output, Input Variables, Calculated Microphone Locations, and Characteristic F-Function. All four sections are included as output when either option is requested by the input. The computer printout for a sample case is provided in Appendix C.

The Stochastic Search Output Section comprises the value of the input parameters that defines the iterative process. These parameters include the number of iterative evaluations, tolerance for convergence, and the number of microphone coordinates used to generate statistical information. The remainder of this portion of the output includes calculated values from the stochastic search process. These values are the statistical information that corresponds to the search range for each microphone coordinate and the value of the determinant associated with the successive introduction of a mode-microphone combination in the equation system. The final output value in this section is the conditioning number of the equation system characterized by the microphone location associated with the largest value of the determinants. A print indicator is available to the user for restricting the depth of output in this section. However, the final conditioning number of the equation system is always included in the output.

Input Variable Section includes the value of various input parameters that were supplied by the user. The three indices that define the modal structure, the circumferential and radial order, and the wave direction indicator are listed. Under the Test Geometry and Conditions Input heading, there are various parameters that define the fan duct geometry and operating conditions observed during the experimental program. These parameters include the duct radius, duct hub-tip ratio, axial Mach number, and frequency. Additional stochastic search values include the axial, radial, and circumferential search ranges that bound the locations where microphones can be placed.

The Calculated Microphone Location Section includes the cylindrical coordinates of the selected microphone locations. One microphone will be located for each mode supplied by the user as input.

The final section Characteristic F-Functions, includes the value of the F-functions $F(k^{n}_{m\mu}r)$, at the measurement and prediction locations corresponding to each mode in the sequence listed in an above section. This section is a portion of the output if requested by the user at input.

3.5.2 Sample Cases

Two separate cases are presented in the sample printout to illustrate the option of whether the microphone coordinates are calucated by maximi ing the determinant of the coherent wave equation system or whether th y are specified either arbitrarily or as a set of existing microphone locations. These sample cases demonstrate the execution of each option with data listed in Figure 6. The length units in the printout are in centimeters; the time units in seconds.

The first sample case illustrates the option of determining microphone locations for a situation where three modes are propagating in a half meter diameter annular duct. Microphones at these locations can be used to obtain acoustic pressure measurements, at a frequency of 6200 Hertz, for determining the amplitude and phase of the (-4,0), (-4,1), and (-4,2) modes. Microphone placement must be restricted to within the geometric constraints of the test facility. In this situation the microphone coordinates are bounded within a 30-cm axial and a 240 degree circumferential search range. Output from the Microphone Location Program for this sample case comprises the stochastic search calculations, the input parameters, and the calculated microphone locations.

The stochastic search procedure has converged to a microphone location when the change in the determinant becomes as small as possible for small changes in microphone locations or when the maximum number of searches has been completed. In this case both microphone locations determined by the procedure converged in two searches. Upon convergence, the thirty largest values of the determinant and the corresponding microphone coordinates (axial, circumferential, and radial) are listed. Also, the conditioning number is listed for the coherent-wave equation system described by the cumulative set of microphone locations. A conditioning number of 2.5 was obtained for the three wall mounted microphones, indicating that the determinant of the equation system is well away from zero. Thus, these three locations are acceptable for calculating the modal structure by matrix inversion techniques.

A listing of the input variables followed this portion of the output. This listing includes test geometry and condition parameters, stochastic search parameters, and mode group. The intermediate values associated with the mode group are also tabulated. These values include the axial wave number in units of degrees-per-length; the eigen value, $K'o_{m,\mu}$; and the value of the eigen vector $F(Ko_{m,\mu}r)$ at the microphone locations.

The coordinates of the selected microphone locations are listed. These microphones were placed on the duct wall corresponding to a radial coordinates of 25 cm. The first location was specified at a reference location with zero cm axial and zero degrees circumferential coordinates. Two locations identified by the program are placed at the same circumferential location as the first microphone. The placement of microphones for modes with the same circumferential order is restricted to the same circumferential coordinate because the maximization of the determinant associated with the equation system is independent of this coordinate. The axial locations of the two selected microphone locations are 30 cm and 15 cm.

The second sample case illustrates the option of specifying the microphone locations. The case was executed with three flush mounted wall microphones at the same circumferential coordinate. The axial distance between each microphone was specified to be ten centimeters. This case is similar to the first case because the same mode group, frequency, and geometric configuration were used.

Output from the Microphone Location Program for this second sample case comprises the conditioning number and eigen values of the equation system associated with the successive introduction of each microphone location. A final conditioning number of 2.6 was obtained for the three microphone locations, indicating that these locations are also acceptable for determining the amplitude and phases of the (-4,0), (-4,1), (-4,2) mode structure. The input variables, microphone locations, and characteristic numbers are listed in a similar manner as the previous sample case. This option can be utilized to calculate a conditioning number for an existing set of microphone coordinates. The conditioning number for numerous groups of candidate microphone locations can be compared to allow the future user to obtain a subset of microphone locations from an existing set that is acceptable for fan sound mode structure determination.

3.6 MACHINE REQUIREMENTS

The Microphone Location Program can be compiled, linkage edited, and executed in 384K bytes of core storage.

The following mathematical functions and procedures are required.

CMPLX	Expresses two real arguments in complex form
CABS	Modulus of a complex argument
CEXP	Exponentiation of a complex argument
AIMAG	Obtain imaginary part of a complex argument
REAL	Obtain real part of a complex argument
FLOAT	Conversion from integer to real
IFIX	Conversion from real to integer
ABS	Absolute value of a real number
IABS	Absolute value of an integer
SQRT	Square root of a real value
MAX0	Obtain maximum value of input integers
ALOG	Natural logarithim of a real positive argument
SIN	Sine of a real argument
COS	Cosine of a real argument
ATAN	Arc tangent of a real argument
EXP	Exponentiation of a real argument

3.7 RESOURCE ESTIMATES

The time required for the Central Processor Unit (CPU) to process a job depends on the number of modes and the complexity of the mode group. For simple mode structures the average estimate of CPU time per mode is five seconds. For complex mode structures the average CPU time per mode is sixty seconds. Total CPU time can be reduced if the printing of internal calculations is not requested.

REFERENCES

1. Tyler, J. M. and Sofrin, T. G.: "Axial Flow Compressor Noise Studies," *Trans. SAE*, 70, p. 309, (1962)

2. Sofrin, T. G. and McCann, J. C.: "Pratt & Whitney Aircraft Experience in Compressor - Noise Reduction," For Presentation at the 72nd Meeting of Acoustical Soc. Amer., Los Angeles, CA, Nov. 1966.

3. Posey, J. W.: "Comparison of Cross-Spectral and Signal Enhancement Methods for Mapping Steady-State Acoustic Fields in Turbomachinery Ducts," NASA TM X-73916, (1976)

4. Pickett, G. F.: "Fan Sound Mode Structure Determination - Final Report," NASA CR 135293, (1977)

5. Rosanoff, R. A. and Ginsburg, T. A.: "Matrix Error Analysis for Engineers," AFFDL-TR-66-80 p. 889

6. Subroutines BESJ and BESY were adapted from the IBM Scientific Subroutine Package.

7. Subroutines, EIGCC, EBALAC, ELRH1C, ELRH2C, EBBCKC, and UERTST are the property of the International Mathematical and Statistical Libraries, Inc. of Houston, Texas.

APPENDIX A

Calculation of the Characteristic Numbers

The characteristic numbers $K'^{\sigma}_{m\mu}$ and $Q^{\sigma}_{m\mu}$ are defined to be the paired roots of the simultaneous equations

$$\left[\frac{d}{dr'} J_m (K'^{\sigma}_{m\mu} r') + Q^{\sigma}_{m\mu} \frac{d}{dr'} Y_m (K'^{\sigma}_{m\mu} r')\right]_{r' = 1} = 0 \qquad (1)$$

$$\left[\frac{d}{dr'} J_m (\sigma K'^{\sigma}_{m\mu} r') + Q^{\sigma}_{m\mu} \frac{d}{dr'} Y_m (\sigma K'^{\sigma}_{m\mu} r')\right]_{r' = 1} = 0 \qquad (2)$$

For a given circumferential mode number, m, radial order, μ, and hub/tip ratio, σ, (where σ is not equal to zero); J_m and Y_m are the Bessel functions of the first and second kinds of order m.

The following relations are used in the formulation of a solution

$$\frac{d}{dr'} J_m (x) = J'_m (x) \frac{dx}{dr'} \qquad (3)$$

$$\frac{d}{dr'} Y_m (x) = Y'_m (x) \frac{dx}{dr'} \qquad (4)$$

$$J_{m+1} (x) = \frac{2m}{x} J_m (x) - J_{m-1} (x) \qquad (5)$$

$$J_m' (x) = \frac{1}{2} [J_{m-1} (x) - J_{m+1} (x)] \qquad (6)$$

$$= \frac{1}{2} [J_{m-1} (x) - \frac{2m}{x} J_m (x) + J_{m-1} (x)]$$

$$= J_{m-1} (x) - \frac{m}{x} J_m (x)$$

$$Y'_m(x) = \frac{2}{\pi x J_m(x)} + J'_m(x)\frac{Y_m(x)}{J_m(x)}$$

$$= \frac{2}{\pi x J_m(x)} + [J_{m-1}(x) - \frac{m}{x} J_m(x)]\frac{Y_m(x)}{J_m(x)} \tag{7}$$

Letting $K = K^{'\sigma}_{m\mu}$ and $Q = Q^{\sigma}_{m\mu}$, and evaluating at $r' = 1$; (1) and (2) become

$$J'_m(K)K + Q\,Y'_m(K)K = 0 \tag{8}$$

$$J'_m(\sigma K)\sigma K + Q\,Y'_m(\sigma K)\sigma K = 0 \tag{9}$$

From (8), $Q = -\dfrac{J'_m(K)K}{Y'_m(K)K}$ substituting into (9) yields

$$J'_m(\sigma K)\sigma K - \frac{J'_m(K)K}{Y'_m(K)K}\,Y'_m(\sigma K)\sigma K = 0 \tag{10}$$

Let $f(K) = J'_m(\sigma K)\,Y'_m(K)\,\sigma K^2 - J'_m(K)\,Y'_m(\sigma K)\,\sigma K^2 = 0$ (11)

Using the expressions in (5), (6), (7), and (11) then:

$$f(K) = \sigma K^2 [J_{m-1}(\sigma K) - \frac{m}{\sigma K} J_m(\sigma K)] \left\{\frac{2}{\pi K J_m(K)} + [J_{m-1}(K) - \frac{m}{K} J_m(K)]\frac{Y_m(K)}{J_m(K)}\right\}$$

$$- \sigma K^2 [J_{m-1}(K) - \frac{m}{K} J_m(K)] \left\{\frac{2}{\pi\sigma K J_m(\sigma K)} + [J_{m-1}(\sigma K) - \frac{m}{\sigma K} J_m(\sigma K)]\frac{Y_m(\sigma K)}{J_m(\sigma K)}\right\} = 0 \tag{12}$$

$$f(K) = \sigma K^2 \left\{ \frac{2\,[J_{m-1}(\sigma K) - \frac{m}{\sigma K} J_m(\sigma K)]}{\pi K J_m(K)} - \frac{2\,[J_{m-1}(K) - \frac{m}{K} J_m(K)]}{\pi\sigma K J_m(\sigma K)} + \right.$$

$$\left. [J_{m-1}(K) - \frac{m}{K} J_m(K)]\,[J_{m-1}(\sigma K) - \frac{m}{\sigma K} J_m(\sigma K)]\,[\frac{Y_m(K)}{J_m(K)} - \frac{Y_m(\sigma K)}{J_m(\sigma K)}] \right\} = 0 \tag{13}$$

Equation (13) is evaluated for values of $\tilde{K}_i = M + 3(i-1)$; $i = 1, 2, 3, \ldots$ until $f(\tilde{K}_j)\, f(\tilde{K}_{j-1}) < 0$ for some j. A procedure employing a combination of false position and bisection techniques is then used to obtain a value of $K'^{\sigma}_{m\mu}$ in the interval $[\tilde{K}_{j-1}, \tilde{K}_j]$.

Having calculated a value of $K = K'^{\sigma}_{m\mu}$, the corresponding value of $Q = Q^{\sigma}_{m\mu}$ can be calculated. Combining (8) and (9) yields,

$$[J'_m(K) + J'_m(\sigma K)\sigma]K + Q[Y'_m(K) + Y'_m(\sigma K)\sigma]K = 0 \tag{14}$$

from which

$$Q = -\frac{J'_m(K) + J'_m(\sigma K)\sigma}{Y'_m(K) + Y'_m(\sigma K)\sigma} \tag{15}$$

For $\sigma = 0$, $Q^{\sigma}_{m\mu} = 0$ and $K'^{\sigma}_{m\mu} = 0$ is defined to be the root of

$$\left[\frac{d}{dr'} J_m(K'^{\sigma}_{m\mu} r')\right]_{r'=1} = 0 \tag{16}$$

Letting $K = K'^{\sigma}_{m\mu}$, and evaluating at $r' = 1$, (16) becomes

If $f(K) = J_m'(K)K = 0$, then (6) yields (17)

$$f(K) = [J_{m-1}(K) - \frac{m}{K} J_m(K)]K = 0 \tag{18}$$

Equation (18) is evaluated for values of $\tilde{K}_i = m + 3(i-1)$; $i = 1, 2, 3, \ldots$ until $f(\tilde{K}_j)\, f(\tilde{K}_{j-1}) < 0$ for some value of j. A procedure employing a combination of false position and bisection techniques is then used to obtain a value of $K'^{\sigma}_{m\mu}$ in the interval $(\tilde{K}_{j-1}, \tilde{K}_j)$.

APPENDIX B

MICROPHONE LOCATION PROGRAM

PROGRAM LISTING

PRECEDING PAGE BLANK NOT FILMED

ORIGINAL PAGE IS
OF POOR QUALITY

```
PRATT & WHITNEY AIRCRAFT DIVISION                                    VER     07/25/77
SC.PANLIB.L4                                                         9.0     12.50.00

++WRITE PRINT,T89901
C          DATA SET T89901      AT LEVEL 011 AS OF 06/16/77
C          DATA SET T89901      AT LEVEL 008 AS OF 04/06/77
C
C THIS PROGRAM CALCULATES THE 'BEST' PLACEMENT FOR MICROPHONES IN A DUCT
C
      DIMENSION AA(200), IND(10), WORK(5100), EIGVAL(50), EIGVEC(50,50),
     1          EQ(50,50), AMPL(50), PHASE(50)
      EXTERNAL DETCAL
      COMMON /MATRX/  NDIM1, ONE, ZERO, DET, MATRIX(50,50)
      COMMON /PROBES/ IWALL, ISAME, NPROBE, MNPROB
      COMMON /CNSTNT/ NMIKES, NMODES, SIGMA, B, MX, FREQ, A, OMEGA
      COMMON /EMUS/   EMU(50,50), IEMPRT
      COMMON /KQMU/   KMU(50), QMU(50)
      COMMON /MODES/  MODE(50), MU(50), IWAVE(50)
      COMMON /WAVENO/ KX(50)
      COMMON /MIKES/  NDIM, X(50), R(50), THETA(50)
      COMMON /SEARCH/ NVAR, IPRNT, NSRCH, NEVAL, IEXPNT, NTABLE, NSTAT
      COMMON /BESSL/  DUM1(5), PI
      COMMON /BOUNDS/ XBOUND(2,50), RBOUND(2,50), TBOUND(2,50), XA, XB,
     1                THMIN, THMAX, XLIM, RLIM, THLIM
      COMMON /CONDTN/ GOODCN
      COMMON /ANGLES/ DEGRAD, RADDEG
      REAL KMU, MX
      COMPLEX ONE, ZERO, DET, MATRIX, FACTR, DIVSR, EXPNT, KX, EIGVAL,
     1        EIGVEC, EQ
C
C INPUT CASE VARIABLES
C
   20 CALL INPUT( IEND )
      IF( IEND .GT. 0 )                     GO TO 9999
C
C CALCULATE CHARACTERISTIC NUMBERS KMU AND QMU FOR EACH SET OF
C  CIRCUMFERENTIAL MODE NUMBER AND RADIAL ORDER
C
      CALL KQCAL
C
C CALCULATE AXIAL WAVE NUMBER
C
      FLOW          = OMEGA / A
      AMACH         = 1. - MX * MX
      DO 40 I=1,NMODES
      RADICL        = FLOW ** 2 - AMACH * ( KMU(I) / B ) ** 2
      IF( RADICL )                          25, 30, 30
   25 KX(I)         = CMPLX( -MX * FLOW / AMACH, IWAVE(I) *
     1                SQRT( ABS( RADICL ) ) / AMACH )
                                            GO TO 40
   30 KX(I)         = CMPLX( ( -MX * FLOW + IWAVE(I) * SQRT( RADICL ) ) /
     1                AMACH, 0.0 )
   40 CONTINUE
C
C DETERMINE MICROPHONE LOCATION COMPONENTS, X, R, AND THETA, USING A
C  STOCHASTIC SEARCH TECHNIQUE
```

PRECEDING PAGE BLANK NOT FILMED

```
C
  120 IWALL          = 0
      IPROBE         = 0
      MPROBE         = 0
C
C SET UP FIRST ROW OF MATRIX AND INITIAL DETERMINANT VALUE
C
C CALCULATE CHARACTERISTIC E-FUNCTION VALUES ASSOCIATED WITH R
C
      RPRIME         = R(1) / B
      CALL EMUCAL( RPRIME, EMU(1,1), NMODES )
      DO 140 J=1,NMODES
      EXPNT          = CMPLX( 0.0, REAL( KX(J) ) * X(1) + MODE(J) *
     1                 THETA(1) )
      MATRIX(1,J)    = EMU(J,1) * CEXP( EXPNT ) * EXP( -AIMAG( KX(J) ) *
     1                 X(1) )
      EQ(1,J)        = MATRIX(1,J)
  140 CONTINUE
      DET            = MATRIX(1,1)
C
C SET UP ARRAYS FOR STOCHASTIC SEARCH
C
      IND(1)         = NVAR
      IND(2)         = NSRCH
      IND(3)         = NEVAL
      IND(4)         = NTABLE
      IND(5)         = NSTAT
      IND(6)         = 0
      IND(7)         = 0
      IND(8)         = IEXPNT
      IND(9)         = IPRNT
      IND(10)        = 1
      AVG            = .9 * ( XB - XA )
C
      MKSTRT         = NDIM + 1
      DO 440 JJ=MKSTRT,NMIKES
C
      WRITE(6,9000)
 9000 FORMAT (1H1, T44, '***  MICROPHONE LOCATION COMPUTER PROGRAM  ***'
     1)
      IF( JJ .LE. NMIKES )                    GO TO 147
C
C ALL MICROPHONE LOCATIONS HAVE BEEN INPUT. CALCULATE CONDITION NUMBER
C
      DO 430 JJJ=1,NMIKES
      J              = JJJ
                                              GO TO 175
  147 J              = JJ
      IF( IPRNT .GT. 0 )                      WRITE(6,9001) J
 9001 FORMAT( /, 1X, '... STOCHASTIC SEARCH OUTPUT FOR MICROPHONE ', I2)
C
C IF MODE NUMBERS CHANGE, RESET ISAME INDICATOR AND ATTEMPT TO PLACE
C  MIKE ON DUCT WALL
```

```
C
      IF( ISAME .EQ. 0 )                GO TO 150
      IF( MODE(J) .EQ. MODE(1) )        GO TO 150
      ISAME        = 0
      IWALL        = 0
  150 NDIM1        = J - 1
      NDIM         = J
  155 AA(1)        = XA
      IF( IWALL .NE. 0 )                AA(1) = XA + AVG
      AA(2)        = THMIN
      IF( SIGMA )                       160, 160, 165
  160 AA(3)        = .1
                                        GO TO 170
  165 AA(3)        = B * SIGMA
  170 AA(4)        = XB
      AA(5)        = THMAX
      AA(6)        = B
C
C SET UP BOUND ARRAYS
C
      XBOUND(1,J)  = AA(1)
      XBOUND(2,J)  = AA(4)
      TBOUND(1,J)  = AA(2)
      TBOUND(2,J)  = AA(5)
      RBOUND(1,J)  = AA(3)
      RBOUND(2,J)  = AA(6)
      CALL STOCH( DETCAL, IND, AA )
C
C SET UP MICROPHONE LOCATION COMPONENT ARRAYS
C
      X(J)         = AA(1)
      THETA(J)     = AA(2)
      R(J)         = AA(3)
C
C SET UP ROW J OF MATRIX USING THE VALUES OF X, THETA, AND R
C
C CALCULATE CHARACTERISTIC E-FUNCTION VALUES ASSOCIATED WITH R
C
  175 RPRIME       = R(J) / B
      CALL EMUCAL( RPRIME, EMU(1,J), NMODES )
      DO 180 I=1,NMODES
      EXPNT        = CMPLX( 0.0, REAL( KX(I) ) * X(J) + MODE(I) *
     1               THETA(J) )
      MATRIX(J,I)  = EMU(I,J) * CEXP( EXPNT ) * EXP( -AIMAG( KX(I) ) *
     1               X(J) )
      EQ(J,I)      = MATRIX(J,I)
  180 CONTINUE
      IF( JJ .GT. NMIKES )              GO TO 230
C
C TRIANGULARIZE THE J X J MATRIX
C
      FACTR        = ONE
      DO 220 I=1,NDIM1
```

```
      FACTR           = - ( MATRIX(J,I) * CONJG( MATRIX(I,I) ) /
     1                  CABS( MATRIX(I,I) ) ** 2 ) * FACTR
      DIVSR           = - MATRIX(I,I) * CONJG( MATRIX(J,I) ) /
     1                  CABS( MATRIX(J,I) ) ** 2
      MATRIX(J,I)  = ZERO
      ISTRT        = I + 1
      DO 200 K=ISTRT,NMODES
      MATRIX(J,K)  = MATRIX(I,K) + DIVSR * MATRIX(J,K)
  200 CONTINUE
  220 CONTINUE
C
C
C CALCULATE CONDITION NUMBER OF THE MATRIX
C
  230 MIKENO          = J
      CALL EIGCC( EQ, MIKENO, 50, 2, EIGVAL, EIGVEC, 50, WORK, IERR )
C
      EIGMIN          = CABS( EIGVAL(1) )
      EIGMAX          = EIGMIN
      DO 300 I=2,MIKENO
      EIGEN           = CABS( EIGVAL(I) )
      IF( EIGEN - EIGMIN )                240, 260, 260
  240 EIGMIN          = EIGEN
  260 IF( EIGEN - EIGMAX )                GO TO 300
  280 EIGMAX          = EIGEN             300, 300, 280
  300 CONTINUE
C
      CONDNO          = EIGMAX / EIGMIN
C
C CALCULATE AMPLITUDE AND PHASE OF EIGENVALUES. PRINT RESULTANT VALUES
C
      DO 310 L=1,MIKENO
      AMPL(L)         = CABS( EIGVAL(L) )
      PHASE(L)        = ATAN( AIMAG( EIGVAL(L) ) / REAL( EIGVAL(L) ) )
     1                  * RADDEG
  310 CONTINUE
      WRITE(6,9002) J, CONDNO
 9002 FORMAT( /, 1X, '... MICROPHONE ', I2, ' ...', //, T5, 'CONDITION N
     1UMBER = ', F10.4, T44, 'MODE', T59, 'EIGENVALUES', /, T52,
     2'AMPLITUDE           PHASE', / )
      DO 315 L=1,MIKENO
      WRITE(6,9003) L, AMPL(L), PHASE(L)
 9003 FORMAT( 44X, I2, 5X, F10.4, 5X, F10.4 )
  315 CONTINUE
      IF( JJ .GT. NMIKES )                GO TO 430
C
C IF CONDITION NUMBER IS 'GOOD', CONTINUE. OTHERWISE PUT MIKE ON PROBE
C
      IF( CONDNO - GOODCN )               360, 360, 320
  320 IF( IWALL .LT. 0 )                  GO TO 380
C
C MIKE IS EITHER ON WALL OR NOT THE FIRST MIKE ON A PROBE. IN ANY EVENT,
```

```
C  TRY AND PUT MIKE ON ANOTHER PROBE
C
      IPROBE        = 1 + IPROBE
      IF( IPROBE .LE. NPROBE )          GO TO 340
C
C ERROR - THE NUMBER OF PROBES EXCEEDS THE MAXIMUM ALLOWABLE
C
  330 WRITE(6,1000)
 1000 FORMAT( //, 1X, '***** THE NUMBER OF PROBES REQUIRED EXCEEDS THE
     1MAXIMUM PERMITTED. EXECUTION WILL BE TERMINATED  *****' )
                                        GO TO 9999
  340 MPROBE        = 1
      IWALL         = -1
                                        GO TO 155
C
C RESULTS ACCEPTABLE. IF ON WALL, CONTINUE. IF ON PROBE, READJUST PROBE
C  INFORMATION
C
  360 IF( IWALL .EQ. 0 )                GO TO 420
      MPROBE        = 1 + MPROBE
      IF( MPROBE .GT. MNPROB )          GO TO 380
      IWALL         = 1
                                        GO TO 420
  380 IPROBE        = 1 + IPROBE
      IF( IPROBE .GT. NPROBE )          GO TO 330
      IWALL         = -1
      MPROBE        = 1
C
C CALCULATE THE DETERMINANT OF THE J X J MATRIX
C
  420 DET           = MATRIX(J,J) * FACTR * DET
                                        GO TO 440
  430 CONTINUE
  440 CONTINUE
C
C PRINT RESULTS
C
      CALL PRINT
C
C RECYCLE FOR NEXT CASE
C
                                        GO TO 20
 9999 STOP
      END
      SUBROUTINE ANGPOS( ANGLE, NUMBER )
C
C THIS SUBROUTINE CONVERTS NEGATIVE ANGLES TO CORRESPONDING POSITIVE
C  ANGLES
C
      DIMENSION ANGLE(1)
      DATA DEGREE / 360.0 /
C
      DO 80 I=1,NUMBER
```

```
      IF( ANGLE(I) )                           20, 80, 80
   20 DO 40 J=1,10
      DELTA           = J * DEGREE
      IF( ANGLE(I) + DELTA )                   40, 60, 60
   40 CONTINUE
   60 ANGLE(I)        = DELTA + ANGLE(I)
   80 CONTINUE
C
 9999 RETURN
      END
      FUNCTION BESL1( X )
C
C THIS FUNCTION CALCULATES VALUES OF THE EQUATION DEFINING THE SYSTEM OF
C  DIFFERENTIAL EQUATIONS FOR A NON-ZERO HUB/TIP RATIO
C
      COMMON /BESSL/ ISIGN, JSIGN, DELKMU, TOL, M, PI
      COMMON /CNSTNT/ DUM1(2), SIGMA, DUM2(5)
C
      X1              = X * SIGMA
      CALL BESJ( X1, M-JSIGN, EMJM1, TOL, IER1 )
      CALL BESJ( X1, M , EMJX1, TOL, IER2 )
      CALL BESJ( X, M, EMJ, TOL, IER3 )
      CALL BESJ( X, M-JSIGN, EMJP1, TOL, IER4 )
      CALL BESY( X, M, EMYX, IER5 )
      CALL BESY( X1, M, EMYX1, IER6 )
C
      EMJM1           = JSIGN * ISIGN * EMJM1
      EMJX1           = ISIGN * EMJX1
      EMJ             = ISIGN * EMJ
      EMJP1           = JSIGN * ISIGN * EMJP1
      EMYX            = ISIGN * EMYX
      EMYX1           = ISIGN * EMYX1
C
      A1              = EMJM1 - ( M * JSIGN / X1 ) * EMJX1
      A2              = EMJP1 - ( M * JSIGN / X ) * EMJ
      A3              = 2. * A1 / ( PI * X * EMJ )
      A4              = 2. * A2 / ( PI * X1 * EMJX1 )
      A5              = A1 * A2 * ( EMYX / EMJ - EMYX1 / EMJX1 )
C
      BESL1           = X1 * X * ( A3 - A4 + A5 )
      RETURN
      END
      FUNCTION BESL2( X )
C
C THIS FUNCTION CALCULATES VALUES OF THE EQUATION DEFINING THE SYSTEM OF
C  DIFFERENTIAL EQUATIONS FOR A HUB/TIP RATIO OF ZERO
C
      COMMON /BESSL/ ISIGN, JSIGN, DELKMU, TOL, M, PI
      COMMON /CNSTNT/ DUM1(2), SIGMA, DUM2(5)
C
      CALL BESJ( X, M-JSIGN, EMJM1, TOL, IER1 )
      CALL BESJ( X, M, EMJ, TOL, IER2 )
C
```

36

```
      EMJM1          = JSIGN * ISIGN * EMJM1
      EMJ            = ISIGN * EMJ
      BESL2          = X * EMJM1 - M * JSIGN * EMJ
      RETURN
      END
      SUBROUTINE BESJ( X, N, BJ, D, IER )
C
C THIS SUBROUTINE CALCULATES THE J BESSEL FUNCTION FOR A GIVEN ARGUMENT,
C  X, AND ORDER N. THIS SUBROUTINE WAS TAKEN FROM THE IBM SCIENTIFIC
C  SUBROUTINE PACKAGE
C
      BJ             = 0.0
      IF( N .GE. 0 )                      GO TO 20
C
C ERROR - NEGATIVE ORDER. SET ERROR INDICATOR TO 1 AND RETURN
C
      IER            = 1
                                          GO TO 9999
   20 IF( X )                             40, 30, 60
   30 IF( N .GT. 0 )                      GO TO 40
      BJ             = 1.0
                                          GO TO 9999
C
C ERROR - ARGUMENT ZERO OF NEGATIVE. SET ERROR INDICATOR TO 2 AND RETURN
C
   40 IER            = 2
                                          GO TO 9999
C
C CALCULATE MAXIMUM ORDER NUMBER THAT CAN BE PROCESSED FOR X.
C   IF X .LE. 15, N MUST BE LESS THAN 20 + 10*X - X**(2/3)
C   IF X .GT. 15, N MUST BE LESS THAN 90 + X/2
C
   60 IF( X - 15. )                       80, 80, 100
   80 NTEST          = 20. + 10. * X - X ** 2 / 3.
                                          GO TO 120
  100 NTEST          = 90. + X / 2.
  120 IF( N .LT. NTEST )                  GO TO 140
C
C ERROR - ORDER RANGE COMPARED TO X IS NOT CORRECT. SET ERROR INDICATOR
C  TO 4 AND RETURN.
C
      IER            = 4
                                          GO TO 9999
  140 IER            = 0
      N1             = N + 1
      BPREV          = 0.0
C
C COMPUTE STARTING VALUE OF M
C
      IF( X - 5. )                        160, 180, 180
  160 MA             = X + 6.
                                          GO TO 200
  180 MA             = 1.4 * X + 60. / X
```

```
  200 MB            = N + IFIX( X ) / 4 + 2
      MZERO         = MAXO( MA, MB )
C
C SET UPPER LIMIT OF M
C
      MMAX          = NTEST
  220 DO 320 M=MZERO,MMAX,3
C
C SET F(M), F(M-1)
C
      FM1           = 1.0E-28
      FM            = 0.0
      ALPHA         = 0.0
      JT            = 1
      IF( ( M / 2 ) * 2 .EQ. M )        JT = -1
      M2            = M - 2
      DO 280 K=1,M2
      MK            = M - K
      BMK           = 2. * FLOAT( MK ) * FM1 / X - FM
      FM            = FM1
      FM1           = BMK
      IF( MK - N - 1 )                  260, 240, 260
  240 BJ            = BMK
  260 JT            = -JT
      S             = 1 + JT
      ALPHA         = ALPHA + BMK * S
  280 CONTINUE
C
      BMK           = 2. * FM1 / X - FM
      IF( N .EQ. 0 )                    BJ = BMK
      ALPHA         = ALPHA + BMK
      BJ            = BJ / ALPHA
      IF( ABS( BJ - BPREV ) - ABS( D * BJ ) )  9999, 9999, 300
  300 BPREV         = BJ
  320 CONTINUE
C
C ERROR - REQUIRED TOLERANCE NOT OBTAINED. SET ERROR INDICATOR TO 3 AND
C   RETURN
C
      IER           = 3
 9999 RETURN
      END
      SUBROUTINE BESY( X, N, BY, IER )
C
C THIS SUBROUTINE CALCULATES THE Y BESSEL FUNCTION FOR A GIVEN ARGUMENT,
C  X, AND ORDER N. THIS SUBROUTINE WAS TAKEN FROM THE IBM SCIENTIFIC
C  SUBROUTINE PACKAGE
C
      IER           = 0
      IF( N .GE. 0 )                    GO TO 20
C
C ERROR - NEGATIVE ORDER. SET ERROR INDICATOR TO 1 AND RETURN
C
```

```
      IER              = -1
                                          GO TO 9999
   20 IF( X )                             40, 40, 60
C
C ERROR - ARGUMENT ZERO OR NEGATIVE. SET ERROR INDICATOR TO 2 AND RETURN
C
   40 IER              = 2
                                          GO TO 9999
C
C BRANCH IF X IS LESS THAN OR EQUAL TO 4.
C
   60 IF( X - 4. )                        100, 100, 80
C
C CALCULATE Y0 AND Y1 FOR X GREATER THAN 4.
C
   80 T1               = 4. / X
      T2               = T1 * T1
      P0               = ( ( ( ( -.0000037043 * T2 + .0000173565 ) * T2 -
     1                   .0000487613 ) * T2 + .00017343 ) * T2 - .001753062
     2                   ) * T2 + .3989423
      Q0               = ( ( ( ( .0000032312 * T2 - .0000142078 ) * T2 +
     1                   .0000342468 ) * T2 - .0000869791 ) * T2 +
     2                   .0004564324 ) * T2 - .01246694
      P1               = ( ( ( ( .0000042414 * T2 - .0000020092 ) * T2 +,
     1                   .0000580759 ) * T2 - .000223203 ) * T2 +
     2                   .002921826 ) * T2 + .3989423
      Q1               = ( ( ( ( -.0000036594 * T2 + .00001622 ) * T2 -
     1                   .0000398708 ) * T2 + .0001064741 ) * T2 -
     2                   .00063904 ) * T2 + .03740084
      A                = 2. / SQRT( X )
      B                = A * T1
      C                = X - .7853982
      Y0               = A * P0 * SIN( C ) + B * Q0 * COS( C )
      Y1               = -A * P1 * COS( C ) + B * Q1 * SIN( C )
                                          GO TO 160
C
C CALCULATE Y0 AND Y1 FOR X LESS THAN OR EQUAL TO 4
C
  100 XX               = .5 * X
      X2               = XX * XX
      T                = ALOG( XX ) + .5772157
      SUM              = 0.0
      TERM             = T
      Y0               = T
      DO 120 L=1,15
      IF( L .NE. 1 )                          SUM = 1. / FLOAT( L-1) + SUM
      FL               = L
      TS               = T - SUM
      TERM             = ( -X2 * TERM / ( FL ** 2 ) ) * ( 1. - 1. / ( FL *
     1                   TS ) )
      Y0               = TERM + Y0
  120 CONTINUE
      TERM             = XX * ( T - .5 )
```

```
      SUM         = 0.0
      Y1          = TERM
      DO 140 L=2,16
      SUM         = 1. / FLOAT( L-1 ) + SUM
      FL          = L
      FL1         = FL - 1.
      TS          = T - SUM
      TERM        = ( -X2 * TERM / ( FL * FL1 ) ) * ( ( TS - .5 / FL )
     1              / ( TS + .5 / FL1 ) )
      Y1          = TERM + Y1
  140 CONTINUE
      PI2         = .6366198
      YO          = PI2 * YO
      Y1          = PI2 * ( Y1 - 1. / X )
C
C CHECK IF ONLY YO OR Y1 IS DESIRED
C
  160 IF( N .GT. 1 )                         GO TO 180
C
C RETURN YO OR Y1 AS REQUIRED
C
      BY          = YO
      IF( N .EQ. 1 )                         BY = Y1
                                             GO TO 9999
C
C PERFORM RECURRENCE OPERATIONS TO FIND YN(X)
C
  180 YA          = YO
      YB          = Y1
      K           = 1
  200 T = FLOAT( 2*K ) / X
      YC          = T * YB - YA
      IF( ABS( YC ) - 1.0E70 )               240, 240, 220
C
C ERROR - BY HAS EXCEEDED MAGNITUDE OF 10**70. SET ERROR INDICATOR TO 3
C  AND RETURN
C
  220 IER         = 3
                                             GO TO 9999
  240 K           = 1 + K
      IF( K .EQ. N )                         GO TO 260
      YA          = YB
      YB          = YC
                                             GO TO 200
  260 BY          = YC
 9999 RETURN
      END
      BLOCK DATA
      COMMON /DFAULT/ NLOC, HTR, OR, EMX, FRQ, M(50), MUS(50), IDIR(50),
     1               CONDNO, ILMU, XMIN, XMAX, DELX, DELTH,
     2               DELR, ND, XM(50), RM(50), THM(50), IPT, NSRH,
     3               NVAL, IXPNT, NPROB, MPROB, NTAB, NST, SPEED, TMIN,
     4               TMAX
```

```
C
C GEOMETRY AND TEST CONDITION DEFAULT VALUES
C
C     NLOC     - NUMBER OF MICROPHONE LOCATIONS DESIRED
C     HTR      - HUB / TIP RATIO
C     OR       - OUTER RADIUS OF DUCT
C     EMX      - AXIAL MACH NUMBER
C     FRQ      - TEST FREQUENCY
C
      DATA NLOC / 2 /, HTR / 0.438 /, OR / 5.0 /, EMX / -0.07 /,
     1     FRQ / 3100.0 /, SPEED / 13566.24 /
C
C MODE DEFAULT VALUES
C
C     M        - CIRCUMFERENTIAL MODE NUMBER
C     MUS      - RADIAL ORDER
C     IDIR     - WAVE DIRECTION INDICATOR
C
      DATA M / 2*-2, 48*0 /, MUS / 0, 1, 48*0 /, IDIR / 2*1, 48*0 /
C
C CONDITION NUMBER DEFAULT VALUE
C
C     CONDNO - MAXIMUM CONDITION NUMBER
C
      DATA CONDNO / 100.0 /
C
C STOCHASTIC SEARCH PARAMETER BOUND-DEFAULT VALUES
C
C     XMIN     - MINIMUM X VALUE OF DUCT
C     XMAX     - MAXIMUM X VALUE OF DUCT
C     DELX     - MINIMUM AXIAL DISTANCE ALLOWED BETWEEN MICROPHONES
C     DELR     - MINIMUM RADIAL DISTANCE ALLOWED BETWEEN MICROPHONES
C     DELTH    - MINIMUM ANGULAR DISTANCE ALLOWED BETWEEN MICROPHONES (DEG)
C     TMIN     - MINIMUM BOUND FOR THETA
C     TMAX     - MAXIMUM BOUND FOR THETA
C
      DATA XMIN / 0.0 /, XMAX / 12.0 /, DELX / 1.0 /, DELR / 1.0 /,
     1     DELTH / 11.5 /, TMIN / 0.0 /, TMAX / 360. /
C
C MICROPHONE LOCATION DEFAULT VALUES
C
C     ND       - NUMBER OF INITIAL FIXED MICROPHONE LOCATIONS
C     XM       - AXIAL COMPONENTS OF MICROPHONE LOCATIONS
C     RM       - RADIAL COMPONENTS OF MICROPHONE LOCATIONS
C     THM      - ANGULAR COMPONENTS OF MICROPHONE LOCATIONS
C
      DATA ND / 1 /, XM / 50*0.0 /, RM / 2*5.0, 48*0.0 /,
     1     THM / 50*0.0 /
C
C STOCHASTIC SEARCH DEFAULT VALUES
C
C     IPT      - PRINT INDICATOR FOR SEARCHES
C     NSRH     - MAXIMUM NUMBER OF SEARCHES
```

```
C     NVAL    - NUMBER OF EVALUATIONS PER SEARCH
C     IXPNT   - EXPONENT OF TOLERANCE ( E.G. TOL = 10. ** IXPNT )
C     NTAB    - NUMBER OF ELEMENTS IN DETERMINANT TABLE
C     NST     - NUMBER OF ELEMENTS TO BE USED IN CALCULATING STATISTICS
C     NVAR    - NUMBER OF INDEPENDENT VARIABLES
C
      DATA IPT / 0 /, NSRH / 50 /, NVAL / 500 /, IXPNT / -2 /,
     1     NTAB / 30 /, NST / 30 /
      COMMON /SEARCH/ NVAR, DUM(6)
      DATA NVAR / 3 /
C
C PROBE DEFAUALT VALUES
C
C     NPROB   - MAXIMUM NUMBER OF PROBES ALLOWED IN DUCT
C     MPROB   - MAXIMUM NUMBER OF MICROPHONES PER PROBE ALLOWED
C
      DATA NPROB / 1 /, MPROB / 1 /
C
C GENERAL PRINT DEFAULT VALUES
C
C     IEMU    - PRINT INDICATOR FOR CHARACTERISTIC E-FUNCTION VALUES
C
      DATA IEMU / 0 /
C
C BESSEL FUNCTION VALUES
C
      COMMON /BESSL/  DUM2(2), DELKMU, TOL, MM, PI
      DATA DELKMU / 3.0 /, TOL / .0001 /, PI / 3.141593 /
C
C ANGULAR CONVERSION VALUES
C
C     DEGRAD - DEGREES TO RADIANS
C     RADDEG - RADIANS TO DEGREES
C
      COMMON /ANGLES/ DEGRAD, RADDEG
      DATA DEGRAD / 0.0174533 /, RADDEG / 57.29578 /
C
      COMPLEX ONE, ZERO, DUM3
      COMMON /MATRX/ NDIM1, ONE, ZERO, DUM3(2501)
      DATA ONE / (1.,0.) /, ZERO / (0.,0.) /
      END
      SUBROUTINE EMUCAL( RPRIME, EMU, NMODES )
C
C THIS SUBROUTINE CALCULATES NMODES CHARACTERISTIC E-FUNCTION VALUES FOR
C  A PARTICULAR RADIAL VALUE, RPRIME.
C
      DIMENSION EMU(1)
      COMMON /KQMU/   KMU(50), QMU(50)
      COMMON /MODES/  MODE(50), DUM2(100)
      COMMON /BESSL/  ISIGN, JSIGN, DELKMU, TOL, M, PI
      REAL KMU
      DO 40 I=1,NMODES
      M            = IABS( MODE(I) )
```

```
      ISIGN         = 1
      IF( M .NE. 0 )                 ISIGN = MODE(I) / M
      IF( ISIGN .GE. 0 )             GO TO 20
C
C NEGATIVE MODE NUMBER. IF EVEN, SIGN OF BESSEL FUNCTION WILL BE +1. IF
C  ODD, SIGN OF BESSEL FUNCTION WILL BE -1.
C
      IF( ( M / 2 ) * 2 .EQ. M )     ISIGN = 1
   20 CONST         = KMU(I) * RPRIME
C
C CALCULATE BESSEL FUNCTIONS OF FIRST AND SECOND KIND FOR KMU(I)*RPRIME
C
      CALL BESJ( CONST, M, EMJ, TOL, IER1 )
      CALL BESY( CONST, M, EMY, IER2 )
      EMJ           = ISIGN * EMJ
      EMY           = ISIGN * EMY
C
C CALCULATE CHARACTERISTIC E-FUNCTION
C
      EMU(I)        = EMJ + QMU(I) * EMY
   40 CONTINUE
 9999 RETURN
      END
      FUNCTION FALZIP (FUNCT, AL , BR , TOL, ROOT, ITER, YY)
C
C CORRESPONDS TO OLD VERSION (FALSIE) ARGUMENT LIST AS FOLLOWS (THIS IS
C  FOR INTERNAL PURPOSES ONLY, IN USE THE TWO ARE INTERCHANGEABLE).
C
C     FUNCTION FALSIE (AXR,   XXL, XXR, TOL, ROOT, ITER, YY)
C
C
C  THIS ROUTINE USES A COMBINATION OF FALSE POSITION AND BISECTION
C    TECHNIQUES TO SOLVE FOR A ROOT ('ROOT') OF A GIVEN FUNCTION
C    ('FUNCT') WHICH HAS ONE ARGUMENT (THE INDEPENDENT VARIABLE).
C
C   'AL,BR' DEFINES THE INTERVAL TO BE SEARCHED.
C
C  THE VALUE RETURNED BY THE FUNCTION IS FALZIP.  FUNCT(FALZIP) = ROOT
C
C  THE SEARCH CONTINUES UNTIL TWO SUBSEQUENT GUESSES ARE WITHIN 'TOL'
C    OF EACH OTHER, OR UNTIL 'ITER' ITERATIONS HAVE TAKEN PLACE.
C
C  'YY' IS RETURNED AS FUNCT(FALZIP), AND SHOULD BE CLOSE TO 'ROOT'.
C
C  THE TECHNIQUE WAS ADAPTED FROM AN ALGO  SUBROUTINE APPEARING IN THE
C    COMPUTER JOURNAL 12 (1969) --  'EIGENVALUES OF A*X = LAMBDA*B*X
C    WITH BAND SYMMETRIC A AND B' BY G. PETERS + J.H. WILKINSON
C
      EXTERNAL FUNCT
      REAL INTERP
      IWRIT = 6
C
C  J IS COUNT OF ITERATIONS.
```

```
    1 J = 0
      A = AL
      B = BR
C
C  EVALUATE FUNCTION AT LEFT (A) AND RIGHT (B) BRACKETS.
      AF = FUNCT (A)
      BF = FUNCT (B)
C
C  THE FOLLOWING (THROUGH STATEMENT 3) DETERMINES IF THE FUNCTION IS OF
C    OPPOSITE SIGN AT THE ENDPOINTS GIVEN.
      ISW = 1
      IF (BF - ROOT) 2, 75, 3
    2 ISW = -1
    3 IF ((AF - ROOT) * ISW) 50, 80, 85
C
C  STATEMENT 5 INCREMENTS THE COUNTER J; FIRST TIME THROUGH GO TO 50.
    5 J = J + 1
C
C  IF LEFT BRACKET HAS 'SAME' FUNCTION VALUE AS RIGHT, USE BISECTION.
C    OTHERWISE, SET UP INTERPOLATED POINT FOR POSSIBLE USE.
      IF (ABS((AF - BF)/BF) - 1.E-5) 10, 10, 15
   10 INTERP = BISECT
      GO TO 20
   15 INTERP = (A*BF - B*AF + (B-A)*ROOT) / (BF-AF)
C
C  IF WITHIN A TOLERANCE OF THE BRACKET B, MOVE THE INTERPOLATED POINT
C    ONE TOLERANCE AWAY.
   20 IF ((ABS(INTERP-B)/ABS(INTERP+B)) -2.*TOL) 22,23,23
   22 INTERP = B + (C - B) / ABS (C - B) * TOL
C
C  SET A=B (B IS ALWAYS THE POINT WITH SMALLEST (ABS) VALUE OF FUNCTION.
   23 A = B
      AF = BF
C
C  USE POINT CLOSEST TO B (INTERP OR BISECT) AS NEW B AND EVALUATE BF.
      IF ((INTERP - BISECT) * (B - INTERP)) 30, 25, 25
   25 B = INTERP
      GO TO 35
   30 B = BISECT
   35 BF = FUNCT(B)
      BFMR = BF - ROOT
C
C  IF CF IS ON THE SAME SIDE OF THE ROOT AS BF, LET POINT C = POINT A.
   40 IF ((CF - ROOT) * BFMR) 55, 75, 50
   50 C = A
      CF = AF
C
C  IF CF IS CLOSER (ABS) TO ROOT THAN BF, SWITCH POINTS B AND C.
C    IN ANY CASE, B AND C ARE THE TWO BRACKETS. ALSO BF IS CLOSER TO THE
C    ROOT THAN CF IS.
   55 IF (ABS(BF - ROOT) - ABS(CF - ROOT)) 60, 60, 57
   57 A  = B
      AF = BF
```

```
      B  = C
      BF = CF
      C  = A
      CF = AF
C
C  SET UP BISECTION POINT.  IF CLOSE ENOUGH, FINISH UP.  OTHERWISE GO
C     BACK IF ITERATION COUNT DOESN'T EXCEED MAXIMUM.
   60 BISECT = (B + C) / 2.
      IF ((ABS(BISECT-B)/ABS(BISECT+B)) -2.*TOL) 75,65,65
   65 IF (J - ITER) 5, 70, 70
   70 WRITE(IWRIT,1000)J,B,BF,C,CF
 1000 FORMAT (1H0/// 30X, 'IN FALZIP, AFTER', I4, ' ITERATIONS' //
     1     10X, 'BRACKET 1 = ', G15.8, 5X, 'FUNCTION = ', G15.8/
     2     10X, 'BRACKET 2 = ', G15.8, 5X, 'FUNCTION = ', G15.8/
     3     5X, 'BRACKET 1 WAS RETURNED AS RESULT.')
   75 FALZIP = B
      YY = BF
      RETURN
   80 FALZIP = A
      YY = AF
      RETURN
   85 WRITE(IWRIT,1100)ROOT,A,AF,B,BF
 1100 FORMAT ('0***IN FALZIP, ROOT GIVEN (=', G15.8, ') DIDN''T FALL BET
     1WEEN VALUES OF FUNCTION AT BRACKETS GIVEN***'/
     2    10X, 'BRACKET 1 = ', G15.8, 5X, 'FUNCTION = ', G15.8 /
     3    10X, 'BRACKET 2 = ', G15.8, 5X, 'FUNCTION = ', G15.8 /
     4          40X, 'TERMINATING RUN' )
      STOP
      END
      SUBROUTINE INPUT( IEND )
C
C THIS SUBROUTINE INPUTS THE DATA REQUIRED FOR THE EXECUTION OF A CASE
C
      DIMENSION MXMODE(50), ISAVE(50,3)
      COMMON /DFAULT/ NLOC, HTR, OR, EMX, FRQ, M(50), MUS(50), IDIR(50),
     1                CONDNO, IEMU, XMIN, XMAX, DELX, DELTH,
     2                DELR, ND, XM(50), RM(50), THM(50), IPT, NSRH,
     3                NVAL, IXPNT, NPROB, MPROB, NTAB, NST, SPEED, TMIN,
     4                TMAX
      COMMON /CNSTNT/ NMIKES, NMODES, SIGMA, B, MX, FREQ, A, OMEGA
      COMMON /MODES/  MODE(50), MU(50), IHAVE(50)
      COMMON /BESSL/  DUM1(5), PI
      COMMON /ANGLES/ DEGRAD, RADDEG
      COMMON /EMUS/   EMU(50,50), IEMPRT
      COMMON /BOUNDS/ XBOUND(2,50), RBOUND(2,50), TBOUND(2,50), XA, XB,
     1                THMIN, THMAX, XLIM, RLIM, THLIM
      COMMON /MIKES/  NDIM, X(50), R(50), THETA(50)
      COMMON /SEARCH/ NVAR, IPRNT, NSRCH, NEVAL, IEXPNT, NTABLE, NSTAT
      COMMON /PROBES/ IHALL, ISAME, NPROBE, MNPROB
      COMMON /CONDIN/ GOODCN
C
      NAMELIST /INDATA/ NLOC, HTR, OR, EMX, FRQ, M, MUS, IDIR, CONDNO,
     1                   IEMU, XMIN, XMAX, DELX, DELTH, DELR,
```

```
     2                    ND, XM, RM, THM, IPT, NSRH, NVAL, IXPNT,
     3                    NPROB, MPROB, NTAB, NST, SPEED, THIN, THAX
      REAL MX
      IEND           = 0
      READ(5,INDATA,END=9998)
C
C CHANGE INPUT UNITS TO INTERNAL UNITS
C
      GOODCN         = CONDNO
      NMIKES         = NLOC
      NMODES         = NLOC
      SIGMA          = HTR
      B              = DR
      MX             = EMX
      FREQ           = FRQ
      IEMPRT         = IEMU
      XA             = XMIN
      XB             = XMAX
      XLIM           = DELX
      RLIM           = DELR
      THLIM          = DEGRAD * DELTH
      NDIM           = ND
      IPRNT          = IPT
      NSRCH          = NSRH
      NEVAL          = NVAL
      IEXPNT         = IXPNT
      NPROBE         = NPROB
      MNPROB         = MPROB
      NTABLE         = NTAB
      NSTAT          = NST
      THMIN          = DEGRAD * THIN
      THMAX          = DEGRAD * THAX
      A              = SPEED
      DO 20 I=1,NMODES
      MODE(I)        = M(I)
      MU(I)          = MUS(I)
      IWAVE(I)       = IDIR(I)
   20 CONTINUE
      DO 40 I=1,NDIM
      X(I)           = XM(I)
      R(I)           = RM(I)
      THETA(I)       = THM(I) * DEGRAD
   40 CONTINUE
C
C ORDER MODE NUMBERS FROM LARGEST TO SMALLEST. ALSO ORDER RADIALS WITHIN
C  EACH MODE NUMBER FROM LARGEST TO SMALLEST
C
      IF( NMODES .LE. 1 )             GO TO 280
      DO 50 I=1,20
      MXMODE(I)      = 1
   50 CONTINUE
      DO 220 I=2,NMODES
      MAX            = MXMODE(I)
```

```
      IF( IABS( MODE(I) ) - IABS( MODE(MAX) ) )  120, 60, 80
   60 IF( MU(I) .LE. MU(MAX) )                  GO TO 120
   80 DO 100 J=2,I
      JJ              = I - J + 2
      MXMODE(JJ)      = MXMODE(JJ-1)
  100 CONTINUE
      MXMODE(1)       = I
                                                GO TO 220
  120 DO 200 K=2,I
      NEXT            = MXMODE(K)
      IF( IABS( MODE(I) ) - IABS( MODE(NEXT) ) )  200, 140, 160
  140 IF( MU(I) .LE. MU(NEXT) )                 GO TO 200
  160 J1              = K + 1
      DO 180 J=J1,I
      JJ              = I - J + J1
      MXMODE(JJ)      = MXMODE(JJ-1)
  180 CONTINUE
      MXMODE(K)       = I
                                                GO TO 220
  200 CONTINUE
      MXMODE(I)       = I
  220 CONTINUE
C
      DO 240 I=1,NMODES
      MAX             = MXMODE(I)
      ISAVE(I,1)      = MODE(MAX)
      ISAVE(I,2)      = MU(MAX)
      ISAVE(I,3)      = IWAVE(MAX)
  240 CONTINUE
C
      DO 260 I=1,NMODES
      MODE(I)         = ISAVE(I,1)
      MU(I)           = ISAVE(I,2)
      IWAVE(I)        = ISAVE(I,3)
  260 CONTINUE
C
C IF THE FIRST TWO MODE NUMBERS ARE EQUAL, SET ISAME INDICATOR TO 1
C
  280 ISAME           = 0
      IF( MODE(1) .EQ. MODE(2) )                ISAME = 1
C
C CALCULATE RADIAN FREQUENCY
C
      OMEGA           = 2. * PI * FREQ
C
C ZERO COORDINATE BOUND ARRAYS
C
      DO 320 J=1,NLOC
      DO 300 I=1,2
      XBOUND(I,J)     = 0.0
      RBOUND(I,J)     = 0.0
      TBOUND(I,J)     = 0.0
  300 CONTINUE
```

```
  320 CONTINUE
                                         GO TO 9999
C
C END OF DATA SET
C
 9998 IEND           = 1
 9999 RETURN
      END
      SUBROUTINE KMUCAL( VALUE, DELTA, KMU, RIGHT )
C
C THIS SUBROUTINE CALCULATES THE CHARACTERISTIC NUMBER, KMU
C
      EXTERNAL BESL1, BESL2
      COMMON /CNSTNT/ DUM1(2), SIGMA, DUM2(5)
      REAL KMU, LEFT
   30 IPLUS          = 0
      IMINUS         = 0
   35 IF( SIGMA )                        50, 40, 50
C
   40 KMU            = BESL2( VALUE )
                                         GO TO 60
   50 KMU            = BESL1( VALUE )
C
   60 IF( KMU )                          80, 65, 70
   65 RIGHT          = VALUE
                                         GO TO 130
   70 IPLUS          = 1
                                         GO TO 90
   80 IMINUS         = 1
C
C DETERMINE IF LEFT AND RIGHT BRACKETS HAVE BEEN FOUND.
C
   90 IF( IPLUS .EQ. 1 .AND. IMINUS .EQ. 1 )  GO TO 100
C
C BRACKETS NOT FOUND. RECYLE.
C
      VALUSV         = VALUE
      VALUE          = DELTA + VALUE
                                         GO TO 35
C
C BRACKETS FOUND, CALCULATE KMU
C
  100 LEFT           = VALUSV
      RIGHT          = VALUE
      IF( SIGMA )                        110, 120, 110
  110 KMU            = FALZIP( BESL1, LEFT, RIGHT, .001, 0.0, 75, YY )
                                         GO TO 130
  120 KMU            = FALZIP( BESL2, LEFT, RIGHT, .001, 0.0, 75, YY )
  130 RETURN
      END
      SUBROUTINE KQCAL
C
C THIS SUBROUTINE CALCULATES THE CHARACTERISTIC NUMBERS KMU AND QMU
```

```
C
      COMMON /MODES/  MODE(50), MU(50), INAVE(50)
      COMMON /KQMU/   KMU(50), QMU(50)
      COMMON /BESSL/  ISIGN, JSIGN, DELKMU, TOL, M, PI
      COMMON /CNSTNT/ NMIKES, NMODES, SIGMA, DUM2(5)
      REAL KMU, KMUPRM
C
      DO 100 I=1,NMODES
C
C CALCULATE ORDER FOR BESSEL FUNCTION EVALUATION
C
      M            = IABS( MODE(I) )
      IF( M .NE. 0 )                  GO TO 10
      ISIGN        = 1
      JSIGN        = -1
      BRAKTL       = .1
                                      GO TO 20
   10 ISIGN        = MODE(I) / M
      JSIGN        = ISIGN
      BRAKTL       = M
      IF( ISIGN .GE. 0 )              GO TO 20
C
C NEGATIVE ORDER. IF EVEN, SIGN OF BESSEL FUNCTION WILL BE +1. IF ODD,
C  SIGN OF BESSEL FUNCTION WILL BE -1.
C
      IF( ( M / 2 ) * 2 .EQ. M )      ISIGN = 1
C
   20 NUMMUS       = MU(I) + 1
C
C CALCULATE CHARACTERISTIC NUMBER KMU CORRESPONDING TO MODE(I) AND MU(I)
C  THE VALUE OF KMU WILL BE THE MU(I)+1 ROOT OF THE EQUATION DEFINING
C  THE SYSTEM OF SIMULTANEOUS EQUATIONS
C
      KMUPRM       = 0.0
      DO 40 J=1,NUMMUS
      IF( M .EQ. 0 .AND. J .EQ. 1 )   GO TO 40
      CALL KMUCAL( BRAKTL, DELKMU, KMUPRM, BRAKTR )
      BRAKTL       = BRAKTR
   40 CONTINUE
      KMU(I)       = KMUPRM
C
C CALCULATE CHARACTERISTIC NUMBER QMU CORRESPONDING TO MODE(I) AND MU(I)
C  IF THE HUB/TIP RATIO IS ZERO, SET QMU TO ZERO AND CONTINUE
C
      IF( SIGMA )                     60, 60, 80
   60 QMU(I)       = 0.0
                                      GO TO 100
   80 IF( KMU(I) )                    90, 60, 90
   90 CALL BESJ( KMUPRM, M-JSIGN, EMM1, TOL, IER )
      EMM1         = ISIGN * JSIGN * EMM1
      CALL BESJ( KMUPRM, M, EMJ, TOL, IER2 )
      CALL BESY( KMUPRM, M, EMY, IER3 )
      EMJ          = ISIGN * EMJ
```

```
      EMY             = ISIGN * EMY
      CALL BESY( KMUPRM, M-JSIGN, EYM1, IER4 )
      EYM1            = ISIGN * JSIGN * EYM1
      A               = EMM1 - ( M * JSIGN * EMJ ) / KMUPRM
      B               = EYM1 - ( M * JSIGN * EMY ) / KMUPRM
      SIGMAK          = SIGMA * KMUPRM
      CALL BESJ( SIGMAK, M-JSIGN, EMM1, TOL, IER5 )
      CALL BESJ( SIGMAK, M, EMJ, TOL, IER6 )
      CALL BESY( SIGMAK, M, EMY, IER7 )
      CALL BESY( SIGMAK, M-JSIGN, EYM1, IER8 )
      EMM1            = ISIGN * JSIGN * EMM1
      EYM1            = ISIGN * JSIGN * EYM1
      EMJ             = ISIGN * EMJ
      EMY             = ISIGN * EMY
      C               = EMM1 - ( M * JSIGN * EMJ ) / SIGMAK
      D               = EYM1 - ( M * JSIGN * EMY ) / SIGMAK
      QMU(I)          = - ( A + C * SIGMA ) / ( B + D * SIGMA )
  100 CONTINUE
 9999 RETURN
      END
      SUBROUTINE PRINT
C
C THIS SUBROUTINE PRINTS INPUT AND CALCULATED VALUES
C
      COMMON /CNSTNT/ NMIKES, NMODES, SIGMA, B, MX, FREQ, A, OMEGA
      COMMON /MODES/  MODE(50), MU(50), IWAVE(50)
      COMMON /ANGLES/ DEGRAD, RADDEG
      COMMON /BOUNDS/ XBOUND(2,50), RBOUND(2,50), TBOUND(2,50), XA, XB,
     1                THMIN, THMAX, XLIM, RLIM, THLIM
      COMMON /EMUS/   EMU(50,50), IEMPKT
      COMMON /MIKES/  NOIM, X(50), R(50), THETA(50)
      COMMON /SEARCH/ NVAR, IPRNT, NSRCH, NEVAL, IEXPNT, NTABLE, NSTAT
      COMMON /WAVENO/ KX(50)
      COMMON /KQMU/   KMU(50), QMU(50)
      REAL MX, KMU
      COMPLEX KX, KXANGL
C
C CONVERT INTERNAL UNITS TO OUTPUT UNITS
C
      THLIM           = RADDEG * THLIM
      THMIN           = RADDEG * THMIN
      THMAX           = RADDEG * THMAX
      DO 40 J=1,NMIKES
      DO 20 I=1,2
      TBOUND(I,J)     = RADDEG * TBOUND(I,J)
   20 CONTINUE
      THETA(J)        = RADDEG * THETA(J)
   40 CONTINUE
C
C CONVERT ANY NEGATIVE ANGLES TO POSITIVE ANGLES FOR PRINTING
C
      CALL ANGPOS( THETA, NMIKES )
C
```

```
C PRINT INPUT
C
      WRITE(6,9000)
 9000 FORMAT( 1H1, T44, '***  MICROPHONE LOCATION COMPUTER PROGRAM  ***'
     1)
      WRITE(6,9013)
 9013 FORMAT( //, T56, '... INPUT VARIABLES ...' )
      WRITE(6,9001) NMIKES, NMODES
 9001 FORMAT( //, T5, 'NUMBER OF MICROPHONE LOCATIONS = ', I2, T51,
     1'NUMBER OF (MODE,MU) SETS = ', I2 )
      WRITE(6,9002)
 9002 FORMAT( //, 1X, '... INPUT MODES ...', //, T5, 'MODE', T14,
     1'CIRCUMFERENTIAL', T34, 'RADIAL', T47, 'WAVE', T61, 'AXIAL WAVE NU
     2MBER', T89, 'KMU', /, T16, 'MODE NUMBER', T34, 'ORDER', T47,
     3'INDICATOR', T62, 'REAL', T71, 'IMAGINARY', / )
      DO 80 I=1,NMODES
      KXANGL       = RADDEG * KX(I)
      WRITE(6,9003) I, MODE(I), MU(I), IWAVE(I), KXANGL, KMU(I)
 9003 FORMAT( 5X, I2, 11X, I4, 13X, I2, 10X, I2, 9X, F10.4, 1X, F10.4,
     15X, F10.4 )
   80 CONTINUE
      WRITE(6,9005) SIGMA, B, MX, FREQ, A, OMEGA
 9005 FORMAT( ///, 1X, '... TEST GEOMETRY AND CONDITIONS ...', //, T5,
     1'HUB / TIP RATIO = ', F8.3, T42, 'OUTER RADIUS OF DUCT = ', F8.3,
     2T84, 'AXIAL MACH NUMBER = ', F8.3, /, T5, 'FREQUENCY = ', F10.3,
     3T42, 'SPEED OF SOUND = ', F9.2, T84, 'RADIAN FREQUENCY =', F12.3 )
      WRITE(6,9006) NVAR, NSRCH, NEVAL
 9006 FORMAT( ///, 1X, '... STOCHASTIC SEARCH VALUES ...', //, T5,
     1'NUMBER OF VARIABLES = ', I2, T42, 'MAXIMUM NUMBER OF SEARCHES = '
     2, I5, T84, 'NUMBER OF EVALUATIONS PER SEARCH = ', I5 )
      WRITE(6,9007)
 9007 FORMAT( //, T5, 'MICROPHONE', T59, 'PARAMETER BOUNDS',/, T20,
     1'MINIMUM X', 4X, 'MAXIMUM X', 5X, 'MINIMUM R', 4X, 'MAXIMUM R',
     25X, 'MINIMUM THETA', 4X, 'MAXIMUM THETA', / )
      DO 100 J=2,NMIKES
      WRITE(6,9008) J, ( XBOUND(I,J),I=1,2 ), ( RBOUND(I,J),I=1,2 ),
     1                ( TBOUND(I,J),I=1,2 )
 9008 FORMAT( 6X, I2, 3X, 2(7X,F7.3,6X,F7.3), 9X, F7.3, 10X, F7.3 )
  100 CONTINUE
C
C PRINT MICROPHONE LOCATIONS
C
      WRITE(6,9000)
      WRITE(6,9009)
 9009 FORMAT( //, 1X, '... CALCULATED MICROPHONE LOCATIONS ...', //, T5,
     1'MICROPHONE', T25, 'X', T42, 'R', T57, 'THETA', / )
      DO 120 I=1,NMIKES
      WRITE(6,9010) I, X(I), R(I), THETA(I)
 9010 FORMAT( 4X, I2, 9X, 3(F12.6,5X) )
  120 CONTINUE
C
C PRINT CHARACTERISTIC E-FUNCTION VALUES IF REQUESTED
C
```

```
      IF( IEMPRT .LE. 0 )                    GO TO 9999
      WRITE(6,9011) ( I,I=1,NMODES )
 9011 FORMAT( ////, 1X, '... CHARACTERISTIC E-FUNCTION VALUES (FINAL SOL
     1UTION VALUES) ...', //, 1X, 'LOCATION', T68, 'MODES', /, 8X,
     215(6X,I2), / )
      DO 140 J=1,NMIKES
      WRITE(6,9012) J, ( EMU(I,J),I=1,NMODES )
 9012 FORMAT( 4X, I2, 5X, 15(1X,F7.3) )
  140 CONTINUE
C
 9999 RETURN
      END
      SUBROUTINE STOCH( FCT, IND, WORK )
      DIMENSION IND(1), WORK(1)
      EXTERNAL FCT
      COMMON /INITL/ NPARM, NSERCH, NEVAL, NTABLE, NSTAT, ISEED, IDIST,
     1               IEXPNT, IPRINT, ITYPE, LT, JT, ICNVRG, TOL
      COMMON /INDEX/ DUM1(2), INDEX4, INDEX5, INDEX6, INDEX7
      COMMON /SEED/   JSEED
C
C INITIALIZE PARAMETERS FOR SEARCH
C
      CALL INITAL( WORK, IND )
C
C SEARCH LOOP
C
      DO 120 I=1,NSERCH
      JSEED          = ISEED
C
C GENERATE STATISTICS FOR SEARCH I
C
      CALL STAT( WORK, I )
C
C EVALUATION LOOP
C
      DO 100 J=1,NEVAL
C
C GENERATE RANDOM PARAMETER VALUES
C
      CALL RANDOM( WORK )
C
C EVALUATE FUNCTION WITH RANDOM PARAMETER VALUES
C
      GP             = FCT( O, WORK(INDEX7+1), IE )
C
C IF FUNCTION VALUE (GP) IS MEANINGFUL UPDATE G TABLE. IF NOT GO ON TO
C  NEXT EVALUATION
C
      IF( IE .GT. 0 )                        GO TO 100
      CALL UPDATE( WORK, GP )
  100 CONTINUE
C
C PRINT RESULTS OF SEARCH I IF REQUESTED
```

```
C
      IF( IPRINT .EQ. 2 )               CALL PRINTT( WORK, I )
C
C TEST FOR CONVERGENCE. IF CONVERGED, RETURN WITH VALUES OF VARIABLES
C  IN THE FIRST NPARM ELEMENTS OF WORK ARRAY. IF NOT, GO ON TO NEXT
C  SEARCH.
C
      CALL CONVRG( WORK )
      IF( ICNVRG .GT. 0 )               GO TO 140
      NOCALL          = FCT( 1, WORK(INDEX7+1), IE )
  120 CONTINUE
C
C PRINT RESULTS OF LAST SEARCH IF REQUESTED
C
      I               = I - 1
  140 IF( IPRINT .EQ. 1 )               CALL PRINTT( WORK, I )
      IBEST           = IFIX( WORK(INDEX4+1) + .3 )
      DO 160 I=1,NPARM
      JNDEX           = INDEX6 + NTABLE * ( I - 1 )
      WORK(I)         = WORK(JNDEX+IBEST)
  160 CONTINUE
      IND(6)          = ISEED
C
 9999 RETURN
      END
      SUBROUTINE INITAL( WORK, IND )
C
C THIS SUBROUTINE INITIALIZES PARAMETERS FOR THE STOCHASTIC SEARCH
C
      DIMENSION WORK(1), IND(1)
      COMMON /INITL/ NPARM, NSERCH, NEVAL, NTABLE, NSTAT, ISEED, IDIST,
     1               IEXPNT, IPRINT, ITYPE, LT, JT, ICNVRG, TOL
      COMMON /INDEX/ INDEX2, INDEX3, INDEX4, INDEX5, INDEX6, INDEX7
      COMMON /LOOPS/ NLOOP
C
      NPARM           = IND(1)
      NSERCH          = IND(2)
      NEVAL           = IND(3)
      NTABLE          = IND(4)
      NSTAT           = IND(5)
      ISEED           = IND(6)
      IDIST           = IND(7)
      IEXPNT          = IND(8)
      IPRINT          = IND(9)
      ITYPE           = IND(10)
      INDEX2          = 2 * NPARM
      INDEX3          = 3 * NPARM
      INDEX4          = 4 * NPARM
      INDEX5          = INDEX4 + NTABLE
      INDEX6          = INDEX5 + NTABLE
      INDEX7          = INDEX6 + NTABLE * NPARM
      NLOOP           = NPARM / 7
      IF( 7 * NLOOP .NE. NPARM )        NLOOP = 1 + NLOOP
```

```
C
      LT              = NTABLE
      JT              = NTABLE
      ICNVRG          = 0
      TOL             = 10. ** FLOAT( IEXPNT )
      IF( NTABLE .LE. 1 )              NTABLE = 2
      IF( NSTAT .LE. 1 )               NSTAT = 2
      IF( NSTAT .GT. NTABLE )          NSTAT = NTABLE
      IF( ISEED .LE. 0 )               ISEED = 1
C
C SET UP K, G, AND P TABLES
C
      DO 20 I=1,NTABLE
      WORK(INDEX4+I) = I
   20 CONTINUE
      DO 40 I=1,NTABLE
      WORK(INDEX5+I) = 0.0
   40 CONTINUE
      LEMENT          = NTABLE * NPARM
      DO 60 I=1,LEMENT
      WORK(INDEX6+I) = 0.0
   60 CONTINUE
 9999 RETURN
      END
      SUBROUTINE STAT( WORK, ISERCH )
C
C THIS SUBROUTINE GENERATES THE STATISTICS REQUIRED FOR THE SEARCH
C
      DIMENSION WORK(1)
      COMMON /INITL/ NPARM, DUM1(2), NTABLE, NSTAT, DUM2(9)
      COMMON /INDEX/ INDEX2, INDEX3, INDEX4, INDEX5, INDEX6, INDEX7
C
      IF( ISERCH .GT. 1 )               GO TO 40
C
C FIRST SEARCH - CALCULATE MEAN AND STANDARD DEVIATION FROM INPUT
C  PARAMETER BOUNDS
C
      DO 20 I=1,NPARM
      BL              = WORK(I)
      BR              = WORK(NPARM+I)
      WORK(INDEX2+I) = .5 * ( BR + BL )
      WORK(INDEX3+I) = .5 * ( BR - BL )
   20 CONTINUE
                                        GO TO 9999
C
C DETERMINE MEAN AND STANDARD DEVIATION VALUES FROM P TABLE
C
   40 DO 160 J=1,NPARM
      JNDEX           = INDEX6 + NTABLE * ( J - 1 )
      PAVG            = WORK(JNDEX+1)
      DO 120 I=2,NSTAT
      PAVG            = WORK(JNDEX+I) + PAVG
  120 CONTINUE
```

```
      PAVG            = PAVG / NSTAT
      II           = IFIX( WORK(INDEX4+J) + .3 )
      WORK(INDEX2+J) = WORK(JNDEX+II)
      PSIGMA       = 0.0
      DO 140 I=1,NSTAT
      PSIGMA       = ( WORK(JNDEX+I) - PAVG ) ** 2 + PSIGMA
  140 CONTINUE
      WORK(INDEX3+J) = SQRT( PSIGMA / ( NSTAT - 1 ) )
  160 CONTINUE
C
 9999 RETURN
      END
      SUBROUTINE RANDOM( WORK )
C
C THIS SUBROUTINE GENERATES RANDOM VALUES USED IN THIS EVALUATION
C
      DIMENSION WORK(1)
      COMMON /INITL/ NPARM, DUM1(2), NTABLE, NSTAT, ISEED, IDIST,
     1              DUM2(7)
      COMMON /INDEX/ INDEX2, INDEX3, DUM3(3), INDEX7
      COMMON /PROBES/ IWALL, ISAME, DUM4(2)
      COMMON /CNSINT/ DUM5(3), B, DUM6(5)
      COMMON /MIKES/  NDIM, X(50), R(50), THETA(50)
      COMMON /BOUNDS/ DUM7(301), XB, DUM8(5)
C
      DO 340 J=1,NPARM
      IF( ISAME .GT. 0 )                  GO TO 120
C
C MODE NUMBER IS NOT THE SAME AS THE FIRST MODE NUMBER
C
      IF( IWALL )                         40, 20, 60
   20 IF( J .LE. 2 )                      GO TO 280
      WORK(INDEX7+J) = B
                                          GO TO 9999
   40 IF( J .GT. 1 )                      GO TO 280
      WORK(INDEX7+J) = XB
                                          GO TO 340
   60 IF( J - 2 )                         80, 100, 280
   80 WORK(INDEX7+J) = X(NDIM-1)
                                          GO TO 340
  100 WORK(INDEX7+J) = THETA(NDIM-1)
                                          GO TO 340
C
C MODE NUMBER IS THE SAME AS THE FIRST MODE NUMBER
C
  120 IF( IWALL )                         140, 160, 220
  140 IF( J .GT. 1 )                      GO TO 280
      WORK(INDEX7+J) = XB
                                          GO TO 340
  160 IF( J - 2 )                         280, 180, 200
  180 WORK(INDEX7+J) = 0.0
                                          GO TO 340
  200 WORK(INDEX7+J) = B
```

```
                                        GO TO 9999
  220 IF( J - 2 )                       240, 260, 280
  240 WORK(INDEX7+J) = X(NDIM-1)
                                        GO TO 340
  260 WORK(INDEX7+J) = THETA(NDIM-1)
                                        GO TO 340
C
C GENERATE RANDOM VARIABLE
C
  280 CALL STVAR( IDIST, 1., XX, ISEED )
      XX             - WORK(INDEX2+J) + XX * WORK(INDEX3+J)
C
C CHECK THAT THE VALUE OF XX IS WITHIN REQUIRED BOUNDS. IF NOT,
C  REGENERATE A NEW VALUE OF XX
C
      IF( XX - WORK(J) )                280, 300, 300
  300 IF( XX - WORK(NPARM+J) )          320, 320, 280
  320 WORK(INDEX7+J) = XX
  340 CONTINUE
C
 9999 RETURN
      END
      SUBROUTINE UPDATE( WORK, GP )
C
C THIS SUBROUTINE UPDATES THE G TABLE WITH THE CURRENT VALUE OF GP
C
      DIMENSION WORK(1)
      COMMON /INITL/ NPARM, DUM1(2), NTABLE, DUM2(5), ITYPE, LT, JT,
     1               DUM3(2)
      COMMON /INDEX/ INDEX2, INDEX3, INDEX4, INDEX5, INDEX6, INDEX7
C
      LASTG          = IFIX( WORK(INDEX4+NTABLE) + .3 )
      GTEST          = WORK(INDEX5+LASTG) - GP
      IF( ITYPE .GT. 0 )                GTEST = GP - WORK(INDEX5+LASTG)
C
C COMPARE GP WITH THE K(NTABLE) ELEMENT OF THE G TABLE
C
      IF( GTEST )                       9999, 9999, 20
   20 KT             = LASTG
      WORK(INDEX5+LASTG) = GP
      DO 40 I=1,NPARM
      JNDEX          = INDEX6 + NTABLE * ( I - 1 ) + LASTG
      WORK(JNDEX)    = WORK(INDEX7+I)
   40 CONTINUE
C
C DETERMINE POSITION OF GP IN G TABLE
C
   60 JT             = JT - 1
      NEXTG          = IFIX( WORK(INDEX4+JT) + .3 )
      GTEST          = WORK(INDEX5+NEXTG) - GP
      IF( ITYPE .GT. 0 )                GTEST = GP - WORK(INDEX5+NEXTG)
      IF( GTEST )                       100, 100, 80
   80 IF( JT .GT. 1 )                   GO TO 60
```

,GINAL PAGE IS
POOR QUALITY


```
      JT             = 0
                                               GO TO 120
  100 IF( JT .EQ. LT-1 )                       GO TO 160
C
C UPDATE K (INDEX) ARRAY TO REFLECT ENTRY OF GP INTO THE G TABLE
C
  120 WORK(INDEX4+LT) = WORK(INDEX4+LT-1)
      IF( JT .GE. LT-2 )                       GO TO 140
      LT             = LT - 1
                                               GO TO 120
  140 WORK(INDEX4+LT-1) = KT
  160 LT             = NTABLE
      JT             = NTABLE
 9999 RETURN
      END
      SUBROUTINE CONVRG( WORK )
C
C THIS SUBROUTINE TESTS FOR THE CONVERGENCE OF THE INDEPENDENT PARA-
C  METERS
C
      DIMENSION WORK(1)
      COMMON /INITL/ NPARM, DUM1(2), NTABLE, DUM2(8), ICNVRG, TOL
      COMMON /INDEX/ DUM3(4), INDEX6, INDEX7
      COMMON /PROBES/ IWALL, ISAME, DUM4(2)
C
      DO 100 I=1,NPARM
      IF( ISAME .GT. 0 )                       GO TO 15
C
C MODE NUMBER IS NOT EQUAL TO THE FIRST MODE NUMBER
C
      IF( IWALL )                              35, 5, 10
    5 IF( I .LE. 2 )                           GO TO 35
                                               GO TO 100
   10 IF( I .LE. 2 )                           GO TO 100
                                               GO TO 35
C
C MODE NUMBER IS EQUAL TO FIRST MODE NUMBER
C
   15 IF( IWALL )                              20, 25, 30
   20 IF( I .EQ. 2 )                           GO TO 100
                                               GO TO 35
   25 IF( I .GE. 2 )                           GO TO 100
                                               GO TO 35
   30 IF( I .LE. 2 )                           GO TO 100
C
C CALCULATE MINIMUM AND MAXIMUM P VALUES
C
   35 JNDEX          = INDEX6 + NTABLE * ( I - 1 )
      PMIN           = WORK(JNDEX+1)
      PMAX           = PMIN
      DO 80 J=2,NTABLE
      P              = WORK(JNDEX+J)
      IF( P - PMIN )                           40, 45, 45
```

```
   40 PMIN          = P                            GO TO 80
   45 IF( P - PMAX )                               80, 80, 60
   60 PMAX          = P
   80 CONTINUE
C
      TEST          = ABS( PMAX - PMIN ) / AMAX1( 1., ABS( PMAX + PMIN ))
      IF( TEST - TOL )                             100, 100, 9999
  100 CONTINUE
C
C CONVERGENCE CRITERIUM MET FOR ALL PARAMETERS, SET CONVERGENCE
C  INDICATOR
C
      ICNVRG        = 1
 9999 RETURN
      END
      SUBROUTINE SIVAR( J, C, X, IB )
C
C THIS SUBROUTINE DETERMINES RANDOM VARIABLES FROM EITHER A NORMAL OR AN
C  EXPONENTIAL DISTRIBUTION
C
      IF( J .GT. 0 )                               GO TO 160
C
C NORMAL DISTRIBUTION
C
      K             = 0                            GO TO 40
   20 K             = 1 + K
   40 CALL RAND( R0, IB )
      R1            = R0
   60 CALL RAND( R2, IB )
      IF( R2 - R1 )                                80, 100, 100
   80 CALL RAND( R1, IB )
      IF( R1 - R2 )                                60, 20, 20
  100 CALL RAND( R2, IB )
      FK            = K
      IF( R2 - .5 )                                120, 140, 140
  120 X             = C * ( FK + R0 )              GO TO 9999
  140 X             = -C * ( FK + R0 )             GO TO 9999
C
C EXPONENTIAL DISTRIBUTION
C
  160 K             = 0                            GO TO 200
  180 K             = 1 + K
  200 CALL RAND( R0, IB )
      R1            = R0
  220 CALL RAND( R2, IB )
      IF( R2 - R1 )                                240, 260, 260
  240 CALL RAND( R1, IB )
      IF( R1 - R2 )                                220, 180, 180
```

```
  260 FK             = K
      X              = FK + RO
      G              = EXP( X - .5 * X * X - .5 )
      CALL RAND( R2, IB )
      IF( R2 - G )                   280, 280, 160
  280 CALL RAND( R2, IB )
      IF( R2 - .5 )                  300, 300, 320
  300 X              = C * X
                                     GO TO 9999
  320 X              = -C * X
C
 9999 RETURN
      END
      SUBROUTINE RAND( YFL, IX )
C
C THIS SUBROUTINE CALCULATES RANDOM NUMBERS USED BY THE NORMAL OR
C  EXPONENTIAL DISTRIBUTION
C
      IY             = IX * 65539
      IF( IY .LT. 0 )                IY = 2147483647 + 1 + IY
      YFL            = IY
      YFL            = .4656613E-9 * YFL
      IX             = IY
 9999 RETURN
      END
      SUBROUTINE PRINTT( WORK, ISERCH )
C
C THIS SUBROUTINE PRINTS THE G TABLE AND ASSOCIATED PARAMETER VALUES FOR
C  SEARCHES
C
      DIMENSION WORK(1), KTYPE(2), KDIST(3,2), IVAR(3), SDEV(3)
      COMMON /INITL/ NPARM, NSERCH, NEVAL, NTABLE, NSTAT, ISEED, IDIST,
     1               IEXPNT, IPRINT, ITYPE, DUM1(3), TOL
      COMMON /INDEX/ INDEX2, INDEX3, INDEX4, INDEX5, INDEX6, INDEX7
      COMMON /LOOPS/  NLOOP
      COMMON /SEED/   JSEED
      COMMON /ANGLES/ DEGRAD, RADDEG
      DATA KTYPE / 4HMINI, 4HMAXI /
      DATA KDIST / 4H  GA, 4HUSSI, 4HAN  , 4HEXPO, 4HNENT, 4HIAL  /
      DATA IVAR / 4H  X , 4HTHET, 4H  R  /
C
      IF( ISERCH .NE. 1 .OR. IPRINT .NE. 2 )  GO TO 20
C
C PRINT INPUT PARAMETERS
C
      WRITE(6,1000) KTYPE(ITYPE+1), ( KDIST(I,IDIST+1),I=1,3 ), TOL,
     1                NPARM, NTABLE, NSTAT, NSERCH, NEVAL, JSEED
 1000 FORMAT( //, T36, 'A ', A4, 'MUM IS BEING SOUGHT', //, T36,
     1         'A ', 3A4, 'DISTRIBUTION IS BEING USED', //, T36,
     2         'TOLERANCE FOR CONVERGENCE', T84, '= ', G10.4, //, T36,
     3         'NUMBER OF PARAMETERS IN FUNCTION', T84, '= ', I6, //, T36
     4,        'NUMBER OF ELEMENTS IN DETERMINANT TABLE', T84, '= ', I6,
     5//, T36,'NUMBER OF ELEMENTS USED TO GENERATE STATISTICS  =', I6,//
```

```
     6           T36, 'MAXIMUM NUMBER OF SEARCHES', T84, '= ', I6, //,
     7           T36, 'NUMBER OF EVALUATIONS OF DETERMINANT PER SEARCH', T
     8 84, '= ',I6, //, T36, 'FIRST RANDOM NUMBER', T84, '= ', Z8 )
C
C
C PRINT STANDARD DEVIATIONS IF PRINTING AFTER EACH SEARCH
C
   20 IF( IPRINT .NE. 2 )                      GO TO 30
      DO 25 I=1,NPARM
      SDEV(I)          = WORK(INDEX3+I)
   25 CONTINUE
      SDEV(2)          = RADDEG * SDEV(2)
      WRITE(6,1004) ( IVAR(I), SDEV(I),I=1,NPARM )
 1004 FORMAT( //, T62, 'STANDARD DEVIATION', (/,52X,A4,6X,G15.7) )
C
C PRINT C TABLE AND PARAMETER VALUES
C
   30 WRITE(6,1001) ISERCH
 1001 FORMAT( //, T44, 'RESULTS OF THE DETERMINANT TABLE AFTER SEARCH',
     1        I5 )
      WRITE(6,1002) ( IVAR(I),I=1,3 )
 1002 FORMAT( //, 6X, 'I', 7X, 'DETERMINANT', 2X, 7(5X,A4,6X), / )
      DO 40 I=1,NTABLE
      K               = IFIX( WORK(INDEX4+I) + .3 )
      ANGLE           = WORK(INDEX6+K+NTABLE) * RADDEG
      WRITE(6,1003) I, WORK(INDEX5+K), WORK(INDEX6+K), ANGLE,
     1             WORK(INDEX6+K+2*NTABLE)
 1003 FORMAT( 4X, I3, 5X, 8G15.7 )
   40 CONTINUE
 9999 RETURN
      END
      FUNCTION DETCAL( IND, A, IE )
      DIMENSION A(1)
      COMPLEX ONE, ZERO, DET, MATRIX, FACTR, DIVSR, DETT, KX, EXPNT
      COMMON /MATRX / NDIM1, ONE, ZERO, DET, MATRIX(50,50)
      COMMON /CNSTNT/ NMIKES, NMODES, SIGMA, B, DUM1(4)
      COMMON /PRODES/ IWALL, DUM3(3)
      COMMON /MODES/  MODE(50), MU(50), IWAVE(50)
      COMMON /EMUS/   EMU(50,50), IEMPRT
      COMMON /BOUNDS/ XBOUND(2,50), RBOUND(2,50), TBOUND(2,50), XA, XB,
     1                THMIN, THMAX, XLIM, RLIM, THLIM
      COMMON /MIKES/  NDIM, X(50), R(50), THETA(50)
      COMMON /BESSL/  DUM2(3), TOL, M, PI
      COMMON /WAVENO/ KX(50)
C
      IE              = 0
      IF( IND .NE. 0 )                         GO TO 9999
C
C CHECK THAT THE CURRENT MICROPHONE LOCATION IS SUFFICIENTLY REMOVED
C FROM PREVIOUS MICROPHONE LOCATIONS
C
      DO 15 I=1,NDIM1
      IF( ABS( A(I) - X(I) ) - XLIM )  5, 5, 15
```

```
    5 IF( ABS( A(2) - THETA(I) ) - THLIM )  10, 10, 15
   10 IF( IWALL .LE. 0 )                 GO TO 9998
      IF( ABS( A(3) - R(I) ) - RLIM )  9998, 9998, 15
   15 CONTINUE
C
C CALCULATE NDIM CHARACTERISTIC E-FUNCTION VALUES FOR CURRENT VALUE OF K
C
      RPRIME       = A(3) / B
      CALL EMUCAL( RPRIME, EMU(1,NDIM), NDIM )
C
C SET UP ROW NDIM FROM CURRENT VALUES OF X, R, AND THETA
C
      DO 20 J=1,NDIM
      EXPNT          = CMPLX( 0.0, REAL( KX(J) ) * A(1) + MODE(J) * A(2) )
      MATRIX(NDIM,J) = EMU(J,NDIM) * CEXP( EXPNT ) * EXP( -A(1) *
     1                 AIMAG( KX(J) ) )
   20 CONTINUE
C
C CALCULATE VALUE OF DETERMINANT FOR CURRENT PARAMETERS
C
      FACTR          = ONE
      DO 60 I=1,NDIM1
      FACTR          = - ( MATRIX(NDIM,I) * CONJG( MATRIX(I,I) ) /
     1                 CABS( MATRIX(I,I) ) ** 2 ) * FACTR
      DIVSR          = - MATRIX(I,I) * CONJG( MATRIX(NDIM,I) ) /
     1                 CABS( MATRIX(NDIM,I) ) ** 2
      MATRIX(NDIM,I) = ZERO
      JJ             = I + 1
      DO 40 J=JJ,NDIM
      MATRIX(NDIM,J) = MATRIX(I,J) + DIVSR * MATRIX(NDIM,J)
   40 CONTINUE
   60 CONTINUE
C
      DETT           = MATRIX(NDIM,NDIM) * FACTR * DET
      DETCAL         = CABS( DETT )
                                          GO TO 9999
 9998 IE             = 1
 9999 RETURN
      END
***** ABOVE ACTION SATISFACTORILY COMPLETED *****
```

APPENDIX C
MICROPHONE LOCATION PROGRAM
SAMPLE CASE

*** MICROPHONE LOCATION COMPUTER PROGRAM ***

... STOCHASTIC SEARCH OUTPUT FOR MICROPHONE 2

A MAXIMUM IS BEING SOUGHT

A GAUSSIAN DISTRIBUTION IS BEING USED

TOLERANCE FOR CONVERGENCE	=	.1000E-01
NUMBER OF PARAMETERS IN FUNCTION	=	3
NUMBER OF ELEMENTS IN DETERMINANT TABLE	=	30
NUMBER OF ELEMENTS USED TO GENERATE STATISTICS	=	30
MAXIMUM NUMBER OF SEARCHES	=	50
NUMBER OF EVALUATIONS OF DETERMINANT PER SEARCH	=	500
FIRST RANDOM NUMBER	=	00000001

	STANDARD DEVIATION
X	15.00000
THET	120.0000
R	7.000000

RESULTS OF THE DETERMINANT TABLE AFTER SEARCH 1

I	DETERMINANT	X	THET	R
1	.1257972	29.96037	.0	25.00000
2	.1257714	29.95030	.0	25.00000
3	.1256930	29.92006	.0	25.00000
4	.1253359	29.78271	.0	25.00000
5	.1252040	29.73334	.0	25.00000
6	.124[illegible]	29.58609	.0	25.00000
7	.124[illegible]	29.51088	.0	25.00000
8	.1245966	29.50372	.0	25.00000
9	.1244661	29.46208	.0	25.00000
10	.1243634	29.41608	.0	25.00000
11	.1242754	29.28353	.0	25.00000
12	.1240705	29.30722	.0	25.00000
13	.1240295	29.25175	.0	25.00000
14	.123[illegible]	29.20473	.0	25.00000
15	.123[illegible]	29.12645	.0	25.00000
16	.123[illegible]	29.05786	.0	25.00000
17	.1229719	28.90337	.0	25.00000
18	.122[illegible]	28.86852	.0	25.00000
19	.122[illegible]	28.72243	.0	25.00000
20	.122[illegible]	28.49878	.0	25.00000
21	.121[illegible]	28.45319	.0	25.00000
22	.121[illegible]	28.41350	.0	25.00000
23	.120[illegible]	28.39080	.0	25.00000
24	.1207814	28.12362	.0	25.00000
25	.120[illegible]	28.08293	.0	25.00000

ORIGINAL PAGE IS
OF POOR QUALITY

26	.1204733	28.01587	.0	25.00000
27	.1204017	27.99095	.0	25.00000
28	.1202583	27.94125	.0	25.00000
29	.1201088	27.88945	.0	25.00000
30	.1169528	27.49429	.0	25.00000

	STANDARD DEVIATION
X	.7230805
THET	.0
R	.0

RESULTS OF THE DETERMINANT TABLE AFTER SEARCH 2

I	DETERMINANT	X	THET	R
1	.1258985	29.99931	.0	25.00000
2	.1258861	29.99536	.0	25.00000
3	.1258847	29.99394	.0	25.00000
4	.1258824	29.99313	.0	25.00000
5	.1258801	29.99214	.0	25.00000
6	.1258789	29.99168	.0	25.00000
7	.1258663	29.98888	.0	25.00000
8	.1258639	29.98586	.0	25.00000
9	.1258577	29.98355	.0	25.00000
10	.1258566	29.98331	.0	25.00000
11	.1258526	29.98154	.0	25.00000
12	.1258502	29.98059	.0	25.00000
13	.1258475	29.97963	.0	25.00000
14	.1258473	29.97955	.0	25.00000
15	.1258451	29.97881	.0	25.00000
16	.1258398	29.97678	.0	25.00000
17	.1258266	29.97150	.0	25.00000
18	.1258237	29.97061	.0	25.00000
19	.1258146	29.96684	.0	25.00000
20	.1258128	29.96635	.0	25.00000
21	.1258072	29.96402	.0	25.00000
22	.1257991	29.96091	.0	25.00000
23	.1257959	29.96080	.0	25.00000
24	.1257973	29.96057	.0	25.00000
25	.1257967	29.96010	.0	25.00000
26	.1257932	29.95866	.0	25.00000
27	.1257921	29.95831	.0	25.00000
28	.1257899	29.95738	.0	25.00000
29	.1257840	29.95534	.0	25.00000
30	.1257815	29.95410	.0	25.00000

... MICROPHONE 2 ...

CONDITION NUMBER = 1.1713

MODE	EIGENVALUES AMPLITUDE	PHASE
1	0.3840	3.6088
2	0.3279	-62.3158

*** MICROPHONE LOCATION COMPUTER PROGRAM ***

--- STOCHASTIC SEARCH OUTPUT FOR MICROPHONE 3

A MAXIMUM IS BEING SOUGHT

A GAUSSIAN DISTRIBUTION IS BEING USED

TOLERANCE FOR CONVERGENCE	=	.1000E-01
NUMBER OF PARAMETERS IN FUNCTION	=	3
NUMBER OF ELEMENTS IN DETERMINANT TABLE	=	50
NUMBER OF ELEMENTS USED TO GENERATE STATISTICS	=	30
MAXIMUM NUMBER OF SEARCHES	=	50
NUMBER OF EVALUATIONS OF DETERMINANT PER SEARCH	=	500
FIRST RANDOM NUMBER	=	40760C93

	STANDARD DEVIATION
X	15.00000
THET	120.0000
R	7.000000

RESULTS OF THE DETERMINANT TABLE AFTER SEARCH 1

I	DETERMINANT	X	THET	R
1	.2213760E-01	14.99974	.0	25.00000
2	.2213760E-01	14.99685	.0	25.00000
3	.2213758E-01	15.01359	.0	25.00000
4	.2213758E-01	15.00974	.0	25.00000
5	.2213749E-01	14.96857	.0	25.00000
6	.2213749E-01	15.02761	.0	25.00000
7	.2213734E-01	14.95290	.0	25.00000
8	.2213719E-01	14.93913	.0	25.00000
9	.2213702E-01	14.92862	.0	25.00000
10	.2213547E-01	15.14271	.0	25.00000
11	.2213549E-01	15.14867	.0	25.00000
12	.2213481E-01	15.16328	.0	25.00000
13	.2213424E-01	15.18005	.0	25.00000
14	.2213420E-01	15.18047	.0	25.00000
15	.2213350E-01	14.79559	.0	25.00000
16	.2213320E-01	15.20515	.0	25.00000
17	.2213196E-01	14.76641	.0	25.00000
18	.2213168E-01	14.76126	.0	25.00000
19	.2213088E-01	14.74622	.0	25.00000
20	.2213076E-01	15.25728	.0	25.00000
21	.2212992E-01	15.27172	.0	25.00000
22	.2212970E-01	15.27469	.0	25.00000
23	.2212626E-01	14.70084	.0	25.00000
24	.2212578E-01	14.66279	.0	25.00000
25	.2212311E-01	14.[illegible]	.0	25.00000

26	.2212261E-01	15.37891	.0	25.00000
27	.2211847E-01	14.57029	.0	25.00000
28	.2211582E-01	14.54162	.0	25.00000
29	.2211365E-01	15.47976	.0	25.00000
30	.2211104E-01	14.49424	.0	25.00000

	STANDARD DEVIATION
X	.2604912
THET	.0
R	.0

RESULTS OF THE DETERMINANT TABLE AFTER SEARCH 2

I	DETERMINANT	X	THET	R
1	.2213763E-01	14.99976	.0	25.00000
2	.2213763E-01	15.00026	.0	25.00000
3	.2213763E-01	15.00675	.0	25.00000
4	.2213762E-01	15.00097	.0	25.00000
5	.2213762E-01	15.00097	.0	25.00000
6	.2213760E-01	15.00254	.0	25.00000
7	.2213760E-01	14.99989	.0	25.00000
8	.2213760E-01	14.99974	.0	25.00000
9	.2213760E-01	14.99685	.0	25.00000
10	.2213760E-01	14.99913	.0	25.00000
11	.2213760E-01	15.00025	.0	25.00000
12	.2213759E-01	14.98456	.0	25.00000
13	.2213758E-01	15.01359	.0	25.00000
14	.2213758E-01	15.00974	.0	25.00000
15	.2213758E-01	14.99763	.0	25.00000
16	.2213758E-01	15.00692	.0	25.00000
17	.2213758E-01	14.98695	.0	25.00000
18	.2213758E-01	14.98900	.0	25.00000
19	.2213757E-01	14.97910	.0	25.00000
20	.2213757E-01	15.02236	.0	25.00000
21	.2213757E-01	15.00048	.0	25.00000
22	.2213757E-01	14.98255	.0	25.00000
23	.2213757E-01	15.00695	.0	25.00000
24	.2213757E-01	14.98425	.0	25.00000
25	.2213757E-01	14.99131	.0	25.00000
26	.2213756E-01	15.01321	.0	25.00000
27	.2213756E-01	14.98751	.0	25.00000
28	.2213756E-01	14.97903	.0	25.00000
29	.2213755E-01	15.02011	.0	25.00000
30	.2213755E-01	15.01502	.0	25.00000

... MICROPHONE 3 ...

CONDITION NUMBER = 2.4900

MODE	EIGENVALUES AMPLITUDE	PHASE
1	0.4767	-20.9473
2	0.2426	74.1221
3	0.1914	54.3685

*** MICROPHONE LOCATION COMPUTER PROGRAM ***

... INPUT VARIABLES ...

NUMBER OF MICROPHONE LOCATIONS = 3　　　NUMBER OF (MODE,MU) SETS = 3

... INPUT MODES ...

MODE	CIRCUMFERENTIAL MODE NUMBER	RADIAL ORDER	WAVE INDICATOR	AXIAL WAVE NUMBER REAL	AXIAL WAVE NUMBER IMAGINARY	KMU
1	-4	2	1	65.0738	0.0	12.9210
2	-4	1	1	69.0154	0.0	8.7800
3	-4	0	1	71.0845	0.0	5.2510

... TEST GEOMETRY AND CONDITIONS ...

HUB / TIP RATIO = 0.440　　　OUTER RADIUS OF DUCT = 25.000　　　AXIAL MACH NUMBER = -0.100
FREQUENCY = 6200.000　　　SPEED OF SOUND = 34345.00　　　RADIAN FREQUENCY = 38955.750

... STOCHASTIC SEARCH VALUES ...

NUMBER OF VARIABLES = 3　　　MAXIMUM NUMBER OF SEARCHES = 50　　　NUMBER OF EVALUATIONS PER SEARCH = 500

PARAMETER BOUNDS

MICROPHONE	MINIMUM X	MAXIMUM X	MINIMUM R	MAXIMUM R	MINIMUM THETA	MAXIMUM THETA
2	0.0	30.000	11.000	25.000	0.0	240.000
3	0.0	30.000	11.000	25.000	0.0	240.000

*** MICROPHONE LOCATION COMPUTER PROGRAM ***

... CALCULATED MICROPHONE LOCATIONS ...

MICROPHONE	X	R	THETA
1	0.0	25.000000	0.0
2	29.999313	25.000000	0.0
3	14.999759	25.000000	0.0

... CHARACTERISTIC E-FUNCTION VALUES (FINAL SOLUTION VALUES) ...

LOCATION	MODES 1	2	3
1	0.233	-0.315	0.404
2	0.233	-0.315	0.404
3	0.233	-0.315	0.404

*** MICROPHONE LOCATION COMPUTER PROGRAM ***

... MICROPHONE 1 ...

CONDITION NUMBER = 1.0403

MODE	EIGENVALUES AMPLITUDE	EIGENVALUES PHASE
1	0.2332	0.0

... MICROPHONE 2 ...

CONDITION NUMBER = 1.0652

MODE	EIGENVALUES AMPLITUDE	EIGENVALUES PHASE
1	0.2317	41.3619
2	0.2135	-0.9155

... MICROPHONE 3 ...

CONDITION NUMBER = 2.6302

MODE	EIGENVALUES AMPLITUDE	EIGENVALUES PHASE
1	0.2193	-77.1510
2	0.2935	51.0947
3	0.1116	7.7926

*** MICROPHONE LOCATION COMPUTER PROGRAM ***

... INPUT VARIABLES ...

NUMBER OF MICROPHONE LOCATIONS = 3 NUMBER OF (MODE,MU) SETS = 3

... INPUT MODES ...

MODE	CIRCUMFERENTIAL MODE NUMBER	RADIAL ORDER	WAVE INDICATOR	AXIAL WAVE NUMBER REAL	AXIAL WAVE NUMBER IMAGINARY	KMU
1	-4	2	1	65.0738	0.0	12.9210
2	-4	1	1	69.0154	0.0	6.7800
3	-4	0	1	71.0845	0.0	5.2510

... TEST GEOMETRY AND CONDITIONS ...

HUB / TIP RATIO = 0.440 OUTER RADIUS OF DUCT = 25.000 AXIAL MACH NUMBER = -0.100
FREQUENCY = 6200.000 SPEED OF SOUND = 34345.00 RADIAN FREQUENCY = 38955.750

... STOCHASTIC SEARCH VALUES ...

NUMBER OF VARIABLES = 3 MAXIMUM NUMBER OF SEARCHES = 50 NUMBER OF EVALUATIONS PER SEARCH = 500

MICROPHONE	PARAMETER BOUNDS MINIMUM X	MAXIMUM X	MINIMUM R	MAXIMUM R	MINIMUM THETA	MAXIMUM THETA
2	0.0	0.0	0.0	0.0	0.0	0.0
3	0.0	0.0	0.0	0.0	0.0	0.0

*** MICROPHONE LOCATION COMPUTER PROGRAM ***

... CALCULATED MICROPHONE LOCATIONS ...

MICROPHONE	X	R	THETA
1	0.0	25.000000	0.0
2	10.000000	25.000000	0.0
3	20.000000	25.000000	0.0

... CHARACTERISTIC E-FUNCTION VALUES (FINAL SOLUTION VALUES) ...

LOCATION	MODES 1	2	3
1	0.233	-0.315	0.404
2	0.233	-0.315	0.404
3	0.233	-0.315	0.404