NASA TECHNICAL NOTE

NASA TN D-7344

NASA TN D-7344

FORTRAN PROGRAM FOR CALCULATING VELOCITIES AND STREAMLINES ON THE HUB-SHROUD MID-CHANNEL FLOW SURFACE OF AN AXIAL- OR MIXED-FLOW TURBOMACHINE

II - Programmer's Manual

by Theodore Katsanis and William D. McNally

Lewis Research Center

Cleveland, Ohio 44135

NATIONAL AERONAUTICS AND SPACE ADMINISTRATION • WASHINGTON, D. C. • APRIL 1974

1. Report No.
NASA TN D-7344

2. Government Accession No.

3. Recipient's Catalog No.

4. Title and Subtitle
FORTRAN PROGRAM FOR CALCULATING VELOCITIES AND STREAMLINES ON THE HUB-SHROUD MID-CHANNEL FLOW SURFACE OF AN AXIAL- OR MIXED-FLOW TURBOMACHINE
II - PROGRAMMER'S MANUAL

5. Report Date
APRIL 1974

6. Performing Organization Code

7. Author(s)
Theodore Katsanis and William D. McNally

8. Performing Organization Report No.
E-7516

10. Work Unit No.
501-24

9. Performing Organization Name and Address
Lewis Research Center
National Aeronautics and Space Administration
Cleveland, Ohio 44135

11. Contract or Grant No.

12. Sponsoring Agency Name and Address
National Aeronautics and Space Administration
Washington, D. C. 20546

13. Type of Report and Period Covered
Technical Note

14. Sponsoring Agency Code

15. Supplementary Notes

16. Abstract
A FORTRAN-IV computer program, MERIDL, has been developed that obtains a subsonic or shock-free transonic flow solution on the hub-shroud mid-channel flow surface of a turbomachine. The blade row may be fixed or rotating and may be twisted and leaned. Flow may be axial or mixed, up to 45° from axial. Upstream and downstream flow variables can vary from hub to shroud, and provision is made to correct for loss of stagnation pressure. The results include velocities, streamlines, and flow angles on the flow surface and approximate blade surface velocities. Subsonic solutions are obtained by a finite-difference stream-function solution. Transonic solutions are obtained by a velocity-gradient method, using information from a finite-difference stream-function solution at a reduced mass flow.

17. Key Words (Suggested by Author(s))
Meridional plane; Turbomachine; Mid-channel flow surface; Axial flow turbomachine; Mixed-flow turbomachine; Transonic flow

18. Distribution Statement
Unclassified - unlimited

Cat. 01

19. Security Classif. (of this report)
Unclassified

20. Security Classif. (of this page)
Unclassified

21. No. of Pages
212

22. Price*
$5.75

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

CONTENTS

FORTRAN PROGRAM FOR CALCULATING VELOCITIES AND STREAMLINES ON THE HUB-SHROUD MID-CHANNEL FLOW SURFACE OF AN AXIAL- OR MIXED-FLOW TURBOMACHINE

II - PROGRAMMER'S MANUAL

by Theodore Katsanis and William D. McNally
Lewis Research Center

SUMMARY

A FORTRAN-IV computer program, MERIDL, has been developed which obtains a subsonic or transonic, nonviscous flow solution on the hub-shroud mid-channel flow surface of a turbomachine. The flow must be essentially subsonic, but there may be locally supersonic flow. The solution is for two-dimensional, adiabatic shock-free flow. The blade row may be fixed or rotating and may be twisted and leaned. The flow may be axial or mixed, up to approximately 45° from axial. Upstream and downstream flow conditions can vary from hub to shroud, and provision is made for an approximate correction for loss of stagnation pressure.

The basic analysis is based on the stream function and consists of the solution of the simultaneous, nonlinear, finite-difference equations of the stream function. This basic solution, however, is limited to strictly subsonic flow. When there is locally supersonic flow, a transonic solution must be obtained. The transonic solution is obtained by a combination of a finite-difference stream-function solution and a velocity-gradient solution. The finite-difference solution at a reduced mass flow provides information which is used to obtain a velocity-gradient solution at the full mass flow.

The program input consists of blade and flow channel geometry, upstream and downstream flow conditions from hub to shroud, and mass flow. The output includes streamline coordinates, flow angles, and velocities on the mid-channel flow surface; incidence and deviation angles at the blade leading and trailing edges; and approximations to the blade surface velocities. The output may also include input information for a blade-to-blade flow analysis program.

The program is reported in two volumes with part I as the user's manual and part II as the programmer's manual. Part I contains all the information necessary to use the program as is. It explains the equations involved and the method of solution and gives a numerical example to illustrate the use of the program. This report, part II, contains all information necessary to understand the operation of the program. It explains the overall program procedure and gives a detailed description of all the subroutines. There is also a dictionary of variable names and a complete program listing.

INTRODUCTION

The design of blades for compressors and turbines ideally requires analysis methods for unsteady, rotational, three-dimensional, viscous flow through a turbomachine. Clearly, such solutions are impossible at the present time, even on the largest and fastest computers. The usual approach at present is to analyze only steady flows and to separate inviscid solutions from viscous solutions. Three-dimensional inviscid solutions are just beginning to be contemplated for coming generations of computers. So at present, inviscid analyses usually involve a combination of several two-dimensional solutions on intersecting families of stream surfaces to obtain what is called a quasi-three-dimensional solution.

Since there are several choices of two-dimensional surfaces to analyze and many ways of combining them, there are many approaches to obtaining a quasi-three-dimensional solution. Most two-dimensional solutions are either on a blade-to-blade surface of revolution (Wu's S_1 surface, ref. 1) or on the meridional or mid-channel stream surface between two blades (Wu's S_2 surface). However, when three-dimensional effects are most important, significant information can often be obtained from a solution on a passage cross-sectional surface (normal to the flow). This is called a channel solution (see fig. 1).

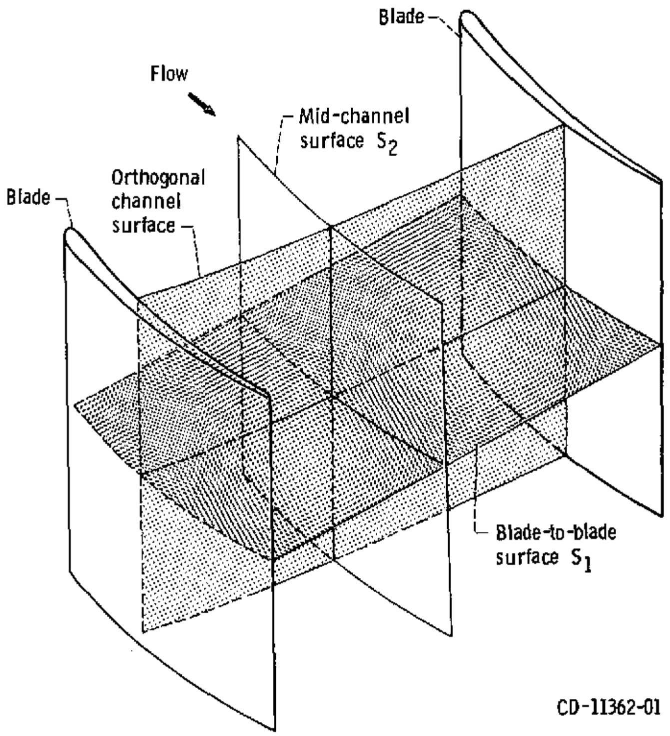


Figure 1. - Two-dimensional analysis surfaces in a turbomachine.

In this report a solution to the equations of flow on the meridional S_2 surface is carried out. This solution surface is chosen when the turbomachine under consideration has significant variation in flow properties in the hub-shroud direction. A solution on the meridional surface will show this variation. The solution can be obtained either by the quasi-orthogonal method, which solves the velocity-gradient equation from hub to shroud on the meridional flow plane (ref. 2), or by a finite-difference method, which solves a finite-difference equation for stream function on the same flow plane. The quasi-orthogonal method is efficient in many cases and can obtain solutions into the transonic regime. However, there is difficulty in obtaining a solution when aspect ratios are above 1. Difficulties are also encountered with curved passages and low hub-tip ratio blades. For such cases, the most promising method is the finite-difference solution, but this solution is limited to completely subsonic flows.

Two finite-difference programs for flow on the mid-channel surface of a turbomachine have been reported in the literature (refs. 3 and 4). Since both are finite-difference methods, they are necessarily limited to subsonic flow cases. Marsh's method (ref. 3), termed the matrix throughflow method, closely follows the development given by Wu in reference 1. However, the computer program was not included in reference 3, nor is it available to the general user. Davis' program is provided in reference 4 but is limited to certain families of compressor blades and flow surfaces.

The method described in this report uses both the finite-difference and the quasi-orthogonal (velocity gradient) methods, combined in a way which takes maximum advantage of both. The finite-difference method is used to obtain a subsonic flow solution. The velocity-gradient method is then used, if necessary, to extend the range of solutions into the transonic regime.

A computer program, MERIDL, has been written to perform these calculations. This program is written for axial- or mixed-flow turbomachines, both compressors and turbines, up to approximately 45° from axial. Upstream and downstream flow conditions can vary from hub to shroud. The solution is for compressible, shock-free flow, or incompressible flow. Provision is made for an approximate correction for loss of stagnation pressure through the blade row. The blade row may be either fixed or rotating and may be twisted and leaned. The blades can have high aspect ratio and arbitrary thickness distribution.

The solution obtained by this program also provides the information necessary for a more detailed blade shape analysis on blade-to-blade surfaces (fig. 1). A useful program for this purpose is TSONIC (ref. 5). Information needed to prepare all the input for TSONIC is calculated and printed by this program.

The MERIDL program has been implemented on the NASA Lewis time-sharing IBM-TSS/360-67 computer. For the numerical example of this report, storage of variables required 60 000 words for a 21 × 41 grid of 861 points. Variable storage could be easily reduced by equivalencing of variables or by using a coarser mesh. Storage for the program code is 18 000 words. This storage could be reduced by overlay of code. Run times for the program range from 3 to 15 minutes on IBM 360-67 equipment, depending upon the mesh size used and the compressibility of the flow.

The MERIDL program is reported in two volumes, with the user's manual presented as part I in reference 6 and the programmer's manual presented as part II in this report. Part I contains all the information necessary to use the program as is. It explains the method of solution and gives a numerical example to illustrate the use of the program. Part I includes the sections METHOD OF ANALYSIS, DESCRIPTION OF INPUT AND OUTPUT, NUMERICAL EXAMPLE, and appendixes which derive the mathematical equations used. This report, part II, contains all information necessary to understand the operation of the program. It explains the overall program procedure and gives a detailed description of all the subroutines. There is also a dictionary of variable names and a complete program listing. The appendixes explain numerical techniques used and derive certain numerical algorithms. So, part II includes the sections OVERALL PROGRAM PEOCEDURE, DETAILED PROGRAM PROCEDURE, MAIN DICTIONARY, PROGRAM LISTING, and appendixes which derive the numerical methods used.

OVERALL PROGRAM PROCEDURE

This main section gives an overall view of the program calculation procedure. The next main section should be consulted for the detailed program procedure. Reference will be made to the proper section or appendix for the equations and their derivation or for the numerical techniques used.

The main program guides the overall flow of the program. All the main subroutines are called by it. Figure 2 is a flow chart for the main program.

The first step is to read and print out all the input data. This is done by the INPUT subroutine. Upstream and downstream flow conditions can be given either as a function of the streamline or as a function of radius. For program calculations, both the stream function and the radius are needed. INPUT estimates values of either stream function or radius, whichever was not given as input, based on the area distribution. These values are later adjusted with each iteration. The next step is to call INPLOT, which plots all the upstream and downstream input flow variables as well as the input blade sections from hub to tip.

The next subroutine is MESHO, which calculates the coordinates of the orthogonal

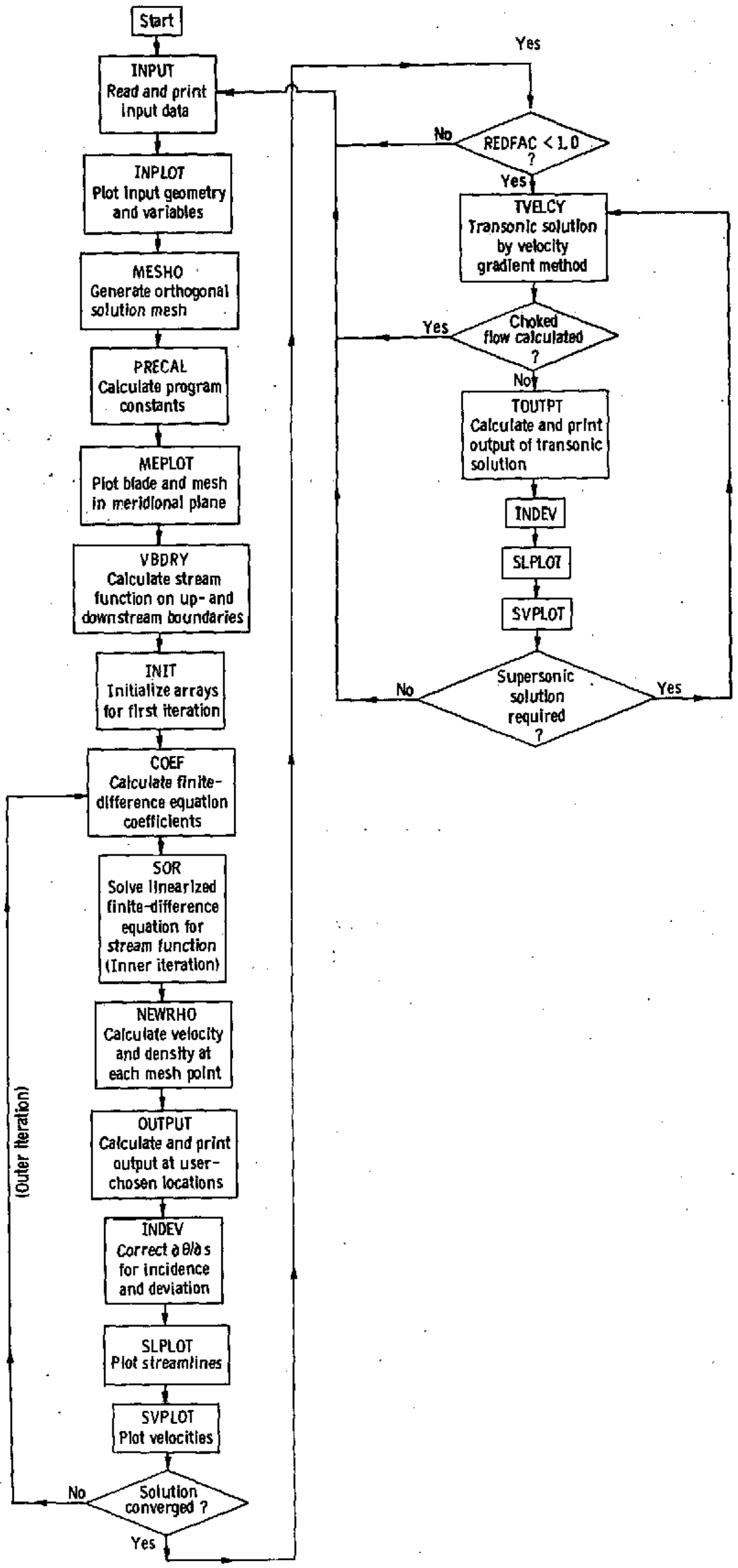


Figure 2. - Flow chart of main program.

mesh in the solution region. Details of the numerical technique are given in reference 7. After this PRECAL is called to calculate quantities which remain fixed throughout the calculations. These quantities include the s and t mesh coordinates, hub and tip wall curvatures, and leading- and trailing-edge z- and r-coordinates at horizontal mesh lines. Subroutine PRECAL also calls THETOM, THIKOM, and LOSSOM. Subroutine THETOM calculates $\partial\theta/\partial s$ and $\partial\theta/\partial t$ at the orthogonal mesh points. (All symbols are defined in appendix E.) These partials are used to calculate the blade flow angle β and the tangential velocity W_θ after the meridional velocity W_m has been calculated. Subroutine THIKOM calculates the tangential blade thickness t_θ at the orthogonal mesh points. Subroutine LOSSOM calculates the ratio of actual to ideal relative stagnation pressure downstream of the blade and then distributes the loss linearly through the blade row from leading to trailing edge. The method of making loss corrections is discussed in appendix D of part I (ref. 6). Finally, PRECAL makes corrections in mass flow, wheel speed, and whirl for the reduced-mass-flow solution if the full-mass-flow solution cannot be obtained directly (i.e., when REDFAC < 1.0).

Next MEPLOT is called to plot the meridional plane view of the blade and passage and to plot the orthogonal mesh. Then VBDRY is called to calculate the stream-function values along the upstream and downstream boundaries of the orthogonal mesh. This is done by using the velocity-gradient equation derived in appendix C of part I (ref. 6). Iteration is required to establish the correct temperature, density, and whirl to use in the velocity-gradient equation. Now INIT is called to initialize array variables as required for the first iteration. Most variables are set either to zero or to some value which will avoid division by zero later on.

At this point, everything is ready to solve the stream-function finite-difference equations. These equations are nonlinear. They are solved by an iterative procedure, with two levels of iteration. The inner iteration solves a linearized equation, and the outer iteration makes corrections to the linearized equation so that the solution converges to the solution of the original nonlinear equation. There are three subroutines called to obtain the solution to the linearized equation: COEF, SOR, and NEWRHO. Then there are four subroutines to print and plot this information and prepare for the next outer iteration: OUTPUT, INDEV, SLPLOT, and SVPLOT. These seven subroutines are repeated until convergence is obtained.

Subroutine COEF calculates the coefficients of the finite-difference equations. These coefficients are derived in appendix A. Because of the sensitivity of the calculations to the value of $\partial(rV_\theta)/\partial r$, this value is damped from iteration to iteration. Thus, only a portion of the predicted change in value is actually used. This portion is specified by the input value of DNEW.

Subroutine SOR solves the finite-difference equations for the stream function u by successive overrelaxation using an optimum overrelaxation factor (ORF). This is the inner iteration. The optimum overrelaxation factor is calculated by subroutine SOR on the first iteration. Subroutine NEWRHO calculates velocity components at each mesh point by differentiating the stream function numerically along the orthogonal mesh lines. These values are used to calculate new densities at each mesh point. When whirl is not given as input, NEWRHO also makes reinitialization calls to readjust the estimated values of stream function to go with the input temperature, density, and tangential velocity. See appendix B. Subroutine NEWRHO also calculates values of ξ and ζ (eqs. (A1) to (A3)) at the mesh points to be used in COEF on the next iteration. And NEWRHO checks the relative change in velocity from the previous iteration at each mesh point. The maximum relative change in velocity is checked to see if the solution is converged.

Now that a solution (converged or not) has been obtained, OUTPUT is called. Subroutine OUTPUT first calculates other velocity components and flow angles at all mesh points. Then OUTPUT calculates streamline curvature and critical velocity ratio at each mesh point. Subroutine BLDVEL is called to calculate the blade surface velocities, as explained in appendix G of part I (ref. 6). Also BLDVEL calculates the average blade-to-blade density to be used in NEWRHO in the next ieration. And BLDVEL calculates F_r at each point by using equation (A4). The radial vector F_r is used by COEF in calculating the coefficients of the finite-difference equations. After returning from BLDVEL, OUTPUT will print out data at the orthogonal mesh points, if desired. Then, if output is desired along streamlines, the necessary interpolation will be done and data will be printed for all streamlines. Similarly, interpolation will be done and data printed for hub-tip station lines.

After OUTPUT, INDEV is called. Subroutine INDEV calculates a correction to $\partial\theta/\partial s$ for a short distance into the blade to match the mean surface within the blade to the free-stream flow angles, both upstream and downstream. The method for doing this is described in appendix F of part I (ref. 6). Subroutine INDEV also calculates and prints out incidence and deviation angles if this is requested. If desired, SLPLOT will plot the streamlines and SVPLOT will plot the mean and blade surface velocities.

At this point, the main program will start a new iteration by going back to COEF if the solution has not converged. If the solution has converged, there are two possibilities. If REDFAC is 1, the final solution has been obtained and the program is through. If there are data for another case, the program will start this case; otherwise the program is stopped. If REDFAC is less than 1, the final approximate full-mass-flow solution will be calculated by TVELCY. First, the mass flow, rotational speed, and inlet and output whirl are restored to their full values. This requires reinitialization calls of LAMDAF and RVTHTA for inlet and outlet whirl. Then TVELCY calculates $\partial W_m/\partial m$ and $\partial W_\theta/\partial m$ for use in the velocity-gradient equation. These quantities are first calculated from the reduced-mass-flow solution and then are adjusted by dividing by REDFAC. Now the velocity-gradient equation (derived in appendix C of part I (ref. 6)) is solved along each vertical mesh line. Iteration is required to establish the correct temperature, density, and whirl to use in the velocity-gradient equation. When TVELCY is through, TOUTPT is called. Subroutine TOUTPT is an alternate entry point for OUTPUT. The only difference is that the flow angles are considered to be known, and the velocity components are calculated from the velocity magnitude and the known flow angles. Then the same sequence of INDEV, SLPLOT, and SVPLOT is called as for the finite-difference solution. Normally, only the smaller (subsonic) of two possible solutions is obtained by TVELCY (part I, appendix C); but if desired, both the larger ("supersonic") and smaller solutions can be obtained. If both solutions are desired, TVELCY, TOUTPT, INDEV, SLPLOT, and SVPLOT are called again. This completes the program. If there are data for another case, the program will start on this case: otherwise the program is stopped.

DETAILED PROGRAM PROCEDURE

This main section gives the detailed program procedure for all the subroutines. The previous main section should be consulted for an overall view of the program calculation procedure.

Most of the subroutines in MERIDL use the same set of variables. These variables are all defined in the MAIN DICTIONARY. All subroutines are described prior to the main dictionary. First, the main subroutines and other subroutines which use the main dictionary are described, and then the remaining subroutines with special dictionaries are described.

The calling relation of all subroutines is shown in figure 3. Note that figure 3 is not a flow chart. A tabulation of all subroutines called and all COMMON blocks for each subroutine is given in table I.

The first subsections presented here describe the general aspects of the programs, including storage requirements, conventions used, and description of labeled COMMON blocks. They are followed by a detailed description of the subroutines.

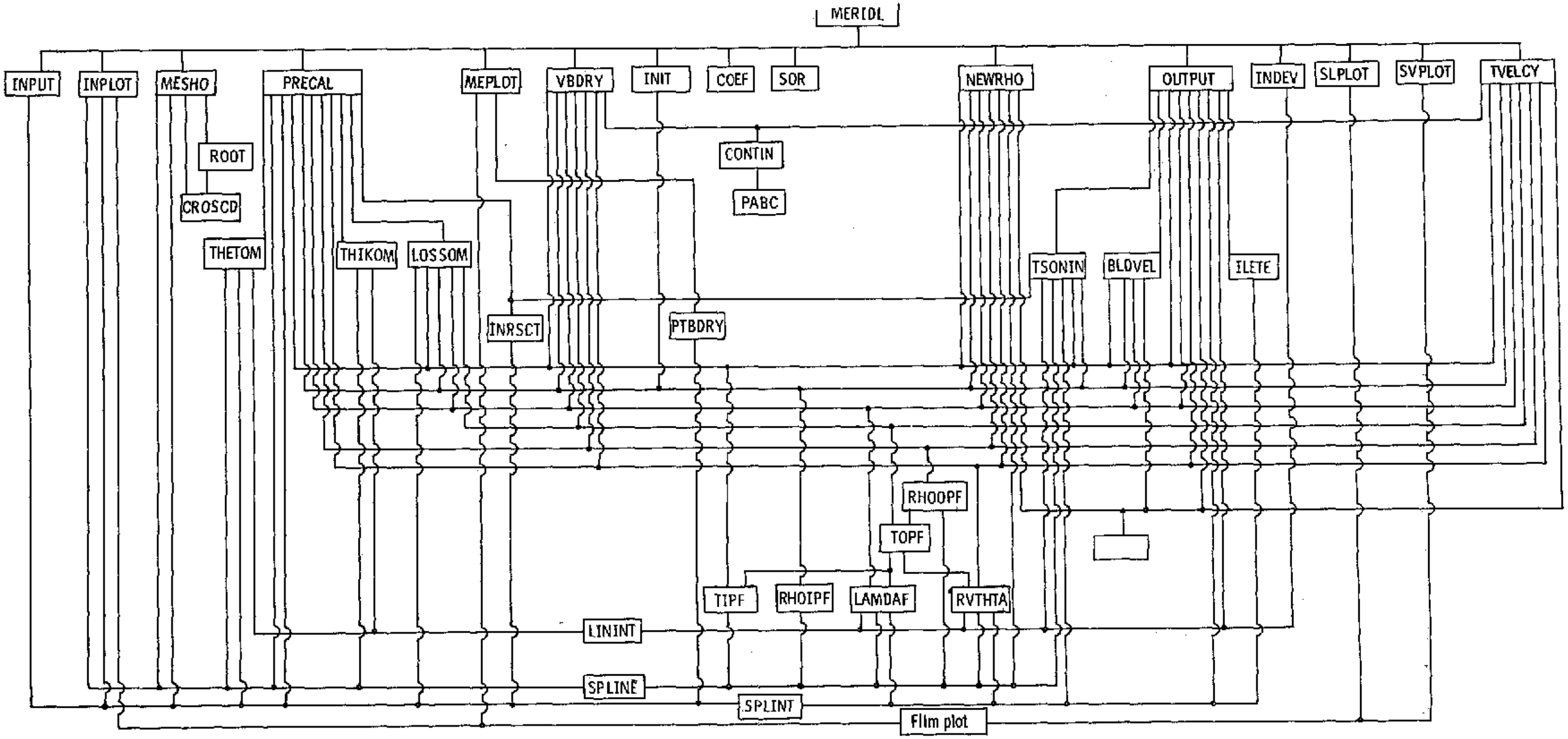


Figure 3. - *Calling relation of subroutines.* Called subroutines are always below the calling subroutine. (This is not a flow chart.)

TABLE I. - SUBROUTINE CALLS AND COMMON BLOCKS

Subroutine name or entry point	COMMON blocks (a)	Called subroutines	Calling subroutines
MERIDL	/---/ /INPUTT/	INPUT INPLOT MESHO PRECAL MEPLOT VBDRY INIT COEF SOR NEWRHO OUTPUT INDEV SLPLOT SVPLOT TVELCY TOUTPT (OUTPUT)	None - main program
INPUT	/---/ /INPUTT/ /CALCON/ /INTITL/	SPLINT	MERIDL
INPLOT	/INPUTT/ /CALCON/ /INTITL/	Microfilm plot subroutines SPLINT SPLINE	MERIDL
MESHO	/INPUTT/ /CALCON/ /CROSCM/	CROSCD SPLINT SPLINE ROOT	MERIDL
CROSCD	/INPUTT/ /CROSCM/	None	MESHO ROOT
PRECAL	/INPUTT/ /CALCON/	LAMNIT (LAMDAF) RVTNIT (RVTHTA) TIPNIT (TIPF) RHINIT (RHOIPF) RHONIT (RHOOPF) SPLINE INRSCT SPLINT THETOM THIKOM LOSSOM	MERIDL
THETOM	/INPUTT/ /CALCON/ /INDCOM/	SPLINT SPLINE LININT	PRECAL
THIKOM	/INPUTT/ /CALCON/	SPLINE LININT	PRECAL
LOSSOM	/---/ /INPUTT/ /CALCON/	TIPF LAMDAF TOPF RHOIPF SPLINT	PRECAL
MEPLOT	/INPUTT/ /CALCON/ /PLTCOM/	PTBDRY Microfilm plot subroutines	MERIDL
PTBDRY	/INPUTT/ /CALCON/ /PLTCOM/	SPLINT	MEPLOT SLPLOT
VBDRY	/---/ /INPUTT/ /CALCON/ /VARCOM/	TIPF RHOIPF LAMDAF TOPF RHOOPF RVTHTA CONTIN	MERIDL
CONTIN	None	PABC	VBDRY TVELCY
PABC	None	None	CONTIN
INIT	/INPUTT/ /CALCON/ /VARCOM/	RHOIPF	MERIDL
COEF	/---/ /INPUTT/ /CALCON/ /VARCOM/	None	MERIDL
SOR	/---/ /INPUTT/ /CALCON/ /VARCOM/	None	MERIDL
NEWRHO	/---/ /INPUTT/ /CALCON/ /VARCOM/	LAMNIT (LAMDAF) RVTNIT (RVTHTA) TIPNIT (TIPF) RHINIT (RHOPIF) RHONIT (RHOOPF) LAMDAF TIPF RHOIPF SLOPES SPLINE RVTHTA	MERIDL
OUTPUT (entry point - TOUTPT)	/---/ /INPUTT/ /CALCON/ /VARCOM/ /SLCOM/ /STACOM/	SLOPES TIPF LAMDAF BLDVEL RVTHTA SPLINT LININT ILETE TSONIN	MERIDL
TSONIN	/---/ /INPUTT/ /CALCON/ /VARCOM/ /SLCOM/	SPLINE TIPF RHOIPF INRSCT LININT SPLINT	OUTPUT TOUTPT

[a] --- denotes an unlabeled COMMON block.

TABLE I. - Concluded. SUBROUTINE CALLS AND COMMON BLOCKS

Subroutine name or entry point	COMMON blocks (a)	Called subroutines	Calling subroutines
BLDVEL	/---/ /INPUTT/ /CALCON/ /VARCOM/	SLOPES LAMDAF TIPF RHOIPF	OUTPUT TOUTPT
ILETE	/INPUTT/ /CALCON	SPLINT SPLENT (SPLINT)	OUTPUT TOUTPT
INDEV	/---/ /INPUTT/ /CALCON/ /VARCOM/ /INDCOM/	LININT	MERIDL
SLPLOT	/---/ /INPUTT/ /SLCOM/ /PLTCOM/	Microfilm plot subroutines PTBDRY	MERIDL
SVPLOT	/---/ /INPUTT/ /SLCOM/	Microfilm plot subroutines	MERIDL
TVELCY	/---/ /INPUTT/ /CALCON/ /VARCOM/	LAMNIT (LAMDAF) RVTNIT (RVTHTA) SLOPES LAMDAF RVTHTA TIPF TOPF RHOIPF RHOOPF CONTIN	MERIDL
TIPF (see entry TIPNIT)	/INPUTT/ /CALCON/	None	LOSSOM VBDRY NEWRHO OUTPUT TOUTPT TSONIN BLDVEL TVELCY TOPF
TIPNIT (entry point for TIPF)	/INPUTT/ /CALCON/	SPLINE	PRECAL NEWRHO
RHOIPF (see entry RHINIT)	/INPUTT/ /CALCON/	None	LOSSOM VBDRY INIT NEWRHO TSONIN BLDVEL TVELCY
RHINIT (entry point for RHOIPF)	/INPUTT/ /CALCON/	SPLINE	PRECAL NEWRHO
LAMDAF (see entry LAMNIT)	/---/ /INPUTT/ /CALCON/ /VARCOM/	None	LOSSOM VBDRY NEWRHO OUTPUT TOUTPT BLDVEL TVELCY TOPF
LAMNIT (entry point for LAMDAF)	/---/ /INPUTT/ /CALCON/ /VARCOM/	LININT SPLINT SPLINE	PRECAL NEWRHO TVELCY
TOPF	INPUTT CALCON	TIPF LAMDAF RVTHTA	LOSSOM VBDRY TVELCY
RHOOPF (see entry RHONIT)	/INPUTT/ /CALCON	None	VBDRY TVELCY
RHONIT (entry point for RHOOPF)	/INPUTT/ /CALCON/	SPLINE TOPF	PRECAL NEWRHO
RVTHTA (see entry RVTNIT)	/---/ /INPUTT/ /CALCON/ /VARCOM/	None	VBDRY NEWRHO OUTPUT TOUTPT TVELCY TOPF
RVTNIT (entry point for RVTHTA)	/---/ /INPUTT/ /CALCON/ /VARCOM/	LININT SPLINT SPLINE	PRECAL NEWRHO TVELCY
INRSCT	/---/	SPLINT	PRECAL TSONIN
ROOT	/---/	CROSCD	MESHO
LININT	None	None	THETOM THIKOM OUTPUT TOUTPUT TSONIN INDEV LAMNIT RVTNIT
SPLINE	/---/	None	INPLOT MESHO PRECAL THETOM THIKOM NEWRHO TSONIN TIPNIT RHINIT LAMNIT RHONIT RVTNIT
SPLINT (see entry SPLENT)	/---/	None	INPUT INPLOT MESHO PRECAL THETOM LOSSOM PTBDRY OUTPUT TOUTPUT TSONIN ILETE LAMNIT RVTNIT INRSCT
SPLENT (entry point for SPLINT)	/---/	None	ILETE
SLOPES	None	None	NEWRHO OUTPUT TOUTPUT BLDVEL TVELCY
Microfilm plot subroutines	None	Not applicable	INPLOT MEPLOT SLPLOT SVPLOT

[a] --- denotes an unlabeled COMMON block.

STORAGE REQUIREMENTS

The MERIDL program has been implemented on the NASA Lewis time-sharing IBM-TSS/360-67 computer. Storage for the program code is approximately 18 000 words. For the numerical example of part I (ref. 6), storage of variables required approximately 60 000 words for a 21 × 41 grid of 861 points. As dimensioned for a 100 × 101 grid, storage of variables would require about 680 000 words. The user can reduce the storage requirements for variables, as desired, by changing the dimensions. The main dictionary indicates how each variable should be dimensioned to reduce the storage required. This is indicated by reference to certain input variables, such as MM, MHT, NHUB, NTIP, NBLPL, NPPP, and so forth. The variables with the most significant effect on storage requirements are MM and MHT.

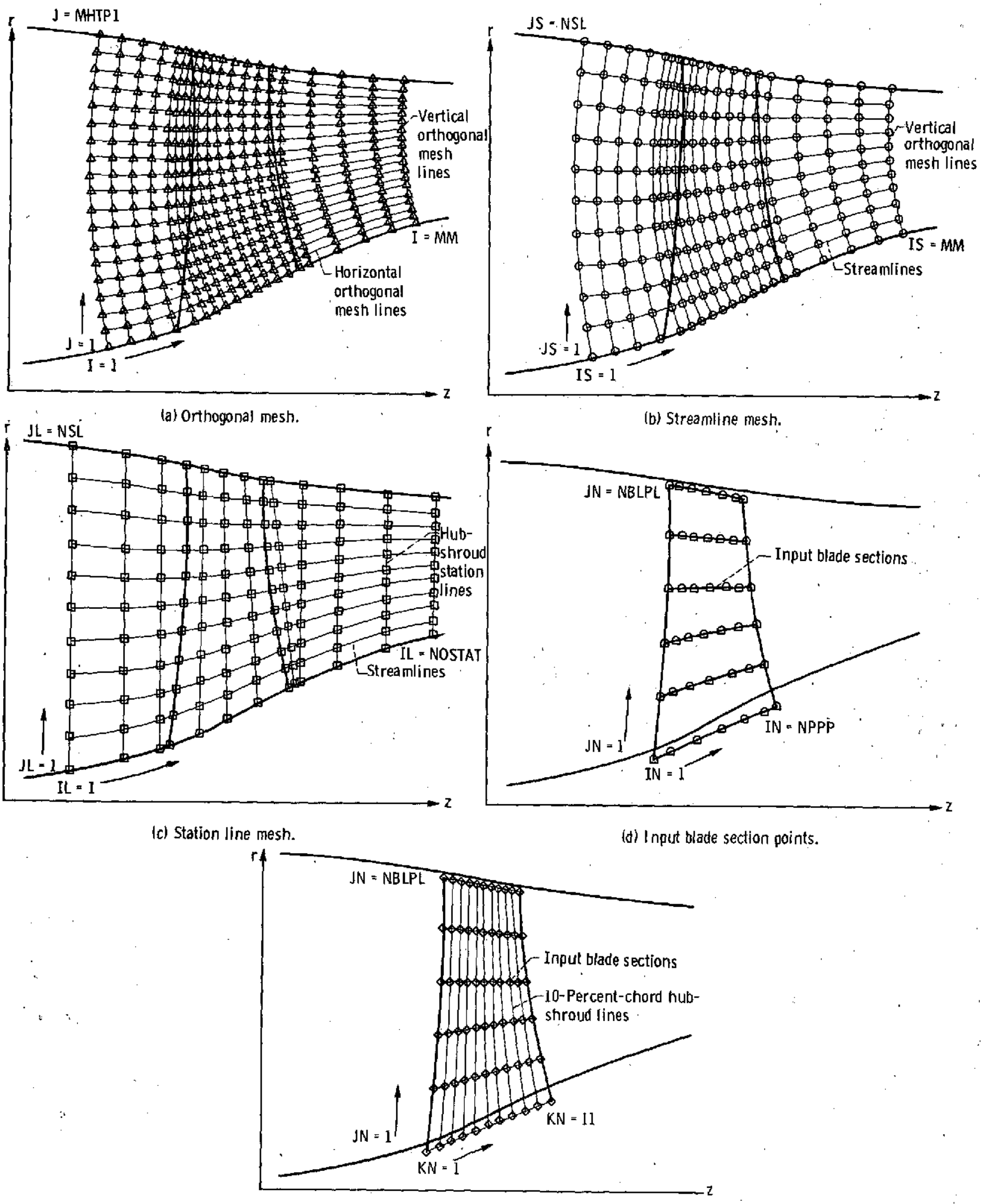
As an example, consider the two-dimensional array ALPHA. This variable is in the /VARCOM/ COMMON block and is dimensioned ALPHA (100,101) in the program listing. In the main dictionary, it is listed as ALPHA (MM, MHTP1). Suppose that the maximum desired value for MM is 60 and that for MHT it is 40. Since MHTP1 is MHT + 1, the maximum value for MHTP1 would be 41. Then ALPHA should be dimensioned ALPHA (60,41).

Similarly, all other dimensioned variables should have their dimension changed as required. Most dimensioned variables are in COMMON blocks, but there are a few which are dimensioned locally only. In addition, the calls to LININT must be changed to reflect any changes in the dimensions of the first two LININT arguments.

CONVENTIONS USED IN PROGRAM

For convenience, a number of conventions are used in naming variables and assigning subscripts.

In addition to the basic orthogonal mesh, there are four special mesh schemes used, as illustrated in figure 4. For each mesh, different conventions are used to indicate mesh position. The subscripts I and J are used to denote orthogonal mesh position. The I is used to denote the vertical mesh number, and the J is used to denote the horizontal mesh line number. The subscripts IS and JS are used in a similar manner to denote streamline mesh points, and IL and JL the station-line mesh points. Likewise, IN and JN denote points on the input blade sections, and KN and JN denote points on the alter-



(a) Orthogonal mesh.

(b) Streamline mesh.

(c) Station line mesh.

(d) Input blade section points.

(e) Alternate blade mesh.

Figure 4. - Five meshs used in MERIDL.

nate blade mesh located at 10-percent-chord intervals in the THETOM subroutine. Note that I and IS take on the same values, as do JS and JL.

In variable names, I or IN indicates the inlet (upstream of blade) and O or OUT indicates the outlet. Variables ending with OM are generally variables defined on the orthogonal mesh.

Velocity components on the orthogonal mesh usually have SUB in the name, such as WSUBZ for W_z. Velocity components along streamlines end in SL (WZSL), while velocity components on station lines end in ST (WZST). The letters H or HUB in a variable name indicate the hub, and T or TIP the tip. LE is used for leading edge and TE for trailing edge. The letters TH indicate a variable in the θ-direction, SURF a variable on a blade surface, and BL a variable in the blade region. In a variable name, TEM indicates a temporary variable; P is used to indicate a prime superscript, and PP double prime; D is used for derivative. Usually, several conventions are combined in each variable. For example, TIP is used for T_i', TPPTIP for T''/T_i', and DPDR is used for $\partial p/\partial r$.

All subroutines used for plotting have PLOT in the name. Variables used for plotting have PLT in the name.

LABELED COMMON BLOCKS

Most variables which are used in more than one subroutine are placed in labeled COMMON blocks. A brief description of each labeled block is given. The same variable names are used in different subroutines for every variable in a COMMON block. The labeled COMMON blocks are as follows:

- /INPUTT/ is used for all input quantities.
- /CALCON/ is used for calculated constants which are initially calculated and are not changed later.
- /VARCOM/ is used for all orthogonal mesh point arrays which are changed in each iteration.
- /SLCOM/ is used for output data along streamlines.
- /STACOM/ is used for output data along station lines.
- /INDCOM/ is used for quantities calculated by THETOM to be used by INDEV.
- /PLTCOM/ is used to plot data for hub, shroud, and blade leading and trailing edges.
- /CROSCM/ is used to store quantities required by CROSCD.
- /INTITL/ is used to store the input title for use by INPLOT.

Table I shows which COMMON blocks are needed in each subroutine.

MAIN PROGRAM

The program is segmented into several main subroutines called by the main pro-ram, as indicated in figure 3. The subroutines are called in sequence, except for the uter iteration and a switch to obtain a supersonic final solution. The outer iteration is loop consisting of calls to COEF, SOR, NEWRHO, OUTPUT, INDEV, SLPLOT, and VPLOT. This calling sequence and the outer iteration loop are shown more clearly in he flow chart for the main program, given in figure 2. Flow charts for some of the ubroutines are also given with the subroutine descriptions.

SUBROUTINES

Subroutine INPUT

Subroutine INPUT reads and prints all input data cards and initializes some varia-les for use later in the program.

All input cards are first read and printed on the output listing in the same form and rder in which they are given. All array bounds are then checked to see if they are ithin limits, and some miscellaneous constants are initialized. Finally, estimates are nade of various required upstream and downstream flow conditions which were not given s input because other input options were used.

Subroutine INPLOT

Subroutine INPLOT makes microfilm plots of a portion of the given input data. By hecking these plots, the user can see if his input data have been given correctly and moothly. It is important that the plotted data be smooth, since spline curve fits of iese data are used extensively by the program.

Two main sections of input are plotted: the upstream and downstream flow condi-ions, and the input blade sections from hub to shroud. A separate plot is made of each f the three given distributions of upstream flow variables and two distributions of down-tream flow variables from hub to shroud. On each plot, one data point is plotted at very 1 percent of stream function or radius from hub to shroud. The NIN and NOUT nput points are also marked. Each input blade section is then plotted, using only the in-ut points for plotting and marking. After each individual blade section is plotted, a mul-ple plot is made of all sections together. Examples of all the plots are given in figures (a) to (g) of part I (ref. 6).

Subroutine INPLOT and the other plot routines, MEPLOT, SLPLOT, and SVPLOT, all rely heavily on the NASA Lewis in-house microfilm plotting package described in reference 8. These four routines as well as PTBDRY, which is called by MEPLOT and SLPLOT, are self-contained and can be easily removed from MERIDL without disturbing the remainder of the calculations.

Subroutine MESHO

Subroutine MESHO calculates the coordinates of an orthogonal mesh covering the solution region from upstream to downstream of the blade row and from hub to shroud. Subroutine MESHO makes use of four other subroutines - ROOT, CROSCD, SPLINE, and SPLINT. A flow chart for MESHO is given in figure 5. The method used for generating the mesh is explained thoroughly in reference 7.

Subroutine MESHO begins with input geometry describing the hub and shroud of the flow passage and the numbers of mesh points desired in the horizontal and vertical directions. First, MESHO calculates the horizontal, or streamwise, orthogonals. It does this by extending lines vertically from each of the input points on the hub to the shroud. Each of these lines is then divided into equal increments, the number depending upon the number of streamwise orthogonals. Streamwise spline curves are fit through the resulting points to give the horizontal orthogonals shown in figure 6.

Vertical orthogonal lines are then constructed one at a time, moving from left to right between each pair of adjacent horizontal orthogonals, proceeding from hub to shroud, as shown in figure 7. The procedure for calculating these lines, shown in figure 8, is analogous to a technique for solving ordinary differential equations known as the improved Euler method or Heun's method (ref. 9). Beginning at a known orthogonal mesh point on the lower orthogonal, a normal is constructed (line ① in fig. 8) to the upper orthogonal. Then the intersection coordinates of this line with the upper orthogonal and the slope of the upper orthogonal at the intersection are calculated. ROOT and CROSCD are used in this process. Line ② in figure 8 is then constructed in such a way that it is perpendicular to the tangent to the upper orthogonal at the intersection point and passes through the original starting point on the lower orthogonal. The coordinates of the intersections of both lines ① and ② are now known on the upper orthogonal. The desired new orthogonal mesh point is the average of these two sets of coordinates.

This process of constructing vertical orthogonal links is continued until the shroud is reached by all vertical orthogonals. This completes the generation of the orthogonal mesh.

Notice in MESHO that the locations of the upstream and downstream boundaries of the orthogonal mesh at the hub are fixed by the inputs ZOMIN and ZOMOUT (fig. 7). The

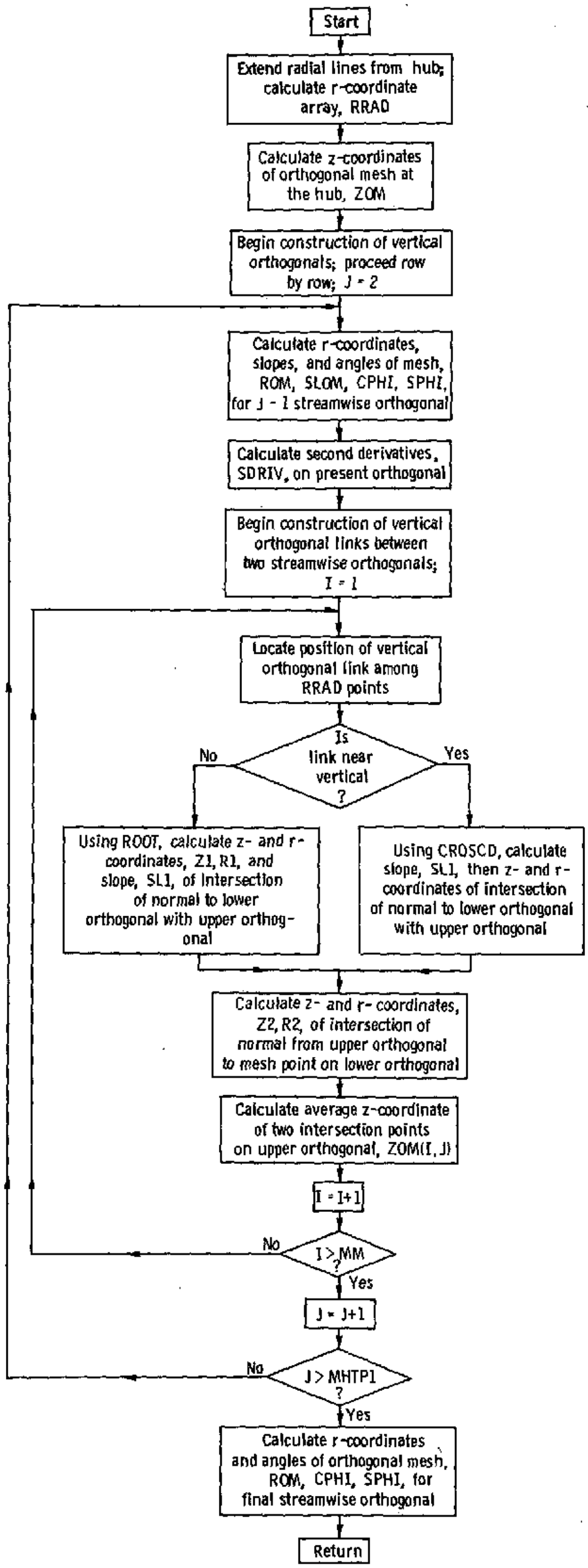


Figure 5. - Flow chart for MESHO.

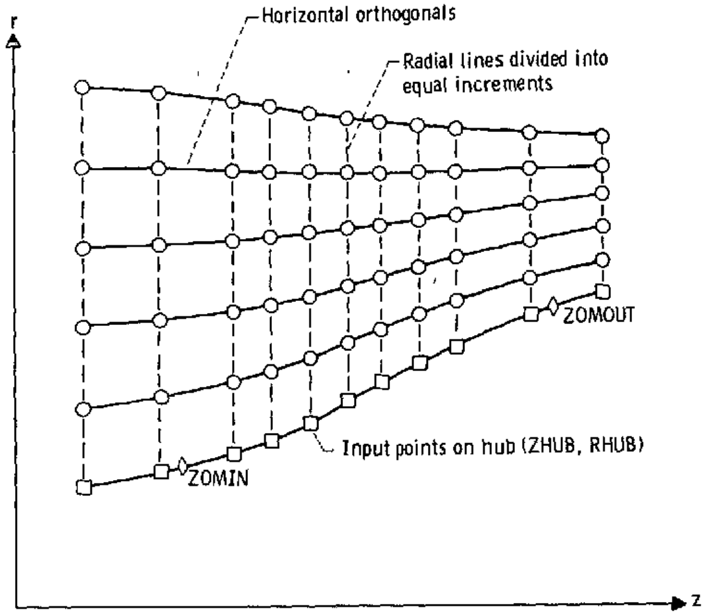


Figure 6. - "Horizontal" orthogonals obtained by spline curve fitting.

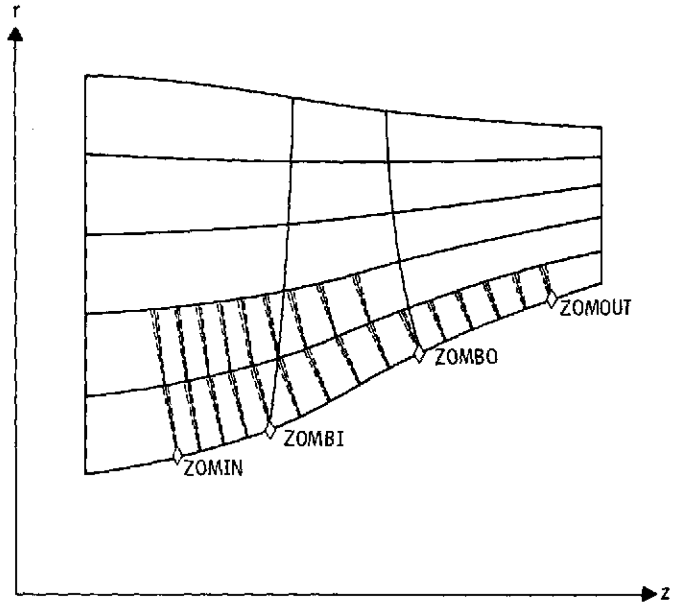


Figure 7. - Process for generating "vertical" orthogonal links.

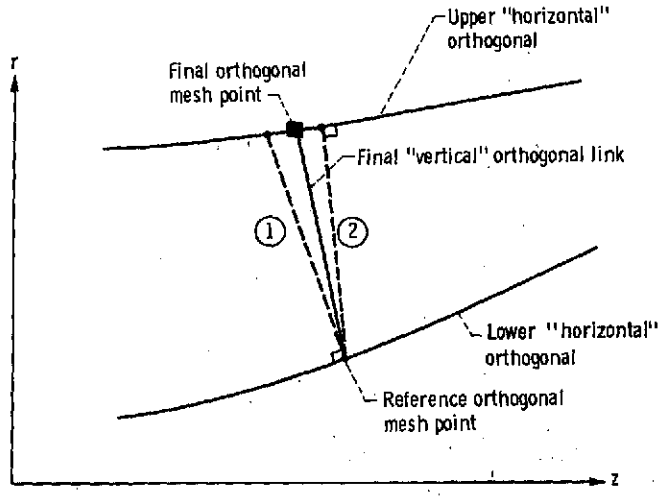


Figure 8. - Calculation procedure for a "vertical" orthogonal link.

locations of these boundaries at the tip, however, cannot be given ahead of time and are totally dependent upon the orthogonal mesh generation procedure.

Axial distance between vertical orthogonal origins at the hub is determined by the number of mesh lines requested in the following three regions: MBI mesh lines upstream of the blade from ZOMIN to ZOMBI; MBO - MBI mesh lines from ZOMBI to ZOMBO; and MM - MBO mesh lines downstream of the blade from ZOMBO to ZOMOUT (fig. 7). The number of horizontal orthogonals is MHT + 1, which is the same in all three regions.

Subroutine CROSCD

The ROOT subroutine (p. 52) requires the calling of a special function or subroutine. In MERIDL, that routine is CROSCD. It is called by ROOT to calculate for a given z-coordinate the difference in r-coordinates of line ① and the upper horizontal orthogonal in figure 8. (ROOT finds the z-coordinate where this difference shrinks to zero, that is, the intersection of the straight line ① and the horizontal orthogonal curve.) The input argument for CROSCD is

Z value of z-coordinate in ROOT

The following two values are given as output:

RMR difference in r-coordinates of straight line and curve

SL1 slope of horizontal orthogonal at z

CROSCD uses the equations of a cubic spline curve to interpolate and calculate r as a function of z on the horizontal orthogonal. The spline curve information is transmitted to CROSCD from MESHO in the /CROSCM/ COMMON.

Subroutine PRECAL

Subroutine PRECAL calculates many of the fixed constants which will be needed by the subroutines in the outer iterative loop of MERIDL. Figure 9 gives a flow chart for PRECAL.

First, PRECAL initializes the subroutines for calculating upstream and downstream flow conditions. To do this it calls LAMDAF, RVTHTA, TIPF, RHOIPF, and RHOOPF, entering at the special entry points of these routines used for initialization.

The array of blade-to-blade spacing B (the BTH array) is then initialized to the blade pitch (in radians) at every point on the solution mesh. This array is modified in the blade region later in PRECAL when THIKOM and LOSSOM are called.

In the cases where output streamline values (FLFR array) were not read in (NSL = 0), PRECAL assigns eleven (11) values to FLFR from 0 to 1.0, in increments of 0.1. Also, if the given endpoints of FLFR do not equal 0 and 1.0, PRECAL adds these values as endpoints.

Then, PRECAL uses the z- and r-coordinates of the orthogonal mesh (ZOM and ROM), calculated in MESHO, to calculate the s and t arrays (SOM and TOM) on the orthogonal mesh. Adjacent points are linked with straight line segments in this calculation of s and t, but the correction between arc length and chord length is not significant for adjacent points.

The curvatures of the hub and shroud profiles are then calculated where these profiles are intersected by the upstream and downstream boundaries of the orthogonal mesh. These curvatures are later required in the VBDRY subroutine.

The z- and r-coordinate arrays (ZLE,RLE and ZTE,RTE) are then set up at points which define the leading and trailing edges of the blade. These values are the first and last values for each blade plane from the ZBL and RBL blade-coordinate input arrays. The intersections of these leading and trailing edges with the hub and shroud are also calculated with INRSCT calls.

Various quantities are then calculated on the orthogonal mesh at or near the leading and trailing edges of the blade. With INRSCT calls, the z- and r-coordinates of intersections of horizontal mesh lines with the blade edges are calculated. Vertical mesh line numbers (ILE and ITE) of mesh points which lie just within the blade leading and trailing edges are then calculated by comparing the z-coordinates of mesh points along the orthogonals with the z-coordinates of intersections of the orthogonals with the blade edges. The s- and t-coordinates are then calculated for the points where the horizontal mesh lines cross the blade edges.

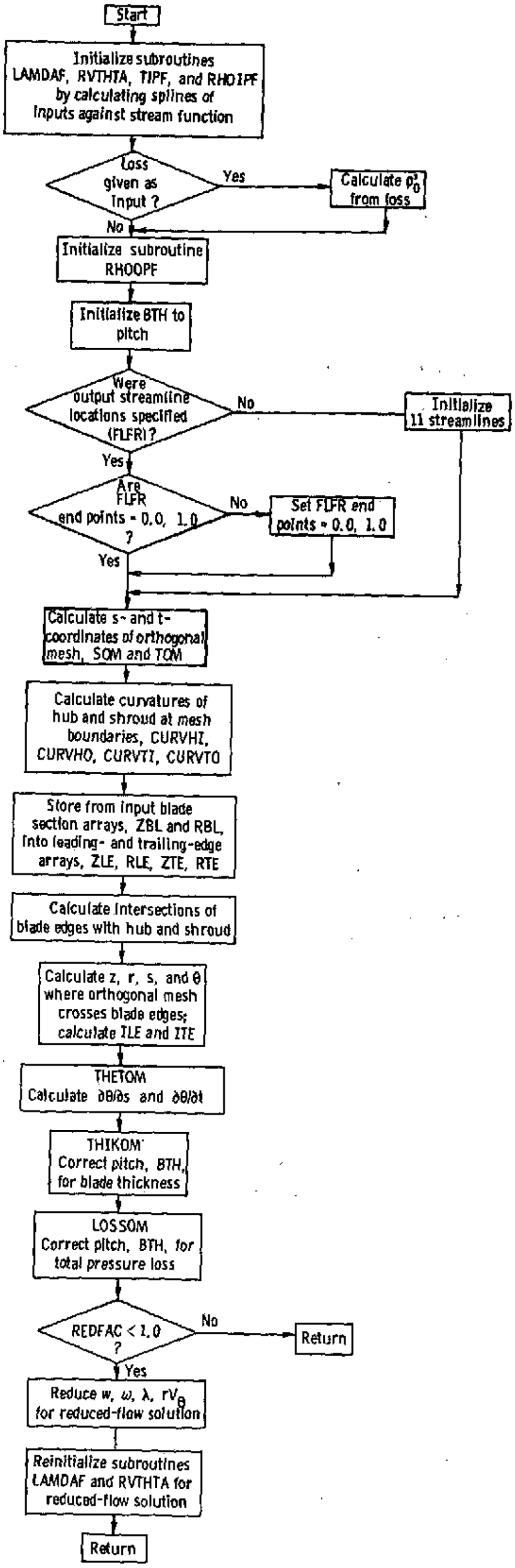


Figure 9. - Flow chart for PRECAL.

Then PRECAL calls three other subroutines, THETOM, THIKOM, and LOSSOM. The THETOM routine calculates $\partial\theta/\partial s$ and $\partial\theta/\partial t$ at the orthogonal mesh points. Subroutine THIKOM makes corrections to the BTH array to account for blade thickness, and LOSSOM calculates the relative total pressure loss at the downstream boundary of the orthogonal mesh. This loss is distributed linearly through the blade row by making an additional correction to the BTH array.

Finally, in PRECAL, corrections are made to some upstream and downstream input arrays and corresponding boundary conditions in the case where a reduced-mass-flow solution is to be obtained (REDFAC < 1.0). The wheel speed, mass flow, whirl, and tangential velocity are all reduced by REDFAC; and the upstream and downstream boundary conditions of whirl are reinitialized by LAMDAF and RVTHTA calls.

Subroutine THETOM

Subroutine THETOM calculates the gradients $\partial\theta/\partial s$ and $\partial\theta/\partial t$ at the orthogonal mesh points which lie within the leading and trailing edges of the blade. This process is thoroughly described in appendix C.

Theta coordinates of the mean blade surface (THBL) are given at the input blade section points (ZBL,RBL). Gradients of the θ-coordinate are required in the s- and t-directions at the orthogonal mesh points within the blade for use by the NEWRHO subroutine.

Subroutine THETOM makes use of the technique of defining an alternate mesh which is entirely contained within the blade on which $\partial\theta/\partial z$ and $\partial\theta/\partial r$ are obtained. By interpolation, $\partial\theta/\partial z$ and $\partial\theta/\partial r$ are then obtained at the required orthogonal mesh points. Finally, $\partial\theta/\partial s$ and $\partial\theta/\partial t$ are calculated from $\partial\theta/\partial z$ and $\partial\theta/\partial r$ at these points.

Subroutine THIKOM

Subroutine THIKOM calculates the blade thickness in the θ-direction at the points of the orthogonal mesh which lie within the blade edges. (Input blade thicknesses are not given at the orthogonal mesh points, nor are they given in the θ-direction. They are given normal to the blade mean camber line along each input blade section.)

THIKOM first calculates the s'-coordinate and then the angle κ between the mean camber line and the s'-coordinate direction, as shown in figure 10. (The s'-coordinate corresponds to the input blade section direction.) With these angles, approximate thicknesses in the tangential direction t_θ are calculated from thicknesses normal to the meanline t_n by the equation

$$t_\theta = \frac{t_n}{\cos \kappa}$$

This calculation is subject to error for highly cambered or highly staggered blade sections but is adequate here since t_θ is only used as a blockage correction to the BTH array.

After t_θ is obtained at the input points, LININT is called to interpolate and obtain it at the orthogonal mesh points. Then it is subtracted from the BTH array.

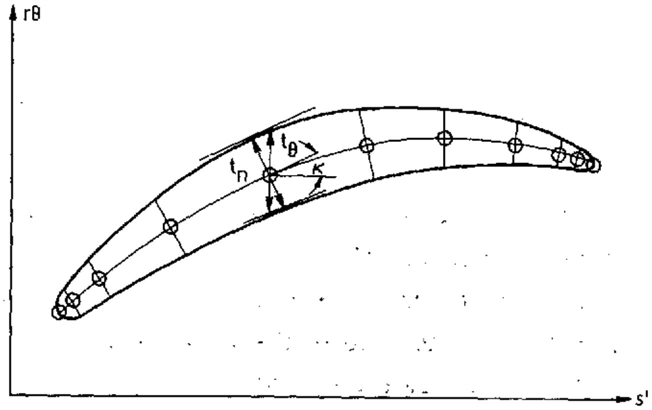


Figure 10. - Calculation of thickness in tangential direction.

Subroutine LOSSOM

Subroutine LOSSOM calculates the downstream relative total pressure loss and distributes it upstream through the blade row as an area correction. This correction is made to the BTH array. The loss is calculated as 1 minus the ratio of actual to ideal relative total pressure along the hub-shroud input line downstream of the blade.

$$\text{Loss} = 1 - \frac{p_o''}{\left(p_o''\right)_{ideal}}$$

In one input option, p_i', T_i', and p_o' are given and T_o' is then calculated from Euler's equation. Then using the relations

$$\frac{p_o''}{p_o'} = \left(\frac{T_o''}{T_o'}\right)^{\gamma/(\gamma-1)}$$

and

$$\frac{\left(p_o''\right)_{ideal}}{p_i'} = \left(\frac{T_o''}{T_i'}\right)^{\gamma/(\gamma-1)}$$

we form the ratio of actual to ideal relative total pressure

$$\frac{p_o''}{\left(p_o''\right)_{ideal}} = \frac{p_o'}{p_i'}\left(\frac{T_i'}{T_o'}\right)^{\gamma/(\gamma-1)}$$

There is an alternate input option where loss is given as input. In that case, relative total pressure ratio is calculated as

$$\frac{p_o''}{\left(p_o''\right)_{ideal}} = 1 - \text{Loss}$$

Subroutine LOSSOM then estimates the stream function, based on area, at points on the downstream boundary of the orthogonal mesh. Subroutine SPLINT is then called to interpolate the values of relative total pressure ratio at these same downstream boundary mesh points. From the interpolated values of pressure ratio, loss is computed by using the preceding equations. It is assumed that the horizontal mesh lines are close approximations to the actual streamlines. Thus, the loss is distributed along horizontal mesh lines. Along any mesh line, loss is assumed to be constant in the region downstream of the blade trailing edge and equal to the downstream boundary value. Between the blade leading and trailing edges, loss is distributed linearly from zero at the leading edge to full value at the trailing edge. It is assumed that no loss occurs upstream of the blade. Loss is included in the stream-function solution by reducing the value of B as follows:

$$B_{net} = B(1 - \text{Loss})$$

where B_{net} is the final value stored in the BTH array.

Subroutine MEPLOT

Subroutine MEPLOT makes two microfilm plots of the blade in the meridional plane. The first plot shows the input hub and shroud geometry and the blade leading and trailing edges. The edges are obtained from the ZBL and RBL input arrays. The second plot shows the same hub, shroud, and blade but with the generated orthogonal mesh superimposed on the solution region. Examples of these plots are given in figures 16(h) and (i) of part I (ref. 6).

Prior to making these two plots, MEPLOT calls PTBDRY to obtain the boundary points to be plotted on the hub, the shroud, and the blade edges and to scale the plots.

Subroutine PTBDRY

Subroutine PTBDRY obtains coordinates used in plotting the hub, the shroud, and the blade edges by the two routines MEPLOT and SLPLOT. Using SPLINT calls, PTBDRY interpolates on the input arrays ZHUB,RHUB and ZTIP,RTIP to obtain 100 plotting points on both hub and shroud. The same is done with the blade leading and trailing edges. After the plot points are obtained, PTBDRY searches the range of values to be plotted for the maximum in both X- and Y-directions and adjusts the range of plotted points so that it is the same in both directions. The computed information is stored in the /PLTCOM/ COMMON.

Subroutine VBDRY

Subroutine VBDRY calculates the value of the stream function along the upstream and downstream boundaries of the mesh region. The calculation is based on the velocity-gradient equation (C9) of part I (ref. 6).

There are four arguments for VBDRY so that the same coding can be used for either the upstream or downstream boundary. These arguments are LOC, TIPF, RHOIPF, and LAMDAF. The argument LOC is the value of I for the desired boundary, that is, LOC = 1 on the upstream boundary and LOC = MM on the downstream boundary. The other three arguments are function subroutines. That is, TIPF, RHOIPF, and LAMDAF will refer to the subroutines of the same name at the upstream boundary but will refer to TOPF, RHOOPF, and RVTHTA, respectively, at the downstream boundary. Figure 11 is a flow chart for VBDRY.

The first step is to set CURVH and CURVT to the value of the meridional curvature at the hub and tip. For this, values previously calculated in PRECAL are used. To

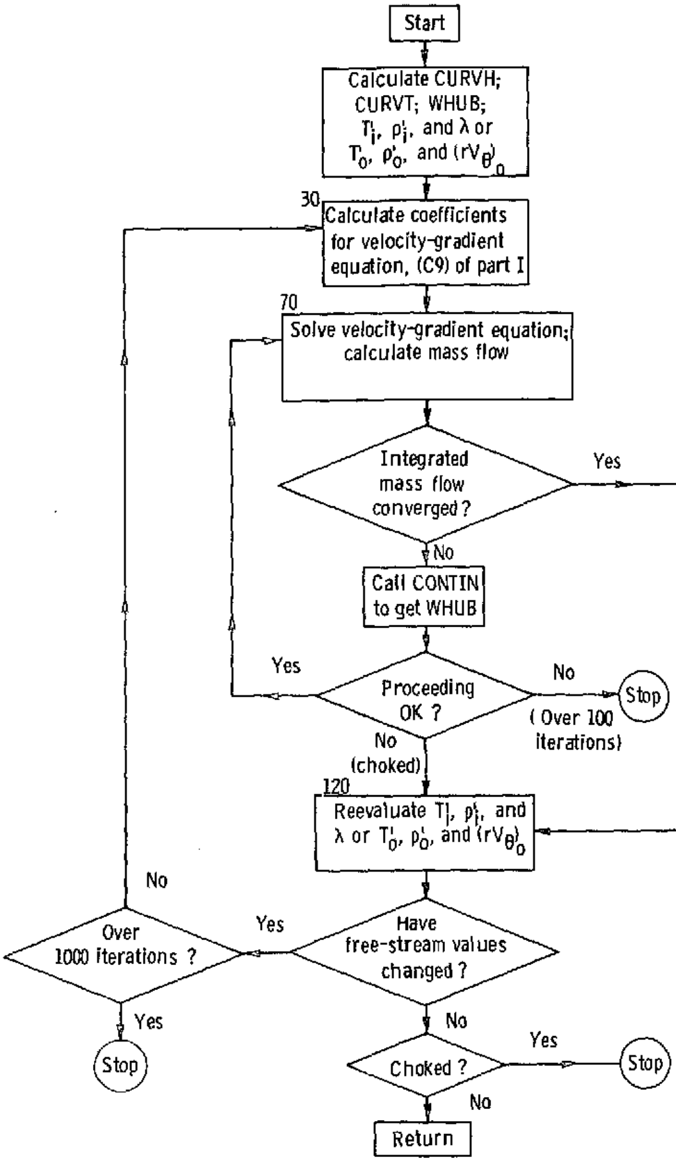


Figure 11. - Flow chart for VBDRY.

start the iterative procedure, a reasonable estimate of the hub relative velocity is needed (WHUB). This estimate is based on a one-dimensional calculation. Before any coefficients for the velocity-gradient equation can be calculated, values for whirl and free-stream absolute stagnation temperature and density are needed. These are all functions of stream function. An initial value of stream function is estimated based on area distribution. Values for whirl and free-stream absolute stagnation temperature and density are then obtained from subroutines LAMDAF, TIPF, and RHOIPF and are stored in arrays.

Now the coefficients of the velocity-gradient equation (C9) of part I (ref. 6) can be calculated. Either equation (C10) or (C11) of part I will be used, whichever is appropriate. The curvature, $1/r_c$, is assumed to vary linearly along the boundary. It is assumed that the boundary is normal to the streamlines so that $\alpha = \varphi$. The quantity $\cos\ \varphi$ has been previously calculated by MESHO.

After the coefficients are calculated, the velocity-gradient equation is solved numerically. WHUB is the initial value of W on the hub. The first iteration will use the value of WHUB calculated previously by VBDRY. Later iterations will use estimated values calculated by CONTIN. Once WHUB is specified, the numerical solution to the velocity-gradient equation is calculated by the Heun method (ref. 9). The equations used in the Heun method for this case are

$$\left.\begin{aligned} W^*_{j+1} &= W_j + (dW)_j && \text{first estimate of } W_{j+1} \\ W^{**}_{j+1} &= W_j + (dW)^*_{j+1} && \text{second estimate of } W_{j+1} \\ W_{j+1} &= \frac{W^*_{j+1} + W^{**}_{j+1}}{2} && \text{average of two estimates of } W_{j+1} \end{aligned}\right\} \qquad (1)$$

where $(dW)_j$ (eq. (C9) of part I) is evaluated at t_j and W_j, and where $(dW)^*_{j+1}$ is evaluated at t_{j+1} and W^*_{j+1}. At the same time the solution of the velocity-gradient equa-equation is being calculated, the mass flow integration is also being calculated by trapezoidal integration:

$$w = \int_0^{t_{tip}} \rho W \cos\beta\ rB\ dt$$

The equations used in calculating the integrand are (B13), (C5), (C6), (C7), and (D4) of part I (ref. 6). At the end of the DO loop at statement 80, the integrated mass flow, UOM (LOC, MHTPI), has been calculated for the specified value of WHUB. This value is checked to see if it is within the tolerance to MSFL. If not, CONTIN is called to provide the next estimate for WHUB. See CONTIN for a description of the procedure for finding the correct value for WHUB. Provision is made to adjust WHUB to avoid problems in calculating either β (if WHUB is too small) or T/T_i' (if WHUB is too large). (See eqs. (C6) or (B13) and (D4) of part I.) After a few iterations, usually four or five and not more than 100, a solution will be found that satisfies continuity. This completes the inner iteration of this subroutine. Then the values of the absolute stagnation temperature T'_i or T'_o, the absolute stagnation density ρ'_i or ρ'_o, and the whirl λ or $\left(rV_\theta\right)_o$ are recalculated from the new stream-function values. If there is a significant change in any of these values, the solution will be repeated (REPEAT = .TRUE.).

When a final acceptable solution is found, the program returns to the main program. If the passage is choked or if an acceptable solution cannot be found in 1000 total iterations, the program will print an error message and stop. See the section Error Messages in part I (ref. 6) for suggestions on what to do when an error message is encountered.

Subroutine INIT

This subroutine initializes certain arrays in /VARCOM/. This is necessary to start the outer iteration running from COEF to SVPLOT. For the initial iteration, it is assumed that $\rho = \rho'_i$ throughout the passage. All other values are set to zero, except for W_s, W_t, and W_z, which are set to values which will avoid division by zero.

Subroutine COEF

Subroutine COEF calculates the coefficients a_1, a_2, a_3, and a_4 and the constants k_o for the finite-difference equations. The finite-difference equation is (A5) or (A7). The coefficients are calculated by the procedure of equation (A8), and the constants are calculated by equation (A9). Within the blade row, the value of the constant k_o depends on $\partial(rV_\theta)/\partial r$. This gradient tends to be unstable with iteration, so that usually damping is required between iterations. The damping rate is controlled by the input variable DNEW. Suggestions for choosing proper values for DNEW are given in the INPUT section of part I (ref. 6). For every outer iteration, the maximum and minimum values of $\partial(rV_\theta)/\partial r$ and the maximum predicted change in $\partial(rV_\theta)/\partial r$ are calculated and printed. When it is indicated by the value of IDEBUG, the coefficients a_i and the constants k_o will be printed.

Subroutine SOR

Subroutine SOR solves the finite-difference equations (A5) by the method of overrelaxation (ref. 10). Equation (A5) holds at every interior point of the orthogonal mesh where the value of u is initially unknown. Thus, if there are n interior points, we have n equations with n unknowns. Equation (A5) is nonlinear but can be linearized by using values from the previous outer iteration for the nonlinear terms or factors.

SOR solves only the linearized equations.

The overrelaxation iteration is the inner iteration; it is optimized by using an optimum overrelaxation factor (ORF). The calculation of ORF is done only the first time that SOR is called. The optimum value for the overrelaxation factor Ω is estimated by using equations (B3) and (B1) of reference 11. At each interior point, u_o^{m+1} is calculated from the values of u at the neighboring points by

$$u_o^{m+1} = \sum_{i=1}^{4} a_i u_i$$

where each u_i is the most recently calculated value for the point. To start, $u_o^0 = 1$ at the interior points and $u_o^0 = 0$ at the boundary points. The maximum (LMAX) and minimum (LMIN) values over all the interior mesh points of the ratio u_o^{m+1}/u_o^m are calculated for $m = 1, 2, 3, \ldots$ until the LMAX and LMIN ratios are close to each other. Then the optimum overrelaxation factor (ORF) is calculated by $ORF = 2\Big/\left(1 + \sqrt{1 - LMAX}\right)$. The theory for calculating ORF is derived in reference 10.

With an optimum value for the overrelaxation factor Ω, the solution to equation (A5) is calculated by overrelaxation by

$$u_o^{m+1} = u_o^m + \Omega\left(\sum_{i=1}^{4} a_i u_i + k_o - u_o^m\right)$$

where each u_i is the most recently calculated value at an interior point or is a boundary value. During each iteration, the maximum change of the stream function is calculated. When this maximum change is reduced below 10^{-5}, the iteration is stopped, and the current estimate of the stream function is accepted as the solution.

Subroutine NEWRHO

Subroutine NEWRHO calculates the velocity magnitude and components and the density at each point of the orthogonal mesh. Figure 12 is a flow chart for NEWRHO.

Normally, the upstream and downstream flow conditions, including whirl, are given as a function of the stream function. However, this information may be given as a function of position from hub to tip. In this case, an initial estimate of streamline position is made in PRECAL. Then adjustments are made in each iteration. This is done in

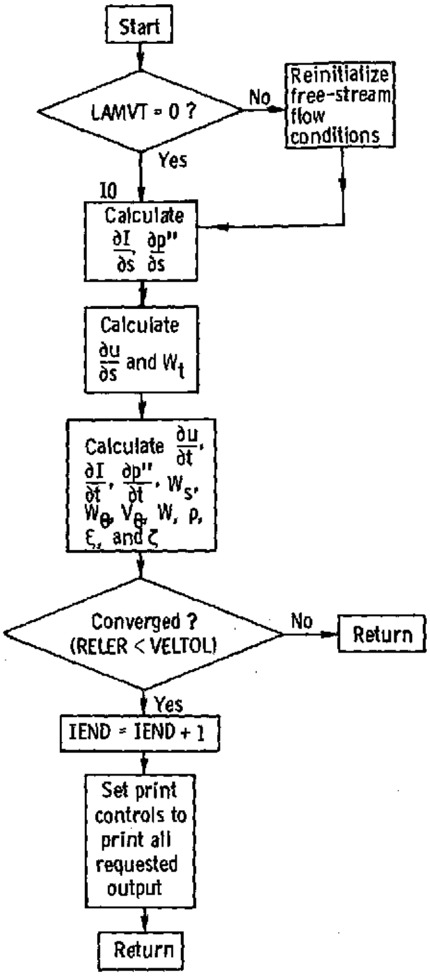


Figure 12. - Flow chart for NEWRHO.

NEWRHO by reinitializing the subroutines for calculating upstream and downstream flow conditions (LAMDAF, RVTHTA, TIPF, RHOIPF, and RHOOPF). An explanation of how upstream and downstream flow conditions are matched to the stream function solution is given in appendix B.

The main function of NEWRHO is to calculate the partial derivatives of the stream function in the s- and t-directions. These partials are used to calculate the velocity components. These components, together with either the blade shape or the specified whirl, determine the relative velocity magnitude. With the relative velocity known, the density can be calculated. Subroutine NEWRHO calculates ξ and ζ for the next iteration.

The first major loop in NEWRHO calculates $\partial I/\partial s$ and $\partial p''/\partial s$. This is done by first calculating I and p'' along the horizontal mesh lines. The actual relative stagnation pressure p'' is calculated by

$$p'' = p_i'RT_i'\left(\frac{T''}{T_i'}\right)^{\gamma/(\gamma-1)}\left(1 - \frac{p''_{ideal} - p''}{p''_{ideal}}\right) \tag{2}$$

where

$$\frac{T''}{T_i'} = 1 - \frac{2\omega\lambda - (\omega r)^2}{2c_pT_i'} \tag{3}$$

Equation (3) is the same as equation (B13) of part I with $W = 0$. The rothalpy I is calculated from equation (B7) of part I (ref. 6). Then, $\partial I/\partial s$ and $\partial p''/\partial s$ are calculated by calling the subroutine SLOPES.

The next loop calculates W_t. First, SPLINE is called to calculate $\partial u/\partial s$ along horizontal mesh lines. Then W_t is calculated by equation (G11) of part I.

The final major loop calculates partial derivatives in the t-direction and then calculates W_s, W_θ, V_θ, W, ρ, ξ, and ζ at every mesh point. The first inner loop calculates T''/T_i' and p'' by equations (3) and (2) along vertical mesh lines. The values of the t-coordinate and stream function u are also stored in temporary arrays. Then SPLINE is called to calculate $\partial u/\partial t$, and SLOPES is called twice to calculate $\partial I/\partial t$ and $\partial p''/\partial t$. The second inner loop performs the remaining calculations. Equation (G10) of part I is used to calculate W_s. Within the blade W_θ can be calculated from W_s, W_t, $\partial\theta/\partial s$, and $\partial\theta/\partial t$. Since

$$W_\theta = W_m \tan\beta$$

$$\tan\beta = r\frac{d\theta}{dm} = r\left(\frac{\partial\theta}{\partial s}\frac{ds}{dm} + \frac{\partial\theta}{\partial t}\frac{dt}{dm}\right)$$

$$\frac{ds}{dm} = \frac{W_s}{W_m}$$

$$\frac{dt}{dm} = \frac{W_t}{W_m}$$

we have

$$W_\theta = r\left(W_s \frac{\partial \theta}{\partial s} + W_t \frac{\partial \theta}{\partial t}\right)$$

within the blade. Outside the blade,

$$W_\theta = \begin{cases} \dfrac{\lambda}{r} - \omega r & \text{upstream of blade} \\[2ex] \dfrac{\left(rV_\theta\right)_o}{r} - \omega r & \text{downstream of blade} \end{cases}$$

Then V_θ and W are calculated by

$$V_\theta = W_\theta + \omega r$$

$$W = \sqrt{W_\theta^2 + W_s^2 + W_t^2}$$

The ideal density ρ is calculated by

$$\rho = \rho_i' \left(\frac{T}{T_i'}\right)^{1/(\gamma - 1)}$$

where T/T_i' is calculated by equation (B13) of part I (ref. 6). Then $\partial p/\partial r$ and $\partial I/\partial r$ are calculated by

$$\frac{\partial p}{\partial r} = \frac{\partial p}{\partial s} \sin \varphi + \frac{\partial p}{\partial t} \cos \varphi$$

$$\frac{\partial I}{\partial r} = \frac{\partial I}{\partial s} \sin \varphi + \frac{\partial I}{\partial t} \cos \varphi$$

Relative total temperature T'' is also needed and is calculated from previously calculated values of T''/T_i' and T_i'. This gives all the quantities needed to calculate ξ and ζ from equations (A2) and (A3) of part I.

After all calculations are done, the iteration number and the maximum relative change in velocity are printed. Also, if the solution is converged on velocity, the print control variables are set to 1 whenever a positive value is specified as input. This results in output being printed for each item asked for after convergence.

There are also two error messages for NEWRHO in case the velocity at some point becomes too large or if the upstream whirl is too large. Suggestions for correcting input are given in the section Error Messages in part I.

Subroutine OUTPUT

The OUTPUT subroutine calculates and prints all the major output data from MERIDL. A flow chart for OUTPUT is shown in figure 13. Depending upon the wishes of the user, OUTPUT has the potential for printing output on three separate sets of points. These points are illustrated in figure 14. Output may be obtained (1) at the orthogonal mesh points, (2) along streamlines where they are crossed by vertical orthogonal mesh lines, and (3) along streamlines where they are crossed by user-designated hub-shroud station lines. A detailed description of the output in each case is given in part I under Printed Output.

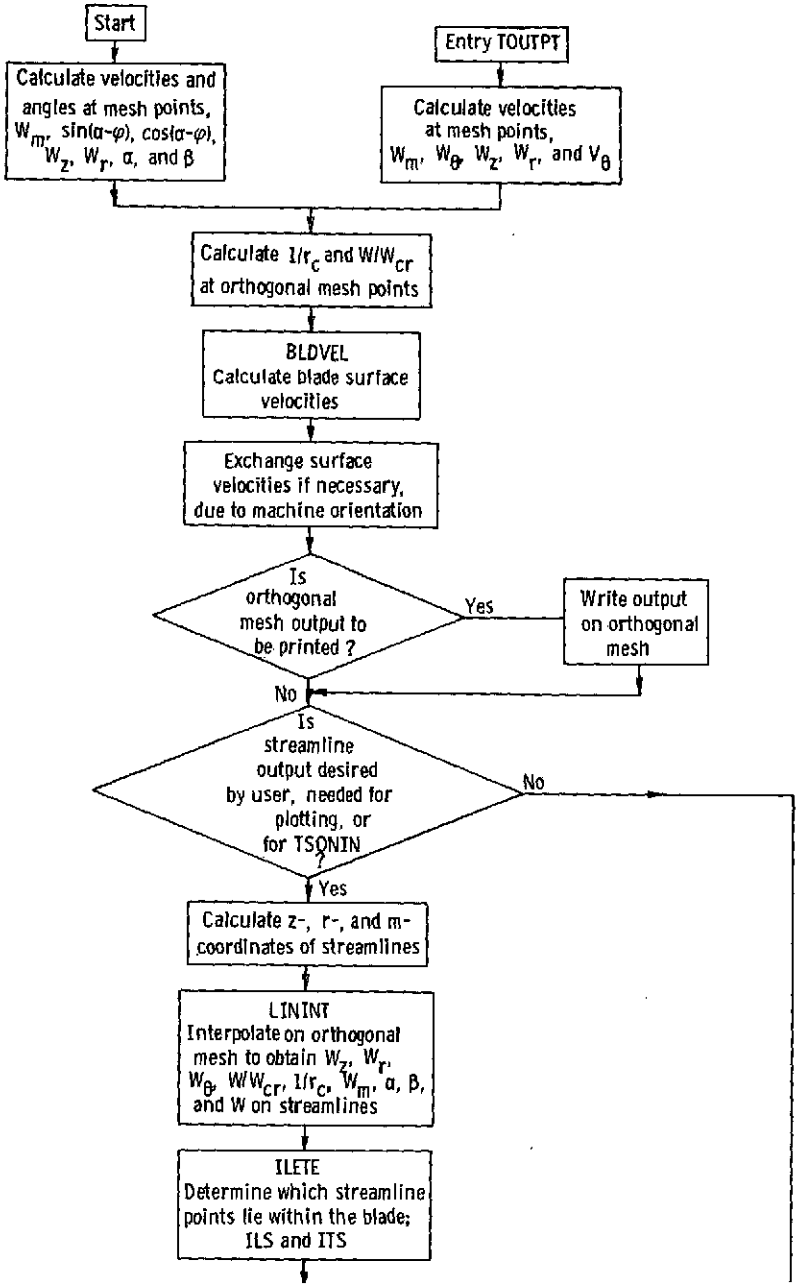
The printing of output is controlled by the iteration counter ITER and the input variables IMESH, ISLINE, and ISTATL. Because of the large volumes of output possible, it is only given at the locations requested by these variables and when ITER is an integer multiple of these variables.

No matter what the values of IMESH, ISLINE, and ISTATL, data are calculated at the orthogonal mesh points for every iteration. (Whether or not it is printed depends upon IMESH.) Output along streamlines and on station lines is then interpolated from the calculated data at the orthogonal mesh points if the values of ISLINE or ISTATL indicate that the user desires these outputs at the current iteration. Output along streamlines is also calculated if it is needed for plotting (controlled by IPLOT) or if it is needed for calculating the input to the TSONIC program (controlled by ITSON).

The first sections of the OUTPUT routine calculate data on the orthogonal mesh. At the main entry to this routine, W_s, W_t, and W_θ are known from NEWRHO; and the other velocity components and flow angles are calculated as follows:

$$W_m = \sqrt{W_s^2 + W_t^2}$$

$$\sin(\alpha - \varphi) = \frac{W_t}{W_m}$$


Start
Entry TOUTPT
Calculate velocities and angles at mesh points, W_m, $\sin(\alpha-\varphi)$, $\cos(\alpha-\varphi)$, W_z, W_r, α, and β
Calculate velocities at mesh points, W_m, W_θ, W_z, W_r, and V_θ
Calculate $1/r_c$ and W/W_{cr} at orthogonal mesh points
BLDVEL Calculate blade surface velocities
Exchange surface velocities if necessary, due to machine orientation
Is orthogonal mesh output to be printed ?
Yes
Write output on orthogonal mesh
No
Is streamline output desired by user, needed for plotting, or for TSONIN ?
No
Yes
Calculate z-, r-, and m-coordinates of streamlines
LININT Interpolate on orthogonal mesh to obtain W_z, W_r, W_θ, W/W_{cr}, $1/r_c$, W_m, α, β, and W on streamlines
ILETE Determine which streamline points lie within the blade; ILS and ITS

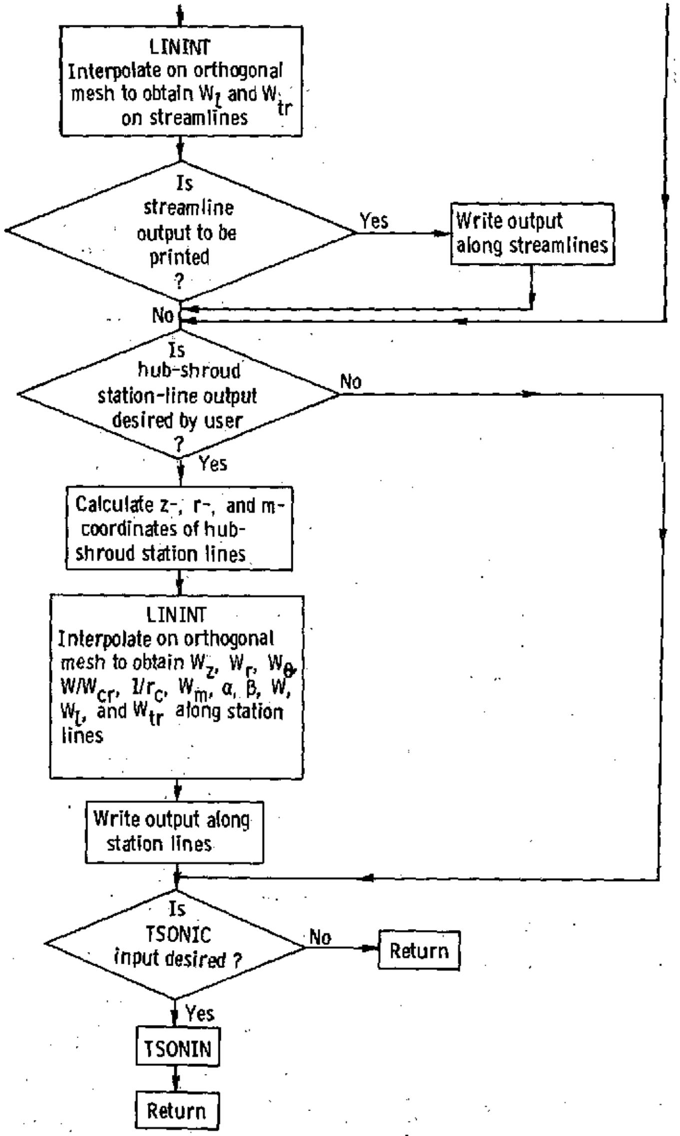


Figure 13. - Flow chart for OUTPUT.

$$\cos(\alpha - \varphi) = \frac{W_s}{W_m}$$

$$W_z = W_s \cos \varphi - W_t \sin \varphi$$

$$W_r = W_t \cos \varphi + W_s \sin \varphi$$

$$\alpha = \tan^{-1}\left(\frac{W_r}{W_z}\right)$$

$$\beta = \tan^{-1}\left(\frac{W_\theta}{W_m}\right)$$

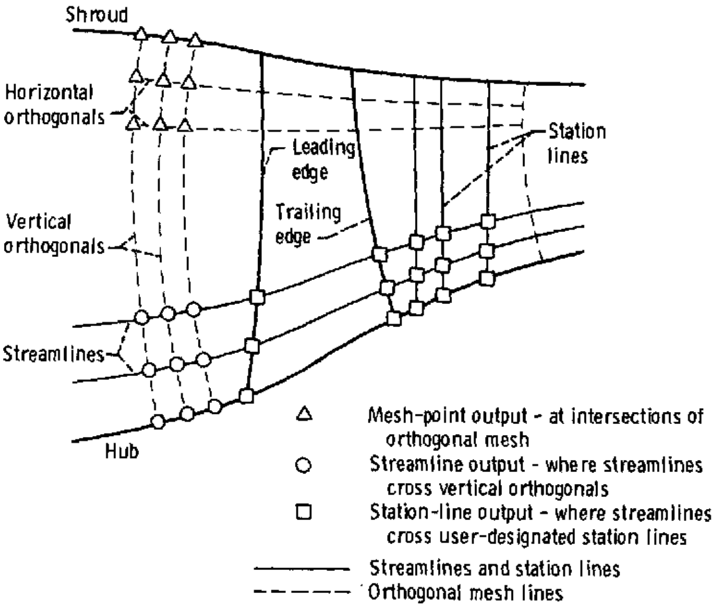


Figure 14. - Location of three major types of output.

This coding is followed by an entry point TOUTPT which is used only after TVELCY has been called to obtain transonic velocities (see the block diagram, fig. 2, when REDFAC < 1.0). From this entry point, the velocity components are calculated some-what differently since W has been recalculated by TVELCY, as well as β upstream and downstream of the blade. The angle α is assumed to be the same as in the final subsonic iteration. With W, β, and α known, the velocity components are now calculated as follows:

$$W_m = W \cos \beta$$

$$W_\theta = W \sin \beta$$

$$W_z = W_m \cos \alpha$$

$$W_r = W_m \sin \alpha$$

$$V_\theta = W_\theta + \omega r$$

At this point in the program, all velocity components and flow angles have been calculated, regardless of the entry point. With velocity components and flow angles known, streamline curvature is obtained from

$$\frac{1}{r_c} = \frac{d\alpha}{dm} = \frac{\partial \alpha}{\partial s} \cos(\alpha - \varphi) + \frac{\partial \alpha}{\partial t} \sin(\alpha - \varphi)$$

Then critical velocity ratio is obtained from

$$T'' = T_i' - \frac{2\omega\lambda - (\omega r)^2}{2c_p}$$

$$\frac{W}{W_{cr}} = \frac{W}{\sqrt{\frac{2\gamma R}{\gamma + 1} T''}}$$

The subroutine BLDVEL is then called to calculate and return an estimate of blade surface velocities. Finally, a check is made to see if the suction- and pressure-surface velocities have to be exchanged because of the orientation of the turbine or compressor. At this point, all desired information has been calculated on the orthogonal mesh and is printed if ITER is a multiple of IMESH.

The next section of the OUTPUT routine calculates output on the streamlines where they are intersected by vertical orthogonal mesh lines. This output is calculated only if ITER is a multiple of ISLINE, IPLOT, or ITSON. First, streamline z- and r-coordinates are calculated. The m-coordinates are then calculated from these, using the $z = 0$ point along a streamline to correspond to $m = 0$. Interpolations are then made by using LININT and the orthogonal mesh data to obtain W_z, W_r, W_θ, W/W_{cr}, and $1/r_c$. By using variations of the preceding formulas, W_m, α, β, and W are calculated from these values. Subroutine ILETE is called to establish which mesh points along streamlines are between the blade leading and trailing edges. Subroutine LININT is then used to obtain W_l and W_{tr} at these points. Finally, this output is printed if ITER is a multiple of ISLINE.

The next section of the OUTPUT routine calculates output on user-designated hub-shroud station lines where they intersect the streamlines. This output is calculated and printed in the hub-shroud direction, in contrast to the throughflow direction of the previous two sets of output. It is only calculated if ITER is a multiple of ISTATL. The z- and r-coordinates of the station lines are calculated first. All "regular" station lines are straight lines (not necessarily radial) from the hub to the shroud. "Blade edge" station lines are those whose hub and tip coordinates correspond to the intersections of the blade leading and trailing edges with the hub and tip. Coordinates along these station lines will follow these edges even when the edges are curved. After the z- and r-coordinates are established, m-coordinates are calculated from these, again using $z = 0$ as the reference for $m = 0$. Interpolations are then made using LININT with the orthogonal mesh values to obtain W_z, W_r, W_θ, W/W_{cr}, and streamline curvature. The values of W_m, α, β, and W are calculated from these. LININT is then called to obtain W_l and W_{tr} for station lines which lie within the blade. The station-line output is then printed.

The final small section of OUTPUT calls the TSONIN subroutine to obtain input data for the TSONIC program (ref. 5). This call is only made if ITER is a multiple of ITSON.

Subroutine BLDVEL

This subroutine calculates blade-surface velocities and densities and F_r. First, $\partial(rV_\theta)/\partial t$ and $\partial(rV_\theta)/\partial s$ are calculated by using the SLOPES subroutine. Then, $[d(rV_\theta)/dm]\, B \cos\beta$ is calculated, and W_l and W_{tr} are calculated by equation (G4) of part I (ref. 6). From this, ρ_l and ρ_{tr} are calculated by equations (B13) and (D4) of part I. The average density ρ_{av} is calculated by Simpson's rule

$$\rho_{av} = \frac{\rho_l + 4\rho_{mid} + \rho_{tr}}{6}$$

This quantity is used in NEWRHO in the next iteration. Then, the predicted value of F_r is calculated by

$$F_r = \frac{W}{B}\left(\frac{\partial\theta}{\partial s}\sin\varphi + \frac{\partial\theta}{\partial t}\cos\varphi\right)DFDM \qquad (4)$$

where

$$DFDM = -B\cos\beta\,\frac{\partial(rV_\theta)}{\partial m}$$

Equation (4) is obtained from equations (B23) and (G2) of part I (ref. 6). The new value for F_r is calculated from the old F_r and the predicted value of F_r by using the input damping factor FNEW, as explained in the section INPUT of part I.

At the end, the minimum and maximum predicted values of F_r and the maximum change in F_r are calculated and printed. If debug output is requested, the arrays which change each iteration are printed.

Subroutine ILETE

The points where streamlines are intersected by the vertical orthogonal mesh lines are the streamline mesh points. These are, in general, different from the orthogonal mesh points. Subroutine ILETE calculates two integer arrays, ILS and ITS. They con-

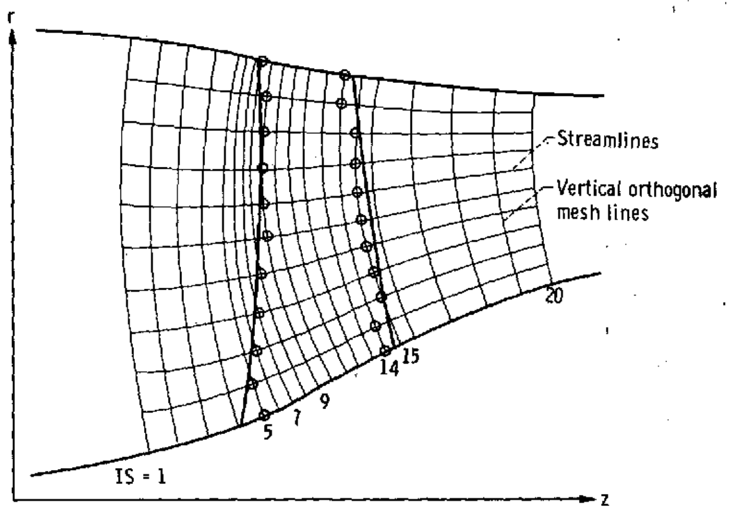


Figure 15. - Location of ILS, ITS points by ILETE.

tain the numbers of the vertical mesh lines at the first intersection of a streamline with a vertical mesh line inside the blade region at the leading and trailing edges of the blades. These points are illustrated in figure 15. The ILS and ITS arrays are used in OUTPUT in the calculation of blade surface velocities along streamlines.

Subroutine TSONIN

Subroutine TSONIN generates and prints the data required as input to the TSONIC blade-to-blade analysis program (ref. 5). Subroutine TSONIN is only called when ITER is a multiple of ITSON. The data generated are printed for each of the stream surfaces from hub to shroud, using 1 percent of the mass flow about a streamline to define a stream surface or flow channel.

A complete description of the TSONIC input is given in the TSONIC report (ref. 5). The output generated in TSONIN is slightly different from what is required by TSONIC. These differences and the changes which have to be made to make these data acceptable to TSONIC are described in part 1.

Subroutine INDEV

Subroutine INDEV recalculates $\partial\theta/\partial s$ to allow for incidence and deviation. This means that the mid-channel flow surface differs from the blade mean camber line near the leading and trailing edges, so as to match the upstream and downstream flow angles. Figure 16 shows the procedure as applied to the leading edge. A similar correction is made at the trailing edge. A correction for blockage is made so as to satisfy both continuity and tangential momentum at blade leading and trailing edges.

The calculation starts at the hub and proceeds to successive horizontal mesh lines up to the tip. Both incidence and deviation corrections are calculated for each horizontal mesh line. The corrected $(\partial\theta/\partial s)_{bf}$ of the blade leading or trailing edge is calculated from equations (F1) and (F2) of part I (ref. 6). These equations relate $(\partial\theta/\partial s)_{bf}$ to the flow angles β_{bf} and β_{fs}.

The corrections to $\partial\theta/\partial s$ are made so that the difference varies linearly from the blade leading or trailing edge for the distance specified in appendix F of part I. After the corrections are made, the incidence and deviation angles are printed if requested.

No correction is made to $\partial\theta/\partial t$ since it is nearly normal to the flow.

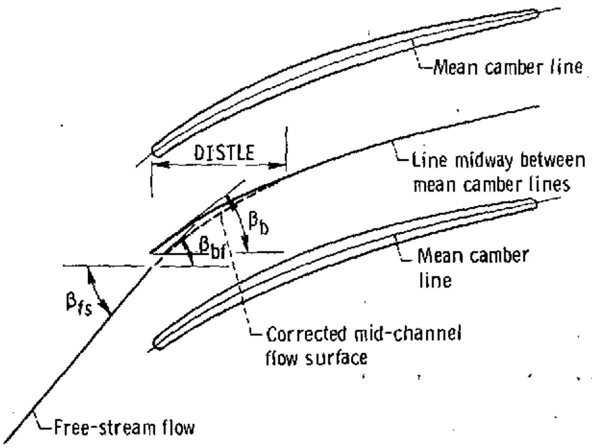


Figure 16. - Corrected mid-channel flow surface. The corrected mid-channel flow surface is used to calculate $(\partial\theta/\partial s)_{bf}$. Incidence = $\beta_{bf} - \beta_b$.

Subroutine SLPLOT

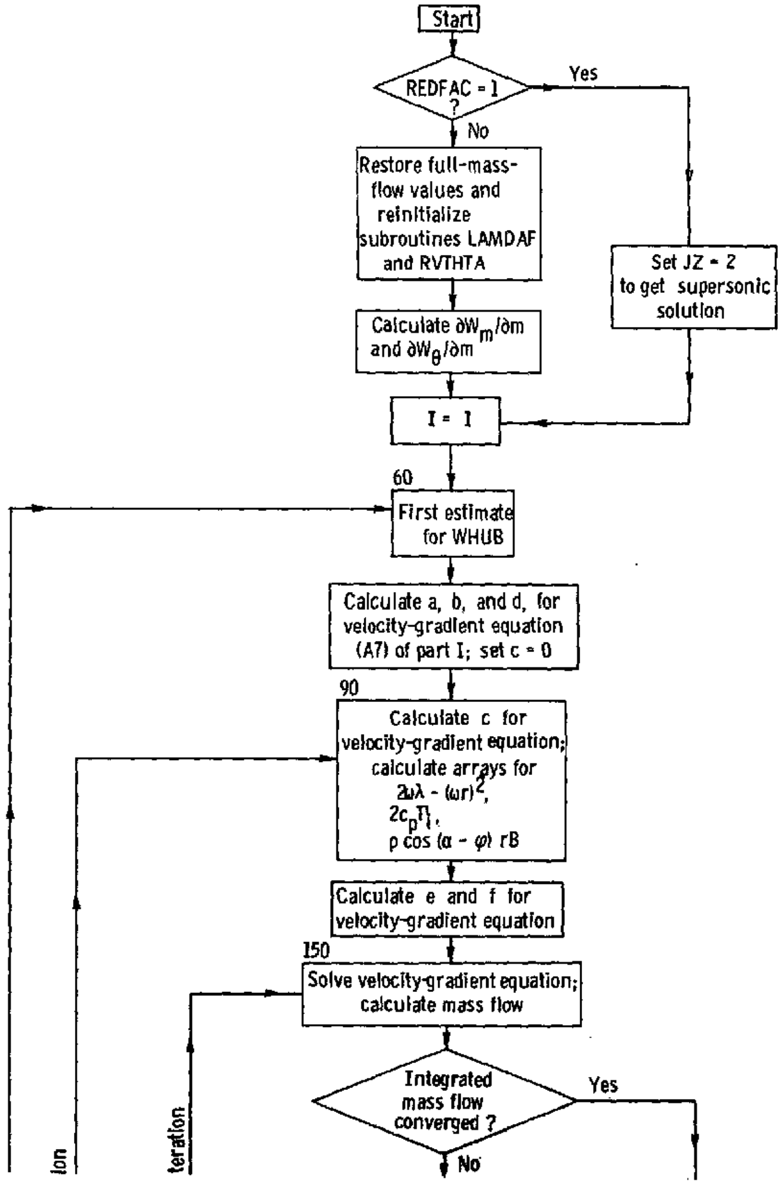
Subroutine SLPLOT makes a microfilm plot of the streamlines in the hub-shroud meridional flow plane. The first small section of coding plots a separate frame of film identifying either a subsonic solution and iteration number or a transonic solution. The remaining coding plots the hub, shroud, and blade profiles in the meridional plane and then adds the streamlines to the same plot. An example of this plot is given in figure 16(j) of part 1.

Subroutine SVPLOT

Subroutine SVPLOT makes microfilm plots of relative velocities on all streamlines from hub to shroud. These plots are only made when ITER is a multiple of IPLOT or when ITER = 1. A separate plot is made for the velocities on each streamline. These plots include mean flow velocities and blade surface velocities plotted against meridional coordinates. Examples of these plots are given in figures 16(k) and (l) of part I (ref. 6). After the separate streamline plots are made, three composite plots are made. The first contains the mean flow velocities for all streamlines. The second and third contain the suction- and pressure-surface velocities, respectively, for all streamlines.

Subroutine TVELCY

Subroutine TVELCY calculates the full-mass-flow, transonic solution when REDFAC is less than 1. The velocity-gradient equation developed in appendix C of part I is used to obtain the solution. Figure 17 is a flow chart for TVELCY.



Start
REDFAC = 1 ?
Yes
No
Restore full-mass-flow values and reinitialize subroutines LAMDAF and RVTHTA
Set JZ = 2 to get supersonic solution
Calculate ∂Wm/∂m and ∂Wθ/∂m
I = 1
60
First estimate for WHUB
Calculate a, b, and d, for velocity-gradient equation (A7) of part I; set c = 0
90
Calculate c for velocity-gradient equation; calculate arrays for 2ωλ - (ωr)², 2cpTi', ρ cos (α - φ) rB
Calculate e and f for velocity-gradient equation
150
Solve velocity-gradient equation; calculate mass flow
Integrated mass flow converged ?
Yes
No
ion
teration

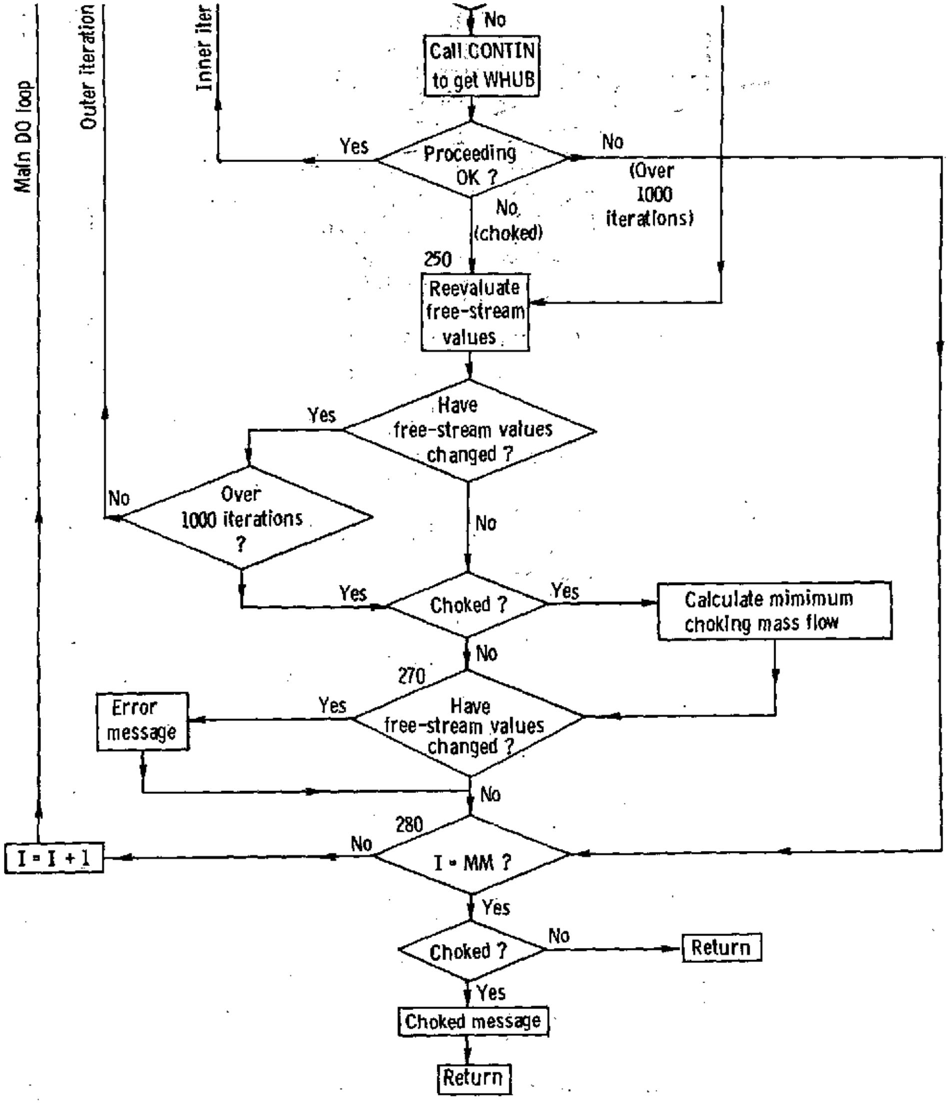


Figure 17. - Flow chart for TVELCY.

The first step in the program is to restore the full value of mass flow, rotational speed, and inlet and outlet whirl. The subroutines LAMDAF and RVTHTA must then be reinitialized.

Next, $\partial W_m/\partial m$ and $\partial W_\theta/\partial m$ are calculated. These are calculated from the partials with respect to s and t by using the angle $\alpha - \varphi$. Since the calculations are based on the reduced-mass-flow values of W_m and W_θ, the result must be divided by REDFAC to obtain the full-mass-flow values.

Statement 60 is the beginning of the outer DO loop. It starts at the upstream boundary and solves the velocity-gradient equation for each vertical mesh line. The initial estimate of W on the hub (WHUB) is set equal to the reduced-mass-flow value for W divided by REDFAC. For a given vertical line, the coefficients a, b, c, d, e, and f of the velocity-gradient equation (A7) of part I are calculated. The coefficients are calculated by equations (A8) to (A11) of part I. Of these coefficients, a, b, and d will not be changed after the initial calculation, so they are calculated first. The initial arrays for whirl, temperature, and density are calculated at the same time.

For each vertical mesh line, an inner and an outer iteration is required. Each outer iteration consists of solving the velocity-gradient equation for a given distribution of upstream and downstream flow conditions. The inner iteration solves the velocity-gradient equation by varying W_{hub} at each inner iteration until continuity is satisfied. The outer iteration starts at statement 90. None of the coefficients change during the inner iteration, so the remaining coefficients, c, e, and f, are calculated from equations (A9) to (A11) of part I before starting the inner iteration. Also, part of the integrand for the mass flow integration is calculated now. This part is RCARB, which is equal to $\rho_i' \cos(\alpha - \varphi) rB$.

At statement 140, the inner iteration starts. First, initial values are set. The numerical solution of the velocity-gradient equation and the mass flow integration are done in the DO 200 loop. Trial values of WHUB are used in the velocity-gradient equation, until the solution obtained results in the input mass flow across the vertical mesh line. The first iteration will use the value calculated by the statement after statement 60. Later iterations will use estimated values calculated by CONTIN. Once WHUB is specified, the numerical solution to the velocity-gradient equation is calculated by the Heun method, as described for VBDRY. The solution procedure is the same, except that dW in equation (1) is evaluated by equation (A7) of part I (ref. 6). The mass flow is calculated by trapezoidal integration of

$$w = \int_0^{t_{tip}} \rho W \cos \beta \cos(\alpha - \varphi) rB \, dt \tag{5}$$

As explained in appendix D of part I (ref. 6), ρ is the ideal density and B is reduced to reflect any loss of stagnation pressure.

The inner iteration ends when the velocity-gradient solution gives the correct mass flow in equation (5). (If the correct mass flow is not obtained in 100 iterations, an error message is printed, and the program goes to the next vertical line.) After the end of the inner iteration, at statement 250, the upstream and downstream flow conditions are checked. If there is a significant change in the value of inlet or outlet stagnation temperature or density, or whirl, these values will be adjusted and the inner iteration will be repeated by going back to statement 90, unless there has been a total of over 1000 iterations for a given vertical mesh line. The outer iteration is completed when there is no significant change in the solution, and the program goes to the next vertical line (the DO 280 loop). After all vertical lines have been completed, control is returned to the main program. If the blade is choked, a message is printed with the choking mass flow.

Function TOPF

Function TOPF calculates downstream stagnation temperature T'_o from the upstream stagnation temperature and the change in whirl. That is,

$$T'_o = T'_i + \frac{\omega\left[\left(rV_\theta\right)_o - \lambda\right]}{c_p}$$

The input argument (SF) is the value of the stream function (between 0 and 1). The function TOPF is then T'_o for this streamline.

Functions TIPF, RHOIPF, LAMDAF, RHOOPF, and RVTHTA

These five routines are similar. Their purpose is to calculate one of the free-stream quantities as a function of stream function. Interpolation is by means of a spline fit curve.

All these subroutines have an alternate entry point for initialization. The initializing call results in a SPLINE call to calculate the coefficients for the spline fit.

If the free-stream quantities are not given as input as a function of stream function (i.e., if LSFR = 1), the stream function is first estimated and later iterated to be adjusted to the correct stream-function value. These adjustments to the stream function (SFIN and SFOUT) are done in LAMDAF and RVTHTA.

The input argument for all these subroutines is SF, which is the value of the stream function.

Subroutine CONTIN

Subroutine CONTIN is a curve-fitting routine. On each call the calling programs must furnish a point on the curve, and then CONTIN will specify the next value of the abscissa. The calling program must then calculate the ordinate corresponding to this abscissa. After three calls, a parabola is fitted through the three points, and this is used to estimate the abscissa where the desired ordinate will be obtained. XEST is the value of the abscissa, and YCALC is the value of the ordinate on each call. XEST is changed by CONTIN to return the next value of the abscissa to the calling program.

Figure 18 is a flow chart for CONTIN. Flow through the program is controlled by the value of IND. For each new case, IND is set to 1 by the calling program. Then CONTIN changes the value of IND on later calls. The significance of IND on the various calls is given in table II. XDEL is the maximum increment for the change in XEST. On the first two calls, usually XEST is increased by XDEL each time. The exception is when YCALC is greater than YGIV and the subsonic solution is desired (JZ = 1). Then XEST is decreased by XDEL each time.

On the third and later calls, there are always three points so that a parabola can be fitted through the three points. The parabolic coefficients are calculated by subroutine PABC. Anytime that XEST falls outside the range of previously calculated values, a shift is made until XEST is within the desired range.

When the parabolic curve is close to a straight line, equation (D13) is used instead of the quadratic formula. The reason for this is explained in appendix D.

Figure 19 illustrates the procedure for a typical case. On the first call to CONTIN, IND = 1 and YCALC corresponding to XEST is furnished by the calling program. Suppose that YCALC is less than YGIV and that the subsonic solution is requested. Then XEST becomes XORIG, and YCALC becomes Y(1) in figure 19. XORIG will be the origin for the curve fitting so that X(1) = 0 in this case. Next CONTIN increases XEST by XDEL. Then a return is made to the calling program to obtain the YCALC which corresponds to this value of XEST. On the second call to CONTIN, the new value of YCALC becomes Y(2) and XEST - XORIG becomes X(2), as indicated in figure 19. Subroutine CONTIN increases XEST by XDEL again, and a return is made to obtain YCALC for the third time. On the third call to CONTIN, the new value of YCALC becomes Y(3) and XEST - XORIG becomes X(3). This gives the three points shown in figure 19. The curve shown represents the true curve of YCALC against XEST.

At this time, a check is made to determine whether the solution is within the range

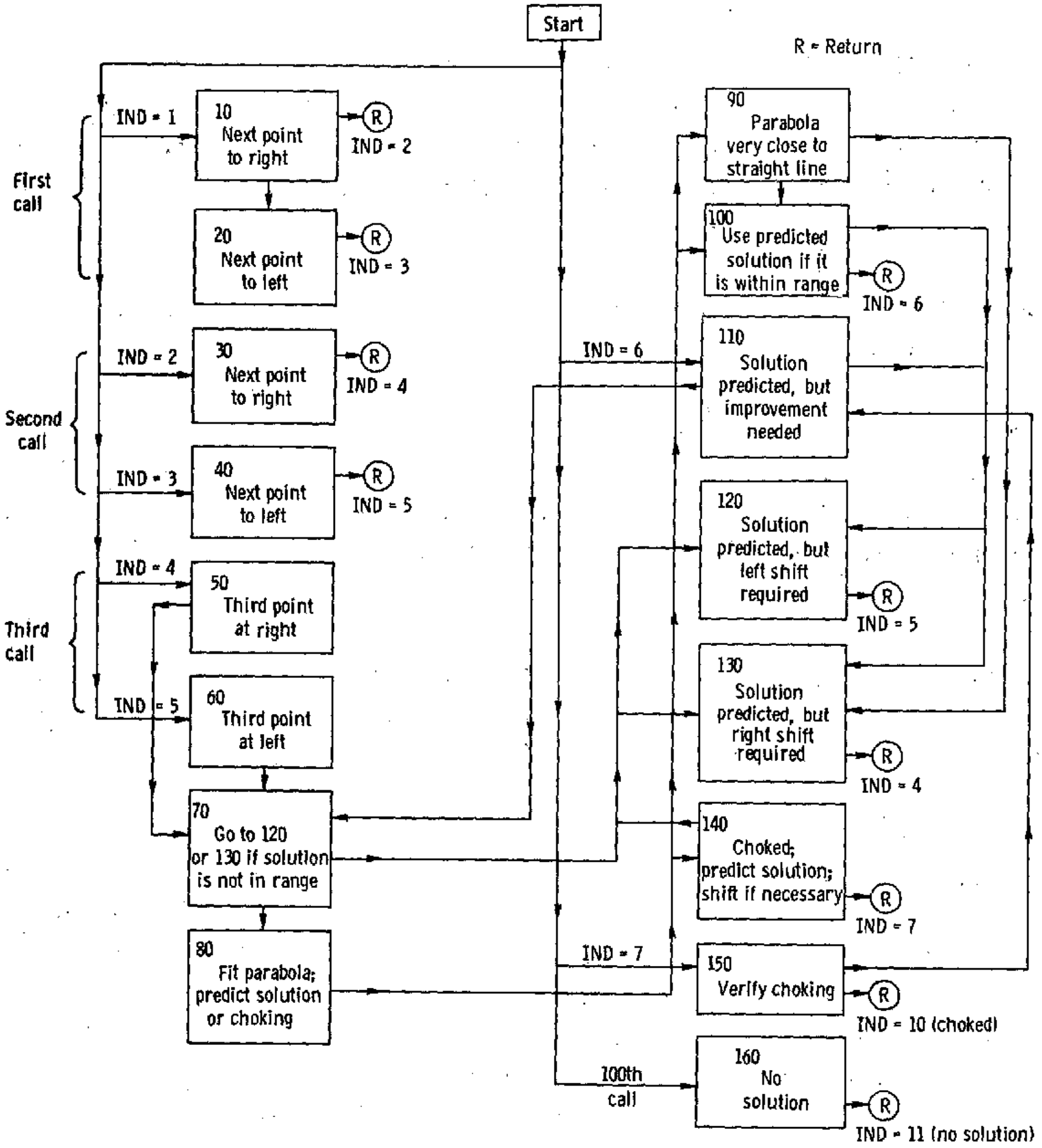


Figure 18. - Flow chart for CONTIN.

TABLE II. - SIGNIFICANCE OF IND IN VARIOUS

CALLS TO CONTIN

Value of IND	Call	Significance
1	First	First call
2	Second	JZ = 1, YCALC less than WTFL, or JZ = 2
3	Second	JZ = 1 and YCALC greater than WTFL
4	Third	IND = 2 on second call
	Fourth or later	Right shift was made so that XEST will be within range of stored previous values.
5	Third	IND = 3 on second call
	Fourth or later	Left shift was made so that XEST will be within range of stored previous values.
6	Fourth or later	Subsonic or supersonic solution is predicted by quadratic fit and is within range of solutions obtained.
7	Fourth or later	Choked flow is predicted by quadratic fit and is within range of solutions obtained.
10	Never	Choked solution found
11	Never	100 calls made but no solution found

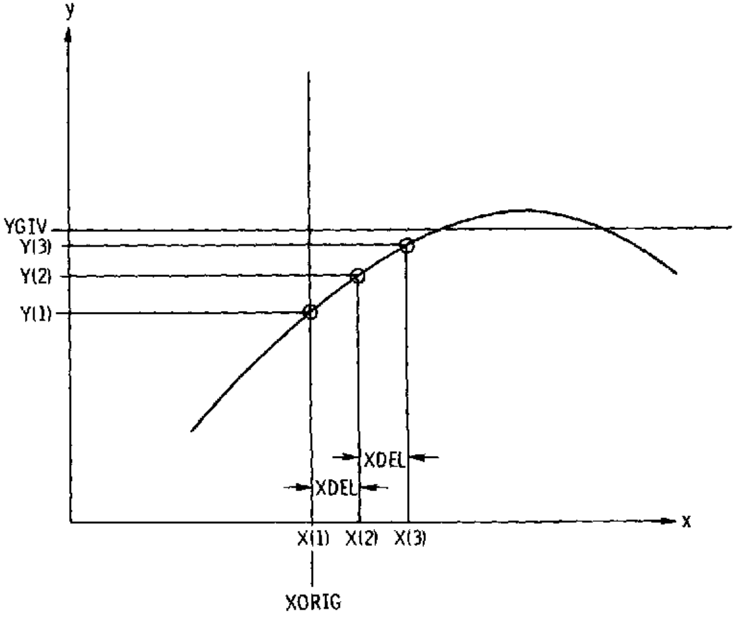


Figure 19. - Starting procedure for CONTIN.

of the three points obtained. If not, additional points are calculated, and the three points are shifted as required. For example, in figure 19, a shift to the right is required. In this case, point 2 would become point 1, point 3 would become point 2, and XEST would be increased by XDEL. This procedure is repeated until either the solution or the maximum point is within the range of the three points obtained.

Since the curve represents mass flow as a function of the velocity at some point, the curve will be of the type shown. The maximum point on the curve is the choking mass flow. This type of curve is approximated well by a quadratic curve. After it has been determined that a solution is within the range of the three points (i.e., $Y(1) \leq YGIV \leq Y(3)$ for a subsonic solution), a parabola is fitted through the three points. This situation is illustrated in figure 20. The next value of XEST is deter-

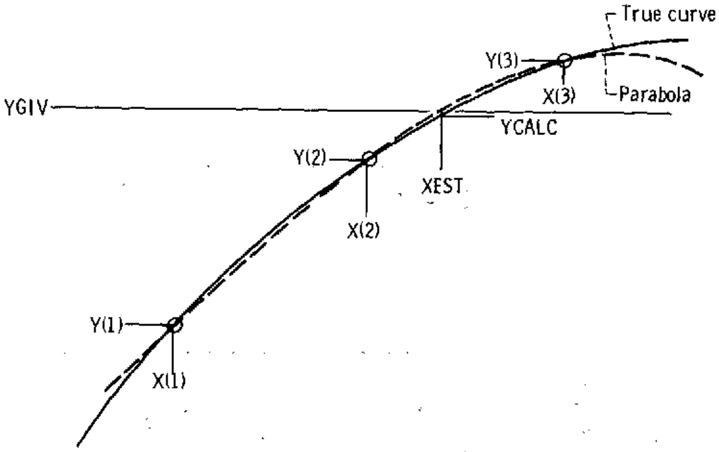


Figure 20. - Approximating curve with a parabola.

mined by the point where the parabolic curve intersects the YGIV line. Then the return is made to obtain YCALC. If YCALC is sufficiently close to YGIV, this will be the solution. Otherwise, CONTIN is called again, XEST - XORIG becomes X(2), YCALC becomes Y(2), and the procedure is repeated (as many as 100 times) until YCALC is sufficiently close to YGIV.

The detailed operation of subroutine CONTIN is given in figure 18 and table II. The calling statement for CONTIN is

```
CALL CONTIN(XEST,YCALC,IND,JZ,YGIV,XDEL)
```

The input variables for CONTIN are

XEST last value of X used to calculate YCALC

YCALC value of Y corresponding to XEST (calling program calculates YCALC)

IND controls sequence of calculation in CONTIN; calling program sets IND = 1 to indicate a new solution

JZ determines whether subsonic or supersonic solution will be obtained:

JZ = 1, subsonic solution

JZ = 2, supersonic solution

YGIV value of Y desired for solution

XDEL maximum permissible change in XEST between iterations

The output variables for CONTIN are

XEST value of X to be used to calculate the next value for YCALC

IND used to control next iteration in CONTIN and to indicate when a choked solution is found or when no solution can be found (table II)

The internal variables for CONTIN are

ACB2 $a(c - y)/b^2$

APA coefficient a of X^2 in quadratic fit

BPB coefficient b of X in quadratic fit

CPC constant C in quadratic fit

DISCR discriminant, $\sqrt{b^2 - 4ac}$

NCALL number of times CONTIN has been called for a given case

X array of three values of XEST - XORIG

XORIG value of XEST on initial call, modified by right or left shifts

XOSHFT amount of change of XORIG

Y array of three values of YCALC

Subroutine PABC

Subroutine PABC calculates coefficients A, B, and C of the parabola $y = Ax^2 + Bx + C$ passing through three given X, Y points.

Subroutine INRSCT

Subroutine INRSCT calculates the coordinates of the point of intersection of two spline curves lying on a common plane which are known to cross within the range of the end points of each. In a general x-y coordinate system, the first spline curve is supplied to INRSCT as a function of x

$$y = f(x)$$

and the second as a function of y

$$x = g(y)$$

The solution technique consists of systematically constructing pairs of tangent slopes to the two curves and locating the points of intersection of the two slopes. Each intersection point provides new coordinates from which new slopes and an intersection are calculated. These intersections quickly converge to the intersection point of the original curves.

This technique is illustrated in figure 21. The original trial x-coordinate is always

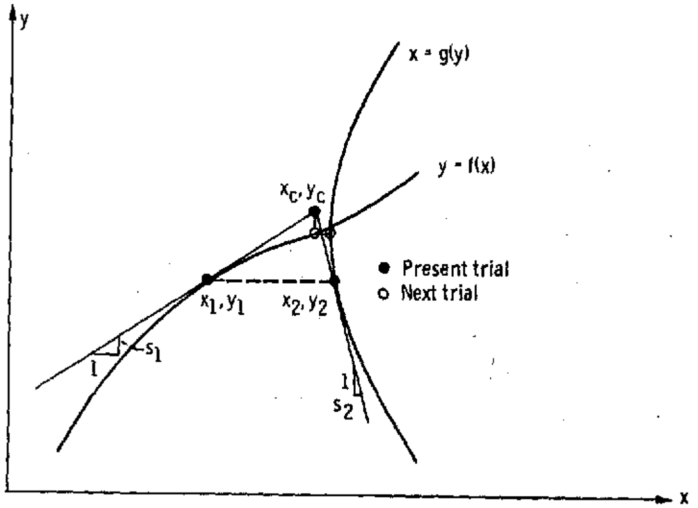


Figure 21. - Procedure for calculating intersections in INRSCT.

midway between the end points for f(x). This value is x_1, from which y_1 and slope s_1 are calculated by SPLINT. The calculated y_1 is then used as input to SPLINT for g(y). From this SPLINT call, x_2 and s_2 are calculated, as shown in figure 21. The intersection point of the two slopes is calculated from

$$x_c = x_2 + \frac{s_1 s_2 (x_2 - x_1)}{1 - s_1 s_2}$$

$$y_c = y_1 + \frac{s_1 (x_2 - x_1)}{1 - s_1 s_2}$$

Then x_c becomes x_1 for the following iteration of this process.

To check convergence of this process, the distance is calculated between each pair of intersection points x_c, y_c for adjacent iterations. When this distance becomes less than the tolerance, an exit is made from INRSCT. Failing to meet the tolerance in 20 iterations causes an error message to be printed.

The calling statement for subroutine INRSCT is

CALL INRSCT(XCURV1,YCURV1,N1,XCURV2,YCURV2,N2,XCROSS,YCROSS)

The input arguments for INRSCT are

XCURV1(N1)	x-coordinates for $f(x)$
YCURV1(N1)	y-coordinates for $y = f(x)$
XCURV2(N2)	x-coordinates for $x = g(y)$
YCURV2(N2)	y-coordinates for $g(y)$
N1	number of spline points for $f(x)$
N2	number of spline points for $g(y)$

The output arguments for INRSCT are

XCROSS	x-coordinate of intersection of two input curves
YCROSS	y-coordinate of intersection of two input curves

Subroutine ROOT

Subroutine ROOT finds a root for $f(x) = y$ by the bisection method. The function $f(x)$ must be defined on the interval $[a, b]$ by the subroutine FUNCT. FUNCT is a dummy name; any subroutine name may be used in the calling program. In MERIDL, FUNCT is CROSCD.

The interval is bisected 20 times by ROOT. This gives a resolution of x of 10^{-6} times the interval length. After the root has been located, the difference $f(x) - y$ is checked to see if it is less than TOLERY. If not, a message is printed with details on the iterated calculations.

The calling statement for ROOT is

```
CALL ROOT(A,B,Y,FUNCT,TOLERY,X,DFX)
```

The input arguments for ROOT are

A	a
B	b
Y	y
FUNCT	external subroutine to calculate $f(x)$
TOLERY	tolerance on solution (x is accepted as a root if $\lvert f(x) - y \rvert < \text{TOLERY}$)

The output arguments for ROOT are

X	value at x such that $f(x) = y$
DFX	$f'(x)$

The calling sequence for FUNCT must be

```
FUNCT(X, FX, DFX, )
```

These arguments are defined as follows:

X	x
FX	$f(x)$
DFX	$f'(x)$

Subroutine LININT

Subroutine LININT is a general-purpose subroutine for two-dimensional interpolation. It is called many times by several subroutines.

Subroutine LININT locates the point (x_o, y_o) in a two-dimensional mesh with coordinates stored in the x and y arrays. Then the value of z_o at x_o, y_o is interpolated from the z-array values corresponding to the x and y arrays. Figure 22 is a flow chart for LININT.

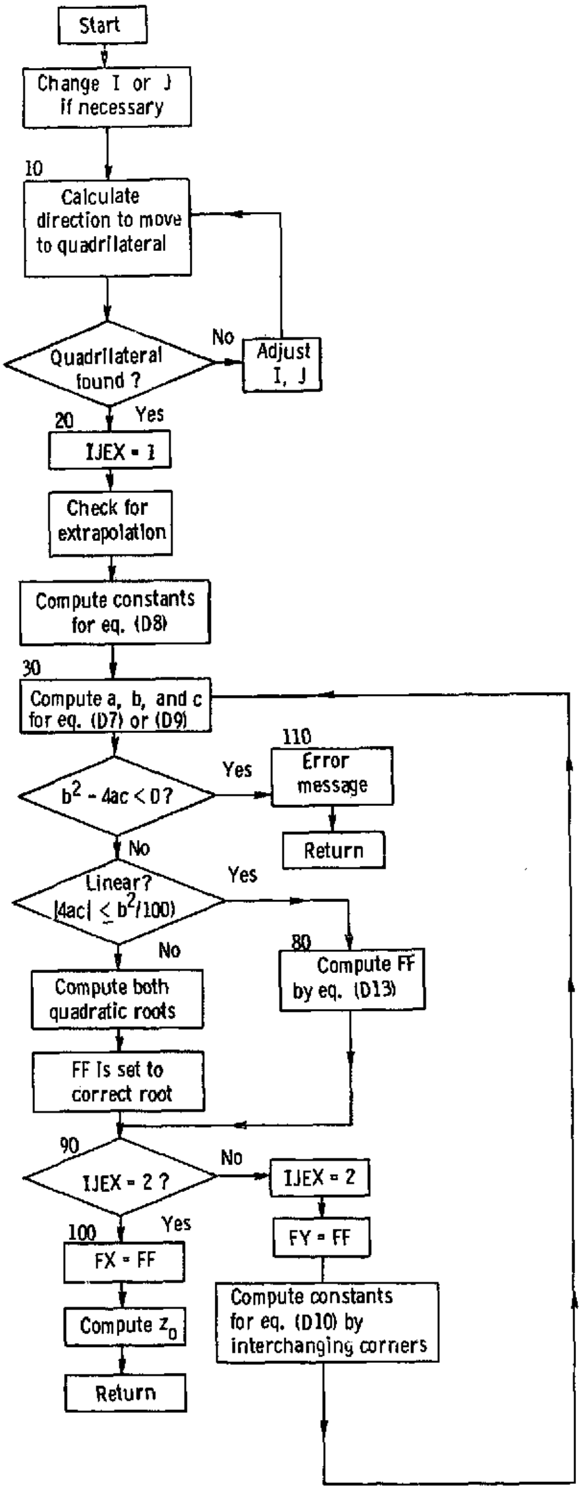


Figure 22. - Flow chart for LININT.

A typical mesh is shown in figure 23. The mesh need not be orthogonal; but it must consist of two sets of lines, with one set running more or less horizontally (never verti-

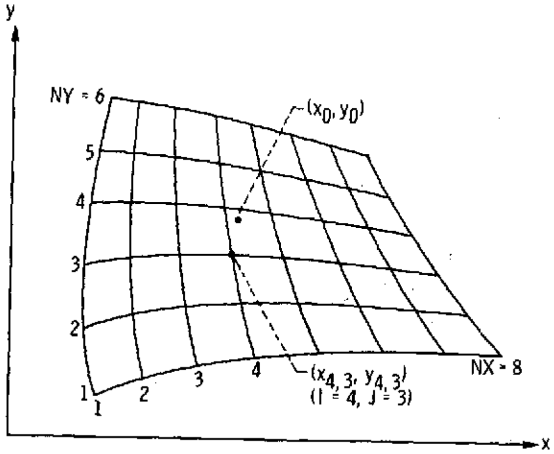


Figure 23. - Typical mesh for LININT.

cal) and the other set running more or less vertically (never horizontal). The number of vertical lines is NX, and I denotes the number of the line (running from 1 at the left to NX at the right). The number of horizontal lines is NY, and J denotes the number of the line (running from 1 at the bottom to NY at the top). The lines between mesh points are assumed to be straight lines.

At the outset, some value of I and J must be specified. Any value within the prescribed limits is legal. On repeated calls to LININT, usually the value from the preceding call is used. The values of I and J desired are the numbers shown at the bottom of figure 23. In this figure I = 4, J = 3. The procedure is to check to see on which side of each of the four boundary lines the point lies. The variables ABOVE and RIGHT are used to indicate the position. ABOVE = -1 indicates the point is below the bottom line, ABOVE = 0 the point is between the bottom and top lines, and ABOVE = 1 the point is above the top line. Similarly, RIGHT = -1 indicates the point is to the left of the left line, RIGHT = 0, the point is between the left and right lines, and RIGHT = 1 the point is to the right of the right line. Thus, when ABOVE = RIGHT = 0, we have the correct mesh region. If not, I and/or J are incremented by plus or minus 1 to move to the proper adjacent region. In this way, eventually the proper region will be found. If the point lies entirely outside the region defined, the nearest mesh region to the point (x_0, y_0) will be found. In this case, extrapolation is required, and the variable EXTRAP is used to indicate the direction of extrapolation. EXTRAP is dimensioned 2. EXTRAP(1) corresponds to ABOVE, and EXTRAP(2) to RIGHT.

After the proper mesh-point region is found, interpolation between the function values at the four corners is used. The method used is described in appendix D. First, the quadratic coefficients are calculated by equation (D8) or (D10). Then, the quadratic equation (D7) or (D9) is solved either by the quadratic formula, or by the binomial expansion, equation (D13), as explained in appendix D.

The same coding is used to calculate both f_x and f_y. After these values are obtained, equation (D14) is used to calculate the interpolated value of z_o.

The calling statement for LININT is

```
CALL LININT(X,Y,Z, NX,NY,NDIMX,NDIMY, X0,Y0,Z0,I,J)
```

The input variables for LININT are

X	two-dimensional array of x-coordinates of mesh points
Y	two-dimensional array of y-coordinates of mesh points
Z	two-dimensional array of z-function values at mesh points
NX	number of mesh points in the x-direction
NY	number of mesh points in the y-direction
NDIMX	dimension of X, Y, and Z arrays in the x-direction
NDIMY	dimension of X, Y, and Z arrays in the y-direction
X0	x-coordinate of interpolation point
Y0	y-coordinate of interpolation point
I	initial guess at number of vertical mesh line to the left of (X0, Y0)
J	initial guess at number of horizontal mesh line below (X0, Y0)

The output variables for LININT are

Z0	interpolated value of Z at (X0, Y0)
I	number of vertical mesh line to the left of (X0, Y0)
J	number of horizontal mesh line below (X0, Y0)

The internal variables for LININT are

ABOVE	integer, 1 indicates (X0, Y0) is above the current I,J region, 0 within, and -1 below
ACB2	ac/b^2 (eq. (D13))
CASE	used to indicate whether F1 or F2 is the proper solution
DISCR	discriminate, $b^2 - 4ac$ (eq. (D7) or (D9))

EXTRAP	array to indicate extrapolation either horizontally or vertically
FA	$-b/2a$ (eq. (D7) or (D9))
FB	$\sqrt{(b^2 - 4ac)}/2a$ (eq. (D7) or (D9))
FF	f_x or f_y
FX	f_x
FY	f_y
F1	$\left(-b - \sqrt{b^2 - 4ac}\right)/2a$
F2	$\left(-b + \sqrt{b^2 - 4ac}\right)/2a$
IJEX	indicator, first or second pass through coding to calculate f_x or f_y
IN	new value for I
JN	new value for J
QA	a (eq. (D8) or (D10))
QB	b (eq. (D8) or (D10))
QC	c (eq. (D8) or (D10))
RIGHT	integer, 1 indicates X0, Y0 is to the right of the current I, J region, 0 within, and -1 left
X01	x_{01} (see appendix D for notation)
X02	x_{02} or x_{03}
X13	x_{13} or x_{12}
X21	x_{21} or x_{31}
X42	x_{42} or x_{43}
Y01	y_{01}
Y02	y_{02} or y_{03}
Y13	y_{13} or y_{12}
Y21	y_{21} or y_{31}
Y42	y_{42} or y_{43}

Subroutine SPLINE

Subroutine SPLINE calculates the first and second derivatives of a cubic spline curve at the spline points. SPLINE solves a tridiagonal matrix given in reference 12 to obtain the coefficients for the piecewise cubic polynomial function giving the spline fit curve. The SPLINE routine is based on the end-point condition that the second derivative at either end point is one-half that of the next spline point.

The calling statement for SPLINE is

```
CALL SPLINE(X,Y,N,SLOPE,EM)
```

The input variables for SPLINE are

X	array of ordinates
Y	array of function values corresponding to X
N	number of X and Y values given

The output variables for SPLINE are

SLOPE	array of first derivatives
EM	array of second derivatives

Subroutine SPLINT

Subroutine SPLINT is used for interpolation, including interpolation of the derivative. The interpolation is based on the cubic spline curve, with the same end conditions as SPLINE. The alternate entry point, SPLENT, allows for interpolation at a new set of points based on the spline curve of the previous SPLINE call.

The input variables for SPLINT are

X	array of spline point ordinates
Y	array of function values at spline points
N	number of X and Y values given
Z	array of ordinates at which interpolated values and derivatives are desired
MAX	number of Z values given

The output variables for SPLINT are

YINT	array of interpolated function values
DYDX	array of interpolated derivatives

Subroutine SLOPES

Subroutine SLOPES calculates the first derivatives (slopes) based on a parabolic fit through three adjacent points. This subroutine is used when the input points may not be sufficiently smooth for the SPLINE subroutine.

The calling statement for subroutine SLOPES is

CALL SLOPES(X,Y,N,SLOPE)

The input arguments for SLOPES are

X	array of ordinates
Y	array of function values corresponding to X
N	number of X and Y values given

The output variable for SLOPES is

SLOPE	array of first derivatives

MAIN DICTIONARY

The main dictionary for MERIDL is given in this section. It contains the definitions of variables for all the principal subroutines (from INPUT to RVTHTA, see table of contents) of the program. The remaining subroutines (CONTIN to SLOPES) are of a general-purpose nature and have their own local dictionaries included in their descriptions.

All important variables are included in the main dictionary. These include all COMMON variables, any dimensioned variables in the subroutines, and all important undimensioned variables. Only locally used undimensioned variables of minor importance are not included.

The names of all dimensioned variables are followed by the variables which determine what the dimensions should be. For example, the three-dimensional array A is dimensioned A(4,100,101) in the /VARCOM/ COMMON but is listed as A(4,MM,MHTP1) in the dictionary. This enables the user to easily alter the dimension of A (and reduce the program's variable storage) if he knows maximum limits to MM and MHTP1 for his application. See the section STORAGE REQUIREMENTS for further explanation.

The dictionary also indicates the COMMON blocks or the subroutines in which each variable is used. Variables in COMMON are used in many subroutines. The COMMON blocks are listed for each subroutine in table I.

Variable name	COMMON block	Subroutine	Description and comments
A		MESHO	left-hand boundary on an interval of z-coordinate, m
A(4,MM,MHTP1)	VARCOM		coefficients of finite-difference equation (A7) for stream function, u
A0		COEF	a_o (eq. (A8))
AA(MHTP1)		VBDRY	coefficients, a, of velocity-gradient equation ((C9), part I)
AAA(NHUB)		MESHO	dummy array of slopes of a spline fit of horizontal rows in RRAD array
AAA(100)		PTBDRY	dummy array of slopes of spline fit curves
AAA(MHTP1 or MM)		NEWRHO	dummy array used in SPLINE calls
AAA(MHTP1 or NOSTAT or NSL or 20)		OUTPUT	dummy array used in SPLINT calls
AAA(NIN or NOUT)		LAMDAF RVTHTA	dummy array used in SPLINT calls
AANDK(integer variable)		TSONIN	input for TSONIC (ref. 5)
ALPHA(MM,MHTP1)	VARCOM		α at orthogonal mesh points, rad
ALPHLE		INDEV	α_{le}, rad
ALPHTE		INDEV	α_{te}, rad
ALPSL(MM,NSL)	SLCOM		α at points along streamlines where they cross vertical mesh lines, rad

Variable name	COMMON block	Subroutine	Description and comments
ALPST(NSL,NOSTAT)	STACOM		α at points along station lines where they cross streamlines, rad
ALVERT(MHTP1)		OUTPUT	temporary storage for values of α from ALPHA array on vertical mesh lines, rad
ANG(NPPP)		INPLOT THIKOM TSONIN	angles from meridional plane of blade-section mean camber lines at blade-section input points, rad
ANGR(11,NBLPL)		THETOM	angles with respect to radius of hub-shroud lines of alternate mesh (fig. 26), rad
ANGZ(11,NBLPL)		THETOM	angles with respect to z-axis of input blade sections at alternate mesh points (fig. 26), rad
AR	INPUTT		input gas constant, R, J/(kg)(K)
ATVEL(MHTP1)		TVELCY	coefficients, a, of velocity-gradient equation ((A7), part I) at orthogonal mesh points along vertical mesh lines
B		MESHO	right-hand boundary on interval of z-coordinate, m
BB(MHTP1)		VBDRY	coefficients, b, of velocity-gradient equation ((C9), part I)

Variable name	COMMON block	Subroutine	Description and comments
BDY(4)		VBDRY	variable containing words INLET and OUTLET used in printing error message
BESP(MM)		TSONIN	normal thicknesses of a stream channel, at MR,RMSP points, printed as input for TSONIC (ref. 5), m
BETA(MM,MHTP1)	VARCOM		β at orthogonal mesh points, rad
BETAI		TSONIN	β_i at blade leading edge (input to TSONIC, ref. 5), deg
BETAO		TSONIN	β_o at blade trailing edge (input to TSONIC, ref. 5), deg
BETSL(MM,NSL)	SLCOM		β at points along streamlines where they cross vertical mesh lines, rad
BETST(NSL,NOSTAT)	STACOM		β at points along station lines where they cross streamlines, rad
BLDAT(integer variable)		TSONIN	input for TSONIC (ref. 5)
BLDCRD		INDEV	true blade chord along a horizontal mesh line, m
BLDEV		INDEV	deviation angle, corrected for blockage, where a horizontal mesh line intersects trailing edge $\left(\beta_{bf} - \beta_b\right)_{te}$, deg

Variable name	COMMON block	Subroutine	Description and comments
BLINC		INDEV	incidence angle, corrected for blockage, where a horizontal mesh line intersects leading edge $\left(\beta_{bf} - \beta_b\right)_{le}$, deg
BLNK		INPLOT	blank word used in some plot titles
BRNG		INPLOT SVPLOT	bottom or lower range of values on a given plot
BTABLD		INDEV	blade mean camber line angle at leading or trailing edge, β_b, rad
BTFSLE		INDEV	upstream flow angle, β_f, extrapolated linearly along a horizontal mesh line to blade leading edge, rad
BTFSTE		INDEV	downstream flow angle, β_f, extrapolated linearly along a horizontal mesh line back to blade trailing edge, rad
BTH(MM, MHTP1)	CALCON		B at orthogonal mesh points, rad (These values are corrected for total pressure loss through the blade row.)
BTHLE		INDEV	B_{le}, rad
BTHSL		TSONIN	B along a streamline, rad
BTHTE		INDEV	B_{te}, rad

Variable name	COMMON block	Subroutine	Description and comments
BTVEL(MHTP1)		TVELCY	coefficients, b, of velocity-gradient equation ((A7), part I) at orthogonal mesh points along vertical mesh lines
C1		COEF	c_1 (eq. (A8))
C2		COEF	c_2 (eq. (A8))
CAMP(MM, MHTP1)	VARCOM		$\cos(\alpha - \varphi)$ at orthogonal mesh points
CAMPLE		INDEV	$\cos(\alpha - \varphi)_{le}$
CAMPTE		INDEV	$\cos(\alpha - \varphi)_{te}$
CBETA		TVELCY	$\cos \beta$
CCA(MHTP1)		VBDRY	coefficients, c_a, of velocity-gradient equation ((C9), part I)
CCB(MHTP1)		VBDRY	coefficients, c_b, of velocity-gradient equation ((C9), part I)
CHANGE		SOR	change in value of stream function at a mesh point during an overrelaxation iteration
CHFL		TVELCY	choking mass flow for a vertical orthogonal mesh line, kg/sec
CHLIM		TVELCY	minimum choking mass flow per passage, kg/sec

Variable name	COMMON block	Subroutine	Description and comments
CHORDF		TSONIN	length of blade section along streamline in m-direction (input to TSONIC, ref. 5), m
COSBTA		VBDRY	cos β
CP	CALCON		c_p, J/(kg)(K)
CPHI(MM, MHTP1)	CALCON		cos(φ) at orthogonal mesh points
CPHILE		INDEV	$\cos(\varphi_{le})$
CPHITE		INDEV	$\cos(\varphi_{te})$
CPTIP(MHTP1)		TVELCY	$2c_pT_i'$ at orthogonal mesh points along vertical mesh lines, (N)(m)/kg
CTVEL(MHTP1)		TVELCY	coefficients, c, of velocity-gradient equation ((A7), part I) at orthogonal mesh points along vertical mesh lines
CURV(MM, MHTP1)	VARCOM		$1/r_c$ at orthogonal mesh points, 1/m
CURVH		VBDRY	CURVHI or CURVHO, 1/m
CURVHI	CALCON		curvature of hub at point where it is intersected by first (upstream) vertical orthogonal mesh line, 1/m
CURVHO	CALCON		curvature of hub at point where it is intersected by last (downstream) vertical orthogonal mesh line 1/m

Variable name	COMMON block	Subroutine	Description and comments
CURVSL(MM,NSL)	SLCOM		$1/r_c$ at points along streamlines where they cross vertical orthogonal mesh lines, 1/m
CURVST(NSL,NOSTAT)	STACOM		$1/r_c$ at points along station lines where they cross streamlines, 1/m
CURVT		VBDRY	CURVTI or CURVTO, 1/m
CURVTI	CALCON		curvature of shroud at point where it is intersected by first (upstream) vertical orthogonal mesh line, 1/m
CURVTO	CALCON		curvature of shroud at point where it is intersected by last (downstream) vertical orthogonal mesh line, 1/m
D1		COEF	d_1 (eq. (A8))
D2		COEF	d_2 (eq. (A8))
DALDS(MM)		OUTPUT	$\partial\alpha/\partial s$ at mesh points along horizontal mesh lines, rad/m
DALDT(MM,MHTP1)		OUTPUT	$\partial\alpha/\partial t$ at orthogonal mesh points, rad/m
DALVER(MHTP1)		OUTPUT	$\partial\alpha/\partial t$ at mesh points along vertical mesh lines, rad/m
DBL		TSONIN	one-half of tangential blade thickness (in radians) at intersection of a streamline with blade leading or trailing edge

Variable name	COMMON block	Subroutine	Description and comments
DCHANG		COEF	maximum value of change in estimated values of $\partial(rV_\theta)/\partial r$ at a mesh point between any two outer iterations, m/sec
DEGRAD		OUTPUT INDEV	conversion constant from radians to degrees
DEL		INPLOT	increment between plotted stream function or radius points
DELCH		OUTPUT	1 percent of average meridional chord length of blade, m
DELM		TSONIN	increment of meridional distance, m
DELMAX		VBDRY TVELCY	increment for W_{hub} at each iteration to satisfy continuity, m/sec
DELR		MESHO PTBDRY OUTPUT	increment in r-coordinate, m
DELRHO(MM, MHTP1)	VARCOM		difference in density, between suction and pressure surfaces, at orthogonal mesh points, kg/m^3
DELRNG		INPLOT	maximum range of points on a plot in either bottom-top or left-right directions
DELRTH		INPLOT	plotted blade spacing of a blade section, m

Variable name	COMMON block	Subroutine	Description and comments
DELT		INPLOT	tangential blade thickness, m
DELZ		MESHO PTBDRY THETOM OUTPUT	increment in z-coordinate, m
DENS		TVELCY	ρ_i' or ρ_o', kg/m^3
DENTOL		TSONIN	density tolerance (input for TSONIC, ref. 5)
DFDM(MM, MHTP1)	VARCOM		$-B \cos \beta \, [d(rV_\theta)/dm]$ at orthogonal mesh points (eq. (4)), m/sec
DFDS(MM)		BLDVEL	$\partial(rV_\theta)/\partial s$ at mesh points along horizontal mesh lines, m/sec
DFDT(MM, MHTP1)		BLDVEL	$\partial(rV_\theta)/\partial t$ at orthogonal mesh points, m/sec
DFVERT(MHTP1)		BLDVEL	$\partial(rV_\theta)/\partial t$ at mesh points along vertical mesh lines, m/sec
DIDR		NEWRHO	$\partial I/\partial r$, m/sec^2
DIDS(MM, MHTP1)		NEWRHO	$\partial I/\partial s$ at orthogonal mesh points, m/sec^2
DIDT(MHTP1)		NEWRHO	$\partial I/\partial t$ at mesh points along vertical mesh lines, m/sec^2
DIST(NPPP)		THIKOM TSONIN	distances on meridional plane along lines connecting input blade-section points (ZBL,RBL), m

Variable name	COMMON block	Subroutine	Description and comments
DISTLE		INDEV	distance along horizontal mesh line from leading edge of blade for which a blade shape correction is made for incidence, m
DISTTE		INDEV	distance along horizontal mesh line from trailing edge of blade for which a blade shape correction is made for deviation, m
DLAM		TVELCY	change in rV_θ between points on vertical mesh lines, m^2/sec
DLDU(MM, MHTP1)	VARCOM		gradients of rV_θ with respect to stream function, $d(rV_\theta)/du$, at orthogonal mesh points, m^2/sec (This array is only defined and used in regions outside of blade row.)
DMAX		COEF	maximum calculated value of $\partial(rV_\theta)/\partial r$ at any mesh point, m/sec
DMD2		PTBDRY	expansion distance on smaller range of a plot
DMIN		COEF	minimum calculated value of $\partial(rV_\theta)/\partial r$ at any mesh point, m/sec
DNEW	INPUTT		input damping factor on calculation of $\partial(rV_\theta)/\partial r$ within blade row from outer iteration to outer iteration

Variable name	COMMON block	Subroutine	Description and comments
DPDR		NEWRHO	$\partial p''/\partial r$, N/m^3
DPDS(MM, MHTP1)		NEWRHO	$\partial p''/\partial s$ at orthogonal mesh points, N/m^3
DPDT(MHTP1)		NEWRHO	$\partial p''/\partial t$ at mesh points along vertical mesh lines, N/m^3
DPREL		TVELCY	change in p'' between points on vertical mesh lines, N/m^2
DRBL		TSONIN	tangential blade thickness at intersection of a meridional streamline with blade leading or trailing edge, m
DRBTH		THIKOM	interpolated value of tangential blade thickness, m
DRTHBL(NPPP, NBLPL)		THIKOM TSONIN	tangential blade thickness at ZBL, RBL input points, m
DTDRLE		INDEV	$(\partial \theta/\partial r)_{le}$, rad/m
DTDROM		THETOM	$\partial \theta/\partial r$ on orthogonal mesh, rad/m
DTDRTE		INDEV	$(\partial \theta/\partial r)_{te}$, rad/m
DTDS(NPPP)		THIKOM TSONIN	$\partial \theta/\partial s$, rad/m
DTDSFL		INDEV	$(\partial \theta/\partial s)_{bf}$ at leading or trailing edge, rad/m

Variable name	COMMON block	Subroutine	Description and comments
DTDSLE(MHTP1)		INDEV	$\partial\theta/\partial s$ of mid-channel flow surface at points where horizontal mesh lines cross leading edge of blade, rad/m
DTDSTE(MHTP1)		INDEV	$\partial\theta/\partial s$ of mid-channel flow surface at points where horizontal mesh lines cross trailing edge of blade, rad/m
DTDTLE		INDEV	$(\partial\theta/\partial t)_{le}$, rad/m
DTDTTE		INDEV	$(\partial\theta/\partial t)_{te}$, rad/m
DTDZLE		INDEV	$(\partial\theta/\partial z)_{le}$, rad/m
DTDZOM		THETOM	$\partial\theta/\partial z$ on orthogonal mesh, rad/m
DTDZTE		INDEV	$(\partial\theta/\partial z)_{te}$, rad/m
DTHDM(NPPP)		INPLOT	$\partial\theta/\partial s'$ used to estimate blade angle to calculate tangential blade thickness from TNBL, rad/m
DTHDR(11,NBLPL)	INDCOM		$\partial\theta/\partial r$ on alternate blade mesh (fig. 26), rad/m
DTHDS(MM,MHTP1)	CALCON		$\partial\theta/\partial s$ at orthogonal mesh points, rad/m
DTHDSP(11,NBLPL)		THETOM	$\partial\theta/\partial s'$ on alternate blade mesh (fig. 26), rad/m
DTHDT(MM,MHTP1)	CALCON		$\partial\theta/\partial t$ at orthogonal mesh points, rad/m

Variable name	COMMON block	Subroutine	Description and comments
DTHDTP(11,NBLPL)		THETOM	$\partial\theta/\partial t'$ on alternate blade mesh (fig. 26), rad/m
DTHDZ(11,NBLPL)	INDCOM		$\partial\theta/\partial z$ on alternate blade mesh (fig. 26), rad/m
DTIP		TVELCY	change in T_i' between points on vertical mesh lines, K
DTPP		TVELCY	change in T'' between points on vertical mesh lines, K
DTVEL(MHTP1)		TVELCY	coefficients, d, of velocity-gradient equation ((A7), part I) at mesh points along vertical mesh lines
DUDS(MM)		NEWRHO	$\partial u/\partial s$ along horizontal mesh lines, 1/m
DUDT(MHTP1)		NEWRHO	$\partial u/\partial t$ at mesh points along vertical mesh lines, 1/m
DVDRT		COEF	updated estimate of $\partial(rV_\theta)/\partial r$ at a mesh point, m/sec
DVTHDR(MM,MHTP1)		COEF	$\partial(rV_\theta)/\partial r$ at orthogonal mesh points, m/sec
DWMDM(MM,MHTP1)		TVELCY	dW_m/dm at orthogonal mesh points, 1/sec
DWMDS(MM)		TVELCY	$\partial W_m/\partial s$ along horizontal mesh lines, 1/sec
DWMDT(MM,MHTP1)		TVELCY	$\partial W_m/\partial t$ at orthogonal mesh points, 1/sec
DWMVER(MHTP1)		TVELCY	$\partial W_m/\partial t$ along vertical mesh lines, 1/sec

Variable name	COMMON block	Subroutine	Description and comments
DWTDM(MM,MHTP1)		TVELCY	dW_θ/dm at orthogonal mesh points, 1/sec
DWTDS(MM)		TVELCY	$\partial W_\theta/\partial s$ along horizontal mesh lines, 1/sec
DWTDT(MM,MHTP1)		TVELCY	$\partial W_\theta/\partial t$ at orthogonal mesh points, 1/sec
DWTVER(MHTP1)		TVELCY	$\partial W_\theta/\partial t$ at mesh points along vertical mesh lines, 1/sec
DYDX(10)		INPUT	temporary storage for gradients
DYDX(MM or MHTP1)		PRECAL	temporary storage for derivative of several SPLINE and SPLINT calls
DYDX(NBLPL)		THETOM	temporary storage for derivative of several SPLINE calls
DYDX2(MM or MHTP1)		PRECAL	temporary storage for second derivatives calculated by SPLINE calls
DYDX2(NBLPL)		THETOM	temporary storage for second derivatives of several SPLINE calls
EM(NIN or NOUT)		TIPF RHOIPF RHOOPF LAMDAF RVTHTA	second derivatives of spline-fit curves

Variable name	COMMON block	Subroutine	Description and comments
EOP		INPLOT MEPLOT SLPLOT SVPLOT	end of plot indicator (EOP = 1.0)
ERROR		SOR	maximum absolute value of change in u at any point for an overrelaxation iteration
ERSOR(interger variable)		TSONIN	input for TSONIC (ref. 5)
ETVEL(MHTP1)		TVELCY	coefficients, e, of velocity-gradient equation ((A7), part I) at mesh points along vertical mesh lines
EXPON	CALCON		$1/(\gamma - 1)$
EXTRAP		INDEV	distance along horizontal mesh line from blade leading or trailing edge to first mesh point outside of blade, m
FCHANG		BLDVEL	maximum value of change in F_r at any mesh point between any two outer iterations
FLFR(NSL)	INPUTT		input values of stream function designating streamlines along which output is to be printed
FMAX		BLDVEL	maximum new predicted value of F_r at any mesh point during an outer iteration

Variable name	COMMON block	Subroutine	Description and comments
FMIN		BLDVEL	minimum new predicted value of F_r at any mesh point during an outer iteration
FNEW	INPUTT		input damping factor on calculation of F_r from outer iteration to outer iteration
FR(MM, MHTP1)	VARCOM		F_r at orthogonal mesh points (eq. (A4)), m/sec^2
FRAC		VBDRY	stream function, u, at mesh point on vertical boundary
FRT		BLDVEL	predicted value of F_r at a mesh point
FST(MM, MHTP1)		BLDVEL	rV_θ at orthogonal mesh points, m^2/sec
FTVEL(MHTP1)		TVELCY	coefficients, f, of velocity-gradient equation ((A7), part I) at mesh points along vertical mesh lines
FVERT(MHTP1)		BLDVEL	temporary storage for values of rV_θ from FST array on vertical mesh lines, m^2/sec
GAM	INPUTT		input, γ
GRAD(101)		INPLOT	dummy array for derivatives calculated by SPLINT calls in INPLOT
GRAD(MHTP1)		LOSSOM	dummy array of derivatives calculated in SPLINT call

Variable name	COMMON block	Subroutine	Description and comments
H1		COEF	h_1 (eq. (A8), fig. 24)
H2		COEF	h_2 (eq. (A8), fig. 24)
H3		COEF	h_3 (eq. (A8), fig. 24)
H4		COEF	h_4 (eq. (A8), fig. 24)
IDEBUG	INPUTT		integer input indicating multiple of outer iterations at which debug output is printed
IEND	Blank		integer indicator of stage of solution to which program has proceeded: IEND = -1, prior to convergence of subsonic solution IEND = 0, between convergence of subsonic solution and beginning of transonic solution IEND = 1, during first transonic solution with all velocities smaller than choking-mass-flow solution IEND = 2, during second transonic solution with all velocities greater than choking-mass-flow solution
IL		VBDRY	integer (1 or 3) to identify proper word in BDY array

Variable name	COMMON block	Subroutine	Description and comments
ILE(MHTP1)	CALCON		vertical mesh line numbers of first mesh point inside blade region at leading edge
ILS(NSL)	SLCOM		vertical mesh line number of first intersection of a streamline with a vertical mesh line inside blade region at leading edge
IMESH	INPUTT		integer input indicating the multiple of outer iterations at which major output is printed for orthogonal mesh
IND		VBDRY TVELCY	integer which indicates solution procedure in CONTIN
INTVL		TSONIN	input for TSONIC (ref. 5)
IPLOT	INPUTT		integer input indicating multiple of outer iterations at which major output is plotted on microfilm
IPLT		MEPLOT	indicates which of two plots is being made
ISLINE	INPUTT		integer input indicating multiple of outer iterations at which major output is printed along streamlines

Variable name	COMMON block	Subroutine	Description and comments
ISTATL	INPUTT		integer input indicating multiple of outer iterations at which major output is printed along station lines
ISUPER	INPUTT		integer input indicating whether only subsonic, or both subsonic and supersonic, solutions of velocity-gradient equation are to be calculated
ITE(MHTP1)	CALCON		vertical mesh line numbers of last mesh point inside blade region at trailing edge
ITER	Blank		outer iteration counter, incremented by 1 at beginning of each outer iteration
ITS(NSL)	SLCOM		vertical mesh line number of last intersection of a streamline with a vertical mesh line inside blade region at trailing edge
ITSON	INPUTT		integer input indicating multiple of outer iterations at which information is printed as input for TSONIC program (ref. 5)
JZ		VBDRY TVELCY	integer used to indicate to CONTIN that subsonic (JZ = 1) or supersonic (JZ = 2) solution is desired

Variable name	COMMON block	Subroutine	Description and comments
K(MM, MHTP1) (real variable)	VARCOM		k_o (eq. (A9)) at orthogonal mesh points
KNEW(real variable)		COEF	updated value of k_o (eq. (A9)) at a mesh point
LAMBDA(MHTP1) (real variable)		VBDRY TVELCY	λ for mesh points along vertical mesh lines, m^2/sec
LAMBDO(MHTP1) (real variable)		TVELCY	$\left(rV_\theta\right)_o$, for mesh points along vertical mesh lines, m^2/sec
LAMDAI(real variable)		PRECAL LOSSOM	λ, m^2/sec
LAMIN(NIN) (real variable)	INPUTT		input values of λ at points along line from hub to shroud on which upstream flow conditions are given, m^2/sec
LAMOUT(NOUT) (real variable)	INPUTT		input values of $\left(rV_\theta\right)_o$ at points along line from hub to shroud on which downstream flow conditions are given, m^2/sec
LAMVT	INPUTT		input integer (0 or 1) indicating whether upstream and downstream whirl (0) or tangential velocity (1) is given as input
LMAX(real variable)		SOR	maximum value of RATIO over all mesh points

Variable name	COMMON block	Subroutine	Description and comments
LMIN(real variable)		SOR	minimum value of RATIO over all mesh points
LOC		VBDRY	integer (1 or MM) indicating vertical mesh line number for which VBDRY is called
LOSOUT(NOUT) (real variable)	INPUTT		input fraction of absolute total pressure loss, at points along line from hub to shroud on which downstream flow conditions are given
LRNG(real variable)		INPLOT SVPLOT	left-most point of range of a plot
LSFR	INPUTT		input integer (0 or 1) indicating whether upstream and downstream flow conditions are input as a function of stream function (0) or radius (1)
LTPL	INPUTT		input integer (0 or 1) indicating whether downstream total pressure (0) or fractional loss of stagnation pressure (1) is given in input
MARK	CROSCM		marker in MESHO and CROSCD to indicate if CROSCD is being called for the first time

Variable name	COMMON block	Subroutine	Description and comments
MARK(NOSTAT)		OUTPUT	integers between 1 and 4 indicating whether output station lines are outside blade, within blade, or on leading or trailing edge
MBI	INPUTT		input number of vertical mesh lines from left boundary of orthogonal mesh (ZOMIN) to first point of mesh-size change (ZOMBI)
MBL(NPPP, NBLPL) (real variable)		INPLOT	s'-coordinate, corresponding to ZBL and RBL, used for plotting input blade sections, m
MBLD		SVPLOT	number of suction-surface or pressure-surface velocities on a plot
MBO	INPUTT		input total number of vertical mesh lines from left boundary of orthogonal mesh (ZOMIN) to point of second mesh-size change (ZOMBO)
MCT		OUTPUT	integer (1 or 2) indicating whether a compressor (1) or a turbine (2) is being analyzed
MHT	INPUTT		input total number of horizontal mesh spaces from hub to shroud of orthogonal mesh; maximum of 100

Variable name	COMMON block	Subroutine	Description and comments
MHTP1	CALCON		MHT + 1
MLESP(5) (real variable)		TSONIN	m-coordinates of blade surface spline points near leading edge of a blade section, printed to make a layout to obtain input for TSONIC (ref. 5), m
MM	INPUTT		input total number of vertical mesh lines from left to right boundaries of orthogonal mesh (ZOMIN to ZOMOUT), maximum of 100
MMM1	CALCON		MM - 1
MR(MM)(real variable)		TSONIN	m-coordinates of points defining a stream channel, printed as input for TSONIC (ref. 5), m
MSFL(real variable)	INPUTT		input total mass flow through entire circumferential annulus of machine, kg/sec
MSL(MM,NSL) (real variable)	SLCOM		m-coordinates of points along streamlines where they cross vertical mesh lines, m (Origin of m-coordinate along a streamline corresponds to point where z = 0 along streamline.)

Variable name	COMMON block	Subroutine	Description and comments
MSP(MM)(real variable)		TSONIN	m-coordinates of blade surface spline points, given as input for TSONIC (ref. 5), m
MST(NSL,NOSTAT), (real variable)	STACOM		m-coordinates of points where station lines cross streamlines, m (Origin of m-coordinates along a streamline corresponds to point where z = 0 along streamline.)
MTEM(NOSTAT) (real variable)		OUTPUT	temporary storage for values of m-coordinate for MST array, m
MTESP(5) (real variable)		TSONIN	m-coordinates of blade surface spline points near trailing edge of a blade section, printed to make a layout to obtain input for TSONIC (ref. 5), m
NBL	INPUTT		input number of blades in blade row
NBLPL	INPUTT		number of input blade planes or blade sections on which data (ZBL, RBL, THBL, TNBL) are given to describe mean flow surface and blade thickness, maximum of 50
NBLPTS		TSONIN	number of spline points on suction or pressure surface of a blade section

Variable name	COMMON block	Subroutine	Description and comments
NCOUNT		VBDRY TVELCY	total number of iterations or attempts at satisfying velocity-gradient equation
NHUB	INPUTT		number of input data points in ZHUB and RHUB arrays, maximum of 50
NIN	INPUTT		number of input data points in upstream arrays of flow properties (SFIN, RADIN, TIP, PRIP, LAMIN, VTHIN), maximum of 50
NOSTAT	INPUTT		input number of hub-shroud stations (located by coordinates in ZHST and ZTST) at which output is desired, maximum of 50
NOUT	INPUTT		number of input data points in downstream arrays of flow properties (SFOUT, RADOUT, PROP, LOSOUT, LAMOUT, VTHOUT), maximum of 50
NPPP	INPUTT		number of input data points per blade section or blade plane in ZBL, RBL, THBL, and TNBL arrays, maximum of 50
NREAD	Blank		integer number of input tape-reading unit of computer which is running MERIDL

Variable name	COMMON block	Subroutine	Description and comments
NRSP		TSONIN	input for TSONIC (ref. 5)
NSL	INPUTT		input number of streamlines from hub to shroud (designated by values in FLFR) at which output is desired, maximum of 50
NTIP	INPUTT		number of input data points in ZTIP and RTIP arrays, maximum of 50
NWRIT	Blank		integer number of output tape-writing unit of computer which is running MERIDL
OMEGA	INPUTT		input rotational speed, ω, rad/sec
ORF		SOR	overrelaxation factor
ORF		TSONIN	overrelaxation factor (input for TSONIC, ref. 5)
ORFMAX		SOR	current estimate for maximum value of ORF calculated using LMAX
ORFMIN		SOR	current estimate for minimum value of ORF calculated using LMIN
PHI		OUTPUT	φ, deg
PITCH	CALCON		2π/NBL, rad
PLOSS(MM,MHTP1)	CALCON		fractional loss of relative total pressure at orthogonal mesh points

Variable name	COMMON block	Subroutine	Description and comments
PLOSSL		TSONIN	fractional loss of relative total pressure at a ZSL,RSL point along a streamline
PLOSTE		INDEV	fractional loss of relative total pressure at blade trailing edge
PLTX(101)		INPLOT	temporary storage of x-plot coordinates of many arrays in INPLOT
PLTY(101)		INPLOT	temporary storage of y-plot coordinates of many arrays in INPLOT
PPTHBL(NPPP,NBLPL)		INPLOT	values of $r\theta$ for pressure surface of adjacent blade, used for plotting input blade sections, m
PRATIO(MHTP1)		LOSSOM	$p_o'' / (p_o'')_{ideal}$ for each horizontal mesh line downstream of blade
PREL(MM or MHTP1)		NEWRHO TVELCY	p'' at mesh points along horizontal or vertical mesh lines, N/m^2
PRELN		TVELCY	new p'', N/m^2
PRINP		PRECAL LOSSOM	p_i', N/m^2
PRIP(NIN)	INPUTT		input, p_i', at points along line from hub to shroud on which upstream flow conditions are given, N/m^2

Variable name	COMMON block	Subroutine	Description and comments
PROP(NOUT)	INPUTT		input, p'_o, at points along line from hub to shroud on which downstream flow conditions are given, N/m^2
PTHBL(NPPP,NBLPL)		INPLOT	values of $r\theta$ for pressure surface, used for plotting input blade sections, m
R1		MESHO	r-coordinate of intersection of line ①, fig. 8, with upper horizontal mesh line, m
R2		MESHO	r-coordinate of intersection of line ②, fig. 8, with upper horizontal mesh line, m
RADIN(NIN)	INPUTT		input r-coordinates of points along line from hub to shroud on which upstream flow conditions are given, m
RADOUT(NOUT)	INPUTT		input r-coordinates of points along line from hub to shroud on which downstream flow conditions are given, m
RATIO		SOR	u_i^{m+1}/u_i^m for use in eqs. (B2) and (B3) of ref. 11
RBL(NPPP,NBLPL)	INPUTT		input array of r-coordinates, corresponding to ZBL, of points describing mean blade surface, m

Variable name	COMMON block	Subroutine	Description and comments
RBLTEM(NBLPL)		OUTPUT	temporary storage for values in RLE and RTE arrays, m
RBRNG	PLTCOM		r-coordinate of bottom boundary of a plot of meridional plane or orthogonal mesh, done in MEPLOT, m
RCARB(MHTP1)		TVELCY	$\rho \cos(\alpha - \varphi) rB$ along a vertical mesh line (eq. (5)), kg/m^2
RCURV		CROSCD	r-coordinate of horizontal mesh line at input z-coordinate, m
REDFAC	INPUTT		input factor used to reduce mass flow (MSFL) in order to assure subsonic flow throughout flow passage
REFR	CROSCM		reference r-coordinate in MESHO from which orthogonal mesh is extended by addition of another "link," m
REFSL	CROSCM		reference slope in MESHO of vertical link being extended from a known orthogonal mesh point to a new mesh point
REFZ	CROSCM		reference z-coordinate in MESHO from which orthogonal mesh is extended by addition of another "link," m

Variable name	COMMON block	Subroutine	Description and comments
RELER		NEWRHO	maximum relative change in W at any mesh point between two outer iterations
RELTOP(NOUT)		PRECAL LOSSOM	p_i''/p_o'' at hub-shroud input points downstream of blade row
REPEAT		VBDRY TVELCY	logical variable indicating that velocity-gradient solution should be repeated with new values of TIPBDY, RHOIP, and LAMBDA
RHIN	CALCON		r-coordinate of intersection with hub profile of line on which upstream flow conditions are given, m
RHO(MM, MHTP1)	VARCOM		ρ, at orthogonal mesh points, kg/m^3
RHOAV(MM, MHTP1)	VARCOM		average density across flow channel from suction surface to pressure surface, at orthogonal mesh points, kg/m^3
RHOIP(MHTP1)		VBDRY TVELCY	ρ_i' for mesh points along vertical mesh lines, kg/m^3
RHOIP(NIN)		RHOIPF	ρ_i' at input points of upstream flow conditions, kg/m^3
RHOIP		INIT	ρ_i' at hub, kg/m^3

Variable name	COMMON block	Subroutine	Description and comments
RHOIP		TSONIN	ρ_i' for a streamline, kg/m^3, input for TSONIC (ref. 5)
RHOL		BLDVEL	ρ_l, kg/m^3
RHOOP(MHTP1)		TVELCY	ρ_o' for mesh points along vertical mesh lines, kg/m^3
RHOOP(NOUT)		RHOOPF	ρ_i' at input points of downstream flow conditions, kg/m^3
RHOSL		TSONIN	ρ at a ZSL, RSL point along a streamline, kg/m^3
RHOT		BLDVEL	ρ_{tr}, kg/m^3
RHOUT	CALCON		r-coordinate of intersection with hub profile of line on which downstream flow conditions are given, m
RHOW		VBDRY	ρW, $kg/(m^2)(sec)$
RHPLT(100)	PLTCOM		r-coordinates used for plotting hub profile, RHUB, in MEPLOT, m
RHUB(NHUB)	INPUTT		input r-coordinates of points defining hub or bottom boundary of flow channel, m
RILOM(MHTP1)		LAMDAF	radii for spline fit of stream function against radius
RLE(NBLPL)	CALCON		r-coordinates of input blade section points (from RBL) defining leading edge of blade, m

Variable name	COMMON block	Subroutine	Description and comments
RLEH	CALCON		r-coordinate of intersection of leading edge of blade with hub profile, m
RLEOM(MHTP1)	CALCON		r-coordinates of intersections of horizontal mesh lines with blade leading edge, m
RLEP		TSONIN	r-coordinate of point near leading edge of blade section along meridional streamline, m
RLESL		TSONIN	r-coordinate of intersection of a streamline with leading edge of blade, m
RLET	CALCON		r-coordinate of intersection of leading edge of blade with shroud profile, m
RLINE		CROSCD	r-coordinate on straight-line vertical orthogonal link at input z-coordinate, m
RLPLT(100)	PLTCOM		r-coordinates used for plotting blade leading edge, RLE, in MEPLOT, m
RMEAN		VBDRY	mean radius $(r_{hub} + r_{tip})/2$, m
RMR		MESHO CROSCD	RCURV - RLINE in CROSCD, m
RMSP(MM)		TSONIN	r-coordinates of points defining a stream channel, printed as input for TSONIC (ref. 5), m

Variable name	COMMON block	Subroutine	Description and comments
ROLOM(MHTP1)		RVTHTA	radii for spline fit of stream function against radius
ROM(MM, MHTP1)	CALCON		r-coordinates of orthogonal mesh, m
ROTI(MM or MHTP1)		NEWRHO	rothalpy, I, at mesh points along horizontal or vertical mesh lines, m^2/sec^2
RPC(11, NBLPL)	INDCOM		r-coordinates of alternate mesh (fig. 26)
RRAD(NHUB, MHTP1)		MESHO	r-coordinates of points along radial lines from input points on hub profile to shroud profile, m
RRNG		INPLOT SVPLOT	right-most point of range of a plot
RRTHBL(NPPP, NBLPL)		INPLOT	values of $r\theta$ coordinate of blade mean camber line of adjacent blade, used for plotting input blade sections, m
RSL(MM, NSL)	SLCOM		array of r-coordinates of points along streamlines where they cross vertical mesh lines, m
RSLTEM(NSL)		OUTPUT	temporary storage for calculated values to be put into RSL array, m
RSPLT(100)	PLTCOM		r-coordinates used for plotting shroud profile, RTIP, in MEPLOT, m

Variable name	COMMON block	Subroutine	Description and comments
RST(NSL,NOSTAT)	STACOM		r-coordinates of points along station lines where they cross streamlines, m
RTE(NBLPL)	CALCON		r-coordinates of input blade section points (from RBL) defining trailing edge of blade, m
RTEH	CALCON		r-coordinate of intersection of trailing edge of blade with hub profile, m
RTEM(10)		INPUT	temporary storage for r-coordinates, m
RTEM(MM)		MEPLOT	temporary storage of r-coordinates from ROM for plotting, m
RTEM(NBLPL)		THETOM	temporary storage of r-coordinates from RPC, m
RTEM(MHTP1 or NOSTAT or 20)		OUTPUT	temporary storage for values from ROM array on vertical mesh lines; also temporary storage for values from RST array along station lines, m
RTEOM(MHTP1)	CALCON		r-coordinates of intersections of horizontal mesh lines with blade trailing edge, m
RTEP		TSONIN	r-coordinate of point near trailing edge of blade section along meridional streamline, m

Variable name	COMMON block	Subroutine	Description and comments
RTESL		TSONIN	r-coordinate of intersection of a streamline with trailing edge of blade, m
RTET	CALCON		r-coordinate of intersection of trailing edge of blade with shroud profile, m
RTHBL(NPPP,NBLPL)		INPLOT	$r\theta$-coordinate of blade mean camber line, corresponding to points in ZBL,RBL, used for plotting input blade sections, m
RTIN	CALCON		calculated r-coordinate of intersection with shroud profile of line on which upstream flow conditions are given, m
RTIP(NTIP)	INPUTT		input r-coordinates of points defining shroud or top boundary of flow channel, m
RTLEP1(5)		TSONIN	$r\theta$ at blade suction-surface spline points near leading edge of a blade section, referenced to zero at leading edge, used to make a layout to obtain input for TSONIC (ref. 5), m

Variable name	COMMON block	Subroutine	Description and comments
RTLEP2(5)		TSONIN	$r\theta$ at blade pressure-surface spline points near leading edge of a blade section, referenced to zero at leading edge, used to make a layout to obtain input for TSONIC (ref. 5), m
RTMP(NHUB)	CROSCM		temporary storage for portions of RRAD array in MESHO and CROSCD, m
RTOLER		VBDRY TVELCY	tolerance on relative error of subsequent calculated values of integrated mass flow
RTOUT	CALCON		calculated r-coordinate of intersection with shroud profile of line on which downstream flow conditions are given, m
RTPLT(100)	PLTCOM		r-coordinate used for plotting blade trailing edge, RTE, in MEPLOT, m
RTRNG	PLTCOM		r-coordinate of top boundary of a plot of meridional plane or orthogonal mesh, done in MEPLOT, m

Variable name	COMMON block	Subroutine	Description and comments
RTTEP1(5)		TSONIN	$r\theta$ at blade suction-surface spline points near trailing edge of a blade section, referenced to zero at trailing edge, used to make a layout to obtain input for TSONIC (ref. 5), m
RTTEP2(5)		TSONIN	$r\theta$ at blade pressure-surface spline points near trailing edge of a blade section, referenced to zero at trailing edge, used to make a layout to obtain input for TSONIC (ref. 5), m
RVA		VBDRY	$\rho_j W_j \cos \beta_j \, 2\pi r_j/NBL$, kg/(m)(sec)
RVA		TVELCY	$\rho_j W_j \cos \beta_j \cos(\alpha - \varphi)_j r_j B_j$, kg/(m)(sec)
RVAS		VBDRY	$\rho_{j+1} W_{j+1} \cos \beta_{j+1} \, 2\pi r_{j+1}/$ NBL, kg/(m)(sec)
RVAS		TVELCY	$\rho_{j+1} W_{j+1} \cos \beta_{j+1} \cos(\alpha - \varphi)_{j+1} r_{j+1} B_{j+1}$ kg/(m)(sec)
SAL		TVELCY	$\sin \alpha$
SAMP(MM,MHTP1)	VARCOM		$\sin(\alpha - \varphi)$ at orthogonal mesh points
SAMPLE		INDEV	$\sin(\alpha - \varphi)_{le}$
SAMPTE		INDEV	$\sin(\alpha - \varphi)_{te}$

Variable name	COMMON block	Subroutine	Description and comments
SBETA		TVELCY	sin β
SDIST		INDEV	DISTLE (or DISTTE) plus the distance from blade leading (or trailing) edge out to first adjacent mesh point, m
SDRIV(NHUB)	CROSCM		second derivatives of a spline fit of horizontal rows in RRAD array in MESHO, 1/m
SF(MHTP1)		LOSSOM	estimate of stream function at mesh points on downstream boundary of orthogonal mesh
SFIN(NIN)	INPUTT		input values of stream function along hub-shroud line on which upstream flow conditions are given
SFOUT(NOUT)	INPUTT		input values of stream function along hub-shroud line on which downstream flow conditions are given
SINBTA		VBDRY	sin β
SL1		MESHO CROSCD	slope of horizontal orthogonal at its point of intersection by a radial line in CROSCD
SLCRD(integer variable)		TSONIN	input for TSONIC (ref. 5)
SLEOM(MHTP1)	CALCON		s-coordinates of intersections of horizontal mesh lines with blade leading edge, m

Variable name	COMMON block	Subroutine	Description and comments
SLIDLE		INDEV	solidity at leading edge of blade where it is intersected by a horizontal mesh line
SLIDTE		INDEV	solidity at trailing edge of blade where it is intersected by a horizontal mesh line
SLOM(MM)		MESHO	slopes of horizontal mesh lines at mesh points
SLOPE(NIN or NOUT)		TIPF RHOIPF RHOOPF LAMDAF RVTHTA	derivatives of spline-fit curves
SOM(MM, MHTP1)	CALCON		s-coordinates of orthogonal mesh, m
SPHI(MM, MHTP1)	CALCON		sin φ at orthogonal mesh points
SPHILE		INDEV	sin φ_{le}
SPHITE		INDEV	sin φ_{te}
SPLNO1		TSONIN	input for TSONIC (ref. 5)
SPLNO2		TSONIN	input for TSONIC (ref. 5)
SRE(integer variable)	Blank		switch used to turn on and off printing of error or warning messages in some subroutines

Variable name	COMMON block	Subroutine	Description and comments
SRW(integer variable)	Blank		switch used to turn on and off printing of debug information in some subroutines
SSTHBL(NPPP,NBLPL)		INPLOT	values of $r\theta$ for suction surface of adjacent blade, used for plotting input blade sections, m
STEOM(MHTP1)	CALCON		s-coordinates of intersections of horizontal mesh lines with blade trailing edge, m
STHBL(NPPP,NBLPL)		INPLOT	values of $r\theta$ on suction surface, used for plotting input blade sections, m
STRFN(integer variable)		TSONIN	input for TSONIC (ref. 5)
SURVL(integer variable)		TSONIN	input for TSONIC (ref. 5)
SYM		INPLOT MEPLOT SLPLOT SVPLOT	indicator used to select a special plot symbol from a table
SYN		MEPLOT	indicator used to select a special plot symbol from a table
SZRBL(NPPP)		THETOM	arc length along input blade section in meridional plane, m
SZRPC(11 or NBLPL)		THETOM	arc length along vertical or horizontal lines of alternate mesh (fig. 26)

Variable name	COMMON block	Subroutine	Description and comments
TANBBL		INDEV	tan β_b, tangent of blade mean camber line angle at leading or trailing edge
TANBFL		INDEV	tan β_{bf}, tangent of flow angle at leading or trailing edge, corrected for blockage
TEMPER		TVELCY	T_i' or T_o', K
TGROG	CALCON		$2\gamma R/(\gamma + 1)$
THBL(NPPP,NBLPL)	INPUTT		input array of θ-coordinates, corresponding to RBL, ZBL, of points describing mean blade surface, rad
THLEOM(MHTP1)	CALCON		θ-coordinates of intersections of horizontal mesh lines with blade leading edge, rad
THLESL		TSONIN	θ-coordinate of intersection of streamline with blade leading edge, rad
THPC(11,NBLPL)		THETOM	θ-coordinates of points on alternate mesh (fig. 26), rad
THSL		TSONIN	θ-coordinate (relative to MERIDL origin, not TSONIC origin) of mean blade surface at points along meridional streamlines, rad

Variable name	COMMON block	Subroutine	Description and comments
THSP1(MM)		TSONIN	θ-coordinates of blade suction-surface spline points, given as input for TSONIC (ref. 5), rad
THSP2(MM)		TSONIN	θ-coordinates of blade pressure-surface spline points given as input to TSONIC, rad
THETOM(MHTP1)	CALCON		θ-coordinates of intersections of horizontal mesh lines with blade trailing edge, rad
THTESL		TSONIN	θ-coordinate of intersection of meridional streamline with blade trailing edge, rad
TINP		PRECAL LOSSOM	T_i', K
TIP(NIN)	INPUTT		input T_i' at points along the line from hub to shroud on which upstream flow conditions are given, K
TIPBDY(MHTP1)		VBDRY	T_i' at points on upstream or downstream boundary of orthogonal mesh, K
TIPT(MHTP1)		TVELCY	T_i' at points along vertical mesh lines, K
TIPTEM		TSONIN	T_i' along a streamline, K, input TIP for TSONIC (ref. 5)

Variable name	COMMON block	Subroutine	Description and comments
TITLEI(20)	INTITL		alphanumerical contents of input title card
TITL1(9)		INPLOT	plot title INLET ABSOLUTE TOTAL TEMPERATURE
TITL1(15)		MEPLOT	plot title HUB, SHROUD, AND BLADE BOUNDARIES IN MERIDIONAL PLANE
TITL1(10)		SLPLOT	plot title STREAMLINE PLOT IN MERIDIONAL PLANE
TITL1(12)		SVPLOT	plot title MERIDIONAL AND SURFACE RELATIVE VELOCITIES
TITL2(8)		INPLOT	plot title INLET ABSOLUTE TOTAL PRESSURE
TITL2(10)		MEPLOT	plot title ORTHOGONAL MESH IN MERIDIONAL PLANE
TITL2(3)		SLPLOT	plot title Z DIRECTION
TITL2(9)		SVPLOT	plot title STREAMLINE NO. XXXX, U = XXXXXXXX
TITL3(5)		INPLOT	plot title INLET ABSOLUTE WHIRL
TITL3(3)		MEPLOT	plot title Z DIRECTION
TITL3(3)		SLPLOT	plot title R DIRECTION
TITL3(14)		SVPLOT	plot title MERIDIONAL RELATIVE VELOCITIES FOR ALL STREAMLINES

Variable name	COMMON block	Subroutine	Description and comments
TITL4(9)		INPLOT	plot title INLET ABSOLUTE TANGENTIAL VELOCITY
TITL4(3)		MEPLOT	plot title R DIRECTION
TITL4(11)		SLPLOT	plot title SUBSONIC SOLUTION, ITERATION NO. XXXX
TITL4(15)		SVPLOT	plot title SUCTION SURFACE RELATIVE VELOCITIES FOR ALL STREAMLINES
TITL5(8)		INPLOT	plot title OUTLET ABSOLUTE TOTAL PRESSURE
TITL5(5)		SLPLOT	plot title TRANSONIC SOLUTION
TITL5(16)		SVPLOT	plot title PRESSURE SURFACE RELATIVE VELOCITIES FOR ALL STREAMLINES
TITL6(9)		INPLOT	plot title OUTLET ABSOLUTE TOTAL PRESSURE LOSS
TITL6(6)		SVPLOT	plot title MERIDIONAL COORDINATE
TITL7(6)		INPLOT	plot title OUTLET ABSOLUTE WHIRL
TITL7(2)		SVPLOT	plot title VELOCITY
TITL8(9)		INPLOT	plot title OUTLET ABSOLUTE TANGENTIAL VELOCITY

Variable name	COMMON block	Subroutine	Description and comments
TITL10(13)		INPLOT	plot title INPUT BLADE SECTIONS FROM ZBL, RBL, THBL, TNBL
TITL11(6)		INPLOT	plot title BLADE SECTION NO. XXXX
TITL12(6)		INPLOT	plot title COMBINED BLADE SECTIONS
TITL13(4)		INPLOT	plot title STREAM FUNCTION
TITL14(2)		INPLOT	plot title RADIUS
TITL15(5)		INPLOT	plot title INPUT ARRAY - TIP
TITL16(5)		INPLOT	plot title INPUT ARRAY - PRIP
TITL17(5)		INPLOT	plot title INPUT ARRAY - LAMIN
TITL18(5)		INPLOT	plot title INPUT ARRAY - VTHIN
TITL19(5)		INPLOT	plot title INPUT ARRAY - PROP
TITL20(5)		INPLOT	plot title INPUT ARRAY - LOSOUT
TITL21(5)		INPLOT	plot title INPUT ARRAY - LAMOUT
TITL22(5)		INPLOT	plot title INPUT ARRAY - VTHOUT
TITL24(10)		INPLOT	plot title BLADE SECTION MERIDIONAL COORDINATE

Variable name	COMMON block	Subroutine	Description and comments
TITL25(9)		INPLOT	plot title TANGENTIAL COORDINATE - RADIUS*THETA
TLEREF		TSONIN	θ-coordinate of leading edge of blade, relative to TSONIC origin (ref. 5), rad
TNBL(NPPP,NBLPL)	INPUTT		input array of blade normal thicknesses, corresponding to ZBL,RBL coordinates, m
TOLER		MESHO	tolerance used in ROOT calls
TOLER		TIPF RHOIPF RHOOPF LAMDAF RVTHTA	tolerance for a point close to a spline point
TOM(MM,MHTP1)	CALCON		t-coordinates of orthogonal mesh, m
TOP(MHTP1)		TVELCY	T'_o at points along vertical mesh lines, K
TOP		PRECAL LOSSOM	T'_o, K
TPP		NEWRHO TVELCY OUTPUT	T'' at a mesh point, K
TPPN		TVELCY	new T'', K
TPPTIP(MM or MHTP1)		NEWRHO	T''/T'_i along vertical or horizontal mesh lines

Variable name	COMMON block	Subroutine	Description and comments
TRNG		INPLOT SVPLOT	top or upper range of values on a given plot
TTEM(NBLPL)		PRECAL THETOM	temporary storage of θ-coordinates, rad
TTEREF		TSONIN	θ-coordinate of trailing edge of blade, relative to TSONIC origin (ref. 5), rad
TTIP		VBDRY NEWRHO BLDVEL TVELCY	T/T_i'
TVERT(MHTP1)		NEWRHO BLDVEL OUTPUT TVELCY	temporary storage for values from TOM array on a vertical mesh line, m
TWLMR(MHTP1)		TVELCY	$2\omega\lambda - (\omega r)^2$ at points along vertical mesh lines, m^2/sec^2
TWLMR		VBDRY NEWRHO BLDVEL	$2\omega\lambda - (\omega r)^2$, m^2/sec^2
UBDEV		INDEV	deviation angle, neglecting blockage correction, where horizontal orthogonal intersects blade, $\left(\beta_{fs} - \beta_b\right)_{te}$, deg
UBINC		INDEV	incidence angle, neglecting blockage correction, where horizontal orthogonal intersects blade, $\left(\beta_{fs} - \beta_b\right)_{le}$, deg

Variable name	COMMON block	Subroutine	Description and comments
UILOM(MHTP1)		LAMDAF	stream-function values for spline fit of stream function against radius
UNEW		SOR	new estimate for u at a mesh point
UOLOM(MHTP1)		RVTHTA	stream-function values for spline fit of stream function against radius
UOM(MM, MHTP1)	VARCOM		stream function, u, at orthogonal mesh points
UTEM(MHTP1 or 20)		OUTPUT	temporary storage for values from UOM array on vertical mesh lines; also stream function at 20 equally spaced points from hub to shroud
UVERT(MHTP1, 2)		SOR	temporary storage for boundary values of stream function on upstream and downstream boundaries of orthogonal mesh
UVERT(MHTP1)		NEWRHO	temporary storage for values from UOM array along vertical mesh lines
VELTOL	INPUTT		input convergence tolerance on maximum velocity change in each outer iteration, over all mesh points, for reduced mass flow
VTH(MM, MHTP1)	VARCOM		V_θ at orthogonal mesh points, m/sec

Variable name	COMMON block	Subroutine	Description and comments
VTHIN(NIN)	INPUTT		input values of $(V_\theta)_i$ at points along line from hub to shroud on which upstream flow conditions are given, m/sec
VTHOUT(NOUT)	INPUTT		input values of $(V_\theta)_o$ at points along line from hub to shroud on which downstream flow conditions are given, m/sec
W(MM, MHTP1)	VARCOM		W at orthogonal mesh points, m/sec
WAS		VBDRY TVELCY	first estimate of W_{j+1} at next mesh point along vertical mesh line (eq. (1)), W^*_{j+1}, m/sec
WASS		VBDRY TVELCY	second estimate of W_{j+1} at next mesh point along vertical mesh line (eq. (1)), W^{**}_{j+1}, m/sec
WBDRY(MHTP1)		VBDRY	W on upstream or downstream boundary of orthogonal mesh, calculated by velocity-gradient equation ((C9), part I), m/sec
WFLF		VBDRY	$\left(r^2 - r^2_{hub}\right)/\left(r^2_{tip} - r^2_{hub}\right)$, area fraction, estimate of stream function at radius, r
WHIRL		TVELCY	λ or $\left(rV_\theta\right)_o$, m^2/sec

Variable name	COMMON block	Subroutine	Description and comments
WHUB		VBDRY TVELCY	estimate of W_{hub}, m/sec
WLSSL(MM,NSL)	SLCOM		W_l at points along streamlines where they cross vertical mesh lines, m/sec
WLSST(NSL,NOSTAT)	STACOM		W_l at points along station lines where they cross streamlines, m/sec
WLSURF(MM,MHTP1)	VARCOM		W_l on orthogonal mesh, m/sec
WMSL(MM,NSL)	SLCOM		W_m at points where streamlines cross vertical mesh lines, m/sec
WMST(NSL,NOSTAT)	STACOM		W_m at points where station lines cross streamlines, m/sec
WMVERT(MHTP1)		TVELCY	temporary storage for values from WSUBM array on vertical mesh lines, m/sec
WRSL(MM,NSL)	SLCOM		W_r at points where streamlines cross vertical mesh lines, m/sec
WRST(NSL,NOSTAT)	STACOM		W_r at points where station lines cross streamlines, m/sec
WSL(MM,NSL)	SLCOM		W at points where streamlines cross vertical mesh lines, m/sec
WSM		VBDRY	W_s at mean radius, m/sec

Variable name	COMMON block	Subroutine	Description and comments
WSQ		VBDRY NEWRHO TVELCY	W^2, m^2/sec^2
WSQ		BLDVEL	W_l^2 or W_{tr}^2, m^2/sec^2
WST(NSL,NOSTAT)	STACOM		W at points where station lines cross streamlines, m/sec
WSUBM(MM,MHTP1)	VARCOM		W_m at orthogonal mesh points, m/sec
WSUBR(MM,MHTP1)	VARCOM		W_r at orthogonal mesh points, m/sec
WSUBS(MM,MHTP1)	VARCOM		W_s at orthogonal mesh points, m/sec
WSUBT(MM,MHTP1)	VARCOM		W_t at orthogonal mesh points, m/sec
WSUBZ(MM,MHTP1)	VARCOM		W_z at orthogonal mesh points, m/sec
WTEMP		NEWRHO	new calculated value of W at a mesh point, m/sec
WTFL		TSONIN	mass flow per blade in a stream sheet, kg/sec, input to TSONIC (ref. 5)
WTH(MM,MHTP1)	VARCOM		W_θ at orthogonal mesh points, m/sec
WTHETA		VBDRY TVELCY	W_θ, m/sec
WTHSL(MM,NSL)	SLCOM		W_θ at points where streamlines cross vertical mesh lines, m/sec

Variable name	COMMON block	Subroutine	Description and comments
WTHST(NSL,NOSTAT)	STACOM		W_θ at points where station lines cross streamlines, m/sec
WTSSL(MM,NSL)	SLCOM		W_{tr} at points where streamlines cross vertical mesh lines, m/sec
WTSST(NSL,NOSTAT)	STACOM		W_{tr} at points where station lines cross streamlines, m/sec
WTSURF(MM,MHTP1)	VARCOM		W_{tr} on orthogonal mesh, m/sec
WTVERT(MHTP1)		TVELCY	WTH on vertical mesh lines, m/sec
WWCR(MM,MHTP1)	VARCOM		W/W_{cr} at orthogonal mesh points
WWCRSL(MM,NSL)	SLCOM		W/W_{cr} at points where streamlines cross vertical mesh lines
WWCRST(NSL,NOSTAT)	STACOM		W/W_{cr} at points where station lines cross streamlines
WZSL(MM,NSL)	SLCOM		W_z at points where streamlines cross vertical mesh lines, m/sec
WZST(NSL,NOSTAT)	STACOM		W_z at points where station lines cross streamlines, m/sec
XIOM(MM,MHTP1)	VARCOM		ξ at orthogonal mesh points, (eq. (A2)), 1/m

Variable name	COMMON block	Subroutine	Description and comments
XIOMT		NEWRHO	new estimated value of ξ at a mesh point
XNEW		NEWRHO	percentage of new calculated value of XIOMT used in updating XIOM
Z		CROSCD	reference z-coordinate, m
Z1		MESHO	z-coordinate of intersection of line ①, fig. 8, with upper horizontal mesh line, m
Z2		MESHO	z-coordinate of intersection of line ②, fig. 8, with upper horizontal mesh line, m
ZBL(NPPP,NBLPL)	INPUTT		input array of z-coordinates of points describing blade surface, m
ZBLTEM(NBLPL)		OUTPUT	temporary storage for values in ZLE and ZTE arrays, m
ZEROM		OUTPUT	translation distance on m-coordinate so that $m = 0$ corresponds to $z = 0$, m
ZETOM(MM,MHTP1)	VARCOM		ζ at orthogonal mesh points (eq. (A3)), m/sec^2
ZETOMT		NEWRHO	new estimated value of ζ at a mesh point

Variable name	COMMON block	Subroutine	Description and comments
ZHIN	INPUTT		input z-coordinate of intersection with hub profile of line on which upstream flow conditions are given, m
ZHOUT	INPUTT		input z-coordinate of intersection with hub profile of line on which downstream flow conditions are given, m
ZHPLT(100)	PLTCOM		z-coordinates used for plotting hub profile, ZHUB, in MEPLOT, m
ZHST(NOSTAT)	INPUTT		input z-coordinates of intersections of hub-shroud output station lines with hub profile, m
ZHUB(NHUB)	INPUTT		input z-coordinates of points defining hub or bottom boundary of flow channel, m
ZILOM		LAMDAF	z-coordinate corresponding to RILOM
ZLE(NBLPL)	CALCON		z-coordinates of input blade section points (from ZBL) defining leading edge of blade, m
ZLEH		PRECAL	z-coordinate of intersection of leading edge of blade with hub profile, m

Variable name	COMMON block	Subroutine	Description and comments
ZLEOM(MHTP1)	CALCON		z-coordinates of intersections of horizontal mesh lines with blade leading edge, m
ZLESL		TSONIN	z-coordinate of intersection of a streamline with leading edge of blade, m
ZLET		PRECAL	z-coordinate of intersection of leading edge of blade with shroud profile, m
ZLPLT(100)	PLTCOM		z-coordinates used for plotting blade leading edge, ZLE, in MEPLOT, m
ZLRNG	PLTCOM		z-coordinate of left-hand boundary of a plot of meridional plane or orthogonal mesh, done in MEPLOT, m
ZNEW		NEWRHO	percentage of new calculated value of ZETOMT used in updating ZETOM
ZOLOM		RVTHTA	z-coordinate corresponding to ROLOM
ZOM(MM,MHTP1)	CALCON		z-coordinates of orthogonal mesh, m
ZOMBI	INPUTT		input z-coordinate of intersection of vertical mesh line with hub profile where first change in mesh spacing occurs (MBI), m

Variable name	COMMON block	Subroutine	Description and comments
ZOMBO	INPUTT		input z-coordinate of intersection of vertical mesh line with hub profile where second change in mesh spacing occurs (MBO), m
ZOMIN	INPUTT		input z-coordinate of intersection of left boundary of orthogonal mesh with hub profile, m
ZOMOUT	INPUTT		input z-coordinate of intersection of right boundary of orthogonal mesh (MM) with hub profile, m
ZPC(11,NBLPL)	INDCOM		z-coordinates of points of alternate mesh (fig. 26)
ZRRNG	PLTCOM		z-coordinate of right-hand boundary of a plot of meridional plane or orthogonal mesh, done in MEPLOT, m
ZSL(MM,NSL)	SLCOM		z-coordinates of points where streamlines cross vertical mesh lines, m
ZSLTEM(NSL)		OUTPUT	temporary storage for calculated values to be put into ZSL array, m
ZSPL		ILETE	z-coordinate on leading or trailing edge of blade corresponding to a streamline, m
ZSPLT(100)	PLTCOM		z-coordinates used for plotting shroud profile, ZTIP, in MEPLOT, m

Variable name	COMMON block	Subroutine	Description and comments
ZST(NSL,NOSTAT)	STACOM		z-coordinates of points where station lines cross streamlines, m
ZTE(NBLPL)	CALCON		z-coordinates of input blade section points (from ZBL) defining trailing edge of blade, m
ZTEH		PRECAL	z-coordinate of intersection of trailing edge of blade with hub profile, m
ZTEM(10)		INPUT	temporary storage for z-coordinate, m
ZTEM(MM)		MEPLOT	temporary storage of z-coordinates from ZOM for plotting, m
ZTEM(NBLPL)		THETOM	temporary storage of z-coordinates from ZPC, m
ZTEM(MHTP1 or NOSTAT or 20)		OUTPUT	temporary storage for values from ZOM array on vertical mesh lines; also temporary storage for values from ZST array along station lines, m
ZTEOM(MHTP1)	CALCON		z-coordinates of intersections of horizontal mesh lines with blade trailing edge, m
ZTESL		TSONIN	z-coordinate of intersection of a streamline with trailing edge of blade, m

Variable name	COMMON block	Subroutine	Description and comments
ZTET		PRECAL	z-coordinate of intersection of trailing edge of blade with shroud profile, m
ZTIN	INPUTT		input z-coordinate of intersection of line on which upstream flow conditions are given with shroud profile, m
ZTIP(NTIP)	INPUTT		input z-coordinates of points defining shroud or top boundary of flow channel, m
ZTOUT	INPUTT		input z-coordinate of intersection of line on which downstream flow conditions are given with shroud profile, m
ZTPLT(100)	PLTCOM		z-coordinate used for plotting blade trailing edge, ZTE, in MEPLOT, m
ZTST(NOSTAT)	INPUTT		input z-coordinates of intersections of hub-shroud output station lines with shroud profile, m

```
      COMMON SRW,SRE,ITER,IEND,NREAD,NWRIT
      COMMON/INPUTT/GAM,AR,MSFL,OMEGA,REDFAC,VELTOL,FNEW,DNEW,MBI,MBO,
     1   MM,MHT,NBL,NHUB,NTIP,NIN,NCUT,NBLPL,NPPP,NOSTAT,NSL,LSFR,
     2   LTPL,LAMVT,IMESH,ISLINE,ISTATL,IPLCT,ISUPER,ITSON,IDEBUG,
     3   ZOMIN,ZOMBI,ZCMBO,ZOMOUT,ZHIN,ZTIN,ZHOUT,ZTOUT,ZHUB(50),
     4   RHUB(50),ZTIP(50),RTIP(50),SFIN(50),RADIN(50),TIP(50),PRIP(50),
     5   LAMIN(50),VTHIN(50),SFOUT(50),RADOUT(50),PROP(50),LOSOUT(50),
     6   LAMOUT(50),VTHOUT(50),ZHST(50),ZTST(50),FLFR(50),
     7   ZBL(50,50),RBL(50,50),THBL(50,50),TNBL(50,50)
      EXTERNAL TIPF,TOPF,RHOIPF,RHOOPF,LAMDAF,RVTHTA
      INTEGER SRW,SRE
   10 IEND= -1
      ITER= 0
C
C--READ AND PLOT INPUT DATA
      CALL INPUT
      CALL INPLOT
C
C--GENERATE ORTHOGONAL MESH
      CALL MESHO
C
C--CALCULATE ALL PRELIMINARY FIXED CONSTANTS
      CALL PRECAL
C
C--PLOT ORTHOGONAL MESH
      CALL MEPLOT
C
C--CALCULATE STREAM FUNCTION ON UPSTREAM AND DOWNSTREAM
C--BOUNDARIES OF THE ORTHOGONAL MESH
      CALL VBDRY(1,TIPF,RHOIPF,LAMDAF)
      CALL VBDRY(MM,TOPF,RHOOPF,RVTHTA)
C
C--CALCULATE COEFFICIENTS, SOLVE DIFFERENTIAL EQUATIONS FOR STREAM
C--FUNCTION, AND COMPUTE NEW VELOCITIES AND DENSITIES
      CALL INIT
   20 ITER = ITER+1
      CALL COEF
      CALL SOR
      CALL NEWRHO
C
C--CALCULATE AND PRINT MAJOR OUTPUT DATA
      CALL OUTPUT
      CALL INDEV
C
C--PLOT STREAMLINES AND PLOT VELOCITIES
      CALL SLPLOT
      CALL SVPLOT
      IF (IEND.LT.0) GO TO 20
      IF (REDFAC.EQ.1.0) GO TO 10
C
C--OBTAIN TRANSONIC SOLUTION WITH FULL MASS FLOW
   30 CALL TVELCY
      IF (ISUPER.EQ.2) GO TO 10
      REDFAC = 1.0
```

```
      CALL TOUTPT
      CALL INDEV
      CALL SLPLOT
      CALL SVPLOT
      IF (ISUPER.EQ.0) GO TO 10
      ISUPER = 0
      GO TO 30
      END
```

```
      SUBROUTINE INPUT
C
C--INPUT READS AND PRINTS ALL INPUT DATA CARDS
C
      COMMON SRW,SRE,ITER,IEND,NREAD,NWRIT
      COMMON/INPUTT/GAM,AR,MSFL,OMEGA,REDFAC,VELTOL,FNEW,DNEW,MBI,MBO,
     1   MM,MHT,NBL,NHUB,NTIP,NIN,NOUT,NBLPL,NPPP,NOSTAT,NSL,LSFR,
     2   LTPL,LAMVT,IMESH,ISLINE,ISTATL,IPLOT,ISUPER,ITSON,IDEBUG,
     3   ZOMIN,ZOMBI,ZOMBO,ZOMOUT,ZHIN,ZTIN,ZHOUT,ZTOUT,ZHUB(50),
     4   RHUB(50),ZTIP(50),RTIP(50),SFIN(50),RADIN(50),TIP(50),PRIP(50),
     5   LAMIN(50),VTHIN(50),SFOUT(50),RADOUT(50),PROP(50),LOSOUT(50),
     6   LAMOUT(50),VTHOUT(50),ZHST(50),ZTST(50),FLFR(50),
     7   ZBL(50,50),RBL(50,50),THBL(50,50),TNBL(50,50)
      COMMON/CALCON/MMM1,MHTP1,CP,EXPON,TGROG,PITCH,CURVHI,CURVTI,
     1   CURVHO,CURVTO,RHIN,RTIN,RHOUT,RTOUT,RLEH,RLET,RTEH,RTET,
     2   ZLE(50),RLE(50),ZTE(50),RTE(50),ZLEOM(101),RLEOM(101),
     3   SLEOM(101),THLEOM(101),ZTEOM(101),RTEOM(101),STEOM(101),
     4   THTEOM(101),ILE(101),ITE(101),ZOM(100,101),ROM(100,101),
     5   SOM(100,101),TOM(100,101),BTH(100,101),DTHDS(100,101),
     6   DTHDT(100,101),PLOSS(100,101),CPHI(100,101),SPHI(100,101)
      COMMON/INTITL/TITLEI(20)
      DIMENSION ZTEM(10),RTEM(10),DYDX(10)
      REAL MSFL,LAMIN,LAMOUT,LOSOUT
C
C--READ AND PRINT INPUT DATA
C
      NREAD = 5
      NWRIT = 6
   10 WRITE(NWRIT,1000)
      READ (NREAD,1050) (TITLEI(I),I=1,20)
      WRITE(NWRIT,1050) (TITLEI(I),I=1,20)
      WRITE(NWRIT,1100)
      READ (NREAD,1030) GAM,AR,MSFL,OMEGA,REDFAC,VELTOL,FNEW,DNEW
      IF (REDFAC.LE.0.) REDFAC=1.0
      IF (VELTOL.LE.0.) VELTOL=.01
      IF (FNEW.LE.0.) FNEW=0.5
      IF (DNEW.LE.0.) DNEW=0.5
      VELTOL = VELTOL*AMIN1(FNEW,DNEW)
      WRITE(NWRIT,1040) GAM,AR,MSFL,OMEGA,REDFAC,VELTOL,FNEW,DNEW
      WRITE(NWRIT,1110)
      READ (NREAD,1010) MBI,MBO,MM,MHT,NBL,NHUB,NTIP,NIN,NOUT,NBLPL,
     1NPPP,NOSTAT,NSL
      WRITE(NWRIT,1020) MBI,MBO,MM,MHT,NBL,NHUB,NTIP,NIN,NOUT,NBLPL,
     1NPPP,NOSTAT,NSL
      WRITE(NWRIT,1120)
      READ (NREAD,1010) LSFR,LTPL,LAMVT
```

```
      WRITE(NWRIT,1020) LSFR,LTPL,LAMVT
      WRITE(NWRIT,1130)
      READ (NREAD,1030) ZOMIN,ZOMBI,ZOMBO,ZOMOUT
      WRITE(NWRIT,1040) ZOMIN,ZOMBI,ZOMBO,ZOMOUT
      WRITE(NWRIT,1140)
      READ (NREAD,1030) (ZHUB(I),I=1,NHUB)
      WRITE(NWRIT,1040) (ZHUB(I),I=1,NHUB)
      WRITE(NWRIT,1150)
      READ (NREAD,1030) (RHUB(I),I=1,NHUB)
      WRITE(NWRIT,1040) (RHUB(I),I=1,NHUB)
      WRITE(NWRIT,1160)
      READ (NREAD,1030) (ZTIP(I),I=1,NTIP)
      WRITE(NWRIT,1040) (ZTIP(I),I=1,NTIP)
      WRITE(NWRIT,1170)
      READ (NREAD,1030) (RTIP(I),I=1,NTIP)
      WRITE(NWRIT,1040) (RTIP(I),I=1,NTIP)
      WRITE(NWRIT,1180)
      READ (NREAD,1030) ZHIN,ZTIN
      WRITE(NWRIT,1040) ZHIN,ZTIN
      IF (LSFR.EQ.1) GO TO 20
      WRITE(NWRIT,1190)
      READ (NREAD,1030) (SFIN(I),I=1,NIN)
      WRITE(NWRIT,1040) (SFIN(I),I=1,NIN)
      GO TO 30
   20 WRITE(NWRIT,1200)
      READ (NREAD,1030) (RADIN(I),I=1,NIN)
      WRITE(NWRIT,1040) (RADIN(I),I=1,NIN)
   30 WRITE(NWRIT,1210)
      READ (NREAD,1030) (TIP(I),I=1,NIN)
      WRITE(NWRIT,1040) (TIP(I),I=1,NIN)
      WRITE(NWRIT,1220)
      READ (NREAD,1030) (PRIP(I),I=1,NIN)
      WRITE(NWRIT,1040) (PRIP(I),I=1,NIN)
      IF (LAMVT.EQ.1) GO TO 40
      WRITE(NWRIT,1230)
      READ (NREAD,1030) (LAMIN(I),I=1,NIN)
      WRITE(NWRIT,1040) (LAMIN(I),I=1,NIN)
      GO TO 50
   40 WRITE(NWRIT,1240)
      READ (NREAD,1030) (VTHIN(I),I=1,NIN)
      WRITE(NWRIT,1040) (VTHIN(I),I=1,NIN)
   50 WRITE(NWRIT,1250)
      READ (NREAD,1030) ZHOUT,ZTOUT
      WRITE(NWRIT,1040) ZHOUT,ZTOUT
      IF (LSFR.EQ.1) GO TO 60
      WRITE(NWRIT,1260)
      READ (NREAD,1030) (SFOUT(I),I=1,NOUT)
      WRITE(NWRIT,1040) (SFOUT(I),I=1,NOUT)
      GO TO 70
   60 WRITE(NWRIT,1270)
      READ (NREAD,1030) (RADOUT(I),I=1,NOUT)
      WRITE(NWRIT,1040) (RADOUT(I),I=1,NOUT)
   70 IF (LTPL.EQ.1) GO TO 80
      WRITE(NWRIT,1280)
      READ (NREAD,1030) (PROP(I),I=1,NOUT)
      WRITE(NWRIT,1040) (PROP(I),I=1,NOUT)
      GO TO 90
   80 WRITE(NWRIT,1290)
      READ (NREAD,1030) (LOSOUT(I),I=1,NOUT)
      WRITE(NWRIT,1040) (LOSOUT(I),I=1,NOUT)
```

```
   90 IF (LAMVT.EQ.1) GO TO 100
      WRITE(NWRIT,1300)
      READ (NREAD,1030) (LAMOUT(I),I=1,NOUT)
      WRITE(NWRIT,1040) (LAMOUT(I),I=1,NOUT)
      GO TO 110
  100 WRITE(NWRIT,1310)
      READ (NREAD,1030) (VTHOUT(I),I=1,NOUT)
      WRITE(NWRIT,1040) (VTHOUT(I),I=1,NOUT)
  110 WRITE(NWRIT,1320)
      DO 120 JN=1,NBLPL
      READ (NREAD,1030) (ZBL(IN,JN),IN=1,NPPP)
  120 WRITE(NWRIT,1040) (ZBL(IN,JN),IN=1,NPPP)
      WRITE(NWRIT,1330)
      DO 130 JN=1,NBLPL
      READ (NREAD,1030) (RBL(IN,JN),IN=1,NPPP)
  130 WRITE(NWRIT,1040) (RBL(IN,JN),IN=1,NPPP)
      WRITE(NWRIT,1340)
      DO 140 JN=1,NBLPL
      READ (NREAD,1030) (THBL(IN,JN),IN=1,NPPP)
  140 WRITE(NWRIT,1040) (THBL(IN,JN),IN=1,NPPP)
      WRITE(NWRIT,1350)
      DO 150 JN=1,NBLPL
      READ (NREAD,1030) (TNBL(IN,JN),IN=1,NPPP)
  150 WRITE(NWRIT,1040) (TNBL(IN,JN),IN=1,NPPP)
      IF (NOSTAT.EQ.0) GO TO 160
      WRITE(NWRIT,1360)
      READ (NREAD,1030) (ZHST(I),I=1,NOSTAT)
      WRITE(NWRIT,1040) (ZHST(I),I=1,NOSTAT)
      WRITE(NWRIT,1370)
      READ (NREAD,1030) (ZTST(I),I=1,NOSTAT)
      WRITE(NWRIT,1040) (ZTST(I),I=1,NOSTAT)
  160 IF (NSL.EQ.0) GO TO 170
      WRITE(NWRIT,1380)
      READ (NREAD,1030) (FLFR(I),I=1,NSL)
      WRITE(NWRIT,1040) (FLFR(I),I=1,NSL)
  170 WRITE(NWRIT,1390)
      READ (NREAD,1010) IMESH,ISLINE,ISTATL,IPLOT,ISUPER,ITSON,IDEBUG
      WRITE(NWRIT,1020) IMESH,ISLINE,ISTATL,IPLOT,ISUPER,ITSON,IDEBUG
      WRITE(NWRIT,1000)
      IF (MM.LE.100.AND.MHT.LE.100.AND.NHUB.LE.50.AND.NTIP.LE.50.AND.
     1NIN.LE.50.AND.NOUT.LE.50.AND.NBLPL.LE.50.AND.NPPP.LE.50.AND.
     2NOSTAT.LE.50.AND.NSL.LE.50.AND.LSFR.GE.0.AND.LSFR.LE.1.AND.
     3LTPL.GE.0.AND.LTPL.LE.1.AND.LAMVT.GE.0.AND.LAMVT.LE.1) GO TO 180
      WRITE(NWRIT,1400)
      STOP
C
C--CALCULATE MISCELLANEOUS CONSTANTS
C
  180 MMM1 = MM-1
      MHTP1= MHT+1
      EXPON= 1./(GAM-1.)
      CP = AR*GAM*EXPON
      TGROG= 2.*GAM*AR/(GAM+1.)
      PITCH= 2.*3.1415927/FLOAT(NBL)
      MSFL = MSFL/FLOAT(NBL)
C
C--ASSUME VALUES FOR ZHIN,ZTIN,ZHOUT,AND ZTOUT
C--IF THEY WERE NOT GIVEN AS INPUT
C
```

```
      IF (LSFR.EQ.1) GO TO 200
      IF (ZHIN.NE.0..OR.ZTIN.NE.0.) GO TO 190
      ZHIN = ZOMIN
      ZTIN = ZOMIN
  190 IF (ZHOUT.NE.0..OR.ZTOUT.NE.0.) GO TO 200
      ZHOUT = ZOMOUT
      ZTOUT = ZOMOUT
C
C--CALCULATE ESTIMATED UPSTREAM AND DOWNSTREAM VALUES OF
C--STREAM FUNCTION, IF RADIUS WAS GIVEN AS INPUT
C
  200 ZTEM(1)= ZHIN
      ZTEM(2)= ZHOUT
      CALL SPLINT(ZHUB,RHUB,NHUB,ZTEM,2,RTEM,DYDX)
      RHIN= RTEM(1)
      RHOUT= RTEM(2)
      ZTEM(1)= ZTIN
      ZTEM(2)= ZTOUT
      CALL SPLINT(ZTIP,RTIP,NTIP,ZTEM,2,RTEM,DYDX)
      RTIN= RTEM(1)
      RTOUT= RTEM(2)
      RINSQ = RTIN**2-RHIN**2
      ROUTSQ = RTOUT**2-RHOUT**2
      IF (LSFR.EQ.0) GO TO 230
      DO 210 J=1,NIN
  210 SFIN(J) = (RADIN(J)**2-RHIN**2)/RINSQ
      DO 220 J=1,NOUT
  220 SFOUT(J) = (RADOUT(J)**2-RHOUT**2)/ROUTSQ
      GO TO 260
C
C--CALCULATE ESTIMATED UPSTREAM AND DOWNSTREAM VALUES OF
C--RADIUS, IF STREAM FUNCTION WAS GIVEN AS INPUT
C
  230 DO 240 J=1,NIN
  240 RADIN(J) = SQRT(RHIN**2+SFIN(J)*RINSQ)
      DO 250 J=1,NOUT
  250 RADOUT(J) = SQRT(RHOUT**2+SFOUT(J)*ROUTSQ)
C
C--CALCULATE ESTIMATED UPSTREAM AND DOWNSTREAM TANGENTIAL VELOCITIES,
C--IF WHIRL WAS GIVEN AS INPUT
C
  260 IF (LAMVT.EQ.1) GO TO 290
      DO 270 J=1,NIN
  270 VTHIN(J) = LAMIN(J)/RADIN(J)
      DO 280 J=1,NOUT
  280 VTHOUT(J) = LAMOUT(J)/RADOUT(J)
      IF (LSFR.EQ.1) LAMVT=1
      RETURN
C
C--CALCULATE ESTIMATED UPSTREAM AND DOWNSTREAM WHIRL,
C--IF TANGENTIAL VELOCITY WAS GIVEN AS INPUT
C
  290 DO 300 J=1,NIN
  300 LAMIN(J) = RADIN(J)*VTHIN(J)
      DO 310 J=1,NOUT
  310 LAMOUT(J) = RADOUT(J)*VTHOUT(J)
C
C--FORMAT STATEMENTS
C
```

```
1000 FORMAT (1H1)
1010 FORMAT (16I5)
1020 FORMAT (2X,16(2X,I5))
1030 FORMAT (8F10.5)
1040 FORMAT (1X,8G16.7)
1050 FORMAT (20A4)
1100 FORMAT (///4X,20HGENERAL  INPUT  DATA/7X,3HGAM,14X,2HAR,13X,
    14HMSFL,11X,5HOMEGA,11X,6HREDFAC,10X,6HVELTOL,10X,4HFNEW,11X,
    24HDNEW)
1110 FORMAT (95H        MBI     MBO     MM      MHT     NBL    NHUB    NTIP
    1  NIN   NOUT   NBLPL   NPPP  NCSTAT   NSL)
1120 FORMAT (25H        LSFR    LTPL    LAMVT)
1130 FORMAT (///4X,29HHUB  AND  SHROUD  INPUT  DATA/7X,5HZOMIN,11X,
    1 5HZOMBI,11X,5HZOMBO,10X,6HZOMOUT)
1140 FORMAT (7X,11HZHUB  ARRAY)
1150 FORMAT (7X,11HRHUB  ARRAY)
1160 FORMAT (7X,11HZTIP  ARRAY)
1170 FORMAT (7X,11HRTIP  ARRAY)
1180 FORMAT (///4X,21HUPSTREAM  INPUT  DATA/7X,4HZHIN,11X,4HZTIN)
1190 FORMAT (7X,11HSFIN  ARRAY)
1200 FORMAT (7X,12HRADIN  ARRAY)
1210 FORMAT (7X,10HTIP  ARRAY)
1220 FORMAT (7X,11HPRIP  ARRAY)
1230 FORMAT (7X,12HLAMIN  ARRAY)
1240 FORMAT (7X,12HVTHIN  ARRAY)
1250 FORMAT (///4X,23HDOWNSTREAM  INPUT  DATA/7X,5HZHOUT,10X,5HZTOUT)
1260 FORMAT (7X,12HSFOUT  ARRAY)
1270 FORMAT (7X,13HRADOUT  ARRAY)
1280 FORMAT (7X,11HPROP  ARRAY)
1290 FORMAT (7X,13HLOSOUT  ARRAY)
1300 FORMAT (7X,13HLAMOUT  ARRAY)
1310 FORMAT (7X,13HVTHOUT  ARRAY)
1320 FORMAT (///4X,54HBLADE  MEAN  CAMBER  LINE  AND  THICKNESS  INPUT
    1 DATA/7X,10HZBL  ARRAY)
1330 FORMAT (7X,10HRBL  ARRAY)
1340 FORMAT (7X,11HTHBL  ARRAY)
1350 FORMAT (7X,11HTNBL  ARRAY)
1360 FORMAT (///4X,31HOUTPUT  STATION  LOCATION  DATA/7X,11HZHST  ARRAY
    1)
1370 FORMAT (7X,11HZTST  ARRAY)
1380 FORMAT (///4X,40HOUTPUT  STREAMLINE  FLOW  FRACTION  DATA/7X,11HFL
    1FR  ARRAY)
1390 FORMAT (///4X,28HOUTPUT  PRINT  CONTROL  DATA/6X,48HIMESH  ISLINE
    1ISTATL IPLOT  ISUPER ITSON  IDEBUG)
1400 FORMAT (94H1 MM,MHT,NHUB,NTIP,NIN,NOUT,NBLPL,NPPP,NCSTAT,NSL,LSFR,
    1LTPL,OR LAMVT IS TOO LARGE OR TOO SMALL)
     RETURN
     END
```

```
      SUBROUTINE INPLOT
C
C--INPLOT PLOTS THE UPSTREAM AND DOWNSTREAM INPUT FLOW VARIABLES
C--AS WELL AS THE INPUT BLADE SECTIONS FROM HUB TO SHROUD
C
```

```
      COMMON/INPUTT/GAM,AR,MSFL,OMEGA,REDFAC,VELTOL,FNEW,DNEW,MBI,MBO,
     1   MM,MHT,NBL,NHUB,NTIP,NIN,NOUT,NBLPL,NPPP,NOSTAT,NSL,LSFR,
     2   LTPL,LAMVT,IMESH,ISLINE,ISTATL,IPLOT,ISUPER,ITSCN,IDEBUG,
     3   ZOMIN,ZOMBI,ZOMBO,ZOMOUT,ZHIN,ZTIN,ZHOUT,ZTOUT,ZHUB(50),
     4   RHUB(50),ZTIP(50),RTIP(50),SFIN(50),RADIN(50),TIP(50),PRIP(50),
     5   LAMIN(50),VTHIN(50),SFOUT(50),RADOUT(50),PROP(50),LOSOUT(50),
     6   LAMOUT(50),VTHOUT(50),ZHST(50),ZTST(50),FLFR(50),
     7   ZBL(50,50),RBL(50,50),THBL(50,50),TNBL(50,50)
      COMMON/CALCON/MMM1,MHTP1,CP,EXPON,TGROG,PITCH,CURVHI,CURVTI,
     1   CURVHO,CURVTO,RHIN,RTIN,RHOUT,RTOUT,RLEH,RLET,RTEH,RTET,
     2   ZLE(50),RLE(50),ZTE(50),RTE(50),ZLEOM(101),RLEOM(101),
     3   SLEOM(101),THLEOM(101),ZTEOM(101),RTEOM(101),STEOM(101),
     4   THTEOM(101),ILE(101),ITE(101),ZOM(100,101),ROM(100,101),
     5   SOM(100,101),TOM(100,101),BTH(100,101),DTHDS(100,101),
     6   DTHDT(100,101),PLOSS(100,101),CPHI(100,101),SPHI(100,101)
      COMMON/INTITL/TITLEI(20)
      DIMENSION MBL(50,50),RTHBL(50,50),STHBL(50,50),PTHBL(50,50),
     1   RRTHBL(50,50),SSTHBL(50,50),PPTHBL(50,50),
     2   DTHDM(50),ANG(50),PLTX(101),PLTY(101),GRAD(101),
     3   TITL1(9),TITL2(8),TITL3(5),TITL4(9),TITL5(8),TITL6(9),TITL7(6),
     4   TITL8(9),TITL10(13),TITL11(6),TITL12(6),TITL13(4),
     5   TITL14(2),TITL15(5),TITL16(5),TITL17(5),TITL18(5),TITL19(5),
     6   TITL20(5),TITL21(5),TITL22(5),TITL24(10),TITL25(9)
      REAL MBL,LAMIN,LAMOUT,LOSOUT,LRNG
      DATA TITL1/' INL','ET A','BSOL','UTE ','TOTA','L TE','MPER','ATUR'
     1,'E   '/
      DATA TITL2/'INLE','T AB','SOLU','TE T','OTAL',' PRE','SSUR','E   '
     1/
      DATA TITL3/'INLE','T AB','SOLU','TE W','HIRL'/
      DATA TITL4/'INLE','T AB','SOLU','TE T','ANGE','NTIA','L VE','LOCI'
     1,'TY  '/
      DATA TITL5/'OUTL','ET A','BSOL','UTE ','TOTA','L PR','ESSU','RE  '
     1/
      DATA TITL6/'OUTL','ET A','BSOL','UTE ','TOTA','L PR','ESSU','RE L'
     1,'OSS '/
      DATA TITL7/'OUTL','ET A','BSOL','UTE ','WHIR','L   '/
      DATA TITL8/'OUTL','ET A','BSOL','UTE ','TANG','ENTI','AL V','ELOC'
     1,'ITY '/
      DATA TITL10/'INPU','T BL','ADE ','SECT','IONS','$C1$','L2FR',
     1'OM Z','BL, ','RBL,',' THB','L, T','NBL '/
      DATA TITL11/'   B','LADE',' SEC','TION',' NO.','XXXX'/
      DATA TITL12/'COMB','INED',' BLA','DE S','ECTI','ONS '/
      DATA TITL13/'STRE','AM F','UNCT','ION '/
      DATA TITL14/'  RA','DIUS'/
      DATA TITL15/'INPU','T AR','RAY ','- TI','P   '/
      DATA TITL16/'INPU','T AR','RAY ','- PR','IP  '/
      DATA TITL17/'INPU','T AR','RAY ','- LA','MIN '/
      DATA TITL18/'INPU','T AR','RAY ','- VT','HIN '/
      DATA TITL19/'INPU','T AR','RAY ','- PR','OP  '/
      DATA TITL20/'INPU','T AR','RAY ','- LO','SOUT'/
      DATA TITL21/'INPU','T AR','RAY ','- LA','MOUT'/
      DATA TITL22/'INPU','T AR','RAY ','- VT','HOUT'/
      DATA TITL24/'BLAD','E  S','ECTI','ON  ','MERI','DION','AL  ',
     1'COOR','DINA','TE  '/
      DATA TITL25/'TANG','ENTI','AL C','OORD','INAT','E - ','RADI',
     1'US*T','HETA'/
      DATA BLNK/'    '/
      DATA SYM/'X'/
      IF (IPLOT.LE.0) RETURN
```

```
C
C--PLOT TITLE ON MICROFILM
C
      CALL LRSIZE(0.0,20.0,0.0,10.0)
      CALL LRCHSZ(4)
      CALL LRLEGN(TITLEI,80,0,1.0,5.0,1.0)
      CALL LRCHSZ(2)
      CALL LRSIZE(0.0,10.0,0.0,10.0)
      CALL LRMON
      CALL LRXLEG(BLNK,1)
      CALL LRMOFF
C
C--PREPARE FOR PLOTTING OF INLET CONDITIONS
C
      IF (LSFR.EQ.1) GO TO 20
      PLTY(1) = SFIN(1)
      PLTY(101) = SFIN(NIN)
      DEL = (SFIN(NIN)-SFIN(1))/100.
      DO 10 J=2,100
   10 PLTY(J) = PLTY(J-1)+DEL
      BRNG = AMIN1(SFIN(1),SFOUT(1))
      TRNG = AMAX1(SFIN(NIN),SFOUT(NOUT))
      GO TO 40
   20 PLTY(1) = RADIN(1)
      PLTY(101) = RADIN(NIN)
      DEL = (RADIN(NIN)-RADIN(1))/100.
      DO 30 J=2,100
   30 PLTY(J) = PLTY(J-1)+DEL
      BRNG = AMIN1(RADIN(1),RADOUT(1))
      TRNG = AMAX1(RADIN(NIN),RADOUT(NOUT))
C
C--PLOT INLET ABSOLUTE TOTAL TEMPERATURE
C
   40 LRNG = TIP(1)
      RRNG = TIP(1)
      DO 50 J=1,NIN
      LRNG = AMIN1(LRNG,TIP(J))
   50 RRNG = AMAX1(RRNG,TIP(J))
      CALL LRMRGN(1.0,1.0,2.0,1.0)
      CALL LRANGE(LRNG,RRNG,BRNG,TRNG)
      CALL LRGRID(1,1,11.0,11.0)
      CALL LRCHSZ(4)
      CALL LRLEGN(TITL1,36,0,1.0,0.5,0.0)
      CALL LRCHSZ(2)
      CALL LRLEGN(TITL15,20,0,4.0,1.3,0.0)
      IF (LSFR.EQ.0) CALL LRLEGN(TITL13,16,1,0.2,4.2,0.0)
      IF (LSFR.EQ.1) CALL LRLEGN(TITL14,8,1,0.2,4.7,0.0)
      CALL LRCHSZ(4)
      IF (LSFR.EQ.0) CALL SPLINT(SFIN,TIP,NIN,PLTY,101,PLTX,GRAD)
      IF (LSFR.EQ.1) CALL SPLINT(RADIN,TIP,NIN,PLTY,101,PLTX,GRAD)
      CALL LRCURV(PLTX,PLTY,101,2,SYM,0.0)
      IF (LSFR.EQ.0) CALL LRCURV(TIP,SFIN,NIN,4,SYM,1.0)
      IF (LSFR.EQ.1) CALL LRCURV(TIP,RADIN,NIN,4,SYM,1.0)
C
C--PLOT INLET ABSOLUTE TOTAL PRESSURE
C
      LRNG = PRIP(1)
      RRNG = PRIP(1)
      DO 60 J=1,NIN
```

```
      LRNG = AMIN1(LRNG,PRIP(J))
   60 RRNG = AMAX1(RRNG,PRIP(J))
      CALL LRANGE(LRNG,RRNG,BRNG,TRNG)
      CALL LRLEGN(TITL2,32,0,1.6,0.5,0.0)
      CALL LRCHSZ(2)
      CALL LRLEGN(TITL16,20,0,4.0,1.3,0.0)
      IF (LSFR.EQ.0) CALL LRLEGN(TITL13,16,1,0.2,4.2,0.0)
      IF (LSFR.EQ.1) CALL LRLEGN(TITL14,8,1,0.2,4.7,0.0)
      CALL LRCHSZ(4)
      IF (LSFR.EQ.0) CALL SPLINT(SFIN,PRIP,NIN,PLTY,101,PLTX,GRAD)
      IF (LSFR.EQ.1) CALL SPLINT(RADIN,PRIP,NIN,PLTY,101,PLTX,GRAD)
      CALL LRCURV(PLTX,PLTY,101,2,SYM,0.0)
      IF (LSFR.EQ.0) CALL LRCURV(PRIP,SFIN,NIN,4,SYM,1.0)
      IF (LSFR.EQ.1) CALL LRCURV(PRIP,RADIN,NIN,4,SYM,1.0)
C
C--PLOT INLET ABSOLUTE WHIRL
C
      IF (LAMVT.EQ.1) GO TO 80
      LRNG = LAMIN(1)
      RRNG = LAMIN(1)
      DO 70 J=1,NIN
      LRNG = AMIN1(LRNG,LAMIN(J))
   70 RRNG = AMAX1(RRNG,LAMIN(J))
      CALL LRANGE(LRNG,RRNG,BRNG,TRNG)
      CALL LRLEGN(TITL3,20,0,2.5,0.5,0.0)
      CALL LRCHSZ(2)
      CALL LRLEGN(TITL17,20,0,4.0,1.3,0.0)
      IF (LSFR.EQ.0) CALL LRLEGN(TITL13,16,1,0.2,4.2,0.0)
      IF (LSFR.EQ.1) CALL LRLEGN(TITL14,8,1,0.2,4.7,0.0)
      CALL LRCHSZ(4)
      IF (LSFR.EQ.0) CALL SPLINT(SFIN,LAMIN,NIN,PLTY,101,PLTX,GRAD)
      IF (LSFR.EQ.1) CALL SPLINT(RADIN,LAMIN,NIN,PLTY,101,PLTX,GRAD)
      CALL LRCURV(PLTX,PLTY,101,2,SYM,0.0)
      IF (LSFR.EQ.0) CALL LRCURV(LAMIN,SFIN,NIN,4,SYM,1.0)
      IF (LSFR.EQ.1) CALL LRCURV(LAMIN,RADIN,NIN,4,SYM,1.0)
      GO TO 110
C
C--PLOT INLET ABSOLUTE TANGENTIAL VELOCITY
C
   80 LRNG = VTHIN(1)
      RRNG = VTHIN(1)
      DO 90 J=1,NIN
      LRNG = AMIN1(LRNG,VTHIN(J))
   90 RRNG = AMAX1(RRNG,VTHIN(J))
      CALL LRANGE(LRNG,RRNG,BRNG,TRNG)
      CALL LRLEGN(TITL4,36,0,1.1,0.5,0.0)
      CALL LRCHSZ(2)
      CALL LRLEGN(TITL18,20,0,4.0,1.3,0.0)
      IF (LSFR.EQ.0) CALL LRLEGN(TITL13,16,1,0.2,4.2,0.0)
      IF (LSFR.EQ.1) CALL LRLEGN(TITL14,8,1,0.2,4.7,0.0)
      CALL LRCHSZ(4)
      IF (LSFR.EQ.0) CALL SPLINT(SFIN,LAMIN,NIN,PLTY,101,PLTX,GRAD)
      IF (LSFR.EQ.1) CALL SPLINT(RADIN,LAMIN,NIN,PLTY,101,PLTX,GRAD)
      RINSQ = RTIN**2-RHIN**2
      DO 100 J=1,101
      IF (LSFR.EQ.0) PLTX(J)=PLTX(J)/SQRT(RHIN**2+PLTY(J)*RINSQ)
  100 IF (LSFR.EQ.1) PLTX(J)=PLTX(J)/PLTY(J)
      CALL LRCURV(PLTX,PLTY,101,2,SYM,0.0)
      IF (LSFR.EQ.0) CALL LRCURV(VTHIN,SFIN,NIN,4,SYM,1.0)
```

```
      IF (LSFR.EQ.1) CALL LRCURV(VTHIN,RADIN,NIN,4,SYM,1.0)
C
C--PREPARE FOR PLOTTING OF OUTLET CONDITIONS
C
  110 IF (LSFR.EQ.1) GO TO 130
      PLTY(1) = SFOUT(1)
      PLTY(101) = SFOUT(NOUT)
      DEL = (SFOUT(NOUT)-SFOUT(1))/100.
      DO 120 J=2,100
  120 PLTY(J) = PLTY(J-1)+DEL
      GO TO 150
  130 PLTY(1) = RADOUT(1)
      PLTY(101) = RADOUT(NOUT)
      DEL = (RADOUT(NOUT)-RADOUT(1))/100.
      DO 140 J=2,100
  140 PLTY(J) = PLTY(J-1)+DEL
C
C--PLOT OUTLET ABSOLUTE TOTAL PRESSURE
C
  150 IF (LTPL.EQ.1) GO TO 170
      LRNG = PROP(1)
      RRNG = PROP(1)
      DO 160 J=1,NOUT
      LRNG = AMIN1(LRNG,PROP(J))
  160 RRNG = AMAX1(RRNG,PROP(J))
      CALL LRANGE(LRNG,RRNG,BRNG,TRNG)
      CALL LRLEGN(TITL5,32,0,1.5,0.5,0.0)
      CALL LRCHSZ(2)
      CALL LRLEGN(TITL19,20,0,4.0,1.3,0.0)
      IF (LSFR.EQ.0) CALL LRLEGN(TITL13,16,1,0.2,4.2,0.0)
      IF (LSFR.EQ.1) CALL LRLEGN(TITL14,8,1,0.2,4.7,0.0)
      CALL LRCHSZ(4)
      IF (LSFR.EQ.0) CALL SPLINT(SFOUT,PROP,NOUT,PLTY,101,PLTX,GRAD)
      IF (LSFR.EQ.1) CALL SPLINT(RADOUT,PROP,NOUT,PLTY,101,PLTX,GRAD)
      CALL LRCURV(PLTX,PLTY,101,2,SYM,0.0)
      IF (LSFR.EQ.0) CALL LRCURV(PROP,SFOUT,NOUT,4,SYM,1.0)
      IF (LSFR.EQ.1) CALL LRCURV(PROP,RADOUT,NOUT,4,SYM,1.0)
      GO TO 190
C
C--PLOT OUTLET ABSOLUTE TOTAL PRESSURE LOSS
C
  170 LRNG = LOSOUT(1)
      RRNG = LOSOUT(1)
      DO 180 J=1,NOUT
      LRNG = AMIN1(LRNG,LOSOUT(J))
  180 RRNG = AMAX1(RRNG,LOSOUT(J))
      CALL LRANGE(LRNG,RRNG,BRNG,TRNG)
      CALL LRLEGN(TITL6,36,0,1.0,0.5,0.0)
      CALL LRCHSZ(2)
      CALL LRLEGN(TITL20,20,0,4.0,1.3,0.0)
      IF (LSFR.EQ.0) CALL LRLEGN(TITL13,16,1,0.2,4.2,0.0)
      IF (LSFR.EQ.1) CALL LRLEGN(TITL14,8,1,0.2,4.7,0.0)
      CALL LRCHSZ(4)
      IF (LSFR.EQ.0) CALL SPLINT(SFOUT,LOSOUT,NOUT,PLTY,101,PLTX,GRAD)
      IF (LSFR.EQ.1) CALL SPLINT(RADOUT,LOSOUT,NOUT,PLTY,101,PLTX,GRAD)
      CALL LRCURV(PLTX,PLTY,101,2,SYM,0.0)
      IF (LSFR.EQ.0) CALL LRCURV(LOSOUT,SFOUT,NOUT,4,SYM,1.0)
      IF (LSFR.EQ.1) CALL LRCURV(LOSOUT,RADOUT,NOUT,4,SYM,1.0)
```

```
C
C--PLOT OUTLET ABSOLUTE WHIRL
C
  190 IF (LAMVT.EQ.1) GO TO 210
      LRNG = LAMOUT(1)
      RRNG = LAMOUT(1)
      DO 200 J=1,NOUT
      LRNG = AMIN1(LRNG,LAMOUT(J))
  200 RRNG = AMAX1(RRNG,LAMOUT(J))
      CALL LRANGE(LRNG,RRNG,BRNG,TRNG)
      CALL LRLEGN(TITL7,24,0,2.0,0.5,0.0)
      CALL LRCHSZ(2)
      CALL LRLEGN(TITL21,20,0,4.0,1.3,0.0)
      IF (LSFR.EQ.0) CALL LRLEGN(TITL13,16,1,0.2,4.2,0.0)
      IF (LSFR.EQ.1) CALL LRLEGN(TITL14,8,1,0.2,4.7,0.0)
      CALL LRCHSZ(4)
      IF (LSFR.EQ.0) CALL SPLINT(SFOUT,LAMOUT,NOUT,PLTY,101,PLTX,GRAD)
      IF (LSFR.EQ.1) CALL SPLINT(RADOUT,LAMOUT,NOUT,PLTY,101,PLTX,GRAD)
      CALL LRCURV(PLTX,PLTY,101,2,SYM,0.0)
      IF (LSFR.EQ.0) CALL LRCURV(LAMOUT,SFOUT,NOUT,4,SYM,1.0)
      IF (LSFR.EQ.1) CALL LRCURV(LAMOUT,RADOUT,NOUT,4,SYM,1.0)
      GO TO 240
C
C--PLOT OUTLET ABSOLUTE TANGENTIAL VELOCITY
C
  210 LRNG = VTHOUT(1)
      RRNG = VTHOUT(1)
      DO 220 J=1,NOUT
      LRNG = AMIN1(LRNG,VTHOUT(J))
  220 RRNG = AMAX1(RRNG,VTHOUT(J))
      CALL LRANGE(LRNG,RRNG,BRNG,TRNG)
      CALL LRLEGN(TITL8,36,0,1.0,0.5,0.0)
      CALL LRCHSZ(2)
      CALL LRLEGN(TITL22,20,0,4.0,1.3,0.0)
      IF (LSFR.EQ.0) CALL LRLEGN(TITL13,16,1,0.2,4.2,0.0)
      IF (LSFR.EQ.1) CALL LRLEGN(TITL14,8,1,0.2,4.7,0.0)
      CALL LRCHSZ(4)
      IF (LSFR.EQ.0) CALL SPLINT(SFOUT,LAMOUT,NOUT,PLTY,101,PLTX,GRAD)
      IF (LSFR.EQ.1) CALL SPLINT(RADOUT,LAMOUT,NOUT,PLTY,101,PLTX,GRAD)
      ROUTSQ = RTOUT**2-RHOUT**2
      DO 230 J=1,101
      IF (LSFR.EQ.0) PLTX(J)=PLTX(J)/SQRT(RHOUT**2+PLTY(J)*ROUTSQ)
  230 IF (LSFR.EQ.1) PLTX(J)=PLTX(J)/PLTY(J)
      CALL LRCURV(PLTX,PLTY,101,2,SYM,0.0)
      IF (LSFR.EQ.0) CALL LRCURV(VTHOUT,SFOUT,NOUT,4,SYM,1.0)
      IF (LSFR.EQ.1) CALL LRCURV(VTHOUT,RADOUT,NOUT,4,SYM,1.0)
C
C--PLOT INPUT BLADE SECTIONS
C
C--CALCULATE BLADE SECTION PLOT COORDINATES ALONG MERIDIONAL PLANE
  240 DO 250 JN=1,NBLPL
      MBL(1,JN) = ZBL(1,JN)
      DO 250 IN=2,NPPP
  250 MBL(IN,JN) = MBL(IN-1,JN)+SQRT((ZBL(IN,JN)-ZBL(IN-1,JN))**2+
     1(RBL(IN,JN)-RBL(IN-1,JN))**2)
C--CALCULATE TANGENTIAL PLOT COORDINATES
      DO 260 JN=1,NBLPL
      CALL SPLINE(MBL(1,JN),THBL(1,JN),NPPP,DTHDM,ANG)
      DELRTH = (RBL(1,JN)+RBL(NPPP,JN))/2.*PITCH
```

```
      DO 260 IN=1,NPPP
      ANG(IN) = ATAN(RBL(IN,JN)*DTHCM(IN))
      DELT = TNBL(IN,JN)/COS(ANG(IN))
      RTHBL(IN,JN) = RBL(IN,JN)*THBL(IN,JN)
      STHBL(IN,JN) = RTHBL(IN,JN)+DELT/2.
      PTHBL(IN,JN) = RTHBL(IN,JN)-DELT/2.
      RRTHBL(IN,JN) = RTHBL(IN,JN)+DELRTH
      SSTHBL(IN,JN) = STHBL(IN,JN)+DELRTH
  260 PPTHBL(IN,JN) = PTHBL(IN,JN)+DELRTH
C--CALCULATE RANGE OF PLOTS, AND SET UP FOR PLOTTING INDIVIDUAL
C--BLADE SECTIONS
      LRNG = MBL(1,1)
      RRNG = MBL(NPPP,1)
      BRNG = PTHBL(1,1)
      TRNG = SSTHBL(NPPP,NBLPL)
      DO 270 JN=1,NBLPL
      LRNG = AMIN1(LRNG,MBL(1,JN))
      RRNG = AMAX1(RRNG,MBL(NPPP,JN))
      DO 270 IN=1,NPPP
      BRNG = AMIN1(BRNG,PTHBL(IN,JN))
  270 TRNG = AMAX1(TRNG,SSTHBL(IN,JN))
      RRTEM = RRNG
      DELLR = RRNG-LRNG
      DELBT = TRNG-BRNG
      DELRNG = AMAX1(DELLR,DELBT)
      RRNG = LRNG+DELRNG
      TRNG = BRNG+DELRNG
      CALL LRANGE(LRNG,RRNG,BRNG,TRNG)
C--PLOT BLADE SECTIONS AND SHOW SOLIDITY
      CALL LRLEGN(TITL10,52,0,2.7,0.7,0.0)
      DO 280 JN=1,NBLPL
      CALL LRCHSZ(3)
      CALL LRCNVT(JN,1,TITL11(6),1,4,0)
      CALL LRLEGN(TITL11,24,0,3.0,9.5,0.0)
      CALL LRCHSZ(2)
      CALL LRLEGN(TITL24,40,0,2.8,1.3,0.0)
      CALL LRLEGN(TITL25,36,1,0.2,3.3,0.0)
      CALL LRCHSZ(4)
      CALL LRCURV(MBL(1,JN),RTHBL(1,JN),NPPP,2,SYM,0.0)
      CALL LRCURV(MBL(1,JN),RTHBL(1,JN),NPPP,4,SYM,0.0)
      CALL LRCURV(MBL(1,JN),STHBL(1,JN),NPPP,2,SYM,0.0)
      CALL LRCURV(MBL(1,JN),PTHBL(1,JN),NPPP,2,SYM,0.0)
      CALL LRCURV(MBL(1,JN),RRTHBL(1,JN),NPPP,2,SYM,0.0)
      CALL LRCURV(MBL(1,JN),RRTHBL(1,JN),NPPP,4,SYM,0.0)
      CALL LRCURV(MBL(1,JN),SSTHBL(1,JN),NPPP,2,SYM,0.0)
  280 CALL LRCURV(MBL(1,JN),PPTHBL(1,JN),NPPP,2,SYM,1.0)
C--CALCULATE RANGE OF PLOT, AND SET UP FOR PLOT OF MULTIPLE
C--BLADE SECTIONS
      RRNG = RRTEM
      TRNG = STHBL(NPPP,NBLPL)
      DO 290 JN=1,NBLPL
      DO 290 IN=1,NPPP
  290 TRNG = AMAX1(TRNG,STHBL(IN,JN))
      DELBT = TRNG-BRNG
      DELRNG = AMAX1(DELLR,DELBT)
      RRNG = LRNG+DELRNG
      TRNG = BRNG+DELRNG
      CALL LRANGE(LRNG,RRNG,BRNG,TRNG)
C--PLOT MULTIPLE BLADE SECTIONS
```

```
      CALL LRGRID(3,3,11.0,11.0)
      CALL LRCHSZ(3)
      CALL LRLEGN(TITL12,24,0,3.4,9.5,0.0)
      CALL LRCHSZ(2)
      CALL LRLEGN(TITL24,40,0,2.8,1.3,0.0)
      CALL LRLEGN(TITL25,36,1,0.2,3.3,0.0)
      CALL LRCHSZ(4)
      ECP = 0.0
      DO 300 JN=1,NBLPL
      IF (JN.EQ.NBLPL) ECP=1.0
      CALL LRCURV(MBL(1,JN),RTHBL(1,JN),NPPP,2,SYM,0.0)
      CALL LRCURV(MBL(1,JN),RTHBL(1,JN),NPPP,4,SYM,0.0)
      CALL LRCURV(MBL(1,JN),STHBL(1,JN),NPPP,2,SYM,0.0)
  300 CALL LRCURV(MBL(1,JN),PTHBL(1,JN),NPPP,2,SYM,ECP)
      CALL LRCURV(ZBL,RBL,0,1,SYM,1.0)
      RETURN
      END


      SUBROUTINE MESHO
C
C--MESHO CALCULATES COORDINATES OF AN ORTHOGONAL MESH
C--COVERING THE SOLUTION REGION
C
      COMMON/INPUTT/GAM,AR,MSFL,OMEGA,REDFAC,VELTOL,FNEW,DNEW,MBI,MBO,
     1  MM,MHT,NBL,NHUB,NTIP,NIN,NOUT,NBLPL,NPPP,NCSTAT,NSL,LSFR,
     2  LTPL,LAMVT,IMESH,ISLINE,ISTATL,IPLOT,ISUPER,ITSON,IDEBUG,
     3  ZOMIN,ZOMBI,ZOMBO,ZOMOUT,ZHIN,ZTIN,ZHOUT,ZTOUT,ZHUB(50),
     4  RHUB(50),ZTIP(50),RTIP(50),SFIN(50),RADIN(50),TIP(50),PRIP(50),
     5  LAMIN(50),VTHIN(50),SFOUT(50),RADOUT(50),PROP(50),LOSOUT(50),
     6  LAMOUT(50),VTHOUT(50),ZHST(50),ZTST(50),FLFR(50),
     7  ZBL(50,50),RBL(50,50),THBL(50,50),TNBL(50,50)
      COMMON/CALCON/MMM1,MHTP1,CP,EXPON,TGROG,PITCH,CURVHI,CURVTI,
     1  CURVHO,CURVTO,RHIN,RTIN,RHOUT,RTOUT,RLEH,RLET,RTEH,RTET,
     2  ZLE(50),RLE(50),ZTE(50),RTE(50),ZLEOM(101),RLEOM(101),
     3  SLEOM(101),THLEOM(101),ZTEOM(101),RTEOM(101),STEOM(101),
     4  THTEOM(101),ILE(101),ITE(101),ZOM(100,101),ROM(100,101),
     5  SOM(100,101),TOM(100,101),BTH(100,101),DTHDS(100,101),
     6  DTHDT(100,101),PLOSS(100,101),CPHI(100,101),SPHI(100,101)
      COMMON/CROSCM/RTMP(100),SDRIV(100),REFZ,REFR,REFSL,MARK
      DIMENSION RRAD(100,101),SLOM(100),AAA(100)
      EXTERNAL CROSCD
      MARK = 1
      CALL CROSCD(ZOM(1,1),RMR,SL1)
      MARK = 0
C
C--STORE RRAD ON HUB CONTOUR
      DO 10 I=1,NHUB
   10 RRAD(I,1)= RHUB(I)
C
C--CALCULATE RRAD ON SHROUD CONTOUR
      CALL SPLINT(ZTIP,RTIP,NTIP,ZHUB,NHUB,RRAD(1,MHTP1),AAA)
C
C--COMPUTE RRAD ON RADIAL LINES FROM HUB TO TIP
      DO 20 I=1,NHUB
      DELR = (RRAD(I,MHTP1)-RRAD(I,1))/FLOAT(MHT)
```

```
      DO 20 J=2,MHT
   20 RRAD(I,J)= RRAD(I,J-1)+DELR
C
C--COMPUTE ZOM ON HUB
      MBIM1 = MBI-1
      DELZ= (ZOMBI-ZOMIN)/FLOAT(MBIM1)
      ZOM(1,1)= ZOMIN
      DO 30 I=2,MBI
   30 ZOM(I,1)= ZOM(I-1,1)+DELZ
      DELZ = (ZOMBO-ZOMBI)/FLOAT(MBO-MBI)
      MBIP1 = MBI+1
      DO 40 I=MBIP1,MBO
   40 ZOM(I,1)= ZOM(I-1,1)+DELZ
      DELZ= (ZOMOUT-ZOMBO)/FLOAT(MM-MBO)
      MBOP1 = MBO+1
      DO 50 I=MBOP1,MM
   50 ZOM(I,1)= ZOM(I-1,1)+DELZ
C
C--COMPUTE ENTRIES TO ZOM AND ROM ROW BY ROW FROM HUB TO SHROUD
C
      DO 150 J=2,MHTP1
C
C--CALCULATE R-COORDINATES, SLOPES, AND ANGLES OF PREVIOUS ROW
      CALL SPLINT(ZHUB,RRAD(1,J-1),NHUB,ZOM(1,J-1),MM,ROM(1,J-1),SLOM)
      DO 60 I=1,MM
      CPHI(I,J-1) = 1./SQRT(1.+SLOM(I)**2)
   60 SPHI(I,J-1) = SLOM(I)*CPHI(I,J-1)
C
C--CALCULATE RTMP, AND SECOND DERIVATIVES, SDRIV, ON PRESENT ROW
      DO 70 I=1,NHUB
   70 RTMP(I)= RRAD(I,J)
      CALL SPLINE(ZHUB,RTMP,NHUB,AAA,SDRIV)
C
C--MOVE ALONG PRESENT ROW, ONE POINT AT A TIME, LOCATING ZOM
C--COORDINATES OF ORTHOGONAL MESH POINTS ALONG THE ROW
      DO 140 I=1,MM
      REFZ= ZOM(I,J-1)
      REFR= ROM(I,J-1)
      DO 80 K=2,NHUB
      IF (ZHUB(K).LT.ZOM(I,J-1)) GO TO 80
      DELR = RRAD(K-1,J)-RRAD(K-1,J-1)-(ZOM(I,J-1)-ZHUB(K-1))/
     1(ZHUB(K)-ZHUB(K-1))*(RRAD(K-1,J)-RRAD(K-1,J-1)-RRAD(K,J)+
     2RRAD(K,J-1))
      GO TO 90
   80 CONTINUE
   90 IF (ABS(SLOM(I)).LE.0.0001) GO TO 120
C
C--LOCATE INTERSECTION OF LINE NORMAL TO PREVIOUS ROW WITH PRESENT ROW
      REFSL= -1./SLOM(I)
      DELZ= DELR/REFSL
      IF (ABS(REFSL).GT.10.) DELZ=DELR/SIGN(10.,REFSL)
      IF (REFSL.LT.0.) GO TO 100
      A= ZOM(I,J-1)
      B= A+2.0*DELZ
      GO TO 110
  100 B= ZOM(I,J-1)
      A= B+2.0*DELZ
  110 TOLER= DELR/110.
      IF(ABS(SLOM(I)).LE..01) TOLER = TOLER/ABS(SLOM(I))*.01
```

```
      CALL ROOT(A,B,0.,CROSCD,TOLER,Z1,SL1)
      R1= REFR+REFSL*(Z1-REFZ)
      GO TO 130
C
C--LOCATE INTERSECTION IF NORMAL TO PREVIOUS ROW IS RADIAL
  120 REFSL= 0.
      CALL CROSCD(ZOM(I,J-1),RMR,SL1)
      Z1= ZOM(I,J-1)
      R1= RMR
C
C--CALCULATE FINAL LOCATION OF ZOM
  130 Z2= (ZOM(I,J-1)+(ROM(I,J-1)-R1)*SL1+Z1*SL1**2)/(1.+SL1**2)
      R2= R1+SL1*(Z2-Z1)
  140 ZOM(I,J)= (Z1+Z2)/2.
  150 CONTINUE
C
C--CALCULATE R-COORDINATES, SLOPES, AND ANGLES OF FINAL ROW
      CALL SPLINT(ZHUB,RRAD(1,MHTP1),NHUB,ZOM(1,MHTP1),MM,
     1ROM(1,MHTP1),SLOM)
      DO 160 I=1,MM
      CPHI(I,MHTP1) = 1./SQRT(1.+SLOM(I)**2)
  160 SPHI(I,MHTP1) = SLOM(I)*CPHI(I,MHTP1)
      RETURN
      END




      SUBROUTINE CROSCD(Z,RMR,SL1)
C
C--CROSCD CALCULATES R AS A FUNCTION OF Z ALONG A CURVE AND ITS
C--INTERSECTING STRAIGHT LINE, AND COMPUTES THE DIFFERENCE BETWEEN THE
C--VALUES OF R ON THE STRAIGHT LINE AND CURVE FOR A GIVEN VALUE OF Z
C
      COMMON/INPUTT/GAM,AR,MSFL,OMEGA,REDFAC,VELTOL,FNEW,DNEW,MBI,MBO,
     1   MM,MHT,NBL,NHUB,NTIP,NIN,NOUT,NBLPL,NPPP,NOSTAT,NSL,LSFR,
     2   LTPL,LAMVT,IMESH,ISLINE,ISTATL,IPLOT,ISUPER,ITSCN,IDEBUG,
     3   ZOMIN,ZOMBI,ZOMBO,ZOMOUT,ZHIN,ZTIN,ZHOUT,ZTOUT,ZHUB(50),
     4   RHUB(50),ZTIP(50),RTIP(50),SFIN(50),RADIN(50),TIP(50),PRIP(50),
     5   LAMIN(50),VTHIN(50),SFOUT(50),RADOUT(50),PRCP(50),LOSOUT(50),
     6   LAMOUT(50),VTHOUT(50),ZHST(50),ZTST(50),FLFR(50),
     7   ZBL(50,50),RBL(50,50),THBL(50,50),TNBL(50,50)
      COMMON/CROSCM/RTMP(100),SDRIV(100),REFZ,REFR,REFSL,MARK
      IF(MARK.EQ.1) RETURN
C
C--LOCATE POSITION OF Z IN ZHUB ARRAY
      DO 10 I=2,NHUB
      IF (Z.LE.ZHUB(I)) GO TO 20
   10 CONTINUE
C
C--COMPUTE R-COORDINATE (RCURV) AND SLOPE (SL1) ON THE CURVE
C--FOR THE GIVEN VALUE OF  Z
   20 DELZ = ZHUB(I)-ZHUB(I-1)
      SDI = SDRIV(I)
      SDI1 = SDRIV(I-1)
      ZHMZ = ZHUB(I)-Z
      ZMZH = Z-ZHUB(I-1)
      RTI = RTMP(I)/DELZ
```

```
      RTI1 = RTMP(I-1)/DELZ
      RCURV = SDI1*ZHMZ**3/6./DELZ+SDI*ZMZH**3/6./DELZ+(RTI-SDI*DELZ/6.)
     1*ZMZH+(RTI1-SDI1*DELZ/6.)*ZHMZ
      SL1 = -SDI1*ZHMZ**2/2./DELZ + SDI*ZMZH**2/2./DELZ + RTI - RTI1 -
     1 (SDI - SDI1)*DELZ/6.
      IF (REFSL.EQ.0.) GO TO 30
C
C--COMPUTE R-COORDINATE (RLINE) ON STRAIGHT LINE
C--FOR GIVEN VALUE OF  Z
      RLINE= REFR+REFSL*(Z-REFZ)
C
C--COMPUTE DIFFERENCE IN R-COORDINATES
      RMR= RCURV-RLINE
      RETURN
C
C--SPECIAL CASE FOR RADIAL STRAIGHT LINE
   30 RMR = RCURV
      RETURN
      END


      SUBROUTINE PRECAL
C
C--PRECAL CALCULATES MANY OF THE REQUIRED FIXED CONSTANTS
C
      COMMON/INPUTT/GAM,AR,MSFL,OMEGA,REDFAC,VELTOL,FNEW,ONEW,MBI,MBO,
     1   MM,MHT,NBL,NHUB,NTIP,NIN,NOUT,NBLPL,NPPP,NOSTAT,NSL,LSFR,
     2   LTPL,LAMVT,IMESH,ISLINE,ISTATL,IPLOT,ISUPER,ITSCN,IDEBUG,
     3   ZOMIN,ZOMBI,ZOMBO,ZOMOUT,ZHIN,ZTIN,ZHOUT,ZTOUT,ZHUB(50),
     4   RHUB(50),ZTIP(50),RTIP(50),SFIN(50),RADIN(50),TIP(50),PRIP(50),
     5   LAMIN(50),VTHIN(50),SFOUT(50),RADOUT(50),PROP(50),LOSCUT(50),
     6   LAMOUT(50),VTHOUT(50),ZHST(50),ZTST(50),FLFR(50),
     7   ZBL(50,50),RBL(50,50),THBL(50,50),TNBL(50,50)
      COMMON/CALCON/MMM1,MHTP1,CP,EXPON,TGROG,PITCH,CURVHI,CURVTI,
     1   CURVHO,CURVTO,RHIN,RTIN,RHOUT,RTOUT,RLEH,RLET,RTEH,RTET,
     2   ZLE(50),RLE(50),ZTE(50),RTE(50),ZLEOM(101),RLEOM(101),
     3   SLEOM(101),THLEOM(101),ZTEOM(101),RTEOM(101),STEOM(101),
     4   THTEOM(101),ILE(101),ITE(101),ZOM(100,101),ROM(100,101),
     5   SOM(100,101),TOM(100,101),BTH(100,101),DTHDS(100,101),
     6   DTHDT(100,101),PLOSS(100,101),CPHI(100,101),SPHI(100,101)
      DIMENSION DYDX(100),DYDX2(100),TTEM(100),RELTOP(50)
      REAL MSFL,LAMDAF,LAMIN,LAMOUT,LAMDAI,LOSOUT
C
C--INITIALIZE TIPF, RHOIPF, LAMDAF, RHOOPF, AND RVTHTA
C
      CALL LAMNIT
      CALL RVTNIT
      CALL TIPNIT
      CALL RHINIT
C--CALCULATE PROP, IF LOSOUT WAS GIVEN AS INPUT
      IF (LTPL.EQ.0) GO TO 20
      DO 10 J=1,NOUT
      TINP = TIPF(SFOUT(J))
      LAMDAI = LAMDAF(SFOUT(J),1,1)
      TOP = TOPF(SFOUT(J))
      PRINP = RHOIPF(SFOUT(J))*AR*TINP
```

```
      RELTOP(J) = 1.-LOSOUT(J)
   10 PROP(J) = RELTOP(J)*PRINP*(TOP/TINP)**(GAM*EXPON)
   20 CALL RHONIT
C
C--INITIALIZE THE BTH ARRAY
C
      DO 30 J=1,MHTP1
      DO 30 I=1,MM
   30 BTH(I,J)= PITCH
C
C--INITIALIZE THE FLFR ARRAY IF IT WAS NOT READ IN
C
      IF (NSL.GE.1) GO TO 50
      NSL = 11
      FLFR(1) = 0.
      FLFR(11) = 1.0
      DO 40 J=2,10
   40 FLFR(J) = FLFR(J-1)+0.1
      GO TO 80
C
C--SET END POINTS FOR FLFR ARRAY
C
   50 IF (FLFR(1).EQ.0.) GO TO 70
      TEMP1 = 0.
      DO 60 JL=1,NSL
      TEMP2 = FLFR(JL)
      FLFR(JL) = TEMP1
   60 TEMP1 = TEMP2
      NSL = NSL+1
      FLFR(NSL) = TEMP1
   70 IF (FLFR(NSL).EQ.1.0) GO TO 80
      NSL = NSL+1
      FLFR(NSL) = 1.0
C
C--CALCULATE SOM FROM THE ZOM,ROM ARRAYS
C
   80 DO 90 J=1,MHTP1
      SOM(1,J) = 0.
      DO 90 I=2,MM
   90 SOM(I,J)= SOM(I-1,J)+SQRT((ZOM(I,J)-ZOM(I-1,J))**2+(ROM(I,J)-
     1ROM(I-1,J))**2)
C
C--CALCULATE TOM FROM THE ZOM,ROM ARRAYS
C
      DO 100 I=1,MM
      TOM(I,1) = 0.
      DO 100 J=2,MHTP1
  100 TOM(I,J)= TOM(I,J-1)+SQRT((ZOM(I,J)-ZOM(I,J-1))**2+(ROM(I,J)-
     1ROM(I,J-1))**2)
C
C--CALCULATE CURVATURES ON HUB AND SHROUD
C--AT ENDS OF ORTHOGONAL MESH
C
      CALL SPLINE(ZOM(1,1),ROM(1,1),MM,DYDX,DYDX2)
      CURVHI= DYDX2(1)/(1.+DYDX(1)**2)**1.5
      CURVHO= DYDX2(MM)/(1.+DYDX(MM)**2)**1.5
      CALL SPLINE(ZOM(1,MHTP1),ROM(1,MHTP1),MM,DYDX,DYDX2)
      CURVTI= DYDX2(1)/(1.+DYDX(1)**2)**1.5
      CURVTO= DYDX2(MM)/(1.+DYDX(MM)**2)**1.5
```

```
C
C--CALCULATE LEADING EDGE ARRAY, ZLE,RLE ,FROM ZBL AND RBL ARRAYS
C--CALCULATE INTERSECTION OF LEADING EDGE WITH HUB AND SHROUD PROFILES
C
      DO 110 JN=1,NBLPL
      ZLE(JN) = ZBL(1,JN)
  110 RLE(JN) = RBL(1,JN)
      CALL INRSCT(ZHUB,RHUB,NHUB,ZLE,RLE,NBLPL,ZLEH,RLEH)
      CALL INRSCT(ZTIP,RTIP,NTIP,ZLE,RLE,NBLPL,ZLET,RLET)
C
C--CALCULATE TRAILING EDGE ARRAY, ZTE,RTE ,FROM ZBL AND RBL ARRAYS
C--CALCULATE INTERSECTIONS OF TRAILING EDGE WITH HUB AND SHROUD PROFILES
C
      DO 120 JN=1,NBLPL
      ZTE(JN) = ZBL(NPPP,JN)
  120 RTE(JN) = RBL(NPPP,JN)
      CALL INRSCT(ZHUB,RHUB,NHUB,ZTE,RTE,NBLPL,ZTEH,RTEH)
      CALL INRSCT(ZTIP,RTIP,NTIP,ZTE,RTE,NBLPL,ZTET,RTET)
C
C--CALCULATE ORTHOGONAL MESH ARRAYS AT THE LEADING EDGE
C--ZLEOM,RLEOM,SLEOM,THLEOM
C--CALCULATE ILE ARRAY OF MESH POINT LOCATIONS INSIDE BLADE
C--LEADING EDGE
C
      ZLEOM(1) = ZLEH
      RLEOM(1) = RLEH
      ZLEOM(MHTP1) = ZLET
      RLEOM(MHTP1) = RLET
      DO 130 J=2,MHT
  130 CALL INRSCT(ZOM(1,J),ROM(1,J),MM,ZLE,RLE,NBLPL,ZLEOM(J),RLEOM(J))
      DO 160 J=1,MHTP1
      DO 140 I=1,MM
      IF (ZLEOM(J).LE.ZOM(I,J)) GO TO 150
  140 CONTINUE
  150 ILE(J) = I
      ILEJ = I-1
  160 SLEOM(J) = SOM(ILEJ,J)+SQRT((ZLEOM(J)-ZOM(ILEJ,J))**2+(RLEOM(J)-
     1ROM(ILEJ,J))**2)
      DO 170 JN=1,NBLPL
  170 TTEM(JN) = THBL(1,JN)
      CALL SPLINT(RLE,TTEM,NBLPL,RLEOM,MHTP1,THLEOM,CYCX)
C
C--CALCULATE ORTHOGONAL MESH ARRAYS AT THE TRAILING EDGE
C--ZTEOM,RTEOM,STEOM,THTEOM
C--CALCULATE ITE ARRAY OF MESH POINT LOCATIONS INSIDE BLADE
C--TRAILING EDGE
C
      ZTEOM(1) = ZTEH
      RTEOM(1) = RTEH
      ZTEOM(MHTP1) = ZTET
      RTEOM(MHTP1) = RTET
      DO 180 J=2,MHT
  180 CALL INRSCT(ZOM(1,J),ROM(1,J),MM,ZTE,RTE,NBLPL,ZTEOM(J),RTEOM(J))
      DO 210 J=1,MHTP1
      ILEJ = ILE(J)-1
      DO 190 I=ILEJ,MM
      IF (ZTEOM(J).LT.ZOM(I,J)) GO TO 200
  190 CONTINUE
  200 ITE(J) = I-1
```

```
      ITEJ = I-1
  210 STEOM(J) = SOM(ITEJ,J)+SQRT((ZTEOM(J)-ZOM(ITEJ,J))**2+(RTEOM(J)-
     1ROM(ITEJ,J))**2)
      DO 220 JN=1,NBLPL
  220 TTEM(JN) = THBL(NPPP,JN)
      CALL SPLINT(RTE,TTEM,NBLPL,RTEOM,MHTP1,THTEOM,DYDX)
C
C--CALCULATE THETA GRADIENTS ON THE ORTHOGONAL MESH
C
      CALL THETOM
C
C--CORRECT  BTH  FOR BLADE THICKNESS ON THE ORTHOGONAL MESH
C
      CALL THIKOM
C
C--CALCULATE ACTUAL-TO-IDEAL RELATIVE TOTAL PRESSURE RATIO
C--DOWNSTREAM OF BLADE, AND CALCULATE LOSS ON ORTHOGONAL MESH
C--CORRECT  BTH  FOR TOTAL PRESSURE LOSS
C
      CALL LOSSOM
C
C--REDUCE MASSFLOW, WHEEL SPEED, AND WHIRL FOR REDUCED FLOW SOLUTION
C
      IF (REDFAC.EQ.1.0) RETURN
      OMEGA = OMEGA*REDFAC
      MSFL = MSFL*REDFAC
      DO 230 J=1,NIN
      LAMIN(J) = LAMIN(J)*REDFAC
  230 VTHIN(J) =  VTHIN(J)*REDFAC
      DO 240 J=1,NOUT
      LAMOUT(J)= LAMOUT(J)*REDFAC
  240 VTHOUT(J) = VTHOUT(J)*REDFAC
C
C--RE-INITIALIZE LAMDAF AND RVTHTA FOR REDUCED FLOW
C
      CALL LAMNIT
      CALL RVTNIT
      RETURN
      END




      SUBROUTINE THETOM
C
C--THETOM CALCULATES THE DERIVATIVES OF THETA WITH RESPECT TO  S  AND  T
C--DIRECTIONS ON THE ORTHOGONAL MESH
C
      COMMON/INPUTT/GAM,AR,MSFL,OMEGA,REDFAC,VELTOL,FNEW,DNEW,MBI,MBO,
     1   MM,MHT,NBL,NHUB,NTIP,NIN,NOUT,NBLPL,NPPP,NCSTAT,NSL,LSFR,
     2   LTPL,LAMVT,IMESH,ISLINE,ISTATL,IPLOT,ISUPER,ITSON,IDEBUG,
     3   ZOMIN,ZOMBI,ZOMBO,ZOMOUT,ZHIN,ZTIN,ZHOUT,ZTOUT,ZHUB(50),
     4   RHUB(50),ZTIP(50),RTIP(50),SFIN(50),RADIN(50),TIP(50),PRIP(50),
     5   LAMIN(50),VTHIN(50),SFOUT(50),RADOUT(50),PROP(50),LOSOUT(50),
     6   LAMOUT(50),VTHOUT(50),ZHST(50),ZTST(50),FLFR(50),
     7   ZBL(50,50),RBL(50,50),THBL(50,50),TNBL(50,50)
      COMMON/CALCON/MMM1,MHTP1,CP,EXPON,TGROG,PITCH,CURVHI,CURVTI,
     1   CURVHO,CURVTO,RHIN,RTIN,RHOUT,RTOUT,RLEH,RLET,RTEH,RTET,
```

```
     2    ZLE(50),RLE(50),ZTE(50),RTE(50),ZLEOM(101),RLEOM(101),
     3    SLEOM(101),THLEOM(101),ZTEOM(101),RTEOM(101),STEOM(101),
     4    THTEOM(101),ILE(101),ITE(101),ZOM(100,101),ROM(100,101),
     5    SOM(100,101),TOM(100,101),BTH(100,101),DTHDS(100,101),
     6    DTHDT(100,101),PLOSS(100,101),CPHI(100,101),SPHI(100,101)
      COMMON/INDCOM/ZPC(11,50),RPC(11,50),DTHDZ(11,50),DTHDR(11,50)
      DIMENSION THPC(11,50),ANGZ(11,50),ANGR(11,50),DTHDSP(11,50),
     1    DTHDTP(11,50),SZRBL(50),SZRPC(50),ZTEM(50),RTEM(50),TTEM(50),
     2    DYDX(50),DYDX2(50)
C
C--CALCULATE GRADIENTS OF THETA WITH RESPECT TO DISTANCE ALONG INPUT
C--ZBL,RBL LINES
C
C--LOCATE INTERSECTIONS OF INPUT ZBL,RBL LINES WITH LINES FROM
C--HUB TO TIP AT TEN PERCENT CHORD INTERVALS
      DO 50 JN=1,NBLPL
      DELZ= 0.1*(ZBL(NPPP,JN)-ZBL(1,JN))
      ZPC(1,JN)= ZBL(1,JN)
      DO 10 KN=2,11
   10 ZPC(KN,JN)= ZPC(KN-1,JN)+DELZ
C--CALCULATE R COORDINATES AND ANGLES WITH RESPECT TO Z AXIS AT
C--INTERSECTION POINTS
      CALL SPLINT(ZBL(1,JN),RBL(1,JN),NPPP,ZPC(1,JN),11,RPC(1,JN),
     1ANGZ(1,JN))
      DO 20 KN=1,11
   20 ANGZ(KN,JN) = ATAN(ANGZ(KN,JN))
C--CALCULATE ARC LENGTH ALONG INPUT LINES USING INPUT POINTS
      SZRBL(1)= 0.
      DO 30 IN=2,NPPP
   30 SZRBL(IN) = SZRBL(IN-1)+SQRT((ZBL(IN,JN)-ZBL(IN-1,JN))**2
     1+(RBL(IN,JN)-RBL(IN-1,JN))**2)
C--CALCULATE ARC LENGTH ALONG INPUT LINES USING POINTS AT TEN
C--PERCENT OF CHORD
      SZRPC(1)= 0.
      DO 40 KN=2,11
   40 SZRPC(KN) = SZRPC(KN-1)+SQRT((ZPC(KN,JN)-ZPC(KN-1,JN))**2
     1+(RPC(KN,JN)-RPC(KN-1,JN))**2)
C--CALCULATE THETA AND CHANGE OF THETA WITH ARC LENGTH ALONG INPUT LINES
   50 CALL SPLINT(SZRBL,THBL(1,JN),NPPP,SZRPC,11,THPC(1,JN),DTHDSP(1,JN)
     1)
C
C--CALCULATE GRADIENT OF THETA WITH RESPECT TO DISTANCE UP TEN PERCENT
C--CHORD ZPC,RPC LINES
C
      DO 80 KN=1,11
C--CALCULATE SLOPES AND ANGLES WITH RESPECT TO R AXIS ALONG THE
C--TEN PERCENT CHORD, HUB-TIP LINES
      DO 60 JN=1,NBLPL
      ZTEM(JN)= ZPC(KN,JN)
      RTEM(JN)= RPC(KN,JN)
   60 TTEM(JN)= THPC(KN,JN)
      CALL SPLINE(RTEM,ZTEM,NBLPL,DYDX,DYDX2)
      SZRPC(1)= 0.
      ANGR(KN,1)= ATAN(DYDX(1))
      DO 70 JN=2,NBLPL
      SZRPC(JN) = SZRPC(JN-1)+SQRT((RPC(KN,JN)-RPC(KN,JN-1))**2
     1+(ZPC(KN,JN)-ZPC(KN,JN-1))**2)
   70 ANGR(KN,JN)= ATAN(DYDX(JN))
C--CALCULATE CHANGE OF THETA WITH ARC LENGTH ALONG HUB-TIP LINES
```

```
      CALL SPLINE(SZRPC,TTEM,NBLPL,DYDX,DYDX2)
      DO 80 JN=1,NBLPL
   80 DTHDTP(KN,JN)= DYDX(JN)
C
C--CALCULATE DTHDZ AND DTHDR FROM DTHDSP AND DTHDTP AT POINTS OF
C--INTERSECTION OF INPUT LINES AND HUB-TIP LINES
C
      DO 90 JN=1,NBLPL
      DO 90 KN=1,11
      COSAB= COS(ANGZ(KN,JN)+ANGR(KN,JN))
      DTHDZ(KN,JN)= (DTHDSP(KN,JN)*COS(ANGR(KN,JN))-DTHDTP(KN,JN)*SIN(
     1ANGZ(KN,JN)))/COSAB
   90 DTHDR(KN,JN)= (-DTHDSP(KN,JN)*SIN(ANGR(KN,JN))+DTHDTP(KN,JN)*COS(
     1ANGZ(KN,JN)))/COSAB
C
C--INTERPOLATE TO OBTAIN DTHDZ AND DTHDR AT THE POINTS OF THE ORTHOGONAL
C--MESH
C--ROTATE DTDZOM AND DTDROM ON ORTHOGONAL MESH TO OBTAIN DTHDS AND DTHDT
C--THE GRADIENTS OF THETA IN THE S AND T DIRECTIONS
C
      II = 1
      JJ = 1
      DO 100 J=1,MHTP1
      ILEJ = ILE(J)
      ITEJ = ITE(J)
      DO 100 I=ILEJ,ITEJ
      CALL LININT(ZPC,RPC,DTHDZ,11,NBLPL,11,50,ZOM(I,J),ROM(I,J),
     1DTDZOM,II,JJ)
      CALL LININT(ZPC,RPC,DTHDR,11,NBLPL,11,50,ZOM(I,J),ROM(I,J),
     1DTDROM,II,JJ)
      DTHDS(I,J) = DTDZOM*CPHI(I,J)+DTDROM*SPHI(I,J)
  100 DTHDT(I,J) = DTDROM*CPHI(I,J)-DTDZOM*SPHI(I,J)
      RETURN
      END
```

```
      SUBROUTINE THIKOM
C
C--THIKOM CALCULATES THE BLADE THICKNESS IN THE THETA DIRECTION AT
C--THE POINTS OF THE ORTHOGONAL MESH
C
      COMMON/INPUTT/GAM,AR,MSFL,OMEGA,REDFAC,VELTOL,FNEW,DNEW,MBI,MBO,
     1   MM,MHT,NBL,NHUB,NTIP,NIN,NOUT,NBLPL,NPPP,NOSTAT,NSL,LSFR,
     2   LTPL,LAMVT,IMESH,ISLINE,ISTATL,IPLOT,ISUPER,ITSCN,IDEBUG,
     3   ZOMIN,ZOMBI,ZOMBO,ZOMOUT,ZHIN,ZTIN,ZHOUT,ZTOUT,ZHUB(50),
     4   RHUB(50),ZTIP(50),RTIP(50),SFIN(50),RADIN(50),TIP(50),PRIP(50),
     5   LAMIN(50),VTHIN(50),SFOUT(50),RADOUT(50),PROP(50),LOSOUT(50),
     6   LAMOUT(50),VTHOUT(50),ZHST(50),ZTST(50),FLFR(50),
     7   ZBL(50,50),RBL(50,50),THBL(50,50),TNBL(50,50)
      COMMON/CALCON/MMM1,MHTP1,CP,EXPON,TGROG,PITCH,CURVHI,CURVTI,
     1   CURVHO,CURVTO,RHIN,RTIN,RHOUT,RTOUT,RLEH,RLET,RTEH,RTET,
     2   ZLE(50),RLE(50),ZTE(50),RTE(50),ZLEOM(101),RLEOM(101),
     3   SLEOM(101),THLEOM(101),ZTEOM(101),RTEOM(101),STEOM(101),
     4   THTEOM(101),ILE(101),ITE(101),ZOM(100,101),ROM(100,101),
     5   SOM(100,101),TOM(100,101),BTH(100,101),DTHDS(100,101),
     6   DTHDT(100,101),PLOSS(100,101),CPHI(100,101),SPHI(100,101)
```

```
      DIMENSION DIST(50),DTDS(50),ANG(50),DRTHBL(50,50)
C
C--CALCULATE BLADE THICKNESS IN THE THETA DIRECTION FROM INPUT
C--THICKNESSES NORMAL TO MEAN CAMBER LINE
C
      DO 20 JN=1,NBLPL
      DIST(1) = 0.
      DO 10 IN=2,NPPP
   10 DIST(IN) = DIST(IN-1)+SQRT((ZBL(IN,JN)-ZBL(IN-1,JN))**2+
     1(RBL(IN,JN)-RBL(IN-1,JN))**2)
      CALL SPLINE(DIST,THBL(1,JN),NPPP,DTDS,ANG)
      DO 20 IN=1,NPPP
      ANG(IN) = ATAN(RBL(IN,JN)*DTDS(IN))
   20 DRTHBL(IN,JN) = TNBL(IN,JN)/COS(ANG(IN))
C
C--INTERPOLATE TO OBTAIN BLADE THICKNESS IN THETA DIRECTION AT THE
C--POINTS OF THE ORTHOGONAL MESH
C
      II = 1
      JJ = 1
      DO 30 J=1,MHTP1
      ILEJ = ILE(J)
      ITEJ = ITE(J)
      DO 30 I=ILEJ,ITEJ
      CALL LININT(ZBL,RBL,DRTHBL,NPPP,NBLPL,50,50,ZOM(I,J),ROM(I,J),
     1DRBTH,II,JJ)
      BTH(I,J) = BTH(I,J)-DRBTH/ROM(I,J)
   30 CONTINUE
      RETURN
      END



      SUBROUTINE LOSSOM
C
C--LOSSOM COMPUTES THE RATIO OF ACTUAL TO IDEAL RELATIVE TOTAL PRESSURE
C--DOWNSTREAM OF THE BLADE, AND THEN DISTRIBUTES THIS LOSS LINEARLY
C--THROUGH THE BLADE ROW FROM TRAILING TO LEADING EDGE BY ADDING
C--ONTO THE BLADE THICKNESS AT THE ORTHOGONAL MESH POINTS
C
      COMMON SRW,SRE,ITER,IEND,NREAD,NWRIT
      COMMON/INPUTT/GAM,AR,MSFL,OMEGA,REDFAC,VELTOL,FNEW,DNEW,MBI,MBO,
     1   MM,MHT,NBL,NHUB,NTIP,NIN,NOUT,NBLPL,NPPP,NOSTAT,NSL,LSFR,
     2   LTPL,LAMVT,IMESH,ISLINE,ISTATL,IPLOT,ISUPER,ITSON,IDEBUG,
     3   ZOMIN,ZOMBI,ZOMBO,ZOMOUT,ZHIN,ZTIN,ZHOUT,ZTOUT,ZHUB(50),
     4   RHUB(50),ZTIP(50),RTIP(50),SFIN(50),RADIN(50),TIP(50),PRIP(50),
     5   LAMIN(50),VTHIN(50),SFOUT(50),RADOUT(50),PROP(50),LOSOUT(50),
     6   LAMOUT(50),VTHOUT(50),ZHST(50),ZTST(50),FLFR(50),
     7   ZBL(50,50),RBL(50,50),THBL(50,50),TNBL(50,50)
      COMMON/CALCON/MMM1,MHTP1,CP,EXPON,TGROG,PITCH,CURVHI,CURVTI,
     1   CURVHO,CURVTO,RHIN,RTIN,RHOUT,RTOUT,RLEH,RLET,RTEH,RTET,
     2   ZLE(50),RLE(50),ZTE(50),RTE(50),ZLEOM(101),RLEOM(101),
     3   SLEOM(101),THLEOM(101),ZTEOM(101),RTEOM(101),STEOM(101),
     4   THTEOM(101),ILE(101),ITE(101),ZOM(100,101),ROM(100,101),
     5   SOM(100,101),TOM(100,101),BTH(100,101),DTHDS(100,101),
     6   DTHDT(100,101),PLOSS(100,101),CPHI(100,101),SPHI(100,101)
      DIMENSION RELTOP(50),SF(101),GRAD(101),PRATIO(101)
      REAL LAMDAF,LAMIN,LAMOUT,LAMDAI,LOSOUT
```

```
C
C--CALCULATE ACTUAL-TO-IDEAL RELATIVE TOTAL PRESSURE RATIO
C--ON DOWNSTREAM INPUT BOUNDARY
C
      DO 20 J=1,NOUT
      TINP = TIPF(SFOUT(J))
      LAMDAI= LAMDAF(SFOUT(J),1,1)
      TOP = TOPF(SFOUT(J))
      PRINP = RHOIPF(SFOUT(J))*AR*TINP
      IF (LTPL.EQ.1) GO TO 10
      RELTOP(J) = PROP(J)/PRINP*(TINP/TOP)**(GAM*EXPON)
      GO TO 20
   10 RELTOP(J) = 1.-LOSOUT(J)
   20 CONTINUE
C
C--DISTRIBUTE LOSS ON ORTHOGONAL MESH WITHIN BLADES, ONE ORTHOGONAL
C--MESH LINE AT A TIME
C
      DO 30 J=1,MHTP1
   30 SF(J)= (ROM(MM,J)**2-ROM(MM,1)**2)/(ROM(MM,MHTP1)**2-ROM(MM,1)**2)
      CALL SPLINT(SFOUT,RELTOP,NOUT,SF,MHTP1,PRATIO,GRAD)
      DO 40 J=1,MHTP1
      ILEJ = ILE(J)
      SLENTH = STEOM(J)-SLEOM(J)
      DO 40 I=ILEJ,MM
      DELS = AMIN1(SLENTH,SOM(I,J)-SLEOM(J))
      PLOSS(I,J) = (1.-PRATIO(J))*DELS/SLENTH
   40 BTH(I,J) = BTH(I,J)*(1.-PLOSS(I,J))
      IF (IDEBUG.LE.0) RETURN
      WRITE(NWRIT,1010)
      WRITE(NWRIT,1000) ((I,J,SOM(I,J),TOM(I,J),BTH(I,J),DTHDT(I,J),
     1PLOSS(I,J),CPHI(I,J),SPHI(I,J),I=1,MM),J=1,MHTP1)
      RETURN
 1000 FORMAT(2I6,7G16.6)
 1010 FORMAT(1H1///35X,47HCONSTANT  QUANTITIES  ON  THE  ORTHOGONAL  ME
     1SH/5X,1HI,5X,1HJ,7X,3HSOM,13X,3HTOM,13X,3HBTH,12X,5HDTHDT,11X,
     25HPLOSS,11X,4HCPHI,12X,4HSPHI)
      END
```

```
      SUBROUTINE MEPLOT
C
C--MEPLOT PLOTS THE BLADE GEOMETRY AND THE GENERATED ORTHOGONAL MESH
C
      COMMON/INPUTT/GAM,AR,MSFL,OMEGA,REDFAC,VELTOL,FNEW,DNEW,MBI,MBO,
     1   MM,MHT,NBL,NHUB,NTIP,NIN,NOUT,NBLPL,NPPP,NOSTAT,NSL,LSFR,
     2   LTPL,LAMVT,IMESH,ISLINE,ISTATL,IPLOT,ISUPER,ITSON,IDEBUG,
     3   ZOMIN,ZOMBI,ZOMBO,ZOMOUT,ZHIN,ZTIN,ZHOUT,ZTOUT,ZHUB(50),
     4   RHUB(50),ZTIP(50),RTIP(50),SFIN(50),RADIN(50),TIP(50),PRIP(50),
     5   LAMIN(50),VTHIN(50),SFOUT(50),RADOUT(50),PROP(50),LOSOUT(50),
     6   LAMOUT(50),VTHOUT(50),ZHST(50),ZTST(50),FLFR(50),
     7   ZBL(50,50),RBL(50,50),THBL(50,50),TNBL(50,50)
      COMMON/CALCON/MMM1,MHTP1,CP,EXPON,TGROG,PITCH,CURVHI,CURVTI,
     1   CURVHO,CURVTO,RHIN,RTIN,RHOUT,RTOUT,RLEH,RLET,RTEH,RTET,
     2   ZLE(50),RLE(50),ZTE(50),RTE(50),ZLEOM(101),RLEOM(101),
     3   SLEOM(101),THLEOM(101),ZTEOM(101),RTEOM(101),STEOM(101),
```

```
     4   THTEOM(101),ILE(101),ITE(101),ZOM(100,101),ROM(100,101),
     5   SOM(100,101),TOM(100,101),BTH(100,101),DTHDS(100,101),
     6   DTHDT(100,101),PLOSS(100,101),CPHI(100,101),SPHI(100,101)
      COMMON/PLTCOM/ZLRNG,ZRRNG,RBRNG,RTRNG,ZHPLT(100),RHPLT(100),
     1   ZSPLT(100),RSPLT(100),ZLPLT(100),RLPLT(100),ZTPLT(100),
     2   RTPLT(100)
      DIMENSION TITL1(15),TITL2(10),TITL3(3),TITL4(3),ZTEM(101),
     1RTEM(101)
      DATA TITL1/'HUB,',' SHR','OUD,',' AND',' BLA','CE B','OUND','ARIE'
     1,'S$C1','$RBI','N ME','RIDI','ONAL',' PLA','NE  '/
      DATA TITL2/'ORTH','OGON','AL M','ESH$','C1$L','2IN ','MERI','DION'
     1,'AL P','LANE'/
      DATA TITL3/'Z  D','IREC','TION'/
      DATA TITL4/'R  D','IREC','TION'/
      DATA SYM/'X'/
      DATA SYN/'O'/
      IF (IPLOT.LE.0) RETURN
C
C--OBTAIN PLOT BOUNDARIES, AND SCALE THE PLOT
      CALL PTBDRY
C
C--PLOT BLADE GEOMETRY AND PLOT ORTHOGONAL MESH
      CALL LRMRGN(1.0,1.0,2.0,1.0)
      CALL LRANGE(ZLRNG,ZRRNG,RBRNG,RTRNG)
      CALL LRGRID(-1,-1,1.0,1.0)
      IPLT= 1
      CALL LRLEGN(TITL1,60,0,1.3,0.7,0.0)
   10 IF (IPLT.EQ.2) CALL LRLEGN(TITL2,40,0,3.4,0.7,0.0)
      CALL LRCHSZ(2)
      CALL LRLEGN(TITL3,12,0,4.5,1.5,0.0)
      CALL LRLEGN(TITL4,12,1,0.4,4.5,0.0)
      CALL LRCHSZ(4)
      CALL LRCURV(ZHPLT,RHPLT,100,2,SYM,0.0)
      CALL LRCURV(ZSPLT,RSPLT,100,2,SYM,0.0)
      CALL LRCURV(ZLPLT,RLPLT,100,2,SYM,0.0)
      CALL LRCURV(ZTPLT,RTPLT,100,2,SYM,0.0)
      IF (IPLT.EQ.2) GO TO 20
      CALL LRCURV(ZHUB,RHUB,NHUB,4,SYM,0.0)
      CALL LRCURV(ZTIP,RTIP,NTIP,4,SYM,0.0)
      DO 15 JN=1,NBLPL
   15 CALL LRCURV(ZBL(1,JN),RBL(1,JN),NPPP,2,SYM,0.0)
      CALL LRCURV(ZLE,RLE,NBLPL,3,SYN,0.0)
      CALL LRCURV(ZTE,RTE,NBLPL,3,SYN,1.0)
      IPLT= 2
      GO TO 10
C--PLOT VERTICAL MESH LINES
   20 DO 40 I=1,MM
      ZTEM(1)= ZOM(I,1)
      RTEM(1)= ROM(I,1)
      DO 30 J=2,MHTP1
      ZTEM(J)= ZOM(I,J)
   30 RTEM(J)= ROM(I,J)
   40 CALL LRCURV(ZTEM,RTEM,MHTP1,2,SYM,0.0)
C--PLOT HORIZONTAL MESH LINES
      EOP= 0.0
      DO 50 J=2,MHT
      IF (J.EQ.MHT) EOP=1.0
   50 CALL LRCURV(ZOM(1,J),ROM(1,J),MM,2,SYM,EOP)
      CALL LRCURV(ZTEM,RTEM,0,1,SYM,1.0)
      RETURN
      END
```

```
      SUBROUTINE PTBDRY
C
C--PTBDRY OBTAINS THE HUB AND SHROUD AND BLADE LEADING AND TRAILING EDGE
C--BOUNDARIES FOR PLOTTING, AND SCALES THE PLOT
C
      COMMON/INPUTT/GAM,AR,MSFL,OMEGA,REDFAC,VELTOL,FNEW,DNEW,MBI,MBO,
     1   MM,MHT,NBL,NHUB,NTIP,NIN,NOUT,NBLPL,NPPP,NCSTAT,NSL,LSFR,
     2   LTPL,LAMVT,IMESH,ISLINE,ISTATL,IPLOT,ISUPER,ITSON,IDEBUG,
     3   ZOMIN,ZOMBI,ZOMBO,ZOMOUT,ZHIN,ZTIN,ZHOUT,ZTOUT,ZHUB(50),
     4   RHUB(50),ZTIP(50),RTIP(50),SFIN(50),RADIN(50),TIP(50),PRIP(50),
     5   LAMIN(50),VTHIN(50),SFOUT(50),RADOUT(50),PROP(50),LOSOUT(50),
     6   LAMOUT(50),VTHOUT(50),ZHST(50),ZTST(50),FLFR(50),
     7   ZBL(50,50),RBL(50,50),THBL(50,50),TNBL(50,50)
      COMMON/CALCON/MMM1,MHTP1,CP,EXPON,TGROG,PITCH,CURVHI,CURVTI,
     1   CURVHO,CURVTO,RHIN,RTIN,RHOUT,RTOUT,RLEH,RLET,RTEH,RTET,
     2   ZLE(50),RLE(50),ZTE(50),RTE(50),ZLEOM(101),RLEOM(101),
     3   SLEOM(101),THLEOM(101),ZTEOM(101),RTEOM(101),STEOM(101),
     4   THTEOM(101),ILE(101),ITE(101),ZOM(100,101),ROM(100,101),
     5   SOM(100,101),TOM(100,101),BTH(100,101),DTHDS(100,101),
     6   DTHDT(100,101),PLOSS(100,101),CPHI(100,101),SPHI(100,101)
      COMMON/PLTCOM/ZLRNG,ZRRNG,RBRNG,RTRNG,ZHPLT(100),RHPLT(100),
     1   ZSPLT(100),RSPLT(100),ZLPLT(100),RLPLT(100),ZTPLT(100),
     2   RTPLT(100)
      DIMENSION AAA(100)
C
C--OBTAIN PLOT POINTS ON HUB
C
      DELZ = (ZHUB(NHUB)-ZHUB(1))/99.
      ZHPLT(1) = ZHUB(1)
      DO 10 I=2,100
   10 ZHPLT(I) = ZHPLT(I-1)+DELZ
      CALL SPLINT(ZHUB,RHUB,NHUB,ZHPLT,100,RHPLT,AAA)
C
C--OBTAIN PLOT POINTS ON SHROUD
C
      DELZ = (ZTIP(NTIP)-ZTIP(1))/99.
      ZSPLT(1) = ZTIP(1)
      DO 20 I=2,100
   20 ZSPLT(I) = ZSPLT(I-1)+DELZ
      CALL SPLINT(ZTIP,RTIP,NTIP,ZSPLT,100,RSPLT,AAA)
C
C--OBTAIN PLOT POINTS UP BLADE LEADING EDGE
C
      DELR = (RLET-RLEH)/99.
      RLPLT(1) = RLEH
      RLPLT(100) = RLET
      DO 30 J=2,99
   30 RLPLT(J) = RLPLT(J-1)+DELR
      CALL SPLINT(RLE,ZLE,NBLPL,RLPLT,100,ZLPLT,AAA)
C
C--OBTAIN PLOT POINTS UP BLADE TRAILING EDGE
C
      DELR = (RTET-RTEH)/99.
      RTPLT(1) = RTEH
      RTPLT(100) = RTET
      DO 40 J=2,99
   40 RTPLT(J) = RTPLT(J-1)+DELR
      CALL SPLINT(RTE,ZTE,NBLPL,RTPLT,100,ZTPLT,AAA)
C
C--CALCULATE THE RANGE OF THE PLOT
C
```

```
      ZLRNG = AMIN1(ZHUB(1),ZTIP(1))
      ZRRNG = AMAX1(ZHUB(NHUB),ZTIP(NTIP))
      DO 48 J=1,MHTP1
      ZLRNG = AMIN1(ZLRNG,ZOM(1,J))
   48 ZRRNG = AMAX1(ZRRNG,ZOM(MM,J))
      DELZ = ZRRNG-ZLRNG
      ZLRNG = ZLRNG-0.05*DELZ
      ZRRNG = ZRRNG+0.05*DELZ
      RBRNG = RHUB(1)
      DO 50 I=2,NHUB
   50 RBRNG = AMIN1(RBRNG,RHUB(I))
      RTRNG = RTIP(1)
      DO 60 I=2,NTIP
   60 RTRNG = AMAX1(RTRNG,RTIP(I))
      DELR = RTRNG-RBRNG
      RBRNG = RBRNG-0.05*DELR
      RTRNG = RTRNG+0.05*DELR
C
C--CHOOSE MAXIMUM RANGE, AND EXPAND RANGE IN THE OTHER DIRECTION
C
      DMD2 = 1.1*ABS(DELZ-DELR)/2.
      IF (DELR.GT.DELZ) GO TO 70
      RTRNG = RTRNG+DMD2
      RBRNG = RBRNG-DMD2
      RETURN
   70 ZRRNG = ZRRNG+DMD2
      ZLRNG = ZLRNG-DMD2
      RETURN
      END



      SUBROUTINE VBDRY(LOC,TIPF,RHOIPF,LAMDAF)
C
C--VBDRY CALCULATES THE DISTRIBUTION OF STREAM FUNCTION ALONG THE
C--UPSTREAM AND DOWNSTREAM BOUNDARIES OF THE ORTHOGONAL MESH
C
      COMMON SRW,SRE,ITER,IEND,NREAD,NWRIT
      COMMON/INPUTT/GAM,AR,MSFL,OMEGA,REDFAC,VELTOL,FNEW,DNEW,MBI,MBO,
     1   MM,MHT,NBL,NHUB,NTIP,NIN,NOUT,NBLPL,NPPP,NOSTAT,NSL,LSFR,
     2   LTPL,LAMVT,IMESH,ISLINE,ISTATL,IPLOT,ISUPER,ITSON,IDEBUG,
     3   ZOMIN,ZOMBI,ZOMBO,ZOMOUT,ZHIN,ZTIN,ZHOUT,ZTOUT,ZHUB(50),
     4   RHUB(50),ZTIP(50),RTIP(50),SFIN(50),RADIN(50),TIP(50),PRIP(50),
     5   LAMIN(50),VTHIN(50),SFOUT(50),RADOUT(50),PROP(50),LOSOUT(50),
     6   LAMOUT(50),VTHOUT(50),ZHST(50),ZTST(50),FLFR(50),
     7   ZBL(50,50),RBL(50,50),THBL(50,50),TNBL(50,50)
      COMMON/CALCON/MMM1,MHTP1,CP,EXPON,TGROG,PITCH,CURVHI,CURVTI,
     1   CURVHO,CURVTO,RHIN,RTIN,RHOUT,RTOUT,RLEH,RLET,RTEH,RTET,
     2   ZLE(50),RLE(50),ZTE(50),RTE(50),ZLEOM(101),RLEOM(101),
     3   SLEOM(101),THLEOM(101),ZTEOM(101),RTEOM(101),STEOM(101),
     4   THTEOM(101),ILE(101),ITE(101),ZOM(100,101),ROM(100,101),
     5   SOM(100,101),TOM(100,101),BTH(100,101),DTHDS(100,101),
     6   DTHDT(100,101),PLOSS(100,101),CPHI(100,101),SPHI(100,101)
      COMMON/VARCOM/A(4,100,101),UOM(100,101),K(100,101),RHO(100,101),
     1   WSUBS(100,101),WSUBT(100,101),WSUBZ(100,101),WSUBR(100,101),
     2   WSUBM(100,101),WTH(100,101),VTH(100,101),W(100,101),
     3   ALPHA(100,101),BETA(100,101),WWCR(100,101),CURV(100,101),
     4   WLSURF(100,101),WTSURF(100,101),CAMP(100,101),SAMP(100,101),
```

```
     5   RHOAV(100,101),DELRHO(100,101),FR(100,101),DFCM(100,101),
     6   XIOM(100,101),ZETOM(100,101),DLDU(100,101)
      DIMENSION TIPBDY(101),RHOIP(101),LAMBDA(101),AA(101),BB(101),
     1   CCA(101),CCB(101),WBDRY(101),BDY(4)
      REAL LAMBDA,LAMDAF,MSFL
      LOGICAL REPEAT
      EXTERNAL TIPF,RHOIPF,LAMDAF
      DATA BDY/4HINLE,4HT    ,4HOUTL,4HET  /
C
C--SET INITIAL WHUB AND DELMAX
C
      IL= 1
      IF (LOC.GT.1) IL=3
      CURVH = CURVHI
      CURVT = CURVTI
      IF(LOC.EQ.1) GO TO 10
      CURVH = CURVHO
      CURVT = CURVTO
   10 JZ = 1
      RMEAN = (ROM(LOC,1)+ROM(LOC,MHTP1))/2.
      RHOIP(1)= RHOIPF(.5)
      WSM= MSFL/RHOIP(1)/RMEAN/PITCH/TOM(LOC,MHTP1)
      WTHETA= LAMDAF(.5,LOC,1)/RMEAN-OMEGA*RMEAN
      WHUB = SQRT(WSM**2+WTHETA**2)
      DELMAX = WHUB/20.
      RTOLER = 1.E-4
C
C--CALCULATE INITIAL ESTIMATE OF TIP,RHOIP,AND LAMBDA
C
      RH2 = ROM(LOC,1)**2
      DELR2 = ROM(LOC,MHTP1)**2-RH2
      DO 20 J=1,MHTP1
      WFLF = (ROM(LOC,J)**2-RH2)/DELR2
      UOM(LOC,J) = WFLF
      TIPBDY(J)= TIPF(WFLF)
      RHOIP(J)= RHOIPF(WFLF)
   20 LAMBDA(J)= LAMDAF(WFLF,LOC,J)
      NCOUNT = 0
C
C--CALCULATE COEFFICIENTS A, B, AND C FOR THE VELOCITY GRADIENT EQUATION
C
   30 DO 40 J=1,MHTP1
      AA(J) = CURVH+TOM(LOC,J)/TOM(LOC,MHTP1)*(CURVT-CURVH)
      OMR2 = OMEGA*ROM(LOC,J)**2
      BB(J) = -(LAMBDA(J)-OMR2)/ROM(LOC,J)**2*(AA(J)*(LAMBDA(J)-OMR2)+
     1   (LAMBDA(J)+OMR2)/ROM(LOC,J)*CPHI(LOC,J))
   40 CONTINUE
      DO 50 J=1,MHT
      CCA(J)= CP*(TIPBDY(J+1)-TIPBDY(J))-OMEGA*(LAMBDA(J+1)-LAMBDA(J))
   50 CCB(J) = (CP-AR)*(TIPBDY(J+1)-TIPBDY(J))-AR/(RHOIP(J)+RHOIP(J+1))*
     1(TIPBDY(J)+TIPBDY(J+1))*(RHOIP(J+1)-RHOIP(J))
C
C--SOLVE THE VELOCITY GRADIENT EQUATION ALONG THE BOUNDARY
C
      REPEAT = .FALSE.
      UOM(LOC,1) = 0.
   60 IND= 1
   70 WBDRY(1) = WHUB
      NCOUNT = NCOUNT+1
```

```
      WSQ= WBDRY(1)**2
      TWLMR= 2.*OMEGA*LAMBDA(1)-(OMEGA*ROM(LOC,1))**2
      TTIP= 1.-(WSQ+TWLMR)/CP/TIPBDY(1)/2.
      IF (TTIP.LT.0.) GO TO 100
      RHOW = RHOIP(1)*TTIP**EXPON*WBDRY(1)
      SINBTA = (LAMBDA(1)/ROM(LOC,1)-OMEGA*ROM(LOC,1))/WBDRY(1)
      IF (ABS(SINBTA).GT.1.) GO TO 90
      COSBTA = SQRT(1.-SINBTA**2)
      RVA = RHOW*COSBTA*ROM(LOC,1)*PITCH
      DO 80 J=1,MHT
      DELTA= TOM(LOC,J+1)-TOM(LOC,J)
      CC = CCA(J)-CCB(J)*TTIP
      WAS = WBDRY(J)*(1.+AA(J)*DELTA)+BB(J)/WBDRY(J)*DELTA+CC/WBDRY(J)
      WSQ = WAS**2
      TWLMR = 2.*OMEGA*LAMBDA(J+1)-(OMEGA*ROM(LOC,J+1))**2
      TTIP = 1.-(WSQ+TWLMR)/CP/TIPBDY(J+1)/2.
      CC = CCA(J)-CCB(J)*TTIP
      WASS = WBDRY(J)+AA(J+1)*WAS*DELTA+BB(J+1)/WAS*DELTA+CC/WAS
      WBDRY(J+1) = (WAS+WASS)/2.
      WSQ = WBDRY(J+1)**2
      TTIP= 1.-(WSQ+TWLMR)/CP/TIPBDY(J+1)/2.
      IF (TTIP.LT.0.) GO TO 100
      RHOW = RHOIP(J+1)*TTIP**EXPON*WBDRY(J+1)
      SINBTA = (LAMBDA(J+1)/ROM(LOC,J+1)-OMEGA*ROM(LOC,J+1))/WBDRY(J+1)
      IF (ABS(SINBTA).GT.1.) GO TO 90
      COSBTA = SQRT(1.-SINBTA**2)
      RVAS= RHOW*COSBTA*ROM(LOC,J+1)*PITCH
      UOM(LOC,J+1) = (RVA+RVAS)*DELTA/2.+UOM(LOC,J)
   80 RVA= RVAS
C
C--CHECK CONTINUITY AND ESTIMATE NEW VALUE FOR W AT HUB
C
      IF(IND.GE.6.AND.ABS(MSFL-UOM(LOC,MHTP1)).LE.MSFL*RTOLER) GO TO 120
      CALL CONTIN(WHUB,UOM(LOC,MHTP1),IND,JZ,MSFL,DELMAX)
      IF (IND.LT.10) GO TO 70
      IF (IND.EQ.10) GO TO 120
      GO TO 110
   90 WHUB= WHUB+.5*DELMAX
      IF (NCOUNT.LT.1000) GO TO 60
      GO TO 110
  100 WHUB = WHUB-.5*DELMAX
      IF (NCOUNT.LT.1000) GO TO 60
  110 WRITE(NWRIT,1010) BDY(IL),BDY(IL+1)
      STOP
C
C--SOLUTION OBTAINED.  UPDATE TIP,RHOIP,AND LAMBDA
C
  120 CONTINUE
      DO 130 J=2,MHTP1
      FRAC = UOM(LOC,J)/MSFL
      UOM(LOC,J) = FRAC
      TVAR= TIPF(FRAC)
      IF (ABS(TVAR-TIPBDY(J)).GT.TIPBDY(J)*RTOLER) REPEAT=.TRUE.
      TIPBDY(J)= TVAR
      TVAR = RHOIPF(FRAC)
      IF (ABS(TVAR-RHOIP(J)).GT.RHOIP(J)*RTOLER) REPEAT=.TRUE.
      RHOIP(J) = TVAR
      TVAR = LAMDAF(FRAC,LOC,J)
      IF (ABS(TVAR-LAMBDA(J)).GT.ABS(LAMBDA(J))*RTOLER) REPEAT=.TRUE.
```

```
  130 LAMBDA(J) = TVAR
      WHUB= WBDRY(1)
      IF (REPEAT.AND.NCOUNT.GE.1000) GO TO 110
      IF (REPEAT) GO TO 30
      IF (IND.NE.10) RETURN
      WRITE(NWRIT,1000) BDY(IL),BDY(IL+1),UOM(LOC,MHTP1)
      STOP
 1000 FORMAT (26HL PASSAGE IS CHOKED AT THE,2A4,21H WITH A MASS FLOW OF
     1,G14.6)
 1010 FORMAT (2HL ,2A4,39H BOUNDARY CONDITIONS CANNOT BE OBTAINED)
      END
```

```
      SUBROUTINE INIT
C
C--INIT ASSIGNS INITIAL VALUES TO THE ARRAY VARIABLES
C
      COMMON/INPUTT/GAM,AR,MSFL,OMEGA,REDFAC,VELTOL,FNEW,DNEW,MBI,MBO,
     1  MM,MHT,NBL,NHUB,NTIP,NIN,NOUT,NBLPL,NPPP,NOSTAT,NSL,LSFR,
     2  LTPL,LAMVT,IMESH,ISLINE,ISTATL,IPLOT,ISUPER,ITSON,IDEBUG,
     3  ZOMIN,ZOMBI,ZOMBO,ZOMOUT,ZHIN,ZTIN,ZHOUT,ZTOUT,ZHUB(50),
     4  RHUB(50),ZTIP(50),RTIP(50),SFIN(50),RADIN(50),TIP(50),PRIP(50),
     5  LAMIN(50),VTHIN(50),SFOUT(50),RADOUT(50),PROP(50),LOSOUT(50),
     6  LAMOUT(50),VTHOUT(50),ZHST(50),ZTST(50),FLFR(50),
     7  ZBL(50,50),RBL(50,50),THBL(50,50),TNBL(50,50)
      COMMON/CALCCN/MMM1,MHTP1,CP,EXPON,TGROG,PITCH,CURVHI,CURVTI,
     1  CURVHO,CURVTO,RHIN,RTIN,RHOUT,RTOUT,RLEH,RLET,RTEH,RTET,
     2  ZLE(50),RLE(50),ZTE(50),RTE(50),ZLEOM(101),RLEOM(101),
     3  SLEOM(101),THLEOM(101),ZTEOM(101),RTEOM(101),STEOM(101),
     4  THTEOM(101),ILE(101),ITE(101),ZOM(100,101),ROM(100,101),
     5  SOM(100,101),TOM(100,101),BTH(100,101),DTHDS(100,101),
     6  DTHDT(100,101),PLOSS(100,101),CPHI(100,101),SPHI(100,101)
      COMMON/VARCOM/A(4,100,101),UOM(100,101),K(100,101),RHO(100,101),
     1  WSUBS(100,101),WSUBT(100,101),WSUBZ(100,101),WSUBR(100,101),
     2  WSUBM(100,101),WTH(100,101),VTH(100,101),W(100,101),
     3  ALPHA(100,101),BETA(100,101),WWCR(100,101),CURV(100,101),
     4  WLSURF(100,101),WTSURF(100,101),CAMP(100,101),SAMP(100,101),
     5  RHOAV(100,101),DELRHO(100,101),FR(100,101),DFDM(100,101),
     6  XIOM(100,101),ZETOM(100,101),DLDU(100,101)
      REAL K
      RHOIP = RHOIPF(0.)
      DO 10 J=1,MHTP1
      DO 10 I=1,MM
      WSUBS(I,J) = CPHI(I,J)
      WSUBT(I,J) = -SPHI(I,J)
      WSUBZ(I,J) = 1.
      W(I,J) = 0.
      WTH(I,J) = 0.
      VTH(I,J) = 0.
      RHO(I,J) = RHOIP
      RHOAV(I,J) = RHOIP
      DELRHO(I,J) = 0.
      XIOM(I,J) = 0.
      ZETOM(I,J) = 0.
      FR(I,J) = 0.
      DFDM(I,J) = 0.
```

```
      DLDU(I,J) = 0.
   10 K(I,J) = 0.
      RETURN
      END
```

```
      SUBROUTINE COEF
C
C--COEF CALCULATES COEFFICIENTS, A AND K,
C--FOR THE SYSTEM OF MATRIX EQUATIONS,   A*U=K
C
      COMMON SRW,SRE,ITER,IEND,NREAD,NWRIT
      COMMON/INPUTT/GAM,AR,MSFL,OMEGA,REDFAC,VELTOL,FNEW,DNEW,MBI,MBO,
     1   MM,MHT,NBL,NHUB,NTIP,NIN,NOUT,NBLPL,NPPP,NCSTAT,NSL,LSFR,
     2   LTPL,LAMVT,IMESH,ISLINE,ISTATL,IPLOT,ISUPER,ITSON,IDEBUG,
     3   ZOMIN,ZOMBI,ZOMBO,ZOMOUT,ZHIN,ZTIN,ZHOUT,ZTOUT,ZHUB(50),
     4   RHUB(50),ZTIP(50),RTIP(50),SFIN(50),RADIN(50),TIP(50),PRIP(50),
     5   LAMIN(50),VTHIN(50),SFOUT(50),RADOUT(50),PROP(50),LOSOUT(50),
     6   LAMOUT(50),VTHOUT(50),ZHST(50),ZTST(50),FLFR(50),
     7   ZBL(50,50),RBL(50,50),THBL(50,50),TNBL(50,50)
      COMMON/CALCON/MMM1,MHTP1,CP,EXPON,TGROG,PITCH,CURVHI,CURVTI,
     1   CURVHO,CURVTO,RHIN,RTIN,RHOUT,RTOUT,RLEH,RLET,RTEH,RTET,
     2   ZLE(50),RLE(50),ZTE(50),RTE(50),ZLEOM(101),RLEOM(101),
     3   SLEOM(101),THLEOM(101),ZTEOM(101),RTEOM(101),STEOM(101),
     4   THTEOM(101),ILE(101),ITE(101),ZOM(100,101),ROM(100,101),
     5   SOM(100,101),TOM(100,101),BTH(100,101),DTHDS(100,101),
     6   DTHDT(100,101),PLOSS(100,101),CPHI(100,101),SPHI(100,101)
      COMMON/VARCOM/A(4,100,101),UOM(100,101),K(100,101),RHO(100,101),
     1   WSUBS(100,101),WSUBT(100,101),WSUBZ(100,101),WSUBR(100,101),
     2   WSUBM(100,101),WTH(100,101),VTH(100,101),W(100,101),
     3   ALPHA(100,101),BETA(100,101),WWCR(100,101),CURV(100,101),
     4   WLSURF(100,101),WTSURF(100,101),CAMP(100,101),SAMP(100,101),
     5   RHOAV(100,101),DELRHO(100,101),FR(100,101),DFDM(100,101),
     6   XIOM(100,101),ZETOM(100,101),DLDU(100,101)
      DIMENSION DVTHDR(100,101)
      REAL MSFL,K,KNEW
C--CALCULATE COEFFICIENTS AND CONSTANTS FOR FINITE DIFFERENCE EQUATIONS
      WRITE(NWRIT,1030) ITER
      DCHANG = 0.
      DMAX = -1.E20
      DMIN = 1.E20
      DO 30 J=2,MHT
      H4 = SOM(2,J)-SOM(1,J)
      DO 30 I=2,MMM1
      H1 = TOM(I,J)-TOM(I,J-1)
      H2 = TOM(I,J+1)-TOM(I,J)
      H3 = H4
      H4 = SOM(I+1,J)-SOM(I,J)
      A0 = 2./H1/H2+2./H3/H4
      C1 = H1+H2
      C2 = H3+H4
      D1 = (BTH(I,J+1)-BTH(I,J-1))/BTH(I,J)+(RHO(I,J+1)-RHO(I,J-1))/
     1RHO(I,J)
      D1 = D1/C1+CPHI(I,J)/ROM(I,J)+(SPHI(I+1,J)-SPHI(I-1,J))/C2/
     1CPHI(I,J)
      D2 = (BTH(I+1,J)-BTH(I-1,J))/BTH(I,J)+(RHO(I+1,J)-RHO(I-1,J))/
```

```
     1RHO(I,J)
      D2 = D2/C2+SPHI(I,J)/RCM(I,J)-(SPHI(I,J+1)-SPHI(I,J-1))/C1/
     1CPHI(I,J)
      KNEW = XIOM(I,J)*W(I,J)**2+ZETOM(I,J)
      IF(I.GE.ILE(J).AND.I.LE.ITE(J)) GC TC 10
      KNEW = KNEW+WTH(I,J)/MSFL*BTH(I,J)*RHO(I,J)*WSUBZ(I,J)*DLDU(I,J)
      GC TC 20
   10 DVDRT = (ROM(I+1,J)*VTH(I+1,J)-RCM(I-1,J)*VTH(I-1,J))/C2*SPHI(I,J)
     1+(ROM(I,J+1)*VTH(I,J+1)-ROM(I,J-1)*VTH(I,J-1))/C1*CPHI(I,J)
      DCHANG = AMAX1(DCHANG,ABS(DVDRT-DVTHDR(I,J)))
      DMAX = AMAX1(DMAX,DVDRT)
      DMIN = AMIN1(DMIN,DVDRT)
      DVTHDR(I,J) = DNEW*DVDRT+(1.-DNEW)*DVTHDR(I,J)
      KNEW = KNEW+WTH(I,J)/ROM(I,J)*DVTHDR(I,J)+FR(I,J)
   20 KNEW = KNEW*ROM(I,J)/AO*BTH(I,J)/MSFL*RHO(I,J)/WSUBZ(I,J)
      K(I,J) = KNEW
      A(1,I,J) = (2./H1+D1)/AO/C1
      A(2,I,J) = (2./H2-D1)/AO/C1
      A(3,I,J) = (2./H3+D2)/AO/C2
      A(4,I,J) = (2./H4-D2)/AO/C2
   30 CCNTINUE
      WRITE(NWRIT,1020) DMAX,DMIN,DCHANG
      IF (IDEBUG.LE.0) RETURN
      IF ((ITER/IDEBUG)*IDEBUG.NE.ITER.AND.ITER.NE.1) RETURN
      WRITE(NWRIT,1010)
      DC 40 J=2,MHT
      DC 40 I=2,MMM1
   40 WRITE(NWRIT,1000) I,J,(A(IJ,I,J),IJ=1,4),K(I,J)
      RETURN
 1000 FORMAT (2I6,5G16.6)
 1010 FORMAT (1H1///24X,57HCOEFFICIENTS  OF  MATRIX  EQUATION  FOR  STR
     1EAM  FUNCTION/5X,1HI,5X,1HJ,6X,4HA(1),12X,4HA(2),12X,4HA(3),12X,
     24HA(4),13X,1HK)
 1020 FORMAT (///5X,37HMAXIMUM CALCULATED VALUE OF DVTHDR  =,G13.5/5X,
     137HMINIMUM CALCULATED VALUE OF DVTHDR  =,G13.5/5X,37HMAXIMUM CALCU
     2LATED CHANGE IN DVTHDR =,G13.5)
 1030 FORMAT (/////14H ITERATION NO.,I3,2H :)
      END




      SUBROUTINE SOR
C
C--SOR SOLVES THE SET OF MATRIX EQUATIONS,  A*U=K
C--BY THE SUCCESSIVE OVERRELAXATION TECHNIQUE
C
      COMMON SRW,SRE,ITER,IEND,NREAD,NWRIT
      COMMON/INPUTT/GAM,AR,MSFL,OMEGA,REDFAC,VELTOL,FNEW,DNEW,MBI,MBO,
     1   MM,MHT,NBL,NHUB,NTIP,NIN,NOUT,NBLPL,NPPP,NOSTAT,NSL,LSFR,
     2   LTPL,LAMVT,IMESH,ISLINE,ISTATL,IPLCT,ISUPER,ITSON,IDEBUG,
     3   ZOMIN,ZOMBI,ZOMBO,ZOMOUT,ZHIN,ZTIN,ZHOUT,ZTOUT,ZHUB(50),
     4   RHUB(50),ZTIP(50),RTIP(50),SFIN(50),RADIN(50),TIP(50),PRIP(50),
     5   LAMIN(50),VTHIN(50),SFOUT(50),RADOUT(50),PROP(50),LOSOUT(50),
     6   LAMOUT(50),VTHOUT(50),ZHST(50),ZTST(50),FLFR(50),
     7   ZBL(50,50),RBL(50,50),THBL(50,50),TNBL(50,50)
      COMMON/CALCON/MMM1,MHTP1,CP,EXPON,TGROG,PITCH,CURVHI,CURVTI,
     1   CURVHO,CURVTO,RHIN,RTIN,RHOUT,RTOUT,RLEH,RLET,RTEH,RTET,
```

```
     2    ZLE(50),RLE(50),ZTE(50),RTE(50),ZLEOM(101),RLEOM(101),
     3    SLEOM(101),THLEOM(101),ZTEOM(101),RTEOM(101),STEOM(101),
     4    THTEOM(101),ILE(101),ITE(101),ZOM(100,101),ROM(100,101),
     5    SOM(100,101),TOM(100,101),BTH(100,101),DTHDS(100,101),
     6    DTHDT(100,101),PLOSS(100,101),CPHI(100,101),SPHI(100,101)
      COMMON/VARCOM/A(4,100,101),UOM(100,101),K(100,101),RHO(100,101),
     1    WSUBS(100,101),WSUBT(100,101),WSUBZ(100,101),WSUBR(100,101),
     2    WSUBM(100,101),WTH(100,101),VTH(100,101),W(100,101),
     3    ALPHA(100,101),BETA(100,101),WWCR(100,101),CURV(100,101),
     4    WLSURF(100,101),WTSURF(100,101),CAMP(100,101),SAMP(100,101),
     5    RHOAV(100,101),DELRHO(100,101),FR(100,101),DFDM(100,101),
     6    XIOM(100,101),ZETOM(100,101),DLDU(100,101)
      DIMENSION UVERT(101,2)
      REAL K,LMAX,LMIN
      IF (ITER.GT.1) GO TO 70
C
C--STORE U BOUNDARY VALUES AND SET BOUNDARY VALUE TO ZERO
C--TO CALCULATE OPTIMUM ORF
      DO 10 I=2,MMM1
      UOM(I,1) = 0.
   10 UOM(I,MHTP1) = 0.
      DO 20 J=2,MHT
      UVERT(J,1) = UOM(1,J)
      UOM(1,J) = 0.
      UVERT(J,2) = UOM(MM,J)
      UOM(MM,J) = 0.
      DO 20 I=2,MMM1
   20 UOM(I,J) = 1.
C
C--CALCULATE OPTIMUM ORF
   30 LMAX = 0.
      LMIN = 1.
      DO 40 J=2,MHT
      DO 40 I=2,MMM1
      UNEW = A(1,I,J)*UOM(I,J-1)+A(2,I,J)*UOM(I,J+1)+A(3,I,J)*UOM(I-1,J)
     1   +A(4,I,J)*UOM(I+1,J)
      RATIO = UNEW/UOM(I,J)
      LMAX = AMAX1(LMAX,RATIO)
      LMIN = AMIN1(LMIN,RATIO)
   40 UOM(I,J) = UNEW
      IF (LMAX.GT.1.) LMAX=1.
      ORFMAX = 2./(1.+SQRT(1.-LMAX))
      ORFMIN = 2./(1.+SQRT(1.-LMIN))
      IF ((ORFMAX-ORFMIN).GT.0.2*(2.-ORFMAX)) GO TO 30
      ORF = ORFMAX
      WRITE (NWRIT,1000) ORF
C
C--RESTORE U BOUNDARY VALUES
      DO 50 I=2,MMM1
   50 UOM(I,MHTP1) = 1.
      DO 60 J=2,MHT
      UOM(1,J) = UVERT(J,1)
   60 UOM(MM,J) = UVERT(J,2)
C
C--SOLVE MATRIX EQUATION BY SOR
   70 ERROR = 0.
      DO 80 J=2,MHT
      DO 80 I=2,MMM1
      CHANGE = ORF*(A(1,I,J)*UOM(I,J-1)+A(2,I,J)*UOM(I,J+1)+A(3,I,J)
```

```
     1    *UOM(I-1,J)+A(4,I,J)*UOM(I+1,J)+K(I,J)-UOM(I,J))
      ERROR = AMAX1(ERROR,ABS(CHANGE))
   80 UOM(I,J) = UOM(I,J)+CHANGE
      IF(ERROR.GT.1.E-5) GO TO 70
      RETURN
 1000 FORMAT (///5X,40HCALCULATED OVERRELAXATION FACTOR (ORF) =,F7.3)
      END




      SUBROUTINE NEWRHO
C
C--NEWRHO CALCULATES VELOCITY COMPONENTS, VELOCITY MAGNITUDE,
C--AND NEW DENSITY AT EACH MESH POINT
C
      COMMON SRW,SRE,ITER,IEND,NREAD,NWRIT
      COMMON/INPUTT/GAM,AR,MSFL,OMEGA,REDFAC,VELTOL,FNEW,DNEW,MBI,MBO,
     1    MM,MHT,NBL,NHUB,NTIP,NIN,NOUT,NBLPL,NPPP,NOSTAT,NSL,LSFR,
     2    LTPL,LAMVT,IMESH,ISLINE,ISTATL,IPLOT,ISUPER,ITSON,IDEBUG,
     3    ZOMIN,ZOMBI,ZOMBO,ZOMOUT,ZHIN,ZTIN,ZHOUT,ZTOUT,ZHUB(50),
     4    RHUB(50),ZTIP(50),RTIP(50),SFIN(50),RADIN(50),TIP(50),PRIP(50),
     5    LAMIN(50),VTHIN(50),SFOUT(50),RADOUT(50),PROP(50),LOSOUT(50),
     6    LAMOUT(50),VTHOUT(50),ZHST(50),ZTST(50),FLFR(50),
     7    ZBL(50,50),RBL(50,50),THBL(50,50),TNBL(50,50)
      COMMON/CALCON/MMM1,MHTP1,CP,EXPON,TGROG,PITCH,CURVHI,CURVTI,
     1    CURVHO,CURVTO,RHIN,RTIN,RHOUT,RTOUT,RLEH,RLET,RTEH,RTET,
     2    ZLE(50),RLE(50),ZTE(50),RTE(50),ZLEOM(101),RLEOM(101),
     3    SLEOM(101),THLEOM(101),ZTEOM(101),RTEOM(101),STEOM(101),
     4    THTEOM(101),ILE(101),ITE(101),ZOM(100,101),ROM(100,101),
     5    SOM(100,101),TOM(100,101),BTH(100,101),DTHDS(100,101),
     6    DTHDT(100,101),PLOSS(100,101),CPHI(100,101),SPHI(100,101)
      COMMON/VARCOM/A(4,100,101),UOM(100,101),K(100,101),RHO(100,101),
     1    WSUBS(100,101),WSUBT(100,101),WSUBZ(100,101),WSUBR(100,101),
     2    WSUBM(100,101),WTH(100,101),VTH(100,101),W(100,101),
     3    ALPHA(100,101),BETA(100,101),WWCR(100,101),CURV(100,101),
     4    WLSURF(100,101),WTSURF(100,101),CAMP(100,101),SAMP(100,101),
     5    RHOAV(100,101),DELRHO(100,101),FR(100,101),DFDM(100,101),
     6    XIOM(100,101),ZETOM(100,101),DLDU(100,101)
      DIMENSION DUDS(100),TVERT(101),UVERT(101),DUDT(101),TPPTIP(101),
     1    PREL(101),ROTI(101),DIDT(101),DPDT(101),AAA(101),
     2    DIDS(100,101),DPDS(100,101)
      REAL MSFL,LAMDAF
      INTEGER SRW,SRE
    1 RELER = 0.
      XNEW = 1.0
      ZNEW = 1.0
C
C--REINITIALIZE LAMDAF,RVTHTA,TIPF,RHOIPF,RHOOPF
C
      IF(LAMVT.EQ.0) GO TO 10
      CALL LAMNIT
      CALL RVTNIT
      CALL TIPNIT
      CALL RHINIT
      CALL RHONIT
C
C--CALCULATE PARTIAL DERIVATIVES OF ROTHALPY (ROTI), AND RELATIVE TOTAL
```

```
C--PRESSURE (PREL) IN THE S DIRECTION
C
   10 DO 30 J=1,MHTP1
      DO 20 I=1,MM
      TPPTIP(I) = 1.-(2.*OMEGA*LAMDAF(UOM(I,J),I,J)-(OMEGA*ROM(I,J))**2)
     1/2./CP/TIPF(UOM(I,J))
      IF (TPPTIP(I).LT.0.) GO TO 80
      PREL(I) = RHOIPF(UOM(I,J))*AR*TIPF(UOM(I,J))*TPPTIP(I)**(GAM*EXPON
     1)*(1.-PLOSS(I,J))
   20 ROTI(I) = CP*TIPF(UOM(I,J))-OMEGA*LAMDAF(UOM(I,J),I,J)
      CALL SLOPES(SOM(1,J),ROTI,MM,DIDS(1,J))
   30 CALL SLOPES(SOM(1,J),PREL,MM,DPDS(1,J))
C
C--CALCULATE WSUBT FROM THE PARTIAL OF UOM WITH RESPECT TO S USING THE
C--AVERAGE BLADE-TO-BLADE DENSITY FOR CONTINUITY
C
      DO 40 J=1,MHTP1
      CALL SPLINE(SOM(1,J),UOM(1,J),MM,DUDS,AAA)
      DO 40 I=1,MM
      WSUBT(I,J) = (-DUDS(I)/ROM(I,J)/BTH(I,J)*MSFL-DFDM(I,J)*
     1   DELRHO(I,J)/12.*COS(BETA(I,J))*SAMP(I,J))/RHOAV(I,J)
   40 CONTINUE
C
C--CALCULATE DERIVATIVES IN THE T DIRECTION OF THE SAME VARIABLES, AND
C--CALCULATE NEW VELOCITIES AND NEW DENSITY
C
      DO 60 I=1,MM
      DO 50 J=1,MHTP1
      TVERT(J) = TOM(I,J)
      UVERT(J) = UOM(I,J)
      TPPTIP(J) = 1.-(2.*OMEGA*LAMDAF(UOM(I,J),I,J)-(OMEGA*ROM(I,J))**2)
     1/2./CP/TIPF(UOM(I,J))
      IF (TPPTIP(J).LT.0.) GO TO 80
      PREL(J) = RHOIPF(UOM(I,J))*AR*TIPF(UOM(I,J))*TPPTIP(J)**(GAM*EXPON
     1)*(1.-PLOSS(I,J))
   50 ROTI(J) = CP*TIPF(UOM(I,J))-OMEGA*LAMDAF(UOM(I,J),I,J)
      CALL SPLINE(TVERT,UVERT,MHTP1,DUDT,AAA)
      CALL SLOPES(TVERT,ROTI,MHTP1,DIDT)
      CALL SLOPES(TVERT,PREL,MHTP1,DPDT)
      DO 60 J=1,MHTP1
      WSUBS(I,J) = (DUDT(J)/ROM(I,J)/BTH(I,J)*MSFL-DFDM(I,J)*
     1   DELRHO(I,J)/12.*COS(BETA(I,J))*CAMP(I,J))/RHOAV(I,J)
      WTH(I,J) = ROM(I,J)*(WSUBS(I,J)*DTHDS(I,J)+WSUBT(I,J)*DTHDT(I,J))
      IF (I.LT.ILE(J)) WTH(I,J)=LAMDAF(UOM(I,J),I,J)/ROM(I,J)-OMEGA*
     1ROM(I,J)
      IF (I.GT.ITE(J)) WTH(I,J)=RVTHTA(UOM(I,J),I,J)/ROM(I,J)-OMEGA*
     1ROM(I,J)
      VTH(I,J) = WTH(I,J)+OMEGA*ROM(I,J)
      WSQ = WTH(I,J)**2+WSUBS(I,J)**2+WSUBT(I,J)**2
      WTEMP = SQRT(WSQ)
      IF(W(I,J).NE.0.) RELER = AMAX1(RELER,ABS((WTEMP-W(I,J))/W(I,J)))
      W(I,J) = WTEMP
      TWLMR = 2.*OMEGA*LAMDAF(UOM(I,J),I,J)-(OMEGA*ROM(I,J))**2
      TTIP = 1.-(WSQ+TWLMR)/CP/TIPF(UOM(I,J))/2.
      IF(TTIP.LT.0.) GO TO 70
      RHO(I,J) = RHOIPF(UOM(I,J))*TTIP**EXPON
      TPP = TPPTIP(J)*TIPF(UOM(I,J))
      DPDR = DPDS(I,J)*SPHI(I,J)+DPDT(J)*CPHI(I,J)
      DIDR = DIDS(I,J)*SPHI(I,J)+DIDT(J)*CPHI(I,J)
```

```
      XIOMT = (AR/PREL(J)*DPDR-(DICR+OMEGA**2*ROM(I,J))/TPP)/2./CP
      ZETOMT = OMEGA**2*ROM(I,J)-AR/PREL(J)*TPP*DPDR
      XIOM(I,J) = XNEW*XIOMT+(1.-XNEW)*XIOM(I,J)
      ZETOM(I,J) = ZNEW*ZETOMT+(1.-ZNEW)*ZETOM(I,J)
   60 CONTINUE
      WRITE(NWRIT,1020) ITER,RELER
C
C--ADJUST PRINTING CONTROL VARIABLES
C
      IF (RELER.GE.VELTOL) RETURN
      IF (RELER.EQ.0.) RETURN
      IEND = IEND+1
      IF (IMESH.GT.1) IMESH=1
      IF (ISLINE.GT.1) ISLINE=1
      IF (ISTATL.GT.1) ISTATL=1
      IF (IPLOT.GT.1) IPLOT=1
      IF (ITSON.GT.1) ITSON=1
      IF (IDEBUG.GT.1) IDEBUG=1
      RETURN
   70 WRITE(NWRIT,1000)
      STOP
   80 WRITE(NWRIT,1010)
      STOP
 1000 FORMAT(68H1PROGRAM STOPPED IN NEWRHO DUE TO EXCESSIVE STREAM FUNCT
     1ION GRADIENT)
 1010 FORMAT(62HL THE UPSTREAM INPUT WHIRL OR TANGENTIAL VELOCITY IS TOO
     1 LARGE)
 1020 FORMAT(///5X,9HITERATION,I3,41H,    MAXIMUM RELATIVE CHANGE IN VELO
     1CITY =,G11.4)
      END




      SUBROUTINE OUTPUT
C
C--OUTPUT CALCULATES AND PRINTS THE MAJOR OUTPUT DATA
C--AT THE ORTHOGONAL MESH POINTS, ALONG THE STREAMLINES,
C--AND ALONG STATION LINES FROM HUB TO SHROUD
C
      COMMON SRW,SRE,ITER,IEND,NREAD,NWRIT
      COMMON/INPUTT/GAM,AR,MSFL,OMEGA,REDFAC,VELTOL,FNEW,DNEW,MBI,MBO,
     1   MM,MHT,NBL,NHUB,NTIP,NIN,NOUT,NBLPL,NPPP,NOSTAT,NSL,LSFR,
     2   LTPL,LAMVT,IMESH,ISLINE,ISTATL,IPLOT,ISUPER,ITSON,IDEBUG,
     3   ZOMIN,ZOMBI,ZOMBO,ZOMOUT,ZHIN,ZTIN,ZHOUT,ZTOUT,ZHUB(50),
     4   RHUB(50),ZTIP(50),RTIP(50),SFIN(50),RADIN(50),TIP(50),PRIP(50),
     5   LAMIN(50),VTHIN(50),SFOUT(50),RADOUT(50),PROP(50),LOSOUT(50),
     6   LAMOUT(50),VTHOUT(50),ZHST(50),ZTST(50),FLFR(50),
     7   ZBL(50,50),RBL(50,50),THBL(50,50),TNBL(50,50)
      COMMON/CALCON/MMM1,MHTP1,CP,EXPON,TGROG,PITCH,CURVHI,CURVTI,
     1   CURVHO,CURVTO,RHIN,RTIN,RHOUT,RTOUT,RLEH,RLET,RTEH,RTET,
     2   ZLE(50),RLE(50),ZTE(50),RTE(50),ZLEOM(101),RLEOM(101),
     3   SLEOM(101),THLEOM(101),ZTEOM(101),RTEOM(101),STEOM(101),
     4   THTEOM(101),ILE(101),ITE(101),ZOM(100,101),ROM(100,101),
     5   SOM(100,101),TOM(100,101),BTH(100,101),DTHDS(100,101),
     6   DTHDT(100,101),PLOSS(100,101),CPHI(100,101),SPHI(100,101)
      COMMON/VARCOM/A(4,100,101),UOM(100,101),K(100,101),RHO(100,101),
     1   WSUBS(100,101),WSUBT(100,101),WSUBZ(100,101),WSUBR(100,101),
```

```
     2    WSUBM(100,101),WTH(100,101),VTH(100,101),W(100,101),
     3    ALPHA(100,101),BETA(100,101),WWCR(100,101),CURV(100,101),
     4    WLSURF(100,101),WTSURF(100,101),CAMP(100,101),SAMP(100,101),
     5    RHOAV(100,101),DELRHO(100,101),FR(100,101),DFDM(100,101),
     6    XIOM(100,101),ZETOM(100,101),DLDU(100,101)
      COMMON/SLCOM/ ILS(50),ITS(50),ZSL(100,50),RSL(100,50),MSL(100,50),
     1    WZSL(100,50),WRSL(100,50),WMSL(100,50),WTHSL(100,50),
     2    ALPSL(100,50),BETSL(100,50),WSL(100,50),WWCRSL(100,50),
     3    CURVSL(100,50),WLSSL(100,50),WTSSL(100,50)
      COMMON/STACOM/ZST(50,50),RST(50,50),MST(50,50),WZST(50,50),
     1    WRST(50,50),WMST(50,50),WTHST(50,50),ALPST(50,50),BETST(50,50),
     2    WST(50,50),WWCRST(50,50),CURVST(50,50),WLSST(50,50),
     3    WTSST(50,50)
      DIMENSION DALDS(100),TVERT(101),ALVERT(101),DALVER(101),
     1    ZTEM(101),RTEM(101),UTEM(101),ZSLTEM(50),RSLTEM(50),
     2    ZBLTEM(50),RBLTEM(50),MTEM(50),MARK(50),AAA(101),
     3    DALDT(100,101)
      REAL LAMDAF,LAMIN,LAMOUT,MSL,MST,MTEM
C
C--CALCULATE VELOCITY COMPONENTS AND FLOW ANGLES
C
      DEGRAD = 180./3.1415927
      DO 10 J=1,MHTP1
      DO 10 I=1,MM
      WSUBM(I,J) = SQRT(WSUBS(I,J)**2+WSUBT(I,J)**2)
      SAMP(I,J) = WSUBT(I,J)/WSUBM(I,J)
      CAMP(I,J) = WSUBS(I,J)/WSUBM(I,J)
      WSUBZ(I,J) = WSUBS(I,J)*CPHI(I,J)-WSUBT(I,J)*SPHI(I,J)
      WSUBR(I,J) = WSUBT(I,J)*CPHI(I,J)+WSUBS(I,J)*SPHI(I,J)
      ALPHA(I,J) = ATAN(WSUBR(I,J)/WSUBZ(I,J))
   10 BETA(I,J) = ATAN(WTH(I,J)/WSUBM(I,J))
      GO TO 30
C
      ENTRY TOUTPT
C
C--CALCULATE VELOCITY COMPONENTS AFTER TRANSONIC SOLUTION
C
      DO 20 J=1,MHTP1
      DO 20 I=1,MM
      WSUBM(I,J) = W(I,J)*COS(BETA(I,J))
      WTH(I,J) = W(I,J)*SIN(BETA(I,J))
      WSUBZ(I,J) = WSUBM(I,J)*COS(ALPHA(I,J))
      WSUBR(I,J) = WSUBM(I,J)*SIN(ALPHA(I,J))
   20 VTH(I,J) = WTH(I,J)+OMEGA*ROM(I,J)
C
C--CALCULATE STREAMLINE CURVATURE AND CRITICAL VELOCITY RATIO
C
   30 DO 50 I=1,MM
      DO 40 J=1,MHTP1
      TVERT(J) = TOM(I,J)
   40 ALVERT(J) = ALPHA(I,J)
      CALL SLOPES(TVERT,ALVERT,MHTP1,DALVER)
      DO 50 J=1,MHTP1
   50 DALDT(I,J) = DALVER(J)
      DO 60 J=1,MHTP1
      CALL SLOPES(SOM(1,J),ALPHA(1,J),MM,DALDS)
      DO 60 I=1,MM
      CURV(I,J) = DALDS(I)*CAMP(I,J)+DALDT(I,J)*SAMP(I,J)
      TPP = TIPF(UOM(I,J))-(2.*OMEGA*LAMDAF(UOM(I,J),I,J)-(OMEGA*
     1ROM(I,J))**2)/2./CP
```

```
      IF (TPP.LE.0.) TPP=1.
   60 WWCR(I,J) = W(I,J)/SQRT(TGROG*TPP)
C
C--COMPUTE BLADE SURFACE VELOCITIES BY STANITZ METHOD
C
      CALL BLDVEL
C
C--CHECK IF UPPER OR LOWER SURFACE IS SUCTION SURFACE
C
      MCT = 2
      IF ((LAMDAF(.5,ILE(1),1)-RVTHTA(.5,ILE(1),1))*OMEGA.LT.0.) MCT=1
      IF ((LAMDAF(.5,ILE(1),1)-RVTHTA(.5,ILE(1),1)).GT.0.) GO TO 80
      DO 70 J=1,MHTP1
      DO 70 I=1,MM
      WDUM = WLSURF(I,J)
      WLSURF(I,J) = WTSURF(I,J)
   70 WTSURF(I,J) = WDUM
C
C--PRINT OUTPUT ROW BY ROW FROM HUB TO TIP
C
   80 IF (IMESH.LE.0) GO TO 100
      IF ((ITER/IMESH)*IMESH.NE.ITER.AND.ITER.NE.1) GO TO 100
      WRITE(NWRIT,1000)
      IF (REDFAC.LT.1.0) WRITE(NWRIT,1150) ITER
      IF (REDFAC.EQ.1.0.AND.IEND.LE.0) WRITE(NWRIT,1160) ITER
      IF (REDFAC.EQ.1.0.AND.IEND.EQ.1) WRITE(NWRIT,1170)
      IF (REDFAC.EQ.1.0.AND.IEND.EQ.2) WRITE(NWRIT,1180)
      DO 90 J=1,MHTP1
      WRITE(NWRIT,1010) J
      WRITE(NWRIT,1020)
      DO 90 I=1,MM
      PHI = ARSIN(SPHI(I,J))*DEGRAD
      ALPHA(I,J) = ALPHA(I,J)*DEGRAD
      BETA(I,J) = BETA(I,J)*DEGRAD
      WRITE(NWRIT,1030) I,J,ZOM(I,J),ROM(I,J),UOM(I,J),WSUBM(I,J),
     1WTH(I,J),W(I,J),WWCR(I,J),ALPHA(I,J),BETA(I,J),PHI
      ALPHA(I,J) = ALPHA(I,J)/DEGRAD
   90 BETA(I,J) = BETA(I,J)/DEGRAD
C
C--INTERPOLATE TO OBTAIN OUTPUT DATA ON STREAMLINES
C
  100 IF (ISLINE.LE.0) GO TO 110
      IF ((ITER/ISLINE)*ISLINE.EQ.ITER.OR.ITER.EQ.1) GO TO 130
  110 IF (IPLOT.LE.0) GO TO 120
      IF ((ITER/IPLOT)*IPLOT.EQ.ITER.OR.ITER.EQ.1) GO TO 130
  120 IF (ITSON.LE.0) GO TO 220
      IF ((ITER/ITSON)*ITSON.NE.ITER) GO TO 220
C--CALCULATE STREAMLINE ZSL,RSL COORDINATES FOR PRINT OUT
  130 DO 150 I=1,MM
      DO 140 J=1,MHTP1
      ZTEM(J) = ZOM(I,J)
      RTEM(J) = ROM(I,J)
  140 UTEM(J) = UOM(I,J)
      CALL SPLINT(UTEM,RTEM,MHTP1,FLFR,NSL,RSLTEM,AAA)
      CALL SPLINT(RTEM,ZTEM,MHTP1,RSLTEM,NSL,ZSLTEM,AAA)
      DO 150 JS=1,NSL
      ZSL(I,JS) = ZSLTEM(JS)
  150 RSL(I,JS) = RSLTEM(JS)
C--CALCULATE STREAMLINE MSL COORDINATES FOR PRINT OUT AND PLOTTING
```

```
      DO 170 JS=1,NSL
      MSL(1,JS) = 0.
      DO 160 IS=2,MM
  160 MSL(IS,JS) = MSL(IS-1,JS)+SQRT((ZSL(IS,JS)-ZSL(IS-1,JS))**2
     1   +(RSL(IS,JS)-RSL(IS-1,JS))**2)
      CALL SPLINT(ZSL(1,JS),MSL(1,JS),MM,0.,1,ZEROM,SLREF)
      DO 170 IS=1,MM
  170 MSL(IS,JS) = MSL(IS,JS)-ZEROM
C--INTERPOLATE TO OBTAIN OUTPUT DATA
      II = 1
      JJ = 1
      DO 180 JS=1,NSL
      DO 180 IS=1,MM
      CALL LININT(ZOM,ROM,WSUBZ,MM,MHTP1,100,101,ZSL(IS,JS),RSL(IS,JS),
     1WZSL(IS,JS),II,JJ)
      CALL LININT(ZOM,ROM,WSUBR,MM,MHTP1,100,101,ZSL(IS,JS),RSL(IS,JS),
     1WRSL(IS,JS),II,JJ)
      CALL LININT(ZOM,ROM,WTH,MM,MHTP1,100,101,ZSL(IS,JS),RSL(IS,JS),
     1WTHSL(IS,JS),II,JJ)
      CALL LININT(ZOM,ROM,WWCR,MM,MHTP1,100,101,ZSL(IS,JS),RSL(IS,JS),
     1WWCRSL(IS,JS),II,JJ)
      CALL LININT(ZOM,ROM,CURV,MM,MHTP1,100,101,ZSL(IS,JS),RSL(IS,JS),
     1CURVSL(IS,JS),II,JJ)
      WMSL(IS,JS) = SQRT(WZSL(IS,JS)**2+WRSL(IS,JS)**2)
      ALPSL(IS,JS) = ATAN(WRSL(IS,JS)/WZSL(IS,JS))*DEGRAD
      BETSL(IS,JS) = ATAN(WTHSL(IS,JS)/WMSL(IS,JS))*DEGRAD
  180 WSL(IS,JS) = SQRT(WMSL(IS,JS)**2+WTHSL(IS,JS)**2)
C
C--CALCULATE ILS AND ITS ARRAYS OF STREAMLINE LOCATIONS INSIDE BLADE
C--LEADING AND TRAILING EDGES
C
      CALL ILETE
C
C--CALCULATE BLADE SURFACE VELOCITIES ON STREAMLINES BY INTERPOLATION
C
      DO 190 JS=1,NSL
      DO 190 IS=1,MM
      WLSSL(IS,JS) = 0.
  190 WTSSL(IS,JS) = 0.
      II = 1
      JJ = 1
      DO 200 JS=1,NSL
      ILSJ = ILS(JS)
      ITSJ = ITS(JS)
      DO 200 IS=ILSJ,ITSJ
      CALL LININT(ZOM,ROM,WLSURF,MM,MHTP1,100,101,ZSL(IS,JS),RSL(IS,JS),
     1WLSSL(IS,JS),II,JJ)
  200 CALL LININT(ZOM,ROM,WTSURF,MM,MHTP1,100,101,ZSL(IS,JS),RSL(IS,JS),
     1WTSSL(IS,JS),II,JJ)
C
C--PRINT OUTPUT ON STREAMLINES
C
      IF (ISLINE.LE.0) GO TO 220
      IF ((ITER/ISLINE)*ISLINE.NE.ITER.AND.ITER.NE.1) GO TO 220
      WRITE(NWRIT,1040)
      IF (REDFAC.LT.1.0) WRITE(NWRIT,1150) ITER
      IF (REDFAC.EQ.1.0.AND.IEND.LE.0) WRITE(NWRIT,1160) ITER
      IF (REDFAC.EQ.1.0.AND.IEND.EQ.1) WRITE(NWRIT,1170)
      IF (REDFAC.EQ.1.0.AND.IEND.EQ.2) WRITE(NWRIT,1180)
      DO 210 JS=1,NSL
```

```
      WRITE(NWRIT,1050) JS,FLFR(JS)
      WRITE(NWRIT,1060)
      DO 210 IS=1,MM
      WRITE(NWRIT,1070) ZSL(IS,JS),RSL(IS,JS),MSL(IS,JS),WMSL(IS,JS),
     1WTHSL(IS,JS),WSL(IS,JS),WWCRSL(IS,JS),ALPSL(IS,JS),BETSL(IS,JS),
     2CURVSL(IS,JS),WLSSL(IS,JS),WTSSL(IS,JS)
      ALPSL(IS,JS) = ALPSL(IS,JS)/DEGRAD
  210 BETSL(IS,JS) = BETSL(IS,JS)/DEGRAD
C
C--INTERPOLATE TO OBTAIN OUTPUT DATA ON HUB-SHROUD STATION LINES
C
  220 IF (ISTATL.LE.0.OR.NOSTAT.EQ.0) GO TO 410
      IF ((ITER/ISTATL)*ISTATL.NE.ITER.AND.ITER.NE.1) GO TO 410
C--CALCULATE ZST AND RST ARRAYS
C--STORE HUB AND SHROUD POINTS INTO ZST AND RST ARRAYS
      CALL SPLINT(ZHUB,RHUB,NHUB,ZHST,NOSTAT,RTEM,AAA)
      DO 230 IL=1,NOSTAT
      ZST(1,IL) = ZHST(IL)
  230 RST(1,IL) = RTEM(IL)
      CALL SPLINT(ZTIP,RTIP,NTIP,ZTST,NOSTAT,RTEM,AAA)
      DO 240 IL=1,NOSTAT
      ZST(NSL,IL) = ZTST(IL)
  240 RST(NSL,IL) = RTEM(IL)
C--CALCULATE INTERIOR POINTS IN ZST AND RST ARRAYS
      DO 350 IL=1,NOSTAT
      MARK(IL) = 1
      RTEM(1) = RST(1,IL)
      RTEM(20) = RST(NSL,IL)
      DELR = (RTEM(20)-RTEM(1))/19.0
      ZTEM(1) = ZST(1,IL)
      ZTEM(20) = ZST(NSL,IL)
      DELZ = (ZTEM(20)-ZTEM(1))/19.0
      DO 250 J=2,19
  250 RTEM(J) = RTEM(J-1)+DELR
C--CHECK FOR LEADING OR TRAILING EDGE STATION
      DELCH = (ZTE(1)-ZLE(1)+ZTE(NBLPL)-ZLE(NBLPL))*0.005
      IF ((ZST(1,IL).GT.(ZLE(1)-DELCH).AND.ZST(1,IL).LT.(ZLE(1)+DELCH))
     1.AND.(ZST(NSL,IL).GT.(ZLE(NBLPL)-DELCH).AND.ZST(NSL,IL).LT.
     2(ZLE(NBLPL)+DELCH)))  MARK(IL)=2
      IF ((ZST(1,IL).GT.(ZTE(1)-DELCH).AND.ZST(1,IL).LT.(ZTE(1)+DELCH))
     1.AND.(ZST(NSL,IL).GT.(ZTE(NBLPL)-DELCH).AND.ZST(NSL,IL).LT.
     2(ZTE(NBLPL)+DELCH)))  MARK(IL)=3
      IF (ZST(1,IL).GT.(ZLE(1)+DELCH).AND.ZST(1,IL).LT.(ZTE(1)-DELCH))
     1MARK(IL)=4
      IF (MARK(IL).EQ.2) GO TO 270
      IF (MARK(IL).EQ.3) GO TO 290
C--REGULAR STATION
      DO 260 J=2,19
  260 ZTEM(J) = ZTEM(J-1)+DELZ
      GO TO 310
C--LEADING EDGE STATION
  270 DO 280 JN=1,NBLPL
      ZBLTEM(JN) = ZLE(JN)
  280 RBLTEM(JN) = RLE(JN)
      CALL SPLINT(RBLTEM,ZBLTEM,NBLPL,RTEM,20,ZTEM,AAA)
      GO TO 310
C--TRAILING EDGE STATION
  290 DO 300 JN=1,NBLPL
      ZBLTEM(JN) = ZTE(JN)
  300 RBLTEM(JN) = RTE(JN)
```

```
      CALL SPLINT(RBLTEM,ZBLTEM,NBLPL,RTEM,20,ZTEM,AAA)
C--INTERPOLATE FOR STREAM FUNCTION
  310 UTEM(1) = 0.
      UTEM(20) = 1.
      II = 1
      JJ = 1
      DO 320 J=2,19
  320 CALL LININT(ZOM,ROM,UOM,MM,MHTP1,100,101,ZTEM(J),RTEM(J),UTEM(J),
     1II,JJ)
C--CALCULATE STATION LINE RST COORDINATES FOR PRINT OUT
      CALL SPLINT(UTEM,RTEM,20,FLFR,NSL,RST(1,IL),AAA)
      DELR = RST(NSL,IL)-RST(1,IL)
      DELZ = ZST(NSL,IL)-ZST(1,IL)
      NSLM1 = NSL-1
C--CALCULATE STATION LINE ZST COORDINATES FOR PRINT OUT
      IF (MARK(IL).EQ.2.OR.MARK(IL).EQ.3) GO TO 340
      DO 330 JL=2,NSLM1
  330 ZST(JL,IL) = ZST(1,IL)+(RST(JL,IL)-RST(1,IL))/DELR*DELZ
      GO TO 350
  340 CALL SPLINT(RBLTEM,ZBLTEM,NBLPL,RST(1,IL),NSL,ZST(1,IL),AAA)
  350 CONTINUE
C--CALCULATE STATION LINE MST COORDINATES FOR PRINT OUT
      DO 380 JL=1,NSL
      DO 360 IL=1,NOSTAT
      ZTEM(IL) = ZST(JL,IL)
  360 RTEM(IL) = RST(JL,IL)
      MTEM(1) = 0.
      DO 370 IL=2,NOSTAT
  370 MTEM(IL) = MTEM(IL-1)+SQRT((ZTEM(IL)-ZTEM(IL-1))**2+(RTEM(IL)-
     1RTEM(IL-1))**2)
      CALL SPLINT(ZTEM,MTEM,NOSTAT,0.,1,ZEROM,SLREF)
      DO 380 IL=1,NOSTAT
  380 MST(JL,IL) = MTEM(IL)-ZEROM
C
C--INTERPOLATE TO OBTAIN OUTPUT DATA ON STATION LINES
      II = 1
      JJ = 1
      DO 390 IL=1,NOSTAT
      DO 390 JL=1,NSL
      CALL LININT(ZOM,ROM,WSUBZ,MM,MHTP1,100,101,ZST(JL,IL),RST(JL,IL),
     1WZST(JL,IL),II,JJ)
      CALL LININT(ZOM,ROM,WSUBR,MM,MHTP1,100,101,ZST(JL,IL),RST(JL,IL),
     1WRST(JL,IL),II,JJ)
      CALL LININT(ZOM,ROM,WTH,MM,MHTP1,100,101,ZST(JL,IL),RST(JL,IL),
     1WTHST(JL,IL),II,JJ)
      CALL LININT(ZOM,ROM,WWCR,MM,MHTP1,100,101,ZST(JL,IL),RST(JL,IL),
     1WWCRST(JL,IL),II,JJ)
      CALL LININT(ZOM,ROM,CURV,MM,MHTP1,100,101,ZST(JL,IL),RST(JL,IL),
     1CURVST(JL,IL),II,JJ)
      WMST(JL,IL) = SQRT(WZST(JL,IL)**2+WRST(JL,IL)**2)
      ALPST(JL,IL) = ATAN(WRST(JL,IL)/WZST(JL,IL))*DEGRAD
      BETST(JL,IL) = ATAN(WTHST(JL,IL)/WMST(JL,IL))*DEGRAD
      WST(JL,IL) = SQRT(WMST(JL,IL)**2+WTHST(JL,IL)**2)
      WLSST(JL,IL) = 0.
      WTSST(JL,IL) = 0.
      IF (MARK(IL).EQ.1) GO TO 390
      CALL LININT(ZOM,ROM,WLSURF,MM,MHTP1,100,101,ZST(JL,IL),RST(JL,IL),
     1WLSST(JL,IL),II,JJ)
      CALL LININT(ZOM,ROM,WTSURF,MM,MHTP1,100,101,ZST(JL,IL),RST(JL,IL),
     1WTSST(JL,IL),II,JJ)
```

```
  390 CONTINUE
C
C--PRINT OUTPUT ALONG HUB-SHROUD STATION LINES
C
      WRITE(NWRIT,1080)
      IF (REDFAC.LT.1.0) WRITE(NWRIT,1150) ITER
      IF (REDFAC.EQ.1.0.AND.IEND.LE.0) WRITE(NWRIT,1160) ITER
      IF (REDFAC.EQ.1.0.AND.IEND.EQ.1) WRITE(NWRIT,1170)
      IF (REDFAC.EQ.1.0.AND.IEND.EQ.2) WRITE(NWRIT,1180)
      DO 400 IL=1,NOSTAT
      IF (MARK(IL).EQ.1) WRITE(NWRIT,1090) IL
      IF (MARK(IL).EQ.2) WRITE(NWRIT,1100) IL
      IF (MARK(IL).EQ.3) WRITE(NWRIT,1110) IL
      IF (MARK(IL).EQ.4) WRITE(NWRIT,1120) IL
      WRITE(NWRIT,1130)
      DO 400 JL=1,NSL
      WRITE(NWRIT,1140) RST(JL,IL),ZST(JL,IL),MST(JL,IL),FLFR(JL),
     1WMST(JL,IL),WTHST(JL,IL),WST(JL,IL),WWCRST(JL,IL),ALPST(JL,IL),
     2BETST(JL,IL),CURVST(JL,IL),WLSST(JL,IL),WTSST(JL,IL)
      ALPST(JL,IL) = ALPST(JL,IL)/DEGRAD
  400 BETST(JL,IL) = BETST(JL,IL)/DEGRAD
C
C--CALCULATE DATA FOR INPUT TO THE TSONIC PROGRAM
C
  410 IF (ITSON.LE.0) RETURN
      IF ((ITER/ITSON)*ITSON.NE.ITER) RETURN
      CALL TSONIN
      RETURN
C
C--FORMAT STATEMENTS
C
 1000 FORMAT (1H1///28X,79H***  STREAM FUNCTION, INTERIOR VELOCITIES, V
     1ELOCITY COMPONENTS, AND ANGLES  ***/44X,41HAT ALL MESH POINTS OF T
     2HE ORTHOGONAL MESH/44X,41(1H*))
 1010 FORMAT (///42X,39H** HORIZONTAL ORTHOGONAL MESH LINE NO. ,
     1I2,3H **//)
 1020 FORMAT (1X,10HMESH-POINT,3X,5HAXIAL,8X,6HRADIAL,6X,6HSTREAM,4X,
     16HMERID.,3X,9HREL.TANG.,4X,4HREL.,3X,9HCRIT.VEL.,3X,6HMERID.,3X,
     28HREL.FLOW,3X,4HMESH/1X,9HCOLM  ROW,4X,6HCOORD.,7X,6HCOORD.,7X,
     35HFUNC.,5X,4HVEL.,6X,4HVEL.,7X,4HVEL.,5X,5HRATIO,3(5X,5HANGLE)/
     42X,8H(I)  (J),5X,3H(Z),10X,3H(R),10X,3H(U),6X,4H(WM),5X,5H(WTH),
     57X,3H(W),5X,7H(W/WCR),3X,7H(ALPHA),3X,6H(BETA),5X,5H(PHI))
 1030 FORMAT (1X,I3,2X,I3,2X,2(G12.5,1X),F8.4,3(1X,F9.2),1X,F9.3,
     13(3X,F7.2))
 1040 FORMAT (1H1///15X,99H***  STREAM FUNCTION, INTERIOR VELOCITIES, V
     1ELOCITY COMPONENTS, ANGLES, AND SURFACE VELOCITIES  ***/56X,17HALO
     2NG STREAMLINES/56X,17(1H*))
 1050 FORMAT(///36X,20H** STREAMLINE NUMBER,I3,23H  --  STREAM FUNCTION
     1=,F8.4,3H **//)
 1060 FORMAT (4X,5HAXIAL,8X,6HRADIAL,7X,6HMERID.,6X,6HMERID.,2X,
     19HREL.TANG.,2X,4HREL.,2X,9HCRIT.VEL.,2X,6HMERID.,2X,8HREL.FLOW,
     22X,7HSTREAM.,3X,9HSUCT.SUR.,1X,9HPRES.SUR./4X,6HCOORD.,7X,
     36HCOORD.,7X,6HCOORD.,7X,4HVEL.,5X,4HVEL.,5X,4HVEL.,4X,5HRATIO,
     42(4X,5HANGLE),5X,5HCURV.,6X,4HVEL.,6X,4HVEL./5X,3H(Z),10X,3H(R),
     510X,3H(M),9X,4H(WM),4X,5H(WTH),5X,3H(W),4X,7H(W/WCR),2X,
     67H(ALPHA),2X,6H(BETA),3X,9H(1./DIST),4X,4H(WS),6X,4H(WP))
 1070 FORMAT (3(1X,G12.5),3(1X,F8.2),1X,F7.3,2(2X,F7.2),2X,G11.4,
     1F8.2,2X,F8.2)
 1080 FORMAT (1H1///15X,99H***  STREAM FUNCTION, INTERIOR VELOCITIES, V
```

```
     1ELOCITY COMPONENTS, ANGLES, AND SURFACE VELOCITIES  ***/28X,72HALO
     2NG LINES FROM HUB TO SHROUD AT VARIOUS STATIONS THROUGH THE BLADE
     3ROW/28X,72(1H*))
 1090 FORMAT(///49X,26H** HUB-SHROUD STATION NO. ,I2,3H **//)
 1100 FORMAT(///49X,26H** HUB-SHROUD STATION NO. ,I2,3H **,16X,
     118H** LEADING EDGE **//)
 1110 FORMAT(///49X,26H** HUB-SHROUD STATION NO. ,I2,3H **,15X,
     119H** TRAILING EDGE **//)
 1120 FORMAT (///49X,26H** HUB-SHROUD STATION NO. ,I2,3H **,16X,
     118H** WITHIN BLADE **//)
 1130 FORMAT (4X,6HRADIAL,7X,5HAXIAL,8X,6HMERID.,4X,6HSTREAM,3X,
     16HMERID.,2X,9HREL.TANG.,2X,4HREL.,2X,9HCRIT.VEL.,2X,6HMERID.,2X,
     28HREL.FLOW,2X,7HSTREAM.,3X,9HSUCT.SUR.,1X,9HPRES.SUR./4X,
     36HCOORD.,7X,6HCOORD.,7X,6HCOORD.,5X,5HFUNC.,4X,4HVEL.,5X,4HVEL.,
     45X,4HVEL.,4X,5HRATIO,2(4X,5HANGLE),5X,5HCURV.,6X,4HVEL.,6X,4HVEL./
     55X,3H(R),10X,3H(Z),10X,3H(M),8X,3H(U),5X,4H(WM),4X,5H(WTH),5X,
     63H(W),4X,7H(W/WCR),2X,7H(ALPHA),2X,6H(BETA),3X,9H(1./DIST),4X,
     74H(WS),6X,4H(WP))
 1140 FORMAT (1X,3(G12.5,1X),F6.4,3(1X,F8.2),1X,F7.3,2(2X,F7.2),2X,
     1G11.4,F8.2,2X,F8.2)
 1150 FORMAT (/53X,23(1H*)/53X,23H*  REDUCED  MASSFLOW  */53X,23(1H*)/
     153X,18H*  ITERATION  NO. ,I2,3H  */53X,23(1H*))
 1160 FORMAT (/52X,25(1H*)/52X,25H*   FULL   MASSFLOW   */52X,25(1H*)/
     152X,19H*   ITERATION  NO. ,I2,4H   */52X,25(1H*))
 1170 FORMAT (/52X,25(1H*)/52X,25H*   FULL   MASSFLOW   */42X,45(1H*)/
     142X,1H*,12X,19HTRANSONIC  SOLUTION,12X,1H*/42X,45H*  BY VELOCITY G
     2RADIENT APPROXIMATE METHOD  */35X,59(1H*)/35X,59H*  ALL VELOCITIES
     3 SMALLER THAN CHOKING MASSFLOW SOLUTION  */35X,59(1H*))
 1180 FORMAT (/52X,25(1H*)/52X,25H*   FULL   MASSFLOW   */42X,45(1H*)/
     142X,1H*,12X,19HTRANSONIC  SOLUTION,12X,1H*/42X,45H*  BY VELOCITY G
     2RADIENT APPROXIMATE METHOD  */35X,59(1H*)/35X,59H*  ALL VELOCITIES
     3 LARGER THAN CHOKING MASSFLOW SOLUTION   */35X,59(1H*))
      END


      SUBROUTINE BLDVEL
C
C--BLDVEL CALCULATES BLADE SURFACE VELOCITIES AND FR
C
      COMMON SRW,SRE,ITER,IEND,NREAD,NWRIT
      COMMON/INPUTT/GAM,AR,MSFL,OMEGA,REDFAC,VELTOL,FNEW,DNEW,MBI,MBO,
     1   MM,MHT,NBL,NHUB,NTIP,NIN,NOUT,NBLPL,NPPP,NOSTAT,NSL,LSFR,
     2   LTPL,LAMVT,IMESH,ISLINE,ISTATL,IPLOT,ISUPER,ITSON,IDEBUG,
     3   ZOMIN,ZOMBI,ZOMBO,ZOMOUT,ZHIN,ZTIN,ZHOUT,ZTOUT,ZHUB(50),
     4   RHUB(50),ZTIP(50),RTIP(50),SFIN(50),RADIN(50),TIP(50),PRIP(50),
     5   LAMIN(50),VTHIN(50),SFOUT(50),RADOUT(50),PROP(50),LOSOUT(50),
     6   LAMOUT(50),VTHOUT(50),ZHST(50),ZTST(50),FLFR(50),
     7   ZBL(50,50),RBL(50,50),THBL(50,50),TNBL(50,50)
      COMMON/CALCON/MMM1,MHTP1,CP,EXPON,TGROG,PITCH,CURVHI,CURVTI,
     1   CURVHO,CURVTO,RHIN,RTIN,RHOUT,RTOUT,RLEH,RLET,RTEH,RTET,
     2   ZLE(50),RLE(50),ZTE(50),RTE(50),ZLEOM(101),RLEOM(101),
     3   SLEOM(101),THLEOM(101),ZTEOM(101),RTEOM(101),STEOM(101),
     4   THTEOM(101),ILE(101),ITE(101),ZOM(100,101),ROM(100,101),
     5   SOM(100,101),TOM(100,101),BTH(100,101),DTHDS(100,101),
     6   DTHDT(100,101),PLOSS(100,101),CPHI(100,101),SPHI(100,101)
      COMMON/VARCOM/A(4,100,101),UOM(100,101),K(100,101),RHO(100,101),
```

```
     1    WSUBS(100,101),WSUBT(100,101),WSUBZ(100,101),WSUBR(100,101),
     2    WSUBM(100,101),WTH(100,101),VTH(100,101),W(100,101),
     3    ALPHA(100,101),BETA(100,101),WWCR(100,101),CURV(100,101),
     4    WLSURF(100,101),WTSURF(100,101),CAMP(100,101),SAMP(100,101),
     5    RHOAV(100,101),DELRHO(100,101),FR(100,101),DFDM(100,101),
     6    XIOM(100,101),ZETOM(100,101),DLDU(100,101)
      DIMENSION TVERT(101),FVERT(101),DFVERT(101),DFDS(100),
     1    FST(100,101),DFDT(100,101)
      REAL MSFL,LAMDAF
      INTEGER SRW,SRE
   10 FCHANG = 0.
      FMAX = -1.E20
      FMIN = 1.E20
C---CALCULATE DFDT
      DO 30 I=1,MM
      DO 20 J=1,MHTP1
      TVERT(J) = TOM(I,J)
      FST(I,J) = VTH(I,J)*ROM(I,J)
      FVERT(J) = FST(I,J)
   20 CONTINUE
      CALL SLOPES(TVERT,FVERT,MHTP1,DFVERT)
      DO 30 J=1,MHTP1
      DFDT(I,J) = DFVERT(J)
   30 CONTINUE
C---CALCULATE DFDS, THEN DFDM AND BLADE SURFACE VELOCITIES
      DO 40 J=1,MHTP1
      CALL SLOPES(SOM(1,J),FST(1,J),MM,DFDS)
      DO 40 I=1,MM
      DFDM(I,J) = -(DFDS(I)*CAMP(I,J)+DFDT(I,J)*SAMP(I,J))*BTH(I,J)*
     1COS(BETA(I,J))
      WLSURF(I,J) = W(I,J)+DFDM(I,J)/2.
      WTSURF(I,J) = W(I,J)-DFDM(I,J)/2.
C---CALCULATE BLADE-TO-BLADE AVERAGE DENSITY
      TWLMR = 2.*OMEGA*LAMDAF(UOM(I,J),I,J)-(OMEGA*ROM(I,J))**2
      WSQ = WLSURF(I,J)**2
      TTIP = 1.-(WSQ+TWLMR)/CP/TIPF(UOM(I,J))/2.
      IF(TTIP.LT.0.) TTIP = 0.
      RHOL = RHOIPF(UOM(I,J))*TTIP**EXPON
      WSQ = WTSURF(I,J)**2
      TTIP = 1.-(WSQ+TWLMR)/CP/TIPF(UOM(I,J))/2.
      IF(TTIP.LT.0.) TTIP = 0.
      RHOT = RHOIPF(UOM(I,J))*TTIP**EXPON
      DELRHO(I,J) = RHOL-RHOT
      RHOAV(I,J) = (RHOL+4.*RHO(I,J)+RHOT)/6.
C---CALCULATE F-SUB-R FOR SUBROUTINE COEF
      FRT = W(I,J)/BTH(I,J)*(DTHDS(I,J)*SPHI(I,J)+DTHDT(I,J)*CPHI(I,J))*
     1DFDM(I,J)
      FCHANG = AMAX1(FCHANG,ABS(FRT-FR(I,J)))
      FMAX = AMAX1(FMAX,FRT)
      FMIN = AMIN1(FMIN,FRT)
      FR(I,J) = FNEW*FRT+(1.-FNEW)*FR(I,J)
   40 CONTINUE
      IF (IEND.LT.1) WRITE(NWRIT,1030) FMAX,FMIN,FCHANG
C---PRINT DEBUG OUTPUT IF REQUESTED
      IF (IDEBUG.LE.0) RETURN
      IF ((ITER/IDEBUG)*IDEBUG.NE.ITER.AND.ITER.NE.1) RETURN
      WRITE(NWRIT,1010)
      WRITE(NWRIT,1000) ((I,J,WSUBS(I,J),WSUBT(I,J),VTH(I,J),RHO(I,J),
     1RHOAV(I,J),DELRHO(I,J),DLDU(I,J),I=1,MM),J=1,MHTP1)
```

```
      WRITE(NWRIT,1020)
      WRITE(NWRIT,1000) ((I,J,DTHDS(I,J),FR(I,J),DFDM(I,J),XIOM(I,J),
     1ZETOM(I,J),CAMP(I,J),SAMP(I,J),I=1,MM),J=1,MHTP1)
      RETURN
 1000 FORMAT(2I6,7G16.6)
 1010 FORMAT(1H1///35X,47HCHANGING  QUANTITIES  ON  THE  ORTHOGONAL  ME
     1SH/5X,1HI,5X,1HJ,6X,5HWSUBS,12X,5HWSUBT,12X,3HVTH,12X,3HRHO,12X,
     25HRHOAV,10X,6HDELRHO,11X,4HDLDU)
 1020 FORMAT(////5X,1HI,5X,1HJ,6X,5HDTHDS,12X,2HFR,13X,4HDFDM,12X,
     14HXIOM,12X,5HZETOM,11X,4HCAMP,12X,4HSAMP)
 1030 FORMAT(///5X,33HMAXIMUM CALCULATED VALUE OF FR  =,G13.5/5X,33HMINI
     1MUM CALCULATED VALUE OF FR  =,G13.5/5X,33HMAXIMUM CALCULATED CHANG
     2E IN FR =,G13.5)
      END
```

```
      SUBROUTINE ILETE
C
C--ILETE CALCULATES THE INTEGER ARRAYS OF MESH POINT LOCATIONS WHICH ARE
C--JUST INSIDE THE LEADING AND TRAILING EDGES OF THE BLADE
C
      COMMON/INPUTT/GAM,AR,MSFL,OMEGA,REDFAC,VELTOL,FNEW,DNEW,MBI,MBO,
     1   MM,MHT,NBL,NHUB,NTIP,NIN,NOUT,NBLPL,NPPP,NOSTAT,NSL,LSFR,
     2   LTPL,LAMVT,IMESH,ISLINE,ISTATL,IPLOT,ISUPER,ITSON,IDEBUG,
     3   ZOMIN,ZOMBI,ZOMBO,ZOMOUT,ZHIN,ZTIN,ZHOUT,ZTOUT,ZHUB(50),
     4   RHUB(50),ZTIP(50),RTIP(50),SFIN(50),RADIN(50),TIP(50),PRIP(50),
     5   LAMIN(50),VTHIN(50),SFOUT(50),RADOUT(50),PROP(50),LOSOUT(50),
     6   LAMOUT(50),VTHOUT(50),ZHST(50),ZTST(50),FLFR(50),
     7   ZBL(50,50),RBL(50,50),THBL(50,50),TNBL(50,50)
      COMMON/CALCON/MMM1,MHTP1,CP,EXPON,TGROG,PITCH,CURVHI,CURVTI,
     1   CURVHO,CURVTO,RHIN,RTIN,RHOUT,RTOUT,RLEH,RLET,RTEH,RTET,
     2   ZLE(50),RLE(50),ZTE(50),RTE(50),ZLEOM(101),RLEOM(101),
     3   SLEOM(101),THLEOM(101),ZTEOM(101),RTEOM(101),STEOM(101),
     4   THTEOM(101),ILE(101),ITE(101),ZOM(100,101),ROM(100,101),
     5   SOM(100,101),TOM(100,101),BTH(100,101),DTHDS(100,101),
     6   DTHDT(100,101),PLOSS(100,101),CPHI(100,101),SPHI(100,101)
      COMMON/SLCOM/ILS(50),ITS(50),ZSL(100,50),RSL(100,50),MSL(100,50),
     1   WZSL(100,50),WRSL(100,50),WMSL(100,50),WTHSL(100,50),
     2   ALPSL(100,50),BETSL(100,50),WSL(100,50),WWCRSL(100,50),
     3   CURVSL(100,50),WLSSL(100,50),WTSSL(100,50)
C--LEADING EDGE
      CALL SPLINT(RLE,ZLE,NBLPL,RLE(1),1,ZSPL,DZDR)
      DO 20 J=1,NSL
      I = 0
   10 I = I+1
      CALL SPLENT(RLE,ZLE,NBLPL,RSL(I,J),1,ZSPL,DZDR)
      IF (ZSPL.GT.ZSL(I,J)) GO TO 10
   20 ILS(J) = I
C--TRAILING EDGE
      CALL SPLINT(RTE,ZTE,NBLPL,RTE(1),1,ZSPL,DZDR)
      DO 40 J=1,NSL
      I = ILS(J)-1
   30 I = I+1
      CALL SPLENT(RTE,ZTE,NBLPL,RSL(I,J),1,ZSPL,DZDR)
      IF (ZSPL.GE.ZSL(I,J)) GO TO 30
```

```
   40 ITS(J) = I-1
      RETURN
      END




      SUBROUTINE TSONIN
C
C--TSONIN CALCULATES AND PRINTS OUT DATA AS INPUT TO THE
C--TSONIC  BLADE-TO-BLADE ANALYSIS PROGRAM
C
      COMMON SRW,SRE,ITER,IEND,NREAD,NWRIT
      COMMON/INPUTT/GAM,AR,MSFL,OMEGA,REDFAC,VELTOL,FNEW,DNEW,MBI,MBO,
     1   MM,MHT,NBL,NHUB,NTIP,NIN,NOUT,NBLPL,NPPP,NOSTAT,NSL,LSFR,
     2   LTPL,LAMVT,IMESH,ISLINE,ISTATL,IPLOT,ISUPER,ITSON,IDEBUG,
     3   ZOMIN,ZOMBI,ZOMBO,ZOMOUT,ZHIN,ZTIN,ZHOUT,ZTOUT,ZHUB(50),
     4   RHUB(50),ZTIP(50),RTIP(50),SFIN(50),RADIN(50),TIP(50),PRIP(50),
     5   LAMIN(50),VTHIN(50),SFOUT(50),RADOUT(50),PROP(50),LOSOUT(50),
     6   LAMOUT(50),VTHOUT(50),ZHST(50),ZTST(50),FLFR(50),
     7   ZBL(50,50),RBL(50,50),THBL(50,50),TNBL(50,50)
      COMMON/CALCON/MMM1,MHTP1,CP,EXPON,TGROG,PITCH,CURVHI,CURVTI,
     1   CURVHO,CURVTO,RHIN,RTIN,RHOUT,RTOUT,RLEH,RLET,RTEH,RTET,
     2   ZLE(50),RLE(50),ZTE(50),RTE(50),ZLEOM(101),RLEOM(101),
     3   SLEOM(101),THLEOM(101),ZTEOM(101),RTEOM(101),STEOM(101),
     4   THTEOM(101),ILE(101),ITE(101),ZOM(100,101),ROM(100,101),
     5   SOM(100,101),TOM(100,101),BTH(100,101),DTHDS(100,101),
     6   DTHDT(100,101),PLOSS(100,101),CPHI(100,101),SPHI(100,101)
      COMMON/VARCOM/A(4,100,101),UOM(100,101),K(100,101),RHO(100,101),
     1   WSUBS(100,101),WSUBT(100,101),WSUBZ(100,101),WSUBR(100,101),
     2   WSUBM(100,101),WTH(100,101),VTH(100,101),W(100,101),
     3   ALPHA(100,101),BETA(100,101),WWCR(100,101),CURV(100,101),
     4   WLSURF(100,101),WTSURF(100,101),CAMP(100,101),SAMP(100,101),
     5   RHOAV(100,101),DELRHO(100,101),FR(100,101),DFDM(100,101),
     6   XIOM(100,101),ZETOM(100,101),DLDU(100,101)
      COMMON/SLCOM/ILS(50),ITS(50),ZSL(100,50),RSL(100,50),MSL(100,50),
     1   WZSL(100,50),WRSL(100,50),WMSL(100,50),WTHSL(100,50),
     2   ALPSL(100,50),BETSL(100,50),WSL(100,50),WWCRSL(100,50),
     3   CURVSL(100,50),WLSSL(100,50),WTSSL(100,50)
      DIMENSION MSP(100),THSP1(100),THSP2(100),MR(100),RMSP(100),
     1   BESP(100),DIST(50),DTDS(50),ANG(50),DRTHBL(50,50),
     2   MLESP(5),MTESP(5),RTLEP1(5),RTLEP2(5),RTTEP1(5),RTTEP2(5)
      REAL MSFL,MSP,MR,MSL,MLESP,MTESP
      INTEGER BLDAT,AANDK,ERSOR,STRFN,SLCRD,SURVL
C
C--PRELIMINARY CALCULATIONS
      WRITE(NWRIT,1000)
      WTFL = MSFL/100.
      ORF = 0.
      DENTOL = 0.001
      NRSP = MM
      BLDAT = 1
      AANDK = 0
      ERSOR = 0
      STRFN = 2
      SLCRD = 2
      INTVL = 2
      SURVL = 3
      DO 20 JN=1,NBLPL
```

```
      DIST(1) = 0.
      DO 10 IN=2,NPPP
   10 DIST(IN) = DIST(IN-1)+SQRT((ZBL(IN,JN)-ZBL(IN-1,JN))**2+
     1(RBL(IN,JN)-RBL(IN-1,JN))**2)
      CALL SPLINE(DIST,THBL(1,JN),NPPP,DTDS,ANG)
      DO 20 IN=1,NPPP
      ANG(IN) = ATAN(RBL(IN,JN)*DTDS(IN))
   20 DRTHBL(IN,JN) = TNBL(IN,JN)/COS(ANG(IN))
C
C--CALCULATE AND PRINT OUT TSONIC DATA ALONG EACH OF THE STREAMLINES
C
      DO 70 JS=1,NSL
      II = 1
      JJ = 1
      TIPTEM = TIPF(FLFR(JS))
      RHOIP = RHOIPF(FLFR(JS))
C--INTERSECTION OF STREAMLINE WITH BLADE LEADING AND TRAILING EDGES
      CALL INRSCT(ZSL(1,JS),RSL(1,JS),MM,ZLE,RLE,NBLPL,ZLESL,RLESL)
      CALL INRSCT(ZSL(1,JS),RSL(1,JS),MM,ZTE,RTE,NBLPL,ZTESL,RTESL)
C--INLET AND OUTLET FLOW ANGLES
      CALL LININT(ZOM,ROM,BETA,MM,MHTP1,100,101,ZLESL,RLESL,BETAI,II,JJ)
      CALL LININT(ZOM,ROM,BETA,MM,MHTP1,100,101,ZTESL,RTESL,BETAO,II,JJ)
      BETAI = BETAI*57.295780
      BETAO = BETAO*57.295780
C
C--CALCULATE STREAMSHEET LOCATION AND THICKNESS
      DO 30 IS=1,MM
      MR(IS) = MSL(IS,JS)-MSL(1,JS)
      RMSP(IS) = RSL(IS,JS)
      CALL LININT(ZOM,ROM,RHO,MM,MHTP1,100,101,ZSL(IS,JS),RSL(IS,JS),
     1RHOSL,II,JJ)
      CALL LININT(ZOM,ROM,BTH,MM,MHTP1,100,101,ZSL(IS,JS),RSL(IS,JS),
     1BTHSL,II,JJ)
      CALL LININT(ZOM,ROM,PLOSS,MM,MHTP1,100,101,ZSL(IS,JS),RSL(IS,JS),
     1PLOSSL,II,JJ)
   30 BESP(IS) = WTFL/(RHOSL*WMSL(IS,JS)*RSL(IS,JS)*BTHSL)*(1.-PLOSSL)
C
C--CALCULATE BLADE SURFACE COORDINATES
      II = 1
      JJ = 1
      NBLPTS = ITS(JS)-ILS(JS)+3
      SPLNO1 = NBLPTS
      SPLNO2 = NBLPTS
      ILSJ = ILS(JS)
      ITSJ = ITS(JS)
      MSP(1) = 0.
      DELM = SQRT((ZSL(ILSJ,JS)-ZLESL)**2+(RSL(ILSJ,JS)-RLESL)**2)
      CALL LININT(ZBL,RBL,THBL,NPPP,NBLPL,50,50,ZLESL,RLESL,
     1THLESL,II,JJ)
      CALL LININT(ZBL,RBL,DRTHBL,NPPP,NBLPL,50,50,ZLESL,RLESL,
     1DRBL,II,JJ)
      DBL = DRBL/RLESL/2.
      THSP1(1) = DBL
      THSP2(1) = -DBL
      ISB = 2
      DO 40 IS=ILSJ,ITSJ
      MSP(ISB) = MR(IS)-MR(ILSJ)+DELM
      CALL LININT(ZBL,RBL,THBL,NPPP,NBLPL,50,50,ZSL(IS,JS),RSL(IS,JS),
     1THSL,II,JJ)
      CALL LININT(ZBL,RBL,DRTHBL,NPPP,NBLPL,50,50,ZSL(IS,JS),RSL(IS,JS),
```

```
     1DRBL,II,JJ)
      DBL = DRBL/RSL(IS,JS)/2.
      THSP1(ISB) = THSL-THLESL+DBL
      THSP2(ISB) = THSL-THLESL-DBL
   40 ISB = ISB+1
      DELM = SQRT((ZTESL-ZSL(ITSJ,JS))**2+(RTESL-RSL(ITSJ,JS))**2)
      MSP(NBLPTS) = MSP(NBLPTS-1)+DELM
      CHORDF = MSP(NBLPTS)
      CALL LININT(ZBL,RBL,THBL,NPPP,NBLPL,50,50,ZTESL,RTESL,
     1THTESL,II,JJ)
      CALL LININT(ZBL,RBL,DRTHBL,NPPP,NBLPL,50,50,ZTESL,RTESL,
     1DRBL,II,JJ)
      DBL = DRBL/RTESL/2.
      THSP1(NBLPTS) = THTESL-THLESL+DBL
      THSP2(NBLPTS) = THTESL-THLESL-DBL
C
C--SHIFT STREAMSHEET MERIDIONAL COORDINATES TO ORIGIN AT BLADE
C--LEADING EDGE
      DELM = MR(ILSJ)-MSP(2)
      DO 50 IS=1,MM
   50 MR(IS) = MR(IS)-DELM
C
C--CALCULATE SPECIAL ARRAYS OF LOCAL BLADE SURFACE R*THETA COORDINATES
C--AT LEADING AND TRAILING EDGES OF BLADE SECTION
      NSPTS = 5
      IF (NBLPTS.LT.5) NSPTS=NBLPTS
      TLEREF = (THSP1(1)+THSP2(1))/2.
      TTEREF = (THSP1(NBLPTS)+THSP2(NBLPTS))/2.
      DO 60 I=1,NSPTS
      J = NBLPTS-NSPTS+I
      MLESP(I) = MSP(I)
      CALL SPLINT(MR,RMSP,MM,MLESP(I),1,RLEP,DRDM)
      RTLEP1(I) = RLEP*(THSP1(I)-TLEREF)
      RTLEP2(I) = RLEP*(THSP2(I)-TLEREF)
      MTESP(I) = MSP(J)
      CALL SPLINT(MR,RMSP,MM,MTESP(I),1,RTEP,DRDM)
      RTTEP1(I) = RTEP*(THSP1(J)-TTEREF)
   60 RTTEP2(I) = RTEP*(THSP2(J)-TTEREF)
C
C--PRINT TSONIC DATA
      WRITE(NWRIT,1010) JS,FLFR(JS)
      WRITE(NWRIT,1020)
      WRITE(NWRIT,1160) GAM,AR,TIPTEM,RHOIP,WTFL,OMEGA,ORF
      WRITE(NWRIT,1030)
      WRITE(NWRIT,1170) BETAI,BETAO,CHORDF
      WRITE(NWRIT,1040)
      WRITE(NWRIT,1170) REDFAC,DENTOL
      WRITE(NWRIT,1050)
      WRITE(NWRIT,1180) NBL,NRSP
      WRITE(NWRIT,1060)
      WRITE(NWRIT,1190) SPLNO1
      WRITE(NWRIT,1070)
      WRITE(NWRIT,1170) (MSP(I),I=1,NBLPTS)
      WRITE(NWRIT,1080)
      WRITE(NWRIT,1170) (THSP1(I),I=1,NBLPTS)
      WRITE(NWRIT,1090)
      WRITE(NWRIT,1190) SPLNO2
      WRITE(NWRIT,1100)
      WRITE(NWRIT,1170) (MSP(I),I=1,NBLPTS)
      WRITE(NWRIT,1110)
```

```
      WRITE(NWRIT,1170) (THSP2(I),I=1,NBLPTS)
      WRITE(NWRIT,1120)
      WRITE(NWRIT,1170) (MR(I),I=1,MM)
      WRITE(NWRIT,1130)
      WRITE(NWRIT,1170) (RMSP(I),I=1,MM)
      WRITE(NWRIT,1140)
      WRITE(NWRIT,1170) (BESP(I),I=1,MM)
      WRITE(NWRIT,1150)
      WRITE(NWRIT,1200) BLDAT,AANDK,ERSOR,STRFN,SLCRC,INTVL,SURVL
C
C--PRINT SPECIAL LOCAL ARRAYS OF BLADE SURFACE DATA
      WRITE(NWRIT,1210)
      WRITE(NWRIT,1170) (MLESP(I),I=1,NSPTS)
      WRITE(NWRIT,1220)
      WRITE(NWRIT,1170) (RTLEP1(I),I=1,NSPTS)
      WRITE(NWRIT,1230)
      WRITE(NWRIT,1170) (RTLEP2(I),I=1,NSPTS)
      WRITE(NWRIT,1240)
      WRITE(NWRIT,1170) (MTESP(I),I=1,NSPTS)
      WRITE(NWRIT,1250)
      WRITE(NWRIT,1170) (RTTEP1(I),I=1,NSPTS)
      WRITE(NWRIT,1260)
      WRITE(NWRIT,1170) (RTTEP2(I),I=1,NSPTS)
      WRITE(NWRIT,1270)
   70 CONTINUE
      RETURN
C
C--FORMAT STATEMENTS
C
 1000 FORMAT (1H1///41X,39(1H*)/41X,39H***  INPUT DATA FOR TSONIC PROGRA
     1M   ***/41X,39(1H*)////)
 1010 FORMAT (4X,17HSTREAMLINE NUMBER,I3,23H  --  STREAM FUNCTION =,
     1F8.4//)
 1020 FORMAT (7X,3HGAM,14X,2HAR,13X,3HTIP,12X,5HRHOIP,12X,4HWTFL,27X,
     15HOMEGA,12X,3HORF)
 1030 FORMAT (6X,5HBETAI,10X,5HBETAO,11X,6HCHORDF,11X,5HSTGRF)
 1040 FORMAT (6X,6HREDFAC,10X,6HDENTOL)
 1050 FORMAT (6X,8HMBI  MBO,9X,18HMM  NBBI  NBL NRSP)
 1060 FORMAT (7X,3HRI1,12X,3HRO1,12X,5HBETI1,11X,5HBETO1,11X,6HSPLNO1)
 1070 FORMAT (7X,4HMSP1,2X,5HARRAY)
 1080 FORMAT (7X,5HTHSP1,2X,5HARRAY)
 1090 FORMAT (7X,3HRI2,12X,3HRO2,12X,5HBETI2,11X,5HBETO2,11X,6HSPLNO2)
 1100 FORMAT (7X,4HMSP2,2X,5HARRAY)
 1110 FORMAT (7X,5HTHSP2,2X,5HARRAY)
 1120 FORMAT (7X,9HMR   ARRAY)
 1130 FORMAT (7X,11HRMSP  ARRAY)
 1140 FORMAT (7X,11HBESP  ARRAY)
 1150 FORMAT (5X,47HBLDAT  AANDK  ERSOR  STRFN  SLCRC  INTVL  SURVL)
 1160 FORMAT (1X,5G16.7,16X,2G16.7)
 1170 FORMAT (1X,8G16.7)
 1180 FORMAT (30X,2I5)
 1190 FORMAT (65X,G16.7)
 1200 FORMAT (1X,7I7)
 1210 FORMAT (7X,5HMLESP,2X,5HARRAY)
 1220 FORMAT (7X,6HRTLEP1,2X,5HARRAY)
 1230 FORMAT (7X,6HRTLEP2,2X,5HARRAY)
 1240 FORMAT (7X,5HMTESP,2X,5HARRAY)
 1250 FORMAT (7X,6HRTTEP1,2X,5HARRAY)
 1260 FORMAT (7X,6HRTTEP2,2X,5HARRAY)
```

```
 1270 FORMAT (1H1)
      END
```

```
      SUBROUTINE INDEV
C
C--INDEV CALCULATES A CORRECTION TO DTHDS TO ALLOW FOR INCIDENCE AND
C--DEVIATION (AFTER BLOCKAGE CORRECTION)
C
      COMMON SRW,SRE,ITER,IEND,NREAC,NWRIT
      COMMON/INPUTT/GAM,AR,MSFL,OMEGA,REDFAC,VELTOL,FNEW,DNEW,MBI,MBO,
     1   MM,MHT,NBL,NHUB,NTIP,NIN,NOUT,NBLPL,NPPP,NOSTAT,NSL,LSFR,
     2   LTPL,LAMVT,IMESH,ISLINE,ISTATL,IPLCT,ISUPER,ITSCN,IDEBUG,
     3   ZOMIN,ZOMBI,ZOMBO,ZOMOUT,ZHIN,ZTIN,ZHOUT,ZTOUT,ZHUB(50),
     4   RHUB(50),ZTIP(50),RTIP(50),SFIN(50),RADIN(50),TIP(50),PRIP(50),
     5   LAMIN(50),VTHIN(50),SFOUT(50),RADOUT(50),PROP(50),LOSOUT(50),
     6   LAMOUT(50),VTHOUT(50),ZHST(50),ZTST(50),FLFR(50),
     7   ZBL(50,50),RBL(50,50),THBL(50,50),TNBL(50,50)
      COMMON/CALCON/MMM1,MHTP1,CP,EXPON,TGROG,PITCH,CURVHI,CURVTI,
     1   CURVHO,CURVTO,RHIN,RTIN,RHOUT,RTOUT,RLEH,RLET,RTEH,RTET,
     2   ZLE(50),RLE(50),ZTE(50),RTE(50),ZLEOM(101),RLEOM(101),
     3   SLEOM(101),THLEOM(101),ZTEOM(101),RTEOM(101),STEOM(101),
     4   THTEOM(101),ILE(101),ITE(101),ZOM(100,101),ROM(100,101),
     5   SOM(100,101),TOM(100,101),BTH(100,101),DTHDS(100,101),
     6   DTHDT(100,101),PLOSS(100,101),CPHI(100,101),SPHI(100,101)
      COMMON/VARCOM/A(4,100,101),UOM(100,101),K(100,101),RHO(100,101),
     1   WSUBS(100,101),WSUBT(100,101),WSUBZ(100,101),WSUBR(100,101),
     2   WSUBM(100,101),WTH(100,101),VTH(100,101),W(100,101),
     3   ALPHA(100,101),BETA(100,101),WWCR(100,101),CURV(100,101),
     4   WLSURF(100,101),WTSURF(100,101),CAMP(100,101),SAMP(100,101),
     5   RHOAV(100,101),DELRHO(100,101),FR(100,101),DFDM(100,101),
     6   XIOM(100,101),ZETOM(100,101),DLDU(100,101)
      COMMON/INDCOM/ZPC(11,50),RPC(11,50),DTHDZ(11,50),DTHDR(11,50)
      DIMENSION DTDSLE(101),DTDSTE(101)
      DEGRAD = 180./3.1415927
      II = 1
      JJ = 1
      IID = 1
      JJD = 1
      IF (IMESH.LE.0) GO TO 10
      IF ((ITER/IMESH)*IMESH.EQ.ITER.OR.ITER.EQ.1) GO TO 30
   10 IF (ISLINE.LE.0) GO TO 20
      IF ((ITER/ISLINE)*ISLINE.EQ.ITER.OR.ITER.EQ.1) GO TO 30
   20 IF (ISTATL.LE.0) GO TO 40
      IF ((ITER/ISTATL)*ISTATL.NE.ITER.AND.ITER.NE.1) GO TO 40
   30 WRITE(NWRIT,1010)
      IF (REDFAC.LT.1.0) WRITE(NWRIT,1100) ITER
      IF (REDFAC.EQ.1.0.AND.IEND.LE.0) WRITE(NWRIT,1110) ITER
      IF (REDFAC.EQ.1.0.AND.IEND.EQ.1) WRITE(NWRIT,1120)
      IF (REDFAC.EQ.1.0.AND.IEND.EQ.2) WRITE(NWRIT,1130)
      WRITE(NWRIT,1020)
   40 DO 120 J=1,MHTP1
C
C--CORRECT DTHDS FOR INCIDENCE AT BLADE LEADING EDGE
C
      I = ILE(J)-1
```

```
      EXTRAP = SLEOM(J)-SOM(I,J)
      BTFSLE = BETA(I,J)+EXTRAP*(BETA(I,J)-BETA(I-1,J))/(SOM(I,J)-
     1SOM(I-1,J))
      CALL LININT(ZOM,ROM,BTH,MM,MHTP1,100,101,ZLEOM(J),RLEOM(J),BTHLE,
     1II,JJ)
      TANBFL = TAN(BTFSLE)*BTHLE/PITCH
      SPHILE = SPHI(I,J)+EXTRAP*(SPHI(I+1,J)-SPHI(I,J))/(SOM(I+1,J)-
     1SOM(I,J))
      CPHILE = CPHI(I,J)+EXTRAP*(CPHI(I+1,J)-CPHI(I,J))/(SOM(I+1,J)-
     1SOM(I,J))
      ALPHLE = ALPHA(I,J)+EXTRAP*(ALPHA(I+1,J)-ALPHA(I,J))/(SOM(I+1,J)-
     1SOM(I,J))
      CALL LININT(ZPC,RPC,DTHDZ,11,NBLPL,11,50,ZLEOM(J),RLEOM(J),DTDZLE,
     1IID,JJD)
      CALL LININT(ZPC,RPC,DTHDR,11,NBLPL,11,50,ZLEOM(J),RLEOM(J),DTDRLE,
     1IID,JJD)
      DTDTLE = DTDRLE*CPHILE-DTDZLE*SPHILE
      IF(ITER.EQ.1) DTDSLE(J) = DTDRLE*SPHILE+DTDZLE*CPHILE
      TANBBL = RLEOM(J)*(DTDRLE*SIN(ALPHLE)+DTDZLE*COS(ALPHLE))
      BTABLD = ATAN(TANBBL)
      BLINC = (ATAN(TANBFL)-BTABLD)*DEGRAD
      UBINC = (BTFSLE-BTABLD)*DEGRAD
      EXTRAP = SOM(I+1,J)-SLEOM(J)
      SAMPLE = SAMP(I+1,J)+EXTRAP*(SAMP(I+1,J)-SAMP(I+2,J))/(SOM(I+2,J)-
     1SOM(I+1,J))
      CAMPLE = CAMP(I+1,J)+EXTRAP*(CAMP(I+1,J)-CAMP(I+2,J))/(SOM(I+2,J)-
     1SOM(I+1,J))
      DTDSFL = (TANBFL/RLEOM(J)-DTDTLE*SAMPLE)/CAMPLE
      BLDCRD = (RLEOM(J)+RTEOM(J))/2.*(THLEOM(J)-THTEOM(J))
      BLDCRD = SQRT(BLDCRD**2+(STEOM(J)-SLEOM(J))**2)
      SLIDLE = BLDCRD/PITCH/RLEOM(J)
      DISTLE = AMIN1(.5,AMAX1(1./6.,(11.-4.*SLIDLE)/18.))*(STEOM(J)-
     1SLEOM(J))
      I = ILE(J)
   50 SDIST = SLEOM(J)+DISTLE-SOM(I,J)
      IF(SDIST.LE.0.) GO TO 60
      DTHDS(I,J) = DTHDS(I,J)+(DTDSFL-DTDSLE(J))*SDIST/DISTLE
      I = I+1
      GO TO 50
   60 DTDSLE(J) = DTDSFL
C
C--CORRECT DTHDS FOR DEVIATION AT BLADE TRAILING EDGE
C
      I = ITE(J)+1
      EXTRAP = SOM(I,J)-STEOM(J)
      BTFSTE = BETA(I,J)+EXTRAP*(BETA(I,J)-BETA(I+1,J))/(SOM(I+1,J)-
     1SOM(I,J))
      CALL LININT(ZOM,ROM,BTH,MM,MHTP1,100,101,ZTEOM(J),RTEOM(J),BTHTE,
     1II,JJ)
      CALL LININT(ZOM,ROM,PLOSS,MM,MHTP1,100,101,ZTEOM(J),RTEOM(J),
     1PLOSTE,II,JJ)
      TANBFL = TAN(BTFSTE)*BTHTE/PITCH/(1.-PLOSTE)
      SPHITE = SPHI(I,J)+EXTRAP*(SPHI(I-1,J)-SPHI(I,J))/(SOM(I,J)-
     1SOM(I-1,J))
      CPHITE = CPHI(I,J)+EXTRAP*(CPHI(I-1,J)-CPHI(I,J))/(SOM(I,J)-
     1SOM(I-1,J))
      ALPHTE = ALPHA(I,J)+EXTRAP*(ALPHA(I-1,J)-ALPHA(I,J))/(SOM(I,J)-
     1SOM(I-1,J))
      CALL LININT(ZPC,RPC,DTHDZ,11,NBLPL,11,50,ZTEOM(J),RTEOM(J),DTDZTE,
```

```
     1IID,JJD)
      CALL LININT(ZPC,RPC,DTHDR,11,NBLPL,11,50,ZTEOM(J),RTEOM(J),DTDRTE,
     1IID,JJD)
      DTDTTE = DTDRTE*CPHITE-DTDZTE*SPHITE
      IF(ITER.EQ.1) DTDSTE(J) = DTDRTE*SPHITE+DTDZTE*CPHITE
      TANBBL = RTEOM(J)*(DTDRTE*SIN(ALPHTE)+DTDZTE*COS(ALPHTE))
      BTABLD = ATAN(TANBBL)
      BLDEV = (ATAN(TANBFL)-BTABLD)*DEGRAD
      UBDEV = (BTFSTE-BTABLD)*DEGRAD
      IF (IMESH.LE.0) GO TO 70
      IF ((ITER/IMESH)*IMESH.EQ.ITER.OR.ITER.EQ.1) GO TO 90
   70 IF (ISLINE.LE.0) GO TO 80
      IF ((ITER/ISLINE)*ISLINE.EQ.ITER.OR.ITER.EQ.1) GO TO 90
   80 IF (ISTATL.LE.0) GO TO 100
      IF ((ITER/ISTATL)*ISTATL.NE.ITER.AND.ITER.NE.1) GO TO 100
   90 WRITE(NWRIT,1000) J,BLINC,UBINC,BLDEV,UBDEV
  100 EXTRAP = SLEOM(J)-SOM(I-1,J)
      SAMPTE = SAMP(I-1,J)+EXTRAP*(SAMP(I-1,J)-SAMP(I-2,J))/(SOM(I-1,J)-
     1SOM(I-2,J))
      CAMPTE = CAMP(I-1,J)+EXTRAP*(CAMP(I-1,J)-CAMP(I-2,J))/(SOM(I-1,J)-
     1SOM(I-2,J))
      DTDSFL = (TANBFL/RTEOM(J)-DTDTTE*SAMPTE)/CAMPTE
      SLIDTE = BLDCRD/PITCH/RTEOM(J)
      DISTTE = AMIN1(.5,AMAX1(1./6.,(11.-4.*SLIDTE)/18.))*(STEOM(J)-
     1SLEOM(J))
      I = ITE(J)
  110 SDIST = SOM(I,J)-STEOM(J)+DISTTE
      IF(SDIST.LE.0.) GO TO 120
      DTHDS(I,J) = DTHDS(I,J)+(DTDSFL-DTDSTE(J))*SDIST/DISTTE
      I = I-1
      GO TO 110
  120 DTDSTE(J) = DTDSFL
      WRITE(NWRIT,1140)
      RETURN
C
C--FORMAT STATEMENTS
C
 1000 FORMAT (35X,1H*,2X,I3,3X,2(1H*,F9.2,2X,F9.2,4X),1H*)
 1010 FORMAT (1H1,44X,40H***   INCIDENCE AND DEVIATION ANGLES   ***/
     150X,30(1H*))
 1020 FORMAT (//35X,10H*  MESH  *,8X,9HINCIDENCE,7X,1H*,8X,9HDEVIATION,
     17X,1H*/35X,10H*  LINE  *,3X,7HBLOCKED,3X,9HUNBLOCKED,2X,1H*,3X,
     27HBLOCKED,3X,9HUNBLOCKED,2X,1H*)
 1100 FORMAT (/53X,23(1H*)/53X,23H*  REDUCED  MASSFLOW  */53X,23(1H*)/
     153X,18H*  ITERATION  NO. ,I2,3H  */53X,23(1H*))
 1110 FORMAT (/52X,25(1H*)/52X,25H*    FULL    MASSFLOW    */52X,25(1H*)/
     152X,19H*   ITERATION  NO. ,I2,4H   */52X,25(1H*))
 1120 FORMAT (/52X,25(1H*)/52X,25H*    FULL    MASSFLOW    */42X,45(1H*)/
     142X,1H*,12X,19HTRANSONIC  SOLUTION,12X,1H*/42X,45H*  BY VELOCITY G
     2RADIENT APPROXIMATE METHOD  */35X,59(1H*)/35X,59H*  ALL VELOCITIES
     3 SMALLER THAN CHOKING MASSFLOW SOLUTION  */35X,59(1H*))
 1130 FORMAT (/52X,25(1H*)/52X,25H*    FULL    MASSFLOW    */42X,45(1H*)/
     142X,1H*,12X,19HTRANSONIC  SOLUTION,12X,1H*/42X,45H*  BY VELOCITY G
     2RADIENT APPROXIMATE METHOD  */35X,59(1H*)/35X,59H*  ALL VELOCITIES
     3 LARGER THAN CHOKING MASSFLOW SOLUTION   */35X,59(1H*))
 1140 FORMAT (1H1)
      END
```

```
      SUBROUTINE SLPLOT
C
C--SLPLOT PLOTS THE STREAMLINES IN THE HUB-SHROUD FLOW PLANE
C
      COMMON SRW,SRE,ITER,IEND
      COMMON/INPUTT/GAM,AR,MSFL,OMEGA,REDFAC,VELTOL,FNEW,CNEW,MBI,MBO,
     1   MM,MHT,NBL,NHUB,NTIP,NIN,NOUT,NBLPL,NPPP,NOSTAT,NSL,LSFR,
     2   LTPL,LAMVT,IMESH,ISLINE,ISTATL,IPLOT,ISUPER,ITSON,IDEBUG,
     3   ZOMIN,ZOMBI,ZOMBO,ZOMOUT,ZHIN,ZTIN,ZHOUT,ZTOUT,ZHUB(50),
     4   RHUB(50),ZTIP(50),RTIP(50),SFIN(50),RADIN(50),TIP(50),PRIP(50),
     5   LAMIN(50),VTHIN(50),SFOUT(50),RADOUT(50),PROP(50),LOSOUT(50),
     6   LAMOUT(50),VTHOUT(50),ZHST(50),ZTST(50),FLFR(50),
     7   ZBL(50,50),RBL(50,50),THBL(50,50),TNBL(50,50)
      COMMON/SLCOM/ILS(50),ITS(50),ZSL(100,50),RSL(100,50),MSL(100,50),
     1   WZSL(100,50),WRSL(100,50),WMSL(100,50),WTHSL(100,50),
     2   ALPSL(100,50),BETSL(100,50),WSL(100,50),WWCRSL(100,50),
     3   CURVSL(100,50),WLSSL(100,50),WTSSL(100,50)
      COMMON/PLTCOM/ZLRNG,ZRRNG,RBRNG,RTRNG,ZHPLT(100),RHPLT(100),
     1   ZSPLT(100),RSPLT(100),ZLPLT(100),RLPLT(100),ZTPLT(100),
     2   RTPLT(100)
      DIMENSION TITL1(10),TITL2(3),TITL3(3),TITL4(11),TITL5(5)
      REAL MSL
      DATA TITL1/'STRE','AMLI','NE P','LOT$','C1$L','2IN ','MERI','DION'
     1,'AL P','LANE'/
      DATA TITL2/'Z  D','IREC','TION'/
      DATA TITL3/'R  D','IREC','TION'/
      DATA TITL4/'SUBS','ONIC','$C1S','OLUT','ION$','C2IT','ERAT','ION$'
     1,'C1NO','.','XXXX'/
      DATA TITL5/'TRAN','SONI','C$C1','SOLU','TION'/
      DATA SYM/'X'/
      IF (IPLOT.LE.0) RETURN
      IF((ITER/IPLOT)*IPLOT.NE.ITER.AND.ITER.NE.1) RETURN
C
C--PLOT THE ITERATION NUMBER
      CALL LRGRID(1,1,0.0,0.0)
      CALL LRCNVT(ITER,1,TITL4(11),1,4,0)
      IF (IEND.LE.0) CALL LRLEGN(TITL4,44,0,4.2,6.0,1.0)
      IF (IEND.GT.0) CALL LRLEGN(TITL5,20,0,4.2,5.5,1.0)
C
C--PLOT BLADE GEOMETRY AND STREAMLINES
C
      CALL LRMRGN(1.0,1.0,2.0,1.0)
      CALL LRANGE(ZLRNG,ZRRNG,RBRNG,RTRNG)
      CALL LRGRID(-1,-1,1.0,1.0)
      CALL LRLEGN(TITL1,40,0,3.5,0.7,0.0)
      CALL LRCHSZ(2)
      CALL LRLEGN(TITL2,12,0,4.5,1.5,0.0)
      CALL LRLEGN(TITL3,12,1,0.4,4.5,0.0)
      CALL LRCHSZ(4)
      CALL LRCURV(ZHPLT,RHPLT,100,2,SYM,0.0)
      CALL LRCURV(ZSPLT,RSPLT,100,2,SYM,0.0)
      CALL LRCURV(ZLPLT,RLPLT,100,2,SYM,0.0)
      CALL LRCURV(ZTPLT,RTPLT,100,2,SYM,0.0)
C--PLOT STREAMLINES
      EOP = 0.0
      NSL1 = NSL-1
      DO 10 JS=2,NSL1
      IF (JS.EQ.NSL1) EOP=1.0
```

```
   10 CALL LRCURV(ZSL(1,JS),RSL(1,JS),MM,2,SYM,ECP)
      CALL LRCURV(ZSL,RSL,0,1,SYM,1.C)
      RETURN
      END




      SUBROUTINE SVPLOT
C
C--SVPLOT PLOTS THE MEAN STREAM SURFACE AND BLADE SURFACE OUTPUT
C--VELOCITIES ALONG ALL STREAMLINES
C
      COMMON SRW,SRE,ITER,IEND
      COMMON/INPUTT/GAM,AR,MSFL,OMEGA,REDFAC,VELTOL,FNEW,ONEW,MBI,MBO,
     1   MM,MHT,NBL,NHUB,NTIP,NIN,NOUT,NBLPL,NPPP,NOSTAT,NSL,LSFR,
     2   LTPL,LAMVT,IMESH,ISLINE,ISTATL,IPLOT,ISUPER,ITSCN,IDEBUG,
     3   ZOMIN,ZOMBI,ZOMBO,ZOMOUT,ZHIN,ZTIN,ZHOUT,ZTOUT,ZHUB(50),
     4   RHUB(50),ZTIP(50),RTIP(50),SFIN(50),RADIN(50),TIP(50),PRIP(50),
     5   LAMIN(50),VTHIN(50),SFOUT(50),RADOUT(50),PRCP(50),LOSOUT(50),
     6   LAMOUT(50),VTHOUT(50),ZHST(50),ZTST(50),FLFR(50),
     7   ZBL(50,50),RBL(50,50),THBL(50,50),TNBL(50,50)
      COMMON/SLCOM/ILS(50),ITS(50),ZSL(100,50),RSL(100,50),MSL(100,50),
     1   WZSL(100,50),WRSL(100,50),WMSL(100,50),WTHSL(100,50),
     2   ALPSL(100,50),BETSL(100,50),WSL(100,50),WWCRSL(100,50),
     3   CURVSL(100,50),WLSSL(100,50),WTSSL(100,50)
      DIMENSION TITL1(12),TITL2(9),TITL3(14),TITL4(15),
     1   TITL5(16),TITL6(6),TITL7(2)
      REAL MSL,LRNG
      DATA TITL1/'MERI','DION','AL A','ND S','URFA','CE$C','1$RI','RELA'
     1,'TIVE',' VEL','OCIT','IES '/
      DATA TITL2/'  ST','REAM','LINE',' NO.','XXXX','    ','U = ','XXXX'
     1,'XXXX'/
      DATA TITL3/'MERI','DION','AL R','ELAT','IVE ','VELC','CITI','ES$C'
     1,'1$R6','FOR ','ALL ','STRE','AMLI','NES '/
      DATA TITL4/'SUCT','ION ','SURF','ACE ','RELA','TIVE',' VEL','OCIT'
     1,'IES$','C1$R','8FOR',' ALL',' STR','EAML','INES'/
      DATA TITL5/'PRES','SURE',' SUR','FACE',' REL','ATIV','E VE','LOCI'
     1,'TIES','$C1$','R8FO','R AL','L ST','REAM','LINE','S   '/
      DATA TITL6/'  ME','RIDI','ONAL','  CO','ORDI','NATE'/
      DATA TITL7/'VELO','CITY'/
      DATA SYM/'X'/
      IF (IPLOT.LE.0) RETURN
      IF ((ITER/IPLOT)*IPLOT.NE.ITER.AND.ITER.NE.1) RETURN
C
C--COMPUTE RANGE OF PLOTS, AND SET UP FOR PLOTTING
C
      LRNG = MSL(1,1)
      RRNG = MSL(1,1)
      BRNG = 1000.
      TRNG = 0.
      DO 30 JS=1,NSL
      LRNG = AMIN1(LRNG,MSL(1,JS))
      RRNG = AMAX1(RRNG,MSL(MM,JS))
      ILSJ = ILS(JS)
      ITSJ = ITS(JS)
      DO 10 IS=ILSJ,ITSJ
      BRNG = AMIN1(BRNG,WLSSL(IS,JS))
      BRNG = AMIN1(BRNG,WTSSL(IS,JS))
```

```
      TRNG = AMAX1(TRNG,WLSSL(IS,JS))
   10 TRNG = AMAX1(TRNG,WTSSL(IS,JS))
      DO 20 IS=1,MM
      BRNG = AMIN1(BRNG,WSL(IS,JS))
   20 TRNG = AMAX1(TRNG,WSL(IS,JS))
   30 CONTINUE
      CALL LRMRGN(1.0,1.0,2.0,1.0)
      CALL LRANGE(LRNG,RRNG,BRNG,TRNG)
      CALL LRGRID(1,1,11.0,11.0)
C
C--PLOT VELOCITIES ON EACH STREAMLINE
C
      DO 40 JS=1,NSL
      ILSJ = ILS(JS)
      MBLD = ITS(JS)-ILS(JS)+1
      IF (JS.EQ.1) CALL LRLEGN(TITL1,48,0,2.5,0.7,0.0)
      CALL LRCHSZ(3)
      CALL LRCNVT(JS,1,TITL2(5),1,4,0)
      CALL LRCNVT(FLFR(JS),3,TITL2(8),3,8,4)
      CALL LRLEGN(TITL2,36,0,2.2,9.5,0.0)
      CALL LRCHSZ(2)
      CALL LRLEGN(TITL6,24,0,3.4,1.3,0.0)
      CALL LRLEGN(TITL7,8,1,0.2,4.9,0.0)
      CALL LRCHSZ(4)
      CALL LRCURV(MSL(1,JS),WSL(1,JS),MM,2,SYM,0.0)
      CALL LRCURV(MSL(1,JS),WSL(1,JS),MM,4,SYM,0.0)
      CALL LRCURV(MSL(ILSJ,JS),WLSSL(ILSJ,JS),MBLD,2,SYM,0.0)
      CALL LRCURV(MSL(ILSJ,JS),WLSSL(ILSJ,JS),MBLD,4,SYM,0.0)
      CALL LRCURV(MSL(ILSJ,JS),WTSSL(ILSJ,JS),MBLD,2,SYM,0.0)
   40 CALL LRCURV(MSL(ILSJ,JS),WTSSL(ILSJ,JS),MBLD,4,SYM,1.0)
C
C--PLOT MERIDIONAL VELOCITIES FOR ALL STREAMLINES
C
      CALL LRGRID(3,3,11.0,11.0)
      CALL LRLEGN(TITL3,56,0,1.7,0.7,0.0)
      CALL LRCHSZ(2)
      CALL LRLEGN(TITL6,24,0,3.4,1.3,0.0)
      CALL LRLEGN(TITL7,8,1,0.2,4.9,0.0)
      CALL LRCHSZ(4)
      EOP = 0.0
      DO 50 JS=1,NSL
      IF (JS.EQ.NSL) EOP=1.0
   50 CALL LRCURV(MSL(1,JS),WSL(1,JS),MM,2,SYM,EOP)
C
C--PLOT SUCTION SURFACE VELOCITIES FOR ALL STREAMLINES
C
      CALL LRLEGN(TITL4,60,0,1.2,0.7,0.0)
      CALL LRCHSZ(2)
      CALL LRLEGN(TITL6,24,0,3.4,1.3,0.0)
      CALL LRLEGN(TITL7,8,1,0.2,4.9,0.0)
      CALL LRCHSZ(4)
      EOP = 0.0
      DO 60 JS=1,NSL
      IF (JS.EQ.NSL) EOP=1.0
      ILSJ = ILS(JS)
      MBLD = ITS(JS)-ILS(JS)+1
   60 CALL LRCURV(MSL(ILSJ,JS),WLSSL(ILSJ,JS),MBLD,2,SYM,EOP)
C
```

```
C--PLOT PRESSURE SURFACE VELOCITIES FOR ALL STREAMLINES
C
      CALL LRLEGN(TITL5,64,0,1.2,0.7,0.0)
      CALL LRCHSZ(2)
      CALL LRLEGN(TITL6,24,0,3.4,1.3,0.0)
      CALL LRLEGN(TITL7,8,1,0.2,4.9,0.0)
      CALL LRCHSZ(4)
      EOP = 0.0
      DO 70 JS=1,NSL
      IF (JS.EQ.NSL) EOP=1.0
      ILSJ = ILS(JS)
      MBLD = ITS(JS)-ILS(JS)+1
   70 CALL LRCURV(MSL(ILSJ,JS),WTSSL(ILSJ,JS),MBLD,2,SYM,EOP)
      CALL LRCURV(ZSL,RSL,0,1,SYM,1.0)
      RETURN
      END




      SUBROUTINE TVELCY
C
C--TVELCY CALCULATES THE FULL MASSFLOW, TRANSONIC SOLUTION
C--USING VELOCITY GRADIENT EQUATIONS
C
      COMMON SRW,SRE,ITER,IEND,NREAD,NWRIT
      COMMON/INPUTT/GAM,AR,MSFL,OMEGA,REDFAC,VELTOL,FNEW,ONEW,MBI,MBO,
     1   MM,MHT,NBL,NHUB,NTIP,NIN,NOUT,NBLPL,NPPP,NOSTAT,NSL,LSFR,
     2   LTPL,LAMVT,IMESH,ISLINE,ISTATL,IPLOT,ISUPER,ITSON,IDEBUG,
     3   ZOMIN,ZOMBI,ZOMBO,ZOMOUT,ZHIN,ZTIN,ZHOUT,ZTOUT,ZHUB(50),
     4   RHUB(50),ZTIP(50),RTIP(50),SFIN(50),RADIN(50),TIP(50),PRIP(50),
     5   LAMIN(50),VTHIN(50),SFOUT(50),RADOUT(50),PROP(50),LOSOUT(50),
     6   LAMOUT(50),VTHOUT(50),ZHST(50),ZTST(50),FLFR(50),
     7   ZBL(50,50),RBL(50,50),THBL(50,50),TNBL(50,50)
      COMMON/CALCON/MMM1,MHTP1,CP,EXPON,TGROG,PITCH,CURVHI,CURVTI,
     1   CURVHO,CURVTO,RHIN,RTIN,RHOUT,RTOUT,RLEH,RLET,RTEH,RTET,
     2   ZLE(50),RLE(50),ZTE(50),RTE(50),ZLEOM(101),RLEOM(101),
     3   SLEOM(101),THLEOM(101),ZTEOM(101),RTEOM(101),STEOM(101),
     4   THTEOM(101),ILE(101),ITE(101),ZOM(100,101),ROM(100,101),
     5   SOM(100,101),TOM(100,101),BTH(100,101),DTHDS(100,101),
     6   DTHDT(100,101),PLOSS(100,101),CPHI(100,101),SPHI(100,101)
      COMMON/VARCOM/A(4,100,101),UOM(100,101),K(100,101),RHO(100,101),
     1   WSUBS(100,101),WSUBT(100,101),WSUBZ(100,101),WSUBR(100,101),
     2   WSUBM(100,101),WTH(100,101),VTH(100,101),W(100,101),
     3   ALPHA(100,101),BETA(100,101),WWCR(100,101),CURV(100,101),
     4   WLSURF(100,101),WTSURF(100,101),CAMP(100,101),SAMP(100,101),
     5   RHOAV(100,101),DELRHO(100,101),FR(100,101),DFDM(100,101),
     6   XIOM(100,101),ZETOM(100,101),DLDU(100,101)
      DIMENSION DWMDS(100),DWTDS(100),TVERT(101),
     1   WMVERT(101),WTVERT(101),TWLMR(101),CPTIP(101),RCARB(101),
     2   DWMVER(101),DWTVER(101),ATVEL(101),BTVEL(101),CTVEL(101),
     3   DTVEL(101),ETVEL(101),FTVEL(101),LAMBDA(101),LAMBDO(101),
     4   TIPT(101),TOP(101),RHOIP(101),RHOOP(101),
     5   DWMDM(100,101),DWTDM(100,101),DWMDT(100,101),DWTDT(100,101)
      REAL MSFL,LAMBDA,LAMBDO,LAMOUT,LAMIN,LAMDAF
      INTEGER SRW,SRE
      LOGICAL REPEAT
C
```

```
C--RESTORE FULL MASS FLOW VALUES, AND REINITIALIZE LAMDAF AND RVTHTA
C
      IEND = IEND+1
      JZ = 1
      IF (REDFAC.EQ.1.0) JZ=2
      IF (REDFAC.EQ.1.0) GO TO 60
      WRITE(NWRIT,1040)
      OMEGA = OMEGA/REDFAC
      MSFL = MSFL/REDFAC
      DO 10 J =1,NIN
      LAMIN(J) = LAMIN(J)/REDFAC
   10 VTHIN(J) = VTHIN(J)/REDFAC
      DO 20 J =1,NOUT
      LAMOUT(J) = LAMOUT(J)/REDFAC
   20 VTHOUT(J) = VTHOUT(J)/REDFAC
      CALL LAMNIT
      CALL RVTNIT
C
C--CALCULATE PARTIALS WITH RESPECT TO T OF WSUBM AND WSUBT
C
      DO 40 I=1,MM
      DO 30 J=1,MHTP1
      TVERT(J) = TOM(I,J)
      WMVERT(J) = WSUBM(I,J)
   30 WTVERT(J) = WTH(I,J)
      CALL SLOPES(TVERT,WMVERT,MHTP1,DWMVER)
      CALL SLOPES(TVERT,WTVERT,MHTP1,DWTVER)
      DO 40 J=1,MHTP1
      DWMDT(I,J) = DWMVER(J)
   40 DWTDT(I,J) = DWTVER(J)
C
C--CALCULATE PARTIALS WITH RESPECT TO S OF WSUBM AND WSUBT, AND THEN
C--CALCULATE PARTIALS WITH RESPECT TO M OF WSUBM AND WSUBT
C
      DO 50 J=1,MHTP1
      CALL SLOPES(SOM(1,J),WSUBM(1,J),MM,DWMDS)
      CALL SLOPES(SOM(1,J),WTH(1,J),MM,DWTDS)
      DO 50 I=1,MM
      DWMDM(I,J) = (DWMDS(I)*CAMP(I,J)+DWMDT(I,J)*SAMP(I,J))/REDFAC
   50 DWTDM(I,J) = (DWTDS(I)*CAMP(I,J)+DWTDT(I,J)*SAMP(I,J))/REDFAC
      RTOLER = 1.E-4
      CHLIM = MSFL
      MEAN = MHT/2+1
C
C--SOLVE VELOCITY GRADIENT EQUATION ON EACH VERTICAL MESH LINE
C
   60 DO 280 I=1,MM
      WHUB = W(I,1)/REDFAC
      DELMAX = W(I,MEAN)/20./REDFAC
      NCOUNT = 0
C
C--CALCULATE COEFFICIENTS A, B, AND C FOR THE VELOCITY GRADIENT EQUATION
C--INITIALIZE COEFFICIENT C TO ZERO
      DO 80 J=1,MHTP1
      LAMBDA(J) = LAMDAF(UOM(I,J),I,J)
      LAMBDO(J) = RVTHTA(UOM(I,J),I,J)
      TIPT(J) = TIPF(UOM(I,J))
      TOP(J) = TOPF(UOM(I,J))
      RHOIP(J) = RHOIPF(UOM(I,J))
      RHOOP(J) = RHOOPF(UOM(I,J))
```

```
      BTVEL(J) = 0.
      CTVEL(J) = 0.
      DTVEL(J) = 0.
      IF(I.LT.ILE(J).OR.I.GT.ITE(J)) GO TO 70
      SAL = SIN(ALPHA(I,J))
      SBETA = SIN(BETA(I,J))
      CBETA = COS(BETA(I,J))
      ATVEL(J) = CBETA**2*CAMP(I,J)*CURV(I,J)-SBETA**2*CPHI(I,J)/
     1ROM(I,J)+DTHDT(I,J)*SAL*CBETA*SBETA
      BTVEL(J) = CBETA*SAMP(I,J)*DWMDM(I,J)-2.*OMEGA*SBETA*CPHI(I,J)
     1+ROM(I,J)*DTHDT(I,J)*CBETA*(DWTDM(I,J)+2.*OMEGA*SAL)
      GO TO 80
   70 ATVEL(J) = CAMP(I,J)*CURV(I,J)
      DTVEL(J) = DWMDM(I,J)*SAMP(I,J)
   80 CONTINUE
C
C--CALCULATE C COEFFICIENT FOR THE VELOCITY GRADIENT EQUATION AND OTHER
C--CONSTANTS FOR CHECKING CONTINUITY
   90 DO 120 J=1,MHTP1
      OMR2 = OMEGA*ROM(I,J)**2
      TWLMR(J) = 2.*OMEGA*LAMBDA(J)-OMEGA*OMR2
      CPTIP(J) = 2.*CP*TIPT(J)
      IF(I.GE.ILE(J)) GO TO 100
      WHIRL = LAMBDA(J)
      TEMPER = TIPT(J)
      DENS = RHOIP(J)
      GO TO 110
  100 IF(I.LE.ITE(J)) GO TO 120
      WHIRL = LAMBDO(J)
      TEMPER = TOP(J)
      DENS = RHOOP(J)
  110 CTVEL(J) = -(WHIRL-OMR2)/ROM(I,J)**2*(CURV(I,J)*(WHIRL-OMR2)*
     1CAMP(I,J)+(WHIRL+OMR2)/ROM(I,J)*CPHI(I,J))
  120 RCARB(J) = RHOIP(J)*CAMP(I,J)*ROM(I,J)*BTH(I,J)
C
C--CALCULATE COEFFICIENTS E AND F FOR THE VELOCITY GRADIENT EQUATION
      TPP = TIPT(1)-TWLMR(1)/2./CP
      IF(TPP.LT.0.) GO TO 290
      PREL = RHOIP(1)*AR*TIPT(1)*(TPP/TIPT(1))**(GAM*EXPON)*(1.-
     1PLOSS(I,1))
      DO 130 J=2,MHTP1
      DTIP = TIPT(J)-TIPT(J-1)
      DLAM = LAMBDA(J)-LAMBDA(J-1)
      TPPN = TIPT(J)-TWLMR(J)/2./CP
      IF (TPPN.LT.0.) GO TO 290
      PRELN = RHOIP(J)*AR*TIPT(J)*(TPPN/TIPT(J))**(GAM*EXPON)*(1.-
     1PLOSS(I,J))
      DTPP = TPPN-TPP
      DPREL = PRELN-PREL
      ETVEL(J-1) = CP*DTIP-OMEGA*DLAM-CP*DTPP+AR/(PRELN+PREL)*(TPPN+TPP)
     1*DPREL
      FTVEL(J-1) =  DTPP/(TPPN+TPP)-AR/CP*DPREL/(PRELN+PREL)
      TPP = TPPN
  130 PREL = PRELN
C
C--OBTAIN NUMERICAL SOLUTION TO THE VELOCITY GRADIENT EQUATION
C--FOR AN ESTIMATED VALUE OF W AT THE HUB
C
      REPEAT = .FALSE.
```

```
  140 IND = 1
  150 W(I,1) = WHUB
      NCOUNT = NCOUNT+1
      IF (I.GE.ILE(1).AND.I.LE.ITE(1)) GO TO 160
      WHIRL = LAMBDA(1)
      IF (I.GT.ITE(1)) WHIRL = LAMBDO(1)
      SBETA = (WHIRL/ROM(I,1)-OMEGA*ROM(I,1))/WHUB
      IF(ABS(SBETA).GT.1.) GO TO 210
      BETA(I,1) = ARSIN(SBETA)
  160 CBETA = COS(BETA(I,1))
  170 WSQ = WHUB**2
      TTIP = 1.-(WSQ+TWLMR(1))/CPTIP(1)
      IF(TTIP.LT.0.) GO TO 220
      RVA = TTIP**EXPON*WHUB*CBETA*RCARB(1)
      DO 200 J=1,MHT
      DELTA = TOM(I,J+1)-TOM(I,J)
      WAS = W(I,J)+(ATVEL(J)*W(I,J)+BTVEL(J)+CTVEL(J)/W(I,J)+CBETA*
     1DTVEL(J))*DELTA+ETVEL(J)/W(I,J)+FTVEL(J)*W(I,J)
      IF (I.GE.ILE(J+1).AND.I.LE.ITE(J+1)) GO TO 180
      WHIRL = LAMBDA(J+1)
      IF (I.GT.ITE(J+1)) WHIRL = LAMBDO(J+1)
      WTHETA = (WHIRL/ROM(I,J+1)-OMEGA*ROM(I,J+1))
      SBETA = WTHETA/WAS
      IF(ABS(SBETA).GT.1.) GO TO 210
      BETA(I,J+1) = ARSIN(SBETA)
  180 CBETA = COS(BETA(I,J+1))
      WASS = W(I,J)+(ATVEL(J+1)*WAS+BTVEL(J+1)+CTVEL(J+1)/WAS+CBETA*
     1DTVEL(J+1))*DELTA+ETVEL(J)/WAS+FTVEL(J)*WAS
      W(I,J+1) = (WAS+WASS)/2.
      WSQ = W(I,J+1)**2
      TTIP = 1.-(WSQ+TWLMR(J+1))/CPTIP(J+1)
      IF(TTIP.LT.0.) GO TO 220
      IF(I.GE.ILE(J+1).AND.I.LE.ITE(J+1)) GO TO 190
      SBETA = WTHETA/W(I,J+1)
      IF(ABS(SBETA).GT.1.) GO TO 210
      BETA(I,J+1) = ARSIN(SBETA)
  190 CBETA = COS(BETA(I,J+1))
      RVAS = TTIP**EXPON*W(I,J+1)*CBETA*RCARB(J+1)
      UOM(I,J+1) = (RVA+RVAS)*DELTA/2.+UOM(I,J)
  200 RVA = RVAS
C
C--CHECK CONTINUITY AND ESTIMATE NEW VALUE FOR W AT THE HUB
C
      IF(IND.GE.6.AND.ABS(MSFL-UOM(I,MHTP1)).LE.MSFL*RTOLER) GO TO 250
      CALL CONTIN(WHUB,UOM(I,MHTP1),IND,JZ,MSFL,DELMAX)
      IF(IND.LT.10) GO TO 150
      IF(IND.EQ.10) GO TO 250
      GO TO 230
  210 WHUB = WHUB+0.5*DELMAX
      IF(NCOUNT.LT.1000) GO TO 140
      GO TO 230
  220 WHUB = WHUB-0.5*DELMAX
      IF(NCOUNT.LT.1000) GO TO 140
  230 WRITE (NWRIT,1010) I
      IMESH = 1
      ISLINE = 0
      ISTATL = 0
      DO 240 J=1,MHTP1
  240 UOM(I,J) = UOM(I,J)/MSFL
      GO TO 280
```

```
C
C---SOLUTION OBTAINED, CHECK ACCURACY OF TIP, LAMBCA, AND RHOIP
C
  250 CONTINUE
      DO 260 J=2,MHTP1
      UOM(I,J) = UOM(I,J)/MSFL
      TVAR = TIPF(UOM(I,J))
      IF(ABS(TVAR-TIPT(J)).GT.TVAR*RTOLER) REPEAT = .TRUE.
      TIPT(J) = TVAR
      TVAR = TOPF(UOM(I,J))
      IF(ABS(TVAR-TOP(J)).GT.TVAR*RTOLER) REPEAT = .TRUE.
      TOP(J) = TVAR
      TVAR = RHOIPF(UOM(I,J))
      IF (ABS(TVAR-RHOIP(J)).GT.TVAR*RTOLER) REPEAT = .TRUE.
      RHOIP(J) = TVAR
      TVAR = RHOOPF(UOM(I,J))
      IF (ABS(TVAR-RHOOP(J)).GT.TVAR*RTOLER) REPEAT = .TRUE.
      RHOOP(J) = TVAR
      TVAR = LAMDAF(UOM(I,J),I,J)
      IF(ABS(TVAR-LAMBDA(J)).GT.ABS(TVAR)*RTOLER) REPEAT = .TRUE.
      LAMBDA(J) = TVAR
      TVAR = RVTHTA(UOM(I,J),I,J)
      IF(ABS(TVAR-LAMBDO(J)).GT.ABS(TVAR)*RTOLER) REPEAT = .TRUE.
  260 LAMBDO(J) = TVAR
      WHUB = W(I,1)
      IF(REPEAT.AND.NCOUNT.LT.1000) GO TO 90
      IF(IND.NE.10) GO TO 270
      CHFL = UOM(I,MHTP1)*MSFL*FLOAT(NBL)
      CHLIM = AMIN1(CHLIM,CHFL)
      WRITE(NWRIT,1000) I,CHFL
  270 IF(REPEAT) WRITE(NWRIT,1010) I
  280 CONTINUE
C
C--FINISHED VELOCITY GRADIENT SOLUTION ON EACH VERTICAL MESH LINE
C--CHECK CHOKE LIMIT
      IF (CHLIM.GT.(0.9999*MSFL)) RETURN
      ISUPER = 2
      WRITE(NWRIT,1030) MSFL,CHLIM
      RETURN
  290 WRITE(NWRIT,1020)
      STOP
C
C--FORMAT STATEMENTS
C
 1000 FORMAT (69HLMSFL EXCEEDS CHOKING MASS FLOW FOR VERTICAL ORTHOGONAL
     1 MESH LINE I =,I3/22H   CHOKING MASS FLOW =,G15.6)
 1010 FORMAT (88HL  A VELOCITY GRADIENT SOLUTION CANNOT BE OBTAINED FOR
     1VERTICAL ORTHOGONAL MESH LINE I =,I3/4X,56HANY SUBSEQUENT OUTPUT F
     2OR THAT MESH LINE MAY BE IN ERROR)
 1020 FORMAT (62HL THE UPSTREAM INPUT WHIRL OR TANGENTIAL VELOCITY IS TO
     1O LARGE)
 1030 FORMAT (51HL  CHOKING MASSFLOW IS LESS THAN THE INPUT MASSFLOW/6X,
     116HINPUT MASSFLOW =,G13.5/6X,26HMINIMUM CHOKING MASSFLOW =,G13.5/
     26X,92HA SOLUTION CAN ONLY BE OBTAINED IF INPUT MASSFLOW IS LESS TH
     3AN THIS MINIMUM CHOKING MASSFLOW)
 1040 FORMAT (//52X,25(1H*)/52X,25H*    FULL   MASSFLOW    */42X,45(1H*)
     1/42X,1H*,12X,19HTRANSONIC  SOLUTION,12X,1H*/42X,45H*  BY VELOCITY
     2GRADIENT APPROXIMATE METHOD  */42X,45(1H*)/////)
      END
```

```
      FUNCTION TOPF(SF)
C
C--TOPF CALCULATES DOWNSTREAM ABSOLUTE TOTAL TEMPERATURE
C--AS A FUNCTION OF STREAM FUNCTION
C
      COMMON/INPUTT/GAM,AR,MSFL,OMEGA,REDFAC,VELTOL,FNEW,ONEW,MBI,MBO,
     1   MM,MHT,NBL,NHUB,NTIP,NIN,NOUT,NBLPL,NPPP,NOSTAT,NSL,LSFR,
     2   LTPL,LAMVT,IMESH,ISLINE,ISTATL,IPLOT,ISUPER,ITSON,IDEBUG,
     3   ZOMIN,ZOMBI,ZOMBO,ZOMOUT,ZHIN,ZTIN,ZHOUT,ZTOUT,ZHUB(50),
     4   RHUB(50),ZTIP(50),RTIP(50),SFIN(50),RADIN(50),TIP(50),PRIP(50),
     5   LAMIN(50),VTHIN(50),SFOUT(50),RADOUT(50),PROP(50),LOSOUT(50),
     6   LAMOUT(50),VTHOUT(50),ZHST(50),ZTST(50),FLFR(50),
     7   ZBL(50,50),RBL(50,50),THBL(50,50),TNBL(50,50)
      COMMON/CALCON/MMM1,MHTP1,CP,EXPON,TGROG,PITCH,CURVHI,CURVTI,
     1   CURVHO,CURVTO,RHIN,RTIN,RHOUT,RTOUT,RLEH,RLET,RTEH,RTET,
     2   ZLE(50),RLE(50),ZTE(50),RTE(50),ZLEOM(101),RLEOM(101),
     3   SLEOM(101),THLEOM(101),ZTEOM(101),RTEOM(101),STEOM(101),
     4   THTEOM(101),ILE(101),ITE(101),ZOM(100,101),ROM(100,101),
     5   SOM(100,101),TOM(100,101),BTH(100,101),DTHDS(100,101),
     6   DTHDT(100,101),PLOSS(100,101),CPHI(100,101),SPHI(100,101)
      REAL LAMDAF
      TOPF = TIPF(SF)-OMEGA/CP*(LAMDAF(SF,ILE(1),1)-RVTHTA(SF,ILE(1),1))
      RETURN
      END
```

```
      FUNCTION TIPF(SF)
C
C--TIPF CALCULATES UPSTREAM ABSOLUTE TOTAL TEMPERATURE
C--AS A FUNCTION OF STREAM FUNCTION
C
      COMMON/INPUTT/GAM,AR,MSFL,OMEGA,REDFAC,VELTOL,FNEW,ONEW,MBI,MBO,
     1   MM,MHT,NBL,NHUB,NTIP,NIN,NOUT,NBLPL,NPPP,NOSTAT,NSL,LSFR,
     2   LTPL,LAMVT,IMESH,ISLINE,ISTATL,IPLOT,ISUPER,ITSON,IDEBUG,
     3   ZOMIN,ZOMBI,ZOMBO,ZOMOUT,ZHIN,ZTIN,ZHOUT,ZTOUT,ZHUB(50),
     4   RHUB(50),ZTIP(50),RTIP(50),SFIN(50),RADIN(50),TIP(50),PRIP(50),
     5   LAMIN(50),VTHIN(50),SFOUT(50),RADOUT(50),PROP(50),LOSOUT(50),
     6   LAMOUT(50),VTHOUT(50),ZHST(50),ZTST(50),FLFR(50),
     7   ZBL(50,50),RBL(50,50),THBL(50,50),TNBL(50,50)
      COMMON/CALCON/MMM1,MHTP1,CP,EXPON,TGROG,PITCH,CURVHI,CURVTI,
     1   CURVHO,CURVTO,RHIN,RTIN,RHOUT,RTOUT,RLEH,RLET,RTEH,RTET,
     2   ZLE(50),RLE(50),ZTE(50),RTE(50),ZLEOM(101),RLEOM(101),
     3   SLEOM(101),THLEOM(101),ZTEOM(101),RTEOM(101),STEOM(101),
     4   THTEOM(101),ILE(101),ITE(101),ZOM(100,101),ROM(100,101),
     5   SOM(100,101),TOM(100,101),BTH(100,101),DTHDS(100,101),
     6   DTHDT(100,101),PLOSS(100,101),CPHI(100,101),SPHI(100,101)
      DIMENSION SLOPE(50),EM(50)
      K = 2
      IF(ABS(SF-SFIN(1)).GT.TOLER) GO TO 10
      TIPF = TIP(1)
      RETURN
   10 IF(SF-SFIN(1)) 20,20,30
   20 TIPF = TIP(1)+(SF-SFIN(1))*SLOPE(1)
      RETURN
   30 IF(ABS(SF-SFIN(K)).GT.TOLER) GO TO 40
      TIPF = TIP(K)
```

```
      RETURN
   40 IF(SF-SFIN(K)) 70,70,50
   50 K=K+1
      IF(K-NIN) 30,30,60
   60 TIPF = TIP(NIN)+(SF-SFIN(NIN))*SLOPE(NIN)
      RETURN
   70 SK = SFIN(K)-SFIN(K-1)
      TIPF    = EM(K-1)*(SFIN(K)-SF)**3/6./SK+EM(K)*(SF-SFIN(K-1))**3/
     1   6./SK+(TIP    (K)/SK-EM(K)*SK/6.)*(SF-SFIN(K-1))+(TIP(K-1)/
     2   SK-EM(K-1)*SK/6.)*(SFIN(K)-SF)
      RETURN
      ENTRY TIPNIT(NNN)
      CALL SPLINE(SFIN,TIP,NIN,SLOPE,EM)
      TOLER = ABS(SFIN(NIN)-SFIN(1))/FLOAT(NIN)*1.E-6
      RETURN
      END




      FUNCTION RHOIPF(SF)
C
C--RHOIPF CALCULATES UPSTREAM ABSOLUTE TOTAL DENSITY
C--AS A FUNCTION OF STREAM FUNCTION
C
      COMMON/INPUTT/GAM,AR,MSFL,OMEGA,REDFAC,VELTOL,FNEW,ONEW,MBI,MBO,
     1   MM,MHT,NBL,NHUB,NTIP,NIN,NCUT,NBLPL,NPPP,NOSTAT,NSL,LSFR,
     2   LTPL,LAMVT,IMESH,ISLINE,ISTATL,IPLOT,ISUPER,ITSON,IDEBUG,
     3   ZOMIN,ZOMBI,ZOMBO,ZOMOUT,ZHIN,ZTIN,ZHOUT,ZTOUT,ZHUB(50),
     4   RHUB(50),ZTIP(50),RTIP(50),SFIN(50),RADIN(50),TIP(50),PRIP(50),
     5   LAMIN(50),VTHIN(50),SFOUT(50),RADOUT(50),PROP(50),LOSOUT(50),
     6   LAMOUT(50),VTHOUT(50),ZHST(50),ZTST(50),FLFR(50),
     7   ZBL(50,50),RBL(50,50),THBL(50,50),TNBL(50,50)
      COMMON/CALCON/MMM1,MHTP1,CP,EXPON,TGROG,PITCH,CURVHI,CURVTI,
     1   CURVHO,CURVTO,RHIN,RTIN,RHOUT,RTOUT,RLEH,RLET,RTEH,RTET,
     2   ZLE(50),RLE(50),ZTE(50),RTE(50),ZLEOM(101),RLEOM(101),
     3   SLEOM(101),THLEOM(101),ZTEOM(101),RTEOM(101),STEOM(101),
     4   THTEOM(101),ILE(101),ITE(101),ZOM(100,101),ROM(100,101),
     5   SOM(100,101),TOM(100,101),BTH(100,101),DTHDS(100,101),
     6   DTHDT(100,101),PLOSS(100,101),CPHI(100,101),SPHI(100,101)
      DIMENSION SLOPE(50),EM(50),RHOIP(50)
      K = 2
      IF(ABS(SF-SFIN(1)).GT.TOLER) GO TO 10
      RHOIPF = RHOIP(1)
      RETURN
   10 IF(SF-SFIN(1)) 20,20,30
   20 RHOIPF = RHOIP(1)+(SF-SFIN(1))*SLOPE(1)
      RETURN
   30 IF(ABS(SF-SFIN(K)).GT.TOLER) GO TO 40
      RHOIPF = RHOIP(K)
      RETURN
   40 IF(SF-SFIN(K)) 70,70,50
   50 K=K+1
      IF(K-NIN) 30,30,60
   60 RHOIPF = RHOIP(NIN)+(SF-SFIN(NIN))*SLOPE(NIN)
      RETURN
   70 SK = SFIN(K)-SFIN(K-1)
      RHOIPF = EM(K-1)*(SFIN(K)-SF)**3/6./SK+EM(K)*(SF-SFIN(K-1))**3/
```

```
  1    6./SK+(RHOIP (K)/SK-EM(K)*SK/6.)*(SF-SFIN(K-1))+(RHOIP(K-1)/
  2    SK-EM(K-1)*SK/6.)*(SFIN(K)-SF)
      RETURN
      ENTRY RHINIT(NNN)
      DO 80 J=1,NIN
   80 RHOIP(J) = PRIP(J)/AR /TIP(J)
      CALL SPLINE(SFIN,RHOIP,NIN,SLOPE,EM)
      TOLER = ABS(SFIN(NIN)-SFIN(1))/FLOAT(NIN)*1.E-6
      RETURN
      END
```

```
      FUNCTION LAMDAF(SF,I,J)
C
C--LAMDAF CALCULATES PREWHIRL, LAMBDA, AS A FUNCTION OF STREAM
C--FUNCTION UPSTREAM OF THE BLADE
C
      COMMON SRW,SRE,ITER,IEND,NREAD,NWRIT
      COMMON/INPUTT/GAM,AR,MSFL,OMEGA,REDFAC,VELTOL,FNEW,DNEW,MBI,MBO,
     1   MM,MHT,NBL,NHUB,NTIP,NIN,NCUT,NBLPL,NPPP,NOSTAT,NSL,LSFR,
     2   LTPL,LAMVT,IMESH,ISLINE,ISTATL,IPLOT,ISUPER,ITSON,IDEBUG,
     3   ZOMIN,ZOMBI,ZOMBO,ZOMOUT,ZHIN,ZTIN,ZHOUT,ZTOUT,ZHUB(50),
     4   RHUB(50),ZTIP(50),RTIP(50),SFIN(50),RADIN(50),TIP(50),PRIP(50),
     5   LAMIN(50),VTHIN(50),SFOUT(50),RADOUT(50),PROP(50),LOSOUT(50),
     6   LAMOUT(50),VTHOUT(50),ZHST(50),ZTST(50),FLFR(50),
     7   ZBL(50,50),RBL(50,50),THBL(50,50),TNBL(50,50)
      COMMON/CALCON/MMM1,MHTP1,CP,EXPON,TGROG,PITCH,CURVHI,CURVTI,
     1   CURVHO,CURVTO,RHIN,RTIN,RHOUT,RTOUT,RLEH,RLET,RTEH,RTET,
     2   ZLE(50),RLE(50),ZTE(50),RTE(50),ZLEOM(101),RLEOM(101),
     3   SLEOM(101),THLEOM(101),ZTEOM(101),RTEOM(101),STEOM(101),
     4   THTEOM(101),ILE(101),ITE(101),ZOM(100,101),ROM(100,101),
     5   SOM(100,101),TOM(100,101),BTH(100,101),DTHDS(100,101),
     6   DTHDT(100,101),PLOSS(100,101),CPHI(100,101),SPHI(100,101)
      COMMON/VARCOM/A(4,100,101),UOM(100,101),K(100,101),RHO(100,101),
     1   WSUBS(100,101),WSUBT(100,101),WSUBZ(100,101),WSUBR(100,101),
     2   WSUBM(100,101),WTH(100,101),VTH(100,101),W(100,101),
     3   ALPHA(100,101),BETA(100,101),WWCR(100,101),CURV(100,101),
     4   WLSURF(100,101),WTSURF(100,101),CAMP(100,101),SAMP(100,101),
     5   RHOAV(100,101),DELRHO(100,101),FR(100,101),DFDM(100,101),
     6   XIOM(100,101),ZETOM(100,101),DLDU(100,101)
      DIMENSION SLOPE(50),EM(50),AAA(50),RILOM(101),UILOM(101)
      REAL LAMDAF,LAMIN,LAMOUT
      KK = 2
      IF(ABS(SF-SFIN(1)).GT.TOLER) GO TO 10
      LAMDAF = LAMIN(1)
      IF (I.LT.ILE(J)) DLDU(I,J)=SLOPE(1)
      RETURN
   10 IF(SF-SFIN(1)) 20,20,30
   20 LAMDAF = LAMIN(1)+(SF-SFIN(1))*SLOPE(1)
      IF (I.LT.ILE(J)) DLDU(I,J)=SLOPE(1)
      RETURN
   30 IF(ABS(SF-SFIN(KK)).GT.TOLER) GO TO 40
      LAMDAF = LAMIN(KK)
      IF (I.LT.ILE(J)) DLDU(I,J)=SLOPE(KK)
      RETURN
   40 IF(SF-SFIN(KK)) 70,70,50
   50 KK=KK+1
```

```
      IF(KK-NIN) 30,30,60
   60 LAMDAF = LAMIN(NIN)+(SF-SFIN(NIN))*SLOPE(NIN)
      IF (I.LT.ILE(J)) DLDU(I,J)=SLOPE(NIN)
      RETURN
   70 SK = SFIN(KK)-SFIN(KK-1)
      LAMDAF = EM(KK-1)*(SFIN(KK)-SF)**3/6./SK+EM(KK)*(SF-SFIN(KK-1))**3
     1   /6./SK+(LAMIN(KK)/SK-EM(KK)*SK/6.)*(SF-SFIN(KK-1))+(LAMIN(KK-1)
     2   /SK-EM(KK-1)*SK/6.)*(SFIN(KK)-SF)
      IF (I.LT.ILE(J)) DLDU(I,J)= -EM(KK-1)*(SFIN(KK)-SF)**2/2./SK+
     1   EM(KK)*(SFIN(KK-1)-SF)**2/2./SK+(LAMIN(KK)-LAMIN(KK-1))/SK-
     2   (EM(KK)-EM(KK-1))*SK/6.
      RETURN
      ENTRY LAMNIT(NNN)
      IF (ITER.EQ.0) GO TO 100
      II = MBI
      JJ = 1
      DO 80 KK=1,MHTP1
      DIST = FLOAT(KK-1)/FLOAT(MHT)
      RILOM(KK) = RHIN+DIST*(RTIN-RHIN)
      ZILOM = ZHIN+DIST*(ZTIN-ZHIN)
   80 CALL LININT(ZOM,ROM,UOM,MM,MHTP1,100,101,ZILOM,RILOM(KK),
     1UILOM(KK),II,JJ)
      IF (LSFR.EQ.0) CALL SPLINT(UILOM,RILOM,MHTP1,SFIN,NIN,RADIN,AAA)
      IF (LSFR.EQ.1) CALL SPLINT(RILOM,UILOM,MHTP1,RADIN,NIN,SFIN,AAA)
      IF (LSFR.EQ.1) GO TO 100
      DO 90 KK=1,NIN
   90 LAMIN(KK)= RADIN(KK)*VTHIN(KK)
  100 CALL SPLINE(SFIN,LAMIN,NIN,SLOPE,EM)
      TOLER = ABS(SFIN(NIN)-SFIN(1))/FLOAT(NIN)*1.E-6
      RETURN
      END
```

```
      FUNCTION RHOOPF(SF)
C
C--RHOOPF CALCULATES DOWNSTREAM ABSOLUTE TOTAL DENSITY
C--AS A FUNCTION OF STREAM FUNCTION
C
      COMMON/INPUTT/GAM,AR,MSFL,OMEGA,REDFAC,VELTOL,FNEW,DNEW,MBI,MBO,
     1   MM,MHT,NBL,NHUB,NTIP,NIN,NOUT,NBLPL,NPPP,NDSTAT,NSL,LSFR,
     2   LTPL,LAMVT,IMESH,ISLINE,ISTATL,IPLOT,ISUPER,ITSON,IDEBUG,
     3   ZOMIN,ZOMBI,ZOMBO,ZOMOUT,ZHIN,ZTIN,ZHOUT,ZTOUT,ZHUB(50),
     4   RHUB(50),ZTIP(50),RTIP(50),SFIN(50),RADIN(50),TIP(50),PRIP(50),
     5   LAMIN(50),VTHIN(50),SFOUT(50),RADOUT(50),PROP(50),LOSOUT(50),
     6   LAMOUT(50),VTHOUT(50),ZHST(50),ZTST(50),FLFR(50),
     7   ZBL(50,50),RBL(50,50),THBL(50,50),TNBL(50,50)
      COMMON/CALCON/MMM1,MHTP1,CP,EXPON,TGROG,PITCH,CURVHI,CURVTI,
     1   CURVHO,CURVTO,RHIN,RTIN,RHOUT,RTOUT,RLEH,RLET,RTEH,RTET,
     2   ZLE(50),RLE(50),ZTE(50),RTE(50),ZLEOM(101),RLEOM(101),
     3   SLEOM(101),THLEOM(101),ZTEOM(101),RTEOM(101),STEOM(101),
     4   THTEOM(101),ILE(101),ITE(101),ZOM(100,101),ROM(100,101),
     5   SOM(100,101),TOM(100,101),BTH(100,101),DTHDS(100,101),
     6   DTHDT(100,101),PLOSS(100,101),CPHI(100,101),SPHI(100,101)
      DIMENSION SLOPE(50),EM(50),RHOOP(50)
      K = 2
      IF(ABS(SF-SFOUT(1)).GT.TOLER) GO TO 10
```

```
      RHCOPF = RHOOP(1)
      RETURN
   10 IF(SF-SFOUT(1)) 20,20,30
   20 RHOOPF = RHOOP(1)+(SF-SFOUT(1))*SLOPE(1)
      RETURN
   30 IF(ABS(SF-SFOUT(K)).GT.TOLER) GO TO 40
      RHOOPF = RHOOP(K)
      RETURN
   40 IF(SF-SFOUT(K)) 70,70,50
   50 K=K+1
      IF(K-NOUT) 30,30,60
   60 RHOOPF = RHOOP(NOUT)+(SF-SFOUT(NOUT))*SLOPE(NOUT)
      RETURN
   70 SK = SFOUT(K)-SFOUT(K-1)
      RHOOPF = EM(K-1)*(SFOUT(K)-SF)**3/6./SK+EM(K)*(SF-SFOUT(K-1))**3
     1   /6./SK+(RHOOP(K)/SK-EM(K)*SK/6.)*(SF-SFOUT(K-1))+(RHOOP(K-1)/
     2   SK-EM(K-1)*SK/6.)*(SFOUT(K)-SF)
      RETURN
      ENTRY RHONIT(NNN)
      DO 80 J=1,NOUT
   80 RHOOP(J) = PROP(J)/AR/TOPF(SFOUT(J))
      CALL SPLINE(SFOUT,RHOOP,NOUT,SLOPE,EM)
      TOLER = ABS(SFOUT(NOUT)-SFOUT(1))/FLOAT(NOUT)*1.E-6
      RETURN
      END
```

```
      FUNCTION RVTHTA(SF,I,J)
C
C--RVTHTA CALCULATES  R * V-THETA  AS A FUNCTION OF STREAM FUNCTION
C--DOWNSTREAM OF THE BLADE
C
      COMMON SRW,SRE,ITER,IEND,NREAD,NWRIT
      COMMON/INPUTT/GAM,AR,MSFL,OMEGA,REDFAC,VELTOL,FNEW,DNEW,MBI,MBO,
     1   MM,MHT,NBL,NHUB,NTIP,NIN,NOUT,NBLPL,NPPP,NOSTAT,NSL,LSFR,
     2   LTPL,LAMVT,IMESH,ISLINE,ISTATL,IPLOT,ISUPER,ITSON,IDEBUG,
     3   ZOMIN,ZOMBI,ZOMBO,ZOMOUT,ZHIN,ZTIN,ZHOUT,ZTOUT,ZHUB(50),
     4   RHUB(50),ZTIP(50),RTIP(50),SFIN(50),RADIN(50),TIP(50),PRIP(50),
     5   LAMIN(50),VTHIN(50),SFOUT(50),RADOUT(50),PROP(50),LOSOUT(50),
     6   LAMOUT(50),VTHOUT(50),ZHST(50),ZTST(50),FLFR(50),
     7   ZBL(50,50),RBL(50,50),THBL(50,50),TNBL(50,50)
      COMMON/CALCON/MMM1,MHTP1,CP,EXPON,TGROG,PITCH,CURVHI,CURVTI,
     1   CURVHO,CURVTO,RHIN,RTIN,RHOUT,RTOUT,RLEH,RLET,RTEH,RTET,
     2   ZLE(50),RLE(50),ZTE(50),RTE(50),ZLEOM(101),RLEOM(101),
     3   SLEOM(101),THLEOM(101),ZTEOM(101),RTEOM(101),STEOM(101),
     4   THTEOM(101),ILE(101),ITE(101),ZOM(100,101),ROM(100,101),
     5   SOM(100,101),TOM(100,101),BTH(100,101),DTHCS(100,101),
     6   DTHDT(100,101),PLOSS(100,101),CPHI(100,101),SPHI(100,101)
      COMMON/VARCOM/A(4,100,101),UOM(100,101),K(100,101),RHO(100,101),
     1   WSUBS(100,101),WSUBT(100,101),WSUBZ(100,101),WSUBR(100,101),
     2   WSUBM(100,101),WTH(100,101),VTH(100,101),W(100,101),
     3   ALPHA(100,101),BETA(100,101),WWCR(100,101),CURV(100,101),
     4   WLSURF(100,101),WTSURF(100,101),CAMP(100,101),SAMP(100,101),
     5   RHOAV(100,101),DELRHO(100,101),FR(100,101),DFDM(100,101),
     6   XIOM(100,101),ZETOM(100,101),DLDU(100,101)
      DIMENSION SLOPE(50),EM(50),AAA(50),ROLOM(101),UOLOM(101)
      REAL LAMIN,LAMOUT
```

```
      KK = 2
      IF(ABS(SF-SFOUT(1)).GT.TOLER) GO TO 10
      RVTHTA = LAMOUT(1)
      IF (I.GT.ITE(J)) DLDU(I,J)=SLCFE(1)
      RETURN
   10 IF(SF-SFOUT(1)) 20,20,30
   20 RVTHTA = LAMOUT(1)+(SF-SFOUT(1))*SLOPE(1)
      IF (I.GT.ITE(J)) DLDU(I,J)=SLOPE(1)
      RETURN
   30 IF(ABS( SF-SFOUT(KK)).GT.TOLER) GO TO 40
      RVTHTA = LAMOUT(KK)
      IF (I.GT.ITE(J)) DLDU(I,J)=SLOPE(KK)
      RETURN
   40 IF(SF-SFOUT(KK)) 70,70,50
   50 KK=KK+1
      IF(KK-NOUT) 30,30,60
   60 RVTHTA = LAMOUT(NOUT)+( SF-SFOUT(NOUT))* SLOPE(NOUT)
      IF (I.GT.ITE(J)) DLDU(I,J)=SLCFE(NOUT)
      RETURN
   70 SK =  SFOUT(KK)-SFOUT(KK-1)
      RVTHTA = EM(KK-1)*(SFOUT(KK)-SF)**3/6./SK +EM(KK)*( SF-SFOUT(KK-1))
     1    **3/6./SK+(LAMOUT(KK)/SK-EM(KK)*SK/6.)*(SF-SFCUT(KK-1))+
     2    (LAMOUT(KK-1)/SK-EM(KK-1)*SK/6.)*( SFOUT(KK)-SF)
      IF (I.GT.ITE(J)) DLDU(I,J)= -EM(KK-1)*(SFOUT(KK)-SF)**2/2./SK+
     1    EM(KK)*( SFOUT(KK-1)-SF)**2/2./SK+(LAMOUT(KK)-LAMOUT(KK-1))
     2    /SK-(EM(KK)-EM(KK-1))*SK/6.
      RETURN
      ENTRY RVTNIT(NNN)
      IF(ITER.EQ.0) GO TO 100
      II = MBO
      JJ = 1
      DO 80 KK=1,MHTP1
      DIST = FLOAT(KK-1)/FLOAT(MHT)
      ROLOM(KK) = RHOUT+DIST*(RTOUT-RHOUT)
      ZOLOM = ZHOUT +DIST*(ZTOUT-ZHOUT)
   80 CALL LININT(ZOM,ROM,UOM,MM,MHTP1,100,101,ZOLOM,RCLOM(KK),UOLOM(KK)
     1,II,JJ)
      IF (LSFR.EQ.0) CALL SPLINT(UOLOM,ROLOM,MHTP1,SFOUT,NOUT,RADOUT,
     1AAA)
      IF (LSFR.EQ.1) CALL SPLINT(ROLCM,UOLOM,MHTP1,RADCUT,NOUT,SFOUT,
     1AAA)
      IF (LSFR.EQ.1) GC TC 100
      DO 90 KK=1,NOUT
   90 LAMOUT(KK)= RADOUT(KK)*VTHOUT(KK)
  100 CALL SPLINE(SFOUT,LAMOUT,NOUT,SLOPE,EM)
      TCLER = ABS(SFOUT(NOUT)-SFOUT(1))/FLOAT(NOUT)*1.E-6
      RETURN
      END


      SUBROUTINE CONTIN(XEST,YCALC,IND,JZ,YGIV,XDEL)
C
C--CONTIN CALCULATES AN ESTIMATE OF THE RELATIVE FLOW VELOCITY
C--FOR USE IN THE VELOCITY GRACIENT EQUATION
C
      DIMENSION X(3),Y(3)
```

```
      NCALL = NCALL+1
      IF (IND.NE.1.AND.NCALL.GT.100) GO TO 160
      GO TO (10,30,40,50,60,110,150),IND
C--FIRST CALL
   10 NCALL = 1
      XORIG = XEST
      IF (YCALC.GT.YGIV.AND.JZ.EQ.1) GO TO 20
      IND = 2
      Y(1) = YCALC
      X(1) = 0.
      XEST = XEST+XDEL
      RETURN
   20 IND = 3
      Y(3) = YCALC
      X(3) = 0.
      XEST = XEST-XDEL
      RETURN
C--SECOND CALL
   30 IND = 4
      Y(2) = YCALC
      X(2) = XEST-XORIG
      XEST = XEST+XDEL
      RETURN
   40 IND = 5
      Y(2) = YCALC
      X(2) = XEST-XORIG
      XEST = XEST-XDEL
      RETURN
C--THIRD OR LATER CALL - FIND SUBSONIC OR SUPERSONIC SOLUTION
   50 Y(3) = YCALC
      X(3) = XEST-XORIG
      GO TO 70
   60 Y(1) = YCALC
      X(1) = XEST-XORIG
   70 IF (YGIV.LT.AMIN1(Y(1),Y(2),Y(3))) GO TO (120,130),JZ
   80 IND = 6
      CALL PABC(X,Y,APA,BPB,CPC)
      DISCR = BPB**2-4.*APA*(CPC-YGIV)
      IF (DISCR.LT.0.) GO TO 140
      IF (ABS(400.*APA*(CPC-YGIV)).LE.BPB**2) GO TO 90
      XEST = -BPB-SIGN(SQRT(DISCR),APA)
      IF (JZ.EQ.1.AND.APA.GT.0..AND.Y(3).GT.Y(1)) XEST = -BPB+
     1SQRT(DISCR)
      IF (JZ.EQ.2.AND.APA.LT.0.) XEST = -BPB-SQRT(DISCR)
      XEST = XEST/2./APA
      GO TO 100
   90 IF (JZ.EQ.2.AND.BPB.GT.0.) GO TO 130
      ACB2 = APA/BPB*(CPC-YGIV)/BPB
      IF (ABS(ACB2).LE.1.E-8) ACB2=0.
      XEST = -(CPC-YGIV)/BPB*(1.+ACB2+2.*ACB2**2)
  100 IF (XEST.GT.X(3)) GO TO 130
      IF (XEST.LT.X(1)) GO TO 120
      XEST = XEST+XORIG
      RETURN
C--FOURTH OR LATER CALL - NOT CHOKED
  110 IF(XEST-XORIG.GT.X(3)) GO TO 130
      IF(XEST-XORIG.LT.X(1)) GO TO 120
      Y(2) = YCALC
      X(2) = XEST-XORIG
      GO TO 70
```

```
C--THIRD OR LATER CALL - SOLUTION EXISTS,
C--BUT RIGHT OR LEFT SHIFT REQUIRED
  120 IND = 5
C--LEFT SHIFT
      XEST = X(1)-XDEL+XORIG
      XOSHFT = XEST-XORIG
      XORIG = XEST
      Y(3)  = Y(2)
      X(3) = X(2)-XOSHFT
      Y(2) = Y(1)
      X(2) = X(1)-XOSHFT
      RETURN
  130 IND = 4
C--RIGHT SHIFT
      XEST = X(3)+XDEL+XORIG
      XOSHFT = XEST-XORIG
      XORIG = XEST
      Y(1) = Y(2)
      X(1) = X(2)-XOSHFT
      Y(2) = Y(3)
      X(2) = X(3)-XOSHFT
      RETURN
C--THIRD OR LATER CALL - APPEARS TO BE CHOKED
  140 XEST = -BPB/2./APA
      IND = 7
      IF (XEST.LT.X(1)) GO TO 120
      IF(XEST.GT.X(3)) GO TO 130
      XEST = XEST+XORIG
      RETURN
C--FOURTH OR LATER CALL - PROBABLY CHOKED
  150 IF (YCALC.GE.YGIV) GO TO 110
      IND = 10
      RETURN
C--NO SOLUTION FOUND IN 100 ITERATIONS
  160 IND = 11
      RETURN
      END
```

```
      SUBROUTINE PABC(X,Y,A,B,C)
C
C--PABC CALCULATES COEFFICIENTS A,B,C OF THE PARABOLA
C--Y=A*X**2+B*X+C, PASSING THROUGH THE GIVEN X,Y POINTS
C
      DIMENSION X(3),Y(3)
      C1 = X(3)-X(1)
      C2 = (Y(2)-Y(1))/(X(2)-X(1))
      A = (C1*C2-Y(3)+Y(1))/C1/(X(2)-X(3))
      B = C2-(X(1)+X(2))*A
      C = Y(1)-X(1)*B-X(1)**2*A
      RETURN
      END
```

```
      SUBROUTINE INRSCT(XCURV1,YCURV1,N1,XCURV2,YCURV2,N2,XCROSS,YCROSS)
C
C--INRSCT CALCULATES THE COORDINATES (XCROSS,YCROSS) OF THE POINT
C--OF INTERSECTION OF TWO SPLINE CURVES, YCURV1=F(XCURV1) AND
C--XCURV2=G(YCURV2), LYING ON A PLANE
C
      COMMON SRW,SRE,ITER,IENC,NREAD,NWRIT
      DIMENSION XCURV1(N1),YCURV1(N1),XCURV2(N2),YCURV2(N2)
      NCOUNT = 0
      TOLER = (ABS(XCURV1(N1)-XCURV1(1))+ABS(YCURV2(N2)-YCURV2(1)))/1.E5
      XTEMP = XCURV1(1)
      YTEMP = YCURV1(1)
      XCROSS = (XCURV1(1)+XCURV1(N1))/2.
C--COMPUTE INTERSECTION POINT AND SLOPE ON CURVE 1
   10 X1 = XCROSS
      CALL SPLINT(XCURV1,YCURV1,N1,X1,1,Y1,S1)
C--COMPUTE INTERSECTION POINT AND SLOPE ON CURVE 2
      Y2 = Y1
      CALL SPLINT(YCURV2,XCURV2,N2,Y2,1,X2,S2)
C--COMPUTE COORDINATES OF POINT WHERE TWO SLOPES INTERSECT
      S1S2 = S1*S2
      XCROSS = X2+S1S2*(X2-X1)/(1.-S1S2)
      YCROSS = Y1+S1  *(X2-X1)/(1.-S1S2)
C--COMPUTE DISTANCE AWAY FROM PREVIOUS SLOPE INTERSECTION POINT
      DIST = SQRT((YCROSS-YTEMP)**2+(XCROSS-XTEMP)**2)
      IF (DIST.LT.TOLER) RETURN
      NCOUNT = NCOUNT+1
      IF (NCOUNT.GT.20) GO TO 20
      XTEMP = XCROSS
      YTEMP = YCROSS
      GO TO 10
   20 WRITE(NWRIT,1000) TOLER,DIST
      RETURN
 1000 FORMAT (6X,46HINRSCT HAS FAILED TO CONVERGE IN 20 ITERATIONS/
     110X,11HTOLERANCE =,G14.6/10X,47HDISTANCE BETWEEN LAST TWO INTERSEC
     2TION POINTS =,G14.6)
      END
```

```
      SUBROUTINE ROOT(A,B,Y,FUNCT,TOLERY,X,DFX)
C
C--ROOT FINDS A ROOT FOR (FUNCT MINUS Y) IN THE INTERVAL (A,B)
C
      COMMON SRW,SRE,ITER,IEND,NREAD,NWRIT
      INTEGER SRW,SRE
      ISRW= 0
   10 IF (SRW.EQ.21) WRITE(NWRIT,1010) A,B,Y,TOLERY
      X1 = A
      CALL FUNCT(X1,FX1,DFX)
      IF(SRW.EQ.21) WRITE(NWRIT,1020) X1,FX1,DFX
      X2 = B
   20 DO 40 I=1,20
      X = (X1+X2)/2.
      CALL FUNCT(X,FX,DFX)
      IF(SRW.EQ.21) WRITE(NWRIT,1020) X,FX,DFX
      IF((FX1-Y)*(FX-Y).GT.0.) GO TO 30
```

```
      X2 = X
      GO TO 40
   30 X1 = X
      FX1 = FX
   40 CONTINUE
      IF(ABS(Y-FX).LT.TOLERY) RETURN
      IF (ISRW.EQ.1) GO TO 50
      WRITE(NWRIT,1000)
      ISRW= 1
      JSRW= SRW
      SRW= 21
      GO TO 10
   50 SRW= JSRW
      RETURN
 1000 FORMAT(72H1ROOT HAS FAILED TO LOCATE A ROOT IN THE INTERVAL (A,B)
     1IN 20 ITERATIONS)
 1010 FORMAT(22H ROOT ARGUMENTS -- A =,G13.5,3X,3HB =,G13.5,3X,3HY =,
     1G13.5,3X,8HTOLERY =,G13.5/16X,1HX,17X,2HFX,15X,3HDFX)
 1020 FORMAT(8X,G16.5,2G18.5)
      END
```

```
      SUBROUTINE LININT(X,Y,Z,NX,NY,NDIMX,NDIMY,XO,YO,ZO,I,J)
C
C--LININT LOCATES THE POINT (XO,YO) IN A 2-D  MESH WITH
C--COORDINATES STORED IN THE X AND Y ARRAYS.  THEN THE VALUE OF ZO AT
C--(XO,YO) IS INTERPOLATED FROM THE Z ARRAY VALUES CORRESPONDING
C--TO THE X AND Y ARRAYS
C
      DIMENSION X(NDIMX,NDIMY),Y(NDIMX,NDIMY),Z(NDIMX,NDIMY)
      DIMENSION EXTRAP(2)
      INTEGER ABOVE,RIGHT
C--FIND I,J SUCH THAT (XO,YO) IS IN COLUMN I FROM THE LEFT AND IN ROW J
C--FROM THE BOTTOM
      IF(NX.LT.2.OR.NY.LT.2) STOP
      IF(I.LE.0) I = 1
      IF(I.GE.NX) I = NX-1
      IF(J.LE.0) J = 1
      IF(J.GE.NY) J = NY-1
   10 ABOVE = -1
      RIGHT = -1
      IF(YO.GE.Y(I,J)+(XO-X(I,J))/(X(I+1,J)-X(I,J))*(Y(I+1,J)-Y(I,J)))
     1   ABOVE = ABOVE+1
      IF(YO.GT.Y(I,J+1)+(XO-X(I,J+1))/(X(I+1,J+1)-X(I,J+1))*
     1   (Y(I+1,J+1)-Y(I,J+1)))  ABOVE = ABOVE+1
      IF(XO.GE.X(I,J)+(YO-Y(I,J))/(Y(I,J+1)-Y(I,J))*(X(I,J+1)-X(I,J)))
     1   RIGHT = RIGHT+1
      IF(XO.GT.X(I+1,J)+(YO-Y(I+1,J))/(Y(I+1,J+1)-Y(I+1,J))*
     1   (X(I+1,J+1)-X(I+1,J)))  RIGHT = RIGHT+1
      IN = I+RIGHT
      JN = J+ABOVE
      IF(IN.LT.1.OR.IN.GE.NX) RIGHT = 0
      IF(JN.LT.1.OR.JN.GE.NY) ABOVE = 0
      IF(ABOVE**2+RIGHT**2.EQ.0) GO TO 20
      I = I+RIGHT
      J = J+ABOVE
```

```
      GO TO 10
   20 IJEX = 1
C-- SET EXTRAP TO INDICATE EXTRAPOLATION
      EXTRAP(1) = 0.
      EXTRAP(2) = 0.
      IF(IN.LT.1) EXTRAP(2) = -1.
      IF(IN.GE.NX) EXTRAP(2) = 1.
      IF(JN.LT.1) EXTRAP(1) = -1.
      IF(JN.GE.NY) EXTRAP(1) = 1.
C--CALCULATE CONSTANTS TO CALCULATE FY
      Y13 = Y(I,J)-Y(I,J+1)
      X13 = X(I,J)-X(I,J+1)
      Y42 = Y(I+1,J+1)-Y(I+1,J)
      X42 = X(I+1,J+1)-X(I+1,J)
      Y01 = Y0-Y(I,J)
      X01 = X0-X(I,J)
      Y02 = Y0-Y(I+1,J)
      X02 = X0-X(I+1,J)
      Y21 = Y(I+1,J)-Y(I,J)
      X21 = X(I+1,J)-X(I,J)
C--CALCULATE COEFFICIENTS OF QUADRATIC EQUATION FOR FRACTIONAL DISTANCE
C--IN QUADRILATERAL
   30 QA = Y13*X42-X13*Y42
      QB = X13*Y02-Y13*X02+Y01*X42-X01*Y42
      QC = Y01*X21-X01*Y21
      DISCR = QB**2-4.*QA*QC
      IF(DISCR.LT.0.) GO TO 110
C--CHECK TO SEE IF QUADRATIC EQUATION IS CLOSE TO LINEAR
      IF(ABS(4.*QA*QC).LE.QB**2*.01) GO TO 80
      FA = -QB/2./QA
      FB = SQRT(DISCR)/2./QA
      F1 = FA+FB
      F2 = FA-FB
C--CHECK TO DETERMINE WHETHER F1 OR F2 IS THE PROPER SOLUTION
      CASE = -1.
      IF(EXTRAP(IJEX)) 40,50,60
C--EXTRAPOLATION BELOW OR TO LEFT (FF LESS THAN 0.)
   40 IF(F1.LT..01) CASE = CASE+1.
      IF(F2.LT..01) CASE = CASE+2.
      IF(CASE.LT.1.5) GO TO 70
      CASE = CASE-1.
      IF(F2.LT.F1) CASE = CASE-1.
      GO TO 70
C--NO EXTRAPOLATION
   50 IF(ABS(F1-.5).LT..51) CASE = CASE+1.
      IF(ABS(F2-.5).LT..51) CASE = CASE+2.
      GO TO 70
C--EXTRAPOLATION ABOVE OR TO RIGHT (FF GREATER THAN 1.)
   60 IF(F1.GT..99) CASE = CASE+1.
      IF(F2.GT..99) CASE = CASE+2.
      IF(CASE.LT.1.5) GO TO 70
      CASE = CASE-1.
      IF(F1.LT.F2) CASE = CASE-1.
   70 IF(ABS(CASE-.5).GT..6) GO TO 110
      FF = (1.-CASE)*F1+CASE*F2
      GO TO 90
C--IF QUADRATIC EQUATION IS NEAR LINEAR, USE BINOMIAL EXPANSION FOR FF
   80 ACB2 = QA/QB*QC/QB
      IF(ABS(ACB2).LT.1.E-8) ACB2 = 0.
```

```
      FF = -QC/QB*(1.+ACB2+2.*ACB2**2)
   90 IF(IJEX.EQ.2) GO TO 100
      IJEX = IJEX+1
      FY = FF
C--INTERCHANGE CORNER POINTS TO GET FX
      Y13 = Y(I,J)-Y(I+1,J)
      X13 = X(I,J)-X(I+1,J)
      Y42 = Y(I+1,J+1)-Y(I,J+1)
      X42 = X(I+1,J+1)-X(I,J+1)
      Y02 = Y0-Y(I,J+1)
      X02 = X0-X(I,J+1)
      Y21 = Y(I,J+1)-Y(I,J)
      X21 = X(I,J+1)-X(I,J)
      GO TO 30
C--CALCULATE INTERPOLATED VALUE
  100 FX = FF
      Z0 = Z(I,J)*(1.-FX)*(1.-FY)+Z(I+1,J)*FX*(1.-FY)+Z(I,J+1)*(1.-FX)
     1   *FY+Z(I+1,J+1)*FX*FY
      RETURN
C-- PRINT ERROR MESSAGE IF THERE IS A PROBLEM IN OBTAINING A SOLUTION
  110 Z0 = 0.
      WRITE(6,1000) I,J
      RETURN
 1000 FORMAT(38H1LININT CANNOT FIND INTERPOLATED VALUE/4H I =,I6,4H J =,
     1I6)
      END
```

```
      SUBROUTINE SPLINE (X,Y,N,SLOPE,EM)
C
C--SPLINE CALCULATES FIRST AND SECOND DERIVATIVES AT SPLINE POINTS
C--END CONDITION - SECOND DERIVATIVES AT EITHER END POINT IS
C--ONE HALF THAT AT THE ADJACENT POINT
C
      COMMON SRW,SRE,ITER,IEND,NREAD,NWRIT
      DIMENSION X(N),Y(N),EM(N),SLOPE(N)
      DIMENSION G(101),SB(101)
      INTEGER SRW,SRE
      SB(1) = -0.5
      G(1) = 0.
      NO=N-1
      IF (NO.LT.2) SB(1)=0.
      IF(NO.LT.2) GO TO 20
      DO 10 I=2,NO
      A = (X(I)-X(I-1))/6.
      C = (X(I+1)-X(I))/6.
      W = 2.*(A+C)-A*SB(I-1)
      SB(I) = C/W
      F = (Y(I+1)-Y(I))/(X(I+1)-X(I))-(Y(I)-Y(I-1))/(X(I)-X(I-1))
   10 G(I) = (F-A*G(I-1))/W
   20 EM(N) = G(N-1)/(2.+SB(N-1))
      DO 30 I=2,N
      K = N+1-I
   30 EM(K) = G(K)-SB(K)*EM(K+1)
      SLOPE(1) = (X(1)-X(2))/6.*(2.*EM(1)+EM(2))+(Y(2)-Y(1))/(X(2)-X(1))
      DO 40 I=2,N
   40 SLOPE(I) = (X(I)-X(I-1))/6.*(2.*EM(I)+EM(I-1))+(Y(I)-Y(I-1))/
```

```
     1   (X(I)-X(I-1))
      IF(SRW.EQ.13) WRITE(NWRIT,1000) N,(X(I),Y(I),SLOPE(I),EM(I),I=1,N)
      RETURN
 1000 FORMAT (2X,15HNO. OF POINTS =,I3/10X,1HX,19X,1HY,19X,5HSLOPE,15X,
     12HEM/(4G20.8))
      END




      SUBROUTINE SPLINT (X,Y,N,Z,MAX,YINT,DYDX)
C
C--SPLINT CALCULATES INTERPOLATED POINTS AND DERIVATIVES
C--FOR A SPLINE CURVE
C--END CONDITION - SECOND DERIVATIVE AT EITHER END POINT IS ONE-HALF
C--THAT AT THE ADJACENT POINT
C
      COMMON SRW,SRE,ITER,IEND,NREAD,NWRIT
      DIMENSION X(N),Y(N),Z(MAX),YINT(MAX),DYDX(MAX)
      DIMENSION G(101),SB(101),EM(101)
      INTEGER SRW,SRE
      IF(MAX.LE.0) RETURN
      TOLER= ABS(X(N)-X(1))/FLOAT(N)*1.E-5
      SB(1) = -.5
      G(1) = 0.
      NO=N-1
      IF(NO.LT.2) GO TO 20
      DO 10 I=2,NO
      A = (X(I)-X(I-1))/6.
      C = (X(I+1)-X(I))/6.
      W = 2.*(A+C)-A*SB(I-1)
      SB(I) = C/W
      F = (Y(I+1)-Y(I))/(X(I+1)-X(I))-(Y(I)-Y(I-1))/(X(I)-X(I-1))
   10 G(I) = (F-A*G(I-1))/W
   20 EM(N) = G(N-1)/(2.+SB(N-1))
      DO 30 I=2,N
      K = N+1-I
   30 EM(K) = G(K)-SB(K)*EM(K+1)
C
      ENTRY SPLENT (X,Y,N,Z,MAX,YINT,DYDX)
      DO 120 I=1,MAX
      K=2
      IF (ABS(Z(I)-X(1)).LT.TOLER) GO TO 40
      IF (Z(I).GT.X(1)) GO TO 50
      GO TO 80
   40 YINT(I)=Y(1)
      SK = X(K)-X(K-1)
      GO TO 110
   50 IF (ABS(Z(I)-X(K)).LT.TOLER) GO TO 60
      IF (Z(I).GT.X(K)) GO TO 70
      GO TO 100
   60 YINT(I)=Y(K)
      SK = X(K)-X(K-1)
      GO TO 110
   70 K=K+1
      IF(K-N) 50,50,90
   80 DYDX(I) = (X(1)-X(2))/6.*(2.*EM(1)+EM(2))+(Y(2)-Y(1))/(X(2)-X(1))
      YINT(I) = Y(1)+DYDX(I)*(Z(I)-X(1))
```

```
      GC TC 120
   90 DYDX(I) = (X(N)-X(N-1))/6.*(EM(N-1)+2.*EM(N))+(Y(N)-Y(N-1))/(X(N)
     1-X(N-1))
      YINT(I) = Y(N)+DYDX(I)*(Z(I)-X(N))
      GO TO 120
  100 SK = X(K)-X(K-1)
      YINT(I) = EM(K-1)*(X(K)-Z(I))**3/6./SK  +EM(K)*(Z(I)-X(K-1))**3/6.
     1  /SK+(Y(K)/SK  -EM(K)*SK  /6.)*(Z(I)-X(K-1))+(Y(K-1)/SK  -EM(K-1)
     2  *SK/6.)*(X(K)-Z(I))
  110 DYDX(I)=-EM(K-1)*(X(K)-Z(I))**2/2.0/SK  +EM(K)*(X(K-1)-Z(I))**2/2.
     1   /SK+(Y(K)-Y(K-1))/SK  -(EM(K)-EM(K-1))*SK/6.
  120 CONTINUE
      MXA = MAX0(N,MAX)
      IF(SRW.EQ.16) WRITE(NWRIT,1000) N,MAX,(X(I),Y(I),Z(I),YINT(I),
     1DYDX(I),I=1,MXA)
      RETURN
 1000 FORMAT (2X,21HNO. OF POINTS GIVEN =,I3,30H, NO. OF INTERPOLATED PO
     1INTS =,I3/10X,1HX,19X,1HY,16X,11HX-INTERPOL.,9X,11HY-INTERPOL.,
     28X,14HDYDX-INTERPOL./(5E20.8))
      END
```

```
      SUBROUTINE SLOPES(X,Y,N,SLOPE)
C
C--SLOPES CALCULATES FIRST DERIVATIVES, SLOPE, OF THE FUNCTION, Y,
C--WITH RESPECT TO X, USING A PARABOLIC FIT THROUGH EACH SET OF
C--THREE ADJACENT POINTS ON THE CURVE
C
      DIMENSION X(N),Y(N),SLOPE(N)
      N1 = N-1
      N2 = N-2
      IF (N1.LT.2) GO TO 20
C--MID POINTS
      DO 10 I=2,N1
      X3X2 = X(I+1)-X(I)
      X2X1 = X(I)-X(I-1)
      X3X1 = X(I+1)-X(I-1)
      Y3Y2 = Y(I+1)-Y(I)
      Y2Y1 = Y(I)-Y(I-1)
   10 SLOPE(I) = (X2X1**2*Y3Y2+X3X2**2*Y2Y1)/(X3X2*X2X1*X3X1)
C--FIRST POINT
      X3X2 = X(3)-X(2)
      X2X1 = X(2)-X(1)
      X3X1 = X(3)-X(1)
      Y3Y1 = Y(3)-Y(1)
      Y2Y1 = Y(2)-Y(1)
      SLOPE(1) = (X3X1**2*Y2Y1-X2X1**2*Y3Y1)/(X3X2*X2X1*X3X1)
C--LAST POINT
      X3X2 = X(N)-X(N1)
      X2X1 = X(N1)-X(N2)
      X3X1 = X(N)-X(N2)
      Y3Y2 = Y(N)-Y(N1)
      Y3Y1 = Y(N)-Y(N2)
      SLOPE(N) = (X3X1**2*Y3Y2-X3X2**2*Y3Y1)/(X3X2*X2X1*X3X1)
      RETURN
```

```
C--TWO POINT FUNCTION
   20 SLOPE(1) = (Y(2)-Y(1))/(X(2)-X(1))
      SLOPE(2) = SLOPE(1)
      RETURN
      END
```

Lewis Research Center,
National Aeronautics and Space Administration,
Cleveland, Ohio, August 20, 1973,
501-24.

APPENDIX A

FINITE-DIFFERENCE FORM OF STREAM-FUNCTION EQUATION

The stream-function equation was derived as equation (B18) of part I (ref. 6):

$$\frac{\partial^2 u}{\partial s^2} + \frac{\partial^2 u}{\partial t^2} - \frac{\partial u}{\partial s}\left[\frac{\sin\varphi}{r} + \frac{1}{B}\frac{\partial B}{\partial s} + \frac{1}{\rho}\frac{\partial \rho}{\partial s} - \frac{1}{\cos\varphi}\frac{\partial(\sin\varphi)}{\partial t}\right] - \frac{\partial u}{\partial t}\left[\frac{\cos\varphi}{r} + \frac{1}{B}\frac{\partial B}{\partial t} + \frac{1}{\rho}\frac{\partial \rho}{\partial t} + \frac{1}{\cos\varphi}\right.$$

$$\left.\times\frac{\partial(\sin\varphi)}{\partial s}\right] + \frac{rB\rho}{wW_z}\left\{\frac{W_\theta}{r}\left[\sin\varphi\frac{\partial(rV_\theta)}{\partial s} + \cos\varphi\frac{\partial(rV_\theta)}{\partial t}\right] + \xi W^2 + \zeta + F_r\right\} = 0 \qquad \text{(A1)}$$

where

$$\xi = \frac{1}{2c_p}\left(\frac{R}{p''}\frac{\partial p''}{\partial r} - \frac{1}{T''}\frac{\partial I}{\partial r} - \frac{\omega^2 r}{T''}\right) \qquad \text{(A2)}$$

$$\zeta = \omega^2 r - \frac{RT''}{p''}\frac{\partial p''}{\partial r} \qquad \text{(A3)}$$

$$F_r = -\frac{\partial\theta}{\partial r} W \frac{\partial W}{\partial\theta} \qquad \text{(A4)}$$

Equation (A4) was derived as equation (B23) of part I.

The s- and t-coordinates are the coordinates along the orthogonal mesh generated by the program. At each point of this mesh where the value of the stream function is unknown, a finite-difference approximation to equation (A1) can be written. Adjacent to the boundary the boundary conditions are included. If there are n unknown values, n nonlinear equations are obtained in n unknowns. The equations are nonlinear since the coefficients involve the density, which depends on the solution; and since the final term depends on the solution in a nonlinear manner. The equations may be solved by an iterative procedure, with two levels of iteration. The inner iteration solves a linearized equation, and the outer iteration makes corrections to the linearized equation so that the solution converges to the solution of the original nonlinear equation.

A typical mesh point with the numbering used to indicate neighboring mesh points is shown in figure 24. The value of the stream function or the other variables at 0 is de-

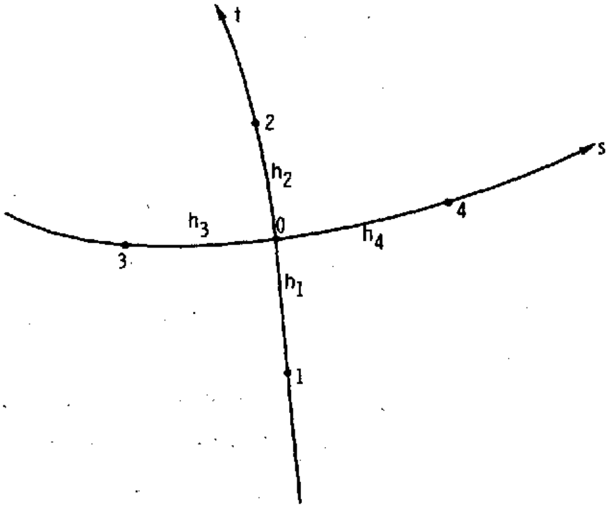


Figure 24. - Notation for adjacent mesh points and mesh spaces.

noted by using the subscript 0, and similarly for the neighboring points. It can be shown that equation (A1) can be approximated by

$$
\left[\frac{2u_1}{h_1(h_1+h_2)}+\frac{2u_2}{h_2(h_1+h_2)}-\frac{2u_0}{h_1h_2}\right]+\left[\frac{2u_3}{h_3(h_3+h_4)}+\frac{2u_4}{h_4(h_3+h_4)}-\frac{2u_0}{h_3h_4}\right]
$$

$$
-\frac{u_4-u_3}{h_3+h_4}\left[\frac{\sin\varphi_0}{r_0}+\frac{1}{B_0}\left(\frac{B_4-B_3}{h_3+h_4}\right)+\frac{1}{\rho_0}\left(\frac{\rho_4-\rho_3}{h_3+h_4}\right)-\frac{1}{\cos\varphi_0}\left(\frac{\sin\varphi_2-\sin\varphi_1}{h_1+h_2}\right)\right]
$$

$$
-\frac{u_2-u_1}{h_1+h_2}\left[\frac{\cos\varphi_0}{r_0}+\frac{1}{B_0}\left(\frac{B_2-B_1}{h_1+h_2}\right)+\frac{1}{\rho_0}\left(\frac{\rho_2-\rho_1}{h_1+h_2}\right)+\frac{1}{\cos\varphi_0}\left(\frac{\sin\varphi_4-\sin\varphi_3}{h_3+h_4}\right)\right]
$$

$$
+\frac{r_0B_0\rho_0}{w\left(W_z\right)_0}\left\{\frac{\left(W_\theta\right)_0}{r_0}\left[\sin\varphi\,\frac{\partial(rV_\theta)}{\partial s}+\cos\varphi\,\frac{\partial(rV_\theta)}{\partial t}\right]_0+\xi_0W_0^2+\zeta_0+\left(F_r\right)_0\right\}=0 \qquad \text{(A5)}
$$

where the partials of rV_θ are calculated by different methods upstream, downstream, and within the blade. Upstream and downstream of the blade, equations (B19) and (B20) of part I are used. Within the blade, a finite-difference approximation is used with values of V_θ from the previous iteration. The final result to be used in equation (A5) is

$$\left[\sin\varphi\frac{\partial(rV_\theta)}{\partial s}+\cos\varphi\frac{\partial(rV_\theta)}{\partial t}\right]_0=\begin{cases}\dfrac{r_0B_0\rho_0(W_z)_0}{w}\left(\dfrac{d\lambda}{du}\right)_0 & \text{upstream}\\[2ex] \sin\varphi_0\dfrac{\left[r_4(V_\theta)_4-r_3(V_\theta)_3\right]}{h_3+h_4}+\cos\varphi_0\dfrac{\left[r_2(V_\theta)_2-r_1(V_\theta)_1\right]}{h_1+h_2} & \text{within blade}\\[2ex] \dfrac{r_0B_0\rho_0(W_z)_0}{w}\left[\dfrac{d(rV_\theta)}{du}\right]_0 & \text{downstream}\end{cases} \tag{A6}$$

In setting up the equations for solution, the coefficients of the u_i in equation (A5) must be calculated. This was done by expressing equation (A5) as

$$u_0=\sum_{i=1}^{4}a_iu_i+k_0 \tag{A7}$$

where

$$\left.\begin{aligned}
a_0 &= \frac{2}{h_1h_2} + \frac{2}{h_3h_4} \\
c_1 &= h_1 + h_2 \\
c_2 &= h_3 + h_4 \\
d_1 &= \frac{\dfrac{B_2 - B_1}{B_0} + \dfrac{\rho_2 - \rho_1}{\rho_0}}{c_1} + \frac{\cos \varphi_0}{r_0} + \frac{\sin \varphi_4 - \sin \varphi_3}{c_2 \cos \varphi_0} \\
d_2 &= \frac{\dfrac{B_4 - B_3}{B_0} - \dfrac{\rho_4 - \rho_3}{\rho_0}}{c_2} + \frac{\sin \varphi_0}{r_0} - \frac{\sin \varphi_2 - \sin \varphi_1}{c_1 \cos \varphi_0} \\
a_1 &= \frac{\left(\dfrac{2}{h_1} + d_1\right)}{a_0c_1} \\
a_2 &= \frac{\left(\dfrac{2}{h_2} - d_1\right)}{a_0c_1} \\
a_3 &= \frac{\left(\dfrac{2}{h_3} + d_2\right)}{a_0c_2} \\
a_4 &= \frac{\left(\dfrac{2}{h_4} - d_2\right)}{a_0c_2}
\end{aligned}\right\} \qquad \text{(A8)}$$

$$
k_0 = \begin{cases} \dfrac{r_0 B_0 \rho_0}{a_0 w (W_z)_0} \left[\dfrac{(W_\theta)_0 B_0 \rho_0 (W_z)_0}{w} \left(\dfrac{d\lambda}{du}\right)_0 + \xi_0 W_0^2 + \zeta_0 \right] & \text{upstream} \\[2ex] \dfrac{r_0 B_0 \rho_0}{a_0 w (W_z)_0} \left[\dfrac{(W_\theta)_0}{r_0} \left\{ \left[r_4 (V_\theta)_4 - r_3 (V_\theta)_3 \right] \dfrac{\sin \varphi_0}{c_2} + \left[r_2 (V_\theta)_2 - r_1 (V_\theta)_1 \right] \dfrac{\cos \varphi_0}{c_1} \right\} + \xi_0 W_0^2 + \zeta_0 + (F_r)_0 \right] & \text{within blade} \\[2ex] \dfrac{r_0 B_0 \rho_0}{a_0 w (W_z)_0} \left\{ \dfrac{(W_\theta)_0 B_0 \rho_0 (W_z)_0}{w} \left[\dfrac{d(rV_\theta)}{du} \right]_0 + \xi_0 W_0^2 + \zeta_0 \right\} & \text{downstream} \end{cases} \qquad \text{(A9)}
$$

Equation (A8) is written in the form corresponding to the calculation of the coefficients in subroutine COEF. The constant k_0 is calculated from equation (A9) in subroutine COEF. The quantities ξ and ζ are calculated in subroutine NEWRHO from equations (A2) and (A3). The quantity F_r is calculated in subroutine BLDVEL when the blade surface velocities are calculated. The quantities $d\lambda/du$ and $d(rV_\theta)/du$ are calculated by subroutines LAMDAF and RVTHTA when they are called by NEWRHO to calculate λ or $\left(rV_\theta\right)_0$.

APPENDIX B

MATCHING UPSTREAM AND DOWNSTREAM FLOW CONDITIONS TO STREAM-FUNCTION SOLUTION

The work done by each blade row is determined by the change in whirl along streamlines. That is,

$$H_0 - H_i = \omega\left[\left(rV_\theta\right)_0 - \lambda\right] \tag{B1}$$

In this program, whirl can vary as desired from hub to tip, but for each streamline the work done is determined by equation (B1). Also, the equation relating velocity W to temperature and density requires knowledge of upstream total temperature and whirl for that particular streamline. For this reason, it is most desirable to express upstream and downstream conditions as a function of stream function rather than radius. However, if experimental data are being used, measurements are obtained as a function of position or radius. In this case the stream function is not known, but the distribution by radius can be used for input to the program. Then by estimation and iteration the correct distribution by stream function will be obtained.

If whirl is given as a function of stream function as input (i.e., LSFR = LAMVT = 0), no changes need be made after the first initialization. If tangential velocity V_θ is given as input (LAMVT = 1), certain subroutines must be reinitialized in every iteration. There are two possibilities: one that V_θ is given as a function of stream function (LSFR = 0), and the second that V_θ is given as a function of radius (LSFR = 1). In either case, what is needed is the relation between stream function and radius along the input lines. This relation is determined by the stream-function solution obtained by SOR. In each iteration, then, reinitialization calls are made by NEWRHO, if LAMVT = 1. If LSFR = 0, SFIN and SFOUT are given as input, and RADIN and RADOUT are corrected by the initilization calls to LAMNIT and RVTNIT. If LSFR = 1, RADIN and RADOUT are given as input, and SFIN and SFOUT are corrected by the same calls. In either case, SPLINT calls are made to readjust the spline fit coefficients for all five subroutines, LAMDAF, RVTHTA, TIPF, RHOIPF, and RHOOPF.

APPENDIX C

CALCULATION OF PARTIAL DERIVATIVES OF THETA ON ORTHOGONAL MESH

In the THETOM subroutine, $\partial\theta/\partial s$ and $\partial\theta/\partial t$ are calculated at the orthogonal mesh points which lie between the leading and trailing edges of the blade. This process is executed in a series of intermediate steps because input points on the different blade planes are not required to fall on a smooth curve running from hub to shroud. Also, the angle φ is known only at mesh points, so that $\partial\theta/\partial s$ and $\partial\theta/\partial t$ cannot be obtained directly. Therefore, $\partial\theta/\partial z$ and $\partial\theta/\partial r$ are obtained as an intermediate step. It is more accurate to calculate partial derivatives first and then interpolate and transform the partials than it would be to interpolate θ itself and then calculate the partials along mesh lines.

The orthogonal mesh on a typical blade is illustrated in figure 25. Notice that

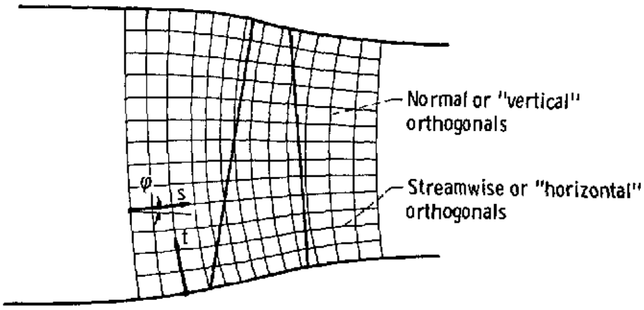


Figure 25. - Orthogonal finite-difference mesh on solution region.

some of the t mesh lines cross the leading and trailing edges of the blade. To alleviate the problem of finding θ-gradients on this mesh, they are first obtained on an alternative mesh, shown in figure 26, of s'- and t'-coordinates which lie entirely within the blade. Then by interpolation they are obtained at the desired orthogonal mesh points. (Recall that z, r, and θ are given as input on a number of blade sections from hub to shroud.)

The step-by-step procedure to obtain $\partial\theta/\partial s$ and $\partial\theta/\partial t$ is as follows:

(1) Calculate z- and r-coordinates (ZPC and RPC) of the mesh points on the s'-t' mesh. The s' mesh lines lie along the input blade sections (ZBL,RBL points). The t' mesh lines run from hub to shroud at 10-percent meridional chord locations (fig. 26). Once z-coordinates are calculated along the input blade sections, SPLINT calls are

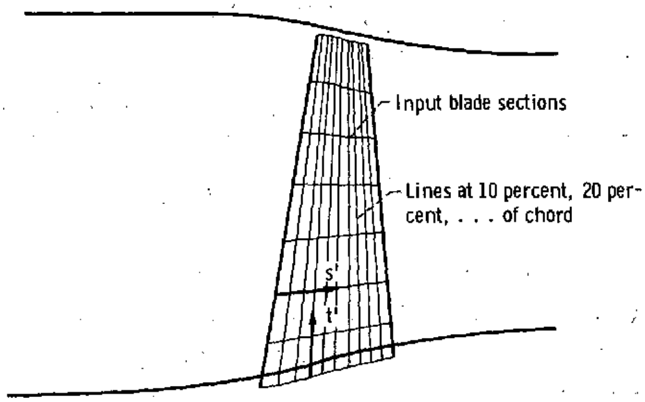


Figure 26. - Alternate mesh on which gradients of θ are obtained.

made in the s'-direction to obtain the corresponding r-coordinates and angles of the s'-coordinate line with respect to the z-axis $\alpha_{s'}$. (See fig. 27.)

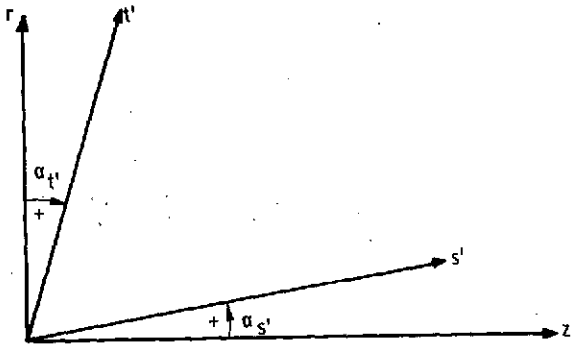


Figure 27. - Relation of s'-t' mesh to z- and r-directions.

(2) Calculate arc length SZRBL along the input blade section (s'-coordinate line) by using the input ZBL,RBL coordinates.

(3) Calculate arc length SZRPC along the s'-coordinate line by using the calculated ZPC,RPC coordinates.

(4) By using SPLINT calls in the s'-direction, calculate θ and $\partial\theta/\partial s'$ at the ZPC,RPC points.

(5) By using SPLINE calls along the t'-coordinate line, calculate angles between the t' lines and the radial direction $\alpha_{t'}$. (See fig. 27 for sign of $\alpha_{t'}$.)

(6) Calculate arc length SZRPC along the t'-coordinate line by using the ZPC,RPC coordinates.

(7) By using SPLINE calls in the t'-direction, calculate $\partial\theta/\partial t'$ at the ZPC, RPC points.

(8) Calculate $\partial\theta/\partial z$ and $\partial\theta/\partial r$ from $\partial\theta/\partial s'$ and $\partial\theta/\partial t'$ at the s'- and t'-coordinate points, with the following equations:

$$\frac{\partial\theta}{\partial z} = \frac{\partial\theta}{\partial s'} \frac{\cos \alpha_{t'}}{\cos(\alpha_{s'} + \alpha_{t'})} - \frac{\partial\theta}{\partial t'} \frac{\sin \alpha_{s'}}{\cos(\alpha_{s'} + \alpha_{t'})} \tag{C1}$$

$$\frac{\partial\theta}{\partial r} = -\frac{\partial\theta}{\partial s'} \frac{\sin \alpha_{t'}}{\cos(\alpha_{s'} + \alpha_{t'})} + \frac{\partial\theta}{\partial t'} \frac{\cos \alpha_{s'}}{\cos(\alpha_{s'} + \alpha_{t'})} \tag{C2}$$

(The $\partial\theta/\partial z$ and $\partial\theta/\partial r$ gradients are the ones which will be interpolated back to the orthogonal mesh and then transformed to get $\partial\theta/\partial s$ and $\partial\theta/\partial t$.)

(9) Interpolate, by using LININT calls, from $\partial\theta/\partial z$ and $\partial\theta/\partial r$ on the s'-t' mesh to obtain $\partial\theta/\partial z$ and $\partial\theta/\partial r$ at the s-t points of the orthogonal mesh which lie between the leading and trailing edges of the blade.

(10) Rotate the r- and z-coordinate lines through the angle φ to obtain $\partial\theta/\partial s$ and $\partial\theta/\partial t$ at the orthogonal mesh points within the blade (see fig. 28). The following equations are used:

$$\frac{\partial\theta}{\partial s} = \frac{\partial\theta}{\partial z} \cos \varphi + \frac{\partial\theta}{\partial r} \sin \varphi \tag{C3}$$

$$\frac{\partial\theta}{\partial t} = \frac{\partial\theta}{\partial r} \cos \varphi - \frac{\partial\theta}{\partial z} \sin \varphi \tag{C4}$$

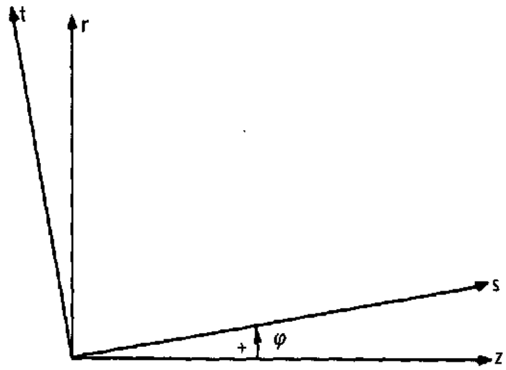


Figure 28. - Relation of z- and r-directions to s- and t-directions.

APPENDIX D

LINEAR INTERPOLATION IN A QUADRILATERAL

There are several instances where it is required for the program to interpolate from a two-dimensional array of values on a grid. If the grid were rectangular, this would be straightforward. However, usually this is not the case. In most cases the grid is a rectangular grid which is deformed like a net that has stretched out of shape. Thus, each region has four sides, but the corners are not necessarily right angles. The method of interpolation is the simplest possible. First, we find the particular quadrilateral containing the point, as shown in figure 29. All that is necessary is to

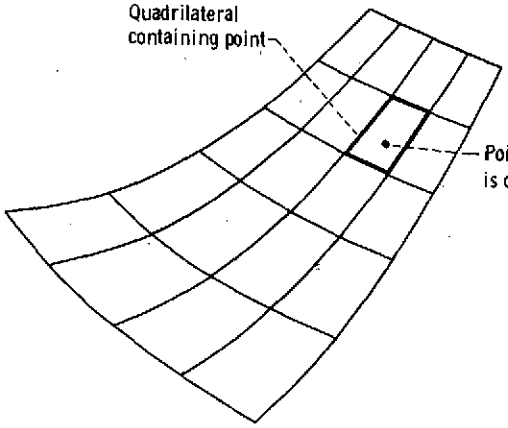


Figure 29. - Typical mesh region.

interpolate linearly within the quadrilateral. The interpolation is linear in the sense that it is linear along the boundary and between corresponding points along the boundary.

An illustration should clarify the manner of interpolation. Suppose it is desired to find the value at point P in figure 30. It is assumed that values of the function are known at the corner points A, B, C, and D. The function values at these points will be designated F_A, F_B, F_C, and F_D. Suppose that the point P lies on a line between points three-quarters of the way along AB and CD, as shown. Also suppose that P lies on a line between points two-thirds of the way along BD and AC, as shown. Then, we can interpolate linearly along AB and CD, followed by linear interpolation along the vertical line through P. If F is the interpolated value of P, we obtain

$$F = \frac{1}{12} F_A + \frac{1}{4} F_B + \frac{1}{6} F_C + \frac{1}{2} F_D$$

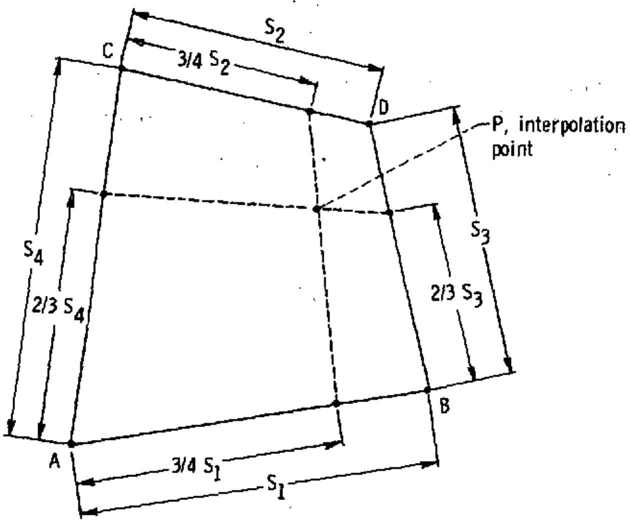


Figure 30. - Example of linear interpolation in a quadrilateral.

The same result is obtained if we interpolate linearly along BD and AC, followed by linear interpolation along the horizontal line through P.

Figure 31 shows a quadrilateral containing a point P_0 where it is desired to inter-

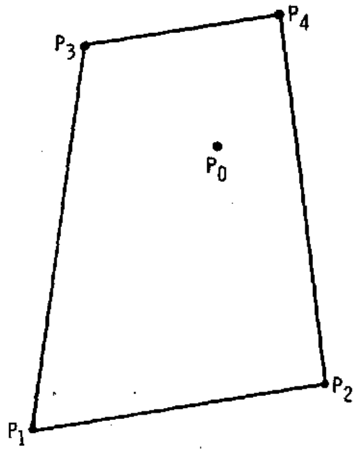


Figure 31. - Typical quadrilateral.

polate. It is assumed that the values of the function to be interpolated are known at the four corners, and that the coordinates of the point P_0 are given. The function values are denoted by z, and the coordinates by x and y. Subscripts are used to indicate the point. There are 14 values required to perform the interpolation: the coordinates of the four corners (eight values), the coordinates of the interpolation point (two values), and the function values at the four corners. If it is assumed that these 14 values are known, an equation for linear interpolation can be derived.

Figure 32 shows the same quadilateral as figure 31 but with the added lines P_5P_6 and P_7P_8. The line P_5P_6 passes through the point P_0 and is chosen so that $P_1P_5 : P_1P_3 = P_2P_6 : P_2P_4$. Similarly, P_7P_8 passes through P_0 and $P_1P_7 : P_1P_2 = P_3P_8 : P_3P_4$. Now, let

$$f_x = \frac{P_1P_7}{P_1P_2} \tag{D1}$$

$$f_y = \frac{P_1P_5}{P_1P_3} \tag{D2}$$

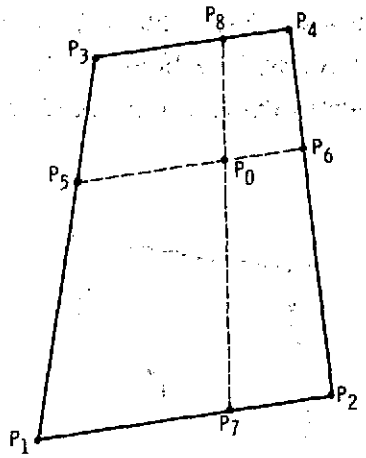


Figure 32. - Typical quadrilateral with interpolation lines.

The coordinates of any point P_i will be designated by (x_i, y_i). The difference of any two x or y values will be designated by $x_{ij} = x_i - x_j$ or $y_{ij} = y_i - y_j$. Thus,

$$f_y = \frac{x_{51}}{x_{31}} = \frac{y_{51}}{y_{31}} = \frac{x_{62}}{x_{42}} = \frac{y_{62}}{y_{42}} \tag{D3}$$

The equation of line P_5P_6 is

$$\frac{y - y_5}{x - x_5} = \frac{y_{65}}{x_{65}} \tag{D4}$$

By using equation (D3), y_5, y_6, x_5, and x_6 can be expressed in terms of f_y and the known values. For example,

$$y_5 = y_1 + y_{51} = y_1 - f_y y13$$

In a similar manner we obtain

$$\left.\begin{aligned} y_5 &= y_1 - f_y y13 \\ y_6 &= y_2 + f_y y_{42} \\ x_5 &= x_1 - f_y x_{13} \\ x_6 &= x_2 + f_y x_{42} \end{aligned}\right\} \tag{D5}$$

By substituting equations (D5) in (D4), we obtain

$$\frac{y - y_1 + f_y y_{13}}{x - x_1 + f_y x_{13}} = \frac{y_2 + f_y y_{42} - y_1 + f_y y_{13}}{x_2 + f_y x_{42} - x_1 + f_y x_{13}} \tag{D6}$$

This line passes through P_0, so when $x = x_0$, $y = y_0$. When this substitution is made and we multiply through by the denominators, we obtain a quadratic in f_y,

$$af_y^2 + bf_y + c = 0 \tag{D7}$$

where

$$\left.\begin{aligned} a &= y_{13}x_{42} - x_{13}y_{42} \\ b &= x_{13}y_{02} - y_{13}x_{02} + y_{01}x_{42} - x_{01}y_{42} \\ c &= y_{01}x_{21} - x_{01}y_{21} \end{aligned}\right\} \tag{D8}$$

In a similar manner, we can obtain a quadratic in f_x:

$$af_x^2 + bf_x + c = 0 \tag{D9}$$

where

$$\left.\begin{aligned} a &= y_{12}x_{43} - x_{12}y_{43} \\ b &= x_{12}y_{03} - y_{12}x_{03} + y_{01}x_{43} - x_{01}y_{43} \\ c &= y_{01}x_{31} - x_{01}y_{31} \end{aligned}\right\} \tag{D10}$$

If $a \neq 0$ in equation (D7) or (D9), there are two solutions for f_x or f_y. However, there will be only one value between zero and one. When two sides are parallel, a will be zero and only one solution exists. Caution is needed when a is not zero but is very small. In this case there is one and only one solution between zero and one; but if the usual quadratic formula is used, the answer will be inaccurate. The solution, however, can be accurately calculated by using a binomial expansion.

If we let f represent either f_x or f_y, the solution to either (D7) or (D9) can be written as

$$f = -\frac{b}{2a}\left(1 \pm \sqrt{1 - \frac{4ac}{b^2}}\right) \tag{D11}$$

When a is zero or small in magnitude, we want the root that is closest to zero. This is obtained by choosing the minus sign for the last term. Now we expand

$$\left(1 - \frac{4ac}{b^2}\right)^{1/2}$$

by the binomial series, to obtain

$$\sqrt{1 - \frac{4ac}{b^2}} = 1 - \frac{2ac}{b^2} - \frac{2a^2c^2}{b^4} - \frac{4a^3c^3}{b^6} - \frac{10a^4c^4}{b^8} - \ldots \tag{D12}$$

for $|4ac| < b^2$. Substituting equation (D12) in equation (D11) with the minus sign gives

$$f = -\frac{c}{b}\left(1 + \frac{ac}{b^2} + \frac{2a^2c^2}{b^4} + \frac{5a^3c^3}{b^6} + \ldots\right) \tag{D13}$$

Equation (D13) is used when ac/b^2 is small. Otherwise, the usual quadratic formula is used. In the program (i.e., in subroutine LININT, and also in subroutine CONTIN), equation (D13) is used whenever $|4ac| \le b^2/100$. Only three terms of the series are used; the term $5a^3c^3/b^6$ is dropped. This leads to a maximum relative error of less than 10^{-7}. When $|4ac| > b^2/100$, the quadratic formula will lose no more than two or three decimal places in accuracy.

There is one further point that must be considered. Up to this point, it has been assumed that the interpolation point is within the overall grid area, and thus we only need to interpolate within a quadrilateral. However, there are cases where extrapolation is necessary. In this case, the nearest quadrilateral is identified, and extrapolation is used. The procedure is similar, but one of the f's must be either negative or greater than 1. The problem, then, is to determine which f to use. Since the direction of the extrapolation is known, it is known whether f is negative or greater than 1. For example, suppose it was necessary to extrapolate below the bottom of the grid area. Then f_y must be negative. If only one of the two possible values is negative, the question is settled. If both are negative, the larger value (closest to zero) is used.

After both f_x and f_y are obtained, the linear interpolation can be performed to obtain z_0. Linear interpolation along P_1P_2 and P_3P_4 is followed by linear interpolation along P_7P_8. These interpolations are calculated by

$$z_7 = z_1 + f_x(z_2 - z_1)$$

$$z_8 = z_3 + f_x(z_4 - z_3)$$

$$z_0 = z_7 + f_y(z_8 - z_7)$$

Combining these equations, we get

$$z_0 = z_1(1 - f_x)(1 - f_y) + z_2f_x(1 - f_y) + z_3(1 - f_x)f_y + z_4f_xf_y \tag{D14}$$

APPENDIX E

SYMBOLS

B	tangential space between blades, corrected for loss of total pressure, rad
c_p	specific heat at constant pressure, J/(kg)(K)
$\bar{F}$	vector normal to mid-channel stream surface and proportional to tangential pressure gradient, N/kg
H	absolute total enthalpy, J/kg
I	rothalpy, $c_p T_i' - \omega\lambda$, meters^2/sec^2
m	meridional streamline distance, meters
p	pressure, N/meter^2
R	gas constant, J/(kg)(K)
r	radius from axis of rotation, meters
r_c	radius of curvature of meridional streamline, meters
s	distance along orthogonal mesh lines in throughflow direction (fig. 25), meters
T	temperature, K
t	distance along orthogonal mesh lines in direction across flow (fig. 25), meters
u	normalized stream function
V	absolute fluid velocity, meters/sec
W	fluid velocity relative to blade, meters/sec
W_{j+1}	W at next point, meters/sec
W_{j+1}^*	first estimate of W_{J+1}, meters/sec
W_{j+1}^{**}	second estimate of W_{j+1}, meters/sec
w	mass flow, kg/sec
z	axial coordinate, meters
α	angle between meridional streamline and axis of rotation (fig. 2, part I), rad
β	angle between relative velocity vector and meridional plane (fig. 2, part I), rad
γ	specific-heat ratio
ζ	coefficient in stream-function equation, defined in eq. (A3), meters/sec^2
θ	relative angular coordinate (fig. 2, part I), rad

λ prerotation, $(rV_\theta)_i$, meters^2/sec

ξ coefficient in stream-function equation, defined in eq. (A2), 1/meter

ρ density, kg/meter^3

φ angle between s-coordinate line and axis of rotation (fig. 25), rad

Ω overrelaxation factor

ω rotational speed (fig. 2, part I), rad/sec

Subscripts:

av average blade-to-blade value

b blade

bf blade flow

cr critical

fs free stream

hub hub

i inlet

l blade surface facing direction of positive rotation

le leading edge

m component in direction of meridional streamline

net net

o outlet

r component in radial direction

s component in s-direction

t component in t-direction

te trailing edge

tip tip

tr blade surface facing direction of negative rotation

z component in axial direction

θ component in tangential direction

Superscripts:

' absolute stagnation condition

'' relative stagnation condition

REFERENCES

1. Wu, Chung, Hua: A General Theory of Three-Dimensional Flow in Subsonic and Supersonic Turbomachines of Axial-, Radial-, and Mixed-Flow Types. NACA TN 2604, 1952.

2. Katsanis, Theodore: Use of Arbitrary Quasi-Orthogonals for Calculating Flow Distribution in the Meridional Plane of a Turbomachine. NASA TN D-2546, 1964.

3. Marsh, H.: A Digital Computer Program for the Through-Flow Fluid Mechanics in an Arbitrary Turbomachine Using a Matrix Method. R&M-3509, Aeronautical Research Council, Gt. Britain, 1968.

4. Davis, W. R.: A Matrix Method Applied to the Analysis of the Flow in Turbomachinary. Rep. ME/A-71-6, Carleton University, Ottawa, Canada, Sept. 1971.

5. Katsanis, Theodore: FORTRAN Program for Calculating Transonic Velocities on a Blade-to-Blade Stream Surface of a Turbomachine. NASA TN D-5427, 1969.

6. Katsanis, Theodore; and McNally, William D.: FORTRAN Program for Calculating Velocities and Streamlines on the Hub-Shroud Mid-Channel Flow Surface of an Axial- or Mixed-Flow Turbomachine. I - User's Manual. NASA TN D-7343, 1973.

7. McNally, William D.: FORTRAN Program for Generating a Two-Dimensional Orthogonal Mesh Between Two Arbitrary Boundaries. NASA TN D-6766, 1972.

8. Kannenberg, Robert G.: CINEMATIC - FORTRAN Subprograms for Automatic Computer Microfilm Plotting. NASA TM-X-1866, 1969.

9. McCracken, Daniel D.; and Dorn, William S.: Numerical Methods and FORTRAN Programming. John Wiley & Sons, Inc., 1964.

10. Varga, Richard S.: Matrix Iterative Analysis. Prentice-Hall, Inc., 1962.

11. Katsanis, Theodore: A Computer Program for Calculating Velocities and Streamlines for Two-Dimensional, Incompressible Flow in Axial Blade Rows. NASA TN D-3762, 1967.

12. Walsh, J. L.: Ahlberg, J. H.; and Nilson, E. N.: Best Approximation Properties of the Spline Fit. J. Math. Mech., vol. 11, no. 2, 1962, pp. 225-234.

☆ U.S. GOVERNMENT PRINTING OFFICE: 1974—739-159/108