NASA TECHNICAL NOTE

NASA TN D-8431

NASA TN D-8431

REVISED FORTRAN PROGRAM FOR CALCULATING VELOCITIES AND STREAMLINES ON THE HUB-SHROUD MIDCHANNEL STREAM SURFACE OF AN AXIAL-, RADIAL-, OR MIXED-FLOW TURBOMACHINE OR ANNULAR DUCT

II - Programmer's Manual

Theodore Katsanis and William D. McNally

Lewis Research Center

Cleveland, Ohio 44135

NATIONAL AERONAUTICS AND SPACE ADMINISTRATION • WASHINGTON D. C. • JULY 1977

1. Report No.	2. Government Accession No.	3. Recipient's Catalog No.
NASA TN D-8431		
4. Title and Subtitle REVISED FORTRAN PROGRAM FOR CALCULATING VELOCITIES AND STREAMLINES ON THE HUB-SHROUD MIDCHANNEL STREAM SURFACE OF AN AXIAL-, RADIAL-, OR MIXED-FLOW TURBOMACHINE OR ANNULAR DUCT. II - PROGRAMMER'S MANUAL		5. Report Date July 1977
		6. Performing Organization Code
7. Author(s) Theodore Katsanis and William D. McNally		8. Performing Organization Report No. E-8969
		10. Work Unit No. 505-04
9. Performing Organization Name and Address National Aeronautics and Space Administration, Lewis Research Center, Cleveland, Ohio 44135		11. Contract or Grant No.
		13. Type of Report and Period Covered Technical Note
12. Sponsoring Agency Name and Address National Aeronautics and Space Administration, Washington, D. C. 20546		14. Sponsoring Agency Code

15. Supplementary Notes

Supersedes NASA TN D-7344 (FORTRAN Program for Calculating Velocities and Streamlines on the Hub-Shroud Midchannel Flow Surface of an Axial- or Mixed-Flow Turbomachine. II - Programmer's Manual, 1974.)

16. Abstract

A FORTRAN-IV computer program has been developed that obtains a detailed subsonic or shock-free transonic flow solution on the hub-shroud midchannel stream surface of a turbomachine. The blade row may be fixed or rotating, and the blades may be twisted and leaned. Flow may be axial, mixed, or radial. This program is a revision of a previous program and this report supersedes NASA TN D-7344. Upstream and downstream flow variables may 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 stream 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))		18. Distribution Statement	
Meridional plane; Turbomachine flow; Midchannel stream surface; Axial-flow turbomachine; Mixed-flow turbomachine; Transonic flow; Radial-inflow turbine; Centrifugal compressor		Unclassified - unlimited STAR Category 02	
19. Security Classif. (of this report)	20. Security Classif. (of this page)	21. No. of Pages	22. Price*
Unclassified	Unclassified	231	A11

* For sale by the National Technical Information Service, Springfield, Virginia 22161

CONTENTS

	Page
SUMMARY	1
INTRODUCTION	2
OVERALL PROGRAM PROCEDURE	5
DETAILED PROGRAM PROCEDURE	8
STORAGE REQUIREMENTS	13
CONVENTIONS USED IN PROGRAM	13
LABELED COMMON BLOCKS	15
ROTATED COORDINATES	16
INCOMPRESSIBLE FLOW	19
MAIN PROGRAM	19
SUBROUTINES	19
Subroutine INPUT	19
Subroutine MESHO	20
Subroutine PRECAL	24
Subroutine THETOM	26
Subroutine THIKOM	27
Subroutine INIT	27
Subroutine COEF	27
Subroutine SOR	28
Subroutine LOSSOM	29
Subroutine NEWRHO	31
Subroutine OUTPUT	34
Subroutine BLDVEL	38
Subroutine ILETE	39
Subroutine INDEV	40
Subroutine TSONIN	41
Subroutine TVELCY	42
Subroutine LINDV	47
Functions LAMDAF, RVTHTA, TIPF, and RHOIPF	47
Subroutine CONTIN	48
Subroutine PABC	52
Subroutine INRSCT	52
Subroutine LININT	54
Subroutine ROTATE	58

Page

Subroutine SPLINE 59
Subroutine SPLINT 60
Subroutine SLOPES 60
Subroutines SPLISL and SPINSL 61
Plotting Subroutines 61

MAIN DICTIONARY 61

PROGRAM LISTING 103

APPENDIXES
A - FINITE-DIFFERENCE FORM OF STREAM-FUNCTION EQUATION 201
B - MATCHING UPSTREAM AND DOWNSTREAM FLOW CONDITIONS TO STREAM-FUNCTION SOLUTION 206
C - CALCULATION OF PARTIAL DERIVATIVES OF THETA ON ORTHOGONAL MESH 207
D - INCOMPRESSIBLE STREAM-FUNCTION EQUATION 211
E - GENERATION OF LEADING- AND TRAILING-EDGE RADII ON TSONIC BLADE SECTIONS 212
F - CALCULATION OF CHANGE OF DENSITY DUE TO BLOCKAGE AT BLADE LEADING OR TRAILING EDGE 215
G - LINEAR INTERPOLATION IN A QUADRILATERAL 219
H - SYMBOLS 225

REFERENCES 228

REVISED FORTRAN PROGRAM FOR CALCULATING VELOCITIES AND STREAMLINES ON THE HUB-SHROUD MIDCHANNEL STREAM SURFACE OF AN AXIAL-, RADIAL-, OR MIXED-FLOW TURBOMACHINE OR ANNULAR DUCT

II - PROGRAMMER'S MANUAL*

by Theodore Katsanis and William D. McNally

Lewis Research Center

SUMMARY

A FORTRAN-IV computer program has been developed that obtains a detailed subsonic or transonic flow solution on the hub-shroud midchannel stream surface of a single blade row of a turbomachine. A solution can also be obtained for an annular duct without blades. 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 the blades may be twisted and leaned. The flow may be axial, mixed, or radial. Upstream and downstream flow conditions can vary from hub to shroud, and provision is made for an approximate correction for loss of stagnation pressure. Viscous forces are neglected along solution mesh lines running from hub to shroud.

The present program is a revision of a previous program and this report supersedes NASA TN D-7344. The primary revisions are to extend the program to handle nonaxial flows without restriction, to handle annular ducts without blades, to allow for any specified streamwise loss distribution, and to make numerous detailed improvements for more accurate and efficient calculations.

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 that is used to obtain a velocity-gradient solution at the full mass flow.

The program is reported in two volumes, with part I as the user's manual and part II as the programmer's manual. This report, part II, contains all the 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.

---

*Supersedes NASA TN D-7344.

INTRODUCTION

The design of blades for compressors and turbines ideally requires analytical methods for unsteady, three-dimensional, turbulent, 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 used with the present generation of computers. However, they use excessive computer time. 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 midchannel 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 (fig. 1).

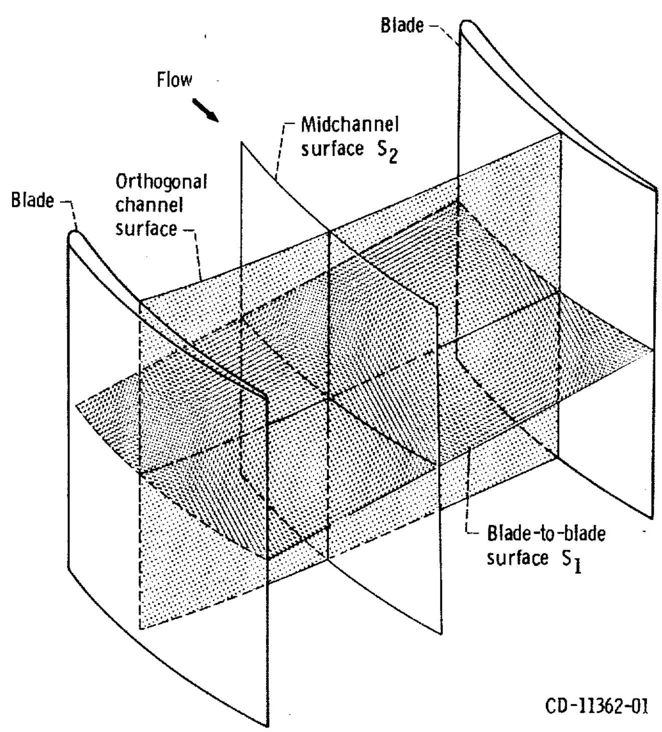


Figure 1. - Two-dimensional analysis surfaces in a turbomachine.

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, especially when input is needed to use in blade-to-blade calculations. The solution can be obtained either by the quasi-orthogonal method, which solves the velocity-gradient equation from hub to shroud on the meridional stream surface (ref. 2), or by a finite-difference method, which solves a finite-difference equation for stream function on the same stream surface. 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 blade aspect ratios are above 1. Difficulties are also encountered with curved passages and blades with low hub-tip ratios. For such cases, the most promising method is the finite-difference solution, but this solution is limited to completely subsonic flows.

Finite-difference programs for flow on the midchannel surface of a turbomachine have been reported in the literature. However, many of these programs are proprietary or are of limited generality. The program reported herein is very general and has been thoroughly tested and refined as the result of extensive usage at the Lewis Research Center.

The program described in this report uses both the finite-difference and the quasi-orthogonal (velocity gradient) methods, combined in a way that 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 called MERIDL has been written to perform these calculations. This program is written for an axial-, mixed-, or radial-flow turbomachine blade row, either a compressor or turbine, or for an annular duct. 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 the blades may be twisted and leaned. The blades can have a 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. 3). Information needed to prepare all the input for TSONIC is calculated and printed by MERIDL.

The MERIDL program reported herein is a revision of the program described in references 4 and 5. Two types of changes were made: first, extensions to the capability of the program to handle cases beyond those originally offered; and second, revisions to improve the accuracy and reduce the run time of the program. Although the input form has been extended to handle additional input where required, any input that was satisfactory for the original MERIDL program is still satisfactory for the revised MERIDL program. The following list itemizes the major extensions and revisions to the program (additional internal changes are also documented in this report):

(1) The program has been extended to handle nonaxial flows without restriction as to the direction of flow.

(2) The program has been extended to handle an annular duct without blades.

(3) The program has been extended to permit the user to specify an arbitrary streamwise distribution of loss within the blade row. This is in addition to the original provision for hub-to-shroud loss distribution.

(4) The program has been modified so that the blade thickness can be specified precisely by a set of tangential thickness coordinates. The original program required specification of thickness normal to the mean camber line on an input blade section. This normal thickness was influenced by blade lean, camber, and nonparallel blade surfaces and was difficult to specify accurately for some blade shapes.

(5) If desired, the leading- and trailing-edge mean camber line tangent angles can be specified as input. This simplifies the specification of some blade shapes.

(6) Output quantities have been added to station-line output to give absolute velocity components and to give static as well as absolute and relative total temperature, density, and pressure.

(7) Several informational messages have been added to the output.

(8) Additional error messages have been provided.

(9) Upstream and downstream boundary conditions have been changed to give improved convergence and a better quality solution near these boundaries.

(10) Interpolational and calculational procedures near the leading and trailing edges have been improved to give better convergence and smoother solutions in these regions.

(11) Numerous small changes have been made to improve the accuracy and reduce the run time of the 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 requirements for the program code depend on the computer system and compiler being used. Run times for the program range from 3 to 15 minutes on IBM 360-67 equipment.

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 describes the method of analysis and the input and output, gives a numerical example, and derives the mathematical equations used (in the appendixes). 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 the numerical techniques used and derive certain numerical algorithms.

OVERALL PROGRAM PROCEDURE

This section gives an overall view of the program calculational procedure. The next 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 principal subroutines are called by it. Figure 2 is a flow chart of the main program. The first step is to read and print out all the input data. This is done by the subroutine INPUT. 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. Subroutine 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. INPUT also calculates tangential blade thickness, if it is not given directly as input. The next step is to call subroutine INPLOT, which plots all the upstream and downstream input flow variables as well as the input blade sections from hub to shroud.

The next subroutine is MESHO, which calculates the coordinates of the orthogonal mesh in the solution region. After this, subroutine PRECAL is called to calculate those quantities that remain fixed throughout the calculations. These quantities include the s and t mesh coordinates, hub and shroud wall curvatures, and leading- and trailing-edge z- and r-coordinates at horizontal mesh lines. Subroutine PRECAL also calls THETOM and THIKOM. Subroutine THETOM calculates $\partial\theta/\partial s$ and $\partial\theta/\partial t$ at the orthogonal mesh points. (All symbols are defined in appendix H.) These partials are used later 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. 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 subroutine MEPLOT is called to plot the meridional plane view of the blade and passage and to plot the orthogonal mesh. Then subroutine INIT is called to initialize array variables as required for the first iteration. Most variables are set either to zero or to some value that will avoid division by zero later on.

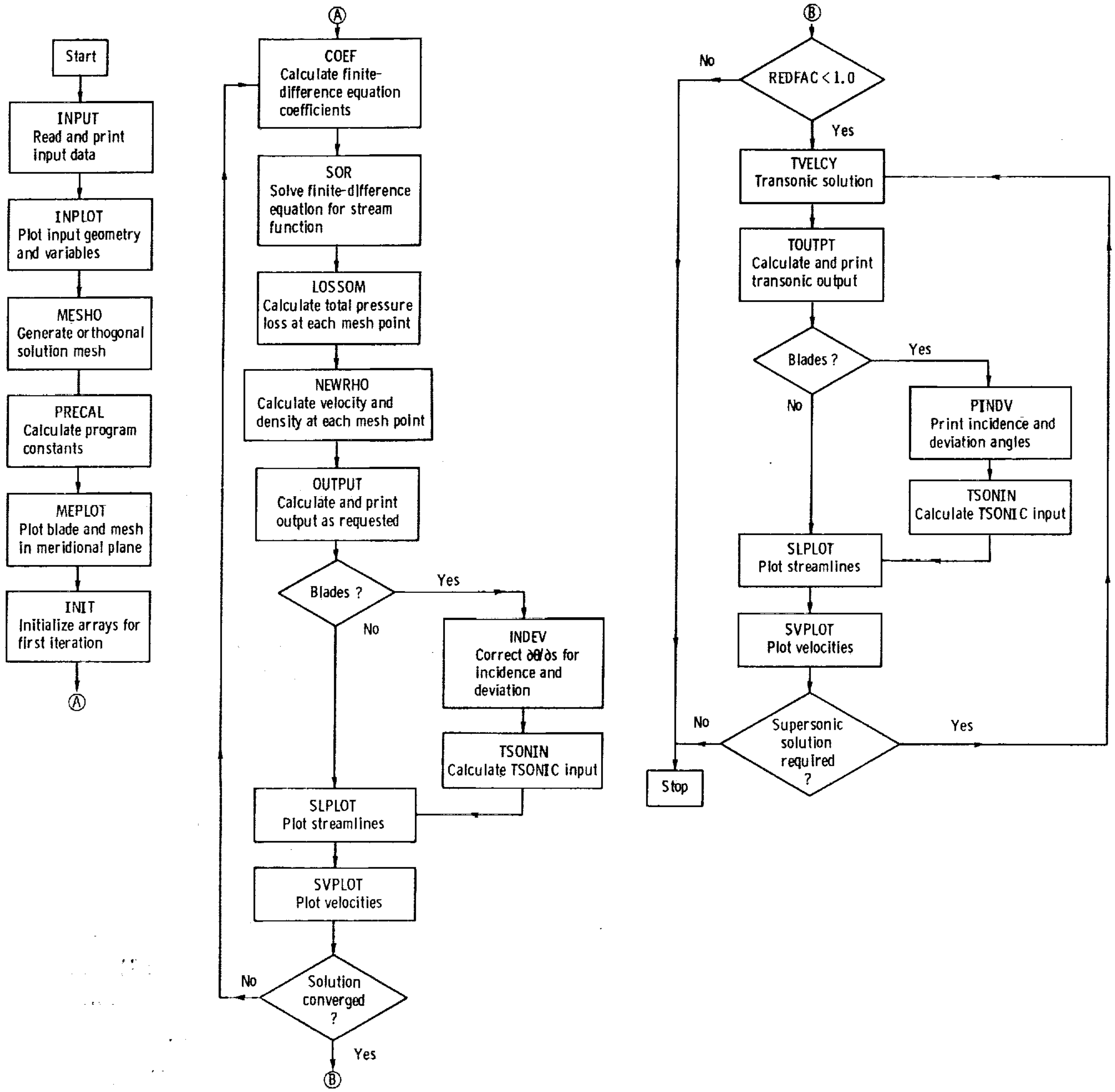


Figure 2. - Flow chart of main program.

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 four subroutines called to obtain the solution to the linearized equation: COEF, SOR, LOSSOM, 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. Calls to these eight 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 $\partial(rV_\theta)/\partial t$, 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 LOSSOM calculates the ratio of actual to ideal relative stagnation pressure downstream of the blade and then distributes the loss linearly, or as specified by the input, through the blade row from the leading to the trailing edge, or through the annular passage if no blades are present. The method of making loss corrections is discussed in appendix D of part I (ref. 6).

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 (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. If there are blades, 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 iteration. And BLDVEL calculates F_t at each point by using equation (A4). The vector component F_t 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-shroud station lines.

After OUTPUT, subroutines INDEV and TSONIN are called if there are blades. Subroutine INDEV calculates a correction to $\partial\theta/\partial s$ for a short distance into the blade row to match the mean surface within the blade row 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). INDEV also calculates and prints out incidence and deviation angles if this is requested. Then TSONIN will calculate and print TSONIC input if desired. Also 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 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 outlet 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. After TOUTPT, if there are blades, PINDV is called to print incidence and deviation angles. Then the same sequence of TSONIN, 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 (ref. 6), appendix C); but if desired, both the larger ("supersonic") and smaller solutions can be obtained. If both solutions are desired, TVELCY, TOUTPT, PINDV, TSONIN, SLPLOT, and SVPLOT are called again. This completes the program.

DETAILED PROGRAM PROCEDURE

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

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

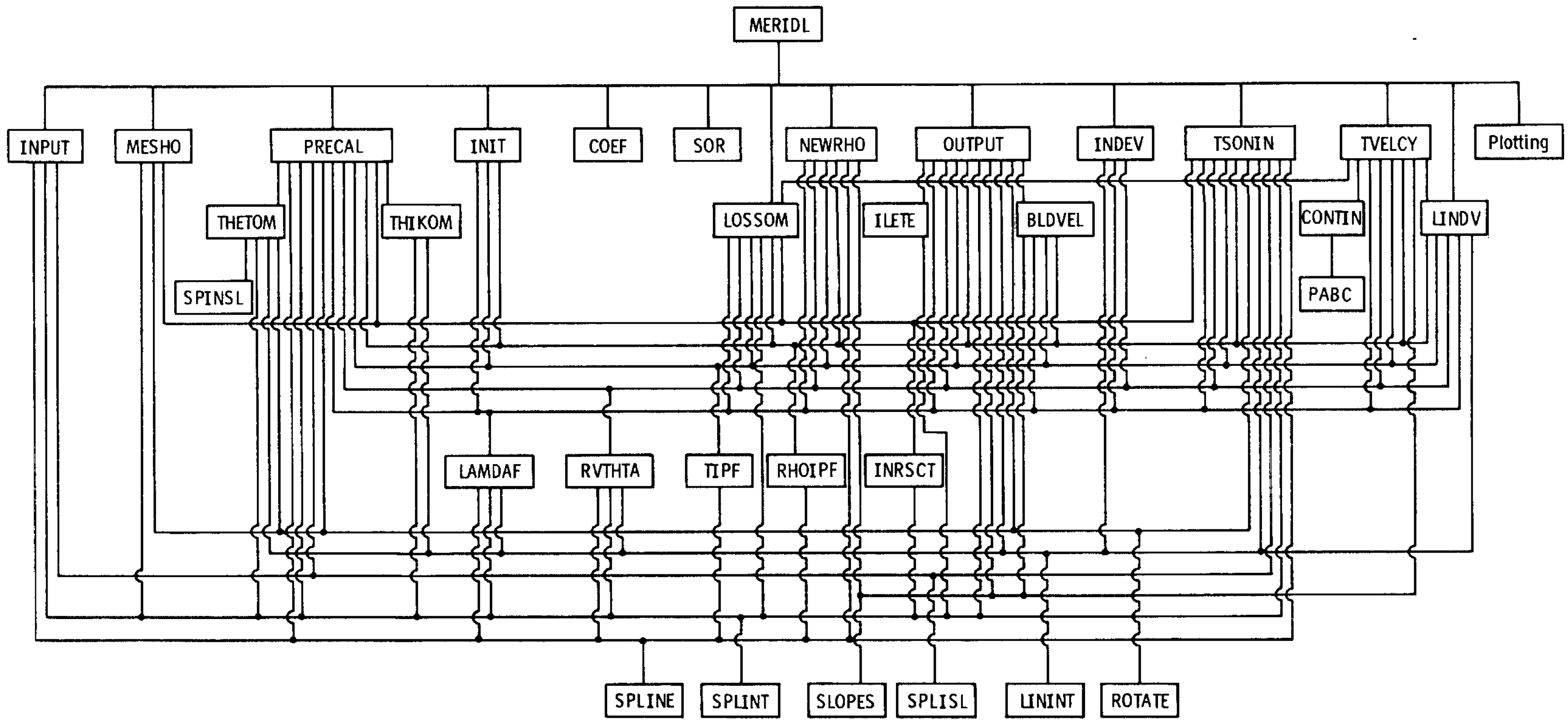


Figure 3. - Calling relation of subroutines. Called subroutines are always below the calling subroutine. (This is not a flow chart.)

TABLE I. - SUBROUTINE CALLS AND COMMON BLOCKS

Subroutine name or entry point	COMMON blocks (a)	Called subroutines	Calling subroutines
MERIDL	/---/ /INPUTT/ /CALCON/ /VARCOM/ /ROTATN/ /SLCOM/ /INDCOM/ /PLTCOM/	INPUT MESHO PRECAL INIT COEF SOR LOSSOM NEWRHO OUTPUT INDEV TSONIN TVELCY TOUTPT (OUTPUT) PINDV (LINDV) Plotting subroutines	None - main program
INPUT	/---/ /INPUTT/ /CALCON/	SPLINE SPLINT SPLENT (SPLINT) SPLISL	MERIDL
MESHO	/INPUTT/ /CALCON/ /ROTATN/	INRSCT ROTATE SPLINT SPLENT (SPLINT)	MERIDL
PRECAL	/---/ /INPUTT/ /CALCON/ /VARCOM/ /ROTATN/	THETOM THIKOM LAMNIT (LAMDAF) RVTNIT (RVTHTA) TIPNIT (TIPF) RHINIT (RHOIPF) INRSCT ROTATE SPLINE SPLINT SPLISL	MERIDL
THETOM	/---/ /INPUTT/ /CALCON/ /ROTATN/ /INDCOM/	LININT ROTATE SPLINT SPINSL	PRECAL
THIKOM	/INPUTT/ /CALCON/ /ROTATN/ /INDCOM/	LININT SPLINT	PRECAL
INIT	/INPUTT/ /CALCON/ /VARCOM/	LAMDAF TIPF RHOIPF	MERIDL
COEF	/---/ /INPUTT/ /CALCON/ /VARCOM/	None	MERIDL
SOR	/---/ /INPUTT/ /CALCON/ /VARCOM/	None	MERIDL
LOSSOM (see entry LOSSTV)	/---/ /INPUTT/ /CALCON/ /VARCOM/ /ROTATN/	LAMDAF RVTHTA TIPF RHOIPF LAMNIT (LAMDAF) RVTNIT (RVTHTA) TIPNIT (TIPF) RHINIT (RHOIPF) INRSCT SPLINT SPLENT (SPLINT)	MERIDL
LOSSTV (entry point for LOSSOM)	/---/ /INPUTT/ /CALCON/ /VARCOM/ /ROTATN/	LAMDAF RVTHTA TIPF RHOIPF INRSCT SPLINT SPLENT (SPLINT)	TVELCY
NEWRHO	/---/ /INPUTT/ /CALCON/ /VARCOM/	LAMDAF RVTHTA TIPF RHOIPF SPLINE SLOPES	MERIDL
OUTPUT (entry point TOUTPT)	/---/ /INPUTT/ /CALCON/ /VARCOM/ /ROTATN/ /SLCOM/	BLDVEL ILETE LAMDAF RVTHTA TIPF RHOIPF LININT ROTATE SPLINT SLOPES	MERIDL
BLDVEL	/---/ /INPUTT/ /CALCON/ /VARCOM/	LAMDAF TIPF RHOIPF SLOPES	OUTPUT TOUTPT
ILETE	/INPUT/ /CALCON/ /SLCOM/	SPLINT SPLENT (SPLINT)	OUTPUT TOUTPT
INDEV	/---/ /INPUTT/ /CALCON/ /VARCOM/ /ROTATN/ /INDCOM/	LAMDAF RVTHTA LININT	MERIDL

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

TABLE I. - Continued.

Subroutine name or entry point	COMMON blocks (a)	Called subroutines	Calling subroutines
TSONIN	/---/ /INPUTT/ /CALCON/ /VARCOM/ /ROTATN/ /SLCOM/ /INDCOM/	LAMDAF RVTHTA TIPF RHOIPF INRSCT LININT ROTATE SPLINE SPLINT SPLISL	MERIDL
TVELCY	/---/ /INPUTT/ /CALCON/ /VARCOM/	LOSSTV (LOSSOM) LINDV TINDV (LINDV) LAMDAF RVTHTA TIPF RHOIPF LAMNIT (LAMDAF) RVTNIT (RVTHTA) CONTIN SLOPES	MERIDL
LINDV (entry point - TINDV; see also entry PINDV)	/---/ /INPUTT/ /CALCON/ /VARCOM/ /ROTATN/ /INDCOM/	LAMDAF RVTHTA TIPF RHOIPF LININT	TVELCY
PINDV (entry point for LINDV)	/---/ /INPUTT/ /CALCON/ /VARCOM/ /ROTATN/ /INDCOM/	LAMDAF RVTHTA	MERIDL
LAMDAF (see entry LAMNIT)	/---/ /INPUTT/ /CALCON/ /VARCOM/ /ROTATN/	None	INIT LOSSOM LOSSTV NEWRHO OUTPUT TOUTPT BLDVEL INDEV TSONIN TVELCY LINDV TINDV PINDV
LAMNIT (entry point for LAMDAF)	/---/ /INPUTT/ /CALCON/ /VARCOM/ /ROTATN/	LININT SPLINE SPLINT	PRECAL LOSSOM TVELCY
RVTHTA (see entry RVTNIT)	/---/ /INPUTT/ /CALCON/ /VARCOM/ /ROTATN/	None	LOSSOM LOSSTV (LOSSOM) NEWRHO OUTPUT TOUTPT (OUTPUT) INDEV TSONIN TVELCY LINDV TINDV (LINDV)
RVTNIT (entry point for RVTHTA)	/---/ /INPUTT/ /CALCON/ /VARCOM/ /ROTATN/	LININT SPLINE SPLINT	PRECAL LOSSOM TVELCY
TIPF (see entry TIPNIT)	/INPUTT/ /CALCON/	None	INIT LOSSOM LOSSTV (LOSSOM) NEWRHO OUTPUT TOUTPT (OUTPUT) BLDVEL TSONIN TVELCY LINDV TINDV (LINDV)
TIPNIT (entry point for TIPF)	/INPUTT/ /CALCON/	SPLINE	PRECAL LOSSOM
RHOIPF (see entry RHINIT)	/INPUTT/ /CALCON/	None	INIT LOSSOM LOSSTV (LOSSOM) NEWRHO OUTPUT TOUTPT (OUTPUT) BLDVEL TSONIN TVELCY LINDV TINDV (LINDV)
RHINIT (entry point for RHOIPF)	/INPUTT/ /CALCON/	SPLINE	PRECAL LOSSOM
CONTIN	/---/	PABC	TVELCY
PABC	None	None	CONTIN
INRSCT	/---/	SPLINT	MESHO PRECAL LOSSOM TSONIN

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

TABLE I. - Concluded.

Subroutine name or entry point	COMMON blocks (a)	Called subroutines	Calling subroutines
LININT	/---/	None	THETOM THIKOM OUTPUT TOUTPT INDEV TSONIN LINDV TINDV LAMNIT RVTNIT
ROTATE	None	None	MESHO PRECAL THETOM OUTPUT TOUTPT (OUTPUT) TSONIN
SPLINE	/---/	None	INPUT PRECAL NEWRHO TSONIN LAMNIT (LAMDAF) RVTNIT (RVTHTA) TIPNIT (TIPF) RHINIT (RHOIPF)
SPLINT (see entry SPLENT)	/---/	None	INPUT MESHO PRECAL THETOM THIKOM LOSSOM LOSSTV (LOSSOM) OUTPUT TOUTPT (OUTPUT) TSONIN ILETE LAMNIT (LAMDAF) RVTNIT (RVTHTA) INRSCT
SPLENT (entry point for SPLINT)	/---/	None	INPUT MESHO LOSSOM LOSSTV (LOSSOM) ILETE
SLOPES	None	None	NEWRHO OUTPUT TOUTPT (OUTPUT) BLDVEL TVELCY
SPLISL	/---/	None	INPUT PRECAL TSONIN
SPINSL	/---/	None	THETOM
Plotting subroutines	/INPUT/ /CALCON/ /ROTATN/ /SLCOM/ /PLTCOM/	Not applicable	MERIDL

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

aries are described.

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

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

STORAGE REQUIREMENTS

The MERIDL program has been implemented on the NASA Lewis time-sharing IBM-TSS/360-67 computer. The program consists of approximately 5000 lines of code. 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 700 000 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.

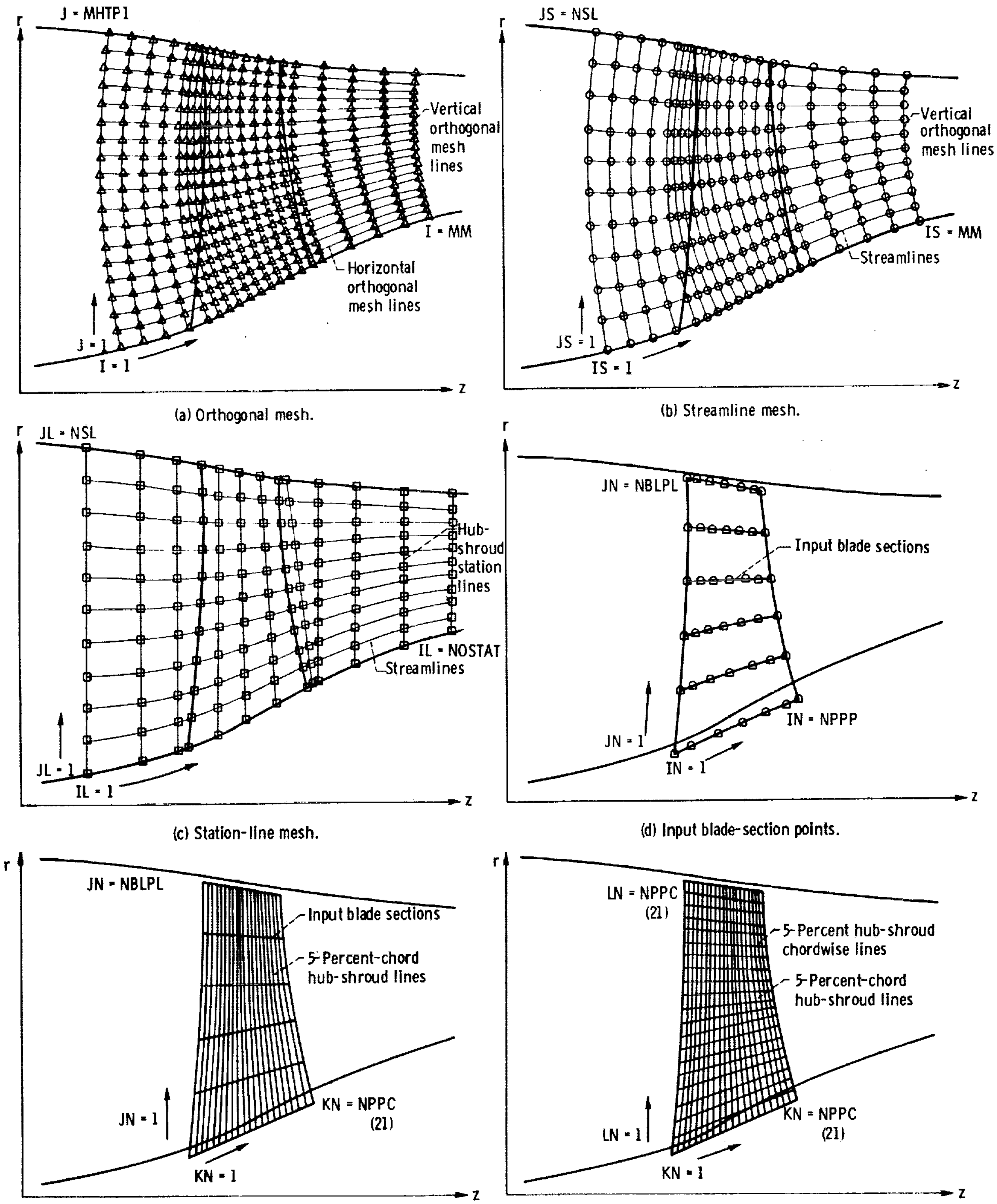
As an example, consider the two-dimensional array ALPHA. This variable is in the /VARCOM/ COMMON block and is dimensioned ALPHA (100, 101) in the program listing. In the main dictionary, it is listed as ALPHA (MM, MHTP1). Suppose that the maximum desired value for MM is 60 and that for MHT it is 40. 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 that 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, and calls to ROTATE must be changed to reflect the dimensions of the second, third, and last two 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 five special mesh schemes used, as illustrated in figure 4. For each mesh, different conventions are used to indicate



(a) Orthogonal mesh.

(b) Streamline mesh.

(c) Station-line mesh.

(d) Input blade-section points.

(e) Semi-alternate mesh.

(f) Full alternate mesh.

Figure 4. - Six meshes used in MERIDL.

mesh position. The subscripts I and J are used to denote orthogonal mesh position. The I is used to denote the vertical mesh line 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 LN denote points on the alternate blade mesh located at 5-percent-chord and 5-percent-span 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, and variables ending with ROT or R are usually coordinates with respect to the rotated axes.

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 to indicate 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 for $\partial p/\partial r$.

All subroutines used for plotting have PLOT in the name.

LABELED COMMON BLOCKS

Most variables that 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 constants that are initially calculated and are usually not changed later.

/VARCOM/ is used for all orthogonal mesh-point arrays that are changed in each iteration.

/ROTATN/ is used for coordinates with respect to rotated axes.

/SLCOM/ is used for output data along streamlines.

/INDCOM/ is used for quantities calculated by THETOM to be used by INDEV, LINDV, and TSONIN.

/PLTCOM/ is used to plot data for hub, shroud, and blade leading and trailing edges.
Table I shows which COMMON blocks are needed in each subroutine.

ROTATED COORDINATES

Spline curves are used for most geometrical curve fitting in the MERIDL program. Since spline curves are limited to angles somewhat less than 90°, an option to use rotated coordinates may be exercised by the user when flow angles are much over 45° from axial. Rotated z- and r-coordinates are illustrated in figure 5.

Subroutine ROTATE is used by MERIDL to transfer from unrotated to rotated coordinates (and vice versa). The option to work in rotated coordinates is specified through the input variables LROT and ANGROT. If ANGROT is not given as input (LROT = 0), there will be no difference between unrotated and rotated coordinates in the program, although the ROTATE calls are still made in the subroutines.

All coordinates read into MERIDL as input are unrotated coordinates. Most of these are never rotated by the program. Likewise, some geometrical arrays calculated by

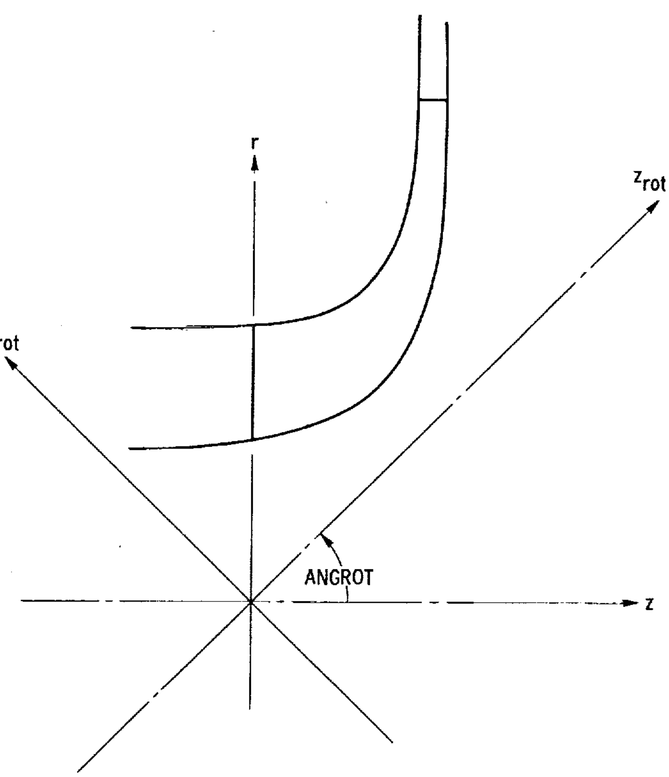


Figure 5. - Rotated z- and r-coordinates.

the program are never rotated. On the other hand, many geometrical variables are calculated in the rotated system. Some of these are later unrotated, while others are not. The variables associated with each of these options are summarized in table II, which shows whether a variable was calculated or used in either rotated or unrotated z- and r-coordinates. Table III shows which subroutines are involved in calculating, rotating, and unrotating these variables.

In the description of individual subroutines, reference is made to both rotated and unrotated points. Unrotated points always refer to the z and r input coordinate directions. Rotated points refer to coordinates that have been rotated as shown in figure 5.

TABLE II. - FORTRAN z- AND r-VARIABLES AFFECTED BY ROTATION

(a) Unrotated variables

Read in unrotated and never rotated	
ZBL(NPPP, NBLPL)	RBL(NPPP, NBLPL)
ZHUB(NHUB)	RHUB(NHUB)
ZTIP(NTIP)	RTIP(NTIP)
ZOMIN	ROMIN
ZOMBI	ROMBI
ZOMBO	ROMBO
ZOMOUT	ROMOUT
ZHIN	RHIN
ZTIN	RTIN
ZHOUT	RHOUT
ZTOUT	RTOUT
	RADIN(NIN)
	RADOUT(NOUT)
Calculated unrotated and never rotated	
ZOM(MM, MHTP1)	ROM(MM, MHTP1)
ZLEOM(MHTP1)	RLEOM(MHTP1)
ZTEOM(MHTP1)	RTEOM(MHTP1)

(b) Rotated variables

Read in unrotated and then rotated for internal use	
ZHST(NOSTAT)	RHST(NOSTAT)
ZTST(NOSTAT)	RTST(NOSTAT)
Calculated rotated and never unrotated	
ZOMROT(MM, MHTP1)	ROMROT(MM, MHTP1)
ZBLROT(NPPP, NBLPL)	RBLROT(NPPP, NBLPL)
ZPCT1(NBLPL)	RPCT1(NBLPL)
ZPCT2(NBLPC)	RPCT2(NBLPC)
ZHROT(NHUB)	RHROT(NHUB)
ZTROT(NTIP)	RTROT(NTIP)
ZLE(NBLPL)	RLE(NBLPL)
ZTE(NBLPL)	RTE(NBLPL)
ZLEOMR(MHTP1)	RLEOMR(MHTP1)
ZTEOMR(MHTP1)	RTEOMR(MHTP1)
ZLEH	RLEH
ZLET	RLET
ZTEH	RTEH
ZTET	RTET
ZLESL	RLESL
ZTESL	RTESL
ZRAD(NHUB or NTIP, MHTP1)	RRAD(NHUB or NTIP, MHTP1)
Calculated rotated and only unrotated for printing and plotting	
ZSL(MM, NSL)	RSL(MM, NSL)
ZST(NSL, NOSTAT)	RST(NSL, NOSTAT)
ZPC(NPPC, NBLPL)	RPC(NPPC, NBLPL)

TABLE III. - SUBROUTINES INVOLVED WITH ROTATION

Subroutine	Variables read in or calculated	Rotated or unrotated	Comments
INPUT	ZBL, RBL	Unrotated	Read in unrotated
	ZHUB, RHUB		
	ZTIP, RTIP		
	ZOMIN, ROMIN		
	ZOMBI, ROMBI		
	ZOMBO, ROMBO		
	ZOMOUT, ROMOUT		
	ZHIN, RHIN		
	ZTIN, RTIN		
	ZHOUT, RHOUT		
	ZTOUT, RTOUT		
	ZHST, RHST		
	ZTST, RTST		
	RADIN		
	RADOUT		
MESHO	ZHROT, RHROT	Rotated	Rotated from ZHUB, RHUB
	ZTROT, RTROT	Rotated	Rotated from ZTIP, RTIP
	ZOMROT, ROMROT	Rotated	Calculated
	ZOM, ROM	Unrotated	Unrotated from ZOMROT, ROMROT
PRECAL	ZHST, RHST	Rotated	Rotated from input values with same name
	ZTST, RTST		Rotated from input values with same name
	ZBLROT, RBLROT		Rotated from ZBL, RBL
	ZLE, RLE		Calculated
	ZTE, RTE		
	ZLEH, RLEH		
	ZLET, RLET		
	ZTEH, RTEH		
	ZTET, RTET		
	ZLEOMR, RLEOMR		
	ZTEOMR, RTEOMR		
	ZLEOM, RLEOM	Unrotated	Unrotated from ZLEOMR, RLEOMR
	ZTEOM, RTEOM	Unrotated	Unrotated from ZTEOMR, RTEOMR
THETOM	ZPC, RPC	Rotated	Calculated
	ZPCT1, RPCT1		Calculated on semi-alternate mesh (fig. 26)
	ZPCT2, RPCT2		Calculated on alternate mesh (fig. 27)
	ZRAD, RRAD		Calculated
OUTPUT	ZSL, RSL	Rotated	Calculated rotated, unrotated for printing, then rotated for later use
	ZST, RST	Rotated	Calculated rotated, unrotated for printing, and left unrotated
TSONIN	ZLESL, RLESL	Rotated	Calculated
	ZTESL, RTESL	Rotated	Calculated

INCOMPRESSIBLE FLOW

Provision has been made for incompressible flow analysis by MERIDL. The main difference is that the density at each point is constant; so the density arrays are initialized to the input density value. A streamwise loss of total pressure that is uniform from hub to shroud has no effect on the solution and is not considered for the incompressible case. The present method of solution is very sensitive to a hub-shroud variation of total pressure for incompressible flow, so this variation is likewise not considered. Thus, the PERLOS, PRIP, PROP, and LOSOUT arrays are not used, and any calculations involving temperature or pressure are omitted from the calculation for incompressible flow. A derivation of the necessary changes to the stream-function equation (A1) is given in appendix D.

The subroutines with differences for an incompressible flow calculation are INPUT, PRECAL, INIT, COEF, LOSSOM, NEWRHO, OUTPUT, BLDVEL, and TSONIN. These differences are mainly that the variable arrays RHO and RHOAV are set to the input density, and thereafter all calculations of density, temperature, and pressure (including ξ and ζ) are omitted.

MAIN PROGRAM

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

SUBROUTINES

Subroutine INPUT

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

All input cards are first read and printed on the output listing in the same form and order in which they are given. All array bounds are then checked to see if they are within limits, and some miscellaneous constants are initialized. Estimates are made of various required upstream and downstream flow conditions that were not given as input because other input options were used. Finally, blade surface and/or blade thickness coordinates are calculated when not given as input.

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 three other subroutines - ROTATE, SPLINT, and INRSCT. A flow chart for MESHO is given in figure 6.

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. MESHO initially rotates the hub and shroud geometry through the input angle, ANGROT, if ANGROT is specified, so that the mesh generation is done in the rotated coordinate system.

Then MESHO calculates the horizontal, or streamwise, orthogonals, as follows. If NHUB equals NTIP, lines are extended from each of the input points on the hub to the corresponding points on the shroud. If NHUB and NTIP are unequal, lines are extended from input points on the surface with a larger number of points to an equal number of equally spaced points on the opposite surface. In either case, each of these hub-shroud lines is then divided into MHT equal increments. The resulting coordinates are in the ZRAD and RRAD arrays. The hub-shroud lines and resulting horizontal orthogonals are shown in figure 7.

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 8. Before this process begins, however, the input mesh boundary points on the hub - ZOMIN, ZOMBI, ZOMBO, and ZOMOUT - are calculated in the rotated coordinate system. Rotated orthogonal mesh points (ZOMROT) are then calculated on the hub between these boundary points. The corresponding r-coordinates (ROMROT) and slopes (SLOM) are obtained by a SPLINT call.

The procedure for calculating vertical orthogonal links between the horizontal orthogonals is then begun. This procedure, shown in figure 9, is analogous to a technique for solving ordinary differential equations known as the improved Euler method or Heun's method (ref. 7). Beginning at known orthogonal mesh points on the lower orthogonal, normals are constructed (such as line ① in fig. 9) to the upper orthogonal. The intersection coordinates of these lines with the upper orthogonal are obtained with INRSCT calls, and then the slopes of the upper orthogonal at the intersections are obtained by a SPLINT call. Lines such as line ② in figure 9 are then constructed in such

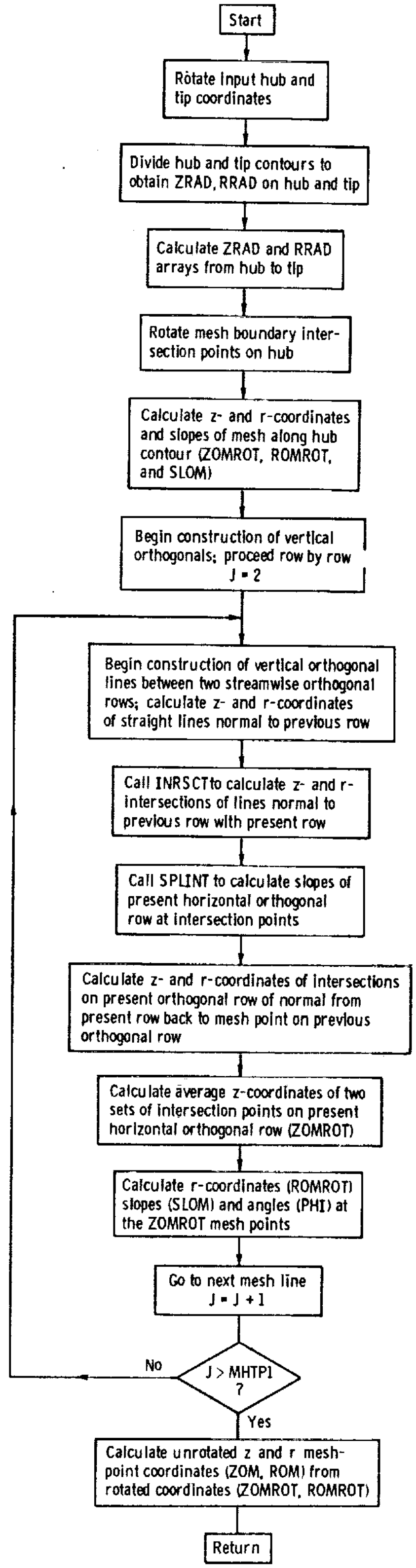


Figure 6. - Flow chart for MESHO.

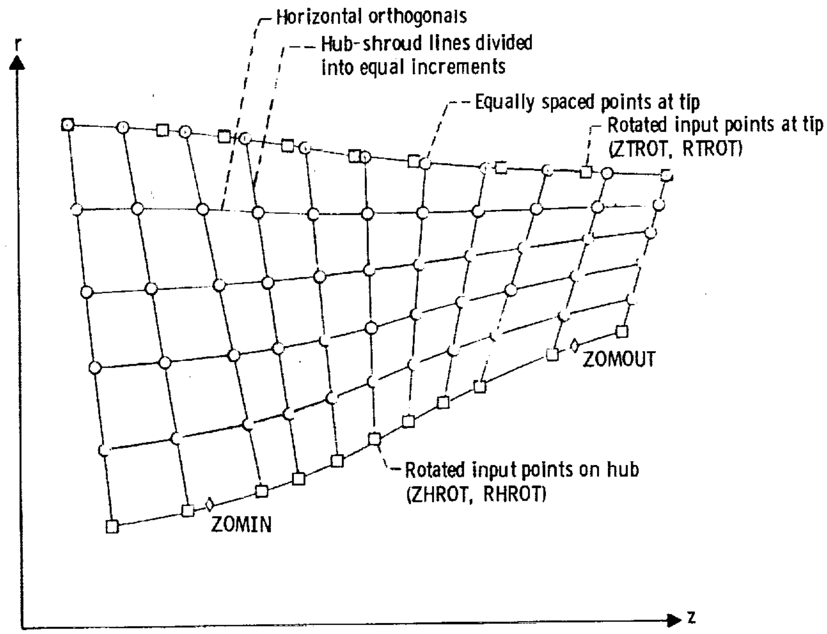


Figure 7. - "Horizontal" orthogonals obtained by spline curve fitting.

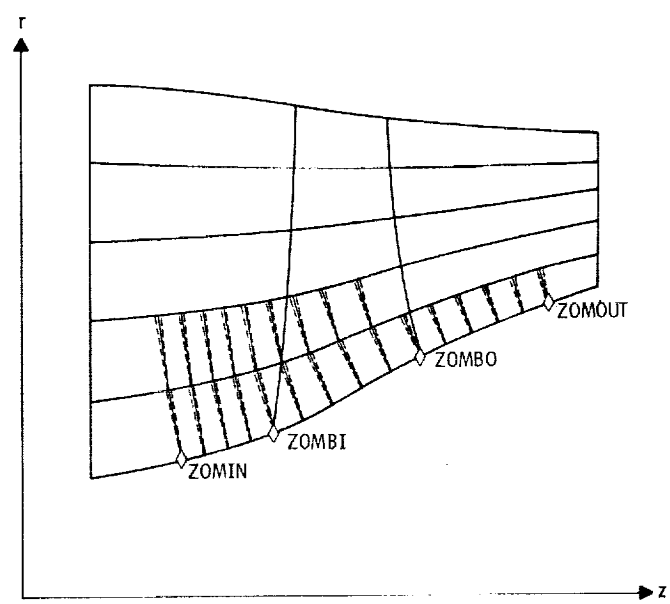


Figure 8. - Process for generating "vertical" orthogonal links.

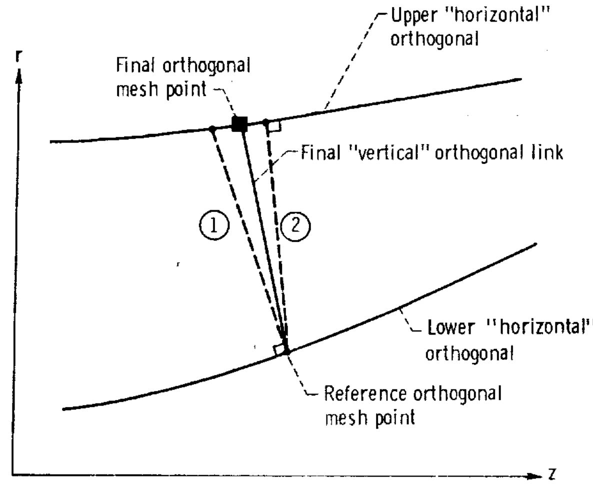


Figure 9. - Calculation procedure for a "vertical" orthogonal link.

a way that they are perpendicular to the tangents to the upper orthogonal at the intersection points and pass through the original starting points on the lower orthogonal. The rotated z-coordinates of the intersections of both sets of lines, ① and ②, are now known on the upper orthogonal. The desired new orthogonal mesh point z-coordinates (ZOMROT) are the average of these two sets of z-coordinates. The corresponding rotated r-coordinates (ROMROT) and slopes are then calculated by a SPLINT call. Mesh angles (PHI) can now be obtained.

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. Finally, the unrotated mesh coordinates (ZOM, ROM) are calculated from ZOMROT, ROMROT by a call to ROTATE.

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

Streamwise distance between vertical orthogonals 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. 8). The number of horizontal orthogonals is MHT + 1, which is the same in all three regions.

Subroutine PRECAL

Subroutine PRECAL calculates many of the fixed constants that will be needed by the subroutines in the outer iterative loop of MERIDL. Figure 10 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, and RHOIPF, 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 is called.

In the cases where output streamline values (FLFR array) were not read in (NSL = 0), PRECAL assigns 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. Straight-line distances between adjacent points are used in this calculation of s and t, because the correction between arc length and chord length is not significant for adjacent points.

If input hub and shroud station-line arrays were given (NOSTAT > 0), these arrays are then put into the rotated reference frame with ROTATE calls. Rotated blade geometry arrays (ZBLROT, RBLROT) are likewise calculated from the input arrays (ZBL, RBL).

If there is no blade row in the solution region (MBI = 0), PRECAL then stores dummy values into the ILE and ITE arrays. A large section of code that pertains only to solutions with blades present is then skipped.

In the case where blades are present, the rotated z- and r-coordinate arrays that define the leading and trailing edges of the blades (ZLE, RLE and ZTE, RTE) are then obtained. These are the first and last values for each blade plane from the ZBLROT and RBLROT blade-coordinate 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 then the trailing, edge of the blade. With INRSCT calls, the rotated 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 that lie just within the blade leading and trailing edges are then calculated by comparing the rotated z-coordinates of mesh points along the orthogonals with the rotated z-coordinates of intersections of the horizontal mesh lines with the blade edges. The s-coordinates are then calculated for the points where the horizontal mesh lines cross the blade edges.

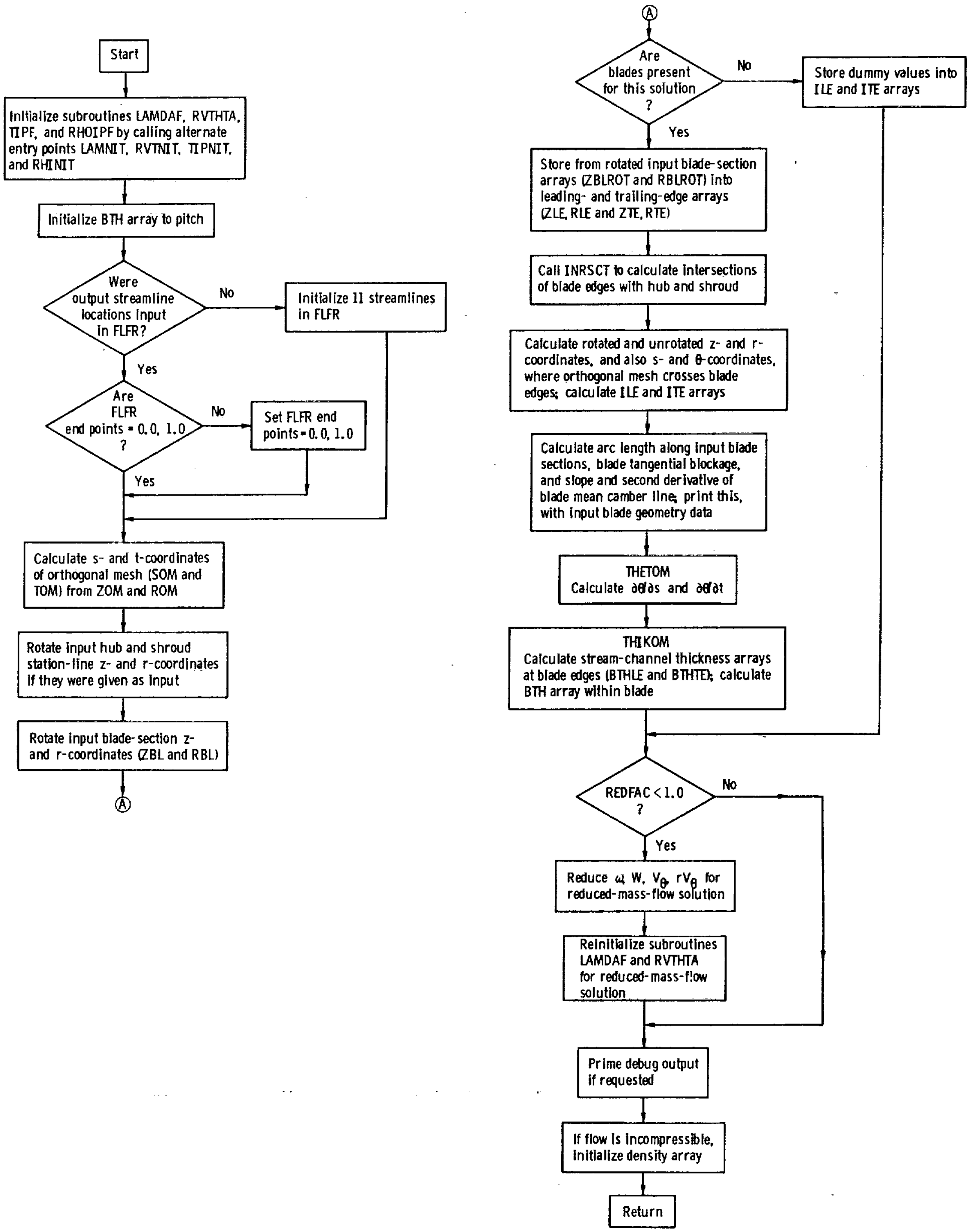


Figure 10. - Flow chart for PRECAL.

Theta coordinates are then calculated at these same points by means of SPLINT calls. Finally, ROTATE is used to calculate unrotated z- and r-coordinates of intersections of horizontal mesh lines with the blade edges.

Next, arc lengths along the input blade-section mean camber lines and blade blockage are calculated. SPLINE or SPLISL calls are then used to calculate the slope and the second derivative of the mean camber line θ-coordinate as a function of arc length, on the same input sections. The input blade geometry, blockage, arc length, and first and second derivatives are printed in the output.

At this point, PRECAL calls two other subroutines, THETOM and THIKOM. The THETOM routine calculates $\partial\theta/\partial s$ and $\partial\theta/\partial t$ at the orthogonal mesh points. The THIKOM routine calculates the stream-channel thickness arrays at the blade edges (BTHLE and BTHTE) and makes corrections to the BTH array to account for blade thickness.

Then PRECAL reduces certain parameters for the case where a reduced-mass-flow solution will have to be obtained (REDFAC $<$ 1.0). Wheel speed (OMEGA) and mass flow (MSFL) are reduced by REDFAC, as well as whirl (LAMIN, LAMOUT) and tangential velocity (VTHIN, VTHOUT). Subroutines LAMDAF and RVTHTA are then reinitialized by LAMNIT and RVTNIT calls.

Finally, PRECAL prints several arrays of debug information, if they are called for. Also, if the flow is incompressible (GAM = 0.), the density array is set to the input density given in the variable AR.

Subroutine THETOM

Subroutine THETOM calculates the gradients $\partial\theta/\partial s$ and $\partial\theta/\partial t$ at the orthogonal mesh points that 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 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 first calculates the stream-channel thickness arrays BTHLE and BTHTE at the points where the orthogonal mesh lines cross the leading and trailing edges of the blades. The tangential blade thickness TTBL is known at the blade edges where they are crossed by the input blade sections. SPLINT calls are used to interpolate and obtain this thickness where the blade edges are intersected by the horizontal orthogonal mesh lines. These thicknesses are subtracted from the pitch to obtain BTHLE and BTHTE.

Then THIKOM interpolates with LININT on the alternate mesh array TTPC of tangential blade thickness to obtain blade thickness at the points of the rotated orthogonal mesh, ZOMROT and ROMROT. A correction is then made to the BTH array at each mesh point by subtracting this blade thickness.

Subroutine INIT

Subroutine INIT 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 $\rho = \rho''$ throughout the passage. All other values are set to zero, except for W_s, W_z, and $\cos(\alpha - \varphi)$, which are set to values that 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_0 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_0 depends on $\partial(rV_\theta)/\partial t$. This gradient tends to be unstable with iteration, so that damping is usually 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 $\partial(rV_\theta)/\partial t$ and the maximum predicted change in $\partial(rV_\theta)/\partial t$ are calculated and printed. When it is indicated by the value of IDEBUG, the coefficients a_i and the constants k_0 will be printed.

Subroutine SOR

Subroutine SOR solves the finite-difference equations (A5) by the method of overrelaxation (ref. 8). 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 9. At each interior point, u_0^{m+1} is calculated from the values of u at the neighboring points by

$$u_0^{m+1} = \sum_{i=1}^{4} a_i u_i$$

where each u_i is the most recently calculated value for the point. To start, $u_0^0 = 1$ at the interior points and $u_0^0 = 0$ along the hub and shroud. The maximum (LMAX) and minimum (LMIN) values over all the interior mesh points of the ratio u_0^{m+1}/u_0^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 = $2/\left(1 + \sqrt{1 - \text{LMAX}}\right)$. The theory for calculating ORF is derived in reference 8.

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

$$u_0^{m+1} = u_0^m + \Omega\left(\sum_{i=1}^{4} a_i u_i + k_0 - u_0^m\right)$$

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 LOSSOM interpolates the total pressure loss at the downstream input station and then distributes this loss on the orthogonal mesh as specified by the input. The loss is stored in the PLOSS array at each orthogonal mesh point. The loss is assumed to be zero for incompressible flow. A flow chart for LOSSOM is given in figure 11.

LOSSOM begins by making reinitialization calls for LAMDAF, RVTHTA, TIPF, and RHOIPF on each iteration if whirl is not given as a function of the stream function. The reinitialization is not needed for the LOSSTV entry point, which is used only for the final transonic velocity-gradient solution. Also, only one vertical mesh line (IM = II) is calculated at a time from the LOSSTV entry point.

The loss is then calculated as 1.0 minus the ratio of actual to ideal relative total pressure,

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

In one input option, loss is given directly; in the other option, p_i', T_i', λ, $(rV_\theta)_o$, and p_o' are given and the loss is calculated from Euler's equation by using the relations

$$T_o' = T_i' - \frac{\omega\left[\lambda - (rV_\theta)_o\right]}{c_p} \tag{1}$$

and

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

If the loss is calculated, it is then printed; and if a negative loss is calculated, a warning message is printed.

At this point, SPLINT is called to calculate the spline-fit curve for full downstream loss as a function of stream function from hub to shroud. Then SPLINT is called through the SPLENT entry point to get the full downstream loss corresponding to the stream function for each orthogonal mesh point.

The actual loss to be applied at each mesh point (a percentage of the full down-

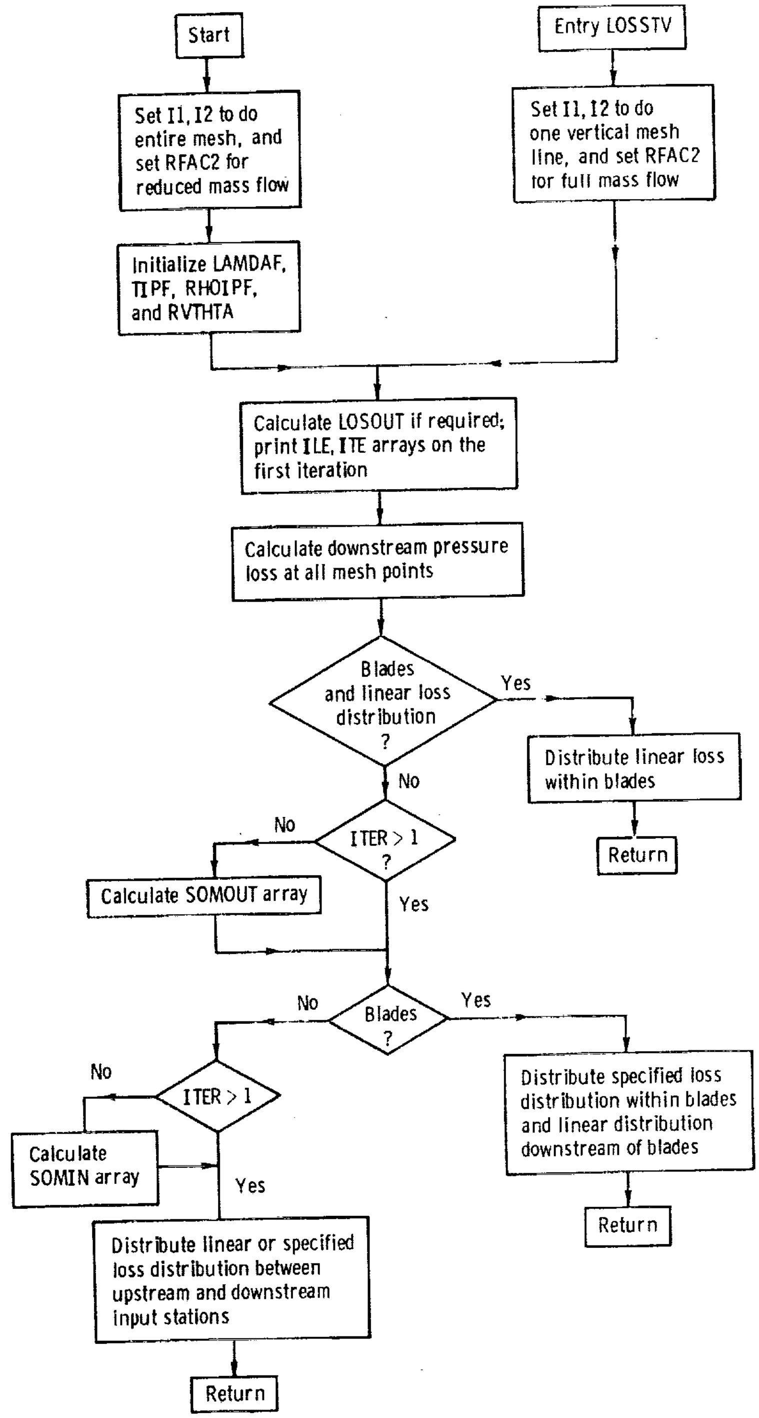


Figure 11. - Flow chart for LOSSOM.

stream value) is calculated in one of several ways. The most common situation occurs when blades are present and there is a linear streamwise loss distribution (NLOSS = 0). In this case, loss is distributed linearly within the blades from zero loss at the leading edge to full loss at the trailing edge. Full loss is also used from the trailing edge to the downstream boundary.

In other situations where there are no blades or a streamwise loss distribution is given as input, or both, some additional arrays are calculated on the first iteration. In these cases, the s-distances (SOMOUT array) from the upstream boundary to the downstream input station along each horizontal mesh line are calculated. If there are no blades, the s-distances (SOMIN array) from the upstream boundary to the upstream input station are also calculated.

For the case where blades are present and a streamwise distribution of loss within the blade row is given as input, the loss is distributed within the blades according to the values in the input PERLOS array. A linear distribution of loss is applied downstream of the blade from the final value of PERLOS at the trailing edge to full loss at the downstream input station. Full loss is used from there to the downstream solution boundary.

In the case where no blades are present, loss is distributed between the upstream and downstream input stations, either linearly or according to specified input distribution in the PERLOS array. Full loss is then used from the downstream input station to the downstream solution boundary.

Subroutine NEWRHO

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

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 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 W_s, W_θ, V_θ, W, ρ, ξ, and ζ at every mesh point. The first inner loop stores values of t-distances and the stream function u in temporary arrays. Then SPLINE is called to calculate $\partial u/\partial t$. The second inner loop performs

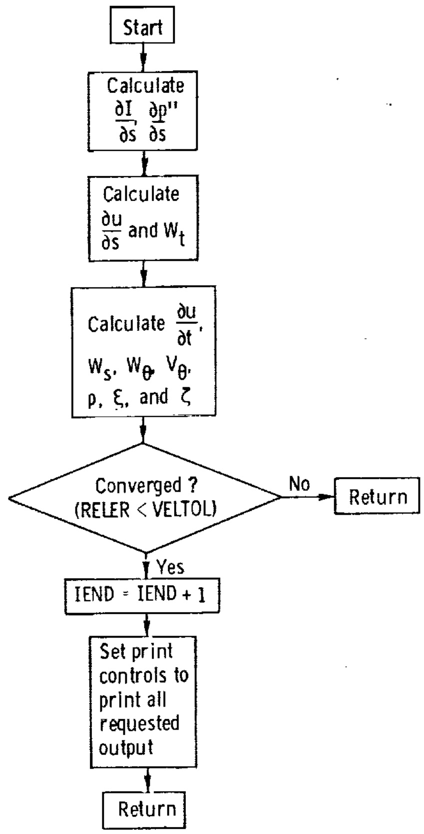


Figure 12. - Flow chart for NEWRHO.

further calculations. Equation (G10) of part I is used to calculate W_s. Within the blade row, W_θ is 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 row. Outside the blade row,

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

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 relative stagnation pressure p'' is calculated by

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

where

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

Equation (3) is the same as equation (D5) of part I with $W = 0$. The density ρ is calculated by

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

where T/T_i' is calculated by equation (D5) of part I. This completes the second inner loop to statement 30. Then SLOPES is called to calculate $\partial T''/\partial t$ and $\partial p''/\partial t$. This gives all the quantities necessary for the final inner loop to calculate ξ and ζ from equations (A2) and (A3) of part I.

After all calculations are done, the maximum and average 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 any 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.

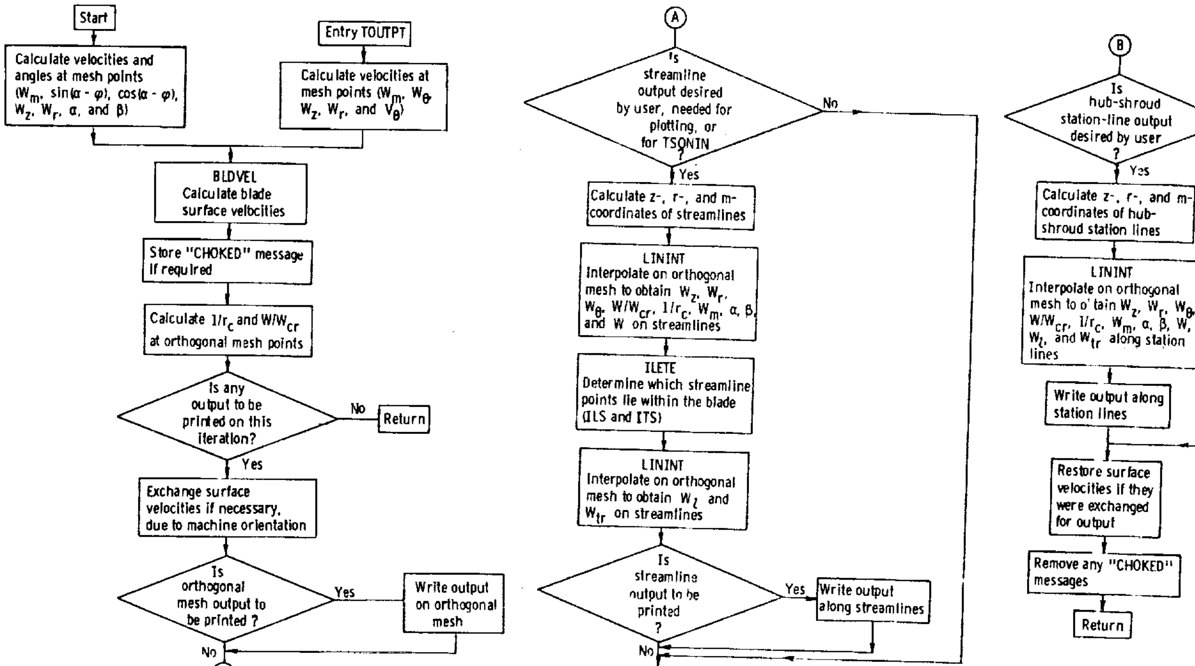


Figure 13. - Flow chart for OUTPUT.

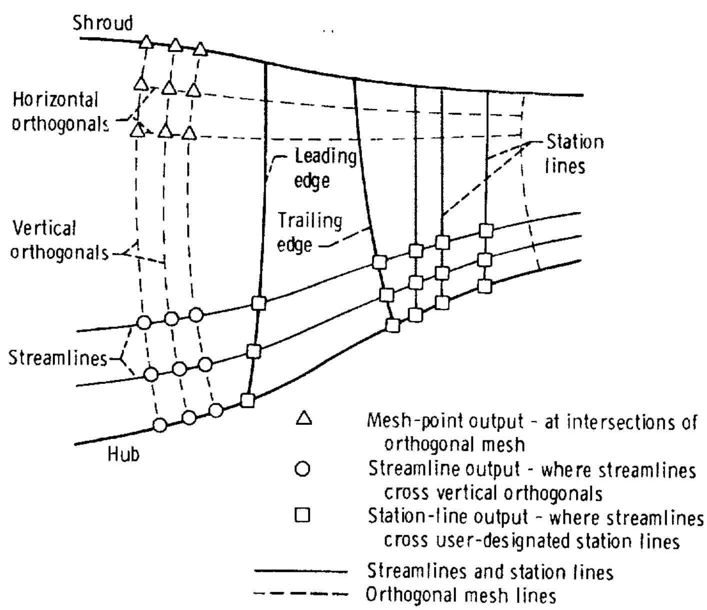


Figure 14. - Location of three major types of output.

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:

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

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

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

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

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

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

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

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 somewhat 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:

$$W_m = W \cos \beta$$

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

$$W_z = W_m \cos \alpha$$

$$W_r = W_m \sin \alpha$$

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

Subroutine BLDVEL is then called to calculate estimated blade surface velocities. If there are any choked vertical mesh lines in the transonic solution, the "choked" message is stored where required.

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 the critical velocity ratio is obtained from

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

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

If no output is to be printed, no further calculations are made by OUTPUT.

Now, 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 upstream mesh boundary 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 the upstream mesh boundary as the reference for $m = 0$.

For a station line on the leading or trailing edge, free-stream values are extrapolated along mesh lines to the leading or trailing edge and then interpolated along the leading or trailing edge at the specified output streamlines. The quantities W_z, W_r, and β are extrapolated and interpolated in this manner, and thus W_m and W_θ are calculated. On the other hand for a station line that is not on the leading or trailing edge, interpolations from the orthogonal mesh are made by LININT to obtain W_z, W_r, and W_θ, and then W_m and β are calculated. For all station lines, the meridional streamline curvature and the fractional total pressure loss are then interpolated from the orthogonal mesh by LININT. Now α and W are calculated by using the equations given previously. LININT is then called to interpolate W_l and W_{tr} for station lines that lie within the blade. Finally, the remaining station-line output V_θ, V, β_{abs}, T'', W/W_{cr}, p'', T', p', T, ρ, and p is calculated at each point. The station-line output is then printed.

The final small section of OUTPUT then restores W_l and W_{tr} to the proper arrays if they were interchanged to correspond to suction and pressure surfaces for printout, and any "choked" messages are removed.

Subroutine BLDVEL

Subroutine BLDVEL calculates blade surface velocities and densities and F_t. First, $\partial(rV_\theta)/\partial t$ and $\partial(rV_\theta)/\partial s$ are calculated by using the SLOPES subroutine. Then, $\left[d(rV_\theta)/dm\right] B \cos\beta$ 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 (D4) and (D5) 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_t is calculated by

$$F_t = \frac{W}{B} \frac{\partial \theta}{\partial t} \text{ DFDM} \tag{4}$$

where

$$\text{DFDM} = -B \cos \beta \frac{d(rV_\theta)}{dm}$$

Equation (4) is obtained from equations (B25) and (G2) of part I. The new value for F_t is calculated from the old F_t and the predicted value of F_t 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_t and the maximum change in F_t are calculated and printed. If debug output is requested, the arrays that 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 contain 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.

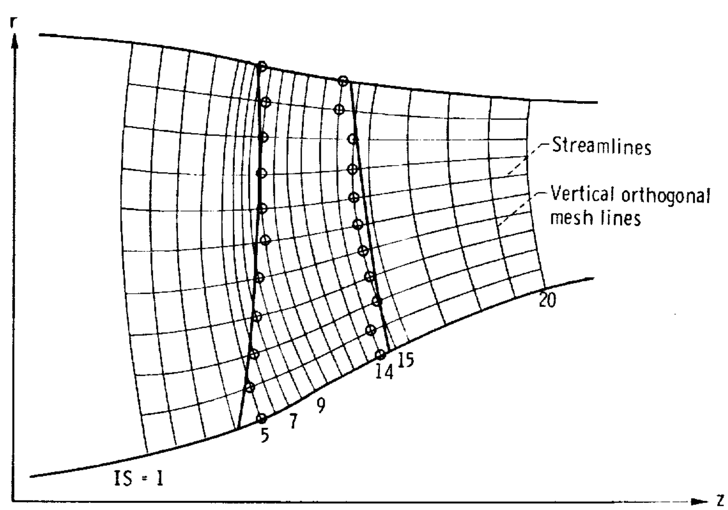


Figure 15. - Location of ILS, ITS points by ILETE.

Subroutine INDEV recalculates $\partial\theta/\partial s$ to allow for incidence and deviation. This means that the midchannel 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.

First, the blade mean camber angle $\beta_{b,le}$ at the leading edge is calculated. Then the flow angle corrected for blockage at the leading edge β_{bf} is calculated from equation (F1) of part I. The corrections to $\partial\theta/\partial s$ are made so that the difference between β_{bf} and β_b varies linearly from the blade leading edge to the distance specified in appendix F of part I. This distance is DISTLE. The interpolation to calculate β_{bf} (BETAIJ) at each orthogonal mesh point near the leading edge is done next, followed by the calculation of $(\partial\theta/\partial s)_{bf}$ (DTHDS(I, J)) from equation (F2) of part I. The calculation of blocked and unblocked incidence angles completes the leading-edge calculation. The trailing-edge deviation calculation is done in the same manner as the incidence calculation. Finally, the incidence and deviation angles are printed if there was any output requested for the current iteration.

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

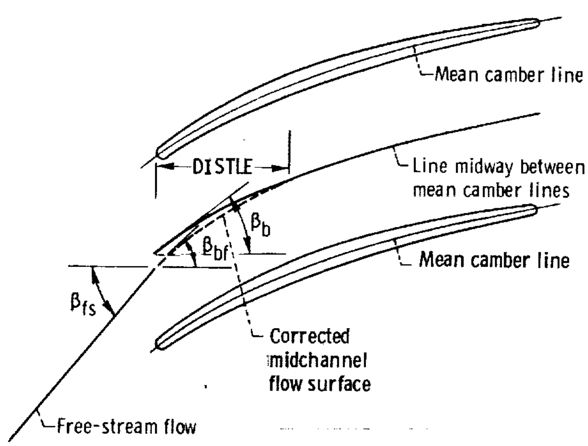


Figure 16. - Corrected midchannel flow surface. The corrected midchannel flow surface is used to calculate $(\partial\theta/\partial s)_{bf}$. Incidence = $\beta_{bf} - \beta_b$.

Subroutine TSONIN generates and prints the data required as input to the TSONIC blade-to-blade analysis program (ref. 3). 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. 3). The output generated in TSONIN can in general be directly submitted to the TSONIC program. However, the output should be inspected before doing so, because slight changes are sometimes required, depending upon how the user wishes to run the TSONIC program. These changes are described in part I.

Along each output streamline, TSONIN obtains the upstream and downstream flow conditions T_i', ρ_i', λ, and $(rV_\theta)_0$ with calls to TIPF, RHOIPF, LAMDAF, and RVTHTA. LININT calls are then used to obtain all the variables required to calculate blade-to-blade streamsheet thickness b, as well as loss distribution along the streamsheet. The thickness b is obtained from

$$(\rho W_m)_{av} = \rho_{av} W_m + \frac{\rho_l - \rho_{tr}}{12} \cos \beta (W_l - W_{tr})$$

which is derived from equation (G9) of part I, and

$$b = \frac{w}{(\rho W_m)_{av} rB}$$

Then TSONIN calculates the blade surface geometry on blade-to-blade stream surfaces by a method described in reference 10. This process is complicated by the fact that leading- and trailing-edge radii are not used by MERIDL and have to be generated by TSONIN within the blade surface envelope. The origin for θ-coordinates for TSONIC is at the center of the leading-edge radius. Since the leading-edge radius is not known at the outset, θ-coordinates are initially calculated from the intersection of the mean camber line with the leading edge (appendix E). After the leading-edge radius has been determined, $\Delta\theta$, the difference in θ from the intersection of the mean camber line with the leading edge to the center of the leading-edge radius, is calculated and subtracted from all calculated blade surface θ-coordinates. The technique used to generate the blade leading- and trailing-edge radii and calculate $\Delta\theta$ are described in appendix E.

Subroutine TSONIN calculates the blade surface coordinates for each point where the meridional streamline is intersected by a vertical orthogonal mesh line, as explained in appendix E. If the blade envelope has no thickness at the leading or trailing edge, TSONIN gives it a leading-edge diameter equal to one-tenth of 1 percent of meridional chord. Any surface points too close to the leading- or trailing-edge points are then omitted from the set of surface coordinates.

Then TSONIN calculates leading- and trailing-edge radii within the surface envelope as described in appendix E. The points of tangency of the radii with blade surfaces, and the tangency angles, are also obtained. The tangency points are then made the first and last points on each of the surfaces, and points outside of these or too close to these are excluded. All θ-coordinates are then shifted to TSONIC section origin (see appendix E). Finally, TSONIN calculates r-coordinates for each surface point, surface slopes, second derivatives, and curvatures and prints this information for both blade surfaces. This process is repeated for each streamline.

Subroutine TVELCY

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

After statement 55, variables are initialized for the main loop on vertical mesh lines. To start, I = 0 and INCR = 1.

Statement 60 is the beginning of the main loop that ends at statement 290. The main loop starts at the upstream boundary and solves the velocity-gradient equation for each vertical mesh line. If there are blades, the procedure is to move downstream to each of the vertical mesh lines in sequence until the blade leading edge is reached. At this time, LINDV is called to make incidence corrections to β for a short distance beyond the leading edge, as described in appendix F of part I. After all leading-edge corrections to β are made, there is a jump to the downstream boundary. Then the procedure is to move upstream to the blade trailing edge, at which time TINDV is called to make deviation corrections to β. The program then proceeds upstream until a solution has been obtained for every vertical mesh line.

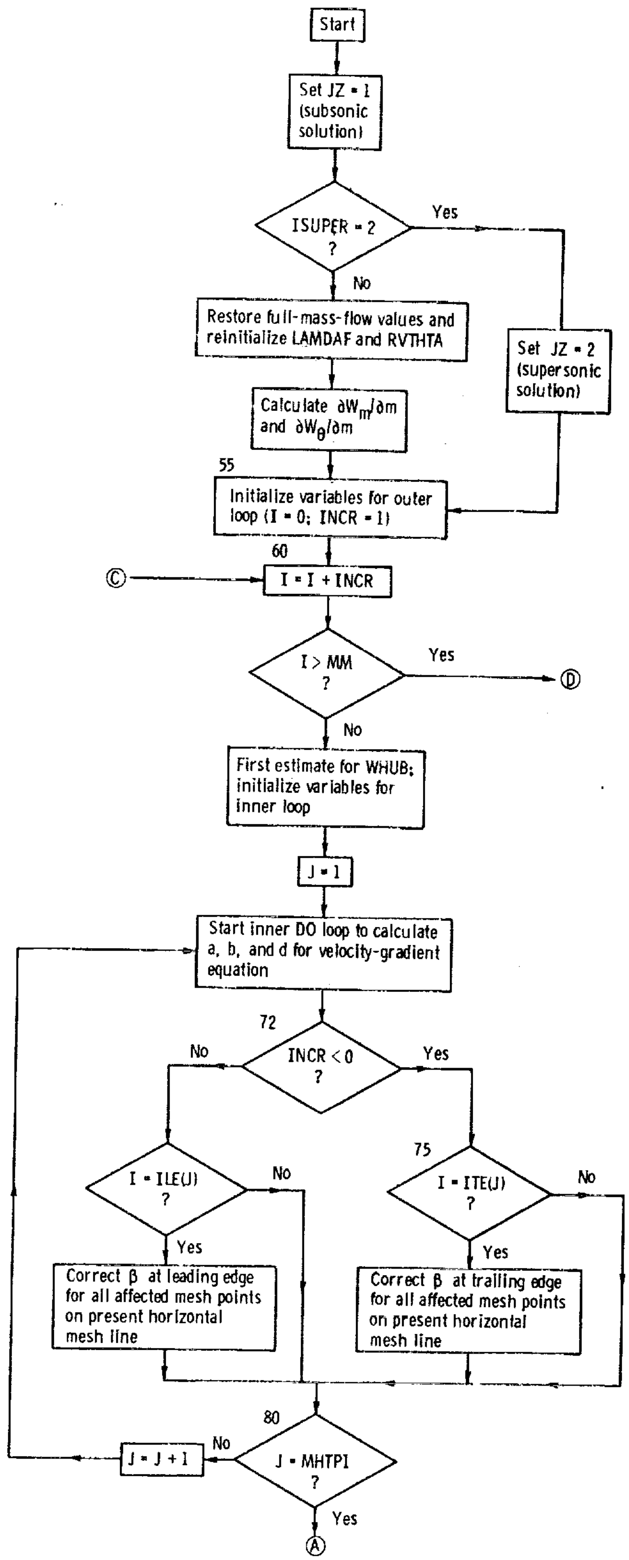


Figure 17. - Flow chart for TVELCY.

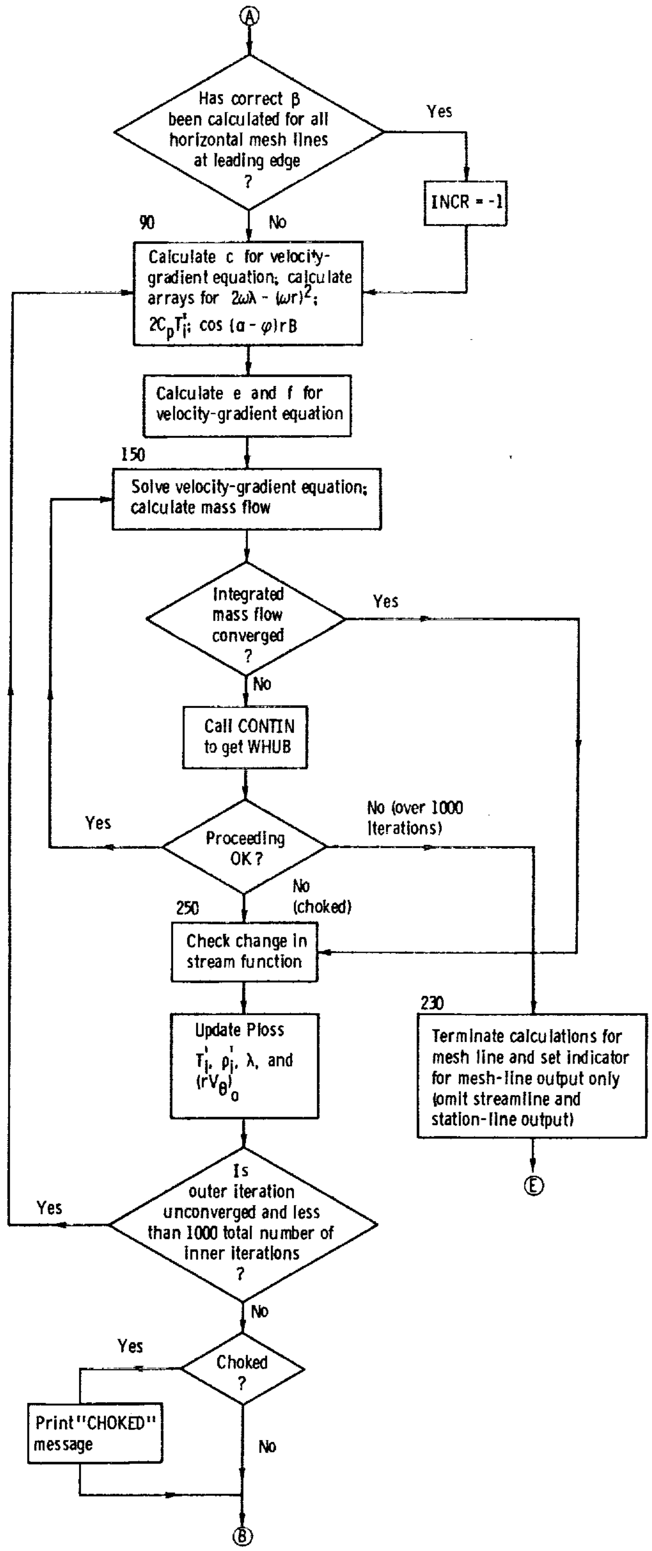


Figure 17. - Continued.

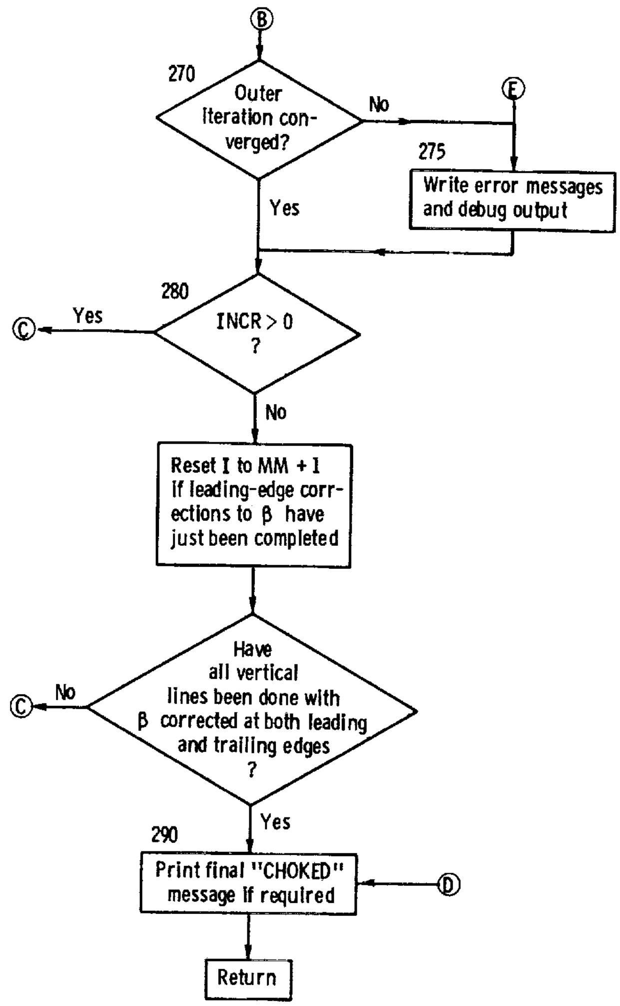


Figure 17. - Concluded.

At statement 60, INCR is added to I to determine the next vertical mesh line. INCR is 1 at the start. After all incidence corrections to β are made, INCR is changed to -1. Then the solution will be found at the downstream boundary (I = MM) and I will decrease.

The initial estimate of W on the hub (WHUB) is set equal to the reduced-mass-flow value for W divided by REDFAC. The first inner DO loop to statement 80 calculates coefficients a, b, and d for the velocity-gradient equation (A7) of part I. These coefficients are calculated by equations (A8) to (A10) of part I. The initial arrays for whirl, temperature, and density are calculated at the same time. In this same loop a check is made to see if LINDV or TINDV should be called to make incidence or deviation corrections to β. After the DO loop to statement 80, INCR will be set to -1 if all the incidence corrections have been made.

The outer iteration for a given vertical mesh line begins at statement 90. The first inner loop here calculates coefficient c for the velocity-gradient equation from equation (A9) or (A10) of part I, as well as $2\omega\lambda - (\omega r)^2$, $2c_pT_i'$, and $\cos(\alpha - \varphi)rB$ at each mesh point. The next DO loop to statement 130 calculates coefficients e and f from equation (A11) of part I.

At statement 140, IND is set to 1 to indicate the beginning of the inner iteration procedure. Each inner iteration then begins at statement 150. 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 second 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 (ref. 7). The equations used in the Heun method for this case are

$$\left.\begin{aligned} W_{j+1}^{*} &= W_j + (dW)_j && \text{first estimate for } W_{j+1} \\ W_{j+1}^{**} &= W_j + (dW)_{j+1}^{*} && \text{second estimate for } W_{j+1} \\ W_{j+1} &= \frac{W_{j+1}^{*} + W_{j+1}^{**}}{2} && \text{average of two estimates for } W_{j+1} \end{aligned}\right\} \qquad (5)$$

where $(dW)_j$ (eq. (A7) of part I) is evaluated at the j^{th} mesh point from the hub with $W = W_j$ and where $(dW)_{j+1}^{*}$ is evaluated at the $j + 1$ mesh point with $W = W_{j+1}^{*}$. At the same time that the solution of the velocity-gradient equation is being calculated, the mass-flow integration is also being calculated by trapezoidal integration of

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

The inner iteration ends when the velocity-gradient solution gives the correct mass flow in this equation (or if the choking mass flow is less than the input mass flow). If the correct mass flow is not obtained in 100 iterations, an error message and debug information are printed, and the program goes on to the next vertical line.

After the end of the inner iteration, at statement 250, the new stream-function values are compared with the previous outer iteration; if there is a change of more than 0.01 percent, the inner iteration will be repeated (set REPEAT = . TRUE.). Then the

PLOSS array is updated by calling LOSSTV, and arrays of T_i', ρ_i', λ, and $(rV_\theta)_0$ are all adjusted to new values. At this point there will be another outer iteration if the solution has not converged and there are less than 1000 total iterations. If there are over 1000 total iterations for any vertical mesh line, the calculation for that mesh line is terminated. After the termination of the outer iteration, error messages are printed if there is choking or if a converged solution cannot be found. If INCR = 1, the program moves downstream to the next vertical line. At the appropriate point the procedure shifts to the downstream boundary and moves upstream until all vertical mesh lines have had a solution. This may involve redoing some vertical mesh lines, since the deviation region could extend to a vertical mesh line that crossed the incidence region.

After all mesh lines have been solved, a final choking message is printed if any vertical mesh line was choked. Control is then returned to the main program.

Subroutine LINDV

Subroutine LINDV recalculates β to allow for incidence and deviation in a manner similar to INDEV. LINDV is called only for the velocity-gradient solution, so that corrections are made to β instead of to $\partial\theta/\partial s$. Also a density correction is made to satisfy flow continuity at the blade leading and trailing edges (appendix F). Otherwise, the calculation is similar to that in INDEV. The first part of the subroutine does the incidence calculation only. The deviation calculation is done from the TINDV entry point. The final entry point is PINDV and is used only for printing previously calculated incidence and deviation angles.

Functions LAMDAF, RVTHTA, TIPF, and RHOIPF

These four 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 IV. 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 (G13) is used instead of the quadratic formula. The reason for this is explained in appendix G.

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 that 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 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

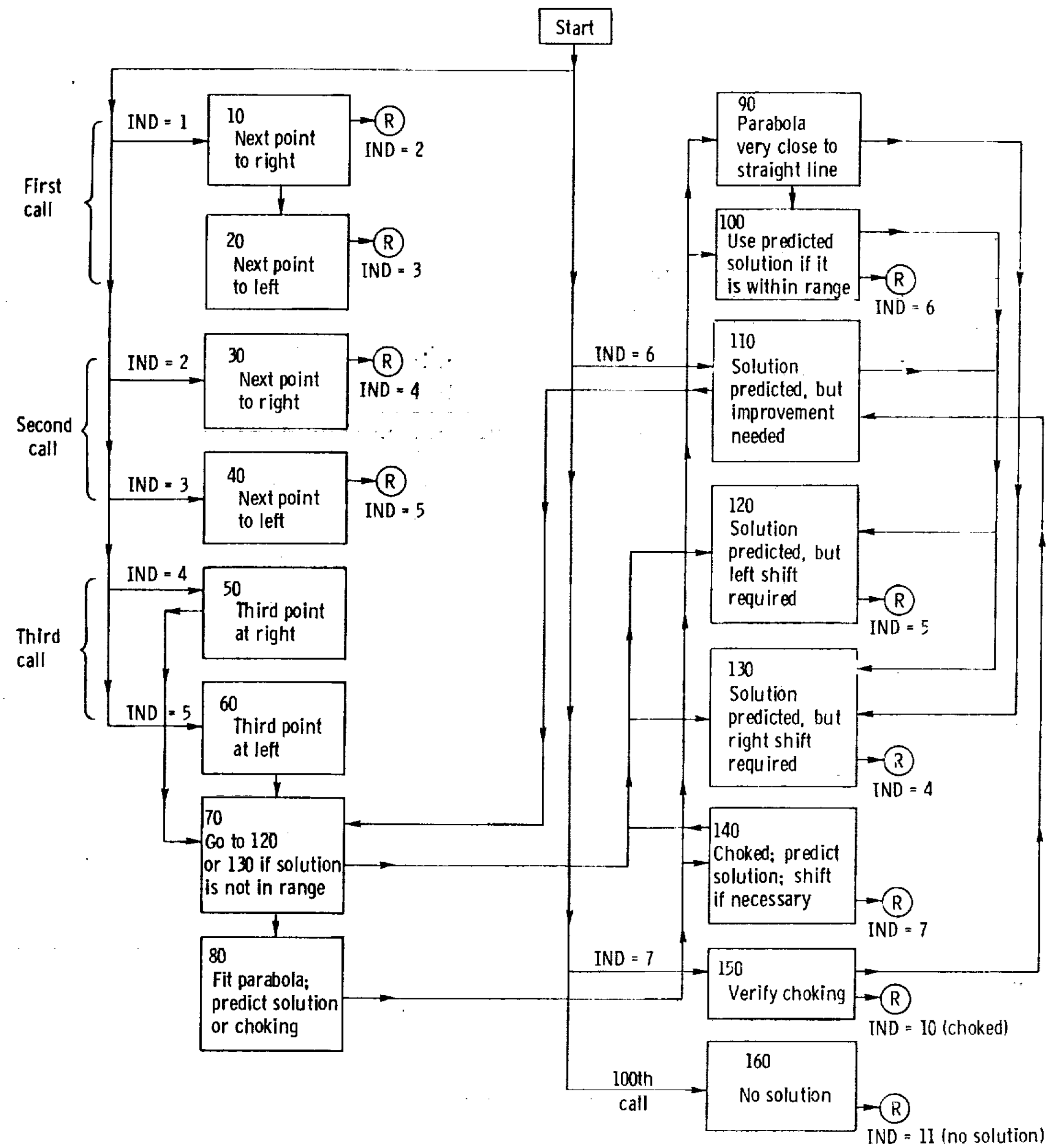


Figure 18. - Flow chart for CONTIN. R = Return.

TABLE IV. - SIGNIFICANCE OF IND IN VARIOUS

CALLS TO CONTIN

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

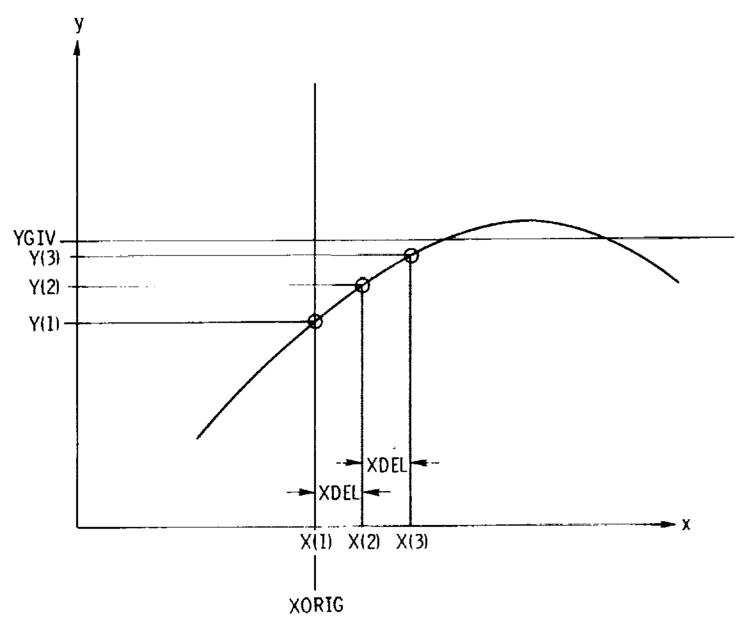


Figure 19. - Starting procedure for CONTIN.

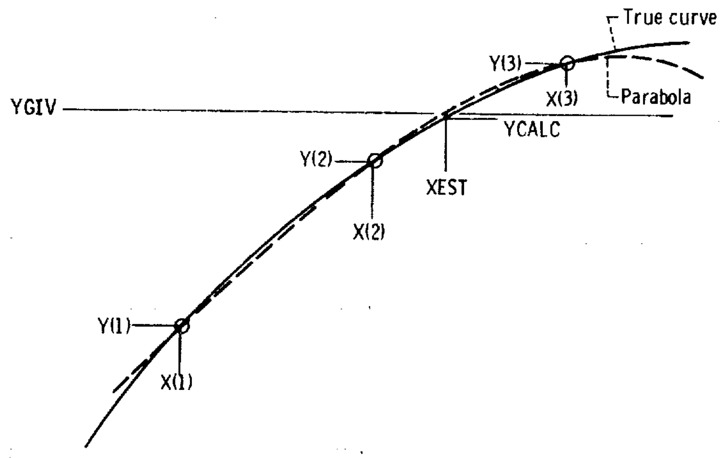


Figure 20. - Approximating curve with a parabola.

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) ≤ YGIV ≤ 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 determined 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 IV. 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 IV)

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 that 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 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

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

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

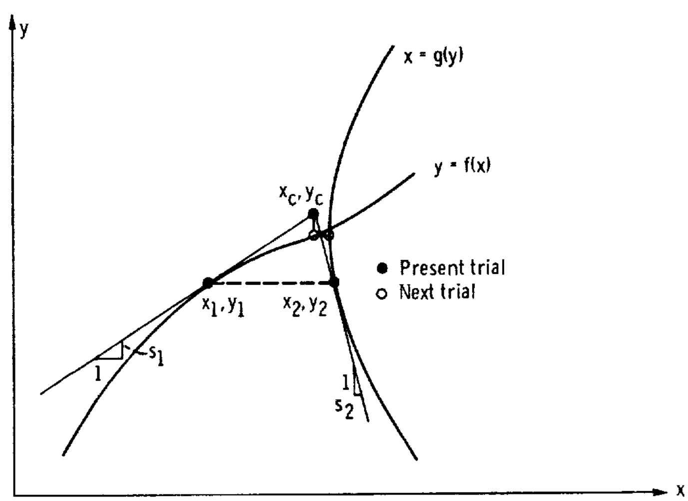


Figure 21. - Procedure for calculating intersections in INRSCT.

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 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.

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 vertically) and the other set running more or less vertically (never horizontally). 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 to the top). The lines between mesh points are assumed to be straight lines.

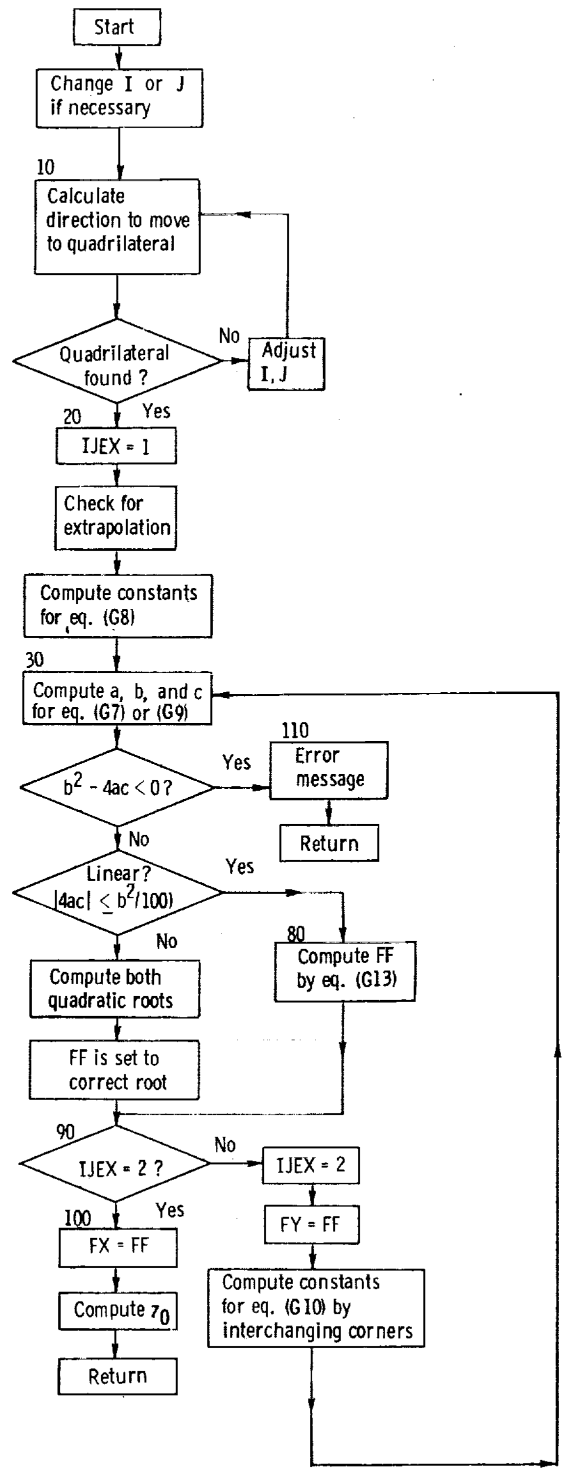


Figure 22. - Flow chart for LININT.

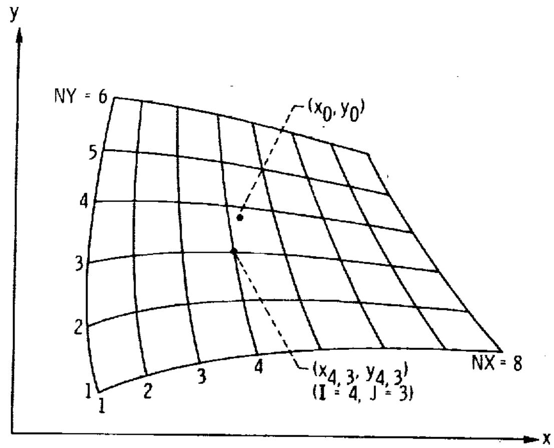


Figure 23. - Typical mesh for LININT.

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 and 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 that the point is below the bottom line; ABOVE = 0, that the point is between the bottom and top lines; and ABOVE = 1, that the point is above the top line. Similarly, RIGHT = -1 indicates that the point is to the left of the left line; RIGHT = 0, that the point is between the left and right lines; and RIGHT = 1, that 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 is described in appendix G. First, the quadratic coefficients are calculated by equation (G8) or (G10). Then, the quadratic equation (G7) or (G9) is solved either by the quadratic formula or by the binomial expansion, equation (G13), as explained in appendix G.

The same coding is used to calculate both f_x and f_y. After these values are obtained, equation (G14) is used to calculate the interpolated value of z_0.

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 x-direction
NY	number of mesh points in y-direction
NDIMX	dimension of X-, Y-, and Z-arrays in x-direction
NDIMY	dimension of X-, Y-, and Z-arrays in 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 left of (X0, Y0)
J	number of horizontal mesh line below (X0, Y0)

The internal variables for LININT are

ABOVE	integer, 1 indicates that (X0, Y0) is above the current I, J region, 0 within, and -1 below
ACB2	ac/b^2 (eq. (G13))
CASE	used to indicate whether F1 or F2 is the proper solution
DISCR	discriminant, $b^2 - 4ac$ (eq. (G7) or (G9))
EXTRAP	array to indicate extrapolation either horizontally or vertically
FA	$-b/2a$ (eq. (G7) or (G9))
FB	$\sqrt{(b^2 - 4ac)}/2a$ (eq. (G7) or (G9))
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. (G8) or (G10))
QB	b (eq. (G8) or (G10))
QC	c (eq. (G8) or (G10))
RIGHT	integer, 1 indicates that X0, Y0 is to the right of the current I, J region, 0 within, and -1 left
X01	x_{01} (see appendix G 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 ROTATE

Subroutine ROTATE is a general-purpose subroutine to rotate coordinates of one- or two-dimensional arrays of x- and y-coordinates. The rotated coordinates calculated by ROTATE may be placed in the original input arrays, or they may be placed in new arrays.

The calling statement for ROTATE is

```
CALL ROTATE(ANGROT, X, Y, NX, NY, NDIMX, NDIMY, XROT, YROT)
```

The input variables for ROTATE are

ANGROT angle of rotation, rad

X one- or two-dimensional array of x-coordinates

Y one- or two-dimensional array of y-coordinates

NX number of points to be rotated for a one-dimensional array; number of points denoted by first subscript for a two-dimensional array

NY number of points denoted by second subscript for a two-dimensional array (NY = 1 for a one-dimensional array)

NDIMX dimension for first subscript of X, Y, XROT, and YROT arrays

NDIMY dimension for second subscript of X, Y, XROT, and YROT arrays

The output variables for ROTATE are

XROT one- or two-dimensional array of rotated x-coordinates

YROT one- or two-dimensional array of rotated y-coordinates

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 11 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 first and second derivatives. 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 SPLINT call.

The calling statement for SPLINT is

```
CALL SPLINT(X, Y, N, Z, MAX, YINT, DYDX, D2YDX2)
```

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
D2YDX2	array of interpolated second 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

Subroutines SPLISL and SPINSL

Subroutines SPLISL and SPINSL are the same as SPLINE and SPLINT, respectively, except that the end condition is a specified end-point slope. The input and output variables are the same, but with two added input variables, Y1P and YNP, which are the slopes at the first and last spline points.

Plotting Subroutines

There are four subroutines that do the plotting for MERIDL: INPLOT, MEPLOT, SLPLOT, and SVPLOT. In addition, another subroutine, PTBDRY, is called by MEPLOT to calculate hub and shroud, and leading- and trailing-edge boundaries. The plotting routines use the NASA Lewis in-house microfilm plotting package described in reference 12. These five routines are self-contained and can easily be removed from MERIDL without disturbing the rest of the calculations. On the other hand, if the user wants to obtain plots, he can code his own plotting routines by referring to the program listing which follows and consulting reference 12 to determine the functions of the various plotting calls.

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 RHOIPF, see table of contents) of the program. The remaining subroutines (CONTIN or SPINSL) 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 that 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 reduce 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.

MAIN DICTIONARY

Variable name	COMMON block	Subroutine	Description and comments
A(4, MM, MHTP1)	VARCOM		Coefficients of finite-difference equation (A7) for stream function, u
A0		COEF	a_0 (eq. (A8))
AAA(MM or MHTP1, etc.)		MESHO THIKOM NEWRHO OUTPUT TSONIN LAMDAF RVTHTA	Dummy array used in SPLINE and SPLINT calls
ALPHA(MM, MHTP1)	VARCOM		α at orthogonal mesh points, rad
ALPHLE		INDEV LINDV	α_{le}, rad
ALPHSP(MM)		TSONIN	α at TSONIC input points
ALPHTE		INDEV LINDV	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)		OUTPUT	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)		INPUT	Angles from meridional plane of blade-section mean camber lines at blade-section input points, rad
ANGR(NPPC, NBLPC)		THETOM	Angles with respect to radius of hub-shroud lines of alternate mesh (fig. 28), rad
ANGROT	INPUTT		Input angle of rotation, deg
ANGT1(NBLPL)		THETOM	Values from ANGZ array along constant-percent-chord line, rad
ANGT2(NBLPC)		THETOM	Values for ANGZ array along constant-percent-chord line, rad
ANGZ(NPPC, NBLPC)		THETOM	Angles with respect to z-axis of input blade sections (fig. 28), rad
AR	INPUTT		Input gas constant, R, J/(kg)(K)
ARTEM		TSONIN	Temporary AR, J/(kg)(K)
ATVEL(MHTP1)		TVELCY	Coefficients, a, of velocity-gradient equation ((A7), part I) at mesh points along vertical mesh lines
BBB(MM or MHTP1, etc.)		MESHO THIKOM OUTPUT TSONIN LAMDAF RVTHTA	Dummy array used in SPLINE and SPLINT calls
BEABST(NSL)		OUTPUT	β_{abs} at points where station lines cross streamlines, deg

Variable name	COMMON block	Subroutine	Description and comments
BESP(MM)		TSONIN	Input for TSONIC (ref. 3), m
BETA(MM, MHTP1)	VARCOM		β at orthogonal mesh points, rad
BETABF		INDEV LINDV	β_{bf} within blade, rad
BETAFS		INDEV LINDV	β_{fs} outside of blade, rad
BETAIJ		OUTPUT INDEV	β at orthogonal mesh point, rad
BETALE(NBLPL)	INPUTT		Input blade angles at leading edge, deg
BETATE(NBLPL)	INPUTT		Input blade angles at trailing edge, deg
BETI1		TSONIN	Input for TSONIC (ref. 3), deg
BETI2		TSONIN	Input for TSONIC (ref. 3), deg
BETO1		TSONIN	Input for TSONIC (ref. 3), deg
BETO2		TSONIN	Input for TSONIC (ref. 3), deg
BETSL(MM, NSL)	SLCOM		β at points along streamlines where they cross vertical mesh lines, rad
BETST(NSL)		OUTPUT	β at points along station lines where they cross streamlines, rad
BFACTR		TSONIN	Multiplying factor for BESP and ZMSFL
BLDCRD		INDEV LINDV	True blade chord along a horizontal mesh line, m
BLDEV(MHTP1)		LINDV	Deviation angle, corrected for blockage, where a horizontal mesh line intersects trailing edge, $(\beta_{bf} - \beta_b)_{te}$, deg
BLDEV		INDEV	Deviation angle, corrected for blockage, where a horizontal mesh line

Variable name	COMMON block	Subroutine	Description and comments
			intersects trailing edge, $(\beta_{bf} - \beta_b)_{te}$, deg
BLINC(MHTP1)		LINDV	Incidence angle, corrected for blockage, where a horizontal mesh line intersects leading edge, $(\beta_{bf} - \beta_b)_{le}$, deg
BLINC		INDEV	Incidence angle, corrected for blockage, where a horizontal mesh line intersects leading edge, $(\beta_{bf} - \beta_b)_{le}$, deg
BLNK		OUTPUT	Blank word used in some plot titles
BLOCK(NPPP)		PRECAL	Fractional blockage along input blade sections
BTBFLE(MHTP1)	INDCOM		β_{bf} where a horizontal mesh line intersects leading edge, rad
BTBFTE(MHTP1)	INDCOM		β_{bf} where a horizontal mesh line intersects trailing edge, rad
BTBLLE(MHTP1)		LINDV	β_b where a horizontal mesh line intersects leading edge, rad
BTBLTE(MHTP1)		LINDV	β_b where a horizontal mesh line intersects trailing edge, rad
BTFSEX(MHTP1)		OUTPUT	β_{fs} extrapolated to leading or trailing edge of blade, rad
BTH(MM, MHTP1)	CALCON		B at orthogonal mesh points, rad
BTHLE(MHTP1)	INDCOM		B_{le}, rad
BTHSL		TSONIN	B along a streamline, rad
BTHTE(MHTP1)	INDCOM		B_{te}, rad
BTVEL(MHTP1)		TVELCY	Coefficients, b, of velocity-gradient equation ((A7), part I) at mesh points along vertical mesh lines
C1		COEF	c_1 (eq. (A8))

Variable name	COMMON block	Subroutine	Description and comments
C2		COEF	c_2 (eq. (A8))
CAMP(MM, MHTP1)	VARCOM		$\cos(\alpha - \varphi)$ at orthogonal mesh points
CBETA		TVELCY	$\cos\ \beta$
CHANGE		SOR	Change in value of stream function at a mesh point during an over-relaxation iteration
CHFL		TVELCY	Choking mass flow for a vertical orthogonal mesh line, kg/sec
CHFRMS		TVELCY	Ratio of minimum choking mass flow to input mass flow
CHLIM		TVELCY	Minimum choking mass flow per passage, kg/sec
CHORD		TSONIN	Length of blade section along streamline (m-direction) (input to TSONIC, ref. 3), m
COSAB		THETOM	Cosine of ANGZ + ANGR
CP	CALCON		c_p, J/(kg)(K)
CPHI(MM, MHTP1)	CALCON		$\cos\ \varphi$ at orthogonal mesh points
CPTIP(MHTP1)		TVELCY	$2c_pT_i'$ along vertical mesh lines, (N)(m)/kg
CPTIP		LINDV	$2c_pT_i'$, (N)(m)/kg
CTVEL(MHTP1)		TVELCY	Coefficients, c, of velocity-gradient equation ((A7), part I) at mesh points along vertical mesh lines
CURV(MM, MHTP1)	VARCOM		$1/r_c$ at orthogonal mesh points, 1/m
CURV1(MM)		TSONIN	Curvature of upper blade surface at TSONIC input points, 1/m
CURV2(MM)		TSONIN	Curvature of lower blade surface at TSONIC input points, 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)		OUTPUT	$1/r_c$ at points along station lines where they cross streamlines, 1/m
D1		COEF	d_1 (eq. (A8))
D2		COEF	d_2 (eq. (A8))
D2BDM2(MM)		TSONIN	d^2B/dm^2, 1/m
D2TDM1(MM)		TSONIN	Second derivative of upper blade surface at TSONIC input points, rad/m^2
D2TDM2(MM)		TSONIN	Second derivative of lower blade surface at TSONIC input points, rad/m^2
D2YDX2(NPPP or MHTP1)		PRECAL	Second derivative of θ along input blade sections, rad/m^2
D2YDX2(NPPC or NBLPC)		THETOM	Dummy second derivative in SPLINT calls
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
DAMP		TSONIN	Damping factor on iteration for leading- or trailing-edge radii (appendix E)
DBDM(MM)		TSONIN	dB/dm
DBL		TSONIN	One-half of tangential blade thickness (in rad) at intersection of a streamline with blade leading or trailing edge

Variable name	COMMON block	Subroutine	Description and comments
DBTH		THIKOM	Tangential blade thickness at orthogonal mesh point, rad
DCHANG		COEF	Maximum change in estimated values of $\partial(rV_\theta)/\partial t$ at a mesh point between any two outer iterations, m/sec
DEGRAD		OUTPUT TSONIN INDEV LINDV	Conversion constant from radians to degrees
DELCH		OUTPUT	1 percent of average meridional chord length of blade, m
DELM		TSONIN	Increment of meridional distance, m
DELMAX		TVELCY	Maximum increment for W_{hub} at each iteration to satisfy continuity, m/sec
DELMSP		TSONIN	Minimum distance of blade surface points from leading or trailing edge (appendix E), m
DELR		MESHO THETOM 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
DELT		MESHO	Tangential blade thickness, m
DELTH		TSONIN	Shift in θ origin from MERIDL to TSONIC, rad
DELZ		MESHO THETOM OUTPUT	Increment in z-coordinate, m
DFDM(MM, MHTP1)	VARCOM		$-B \cos \beta \left[d(rV_\theta)/dm\right]$ at orthogonal mesh points (eq. (4)), m/sec

Variable name	COMMON block	Subroutine	Description and comments
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
DIP(NBLPL)		PRECAL	Distance up leading or trailing edge of blade, m
DISEOM(MHTP1)		THIKOM	Distance up leading or trailing edge of blade, m
DIST(NPPP)		INPUT	Distances on meridional plane along lines connecting input blade-section points (ZBL, RBL), m
DIST(NBLPL)		THIKOM	Distances on meridional plane along lines connecting input blade-section points (ZBL, RBL), m
DISTLE		INDEV LINDV	Distance along horizontal mesh line from leading edge of blade for which a blade shape correction is made for incidence, m
DISTTE		INDEV LINDV	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 $\partial(rV_\theta)/\partial t$ at any mesh point, m/sec

Variable name	COMMON block	Subroutine	Description and comments
DMIN		COEF	Minimum calculated value of $\partial(rV_\theta)/\partial t$ at any mesh point, m/sec
DNEW	INPUTT		Input damping factor on calculation of $\partial(rV_\theta)/\partial t$ within blade row from outer iteration to outer iteration
DOM(MHTP1)		PRECAL	Distance up leading or trailing edge of blade, m
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
DRHOSL		TSONIN	Difference in density between suction and pressure surfaces along a streamline, kg/m^3
DTAN1		TSONIN	$d\theta/dm$ at upper blade surface leading- or trailing-edge tangency point, rad/m
DTAN2		TSONIN	$d\theta/dm$ at lower blade surface leading- or trailing-edge tangency point, rad/m
DTDM1(MM)		TSONIN	$d\theta/dm$ on upper blade surface at TSONIC input points, rad/m
DTDM2(MM)		TSONIN	$d\theta/dm$ on lower blade surface at TSONIC input points, rad/m
DTDRLE		INDEV LINDV	$(\partial\theta/\partial r)_{le}$, rad/m
DTDROM		THETOM	$\partial\theta/\partial r$ on orthogonal mesh, rad/m
DTDRTE		INDEV LINDV	$(\partial\theta/\partial r)_{te}$, rad/m
DTDS(NPPP)		INPUT	$\partial\theta/\partial s$, rad/m
DTDT(MHTP1)		NEWRHO	dT''/dt along vertical mesh lines, K/m

Variable name	COMMON block	Subroutine	Description and comments
DTDZLE		INDEV LINDV	$(\partial\theta/\partial z)_{le}$, rad/m
DTDZOM		THETOM	$\partial\theta/\partial z$ on orthogonal mesh, rad/m
DTDZTE		INDEV, LINDV	$(\partial\theta/\partial z)_{te}$, rad/m
DTHDR(NPPC, NBLPC)	INDCOM		$\partial\theta/\partial r$ on alternate blade mesh (fig. 27), rad/m
DTHDS(MM, MHTP1)	CALCON		$\partial\theta/\partial s$ at orthogonal mesh points, rad/m
DTHDSL		INPUT PRECAL THETOM	$d\theta/ds$ at blade leading edge, rad/m
DTHDSP(NPPC, NBLPC)		THETOM	$\partial\theta/\partial s'$ on alternate blade mesh (fig. 27), rad/m
DTHDST		INPUT PRECAL THETOM	$d\theta/ds$ at blade trailing edge, rad/m
DTHDT(MM, MHTP1)	CALCON		$\partial\theta/\partial t$ at orthogonal mesh points, rad/m
DTHDTP(NPPC, NBLPC)		THETOM	$\partial\theta/\partial t'$ on alternate blade mesh (fig. 27), rad/m
DTHDZ(NPPC, NBLPC)	INDCOM		$\partial\theta/\partial z$ on alternate blade mesh (fig. 27), rad/m
DTHEOM(MHTP1)		THIKOM	Blade thickness along leading or trailing edge, rad
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
DTST1(NBLPL)		THETOM	$d\theta/ds'$ at input blade planes along 5-percent-chord lines, rad/m

Variable name	COMMON block	Subroutine	Description and comments
DTST2(NBLPC)		THETOM	$d\theta/ds'$ at alternate mesh points along 5-percent-chord lines, rad/m
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
DVTEMP		COEF	Updated estimate of $d(rV_\theta)/dt$ at a mesh point, m/sec
DVTHDT(MM, MHTP1)		COEF	$d(rV_\theta)/dt$ 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
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(MHTP1 or NPPP, etc.)		INPUT PRECAL THETOM	Temporary storage for derivatives of SPLINE or SPLINT calls

Variable name	COMMON block	Subroutine	Description and comments
EM(NIN or NOUT)		TIPF RHOIPF LAMDAF RVTHTA	Second derivatives of spline-fit curves
ERR		NEWRHO	Relative change in W at a mesh point
ERROR		SOR	Maximum absolute value of change in u at any point for an overrelaxation iteration
ETVEL(MHTP1)		TVELCY	Coefficients, e, of velocity-gradient equation ((A7), part I) at mesh points along vertical mesh lines
EXFRAC		OUTPUT LINDV	Extrapolation fraction at leading and trailing edges
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_t 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_t at any mesh point during an outer iteration
FMIN		BLDVEL	Minimum new predicted value of F_t at any mesh point during an outer iteration
FNEW	INPUTT		Input damping factor on calculation of F_t from outer iteration to outer iteration

Variable name	COMMON block	Subroutine	Description and comments
FST(MM, MHTP1)		BLDVEL	rV_θ at orthogonal mesh points, m^2/sec
FT(MM, MHTP1)	VARCOM		F_t at orthogonal mesh points (eq. (A4)), m/sec^2
FTT		BLDVEL	Predicted value of F_t at a mesh point
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
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)
IARG		LINDV	Integer indicator for mesh line where deviation correction begins
ICARDS		TSONIN	Integer control on writing of TSONIC card image input to a file
ICOUNT		SOR NEWRHO TSONIN TVELCY LINDV	Integer internal iteration counter
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:

Variable name	COMMON block	Subroutine	Description and comments
			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
ILE(MHTP1)	CALCON		Vertical mesh line numbers of first mesh point inside blade region at leading edge
ILOS		LOSSOM	Integer control on print indicating negative loss
ILS(NSL)	SLCOM		Vertical mesh line number of first intersection of a streamline with a vertical mesh line inside blade region at leading edge
IMAX		TVELCY	Integer indicator of final vertical mesh line where an incidence correction is made
IMESH	INPUTT		Integer input indicating the multiple of outer iterations at which major output is printed for orthogonal mesh
INCR		TVELCY	Integer increment for loop on vertical mesh lines
IND		TVELCY	Integer that indicates solution procedure in CONTIN
IPLOT	INPUTT		Integer input indicating multiple of outer iterations at which major output is plotted

Variable name	COMMON block	Subroutine	Description and comments
IPRNT		INDEV	Integer indicator controlling output
IREVRS		TVELCY	Integer indicator at end of forward iteration
ISLINE	INPUTT		Integer input indicating multiple of outer iterations at which major output is printed along streamlines
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
ITEMIN		TVELCY	Integer indicating smallest value of IARG
ITEMP		TVELCY	Temporary storage for IARG
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. 3)
JARG		LINDV	Integer indicator for mesh line where incidence correction ends

Variable name	COMMON block	Subroutine	Description and comments
JZ		TVELCY	Integer used to indicate to CONTIN that subsonic (JZ = 1) or supersonic (JZ = 2) solution is desired
K(MM, MHTP1) (real variable)	VARCOM		k_0 (eq. (A9)) at orthogonal mesh points
KNEW (real variable)		COEF	Updated value of k_0 (eq. (A9)) at a mesh point
LAMBDA(MHTP1) (real variable)		TVELCY	λ for mesh points along vertical mesh lines, m^2/sec
LAMBDA (real variable)		NEWRHO OUTPUT	λ, m^2/sec
LAMBDO(MHTP1) (real variable)		TVELCY	$(rV_\theta)_o$ for mesh points along vertical mesh lines, m^2/sec
LAMBDO (real variable)		NEWRHO	$(rV_\theta)_o$, 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 $(rV_\theta)_o$ 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
LBLAD	INPUTT		Input integer (0, 1, or 2) indicating type of blade geometry input
LETEAN	INPUTT		Input integer (0 or 1) indicating whether leading- and trailing-edge blade angles are given

Variable name	COMMON block	Subroutine	Description and comments
LINC		TVELCY LINDV	Integer indicating completion of transonic incidence correction at leading edge
LIPS		TSONIN	Input for TSONIC (ref. 3)
LMAX (real variable)		SOR	Maximum value of RATIO over all mesh points
LMIN (real variable)		SOR	Minimum value of RATIO over all mesh points
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
LOSS		TSONIN	Input for TSONIC (ref. 3)
LROT	INPUTT		Input integer (0 or 1) indicating if roration of coordinate system should be used
LRVB		TSONIN	Input for TSONIC (ref. 3)
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
LWCR		TSONIN	Input for TSONIC (ref. 3)
MARK		OUTPUT	Integers between 1 and 4 indicating whether output station lines are outside blade, within blade, or on leading or trailing edge
MAXFLO (real variable)		TVELCY	Maximum mass flow for which a solution can be obtained for a vertical mesh line, kg/sec

Variable name	COMMON block	Subroutine	Description and comments
MBBI		TSONIN	Input for TSONIC (ref. 3)
MBI	INPUTT		Input number of vertical mesh lines from left boundary of orthogonal mesh (ZOMIN) to first point of mesh-size change (ZOMBI)
MBITS		TSONIN	Input for TSONIC (ref. 3)
MBO	INPUTT		Input total number of vertical mesh lines from left boundary of orthogonal mesh (ZOMIN) to point of second mesh-size change (ZOMBO)
MBOTS		TSONIN	Input for TSONIC (ref. 3)
MHT	INPUTT		Input total number of horizontal mesh spaces from hub to shroud of orthogonal mesh; maximum of 100
MHTP1	CALCON		MHT + 1
MINFLO (real variable)		TVELCY	Minimum mass flow for which a solution can be obtained for a vertical mesh line, kg/sec
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
MMTS		TSONIN	Input for TSONIC (ref. 3)
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 is upstream boundary of orthogonal mesh.)

Variable name	COMMON block	Subroutine	Description and comments
MST(NSL) (real variable)		OUTPUT	m-coordinates of points where station lines cross streamlines, m (Origin of m-coordinates is upstream boundary of orthogonal mesh.)
NADD		TVELCY	Integer number of times WHUB is increased for restart of iteration procedure
NBL	INPUTT		Input number of blades in blade row
NBLPC	INDCOM		Integer number of blade planes in alternate mesh
NBLPL	INPUTT		Number of input blade planes or blade sections on which data (ZBL, RBL, THBL, TNBL, etc.) 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
NCHOK		OUTPUT	Integer number of vertical orthogonal mesh lines that are choked in the transonic solution
NCOUNT		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, and VTHIN), maximum of 50
NLOSS	INPUTT		Number of input data points in PERCRD and PERLOS arrays, maximum of 50
NMAX		MESHO	Maximum of either NHUB or NTIP

Variable name	COMMON block	Subroutine	Description and comments
NOSTAT	INPUTT		Input number of hub-shroud stations (located by coordinates in ZHST, RHST, ZTST, and RTST) 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, and VTHOUT), maximum of 50
NPPC	INDCOM		Integer number of vertical percent-chord lines in alternate mesh
NPPP	INPUTT		Number of input data points per blade section or blade plane in ZBL, RBL, etc., arrays; maximum of 50
NREAD	Blank		Integer number of input read file
NREP		TVELCY	Integer number of repeats on outer iteration loop
NRES		TVELCY	Integer number of restarts of inner iteration for a given vertical orthogonal mesh line
NRSP		TSONIN	Input for TSONIC (ref. 3)
NSL	INPUTT		Input number of streamlines from hub to shroud (designated by values in FLFR) at which output is desired, maximum of 50
NSLTS		TSONIN	Input for TSONIC (ref. 3)
NSPL1		TSONIN	Input for TSONIC (ref. 3)
NSPL2		TSONIN	Input for TSONIC (ref. 3)
NSUB		TVELCY	Integer number of times WHUB is decreased for restart of iteration procedure
NTIP	INPUTT		Number of input data points in ZTIP and RTIP arrays, maximum of 50

Variable name	COMMON block	Subroutine	Description and comments
NWRIT, NWRT1, NWRT2, NWRT3, NWRT4, NWRT5, NWRT6, NWRT7	Blank		Integer numbers of output write files. Several variables are used so that output can be stored in more than one file.
OMEGA	INPUTT		Input rotational speed, ω, rad/sec
OMTEM		TSONIN	Input for TSONIC (ref. 3), rad/sec
ORF		SOR	Overrelaxation factor
ORFMAX		SOR	Current estimate for maximum value of ORF calculated by using LMAX
ORFMIN		SOR	Current estimate for minimum value of ORF calculated by using LMIN
PC		LOSSOM	Fraction of chord from leading edge
PERCRD(NLOSS)	INPUTT		Input fraction of chord for PERLOS input
PERLOS(NLOSS)	INPUTT		Input fraction of loss
PERLS		LOSSOM	Fraction of loss at an orthogonal mesh point
PHI		MESH OUTPUT	φ, deg
PITCH	CALCON		2π/NBL, rad
PLOSS(MM, MHTP1)	CALCON		Fractional loss of relative total pressure at orthogonal mesh points
PLOSSL(MM)		TSONIN	Fractional loss of relative total pressure along a streamline
PLOST(NSL)		OUTPUT	PLOSS at a point on a station line
PLOSTE		LINDV	Fractional loss of relative total pressure at blade trailing edge
PPPST(NSL)		OUTPUT	p'' along a station line, N/m^2
PPST(NSL)		OUTPUT	p' along a station line, N/m^2

Variable name	COMMON block	Subroutine	Description and comments
PREL(MHTP1)		NEWRHO	p'' at mesh points along vertical mesh lines, N/m^2
PREL		TVELCY	p'', N/m^2
PRELN		TVELCY	New p'', N/m^2
PRINP		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
PROP(NOUT)	INPUTT		Input, p_o', at points along line from hub to shroud on which downstream flow conditions are given, N/m^2
PST(NSL)		OUTPUT	p along a station line, N/m^2
R1(MM)		MESHO	r-coordinate of intersection of line (1), fig. 9, 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
RADLE		TSONIN	r at leading edge, m
RADOUT(NOUT)	INPUTT		Input r-coordinates of points along line from hub to shroud on which downstream flow conditions are given, m
RADSP1(MM)		TSONIN	r-coordinates of upper-blade-surface spline points, m
RADSP2(MM)		TSONIN	r-coordinates of lower-blade-surface spline points, m
RADTE		TSONIN	r at trailing edge, m
RATIO		SOR	u_i^{m+1}/u_i^m for use in equations (B2) and (B3) of reference 9
RBL(NPPP, NBLPL)	INPUTT		Input array of r-coordinates, corre-

Variable name	COMMON block	Subroutine	Description and comments
			sponding to ZBL, of points describing mean blade surface, m
RBLR		THIKOM	Rotated RBL points at leading and trailing edges, m
RBLROT(NPPP, NBLPL)	ROTATN		Rotated RBL array, m
RCARB(MHTP1)		TVELCY	$\cos(\alpha - \varphi)rB$ along a vertical mesh line, m
REDFAC	INPUTT		Input factor used to reduce mass flow (MSFL) in order to ensure subsonic flow throughout flow passage
REDTEM		TSONIN	Input for TSONIC (ref. 3)
RELER		NEWRHO	Maximum relative change in W at any mesh point between two outer iterations
RELERA		NEWRHO	Average relative change in W for all mesh points between two outer iterations
REPEAT		TVELCY	Logical variable indicating that velocity-gradient solutions should be repeated with new values of TIPT, RHOIP, and LAMBDA
REVERS		OUTPUT	Indicator of which blade surface (leading or trailing) is suction surface
RFAC2		LOSSOM	REDFAC squared, or 1.0 for transonic solution
RHIN	INPUTT		Input or calculated 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

Variable name	COMMON block	Subroutine	Description and comments
RHOBF		INDEV LINDV	ρ_{bf}, kg/m^3
RHOBFN		LINDV	New ρ_{bf}, kg/m^3
RHOFS		INDEV LINDV	ρ_{fs}, kg/m^3
RHOIP(MHTP1)		TVELCY	ρ_i' (1 - Ploss) 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		NEWRHO OUTPUT LINDV	ρ_i' (1 - Ploss), kg/m^3
RHOIP		TSONIN	ρ_i' input for TSONIC, kg/m^3
RHOL		BLDVEL TVELCY	ρ_l, kg/m^3
RHOP		OUTPUT	ρ', kg/m^3
RHOPP		OUTPUT	ρ'', kg/m^3
RHOSL		TSONIN	ρ at a ZSL, RSL point along a streamline, kg/m^3
RHOST(NSL)		OUTPUT	ρ along a station line, kg/m^3
RHOT		BLDVEL TVELCY	ρ_{tr}, kg/m^3
RHOUT	INPUTT		Input or calculated r-coordinate of intersection with hub profile of line on which downstream flow conditions are given, m
RHOWAV		TVELCY	$(\rho W)_{av}$, $kg/(sec)(m)^2$
RHROT(NHUB)	ROTATN		Rotated RHUB array, m
RHST(NOSTAT)	INPUTT		Input or calculated r-coordinates of intersections of hub-shroud output station lines with hub profile, m

Variable name	COMMON block	Subroutine	Description and comments
RHUB(NHUB)	INPUTT		Input r-coordinates of points defining hub or bottom boundary of flow channel, m
RI1		TSONIN	Input for TSONIC (ref. 3), m
RI2		TSONIN	Input for TSONIC (ref. 3), m
RILOM(MHTP1)		LAMDAF	Radii for spline fit of stream function against radius
RIR		LOSSOM	Rotated r-coordinate of intersection of upstream or downstream input station line with horizontal mesh line, m
RLE(NBLPL)	CALCON		r-coordinates of input blade-section points defining leading edge of blade, m
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
RLEOMR(MHTP1)	ROTATN		Rotated RLEOM array, 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
RMSP(MM)		TSONIN	r-coordinates of points defining a stream channel, printed as input for TSONIC (ref. 3), m
RNOR(2)		MESHO	Rotated r-coordinates of points on straight line normal to previous row, m
RO1		TSONIN	Input for TSONIC (ref. 3), m

Variable name	COMMON block	Subroutine	Description and comments
RO2		TSONIN	Input for TSONIC (ref. 3), m
ROLOM(MHTP1)		RVTHTA	Radii for spline fit of stream function against radius
ROM(MM, MHTP1)	CALCON		r-coordinates of orthogonal mesh, m
ROMBI	INPUTT		Input or calculated r-coordinate of intersection of vertical mesh line with hub profile where first change in mesh spacing occurs (MBI), m
ROMBO	INPUTT		Input or calculated r-coordinate of intersection of vertical mesh line with hub profile where second change in mesh spacing occurs (MBO), m
ROMIN	INPUTT		Input or calculated r-coordinate of intersection of left boundary of orthogonal mesh with hub profile, m
ROMOUT	INPUTT		Input or calculated r-coordinate of intersection of right boundary of orthogonal mesh (MM) with hub profile, m
ROMROT(MM, MHTP1)	ROTATN		Rotated ROM array, m
RPC(NPPC, NBLPC)	INDCOM		r-coordinates of intersections of percent-chord lines with input blade planes; later set to r-coordinates of alternate mesh (fig. 27), m
RPCT1(NBLPL)		THETOM	r-coordinates of intersections of percent-chord lines with input blade planes, m
RPCT2(NBLPC)		THETOM	r-coordinates along percent-chord lines of alternate mesh, m
RRAD(NHUB, MHTP1)		MESHO	r-coordinates of points along lines from input points on hub profile to shroud profile, m

Variable name	COMMON block	Subroutine	Description and comments
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
RST(NSL)		OUTPUT	r-coordinates of points along station lines where they cross streamlines, m
RTE(NBLPL)	CALCON		r-coordinates of input blade-section points defining trailing edge of blade, m
RTEH	CALCON		r-coordinate or intersection of trailing edge of blade with hub profile, m
RTEM(MHTP1, 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
RTEOMR(MHTP1)	ROTATN		Rotated RTEOM array, m
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
RTIN	CALCON		Input or 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

Variable name	COMMON block	Subroutine	Description and comments
RTOLER		TVELCY	Tolerance for relative error of calculated values of integrated mass flow
RTOUT	INPUTT		Input or calculated r-coordinate of intersection with shroud profile of line on which downstream flow conditions are given, m
RTROT(NTIP)	ROTATN		Rotated RTIP array, m
RTST(NOSTAT)	INPUTT		Input or calculated r-coordinates of intersections of hub-shroud output station lines with tip profile, m
RVA		TVELCY	$(\rho W)_{av,j} \cos \beta_j \cos(\alpha - \varphi)_j r_j B_j$, kg/(m)(sec)
RVAS		TVELCY	$(\rho W)_{av,j+1} \cos \beta_{j+1} \times \cos(\alpha - \varphi)_{j+1} r_{j+1} B_{j+1}$, kg/(m)(sec)
RVTHI		TSONIN	Input for TSONIC (ref. 3), m^2/sec
RVTHO		TSONIN	Input for TSONIC (ref. 3), m^2/sec
SAL		TVELCY	$\sin \alpha$
SAMP(MM, MHTP1)	VARCOM		$\sin(\alpha - \varphi)$ at orthogonal mesh points
SBETA		TVELCY	$\sin \beta$
SDIST		INDEV LINDV	DISTLE (or DISTTE) plus distance from blade leading (or trailing) edge out to first adjacent mesh point, m
SF		LAMDAF RVTHTA TIPF RHOIPF	Stream function
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

Variable name	COMMON block	Subroutine	Description and comments
			hub-shroud line on which downstream flow conditions are given
SLENTH		LOSSOM	Overall s-distance over which loss is distributed, m
SLEOM(MHTP1)	CALCON		s-coordinates of intersections of horizontal mesh lines with blade leading edge, m
SLIDLE		INDEV LINDV	Solidity at leading edge of blade where it is intersected by a horizontal mesh line
SLIDTE		INDEV LINDV	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 LAMDAF RVTHTA	Derivatives of spline-fit curves
SOM(MM, MHTP1)	CALCON		s-coordinates of orthogonal mesh, m
SOMIN(MHTP1)		LOSSOM	s-coordinate of intersection of line on which upstream flow conditions are given with horizontal mesh lines, when there are no blades, m
SOMOUT(MHTP1)		LOSSOM	s-coordinates of intersection of line on which downstream flow conditions are given with horizontal mesh lines, m
SPHI(MM, MHTP1)	CALCON		sin φ at orthogonal mesh points
STEOM(MHTP1)	CALCON		s-coordinates of intersections of horizontal mesh lines with blade trailing edge, m
STGR		TSONIN	Input for TSONIC (ref. 3), rad

Variable name	COMMON block	Subroutine	Description and comments
SZRBL(NPPP or NBLPL)		PRECAL THETOM	Arc length along input blade section or along hub-shroud percent-chord lines in meridional plane, m
SZRPC(NPPC or NBLPC)		THETOM	Arc length along vertical or horizontal lines of alternate mesh (fig. 27)
TANBBF		INDEV LINDV	$\tan \beta_{bf}$
TANBBL		INDEV LINDV	$\tan \beta_b$
TBFTIP		LINDV	T_{bf}/T_i'
TGROG	CALCON		$2\gamma R/(\gamma + 1)$
TH1BL(NPPP, NBLPL)	INPUTT		Input array of θ-coordinates, corresponding to ZBL, RBL of points describing upper blade surface, rad
TH2BL(NPPP, NBLPL)	INPUTT		Input array of θ-coordinates, corresponding to ZBL, RBL of points describing lower blade surface, rad
THBL(NPPP, NBLPL)	INPUTT		Input array of θ-coordinates, corresponding to ZBL, RBL 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(NPPC, NBLPC)	INDCOM		θ-coordinates of intersections of percent-chord lines with input blade planes; later set to θ-coordinates of alternate mesh (fig. 27), rad
THPCT1(NBLPL)		THETOM	θ-coordinates of intersections of percent-chord lines with input blade planes, rad

Variable name	COMMON block	Subroutine	Description and comments
THPCT2(NBLPC)		THETOM	θ-coordinates along percent-chord lines of alternate mesh, rad
THSL		TSONIN	θ-coordinate (relative to MERIDL origin, not TSONIC origin) of mean blade surface at points along meridional streamlines, rad
THSP1(MM)		TSONIN	Input for TSONIC (ref. 3), rad
THSP2(MM)		TSONIN	Input for TSONIC (ref. 3), rad
THSTAK		TSONIN	θ-coordinate of blade section in relation to hub blade section, rad
THTEOM(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		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
TIPT(MHTP1)		TVELCY	T_i' at points along vertical mesh lines, K
TIPT		NEWRHO OUTPUT	Temporary T_i'
TIPTEM		TSONIN	T_i' along a streamline, K; input TIP for TSONIC (ref. 3)
TITLEI(20)	INPUTT		Alphanumerical contents of input title card
TLTAN1		TSONIN	θ-coordinate at upper blade surface leading-edge tangency point, rad
TLTAN2		TSONIN	θ-coordinate at lower blade surface leading-edge tangency point, rad

Variable name	COMMON block	Subroutine	Description and comments
TMSL		TSONIN	m-coordinates of tangency points on both blade surfaces, m
TNBL(NPPP, NBLPL)	INPUTT		Input array of blade normal thicknesses, corresponding to ZBL, RBL coordinates, m
TOLER		TIPF RHOIPF LAMDAF RVTHTA	Tolerance for a point close to a spline point
TOM(MM, MHTP1)	CALCON		t-coordinates of orthogonal mesh, m
TOP		LOSSOM	T'_o, K
TPP(MHTP1)		NEWRHO TVELCY OUTPUT	T" along a vertical mesh line, K
TPP		OUTPUT TVELCY LINDV	T", K
TPPN		TVELCY	New T", K
TPPST(NSL)		OUTPUT	T" along a station line, K
TPPTIP		INIT	T''/T'_i
TPST(NSL)		OUTPUT	T' along a station line, K
TST(NSL)		OUTPUT	T along a station line, K
TTBL(NPPP, NBLPL)	INPUTT		Input array of blade tangential thicknesses, corresponding to ZBL, RBL coordinates, rad
TTIP		NEWRHO BLDVEL TVELCY	T/T'_i
TTPC(NPPC, NBLPC)	INDCOM		Tangential blade thicknesses at intersections of percent-chord lines with input blade planes; later set to tang-

Variable name	COMMON block	Subroutine	Description and comments
			gential blade thicknesses on alternate mesh (fig. 27), rad
TTPCT1(NBLPL)		THETOM	Tangential blade thicknesses at intersections of percent-chord lines with input blade planes, rad
TTPCT2(NBLPC)		THETOM	Tangential blade thicknesses along percent-chord lines of alternate mesh, rad
TTTAN1		TSONIN	θ-coordinate at upper blade surface trailing-edge tangency point, rad
TTTAN2		TSONIN	θ-coordinate at lower blade surface trailing-edge tangency point, rad
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		BLDVEL LINDV	$2\omega\lambda - (\omega r)^2$, m^2/sec^2
UBDEV(MHTP1)		LINDV	Deviation angle, neglecting blockage correction, where horizontal orthogonal intersects blade, $(\beta_{fs} - \beta_b)_{le}$, deg
UBDEV		INDEV	Deviation angle, neglecting blockage correction, $(\beta_{fs} - \beta_b)_{te}$, deg
UBINC(MHTP1)		LINDV	Incidence angle, neglecting blockage correction, where horizontal orthogonal intersects blade, $(\beta_{fs} - \beta_b)_{le}$, deg
UBINC		INDEV	Incidence angle, neglecting blockage correction, $(\beta_{fs} - \beta_b)_{le}$, deg

Variable name	COMMON block	Subroutine	Description and comments
UILOM(MHTP1)		LAMDAF	Stream-function values for spline fit of stream function against radius
UNEW(MHTP1)		TVELCY	New estimate for u along a vertical mesh line
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)		NEWRHO	Temporary storage for values from UOM array along vertical mesh lines
VELTEM		TSONIN	Input for TSONIC (ref. 3)
VELTOL	INPUTT		Input convergence tolerance on maximum velocity change in each outer iteration, over all mesh points, for reduced mass flow
VST(NSL)		OUTPUT	V along a station line, m/sec
VTH(MM, MHTP1)	VARCOM		V_θ at orthogonal mesh points, m/sec
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
VTHST(NSL)		OUTPUT	V_θ along station line, m/sec

Variable name	COMMON block	Subroutine	Description and comments
W(MM, MHTP1)	VARCOM		W at orthogonal mesh points, m/sec
WAS		TVELCY	First estimate of W_{j+1} at next mesh point along vertical mesh line (eq. (5)), W^*_{j+1}, m/sec
WASS		TVELCY	Second estimate of W_{j+1} at next mesh point along vertical mesh line (eq. (5)), W^{**}_{j+1}, m/sec
WFS		LINDV	W_{fs} at blade leading or trailing edge, m/sec
WHIRL		TVELCY	λ or $(rV_\theta)_o$, m^2/sec
WHUB		TVELCY	Estimate of W_{hub}, m/sec
WLSRF		TVELCY	W_l, m/sec
WLSSL(MM, NSL)	SLCOM		W_l at points along streamlines where they cross vertical mesh lines, m/sec
WLSST(NSL)		OUTPUT	W_l at points along station lines where they cross streamlines, m/sec
WLSURF(MM, MHTP1)	VARCOM		W_l on orthogonal mesh, m/sec
WMAX		TVELCY	Maximum value of W at hub for which a solution can be obtained for a vertical mesh line, m/sec
WMIN		TVELCY	Minimum value of W at hub for which a solution can be obtained for a vertical mesh line, m/sec
WMSL(MM, NSL)	SLCOM		W_m at points where streamlines cross vertical mesh lines, m/sec
WMSON		LINDV	Sonic value for W_m, m/sec
WMST(NSL)		OUTPUT	W_m at points where station lines cross streamlines, m/sec
WMVERT(MHTP1)		TVELCY	W_m along vertical mesh lines, m/sec

Variable name	COMMON block	Subroutine	Description and comments
WRFSEX(MHTP1)		OUTPUT	W_r extrapolated up to leading or trailing edge of blade, m/sec
WRSL(MM, NSL)	SLCOM		W_r at points where streamlines cross vertical mesh lines, m/sec
WRST(NSL)		OUTPUT	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
WST(NSL)		OUTPUT	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
WTH(MM, MHTP1)	VARCOM		W_θ at orthogonal mesh points, m/sec
WTHETA		TVELCY LINDV	W_θ, m/sec
WTHSL(MM, NSL)	SLCOM		W_θ at points where streamlines cross vertical mesh lines, m/sec
WTHST(NSL)		OUTPUT	W_θ at points where station lines cross streamlines, m/sec
WTSRF		TVELCY	W_{tr}, m/sec
WTSSL(MM, NSL)	SLCOM		W_{tr} at points where streamlines cross vertical mesh lines, m/sec
WTSST(NSL)		OUTPUT	W_{tr} at points where station lines cross streamlines, m/sec

Variable name	COMMON block	Subroutine	Description and comments
WTSURF(MM, MHTP1)	VARCOM		W_{tr} on orthogonal mesh, m/sec
WTVERT(MHTP1)		TVELCY	W_θ along 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)		OUTPUT	W/W_{cr} at points where station lines cross streamlines
WZFSEX(MHTP1)		OUTPUT	W_z extrapolated up to leading or trailing edge of blade, m/sec
WZSL(MM, NSL)	SLCOM		W_z at points where streamlines cross vertical mesh lines, m/sec
WZST(NSL)		OUTPUT	W_z at points where station lines cross streamlines, m/sec
XIOM(MM, MHTP1)	VARCOM		ξ at orthogonal mesh points, (eq. (A2)), 1/m
XIOMT		NEWRHO	New estimated value of ξ at a mesh point
XNEW		NEWRHO	Percentage of new calculated value of XIOMT used in updating XIOM
Z1(MM)		MESHO	z-coordinate of intersection of line (1), figure 9, with upper horizontal mesh line, m
Z2		MESHO	z-coordinate of intersection of line (2), figure 9, with upper horizontal mesh line, m
ZBL(NPPP, NBLPL)	INPUTT		Input array of z-coordinates of points describing blade surface, m
ZBLROT(NPPP, NBLPL)	ROTATN		Rotated ZBL array, m
ZETOM(MM, MHTP1)	VARCOM		ζ at orthogonal mesh points (eq. (A3)), m/sec^2

Variable name	COMMON block	Subroutine	Description and comments
ZETOMT		NEWRHO	New estimated value of ζ at a mesh point
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
ZHROT(NHUB)	ROTATN		Rotated ZHUB array, 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
ZIR		LOSSOM	Rotated z-coordinate of intersection of upstream or downstream input station line with horizontal mesh line, m
ZLE(NBLPL)	CALCON		z-coordinates of input blade section points defining leading edge of blade, m
ZLEH		PRECAL	z-coordinate of intersection of leading edge of blade with hub profile, m
ZLEOM(MHTP1)	CALCON		z-coordinates of intersections of horizontal mesh lines with blade leading edge, m
ZLEOMR(MHTP1)	ROTATN		Rotated ZLEOM array, 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

Variable name	COMMON block	Subroutine	Description and comments
ZLTAN1		TSONIN	m-coordinate at upper blade surface leading-edge tangency point, m
ZLTAN2		TSONIN	m-coordinate at lower blade surface leading-edge tangency point, m
ZMRSP(MM)		TSONIN	Input for TSONIC (ref. 3), m
ZMSFL		TSONIN	Input for TSONIC (ref. 3), kg/sec
ZMSP1(MM)		TSONIN	Input for TSONIC (ref. 3), m
ZMSP2(MM)		TSONIN	Input for TSONIC (ref. 3), m
ZNEW		NEWRHO	Percentage of new calculated value of ZETOMT used in updating ZETOM
ZNOR(2)		MESHO	Rotated z-coordinates of points on straight line normal to previous row, m
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
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
ZOMROT(MM, MHTP1)	ROTATN		Rotated ZOM array, m
ZPC(NPPC, NBLPC)	INDCOM		z-coordinates of intersections of percent-chord lines with input blade

Variable name	COMMON block	Subroutine	Description and comments
			planes; later set to z-coordinates of alternate mesh (fig. 27), m
ZPCT1(NBLPL)		THETOM	z-coordinates of intersections of percent-chord lines with input blade planes, m
ZPCT2(NBLPC)		THETOM	z-coordinates along percent-chord lines of alternate mesh, m
ZRAD(NHUB, MHTP1)		MESHO	z-coordinates of points along lines from input points on hub profile to shroud profile, 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
ZST(NSL)		OUTPUT	z-coordinates of points where station lines cross streamlines, m
ZTE(NBLPL)	CALCON		z-coordinates of input blade-section points defining trailing edge of blade, m
ZTEH		PRECAL	z-coordinate of intersection of trailing edge of blade with hub profile, m
ZTEM(MHTP1, 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
ZTEOMR(MHTP1)	ROTATN		Rotated ZTEOM array, m

Variable name	COMMON block	Subroutine	Description and comments
ZTESL		TSONIN	z-coordinate of intersection of a streamline with trailing edge of blade, m
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
ZTROT(NTIP)	ROTATN		Rotated ZTIP array, m
ZTST(NOSTAT)	INPUTT		Input z-coordinates of intersections of hub-shroud output station lines with shroud profile, m
ZTTAN1		TSONIN	m-coordinate at upper blade surface trailing-edge tangency point, m
ZTTAN2		TSONIN	m-coordinate at lower blade surface trailing-edge tangency point, m

PROGRAM LISTING

```
C
C--MERIDL PROGRAM -- UPDATED TO JULY 1, 1977
C--REPORTS -- NASA TND-8430 AND TND-8431, 1977
C--PROGRAM AVAILABLE FROM COSMIC -- NUMBER LEW-12129
C--FOR INFORMATION CONTACT : DR. T. KATSANIS (MAIL STOP 77-2) OR
C--                          DR. W. MCNALLY  (MAIL STOP  5-9)
C--                          NASA LEWIS RESEARCH CENTER
C--                          CLEVELAND, OHIO  44135
C--                          PHONE :  (216)--433-4000
C
C
      COMMON NREAD,NWRIT,ITER,IEND,NWRT1,NWRT2,NWRT3,NWRT4,NWRT5,NWRT6
      COMMON/INPUTT/GAM,AR,MSFL,OMEGA,REDFAC,VELTOL,FNEW,DNEW,MBI,MBO,
     1    MM,MHT,NBL,NHUB,NTIP,NIN,NOUT,NBLPL,NPPP,NOSTAT,NSL,NLOSS,
     2    LSFR,LTPL,LAMVT,LROT,LBLAD,LFTEAN,ANGROT,IMESH,ISLINE,
     3    ISTATL,IPLOT,ISUPER,ITSON,IDEBUG,ZOMIN,ZOMBI,ZOMBO,ZOMOUT,
     4    ROMIN,ROMBI,ROMBO,ROMOUT,ZHIN,ZTIN,ZHOUT,ZTOUT,RHIN,RTIN,RHOUT,
     5    RTOUT,TITLEI(20),ZHUB(50),RHUB(50),ZTIP(50),RTIP(50),SFIN(50),
     6    RADIN(50),TIP(50),PRIP(50),LAMIN(50),VTHIN(50),SFOUT(50),
     7    RADOUT(50),PROP(50),LOSOUT(50),LAMOUT(50),VTHOUT(50),
     8    BETALE(50),BETATE(50),ZHST(50),ZTST(50),RHST(50),RTST(50),
     9    FLPR(50),PERCRD(50),PERLOS(50),ZBL(50,50),RBL(50,50),
     1    THBL(50,50),TNBL(50,50),TTBL(50,50),TH1BL(50,50),TH2BL(50,50)
      COMMON/CALCON/MMM1,MHTP1,CP,EXPON,TGROG,PITCH,RLEH,RLET,RTEH,RTET,
     1    ZLE(50),RLE(50),ZTE(50),RTE(50),ZLEOM(101),RLEOM(101),
     2    SLEOM(101),THLEOM(101),ZTEOM(101),RTEOM(101),STEOM(101),
     3    THTEOM(101),ILE(101),ITE(101),ZOM(100,101),ROM(100,101),
     4    SOM(100,101),TOM(100,101),BTH(100,101),DTHDS(100,101),
     5    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),FT(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/ROTATN/ZHROT(50),RHROT(50),ZTROT(50),RTROT(50),
     1    ZLEOMR(101),RLEOMR(101),ZTEOMR(101),RTEOMR(101),
     2    ZBLROT(50,50),RBLROT(50,50),ZOMROT(100,101),ROMROT(100,101)
      COMMON/INDCOM/NBLPC,NPPC,ZPC(51,51),RPC(51,51),TTPC(51,51),
     1    THPC(51,51),DTHDZ(51,51),DTHDR(51,51),BTHLE(101),BTHTE(101),
     2    BTBFLE(101),BTBFTE(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)
   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 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 LOSSOM
      CALL NEWRHO
C
C--CALCULATE AND PRINT MAJOR OUTPUT DATA
      CALL OUTPUT
      IF (MBI.NE.0) CALL INDEV
      IF (MBI.NE.0) CALL TSONIN
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
      REDFAC = 1.0
      CALL TOUTPT
      IF (MBI.NE.0) CALL PINDV
      IF (MBI.NE.0) CALL TSONIN
      CALL SLPLOT
      CALL SVPLOT
      IF (ISUPER.EQ.0.OR.ISUPER.EQ.2) GO TO 10
      ISUPER = 2
      GO TO 30
      END



      SUBROUTINE INPUT
C
C--INPUT READS AND PRINTS ALL INPUT DATA CARDS
C
      COMMON NREAD,NWRIT,ITER,IEND,NWRT1,NWRT2,NWRT3,NWRT4,NWRT5,NWRT6
      COMMON/INPUTT/GAM,AR,MSFL,OMEGA,REDFAC,VELTOL,FNEW,DNEW,MBI,MBO,
     1   MM,MHT,NBL,NHUB,NTIP,NIN,NOUT,NBLPL,NPPP,NOSTAT,NSL,NLOSS,
     2   LSFR,LTPL,LAMVT,LROT,LBLAD,LETEAN,ANGROT,IMESH,ISLINE,
     3   ISTATL,IPLOT,ISUPER,ITSON,IDEBUG,ZOMIN,ZOMBI,ZOMBO,ZOMOUT,
     4   ROMIN,ROMBI,ROMBO,ROMOUT,ZHIN,ZTIN,ZHOUT,ZTOUT,RHIN,RTIN,RHOUT,
     5   RTOUT,TITLEI(20),ZHUB(50),RHUB(50),ZTIP(50),RTIP(50),SFIN(50),
     6   RADIN(50),TIP(50),PRIP(50),LAMIN(50),VTHIN(50),SFOUT(50),
```

```
     7   RADOUT(50),PROP(50),LOSOUT(50),LAMOUT(50),VTHOUT(50),
     8   BETALE(50),BETATE(50),ZHST(50),ZTST(50),RHST(50),RTST(50),
     9   FLFR(50),PERCRD(50),PERLOS(50),ZBL(50,50),RBL(50,50),
     1   THBL(50,50),TNBL(50,50),TTBL(50,50),TH1BL(50,50),TH2BL(50,50)
      COMMON/CALCON/MMM1,MHTP1,CP,EXPON,TGROG,PITCH,RLEH,RLET,RTEH,RTET,
     1   ZLE(50),RLE(50),ZTE(50),RTE(50),ZLEOM(101),RLEOM(101),
     2   SLEOM(101),THLEOM(101),ZTEOM(101),RTEOM(101),STEOM(101),
     3   THTEOM(101),ILE(101),ITE(101),ZOM(100,101),ROM(100,101),
     4   SOM(100,101),TOM(100,101),BTH(100,101),DTHDS(100,101),
     5   DTHDT(100,101),PLOSS(100,101),CPHI(100,101),SPHI(100,101)
      DIMENSION DIST(50),DTDS(50),ANG(50)
      REAL MSFL,LAMIN,LAMOUT,LOSOUT
C
C--READ AND PRINT INPUT DATA
C
      NREAD = 5
      NWRIT = 6
      NWRT1 = 6
      NWRT2 = 6
      NWRT3 = 6
      NWRT4 = 6
      NWRT5 = 6
      NWRT6 = 6
   10 READ (NREAD,1050) (TITLEI(I),I=1,20)
      WRITE(NWRIT,1000)
      WRITE(NWRIT,1060) (TITLEI(I),I=1,20)
      READ (NREAD,1030) GAM,AR,MSFL,OMEGA,REDFAC,VELTOL,FNEW,DNEW
      IF (GAM.NE.0.) WRITE(NWRIT,1100)
      IF (GAM.EQ.0.) WRITE(NWRIT,1102)
      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
      WRITE(NWRIT,1040) GAM,AR,MSFL,OMEGA,REDFAC,VELTOL,FNEW,DNEW
      VELTOL = VELTOL*AMIN1(FNEW,DNEW)
      IF (FNEW.LT.1.0.OR.DNEW.LT.1.0) WRITE(NWRIT,1105) VELTOL
      WRITE(NWRIT,1110)
      READ (NREAD,1010) MBI,MBO,MM,MHT,NBL,NHUB,NTIP,NIN,NOUT,NBLPL,
     1NPPP,NOSTAT,NSL,NLOSS
      IF (MBI.EQ.0) NBL=1
      WRITE(NWRIT,1020) MBI,MBO,MM,MHT,NBL,NHUB,NTIP,NIN,NOUT,NBLPL,
     1NPPP,NOSTAT,NSL,NLOSS
      WRITE(NWRIT,1120)
      READ (NREAD,1010) LSFR,LTPL,LAMVT,LROT,LBLAD,LETEAN
      WRITE(NWRIT,1020) LSFR,LTPL,LAMVT,LROT,LBLAD,LETEAN
      ANGROT = 0.
      IF (LROT.EQ.0) GO TO 15
      WRITE(NWRIT,1125)
      READ (NREAD,1030) ANGROT
      WRITE(NWRIT,1040) ANGROT
      ANGROT = ANGROT/57.295780
   15 WRITE(NWRIT,1130)
      READ (NREAD,1030) ZOMIN,ZOMBI,ZOMBO,ZOMOUT,ROMIN,ROMBI,ROMBO,
     1ROMOUT
      WRITE(NWRIT,1040) ZOMIN,ZOMBI,ZOMBO,ZOMOUT,ROMIN,ROMBI,ROMBO,
     1ROMOUT
      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,RHIN,RTIN
      WRITE(NWRIT,1040)  ZHIN,ZTIN,RHIN,RTIN
      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,RHOUT,RTOUT
      WRITE(NWRIT,1040)  ZHOUT,ZTOUT,RHOUT,RTOUT
      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(MBI.EQ.0) GO TO 157
      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)
    IF (LBLAD.EQ.2) GO TO 150
    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)
    IF (LBLAD.EQ.1) GO TO 146
    WRITE(NWRIT,1350)
    DO 145 JN=1,NBLPL
    READ (NREAD,1030)  (TNBL(IN,JN),IN=1,NPPP)
145 WRITE(NWRIT,1040)  (TNBL(IN,JN),IN=1,NPPP)
    GO TO 156
146 WRITE(NWRIT,1352)
    DO 148 JN=1,NBLPL
    READ (NREAD,1030)  (TTBL(IN,JN),IN=1,NPPP)
148 WRITE(NWRIT,1040)  (TTBL(IN,JN),IN=1,NPPP)
    GO TO 156
150 WRITE(NWRIT,1354)
    DO 152 JN=1,NBLPL
    READ (NREAD,1030)  (TH1BL(IN,JN),IN=1,NPPP)
152 WRITE(NWRIT,1040)  (TH1BL(IN,JN),IN=1,NPPP)
    WRITE(NWRIT,1356)
    DO 154 JN=1,NBLPL
    READ (NREAD,1030)  (TH2BL(IN,JN),IN=1,NPPP)
154 WRITE(NWRIT,1040)  (TH2BL(IN,JN),IN=1,NPPP)
156 IF (LETEAN.EQ.0) GO TO 157
    WRITE(NWRIT,1358)
    READ (NREAD,1030)  (BETALE(JN),JN=1,NBLPL)
    WRITE(NWRIT,1040)  (BETALE(JN),JN=1,NBLPL)
    WRITE(NWRIT,1359)
    READ (NREAD,1030)  (BETATE(JN),JN=1,NBLPL)
    WRITE(NWRIT,1040)  (BETATE(JN),JN=1,NBLPL)
157 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)
    IF (LROT.EQ.0) GO TO 158
    WRITE(NWRIT,1365)
    READ (NREAD,1030)  (RHST(I),I=1,NOSTAT)
    WRITE(NWRIT,1040)  (RHST(I),I=1,NOSTAT)
158 WRITE(NWRIT,1370)
    READ (NREAD,1030)  (ZTST(I),I=1,NOSTAT)
    WRITE(NWRIT,1040)  (ZTST(I),I=1,NOSTAT)
    IF (LROT.EQ.0) GO TO 160
    WRITE(NWRIT,1375)
    READ (NREAD,1030)  (RTST(I),I=1,NOSTAT)
    WRITE(NWRIT,1040)  (RTST(I),I=1,NOSTAT)
160 IF (NSL.EQ.0) GO TO 165
    WRITE(NWRIT,1380)
    READ (NREAD,1030)  (FLFR(I),I=1,NSL)
```

```
      WRITE(NWRIT,1040)  (FLFR(I),I=1,NSL)
  165 IF (NLOSS.EQ.0) GO TO 170
      WRITE(NWRIT,1385)
      READ  (NREAD,1030)  (PERCRD(I),I=1,NLOSS)
      WRITE(NWRIT,1040)  (PERCRD(I),I=1,NLOSS)
      WRITE(NWRIT,1386)
      READ  (NREAD,1030)  (PERLOS(I),I=1,NLOSS)
      WRITE(NWRIT,1040)  (PERLOS(I),I=1,NLOSS)
  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,1070)
      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.NLOSS.LE.50.AND.LSFR.GE.0.AND.
     3LSFR.LE.1.AND.LTPL.GE.0.AND.LTPL.LE.1.AND.LAMVT.GE.0.AND.
     4LAMVT.LE.1.AND.LROT.GE.0.AND.LROT.LE.1.AND.LBLAD.GE.0.AND.
     5LBLAD.LE.2.AND.LETEAN.GE.0.AND.LETEAN.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--CALCULATE VALUES FOR RHIN,RTIN,RHOUT, AND RTOUT
C--IF ROTATION IS NOT USED
C
      IF (LROT.NE.0) GO TO 200
      CALL SPLINT(ZHUB,RHUB,NHUB,ZHIN,1,RHIN,DYDX,D2YDX2)
      CALL SPLENT(ZHOUT,1,RHOUT,DYDX,D2YDX2)
      CALL SPLINT(ZTIP,RTIP,NTIP,ZTIN,1,RTIN,DYDX,D2YDX2)
      CALL SPLENT(ZTOUT,1,RTOUT,DYDX,D2YDX2)
C
C--CALCULATE ESTIMATED UPSTREAM AND DOWNSTREAM VALUES OF
C--STREAM FUNCTION, IF RADIUS WAS GIVEN AS INPUT
C
  200 IF (LSFR.EQ.0.AND.LAMVT.EQ.0) GO TO 320
      RINSQ = RTIN**2-RHIN**2
      ROUTSQ = RTOUT**2-RHOUT**2
      IF (LSFR.EQ.0) GO TO 230
      IF(RINSQ*ROUTSQ.EQ.0.) WRITE(NWRIT,1410)
      IF(RINSQ*ROUTSQ.EQ.0.) STOP
      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)
      IF (LSFR.EQ.1) LAMVT=1
      IF (MBI.EQ.0) RETURN
      DO 280 J=1,NOUT
  280 VTHOUT(J) = LAMOUT(J)/RADOUT(J)
      GO TO 320
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)
      IF (MBI.EQ.0) RETURN
      DO 310 J=1,NOUT
  310 LAMOUT(J) = RADOUT(J)*VTHOUT(J)
C
C--CALCULATE TANGENTIAL THICKNESS, IF NORMAL THICKNESS GIVEN AS INPUT
C
  320 IF (MBI.EQ.0) RETURN
      IF (LBLAD.EQ.1) GO TO 370
      IF (LBLAD.EQ.2) GO TO 390
      DIST(1) = 0.
      DO 360 JN=1,NBLPL
      DO 330 IN=2,NPPP
  330 DIST(IN) = DIST(IN-1)+SQRT((ZBL(IN,JN)-ZBL(IN-1,JN))**2+
     1(RBL(IN,JN)-RBL(IN-1,JN))**2)
      IF (LETEAN.EQ.1) GO TO 340
      CALL SPLINE(DIST,THBL(1,JN),NPPP,DTDS,ANG)
      GO TO 350
  340 DTHDSL = TAN(BETALE(JN)/57.295780)/RBL(1,JN)
      DTHDST = TAN(BETATE(JN)/57.295780)/RBL(NPPP,JN)
      CALL SPLISL(DIST,THBL(1,JN),NPPP,DTHDSL,DTHDST,DTDS,ANG)
  350 DO 360 IN=1,NPPP
      ANG(IN) = ATAN(RBL(IN,JN)*DTDS(IN))
  360 TTBL(IN,JN) = TNBL(IN,JN)/COS(ANG(IN))/RBL(IN,JN)
C
C--CALCULATE BLADE SURFACE THETA COORDINATES, IF THEY ARE NOT
C--GIVEN AS INPUT
C
  370 DO 380 JN=1,NBLPL
      DO 380 IN=1,NPPP
      TH1BL(IN,JN) = THBL(IN,JN)+TTBL(IN,JN)/2.
  380 TH2BL(IN,JN) = THBL(IN,JN)-TTBL(IN,JN)/2.
      RETURN
C
C--CALCULATE MEAN CAMBER LINE THETA COORDINATES AND TANGENTIAL THICKNESS
C--IF SURFACE THETA COORDINATES GIVEN AS INPUT
C
  390 DO 400 JN=1,NBLPL
      DO 400 IN=1,NPPP
      THBL(IN,JN) = (TH1BL(IN,JN)+TH2BL(IN,JN))/2.
```

```
  400 TTBL(IN,JN) =  TH1BL(IN,JN)-TH2BL(IN,JN)
      RETURN
C
C--FORMAT STATEMENTS
C
 1000 FORMAT (1H1//50X,21(1H*)/50X,1H*,7X,6HMERIDL,6X,1H*/50X,21H*   PRO
     1GRAM INPUT   */50X,21(1H*)///)
 1010 FORMAT (16I5)
 1020 FORMAT (2X,16(2X,I5))
 1030 FORMAT (8F10.5)
 1040 FORMAT (1X,8G16.7)
 1050 FORMAT (20A4)
 1060 FORMAT (1X,20A4)
 1070 FORMAT (1H1)
 1100 FORMAT (///4X,20HGENERAL  INPUT  DATA/7X,3HGAM,14X,2HAR,13X,
     14HMSFL,11X,5HOMEGA,11X,6HREDFAC,10X,6HVELTOL,10X,4HFNEW,11X,
     24HDNEW)
 1102 FORMAT (///4X,20HGENERAL  INPUT  DATA/7X,3HGAM,13X,3HRHO,13X,
     14HMSFL,11X,5HOMEGA,11X,6HREDFAC,10X,6HVELTOL,10X,4HFNEW,11X,
     24HDNEW)
 1105 FORMAT (7X,71HVELTOL  HAS  BEEN  REDUCED  BY  THE  MINIMUM  OF  FN
     1EW  OR  DNEW  TO  =,8X,6HVELTOL/81X,G16.7)
 1110 FORMAT (103H       MBI    MBO    MM     MHT    NBL   NHUB   NTIP
     1    NIN   NOUT   NBLPL  NPPP  NOSTAT   NSL   NLOSS)
 1120 FORMAT (47H      LSFR   LTPL   LAMVT  LROT   LBLAD  LETEAN)
 1125 FORMAT (6X,6HANGROT)
 1130 FORMAT (///4X,29HHUB  AND  SHROUD  INPUT  DATA/7X,5HZOMIN,11X,
     1 5HZOMBI,11X,5HZOMBO,10X,6HZOMOUT,11X,5HROMIN,11X,5HROMBI,11X,
     2 5HROMBO,10X,6HRCMOUT)
 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,
     1 11X,4HRHIN,11X,4HRTIN)
 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,
     1 10X,5HRHOUT,10X,5HRTOUT)
 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)
 1352 FORMAT (7X,11HTTBL  ARRAY)
 1354 FORMAT (7X,12HTH1BL  ARRAY)
 1356 FORMAT (7X,12HTH2BL  ARRAY)
 1358 FORMAT (7X,13HBETALE  ARRAY)
```

```
1359 FORMAT (7X,13HBETATE  ARRAY)
1360 FORMAT (///4X,31HOUTPUT  STATION  LOCATION  DATA/7X,11HZHST  ARRAY
    1)
1365 FORMAT (7X,11HRHST  ARRAY)
1370 FORMAT (7X,11HZTST  ARRAY)
1375 FORMAT (7X,11HRTST  ARRAY)
1380 FORMAT (///4X,40HOUTPUT  STREAMLINE  FLOW  FRACTION  DATA/7X,11HFL
    1FR  ARRAY)
1385 FORMAT (///4X,28HDISTRIBUTION  OF  LOSS  DATA/7X,13HPERCRD  ARRAY)
1386 FORMAT (7X,13HPERLOS  ARRAY)
1390 FORMAT (///4X,28HOUTPUT  PRINT  CONTROL  DATA/6X,48HIMESH  ISLINE
    1ISTATL IPLOT  ISUPER ITSON  IDEBUG)
1400 FORMAT (1H1,10X,90HMM,MHT,NHUB,NTIP,NIN,NOUT,NBLPL,NPPP,NOSTAT,NSL
    1,NLOSS,LSFR,LTPL,LAMVT,LROT,LBLAD,OR LETEAN/13X,25HIS TOO LARGE OR
    2 TOO SMALL)
1410 FORMAT (1H1,10X,74HWHEN UPSTREAM AND DOWNSTREAM INPUT DATA ARE GIV
    1EN AS A FUNCTION OF RADIUS,/11X,86HTHERE MUST BE A CHANGE IN VALUE
    2 BETWEEN RHIN AND RTIN AND ALSO BETWEEN RHOUT AND RTOUT/11X,57HAND
    3 A CORRESPONDING CHANGE IN THE RADIN AND RADOUT ARRAYS)
     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,NLOSS,
     2    LSFR,LTPL,LAMVT,LROT,LBLAD,LETEAN,ANGROT,IMESH,ISLINE,
     3    ISTATL,IPLOT,ISUPER,ITSON,IDEBUG,ZOMIN,ZOMBI,ZOMBO,ZOMOUT,
     4    ROMIN,ROMBI,ROMBO,ROMOUT,ZHIN,ZTIN,ZHOUT,ZTOUT,RHIN,RTIN,RHOUT,
     5    RTOUT,TITLEI(20),ZHUB(50),RHUB(50),ZTIP(50),RTIP(50),SFIN(50),
     6    RADIN(50),TIP(50),PRIP(50),LAMIN(50),VTHIN(50),SFOUT(50),
     7    RADOUT(50),PROP(50),LOSOUT(50),LAMOUT(50),VTHOUT(50),
     8    BETALE(50),BETATE(50),ZHST(50),ZTST(50),RHST(50),RTST(50),
     9    FLFR(50),PERCRD(50),PERLOS(50),ZBL(50,50),RBL(50,50),
     1    THBL(50,50),TNBL(50,50),TTBL(50,50),TH1BL(50,50),TH2BL(50,50)
      COMMON/CALCON/MMM1,MHTP1,CP,EXPON,TGROG,PITCH,RLEH,RLET,RTEH,RTET,
     1    ZLE(50),RLE(50),ZTE(50),RTE(50),ZLEOM(101),RLEOM(101),
     2    SLEOM(101),THLEOM(101),ZTEOM(101),RTEOM(101),STEOM(101),
     3    THTEOM(101),ILE(101),ITE(101),ZOM(100,101),ROM(100,101),
     4    SOM(100,101),TOM(100,101),BTH(100,101),DTHDS(100,101),
     5    DTHDT(100,101),PLOSS(100,101),CPHI(100,101),SPHI(100,101)
      DIMENSION MBL(50,50),RTHBL(50,50),RTH1BL(50,50),RTH2BL(50,50),
     1    RRTHBL(50,50),RTH3BL(50,50),RTH4BL(50,50),
     2    PLTX(101),PLTY(101),DYDX(101),D2YDX2(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),TITL26(7),
     7    TITL27(2),TITL28(2)
      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 TITL26/'PRES','SURE',' LOS','S DI','STRI','BUTI','ON  '/
      DATA TITL27/'PERC','RD  '/
      DATA TITL28/'PERL','OS  '/
      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))
   40 CALL LRANGE(0.,0.,BRNG,TRNG)
C
C--PLOT INLET ABSOLUTE TOTAL TEMPERATURE
C
      IF (LSFR.EQ.0) CALL SPLINT(SFIN,TIP,NIN,PLTY,101,PLTX,DYDX,
     1D2YDX2)
      IF (LSFR.EQ.1) CALL SPLINT(RADIN,TIP,NIN,PLTY,101,PLTX,DYDX,
     1D2YDX2)
      CALL LRMRGN(1.0,1.0,2.0,1.0)
      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)
      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
      IF (LSFR.EQ.0) CALL SPLINT(SFIN,PRIP,NIN,PLTY,101,PLTX,DYDX,
     1D2YDX2)
      IF (LSFR.EQ.1) CALL SPLINT(RADIN,PRIP,NIN,PLTY,101,PLTX,DYDX,
     1D2YDX2)
      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)
      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
      IF (LSFR.EQ.0) CALL SPLINT(SFIN,LAMIN,NIN,PLTY,101,PLTX,DYDX,
     1D2YDX2)
      IF (LSFR.EQ.1) CALL SPLINT(RADIN,LAMIN,NIN,PLTY,101,PLTX,DYDX,
     1D2YDX2)
      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)
      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 IF (LSFR.EQ.0) CALL SPLINT(SFIN,LAMIN,NIN,PLTY,101,PLTX,DYDX,
     1D2YDX2)
      IF (LSFR.EQ.1) CALL SPLINT(RADIN,LAMIN,NIN,PLTY,101,PLTX,DYDX,
     1D2YDX2)
      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)
      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
      IF (LSFR.EQ.0) CALL SPLINT(SFOUT,PROP,NOUT,PLTY,101,PLTX,DYDX,
     1D2YDX2)
      IF (LSFR.EQ.1) CALL SPLINT(RADOUT,PROP,NOUT,PLTY,101,PLTX,DYDX,
     1D2YDX2)
      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)
      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 IF (LSFR.EQ.0) CALL SPLINT(SFOUT,LOSOUT,NOUT,PLTY,101,PLTX,DYDX,
     1D2YDX2)
      IF (LSFR.EQ.1) CALL SPLINT(RADOUT,LOSOUT,NOUT,PLTY,101,PLTX,DYDX,
```

```
     1D2YDX2)
      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)
      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)
  190 CALL LRCHSZ(0)
      IF (MBT.EQ.0) GO TO 240
C
C--PLOT OUTLET ABSOLUTE WHIRL
C
      IF (LAMVT.EQ.1) GO TO 210
      IF (LSFR.EQ.0) CALL SPLINT(SFOUT,LAMOUT,NOUT,PLTY,101,PLTX,DYDX,
     1D2YDX2)
      IF (LSFR.EQ.1) CALL SPLINT(RADOUT,LAMOUT,NOUT,PLTY,101,PLTX,DYDX,
     1D2YDX2)
      CALL LRCHSZ(4)
      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)
      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 IF (LSFR.EQ.0) CALL SPLINT(SFOUT,LAMOUT,NOUT,PLTY,101,PLTX,DYDX,
     1D2YDX2)
      IF (LSFR.EQ.1) CALL SPLINT(RADOUT,LAMOUT,NOUT,PLTY,101,PLTX,DYDX,
     1D2YDX2)
      CALL LRCHSZ(4)
      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)
      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 PERCRD AND PERLOS
C
  240 IF (NLOSS.EQ.0) GO TO 248
      LRNG = PERCRD(1)
      RRNG = PERCRD(NLOSS)
      BRNG = PERLOS(1)
```

```
      TRNG = PERLOS(NLOSS)
      DO 242 I=1,NLOSS
      LRNG = AMIN1(LRNG,PERCRD(I))
      RRNG = AMAX1(RRNG,PERCRD(I))
      BRNG = AMIN1(BRNG,PERLOS(I))
  242 TRNG = AMAX1(TRNG,PERLOS(I))
      PLTX(1) = PERCRD(1)
      PLTX(101) = PERCRD(NLOSS)
      DEL = (PERCRD(NLOSS)-PERCRD(1))/100.
      DO 244 I=2,100
  244 PLTX(I) = PLTX(I-1)+DEL
      CALL SPLINT(PERCRD,PERLOS,NLOSS,PLTX,101,PLTY,DYDX,D2YDX2)
      CALL LRANGE(LRNG,RRNG,BRNG,TRNG)
      CALL LRCHSZ(4)
      CALL LRLEGN(TITL26,28,0,1.7,0.5,0.0)
      CALL LRCHSZ(2)
      CALL LRLEGN(TITL27,8,0,4.7,1.3,0.0)
      CALL LRLEGN(TITL28,8,1,0.2,4.7,0.0)
      CALL LRCHSZ(4)
      CALL LRCURV(PLTX,PLTY,101,2,SYM,0.0)
      CALL LRCURV(PERCRD,PERLOS,NLOSS,4,SYM,1.0)
      CALL LRCHSZ(0)
C
C--PLOT INPUT BLADE SECTIONS
C
  248 IF (MBI.EQ.0) RETURN
C--CALCULATE BLADE SECTION PLOT COORDINATES ALONG MERIDIONAL PLANE
      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
      DO 260 IN=1,NPPP
      DELRTH = RBL(IN,JN)*PITCH
      RTHBL(IN,JN) = RBL(IN,JN)*THBL(IN,JN)
      RTH1BL(IN,JN) = RBL(IN,JN)*TH1BL(IN,JN)
      RTH2BL(IN,JN) = RBL(IN,JN)*TH2BL(IN,JN)
      RRTHBL(IN,JN) = RTHBL(IN,JN)+DELRTH
      RTH3BL(IN,JN) = RTH1BL(IN,JN)+DELRTH
  260 RTH4BL(IN,JN) = RTH2BL(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 = RTH2BL(1,1)
      TRNG = RTH3BL(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,RTH2BL(IN,JN))
  270 TRNG = AMAX1(TRNG,RTH3BL(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 LRCHSZ(4)
      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),RTH1BL(1,JN),NPPP,2,SYM,0.0)
      CALL LRCURV(MBL(1,JN),RTH2BL(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),RTH3BL(1,JN),NPPP,2,SYM,0.0)
      CALL LRCURV(MBL(1,JN),RTH4BL(1,JN),NPPP,2,SYM,0.0)
      IF (LBLAD.EQ.2) GO TO 275
      CALL LRCURV(MBL(1,JN),RTHBL(1,JN),NPPP,4,SYM,0.0)
      CALL LRCURV(MBL(1,JN),RRTHBL(1,JN),NPPP,4,SYM,1.0)
      GO TO 280
  275 CALL LRCURV(MBL(1,JN),RTH1BL(1,JN),NPPP,4,SYM,0.0)
      CALL LRCURV(MBL(1,JN),RTH2BL(1,JN),NPPP,4,SYM,0.0)
      CALL LRCURV(MBL(1,JN),RTH3BL(1,JN),NPPP,4,SYM,0.0)
      CALL LRCURV(MBL(1,JN),RTH4BL(1,JN),NPPP,4,SYM,1.0)
  280 CONTINUE
C--CALCULATE RANGE OF PLOT, AND SET UP FOR PLOT OF MULTIPLE
C--BLADE SECTIONS
      RRNG = RRTEM
      TRNG = RTH1BL(NPPP,NBLPL)
      DO 290 JN=1,NBLPL
      DO 290 IN=1,NPPP
  290 TRNG = AMAX1(TRNG,RTH1BL(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)
      EOP = 0.0
      DO 300 JN=1,NBLPL
      IF (JN.EQ.NBLPL) EOP=1.0
      CALL LRCURV(MBL(1,JN),RTHBL(1,JN),NPPP,2,SYM,0.0)
      CALL LRCURV(MBL(1,JN),RTH1BL(1,JN),NPPP,2,SYM,0.0)
  300 CALL LRCURV(MBL(1,JN),RTH2BL(1,JN),NPPP,2,SYM,EOP)
      CALL LRCURV(ZBL,RBL,0,1,SYM,1.0)
      CALL LRCHSZ(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,NOSTAT,NSL,NLOSS,
     2    LSFR,LTPL,LAMVT,LROT,LBLAD,LETEAN,ANGROT,IMESH,ISLINE,
     3    ISTATL,IPLOT,ISUPER,ITSON,IDEBUG,ZOMIN,ZOMBI,ZOMBO,ZOMOUT,
     4    ROMIN,ROMBI,ROMBO,ROMOUT,ZHIN,ZTIN,ZHOUT,ZTOUT,RHIN,RTIN,RHOUT,
     5    RTOUT,TITLEI(20),ZHUB(50),RHUB(50),ZTIP(50),RTIP(50),SFIN(50),
     6    RADIN(50),TIP(50),PRIP(50),LAMIN(50),VTHIN(50),SFOUT(50),
     7    RADOUT(50),PROP(50),LOSOUT(50),LAMOUT(50),VTHOUT(50),
     8    BETALE(50),BETATE(50),ZHST(50),ZTST(50),RHST(50),RTST(50),
     9    FLFR(50),PERCRD(50),PERLOS(50),ZBL(50,50),RBL(50,50),
     1    THBL(50,50),TNBL(50,50),TTBL(50,50),TH1BL(50,50),TH2BL(50,50)
      COMMON/CALCON/MMM1,MHTP1,CP,EXPON,TGROG,PITCH,RLEH,RLET,RTEH,RTET,
     1    ZLE(50),RLE(50),ZTE(50),RTE(50),ZLEOM(101),RLEOM(101),
     2    SLEOM(101),THLEOM(101),ZTEOM(101),RTEOM(101),STEOM(101),
     3    THTEOM(101),ILE(101),ITE(101),ZOM(100,101),ROM(100,101),
     4    SOM(100,101),TOM(100,101),BTH(100,101),DTHDS(100,101),
     5    DTHDT(100,101),PLOSS(100,101),CPHI(100,101),SPHI(100,101)
      COMMON/ROTATN/ZHROT(50),RHROT(50),ZTROT(50),RTROT(50),
     1    ZLEOMR(101),RLEOMR(101),ZTEOMR(101),RTEOMR(101),
     2    ZBLROT(50,50),RBLROT(50,50),ZOMROT(100,101),ROMROT(100,101)
      DIMENSION ZRAD(50,101),RRAD(50,101),Z1(100),R1(100),
     1    ZNOR(2),RNOR(2),SLOM(100),AAA(100),BBB(100)
C
C--ROTATE HUB AND TIP COORDINATES
      CALL ROTATE(ANGROT,ZHUB,RHUB,NHUB,1,50,1,ZHROT,RHROT)
      CALL ROTATE(ANGROT,ZTIP,RTIP,NTIP,1,50,1,ZTROT,RTROT)
C
C--DIVIDE HUB AND TIP CONTOURS
      NMAX = MAX0(NHUB,NTIP)
      IF (NHUB.EQ.NTIP) GO TO 8
      IF (NHUB.EQ.NMAX) GO TO 4
      DELH = (ZHROT(NHUB)-ZHROT(1))/FLOAT(NMAX-1)
      DO 2 I=1,NMAX
      ZRAD(I,MHTP1) = ZTROT(I)
      RRAD(I,MHTP1) = RTROT(I)
    2 ZRAD(I,1) = ZHROT(1)+FLOAT(I-1)*DELH
      CALL SPLINT(ZHROT,RHROT,NHUB,ZRAD(1,1),NMAX,RRAD(1,1),AAA,BBB)
      GO TO 15
    4 DELT = (ZTROT(NTIP)-ZTROT(1))/FLOAT(NMAX-1)
      DO 6 I=1,NMAX
      ZRAD(I,1) = ZHROT(I)
      RRAD(I,1) = RHROT(I)
    6 ZRAD(I,MHTP1) = ZTROT(1)+FLOAT(I-1)*DELT
      CALL SPLINT(ZTROT,RTROT,NTIP,ZRAD(1,MHTP1),NMAX,RRAD(1,MHTP1),AAA,
     1BBB)
      GO TO 15
    8 DO 10 I=1,NMAX
      ZRAD(I,1) = ZHROT(I)
      RRAD(I,1) = RHROT(I)
      ZRAD(I,MHTP1) = ZTROT(I)
   10 RRAD(I,MHTP1) = RTROT(I)
   15 CONTINUE
C
C--FILL ZRAD AND RRAD ARRAYS FROM HUB TO TIP
```

```
      DO 20 I=1,NMAX
      DELZ = (ZRAD(I,MHTP1)-ZRAD(I,1))/FLOAT(MHT)
      DELR = (RRAD(I,MHTP1)-RRAD(I,1))/FLOAT(MHT)
      DO 20 J=2,MHT
      ZRAD(I,J) = ZRAD(I,J-1)+DELZ
   20 RRAD(I,J) = RRAD(I,J-1)+DELR
C
C--ROTATE INPUT MESH BOUNDARIES
      CAN = COS(ANGROT)
      SAN = SIN(ANGROT)
      ZOMINR = ZOMIN*CAN+ROMIN*SAN
      ZOMBIR = ZOMBI*CAN+ROMBI*SAN
      ZOMBOR = ZOMBO*CAN+ROMBO*SAN
      ZOMOUR = ZOMOUT*CAN+ROMOUT*SAN
C
C--COMPUTE ZOMROT ON HUB
      ZOMROT(1,1) = ZOMINR
      IF (MBI.EQ.0) GO TO 50
      MBIM1 = MBI-1
      DELZ= (ZOMBIR-ZOMINR)/FLOAT(MBIM1)
      DO 30 I=2,MBI
   30 ZOMROT(I,1)= ZOMROT(I-1,1)+DELZ
      DELZ = (ZOMBOR-ZOMBIR)/FLOAT(MBO-MBI)
      MBIP1 = MBI+1
      DO 40 I=MBIP1,MBO
   40 ZOMROT(I,1)= ZOMROT(I-1,1)+DELZ
      DELZ= (ZOMOUR-ZOMBOR)/FLOAT(MM-MBO)
      MBOP1 = MBO+1
   50 IF (MBI.EQ.0) DELZ = (ZOMOUR-ZOMINR)/FLOAT(MMM1)
      IF (MBI.EQ.0) MBOP1=2
      DO 60 I=MBOP1,MM
   60 ZOMROT(I,1)= ZOMROT(I-1,1)+DELZ
C
C--COMPUTE ROMROT AND SLOPE ON HUB
      CALL SPLINT(ZRAD(1,1),RRAD(1,1),NMAX,ZOMROT(1,1),MM,ROMROT(1,1),
     1SLOM,BBB)
      DO 70 I=1,MM
      PHI = ATAN(SLOM(I))+ANGROT
      SPHI(I,1) = SIN(PHI)
   70 CPHI(I,1) = COS(PHI)
C
C--COMPUTE ZOMROT AND ROMROT  ROW BY ROW FROM HUB TO TIP
C
      DO 100 J=2,MHTP1
C
C--MOVE ALONG PRESENT ROW, ONE POINT AT A TIME, LOCATING
C--COORDINATES OF INTERSECTIONS OF LINES NORMAL TO PREVIOUS ROW
      DO 80 I=1,MM
C
C--CALCULATE POINTS ON STRAIGHT LINE NORMAL TO PREVIOUS ROW
      RNOR(2) = ROMROT(I,J-1)
      ZNOR(2) = ZOMROT(I,J-1)
      RNOR(1) = RNOR(2)-DELR
      ZNOR(1) = ZNOR(2)+SLOM(I)*DELR
C
C--LOCATE INTERSECTION OF LINE NORMAL TO PREVIOUS ROW WITH PRESENT ROW
   80 CALL INRSCT(ZRAD(1,J),RRAD(1,J),NMAX,ZNOR,RNOR,2,Z1(I),R1(I))
C
C--CALCULATE SLOPES OF PRESENT ROW AT INTERSECTION POINTS
```

```
      CALL SPLINT(ZRAD(1,J),RRAD(1,J),NMAX,Z1,MM,AAA,SLOM,BBB)
C
C--CALCULATE INTERSECTIONS OF LINE NORMAL TO PRESENT ROW WITH
C--PRESENT ROW
      DO 90 I=1,MM
      Z2 = (ZOMROT(I,J-1)+(ROMROT(I,J-1)-R1(I))*SLOM(I)+Z1(I)*SLOM(I)**2
     1)/(1.+SLOM(I)**2)
   90 ZOMROT(I,J) = (Z1(I)+Z2)/2.
C
C--CALCULATE ROMROT AND SLOPES AND ANGLES ON PRESENT ROW
      CALL SPLENT(ZOMROT(1,J),MM,ROMROT(1,J),SLOM,BBB)
      DO 100 I=1,MM
      PHI = ATAN(SLOM(I))+ANGROT
      SPHI(I,J) = SIN(PHI)
  100 CPHI(I,J) = COS(PHI)
C
C--UNROTATE ZOMROT AND ROMROT TO GET ZOM AND ROM
C
      CALL ROTATE(-ANGROT,ZOMROT,ROMROT,MM,MHTP1,100,101,ZOM,ROM)
      RETURN
      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,NLOSS,
     2   LSFR,LTPL,LAMVT,LROT,LBLAD,LETEAN,ANGROT,IMESH,ISLINE,
     3   ISTATL,IPLOT,ISUPER,ITSON,IDEBUG,ZOMIN,ZOMBI,ZOMBO,ZOMOUT,
     4   ROMIN,ROMBI,ROMBO,ROMOUT,ZHIN,ZTIN,ZHOUT,ZTOUT,RHIN,RTIN,RHOUT,
     5   RTOUT,TITLEI(20),ZHUB(50),RHUB(50),ZTIP(50),RTIP(50),SFIN(50),
     6   RADIN(50),TIP(50),PRIP(50),LAMIN(50),VTHIN(50),SFOUT(50),
     7   RADOUT(50),PROP(50),LOSOUT(50),LAMOUT(50),VTHOUT(50),
     8   BETALE(50),BETATE(50),ZHST(50),ZTST(50),RHST(50),RTST(50),
     9   FLFR(50),PERCRD(50),PERLOS(50),ZBL(50,50),RBL(50,50),
     1   THBL(50,50),TNBL(50,50),TTBL(50,50),TH1BL(50,50),TH2BL(50,50)
      COMMON/CALCON/MMM1,MHTP1,CP,EXPON,TGROG,PITCH,RLEH,RLET,RTEH,RTET,
     1   ZLE(50),RLE(50),ZTE(50),RTE(50),ZLEOM(101),RLEOM(101),
     2   SLEOM(101),THLEOM(101),ZTEOM(101),RTEOM(101),STEOM(101),
     3   THTEOM(101),ILE(101),ITE(101),ZOM(100,101),ROM(100,101),
     4   SOM(100,101),TOM(100,101),BTH(100,101),DTHDS(100,101),
     5   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','DE B','OUND','ARIE'
     1,'S$C1','$R8I','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)
      IPLT= 1
   10 EOP = 0.
      IF (IPLT.EQ.1.OR.IPLT.EQ.3) CALL LRGRID(-1,-1,1.0,1.0)
      IF (IPLT.EQ.2.OR.IPLT.EQ.4) CALL LRGRID(3,3,11.0,11.0)
      CALL LRCHSZ(4)
      IF (IPLT.EQ.1.OR.IPLT.EQ.2) CALL LRLEGN(TITL1,60,0,1.3,0.7,0.0)
      IF (IPLT.EQ.3.OR.IPLT.EQ.4) 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)
      IF(MBI.EQ.0) GO TO 12
      CALL LRCURV(ZLPLT,RLPLT,100,2,SYM,0.0)
      CALL LRCURV(ZTPLT,RTPLT,100,2,SYM,0.0)
   12 IF (IPLT.GT.2) GO TO 20
      IF(MBI.EQ.0) EOP = 1.
      CALL LRCURV(ZHUB,RHUB,NHUB,4,SYM,0.0)
      CALL LRCURV(ZTIP,RTIP,NTIP,4,SYM,EOP)
      IF(MBI.EQ.0) GO TO 18
      DO 15 JN=1,NBLPL
   15 CALL LRCURV(ZBL(1,JN),RBL(1,JN),NPPP,2,SYM,0.0)
      CALL ROTATE(-ANGROT,ZLE,RLE,NBLPL,1,50,1,ZTEM,RTEM)
      CALL LRCURV(ZTEM,RTEM,NBLPL,3,SYN,0.0)
      CALL ROTATE(-ANGROT,ZTE,RTE,NBLPL,1,50,1,ZTEM,RTEM)
      CALL LRCURV(ZTEM,RTEM,NBLPL,3,SYN,1.0)
   18 IPLT = IPLT+1
      GO TO 10
C--PLOT VERTICAL MESH LINES
   20 DO 40 I=1,MM
      DO 30 J=1,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)
      IPLT = IPLT+1
      IF (IPLT.LE.4) GO TO 10
      CALL LRCURV(ZTEM,RTEM,0,1,SYM,1.0)
      CALL LRCHSZ(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,NOSTAT,NSL,NLOSS,
     2   LSFR,LTPL,LAMVT,LROT,LBLAD,LETEAN,ANGROT,IMESH,ISLINE,
     3   ISTATL,IPLOT,ISUPER,ITSON,IDEBUG,ZOMIN,ZOMBI,ZOMBO,ZOMOUT,
     4   ROMIN,ROMBI,ROMBO,ROMOUT,ZHIN,ZTIN,ZHOUT,ZTOUT,RHIN,RTIN,RHOUT,
     5   RTOUT,TITLEI(20),ZHUB(50),RHUB(50),ZTIP(50),RTIP(50),SFIN(50),
     6   RADIN(50),TIP(50),PRIP(50),LAMIN(50),VTHIN(50),SFOUT(50),
     7   RADOUT(50),PROP(50),LOSOUT(50),LAMOUT(50),VTHOUT(50),
     8   BETALE(50),BETATE(50),ZHST(50),ZTST(50),RHST(50),RTST(50),
     9   FLFR(50),PERCRD(50),PERLOS(50),ZBL(50,50),RBL(50,50),
     1   THBL(50,50),TNBL(50,50),TTBL(50,50),TH1BL(50,50),TH2BL(50,50)
      COMMON/CALCON/MMM1,MHTP1,CP,EXPON,TGROG,PITCH,RLEH,RLET,RTEH,RTET,
     1   ZLE(50),RLE(50),ZTE(50),RTE(50),ZLEOM(101),RLEOM(101),
     2   SLEOM(101),THLEOM(101),ZTEOM(101),RTEOM(101),STEOM(101),
     3   THTEOM(101),ILE(101),ITE(101),ZOM(100,101),ROM(100,101),
     4   SOM(100,101),TOM(100,101),BTH(100,101),DTHDS(100,101),
     5   DTHDT(100,101),PLOSS(100,101),CPHI(100,101),SPHI(100,101)
      COMMON/ROTATN/ZHROT(50),RHROT(50),ZTROT(50),RTROT(50),
     1   ZLEOMR(101),RLEOMR(101),ZTEOMR(101),RTEOMR(101),
     2   ZBLROT(50,50),RBLROT(50,50),ZOMROT(100,101),ROMROT(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),BBB(100)
C
C--OBTAIN PLOT POINTS ON HUB
C
      DELZ = (ZHROT(NHUB)-ZHROT(1))/99.
      ZHPLT(1) = ZHROT(1)
      DO 10 I=2,100
   10 ZHPLT(I) = ZHPLT(I-1)+DELZ
      CALL SPLINT(ZHROT,RHROT,NHUB,ZHPLT,100,RHPLT,AAA,BBB)
      CALL ROTATE(-ANGROT,ZHPLT,RHPLT,100,1,100,1,ZHPLT,RHPLT)
C
C--OBTAIN PLOT POINTS ON SHROUD
C
      DELZ = (ZTROT(NTIP)-ZTROT(1))/99.
      ZSPLT(1) = ZTROT(1)
      DO 20 I=2,100
   20 ZSPLT(I) = ZSPLT(I-1)+DELZ
      CALL SPLINT(ZTROT,RTROT,NTIP,ZSPLT,100,RSPLT,AAA,BBB)
      CALL ROTATE(-ANGROT,ZSPLT,RSPLT,100,1,100,1,ZSPLT,RSPLT)
      IF (MBI.EQ.0) GO TO 50
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,BBB)
      CALL ROTATE(-ANGROT,ZLPLT,RLPLT,100,1,100,1,ZLPLT,RLPLT)
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,BBB)
      CALL ROTATE(-ANGROT,ZTPLT,RTPLT,100,1,100,1,ZTPLT,RTPLT)
C
C--CALCULATE THE RANGE OF THE PLOT
C
   50 ZLRNG = ZHUB(1)
      ZRRNG = ZHUB(NHUB)
      RBRNG = RHUB(1)
      RTRNG = RTIP(1)
C--CHECK HUB AND TIP
      DO 60 I=1,NHUB
      ZLRNG = AMIN1(ZLRNG,ZHUB(I))
      ZRRNG = AMAX1(ZRRNG,ZHUB(I))
      RBRNG = AMIN1(RBRNG,RHUB(I))
   60 RTRNG = AMAX1(RTRNG,RHUB(I))
      DO 70 I=1,NTIP
      ZLRNG = AMIN1(ZLRNG,ZTIP(I))
      ZRRNG = AMAX1(ZRRNG,ZTIP(I))
      RBRNG = AMIN1(RBRNG,RTIP(I))
   70 RTRNG = AMAX1(RTRNG,RTIP(I))
C--CHECK INLET AND OUTLET MESH BOUNDARIES
      DO 80 J=1,MHTP1
      ZLRNG = AMIN1(ZLRNG,ZOM(1,J))
      ZLRNG = AMIN1(ZLRNG,ZOM(MM,J))
      ZRRNG = AMAX1(ZRRNG,ZOM(1,J))
      ZRRNG = AMAX1(ZRRNG,ZOM(MM,J))
      RBRNG = AMIN1(RBRNG,ROM(1,J))
      RBRNG = AMIN1(RBRNG,ROM(MM,J))
      RTRNG = AMAX1(RTRNG,ROM(1,J))
   80 RTRNG = AMAX1(RTRNG,ROM(MM,J))
C--CHECK HUB AND TIP MESH BOUNDARIES
      DO 90 I=1,MM
      ZLRNG = AMIN1(ZLRNG,ZOM(I,1))
      ZLRNG = AMIN1(ZLRNG,ZOM(I,MHTP1))
      ZRRNG = AMAX1(ZRRNG,ZOM(I,1))
      ZRRNG = AMAX1(ZRRNG,ZOM(I,MHTP1))
      RBRNG = AMIN1(RBRNG,ROM(I,1))
      RBRNG = AMIN1(RBRNG,ROM(I,MHTP1))
      RTRNG = AMAX1(RTRNG,ROM(I,1))
   90 RTRNG = AMAX1(RTRNG,ROM(I,MHTP1))
C--CHECK FIRST AND LAST INPUT BLADE SECTIONS
      IF (MBI.EQ.0) GO TO 110
      DO 100 I=1,NPPP
      ZLRNG = AMIN1(ZLRNG,ZBL(I,1))
      ZLRNG = AMIN1(ZLRNG,ZBL(I,NBLPL))
      ZRRNG = AMAX1(ZRRNG,ZBL(I,1))
      ZRRNG = AMAX1(ZRRNG,ZBL(I,NBLPL))
      RBRNG = AMIN1(RBRNG,RBL(I,1))
      RBRNG = AMIN1(RBRNG,RBL(I,NBLPL))
      RTRNG = AMAX1(RTRNG,RBL(I,1))
  100 RTRNG = AMAX1(RTRNG,RBL(I,NBLPL))
  110 DELZ = ZRRNG-ZLRNG
```

```
      ZLRNG = ZLRNG-0.05*DELZ
      ZRRNG = ZRRNG+0.05*DELZ
      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 120
      RTRNG = RTRNG+DMD2
      RBRNG = RBRNG-DMD2
      RETURN
  120 ZRRNG = ZRRNG+DMD2
      ZLRNG = ZLRNG-DMD2
      RETURN
      END




      SUBROUTINE PRECAL
C
C--PRECAL CALCULATES MANY OF THE REQUIRED FIXED CONSTANTS
C
      COMMON NREAD,NWRIT,ITER,IEND,NWRT1,NWRT2,NWRT3,NWRT4,NWRT5,NWRT6
      COMMON/INPUTT/GAM,AR,MSFL,OMEGA,REDFAC,VELTOL,FNEW,DNEW,MBI,MBO,
     1   MM,MHT,NBL,NHUB,NTIP,NIN,NOUT,NBLPL,NPPP,NOSTAT,NSL,NLOSS,
     2   LSFR,LTPL,LAMVT,LROT,LBLAD,LETEAN,ANGROT,IMESH,ISLINE,
     3   ISTATL,IPLOT,ISUPER,ITSON,IDEBUG,ZOMIN,ZOMBI,ZOMBO,ZOMOUT,
     4   ROMIN,ROMBI,ROMBO,ROMOUT,ZHIN,ZTIN,ZHOUT,ZTOUT,RHIN,RTIN,RHOUT,
     5   RTOUT,TITLEI(20),ZHUB(50),RHUB(50),ZTIP(50),RTIP(50),SFIN(50),
     6   RADIN(50),TIP(50),PRIP(50),LAMIN(50),VTHIN(50),SFOUT(50),
     7   RADOUT(50),PROP(50),LOSOUT(50),LAMOUT(50),VTHOUT(50),
     8   BETALE(50),BETATE(50),ZHST(50),ZTST(50),RHST(50),RTST(50),
     9   FLFR(50),PERCRD(50),PERLOS(50),ZBL(50,50),RBL(50,50),
     1   THBL(50,50),TNBL(50,50),TTBL(50,50),TH1BL(50,50),TH2BL(50,50)
      COMMON/CALCON/MMM1,MHTP1,CP,EXPON,TGROG,PITCH,RLEH,RLET,RTEH,RTET,
     1   ZLE(50),RLE(50),ZTE(50),RTE(50),ZLEOM(101),RLEOM(101),
     2   SLEOM(101),THLEOM(101),ZTEOM(101),RTEOM(101),STEOM(101),
     3   THTEOM(101),ILE(101),ITE(101),ZOM(100,101),ROM(100,101),
     4   SOM(100,101),TOM(100,101),BTH(100,101),DTHDS(100,101),
     5   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),FT(100,101),DFDM(100,101),
     6   XIOM(100,101),ZETOM(100,101),DLDU(100,101)
      COMMON/ROTATN/ZHROT(50),RHROT(50),ZTROT(50),RTROT(50),
     1   ZLEOMR(101),RLEOMR(101),ZTEOMR(101),RTEOMR(101),
     2   ZBLROT(50,50),RBLROT(50,50),ZOMROT(100,101),ROMROT(100,101)
      DIMENSION DYDX(101),D2YDX2(101),TTEM(50),DOM(101),DIP(50),
     1   SZRBL(50),BLOCK(50)
      REAL MSFL,LAMIN,LAMOUT
C
C--INITIALIZE TIPF, RHOIPF, LAMDAF, AND RVTHTA
```

```
C
      CALL LAMNIT
      CALL RVTNIT
      IF (GAM.NE.0.) CALL TIPNIT
      IF (GAM.NE.0.) CALL RHINIT
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).LE.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).GE.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--ROTATE HUB AND SHROUD STATION LINE LOCATION ARRAYS
      IF (NOSTAT.EQ.0) GO TO 101
      CALL ROTATE(ANGROT,ZHST,RHST,NOSTAT,1,50,1,ZHST,RHST)
      CALL ROTATE(ANGROT,ZTST,RTST,NOSTAT,1,50,1,ZTST,RTST)
C
C--ROTATE ZBL AND RBL TO GET ZBLROT AND RBLROT
  101 CALL ROTATE(ANGROT,ZBL,RBL,NPPP,NBLPL,50,50,ZBLROT,RBLROT)
C
```

```
C--SET ILE AND ITE ARRAYS WHEN THERE ARE NO BLADES
C
      IF (MBI.NE.0) GO TO 104
      DO 102 J=1,MHTP1
      ILE(J) = MM+1
  102 ITE(J) = MM+1
      GO TO 225
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
  104 DO 110 JN=1,NBLPL
      ZLE(JN) = ZBLROT(1,JN)
  110 RLE(JN) = RBLROT(1,JN)
      CALL INRSCT(ZHROT,RHROT,NHUB,ZLE,RLE,NBLPL,ZLEH,RLEH)
      CALL INRSCT(ZTROT,RTROT,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) = ZBLROT(NPPP,JN)
  120 RTE(JN) = RBLROT(NPPP,JN)
      CALL INRSCT(ZHROT,RHROT,NHUB,ZTE,RTE,NBLPL,ZTEH,RTEH)
      CALL INRSCT(ZTROT,RTROT,NTIP,ZTE,RTE,NBLPL,ZTET,RTET)
C
C--CALCULATE ORTHOGONAL MESH ARRAYS AT THE LEADING EDGE
C--ZLEOM(AND ZLEOMR), RLEOM(AND RLEOMR), SLEOM, AND THLEOM
C--CALCULATE ILE ARRAY OF MESH POINT LOCATIONS INSIDE BLADE
C--LEADING EDGE
C
      ZLEOMR(1) = ZLEH
      RLEOMR(1) = RLEH
      ZLEOMR(MHTP1) = ZLET
      RLEOMR(MHTP1) = RLET
      DO 130 J=2,MHT
  130 CALL INRSCT(ZOMROT(1,J),ROMROT(1,J),MM,ZLE,RLE,NBLPL,ZLEOMR(J),
     1RLEOMR(J))
      DO 160 J=1,MHTP1
      DO 140 I=1,MM
      IF (ZLEOMR(J).LE.ZOMROT(I,J)) GO TO 150
  140 CONTINUE
  150 ILE(J) = I
  160 SLEOM(J) = SOM(I-1,J)+SQRT((ZLEOMR(J)-ZOMROT(I-1,J))**2
     1+(RLEOMR(J)-ROMROT(I-1,J))**2)
      DO 170 JN=1,NBLPL
  170 TTEM(JN) = THBL(1,JN)
      DOM(1) = 0.
      DO 174 J=2,MHTP1
  174 DOM(J) = DOM(J-1)+SQRT((ZLEOMR(J)-ZLEOMR(J-1))**2
     1+(RLEOMR(J)-RLEOMR(J-1))**2)
      DIP(1) = 0.
      DO 176 JN=2,NBLPL
  176 DIP(JN) = DIP(JN-1)+SQRT((ZLE(JN)-ZLE(JN-1))**2+
     1(RLE(JN)-RLE(JN-1))**2)
      CALL SPLINT(DIP,TTEM,NBLPL,DOM,MHTP1,THLEOM,DYDX,D2YDX2)
      CALL ROTATE(-ANGROT,ZLEOMR,RLEOMR,MHTP1,1,101,1,ZLEOM,RLEOM)
C
C--CALCULATE ORTHOGONAL MESH ARRAYS AT THE TRAILING EDGE
```

```
C--ZTEOM(AND ZTEOMR), RTEOM(AND RTEOMR), STEOM, AND THTEOM
C--CALCULATE ITE ARRAY OF MESH POINT LOCATIONS INSIDE BLADE
C--TRAILING EDGE
C
      ZTEOMR(1) = ZTEH
      RTEOMR(1) = RTEH
      ZTEOMR(MHTP1) = ZTET
      RTEOMR(MHTP1) = RTET
      DO 180 J=2,MHT
  180 CALL INRSCT(ZOMROT(1,J),ROMROT(1,J),MM,ZTE,RTE,NBLPL,ZTEOMR(J),
     1RTEOMR(J))
      DO 210 J=1,MHTP1
      ILEJ = ILE(J)-1
      DO 190 I=ILEJ,MM
      IF (ZTEOMR(J).LT.ZOMROT(I,J)) GO TO 200
  190 CONTINUE
  200 ITE(J) = I-1
      ITEJ = I-1
  210 STEOM(J) = SOM(ITEJ,J)+SQRT((ZTEOMR(J)-ZOMROT(ITEJ,J))**2
     1+(RTEOMR(J)-ROMROT(ITEJ,J))**2)
      DO 220 JN=1,NBLPL
  220 TTEM(JN) = THBL(NPPP,JN)
      DOM(1) = 0.
      DO 222 J=2,MHTP1
  222 DOM(J) = DOM(J-1)+SQRT((ZTEOMR(J)-ZTEOMR(J-1))**2
     1+(RTEOMR(J)-RTEOMR(J-1))**2)
      DIP(1) = 0.
      DO 223 JN=2,NBLPL
  223 DIP(JN) = DIP(JN-1)+SQRT((ZTE(JN)-ZTE(JN-1))**2+
     1(RTE(JN)-RTE(JN-1))**2)
      CALL SPLINT(DIP TTEM,NBLPL,DOM,MHTP1,THTEOM,DYDX,D2YDX2)
      CALL ROTATE(-ANGROT,ZTEOMR,RTEOMR,MHTP1,1,101,1,ZTEOM,RTEOM)
C
C--PRINT BLADE GEOMETRY ON INPUT PLANES
C
      WRITE (NWRIT,1000)
      DO 224 JN=1,NBLPL
      SZRBL(1) = 0.
      BLOCK(1) = TTBL(1,JN)/PITCH
      DO 226 IN=2,NPPP
      SZRBL(IN) = SZRBL(IN-1)+SQRT((ZBL(IN,JN)-ZBL(IN-1,JN))**2
     1+(RBL(IN,JN)-RBL(IN-1,JN))**2)
  226 BLOCK(IN) = TTBL(IN,JN)/PITCH
      IF(LETEAN.EQ.1) GO TO 232
      CALL SPLINE (SZRBL,THBL(1,JN),NPPP,DYDX,D2YDX2)
      GO TO 234
  232 DTHDSL = TAN(BETALE(JN)/57.295780)/RBL(1,JN)
      DTHDST = TAN(BETATE(JN)/57.295780)/RBL(NPPP,JN)
      CALL SPLISL(SZRBL,THBL(1,JN),NPPP,DTHDSL,DTHDST,DYDX,D2YDX2)
  234 WRITE (NWRIT,1010) JN,(IN,ZBL(IN,JN),RBL(IN,JN),THBL(IN,JN),
     1TTBL(IN,JN),BLOCK(IN),SZRBL(IN),DYDX(IN),D2YDX2(IN),IN=1,NPPP)
  224 CONTINUE
      WRITE(NWRIT,1040)
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--REDUCE MASSFLOW, WHEEL SPEED, AND WHIRL FOR REDUCED FLOW SOLUTION
C
  225 IF (REDFAC.EQ.1.0) GO TO 260
      OMEGA = OMEGA*REDFAC
      MSFL = MSFL*REDFAC
      DO 230 J=1,NIN
      LAMIN(J)  = LAMIN(J)*REDFAC
  230 VTHIN(J)  = VTHIN(J)*REDFAC
      IF (MBI.EQ.0) GO TO 250
      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 RVTNIT
  250 CALL LAMNIT
C
C--PRINT DEBUG OUTPUT
C
  260 IF (IDEBUG.LE.0) GO TO 270
      WRITE(NWRIT,1020)
      WRITE(NWRIT,1030) ((I,J,SOM(I,J),TOM(I,J),BTH(I,J),DTHDT(I,J),
     1CPHI(I,J),SPHI(I,J),I=1,MM),J=1,MHTP1)
      WRITE(NWRIT,1040)
C
C--FOR INCOMPRESSIBLE FLOW, INITIALIZE THE DENSITY (RHO ARRAY)
C
  270 IF (GAM.NE.0.) RETURN
      DO 280 J=1,MHTP1
      DO 280 I=1,MM
  280 RHO(I,J)  = AR
      RETURN
C
C--FORMAT STATEMENTS
C
 1000 FORMAT (1H1//42X,48(1H*)/42X,48H*    BLADE GEOMETRY ON INPUT BLADE
     1 SECTIONS    */42X,48(1H*)//)
 1010 FORMAT (///4X,28H**  INPUT BLADE PLANE NUMBER,I3,4H  **//2X,2HIN,
     16X,3HZBL,13X,3HRBL,13X,4HTHBL,12X,4HTTBL,10X,8HBLOCKAGE,10X,
     25HSZRBL,10X,6HDTHDSP,10X,7HDTHDSP2/(2X,I2,8G16.7))
 1020 FORMAT (1H1//35X,57(1H*)/35X,57H*    CONSTANT  QUANTITIES  ON  THE
     1  ORTHOGONAL  MESH    */35X,57(1H*)//4X,1HI,4X,1HJ,6X,3HSOM,12X,
     23HTOM,12X,3HBTH,11X,5HDTHDT,10X,4HCPHI,11X,4HSPHI)
 1030 FORMAT (2I5,6G15.5)
 1040 FORMAT (1H1)
      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 NREAD,NWRIT,ITER,IEND,NWRT1,NWRT2,NWRT3,NWRT4,NWRT5,NWRT6
      COMMON/INPUTT/GAM,AR,MSFL,OMEGA,REDFAC,VELTOL,FNEW,DNEW,MBI,MBO,
     1    MM,MHT,NBL,NHUB,NTIP,NIN,NOUT,NBLPL,NPPP,NOSTAT,NSL,NLOSS,
     2    LSFR,LTPL,LAMVT,LROT,LBLAD,LETEAN,ANGROT,IMESH,ISLINE,
     3    ISTATL,IPLOT,ISUPER,ITSON,IDEBUG,ZOMIN,ZOMBI,ZOMBO,ZOMOUT,
     4    ROMIN,ROMBI,ROMBO,ROMOUT,ZHIN,ZTIN,ZHOUT,ZTOUT,RHIN,RTIN,RHOUT,
     5    RTOUT,TITLEI(20),ZHUB(50),RHUB(50),ZTIP(50),RTIP(50),SFIN(50),
     6    RADIN(50),TIP(50),PRIP(50),LAMIN(50),VTHIN(50),SFOUT(50),
     7    RADOUT(50),PROP(50),LOSOUT(50),LAMOUT(50),VTHOUT(50),
     8    BETALE(50),BETATE(50),ZHST(50),ZTST(50),RHST(50),RTST(50),
     9    FLFR(50),PERCRD(50),PERLOS(50),ZBL(50,50),RBL(50,50),
     1    THBL(50,50),TNBL(50,50),TTBL(50,50),TH1BL(50,50),TH2BL(50,50)
      COMMON/CALCON/MMM1,MHTP1,CP,EXPON,TGROG,PITCH,RLEH,RLET,RTEH,RTET,
     1    ZLE(50),RLE(50),ZTE(50),RTE(50),ZLEOM(101),RLEOM(101),
     2    SLEOM(101),THLEOM(101),ZTEOM(101),RTEOM(101),STEOM(101),
     3    THTEOM(101),ILE(101),ITE(101),ZOM(100,101),ROM(100,101),
     4    SOM(100,101),TOM(100,101),BTH(100,101),DTHDS(100,101),
     5    DTHDT(100,101),PLOSS(100,101),CPHI(100,101),SPHI(100,101)
      COMMON/ROTATN/ZHROT(50),RHROT(50),ZTROT(50),RTROT(50),
     1    ZLEOMR(101),RLEOMR(101),ZTEOMR(101),RTEOMR(101),
     2    ZBLROT(50,50),RBLROT(50,50),ZOMROT(100,101),ROMROT(100,101)
      COMMON/INDCOM/NBLPC,NPPC,ZPC(51,51),RPC(51,51),TTPC(51,51),
     1    THPC(51,51),DTHDZ(51,51),DTHDR(51,51),BTHLE(101),BTHTE(101),
     2    BTBFLE(101),BTBFTE(101)
      DIMENSION ANGZ(51,51),ANGR(51,51),DTHDSP(51,51),DTHDTP(51,51),
     1    SZRBL(51),SZRPC(51),ZPCT1(51),RPCT1(51),THPCT1(51),TTPCT1(51),
     2    ANGT1(51),DTST1(51),ZPCT2(51),RPCT2(51),THPCT2(51),TTPCT2(51),
     3    ANGT2(51),DTST2(51),DYDX(51),D2YDX2(51)
      NBLPC = 21
      NPPC = 21
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 FIVE PERCENT CHORD INTERVALS
    5 DO 60 JN=1,NBLPL
      DELZ= (ZBLROT(NPPP,JN)-ZBLROT(1,JN))/FLOAT(NPPC-1)
      ZPC(1,JN)= ZBLROT(1,JN)
      DO 10 KN=2,NPPC
   10 ZPC(KN,JN)= ZPC(KN-1,JN)+DELZ
C
C--CALCULATE R COORDINATES AND ANGLES WITH RESPECT TO Z AXIS AT
C--INTERSECTION POINTS
      CALL SPLINT(ZBLROT(1,JN),RBLROT(1,JN),NPPP,ZPC(1,JN),NPPC,
     1RPC(1,JN),ANGZ(1,JN),D2YDX2)
      DO 20 KN=1,NPPC
   20 ANGZ(KN,JN) = ATAN(ANGZ(KN,JN))+ANGROT
C
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
C--CALCULATE ARC LENGTH ALONG INPUT LINES USING POINTS AT FIVE
C--PERCENT OF CHORD
      SZRPC(1)= 0.
      DO 40 KN=2,NPPC
```

```
   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
C--CALCULATE THETA AND CHANGE OF THETA WITH ARC LENGTH ALONG INPUT LINES
      IF (LETEAN.EQ.1) GO TO 50
      CALL SPLINT(SZRBL,THBL(1,JN),NPPP,SZRPC,NPPC,THPC(1,JN),
     1DTHDSP(1,JN),D2YDX2)
      GO TO 55
   50 DTHDSL = TAN(BETALE(JN)/57.295780)/RBL(1,JN)
      DTHDST = TAN(BETATE(JN)/57.295780)/RBL(NPPP,JN)
      CALL SPINSL(SZRBL,THBL(1,JN),NPPP,DTHDSL,DTHDST,SZRPC,NPPC,
     1THPC(1,JN),DTHDSP(1,JN),D2YDX2)
   55 CONTINUE
C
C--CALCULATE BLADE THICKNESS IN THETA DIRECTION AT POINTS AT
C--FIVE PERCENT OF CHORD
      CALL SPLINT(SZRBL,TTBL(1,JN),NPPP,SZRPC,NPPC,TTPC(1,JN),DYDX,
     1D2YDX2)
   60 CONTINUE
C
C--CALCULATE GRADIENT OF THETA WITH RESPECT TO DISTANCE UP FIVE PERCENT
C--CHORD ZPC,RPC LINES
C
      DO 130 KN=1,NPPC
C
C--STORE DATA AT FIVE PERCENT CHORD POINTS ON INPUT BLADE PLANES INTO
C--TEMPORARY ARRAYS FOR SPLINT CALLS
      DO 70 JN=1,NBLPL
      ZPCT1(JN)  = ZPC(KN,JN)
      RPCT1(JN)  = RPC(KN,JN)
      THPCT1(JN)  = THPC(KN,JN)
      TTPCT1(JN)  = TTPC(KN,JN)
      ANGT1(JN)  = ANGZ(KN,JN)
   70 DTST1(JN)  = DTHDSP(KN,JN)
C
C--CALCULATE ARC LENGTH UP FIVE PERCENT CHORD LINES
      DO 80 JN=2,NBLPL
   80 SZRBL(JN)  = SZRBL(JN-1)+SQRT((ZPCT1(JN)-ZPCT1(JN-1))**2
     1+(RPCT1(JN)-RPCT1(JN-1))**2)
C
C--CALCULATE POINTS ON THE ALTERNATE MESH
      DELR =  (RPC(KN,NBLPL)-RPC(KN,1))/FLOAT(NBLPC-1)
      RPCT2(1)  = RPC(KN,1)
      DO 90 LN=2,NBLPC
   90 RPCT2(LN)  = RPCT2(LN-1)+DELR
      CALL SPLINT(RPCT1,ZPCT1,NBLPL,RPCT2,NBLPC,ZPCT2,DYDX,D2YDX2)
C
C--CALCULATE ARC LENGTH AND ANGLE WITH RESPECT TO R UP FIVE PERCENT
C--CHORD LINES ON THE ALTERNATE MESH
      ANGR(KN,1)  = ATAN(DYDX(1))-ANGROT
      DO 100 LN=2,NBLPC
      SZRPC(LN)  = SZRPC(LN-1)+SQRT((RPCT2(LN)-RPCT2(LN-1))**2
     1+(ZPCT2(LN)-ZPCT2(LN-1))**2)
  100 ANGR(KN,LN)  = ATAN(DYDX(LN))-ANGROT
C
C--CALCULATE THETA AND CHANGE OF THETA WITH RESPECT TO ARC LENGTH UP
C--FIVE PERCENT CHORD LINES
      CALL SPLINT(SZRBL,THPCT1,NBLPL,SZRPC,NBLPC,THPCT2,DYDX,D2YDX2)
      DO 110 LN=1,NBLPC
  110 DTHDTP(KN,LN)  = DYDX(LN)
```

```
C
C--CALCULATE ANGLE WITH RESPECT TO Z, CHANGE OF THETA WITH ARC LENGTH
C--ALONG INPUT LINES, AND TANGENTIAL THICKNESS UP THE FIVE PERCENT
C--CHORD LINES
      CALL SPLINT(SZRBL,ANGT1,NBLPL,SZRPC,NBLPC,ANGT2,DYDX,D2YDX2)
      CALL SPLINT(SZRBL,DTST1,NBLPL,SZRPC,NBLPC,DTST2,DYDX,D2YDX2)
      CALL SPLINT(SZRBL,TTPCT1,NBLPL,SZRPC,NBLPC,TTPCT2,DYDX,D2YDX2)
C
C--STORE CALCULATED VALUES IN TWO-DIMENSIONAL ARRAYS ON THE ALTERNATE
C--MESH
      DO 120 LN=1,NBLPC
      ZPC(KN,LN)  = ZPCT2(LN)
      RPC(KN,LN)  = RPCT2(LN)
      THPC(KN,LN)  = THPCT2(LN)
      TTPC(KN,LN)  = TTPCT2(LN)
      ANGZ(KN,LN)  = ANGT2(LN)
  120 DTHDSP(KN,LN)  = DTST2(LN)
  130 CONTINUE
C
C--CALCULATE DTHDZ AND DTHDR FROM DTHDSP AND DTHDTP ON THE ALTERNATE
C--MESH
C
      DO 140 LN=1,NBLPC
      DO 140 KN=1,NPPC
      COSAB= COS(ANGZ(KN,LN)+ANGR(KN,LN))
      DTHDZ(KN,LN)=  (DTHDSP(KN,LN)*COS(ANGR(KN,LN))-DTHDTP(KN,LN)*SIN(
     1ANGZ(KN,LN)))/COSAB
  140 DTHDR(KN,LN)=  (-DTHDSP(KN,LN)*SIN(ANGR(KN,LN))+DTHDTP(KN,LN)*COS(
     1ANGZ(KN,LN)))/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 150 J=1,MHTP1
      ILEJ = ILE(J)
      ITEJ = ITE(J)
      DO 150 I=ILEJ,ITEJ
      CALL LININT(ZPC,RPC,DTHDZ,NPPC,NBLPC,51,51,ZOMROT(I,J),
     1ROMROT(I,J),DTDZOM,II,JJ)
      CALL LININT(ZPC,RPC,DTHDR,NPPC,NBLPC,51,51,ZOMROT(I,J),
     1ROMROT(I,J),DTDROM,II,JJ)
      DTHDS(I,J)  = DTDZOM*CPHI(I,J)+DTDROM*SPHI(I,J)
  150 DTHDT(I,J)  = DTDROM*CPHI(I,J)-DTDZOM*SPHI(I,J)
C
C--PRINT DEBUG BLADE GEOMETRY ON ALTERNATE MESH
      IF (IDEBUG.LE.0) RETURN
      WRITE(NWRT5,1000)
      CALL ROTATE(-ANGROT,ZPC,RPC,NPPC,NBLPC,51,51,ZPC,RPC)
      WRITE(NWRT5,1010)  ((KN,LN,ZPC(KN,LN),RPC(KN,LN),THPC(KN,LN),
     1TTPC(KN,LN),DTHDSP(KN,LN),DTHDTP(KN,LN),KN=1,NPPC),LN=1,NBLPC)
      CALL ROTATE(ANGROT,ZPC,RPC,NPPC,NBLPC,51,51,ZPC,RPC)
      WRITE(NWRT5,1020)
      RETURN
C
```

```
C--FORMAT STATEMENTS
 1000 FORMAT (1H1//25X,78(1H*)/25X,78H*    BLADE GEOMETRY ON ALTERNATE M
     1ESH      21 HUB-TO-TIP * 21 L.E.-TO-T.E.     */25X,78(1H*)///
     23X,2HKN,3X,2HLN,6X,3HZPC,13X,3HRPC,13X,4HTHPC,12X,4HTTPC,11X,
     36HDTHDSP,10X,6HDTHDTP//)
 1010 FORMAT (2I5,6G16.7)
 1020 FORMAT (1H1)
      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,NLOSS,
     2   LSFR,LTPL,LAMVT,LROT,LBLAD,LETEAN,ANGROT,IMESH,ISLINE,
     3   ISTATL,IPLOT,ISUPER,ITSON,IDEBUG,ZOMIN,ZOMBI,ZOMBO,ZOMOUT,
     4   ROMIN,ROMBI,ROMBO,ROMOUT,ZHIN,ZTIN,ZHOUT,ZTOUT,RHIN,RTIN,RHOUT,
     5   RTOUT,TITLEI(20),ZHUB(50),RHUB(50),ZTIP(50),RTIP(50),SFIN(50),
     6   RADIN(50),TIP(50),PRIP(50),LAMIN(50),VTHIN(50),SFOUT(50),
     7   RADOUT(50),PROP(50),LOSOUT(50),LAMOUT(50),VTHOUT(50),
     8   BETALE(50),BETATE(50),ZHST(50),ZTST(50),RHST(50),RTST(50),
     9   FLFR(50),PERCRD(50),PERLOS(50),ZBL(50,50),RBL(50,50),
     1   THBL(50,50),TNBL(50,50),TTBL(50,50),TH1BL(50,50),TH2BL(50,50)
      COMMON/CALCON/MMM1,MHTP1,CP,EXPON,TGROG,PITCH,RLEH,RLET,RTEH,RTET,
     1   ZLE(50),RLE(50),ZTE(50),RTE(50),ZLEOM(101),RLEOM(101),
     2   SLEOM(101),THLEOM(101),ZTEOM(101),RTEOM(101),STEOM(101),
     3   THTEOM(101),ILE(101),ITE(101),ZOM(100,101),ROM(100,101),
     4   SOM(100,101),TOM(100,101),BTH(100,101),DTHDS(100,101),
     5   DTHDT(100,101),PLOSS(100,101),CPHI(100,101),SPHI(100,101)
      COMMON/ROTATN/ZHROT(50),RHROT(50),ZTROT(50),RTROT(50),
     1   ZLEOMR(101),RLEOMR(101),ZTEOMR(101),RTEOMR(101),
     2   ZBLROT(50,50),RBLROT(50,50),ZOMROT(100,101),ROMROT(100,101)
      COMMON/INDCOM/NBLPC,NPPC,ZPC(51,51),RPC(51,51),TTPC(51,51),
     1   THPC(51,51),DTHDZ(51,51),DTHDR(51,51),BTHLE(101),BTHTE(101),
     2   BTBFLE(101),BTBFTE(101)
      DIMENSION DIST(50),DTH(50),DISEOM(101),DTHEOM(101),
     1   AAA(101),BBB(101)
C
C--CALCULATE STREAM CHANNEL THICKNESS ARRAYS AT LEADING AND TRAILING
C--EDGES,  BTHLE AND BTHTE
C
      DIST(1) = 0.
      DTH(1) = TTBL(1,1)
      DO 30 JN=2,NBLPL
      DIST(JN) = DIST(JN-1)+SQRT((ZBL(1,JN)-ZBL(1,JN-1))**2+
     1(RBL(1,JN)-RBL(1,JN-1))**2)
   30 DTH(JN) = TTBL(1,JN)
      RBLR = RBL(1,1)*COS(ANGROT)-ZBL(1,1)*SIN(ANGROT)
      DISEOM(1) = SQRT((ZLEOM(1)-ZBL(1,1))**2+(RLEOM(1)-RBL(1,1))**2)
      DISEOM(1) = SIGN(DISEOM(1),RLEOMR(1)-RBLR)
      DO 40 J=2,MHTP1
   40 DISEOM(J) = DISEOM(J-1)+SQRT((ZLEOM(J)-ZLEOM(J-1))**2+
     1(RLEOM(J)-RLEOM(J-1))**2)
      CALL SPLINT(DIST,DTH,NBLPL,DISEOM,MHTP1,DTHEOM,AAA,BBB)
      DO 50 J=1,MHTP1
```

```
   50 BTHLE(J) = PITCH-DTHEOM(J)
      DTH(1) = TTBL(NPPP,1)
      DO 60 JN=2,NBLPL
      DIST(JN) = DIST(JN-1)+SQRT((ZBL(NPPP,JN)-ZBL(NPPP,JN-1))**2+
     1(RBL(NPPP,JN)-RBL(NPPP,JN-1))**2)
   60 DTH(JN) = TTBL(NPPP,JN)
      RBLR = RBL(NPPP,1)*COS(ANGROT)-ZBL(NPPP,1)*SIN(ANGROT)
      DISEOM(1) = SQRT((ZTEOM(1)-ZBL(NPPP,1))**2+(RTEOM(1)-RBL(NPPP,1))
     1**2)
      DISEOM(1) = SIGN(DISEOM(1),RTEOMR(1)-RBLR)
      DO 70 J=2,MHTP1
   70 DISEOM(J) = DISEOM(J-1)+SQRT((ZTEOM(J)-ZTEOM(J-1))**2+
     1(RTEOM(J)-RTEOM(J-1))**2)
      CALL SPLINT(DIST,DTH,NBLPL,DISEOM,MHTP1,DTHEOM,AAA,BBB)
      DO 80 J=1,MHTP1
   80 BTHTE(J) = PITCH-DTHEOM(J)
C
C--INTERPOLATE TO OBTAIN BLADE THICKNESS IN THETA DIRECTION AT THE
C--POINTS OF THE ORTHOGONAL MESH, AND CORRECT BTH IN BLADE REGION
C
      II = 1
      JJ = 1
      DO 90 J=1,MHTP1
      ILEJ = ILE(J)
      ITEJ = ITE(J)
      DO 90 I=ILEJ,ITEJ
      CALL LININT(ZPC,RPC,TTPC,NPPC,NBLPC,51,51,ZOMROT(I,J),
     1ROMROT(I,J),DBTH,II,JJ)
      BTH(I,J) = BTH(I,J)-DBTH
   90 CONTINUE
      RETURN
      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,NLOSS,
     2    LSPR,LTPL,LAMVT,LROT,LBLAD,LETEAN,ANGROT,IMESH,ISLINE,
     3    ISTATL,IPLOT,ISUPER,ITSON,IDEBUG,ZOMIN,ZOMBI,ZOMBO,ZOMOUT,
     4    ROMIN,ROMBI,ROMBO,ROMOUT,ZHIN,ZTIN,ZHOUT,ZTOUT,RHIN,RTIN,RHOUT,
     5    RTOUT,TITLEI(20),ZHUB(50),RHUB(50),ZTIP(50),RTIP(50),SFIN(50),
     6    RADIN(50),TIP(50),PRIP(50),LAMIN(50),VTHIN(50),SFOUT(50),
     7    RADOUT(50),PROP(50),LOSOUT(50),LAMOUT(50),VTHOUT(50),
     8    BETALE(50),BETATE(50),ZHST(50),ZTST(50),RHST(50),RTST(50),
     9    FLFR(50),PERCRD(50),PERLOS(50),ZBL(50,50),RBL(50,50),
     1    THBL(50,50),TNBL(50,50),TTBL(50,50),TH1BL(50,50),TH2BL(50,50)
      COMMON/CALCON/MMM1,MHTP1,CP,EXPON,TGROG,PITCH,RLEH,RLET,RTEH,RTET,
     1    ZLE(50),RLE(50),ZTE(50),RTE(50),ZLEOM(101),RLEOM(101),
     2    SLEOM(101),THLEOM(101),ZTEOM(101),RTEOM(101),STEOM(101),
     3    THTEOM(101),ILE(101),ITE(101),ZOM(100,101),ROM(100,101),
     4    SOM(100,101),TOM(100,101),BTH(100,101),DTHDS(100,101),
     5    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),FT(100,101),DFDM(100,101),
     6    XIOM(100,101),ZETOM(100,101),DLDU(100,101)
      REAL K,LAMDAF
      DO 10 J=1,MHTP1
      A(1,1,J) = 0.
      A(2,1,J) = 0.
      A(4,1,J) = 1.
      A(1,MM,J) = 0.
      A(2,MM,J) = 0.
      A(3,MM,J) = 1.
      DO 10 I=1,MM
      WSUBS(I,J) = 1.
      WSUBT(I,J) = 0.
      WSUBZ(I,J) = 1.
      W(I,J) = 0.
      WTH(I,J) = 0.
      VTH(I,J) = 0.
      DELRHO(I,J) = 0.
      XIOM(I,J) = 0.
      ZETOM(I,J) = 0.
      FT(I,J) = 0.
      DFDM(I,J) = 0.
      DLDU(I,J) = 0.
      SAMP(I,J) = 0.
      CAMP(I,J) = 1.
      K(I,J) = 0.
      PLOSS(I,J) = 0.
      IF (GAM.EQ.0.) GO TO 10
      UIJ = TOM(I,J)/TOM(I,MHTP1)*(ROM(I,J)+ROM(I,1))/
     1(ROM(I,MHTP1)+ROM(I,1))
      TPPTIP = 1.0-(2.*OMEGA*LAMDAF(UIJ,I,J)-(OMEGA*ROM(I,J))**2)/
     1(2.*CP*TIPF(UIJ))
      RHO(I,J) = RHOIPF(UIJ)*TPPTIP**EXPON
   10 RHOAV(I,J) = RHO(I,J)
      RETURN
      END


      SUBROUTINE COEF
C
C--COEF CALCULATES COEFFICIENTS, A AND K,
C--FOR THE SYSTEM OF MATRIX EQUATIONS,  A*U=K
C
      COMMON NREAD,NWRIT,ITER,IEND,NWRT1,NWRT2,NWRT3,NWRT4,NWRT5,NWRT6
      COMMON/INPUTT/GAM,AR,MSFL,OMEGA,REDFAC,VELTOL,FNEW,DNEW,MBI,MBO,
     1    MM,MHT,NBL,NHUB,NTIP,NIN,NOUT,NBLPL,NPPP,NOSTAT,NSL,NLOSS,
     2    LSFR,LTPL,LAMVT,LROT,LBLAD,LETEAN,ANGROT,IMESH,ISLINE,
     3    ISTATL,IPLOT,ISUPER,ITSON,IDEBUG,ZOMIN,ZOMBI,ZOMBO,ZOMOUT,
     4    ROMIN,ROMBI,ROMBO,ROMOUT,ZHIN,ZTIN,ZHOUT,ZTOUT,RHIN,RTIN,RHOUT,
     5    RTOUT,TITLEI(20),ZHUB(50),RHUB(50),ZTIP(50),RTIP(50),SFIN(50),
     6    RADIN(50),TIP(50),PRIP(50),LAMIN(50),VTHIN(50),SFOUT(50),
     7    RADOUT(50),PROP(50),LOSOUT(50),LAMOUT(50),VTHOUT(50),
```

```
     8   BETALE(50),BETATE(50),ZHST(50),ZTST(50),RHST(50),RTST(50),
     9   FLFR(50),PERCRD(50),PERLOS(50),ZBL(50,50),RBL(50,50),
     1   THBL(50,50),TNBL(50,50),TTBL(50,50),TH1BL(50,50),TH2BL(50,50)
      COMMON/CALCON/MMM1,MHTP1,CP,EXPON,TGROG,PITCH,RLEH,RLET,RTEH,RTET,
     1   ZLE(50),RLE(50),ZTE(50),RTE(50),ZLEOM(101),RLEOM(101),
     2   SLEOM(101),THLEOM(101),ZTEOM(101),RTEOM(101),STEOM(101),
     3   THTEOM(101),ILE(101),ITE(101),ZOM(100,101),ROM(100,101),
     4   SOM(100,101),TOM(100,101),BTH(100,101),DTHDS(100,101),
     5   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),FT(100,101),DFDM(100,101),
     6   XIOM(100,101),ZETOM(100,101),DLDU(100,101)
      DIMENSION DVTHDT(100,101)
      REAL MSFL,K,KNEW
C
C--CALCULATE COEFFICIENTS AND CONSTANTS FOR FINITE DIFFERENCE EQUATIONS
C
      WRITE(NWRIT,1000) ITER
      DCHANG = 0.
      DMAX = -1.E20
      DMIN = 1.E20
      MMM1 = MM-1
      DO 50 J=2,MHT
      H4 = SOM(2,J)-SOM(1,J)
      DO 50 I=2,MMM1
      IF (ITER.EQ.1) DVTHDT(I,J)=0.
      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)
      C1 = H1+H2
      C2 = H3+H4
      IF (ABS(CPHI(I,J)).LT.0.707) GO TO 10
      DELPHS = (SPHI(I+1,J)-SPHI(I-1,J))/CPHI(I,J)
      DELPHT = (SPHI(I,J+1)-SPHI(I,J-1))/CPHI(I,J)
      GO TO 20
   10 DELPHS = (CPHI(I-1,J)-CPHI(I+1,J))/SPHI(I,J)
      DELPHT = (CPHI(I,J-1)-CPHI(I,J+1))/SPHI(I,J)
   20 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)+DELPHS/C2
      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)/ROM(I,J)-DELPHT/C1
      A0 = 2./H1/H2+2./H3/H4
      A(1,I,J) = (2./H1+D1)/A0/C1
      A(2,I,J) = (2./H2-D1)/A0/C1
      A(3,I,J) = (2./H3+D2)/A0/C2
      A(4,I,J) = (2./H4-D2)/A0/C2
      KNEW = XIOM(I,J)*W(I,J)**2+ZETOM(I,J)
      IF (I.GE.ILE(J).AND.I.LE.ITE(J)) GO TO 30
      KNEW = KNEW+WTH(I,J)/MSFL*BTH(I,J)*RHO(I,J)*WSUBS(I,J)*DLDU(I,J)
      GO TO 40
   30 DVTEMP = (ROM(I,J+1)*VTH(I,J+1)-ROM(I,J-1)*VTH(I,J-1))/C1
      DCH = ABS(DVTEMP-DVTHDT(I,J))
```

```
      DCHANG = AMAX1(DCHANG,DCH)
      IF (DCHANG.EQ.DCH) ICH = I
      IF (DCHANG.EQ.DCH) JCH = J
      DMAX = AMAX1(DMAX,DVTEMP)
      DMIN = AMIN1(DMIN,DVTEMP)
      DVTHDT(I,J) = DNEW*DVTEMP+(1.-DNEW)*DVTHDT(I,J)
      KNEW = KNEW+WTH(I,J)/ROM(I,J)*DVTHDT(I,J)+FT(I,J)
      IF (GAM.EQ.0.) KNEW=KNEW+OMEGA*(LAMDAF(UOM(I,J+1),I,J+1)-
     1LAMDAF(UOM(I,J-1),I,J-1))/C1
   40 KNEW = KNEW*ROM(I,J)/AR*BTH(I,J)/MSFL*RHO(I,J)/WSUBS(I,J)
      K(I,J) = KNEW
   50 CONTINUE
      IF (ITER.GT.1) WRITE(NWRIT,1010) DCHANG,ICH,JCH,DVTHDT(ICH,JCH),
     1DMAX,DMIN
C
C--PRINT DEBUG OUTPUT
C
      IF (IDEBUG.LE.0) RETURN
      IF ((ITER/IDEBUG)*IDEBUG.NE.ITER.AND.ITER.NE.1) RETURN
      WRITE(NWRT5,1020)
      DO 60 J=2,MHT
      DO 60 I=1,MM
   60 WRITE(NWRT5,1030) I,J,(A(IJ,I,J),IJ=1,4),K(I,J)
      WRITE (NWRT5,1040)
      RETURN
C
C--FORMAT STATEMENTS
C
 1000 FORMAT(/////1X,22(1H*)/1X,16H*  ITERATION NO.,I3,3H  */1X,22(1H*))
 1010 FORMAT (//5X,26HMAXIMUM CHANGE IN DVTHDT =,G13.5,11X,6HAT I =,I3,
     15H, J =,I3,1H,,6X,14HWHERE DVTHDT =,G13.5/5X,26HMAXIMUM VALUE  OF
     2DVTHDT =,G13.5/5X,26HMINIMUM VALUE  OF DVTHDT =,G13.5)
 1020 FORMAT (1H1//30X,67(1H*)/30X,67H*     COEFFICIENTS  OF  MATRIX  EQU
     1ATION  FOR  STREAM  FUNCTION     */30X,67(1H*)///  5X,1HI,5X,1HJ,
     26X,4HA(1),12X,4HA(2),12X,4HA(3),12X,4HA(4),13X,1HK//)
 1030 FORMAT (2I6,5G16.6)
 1040 FORMAT (1H1)
      END
```

```
      SUBROUTINE SOR
C
C--SOR SOLVES THE SET OF MATRIX EQUATIONS,  A*U=K
C--BY THE SUCCESSIVE OVERRELAXATION TECHNIQUE
C
      COMMON NREAD,NWRIT,ITER,IEND,NWRT1,NWRT2,NWRT3,NWRT4,NWRT5,NWRT6
      COMMON/INPUTT/GAM,AR,MSFL,OMEGA,REDFAC,VELTOL,FNEW,DNEW,MBI,MBO,
     1   MM,MHT,NBL,NHUB,NTIP,NIN,NOUT,NBLPL,NPPP,NOSTAT,NSL,NLOSS,
     2   LSFR,LTPL,LAMVT,LROT,LBLAD,LETEAN,ANGROT,IMESH,ISLINE,
     3   ISTATL,IPLOT,ISUPER,ITSON,IDEBUG,ZOMIN,ZOMBI,ZOMBO,ZOMOUT,
     4   ROMIN,ROMBI,ROMBO,ROMOUT,ZHIN,ZTIN,ZHOUT,ZTOUT,RHIN,RTIN,RHOUT,
     5   RTOUT,TITLEI(20),ZHUB(50),RHUB(50),ZTIP(50),RTIP(50),SFIN(50),
     6   RADIN(50),TIP(50),PRIP(50),LAMIN(50),VTHIN(50),SFOUT(50),
     7   RADOUT(50),PROP(50),LOSOUT(50),LAMOUT(50),VTHOUT(50),
     8   BETALE(50),BETATE(50),ZHST(50),ZTST(50),RHST(50),RTST(50),
     9   FLFR(50),PERCRD(50),PERLOS(50),ZBL(50,50),RBL(50,50),
```

```
     1   THBL(50,50),TNBL(50,50),TTBL(50,50),TH1BL(50,50),TH2BL(50,50)
      COMMON/CALCON/MMM1,MHTP1,CP,EXPON,TGROG,PITCH,RLEH,RLET,RTEH,RTET,
     1   ZLE(50),RLE(50),ZTE(50),RTE(50),ZLEOM(101),RLEOM(101),
     2   SLEOM(101),THLEOM(101),ZTEOM(101),RTEOM(101),STEOM(101),
     3   THTEOM(101),ILE(101),ITE(101),ZOM(100,101),ROM(100,101),
     4   SOM(100,101),TOM(100,101),BTH(100,101),DTHDS(100,101),
     5   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),FT(100,101),DFDM(100,101),
     6   XIOM(100,101),ZETOM(100,101),DLDU(100,101)
      REAL K,LMAX,LMIN
C
C--AFTER FIRST ITERATION, JUST SOLVE EQUATION BY SOR
C
      IF (ITER.GT.1) GO TO 70
C
C--FIRST ITERATION ONLY, CALCULATE OPTIMUM ORF
C
C--SET BOUNDARY VALUES TO ZERO, AND INTERIOR VALUES TO ONE
      DO 10 I=1,MM
      UOM(I,1) = 0.
   10 UOM(I,MHTP1) = 0.
      DO 20 J=2,MHT
      DO 20 I=1,MM
   20 UOM(I,J) = 1.
C
C--CALCULATE OPTIMUM ORF
      ORFMAX = 2.0
      ICOUNT = 0
   30 LMAX = 0.
      LMIN = 1.
      ORF = ORFMAX
      ICOUNT = ICOUNT+1
      DO 40 J=2,MHT
      DO 40 I=1,MM
      UNEW = A(1,I,J)*UOM(I,J-1)+A(2,I,J)*UOM(I,J+1)
      IF (I.NE.1)  UNEW=UNEW+A(3,I,J)*UOM(I-1,J)
      IF (I.NE.MM) UNEW=UNEW+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.(2.-ORFMAX).OR.(ORF-ORFMAX).GT.0.0005)
     1GO TO 30
      ORF = ORFMAX
      WRITE (NWRIT,1000) ORF
C
C--RESTORE U BOUNDARY VALUE AT SHROUD
      DO 50 I=1,MM
   50 UOM(I,MHTP1) = 1.
C
C--SOLVE MATRIX EQUATION BY SOR
```

```
C
   70 ERROR = 0.
      DO 80 J=2,MHT
      DO 80 I=1,MM
      CHANGE = A(1,I,J)*UOM(I,J-1)+A(2,I,J)*UOM(I,J+1)+K(I,J)-UOM(I,J)
      IF (I.NE.1)  CHANGE=CHANGE+A(3,I,J)*UOM(I-1,J)
      IF (I.NE.MM) CHANGE=CHANGE+A(4,I,J)*UCM(I+1,J)
      CHANGE = ORF*CHANGE
      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 LOSSOM
C
C--LOSSOM COMPUTES THE RATIO OF ACTUAL TO IDEAL RELATIVE TOTAL PRESSURE
C--AT THE DOWNSTREAM INPUT STATION, AND THEN DISTRIBUTES THIS LOSS ON
C--THE ORTHOGONAL MESH AS SPECIFIED BY THE INPUT
C
      COMMON NREAD,NWRIT,ITER,IEND,NWRT1,NWRT2,NWRT3,NWRT4,NWRT5,NWRT6
      COMMON/INPUTT/GAM,AR,MSFL,OMEGA,REDFAC,VELTOL,FNEW,DNEW,MBI,MBO,
     1   MM,MHT,NBL,NHUB,NTIP,NIN,NOUT,NBLPL,NPPP,NOSTAT,NSL,NLOSS,
     2   LSFR,LTPL,LAMVT,LROT,LBLAD,LETEAN,ANGROT,IMESH,ISLINE,
     3   ISTATL,IPLOT,ISUPER,ITSON,IDEBUG,ZOMIN,ZOMBI,ZOMBO,ZOMOUT,
     4   ROMIN,ROMBI,ROMBO,ROMOUT,ZHIN,ZTIN,ZHOUT,ZTOUT,RHIN,RTIN,RHOUT,
     5   RTOUT,TITLEI(20),ZHUB(50),RHUB(50),ZTIP(50),RTIP(50),SFIN(50),
     6   RADIN(50),TIP(50),PRTP(50),LAMIN(50),VTHIN(50),SFOUT(50),
     7   RADOUT(50),PROP(50),LOSOUT(50),LAMOUT(50),VTHOUT(50),
     8   BETALE(50),BETATE(50),ZHST(50),ZTST(50),RHST(50),RTST(50),
     9   FLFR(50),PERCRD(50),PERLOS(50),ZBL(50,50),RBL(50,50),
     1   THBL(50,50),TNBL(50,50),TTBL(50,50),TH1BL(50,50),TH2BL(50,50)
      COMMON/CALCON/MMM1,MHTP1,CP,EXPON,TGROG,PITCH,RLEH,RLET,RTEH,RTET,
     1   ZLE(50),RLE(50),ZTE(50),RTE(50),ZLEOM(101),RLEOM(101),
     2   SLEOM(101),THLEOM(101),ZTEOM(101),RTEOM(101),STEOM(101),
     3   THTEOM(101),ILE(101),ITE(101),ZOM(100,101),ROM(100,101),
     4   SOM(100,101),TOM(100,101),BTH(100,101),DTHDS(100,101),
     5   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),FT(100,101),DFDM(100,101),
     6   XIOM(100,101),ZETOM(100,101),DLDU(100,101)
      COMMON/ROTATN/ZHROT(50),RHROT(50),ZTROT(50),RTROT(50),
     1   ZLEOMR(101),RLEOMR(101),ZTEOMR(101),RTEOMR(101),
     2   ZBLROT(50,50),RBLROT(50,50),ZOMROT(100,101),ROMROT(100,101)
      DIMENSION ZTEMR(2),RTEMR(2),SOMIN(101),SOMOUT(101)
      REAL LAMDAF,LOSOUT
      I1 = 1
      I2 = MM
      RFAC2 = REDFAC**2
C
```

```
C--REINITIALIZE LAMDAF AND RVTHTA FOR INCOMPRESSIBLE CASE
C
      IF (GAM.NE.0.) GO TO 5
      IF (LAMVT.EQ.0.AND.LSFR.EQ.0) RETURN
      CALL LAMNIT
      IF (MBI.NE.0) CALL RVTNIT
      RETURN
C
C--REINITIALIZE LAMDAF, RVTHTA, TIPF, AND RHOIPF
C
    5 IF (LAMVT.EQ.0.AND.LSFR.EQ.0) GO TO 10
      CALL LAMNIT
      CALL TIPNIT
      CALL RHINIT
      IF (MBI.NE.0) CALL RVTNIT
      GO TO 10
C
C--ENTRY POINT TO UPDATE PLOSS FOR TVELCY
C
      ENTRY LOSSTV(II)
      IF (GAM.EQ.0.) RETURN
      I1 = II
      I2 = II
      RFAC2 = 1.
C
C--CALCULATE LOSOUT ON DOWNSTREAM INPUT BOUNDARY, IF NOT GIVEN AS INPUT
   10 IF (LTPL.EQ.1) GO TO 30
      IF (I1.NE.1) GO TO 30
      ILOS = 0
      DO 20 JN=1,NOUT
      TINP = TIPF(SFOUT(JN))
      TOP = TINP-OMEGA/CP*(LAMDAF(SFOUT(JN),ILE(1),1)-RVTHTA(SFOUT(JN),
     1ILE(1),1))/RFAC2
      PRINP = RHOIPF(SFOUT(JN))*AR*TINP
      LOSOUT(JN) = 1.-PROP(JN)/PRINP*(TINP/TOP)**(GAM*EXPON)
   20 IF (LOSOUT(JN).LT.-.001) ILOS=1
      IF (ILOS.EQ.1) WRITE (NWRIT,1020)
   30 IF (ITER.GT.1) GO TO 35
      IF (LTPL.EQ.0) WRITE(NWRIT,1000) (JN,LOSOUT(JN),JN=1,NOUT)
      WRITE (NWRIT,1010) (J,ILE(J),ITE(J),J=1,MHTP1)
C
C--DISTRIBUTE TOTAL PRESSURE LOSS AT POINTS OF ORTHOGONAL MESH
C
   35 CALL SPLINT(SFOUT,LOSOUT,NOUT,0.,1,TEMP,TEMP1,TEMP2)
      DO 40 J=1,MHTP1
      DO 40 I=I1,I2
   40 CALL SPLENT(UOM(I,J),1,PLOSS(I,J),TEMP1,TEMP2)
      IF(MBI.EQ.0.OR.NLOSS.GT.0) GO TO 60
C--WITH BLADES, AND LINEAR DISTRIBUTION OF LOSS WITHIN BLADES
      DO 50 J=1,MHTP1
      SLENTH = STEOM(J)-SLEOM(J)
      DO 50 I=I1,I2
      PC = (SOM(I,J)-SLEOM(J))/SLENTH
   50 PLOSS(I,J) = AMIN1(1.,AMAX1(0.,PC))*PLOSS(I,J)
      RETURN
C--NO BLADES, OR INPUT DISTRIBUTION OF LOSS, CALCULATE SOMOUT
   60 IF(ITER.GT.1) GO TO 85
      CAN = COS(ANGROT)
      SAN = SIN(ANGROT)
      ZTEMR(1) = ZHOUT*CAN+RHOUT*SAN
```

```
      RTEMR(1) = RHOUT*CAN-ZHOUT*SAN
      ZTEMR(2) = ZTOUT*CAN+RTOUT*SAN
      RTEMR(2) = RTOUT*CAN-ZTOUT*SAN
      DO 80 J=1,MHTP1
      CALL INRSCT(ZOMROT(1,J),ROMROT(1,J),MM,ZTEMR,RTEMR,2,ZIR,RIR)
      DO 70 I=2,MM
      IF (ZIR.LE.ZOMROT(I,J)) GO TO 80
   70 CONTINUE
      I = MM+1
   80 SOMOUT(J) = SOM(I-1,J)+SQRT((ZIR-ZOMROT(I-1,J))**2+
     1(RIR-ROMROT(I-1,J))**2)
   85 IF(MBI.EQ.0) GO TO 100
C--WITH BLADES, AND INPUT DISTRIBUTION OF LOSS WITHIN BLADES, AND LINEAR
C--DISTRIBUTION OF LOSS FROM TRAILING EDGE TO DOWNSTREAM INPUT STATION
      CALL SPLINT(PERCRD,PERLOS,NLOSS,0.,1,PERLS,TEMP1,TEMP2)
      DELPL = 1.-PERLOS(NLOSS)
      DO 90 J=1,MHTP1
      SLENTH = STEOM(J)-SLEOM(J)
      DELPCO = DELPL/(SOMOUT(J)-STEOM(J))
      PERLS = 0.
      DO 90 I=I1,I2
      IF (I.LT.ILE(J)) GO TO 90
      IF (I.LE.ITE(J)) PC=(SOM(I,J)-SLEOM(J))/SLENTH
      IF (I.LE.ITE(J)) CALL SPLENT(PC,1,PERLS,TEMP1,TEMP2)
      IF (I.GT.ITE(J)) PERLS=PERLOS(NLOSS)+(SOM(I,J)-STEOM(J))*DELPCO
      IF (PERLS.GT.1.0) PERLS=1.0
   90 PLOSS(I,J) = PERLS*PLOSS(I,J)
      RETURN
C--NO BLADES, CALCULATE SOMIN
  100 IF(ITER.GT.1) GO TO 135
      ZTEMR(1) = ZHIN*CAN+RHIN*SAN
      RTEMR(1) = RHIN*CAN-ZHIN*SAN
      ZTEMR(2) = ZTIN*CAN+RTIN*SAN
      RTEMR(2) = RTIN*CAN-ZTIN*SAN
      DO 130 J=1,MHTP1
      CALL INRSCT(ZOMROT(1,J),ROMROT(1,J),MM,ZTEMR,RTEMR,2,ZIR,RIR)
      DO 110 I=1,MM
      IF (ZIR.LE.ZOMROT(I,J)) GO TO 120
  110 CONTINUE
  120 IF (I.EQ.1) SOMIN(J)=0.
  130 IF (I.NE.1) SOMIN(J)=SOM(I-1,J)+SQRT((ZIR-ZOMROT(I-1,J))**2+
     1(RIR-ROMROT(I-1,J))**2)
C--NO BLADES, AND LINEAR OR INPUT DISTRIBUTION OF LOSS FROM
C--UPSTREAM TO DOWNSTREAM INPUT STATIONS
  135 IF (NLOSS.GT.0) CALL SPLINT(PERCRD,PERLOS,NLOSS,0.,1,PERLS,
     1TEMP1,TEMP2)
      DO 140 J=1,MHTP1
      SLENTH = SOMOUT(J)-SOMIN(J)
      DO 140 I=I1,I2
      PC = (SOM(I,J)-SOMIN(J))/SLENTH
      PERLS = PC
      IF (NLOSS.GT.0) CALL SPLENT(PC,1,PERLS,TEMP1,TEMP2)
      IF (PC.LE.0.) PERLS=0.
      IF (PC.GE.1.) PERLS=1.
  140 PLOSS(I,J) = PERLS*PLOSS(I,J)
      RETURN
 1000 FORMAT (//5X,31HINITIAL CALCULATED LOSOUT ARRAY/10X,2HJN,6X,6HLOSO
     1UT/(9X,I2,3X,F10.6))
 1010 FORMAT (//5X,29HCALCULATED ILE AND ITE ARRAYS/10X,1HJ,5X,3HILE,3X,
```

```
     13HITE/(9X,I2,4X,I3,3X,I3))
 1020 FORMAT (1H1,5X,82HINPUT PROP VALUES ARE LARGER THAN IDEAL TOTAL PR
     1ESSURE, RESULTING IN NEGATIVE LOSS)
      END
```

```
      SUBROUTINE NEWRHO
C
C--NEWRHO CALCULATES VELOCITY COMPONENTS, VELOCITY MAGNITUDE,
C--AND NEW DENSITY AT EACH MESH POINT
C
      COMMON NREAD,NWRIT,ITER,IEND,NWRT1,NWRT2,NWRT3,NWRT4,NWRT5,NWRT6
      COMMON/INPUTT/GAM,AR,MSFI,OMEGA,REDFAC,VELTOL,FNEW,DNEW,MBI,MBO,
     1   MM,MHT,NBL,NHUB,NTIP,NIN,NOUT,NBLPL,NPPP,NOSTAT,NSL,NLOSS,
     2   LSFR,LTPL,LAMVT,LROT,LBLAD,LETEAN,ANGROT,IMESH,ISLINE,
     3   ISTATL,IPLOT,ISUPER,ITSON,IDEBUG,ZOMIN,ZOMBI,ZOMBO,ZOMOUT,
     4   ROMIN,ROMBI,ROMBO,ROMOUT,ZHIN,ZTIN,ZHOUT,ZTOUT,RHIN,RTIN,RHOUT,
     5   RTOUT,TITLEI(20),ZHUB(50),RHUB(50),ZTIP(50),RTIP(50),SFIN(50),
     6   RADIN(50),TIP(50),PRIP(50),LAMIN(50),VTHIN(50),SFOUT(50),
     7   RADOUT(50),PROP(50),LOSOUT(50),LAMOUT(50),VTHOUT(50),
     8   BETALE(50),BETATE(50),ZHST(50),ZTST(50),RHST(50),RTST(50),
     9   FLFR(50),PERCRD(50),PERLOS(50),ZBL(50,50),RBL(50,50),
     1   THBL(50,50),TNBL(50,50),TTBL(50,50),TH1BL(50,50),TH2BL(50,50)
      COMMON/CALCON/MMM1,MHTP1,CP,EXPON,TGROG,PITCH,RLEH,RLET,RTEH,RTET,
     1   ZLE(50),RLE(50),ZTE(50),RTE(50),ZLEOM(101),RLEOM(101),
     2   SLEOM(101),THLEOM(101),ZTEOM(101),RTEOM(101),STEOM(101),
     3   THTEOM(101),ILE(101),ITE(101),ZOM(100,101),ROM(100,101),
     4   SOM(100,101),TOM(100,101),BTH(100,101),DTHDS(100,101),
     5   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),FT(100,101),DFDM(100,101),
     6   XIOM(100,101),ZETOM(100,101),DLDU(100,101)
      DIMENSION DUDS(100),TVERT(101),UVERT(101),DUDT(101),TPP(101),
     1   PREL(101),DPDT(101),DTDT(101),AAA(101)
      REAL MSFL,LAMDAF,LAMBDA,LAMBDO
      RELER = 0.
      RELERA = 0.
      XNEW = 1.0
      ZNEW = 1.0
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 10 J=1,MHTP1
      CALL SPLINE(SOM(1,J),UOM(1,J),MM,DUDS,AAA)
      DO 10 I=1,MM
      WSUBT(I,J) = (-DUDS(I)*MSFL/(ROM(I,J)*BTH(I,J))-
     1DFDM(I,J)*DELRHO(I,J)/12.*COS(BETA(I,J))*SAMP(I,J))/RHOAV(I,J)
   10 CONTINUE
C
C--CALCULATE DERIVATIVES IN THE T DIRECTION OF THE SAME VARIABLES, AND
C--CALCULATE NEW VELOCITIES AND NEW DENSITY
```

```
C
      IREL = 1
      JREL = 1
      ICOUNT = 0
      DO 40 I=1,MM
      DO 20 J=1,MHTP1
      TVERT(J) = TOM(I,J)
   20 UVERT(J) = UOM(I,J)
      CALL SPLINE(TVERT,UVERT,MHTP1,DUDT,AAA)
      DO 30 J=1,MHTP1
      WSUBS(I,J) = (DUDT(J)*MSFL/(ROM(I,J)*BTH(I,J))-
     1DFDM(I,J)*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))
      OMR = OMEGA*ROM(I,J)
      LAMBDA = LAMDAF(UOM(I,J),I,J)
      LAMBDO = RVTHTA(UOM(I,J),I,J)
      IF (I.LT.ILE(J)) WTH(I,J)=LAMBDA/ROM(I,J)-OMR
      IF (I.GT.ITE(J)) WTH(I,J)=LAMBDO/ROM(I,J)-OMR
      VTH(I,J) = WTH(I,J)+OMR
      WSQ = WTH(I,J)**2+WSUBS(I,J)**2+WSUBT(I,J)**2
      WTEMP = SQRT(WSQ)
      ERR = 0.
      IF (W(I,J).NE.0.) ERR=ABS((WTEMP-W(I,J))/W(I,J))
      RELER = AMAX1(RELER,ERR)
      IF (RELER.EQ.ERR) IREL = I
      IF (RELER.EQ.ERR) JREL = J
      IF (ERR.GE.VELTOL) ICOUNT=ICOUNT+1
      RELERA = RELERA+ERR
      W(I,J) = WTEMP
      IF (GAM.EQ.0.) GO TO 30
      TIPT = TIPF(UOM(I,J))
      RHOIP = RHOIPF(UOM(I,J))*(1.-PLOSS(I,J))
      TPP(J) = TIPT-(2.*OMEGA*LAMBDA-OMR**2)/CP/2.
      IF (TPP(J).LT.0.) GO TO 60
      PREL(J) = RHOIP*AR*TIPT*(TPP(J)/TIPT)**(GAM*EXPON)
      TTIP = (TPP(J)-WSQ/CP/2.)/TIPT
      IF (TTIP.LT.0.) GO TO 50
      RHO(I,J) = RHOIP*TTIP**EXPON
   30 CONTINUE
      IF (GAM.EQ.0.) GO TO 40
      CALL SLOPES(TVERT,TPP,MHTP1,DTDT)
      CALL SLOPES(TVERT,PREL,MHTP1,DPDT)
      DO 35 J=1,MHTP1
      XIOMT = (AR/PREL(J)*DPDT(J)/CP-DTDT(J)/TPP(J))/2.
      ZETOMT = OMEGA**2*ROM(I,J)*CPHI(I,J)-AR/PREL(J)*TPP(J)*DPDT(J)
      XIOM(I,J) = XNEW*XIOMT+(1.-XNEW)*XIOM(I,J)
   35 ZETOM(I,J) = ZNEW*ZETOMT+(1.-ZNEW)*ZETOM(I,J)
   40 CONTINUE
      RELERA = RELERA/FLOAT(MM*MHTP1)
      IF (ITER.GT.1) WRITE(NWRIT,1020) RELER,IREL,JREL,RELERA,ICOUNT
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
   50 WRITE(NWRIT,1000)
      STOP
   60 WRITE(NWRIT,1010)
      STOP
 1000 FORMAT(////68H PROGRAM STOPPED IN NEWRHO DUE TO EXCESSIVE STREAM F
     1UNCTION GRADIENT)
 1010 FORMAT(////61H THE UPSTREAM INPUT WHIRL OR TANGENTIAL VELOCITY IS
     1TOO LARGE)
 1020 FORMAT(/  5X,37HMAXIMUM RELATIVE CHANGE IN VELOCITY =,G11.4,
     18H   AT I =,I3,5H, J =,I3/5X,37HAVERAGE RELATIVE CHANGE IN VELOCITY
     2 =,G11.4/5X,37HNUMBER OF UNCONVERGED MESH POINTS    =,I5)
      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 NREAD,NWRIT,ITER,IEND,NWRT1,NWRT2,NWRT3,NWRT4,NWRT5,NWRT6
      COMMON/INPUTT/GAM,AR,MSFL,OMEGA,REDFAC,VELTOL,FNEW,DNEW,MBI,MBO,
     1   MM,MHT,NBL,NHUB,NTIP,NIN,NOUT,NBLPL,NPPP,NOSTAT,NSL,NLOSS,
     2   LSFR,LTPL,LAMVT,LROT,LBLAD,LETEAN,ANGROT,IMESH,ISLINE,
     3   ISTATL,IPLOT,ISUPER,ITSON,IDEBUG,ZOMIN,ZOMBI,ZOMBO,ZOMOUT,
     4   ROMIN,ROMBI,ROMBO,ROMOUT,ZHIN,ZTIN,ZHOUT,ZTOUT,RHIN,RTIN,RHOUT,
     5   RTOUT,TITLEI(20),ZHUB(50),RHUB(50),ZTIP(50),RTIP(50),SFIN(50),
     6   RADIN(50),TIP(50),PRIP(50),LAMIN(50),VTHIN(50),SFOUT(50),
     7   RADOUT(50),PROP(50),LOSOUT(50),LAMOUT(50),VTHOUT(50),
     8   BETALE(50),BETATE(50),ZHST(50),ZTST(50),RHST(50),RTST(50),
     9   FLFR(50),PERCRD(50),PERLOS(50),ZBL(50,50),RBL(50,50),
     1   THBL(50,50),TNBL(50,50),TTBL(50,50),TH1BL(50,50),TH2BL(50,50)
      COMMON/CALCON/MMM1,MHTP1,CP,EXPON,TGROG,PITCH,RLEH,RLET,RTEH,RTET,
     1   ZLE(50),RLE(50),ZTE(50),RTE(50),ZLEOM(101),RLEOM(101),
     2   SLEOM(101),THLEOM(101),ZTEOM(101),RTEOM(101),STEOM(101),
     3   THTEOM(101),ILE(101),ITE(101),ZOM(100,101),ROM(100,101),
     4   SOM(100,101),TOM(100,101),BTH(100,101),DTHDS(100,101),
     5   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),FT(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/ROTATN/ZHROT(50),RHROT(50),ZTROT(50),RTROT(50),
     1   ZLEOMR(101),RLEOMR(101),ZTEOMR(101),RTEOMR(101),
     2   ZBLROT(50,50),RBLROT(50,50),ZOMROT(100,101),ROMROT(100,101)
      DIMENSION ZST(50),RST(50),MST(50),WZST(50),WRST(50),WMST(50),
```

```
     1    WTHST(50),ALPST(50),BETST(50),WST(50),WWCRST(50),CURVST(50),
     2    WLSST(50),WTSST(50),PLOST(50),VTHST(50),VST(50),BEABST(50),
     3    PST(50),TST(50),RHOST(50),PPST(50),TPST(50),PPPST(50),TPPST(50)
      DIMENSION DALDS(100),TVERT(101),ALVERT(101),DALVER(101),
     1    ZTEM(101),RTEM(101),UTEM(101),ZSLTEM(50),RSLTEM(50),
     2    AAA(101),BBB(101),WZFSEX(101),WRFSEX(101),BTFSEX(101),
     3    DALDT(100,101),CHOMES(2,100),ALTEM(100),BETEM(100),CHOK(2)
      DATA CHOMES/200*'    '/,CHOK/'  CH','OKED'/,BLNK/'    '/,NCHOK/0/
      REAL LAMDAF,LAMBDA,MSL,MST,MTEM
C
C--CALCULATE VELOCITY COMPONENTS AND FLOW ANGLES ON ORTHOGONAL MESH
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)  = ATAN2(WSUBR(I,J),WSUBZ(I,J))
   10 BETA(I,J)  = ATAN2(WTH(I,J),WSUBM(I,J))
      GO TO 30
C
      ENTRY TOUTPT
C
C--CALCULATE VELOCITY COMPONENTS ON MESH, 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--COMPUTE BLADE SURFACE VELOCITIES
C
   30 CALL BLDVEL
C
C--STORE 'CHOKED' MESSAGE FOR APPROPRIATE VERTICAL ORTHOGONAL
C--MESH LINES
C
      NCHOK = 0
      DO 25 I=1,MM
      IF (UOM(I,MHTP1).GT.0.9999) GO TO 25
      NCHOK = NCHOK+1
      CHOMES(1,I) = CHOK(1)
      CHOMES(2,I) = CHOK(2)
   25 CONTINUE
C
C--CALCULATE STREAMLINE CURVATURE AND CRITICAL VELOCITY RATIO ON MESH
C
      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)
      IF (GAM.EQ.0.) GO TO 60
      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.
      WWCR(I,J) = W(I,J)/SQRT(TGROG*TPP)
   60 CONTINUE
C
C--CHECK PRINT AND PLOT INDICATORS TO SEE IF OUTPUT CALCULATIONS
C--SHOULD BE MADE
C
      IF (IMESH.LE.0) GO TO 32
      IF ((ITER/IMESH)*IMESH.EQ.ITER.OR.ITER.EQ.1) GO TO 38
   32 IF (ISLINE.LE.0) GO TO 33
      IF ((ITER/ISLINE)*ISLINE.EQ.ITER.OR.ITER.EQ.1) GO TO 38
   33 IF (ISTATL.LE.0) GO TO 34
      IF ((ITER/ISTATL)*ISTATL.EQ.ITER.OR.ITER.EQ.1) GO TO 38
   34 IF (IPLOT.LE.0) GO TO 35
      IF ((ITER/IPLOT)*IPLOT.EQ.ITER.OR.ITER.EQ.1) GO TO 38
   35 IF (ITSON.LE.0) RETURN
      IF ((ITER/ITSON)*ITSON.NE.ITER) RETURN
   38 IF (MBI.EQ.0) GO TO 80
C
C--CHECK IF UPPER OR LOWER SURFACE IS SUCTION SURFACE
C
      REVERS = 0.0
      IF ((LAMDAF(.5,ILE(1),1)-RVTHTA(.5,ILE(1),1)).GT.0.) GO TO 80
      REVERS = 1.0
      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 ON ORTHOGONAL MESH
C
   80 IF (IMESH.LE.0) GO TO 100
      IF ((ITER/IMESH)*IMESH.NE.ITER.AND.ITER.NE.1) GO TO 100
      WRITE(NWRT1,1000)
      IF (REDFAC.LT.1.0) WRITE(NWRT1,1150) ITER
      IF (REDFAC.EQ.1.0.AND.IEND.LE.0) WRITE(NWRT1,1160) ITER
      IF (REDFAC.EQ.1.0.AND.IEND.GE.1.AND.ISUPER.LE.1) WRITE(NWRT1,1170)
      IF (REDFAC.EQ.1.0.AND.IEND.GE.1.AND.ISUPER.EQ.2) WRITE(NWRT1,1180)
      DO 90 J=1,MHTP1
      WRITE(NWRT1,1010) J
      WRITE(NWRT1,1020)
      DO 90 I=1,MM
      PHI = ATAN2(SPHI(I,J),CPHI(I,J))*DEGRAD
      ALPHIJ = ALPHA(I,J)*DEGRAD
      BETAIJ = BETA(I,J)*DEGRAD
   90 WRITE(NWRT1,1030) I,J,ZOM(I,J),ROM(I,J),UOM(I,J),WSUBM(I,J),
     1WTH(I,J),W(I,J),WWCR(I,J),ALPHIJ,BETAIJ,PHI,CHOMES(1,I),
     2CHOMES(2,I)
C
C--CALCULATION OF 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
C--CALCULATE STREAMLINE ZSL,RSL COORDINATES FOR PRINT OUT
  130 DO 150 I=1,MM
      DO 140 J=1,MHTP1
      ZTEM(J) = ZOMROT(I,J)
      RTEM(J) = ROMROT(I,J)
  140 UTEM(J) = UOM(I,J)
      CALL SPLINT(UTEM,RTEM,MHTP1,FLFR,NSL,RSLTEM,AAA,BBB)
      CALL SPLINT(RTEM,ZTEM,MHTP1,RSLTEM,NSL,ZSLTEM,AAA,BBB)
      DO 150 JS=1,NSL
      ZSL(I,JS) = ZSLTEM(JS)
  150 RSL(I,JS) = RSLTEM(JS)
C
C--CALCULATE STREAMLINE MSL COORDINATES FOR PRINT OUT AND PLOTTING
      DO 160 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)
C
C--INTERPOLATE TO OBTAIN OUTPUT DATA ON STREAMLINES
C
      II = 1
      JJ = 1
      DO 180 JS=1,NSL
      DO 180 IS=1,MM
      CALL LININT(ZOMROT,ROMROT,WSUBZ,MM,MHTP1,100,101,ZSL(IS,JS),
     1RSL(IS,JS),WZSL(IS,JS),II,JJ)
      CALL LININT(ZOMROT,ROMROT,WSUBR,MM,MHTP1,100,101,ZSL(IS,JS),
     1RSL(IS,JS),WRSL(IS,JS),II,JJ)
      CALL LININT(ZOMROT,ROMROT,WTH,MM,MHTP1,100,101,ZSL(IS,JS),
     1RSL(IS,JS),WTHSL(IS,JS),II,JJ)
      CALL LININT(ZOMROT,ROMROT,WWCR,MM,MHTP1,100,101,ZSL(IS,JS),
     1RSL(IS,JS),WWCRSL(IS,JS),II,JJ)
      CALL LININT(ZOMROT,ROMROT,CURV,MM,MHTP1,100,101,ZSL(IS,JS),
     1RSL(IS,JS),CURVSL(IS,JS),II,JJ)
      WMSL(IS,JS) = SQRT(WZSL(IS,JS)**2+WRSL(IS,JS)**2)
      ALPSL(IS,JS) = ATAN2(WRSL(IS,JS),WZSL(IS,JS))
      BETSL(IS,JS) = ATAN2(WTHSL(IS,JS),WMSL(IS,JS))
  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
      IF (MBI.EQ.0) GO TO 185
      CALL ILETE
C
C--INTERPOLATION FOR BLADE SURFACE VELOCITIES ON STREAMLINES
  185 DO 190 JS=1,NSL
      DO 190 IS=1,MM
      WLSSL(IS,JS) = 0.
  190 WTSSL(IS,JS) = 0.
      IF (MBI.EQ.0) GO TO 205
```

```
      II = 1
      JJ = 1
      DO 200 JS=1,NSL
      ILSJ = ILS(JS)
      ITSJ = ITS(JS)
      DO 200 IS=ILSJ,ITSJ
      CALL LININT(ZOMROT,ROMROT,WLSURF,MM,MHTP1,100,101,ZSL(IS,JS),
     1RSL(IS,JS),WLSSL(IS,JS),II,JJ)
  200 CALL LININT(ZOMROT,ROMROT,WTSURF,MM,MHTP1,100,101,ZSL(IS,JS),
     1RSL(IS,JS),WTSSL(IS,JS),II,JJ)
C
C--PRINT OUTPUT ON STREAMLINES
C
  205 IF (ISLINE.LE.0) GO TO 220
      IF ((ITER/ISLINE)*ISLINE.NE.ITER.AND.ITER.NE.1) GO TO 220
      WRITE(NWRT2,1040)
      IF (REDFAC.LT.1.0) WRITE(NWRT2,1150) ITER
      IF (REDFAC.EQ.1.0.AND.IEND.LE.0) WRITE(NWRT2,1160) ITER
      IF (REDFAC.EQ.1.0.AND.IEND.GE.1.AND.ISUPER.LE.1) WRITE(NWRT2,1170)
      IF (REDFAC.EQ.1.0.AND.IEND.GE.1.AND.ISUPER.EQ.2) WRITE(NWRT2,1180)
      CALL ROTATE(-ANGROT,ZSL,RSL,MM,NSL,100,50,ZSL,RSL)
      DO 210 JS=1,NSL
      DO 207 IS=1,MM
      ALTEM(IS) = ALPSL(IS,JS)*DEGRAD
  207 BETEM(IS) = BETSL(IS,JS)*DEGRAD
      WRITE(NWRT2,1050) JS,FLFR(JS)
      WRITE(NWRT2,1060)
  210 WRITE(NWRT2,1070) (ZSL(IS,JS),RSL(IS,JS),MSL(IS,JS),WMSL(IS,JS),
     1WTHSL(IS,JS),WSL(IS,JS),WWCRSL(IS,JS),ALTEM(IS),BETEM(IS),
     2CURVSL(IS,JS),WLSSL(IS,JS),WTSSL(IS,JS),CHOMES(1,IS),CHOMES(2,IS),
     3IS=1,MM)
      CALL ROTATE(ANGROT,ZSL,RSL,MM,NSL,100,50,ZSL,RSL)
C
C--CALCULATION OF 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
      WRITE(NWRT3,1080)
      IF (REDFAC.LT.1.0) WRITE(NWRT3,1150) ITER
      IF (REDFAC.EQ.1.0.AND.IEND.LE.0) WRITE(NWRT3,1160) ITER
      IF (REDFAC.EQ.1.0.AND.IEND.GE.1.AND.ISUPER.LE.1) WRITE(NWRT3,1170)
      IF (REDFAC.EQ.1.0.AND.IEND.GE.1.AND.ISUPER.EQ.2) WRITE(NWRT3,1180)
C
C--CALCULATE ZST AND RST ARRAYS
      CALL SPLINT(ZHROT,RHROT,NHUB,ZHST,NOSTAT,RHST,AAA,BBB)
      CALL SPLINT(ZTROT,RTROT,NTIP,ZTST,NOSTAT,RTST,AAA,BBB)
      DO 400 IL=1,NOSTAT
      MARK = 1
      RTEM(1)  = RHST(IL)
      RTEM(20) = RTST(IL)
      DELR = (RTEM(20)-RTEM(1))/19.0
      DO 230 J=2,19
  230 RTEM(J) = RTEM(J-1)+DELR
      ZST(1)   = ZHST(IL)
      ZST(NSL) = ZTST(IL)
      ZTEM(1)  = ZHST(IL)
      ZTEM(20) = ZTST(IL)
      DELZ = (ZTEM(20)-ZTEM(1))/19.0
C
```

```
C--CHECK FOR LEADING OR TRAILING EDGE STATION
      IF (MBI.EQ.0) GO TO 240
      DELCH = ABS(ZTEOMR(1)-ZLEOMR(1)+ZTEOMR(MHTP1)-ZLEOMR(MHTP1))*0.005
      IF(ABS(ZST(1)-ZLEOMR(1)).LT.DELCH.AND.ABS(ZST(NSL)-
     1ZLEOMR(MHTP1)).LT.DELCH)  MARK=2
      IF(ABS(ZST(1)-ZTEOMR(1)).LT.DELCH.AND.ABS(ZST(NSL)-
     1ZTEOMR(MHTP1)).LT.DELCH)  MARK=3
      IF (ZST(1).GT.(ZLEOMR(1)+DELCH).AND.ZST(1).LT.(ZTEOMR(1)-
     1DELCH))  MARK=4
      IF (MARK.EQ.2) GO TO 260
      IF (MARK.EQ.3) GO TO 270
C--REGULAR STATION
  240 DO 250 J=2,19
  250 ZTEM(J) = ZTEM(J-1)+DELZ
      GO TO 280
C--LEADING EDGE STATION
  260 CALL SPLINT(RLE,ZLE,NBLPL,RTEM,20,ZTEM,AAA,BBB)
      GO TO 280
C--TRAILING EDGE STATION
  270 CALL SPLINT(RTE,ZTE,NBLPL,RTEM,20,ZTEM,AAA,BBB)
C
C--INTERPOLATE FOR STREAM FUNCTION
  280 UTEM(1)  = 0.
      UTEM(20) = 1.
      II = 1
      JJ = 1
      DO 290 J=2,19
  290 CALL LININT(ZOMROT,ROMROT,UOM,MM,MHTP1,100,101,ZTEM(J),RTEM(J),
     1UTEM(J),II,JJ)
C
C--CALCULATE STATION LINE RST COORDINATES FOR PRINT OUT
      CALL SPLINT(UTEM,RTEM,20,FLFR,NSL,RST,AAA,BBB)
      DELR = RST(NSL)-RST(1)
      DELZ = ZST(NSL)-ZST(1)
      NSLM1 = NSL-1
C
C--CALCULATE STATION LINE ZST COORDINATES FOR PRINT OUT
      GO TO (300,320,330,300), MARK
  300 DO 310 JL=2,NSLM1
  310 ZST(JL) = ZST(1)+(RST(JL)-RST(1))/DELR*DELZ
      GO TO 340
  320 CALL SPLINT(RLE,ZLE,NBLPL,RST,NSL,ZST,AAA,BBB)
      GO TO 340
  330 CALL SPLINT(RTE,ZTE,NBLPL,RST,NSL,ZST,AAA,BBB)
C
C--CALCULATE STATION LINE MST COORDINATES FOR PRINT OUT
  340 DO 350 JL=1,NSL
  350 MST(JL) = 0.
      IF (ISLINE.LE.0) GO TO 370
      IF ((ITER/ISLINE)*ISLINE.NE.ITER.AND.ITER.NE.1) GO TO 370
      II = 1
      JJ = 1
      DO 360 JL=1,NSL
  360 CALL LININT(ZSL,RSL,MSL,MM,NSL,100,50,ZST(JL),RST(JL),MST(JL),
     1II,JJ)
C
C--INTERPOLATE TO OBTAIN OUTPUT DATA ON STATION LINES
C
  370 II = 1
```

```
      JJ = 1
      IF (MARK.NE.2.AND.MARK.NE.3) GO TO 386
C--SPECIAL CASE OF LEADING OR TRAILING EDGE STATION
C--EXTRAPOLATE FROM FREE STREAM FOR VELOCITIES AND FLOW ANGLE
      IF (MARK.EQ.3) GO TO 376
      DO 375 J=1,MHTP1
      I = ILE(J)-1
      EXFRAC = (SLEOM(J)-SOM(I,J))/(SOM(I,J)-SOM(I-1,J))
      WZFSEX(J) = WSUBZ(I,J)+EXFRAC*(WSUBZ(I,J)-WSUBZ(I-1,J))
      WRFSEX(J) = WSUBR(I,J)+EXFRAC*(WSUBR(I,J)-WSUBR(I-1,J))
      BTFSEX(J) = BETA(I,J)+EXFRAC*(BETA(I,J)-BETA(I-1,J))
  375 RTEM(J) = RLEOMR(J)
      GO TO 378
  376 DO 377 J=1,MHTP1
      I = ITE(J)+1
      EXFRAC = (SOM(I,J)-STEOM(J))/(SOM(I+1,J)-SOM(I,J))
      WZFSEX(J) = WSUBZ(I,J)+EXFRAC*(WSUBZ(I,J)-WSUBZ(I+1,J))
      WRFSEX(J) = WSUBR(I,J)+EXFRAC*(WSUBR(I,J)-WSUBR(I+1,J))
      BTFSEX(J) = BETA(I,J)+EXFRAC*(BETA(I,J)-BETA(I+1,J))
  377 RTEM(J) = RTEOMR(J)
  378 JLTE = 1
      DO 384 JL=1,NSL
      DO 380 J=JLTE,MHT
      IF (RST(JL).LE.RTEM(J+1)) GO TO 382
  380 CONTINUE
  382 JLTE = J
      EXFRAC = (RST(JL)-RTEM(J))/(RTEM(J+1)-RTEM(J))
      WZST(JL) = WZFSEX(J)+EXFRAC*(WZFSEX(J+1)-WZFSEX(J))
      WRST(JL) = WRFSEX(J)+EXFRAC*(WRFSEX(J+1)-WRFSEX(J))
      BETST(JL) = BTFSEX(J)+EXFRAC*(BTFSEX(J+1)-BTFSEX(J))
      WMST(JL) = SQRT(WZST(JL)**2+WRST(JL)**2)
      WTHST(JL) = WMST(JL)*TAN(BETST(JL))
  384 BETST(JL) = BETST(JL)*DEGRAD
      GO TO 390
C--NORMAL CASE OF FREESTREAM STATION, OR STATION WITHIN BLADE
  386 DO 388 JL=1,NSL
      CALL LININT(ZOMROT,ROMROT,WSUBZ,MM,MHTP1,100,101,ZST(JL),
     1RST(JL),WZST(JL),II,JJ)
      CALL LININT(ZOMROT,ROMROT,WSUBR,MM,MHTP1,100,101,ZST(JL),
     1RST(JL),WRST(JL),II,JJ)
      CALL LININT(ZOMROT,ROMROT,WTH,MM,MHTP1,100,101,ZST(JL),
     1RST(JL),WTHST(JL),II,JJ)
      WMST(JL) = SQRT(WZST(JL)**2+WRST(JL)**2)
  388 BETST(JL) = ATAN2(WTHST(JL),WMST(JL))*DEGRAD
C
C--CALCULATE OTHER OUTPUT DATA ON STATION LINES
C
  390 DO 392 JL=1,NSL
      CALL LININT(ZOMROT,ROMROT,CURV,MM,MHTP1,100,101,ZST(JL),
     1RST(JL),CURVST(JL),II,JJ)
      CALL LININT(ZOMROT,ROMROT,PLOSS,MM,MHTP1,100,101,ZST(JL),
     1RST(JL),PLOST(JL),II,JJ)
      ALPST(JL) = ATAN2(WRST(JL),WZST(JL))*DEGRAD
      WST(JL) = SQRT(WMST(JL)**2+WTHST(JL)**2)
      WLSST(JL) = 0.
      WTSST(JL) = 0.
      IF (MARK.EQ.1) GO TO 392
      CALL LININT(ZOMROT,ROMROT,WLSURF,MM,MHTP1,100,101,ZST(JL),
     1RST(JL),WLSST(JL),II,JJ)
      CALL LININT(ZOMROT,ROMROT,WTSURF,MM,MHTP1,100,101,ZST(JL),
```

```
     1RST(JL),WTSSST(JL),II,JJ)
 392 CONTINUE
     CALL ROTATE(-ANGROT,ZST,RST,NSL,1,50,1,ZST,RST)
C
C--CALCULATE EXTRA OUTPUT DATA ON STATION LINES
C
      DO 396 JL=1,NSL
      LAMBDA = LAMDAF(FLFR(JL),ILE(1),1)
      OMR = OMEGA*RST(JL)
      VTHST(JL) = WTHST(JL)+OMR
      VSQ = WMST(JL)**2+VTHST(JL)**2
      VST(JL) = SQRT(VSQ)
      BEABST(JL) = ATAN2(VTHST(JL),WMST(JL))*DEGRAD
      IF (GAM.EQ.0.) GO TO 396
      TIPT = TIPF(FLFR(JL))
      RHOIP = RHOIPF(FLFR(JL))*(1.-PLOST(JL))
      TPPST(JL) = TIPT-(2.*OMEGA*LAMBDA-OMR**2)/2./CP
      WWCRST(JL) = WST(JL)/SQRT(TGROG*TPPST(JL))
      RHOPP = RHOIP*(TPPST(JL)/TIPT)**EXPON
      PPPST(JL) = RHOPP*AR*TPPST(JL)
      TPST(JL) = TIPT+(OMR*VTHST(JL)-OMEGA*LAMBDA)/CP
      RHOP = RHOIP*(TPST(JL)/TIPT)**EXPON
      PPST(JL) = RHOP*AR*TPST(JL)
      TST(JL) = TPST(JL)-VSQ/2./CP
      RHOST(JL) = RHOP*(TST(JL)/TPST(JL))**EXPON
      PST(JL) = RHOST(JL)*AR*TST(JL)
  396 CONTINUE
C
C--PRINT OUTPUT ALONG HUB-SHROUD STATION LINES
C
      IF (NCHOK.GT.0) WRITE(NWRT3,1085) NCHOK
      IF (MARK.EQ.1) WRITE(NWRT3,1090) IL
      IF (MARK.EQ.2) WRITE(NWRT3,1100) IL
      IF (MARK.EQ.3) WRITE(NWRT3,1110) IL
      IF (MARK.EQ.4) WRITE(NWRT3,1120) IL
      WRITE(NWRT3,1130)
      WRITE(NWRT3,1140) (RST(JL),ZST(JL),MST(JL),FLFR(JL),WMST(JL),
     1WTHST(JL),WST(JL),WWCRST(JL),ALPST(JL),BETST(JL),CURVST(JL),
     2WLSST(JL),WTSST(JL),JL=1,NSL)
      WRITE(NWRT3,1142)
      WRITE(NWRT3,1144) (RST(JL),ZST(JL),PST(JL),TST(JL),RHOST(JL),
     1VTHST(JL),VST(JL),PPST(JL),TPST(JL),BEABST(JL),PPPST(JL),
     2TPPST(JL),JL=1,NSL)
  400 CONTINUE
C
C--REVERSE UPPER AND LOWER SURFACE VELOCITIES, IF NECESSARY
C
  410 IF(REVERS.EQ.0.) GO TO 430
      DO 420 J=1,MHTP1
      DO 420 I=1,MM
      WDUM = WLSURF(I,J)
      WLSURF(I,J) = WTSURF(I,J)
  420 WTSURF(I,J) = WDUM
      REVERS = 0.0
C
C--REMOVE 'CHOKED' MESSAGE, IF NECESSARY
C
  430 IF (NCHOK.EQ.0) RETURN
      DO 440 I=1,MM
```

```
      CHOMES(1,I) = BLNK
  440 CHOMES(2,I) = BLNK
      RETURN
C
C--FORMAT STATEMENTS
C
 1000 FORMAT (1H1////28X,79(1H*)/28X,79H***  STREAM FUNCTION, INTERIOR V
     1ELOCITIES, VELOCITY COMPONENTS, AND ANGLES  ***/44X,41HAT ALL MESH
     2 POINTS OF THE 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),2A4)
 1040 FORMAT (1H1////15X,99(1H*)/15X,99H***  STREAM FUNCTION, INTERIOR V
     1ELOCITIES, VELOCITY COMPONENTS, ANGLES, AND SURFACE VELOCITIES  **
     2*/56X,17HALONG 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,F8.2,2A4))
 1080 FORMAT (1H1////15X,99(1H*)/15X,99H***  STREAM FUNCTION, INTERIOR V
     1ELOCITIES, VELOCITY COMPONENTS, ANGLES, AND SURFACE VELOCITIES  **
     2*/28X,72HALONG LINES FROM HUB TO SHROUD AT VARIOUS STATIONS THROUG
     3H THE BLADE ROW/28X,72(1H*))
 1085 FORMAT ( //28X,19HBEWARE.   THERE ARE,I3,49H VERTICAL ORTHOGONAL M
     1ESH LINES WHICH ARE CHOKED./28X,82HLOCATIONS OF THESE LINES ARE GI
     2VEN ABOVE AT THE BEGINNING OF THE TRANSONIC OUTPUT./28X,87HOUTPUT
     3ON ANY STATION LINES LOCATED NEAR THESE CHOKED ORTHOGONAL LINES MA
     4Y BE IN ERROR.)
 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))
```

```
1142 FORMAT (//4X,6HRADIAL,7X,5HAXIAL,8X,3(6HSTATIC,3X),9HABS.TANG.,2X,
    14HABS.,3X,2(8HABS.TOT.,1X),8HABS.FLOW,2X,2(1X,8HREL.TOT.)/
    24X,2(6HCOORD.,7X),1X,5HPRES.,4X,5HTEMP.,4X,5HDENS.,2(5X,4HVEL.),
    34X,5HPRES.,4X,5HTEMP.,4X,5HANGLE,6X,5HPRES.,4X,5HTEMP./  5X,3H(R)
    4,10X,3H(Z),10X,3H(P),6X,3H(T),6X,5H(RHO),4X,5H(VTH),5X,3H(V),5X,
    54H(PP),5X,4H(TP),4X,6H(BETA),6X,5H(PPP),4X,5H(TPP))
1144 FORMAT ((1X,2(G12.5,1X),F9.2,1X,F7.2,1X,F10.7,1X,2(F8.2,1X),F9.1,
    11X,F7.2,1X,F7.2,3X,F9.1,1X,F7.2))
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, BLADE-TO-BLADE
C--AVERAGE DENSITY, AND FT
C
      COMMON NREAD,NWRIT,ITER,IEND,NWRT1,NWRT2,NWRT3,NWRT4,NWRT5,NWRT6
      COMMON/INPUTT/GAM,AR,MSFL,OMEGA,REDFAC,VELTOL,FNEW,DNEW,MBI,MBO,
     1   MM,MHT,NBL,NHUB,NTIP,NIN,NOUT,NBLPL,NPPP,NOSTAT,NSL,NLOSS,
     2   LSFR,LTPL,LAMVT,LROT,LBLAD,LETEAN,ANGROT,IMESH,ISLINE,
     3   ISTATL,IPLOT,ISUPER,ITSON,IDEBUG,ZOMIN,ZOMBI,ZOMBO,ZOMOUT,
     4   ROMIN,ROMBI,ROMBO,ROMOUT,ZHIN,ZTIN,ZHOUT,ZTOUT,RHIN,RTIN,RHOUT,
     5   RTOUT,TITLEI(20),ZHUB(50),RHUB(50),ZTIP(50),RTIP(50),SFIN(50),
     6   RADIN(50),TIP(50),PRIP(50),LAMIN(50),VTHIN(50),SFOUT(50),
     7   RADOUT(50),PROP(50),LOSOUT(50),LAMOUT(50),VTHOUT(50),
     8   BETALE(50),BETATE(50),ZHST(50),ZTST(50),RHST(50),RTST(50),
     9   FLFR(50),PERCRD(50),PERLOS(50),ZBL(50,50),RBL(50,50),
     1   THBL(50,50),TNBL(50,50),TTBL(50,50),TH1BL(50,50),TH2BL(50,50)
      COMMON/CALCON/MMM1,MHTP1,CP,EXPON,TGROG,PITCH,RLEH,RLET,RTEH,RTET,
     1   ZLE(50),RLE(50),ZTE(50),RTE(50),ZLEOM(101),RLEOM(101),
     2   SLEOM(101),THLEOM(101),ZTEOM(101),RTEOM(101),STEOM(101),
     3   THTEOM(101),ILE(101),ITE(101),ZOM(100,101),ROM(100,101),
     4   SOM(100,101),TOM(100,101),BTH(100,101),DTHDS(100,101),
     5   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),FT(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 LAMDAF
   10 FCHANG = 0.
```

```
      FMAX = -1.E20
      FMIN = 1.E20
C
C---CALCULATE DFDT
C
      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
C---CALCULATE DFDS, THEN DFDM AND BLADE SURFACE VELOCITIES
C
      DO 50 J=1,MHTP1
      CALL SLOPES(SOM(1,J),FST(1,J),MM,DFDS)
      DO 50 I=1,MM
      DFDM(I,J) = 0.
      IF (I.GE.ILE(J).AND.I.LE.ITE(J)) DFDM(I,J)=-(DFDS(I)*CAMP(I,J)+
     1DFDT(I,J)*SAMP(I,J))*BTH(I,J)*COS(BETA(I,J))
      WLSURF(I,J) = W(I,J)+DFDM(I,J)/2.
      WTSURF(I,J) = W(I,J)-DFDM(I,J)/2.
C
C---CALCULATE BLADE-TO-BLADE AVERAGE DENSITY
C
      IF (GAM.EQ.0.) GO TO 40
      TWLMR = 2.*OMEGA*LAMDAF(UOM(I,J),I,J)-(OMEGA*ROM(I,J))**2
      WSQ = WLSURF(I,J)**2
      TIPIJ = TIPF(UOM(I,J))
      TTIP = 1.-(WSQ+TWLMR)/CP/TIPIJ/2.
      IF(TTIP.LT.0.) TTIP = 0.
      RHOIJ = RHOIPF(UOM(I,J))*(1.-PLOSS(I,J))
      RHOL = RHOIJ*TTIP**EXPON
      WSQ = WTSURF(I,J)**2
      TTIP = 1.-(WSQ+TWLMR)/CP/TIPIJ/2.
      IF(TTIP.LT.0.) TTIP = 0.
      RHOT = RHOIJ*TTIP**EXPON
      DELRHO(I,J) = RHOL-RHOT
      RHOAV(I,J) = (RHOL+4.*RHO(I,J)+RHOT)/6.
C
C---CALCULATE F-SUB-T FOR SUBROUTINE COEF
C
   40 FTT = W(I,J)/BTH(I,J)*DTHDT(I,J)*DFDM(I,J)
      FCH = ABS(FTT-FT(I,J))
      FCHANG = AMAX1(FCHANG,FCH)
      IF (FCHANG.EQ.FCH) ICH=I
      IF (FCHANG.EQ.FCH) JCH=J
      FMAX = AMAX1(FMAX,FTT)
      FMIN = AMIN1(FMIN,FTT)
      FT(I,J) = FNEW*FTT+(1.-FNEW)*FT(I,J)
   50 CONTINUE
      IF (IEND.LT.1) WRITE(NWRIT,1040) FCHANG,ICH,JCH,FT(ICH,JCH),
     1FMAX,FMIN
C
```

```
C---PRINT DEBUG OUTPUT IF REQUESTED
C
      IF (IDEBUG.LE.0) RETURN
      IF ((ITER/IDEBUG)*IDEBUG.NE.ITER.AND.ITER.NE.1) RETURN
      WRITE(NWRT5,1020)
      WRITE(NWRT5,1000) ((I,J,WSUBS(I,J),WSUBT(I,J),VTH(I,J),RHO(I,J),
     1RHOAV(I,J),DELRHO(I,J),DLDU(I,J),PLOSS(I,J),I=1,MM),J=1,MHTP1)
      WRITE(NWRT5,1030)
      WRITE(NWRT5,1010) ((I,J,DTHDS(I,J),FT(I,J),DFDM(I,J),XIOM(I,J),
     1ZETOM(I,J),CAMP(I,J),SAMP(I,J),I=1,MM),J=1,MHTP1)
      RETURN
C
C--FORMAT STATEMENTS
C
 1000 FORMAT (2I5,8G15.5)
 1010 FORMAT (2I5,7G15.5)
 1020 FORMAT (1H1////35X,57(1H*)/35X,57H*    CHANGING  QUANTITIES  ON  T
     1HE  ORTHOGONAL  MESH    */35X,57(1H*)//    4X,1HI,4X,1HJ,5X,
     25HWSUBS,11X,5HWSUBT,11X,3HVTH,11X,3HRHO,11X,5HRHOAV,9X,6HDELRHO,
     310X,4HDLDU,11X,5HPLOSS)
 1030 FORMAT (////4X,1HI,4X,1HJ,5X,5HDTHDS,11X,2HFT,12X,4HDFDM,11X,
     14HXIOM,11X,5HZETOM,10X,4HCAMP,11X,4HSAMP)
 1040 FORMAT (/5X,22HMAXIMUM CHANGE IN FT =,G13.5,15X,6HAT I =,I3,
     15H, J =,I3,1H,,6X,10HWHERE FT =,G13.5/5X,22HMAXIMUM VALUE  OF FT =
     2,G13.5/5X,22HMINIMUM VALUE  OF FT =,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,NLOSS,
     2   LSFR,LTPL,LAMVT,LROT,LBLAD,LETEAN,ANGROT,IMESH,ISLINE,
     3   ISTATL,IPLOT,ISUPER,ITSON,IDEBUG,ZOMIN,ZOMBI,ZOMBO,ZOMOUT,
     4   ROMIN,ROMBI,ROMBO,ROMOUT,ZHIN,ZTIN,ZHOUT,ZTOUT,RHIN,RTIN,RHOUT,
     5   RTOUT,TITLEI(20),ZHUB(50),RHUB(50),ZTIP(50),RTIP(50),SFIN(50),
     6   RADIN(50),TIP(50),PRIP(50),LAMIN(50),VTHIN(50),SFOUT(50),
     7   RADOUT(50),PROP(50),LOSOUT(50),LAMOUT(50),VTHOUT(50),
     8   BETALE(50),BETATE(50),ZHST(50),ZTST(50),RHST(50),RTST(50),
     9   FLFR(50),PERCRD(50),PERLOS(50),ZBL(50,50),RBL(50,50),
     1   THBL(50,50),TNBL(50,50),TTBL(50,50),TH1BL(50,50),TH2BL(50,50)
      COMMON/CALCON/MMM1,MHTP1,CP,EXPON,TGROG,PITCH,RLEH,RLET,RTEH,RTET,
     1   ZLE(50),RLE(50),ZTE(50),RTE(50),ZLEOM(101),RLEOM(101),
     2   SLEOM(101),THLEOM(101),ZTEOM(101),RTEOM(101),STEOM(101),
     3   THTEOM(101),ILE(101),ITE(101),ZOM(100,101),ROM(100,101),
     4   SOM(100,101),TOM(100,101),BTH(100,101),DTHDS(100,101),
     5   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,TEMP)
      DO 20 J=1,NSL
```

```
      I = 0
   10 I = I+1
      CALL SPLENT(RSL(I,J),1,ZSPL,DZDR,TEMP)
      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,TEMP)
      DO 40 J=1,NSL
      I = ILS(J)-1
   30 I = I+1
      CALL SPLENT(RSL(I,J),1,ZSPL,DZDR,TEMP)
      IF (ZSPL.GE.ZSL(I,J)) GO TO 30
   40 ITS(J) = I-1
      RETURN
      END
```

```
      SUBROUTINE INDEV
C
C--INDEV CALCULATES A CORRECTION TO DTHDS TO ALLOW FOR INCIDENCE AND
C--DEVIATION (AFTER BLOCKAGE CORRECTION)
C
      COMMON NREAD,NWRIT,ITER,IEND,NWRT1,NWRT2,NWRT3,NWRT4,NWRT5,NWRT6
      COMMON/INPUTT/GAM,AR,MSFL,OMEGA,REDFAC,VELTOL,FNEW,DNEW,MBI,MBO,
     1   MM,MHT,NBL,NHUB,NTIP,NIN,NOUT,NBLPL,NPPP,NOSTAT,NSL,NLOSS,
     2   LSFR,LTPL,LAMVT,LROT,LBLAD,LETEAN,ANGROT,IMESH,ISLINE,
     3   ISTATL,IPLOT,ISUPER,ITSON,IDEBUG,ZOMIN,ZOMBI,ZOMBO,ZOMOUT,
     4   ROMIN,ROMBI,ROMBO,ROMOUT,ZHIN,ZTIN,ZHOUT,ZTOUT,RHIN,RTIN,RHOUT,
     5   RTOUT,TITLEI(20),ZHUB(50),RHUB(50),ZTIP(50),RTIP(50),SFIN(50),
     6   RADIN(50),TIP(50),PRIP(50),LAMIN(50),VTHIN(50),SFOUT(50),
     7   RADOUT(50),PROP(50),LOSOUT(50),LAMOUT(50),VTHOUT(50),
     8   BETALE(50),BETATE(50),ZHST(50),ZTST(50),RHST(50),RTST(50),
     9   FLFR(50),PERCRD(50),PERLOS(50),ZBL(50,50),RBL(50,50),
     1   THBL(50,50),TNBL(50,50),TTBL(50,50),TH1BL(50,50),TH2BL(50,50)
      COMMON/CALCON/MMM1,MHTP1,CP,EXPON,TGROG,PITCH,RLEH,RLET,RTEH,RTET,
     1   ZLE(50),RLE(50),ZTE(50),RTE(50),ZLEOM(101),RLEOM(101),
     2   SLEOM(101),THLEOM(101),ZTEOM(101),RTEOM(101),STEOM(101),
     3   THTEOM(101),ILE(101),ITE(101),ZOM(100,101),ROM(100,101),
     4   SOM(100,101),TOM(100,101),BTH(100,101),DTHDS(100,101),
     5   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),FT(100,101),DFDM(100,101),
     6   XIOM(100,101),ZETOM(100,101),DLDU(100,101)
      COMMON/ROTATN/ZHROT(50),RHROT(50),ZTROT(50),RTROT(50),
     1   ZLEOMR(101),RLEOMR(101),ZTEOMR(101),RTEOMR(101),
     2   ZBLROT(50,50),RBLROT(50,50),ZOMROT(100,101),ROMROT(100,101)
      COMMON/INDCOM/NBLPC,NPPC,ZPC(51,51),RPC(51,51),TTPC(51,51),
     1   THPC(51,51),DTHDZ(51,51),DTHDR(51,51),BTHLE(101),BTHTE(101),
     2   BTBFLE(101),BTBFTE(101)
      REAL LAMDAF
    5 IPRNT = 0
      IF (IPRNT.EQ.1) RETURN
```

```
      DEGRAD = 180./3.1415927
      II = 1
      JJ = 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(NWRT6,1000)
      IF (REDFAC.LT.1.0) WRITE(NWRT6,1010) ITER
      IF (REDFAC.EQ.1.0.AND.IEND.LE.0) WRITE(NWRT6,1020) ITER
      WRITE(NWRT6,1030)
      IPRNT = 1
C
C--CORRECT DTHDS, AND CALCULATE INCIDENCE AND DEVIATION, ROW BY ROW
C--FROM HUB TO TIP
C
   40 DO 100 J=1,MHTP1
C
C--CALCULATE BLADE MEAN CAMBER ANGLE AT LEADING EDGE
      I = ILE(J)-1
      EXTRAP = SLEOM(J)-SOM(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,NPPC,NBLPC,51,51,ZLEOMR(J),RLEOMR(J),
     1DTDZLE,II,JJ)
      CALL LININT(ZPC,RPC,DTHDR,NPPC,NBLPC,51,51,ZLEOMR(J),RLEOMR(J),
     1DTDRLE,II,JJ)
      TANBBL = RLEOM(J)*(DTDRLE*SIN(ALPHLE)+DTDZLE*COS(ALPHLE))
      IF (ITER.EQ.1) BTBFLE(J)=ATAN(TANBBL)
      BTBLLE = ATAN(TANBBL)*DEGRAD
C
C--CALCULATE BLADE FLOW ANGLE AT LEADING EDGE, CORRECTED FOR BLOCKAGE
      BETAFS = BETA(I,J)+EXTRAP*(BETA(I,J)-BETA(I-1,J))/(SOM(I,J)-
     1SOM(I-1,J))
      RHOFS = RHO(I,J)+EXTRAP*(RHO(I,J)-RHO(I-1,J))/
     1(SOM(I,J)-SOM(I-1,J))
      RHOBF = RHOAV(I+1,J)-(SOM(I+1,J)-SOM(I,J)-EXTRAP)/
     1(SOM(I+2,J)-SOM(I+1,J))*(RHOAV(I+2,J)-RHOAV(I+1,J))
      TANBBF = TAN(BETAFS)*BTHLE(J)/PITCH*RHOBF/RHOFS
      BETABF = ATAN(TANBBF)
C
C--CALCULATE DISTANCE FOR DTHDS CORRECTION
      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))
C
C--CORRECT DTHDS FOR INCIDENCE NEAR THE LEADING EDGE,
C--USING LINEAR CORRECTION FOR ANGLE
      I = ILE(J)
   50 SDIST = SLEOM(J)+DISTLE-SOM(I,J)
      IF(SDIST.LE.0.) GO TO 60
      TANBIJ = ROM(I,J)*(DTHDS(I,J)*CAMP(I,J)+DTHDT(I,J)*SAMP(I,J))
      BETAIJ = ATAN(TANBIJ)
      BETAIJ = BETAIJ+(BETABF-BTBFLE(J))*SDIST/DISTLE
      TANBIJ = TAN(BETAIJ)
```

```
      DTHDS(I,J) = (TANBIJ/ROM(I,J)-DTHDT(I,J)*SAMP(I,J))/CAMP(I,J)
      I = I+1
      GO TO 50
   60 BTBFLE(J) = BETABF
C
C--CALCULATE INCIDENCE ANGLES
      BLINC = BETABF*DEGRAD-BTBLLE
      UBINC = BETAFS*DEGRAD-BTBLLE
C
C--CALCULATE BLADE MEAN CAMBER ANGLE AT TRAILING EDGE
      I = ITE(J)+1
      EXTRAP = SOM(I,J)-STEOM(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,NPPC,NBLPC,51,51,ZTEOMR(J),RTEOMR(J),
     1DTDZTE,II,JJ)
      CALL LININT(ZPC,RPC,DTHDR,NPPC,NBLPC,51,51,ZTEOMR(J),RTEOMR(J),
     1DTDRTE,II,JJ)
      TANBBL = RTEOM(J)*(DTDRTE*SIN(ALPHTE)+DTDZTE*COS(ALPHTE))
      IF (ITER.EQ.1) BTBFTE(J)=ATAN(TANBBL)
      BTBLTE = ATAN(TANBBL)*DEGRAD
C
C--CALCULATE BLADE FLOW ANGLE AT TRAILING EDGE, CORRECTED FOR BLOCKAGE
      BETAFS = BETA(I,J)+EXTRAP*(BETA(I,J)-BETA(I+1,J))/(SOM(I+1,J)-
     1SOM(I,J))
      RHOFS = RHO(I,J)+EXTRAP*(RHO(I,J)-RHO(I+1,J))/
     1(SOM(I+1,J)-SOM(I,J))
      RHOBF = RHOAV(I-1,J)+(SOM(I,J)-SOM(I-1,J)-EXTRAP)/
     1(SOM(I-1,J)-SOM(I-2,J))*(RHOAV(I-1,J)-RHOAV(I-2,J))
      TANBBF = TAN(BETAFS)*BTHTE(J)/PITCH*RHOBF/RHOFS
      BETABF = ATAN(TANBBF)
C
C--CALCULATE DISTANCE FOR DTHDS CORRECTION
      SLIDTE = BLDCRD/PITCH/RTEOM(J)
      DISTTE = AMIN1(.5,AMAX1(1./6.,(11.-4.*SLIDTE)/18.))*(STEOM(J)-
     1SLEOM(J))
C
C--CORRECT DTHDS FOR DEVIATION NEAR THE TRAILING EDGE,
C--USING LINEAR CORRECTION FOR ANGLE
      I = ITE(J)
   70 SDIST = SOM(I,J)-STEOM(J)+DISTTE
      IF (SDIST.LE.0.) GO TO 80
      TANBIJ = ROM(I,J)*(DTHDS(I,J)*CAMP(I,J)+DTHDT(I,J)*SAMP(I,J))
      BETAIJ = ATAN(TANBIJ)
      BETAIJ = BETAIJ+(BETABF-BTBFTE(J))*SDIST/DISTTE
      TANBIJ = TAN(BETAIJ)
      DTHDS(I,J) = (TANBIJ/ROM(I,J)-DTHDT(I,J)*SAMP(I,J))/CAMP(I,J)
      I = I-1
      GO TO 70
   80 BTBFTE(J) = BETABF
C
C--CALCULATE DEVIATION ANGLES
      BLDEV = BETABF*DEGRAD-BTBLTE
      UBDEV = BETAFS*DEGRAD-BTBLTE
C
C--PRINT INCIDENCE AND DEVIATION ANGLES
      IF (IPRNT.EQ.0) GO TO 100
      IF ((LAMDAF(.5,ILE(1),1)-RVTHTA(.5,ILE(1),1)).GT.0.) GO TO 90
      BLINC = -BLINC
```

```
      UBINC = -UBINC
      BLDEV = -BLDEV
      UBDEV = -UBDEV
   90 WRITE(NWRT6,1040) J,BLINC,UBINC,BTBLLE,BLDEV,UBDEV,BTBLTE
  100 CONTINUE
      IF (IPRNT.GT.0) WRITE(NWRT6,1050)
      RETURN
C
C--FORMAT STATEMENTS
C
 1000 FORMAT (1H1////44X,40(1H*)/44X,40H***  INCIDENCE AND DEVIATION ANG
     1LES   ***/49X,30(1H*)//)
 1010 FORMAT (/53X,23(1H*)/53X,23H*  REDUCED  MASSFLOW  */53X,23(1H*)/
     153X,18H*  ITERATION  NO. ,I2,3H  */53X,23(1H*))
 1020 FORMAT (/52X,25(1H*)/52X,25H*    FULL   MASSFLOW    */52X,25(1H*)/
     152X,19H*   ITERATION  NO. ,I2,4H   */52X,25(1H*))
 1030 FORMAT (//24X,10H*  MESH  *,8X,9HINCIDENCE,7X,11HBLADE ANGLE,2H *,
     18X,9HDEVIATION,7X,11HBLADE ANGLE,2H */24X,10H*  LINE  *,3X,
     27HBLOCKED,3X,9HUNBLOCKED,4X,7HAT L.E.,3X,1H*,3X,7HBLOCKED,3X,
     39HUNBLOCKED,4X,7HAT T.E.,3X,1H*)
 1040 FORMAT (24X,1H*,2X,I3,3X,2(1H*,3(F9.2,2X),3X),1H*)
 1050 FORMAT (1H1)
      END




      SUBROUTINE TSONIN
C
C--TSONIN CALCULATES AND PRINTS OUT DATA AS INPUT TO THE
C--TSONIC  BLADE-TO-BLADE ANALYSIS PROGRAM
C
      COMMON NREAD,NWRIT,ITER,IEND,NWRT1,NWRT2,NWRT3,NWRT4,NWRT5,NWRT6
      COMMON/INPUTT/GAM,AR,MSFI,OMEGA,REDFAC,VELTOL,FNEW,DNEW,MBI,MBO,
     1   MM,MHT,NBL,NHUB,NTIP,NIN,NOUT,NBLPL,NPPP,NOSTAT,NSL,NLOSS,
     2   LSFR,LTPL,LAMVT,LROT,LBLAD,LFTEAN,ANGROT,IMESH,ISLINE,
     3   ISTATL,IPLOT,ISUPER,ITSON,IDEBUG,ZOMIN,ZOMBI,ZOMBO,ZOMOUT,
     4   ROMIN,ROMBI,ROMBO,ROMOUT,ZHIN,ZTIN,ZHOUT,ZTOUT,RHIN,RTIN,RHOUT,
     5   RTOUT,TITLEI(20),ZHUB(50),RHUB(50),ZTIP(50),RTIP(50),SFIN(50),
     6   RADIN(50),TIP(50),PRIP(50),LAMIN(50),VTHIN(50),SFOUT(50),
     7   RADOUT(50),PROP(50),LOSOUT(50),LAMOUT(50),VTHOUT(50),
     8   BETALE(50),BETATE(50),ZHST(50),ZTST(50),RHST(50),RTST(50),
     9   FLFR(50),PERCRD(50),PERLOS(50),ZBL(50,50),RBL(50,50),
     1   THBL(50,50),TNBL(50,50),TTBL(50,50),TH1BL(50,50),TH2BL(50,50)
      COMMON/CALCON/MMM1,MHTP1,CP,EXPON,TGROG,PITCH,RLEH,RLET,RTEH,RTET,
     1   ZLE(50),RLE(50),ZTE(50),RTE(50),ZLEOM(101),RLEOM(101),
     2   SLEOM(101),THLEOM(101),ZTEOM(101),RTEOM(101),STEOM(101),
     3   THTEOM(101),ILE(101),ITE(101),ZOM(100,101),ROM(100,101),
     4   SOM(100,101),TOM(100,101),BTH(100,101),DTHDS(100,101),
     5   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),FT(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/INDCOM/NBLPC,NPPC,ZPC(51,51),RPC(51,51),TTPC(51,51),
     1    THPC(51,51),DTHDZ(51,51),DTHDR(51,51),BTHLE(101),BTHTE(101),
     2    BTBFLE(101),BTBFTE(101)
      COMMON/ROTATN/ZHROT(50),RHROT(50),ZTROT(50),RTROT(50),
     1    ZLEOMR(101),RLEOMR(101),ZTEOMR(101),RTEOMR(101),
     2    ZBLROT(50,50),RBLROT(50,50),ZOMROT(100,101),ROMROT(100,101)
      DIMENSION ZMSP1(100),ZMSP2(100),THSP1(100),THSP2(100),DTDM1(100),
     1    DTDM2(100),D2TDM1(100),D2TDM2(100),CURV1(100),CURV2(100),
     2    RADSP1(100),RADSP2(100),ALPHSP(100),ZMRSP(100),RMSP(100),
     3    BESP(100),PLOSSL(100),DBDM(100),D2BDM2(100),DIST(100),
     4    AAA(100),BBB(100)
      REAL MSFL,MSL,LAMDAF
C
C--PRELIMINARY CALCULATIONS
C
      IF (ITSON.LE.0) RETURN
      IF ((ITER/ITSON)*ITSON.NE.ITER) RETURN
      WRITE(NWRT4,1000)
      IF (REDFAC.LT.1.0) WRITE(NWRT4,1010) ITER
      IF (REDFAC.EQ.1.0.AND.IEND.LE.0) WRITE(NWRT4,1020) ITER
      IF (REDFAC.EQ.1.0.AND.IEND.GE.1.AND.ISUPER.LE.1) WRITE(NWRT4,1030)
      IF (REDFAC.EQ.1.0.AND.IEND.GE.1.AND.ISUPER.EQ.2) WRITE(NWRT4,1040)
      ARTEM = AR
      ZMSFL = MSFL/100./REDFAC
      OMTEM = OMEGA/REDFAC
      REDTEM = 1.0
      VELTEM = .01
      MBITS = 21
      MBOTS = 61
      MMTS = 81
      MBBI = 20
      NRSP = MM
      NSLTS = 0
      LRVB = 0
      LOSS = 0
      IF (NLOSS.GT.0) LOSS=1
      LWCR = 1
      LIPS = 0
      IIMESH = 0
      IISLIN = 5
      IIBSUR = 1
      IIPLOT = 1
      IIDEBG = 0
      DEGRAD = 180./3.1415927
      BFACTR = 1.0
C
C--CALCULATE AND PRINT OUT TSONIC DATA ALONG EACH OF THE STREAMLINES
C--ONE STREAMLINE AT A TIME
C
      DO 310 JS=1,NSL
      II = 1
      JJ = 1
      TIPTEM = TIPF(FLFR(JS))
      RHOIP = RHOIPF(FLFR(JS))
      RVTHI = LAMDAF(FLFR(JS),ILE(1),1)/REDFAC
      RVTHO = RVTHTA(FLFR(JS),ILE(1),1)/REDFAC
      IF (GAM.NE.0.) GO TO 5
      ARTEM = 0.
```

```
      TIPTEM = 0.
      RHOIP = AR
C
C--INTERSECTION OF STREAMLINE WITH BLADE LEADING AND TRAILING EDGES
C
    5 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
C--CALCULATE STREAMSHEET LOCATION AND THICKNESS, AND LOSS DISTRIBUTION
C
      CALL ROTATE(-ANGROT,ZSL(1,JS),RSL(1,JS),MM,1,100,1,DIST,RMSP)
      DO 10 IS=1,MM
      ZMRSP(IS) = MSL(IS,JS)-MSL(1,JS)
      CALL LININT(ZOMROT,ROMROT,RHOAV,MM,MHTP1,100,101,ZSL(IS,JS),
     1RSL(IS,JS),RHOSL,II,JJ)
      CALL LININT(ZOMROT,ROMROT,BTH,MM,MHTP1,100,101,ZSL(IS,JS),
     1RSL(IS,JS),BTHSL,II,JJ)
      CALL LININT(ZOMROT,ROMROT,PLOSS,MM,MHTP1,100,101,ZSL(IS,JS),
     1RSL(IS,JS),PLOSSL(IS),II,JJ)
      CALL LININT(ZOMROT,ROMROT,DELRHO,MM,MHTP1,100,101,ZSL(IS,JS),
     1RSL(IS,JS),DRHOSL,II,JJ)
      CALL LININT(ZOMROT,ROMROT,WLSURF,MM,MHTP1,100,101,ZSL(IS,JS),
     1RSL(IS,JS),WLSFSL,II,JJ)
      CALL LININT(ZOMROT,ROMROT,WTSURF,MM,MHTP1,100,101,ZSL(IS,JS),
     1RSL(IS,JS),WTSFSL,II,JJ)
      ROWMAV = RHOSL*WMSL(IS,JS)+DRHOSL/12.*COS(BETSL(IS,JS))*
     1(WLSFSL-WTSFSL)
   10 BESP(IS) = ZMSFL/ROWMAV/RMSP(IS)/BTHSL*BFACTR
      ZMSFL = ZMSFL*BFACTR
C
C--CALCULATE BLADE SURFACE COORDINATES WITH RESPECT TO MERIDL ORIGIN
C--AT ALL POINTS ON BLADE WHERE VERTICAL ORTHOGONALS PASS THROUGH
C--THE STREAMLINE
C
      II = 1
      JJ = 1
      NBLPTS = ITS(JS)-ILS(JS)+3
      ILSJ = ILS(JS)
      ITSJ = ITS(JS)
      ZMSP1(1) = 0.
      DELM = SQRT((ZSL(ILSJ,JS)-ZLESL)**2+(RSL(ILSJ,JS)-RLESL)**2)
      CALL LININT(ZPC,RPC,THPC,NPPC,NBLPC,51,51,ZLESL,RLESL,
     1THLESL,II,JJ)
      ISB = 2
      DO 20 IS=ILSJ,ITSJ
      ZMSP1(ISB) = ZMRSP(IS)-ZMRSP(ILSJ)+DELM
      CALL LININT(ZPC,RPC,THPC,NPPC,NBLPC,51,51,ZSL(IS,JS),
     1RSL(IS,JS),THSL,II,JJ)
      CALL LININT(ZPC,RPC,TTPC,NPPC,NBLPC,51,51,ZSL(IS,JS),
     1RSL(IS,JS),DBL,II,JJ)
      THSP1(ISB) = THSL-THLESL+DBL/2.
      THSP2(ISB) = THSL-THLESL-DBL/2.
   20 ISB = ISB+1
      DELM = SQRT((ZTESL-ZSL(ITSJ,JS))**2+(RTESL-RSL(ITSJ,JS))**2)
      ZMSP1(NBLPTS) = ZMSP1(NBLPTS-1)+DELM
      CHORD = ZMSP1(NBLPTS)
      CALL LININT(ZPC,RPC,THPC,NPPC,NBLPC,51,51,ZTESL,RTESL,
     1THTESL,II,JJ)
      CALL LININT(ZPC,RPC,TTPC,NPPC,NBLPC,51,51,ZLESL,RLESL,
```

```
     1DBL,II,JJ)
      IF (DBL.LT.CHORD/1000.) DBL=CHORD/1000.
      THSP1(1) =  DBL/2.
      THSP2(1) = -DBL/2.
      CALL LININT(ZPC,RPC,TTPC,NPPC,NBLPC,51,51,ZTESL,RTESL,
     1DBL,II,JJ)
      IF (DBL.LT.CHORD/1000.) DBL=CHORD/1000.
      THSP1(NBLPTS) = THTESL-THLESL+DBL/2.
      THSP2(NBLPTS) = THTESL-THLESL-DBL/2.
      DO 25 IS=1,NBLPTS
   25 ZMSP2(IS) = ZMSP1(IS)
C
C--SHIFT STREAMSHEET MERIDIONAL COORDINATES TO ORIGIN AT BLADE
C--LEADING EDGE,
C--AND CALCULATE FIRST AND SECOND DERIVATIVES OF STREAMSHEET
C
      DELM = ZMRSP(ILSJ)-ZMSP1(2)
      DO 30 IS=1,MM
   30 ZMRSP(IS) = ZMRSP(IS)-DELM
      CALL SPLINE(ZMRSP,BESP,MM,DBDM,D2BDM2)
C
C--ELIMINATE ANY BLADE SURFACE POINTS VERY CLOSE TO THE
C--LEADING AND TRAILING EDGES
C
      ILSJ1 = ILS(JS)
      ILSJ2 = ILS(JS)
      DELMSP = 0.10*CHORD/FLOAT(NBLPTS-1)
   40 IF ((ZMSP1(2)-ZMSP1(1)).GT.DELMSP) GO TO 60
      DO 50 IS=3,NBLPTS
      ZMSP1(IS-1) = ZMSP1(IS)
      ZMSP2(IS-1) = ZMSP2(IS)
      THSP1(IS-1) = THSP1(IS)
   50 THSP2(IS-1) = THSP2(IS)
      NBLPTS = NBLPTS-1
      ILSJ1 = ILSJ1+1
      ILSJ2 = ILSJ2+1
      GO TO 40
   60 IF ((ZMSP1(NBLPTS)-ZMSP1(NBLPTS-1)).GT.DELMSP) GO TO 70
      ZMSP1(NBLPTS-1) = ZMSP1(NBLPTS)
      ZMSP2(NBLPTS-1) = ZMSP2(NBLPTS)
      THSP1(NBLPTS-1) = THSP1(NBLPTS)
      THSP2(NBLPTS-1) = THSP2(NBLPTS)
      NBLPTS = NBLPTS-1
      GO TO 60
C
C--CALCULATE GRADIENTS ON BOTH BLADE SURFACES
C--CALCULATE RADII FROM CENTERLINE, AT LEADING AND TRAILING EDGES
C
   70 CALL SPLINE(ZMSP1,THSP1,NBLPTS,DTDM1,D2TDM1)
      CALL SPLINE(ZMSP2,THSP2,NBLPTS,DTDM2,D2TDM2)
      CALL SPLINT(ZMRSP,RMSP,MM,0.,1,RADLE,TEM1,TEM2)
      CALL SPLINT(ZMRSP,RMSP,MM,CHORD,1,RADTE,TEM1,TEM2)
C
C--CALCULATE LEADING EDGE RADIUS, POINTS OF TANGENCY, AND
C--TANGENCY ANGLES
C
      ICOUNT = 0
      DAMP = 1.
   80 BETI1 = ATAN(RADLE*DTDM1(1))
```

```
      BETI2 = ATAN(RADLE*DTDM2(1))
      RI1 = RADLE*(THSP1(1)-THSP2(1))*COS((BETI1+BETI2)/2.)/2.
   90 ZLTAN1 = RI1*(1.-SIN(BETI1))
      ZLTAN2 = RI1*(1.+SIN(BETI2))
      CALL SPLINT(ZMSP1,THSP1,NBLPTS,ZLTAN1,1,TLTAN1,DTAN1,TEM1)
      CALL SPLINT(ZMSP2,THSP2,NBLPTS,ZLTAN2,1,TLTAN2,DTAN2,TEM2)
      BETI1 = ATAN(RADLE*DTAN1)
      BETI2 = ATAN(RADLE*DTAN2)
      RI1NEW = RADLE*(TLTAN1-TLTAN2)/(COS(BETI1)+COS(BETI2))
      IF (ABS((RI1NEW-RI1)/RI1).LT..001) GO TO 110
      ICOUNT = ICOUNT+1
      IF (ICOUNT.LE.100) GO TO 100
      WRITE(NWRT4,1200)
      GO TO 110
  100 RI1 = (DAMP*RI1+RI1NEW)/(DAMP+1.)
      IF (RI1.GT.0.) GO TO 90
      DAMP = DAMP+1.
      GO TO 80
  110 RI1 = RI1NEW
      RI2 = RI1
C
C--CALCULATE TRAILING EDGE RADIUS, POINTS OF TANGENCY, AND
C--TANGENCY ANGLES
C
      ICOUNT = 0
      DAMP = 1.
  120 BETO1 = ATAN(RADTE*DTDM1(NBLPTS))
      BETO2 = ATAN(RADTE*DTDM2(NBLPTS))
      RO1 = RADTE*(THSP1(NBLPTS)-THSP2(NBLPTS))*COS((BETO1+BETO2)/2.)/2.
  130 ZTTAN1 = CHCRD-RO1*(1.+SIN(BETO1))
      ZTTAN2 = CHCRD-RO1*(1.-SIN(BETO2))
      CALL SPLINT(ZMSP1,THSP1,NBLPTS,ZTTAN1,1,TTTAN1,DTAN1,TEM1)
      CALL SPLINT(ZMSP2,THSP2,NBLPTS,ZTTAN2,1,TTTAN2,DTAN2,TEM2)
      BETO1 = ATAN(RADTE*DTAN1)
      BETO2 = ATAN(RADTE*DTAN2)
      RO1NEW = RADTE*(TTTAN1-TTTAN2)/(COS(BETO1)+COS(BETO2))
      IF (ABS((RO1NEW-RO1)/RO1).LT..001) GO TO 150
      ICOUNT = ICOUNT+1
      IF (ICOUNT.LE.100) GO TO 140
      WRITE(NWRT4,1210)
      GO TO 150
  140 RO1 = (DAMP*RO1+RO1NEW)/(DAMP+1.)
      IF (RO1.GT.0.) GO TO 130
      DAMP = DAMP+1.
      GO TO 120
  150 RO1 = RO1NEW
      RO2 = RO1
C
C--SUBSTITUTE POINTS OF TANGENCY FOR FIRST AND LAST POINTS IN
C--SURFACE COORDINATE ARRAYS
C
      ZMSP1(1) = ZLTAN1
      ZMSP2(1) = ZLTAN2
      ZMSP1(NBLPTS) = ZTTAN1
      ZMSP2(NBLPTS) = ZTTAN2
      THSP1(1) = TLTAN1
      THSP2(1) = TLTAN2
      THSP1(NBLPTS) = TTTAN1
      THSP2(NBLPTS) = TTTAN2
      NSPL1 = NBLPTS
```

```
      NSPL2 = NBLPTS
C
C--ELIMINATE SURFACE POINTS BETWEEN BLADE EDGES AND TANGENCY POINTS
C--ALSO ELIMINATE ANY SURFACE POINTS TOO CLOSE TO TANGENCY POINTS
C
      DELMSP = 0.10*CHORD/FLOAT(NBLPTS-1)
  160 IF (ZMSP1(2).GT.ZMSP1(1)+DELMSP) GO TO 180
      DO 170 IS=3,NSPL1
      ZMSP1(IS-1) = ZMSP1(IS)
  170 THSP1(IS-1) = THSP1(IS)
      NSPL1 = NSPL1-1
      ILSJ1 = ILSJ1+1
      GO TO 160
  180 IF (ZMSP2(2).GT.ZMSP2(1)+DELMSP) GO TO 200
      DO 190 IS=3,NSPL2
      ZMSP2(IS-1) = ZMSP2(IS)
  190 THSP2(IS-1) = THSP2(IS)
      NSPL2 = NSPL2-1
      ILSJ2 = ILSJ2+1
      GO TO 180
  200 IF (ZMSP1(NSPL1-1).LT.ZMSP1(NSPL1)-DELMSP) GO TO 210
      ZMSP1(NSPL1-1) = ZMSP1(NSPL1)
      THSP1(NSPL1-1) = THSP1(NSPL1)
      NSPL1 = NSPL1-1
      GO TO 200
  210 IF (ZMSP2(NSPL2-1).LT.ZMSP2(NSPL2)-DELMSP) GO TO 220
      ZMSP2(NSPL2-1) = ZMSP2(NSPL2)
      THSP2(NSPL2-1) = THSP2(NSPL2)
      NSPL2 = NSPL2-1
      GO TO 210
C
C--CALCULATE TANGENTIAL COORDINATE SHIFT FROM MERIDL ORIGIN TO
C--TSONIC ORIGIN,  AND SHIFT COORDINATES
C
  220 DELTH = (TLTAN1*COS(BETI2)+TLTAN2*COS(BETI1))/(COS(BETI1)+
     1COS(BETI2))
      DO 230 IS=1,NSPL1
  230 THSP1(IS) = THSP1(IS)-DELTH
      DO 240 IS=1,NSPL2
  240 THSP2(IS) = THSP2(IS)-DELTH
C
C--CALCULATE STAGGER AND STACKING COORDINATE
C
      STGR = (TTTAN1*COS(BETO2)+TTTAN2*COS(BETO1))/(COS(BETO1)+
     1COS(BETO2))-DELTH
      THSTAK = THLESL+DELTH
C
C--CALCULATE RADII FROM CENTERLINE TO BLADE SURFACE POINTS
C
      CALL SPLINT(ZMRSP,RMSP,MM,ZMSP1,NSPL1,RADSP1,AAA,BBB)
      CALL SPLINT(ZMRSP,RMSP,MM,ZMSP2,NSPL2,RADSP2,AAA,BBB)
C
C--CALCULATE SLOPES, SECOND DERIVATIVES, AND CURVATURES ON UPPER
C--BLADE SURFACE
C
      SLOPE1 = TAN(BETI1)/RADSP1(1)
      SLOPEN = TAN(BETO1)/RADSP1(NSPL1)
      CALL SPLISL(ZMSP1,THSP1,NSPL1,SLOPE1,SLOPEN,DTDM1,D2TDM1)
      BETI1 = BETI1*DEGRAD
```

```
      BETO1 = BETO1*DEGRAD
      TMSL = ZMSP1(1)-ZMRSP(1)
      CALL SPLINT(MSL(1,JS),ALPSL(1,JS),MM,TMSL,1,ALPHSP(1),TEM1,TEM2)
      NSPLM = NSPL1-1
      DO 250 IS=2,NSPLM
      ITEM = ILSJ1+IS-2
  250 ALPHSP(IS) = ALPSL(ITEM,JS)
      TMSL = ZMSP1(NSPL1)-ZMRSP(1)
      CALL SPLINT(MSL(1,JS),ALPSL(1,JS),MM,TMSL,1,ALPHSP(NSPL1),
     1TEM1,TEM2)
      DO 260 IS=1,NSPL1
  260 CURV1(IS) = (RADSP1(IS)*D2TDM1(IS)+SIN(ALPHSP(IS))*DTDM1(IS))/
     1(1.+(RADSP1(IS)*DTDM1(IS))**2)**1.5
C
C--CALCULATE SLOPES, SECOND DERIVATIVES, AND CURVATURES ON LOWER
C--BLADE SURFACE
C
      SLOPE1 = TAN(BETI2)/RADSP2(1)
      SLOPEN = TAN(BETO2)/RADSP2(NSPL2)
      CALL SPLISL(ZMSP2,THSP2,NSPL2,SLOPE1,SLOPEN,DTDM2,D2TDM2)
      BETI2 = BETI2*DEGRAD
      BETO2 = BETO2*DEGRAD
      TMSL = ZMSP2(1)-ZMRSP(1)
      CALL SPLINT(MSL(1,JS),ALPSL(1,JS),MM,TMSL,1,ALPHSP(1),TEM1,TEM2)
      NSPLM = NSPL2-1
      DO 270 IS=2,NSPLM
      ITEM = ILSJ2+IS-2
  270 ALPHSP(IS) = ALPSL(ITEM,JS)
      TMSL = ZMSP2(NSPL2)-ZMRSP(1)
      CALL SPLINT(MSL(1,JS),ALPSL(1,JS),MM,TMSL,1,ALPHSP(NSPL2),
     1TEM1,TEM2)
      DO 280 IS=1,NSPL2
  280 CURV2(IS) = (RADSP2(IS)*D2TDM2(IS)+SIN(ALPHSP(IS))*DTDM2(IS))/
     1(1.+(RADSP2(IS)*DTDM2(IS))**2)**1.5
C
C--PRINT TSONIC DATA
C
      WRITE(NWRT4,1050) JS,FLFR(JS)
      IF (BFACTR.NE.1.0) WRITE(NWRT4,1055)
      WRITE(NWRT4,1060)
      WRITE(NWRT4,1310) GAM,ARTEM,TIPTEM,RHOIP,OMTEM,ZMSFL
      WRITE(NWRT4,1070)
      WRITE(NWRT4,1340) VELTEM
      WRITE(NWRT4,1080)
      WRITE(NWRT4,1320) NBL,NSPL1,NSPL2,NRSP
      WRITE(NWRT4,1090)
      WRITE(NWRT4,1330) LRVB,LOSS,LWCR,LIPS
      WRITE(NWRT4,1100)
      WRITE(NWRT4,1310) RVTHI,RVTHO
      WRITE(NWRT4,1110)
      WRITE(NWRT4,1350) CHORD,STGR,DELTH,THSTAK
      WRITE(NWRT4,1120)
      WRITE(NWRT4,1360) RI1,BETI1,RO1,BETO1
      WRITE(NWRT4,1130)
      WRITE(NWRT4,1370) (IS,ZMSP1(IS),THSP1(IS),DTDM1(IS),D2TDM1(IS),
     1CURV1(IS),RADSP1(IS),IS=1,NSPL1)
      WRITE(NWRT4,1140)
      WRITE(NWRT4,1360) RI2,BETI2,RO2,BETO2
      WRITE(NWRT4,1150)
```

```
      WRITE(NWRT4,1370)  (IS,ZMSP2(IS),THSP2(IS),DTDM2(IS),D2TDM2(IS),
     1CURV2(IS),RADSP2(IS),IS=1,NSPL2)
      WRITE(NWRT4,1160)
      WRITE(NWRT4,1380)  (IS,ZMRSP(IS),RMSP(IS),BESP(IS),WWCRSL(IS,JS),
     1PLOSSL(IS),DBDM(IS),D2BDM2(IS),IS=1,MM)
      WRITE(NWRT4,1170)
      WRITE(NWRT4,1300)
C
C--WRITE OUTPUT AGAIN IN CARD IMAGE FORMAT
C
  300 ICARDS = 0
      NWRT7 = 6
      IF (ICARDS.EQ.0) GO TO 310
      WRITE(NWRT7,1400) JS,FLFR(JS)
      WRITE(NWRT7,1450) GAM,ARTEM,TIPTEM,RHOIP,OMTEM,ZMSFL
      WRITE(NWRT7,1460) REDTEM,VELTEM
      WRITE(NWRT7,1440) MBITS,MBOTS,MMTS,MBBI,NBL,NSPL1,NSPL2,NRSP,NSLTS
      WRITE(NWRT7,1440) LRVB,LCSS,LWCR,LIPS
      WRITE(NWRT7,1460) RVTHI,RVTHO
      WRITE(NWRT7,1470) CHORD,STGR
      WRITE(NWRT7,1480) RI1,BETI1,RO1,BETO1
      WRITE(NWRT7,1420) (ZMSP1(IS),IS=1,NSPL1)
      WRITE(NWRT7,1430) (THSP1(IS),IS=1,NSPL1)
      WRITE(NWRT7,1480) RI2,BETI2,RO2,BETO2
      WRITE(NWRT7,1420) (ZMSP2(IS),IS=1,NSPL2)
      WRITE(NWRT7,1430) (THSP2(IS),IS=1,NSPL2)
      WRITE(NWRT7,1420) (ZMRSP(IS),IS=1,MM)
      WRITE(NWRT7,1420) (RMSP(IS),IS=1,MM)
      WRITE(NWRT7,1430) (BESP(IS),IS=1,MM)
      WRITE(NWRT7,1410) (WWCRSL(IS,JS),IS=1,MM)
      WRITE(NWRT7,1410) (PLOSSL(IS),IS=1,MM)
      WRITE(NWRT7,1440) IIMESH,IISLIN,IIBSUR,IIPLOT,IIDEBG
      IF (NWRT7.EQ.NWRT4) WRITE(NWRT4,1300)
  310 CONTINUE
      RETURN
C
C--FORMAT STATEMENTS
C
 1000 FORMAT (///45X,39(1H*)/45X,39H***  INPUT DATA FOR TSONIC PROGRAM
     1***/50X,29(1H*)//)
 1010 FORMAT (/53X,23(1H*)/53X,23H*  REDUCED  MASSFLOW  */53X,23(1H*)/
     153X,18H*  ITERATION  NO. ,I2,3H  */53X,23(1H*)////)
 1020 FORMAT (/52X,25(1H*)/52X,25H*    FULL    MASSFLOW    */52X,25(1H*)/
     152X,19H*   ITERATION  NO. ,I2,4H    */52X,25(1H*)////)
 1030 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*)////)
 1040 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*)////)
 1050 FORMAT (2X,76(1H*)/2X,38H*  TSONIC INPUT  --  STREAMLINE NUMBER,I3
     1,23H  --  STREAM FUNCTION =,F8.4,4H   */2X,110(1H*)/2X,110H*  NOTE
     2 -- THE ORIGIN FOR MERIDIONAL AND TANGENTIAL COORDINATES ON THIS B
     3LADE SECTION IS THE TSONIC ORIGIN,  */2X,93H*  THAT IS, THE FARTHE
     4ST POINT UPSTREAM ON THE LEADING EDGE RADIUS.   THE MERIDL COORDIN
     5ATES,16X,1H*/2X,95H*  WHICH HAVE A DIFFERENT ORIGIN, HAVE BEEN SHI
     6FTED BY THE PROGRAM TO GIVE THESE TSONIC INPUTS.,14X,1H*/2X,
     7110(1H*)//)
```

```
1055 FORMAT (2X,116(1H*)/2X,99H*   NOTE  --  ZMSFL AND THE BESP ARRAY IN
    1 THE FOLLOWING OUTPUT HAVE BOTH BEEN MULTIPLIED BY BFACTR =,G14.7,
    23H   */2X,116(1H*)///)
1060 FORMAT (5X,3HGAM,13X,2HAR,13X,3HTIP,11X,5HRHOIP,10X,5HOMEGA,12X,
    15HZMSFL)
1070 FORMAT (5X,6HREDFAC,8X,6HVELTOL)
1080 FORMAT (5X,53HMBI    MBO    MM    MBBI    NBL   NSPL1   NSPL2   NRSP    NS
    1L)
1090 FORMAT (5X,22HLRVB  LOSS  LWCR  LIPS)
1100 FORMAT (5X,5HRVTHI,10X,5HRVTHO,11X,4HFSMI,11X,4HFSMO,11X,4HSSM1,
    111X,4HSSM2)
1110 FORMAT (5X,5HCHORD,10X,4HSTGR,42X,5HDELTA,10X,6HTHSTAK)
1120 FORMAT (//5X,25HBLADE  SURFACE  1    *****,15X,3HRI1,12X,5HBETI1,
    110X,3HRO1,11X,5HBETO1)
1130 FORMAT (//5X,5HPOINT,6X,5HZMSP1,10X,5HTHSP1,6X,10HDERIVATIVE,3X,
    114H2ND DERIVATIVE,3X,9HCURVATURE,8X,6HRADIUS)
1140 FORMAT (//5X,25HBLADE  SURFACE  2    *****,15X,3HRI2,12X,5HBETI2,
    110X,3HRO2,11X,5HBETO2)
1150 FORMAT (//5X,5HPOINT,6X,5HZMSP2,10X,5HTHSP2,6X,10HDERIVATIVE,3X,
    114H2ND DERIVATIVE,3X,9HCURVATURE,8X,6HRADIUS)
1160 FORMAT (//5X,29HSTREAM  CHANNEL  DATA   *****//5X,5HPOINT,6X,
    15HZMRSP,10X,4HRMSP,10X,4HBESP,10X,5HWCWCR,10X,5HPLOSS,19X,4HDBDM,
    210X,6HD2BDM2)
1170 FORMAT (//5X,36HIMESH  ISLINE  IBSURF  IPLOT  IDEBUG)
1200 FORMAT (/2X,62HA LEADING EDGE RADIUS COULD NOT BE OBTAINED IN 100
    1ITERATIONS./2X,64HRI1, RI2, BETI1, BETI2, STGR, DELTA, AND THSTAK
    2MAY BE IN ERROR////)
1210 FORMAT (/2X,63HA TRAILING EDGE RADIUS COULD NOT BE OBTAINED IN 100
    1 ITERATIONS./2X,49HRO1, RO2, BETO1, BETO2, AND STGR MAY BE IN ERRO
    2R.////)
1300 FORMAT (1H1)
1310 FORMAT (1X,G14.7,7G15.7)
1320 FORMAT (27X,I4,2X,I4,3X,I4,3X,I4)
1330 FORMAT (2X,4I6)
1340 FORMAT (19X,F6.4)
1350 FORMAT (1X,G14.7,1X,G14.7,32X,2G15.7)
1360 FORMAT (40X,4G15.7)
1370 FORMAT (3X,I5,2X,6G15.7)
1380 FORMAT (3X,I5,2X,5G15.7,8X,2G15.7)
1400 FORMAT (5X,17HSTREAMLINE NUMBER,I3,23H   --   STREAM FUNCTION =,
    1F8.4)
1410 FORMAT (8F10.5)
1420 FORMAT (8F10.6)
1430 FORMAT (8F10.7)
1440 FORMAT (16I5)
1450 FORMAT (F10.5,2F10.4,F10.7,F10.3,F10.8)
1460 FORMAT (2F10.4)
1470 FORMAT (F10.6,F10.7)
1480 FORMAT (F10.7,F10.4,F10.7,F10.4)
     END



     SUBROUTINE SLPLOT
C
C--SLPLOT PLOTS THE STREAMLINES IN THE HUB-SHROUD FLOW PLANE
C
```

```
      COMMON NREAD,NWRIT,ITER,IEND,NWRT1,NWRT2,NWRT3,NWRT4,NWRT5,NWRT6
      COMMON/INPUTT/GAM,AR,MSFL,OMEGA,REDFAC,VELTOL,FNEW,DNEW,MBI,MBO,
     1   MM,MHT,NBL,NHUB,NTIP,NIN,NOUT,NBLPL,NPPP,NOSTAT,NSL,NLOSS,
     2   LSFR,LTPL,LAMVT,LROT,LBLAD,LETEAN,ANGROT,IMESH,ISLINE,
     3   ISTATL,IPLOT,ISUPER,ITSON,IDEBUG,ZOMIN,ZOMBI,ZOMBO,ZOMOUT,
     4   ROMIN,ROMBI,ROMBO,ROMOUT,ZHIN,ZTIN,ZHOUT,ZTOUT,RHIN,RTIN,RHOUT,
     5   RTOUT,TITLEI(20),ZHUB(50),RHUB(50),ZTIP(50),RTIP(50),SFIN(50),
     6   RADIN(50),TIP(50),PRIP(50),LAMIN(50),VTHIN(50),SFOUT(50),
     7   RADOUT(50),PROP(50),LOSOUT(50),LAMOUT(50),VTHOUT(50),
     8   BETALE(50),BETATE(50),ZHST(50),ZTST(50),RHST(50),RTST(50),
     9   FLFR(50),PERCRD(50),PERLOS(50),ZBL(50,50),RBL(50,50),
     1   THBL(50,50),TNBL(50,50),TTBL(50,50),TH1BL(50,50),TH2BL(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)
      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 LRCHSZ(4)
      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)
      IF (MBI.EQ.0) GO TO 5
      CALL LRCURV(ZLPLT,RLPLT,100,2,SYM,0.0)
      CALL LRCURV(ZTPLT,RTPLT,100,2,SYM,0.0)
C--PLOT STREAMLINES
    5 EOP = 0.0
      CALL ROTATE(-ANGROT,ZSL,RSL,MM,NSL,100,50,ZSL,RSL)
      DO 10 JS=1,NSL
      IF (JS.EQ.NSL) EOP=1.0
   10 CALL LRCURV(ZSL(1,JS),RSL(1,JS),MM,2,SYM,EOP)
      CALL LRCURV(ZSL,RSL,0,1,SYM,1.0)
```

```
      SUBROUTINE SVPLOT
C
C--SVPLOT PLOTS THE MEAN STREAM SURFACE AND BLADE SURFACE OUTPUT
C--VELOCITIES ALONG ALL STREAMLINES
C
      COMMON NREAD,NWRIT,ITER,IEND,NWRT1,NWRT2,NWRT3,NWRT4,NWRT5,NWRT6
      COMMON/INPUTT/GAM,AR,MSFL,OMEGA,REDFAC,VELTOL,FNEW,DNEW,MBI,MBO,
     1   MM,MHT,NBL,NHUB,NTIP,NIN,NOUT,NBLPL,NPPP,NOSTAT,NSL,NLOSS,
     2   LSFR,LTPL,LAMVT,LROT,LBLAD,LETEAN,ANGROT,IMESH,ISLINE,
     3   ISTATL,IPLOT,ISUPER,ITSON,IDEBUG,ZOMIN,ZOMBI,ZOMBO,ZOMOUT,
     4   ROMIN,ROMBI,ROMBO,ROMOUT,ZHIN,ZTIN,ZHOUT,ZTOUT,RHIN,RTIN,RHOUT,
     5   RTOUT,TITLEI(20),ZHUB(50),RHUB(50),ZTIP(50),RTIP(50),SFIN(50),
     6   RADIN(50),TIP(50),PRIP(50),LAMIN(50),VTHIN(50),SFOUT(50),
     7   RADOUT(50),PROP(50),LOSOUT(50),LAMOUT(50),VTHOUT(50),
     8   BETALE(50),BETATE(50),ZHST(50),ZTST(50),RHST(50),RTST(50),
     9   FLFR(50),PERCRD(50),PERLOS(50),ZBL(50,50),RBL(50,50),
     1   THBL(50,50),TNBL(50,50),TTBL(50,50),TH1BL(50,50),TH2BL(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$C','1$R1','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 ','VELO','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))
      IF (MBI.EQ.0) GO TO 15
      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))
   15 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
      EOP = 0.0
      IF (MBI.EQ.0) EOP=1.0
      CALL LRCHSZ(4)
      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,EOP)
      IF (MBI.EQ.0) GO TO 40
      ILSJ = ILS(JS)
      MBLD = ITS(JS)-ILS(JS)+1
      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)
      CALL LRCURV(MSL(ILSJ,JS),WTSSL(ILSJ,JS),MBLD,4,SYM,1.0)
   40 CONTINUE
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)
      CALL LRCHSZ(0)
      IF (MBI.EQ.0) RETURN
C
C--PLOT SUCTION SURFACE VELOCITIES FOR ALL STREAMLINES
C
      CALL LRCHSZ(4)
      CALL LRLEGN(TITL4,60,0,1.2,0.7,0.0)
      CALL LRCHSZ(2)
```

```
      CALL LBLEGN(TITL6,24,0,3.4,1.3,0.0)
      CALL LBLEGN(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 LBLEGN(TITL5,64,0,1.2,0.7,0.0)
      CALL LRCHSZ(2)
      CALL LBLEGN(TITL6,24,0,3.4,1.3,0.0)
      CALL LBLEGN(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)
      CALL LRCHSZ(0)
      RETURN
      END



      SUBROUTINE TVELCY
C
C--TVELCY CALCULATES THE FULL MASSFLOW, TRANSONIC SOLUTION
C--USING VELOCITY GRADIENT EQUATIONS
C
      COMMON NREAD,NWRIT,ITER,IEND,NWRT1,NWRT2,NWRT3,NWRT4,NWRT5,NWRT6
      COMMON/INPUTT/GAM,AR,MSFL,OMEGA,REDFAC,VELTOL,FNEW,DNEW,MBI,MBO,
     1   MM,MHT,NBL,NHUB,NTIP,NIN,NOUT,NBLPL,NPPP,NOSTAT,NSL,NLOSS,
     2   LSFR,LTPL,LAMVT,LROT,LBLAD,LETEAN,ANGROT,IMESH,ISLINE,
     3   ISTATL,IPLOT,ISUPER,ITSON,IDEBUG,ZOMIN,ZOMBI,ZOMBO,ZOMOUT,
     4   ROMIN,ROMBI,ROMBO,ROMOUT,ZHIN,ZTIN,ZHOUT,ZTOUT,RHIN,RTIN,RHOUT,
     5   RTOUT,TITLEI(20),ZHUB(50),RHUB(50),ZTIP(50),RTIP(50),SFIN(50),
     6   RADIN(50),TIP(50),PRIP(50),LAMIN(50),VTHIN(50),SFOUT(50),
     7   RADOUT(50),PROP(50),ICSOUT(50),LAMOUT(50),VTHOUT(50),
     8   BETALE(50),BETATE(50),ZHST(50),ZTST(50),RHST(50),RTST(50),
     9   FLFR(50),PERCRD(50),PERLOS(50),ZBL(50,50),RBL(50,50),
     1   THBL(50,50),TNBL(50,50),TTBL(50,50),TH1BL(50,50),TH2BL(50,50)
      COMMON/CALCON/MMM1,MHTP1,CP,EXPON,TGROG,PITCH,RLEH,RLET,RTEH,RTET,
     1   ZLE(50),RLE(50),ZTE(50),RTE(50),ZLEOM(101),RLEOM(101),
     2   SLEOM(101),THLEOM(101),ZTEOM(101),RTEOM(101),STEOM(101),
     3   THTEOM(101),ILE(101),ITE(101),ZOM(100,101),ROM(100,101),
     4   SOM(100,101),TOM(100,101),BTH(100,101),DTHDS(100,101),
     5   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),FT(100,101),DFDM(100,101),
     6    XTOM(100,101),ZETOM(100,101),DLDU(100,101)
      DIMENSION DWMDS(100),DWTDS(100),TVERT(101),
     1    WMVERT(101),WTVERT(101),TWLMP(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),RHOIP(101),UNEW(101),DWMDM(100,101),DWTDM(100,101),
     5    DWMDT(100,101),DWTDT(100,101)
      REAL MSFL,LAMBDA,LAMBDO,LAMOUT,LAMIN,LAMDAF,MAXFLO,MINFLO
      LOGICAL REPEAT
C
C--RESTORE FULL MASS FLOW VALUES, AND REINITIALIZE LAMDAF AND RVTHTA
C
      IEND = IEND+1
      JZ = 1
      IF (ISUPER.EQ.2) JZ=2
      IF (ISUPER.EQ.2) GO TO 55
      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
      IF (MBI.NE.0) CALL RVTNIT
C
C--CALCULATE PARTIALS WITH RESPECT TO T OF WSUBM AND WTH
C
      DO 40 I=1,MM
      DO 30 J=1,MHTP1
      DFDM(I,J)  = DFDM(I,J)/REDFAC
      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 WTH, AND THEN
C--CALCULATE PARTIALS WITH RESPECT TO M OF WSUBM AND WTH
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
      MEAN = MHT/2+1
   55 CHLIM = MSFL*FLOAT(NBL)
      UNEW(1)  = 0.
C
C--INITIALIZE VARIABLES FOR LOOP ON VERTICAL MESH LINES
```

```
C
      LINC = 0
      ICOUNT = 0
      IREVRS = 0
      ITEMIN = MM
      ITEMP = MM
      IMAX = 0
      INCR = 1
      I = 0
C
C--SOLVE VELOCITY GRADIENT EQUATION ON EACH VERTICAL MESH LINE
C
C--BEGINNING OF LOOP ON VERTICAL MESH LINES
   60 I = I+INCR
      IF (I.GT.MM) GO TO 290
      WHUB = W(I,1)/REDFAC
      DELMAX = W(I,MEAN)/20./REDFAC
      WMAX = WHUB
      WMIN = WHUB
      MAXFLO = -1.E10
      MINFLO =  1.E10
      NADD = 0
      NSUB = 0
      NREP = 0
      NCOUNT = 0
C
C--CALCULATE COEFFICIENTS A, B, AND D FOR THE VELOCITY GRADIENT EQUATION
C--INITIALIZE COEFFICIENT C TO ZERO
C
      DO 80 J=1,MHTP1
      LAMBDA(J) = LAMDAF(UOM(I,J),I,J)
      IF (MBI.NE.0) LAMBDO(J) = RVTHTA(UOM(I,J),I,J)
      TIPT(J) = TIPF(UOM(I,J))
      RHOIP(J) = RHOIPF(UOM(I,J))*(1.-PLOSS(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 72
   70 ATVEL(J) = CAMP(I,J)*CURV(I,J)
      DTVEL(J) = DWMDM(I,J)*SAMP(I,J)
C
C--CORRECT FLOW ANGLES AT LEADING AND TRAILING EDGES, ANALOGOUS TO INDEV
C
   72 IF (INCR.LT.0) GO TO 75
      IF (I.EQ.ILE(J)) CALL LINDV(J,LINC,ICOUNT)
      IF (LINC.NE.1) GO TO 80
      IMAX = I
      IREVRS = 1
      ICOUNT = 0
      GO TO 80
   75 IF (I.EQ.ITE(J)) CALL TINDV(ITEMP,J,ICOUNT)
```

```
      ITEMIN = MIN0(ITEMIN,ITEMP)
   80 CONTINUE
      IF (LINC.EQ.1) INCR=-1
C
C--CALCULATE C COEFFICIENT FOR THE VELOCITY GRADIENT EQUATION AND OTHER
C--CONSTANTS FOR CHECKING CONTINUITY -- BEGIN OUTER ITERATION PROCEDURE
C
   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)
      GO TO 110
  100 IF(I.LE.ITE(J)) GO TO 120
      WHIRL = LAMBDO(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) = CAMP(I,J)*ROM(I,J)*BTH(I,J)
C
C--CALCULATE COEFFICIENTS E AND F FOR THE VELOCITY GRADIENT EQUATION
C
      TPP = TIPT(1)-TWLMR(1)/2./CP
      IF(TPP.LT.0.) GO TO 300
      PREL = RHOIP(1)*AR*TIPT(1)*(TPP/TIPT(1))**(GAM*EXPON)
      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 300
      PRELN = RHOIP(J)*AR*TIPT(J)*(TPPN/TIPT(J))**(GAM*EXPON)
      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.
C
C--RESTART OF INNER ITERATION PROCEDURE
  140 IND = 1
C
C--CONTINUATION OF INNER ITERATION PROCEDURE
C--BEGIN VELOCITY GRADIENT SOLUTION AT HUB
  150 W(I,1) = WHUB
      NCOUNT = NCOUNT+1
C
C--CALCULATE RVA AT THE HUB
      WSQ = WHUB**2
      TTIP = 1.-(WSQ+TWLMR(1))/CPTIP(1)
      IF (TTIP.LT.0.) GO TO 220
      RHO(I,1) = RHOIP(1)*TTIP**EXPON
      IF (I.GE.ILE(1).AND.I.LE.ITE(1)) GO TO 160
```

```
C--RVA OUTSIDE OF THE BLADE
      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)
      CBETA = COS(BETA(I,1))
      RVA = RHO(I,1)*WHUB*CBETA*RCARB(1)
      GO TO 170
C--RVA INSIDE OF THE BLADE
  160 WLSRF = WHUB+DFDM(I,1)/2.
      WSQ = WLSRF**2
      TTIP = 1.-(WSQ+TWLMR(1))/CPTIP(1)
      IF (TTIP.LT.0.) TTIP=0.
      RHOL = RHOIP(1)*TTIP**EXPON
      WTSRF = WHUB-DFDM(I,1)/2.
      WSQ = WTSRF**2
      TTIP = 1.-(WSQ+TWLMR(1))/CPTIP(1)
      IF (TTIP.LT.0.) TTIP=0.
      RHOT = RHOIP(1)*TTIP**EXPON
      RHOWAV = (RHOL*WLSRF+4.*RHO(I,1)*WHUB+RHOT*WTSRF)/6.
      CBETA = COS(BETA(I,1))
      RVA = RHOWAV*CBETA*RCARB(1)
C
C--CONTINUE VELOCITY GRADIENT SOLUTION UP VERTICAL MESH LINE FROM HUB
C--TO SHROUD
C
  170 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)
C
C--CALCULATE RVAS AT POSITION J+1 ON VERTICAL MESH LINE
      IF (I.GE.ILE(J+1).AND.I.LE.ITE(J+1)) GO TO 180
C--RVAS OUTSIDE OF THE BLADE
      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
      RHO(I,J+1) = RHOIP(J+1)*TTIP**EXPON
      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)
      CBETA = COS(BETA(I,J+1))
      RVAS = RHO(I,J+1)*W(I,J+1)*CBETA*RCARB(J+1)
      GO TO 195
C--RVAS INSIDE OF THE BLADE
  190 WLSRF = W(I,J+1)+DFDM(I,J+1)/2.
      WSQ = WLSRF**2
      TTIP = 1.-(WSQ+TWLMR(J+1))/CPTIP(J+1)
```

```
      IF (TTIP.LT.0.) TTIP=0.
      RHOL = RHOIP(J+1)*TTIP**EXPON
      WTSRF = W(I,J+1)-DFDM(I,J+1)/2.
      WSQ = WTSRF**2
      TTIP = 1.-(WSQ+TWLMR(J+1))/CPTIP(J+1)
      IF (TTIP.LT.0.) TTIP=0.
      RHOT = RHOIP(J+1)*TTIP**EXPON
      RHOWAV = (RHOL*WLSRF+4.*RHO(I,J+1)*W(I,J+1)+RHOT*WTSRF)/6.
      CBETA = COS(BETA(I,J+1))
      RVAS = RHOWAV*CBETA*RCARB(J+1)
C
C--INCREMENT THE MASSFLOW
  195 UNEW(J+1) = (RVA+RVAS)*DELTA/2.+UNEW(J)
  200 RVA = RVAS
C
C--STORE MAX AND MIN VALUES FOR WHUB AND INTEGRATED MASSFLOW
C
      MAXFLO = AMAX1(UNEW(MHTP1),MAXFLO)
      MINFLO = AMIN1(UNEW(MHTP1),MINFLO)
      WMAX = AMAX1(WHUB,WMAX)
      WMIN = AMIN1(WHUB,WMIN)
C
C--CHECK CONTINUITY AND ESTIMATE NEW VALUE FOR W AT THE HUB
C
      IF(IND.GE.6.AND.ABS(MSFL-UNEW(MHTP1)).LE.MSFL*RTOLER) GO TO 250
      CALL CONTIN(WHUB,UNEW(MHTP1),IND,JZ,MSFL,DELMAX)
      IF(IND.LT.10) GO TO 150
C
C--END OF INNER ITERATION PROCEDURE
C
C--IND=10 INDICATES CHOKED FLOW
      IF(IND.EQ.10) GO TO 250
C--IND=11 INDICATES NO SOLUTION FOUND IN 100 ITERATIONS
      GO TO 230
C
C--CHANGE WHUB FOR RESTART.  VELOCITIES TOO SMALL.  SBETA.GT.1.0
C
  210 WHUB = WHUB+.45*DELMAX
      NADD = NADD+1
      IF(NCOUNT.LT.1000) GO TO 140
      GO TO 230
C
C--CHANGE WHUB FOR RESTART.  VELOCITIES TOO BIG.  TEMPERATURE NEGATIVE
C
  220 WHUB = WHUB-.45*DELMAX
      NSUB = NSUB+1
      IF(NCOUNT.LT.1000) GO TO 140
C
C--NO SOLUTION CAN BE FOUND.  PRINT MESH OUTPUT ONLY, AND
C--OMIT STREAMLINE AND STATION LINE OUTPUT
C
  230 IMESH = 1
      ISLINE = 0
      ISTATL = 0
      DO 240 J=1,MHTP1
  240 UOM(I,J) = UOM(I,J)/MSFL
      GO TO 275
C
C--SOLUTION OBTAINED -- CHECK ACCURACY OF UOM
```

```
C
  250 NREP = NREP+1
      DO 260 J=2,MHTP1
      UTEMP = UNEW(J)/MSFL
      IF (ABS(UTEMP-UOM(I,J)).GT.RTOLER) REPEAT = .TRUE.
  260 UOM(I,J) = UTEMP
C
C--UPDATE PLOSS, TIPT, RHOIP, LAMBDA, AND LAMBDO
C
      CALL LOSSTV(I)
      DO 265 J=2,MHTP1
      UTEMP = UOM(I,J)
      TIPT(J) = TIPF(UTEMP)
      RHOIP(J) = RHOIPF(UTEMP)*(1.-PLOSS(I,J))
      LAMBDA(J) = LAMDAF(UTEMP,I,J)
  265 IF (MBI.NE.0) LAMBDO(J) = RVTHTA(UTEMP,I,J)
C
C--SET WHUB, AND CHECK IF ANOTHER OUTER ITERATION IS NECESSARY
C
      WHUB = W(I,1)
      IF(REPEAT.AND.NCOUNT.LT.1000) GO TO 90
C
C--END OF OUTER ITERATION PROCEDURE
C
      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 (.NOT.REPEAT) GO TO 280
C
C--PRINT ERROR MESSAGES IF A SATISFACTORY SOLUTION CANNOT BE OBTAINED
C
  275 WRITE(NWRIT,1010) I
      IF (IND.EQ.11) WRITE(NWRIT,1050)
      IF (NCOUNT.GE.1000) WRITE(NWRIT,1060)
      WRITE(NWRIT,1070) MAXFLO,MINFLO,WMAX,WMIN
      NRES = NADD+NSUB+NREP
      IF (NRES.GT.0) WRITE(NWRIT,1080) NRES,NADD,NSUB,NREP
      WRITE(NWRIT,1090)
C
C--CHECK IF ALL VERTICAL MESH LINES HAVE BEEN DONE
C
  280 IF (INCR.GT.0) GO TO 60
      IF (IREVRS.EQ.1) I=MM+1
      IREVRS = 0
      IF (I.GT.ITEMIN) GO TO 60
      IF (I.GT.IMAX+1) GO TO 60
      IF (ICOUNT.LT.MHTP1) GO TO 60
C--END OF LOOP ON VERTICAL MESH LINES
  290 CONTINUE
C
C--FINISHED VELOCITY GRADIENT SOLUTION ON EACH VERTICAL MESH LINE
C--CHECK CHOKE LIMIT
C
      OMSFL = MSFL*FLOAT(NBL)
      CHFRMS = CHLIM/OMSFL
      IF (CHLIM.GT.(0.9999*OMSFL)) RETURN
      WRITE(NWRIT,1030) CHFRMS,OMSFL,CHLIM
      RETURN
  300 WRITE(NWRIT,1020)
```

```
      STOP
C
C--FORMAT STATEMENTS
C
 1000 FORMAT (1HL,10X,68HMSFL EXCEEDS CHOKING MASS FLOW FOR VERTICAL ORT
     1HOGONAL MESH LINE I =,I3/12X,19HCHOKING MASS FLOW =,G15.6)
 1010 FORMAT (1HL,10X,85HA VELOCITY GRADIENT SOLUTION CANNOT BE OBTAINED
     1 FOR VERTICAL ORTHOGONAL MESH LINE I =,I3/12X,56HANY SUBSEQUENT OU
     2TPUT FOR THAT MESH LINE MAY BE IN ERROR)
 1020 FORMAT (1HL,10X,60HTHE UPSTREAM INPUT WHIRL OR TANGENTIAL VELOCITY
     1 IS TOO LARGE)
 1030 FORMAT (1HL,10X,19HCHOKING MASSFLOW IS,F9.5,22H OF THE INPUT MASSF
     1LOW/12X,16HINPUT MASSFLOW =,G13.5/12X,26HMINIMUM CHOKING MASSFLOW
     2=,G13.5)
 1040 FORMAT (1H1//52X,25(1H*)/52X,25H*    FULL    MASSFLOW    */42X,
     145(1H*)/42X,1H*,12X,19HTRANSONIC  SOLUTION,12X,1H*/42X,45H*  BY VE
     2LOCITY GRADIENT APPROXIMATE METHOD  */42X,45(1H*)/////)
 1050 FORMAT (10X,51HCONTIN COULD NOT FIND A SOLUTION IN 100 ITERATIONS.
     1)
 1060 FORMAT (10X,81HITERATION PROCEDURE HAD TO BE RESTARTED TO AVOID NE
     1GATIVE TEMPERATURE OR VELOCITY/12X,87HMAGNITUDE LESS THAN TANGENTI
     2AL VELOCITY, OR AFTER ADJUSTMENT OF STAGNATION TEMPERATURE,/12X,29
     3HSTAGNATION DENSITY, OR WHIRL./10X,84HRESTART OF ITERATION PROCEDU
     4RE (LOOP TO STATEMENT 90) WAS ABORTED AFTER 1000 OR MORE/12X,41HTO
     5TAL ITERATIONS (LOOP TO STATEMENT 150).)
 1070 FORMAT (1HL,10X,63HTHE MAXIMUM MASSFLOW FOR WHICH A SOLUTION COULD
     1 BE OBTAINED WAS,G16.7/10X,63HTHE MINIMUM MASSFLOW FOR WHICH A SOL
     2UTION COULD BE OBTAINED WAS,G16.7/10X,76HTHE MAXIMUM VALUE OF W AT
     3 THE HUB FOR WHICH A SOLUTION COULD BE OBTAINED WAS,G16.7/10X,76HT
     4HE MINIMUM VALUE OF W AT THE HUB FOR WHICH A SOLUTION COULD BE OBT
     5AINED WAS,G16.7)
 1080 FORMAT (1HL,10X,37HTHE ITERATION PROCEDURE WAS RESTARTED,I5,6H TIM
     1ES/12X,18HWHUB WAS INCREASED,I4,6H TIMES/12X,18HWHUB WAS DECREASED
     2,I4,6H TIMES/12X,53HBOUNDARY VALUES (TIPBDY, RHOIP, LAMBDA) WERE A
     3DJUSTED,I4,6H TIMES)
 1090 FORMAT (/10X,120(1H*))
      END
```

```
      SUBROUTINE LINDV(JARG,LINC,ICOUNT)
C
C--LINDV AND TINDV CORRECT THE BETA FLOW ANGLES INTO THE
C--LEADING AND TRAILING EDGES RESPECTIVELY
C
      COMMON NREAD,NWRIT,ITER,IEND,NWRT1,NWRT2,NWRT3,NWRT4,NWRT5,NWRT6
      COMMON/INPUTT/GAM,AR,MSFL,OMEGA,REDFAC,VELTOL,FNEW,DNEW,MBI,MBO,
     1   MM,MHT,NBL,NHUB,NTIP,NIN,NOUT,NBLPL,NPPP,NOSTAT,NSL,NLOSS,
     2   LSFR,LTPL,LAMVT,LROT,LBLAD,LETEAN,ANGROT,IMESH,ISLINE,
     3   ISTATL,IPLOT,ISUPER,ITSON,IDEBUG,ZCMIN,ZOMBI,ZOMBO,ZOMOUT,
     4   ROMIN,ROMBI,ROMBO,ROMOUT,ZHIN,ZTIN,ZHOUT,ZTOUT,RHIN,RTIN,RHOUT,
     5   RTOUT,TITLEI(20),ZHUB(50),RHUB(50),ZTIP(50),RTIP(50),SFIN(50),
     6   RADIN(50),TIP(50),PRIP(50),LAMIN(50),VTHIN(50),SFOUT(50),
     7   RADOUT(50),PROP(50),LOSOUT(50),LAMOUT(50),VTHOUT(50),
     8   BETALE(50),BETATE(50),ZHST(50),ZTST(50),RHST(50),RTST(50),
     9   FLFR(50),PERCRD(50),PERLOS(50),ZBL(50,50),RBL(50,50),
     1   THBL(50,50),TNBL(50,50),TTBL(50,50),TH1BL(50,50),TH2BL(50,50)
```

```
      COMMON/CALCON/MMM1,MHTP1,CP,EXPON,TGROG,PITCH,RLEH,RLET,RTEH,RTET,
     1   ZLE(50),RLE(50),ZTE(50),RTE(50),ZLEOM(101),RLEOM(101),
     2   SLEOM(101),THLEOM(101),ZTEOM(101),RTEOM(101),STEOM(101),
     3   THTEOM(101),ILE(101),ITE(101),ZOM(100,101),ROM(100,101),
     4   SOM(100,101),TOM(100,101),BTH(100,101),DTHDS(100,101),
     5   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),FT(100,101),DFDM(100,101),
     6   XIOM(100,101),ZETOM(100,101),DLDU(100,101)
      COMMON/ROTATN/ZHROT(50),RHROT(50),ZTROT(50),RTROT(50),
     1   ZLEOMR(101),RLEOMR(101),ZTEOMR(101),RTEOMR(101),
     2   ZBLROT(50,50),RBLROT(50,50),ZOMROT(100,101),ROMROT(100,101)
      COMMON/INDCOM/NBLPC,NPPC,ZPC(51,51),RPC(51,51),TTPC(51,51),
     1   THPC(51,51),DTHDZ(51,51),DTHDR(51,51),BTHLE(101),BTHTE(101),
     2   BTBFLE(101),BTBFTE(101)
      DIMENSION BLINC(101),UBINC(101),BTBLLE(101),BLDEV(101),UBDEV(101),
     1   BTBLTE(101)
      REAL LAMDAF
C
C--LINDV CORRECTS THE BETA FLOW ANGLES INTO THE LEADING EDGE
C
      II = 1
      JJ = 1
      J = JARG
      ICOUNT = ICOUNT+1
      DEGRAD = 180./3.1415927
C
C--CALCULATE BLADE MEAN CAMBER ANGLE AT LEADING EDGE
      I = ILE(J)-1
      ALPHLE = ALPHA(I,J)+(SLEOM(J)-SOM(I,J))*(ALPHA(I+1,J)-ALPHA(I,J))/ -
     1(SOM(I+1,J)-SOM(I,J))
      CALL LININT(ZPC,RPC,DTHDZ,NPPC,NBLPC,51,51,ZLEOMR(J),RLEOMR(J),     -
     1DTDZLE,II,JJ)
      CALL LININT(ZPC,RPC,DTHDR,NPPC,NBLPC,51,51,ZLEOMR(J),RLEOMR(J),     -
     1DTDRLE,II,JJ)
      TANBBL = RLEOM(J)*(DTDRLE*SIN(ALPHLE)+DTDZLE*COS(ALPHLE))
      BTBLLE(J) = ATAN(TANBBL)*DEGRAD
C
C--CALCULATE BLADE FLOW ANGLE AT LEADING EDGE, CORRECTED FOR BLOCKAGE
      EXFRAC = (SLEOM(J)-SOM(I,J))/(SOM(I,J)-SOM(I-1,J))
      BETAFS = BETA(I,J)+EXFRAC*(BETA(I,J)-BETA(I-1,J))
      RHOFS = RHO(I,J)+EXFRAC*(RHO(I,J)-RHO(I-1,J))
      WFS = W(I,J)+EXFRAC*(W(I,J)-W(I-1,J))
      ULE = UOM(I,J)+EXFRAC*(UOM(I,J)-UOM(I-1,J))
      TWLMR = 2.*OMEGA*LAMDAF(ULE,ILE(J),J)-(OMEGA*RLEOM(J))**2
      TIPRIM = TIPF(ULE)
      CPTIP = 2.*CP*TIPRIM
      RHOIP = RHOIPF(ULE)
      CONST1 = TAN(BETAFS)/RHOFS*BTHLE(J)/PITCH
      CONST2 = (RHOFS*PITCH*WFS/BTHLE(J))**2/(1.+(TAN(BETAFS))**2)
      RHOBFN = RHOFS
      TPP = TIPRIM-TWLMR/2./CP
      WTHETA = WFS*SIN(BETAFS)
      WMSON = SQRT(2.*GAM*AR*TPP/(GAM+1.)-(GAM-1.)/(GAM+1.)*WTHETA**2)
      IF(WFS*COS(BETAFS).GT.WMSON) GO TO 14
```

```
   10 RHOBF = RHOBFN
      TANBBF = CONST1*RHOBF
      WSQBF = CONST2/RHOBF**2*(1.+TANBBF**2)
      TBFTIP = 1.-(WSQBF+TWLMR)/CPTIP
      IF(TBFTIP.LT.0.) GO TO 16
      RHOBFN = RHOIP*TBFTIP**EXPON
      IF (ABS(RHOBFN-RHOBF)/RHOBFN.GT..0001) GO TO 10
      GO TO 18
   14 RHOBF = RHOBFN
      TBFTIP = (RHOBF/RHOIP)**(1./EXPON)
      WSQBF = (1.-TBFTIP)*CPTIP-TWLMR
      RHSQBF = CONST2/(WSQBF-CONST2*CONST1**2)
      IF(RHSQBF.LT.0.) GO TO 16
      RHOBFN = SQRT(RHSQBF)
      IF(ABS(RHOBFN-RHOBF)/RHOBFN.GT..0001) GO TO 14
      WRITE(NWRIT,1060) J
      GO TO 18
   16 RHOBFN = RHOFS
      WRITE(NWRIT,1070) J
   18 TANBBF = CONST1*RHOBFN
      BETABF = ATAN(TANBBF)
C
C--CALCULATE DISTANCE FOR BETA CORRECTION
      BLDCRD = (RLEOM(J)+RTEOM(J))/2.*(THTEOM(J)-THLEOM(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))
C
C--CORRECT BETA FOR INCIDENCE NEAR THE LEADING EDGE,
C--USING LINEAR CORRECTION FOR ANGLE
      I = ILE(J)
   20 SDIST = SLEOM(J)+DISTLE-SOM(I,J)
      IF (SDIST.LE.0.) GO TO 30
      BETA(I,J) = BETA(I,J)+(BETABF-BTBFLE(J))*SDIST/DISTLE
      I = I+1
      GO TO 20
C
C--CALCULATE INCIDENCE ANGLES
   30 BLINC(J) = BETABF*DEGRAD-BTBLLE(J)
      UBINC(J) = BETAFS*DEGRAD-BTBLLE(J)
      IF (ICOUNT.EQ.MHTP1) LINC=1
      RETURN
C
C--TINDV CORRECTS THE BETA FLOW ANGLES INTO THE TRAILING EDGE
C
      ENTRY TINDV(IARG,JARG,ICOUNT)
      J = JARG
      ICOUNT = ICOUNT+1
C
C--CALCULATE BLADE MEAN CAMBER ANGLE AT TRAILING EDGE
      I = ITE(J)+1
      ALPHTE = ALPHA(I,J)+(STEOM(J)-SOM(I,J))*(ALPHA(I,J)-ALPHA(I-1,J))/ -
     1(SOM(I,J)-SOM(I-1,J))
      CALL LININT(ZPC,RPC,DTHDZ,NPPC,NBLPC,51,51,ZTEOMR(J),RTEOMR(J),   -
     1DTDZTE,II,JJ)
      CALL LININT(ZPC,RPC,DTHDR,NPPC,NBLPC,51,51,ZTEOMR(J),RTEOMR(J),   -
     1DTDRTE,II,JJ)
      TANBBL = RTEOM(J)*(DTDRTE*SIN(ALPHTE)+DTDZTE*COS(ALPHTE))
```

```
      BTBLTE(J)  =  ATAN(TANBBL)*DEGRAD
C
C--CALCULATE BLADE FLOW ANGLE AT TRAILING EDGE, CORRECTED FOR BLOCKAGE
      EXFRAC =  (SOM(I,J)-STEOM(J))/(SOM(I+1,J)-SOM(I,J))
      BETAFS = BETA(I,J)+EXFRAC*(BETA(I,J)-BETA(I+1,J))
      RHOFS = RHO(I,J)+EXFRAC*(RHO(I,J)-RHO(I+1,J))
      WFS = W(I,J)+EXFRAC*(W(I,J)-W(I+1,J))
      UTE = UOM(I,J)+EXFRAC*(UOM(I,J)-UOM(I+1,J))
      PLOSTE = PLOSS(I,J)+EXFRAC*(PLOSS(I,J)-PLOSS(I+1,J))
      TWLMR = 2.*OMEGA*LAMDAF(UTE,ITE(J),J)-(OMEGA*RTEOM(J))**2
      TIPRIM = TIPF(UTE)
      CPTIP = 2.*CP*TIPRIM
      RHOIP = RHOIPF(UTE)*(1.-PLOSTE)
      CONST1 = TAN(BETAFS)/RHOFS*BTHTE(J)/PITCH
      CONST2 =  (RHOFS*PITCH*WFS/BTHTE(J))**2/(1.+(TAN(BETAFS))**2)
      RHOBFN = RHOFS
      TPP = TIPRIM-TWLMR/2./CP
      WTHETA = WFS*SIN(BETAFS)
      WMSON = SQRT(2.*GAM*AR*TPP/(GAM+1.)-(GAM-1.)/(GAM+1.)*WTHETA**2)
      IF(WFS*COS(BETAFS).GT.WMSON) GO TO 44
   40 RHOBF = RHOBFN
      TANBBF = CONST1*RHOBF
      WSQBF = CONST2/RHOBF**2*(1.+TANBBF**2)
      TBFTIP = 1.-(WSQBF+TWLMR)/CPTIP
      IF(TBFTIP.LT.0.) GO  TO 46
      RHOBFN = RHOIP*TBFTIP**EXPON
      IF (ABS(RHOBFN-RHOBF)/RHOBFN.GT..0001) GO TO 40
      GO TO 48
   44 RHOBF = RHOBFN
      TBFTIP =  (RHOBF/RHOIP)**(1./EXPON)
      WSQBF =  (1.-TBFTIP)*CPTIP-TWLMR
      RHSQBF = CONST2/(WSQBF-CONST2*CONST1**2)
      IF(RHSQBF.LT.0.) GO TO 46
      RHOBFN = SQRT(RHSQBF)
      IF(ABS(RHOBFN-RHOBF)/RHOBFN.GT..0001) GO TO 44
      WRITE(NWRIT,1080) J
      GO TO 48
   46 RHOBFN = RHOFS
      WRITE(NWRIT,1090) J
   48 TANBBF = CONST1*RHOBFN
      BETABF = ATAN(TANBBF)
C
C--CALCULATE DISTANCE FOR BETA CORRECTION
      BLDCRD =  (RLEOM(J)+RTEOM(J))/2.*(THTEOM(J)-THLEOM(J))
      BLDCRD = SQRT(BLDCRD**2+(STEOM(J)-SLEOM(J))**2)
      SLIDTE = BLDCRD/PITCH/RTEOM(J)
      DISTTE = AMIN1(.5,AMAX1(1./6.,(11.-4.*SLIDTE)/18.))*(STEOM(J)-
     1SLEOM(J))
C
C--CORRECT BETA FOR DEVIATION NEAR THE TRAILING EDGE,
C--USING LINEAR CORRECTION FOR ANGLE
      I = ITE(J)
   50 SDIST = SOM(I,J)-STEOM(J)+DISTTE
      IF (SDIST.LE.0.) GO TO 60
      BETA(I,J)  = BETA(I,J)+(BETABF-BTBFTE(J))*SDIST/DISTTE
      I = I-1
      GO TO 50
C
C--CALCULATE DEVIATION ANGLES
```

```
   60 BLDEV(J) = BETABF*DEGRAD-BTBLTE(J)
      UBDEV(J) = BETAFS*DEGRAD-BTBLTE(J)
      IARG = I+1
      RETURN
C
C--PINDV PRINTS THE INCIDENCE AND DEVIATION ANGLES
C
      ENTRY PINDV
      IF ((LAMDAF(.5,ILE(1),1)-RVTHTA(.5,ILE(1),1)).GT.0.) GO TO 80
      DO 70 J=1,MHTP1
      BLINC(J) = -BLINC(J)
      UBINC(J) = -UBINC(J)
      BLDEV(J) = -BLDEV(J)
   70 UBDEV(J) = -UBDEV(J)
   80 WRITE(NWRT6,1000)
      IF (ISUPER.LE.1) WRITE(NWRT6,1010)
      IF (ISUPER.EQ.2) WRITE(NWRT6,1020)
      WRITE(NWRT6,1030)
      WRITE(NWRT6,1040) (J,BLINC(J),UBINC(J),BTBLLE(J),BLDEV(J),
     1UBDEV(J),BTBLTE(J),J=1,MHTP1)
      WRITE(NWRT6,1050)
      RETURN
C
C--FORMAT STATEMENTS
C
 1000 FORMAT (1H1////44X,40(1H*)/44X,40H***  INCIDENCE AND DEVIATION ANG
     1LES   ***/  49X,30(1H*)//)
 1010 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*))
 1020 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*))
 1030 FORMAT (//24X,10H*  MESH  *,8X,9HINCIDENCE,7X,11HBLADE ANGLE,2H *,
     18X,9HDEVIATION,7X,11HBLADE ANGLE,2H */24X,10H*  LINE  *,3X,
     27HBLOCKED,3X,9HUNBLOCKED,4X,7HAT L.E.,3X,1H*,3X,7HBLOCKED,3X,
     39HUNBLOCKED,4X,7HAT T.E.,3X,1H*)
 1040 FORMAT ((24X,1H*,2X,I3,3X,2(1H*,3(F9.2,2X),3X),1H*))
 1050 FORMAT (1H1)
 1060 FORMAT (45HLSUPERSONIC CORRECTION - LEADING EDGE FOR J =,I3)
 1070 FORMAT (45HLNO DENSITY CORRECTION - LEADING EDGE FOR J =,I3)
 1080 FORMAT (46HLSUPERSONIC CORRECTION - TRAILING EDGE FOR J =,I3)
 1090 FORMAT (46HLNO DENSITY CORRECTION - TRAILING EDGE FOR J =,I3)
      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 NREAD,NWRIT,ITER,IEND,NWRT1,NWRT2,NWRT3,NWRT4,NWRT5,NWRT6
      COMMON/INPUTT/GAM,AR,MSFL,OMEGA,REDFAC,VELTOL,FNEW,DNEW,MBI,MBO,
     1   MM,MHT,NBL,NHUB,NTIP,NIN,NOUT,NBLPL,NPPP,NOSTAT,NSL,NLOSS,
     2   LSFR,LTPL,LAMVT,LROT,LBLAD,LETEAN,ANGROT,IMESH,ISLINE,
```

```
     3    ISTATL,IPLOT,ISUPER,ITSON,IDEBUG,ZOMIN,ZOMBI,ZOMBO,ZOMOUT,
     4    ROMIN,ROMBI,ROMBO,ROMOUT,ZHIN,ZTIN,ZHOUT,ZTOUT,RHIN,RTIN,RHOUT,
     5    RTOUT,TITLEI(20),ZHUB(50),RHUB(50),ZTIP(50),RTIP(50),SFIN(50),
     6    RADIN(50),TIP(50),PRIP(50),LAMIN(50),VTHIN(50),SFOUT(50),
     7    RADOUT(50),PRCP(50),LOSOUT(50),LAMOUT(50),VTHOUT(50),
     8    BETALE(50),BETATE(50),ZHST(50),ZTST(50),RHST(50),RTST(50),
     9    FLFR(50),PERCRD(50),PERLOS(50),ZBL(50,50),RBL(50,50),
     1    THBL(50,50),TNBL(50,50),TTBL(50,50),TH1BL(50,50),TH2BL(50,50)
      COMMON/CALCON/MMM1,MHTP1,CP,EXPON,TGROG,PITCH,RLEH,RLET,RTEH,RTET,
     1    ZLE(50),RLE(50),ZTE(50),RTE(50),ZLEOM(101),RLEOM(101),
     2    SLEOM(101),THLEOM(101),ZTEOM(101),RTEOM(101),STEOM(101),
     3    THTEOM(101),ILE(101),ITE(101),ZOM(100,101),ROM(100,101),
     4    SOM(100,101),TOM(100,101),BTH(100,101),DTHDS(100,101),
     5    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),FT(100,101),DFDM(100,101),
     6    XIOM(100,101),ZETOM(100,101),DLDU(100,101)
      COMMON/ROTATN/ZHROT(50),RHROT(50),ZTROT(50),RTROT(50),
     1    ZLEOMR(101),RLEOMR(101),ZTEOMR(101),RTEOMR(101),
     2    ZBLROT(50,50),RBLROT(50,50),ZOMROT(100,101),ROMROT(100,101)
      DIMENSION SLOPE(50),EM(50),AAA(50),BBB(50),RILOM(101),UILOM(101)
      REAL LAMDAF,LAMIN
      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
      IF (LSFR.EQ.0.AND.LAMVT.EQ.0) GO TO 100
      II = MBI
      JJ = 1
```

```
      CAN = COS(ANGROT)
      SAN = SIN(ANGROT)
      ZHINRO = ZHIN*CAN+RHIN*SAN
      RHINRO = RHIN*CAN-ZHIN*SAN
      ZTINRO = ZTIN*CAN+RTIN*SAN
      RTINRO = RTIN*CAN-ZTIN*SAN
      DO 80 KK=1,MHTP1
      DIST = FLOAT(KK-1)/FLOAT(MHT)
      RILOM(KK) = RHIN+DIST*(RTIN-RHIN)
      ZIROT = ZHINRO+DIST*(ZTINRO-ZHINRO)
      RIROT = RHINRO+DIST*(RTINRO-RHINRO)
   80 CALL LININT(ZOMROT,ROMROT,UOM,MM,MHTP1,100,101,ZIROT,RIROT,
     1UILOM(KK),IT,JJ)
      IF (LSFR.EQ.0) CALL SPLINT(UILOM,RILOM,MHTP1,SFIN,NIN,RADIN,AAA,
     1BBB)
      IF (LSFR.EQ.1) CALL SPLINT(RILOM,UILOM,MHTP1,RADIN,NIN,SFIN,AAA,
     1BBB)
      IF (LSFR.EQ.1.OR.LAMVT.EQ.0) 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 RVTHTA(SF,I,J)
C
C--RVTHTA CALCULATES  R * V-THETA  AS A FUNCTION OF STREAM FUNCTION
C--DOWNSTREAM OF THE BLADE
C
      COMMON NREAD,NWRIT,ITER,IEND,NWRT1,NWRT2,NWRT3,NWRT4,NWRT5,NWRT6
      COMMON/INPUTT/GAM,AR,MSFL,OMEGA,REDFAC,VELTOL,FNEW,DNEW,MBI,MBO,
     1   MM,MHT,NBL,NHUB,NTIP,NIN,NOUT,NBLPL,NPPP,NOSTAT,NSL,NLOSS,
     2   LSFR,LTPL,LAMVT,LROT,LBLAD,LETEAN,ANGROT,IMESH,ISLINE,
     3   ISTATL,IPLOT,ISUPER,ITSON,IDEBUG,ZOMIN,ZOMBI,ZOMBO,ZOMOUT,
     4   ROMIN,ROMBI,ROMBO,ROMOUT,ZHIN,ZTIN,ZHOUT,ZTOUT,RHIN,RTIN,RHOUT,
     5   RTOUT,TITLEI(20),ZHUB(50),RHUB(50),ZTIP(50),RTIP(50),SFIN(50),
     6   RADIN(50),TIP(50),PRTIP(50),LAMIN(50),VTHIN(50),SFOUT(50),
     7   RADOUT(50),PROP(50),LOSOUT(50),LAMOUT(50),VTHOUT(50),
     8   BETALE(50),BETATE(50),ZHST(50),ZTST(50),RHST(50),RTST(50),
     9   FLFR(50),PERCRD(50),PERLOS(50),ZBL(50,50),RBL(50,50),
     1   THBL(50,50),TNBL(50,50),TTBL(50,50),TH1BL(50,50),TH2BL(50,50)
      COMMON/CALCON/MMM1,MHTP1,CP,EXPON,TGROG,PITCH,RLEH,RLET,RTEH,RTET,
     1   ZLE(50),RLE(50),ZTE(50),RTE(50),ZLEOM(101),RLEOM(101),
     2   SLEOM(101),THLEOM(101),ZTEOM(101),RTEOM(101),STEOM(101),
     3   THTEOM(101),ILE(101),ITE(101),ZOM(100,101),ROM(100,101),
     4   SOM(100,101),TOM(100,101),BTH(100,101),DTHDS(100,101),
     5   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),FT(100,101),DFDM(100,101),
     6   XIOM(100,101),ZETOM(100,101),DLDU(100,101)
      COMMON/ROTATN/ZHROT(50),RHROT(50),ZTROT(50),RTROT(50),
```

```
     1    ZLEOMR(101),RLEOMR(101),ZTEOMR(101),RTEOMR(101),
     2    ZBLROT(50,50),RBLROT(50,50),ZOMROT(100,101),ROMROT(100,101)
      DIMENSION SLOPE(50),EM(50),AAA(50),BBB(50),ROLOM(101),UOLOM(101)
      REAL LAMOUT
      KK = 2
      IF(ABS(SF-SFOUT(1)).GT.TOLER) GO TO 10
      RVTHTA = LAMOUT(1)
      IF (I.GT.ITE(J)) DLDU(I,J)=SLOPE(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)=SLOPE(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-SFOUT(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
      IF (LSFR.EQ.0.AND.LAMVT.EQ.0) GO TO 100
      II = MBO
      JJ = 1
      CAN = COS(ANGROT)
      SAN = SIN(ANGROT)
      ZHORO = ZHOUT*CAN+RHOUT*SAN
      RHORO = RHOUT*CAN-ZHOUT*SAN
      ZTORO = ZTOUT*CAN+RTOUT*SAN
      RTORO = RTOUT*CAN-ZTOUT*SAN
      DO 80 KK=1,MHTP1
      DIST = FLOAT(KK-1)/FLOAT(MHT)
      ROLOM(KK) = RHOUT+DIST*(RTOUT-RHOUT)
      ZOROT = ZHORO+DIST*(ZTORO-ZHORO)
      ROROT = RHORO+DIST*(RTORO-RHORO)
   80 CALL LININT(ZOMROT,ROMROT,UOM,MM,MHTP1,100,101,ZOROT,ROROT,
     1UOLOM(KK),II,JJ)
      IF (LSFR.EQ.0) CALL SPLINT(UOLOM,ROLOM,MHTP1,SFOUT,NOUT,RADOUT,
     1AAA,BBB)
      IF (LSFR.EQ.1) CALL SPLINT(ROLOM,UOLOM,MHTP1,RADOUT,NOUT,SFOUT,
     1AAA,BBB)
      IF (LSFR.EQ.1.OR.LAMVT.EQ.0) GO TO 100
      DO 90 KK=1,NOUT
   90 LAMOUT(KK)= RADOUT(KK)*VTHOUT(KK)
  100 CALL SPLINE(SFOUT,LAMOUT,NOUT,SLOPE,EM)
      TOLER = ABS(SFOUT(NOUT)-SFOUT(1))/FLOAT(NOUT)*1.E-6
      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,DNEW,MBI,MBO,
     1   MM,MHT,NBL,NHUB,NTIP,NIN,NOUT,NBLPL,NPPP,NOSTAT,NSL,NLOSS,
     2   LSFR,LTPL,LAMVT,LROT,LBLAD,LETEAN,ANGROT,IMESH,ISLINE,
     3   ISTATI,IPLOT,ISUPER,ITSON,IDEBUG,ZOMIN,ZOMBI,ZOMBO,ZOMOUT,
     4   ROMIN,ROMBI,ROMBO,ROMOUT,ZHIN,ZTIN,ZHOUT,ZTOUT,RHIN,RTIN,RHOUT,
     5   RTOUT,TITLEI(20),ZHUB(50),RHUB(50),ZTIP(50),RTIP(50),SFIN(50),
     6   RADIN(50),TIP(50),PRIP(50),LAMIN(50),VTHIN(50),SFOUT(50),
     7   RADOUT(50),PROP(50),LOSOUT(50),LAMOUT(50),VTHOUT(50),
     8   BETALE(50),BETATE(50),ZHST(50),ZTST(50),RHST(50),RTST(50),
     9   FLFR(50),PERCRD(50),PERLOS(50),ZBL(50,50),RBL(50,50),
     1   THBL(50,50),TNBL(50,50),TTBL(50,50),TH1BL(50,50),TH2BL(50,50)
      COMMON/CALCON/MMM1,MHTP1,CP,EXPON,TGROG,PITCH,RLEH,RLET,RTEH,RTET,
     1   ZLE(50),RLE(50),ZTE(50),RTE(50),ZLEOM(101),RLEOM(101),
     2   SLEOM(101),THLEOM(101),ZTEOM(101),RTEOM(101),STEOM(101),
     3   THTEOM(101),ILE(101),ITE(101),ZOM(100,101),ROM(100,101),
     4   SOM(100,101),TOM(100,101),BTH(100,101),DTHDS(100,101),
     5   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,MSFI,OMEGA,REDFAC,VELTOL,FNEW,DNEW,MBI,MBO,
```

```
     1    MM,MHT,NBL,NHUB,NTIP,NIN,NOUT,NBLPL,NPPP,NOSTAT,NSL,NLOSS,
     2    LSFR,LTPL,LAMVT,LROT,LBLAD,LETEAN,ANGROT,IMESH,ISLINE,
     3    ISTATL,IPLOT,ISUPER,ITSON,IDEBUG,ZOMIN,ZOMBI,ZOMBO,ZOMOUT,
     4    ROMIN,ROMBI,ROMBO,ROMOUT,ZHIN,ZTIN,ZHOUT,ZTOUT,RHIN,RTIN,RHOUT,
     5    RTOUT,TITLEI(20),ZHUB(50),RHUB(50),ZTIP(50),RTIP(50),SFIN(50),
     6    RADIN(50),TIP(50),PRIP(50),LAMIN(50),VTHIN(50),SFOUT(50),
     7    RADOUT(50),PROP(50),LOSOUT(50),LAMOUT(50),VTHOUT(50),
     8    BETALE(50),BETATE(50),ZHST(50),ZTST(50),RHST(50),RTST(50),
     9    FLFR(50),PERCRD(50),PERLOS(50),ZBL(50,50),RBL(50,50),
     1    THBL(50,50),TNBL(50,50),TTBL(50,50),TH1BL(50,50),TH2BL(50,50)
      COMMON/CALCON/MMM1,MHTP1,CP,EXPON,TGROG,PITCH,RLEH,RLET,RTEH,RTET,
     1    ZLE(50),RLE(50),ZTE(50),RTE(50),ZLEOM(101),RLEOM(101),
     2    SLEOM(101),THLEOM(101),ZTEOM(101),RTEOM(101),STEOM(101),
     3    THTEOM(101),ILE(101),ITE(101),ZOM(100,101),ROM(100,101),
     4    SOM(100,101),TOM(100,101),BTH(100,101),DTHDS(100,101),
     5    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



      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 GRADIENT 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),AP )
      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 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,TEMP)
C--COMPUTE INTERSECTION POINT AND SLOPE ON CURVE 2
      Y2 = Y1
      CALL SPLINT(YCURV2,XCURV2,N2,Y2,1,X2,S2,TEMP)
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 LININT(X,Y,Z,NX,NY,NDIMX,NDIMY,X0,Y0,Z0,I,J)
C
C--LININT LOCATES THE POINT (X0,Y0) IN A 2-D  MESH WITH
C--COORDINATES STORED IN THE X AND Y ARRAYS.  THEN THE VALUE OF Z0 AT
C--(X0,Y0) IS INTERPOLATED FROM THE Z ARRAY VALUES CORRESPONDING
C--TO THE X AND Y ARRAYS
C
      COMMON NREAD,NWRIT
      DIMENSION X(NDIMX,NDIMY),Y(NDIMX,NDIMY),Z(NDIMX,NDIMY)
      DIMENSION EXTRAP(2)
      INTEGER ABOVE,RIGHT
C--FIND I,J SUCH THAT (X0,Y0) 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
      ICOUNT = 0
      ICNTMX = 2*(NX+NY)
```

```
   10 ABOVE = -1
      RIGHT = -1
      IF(Y0.GE.Y(I,J)+(X0-X(I,J))/(X(I+1,J)-X(I,J))*(Y(I+1,J)-Y(I,J)))
     1   ABOVE = ABOVE+1
      IF(Y0.GT.Y(I,J+1)+(X0-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(X0.GE.X(I,J)+(Y0-Y(I,J))/(Y(I,J+1)-Y(I,J))*(X(I,J+1)-X(I,J)))
     1   RIGHT = RIGHT+1
      IF(X0.GT.X(I+1,J)+(Y0-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
      ICOUNT = ICOUNT+1
      IF(ICOUNT.GT.ICNTMX) GO TO 110
      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(NWRIT,1000) I,J
      RETURN
 1000 FORMAT(38H1LININT CANNOT FIND INTERPOLATED VALUE/4H I =,I6,4H J =,
     1I6)
      END
```

```
      SUBROUTINE ROTATE(ANGROT,X,Y,NX,NY,NDIMX,NDIMY,XROT,YROT)
      DIMENSION X(NDIMX,NDIMY),Y(NDIMX,NDIMY),XROT(NDIMX,NDIMY),
     1    YROT(NDIMX,NDIMY)
      CAN = COS(ANGROT)
      SAN = SIN(ANGROT)
      DO 10 J=1,NY
      DO 10 I=1,NX
      TEMP = X(I,J)*CAN+Y(I,J)*SAN
      YROT(I,J) = Y(I,J)*CAN-X(I,J)*SAN
   10 XROT(I,J) = TEMP
```

```
      RETURN
      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 END POINTS ARE
C--SDR1 AND SDRN TIMES SECOND DERIVATIVES AT ADJACENT POINTS
C
      COMMON NREAD,NWRIT
      DIMENSION X(N),Y(N),SLOPE(N),EM(N)
      DIMENSION G(101),SB(101)
      IERR = 0
      SDR1 = .5
      SDRN = .5
      C = X(2)-X(1)
      IF (C.EQ.0.) GO TO 50
      SB(1) = -SDR1
      G(1) = 0.
      NO = N-1
      IF (NO.LE.0) GO TO 60
      IF (NO.EQ.1) GO TO 20
      DO 10 I=2,NO
      A = C
      C = X(I+1)-X(I)
      IF (A*C.EQ.0.) GO TO 50
      IF (A*C.LT.0.) IERR = 1
      W = 2.*(A+C)-A*SB(I-1)
      SB(I) = C/W
      F = (Y(I+1)-Y(I))/C-(Y(I)-Y(I-1))/A
   10 G(I) = (6.*F-A*G(I-1))/W
   20 EM(N) = SDRN*G(N-1)/(1.+SDRN*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 (IERR.EQ.0) RETURN
   50 WRITE(NWRIT,1000)
      WRITE(NWRIT,1020) N,(X(I),Y(I),I=1,N)
      IF (IERR.EQ.0) STOP
      WRITE(NWRIT,1030)
      RETURN
   60 WRITE(NWRIT,1010)
      WRITE(NWRIT,1020) N,(X(I),Y(I),I=1,N)
      STOP
 1000 FORMAT (1H1,10X,44HSPLINE ERROR -- ONE OF THREE POSSIBLE CAUSES/
     117X,51H1.  ADJACENT X POINTS ARE DUPLICATES OF EACH OTHER./
     217X,38H2.  SOME X POINTS ARE OUT OF SEQUENCE./
     317X,32H3.  SOME X POINTS ARE UNDEFINED.)
 1010 FORMAT (1H1,10X,62HSPLINE ERROR -- NUMBER OF SPLINE POINTS GIVEN I
     1S LESS THAN TWO)
 1020 FORMAT (//17X,18HNUMBER OF POINTS =,I4//17X,8HX  ARRAY,6X,8HY  ARR
```

```
     1AY/(17X,2G13.5))
1030 FORMAT (1H1)
     END
```

```
      SUBROUTINE SPLINT (X,Y,N,Z,MAX,YINT,DYDX,D2YDX2)
C
C--SPLINT CALCULATES INTERPOLATED POINTS AND DERIVATIVES
C--FOR A SPLINE CURVE
C--END CONDITION - SECOND DERIVATIVES AT END POINTS ARE
C--SDR1 AND SDRN TIMES SECOND DERIVATIVES AT ADJACENT POINTS
C
      COMMON NREAD,NWRIT
      DIMENSION X(N),Y(N),Z(MAX),YINT(MAX),DYDX(MAX),D2YDX2(MAX)
      DIMENSION G(101),SB(101),EM(101)
      IERR = 0
      SDR1 = .5
      SDRN = .5
      TOLER= ABS(X(N)-X(1))/FLOAT(N)*1.E-5
      C = X(2)-X(1)
      IF (C.EQ.0.) GO TO 130
      SB(1) = -SDR1
      G(1) = 0.
      NO = N-1
      IF (NO.LE.0) GO TO 140
      IF (NO.EQ.1) GO TO 20
      DO 10 I=2,NO
      A = C
      C = X(I+1)-X(I)
      IF (A*C.EQ.0.) GO TO 130
      IF (A*C.LT.0.) IERR = 1
      W = 2.*(A+C)-A*SB(I-1)
      SB(I) = C/W
      F = (Y(I+1)-Y(I))/C-(Y(I)-Y(I-1))/A
   10 G(I) = (6.*F-A*G(I-1))/W
   20 EM(N) = SDRN*G(N-1)/(1.+SDRN*SB(N-1))
      DO 30 I=2,N
      K = N+1-I
   30 EM(K) = G(K)-SB(K)*EM(K+1)
      IF (MAX.LE.0) RETURN
C
      ENTRY SPLENT (Z,MAX,YINT,DYDX,D2YDX2)
      DO 120 I=1,MAX
      K=2
      IF (ABS(Z(I)-X(1)).LT.TOLER) GO TO 40
      IF (Z(I).GT.2.0*X(1)-X(2)) 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 IF (K.GE.N) GO TO 90
   K = K+1
   GO TO 50
80 S2 = X(2)-X(1)
   Y0 = EM(1)*S2**2+2.*Y(1)-Y(2)
   DYDX(I) = (Y(2)-Y(1))/S2-7.*EM(1)/6.*S2
   YINT(I) = Y0+DYDX(I)*(Z(I)-X(1)+S2)
   D2YDX2(I) = 0.
   GO TO 120
90 IF (Z(I).LT.2.*X(N)-X(N-1)) GO TO 100
   SN = X(N)-X(N-1)
   YNP1 = EM(N)*SN**2+2.*Y(N)-Y(N-1)
   DYDX(I) = (Y(N)-Y(N-1))/SN+7.*EM(N)/6.*SN
   YINT(I) = YNP1+DYDX(I)*(Z(I)-X(N)-SN)
   D2YDX2(I) = 0.
   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.
   D2YDX2(I) = EM(K)-(X(K)-Z(I))/SK*(EM(K)-EM(K-1))
120 CONTINUE
   IF (IERR.EQ.0) RETURN
130 WRITE(NWRIT,1000)
   WRITE(NWRIT,1020) N,(X(I),Y(I),I=1,N)
   IF (IERR.EQ.0) STOP
   WRITE(NWRIT,1030)
   RETURN
140 WRITE(NWRIT,1010)
   WRITE(NWRIT,1020) N,(X(I),Y(I),I=1,N)
   STOP
1000 FORMAT (1H1,10X,44HSPLINT ERROR -- ONE OF THREE POSSIBLE CAUSES/
  117X,51H1.  ADJACENT X POINTS ARE DUPLICATES OF EACH OTHER./
  217X,38H2.  SOME X POINTS ARE OUT OF SEQUENCE./
  317X,32H3.  SOME X POINTS ARE UNDEFINED.)
1010 FORMAT (1H1,10X,62HSPLINT ERROR -- NUMBER OF SPLINE POINTS GIVEN I
  1S LESS THAN TWO)
1020 FORMAT (//17X,18HNUMBER OF POINTS =,I4//17X,8HX  ARRAY,6X,8HY  ARR
  1AY/(17X,2G13.5))
1030 FORMAT (1H1)
   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
```

```
      SUBROUTINE SPLISL(X,Y,N,Y1P,YNP,SLOPE,EM)
C
C--SPLISL CALCULATES FIRST AND SECOND DERIVATIVES AT SPLINE POINTS
C--END CONDITION - FIRST DERIVATIVES SPECIFIED AT END POINTS
C
      COMMON NREAD,NWRIT
      DIMENSION X(N),Y(N),SLOPE(N),EM(N)
      DIMENSION G(101),SB(101)
      IERR = 0
      C = X(2)-X(1)
      IF (C.EQ.0.) GO TO 50
      SB(1) = .5
      F = (Y(2)-Y(1))/C-Y1P
      G(1) = 3.*F/C
      NO = N-1
      IF (NO.LE.0) GO TO 60
      IF (NO.EQ.1) GO TO 20
      DO 10 I=2,NO
      A = C
      C = X(I+1)-X(I)
      IF (A*C.EQ.0.) GO TO 50
      IF (A*C.LT.0.) IERR = 1
      W = 2.*(A+C)-A*SB(I-1)
      SB(I) = C/W
      F = (Y(I+1)-Y(I))/C-(Y(I)-Y(I-1))/A
   10 G(I) = (6.*F-A*G(I-1))/W
   20 W = C*(2.-SB(N-1))
      F = YNP-(Y(N)-Y(N-1))/C
```

```
      EM(N) = (6.*F-C*G(N-1))/W
      DO 30 I=2,N
      K = N+1-I
   30 EM(K) = G(K)-SB(K)*EM(K+1)
      SLOPE(1) = Y1P
      DO 40 I=2,NO
   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))
      SLOPE(N) = YNP
      IF (IERR.EQ.0) RETURN
   50 WRITE(NWRIT,1000)
      WRITE(NWRIT,1020) N,(X(I),Y(I),I=1,N)
      IF (IERR.EQ.0) STOP
      WRITE(NWRIT,1030)
      RETURN
   60 WRITE(NWRIT,1010)
      WRITE(NWRIT,1020) N,(X(I),Y(I),I=1,N)
      STOP
 1000 FORMAT (1H1,10X,44HSPLISL ERROR -- ONE OF THREE POSSIBLE CAUSES/
     117X,51H1.  ADJACENT X POINTS ARE DUPLICATES OF EACH OTHER./
     217X,38H2.  SOME X POINTS ARE OUT OF SEQUENCE./
     317X,32H3.  SOME X POINTS ARE UNDEFINED.)
 1010 FORMAT (1H1,10X,62HSPLISL ERROR -- NUMBER OF SPLINE POINTS GIVEN I
     1S LESS THAN TWO)
 1020 FORMAT (//17X,18HNUMBER OF POINTS =,I4//17X,8HX  ARRAY,6X,8HY  ARR
     1AY/(17X,2G13.5))
 1030 FORMAT (1H1)
      END



      SUBROUTINE SPINSL(X,Y,N,Y1P,YNP,Z,MAX,YINT,DYDX,D2YDX2)
C
C--SPINSL CALCULATES INTERPOLATED POINTS AND DERIVATIVES
C--FOR A SPLINE CURVE
C--END CONDITION - FIRST DERIVATIVES SPECIFIED AT END POINTS
C
      COMMON NREAD,NWRIT
      DIMENSION X(N),Y(N),Z(MAX),YINT(MAX),DYDX(MAX),D2YDX2(MAX)
      DIMENSION G(101),SB(101),EM(101)
      IERR = 0
      TOLER= ABS(X(N)-X(1))/FLOAT(N)*1.E-5
      C = X(2)-X(1)
      IF (C.EQ.0.) GO TO 130
      SB(1) = .5
      F = (Y(2)-Y(1))/C-Y1P
      G(1) = 3.*F/C
      NO = N-1
      IF (NO.LE.0) GO TO 140
      IF (NO.EQ.1) GO TO 20
      DO 10 I=2,NO
      A = C
      C = X(I+1)-X(I)
      IF (A*C.EQ.0.) GO TO 130
      IF (A*C.LT.0.) IERR = 1
      W = 2.*(A+C)-A*SB(I-1)
      SB(I) = C/W
```

```
      F = (Y(I+1)-Y(I))/C-(Y(I)-Y(I-1))/A
   10 G(I) = (6.*F-A*G(I-1))/W
   20 W = C*(2.-SB(N-1))
      F = YNP-(Y(N)-Y(N-1))/C
      EM(N) = (6.*F-C*G(N-1))/W
      DO 30 I=2,N
      K = N+1-I
   30 EM(K) = G(K)-SB(K)*EM(K+1)
      IF (MAX.LE.0) RETURN
C
      ENTRY SPENSL (Z,MAX,YINT,DYDX,D2YDX2)
      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 IF (K.GE.N) GO TO 90
      K = K+1
      GO TO 50
   80 DYDX(I) = Y1P
      YINT(I) = Y(1)+Y1P*(Z(I)-X(1))
      D2YDX2(I) = 0.
      GO TO 120
   90 DYDX(I) = YNP
      YINT(I) = Y(N)+YNP*(Z(I)-X(N))
      D2YDX2(I) = 0.
      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.
      D2YDX2(I) = EM(K)-(X(K)-Z(I))/SK*(EM(K)-EM(K-1))
  120 CONTINUE
      IF (IERR.EQ.0) RETURN
  130 WRITE(NWRIT,1000)
      WRITE(NWRIT,1020) N,(X(I),Y(I),I=1,N)
      IF (IERR.EQ.0) STOP
      WRITE(NWRIT,1030)
      RETURN
  140 WRITE(NWRIT,1010)
      WRITE(NWRIT,1020) N,(X(I),Y(I),I=1,N)
      STOP
 1000 FORMAT (1H1,10X,44HSPINSL ERROR -- ONE OF THREE POSSIBLE CAUSES/
    117X,51H1.  ADJACENT X POINTS ARE DUPLICATES OF EACH OTHER./
    217X,38H2.  SOME X POINTS ARE OUT OF SEQUENCE./
    317X,32H3.  SOME X POINTS ARE UNDEFINED.)
 1010 FORMAT (1H1,10X,62HSPINSL ERROR -- NUMBER OF SPLINE POINTS GIVEN I
    1S LESS THAN TWO)
 1020 FORMAT (//17X,18HNUMBER OF POINTS =,I4//17X,8HX  ARRAY,6X,8HY  ARR
```

```
     1AY/(17X,2G13.5))
 1030 FORMAT (1H1)
      END
```

```
  DATA 12 - AXIAL COMPRESSOR ROTOR - INLET WHIRL - S.I. UNITS
1.4        287.053   42.829880 -826.5498 0.99999   0.01      0.5       0.5
  11   31   41   20   17    6    6   11   11   11   15    4   11   11
   1    0    1
 -0.045720  0.          0.121920  0.167640 -0.        -0.        -0.        -0.
 -0.091410 -0.040630  0.010180  0.060990  0.137190  0.213390 -0.        -0.
  0.117866  0.143012  0.155448  0.161300  0.164074  0.164592 -0.        -0.
 -0.091410 -0.040630  0.010180  0.060990  0.137190  0.213390 -0.        -0.
  0.324612  0.315986  0.311140  0.308366  0.307330  0.306568 -0.        -0.
 -0.015210 -0.015210 -0.        -0.        -0.        -0.        -0.        -0.
  0.150663  0.173766  0.193335  0.211318  0.228051  0.243779  0.258684  0.272857
  0.286360  0.299283  0.313121 -0.        -0.        -0.        -0.        -0.
288.15    288.15    288.15    288.15    288.15    288.15    288.15    288.15
288.15    288.15    288.15
101352.93 101352.93 101352.93 101352.93 101352.93 101352.93 101352.93 101352.93
101352.93 101352.93 101352.93
 -5.943600-15.136368-21.028152-25.746456-29.733240-33.186624-36.234624-38.953440
-41.404032-43.607736-45.796200 -0.        -0.        -0.        -0.        -0.
  0.137190  0.137190 -0.        -0.        -0.        -0.        -0.        -0.
  0.164074  0.183703  0.199735  0.214793  0.229027  0.242651  0.255727  0.268407
  0.280690  0.292730  0.307330 -0.        -0.        -0.        -0.        -0.
129221.63 129221.63 129221.63 129221.63 129221.63 129221.63 129221.63 129221.63
129221.63 129221.63 129221.63
-168.3715 -159.1483 -153.3876 -149.5227 -146.6027 -144.3655 -142.9390 -141.8753
-141.1803 -142.2349 -146.4107
 -0.        0.001128  0.006308  0.013692  0.022299  0.033270  0.044422  0.055700
  0.067045  0.078400  0.089707  0.098679  0.106450  0.111945  0.112818 -0.
  0.001305  0.002325  0.007352  0.014530  0.022915  0.033637  0.044579  0.055695
  0.066934  0.078249  0.089588  0.098644  0.106534  0.112141  0.112984 -0.
  0.002751  0.003683  0.008551  0.015512  0.023659  0.034101  0.044794  0.055697
  0.066768  0.077965  0.089245  0.098300  0.106226  0.111882  0.112691 -0.
  0.004231  0.005092  0.009809  0.016559  0.024469  0.034626  0.045051  0.055710
  0.066567  0.077589  0.088737  0.097723  0.105618  0.111269  0.112045 -0.
  0.005713  0.006515  0.011091  0.017640  0.025319  0.035190  0.045336  0.055731
  0.066344  0.077146  0.088107  0.096971  0.104781  0.110385  0.111129 -0.
  0.007170  0.007926  0.012370  0.018730  0.026187  0.035776  0.045640  0.055758
  0.066104  0.076656  0.087389  0.096088  0.103770  0.109294  0.110006 -0.
  0.008562  0.009280  0.013603  0.019786  0.027035  0.036354  0.045944  0.055785
  0.065859  0.076146  0.086626  0.095135  0.102663  0.108082  0.108766 -0.
  0.009889  0.010577  0.014789  0.020809  0.027861  0.036923  0.046245  0.055812
  0.065609  0.075620  0.085829  0.094128  0.101479  0.106776  0.107434 -0.
  0.011142  0.011806  0.015916  0.021786  0.028654  0.037471  0.046536  0.055834
  0.065356  0.075087  0.085017  0.093093  0.100252  0.105414  0.106048 -0.
  0.012245  0.012890  0.016909  0.022644  0.029348  0.037947  0.046780  0.055839
  0.065112  0.074589  0.084261  0.092132  0.099110  0.104145  0.104759 -0.
  0.013208  0.013839  0.017775  0.023387  0.029945  0.038350  0.046979  0.055825
  0.064878  0.074131  0.083574  0.091260  0.098076  0.102996  0.103595 -0.
  0.155483  0.155588  0.156068  0.156752  0.157550  0.158566  0.159600  0.160645
  0.161696  0.162749  0.163796  0.164628  0.165348  0.165857  0.165938 -0.
  0.176386  0.176450  0.176765  0.177216  0.177742  0.178415  0.179102  0.179800
  0.180505  0.181215  0.181927  0.182496  0.182991  0.183343  0.183396 -0.
  0.195260  0.195296  0.195485  0.195755  0.196071  0.196475  0.196890  0.197313
  0.197742  0.198176  0.198613  0.198964  0.199271  0.199490  0.199522 -0.
  0.212607  0.212623  0.212713  0.212841  0.212991  0.213183  0.213380  0.213582
  0.213788  0.213997  0.214208  0.214378  0.214528  0.214635  0.214650 -0.
```

0.228744	0.228745	0.228756	0.228772	0.228790	0.228814	0.228838	0.228862
0.228888	0.228914	0.228940	0.228961	0.228979	0.228993	0.228994	-0.
0.243892	0.243883	0.243832	0.243758	0.243671	0.243560	0.243445	0.243328
0.243208	0.243085	0.242960	0.242859	0.242770	0.242706	0.242697	-0.
0.258213	0.258196	0.258095	0.257951	0.257781	0.257563	0.257339	0.257109
0.256873	0.256632	0.256387	0.256188	0.256012	0.255885	0.255869	-0.
0.271827	0.271804	0.271664	0.271464	0.271231	0.270930	0.270621	0.270303
0.269978	0.269646	0.269308	0.269032	0.268788	0.268613	0.268591	-0.
0.284826	0.284799	0.284630	0.284388	0.284106	0.283744	0.283370	0.282988
0.282597	0.282196	0.281788	0.281456	0.281161	0.280949	0.280923	-0.
0.297289	0.297259	0.297070	0.296802	0.296488	0.296085	0.295672	0.295248
0.294814	0.294370	0.293917	0.293549	0.293222	0.292987	0.292958	-0.
0.309276	0.309245	0.309052	0.308776	0.308455	0.308042	0.307618	0.307184
0.306740	0.306286	0.305823	0.305446	0.305111	0.304870	0.304841	-0.
-0.	0.003439	0.018370	0.037211	0.055758	0.074409	0.087979	0.096447
0.099821	0.098149	0.091503	0.082670	0.072427	0.063700	0.062213	
-0.013224	-0.010074	0.004766	0.023910	0.043414	0.064167	0.080826	0.093339
0.101676	0.105831	0.105819	0.102830	0.098063	0.093440	0.092668	
-0.023111	-0.020171	-0.005378	0.014009	0.034235	0.056558	0.075521	0.091061
0.103130	0.111696	0.116741	0.118239	0.117703	0.116268	0.116000	
-0.029779	-0.027018	-0.012353	0.007090	0.027724	0.051073	0.071641	0.089362
0.104184	0.116061	0.124961	0.129922	0.132681	0.133747	0.133841	
-0.034282	-0.031670	-0.017193	0.002177	0.023000	0.047005	0.068698	0.088021
0.104918	0.119344	0.131259	0.138962	0.144355	0.147433	0.147798	
-0.037269	-0.034785	-0.020532	-0.001322	0.019542	0.043942	0.066421	0.086925
0.105407	0.121820	0.136124	0.146026	0.153552	0.158271	0.158843	
-0.039040	-0.036662	-0.022655	-0.003667	0.017126	0.041714	0.064700	0.086037
0.105681	0.123592	0.139731	0.151345	0.160545	0.166562	0.167290	
-0.040000	-0.037711	-0.023963	-0.005234	0.015412	0.040050	0.063351	0.085277
0.105790	0.124854	0.142437	0.155415	0.165964	0.173032	0.173882	
-0.040327	-0.038112	-0.024628	-0.006183	0.014263	0.038847	0.062317	0.084642
0.105790	0.125733	0.144440	0.158498	0.170124	0.178039	0.178986	
-0.039946	-0.037791	-0.024571	-0.006431	0.013762	0.038175	0.061644	0.084141
0.105641	0.126119	0.145548	0.160322	0.172676	0.181170	0.182184	
-0.039061	-0.036953	-0.023993	-0.006169	0.013734	0.037894	0.061235	0.083735
0.105370	0.126120	0.145962	0.161168	0.173978	0.182840	0.183900	
0.002362	0.002624	0.003829	0.005310	0.006707	0.008007	0.008809	0.009110
0.008862	0.008018	0.006572	0.004975	0.003254	0.001838	0.001613	
0.002173	0.002395	0.003490	0.004832	0.006094	0.007261	0.007972	0.008226
0.007986	0.007221	0.005926	0.004504	0.002976	0.001723	0.001535	
0.002035	0.002228	0.003234	0.004464	0.005618	0.006682	0.007324	0.007545
0.007318	0.006615	0.005435	0.004146	0.002765	0.001632	0.001470	
0.001933	0.002103	0.003036	0.004178	0.005246	0.006228	0.006818	0.007016
0.006799	0.006147	0.005058	0.003870	0.002600	0.001560	0.001417	
0.001855	0.002009	0.002885	0.003955	0.004956	0.005874	0.006423	0.006604
0.006397	0.005784	0.004765	0.003656	0.002472	0.001502	0.001373	
0.001798	0.001939	0.002769	0.003784	0.004732	0.005601	0.006119	0.006286
0.006087	0.005505	0.004540	0.003492	0.002373	0.001457	0.001339	
0.001755	0.001887	0.002683	0.003655	0.004563	0.005394	0.005889	0.006047
0.005855	0.005296	0.004372	0.003368	0.002298	0.001422	0.001311	
0.001724	0.001850	0.002622	0.003563	0.004442	0.005246	0.005723	0.005875
0.005687	0.005146	0.004250	0.003279	0.002243	0.001397	0.001292	
0.001705	0.001827	0.002581	0.003502	0.004361	0.005147	0.005613	0.005760
0.005576	0.005045	0.004170	0.003220	0.002207	0.001379	0.001277	
0.001694	0.001814	0.002559	0.003468	0.004316	0.005090	0.005550	0.005695
0.005513	0.004989	0.004124	0.003186	0.002186	0.001369	0.001269	
0.001692	0.001811	0.002552	0.003457	0.004301	0.005072	0.005530	0.005675
0.005493	0.004972	0.004110	0.003176	0.002180	0.001366	0.001267	
-0.015210	0.	0.112819	0.137190	-0.	-0.	-0.	-0.
-0.015210	0.013207	0.103595	0.137190	-0.	-0.	-0.	-0.
0.00	0.10	0.20	0.30	0.40	0.50	0.60	0.70
0.80	0.90	1.00					
0.0	0.1	0.2	0.3	0.4	0.5	0.6	0.7

```
  0.8           0.9           1.0
  0.0           0.080         0.158         0.235         0.314         0.400         0.490         0.584
  0.693         0.822         1.0
     20     20     20     5     1     20     0
```

Lewis Research Center,
National Aeronautics and Space Administration,
Cleveland, Ohio, March 22, 1977,
505-04.

APPENDIX A

FINITE-DIFFERENCE FORM OF STREAM-FUNCTION EQUATION

The stream-function equation was derived as equation (B17) of part I (ref. 6):

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

$$+ \frac{rB\rho}{wW_s}\left[\frac{W_\theta}{r}\frac{\partial(rV_\theta)}{\partial t} + \xi W^2 + \zeta + F_t\right] = 0 \qquad \text{(A1)}$$

where

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

$$\zeta = \omega^2 r \cos\varphi - \frac{RT''}{p''}\frac{\partial p''}{\partial t} \qquad \text{(A3)}$$

$$F_t = \frac{\partial \theta}{\partial t}\frac{1}{\rho}\frac{\partial p}{\partial \theta} \qquad \text{(A4)}$$

Equation (A4) was derived as equation (B5) of part I.

The s and t are the distances 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 of 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-

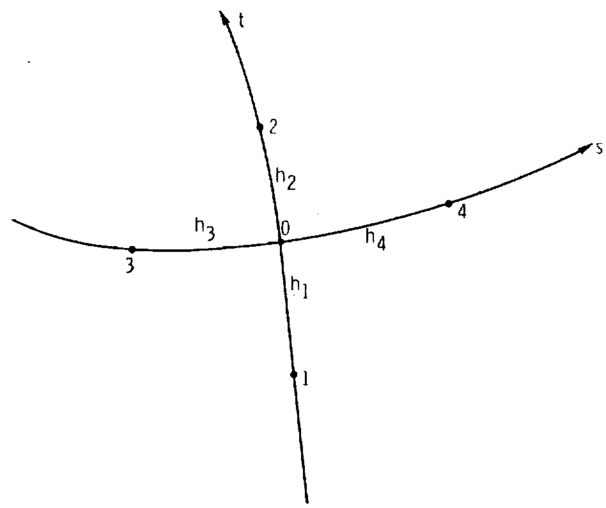


Figure 24. - Notation for adjacent mesh points and mesh spaces.

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

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

$$-\frac{u_4-u_3}{h_3+h_4}\left[\frac{\sin\varphi_0}{r_0}+\frac{1}{B_0}\left(\frac{B_4-B_3}{h_3+h_4}\right)+\frac{1}{\rho_0}\left(\frac{\rho_4-\rho_3}{h_3+h_4}\right)-\left(\frac{\partial\varphi}{\partial t}\right)_0\right]$$

$$-\frac{u_2-u_1}{h_1+h_2}\left[\frac{\cos\varphi_0}{r_0}+\frac{1}{B_0}\left(\frac{B_2-B_1}{h_1+h_2}\right)+\frac{1}{\rho_0}\left(\frac{\rho_2-\rho_1}{h_1+h_2}\right)+\left(\frac{\partial\varphi}{\partial s}\right)_0\right]$$

$$+\frac{r_0B_0\rho_0}{w(W_s)_0}\left\{\frac{(W_\theta)_0}{r_0}\left[\frac{\partial(rV_\theta)}{\partial t}\right]_0+\xi_0W_0^2+\zeta_0+(F_t)_0\right\}=0 \quad \text{(A5)}$$

where $\partial(rV_\theta)/\partial t$ is calculated by different methods upstream, downstream, and within the blade row. Upstream and downstream of the blade, equations (B21) and (B22) of part I are used. Within the blade row, 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

$$\left[\frac{\partial(rV_\theta)}{\partial t}\right]_0 = \begin{cases} \dfrac{r_0 B_0 \rho_0 (W_s)_0}{w}\left(\dfrac{d\lambda}{du}\right)_0 & \text{upstream} \\[2ex] \dfrac{\left[r_2(V_\theta)_2 - r_1(V_\theta)_1\right]}{h_1 + h_2} & \text{within blade row} \\[2ex] \dfrac{r_0 B_0 \rho_0 (W_s)_0}{w}\left[\dfrac{d(rV_\theta)_0}{du}\right]_0 & \text{downstream} \end{cases} \tag{A6}$$

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

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

where the coefficients are calculated as follows:

$$\left.\begin{aligned} a_0 &= \frac{2}{h_1 h_2} + \frac{2}{h_3 h_4} \\ c_1 &= h_1 + h_2 \\ c_2 &= h_3 + h_4 \\ \left(\frac{\partial\varphi}{\partial s}\right)_0 &= \begin{cases} \dfrac{\sin\varphi_4 - \sin\varphi_3}{c_2 \cos\varphi_0} & \text{if } |\cos\varphi_0| \geq \dfrac{\sqrt{2}}{2} \\[2ex] \dfrac{\cos\varphi_3 - \cos\varphi_4}{c_2 \sin\varphi_0} & \text{if } |\cos\varphi_0| < \dfrac{\sqrt{2}}{2} \end{cases} \end{aligned}\right\} \tag{A8}$$

(continued)

$$
\left.\begin{aligned}
\left(\frac{\partial\varphi}{\partial t}\right)_0 &= \begin{cases} \dfrac{\sin\varphi_2 - \sin\varphi_1}{c_1 \cos\varphi_0} & \text{if } |\cos\varphi_0| \geq \dfrac{\sqrt{2}}{2} \\[2ex] \dfrac{\cos\varphi_1 - \cos\varphi_2}{c_1 \sin\varphi_0} & \text{if } |\cos\varphi_0| < \dfrac{\sqrt{2}}{2} \end{cases} \\[2ex]
d_1 &= \frac{\dfrac{B_2 - B_1}{B_0} + \dfrac{\rho_2 - \rho_1}{\rho_0}}{c_1} + \frac{\cos\varphi_0}{r_0} + \left(\frac{\partial\varphi}{\partial s}\right)_0 \\[2ex]
d_2 &= \frac{\dfrac{B_4 - B_3}{B_0} + \dfrac{\rho_4 - \rho_3}{\rho_0}}{c_2} + \frac{\sin\varphi_0}{r_0} - \left(\frac{\partial\varphi}{\partial t}\right)_0 \\[2ex]
a_1 &= \frac{\left(\dfrac{2}{h_1} + d_1\right)}{a_0 c_1} \\[2ex]
a_2 &= \frac{\left(\dfrac{2}{h_2} - d_1\right)}{a_0 c_1} \\[2ex]
a_3 &= \frac{\left(\dfrac{2}{h_3} + d_2\right)}{a_0 c_2} \\[2ex]
a_4 &= \frac{\left(\dfrac{2}{h_4} - d_2\right)}{a_0 c_2}
\end{aligned}\right\} \qquad \text{(A8)}
$$

$$
k_0 = \begin{cases} \dfrac{r_0 B_0 \rho_0}{a_0 w (W_s)_0} \left[\dfrac{(W_\theta)_0 B_0 \rho_0 (W_s)_0}{w} \left(\dfrac{d\lambda}{du}\right)_0 + \xi_0 W_0^2 + \zeta_0 \right] & \text{upstream} \\[2ex] \dfrac{r_0 B_0 \rho_0}{a_0 w (W_s)_0} \left\{ \dfrac{(W_\theta)_0}{r_0} \left[\dfrac{r_2 (V_\theta)_2 - r_1 (V_\theta)_1}{c_1} \right] + \xi_0 W_0^2 + \zeta_0 + (F_t)_0 \right\} & \text{within blade row} \\[2ex] \dfrac{r_0 B_0 \rho_0}{a_0 w (W_s)_0} \left\{ \dfrac{(W_\theta)_0 B_0 \rho_0 (W_s)_0}{w} \left[\dfrac{d(rV_\theta)_o}{du} \right]_0 + \xi_0 W_0^2 + \zeta_0 \right\} & \text{downstream} \end{cases} \tag{A9}
$$

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_t is calculated in subroutine BLDVEL when the blade surface velocities are calculated. The quantities $d\lambda/du$ and $d(rV_\theta)_o/du$ are calculated by subroutines LAMDAF and RVTHTA when they are called by NEWRHO to calculate λ or $(rV_\theta)_o$.

Equation (A8) is used at all interior points of the mesh region. Along the boundaries, the boundary conditions give different coefficients. The stream function is known to be 0.0 at the hub and 1.0 at the shroud. At the upstream and downstream boundaries, the boundary condition is that the normal derivative of the stream function is zero. The finite difference expression for this is

$$u_0 = u_4 \quad \text{on the upstream boundary}$$

$$u_0 = u_3 \quad \text{on the downstream boundary}$$

Since the coefficients for these equations do not depend on the solution, they are specified in subroutine INIT.

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 LOSSOM if LAMVT = 1. If LSFR = 0, SFIN and SFOUT are given as input, and RADIN and RADOUT are corrected by the initialization 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 four subroutines - LAMDAF, RVTHTA, TIPF, and RHOIPF.

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 that lie between the leading and trailing edges of the blade. The information needed to make this calculation exists as $\theta(z, r)$ on the input blade sections. The THETOM procedure is designed so that an accurate calculation is maintained in the transition from input blade mesh to orthogonal mesh.

The orthogonal mesh on a typical blade is illustrated in figure 25. Note that some of the t mesh lines cross the leading and trailing edges of the blade. To alleviate the problem of calculating θ-gradients on this mesh, they are first obtained on an alternate mesh, shown in figures 26 and 27, of s'- and t'-coordinates. Then, by interpolation, $\partial\theta/\partial s$ and $\partial\theta/\partial t$ are obtained at the desired orthogonal mesh points.

There are several reasons why it is convenient to use an alternate mesh to calculate $\partial\theta/\partial s$ and $\partial\theta/\partial t$. First, there are usually not sufficient input planes or points

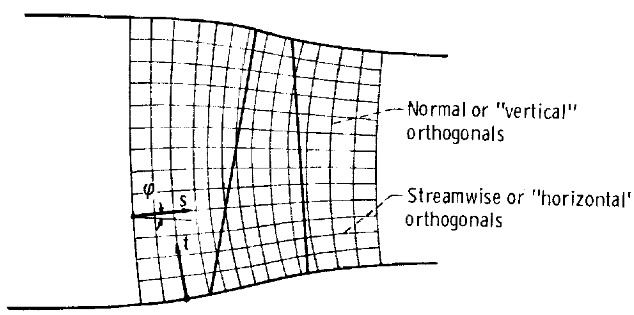


Figure 25. - Orthogonal finite-difference mesh on solution region.

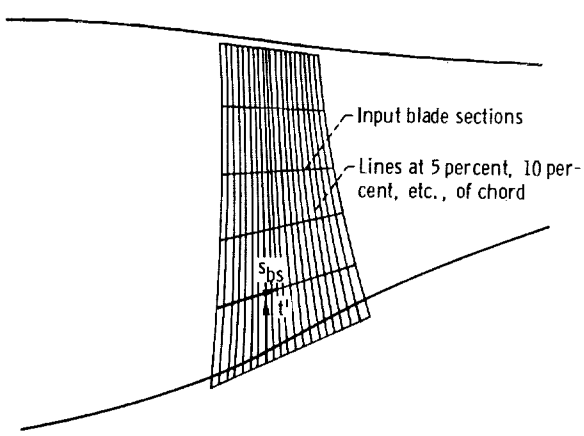


Figure 26. - Semi-alternate mesh with 5-percent-chord hub-shroud lines.

to permit an accurate direct calculation of $\partial\theta/\partial s$ and $\partial\theta/\partial t$ using the input blade-section points alone. Second, corresponding points on adjacent input blade planes are not required to fall on smooth curves from hub to shroud. Finally, the angle φ is known only on the orthogonal mesh, and not at input points, so that $\partial\theta/\partial s$ and $\partial\theta/\partial t$ cannot be obtained directly at the input points and then interpolated to the orthogonal mesh. Therefore, a fine-grid alternate mesh is used on which $\partial\theta/\partial z$ and $\partial\theta/\partial r$ are calculated. These are then interpolated to the required orthogonal mesh points and transformed to $\partial\theta/\partial s$ and $\partial\theta/\partial t$. Note that it is more accurate to calculate partial derivatives first and then interpolate and transform the partials to the s- and t-directions, than it would be to interpolate θ itself from the input mesh to the orthogonal mesh and then calculate the partials along mesh lines.

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

(1) Calculate rotated z-coordinates (ZPC) of points along the input blade sections at 5-percent-meridional-chord locations, that is, at the semi-alternate mesh points of

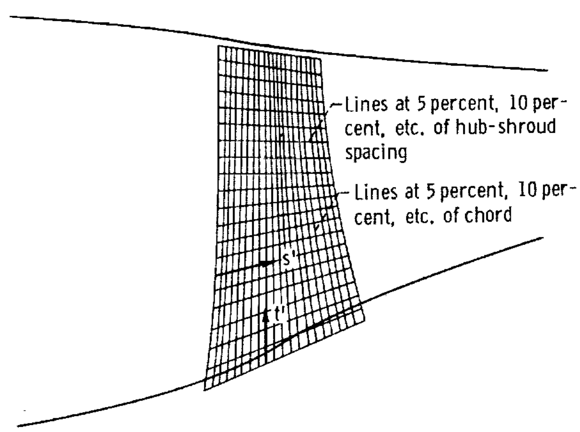


Figure 27. - Full alternate mesh on which gradients of θ are obtained.

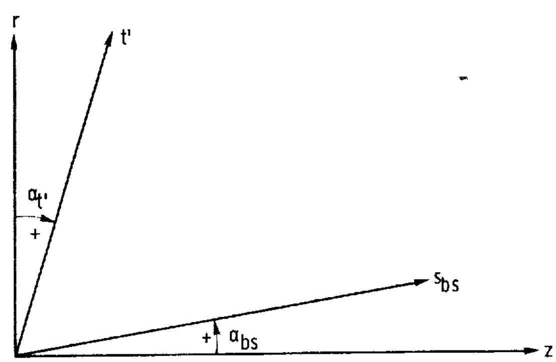


Figure 28. - Relation of semi-alternate mesh to z- and r-directions.

figure 26.

(2) Use SPLINT calls along each input blade section to obtain corresponding rotated r-coordinates (RPC) and angles with respect to the unrotated z-axis α_{bs} (fig. 28).

(3) Calculate arc length SZRBL along each input blade section (s_{bs} direction) using the ZBL, RBL coordinates.

(4) Calculate arc length SZRPC along the same blade sections using the calculated ZPC, RPC coordinates of the semi-alternate mesh.

(5) Use SPLINT calls in the s_{bs}-direction (or SPINSL if BETALE and BETATE are specified) to calculate θ and $\partial\theta/\partial s_{bs}$ at the ZPC, RPC points from known θ at the ZBL, RBL points.

(6) Use SPLINT calls in the s_{bs}-direction to calculate blade thickness in the θ-direction (TTPC) at the ZPC, RPC points from TTBL at the ZBL, RBL points. This concludes the calculation of variables at the semi-alternate mesh points of figure 26.

(7) A procedure is then begun to obtain required distances, angles, and gradients on a finer grid of points along the lines in the t'-direction, that is, at the points of the full alternate mesh of figure 27. Store values of z, r, θ, blade thickness, α_{bs}, and $\partial\theta/\partial s_{bs}$ into arrays along the t'-lines.

(8) Calculate arc length SZRBL along the t'-lines using the ZPC, RPC coordinates (stored for each line in ZPCT1, RPCT1).

(9) Calculate r-coordinates (RPCT2) of points along the t'-lines at 5-percent distance increments from hub to shroud (where s' and t' cross, fig. 27). Use SPLINT calls to obtain corresponding z-coordinates (ZPCT2).

(10) Calculate arc length SZRPC up the t'-lines using the ZPCT2, RPCT2 coordinates of the full alternate mesh. Also calculate angles with respect to the unrotated r-axis $\alpha_{t'}$ (fig. 28).

(11) Use SPLINT calls in the t'-direction to obtain θ and $\partial\theta/\partial t'$ at the full alternate mesh points (ZPCT2, RPCT2) from known θ at the semi-alternate mesh points (ZPCT1, RPCT1).

(12) Use SPLINT calls in the t'-direction to obtain α_{bs}, $\partial\theta/\partial s_{bs}$, and blade thickness at the full alternate mesh points (ZPCT2, RPCT2) from known values at the semi-alternate mesh points (ZPCT1, RPCT1).

(13) Store calculated values of z, r, θ, blade thickness, α_{bs}, and $\partial\theta/\partial s_{bs}$ at the full alternate mesh points into two-dimensional arrays ZPC, RPC, THPC, TTPC, ANGZ, and DTHDSP. This procedure, from step 7 to step 13, is executed for each of the t'-lines of the alternate mesh.

(14) Calculate $\partial\theta/\partial z$ and $\partial\theta/\partial r$ from $\partial\theta/\partial s_{bs}$ and $\partial\theta/\partial t'$ at the s'- and t'-points of the full alternate mesh with the following equations:

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

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

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

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

(16) Transform the $\partial\theta/\partial z$ and $\partial\theta/\partial r$ to obtain $\partial\theta/\partial s$ and $\partial\theta/\partial t$ at the orthogonal mesh points within the blade (fig. 25). The following equations are used:

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

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

APPENDIX D

INCOMPRESSIBLE STREAM-FUNCTION EQUATION

The stream-function equation is modified slightly for incompressible flow. The only terms that must be altered in equation (B14) of part I are the two terms $\partial I/\partial t$ and $T\ \partial s/\partial t$. Since it is assumed that there is uniform total pressure upstream and no total pressure loss, $\partial s/\partial t = 0$. By definition, $I = H_i - \omega\lambda$. For uniform upstream stagnation conditions, H_i is constant, so that

$$\frac{\partial I}{\partial t} = -\omega \frac{\partial \lambda}{\partial t} \qquad \text{(D1)}$$

For incompressible flow, equation (D1) is used instead of equation (B16) of part I in equation (B14) of part I. The result is that equations (A1) and (A4) are unchanged and equations (A2) and (A3) are replaced by

$$\xi = 0 \qquad \text{(D2)}$$

$$\zeta = \omega \frac{\partial \lambda}{\partial t} \qquad \text{(D3)}$$

However, in the program, the variable ZETA is not used for this purpose, but $\omega\ \partial\lambda/\partial t$ is added at the proper point in subroutine COEF.

APPENDIX E

GENERATION OF LEADING- AND TRAILING-EDGE RADII ON TSONIC BLADE SECTIONS

In the TSONIN subroutine, blade-section geometry along streamlines is calculated for input to the TSONIC program (ref. 3). In this process, leading- and trailing-edge radii must be generated within the blade-coordinate envelope used in MERIDL.

The blade envelope in the leading-edge region along a streamline for MERIDL is shown in figure 29. The envelope has thickness at the leading edge, which is the way blockage is modeled in MERIDL. The points where the meridional streamline is intersected by MERIDL vertical orthogonal mesh lines are also indicated. These are projected in the θ-direction to the blade surfaces to obtain potential TSONIC input points, as shown in figure 29. Some of these points near the leading or trailing edge will later be eliminated.

Subroutine TSONIN calculates a leading-edge radius within the envelope of figure 29, as shown in figure 30. This leading-edge radius touches three sides of the blade-section envelope and is entirely contained within it. The technique for calculating this radius is as follows. A similar technique is used at the trailing edge.

(1) Calculate the r-coordinate at the leading edge of the blade section r_{le}. This r_{le} will be used for all points in the leading-edge region, not just those at A and B.

(2) Initially set a counter (ICOUNT) to zero and a damping factor (DAMP) to 1.

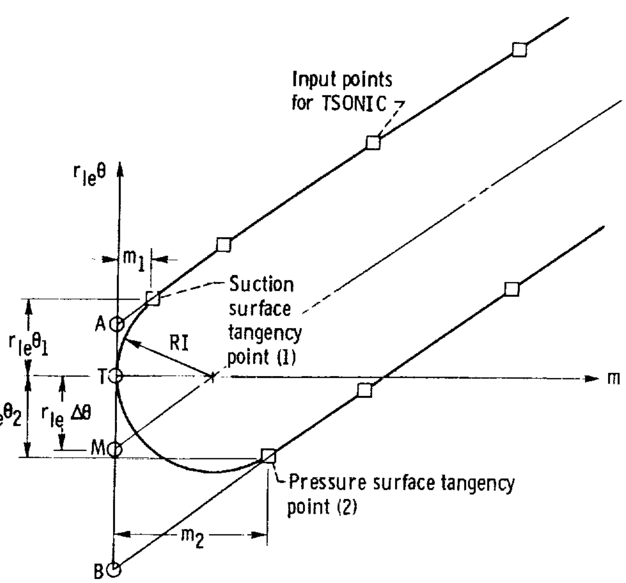


Figure 30. - Envelope of blade surface coordinates with respect to TSONIC origin.

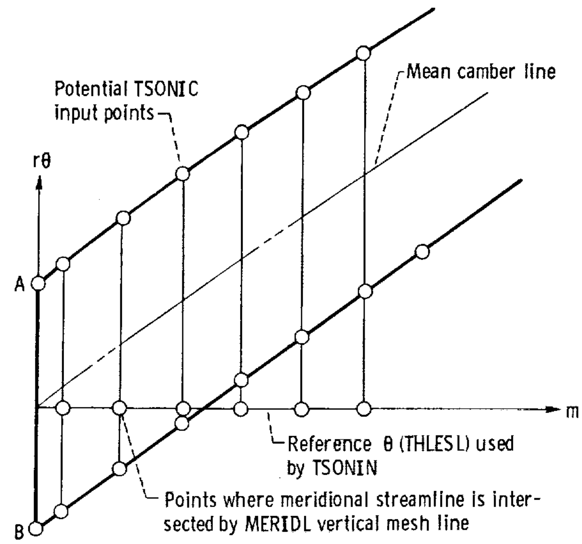


Figure 29. - Envelope of blade surface for MERIDL flow model.

(3) Initially estimate the tangency angles β_1 and β_2 (at points 1 and 2, fig. 30) from the slopes of the two blade surfaces at their end points (A and B in fig. 29).

(4) Initially estimate a leading-edge radius, RI, using θ-coordinates at points A and B:

$$RI = \frac{r_{le}\theta_A - r_{le}\theta_B}{2} \cos\left(\frac{\beta_1 + \beta_2}{2}\right)$$

(5) With the estimated RI, calculate m-coordinates of tangency points from (fig. 30)

$$m_1 = RI\left[1. - \sin(\beta_1)\right]$$

$$m_2 = RI\left[1. + \sin(\beta_2)\right]$$

(6) With SPLINT calls on each of the blade surfaces and m_1 and m_2, calculate new estimates of the tangency point θ-coordinates θ_1 and θ_2 and surface slopes $d\theta_1/dm$ and $d\theta_2/dm$ at these points.

(7) Using r_{le} and the surface slopes, calculate a new estimate of the tangency angles β_1 and β_2:

$$\beta_1 = \tan^{-1} \frac{r_{le}\, d\theta_1}{dm}$$

$$\beta_2 = \tan^{-1} \frac{r_{le}\, d\theta_2}{dm}$$

(8) Estimate new leading-edge radius, using updated tangency point θ-coordinates and β's:

$$RI_{new} = \frac{r_{le}\theta_1 - r_{le}\theta_2}{\cos\beta_1 + \cos\beta_2}$$

(9) Check relative change in RI. If tolerance is met, RI is set to RI_{new} and accepted. If the tolerance is not met, RI is recalculated as follows:

$$RI = \frac{(DAMP)(RI) + RI_{new}}{DAMP + 1.}$$

and the iteration loop from steps 5 to 9 is repeated.

If the calculated RI ever becomes negative in this procedure, DAMP is incremented by 1 and the process is begun again at step 3. The counter ICOUNT is incremented by 1 for each loop through steps 5 to 9. If ICOUNT reaches 100, an error message is printed and the current value of RI is accepted.

The leading-edge radius calculated by this process will be that shown in figure 30. The TSONIC origin in this figure is at point T, and the MERIDL origin is at point M. The $\Delta\theta$ from point M to point T is calculated as follows:

$$\Delta\theta = \frac{\theta_1\cos\beta_2 + \theta_2\cos\beta_1}{\cos\beta_1 + \cos\beta_2}$$

and is subtracted from the surface coordinates relative to MERIDL origin to obtain those relative to TSONIC origin.

APPENDIX F

CALCULATION OF CHANGE OF DENSITY DUE TO BLOCKAGE AT BLADE LEADING OR TRAILING EDGE

For the transonic velocity-gradient solution, incidence and deviation corrections to the assumed midchannel stream surface are made near the leading and trailing edges, as described in appendix F of part I. There is no correct existing value for the density within the blade row to use in this process. However, this density, ρ_{bf}, can be calculated from the free-stream density ρ_{fs} by making a blockage correction with the continuity equation. Iteration is required to solve the equation involved in this calculation. This calculation is done in subroutine LINDV.

From the assumption of continuous angular momentum and from continuity across the leading and trailing edges (but allowing W_m to be discontinuous), the following equation is derived as equation (F1) of part I:

$$\tan \beta_{bf} = \left(\frac{\rho_{bf}}{\rho_{fs}}\right)\left(\frac{B_{le}}{\text{Pitch}}\right)\tan \beta_{fs} \tag{F1}$$

Also, from continuity, we have

$$(\rho W \cos \beta)_{fs} \, (\text{Pitch}) = (\rho W \cos \beta)_{bf} \, B_{le}$$

By using the relation $\cos^2\beta = 1/(1 + \tan^2\beta)$, we can solve for W_{bf}^2 to obtain

$$W_{bf}^2 = W_{fs}^2 \frac{\left(1 + \tan^2\beta_{bf}\right)}{\left(1 + \tan^2\beta_{fs}\right)} \frac{\rho_{fs}^2}{\rho_{bf}^2} \frac{\text{Pitch}^2}{B_{le}^2} \tag{F2}$$

Finally, W_{bf} is related to ρ_{bf} by the relation

$$\rho_{bf} = \rho_i'(1 - \text{Ploss})\left[1 - \frac{W_{bf}^2 + 2\omega\lambda - (\omega r)^2}{2c_p T_i'}\right]^{1/(\gamma-1)} \tag{F3}$$

where

$$\text{Ploss} = \frac{p''_{ideal} - p''}{p''_{ideal}}$$

Equations (F1) to (F3) can be solved iteratively. The procedure to be used depends, however, on whether the meridional component of velocity is subsonic or supersonic. Also, it should be noted that with high subsonic velocity $(W_m)_{fs}$, there may be no solution possible for $(\rho W_m)_{bf}$, especially with large blockage.

The equations used for the iterative solution are as follows: Let

$$k_1 = \frac{(\tan \beta_{fs})B_{le}}{\rho_{fs}(\text{Pitch})} \tag{F4}$$

$$k_2 = \frac{\left(\rho_{fs} W_{fs} \dfrac{\text{Pitch}}{B_{le}}\right)^2}{1 + \tan^2 \beta_{fs}} \tag{F5}$$

In the program code, k_1 is the variable CONST1 and k_2 is CONST2. For the initial estimate in the iteration, use $\rho_{bf} = \rho_{fs}$, which is already known.

The usual case is when W_m is subsonic. In this case the sequence is to calculate $\tan \beta_{bf}$, then W^2_{bf}, followed by the new ρ_{bf}. The equations for this (from eq. (F1) to (F3)) are

$$\tan \beta_{bf} = k_1 \rho_{bf} \tag{F6}$$

$$W^2_{bf} = \frac{k_2\left(1 + \tan^2 \beta_{bf}\right)}{\rho^2_{bf}} \tag{F7}$$

$$\rho_{bf} = \rho'_i(1 - \text{Ploss})\left[1 - \frac{W^2_{bf} + 2\omega\lambda - (\omega r)^2}{2c_p T'_i}\right]^{1/(\gamma-1)} \tag{F8}$$

Equations (F6) to (F8) are then iterated. This will converge to the subsonic solution if it exists.

The other case is when W_m is supersonic. In this case the sequence is reversed to calculate W^2_{bf}, followed by the new ρ_{bf}. The equations for this (from eq. (F1) to (F3)) are

$$W_{bf}^2 = \left\{1 - \left[\frac{\rho_{bf}}{\rho_i'(1 - \text{Ploss})}\right]^{\gamma-1}\right\} 2c_p T_i' - 2\omega\lambda + (\omega r)^2 \tag{F9}$$

$$\rho_{bf} = \sqrt{\frac{k_2}{W_{bf}^2 - k_2 k_1^2}} \tag{F10}$$

Equations (F9) and (F10) are then iterated. This will converge to the supersonic solution if it exists.

Since the procedure depends on whether W_m is subsonic or supersonic, W_m is checked to determine which procedure to use. Actually, we want to know the value of W_m that corresponds to the maximum value of ρW_m. This occurs when $d(\rho W_m)/dW_m = 0$. By differentiating

$$\rho W_m = \rho_i' \left[1 - \frac{W_m^2 + W_\theta^2 + 2\omega\lambda - (\omega r)^2}{2c_p T_i'}\right]^{1/(\gamma-1)} (1 - \text{Ploss}) W_m$$

we obtain

$$\frac{d(\rho W_m)}{dW_m} = \rho(1 - \text{Ploss}) \left[1 - \frac{W_m^2}{(\gamma - 1)c_p T}\right] = 0$$

Hence,

$$\left(W_m^2\right)_{sonic} = \gamma RT \tag{F11}$$

as expected. But

$$T = T'' - \frac{W_m^2 + W_\theta^2}{2c_p}$$

Substitute this into equation (F11) and solve for W_m to obtain

$$(W_m)_{sonic} = \sqrt{\frac{2\gamma RT''}{\gamma + 1} - \left(\frac{\gamma - 1}{\gamma + 1}\right)W_\theta^2} \qquad \text{(F12)}$$

In subroutine LINDV, $(W_m)_{sonic}$ is calculated by equation (F12). If $(W_m)_{fs}$ is less than $(W_m)_{sonic}$, equations (F6) to (F8) are used to calculate ρ_{bf} iteratively. And if $(W_m)_{fs}$ is greater than $(W_m)_{sonic}$, equations (F9) and (F10) are used. In either case a solution may not exist because the blockage is enough so that $(\rho W_m)_{bf}$ is larger than $(\rho W_m)_{sonic}$. In this case an error message is printed and ρ_{bf} is set equal to ρ_{fs}. Calculation will proceed but may be inaccurate.

APPENDIX G

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 that 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 31. All that is necessary is to interpolate linearly within the quadrilateral. The interpolation 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 32. 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

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

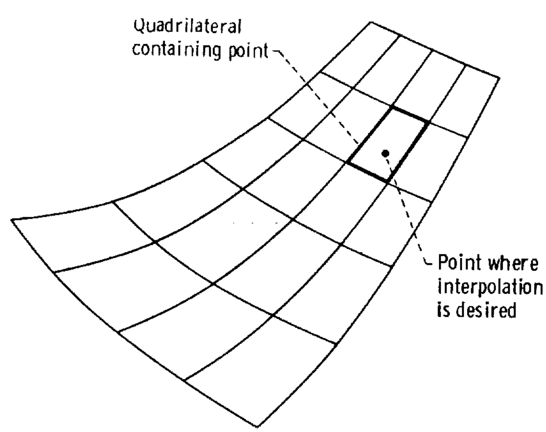


Figure 31. - Typical mesh region.

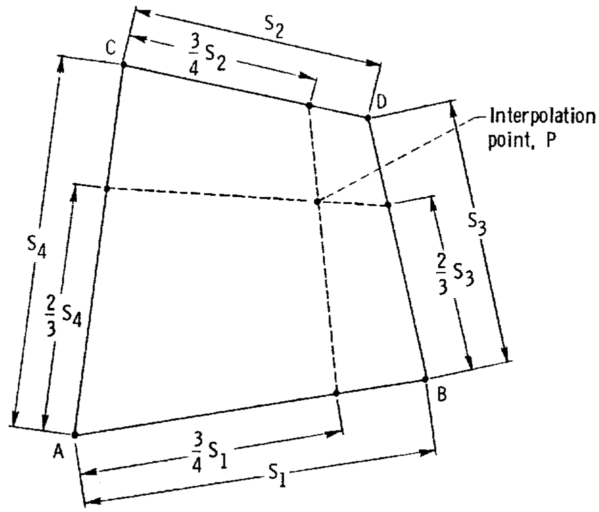


Figure 32. - Example of linear interpolation in a quadrilateral.

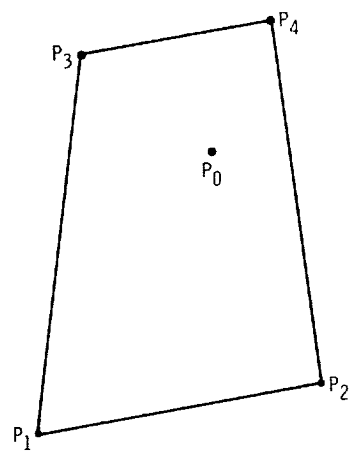


Figure 33. - Typical quadrilateral.

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

Figure 33 shows a quadrilateral containing a point P_0 where it is desired to interpolate. 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 these 14 values are known, an equation for linear interpolation can be derived.

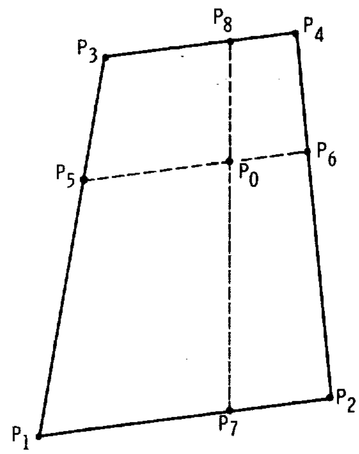


Figure 34. - Typical quadrilateral with interpolation lines.

Figure 34 shows the same quadilateral as figure 33 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

$$f_x = \frac{P_1P_7}{P_1P_2} \tag{G1}$$

$$f_y = \frac{P_1P_5}{P_1P_3} \tag{G2}$$

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,

$$f_y = \frac{x_{51}}{x_{31}} = \frac{y_{51}}{y_{31}} = \frac{x_{62}}{x_{42}} = \frac{y_{62}}{y_{42}} \tag{G3}$$

The equation of line P_5P_6 is

$$\frac{y - y_5}{x - x_5} = \frac{y_{65}}{x_{65}} \tag{G4}$$

By using equation (G3), y_5, y_6, x_5, and x_6 can be expressed in terms of f_y and the known values. For example,

$$y_5 = y_1 + y_{51} = y_1 - f_y y_{13}$$

In a similar manner, we obtain

$$\left.\begin{aligned} y_5 &= y_1 - f_y y_{13} \\ y_6 &= y_2 + f_y y_{42} \\ x_5 &= x_1 - f_y x_{13} \\ x_6 &= x_2 + f_y x_{42} \end{aligned}\right\} \qquad \text{(G5)}$$

By substituting equations (G5) into (G4), we obtain

$$\frac{y - y_1 + f_y y_{13}}{x - x_1 + f_y x_{13}} = \frac{y_2 + f_y y_{42} - y_1 + f_y y_{13}}{x_2 + f_y x_{42} - x_1 + f_y x_{13}} \qquad \text{(G6)}$$

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:

$$af_y^2 + bf_y + c = 0 \qquad \text{(G7)}$$

where

$$\left.\begin{aligned} a &= y_{13}x_{42} - x_{13}y_{42} \\ b &= x_{13}y_{02} - y_{13}x_{02} + y_{01}x_{42} - x_{01}y_{42} \\ c &= y_{01}x_{21} - x_{01}y_{21} \end{aligned}\right\} \qquad \text{(G8)}$$

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

$$af_x^2 + bf_x + c = 0 \tag{G9}$$

where

$$\left.\begin{aligned} a &= y_{12}x_{43} - x_{12}y_{43} \\ b &= x_{12}y_{03} - y_{12}x_{03} + y_{01}x_{43} - x_{01}y_{43} \\ c &= y_{01}x_{31} - x_{01}y_{31} \end{aligned}\right\} \tag{G10}$$

If $a \neq 0$ in equation (G7) or (G9), there are two solutions for f_x or f_y. However, there will be only one value between zero and 1. 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 1; 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 (G7) or (G9) can be written as

$$f = -\frac{b}{2a}\left(1 \pm \sqrt{1 - \frac{4ac}{b^2}}\right) \tag{G11}$$

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 binominal series, to obtain

$$\sqrt{1 - \frac{4ac}{b^2}} = 1 - \frac{2ac}{b^2} - \frac{2a^2c^2}{b^4} - \frac{4a^3c^3}{b^6} - \frac{10a^4c^4}{b^8} - \cdots \tag{G12}$$

for $|4ac| < b^2$. Substituting equation (G12) into equation (G11), with the minus sign, gives

$$f = -\frac{c}{b}\left(1 + \frac{ac}{b^2} + \frac{2a^2c^2}{b^4} + \frac{5a^3c^3}{b^6} + \ldots\right) \tag{G13}$$

Equation (G13) 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 (G13) is used whenever $|4ac| \leq b^2/100$. 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

$$z_0 = z_1(1 - f_x)(1 - f_y) + z_2f_x(1 - f_y) + z_3(1 - f_x)f_y + z_4f_xf_y \tag{G14}$$

APPENDIX H

SYMBOLS

$a_1, \ldots, a_4$	coefficients for finite-difference equations
B	tangential space between blades, rad
b	blade-to-blade streamsheet thickness
c_p	specific heat at constant pressure, J/(kg)(K)
F	vector normal to midchannel 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
k_0	constants for finite-difference equations
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. 4, part I), rad
β	angle between relative velocity vector and meridional plane (fig. 4, part I), rad
β_{abs}	angle between absolute velocity vector and meridional plane
γ	specific-heat ratio
ζ	coefficient in stream-function equation, defined in eq. (A3), meters/sec^2
θ	relative angular coordinate (fig. 4, 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. 4, 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
mid	midchannel
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. Katsanis, Theodore: FORTRAN Program for Calculating Transonic Velocities on a Blade-to-Blade Stream Surface of a Turbomachine. NASA TN D-5427, 1969.

4. 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.

5. 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. II - Programmer's Manual. NASA TN D-7344, 1974.

6. Katsanis, Theodore; and McNally, William D.: Revised FORTRAN Program for Calculating Velocities and Streamlines on the Hub-Shroud Midchannel Stream Surface of an Axial-, Radial-, or Mixed-Flow Turbomachine or Annular Duct. I - User's Manual. NASA TN D-8430, 1977.

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

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

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

10. Schumann, Lawrence F.: FORTRAN Program for Calculating Leading- and Trailing-Edge Geometry of Turbomachine Blades. NASA TM X-73679, 1977.

11. 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.

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

 *U.S. GOVERNMENT PRINTING OFFICE: 1977 - 735-004/15