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
# 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

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

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

# 15. Supplementary Notes

# 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))
Meridional plane; Turbomachine flow; Mid-channel stream surface; Axial-flow turbomachine; Mixed-flow turbomachine; Transonic flow; Radial-inflow turbine; Centrifugal compressor

# 18. Distribution Statement
Unclassified - unlimited
STAR Category 02
## CONTENTS

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

![Diagram of two-dimensional analysis surfaces in a turbomachine.](image)
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 satisfac-
tory for the original MERIDL program is still satisfactory for the revised MERIDL pro-
gram. 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 $21 \times 41$ 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 \( \beta \) and the tangential velocity \( W_\theta \) after the meridional velocity \( W_m \) has been calculated. Subroutine THIKOM calculates the tangential blade thickness \( t_\theta \) 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 MEPLLOT 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.
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 pre-
pare 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 \( \psi \) 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 \( \xi \) and \( \zeta \) (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 RVHTA for inlet and outlet whirl. Then TVELCY calculates $\partial W_m/\partial m$ and $\partial W_g/\partial m$ 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 flowchart.)
<table>
<thead>
<tr>
<th>Subroutine name or entry point</th>
<th>COMMON blocks (a)</th>
<th>Called subroutines</th>
<th>Calling subroutines</th>
<th>Subroutine name or entry point</th>
<th>COMMON blocks (a)</th>
<th>Called subroutines</th>
<th>Calling subroutines</th>
</tr>
</thead>
<tbody>
<tr>
<td>MERIDL</td>
<td>( /---/ )</td>
<td>( INPUT )</td>
<td>None - main program</td>
<td>LOSSTV (see entry point LOSSTV)</td>
<td>( /---/ )</td>
<td>( )</td>
<td>MERIDL</td>
</tr>
<tr>
<td></td>
<td></td>
<td>( MESH )</td>
<td></td>
<td></td>
<td></td>
<td>( )</td>
<td>TVELCY</td>
</tr>
<tr>
<td></td>
<td></td>
<td>( PRECAL )</td>
<td></td>
<td></td>
<td></td>
<td>( )</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>( INIT )</td>
<td></td>
<td></td>
<td></td>
<td>( )</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>( COEF )</td>
<td></td>
<td></td>
<td></td>
<td>( )</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>( SOR )</td>
<td>SYNTAX</td>
<td></td>
<td></td>
<td>( )</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>( LOSSOM )</td>
<td>SYNTAX</td>
<td></td>
<td></td>
<td>( )</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>( THETOM )</td>
<td>SYNTAX</td>
<td></td>
<td></td>
<td>( )</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>( THIKOM )</td>
<td>SYNTAX</td>
<td></td>
<td></td>
<td>( )</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>( INIT )</td>
<td>SYNTAX</td>
<td></td>
<td></td>
<td>( )</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>( COEF )</td>
<td>SYNTAX</td>
<td></td>
<td></td>
<td>( )</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>( SOR )</td>
<td>SYNTAX</td>
<td></td>
<td></td>
<td>( )</td>
<td></td>
</tr>
</tbody>
</table>

\(* /---/ denotes unlabeled COMMON block.*
<table>
<thead>
<tr>
<th>Subroutine name or entry point</th>
<th>COMMON blocks (a)</th>
<th>Called subroutines</th>
<th>Calling subroutines</th>
<th>Subroutine name or entry point</th>
<th>COMMON blocks (a)</th>
<th>Called subroutines</th>
<th>Calling subroutines</th>
</tr>
</thead>
<tbody>
<tr>
<td>TSONIN</td>
<td>----/</td>
<td>LAMDAF</td>
<td>MERIDL</td>
<td>----/</td>
<td>None</td>
<td>LO6SOM</td>
<td>TSONIN (LINDV)</td>
</tr>
<tr>
<td></td>
<td>/INPUTT/</td>
<td>RVHTHA</td>
<td></td>
<td>/INPUTT/</td>
<td>None</td>
<td>LO6SOM</td>
<td>TVELCY (LOSSOM)</td>
</tr>
<tr>
<td></td>
<td>/CALCOM/</td>
<td>TIPF</td>
<td></td>
<td>/CALCOM/</td>
<td>None</td>
<td>LO6SOM</td>
<td>TVELCY (LOSSOM)</td>
</tr>
<tr>
<td></td>
<td>/VARCOM/</td>
<td>RHOFIF</td>
<td></td>
<td>/VARCOM/</td>
<td>None</td>
<td>LO6SOM</td>
<td>TVELCY (LOSSOM)</td>
</tr>
<tr>
<td></td>
<td>/ROTATN/</td>
<td>LAMNIT (LAMDAF)</td>
<td></td>
<td>/ROTATN/</td>
<td>None</td>
<td>LO6SOM</td>
<td>TVELCY (LOSSOM)</td>
</tr>
<tr>
<td></td>
<td>/SLCOM/</td>
<td>RVHTHA</td>
<td></td>
<td>/SLCOM/</td>
<td>None</td>
<td>LO6SOM</td>
<td>TVELCY (LOSSOM)</td>
</tr>
<tr>
<td></td>
<td>/INDCOM/</td>
<td>SPLINE</td>
<td></td>
<td>/INDCOM/</td>
<td>None</td>
<td>LO6SOM</td>
<td>TVELCY (LOSSOM)</td>
</tr>
<tr>
<td>TVELCY</td>
<td>----/</td>
<td>LOSSTV (LOSSOM)</td>
<td>MERIDL</td>
<td>----/</td>
<td>None</td>
<td>LO6SOM</td>
<td>TVELCY (LOSSOM)</td>
</tr>
<tr>
<td></td>
<td>/INPUTT/</td>
<td>TINDV (LINDV)</td>
<td></td>
<td>/INPUTT/</td>
<td>None</td>
<td>LO6SOM</td>
<td>TVELCY (LOSSOM)</td>
</tr>
<tr>
<td></td>
<td>/CALCOM/</td>
<td>LAMDAF</td>
<td></td>
<td>/CALCOM/</td>
<td>None</td>
<td>LO6SOM</td>
<td>TVELCY (LOSSOM)</td>
</tr>
<tr>
<td></td>
<td>/VARCOM/</td>
<td>RVHTHA</td>
<td></td>
<td>/VARCOM/</td>
<td>None</td>
<td>LO6SOM</td>
<td>TVELCY (LOSSOM)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>TIPF</td>
<td></td>
<td></td>
<td>None</td>
<td>LO6SOM</td>
<td>TVELCY (LOSSOM)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>RHOFIF</td>
<td></td>
<td></td>
<td>None</td>
<td>LO6SOM</td>
<td>TVELCY (LOSSOM)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>LAMNIT (LAMDAF)</td>
<td></td>
<td></td>
<td>None</td>
<td>LO6SOM</td>
<td>TVELCY (LOSSOM)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>RVHTTA</td>
<td></td>
<td></td>
<td>None</td>
<td>LO6SOM</td>
<td>TVELCY (LOSSOM)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(see entry RVHTTA)</td>
<td></td>
<td></td>
<td>None</td>
<td>LO6SOM</td>
<td>TVELCY (LOSSOM)</td>
</tr>
<tr>
<td>LINDV (entry point - TINDV; see also entry PINDV)</td>
<td>----/</td>
<td>LAMDAF</td>
<td>TVELCY</td>
<td></td>
<td>----/</td>
<td>None</td>
<td>LO6SOM</td>
</tr>
<tr>
<td></td>
<td>/INPUTT/</td>
<td>RVHTHA</td>
<td></td>
<td>/INPUTT/</td>
<td>None</td>
<td>LO6SOM</td>
<td>TVELCY (LOSSOM)</td>
</tr>
<tr>
<td></td>
<td>/CALCOM/</td>
<td>TIPF</td>
<td></td>
<td>/CALCOM/</td>
<td>None</td>
<td>LO6SOM</td>
<td>TVELCY (LOSSOM)</td>
</tr>
<tr>
<td></td>
<td>/VARCOM/</td>
<td>RHOFIF</td>
<td></td>
<td>/VARCOM/</td>
<td>None</td>
<td>LO6SOM</td>
<td>TVELCY (LOSSOM)</td>
</tr>
<tr>
<td></td>
<td>/ROTATN/</td>
<td>LINDT (LAMDAF)</td>
<td></td>
<td>/ROTATN/</td>
<td>None</td>
<td>LO6SOM</td>
<td>TVELCY (LOSSOM)</td>
</tr>
<tr>
<td></td>
<td>/SLCOM/</td>
<td>CONTIN</td>
<td></td>
<td>/SLCOM/</td>
<td>None</td>
<td>LO6SOM</td>
<td>TVELCY (LOSSOM)</td>
</tr>
<tr>
<td></td>
<td>/INDCOM/</td>
<td>SLOPES</td>
<td></td>
<td>/INDCOM/</td>
<td>None</td>
<td>LO6SOM</td>
<td>TVELCY (LOSSOM)</td>
</tr>
<tr>
<td>PINIV (entry point for LINDV)</td>
<td>----/</td>
<td>LAMDAF</td>
<td>MERIDL</td>
<td>----/</td>
<td>None</td>
<td>LO6SOM</td>
<td>TVELCY (LOSSOM)</td>
</tr>
<tr>
<td></td>
<td>/INPUTT/</td>
<td>RVHTHA</td>
<td></td>
<td>/INPUTT/</td>
<td>None</td>
<td>LO6SOM</td>
<td>TVELCY (LOSSOM)</td>
</tr>
<tr>
<td></td>
<td>/CALCOM/</td>
<td>TIPF</td>
<td></td>
<td>/CALCOM/</td>
<td>None</td>
<td>LO6SOM</td>
<td>TVELCY (LOSSOM)</td>
</tr>
<tr>
<td></td>
<td>/VARCOM/</td>
<td>RHOFIF</td>
<td></td>
<td>/VARCOM/</td>
<td>None</td>
<td>LO6SOM</td>
<td>TVELCY (LOSSOM)</td>
</tr>
<tr>
<td></td>
<td>/ROTATN/</td>
<td>TINDV (LINDV)</td>
<td></td>
<td>/ROTATN/</td>
<td>None</td>
<td>LO6SOM</td>
<td>TVELCY (LOSSOM)</td>
</tr>
<tr>
<td></td>
<td>/SLCOM/</td>
<td>MERIDL</td>
<td></td>
<td>/SLCOM/</td>
<td>None</td>
<td>LO6SOM</td>
<td>TVELCY (LOSSOM)</td>
</tr>
<tr>
<td></td>
<td>/INDCOM/</td>
<td>SLOPES</td>
<td></td>
<td>/INDCOM/</td>
<td>None</td>
<td>LO6SOM</td>
<td>TVELCY (LOSSOM)</td>
</tr>
<tr>
<td>LAMDAF (see entry LAMNIT)</td>
<td>----/</td>
<td>None</td>
<td>INIT (see entry RHINRT)</td>
<td>----/</td>
<td>None</td>
<td>LO6SOM</td>
<td>TVELCY (LOSSOM)</td>
</tr>
<tr>
<td></td>
<td>/INPUTT/</td>
<td></td>
<td></td>
<td>/INPUTT/</td>
<td>None</td>
<td>LO6SOM</td>
<td>TVELCY (LOSSOM)</td>
</tr>
<tr>
<td></td>
<td>/CALCOM/</td>
<td></td>
<td></td>
<td>/CALCOM/</td>
<td>None</td>
<td>LO6SOM</td>
<td>TVELCY (LOSSOM)</td>
</tr>
<tr>
<td></td>
<td>/VARCOM/</td>
<td></td>
<td></td>
<td>/VARCOM/</td>
<td>None</td>
<td>LO6SOM</td>
<td>TVELCY (LOSSOM)</td>
</tr>
<tr>
<td></td>
<td>/ROTATN/</td>
<td></td>
<td></td>
<td>/ROTATN/</td>
<td>None</td>
<td>LO6SOM</td>
<td>TVELCY (LOSSOM)</td>
</tr>
<tr>
<td>LAMNIT (entry point for LAMDAF)</td>
<td>----/</td>
<td>RHINRT (entry point for RHOFIF)</td>
<td>----/</td>
<td></td>
<td>/INPUTT/</td>
<td>None</td>
<td>LO6SOM</td>
</tr>
<tr>
<td></td>
<td>/INPUTT/</td>
<td></td>
<td></td>
<td>/INPUTT/</td>
<td>None</td>
<td>LO6SOM</td>
<td>TVELCY (LOSSOM)</td>
</tr>
<tr>
<td></td>
<td>/CALCOM/</td>
<td></td>
<td></td>
<td>/CALCOM/</td>
<td>None</td>
<td>LO6SOM</td>
<td>TVELCY (LOSSOM)</td>
</tr>
<tr>
<td></td>
<td>/VARCOM/</td>
<td></td>
<td></td>
<td>/VARCOM/</td>
<td>None</td>
<td>LO6SOM</td>
<td>TVELCY (LOSSOM)</td>
</tr>
<tr>
<td></td>
<td>/ROTATN/</td>
<td></td>
<td></td>
<td>/ROTATN/</td>
<td>None</td>
<td>LO6SOM</td>
<td>TVELCY (LOSSOM)</td>
</tr>
<tr>
<td></td>
<td>/SLCOM/</td>
<td></td>
<td></td>
<td>/SLCOM/</td>
<td>None</td>
<td>LO6SOM</td>
<td>TVELCY (LOSSOM)</td>
</tr>
<tr>
<td></td>
<td>/INDCOM/</td>
<td></td>
<td></td>
<td>/INDCOM/</td>
<td>None</td>
<td>LO6SOM</td>
<td>TVELCY (LOSSOM)</td>
</tr>
</tbody>
</table>

*a/----/ denotes unlabeled COMMON block.
<table>
<thead>
<tr>
<th>Subroutine name or entry point</th>
<th>COMMON blocks (a)</th>
<th>Called subroutines</th>
<th>Calling subroutines</th>
<th>Subroutine name or entry point</th>
<th>COMMON block (a)</th>
<th>Called subroutines</th>
<th>Calling subroutines</th>
</tr>
</thead>
<tbody>
<tr>
<td>LININT</td>
<td>/---/</td>
<td>None</td>
<td>THEROM</td>
<td>SPLINT (entry point for SPLINT)</td>
<td>/---/</td>
<td>None</td>
<td>INPUT</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>THIKOM</td>
<td></td>
<td></td>
<td></td>
<td>MESHJO</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>OUTPUT</td>
<td></td>
<td></td>
<td></td>
<td>LOSSOM</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>TOUTPT</td>
<td></td>
<td></td>
<td></td>
<td>LOSSTV (LOSSOM)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>INDEV</td>
<td></td>
<td></td>
<td></td>
<td>ILETE</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>TSONIN</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>LINDV</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>TINDV</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>LAMNIT</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>RVTNIT</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ROTATE</td>
<td>None</td>
<td>None</td>
<td>MESHO</td>
<td>SPLEL (---/)</td>
<td>None</td>
<td>None</td>
<td>INPUT</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>PRECAL</td>
<td></td>
<td></td>
<td></td>
<td>PRECAL</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>THEROM</td>
<td></td>
<td></td>
<td></td>
<td>TSONIN</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>OUTPUT</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>TOUTPT (OUTPUT)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>TSONIN</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SPLINE</td>
<td>/---/</td>
<td>None</td>
<td>INPUT</td>
<td>SPINSL (---/)</td>
<td>None</td>
<td>None</td>
<td>THETOM</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>PRECAL</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>NEWRHO</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>TSONIN</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>LAMNIT (LAMDAF)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>RVTNIT (RVTHTA)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>TIPNIT (TIPF)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>RHINIT (RHOIPF)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SPLINT (see entry SPLENT)</td>
<td>/---/</td>
<td>None</td>
<td>INPUT</td>
<td>/INPUT/</td>
<td>Not applicable</td>
<td>MERIDL</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>PRECAL</td>
<td>/CAL/CON/</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>THEROM</td>
<td>/ROTATN/</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>THIKOM</td>
<td>/SLOM/</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>LOSSOM</td>
<td>/PLTCON/</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>LOSSTV (LOSSOM)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>OUTPUT</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>TOUTPT (OUTPUT)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>TSONIN</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>ILETE</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>LAMNIT (LAMDAF)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>RVTNIT (RVTHTA)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>INRECT</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

* /---/ 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 x 41 grid of 861 points. As dimensioned for a 100 x 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
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 \( W_{SUBZ} \) for \( W_z \). Velocity components along streamlines end in \( SL \) (\( W_{ZSL} \)), while velocity components on station lines end in \( ST \) (\( W_{ZST} \)). 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 \( \theta \)-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 \( T''_I/T'_I \), 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

![Diagram of Rotated z- and r-coordinates](image-url)

*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

<table>
<thead>
<tr>
<th>(a) Unrotated variables</th>
<th>(b) Rotated variables</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Read in unrotated and never rotated</strong></td>
<td><strong>Read in unrotated and then rotated for internal use</strong></td>
</tr>
<tr>
<td>ZBL(NPPP, NBLPL)</td>
<td>ZHST(NOSTAT)</td>
</tr>
<tr>
<td>ZHUB(NHUB)</td>
<td>RHST(NOSTAT)</td>
</tr>
<tr>
<td>ZTIP(NTIP)</td>
<td>ZTST(NOSTAT)</td>
</tr>
<tr>
<td>ZOMIN</td>
<td>ZTST(NOSTAT)</td>
</tr>
<tr>
<td>ZOMBI</td>
<td>ZTST(NOSTAT)</td>
</tr>
<tr>
<td>ZOMBO</td>
<td>ZTST(NOSTAT)</td>
</tr>
<tr>
<td>ZOMUX</td>
<td>ZTST(NOSTAT)</td>
</tr>
<tr>
<td>ZHIN</td>
<td>ZTST(NOSTAT)</td>
</tr>
<tr>
<td>ZTIN</td>
<td>ZTST(NOSTAT)</td>
</tr>
<tr>
<td>ZHOUT</td>
<td>ZTST(NOSTAT)</td>
</tr>
<tr>
<td>ZTOUT</td>
<td>ZTST(NOSTAT)</td>
</tr>
<tr>
<td><strong>Calculated unrotated and never rotated</strong></td>
<td><strong>Calculated rotated and never unrotated</strong></td>
</tr>
<tr>
<td>ZOM(MM, MHTP1)</td>
<td>ZOMROT(MM, MHTP1)</td>
</tr>
<tr>
<td>ZEOM(MHTP1)</td>
<td>ZOMROT(MM, MHTP1)</td>
</tr>
<tr>
<td>ZTEOM(MHTP1)</td>
<td>ZOMROT(MM, MHTP1)</td>
</tr>
<tr>
<td><strong>Calculated rotated and only unrotated for printing and plotting</strong></td>
<td><strong>Calculated rotated and only unrotated for printing and plotting</strong></td>
</tr>
<tr>
<td>ZSL(MM, NSL)</td>
<td>ZSL(MM, NSL)</td>
</tr>
<tr>
<td>ZST(NSL, NOSTAT)</td>
<td>ZST(NSL, NOSTAT)</td>
</tr>
<tr>
<td>ZPC(NPPC, NBLPL)</td>
<td>ZPC(NPPC, NBLPL)</td>
</tr>
</tbody>
</table>

17
<table>
<thead>
<tr>
<th>Subroutine</th>
<th>Variables read in or calculated</th>
<th>Rotated or unrotated</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>INPUT</td>
<td>ZBL, RBL</td>
<td>Unrotated</td>
<td>Read in unrotated</td>
</tr>
<tr>
<td></td>
<td>ZHUB, RHUB</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>ZTIP, RTIP</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>ZOMIN, ROMIN</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>ZOMBI, ROMBI</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>ZOMBO, ROMBO</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>ZOMOUT, ROMOUT</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>ZHIN, RHIN</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>ZTIN, RTIN</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>ZHOUT, RHOUT</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>ZTOUT, RTTOUT</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>ZHST, RHST</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>ZTST, RTST</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>RADIN</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>RADOUT</td>
<td></td>
<td></td>
</tr>
<tr>
<td>MESHIO</td>
<td>ZHROT, RHROT</td>
<td>Rotated</td>
<td>Rotated from ZHUB, RHUB</td>
</tr>
<tr>
<td></td>
<td>ZTROT, RTROT</td>
<td>Rotated</td>
<td>Rotated from ZTIP, RTIP</td>
</tr>
<tr>
<td></td>
<td>ZOMROT, ROMROT</td>
<td>Rotated</td>
<td>Calculated</td>
</tr>
<tr>
<td></td>
<td>ZOM, ROM</td>
<td>Unrotated</td>
<td>Unrotated from ZOMROT, ROMROT</td>
</tr>
<tr>
<td>PRECAL</td>
<td>ZHST, RHST</td>
<td>Rotated</td>
<td>Rotated from input values with same name</td>
</tr>
<tr>
<td></td>
<td>ZTST, RTST</td>
<td>Rotated</td>
<td>Rotated from input values with same name</td>
</tr>
<tr>
<td></td>
<td>ZBLROT, RBLROT</td>
<td>Rotated</td>
<td>Rotated from ZBL, RBL</td>
</tr>
<tr>
<td></td>
<td>ZLE, RLE</td>
<td></td>
<td>Calculated</td>
</tr>
<tr>
<td></td>
<td>ZTE, RTE</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>ZLEH, RTEH</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>ZLET, RLET</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>ZTEH, RTEH</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>ZTET, RTET</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>ZLEOMR, RLEOMR</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>ZTEOMR, RTEOMR</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>ZLEOM, RLEOM</td>
<td>Unrotated</td>
<td>Unrotated from ZLEOMR, RLEOMR</td>
</tr>
<tr>
<td></td>
<td>ZTEOM, RTEOM</td>
<td>Unrotated</td>
<td>Unrotated from ZTEOMR, RTEOMR</td>
</tr>
<tr>
<td>THERMOM</td>
<td>ZPC, RPC</td>
<td>Rotated</td>
<td>Calculated</td>
</tr>
<tr>
<td></td>
<td>ZPCT1, RPCT1</td>
<td></td>
<td>Calculated on semi-alternate mesh (fig. 26)</td>
</tr>
<tr>
<td></td>
<td>ZPCT2, RPCT2</td>
<td></td>
<td>Calculated on alternate mesh (fig. 27)</td>
</tr>
<tr>
<td></td>
<td>ZRAD, RRAD</td>
<td></td>
<td>Calculated</td>
</tr>
<tr>
<td>OUTPUT</td>
<td>ZSL, RSL</td>
<td>Rotated</td>
<td>Calculated rotated, unrotated for printing, then rotated for later use</td>
</tr>
<tr>
<td></td>
<td>ZST, RST</td>
<td>Rotated</td>
<td>Calculated rotated, unrotated for printing, and left unrotated</td>
</tr>
<tr>
<td>TSONIN</td>
<td>ZLES, RLES</td>
<td>Rotated</td>
<td>Calculated</td>
</tr>
<tr>
<td></td>
<td>ZTESL, RTESL</td>
<td>Rotated</td>
<td>Calculated</td>
</tr>
</tbody>
</table>
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 $\xi$ and $\zeta$) 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
I state
Input hub and tip coordinates
Divide hub and tip contours to obtain ZRAD, RRAD on hub and tip
Calculate ZRAD and RRAD arrays from hub to tip
Rotate mesh boundary intersection points on hub
Calculate r- and r-coordinates and slopes of mesh along hub contour (ZOMROT, ROMROT, and SLOM)
Begin construction of vertical orthogonals, proceed row by row j = 2

Begin construction of vertical orthogonal lines between two streamwise orthogonal rows, calculate r- and r-coordinates of straight lines normal to previous row
Call INRCT to calculate z- and r-intersections of lines normal to previous row with present row
Call SPLNT to calculate slopes of present horizontal orthogonal row at intersection points
Calculate z- and r-coordinates of intersections on present orthogonal row of normal from present row back to mesh point on previous orthogonal row
Calculate average z-coordinates of two sets of intersection points on present horizontal orthogonal row (ZOMROT)
Calculate r-coordinates (ROMROT) slopes (SLOM) and angles (PHI) at the ZOMROT mesh points
Go to next mesh line

j > MAXPRI ?
No
Yes
Calculate unrotated z and r meshpoint coordinates (ZOM, ROM) from rotated coordinates (ZOMROT, ROMROT)
Return

Figure 6. - Flow chart for MESH0.
Figure 7. - "Horizontal" orthogonals obtained by spline curve fitting.

Figure 8. - Process for generating "vertical" orthogonal links.
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, RVHTA, 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 MESH0 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.
Initialize subroutines LAMDAF, RVTHIA, TIPF, and RHQDFP by calling alternate entry points LAMNIT, RVNTN, TIPNT, and RHNIT.

Initialize BTH array to pitch

Are output streamline locations input in FLFR?

Yes

Initialize II streamlines in FLFR

No

Set FLFR and points = 0, 0, 1.0

Calculate s- and t-coordinates of orthogonal mesh (SOM and ROM) from ZOM and ROM

Rotate input hub and shroud station-line z- and r-coordinates if they were given as input

Rotate input blade-section z- and r-coordinates (ZBL and RBL)

Are FLFR and points = 0, 0, 1.0?

Yes

No

Calculate rotated and unrotated z- and r-coordinates, and also s- and t-coordinates, where orthogonal mesh crosses blade edges, calculate ILE and ITE arrays

Calculate arc length along input blade sections, blade tangential blockage, and slope and second derivative of blade mean camber line, print this, with input blade geometry data

Calculate stream-channel thickness arrays at blade edges (BTHLE and BTHTE), calculate BTH array within blade

REDTAC < 1.0?

Yes

Reduce \( \omega, W, V_0, \Delta \) for reduced-mass-flow solution

No

Reinitialize subroutines LAMDAF and RVTHIA for reduced-mass-flow solution

Prime debug output if requested

If flow is incompressible, initialize density array

Return

Store dummy values into ILE and ITE arrays

Store from rotated input blade-section arrays (ZBLROT and RBLROT) into leading- and trailing-edge arrays (ZLE, RLE and ZTE, RTE)

Call INRSCT to calculate intersections of blade edges with hub and shroud

Calculate rotated and unrotated z- and r-coordinates, and also s- and t-coordinates, where orthogonal mesh crosses blade edges, calculate ILE and ITE arrays

Calculate arc length along input blade sections, blade tangential blockage, and slope and second derivative of blade mean camber line, print this, with input blade geometry data

Calculate stream-channel thickness arrays at blade edges (BTHLE and BTHTE), calculate BTH array within blade

REDTAC < 1.0?

Yes

Reduce \( \omega, W, V_0, \Delta \) for reduced-mass-flow solution

No

Reinitialize subroutines LAMDAF and RVTHIA for reduced-mass-flow solution

Prime debug output if requested

If flow is incompressible, initialize density array

Return

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 $\theta$-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 re-initialized 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 $\theta$-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 $a_1$ and the constants $k_0$ will be printed.
Subroutine SOR

Subroutine SOR solves the finite-difference equations (A5) by the method of over-relaxation (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 $\Omega$ 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 $\text{ORF} = 2/(1 + \sqrt{1 - \text{LMAX}})$. The theory for calculating ORF is derived in reference 8.

With an optimum value for the overrelaxation factor $\Omega$, 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

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,

\[ \text{Loss} = 1 - \frac{p''_o}{(p''_o)_{\text{ideal}}} \]

In one input option, loss is given directly; in the other option, \( p'_i, T'_i, \lambda, (rV_o)_o \), and \( p'_o \) are given and the loss is calculated from Euler's equation by using the relations

\[ T'_o = T'_i - \frac{\omega\left[ \lambda - (rV_o)_o \right]}{c_p} \]  

(1)

and

\[ \frac{p''_o}{(p''_o)_{\text{ideal}}} = \frac{p'_o}{p'_i} = \frac{p'_i}{p'_o_{\text{ideal}}} = \frac{p'_o (T'_i)^{\gamma/(\gamma-1)}}{p'_i (T'_o)^{\gamma/(\gamma-1)}} \]

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 $\xi$ and $\zeta$ 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 $W_s$, $W_\theta$, $V_\theta$, $W$, $\rho$, $\xi$, and $\zeta$ 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
Calculate $W_s$, $W_t$, $V_b$, $p$, $z$, and $\zeta$.

Converged? (RELER < VELTOL)

Yes

$\text{IENT} = \text{IENT} + 1$

Set print controls to print all requested output

Return

Figure 12. - Flow chart for NEWRHO.

Further calculations. Equation (G10) of part I is used to calculate $W_s$. Within the blade row, $W_\theta$ is calculated from $W_s$, $W_t$, $\partial \theta / \partial s$, and $\partial \theta / \partial t$. Since

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

$$\tan \beta = r \frac{\partial \theta}{\partial m} = r \frac{\partial \theta}{\partial s} \frac{\partial s}{\partial m} + r \frac{\partial \theta}{\partial t} \frac{\partial t}{\partial m}$$

$$\frac{\partial s}{\partial m} = \frac{W_s}{W_m}$$
\[ \frac{dt}{dm} = \frac{W_t}{W_m} \]

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,

\[
W_\theta = \begin{cases} 
\frac{\Lambda - \omega r}{r} & \text{upstream of blade} \\
(rV_\theta) & \text{downstream of blade} \\
\frac{\omega r}{r} & \text{} 
\end{cases}
\]

Then \( V_\theta \) 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

\[ p'' = \rho_1 T_1' \left( \frac{T''}{T_1'} \right)^{\gamma/(\gamma - 1)} \left( 1 - \frac{p''_{\text{ideal}} - p''}{p''_{\text{ideal}}} \right) \quad (2) \]

where

\[ \frac{T''}{T_1'} = 1 - \frac{2\omega \lambda - (\omega r)^2}{2c_p T_1'} \quad (3) \]

Equation (3) is the same as equation (D5) of part I with \( W = 0 \). The density \( \rho \) is calculated by
\[ \rho = \rho_0 \left( \frac{T}{T_1} \right)^{\frac{1}{\gamma - 1}} \left( 1 - \frac{\Pi'' - \Pi'''}{p_{\text{ideal}}'} \right) \]

where \( T/T_1 \) 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 \( \xi \) and \( \zeta \) 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 $W_{\theta}$ 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 $\beta$ upstream and downstream of the blade. The angle $\alpha$ is assumed to be the same as in the final subsonic iteration. With $W$, $\beta$, and $\alpha$ 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

\[ \frac{1}{r_c} = \frac{dx}{dm} = \frac{d\alpha}{ds} \cos(\alpha - \varphi) + \frac{d\alpha}{dt} \sin(\alpha - \varphi) \]

Then the critical velocity ratio is obtained from

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

\[ \frac{W}{W_{cr}} = \frac{W}{\sqrt{2\gamma R T'' t}} \]

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 \( W_z, W_r, W_\theta, W/W_{cr} \), and \( 1/r_c \). By using variations of the preceding formulas, \( W_m, \alpha, \beta, \) and \( W \) are calculated from these values. Subroutine ILETÉ 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 \( \beta \) are extrapolated and interpolated in this manner, and thus \( W_m \) and \( W_\theta \) 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 \( W_\theta \), and then \( W_m \) and \( \beta \) 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 \( \alpha \) and \( W \) are calculated by using the equations given previously. LININT is then called to interpolate \( W_\ell \) and \( W_{tr} \) for station lines that lie within the blade. Finally, the remaining station-line output \( V_\theta, V, \beta_{abs}, T'', W/W_{cr}, p'', T', p', T, \rho, \) and \( p \) is calculated at each point. The station-line output is then printed.

The final small section of OUTPUT then restores \( W_\ell \) 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, \([d(rV_\theta)/dm]B \cos \beta\) is calculated, and \( W_\ell \) and \( W_{tr} \) are calculated by equation (G4) of part I (ref. 6). From this, \( \rho_\ell \) and \( \rho_{tr} \) are calculated by equations (D4) and (D5) of part I. The average density \( \rho_{av} \) is calculated by Simpson's rule

\[
\rho_{av} = \frac{\rho_\ell + 4\rho_{mid} + \rho_{tr}}{6}
\]

This quantity is used in NEWRHO in the next iteration. Then, the predicted value of \( F_t \) is calculated by
\[ F_t = \frac{W}{B} \frac{3\theta}{2t} \text{DFDM} \]  \hspace{1cm} (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 \( F_{\text{NEW}} \), 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.](image-url)
Subroutine INDEV

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 $\beta_{bf}$ is calculated from equation (F1) of part I. The corrections to $\partial \theta / \partial s$ are made so that the difference between $\beta_{bf}$ and $\beta_b$ 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 $\beta_{bf}$ (BETALI) 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](image-url)
Subroutine TSONIN

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 $T'_1$, $\rho'_1$, $\lambda$, and $(rV_0)_o$ 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

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

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

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

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 $\theta$-coordinates for TSONIC is at the center of the leading-edge radius. Since the leading-edge radius is not known at the outset, $\theta$-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 $\theta$ 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 $\theta$-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 \( \theta \)-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 \( W_\theta \), 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 \( \text{INCR} = 1 \).

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 \( \beta \) for a short distance beyond the leading edge, as described in appendix F of part I. After all leading-edge corrections to \( \beta \) 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 \( \beta \). The program then proceeds upstream until a solution has been obtained for every vertical mesh line.
Start

Set \( JZ = 1 \) (subsonic solution)

\( ISUPER = 2 \) ?

Yes

No

Yes

Restore full mass flow values and relinitialize LAMBDA and RVHEA

Calculate \( d\Omega/d\alpha \) and \( d\Omega/d\alpha \)

Initialize variables for outer loop \( (I = 0; \text{INCR} = 1) \)

\( I = I + \text{INCR} \)

\( I > MM \) ?

Yes

No

First estimate for \( \eta HUB \); initialize variables for inner loop

\( J = 1 \)

Start inner DO loop to calculate \( a, b, \) and \( c \) for velocity-gradient equation

\( \text{INCR} < 0 \) ?

Yes

No

Yes

No

Correct \( B \) at leading edge for all affected mesh points on present horizontal mesh line

\( J = J + 1 \)

Yes

No

Yes

No

Correct \( B \) at trailing edge for all affected mesh points on present horizontal mesh line

Figure 17 - Flow chart for 'VELCY.'
Has correct $\beta$ been calculated for all horizontal mesh lines at leading edge?

- Yes: INCR = 3
- No: Calculate $e$ for velocity-gradient equation, calculate arrays for $\sqrt{\gamma - \omega r^2}$, $X_P^2 \cos (\omega - \phi B)

Calculate $e$ and $f$ for velocity-gradient equation

Solve velocity-gradient equation; calculate mass flow

Integrated mass flow converged?
- No: Call CONTIN to get WRUHB
- Yes: Proceeding OK?
- No (choked): Check change in stream function
- Yes: Update $P_{loss}$, $T_{eq}$, $p_t$, $\lambda$, and $\bar{H}_{e0}$

Is outer iteration unconverged and less than 1000 total number of inner iterations?
- No: Choked?
- Yes: Print "CHOKED" message

No over 1000 iterations?
- Yes: Terminate calculations for mesh line and set indicator for mesh-line output only (limit streamline and station-line output)
- No: 230

Figure 17. - Continued.
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 $\beta$ 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 $\beta$. 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 \( 2\omega \lambda - (\omega r)^2, 2c_\rho T_j \), 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

\[
\begin{align*}
W_{j+1}^* &= W_j + (dW)_j \quad \text{first estimate for } W_{j+1} \\
W_{j+1}^{**} &= W_j + (dW)_{j+1}^* \quad \text{second estimate for } W_{j+1} \\
W_{j+1} &= \frac{W_{j+1}^* + W_{j+1}^{**}}{2} \quad \text{average of two estimates for } W_{j+1}
\end{align*}
\]

(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 \((dW)_{j+1}^*\) 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

\[ w = \int_{0}^{\text{tip}} \rho W \cos \beta \cos(\alpha - \varphi)rB \, dt \]

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 \( \text{REPEAT} = \text{.TRUE.} \)). Then the
PLOSS array is updated by calling LOSSTV, and arrays of $T_1$, $\rho_1$, $\lambda$, 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 $\text{INCR} = 1$, 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 $\beta$ 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 $\beta$ 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 $\text{LSFR} = 1$), the stream function is first estimated and later iterated to be adjusted to the correct stream-function value. These adjustments to the stream function ($\text{SFIN}$ and $\text{SFOUT}$) are done in LAMDAF and RVTHTA.

The input argument for all these subroutines is $\text{SF}$, 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

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

Figure 19. - Starting procedure for CONTIN.
or the maximum point is within the range of the three points obtained.

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

51
JZ determines whether subsonic or supersonic solution will be obtained:

\[ JZ = 1, \text{ subsonic solution} \]
\[ JZ = 2, \text{ supersonic solution} \]

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 sup-
plied 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

\[ \text{CALL INRSCT}(XCURV1, YCURV1, N1, XCURV2, YCURV2, N2, XCROSS, YCROSS) \]

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 $N_X$, and $I$ denotes the number of the line (running from 1 at the left to $N_X$ at the right). The number of horizontal lines is $N_Y$, and $J$ denotes the number of the line (running from 1 at the bottom to $N_Y$ to the top). The lines between mesh points are assumed to be straight lines.
Figure 22 - Flow chart 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 \quad f_x
FY \quad f_y
F_1 \quad \frac{(-b - \sqrt{b^2 - 4ac})}{2a}
F_2 \quad \frac{(-b + \sqrt{b^2 - 4ac})}{2a}
IJEX \quad \text{indicator, first or second pass through coding to calculate } f_x \text{ or } f_y
IN \quad \text{new value for } I
JN \quad \text{new value for } J
QA \quad a \ (\text{eq. (G8) or (G10)})
QB \quad b \ (\text{eq. (G8) or (G10)})
QC \quad c \ (\text{eq. (G8) or (G10)})
RIGHT \quad \text{integer, 1 indicates that } X_0, Y_0 \text{ is to the right of the current } I, J \text{ region,}
\quad \text{0 within, and } -1 \text{ left}
X_{01} \quad x_{01} \ (\text{see appendix G for notation})
X_{02} \quad x_{02} \text{ or } x_{03}
X_{13} \quad x_{13} \text{ or } x_{12}
X_{21} \quad x_{21} \text{ or } x_{31}
X_{42} \quad x_{42} \text{ or } x_{43}
Y_{01} \quad y_{01}
Y_{02} \quad y_{02} \text{ or } y_{03}
Y_{13} \quad y_{13} \text{ or } y_{12}
Y_{21} \quad y_{21} \text{ or } y_{31}
Y_{42} \quad y_{42} \text{ 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 RHOIFP, 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.

<table>
<thead>
<tr>
<th>Variable name</th>
<th>COMMON block</th>
<th>Subroutine</th>
<th>Description and comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>A(4, MM, MHTP1)</td>
<td>VARCOM</td>
<td>A0</td>
<td>Coefficients of finite-difference equation (A7) for stream function, u</td>
</tr>
<tr>
<td>AAA(MM or MHTP1, etc.)</td>
<td></td>
<td>COEF</td>
<td>a0 (eq. (A5))</td>
</tr>
<tr>
<td>ALPHA(MM, MHTP1)</td>
<td>VARCOM</td>
<td>MESH, THIK, NEWRHO, OUTPUT, TSONIN, LAMDAF, RVTHTA</td>
<td>( \alpha ) at orthogonal mesh points, rad</td>
</tr>
<tr>
<td>ALPHLE</td>
<td>INDEV, LINDV</td>
<td></td>
<td>( \alpha_{le} ), rad</td>
</tr>
<tr>
<td>ALPHSP(MM)</td>
<td>TSONIN</td>
<td></td>
<td>( \alpha ) at TSONIC input points</td>
</tr>
<tr>
<td>ALPHTE</td>
<td>INDEV, LINDV</td>
<td></td>
<td>( \alpha_{te} ), rad</td>
</tr>
<tr>
<td>ALPSL(MM, NSL)</td>
<td>SLCOM</td>
<td></td>
<td>( \alpha ) at points along streamlines where they cross vertical mesh lines, rad</td>
</tr>
</tbody>
</table>

62
<table>
<thead>
<tr>
<th>Variable name</th>
<th>COMMON block</th>
<th>Subroutine</th>
<th>Description and comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>ALPST(NSL)</td>
<td>OUTPUT</td>
<td></td>
<td>$\alpha$ at points along station lines where they cross streamlines, rad</td>
</tr>
<tr>
<td>ALVERT(MHTP1)</td>
<td>OUTPUT</td>
<td></td>
<td>Temporary storage for values of $\alpha$ from ALPHA array on vertical mesh lines, rad</td>
</tr>
<tr>
<td>ANG(NPPP)</td>
<td>INPUT</td>
<td></td>
<td>Angles from meridional plane of blade-section mean camber lines at blade-section input points, rad</td>
</tr>
<tr>
<td>ANGR(NPPC, NBLPC)</td>
<td>TETOM</td>
<td></td>
<td>Angles with respect to radius of hub-shroud lines of alternate mesh (fig. 28), rad</td>
</tr>
<tr>
<td>ANGROT</td>
<td>INPUTT</td>
<td></td>
<td>Input angle of rotation, deg</td>
</tr>
<tr>
<td>ANGT1(NBLPL)</td>
<td>TETOM</td>
<td></td>
<td>Values from ANGZ array along constant-percent-chord line, rad</td>
</tr>
<tr>
<td>ANGT2(NBLPC)</td>
<td>TETOM</td>
<td></td>
<td>Values for ANGZ array along constant-percent-chord line, rad</td>
</tr>
<tr>
<td>ANGZ(NPPC, NBLPC)</td>
<td>TETOM</td>
<td></td>
<td>Angles with respect to z-axis of input blade sections (fig. 28), rad</td>
</tr>
<tr>
<td>AR</td>
<td>INPUTT</td>
<td></td>
<td>Input gas constant, $R$, J/(kg)(K)</td>
</tr>
<tr>
<td>ARTEM</td>
<td>TSONIN</td>
<td></td>
<td>Temporary AR, J/(kg)(K)</td>
</tr>
<tr>
<td>ATVEL(MHTP1)</td>
<td>TVELCY</td>
<td></td>
<td>Coefficients, $a$, of velocity-gradient equation ((A7), part I) at mesh points along vertical mesh lines</td>
</tr>
<tr>
<td>BBB(MM or MHTP1, etc.)</td>
<td>MESHO, THIKOM, OUTPUT TSONIN LAMDAF RVHTA</td>
<td>Dummy array used in SPLINE and SPLINT calls</td>
<td></td>
</tr>
<tr>
<td>BEABST(NSL)</td>
<td>OUTPUT</td>
<td></td>
<td>$\beta_{abs}$ at points where station lines cross streamlines, deg</td>
</tr>
<tr>
<td>Variable name</td>
<td>COMMON block</td>
<td>Subroutine</td>
<td>Description and comments</td>
</tr>
<tr>
<td>-----------------------</td>
<td>--------------</td>
<td>------------</td>
<td>--------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>BESP (MM)</td>
<td></td>
<td>TSONIN</td>
<td>Input for TSONIC (ref. 3), m</td>
</tr>
<tr>
<td>BETA(MM, MHTP1)</td>
<td>VARCOM</td>
<td></td>
<td>$\beta$ at orthogonal mesh points, rad</td>
</tr>
<tr>
<td>BETABF</td>
<td>INDEV, LINDV</td>
<td></td>
<td>$\beta_{bf}$ within blade, rad</td>
</tr>
<tr>
<td>BETAFS</td>
<td>INDEV, LINDV</td>
<td></td>
<td>$\beta_{fs}$ outside of blade, rad</td>
</tr>
<tr>
<td>BETAIJ</td>
<td>OUTPUT, INDEV</td>
<td></td>
<td>$\beta$ at orthogonal mesh point, rad</td>
</tr>
<tr>
<td>BETALE (NBLPL)</td>
<td>INPUTT</td>
<td></td>
<td>Input blade angles at leading edge, deg</td>
</tr>
<tr>
<td>BETATE (NBLPL)</td>
<td>INPUTT</td>
<td></td>
<td>Input blade angles at trailing edge, deg</td>
</tr>
<tr>
<td>BETI1</td>
<td>TSONIN</td>
<td></td>
<td>Input for TSONIC (ref. 3), deg</td>
</tr>
<tr>
<td>BETI2</td>
<td>TSONIN</td>
<td></td>
<td>Input for TSONIC (ref. 3), deg</td>
</tr>
<tr>
<td>BETO1</td>
<td>TSONIN</td>
<td></td>
<td>Input for TSONIC (ref. 3), deg</td>
</tr>
<tr>
<td>BETO2</td>
<td>TSONIN</td>
<td></td>
<td>Input for TSONIC (ref. 3), deg</td>
</tr>
<tr>
<td>BETSL (MM, NSL)</td>
<td>SLCOM</td>
<td></td>
<td>$\beta$ at points along streamlines where they cross vertical mesh lines, rad</td>
</tr>
<tr>
<td>BETST (NSL)</td>
<td>OUTPUT</td>
<td></td>
<td>$\beta$ at points along station lines where they cross streamlines, rad</td>
</tr>
<tr>
<td>BFACTR</td>
<td>TSONIN</td>
<td></td>
<td>Multiplying factor for BESP and ZMSFL</td>
</tr>
<tr>
<td>BLDCRD</td>
<td>INDEV, LINDV</td>
<td></td>
<td>True blade chord along a horizontal mesh line, m</td>
</tr>
<tr>
<td>BLDEV (MHTP1)</td>
<td>LINDV</td>
<td></td>
<td>Deviation angle, corrected for blockage, where a horizontal mesh line intersects trailing edge, $(\beta_{bf} - \beta_{te})$, deg</td>
</tr>
<tr>
<td>BLDEV</td>
<td>INDEV</td>
<td></td>
<td>Deviation angle, corrected for blockage, where a horizontal mesh line</td>
</tr>
<tr>
<td>Variable name</td>
<td>COMMON block</td>
<td>Subroutine</td>
<td>Description and comments</td>
</tr>
<tr>
<td>---------------</td>
<td>--------------</td>
<td>------------</td>
<td>-------------------------</td>
</tr>
<tr>
<td>BLINC(MHTP1)</td>
<td></td>
<td>LINDV</td>
<td>intersects trailing edge, $(\beta_{bf} - \beta_{b_{te}})$, deg</td>
</tr>
<tr>
<td>BLINC</td>
<td></td>
<td>INDEV</td>
<td>intersects leading edge, $(\beta_{bf} - \beta_{b_{le}})$, deg</td>
</tr>
<tr>
<td>BLNK</td>
<td></td>
<td>OUTPUT</td>
<td>Blank word used in some plot titles</td>
</tr>
<tr>
<td>BLOCK(NPPP)</td>
<td></td>
<td>PRECAL</td>
<td>Fractional blockage along input blade sections</td>
</tr>
<tr>
<td>BTBFLE(MHTP1)</td>
<td>INDCOM</td>
<td></td>
<td>$\beta_{bf}$ where a horizontal mesh line intersects leading edge, rad</td>
</tr>
<tr>
<td>BTBFTE(MHTP1)</td>
<td>INDCOM</td>
<td></td>
<td>$\beta_{bf}$ where a horizontal mesh line intersects trailing edge, rad</td>
</tr>
<tr>
<td>BTBLLE(MHTP1)</td>
<td></td>
<td>LINDV</td>
<td>$\beta_{b_{le}}$ where a horizontal mesh line intersects leading edge, rad</td>
</tr>
<tr>
<td>BTBLTE(MHTP1)</td>
<td></td>
<td>LINDV</td>
<td>$\beta_{b_{le}}$ where a horizontal mesh line intersects trailing edge, rad</td>
</tr>
<tr>
<td>BTFSEX(MHTP1)</td>
<td></td>
<td>OUTPUT</td>
<td>$\beta_{fs}$ extrapolated to leading or trailing edge of blade, rad</td>
</tr>
<tr>
<td>BTH(MM, MHTP1)</td>
<td></td>
<td>CALCON</td>
<td>$B$ at orthogonal mesh points, rad</td>
</tr>
<tr>
<td>BTHLE(MHTP1)</td>
<td>INDCOM</td>
<td></td>
<td>$B_{le}$, rad</td>
</tr>
<tr>
<td>BTHSL</td>
<td></td>
<td>TSONIN</td>
<td>$B$ along a streamline, rad</td>
</tr>
<tr>
<td>BTHTE(MHTP1)</td>
<td>INDCOM</td>
<td></td>
<td>$B_{te}$, rad</td>
</tr>
<tr>
<td>BTVEL(MHTP1)</td>
<td></td>
<td>TVELCY</td>
<td>Coefficients, $b$, of velocity-gradient equation ((A7), part I) at mesh points along vertical mesh lines</td>
</tr>
<tr>
<td>C1</td>
<td></td>
<td>COEF</td>
<td>$c_1$ (eq. (A8))</td>
</tr>
<tr>
<td>Variable name</td>
<td>COMMON block</td>
<td>Subroutine</td>
<td>Description and comments</td>
</tr>
<tr>
<td>---------------------</td>
<td>--------------</td>
<td>------------</td>
<td>---------------------------------------------------------------</td>
</tr>
<tr>
<td>C2</td>
<td></td>
<td>COEF</td>
<td>$c_2$ (eq. (A8))</td>
</tr>
<tr>
<td>CAMP(MM, MHTP1)</td>
<td>VARCOM</td>
<td>TVELCY</td>
<td>$\cos(\alpha - \varphi)$ at orthogonal mesh points</td>
</tr>
<tr>
<td>CBETA</td>
<td></td>
<td>SOR</td>
<td>$\cos \beta$</td>
</tr>
<tr>
<td>CHANGE</td>
<td></td>
<td>TVELCY</td>
<td>Change in value of stream function at a mesh point during an over-relaxation iteration</td>
</tr>
<tr>
<td>CHFL</td>
<td></td>
<td>TVELCY</td>
<td>Choking mass flow for a vertical orthogonal mesh line, kg/sec</td>
</tr>
<tr>
<td>CHFRMS</td>
<td></td>
<td>TVELCY</td>
<td>Ratio of minimum choking mass flow to input mass flow</td>
</tr>
<tr>
<td>CHLIM</td>
<td></td>
<td>TVELCY</td>
<td>Minimum choking mass flow per passage, kg/sec</td>
</tr>
<tr>
<td>CHORD</td>
<td></td>
<td>TSONIN</td>
<td>Length of blade section along streamline (m-direction) (input to TSONIC, ref. 3), m</td>
</tr>
<tr>
<td>COSAB</td>
<td></td>
<td>THETOM</td>
<td>$\cos \alpha$ of ANGZ + ANGR</td>
</tr>
<tr>
<td>CP</td>
<td>CALCON</td>
<td></td>
<td>$c_p$, J/(kg)(K)</td>
</tr>
<tr>
<td>CPHI(MM, MHTP1)</td>
<td>CALCON</td>
<td></td>
<td>$\cos \varphi$ at orthogonal mesh points</td>
</tr>
<tr>
<td>CPTIP(MHTP1)</td>
<td></td>
<td>TVELCY</td>
<td>$2c_p T'_1$, along vertical mesh lines, (N)(m)/kg</td>
</tr>
<tr>
<td>CPTIP</td>
<td></td>
<td>LINDV</td>
<td>$2c_p T'_1$, (N)(m)/kg</td>
</tr>
<tr>
<td>CTVEL(MHTP1)</td>
<td></td>
<td>TVELCY</td>
<td>Coefficients, $c$, of velocity-gradient equation ((A7), part I) at mesh points along vertical mesh lines</td>
</tr>
<tr>
<td>CURV(MM, MHTP1)</td>
<td>VARCOM</td>
<td>TSONIN</td>
<td>$1/r_c$ at orthogonal mesh points, 1/m</td>
</tr>
<tr>
<td>CURV1(MM)</td>
<td></td>
<td>TSONIN</td>
<td>Curvature of upper blade surface at TSONIC input points, 1/m</td>
</tr>
<tr>
<td>CURV2(MM)</td>
<td></td>
<td>TSONIN</td>
<td>Curvature of lower blade surface at TSONIC input points, 1/m</td>
</tr>
<tr>
<td>Variable name</td>
<td>COMMON block</td>
<td>Subroutine</td>
<td>Description and comments</td>
</tr>
<tr>
<td>---------------------</td>
<td>--------------</td>
<td>------------</td>
<td>------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>CURVSL(MM, NSL)</td>
<td>SLCOM</td>
<td></td>
<td>$1/r_c$ at points along streamlines where they cross vertical orthogonal mesh lines, 1/m</td>
</tr>
<tr>
<td>CURVST(NSL)</td>
<td>OUTPUT</td>
<td></td>
<td>$1/r_c$ at points along station lines where they cross streamlines, 1/m</td>
</tr>
<tr>
<td>D1</td>
<td>COEF</td>
<td></td>
<td>$d_1$ (eq. (A8))</td>
</tr>
<tr>
<td>D2</td>
<td>COEF</td>
<td></td>
<td>$d_2$ (eq. (A8))</td>
</tr>
<tr>
<td>D2BDM2(MM)</td>
<td>TSONIN</td>
<td></td>
<td>$d^2B/dm^2$, 1/m</td>
</tr>
<tr>
<td>D2TDM1(MM)</td>
<td>TSONIN</td>
<td></td>
<td>Second derivative of upper blade surface at TSONIC input points, rad/m^2</td>
</tr>
<tr>
<td>D2TDM2(MM)</td>
<td>TSONIN</td>
<td></td>
<td>Second derivative of lower blade surface at TSONIC input points, rad/m^2</td>
</tr>
<tr>
<td>D2YDX2(NPPP or MHTP1)</td>
<td>PRECAL</td>
<td></td>
<td>Second derivative of $\theta$ along input blade sections, rad/m^2</td>
</tr>
<tr>
<td>D2YDX2(NPPC or NBLPC)</td>
<td>THETOM</td>
<td></td>
<td>Dummy second derivative in SPLINT calls</td>
</tr>
<tr>
<td>DALDS(MM)</td>
<td>OUTPUT</td>
<td></td>
<td>$\partial\alpha/\partial s$ at mesh points along horizontal mesh lines, rad/m</td>
</tr>
<tr>
<td>DALDT(MM, MHTP1)</td>
<td>OUTPUT</td>
<td></td>
<td>$\partial\alpha/\partial t$ at orthogonal mesh points, rad/m</td>
</tr>
<tr>
<td>DALVER(MHTP1)</td>
<td>OUTPUT</td>
<td></td>
<td>$\partial\alpha/\partial t$ at mesh points along vertical mesh lines, rad/m</td>
</tr>
<tr>
<td>DAMP</td>
<td>TSONIN</td>
<td></td>
<td>Damping factor on iteration for leading- or trailing-edge radii (appendix E)</td>
</tr>
<tr>
<td>DBDM(MM)</td>
<td>TSONIN</td>
<td></td>
<td>$dB/dm$</td>
</tr>
<tr>
<td>DBL</td>
<td>TSONIN</td>
<td></td>
<td>One-half of tangential blade thickness (in rad) at intersection of a streamline with blade leading or trailing edge</td>
</tr>
<tr>
<td>Variable name</td>
<td>COMMON block</td>
<td>Subroutine</td>
<td>Description and comments</td>
</tr>
<tr>
<td>------------------------</td>
<td>--------------</td>
<td>------------</td>
<td>------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>DBTH</td>
<td></td>
<td>THIKOM</td>
<td>Tangential blade thickness at orthogonal mesh point, rad</td>
</tr>
<tr>
<td>DCHANG</td>
<td></td>
<td>COEF</td>
<td>Maximum change in estimated values of $\partial (r\dot{V}_\theta)/\partial t$ at a mesh point between any two outer iterations, m/sec</td>
</tr>
<tr>
<td>DEGRAD</td>
<td>OUTPUT</td>
<td>TSONIN</td>
<td>Conversion constant from radians to degrees</td>
</tr>
<tr>
<td></td>
<td></td>
<td>INDEV</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>LINDV</td>
<td></td>
</tr>
<tr>
<td>DELCH</td>
<td>OUTPUT</td>
<td></td>
<td>1 percent of average meridional chord length of blade, m</td>
</tr>
<tr>
<td>DELM</td>
<td>TSONIN</td>
<td></td>
<td>Increment of meridional distance, m</td>
</tr>
<tr>
<td>DELMAX</td>
<td>TVELCY</td>
<td></td>
<td>Maximum increment for $W_{hub}$ at each iteration to satisfy continuity, m/sec</td>
</tr>
<tr>
<td>DELMSP</td>
<td>TSONIN</td>
<td></td>
<td>Minimum distance of blade surface points from leading or trailing edge (appendix E), m</td>
</tr>
<tr>
<td>DELR</td>
<td>MESHO</td>
<td></td>
<td>Increment in $r$-coordinate, m</td>
</tr>
<tr>
<td></td>
<td>THETOM</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>OUTPUT</td>
<td></td>
<td></td>
</tr>
<tr>
<td>DELRHO(MM, MHTP1)</td>
<td>VARCOM</td>
<td></td>
<td>Difference in density, between suction and pressure surfaces, at orthogonal mesh points, kg/m$^3$</td>
</tr>
<tr>
<td>DELT</td>
<td>MESHO</td>
<td></td>
<td>Tangential blade thickness, m</td>
</tr>
<tr>
<td>DELTH</td>
<td>TSONIN</td>
<td></td>
<td>Shift in $\theta$ origin from MERIDL to TSONIC, rad</td>
</tr>
<tr>
<td>DELZ</td>
<td>MESHO</td>
<td></td>
<td>Increment in $z$-coordinate, m</td>
</tr>
<tr>
<td></td>
<td>THETOM</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>OUTPUT</td>
<td></td>
<td></td>
</tr>
<tr>
<td>DFDM(MM, MHTP1)</td>
<td>VARCOM</td>
<td></td>
<td>$-B \cos \beta \left[\partial (r\dot{V}_\theta)/\partial m\right]$ at orthogonal mesh points (eq. (4)), m/sec</td>
</tr>
<tr>
<td>Variable name</td>
<td>COMMON block</td>
<td>Subroutine</td>
<td>Description and comments</td>
</tr>
<tr>
<td>---------------</td>
<td>--------------</td>
<td>------------</td>
<td>--------------------------</td>
</tr>
<tr>
<td>DFDS(MM)</td>
<td></td>
<td>BLDVEL</td>
<td>$\partial (rV_{\theta})/\partial s$ at mesh points along horizontal mesh lines, m/sec</td>
</tr>
<tr>
<td>DFDT(MM, MHTP1)</td>
<td></td>
<td>BLDVEL</td>
<td>$\partial (rV_{\theta})/\partial t$ at orthogonal mesh points, m/sec</td>
</tr>
<tr>
<td>DFVERT(MHTP1)</td>
<td></td>
<td>BLDVEL</td>
<td>$\partial (rV_{\theta})/\partial t$ at mesh points along vertical mesh lines, m/sec</td>
</tr>
<tr>
<td>DIP(NBLPL)</td>
<td></td>
<td>PRECAL</td>
<td>Distance up leading or trailing edge of blade, m</td>
</tr>
<tr>
<td>DISEOM(MHTP1)</td>
<td></td>
<td>THIKOM</td>
<td>Distance up leading or trailing edge of blade, m</td>
</tr>
<tr>
<td>DIST(NPPP)</td>
<td></td>
<td>INPUT</td>
<td>Distances on meridional plane along lines connecting input blade-section points (ZBL, RBL), m</td>
</tr>
<tr>
<td>DIST(NBLPL)</td>
<td></td>
<td>THIKOM</td>
<td>Distances on meridional plane along lines connecting input blade-section points (ZBL, RBL), m</td>
</tr>
<tr>
<td>DISTLE</td>
<td></td>
<td>INDEV</td>
<td>Distance along horizontal mesh line from leading edge of blade for which a blade shape correction is made for incidence, m</td>
</tr>
<tr>
<td>DISTTE</td>
<td></td>
<td>INDEV</td>
<td>Distance along horizontal mesh line from trailing edge of blade for which a blade shape correction is made for deviation, m</td>
</tr>
<tr>
<td>DLAM</td>
<td></td>
<td>TVELCY</td>
<td>Change in $rV_{\theta}$ between points on vertical mesh lines, m$^2$/sec</td>
</tr>
<tr>
<td>DLDU(MM, MHTP1)</td>
<td></td>
<td>VARCOM</td>
<td>Gradients of $rV_{\theta}$ 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.)</td>
</tr>
<tr>
<td>DMAX</td>
<td></td>
<td>COEF</td>
<td>Maximum calculated value of $\partial (rV_{\theta})/\partial t$ at any mesh point, m/sec</td>
</tr>
<tr>
<td>Variable name</td>
<td>COMMON block</td>
<td>Subroutine</td>
<td>Description and comments</td>
</tr>
<tr>
<td>---------------</td>
<td>--------------</td>
<td>------------</td>
<td>--------------------------</td>
</tr>
<tr>
<td>DMIN</td>
<td></td>
<td>COEF</td>
<td>Minimum calculated value of ( \frac{\partial (rV_\phi)}{\partial t} ) at any mesh point, m/sec</td>
</tr>
<tr>
<td>DNEW</td>
<td>INPUTT</td>
<td></td>
<td>Input damping factor on calculation of ( \frac{\partial (rV_\phi)}{\partial t} ) within blade row from outer iteration to outer iteration</td>
</tr>
<tr>
<td>DOM(MHTP1)</td>
<td>PRECAL</td>
<td></td>
<td>Distance up leading or trailing edge of blade, m</td>
</tr>
<tr>
<td>DPDT(MHTP1)</td>
<td>NEWRHO</td>
<td></td>
<td>( \frac{\partial p''}{\partial t} ) at mesh points along vertical mesh lines, N/m³</td>
</tr>
<tr>
<td>DP REL</td>
<td>TVELCY</td>
<td></td>
<td>Change in ( p'' ) between points on vertical mesh lines, N/m²</td>
</tr>
<tr>
<td>DRHOSL</td>
<td>TSONIN</td>
<td></td>
<td>Difference in density between suction and pressure surfaces along a streamline, kg/m³</td>
</tr>
<tr>
<td>DTAN1</td>
<td>TSONIN</td>
<td></td>
<td>( \frac{\partial \theta}{\partial m} ) at upper blade surface leading- or trailing-edge tangency point, rad/m</td>
</tr>
<tr>
<td>DTAN2</td>
<td>TSONIN</td>
<td></td>
<td>( \frac{\partial \theta}{\partial m} ) at lower blade surface leading- or trailing-edge tangency point, rad/m</td>
</tr>
<tr>
<td>DTDM1(MM)</td>
<td>TSONIN</td>
<td></td>
<td>( \frac{\partial \theta}{\partial m} ) on upper blade surface at TSONIC input points, rad/m</td>
</tr>
<tr>
<td>DTDM2(MM)</td>
<td>TSONIN</td>
<td></td>
<td>( \frac{\partial \theta}{\partial m} ) on lower blade surface at TSONIC input points, rad/m</td>
</tr>
<tr>
<td>DTDRLE</td>
<td>INDEV</td>
<td></td>
<td>( \frac{\partial \theta}{\partial r} )le, rad/m</td>
</tr>
<tr>
<td>DTDROM</td>
<td>THETOM</td>
<td></td>
<td>( \frac{\partial \theta}{\partial r} ) on orthogonal mesh, rad/m</td>
</tr>
<tr>
<td>DTDRTE</td>
<td>INDEV</td>
<td></td>
<td>( \frac{\partial \theta}{\partial r} )te, rad/m</td>
</tr>
<tr>
<td>DTDS(NPPP)</td>
<td>INPUT</td>
<td></td>
<td>( \frac{\partial \theta}{\partial s} ), rad/m</td>
</tr>
<tr>
<td>DTDT(MHTP1)</td>
<td>NEWRHO</td>
<td></td>
<td>( \frac{\partial T''}{\partial t} ) along vertical mesh lines, K/m</td>
</tr>
<tr>
<td>Variable name</td>
<td>COMMON block</td>
<td>Subroutine</td>
<td>Description and comments</td>
</tr>
<tr>
<td>---------------------</td>
<td>--------------</td>
<td>------------</td>
<td>---------------------------------------------------------------</td>
</tr>
<tr>
<td>DTDZLE</td>
<td>INDEV</td>
<td></td>
<td>$(\partial \theta/\partial z)_{le}$, rad/m</td>
</tr>
<tr>
<td>DTDZOM</td>
<td>LINDV</td>
<td></td>
<td>$\partial \theta/\partial z$ on orthogonal mesh, rad/m</td>
</tr>
<tr>
<td>DTDZTE</td>
<td>INDEV,</td>
<td></td>
<td>$(\partial \theta/\partial z)_{te}$, rad/m</td>
</tr>
<tr>
<td>DTHDR(NPPC, NBLPC)</td>
<td>INDCOM</td>
<td></td>
<td>$\partial \theta/\partial r$ on alternate blade mesh</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>(fig. 27), rad/m</td>
</tr>
<tr>
<td>DTHDS(MM, MHTP1)</td>
<td>CALCON</td>
<td></td>
<td>$\partial \theta/\partial s$ at orthogonal mesh points,</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>rad/m</td>
</tr>
<tr>
<td>DTHDSL</td>
<td>INPUT</td>
<td>PRECAL</td>
<td>$d\theta/ds$ at blade leading edge, rad/m</td>
</tr>
<tr>
<td></td>
<td></td>
<td>THETOM</td>
<td></td>
</tr>
<tr>
<td>DTHDSP(NPPC, NBLPC)</td>
<td>THETOM</td>
<td></td>
<td>$\partial \theta/\partial s'$ on alternate blade mesh</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>(fig. 27), rad/m</td>
</tr>
<tr>
<td>DTHDST</td>
<td>INPUT</td>
<td>PRECAL</td>
<td>$d\theta/ds$ at blade trailing edge, rad/m</td>
</tr>
<tr>
<td></td>
<td></td>
<td>THETOM</td>
<td></td>
</tr>
<tr>
<td>DTHDT(MM, MHTP1)</td>
<td>CALCON</td>
<td></td>
<td>$\partial \theta/\partial t$ at orthogonal mesh points,</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>rad/m</td>
</tr>
<tr>
<td>DTHDTP(NPPC, NBLPC)</td>
<td>THETOM</td>
<td></td>
<td>$\partial \theta/\partial t'$ on alternate blade mesh</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>(fig. 27), rad/m</td>
</tr>
<tr>
<td>DTHDZ(NPPC, NBLPC)</td>
<td>INDCOM</td>
<td></td>
<td>$\partial \theta/\partial z$ on alternate blade mesh</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>(fig. 27), rad/m</td>
</tr>
<tr>
<td>DTHEOM(MHTP1)</td>
<td>THIKOM</td>
<td></td>
<td>Blade thickness along leading or trailing edge, rad</td>
</tr>
<tr>
<td>DTIP</td>
<td>TVELCY</td>
<td></td>
<td>Change in $T_1$ between points on vertical mesh lines, K</td>
</tr>
<tr>
<td>DTPP</td>
<td>TVELCY</td>
<td></td>
<td>Change in $T''$ between points on vertical mesh lines, K</td>
</tr>
<tr>
<td>DTST1(NBLPL)</td>
<td>THETOM</td>
<td></td>
<td>$d\theta/ds'$ at input blade planes along 5-percent-chord lines, rad/m</td>
</tr>
<tr>
<td>Variable name</td>
<td>COMMON block</td>
<td>Description and comments</td>
<td></td>
</tr>
<tr>
<td>--------------</td>
<td>--------------</td>
<td>-------------------------</td>
<td></td>
</tr>
<tr>
<td>DTST2(NBLPC)</td>
<td>THETOM</td>
<td>$\frac{d\phi}{ds}$ at alternate mesh points along 5-percent-chord lines, rad/m</td>
<td></td>
</tr>
<tr>
<td>DTVEL(MHTP1)</td>
<td>TVELCY</td>
<td>Coefficients, $d$, of velocity-gradient equation ((A7), part I) at mesh points along vertical mesh lines</td>
<td></td>
</tr>
<tr>
<td>DUDS(MM)</td>
<td>NEWRHO</td>
<td>$\frac{dw}{ds}$ along horizontal mesh lines, 1/m</td>
<td></td>
</tr>
<tr>
<td>DUDT(MHTP1)</td>
<td>NEWRHO</td>
<td>$\frac{dw}{dt}$ at mesh points along vertical mesh lines, 1/m</td>
<td></td>
</tr>
<tr>
<td>DVTEMP</td>
<td>COEF</td>
<td>Updated estimate of $\frac{d(rV_{\phi})}{dt}$ at a mesh point, m/sec</td>
<td></td>
</tr>
<tr>
<td>DVTHDT(MM, MHTP1)</td>
<td>COEF</td>
<td>$\frac{d(rV_{\phi})}{dt}$ at orthogonal mesh points, m/sec</td>
<td></td>
</tr>
<tr>
<td>DWMDM(MM, MHTP1)</td>
<td>TVELCY</td>
<td>$\frac{dW_m}{dm}$ at orthogonal mesh points, 1/sec</td>
<td></td>
</tr>
<tr>
<td>DWMDS(MM)</td>
<td>TVELCY</td>
<td>$\frac{dW_m}{ds}$ along horizontal mesh lines, 1/sec</td>
<td></td>
</tr>
<tr>
<td>DWMDT(MM, MHTP1)</td>
<td>TVELCY</td>
<td>$\frac{dW_m}{dt}$ at orthogonal mesh points, 1/sec</td>
<td></td>
</tr>
<tr>
<td>DWMVER(MHTP1)</td>
<td>TVELCY</td>
<td>$\frac{dW_m}{dt}$ along vertical mesh lines, 1/sec</td>
<td></td>
</tr>
<tr>
<td>DWTDM(MM, MHTP1)</td>
<td>TVELCY</td>
<td>$\frac{dW_0}{dm}$ at orthogonal mesh points, 1/sec</td>
<td></td>
</tr>
<tr>
<td>DWTDS(MM)</td>
<td>TVELCY</td>
<td>$\frac{dW_0}{ds}$ along horizontal mesh lines, 1/sec</td>
<td></td>
</tr>
<tr>
<td>DWTDT(MM, MHTP1)</td>
<td>TVELCY</td>
<td>$\frac{dW_0}{dt}$ at orthogonal mesh points, 1/sec</td>
<td></td>
</tr>
<tr>
<td>DWTVER(MHTP1)</td>
<td>TVELCY</td>
<td>$\frac{dW_0}{dt}$ at mesh points along vertical mesh lines, 1/sec</td>
<td></td>
</tr>
<tr>
<td>DYDX(MHTP1 or NPPP, etc.)</td>
<td>INPUT PRECAL THETOM</td>
<td>Temporary storage for derivatives of SPLINE or SPLINT calls</td>
<td></td>
</tr>
<tr>
<td>Variable name</td>
<td>COMMON block</td>
<td>Subroutine</td>
<td>Description and comments</td>
</tr>
<tr>
<td>---------------</td>
<td>--------------</td>
<td>------------</td>
<td>--------------------------</td>
</tr>
<tr>
<td>EM(NIN or NOUT)</td>
<td>TIPF RHOIPF LAMDAF RVHTA</td>
<td>TIPF</td>
<td>Second derivatives of spline-fit curves</td>
</tr>
<tr>
<td>ERR</td>
<td>NEWRHO</td>
<td>SOR</td>
<td>Relative change in W at a mesh point</td>
</tr>
<tr>
<td>ERROR</td>
<td></td>
<td></td>
<td>Maximum absolute value of change in u at any point for an overrelaxation iteration</td>
</tr>
<tr>
<td>ETVEL(MHTP1)</td>
<td>TVELCY</td>
<td></td>
<td>Coefficients, e, of velocity-gradient equation ((A7), part I) at mesh points along vertical mesh lines</td>
</tr>
<tr>
<td>EXFRAC</td>
<td>OUTPUT LINDV</td>
<td></td>
<td>Extrapolation fraction at leading and trailing edges</td>
</tr>
<tr>
<td>EXPON</td>
<td>CALCON</td>
<td></td>
<td>$1/(\gamma - 1)$</td>
</tr>
<tr>
<td>EXTRAP</td>
<td>INDEV</td>
<td></td>
<td>Distance along horizontal mesh line from blade leading or trailing edge to first mesh point outside of blade, m</td>
</tr>
<tr>
<td>FCHANG</td>
<td>BLDVEL</td>
<td></td>
<td>Maximum value of change in $F_t$ at any mesh point between any two outer iterations</td>
</tr>
<tr>
<td>FLFR(NSL)</td>
<td>INPUTT</td>
<td></td>
<td>Input values of stream function designating streamlines along which output is to be printed</td>
</tr>
<tr>
<td>FMAX</td>
<td>BLDVEL</td>
<td></td>
<td>Maximum new predicted value of $F_t$ at any mesh point during an outer iteration</td>
</tr>
<tr>
<td>FMIN</td>
<td>BLDVEL</td>
<td></td>
<td>Minimum new predicted value of $F_t$ at any mesh point during an outer iteration</td>
</tr>
<tr>
<td>FNEW</td>
<td>INPUTT</td>
<td></td>
<td>Input damping factor on calculation of $F_t$ from outer iteration to outer iteration</td>
</tr>
<tr>
<td>Variable name</td>
<td>COMMON block</td>
<td>Subroutine</td>
<td>Description and comments</td>
</tr>
<tr>
<td>---------------</td>
<td>--------------</td>
<td>------------</td>
<td>--------------------------</td>
</tr>
<tr>
<td>FST(MM, MHTP1)</td>
<td></td>
<td>BLDVEL</td>
<td>$rV_\theta$ at orthogonal mesh points, m$^2$/sec</td>
</tr>
<tr>
<td>FT(MM, MHTP1)</td>
<td>VARCOM</td>
<td></td>
<td>$F_t$ at orthogonal mesh points (eq. (A4)), m/sec$^2$</td>
</tr>
<tr>
<td>FTT</td>
<td>BLDVEL</td>
<td></td>
<td>Predicted value of $F_t$ at a mesh point</td>
</tr>
<tr>
<td>FTVEL(MHTP1)</td>
<td>TVELCY</td>
<td></td>
<td>Coefficients, $f$, of velocity-gradient equation ((A7), part I) at mesh points along vertical mesh lines</td>
</tr>
<tr>
<td>FVERT(MHTP1)</td>
<td>BLDVEL</td>
<td></td>
<td>Temporary storage for values of $rV_\theta$ from FST array on vertical mesh lines, m$^2$/sec</td>
</tr>
<tr>
<td>GAM</td>
<td>INPUTT</td>
<td></td>
<td>Input, $\gamma$</td>
</tr>
<tr>
<td>H1</td>
<td>COEF</td>
<td></td>
<td>$h_1$ (eq. (A8), fig. 24)</td>
</tr>
<tr>
<td>H2</td>
<td>COEF</td>
<td></td>
<td>$h_2$ (eq. (A8), fig. 24)</td>
</tr>
<tr>
<td>H3</td>
<td>COEF</td>
<td></td>
<td>$h_3$ (eq. (A8), fig. 24)</td>
</tr>
<tr>
<td>H4</td>
<td>COEF</td>
<td></td>
<td>$h_4$ (eq. (A8), fig. 24)</td>
</tr>
<tr>
<td>IARG</td>
<td>LINDV</td>
<td></td>
<td>Integer indicator for mesh line where deviation correction begins</td>
</tr>
<tr>
<td>ICARDS</td>
<td>TSONIN</td>
<td></td>
<td>Integer control on writing of TSONIC card image input to a file</td>
</tr>
<tr>
<td>ICOUNT</td>
<td>SOR</td>
<td></td>
<td>Integer internal iteration counter</td>
</tr>
<tr>
<td></td>
<td>NEWRHO</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>TSONIN</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>TVELCY</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>LINDV</td>
<td></td>
<td></td>
</tr>
<tr>
<td>IDEBUG</td>
<td>INPUTT</td>
<td></td>
<td>Integer input indicating multiple of outer iterations at which debug output is printed</td>
</tr>
<tr>
<td>IEND</td>
<td>Blank</td>
<td></td>
<td>Integer indicator of stage of solution to which program has proceeded:</td>
</tr>
</tbody>
</table>

74
<table>
<thead>
<tr>
<th>Variable name</th>
<th>COMMON block</th>
<th>Subroutine</th>
<th>Description and comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>ILE(MHTPI)</td>
<td>CALCON</td>
<td></td>
<td>Vertical mesh line numbers of first mesh point inside blade region at leading edge</td>
</tr>
<tr>
<td>ILOS</td>
<td>LOSSOM</td>
<td></td>
<td>Integer control on print indicating negative loss</td>
</tr>
<tr>
<td>ILS(NSL)</td>
<td>SLCOM</td>
<td></td>
<td>Vertical mesh line number of first intersection of a streamline with a vertical mesh line inside blade region at leading edge</td>
</tr>
<tr>
<td>IMAX</td>
<td>TVELCY</td>
<td></td>
<td>Integer indicator of final vertical mesh line where an incidence correction is made</td>
</tr>
<tr>
<td>IMESH</td>
<td>INPUTT</td>
<td></td>
<td>Integer input indicating the multiple of outer iterations at which major output is printed for orthogonal mesh</td>
</tr>
<tr>
<td>INCR</td>
<td>TVELCY</td>
<td></td>
<td>Integer increment for loop on vertical mesh lines</td>
</tr>
<tr>
<td>IND</td>
<td>TVELCY</td>
<td></td>
<td>Integer that indicates solution procedure in CONTIN</td>
</tr>
<tr>
<td>IPLOT</td>
<td>INPUTT</td>
<td></td>
<td>Integer input indicating multiple of outer iterations at which major output is plotted</td>
</tr>
<tr>
<td>Variable name</td>
<td>COMMON block</td>
<td>Subroutine</td>
<td>Description and comments</td>
</tr>
<tr>
<td>----------------</td>
<td>--------------</td>
<td>------------</td>
<td>--------------------------</td>
</tr>
<tr>
<td>IPRNT</td>
<td></td>
<td>INDEV</td>
<td>Integer indicator controlling output</td>
</tr>
<tr>
<td>IREVRS</td>
<td></td>
<td>TVELCY</td>
<td>Integer indicator at end of forward iteration</td>
</tr>
<tr>
<td>ISLINE</td>
<td>INPUTT</td>
<td></td>
<td>Integer input indicating multiple of outer iterations at which major output is printed along streamlines</td>
</tr>
<tr>
<td>ISTATL</td>
<td>INPUTT</td>
<td></td>
<td>Integer input indicating multiple of outer iterations at which major output is printed along station lines</td>
</tr>
<tr>
<td>ISUPER</td>
<td>INPUTT</td>
<td></td>
<td>Integer input indicating whether only subsonic, or both subsonic and supersonic, solutions of velocity-gradient equation are to be calculated</td>
</tr>
<tr>
<td>ITE(MHTPI)</td>
<td></td>
<td>CALCON</td>
<td>Vertical mesh line numbers of last mesh point inside blade region at trailing edge</td>
</tr>
<tr>
<td>ITEMPIN</td>
<td></td>
<td>TVELCY</td>
<td>Integer indicating smallest value of IARG</td>
</tr>
<tr>
<td>ITEMP</td>
<td></td>
<td>TVELCY</td>
<td>Temporary storage for IARG</td>
</tr>
<tr>
<td>ITER</td>
<td>Blank</td>
<td></td>
<td>Outer iteration counter, incremented by 1 at beginning of each outer iteration</td>
</tr>
<tr>
<td>ITS(NSL)</td>
<td></td>
<td>SLCOM</td>
<td>Vertical mesh line number of last intersection of a streamline with a vertical mesh line inside blade region at trailing edge</td>
</tr>
<tr>
<td>ITSON</td>
<td>INPUTT</td>
<td></td>
<td>Integer input indicating multiple of outer iterations at which information is printed as input for TSONIC program (ref. 3)</td>
</tr>
<tr>
<td>JARG</td>
<td></td>
<td>LINDV</td>
<td>Integer indicator for mesh line where incidence correction ends</td>
</tr>
</tbody>
</table>

76
<table>
<thead>
<tr>
<th>Variable name</th>
<th>COMMON block</th>
<th>Subroutine</th>
<th>Description and comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>JZ</td>
<td></td>
<td>TVELCY</td>
<td>Integer used to indicate to CONTIN that subsonic ( JZ = 1 ) or supersonic ( JZ = 2 ) solution is desired</td>
</tr>
<tr>
<td>K(MM, MHTP1)</td>
<td>VARCOM</td>
<td></td>
<td>( k_0 ) (eq. (A9)) at orthogonal mesh points</td>
</tr>
<tr>
<td>KNEW</td>
<td></td>
<td>COEF</td>
<td>Updated value of ( k_0 ) (eq. (A9)) at a mesh point</td>
</tr>
<tr>
<td>LAMBDA(MHTP1)</td>
<td></td>
<td>TVELCY</td>
<td>( \lambda ) for mesh points along vertical mesh lines, ( m^2/sec )</td>
</tr>
<tr>
<td>LAMBDO(MHTP1)</td>
<td></td>
<td>TVELCY</td>
<td>( (rV_\theta)_o ) for mesh points along vertical mesh lines, ( m^2/sec )</td>
</tr>
<tr>
<td>LAMBDO</td>
<td></td>
<td>NEWRHO</td>
<td>( (rV_\theta)_o ), ( m^2/sec )</td>
</tr>
<tr>
<td>LAMIN(NIN)</td>
<td>INPUTT</td>
<td></td>
<td>Input values of ( \lambda ) at points along line from hub to shroud on which upstream flow conditions are given, ( m^2/sec )</td>
</tr>
<tr>
<td>LAMOUT(NOUT)</td>
<td>INPUTT</td>
<td></td>
<td>Input values of ( (rV_\theta)_o ) at points along line from hub to shroud on which downstream flow conditions are given, ( m^2/sec )</td>
</tr>
<tr>
<td>LAMVT</td>
<td>INPUTT</td>
<td></td>
<td>Input integer (0 or 1) indicating whether upstream and downstream whirl (0) or tangential velocity (1) is given as input</td>
</tr>
<tr>
<td>LBLAD</td>
<td>INPUTT</td>
<td></td>
<td>Input integer (0, 1, or 2) indicating type of blade geometry input</td>
</tr>
<tr>
<td>LETEAN</td>
<td>INPUTT</td>
<td></td>
<td>Input integer (0 or 1) indicating whether leading- and trailing-edge blade angles are given</td>
</tr>
<tr>
<td>Variable name</td>
<td>COMMON block</td>
<td>Subroutine</td>
<td>Description and comments</td>
</tr>
<tr>
<td>---------------</td>
<td>--------------</td>
<td>------------</td>
<td>-------------------------</td>
</tr>
<tr>
<td>LINC</td>
<td>TVELCY, LINDV</td>
<td></td>
<td>Integer indicating completion of transonic incidence correction at leading edge</td>
</tr>
<tr>
<td>LIPS</td>
<td>TSONIN</td>
<td></td>
<td>Input for TSONIC (ref. 3)</td>
</tr>
<tr>
<td>LMAX</td>
<td>SOR</td>
<td></td>
<td>Maximum value of RATIO over all mesh points</td>
</tr>
<tr>
<td>LMIN</td>
<td>SOR</td>
<td></td>
<td>Minimum value of RATIO over all mesh points</td>
</tr>
<tr>
<td>LOSOUT(OUT)</td>
<td>INPUTT</td>
<td></td>
<td>Input fraction of absolute total pressure loss, at points along line from hub to shroud on which downstream flow conditions are given</td>
</tr>
<tr>
<td>LOSS</td>
<td>TSONIN</td>
<td></td>
<td>Input for TSONIC (ref. 3)</td>
</tr>
<tr>
<td>LROT</td>
<td>INPUTT</td>
<td></td>
<td>Input integer (0 or 1) indicating if rotation of coordinate system should be used</td>
</tr>
<tr>
<td>LRVB</td>
<td>TSONIN</td>
<td></td>
<td>Input for TSONIC (ref. 3)</td>
</tr>
<tr>
<td>LSFR</td>
<td>INPUTT</td>
<td></td>
<td>Input integer (0 or 1) indicating whether upstream and downstream flow conditions are input as a function of stream function (0) or radius (1)</td>
</tr>
<tr>
<td>LTPL</td>
<td>INPUTT</td>
<td></td>
<td>Input integer (0 or 1) indicating whether downstream total pressure (0) or fractional loss of stagnation pressure (1) is given in input</td>
</tr>
<tr>
<td>LWCR</td>
<td>TSONIN</td>
<td></td>
<td>Input for TSONIC (ref. 3)</td>
</tr>
<tr>
<td>MARK</td>
<td>OUTPUT</td>
<td></td>
<td>Integers between 1 and 4 indicating whether output station lines are outside blade, within blade, or on leading or trailing edge</td>
</tr>
<tr>
<td>MAXFLO</td>
<td>TVELCY</td>
<td></td>
<td>Maximum mass flow for which a solution can be obtained for a vertical mesh line, kg/sec</td>
</tr>
<tr>
<td>Variable name</td>
<td>COMMON block</td>
<td>Subroutine</td>
<td>Description and comments</td>
</tr>
<tr>
<td>---------------</td>
<td>--------------</td>
<td>------------</td>
<td>-------------------------</td>
</tr>
<tr>
<td>MBBI</td>
<td>TSONIN</td>
<td></td>
<td>Input for TSONIC (ref. 3)</td>
</tr>
<tr>
<td>MBI</td>
<td>INPUTT</td>
<td>TSONIN</td>
<td>Input number of vertical mesh lines from left boundary of orthogonal mesh (ZOMIN) to first point of mesh-size change (ZOMBI)</td>
</tr>
<tr>
<td>MBITS</td>
<td>TSONIN</td>
<td></td>
<td>Input for TSONIC (ref. 3)</td>
</tr>
<tr>
<td>MBO</td>
<td>INPUTT</td>
<td></td>
<td>Input total number of vertical mesh lines from left boundary of orthogonal mesh (ZOMIN) to point of second mesh-size change (ZOMBO)</td>
</tr>
<tr>
<td>MBOTS</td>
<td>TSONIN</td>
<td></td>
<td>Input for TSONIC (ref. 3)</td>
</tr>
<tr>
<td>MHT</td>
<td>INPUTT</td>
<td></td>
<td>Input total number of horizontal mesh spaces from hub to shroud of orthogonal mesh; maximum of 100 MHT + 1</td>
</tr>
<tr>
<td>MHTPl</td>
<td>CALCON</td>
<td></td>
<td>Minimum mass flow for which a solution can be obtained for a vertical mesh line, kg/sec</td>
</tr>
<tr>
<td>MINFO (real variable)</td>
<td>TVELCY</td>
<td></td>
<td>Input total number of vertical mesh lines from left to right boundaries of orthogonal mesh (ZOMIN to ZOMOUT), maximum of 100 MM - 1</td>
</tr>
<tr>
<td>MM</td>
<td>INPUTT</td>
<td></td>
<td>Input for TSONIC (ref. 3)</td>
</tr>
<tr>
<td>MMM1</td>
<td>CALCON</td>
<td></td>
<td>Input total mass flow through entire circumferential annulus of machine, kg/sec</td>
</tr>
<tr>
<td>MMTS</td>
<td>TSONIN</td>
<td></td>
<td>m-coordinates of points along streamlines where they cross vertical mesh lines, m (Origin of m-coordinate is upstream boundary of orthogonal mesh.)</td>
</tr>
<tr>
<td>Variable name</td>
<td>COMMON block</td>
<td>Subroutine</td>
<td>Description and comments</td>
</tr>
<tr>
<td>---------------</td>
<td>--------------</td>
<td>------------</td>
<td>-------------------------</td>
</tr>
<tr>
<td>MST(NSL) (real variable)</td>
<td>OUTPUT</td>
<td>m-coordinates of points where station lines cross streamlines, m (Origin of m-coordinates is upstream boundary of orthogonal mesh.)</td>
<td></td>
</tr>
<tr>
<td>NADD</td>
<td>TVELCY</td>
<td>Integer number of times WHUB is increased for restart of iteration procedure</td>
<td></td>
</tr>
<tr>
<td>NBL</td>
<td>INPUTT</td>
<td>Input number of blades in blade row</td>
<td></td>
</tr>
<tr>
<td>NBLPC</td>
<td>INDCOM</td>
<td>Integer number of blade planes in alternate mesh</td>
<td></td>
</tr>
<tr>
<td>NBLPL</td>
<td>INPUTT</td>
<td>Number of input blade planes or blade sections on which data (ZBL, RBL, TIBL, TNBL, etc.) are given to describe mean flow surface and blade thickness, maximum of 50</td>
<td></td>
</tr>
<tr>
<td>NBLPTS</td>
<td>TSONIN</td>
<td>Number of spline points on suction or pressure surface of a blade section</td>
<td></td>
</tr>
<tr>
<td>NCHOK</td>
<td>OUTPUT</td>
<td>Integer number of vertical orthogonal mesh lines that are choked in the transonic solution</td>
<td></td>
</tr>
<tr>
<td>NCOUNT</td>
<td>TVELCY</td>
<td>Total number of iterations or attempts at satisfying velocity-gradient equation</td>
<td></td>
</tr>
<tr>
<td>NHUB</td>
<td>INPUTT</td>
<td>Number of input data points in ZHUB and RHUB arrays, maximum of 50</td>
<td></td>
</tr>
<tr>
<td>NIN</td>
<td>INPUTT</td>
<td>Number of input data points in upstream arrays of flow properties (SFIN, RADIN, TIP, PRIP, LAMIN, and VTHIN), maximum of 50</td>
<td></td>
</tr>
<tr>
<td>NLOSS</td>
<td>INPUTT</td>
<td>Number of input data points in PERCRD and PERLOS arrays, maximum of 50</td>
<td></td>
</tr>
<tr>
<td>NMAX</td>
<td>MESHO</td>
<td>Maximum of either NHUB or NTIP</td>
<td></td>
</tr>
<tr>
<td>Variable name</td>
<td>COMMON block</td>
<td>Subroutine</td>
<td>Description and comments</td>
</tr>
<tr>
<td>---------------</td>
<td>--------------</td>
<td>------------</td>
<td>-------------------------</td>
</tr>
<tr>
<td>NOSTAT</td>
<td>INPUTT</td>
<td>TVELCY</td>
<td>Input number of hub-shroud stations (located by coordinates in ZHST, RHST, ZTST, and RTST) at which output is desired, maximum of 50</td>
</tr>
<tr>
<td>NOUT</td>
<td>INPUTT</td>
<td>TVELCY</td>
<td>Number of input data points in downstream arrays of flow properties (SFOUT, RADOUT, PROP, LOSOUT, LAMOUT, and VTHOUT), maximum of 50</td>
</tr>
<tr>
<td>NPPC</td>
<td>INDCOM</td>
<td>TSONIN</td>
<td>Integer number of vertical percent-chord lines in alternate mesh</td>
</tr>
<tr>
<td>NPPP</td>
<td>INPUTT</td>
<td>TVELCY</td>
<td>Number of input data points per blade section or blade plane in ZBL, RBL, etc., arrays; maximum of 50</td>
</tr>
<tr>
<td>NREAD</td>
<td>Blank</td>
<td>TVELCY</td>
<td>Integer number of input read file</td>
</tr>
<tr>
<td>NREP</td>
<td>TVELCY</td>
<td>TVELCY</td>
<td>Integer number of repeats on outer iteration loop</td>
</tr>
<tr>
<td>NRES</td>
<td>TVELCY</td>
<td>TVELCY</td>
<td>Integer number of restarts of inner iteration for a given vertical orthogonal mesh line</td>
</tr>
<tr>
<td>NRSP</td>
<td>TSONIN</td>
<td>TSONIN</td>
<td>Input for TSONIC (ref. 3)</td>
</tr>
<tr>
<td>NSL</td>
<td>INPUTT</td>
<td>TSONIN</td>
<td>Input number of streamlines from hub to shroud (designated by values in FLFR) at which output is desired, maximum of 50</td>
</tr>
<tr>
<td>NSLTS</td>
<td>TSONIN</td>
<td>TSONIN</td>
<td>Input for TSONIC (ref. 3)</td>
</tr>
<tr>
<td>NSPL1</td>
<td>TSONIN</td>
<td>TSONIN</td>
<td>Input for TSONIC (ref. 3)</td>
</tr>
<tr>
<td>NSPL2</td>
<td>TSONIN</td>
<td>TSONIN</td>
<td>Input for TSONIC (ref. 3)</td>
</tr>
<tr>
<td>NSUB</td>
<td>TVELCY</td>
<td>TVELCY</td>
<td>Integer number of times WHUB is decreased for restart of iteration procedure</td>
</tr>
<tr>
<td>NTIP</td>
<td>INPUTT</td>
<td>TVELCY</td>
<td>Number of input data points in ZTIP and RTIP arrays, maximum of 50</td>
</tr>
<tr>
<td>Variable name</td>
<td>COMMON block</td>
<td>Subroutine</td>
<td>Description and comments</td>
</tr>
<tr>
<td>---------------------</td>
<td>--------------</td>
<td>------------</td>
<td>--------------------------</td>
</tr>
<tr>
<td>NWRIT, NWRT1,</td>
<td>Blank</td>
<td></td>
<td>Integer numbers of output write files. Several variables are used so that output can be stored in more than one file.</td>
</tr>
<tr>
<td>NWRT2, NWRT3,</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>NWRT4, NWRT5,</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>NWRT6, NWRT7</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>OMEGA</td>
<td>INPUTT</td>
<td>TSONIN</td>
<td>Input rotational speed, ( \omega ), rad/sec</td>
</tr>
<tr>
<td>OMTEM</td>
<td>SOR</td>
<td>SOR</td>
<td>Overrelaxation factor</td>
</tr>
<tr>
<td>ORF</td>
<td>SOR</td>
<td>SOR</td>
<td>Current estimate for maximum value of ORF calculated by using LMAX</td>
</tr>
<tr>
<td>ORFMAX</td>
<td>SOR</td>
<td>SOR</td>
<td>Current estimate for minimum value of ORF calculated by using LMIN</td>
</tr>
<tr>
<td>ORFMIN</td>
<td>SOR</td>
<td>SOR</td>
<td></td>
</tr>
<tr>
<td>PC</td>
<td>LOSSOM</td>
<td>LOSSOM</td>
<td>Fraction of chord from leading edge</td>
</tr>
<tr>
<td>PERCRD(NLOSS)</td>
<td>INPUTT</td>
<td>TSONIN</td>
<td>Input fraction of chord for PERLOS input</td>
</tr>
<tr>
<td>PERLOS(NLOSS)</td>
<td>INPUTT</td>
<td>LOSSOM</td>
<td>Input fraction of loss</td>
</tr>
<tr>
<td>PERLS</td>
<td></td>
<td>LOSSOM</td>
<td>Fraction of loss at an orthogonal mesh point</td>
</tr>
<tr>
<td>PHI</td>
<td>MESH</td>
<td>OUTPUT</td>
<td>( \varphi ), deg</td>
</tr>
<tr>
<td>PITCH</td>
<td>CALCON</td>
<td></td>
<td>( 2\pi/NBL ), rad</td>
</tr>
<tr>
<td>PLOSS(MM, MHTP1)</td>
<td>CALCON</td>
<td>TSONIN</td>
<td>Fractional loss of relative total pressure at orthogonal mesh points</td>
</tr>
<tr>
<td>PLOSSL(MM)</td>
<td></td>
<td>TSONIN</td>
<td>Fractional loss of relative total pressure along a streamline</td>
</tr>
<tr>
<td>PLOST(NSL)</td>
<td>OUTPUT</td>
<td></td>
<td>PLOSS at a point on a station line</td>
</tr>
<tr>
<td>PLOSTE</td>
<td>LINDV</td>
<td></td>
<td>Fractional loss of relative total pressure at blade trailing edge</td>
</tr>
<tr>
<td>PPPST(NSL)</td>
<td>OUTPUT</td>
<td></td>
<td>( p'' ) along a station line, N/m^2</td>
</tr>
<tr>
<td>PPST(NSL)</td>
<td>OUTPUT</td>
<td></td>
<td>( p' ) along a station line, N/m^2</td>
</tr>
<tr>
<td>Variable name</td>
<td>COMMON block</td>
<td>Subroutine</td>
<td>Description and comments</td>
</tr>
<tr>
<td>--------------------------</td>
<td>--------------</td>
<td>------------</td>
<td>---------------------------------------------------------------</td>
</tr>
<tr>
<td>PI (MH TP)</td>
<td></td>
<td>NEWRHO</td>
<td>p&quot; at mesh points along vertical mesh lines, N/m²</td>
</tr>
<tr>
<td>PI</td>
<td></td>
<td>TVELCY</td>
<td>p&quot;, N/m²</td>
</tr>
<tr>
<td>PRINP</td>
<td></td>
<td>TVELCY</td>
<td>New p&quot;, N/m²</td>
</tr>
<tr>
<td>PRIP (NIN)</td>
<td>INPUTT</td>
<td>LOSSOM</td>
<td>p'_i, N/m²</td>
</tr>
<tr>
<td>PROP (NOUT)</td>
<td>INPUTT</td>
<td></td>
<td>Input, p'_i, at points along line from hub to shroud on which upstream flow conditions are given, N/m²</td>
</tr>
<tr>
<td>PST (NSL)</td>
<td>OUTPUT</td>
<td></td>
<td>p along a station line, N/m²</td>
</tr>
<tr>
<td>R1 (MM)</td>
<td>MESHO</td>
<td></td>
<td>r-coordinate of intersection of line 1, fig. 9, with upper horizontal mesh line, m</td>
</tr>
<tr>
<td>RADIN (NIN)</td>
<td>INPUTT</td>
<td></td>
<td>Input r-coordinates of points along line from hub to shroud on which upstream flow conditions are given, m</td>
</tr>
<tr>
<td>RADLE</td>
<td></td>
<td>TSONIN</td>
<td>r at leading edge, m</td>
</tr>
<tr>
<td>RADOUT (NOUT)</td>
<td>INPUTT</td>
<td></td>
<td>Input r-coordinates of points along line from hub to shroud on which downstream flow conditions are given, m</td>
</tr>
<tr>
<td>RADSP1 (MM)</td>
<td>TSONIN</td>
<td></td>
<td>r-coordinates of upper-blade-surface spline points, m</td>
</tr>
<tr>
<td>RADSP2 (MM)</td>
<td>TSONIN</td>
<td></td>
<td>r-coordinates of lower-blade-surface spline points, m</td>
</tr>
<tr>
<td>RADTE</td>
<td>TSONIN</td>
<td></td>
<td>r at trailing edge, m</td>
</tr>
<tr>
<td>RATIO</td>
<td>SOR</td>
<td></td>
<td>u_{m,i}^{m+1}/u_{m,i}^m for use in equations (B2) and (B3) of reference 9</td>
</tr>
<tr>
<td>RBL (NPPP, NBLPL)</td>
<td>INPUTT</td>
<td></td>
<td>Input array of r-coordinates, corre-</td>
</tr>
<tr>
<td>Variable name</td>
<td>COMMON block</td>
<td>Subroutine</td>
<td>Description and comments</td>
</tr>
<tr>
<td>---------------</td>
<td>--------------</td>
<td>------------</td>
<td>--------------------------</td>
</tr>
<tr>
<td>RBLR</td>
<td></td>
<td>THIKOM</td>
<td>Rotated RBL points at leading and trailing edges, m</td>
</tr>
<tr>
<td>RBLROT(NPPP, NBLPL)</td>
<td>ROTATN</td>
<td></td>
<td>Rotated RBL array, m</td>
</tr>
<tr>
<td>RCARB(MHTP1)</td>
<td></td>
<td>TVELCY</td>
<td>( \cos(\alpha - \phi) rB ) along a vertical mesh line, m</td>
</tr>
<tr>
<td>REDFAC</td>
<td></td>
<td>INPUTT</td>
<td>Input factor used to reduce mass flow (MSFL) in order to ensure subsonic flow throughout flow passage</td>
</tr>
<tr>
<td>REDTEM</td>
<td></td>
<td>TSONIN</td>
<td>Input for TSONIC (ref. 3)</td>
</tr>
<tr>
<td>RELER</td>
<td></td>
<td>NEWRHO</td>
<td>Maximum relative change in W at any mesh point between two outer iterations</td>
</tr>
<tr>
<td>RELERA</td>
<td></td>
<td>NEWRHO</td>
<td>Average relative change in W for all mesh points between two outer iterations</td>
</tr>
<tr>
<td>REPEAT</td>
<td></td>
<td>TVELCY</td>
<td>Logical variable indicating that velocity-gradient solutions should be repeated with new values of TIPT, RHOIP, and LAMBDA</td>
</tr>
<tr>
<td>REVERS</td>
<td></td>
<td>OUTPUT</td>
<td>Indicator of which blade surface (leading or trailing) is suction surface</td>
</tr>
<tr>
<td>RFAC2</td>
<td></td>
<td>LOSSOM</td>
<td>REDFAC squared, or 1.0 for transonic solution</td>
</tr>
<tr>
<td>RHIN</td>
<td></td>
<td>INPUTT</td>
<td>Input or calculated r-coordinate of intersection with hub profile of line on which upstream flow conditions are given, m</td>
</tr>
<tr>
<td>RHO(MM, MHTP1)</td>
<td>VARCOM</td>
<td></td>
<td>( \rho ), at orthogonal mesh points, kg/m(^3)</td>
</tr>
<tr>
<td>RHOAV(MM, MHTP1)</td>
<td>VARCOM</td>
<td></td>
<td>Average density across flow channel from suction surface to pressure surface, at orthogonal mesh points, kg/m(^3)</td>
</tr>
<tr>
<td>Variable name</td>
<td>COMMON block</td>
<td>Subroutine</td>
<td>Description and comments</td>
</tr>
<tr>
<td>--------------</td>
<td>--------------</td>
<td>-------------</td>
<td>-------------------------</td>
</tr>
<tr>
<td>RHOBF</td>
<td>INDEV</td>
<td>LINDV</td>
<td>$\rho_{bf}$, kg/m$^3$</td>
</tr>
<tr>
<td>RHOBFN</td>
<td>LINDV</td>
<td></td>
<td>New $\rho_{bf}$, kg/m$^3$</td>
</tr>
<tr>
<td>RHOFS</td>
<td>INDEV</td>
<td>LINDV</td>
<td>$\rho_{fs}$, kg/m$^3$</td>
</tr>
<tr>
<td>RHOIP(MHTP1)</td>
<td>TVELCY</td>
<td></td>
<td>$\rho_i$ (1 - Ploss) for mesh points along vertical mesh lines, kg/m$^3$</td>
</tr>
<tr>
<td>RHOIP(NIN)</td>
<td>RHOIPF</td>
<td></td>
<td>$\rho_i$ at input points of upstream flow conditions, kg/m$^3$</td>
</tr>
<tr>
<td>RHOIP</td>
<td>NEWRHO</td>
<td>OUTPUT</td>
<td>$\rho_i$ (1 - Ploss), kg/m$^3$</td>
</tr>
<tr>
<td>RHOIP</td>
<td>TSONIN</td>
<td></td>
<td>$\rho_i$ input for TSONIC, kg/m$^3$</td>
</tr>
<tr>
<td>RHOL</td>
<td>BLDVEL</td>
<td>TVELCY</td>
<td>$\rho_l$, kg/m$^3$</td>
</tr>
<tr>
<td>RHOP</td>
<td>OUTPUT</td>
<td></td>
<td>$\rho'$, kg/m$^3$</td>
</tr>
<tr>
<td>RHOPP</td>
<td>OUTPUT</td>
<td></td>
<td>$\rho''$, kg/m$^3$</td>
</tr>
<tr>
<td>RHOSL</td>
<td>TSONIN</td>
<td></td>
<td>$\rho$ at a ZSL, RSL point along a streamline, kg/m$^3$</td>
</tr>
<tr>
<td>RHOST(NSL)</td>
<td>OUTPUT</td>
<td></td>
<td>$\rho$ along a station line, kg/m$^3$</td>
</tr>
<tr>
<td>RHOST</td>
<td>BLDVEL</td>
<td>TVELCY</td>
<td>$\rho_{tr}$, kg/m$^3$</td>
</tr>
<tr>
<td>RHOWAV</td>
<td></td>
<td>TVELCY</td>
<td>$(\rho W)_{av}$, kg/(sec)(m)$^2$</td>
</tr>
<tr>
<td>RHROT(NHUB)</td>
<td>ROTATN</td>
<td></td>
<td>Rotated RHUB array, m</td>
</tr>
<tr>
<td>RHST(NOSTAT)</td>
<td>INPUTT</td>
<td></td>
<td>Input or calculated r-coordinates of intersections of hub-shroud output station lines with hub profile, m</td>
</tr>
<tr>
<td>Variable name</td>
<td>COMMON block</td>
<td>Subroutine</td>
<td>Description and comments</td>
</tr>
<tr>
<td>---------------</td>
<td>--------------</td>
<td>------------</td>
<td>-------------------------</td>
</tr>
<tr>
<td>RHUB(NHUB)</td>
<td>INPUTT</td>
<td></td>
<td>Input r-coordinates of points defining hub or bottom boundary of flow channel, m</td>
</tr>
<tr>
<td>RI1</td>
<td>TSONIN</td>
<td></td>
<td>Input for TSONIC (ref. 3), m</td>
</tr>
<tr>
<td>RI2</td>
<td>TSONIN</td>
<td></td>
<td>Input for TSONIC (ref. 3), m</td>
</tr>
<tr>
<td>RILOM(MHTP1)</td>
<td>LAMDAF</td>
<td></td>
<td>Radii for spline fit of stream function against radius</td>
</tr>
<tr>
<td>RIR</td>
<td>LOSSOM</td>
<td></td>
<td>Rotated r-coordinate of intersection of upstream or downstream input station line with horizontal mesh line, m</td>
</tr>
<tr>
<td>RLE(NBLPL)</td>
<td>CALCON</td>
<td></td>
<td>r-coordinates of input blade-section points defining leading edge of blade, m</td>
</tr>
<tr>
<td>RLEH</td>
<td>CALCON</td>
<td></td>
<td>r-coordinate of intersection of leading edge of blade with hub profile, m</td>
</tr>
<tr>
<td>RLEOM(MHTP1)</td>
<td>CALCON</td>
<td></td>
<td>r-coordinates of intersections of horizontal mesh lines with blade leading edge, m</td>
</tr>
<tr>
<td>RLEOMR(MHTP1)</td>
<td>ROTATN</td>
<td></td>
<td>Rotated RLEOM array, m</td>
</tr>
<tr>
<td>RLESL</td>
<td>TSONIN</td>
<td></td>
<td>r-coordinate of intersection of a streamline with leading edge of blade, m</td>
</tr>
<tr>
<td>RLET</td>
<td>CALCON</td>
<td></td>
<td>r-coordinate of intersection of leading edge of blade with shroud profile, m</td>
</tr>
<tr>
<td>RMSP(MM)</td>
<td>TSONIN</td>
<td></td>
<td>r-coordinates of points defining a stream channel, printed as input for TSONIC (ref. 3), m</td>
</tr>
<tr>
<td>RNOR(2)</td>
<td>MESHO</td>
<td></td>
<td>Rotated r-coordinates of points on straight line normal to previous row, m</td>
</tr>
<tr>
<td>RO1</td>
<td>TSONIN</td>
<td></td>
<td>Input for TSONIC (ref. 3), m</td>
</tr>
<tr>
<td>Variable name</td>
<td>COMMON block</td>
<td>Subroutine</td>
<td>Description and comments</td>
</tr>
<tr>
<td>---------------</td>
<td>--------------</td>
<td>------------</td>
<td>-------------------------</td>
</tr>
<tr>
<td>RO2</td>
<td></td>
<td>TSONIN</td>
<td>Input for TSONIC (ref. 3), m</td>
</tr>
<tr>
<td>ROLOM(MHTP1)</td>
<td></td>
<td>RVTHTA</td>
<td>Radii for spline fit of stream function against radius</td>
</tr>
<tr>
<td>ROM(MM, MHTP1)</td>
<td>CALCON</td>
<td></td>
<td>r-coordinates of orthogonal mesh, m</td>
</tr>
<tr>
<td>ROMBI</td>
<td>INPUTT</td>
<td></td>
<td>Input or calculated r-coordinate of intersection of vertical mesh line with hub profile where first change in mesh spacing occurs (MBI), m</td>
</tr>
<tr>
<td>ROMBO</td>
<td>INPUTT</td>
<td></td>
<td>Input or calculated r-coordinate of intersection of vertical mesh line with hub profile where second change in mesh spacing occurs (MBO), m</td>
</tr>
<tr>
<td>ROMIN</td>
<td>INPUTT</td>
<td></td>
<td>Input or calculated r-coordinate of intersection of left boundary of orthogonal mesh with hub profile, m</td>
</tr>
<tr>
<td>ROMOUT</td>
<td>INPUTT</td>
<td></td>
<td>Input or calculated r-coordinate of intersection of right boundary of orthogonal mesh (MM) with hub profile, m</td>
</tr>
<tr>
<td>ROMROT(MM, MHTP1)</td>
<td>ROTATN</td>
<td></td>
<td>Rotated ROM array, m</td>
</tr>
<tr>
<td>RPC(NPPC, NBLPC)</td>
<td>INDCOM</td>
<td></td>
<td>r-coordinates of intersections of percent-chord lines with input blade planes; later set to r-coordinates of alternate mesh (fig. 27), m</td>
</tr>
<tr>
<td>RPCT1(NBLPL)</td>
<td>THETOM</td>
<td></td>
<td>r-coordinates of intersections of percent-chord lines with input blade planes, m</td>
</tr>
<tr>
<td>RPCT2(NBLPC)</td>
<td>THETOM</td>
<td></td>
<td>r-coordinates along percent-chord lines of alternate mesh, m</td>
</tr>
<tr>
<td>RRAD(NHUB, MHTP1)</td>
<td>MESHO</td>
<td></td>
<td>r-coordinates of points along lines from input points on hub profile to shroud profile, m</td>
</tr>
<tr>
<td>Variable name</td>
<td>COMMON block</td>
<td>Subroutine</td>
<td>Description and comments</td>
</tr>
<tr>
<td>---------------------</td>
<td>--------------</td>
<td>------------</td>
<td>-----------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>RSL(MM, NSL)</td>
<td>SLCOM</td>
<td></td>
<td>Array of r-coordinates of points along streamlines where they cross vertical mesh lines, m</td>
</tr>
<tr>
<td>RSLTEM(NSL)</td>
<td>OUTPUT</td>
<td></td>
<td>Temporary storage for calculated values to be put into RSL array, m</td>
</tr>
<tr>
<td>RST(NSL)</td>
<td>OUTPUT</td>
<td></td>
<td>r-coordinates of points along station lines where they cross streamlines, m</td>
</tr>
<tr>
<td>RTE(NBLPL)</td>
<td>CALCON</td>
<td></td>
<td>r-coordinates of input blade-section points defining trailing edge of blade, m</td>
</tr>
<tr>
<td>RTEH</td>
<td>CALCON</td>
<td></td>
<td>r-coordinate or intersection of trailing edge of blade with hub profile, m</td>
</tr>
<tr>
<td>RTEOM(MHTP1, NOSTAT, or 20)</td>
<td>OUTPUT</td>
<td></td>
<td>Temporary storage for values from ROM array on vertical mesh lines; also temporary storage for values from RST array along station lines, m</td>
</tr>
<tr>
<td>RTEOMR(MHTP1)</td>
<td>ROTATN</td>
<td></td>
<td>Rotated RTEOM array, m</td>
</tr>
<tr>
<td>RTESL</td>
<td>TSONIN</td>
<td></td>
<td>r-coordinate of intersection of a streamline with trailing edge of blade, m</td>
</tr>
<tr>
<td>RTET</td>
<td>CALCON</td>
<td></td>
<td>r-coordinate of intersection of trailing edge of blade with shroud profile, m</td>
</tr>
<tr>
<td>RTIN</td>
<td>CALCON</td>
<td></td>
<td>Input or calculated r-coordinate of intersection with shroud profile of line on which upstream flow conditions are given, m</td>
</tr>
<tr>
<td>RTIP(NTIP)</td>
<td>INPUTT</td>
<td></td>
<td>Input r-coordinates of points defining shroud or top boundary of flow channel, m</td>
</tr>
<tr>
<td>Variable name</td>
<td>COMMON block</td>
<td>Subroutine</td>
<td>Description and comments</td>
</tr>
<tr>
<td>---------------</td>
<td>--------------</td>
<td>------------</td>
<td>--------------------------</td>
</tr>
<tr>
<td>RTOLER</td>
<td></td>
<td>TVELCY</td>
<td>Tolerance for relative error of calculated values of integrated mass flow</td>
</tr>
<tr>
<td>RTOUT</td>
<td>INPUTT</td>
<td></td>
<td>Input or calculated r-coordinate of intersection with shroud profile of line on which downstream flow conditions are given, m</td>
</tr>
<tr>
<td>RTROT(NTIP)</td>
<td>ROTATN</td>
<td></td>
<td>Rotated RTIP array, m</td>
</tr>
<tr>
<td>RTST(NOSTAT)</td>
<td>INPUTT</td>
<td></td>
<td>Input or calculated r-coordinates of intersections of hub-shroud output station lines with tip profile, m</td>
</tr>
<tr>
<td>RVA</td>
<td>TVELCY</td>
<td></td>
<td>$(\rho W)_{av,j} \cos \beta_j \cos(\alpha - \varphi)_j r_j B_j$, kg/(m)(sec)</td>
</tr>
<tr>
<td>RVAS</td>
<td>TVELCY</td>
<td></td>
<td>$(\rho W)<em>{av,j+1} \cos \beta</em>{j+1}$ $\times \cos(\alpha - \varphi)<em>{j+1} r</em>{j+1} B_{j+1}$, kg/(m)(sec)</td>
</tr>
<tr>
<td>RVTHI</td>
<td>TSONIN</td>
<td></td>
<td>Input for TSONIC (ref. 3), m$^2$/sec</td>
</tr>
<tr>
<td>RVTHO</td>
<td>TSONIN</td>
<td></td>
<td>Input for TSONIC (ref. 3), m$^2$/sec</td>
</tr>
<tr>
<td>SAL</td>
<td>TVELCY</td>
<td></td>
<td>$\sin \alpha$</td>
</tr>
<tr>
<td>SAMP(MM, MHTP1)</td>
<td>VARCOM</td>
<td></td>
<td>$\sin(\alpha - \varphi)$ at orthogonal mesh points</td>
</tr>
<tr>
<td>SBETA</td>
<td>TVELCY</td>
<td></td>
<td>$\sin \beta$</td>
</tr>
<tr>
<td>SDIST</td>
<td>INDEV</td>
<td></td>
<td>DISTLE (or DISTTE) plus distance from blade leading (or trailing) edge out to first adjacent mesh point, m</td>
</tr>
<tr>
<td>SF</td>
<td>LAMDAF</td>
<td></td>
<td>Stream function</td>
</tr>
<tr>
<td>SFIN(NIN)</td>
<td>INPUTT</td>
<td></td>
<td>Input values of stream function along hub-shroud line on which upstream flow conditions are given</td>
</tr>
<tr>
<td>SFOUT(NOUT)</td>
<td>INPUTT</td>
<td></td>
<td>Input values of stream function along</td>
</tr>
<tr>
<td>Variable name</td>
<td>COMMON block</td>
<td>Subroutine</td>
<td>Description and comments</td>
</tr>
<tr>
<td>---------------</td>
<td>--------------</td>
<td>------------</td>
<td>--------------------------</td>
</tr>
<tr>
<td>SLENTH</td>
<td></td>
<td>LOSSOM</td>
<td>hub-shroud line on which downstream flow conditions are given</td>
</tr>
<tr>
<td>SLEOM(MHTP1)</td>
<td></td>
<td>CALCON</td>
<td>Overall s-distance over which loss is distributed, m</td>
</tr>
<tr>
<td>SLIDLE</td>
<td></td>
<td>INDEV</td>
<td>s-coordinates of intersections of horizontal mesh lines with blade leading edge, m</td>
</tr>
<tr>
<td>SLIDTE</td>
<td></td>
<td>INDEV</td>
<td>Solidity at leading edge of blade where it is intersected by a horizontal mesh line</td>
</tr>
<tr>
<td>SLOM(MM)</td>
<td></td>
<td>MESHO</td>
<td>Solidity at trailing edge of blade where it is intersected by a horizontal mesh line</td>
</tr>
<tr>
<td>SLOPE(NIN or NOUT)</td>
<td></td>
<td>TIPF, RHOIPF, LAMDAF, RVTHTA</td>
<td>Slopes of horizontal mesh lines at mesh points</td>
</tr>
<tr>
<td>SOM(MM, MHTP1)</td>
<td></td>
<td>CALCON</td>
<td>s-coordinates of orthogonal mesh, m</td>
</tr>
<tr>
<td>SOMIN(MHTP1)</td>
<td></td>
<td>LOSSOM</td>
<td>s-coordinate of intersection of line on which upstream flow conditions are given with horizontal mesh lines, when there are no blades, m</td>
</tr>
<tr>
<td>SOMOUT(MHTP1)</td>
<td></td>
<td>LOSSOM</td>
<td>s-coordinates of intersection of line on which downstream flow conditions are given with horizontal mesh lines, m</td>
</tr>
<tr>
<td>SPHI(MM, MHTP1)</td>
<td></td>
<td>CALCON</td>
<td>sin φ at orthogonal mesh points</td>
</tr>
<tr>
<td>STEOM(MHTP1)</td>
<td></td>
<td>CALCON</td>
<td>s-coordinates of intersections of horizontal mesh lines with blade trailing edge, m</td>
</tr>
<tr>
<td>STGR</td>
<td></td>
<td>TSONIN</td>
<td>Input for TSONIC (ref. 3), rad</td>
</tr>
</tbody>
</table>

90
<table>
<thead>
<tr>
<th>Variable name</th>
<th>COMMON block</th>
<th>Subroutine</th>
<th>Description and comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>SZRBL(NPPP or NBLPL)</td>
<td></td>
<td>PRECAL</td>
<td>Arc length along input blade section or along hub-shroud percent-chord lines in meridional plane, m</td>
</tr>
<tr>
<td>SZRPC(NPPC or NBLPC)</td>
<td></td>
<td>THETOM</td>
<td>Arc length along vertical or horizontal lines of alternate mesh (fig. 27)</td>
</tr>
<tr>
<td>TANBBBF</td>
<td></td>
<td>INDEV</td>
<td>(\tan \beta_{bf})</td>
</tr>
<tr>
<td>TANBBL</td>
<td></td>
<td>INDEV</td>
<td>(\tan \beta_{b})</td>
</tr>
<tr>
<td>TBFTIP</td>
<td></td>
<td>LINDV</td>
<td>(\frac{T_{bf}}{T_{i}})</td>
</tr>
<tr>
<td>TGROG</td>
<td></td>
<td>CALCON</td>
<td>(2\gamma R/(\gamma + 1))</td>
</tr>
<tr>
<td>TH1BL(NPPP, NBLPL)</td>
<td></td>
<td>INPUTT</td>
<td>Input array of (\theta)-coordinates, corresponding to ZBL, RBL of points describing upper blade surface, rad</td>
</tr>
<tr>
<td>TH2BL(NPPP, NBLPL)</td>
<td></td>
<td>INPUTT</td>
<td>Input array of (\theta)-coordinates, corresponding to ZBL, RBL of points describing lower blade surface, rad</td>
</tr>
<tr>
<td>THBL(NPPP, NBLPL)</td>
<td></td>
<td>INPUTT</td>
<td>Input array of (\theta)-coordinates, corresponding to ZBL, RBL of points describing mean blade surface, rad</td>
</tr>
<tr>
<td>THLEOM(MHTP1)</td>
<td></td>
<td>CALCON</td>
<td>(\theta)-coordinates of intersections of horizontal mesh lines with blade leading edge, rad</td>
</tr>
<tr>
<td>THLESL</td>
<td></td>
<td>TSONIN</td>
<td>(\theta)-coordinate of intersection of streamline with blade leading edge, rad</td>
</tr>
<tr>
<td>THPC(NPPC, NBLPC)</td>
<td></td>
<td>INDCOM</td>
<td>(\theta)-coordinates of intersections of percent-chord lines with input blade planes; later set to (\theta)-coordinates of alternate mesh (fig. 27), rad</td>
</tr>
<tr>
<td>THPCT1(NBLPL)</td>
<td></td>
<td>THETOM</td>
<td>(\theta)-coordinates of intersections of percent-chord lines with input blade planes, rad</td>
</tr>
<tr>
<td>Variable name</td>
<td>COMMON block</td>
<td>Subroutine</td>
<td>Description and comments</td>
</tr>
<tr>
<td>---------------</td>
<td>--------------</td>
<td>------------</td>
<td>-------------------------</td>
</tr>
<tr>
<td>THPCT2(NBLPC)</td>
<td></td>
<td>THETOM</td>
<td>( \theta )-coordinates along percent-chord lines of alternate mesh, rad</td>
</tr>
<tr>
<td>THSL</td>
<td></td>
<td>TSONIN</td>
<td>( \theta )-coordinate (relative to MERIDL origin, not TSONIC origin) of mean blade surface at points along meridional streamlines, rad</td>
</tr>
<tr>
<td>THSP1(MM)</td>
<td></td>
<td>TSONIN</td>
<td>Input for TSONIC (ref. 3), rad</td>
</tr>
<tr>
<td>THSP2(MM)</td>
<td></td>
<td>TSONIN</td>
<td>Input for TSONIC (ref. 3), rad</td>
</tr>
<tr>
<td>THSTAK</td>
<td></td>
<td>TSONIN</td>
<td>( \theta )-coordinate of blade section in relation to hub blade section, rad</td>
</tr>
<tr>
<td>THTEOM(MHTP1)</td>
<td>CALCON</td>
<td></td>
<td>( \theta )-coordinates of intersections of horizontal mesh lines with blade trailing edge, rad</td>
</tr>
<tr>
<td>THTESL</td>
<td></td>
<td>TSONIN</td>
<td>( \theta )-coordinate of intersection of meridional streamline with blade trailing edge, rad</td>
</tr>
<tr>
<td>TINP</td>
<td></td>
<td>LOSSOM</td>
<td>( T'_i ), K</td>
</tr>
<tr>
<td>TIP(NIN)</td>
<td></td>
<td>INPUTT</td>
<td>Input ( T'_i ) at points along the line from hub to shroud on which upstream flow conditions are given, K</td>
</tr>
<tr>
<td>TIPT(MHTP1)</td>
<td></td>
<td>TVELCY</td>
<td>( T'_i ) at points along vertical mesh lines, K</td>
</tr>
<tr>
<td>TIPT</td>
<td></td>
<td>NEWRHO</td>
<td>Temporary ( T'_i )</td>
</tr>
<tr>
<td>TIPTEM</td>
<td></td>
<td>TSONIN</td>
<td>( T'_i ) along a streamline, K; input TIP for TSONIC (ref. 3)</td>
</tr>
<tr>
<td>TITLEI(20)</td>
<td>INPUTT</td>
<td></td>
<td>Alphanumerical contents of input title card</td>
</tr>
<tr>
<td>TLTAN1</td>
<td></td>
<td>TSONIN</td>
<td>( \theta )-coordinate at upper blade surface leading-edge tangency point, rad</td>
</tr>
<tr>
<td>TLTAN2</td>
<td></td>
<td>TSONIN</td>
<td>( \theta )-coordinate at lower blade surface leading-edge tangency point, rad</td>
</tr>
<tr>
<td>Variable name</td>
<td>COMMON block</td>
<td>Subroutine</td>
<td>Description and comments</td>
</tr>
<tr>
<td>---------------</td>
<td>--------------</td>
<td>------------</td>
<td>--------------------------</td>
</tr>
<tr>
<td>TMSL</td>
<td></td>
<td>TSONIN</td>
<td>m-coordinates of tangency points on both blade surfaces, m</td>
</tr>
<tr>
<td>TNBL(NPPP, NBLPL)</td>
<td>INPUTT</td>
<td></td>
<td>Input array of blade normal thicknesses, corresponding to ZBL, RBL coordinates, m</td>
</tr>
<tr>
<td>TOLER</td>
<td></td>
<td>TIPF</td>
<td>Tolerance for a point close to a spline point</td>
</tr>
<tr>
<td></td>
<td></td>
<td>RHOIPF</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>LAMDAF</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>RVTHTA</td>
<td></td>
</tr>
<tr>
<td>TOM(MM, MHTP1)</td>
<td>CALCON</td>
<td></td>
<td>t-coordinates of orthogonal mesh, m</td>
</tr>
<tr>
<td>TOP</td>
<td></td>
<td>LOSSOM</td>
<td>T'_0, K</td>
</tr>
<tr>
<td>TPP(MHTP1)</td>
<td></td>
<td>NEWRHO</td>
<td>T'' along a vertical mesh line, K</td>
</tr>
<tr>
<td></td>
<td></td>
<td>TVELCY</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>OUTPUT</td>
<td></td>
</tr>
<tr>
<td>TPP</td>
<td></td>
<td>OUTPUT</td>
<td>T'', K</td>
</tr>
<tr>
<td></td>
<td></td>
<td>TVELCY</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>LINDV</td>
<td></td>
</tr>
<tr>
<td>TPPN</td>
<td></td>
<td>TVELCY</td>
<td>New T'', K</td>
</tr>
<tr>
<td>TPPST(NSL)</td>
<td></td>
<td>OUTPUT</td>
<td>T'' along a station line, K</td>
</tr>
<tr>
<td>TPPTIP</td>
<td></td>
<td>INIT</td>
<td>T''/T_1</td>
</tr>
<tr>
<td>TPST(NSL)</td>
<td></td>
<td>OUTPUT</td>
<td>T along a station line, K</td>
</tr>
<tr>
<td>TST(NSL)</td>
<td></td>
<td>OUTPUT</td>
<td>T along a station line, K</td>
</tr>
<tr>
<td>TTBL(NPPP, NBLPL)</td>
<td>INPUTT</td>
<td></td>
<td>Input array of blade tangential thicknesses, corresponding to ZBL, RBL coordinates, m</td>
</tr>
<tr>
<td>TTIP</td>
<td></td>
<td>NEWRHO</td>
<td>T/T_1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>BLDVEL</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>TVELCY</td>
<td></td>
</tr>
<tr>
<td>TTPC(NPPC, NBLPC)</td>
<td>INDCOM</td>
<td></td>
<td>Tangential blade thicknesses at intersections of percent-chord lines with input blade planes; later set to tang-</td>
</tr>
<tr>
<td>Variable name</td>
<td>COMMON block</td>
<td>Subroutine</td>
<td>Description and comments</td>
</tr>
<tr>
<td>---------------</td>
<td>--------------</td>
<td>------------</td>
<td>-------------------------</td>
</tr>
<tr>
<td>TTPCT1(NBLPL)</td>
<td>THETOM</td>
<td></td>
<td>Tangential blade thicknesses on alternate mesh (fig. 27), rad</td>
</tr>
<tr>
<td>TTPCT2(NBLPC)</td>
<td>THETOM</td>
<td></td>
<td>Tangential blade thicknesses at intersections of percent-chord lines with input blade planes, rad</td>
</tr>
<tr>
<td>TTTAN1</td>
<td>TSONIN</td>
<td></td>
<td>Tangential blade thicknesses along percent-chord lines of alternate mesh, rad</td>
</tr>
<tr>
<td>TTTAN2</td>
<td>TSONIN</td>
<td></td>
<td>0-coordinate at upper blade surface trailing-edge tangency point, rad</td>
</tr>
<tr>
<td>TVERT(MHTP1)</td>
<td>NEWRHO, BLDVEL, OUTPUT, TVELCY</td>
<td></td>
<td>Temporary storage for values from TOM array on a vertical mesh line, m</td>
</tr>
<tr>
<td>TWLMR(MHTP1)</td>
<td>TVELCY</td>
<td></td>
<td>$2\omega\lambda - (\omega r)^2$ at points along vertical mesh lines, $m^2/sec^2$</td>
</tr>
<tr>
<td>TWLMR</td>
<td>BLDVEL, LINDV</td>
<td></td>
<td>$2\omega\lambda - (\omega r)^2$, $m^2/sec^2$</td>
</tr>
<tr>
<td>UBDEV(MHTP1)</td>
<td>LINDV</td>
<td></td>
<td>Deviation angle, neglecting blockage correction, where horizontal orthogonal intersects blade, $(\beta_{fs} - \beta_{le})_v$, deg</td>
</tr>
<tr>
<td>UBDEV</td>
<td>INDEV</td>
<td></td>
<td>Deviation angle, neglecting blockage correction, $(\beta_{fs} - \beta_{te})_v$, deg</td>
</tr>
<tr>
<td>UBIN(MHTP1)</td>
<td>LINDV</td>
<td></td>
<td>Incidence angle, neglecting blockage correction, where horizontal orthogonal intersects blade, $(\beta_{fs} - \beta_{le})_v$, deg</td>
</tr>
<tr>
<td>UBIN</td>
<td>INDEV</td>
<td></td>
<td>Incidence angle, neglecting blockage correction, $(\beta_{fs} - \beta_{le})_v$, deg</td>
</tr>
</tbody>
</table>

94
<table>
<thead>
<tr>
<th>Variable name</th>
<th>COMMON block</th>
<th>Subroutine</th>
<th>Description and comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>UILOM(MHTP1)</td>
<td></td>
<td>LAMDAF</td>
<td>Stream-function values for spline fit of stream function against radius</td>
</tr>
<tr>
<td>UNEW(MHTP1)</td>
<td></td>
<td>TVELCY</td>
<td>New estimate for ( u ) along a vertical mesh line</td>
</tr>
<tr>
<td>UNEW</td>
<td></td>
<td>SOR</td>
<td>New estimate for ( u ) at a mesh point</td>
</tr>
<tr>
<td>UOLOM(MHTP1)</td>
<td></td>
<td>RVTHTA</td>
<td>Stream-function values for spline fit of stream function against radius</td>
</tr>
<tr>
<td>UOM(MM, MHTP1)</td>
<td>VARCOM</td>
<td></td>
<td>Stream function ( u ) at orthogonal mesh points</td>
</tr>
<tr>
<td>UTEM(MHTP1 or 20)</td>
<td></td>
<td>OUTPUT</td>
<td>Temporary storage for values from UOM array on vertical mesh lines; also stream function at 20 equally spaced points from hub to shroud</td>
</tr>
<tr>
<td>UVERT(MHTP1)</td>
<td></td>
<td>NEWRHO</td>
<td>Temporary storage for values from UOM array along vertical mesh lines</td>
</tr>
<tr>
<td>VELTEM</td>
<td></td>
<td>TSONIN</td>
<td>Input for TSONIC (ref. 3)</td>
</tr>
<tr>
<td>VELTOL</td>
<td>INPUTT</td>
<td></td>
<td>Input convergence tolerance on maximum velocity change in each outer iteration, over all mesh points, for reduced mass flow</td>
</tr>
<tr>
<td>VST(NSL)</td>
<td></td>
<td>OUTPUT</td>
<td>( V ) along a station line, m/sec</td>
</tr>
<tr>
<td>VTH(MM, MHTP1)</td>
<td>VARCOM</td>
<td></td>
<td>( V_\theta ) at orthogonal mesh points, m/sec</td>
</tr>
<tr>
<td>VTHIN(NIN)</td>
<td>INPUTT</td>
<td></td>
<td>Input values of ( (V_\theta)_1 ) at points along line from hub to shroud on which upstream flow conditions are given, m/sec</td>
</tr>
<tr>
<td>VTHOUT(NOUT)</td>
<td>INPUTT</td>
<td></td>
<td>Input values of ( (V_\theta)_0 ) at points along line from hub to shroud on which downstream flow conditions are given, m/sec</td>
</tr>
<tr>
<td>VTHST(NSL)</td>
<td>OUTPUT</td>
<td></td>
<td>( V_\theta ) along station line, m/sec</td>
</tr>
<tr>
<td>Variable name</td>
<td>COMMON block</td>
<td>Subroutine</td>
<td>Description and comments</td>
</tr>
<tr>
<td>---------------</td>
<td>--------------</td>
<td>------------</td>
<td>--------------------------</td>
</tr>
<tr>
<td>W(MM, MHTP1)</td>
<td>VARCOM</td>
<td></td>
<td>W at orthogonal mesh points, m/sec</td>
</tr>
<tr>
<td>WAS</td>
<td></td>
<td>TVELCY</td>
<td>First estimate of $W_{j+1}^\star$ at next mesh point along vertical mesh line (eq. (5)), $W_{j+1}^\star$, m/sec</td>
</tr>
<tr>
<td>WASS</td>
<td></td>
<td>TVELCY</td>
<td>Second estimate of $W_{j+1}^\star\star$ at next mesh point along vertical mesh line (eq. (5)), $W_{j+1}^\star\star$, m/sec</td>
</tr>
<tr>
<td>WFS</td>
<td></td>
<td></td>
<td>$W_{fs}$ at blade leading or trailing edge, m/sec</td>
</tr>
<tr>
<td>WHURL</td>
<td></td>
<td>TVELCY</td>
<td>$\lambda$ or $(\rho V_0)^2$, m$^2$/sec</td>
</tr>
<tr>
<td>WIIUB</td>
<td></td>
<td>TVELCY</td>
<td>Estimate of $W_{hub}$, m/sec</td>
</tr>
<tr>
<td>WLSRF</td>
<td></td>
<td>TVELCY</td>
<td>$W_\ell$, m/sec</td>
</tr>
<tr>
<td>WLSSL(MM, NSL)</td>
<td>SLCOM</td>
<td></td>
<td>$W_\ell$ at points along streamlines where they cross vertical mesh lines, m/sec</td>
</tr>
<tr>
<td>WLSST(NSL)</td>
<td></td>
<td>OUTPUT</td>
<td>$W_\ell$ at points along station lines where they cross streamlines, m/sec</td>
</tr>
<tr>
<td>WLSURF(MM, MHTP1)</td>
<td>VARCOM</td>
<td></td>
<td>$W_\ell$ on orthogonal mesh, m/sec</td>
</tr>
<tr>
<td>WMAX</td>
<td></td>
<td>TVELCY</td>
<td>Maximum value of $W$ at hub for which a solution can be obtained for a vertical mesh line, m/sec</td>
</tr>
<tr>
<td>WMIN</td>
<td></td>
<td>TVELCY</td>
<td>Minimum value of $W$ at hub for which a solution can be obtained for a vertical mesh line, m/sec</td>
</tr>
<tr>
<td>WMSL(MM, NSL)</td>
<td>SLCOM</td>
<td></td>
<td>$W_m$ at points where streamlines cross vertical mesh lines, m/sec</td>
</tr>
<tr>
<td>WMSON</td>
<td></td>
<td>LINDV</td>
<td>Sonic value for $W_m$, m/sec</td>
</tr>
<tr>
<td>WMST(NSL)</td>
<td></td>
<td>OUTPUT</td>
<td>$W_m$ at points where station lines cross streamlines, m/sec</td>
</tr>
<tr>
<td>WMVERT(MHTP1)</td>
<td></td>
<td>TVELCY</td>
<td>$W_m$ along vertical mesh lines, m/sec</td>
</tr>
<tr>
<td>Variable name</td>
<td>COMMON block</td>
<td>Subroutine</td>
<td>Description and comments</td>
</tr>
<tr>
<td>--------------------</td>
<td>--------------</td>
<td>------------</td>
<td>-----------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>WRFSEX(MHTP1)</td>
<td></td>
<td>OUTPUT</td>
<td>$w_r$ extrapolated up to leading or trailing edge of blade, m/sec</td>
</tr>
<tr>
<td>WRSL(MM, NSL)</td>
<td>SLCOM</td>
<td></td>
<td>$w_r$ at points where streamlines cross vertical mesh lines, m/sec</td>
</tr>
<tr>
<td>WRST(NSL)</td>
<td></td>
<td>OUTPUT</td>
<td>$w_r$ at points where station lines cross streamlines, m/sec</td>
</tr>
<tr>
<td>WSL(MM, NSL)</td>
<td>SLCOM</td>
<td></td>
<td>$w$ at points where streamlines cross vertical mesh lines, m/sec</td>
</tr>
<tr>
<td>WST(NSL)</td>
<td></td>
<td>OUTPUT</td>
<td>$w$ at points where station lines cross streamlines, m/sec</td>
</tr>
<tr>
<td>WSUBM(MM, MHTP1)</td>
<td>VARCOM</td>
<td></td>
<td>$w_m$ at orthogonal mesh points, m/sec</td>
</tr>
<tr>
<td>WSUBR(MM, MHTP1)</td>
<td>VARCOM</td>
<td></td>
<td>$w_r$ at orthogonal mesh points, m/sec</td>
</tr>
<tr>
<td>WSUBS(MM, MHTP1)</td>
<td>VARCOM</td>
<td></td>
<td>$w_s$ at orthogonal mesh points, m/sec</td>
</tr>
<tr>
<td>WSUBT(MM, MHTP1)</td>
<td>VARCOM</td>
<td></td>
<td>$w_t$ at orthogonal mesh points, m/sec</td>
</tr>
<tr>
<td>WSUBZ(MM, MHTP1)</td>
<td>VARCOM</td>
<td></td>
<td>$w_z$ at orthogonal mesh points, m/sec</td>
</tr>
<tr>
<td>WTEMP</td>
<td></td>
<td>NEWRHO</td>
<td>New calculated value of $w$ at a mesh point, m/sec</td>
</tr>
<tr>
<td>WTH(MM, MHTP1)</td>
<td>VARCOM</td>
<td></td>
<td>$w_\theta$ at orthogonal mesh points, m/sec</td>
</tr>
<tr>
<td>WTHETA</td>
<td>TVELCY</td>
<td></td>
<td>$w_\theta$, m/sec</td>
</tr>
<tr>
<td>WTHSL(MM, NSL)</td>
<td>SLCOM</td>
<td></td>
<td>$w_\theta$ at points where streamlines cross vertical mesh lines, m/sec</td>
</tr>
<tr>
<td>WTHST(NSL)</td>
<td></td>
<td>OUTPUT</td>
<td>$w_\theta$ at points where station lines cross streamlines, m/sec</td>
</tr>
<tr>
<td>WTSRF</td>
<td>TVELCY</td>
<td></td>
<td>$w_{tr}$, m/sec</td>
</tr>
<tr>
<td>WTSSL(MM, NSL)</td>
<td>SLCOM</td>
<td></td>
<td>$w_{tr}$ at points where streamlines cross vertical mesh lines, m/sec</td>
</tr>
<tr>
<td>WTSST(NSL)</td>
<td></td>
<td>OUTPUT</td>
<td>$w_{tr}$ at points where station lines cross streamlines, m/sec</td>
</tr>
<tr>
<td>Variable name</td>
<td>COMMON block</td>
<td>Subroutine</td>
<td>Description and comments</td>
</tr>
<tr>
<td>---------------------------</td>
<td>--------------</td>
<td>------------</td>
<td>------------------------------------------------</td>
</tr>
<tr>
<td>WTSURF (MM, MHTP1)</td>
<td>VARCOM</td>
<td>TVELCY</td>
<td>$W_{tr}$ on orthogonal mesh, m/sec</td>
</tr>
<tr>
<td>WTVERT (MM, MHTP1)</td>
<td></td>
<td></td>
<td>$W_{\theta}$ along vertical mesh lines, m/sec</td>
</tr>
<tr>
<td>WWCR (MM, MHTP1)</td>
<td>VARCOM</td>
<td></td>
<td>$W/W_{cr}$ at orthogonal mesh points</td>
</tr>
<tr>
<td>WWCRSL (MM, NSL)</td>
<td>SLCOM</td>
<td></td>
<td>$W/W_{cr}$ at points where streamlines cross vertical mesh lines</td>
</tr>
<tr>
<td>WWCRST (NSL)</td>
<td>OUTPUT</td>
<td></td>
<td>$W/W_{cr}$ at points where station lines cross streamlines</td>
</tr>
<tr>
<td>WZFSEX (MM, MHTPI)</td>
<td>OUTPUT</td>
<td></td>
<td>$W_z$ extrapolated up to leading or trailing edge of blade, m/sec</td>
</tr>
<tr>
<td>WZSL (MM, NSL)</td>
<td>SLCOM</td>
<td></td>
<td>$W_z$ at points where streamlines cross vertical mesh lines, m/sec</td>
</tr>
<tr>
<td>WZST (NSL)</td>
<td>OUTPUT</td>
<td></td>
<td>$W_z$ at points where station lines cross streamlines, m/sec</td>
</tr>
<tr>
<td>XIOM (MM, MHTP1)</td>
<td>VARCOM</td>
<td></td>
<td>$\xi$ at orthogonal mesh points, (eq. (A2)), 1/m</td>
</tr>
<tr>
<td>XIOMT</td>
<td>NEWRHO</td>
<td></td>
<td>New estimated value of $\xi$ at a mesh point</td>
</tr>
<tr>
<td>XNEW</td>
<td>NEWRHO</td>
<td></td>
<td>Percentage of new calculated value of XIOMT used in updating XIOM</td>
</tr>
<tr>
<td>Z1 (MM)</td>
<td>MESH0</td>
<td></td>
<td>$z$-coordinate of intersection of line (1), figure 9, with upper horizontal mesh line, m</td>
</tr>
<tr>
<td>Z2</td>
<td>MESH0</td>
<td></td>
<td>$z$-coordinate of intersection of line (2), figure 9, with upper horizontal mesh line, m</td>
</tr>
<tr>
<td>ZBL (NPPP, NBLPL)</td>
<td>INPUTT</td>
<td></td>
<td>Input array of $z$-coordinates of points describing blade surface, m</td>
</tr>
<tr>
<td>ZBLROT (NPPP, NBLPL)</td>
<td>ROTATN</td>
<td></td>
<td>Rotated ZBL array, m</td>
</tr>
<tr>
<td>ZETOM (MM, MHTP1)</td>
<td>VARCOM</td>
<td></td>
<td>$\xi$ at orthogonal mesh points (eq. (A3)), m/sec 2</td>
</tr>
<tr>
<td>Variable name</td>
<td>COMMON block</td>
<td>Subroutine</td>
<td>Description and comments</td>
</tr>
<tr>
<td>--------------------</td>
<td>--------------</td>
<td>------------</td>
<td>-------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>ZETOMT</td>
<td></td>
<td>NEWRHO</td>
<td>New estimated value of $\zeta$ at a mesh point</td>
</tr>
<tr>
<td>ZHIN</td>
<td>INPUTT</td>
<td></td>
<td>Input $z$-coordinate of intersection with hub profile of line on which up-</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>stream flow conditions are given, m</td>
</tr>
<tr>
<td>ZHOUT</td>
<td>INPUTT</td>
<td></td>
<td>Input $z$-coordinate of intersection with hub profile of line on which down-</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>stream flow conditions are given, m</td>
</tr>
<tr>
<td>ZHROT(NHUB)</td>
<td>ROTATN</td>
<td></td>
<td>Rotated ZHUB array, m</td>
</tr>
<tr>
<td>ZHST(NOSTAT)</td>
<td>INPUTT</td>
<td></td>
<td>Input $z$-coordinates of intersections of hub-shroud output station lines</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>with hub profile, m</td>
</tr>
<tr>
<td>ZHUB(NHUB)</td>
<td>INPUTT</td>
<td></td>
<td>Input $z$-coordinates of points defining hub or bottom boundary of flow</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>channel, m</td>
</tr>
<tr>
<td>ZIR</td>
<td>LOSSOM</td>
<td></td>
<td>Rotated $z$-coordinate of intersection of upstream or downstream input</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>station line with horizontal mesh line, m</td>
</tr>
<tr>
<td>ZLE(NBLPL)</td>
<td>CALCON</td>
<td></td>
<td>$z$-coordinates of input blade section points defining leading edge of blade</td>
</tr>
<tr>
<td>ZLEH</td>
<td>PRECAL</td>
<td></td>
<td>$z$-coordinate of intersection of leading edge of blade with hub profile, m</td>
</tr>
<tr>
<td>ZLEOM(MHTP1)</td>
<td>CALCON</td>
<td></td>
<td>$z$-coordinates of intersections of horizontal mesh lines with blade leading</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>edge, m</td>
</tr>
<tr>
<td>ZLEOMR(MHTP1)</td>
<td>ROTATN</td>
<td></td>
<td>Rotated ZLEOM array, m</td>
</tr>
<tr>
<td>ZLESL</td>
<td>TSONIN</td>
<td></td>
<td>$z$-coordinate of intersection of a streamline with leading edge of blade, m</td>
</tr>
<tr>
<td>ZLET</td>
<td>PRECAL</td>
<td></td>
<td>$z$-coordinate of intersection of leading edge of blade with shroud profile,</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>m</td>
</tr>
<tr>
<td>Variable name</td>
<td>COMMON block</td>
<td>Subroutine</td>
<td>Description and comments</td>
</tr>
<tr>
<td>---------------</td>
<td>--------------</td>
<td>------------</td>
<td>-------------------------</td>
</tr>
<tr>
<td>ZLTAN1</td>
<td></td>
<td>TSONIN</td>
<td>m-coordinate at upper blade surface leading-edge tangency point, m</td>
</tr>
<tr>
<td>ZLTAN2</td>
<td></td>
<td>TSONIN</td>
<td>m-coordinate at lower blade surface leading-edge tangency point, m</td>
</tr>
<tr>
<td>ZMRSP(MM)</td>
<td></td>
<td>TSONIN</td>
<td>Input for TSONIC (ref. 3), m</td>
</tr>
<tr>
<td>ZMSFL</td>
<td></td>
<td>TSONIN</td>
<td>Input for TSONIC (ref. 3), kg/sec</td>
</tr>
<tr>
<td>ZMSP1(MM)</td>
<td></td>
<td>TSONIN</td>
<td>Input for TSONIC (ref. 3), m</td>
</tr>
<tr>
<td>ZMSP2(MM)</td>
<td></td>
<td>TSONIN</td>
<td>Input for TSONIC (ref. 3), m</td>
</tr>
<tr>
<td>ZNEW</td>
<td></td>
<td>NEWRHO</td>
<td>Percentage of new calculated value of ZETOMT used in updating ZETOM</td>
</tr>
<tr>
<td>ZNOR(2)</td>
<td></td>
<td>MESHO</td>
<td>Rotated z-coordinates of points on straight line normal to previous row, m</td>
</tr>
<tr>
<td>ZOM(MM, MHTP1)</td>
<td>CALCON</td>
<td>TSONIN</td>
<td>z-coordinates of orthogonal mesh, m</td>
</tr>
<tr>
<td>ZOMBI</td>
<td>INPUTT</td>
<td></td>
<td>Input z-coordinate of intersection of vertical mesh line with hub profile where first change in mesh spacing occurs (MBI), m</td>
</tr>
<tr>
<td>ZOMBO</td>
<td>INPUTT</td>
<td></td>
<td>Input z-coordinate of intersection of vertical mesh line with hub profile where second change in mesh spacing occurs (MBO), m</td>
</tr>
<tr>
<td>ZOMIN</td>
<td>INPUTT</td>
<td></td>
<td>Input z-coordinate of intersection of left boundary of orthogonal mesh with hub profile, m</td>
</tr>
<tr>
<td>ZOMOUT</td>
<td>INPUTT</td>
<td></td>
<td>Input z-coordinate of intersection of right boundary of orthogonal mesh (MM) with hub profile, m</td>
</tr>
<tr>
<td>ZOMROT(MM, MHTP1)</td>
<td>ROTATN</td>
<td></td>
<td>Rotated ZOM array, m</td>
</tr>
<tr>
<td>ZPC(NPPC, NBLPC)</td>
<td>INDCOM</td>
<td></td>
<td>z-coordinates of intersections of percent-chord lines with input blade</td>
</tr>
</tbody>
</table>

100
<table>
<thead>
<tr>
<th>Variable name</th>
<th>COMMON block</th>
<th>Subroutine</th>
<th>Description and comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>ZPCT1(NBLPL)</td>
<td></td>
<td>THETOM</td>
<td>z-coordinates of intersections of percent-chord lines with input blade planes, m</td>
</tr>
<tr>
<td>ZPCT2(NBLPC)</td>
<td></td>
<td>THETOM</td>
<td>z-coordinates along percent-chord lines of alternate mesh, m</td>
</tr>
<tr>
<td>ZRAD(NHUB, MHTP1)</td>
<td></td>
<td>MESH0</td>
<td>z-coordinates of points along lines from input points on hub profile to shroud profile, m</td>
</tr>
<tr>
<td>ZSL(MM, NSL)</td>
<td>SLCOM</td>
<td></td>
<td>z-coordinates of points where streamlines cross vertical mesh lines, m</td>
</tr>
<tr>
<td>ZSLTEM(NSL)</td>
<td>OUTPUT</td>
<td></td>
<td>Temporary storage for calculated values to be put into ZSL array, m</td>
</tr>
<tr>
<td>ZSPL</td>
<td>ILETE</td>
<td></td>
<td>z-coordinate on leading or trailing edge of blade corresponding to a streamline, m</td>
</tr>
<tr>
<td>ZST(NSL)</td>
<td>OUTPUT</td>
<td></td>
<td>z-coordinates of points where station lines cross streamlines, m</td>
</tr>
<tr>
<td>ZTE(NBLPL)</td>
<td>CALCON</td>
<td></td>
<td>z-coordinates of input blade-section points defining trailing edge of blade, m</td>
</tr>
<tr>
<td>ZTEH</td>
<td>PRECAL</td>
<td></td>
<td>z-coordinate of intersection of trailing edge of blade with hub profile, m</td>
</tr>
<tr>
<td>ZTEM(MHTP1, NOSTAT, or 20)</td>
<td>OUTPUT</td>
<td></td>
<td>Temporary storage for values from ZOM array on vertical mesh lines; also temporary storage for values from ZST array along station lines, m</td>
</tr>
<tr>
<td>ZTEOM(MHTP1)</td>
<td>CALCON</td>
<td></td>
<td>z-coordinates of intersections of horizontal mesh lines with blade trailing edge, m</td>
</tr>
<tr>
<td>ZTEOMR(MHTP1)</td>
<td>ROTATN</td>
<td></td>
<td>Rotated ZTEOM array, m</td>
</tr>
<tr>
<td>Variable name</td>
<td>COMMON block</td>
<td>Subroutine</td>
<td>Description and comments</td>
</tr>
<tr>
<td>---------------</td>
<td>--------------</td>
<td>------------</td>
<td>--------------------------</td>
</tr>
<tr>
<td>ZTESL</td>
<td></td>
<td>TSONIN</td>
<td>z-coordinate of intersection of a streamline with trailing edge of blade, m</td>
</tr>
<tr>
<td>ZTET</td>
<td></td>
<td>PRECAL</td>
<td>z-coordinate of intersection of trailing edge of blade with shroud profile, m</td>
</tr>
<tr>
<td>ZTIN</td>
<td>INPUTT</td>
<td></td>
<td>Input z-coordinate of intersection of line on which upstream flow conditions are given with shroud profile, m</td>
</tr>
<tr>
<td>ZTIP(NTIP)</td>
<td>INPUTT</td>
<td></td>
<td>Input z-coordinates of points defining shroud or top boundary of flow channel, m</td>
</tr>
<tr>
<td>ZTOUT</td>
<td>INPUTT</td>
<td></td>
<td>Input z-coordinate of intersection of line on which downstream flow conditions are given with shroud profile, m</td>
</tr>
<tr>
<td>ZTROT(NTIP)</td>
<td>ROTATN</td>
<td></td>
<td>Rotated ZTIP array, m</td>
</tr>
<tr>
<td>ZTST(NOSTAT)</td>
<td>INPUTT</td>
<td></td>
<td>Input z-coordinates of intersections of hub-shroud output station lines with shroud profile, m</td>
</tr>
<tr>
<td>ZTTAN1</td>
<td>TSONIN</td>
<td></td>
<td>m-coordinate at upper blade surface trailing-edge tangency point, m</td>
</tr>
<tr>
<td>ZTTAN2</td>
<td>TSONIN</td>
<td></td>
<td>m-coordinate at lower blade surface trailing-edge tangency point, m</td>
</tr>
</tbody>
</table>
PROGRAM LISTING

C--MERIDL PROGRAM -- UPDATED TO JULY 1, 1977
C--REPORTS -- NASA TND-8433 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) 33-4000

COMMON NREAD, NWRT1, ITER, NWRT2, NWRT3, NWRT4, NWRT5, NWRT6
COMMON/INPUT/GAM, AR, MSFL, OMEGA, REDFAC, VELTOL, FNEW, DNEW, MBI, MRO,
  MM, MHT, NBL, NHub, WP, FIN, NOUT, NBLPL, NPPP, NOSTAT, NSSL, NLOSS,
  LSPR, LPTL, LAMVT, LROT, LBLAD, LPTETAN, ANGROT, IMESH, ISLINE,
  ISTATL, IPILOT, ISPsUP, ITSON, IDBBUG, ZMIN, ZOMIN, ZOMBT, ZOMBO, ZOMOUT,
  RMIN, RMOBI, RBMO, ROMOUT, ZMIN, ZTIN, ZHOUT, ZHOUT, ZHOUT, RHOUT,
  RTOUT, TITLE(2), ZHUB(50), NHub(50), WP(50), RTIP(50), SPIN(50),
  RADIN(50), TIP(50), PRIP(50), LAMIN(50), VTHIN(50), SFOUT(50),
  RADOUT(50), PROP(50), LOSOUT(50), LAMOUT(50), VTHOUT(50),
  BETATE(50), BETATE(50), ZHST(50), ZTST(50), BSTH(50), RTST(50),
  FLFR(50), PERCRD(50), PERLOS(50), ZBL(50, 50), RBL(50, 50),
  THBL(50, 50), TBTL(50, 50), TSTL(50, 50), TH1BL(50, 50), TH2BL(50, 50),
COMMON/CALCON/MM1, MHTP1, CP, EXPON, TGROG, PITCH, RLEH, RLET, R"EH, RTET,
  ZLE(50), RLE(50), ZTE(50), RTE(50), ZLEOM(101), RLEOM(101),
  SLEOM(101), TLEOM(101), ZTEOM(101), RTEOM(101), STEOM(101),
  TTMEM(101), ILE(101), IZE(101), ZOM(100, 101), ROM(100, 101),
  SMO(100, 101), TMO(100, 101), ZTM(100, 101), RTM(100, 101),
  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),
  WSUBS(100, 101), WSUBP(100, 101), WSUBZ(100, 101), WSUBR(100, 101),
  WSUBM(100, 101), WTM(100, 101), WTH(100, 101), W(100, 101),
  ALPHA(100, 101), BETA(100, 101), WWCR(100, 101), CURV(100, 101),
  WLSURF(100, 101), WTSURF(100, 101), CAMP(100, 101), SAMP(100, 101),
  RHOAV(100, 101), DELRHO(100, 101), PT(100, 101), DFDM(100, 101),
  XION(100, 101), ZETOM(100, 101), DLDO(100, 101),
COMMON/SLOCOM/ILS(50), ITS(50), ZSL(100, 50), RSL(100, 50), MSL(100, 50),
  WZSL(100, 50), WSL(100, 50), MWSL(100, 50), WTHSL(100, 50),
  ALPSL(100, 50), BETSL(100, 50), WSL(100, 50), WWCRLS(100, 50),
  CURVSL(100, 50), WLSSL(100, 50), WTSSSL(100, 50),
COMMON/ROTATN/HRHOT(50), HRHOT(50), ZTROT(50), RTROT(50),
  ZLEOMR(101), RLEOMR(101), ZTMOMR(101), RTMOMR(101),
  ZBLROT(50, 50), RBLROT(50, 50), ZOMR(100, 101), RDOMR(100, 101),
COMMON/INDCOM/IDLCPLC, NP, ZPC(51, 51), RPC(51, 51), TPC(51, 51),
  THPC(51, 51), DTHDR(51, 51), DTHDR(51, 51), BTHL(101), BTHTE(101),
  RTHFLE(101), BTHFTE(101),
COMMON/PLTCOM/PLNG, ZRNG, RBNG, RTRNG, ZHPLT(100), RHPLT(100),
  ZSPLT(100), RSPLT(100), ZLPBLT(100), RLPBLT(100), ZTPLT(100),
  RP(100)
10 IEND = -1
ITER = 0

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 MEPLLOT
C
C--CALCULATE COEFFICIENTS, SOLVE DIFFERENTIAL EQUATIONS FOR STREAM
C--FUNCTION, AND COMPUTE NEW VELOCITIES AND DENSITIES
CALL INIT
20 ITER = ITR+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,ITEND,NWRT1,NWRT2,NWRT3,NWRT4,NWRT5,NWRT6
COMMON/INPUT/GAM,AR,MSFL,OMEGA,REDFAC,VELTOL,FNEW,DNEW,MBI,MBO,
1 MM,MHT,NBL,RHUB,NTIP,NIN,NOUT,MBLPL,NPPP,NOSTAT,NSL,NLOSS,
2 LSFR,LTPL,LAMVT,LROT,LBLAD,LETEAN,ANGROT,IMESH,ISLINE,
3 ISTATL,IPLOT,ISUPER,ITSON,TDEBUG,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),SPOUT(50),
DIMENSION RADOUT(50),PROP(50),LOSOUT(50),LAMOUT(50),VTHOUT(50),
          BETALE(50),BETATE(50),ZHST(50),RSTHT(50),RTST(50),
          PERBL(50),PERBL(50),PERLS(50),ZBL(50,50),REL(50,50),
          THBL(50,50),THBL(50,50),THBL(50,50),ZTHBL(50,50),TH2BL(50,50),
          COMMON/CALCON/MMM1,MHTP1,CP,EXPON,TGROG,PITCH,PLEH,PLET,RTGH,
          ZLE(50),ZLE(50),ZTE(50),ZTE(50),ZLEOM(100),ZLEOM(100),
          SLEOM(100),SLEOM(100),STROM(100),STROM(100),
          TTHEOM(100),TILEOM(100),TILE(100),TOM(100),TOM(100),
          BTHEOM(100),BTHEOM(100),BTHEOM(100),BTHEOM(100),
          DTHT(100,100),PLOSS(100,100),CPHI(100,100),CPHI(100,100),
          DIST(50),DTDS(50),ANG(50)
          REAL MSFL,LAMIN,LAMOUT,LOSOUT

C
C--READ AND PRINT INPUT DATA
C
NREAD = 5
NWRIT = 6
NWRIT1 = 6
NWRIT2 = 6
NWRIT3 = 6
NWRIT4 = 6
NWRIT5 = 6
NWRIT6 = 6
10 READ (NREAD,1050) (TITLEI(I),I=1,2)
  WRITE(NWRIT,1000)
  WRITE(NWRIT,1060) (TITLEI(I),I=1,2)
  READ (NREAD,1030) GAM,AR,MSFL,OMEGA,REDFAC,VELTOL,FNEW,DNEW
  IF (GAM.NE.0.) WRITE(NWRIT,1100)
  IF (GAM.EQ.0.) WRITE(NWRIT,1160)
  IF (REDFAC.LE.0.) REDFAC=1.0
  IF (VELTOL.LE.0.) VELTOL=0.5
  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.0.0.OR.DNEW.LT.0.0) WRITE(NWRIT,1100) VELTOL
  WRITE(NWRIT,1110)
  READ (NREAD,1010) MBI,MBO,MM,HHT,NBL,NHUB,NTIP,NIN,NOUT,NBLPL,
    NPPP,NOSTAT,NSL,NLOSS
  IF (MBI.EQ.0) NBL=1
  WRITE(NWRIT,1020) MBI,MBO,MM,HHT,NBL,NHUB,NTIP,NIN,NOUT,NBLPL,
    NPPP,NOSTAT,NSL,NLOSS
  WRITE(NWRIT,1120)
  READ (NREAD,1010) LSFR,LTPL,LAMVT,LROT,LBLAD,LETEAN
  WRITE(NWRIT,1120) LSFR,LTPL,LAMVT,LROT,LBLAD,LETEAN
  ANGROT = 0.
  IF (LROT.EQ.0) GO TO 15
  WRITE(NWRIT,1115)
  READ (NREAD,1030) ANGROT
  WRITE(NWRIT,1040) ANGROT
  ANGROT = ANGROT/57.295780
  WRITE(NWRIT,1130)
  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, PHIN, RTIN
WRITE (NWRIT, 1040) ZHIN, ZTIN, PHIN, RTIN
IF (LSFR.EQ.1) GO TO 20
WRITE (NWRIT, 1190)
READ (NREAD, 1030) (SPIN(I), I=1, NIN)
WRITE (NWRIT, 1040) (SPIN(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)
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)
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)
7C 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
8C WRITE (NWRIT, 1290)
READ (NREAD, 1030) (LOSOUT(I), I=1, NOUT)
WRITE (NWRIT, 1040) (LOSOUT(I), I=1, NOUT)
9C 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)
    WRITE (NWRIT,1330)
    DO 120 JN=I,NBLPL
    READ (NREAD,1030) (ZBL(IN,JN),IN=1,NPPP)
120 WRITE (NWRIT,1320)
    DO 130 JN=I,NBLPL
    READ (NREAD,1030) (RBL(IN,JN),IN=1,NPPP)
    WRITE (NWRIT,1340)
    DO 130 JN=I,NBLPL
    READ (NREAD,1030) (RBL(IN,JN),IN=1,NPPP)
    IF (LBLAD.EQ.2) GO TO 150
    WRITE (NWRIT,1340)
    DO 140 JN=I,NBLPL
    READ (NREAD,1030) (THBL(IN,JN),IN=I,NPPP)
    WRITE (NWRIT,1350)
    DO 140 JN=I,NBLPL
    READ (NREAD,1030) (THBL(IN,JN),IN=I,NPPP)
    IF (LBLAD.EQ.1) GO TO 146
    WRITE (NWRIT,1350)
    DO 145 JN=I,NBLPL
    READ (NREAD,1030) (TNBL(IN,JN),IN=I,NPPP)
    WRITE (NWRIT,1360)
    DO 145 JN=I,NBLPL
    READ (NREAD,1030) (TNBL(IN,JN),IN=I,NPPP)
    IF (LETIEAN.EQ.0) GO TO 157
    WRITE (NWRIT,1360)
    DO 152 JN=I,NBLPL
    READ (NREAD,1030) (TH1BL(IN,JN),IN=I,NPPP)
    WRITE (NWRIT,1370)
    DO 152 JN=I,NBLPL
    READ (NREAD,1030) (TH1BL(IN,JN),IN=I,NPPP)
    IF (LETIEAN.EQ.0) GO TO 156
    WRITE (NWRIT,1370)
    READ (NREAD,1030) (TH2BL(IN,JN),IN=I,NPPP)
    WRITE (NWRIT,1380)
    READ (NREAD,1030) (TH2BL(IN,JN),IN=I,NPPP)
146 WRITE (NWRIT,1352)
    DO 148 JN=I,NBLPL
    READ (NREAD,1030) (TTBL(IN,JN),IN=I,NPPP)
    WRITE (NWRIT,1354)
    DO 148 JN=I,NBLPL
    READ (NREAD,1030) (TTBL(IN,JN),IN=I,NPPP)
    WRITE (NWRIT,1356)
    READ (NREAD,1030) (TTBL(IN,JN),IN=I,NPPP)
150 IF (LETIEAN.EQ.0) GO TO 157
    WRITE (NWRIT,1356)
    READ (NREAD,1030) (BETALE(JN),JN=I,NBLPL)
    WRITE (NWRIT,1360)
    READ (NREAD,1030) (BETALE(JN),JN=I,NBLPL)
    WRITE (NWRIT,1365)
    READ (NREAD,1030) (BETATE(JN),JN=I,NBLPL)
    WRITE (NWRIT,1370)
    READ (NREAD,1030) (BETATE(JN),JN=I,NBLPL)
156 IF (LETIEAN.EQ.0) GO TO 157
    WRITE (NWRIT,1358)
    READ (NREAD,1030) (ZHST(I),I=I,NOSTAT)
    WRITE (NWRIT,1040) (ZHST(I),I=I,NOSTAT)
    IF (LROT.EQ.0) GO TO 158
    WRITE (NWRIT,1360)
    READ (NREAD,1030) (ZHST(I),I=I,NOSTAT)
    WRITE (NWRIT,1040) (ZHST(I),I=I,NOSTAT)
    IF (LROT.EQ.0) GO TO 158
    WRITE (NWRIT,1365)
    READ (NREAD,1030) (RHST(I),I=I,NOSTAT)
    WRITE (NWRIT,1040) (RHST(I),I=I,NOSTAT)
158 WRITE (NWRIT,1370)
    READ (NREAD,1030) (ZTST(I),I=I,NOSTAT)
    WRITE (NWRIT,1040) (ZTST(I),I=I,NOSTAT)
    IF (LROT.EQ.0) GO TO 160
    WRITE (NWRIT,1375)
    READ (NREAD,1030) (ZTST(I),I=I,NOSTAT)
    WRITE (NWRIT,1040) (ZTST(I),I=I,NOSTAT)
160 IF (LSL.EQ.0) GO TO 165
    WRITE (NWRIT,1380)
    READ (NREAD,1030) (FLFR(I),I=I,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.10C. AND. NHUB.LE.50. AND. NTIP.LE.50. AND.
1 NIN.LE.50. AND. NOUT.LE.50. AND. NBLPL.LE.50. AND. NPPP.LE.50. AND.
2 NOSTAT.LE.50. AND. NSL.LE.50. AND. NLOSS.LE.50. AND. LSFR.GE.0. AND.
3 LSFR.LE.1. AND. LTLPL.GE.0. AND. LTLPL.LE.1. AND. LAMVT.GE.0. AND.
4 LAMVT.LE.1. AND. LROT.GE.0. AND. LROT.LE.1. AND. LBLAD.GE.0. AND.
5 LBLAD.LE.2. AND. LTEAN.GE.0. AND. LTEAN.LE.1) GO TO 180
WRITE (NWRIT, 1400)
STOP

C--CALCULATE MISCELLANEOUS CONSTANTS
C
180 MMM1 = MM-1
MHTPI = MHT+1
EXPO = 1./(GAM-1.)
CP = AR*GAM*EXPO
TGRO = 2.*GAM*AR/(GAM+1.)
PITCH = 2.*3.1415927/FLOAT(NBL)
MSFL = MSFL/FLOAT(NBL)

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 SPLINT(ZTIP, RTIP, NTIP, ZTIN, 1, RTIN, DYDX, D2YDX2)
CALL SPLINT(ZTOUT, RTOUT, D2YDX2)

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--CALCULATE ESTIMATED UPSTREAM AND DOWNSTREAM VALUES OF
C--RADIUS, IF STREAM FUNCTION WAS GIVEN AS INPUT
C
230 DO 240 J=1,NIN

108
240 RADIN(J) = SQRT(RHIN**2+SPIN(J)*RINSQ)
250 RADOUT(J) = SQRT(RHOUT**2+SPOUT(J)*ROUTSQ)

C--CALCULATE ESTIMATED UPSTREAM AND DOWNSTREAM TANGENTIAL VELOCITIES,
C--IF WHIRL WAS GIVEN AS INPUT

260 IF (LAMVT.EQ.1) GO TO 290
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--CALCULATE ESTIMATED UPSTREAM AND DOWNSTREAM WHIRL,
C--IF TANGENTIAL VELOCITY WAS GIVEN AS INPUT

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--CALCULATE TANGENTIAL THICKNESS, IF NORMAL THICKNESS GIVEN AS INPUT

320 IF (MBI.EQ.0) RETURN
IF (LBLAD.EQ.1) GO TO 370
IF (LBLAD.EQ.2) GO TO 390
DIST(1) = 0.
DO 330 JN=1,NBLPL
DO 330 IN=2,NPPP
330 DIST(IN) = DIST(IN-1) + SQRT((ZBL(IN,JN) - ZBL(IN-1,JN))**2 + (RBL(IN,JN) - RBL(IN-1,JN))**2)
IF (LETEAN.EQ.1) GO TO 340
CALL SPLINE(DIST,THBL(I,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(I,JN),NPPP,DTHDSL,DTHDST,DTDS,ANG)
350 DO 360 IN=1,NPPP
ANG(IN) = ATAN(RBL(IN,JN)*DTDS(IN))
360 TTBL(IN,JN) = TBL(IN,JN)/COS(ANG(IN)) RBL(IN,JN)

C--CALCULATE BLADE SURFACE THETA COORDINATES, IF THEY ARE NOT
C--GIVEN AS INPUT

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--CALCULATE MEAN CAMBER LINE THETA COORDINATES AND TANGENTIAL THICKNESS
C--IF SURFACE THETA COORDINATES GIVEN AS INPUT

390 DO 400 JN=1,NBLPL
DO 400 IN=1,NPPP
THBL(IN,JN) = (TH1BL(IN,JN)+TH2BL(IN,JN))/2.
OC TBL(IN,JN) = TH1BL(IN,JN) - TH2BL(IN,JN)
RETURN
C
C--FORMAT STATEMENTS
C
1000 FORMAT (1H//5CX,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 (1HI)
1100 FORMAT (///4X,20HGENERAL INPUT DATA/7X,3HGM,14X,2HAR,13X,
14HMSFL,11X,5HOME,11X,6HREDFAC,10X,6HVELTOL,10X,4HFNEW,11X,
24HNEW)
1102 FORMAT (///4X,20HGENERAL INPUT DATA/7X,3HGM,13X,3HRHO,13X,
14HMSFL,11X,5HOME,11X,6HREDFAC,10X,6HVELTOL,10X,4HFNEW,11X,
24HNEW)
1105 FORMAT (7X,7HVELTOL HAS BEEN REduced BY THE MINIMUM OF FN
1EW OR DNEW TO =,8X,6HVELTOL/8X,G16.7)
1110 FORMAT (103H MBI MBO MM MHT NBL NHUB NTIP
1NIN NOUT NBLPL NPPP NOSTAT NSL NLOSS)
1120 FORMAT (47HLSPR LTPL LAMVT LROT LBLAD LETEAN)
1125 FORMAT (6X,6HANGROT)
1130 FORMAT (///4X,29HHUB AND SHROUD INPUT DATA/7X,5HZOMIN,11X,
5HZOMBT,11X,5HANGB,10X,6HZOMOUT,11X,5HROMIN,11X,5HROMBI,11X,
25HROMBO,10X,6HRCMOUT)
1140 FORMAT (7X,11Hzhub ARRAY)
1150 FORMAT (7X,11Hrhub ARRAY)
1160 FORMAT (7X,11HrTip ARRAY)
1170 FORMAT (7X,11HrTip ARRAY)
1180 FORMAT (///4X,21HUPSTREAM INPUT DATA/7X,4HZhin,11X,4HZtin,
11X,4HRhin,11X,4HRtin)
1190 FORMAT (7X,11Hspin ARRAY)
1200 FORMAT (7X,12Hradin ARRAY)
1210 FORMAT (7X,10HTip ARRAY)
1220 FORMAT (7X,11HrTip ARRAY)
1230 FORMAT (7X,12HLamin ARRAY)
1240 FORMAT (7X,12Htthin ARRAY)
1250 FORMAT (///4X,23HDOWNSTREAM INPUT DATA/7X,5HZhout,10X,5HZtout,
10X,5HRHout,10X,5HRtout)
1260 FORMAT (7X,12Hspout ARRAY)
1270 FORMAT (7X,13Hradout ARRAY)
1280 FORMAT (7X,11HPROP ARRAY)
1290 FORMAT (7X,13Hlosout ARRAY)
1300 FORMAT (7X,13HLamout ARRAY)
1310 FORMAT (7X,13Hythout ARRAY)
1320 FORMAT (///4X,54HBLADE MEAN CAMBER LINE AND THICKNESS INPUT
1DATA/7X,10H2BL ARRAY)
1330 FORMAT (7X,10HrBL ARRAY)
1340 FORMAT (7X,11Hthel BL ARRAY)
1350 FORMAT (7X,11Htnbl ARRAY)
1352 FORMAT (7X,11HTTBL ARRAY)
1354 FORMAT (7X,12Hth1BL ARRAY)
1356 FORMAT (7X,12Hth2BL ARRAY)
1358 FORMAT (7X,13Hbetal ARRAY)
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/GA_,AR,MSFL,OMEGA,RED PAC,VEI TOL,FNEW,DNEW,MRI,MRO,
1 MM,MHT,NBL,NHUB,NTP,NIN,NOUT,NBLPL,NPPP,NOSTAT,NSL,NLOSS,
2 LSFR,LTPL,LAMVT,LROT,LBLAD,LETEN,ANGROT,IMESH,ISLINE,
3 ISTATL,IPLOT,ISUPER,ITSON,ITDEBUG,ZMIN,ZOMBI,ZOMBO,ZOMOUT,
4 ROMIN,ROMBI,ECMRO,ROMOUT,ZHIN,ztIN,ZHOUT,ZTOUT,RHIN,RTIN,RHOUT,
5 RTOUT,TITLEI(20),ZHUB(50),RHUB(50),ZTIP(50),RTIP(50),SPIN(50),
6 RADIN(50),TIP(50),PRIP(50),LAMIN(50),VTHIN(50),SPOUT(50),
7 RADOUT(50),PROP(50),LOSOUT(50),LAMOUT(50),VTHOUT(50),
8 BETALF(50),BETATE(50),ZHST(50),ZTST(50),RHST(50),RTST(50),
9 PLFR(50),PERCRD(50),PERLOS(50),ZBL(50,50),RBL(50,50),
10 THBL(50,50),TNBL(50,50),TTBL(50,50),TH1BL(50,50),TH2BL(50,50),
11 COMMON/CALCON/MMM,CP,EXPON,THGROG,PITCH,REH,RLE,RTK
12 REAL MBL,(50,50),ZBL(50,50),ZTBL(50,50),RTHBL(50,50),RTH2BL(50,50),
13 COMMON/INPUTT/MMM,CP,EXPON,THGROG,PITCH,REH,RLE,RTK
14 REAL MBL,(50,50),ZBL(50,50),ZTBL(50,50),RTHBL(50,50),RTH2BL(50,50),
15 1 RTHBL(50,50),RTH2BL(50,50),RTH4BL(50,50),
16 PLTX(101),PLTY(101),DYDX(101),D2YDX2(101),
17 TITL1(9),TITL2(9),TITL3(5),TITL4(9),TITL5(8),TITL6(9),TITL7(6),
18 TITL8(9),TITL10(13),TITL11(6),TITL12(6),TITL13(4),
19 TITL14(2),TITL15(5),TITL16(5),TITL17(5),TITL18(5),TITL19(5),
20 TITL20(5),TITL21(5),TITL22(5),TITL24(10),TITL25(9),TITL26(7),
21 TITL27(2),TITL28(2)
REAL MBL,LAMIN,LAMOUT,LOSOUT,LRNG
DATA TITL1/"ILN","AT","BSOL","UTE ","TOTA","IL TE","MPER","ATUR"/
1,"E "/
DATA TITL2/"INLE","AT","BSOL","UTE ","TOTA","IL TE","MPER","ATUR"/
1,"E "/
111
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 TITLS/'OUTL','ET A','BSOL','UTE ','TOTA','L PR','ESSU','RE '
1/
DATA TITL5/'OUTL','ET A','BSOL','UTE ','WHIR','L '/
DATA TITL6/'OUTL','ET A','BSOL','UTE ','TANG','ENTI','AL V','ELOC'
1,'ITY'/
DATA TITL10/'INPU','T BL','ADE ','SECT','IONS','$C1$','L2PR',
1'OM Z','BL ',''PBL',' T','HE ','L T','NBL '/
DATA TITL11/'B','LADE ','SECT','TION','NO. ','XXXX'/
DATA TITL12/'COMB','INED',' BLA','DE S','ECTI','IONS '/
DATA TITL13/'STRE','AM P','ONCT','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','NOUT'/
DATA TITL22/'INPU','T AR','RAY ','- VT','HOUT'/
DATA TITL23/'BLAD','E S','ECTI','ON ','MERI','DION','AL '
1'COOR','DINA ','TE '/
DATA TITL24/'TANG','ENTI','AL C','ORDER','INAT','E - ','RADI',
1'US,T','HETA'/
DATA TITL25/'PRES','SURE',' LOS','S DI','STRI','BUTI','ON '/
DATA TITL26/'PERC','OS '/
DATA TITL27/'PERC','RD '/
DATA TITL28/'PERC','OS '/
DATA BLNK'/ '
DATA SYM/X'/
IF (IPLT.LE.0) RETURN

C--PLOT TITLE ON MICROFILM
C
CALL LRSIZE (0.0, 20.0, 0.0, 10.0)
CALL LRCHSZ(4)
CALL LRLEGN(TITL3, 80, 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--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 = AMINI(SFIN(1),SFOUT(1))
TRNG = AMAX1(SFIN(NIN),SFOUT(NOUT))
GO TO 40
20   PLTY(1) = RADIN(1)
PLTY(1C1) = 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 LRLEGN(TITL1, 36, 1, 0, 2.0, 0.1)
CALL LRGRID(1, 11.0, 11.0)
CALL LRCHSZ(4)
CALL LRLEGN(TITL1, 36, 1, 0, 2.0, 0.1)
CALL LRCHSZ(2)
CALL LRLEGN(TITL15, 20, 0, 4.0, 1.0, 0.1)
IF (LSFR.EQ.0) CALL LRLEGN(TITL13, 16, 1, 0.2, 4.2, 0.1)
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, 1, 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(PHIP, SFIN, NIN, 4, SYM, 1.0)
IF (LSFR.EQ.1) CALL LRCURV(PHIP, 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)
113
GO TO 110

C--PLOT INLET ABSOLUTE TANGENTIAL VELOCITY

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(TITL18,20,0.4,0.3,0.0)
   IF (LSFR.EQ.0) CALL LRLEGN(TITL13,16,1.0,2.4,2.0)
   IF (LSFR.EQ.1) CALL LRLEGN(TITL14,16,0.2,4.7,0.0)
   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)_RTNSQ)
   100 IF (LSFR.EQ.1) PLTX(J)=PLTX(J)/PLTY(J)
   CALL LPCURV(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--PREPARE FOR PLOTTING OF OUTLET CONDITIONS

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--PLOT OUTLET ABSOLUTE TOTAL PRESSURE

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(TITL19,20,0.4,0.3,0.0)
   IF (LSFR.EQ.0) CALL LRLEGN(TITL13,16,1.0,2.4,2.0)
   IF (LSFR.EQ.1) CALL LRLEGN(TITL14,16,0.2,4.7,0.0)
   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--PLOT OUTLET ABSOLUTE TOTAL PRESSURE LOSS

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
CALL LRCHSZ
CALL LRLEGN
IF (LSFR.EQ.0) CALL LRLEGN(TITL13,16,1,0,2,4,2,0)
IF (LSFR.EQ.1) CALL LPLEGN(TITL14,8,1,0,2,4,7,0)
CALL LRCHSZ(4)
CALL LRCURV(PLTX,PLTY,101,2,SYM,0.0)
IF (LSFR.EQ.0) CALL LRCURV(LOSOUT,SPOUT,NOUT,4,SYM,1.0)
IF (LSFR.EQ.1) CALL LRCURV(LOSOUT,RADOUT,NOUT,4,SYM,1.0)

19C CALL LRCHSZ(0)
IF (MBT.EQ.0) GO TO 240
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(TITL21,20,0,4.0,1.3,0.3)
IF (LSFR.EQ.0) CALL LRLEGN(TITL13,16,1,0,2,4,2,0)
IF (LSFR.EQ.1) CALL LRLEGN(TITL14,8,1,0,2,4,7,0)
CALL LRCHSZ(4)
CALL LRCURV(PLTX,PLTY,101,2,SYM,0.0)
IF (LSFR.EQ.0) CALL LRCURV(LAMOUT,SPOUT,NOUT,4,SYM,1.0)
IF (LSFR.EQ.1) CALL LRCURV(LAMOUT,RADOUT,NOUT,4,SYM,1.0)
GO TO 240
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)
IF (LSFR.EQ.1) CALL LRLEGN(TITL14,8,1,0,2,4,7,0)
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,SPOUT,NOUT,4,SYM,1.0)
IF (LSFR.EQ.1) CALL LRCURV(VTHOUT,RADOUT,NOUT,4,SYM,1.0)
C--PLOT PERCRD AND PERLOS
C
240 IF (MLOSS.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(TITL26,28,9,1.7,.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--PLOT INPUT BLADE SECTIONS
C
248 IF (MBI.EQ.0) RETURN
C--CALCULATE BLADE SECTION PLOT COORDINATES ALONG MERIDIONAL PLANE
DO 250 JN=I,NBLPL
MBL(I,JN) = ZBL(I,JN)
DO 250 IN=2,NPPP
250 MBL(IN,JN) = MBL(IN-1,JN) + SQRT((ZBL(IN,JN) - ZBL(IN-1,JN))**2 + (RBL(IN,JN) - RBL(IN-1,JN))**2)
C--CALCULATE TANGENTIAL PLOT COORDINATES
DO 260 JN=I,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)
RTH3BL(IN,JN) = RTHBL(IN,JN) + DELRTH
RTH4BL(IN,JN) = RTH3BL(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))
BRNG = AMAX1(BRNG,MBL(NPPP,JN))
DO 270 IN=1,NPPP
BRNG = AMIN1(BRNG,RTH2BL(IN,JN))
270 TRNG = AMAX1(TRNG,RTH3BL(IN,JN))
RTEM = BRNG
DELLR = BRNG - LRNG
DELB = TRNG - BRNG
DELRNG = AMAX1(DELLR,DELB)
RRNG = LRNG + DELRNG
TRNG = BRNG+DEIRNG
CALL LRANGE(LRNG,RRNG,BRNG,TRNG)
C--PLOT BLADE SECTIONS AND SHOW SOLIDITY
CALL LRCHSZ(4)
CALL LRLEGN(TITL1C, 52, 0, 2.7, C.7, 0.0)
DO 280 JN=1,NBLPL
CALL LRCHSZ(3)
CALL LRCHSZ(2)
CALL LRLEGN(TITL1D, 36, 1, 0.2, 3.3, 0.0)
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), 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), RTH3BL(1,JN), NPPP, 2, SYM, 0.0)
CALL LRCURV(MBL(1,JN), RTH4BL(1,JN), NPPP, 2, SYM, 0.0)
CALL LRCURV(MBL(1,JN), RTH5BL(1,JN), NPPP, 2, SYM, 0.0)
IF (LBLAD.EQ.2) GO TO 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)
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)
GO TO 280
280 CONTINUE
C--CALCULATE RANGE OF PLOT, AND SET UP FOR PLOT OF MULTIPLE
C--BLADE SECTIONS
RRNG = RTEM
TRNG = RTH1BL(NPPP,NBLPL)
DO 290 JN=1,NBLPL
DO 290 IN=1,NPPP
290 TRNG = AMAX1(TRNG, RTH1BL(IN,JN))
DELT = TRNG-BRNG
DELRNG = AMAX1(DELLR,DELT)
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), 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), RTH3BL(1,JN), NPPP, 2, SYM, 0.0)
300 CALL LRCURV(MBL(1,JN), RTH4BL(1,JN), NPPP, 2, SYM, EOP)
CALL LRCURV(ZBL,RBL,0,1,SYM,1.0)
CALL LRCHSZ(0)
RETURN
END
SUBROUTINE MESHO

C -- MESHO CALCULATES COORDINATES OF AN ORTHOGONAL MESH
C -- COVERING THE SOLUTION REGION

COMMON /INPUTT/GAM, AR, MSFL, OMFGA, REDFAC, VELTOL, FNEW, DNEW, MBI, MBO,
1 LSFR, LTLT, LAMVT, LROT, LBLAD, LTET, ANGROT, IMESH, ISLINE,
2 ISTATL, IFLOT, ISUPER, ISON, IDEBUG, ZOMIN, ZOMBI, ZOMBO, ZOMOUT,
3 ROMIN, ROMB1, ROMBO, RONSOUT, ZHUB, ZTIP, ZHUBIN, ZHOUR, ZOUT, RHIN, RTIN, RHOUT,
4 RTOUT, TITLE(20), ZHUB(50), ZHUB(50), ZHUB(50), ZHUB(50), ZHUB(50), ZHUB(50),
5 RADIN(50), TIP(50), PRIP(50), LAMIN(50), VTHIN(50), SOUT(50),
6 RADOUT(50), TIP(50), PROTO(50), LCONSOUT(50), LAMOUT(50), VTHOUT(50),
7 BETAM(50), BETATE(50), ZHST(50), ZST(50), RHST(50), RTST(50),
8 FLFRF(50), PERC(50), PERLOS(50), ZBL(50,50), ZBL(50,50),
9 RADIN(50), TIP(50), RADOUT(50), PROTO(50), VTHIN(50),
10 BETA1(50), BETA2(50), ZHST(50), ZST(50), RHST(50), RTST(50),
11 FLFRF(50), PERC(50), PERLOS(50), ZBL(50,50), ZBL(50,50),
12 RBL(50,50), RBL(50,50), ZTST(50), ZTST(50), RHST(50), RTST(50),
13 FLFRF(50), PERC(50), PERLOS(50), ZBL(50,50), ZBL(50,50),
14 RBL(50,50), RBL(50,50), ZTST(50), ZTST(50), RHST(50), RTST(50),
15 CONTINUE

C -- ROTATE HUB AND TIP COORDINATES
CALL ROTATE (ANGROT, ZHUB, RHUB, NHUB, 1, 5C, 1, ZHROT, RHROT)
CALL ROTATE (ANGROT, ZTIP, RHUB, NTIP, 1, 5C, 1, ZTROT, RTROT)

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, MHTPI) = ZTROT (I)
RRAD (I, MHTPI) = 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, MHTPI) = ZTROT (1) + FLOAT (I - 1) * DELT
CALL SPLINT (ZTROT, RTROT, NTIP, ZRAD (1, MHTPI), NMAX, RRAD (1, MHTPI), AAA, 1
BBB)
GO TO 15
8 DO 10 I = 1, NMAX
ZRAD (I, 1) = ZHROT (I)
RRAD (I, 1) = RHROT (I)
ZRAD (I, MHTPI) = ZTROT (I)
10 RRAD (I, MHTPI) = RTROT (I)
15 CONTINUE

C -- FILL ZRAD AND RRAD ARRAYS FROM HUB TO TIP
DO 20 I=1,NMAX
DELZ = (ZRAD(I,MHTPI)-ZRAD(I,1))/FLOAT(MHT)
DELR = (RRAD(I,MHTPI)-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 ZOMPRT(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) MBOPI=2
DO 60 I=MBOPI,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),
1 SLOM,BBB)
DO 70 I=1,MM
PHI = ATAN(SLOM(I))*ANGROT
Sphi(I,1) = SIN(PHI)
Cphi(I,1) = COS(PHI)
70
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(I,J),RRAD(I,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 SPLINT(ZOMROT(I,J),MM,ROMROT(I,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
CALL ROTATE(-ANGROT,ZOMROT,ROMROT,MM,MHTPI,100,101,ZOM,ROM)
RETURN
END

SUBROUTINE MEPLT

C C--MEPLT PLOTS THE BLADE GEOMETRY AND THE GENERATED ORTHOGONAL MESH

C COMMON/INPUTT/GAM,AR,MSFL,OMEGA,REDFAC,VELTOL,FNEW,DNEW,MBI,MBO,
1 MM,NHT,NBL,NHUB,Ntip,NOUT,NBLPL,NPPP,NOSTAT,NSL,NLOSS,
2 LSPR,LTPL,LAMVT,LROT,LBLAD,LETEAN,ANGROT,IMESH,ISLINE,
3 ISTAT,IPLOT,ISUPER,ITSON,IDEBUG,CMIN,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),RTP(50),SPIN(50),
6 RADIN(50),TIP(50),PRIP(50),LAMIN(50),VTIP(50),STIP(50),
7 RADOUT(50),PRCP(50),LOSSOUT(50),LAMOUT(50),VTHOUT(50),
8 BETALE(50),BETATE(50),ZSTH(50),ZTST(50),PSTH(50),RTST(50),
9 PFR(50),PERCRD(50),PERLOS(50),ZBL(50,50),RBL(50,50),
1 THBL(50,50),TNBL(50,50),TTBL(50,50),THBL(50,50),TH2BL(50,50)

COMMON/CALCON/MMM1,MHTPI,CP,EXPON,TGROG,PITCH,BLEH,REL,RET,
1 ZLE(50),RLF(50),ZTE(50),RTE(50),ZLEOM(101),RLEOM(101),
2 SLEOM(101),TLEOM(101),ZTEOM(101),RTEOM(101),STEO(101),
3 THTEOM(101),TEL(101),TTE(101),ZOM(100,101),ROM(100,101),
4 SOM(100,101),TOM(100,101),BTH(100,101),DTHDS(100,101),
5 DTHDT(100,101),LOSS(100,101),CPHI(100,101),SPHI(100,101),

COMMON/PLTCOM/ZLRNG,ZRRNG,RBBNG,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),
1 ITERM(101)

DATA TITL1/'HUB','SHR','OUD','AND','BLA','DE B','FOWN','ARIE'
1.'S&C1','$RII','N ME','RTI','WAL','PLA','NE /
DATA TITL2/'ORT','ON ','LANE'/'
DATA TITL3/'Z D','IREC','TION'/
DATA TITL4/'R D','IREC','TION'/
DATA SYM/'X'/

120
DATA SYN/'O'/
IF (IPLT LE.0) RETURN

C--OBTAIN PLOT BOUNDARIES, AND SCALE THE PLOT
CALL PTBDRY

C--PLOT BLADE GEOMETRY AND PLOT ORTHOGONAL MESH
CALL LRMNGN(1.0,1.0,2.0,1.0)
CALL LRANGE(ZLRNG,ZRRNG,RBRNG,RTRNG)
IPLT = 1

10 EOP = 0.
IF (IPLT.EQ.1.0.OR.IPLT.EQ.3) CALL LRGPID(-1.0,-1.0,1.0)
IF (IPLT.EQ.2.0.OR.IPLT.EQ.4) CALL LRGPID(3.3,11.0,11.0)
CALL LRCHSZ(4)
IF (IPLT.EQ.1.0.OR.IPLT.EQ.2) CALL LRLEGN(TITL1,6C,0.13,0.7,0.0)
IF (IPLT.EQ.3.0.OR.IPLT.EQ.4) CALL LRLEGN(TITL2,4C,0.3,0.7,0.0)
CALL LRCHSZ(2)
CALL LRLEGN(TITL3,12,0.45,1.5,0.0)
CALL LRLEGN(TITL4,12,1.0,4.5,0.0)
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,C.0)
CALL LRCURV(ZTIP,RTIP,NTIP,4,SYM,EOP)
IF(MBI.EQ.0) GO TO 18
DO 15 JN=I,NBLPL
15 CALL LRCURV(ZBL(1,JN),RBL(1,JN),NPPP,2,SYM,C.0)
CALL ROTATE(-ANGROT,ZLE,RLE,NBLPL,1,ZTEM,RTEM)
CALL LRCURV(ZTEM,RTEM,NBLPL,9,SYM,1.0)
18 IPLT = IPLT+1
GO TO 10

C--PLOT VERTICAL MESH LINES
20 DO 40 I=I,MM
    ZTEM(J) = ZOM(I,J)
    RTEM(J) = ROM(I,J)
40 CALL LRCURV(ZTEM,RTEM,MM,2,SYM,0.0)

C--PLOT HORIZONTAL MESH LINES
40 EOP = 0.0
DO 50 J=1,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 PTBDP

C--PTBDP OBTAINS THE HUB AND SHROUD AND BLADE LEADING AND "RAILING EDGE"
C--BOUNDARIES FOR PLOTTING, AND SCALES THE PLOT

COMMON/INPUTT/GAM, AP, MSFL, OMEGA, REDFAC, VELTOL, FNEW, DNEW, MBI, MBO,
1 MM, MHT, NL, NUB, NTIP, XIN, OUT, NBLPL, NPPP, NOSTAT, NSL, NLOSS,
2 LSF, LTL, LMVT, LROT, LBLAD, LETEAN, ANGROT, IEMESH, ISLINE,
3 ISTATL, IPLT, ISUPER, ITSON, IDEBUG, ZOMIN, ZOMBI, ZOMBO, ZOMOUT,
4 ROMIN, ROMBI, ROMBO, ROMOUT, ZHIN, ZTIN, ZOUT, RTIN, RTIN, RHOUT",
5 RTOUT, TITLE(20), ZHUB(50), RHUB(50), ZTIP(50), RTIP(50), SPIN(50),
6 RADIN(50), TIP(50), PHIP(50), LAMIN(50), VTHON(50), SPFOUT(50),
7 RADOUT(50), PRO(50), LOSSOUT(50), LAMOUT(50), VTHOUT(50),
8 DETAL(50), BETATE(50), ZHST(50), ZTST(50), RHST(50), RTST(50),
9 FLFR(50), PEBED(50), PERLOS(50), ZBL (50,50), RBL(50,50),
10 THRL(50,50), TNBL(50,50), TTBL(50,50), TH1BL(50,50), TH2BL(50,50),
11 COMMON/CALCON/MMMT, MHTPI, CP, FPON, TRGDG, PITCH, RLEH, RLET, RTEH, RTEL,
12 ZLLE(50), RLE(50), ZTE(50), RTE(50), ZLEM(101), RLEM(101),
13 SLEOM(101), ILEOM(101), ZTFOM(101), PTOM(101), STEOM(101),
14 THTEOM(101), ILE(101), ITE(101), ZOM(100,101), ROM(100,101),
15 SOM(100,101), TOM(100,101), BTH(100,101), DTHS(100,101),
16 DTHDT(100,101), PLOSS(100,101), CPHI(100,101), SPHI(100,101),
17 COMMON/ROTATN/ZHROT(50), RHROT(50), ZTROT(50), RTROT(50),
18 ZLELMB(101), BLEMB(101), ZTEOMR(101), RTEOMR(101),
19 ZBLROT(50,50), RBLROT(50,50), ZOMROT(100,101), ROMROT(100,101),
20 COMMON/PLTCOM/ZLENG, ZRRNG, RRNG, RTNG, ZHPLT(100), RHPLT(100),
21 ZSPLT(100), RSPLT(100), ZLPLT(100), RPLT(100), ZTPLT(100),
22 RTPLT(100)
DIMENSION AAA(100), BBB(100)

C--OBTAIN PLOT POINTS ON HUB

C
DELZ = (ZHROT(NHUB)-ZHROT(1))/99.
ZHLPT(1) = ZHROT(1)
DO 10 I=2,100
10 ZHLPT(I) = ZHLPT(I-1) + DELZ
CALL SPLINT(ZHROT,RHROT,NHUB,ZHLPT,100,AA,BB)
CALL ROTATE(-ANGROT, ZHLPT, RHPLT, 100, 1, 100, 1, ZHLPT, RHPLT)

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,AA,BB)
CALL ROTATE(-ANGROT, ZSPLT, RSPLT, 100, 1, 100, 1, ZSPLT, RSPLT)
IF (MBI.EQ.0) GO TO 50

C--OBTAIN PLOT POINTS UP BLADE LEADING EDGE

C
DELR = (RLET-RELEH)/99.
RLPLT(1) = RELEH
RLPLT(100) = RLET
DO 30 J=2,99
30 RLPLT(J) = RLPLT(J-1) + DELR
CALL SPLINT(RELEH,ZLE,NBLPL,RLPLT,100,ZLPLT,AA,BB)
CALL ROTATE(-ANGROT, ZLPLT, RLPLT, 100, 1, 100, 1, ZLPLT, RLPLT)
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))
RTRNG = AMAX1(RTRNG,RTIP(I))
60 RTRNG = AMAX1(RTRNG,RTIP(I))

C--CHECK INLET AND OUTLET MESH BOUNDARIES

DO 80 J=1,MHTPI
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))
RTRNG = AMAX1(RTRNG,ROM(MM,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,MHTPI))
ZRRNG = AMAX1(ZRRNG,ZOM(I,1))
ZRRNG = AMAX1(ZRRNG,ZOM(I,MHTPI))
RBRNG = AMIN1(RBRNG,ROM(I,1))
RBRNG = AMIN1(RBRNG,ROM(I,MHTPI))
RTRNG = AMAX1(RTRNG,ROM(I,1))
RTRNG = AMAX1(RTRNG,ROM(I,MHTPI))
90 RTRNG = AMAX1(RTRNG,ROM(I,MHTPI))

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))
RTRNG = AMAX1(RTRNG,RBL(I,NBLPL))
100 RTRNG = AMAX1(RTRNG,RBL(I,NBLPL))
110 DELZ = ZRRNG-ZLRNG

123
ZLRNG = ZLRNG - 0.05 * DELZ
ZRRNG = ZRRNG + 0.05 * DELZ
DELR = RTRNG - RBRNG
RBRNG = RBRNG - 0.05 * DELR
RTRNG = RTRNG + 0.05 * DELR

C-- CHOOSE MAXIMUM RANGE, AND EXPAND RANGE IN THE OTHER DIRECTION
C
DM2 = 1.1 * ABS (DELZ - DELR) / 2.
IF (DELR .GT. DELZ) GO TO 120
RTRNG = RTRNG + DM2
RBRNG = RBRNG - DM2
RETURN
120 ZRRNG = ZRRNG / DM2
ZLRNG = ZLRNG - DM2
RETURN
END

SUBROUTINE PRECAL
C
C-- PRECAL CALCULATES MANY OF THE REQUIRED FIXED CONSTANTS
C
COMMON NREAD, NWRT, ITER, IEND, NWRT1, NWRT2, NWRT3, NWRT4, NWRT5, NWRT6
COMMON /INPUT//GAM, AR, MSF1, OMEGA, REDFAC, VELTOL, FNEW, DNEW, MB1, MB0,
1 MM, MHT, NBL, NHB, NTIP, NIN, NOT出, NBLPL, NPPP, NOSTA, NSL, NLOSS,
2 LSPR, LTPL, LAMVT, LROT, LBLAD, LTEAN, ANGROT, IMESH, ISLINE,
3 ISTATL, ILOT, ISUPER, ITSON, IDEBUG, ZOMIN, ZOMBI, ZOMOUT, ZLINK,
4 ROMIN, ROMBI, ROMOUT, ZHIN, ZTN, ZOUT, ZOUT, RHIN, RHUT, RHOUT,
5 RTOUT, TITLE (20), ZHUB (50), RHUB (50), ZTIP (50), RTIP (50), SPIN (50),
6 RADIN (50), TIP (50), PRIP (50), LAMIN (50), VTHIN (50), VTHOUT (50),
7 RADOUT (50), PROP (50), LOST (50), LAMOUT (50), VTHOUT (50),
8 BETAE (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), TH2BL (50, 50),
2 COMMON/INPUT//TIPF, RHOIPF, LAMDAF, AND RVTHTA
124

COMMON/MMM1, MHT1, CP, EXFON, TGROG, PITCH, RELH, RELT, RTHE, RTET,
1 ZLE (50), BLE (50), ZTE (50), RTE (50), ZLEM (101), RLEM (101),
2 SEL (101), TLES (101), ZTEM (101), RTEM (101), STEM (101),
3 TTEM (101), YLE (101), YTE (101), YZOM (100, 101), YROM (100, 101),
4 SOM (100, 101), YOM (100, 101), BTH (100, 101), BTHD (100, 101),
5 DTHDT (100, 101), PLOSS (100, 101), CPHI (100, 101), SPHI (100, 101),
6 COMMON/VARCOM/A(4, 100, 101), KOM (100, 101), K (100, 101), RHO (100, 101),
1 WSUBS (100, 101), WSUBU (101, 100), WSUBZ (100, 101), WSUBR (100, 101),
2 WSUBB (100, 101), WTH (100, 101), WTH (100, 101), W (100, 101),
3 ALPH (100, 101), BETA (100, 101), WCRF (100, 101), CURV (100, 101),
4 WLSURF (100, 101), WTSURF (100, 101), CAMP (100, 101), SAMP (100, 101),
5 ROHAP (100, 101), DELRHO (100, 101), PT (100, 101), DSYM (100, 101),
6 XOM (100, 101), ZETOM (100, 101), DLDU (100, 101),
7 COMMON/ROTATN/ 
8 COMMON/ROTATN/ ZHROT (50), RHROT (50), ZTROT (50), RTROT (50),
1 ZLEM (101), BLEM (101), ZTEOM (101), RTEOM (101),
2 REQ (101), RPL (101), ZLEOM (101), ZLEOM (101),
3 ZBLROT (50, 50), RBLROT (50, 50), ZOMROT (100, 101), ROMROT (100, 101),
4 DYNMEN DDX (101), DDX (101), TTEM (50), DOM (101), DIP (50),
1 SZBL (50), BLOCK (50),
REAL MSF1, LAMIN, LAMOUT

C
C-- INITIALIZE TIPF, RHOIPF, LAMDAF, AND RVTHTA
CALL LAMNIT
CALL RVTNIT
IF (GAM.NE.0.) CALL TIPNIT
IF (GAM.NE.0.) CALL RHINIT

C--INITIALIZE THE BTH ARRAY
C
DO 30 J=I,MHTP1
DO 30 I=I,MM
30 BTH(I,J) = PITCH
C
C--INITIALIZE THE FLFR ARRAY IF IT WAS NOT READ IN
C
IF (NSL.GE. I) 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=I,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=I,MHTP1
SOM(I,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-1)-
1 ROM(I,J))**2)
C
C--CALCULATE TOM FROM THE ZOM,ROM ARRAYS
C
DO 100 I=I,MM
TOM(I,I) = 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-1)
1 ROM(I,J))**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--SET ILE AND ITE ARRAYS WHEN THERE ARE NO BLADES

C
IF (MBI.NE.1) GO TO 114
DO 102 J=1,MTIP
ILE(J) = MM+1
102 ITE(J) = MM+1
GO TO 225

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,NHUB,ZLE,RLE,NBLPL,ZLET,RLET)

C--CALCULATE TRAILING EDGE ARRAY, ZTE, RTE, FROM ZEL AND RPL 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,NHUB,ZTE,RTE,NBLPL,ZTEH,RTEH)

C--CALCULATE ORTHOGONAL MESH ARRAYS AT THE LEADING EDGE
C--ZLEOM (AND ZLEOMR), RLEOM (AND RLEOMR), ZLEOM, 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
130 CALL INRSCT(ZOMROT(I,J),HOMROT(I,J),MM,ZLE,RLE,NBLPL,ZLEOMR(J),
130 RLEOMR(J))
DO 140 J=1,MTIP
DO 140 I=1,MM
IF (ZLEOMR(J).LT.ZOMROT(I,J)) GO TO 15
CONTINUE
150 ILE(J) = I
160 SLEOM(J) = SUM(I-1,J)+SQRT((ZLEOM(J)-ZOMROT(I-1,J))**2
1+(RLEOM(J)-HOMROT(I-1,J))**2)
DO 170 JN=1,NBLPL
170 TTEM(JN) = THBL(1,JN)
DOM(1) = 0.
DO 174 J=2,MTIP
174 DOM(J) = DOM(J-1)+SQRT((ZLEOM(J)-ZLEOM(J-1))**2
1+(RLEOM(J)-RLEOM(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,MTIP1,THLEOM,DYDX,D2YDX2)
CALL ROTATE(-ANGROT,ZLEOM,RLEOM,MTIP1,1,1,1,ZLEOM,RLEOM)

C--CALCULATE ORTHOGONAL MESH ARRAYS AT THE TRAILING EDGE

126
C--ZTEOM (AND ZTEOMR), RTPOM (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) = ZTEOMR(1) = RTEOMR(MHTP1) = RTEOMR(1)
DO 18C J=2, MHT
18C CALL INRST (ZOMROT(1,J), RROMROT(1,J), MM, ZTE, RTE, NBLPL, ZTEOMR(J),
1PTEOMR(J))
DO 210 J=1, MHTP1
ILEJ = ILE(J)-1
DO 19C I=ILEJ, MM
IF (ZTEOMR(J) .LT. ZOMROT(ITEJ,J)) GO TO 230
19C CONTINUE
260 ITE(J) = I-1
ITEJ = I-I
21C STEOM(J) = SOM(ITEJ,J) + SQRT ((ZTEOMR(J)-ZOMROT(ITEJ,J)) ** 2
1+(RTEOMR(J)-RROMROT(ITEJ,J)) ** 2)
DO 220 JN=I, NBLPL
22C TTEM(JN) = THBL(NPPP,JN)
DOM(1) = 0.
DO 222 J=2, MHTP1
222 DOM(J) = DOM(J-I)+ SQRT ((ZTE(J)-ZTE(J-I)) ** 2
1+(RTE(J)-RTE(J-I)) ** 2)
DIP(1) = 0.
DO 223 JN=I, NBLPL
223 DIP(JN) = DIP(JN-I)+ SQRT ((Z(JN)-Z(JN-I)) ** 2
1+ (R(JN)-R(JN-I)) ** 2)
CALL SPLINT (DIP, TTEM, NBLPL, DOM, MHTP1, THTEOM, DYDX, D2YDX2)
CALL ROTATE (-ANGROT, ZTEOMR, RTEOMR, MHTPI, 1, ICI, I, ZTEOM, RTEOM)
C
C--PRINT BLADE GEOMETRY ON INPUT PLANES
C
WRITE (NWRIT, 1000)
DO 224 JN=I, NBLPL
SZBL(1) = 0.
BLOCK(1) = TTBL(1,JN)/PITCH
DO 226 IN=2, NPPP
SZBL(IN) = SZBL(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 (LETFRN.EQ.1) GO TO 232
CALL SPLINE (SZBL, THBL(IN,JN), NPPP, DYDX, D2YDX2)
GO TO 234
232 DTHDSL = TAN (BETALE(JN)/57.29578) / RBL(1,JN)
DTHDST = TAN (BETATE(JN)/57.29578) / RBL(NPPP,JN)
CALL SPLINSL (SZBL, THBL(IN,JN), NPPP, DTHDSL, DTHDST, DYDX, D2YDX2)
234 WRITE (NWRIT, 1030) JN, (IN, ZBL(IN,JN), RBL(IN,JN), THBL(IN,JN),
1TTBL(IN,JN), BLOCK(IN), SZBL(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
127
CALL THIKOM
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 (MBJ.EQ.0) GO TO 250
  DO 240 J=1,NOUT
    LAMOUT(J) = LAMOUT(J)*REDFAC
  240 VTHOUT(J) = VTHOUT(J)*REDFAC
C--RE-INITIALIZE LAMDAF AND BVTHTA FOR REDUCED FLOW
C
250 CALL BVTNIT
260 CALL LAMNIT
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),
    CPHI(I,J),SPHI(I,J),I=1,MM),J=1,MHTPI)
  WRITE(NWRIT,1040)
  RETURN
C--FOR INCOMPRESSIBLE FLOW, INITIALIZE THE DENSITY (RHO ARAY)
C
270 IF (GAM.NE.0.) RETURN
  DO 280 J=1,MHTPI
    DO 280 I=1,MM
      I_HO(I,J) = AR
  280 RETURN
C--FORMAT STATEMENTS
C
1000 FORMAT (1H1,42X,48(1H*),/42X,48H* BLADE GEOMETRY ON INPUT BLADE
  1 SECTIONS */42X,48(1H*),/)
1010 FORMAT (1H1,42X,28H** INPUT BLADE PLANE NUMBER,13,4H ***/2X,2HIN,
  16X,3HUBL,13X,3HRL,13X,4HTHBL,12X,4HTHBL,10X,3HBLOCKAGE,10X,
  25H5ZRBL,10X,6HDTHDSP,10X,7HDTHDSP2/(2X,1I2,8G16.7))
1020 FORMAT (1H1,35X,57(1H*),/35X,57H* CONSTANT QUANTITIES ON THE
  1 ORTHOGONAL MESH */35X,57(1H*),/4X,1H1,4X,1HJ,6X,3HSOM,12X,
  23HTOM,12X,3HBM,11X,5HDTHDT,10X,4HCPI,11X,4HSPI)
1030 FORMAT (2I5,6G15.5)
1040 FORMAT (1H1)
END

SUBROUTINE THETOM
C--THETOM CALCULATES THE DERIVATIVES OF THETA WITH RESPECT TO S AND T
C--DIRECTIONS ON THE ORTHOGONAL MESH
C
128
COMMON NREAD, NWRIT, ITER, IEND, NWRT1, NWRT2, NWRT3, NWRT4, NWRT5, NWRT6
COMMON/INPUT/GAM, AR, MSFL, OMEGA, REDFAC, VEITOL, PNEW, DNEW, MBI, MBO,
MM, MHT, MBL, NHUB, MTIP, NIN, NOUT, NLPL, NPPP, NOSTAT, NSL, NLOSS,
LSFR, LTPL, LAMVT, LRBL, LBLAD, LITEAN, ANGROT, IMSHE, ISLINE,
ISTATL, IPILOT, ISUPER, IITSON, IZOMIN, IZOMBI, IZOMO, IZOMOUT,
ROMIN, ROMBI, ROMBO, ROMOUT, ZHIN, ZTIN, ZHOUT, ZTOT, RIN, RTIN, RTOUT,
BTOUT, TITLE(20), ZHUB(50), RHUB(50), ZTIP(50), RTIP(50), SFIN(50),
RADIN(50), TIP(50), PRIP(50), LAMIN(50), VTIN(50), VTHIN(50), SFIN(50),
RADOUT(50), PROP(50), LOSOUT(50), LAMOUT(50), VTHOUT(50),
BETALE(50), BETATE(50), ZHST(50), ZTST(50), RHST(50), RTST(50),
FLFR(50), PERCRD(50), PERLOS(50), ZBL(50, 50), PBL(50, 50),
ZTBL(50, 50), ZTBL(50, 50), TBL(50, 50), TBL(50, 50),
COMMON/CALC/P/M1, MHTP1, CP, EXPON, TGROG, PITCH, RLEH, RRET, RTEH, RTET,
I1, Z1E(50), RLE(50), ZTP(50), RTE(50), ZLEOM(101), RLEOM(101),
SLE(101), THE10M(101), ZTEOM(101), RTEOM(101), STEOM(101),
SOM(101, 101), TOS(100, 101), TIS(100, 101), TISE(100, 101),
ZTHERM(100, 101), PLOSS(100, 101), CPH1(100, 101), SPHI(100, 101),
COMMON/INDC/P/N/1, N/P/P/C1, CP/C, D2Z/C, D2Z/C, Z/CPC(101), R/CPC(101),
NPPC = 21,

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--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,
           1RGC(1, JN), ANGZ(1, JN), D2YDX2)

DO 20 KN=1, NPPC
   20 ANGZ(KN, JN) = ATAN(ANGZ(KN, JN)) + ANGROT

C--CALCULATE ARC LENGTH ALONG INPUT LINES USING INPUT POINTS

SZRB(1) = 0.

S0 30 NZRBC(IN) = (SZRB(IN)+SQRT((ZBL(IN, JN)-ZBL(IN-1, JN))**2
                        + (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.

S0 DO 400 KN=2, NPPC

129
40 SZRPC(KN) = SZRPC(KN-1) + SQR(T((ZPC(KN,JN) - ZPC(KN-1,JN))**2
1 + (RPC(KN,JN) - RPC(KN-1,JN))**2)

C--CALCULATE THETA AND CHANGE OF THETA WITH ARC LENGTH ALONG INPUT LINES
   IF (LETEAN.EQ.1) GO TO 50
   CALL SPLINT(SZRB1,THBL(1,JN),NPPP,SZRPC,NPPC,THPC(1,JN),
   1 DTHDSP(1,JN),D2YDX2)
   GO TO 55
50 DTHDSL = TAN(BETALE(JN)/57.29578_)/RBL(1,JN)
   DTHDST = TAN(BETATE(JN)/57.295780)/RBL(NPPP,JN)
   CALL SPINT(SZRB1,THBL(1,JN),NPPP,DTHDSL,DTHDST,SZRPC,NPPC,
   1 THPC(1,JN),DTHDSP(1,JN),D2YDX2)
55 CONTINUE

C--CALCULATE BLADE THICKNESS IN THETA DIRECTION AT POINTS AT
C--FIVE PERCENT OF CHORD
   CALL SPLINT(SZRB1,TTBL(1,JN),NPPP,SZRPC,NPPC,TPPC(1,JN),DYDX,
   1 D2YDX2)
60 CONTINUE

C--CALCULATE GRADIENT OF THETA WITH RESPECT TO DISTANCE UP FIVE PERCENT
C--CHORD ZPC,RPC LINES
C
DO 130 KN=I,NPPC
   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) = TPPC(KN,JN)
      ANGT1(JN) = ANGZ(KN,JN)
   DTST1(JN) = DTHDSP(KN,JN)

70 CONTINUE

C--CALCULATE ARC LENGTH UP FIVE PERCENT CHORD LINES
   DO 80 JN=2,NBLPL
     SZRB1(JN) = SZRB1(JN-1) + SQR((ZPCT1(JN) - ZPCT1(JN-1))**2
   1 + (RPCT1(JN) - RPCT1(JN-1))**2)

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
     RPCT2(LN) = RPCT2(LN-1) + DELR
   CALL SPLINT(RPCT1,ZPCT1,NBLPL,RPCT2,RPCT1,NBLPC,ZPCT2,DYDX,D2YDX2)

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) = SZRP(LN-1) + SQR((RPCT2(LN) - RPCT2(LN-1))**2
   1 + (ZPCT2(LN) - ZPCT2(LN-1))**2)
   ANGR(KN,LN) = ATAN(DYDX(LN)) - ANGROT

C--CALCULATE THETA AND CHANGE OF THETA WITH RESPECT TO ARC LENGTH UP
C--FIVE PERCENT CHORD LINES
   CALL SPLINT(SZRB1,THPC1,NBLPL,SZRPC,NBLPC,THPC2,DYDX,D2YDX2)
   DO 110 LN=1,NBLPC
     DTHDSP(KN,LN) = DYDX(LN)
110 CONTINUE
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(SZBL,ANGT1,NBLPL,SZRPC,NBLPC,ANGT2,DYDX,D2YDX2)
CALL SPLINT(SZBL,DSTT1,NBLPL,SZRPC,NBLPC,DSTT2,DYDX,D2YDX2)
CALL SPLINT(SZBL,TTPTC1,NBLPL,SZRPC,NBLPC,TTPTC2,DTST2,D2YDX2)
C--STORE CALCULATED VALUES IN TWO-DIMENSIONAL ARRAYS ON THE ALTERNATE
C--MESH
DO 120 LN=1,NBLPL
   ZPC(KN, LN) = ZPCT2(LN)
   RPC(KN, LN) = RPCT2 (LN)
   THPC(KN, LN) = THPT2 (LN)
   TTPC (KN, LN) = TTPTC2 (LN)
   ANGZ(KN, LN) = ANGT2(LN)
   DTHDSP(KN, LN) = DTST2(LN)
120 CONTINUE
C--CALCULATE DTHDZ AND DTHDR FROM DTHDSP AND DTHDTP ON THE ALTERNATE
C--MESH
DO 140 LN=1,NBLPL
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
   DTHDR(KN, LN)= (-DTHDSP(KN, LN)*SIN(ANGR(KN, LN)) +DTHDTP(KN, LN)*COS(1ANGZ(KN, LN)))/COSAB
140 CONTINUE
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,MHTPI
   ILEJ = ILE(J)
   ITEJ = ITE(J)
   DO 150 I=ILEJ,ITEJ
      CALL LININT(ZPC, RPC, DTHDZ, NPPC,NBLPC,51,51,ZOMROT(I,J) ,
         BROMRT(I, J), DTDZOM,II,JJ)
      CALL LININT(ZPC, RPC, DTHDR, NPPC,NBLPC,51,51,ZOMROT(I, J) ,
         BROMRT(I, J), DTDROM,II,JJ)
      DTHDS(I,J) = DTDZOM*CPHI(I,J) +DTDROM*SPHI (I,J)
      DTHDT(I,J) = DTDZOM*SPHI(I,J) -DTDZOM*SPHI(I,J)
150 CONTINUE
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),
   TTPC (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
SUBROUTINE THIKOM

C--THIKOM CALCULATES THE BLADE THICKNESS IN THE THETA DIRECTION AT
C--THE POINTS OF THE ORTHOGONAL MESH

C

C--CALCULATE STREAM CHANNEL THICKNESS ARRAYS AT LEADING AND TRAILING
C--EDGES, BTLE AND BTTE

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)-BBL(1,JN-1))^2)
30
DTH(JN) = TTBL(1,JN)
RBL = RBL(1,1)*COS(ANGROT)-ZBL(1,1)*SIN(ANGROT)
DISEOM(1) = SQRT((ZLEOM(1,1)-BBL(1,1))^2+
1(RLEOM(1,1)-BBL(1,1))^2)
DISEOM(1) = SIGN(DISEOM(1),RLEOMR(1)-RBLR)
CALL SPLINT(DIST,DTH,NBLPL,DISEOM,MHTPI,DTHEOM,AAA,BBB)
DO 50 J=1,MHTPI

132
50 BTHLE(J) = PITCH-DTHEOM(J)
DTH(1) = TTBL(NPPP, 1)
DO 60 JN=2,NBLPL
DIST(JN) = DIST(JN-I)+SQRT((ZBL(NPPP,JN)-ZBL(NPPP,JN-I))**2+
1 (RBL(NPPP,JN)-RBL(NPPP,JN-I))**2)
60 DTH(JN) = TTBL(NPPP, JN)
RBLR = RBL(NPPP, I)*COS(ANGROT)-ZBL(NPPP, I)*SIN(ANGROT)
DISEOM(1) = SQRT((ZTEOM(1)-ZBL(NPPP, 1))**2+(RTEOM(1)-RBL(NPPP, 1))**2)
DISEOM(1) = SIGN(DISEOM(1), RTEOMR(I)-RBLR)
DO 70 J=2, MHTPI
70 DISEOM(J) = DISEOM(J-I)+SQRT((ZTEOM(J)-ZTEOM(J-I))**2+
1 (RTEOM(J)-RTEOM(J-I))**2)
CALL SPLINT(DIST, DTH, NBLPL, DISEOM, MHTPI, DTHEOM, AAA, BBB)
DO 80 J=1, MHTPI
80 BTHTE(J) = PITCH-DTHEOM(J)
C--INTERPOLATE TO OBTAIN BLADE THICKNESS IN THETA DIRECTION AT THE
C--POINTS OF THE ORTHOGONAL MESH, AND CORRECT BTH IN BLADE REGION
II = 1
JJ = 1
DO 90 J=1, MHTPI
ILEJ = ILE(J)
ITEJ = ITE(J)
DO 90 I=ILEJ, ITEJ
CALL LININT(ZPC, RPC, TTPC, NPPC, NBLPC, 51, 51, ZOMROT(I, J),
1 RROMROT(I, J), DBTH, II, JJ)
BTH(I, J) = BTH(I, J)-DBTH
90 CONTINUE
RETURN
END

SUBROUTINE INIT
C--INIT ASSIGN INITIAL VALUES TO THE ARRAY VARIABLES
COMMON/INPUT/GAM, AR, MSPL, OMEGA, REDFAC, VEITOL, FNEW, DNEW, MI, MBO,
1 MM, MHT, NBL, NHBB, NTIP, NIN, NOUT, NBLPL, NPPP, NOSTAT, NSL, NLOSS,
2 LSPR, LTLPL, LAMVT, LROT, LBLAD, LTEAEN, ANGROT, IMESH, ISLINE,
3 ISTATL, IPILOT, ITSUPER, ITSON, IDBUG, ZOMIN, ZOMBI, ZOMBO, ZOMOUT,
4 RMIN, RMBI, RMOBO, ROMOUT, ZHIN, ZTIN, ZTOUT, RHIN, RTIN, RTOUT,
5 RTOUT, TITLE(20), ZHUB(50), RHUB(50), TZIP(50), RTIP(50), SPIN(50),
6 RADIN(50), TIP(50), PRIP(50), LAMIN(50), VTHIN(50), SPOUT(50),
7 RADOUT(50), PROP(50), LOSOUT(50), LAMOUT(50), VTHOUT(50),
8 BETALE(50), BETATE(50), ZHST(50), ZTST(50), RST(50), RTST(50),
9 PLFR(50), PERCRD(50), PERLOS(50), ZBL(50,50), RBL(50,50),
1 THBL(50,50), THNBL(50,50), TNBL(50,50), TH2BL(50,50)
COMMON/CALCON/MMM, MHTPI, CP, EXPON, TGREEK, PITCH, RLEH, RTEH, RTET,
1 ZLE(50), RLE(50), ZTE(50), RTE(50), ZLEOM(101), RLEOM(101),
2 SLEOM(101), TLEOM(101), ZTEOM(101), RTEOM(101), STEOM(101),
3 THTEOM(101), ILE(101), TTE(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), CPHEL(100,101), SPHEL(100,101),
COMMON/VARCOM/A(4,100,101), UOM(100,101), K(100,101), RHO(100,101),
REAL K,LAMDAF
DO 10 J=I,MHTPI
WSUBS (100,101), WSUBT (100,101), WSUBZ (100,101), WSUBR (100,101),
WSUBF (100,101), WTSURF (100,101), CAMP (100,101), SAMF (100,101),
RHOAV (100,101), DELRHO (100,101), FT (100,101), DFDM (100,101),
XION (100,101), ZETOM (100,101), DLDU (100,101)
REAL K,LAMDAF
DO 10 J=I,MHTPI
A (I,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=I,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.
XION(I,J) = 0.
ZETOM(I,J) = 0.
PT(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,MHTPI) *(ROM (I,J)+ROM (I,1))/
1 (ROM (I,MHTPI)+ROM (I,1))
TPPTIP = 1.0-(2.*OMEGA*LAMDAF (UIJ,I,J) -(OMEGA*ROM (I,J))**2)/
1 (2.*CP*TPPF (UIJ))
RHO (I,J) = RHOIPF (UIJ) *TPPTIP**EXPO
10 RHOAV (I,J) = RHO (I,J)
RETURN
END

SUBROUTINE COEF
C--COEF CALCULATES COEFFICIENTS, A AND K,
C--FOR THE SYSTEM OF MATRIX EQUATIONS, A*U=K
C
COMMON NREAD, NWRT, ITER, END, NWRT1, NWRT2, NWRT3, NWRT4, NWRT5, NWRT6
COMMON /INPUT/GAM, AR, MSPL, OMEGA, REDPAC, VELTOL, FNW, DNEW, MBI, MBO,
1 MM, MHT, NBL, NHUB, NTIP, NOUT, NBLPL, NPPP, NOSTAT, NSL, ELSS,
2 LSFR, LTP, LAMT, LROT, LBLAD, LTEAN, ANGROT, IMESH, ISLINE,
3 ISTATE, IPILOT, ISUPER, ITSON, IDEBUG, ZMIN, ZOMO, ZOMBO, ZOMOUT,
4 ROMIN, ROMBI, ROMBO, ROMOUT, ZTHIN, ZTHO, ZTOUT, RHIN, RTHIN, RTHOUT,
5 RTOUT, TITLE (20), ZHUB (50), RHUB (50), ZTIP (50), ZTIP (50), SPIN (50),
6 RADIN (50), TIP (50), FIP (50), LAMIN (50), VTHIN (50), SPOUT (50),
7 RADOUT (50), PFOV (50), LOSOUT (50), LAMOUT (50), VTHOUT (50),
DIMENSION DVTHDT (100,101)
REAL MSFL, K, KNEW

C--CALCULATE COEFFICIENTS AND CONSTANTS FOR FINITE DIFFERENCE EQUATIONS
C
WRITE(NWRTI, 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, MMMI
IF (ITER.EQ.1) DVTHDT(I,J) = 0.
H1 = TOM(I,J) - TCM(I,J-I)
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)
GO TO 20
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) / RP(I,J) + DELPHS/C2
D2 = (BTH(I,J+1) - BTH(I,J-1))/BTH(I,J) + (RHO(I,J+1) - RHO(I,J-1))/1RHO(I,J)
D2 = D2/C2 + SPHI(I,J) / RP(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.FQ.DCH) JCH = J
DMAX = AMAX1(DMAX,DTEMP)
DMIN = AMINI(DMIN,DTEMP)
DVTHDT(I,J) = DNEW*DTEMP+(1.-DNEW)*DVTHDT(I,J)
KNEW = KNEW+3TH(I,J)/ROM(I,J)*DVTHDT(I,J)+FT(I,J)
IF (GAM.EQ.0.) NNEW=KNEW+OMEGA*(LAMDAF(UOM(I,J+I),I,J+1)-
LMADAF(UOM(I,J-1),I,J-1))/CI
K(I,J) = KNE
CONTINUE
IF (ITER.GT.1) WRITE(NWRITE,1010) DCHANG,ICH,JCH,DVTHDT(ICH,JCH),
DMAX,DMIN
C--PRINT DEBUG OUTPUT
C
IF (IDEBUG.LE.0) RETURN
IF (((ITER/IDEBUG)IDEBUG.NE.ITER.NF.1) RETURN
WRITE(NWRITE,1020)
DO 60 J=2,MHT
DO 60 I=I,MM
WRITE(NWRITE,1030) I,J,A(IJ,I,J),K(I,J)
WRITE(NWRITE,1040)
RETURN
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,26HMINIMUM VALUE OF
2DVTHDT =,G13.5/5X,26HMINIMUM VALUE OF DVTHDT =,G13.5)
1020 FORMAT (1H1///3CX,67(1H*)///3CX,67H* COEFFICIENTS OF MATRIX EQU
1ATION FOR STREAM FUNCTION */3CX,67(1H*)///5X,1HI,5X,1HJ,
26X,4HA(1),12X,4HA(2),12X,4HA(3),12X,4HA(4),13X,1HK///)
1030 FORMAT (216,5G16.6)
10_0 FORMAT (IHI)
END
SUBROUTINE SOR
C--SOR SOLVES THE SET OF MATRIX EQUATIONS, A*U=K
C--BY THE SUCCESSIVE OVERRELAXATION TECHNIQUE
C
COMMON NREAD,NWRITE,ITER,IEND,NNRT1,NNRT2,NNRT3,NNRT4,NNRT5,NNRT6
COMMON/INPUTT/GAM,AR,MSFL,OMEGA,REDFAC,VELTOL,FNEW,DNEW,MBI,MBO,
1 MM,MHT,NBL,HUB,NPL,NOUT,NPLPL,NPPP,NOSTAT,NSL,NLOSS,
2 LSPR,LPL,LAMUT,LROT,LBLAD,LETEAN,ANGROT,IMESH,ISLINE,
3 ISTATL,IPLT,ITSON,IDEBUG,ZMIN,ZMIN,ZMOB,ZMOM,ZMOUT,
4 ROMIN,ROMBI,ROMBO,ROMOUT,ZHIN,ZTIN,ZOUT,ZHIN,ZTIN,ZHOUT,
5 RTOUT,TITLE(20),ZHUB(50),RHUB(50),NZTIP(50),ZTIP(50),RTIP(50),
6 RADIN(50),TIP(50),PRIP(50),LAMIN(50),VTIN(50),SPOT(50),
7 RADOUT(50),PROP(50),LOSSOUT(50),LAMOUT(50),VTHOUT(50),
8 BETALE(50),BETATE(50),ZHT(50),ZTST(50),RHST(50),RTST(50),
9 FLFR(50),PERCRD(50),PERLOS(50),ZBL(50,50),RBL(50,50),
136

136
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) = O.
10 UOM(I,HMTPI) = O.
DO 20 J=2,MHT
   DO 10 I=1,MM
   UOM(I,J) = 1.
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(I,I,J)*UOM(I,J-1) + A(I,I,J)*UOM(I,J+1)
      IF (I.NE.1) UNEW = UNEW + A(I-1,I,J)*UOM(I-1,J)
      IF (I.NE.MM) UNEW = UNEW + A(I+1,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.1.0) LMAX = 1.
   ORFMAX = 2./SQRT(1.-LMAX)
   ORFMIN = 2./SQRT(1.-LMIN)
   IF (((ORFMAX-ORFMIN)*GT.(2.-ORFMAX)*OR.(ORF-ORFMAX)*GT.0.0005)
      GO TO 30
   ORF = ORFMAX
   WRITE (NWRITE,1000) ORF
C
C--RESTORE U BOUNDARY VALUE AT SHROUD
DO 50 I=1,MM
   UOM(I,HMTPI) = 1.
50 UOM(I,HMTPI) = 1.
C
C--SOLVE MATRIX EQUATION BY SOR
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)*UOM(I+1, J)
CHANGE = ORF*CHANGE
ERROR = AMAX1 (ERROR, ABS (CHANGE))
UOM(I, J) = UOM(I, J) + CHANGE
IF (ERROR.GT.1.E-5) GO TO 70
RETURN
FORMAT (/5X, 40HCALCULATED OVERRELAXATION FACTOR (ORF) =, F7.3) END

SUBROUTINE LOSSOM

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, NWRT1, NWRT2, NWRT3, NWRT4, NWRT5, NWRT6
COMMON/INPUT/GAM, AR, MSF1, OMEGA, REDFAC, VELTOL, FNWEK, DNFW, MB1, MBO,
1 MM, MHTPI, CP, EXPON, TGROG, PITCH, RLEH, RTEH, RTET,
2 SLEOM(101), THEOM(101), ZROM(100, 101), RROM(100, 101), STEOM(101),
3 THTEOM(101), ILE(101), IT(101), ZOM(100, 101), ROM(100, 101),
4 SOM(100, 101), TOM(100, 101), RTH(100, 101), DTHDS(100, 101),
5 DTHDT(100, 101), PLOSS(100, 101), GPHI(100, 101), SPHI(100, 101)
COMMON/VARCOM/A(4, 100, 101), UOM(100, 101), K(100, 101), RH0(100, 101),
1 WSUBS(100, 101), WSUBT(100, 101), WSUBZ(100, 101), WSUBZ(100, 101),
2 WSDBM(100, 101), WTH(100, 101), VTH(100, 101), W1(100, 101),
3 ALPHA(100, 101), BETA(100, 101), WCR(100, 101), CBV(100, 101),
4 WLSURF(100, 101), WTSURF(100, 101), WCMP(100, 101), WAM(100, 101),
5 RHOAV(100, 101), DELRHO(100, 101), PT(100, 101), DFD(100, 101),
6 XION(100, 101), ZETON(100, 101), LDOM(100, 101),
COMMON/ROTATN/ZHROT(50), ZHROT(50), ZTROT(50), RTROT(50),
1 ZLEOM(101), ZLEOMR(101), ZTEOM(101), ZTEOMR(101),
2 ZBLROT(50, 50), ZBLROT(50, 50), ZOMROT(100, 101), ZROM(100, 101),
DIMENSION ZTENR(2), ZTEMR(2), ZOMIN(101), ZOMOUT(101)

REAL LAMDAF, LOSSOUT
1 I = 1
2 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
II = 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
IL0S = 0
DO 20 JN=1,NOUT
TNP = TIPF(SFOUT(JN))
TOP = TNP-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**2(GAM*EXPON)
20 IF (LOSOUT(JN).LT.-.001) ILO5=I
IF (ILOS.EQ.I) 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=I,MHTP1)
C
C--DISTRIBUTE TOTAL PRESSURE LOSS AT POINTS OF ORTHOGONAL MESH
C
35 CALL SPLINT(SFOUT,LOSOUT,NOUT,O.,1,TRMP,TEMPI,TEMP2)
DO 40 J=1,MHTP1
DO 40 I=I1,I2
40 CALL SPLINT(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) = AMINI(PC,AMAXI(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(I) = ZHOUT*CAN+RHOUT*SAN
RTEM(1) = RHOUT*CAN-ZHOUT*SAN
ZTEM(2) = ZTOUT*CAN+RTOUT*SAN
RTEM(2) = RTOUT*CAN-ZTOUT*SAN
DO 80 J=1, MHTP1
CALL INRSCT(ZOMROT(1,J), ROMROT(1,J), MM,ZTEM,RTEM,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 +
   (RIR-ROMROT(I-1,J))**2)
85 IF(1BI.EQ.C) GO TO 100
INRSCT(ZOMROT(1,J), ROMROT(1,J), MM,ZTEM,RTEM,2,ZIR,RIR)
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=1, 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 SPLINT(PC,1,PERLS,TEMP1,TEMP2)
IF (I.GT.ITE(J)) PERLS = PERLOS(NLOSS) + (SOM(I,J) - STEOM(J)) * DELPCO
IF (PERLS.GE.1.0) PERLS = 1.0
RETURN
90 PLOSS(I,J) = PERLS*PLOSS(I,J)
RETURN
C-- WITH BLADES, AND LINEAR DISTRIBUTION OF LOSS FROM TRAILING EDGE TO DOWNSTREAM INPUT STATIONS
C-- UPSTREAM TO DOWNSTREAM INPUT STATIONS
100 IF (ITER.GT.1) GO TO 135
ZTEM(1) = ZHIN*CAN+PHIN*SAN
RTEM(1) = RHIN*CAN-ZHIN*SAN
ZTEM(2) = ZTIN*CAN+RTIN*SAN
RTEM(2) = RTIN*CAN-ZTIN*SAN
DO 130 J=1, MHTP1
CALL INRSCT(ZOMROT(1,J), ROMROT(1,J), MM,ZTEM,RTEM,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 +
   (RIR-ROMROT(I-1,J))**2)
C-- NO BLADES, AND LINEAR OR INPUT DISTRIBUTION OF LOSS FROM
135 IF (NLOSS.GT.0) CALL SPLINT(PERCRD,PERLOS,NLOSS,0.,1,PERLS,
   TEMP1,TEMP2)
DO 140 J=1, MHTP1
SLENTH = SOMOUT(J) - SOMIN(J)
DO 140 I=1, I2
PC = (SOM(I,J) - SOMIN(J))/SLENTH
PERLS = PC
IF (NLOSS.GT.0) CALL SPLINT(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,12,3X,F10.6))
1010 FORMAT (/5X, 29HCALCULATED ILE AND ITE ARRAYS/10X,1HJ,5X,3HILE,3X,
SUBROUTINE NEWRHO
C
C--NEWRHO CALCULATES VELOCITY COMPONENTS, VELOCITY MAGNITUDE,
C--AND NEW DENSITY AT EACH MESH POINT
C
COMMON WHEAD, NWRITE, ITER, IEND, NWRITE1, NWRITE2, NWRITE3, NWRITE4, NWRITE5, NWRITE6
COMMON/INPUT/GAM, AR, MSPL, OMEGA, REDFAC, VELTOl, FNEW, DNEW, MRI, MBO,
1 MM, MHT, RHU, NTIP, MIN, NOUT, NBLPL, NPPP, NOSTA, NSL, NLOSS,
2 LSFR, LTPL, LAMVT, LSOt, LBLAD, LETHN, ANGROT, IMESH, ISLINE,
3 ISTAT1, IPILOT, ISUPER, ITSON, IDEBUG, ZMIN, ZOMBI, ZOMBO, ZOMOUT,
4 DOMIN, DOMBI, BOMBO, BOMOUT, ZHIN, ZTIN, ZHOUT, ZTOUT, RHIN, RTIN, ROUTH,
5 RTOUT, TITLEI (20), ZHUB (50), RHUB (50), ZTIP (50), RTIP (50), SPF (50),
6 RADIN (50), TIP (50), PRIP (50), LAMIN (50), VTHIN (50), SPTIP (50),
7 RADOUT (50), PROP (50), LOSSOUT (50), LAMOUT (50), VTHOUT (50),
8 BETALE (50), BETATE (50), ZHST (50), ZTST (50), RHOST (50), RST (50),
9 PLFR (50), PECCRD (50), PEBLOS (50), ZBL (50, 50), RBL (50, 50),
1 THBL (50, 50), TNLB (50, 50), TTB (50, 50), TH1BL (50, 50), TH2BL (50, 50)
COMMON/CALCGN/HHH, HHTP, CP, EXPON, TGR0G, PITCH, BLEN, BLET, RTEH, RTET,
1 ZLE (50), RLE (50), ZTI (50), RTI (50), ZLEM (101), RLEM (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 SGOM (100, 101), TEGOM (100, 101), BTH (100, 101), DTHDS (100, 101),
5 DTHDT (100, 101), PLOSS (100, 101), CPHI (100, 101), SPHI (100, 101)
COMMON/VARC/AR (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), WTH (100, 101), W (100, 101),
3 ALPHA (100, 101), BETA (100, 101), WCR (100, 101), CURV (100, 101),
4 WLSURP (100, 101), WSURF (100, 101), WSURF (100, 101), CAMP (100, 101),
5 RHOAV (100, 101), DELRHO (100, 101), FT (100, 101), DFDN (100, 101),
6 XIOM (100, 101), ZETOM (100, 101), DLDU (100, 101), DLDU (100, 101)
DIMENSION DUDS (100), DVT (101), DUVT (101), DDT (101), TPP (101),
1 PREL (101), DPD (101), DDT (101), AAA (101)
REAL MSPL, LAMDA, LAMBDA, LAMBDO
REAL 0.
REAL 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) * MSPL / ROM (I, J) * BTH (I, J)) -
1 DFDN (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
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) = (DDBT(J) * MSFL / (ROM(I,J) * BTH(I,J)) -
1DFDM(I,J) * DGLRHQ(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)
LAMDBA = LAMDAF(UOM(I,J), I, J)
LAMBD0 = RVTHTA(UOM(I,J), I, J)
IF (I.LT.ILE(J)) WTH(I,J) = LAMDBA / ROM(I,J) -OMR
IF (I.GT.ITE(J)) WTH(I,J) = LAMBD0 / ROM(I,J) -OMR
VTH(I,J) = WTH(I,J) + OMR
WSQ = WTH(I,J)**2 + WSUBS(I,J)**2 + WSUBT(I,J)**2
WTMP = SQRT(WSQ)
ERR = 0.
IF (W(I,J).NE.0.) ERR = ABS(WTMP - 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 + I
RELERA = RELERA + ERR
W(I,J) = WTMP
IF (GAM.EQ.0.) GO TO 30
TTPT = TTPP(UOM(I,J))
RHOIP = RHOIPF(UOM(I,J)) * (1. - PLOSS(I,J))
TTPP(J) = TTPT - (2. * OMEGA * LAMDBA - OMR)**2 / CP/2.
IF (TTPP(J).LT.0.) GO TO 60
PREL(J) = RHOIP * AR**TIPT**2 / (TTPP(J) / TTPT)**(GAM*EXPON)
TTIP = (TTPP(J) - WSQ / CP / 2.) / TTPT
IF (TTIP .LT. 0.) GO TO 50
RHO(I,J) = RHOIP**TIPT**EXPPON
30 CONTINUE
IF (GAM.EQ.0.) GO TO 40
CALL SLOPES(TVERT, TTPP, MHTP1, DTDT)
CALL SLOPES(TVERT, PREL, MHTP1, DPDT)
DO 35 J=1, MHTP1
XIOM = (AR / PREL(J) * DPDT(J) / CP - DTDT(J) / TTPP(J)) / 2.
ZETOM = OMEGA**2 ** ROM(I,J) * CPHI(I,J) -AR / PREL(J) * TTPP(J) * DPDT(J)
XIOM(I,J) = XNEW**XIOM + (1. - XNEW) * XIOM(I,J)
35 ZETOM(I,J) = ZNEW**ZETOM + (1. - ZNEW) * ZETOM(I,J)
40 CONTINUE
RELER = RELERA / FLOAT(MM * MHTP1)
IF (ITER.GT.1) WRITE(NWRIT,1020) RELER, IREL, JREL, RELERA, ICOUNT
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 (IPLAT.GT.1) IPLAT = 1
142
IF (ITSON.GT.1) ITSON=1
IF (IDEBUG.GT.1) IDEBUG=1
RETURN
5C WRITE(NWRIT,1000)
STOP
60 WRITE(NWRIT,1010)
STOP
1000 FORMAT(///68H PROGRAM STOPPED IN NEWRHO DUE TO EXCESSIVE STREAM FUNCTION GRADIENT)
1010 FORMAT(///61H THE UPSTREAM INPUT WHIRL OR TANGENTIAL VELOCITY IS 1000 LARGE)
1020 FORMAT(/ 5X,57HMAXIMUM RELATIVE CHANGE IN VELOCITY =,G11.4,
18H AT I =,I3,5H, J =,I3/5X,57HAVGVERAGE RELATIVE CHANGE IN VELOCITY
2 =,G11.4/5X,57NUMBER 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, NWRIT1, NWRIT2, NWRIT3, NWRIT4, NWRIT5, NWRIT6
COMMON INPUT/GAM, NR, MSFIV, OMEGA, REFPAC, VELTOI, FNEW, DNEW, MBI, MBO,
1 MM, MHT, NBL, NUB, NTIP, KTIN, NOUT, NLBPL, NPP, NOSTAT, NSL, NLOSS,
2 LSFR, LPL, LAMIT, LR, LBLAD, LETEAN, ANGROT, IIMSH, IISLINE,
3 ISTATL, ILPL, ISUPFR, ITSON, IDEBUG, ZOMIN, ZOMB, ZOMBO, ZOMOUT,
4 ROMIN, ROMBI, ROMBO, ROMOUT, ZHRH, ZTIN, ZHOUR, ZTOUT, RHIN, RTIN, RTOUT,
5 RTOUT, TITLE(22), ZHUB(50), RHUB(50), ZTIP(50), RTIP(50), SPIN(50),
6 RAIN(50), TIP(50), PRIP(50), LAMIN(50), WHIN(50), SPOUT(50),
7 RADOUT(50), PROP(50), LCSOUT(50), LAMOUT(50), VTHOUT(50),
8 BETAF(50), BETAT(50), ZHST(50), ZTST(50), RHTST(50), RTST(50),
9 PLFR(50), PERCRD(50), PERLOS(50), ZBL(50,50), RBL(50,50),
10 TBBL(50,50), TNBL(50,50), TTBL(50,50), TH1BL(50,50), TH2BL(50,50)
COMMON/CALCCN/MMM, MHTP, DP, EXPO, TGROT, PITCH, BLEH, FLET, RTBH, RTET,
1 ZTE(50), RPE(50), ZTE(50), RTF(50), ZTEOM(101), RLEOM(101),
2 SLEOM(101), TLEOM(101), ZTEOM(101), RLEOM(101), STEOM(101),
3 TLEOM(101), LIE(101), ZLEOM(101), RLEOM(101), SLEOM(101),
4 ZGMB(100,101), TGBM(100,101), SGBM(100,101), DTMHD(100,101),
5 DTHD3(100,101), PLOSS(100,101), ZHBTM(100,101), SPMH(100,101),
6 COMG/MVARCOM/A(4,100,101), K(100,101), RHO(100,101),
1 WSUBS(100,101), WSUBB(100,101), WUSPR(100,101),
2 WSUBM(100,101), WTM(100,101), VMTH(100,101),
3 ALPHA(100,101), BETA(100,101), WCER(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), PT(100,101), DPM(100,101),
6 XION(100,101), ZETOM(100,101), DLDU(100,101),
7 COMMON/SLCOM/ILS(50), ITS(50), ZSL(100,50), RSL(100,50), MSL(100,50),
1 WZSL(100,50), WRS(100,50), MWSL(100,50), WTHS(100,50),
2 ALPSL(100,50), BETSL(100,50), WSL(100,50), WWSL(100,50),
3 CURVSL(100,50), WLSS(100,50), WTS(100,50),
4 COMMON/ROTATN/ZHROT(50), RHROT(50), ZTRAT(50), RTRAT(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),
C--CALCULATE VELOCITY COMPONENTS AND FLOW ANGLES ON ORTHOGONAL MESH

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) - SPHI(I,J)

WSUBR(I,J) = WSUBT(I,J) + CPHI(I,J) + SPHI(I,J)

ALPHA(I,J) = ATAN2(WSUBR(I,J),WSUBZ(I,J))

BETA(I,J) = ATAN2(WTH(I,J),WSUBM(I,J))

GO TO 30

C
ENTRY TOUTPT

C--CALCULATE VELOCITY COMPONENTS ON MESH, AFTER TRANSONIC SOLUTION

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) = WSHBM(I,J)*COS(ALPHA(I,J))

WSUBR(I,J) = WSUBM(I,J)*SIN(ALPHA(T,J))

VTH(I,J) = WTH(I,J) + OMEGAW'ROR(I,J)

C
C--COMPUTE BLADE SURFACE VELOCITIES

30 CALL BLDVEL

C--STORE 'CHOKED' MESSAGE FOR APPROPRIATE VERTICAL ORTHOGONAL MESH LINES

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)

CONTINUE

C
C--CALCULATE STREAMLINE CURVATURE AND CRITICAL VELOCITY RATIO ON MESH

DO 50 I=1,MM

TVERT(J) = TOM(J,I)

ALVERT(J) = ALPHA(I,J)

CALL SLOPES(TVERT,ALVERT,MHTP1,DALVER)

DO 50 J=1,MHTP1

DALDT(I,J) = DALVER(J)

50 CONTINUE
DO 60 J=I,MHTPI
CALL SLOPES(SOM(1,J),ALPHA(1,J),MM,DALDS).
DO 60 I=I,MM
CURV(I,J) = DALDS(I)*CAMP(I,J)+DALDT(I,J)*SAMP(I,J)
IF (GAM.EQ.C.) GO TO 60
TPP = TIPF(UOM(I,J))-(2.*OMEGA*LAMDAF(UOM(I,J),I,J)-(OMEGA*
1ROM(I,J)*D2)/2./CP
IF (TPP.LE.C.) TPP=1,
WWCR(I,J) = W(I,J)/SQRT(TGROG*TPP)
60 CONTINUE

C--CHECK PRINT AND PLOT INDICATORS TO SEE IF OUTPUT CALCULATIONS
C--SHOULD BE MADE

IF (IMESH.LE.0) GO TO 32
IF ((ITER/IMESH)*IMESH.EQ.ITER.OR.ITER.EQ.1) GO TO 38
32 IF (ISLINE.LE.O) 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.1) GO TO 35
IF ((ITER/IPLOT)*IPlot.EQ.ITER.OR.ITER.EQ.1) GO TO 38
35 IF (ITSON.LE.0) GO TO 36
IF ((ITER/ITSON)*ITSON.NE.ITER) RETURN
38 IF (MBI.EQ.O) GO TO 50

C--CHECK IF UPPER OR LOWER SURFACE IS SUCTION SURFACE
REVERS = 0.0
IF ((LAMDAF(.5,IIF(1) ,I)-RVTHTA(.5,IIF(1),I)) .GT.0) GO TO 50
REVERS = 1.0
DO 70 J=I,MHTPI
DO 70 I=I,MM
WDUM = WLSURF(I,J)
WLSURF(I,J) = WTSURF(I,J)
WTSURF(I,J) = WDUM
70 CONTINUE

C--PRINT OUTPUT ROW BY ROW FROM HUB TO TIP ON ORTHOGONAL MESH
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,MHTPI
WRITE(NWRT1,1010) J
WRITE(NWRT1,1020)
DO 90 I=I,MM
PHI = ATAN2(SPHI(I,J),CPSI(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),WSURM(I,J),
1WTH(I,J),W(I,J),WWCR(I,J),ALPHIJ,BETAIJ,PHI,CHOMES(1,I),
2CHOMES(2,I)
C--CALCULATION OF OUTPUT DATA ON STREAMLINES
C

10 IF (ISLINE.LE.0) GO TO 110
   IF ((ITER/ISLINE)*ISLINE.EQ.ITER.OR.ITER.EQ.1) GO TO 130
110 IF (YPLOT.LE.0) GO TO 120
   IF ((ITER/YPLOT)*YPLOT.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=I,M
   DO 140 J=I,MHTPI
      ZTEM(J) = ZOMROT(I,J)
      RTEM(J) = ROMROT(I,J)
   140 UTEM(J) = UOM(I,J)
      CALL SPLINT (UTEM,RTEM,MHTPI,FPLFR,NSL,RSLTEM,AAA,BBB)
      CALL SPLINT (UTEM,ZTEM,MHTPI,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(I,JS) = 0.
   MSL(IS,J5) = MSL(IS-I,JS)*SQRT((ZSL(IS,JS)-ZSL(IS-I,JZ))**2
   1+ (RSL(IS,JS)-RSL(IS-1,JS))**2)
   MSL(IS,JS) = MSL(IS,JS)+SQRT((ZSL(IS,JS)-ZSL(IS-I,JS))**2
   1+ (RSL(IS,JS)-RSL(IS-1,JS))**2)
C
C--INTERPOLATE TO OBTAIN OUTPUT DATA ON STREAMLINES
   II = 1
   JJ = 1
   DO 180 JS=1,NSL
      DO 180 IS=1,M
         CALL LININT(ZOMROT,ROMROT,WSUBZ,MM,MHTPI,100,101,ZSL(IS,JS),
         1RSL(IS,JS),ZSLTEM(IS,JS),II,JJ)
         CALL LININT(ZOMROT,ROMROT,WSUBR,MM,MHTPI,100,101,ZSL(IS,JS),
         1RSL(IS,JS),WRSI(IS,JS),II,JJ)
         CALL LININT(ZOMROT,ROMROT,WH,MM,MHTPI,100,101,ZSL(IS,JS),
         1RSL(IS,JS),WTHSL(IS,JS),II,JJ)
         CALL LININT(ZOMROT,ROMROT,WWC,MM,MHTPI,100,101,ZSL(IS,JS),
         1RSL(IS,JS),WWCRSL(IS,JS),II,JJ)
         CALL LININT(ZOMROT,ROMROT,CURV,MM,MHTPI,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))
         BESL(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,M
      WLSSL(IS,JS) = 0.
      WTSSL(IS,JS) = 0.
   190 WSSL(IS,JS) = WSSL(IS,JS)+SQRT((ZSL(IS,JS)-ZSL(IS-I,JS))**2
   1+ (RSL(IS,JS)-RSL(IS-1,JS))**2)
   IF (MBI.EQ.0) GO TO 265
II = 1
JJ = 1
DO 200 JS=1,NSL
ILSJ = ILS(JS)
ITSJ = ITS(JS)
DO 200 IS=ILSJ,ITSJ
CALL LININT(ZOMROT,ROMROT,WLSRF,MM,MHTP1,100,101,ZSL(IS,JS),
1RSL(IS,JS),WLSSL(IS,JS),II,JJ)
200 CALL LININT(ZOMROT,ROMROT,WTSURF,MM,MHTPI,100,101,ZSL(IS,JS),
1RSL(IS,JS),WTSSL(IS,JS),II,JJ)
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,IC0,50,ZSL,RSL)
207 BETEM(IS) = BETSL(IS,JS)*DEGRAD
WRITE (NWBT2,1050) JS,FLFR(JS)
WRITE (NWRT2,1060)
210 WRITE(NWRT2,1070) (ZSL(IS,JS),RSL(IS,JS),MSL(IS,JS),WMSL(IS,JS),
IWTHSL(IS,JS),WSL(IS,JS),WWCRSL(IS,JS),ALTFM(IS),BETEM(IS),
2CURVSL(IS,JS),WLSSL(IS,JS),HTSSL(IS,JS),CHOMES(I,IS),CHOMES(J,IS) ,
3IS=I,MM)
220 IF (ISTATL.LE.0.OR.NOSTAT.EQ.0) GO TO _10
IF (((ITER/ISTATL)*ISTATL.NE.ITER.AND.ITER.NE.1) GO TO 410
WRITE (NWRT3, I080)
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
CALCULATE ZST AND RST ARRAYS
CALL SPLINT(ZHROT,RHROT,NHUB,ZHST,NOSTAT,ZST,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(MSL) = ZTST(IL)
ZTEM(1) = ZHST(IL)
ZTEM(20) = ZTST(IL)
DELZ = (ZTEM(20)-ZTEM(1))/19.0
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)-
     ZLEOMR(MHTP1)).LT.DELCH) MARK=2
   IF (ABS(ZST(1)-ZLEOMR(1)).LT.DELCH.AND. ABS(ZST(NSL)-
     ZTEOMR(MHTP1)).LT.DELCH) MARK=3
   IF (ZST(1).GT.(ZLEOMR(1)+DELCH).AND.ZST(1).LT.(ZTEOMR(1)-
     DELCH)) 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--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,MMTPI,100,101,ZTEM(J),RTEM(J),
     UTEM(J),II,JJ)
   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--CALCULATE STATION LINE ZST COORDINATES FOR PRINT OUT
   CALL SPLINT(RLE,ZLE,NBLPL,RST,NSL,ZST,AAA,BBB)
   CALL SPLINT(RTE,ZTE,NBLPL,RST,NSL,ZST,AAA,BBB)
   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,MMTPI,100,50,ZST(JL),RST(JL),MST(JL),
     II,JJ)
   C--INTERPOLATE TO OBTAIN OUTPUT DATA ON STATION LINES
   C
   370 II = 1
   148
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) = BLOEMR(J)
GO TO 378
376 DO 377 J = 1, MHTP1
I = ITE(J) + 1
EXFRAC = (SOM(I, J) - STEOM(J)) / (SOM(I, 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) = RTEMR(J)
378 JLTE = 1
DO 384 JLTE = MHT
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)
384 BETST(JL) = ATAN2(WHST(JL), WMST(JL)) * DEGRAD
388 BETST(JL) = ATAN2(WZST(JL)**2 + WRST(JL)**2)
389 WTSST(JL) = 0.
WTSST(JL) = 0.
IF (MARK.EQ.1) GO TO 392
C--NORMAL CASE OF FREESTREAM STATION, OR STATION WITHIN BLADE
386 DO 388 JLTE = MHT
DO 380 J = JLTE, MHT
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), WHST(JL), II, JJ)
WMST(JL) = SQRT(WZST(JL)**2 + WRST(JL)**2)
388 BETST(JL) = ATAN2(WZST(JL), WMST(JL)) * DEGRAD
C--CALCULATE OTHER OUTPUT DATA ON STATION LINES
C 390 DO 392 JLTE = MHT
DO 388 JLTE = MHT
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
WSST(JL) = SQRT(WST(JL)**2 + WTHST(JL)**2)
WLSTST(JL) = 0.
WTSTST(JL) = 0.
IF (MARK.EQ.1) GO TO 392
C--NORMAL CASE OF FREESTREAM STATION, OR STATION WITHIN BLADE
C 390 DO 392 JLTE = MHT
DO 388 JLTE = MHT
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
WSST(JL) = SQRT(WST(JL)**2 + WTHST(JL)**2)
WLSTST(JL) = 0.
WTSTST(JL) = 0.
IF (MARK.EQ.1) GO TO 392
C--CALCULATE OTHER OUTPUT DATA ON STATION LINES
C 390 DO 392 JLTE = MHT
DO 388 JLTE = MHT
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
WSST(JL) = SQRT(WST(JL)**2 + WTHST(JL)**2)
WLSTST(JL) = 0.
WTSTST(JL) = 0.
IF (MARK.EQ.1) GO TO 392
C--NORMAL CASE OF FREESTREAM STATION, OR STATION WITHIN BLADE
C 390 DO 392 JLTE = MHT
DO 388 JLTE = MHT
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
WSST(JL) = SQRT(WST(JL)**2 + WTHST(JL)**2)
WLSTST(JL) = 0.
WTSTST(JL) = 0.
IF (MARK.EQ.1) GO TO 392
C--CALCULATE OTHER OUTPUT DATA ON STATION LINES
C 390 DO 392 JLTE = MHT
DO 388 JLTE = MHT
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
WSST(JL) = SQRT(WST(JL)**2 + WTHST(JL)**2)
WLSTST(JL) = 0.
WTSTST(JL) = 0.
CONTINUE
        CALL ROTATE(-ANGROT,ZST,RST,NSL,1,50,1,ZST,RST)
C--CALCULATE EXTRA OUTPUT DATA ON STATION LINES
DO 396 JL=I,NSL
  LAMBDA = LMDAF(FLFR(JL),ILE(1),1)
  OMR = OMEGA*RST(JL)
  VTHST(JL) = WTHST(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
  WMCRST(JL) = WST(JL)/SQRT(TGROG*TPPST(JL))
  RHOPP = RHOIP*(TPPST(JL)/TIPT)**EXPON
  PPSST(JL) = RHOIP*TPPST(JL)
  TST(JL) = TPPST(JL)/TIPT+OMR_VTHST(JL)-OMEGA*LAMBDA)/CP
  RHOST(JL) = RHOST(TST(JL)/TIPT)**EXPON
  PPSST(JL) = RHOST_AR*TPST(JL)
  TST(JL) = TPST(JL)-VSQ/2./CP
  RHOST(JL) = RHOST(TST(JL)/TIPT)**EXPON
  PPSST(JL) = RHOST_AR*TPST(JL)
396 CONTINUE
C--PRINT OUTPUT ALONG HUB-SHROUD STATION LINES
        IF (NCHOK.GT.0) WRITE(NWRT3,1095) 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) (RST(JL),ZST(JL),MST(JL),FLFR(JL),WMST(JL),
                     WTHST(JL),WST(JL),WMCRST(JL),ALPST(JL),BETST(JL),
                     2*WLSST(JL),WTSST(JL),JL=I,NSL)
        WRITE(NWRT3,1142) (RST(JL),ZST(JL),MST(JL),ST(JL),PST(JL),TST(JL),
                         RHOST(JL),
                         1*WLSST(JL),WTSST(JL),JL=I,NSL)
              00 CONTINUE
C--REVERSE UPPER AND LOWER SURFACE VELOCITIES, IF NECESSARY
410 IF (REVERS.EQ.0.) GO TO 430
        DO 420 J=1,MHTPI
          DO 420 I=I,MM
            WDUM = WLSURF(I,J)
            WLSURF(I,J) = WTSURF(I,J)
            WTSURF(I,J) = WDUM
          REVERS = 0.0
C--REMOVE *CHOKED* MESSAGE, IF NECESSARY
430 IF (NCHOK.EQ.0.) RETURN
        DO 440 I=1,MM
        150
CHOMES (1, I) = BLNK
440 CHOMES (2, I) = BLNK
RETURN

C--FORMAT STATEMENTS
C
1000 FORMAT (1H1///28X,79(1H*)/28X,79H*** STRFM FUNCTION, INTERIOR V
1ELOCITIES, VELOCITY COMPONENTS, AND ANGLES ***/44X,41H THAT ALL MESH
2 POINTS OF THE ORTHOGONAL MESH/44X,41(1H*)
1100 FORMAT (///42X,39H** HORIZONTAL ORTHOGONAL MESH LINE NO. ,
112,3H ***)
1020 FORMAT (1X,10H MESH-POINT,3X,5HAXIAL,8X,6HRADIAL,6X,6HSTREAM,4X,
16HMERID.,3X,9HREL.TANG.,4X,4HREL.,3X,9HCRIT.VEL.,3X,6HMERID.,3X,
28HRHEL.FLOW,3X,4HMESS/1X,9HCLM. ROW,4X,6HCOORD.,7X,6HCOORD.,7X,
35HFUNC.,5X,4HVFL.,6X,4HVFL.,7X,4HVFL.,5X,4HVFL.,5X,4HVFL.,5X,4HVFL.,
42X,4H (I), 3), 5X,3H (Z), 10X,3H (P), 10X,3H (U), 5X,4H (WM), 5X,5H (WTH),
57X,3H (W),5X,7H (W/C), 1X,7H (ALPHA), 3X,6H (BETA), 5X,5H (PHI)
1030 FORMAT (1X,12,2X,13,2X,2(G12.5,1X) ,1X,F8.2,3(1X,F8.2) ,IX,F8.2,
13 (3X,F7.2),2A4)
1040 FORMAT (1H1///15X,99(1H*)/15X,99H*** STREAM FUNCTION, INTERIOR V
1ELOCITIES, VELOCITY COMPONENTS, AND SURFACE VELOCITIES **
2*56X,17H ALONG STREAMLINES/56X,17(1H*)
1050 FORMAT (///36X,2CH** STREAMLINE NUMBER,13,23H -- 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,4HREL.FLOW,
22X,7HSTREAM,3X,9HVCURV.,1X,9HRRES.SUR./4X,6HCOORD.,7X,
36HCOORD.,7X,6HCOORD.,7X,4HVFL.,5X,4HVFL.,5X,4HVFL.,4X,5HRATIO,
42X,5HANGLE), 5X,5HCRV.,6X,4HVFL.,6X,4HVFL./5X,3H (2), 10X,3H (P),
51X,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,
15)F8.2,2A4)
1080 FORMAT (1H1///15X,99(1H*)/15X,99H*** STREAM FUNCTION, INTERIOR V
1ELOCITIES, VELOCITY COMPONENTS, ANGLES, AND SURFACE VELOCITIES **
2*28X,72H ALONG LINES FROM HUB TO SHROUD AT VARIOUS STATIONS THROUG
3H THE BLADE ROW/28X,72X (1H*)
1085 FORMAT (///28X,19H BEWARE. 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 (///9X,26H** HUB-SHROUD STATION NO. ,12,3H ***)
1100 FORMAT (///9X,26H** HUB-SHROUD STATION NO. ,12,3H ***,16X,
118H** LEADING EDGE ***)
1110 FORMAT (///9X,26H** HUB-SHROUD STATION NO. ,12,3H ***,15X,
119H** TRAILING EDGE ***)
1120 FORMAT (///9X,26H** HUB-SHROUD STATION NO. ,12,3H ***,15X,
118H** WITHIN BLADE ***)
1130 FORMAT (4X,6HRADIAL,7X,5HAXIAL,8X,6HMERID.,4X,6HSTREAM,3X,
16HMERID.,1X,9HREL.TANG.,2X,4HREL.,2X,9HCRIT.VEL.,2X,6HMERID.,2X,
28HRHEL.FLOW,2X,7HSTREAM,3X,9HVCURV.,1X,9HRRES.SUR./4X,
36HCOORD.,3X,6HCOORD.,7X,6HCOORD.,5X,5HFUNC.,4X,4HVFL.,5X,4HVFL.,
45X,4HVFL.,5X,5HRATIO,2X,4HVFL.,4X,4HVFL.,5X,4HVFL.,5X,4HVFL.,
55X,3H (B), 10X,3H (Z), 1X,3H (M), 6X,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,F9.2) ,1X,F7.3,2(2X,F7.2),2X,
1G11.4,F8.2,2X,F8.2))
SUBROUTINE BLDVEL
C
C--BLDVEL CALCULATES BLADE SURFACE VELOCITIES, BLADE-TO-BLADE
C--AVERAGE DENSITY, AND FT
C
COMMON NREAD,NWRIT,ITER,END,NWRT1,NWRT2,NWRT3,NWRT4,NWRT5,NWRT6
COMMON/INPUT/GAM,AR,MSPL,OMEGA,REDPAC,YELCOL,PNEW,NEWB,MRT,MO,
1 MM,MHT,NBL,NHUB,NTIP,MN,NOUT,MBI,MBO,1
2 LSPL,LTPL,LAMVT,LBOT,BLAD,LETEAN,ANGROT,IMESH,ISLINE,
3 ISTATL,ISUPER,ITSON,DEBUG,ZOMIN,ZOMBI,ZOMBO,ZOMOUT,
4 ROMIN,BOMBI,ROMBO,ROMOUT,ZMIN,ZTIN,ZHOUT,ZOUT,RHIN,RTIN,RHOUT,
5 RTOUT,TITLE (20),ZHUB (50),RHUB (50),ZTIP (50),RTIP (50),
6 RADIN,TIP (50),PRIP (50),LAMIN (50),VTHIN (50),SPOUT (50),
7 RADOUT,PROP (50),LOSSOUT (50),LAMOUT (50),VTHOUT (50),
8 BETALE (50),BETATE (50),ZTHST (50),ZTST (50),RHST (50),RTST (50),
9 PLFR (50),PERCRED (50),PERLOS (50),ZBL (50,50),RBL (50,50),
10 TBL (50,50),TBLB (50,50),TBL (50,50),TBLB (50,50),TH2BL (50,50)
COMMON/CALCON/MMM,I,MMPI,CP,EXPO,TGROG,PITCH,RLER,REL,RTET,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 DTHD (100,101),PLOSS (100,101),CPI (100,101),SPRI (100,101),
COMMON/VARCOM/MM1,MHTP1,CP,EXPON,TGROG,PITCH,RLER,REL,RTET,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 DTHD (100,101),PLOSS (100,101),CPI (100,101),SPRI (100,101),
COMMON/VARCOM/MM1,MHTP1,CP,EXPON,TGROG,PITCH,RLER,REL,RTET,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 DTHD (100,101),PLOSS (100,101),CPI (100,101),SPRI (100,101),

REAL LAMDAF
10 FCHANG = 0.
FMAX = -1.2E20
FMIN = 1.2E20

---CALCULATE DFDT---
DO 30 I=1,MM
DO 20 J=1,MMH1
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,MMH1,DFVERT)
DO 30 J=1,MMH1
DFDT(I,J) = DFVERT(J)
30 CONTINUE

---CALCULATE DFDS, THEN DFDM AND BLADE SURFACE VELOCITIES---
DO 50 J=1,MMH1
CALL SLOPES(SOM(1,J),FST(1,J),MM,DFDS)
DO 50 I=1,MM
DFDM(I,J) = 0.
IF (I.GE.IE(J).AND.I.LE.ITE(J)) DFDM(I,J) = -(DFDS(I)*CAMP(I,J) +
1*DFDT(I,J)*SAMP(I,J)*BTH(I,J)*COS(BETA(I,J))
50 CONTINUE

---CALCULATE BLADE-TO-BLADE AVERAGE DENSITY---
IF (GAM.EQ.0.) GO TO 40
TILMR = 2.*OMEGA*LAMDAF(TOM(I,J),I,J) - (OMEGA*ROM(I,J))**2
WSQ = WLSURF(I,J)**2
TIPIJ = TIPF(TOM(I,J),I,J)
TTIP = 1.-(WSQ*TILMR)/CP/TIPIJ/2.
IF(TTIP.LT.0.) TTIP = 0.
RHOIJ = RHOIPF(TOM(I,J))* (1.-PLOSS(I,J))
RHOL = RHOIJ*TTIP**EXPON
WSQ = WTSURF(I,J)**2
TTIP = 1.-(WSQ+TILMR)/CP/TIPIJ/2.
IF(TTIP.LT.0.) TTIP = 0.
RHOT = RHOIJ*TTIP**EXPON
DELH(RHOL) = RHOL-RHOT
RHOAV(I,J) = (RHOL+4.*RHO(I,J)+RHOT)/6.

---CALCULATE F-SUB-T FOR SUBROUTINE COEF---
PTT = W(I,J)/BTH(I,J)*DTHDT(I,J)*DFDM(I,J)
FCH = ABS(PTT-PT(I,J))
PCHANG = AMAX1(PCHANG,FCH)
P = PCHANG.EQ.FCH) ICH=I
IF (PCHANG.EQ.0.) JCH=J
FMAX = AMAX1(FMAX,PTT)
FMIN = AMIN1(FMIN,PTT)
PT(I,J) = FNEW*PTT+(1.-FNEW)*PT(I,J)
50 CONTINUE
IF (IEND.LT.I) WRITE(NWRIT, IC40) FCHANG,ICH,JCH,PT(ICH,JCH),
FMAX,FMIN
C---PRINT DEBUG OUTPUT IF REQUESTED
C
IF (IDEBUG.LE.0) RETURN
IF ((ITER/IDEBUG).EQ.0.AND.ITER.LE.ITER.NE.1) RETURN
WRITE (NWRT5,1020)
WRITE (NWRT5,1000) ((I,J,WSUBS(I,J),WSUBT(I,J),VTH(I,J),RHO(I,J),
1 RHOAV(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),
1 ZETOM(I,J),CAMP(I,J),SAMP(I,J),I=1,MM),J=1,MHTP1)
RETURN

STATEMENTS

FORMAT (215,8G15.5)
FORMAT (215,7G15.5)
FORMAT (1H1//35X,57(1H*)/35X,57H*

CHANGING QUANTITIES ON THE ORTHOGONAL MESH /35X,57(1H*)// 4X,1HJ,4X,1HJ,5X,
25HWSUBS,11X,5HWSUBT,11X,3HVTH,11X,3HRHO,11X,5HRHOAV,9X,6HDDELRH,
310X,4HDLDU,11X,5HPLSS)
1030 FORMAT (///;4X,1H1,;4X,1HJ,5X,5HDTHDS,IIX,2HFT,12X,_HDFDM,11X,
1;4HXIOM,11X,5HZETOM,10X,;4HCAMP,IIX,;4HSAMP)

FORMAT (/SX,22HMAXIMUM CHANGE IN FT =,G13.5,15X,6HAT I =,I3,
15H, J =,I3,1H,6X,10HWHERE PT =,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,MST2,OMEGA,REDFAC,VELTOL,FNEW,DNEW,MBI,MBO,
1 MM,MHT,WBL,RHUB,NTPN,NIN,NOUT,NBLPL,NPPP,NOSTAT,NSL,WLOSS,
2 LSFL,LPLS,LAMVT,LROT,LBLAD,LETAN,ANGROT,IMESH,ISLINE,
3 NSTATL,ITSON,IDEBUG,ZOMIN,ZOMBI,ZOMBO,ZOMOUT,
4 ROMIN,ROMBI,ROMBO,ROMOUT,ZHIN,ZTIN,ZHOUT,ZTOUT,RHIN,RTHIN,RHOUT,
5 RTOUT,TITLE2(20),2HUB(50),RHUB(50),2ZIP(50),RTIP(50),SFNT(50),
6 RADOUT(50),PROP(50),LAMOUT(50),VOOUT(50),RHOUT(50),
7 BETALE(50),BETATE(50),2HST(50),ZTST(50),RHST(50),FDST(50),
8 FLFR(50),PERT(50),PEROS(50),ZBL(50,50),RBL(50,50),
9 THBL(50,50),TNBL(50,50),TH2EL(50,50),TH2BL(50,50),
COMMON/CALCON/MMMI,
1 2MMMI,MMMI,CP,EXPN,TGROG,PITCH,REH,RELST,RETH,RETT,
1 ZLE(50),RLE(50),ZTE(50),RTE(50),ZLEOM(101),RLEOM(101),
2 SLEOM(101),THEL(101),ZTFEM(101),RTEOM(101),STEOM(101),
3 THEOM(101),ILE(101),ITE(101),ZOM(100,101),ROM(100,101),
4 SOM(100,101),TOM(100,101),BTH(100,101),DTDS(100,101),
5 DTHDTS(100,101),PLOSS(100,101),CPHI(100,101),SPHI(100,101),
6 ZHBL(50,50),ZLPL(50,50),ZBL(50,50),ZBL(50,50),
7 ALPSL(100,50),BETBSL(100,50),WBL(100,50),WWSL(100,50),
8 CURVSL(100,50),WLSSL(100,50),WTSSL(100,50),
COMMON/SLCOM/ILS(50),ITS(50),ZSL(100,50),RSL(100,50),NSL(100,50),
1 RZSL(100,50),RWSL(100,50),WWSL(100,50),WTHSL(100,50),
2 RZSL(100,50),BETBSL(100,50),WBSL(100,50),WWCSL(100,50),
3 CURVSL(100,50),WLSSL(100,50),WTSSL(100,50),
C---LEADING EDGE
CALL SPLINT(RLE,ZLE,NBLPL,RLE(1),1,ZSPL,DZDP,TEMP)
DO 20 J=1,NSL

154
I = 0
10 I = I + 1
   CALL SPLINT (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 SPLINT (RSL(I,J), 1, ZSPL, DZDR, TEMP)
   IF (ZSPL.GE.ZSL(I,J)) GO TO 30
40 ITS(J) = I - I
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, NWIT, ITER, IEND, NWIT1, NWIT2, NWIT3, NWIT4, NWIT5, NWIT6
COMMON/INPUT/GAM, AR, MSFL, OMEGA, REDFAC, VELTOL, FNEW, DNEW, MBI, MBO,
1 MM, MHT, NBL, NHUB, NTIP, HIN, NOUT, NBLPL, NPPP, NOSTAT, NSL, NLOSS,
2 LSFR, LTLPL, LAMVT, LROT, CLLAD, LETEAN, ANGROT, IMESH, ISLINE,
3 ISTATL, IMLPL, ITRAIL, ITSPL, ZOMIN, ZOMBI, ZOMB, ZOMOUT,
4 ROMIN, ROMBI, ROMBO, ROMOUT, ZHIN, ZTIN, ZHOUT, ZTOUT, RHIN, RTIN, RHOUT,
5 RTOUT, TITLE(2C), ZHUB(50), RHUB(50), ZTIP(50), RTIP(50), SPIN(50),
6 RADIN(50), TIP(50), PRIP(50), LAMIN(50), VTHIN(50), SFOUT(50),
7 RADOUT(50), TRIP(50), LOSOUT(50), LAMOUT(50), VTHOUT(50),
8 BETALE(50), BETATE(50), ZHST(50), ZTST(50), RHST(50), RTST(50),
9 PLFR(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, TGRG, PITCH, RLEH, LETH, RTEE, RTFT,
1 ZLE(50), ZLP(50), ZTE(50), RTE(50), ZLEOM(101), RLFO(101),
2 SLEOM(101), THLEOM(101), ZTEOM(101), RTLEOM(101), STLEOM(101),
3 THTEOM(101), ILE(101), TTE(101), ZOM(100,101), ROM(100,101),
4 SOM(100,101), TON(100,101), BTN(100,101), DTHDS(100,101),
5 DTHDT(100,101), PLOSS(100,101), CPHT(100,101), SPHT(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 WLDSURF(100,101), WTDSPRF(100,101), CAMP(100,101), SAMP(100,101),
5 RHODAV(100,101), DELRHO(100,101), PT(100,101), DFDM(100,101),
6 XION (100,101), ZETOM(100,101), DLLU(100,101)
COMMON/ROTATN/ZHROT(50), RHROT(50), ZTROT(50), RTROT(50),
1 ZLEOM(101), RLLEOM(101), ZTEOMR(101), RTEOMR(101),
2 ZBLROT(50,50), RBLROT(50,50), ZOMROT(100,101), ROMROT(100,101),
COMMON/INDCOM/NBLPC, NPCP, ZPCP(51,51), RPCP(51,51), TTPC(51,51),
1 THPC(51,51), DTHDZ(51,51), DTHDR(51,51), DTHLE(101), DTHTE(101),
2 DTBPL(101), DTBPLTE(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.EQ.ITER.OR.ITER.EQ.1) GO TO 40
30 WRITE(NWRT6,1060)
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)
IPRT = 1
C--CORRECT DTHDS, AND CALCULATE INCIDENCE AND DEVIATION, ROW BY ROW
C--FROM HUB TO TIP
C 40 DO 100 J=1,MHTP1
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)-
1*SOM(I,J))
CALL LININT(ZPC,RPC,DTHDL,NPPC,NBLPC,51,51,ZLEOMR(J),PLEOMR(J),
1DTDZLE,II,JJ)
CALL LININT(ZPC,RPC,DTHDR,NPPC,NBLPC,51,51,ZLEOMR(J),PLEOMR(J),
1DTDZRE,II,JJ)
TANBBL = (DLEOM(J)-SLEOM(J))/2.*((4.76631581*(SLEOM(J)-SLIDLE)/18.))
IF (ITER.EQ.1) BTBFLE(J) = ATAN(TANBBL)
BTBLLE = ATAN(TANBBL)*DEGRAD
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)-
1*SOM(I-1,J))
RHOF = RHO(I,J)+EXTRAP*(RHO(I,J)-RHO(I-1,J))/(SOM(I,J)-
1*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)*BTBFLE(J)/PITCH*RHOBF/RHOF
BETABL = ATAN(TANBBF)
C--CALCULATE DISTANCE FOR DTHDS CORRECTION
BLDCRD = (RLEOM(J)+RTEOM(J))/2.*(SLEOM(J)-STLEOM(J))
BLDCRD = SQRT(BLDCRD**2+ (STEOM(J)-SLEOM(J))**2)
SLIDLE = BLDCRD/PITCH/RLEOM(J)
DISTLE = AMIN1(1.5,AAMAX1(1./6., (11.-4.*SLIDLE)/18.))*(STEOM(J)-
1*SLEOM(J))
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/RO(W(I, J) -DTHDT (I, J)*SAMP(I, J))/CAMP(I, J)
I = I+1
GO TO 50
60 BTBFTE(J) = BETABF
C
--CALCULATE INCIDENCE ANGLES
BLINC = BETABF*DEGRAD-PDDLLE
UBINC = BETAPS*DEGRAD-PDDLLE
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) -
1*SOM(I-1, J))
CALL LININT(2PC,RPC,DTHDR,NPPC,NBLPC,51,51,ZTEOMR(J),RTEOMR(J),
1DTHDTE,II,JJ)
CALL LININT(2PC,RPC,DTHDR,NPPC,NBLPC,51,51,ZTEOMR(J),RTEOMR(J),
1DTHDTE,II,JJ)
TANBBL = RTEOM(J)*(DTHDTE*RHO(ALPHTE) +DTHDTE*COS(ALPHTE))
IF (ITER.EQ.1) BTBFTE(J) = ATAN(TANBBL)
BTETE = ATAN(TANBBL)*DEGRAD
C
--CALCULATE BLADE FLOW ANGLE AT TRAILING EDGE, CORRECTED FOR BLOCKAGE
BETAPS = ETA(I, J) +EXTRAP*(ETA(I, J) -ETA(I+1, J))/(SOM(I+1, J) -
1*SOM(I, J))
RHOFS = RHO(I, J) +EXTRAP*(RHO(I, J) -RHO(I+1, J))/(SOM(I+1, J) -
1*SOM(I, J))
RHOBF = RHOAV(I-1, J) +EXTRAP*(RHOAV(I-1, J) -RHOAV(I-2, J))
TANBBF = TAN(BETAPS)*BTHTE(J)/PI*CHeBHOBF/RHOFS
BETABF = ATAN(TANBBF)
C
--CALCULATE DISTANCE FOR DTHDS CORRECTION
SLIDTE = BLDCRD/PITCH/RTEOM(J)
DISTTE = AMIN1(.5,AMAX1(1./6.,(11.-SLIDTE)/15.))*(STEOM(J) -
1*STEOM(J))
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 +BETBFTE(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
--CALCULATE DEVIATION ANGLES
BDE = BETABF*DEGRAD-BTBLSE
UDE = BETAPS*DEGRAD-BTBLSE
C
--PRINT INCIDENCE AND DEVIATION ANGLES
IF (IPRINT.EQ.0) GO TO 100
IF ((LAMDAF(.5, ILE(1), II) -BVHTA(.5, ILE(1), II)).GT.0.) GO TO 97
BLINC = -BLINC
UBINC = -UBINC
BLDEV = -BLDEV
UBDEV = -UBDEV

90 WRITE(NWPT6,1340) J, BLINC, UBINC, BTBLLE, BLDEV, UBDEV, BTBLTE
100 CONTINUE
IF (IPRNT.GT.0) WRITE(NWRT6,1050)
RETURN

C--FORMAT STATEMENTS
C
1000 FORMAT (1H1///44X,40(1H*)/44X,40H*** INCIDENCE AND DEVIATION ANG
1LES ****/4X,3C(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,7HAT T.E.,3X,
39HUNBLOCKED,4X,7HAT T.E.,3X,1H*)
1040 FORMAT (24X,1H*,2X,I3,3X,2(IH*,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, NWRT6, ITER, IEND, NWRT1, NWRT2, NWRT3, NWRT4, NWRT5, NWRT6
COMMON INPUT/GAM, AR, MSF, OMEGA, REDFAC, VEITOL, FNEW, DNEW, MBI, MRO,
1 MM, MHT, NBL, ZHIN, ZOUT, ANGROT, IMESH, TSLINF,
1 ISTATL, IPLOT, ISUPPR, ITSON, TDEBUG, ZOMIN, ZOMB, ZOBL, ZOMOUT,
4 RMIN, RMBI, RCMB, RMCMT, ZHIN, ZOUT, ZTOUT, RHIN, RTIN, RHOUT,
5 STOUT, TITLEI (26), ZHUB (50), RHUB (50), ZTIE (50), RTIE (50), SPIN (50),
6 RADIN (50), RTP (50), PPPIP (50), LAMIN (50), VTHIN (50), VSPOT (50),
7 RADOUT (50), PROP (50), LOGOS (50), LAMOUT (50), VTHOUT (50),
8 BETALF (50), BETATE (50), ZHST (50), ZTST (50), RAST (50), RTST (50),
9 FLFR (50), PERCRD (50), PERLOS (50), ZBL (50, 50), RBL (50, 50),
10 TNBL (50, 50), TNLB (50, 50), TNE2 (50, 50), TNLB (50, 50), TNE2 (50, 50)
COMMON CALCION/MM1, MHTPL, CP, EXPON, TGPROG, PITCH, RLEH, SLET, RSH, PTET,
1 GLE (50), RLE (50), ZTE (50), RLE (50), ZLREOM (100), RLEOM (101),
2 SLEOM (101), TLEOM (101), ZTEOM (101), TPROM (101), STEOM (101),
3 TTHEOM (101), TLE (101), TBE (101), ZOIN (100, 101), ZOM (100, 101),
4 ZIN (100, 101), ZIN (100, 101), ZOUT (100, 101), ZOUT (100, 101),
5 ZTEND (100, 101), PLOSS (100, 101), ZBPHI (100, 101), ZPHI (100, 101),
6 ZTEND (100, 101), PLOSS (100, 101), CPHI (100, 101), SPHI (100, 101),
7 COMMON/VAPCOM/ (4, 100, 1), VCOM (100, 101), VCOM (100, 101),
1 WSUBS (100, 101), WSUB (100, 101), WSUBZ (100, 101), WSUB (100, 101),
2 WSOO (100, 101), WSOO (100, 101), WSOO (100, 101), WSOO (100, 101),
3 ALPHA (100, 101), BETA (100, 101), WWCW (100, 101), CURV (100, 101),
4 WLSUB (100, 101), WLSUB (100, 101), WSUBF (100, 101), WSUBF (100, 101),
5 RAOAV (100, 101), RAOAV (100, 101), PLOT (100, 101), PLOT (100, 101),
6 XTOH (100, 101), XTOH (100, 101), DLDO (100, 101),
7 COMMON/VBLSOM/ (50), ITS (50), ZSL (100, 50), BL (100, 50), MSL (100, 50),
1 WZSL (100, 50), WRSVL (100, 50), WMRL (100, 50), WRHSL (100, 50),
ALPSL(100,50), BESL(100,50), WSL(100,50), WWRSL(100,50),
CURVSL(100,50), WSSL(100,50), WTSSL(100,50),
COMMON/INDCOM/NBLPC, NPPC, ZPC(51,51), RPC(51,51), TTPC(51,51),
THPC(51,51), DTHD2(51,51), DTHDR(51,51), BTHLE(101), BTHTE(101),
BTSEP(101), BTSEP(101),
COMMON/ROTTN/ZHROT(50), RHROT(50), ZTROT(50), RTROT(50),
ALPHA(101), LEOMR(101), ZTEOMR(101), BTEOMR(101),
COMMON/ZROTTN/ZHROT(50), RHROT(50), ZTROT(50), RTROT(50),
DIMENSION ZMSFL(100), ZMSP2(100), THSPI(100), THSP2(100), DTM1(100),
DSTM1(100), D2TSTM1(100), DTM2(100), D2TSTM2(100), CURV1(100), CURV2(100),
RADSPI(100), RADSP2(100), ALPHSP(100), ZMSNP(100), RMSNP(100),
BESP(100), PLOSSL(100), DBDM(100), D2BDM2(100), DTSN(100),
AAA(100), BBB(100),
REAL MSFL, MSL, LAMDAF

C--PRELIMINARY CALCULATIONS
C
IF (ITSON.LE.0) RETURN
IF ((ITER/ITSON)*ITSON).NE.ITER) RETURN
WRITE(NWRT_,1900)
IF (REDFAC.LT.1.0) WRITE(NWRT_,1010) ITER
IF (REDFAC.EQ.1.0.AND.IEND.LE.0) WRITE(NWRT_,1020)
IF (REDFAC.EQ.1.0.AND.IEND.GE.1.AND.ISUPER.LE.1) WRITE(NWRT_,1030)
IF (REDFAC.EQ.1.0.AND.IEND.GE.1.AND.ISUPER.EQ.2) WRITE(NWRT_,1040)
ARTEM = AR
ZMSFL = MSFL/100./REDFAC
OMTEM = OMEGA/REDFAC
REDTM = 1.0
WELTEM = 1.0
MBITS = 21
MBOTS = 61
MMTS = 81
MBBI = 20
NRSP = MM
NSLTS = 0
LRVB = 0
LOSS = 0
IF (NLOSS.GT.0) LOSS=1
LCWR = 1
LIPS = 0
TIMESH = 0
IISLIN = 5
ITBSUR = 1
IIPLOT = 1
IDEBG = 0
DEGRAD = 180./3.1415927
BFACTR = 1.0
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 = TIPP(FIPR(JS))
RHOP = RHOPP(FIPR(JS))
RVTTHI = LAMDAF(FIPR(JS), ILE(1),1)/REDFAC
RVTTHO = RVTHTA(FIPR(JS), ILE(1),1)/REDFAC
IF (GAM.NE.0.) GC TO 5
ARTEM = 0.
TIPTEM = 0.
RHOFP = AR

C--INTERSECTION OF STREAMLINE WITH BLADE LEADING AND TRAILING EDGES
5 CALL INRSCT(ZSL(1,JS),RSL(1,JS),MM,ZLE,RL,ISBLISL,RTSL,SL,RTPL)
   CALL INRSCT(ZSL(1,JS),RSL(1,JS),MM,ZTE,ZT,ISBLISL,RTTSL,RTPL)

C--CALCULATE STREAMSHEET LOCATION AND THICKNESS, AND LOSS DISTRIBUTION
   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),
                   RSL(IS,JS),RHOFL,II,JJ)
       CALL LININT(ZOMROT,ROMROT,BTH,MM,MHTP1,100,101,ZSL(IS,JS),
                   RSL(IS,JS),BTHSL,II,JJ)
       CALL LININT(ZOMROT,ROMROT,PLOSS,MM,MHTP1,100,101,ZSL(IS,JS),
                   RSL(IS,JS),PLOSSL,II,JJ)
       CALL LININT(ZOMROT,ROMROT,DELH,MM,MHTP1,100,101,ZSL(IS,JS),
                   RSL(IS,JS),THL,II,JJ)
       CALL LININT(ZOMROT,ROMROT,DELSEL,MM,MHTP1,100,101,ZSL(IS,JS),
                   RSL(IS,JS),THSL,II,JJ)
       CALL LININT(ZOMROT,ROMROT,PLOSS,MM,MHTP1,100,101,ZSL(IS,JS),
                   RSL(IS,JS),PLOSST,II,JJ)
   10 BESP(IS) = ZMSFL/RMWAV/RMSP(IS)/RTMSP*BFACRT
   ZMSFL = ZMSFL*BFACRT

C--CALCULATE BLADE SURFACE COORDINATES WITH RESPECT TO MERIDL ORIGIN
C--AT ALL POINTS ON BLADE WHERE VERTICAL ORTHOGONALS PASS THROUGH
C--THE STREAMLINE
II = 1
JJ = 1
NBLPTS = ITS(JS)-ILS(JS)+3
ILSJ = ILS(JS)
ITSJ = ITS(JS)
ZMSP1(1) = 0.
DELM = SQRT((ZSL(ILSJ,JS)-ZLSEL)**2+(RSL(ILSJ,JS)-RLESL)**2)
   CALL LININT(ZPC,RPC,THPC,NPPC,NBLPC,51,51,ZLESL,RLESL,
               THLESL,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),
                   RSL(IS,JS),THS,II,JJ)
       CALL LININT(ZPC,RPC,THPC,NPPC,NBLPC,51,51,ZSL(IS,JS),
                   RSL(IS,JS),THSII,II,JJ)
       CALL LININT(ZPC,RPC,THPC,NPPC,NBLPC,51,51,ZSL(IS,JS),
                   RSL(IS,JS),THSII,II,JJ)
       THS1(ISB) = THS1-THLESL+DBL/2.
       THS2(ISB) = THS2-THLESL-DBL/2.
   20 ISB = ISB+1
   DELM = SQRT((2TESL-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,
               THTESL,II,JJ)
   CALL LININT(ZPC,RPC,THPC,NPPC,NBLPC,51,51,ZTESL,RTESL,
1DBL,II,JJ)
IF (DBL.LT.CHORD/1000.) DBL=CHORD/1000.
THSP1(1) = DBL/2.
THSP2(1) = -DBL/2.
CALL LININT(ZEC,RP,C,TPC,NPPC,NBLPC,51,51,RTESL,RTESL,
1DBL,II,JJ)
IF (DBL.LT.CHORD/1000.) DBL=CHORD/1000.
THSP1(NBLPTS) = TMESL-THLES1+DBL/2.
THSP2(NBLPTS) = THLES1-THLES1-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
C
CALL SPLINE(ZMRSP,BESP,MM,DRDM,D2DRDM)
C
C--ELIMINATE ANY BLADE SURFACE POINTS VERY CLOSE TO THE
C--LEADING AND TRAILING EDGES
C
ILSJ1 = ILS(JS)
ILSJ2 = ILS(JS)
DELMSP = C.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,D2DTDM1)
CALL SPLINE(ZMSP2,THSP2,NBLPTS,DTDM2,D2DTDM2)
CALL SPLINT(ZMSP1,RMS,MM,0.,1,RADLE,TEM1,TEM2)
CALL SPLINT(ZMSP2,RMS,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 BETII = ATAN(RADLE*DTDM1(1))
\begin{align*}
\text{BETI2} & = \text{ATAN} (\text{RADLE} \times DTDM2 (1)) \\
\text{RI1} & = \text{RADLE} \times (\text{THSPI} (1) - \text{THSP2} (1)) \times \cos ((\text{BETI1} + \text{BETI2}) / 2) / 2. \\
\text{ZLTAN1} & = \text{RI1} \times (1 + \sin (\text{BETI1})) \\
\text{ZLTAN2} & = \text{RI1} \times (1 + \sin (\text{BETI2})) \\
\text{CALL SPLINT} (\text{ZMSPI}, \text{THSPI}, \text{NBLPTS}, \text{ZLTAN1}, 1, \text{TLTAN1}, \text{DTAN1}, \text{TEM1}) \\
\text{CALL SPLINT} (\text{ZMSPI}, \text{THSP2}, \text{NBLPTS}, \text{ZLTAN2}, 1, \text{TLTAN2}, \text{DTAN2}, \text{TEM2}) \\
\text{BETI1} & = \text{ATAN} (\text{RADLE} \times \text{DTAN1}) \\
\text{BETI2} & = \text{ATAN} (\text{RADLE} \times \text{DTAN2}) \\
\text{RI1NEW} & = \text{RADLE} \times (\text{TLTAN1} - \text{TLTAN2}) / (\cos (\text{BETI1}) \times \cos (\text{BETI2})) \\
\text{IF} (\text{ABS} ((\text{RI1NEW} - \text{RI1}) / \text{RI1}) \lt 0.001) \text{ GO TO 117} \\
\text{ICOUNT} & = \text{ICOUNT} + 1 \\
\text{IF} (\text{ICOUNT} \lt 100) \text{ GO TO 100} \\
\text{WRITE} (\text{NWPT4}, 1200) \\
\text{GO TO 110} \\
\text{RI1} & = (\text{DAMP} \times \text{RI1} \times \text{RI1NEW}) / (\text{DAMP} + 1) \\
\text{IF} (\text{RI1} \lt 0.0) \text{ GC TO 90} \\
\text{DAMP} & = \text{DAMP} + 1. \\
\text{GO TO 80} \\
\text{RI1} & = \text{RI1NEW} \\
\text{RI2} & = \text{RI1} \\
\text{C} \\
\text{C--CALCULATE T-PAILING EDGE RADIUS, POINTS OF TANGENCY, AND} \\
\text{C--TANGENCY ANGLES} \\
\text{C} \\
\text{ICOUNT} & = 0 \\
\text{DAMP} & = 1. \\
\text{120} \\
\text{BETO1} & = \text{ATAN} (\text{RADTE} \times \text{DTDM1} (1)) \\
\text{BETO2} & = \text{ATAN} (\text{RADTE} \times \text{DTDM2} (1)) \\
\text{RO1} & = \text{RADTE} \times (\text{THSPI} (1) - \text{THSP2} (1)) \times \cos ((\text{BETO1} + \text{BETO2}) / 2) / 2. \\
\text{ZTTAN1} & = \text{CHCRD} \times \text{RO1} \times (1 + \sin (\text{BETO1})) \\
\text{ZTTAN2} & = \text{CHCRD} \times \text{RO1} \times (1 + \sin (\text{BETO2})) \\
\text{CALL SPLINT} (\text{ZMSPI}, \text{THSPI}, \text{NBLPTS}, \text{ZTTAN1}, 1, \text{TTTAN1}, \text{DTAN1}, \text{TEM1}) \\
\text{CALL SPLINT} (\text{ZMSPI}, \text{THSP2}, \text{NBLPTS}, \text{ZTTAN2}, 1, \text{TTTAN2}, \text{DTAN2}, \text{TEM2}) \\
\text{BETO1} & = \text{ATAN} (\text{RADTE} \times \text{DTAN1}) \\
\text{BETO2} & = \text{ATAN} (\text{RADTE} \times \text{DTAN2}) \\
\text{RO1NEW} & = \text{RADTE} \times (\text{TTTAN1} - \text{TTTAN2}) / (\cos (\text{BETO1}) \times \cos (\text{BETO2})) \\
\text{IF} (\text{ABS} ((\text{RO1NEW} - \text{RO1}) / \text{RO1}) \lt 0.001) \text{ GO TO 150} \\
\text{ICOUNT} & = \text{ICOUNT} + 1 \\
\text{IF} (\text{ICOUNT} \lt 100) \text{ GO TO 140} \\
\text{WRITE} (\text{NWPT4}, 1210) \\
\text{GO TO 150} \\
\text{140} \\
\text{RO1} & = (\text{DAMP} \times \text{RO1} \times \text{RO1NEW}) / (\text{DAMP} + 1) \\
\text{IF} (\text{RO1} \lt 0.0) \text{ GC TO 130} \\
\text{DAMP} & = \text{DAMP} + 1. \\
\text{GO TO 120} \\
\text{150} \\
\text{RO1} & = \text{RO1NEW} \\
\text{RO2} & = \text{RO1} \\
\text{C} \\
\text{C--SUBSTITUTE POINTS OF TANGENCY FOR FIRST AND LAST POINTS IN} \\
\text{C--SURFACE COORDINATE ARRAYS} \\
\text{C} \\
\text{ZMSPI(1)} & = \text{ZLTAN1} \\
\text{ZMSPI(1)} & = \text{ZLTAN2} \\
\text{ZMSPI(NBLPTS)} & = \text{ZTTAN1} \\
\text{ZMSPI(NBLPTS)} & = \text{ZTTAN2} \\
\text{THSPI(1)} & = \text{TLTAN1} \\
\text{THSPI(1)} & = \text{TLTAN2} \\
\text{THSP2(1)} & = \text{TTTAN1} \\
\text{THSP2(1)} & = \text{TTTAN2} \\
\text{NPL1} & = \text{NBLPTS} \\
\text{162}
NSPL2 = NBLPTS

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 (ZMSPI(2).GT.ZMSPI(1)+DELMSP) GO TO 180
   DO 170 IS=3,NSPL1
   ZMSP1(IS-1) = ZMSP1(IS)
   THSP1(IS-1) = THSP1(IS)
   NSPL1 = NSPL1-1
   ILSJ1 = ILSJ1+1
   GO TO 160
170 IF (ZMSP2(IS-1).GT.ZMSP2(IS)+DELMSP) GO TO 190
   DO 180 IS=3,NSPL2
   ZMSP2(IS-1) = ZMSP2(IS)
   THSP2(IS-1) = THSP2(IS)
   NSPL2 = NSPL2-1
   ILSJ2 = ILSJ2+1
   GO TO 170
180 IF (ZMSPI(NSPL1-1).LT.ZMSPI(NSPL1)-DELMSP) ZMSPI(NSPL1-1) = ZMSPI(NSPL1)
   THSPI(NSPL1-1) = THSPI(NSPL1)
   NSPL1 = NSPL1-1
   GO TO 200
190 IF (ZMSP2(NSPL2-1).LT.ZMSP2(NSPL2)-DFLMSP) ZMSP2(NSPL2-1) = ZMSP2(NSPL2)
   THSP2(NSPL2-1) = THSP2(NSPL2)
   NSPL2 = NSPL2-1
   GO TO 210
200 IF (ZMSPI(NSPL1).LT.ZMSPI(NSPL1)-DELMSP) ZMSPI(NSPL1) = ZMSPI(NSPL1)
   THSPI(NSPL1) = THSPI(NSPL1)
   NSPL1 = NSPL1-1
   GO TO 200
210 C--CALCULATE TANGENTIAL COORDINATE SHIFT FROM MERIDIAN ORIGIN TO
C--TSOIC ORIGIN AND SHIFT COORDINATES
C

220 DELTH = (TLTAN1*COS(BETI2)+TLTAN2*COS(BETI1))/(COS(BETI1)+
   1*COS(BETI2))
   DO 230 IS=1,NSPL1
   THSPI(IS) = THSPI(IS)-DELTH
   DO 240 IS=1,NSPL2
   THSP2(IS) = THSP2(IS)-DELTH
   C--CALCULATE STAGGER AND STACKING COORDINATE
C

STGR = (TTTAN1*COS(BETO2)+TTTAN2*COS(BETO1))/(COS(BETO1)+
   1*COS(BETO2)) -DELTH
   THSTAK = THLESL+DELTH
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--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), TEM1, TEM2)
DO 260 IS=1, NSPL1
260 CURV1(IS) = (RADSPI(IS)*D2TDMI(IS)+SIN(ALPHSP(IS))*DTDM2(IS))/
1(1.+(RADSPI(IS)*DTDM2(IS))^2)**1.5
C--CALCULATE SLOPES, SECOND DERIVATIVES, AND CURVATURES ON LOWER
C--BLADE SURFACE
C SLOPE1 = TAN(BETI2)/RADSP2(1)
SLOPEN = TAN(BETI2)/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), TEM1, TEM2)
DO 280 IS=1, NSPL2
280 CURV2(IS) = (RADSP2(IS)*D2TDM2(IS)+SIN(ALPHSP(IS))*DTDM2(IS))/
1.0(1.+(RADSP2(IS)*DTDM2(IS))^2)**1.5
C--PRINT TSONIC DATA
C WRITE(NWRT4,1050) JS, FLFR(JS)
IF (BFACTR.NE.1.0) WRITE(NWRT4, 1055)
WRITE(NWRT4,1060) GAM, ARTEM, TIPTEM, RHOIP, OMTEN, ZMSPL
WRITE(NWRT4,1070) VELTEM
WRITE(NWRT4,1080) NBL, NSPL1, NSPL2, NRSP
WRITE(NWRT4,1090) LRVB, LOSS, LWCR, LIPS
WRITE(NWRT4,1100)
WRITE(NWRT4,1110) RVTHI, RVTHO
WRITE(NWRT4,1110)
WRITE(NWRT4,1120)
WRITE(NWRT4,1130) CHORD, STGR, DELTH, THSTAK
WRITE(NWRT4,1140) RI1, BETI1, RO1, BETO1
WRITE(NWRT4,1150) RI2, BETI2, RO2, BETO2
WRITE(NWRT4,1160)
WRITE (NWRT4, 1370) (IS, ZMSP2 (IS), THSP2 (IS), DTM2 (IS), D2TM2 (IS),
1 CURV2 (IS), RADSP2 (IS), THSP2 (IS), DTDM2 (IS), D2TM2 (IS),
WRITE (NWRT4, 1160)
WRITE (NWRT4, 1380) (IS, ZMRSP2 (IS), RMSP (IS), BESP (IS), WWCRSL (IS, JS),
WRITE (NWRT4, 1170)
WRITE (NWRT4, 1300)
输出再次以ANIACCARD图象格式输出

C--WRITE OUTPUT AGAIN IN CARD IMAGE FORMAT

30C ICARDS = 0
NWRT7 = 6
IF (ICARDS.EQ.0) GO TO 31C
WRITE (NWRT7, 1400) JS, FLFR (JS)
WRITE (NWRT7, 1450) GAM, ARTEN, TIPTEM, RHOIP, OMTEM, ZMSFL
WRITE (NWRT7, 1460) REDETEM, VELTEM
WRITE (NWRT7, 1440) MBETS, MBOTS, MMTS, MBBI, NSPL1, NSPL2, NSPL3, NRRSP, NSLTS
WRITE (NWRT7, 1460) RVTHI, RVTHO
WRITE (NWRT7, 1470) CHORD, STGR
WRITE (NWRT7, 1480) RI1, BFTI1, 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, 1430) (RMSP (IS), IS=1, MM)
WRITE (NWRT7, 1420) (BESP (IS), IS=1, MM)
WRITE (NWRT7, 1430) (WWCRSL (IS, JS), IS=1, MM)
WRITE (NWRT7, 1410) (PLOSSL (IS), IS=1, MM)
WRITE (NWRT7, 1420) IMESH, IISLIN, IIBSUR, TIPL, TIDERG
IF (NWRT7.EQ. NWRT4) WRITE (NWRT7, 1330) CONTINUE
RETURN

C--FORMAT STATEMENTS

1000 FORMAT (///45X, 39(1H*)/45X, 39H*** INPUT DATA FOR TSONIC PROGRAM
***/50X, 29(1H*)///<
1010 FORMAT (///53X, 23(1H*)/53X, 23H* REDUCED MASSFLOW */53X, 23(1H*)/
153X, 18H* ITERATION NO. */53X, 23(1H*)///<
1020 FORMAT (///52X, 25(1H*)/52X, 25H* FULL MASSFLOW */52X, 25(1H*)/
152X, 19H* ITERATION NO. */52X, 25(1H*)///<
1030 FORMAT (///52X, 25(1H*)/52X, 25H* FULL MASSFLOW */52X, 25(1H*)/
142X, 1H*, 12X, 19TRANSONIC 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 */52X, 25(1H*)/
142X, 1H*, 12X, 19TRANSONIC 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, 1H*, 28H* TSONIC INPUT -- STREAMLINE NUMBER, I3
1, 23H -- STREAM FUNCTION = F8.4, 4H* */2X, 110H(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 MERID COORDIN
5ATES, 16X, 1H*/2X, 95H* WHICH HAVE A DIFFERENT ORIGIN, HAVE BEEN SHI
6PED BY THE PROGRAM TO GIVE THESE TSONIC INPUTS. */14X, 1H*/2X,
7110(1H*)///<
SUBROUTINE SLPLOT

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

166
C--PLOT THE ITERATION NUMBER
CALL LRCHSZ (4)
CALL LRGRID (1, 1, 1, 0, 0, 0)
CALL LRCNH (ITER, 1, TITL (11), 1, 1, 1)
IF (IEND.EQ.0) CALL LLEGEN (TITL4, 4, 4.2, 6.0, 1, 0)
IF (IEND.GT.0) CALL LLEGEN (TITL5, 2, 1, 0.4, 2.5, 1, 0)
C--PLOT BLADE GEOMETRY AND STREAMLINES
C
CALL LRMRGN (1, 1, 1, 0, 0, 0)
CALL LRANGE (ZLRNG, ZRRNG, XBRNG, BBNG, RTRNG)
CALL LRGRID (-1, -1, 1, 0, 0, 1)
CALL LRLGEN (TITL1, 1, 3, 5, 1, 0)
CALL LRCHSZ (2)
CALL LRLGEN (TITL2, 1, 0, 4.5, 1.5, 0)
CALL LRLGEN (TITL3, 1, 1, 0, 4.5, 0)
CALL LRCHSZ (4)
CALL LRCURV (ZFLPLT, RFPLT, 100, 2, SYM, 0.0)
CALL LRCURV (ZSPLT, RSPLT, 100, 2, SYM, 0.0)
IF (MTI.EQ.0) GO TO 5
CALL LRCURV (ZDLPLT, RDPLT, 100, 2, SYM, 0.0)
CALL LRCURV (ZETPLT, RSTPLT, 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
10 IF (JS.EQ.NSL) EOP = 1.0
CALL LRCURV (ZSL, 1, JS, RSL, MM, 2, SYM, EOP)
CALL LRCURV (ZSL, RSL, 6, 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, NWPT1, NWPT2, NWPT3, NWPT4, NWPT5, NWPT6
COMMON/INPUTT/GAM, AR, MSPL, OMEGA, REDPAC, VELTOL, FNEW, DNEW, MBI, QDO,
1 MM, MNT, NBL, NIGOD, NTIP, NIN, NOUT, NBLPL, NPPP, NSTAT, NSL, NLOS,
2 LISFR, LTPL, LAMVT, LROT, LBLAD, LETEA, NANGCT, IMESH, ISLINE,
3 NTSALL, IPECT, ITSTR, ITBLUG, ZOMIN, ZOMRI, ZOMBO, ZOMOUT,
4 NROMIN, NROMBI, NROMBO, NROMOUT, ZFIN, ZHTIN, ZHTOUT, ZFIN, ZHTIN, ZHTOUT,
5 ZFIN, ZHTIN, ZHTOUT, ZFIN, ZHTIN, ZHTOUT,
6 RADOUT(50), TIP(50), PRIP(50), LAMVT(50), VTHIN(50), VTHOUT(50),
7 RADOUT(50), TIP(50), PRIP(50), LAMVT(50), VTHIN(50), VTHOUT(50),
8 BETALE(50), BETATE(50), ZHST(50), ZHTST(50), FSTH(50), BSTH(50),
9 FLPF(50), PERCRD(50), PERSD(50), ZBL(50, 50), RBL(50, 50),
1 NTHBL(50, 50), NBBL(50, 50), THBL(50, 50), TRBL(50, 50), TBL(50, 50),
COMMON/SLCOM/ILS(50), ITP(50), ZSL(100, 50), RSL(100, 50), MSL(100, 50),
1 WSL(100, 50), WRSL(100, 50), WSHL(100, 50), WTHSL(100, 50),
2 ALPSL(100, 50), BETSIM(100, 50), WSL(100, 50), WRSL(100, 50),
3 CURVSL(100, 50), WLSL(100, 50), WTSVL(100, 50),
DIMENSION TITL1(12), TITL2(6), TITL3(14), TITL4(15),
1 TITL5(16), TITL6(6), TITL7(2),
REAL MSL, LRNG,
DATA TITL1/'EBI', 'DION', 'AL A', 'ND S', 'URPA', 'CRED', '1E1', 'EPLA'
1, 'TILE', 'VEL', 'OCIT', 'IFS /
DATA TITL2/'ST', 'REAM', 'LINE', 'ND', 'XXX', '
1, 'TITL3/'MERI', 'DION', 'AL P', 'ELAT', 'TVE', 'VELO', 'CITL', 'FSC'
1, '1RS', 'FOR', 'ALL ', 'STRE', 'AML', 'NES /
DATA TITL4/'SUCT', 'ION ', 'SUPF ', 'ACE', 'RFLA', 'TIVE', 'VEL', 'OCIT'
1, 'IES', 'CIR', 'IFOR', 'ALL ', 'STRE', 'AML', 'INES /
DATA TITL5/'PRES', 'SURE', 'SURE', 'SURE', 'REL', 'ATT', 'VE', 'OCIT'
1, 'TIES', 'SCT', 'PREP', 'PAR', 'AL ', 'STRE', 'AML', 'INES /
DATA TITL6/'ME', 'RDI', 'DNA', 'CO', 'ORDI', 'NAT /
DATA TITL7/VELO', 'CITY /
DATA SYM/XX/
IF (IPLT.LE.0) RETURN
IF (((IPLT/IPLT) * IPLT, NE, ITER, AND, ITER, NE, 1) RETURN
C
C--COMPUTE RANGE OF PLTS, 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(1, JS))
IF (MB1.EQ.0) GO TO 15
ILSJ = ILS(JS)
168
ITSJ = ITS(JS)
DO 10 IS = ILSJ, ITSJ
   BRNG = AMIN1(BRNG, WLSSL(IS, JS))
   TITNG = AMAX1(TITNG, WLSSL(IS, JS))
10   CONTINUE
   CALL LRLEGN(TITL3, 3, 6, 0, 1.7, 0.7, 0.0)
CALL LRCHSZ(2)
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
   CALL LRCHSZ(1)
   IF (MBI .EQ. 0) I?E?URN
50   CONTINUE
C--PLOT MERIDIONAL VELOCITIES FOR ALL STREAMLINES
C
   CALL LRGRID(3, 3, 11.0, 11.0)
   CALL LRLEGN(TITL3, 3, 6, 0, 1.7, 0.7, 0.0)
   CALL LRCHSZ(2)
   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
   CALL LRCHSZ(1)
50   CONTINUE
C--PLOT SUCTION SURFACE VELOCITIES FOR ALL STREAMLINES
C
   CALL LRCHSZ(4)
   CALL LRLEGN(TITL4, 6, 0, 1.2, 0.7, 0.0)
   CALL LRCHSZ(2)
CALL LLEGEN(TITL6,24,", , 1,3,5,7")
CALL LLEGEN("TITL7,4,1,2,4,9,0")
CALL LCHS2(4)
END = 6
DO JS=1,NSL
IF (JS.EQ.NSL) END = 1
ILSJ = TLS(JS)
MRLD = ITS(JS) - ILS(JS) + 1
6 CALL LYPYR(0LSL(0LSJ,JS)) , WLSL(LSLJ,JS), MRLD,2,SYM,EOP)
C CALL LCHS2(4)
RETURN
END

C SUPPORT Routine TVLCL
C
C TVLCL calculates the full massflow, transonic solution
C using velocity gradient equations
C
COMMON NNPAD,NWRIT,ITER,INDM1,NWR12,NWP1,\NWP3,NWP4,NWP5,NWP6
COMMON/INPUT/GAM,ALF,NSL,OMEG,A,REDAC,VEL10,FNEW, DNEW,MIB,MPO,\1
1. MIB,KIB,KN,NETT,NNOUT,0MP, NNP, NOSTAT,NSL,NLOSS,
2. LSPP,LPPL,LAMT, IROT, TPLAD, LETEAN, ANGROT, IMESH, LSLINE,
3. ISTAT, IPTYP, ITSPER, ITSON, IDEBEG, ZOMIN, ZOMBI, ZOMBO, ZOMOUT,
4. ZOMIN, ROBIL, ROMBO, ROMOUT, ZTHN, ZTIN, ZHOUT, ZTONT, ZTHN, RTIN, RHOUT,
5. ZTONT, TITL6(2C), ZHOUT(5C), ZHUB(5C), ZTIP(5C), ZTP(5C), SF (5C),
6. RADIN(5C), TIP(5C), TIP(5C), LAMIN(5C), VTHIN(5C), SPFOU(5C),
7. ADOUT(5C), PRCO(5C), LCSQ(5C), LAMOUT(5C), VTH(5C),
8. BETAK(5C), BETAT(5C), ZBST(5C), ZNST(5C), PHST(5C), RST(5C),
9. FLR(5C), PER(5C), PERL(5C), B(5C), B(5C),
10. THBL(5C), THBL(5C), THBL(5C), THBL(5C), TH2BL(5C), TH2BL(5C),
11. COMMON/CALCON/MM1, MHTP1, CP, EXPON, TLOG, PITCH, RLH, BPL, RTHER, PTET,
12. ZLE(5C), RLE(5C), ZTE(5C), RTE(5C), ZLEOM(101), RLEOM(101),
13. SLEOM(101), SLHOM(101), ZTRH(101), RTOM(101), STEM(101),
14. STM(101), TLM(101), ILT(101), ITZ(101), ZOM(100,101),РОМ(100,101),
15. ZOM(100,101), ZOM(100,101), RTH(100,101), DTHDS(101,101),
16. DTHDS(101,101), DTHDS(101,101), CPHI(100,101), SPHI(100,101),
17. COMMON/VARCC./MM1, I1,1C1, 1C1, MM1, 1C1, 1C1, MM1, 1C1, 1C1,
18. MM1, 1C1, 1C1, MM1, 1C1, 1C1, MM1, 1C1, 1C1,
19. MM1, 1C1, 1C1, MM1, 1C1, 1C1, MM1, 1C1, 1C1,
20. ALPHA(100,101), RETA(100,101), WMCR(100,101), CURV(100,101),

170
C--RESTORE PULL MASS FLOW VALUES, AND RETHINITIALIZE LAMDAF AND RTHTA

C
IEND = IEND+1
JZ = 1
IF (ISUPER.EQ.2) JZ=2
IF (ISUPER.EQ.2) GO TO 55
WRITE(NWPT,140)
OMEGA = OMEGA/REDFAC
MSFL = MSFL/REDFAC
DO 10 J =1,NIN
LAMIN(J) = LAMIN(J)/REDFAC
VTHIN(J) = VTHIN(J)/REDFAC
DO 20 J =1,NOUT
LAMOUT(J) = LAMOUT(J)/REDFAC
VTHOUT(J) = VTHOUT(J)/REDFAC
CALL LAMNIT
IF (MBI.NE.C) CALL RTNIT

C--CALCULATE PARTIALS WITH RESPECT TO T OF WSUBM AND WTH
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,DWVERT)
CALL SLOPES(TVERT,WTVERT,MHTP1,DWTVERT)
DO 40 J =1,MHTP1
DWMDT(I,J) = DWVERT(J)
DWDT(I,J) = DWTVERT(J)

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,DWMDM)
CALL SLOPES(SOM(1,J),WTH(1,J),MM,DWDTDS)
DO 50 I=1,MM
DWMDM(I,J) = (DWMDM(I) *CAMP(I,J) +DWMDT(I,J) *SAMP(I,J))/REDFAC
DWDT(I,J) = (DWDTDS(I) *CAMP(I,J) +DWDTDT(I,J) *SAMP(I,J))/REDFAC
RTOLER = 1.E-4
MEAN = MHT/2+1
55
CHLIM = MSFL*FLOAT(NBL)
UNE4(1) = 0.

C--INITIALIZE VARIABLES FOR LOOP ON VERTICAL MESH LINES

C

C--DIMENSIONS
C
LINC = 0
ICOUNT = 0
IREVRS = 0
ITEMIN = MM
ITEMP = MM
IMAX = 0
INC = 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
   DFLMAX = W(I,MEAN)/20./REDFAC
   WMAX = WHUB
   WMIN = WHUB
   MAXFLO = -1.0E6
   MINFLO = 1.0E6
   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=I,MHTP1
   LAMDA(J) = LAMDAF(UOM(I,J),I,J)
   IF (MBI NE 0) LAMBDG(J) = BVTHTA(UOM(I,J),I,J)
   TPT(J) = TIPF(UOM(I,J))
   RHI(J) = RHOFP(UOM(I,J)) * (1. - LOSS(I,J))
   BTVEL(J) = 0.
   CTVEL(J) = 0.
   DTVEL(J) = 0.
   IF (I.LT.ILE(J) OR I.GT.ITE(J)) GO TO 72
   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) / 
             1-ROM(I,J) + DTHT(I,J) * SAL*CBETA*SBETA
   DTVEL(J) = CBETA*SAMP(I,J)*DWMDM(I,J) - 2.*OMGA*SBETA*CPHI(I,J) 
             1+ROM(I,J) * DTHTD(I,J) * CBETA * (DWTDM(I,J) + 2.*OMGA*SAL)
   GO TO 72
70 ATVEL(J) = CAMP(I,J)*CURV(I,J)
   BTVEL(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.I) GO TO 80
   IMAX = I
   IREVRS = 1
   ICOUNT = 0
   GO TO 80
75 IF (I.EQ.ITE(J)) CALL TINDV(ITEMP,J,ICOUNT)

172
ITEMIN = MIN(TITEMIN,ITEMP)
80 CONTINUE
   IF (LINC.EQ.1) INC=-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*LAMBD(A(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.EQ.ILE(J)) GO TO 120
   WHIRL = LAMBDA(J)
   110 CTVEL(J) = -(WHIRL-OMR2)/ROM(I,J)**2*(CURV(I,J)*(WHIRL-OMP2) *
   CPTIP(J) + (WHIRL+OMR2)/ROM(I,J)*CPHI(I,J))
   120 PCAR9(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 = PHOIP(1)*AR*TIPT(1)*(TPP/TIPT(1))**(GAM*EXPO)
   DO 130 J=2,MHTP1
   DTIP = TIPT(J)-TIPT(J-1)
   DLAM = LAMBD(A(J)-LAMBD(A(J-1)
   TPPN = TIPT(J)-TWLMR(J)/2./CP
   IF (TPPN.LT.0.) GO TO 300
   PRELN = PHOIP(J)*AR*TIPT(J)*(TPPN/TIPT(J))**(GAM*EXPO)
   DTPP = TPPN-TTP
   DPREL = PRELN-PREL
   EVEL(J-1) = CP*DTIP+OMEGA*DLAM+CP*DTPP+AR/(PRELN+PREL)*(TPPN+TTP)
   1*DPREL
   FVEL(J-1) = DTPP/(TPPN+TTP)-AR/CP*DPREL/(PREF+PREL)
   TPP = TPP
130 PRL = 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**EXPO
   IF (I.GE.ILE(1).AND.I.LE.ILE(1)) GO TO 160
C--PVA OUTSIDE OF THE BLADE
WHIRL = LAMBD(A(1))
IF(I.GT.ITE(1)) WHIRL = LAMBO(1)
SBIETA = (WHIRL/R0M(I,1)-OMEGA*ROM(I,1))/WHIRL
IF(ABS(SBIETA).GT.1.) GO TO 21C
BETA(I,1) = ARSIN(SBIETA)
CBETA = COS(BETA(I,1))
RVA = RHO(I,1)*WHUB*CBETA*RCARB(1)
GO TO 170

C--PVA INSIDE OF THE BLADE
16C WLSRF = WHUB+DFDM(I,1)/2.
WSQ = WLSRF**2
T-tip = 1.-(WSQ+TWLMR(1))/CPTIP(1)
IF(T-tip.LT.0.) T-tip=0.
RHOL = RHOIP(1)*T-tip**EXPON
WTSRF = WHUB-DFDM(I,1)/2.
WSQ = WTSRF**2
T-tip = 1.-(WSQ+TWLMR(1))/CPTIP(1)
IF(T-tip.LT.0.) T-tip=0.
RHOT = RHOIP(1)*T-tip**EXPON
RHOWAV = (RHOL*WLSRF+4.*RHO(I,1)*WHUB*RHOT*WTSRF)/6.
CBETA = COS(BETA(I,1))
RVA = RHOWAV*CBETA*RCARB(1)

C--CONTINUE VELOCITY GRADIENT SOLUTION UP VERTICAL MESH LINE FROM HUB
C--TO SHROUD

C 17C DO 200 J=1,MHT
DELT(A = TOM(I,J+1)-TOM(I,J)
WAS = W(I,J)+(ATVEL(J)*W(I,J)+BTVEL(J)+CTVEL(J))/W(I,J)+CBETA*
1DTVRL(J))**2+ETVEL(J)*W(I,J)).DELTA+CTVEL(J))**2+DOVEL(J)**2+W(I,J)
C--CALCULATE RVA'S AT POSITION J+1 ON VERTICAL MESH LINE
IF(I.GE.ITE(J+1) .AND. I.LE.ITE(J+I)) GO TO 180

C--RVA'S OUTSIDE OF THE BLADE
WHIRL = LAMBD(A(J+1))
WHIRL = LAMBO(J+1)
WHET(A = (WHIRL/R0M(I,J+1)-OMEGA*ROM(I,J+1))
SBETA = WTHETA/WAS
IF(ABS(SBETA).GT.1.) GO TO 21C
BETA(I,J+1) = ARSIN(SBETA)
CBETA = COS(BETA(I,J+1))
WASS = W(I,J+1)+(ATVEL(J+1)*W(I,J)+BTVEL(J)+CTVEL(J+1))/W(I,J+1)+CBETA*
1DTVRL(J+1))**2+ETVEL(J+1)*W(I,J+1)+CTVEL(J+1))**2+W(I,J+1)
WSQ = W(I,J+1)**2
T-tip = 1.-(WSQ+TWLMR(J+1))/CPTIP(J+1)
IF(T-tip.LT.0.) T-tip=0.
RHOL = RHOIP(J+1)*T-tip**EXPON
IF(I.GE.ITE(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))
RVA = RHO(I,J+1)*W(I,J+1)*CBETA*RCARB(J+1)
GO TO 195

C--RVA'S INSIDE OF THE BLADE
190 WLSRF = W(I,J+1)+DFDM(I,J+1)/2.
WSQ = WLSRF**2
T-tip = 1.-(WSQ+TWLMR(J+1))/CPTIP(J+1)

174
IF (TTIP.LT.0.) TTIP=0.
RHOL = RHOIP(J+1)*TTIP**EXPON
WSRF = W(I,J+1)-DFDM(I,J+1)/2.
WSQ = WSRF**2
TTIP = 1.-(WSQ+TWLMP(J+1))/CPTIP(J+1)
IF (TTIP.LT.0.) TTIP=0.
RHOT = RHOIP(J+1)*TTIP**EXPON
RHOWAV = (RHOL+WSRF+4.*RHO(I,J+1)*W(I,J+1)+RHOT*WSRF)/6.
CBETA = COS(BETA(I,J+1))
RVAS = RHOWAV*CBETA*RCAP(J+1)

C--INCREMENT THE MASSFLOW
195 UNEW(J+1) = (RVA+RVAS)*DELTA/2.+UNEW(J)
200 RVA = RVAS

C--STORE MAX AND MIN VALUES FOR WHUB AND INTEGRATED MASSFLOW

MAXFLO = AMAX1(UNEW(MHTP1),MAXFLO)
MINFLO = AMIN1(UNEW(MHTP1),MINFLO)
WMAX = AMAX1(WHUB,WMAX)
WMIN = AMIN1(WHUB,WMIN)

C--CHECK CONTINUITY AND ESTIMATE NEW VALUE FOR W AT THE HUB

IF (IND.LT.10) GO TO 230
C
C--END OF INNER ITERATION PROCEDURE
C
C--IND=10 INDICATES CHOKED FLOW
IF (IND.EQ.10) GO TO 250
C
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 LAMDO
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) = RVTH TA(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
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.3) 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
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)
176
STOP
C--FORMAT STATEMENTS
C
1000 FORMAT (1HL,10X,68HMSPL EXCEEDS CHOKING MASS FLOW FOR VERTICAL OR
1HOGONAL MESH LINE I =,I3/12X,19HCOKING MASS FLOW =,G15.6)
1010 FORMAT (1HL,10X,85HVELOCITY GRADIENT SOLUTION CANNOT BE OBTAINED
1 FOR VERTICAL ORTHOGONAL MESH LINE I =,I3/12X,56HANY SUBSEQUENT OUT
2PUT 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 =,F9.5,22H
1MINIMUM CHOKING MASSFLOW =,G13.5/12X,26HMINIMUM CHOKING MASSFLOW
2=G13.5)
1040 FORMAT (1HL,10X,52X,25(1H*)/42X,
15(IH*)/42X,19HTANGENTIAL SOLUTION,12X,1H*/42X,45H* BY VE
2LOCITY GRADIENT APPROXIMATE METHOD */42X,45(H*)///)
1050 FORMAT (10X,51HCONTIN
1COULD 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,8HMAGNITUDE LESS THAN TANGENTI
2AL VELOCITY, OR AFTER ADJUSTMENT OF STAGNATION TEMPERATURE,12X,29
3STAGNATION DENSITY, OR WHIRL./10X,84HRESTART OF ITERATION PROCEDURE
4RE (LOOP TO STATEMENT 9C) WAS ABORTED AFTER 100 OR MORE/12X,41HTO
5ITAL ITERATIONS (LOOP TO STATEMENT 15C).)
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,15,6H TIM
1ES/12X,16HWHUB WAS INCREASED,14,6H TIMES/12X,16HWHUB WAS DECREASED
2,14,6H TIMES/12X,53HBOUNDARY VALUES (TIPBDY, RHOIP, LAMBDA) WERE A
3JUSTED,14,6H TIMES)
1090 FORMAT (/10X,120(IH*))
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,NWRT4,NWRT5,NWRT6
COMMON/INPUTT/GAM,AR,MSFL,OMEGA,REDFAC,VELTOL,FNEW,DNEW,MBI,MBO,
MM,MHT,NBL,NHUB,NTP,NIN,NOUT,NBLPL,NPPP,NOSTAT,NSL,NLOSS,
LSFR,LTPR,LAMVT,LROT,LBLAD,LETAN,ANGROT,IMESH,ISLINE,
ISTATL,FLOT,ISUPER,ITSON,IDEBUG,ZCIN,ZOMBI,ZOMBO,ZOMOUT,
ROMIN,ROMBI,ROMBO,ROMOUT,ZHIN,ZTIN,ZHOUT,ZTOUT,RHIN,RBIN,RHOUT,
RTOUT,TITLEI(20),ZHUB(50),RHUB(50),ZTIP(50),RTIP(50),SPIN(50),
RADIN(50),TIP(50),PRIP(50),LAMIN(50),VTHIN(50),SFOUT(50),
RADOUT(50),PROP(50),LOSOOUT(50),LAMOUT(50),VTHOUT(50),
BETALE(50),BETATE(50),ZHST(50),ZTST(50),RHST(50),RTST(50),
PLFR(50),PERCRD(50),PERLOS(50),ZBL(50,50),RBL(50,50),
ZBL(50,50),TNBL(50,50),TTBL(50,50),TH1BL(50,50),TH2BL(50,50)

177
COMMON/CALCCN/MMM1,MTTP1,CP,EXPN,TGROG,PITCH,RLEH,RLET,RTET, 
1 ZLE(50),PLE(50),ZTE(50),ZTRO(50),ZLEOM(101),RLEOM(101), 
2 SLEOM(101),THLEOM(101),ZTOM(101),RTEOM(101),STEOM(101), 
3 THTEOM(101),ITOE(101),ZTE(101),ZOM(101,101),ROM(101,101), 
4 SOM(100,101),TOM(100,101),RTH(100,101),DTDHS(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),WSUBA(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),WCR(100,101),CURV(100,101), 
4 WSURF(100,101),WTHER(100,101),CAMP(100,101),SAMP(100,101), 
5 RHOAV(100,101),DELPHO(100,101),FTH(100,101),FDM(100,101), 
6 XION(100,101),ZETOM(100,101),DTHDZ(100,101), 
COMMON/ROTATN/ZHROT(50),RHROT(50),ZTROT(50),RTROT(50), 
1 ZLEOMR(101),RLEOMR(101),ZTEOMR(101),RTEOMR(101), 
2 ZBLEOT(50,50),RBLROT(50,50),ZOMR(100,101),ROMR(100,101), 
COMMON/INDCOM/NCPC,NPPC,ZPC(51,51),RPC(51,51),TPPC(51,51), 
1 THPC(51,51),TDTHZ(51,51),TDTHR(51,51),BTHLE(101),BTHTE(101), 
1 BTBFLE(101),BTBFTE(101), 
DIMENSION BLINC(101),UBINC(101),BTBLLE(101),BLDEV(101), 
1 BTBLTE(101), 
REAL LAMDAF 
THE BETA FLOW ANGLFES INTO THE LEADING EDGE 
C--LINDV CORRECTS THE BETA FLOW ANGLFES INTO THE LEADING EDGE 
II = 1 
JJ = 1 
J = JARG 
ICOUNT = ICOUNT+1 
DEGRAD = 180. / 3.1415927 
C--CALCULATE BLADE MEAN CAMEBR ANGLE AT LEADING EDGE 
I = ILE(J)-1 
ALPHLE = ALPHA(I,J)+ (SLEOM(J)-SOM(I,J)) *(ALPHA(I+1,J)-ALPHA(I,J))/ 
(SOM(I+1,J)-SOM(I,J)) 
CALL LININT(ZPC,RPC,VTZDH,NPPC,NBLPC,51,51,ZLEOMR(J),RLEOMR(J), 
1 DTDZLE,II,JJ) 
CALL LININT(ZPC,RPC,VDTHZ,NPPC,NBLPC,51,51,ZLEOMR(J),RLEOMR(J), 
1 DTDZLE,II,JJ) 
TANBBL = RLEOM(J) - (DTDRLE_SIN(ALPHLE) +DTDZLE_COS(ALPHLE)) 
BTBLLE(J) = ATAN(TANBBL)'DEC.PAD 
C--CALCULATE BETA 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)) 
RHOS = 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)) 
TMLR = 2.*OMEGA*LAMDAF(ULE,ILE(J),J) - (CMFGAR*LLEOM(J)) **2 
TPRIM = TIPP(ULE) 
CPTIP = 2.*CP_TIPRIM 
RHOIP = RHOIPF(ULE) 
CONST1 = TAN(BETAFS)/RHOFS_BTHLE(J) / PITCH 
CONST2 = (RHOS*PITCH*WFS/BTHLE(J)) **2 / (1. + (TAN(BETAFS))**2) 
RHOFN = RHOS 
TIP = TIPRIM-TMLR/2./CP 
WTHETA = WFS*SIN(BETAFS) 
WMSON = SQRT(2.*GAM*AR*TIP/((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 = RHOBFP*TBFTIP**EXPON
   IF (ABS(RHOBFN-RHOBF)/RHOBFN.GT..001) GO TO 10
   GO TO 18
14 RHOBF = RHOBFN
   TBFTIP = (RHOBF/RHOBF)**(1./EXPON)
   WSQBF = (1.-TBPTIP)*CPTIF-TWLMR
   RHSQBF = CONST2/WSQBF**2+(1+TANBBF**2)
   IF(RHSQBF.LT.0.) GO TO 16
   RHOBFN = SQRT(RHSQBF)
   IF(ABS(RHOBFN-RHOBF)/RHOBFN.GT..001) GO TO 14
   WRITE(NW1T,1360) J
   GO TO 18
16 RHOBFN = RHOS
   WRITE(NW1T,1070) J
18 TANBBF = CONST1/RHOBFN
   BETAIF = ATAN(TANBBF)
C--CALCULATE DISTANCE FOR BETA CORRECTION
   BLDCRD = (RLEOM(J)+RTEOM(J))/2.*((THTEOM(J)-THLEOM(J))
   BLDCRD = SQRT(BLDCRD**2+(STPOM(J)-SLEOM(J))**2)
   SLIDLE = BLDCRD/PITCH/RLEOM(J)
   DISTLE = AMIN1(.5,AMAX1(1./6.,(11.-4.*SLIDLE)/18.))*(STFOM(J)-SLEOM(J))
C--CORRECT BETA FOR INCIDENCE NEAR THE LEADING EDGE,
C--USING LINEAR CORRECTION FOR ANGLE
   I = ILE(J)
   SDIST = SLEOM(J)*DISTLE-SOM(I,J)
   IF (SDIST.LE.0.) GO TO 30
   BETA(I,J) = BETA(I,J)+(BETAFS-BTBFE(J))*SDIST/DISTLE
   I = I+1
   GO TO 20
C--CALCULATE INCIDENCE ANGLES
30 BINC(J) = BETAFS*DEGRAD-BTBLLE(J)
   UBINC(J) = BETAFS*DEGRAD-BTBLLE(J)
   IF (ICOUNT.EQ.MHTPI) LINC=I
   RETURN
C--TINDV CORRECTS THE BETA FLOW ANGLES INTO THE TRAILING EDGE
C
ENTRY TINDV(IARG,JARG,ICOUNT)
   J = JARG
   ICOUNT = ICOUNT+1
C--CALCULATE BLADE MEAN CAMBER ANGLE AT TRAILING EDGE
   I = ITE(J)+1
   ALPHTE = ALPHA(I,J)+(STFOM(J)-SOM(I,J))*(ALPHA(I,J)-ALPHA(I-1,J))/(-1*(SOM(I,J)-SCM(I-1,J))
   CALL LININT(ZPC,RPC,DTHDZ,NPPC,NBLPC,51,51,ZTEOM(J),RTEOM(J),-1DTDZTE,II,J)
   CALL LININT(ZPC,RPC,DTHDR,NPPC,NBLPC,51,51,ZTEOMR(J),RTEOMR(J),-1DTDZTE,II,J)
   TANBBL = RTEOM(J)*(DTDZTE*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))
RHOPS = 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))
PLOSE = PLOSS(I,J) + EXFRAC * (PLOSS(I,J) - PLOSS(I+1,J))
TMLMR = 2. * OMEGA * LAMDAF * (UTE, ITE(J), J) - (OMEGA * RTEOM(J)) ** 2
TIPIM = TIPF (UTE)
CPTIP = 2. * CP * TIPIM
RHOIP = RHOIPF(UTE) * (1. - PLOSE)
CONST1 = TAN(BETAFS) / RHOFS * BTHTE(J) / PITCH
CONST2 = (RHOPS * PITCH * WFS / BTHTE(J)) ** 2 / (1. + (TAN(BETAFS)) ** 2)
RHOBFN = RHOPS
TIPFP = TIPF * TMLMR / 2. / CP
WTHET = WFS * SIN(BETAFS)
WMSON = SQRT(2. * GAM * AB * TPP / (GAM + 1.)) - (GAM - 1.) / (GAM + 1.) * WTHET ** 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 / TMLMR) / CPTIP
IF (TBFTIP .LT. 0.) GO TO 46
RHOBFN = RHOIP * TBFTIP ** EXPON
IF (ABS(RHOBFN - RHOBF) / RHOBFN .GT. .001) GO TO 40
GO TO 48

44 RHOBF = RHOBFN
TBFTIP = (RHOBF / RHOIP) ** (1./ EXPON)
WSQBF = (1. - TBFTIP) * CPTIP / TMLMR
RHSQBF = CONST2 / (WSQBF - CONST2 * CONST1 ** 2)
IF (RHSQBF .LT. 0.) GO TO 46
RHOBFN = SQRT(RHSQBF)
IF (ABS(RHOBFN - RHOBF) / RHOBFN .GT. .001) GO TO 44
WRITE (NWRIT, 1080) J
GO TO 48

46 RHOBFN = RHOPS
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 = AMINI(.5, AMAXI(I/6., (1. - .5 * SLIDTE) / 18.)) * (STEOM(J) - SLEOM(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 - BTBFTEP(J)) * SDIST / DISTTE
I = I - 1
GO TO 50

C
C--CALCULATE DEVIATION ANGLES

180
60 BLDEV(J) = BETAFF*DEGRAD-BTBLE(J)
UBDEV(J) = BETAFS*DEGRAD-BTBLE(J)
IARG = I+1
RETURN

C--PINDV PRINTS THE INCIDENCE AND DEVIATION ANGLES
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), BTLLE(J), BLDEV(J),
UBDEV(J), BTLTE(J), J=1,MHTP1)
WRITE(NWRT6,1050)
RETURN

C--FORMAT STATEMENTS

1000 FORMAT (II/44X,40(1H*),/44X,40H*** INCIDENCE AND DEVIATION ANGLES,9X,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 CHOIKING 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 CHOIKING MASSFLOW SOLUTION */35X,59(1H*)
1030 FORMAT (/24X,10H* MESH *,9X,9HINCIDENCE,7X,11HBLADE ANGLE,2H *,
18X,9HDEVIATION,7X,11HBLADE ANGLE,2H */24X,10H* LINE *,3X,
27HUNBLOCKED,3X,9HUNBLOCKED,4X,7HAT L.E.,3X,1H*,3X,7HUNBLOCKED,3X,
39HUNBLOCKED,4X,7HAT T.E.,3X,1H*)
1040 FORMAT ((24X,1H*,2X,13X,3X,10F9.2,2X,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--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,MSFI,OMEGA,REDPAK,VELTOL,FXNEW,DNEW,MBL,MBO,
1 MM,MHT,MBL,NNUB,NTP,NNOUT,MBLPL,NP,TOSTAT,NSL,NLOSS,
2 LSFR,LTPL,LANVT,LBOT,LBLAD,LETAN,ANGROT,IMESH,ISLINE,
ISTATL, I_PLOT, ISUPER, ITSON, TDNEUG, ZOMIN, ZOMBI, ZOMBO, ZOMOUT, 
ROMIN, ROMB, ROMOUT, ZHN, ZITIN, ZOUT, RHI, RTIN, RHOUT, 
RTOUT, TIT(2), ZHUB (50), PHUB (50), ZTIP (50), RTIP (50), SPIN (50), 
RADIN (50), TIP (50), PRIP (50), LAMIN (50), VTHIN (50), SPOUT (50), 
RADOUT (50), FRP (50), LCOUT (50), LAMOUT (50), WTHOUT (50), 
DETAE (50), BETATE (50), ZHST (50), ZTST (50), FHST (50), RTST (50), 
FLER (50), PERCPD (50), PERLOS (50), BL5 (50, 50), RBL (50, 50), 
THBL (50, 50), TNBL (50, 50), TBL (50, 50), TH2BL (50, 50), 
COMMON/CALCON/MMM1, MHTP1, CP, EPON, TMOG, PITCH, RLEH, RLET, XTE, 
M ZLE (50), RLE (50), ZTV (50), RTE (50), ZLFOM (10), RLFO (10), 
2 SLEOM (10), ZHEOM (10), ZTEOM (10), RTEOM (10), STEOM (10), 
3 TTEOM (10), ILE (10), ITE (10), ZOM (100, 101), ROM (100, 101), 
4 SOM (100, 101), TON (100, 101), ATH (100, 10), DTBDS (10, 10), 
5 DTHD (10, 10), PLOSS (10, 10), CPHE (100, 101), SPM (100, 101), 
COMMON/VARCOM/A (100, 100, 100), DNO (100, 100), RHO (100, 100), 
1 WSUBS (100, 100), WSUBT (100, 101), WSUBZ (100, 101), WSUBR (100, 101), 
2 WSUBM (100, 101), VTH (100, 101), WTH (100, 101), W (100, 101), 
3 ALPHA (100, 100), ROTA (100, 101), WCR (100, 101), CURV (100, 101), 
4 WLSURF (100, 101), WTSURF (100, 101), CAMP (100, 101), SAMP (100, 101), 
5 RHOAV (100, 100), DELRHO (100, 101), PT (100, 101), DFDM (100, 101), 
DIMENSION SLOPE (50), EM (50), AAA (50), BBB (50), RILOM (101), UILOM (101), 
REAL LAMDAF, LAMIN 
KK = 2 
IF (ABS (SF - SPIN (1)) .GT. TOLER) GO TO 10 
LAMDAF = LAMIN (1) 
IF (I.LT. ILE (J)) DLDU (I, J) = SLOPE (1) 
RETURN 
10 IF (SF - SPIN (1)) 20, 20, 30 
20 LAMDAF = LAMIN (1) + (SF - SPIN (1)) * SLOPE (1) 
IF (I.LT. ILE (J)) DLDU (I, J) = SLOPE (1) 
RETURN 
30 IF (ABS (SF - SPIN (KK)) .GT. TOLER) GO TO 40 
LAMDAF = LAMIN (KK) 
IF (I.LT. ILE (J)) DLDU (I, J) = SLOPE (KK) 
RETURN 
40 IF (SF = SPIN (KK)) 70, 70, 50 
50 KK = KK + 1 
IF (KK = NIN) 30, 30, 60 
LAMDAF = LAMIN (NIN) + (SF - SPIN (NIN)) * SLOPE (NIN) 
IF (I.LT. ILE (J)) DLDU (I, J) = SLOPE (NIN) 
RETURN 
70 SK = SPIN (KK) - SPIN (KK - 1) 
LAMDAF = EM (KK - 1) * (SPIN (KK) - SP) ** 3 / 6. / EM (KK) - EM (KK - 1) * (SPIN (KK) - SP) ** 3 / 6. 
1 IF (I.LT. ILE (J)) DLDU (I, J) = EM (KK - 1) * (SPIN (KK) - SP) ** 3 / 6. 
2 IF (I.LT. ILE (J)) DLDU (I, J) = EM (KK) * (SPIN (KK - 1) - SP) ** 3 / 6. 
RETURN 
ENTRY LAMNI (NIN) 
IF (ITER.EQ.0) GO TO 100 
IF (LSFR.EQ.0.AND.LAMVT.EQ.0) GO TO 100 
I1 = MBI 
JJ = 1
\[ CAN = \cos(\text{ANGROT}) \]
\[ SAN = \sin(\text{ANGROT}) \]
\[ ZHINRO = ZHIN \times CAN + RHIN \times SAN \]
\[ RHINRO = RHIN \times CAN - ZHIN \times SAN \]
\[ ZTINRO = ZTIN \times CAN + NTIN \times SAN \]
\[ RTINRO = RTIN \times CAN - ZTIN \times SAN \]

DO 80 KK=1,MHTP1
DIST = FLOAT(KK-1)/FLOAT(MHT)
RILCM(KK) = RHIN*DIST*(RTINRO-RHINRO)
ZIPOT = ZHINRO*DIST*(ZHINRO-ZHINRO)
Rilot = RHINRO*DIST*(RTINRO-RHINRO)

80 CALL LINTNT(ZHIMROT,RHIMROT,OMROT,MM,MMT,100,1^1,ZHIMRT,RTIMRT,
1UILOM(KK),IT,JJ)
IF (LSFR.EQ.0) CALL SPLINT(UILOM,RILCM,MMT,SPIN,NIN,PADIN,AAA,
1BBB)
IF (LSFR.EQ.1) CALL SPLINT(RIMT,UILOM,MMT,SPIN,NIN,PADIN,AAA,
1BBB)
IF (LSFR.EQ.1.OR.LAMVT.EQ.0) GO TO 100
DO 90 KK=1,NIN
LAMIN(KK) = RADIN(KK)*VTHIN(KK)
90 CALL SPLINE(SPIN,IAMIN,NIN,OPT,F,TOLER)
RETURN
END

FUNCTION RVTHTA(SP,I,J)
C- RVTHTA CALCULATES R * V-THETA AS A FUNCTION OF STREAM FUNCTION
C- DOWNSTREAM OF THE BLADE

COMMON NREAD,NWRIT,ITEM,IFEND,MMT1,MMT,RHIMRO,MMT5,NMT6
COMMON/GAM,AR,MSFL,OMFGA,REDPAC,VELTOL,FINW,DNF,MSI,MRO,
1 MM,RMT,NHT,NHUB,KTP,NIN,NOTH,NBLPL,NPPP,NOSTAT,NSL,WLOSS,
2 LSPF,LTPL,LAMVT,LRIET,LETEAN,ANGROT,TMESH,ISLINE,
3 ISTAT,IPLT,LSTU,ITSON,IDEBUG,ZOMIN,ZOMBI,ZOOMO,ZOMOUT,
4 VATOM,ROMBI,ROMO,PROMO,ROMU,ZHIN,ZTIN,ZHOUT,ZTOUT,RHIN,RTIN,
5 RTOUT,THLET(20),ZHN(50),ZTIN(50),ZTIP(50),SIP(50),SPIN(50),
6 RADIN(50),TIP(50),PPTP(50),LAMIN(50),VTHIN(50),SPIN(50),
7 RADOUT(50),PROP(50),INDOUT(50),LAMOUT(50),VTHOUT(50),
8 BETAE(50),BETAT(50),ZHT(50),ZTT(50),RHST(50),RTST(50),
9 PFFR(50),PERCRD(50),PERLOS(50),ZBL(50,50),
1 THBL(50,50),TBL(50,50),TTBL(50,50),TH2BL(50,50),
COMMON/CALCN/MMM1,MHTP1,CP,EXPON,TGRG,PITCH,RLER,RLRT,RTET,TET,
1 ZLE(50),RLE(50),ZTE(50),BTE(50),ZLEM(101),RLEM(101),
2 SLFR(101),THLEOM(101),ZTHEOM(101),BTHEOM(101),STHEOM(101),
3 THEOM(101),ILE(101),ITE(101),ZOM(101,101),ROM(101,101),
4 SOM(100,101),TOM(100,101),BTH(100,101),DTHDS(100,101),
5 DTHD(100,101),PLOSS(100,101),CPHI(100,101),SPHI(100,101),
COMMON/VARCOM/A(4,100,101),UOM(100,101),K(100,101),RHQ(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),WCCR(100,101),CURV(100,101),
4 WLSURF(100,101),WSURF(100,101),CAMP(100,101),SAMP(100,101),
5 RHOAV(100,101),DELAV(100,101),FT(100,101),DFDM(100,101),
6 XION(100,101),ZETOM(100,101),DLDU(100,101),
COMMON/ROTATH/ZHROT(50),RHROT(50),ZTROT(50),RROT(50),
1 ZLEOMR (101), RLEOMR (101), ZTOOMR (101), RTEOMR (101),
2 ZBLROT (50, 50), RBLROT (50, 50), ZOMROT (100, 101), RROBOT (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)) EQ 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)) EQ 70, 70, 50
50 KK = KK + 1
IF (KK - NOUT) EQ 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) * (SFOUT(KK - 1))
1 ** 3/6. / SK * (LAMOUT(KK) / SK * EM(KK) * SK ** 3/6. ) * (SFOUT(KK - 1)) +
2 (LAMOUT(KK - 1) / SK * EM(KK - 1) / SK ** 3/6. ) * (SFOUT(KK - 1) - 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)) ** 3/6. / SK.
RETURN
ENTRY RVNIT(NNN)
IF (ITER.EQ.4) GO TO 100
IF (LSFR.EQ.0 .AND. LAMVT.EQ.C) GO TO 100
II = MBO
JJ = 1
CAN = COS(ANGROT)
SAN = SIN(ANGROT)
ZHOP = ZHOUT * CAN + RHOUT * SAN
RHOR = RHOUT * CAN - ZHOUT * SAN
ZTOR = ZTOUT * CAN + RTOUT * SAN
RTOR = RTOUT * CAN - ZTOUT * SAN
DO 80 KK = 1, MHTP1
DIST = FLOAT(KK - 1) / FLOAT(MHT)
ROLOM(KK) = RHOUT * DIST * (RTOUT - RHOUT)
ZOROT = ZHOR + DIST * (ZTOR - ZHOR)
OROT = RHOR + DIST * (RTOR - RHOR)
80 CALL LININT(ZOMROT, ROMROT, UOM, MM, MHTP1, 100, 101, ZOROT, ROBOT,
1 UOMOM(KK), II, JJ)
IF (LSFR.EQ.3) CALL SPLINT(UOMOM, ROMOM, MHTP1, SFOUT, NOUT, RADOUT,
1 AAA, BBB)
IF (LSFR.EQ.1) CALL SPLINT(ROMOM, UOMOM, MHTP1, RADOUT, NOUT, SFOUT,
1 AAA, BBB)
IF (LSFR.EQ.1 .OR. LAMVT.EQ.7) 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(SP)

C
C--TIPF CALCULATES UPSTREAM ABSOLUTE TOTAL TEMPERATURE
C--AS A FUNCTION OF STREAM FUNCTION

COMMON/INPUT/GAM, AR, SFL, OMEGA, REDFAC, VELTOL, FNEW, DNEW, MBI, MBO,
1 MM, MHT, NLB, HUB, NTEMP, NI, OUT, NLPL, NEPP, NOSTAT, NSL, NLOSS,
2 NLPP, NLPL, LAMT, LROT, LBIA, LETAN, ANGROT, IMSH, ISLINE,
3 ISTAT, IELOL, ITSP, ITSON, ITEURG, ZOMIN, ZOMBI, ZOMBO, ZOUT,
4 RZOMIN, RZOMBI, RZOMBO, RZOUT, ZHEIN, ZFIN, ZFIN, ZOUT,
5 RZOUT, ITLTFL (2C), ZHUB (5C), RHUB (5C), ZTIP (5C), RTIP (5C), SPIN (5C),
6 RADIN (5C), TIP (5C), PRIP (5C), LAMIN (5C), VTHIN (5C), SOUT (5C),
7 RADOUT (5C), PROP (5C), LOSOUT (5C), LAMOUT (5C), VTHOUT (5C),
8 BETALE (5C), BETAPF (5C), ZSTH (5C), ZSTT (5C), RHST (5C), RTST (5C),
9 FPLR (5C), PERCRD (5C), PERLOS (5C), ZBL (5C), ZRBL (5C),
10 THBL (5C), INBL (5C), TBL (5C), TBL (5C), TBL (5C),
COMMON/CALCON/MM1, MHTP1, CP, EXPON, TBGRG, PITCH, RLEH, RLET, RTEH, RTET,
1 ZLE (N1), ZLE (N1), ZTE (N1), ZTE (N1), RTBCH (101), RTBCH (101),
2 TEBOM (101), TEBOM (101), TEBOM (101), TEBOM (101),
3 TEBOM (101), TEBOM (101), TEBOM (101), TEBOM (101),
4 TEBOM (101), TEBOM (101), TEBOM (101), TEBOM (101),
5 TEBOM (101), TEBOM (101), TEBOM (101), TEBOM (101),
DIMENSION SLOPE (5C), EM (5C)
K = 2
IF (ABS(SP - SFIN(1)) .GT. TOLER) GO TO 10
TIPF = TIP(1)
RETURN
10 TIPF = TIP(1) + (SP - SFIN(1)) * SLOPE(1)
RETURN
30 IF (ABS(SP - SFIN(K)) .GT. TOLER) GO TO 40
TIPF = TIP(K)
RETURN
40 IF (SP - SFIN(K)) 7C, 7C, 50
50 K = K + 1
IF (K .NE. NIN) 30, 3C, 60
60 TIPF = TIP(NIN) + (SP - SFIN(NIN)) * SLOPE(NIN)
RETURN
70 SK = SFIN(K) - SFIN(K-1)
TIPF = EM(K-1) * (SFIN(K) - SP) * 3/6. / SK + EM(K) * (SFIN(K) - SP) * 3/6.*
1 6. / SK + (TIP(K) / SK - EM(K) / SK) * (SP - SFIN(K-1)) * (TIP(K-1))
2 SK = EM(K-1) * SK / 6. + (SP - SFIN(K-1)) * (TIP(K-1))
RETURN
ENTRY TIPF(NIN)
CALL SPLINE(SFIN, TIP, NIN, SLOPE, EM)
TOLER = ABS(SFIN(NIN) - SFIN(1)) / FLOAT(NIN) * 1. E-6
RETURN
END

FUNCTION RHOIPF(SP)

C
C--RHOIPF CALCULATES UPSTREAM ABSOLUTE TOTAL DENSITY
C--AS A FUNCTION OF STREAM FUNCTION

COMMON/INPUT/GAM, AR, SFL, OMEGA, REDFAC, VELTOL, FNEW, DNEW, MBI, MBO,
1 MM, MHT, NLB, HUB, NTEMP, NI, OUT, NLPL, NEPP, NOSTAT, NSL, NLOSS,
SURROGUTE CONTIN(XEST,YCALC,IND,JZ,YGIV,XDEL)
C--CONTIN CALCULATES AN ESTIMATE OF THE RELATIVE FLOW VELOCITY
C--FOR USE IN THE VELOCITY GRADIENT EQUATION
C
DIMENSION X(3),Y(7)
NCALL = NCALL+1
IF (IND.NE.1.AND.NCALL.GT.160) GO TO 160
GO TO (10,30,40,50,60,70,110,150),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+.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 139
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

187
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
188
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,Y1,1,SI,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*S2*(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,4HEINRSCT HAS FAILED TO CONVERGE IN 20 ITERATIONS/
110X,11HTOLERANCE =,G14.6/10X,4DHDISTANCE 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,BELOW
C--FIND I,J SUCH THAT (X0,Y0) IS IN COLUMN I FROM THE LEFT AND IN ROW J
C--FROM THE BOTTOM
IF (J.LT.2.0.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)
ABOVE = -1
RIGHT = -1
IF(Y(i,j) \geq Y(i,j) + (X(i+1,j) - X(i,j)) * (Y(i+1,j) - Y(i,j)))
ABOVE = ABOVE + 1
IF(Y(i+1,j+1) \geq Y(i,j+1))
RIGHT = RIGHT + 1

\text{IF}(Y(i,j) > Y(i,j) + (X(i+1,j) - X(i,j)) * (Y(i+1,j) - Y(i,j)))
ABOVE = ABOVE + 1
\text{IF}(X(i,j) \geq X(i+1,j))
RIGHT = RIGHT + 1

IN = I*RIGHT
JN = J*ABOVE
IF(IN \lt 1 OR IN \geq NX) RIGHT = 0
IF(JN \lt 1 OR JN \geq NY) ABOVE = 0
IF(ABOVE**2+RIGHT**2 \lt 0.0) GO TO 27
I = I*RIGHT
J = J*ABOVE
ICOUNT = I*RIGHT
IF(ICOUNT \gt ICNTMX) GO TO 110
GO TO 10

2C IJEX = 1
C-- SET EXTRAP TO INDICATE EXTRAPOLATION
EXTRAP(1) = 0.
EXTRAP(2) = 0.
IF(IN.LT.1 OR IN.GE.NX) EXTRAP(2) = -1.
IF(IN.GE.NX) EXTRAP(2) = 1.
IF(JN.LT.1 OR JN.GE.NY) EXTRAP(1) = -1.
IF(JN.GE.NY) EXTRAP(1) = 1.

C--CALCULATE CONSTANTS TO CALCULATE PY
Y13 = Y(i,j) - Y(i,j+1)
X13 = X(i,j) - X(i,j+1)
Y23 = Y(i+1,j) - Y(i+1,j+1)
X23 = X(i+1,j) - X(i+1,j+1)
Y31 = Y0 - Y(i,j)
X31 = 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*X13+X13*Y23
QB = X13*Y02-Y13*X02+Y01*X13-Y01*X23
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) \leq 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 (FP LESS THAN 0.)
4C IF(F1.LT.0.01) CASE = CASE+1.
IF(F2.LT.0.01) 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 (PF 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-I.
70 IF(ABS(CASE-.5).LT..6) GO TO 110
PF = (1.-CASE)*F1+CASE*F2
GO TO 90
C--IF QUADRATIC EQUATION IS NEAR LINEAR, USE BINOMIAL EXPANSION FOR PF
80 ACB2 = QA/QB*QC/QB
IF(ABS(ACB2).LT.1.E-8) ACB2 = 0.
PF = -QC/QB*(1.+ACB2+2.*ACB2**2)
90 IF(IJEX.EQ.2) GO TO 10C
IJEX = IJEX+1
FY = PF
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)
Y12 = Y0-Y(I,J+1)
X12 = 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)
RETURN
C--PRINT ERROR MESSAGE IF THERE IS A PROBLEM IN OBTAINING A SOLUTION
110 Z0 = 0.
WRITE(NWRITE,10C) I,J
RETURN
1000 FORMAT(38H1LININT CANNOT FIND INTERPOLATED VALUE/4H I =,I6,4H J =,
116)
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
SUBROUTINE SPLINE (X,Y,N,SLOPE,EM)

C--SPLINE CALCULATES FIRST AND SECOND DERIVATIVES AT SPLINE POINTS
C--END CONDITION - SECOND DERIVATIVES AT END POINTS ARE
C--SDRI AND SDRN TIMES SECOND DERIVATIVES AT ADJACENT POINTS
C

10 COMMON NREAD,NWRIT
DIMENSION X(N),Y(N),SLOPE(N),EM(N)
DIMENSION G(101),SB(101)
IERR = 0
SDRI = .5
SDRN = .5
C = X(2)-X(1)
IF (C.EQ.0.) GO TO 50
SB(1) = -SDRI
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
P = (Y(I+1)-Y(I))/C-(Y(I)-Y(I-1))/A
10 G(I) = (6.*P-A*G(I-1))/W
20 EM(N) = SDRN*G(N-1)/(1.*SDRN*SB(N-1))
DO 30 I=2,N
K = N+I-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 (1/17X,18HNUMBER OF POINTS =,I4//17X,8X ARRAY,6X,8HY ARR
SUBROUTINE SPLINT (X,Y,N,Z,MAX,YINT,DYDX,D2YDX2)

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(10,1)
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) =-SDRI
G(I) = 0.
NO = N-I
IF (NO.LE.0) GO TO 140
IF (NO.EQ.1) GO TO 20
DO 10 I=2,NO
C = X(I+1)-X(I)
IF (C.EQ.0.) GO TO 130
IF (C.LT.0.) 130 IERR = I
W = 2.*(A+C)-A*SB(I-I)
SB (I) = C/W
10 G(I) = (6.*A-G(I-I))/W
20 EM(N) = SDRN*G(N-I)/(1.+SDRN*SB(N-1))
DO 30 I=2,N
K = N+I-I
30 EM(K) = G(K)-SB(K)*EM(K+1)
IF (MAX.LE.0) RETURN

ENTRY SPLINT (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.*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) = (X(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)/SK -EM(K-1)
   2 *SK/6.)*((X(K)-Z(I))**2/2.0/SK +EM(K)*((X(K-1)-Z(I))**2/2.
   1 /SK+Y(K)/(K-1)/SK -(EM(K)-EM(K-1))*SK/6.
   D2YDX2(I) = EM(K)-((X(K)-Z(I))/SK(EM(K)-EM(K-1))
   CONTINUE
   IF (IERR.EQ.0) RETURN

120 WRITE(NWIT,1020)
   WRITE(NWIT,1030)
   RETURN

END

SUBROUTINE SLOPES(X,Y,N,SLOPE)

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 I=2,NI
  X3X2 = X(I+1)-X(I)
  X2X1 = X(I)-X(I-1)
  X1X1 = X(I+1)-X(I-1)
  Y3Y2 = Y(I+1)-Y(I)
  Y2Y1 = Y(I)-Y(I-1)
1C  SLOPE(I) = (X2X1**2*Y2Y2+X3X2**2*Y2Y1)/(X3X2*X2X1*X3X1)
C--FIRST POINT
  X3X2 = X(3)-X(2)
  X2X1 = X(2)-X(1)
  X1X1 = X(3)-X(1)
  Y3Y2 = 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)
  X1X1 = X(N)-X(N2)
  Y3Y2 = Y(N)-Y(N1)
  Y2Y1 = 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 SPLISI(X,Y,N,YIP,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
  G(I) = (6.*F-A*G(I-1))/W
  10 W = C*SB(N-1)
  F = YNP-(Y(N)-Y(N-1))/C
RETURN
END
SUBROUTINE SPINSL(X, Y, N, Y1P, YNP, Z, MAX, YINT, DYDX, D2YDX)

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), D2YDX(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
10 CONTINUE
IF (IERR.EQ.0) RETURN
WRITE(NR7IT,1000)
WRITE(NWRIT,1020) N, (X(I), Y(I), I=1,N)
IF (IERR.EQ.0) STOP
WRITE(NR7IT,1030)
RETURN
60 WRITE(NWRIT,1010)
WRITE(NWR7IT,1020) N, (X(I), Y(I), I=1,N)
STOP
1000 FORMAT (1HI, 10X, 44HSPLN 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 POINTS ARE UNDEFINED.)
1010 FORMAT (1HI, 10X, 62HSPLN ERROR -- NUMBER OF SPLINE POINTS GIVEN IS LESS THAN TWO)
1020 FORMAT (//17X, 18HNUMBER OF POINTS =, 146//17X, 9HX ARRAY, 6X, 8HY ARR
             1AY/(17X, 2G13.5))
1030 FORMAT (1HI)
END

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,N
SLOPE(I) = (X(I) - X(I-1)) / 6.*(EM(I) + EM(I-1)) + (Y(I) - Y(I-1)) / 1
SLOPE(N) = YNP
IF (IERR.EQ.0) RETURN
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(NWR7IT,1020) N, (X(I), Y(I), I=1,N)
STOP
1000 FORMAT (1HI, 10X, 44HSPLN 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 POINTS ARE UNDEFINED.)
1010 FORMAT (1HI, 10X, 62HSPLN ERROR -- NUMBER OF SPLINE POINTS GIVEN IS LESS THAN TWO)
1020 FORMAT (//17X, 18HNUMBER OF POINTS =, 146//17X, 9HX ARRAY, 6X, 8HY ARR
             1AY/(17X, 2G13.5))
1030 FORMAT (1HI)
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.*P-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.3) 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) = YIP
YINT(I) = Y(1) + YIP*(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 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.C / 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(NWRT,1000)
WRITE(NWRT,1020) N, (X(I),Y(I),I=1,N)
IF (IERR.EQ.0) STOP
WRITE(NWRT,1030)
RETURN
140 WRITE(NWRT,1010)
WRITE(NWRT,1020) N, (X(I),Y(I),I=1,N)
STOP
1000 FORMAT (1H1,10X,44HSPENSL 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,62HSPENSL ERROR -- NUMBER OF SPLINE POINTS GIVEN I
IS LESS THAN TWO)
1020 FORMAT (/17X,18HNUMBER OF POINTS = ,I4//17X,8HX ARRAY,6X,8HY ARR
197
Lewis Research Center,
National Aeronautics and Space Administration,
Cleveland, Ohio, March 22, 1977,
505-04.
The stream-function equation was derived as equation (B17) of part I (ref. 6):

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

\[
+ \frac{r B \rho}{w W_s} \left[ \frac{W_\theta}{r} \frac{\partial (r V_\theta)}{\partial t} + \xi w^2 + \zeta + F_t \right] = 0 \quad (A1)
\]

where

\[
\xi = \frac{1}{2} \left( \frac{R}{c_p} \frac{\partial p}{\partial t} - \frac{1}{T''} \frac{\partial T''}{\partial t} \right) \quad (A2)
\]

\[
\zeta = \omega^2 r \cos \varphi - \frac{R T''}{p''} \frac{\partial p''}{\partial t} \quad (A3)
\]

\[
F_t = \frac{\partial \theta}{\partial t} \frac{1}{\rho} \frac{\partial \rho}{\partial \theta} \quad (A4)
\]

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

\[
\left[ \frac{2u_1}{h_1(h_1 + h_2)} + \frac{2u_2}{h_2(h_1 + h_2)} - \frac{2u_0}{h_1h_2} \right] + \left[ \frac{2u_3}{h_3(h_3 + h_4)} + \frac{2u_4}{h_4(h_3 + h_4)} - \frac{2u_0}{h_3h_4} \right] - \frac{u_4 - u_3}{h_3 + h_4} \left[ \frac{\sin \varphi_0}{r_0} + \frac{1}{B_0} \left( \frac{B_4 - B_3}{h_3 + h_4} \right) + \frac{1}{\rho_0} \left( \frac{\rho_4 - \rho_3}{h_3 + h_4} \right) - \frac{\partial \varphi}{\partial t} \right] 0  
- \frac{u_2 - u_1}{h_1 + h_2} \left[ \frac{\cos \varphi_0}{r_0} + \frac{1}{B_0} \left( \frac{B_2 - B_1}{h_1 + h_2} \right) + \frac{1}{\rho_0} \left( \frac{\rho_2 - \rho_1}{h_1 + h_2} \right) + \frac{\partial \varphi}{\partial s} \right] 0  
+ \frac{r_0B_0\rho_0}{w(W_0)} \left( \frac{\partial(rV_\theta)}{\partial t} \right) \left[ \frac{\partial(rV_\theta)}{\partial t} \right] + \xi_0w_0^2 + \xi_0 + (F_t) \right) = 0 \quad (A5)
\]

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 \( V_\theta \) from the previous iteration. The final result to be used in equation (A5) is

202
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

$$u_0 = \sum_{i=1}^{4} a_i u_i + k_0$$  \hspace{1cm} (A7)$$

where the coefficients are calculated as follows:

$$a_0 = \frac{2}{h_1 h_2} + \frac{2}{h_3 h_4}$$

$$c_1 = h_1 + h_2$$

$$c_2 = h_3 + h_4$$

$$\left( \frac{\partial \varphi}{\partial s} \right)_0 = \begin{cases} \frac{\sin \varphi_4 - \sin \varphi_3}{c_2 \cos \varphi_0} & \text{if } |\cos \varphi_0| \geq \frac{\sqrt{2}}{2} \\ \frac{\cos \varphi_3 - \cos \varphi_4}{c_2 \sin \varphi_0} & \text{if } |\cos \varphi_0| < \frac{\sqrt{2}}{2} \end{cases}$$  \hspace{1cm} (A8)$$

(continued)
\[
\frac{\partial \phi}{\partial t} = \begin{cases} 
\frac{\sin \varphi_2 - \sin \varphi_1}{c_1 \cos \varphi_0} & \text{if } |\cos \varphi_0| \geq \frac{\sqrt{2}}{2} \\
\frac{\cos \varphi_1 - \cos \varphi_2}{c_1 \sin \varphi_0} & \text{if } |\cos \varphi_0| < \frac{\sqrt{2}}{2}
\end{cases}
\]

\[
d_1 = \frac{B_2 - B_1 + \rho_2 - \rho_1}{B_0} + \frac{\cos \varphi_0}{\rho_0} + \frac{\partial \phi}{\partial s} + \left(\frac{\partial \phi}{\partial t}\right)_0
\]

\[
d_2 = \frac{B_4 - B_3 + \rho_4 - \rho_3}{B_0} + \frac{\sin \varphi_0}{\rho_0} - \left(\frac{\partial \phi}{\partial t}\right)_0
\]

\[
a_1 = \frac{\left(\frac{2}{h_1} + d_1\right)}{a_0 c_1}
\]

\[
a_2 = \frac{\left(\frac{2}{h_2} - d_1\right)}{a_0 c_1}
\]

\[
a_3 = \frac{\left(\frac{2}{h_3} + d_2\right)}{a_0 c_2}
\]

\[
a_4 = \frac{\left(\frac{2}{h_4} - d_2\right)}{a_0 c_2}
\]
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 $\xi$ and $z$ 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 $d(rV_\theta)_0/du$ are calculated by subroutines LAMDAF and RVTHTA when they are called by NEWRHO to calculate $\lambda$ or $(rV_\theta)_0$.

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

\[ H_0 - H_1 = \omega \left[ (rV_\theta)_0 - \lambda \right] \]  

(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 \( V_\theta \) is given as input (\( LAMVT = 1 \)), certain subroutines must be reinitialized in every iteration. There are two possibilities: one that \( V_\theta \) is given as a function of stream function (\( LSFR = 0 \)), and the second that \( V_\theta \) 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.
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 $\theta$-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...
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 \( \phi \) 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 \( \theta \) 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

\[
\text{Figure 27. - Full alternate mesh on which gradients of } \theta \text{ are obtained.}
\]

\[
\text{Figure 28. - Relation of semi-alternate mesh to } z \text{- and } r \text{-directions.}
\]
(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 $\alpha_{bs}$ (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 $\theta$ and $\partial \theta / \partial s_{bs}$ at the ZPC, RPC points from known $\theta$ at the ZBL, RBL points.

(6) Use SPLINT calls in the $s_{bs}$-direction to calculate blade thickness in the $\theta$-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 $z$, $r$, $\theta$, blade thickness, $\alpha_{bs}$, 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 ZPC1, RPC1).

(9) Calculate $r$-coordinates (RPC2) 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 (ZPC2).

(10) Calculate arc length SZRPC up the $t'$-lines using the ZPC2, RPC2 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 $\theta$ and $\partial \theta / \partial t'$ at the full alternate mesh points (ZPC2, RPC2) from known $\theta$ at the semi-alternate mesh points (ZPC1, RPC1).

(12) Use SPLINT calls in the $t'$-direction to obtain $\alpha_{bs}$, $\partial \theta / \partial s_{bs}$, and blade thickness at the full alternate mesh points (ZPC2, RPC2) from known values at the semi-alternate mesh points (ZPC1, RPC1).

(13) Store calculated values of $z$, $r$, $\theta$, blade thickness, $\alpha_{bs}$, 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:
\[
\frac{\partial \theta}{\partial z} = \frac{\partial \theta}{\partial s_{bs}} \frac{\cos \alpha_{t'}}{\cos(\alpha_{bs} + \alpha_{t'})} + \frac{\partial \theta}{\partial t'} \frac{\sin \alpha_{bs}}{\cos(\alpha_{bs} + \alpha_{t'})} \\
\frac{\partial \theta}{\partial r} = -\frac{\partial \theta}{\partial s_{bs}} \frac{\sin \alpha_{t'}}{\cos(\alpha_{bs} + \alpha_{t'})} + \frac{\partial \theta}{\partial t'} \frac{\cos \alpha_{bs}}{\cos(\alpha_{bs} + \alpha_{t'})}
\]

(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:

\[
\frac{\partial \theta}{\partial s} = \frac{\partial \theta}{\partial z} \cos \varphi + \frac{\partial \theta}{\partial r} \sin \varphi \\
\frac{\partial \theta}{\partial t} = \frac{\partial \theta}{\partial r} \cos \varphi - \frac{\partial \theta}{\partial z} \sin \varphi
\]
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 \( \frac{\partial I}{\partial t} \) and \( T \frac{\partial s}{\partial t} \). Since it is assumed that there is uniform total pressure upstream and no total pressure loss, \( \frac{\partial s}{\partial t} = 0 \). By definition, \( I = H_1 - \omega \lambda \). For uniform upstream stagnation conditions, \( H_1 \) is constant, so that

\[
\frac{\partial I}{\partial t} = -\omega \frac{\partial \lambda}{\partial t} \tag{D1}
\]

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

\[
\xi = 0 \tag{D2}
\]

\[
\xi = \omega \frac{\partial \lambda}{\partial t} \tag{D3}
\]

However, in the program, the variable ZETA is not used for this purpose, but \( \omega \frac{\partial \lambda}{\partial t} \) 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 \( \theta \)-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.
(3) Initially estimate the tangency angles $\beta_1$ and $\beta_2$ (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, $R_I$, using $\theta$-coordinates at points A and B:

$$R_I = \frac{r_{1e} \theta_A - r_{1e} \theta_B}{2} \cos \left( \frac{\beta_1 + \beta_2}{2} \right)$$

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

$$m_1 = R_I [1 - \sin(\beta_1)]$$

$$m_2 = R_I [1 + \sin(\beta_2)]$$

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

(7) Using $r_{1e}$ and the surface slopes, calculate a new estimate of the tangency angles $\beta_1$ and $\beta_2$:
\[ \beta_1 = \tan^{-1} \frac{r_\text{le} \, d\theta_1}{dm} \]

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

(8) Estimate new leading-edge radius, using updated tangency point \( \theta \)-coordinates and \( \beta \)'s:

\[ R_{\text{Inew}} = \frac{r_\text{le} \, \theta_1 - r_\text{le} \, \theta_2}{\cos \beta_1 + \cos \beta_2} \]

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

\[ RI = \frac{(\text{DAMP})(RI) + R_{\text{Inew}}}{\text{DAMP} + 1}. \]

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

If the calculated \( RI \) ever becomes negative in this procedure, \( \text{DAMP} \) is incremented by 1 and the process is begun again at step 3. The counter \( \text{ICOUNT} \) is incremented by 1 for each loop through steps 5 to 9. If \( \text{ICOUNT} \) 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, \( \rho_{bf} \), can be calculated from the free-stream density \( \rho_{fs} \) 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:

\[
\tan \beta_{bf} = \left( \frac{\rho_{bf}}{\rho_{fs}} \right) \frac{B_{le}}{\text{Pitch}} \tan \beta_{fs}
\]  

(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

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

(F2)

Finally, \( W_{bf} \) is related to \( \rho_{bf} \) by the relation

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

(F3)

where

215
\[ P_{\text{loss}} = \frac{p_{\text{ideal}} - p'}{p_{\text{ideal}}} \]

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 \( \rho W_m \) especially with large blockage.

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

\[ k_1 = \frac{(\tan \beta_{fs})B_{le}}{\rho_{fs}(\text{Pitch})} \]  \hspace{2cm} (F4)

\[ k_2 = \frac{\left( \frac{\rho_{fs} W_{fs}}{B_{le}} \right)^2}{1 + \tan^2 \beta_{fs}} \]  \hspace{2cm} (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 \( W_{bf}^2 \) followed by the new \( \rho_{bf} \). The equations for this (from eq. (F1) to (F3)) are

\[ \tan \beta_{bf} = k_1 \rho_{bf} \]  \hspace{2cm} (F6)

\[ W_{bf}^2 = k_2 \left( 1 + \tan^2 \beta_{bf} \right) \]  \hspace{2cm} (F7)

\[ \rho_{bf} = \rho_1^* (1 - \text{Ploss}) \left[ 1 - \frac{W_{bf}^2 + 2\omega \lambda - (\omega r)^2}{2 c_p T_1} \right]^{1/(\gamma-1)} \]  \hspace{2cm} (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 \( W_{bf}^2 \) followed by the new \( \rho_{bf} \). The equations for this (from eq. (F1)
to (F3)) are

\[ W_{bf}^2 = \left(1 - \frac{\rho_{bf}}{\rho_i(1 - Ploss)}\right)^{-1} \left[2c_p T_i^* - 2\omega_\lambda + (\omega\nu)^2\right] \]

(Equation F9)

\[ \rho_{bf} = \sqrt{\frac{k_2}{W_{bf}^2 - k_2 k_1}} \]

(Equation 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 \( \rho W_m \). This occurs when \( d(\rho W_m)/dW_m = 0 \). By differentiating

\[ \rho W_m = \rho_i \left[1 - \frac{W_m^2 + W_\theta^2 + 2\omega_\lambda - (\omega\nu)^2}{2c_p T_i^*}\right]^{1/(\gamma - 1)} \]

we obtain

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

Hence,

\[ \left(W_{msonic}^2\right)^{\gamma RT} \]

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
\[(W_m)_{\text{sonic}} = \sqrt{\frac{2\gamma RT'}{\gamma + 1} - \frac{\gamma - 1}{\gamma + 1} W_0^2} \quad (F12)\]

In subroutine LINDV, \((W_m)_{\text{sonic}}\) is calculated by equation (F12). If \((W_m)_{fs}\) is less than \((W_m)_{\text{sonic}}\), equations (F6) to (F8) are used to calculate \(\rho_{bf}\) iteratively. And if \((W_m)_{fs}\) is greater than \((W_m)_{\text{sonic}}\), 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 \((\rho W_m)_{\text{sonic}}\). In this case an error message is printed and \(\rho_{bf}\) is set equal to \(\rho_{fs}\). 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.
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 shows the same quadrilateral 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

$$f_x = \frac{P_1P_7}{P_1P_2} \quad \text{(G1)}$$

$$f_y = \frac{P_1P_5}{P_1P_3} \quad \text{(G2)}$$

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,

$$f_y = \frac{x_{51}}{x_{31}} = \frac{y_{51}}{y_{31}} = \frac{x_{62}}{x_{42}} = \frac{y_{62}}{y_{42}} \quad \text{(G3)}$$

The equation of line $P_5P_6$ is

$$\frac{y - y_5}{x - x_5} = \frac{y_{65}}{x_{65}} \quad \text{(G4)}$$
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

$$\begin{align*}
    y_5 &= y_1 - f_y y_{13} \\
    y_6 &= y_2 + f_y y_{42} \\
    x_5 &= x_1 - f_y x_{13} \\
    x_6 &= x_2 + f_y x_{42}
\end{align*}$$

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

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

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 $f_y$:

$$af_y^2 + bf_y + c = 0$$

where

$$\begin{align*}
    a &= y_{13} x_{42} - x_{13} y_{42} \\
    b &= x_{13} y_{02} - y_{13} x_{02} + y_{01} x_{42} - x_{01} y_{42} \\
    c &= y_{01} x_{21} - x_{01} y_{21}
\end{align*}$$

In a similar manner, we can obtain a quadratic in $f_x$: 222
\[ af_x^2 + bf_x + c = 0 \]  

where

\[
\begin{align*}
    a &= y_{12x43} - x_{12y43} \\
    b &= x_{12y03} - y_{12x03} + y_{01x43} - x_{01y43} \\
    c &= y_{01x31} - x_{01y31}
\end{align*}
\]  

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 binomial series, to obtain

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

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

\[
z_0 = z_1(1 - f_x)(1 - f_y) + z_2f_x(1 - f_y) + z_3(1 - f_x)f_y + z_4f_xf_y
\]  

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

\[
z_0 = z_1(1 - f_x)(1 - f_y) + z_2f_x(1 - f_y) + z_3(1 - f_x)f_y + z_4f_xf_y
\]
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, \(c_p T^1 \cdot \omega \lambda\), 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

\(W^*_j\) first estimate of \(W_{j+1}\), meters/sec

\(W^{**}_j\) second estimate of \(W_{j+1}\), meters/sec

w mass flow, kg/sec

z axial coordinate, meters
\( \alpha \) angle between meridional streamline and axis of rotation (fig. 4, part I), rad
\( \beta \) angle between relative velocity vector and meridional plane (fig. 4, part I), rad
\( \beta_{\text{abs}} \) angle between absolute velocity vector and meridional plane
\( \gamma \) specific-heat ratio
\( \zeta \) coefficient in stream-function equation, defined in eq. (A3), meters/sec^2
\( \theta \) relative angular coordinate (fig. 4, part I), rad
\( \lambda \) prerotation, \( (rV_\theta)_l \), meters^2/sec
\( \xi \) coefficient in stream-function equation, defined in eq. (A2), 1/meter
\( \rho \) density, kg/meter^3
\( \varphi \) angle between s-coordinate line and axis of rotation (fig. 25), rad
\( \Omega \) overrelaxation factor
\( \omega \) rotational speed (fig. 4, part I), rad/sec

Subscripts:
- \( \text{av} \): average blade-to-blade value
- \( b \): blade
- \( \text{bf} \): blade flow
- \( \text{cr} \): critical
- \( \text{fs} \): free stream
- \( \text{hub} \): hub
- \( i \): inlet
- \( l \): blade surface facing direction of positive rotation
- \( \text{le} \): leading edge
- \( m \): component in direction of meridional streamline
- \( \text{mid} \): midchannel
- \( o \): outlet
- \( r \): component in radial direction
- \( s \): component in s-direction
- \( t \): component in t-direction
- \( \text{te} \): trailing edge
- \( \text{tip} \): tip

226
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


