NASA TECHNICAL NOTE

NASA TN D-5044

c. 1

NASA TN D-5044

FORTRAN PROGRAM FOR CALCULATING VELOCITIES AND STREAMLINES ON A BLADE-TO-BLADE STREAM SURFACE OF A TANDEM BLADE TURBOMACHINE

by Theodore Katsanis and William D. McNally

Lewis Research Center
Cleveland, Ohio

NATIONAL AERONAUTICS AND SPACE ADMINISTRATION • WASHINGTON, D. C. • MARCH 1969

FORTRAN PROGRAM FOR CALCULATING VELOCITIES AND STREAMLINES ON A BLADE-TO-BLADE STREAM SURFACE OF A TANDEM BLADE TURBOMACHINE

By Theodore Katsanis and William D. McNally

Lewis Research Center
Cleveland, Ohio

NATIONAL AERONAUTICS AND SPACE ADMINISTRATION

For sale by the Clearinghouse for Federal Scientific and Technical Information
Springfield, Virginia 22151 – CFSTI price $3.00

ABSTRACT

A FORTRAN IV computer program was written that gives the blade-to-blade solution of the two-dimensional, subsonic, compressible (or incompressible), nonviscous flow problem for a circular or straight infinite cascade of tandem or slotted turbomachine blades. The blades may be fixed or rotating. The flow may be axial, radial, or mixed. The results include streamline coordinates, velocity magnitude and direction throughout the passage, and the blade-surface velocities. The method is based on the stream function using an iterative solution of nonlinear finite-difference equations.

CONTENTS

Page

SUMMARY 1

INTRODUCTION 2

SYMBOLS 3

MATHEMATICAL ANALYSIS 5

NUMERICAL EXAMPLES 9
- Axial-Flow Turbine Rotor Cascade 9
- Mixed-Flow Impeller 12

DESCRIPTION OF INPUT AND OUTPUT 18
- Input 18
- Instructions for Preparing Input 24
  - Units of measurement 24
  - Blade and stream-channel geometry 24
  - Inlet and outlet flow angles 25
  - Defining mesh 25
  - Overrelaxation factor 25
  - Format for input data 27
  - Incompressible flow 27
  - Straight infinite cascade 27
  - Axial flow 27
- Output 27

ERROR CONDITIONS 37

PROGRAM PROCEDURE 38
- Conventions Used in Program 40
- Labeled COMMON Blocks 41
- Subroutine INPUT 41
  - Reading and printing of input 41
  - Calculation of constants and initialization 41
- Subroutine PRECAL 42
  - Calculation of constants 42
  - Calculation of vertical mesh line arrays 42
  - Calculation of horizontal mesh line arrays 42

Subroutine COEF 42
Calculating coefficients and constants throughout region 42
Corrections to coefficients and constants 42
Subroutine COEFBB 43
Subroutine HRB 43
Subroutine AAK 43
Subroutine BDRY12 43
Subroutine BDRY34 43
Subroutine SOR 44
Estimation of optimum overrelaxation factor 44
Solution of matrix equation by subroutine SOR 44
Subroutine SLAX 44
Calculating ρW_m throughout region 45
Plotting streamlines 45
Subroutine SLAV 45
Calculation of $\partial u/\partial \theta$ and streamline locations 45
Calculation of ρW_m 45
Calculation of mass flow parameter ρW on blade surfaces 45
Subroutine TANG 45
Location of points on horizontal mesh lines 46
Calculation of ρW_θ 46
Calculation of mass flow parameter ρW and flow angles at interior points . . . 46
Calculation of mass flow parameter ρW on blade surfaces 46
Subroutine SEARCH 47
Subroutine VELOCY 47
Solving for densities and velocities throughout region 47
Plotting of velocities 47
Subroutine VEL 47
Calculation of ρ and W 47
Printing of velocities 48
Subroutine BLCD 48
Function IPF 49
Main Dictionary 49
Program Listing for Subroutines Using Main Dictionary 60
Subroutine MHORIZ 87
Subroutine DENSTY 88
Subroutine ROOT 90
Subroutine SPLINE 91
Subroutine SPLN22 93

Subroutine SPLINT 94
Lewis Library Subroutine TIME1 96

CONCLUDING REMARKS 96

APPENDIXES -
A - FINITE-DIFFERENCE APPROXIMATION 97
B - NUMERICAL TECHNIQUES USED IN PROGRAM 102

REFERENCES 107

FORTRAN PROGRAM FOR CALCULATING VELOCITIES AND STREAMLINES ON A BLADE-TO-BLADE STREAM SURFACE OF A TANDEM BLADE TURBOMACHINE

by Theodore Katsanis and William D. McNally

Lewis Research Center

SUMMARY

A FORTRAN IV computer program was written that gives the blade-to-blade solution of the two-dimensional, subsonic, compressible (or incompressible), nonviscous flow problem for a circular or straight infinite cascade of tandem or slotted turbomachine blades. The blades may be fixed or rotating. The flow may be axial, radial, or mixed, and there may be a change in stream-channel thickness in the through-flow direction.

The program input consists of blade and stream-channel geometry, total flow conditions, inlet and outlet flow angles, blade-to-blade stream-channel weight flow, and the portion of this weight flow that passes between the front and rear tandem blades (through the slot). The output includes blade-surface velocities, velocity magnitude and direction at all interior mesh points in the blade-to-blade passage, and streamline coordinates throughout the passage.

The method is based on the stream function. The simultaneous, nonlinear, finite-difference equations of the stream function are solved by using two major levels of iteration. The inner iteration consists of the solution of simultaneous linear equations by successive overrelaxation, using an estimated optimum overrelaxation factor. The outer iteration then changes the coefficients of the simultaneous equations to correct for compressibility.

This report includes the FORTRAN IV computer program with an explanation of the equations involved, the method of solution, and the calculation of velocities. Numerical examples are included to illustrate the use of the program, and to show the results which are obtained.

INTRODUCTION

An effort is being made to design compressors and turbines with smaller diameters, fewer stages, and fewer blades per stage. All these factors tend to increase diffusion. Therefore, it is desired to design blades with high diffusion, and at the same time to avoid flow separation. Several ideas for aerodynamic design to permit high diffusion without separation are being investigated, both theoretically and experimentally. Two promising concepts are the tandem blade and the slotted blade.

In the design of tandem or slotted blade rows for compressors or turbines, an analysis is desirable which will give velocity distributions from blade to blade, and particularly over the blade surfaces. Stanitz (refs. 1 and 2) has shown that finite-difference solutions of the stream-function differential equation can be used to obtain these results. Computer programs have been written which generate coefficients for the difference equations, solve the equations, and differentiate the resulting values of stream function to obtain velocities throughout the blade-to-blade passage and on the blade surfaces. This has been done previously by the first author (ref. 3) for a turbomachine with a single blade row without slots.

This report extends the analysis of reference 3 to tandem or slotted blades. A computer program has been written to obtain the numerical solution for ideal, subsonic, compressible (or incompressible) flow for an axial-, radial-, or mixed-flow circular cascade of turbomachine blades. The program may also be used for a straight infinite cascade. The blades may be overlapping or nonoverlapping in the meridional flow direction and may be fixed or rotating. The program may also be used to analyze a turbomachine with one set of splitter blades (see section Mixed-Flow Impeller, p. 12). The coordinates used are meridional streamline distance and angular coordinate in radians.

This report includes the FORTRAN IV computer program (called TANDEM) that was developed, with an explanation of the equations involved and the method of solution. A tandem axial gas-turbine rotor cascade and a mixed-flow impeller are analyzed to illustrate the use of the program. The results obtained for the axial turbine are compared with experimental data. The impeller results are compared with previous analytical results. The report is organized so that the engineer desiring to use the program needs to read only the sections MATHEMATICAL ANALYSIS, NUMERICAL EXAMPLES, and DESCRIPTION OF INPUT AND OUTPUT. Information of interest to a programmer is contained in the sections DESCRIPTION OF INPUT AND OUTPUT and PROGRAM PROCEDURE and in the appendixes.

A TANDEM source deck on tape is available from COSMIC (Computer Software Management and Information Center), Computer Center, University of Georgia, Athens, Georgia 30601. The program number can be obtained from the authors.

SYMBOLS

A	coefficient matrix, eq. (A7)
a_{ij}	typical element of matrix A
$a_0, a_1, a_2, a_3, a_4, a_{12}, a_{34}$	coefficients in eq. (A2)
b	stream-channel thickness normal to meridional streamline, meters
b_{12}, b_{34}	quantities in eq. (A2)
c_p	specific heat at constant pressure, joule/(kg)($^{\circ}$K)
h	spacing between adjacent points, eqs. (A1) to (A4); see fig. 17
$\underline{k}$	constant vector, $\begin{pmatrix} k_1 \\ \cdot \\ \cdot \\ \cdot \\ k_n \end{pmatrix}$, eq. (A7)
m	meridional streamline distance, see figs. 2 and 3
n	number of unknown mesh points
R	gas constant, joule/(kg)($^{\circ}$K)
r	radius from axis of rotation to meridional stream-channel mean line, meters
s	angular blade spacing or pitch, rad
T	temperature, $^{\circ}$K
u	stream function
$\underline{u}$	discrete approximation to stream function at n mesh points, $\begin{pmatrix} u_1 \\ \cdot \\ \cdot \\ \cdot \\ u_n \end{pmatrix}$
$\underline{u}^m$	m^{th} iterate of $\underline{u}$ $\begin{pmatrix} u_1^m \\ \cdot \\ \cdot \\ \cdot \\ u_n^m \end{pmatrix}$

V	absolute fluid velocity, meters/sec
W	fluid velocity relative to blade, meters/sec
w	mass flow per blade flowing through stream channel, kg/sec
z	axial coordinate, meters
α	angle between meridional streamline and axis of rotation, rad; see fig. 1
β	angle between relative velocity vector and meridional plane, rad; see fig. 1
γ	specific-heat ratio
η	outer normal to region
θ	relative angular coordinate, rad; see fig 1
λ	prerotation $(rV_\theta)_{in}$, meters^2/sec
ρ	density, kg/meters^3
Ω	overrelaxation factor, eq. (A8)
ω	rotational speed, rad/sec; see fig. 1

Subscripts:

cr	critical velocity
i	dummy variable
in	inlet or upstream
j	dummy variable
le	leading edge
m	component in direction of meridional streamline
out	outlet or downstream
te	trailing edge
θ	tangential component
0, 1, 2, 3, 4	quantities at these locations in finite-difference expression, eqs. (A1) to (A6); see fig. 17

Superscripts:

T	transpose of vector or matrix
'	absolute stagnation condition
''	relative stagnation condition

MATHEMATICAL ANALYSIS

It is desired to determine the flow distribution through a stationary or rotating cascade of tandem blades on a blade-to-blade surface. The following simplifying assumptions are used in deriving the equations and in obtaining a solution:

(1) The flow is steady relative to the blade.

(2) The fluid is a perfect gas or is incompressible.

(3) The fluid is nonviscous.

(4) There is no loss of energy.

(5) The flow is absolutely irrotational.

(6) The blade-to-blade surface is a surface of revolution. (This does not exclude straight infinite cascades.)

(7) The velocity component normal to the blade-to-blade surface is zero.

(8) The stagnation temperature is uniform across the inlet.

(9) The velocity magnitude and direction is uniform across both the upstream and downstream boundaries.

(10) The relative velocity is subsonic everywhere.

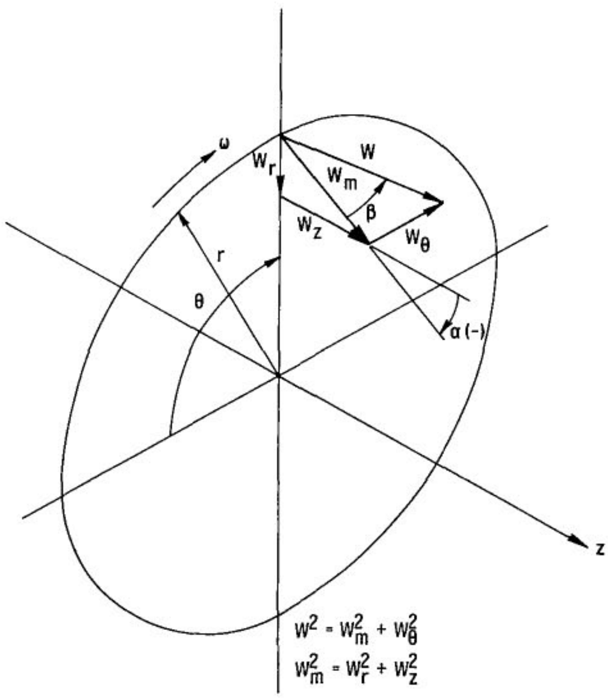


Figure 1. - Cylindrical coordinate system and velocity components.

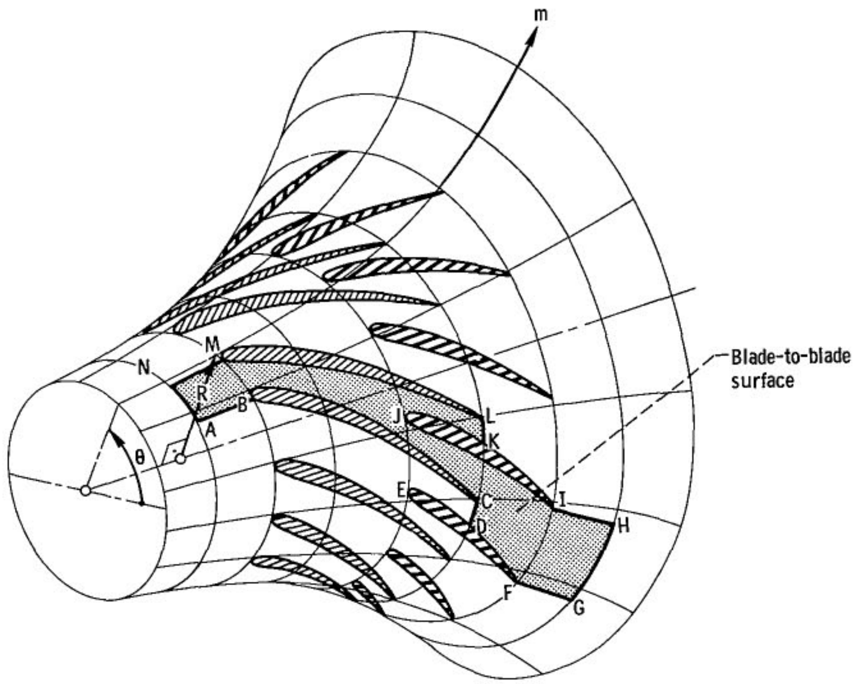


Figure 2. - Blade-to-blade surface of revolution.

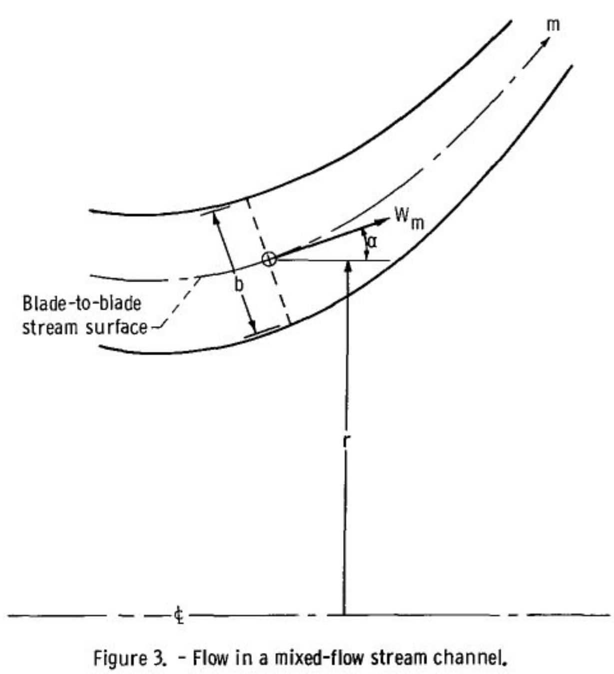


Figure 3. - Flow in a mixed-flow stream channel.

The flow may be axial, radial, or mixed, and there may be a variation in the stream-channel thickness b in the through-flow direction. The proportion of flow between the front and rear blades must be specified as an input to the program. This input may be difficult for the user to estimate; however, correlation with experimental work may yield more reliable values.

The coordinate system is shown in figure 1. Since the variables r and z are not independent on a stream surface, one variable can be eliminated. Therefore, r and θ or z and θ could be used as independent variables. However, for generality, it is better to use the meridional streamline distance m in place of r and z as an independent variable (see fig. 2). Then, m and θ are the two basic independent variables. A stream channel is therefore defined by specifying a meridional streamline radius r and a stream-channel thickness b at several meridional locations m (see fig. 3).

For the mathematical formulation of the problem, the stream function is used. The stream function u used herein is related to the stream function ψ defined in reference 4 by $u = -\psi/w$. With this substitution in equation 12(9) of reference 4 we obtained the basic differential equation which must be satisfied by the stream function under the given assumptions:

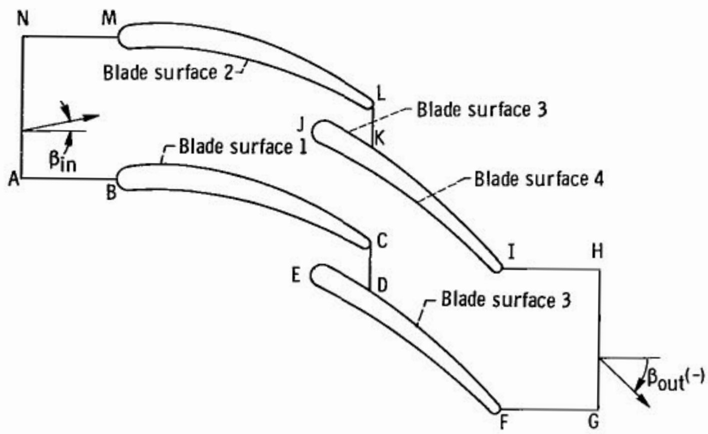
$$\frac{1}{r^2}\frac{\partial^2 u}{\partial \theta^2} + \frac{\partial^2 u}{\partial m^2} - \frac{1}{r^2}\frac{1}{\rho}\frac{\partial \rho}{\partial \theta}\frac{\partial u}{\partial \theta} + \left[\frac{\sin\alpha}{r} - \frac{1}{b\rho}\frac{\partial(b\rho)}{\partial m}\right]\frac{\partial u}{\partial m} = \frac{2b\rho\omega}{w}\sin\alpha \tag{1}$$

The stream function u has the value 0 on the upper surface of the leading blade and 1 on the lower surface of the leading blade. Also, the derivatives of the stream function satisfy the equations

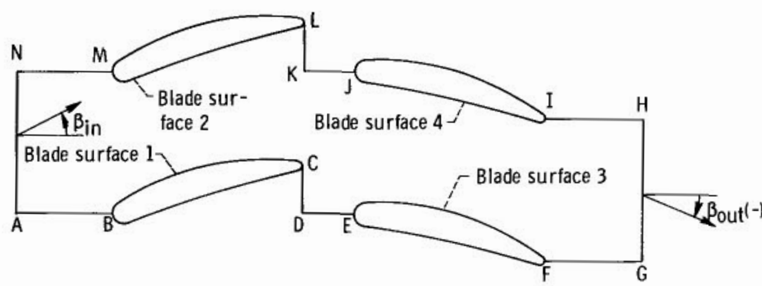
$$\frac{\partial u}{\partial m} = -\frac{b\rho}{w}W_\theta \tag{2}$$

$$\frac{\partial u}{\partial \theta} = \frac{b\rho r}{w}W_m \tag{3}$$

For the solution of equation (1), a finite region is considered (as indicated in fig. 4) with the condition that the flow along corresponding upper and lower portions of the boundary is the same. For example, the flow along AB is the same as along NM. Also, it is assumed that AN is sufficiently far upstream so that the flow is uniform along this boundary, and that the flow angle β_{in} is known. Similarly, it is assumed that the flow is uniform along GH, and that the flow angle β_{out} is known. For an actual blade row, β_{out} may usually be determined by means of experimentally determined rules. Also, it is assumed the flow split is known; that is, the percentage of flow which passes between the



(a) Overlap case.



(b) Nonoverlap case.

Figure 4. - Typical finite flow region.

front and rear blades. Specifying β_{out} and the flow split is mathematically equivalent to specifying the locations of the stagnation points on the trailing edges of both blades.

Since equation (1) is elliptic for subsonic flow, boundary conditions for the entire boundary ABCDEFGHIJKLMNA are required. Along BC, $u = 0$; along LM, $u = 1$; along EF, u is equal to the negative of the fraction of weight flow through JKL; and along IJ, u is equal to the fraction of weight flow crossing a line joining C and J. Along AB, CD, FG, HI, KL, and MN, a periodic condition exists; that is, the value of u along MN, KL, and HI is exactly 1.0 greater than it is along AB, CD, and FG. The same condition holds along DE and JK in the nonoverlapping case (fig. 4(b)).

Along AN and GH, $\partial u/\partial \eta$ is known, where η is in the direction of the outer normal. From equations (2) and (3), since $W_\theta/W_m = \tan \beta$,

$$\frac{\partial u}{\partial m} = - \frac{\partial u}{r\ \partial \theta} \tan \beta$$

Along AN and GH,

$$\frac{\partial u}{\partial \theta} = \frac{u(N) - u(A)}{s} = \frac{1}{s}$$

where s is the angular blade spacing, so that

$$\left(\frac{\partial u}{\partial \eta}\right)_{in} = \frac{\tan \beta_{in}}{sr_{in}} \qquad \text{along AN} \tag{4}$$

$$\left(\frac{\partial u}{\partial \eta}\right)_{out} = - \frac{\tan \beta_{out}}{sr_{out}} \qquad \text{along GH} \tag{5}$$

These are the boundary conditions required to determine a solution to equation (1). The method used for the numerical solution of equation (1) is described in appendix A. The numerical solution involves two levels of iteration because equation (1) is nonlinear. The inner iteration is required to solve equation (1) when it is linearized, and the nonlinear solution is approached by the outer iteration.

After computing a numerical solution to equation (1) in a given flow region, the velocity at any point can be computed from equations (2) and (3) by using numerical differentiation. The streamlines are located by the contours of equal stream-function values.

NUMERICAL EXAMPLES

To illustrate the use of the program and the type of results which can be obtained, two numerical examples are given. The first example is an axial-flow turbine and the other is a mixed-flow impeller.

Axial-Flow Turbine Rotor Cascade

This example is a two-dimensional axial-flow turbine cascade currently undergoing testing at Lewis Research Center. This blade is a modified version of a tandem blade reported in reference 5. It has a blunt leading edge on the rear blade in order to achieve a converging channel between the blades, and it has a wider slot than that reported in reference 5.

The blade shape in m, θ coordinates and the blade-to-blade solution region are shown in figure 5. Input for this example is given in table I. Blade-surface velocities

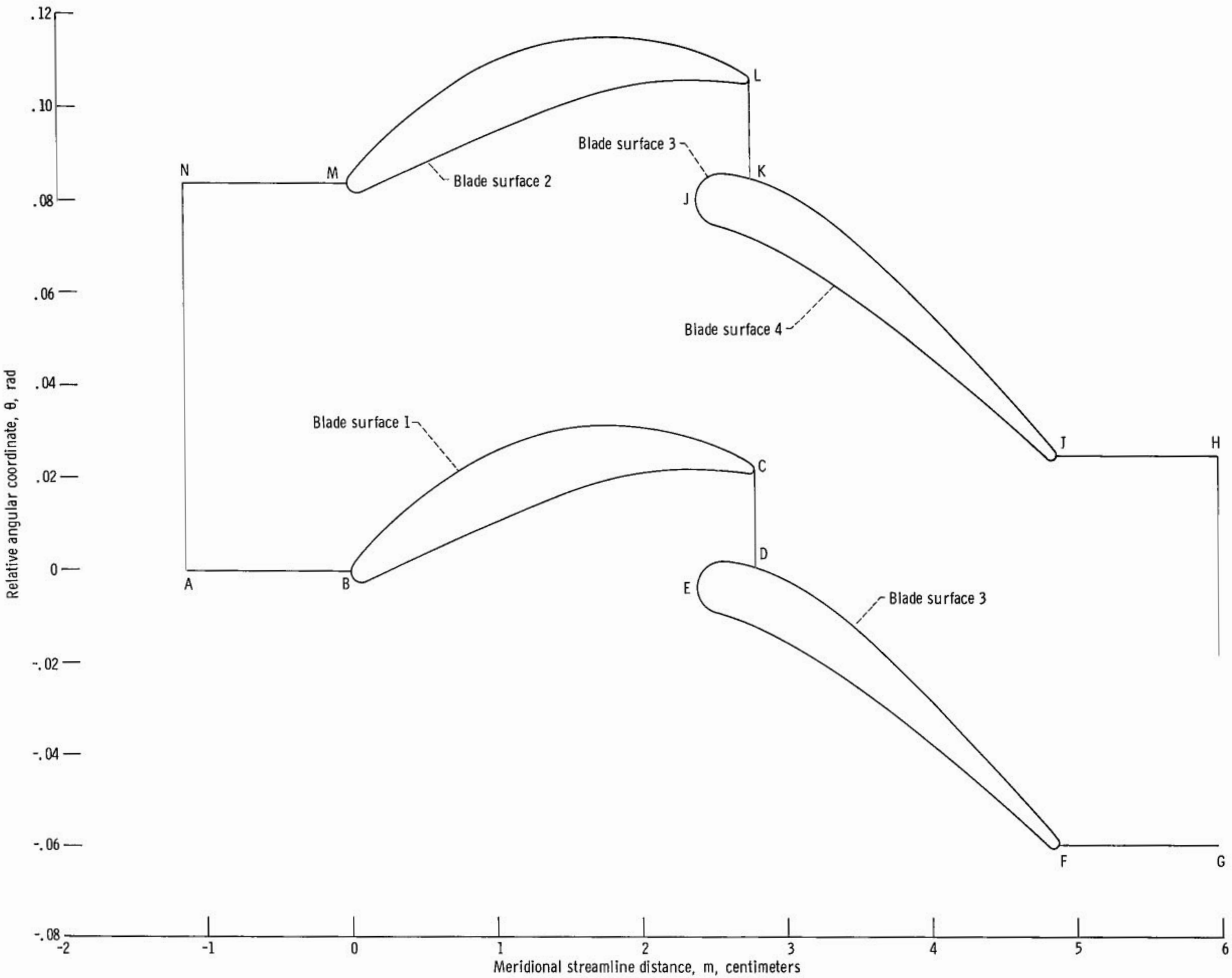


Figure 5. - Blade-to-blade flow region for tandem axial turbine rotor.

TABLE I. - INPUT FOR AXIAL-FLOW TURBINE ROTOR CASCADE

```
  MODIFIED TANDEM AXIAL TURBINE ROTOR
   GAM            AR              TIP            RHOIP           WTFL            WTFLSP          OMEGA           ORF
 1.4000000      287.05300      288.15000       1.2250000      0.3152000E-01   0.1134700E-01   -0              0
   BETAI          BETAO          CHORDF          STGRF           CHORDR          STGRR           MLER            THLER
 48.000000     -47.000000      0.2847000E-01   0.2133300E-01   0.2515000E-01  -0.5459000E-01   0.2441000E-01  -0.3607000E-02
MBI  MBO MBI2 MBO2  MM  NBBI  NBL NRSP
10   32   29   49   58   20   76    2


  BLADE SURFACE 1  --  UPPER SURFACE - FRONT BLADE
   RI1            RO1            BETI1           BETO1           SPLNO1
 0.7620000E-03  0.3810000E-03  50.000000      -29.400000       7.0000000
   MSP1  ARRAY
-0              0.2570000E-02  0.7650000E-02   0.1527000E-01   0.2035000E-01   0.2543000E-01  -0
   THSP1  ARRAY
-0              0.9250000E-02  0.2118000E-01   0.2988000E-01   0.3020000E-01   0.2643000E-01  -0


  BLADE SURFACE 2  --  LOWER SURFACE - FRONT BLADE
   RI2            RO2            BETI2           BETO2           SPLNO2
 0.7620000E-03  0.3810000E-03  25.000000      -6.9000000       5.0000000
   MSP2  ARRAY
-0              0.7650000E-02  0.2035000E-01   0.2543000E-01  -0
   THSP2  ARRAY
-0              0.7140000E-02  0.2039000E-01   0.2094000E-01  -0


  BLADE SURFACE 3  --  UPPER SURFACE - REAR BLADE
   RI3            RO3            BETI3           BETO3           SPLNO3
 0.1778000E-02  0.3810000E-03  -8.1000000     -48.800000       4.0000000
   MSP3  ARRAY
 0              0.6100000E-02  0.1626000E-01   0
   THSP3  ARRAY
 0              0.1640000E-02 -0.2463000E-01   0


  BLADE SURFACE 4  --  LOWER SURFACE - REAR BLADE
   RI4            RO4            BETI4           BETO4           SPLNO4
 0.1778000E-02  0.3810000E-03  -19.700000     -42.500000       4.0000000
   MSP4  ARRAY
 0              0.6100000E-02  0.1372000E-01   0
   THSP4  ARRAY
 0             -0.1200000E-01 -0.2745000E-01   0


   MR   ARRAY
-1.0000000      1.0000000
   RMSP  ARRAY
 0.3238500      0.3238500
   BESP  ARRAY
 0.1000000E-01  0.1000000E-01


  BLDAT  AANDK  ERSOR  STRFN  SLCRD  INTVL  SURVL
    1      1      2      2      2      2      3
```

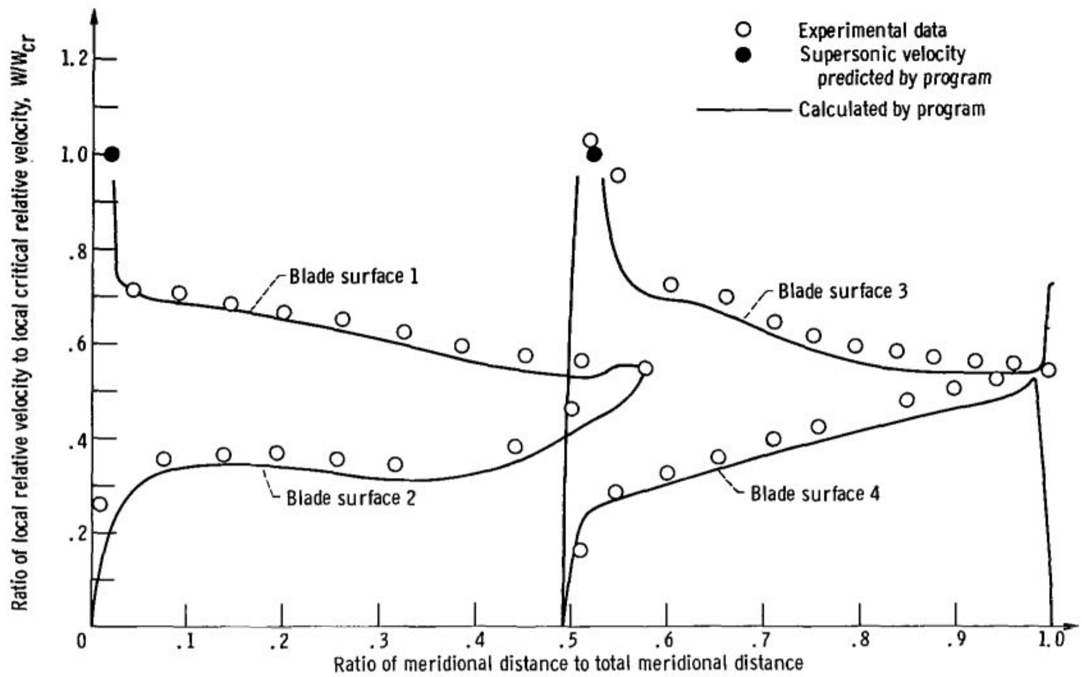


Figure 6. - Surface velocities on tandem axial turbine blade.

are plotted in figure 6, where comparison is made with unreported experimental data for the Lewis turbine cascade. There is close agreement between computed and experimental values on all four blade surfaces.

Execution time was 10 minutes for this example, and it required 16 outer iterations for final convergence to the compressible solution.

Mixed-Flow Impeller

This example is taken from reference 6. In reference 6 a similar stream-function analysis was made. The mesh was set up graphically, and the coefficients were calculated by hand. The solution of the finite-difference equations was obtained by relaxation on a computer. The analysis was done on a blade-to-blade surface of revolution midway between hub and shroud.

The coordinates of the stream channel and the stream-channel radial thickness are given by equations (1) and (2) of reference 7. The radial stream-channel thickness was corrected to obtain the normal thickness required by this program. The hub-shroud profile is shown in figure 7. The blade shape and mesh arrangement are shown in figure 8. Input for this example is given in table II. In figure 9 the blade-surface velocities obtained by the TANDEM program are compared with those obtained originally in refer-

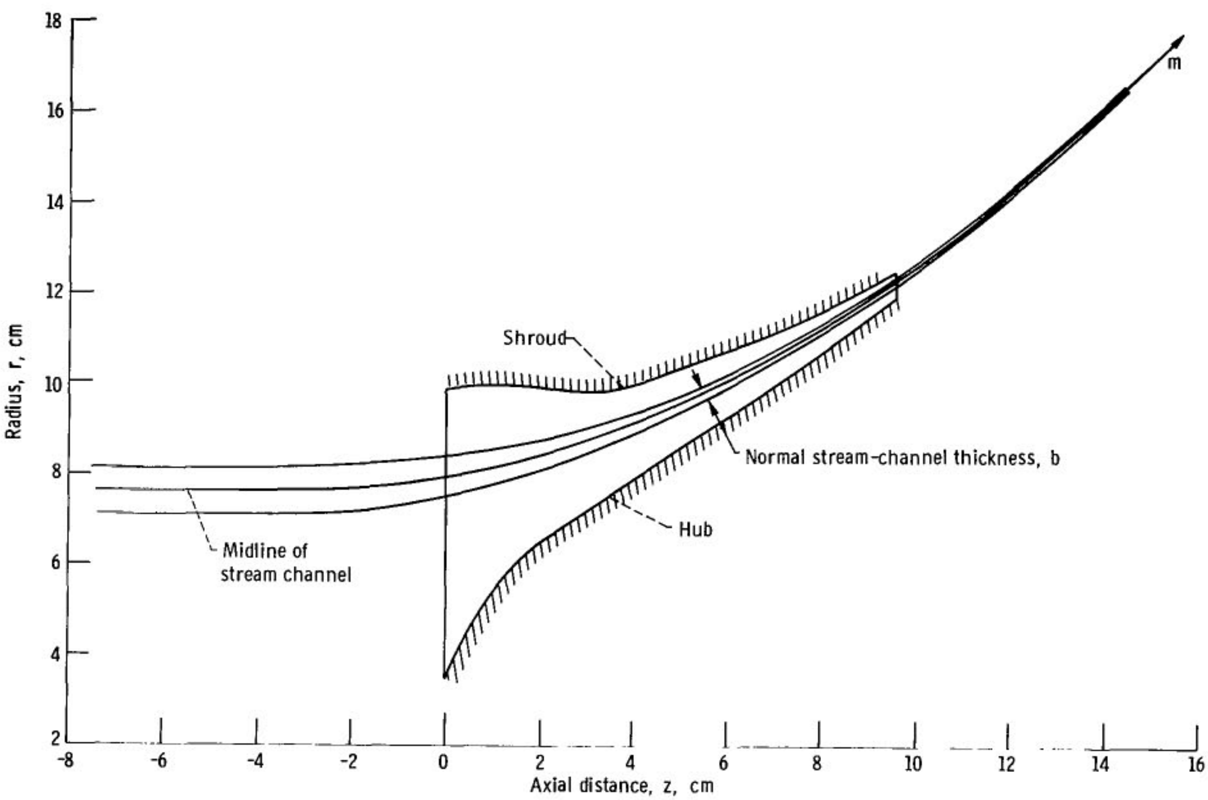


Figure 7. - Hub-shroud profile of mixed-flow impeller showing meridional section of steam tube.

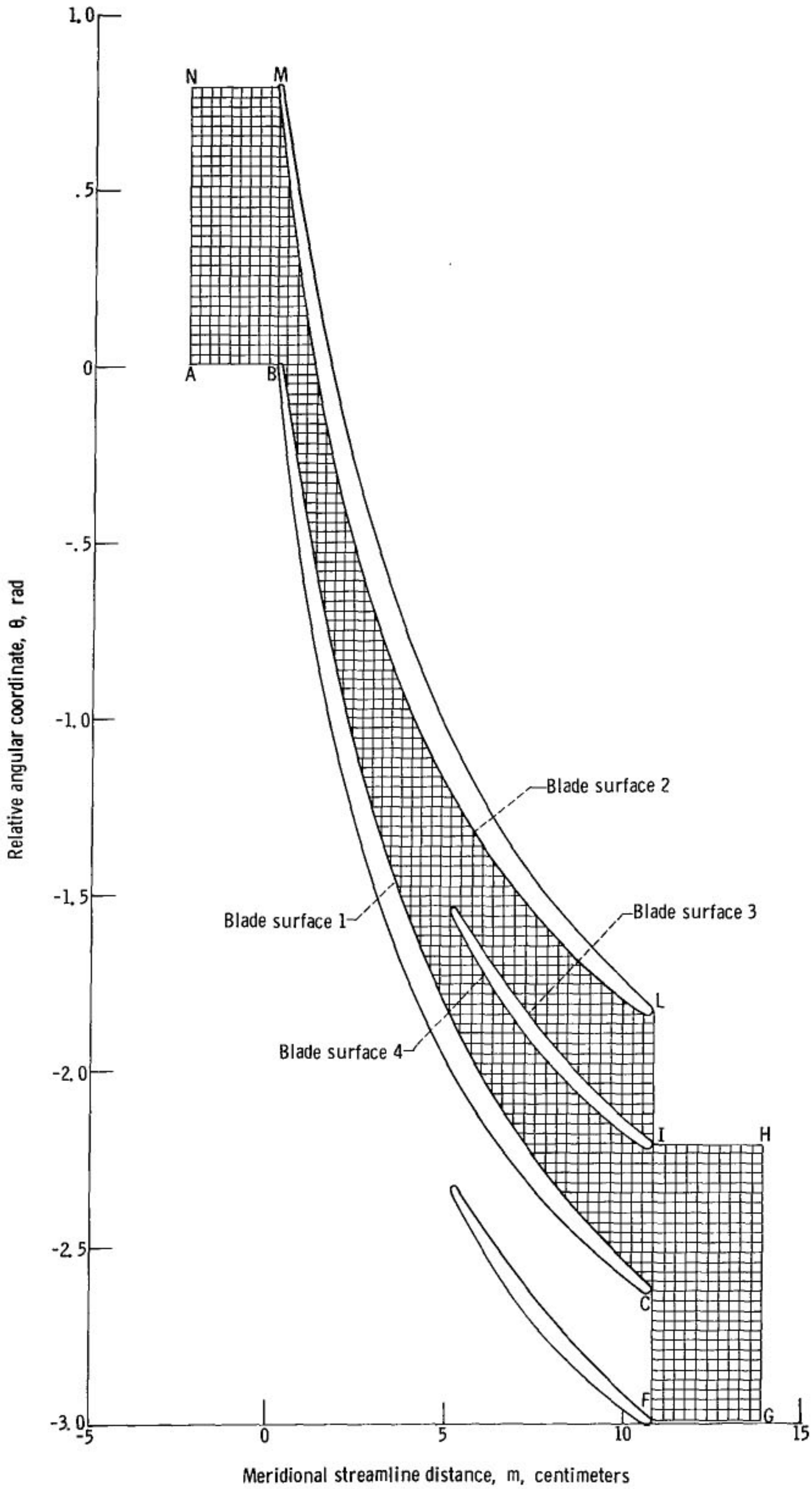
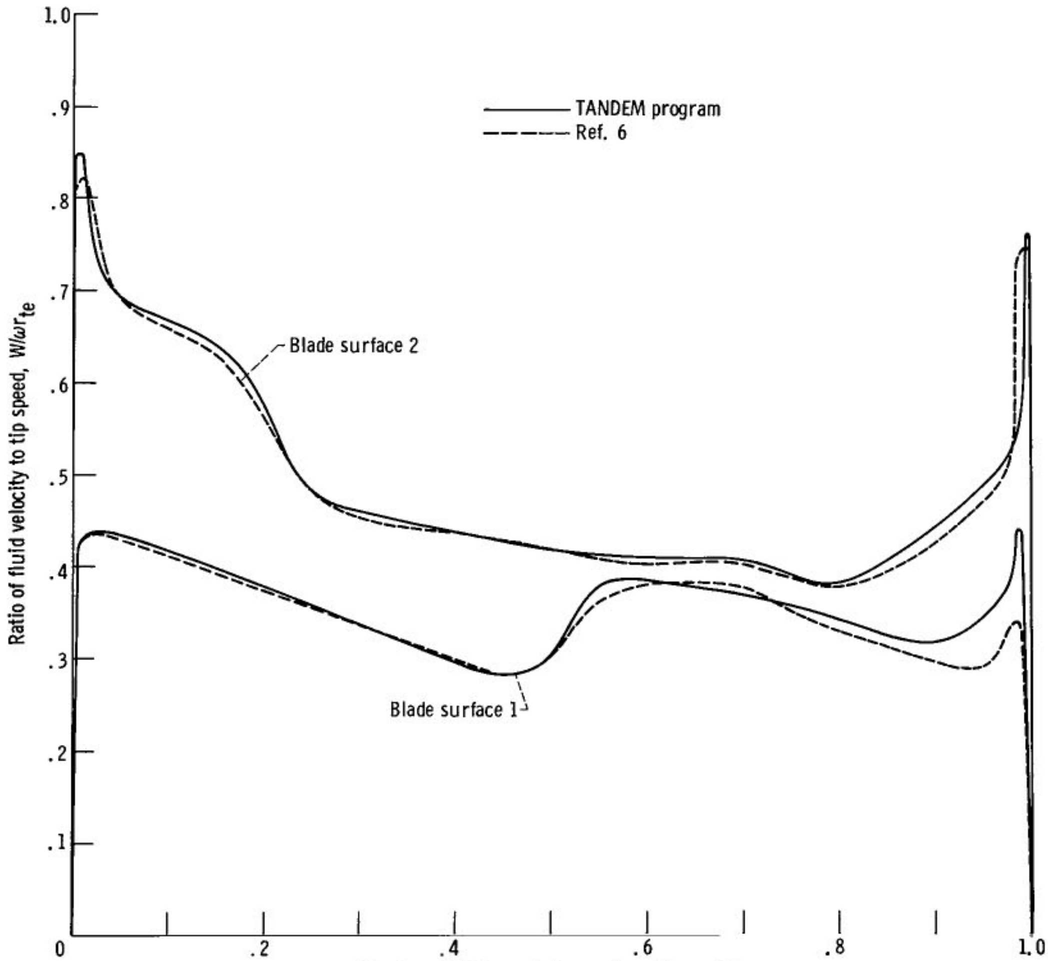


Figure 8. - Blade-to-blade surface for mixed-flow impeller, showing grid used in program.

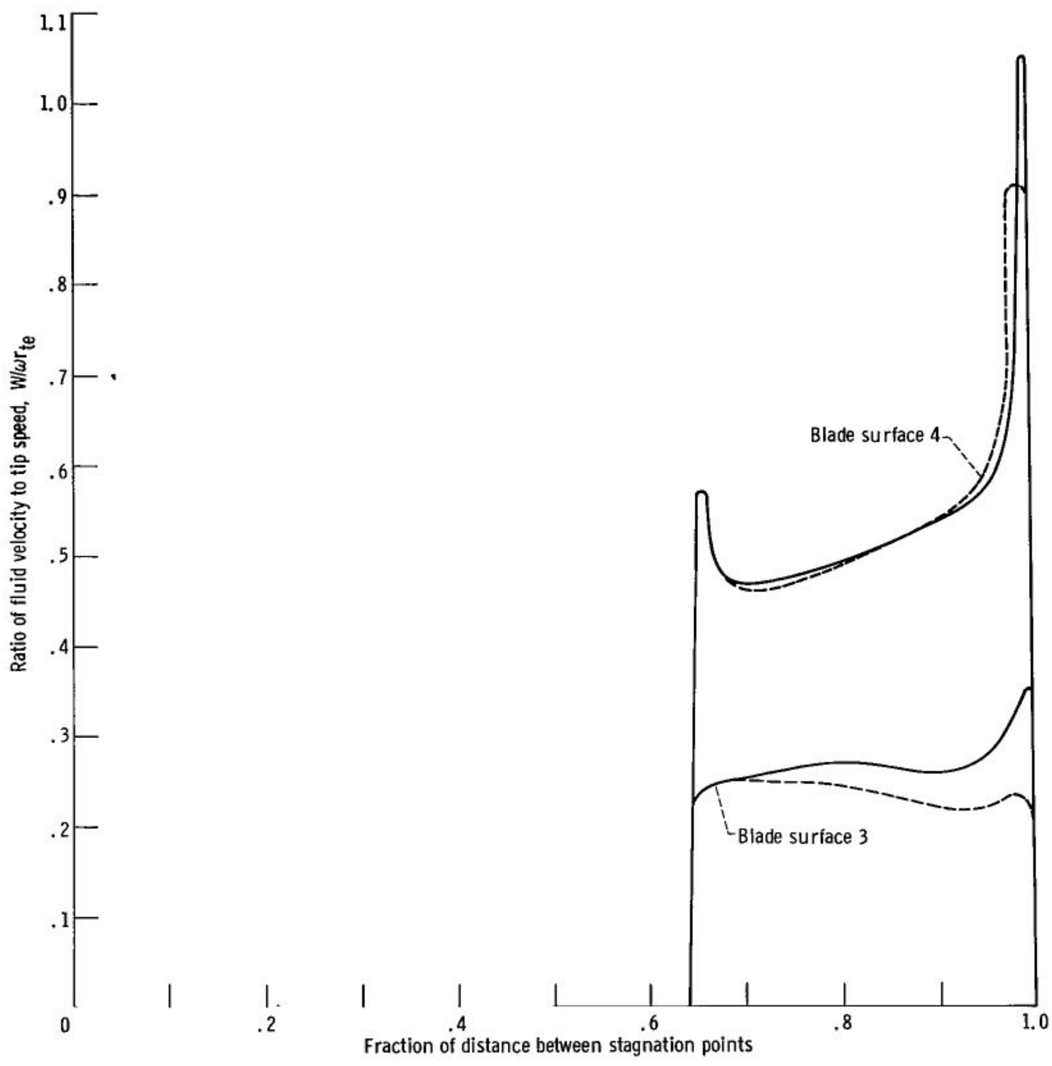
TABLE II. - INPUT FOR MIXED-FLOW IMPELLER

```
MIXED FLOW IMPELLER (NASA TN D-1186)
   GAM            AR            TIP           RHOIP          WTFL            WTFLSP          OMEGA          ORF
 1.5000000     1000.0000     1000000.0     1.0000000     0.3042000E-02   0.1351600E-02   796.00000       0
   BETAI         BETAO          CHORDF         STGRF          CHORDR          STGRR           MLER           THLER
-84.880000    -43.000000     0.1055500    -2.6290000     0.5664000E-01  -0.6649000      0.4891000E-01  -2.3434000
MBI  MBO MBI2 MBO2  MM  NBBI  NBL NRSP
10   47   28   47   57   28    8   18

 BLADE SURFACE 1  --  UPPER SURFACE - FRONT BLADE
   RI1             RO1            BETI1           BETO1           SPLNO1
 0.9140000E-03  0.1846000E-02  -80.000000     -49.000000      6.0000000
   MSP1  ARRAY
 0              0.1214000E-01  0.2651000E-01  0.4766000E-01  0.7360000E-01   0
   THSP1  ARRAY
 0             -0.6250000     -1.2330000     -1.8182000     -2.2750000       0

 BLADE SURFACE 2  --  LOWER SURFACE - FRONT BLADE
   RI2             RO2            BETI2           BETO2           SPLNO2
 0.9140000E-03  0.1846000E-02  -83.000000     -41.500000      6.0000000
   MSP2  ARRAY
 0              0.7880000E-02  0.2004000E-01  0.4006000E-01  0.6828000E-01   0
   THSP2  ARRAY
 0             -0.6310000     -1.2310000     -1.8206000     -2.2954000       0

 BLADE SURFACE 3  --  UPPER SURFACE - REAR BLADE
   RI3             RO3            BETI3           BETO3           SPLNO3
 0.1328000E-02  0.1753000E-02  -60.500000     -51.500000      6.0000000
   MSP3  ARRAY
 0              0.1307000E-01  0.2552000E-01  0.4172000E-01  0.5280000E-01   0
   THSP3  ARRAY
 0             -0.1670000     -0.3370000     -0.5262000     -0.6269000       0

 BLADE SURFACE 4  --  LOWER SURFACE - REAR BLADE
   RI4             RO4            BETI4           BETO4           SPLNO4
 0.1328000E-02  0.1753000E-02  -63.000000     -40.500000      5.0000000
   MSP4  ARRAY
 0              0.1073000E-01  0.2493000E-01  0.4172000E-01  0
   THSP4  ARRAY
 0             -0.2010000     -0.4070000     -0.5818000      0

   MR  ARRAY
-0.3124000E-01 -0.1514000E-01  0.2500000E-03  0.1065000E-01  0.1853000E-01  0.2651000E-01  0.3460000E-01  0.4281000E-01
 0.5115000E-01  0.5964000E-01  0.6828000E-01  0.7709000E-01  0.8607000E-01  0.9524000E-01  0.1046100      0.1141700
 0.1272600      0.1407300
   RMSP  ARRAY
 0.7586000E-01  0.7662000E-01  0.7874000E-01  0.8091000E-01  0.8294000E-01  0.8531000E-01  0.8802000E-01  0.9108000E-01
 0.9447000E-01  0.9820000E-01  0.1022800      0.1067400      0.1114600      0.1165600      0.1220000      0.1277800
 0.1360200      0.1448700
   BESP  ARRAY
 0.1053300E-01  0.1004500E-01  0.8724000E-02  0.7420000E-02  0.6316000E-02  0.5354000E-02  0.4532000E-02  0.3831000E-02
 0.3235000E-02  0.2728000E-02  0.2299000E-02  0.1936000E-02  0.1629000E-02  0.1370000E-02  0.1151000E-02  0.9790000E-03
 0.8250000E-03  0.7240000E-03

 BLDAT  AANDK  ERSOR  STRFN  SLCRD  INTVL  SURVL
   1      1      2      2      2      2      3
```



(a) Full blade.

Figure 9. - Velocity distribution for mixed-flow pump impeller.



(b) Splitter vane.

Figure 9. - Concluded.

ence 6. There is good agreement over most of the blade. Minor discrepancies are probably due to slight differences in the boundary conditions (weight flow split and downstream flow angle). The heights of the peaks near the leading edges are uncertain because the radii are small compared to the mesh spacing.

Execution time was 2 minutes for this example. It required only one outer iteration, since flow was incompressible.

DESCRIPTION OF INPUT AND OUTPUT

The computer program requires as input a geometrical description in m, θ coordinates of the tandem blade segments, a description in m, r coordinates of the stream channel through the blades, appropriate gas constants, and operating conditions such as inlet temperature and density, inlet and outlet flow angles, weight flow, and rotational speed. An estimate of the portion of the weight flow which passes between the tandem blades must also be given. Output obtained from the program includes velocity magnitude and direction at all interior mesh points in the blade-to-blade passage, blade-surface velocities, stream-function values throughout the blade-to-blade region of solution, and streamline locations.

Input

Figure 10 shows the input variables as they are punched on the data cards. The first input data card is for a title, which will serve for problem identification. The remaining cards are for input variables. There are two types of variables, geometric and nongeometric. The geometric input variables are shown in figures 11 to 13. Further explanation of the input variables is given in the section Instructions for Preparing Input.

The input variables are as follows:

GAM	specific-heat ratio, γ
AR	gas constant, joule/(kg)($^{\circ}$K)
TIP	inlet total temperature, T'_{in}, $^{\circ}$K
RHOIP	inlet total density, ρ'_{in}, kg/meter^3
WTFL	mass flow per blade for stream channel, kg/sec
WTFLSP	portion of stream-channel mass flow per blade which flows across the boundary JKL between the front and rear blades, kg/sec; see fig. 11

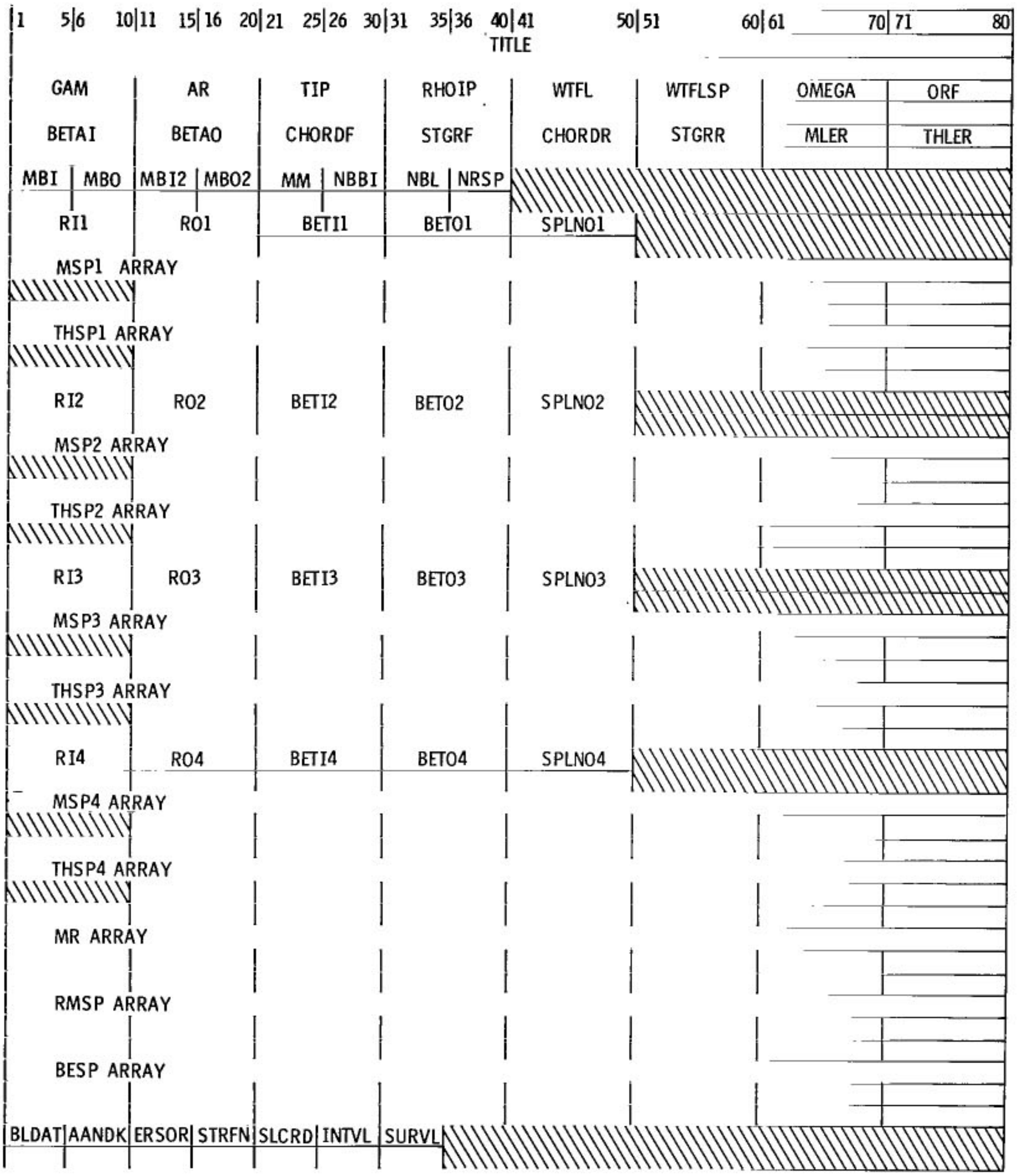
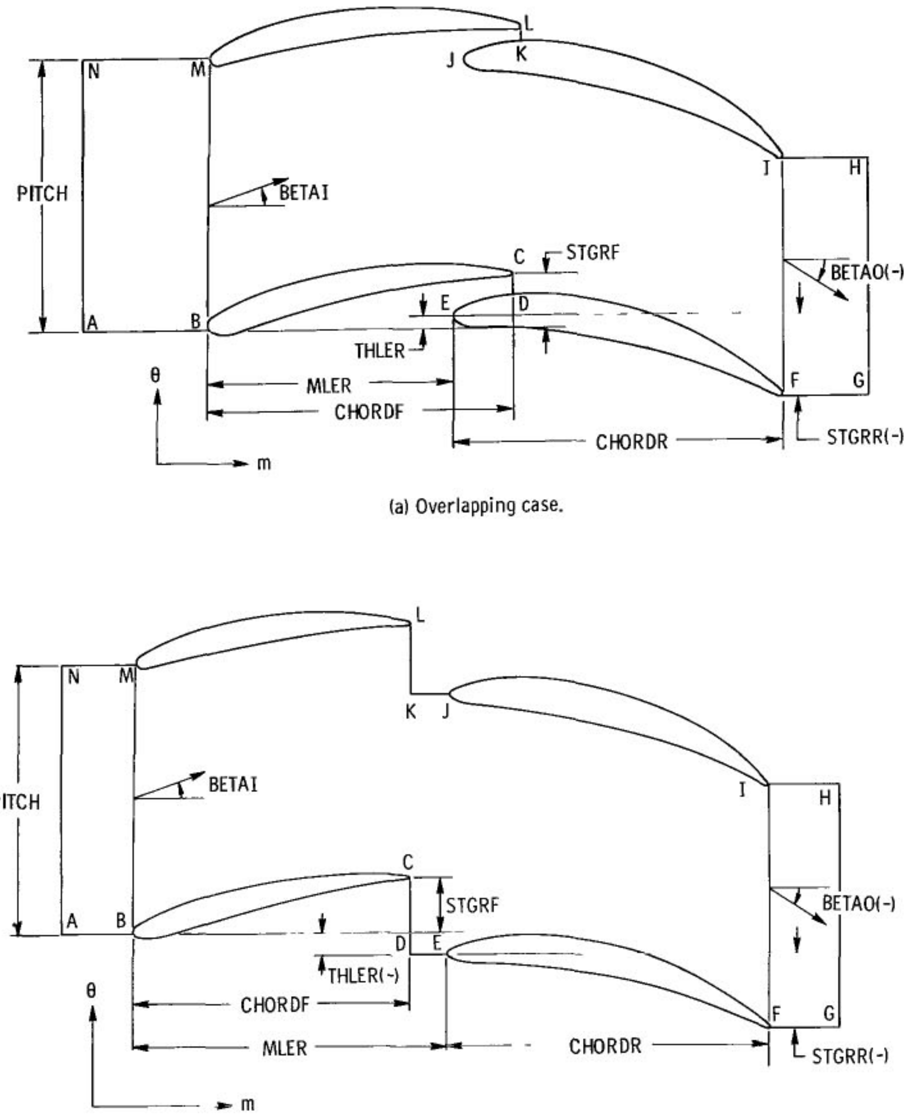


Figure 10. - Input form.



(a) Overlapping case.

(b) Nonoverlapping case.

Figure 11. - Geometric input variables for blade-to-blade flow region.

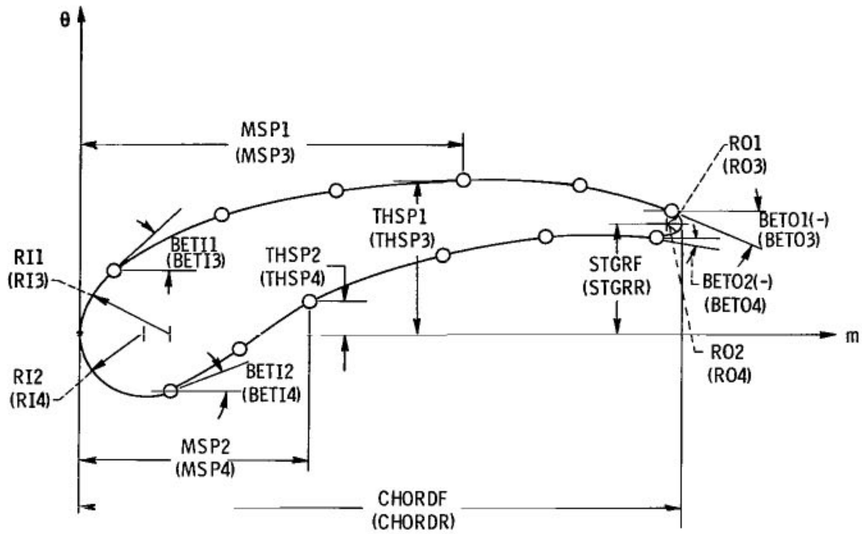


Figure 12. - Geometric input variables on blade. Angles BETI1, 2, 3, 4 and BETO1, 2, 3, 4 must be given as true angle β, not as angles measured in m, θ plane. Either use $\tan\beta = r\,d\theta/dm$ to obtain β, or measure the true angle.

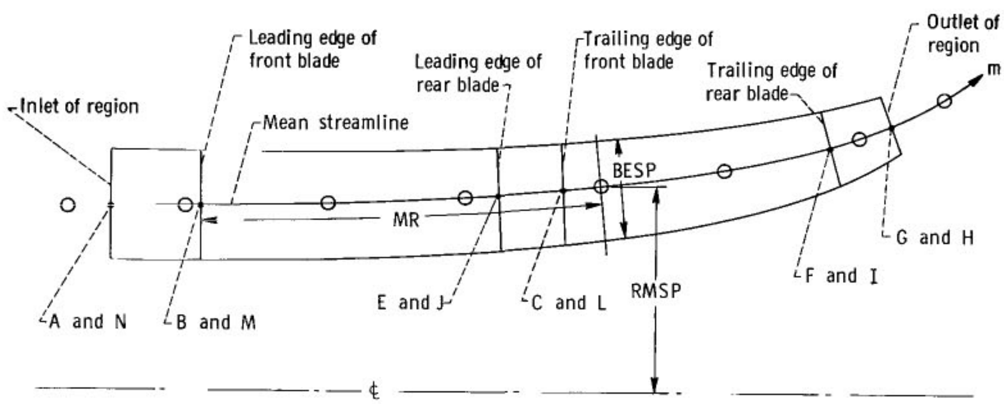


Figure 13. - Geometric input variables describing stream-channel in meridional plane.

OMEGA rotational speed, ω, rad/sec (Note that ω is negative if rotation is in the opposite direction of that shown in fig. 1.)

ORF value of overrelaxation factor Ω to be used in equation (A8) (If ORF = 0, the program calculates an estimated value for the overrelaxation factor; see p. 25 and appendix A for discussion.)

BETAI inlet flow angle β_{le} along BM with respect to m-direction, deg; see fig. 11

BETAO outlet flow angle β_{te} along FI with respect to m-direction, deg; see fig. 11

CHORDF overall length of front blade in m-direction, meters; see fig. 11

STGRF	angular θ-coordinate for center of trailing-edge circle of front blade with respect to center of leading-edge circle of front blade, rad; see fig. 11
CHORDR	overall length of rear blade in m-direction, meters; see fig. 11
STGRR	angular θ-coordinate for center of trailing-edge circle of rear blade with respect to center of leading-edge circle of rear blade, rad; see fig. 11
MLER	m-coordinate of leading edge of rear blade with respect to leading edge of front blade, meters; see fig. 11
THLER	angular θ-coordinate of leading edge of rear blade with respect to leading edge of front blade, rad; see fig. 11
MBI	number of vertical mesh lines from AN to BM inclusive; see fig. 11
MBO	number of vertical mesh lines from AN to CL inclusive; see fig. 11
MBI2	number of vertical mesh lines from AN to EJ inclusive; see fig. 11
MBO2	number of vertical mesh lines from AN to FI inclusive; see fig. 11
MM	total number of vertical mesh lines in m-direction from AN to GH, maximum of 100; see fig. 11
NBBI	number of mesh spaces in θ-direction between AB and MN, maximum of 50; see fig. 11
NBL	number of blades
NRSP	number of spline points for stream-channel radius (RMSP) and thickness (BESP) coordinates, maximum of 50; see fig. 13
RI1, RI2, RI3, RI4	leading-edge radii of the four blade surfaces, meters; see fig. 12
RO1, RO2, RO3, RO4	trailing-edge radii of the four blade surfaces, meters; see fig. 12
BETI1, BETI2, BETI3, BETI4	angles (with respect to m-direction) at tangent points of leading-edge radii with the four blade surfaces, deg; see fig. 12 (These must be true angles in degrees. If angles (i.e., $d\theta/dm$) are measured in the m, θ plane, BETI1, 2, 3, 4 can be obtained from the relation $\tan \beta = r\, d\theta/dm$.)
BETO1, BETO2, BETO3, BETO4	angles (with respect to m-direction) at tangent points of trailing-edge radii with the four blade surfaces, deg; see fig. 12 (These must also be true angles in degrees, like BETI1, 2, 3, 4.)

SPLNO1, SPLNO2, SPLNO3, SPLNO4 — number of blade spline points given for each surface as input, maximum of 50 (These include the first and last points (dummies) that are tangent to the leading- and trailing-edge radii (fig. 12).)

MSP1, MSP2, MSP3, MSP4 — arrays of m-coordinates of spline points on the four blade surfaces, measured from blade leading edges, meters; see fig. 12 (The first and last points in each of these arrays can be blank or have a dummy value, since these points are calculated by the program. If blanks are used, and the last point is on a new card, a blank card must be used.)

THSP1, THSP2, THSP3, THSP4 — arrays of θ-coordinates of spline points corresponding to MSP1, MSP2, etc., rad; see fig. 12 (Dummy values are also used in positions corresponding to those in MSP1, MSP2, etc.)

MR — array of m-coordinates of spline points for stream-channel radii and stream-channel thicknesses, meters; see fig. 13 (MR is measured from leading edge of front blade. These coordinates should cover the entire distance from AN to GH and may extend beyond these bounds. The total number of points is NRSP.)

RMSP — array of r-coordinates of spline points for stream-channel mean streamline, corresponding to the MR array, meters; see fig. 13

BESP — array of stream-channel normal thicknesses corresponding to the MR and RMSP arrays, meters; see fig. 13

The remaining variables, starting with BLDAT, are used to indicate what output is desired. A value of 0 for any of these variables will cause the output associated with that variable to be omitted. A value of 1 will cause the corresponding output to be printed for the final outer iteration only; a value of 2, for the first and final iterations; and a value of 3, for all outer iterations. Care should be used not to call for more output than is really useful. The following list gives the output associated with each of these variables:

BLDAT — all geometrical information which does not change from iteration to iteration; i.e., coordinates and first and second derivatives of all blade-surface spline points; blade coordinates and blade slopes where vertical mesh lines meet each blade surface; radii and stream-channel thicknesses corresponding to each vertical mesh line; m-coordinate, stream-channel radius and thickness, blade-surface angles and slopes where horizontal mesh lines intersect each blade; and ITV and IV arrays (internal variables describing the location of the blade surfaces with respect to the finite-difference grid)

AANDK coefficient array A, vector $\underline{k}$, and indexes of all adjacent points for each point in the finite-difference mesh (This information is needed only for debugging the program.)

ERSOR maximum change in stream function at any point for each iteration of SOR equation (eq. (A8))

STRFN value of stream function at each unknown mesh point in region

SLCRD streamline θ-coordinates at each vertical mesh line, and streamline plot

INTVL velocity and flow angle at each interior mesh point

SURVL m-coordinate, surface velocity, flow angle, distance along surface, and W/W_{cr} based on meridional velocity components where each vertical mesh line meets each blade surface; m-coordinate, surface velocity, flow angle, distance along surface, and W/W_{cr} based on tangential velocity components where each horizontal mesh line meets each blade surface; plot of blade-surface velocities against meridional streamline distance, m (It is suggested that SURVL = 3 be used. This will give surface velocities after each outer iteration, so that satisfactory velocities may be obtained even when final convergence is not reached.)

Instructions for Preparing Input

Units of measurement. - The International System of Units (ref. 8) is used throughout this report. However, the program does not use any constants which depend on the system of units being used. Therefore, any consistent set of units may be used in preparing input for the program. For example, if force, length, temperature, and time are chosen independently, mass units are obtained from force equals mass times acceleration. The gas constant R must then have the units of force times length divided by mass times temperature (energy per unit mass per deg). Density is mass per unit volume, and weight flow is mass per unit time. Output then gives velocity in the chosen units of length per unit time. Since any consistent set of units can be employed, the output is not labeled with any units.

Blade and stream-channel geometry. - The upper and lower surfaces of the front and rear tandem blades are each defined by specifying three things: leading- and trailing-edge radii, angles at which these radii are tangent to the blade surfaces, and m- and θ-coordinates of several points along each surface. These angles and coordinates are used to define a cubic spline curve fit (ref. 9) to the surface. The standard sign convention is used for angles, as indicated in figure 12.

A cubic spline curve is a piecewise cubic polynomial which expresses mathematically the shape taken by an idealized spline passing through the given points. Reference 9 describes a method for determining the equation of the spline curve. When this method is used, few points are required to specify most blade shapes accurately, usually no more than five or six, in addition to the two end points. As a guide, enough points should be specified so that a physical spline passing through these points would accurately follow the blade shape. This means that the spline points should be closer where there is large curvature and farther apart where there is small curvature.

The coordinates for either surface of a particular blade segment are given with respect to the leading edge of that segment, the leading edge being defined as the furthest point upstream on the blade segment.

The mean stream surface of revolution (as seen in the meridional plane, fig. 13) and the stream-channel thickness are also fitted with cubic spline curves. The m-coordinates for the mean stream surface are independent of the m-coordinates for blade surfaces.

Inlet and outlet flow angles. - The values of β_{le} and β_{te} are given as average values on BM and FI, respectively. If the flow is axial, these flow angles are the same as the flow angles at AN and GH. If flow is radial or mixed, and these angles are not known on BM and FI, β_{le} and β_{te} must be calculated by equation (B14).

Defining mesh. - A finite-difference mesh is used for the solution of equation (1). A typical mesh pattern (that used in example 1) is shown in figure 14. The mesh spacing and the extent of the upstream and downstream regions are determined by the values of MBI, MBO, MBI2, MBO2, and MM of the input (fig. 10). The mesh spacing must be chosen so that there are not more than 2000 unknown mesh points.

Values of MBI, MBO2, etc., should be determined so that the mesh which results has blocks which are approximately square. To achieve this, a value for NBBI is first chosen arbitrarily (15 to 20 is typical). NBBI is the number of mesh spaces spanning the blade pitch s, where $s = 2\pi/\mathrm{NBL}$. Dividing s by NBBI gives the mesh spacing HT in the θ-direction in radians. Multiplying HT by an average radius (RMSP) of the stream channel gives an average value for the actual mesh spacing in the θ-direction. CHORDF, CHORDR, and MLER should then be used with this tangential mesh spacing to calculate the approximate number of mesh spaces in the various regions along the meridional axis. This will give MBO, MBI2, and MBO2, once MBI is chosen. Generally, MBI is given a value of 10; MM, likewise, is usually given a value 10 more than MBO2.

Overrelaxation factor. - ORF is the relaxation factor used in each inner iteration in the solution of the simultaneous finite-difference equations (A7). ORF may be set equal to 0, or to some value between 1 and 2. ORF is usually given as 0 for the initial run of a given blade geometry and mesh spacing (MBI, NBBI, etc.). In this case the program uses extra time and calculates an optimum value for ORF. It does this by means of an iterative process, and on each iteration the current estimate of the optimum value for

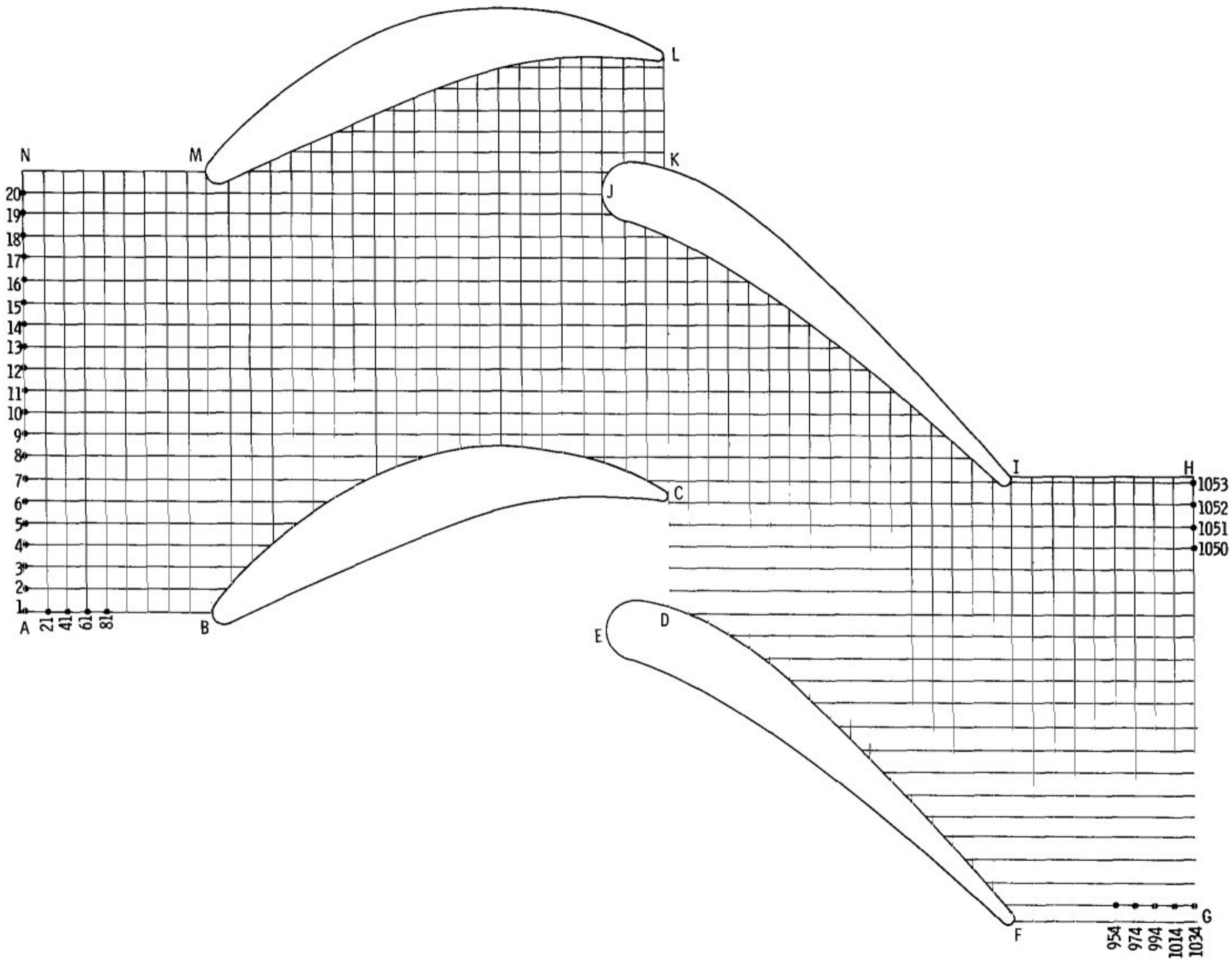


Figure 14. - Mesh used for axial-flow-turbine numerical example. Numbers are mesh point indexes (IP in program). There are 1053 unknown mesh points.

ORF is printed. The final estimate is the one used by the program for ORF. If the user does not change the mesh indexes MBI, MBO, MBI2, MBO2, MM, and NBBI between runs, even though blade geometry or other input does change, he may use this final estimate of ORF in the input, saving the time used in its computation. In all cases, if ORF is not 0, it should have a value greater than 1 and less than 2.

Actually, the value of ORF is not as critical as the user might think. It gets more critical as the optimum value gets close to 2. For any run of a given set of data, only small changes will occur in the rate of convergence in SOR as long as the difference 2.0 - ORF is within 10 percent of its optimum value. A further theoretical discussion of the overrelaxation factor is presented in reference 11 (p. 78).

Format for input data. - All the numbers on the card beginning with MBI and on the card beginning with BLDAT are integers (no decimal point) in a five-column field (see fig. 10). These must all be right adjusted. The input variables on all other data cards are real numbers (punch decimal point) in a ten-column field.

Incompressible flow. - While the program is written for compressible flow, it can be easily used for incompressible flow. To do so, specify GAM = 1.5, AR = 1000, and TIP = 10^6 as input. This results in a single outer iteration of the program to obtain the stream-function solution.

Straight infinite cascade. - The program is as easily applied to straight infinite cascades as to circular cascades. Since the radius and number of blades (NBL) for such a cascade would actually be infinite, an artificial convention must be adopted. The user should pick a value for NBL, for instance 20 or 30. Then, since the blade pitch equal to sr is known, an artificial radius can be computed from

$$r = \frac{NBL*(sr)}{2\pi}$$

This r should be used to compute the θ-coordinates required as input (THSP1, . . ., THSP4, STGR1, STGR2, THLE2) by dividing coordinates in the tangential direction by r.

Axial flow. - For a two-dimensional cascade with constant stream-channel thickness, constant values should be given for the RMSP and BESP arrays. Only two points are required for each of these arrays in this case. The two values of MR should be chosen so that they are further upstream and downstream than the boundaries AN and GH. The two values of RMSP and BESP should equal the constants r and b.

Output

Sample output is given in table III for the axial-flow turbine example. Since the complete output would be lengthy, only the first few lines of each section of output are reproduced herein. Most of the output is optional and is controlled by the final input card, as already described. In many instances output labels are simply internal variable names which are defined in the Main Dictionary.

Each section of the sample output in table III has been numbered to correspond to the following description:

(1) The first output is a listing of the input data. All items are labeled as on the input form (fig. 10).

(2) This is the output corresponding to BLDAT. (See the list of input variables and the Main Dictionary for variable name definitions.)

(3) The relative free-stream velocity W, the relative critical velocity W_{cr}, and the maximum value of the mass flow parameter ρW (corresponding to $W = W_{cr}$) are given at the leading edge of the front blade (BM) and the trailing edge of the rear blade (FI). The inlet (outlet) free-stream flow angle β_{in} (β_{out}) at boundary AN (GH) is given. These angles are based on the input angles BETAI (β_{le}) and BETAO (β_{te}).

(4) These are calculated program constants, including the pitch from blade to blade, the mesh spacing in all solution regions, the minimum and maximum values of IT in the solution region (ITMIN and ITMAX), and the value of the prerotation λ (eq. (B8)).

(5) This is the number of mesh points in the entire solution region at which the stream function is unknown.

(6) This is the boundary value (BV) of the stream function on each of the four blade surfaces.

(7) This is the output corresponding to AANDK.

(8) If the program calculates an optimum overrelaxation factor Ω (i.e., ORF = 0 in the input), the successive estimates to the optimum value of ORF are printed. The last printed value of the estimated optimum ORF is the value of Ω (ORF) used by the program.

(9) This is the output corresponding to ERSOR.

(10) This is the output corresponding to STRFN.

(11) This is the total execution time after obtaining the stream-function solution for each outer iteration.

(12) This is the output corresponding to SLCRD.

(13) This is the output corresponding to INTVL.

(14) This gives the maximum relative change in the density ρ for each outer iteration.

(15) This is the output corresponding to SURVL.

(16) This is the total execution time after all calculations are completed for an outer iteration.

TABLE III. - SAMPLE OUTPUT

```
    MODIFIED TANDEM AXIAL TURBINE ROTOR
     GAM            AR             TIP            RHOIP          WTFL           WTFLSP         OMEGA          ORF
  1.4000000      287.05300      288.15000      1.2250000      0.3152000E-01  0.1134700E-01  -0             0
     BETAI          BETAO          CHORDF         STGRF          CHORDR         STGRR          MLER           THLER
  48.000000      -47.000000     0.2847000E-01  0.2133300E-01  0.2515000E-01  -0.5459000E-01 0.2441000E-01  -0.3607000E-02
 MBI  MBO MBI2 MBO2  MM  NBBI  NBL NRSP
 10   32   29   49   58   20   76   2


   BLADE SURFACE 1  --  UPPER SURFACE - FRONT BLADE
     RI1            RO1            BETI1          BETO1          SPLNO1
  0.7620000E-03  0.3810000E-03  50.000000      -29.400000     7.0000000
     MSP1  ARRAY
 -0              0.2570000E-02  0.7650000E-02  0.1527000E-01  0.2035000E-01  0.2543000E-01  -0
     THSP1  ARRAY
 -0              0.9250000E-02  0.2118000E-01  0.2988000E-01  0.3020000E-01  0.2643000E-01  -0


   BLADE SURFACE 2  --  LOWER SURFACE - FRONT BLADE
     RI2            RO2            BETI2          BETO2          SPLNO2
  0.7620000E-03  0.3810000E-03  25.000000      -6.9000000     5.0000000
     MSP2  ARRAY
 -0              0.7650000E-02  0.2035000E-01  0.2543000E-01  -0
     THSP2  ARRAY
 -0              0.7140000E-02  0.2039000E-01  0.2094000E-01  -0


1  BLADE SURFACE 3  --  UPPER SURFACE - REAR BLADE
     RI3            RO3            BETI3          BETO3          SPLNO3
  0.1778000E-02  0.3810000E-03  -8.1000000     -48.800000     4.0000000
     MSP3  ARRAY
  0              0.6100000E-02  0.1626000E-01  0
     THSP3  ARRAY
  0              0.1640000E-02  -0.2463000E-01 0


   BLADE SURFACE 4  --  LOWER SURFACE - REAR BLADE
     RI4            RO4            BETI4          BETO4          SPLNO4
  0.1778000E-02  0.3810000E-03  -19.700000     -42.500000     4.0000000
     MSP4  ARRAY
  0              0.6100000E-02  0.1372000E-01  0
     THSP4  ARRAY
  0              -0.1200000E-01 -0.2745000E-01 0


     MR  ARRAY
 -1.0000000      1.0000000
     RMSP  ARRAY
  0.3238500      0.3238500
     BESP  ARRAY
  0.1000000E-01  0.1000000E-01


   BLDAT  AANDK  ERSOR  STRFN  SLCRD  INTVL  SURVL
     1      1      2      2      2      2      3
```

TABLE III. - Continued. SAMPLE OUTPUT

BLADE DATA AT INPUT SPLINE POINTS

BLADE SURFACE 1

M	THETA	DERIVATIVE	2ND DERIV.
0.17827F-03	0.15124E-02	3.67996	-448.310
0.25700E-02	0.92500E-02	2.88161	-219.276
0.76500E-02	0.21180E-01	1.83901	-191.199
0.15270F-01	0.29880E-01	0.47565	-166.638
0.20350E-01	0.30200E-01	-0.33906	-154.115
0.25430F-01	0.26430E-01	-1.15680	-167.828
0.28276E-01	0.22358E-01	-1.73991	-241.946

BLADE SURFACE 2

2

M	THETA	DERIVATIVE	2ND DERIV.
0.10840F-02	0.80541E-01	1.43989	-7.45650
0.76500E-02	0.89813E-01	1.38132	-10.3836
0.20350E-01	0.10306	0.43322	-138.924
0.25430F-01	0.10361	-0.18877	-105.953
0.28043E-01	0.10284	-0.37367	-35.5599

BLADE SURFACE 3

M	THETA	DERIVATIVE	2ND DERIV.
0.26439E-01	0.18284E-02	-0.43947	-206.721
0.30510E-01	-0.19670E-02	-1.49683	-312.681
0.40670F-01	-0.28237E-01	-3.17480	-17.6280
0.49466F-01	-0.57422E-01	-3.52722	-62.5065

BLADE SURFACE 4

M	THETA	DERIVATIVE	2ND DERIV.
0.25589E-01	0.73898E-01	-1.10561	-116.003
0.30510E-01	0.67066E-01	-1.66752	-112.352
0.38130F-01	0.51616E-01	-2.31958	-58.7926
0.48922E-01	0.23609E-01	-2.82949	-35.7093

3

	FREESTREAM VELOCITY	MAXIMUM VALUE FOR RHO*W	CRITICAL VELOCITY		BETA CORRECTED TO BOUNDARY
LEADING EDGE B-M	161.064	241.239	310.645	BOUNDARY A-N	48.0000
TRAILING EDGE F-I	157.135	241.239	310.645	BOUNDARY G-H	-47.0000

4

CALCULATED PROGRAM CONSTANTS

PITCH	HT	HM1	HM2	HM3
0.8267349E-01	0.4133675E-02	0.1284737E-02	0.1353333E-02	0.1240588E-02

ITMIN	ITMAX
-14	25

LAMBDA
38.762794

5 NUMBER OF INTERIOR MESH POINTS = 1053

6 SURFACE BOUNDARY VALUES

SURFACE	BV
1	0.
2	1.00000
3	-0.35999
4	0.64001

2 BLADE DATA AT INTERSECTIONS OF VERTICAL MESH LINES WITH BLADES

	BLADE SURFACE 1		BLADE SURFACE 2	
M	TV	DTDMV	TV	DTDMV
0	0	0.10000E 11	0.82673E-01	-0.10000E 11
0.12847E-02	0.53314E-02	3.24254	0.80830E-01	1.43838
0.25695E-02	0.92485E-02	2.88173	0.82671E-01	1.42832
0.38542E-02	0.12772E-01	2.60458	0.84500E-01	1.41752
0.51389E-02	0.15945E-01	2.33654	0.86313E-01	1.40599

STREAM SHEET COORDINATES AND THICKNESS TABLE

IM	M	R	SAL	B	DB/DM
1	-0.11563E-01	0.32385	-0	0.10000E-01	-0
2	-0.10278E-01	0.32385	-0	0.10000E-01	-0
3	-0.89932E-02	0.32385	-0	0.10000E-01	-0
4	-0.77084E-02	0.32385	-0	0.10000E-01	-0
5	-0.64237E-02	0.32385	-0	0.10000E-01	-0

IM	IV ARRAY	ITV ARRAY			
	BLADE SURFACE NO.	1	2	3	4
1	1	0	19	0000	0000
2	21	0	19	0000	0000
3	41	0	19	0000	0000
4	61	0	19	0000	0000
5	81	0	19	0000	0000

TABLE III. - Continued. SAMPLE OUTPUT

2

M COORDINATES OF INTERSECTIONS OF HORIZONTAL MESH LINES WITH BLADE

MH ARRAY - BLADE SURFACE 1

MH	RMH	BEH	BETAH	DTDMH
0	0.3238	0.1000E-01	90.000	0.1000E 11
0.9725E-03	0.3238	0.1000E-01	47.526	3.3728
0.2233E-02	0.3238	0.1000E-01	43.797	2.9608
0.3713E-02	0.3238	0.1000E-01	40.472	2.6347
0.5394E-02	0.3238	0.1000E-01	36.494	2.2844

THETA COORDINATES OF HORIZONTAL MESH LINES

IT	THETA
-14	-0.57871E-01
-13	-0.53738E-01
-12	-0.49604E-01
-11	-0.45470E-01
-10	-0.41337E-01

7

IT	IP	IP1	IP2	IP3	IP4	A(1)	A(2)	A(3)	A(4)	K
IM =	1	IT1 =	0							
0	1	20	2	0	21	0.	0.	0.	1.00000	0.05329
1	2	1	3	1	22	0.	0.	0.	1.00000	0.05329
2	3	2	4	2	23	0.	0.	0.	1.00000	0.05329
3	4	3	5	3	24	0.	0.	0.	1.00000	0.05329
4	5	4	6	4	25	0.	0.	0.	1.00000	0.05329
5	6	5	7	5	26	0.	0.	0.	1.00000	0.05329
6	7	6	8	6	27	0.	0.	0.	1.00000	0.05329
7	8	7	9	7	28	0.	0.	0.	1.00000	0.05329
8	9	8	10	8	29	0.	0.	0.	1.00000	0.05329
9	10	9	11	9	30	0.	0.	0.	1.00000	0.05329
10	11	10	12	10	31	0.	0.	0.	1.00000	0.05329
11	12	11	13	11	32	0.	0.	0.	1.00000	0.05329
12	13	12	14	12	33	0.	0.	0.	1.00000	0.05329
13	14	13	15	13	34	0.	0.	0.	1.00000	0.05329
14	15	14	16	14	35	0.	0.	0.	1.00000	0.05329
15	16	15	17	15	36	0.	0.	0.	1.00000	0.05329
16	17	16	18	16	37	0.	0.	0.	1.00000	0.05329
17	18	17	19	17	38	0.	0.	0.	1.00000	0.05329
18	19	18	20	18	39	0.	0.	0.	1.00000	0.05329
19	20	19	1	19	40	0.	0.	0.	1.00000	0.05329
IM =	2	IT1 =	0							
0	21	40	22	1	41	0.23972	0.23972	0.26028	0.26028	-0.23972
1	22	21	23	2	42	0.23972	0.23972	0.26028	0.26028	-0.
2	23	22	24	3	43	0.23972	0.23972	0.26028	0.26028	-0.
3	24	23	25	4	44	0.23972	0.23972	0.26028	0.26028	-0.
4	25	24	26	5	45	0.23972	0.23972	0.26028	0.26028	-0.
5	26	25	27	6	46	0.23972	0.23972	0.26028	0.26028	-0.
6	27	26	28	7	47	0.23972	0.23972	0.26028	0.26028	-0.
7	28	27	29	8	48	0.23972	0.23972	0.26028	0.26028	-0.
8	29	28	30	9	49	0.23972	0.23972	0.26028	0.26028	-0.
9	30	29	31	10	50	0.23972	0.23972	0.26028	0.26028	-0.
10	31	30	32	11	51	0.23972	0.23972	0.26028	0.26028	-0.
11	32	31	33	12	52	0.23972	0.23972	0.26028	0.26028	-0.
12	33	32	34	13	53	0.23972	0.23972	0.26028	0.26028	-0.

```
8  ESTIMATED OPTIMUM ORF = 2.000000
   ESTIMATED OPTIMUM ORF = 1.999756
   ESTIMATED OPTIMUM ORF = 1.999655
   ESTIMATED OPTIMUM ORF = 1.999614
   ESTIMATED OPTIMUM ORF = 1.999614
```

```
9  ERROR = 1.85355929
   ERROR = 1.85807033
   ERROR = 1.56815425
   ERROR = 1.46978973
   ERROR = 1.28075877
```

```
10          STREAM FUNCTION VALUES
   IM =    1        IT1 =     0
      0.55114593  0.59970274  0.64908738  0.69933973  0.75039311  0.80210201  0.85427140  0.90668324  0.95911737  1.01136483
      1.06323957  1.11458504  1.16528240  1.21525803  1.26448990  1.31301716  1.36094677  1.40845889  1.45579982  1.50326271
   IM =    2        IT1 =     0
      0.49785382  0.54641053  0.59579523  0.64604745  0.69710085  0.74880978  0.80097911  0.85339101  0.90582513  0.95807273
      1.00994742  1.06129277  1.11199026  1.16196582  1.21119776  1.25972487  1.30765460  1.35516658  1.40250759  1.44997048
   IM =    3        IT1 =     0
      0.44394214  0.49235563  0.54170386  0.59201737  0.64320508  0.69509356  0.74746353  0.80007856  0.85270490  0.90512421
      0.95714255  1.00859727  1.05936310  1.10935885  1.15855476  1.20698299  1.25474706  1.30203211  1.34910329  1.39629060
```

```
11 TIME =  2.6014 MIN.
```

12 STREAMLINE COORDINATES

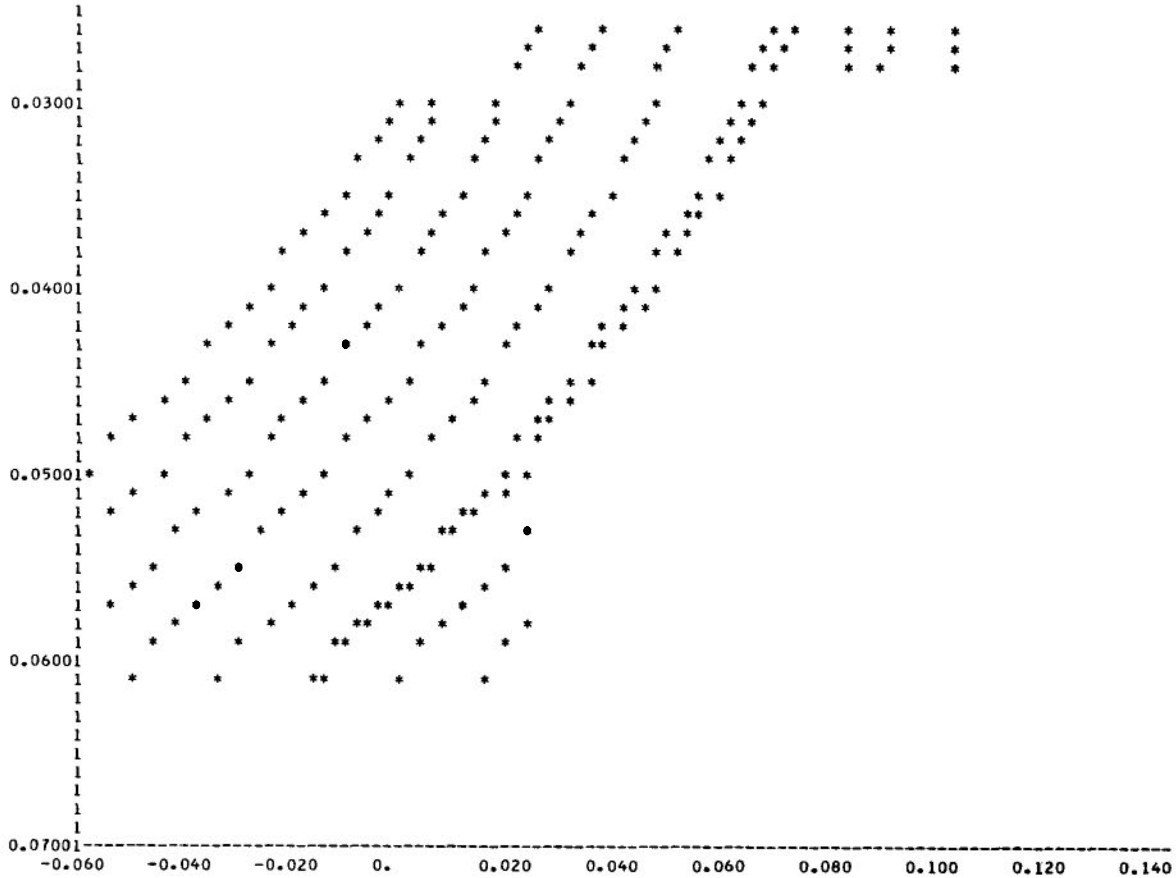
M COORDINATE	STREAM FN.	THETA	STREAM FN.	THETA	STREAM FN.	THETA
-0.1156263E-01	0.6000000	0.4158784E-02	0.8000000	0.2050116E-01	1.0000000	0.3630188E-01
	1.2000000	0.5246917E-01	1.4000000	0.6953478E-01	0.6400063	0.7512779E-02
	0.6000000	0.4158784E-02	0.6000000	0.4158784E-02		
-0.1027789E-01	0.6000000	0.8615924E-02	0.8000000	0.2472470E-01	1.0000000	0.4054115E-01
	1.2000000	0.5692582E-01	1.4000000	0.7418718E-01	0.6400063	0.1190771E-01
	0.6000000	0.8615924E-02	0.6000000	0.8615924E-02		
-0.8993158E-02	0.6000000	0.1305006E-01	0.8000000	0.2892955E-01	1.0000000	0.4477611E-01
	1.2000000	0.6140528E-01	1.4000000	0.7886315E-01	0.6400063	0.1627820E-01
	0.6000000	0.1305006E-01	0.6000000	0.1305006E-01		

TABLE III. - Continued. SAMPLE OUTPUT

12

STREAMLINE PLOTS

12



STREAMLINES ARE PLOTTED WITH THETA ACROSS THE PAGE AND M DOWN THE PAGE

TABLE III. - Concluded. SAMPLE OUTPUT

VELOCITIES AT INTERIOR MESH POINTS

13

IM	VELOCITY	ANGLE(DEG)	VELOCITY	ANGLE(DEG)	VELOCITY	ANGLE(DEG)	VELOCITY	ANGLE(DEG)	VELOCITY	ANGLE(DEG)
IM= 1	157.34	48.89	158.48	48.46	160.10	47.94	161.75	47.47	163.28	47.09
	164.52	46.81	165.38	46.66	165.80	46.62	165.79	46.71	165.37	46.90
	164.59	47.18	163.52	47.53	162.25	47.93	160.87	48.36	159.47	48.78
	158.18	49.15	157.11	49.42	156.39	49.56	156.10	49.53	156.45	49.29

IM	VELOCITY	ANGLE(DEG)	VELOCITY	ANGLE(DEG)	VELOCITY	ANGLE(DEG)	VELOCITY	ANGLE(DEG)	VELOCITY	ANGLE(DEG)
IM= 2	158.45	49.16	159.84	48.80	161.54	48.29	163.08	47.80	164.37	47.35
	165.29	47.00	165.79	46.76	165.84	46.63	165.48	46.63	164.74	46.75
	163.70	46.96	162.44	47.27	161.05	47.64	159.63	48.06	158.31	48.49
	157.20	48.91	156.43	49.25	156.11	49.49	156.31	49.58	157.15	49.46

14 ITERATION NO. 1 MAXIMUM RELATIVE CHANGE IN DENSITY = 0.5774

15

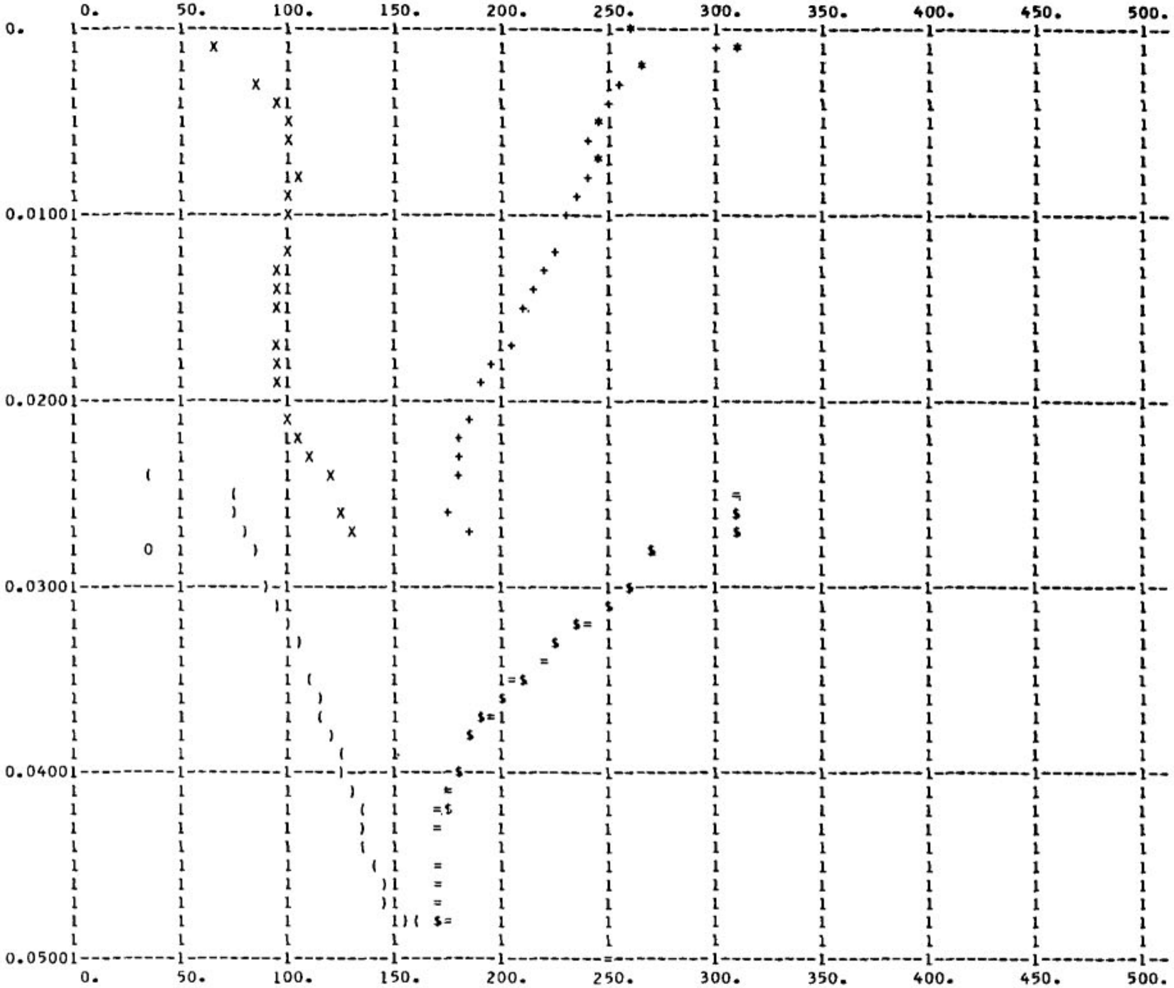
SURFACE VELOCITIES BASED ON MERIDIONAL COMPONENTS

	*	BLADE SURFACE 1				*	BLADE SURFACE 2				*
M	*	VELOCITY	ANGLE(DEG)	SURF. LENGTH	W/WCR	*	VELOCITY	ANGLE(DEG)	SURF. LENGTH	W/WCR	*
0	*	0	90.00	0	0	*	0	-90.00	0	0	*
0.1285E-02	*	297.83	46.40	0.2152E-02	0.9588	*	62.663	24.98	0.1417E-02	0.2017	*
0.2569E-02	*	255.80	43.02	0.3958E-02	0.8234	*	85.707	24.82	0.2833E-02	0.2759	*
0.3854E-02	*	248.70	40.15	0.5676E-02	0.8006	*	96.491	24.66	0.4248E-02	0.3106	*
0.5139E-02	*	245.70	37.11	0.7321E-02	0.7909	*	100.27	24.48	0.5660E-02	0.3228	*

SURFACE VELOCITIES BASED ON TANGENTIAL COMPONENTS

	BLADE SURFACE 1		
M	VELOCITY	ANGLE(DEG)	W/WCR
0	261.28	90.00	0.8411
0.9225E-03	310.64	47.53	1.0000
0.2233E-02	265.58	43.80	0.8549
0.3713E-02	251.53	40.47	0.8097

BLADE SURFACE VELOCITIES



VELOCITY(W) VS. MERIDIONAL STREAMLINE DISTANCE(M) DOWN THE PAGE

\+ - BLADE SURFACE 1, BASED ON MERIDIONAL COMPONENT
* - BLADE SURFACE 1, BASED ON TANGENTIAL COMPONENT
X - BLADE SURFACE 2, BASED ON MERIDIONAL COMPONENT
O - BLADE SURFACE 2, BASED ON TANGENTIAL COMPONENT
$ - BLADE SURFACE 3, BASED ON MERIDIONAL COMPONENT
= - BLADE SURFACE 3, BASED ON TANGENTIAL COMPONENT
) - BLADE SURFACE 4, BASED ON MERIDIONAL COMPONENT
(- BLADE SURFACE 4, BASED ON TANGENTIAL COMPONENT

15

16 TIME = 2.9211 MIN.

ERROR CONDITIONS

The error conditions are as follows:

(1) SPLINT USED FOR EXTRAPOLATION

EXTRAPOLATED VALUE = X.XXX

SPLINT is normally used for interpolation, but may be used for extrapolation in some cases. When this occurs, the above message is printed, as well as the input and output of SPLINT. Calculations proceed normally after this printout.

(2) BLCD CALL NO. XX

M-COORDINATE IS NOT WITHIN BLADE

This message is printed by subroutine BLCD if the M-coordinate given this subroutine as input is not within the bounds of the blade surface for which BLCD is called. The value of m and the blade-surface number are also printed when this happens. This condition may be caused by an error in the integer input items for the program.

The location of the error in the main program is given by means of BLCD CALL NO. XX, which corresponds to locations noted by comment cards at each MHORIZ, ROOT, and BLCD call in the program.

(3) ROOT CALL NO. XX

ROOT HAS FAILED TO CONVERGE IN 1000 ITERATIONS

This message is printed by subroutine ROOT if a root cannot be located. The input to ROOT is also printed. The user should thoroughly check the input to the main program.

The location of the error in the main program is given by means of ROOT CALL NO. XX, which corresponds to locations noted by comment cards at each MHORIZ and ROOT call in the program.

(4) DENSTY CALL NO. XX

NER(1) = XX

RHO*W IS X.XXXX TIMES THE MAXIMUM VALUE FOR RHO*W

This message is printed if the value of ρW at some mesh point is so large that there is no solution for the values of ρ and W. This indicates a locally supersonic condition, which can be eliminated by decreasing WTFL in the input.

If RHO*W is too large, TANDEM still attempts to calculate a solution. This often permits an approximate solution to be obtained which is valid at all the subsonic points in the region. In other cases, the value of W is reduced at some of the points in question during later iterations, resulting in a valid final solution for these points. The program counts the number of times supersonic flow has been located at any point during a given run (NER(1)). When NER(1) = 50, the program is stopped.

The location of the error in the main program is given by means of DENSTY CALL NO. XX, which corresponds to locations noted by comment cards at each DENSTY call in the program.

(5) MM, NBBI, NRSP, OR SOME SPLNO IS TOO LARGE

If this is printed, reduce the appropriate inputs to their allotted maximum values.

(6) WTFL IS TOO LARGE AT BLADE LEADING EDGE

This is printed if WTFL is greater than the choking mass flow for the boundary BM. If this message is printed, WTFL is cut in half by the program and calculations proceed as usual for one outer iteration.

(7) ONE OF THE MH ARRAYS IS TOO LARGE

This is printed if there are more than 100 intersections of horizontal mesh lines with any blade surface. In this case NBBI should be reduced.

(8) THE NUMBER OF INTERIOR MESH POINTS EXCEEDS 2000

This is printed if there are more than the allowable number of finite-difference grid points. Either MM or NBBI must be reduced.

(9) SEARCH CANNOT FIND M IN THE MH ARRAY

If this is printed, the value of m and the blade-surface number are also printed. The user should thoroughly check the input to the main program.

PROGRAM PROCEDURE

The program is segmented into seven main parts, the subroutines INPUT, PRECAL, COEF, SOR, SLAX, TANG, and VELOCY called by the main program TANDEM. In addition, there are several other subroutines. All the subroutines and their relation are shown in figure 15. All information which must be transmitted between the seven main subroutines is placed in COMMON.

Most of the subroutines in TANDEM use the same set of variables. These variables are all defined in the section Main Dictionary (p. 50). All subroutines using these variables are described prior to the main dictionary. The remaining subroutines are described after the main dictionary, and variables are defined with each subroutine.

The program can handle as many as 2000 mesh points on the IBM 2-7094-7044 direct-coupled system with a 32 768-word core. For 2000 mesh points to be handled an overlay arrangement is used, as shown in figure 16. All subroutines not shown are in the main link. The total program storage requirement is $74513_{(8)}$ of which $46770_{(8)}$ is in COMMON blocks which are stored in the main link. The system storage requirement for our computer is $2764_{(8)}$ and unused storage is $300_{(8)}$. If there is a storage problem on the user's computer, the maximum number of mesh points should be reduced. The following program changes are required to change the maximum number of mesh points:

(1) Change the dimension of A, U, K, and RHO in the COMMON/AUKRHO/statement. This statement occurs in most subroutines.

(2) In subroutine INPUT, change the number of values of U, K, and RHO to be ini-

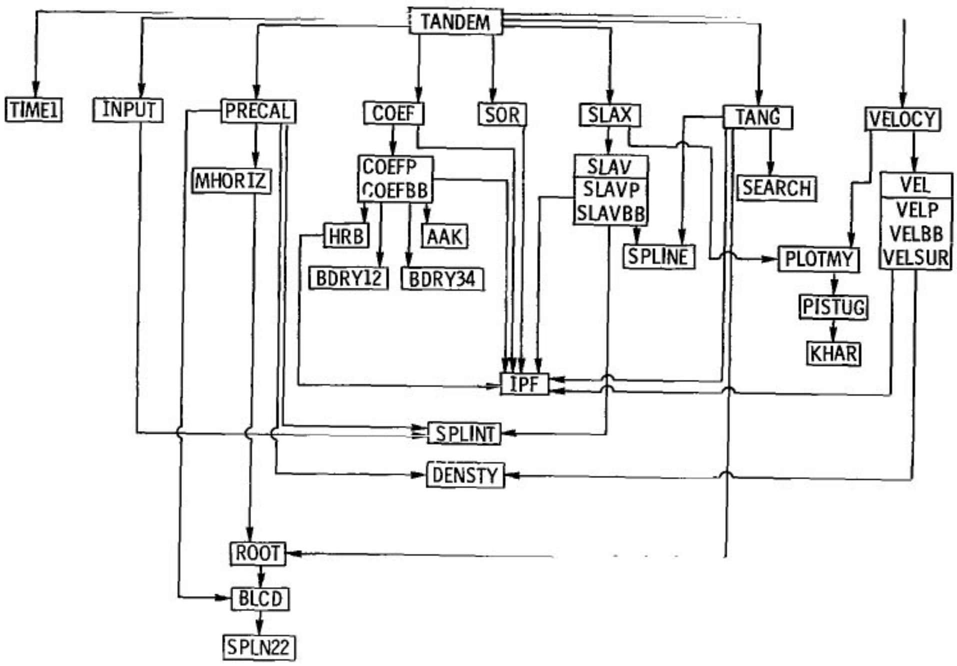


Figure 15. - Calling relation of subroutines.

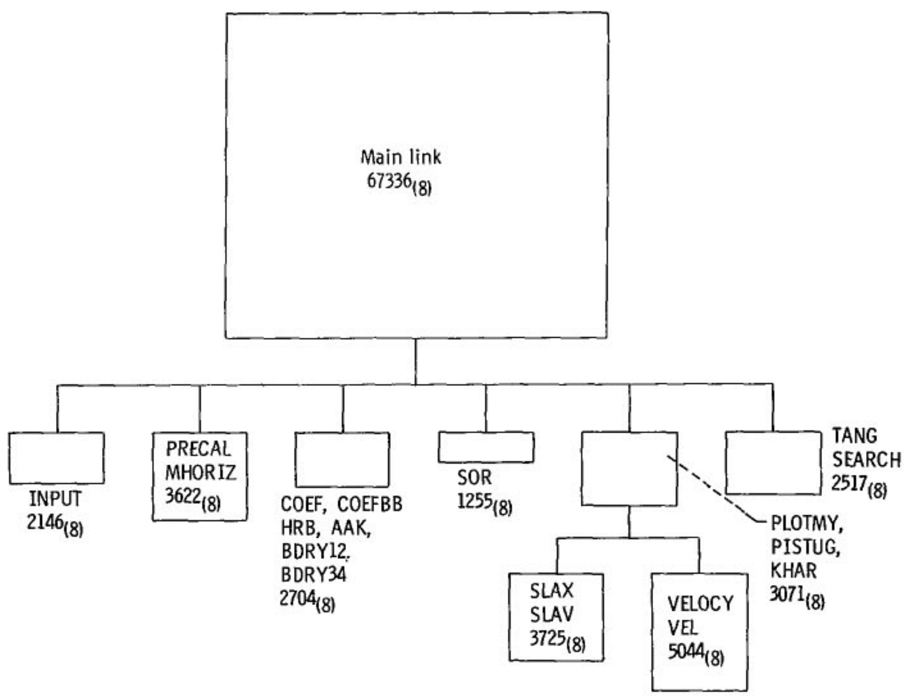


Figure 16. - Arrangement for overlay, showing octal storage requirements.

tialized (the bound on the DO loop near statement 60).

(3) In subroutine PRECAL, change statement 340 and format statement 1150 to reflect the maximum allowable number of mesh points. Statement 340 will cause the program to stop if there are too many mesh points.

(4) Change the dimensions of W, RWM, and BETA in SLAX, SLAV, TANG, VELOCY, and VEL.

(5) If the number of mesh points is reduced to below 1600, the equivalence statements in SLAX, SLAV, TANG, VELOCY, and VEL must be changed.

The first segment of the program is INPUT. This subroutine reads all input data cards, calculates constants, and initializes arrays. The next subroutine is PRECAL, which calculates all quantities which remain constant for a single problem. INPUT and PRECAL are each called once for a given problem. The remaining subroutines are called once for each outer iteration. The subroutine COEF calculates the entries of the matrix A and the vector $\underline{k}$ of equation (A7). These coefficients must be recalculated for each outer iteration. On the first outer iteration subroutine SOR estimates an optimum overrelaxation parameter Ω on the first call if it is not given as input. The same value of Ω is used for each outer iteration. SOR then finds the linear solution to equation (A7) with fixed coefficients by successive overrelaxation. Then subroutine SLAX calculates the streamline locations and ρW_m and plots the streamline locations if desired. Subroutine TANG calculates ρW_θ and then ρW and β throughout the region. Finally, the subroutine VELOCY calculates the density ρ and velocity W throughout the region and on the blade surfaces and plots the surface velocities.

Conventions Used in Program

For convenience, a number of conventions are used in naming variables and assigning subscripts. First, several pairs of variables are spelled the same except for one letter, which is U in one case and L in the other. The U signifies an upper surface BC, DF, EF, or JK, and L signifies a lower surface ML or JI. Another practice is to use the letters I and O in a similar manner, where I refers to the inlet or region ABMN, and O refers to the outlet or region FGHI. Similarly, the letter T refers to θ, and M refers to m. Finally, V is used to refer to vertical mesh lines, and H refers to horizontal mesh lines. For example, DTDMH is an array of the values of $d\theta/dm$ at the intersections of horizontal mesh lines with the blade.

The variable IP is used to number all the mesh points. It starts with IP = 1 at A and is incremented up the vertical mesh lines one by one to the right, ending with IP = NIP at the last mesh point near H. The mesh spacing in the m-direction is labeled HM1, HM2, or HM3, and the spacing in the θ-direction is HT. The subscript IM denotes the number of a vertical line, from IM = 1 at AN to IM = MM at GH. IT denotes the number of a horizontal mesh line. IT is zero along AB, increases to ITMAX at the highest mesh line in the region and decreases to ITMIN for the lowest mesh line in the region.

Labeled COMMON Blocks

For convenience, most variables which are used in more than one subroutine are placed in labeled COMMON blocks. A brief description of each labeled block is given. The same variable names are used in different subroutines for every variable in a COMMON block. The only exception is when EQUIVALENCE is used for variables in /AUKRHO/. The labeled COMMON blocks are as follows:

/INP/ is used for input variables, with the exception of those in /GEOMIN/.

/GEOMIN/ is used for certain geometry input variables which are needed only in BLCD.

/CALCON/ is used for calculated constants which are initially calculated in INPUT or PRECAL and do not change after this.

/AUKRHO/ is used for the arrays A, U, K, and RHO (see section Main Dictionary) or the variables which are made equivalent to some of these.

/BLCDCM/ is used for internal variables for BLCD. /BLCDCM/ is needed only to save certain values when overlay is used.

/HRBAAK/ is used for variables calculated by HRB to be used in AAK.

/RHOS/ is used to store values of ρ on blade surfaces.

/SLA/ is used for streamline θ-coordinates.

/BOX/ is used for internal variables for the spline subroutines in order to reduce storage requirements.

Subroutine INPUT

Reading and printing of input. - The program first reads all input cards for a particular problem. A description of the input required is given in the section Instructions for Preparing Input. All the input data are printed as the first output.

Calculation of constants and initialization. - After all input is read, many of the simpler constants used throughout the program are calculated. Finally, all density arrays are initialized to ρ'_{in}(RHOIP).

Subroutine PRECAL

Calculation of constants. - The prerotation λ and the average relative velocity W_{le} at the blade leading edge are calculated first by an iterative process (eqs. (B7) to (B9)). During this calculation the input weight flow (WTFL) is checked to see if it is larger than the upstream choked flow value. If so, it is cut in half and the computation of λ and W_{le} is repeated. Maximum values of the mass flow parameter ρW (eqs. (B10) to (B12)) and the critical velocity W_{cr} are then calculated at the leading and trailing edges of the blade row. The flow angles β_{in} and β_{out} are also computed at the upstream and downstream boundaries (appendix B) from the values, β_{le} and β_{te}, given at the leading and trailing edges.

Calculation of vertical mesh line arrays. - BLCD is called for the four blade surfaces obtaining θ-coordinates (TV) and slopes (DTDMV) where the vertical grid lines meet the blades. By using the TV array, the integer arrays (ITV and IV) are calculated. Finally, by using DTDMV, the blade-surface angles (BETAV) are calculated.

Calculation of horizontal mesh line arrays. - MHORIZ is called once for each of the four blade surfaces to obtain the m-coordinates (MH) and slopes (DTDMH) where horizontal grid lines meet the blade surfaces. Then by using cubic spline interpolation (SPLINT) and the MH array, RMH and BEH are calculated. Finally, the blade-surface angles BETAH are calculated by using DTDMH.

Subroutine COEF

Subroutine COEF controls the calculation of the finite-difference coefficients of $\underline{u}$ in equations (A2) to (A6) (elements of the matrix A in eq. (A7)). At the same time, it computes the constants of the finite-difference equations (components of $\underline{k}$ in eq. (A7)).

Calculating coefficients and constants throughout region. - COEF progresses from left to right through the blades. COEFP and COEFBB are called along each vertical mesh line for the calculation of the coefficients and constants. COEFP is called in the periodic regions upstream and downstream of the blades, and between front and rear blades for the nonoverlapping case. COEFBB is called in the regions between upper and lower blade surfaces.

Corrections to coefficients and constants. - At certain points in the solution region, corrections must be made to the coefficients and constants calculated by COEFP and COEFBB. This is done at the end of COEF if points B, J, C, E, or F (see fig. 4) on the blade surfaces coincide with mesh points in the solution region. Corrections are also made along line KL and the line to the right of CD. The periodic boundary condition equations are applied herein (see eqs. (A5) and (A6) and the explanation following them).

Subroutine COEFBB

Subroutine COEFBB computes the coefficients a_{ij} and constants k_i along a vertical mesh line from blade to blade. It has a second entry point (COEFP) with completely separate code, which computes coefficients and constants in periodic regions. Both COEFP and COEFBB proceed up a vertical mesh line, one point at a time.

In both the periodic and the blade-to-blade cases, HRB is called initially to compute the values of h, r, and b required in equation (A2). These values are then altered for special cases. In COEFBB they are altered along lines CD and KL, and in COEFP along periodic boundaries. COEFBB also calls BDRY12 and BDRY34 to obtain special values of h, r, and b when mesh points are within one mesh space of a blade boundary. Finally, both COEFP and COEFBB call AAK to compute A and k from equations (A2).

Subroutine HRB

Subroutine HRB calculates values of h, r, and b for use in equations (A2). Each time it is called, it computes these values for a single mesh point.

Subroutine AAK

Subroutine AAK is called by COEFBB and computes the coefficients a_{ij} and the constants k_i of equation (A2) at a single point.

Subroutine BDRY12

Subroutine BDRY12 is called by COEFBB. It alters the values of h and r calculated by HRB for point 1 or 2 (see fig. 17) if either of these points lies on a blade surface. It also defines the constants KAK and KA used to alter A and k in COEFBB or COEF.

Subroutine BDRY34

Subroutine BDRY34 is called by COEFBB. It alters the values of h, r, and b calculated by HRB for point 3 or 4 (see fig. 17) if either of these points lies on a blade surface. It also defines the constants KAK and KA used to alter A and k in COEFBB or COEF.

Subroutine SOR

This subroutine solves the finite-difference matrix equation (A7) by the method of successive overrelaxation (ref. 10). The same section of code is used both for calculating the optimum overrelaxation factor Ω and to solve equation (A7). If a value of ORF greater than 1 and less than 2 is given as input, it is used for the overrelaxation factor. Otherwise a value is estimated by the program.

In equation (A8), the subscript i denotes the index of an unknown mesh point. In the program, i is replaced by IP. The subscript j in equation (A8) denotes the index of neighboring unknown mesh points. For each i, there are only four values of j for which a_{ij} is nonzero, which are the negative values of the coefficients A(IP,1), A(IP, 2), A(IP, 3), and A(IP, 4). The value of j is determined by the index of the proper neighboring point. These indexes are named IP1, IP2, IP3, and IP4. These indexes are defined so that u^m_{IP1} has the coefficient A(IP, 1); the other indexes are defined similarly.

Estimation of optimum overrelaxation factor. - If ORF = 0 as input, the optimum value for the overrelaxation factor Ω is estimated on the first outer iteration by using equations (B3) and (B1) of reference 11. Equation (A8) is used to calculate $\underline{u}^{m+1}$ from $\underline{u}^m$ for equation (B3) of reference 11, with $\Omega = 1$, and $\underline{k} = \underline{0}$. Equation (A8) becomes

$$u_i^{m+1} = -\sum_{j=1}^{i-1} a_{ij}u_j^{m+1} - \sum_{j=i+1}^{n} a_{ij}u_j^m \tag{6}$$

To start, $u_i^0 = 1$ for all i. The maximum value of the ratio u_i^{m+1}/u_i^m is calculated for a given m and is given the name LMAX. After convergence, the optimum value of the overrelaxation factor Ω can be calculated by $\Omega = 2/\left(1 + \sqrt{1 - \text{LMAX}}\right)$. This procedure is explained in appendix B of reference 11.

Solution of matrix equation by subroutine SOR. - With a value of Ω either as input or estimated by the program, equation (A8) can be used iteratively to calculate a sequence $\left\{\underline{u}^m\right\}$ that will converge to a solution to equation (A7). During each iteration, the maximum change of the stream function is calculated. When this maximum change is reduced below 10^{-6}, the iteration is stopped, and the current estimate of the stream function is accepted as the solution.

Subroutine SLAX

Subroutine SLAX, by calling subroutines SLAVP and SLAVBB (entry points of SLAV), computes the meridional mass flow component ρW_m at all points on vertical mesh lines.

Subroutine SLAX also calculates and plots streamline locations.

Calculating ρW_m throughout region. - Subroutine SLAX progresses from left to right through the blades, calling SLAVP and SLAVBB along each vertical mesh line. SLAVP is called in the periodic regions upstream and downstream of the blade and between blades for the nonoverlapping case. SLAVBB is called in the blade-to-blade regions.

Plotting streamlines. - When subroutine SLAX reaches the right end of the region, all information is available from SLAVP and SLAVBB for the streamline plot. The plotting printout is done by PLOTMY, which, with the necessary further subroutines PISTUG and KHAR, is described completely in reference 12.

Subroutine SLAV

Subroutine SLAV has two entry points, SLAVP and SLAVBB. SLAVP is called in periodic regions, SLAVBB from blade to blade. Both entry points make use of a common section of code at the end of SLAV.

Calculation of $\partial u/\partial\theta$ and streamline locations. - SLAVP and SLAVBB compute $\partial u/\partial\theta$ along each vertical mesh line. The derivative $\partial u/\partial\theta$ is estimated at each mesh point from a cubic spline curve (SPLINE) of the stream function u.

SLAVP and SLAVBB also calculate values of θ corresponding to given values of the stream function. These values are printed out and are also used for the streamline plot. The stream function is a one-to-one function of distance in the θ-direction along most vertical mesh lines. Therefore, cubic spline interpolation (SPLINT) can be used to obtain θ as a function of u.

Calculation of ρW_m. - SLAVP and SLAVBB use the derivatives $\partial u/\partial\theta$ to calculate ρW_m at each mesh point. The equation $\partial u/\partial\theta = br\rho W_m/w$ (eq. (3)) is used. Values of ρW_m are stored in RWM for interior mesh points, and in WMB where the blade surfaces are intersected by vertical mesh lines.

Calculation of mass flow parameter ρW on blade surfaces. - Where each vertical mesh line meets a blade surface, ρW is calculated from ρW_m by the equation

$$\rho W = \frac{\rho W_m}{\cos\beta} = \rho W_m \sqrt{1 + \left(r\,\frac{d\theta}{dm}\right)^2}$$

Subroutine TANG

Subroutine TANG calculates the tangential mass flow component ρW_θ at all points on horizontal mesh lines. This process is complicated by the fact that the horizontal mesh lines are shifted in crossing the boundary KL.

Location of points on horizontal mesh lines. - Subroutine TANG begins at the bottom line of the region and proceeds upward to the top of the region, moving from left to right along each horizontal mesh line. On a given mesh line, the first point in the region is located by comparing IT for that mesh line with ITV for each of the four blade surfaces of successive points along the line. After an initial point in the region is located, TANG moves to the right along the line until it encounters the downstream boundary GH or a blade surface. Once again, TANG locates a blade boundary by comparing IT with ITV of the blade surfaces. ROOT is called to calculate mesh spacing at the end points when they are located on one of the blades. TANG stores the meridional distance and stream-function value of each point located along a line into the arrays SPM and USP.

Calculation of ρW_θ. - When a horizontal mesh line exits from the solution region, subroutine SPLINE is called with SPM and USP to calculate $\partial u/\partial m$ at each point along the line. The product ρW_θ is then calculated from $\partial u/\partial m$ by using $\partial u/\partial m = -b\rho W_\theta/w$ (eq. (2)).

Calculation of mass flow parameter ρW and flow angles at interior points. - At each interior point, ρW is calculated by

$$\rho W = \sqrt{(\rho W_m)^2 + (\rho W_\theta)^2}$$

and the angle β is calculated by

$$\tan \beta = \frac{\rho W_\theta}{\rho W_m}$$

These values are stored in W and BETA for all interior points.

Calculation of mass flow parameter ρW on blade surfaces. - Where each horizontal mesh line meets a blade surface, ρW is calculated from ρW_θ by the equation

$$\rho W = \frac{\rho W_\theta}{\sin \beta} = \rho W_\theta \sqrt{1 + \frac{1}{\left(r \frac{d\theta}{dm}\right)^2}}$$

Subroutine SEARCH

Subroutine SEARCH is used by TANG in the calculation of the mass flow parameter ρW on blade surfaces. The distance (DIST) corresponds to some element in the MH array for a particular surface. SEARCH locates that element and returns its subscript to TANG. TANG then uses a corresponding element in the BEH array in calculating ρW.

Subroutine VELOCY

Subroutine VELOCY calculates densities ρ and velocities W from the mass flow parameter ρW at all points throughout the solution region and on the blade surfaces. It also plots the surface velocities.

Solving for densities and velocities throughout region. - VELOCY progresses from left to right through the blades, calling VELP and VELBB for each vertical mesh line. VELP is called in the periodic regions, and VELBB is called from blade to blade. When the right boundary of the solution region is reached, VELSUR is called once to compute the blade-surface velocities.

Plotting of velocities. - After VELOCY calls VELSUR, all information is available for the plot of surface velocities. The velocities are plotted by using different symbols for front and rear blades, upper and lower surfaces, and velocities based on both meridional and tangential components. Velocities based on meridional components are plotted if $|\beta| \leq 60^{\circ}$, and velocities based on tangential components are plotted if $|\beta| \geq 30^{\circ}$. Plotting is done by PLOTMY, which is described in reference 12.

Subroutine VEL

Subroutine VEL has three independent entry points, VELP, VELBB, and VELSUR. VELP and VELBB compute velocities in the periodic and blade-to-blade regions, and VELSUR computes velocities on the blade surfaces. None of these entry points share common code in VEL.

The maximum relative change in density ρ along a blade surface is calculated in VELBB and VELSUR and is called RELER. If RELER is less than 0.001, the outer iteration is considered to be converged, and the calculations are stopped on the following iteration.

Calculation of ρ and W. - Both VELP and VELBB proceed from left to right through a region, and upward at each vertical mesh line from boundary to boundary. VELSUR proceeds along the four blade surfaces one at a time. VELP, VELBB, and VELSUR calculate density and velocity from ρW by calling the DENSTY subroutine at each mesh point and boundary point. Along the blade surfaces, VELBB and VELSUR also calculate the ratio W/W_{cr}.

Printing of velocities. - VELP and VELBB print interior velocities and flow angles as they are calculated. Surface velocities, blade-surface angles, arc lengths, and ratios of velocity to critical velocity are printed at the end of VELSUR.

Subroutine BLCD

Subroutine BLCD calculates the θ-coordinate and $d\theta/dm$ of a blade surface for any given value of m. There are four entry points to BLCD corresponding to the four blade surfaces.

The first time that BLCD is called for a particular blade surface, the coordinates of the first and last spline points are calculated. These points are tangent to the leading- and trailing-edge radii, respectively. The parameters defining the spline curve are also calculated at this time.

Each blade surface is defined by the leading- and trailing-edge radii and by a cubic spline curve, which is a piecewise cubic polynomial. The procedure is to scan the spline points to determine which interval the m-coordinate is in and then to calculate the θ-coordinate and derivative.

The arguments for the entry points of BLCD are defined so as to be called by ROOT to determine the m-coordinate of an intersection of a horizontal mesh line with the blade. Most of the information needed by BLCD is in labeled COMMON blocks. These variables are found in the main dictionary.

The input argument is

M meridional streamline coordinate, m

The output arguments are as follow:

THETA θ-coordinate of blade surface at m

DTDM $d\theta/dm$ of blade surface at m

INF used when $d\theta/dm$ is infinite; INF is normally 0, but set equal to 1 if $d\theta/dm$ is infinite

Function IPF

A mesh point in the solution region can be numbered in one of two ways. The first is by coordinates of mesh line intersection, IM and IT. IM is the number of the vertical mesh line, beginning with 1 at the upstream boundary AN. IT is the number of the horizontal mesh line, beginning with 0 at the leading edge of the upper surface of the front blade. The second numbering system is by point count, using IP. IP increases up each succeeding vertical mesh line from left to right through the solution region. IPF returns the value of IP corresponding to given coordinates, IM and IT.

Main Dictionary

The Main Dictionary applies to all the previously discussed subroutines.

A	array of coefficients of u (i.e., elements of a_{ij} of matrix A in eq. (A7))
A12, A34	a_{12}, a_{34} in eq. (A2)
AA	temporary variable in PRECAL and BLCD
AAA	array used for temporary storage
AANDK	see Input
AATEMP	temporary location for AANDK in SOR
ADD	logical variable in TANG, indicating need to add 1 to stream function at a mesh point prior to spline fit of stream function along a horizontal mesh line
ADDL	logical variable in TANG, indicating entrance into region where ADD applies
ANS	result of calls on ROOT in TANG and DENSTY in VEL
AR	see Input
AZ	a_0 in eq. (A2)
B	array containing stream-channel thickness b at the four points adjacent to a point for which AAK is called
B12, B34	b_{12}, b_{34} in eq. (A2)
BB	temporary variable in PRECAL and BLCD

BE	array of values of b at vertical mesh lines
BEH	array of values of b where horizontal mesh lines meet the four blade surfaces
BESP	see Input
BETA	array of values of β at interior mesh points
BETAH (BETAV)	array of values of β where horizontal (vertical) mesh lines meet the four blade surfaces
BETAI (BETAO)	see Input
BETI (BETO)	array of angles at tangent points of leading- (trailing-) edge radii with the four blade surfaces (see input BETI1, 2, 3, 4 and BETO1, 2, 3, 4)
BLDAT	see Input
BTAIN	free-stream angle β_{in} at upstream boundary AN based upon β_{le}, calculated by eq. (B14)
BTAOUT	free-stream angle β_{out} at downstream boundary GH based upon β_{te}, calculated by eq. (B14)
BV	array of stream-function boundary values on the four blade surfaces
BZ	stream-channel thickness b_0 at a point for which AAK is called
CDMBIT (CDMBOT)	temporary grid locations along meridional axis in INPUT
CHANGE	change in value of stream function at a particular point during an iteration of SOR
CHORD	array containing the meridional chord distances of each of the four blade surfaces (see input CHORDF and CHORDR)
CMM	temporary variable in BLCD
CP	specific heat at constant pressure, c_p
CPTIP	$2c_pT'_{in}$
DBDM	array of slopes at vertical mesh lines of spline curve for stream-channel thickness
DELTV	increment in θ-coordinate in VEL
DIST	meridional distance in SEARCH from a blade leading edge to where a horizontal mesh line meets a blade surface

DMLR	tolerance for mesh points near a boundary in m-direction (If a mesh point is closer than DMLR to a boundary, the point is considered to be on the boundary.)
DTDM	$d\theta/dm$ along a blade surface in BLCD
DTDMH (DTDMV)	array of $d\theta/dm$ where horizontal (vertical) mesh lines meet the four blade surfaces
DTLR	tolerance in θ-direction (see DMLR)
DUDM	array of derivatives of stream function du/dm along horizontal mesh lines in meridional direction
DUDT	array of derivatives of stream function $du/d\theta$ along vertical mesh lines in θ-direction
EM	array of second derivatives of spline curves for each blade surface, calculated by SPLN22 in BLCD
EMK, EMKM1	temporary variables for EM in BLCD
ERROR	maximum absolute value of change in u at any point for an overrelaxation (SOR) iteration
ERSOR	see Input
EXPON	$1/(\gamma - 1)$
FIRST	initial value of some index
GAM	see Input
H	array containing mesh spacing h between the point for which AAK is called and the four points adjacent to it
HM1	mesh spacing in m-direction from upstream boundary through front blade
HM2	mesh spacing in m-direction for overlapping portion of front and rear blades, or between blades for the nonoverlapping case
HM3	mesh spacing in m-direction through rear blade to downstream boundary
HT	mesh spacing in θ-direction from blade to blade
I	temporary integer variable in PRECAL, SLAX, SLAV, and SEARCH
I1, I2	temporary integers in SLAV

IEND	integer variable set equal to 1 when final convergence to a solution is reached in the outer iterations on a given set of data
IH	array containing current number of intersections of horizontal mesh lines with each of the four blade surfaces as intersections are located
IHS	integer variable in BDRY34 and TANG for counting intersections of horizontal mesh lines with blade surfaces
IM	index of mesh line in meridional direction (m-direction)
IM1 (IMT)	integer variable in TANG indicating the vertical mesh line index of the first (final) point in the region of a horizontal mesh line
IM2	IM1 + 1
IMS	array containing total number of intersections of horizontal mesh lines with each of the four blade surfaces
IMSL	temporary variable in PRECAL
IMSS	temporary variable in PRECAL, VELOCY, and VEL
IMTM1	IMT - 1
INF	variable in PRECAL indicating when an infinite slope is located at a blade leading- or trailing-edge in a call on BLCD
INIT	array used to indicate whether BLCD has been called previously on a given blade surface
INTU	temporary integer streamline value in SLAV
INTVL	see Input
IP	index of mesh point
IP1, IP2, IP3, IP4	value of IP at the four adjacent points to the mesh point under consideration
IPCD, IPKL	temporary IP along lines CD and KL in COEF
IPL (IPU)	value of IP where a vertical mesh line meets a lower (upper) surface or boundary
IPLM1 (IPUP1)	value of IP on a vertical mesh line adjacent to a lower (upper) surface in VEL
IS	integer variable in SEARCH for indicating where a horizontal mesh line intersects a blade surface

IT	index of mesh line in θ-direction
IT3, IT4	value of IT for the adjacent points (3 and 4) to mesh point under consideration
ITER	outer iteration counter
ITI	horizontal mesh line index in TANG one period below IT, IT-NBBI
ITMAX (ITMIN)	maximum (minimum) value of IT in mesh region
ITO	value of IT at origin of coordinates at leading edge of front blade
ITV	array of horizontal mesh line indexes (IT) corresponding to intersections of vertical mesh lines with blade surfaces (ITV(IM, SURF) is the IT value for the mesh point in the region on vertical mesh line IM which is closest to blade surface (SURF).)
ITV1, ITV2	temporary storage of ITV in TANG
ITVIM1	temporary ITV in TANG
ITVL (ITVU)	ITV of the lower (upper) blade surface on a given vertical mesh line
ITVLP1	ITVL + 1
ITVM1 (ITVP1)	ITV of a blade surface in COEFBB for the vertical mesh line to left (right) of line under consideration
ITVUM1	ITVU - 1
IV	array containing value of IP at the base of each vertical mesh line
IVMM	temporary storage of IV in COEF
J	temporary integer variable in INPUT, SLAX, and SLAV
K	array of constants; vector $\underline{k}$ in eq. (A7)
KA	integer array indicating which of the four points surrounding a mesh point lie on a boundary
KAK	real array giving the stream-function values of boundary points surrounding a mesh point
KK	integer counter in BLCD
KKK	array containing information used in plotting subroutine PLOTMY
L	temporary integer variable in SLAV
LAMBDA	λ

LAST	final value of some index
LER	array indicating location of error messages printed by program
LMAX	maximum value of u_i^{m+1}/u_i^m for eq. (B2) of ref. 11
LOC	integer variable in SLAV specifying which entry point (SLAVP or SLAVBB) was used
LOWER	integer variable representing one of the lower blade surfaces, 2 or 4
M	meridional coordinate, m
MBI	see Input
MBI2	see Input
MBI2M1	MBI2 - 1
MBI2P1	MBI2 + 1
MBIM1	MBI - 1
MBIP1	MBI + 1
MBIT, MBOT	temporary grid locations along meridional axis
MBO	see Input
MBO2	see Input
MBO2M1	MBO2 - 1
MBO2P1	MBO2 + 1
MBOM1	MBO - 1
MBOP1	MBO + 1
MH	array of m-coordinates of intersections of horizontal mesh lines with the four blade surfaces
MLE	array of m-coordinates of leading edges of the four blade surfaces (see input MLER)
MM	see Input
MMLE	temporary meridional distance in BLCD
MMM1	MM - 1
MMMSP	temporary meridional distance in BLCD
MR	see Input

MRTS	integer switch in PRECAL indicating when infinite derivatives would be encountered in a call on MHORIZ
MSL	temporary storage for MV array during plotting in SLAX
MSP	array of m-coordinates of spline points for each blade surface measured from its leading edge (see input MSP1, 2, 3, 4)
MSPMM	temporary meridional distance in BLCD
MV	array of m-coordinates of vertical mesh lines
MVIM1	temporary value of MV in TANG
NBBI	see Input
NBL	number of blades
NER	array indicating number of times certain error messages are printed by program
NI	number of streamlines blade to blade in SLAV
NIP	number of interior mesh points
NP1, NP2	integer counters in VELOCY indicating number of plotted blade-surface velocities
NRSP	see Input
NSP	number of spline points
NSPI	array containing number of spline points on each of the four blade surfaces (see input SPLNO1, 2, 3, 4)
NSPM1	NSP - 1
OMEGA	see Input
ORF	see Input
ORFOPT	upper bound for estimate of optimum Ω from eqs. (B1) and (B2) of ref. 11
ORFTEM	temporary storage for ORFOPT
P	array containing information used in the plotting subroutine PLOTMY
PITCH	$2\pi/NBL$, θ-coordinate from blade to blade
R	array of densities ρ at the four points adjacent to a point for which AAK is called

RATIO	value of u_i^{m+1}/u_i^m for use in eqs. (B2) and (B3) of ref. 11
RELER	maximum relative change in density at surface mesh points, between two outer iterations
RHO	array of densities ρ at interior mesh points
RHO1	average density ρ at upstream boundary AN
RHOB	temporary storage in VEL for a value of ρ on a blade surface
RHOHB	array of densities ρ at horizontal mesh line intersections with the four blade surfaces
RHOIP	see Input
RHOMBI	average density ρ at leading edge of front blade
RHOMB2	average density ρ at trailing edge of rear blade
RHOMM	average density ρ at downstream boundary GH
RHOT	temporary value of density ρ
RHOVB	array of densities ρ at vertical mesh line intersections with the four blade surfaces
RHOVI	average value of ρW at front-blade leading edge or upstream boundary AN
RHOVO	average value of ρW at rear-blade trailing edge or downstream boundary GH
RHOWMI	maximum value of ρW at leading edge of front blade
RHOWMO	maximum value of ρW at trailing edge of rear blade
RI (RO)	array of leading- (trailing-) edge radii on the four blade surfaces (see input RI1, 2, 3, 4 and RO1, 2, 3, 4)
RM	array of r-coordinates of the mean stream surface radii at vertical mesh lines
RMDTL2 (RMDTU2)	$(r\ d\theta/dm)^2$ at vertical mesh line intersections on lower (upper) blade surfaces
RMH	array of r-coordinates of the mean stream surface radii where horizontal mesh lines meet the four blade surfaces
RMI (RMO)	array of r-coordinates of mean stream surface radii at the inlet (outlet) of the four blade surfaces
RMM	temporary meridional distance in BLCD

RMSP — see Input

RWM — array of ρW_m where vertical mesh lines intersect the four blade surfaces

RWT — array of ρW_θ where horizontal mesh lines intersect the four blade surfaces

RZ — density ρ_0 at point for which AAK is called

S — meridional distance between two adjacent blade-surface spline points in BLCD

S1 (ST) — blade-surface number at beginning (end) of a horizontal mesh line in TANG

SAL — array of values of $\sin \alpha = dr/dm$ at each vertical mesh line

SIGN — integer constant in BLCD

SLCRD — see Input

SPLNO — number of input spline points on a blade surface

SPM — array of m-coordinates along a horizontal mesh line in TANG

SRW — integer code variable that will cause certain subroutines to write out useful data for debugging:

If SRW = 13, SPLINE will write input and output data.
If SRW = 16, SPLINT will write input and output data.
If SRW = 18, SPLN22 will write input and output data.
If SRW = 21, ROOT will write input and successive estimates of the root to which it is converging.

STGR — array of θ-coordinates of center of each trailing-edge radius with respect to the center of its leading-edge radius (see input STGRF and STGRR)

STRFN — see Input

SURF, SURFBV — integer variables referring to one of the four blade surfaces

SURFL — array of blade-surface lengths at vertical mesh line intersections for each of the four blade surfaces

SURVL — see Input

T1, T2 — elapsed time in clock pulses (1/60 sec)

TBI	$\tan \beta_{le}$
TBI1, TBIT	temporary TBI
TBO	$\tan \beta_{te}$
TBOM, TBOT	temporary TBO
TGROG	$2 \gamma R/(\gamma + 1)$
TH	θ-coordinate from leading edge of front blade to a horizontal mesh line
THETA	θ-coordinate of a point along a blade surface in BLCD
THK, THKM1	temporary variables in BLCD
THLE	array of θ-coordinates from origin of front blade to leading edge of each blade surface (see input THLER)
THSP	array of θ-coordinates of spline points for each blade surface measured from its leading edge (see input, THSP1, 2, 3, 4)
TIME	elapsed time in minutes
TINT	array of θ-coordinates in SLAV where plotted streamlines cross vertical mesh lines
TIP	see Input
TPP	T''
TSL	array of θ-coordinates of plotted streamlines
TSP	array of θ-coordinates of points along a vertical mesh line in SLAV
TTIP	T/T'_{in}
TV	array of θ-coordinates where vertical mesh lines meet the four blade surfaces
TWL	$2\omega\lambda$
TWLMR	$2\omega\lambda - (\omega r)^2$
TWW	$2\omega/w$
U	array of stream-function values at each mesh point, or of the eigenvector associated with spectral radius $\rho(L_1)$, as estimated by the power method (ref. 11)
UINT	array of values of stream function for which it is desired to obtain interpolated values of θ-coordinate in SLAV

UNEW	new value of stream-function estimate at a single point, calculated by eq. (6)
UPPER, UPPRBV	integer variables representing one of the upper blade surfaces, 1 or 3
USP	array of values of stream function along a vertical or horizontal mesh line, including boundary points
VI (VO)	average relative velocity at the leading (trailing) edge of the front (rear) blade
W	array of relative velocities W at unknown mesh points, also used for storing ρW
WCR	critical velocity on a blade surface
WCRI (WCRO)	critical velocity at leading (trailing) edge of front (rear) blade
WMB	array of ρW_m where vertical mesh lines intersect the four blade surfaces
WTB	array of ρW_θ where horizontal mesh lines intersect the four blade surfaces
WTFL	see Input
WTFLSP	see Input
WWCRM	array of ratio of blade-surface velocity (based on meridional component) to critical velocity
WWCRT	array of ratio of blade-surface velocity (based on tangential component) to critical velocity
XDOWN	array of m-coordinates where surface velocities are plotted
YACROS	array of surface velocities to be plotted

Program Listing for Subroutines Using Main Dictionary

```
      COMMON SRW,ITER,IEND,LER(2),NER(2)
      COMMON /AUKRHO/ A(2000,4),U(2000),K(2000),RHO(2000)
      COMMON /INP/ GAM,AR,TIP,RHOIP,WTFL,WTFLSP,OMEGA,ORF,BETAI,BETAO,
     1MBI,MBO,MBI2,MBO2,MM,NBBI,NBL,NRSP,MR(50),RMSP(50),BESP(50),
     2BLDAT,AANDK,ERSUR,STRFN,SLCRD,INTVL,SURVL
      COMMON /CALCON/ MBIM1,MBIP1,MBOM1,MBOP1,MBI2M1,MBI2P1,MBO2M1,
     1MBO2P1,MMM1,HM1,HM2,HM3,HT,DTLR,DMLR,PITCH,CP,EXPON,TWW,CPTIP,
     2TGROG,TBI,TBO,LAMBDA,TWL,ITMIN,ITMAX,NIP,IMS(4),BV(4),MV(100),
     3IV(101),ITV(100,4),TV(100,4),DTDMV(100,4),BETAV(100,4),
     4MH(100,4),DTDMH(100,4),BETAH(100,4),RMH(100,4),BEH(100,4),
     5RM(100),BE(100),DBDM(100),SAL(100),AAA(100)
      COMMON /GEOMIN/ CHORD(4),STGR(4),MLE(4),THLE(4),RMI(4),RMO(4),
     1RI(4),RO(4),BETI(4),BETO(4),NSPI(4),MSP(50,4),THSP(50,4)
      COMMON /RHOS/ RHOHB(100,4),RHOVB(100,4)
      COMMON /BLCDCM/ EM(50,4),INIT(4)
      INTEGER BLDAT,AANDK,ERSUR,STRFN,SLCRD,SURVL,AATEMP,SURF,SURFBV,
     1FIRST,UPPER,UPPRBV,S1,ST,SRW
      REAL K,KAK,LAMBDA,LMAX,MH,MLE,MR,MSL,MSP,MV,MVIM1
      CALL TIME1(T1)
   10 IEND = -1
      ITER = 0
      DO 20 SURF=1,4
   20 INIT(SURF) = 0
      CALL INPUT
      CALL PRECAL
   30 CALL COEF
      CALL SOR
      CALL TIME1(T2)
      TIME= (T2-T1)/3600.
      WRITE(6,1000) TIME
      CALL SLAX
      CALL TANG
      CALL VELOCY
      CALL TIME1(T2)
      TIME= (T2-T1)/3600.
      WRITE(6,1000) TIME
      IF(NER(2).GT.0) GO TO 10
      IF (IEND) 30,30,10
 1000 FORMAT (8HLTIME = ,F7.4,5H MIN.)
      END
```

```
      SUBROUTINE INPUT
C
C  INPUT READS AND PRINTS ALL INPUT DATA CARDS AND CALCULATES HORIZONTAL
C  SPACING (MV ARRAY)
C
      COMMON SRW,ITER,IEND,LER(2),NER(2)
      COMMON /AUKRHO/ A(2000,4),U(2000),K(2000),RHO(2000)
      COMMON /INP/ GAM,AR,TIP,RHOIP,WTFL,WTFLSP,OMEGA,ORF,BETAI,BETAO,
     1MBI,MBO,MBI2,MBO2,MM,NBBI,NBL,NRSP,MR(50),RMSP(50),BESP(50),
     2BLDAT,AANDK,ERSUR,STRFN,SLCRD,INTVL,SURVL
      COMMON /CALCON/ MBIM1,MBIP1,MBOM1,MBOP1,MBI2M1,MBI2P1,MBO2M1,
     1MBO2P1,MMM1,HM1,HM2,HM3,HT,DTLR,DMLR,PITCH,CP,EXPON,TWW,CPTIP,
     2TGROG,TBI,TBO,LAMBDA,TWL,ITMIN,ITMAX,NIP,IMS(4),BV(4),MV(100),
     3IV(101),ITV(100,4),TV(100,4),DTDMV(100,4),BETAV(100,4),
     4MH(100,4),DTDMH(100,4),BETAH(100,4),RMH(100,4),BEH(100,4),
     5RM(100),BE(100),DBDM(100),SAL(100),AAA(100)
      COMMON /GEOMIN/ CHORD(4),STGR(4),MLE(4),THLE(4),RMI(4),RMO(4),
     1RI(4),RO(4),BETI(4),BETO(4),NSPI(4),MSP(50,4),THSP(50,4)
      COMMON /RHOS/ RHOHB(100,4),RHOVB(100,4)
      INTEGER BLDAT,AANDK,ERSUR,STRFN,SLCRD,SURVL,AATEMP,SURF,SURFBV,
     1FIRST,UPPER,UPPRBV,S1,ST,SRW
      REAL K,KAK,LAMBDA,LMAX,MH,MLE,MR,MSL,MSP,MV,MVIM1
```

```
C
C   READ AND PRINT ALL INPUT DATA
C
      WRITE(6,1000)
      READ (5,1100)
      WRITE(6,1100)
      WRITE(6,1110)
      READ (5,1030) GAM,AR,TIP,RHOIP,WTFL,WTFLSP,OMEGA,ORF
      WRITE(6,1040) GAM,AR,TIP,RHOIP,WTFL,WTFLSP,OMEGA,ORF
      WRITE(6,1120)
      READ (5,1030)BETAI,BETAO,CHORD(1),STGR(1),CHORD(3),STGR(3),
     1MLE(3),THLE(3)
      WRITE(6,1040)BETAI,BETAO,CHORD(1),STGR(1),CHORD(3),STGR(3),
     1MLE(3),THLE(3)
      WRITE(6,1130)
      READ (5,1010) MBI,MBO,MBI2,MBO2,MM,NBBI,NBL,NRSP
      WRITE(6,1010) MBI,MBO,MBI2,MBO2,MM,NBBI,NBL,NRSP
      DO 10 J=1,4
      IF (J.EQ.1) WRITE(6,1140)
      IF (J.EQ.2) WRITE(6,1150)
      IF (J.EQ.3) WRITE(6,1160)
      IF (J.EQ.4) WRITE(6,1170)
      WRITE(6,1180) J,J,J,J,J
      READ (5,1030) RI(J),RO(J),BETI(J),BETO(J),SPLNO
      WRITE(6,1040) RI(J),RO(J),BETI(J),BETO(J),SPLNO
      NSPI(J)= SPLNO
      NSP = NSPI(J)
      WRITE(6,1190) J
      READ (5,1030) (MSP(I,J),I=1,NSP)
      WRITE(6,1040) (MSP(I,J),I=1,NSP)
      WRITE(6,1200) J
      READ (5,1030) (THSP(I,J),I=1,NSP)
   10 WRITE(6,1040) (THSP(I,J),I=1,NSP)
      WRITE(6,1210)
      READ (5,1030) (MR(I),I=1,NRSP)
      WRITE(6,1040) (MR(I),I=1,NRSP)
      WRITE(6,1220)
      READ (5,1030) (RMSP(I),I=1,NRSP)
      WRITE(6,1040) (RMSP(I),I=1,NRSP)
      WRITE(6,1230)
      READ (5,1030) (BESP(I),I=1,NRSP)
      WRITE(6,1040) (BESP(I),I=1,NRSP)
      WRITE(6,1240)
      READ (5,1010) BLDAT,AANDK,ERSOR,STRFN,SLCRD,INTVL,SURVL
      WRITE(6,1020) BLDAT,AANDK,ERSOR,STRFN,SLCRD,INTVL,SURVL
      IF (MM.LE.100.AND.NBBI.LE.50.AND.NRSP.LE.50.AND.NSPI(1).LE.50
     1.AND.NSPI(2).LE.50.AND.NSPI(3).LE.50.AND.NSPI(4).LE.50) GO TO 20
      WRITE (6,1250)
      STOP
C
C   CALCULATE MV ARRAY
C
   20 HM1 = CHORD(1)/FLOAT(MBO-MBI)
      IF(MBO.GT.MBI2.AND.MBI.NE.MBI2) HM1 = MLE(3)/FLOAT(MBI2-MBI)
      HM2 = 1.E30
      IF(MBI2.NE.MBO) HM2 = (MLE(3)-CHORD(1))/FLOAT(MBI2-MBO)
      HM3 = CHORD(3)/FLOAT(MBO2-MBI2)
      IF(MBO.GT.MBI2.AND.MBO.NE.MBO2) HM3=(CHORD(3)+MLE(3)-CHORD(1))/
     1FLOAT(MBO2-MBO)
      MBOT = MINO(MBO,MBI2)
      COMBOT = AMIN1(CHORD(1),MLE(3))
      DO 30 IM=1,MBOT
   30 MV(IM) = FLOAT(IM-MBI)*HM1
      MBIT = MAXO(MBO,MBI2)
      COMBIT = AMAX1(CHORD(1),MLE(3))
      DO 40 IM=MBOT,MBIT
   40 MV(IM) = COMBOT+FLOAT(IM-MBOT)*HM2
      DO 50 IM = MBIT,MM
   50 MV(IM) = COMBIT+FLOAT(IM-MBIT)*HM3
```

```
C
C  CALCULATE MISCELLANEOUS CONSTANTS
C
      NER(1)=0
      NER(2)=0
      PITCH = 2.*3.1415927/FLOAT(NBL)
      HT= PITCH/FLOAT(NBBI)
      DTLR= HT/1000.
      DMLR = AMIN1(HM1,HM2,HM3)/1000.
      BV(1) = 0.
      BV(2) = 1.
      BV(3) = -WTFLSP/WTFL
      BV(4) = 1.+BV(3)
      MBIM1= MBI-1
      MBIP1= MBI+1
      MBOM1= MBO-1
      MBOP1= MBO+1
      MBI2M1= MBI2-1
      MBI2P1= MBI2+1
      MBO2M1= MBO2-1
      MBO2P1= MBO2+1
      MMM1 = MM-1
      CP = AR/(GAM-1.)*GAM
      EXPON= 1./(GAM-1.)
      TWW= 2.*OMEGA/WTFL
      CPTIP= 2.*CP*TIP
      TGROG= 2.*GAM*AR/(GAM+1.)
      CALL SPLINT(MR,RMSP,NRSP,MV,MM,RM,SAL)
      CALL SPLINT(MR,BESP,NRSP,MV,MM,BE,DBDM)
C
C  CALCULATE GEOMETRICAL CONSTANTS
C
      CHORD(2) = CHORD(1)
      CHORD(4) = CHORD(3)
      STGR(2) = STGR(1)
      STGR(4) = STGR(3)
      MLE(1) = 0.
      MLE(2) = 0.
      MLE(4) = MLE(3)
      THLE(1) = 0.
      THLE(2) = PITCH
      THLE(4) = PITCH+THLE(3)
      RMI(1) = RM(MBI)
      RMI(2) = RM(MBI)
      RMI(3) = RM(MBI2)
      RMI(4) = RM(MBI2)
      RMO(1) = RM(MBO)
      RMO(2) = RM(MBO)
      RMO(3) = RM(MBO2)
      RMO(4) = RM(MBO2)
C
C  INITIALIZE ARRAYS
C
      DO 60 I=1,2000
      U(I) = 1.
      K(I) = 0.
   60 RHO(I) = RHOIP
      DO 70 IM=1,100
      DO 70 SURF=1,4
      RHOHB(IM,SURF) = RHOIP
      RHOVB(IM,SURF) = RHOIP
   70 ITV(IM,SURF) = -10000
      RETURN
 1000 FORMAT (1H1)
 1010 FORMAT (16I5)
 1020 FORMAT (1X,16I7)
 1030 FORMAT (8F10.5)
 1040 FORMAT (1X,8G16.7)
 1100 FORMAT (80H
     1
 1110 FORMAT (7X,3HGAM,14X,2HAR,13X,3HTIP,12X,5HRHOIP,12X,4HWTFL,11X,6HW
     1TFLSP,10X,5HOMEGA,12X,3HURF)
 1120 FORMAT (6X,5HBETAI,10X,5HBETAO,11X,6HCHORDF,11X,5HSTGRF,10X,
```

```
     16HCHORDR,10X,5HSTGRR,12X,4HMLER,11X,5HTHLER)
1130 FORMAT (41H   MBI   MBO MBI2 MBO2   MM   NBBI   NBL NRSP)
1140 FORMAT (53HL     BLADE SURFACE 1  --  UPPER SURFACE - FRONT BLADE)
1150 FORMAT (53HL     BLADE SURFACE 2  --  LOWER SURFACE - FRONT BLADE)
1160 FORMAT (52HL     BLADE SURFACE 3  --  UPPER SURFACE - REAR BLADE)
1170 FORMAT (52HL     BLADE SURFACE 4  --  LOWER SURFACE - REAR BLADE)
1180 FORMAT (7X,2HRI,I1,12X,2HRO,I1,12X,4HBETI,I1,11X,4HBETO,I1,11X,5HS
     1PLNO,I1)
1190 FORMAT (7X,3HMSP,I1,2X,5HARRAY)
1200 FORMAT (7X,4HTHSP,I1,2X,5HARRAY)
1210 FORMAT (16HL        MR  ARRAY)
1220 FORMAT (7X,11HRMSP  ARRAY)
1230 FORMAT (7X,11HBESP  ARRAY)
1240 FORMAT (52HL    BLDAT  AANDK  ERSOR  STRFN  SLCRD  INTVL  SURVL)
1250 FORMAT (41H1 MM,NBBI,NRSP,OR SOME SPLNO IS TOO LARGE)
      END
```

```
      SUBROUTINE PRECAL
C
C  PRECAL CALCULATES ALL REQUIRED FIXED CONSTANTS
C
      COMMON SRW,ITER,IEND,LER(2),NER(2)
      COMMON /AUKRHO/ A(2000,4),U(2000),K(2000),RHO(2000)
      COMMON /INP/ GAM,AR,TIP,RHOIP,WTFL,WTFLSP,OMEGA,ORF,BETAI,BETAO,
     1MBI,MBO,MBI2,MBO2,MM,NBBI,NBL,NRSP,MR(50),RMSP(50),BESP(50),
     2BLDAT,AANDK,ERSOR,STRFN,SLCRD,INTVL,SURVL
      COMMON /CALCON/ MBIM1,MBIP1,MBOM1,MBOP1,MBI2M1,MBI2P1,MBO2M1,
     1MBO2P1,MMM1,HM1,HM2,HM3,HT,DTLR,DMLR,PITCH,CP,EXPON,TWW,CPTIP,
     2TGROG,TBI,TBO,LAMBDA,TWL,ITMIN,ITMAX,NIP,IMS(4),BV(4),MV(100),
     3IV(101),ITV(100,4),TV(100,4),DTDMV(100,4),BETAV(100,4),
     4MH(100,4),DTDMH(100,4),BETAH(100,4),RMH(100,4),BEH(100,4),
     5RM(100),BE(100),DBDM(100),SAL(100),AAA(100)
      INTEGER BLDAT,AANDK,ERSOR,STRFN,SLCRD,SURVL,AATEMP,SURF,SURFBV,
     1FIRST,UPPER,UPPRBV,S1,ST,SRW
      REAL K,KAK,LAMBDA,LMAX,MH,MLE,MR,MSL,MSP,MV,MVIM1
      EXTERNAL BL1,BL2,BL3,BL4
C
C  CALCULATE LAMBDA AND VI
C
      BETAI = BETAI/57.295779
      BETAO = BETAO/57.295779
      TBI = SIN(BETAI)/COS(BETAI)
      TBO = SIN(BETAO)/COS(BETAO)
   10 RHOT = RHOIP
      RHOVI = WTFL/BE(MBI)/PITCH/COS(BETAI)/RM(MBI)
   20 VI = RHOVI/RHOT
      LAMBDA = RM(MBI)*(VI*SIN(BETAI)+OMEGA*RM(MBI))
      TTIP = 1.-(VI**2+2.*OMEGA*LAMBDA-(OMEGA*RM(MBI))**2)/CPTIP
      IF(TTIP.LE.0.) GO TO 30
      RHOMBI = RHOIP*TTIP**EXPON
      IF(ABS(RHOMBI-RHOT)/RHOIP.LT..000001) GO TO 40
      RHOT = RHOMBI
      GO TO 20
   30 WTFL = WTFL/2.
      NER(2)= NER(2)+1
      WRITE(6,1020) WTFL
      IF(NER(2).EQ.10) STOP
      GO TO 10
   40 VI = RHOVI/RHOMBI
      LAMBDA = RM(MBI)*(VI*SIN(BETAI)+OMEGA*RM(MBI))
C
C  CALCULATE MAXIMUM VALUES FOR RHO*W AT LEADING AND TRAILING EDGE
C
      TWL = 2.*OMEGA*LAMBDA
      AA = (TWL-(OMEGA*RM(MBI))**2)/CPTIP
      TPP = TIP*(1.-AA)
      BB = TGROG*TPP
```

```
      TTIP = 1.-BB/CPTIP-AA
      RHOT = RHOIP*TTIP**EXPON
      RHOWMI = RHOT*SQRT(BB)
      AA = (TWL-(OMEGA*RM(MBO2))**2)/CPTIP
      TPP = TIP*(1.-AA)
      BB = TGROG*TPP
      TTIP = 1.-BB/CPTIP-AA
      RHOT = RHOIP*TTIP**EXPON
      RHOWMO = RHOT*SQRT(BB)
C
C   CALCULATE VO AND W-CRITICAL AT BLADE LEADING AND TRAILING EDGE
C
      RHOVO = WTFL/BE(MBO2)/PITCH/COS(BETAO)/RM(MBO2)
      RHOMB2 = RHOIP
      TWLMR = TWL-(OMEGA*RM(MBO2))**2
      LER(1)=1
C     DENSTY CALL NO. 1
      CALL DENSTY(RHOVO,RHOMB2,VO,TWLMR,CPTIP,EXPON,RHOIP,GAM,AR,TIP)
      WCRI = SQRT(TGROG*TIP*(1.-(TWL-(OMEGA*RM(MBI))**2)/CPTIP))
      WCRO = SQRT(TGROG*TIP*(1.-(TWL-(OMEGA*RM(MBO2))**2)/CPTIP))
C
C    CALCULATE BETA CORRECTED TO BOUNDARY A-N AND G-H
C
      TWLMR = TWL-(OMEGA*RM(1))**2
      RHO1 = RHOMBI
      TBI1 = 1.E20
   50 TBIT = (TBI/BE(MBI)*RHO1/RHOMBI+OMEGA*(RM(MBI)**2-RM(1)**2)*RHO1
     1/WTFL*PITCH)*BE(1)
      IF(ABS(TBI1-TBIT).LT..00001) GO TO 60
      TBI1 = TBIT
      RHOVI = WTFL/PITCH*SQRT(1.+TBI1**2)/BE(1)/RM(1)
      LER(1)=2
C     DENSTY CALL NO. 2
      CALL DENSTY (RHOVI,RHO1,AA,TWLMR,CPTIP,EXPON,RHOIP,GAM,AR,TIP)
      GO TO 50
   60 TBI = TBIT
      BTAIN = ATAN(TBI)*57.295779
      TWLMR = TWL-(OMEGA*RM(MM))**2
      RHOMM = RHOMB2
      TBOM = 1.E20
   70 TBOT = (TBO/BE(MBO2)*RHOMM/RHOMB2+OMEGA*(RM(MBO2)**2-RM(MM)**2)*
     1RHOMM/WTFL*PITCH)*BE(MM)
      IF (ABS(TBOM-TBOT).LT..00001) GO TO 80
      TBOM = TBOT
      RHOVO = WTFL/PITCH*SQRT(1.+TBOM**2)/BE(MM)/RM(MM)
      LER(1)=3
C     DENSTY CALL NO. 3
      CALL DENSTY (RHOVO,RHOMM,AA,TWLMR,CPTIP,EXPON,RHOIP,GAM,AR,TIP)
      GO TO 70
   80 TBO = TBOT
      BTAOUT = ATAN(TBO)*57.295779
C
C   CALCULATE TV, ITV, IV, DTDMV, AND BETAV ARRAYS
C
      ITMIN = 0
      ITMAX = NBBI-1
C   TV, ITV, AND DTDMV ON BLADES
      DO 90  IM=MBI,MBO
      LER(2)=1
C     BLCD CALL NO. 1
      CALL BL1(MV(IM),TV(IM,1),DTDMV(IM,1),INF)
      ITV(IM,1)= INT((TV(IM,1)+DTLR)/HT)
      IF (TV(IM,1).GT.-DTLR) ITV(IM,1)=ITV(IM,1)+1
      ITMIN= MINO(ITMIN,ITV(IM,1))
      LER(2)=2
C     BLCD CALL NO. 2
      CALL BL2(MV(IM),TV(IM,2),DTDMV(IM,2),INF)
      ITV(IM,2)= INT((TV(IM,2)-DTLR)/HT)
      IF (TV(IM,2).LT.DTLR) ITV(IM,2)=ITV(IM,2)-1
   90 ITMAX= MAXO(ITMAX,ITV(IM,2))
      DO 110 IM=MB12,MBO2
      LER(2)=3
```

```
C      BLCD CALL NO. 3
       CALL BL3(MV(IM),TV(IM,3),DTDMV(IM,3),INF)
       ITV(IM,3)= INT((TV(IM,3)+DTLR)/HT)
       IF (TV(IM,3).GT.-DTLR) ITV(IM,3)=ITV(IM,3)+1
       IF (IM.GT.MBO) GO TO 100
       TV(IM,3) = TV(IM,3)+PITCH
  100  ITMIN= MINO(ITMIN,ITV(IM,3))
       LER(2)=4
C      BLCD CALL NO. 4
       CALL BL4(MV(IM),TV(IM,4),DTDMV(IM,4),INF)
       ITV(IM,4)= INT((TV(IM,4)-DTLR)/HT)
       IF (TV(IM,4).LT.DTLR) ITV(IM,4)=ITV(IM,4)-1
  110  ITMAX= MAXO(ITMAX,ITV(IM,4))
C  ITV AND IV UPSTREAM OF FRONT BLADE
       FIRST = 0
       LAST = NBBI-1
       DO 120 IM=1,MBIM1
       IV(IM) = NBBI*(IM-1)+1
       ITV(IM,1)= FIRST
  120  ITV(IM,2)= LAST
C  ITV BETWEEN FRONT AND REAR BLADES
       IF (MBOP1.GT.MBI2M1) GO TO 140
       LAST= ITV(MBI2,4)
       FIRST= LAST+1-NBBI
       DO 130 IM=MBOP1,MBI2M1
       ITV(IM,3)= FIRST
  130  ITV(IM,4)= LAST
       ITMIN = MINO(ITMIN,ITV(MBOP1,3))
C  ITV DOWNSTREAM OF REAR BLADE
  140  LAST= ITV(MBO2,4)
       FIRST= LAST+1-NBBI
       DO 150 IM=MBO2P1,MM
       ITV(IM,3)= FIRST
  150  ITV(IM,4)= LAST
       ITMIN = MINO(ITMIN,ITV(MM,3))
C  FINISH CALCULATING IV ARRAY
       IV(MBI)  = NBBI*MBIM1+1
       MBOT = MINO(MBO,MBI2M1)
       IF(MBI.GT.MBOT) GO TO 165
       DO 160 IM=MBI,MBOT
  160  IV(IM+1) = IV(IM)+ITV(IM,2)-ITV(IM,1)+1
  165  IF(MBI2.GT.MBO) GO TO 180
       DO 170 IM=MBI2,MBO
  170  IV(IM+1) = IV(IM)+ITV(IM,2)-ITV(IM,3)+ITV(IM,4)-ITV(IM,1)+2-NBBI
  180  DO 190 IM=MBOP1,MM
  190  IV(IM+1) = IV(IM)+ITV(IM,4)-ITV(IM,3)+1
C  BETAV ARRAY
       DO 200 SURF=1,2
       DO 200 IM=MBI,MBO
  200  BETAV(IM,SURF) = ATAN(DTDMV(IM,SURF)*RM(IM))*57.295779
       DO 210 SURF=3,4
       DO 210 IM=MBI2,MBO2
  210  BETAV(IM,SURF) = ATAN(DTDMV(IM,SURF)*RM(IM))*57.295779
       NIP = IV(MM)+NBBI-1
       WRITE(6,1030) VI,RHOWMI,WCRI,BTAIN,VO,RHOWMO,WCRO,BTAOUT
       WRITE(6,1040) PITCH,HT,HM1,HM2,HM3
       WRITE(6,1050) ITMIN,ITMAX,LAMBDA,NIP
       WRITE(6,1060) (SURF,BV(SURF),SURF=1,4)
       IF(BLDAT.LE.0) GO TO 230
       FIRST = MBI
       LAST = MBO
       WRITE (6,1070)
       DO 220 SURF=1,3,2
       I = SURF+1
       WRITE (6,1080) SURF,I,(MV(IM),TV(IM,SURF),DTDMV(IM,SURF),
      1TV(IM,I),DTDMV(IM,I),IM=FIRST,LAST)
       FIRST = MBI2
  220  LAST = MBO2
       WRITE (6,1090) (IM,MV(IM),RM(IM),SAL(IM),BE(IM),DBDM(IM),IM=1,MM)
  230  CONTINUE
```

```
C
C  CALCULATE MH AND DTOMH ARRAYS
C
      ITO = ITV(1,1)
      MRTS = 1
      IMS(1) = 1
      MH(1,1) = 0.
      DTDMH(1,1) = 1.E10
      LER(2)=5
C     BLCD AND ROOT (VIA MHORIZ) CALL NO. 5
      CALL MHORIZ(MV,ITV(1,1),BL1,MBI,MBO,ITO,HT,DTLR,0,IMS(1),MH(1,1),
     1DTDMH(1,1),MRTS)
      IF (ITV(MBO,1)-ITV(MBO,2)+NBBI.NE.2) GO TO 240
      IMSL = IMS(1)+1
      MH(IMSL,1) = MV(MBO)
      DTDMH(IMSL,1) = -1.E10
      IMS(1) = IMSL
  240 IMS(2) = 0
      MRTS = 1
      LER(2)=6
C     BLCD AND ROOT (VIA MHORIZ) CALL NO. 6
      CALL MHORIZ(MV,ITV(1,2),BL2,MBI,MBO,ITO,HT,DTLR,1,IMS(2),MH(1,2),
     1DTDMH(1,2),MRTS)
      IMS(3) = 0
      IF (ITV(MBI2,3)-ITV(MBI2,4).NE.2.AND.
     1ITV(MBI2,4)-ITV(MBI2,3).NE.NBBI-2) GO TO 250
      MRTS = 1
      IMS(3) = 1
      MH(1,3) = MV(MBI2)
      DTDMH(1,3) = 1.E10
  250 LER(2)=7
C     BLCD AND ROOT (VIA MHORIZ) CALL NO. 7
      CALL MHORIZ(MV,ITV(1,3),BL3,MBI2,MBO,ITO,HT,DTLR,0,IMS(3),MH(1,3),
     1DTDMH(1,3),MRTS)
      MBOT = MAX0(MBO,MBI2)
      LER(2)=8
C     BLCD AND ROOT (VIA MHORIZ) CALL NO. 8
      CALL MHORIZ(MV,ITV(1,3),BL3,MBOT,MBO2,ITO,HT,DTLR,0,IMS(3),
     1MH(1,3),DTDMH(1,3),MRTS)
      IF (ITV(MBO2,3)-ITV(MBO2,4)+NBBI.NE.2) GO TO 260
      IMSL = IMS(3)+1
      MH(IMSL,3) = MV(MBO2)
      DTDMH(IMSL,3) = -1.E10
      IMS(3) = IMSL
  260 IMS(4) = 0
      IF (ITV(MBI2,3)-ITV(MBI2,4).EQ.2.OR.ITV(MBI2,4)-ITV(MBI2,3).EQ.
     1NBBI-2) MRTS = 1
      LER(2)=9
C     BLCD AND ROOT (VIA MHORIZ) CALL NO. 9
      CALL MHORIZ(MV,ITV(1,4),BL4,MBI2,MBO2,ITO,HT,DTLR,1,IMS(4),
     1MH(1,4),DTDMH(1,4),MRTS)
      I = MAX0(IMS(1),IMS(2),IMS(3),IMS(4))
      IF (I.LE.100) GO TO 270
      WRITE(6,1100) I
      STOP
C
C  CORRECT ITV ARRAY FOR OVERLAPPING PORTION OF BLADE SURFACE 3
C
  270 IF (MBI2.GT.MBO) GO TO 290
      DO 280 IM=MBI2,MBO
  280 ITV(IM,3) = ITV(IM,3)+NBBI
  290 IF(BLDAT.LE.0) GO TO 300
      WRITE (6,1110)  (IM,IV(IM),(ITV(IM,SURF),SURF=1,4),IM=1,MM)
C
C  CALCULATE RMH, BEH, AND BETAH ARRAYS
C
  300 IF(BLDAT.GT.0) WRITE(6,1120)
      DO 320 SURF=1,4
      CALL SPLINT(MR,RMSP,NRSP,MH(1,SURF),IMS(SURF),RMH(1,SURF),AAA)
      CALL SPLINT(MR,BESP,NRSP,MH(1,SURF),IMS(SURF),BEH(1,SURF),AAA)
      IMSS = IMS(SURF)
      IF(IMSS.LT.1) GO TO 320
      DO 310 IHS = 1,IMSS
```

```
  310 BETAH(IHS,SURF) = ATAN(DTDMH(IHS,SURF)*RMH(IHS,SURF))*57.295779
      IF (BLDAT.GT.0) WRITE(6,1130) SURF,(MH(IM,SURF),RMH(IM,SURF),
     1BEH(IM,SURF),BETAH(IM,SURF),DTDMH(IM,SURF),IM=1,IMSS)
  320 CONTINUE
      IF (BLDAT.LE.0) GO TO 340
      WRITE (6,1140)
      IT = ITMIN
  330 IF (IT.GT.ITMAX) GO TO 340
      TH = FLOAT(IT)*HT
      WRITE (6,1010) IT,TH
      IT = IT+1
      GO TO 330
  340 IF(NIP.LE.2000) GO TO 350
      WRITE(6,1150)
      STOP
  350 WRITE (6,1000)
      RETURN
 1000 FORMAT (1H1)
 1010 FORMAT (4X,I4,G16.5)
 1020 FORMAT(60HLINPUT WEIGHT FLOW (WTFL) IS TOO LARGE AT BLADE LEADING
     1EDGE/16H WTFL REDUCED TO,G14.6)
 1030 FORMAT (1H1/////24X,10HFREESTREAM,8X,13HMAXIMUM VALUE,
     17X,8HCRITICAL,30X,14HBETA CORRECTED/25X,8HVELOCITY,10X,9HFOR RHO*W
     2,10X,8HVELOCITY,31X,11HTO BOUNDARY/1X,17HLEADING EDGE  B-M,3G18.5,
     312X,12HBOUNDARY A-N,G18.5/1X,17HTRAILING EDGE F-I,3G18.5,12X,
     412HBOUNDARY G-H,G18.5)
 1040 FORMAT (/////5X,28HCALCULATED PROGRAM CONSTANTS//5X,5HPITCH,13X,
     12HHT,13X,3HHM1,13X,3HHM2,13X,3HHM3/1X,5G16.7)
 1050 FORMAT (/5X,5HITMIN,10X,5HITMAX/4X,I5,10X,I5//5X,6HLAMBDA/1X,G16.7
     1///5X,33HNUMBER OF INTERIOR MESH POINTS = ,I5)
 1060 FORMAT (//////5X,23HSURFACE BOUNDARY VALUES//5X,7HSURFACE,7X,2HBV
     1/(5X,I4,4X,F10.5))
 1070 FORMAT (1H1,6X,62HBLADE DATA AT INTERSECTIONS OF VERTICAL MESH LIN
     1ES WITH BLADES)
 1080 FORMAT (1HL,22X,13HBLADE SURFACE,I2,15X,13HBLADE SURFACE,I2/7X,
     1    1HM,14X,2HTV,11X,5HDTDMV,12X,2HTV,11X,5HDTDMV/(5G15.5))
 1090 FORMAT (1H1,13X,44HSTREAM SHEET COORDINATES AND THICKNESS TABLE /
     1    2X,2HIM,7X,1HM,14X,1HR,13X,3HSAL,13X,1HB,12X,5HDB/DM/(1X,I3,
     2    5G15.5))
 1100 FORMAT(34HLONE OF THE MH ARRAYS IS TOO LARGE/7H IT HAS,I5, 8H  POI
     1NTS)
 1110 FORMAT (4H1 IM,9X,8HIV ARRAY,32X,9HITV ARRAY/38X,5HBLADE/37X,7HSUR
     1FACE,3X,1H1,5X,1H2,5X,1H3,5X,1H4/39X,3HNO./(1X,I3,5X,I10,25X,
     24(I4,2X)))
 1120 FORMAT (67H1M COORDINATES OF INTERSECTIONS OF HORIZONTAL MESH LINE
     1S WITH BLADE)
 1130 FORMAT (25HLMH ARRAY - BLADE SURFACE,I2//15X,2HMH,19X,3HRMH,19X,
     1    3HBEH,18X,5HBETAH,17X,5HDTDMH/(5G22.4))
 1140 FORMAT (43H1THETA COORDINATES OF HORIZONTAL MESH LINES//6X,2HIT,
     15X,5HTHETA)
 1150 FORMAT(48HLTHE NUMBER OF INTERIOR MESH POINTS EXCEEDS 2000)
      END
```

```
      SUBROUTINE COEF
C
C  COEF CALCULATES FINITE DIFFERENCE COEFFICIENTS, A, AND CONSTANTS, K,
C  AT ALL UNKNOWN MESH POINTS FOR THE ENTIRE REGION
C
      COMMON SRW,ITER,IEND,LER(2),NER(2)
      COMMON /AUKRHO/ A(2000,4),U(2000),K(2000),RHO(2000)
      COMMON /INP/ GAM,AR,TIP,RHOIP,WTFL,WTFLSP,OMEGA,ORF,BETAI,BETAO,
     1MBI,MBO,MBI2,MBO2,MM,NBBI,NBL,NRSP,MR(50),RMSP(50),BESP(50),
     2BLDAT,AANDK,ERSOR,STRFN,SLCRD,INTVL,SURVL
      COMMON /CALCON/ MBIM1,MBIP1,MBOM1,MBOP1,MBI2M1,MBI2P1,MBO2M1,
     1MBO2P1,MMM1,HM1,HM2,HM3,HT,DTLR,DMLR,PITCH,CP,EXPON,TWW,CPTIP,
     2TGROG,TBI,TBO,LAMBDA,TWL,ITMIN,ITMAX,NIP,IMS(4),BV(4),MV(100),
     3IV(101),ITV(100,4),TV(100,4),DTDMV(100,4),BETAV(100,4),
     4MH(100,4),DTDMH(100,4),BETAH(100,4),RMH(100,4),BEH(100,4),
     5RM(100),BE(100),DBDM(100),SAL(100),AAA(100)
      COMMON /HRBAAK/ H(4),R(4),B(4),KAK(4),KA(4),IH(4),RZ,BZ
      INTEGER BLDAT,AANDK,ERSOR,STRFN,SLCRD,SURVL,AATEMP,SURF,SURFBV,
     1FIRST,UPPER,UPPRBV,S1,ST,SRW
      REAL K,KAK,LAMBDA,LMAX,MH,MLE,MR,MSL,MSP,MV,MVIM1
C  INITIALIZE ARRAYS
      ITER = ITER+1
      IH(1) = 1
      DO 10 I=2,4
   10 IH(I) = 0
      IF(ITV(MBI2,3)-ITV(MBI2,4).EQ.2) IH(3) = 1
      IF(ITV(MBI2,4)-ITV(MBI2,3).EQ.NBBI-2.AND.MBI2.NE.MBOP1) IH(3) = 1
C  INCOMPRESSIBLE CASE
      IF(GAM.NE.1.5.OR.AR.NE.1000..OR.TIP.NE.1.E6) GO TO 20
      IEND = 1
      GO TO 40
C  ADJUSTMENT OF PRINTING CONTROL VARIABLES
   20 IF(ITER.NE.1.AND.ITER.NE.2) GO TO 30
      AANDK = AANDK-1
      ERSOR = ERSOR-1
      STRFN = STRFN-1
      SLCRD = SLCRD-1
      INTVL = INTVL-1
      SURVL = SURVL-1
   30 IF(IEND.NE.0) GO TO 40
      AANDK = AANDK+2
      ERSOR = ERSOR+2
      STRFN = STRFN+2
      SLCRD = SLCRD+2
      INTVL = INTVL+2
      SURVL = SURVL+2
C
C  FIRST VERTICAL MESH LINE
C
   40 DO 50 IP=1,NBBI
      A(IP,1) = 0.
      A(IP,2) = 0.
      A(IP,3) = 0.
      A(IP,4) = 1.
   50 K(IP) = HM1*TBI/PITCH/RM(1)
C
C  UPSTREAM OF BLADES, EXCEPT FOR FIRST VERTICAL MESH LINE
C
      IF(2.GT.MBIM1) GO TO 70
      DO 60 IM=2,MBIM1
   60 CALL COEFP(IM,1,2)
C
C  BETWEEN FRONT BLADES
C
   70 MBOT = MIN0(MBO,MBI2M1)
      IF(MBI.GT.MBOT) GO TO 90
      DO 80 IM=MBI,MBOT
   80 CALL COEFBB(IM,1,2,1)
   90 IF(MBI2.GT.MBO) GO TO 110
C
C  OVERLAPPING CASE
C
      DO 100 IM=MBI2,MBO
      CALL COEFBB(IM,1,4,1)
```

```
  100 CALL COEFBB( IM,3,2,4)
      GO TO 130
C
C   NON - OVERLAPPING CASE
C
  110 IF(MBOP1.GT.MBI2M1) GO TO 130
      DO 120 IM=MBOP1,MBI2M1
  120 CALL COEFP(IM,3,4)
C
C  BETWEEN REAR BLADES
C
  130 MBIT = MAXO(MBI2,MBOP1)
      IF(MBIT.GT.MBO2) GO TO 150
      DO 140 IM=MBIT,MBO2
  140 CALL COEFBB( IM,3,4,3)
C
C  DOWNSTREAM OF BLADES EXCEPT FOR FINAL MESH LINE
C
  150 IF(MBO2P1.GT.MMM1) GO TO 170
      DO 160 IM=MBO2P1,MMM1
  160 CALL COEFP(IM,3,4)
C
C  FINAL VERTICAL MESH LINE
C
  170 IVMM = IV(MM)
      DO 180 IP=IVMM,NIP
      A( IP,1) = 0.
      A( IP,2) = 0.
      A( IP,3) = 1.
      A( IP,4) = 0.
  180 K( IP) = -HM3*TBO/PITCH/RM(MM)
C
C  TAKE CARE OF POINTS ADJACENT TO B, AND CASES WHEN POINTS J,C,E, OR F
C  ARE GRID POINTS
C
C  POINT B
      IP = IV(MBIM1)
      A( IP,4) = 0.
C  POINT J
      IF( ITV(MBI2,3)-ITV(MBI2,4).NE.2) GO TO 190
      IT = ITV(MBI2,4)+1
      IP = IPF(MBI2M1,IT)
      K( IP) = K( IP )+A( IP,4)*BV(4)
      A( IP,4) = 0.
C  POINT C
  190 IF( ITV(MBO,1)-ITV(MBO,2)+NBBI.NE.2) GO TO 200
      IT = ITV(MBO,1)-1
      IP = IPF(MBOP1,IT)
      A( IP,3) = 0.
C  POINT E
  200 IF( ITV(MBI2,4)-ITV(MBI2,3).NE.NBBI-2.OR.MBI2.EQ.MBOP1) GO TO 210
      IP = IV(MBI2M1)
      K( IP) = K( IP )+A( IP,4)*BV(3)
      A( IP,4) = 0.
C  POINT F
  210 IF (( ITV(MBO2,3)-ITV(MBO2,4)+NBBI.NE.2).AND. (ITV(MBO2,3)
     1-ITV(MBO2,4).NE.2)) GO TO 220
      IP = IV(MBO2P1)
      K( IP) = K( IP )+A( IP,3)*BV(3)
      A( IP,3) = 0.
C
C   LINE K-L AND LINE TO RIGHT OF C-D
C
  220 FIRST = MAXO(ITV(MBOP1,3)+NBBI ,ITV(MBO,3))
      IPKL = IPF(MBO,FIRST)
      IPL = IPKL+ITV(MBO,2)-FIRST
      IPCD = IPF(MBOP1,FIRST-NBBI)
  230 IF( IPKL.GT.IPL) RETURN
      K( IPKL) = K( IPKL)+A(IPKL,4)
      K( IPCD) = K( IPCD)-A(IPCD,3)
      IPKL = IPKL+1
      IPCD = IPCD+1
      GO TO 230
      END
```

```
      SUBROUTINE COEFBB(IM,UPPER,LOWER,UPPRBV)
C
C  COEFBB CALCULATES FINITE DIFFERENCE COEFFICIENTS, A, AND CONSTANTS, K
C  ALONG ALL VERTICAL MESH LINES WHICH INTERSECT BLADES
C
      COMMON /AUKRHO/ A(2000,4),U(2000),K(2000),RHO(2000)
      COMMON /INP/ GAM,AR,TIP,RHOIP,WTFL,WTFLSP,OMEGA,ORF,BETAI,BETAO,
     1MBI,MBO,MBI2,MBO2,MM,NBBI,NBL,NRSP,MR(50),RMSP(50),BESP(50),
     2BLDAT,AANDK,ERSOR,STRFN,SLCRD,INTVL,SURVL
      COMMON /CALCON/ MBIM1,MBIP1,MBOM1,MBOP1,MBI2M1,MBI2P1,MBO2M1,
     1MBO2P1,MMM1,HM1,HM2,HM3,HT,DTLR,DMLR,PITCH,CP,EXPON,TWW,CPTIP,
     2TGROG,TBI,TBO,LAMBDA,TWL,ITMIN,ITMAX,NIP,IMS(4),BV(4),MV(100),
     3IV(101),ITV(100,4),TV(100,4),DTDMV(100,4),BETAV(100,4),
     4MH(100,4),DTDMH(100,4),BETAH(100,4),RMH(100,4),BEH(100,4),
     5RM(100),BE(100),DBDM(100),SAL(100),AAA(100)
      COMMON /HRBAAK/ H(4),R(4),B(4),KAK(4),KA(4),IH(4),RZ,BZ
      INTEGER BLDAT,AANDK,ERSOR,STRFN,SLCRD,SURVL,AATEMP,SURF,SURFBV,
     1FIRST,UPPER,UPPRBV,S1,ST,SRW
      REAL K,KAK,LAMBDA,LMAX,MH,MLE,MR,MSL,MSP,MV,MVIM1
      IF(ITV(IM,UPPER).GT.ITV(IM,LOWER)) RETURN
      ITVU = ITV(IM,UPPER)
      ITVL = ITV(IM,LOWER)
      IT   = ITVU - 1
      IPU = IPF(IM,ITVU)
      IPL = IPU+ITVL-ITVU
      DO 90 IP=IPU,IPL
      IT = IT+1
      CALL HRB(IM,IT,IP)
      DO 10 I=1,4
      KAK(I) = 0.
   10 KA(I) = 0
C  FIX HRB VALUES FOR LINES C-D AND K-L
      IF(IM.NE.MBOP1) GO TO 20
      IF(IT.GE.ITV(IM-1,1)) GO TO 20
      IP3 = IPF(IM-1,IT+NBBI)
      R(3) = RHO(IP3)
   20 IF(IM+1.NE.MBOP1) GO TO 60
      IF(MBI2P1-MBOP1) 30,30,40
   30 IF(IT-ITV(IM,3)) 60,50,50
   40 IF(IT.LE.ITV(IM+1,4)) GO TO 60
   50 IP4 = IPF(IM+1,IT-NBBI)
      R(4) = RHO(IP4)
C  FIX HRB VALUES FOR CASES WHERE MESH LINES INTERSECT BLADES
   60 IF (IT .EQ. ITV(IM,UPPER)) CALL BDRY12(1,IM,IT,UPPER,UPPRBV)
      IF (IT .EQ. ITV(IM,LOWER)) CALL BDRY12(2,IM,IT,LOWER,LOWER)
      ITVM1 = ITV(IM-1,UPPER)
      ITVP1 = ITV(IM+1,UPPER)
      IF (IM.EQ.MBO.AND.UPPER.EQ.3) ITVP1 = ITVP1+NBBI
      IF (IM.EQ.MBOP1) ITVM1 = ITVM1-NBBI
      IF (IT.LT.ITVM1) CALL BDRY34(3,IM,UPPER,UPPRBV)
      IF (IT.LT.ITVP1) CALL BDRY34(4,IM,UPPER,UPPRBV)
      IF(IM.EQ.MBI2.AND.LOWER.EQ.4) GO TO 70
      IF(IM.EQ.MBOP1.AND.LOWER.EQ.4.AND.MBI2.GT.MBO) GO TO 70
      IF (IT.GT.ITV(IM-1,LOWER)) CALL BDRY34(3,IM,LOWER,LOWER)
   70 IF(IM.EQ.MBO.AND.LOWER.EQ.2) GO TO 80
      IF (IT.GT.ITV(IM+1,LOWER)) CALL BDRY34(4,IM,LOWER,LOWER)
C  COMPUTE A AND K COEFFICIENTS
   80 CALL AAK(IM,IP)
      DO 90 I=1,4
      K(IP) = K(IP)+KAK(I)*A(IP,I)
   90 IF(KA(I).EQ.1) A(IP,I) = 0.
      RETURN
C
C  COEFP CALCULATES FINITE DIFFERENCE COEFFICIENTS, A, AND CONSTANTS, K,
```

```
C  ALONG ALL VERTICAL MESH LINES WHICH DO NOT INTERSECT BLADES
C
      ENTRY COEFP(IM,UPPER,LOWER)
      ITVU = ITV(IM,UPPER)
      ITVL = ITV(IM,LOWER)
      IT = ITVU-1
      IPU = IV(IM)
      IPL = IV(IM+1)-1
      DO 100 IP=IPU,IPL
      IT = IT+1
      CALL HRB(IM,IT,IP)
      IF (IT.EQ.ITVU) R(1) = RHO(IPL)
      IF (IT.EQ.ITVL) R(2) = RHO(IPU)
      IF(IM.NE.MBOP1) GO TO 100
      IF(IT.GE.ITV(IM-1,1)) GO TO 100
      IP3 = IPF(IM-1,IT+NBBI)
      R(3) = RHO(IP3)
  100 CALL AAK(IM,IP)
      K(IPL) = K(IPL)+A(IPL,2)
      K(IPU) = K(IPU)-A(IPU,1)
      RETURN
      END
```

```
      SUBROUTINE HRB(IM,IT,IP)
C
C  HRB CALCULATES MESH SPACING, H, DENSITIES, RZ AND R, AT GIVEN AND
C  ADJACENT POINTS, AND STREAM SHEET THICKNESSES, BZ AND B, AT GIVEN
C  AND ADJACENT POINTS
C
      COMMON /AUKRHO/ A(2000,4),U(2000),K(2000),RHO(2000)
      COMMON /CALCOV/ MBIM1,MBIP1,MBOM1,MBOP1,MBI2M1,MBI2P1,MBO2M1,
     1MBO2P1,MMM1,HM1,HM2,HM3,HT,DTLR,DMLR,PITCH,CP,EXPON,TWW,CPTIP,
     2TGROG,TBI,TBO,LAMBDA,TWL,ITMIN,ITMAX,NIP,IMS(4),BV(4),MV(100),
     3IV(101),ITV(100,4),TV(100,4),DTDMV(100,4),BETAV(100,4),
     4MH(100,4),DTDMH(100,4),BETAH(100,4),RMH(100,4),BEH(100,4),
     5RM(100),BE(100),DBDM(100),SAL(100),AAA(100)
      COMMON /HRBAAK/ H(4),R(4),B(4),KAK(4),KA(4),IH(4),RZ,BZ
      INTEGER BLDAT,AANDK,ERSOR,STRFN,SLCRD,SURVL,AATEMP,SURF,SURFBV,
     1FIRST,UPPER,UPPRBV,SI,ST,SRW
      REAL K,KAK,LAMBDA,LMAX,MH,MLE,MR,MSL,MSP,MV,MVIM1
      H(1)= HT*RM(IM)
      H(2)= HT*RM(IM)
      H(3)= MV(IM)-MV(IM-1)
      H(4)= MV(IM+1)-MV(IM)
      RZ = RHO(IP)
      IP3 = IPF(IM-1,IT)
      IP4 = IPF(IM+1,IT)
      R(1) = RHO(IP-1)
      R(2) = RHO(IP+1)
      R(3) = RHO(IP3)
      R(4) = RHO(IP4)
      BZ= BE(IM)
      B(3)= BE(IM-1)
      B(4)= BE(IM+1)
      RETURN
      END
```

```
      SUBROUTINE AAK(IM,IP)
C
C   AAK CALCULATES FINITE DIFFERENCE COEFFICIENTS, A, AND CONSTANT, K,
C   AT A SINGLE MESH POINT
C
      COMMON /AUKRHO/ A(2000,4),U(2000),K(2000),RHU(2000)
      COMMON /CALCON/ MBIM1,MBIP1,MBOM1,MBOP1,MBI2M1,MBI2P1,MBO2M1,
     1MBO2P1,MMM1,HM1,HM2,HM3,HT,DTLR,DMLR,PITCH,CP,EXPON,TWW,CPTIP,
     2TGROG,TBI,TBO,LAMBDA,TWL,ITMIN,ITMAX,NIP,IMS(4),BV(4),MV(100),
     3IV(101),ITV(100,4),TV(100,4),DTDMV(100,4),BETAV(100,4),
     4MH(100,4),DTDMH(100,4),BETAH(100,4),RMH(100,4),BEH(100,4),
     5RM(100),BE(100),DBDM(100),SAL(100),AAA(100)
      COMMON /HRBAAK/ H(4),R(4),B(4),KAK(4),KA(4),IH(4),RZ,BZ
      INTEGER BLDAT,AANDK,ERSOR,STRFN,SLCRD,SURVL,AATEMP,SURF,SURFBV,
     1FIRST,UPPER,UPPRBV,S1,ST,SRW
      REAL K,KAK,LAMBDA,LMAX,MH,MLE,MR,MSL,MSP,MV,MVIM1
      A12= 2./H(1)/H(2)
      A34= 2./H(3)/H(4)
      AZ= A12+A34
      B12= (R(2)-R(1))/RZ/(H(1)+H(2))
      B34= (B(4)*R(4)-B(3)*R(3))/BZ/RZ/(H(3)+H(4))-SAL(IM)/RM(IM)
      A(IP,1) = (2./H(1)+B12)/AZ/(H(1)+H(2))
      A(IP,2) = A12/AZ-A(IP,1)
      A(IP,3) = (2./H(3)+B34)/AZ/(H(3)+H(4))
      A(IP,4) = A34/AZ-A(IP,3)
      K(IP) = -TWW*BZ*RZ*SAL(IM)/AZ
      RETURN
      END
```

```
      SUBROUTINE BDRY12(I,IM,IT,SURF,SURFBV)
C
C   BDRY12 CORRECTS VALUES COMPUTED BY HRB WHEN A VERTICAL MESH LINE
C   INTERSECTS A BLADE
C
      COMMON /CALCON/ MBIM1,MBIP1,MBOM1,MBOP1,MBI2M1,MBI2P1,MBO2M1,
     1MBO2P1,MMM1,HM1,HM2,HM3,HT,DTLR,DMLR,PITCH,CP,EXPON,TWW,CPTIP,
     2TGROG,TBI,TBO,LAMBDA,TWL,ITMIN,ITMAX,NIP,IMS(4),BV(4),MV(100),
     3IV(101),ITV(100,4),TV(100,4),DTDMV(100,4),BETAV(100,4),
     4MH(100,4),DTDMH(100,4),BETAH(100,4),RMH(100,4),BEH(100,4),
     5RM(100),BE(100),DBDM(100),SAL(100),AAA(100)
      COMMON /RHOS/ RHOHB(100,4),RHOVB(100,4)
      COMMON /HRBAAK/ H(4),R(4),B(4),KAK(4),KA(4),IH(4),RZ,BZ
      INTEGER BLDAT,AANDK,ERSOR,STRFN,SLCRD,SURVL,AATEMP,SURF,SURFBV,
     1FIRST,UPPER,UPPRBV,S1,ST,SRW
      REAL K,KAK,LAMBDA,LMAX,MH,MLE,MR,MSL,MSP,MV,MVIM1
      H(I) = ABS(FLOAT(IT)*HT-TV(IM,SURF))*RM(IM)
      R(I)= RHOVB(IM,SURF)
      KAK(I)=BV(SURFBV)
      KA(I)=1
      RETURN
      END
```

```
      SUBROUTINE BDRY34(I,IM,SURF,SURFBV)
C
C   BDRY34 CORRECTS VALUES COMPUTED BY HRB WHEN A HORIZONTAL MESH LINE
C   INTERSECTS A BLADE
C
      COMMON /CALCON/ MBIM1,MBIP1,MBOM1,MBOP1,MBI2M1,MBI2P1,MBO2M1,
     1MBO2P1,MMM1,HM1,HM2,HM3,HT,DTLR,DMLR,PITCH,CP,EXPON,TWW,CPTIP,
     2TGROG,TBI,TBO,LAMBDA,TWL,ITMIN,ITMAX,NIP,IMS(4),BV(4),MV(100),
     3IV(101),ITV(100,4),TV(100,4),DTDMV(100,4),BETAV(100,4),
     4MH(100,4),DTDMH(100,4),BETAH(100,4),RMH(100,4),BEH(100,4),
     5RM(100),BE(100),DBDM(100),SAL(100),AAA(100)
      COMMON /RHOS/ RHOHB(100,4),RHOVB(100,4)
      COMMON /HRBAAK/ H(4),R(4),B(4),KAK(4),KA(4),IH(4),RZ,BZ
      INTEGER BLDAT,AANDK,ERSOR,STRFN,SLCRD,SURVL,AATEMP,SURF,SURFBV,
     1FIRST,UPPER,UPPRBV,S1,ST,SRW
      REAL K,KAK,LAMBDA,LMAX,MH,MLE,MR,MSL,MSP,MV,MVIM1
      IH(SURF)=IH(SURF)+1
      IHS=IH(SURF)
      H(I)=ABS(MV(IM)-MH(IHS,SURF))
      R(I)=RHOHB(IHS,SURF)
      B(I)=BEH(IHS,SURF)
      KAK(I)=BV(SURFBV)
      KA(I)=1
      RETURN
      END
```

```
      SUBROUTINE SOR
C
C   SOR SOLVES THE SET OF SIMULTANEOUS EQUATIONS FOR THE STREAM FUNCTION
C   USING THE METHOD OF SUCCESSIVE OVER-RELAXATION
C
      COMMON /AUKRHO/ A(2000,4),U(2000),K(2000),RHO(2000)
      COMMON /INP/ GAM,AR,TIP,RHOIP,WTFL,WTFLSP,OMEGA,ORF,BETAI,BETAO,
     1MBI,MBO,MBI2,MBO2,MM,NBBI,NBL,NRSP,MR(50),RMSP(50),BESP(50),
     2BLDAT,AANDK,ERSOR,STRFN,SLCRD,INTVL,SURVL
      COMMON /CALCON/ MBIM1,MBIP1,MBOM1,MBOP1,MBI2M1,MBI2P1,MBO2M1,
     1MBO2P1,MMM1,HM1,HM2,HM3,HT,DTLR,DMLR,PITCH,CP,EXPON,TWW,CPTIP,
     2TGROG,TBI,TBO,LAMBDA,TWL,ITMIN,ITMAX,NIP,IMS(4),BV(4),MV(100),
     3IV(101),ITV(100,4),TV(100,4),DTDMV(100,4),BETAV(100,4),
     4MH(100,4),DTDMH(100,4),BETAH(100,4),RMH(100,4),BEH(100,4),
     5RM(100),BE(100),DBDM(100),SAL(100),AAA(100)
      INTEGER BLDAT,AANDK,ERSOR,STRFN,SLCRD,SURVL,AATEMP,SURF,SURFBV,
     1FIRST,UPPER,UPPRBV,S1,ST,SRW
      REAL K,KAK,LAMBDA,LMAX,MH,MLE,MR,MSL,MSP,MV,MVIM1
      AATEMP = AANDK
      IF (ORF.GE.2.) ORF=0.
      IF (ORF.GT.1.) GO TO 20
      ORF = 1.
      ORFOPT= 2.
   10 ORFTEM=ORFOPT
      LMAX = 0.
   20 IF (AATEMP.GT.0) WRITE(6,1010)
      ERROR = 0.
C
C   SOLVE MATRIX EQUATION BY SOR, OR CALCULATE OPTIMUM OVERRELAXATION
C   FACTOR
C
      IP = 0
      DO 100 IM=1,MM
      IPU = IV(IM)
      IPL = IV(IM+1)-1
      IT = ITV(IM,1)
      IF (IM.GT.MBO) IJ=ITV(IM,3)
      IF (AATEMP.GT.0) WRITE(6,1020) IM,IT
      DO 100 IP=IPU,IPL
      IF(IT.GT.ITV(IM,4).AND.IT.LE.ITV(IM,3)) IT = IT+ITV(IM,3)
     1-ITV(IM,4)-1
```

```
      IP1 =  IP-1
      IP2 =  IP+1
C   CORRECT IP1 AND IP2 ALONG PERIODIC BOUNDARIES
      IF(IM.GE.MBI.AND.IM.LE.MBO2.AND.(IM.LE.MBO.OR.IM.GE.MBI2))GO TO 30
      IF(IT.EQ.ITV(IM,1).OR.IT.EQ.ITV(IM,3)) IP1 = IP1+NBBI
      IF(IT.EQ.ITV(IM,2).OR.IT.EQ.ITV(IM,4)) IP2 = IP2-NBBI
   30 IT3 = IT
      IT4 = IT
C   CORRECT IT3 AND IT4 ALONG LINES C-D AND K-L
      IF(IM.NE.MBOP1) GO TO 40
      IF(IT.LT.ITV(IM-1,1)) IT3 = IT+NBBI
   40 IF(IM.NE.MBO) GO TO 70
      IF(MBI2-MBO) 50,50,60
   50 IF(IT.GE.ITV(IM,3)) IT4 = IT-NBBI
      GO TO 70
   60 IF (IT.GT.ITV(IM+1,4)) IT4 = IT-NBBI
   70 IP3 =  IPF(IM-1,IT3)
      IP4 =  IPF(IM+1,IT4)
      IF (ORF.GT.1.) GO TO 80
C   CALCULATE NEW ESTIMATE FOR LMAX
      UNEW = A(IP,1)*U(IP1)+A(IP,2)*U(IP2)+A(IP,3)*U(IP3)+A(IP,4)*U(IP4)
      IF (UNEW.LT.1.E-25) U(IP) = 0.
      IF (U(IP).EQ.0.) GO TO 90
      RATIO = UNEW/U(IP)
      LMAX= AMAX1(RATIO,LMAX)
      U(IP) = UNEW
      GO TO 90
C   CALCULATE NEW ESTIMATE FOR STREAM FUNCTION BY SOR
   80 CHANGE = ORF*(K(IP)-U(IP)+A(IP,1)*U(IP1)+A(IP,2)*U(IP2)+A(IP,3)*
     1U(IP3)+A(IP,4)*U(IP4))
      ERROR= AMAX1(ERROR,ABS(CHANGE))
      U(IP) = U(IP)+CHANGE
   90 IF(AATEMP.LE.0) GO TO 100
      WRITE (6,1030) IT,IP,IP1,IP2,IP3,IP4,(A(IP,I),I=1,4),K(IP)
  100 IT = IT+1
      AATEMP = 0
      IF(ORF.GT.1.) GO TO 110
      ORFOPT= 2./(1.+SQRT(ABS(1.-LMAX)))
      WRITE(6,1040) ORFOPT
      IF(ORFTEM-ORFOPT.GT..00001.OR.ORFOPT.GT.1.999) GO TO 10
      WRITE (6,1000)
      ORF = ORFOPT
      GO TO 20
  110 IF(ERSOR.GT.0) WRITE(6,1050) ERROR
      IF(ERROR.GT..000001) GO TO 20
      IF(STRFN.LE.0) RETURN
C
C   PRINT STREAM FUNCTION VALUES FOR THIS ITERATION
C
      WRITE (6,1060)
      MBIT = MINO(MBO,MBI2M1)
      DO 120 IM =1,MBIT
      IPU =  IV(IM)
      IPL =  IV(IM+1)-1
      ITVU =  ITV(IM,1)
      WRITE (6,1020) IM,ITVU
  120 WRITE (6,1070) (U(IP),IP=IPU,IPL)
      IF(MBI2.GT.MBO) GO TO 140
      DO 130 IM=MBI2,MBO
      IPU =  IV(IM)
      IPL =  IV(IM)+ITV(IM,4)-ITV(IM,1)
      ITVU = ITV(IM,1)
      WRITE (6,1020) IM,ITVU
      WRITE (6,1070) (U(IP),IP=IPU,IPL)
      IPU =  IPL+1
      IPL =  IV(IM+1)-1
      IF(IPU.GT.IPL) GO TO 130
      ITVU = ITV(IM,3)
      WRITE (6,1020) IM,ITVU
      WRITE (6,1070) (U(IP),IP=IPU,IPL)
  130 CONTINUE
  140 DO 150 IM=MBOP1,MM
      IPU =  IV(IM)
      IPL =  IV(IM+1)-1
```

```
      ITVU = ITV(IM,3)
      WRITE (6,1020) IM,ITVU
  150 WRITE (6,1070) (U(IP),IP=IPU,IPL)
      RETURN
 1000 FORMAT (1H1)
 1010 FORMAT (82H1  IT    IP     IP1    IP2    IP3    IP4     A(1)        A(2)
     1    A(3)       A(4)          K)
 1020 FORMAT(5H IM =,I4,6X,6HIT1 = ,I4)
 1030 FORMAT(1X,I4,5I6,5F10.5)
 1040 FORMAT(24H ESTIMATED OPTIMUM ORF =,F9.6)
 1050 FORMAT(8H ERROR =,F11.8)
 1060 FORMAT(1H1,10X,22HSTREAM FUNCTION VALUES)
 1070 FORMAT (2X,10F13.8)
      END
```

```
      SUBROUTINE SLAX
C
C  SLAX CALLS SUBROUTINES TO CALCULATE RHO*W-SUB-M THROUGHOUT THE REGION
C  AND ON THE BLADE SURFACES, AND TO CALCULATE AND PLOT THE
C  STREAMLINE LOCATIONS
C
      COMMON /AUKRHO/ A(2000,4),U(2000),K(2000),RHO(2000)
      COMMON /INP/ GAM,AR,TIP,RHOIP,WTFL,WTFLSP,OMEGA,ORF,BETAI,BETAO,
     1MBI,MBO,MBI2,MBO2,MM,NBBI,NBL,NRSP,MR(50),RMSP(50),BESP(50),
     2BLDAT,AANDK,ERSOR,STRFN,SLCRD,INTVL,SURVL
      COMMON /CALCON/ MBIM1,MBIP1,MBOM1,MBOP1,MBI2M1,MBI2P1,MBO2M1,
     1MBO2P1,MMM1,HM1,HM2,HM3,HT,DTLR,DMLR,PITCH,CP,EXPON,TWW,CPTIP,
     2TGROG,TBI,TBO,LAMBDA,TWL,ITMIN,ITMAX,NIP,IMS(4),BV(4),MV(100),
     3IV(101),ITV(100,4),TV(100,4),DTDMV(100,4),BETAV(100,4),
     4MH(100,4),DTDMH(100,4),BETAH(100,4),RMH(100,4),BEH(100,4),
     5RM(100),BE(100),DBDM(100),SAL(100),AAA(100)
      COMMON /SLA/ TSL(800),UINT(8)
      DIMENSION MSL(100),KKK(18),P(4)
      DIMENSION W(2000),RWM(2000),BETA(2000),WMB(100,4),WTB(100,4),
     1XDOWN(800),YACROS(800)
      EQUIVALENCE (A(1,1),W(1)),(A(1,2),RWM(1)),(A(1,3),BETA(1)),
     1(A(1,4),WMB(1)),(A(401,4),WTB(1)),(A(801,4),XDOWN(1)),
     2(K(1),YACROS(1))
      INTEGER BLDAT,AANDK,ERSOR,STRFN,SLCRD,SURVL,AATEMP,SURF,SURFBV,
     1FIRST,UPPER,UPPRBV,S1,ST,SRW
      REAL K,KAK,LAMBDA,LMAX,MH,MLE,MR,MSL,MSP,MV,MVIM1
      DATA (KKK(J),J=4,18,2)/8*1H*/
C
C  CALL SLAVP AND SLAVBB THROUGHOUT THE REGION
C
      ITVU= ITV(1,1)
      ITVL= ITV(1,2)
      DO 10 IM=1,MBIM1
   10 CALL SLAVP(IM,ITVU,ITVL)
      MBOT= MINO(MBO,MBI2M1)
      IF (MBI.GT.MBOT) GO TO 30
      DO 20 IM=MBI,MBOT
      I= 0
   20 CALL SLAVBB(IM,1,2,1,I)
   30 IF (MBOP1.GT.MBI2M1) GO TO 50
      ITVU= ITV(MBOP1,3)
      ITVL= ITV(MBOP1,4)
      DO 40 IM=MBOP1,MBI2M1
   40 CALL SLAVP(IM,ITVU,ITVL)
   50 IF (MBI2.GT.MBO) GO TO 70
      DO 60 IM=MBI2,MBO
      I= 0
      CALL SLAVBB(IM,1,4,1,I)
   60 CALL SLAVBB(IM,3,2,4,I)
   70 MBOT= MAXO(MBOP1,MBI2)
      IF (MBOT.GT.MBO2) GO TO 90
      DO 80 IM= MBOT,MBO2
      I= 0
```

```
   80 CALL SLAVBB(IM,3,4,3,I)
   90 ITVU= ITV(MBO2P1,3)
      ITVL= ITV(MBO2P1,4)
      DO 100 IM=MBO2P1,MM
  100 CALL SLAVP(IM,ITVU,ITVL)
C
C  PLOT STREAMLINES
C
      IF (SLCRD.LE.0) RETURN
      DO 110 IM=1,MM
  110 MSL(IM) = MV(IM)
      KKK(1) = 7
      KKK(2) = 8
      KKK(3) = MM
      P(1) = 1.
      P(3) = 0.
      P(4) = 0.
      WRITE(6,1000)
      CALL PLOTMY(MSL,TSL,KKK,P)
      WRITE(6,1010)
      RETURN
 1000 FORMAT (2HPT,50X,16HSTREAMLINE PLOTS
 1010 FORMAT (2HPL,40X,70HSTREAMLINES ARE PLOTTED WITH THETA ACROSS THE
     1PAGE AND M DOWN THE PAGE)
      END
```

```
      SUBROUTINE SLAV
C
C  SLAV CALCULATES RHO*W-SUB-M THROUGHOUT THE REGION AND ON THE BLADE
C  SURFACES, AND CALCULATES THE STREAMLINE LOCATIONS
C
      COMMON SRW,ITER,IEND,LER(2),NER(2)
      COMMON /AUKRHO/ A(2000,4),U(2000),K(2000),RHO(2000)
      COMMON /INP/ GAM,AR,TIP,RHOIP,WTFL,WTFLSP,OMEGA,ORF,BETAI,BETAO,
     1MBI,MBO,MBI2,MBO2,MM,NBBI,NBL,NRSP,MR(50),RMSP(50),BESP(50),
     2BLDAT,AANDK,ERSUR,STRFN,SLCRD,INTVL,SURVL
      COMMON /CALCON/ MBIM1,MBIP1,MBOM1,MBOP1,MBI2M1,MBI2P1,MBO2M1,
     1MBO2P1,MMM1,HM1,HM2,HM3,HT,DTLR,DMLR,PITCH,CP,EXPON,TWW,CPTIP,
     2TGROG,TBI,TBO,LAMBDA,TWL,ITMIN,ITMAX,NIP,IMS(4),BV(4),MV(100),
     3IV(101),ITV(100,4),TV(100,4),DTDMV(100,4),BETAV(100,4),
     4MH(100,4),DTDMH(100,4),BETAH(100,4),RMH(100,4),BEH(100,4),
     5RM(100),BE(100),DBDM(100),SAL(100),AAA(100)
      COMMON /SLA/ TSL(800),UINT(8)
      DIMENSION TSP(50),USP(50),DUDT(50),TINT(8)
      DIMENSION W(2000),RWM(2000),BETA(2000),WMB(100,4),WTB(100,4),
     1XDOWN(800),YACROS(800)
      EQUIVALENCE (A(1,1),W(1)),(A(1,2),RWM(1)),(A(1,3),BETA(1)),
     1(A(1,4),WMB(1)),(A(401,4),WTB(1)),(A(801,4),XDOWN(1)),
     2(K(1),YACROS(1))
C
C  SLAVP CALCULATES ALONG VERTICAL MESH LINES WHICH DO NOT
C  INTERSECT BLADES
C
      ENTRY SLAVP(IM,ITVU,ITVL)
      LOC= 0
      I1 = 0
      NI= 8
      NSP= ITVL-ITVU+2
      IP = IV(IM)-1
      DO 10 IT=1,NSP
      IP = IP+1
      TSP(IT) = FLOAT(IT+ITVU-1)*HT
   10 USP(IT)= U(IP)
      USP(NSP) = USP(1)+1.
      IP = IV(IM)
      INTU = INT(U(IP)*5.)
      IF (U(IP).GT.0.) INTU=INTU+1
```

```
      DO 20 J=1,5
      UINT(J) = FLOAT(INTU)/5.
   20 INTU = INTU+1
      UINT(6)= BV(4)
   30 IF (UINT(6).GE.U(IP)) GO TO 40
      UINT(6)= UINT(6)+1.
      GO TO 30
   40 IF (UINT(6).LE.U(IP)+1.) GO TO 50
      UINT(6)= UINT(6)-1.
      GO TO 40
   50 UINT(7)= UINT(1)
      UINT(8)= UINT(1)
      GO TO 100
C
C  SLAVBB CALCULATES ALONG VERTICAL MESHLINES WHICH INTERSECT BLADES
C
      ENTRY SLAVBB(IM,UPPER,LOWER,UPPRBV,I)
      INTEGER BLDAT,AANDK,ERSOR,STRFN,SLCRD,SURVL,AATEMP,SURF,SURFBV,
     1FIRST,UPPER,UPPRBV,S1,ST,SRW
      REAL K,KAK,LAMBDA,LMAX,MH,MLE,MR,MSL,MSP,MV,MVIM1
      LOC= 1
      ITVUP1 = ITV(IM,UPPER)
      ITVLM1 = ITV(IM,LOWER)
      ITVU = ITVUP1-1
      ITVL = ITVLM1+1
      NSP = ITVL-ITVU+1
      TSP(1) = TV(IM,UPPER)
      TSP(NSP) = TV(IM,LOWER)
      USP(1) = BV(UPPRBV)
      USP(NSP) = BV(LOWER)
      IP = IPF(IM,ITVUP1)-1
      NSPM1 = NSP-1
      IF(2.GT.NSPM1) GO TO 70
      DO 60 IT=2,NSPM1
      IP = IP+1
      TSP(IT) = FLOAT(IT+ITVU-1)*HT
   60 USP(IT) = U(IP)
   70 I1= I
      I= I1+1
      UINT(I)= BV(UPPRBV)
      INTU = INT(UINT(I)*5.)
      IF (UINT(I).GE.0.) INTU=INTU+1
   80 I = I+1
      UINT(I) = FLOAT(INTU)/5.
      INTU = INTU+1
      IF (UINT(I).LT.BV(LOWER)) GO TO 80
      UINT(I)= BV(LOWER)
      IF(LOWER-UPPER.NE.1) GO TO 90
      I = 7
      UINT(I) = BV(4)
   90 UINT(8) = BV(LOWER)
      NI= I-I1
      IF (NI.EQ.7) NI = 8
C
C  FOR BOTH SLAVP AND SLAVBB, CALCULATE RHO*W-SUB-M IN THE REGION, AND
C  RHO*W AT VERTICAL MESH LINE INTERSECTIONS ON THE BLADE SURFACES
C
  100 CALL SPLINE(TSP,USP,NSP,DUDT,AAA)
      FIRST= (1-LOC)*ITVU+LOC*ITVUP1
      LAST = (1-LOC)*ITVL+LOC*ITVLM1
      IF(FIRST.GT.LAST) GO TO 120
      IT = LOC
      IPU = IPF(IM,FIRST)
      IPL = IPF(IM,LAST)
      DO 110 IP=IPU,IPL
      IT = IT+1
  110 RWM(IP)  = DUDT(IT)*WTFL/BE(IM)/RM(IM)
  120 IF (LOC.EQ.0) GO TO 130
      WMB(IM,UPPER) = DUDT(  1)*WTFL/BE(IM)/RM(IM)
      WMB(IM,LOWER) = DUDT(NSP)*WTFL/BE(IM)/RM(IM)
      RMDTU2 = (RM(IM)*DTDMV(IM,UPPER))**2
      RMDTL2 = (RM(IM)*DTDMV(IM,LOWER))**2
      IF (RMDTU2.GT.10000.) WMB(IM,UPPER) = 0.
      IF (RMDTL2.GT.10000.) WMB(IM,LOWER) = 0.
```

```
      WMB(IM,UPPER) = ABS(WMB(IM,UPPER))*SQRT(1.+RMDTU2)
      WMB(IM,LOWER) = ABS(WMB(IM,LOWER))*SQRT(1.+RMDTL2)
  130 IF (SLCRD.LE.0) RETURN
      I1 = I1+1
      CALL SPLINT(USP,TSP,NSP,UINT(I1),NI,TINT(I1),AAA)
      I2 = NI+I1-1
      DO 140 J=I1,I2
      L= (J-1)*MM+IM
  140 TSL(L)= TINT(J)
      IF (UPPER.EQ.1.AND.LOWER.EQ.4) RETURN
      IF (IM.EQ.1) WRITE(6,1000)
      WRITE(6,1010) MV(IM),(UINT(J),TINT(J),J=1,8)
      RETURN
 1000 FORMAT(1H1////30X,22HSTREAMLINE COORDINATES////5X,12HM COORDINATE,
     13(7X,10HSTREAM FN.,10X,5HTHETA,4X)//)
 1010 FORMAT(1X,7G18.7/(19X,6G18.7))
      END
```

```
      SUBROUTINE TANG
C
C  TANG CALCULATES RHO*W-SUB-THETA AND THEN RHO*W THROUGHOUT THE REGION
C  AND ON THE BLADE SURFACES, AND CALCULATES THE VELOCITY ANGLE, BETA,
C  THROUGHOUT THE REGION
C
      COMMON SRW,ITER,IEND,LER(2),NER(2)
      COMMON /AUKRHO/ A(2000,4),U(2000),K(2000),RHO(2000)
      COMMON /INP/ GAM,AR,TIP,RHOIP,WTFL,WTFLSP,OMEGA,ORF,BETAI,BETAO,
     1MBI,MBO,MBI2,MBO2,MM,NBBI,NBL,NRSP,MR(50),RMSP(50),BESP(50),
     2BLDAT,AANDK,ERSOR,STRFN,SLCRD,INTVL,SURVL
      COMMON /CALCON/ MBIM1,MBIP1,MBOM1,MBOP1,MBI2M1,MBI2P1,MBO2M1,
     1MBO2P1,MMM1,HM1,HM2,HM3,HT,DTLR,DMLR,PITCH,CP,EXPON,TWW,CPTIP,
     2TGROG,IBI,TBO,LAMBDA,TWL,ITMIN,ITMAX,NIP,IMS(4),BV(4),MV(100),
     3IV(101),ITV(100,4),TV(100,4),DTDMV(100,4),BETAV(100,4),
     4MH(100,4),DTDMH(100,4),BETAH(100,4),RMH(100,4),BEH(100,4),
     5RM(100),BE(100),DBDM(100),SAL(100),AAA(100)
      DIMENSION SPM(100),USP(100),DUDM(100)
      DIMENSION W(2000),RWM(2000),BETA(2000),WMB(100,4),WTB(100,4),
     1XDOWN(800),YACROS(800)
      EQUIVALENCE (A(1,1),W(1)),(A(1,2),RWM(1)),(A(1,3),BETA(1)),
     1(A(1,4),WMB(1)),(A(401,4),WTB(1)),(A(801,4),XDOWN(1)),
     2(K(1),YACROS(1))
      INTEGER BLDAT,AANDK,ERSOR,STRFN,SLCRD,SURVL,AATEMP,SURF,SURFBV,
     1FIRST,UPPER,UPPRBV,S1,ST,SRW
      REAL K,KAK,LAMBDA,LMAX,MH,MLE,MR,MSL,MSP,MV,MVIM1
      LOGICAL ADDL,ADD
      EXTERNAL BL1,BL2,BL3,BL4
C
C  PERFORM CALCULATIONS ALONG ONE HORIZONTAL LINE AT A TIME
C
      IT = ITMIN
   10 IF (IT.GT.ITMAX) RETURN
      ADDL = .FALSE.
      ADD  = .FALSE.
      IT1 = IT
      S1 = 0
C
C  ON THE GIVEN HORIZONTAL MESH LINE, FIND A FIRST POINT IN THE REGION
C
      IF(IT.GE.0.AND.IT.LT.NBBI) GO TO 60
      IM = MBIM1
   20 IM = IM+1
      ITV1 = ITV(MBO,3)-NBBI
      IF(MBI2.GT.MBO) ITV1 = ITV(MBOP1,3)
      IF(IM.EQ.MBO.AND.IT.GE.ITV1.AND.IT.LE.ITV(MBO,2)-NBBI) GO TO 200
      IF(IM.GT.MBO2P1) GO TO 200
      DO 40 SURF=1,3,2
      IF (IM.GT.MBO.AND.SURF.EQ.1) GO TO 40
      ITVIM1 = ITV(IM-1,SURF)
```

```
      IF(SURF.EQ.3.AND.IM.EQ.MBOP1) ITVIM1 = ITVIM1-NBBI
      IF( ITI.GE.ITV(IM,SURF).AND.ITI.LT.ITVIM1) GO TO 70
   40 CONTINUE
      SURF = 1
      IF( IM.EQ.MBOP1.AND.IT.EQ.ITV(MBO,1)-1.AND.ITV(MBO,1)-ITV(MBO,2)
     1+NBBI.EQ.2) GO TO 70
      DO 50 SURF=2,4,2
      IF (IM.LE.MBI2.AND.SURF.EQ.4) GO TO 50
      IF (IT.LE.ITV(IM,SURF).AND.IT.GT.ITV(IM-1,SURF)) GO TO 70
   50 CONTINUE
      GO TO 20
C
C  FIRST POINT IS ON BOUNDARY A-N
C
   60 IM1= 1
      IM = 1
      SPM(1) = MV(1)
      USP (1) = U(IT+1)
      GO TO 80
C
C  FIRST POINT IS ON A BLADE SURFACE
C
   70 S1 = SURF
      ITI = IT
      ADD = .FALSE.
      IF (ADDL.AND.S1.EQ.3) ADD = .TRUE.
      IF (ADD) ITI=IT-NBBI
      IM1 = IM-1
      IM2 = IM
      TH = FLOAT(ITI)*HT
      IF (S1.EQ.3.AND.IM1.LT.MBO) TH = TH-FLOAT(NBBI)*HT
      MVIM1 = MV(IM1)
      IF (( IM.EQ.MBIP1.AND.(SURF.EQ.1.OR.SURF.EQ.2)).OR.(IM.EQ.MBI2P1
     1.AND.(SURF.EQ.3.OR.SURF.EQ.4))) MVIM1=MVIM1+(MV(IM2)-MVIM1)/1000.
      LER(2)=10
C     BLCD (VIA ROOT) CALL NO. 10
      IF (S1.EQ.1.AND.IM1.NE.MBO) CALL ROOT(MVIM1,MV(IM2),TH,BL1,DTLR,
     1ANS,AAA)
      LER(2)=11
C     BLCD (VIA ROOT) CALL NO. 11
      IF (S1.EQ.3.AND.IM1.NE.MBO2) CALL ROOT(MVIM1,MV(IM2),TH,BL3,DTLR,
     1ANS,AAA)
      LER(2)=12
C     BLCD (VIA ROOT) CALL NO. 12
      IF (S1.EQ.2) CALL ROOT(MVIM1,MV(IM2),TH,BL2,DTLR,ANS,AAA)
      LER(2)=13
C     BLCD (VIA ROOT) CALL NO. 13
      IF(S1.EQ.4) CALL ROOT(MVIM1,MV(IM2),TH,BL4,DTLR,ANS,AAA)
      IF(S1.EQ.1.AND.IM1.EQ.MBO) ANS = MV(MBO)
      IF(S1.EQ.3.AND.IM1.EQ.MBO2) ANS = MV(MBO2)
      SPM(IM1) = ANS
      USP(IM1)= BV(S1)
C
C  MOVE ALONG HORIZONTAL MESH LINE UNTIL MESH LINE INTERSECTS BOUNDARY
C
   80 ITI= IT
   90 ITV2 = ITV(MBO,3)
      IF (MBI2.GT.MBO) ITV2 = ITV(MBOP1,3)+NBBI
      IF (IM.EQ.MBOP1.AND.IT.GE.ITV2.AND.IT.LE.ITV(MBO,2)) ADD=.TRUE.
      IF (ADD) ADDL = .TRUE.
      IF (ADD) ITI=IT-NBBI
      IF (ADD.AND.S1.EQ.3) USP(IM1)=BV(S1)+1.
      IF (IM.LT.MBI.OR.IM.GT.MBO2) GO TO 120
      DO 100 SURF=1,3,2
      IF (IM.LE.MBI2.AND.SURF.EQ.3) GO TO 100
      IF (IM.EQ.IM2.AND.S1.EQ.4.AND.SURF.EQ.3) GO TO 100
      ITVIM1 = ITV(IM-1,SURF)
      IF (IM.EQ.MBOP1.AND.ADD) ITVIM1 = ITVIM1-NBBI
      IF (ITI.LT.ITV(IM,SURF).AND.ITI.GE.ITVIM1) GO TO 140
  100 CONTINUE
      SURF = 3
      IF( IM.EQ.MBI2.AND.IT.EQ.ITV(MBI2,3)-1.AND.ITV(MBI2,3)-ITV(MBI2,4)
     1.EQ.2) GO TO 140
      DO 110 SURF=2,4,2
```

```
      IF (IM.GT.MBO.AND.SURF.EQ.2) GO TO 110
      IF (IM.EQ.IM2.AND.S1.EQ.3.AND.SURF.EQ.4) GO TO 110
      ITVIM1 = ITV(IM-1,SURF)
      IF (IM.EQ.MBOP1.AND.ADD) ITVIM1 = ITVIM1-NBBI
      IF (ITI.GT.ITV(IM,SURF).AND.ITI.LE.ITVIM1) GO TO 140
  110 CONTINUE
  120 SPM(IM) = MV(IM)
      IP = IPF(IM,ITI)                                                    180
      USP(IM) = U(IP)
      IF (ADD) USP(IM) = USP(IM)+1.
      IF (IM.EQ.MM) GO TO 130
      IM= IM+1
      GO TO 90
C
C  FINAL POINT IS ON BOUNDARY G-H
C
  130 IMT = MM
      GO TO 150
C
C  FINAL POINT IS ON A BLADE SURFACE
C
  140 ST = SURF
      IMT=IM
      IMTM1= IMT-1
      TH = FLOAT(ITI)*HT
      IF (ST.EQ.3.AND.IMT.LE.MBO) TH = TH-FLOAT(NBBI)*HT
      MVIM1 = MV(IMTM1)
      IF ((IMTM1.EQ.MBI).AND.(ST.EQ.1.OR.ST.EQ.2))
     1   MVIM1 = MVIM1+(MV(IMT)-MVIM1)/1000.
      IF ((IMTM1.EQ.MBI2).AND.(ST.EQ.3.OR.ST.EQ.4).AND.
     1   (ITV(MBI2,3)-ITV(MBI2,4).EQ.2.OR.ITV(MBI2,4)-ITV(MBI2,3).EQ.
     2   NBBI-2)) MVIM1 = MVIM1+(MV(IMT)-MVIM1)/1000.
      LER(2)=14
C     BLCD (VIA ROOT) CALL NO. 14
      IF(ST.EQ.1.AND.IMT.NE.MBI)CALL ROOT(MVIM1,MV(IMT),TH,BL1,
     1DTLR,ANS,AAA)                                                       214
      LER(2)=15
C     BLCD (VIA ROOT) CALL NO. 15
      IF(ST.EQ.3.AND.IMT.NE.MBI2)CALL ROOT(MVIM1,MV(IMT),TH,BL3,
     1DTLR,ANS,AAA)                                                       218
      LER(2)=16
C     BLCD (VIA ROOT) CALL NO. 16
      IF(ST.EQ.2)CALL ROOT(MVIM1,MV(IMT),TH,BL2,DTLR,ANS,AAA)             222
      LER(2)=17
C     BLCD (VIA ROOT) CALL NO. 17
      IF(ST.EQ.4)CALL ROOT(MVIM1,MV(IMT),TH,BL4,DTLR,ANS,AAA)             226
      IF(ST.EQ.1.AND.IMT.EQ.MBI) ANS = MV(MBI)
      IF(ST.EQ.3.AND.IMT.EQ.MBI2) ANS = MV(MBI2)
      SPM(IMT) = ANS
      USP(IMT)= BV(ST)
      IF(ST.EQ.3.AND.IMT.LE.MBO) USP(IMT) = BV(4)
      IF(ADD) USP(IMT) = USP(IMT)+1.
      IF (S1.EQ.3.AND.ST.EQ.2) USP(IM1) = BV(S1)+1.
      IF (S1.EQ.3.AND.ST.EQ.3) USP(IM1) = BV(S1)+1.
C
C  CALCULATE RHO*W-SUB-THETA AND THEN RHO*W AND BETA IN THE REGION
C
  150 NSP= IMT-IM1+1
      CALL SPLINE(SPM(IM1),USP(IM1),NSP,DUDM(IM1),AAA(IM1))               258
      FIRST=1
      IF (IM1.NE.1) FIRST=IM2
      LAST= MM
      IF (IMT.NE.MM) LAST=IMTM1
      IF (FIRST.GT.LAST) GO TO 170
      ITI = IT
      IF (FIRST.GT.MBOP1.AND.ADD) ITI=IT-NBBI
      DO 160 I=FIRST,LAST
      IF (ADD.AND.I.EQ.MBOP1) ITI=IT-NBBI
      RWT = -DUDM(I)*WTFL/BE(I)
      IP = IPF(I,ITI)                                                     279
      W(IP) = SQRT(RWT**2+RWM(IP)**2)                                     283
  160 BETA(IP) = ATAN2(RWT,RWM(IP))*57.295779
C
C  CALCULATE RHO*W ON THE BLADE SURFACES
C                                                                         286
  170 IF (IM1.EQ.1) GO TO 180
      CALL SEARCH (SPM(IM1),S1,IHS)                                       295
      ANS = -DUDM(IM1)*WTFL/BEH(IHS,S1)
      WTB(IHS,S1) = ABS(ANS)*SQRT(1.+1./(RMH(IHS,S1)*DTDMH(IHS,S1))**2)   301
  180 IF(IMT.EQ.MM) GO TO 200
      CALL SEARCH (SPM(IMT),ST,IHS)                                       307
      ANS = -DUDM(IMT)*WTFL/BEH(IHS,ST)
      WTB(IHS,ST) = ABS(ANS)*SQRT(1.+1./(RMH(IHS,ST)*DTDMH(IHS,ST))**2)   313
  190 GO TO 20
  200 IT = IT+1
      GO TO 10
      END
```

```
      SUBROUTINE SEARCH (DIST,SURF,IS)
C
C  SEARCH LOCATES THE POSITION OF A GIVEN VALUE OF M IN THE MH ARRAY
C
      COMMON /CALCON/ MBIM1,MBIP1,MBOM1,MBOP1,MBI2M1,MBI2P1,MBO2M1,
     1MBO2P1,MMM1,HM1,HM2,HM3,HT,DTLR,DMLR,PITCH,CP,EXPON,TWW,CPTIP,
     2TGROG,TBI,TBO,LAMBDA,TWL,ITMIN,ITMAX,NIP,IMS(4),BV(4),MV(100),
     3IV(101),ITV(100,4),TV(100,4),DTDMV(100,4),BETAV(100,4),
     4MH(100,4),DTDMH(100,4),BETAH(100,4),RMH(100,4),BEH(100,4),
     5RM(100),BE(100),DBDM(100),SAL(100),AAA(100)
      INTEGER BLDAT,AANDK,ERSOR,STRFN,SLCRD,SURVL,AATEMP,SURF,SURFBV,
     1FIRST,UPPER,UPPRBV,S1,ST,SRW
      REAL K,KAK,LAMBDA,LMAX,MH,MLE,MR,MSL,MSP,MV,MVIM1
      DO 10 I=1,100
      IF (ABS(MH(I,SURF)-DIST).GT.DMLR) GO TO 10
      IS = I
      RETURN
   10 CONTINUE
      WRITE (6,1000) DIST,SURF
      STOP
 1000 FORMAT (38HL SEARCH CANNOT FIND M IN THE MH ARRAY/7H DIST =,G14.6,
     110X,6HSURF =,G14.6)
      END
```

```
      SUBROUTINE VELOCY
C
C  VELOCY CALLS SUBROUTINES TO CALCULATE DENSITIES AND VELOCITIES
C  THROUGHOUT THE REGION AND ON THE BLADE SURFACES, AND IT PLOTS
C  THE SURFACE VELOCITIES
C
      COMMON /AUKRHO/ A(2000,4),U(2000),K(2000),RHO(2000)
      COMMON /INP/ GAM,AR,TIP,RHOIP,WTFL,WTFLSP,OMEGA,ORF,BETAI,BETAO,
     1MBI,MBO,MBI2,MBO2,MM,NBBI,NBL,NRSP,MR(50),RMSP(50),BESP(50),
     2BLDAT,AANDK,ERSOR,STRFN,SLCRD,INTVL,SURVL
      COMMON /CALCON/ MBIM1,MBIP1,MBOM1,MBOP1,MBI2M1,MBI2P1,MBO2M1,
     1MBO2P1,MMM1,HM1,HM2,HM3,HT,DTLR,DMLR,PITCH,CP,EXPON,TWW,CPTIP,
     2TGROG,TBI,TBO,LAMBDA,TWL,ITMIN,ITMAX,NIP,IMS(4),BV(4),MV(100),
     3IV(101),ITV(100,4),TV(100,4),DTDMV(100,4),BETAV(100,4),
     4MH(100,4),DTDMH(100,4),BETAH(100,4),RMH(100,4),BEH(100,4),
     5RM(100),BE(100),DBDM(100),SAL(100),AAA(100)
      DIMENSION KKK(18)
      DIMENSION W(2000),RWM(2000),BETA(2000),WMB(100,4),WTB(100,4),
     1XDOWN(800),YACROS(800)
      EQUIVALENCE (A(1,1),W(1)),(A(1,2),RWM(1)),(A(1,3),BETA(1)),
     1(A(1,4),WMB(1)),(A(401,4),WTB(1)),(A(801,4),XDOWN(1)),
     2(K(1),YACROS(1))
      INTEGER BLDAT,AANDK,ERSOR,STRFN,SLCRD,SURVL,AATEMP,SURF,SURFBV,
     1FIRST,UPPER,UPPRBV,S1,ST,SRW
      REAL K,KAK,LAMBDA,LMAX,MH,MLE,MR,MSL,MSP,MV,MVIM1
      DATA KKK(4)/1H*/,KKK(6)/1HO/,KKK(8)/1H=/,KKK(10)/1H(/,
     1KKK(12)/1H+/,KKK(14)/1HX/,KKK(16)/1H$/,KKK(18)/1H)/
C
C  CALL VELP, VELBB, AND VELSUR THROUGHOUT THE REGION
C
      CALL VELP(1,MBIM1,1,2)
      IF (MBI2.GT.MBO) GO TO 10
      CALL VELBB(MBI,MBI2M1,1,2)
      CALL VELBB(MBI2,MBO,1,4)
      CALL VELBB(MBI2,MBO,3,2)
      CALL VELBB(MBOP1,MBO2,3,4)
      GO TO 20
   10 CALL VELBB(MBI,MBO,1,2)
      CALL VELP(MBOP1,MBI2M1,3,4)
      CALL VELBB(MBI2,MBO2,3,4)
   20 CALL VELP(MBO2P1,MM,3,4)
      CALL VELSUR
C
C  PREPARE INPUT ARRAYS FOR PLOT OF VELOCITIES
C
      IF (SURVL.LE.0) RETURN
      NP2 = 0
C  SURFACES 1 TO 4 - TANGENTIAL COMPONENTS
      DO 50 SURF=1,4
      NP1 = NP2
```

```
      IMSS = IMS(SURF)
      IF(IMSS.LT.1) GO TO 40
      DO 30 IHS=1,IMSS
      IF (ABS(DTDMH(IHS,SURF)*RMH(IHS,SURF)).LT..57735) GO TO 30
      NP1 = NP1+1
      YACROS(NP1) = WTB(IHS,SURF)
      XDOWN(NP1) = MH(IHS,SURF)
   30 CONTINUE
   40 KKK(2*SURF+1) = NP1-NP2
   50 NP2 = NP1
C    SURFACES 1 AND 2 - MERIDIONAL COMPONENTS
      DO 80 SURF=1,2
      NP1 = NP2
      IF (MBIP1.GT.MBOM1) GO TO 70
      DO 60 IM=MBIP1,MBOM1
      IF (ABS(DTDMV(IM,SURF)*RM(IM)).GT.1.7321) GO TO 60
      NP1 = NP1+1
      YACROS(NP1) = WMB(IM,SURF)
      XDOWN(NP1) = MV(IM)
   60 CONTINUE
   70 KKK(2*SURF+9) = NP1-NP2
   80 NP2 = NP1
C    SURFACES 3 AND 4 - MERIDIONAL COMPONENTS
      DO 110 SURF=3,4
      NP1 = NP2
      IF(MBI2P1.GT.MBO2M1) GO TO 100
      DO 90 IM=MBI2P1,MBO2M1
      IF (ABS(DTDMV(IM,SURF)*RM(IM)).GT.1.7321) GO TO 90
      NP1 = NP1+1
      YACROS(NP1) = WMB(IM,SURF)
      XDOWN(NP1) = MV(IM)
   90 CONTINUE
  100 KKK(2*SURF+9) = NP1-NP2
  110 NP2 = NP1
C
C  PLOT VELOCITIES
C
      KKK(1) = 1
      KKK(2) = 8
      P = 5.
      WRITE(6,1000)
      CALL PLOTMY(XDOWN,YACROS,KKK,P)
      WRITE(6,1010)
      RETURN
 1000 FORMAT(2HPT,50X,24HBLADE SURFACE VELOCITIES)
 1010 FORMAT (2HPL,37X,63HVELOCITY(W) VS. MERIDIONAL STREAMLINE DISTANCE
     1(M) DOWN THE PAGE  /2HPL/
     22HPL,50X,50H+ - BLADE SURFACE 1, BASED ON MERIDIONAL COMPONENT/
     32HPL,50X,50H* - BLADE SURFACE 1, BASED ON TANGENTIAL COMPONENT/
     42HPL,50X,50HX - BLADE SURFACE 2, BASED ON MERIDIONAL COMPONENT/
     52HPL,50X,50HO - BLADE SURFACE 2, BASED ON TANGENTIAL COMPONENT/
     62HPL,50X,50H$ - BLADE SURFACE 3, BASED ON MERIDIONAL COMPONENT/
     72HPL,50X,50H= - BLADE SURFACE 3, BASED ON TANGENTIAL COMPONENT/
     82HPL,50X,50H) - BLADE SURFACE 4, BASED ON MERIDIONAL COMPONENT/
     92HPL,50X,50H( - BLADE SURFACE 4, BASED ON TANGENTIAL COMPONENT)
      END
```

```
      SUBROUTINE VEL
C
C  VEL CALCULATES DENSITIES AND VELOCITIES FROM THE PRODUCT OF
C  DENSITY TIMES VELOCITY
C
      COMMON SRW,ITER,IEND,LER(2),NER(2)
      COMMON /AUKRHO/ A(2000,4),U(2000),K(2000),RHO(2000)
      COMMON /INP/ GAM,AR,TIP,RHOIP,WTFL,WTFLSP,OMEGA,ORF,BETAI,BETAO,
     1MBI,MBO,MBI2,MBO2,MM,NBBI,NBL,NRSP,MR(50),RMSP(50),BESP(50),
     2BLDAT,AANDK,ERSOR,STRFN,SLCRD,INTVL,SURVL
```

```
      COMMON /CALCON/ MBIM1,MBIP1,MBOM1,MBOP1,MBI2M1,MBI2P1,MBO2M1,
     1MBO2P1,MMM1,HM1,HM2,HM3,HT,DTLR,DMLR,PITCH,CP,EXPON,TWW,CPTIP,
     2TGROG,TBI,TBO,LAMBDA,TWL,ITMIN,ITMAX,NIP,IMS(4),BV(4),MV(100),
     3IV(101),ITV(100,4),TV(100,4),DTDMV(100,4),BETAV(100,4),
     4MH(100,4),DTDMH(100,4),BETAH(100,4),RMH(100,4),BEH(100,4),
     5RM(100),BE(100),DBDM(100),SAL(100),AAA(100)
      COMMON /RHOS/ RHOHB(100,4),RHOVB(100,4)
      DIMENSION WWCRM(100,4),WWCRT(100,4),SURFL(100,4)
      DIMENSION W(2000),RWM(2000),BETA(2000),WMB(100,4),WTB(100,4),
     1XDOWN(800),YACROS(800)
      EQUIVALENCE (A(1,1),W(1)),(A(1,2),RWM(1)),(A(1,3),BETA(1)),
     1(A(1,4),WMB(1)),(A(401,4),WTB(1)),(A(801,4),XDOWN(1)),
     2(K(1),YACROS(1))
C
C  VELP CALCULATES ALONG VERTICAL MESH LINES WHICH DO NOT
C  INTERSECT BLADES
C
      ENTRY VELP(FIRST,LAST,UPPER,LOWER)
      INTEGER BLDAT,AANDK,ERSOR,STRFN,SLCRD,SURVL,AATEMP,SURF,SURFBV,
     1FIRST,UPPER,UPPRBV,S1,ST,SRW
      REAL K,KAK,LAMBDA,LMAX,MH,MLE,MR,MSL,MSP,MV,MVIM1
      IF (FIRST.GT.LAST) RETURN
      IF (FIRST.EQ.1.AND.INTVL.GT.0) WRITE(6,1000)
      IF (FIRST.EQ.1) RELER = .0
      DO 20 IM=FIRST,LAST
      IPU = IV(IM)
      IPL = IPU+NBBI-1
      TWLMR = 2.*OMEGA*LAMBDA-(OMEGA*RM(IM))**2
      LER(1)=4
      DO 10 IP=IPU,IPL
C     DENSITY CALL NO. 4
      CALL DENSTY(W(IP),RHO(IP),ANS,TWLMR,CPTIP,EXPON,RHOIP,GAM,AR,TIP)
   10 W(IP) = ANS
      IF (INTVL.LE.0) GO TO 20
      WRITE (6,1010) IM,(W(IP),BETA(IP),IP=IPU,IPL)
   20 CONTINUE
      RETURN
C
C  VELBB CALCULATES ALONG VERTICAL MESH LINES WHICH INTERSECT BLADES
C
      ENTRY VELBB(FIRST,LAST,UPPER,LOWER)
      IF (FIRST.GT.LAST) RETURN
      IF (FIRST.NE.MBI) GO TO 30
      SURFL(MBI,1) = 0.
      SURFL(MBI,2) = 0.
      SURFL(MBI2,3) = 0.
      SURFL(MBI2,4) = 0.
   30 DO 70 IM=FIRST,LAST
      ITVU = ITV(IM,UPPER)
      ITVL = ITV(IM,LOWER)
      IPUP1 = IPF(IM,ITVU)
      IPLM1 = IPF(IM,ITVL)
      TWLMR = 2.*OMEGA*LAMBDA-(OMEGA*RM(IM))**2
      WCR = SQRT(TGROG*TIP*(1.-TWLMR/CPTIP))
      IF (ITVL.LT.ITVU) GO TO 50
C  ALONG THE LINE BETWEEN BLADES
      LER(1)=5
      DO 40 IP=IPUP1,IPLM1
C     DENSTY CALL NO. 5
      CALL DENSTY(W(IP),RHO(IP),ANS,TWLMR,CPTIP,EXPON,RHOIP,GAM,AR,TIP)
   40 W(IP) = ANS
      IF (INTVL.LE.0) GO TO 50
      WRITE (6,1010) IM,(W(IP),BETA(IP),IP=IPUP1,IPLM1)
C  ON THE UPPER SURFACE
   50 RHOB = RHOVB(IM,UPPER)
      LER(1)=6
C     DENSTY CALL NO. 6
      CALL DENSTY(WMB(IM,UPPER),RHOVB(IM,UPPER),ANS,TWLMR,CPTIP,
     1EXPON,RHOIP,GAM,AR,TIP)
      WMB(IM,UPPER) = ANS
      WWCRM(IM,UPPER) = WMB(IM,UPPER)/WCR
      IF (IM.EQ.MBI.OR.(IM.EQ.MBI2.AND.UPPER.EQ.3)) GO TO 60
      DELTV = TV(IM-1,UPPER)-TV(IM,UPPER)
```

```
      IF (IM.EQ.MBOP1.AND.UPPER.EQ.3) DELTV = DELTV-PITCH
      SURFL( IM,UPPER) = SURFL(IM-1,UPPER) + SQRT((MV(IM)-MV(IM-1))**2 +
     1(DELTV*(RM(IM)+RM(IM-1))/2.)**2)
   60 RELER = AMAX1(RELER,ABS((RHOB-RHOVB(IM,UPPER))/RHOVB(IM,UPPER)))
C  ON THE LOWER SURFACE
      RHOB = RHOVB(IM,LOWER)
      LER(1)=7
C     DENSTY CALL NO. 7
      CALL DENSTY(WMB(IM,LOWER),RHOVB(IM,LOWER),ANS,TWLMR,CPTIP,
     1EXPON,RHOIP,GAM,AR,TIP)
      WMB(IM,LOWER) = ANS
      WWCRM(IM,LOWER) = WMB(IM,LOWER)/WCR
      IF (IM.EQ.MBI.OR.(IM.EQ.MBI2.AND.LOWER.EQ.4)) GO TO 70
      DELTV = TV(IM-1,LOWER)-TV(IM,LOWER)
      SURFL( IM,LOWER) = SURFL(IM-1,LOWER) + SQRT((MV(IM)-MV(IM-1))**2 +
     1(DELTV*(RM(IM)+RM(IM-1))/2.)**2)
   70 RELER = AMAX1(RELER,ABS((RHOB-RHOVB(IM,LOWER))/RHOVB(IM,LOWER)))
      RETURN
C
C  VELSUR CALCULATES ALONG A BLADE SURFACE
C
      ENTRY VELSUR
      DO 90 SURF=1,4
      IMSS = IMS(SURF)
      IF( IMSS.EQ.0) GO TO 90
      DO 80 IHS=1,IMSS
      TWLMR = 2.*OMEGA*LAMBDA-(OMEGA*RMH(IHS,SURF))**2
      WCR = SQRT(TGROG*TIP*(1.-TWLMR/CPTIP))
      RHOB = RHOHB(IHS,SURF)
      LER(1)=8
C     DENSTY CALL NO. 8
      CALL DENSTY(WTB(IHS,SURF),RHOHB(IHS,SURF),ANS,TWLMR,CPTIP,
     1EXPON,RHOIP,GAM,AR,TIP)
      WTB(IHS,SURF) = ANS
      WWCRT( IHS,SURF) = WTB(IHS,SURF)/WCR
   80 RELER = AMAX1(RELER,ABS((RHOB-RHOHB(IHS,SURF))/RHOHB(IHS,SURF)))
   90 CONTINUE
      IF (RELER.LT..001) IEND = IEND+1
      WRITE(6,1080) ITER,RELER
C
C   WRITE ALL BLADE SURFACE VELOCITIES
C
      IF (SURVL.LE.0) RETURN
      WRITE(6,1020)
      WRITE( 6,1040) (MV(IM),WMB(IM,1),BETAV(IM,1),SURFL(IM,1),
     1WWCRM( IM,1),WMB(IM,2),BETAV(IM,2),SURFL(IM,2),WWCRM(IM,2),
     2IM=MBI,MBO)
      WRITE( 6,1030)
      WRITE( 6,1040) (MV(IM),WMB(IM,3),BETAV(IM,3),SURFL( IM,3),
     1WWCRM( IM,3),WMB(IM,4),BETAV(IM,4),SURFL(IM,4),WWCRM(IM,4),
     2IM=MBI2,MBO2)
      WRITE( 6,1050)
      DO 100 SURF=1,4
      IMSS = IMS(SURF)
      IF( IMSS.LT.1) GO TO 100
      WRITE( 6,1060) SURF
      WRITE( 6,1070) (MH(IHS,SURF),WTB(IHS,SURF),BETAH(IHS,SURF),
     1WWCRT( IHS,SURF),IHS=1,IMSS)
  100 CONTINUE
      RETURN
 1000 FORMAT(1H1////40X,34HVELOCITIES AT INTERIOR MESH POINTS//)
 1010 FORMAT(1HL,3HIM=,I3,5(24H    VELOCITY    ANGLE(DEG))/
     1(5X,5(G15.4,F9.2)))
 1020 FORMAT(1H1////16X,1H*,18X,49HSURFACE VELOCITIES BASED ON MERIDIONA
     1L COMPONENTS,43X,1H*/16X,1H*,53X,1H*,56X,1H*/16X,1H*,19X,15HBLADE
     2SURFACE 1,19X,1H*,20X,15HBLADE SURFACE 2,21X,1H*/7X,1HM,8X,1H*,2(3
     3X,8HVELOCITY,3X,23HANGLE(DEG) SURF. LENGTH,5X,5HW/WCR,6X,1H*,3X))
 1030 FORMAT(///16X,1H*,19X,15HBLADE SURFACE 3,19X,1H*,20X,15HBLADE SURF
     1ACE 4,21X,1H*/7X,1HM,8X,1H*,2(3X,8HVELOCITY,3X,23HANGLE(DEG) SURF.
     2 LENGTH,5X,5HW/WCR,6X,1H*,3X))
 1040 FORMAT(1H ,G13.4,3H  *,G12.4,F9.2,2G15.4,6H  *    ,G12.4,F9.2,
     12G15.4,3H  *)
 1050 FORMAT(1H1////3X,49HSURFACE VELOCITIES BASED ON TANGENTIAL COMPONE
     1NTS)
 1060 FORMAT(//22X,15HBLADE SURFACE  ,I1/7X,1HM,10X,8HVELOCITY,3X,10HANG
     1LE(DEG),3X,5HW/WCR)
 1070 FORMAT(1H ,2G13.4,F9.2,G15.4)
 1080 FORMAT(14HLITERATION NO.,I3,3X,36HMAXIMUM RELATIVE CHANGE IN DENSI
     1TY =,G11.4)
      END
```

```
      SUBROUTINE BLCD
C
C  BLCD CALCULATES BLADE THETA COORDINATE AS A FUNCTION OF M
C
      COMMON SRW,ITER,IEND,LER(2),NER(2)
      COMMON /INP/ GAM,AR,TIP,RHOIP,WTFL,WTFLSP,OMEGA,ORF,BETAI,BETAO,
     1MBI,MBO,MBI2,MBO2,MM,NBBI,NBL,NRSP,MR(50),RMSP(50),BESP(50),
     2BLDAT,AANDK,ERSOR,STRFN,SLCRD,INTVL,SURVL
      COMMON /CALCON/ MBIM1,MBIP1,MBOM1,MBOP1,MBI2M1,MBI2P1,MBO2M1,
     1MBO2P1,MMM1,HM1,HM2,HM3,HT,DTLR,DMLR,PITCH,CP,EXPON,TWW,CPTIP,
     2TGROG,TBI,TBO,LAMBDA,TWL,ITMIN,ITMAX,NIP,IMS(4),BV(4),MV(100),
     3IV(101),ITV(100,4),TV(100,4),DTDMV(100,4),BETAV(100,4),
     4MH(100,4),DTDMH(100,4),BETAH(100,4),RMH(100,4),BEH(100,4),
     5RM(100),BE(100),DBDM(100),SAL(100),AAA(100)
      COMMON /GEOMIN/ CHORD(4),STGR(4),MLE(4),THLE(4),RMI(4),RMO(4),
     1RI(4),RO(4),BETI(4),BETO(4),NSPI(4),MSP(50,4),THSP(50,4)
      COMMON /BLCDCM/ EM(50,4),INIT(4)
      ENTRY  BL1(M,THETA,DTDM,INF)
      INTEGER BLDAT,AANDK,ERSOR,STRFN,SLCRD,SURVL,AATEMP,SURF,SURFBV,
     1FIRST,UPPER,UPPRBV,S1,ST,SRW
      REAL K,KAK,LAMBDA,LMAX,MH,MLE,MR,MSL,MSP,MV,MVIM1
      REAL M,MMLE,MSPMM,MMMSP
      SURF= 1
      SIGN= 1.
      GO TO 10
      ENTRY  BL2(M,THETA,DTDM,INF)
      SURF= 2
      SIGN= -1.
      GO TO 10
      ENTRY  BL3(M,THETA,DTDM,INF)
      SURF= 3
      SIGN= 1.
      GO TO 10
      ENTRY  BL4(M,THETA,DTDM,INF)
      SURF= 4
      SIGN= -1.
   10 INF= 0
      NSP= NSPI(SURF)
      IF (INIT(SURF).EQ.13) GO TO 30
      INIT(SURF)= 13
C
C  INITIAL CALCULATION OF FIRST AND LAST SPLINE POINTS ON BLADE
C
      AA = BETI(SURF)/57.295779
      AA = SIN(AA)
      MSP(1,SURF) =  RI(SURF)*(1.-SIGN*AA)
      BB = SQRT(1.-AA**2)
      THSP(1,SURF) = SIGN*BB*RI(SURF)/RMI(SURF)
      BETI(SURF) = AA/BB/RMI(SURF)
      AA = BETO(SURF)/57.295779
      AA = SIN(AA)
      MSP(NSP,SURF) = CHORD(SURF)-RO(SURF)*(1.+SIGN*AA)
      BB = SQRT(1.-AA**2)
      THSP(NSP,SURF) = STGR(SURF)+SIGN*BB*RO(SURF)/RMO(SURF)
      BETO(SURF) = AA/BB/RMO(SURF)
      DO 20 IA=1,NSP
      MSP(IA,SURF)= MSP(IA,SURF)+MLE(SURF)
   20 THSP(IA,SURF)= THSP(IA,SURF)+THLE(SURF)
      CALL SPLN22(MSP(1,SURF),THSP(1,SURF),BETI(SURF),BETO(SURF),NSP,
     1 AAA,EM(1,SURF))
      IF(BLDAT.LE.0) GO TO 30
      IF (SURF.EQ.1) WRITE(6,1000)
      WRITE(6,1010) SURF
      WRITE (6,1020) (MSP(IA,SURF),THSP(IA,SURF),AAA(IA),EM(IA,SURF),
     1IA=1,NSP)
C
C  BLADE COORDINATE CALCULATION
C
   30 KK = 2
      IF (M.GT.MSP(1,SURF)) GO TO 50
C
C  AT LEADING EDGE RADIUS
C
      MMLE= M-MLE(SURF)
```

```
      IF (MMLE.LT.-DMLR) GO TO 90
      MMLE= AMAX1(0.,MMLE)
      THETA= SQRT(MMLE*(2.*RI(SURF)-MMLE))*SIGN
      IF (THETA.EQ.0.) GO TO 40
      RMM= RI(SURF)-MMLE
      DTDM= RMM/THETA/RMI(SURF)
      THETA= THETA/RMI(SURF)+THLE(SURF)
      RETURN
   40 INF= 1
      DTDM = 1.E10*SIGN
      THETA= THLE(SURF)
      RETURN
C
C  ALONG SPLINE CURVE
C
   50 IF (M.LE.MSP(KK,SURF)) GO TO 60
      IF (KK.GE.NSP) GO TO 70
      KK = KK+1
      GO TO 50
   60 S= MSP(KK,SURF)-MSP(KK-1,SURF)
      EMKM1= EM(KK-1,SURF)
      EMK= EM(KK,SURF)
      MSPMM= MSP(KK,SURF)-M
      MMMSP= M-MSP(KK-1,SURF)
      THK= THSP(KK,SURF)/S
      THKM1= THSP(KK-1,SURF)/S
      THETA= EMKM1*MSPMM**3/6./S + EMK*MMMSP**3/6./S + (THK-EMK*S/6.)*
     1MMMSP+(THKM1-EMKM1*S/6.)*MSPMM
      DTDM= -EMKM1*MSPMM**2/2./S + EMK*MMMSP**2/2./S + THK-THKM1-(EMK-
     1EMKM1)*S/6.
      RETURN
C
C  AT TRAILING EDGE RADIUS
C
   70 CMM= CHORD(SURF)+MLE(SURF)-M
      IF (CMM.LT.-DMLR) GO TO 90
      CMM= AMAX1(0.,CMM)
      THETA= SQRT(CMM*(2.*RO(SURF)-CMM))*SIGN
      IF (THETA.EQ.0.) GO TO 80
      RMM= RO(SURF)-CMM
      DTDM = -RMM/THETA/RMO(SURF)
      THETA = STGR(SURF)+THETA/RMO(SURF)+THLE(SURF)
      RETURN
   80 INF= 1
      DTDM = -1.E10*SIGN
      THETA= THLE(SURF)+STGR(SURF)
      RETURN
C
C  ERROR RETURN
C
   90 WRITE(6,1030) LER(2),M,SURF
      STOP
 1000 FORMAT (1H1,13X,33HBLADE DATA AT INPUT SPLINE POINTS)
 1010 FORMAT(1HL,17X,16HBLADE     SURFACE,I4)
 1020 FORMAT (7X ,1HM,10X,5HTHETA,10X,10HDERIVATIVE,5X,10H2ND DERIV. /
     1(4G15.5))
 1030 FORMAT (14HLBLCD CALL NO.,I3/33H M COORDINATE IS NOT WITHIN BLADE/
     14H M =,G14.6,10X,6HSURF =,G14.6)
      END
```

```
      FUNCTION IPF(IM,IT)
      COMMON /INP/ GAM,AR,TIP,RHOIP,WTFL,WTFLSP,OMEGA,ORF,BETAI,BETAO,
     1MBI,MBO,MBI2,MBO2,MM,NBBI,NBL,NRSP,MR(50),RMSP(50),BESP(50),
     2BLDAT,AANDK,ERSOR,STRFN,SLCRD,INTVL,SURVL
      COMMON /CALCON/ MBIM1,MBIP1,MBOM1,MBOP1,MBI2M1,MBI2P1,MBO2M1,
     1MBO2P1,MMM1,HM1,HM2,HM3,HT,DTLR,DMLR,PITCH,CP,EXPON,TWW,CPTIP,
     2TGROG,TBI,TBO,LAMBDA,TWL,ITMIN,ITMAX,NIP,IMS(4),BV(4),MV(100),
     3IV(101),ITV(100,4),TV(100,4),DTDMV(100,4),BETAV(100,4),
     4MH(100,4),DTDMH(100,4),BETAH(100,4),RMH(100,4),BEH(100,4),
     5RM(100),BE(100),DBDM(100),SAL(100),AAA(100)
      IPF = IV(IM)+IT-ITV(IM,1)-ITV(IM,3)-10000
      IF(IM.LT.MBI2.OR.IM.GT.MBO) RETURN
      IPF = IV(IM)+IT-ITV(IM,1)
      IF(IT.GE.ITV(IM,3)) IPF = IPF-ITV(IM,3)+ITV(IM,4)+1
      RETURN
      END
```

Subroutine MHORIZ

Subroutine MHORIZ calculates the m-coordinates of intersections of all horizontal mesh lines with a blade surface. It locates points of intersection, by checking the ITV array (see main dictionary). If ITV changes between adjacent vertical lines, there must be a horizontal mesh line intersection between those vertical lines. ROOT is called to calculate the m-coordinate of the intersection. The input arguments for MHORIZ are as follows:

MV — array of m-coordinates of vertical mesh lines

ITV — same as ITV of main dictionary, but for a particular surface

BL — subroutine giving blade θ-coordinate as function of m (BL may be BL1, BL2, BL3, or BL4 in the calling statement. These are the entry points of BLCD.)

MBI — value of IM at first vertical mesh line to be checked

MBO — value of IM at last vertical mesh line to be checked

ITO — value of IT at the origin of coordinates at leading edge of front blade

HT — mesh spacing in θ-direction

DTLR — tolerance in θ-direction

KODE — code variable to indicate whether blade surface is upper or lower; KODE = 0 for upper blade surface, KODE = 1 for lower blade surface

MRTS — integer switch indicating infinite slopes at leading or trailing edge of a blade surface

The output arguments for MHORIZ are as follows:

J — counter indicating current number of intersections of horizontal mesh lines with a given blade surface

MH — m-coordinate of intersection of horizontal mesh line with blade

DTDMH — slope $d\theta/dm$ where horizontal mesh line meets blade

The internal variables for MHORIZ are as follows:

IM — vertical mesh line number

ITIND — counter of horizontal mesh lines which intersect blades between two consecutive vertical mesh lines

MVIM — MV at left end of horizontal interval

TI — θ-coordinate of horizontal mesh line which intersects blade

```
      SUBROUTINE MHORIZ(MV,ITV,BL,MBI,MBO,ITO,HT,DTLR,KODE,J,MH,DTDMH,
     1MRTS)
C
C  MHORIZ CALCULATES M COORDINATES OF INTERSECTIONS OF ALL HORIZONTAL
C  MESH LINES WITH A BLADE SURFACE
C  KODE = 0 FOR UPPER BLADE SURFACE
C  KODE = 1 FOR LOWER BLADE SURFACE
C
      COMMON SRW,ITER,IEND,LER(2),NER(2)
      DIMENSION MV(100),ITV(100),MH(100),DTDMH(100)
      INTEGER BLDAT,AANDK,ERSOR,STRFN,SLCRD,SURVL,AATEMP,SURF,SURFBV,
     1FIRST,UPPER,UPPRBV,SI,ST,SRW
      REAL K,KAK,LAMBDA,LMAX,MH,MLE,MR,MSL,MSP,MV,MVIM1
      REAL MVIM
      EXTERNAL BL
      IF (MBI.GE.MBO) RETURN
      IM= MBI
   10 ITIND= 0
   20 IF (ITV(IM+1)-ITV(IM)-ITIND) 30,40,50
   30 J= J+1
      TI= FLOAT(ITV(IM+1)-ITO-ITIND+KODE)*HT
      ITIND= ITIND-1
      MVIM = MV(IM)
      IF (MRTS.EQ.1) MVIM = MVIM+(MV(IM+1)-MVIM)/1000.
      CALL ROOT (MVIM,MV(IM+1),TI,BL,DTLR,MH(J),DTDMH(J))
      GO TO 20
   40 IM= IM+1
      MRTS = 0
      IF (IM.EQ.MBO) RETURN
      GO TO 10
   50 J= J+1
      TI= FLOAT(ITV(IM)-ITO+ITIND+KODE)*HT
      ITIND= ITIND+1
      MVIM = MV(IM)
      IF (MRTS.EQ.1) MVIM = MVIM+(MV(IM+1)-MVIM)/1000.
      CALL ROOT (MVIM,MV(IM+1),TI,BL,DTLR,MH(J),DTDMH(J))
      GO TO 20
      END
```

Subroutine DENSTY

Subroutine DENSTY calculates the subsonic relative velocity W and corresponding density ρ that result in a given value of the mass flow parameter ρW. This is done by using equations (B5) and (B6), which are an algorithm based on Newton's method.

If the value of ρW is too large, there is no solution. In this case an error message is printed, W_{cr} and the corresponding density are calculated as output, and the program continues. Thus, it is possible to get an approximate solution even though there may be one or two points with too large a value for ρW. The input arguments for DENSTY are as follows:

RHOW	ρW
RHO	initial estimate for ρ (ρ'_{in} may be used)
TWLMR	$2\omega\lambda - (\omega r)^2$
CPTIP	$2c_p T'_{in}$
EXPON	$1/(\gamma - 1)$

RHOIP	ρ'_{in}
GAM	γ
AR	R
TIP	T'_{in}
VTOL	convergence tolerance on relative change in W

The output arguments for DENSTY are as follows:

RHO	ρ
VEL	W

The internal variables for DENSTY are as follows:

RHOT	newly calculated estimate for ρ
RHOWP	$d(\rho W)/d\rho$ (eq. (B4))
TEMP	$(T/T'_{in})^{(2-\gamma)/(\gamma-1)}$
TGROG	$2\gamma R/(\gamma + 1)$
TTIP	T/T'_{in}
VELNEW	newly calculated estimate for W

```
      SUBROUTINE DENSTY(RHOW,RHO,VEL,TWLMR,CPTIP,EXPON,RHOIP,GAM,AR,TIP)
C
C  DENSTY CALCULATES DENSITY AND VELOCITY FROM THE WEIGHT FLOW PARAMETER
C  DENSITY TIMES VELOCITY
C
      COMMON SRW,ITER,IEND,LER(2),NER(2)
      VEL = RHOW/RHO
      IF (VEL.NE.0.) GO TO 10
      RHO = RHOIP
      RETURN
   10 TTIP = 1.-(VEL**2+TWLMR)/CPTIP
      IF(TTIP.LT.0.) GO TO 30
      TEMP = TTIP**(EXPON-1.)
      RHOT = RHOIP*TEMP*TTIP
      RHOWP = -VEL**2/GAM*RHOIP/AR*TEMP/TIP+RHOT
      IF(RHOWP.LE.0.) GO TO 30
      VELNEW = VEL+(RHOW-RHOT*VEL)/RHOWP
      IF(ABS(VELNEW-VEL)/VELNEW.LT..0001) GO TO 20
      VEL = VELNEW
      GO TO 10
   20 VEL = VELNEW
      RHO = RHOW/VEL
      RETURN
   30 TGROG = 2.*GAM*AR/(GAM+1.)
      VEL = SQRT(TGROG*TIP*(1.-TWLMR/CPTIP))
      RHO = RHOIP*(1.-(VEL**2+TWLMR)/CPTIP)**EXPON
      RWMORW = RHOW/RHO/VEL
      NER(1) = NER(1)+1
      WRITE(6,1000) LER(1),NER(1),RWMORW
      IF(NER(1).EQ.50) STOP
      RETURN
 1000 FORMAT(16HLDENSTY CALL NO.,I3/9H NER(1) =,I3/10H RHO*W IS ,F7.4,
     134H TIMES THE MAXIMUM VALUE FOR RHO*W)
      END
```

Subroutine ROOT

Subroutine ROOT finds a root for $f(x) = y$ by Newton's method. The function $f(x)$ must be defined on a specific interval $[a, b]$. The values of $f(x)$ are calculated by another subroutine (FUNCT).

The value x^{k+1} is determined from x^k by

$$x^{k+1} = \frac{y - f(x^k)}{f'(x^k)} + x^k$$

The first value of x is $x^0 =$ If x^{k+1} is not in the interval, if $f'(x^k) = 0$, or if $f'(x^k) = \infty$, the interval $[a, b]$ is scanned to see if a suitable starting value of x for Newton's method can be found. If a root cannot be found in 1000 iterations, a message is printed, and the calculations are stopped.

Subroutine ROOT requires that $f(x)$ be calculated by a FORTRAN subroutine subprogram (FUNCT). Any name may be chosen for this subroutine. In TANDEM, FUNCT is either BL1, BL2, BL3, or BL4. The calling sequence is

FUNCT(X, FX, DFX, INF)

These arguments are defined as follows:

FX	$f(x)$
DFX	$f'(x)$
INF	used to indicate an infinite derivative: 0 if $f'(x)$ is finite 1 if $f'(x)$ is infinite

The input arguments for ROOT are as follows:

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

The output arguments for ROOT are as follows:

X value of x such that $f(x) = y$

DFX $f'(x)$

```
      SUBROUTINE ROOT(A,B,Y,FUNCT,TOLERY,X,DFX)
C
C  RCOT FINDS A RCOT FCR (FUNCT MINUS Y) IN THE INTERVAL (A,B)
C
      COMMON SRW,ITER,IEND,LER(2),NER(2)
      INTEGER SRW
      IF (SRW.EQ.21) WRITE(6,1000) A,B,Y,TOLERY
      TOLERX= (B-A)/1000.
      AB2= (A+B)/2.
      I = 0
      X = A
   10 CALL FUNCT(X,FX,DFX,INF)
      IF (SRW.EQ.21) WRITE(6,1010) I,X,FX,DFX,INF
      IF (ABS(Y-FX).LT.TOLERY) RETURN
      IF (I.GE.1000) GC TO 30
      I= I+1
      IF (INF.NE.0 .OR. DFX.EQ.0.) GO TO 20
      X= (Y-FX)/DFX+X
      IF (X.GE.A .AND. X.LE.B) GO TO 10
      X = A+TOLERX*FLOAT(I)
      IF(I.EQ.1) X = B
      GO TC 10
   20 IF (X.LT.AB2) X=X+TOLERX
      IF (X.GE.AB2) X=X-TOLERX
      GO TO 10
   30 WRITE(6,1020) LER(2),A,B,Y
      STOP
 1000 FORMAT (32H1INPUT ARGUMENTS FOR ROOT -- A =G13.5,3X,3HB =,G13.5,
     13X,3HY =,G13.5,3X,8HTOLERY =,G13.5/17H  ITER. NO.     X,17X,
     22HFX,15X,3HCFX,10X,3HINF)
 1010 FORMAT (5X,I3,G16.5,2G18.5,I6)
 1020 FORMAT (14HLROOT CALL NO.,I3/47H ROOT HAS FAILED TC CONVERGE IN 10
     100 ITERATIONS/4H A =,G14.6,10X,3HB =,G14.6,10X,3HY =,G14.6)
      END
```

Subroutine SPLINE

This subroutine is based on the cubic spline curve. SPLINE solves a tridiagonal matrix equation given in reference 9 to obtain the coefficients for the piecewise cubic polynomial function giving the spline fit curve. SPLINE is based on the end condition that the second derivative at either end point is one-half that at the next spline point. The input variables for SPLINE are as follows:

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 as follows:

SLOPE array of first derivatives

EM array of second derivatives

If Q = 13 in COMMON, input and output data for SPLINE are printed. This is useful in debugging.

```
      SUBROUTINE SPLINE (X,Y,N,SLOPE,EM)
C
C   SPLINE CALCULATES FIRST AND SECOND DERIVATIVES AT SPLINE POINTS
C   END CONDITION - SECOND DERIVATIVES ARE THE SAME AT END POINT AND
C   ADJACENT POINT
C
      COMMON Q/BOX/S(100),A(100),B(100),C(100),F(100),W(100),SB(100),
     1G(200)
      DIMENSION X(N),Y(N),EM(N),SLOPE(N)
      INTEGER Q
      DO 10 I=2,N
   10 S(I)=X(I)-X(I-1)
      NO=N-1
      IF(NO.LT.2) GO TO 30
      DO 20 I=2,NO
      A(I)=S(I)/6.
      B(I)=(S(I)+S(I+1))/3.
      C(I)=S(I+1)/6.
   20 F(I)=(Y(I+1)-Y(I))/S(I+1)-(Y(I)-Y(I-1))/S(I)
   30 A(N) = -.5
      B(1)=1.
      B(N)=1.
      C(1) = -.5
      F(1)=0.
      F(N)=0.
      W(1)=B(1)
      SB(1)=C(1)/W(1)
      G(1)=0.
      DO 40 I=2,N
      W(I)=B(I)-A(I)*SB(I-1)
      SB(I)=C(I)/W(I)
   40 G(I)=(F(I)-A(I)*G(I-1))/W(I)
      EM(N)=G(N)
      DO 50 I=2,N
      K=N+1-I
   50 EM(K)=G(K)-SB(K)*EM(K+1)
      SLOPE(1)=-S(2)/6.*(2.*EM(1)+EM(2))+(Y(2)-Y(1))/S(2)
      DO 60 I=2,N
   60 SLOPE(I)=S(I)/6.*(2.*EM(I)+EM(I-1))+(Y(I)-Y(I-1))/S(I)
      IF (Q.EQ.13) WRITE(6,1000) N,(X(I),Y(I),SLOPE(I),EM(I),I=1,N)
      RETURN
 1000 FORMAT (2X,15HNO. OF POINTS =,I3/10X,1HX,19X,1HY,19X,5HSLOPE,15X,
     12HEM/(4F20.8))
      END
```

Subroutine SPLN22

This subroutine is the same as SPLINE, except that, for the end conditions, the slopes are specified. The input variables for SPLN22 are as follows:

X	array of ordinates
Y	array of function values
Y1P	slope at first point
YNP	slope at last point
N	number of X and Y values given

The output variables for SPLN22 are as follows:

SLOPE	array of first derivatives
EM	array of second derivatives

If Q = 18 in COMMON, input and output data for SPLN22 are printed. This is useful in debugging.

```
      SUBROUTINE SPLN22 (X,Y,Y1P,YNP,N,SLOPE,EM)
C
C   SPLN22 CALCULATES FIRST AND SECOND DERIVATIVES AT SPLINE POINTS
C   END CONDITION - DERIVATIVES SPECIFIED AT END POINTS
C
      COMMON Q/BOX/S(100),A(100),B(100),C(100),F(100),W(100),SB(100),
     1G(200)
      DIMENSION X(N),Y(N),EM(N),SLOPE(N)
      INTEGER Q
      DO 10 I=2,N
   10 S(I)=X(I)-X(I-1)
      NO=N-1
      IF(NO.LT.2) GO TO 30
      DO 20 I=2,NO
      A(I)=S(I)/6.
      B(I)=(S(I)+S(I+1))/3.
      C(I)=S(I+1)/6.
   20 F(I)=(Y(I+1)-Y(I))/S(I+1)-(Y(I)-Y(I-1))/S(I)
   30 A(N) = S(N)/6.
      B(1)=S(2)/3.
      B(N) = S(N)/3.
      C(1)=S(2)/6.
      F(1)=(Y(2)-Y(1))/S(2)-Y1P
      F(N) = YNP-(Y(N)-Y(N-1))/S(N)
      W(1)=B(1)
      SB(1)=C(1)/W(1)
      G(1)=F(1)/W(1)
      DO 40 I=2,N
      W(I)=B(I)-A(I)*SB(I-1)
      SB(I)=C(I)/W(I)
```

```
   40 G(I)=(F(I)-A(I)*G(I-1))/W(I)
      EM(N)=G(N)
      DO 50 I=2,N
      K=N+1-I
   50 EM(K)=G(K)-SB(K)*EM(K+1)
      SLOPE(1)=-S(2)/6.*(2.*EM(1)+EM(2))+(Y(2)-Y(1))/S(2)
      DO 60 I=2,N
   60 SLOPE(I)=S(I)/6.*(2.*EM(I)+EM(I-1))+(Y(I)-Y(I-1))/S(I)
      IF (Q.EQ.18) WRITE(6,1000) N,(X(I),Y(I),SLOPE(I),EM(I),I=1,N)
      RETURN
 1000 FORMAT (2X,15HNO. OF POINTS =,I3/10X,1HX,19X,1HY,19X,5HSLOPE,15X,
     12HEM/(4F20.8))
      END
```

Subroutine SPLINT

This subroutine is based on the cubic spline curve, with the same end conditions as SPLINE. The cubic spline curve is then used for interpolation. The input variables for SPLINT are as follows:

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 function values are desired

MAX   number of Z values given

The output variable for SPLINT is as follows:

YINT   array of interpolated function values

If Q = 16 in COMMON, or if some element of the z array falls outside of the interval for the x array, input and output data for SPLINT are printed. This is useful in debugging.

```
      SUBROUTINE SPLINT (X,Y,N,Z,MAX,YINT,DYDX)
C
C  SPLINT CALCULATES INTERPOLATED POINTS AND DERIVATIVES
C  FOR A SPLINE CURVE
C  END CONDITION - SECOND DERIVATIVES ARE THE SAME AT END POINT AND
C  ADJACENT POINT
C
      COMMON Q/BOX/S(100),A(100),B(100),C(100),F(100),W(100),SB(100),
     1G(100),EM(100)
      DIMENSION   X(N),Y(N),Z(MAX),YINT(MAX),DYDX(MAX)
      INTEGER Q
      IF(MAX.LE.0) RETURN
      III = Q
      DO 10 I=2,N
   10 S(I)=X(I)-X(I-1)
      NO=N-1
      IF(NO.LT.2) GO TO 30
      DO 20 I=2,NO
      A(I)=S(I)/6.0
      B(I)=(S(I)+S(I+1))/3.0
      C(I)=S(I+1)/6.0
   20 F(I)=(Y(I+1)-Y(I))/S(I+1)-(Y(I)-Y(I-1))/S(I)
   30 A(N) = -.5
      B(1)=1.0
      B(N)=1.0
      C(1) = -.5
      F(1)=0.0
      F(N)=0.0
      W(1)=B(1)
      SB(1)=C(1)/W(1)
      G(1)=0.0
      DO 40 I=2,N
      W(I)=B(I)-A(I)*SB(I-1)
      SB(I)=C(I)/W(I)
   40 G(I)=(F(I)-A(I)*G(I-1))/W(I)
      EM(N)=G(N)
      DO 50 I=2,N
      K=N+1-I
   50 EM(K)=G(K)-SB(K)*EM(K+1)
      DO 140 I=1,MAX
      K=2
      IF(Z(I)-X(1)) 70,60,90
   60 YINT(I)=Y(1)
      GO TO 130
   70 IF(Z(I).GE.(1.1*X(1)-.1*X(2))) GO TO 120
      WRITE (6,1000) Z(I)
      Q = 16
      GO TO 120
   80 K=N
      IF(Z(I).LE.(1.1*X(N)-.1*X(N-1))) GO TO 120
      WRITE (6,1000) Z(I)
      Q = 16
      GO TO 120
   90 IF(Z(I)-X(K)) 120,100,110
  100 YINT(I)=Y(K)
      GO TO 130
  110 K=K+1
      IF(K-N) 90,90,80
  120 YINT(I) = EM(K-1)*(X(K)-Z(I))**3/6./S(K)+EM(K)*(Z(I)-X(K-1))**3/6.
     1/S(K)+(Y(K)/S(K)-EM(K)*S(K)/6.)*(Z(I)-X(K-1))+(Y(K-1)/S(K)-EM(K-1)
     2*S(K)/6.)*(X(K)-Z(I))
  130 DYDX(I)=-EM(K-1)*(X(K)-Z(I))**2/2.0/S(K)+EM(K)*(X(K-1)-Z(I))**2/2.
     10/S(K)+(Y(K)-Y(K-1))/S(K)-(EM(K)-EM(K-1))*S(K)/6.0
  140 CONTINUE
      MXA = MAX0(N,MAX)
      IF(Q.EQ.16) WRITE(6,1010) N,MAX,(X(I),Y(I),Z(I),YINT(I),DYDX(I),
     1I=1,MXA)
      Q = III
      RETURN
 1000 FORMAT (54H SPLINT USED FOR EXTRAPOLATION.  EXTRAPOLATED VALUE = ,
     1G14.6)
 1010 FORMAT (2X,21HNO. OF POINTS GIVEN =,I3,30H, NO. OF INTERPOLATED PO
     1INTS =,I3/10X,1HX,19X,1HY,16X,11HX-INTERPOL.,9X,11HY-INTERPOL.,
     28X,14HDYDX-INTERPOL./(5E20.8))
      END
```

Lewis Library Subroutine TIME1

This subroutine is part of the Lewis Systems Library. TIME1 gives the time in clock pulses of 1/60th of a second. To get elapsed time in minutes, the clock must be read twice and the difference divided by 3600. TIME1 may be replaced by a user's clock reading subroutine, or it may be removed from the program.

CONCLUDING REMARKS

It is not always possible to obtain sufficient detail on some critical parts of the blade surfaces by using the TANDEM program. Due to storage limitations on the computer, grid spacing may be too large to give the desired detail around small leading- or trailing-edge radii or within slot regions. For this reason, a computer program called MAGNFY has been written to obtain a solution on a finer mesh in a small part of the blade-to-blade region. MAGNFY is described in reference 13.

After TANDEM was written, it was realized that the TANDEM program was significantly improved over the 2DCP program (ref. 3) for a single unslotted blade. Hence, TANDEM was modified to solve the same problem as 2DCP. This modified program, called TURBLE, is described in reference 14. The coding in TURBLE is simpler and more foolproof than that of 2DCP. Also, TURBLE allows more interior mesh points in the solution region, and has its own error package independent of the Lewis computer system. It is intended that TURBLE should supersede both the 2DCP and the 2DINCP (ref. 11) programs.

Lewis Research Center,
National Aeronautics and Space Administration,
Cleveland, Ohio, November 4, 1968,
126-15-02-31-22.

APPENDIX A

FINITE-DIFFERENCE APPROXIMATION

An approximate numerical solution for the stream function u can be obtained by finite-difference methods. These methods involve first establishing a rectangular grid of mesh points in the region, as shown in figure 14. Then at each point where the value of the stream function is unknown, a finite-difference approximation to equation (1) 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. 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.

First, the inlet absolute total density is used for determining the coefficients of the finite-difference approximation to equation (1). This results in n linear equations. These linear equations may be solved iteratively by successive overrelaxation, as described in references 10 and 11. This solution is an approximate solution of equation (1) for the stream function. This approximate solution may be differentiated numerically to obtain approximate velocities from equations (2) and (3). The approximate velocities are then used to obtain a better approximation to the density at each point, and the coefficients of equation (1) are recalculated by using new densities. Thus, the solution to the nonlinear equation (1) is approached by a sequence of solutions to linear equations.

A typical mesh point with the numbering used to indicate neighboring mesh points is shown in figure 17. The value of the stream function or the other variables at 0 is denoted

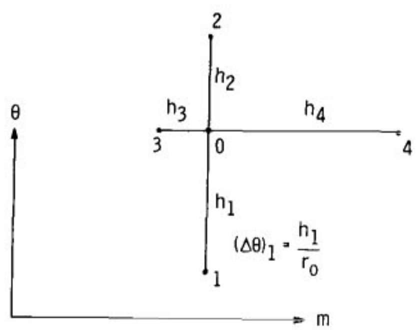


Figure 17. - Notation for adjacent mesh points and mesh spaces.

by using the subscript 0, and similarly for the neighboring points. It can be shown (ref. 10) that equation (1) can be approximated by

$$\left[\frac{2u_1}{h_1(h_1+h_2)}+\frac{2u_2}{h_2(h_1+h_2)}-\frac{2u_0}{h_1h_2}\right]+\left[\frac{2u_3}{h_3(h_3+h_4)}+\frac{2u_4}{h_4(h_3+h_4)}-\frac{2u_0}{h_3h_4}\right]$$
$$-\frac{1}{\rho_0}\left(\frac{\rho_2-\rho_1}{h_1+h_2}\right)\left(\frac{u_2-u_1}{h_1+h_2}\right)+\left[\frac{\sin\alpha_0}{r_0}-\frac{b_4\rho_4-b_3\rho_3}{b_0\rho_0(h_3+h_4)}\right]\left(\frac{u_4-u_3}{h_3+h_4}\right)=\frac{2\omega}{w}b_0\rho_0\sin\alpha_0 \tag{A1}$$

where $h_1 = r_0(\Delta\theta)_1$ and $h_2 = r_0(\Delta\theta)_2$ (since $r_0 = r_1 = r_2$). In setting up equations for solution, the coefficients of the u_i in equation (A1) must be calculated. This was done by expressing equation (A1) as

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

where

$$a_{12}=\frac{2}{h_1h_2}$$

$$a_{34}=\frac{2}{h_3h_4}$$

$$a_0=a_{12}+a_{34}$$

$$b_{12}=\frac{\rho_2-\rho_1}{\rho_0(h_1+h_2)}$$

$$b_{34}=\frac{b_4\rho_4-b_3\rho_3}{b_0\rho_0(h_3+h_4)}-\frac{\sin\alpha_0}{r_0}$$

$$a_1=\frac{1}{a_0(h_1+h_2)}\left(\frac{2}{h_1}+b_{12}\right)$$

$$a_2=\frac{a_{12}}{a_0}-a_1$$

$$a_3=\frac{1}{a_0(h_3+h_4)}\left(\frac{2}{h_3}+b_{34}\right)$$

$$a_4=\frac{a_{34}}{a_0}-a_3$$

$$k_0=-\frac{2\omega}{w}\frac{b_0\rho_0}{a_0}\sin\alpha_0 \tag{A2}$$

This equation can be used at all interior mesh points, and for mesh points adjacent to the blade surfaces BC, ML, and so forth.

Along the boundary where the value of u is unknown, the equation will vary. For example, along the upstream boundary, $\partial u/\partial \eta$ is known, and a finite-difference approximation to $(\partial u/\partial \eta)_{in}$ in equation (4) gives

$$u_0 = u_4 + h_4\left(\frac{\partial u}{\partial \eta}\right)_{in} = u_4 + h_4\left(\frac{\tan \beta_{in}}{sr_{in}}\right) \tag{A3}$$

Similarly, along the downstream boundary, equation (5) gives

$$u_0 = u_3 + h_3\left(\frac{\partial u}{\partial \eta}\right)_{out} = u_3 - h_3\left(\frac{\tan \beta_{out}}{sr_{out}}\right) \tag{A4}$$

For the points along AB, equations can be derived by using the periodic boundary condition. If the point 0 (fig. 18) is on the boundary between A and B, the point 1 is outside the boundary. However, it is known that $u_1 = u_{1,s} - 1$ where the point 1, s is a distance s above point 1 in the θ-direction, as shown in figure 18. Substituting this condition in equation (A2) gives

$$u_0 = a_1 u_{1,s} + \sum_{i=2}^{4} a_i u_i - a_1 + k_0 \tag{A5}$$

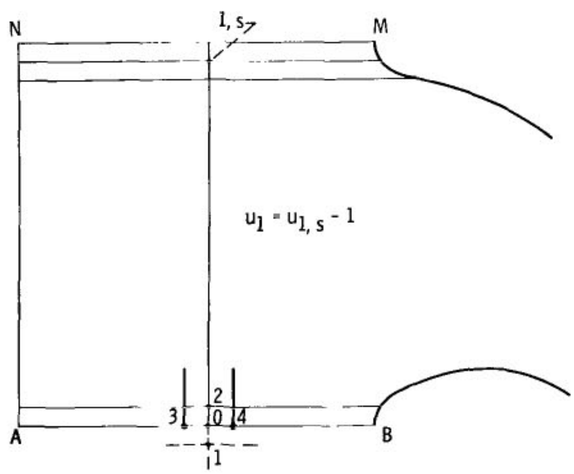


Figure 18. - Mesh point on line AB.

where a_i is the same as defined in equation (A2).

The points along MN are not part of the solution regions, since the value of the stream function at each of them is just 1 greater than the corresponding point along AB. The equation for the first mesh line below NM must be modified, however. In this case $u_2 = u_{2,-s} + 1$, where the point 2, -s is a distance s below point 2 in the negative θ-direction, as indicated in figure 19. Substituting this condition into equation (A2) gives

$$u_0 = a_1u_1 + a_2u_{2,-s} + a_3u_3 + a_4u_4 + a_2 + k_0 \tag{A6}$$

In a similar manner, equations can be derived along the other boundaries (FG, HI, CD, and KL; and DE and JK for the nonoverlapping case, fig. 4) where a periodic condition exists. Rather than give the equation for every possible case, it is easier to state the rule for modifying equation (A2). If an adjacent mesh point i (fig. 17) is outside the mesh region (along a periodic boundary), two changes must be made:

(1) Change the subscript of u from i to i, s if the periodic boundary is along the bottom of the mesh region. Change the i to i, -s along the top of the region.

(2) Subtract a_i from k_0 if the periodic boundary is along the bottom of the mesh region. Add a_i to k_0 along the top of the region.

One of equations (A2) to (A6) can be applied to each mesh point for which the stream function is unknown in the region of interest, giving the same number of equations as there are unknowns. These points where the stream function is unknown are referred to simply as unknown mesh points.

(A7)

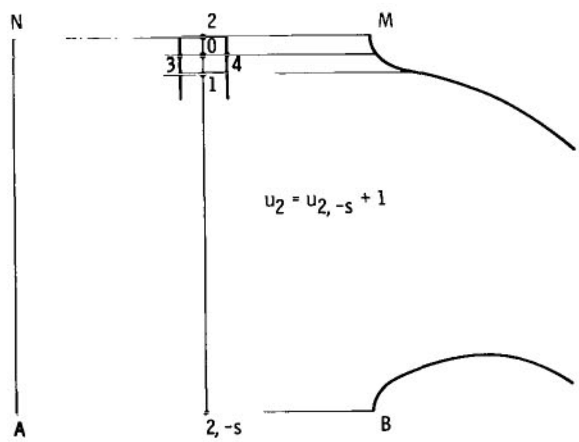


Figure 19. - Mesh point on first line below MN.

This system of n equations is represented in matrix form as

$$\underset{\sim}{A}\underset{\sim}{u} = \underset{\sim}{k} \tag{A7}$$

where $\underline{u} = (u_1, \ldots, u_n)^T$ is a vector whose components are the unknown values of the stream function, A is the coefficient matrix of equations (A2) to (A6), and $\underline{k} = (k_1, \ldots, k_n)^T$ is the vector whose components are the known constants of equations (A2) to (A6). If the mesh size is sufficiently small, the coefficients a_1 to a_4 in equation (A2) will all be positive (for any given continuous functions b and ρ). In this case, the coefficient matrix A is irreducibly diagonally dominant, and there is a unique solution to equation (A2) (ref. 10).

The solution to equation (A2) is obtained by using two levels of iteration. The inner iteration consists of solving equation (A2) by using fixed values of ρ based on the previous inner iteration. The inner iteration is successive overrelaxation using an optimum overrelaxation factor Ω, as described in reference 11 (p. 77). The iterative procedure is given by

$$u_i^{m+1} = u_i^m + \Omega\left(-\sum_{j=1}^{i-1} a_{ij}u_j^{m+1} - \sum_{j=i+1}^{n} a_{ij}u_j^m + k_i - u_i^m\right)$$

$$\text{for } i = 1, 2, \ldots, n \tag{A8}$$

where Ω is the overrelaxation factor. The a_{ij} are the elements of the matrix A, and the k_i are the components of the vector $\underline{k}$ of equation (A7). The u_i^0 are the initial estimates of the u_i and are obtained from the previous inner iteration.

The outer iteration consists of making corrections to the coefficients so as to finally obtain a solution to the nonlinear equation (1). The optimum value of Ω can be determined as described in reference 11 (appendix B). The optimum value of Ω will vary slightly each time the coefficients are corrected; however, the change is usually small, and it has been adequate to use the same overrelaxation factor for the entire calculation.

APPENDIX B

NUMERICAL TECHNIQUES USED IN PROGRAM

Calculation of Velocity and Density

When the stream function u has been calculated, it is then possible to calculate the derivatives $\partial u/\partial m$ and $\partial u/\partial \theta$ by numerical techniques. Then, with equations (2) and (3), and since $W^2 = W_m^2 + W_\theta^2$, values for ρW can be calculated. It is assumed that the values of ω, λ, r, γ, c_p, T'_{in}, and ρ'_{in} are all fixed and known. Then ρ, and hence ρW, is a function of W. The product ρW has its maximum value when $W = W_{cr}$. If ρW is less than this maximum value, there are two values of W which will give this value of ρW, one subsonic and the other supersonic. It is desired to find the subsonic value of W corresponding to the given value of ρW. The method used is Newton's method, which converges quadratically.

It is necessary to express ρW as a function of W. The static temperature T may be expressed as a function of W and r by (see ref. 15, eq. (3))

$$\frac{T}{T'_{in}} = 1 - \frac{W^2 + 2\omega\lambda - (\omega r)^2}{2c_pT'_{in}} \tag{B1}$$

With the assumption of isentropic flow

$$\frac{\rho}{\rho'_{in}} = \left(\frac{T}{T'_{in}}\right)^{\frac{1}{\gamma-1}} \tag{B2}$$

and the following equation is obtained:

$$\rho W = \rho'_{in} W \left[1 - \frac{W^2 + 2\omega\lambda - (\omega r)^2}{2c_pT'_{in}}\right]^{\frac{1}{\gamma-1}} \tag{B3}$$

For Newton's method, the derivative with respect to W is needed,

$$\frac{d(\rho W)}{dW} = -\frac{W^2\rho_{in}'}{\gamma RT_{in}'}\left[1 - \frac{W^2 + 2\omega\lambda - (\omega r)^2}{2c_pT_{in}'}\right]^{\frac{2-\gamma}{\gamma-1}} + \rho_{in}'\left[1 - \frac{W^2 + 2\omega\lambda - (\omega r)^2}{2c_pT_{in}'}\right]^{\frac{1}{\gamma-1}} \tag{B4}$$

Suppose that $(\rho W)_{giv}$ is a given value of ρW. A first estimate of W is

$$W_0 = \frac{(\rho W)_{giv}}{\rho_{in}'} \tag{B5}$$

Then, using Newton's method,

$$W_{n+1} = W_n + \frac{(\rho W)_{giv} - \rho(W_n)W_n}{\left.\frac{d(\rho W)}{dW}\right|_{W=W_n}} \qquad n = 0,\ 1,\ 2,\ .\ .\ . \tag{B6}$$

Since the convergence is quadratic, only a few iterations are needed, and the relative change in W_n is an excellent measure of the relative error in W_n. If an estimate for W is available from a previous iteration, this value is used for W_0 instead of using equation (B5). The algorithm given by equation (B6) is done by subroutine DENSTY.

Calculation of Prerotation λ

The input information for the program determines the value of $\lambda = (rV_\theta)_{in}$. The average value of $(\rho W)_{le}$ can be calculated by

$$(\rho W)_{le} = \frac{w}{r_{le}sb_{le}\cos\beta_{le}} \tag{B7}$$

where β_{le} is the average value of β across BM. The value of W can be estimated by dividing this value of $(\rho W)_{in}$ by ρ_{in}'. Then λ can be estimated by

$$\lambda = r_{le}(W_{le}\sin\beta_{le} + \omega r_{le}) \tag{B8}$$

where W_{le} is the average value of W across BM. From this a better value of ρ_{le} is calculated by

$$\rho_{le} = \rho'_{in}\left[1 - \frac{W_{le}^2 + 2\omega\lambda - (\omega r_{le})^2}{2c_p T'_{in}}\right]^{\frac{1}{\gamma-1}} \tag{B9}$$

Use of this value of ρ_{le} gives a better estimate of the value of W_{le}, and then iteration can be used with equations (B8) and (B9) until there is a negligible change in ρ_{le}. This calculation also gives the value of W_{le} along BM. These calculations are performed in PRECAL.

Calculation of Critical Relative Velocity W_{cr}

For reference, the critical relative velocity W_{cr} is calculated at blade leading and trailing edges. This is given by

$$W_{cr}^2 = \frac{2\gamma R}{\gamma + 1} T'' \tag{B10}$$

where

$$T'' = T'_{in} - \frac{2\omega\lambda - (\omega r)^2}{2c_p} \tag{B11}$$

This calculation is performed by PRECAL.

Calculation of Maximum Value of Mass Flow Parameter ρW

The mass flow parameter ρW attains its maximum value when $W = W_{cr}$. For reference, the maximum values of ρW along BM and along FI are computed by the program. The maximum value of ρW is calculated by

$$(\rho W)_{max} = \rho'_{in} W_{cr}\left[1 - \frac{W_{cr}^2 + 2\omega\lambda - (\omega r)^2}{2c_p T'_{in}}\right]^{\frac{1}{\gamma-1}} \tag{B12}$$

where W_{cr} is calculated by equations (B10) and (B11).

Calculation of Flow Angles β along AN and GH

If the radius or stream-channel thickness b is not constant in the meridional direction, the free-stream inlet and outlet flow angles β change along the meridional axis. (By free-stream velocity or flow angle we mean the velocity or angle that would exist at a point of the stream channel based on conservation of angular momentum, either upstream or downstream of blade). The following relations hold for free-stream conditions:

$$\left.\begin{aligned} \tan\beta &= \frac{W_\theta}{W_m} \\ W_\theta &= V_\theta - \omega r \\ rV_\theta &= \text{Constant} \\ W_m &= \frac{w}{\rho b r s} \end{aligned}\right\} \tag{B13}$$

From this we can derive the following equation for the free-stream angle β at any point along the meridional axis, when it is known at some other reference coordinate of $m = m_*$.

$$\tan\beta = \left[\frac{\tan\beta_*}{b_*}\left(\frac{\rho}{\rho_*}\right) + \frac{\omega(r_*^2 - r^2)\rho s}{w}\right]b \tag{B14}$$

Equation (B14) may be used at either inlet or outlet to calculate β_{in} or β_{out}. This requires iteration, since ρ is not known until β is known.

Equation for Leading- and Trailing-Edge Radii

The equation for the leading- and trailing-edge radii is needed. If the radius r were constant,

$$(m - m_*)^2 + r^2(\theta - \theta_*)^2 = R^2 \tag{B15}$$

where R is the leading- or trailing-edge radius and m_*, θ_* are the coordinates of the center of the radius. Since r changes by a relatively small amount on this circle, it was deemed adequate to use this equation with r taken at the leading or trailing edge. Equation (B15) is used by the program to calculate coordinates on the leading- and trailing-edge radii. It is also used to calculate the points of tangency to the spline curves describing the rest of the blade surfaces, and to calculate slopes on the leading- and trailing-edge radii.

Calculation of Surface Length

It is often desired to plot the velocities as a function of blade-surface length. For convenience, the approximate blade-surface length is calculated by the program. The calculation is based on straight-line distances between each vertical grid line on the blade surface. If h_i is the spacing between vertical grid lines, r_i the radius at the i^{th} vertical grid line, and θ_i the coordinate of the i^{th} vertical grid line, the surface length S_n to the n^{th} grid line is approximately

$$S_n = \sum_{i=2}^{n} \sqrt{h_i^2 + \left(\theta_i - \theta_{i-1}\right)^2 \left(\frac{r_i + r_{i-1}}{2}\right)^2} \tag{B16}$$

This may be in error near the leading or trailing edge, but is quite accurate over most of the blade surface.

REFERENCES

1. Stanitz, John D.: Two-Dimensional Compressible Flow in Turbomachines With Conic Flow Surfaces. NACA TR 935, 1949.

2. Stanitz, John D.; and Ellis, Gaylord O.: Two-Dimensional Compressible Flow in Centrifugal Compressors With Straight Blades. NACA TR 954, 1950.

3. Katsanis, Theodore: Computer Program for Calculating Velocities and Streamlines on a Blade-to-Blade Stream Surface of a Turbomachine. NASA TN D-4525, 1968.

4. Vavra, Michael H.: Aero-Thermodynamics and Flow in Turbomachines. John Wiley & Sons, Inc., 1960.

5. Lueders, H. G.: Experimental Investigation of Advanced Concepts to Increase Turbine Blade Loading. I: Analysis and Design. NASA CR-735, 1967.

6. Kramer, James J.; Stockman, Norbert O.; and Bean, Ralph J.: Incompressible Nonviscous Blade-to-Blade Flow Through a Pump Rotor with Splitter Vanes. NASA TN D-1186, 1962.

7. Kramer, James J.; Stockman, Norbert O.; and Bean, Ralph J.: Nonviscous Flow Through a Pump Impeller on a Blade-to-Blade Surface of Revolution. NASA TN D-1108, 1962.

8. Mechtly, E. A.: The International System of Units. Physical Constants and Conversion Factors. NASA SP-7012, 1964.

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

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

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

12. Dellner, Lois T.: A Set of FORTRAN IV Subroutines for Generating Printed Plots. NASA TM X-1419, 1967.

13. Katsanis, Theodore; and McNally, William D.: FORTRAN Program for Calculating Velocities in a Magnified Region on a Blade-to-Blade Stream Surface of a Turbomachine. NASA TN D-5091, 1969.

14. Katsanis, Theodore; and McNally, William D.: Revised FORTRAN Program for Calculating Velocities and Streamlines on a Blade-to-Blade Stream Surface of a Turbomachine. NASA TM X-1764, 1969.

"The aeronautical and space activities of the United States shall be conducted so as to contribute . . . to the expansion of human knowledge of phenomena in the atmosphere and space. The Administration shall provide for the widest practicable and appropriate dissemination of information concerning its activities and the results thereof."

—NATIONAL AERONAUTICS AND SPACE ACT OF 1958

NASA SCIENTIFIC AND TECHNICAL PUBLICATIONS

TECHNICAL REPORTS: Scientific and technical information considered important, complete, and a lasting contribution to existing knowledge.

TECHNICAL NOTES: Information less broad in scope but nevertheless of importance as a contribution to existing knowledge.

TECHNICAL MEMORANDUMS: Information receiving limited distribution because of preliminary data, security classification, or other reasons.

CONTRACTOR REPORTS: Scientific and technical information generated under a NASA contract or grant and considered an important contribution to existing knowledge.

TECHNICAL TRANSLATIONS: Information published in a foreign language considered to merit NASA distribution in English.

SPECIAL PUBLICATIONS: Information derived from or of value to NASA activities. Publications include conference proceedings, monographs, data compilations, handbooks, sourcebooks, and special bibliographies.

TECHNOLOGY UTILIZATION PUBLICATIONS: Information on technology used by NASA that may be of particular interest in commercial and other non-aerospace applications. Publications include Tech Briefs, Technology Utilization Reports and Notes, and Technology Surveys.

Details on the availability of these publications may be obtained from:

SCIENTIFIC AND TECHNICAL INFORMATION DIVISION

NATIONAL AERONAUTICS AND SPACE ADMINISTRATION

Washington, D.C. 20546