

General Disclaimer

One or more of the Following Statements may affect this Document

- This document has been reproduced from the best copy furnished by the organizational source. It is being released in the interest of making available as much information as possible.
- This document may contain data, which exceeds the sheet parameters. It was furnished in this condition by the organizational source and is the best copy available.
- This document may contain tone-on-tone or color graphs, charts and/or pictures, which have been reproduced in black and white.
- This document is paginated as submitted by the original source.
- Portions of this document are not fully legible due to the historical nature of some of the material. However, it is the best reproduction available from the original submission.



Department of Aerospace Engineering University of Cincinnati

EROSION IN RADIAL INFLOW TURBINES - VOLUME V:
COMPUTER PROGRAMS FOR TRACING PARTICLE TRAJECTORIES

By:

W.B. Clevenger, Jr.

and

W. Tabakoff

(NASA-CR-134787) EROSION IN RADIAL INFLOW
TURBINES. VOLUME 5: COMPUTER PROGRAMS FOR
TRACING PARTICLE TRAJECTORIES Final Report
(Cincinnati Univ.) 176 p HC \$7.00 CSCL 21E

N75-29109

G3/07

Unclas
31433

Supported by:

NATIONAL AERONAUTICS AND SPACE ADMINISTRATION

Lewis Research Center

Contract NGR 36-004-055

1. Report No NASA CR-134787		2. Government Accession No.		3. Recipient's Catalog No.	
4. Title and Subtitle Erosion in Radial Inflow Turbines - Volume V: Computer Programs for Tracing Particle Trajectories				5. Report Date May 1975	
				6. Performing Organization Code	
7. Author(s) W.B. Clevenger and W. Tabakoff				8. Performing Organization Report No.	
9. Performing Organization Name and Address Department of Aerospace Engineering University of Cincinnati Cincinnati, Ohio 45221				10. Work Unit No.	
				11. Contract or Grant No. NGR 36-004-055	
12. Sponsoring Agency Name and Address National Aeronautics and Space Administration				13. Type of Report and Period Covered Contractor Report	
				14. Sponsoring Agency Code	
15. Supplementary Notes Final Report. Project Manager, Jeffrey E. Haas, Fluid Systems Components Division, NASA, Lewis Research Center, Cleveland, Ohio 44135					
16. Abstract This report presents the computer programs that have been used to study the trajectories of particles in the radial inflow turbines. Included are descriptions of the general technique that is followed by each of the programs. A set of subroutines that have been developed during the study are described. Descriptions, listings, and typical examples of each of the main programs are included.					
17. Key Words (Suggested by Author(s)) Radial Turbine Erosion Similarity Particulated Flow Computer Programs for Trajectories			18. Distribution Statement Unclassified - unlimited		
19. Security Classif. (of this report) Unclassified		20. Security Classif. (of this page) Unclassified		21. No. of Pages 176	22. Price*

TABLE OF CONTENTS

	<u>Page</u>
SUMMARY	1
INTRODUCTION	2
GENERAL NUMERICAL TECHNIQUES	4
Main Programs	4
Subroutine BOUNCE	9
Subroutine RESTCO	21
Subroutine RNUMBR	25
Subroutines ALOCAT, BLOCAT, and DLOCAT	27
Subroutine POLATE	27
Function RUNGE	28
PROGRAM DESCRIPTIONS	29
PARDIM - Particles in a Vortex	29
SCRL2D - Particles in a Two-Dimensional Scroll	43
STATOR - Particles in the Stators	61
ROTOR - Particles in the Rotor	84
VANPY - Particles in a Turning Vortex	119
DISCUSSION OF RESULTS	147
CONCLUSIONS	148
REFERENCES	149

SUMMARY

This report presents the computer programs that have been used to study the trajectories of particles in the radial inflow turbines. Included are descriptions of the general technique that is followed by each of the programs. A set of subroutines that have been developed during the study are described. Descriptions, listings, and typical examples of each of the main programs are included.

INTRODUCTION

Particle erosion in gas turbine engines has become important because significant decreases in the operating life and rated performance have resulted when these engines are used in dusty environments. For example, the engines in military helicopters, operating at low altitudes and remote landing fields, have significantly shorter life and a more rapid performance deterioration rate than engines operating from hard surfaced landing fields and at higher altitudes. Although these helicopters have main engines which utilize axial flow turbines, some also have auxiliary power sources for special devices which utilize radial turbines, as shown schematically in Figure 1. Radial inflow turbines have also been used on small portable power plants which are also likely to be used in areas where dust ingestion will occur. Radial inflow turbines also are seriously being considered for future use in advanced helicopter engines and transportation vehicles such as trucks, buses, and automobiles. These engines will at times have incomplete filtering of incoming air, leading to the ingestion of erosive-size particles that could seriously degrade engine performance.

The radial turbine engines have, however, a more serious erosion problem than axial-flow turbines. In radial turbines, the heavier particles can experience a radially outward centrifugal force that is greater than the radially inward component of the aerodynamic drag force. In the axial-flow turbine, the centrifugal force acts perpendicular to the aerodynamic drag force. Thus, in radial turbines the heavier particles can be trapped between the stator and rotor, resulting in the particles striking the trailing edges of the stators and the leading edges of the rotor many times. In axial-flow turbines, the particles generally move outward to the tip region, but all particles have a tendency to pass through the turbines.

As a result of the wide interest in using radial turbines and because erosion seems to be more severe in radial turbines, an investigation was sponsored by the NASA Lewis Research Center

to study erosion phenomena in radial inflow turbines and to find ways of eliminating this erosion. Included in this investigation are analytical studies of particle trajectories through a radial inflow turbine and predictions of the effect of blade materials erosion by the action of particles. The results of these investigations will be published in a series of five volumes.

Volumes I through III (1, 2, 3) have dealt with the concept of erosive particle trajectories and studied the approximate velocities and types of impacts that occurred on surface inside a radial turbine as particles moved through the turbine.

Volume III (3) indicated several possible problem areas within a radial turbine. The first is the region at the end of the scroll, where the more rapidly changing radius of curvature causes more moderate angle impacts by the particles. Because of the nature of the motion of particles in the scroll, those particles that do not immediately enter the stator will tend to accumulate and enter a few of the blade passages near the scroll exit. Volume III also revealed that most heavier particles will become trapped in the vortex region of the turbine and repeatedly strike the trailing edges of the stator and the rotor leading edge.

Volume IV (4) presented an analytical study of the rate at which material is removed by ingested dust impinging on the internal surfaces of a typical radial turbine. The study indicates that there are several regions which experience very severe erosion loss, and other regions that experience moderate levels of erosion loss.

The purpose of this report (Volume V) is to present the computer programs that have been developed to trace the particle trajectories through the various parts of the radial inflow turbine. The programs will be useful for any future study of the erosion phenomena that might be considered for a given turbine. Several studies involving computer programs are cited within this report. These programs have been used to determine the gas flow solutions in a given geometry and do not offer a comparison with the trajectory trace programs.

A set of the program source decks on tape is available from COSMIC (Computer Software Management and Information Center), Computer Center, University of Georgia, Athens, Georgia 30601. The programs can be ordered by using the number of this Report as Identification.

GENERAL NUMERICAL TECHNIQUE

Main Programs

Much of the analytical work that has been done in this series of reports has required the numerical solution of the ordinary differential equations that describe the motion of a particle in a gas flow field. These equations are derived in Volume III (3); and their final form which will be integrated numerically is given as Equation (17) in the same reference. These equations are expressed as a set of first order ordinary differential equations by assuming

$$\begin{aligned}
 y_1 &= r \\
 y_2 &= \delta r / \delta t \\
 y_3 &= \theta \\
 y_4 &= \delta \theta / \delta t \\
 y_5 &= z \\
 y_6 &= \delta z / \delta t
 \end{aligned}
 \tag{1}$$

The equations of motion can then be expressed as

$$\frac{\delta y_1}{\delta t} = y_2$$

$$\frac{\delta y_2}{\delta t} = y_1 y_4^2 + 2 y_1 y_4 \dot{\omega} + y_1 \omega^2 + B |v| v_r - g \cos(\theta - \sigma)$$

$$\frac{\delta y_3}{\delta t} = y_4$$

$$\frac{\delta y_4}{\delta t} = \frac{1}{y_1} [-2y_2 y_4 - 2\omega y_2 + B|v|v_\theta + g \sin(\theta + \sigma)]$$

$$\frac{\delta y_5}{\delta t} = y_6$$

$$\frac{\delta y_6}{\delta t} = B|v|v_z - g \sin\phi \quad (2)$$

Where $B = \frac{1}{2} \frac{C_{DA}}{m} = \frac{3}{4} \frac{C_D}{\rho_p D_p}$ for spherical particles. The terms

V_r , V_θ and V_z represent the velocity of the gas relative to the moving particle.

$$V_r = w_r - y_2$$

$$V_\theta = w_\theta - y_1 y_4$$

$$V_z = w_z - y_6 \quad (3)$$

In the solution of these equations, a fourth order Runge-Kutta integration technique, as described in Reference 5 was used. The form of the integration formula is

$$y_{i+1,j} = y_{i,j} + \frac{h}{6} (k_{1j} + 2k_{2j} + 2k_{3j} + k_{4j}) \quad (4)$$

for the differential equation $\frac{dy}{dx} = f(x,y)$

where $k_{1j} = f_j(x_i, y_i)$

$$k_{2j} = f_j(x_i + \frac{1}{2} h, y_i + \frac{1}{2} h k_{1j})$$

$$k_{3j} = f_j(x_i + \frac{1}{2} h, y_i + \frac{1}{2} h k_{2j})$$

$$k_{4j} = f_j(x_i + h, y_i + h k_{3j}) \quad (5)$$

The values of the function f_j corresponding to the equations of motion (2) are:

$$f_1 = v_2$$

$$f_2 = Y_1 Y_4^2 + 2Y_1 Y_4 \omega + Y_1 \omega^2 + B|V|V_r - g \cos(\theta - \sigma)$$

$$f_3 = Y_4$$

$$f_4 = \frac{1}{Y_1} [-2Y_2 Y_4 - 2\omega Y_2 + B|V|V_\theta + g \sin(\theta + \sigma)]$$

$$f_5 = Y_6$$

$$f_6 = B|V|V_z - g \sin \phi$$

In Equation (4), h represents the time increment between integration steps. In most of the work that is presented here, a value of 1×10^{-5} seconds was used. One of the longest trajectories in the scroll was calculated with different time increments to study the deviation in trajectories as the time increment decreased. The results of this numerical experiment are presented in Figure 2 which shows the variations in the angular position of the particle when different time increments were used. At the smaller time steps, the round off errors can be reduced by switching to double precision. As the time increment increases, the truncation errors can only be reduced if a higher order numerical procedure is considered. In this example, the trajectory is approximately 100 cm long and the deviation of the trajectory by one degree would correspond to a linear distance of 0.3 cm. In most of the regions in the

turbine, the total distance traveled by the particle is much shorter than the distances indicated here; so the corresponding error in the position of the particle would also be much smaller. Also indicated in Figure 2 is the amount of central processing unit (CPU) time that was required for the various solutions. As the time increment decreases, the computational time required to complete the solution increases.

In the programs that were used to calculate the gas flow field, the magnitude and direction of the velocities were determined at the grid points. A linear interpolation technique was used to estimate the magnitude and direction of the gas velocity at the points within the grid. An iteration scheme was used to calculate average gas flow field properties which were needed in the particle trajectory calculations at each time increment. Figure 3 illustrates the general procedure that was followed in this iteration scheme. After integrating the equation of motion over the time segment, the magnitude and direction of the gas velocity and the gas properties at the new particle location are found and used to calculate an improved estimate of the average velocity components and average gas properties. The program compares the new estimated average values with the old average values; and, if necessary, reintegrates the equations of motion to find a corrected new location for the particle. Usually the procedure converges in only a few iterations. In a few cases where the particle size was quite small, the procedure failed to converge in 100 iterations. When a smaller time increment was used, the solution could be found beyond the point where the iteration technique failed to converge.

Generally, all the programs listed in this report will calculate several similarity parameters that are useful in relating different particles that have similar trajectories. These similarity parameters are explained in greater detail in Volume I (1) of this series of reports, but it is worthwhile to define the terms that are calculated by the programs.

The characteristic length as calculated in the programs is

$$\delta = \frac{10}{3} \frac{\rho_p D_p}{\rho_g} \quad (7)$$

This parameter can be used to relate trajectories of different particles in equivalent flow fields as long as the Reynolds number of the particle is generally greater than 500. If the Reynolds number drops below 500, the two particles may not follow precisely the same trajectory, but the trajectories will not be significantly different.

The time constant as calculated by the programs is

$$\tau = \frac{\rho_p D_p^2}{18 \mu_g} \quad (8)$$

This parameter generally applies to very small particles because its application is restricted to cases where the particles Reynolds number is less than 1.

An approximate Reynolds number is calculated using critical gas flow properties and one half the gas velocity. This term calculated by the programs is

$$Re_{cr} = \frac{\rho_g^* (V_{cr}/2.) D_p}{\mu_g^*} \quad (9)$$

Although this term was retained in the programs, experience has shown that it is not necessarily typical of the particles Reynolds number.

Several special duty subroutines were used in conjunction with the main program to solve the particle trajectory. The remainder of this section will describe the subroutines that were used. In some cases, slightly different versions of a subroutine were used because of different types of geometry that occur between cases. The source deck listings of the programs contain the main program and all required subroutines.

Subroutine BOUNCE

The subroutine BOUNCE can be used in any program that deals with the trajectories of particles in a flow field where it is necessary to allow the particles to bounce off a surface. Generally, the main program will trace the particle trajectory and, at each new point along the trajectory, the main program should check the position of the particle. When the new position of the particle is found to be outside the boundaries of the contour surrounding the flow field, the main program specifies in ordered arrays the trajectory characteristics of the particle and the orientation of the boundary surface near the location where the particle impact occurs. The main program then calls BOUNCE, which determines the location where the particle trajectory intersects the surface and the time increment required for the particle to travel from the last point of the completed integration to the surface. The subroutine then returns the bounce point, and the time corresponding to the particle motion away from the bounce point. Figure 4 illustrates the surface, particle trajectory intersection.

Referring to Figure 4, the point \bar{P} is the last point along the trajectory that is still in bounds and point $\bar{P}\bar{P}$ is the next point along the trajectory that is out of bounds. Since these two points are vector quantities within the program, the particle velocity is calculated from the following equation:

$$\bar{V} = (\bar{P}\bar{P} - \bar{P})/h \quad (10)$$

Where h is the time increment used in the integration step.

The description of the surface near the location of the bounce is done in the main program by specifying three points that lie on the surface. These points are indicated by the position vectors \bar{A} , \bar{B} , and \bar{C} within the subroutine. The subroutine then determines vectors $\bar{A}\bar{B}$ and $\bar{A}\bar{C}$ which lie in the surface, where

$$\bar{A}\bar{B} = \bar{B} - \bar{A} \quad (11)$$

$$\bar{A}\bar{C} = \bar{C} - \bar{A} \quad (12)$$

Next the subroutine determines the unit vector normal to the surface by calculating the cross product of $\bar{A}\bar{B}$ and $\bar{A}\bar{C}$, and then dividing by the magnitude of the resulting vector. Indicated below is the equation form of this operation.

$$\bar{A}\bar{B} \times \bar{A}\bar{C} = \begin{vmatrix} \bar{i} & \bar{j} & \bar{k} \\ AB_1 & AB_2 & AB_3 \\ AC_1 & AC_2 & AC_3 \end{vmatrix} \quad (13)$$

$$\begin{aligned} &= (AB_2 AC_3 - AB_3 AC_2) \bar{i} \\ &\quad + (AB_3 AC_1 - AB_1 AC_3) \bar{j} \\ &\quad + (AB_1 AC_2 - AB_2 AC_1) \bar{k} \end{aligned} \quad (14)$$

$$\bar{U}\bar{N} = \frac{\bar{A}\bar{B} \times \bar{A}\bar{C}}{|\bar{A}\bar{B} \times \bar{A}\bar{C}|} = UN_1 \bar{i} + UN_2 \bar{j} + UN_3 \bar{k} \quad (15)$$

If $|\bar{A}\bar{B} \times \bar{A}\bar{C}| = 0$, the subroutine returns $NFIX = 0$. This corresponds to a case where points \bar{A} , \bar{B} , and \bar{C} lie on a straight line and hence are not sufficient to describe a surface.

The next step in the solution of the bounce problem is the determination of the point, $\bar{P}\bar{B}$, where the particle bounces off the surface. This is determined by solving simultaneously the equations of the plane given by the unit vector $\bar{U}\bar{N}$ and the point \bar{A} and the straight line connecting the points \bar{P} and $\bar{P}\bar{P}$ of the trajectory. The equation of the plane is

$$UN_1(x-A_1) + UN_2(y-A_2) + UN_3(z-A_3) = 0 \quad (16)$$

The equations for the line trajectory are

$$\frac{x-P_1}{V_1} = \frac{y-P_2}{V_2} = \frac{z-P_3}{V_3} \quad (17)$$

These can be arranged into a set of three equations with three unknowns.

$$\begin{aligned} UN_1 x + UN_2 y + UN_3 z &= D_1 \\ V_2 x - V_1 y &= D_2 \\ V_3 x - V_1 z &= D_3 \end{aligned} \quad (18)$$

where

$$\begin{aligned} D_1 &= UN_1 A_1 + UN_2 A_2 + UN_3 A_3 \\ D_2 &= V_2 P_1 - V_1 P_2 \\ D_3 &= V_3 P_1 - V_1 P_3 \end{aligned} \quad (19)$$

The solution of these equations does not exist when the coefficient determinant is zero. The coefficient determinant is:

$$\begin{vmatrix} UN_1 & UN_2 & UN_3 \\ V_2 & -V_1 & 0 \\ V_3 & 0 & -V_1 \end{vmatrix} = UN_1 V_1^2 + UN_2 V_1 V_2 + UN_3 V_1 V_3 \quad (20)$$

The coefficient determinant will be zero in the special cases that are outlined below.

Case 1. If $UN_1 = 0$, $V_2 = 0$, $V_3 = 0$, then the surface is parallel to the x axis and the particle motion is also parallel to the x axis. This event could not cause a bounce to occur.

Case 2. If $UN_1 = 0$, $UN_2 = 0$, and $V_3 = 0$, then the surface is parallel to the (x,y) plane and the particle motion is also parallel to the (x,y) plane, then the particle does not bounce.

Case 3. If $UN_1 = 0$, $UN_3 = 0$, and $V_2 = 0$, then the surface is parallel to the (x,y) plane and the particle motion is also parallel to the (x,y) plane, and no particle bounce takes place.

Case 4. If $V_1 = 0$, then the x coordinate of P and PB are the same. The solution of the intersection problem then requires use of only one of the equations for the line segment. The new equation set is

$$\begin{aligned} UN_2 y + UN_3 z &= D_4 \\ V_3 y - V_2 z &= D_5 \end{aligned} \tag{21}$$

where

$$\begin{aligned} D_4 &= D_1 - UN_1 \cdot P_1 \\ D_5 &= V_3 P_2 - V_2 P_3 \end{aligned} \tag{22}$$

The solution of this system of equations, with $PB_1 = P_1$ provides the intersection point. In this case, the determinant can be written as

$$(UN_2 V_2 + UN_3 V_3) = 0 \tag{23}$$

It can be equal to zero if $V_2 = 0$, and $UN_3 = 0$ in which case the surface is parallel to the (x,y) plane and the particle moves in the direction parallel to the z axis. Thus, $PB_2 = P_2$ and $PB_3 = A_3$. This may also occur when $V_3 = 0$, or $UN_2 = 0$ in which case the surface is parallel to the (x,y) plane and the particle moves parallel to the y axis. Then $PB_2 = A_2$, and $PB_3 = P_3$. All cases have been considered so far, since the three velocity components V_1 , V_2 and V_3 cannot be all equal to zero.

The subroutine next calculates the distances between $\bar{P}\bar{B}$ and \bar{P} , $\bar{P}\bar{P}$ and \bar{P} , and $\bar{P}\bar{B}$ and $\bar{P}\bar{P}$. These distances can be expressed in equation form as

$$\begin{aligned} DPB &= |\bar{P}\bar{B} - \bar{P}| \\ DPP &= |\bar{P}\bar{P} - \bar{P}| \\ DPPB &= |\bar{P}\bar{B} - \bar{P}\bar{P}| \end{aligned} \tag{24}$$

Figure 5 illustrates a situation that can occur sometimes near a surface. If the main program specified points on surface 2, the incorrect bounce point is determined as indicated by the dashed extension of the trajectory. Similarly, referring to Figure 6, the incorrect bounce point which is obtained if the points on surface 1 are specified in the main program. This type of error is avoided by consistently using surface 1. The distance $DPP = |\bar{P}\bar{P} - \bar{P}|$ is calculated and the subroutine checks to insure if it is greater than the distance $DPPB = |\bar{P}\bar{P} - \bar{P}\bar{B}|$. If $DPP < DPPB$, then the subroutine returns $NFIX = 2$. The main program should be written to recognize this and to switch to the appropriate surface.

The subroutine next uses one of two methods to determine the velocity components of the particle as the particle bounces from the surface. Method 1 is used when $DPB \geq \frac{1}{2} DPP$. In this case, the particle travels over more than half the time segment before the impact occurs. Figure 7 illustrates the technique that is used. The first step in the subroutine is to determine the point $\bar{P}\bar{N}$ which is the normal projection of the point \bar{P} onto the surface. This is found by solving the equation of the plane, Equation (16), and the equations of the line that passes through the point \bar{P} and is normal to the surface:

$$\frac{x-P_1}{UN_1} = \frac{y-P_2}{UN_2} = \frac{z-P_3}{UN_3} \quad (25)$$

The following set of three equations are obtained from the above relations and the equation of the plane:

$$\begin{aligned} UN_1 x + UN_2 y + UN_3 z &= D_1 \\ UN_2 x - UN_1 y &= D_2 \\ UN_3 x - UN_1 a &= D_3 \end{aligned} \quad (26)$$

where

$$\begin{aligned}
 D_1 &= UN_1 A_1 + UN_2 A_2 + UN_3 A_3 \\
 D_2 &= UN_2 P_1 - UN_1 P_2 \\
 D_3 &= UN_3 P_1 - UN_1 P_3
 \end{aligned} \tag{27}$$

The solution of this system of equations gives the desired position vector \bar{PN} . As in the solution of a similar set of equations for the bounce point, there are special cases that cause the determinant of the coefficient matrix to be zero. This determinant is

$$\begin{vmatrix} UN_1 & UN_2 & UN_3 \\ UN_2 & -UN_1 & 0 \\ UN_3 & 0 & -UN_1 \end{vmatrix} = UN_1 (UN_2^2 + UN_3^2 + UN_1^2) \tag{28}$$

The only possible case when this determinant would be zero occurs when $UN_1 = 0$. If this occurs, then $PN_1 = P_1 = x$ and a second system of equations can be determined such that

$$\begin{aligned}
 UN_2 y + UN_3 z &= D_4 \\
 UN_3 y - UN_2 z &= D_5
 \end{aligned} \tag{29}$$

where

$$\begin{aligned}
 D_4 &= D_1 - UN_1 P_1 = D_1 \\
 D_5 &= UN_3 P_2 - UN_2 P_3
 \end{aligned} \tag{30}$$

The solution of this set yields PN_2 and PN_3 . The determinant of the coefficient matrix of this last equation cannot be zero unless the surface unit normal vector is zero, which should not occur.

The portion of the time increment that is needed to travel from \bar{P} to \bar{PB} is given by

$$DTIME = DPB/VPP \tag{31}$$

This time segment is used to calculate the components of the tangential and normal velocities. The tangential velocity is determined from the equation of the line between the points $\bar{P}\bar{N}$ and $\bar{P}\bar{B}$:

$$\bar{P}\bar{B} = \bar{P}\bar{N} + \bar{v}(\Delta t) \quad (32)$$

The incidence tangential velocity components are expressed in the program as:

$$\begin{aligned} v_{t1} &= (PB_1 - PN_1)/DTIME \\ v_{t2} &= (PB_2 - PN_2)/DTIME \\ v_{t3} &= (PB_3 - PN_3)/DTIME \end{aligned} \quad (33)$$

In a similar way, the points \bar{P} and $\bar{P}\bar{N}$ are used to determine the normal component of the incidence velocity vector.

$$\begin{aligned} v_{n1} &= (PN_1 - P_1)/DTIME \\ v_{n2} &= (PN_2 - P_2)/DTIME \\ v_{n3} &= (PN_3 - P_3)/DTIME \end{aligned} \quad (34)$$

The subroutine RESTCO is then called to determine the normal and tangential restitution coefficients. These coefficients are dependent upon the magnitude of the incidence velocity and the incidence angle. ETA(1) is the normal restitution coefficient, and ETA(2) is the tangential restitution coefficient.

With these, the subroutine then determines the velocity components of the particle as it travels away from the point $\bar{P}\bar{B}$.

$$\bar{V}\bar{P} = \text{ETA}(1) \bar{V}_t + \text{ETA}(2) \bar{V}_n \quad (35)$$

Finally, the subroutine calculates the time elapsed, which is given by

$$T = T + DTIME \quad (36)$$

Method 2 is used when $DPB < \frac{1}{2} DPP$. In this case, the particle travels over a very short distance along its trajectory between \bar{P} and $\bar{P}\bar{P}$ before it encounters the surface, including the case when the point P lies in the surface. Figure 8 illustrates the technique that is used.

The procedure is quite similar to Method 1, except that now, $\bar{P}\bar{N}$ is the normal projection of the point PP onto the surface. This point $\bar{P}\bar{N}$ is found by equations that are similar in form to those used in the previous method. The difference is that in the places where coordinates of \bar{P} occurred in the previous equations, the coordinates of $\bar{P}\bar{P}$ now occur. Next the portion of the time increment that would be used to travel from $\bar{P}\bar{B}$ to $\bar{P}\bar{P}$ is determined using

$$DTIME = H - DPB/VPP \quad (37)$$

The points $\bar{P}\bar{P}$, $\bar{P}\bar{B}$, and $\bar{P}\bar{N}$ are used to resolve the velocity into the tangential and normal components, which are given by

$$v_{ti} = (PN_i - PB_i)DTIME \quad i = 1, 2, 3 \quad (38)$$

$$v_{ni} = (PN_i - PP_i)DTIME \quad i = 1, 2, 3 \quad (39)$$

With the restitution coefficients found from RESTCO, the new velocities are determined as in the previous case. With the time correctly incremented to allow for the particle to travel to the surface, given by

$$T = T + DPB/VPP \quad (40)$$

The following pages contain a listing of the subroutine BOUNCE.

```

SUBROUTINE BOUNCF(A,B,C,P,PP,H,T,ETA,NFIX,PP,VP)
DIMENSION A(3),B(3),C(3),P(3),PP(3),V(3),AB(3),AC(3),G(3,3),D(5),
1 P3(3)
DIMENSION GS(3,3),VP(3),UN(3),PM(3)
DIMENSION FTA(2)
DIMENSION PNP(3),VN(4),VT(4)

```

```

C
NFIX=1
DO 10 I=1,3
V(I)=(PP(I)-P(I))/H
AB(I)=B(I)-A(I)
10 AC(I)=C(I)-A(I)
VPP=SQRT(V(1)**2+V(2)**2+V(3)**2)
C DETERMINE UNIT NORMAL TO SURFACE
C
UN(1)=AB(2)*AC(3)-AB(3)*AC(2)
UN(2)=AB(3)*AC(1)-AB(1)*AC(3)
UN(3)=AB(1)*AC(2)-AB(2)*AC(1)
CMAG= SQRT(UN(1)**2+UN(2)**2+UN(3)**2)
IF( ABS(CMAG).GT.1.0E-12) GO TO 20
NFIX=0
WRITE(6,1000)
1000 FORMAT(47H BOUNCF HAS ZERO UNIT VECTOR DESCRIBING SURFACE)
RETURN
20 DO 30 I=1,3
30 UN(I)=UN(I)/CMAG
C
C DETERMINE THE INTERSECTION POINT, PB, OF THE PLANE AND TRAJECTORY.
C
DETA=UN(1)*V(1)**2+UN(2)*V(1)*V(2)+UN(3)*V(1)*V(3)
D(1)=UN(1)*A(1)+UN(2)*A(2)+UN(3)*A(3)
D(2)=V(2)*P(1)-V(1)*P(2)
D(3)=V(3)*P(1)-V(1)*P(3)
DO 40 J=1,3
40 G(1,J)=UN(J)
G(2,1)= V(2)
G(2,2)=-V(1)
G(2,3)=0.0
G(3,1)= V(3)
G(3,2)=0.0
G(3,3)=-V(1)
C
C IF DETERMINANT EQUALS ZERO, GO TO 90
C
IF( ABS(DETA).LE.1.0E-12) GO TO 90
DO 70 K=1,3
DO 50 I=1,3
DO 50 J=1,3
50 GS(I,J)=G(I,J)
DO 60 I=1,3
60 GS(I,K)= D(I)
PR(K)=GS(1,1)*GS(2,2)*GS(3,3)+GS(1,2)*GS(2,3)*GS(3,1)+GS(1,3)
1 *GS(2,1)*GS(3,2)-GS(3,1)*GS(2,2)*GS(1,3)-GS(3,2)*GS(2,3)*GS(1,1)

```

ORIGINAL PAGE IS
OF POOR QUALITY

```

2 -GS(3,3)*GS(2,1)*GS(1,2)
70 PP(K)=PR(K)/DETA
GO TO 100

```

```

C
C IF DETERMINANT EQUALS ZERO, POINT P IS ON SURFACE, P EQUALS PB
C

```

```

80 PB(1)=P(1)
D(4)=D(1)-UN(1)*P(1)
D(5)=V(3)*P(2)-V(2)*P(3)
DETA=-UN(2)*V(2)-UN(3)*V(3)
IF(ABS(DETA).LT.1.0E-12) GO TO 85
PB(2)=(-D(4)*V(2)-D(5)*UN(3))/DETA
PB(3)=(UN(2)*D(5)-V(3)*D(4))/DETA
GO TO 100

```

```

85 IF(ABS(V(3)).GT.1.0E-12) GO TO 90
PB(2)=P(2)
PB(3)=A(3)
GO TO 100

```

```

90 PB(2)=A(2)
PB(3)=P(3)

```

```

100 CONTINUE
DPP=SQRT((PP(1)-P(1))**2+(PP(2)-P(2))**2+(PP(3)-P(3))**2)
DPB=SQRT((PB(1)-P(1))**2+(PB(2)-P(2))**2+(PB(3)-P(3))**2)
DPPB=SQRT((PP(1)-PB(1))**2+(PP(2)-PB(2))**2+(PP(3)-PB(3))**2)
IF((DPPB.LT.DPP).AND.(DPB.LT.DPP)) GO TO 103
NPIX=2
RETURN

```

```

103 CONTINUE
IF(DPB.LT.(0.5*DPP)) GO TO 180

```

```

C
C DETERMINE THE INTERSECTION POINT, PN, OF THE SURFACE NORMAL THRU P
C IF DETERMINANT EQUALS ZERO, GO TO 140
C

```

```

DETA=UN(1)**3+UN(1)*UN(2)**2+UN(1)*UN(3)**2
IF(ABS(DETA).LT.1.0E-12) GO TO 140
D(2)=P(1)*UN(2)-P(2)*UN(1)
D(3)=P(1)*UN(3)-P(3)*UN(1)
G(2,1)=UN(2)
G(2,2)=-UN(1)
G(2,3)=0.0
G(3,1)=UN(3)
G(3,2)=0.0
G(3,3)=-UN(1)
DO 130 K=1,3
DO 110 I=1,3
DO 110 J=1,3

```

```

110 GS(I,J)=G(I,J)
DO 120 I=1,3

```

```

120 GS(I,K)=D(I)
PN(K)=GS(1,1)*GS(2,2)*GS(3,3)+GS(1,2)*GS(2,3)*GS(3,1)+GS(1,3)
1 *GS(2,1)*GS(3,2)-GS(3,1)*GS(2,2)*GS(1,3)-GS(3,2)*GS(2,3)*GS(1,1)
2 -GS(3,3)*GS(2,1)*GS(1,2)

```

```

130 PN(K)=PN(K)/DETA

```

ORIGINAL PAGE IS
OF POOR QUALITY

```

      GO TO 160
140  PN(1)=P(1)
      D(4)=D(1)
      D(5)=UN(3)*P(2)-UN(2)*P(3)
      DELTA=-UN(2)**2-UN(3)**2
      PN(2)=(-D(4)*UN(2)-D(5)*UN(3))/DELTA
      PN(3)=(UN(2)*D(5)-UN(3)*D(4))/DELTA
160  CONTINUE
C
C   DETERMINE PORTION OF TIME SEGMENT USED TO TRAVEL FROM P TO PB.
C
      DTIME=DPR/VPP
      IF(DTIME.LT.H) GO TO 163
      WRITE(6,1010)
1010  FORMAT(24H DTIME IS GREATER THAN H)
      DTIME=0
      RETURN
C
C   EXTEND LINE PN-PB THE PROPER DISTANCE TO FIND PNP.
C   THEN EXTEND A LINE NORMAL TO THE SURFACE FROM PNP TO GET THE POINT
C   AFTER BOUNCE, PP.
C   FIND VELOCITY COORDINATES BASED ON PP, PB AND TIME REMAINING IN
C   SEGMENT.
C
163  DO 165 I=1,3
      VT(I)=(PP(I)-PN(I))/DTIME
165  VN(I)=(PN(I)-P(I))/DTIME
      VT(4)= SQRT(VT(1)**2+VT(2)**2+VT(3)**2)
      VN(4)= SQRT(VN(1)**2+VN(2)**2+VN(3)**2)
      CALL REFCO(VN(4),VT(4),FTA)
      DO 170 I=1,3
      PNP(I)=PP(I)+FTA(2)*VT(I)*(H-DTIME)
      PP(I)=PNP(I)-FTA(1)*VN(I)*(H-DTIME)
170  VP(I)=(PP(I)-PB(I))/(H-DTIME)
      T=T+H
      RETURN
C
C   IF POINT P LIES ON SURFACE, USE POINT PP TO DETERMINE AFTER
C   BOUNCE STATE.
C
180  CONTINUE
      DELTA=UN(1)**3+UN(1)*UN(2)**2+UN(1)*UN(3)**2
      IF(ABS(DELTA).LE.1.0E-12) GO TO 220
      D(2)=PP(1)*UN(2)-PP(2)*UN(1)
      D(3)=PP(1)*UN(3)-PP(3)*UN(1)
      G(2,1)= UN(2)
      G(2,2)=-UN(1)
      G(2,3)=0.0
      G(3,1)=UN(3)
      G(3,2)=0.0
      G(3,3)=-UN(1)
      DO 210 K=1,3
      DO 190 I=1,3

```

ORIGINAL PAGE IS
OF POOR QUALITY

```

    DO 190 J=1,3
190 GS(I,J)=C(I,J)
    DO 200 I=1,3
200 GS(I,K)=C(I)
    PN(K)=GS(1,1)*GS(2,2)*GS(3,3)+GS(1,2)*GS(2,3)*GS(3,1)+GS(1,3)
    1 *GS(2,1)*GS(3,2)-GS(3,1)*GS(2,2)*GS(1,3)-GS(3,2)*GS(2,3)*GS(1,1)
    2 -GS(3,3)*GS(2,1)*GS(1,2)
210 PN(K)=PN(K)/DETA
    GO TO 240
220 PN(1)=PP(1)
    D(4)=D(1)
    D(5)=UN(3)*PP(2)-UN(2)*PP(3)
    DETA=-UN(2)*UN(2)-UN(3)*UN(3)
    PN(2)=(-D(4)*UN(2)-D(5)*UN(3))/DETA
    PN(3)=(UN(2)*D(5)-UN(3)*D(4))/DETA
240 CONTINUE
C
C DETERMINE PORTION OF TIME SEGMENT REMAINING FOR TRAVEL FROM PB TO PP.
C
    DTIME=H-DPB/VPP
    IF(DTIME.GT.1.0E-12) GO TO 245
    WRITE(6,1020)
1020 FORMAT(21H DTIME LESS THAN ZERO)
    NFIX=0
    RETURN
C
C DETERMINE THE PROPER DISTANCE ALONG PN-PB TO FIND PNP.
C THEN EXTEND A LINE NORMAL TO THE SURFACE FROM PNP TO GET THE POINT
C AFTER BOUNCE, PP.
C FIND VELOCITY COORDINATES BASED ON PP, PB, AND THE TIME H.
C
245 DO 250 I=1,3
    VT(I)=(PN(I)-PB(I))/DTIME
250 VN(I)=(PP(I)-PN(I))/DTIME
    VT(4)= SORT(VT(1)**2+VT(2)**2+VT(3)**2)
    VN(4)= SORT(VN(1)**2+VN(2)**2+VN(3)**2)
    CALL RESTOR(VN(4),VT(4),ETA)
    DO 260 I=1,3
    PNP(I)=PB(I)+ETA(2)*VT(I)*DTIME
    PP(I)=PNP(I)-ETA(1)*VN(I)*DTIME
260 VP(I)=(PP(I)-PB(I))/DTIME
    T=T+H
    RETURN
END

```

ORIGINAL PAGE IS
OF POOR QUALITY

Subroutine RESTCO

It was necessary to develop expressions for the normal and tangential restitution coefficients to describe the momentum loss experienced by a particle when it bounces off a surface. Initially, constant values of restitution coefficients were used. As a result, the particles that struck the surfaces many times, lost momentum with each bounce and eventually came to rest against the surface. This is unrealistic to occur, and therefore the available data on restitution coefficients was reviewed and the information obtained used to develop a set of empirical equations that would describe the restitution coefficient for all incidence velocities and all incidence angles.

Data given by Grant (6) and Ball (7) indicate that the restitution coefficients were functions of both the incidence angle and the particle incidence velocity. The data given by Ball (7) is perhaps more realistic because it applies to Titanium and Stainless Steel alloys which are more typical of blade materials that are used in radial inflow turbines. However, this data is not as extensive as the data given by Grant (6) for an Aluminum alloy. Because the more extensive data gave a better description of the variations that could be expected, the aluminum alloy data was used.

The empirical expressions arrived at for the normal and tangential restitution coefficients are

$$\begin{aligned}\eta_N(\beta, V) &= 1.0 - \psi_1(\beta) \psi_2(V) \\ \eta_T(\beta, V) &= 1.0 - \phi_1(\beta) \phi_2(V)\end{aligned}\tag{41}$$

In this expression, η_N and η_T are the normal and tangential restitution coefficients and ψ_1 , ψ_2 , ϕ_1 , ϕ_2 are empirical functions which go to zero as the argument goes to zero.

Figure 9 shows the data obtained by Grant (6) for the normal restitution coefficient variation as the impingement angle changes. An assumption was made that ψ_2 is one when the velocity is 76.2 meters/sec., and then a polynomial curve was fitted to the data

points. Table 1 gives the coefficients of this polynomial for the normal restitution coefficients. The variation of the normal restitution coefficient with the incidence velocity is shown in Figure 10. The experimental data points of Reference 6 are shown in the same Figure. It was assumed that the normal restitution coefficient increases to one as the incidence velocity decreases to zero. In order to develop the empirical equation for ψ_2 , the value of ψ_1 at 45° was used. A value of ψ_2 equal to one was taken for incidence velocity of 76.2 meters/sec. The following expression was developed to express ψ_2 :

$$\psi_2 = 0.65(1.0 - e^{-V/24.5}) \quad (42)$$

Equation (42) and the polynomial expression for the variations due to the incidence angle were combined into a complete empirical formula for the normal restitution coefficient. Figure 11 illustrates the variation in the restitution coefficient with the incidence velocity and the incidence angles, according to the empirical equation.

In a similar way, the data given in Reference 6 for the tangential restitution coefficients at an incidence velocity of 76.2 meters/sec., was fitted with a polynomial expression as shown in Figure 12. In addition to the experimental data points that were used, it was assumed that the tangential restitution coefficient was equal to one for incidence angles of 0° and 90° . The last two values were used, with the experimental data and these points were also used in the evaluation of the polynomial coefficients which are given in Table 2. Data on the variation of the tangential restitution coefficient with incidence velocity is indicated in Figure 13. It was assumed that the restitution coefficient goes to one as the tangential velocity goes to zero, this and the two points of the experimental data showed linear variation. Figure 14 shows the family of curves of the tangential restitution coefficient versus incidence angles for different incidence velocities.

Because of the linear variation in restitution coefficient with incidence velocities, very low restitution coefficients are calculated for very high incidence velocities. A lower limit of 0.1 was placed on the allowable value of the restitution coefficient as indicated in Figure 14.

The next page contains a listing of the subroutine RESTCO, which receives the normal and tangential velocity components from the calling program and returns the proper values of the restitution coefficients.

SUBROUTINE RESTCO(VN,VT,ETA)

DIMENSION ETA(2),A(10),R(10)

DATA A/1.8190,5.1171,-49.2529,131.03528,-174.4249,117.8723,
1 -24.47885,-16.7297,11.2300,-1.979996/
DATA R/5.7215,-41.8808,178.1685,-424.3882,572.7631,-406.6625,
1 87.1428,70.6511,-50.4982,9.6767/

C DATA IN THIS SUBROUTINE CORRESPONDS TO VELOCITIES IN FT/SEC.
C MATERIAL TYPICAL OF ALUMINUM AND SILICON PARTICLES.
C ETA(1) IS THE NORMAL RESTITUTION COEFFICIENT.
C ETA(2) IS THE TANGENTIAL RESTITUTION COEFFICIENT.

RETA=ATAN2(VN,VT)

V=SQRT(VN**2+VT**2)

PHIONE=V/250.00

PHITWO= A(1)*RETA+A(2)*RETA**2+A(3)*RETA**3+A(4)*RETA**4
1 +A(5)*RETA**5+A(6)*RETA**6+A(7)*RETA**7+A(8)*RETA**8
2 +A(9)*RETA**9+A(10)*RETA**10

PHI=PHIONE*PHITWO

IF(PHI.GT.0.90) PHI=0.90

ETA(2)=1.000-PHI

PSIONE=1.0000-EXP(-V/36.000)

PSITWO= R(1)*RETA+R(2)*RETA**2+R(3)*RETA**3+R(4)*RETA**4
1 +R(5)*RETA**5+R(6)*RETA**6+R(7)*RETA**7+R(8)*RETA**8
1 +R(9)*RETA**9+R(10)*RETA**10

PSI=PSIONE*PSITWO

IF(PSI.GT.0.900) PSI=0.900

ETA(1)=1.000-PSI

RETURN

END

Subroutine RNUMBR

Another subroutine that is used throughout this set of programs is the subroutine RNUMBR. This subroutine uses the Reynolds number determined by the calling program to find the drag coefficient for a sphere. The equations which are used to describe the drag coefficient are:

$$C_D = 4.5 + \frac{24}{Re} \quad Re < 1.0$$

$$C_D = 28.5 - 24.0 (\text{Log } Re) + 9.0682 (\text{Log } Re)^2$$

$$- 1.7713 (\text{Log } Re)^3 + 0.1718 (\text{Log } Re)^4 - 0.0065 (\text{Log } Re)^5$$

$$1.0 < Re < 3000$$

$$C_D = 0.4 \quad 3000 < Re < 2.5 \times 10^{-5} \quad (43)$$

This subroutine includes a factor DGFC, which can be used to determine the drag coefficient for non-spherical particles. A listing of this subroutine follows.

Program Listing RNUMBR

```
SUBROUTINE RNUMBR(RFNOLD, DGFC, CD)
IF( ABS(RFNOLD).LT.1.0E-12) RFNOLD=1.0E-12
IF(RFNOLD.LT.1.0) GO TO 26
IF((RFNOLD.GE.1.0).AND.(RFNOLD.LT.1.0E3)) GO TO 27
CD=DGFC*0.4
RETURN
26 CD=DGFC*(4.5+24.0/RFNOLD)
RETURN
27 ARE=ALOG(RFNOLD)
CD=(28.5-24.0*ARE+9.0682*ARE**2-1.7713*ARE**3+0.1718*ARE**4
1 -0.0065*ARE**5)*DGFC
RETURN
END
```

Subroutines ALOCAT, BLOCAT, and DLOCAT

This series of subroutines are used to determine the location of the particle within a grid pattern so that the gas velocities and flow properties can be determined. Although each subroutine does approximately the same thing, it was found that each flow geometry required slightly different forms of the subroutine. Most of the geometries used in this study are such that the gas flow is described at nodes along the quasi-orthogonals of the flow. Generally, the indexes that locate the nodes are in increasing order from inlet to exit. All of these subroutines check each orthogonal to find the first orthogonal beyond the particle location. The subroutine then sweeps from hub to shroud, or from blade to blade along the stream surface until it determines the index of the streamline just beyond the particle location, then specifies the four grid points surrounding the particle. The subroutines check if the particle is out of bounds. If the particle is in bounds, then NOWT = 0, otherwise the code work NOWT is set equal to a number that depends on the boundary. These subroutines are included with the individual program listings.

Subroutine POLATE

This subroutine interpolates linearly the properties of the gas flow at the location of the particle within a grid, using the known flow properties at the four grid points that surround the particle. Figure 15 illustrates the grid points surrounding the particles that determined by the locating subroutines. The distances from the orthogonals to the particle location, D_1 and D_2 , are determined first. Next the program calculates the distances from AA and AB to the particle location. All of these distances are used to interpolate the value of any flow property A. The same method is used to determine the r, z coordinates of the properties. The subroutine returns this property to the main program as AP.

There are minor differences in the subroutines that are listed with each of the programs. These minor differences occur because of the significant differences in the geometries between different flow fields. For example, the STATOR program uses an $r-\theta$ grid while the ROTOR program uses an $r-z$ grid. However, the subroutines all follow a procedure similar to the one outlined here.

Function RUNGE

This function uses a 4th order Runge-Kutta technique to integrate a system of simultaneous first order ordinary differential equations over one time increment. A more complete description of the subroutine and the input and output quantities can be found in Reference 5. A listing of the subroutine is included in that reference.

PROGRAM DESCRIPTIONS

The following sections will describe the programs, their input and output and other information that will aid in understanding their use. A complete listing of the program is included with a sample set of data to demonstrate the program output.

PARDIM - Particles in a Vortex

This program integrates the equations of motion to determine the trajectory of the particle in inward flowing free vortices, or whirling flows that have no radial components but do have axial components. The force of gravity acting on the particle can be included in cases where gravity must be considered. The solution is essentially a three dimensional one, but instead of allowing particles to bounce, the integration stops when the particle passes boundaries of the fluid flow. The cylindrical coordinates, r , θ and z , are used as indicated in Figure 16.

Method

Figure 17 is a flow diagram of the program PARDIM. The subroutine, VORTEX is used to provide the three components of the gas velocity and the necessary gas properties within the boundaries of the flow field. The function RUNGE is used to integrate the equations of motion. The general iteration technique that is used to calculate the average values of the gas properties at the particle location, and the method used to integrate the equations of motion of the particle have been explained before.

The first call of subroutine VORTEX allows the general characteristics of the gas flow to be introduced into the subroutine. Subsequent calls return the velocity components, static gas temperature and density that are the solution of the free vortex flow at the radial location of the particle. The equations that govern the gas flow field are; conservation of momentum, $\lambda = rV_u = \text{constant}$, the conservation of mass, $\frac{w}{2\pi b} = \rho V_r r = \text{constant}$, and isentropic flow relations.

The solution for the mass flow in the subroutine VORTEX uses an iteration technique that will sometimes fail to converge when the velocities become large. If this occurs, the subroutine prints "VORTEX FAILED TO CONVERGE AT LOCATION R = , " and then sets a parameter that causes the main program to proceed to the next data set. A second message can sometimes be printed by the same subroutine if the flow of gas in a compressible free vortex has a supersonic solution. The message "CHOKED FLOW" will be printed in such case.

Input

There are two sets of input. The first set specifies the nature of the gas flow and consists of 4 cards at the front of the input data. This program is written so than any consistent system of units may be used.

The first four data cards take the following form.

DANGLE, TMAX, RMIN, RMAX, ANGMAX, ZMAX	(6F10.4)
VISREF, TREF, TSUT, GAM, RGAS, DGFC	(E20.5, 5F10.4)
RIN, VRIN, VUIN, VZIN, TT, RHOT	(5F10.4, E10.4)
ALPHA, BETA, GRAVITY	(3F10.4)

These variables are defined later.

The second set of data cards specifies the particle size and density and the initial position and velocity of the particle. Each particle can be described by four input data cards which take the following form.

TITLE	(18A4)
(YR(I), I = 1,6)	(6F10.6)
RHOP, DIAP	(F10.6, E10.4)
NSTEP, H	(I10, E10.4)

For studies that are done with multiple particles, additional second sets of input data cards may be stacked together. When the program completes one set it goes to the next set automatically. An explanation of the input variables of both sets follows.

DANGLE - Data is printed every DANGLE degrees.

TMAX - The program truncates the solution when time exceeds TMAX seconds and goes to the next particle.

RMIN - The program stops and goes to the next data set when the particle radial location is smaller than RMIN. (Length units).

RMAX - The program stops and goes to the next data set when the particle radial location is larger than RMAX. (Length units).

ANGMAX - The program stops and goes to the next data set when the particle angular position is greater than ANGMAX in degrees.

ZMAX - The program stops and goes to the next data set when the particle axial position is greater than ZMAX. (Length units).

VISREF - Reference viscosity corresponding to TREF. Used in Sutherland's Law. (Mass/Length x Time).

TREF - Reference temperature corresponding to VISREF. Used in Sutherland's Law. (Absolute degrees).

TSUT - Constant used in Sutherland's Law. (198.6°R or 110°K).

GAM - Ratio of specific heats.

RGAS - Gas constant. ($\text{Length}^2/\text{Time}^2 \text{ Deg. Abs.}$).

DGFC - Drag Factor - The spherical drag coefficient based on Reynolds Number is multiplied by DGFC. Except in very special cases, this should be 1.0.

RIN - Radius at inlet of the gas stream. (Length).

VRIN, VUIN, VZIN - Radial, tangential, and axial components of the gas velocity at RIN. In an inward flowing vortex, the radial component should be negative. (Length/Time).

TT - Stagnation inlet temperature. (Degrees Abs.).

RHOT - Stagnation inlet density. ($\text{Mass}/\text{Length}^3$).

ALPHA - Angle of Z-axis with respect to the horizontal. (Degrees). See Figure 18.

BETA - Angle of the turbine verticle axis with respect to the gravitational vertical. (Degrees). See Figure 18.

GRAVITY - Acceleration of Gravity. (Length/Time²).

TITLE - The first card is reproduced at the top of the first page of output for each particle. Any statement in columns 2 to 72 will be reproduced.

YR(1) - Initial particle radial position. (Length).

YR(2) - Initial particle radial velocity. (Length/Time). For inward moving particle, the radial component is negative.

YR(3) - Initial particle angular position, θ . (Degrees).

YR(4) - Initial particle angular velocity, $\dot{\theta}$. (Radians/Sec).

YR(5) - Initial particle axial position. (Length).

YR(6) - Initial particle axial velocity. (Length/Time).

RHOP - Particle density. (Mass/Length³).

DIA - Particle diameter. (Length).

NSTEP - An integer that is not used in this program.

H - Integration time increment. (Time).

Output

The first part of the output is an echo check of the first set of data cards which are used to describe the gas flow field. Such data checks are useful in correcting key punch mistakes on the input cards. After initializing these variables, the program calculates and prints the critical gas velocity. This critical velocity is determined based on the gas stagnation temperature. The following print out starts on the next page, and the printed output data for each particle starts at a new page. The first part of the printed output is an echo check of the input data as punched on the second set of data cards. After initializing the variables, the program calculates and prints several similarity parameters that are useful in relating particles that have similar trajectories. Next, the particle position and velocities at every DANGLE degrees are printed. The parameters appearing in the output, that were not defined before in the input data, are listed below.

DELTA - Characteristic length as given in Equation 7. (Length).

TAU - Time constant as given in Equation 8. (Time).

RECR - Reynolds number as given in Equation 9.

T - Time. (Time).

RENOLDS - Reynolds number for the particle at this point.

R/RIN - Normalized radial position of the particle.

STREAMLINE - Normalized radial coordinate of an incompressible flow streamline that would exist in a free vortex starting at RIN with the same initial velocity components given as input. Such a streamline will follow the equation $R/RIN = e^{(\theta \tan \alpha_1)}$ where $\alpha_1 = \arctan (VRIN/VUIN)$ ⁽⁸⁾. The subroutine VORTEX provides a compressible solution, thus STREAMLINE is only provided for comparison purposes.

When the solution is complete, the program writes all the trajectory information at the last point. Also printed are, M, the iteration counter and L, the time increment counter. If the air velocities fail to converge in 100 iterations, the program stops, prints the trajectory data from the last iteration and M = 101.

C DIMENSIONAL SOLUTION AND PRINT OUT FOR PARTICLE TRAJECTORIES.
 C ANY CONSISTENT SYSTEM OF UNITS.

C

```

DIMENSION VR(4),VU(4),YR(6),FR(6),YRS(6),VZ(4),TEMP(2),PHO(2)
DIMENSION STATE(18)
INTEGER RUNGE
READ(5,1000) DANGLE,TMAX,RMIN,RMAX,ANGMAX,ZMAX
WRITE(6,2000) DANGLE,TMAX,RMIN,RMAX,ANGMAX,ZMAX
READ(5,1100) VISPEF,TREF,TSUT,GAM,PGAS,DGFC
WRITE(6,2100) VISREF,TREF,TSUT,GAM,PGAS,DGFC
READ(5,1200) RIN,VRIN,VUIN,VZIN,TT,RHOT
WRITE(6,2200)RIN,VRIN,VUIN,VZIN,TT,RHOT
READ(5,1000) ALPHA,BETA,GPAVTY
WRITE(6,2510) ALPHA,BETA,GRAVTY
IF(ABS(VUIN).LT.1.0E-10) VUIN=1.0E-10
TALPH=VRIN/VUIN
SALPH=SIN(ALPHA/57.29578)
BETA=BETA/57.29578
CVDR=1.0/(GAM-1.0)
CPR= GAM/(GAM-1.0)
ABC=(GAM-1.0)/(GAM+1.0)
VCR=SQRT(2.0*RGAS*GAM*TT/(GAM+1.0))
WRITE(6,2400)VCR
DO 205 NUMB=1,100
READ(5,1400) (STATE(I),I=1,18)
WRITE(6,2700)(STATE(I),I=1,18)
READ(5,1300) (YR(I),I=1,6)
WRITE(6,2300) (YR(I),I=1,6)
READ(5,1500) RHOP, DIA
WRITE(6,2110) RHOP, DIA
READ(5,1510) NSTEP,H
WRITE(6,2120)NSTEP,H

```

C

C CONVERT TO NON-DIMENSIONAL QUANTITIES AND INITIALIZE

C

```

RHOCR=RHOT*(2.0/(GAM+1.0))**CVDR
DELTA=RHOP*DIA/RHOCR/0.3
TCR=TT*2.0/(GAM +1.0)
VISCR=VISREF*((TCR/TREF)**1.5)*(TREF+TSUT)/(TCR+TSUT)
TAU=RHOP*DIA**2/18.0/VISCR
RECF=DIA*RHOP*VCR/VISCR/2.0
WRITE(6,2900) DELTA,TAU,RECF
WRITE(6,2410)
TEMP(2)=TT
PHO(2)=RHOT
YR(3)=YR(3)/57.29577
VR(1)=VRIN
VU(1)=VUIN
VZ(1)=VZIN
N=6
L=-1
M=1
T=0.0

```

```

    ANGLE=YR(3)
    TS=T
    DO 10 I=1,4
    VR(I)=VR(1)
    VZ(I)=VZ(1)
10  VU(I)=VU(1)
    DO 20 I=1,N
20  YPS(I)=YR(I)
    CALL VORTEX(L,VRIN,VUIN,VZIN,RIN,TEMP,RHO,GAM,RGAS)
    L=L+1
    VISTAR=VISPEFF*((TEMP(1)/TREF)**1.5)*(TPEFF+TSUT)/(TEMP(1)+TSUT)
C
C  INTEGRATE USING RUNGE-KUTTA METHOD
C
25  VDIFF=SQRT((VR(2)-YR(2))**2+(VU(2)-YR(1)*YR(4))**2
1   +(VZ(2)-YP(6))**2)
    RENOLD=RHC(1)*VDIFF*DIA/VISTAR
    CALL RNUMPR(RENOLD,1.0,CD)
    BCON=RHC(1)*CD/RHOP/DIA/1.166667
30  IF(RUNGE(N,YR,FR,T,H).NE.1) GO TO 40
    FR(1) =      YR(2)
    FR(2) =      YR(1)*YR(4)**2+BCON*VDIFF*(VR(2)-YR(2))
1   -GRAVITY*COS(YR(3)+BETA)
    FR(3) =      YR(4)
    FR(4) =      -2.0*YR(2)*YR(4)/YR(1)+BCON*VDIFF
1   *(VU(2)-YR(1)*YR(4))/YR(1)
2   +GRAVITY*SIN(YR(3)+BETA)
    FR(5) =      YP(6)
    FR(6) =      BCON*VDIFF*(VZ(2)-YR(6))
1   -GRAVITY*SALPH
    GO TO 30
40  CALL VORTEX(L,VR(4),VU(4),VZ(4),YR(1),TEMP,RHO,GAM,RGAS)
    IF(L.EQ.10000) GO TO 205
C
C  TEST AIR VELOCITY VALUES USED, IF INCORRECT, RESET INTEGRATION
C  VALUES AND GO TO 25, IF CORRECT, GO TO 50
C
    IF((ABS(VR(4)-VR(3)).LT.1.0E-4).AND.(ABS(VU(4)-VU(3)).LT.1.0E-4)
1   .AND.(ABS(VZ(4)-VZ(3)).LT.1.0E-4)) GO TO 50
    VR(2)=(VR(4)+VR(1))/2.0
    VU(2)=(VU(4)+VU(1))/2.0
    VZ(2)=(VZ(4)+VZ(1))/2.0
    VP(3)=VR(4)
    VU(3)=VU(4)
    VZ(3)=VZ(4)
    T=TS
    DO 45 I=1,N
45  YR(I)=YPS(I)
    IF(M.GT.100) GO TO 200
    M=M+1
    GO TO 25
C
C  COMPLETE INTEGRATION STEP. IF REQ'D, WRITE OUTPUT. GO TO 30

```

```

50 M=1
   L=L+1
   TS=T
   THETA=YR(3)*57.29578
   DO 60 I=1,M
60 YP(I)=YR(I)
   VR(2)=1.5*VR(4)-0.5*VR(1)
   VP(3)=2.0*VR(4)-VR(1)
   VR(1)=VP(4)
   VR(4)=VR(3)
   VU(2)=1.5*VU(4)-0.5*VU(1)
   VV(3)=2.0*VU(4)-VU(1)
   VU(1)=VU(4)
   VU(4)=VU(3)
   VZ(2)=1.5*VZ(4)-0.5*VZ(1)
   VZ(3)=2.0*VZ(4)-VZ(1)
   VZ(1)=VZ(4)
   VZ(4)=VZ(3)
   VISTAR=VISREF*((TEMP(1)/TRFF)**1.5)*(TRFF+TSUT)/(TEMP(1)+TSUT)
80 IF((T.GT.TMAX).OR.(YR(1).LT.RMIN).OR.(YR(1).GT.PMAX)
   I .OR.(THETA.GT.ANGMAX).OR.(YR(5).GT.ZMAX)) GO TO 200
   IF((THETA-ANGLE).LE.0.0) GO TO 30
   RTD=YR(4)
   SLINE=EXP(YR(3)*TALPH)
   RORIN=YR(1)/RIN
   WRITE(6,2500) T,YR(1),YR(2),THETA,RTD,YR(5),YR(6),RENULD,RORIN,
   I SLINE
   ANGLE=ANGLE+DANGLE
   GO TO 25
200 CONTINUE
   RTD=YR(4)
   SLINE=EXP(YR(3)*TALPH)
   RORIN=YR(1)/RIN
   WRITE(6,2600) T,YR(1),YR(2),THETA,RTD,YR(5),YR(6),RENULD,RORIN,
   I SLINE
   WRITE(6,2800) M,L
205 CONTINUE
1000 FORMAT(7F10.4)
1100 FORMAT(F20.5,5F10.4)
1200 FORMAT(5F10.4,F10.4)
1300 FORMAT(6F10.6)
1400 FORMAT(18A4)
1500 FORMAT(F10.6,F10.4)
1510 FORMAT(I10,E10.3)
2000 FORMAT(1H0,7X,7HD-ANGLE,10X,5HT-MAX,10X,5HR-MIN,10X,5HR-MAX,6X,
   I 9HMAX-ANGLE,10X,5HZ-MAX,/, (6F15.4))
2100 FORMAT(1H0,5X,14HREF. VISCOSITY,6X,9HREF. TEMP,2X,13HSUTHERLANDS T
   I , 10X,5HCAMMA,6X,9HGAS CONST,4X,11HDPAG FACTOR,/, (F20.5,5F15.4))
2110 FORMAT(1H0,10X,4HRHDP,12X,3HPIA,/, (F15.6,F15.4))
2120 FORMAT(1H0, 9X,5HNSTEP,14X,1HH,/, (I15,F15.3))
2200 FORMAT(1H0,11X,3HRIN,11X,4HVRIN,11X,4HVUIN,11X,4HVZIN,13X,2HTT,
   I 11X,4HPHOT,/, (5F15.4,F15.4))

```



```

2300 FORMAT(1H0, 9X,5HYR(1),10X,5HYP(2),10X,5HYR(3),10X,5HYR(4),10X,
1 5HYR(5),10X,5HYR(6),/, (6F15.6))
2400 FORMAT(1H0,19HCRTICAL VELOCITY =,F10.4)
2410 FORMAT(1H0,8X,1HT, 7X,5HYR(1), 7X,5HYP(2), 7X,5HYR(3), 7X,5HYR(4),
1 7X,5HYP(5), 7X,5HYR(6), 5X,7HRENOLDS, 7X,5HR/RIN, 2X,10HSTREAMLINE)
2500 FORMAT(1H ,F10.2,3F12.5,F12.2,2F12.5,F12.3,2F12.5)
2510 FORMAT(1H0,10X,5HALPHA,11X,4HBETA,2X,13HACC. OF GRAV.,/, (3F15.4))
2600 FORMAT(1H0,F7.3,F12.4,2F13.4,3F12.4,F13.4,F10.4,F12.4)
2700 FORMAT(1H1,/,18A4)
2800 FORMAT(1H ,5X,3HM =,15,5X,3HI =,110)
2900 FORMAT(32HOSIMILARITY PARAMETERS. DELTA =,F10.4,5X,5HTAU =,F12.4,
1 5X,5HRECP =,F12.4)
STOP
END

```

The function routine RUNGE has been removed from the published form of this report to protect the copyright of the authors of Reference 5.

```

SUBROUTINE RNUMBR (RENOLD,DGFC,CD)
IF (ABS(RENOLD).LT.1.0E-12) RENOLD=1.0E-12
IF (RENOLD.LT.1.0) GO TO 26
IF ((RENOLD.GE.1.0).AND.(RENOLD.LT.1.0E3)) GO TO 27
CD=DGFC*0.4
RETURN
26 CD=DGFC*(4.5+24.0/RENOLD)
RETURN
27 ARF=ALOG(RENOLD)
CD=(28.5-24.0*ARF+9.0682*ARF**2-1.7713*ARF**3+0.1718*ARF**4
1 -0.0065*ARF**5)*DGFC
RETURN
END

```

```

SUBROUTINE VORTEX(L,VR,VU,VZ,POS,T,RHO,GAM,PGAS)
DIMENSION T(2),RHO(2)
M=0
IF(1.GE.0) GO TO 10
N=0
CVOR=1.0/(GAM-1.0)
ABC=(GAM-1.0)/(GAM+1.0)
VCR=SQRT(2.0*PGAS*GAM*T(2)/(GAM+1.0))
TOTT=1.0-ABC*((VU**2+VR**2+VZ**2)/(VCR**2))
T(1)=T(2)*TOTT
RHO(1)=RHO(2)*(TOTT**CVOR)
CU=VU*POS
CR=-VP*PLS*RHO(1)/RHO(2)
RETURN
10 IF(M.GT.100) GO TO 20
VVP=VR
VUP=VU
TOTT=1.0-ABC*((VU**2+VR**2+VZ**2)/(VCR**2))
IF(TOTT.LE.0.0) GO TO 15
VU=CU/POS
VR=-CR*((1.0/TOTT)**CVOR)/POS
M=M+1
IF((ABS(VVP-VR).GT.1.0E-6).AND.(ABS(VUP-VU).GT.1.0E-6)) GO TO 10
T(1)=T(2)*TOTT
RHO(1)=RHO(2)*(TOTT**CVOR)
AMACH=SQRT((VU**2+VR**2+VZ**2)/(GAM*PGAS*T(1)))
IF(AMACH.GT.1.0) GO TO 30
GO TO 40
15 WRITE(6,1200) POS
1200 FORMAT(1H0,41HVORTEX FAILED TO CONVERGE AT LOCATION R =,F10.4)
L=10000
RETURN
20 WRITE(6,1000) M,VR,VVP,VU,VUP,TOTT
GO TO 40
30 IF(N.EQ.0) WRITE(6,1100)
N=1
40 CONTINUE
1000 FORMAT(110,5F15.8)
1100 FORMAT(1H0,11HCHOKED FLOW)
RETURN
END

```

ORIGINAL PAGE IS
OF POOR QUALITY

Example

The example presented here uses the ft., lbm., second system of units. The vortex in this case has an inlet radius of 0.2742 ft. and an exit of 0.2467 ft. The axial span of the vortex lies between $z = 0$ and $z = 0.0264$ ft. A cold gas equivalent flow is considered and therefore, the inlet stagnation conditions are standard sea level conditions. Gravity is neglected in this example.

The specific gravity of the particle studied is about 3, its diameter is 236 microns, its initial velocity is equal to one half of the gas velocity and its initial position is slightly below the inlet radius. After 27.3° , the particle leaves the flow field at the inlet radius.

The following pages contain a computer code sheet with the data arranged in the proper columns and the output for this example.

Γ-ANGLE	T-MAX	R-MIN	R-MAX	MAX-ANGLE	Z-MAX
1.0000	0.0032	0.2467	0.2742	360.0000	0.0264
REF. VISCOSITY	REF. TEMP	SUTHERLANDS T	GAMMA	GAS CONST	DRAQ FACTOR
0.10600E-04	492.0000	198.2000	1.4000	1715.4800	1.0000
RIN	VPIN	VUIN	VZIN	TT	RCT
0.2741	-184.0000	780.0000	0.0	518.7000	0.7640E-01
ALPHA	BETA	ACC. OF GRAV.			
0.0	0.0	0.0			

CRITICAL VELOCITY = 1018.8826

ORIGINAL PAGE IS
OF POOR QUALITY

EXAMPLE CASE FOR NASA REPORT V-PART/V-GAS = 501

YR(1)	YR(2)	YR(3)	YR(4)	YR(5)	YR(6)
0.274100	-92.000000	0.0	1422.839844	0.001000	0.0
RHCP	CIA				
187.199997	0.7761E-03				
NSTFP	M				
10	0.100E-C4				

SIMILARITY PARAMETERS. DELTA = 9.5591 TAU = 0.6555E 00 RECR = 0.2004E 04

T	YR(1)	YR(2)	YR(3)	YR(4)	YR(5)	YR(6)	PENOLDS	R/RIA	STREAMLINE
C.10E-04	0.27321	-86.47282	0.81814	1432.94	0.00100	0.0	C.179E 04	C.99674	0.99664
C.20E-04	0.27237	-86.89003	1.64193	1442.55	0.00100	0.0	C.183E 04	C.99369	0.99326
C.30E-04	0.27159	-75.25425	2.47109	1451.67	0.00100	0.0	C.181E 04	C.99084	0.98986
C.40E-04	0.27087	-62.56865	3.30532	1460.25	0.00100	0.0	C.181E 04	C.98820	0.98648
C.50E-04	0.27021	-42.82595	4.14431	1468.29	0.00100	0.0	C.182E 04	C.98577	0.98308
C.60E-04	0.26964	-22.24242	5.03529	1482.65	0.00100	0.0	C.184E 04	C.98153	0.97626
C.80E-04	0.26854	-46.58834	6.6E661	1488.94	0.00100	0.0	C.185E 04	0.97973	0.97285
C.90E-04	0.26811	-40.50073	7.54136	1494.62	0.00100	0.0	C.186E 04	C.97815	0.96543
C.10E-03	0.26773	-24.58320	8.25519	1499.67	0.00100	0.0	C.187E 04	C.97678	0.96601
C.15E-03	0.26742	-28.63975	9.25973	1504.37	0.00100	0.0	C.188E 04	C.97562	0.96259
C.20E-03	0.26716	-22.67404	10.12260	1507.83	0.00100	0.0	C.189E 04	C.97469	0.95918
C.30E-03	0.26683	-10.65171	11.85396	1513.35	0.00100	0.0	C.191E 04	0.97347	0.95237
C.40E-03	0.26675	-4.68316	12.72148	1515.10	0.00100	0.0	C.192E 04	C.97319	0.94897
C.50E-03	0.26673	1.32156	13.58991	1516.17	0.00100	0.0	C.194E 04	0.97313	0.94558
C.60E-03	0.26678	7.34824	14.45875	1516.57	0.00100	0.0	C.195E 04	0.97328	0.94221
C.80E-03	0.26689	13.26304	15.32762	1516.29	0.00100	0.0	C.196E 04	C.97360	0.93884
C.10E-02	0.26704	19.27157	16.19615	1515.32	0.00100	0.0	C.197E 04	C.97426	0.93549
C.15E-02	0.26727	25.36983	17.06392	1513.68	0.00100	0.0	C.198E 04	C.97508	0.93216
C.20E-02	0.26790	37.41931	18.75570	1508.40	0.00100	0.0	0.200E 04	C.97736	0.92553
C.30E-02	0.26833	47.26256	19.65894	1504.78	0.00100	0.0	C.201E 04	C.97823	0.92225
C.40E-02	0.26876	49.11960	20.51993	1500.51	0.00100	0.0	C.202E 04	C.98052	0.91899
C.50E-02	0.26928	55.06662	21.37928	1495.61	0.00100	0.0	C.203E 04	0.98242	0.91574
C.60E-02	0.26986	60.91997	22.23366	1490.10	0.00100	0.0	C.204E 04	C.98454	0.91252
C.80E-02	0.27050	66.73589	23.08569	1483.98	0.00100	0.0	0.205E 04	C.98687	0.90933
C.10E-01	0.27195	78.24197	24.77841	1470.03	0.00100	0.0	C.207E 04	C.99216	0.90301
C.20E-01	0.27274	83.92549	25.61844	1462.18	0.00100	0.0	0.208E 04	C.99511	0.89990
C.30E-01	0.27363	89.55849	26.45383	1453.82	0.00100	0.0	C.209E 04	0.99828	0.89661
C.40E-01	0.2746	95.1780	27.2843	1444.9573	0.0010	0.0	2056.6499	1.0016	0.8937

42

M = 1 I = 32

SCRL2D - Two-dimensional Scroll

This program integrates the equations of motion to determine the trajectory of a particle in a scroll. The force of gravity acting on the particle can be included in the cases where this is necessary. The solution is two dimensional, but the output has been written to include a possible extension in the future to a three-dimensional solution. The particle is allowed to bounce off the scroll wall, and the solution stops when the particle enters the nozzle region of the flow. The cylindrical coordinate system r , θ and z is used as indicated in Figure 16.

Method

Figure 19 is a flow diagram of this program. Unlike the previous cases, there is no need to iterate for the average gas flow properties at the location of the particle. The gas flow solution is assumed to be uniform throughout the scroll and, therefore, the properties of the gas do not change between inlet and exit. The main program uses three subroutines, RUNGE, RNUMBER, and BOUNCE to trace the particle trajectory in the same manner as discussed previously. The subroutine BOUNCE that is used with this program is the unmodified routine that has been improved considerably. The improved subroutine is the one that has been described previously.

Input

There are two sets of input. The first set specifies the nature of the gas flow and consists of two cards at the front of the data. This program is written so that any consistent system of units may be used. An example of this input data is included with the example case presented after the program listing.

The first two data cards take the following form:

GAM, RGAS, WTFL, RHOIP, TIP	(5F10.6)
D(1), A(1), REXIT, SRDEND .	(4F10.6)

These variables will be defined later.

The second set of data cards specify the type of particle and its initial position and velocity. Each particle can be described by six input data cards which take the following form:

TITLE	(18A4)
VISREF, TREF, TSUT, DGFC	(E20.5, 3F10.3)
RHOP, DIAP, H, TMAX, ETA(1), ETA(2)	(F10.2, 3E10.4, 2F10.3)
YR(I), I = 1,6	(6F10.3)
ALPHA, BETA, GRAVITY	(3F10.4)
NSTEP	(I5)

These variables are defined below. For studies that are done with multiple particles, additional sets of input data cards as indicated for the second data card set may be stacked together. When the program completes the trajectory for one particle, it goes to the next set automatically.

Variables

GAM - Ratio of specific heats

RGAS - Gas constant (Length²/Time² Degree Abs.)

WTFL - Mass flow rate of gas (Mass/Time)

RHOIP - Gas inlet stagnation density (Mass/Length³)

TIP - Gas inlet stagnation temperature (Degree Abs.)

D(1) - Dimension of the first station as indicated in Figure 20. (Length)

A(1) - Area of the first station. (Length²)

REXIT - Radial distance that determines exit location. See Figure 20. (Length)

SRDEND - Angular location of the last portion of the regular scroll contour before suppression of the scroll begins. See Figure 20. (Degrees)

- TITLE - The first card is reproduced at the top of the first page of output for each particle. Any statement in columns 2 to 72 will be reproduced.
- VISREF - Reference viscosity corresponding to TREF. Used in Sutherland's Law. (Mass/Length x Time)
- TREF - Reference temperature corresponding to VISREF. Used in Sutherland's Law. (Degrees Abs.)
- TSUT - Constant used in Sutherland's Law. (198.6°R or 110°K)
- DGFC - Drag Factor - The spherical drag coefficient based on Reynolds Number is multiplied by DGFC. Except in very special cases, this should be 1.0.
- RHOP - Particle density. (Mass/Length³)
- DIAP - Particle diameter. (Length)
- H - Integration time increment. (Time). If extremely long computer run times are experienced, this can be made larger. If the program fails to converge, this can be made smaller.
- TMAX - The program stops the solution if time exceeds TMAX. (Time)
- ETA(1) - Normal restitution coefficient.
- ETA(2) - Tangential restitution coefficient.
- YR(1) - Particle's initial radial position. (Length)
- YR(2) - Particle's initial radial velocity. Positive in the outward direction, negative in the inward direction. (Length/Time)
- YR(3) - Particle's initial angular position. (Degrees)
- YR(4) - Particle's initial angular velocity. (Time)
- YR(5) - Particle's initial axial position. (Length)
- YR(6) - Particle's initial axial velocity. (Length/Time)

- Alpha - Angle of z-axis with respect to the horizontal.
See Figure 18. (Degrees)
- Beta - Angle of the turbine vertical axis with respect
to the gravitational vertical. See Figure 18.
(Degrees)
- GRAVITY - Acceleration of gravity. (Length/Time²)
- NSTEP - Integer that determines the amount of printed
output. Output data is printed every NSTEP
time increments.

After initializing the input variables, the program calculates the velocity components and properties of the gas. The gas flow in the scroll is isotropic, with both the mass flow rate and the scroll cross sectional area decreasing uniformly by the same ratio. The velocity is based on the inlet area, but is deflected inward by the angle CH1, where

$$CH1 = \text{arc tan} \left(\frac{D(1)}{2\pi REXIT} \right)$$

Output

The first part of the output is an echo check of the first set of data cards that describe the gas flow. Such data checks are useful in correcting key punch mistakes on the input cards.

The output variables that are not defined in the input are:

- V - Gas velocity (Length/Time)
- VR - Radial Component of gas velocity (Length/Time)
- VU - Tangential component of gas velocity (Length/Time)
- CH1 - Angle between V and VU. (Degrees)
- VCR - Critical Velocity. (Length/Time)
- RHO - Gas density (Mass/Length³)
- TEMP - Gas temperature (Degrees Abs.)

- ORTHOGONAL - Number associated with various radial stations along the scroll.
- THETA(I,1) - Angular position of inner contour. (Degrees)
- THETA(I,2) - Angular position of outer contour. (Degrees)
- R(I,1) - Radial position of inner contour. (Length)
- R(I,2) - Radial position of outer contour. (Length).

The program next skips to a new page and the first part of the printed output is an echo check of the input data as punched on the data cards. After initializing the variables, the program calculates and prints several similarity parameters that are useful in relating this particle to other particles that will have similar trajectories. Next the program writes the trajectory information every NSTEP time increments.

The additional terms of the output that are not defined as part of the input are listed below.

- DELTA - The characteristic length as given in Equation (7). (Length)
- TAU - The time constant as given in Equation (8). (Time)
- RECR - The Reynolds number as given in Equation (9).
- L - A time increment counter.
- T - Time. (Time)
- RENOLD - The particle Reynolds number at this point.

Program Listing.

PARTICLE TRAJECTORY SOLUTION IN A SCROLL

```

DIMENSION STATE(18),YR(6),YRS(6),R(37,2),FR(6),X(2),Y(2),ETA(2)
DIMENSION A(37),D(37),THETA(37,2)
DIMENSION AA(3),B(3),C(3),P(3),PP(3),VP(3),PB(3)
INTEGER RUNGE
READ(5,1010) GAM,RGAS,WTFLL,RHOIP,TIP
WRITE(6,2010)GAM,RGAS,WTFLL,RHOIP,TIP
READ(5,1020) J(1),A(1),REXIT,SRDEND
WRITE(6,2020)D(1),A(1),REXIT,SRDEND

```

INITIALIZE AND BUILD ARRAYS

```

DA=A(1)/36.0
DO 20 I=2,37
A(I)=A(I-1)-DA
IF(A(I).GT.1.0E-12) D(I)=D(I-1)*SQRT(A(I)/A(I-1))
IF(A(I).LE.1.0E-12) D(I)=1.0E-12
R(I,1)=REXIT
R(I,2)=REXIT+D(I)
THETA(I,1)=FLOAT(I-1)*10.
20 THETA(I,2)=THETA(I,1)
START=SRDEND/10.+2.
NSTART=START
DO 25 I=NSTART,37
ANG=360.0-THETA(I,2)
25 R(I,2)=REXIT*SQRT(1.0+TAN(ANG/57.29577)**2)
R(1,1)=REXIT
R(1,2)=REXIT+D(1)
THETA(1,1)=0.0
THETA(1,2)=0.0
EXPON=1.0/(GAM-1.0)
VCR=SQRT(2.0*GAM*RGAS*TIP/(GAM+1.0))
CIRCUM=2.0*3.1415927*REXIT
CHI=ATAN2(D(1),CIRCUM)

```

DETERMINE GAS VELOCITY

```

M=1
VU=WTFLL/RHOIP/A(1)
VR=-VU*TAN(CHI)
VEST=SQRT(VR**2+VU**2)/VCR
30 M=M+1
RHO=RHOIP*(1.0-(GAM-1.0)/(GAM+1.0)*VEST**2)**EXPON
WTFLES=RHO*A(1)*VU
IF(ABS(WTFLES-WTFLL).LT.1.0E-4) GO TO 40
IF(M.GT.90) WRITE(6,2060) VEST,RHO,WTFLES,CHI,VU,VR,VCR
IF(M.GT.101) GO TO 810
VU=WTFLL/RHO/A(1)
VR=-VU*TAN(CHI)
VEST=SQRT(VR**2+VU**2)/VCR
GO TO 30

```

```

40 V=VEST*VCR
   TEMP=TIP*(1.0-(GAM-1.0)/(GAM+1.0)*VEST**2)
   VZ=0.0

```

```

PRINT OUT SOLUTION.

```

```

   CHIW=CHI*57.29578
   WRITE(6,2030) V,VR,VU,CHIW,VCR,RHO,TEMP
   DO 50 I=1,37
50  WRITE(6,2040) I,THETA(I,1),THETA(I,2),R(I,1),R(I,2)
   DO 60 K=1,2
   DO 60 I=1,37
60  THETA(I,K)=THETA(I,K)/57.29578
   SRDEND=SRDEND/57.29577
   VRSAVE= VR
   VUSAVE=VU

```

```

START PARTICLE SOLUTION

```

```

105 READ(5,3000) (STATE(I),I=1,18)
   WRITE(6,4000)(STATE(I),I=1,18)
   READ(5,3010) VISREF,TREF,TSUT,DGFC
   WRITE(6,4010) VISREF,TREF,TSUT,DGFC
   READ(5,3020) RHJP,DIAP,H,TMAX,ETA(1),ETA(2)
   WRITE(6,4020) RHOP,DIAP,H,TMAX,ETA(1),ETA(2)
   READ(5,3030) (YR(I),I=1,6)
   WRITE(6,4030)(YR(I),I=1,6)
   YR(3)=YR(3)/57.29577
   READ(5,1000) ALPHA,BETA,GRAVTY
   WRITE(6,3090) ALPHA,BETA,GRAVTY
   READ(5,3080) NSTEP
   WRITE(6,4080) NSTEP
   RHOCR=RHOIP*(2.0/(GAM +1.0))**(1.0/(GAM -1.0))
   DELTA=RHOP*DIAP/RHOCR/0.3
   TCR=(2.0/(GAM +1.0))*TIP
   VISCR=VISREF*((TCR/TREF)**1.5)*(TREF+TSUT)/(TCR+TSUT)
   TAU=RHJP*DIAP**2/18.0/VISCR
   RECR=DIAP*RHOCR*VCR/VISCR/2.0
   WRITE(6,5020) DELTA,TAU,RECR
   WRITE(6,4090)

```

```

CHECK THAT PARTICLE STARTS INBOUNDS.

```

```

   NOUT=0
   RW=REXIT+D(1)*SQRT(1.0-YR(3)/6.2831854)
   IF(YR(3).LT.0.0) NOUT=1
   IF(YR(1).GE.RW) NOUT=7
   IF(YR(1).LE.REXIT) NOUT=8
   IF(NOUT.EQ.0) GO TO 130
   WRITE(6,4040) NOUT
   GO TO 105
130 CONTINUE

```

INITIALIZE FOR FIRST STEP.

ALPHA=ALPHA/57.29577

BETA=BETA/57.29577

RPART=DIAP/2.

N=6

L=0

NTIME=NSTEP

T=0.0

YR3=YR(3)*57.29577

WRITE(6,5000) L,T,YR(1),YR(2),YR3,YR(4),YR(5),YR(6)

TS=T

DO 620 I=1,N

620 YRS(I)=YR(I)

RWS=RW

VISTAR=VISREF*((TEMP/TREF)**1.5)*(TREF+TSUT)/(TEMP+TSUT)

INTEGRATE USING RUNGE-KUTTA METHOD.

625 IF(YR(3).LT.SRDEND) GO TO 626

VR=V*SIN(YR(3))

VU=V*COS(YR(3))

GO TO 627

626 CONTINUE

VR=VRSAVE

VU=VUSAVE

627 CONTINUE

VDIFF=SQRT((VR-YR(2))**2+(VU-YR(1)*YR(4))**2+(VZ-YR(6))**2)

RENOLD=RHO*VDIFF*DIAP/VISTAR

CALL RNUMBR(RENOLD,DGFC,CD)

BCON=RHO*CD/RHOP/RPART/2.33333

630 IF(RUNGE(N,YR,FR,T,H).NE.1) GO TO 640

FR(1)=YR(2)

FR(2)=YR(1)*YR(4)**2+BCON*VDIFF*(VR-YR(2))-GRAVY*COS(YR(3)+BETA)

FR(3)=YR(4)

FR(4)=-2.0*YR(2)*YR(4)/YR(1)+BCON*VDIFF*(VU-YR(1)*YR(4))/YR(1)

1 +GRAVY*SIN(YR(3)+BETA)/YR(1)

FR(5)=YR(6)

FR(6)=BCON*VDIFF*(VZ-YR(6))-GRAVY*SIN(ALPHA)

GO TO 630

640 CONTINUE

DETERMINE IF WALL INTERACTION OCCURRED.

NOUT=0

IF(YR(3).LT.0.0) NOUT=1

IF(YR(3).GT.6.2831854) NOUT=3

IF(YR(1).LE.REXIT) NOUT=8

IF((NOUT.EQ.1).OR.(NOUT.EQ.3).OR.(NOUT.EQ.8)) GO TO 780

RW=REXIT+D(1)*SQRT(1.0-YR(3)/6.2831854)

IF(YR(1).GE.RW) NOUT=7

IF(YR(3).GT.SRDEND) GO TO 650

IF(NOUT.EQ.0) GO TO 700

```

NTEST=0
IF(YR(3).GE.1.57) NTEST=1
AA(1)=RWS*COS(YRS(3))
AA(2)=RWS*SIN(YRS(3))
AA(3)=YRS(5)
B(1)=RW*COS(YR(3))
B(2)=RW*SIN(YR(3))
B(3)=YR(5)
GO TO 660
650 XTEST=YR(1)*CJS(YR(3))
IF(XTEST.LT.REXIT) GO TO 700
NJUT=9
NTEST=1
AA(1)=REXIT
ANG=6.2831854-YRS(3)
AA(2)=REXIT*TAN(ANG)
AA(3)=YRS(5)
B(1)=REXIT
ANG=6.2831854-YR(3)
B(2)=REXIT*TAN(ANG)
B(3)=YR(5)
660 CONTINUE
C(1)=AA(1)
C(2)=AA(2)
C(3)=YR(5)+ SQRT((AA(1)-B(1))**2+(AA(2)-B(2))**2+(AA(3)-B(3))**2)
P(1)=YRS(1)*CJS(YRS(3))
P(2)=YRS(1)*SIN(YRS(3))
P(3)=YRS(5)
PP(1)=YR(1)*COS(YR(3))
PP(2)=YR(1)*SIN(YR(3))
PP(3)=YR(5)
T=TS
CALL BOUNCE(AA,B,C,P,PP,H,T,ETA,NFIX,PB,VP)
TB=ATAN2(PB(2),PB(1))*57.29577
RB= SQRT(PB(1)**2+PB(2)**2)
IF((NTEST.EQ.1).AND.(TB.LE.0.0)) TB=TB+360.0
ZB=PB(3)
WRITE(6,5030) NOUT, RB, TB, ZB
YR(1)= SQRT(PB(1)**2+PB(2)**2)
YR(3)= ATAN2(PB(2),PB(1))
IF((NTEST.EQ.1).AND.(YR(3).LT.0.0)) YR(3)=YR(3)+6.283184
YR(5)=PB(3)
YR(2)=VP(1)*CJS(YR(3))+VP(2)*SIN(YR(3))
YR(4)=-VP(1)*SIN(YR(3))/YR(1)+VP(2)*COS(YR(3))/YR(1)
YR(6)=VP(3)
700 L=L+1
TS=T
RWS=RW
DO 760 I=1,N
760 YRS(I)=YR(I)
VTIME=NTIME-1
780 IF((NOUT.EQ.1).OR.(NOUT.EQ.3).OR.(NJUT.EQ.8).OR.(T.GT.TMAX))
1 GO TO 800

```

```

IF ((ABS(YR(2)).LT.1.0E-4).AND.(ABS(YR(4)).LT.1.0E-4).AND.
1 (ABS(YR(6)).LT.1.0E-4)) GO TO 800
IF (N TIME.GT.0) GO TO 625
N TIME=NSTEP-1
IF (L.GT.1) N TIME=NSTEP
YR3=YR(3)*57.29578
WRITE(6,4050) L,T,YR(1),YR(2),YR3,YR(4),YR(5),YR(6),RENOLD
GO TO 625
800 CONTINUE
YR3=YR(3)*57.29577
WRITE(6,4060) L,T,YR(1),YR(2),YR3,YR(4),YR(5),YR(6),RENOLD
805 CONTINUE
GO TO 105
810 WRITE(6,2050)

```

FORMAT STATEMENTS

```

1000 FFORMAT(7F10.4)
1010 FFORMAT(5F10.6)
1020 FFORMAT(4F10.6)
2010 FFORMAT(1H1,5X,21HINLET SCROLL SOLUTION,/,12X,3HGAM,11X,4HRGAS,
1 11X,4HWTFL,10X,5HRHOIP,12X,3HTIP,/, (5F15.6))
2020 FFORMAT(1H0,10X,4HD(1),11X,4HA(1),10X,5HPEXIT,9X,6HSRDEND,
1 /,(4F15.6))
2030 FFORMAT(1H0,13X,1HV,13X,2HVR,13X,2HVU,12X,3HCHI,12X,3HVCR,13X,
1 3HRHO,11X,4TFMP,/, (7F15.6),/,11H ORTHOGONAL,5X,10HTHETA(I,1),
2 5X, 10HTHETA(I,2),9X,6HR(I,1),9X,6HR(I,2))
2040 FFORMAT(111,2F15.4,2F15.6)
2050 FFORMAT(37HOGAS FLOW SOLUTION FAILED TO CONVERGE)
2060 FFORMAT(1H0,7E15.8)
3000 FFORMAT(18A4)
3010 FFORMAT(E20.5,3F10.3)
3020 FFORMAT(F10.2,3F10.4,2F10.3)
3030 FFORMAT(6F10.3)
3080 FFORMAT(I5)
3090 FFORMAT(1H0,9X,5HALPHA,11X,4HBETA,8X,7HGRAVITY,/, (3F15.4))
4000 FFORMAT(1H1,18A4)
4010 FFORMAT(1H0,13X,6HVISREF,11X,4HTRFF,11X,4HTSUT,11X,4HDGFC,/,
1 (E20.5,3F15.3))
4020 FFORMAT(1H0,10X,4HRHOP,11X,4HDIAP,14X,1H4,11X,4HTMAX,10X,5HETA-N,
1 10X,5HETA-T,/, (F15.2,3E15.4,2F15.3))
4030 FFORMAT(1H0, 9X,5HYR(1),10X,5HYR(2),10X,5HYR(3),10X, 5HYR(4),
1 10X,5HYR(5),10X,5HYR(6),/, (6F15.6))
4040 FFORMAT(1H0,62H PARTICLE NOT IN PASSAGE AT FIRST POINT GIVEN, GO TO
1 NEXT CASE)
4050 FFORMAT(1H ,15,E10.2,7F15.5)
4060 FFORMAT(1H0,15,E10.2,7F15.5)
4080 FFORMAT(17HOPRINT DATA EVERY,17,2X,7HSTEP(S))
4090 FFORMAT(1H0,3X, 1HL, 9X,1HT,10X,5HYR(1),10X,5HYR(2),10X,5HYR(3),
1 10X,5HYR(4),10X,5HYR(5),10X,5HYR(6),9X,6HRENOLD)
5000 FFORMAT(1H ,15,E10.2,6F15.5)
5020 FFORMAT(32HOSIMILARITY PARAMETERS. DELTA =,F10.4,5X,5HTAU =,F12.4,
1 5X,6HRECR =,F12.4)
5030 FFORMAT(11H BOUNCE OFF,15,F15.5,15X,F15.5,15X,F15.5)
STOP
END

```


The function routine RUNGE has been removed from the published form of this report to protect the copyright of the authors of Reference 5.

```
SUBROUTINE RNJMBR(RENOLD,DGFC,CD)
IF( ABS(RENOLD).LT.1.0E-12) RENOLD=1.0E-12
IF(RENOLD.LT.1.0) GO TO 26
IF((RENOLD.GE.1.0).AND.(RENOLD.LT.1.0E3)) GO TO 27
CD=DGFC*0.4
RETURN
26 CD=DGFC*(4.5+24.0/RFNOLD)
RETURN
27 ARE=ALOG(RENOLD)
CD=(28.5-24.0*ARE+9.0682*ARE**2-1.7713*ARE**3+0.1718*ARE**4
1 -0.0065*ARE**5)*DGFC
RETURN
END
```

```

SUBROUTINE BOUNCE(A,B,C,P,PP,H,T,ETA,NFIX,PB,VP)
DIMENSION A(3),B(3),C(3),P(3),PP(3),V(3),AB(3),AC(3),G(3,3),D(5),
1 PB(3)
DIMENSION GS(3,3),VP(3),UN(3),PN(3)
DIMENSION ETA(2)

NFIX=1
DO 10 I=1,3
V(I)=(PP(I)-P(I))/H
AB(I)=B(I)-A(I)
10 AC(I)=C(I)-A(I)

DETERMINE UNIT NORMAL TO SURFACE

UN(1)=AB(2)*AC(3)-AB(3)*AC(2)
UN(2)=AB(3)*AC(1)-AB(1)*AC(3)
UN(3)=AB(1)*AC(2)-AB(2)*AC(1)
CMAG= SQRT(UN(1)**2+UN(2)**2+UN(3)**2)
IF( ABS(CMAG).GT.1.0E-12) GO TO 20
NFIX=0
RETURN
20 DO 30 I=1,3
30 UN(I)=UN(I)/CMAG

DETERMINE THE INTERSECTION POINT, PB, OF THE PLANE AND TRAJECTORY.

DETA=JN(1)*V(1)**2+UN(2)*V(1)*V(2)+UN(3)*V(1)*V(3)
D(1)=UN(1)*A(1)+UN(2)*A(2)+UN(3)*A(3)
D(2)=V(2)*P(1)-V(1)*P(2)
D(3)=V(3)*P(1)-V(1)*P(3)
DO 40 J=1,3
40 G(I,J)=UN(J)
G(2,1)= V(2)
G(2,2)=-V(1)
G(2,3)=0.0
G(3,1)= V(3)
G(3,2)=0.0
G(3,3)=-V(1)

IF DETERMINANT EQUALS ZERO, GO TO 80

IF( ABS(DETA).LE.1.0E-12) GO TO 80
DO 70 K=1,3
DO 50 I=1,3
DO 50 J=1,3
50 GS(I,J)=G(I,J)
DO 60 I=1,3
60 GS(I,K)= D(I)
PB(K)=GS(1,1)*GS(2,2)*GS(3,3)+GS(1,2)*GS(2,3)*GS(3,1)+GS(1,3)
1 *GS(2,1)*GS(3,2)-GS(3,1)*GS(2,2)*GS(1,3)-GS(3,2)*GS(2,3)*GS(1,1)
2 -GS(3,3)*GS(2,1)*GS(1,2)
70 PB(K)=PB(K)/DETA
GO TO 100

```

RESET MATRIX AND SOLVE FOR PR. IF DET. EQUALS ZERO, GO TO 90

```

80 PB(1)=P(1)
   DETB=-UN(2)*V(2)-UN(3)*V(3)
   IF( ABS(DETB).LE.1.0E-12) GO TO 90
   D(4)=UN(2)*P(2)+UN(3)*P(3)
   D(5)=V(3)*P(2)-V(2)*P(3)
   PB(2)=(-D(4)*V(2)-D(5)*UN(3))/DETB
   PB(3)=(UN(2)*D(5)-V(3)*D(4))/DETB
   GO TO 100

```

SPECIAL CASE THAT YIELDS ZERO DETERMINANT ALWAYS.

```

90 PB(2)=P(2)
   PB(3)=P(3)
100 CONTINUE

```

DETERMINE THE INTERSECTION POINT, PN, OF THE SURFACE NORMAL THRU P
IF DETERMINANT EQUALS ZERO, GO TO 140

```

   DETA=JN(1)**3+JN(1)*UN(2)**2+UN(1)*UN(3)**2
   IF( ABS(DETA).LE.1.0E-12) GO TO 140
   D(2)=P(1)*UN(2)-P(2)*UN(1)
   D(3)=P(1)*UN(3)-P(3)*UN(1)
   G(2,1)= UN(2)
   G(2,2)=-JN(1)
   G(2,3)=0.0
   G(3,1)= UN(3)
   G(3,2)=0.0
   G(3,3)=-UN(1)
   DO 130 K=1,3
   DO 110 I=1,3
   DO 110 J=1,3
110 GS(I,J)=G(I,J)
   DO 120 I=1,3
120 GS(I,K)=D(I)
   PN(K)=GS(1,1)*GS(2,2)*GS(3,3)+GS(1,2)*GS(2,3)*GS(3,1)+GS(1,3)
   1 *GS(2,1)*GS(3,2)-GS(3,1)*GS(2,2)*GS(1,3)-GS(3,2)*GS(2,3)*GS(1,1)
   2 -GS(3,3)*GS(2,1)*GS(1,2)
130 PN(K)=PN(K)/DETA
   GO TO 160
140 IF(( ABS(UN(1)).GT.1.0E-12).AND.( ABS(UN(2)).GT.1.0E-12))GO TO 150
   PN(1)=P(1)
   PN(2)=P(2)
   PN(3)=P(3)
   PV(3)=A(3)
   GO TO 160
150 PN(1)=A(1)
   PN(2)=P(2)
   PV(3)=P(3)
160 CONTINUE

```

C DETERMINE PORTION OF TIME SEGMENT USED TO TRAVEL FROM P TO PB.
C

DPR = SQRT((PB(1)-P(1))**2+(PB(2)-P(2))**2+(PB(3)-P(3))**2)
VPP = SQRT(V(1)**2+V(2)**2+V(3)**2)
DTIME = DPR/VPP

C EXTENT LINE PN-PB THE PROPER DISTANCE TO FIND PNP.
C THEN EXTENT A LINE NORMAL TO THE SURFACE FROM PNP TO GET THE POINT
C AFTER BOUNCE, PP.
C FIND VELOCITY COORDINATES BASED ON PP, PB AND TIME REMAINING IN
C SEGMENT.
C

DO 170 I=1,3
S=(PB(I)-PN(I))/DTIME
PNP=PB(I)+ETA(2)*S*(H-DTIME)
SN=(PN(I)-P(I))/DTIME
PP(I)=PNP-ETA(1)*SN*(H-DTIME)
170 VP(I)=(PP(I)-PB(I))/(H-DTIME)
T=T+H
RETURN
END

Example

The example case presented here uses the ft., lbm., second system of units. The gas flow conditions correspond to inlet stagnation conditions of standard sea level air. The scroll dimension $D(1)$ is 0.4 ft. and the exit radius is 0.3615 ft. The particle inlet velocity is in the same direction as the velocity, its magnitude is approximately one half the gas velocity. The particle has a specific gravity of 3 and a diameter of 12 microns. The normal and tangential restitution coefficients are assumed to be 1.0. Gravity is included.

The following pages contain a computer code sheet with the data arranged in the proper columns, and the output for this example.

INLET SCROLL SOLUTION

GAM	RGAS	WTFL	RHOIP	TIP		
1.400000	1715.479980	0.486000	0.076400	518.699951		
D(1)	A(1)	REXIT	SRDEND			
0.400000	0.080000	0.361500	316.174805			
V	VR	VU	CHI	VCR	RHO	TEMP
80.950897	-14.039830	79.724136	9.987672	1018.882568	0.076199	518.154053

ORTHOGONAL	THETA(I,1)	THETA(I,2)	R(I,1)	R(I,2)
1	0.0	0.0	0.361500	0.761500
2	10.0000	10.0000	0.361500	0.755905
3	20.0000	20.0000	0.361500	0.750230
4	30.0000	30.0000	0.361500	0.744471
5	40.0000	40.0000	0.361500	0.738623
6	50.0000	50.0000	0.361500	0.732684
7	60.0000	60.0000	0.361500	0.726648
8	70.0000	70.0000	0.361500	0.720510
9	80.0000	80.0000	0.361500	0.714266
10	90.0000	90.0000	0.361500	0.707910
11	100.0000	100.0000	0.361500	0.701434
12	110.0000	110.0000	0.361500	0.694833
13	120.0000	120.0000	0.361500	0.688098
14	130.0000	130.0000	0.361500	0.681221
15	140.0000	140.0000	0.361500	0.674194
16	150.0000	150.0000	0.361500	0.667004
17	160.0000	160.0000	0.361500	0.659641
18	170.0000	170.0000	0.361500	0.652092
19	180.0000	180.0000	0.361500	0.644342
20	190.0000	190.0000	0.361500	0.636373
21	200.0000	200.0000	0.361500	0.628166
22	210.0000	210.0000	0.361500	0.619698
23	220.0000	220.0000	0.361500	0.610943
24	230.0000	230.0000	0.361500	0.601869
25	240.0000	240.0000	0.361500	0.592439
26	250.0000	250.0000	0.361500	0.582607
27	260.0000	260.0000	0.361500	0.572317
28	270.0000	270.0000	0.361500	0.561498
29	280.0000	280.0000	0.361500	0.550060
30	290.0000	290.0000	0.361500	0.537882
31	300.0000	300.0000	0.361500	0.524797
32	310.0000	310.0000	0.361500	0.510569
33	320.0000	320.0000	0.361500	0.471905
34	330.0000	330.0000	0.361500	0.417424
35	340.0000	340.0000	0.361500	0.384700
36	350.0000	350.0000	0.361500	0.367077
37	360.0000	360.0000	0.361500	0.361500

EXAMPLE CASE FOR NASA REPORT V-PART/V-GASS = 50%

VISREF		TREF	TSUT	DGFC		
0.10600F-04		492.000	198.200	1.000		
RHOP	DIAP	H	TMAX	ETA-N	ETA-T	
187.20	0.3880F-04	0.1000E-04	0.1000E-05	1.000	1.000	
YR(1)	YR(2)	YR(3)	YR(4)	YR(5)	YR(6)	
0.561500	0.0	0.000100	51.500000	0.0	0.0	
ALPHA	BETA	GRAVITY				
0.0	0.0	32.1740				

PRINT DATA EVERY 100 STEP(S)

SIMILARITY PARAMETERS. DELTA = 0.4999 TAU = 0.1638E-02 RECR = 0.1002E 03

I	T	YR(1)	YR(2)	YR(3)	YR(4)	YR(5)	YR(6)	RENOLD
0	0.0	0.56150	0.0	0.00010	51.50000	0.0	0.0	
100	0.10F-02	0.55653	-6.90235	5.35006	118.81465	0.0	0.0	4.15392
200	0.20F-02	0.54999	-5.85931	12.79208	137.68571	0.0	0.0	2.44109
300	0.30F-02	0.54486	-4.50850	20.91054	144.63770	0.0	0.0	2.56241
400	0.40F-02	0.54076	-3.76134	29.29121	147.53693	0.0	0.0	2.75175
500	0.50F-02	0.53720	-3.40780	37.79132	149.05623	0.0	0.0	2.84865
600	0.60F-02	0.53388	-3.23498	46.36333	150.13127	0.0	0.0	2.89599
700	0.70F-02	0.53070	-3.13445	54.99220	151.06725	0.0	0.0	2.92322
800	0.80F-02	0.52760	-3.06069	63.67128	151.95792	0.0	0.0	2.94302
900	0.90F-02	0.52457	-2.99645	72.40021	152.83026	0.0	0.0	2.96019
1000	0.10F-01	0.52160	-2.93566	81.17876	153.69142	0.0	0.0	2.97642
1100	0.11F-01	0.51869	-2.87635	90.00636	154.54301	0.0	0.0	2.99224
1200	0.12F-01	0.51584	-2.81796	98.88245	155.38512	0.0	0.0	3.00781
1300	0.13F-01	0.51305	-2.76041	107.80649	156.21758	0.0	0.0	3.02315
1400	0.14F-01	0.51032	-2.70373	116.77803	157.04021	0.0	0.0	3.03825
1500	0.15F-01	0.50764	-2.64802	125.79633	157.85300	0.0	0.0	3.05308
1600	0.16F-01	0.50501	-2.59336	134.86092	158.65587	0.0	0.0	3.06764
1700	0.17F-01	0.50244	-2.53983	143.97130	159.44893	0.0	0.0	3.08188
1800	0.18F-01	0.49993	-2.48751	153.12682	160.23227	0.0	0.0	3.09581
1900	0.19F-01	0.49746	-2.43643	162.32692	161.00612	0.0	0.0	3.10940
2000	0.20F-01	0.49505	-2.38664	171.57117	161.77072	0.0	0.0	3.12265
2100	0.21F-01	0.49269	-2.33815	180.85886	162.52635	0.0	0.0	3.13555
2200	0.22F-01	0.49037	-2.29094	190.18961	163.27335	0.0	0.0	3.14811
2300	0.23F-01	0.48810	-2.24500	199.56297	164.01205	0.0	0.0	3.16034
2400	0.24F-01	0.48587	-2.20028	208.97836	164.74275	0.0	0.0	3.17224
2500	0.25F-01	0.48369	-2.15672	218.43538	165.46588	0.0	0.0	3.18384
2600	0.26F-01	0.48155	-2.11422	227.93378	166.18169	0.0	0.0	3.19516
2700	0.27F-01	0.47946	-2.07271	237.47281	166.89049	0.0	0.0	3.20622
2800	0.28F-01	0.47740	-2.03206	247.05246	167.59248	0.0	0.0	3.21706
2900	0.29F-01	0.47539	-1.99217	256.67188	168.28787	0.0	0.0	3.22770
3000	0.30F-01	0.47341	-1.95290	266.33105	168.97672	0.0	0.0	3.23818
3100	0.31F-01	0.47148	-1.91413	276.02954	169.65903	0.0	0.0	3.24853
3200	0.32F-01	0.46958	-1.87573	285.76709	170.33478	0.0	0.0	3.25878
3300	0.33F-01	0.46772	-1.83758	295.54297	171.00374	0.0	0.0	3.26898
3400	0.34F-01	0.46590	-1.79959	305.35693	171.66573	0.0	0.0	3.27913
3500	0.35F-01	0.46412	-1.76163	315.20874	172.32031	0.0	0.0	3.28929
BOUNCE OFF	9	0.44698		323.97534		0.0		
3600	0.36F-01	0.44470	-56.60704	324.23364	114.17044	0.0	0.0	4.76184
3700	0.37F-01	0.39586	-41.26906	332.46655	169.90005	0.0	0.0	1.60757
3800	0.38F-01	0.36255	-24.94926	343.41602	210.00803	0.0	0.0	0.63308
3804	0.38F-01	0.36132	-24.06871	344.01978	211.60571	0.0	0.0	0.60650

STATOR - Particles in the Stators

This program integrates the equations of motion of a particle in order to determine the trajectory of the particle in a stator. The solution neglects the gravity force, is three dimensional and the particle may bounce off any of the four surfaces that surround the channel. The program is restricted by a constant spacing from hub to shroud. The coordinate system is the r, θ, z system which is indicated in Figure 16.

Method

Figure 21 is a flow diagram for this program. It illustrates the iterative technique used to find the average gas properties along the particle trajectory, which was explained previously. As soon as the program determines that a collision has occurred, the program bounces the particle off the surface and then continues the trajectory from this point.

The main program uses the subroutines, RUNGE, RNUMBR, DLOCAT, POLATE, RESET, BOUNCE and RESTLO. All of these subroutines have been described previously except RESET, which resets the average values used after each iteration, and linearly extrapolates the properties to estimate the average value over the next time step.

Input

There are two sets of input. The first set specifies gas flow properties, and consists of two dimensional arrays for magnitude and direction of the flow velocities within the field. This program works with any consistent system of units. An example of the input data is included with the example case presented after the program listing. The first set of input cards take the following form:

TITLE	(18A4)
MX, KMX, NB	(3I5)

GAMMA, TIP, RHOIP, RGAS	(4F10.6)
VISREF, TREF, TSUT, DGFC	(E20.4, 3F10.6)
ZMAX, FIRST	(2F10.6)
NSTEP, H	(I5, E10.2)
R(I,K) Array	(8F10.6)
THETA(I,K) Array	(8F10.6)
V(I,K) Array	(8F10.6)
BETA(I,K) Array	(8F10.6)

The second set of data cards specify the type of particle, its initial position and velocity. Each particle can be described by three input cards which take the following form.

TITLE	(18A4)
DIAP, RHOIP	(E20.4, F10.6)
(YR(I), I=1,6)	(6F10.6)

These variables are defined below. For studies that are done with multiple particles, additional sets of input data cards as indicated for the second data card set may be stacked together. When the program completes the trajectory for one particle, it goes to the next set automatically.

Variables

- TITLE - The first card is reproduced at the top of the first page of output. Any statement in columns 2 to 72 will be reproduced.
- MX - The number of blade to blade constant radius orthogonals.
- KMX - The number of streamlines used in the description of the flow.
- NB - The number of nozzle blades in the ring.
- GAMMA - The ratio of specific heats.

- TIP** - The gas inlet stagnation temperature. (Degrees Abs.)
- RHOIP** - The gas inlet stagnation density. (Mass/Length³)
- RGAS** - The gas constant. (Length²/Time² Degrees Abs.)
- VISREF** - Reference viscosity corresponding to TREF. Used in Sutherland's Law. (Mass/Length x Time)
- TREF** - Reference temperature corresponding to VISREF. Used in Sutherland's Law. (Degrees Abs.)
- TSUT** - Constant used in Sutherland's Law. (198.6°R or 110°K).
- DGFC** - Drag factor. The spherical drag coefficient based on Reynolds number is multiplied by DGFC. Except in very special cases, this should be 1.0.
- ZMAX** - The flow field extends in the axial direction between $z = 0.0$ and $z = ZMAX$. (Length).
- FIRST** - The angular position of the first blade. (Degrees). See Figure 22. All particles are transposed to a corresponding location in the channel of the first blade, its trajectory determined up till the exit from the blade row. The exit conditions are then moved back to the corresponding location at the original channel.
- NSTEP** - Integer that determines the amount of printed output. Output data is printed every NSTEP time increments.
- H** - Integration time increment. (Time). If extremely long computer run times are experienced, this can be made larger. If the program fails to converge, this can be made smaller.
- R array** - The radial position of the grid points of the flow field. (Length). The program is set up to use constant radius lines from blade to blade, as indicated in Figure 22.
- THETA Array** - The angular position of the grid points in the flow field. (Degrees). Indicated on Figure 22.
- V Array** - The normalized gas velocity at the grid point. (V/V_{cr})
- BETA Array** - The direction of the gas velocity vector at the corresponding grid point. (Degrees). Indicated in Figure 22.

- DIAP - Particle diameter. (Length)
- RHOP - Particle density. (Mass/Length³)
- YR(1) - Particle initial radial position. (Length)
- YR(2) - Particle initial radial velocity. The outward direction corresponds to the positive direction and the inward direction corresponds to the negative direction. (Length/Time)
- YR(3) - Particle initial angular position. (Degrees)
- YR(4) - Particle initial angular velocity. (1/Time)
- YR(5) - Particle initial axial position. (Length)
- YR(6) - Particle initial axial velocity. (Length/Time)

OUTPUT

The first part of the output is an echo check of the first set of data cards that describe the gas flow. Such data checks are useful in correcting key punch mistakes on the input cards. These checks cover the first five pages of the output and the four array variables are listed on separate pages. The printed output for each particle starts at the top of a new page with the echo check of the data that corresponds to the data cards for the particle.

Next, the program transposes the initial coordinates of the particle so that the particle enters the first passage. This is done by correcting the angular position of the particle so that the particle is in the required passage. The program notes this correction and writes "PARTICLE ENTERS PASSAGE XX INLET ANGLE CORRECTED TO XXXX". Next, the program calculates the similarity parameters that are useful in relating this particle to other particles that have similar trajectories. Finally, the program writes trajectory information every NSTEP time increments until the solution is complete, and then writes the last solution point and goes on to the next particle.

The additional terms of the output that are not defined as part of the input are listed below.

- DELTA - The characteristic length as given in Equation (7).
(Length)
- TAU - The time constant as given in Equation (8). (Time)
- RECR - The Reynolds number as given in Equation (9).
- M - An^o iteration counter. If the air velocity fails to converge to the proper average values after 100 steps,
M = 101.
- RENOLD - The Reynolds number of the particle at this point.

C
C
PARTICLE TRAJECTORIES IN RADIAL STATOR

INTEGER RUNGE

DIMENSION R(21,21), THETA(21,21), V(21,21), BETA(21,21), STATE(18),
1 ETA(2), YR(6), VR(4), VU(4), VZ(4), TEMPA(4), RHOA(4), VISTAR(4),
2 YRS(6), FR(6), A(3), RT(3), C(3), P(3), PP(3), PB(3), VP(3)

C
C
C
READ FLOW FIELD DATA

READ(5,1030) (STATE(I), I=1, 18)
WRITE(6,2030) (STATE(I), I=1, 18)
READ(5,1010) MX, KMX, NB
WRITE(6,2010) MX, KMX, NB
RFAD(5,1020) GAMMA, TIP, RHOIP, RGAS
WRITE(6,2011) GAMMA, TIP, RHOIP, RGAS
READ(5,1040) VISREF, TREF, TSUT, DGFC
WRITE(6,2031) VISREF, TREF, TSUT, DGFC
READ(5,1020) ZMAX, FIRST
WRITE(6,2012) ZMAX, FIRST
READ(5,1050) NSTEP, H
WRITE(6,2040) NSTEP, H
WRITE(6,2130)
DO 10 I=1, MX
READ(5,1020) (R(I,K), K=1, KMX)
10 WRITE(6,2013) (R(I,K), K=1, KMX)
WRITE(6,2130)
DO 20 I=1, MX
READ(5,1020) (THETA(I,K), K=1, KMX)
20 WRITE(6,2014) (THETA(I,K), K=1, KMX)
WRITE(6,2130)
DO 30 I=1, MX
READ(5,1020) (V(I,K), K=1, KMX)
30 WRITE(6,2015) (V(I,K), K=1, KMX)
WRITE(6,2130)
DO 40 I=1, MX
READ(5,1020) (BETA(I,K), K=1, KMX)
40 WRITE(6,2016) (BETA(I,K), K=1, KMX)
VCR=SQRT(2.0*GAMMA*RGAS*TIP/(GAMMA+1.0))
FIRST=FIRST/57.29577
DO 60 I=1, MX
DO 60 K=1, KMX
THETA(I,K)=THETA(I,K)/57.29577
BETA(I,K)=BETA(I,K)/57.29577
60 V(I,K)=V(I,K)*VCR

C
C
C
READ PARTICLE DATA

50 READ(5,1030) (STATE(I), I=1, 18)
WRITE(6,2030) (STATE(I), I=1, 18)
READ(5,1040) DIAP, RHOP
WRITE(6,2032) DIAP, RHOP
READ(5,1020) (YR(I), I=1, 6)
WRITE(6,2033) (YR(I), I=1, 6)

INITIALIZE

```

YR(3)=YR(3)/57.29577
DO 65 I=1,NB
TTEST=FIRST+6.283186*FLOAT(I-1)/FLOAT(NB)
IF(TTEST.GT.YR(3)) GO TO 66
65 CONTINUE
66 J=I-1
IPASS=J
YR(3)=YR(3)-FIRST-6.283186*FLOAT(J)/FLOAT(NB)
YR3=YR(3)*57.29577
WRITE(6,2140) J,YR3
EXPON=1.0/(GAMMA-1.0)
RHOCR=RHOIP*(2.0)/(GAMMA+1.0)**EXPON
DELTA=RHOP*DIAP/RHOCR/0.3
TCR=TIP*2.0/(GAMMA+1.0)
VISCR=VISREF*((TCR/TREF)**1.5)*(TREF+TSUT)/(TCR+TSUT)
TAU=RHOP*DIAP**2/18.0/VISCR
RECR=DIAP*RHOCR*VCR/VISCR/2.0
WRITE(6,2050) DELTA,TAU,RECR
T=0.0
L=0
YR3=YR(3)*57.29577
WRITE(6,2060) L,T,YR(1),YR(2),YR3,YR(4),YR(5),YR(6)
CP=GAMMA*RGAS/(GAMMA-1.0)
RPART=DIAP/2.0
N=6
M=1
NTIME=NSTEP
TS=T
ALPHA=-90.0/57.29577
TRPS=TREF+TSUT

```

DETERMINE AIR VELOCITIES AND PROPERTIES AT PARTICLE LOCATION

```

CALL DLOCAT(R,THETA,ZMAX,MX,KMX,YR,IP,KP,NOUT)
IF(NOUT.EQ.0) GO TO 70
WRITE(6,2070)
GO TO 50
70 CALL POLATE(R,THETA,V,YR(1),YR(3),IP,KP,VPP,DD)
CALL POLATE(R,THETA,BETA,YR(1),YR(3),IP,KP,BETAP,DD)
VR(1)=VPP*COS(BETAP)*SIN(ALPHA)
VU(1)=VPP*SIN(BETAP)
VZ(1)=VPP*COS(BETAP)*COS(ALPHA)
TEMPA(1)=TIP*(1.0-((GAMMA-1.0)/(GAMMA+1.0))*(VPP/VCR)**2)
RHOA(1)=RHOIP*(TEMPA(1)/TIP)**EXPON
VISTAR(1)=VISREF*((TEMPA(1)/TREF)**1.5)*(TRPS)/(TEMPA(1)+TSUT)
CALL RESET( VR,1)
CALL RESET( VU,1)
CALL RESET( VZ,1)
CALL RESET( TEMPA,1)
CALL RESET( RHOA,1)

```

```

CALL RESET(VISTAR,1)
DO 90 I=1,N
90 YRS(I)=YR(I)

C
C
C INTEGRATE USING RUNGE-KUTTA METHOD

100 VDIFF=SQRT((VR(2)-YR(2))**2+(VU(2)-YR(1)*YR(4))**2
1   +(VZ(2)-YR(6))**2)
RENOLD=RHOA(2)+VDIFF*DIAP/VISTAR(2)
CALL RNUMBR(RENOLD,DGFC,CD)
BCON=RHOA(2)*CD/RHOP/RPART/2.33333
110 IF(RUNGE(N,YR,FR,T,H).NE.1) GO TO 120
FR(1)=YR(2)
FR(2)=YR(1)*YR(4)**2+BCON*VDIFF*(VR(2)-YR(2))
FR(3)=YR(4)
FR(4)=-2.0*YR(2)*YR(4)/YR(1)+BCON*VDIFF*(VU(2)-YR(1)*YR(4))/YR(1)
FR(5)=YR(6)
FR(6)=BCON*VDIFF*(VZ(2)-YR(6))
GO TO 110
120 CONTINUE

C
C
C DETERMINE IF WALL INTERACTION OCCURRED.

CALL DLOCAT(R,THETA,ZMAX,MX,KMX,YR,IP,KP,NOUT)
IF(NOUT.EQ.0) GO TO 150
IF((NOUT.EQ.1).OR.(NOUT.EQ.3)) GO TO 200
GO TO 125
124 IP=IPS
125 A(1)=R(IP,KP)*COS(THETA(IP,KP))
A(2)=R(IP,KP)*SIN(THETA(IP,KP))
B(1)=R(IP-1,KP)*COS(THETA(IP-1,KP))
B(2)=R(IP-1,KP)*SIN(THETA(IP-1,KP))
C(1)=A(1)
C(2)=A(2)
IF((NOUT.EQ.5).OR.(NOUT.EQ.6)) GO TO 130
A(3)=YRS(5)
B(3)=YRS(5)
C(3)=YRS(5)+SQRT((A(1)-B(1))**2+(A(2)-B(2))**2+(A(3)-B(3))**2)
GO TO 140
130 IF(NOUT.EQ.5) A(3)=0.0
IF(NOUT.EQ.6) A(3)=ZMAX
B(3)=A(3)
C(3)=A(3)
140 P(1)=YRS(1)*COS(YRS(3))
P(2)=YRS(1)*SIN(YRS(3))
P(3)=YRS(5)
PP(1)=YR(1)*COS(YR(3))
PP(2)=YR(1)*SIN(YR(3))
PP(3)=YR(5)
T=TS
CALL BOUNCE(A,B,C,P,PP,H,T,ETA,NFIX,PB,VP)
IF(NFIX.EQ.0) GO TO 200
IF(NFIX.EQ.2) GO TO 124

```



```

YR(1)=SQRT(PB(1)**2+PB(2)**2)
YR(3)=ATAN2(PB(2),PB(1))
YR3=YR(3)*57.29577
YR(5)=PB(3)
WRITE(6,2080) NIUT,YR(1),YR3,YR(5)
YR(2)=VP(1)*COS(YR(3))+VP(2)*SIN(YR(3))
YR(4)=(-VP(1)*SIN(YR(3))+VP(2)*COS(YR(3)))/YR(1)
YR(6)=VP(3)
GO TO 170

```

```

C
C DETERMINE AIR VALUES AT NEW LOCATION.
C

```

```

150 CALL POLATE(R,T,HETA,V,YR(1),YR(3),IP,KP,VPP,DD)
CALL POLATE(R,T,HETA,BETA,YR(1),YR(3),IP,KP,BETAP,DD)
VR(4)=VPP*COS(BETAP)*SIN(ALPHA)
VU(4)=VPP*SIN(BETAP)
VZ(4)=VPP*COS(BETAP)*COS(ALPHA)
TEMPA(4)=TIP*(1.0-((GAMMA-1.0)/(GAMMA+1.0))*(VPP/VCR)**2)
RHOA(4)=RHOIP*(TEMPA(4)/TIP)**EXPON
VISTAR(4)=VISREF*((TEMPA(4)/TREF)**1.5)*(TRPS)/(TEMPA(4)+TSUT)

```

```

C
C TEST AIR VALUES USED, IF INCORRECT, RESET INTEGRATION VALUES AND
C GO TO 100. IF CORRECT, GO TO 170.
C

```

```

IF((ABS(VR(4)-VR(3)).LT.1.0E-4).AND.(ABS(VU(4)-VU(3)).LT.1.0E-4)
& .AND.(ABS(VZ(4)-VZ(3)).LT.1.0E-4))GO TO 170

```

```

CALL RESET( VR,2)
CALL RESET( VU,2)
CALL RESET( VZ,2)
CALL RESET( TEMPA,2)
CALL RESET( RHOA,2)
CALL RESET(VISTAR,2)
T=TS

```

```

DO 160 I=1,6
160 YR(I)=YRS(I)
IF(M.GT.100) GO TO 200
M=M+1
GO TO 100

```

```

C
C COMPLETE INTEGRATION STEP.
C

```

```

170 M=1
L=L+1
IPS=IP
TS=T
DO 180 I=1,6
180 YRS(I)=YR(I)
CALL RESET( VZ,3)
CALL RESET( VU,3)
CALL RESET( VR,3)
CALL RESET( TEMPA,3)
CALL RESET( RHOA,3)
CALL RESET(VISTAR,3)

```

NTIME=NTIME-1

C
C
C

IF REQUIRED, WRITE OUTPUT.

```
190 IF((ABS(YR(2))+ABS(YR(4))+ABS(YR(6))).LT.1.0E-4) GO TO 200
    IF(NTIME.GT.0) GO TO 100
    YR3=YR(3)*57.29577
    WRITE(6,2100) L,T,YR(1),YR(2),YR3,YR(4),YR(5),YR(6),RENOLD
    NTIME=NSTEP-1
    IF(L.GT.1) NTIME=NSTEP
    GO TO 100
200 CONTINUE
    YR(3)=YR(3)+FIRST+6.2831853/FLOAT(NB)*FLOAT(IPASS)
    YR3=YR(3)*57.29577
    WRITE(6,2110)
    WRITE(6,2100) L,T,YR(1),YR(2),YR3,YR(4),YR(5),YR(6),RENOLD
    WRITE(6,2120) M
    GO TO 50
```

C
C
C

FORMAT STATEMENTS.

```
1010 FORMAT(14I5)
1020 FJRMAT(8F10.6)
1030 FJRMAT(18A4)
1040 FORMAT(E20.4,3F10.6)
1050 FJRMAT(15,E10.2)
2010 FORMAT(1H0,15HFLOW FIELD DATA,/,8X,2HMX,7X,3HKMX,8X,2HNB,/,
1 (3I10))
2011 FJRMAT(1H0,9X,5HGAMMA,12X,3HTIP,10X,5HRHOIP,11X,4HRGAS,/(4F15.6))
2012 FORMAT(1H0,10X,4HZMAX,10X,5HFIRST,/(2F15.6))
2013 FJRMAT(1H ,4X,6HR(I,K),(8F15.6))
2014 FJRMAT(1H ,10HTETA(I,K),(8F15.6))
2015 FORMAT(1H ,4X,6HV(I,K),(8F15.6))
2016 FJRMAT(1H ,1X,9HBETA(I,K),(8F15.6))
2030 FJRMAT(1H1,18A4)
2031 FORMAT(1H0,13X,6HVISREF,11X,4HTREF,11X,4HTSUT,11X,4HDGFC,/,
1 (E20.4,3F15.6))
2032 FORMAT(1H0,15X,4HDIAP,11X,4HRHOP,/(E20.4,F15.6))
2033 FJRMAT(1H0, 9X,5HYR(1),10X,5HYR(2),10X,5HYR(3),10X,5HYR(4),10X,
1 5HYR(5),10X,5HYR(6), /,(6F15.6))
2040 FORMAT(1H0,4X,5HNSTEP,14X,1HH,/(110,E15.2))
2050 FJRMAT(32HOSIMILARITY PARAMETERS. DELTA =,F10.4,5X,5HTAJ =,E12.4,
1 5X,6HRECR =,E12.4)
2060 FORMAT(1H0,6X,4HSTEP,9X,1HT,5X,5HYR(1),5X,5HYR(2),5X,5HYR(3),5X,
1 5HYR(4),5X,5HYR(5),5X,5HYR(6),4X,6HRENOLD, /,
2 (111,E10.2,F10.5,F10.2,F10.3,F10.2,F10.5,F10.2))
2070 FJRMAT(1H0,43HPARTICLE OUT OF BOUNDS AT FIRST POINT GIVEN)
2080 FJRMAT(11H BOUNCE OFF,14,6X,F10.5,10X,F10.3,10X,F10.5)
2100 FORMAT(1H ,110,E10.2,F10.5,F10.2,F10.3,F10.2,F10.5,F10.2,E20.4)
2110 FORMAT(1H0)
2120 FORMAT(3HOM=,14)
2130 FORMAT(1H1)
2140 FJRMAT(24HPARTICLE ENTERS PASSAGE,15,5X,24HINLET ANGLE CORRECTED
```

```
1TO,(F15.6))
STOP
END
```

```
SUBROUTINE DLLOCAT(R,Z,XMAX,MX,KMX,YR,IP,KP,NOUT)
DIMENSION R(21,21),Z(21,21),YR(6)
NOUT=0
DO 20 I=1,MX
IF(R(I,1).LE.YR(1)) GO TO 30
20 CONTINUE
IP=MX
NOUT=3
RETURN
30 IP=I
IF(IP.NE.1) GO TO 50
NOUT=1
RETURN
50 CONTINUE
DO 70 K=1,KMX
X1=R(IP,K)*COS(Z(IP,K))
Y1=R(IP,K)*SIN(Z(IP,K))
X2=R(IP-1,K)*COS(Z(IP-1,K))
Y2=R(IP-1,K)*SIN(Z(IP-1,K))
PX=YR(1)*COS(YR(3))
PY=YR(1)*SIN(YR(3))
IF(ABS(X1-X2).LT.1.0E-12) GO TO 65
A=(Y1-Y2)/(X1-X2)
B=Y1-A*X1
YTEST=A*PX+B
GO TO 66
65 YTEST=Y1
66 CONTINUE
IF(PY.GE.YTEST) GO TO 80
70 CONTINUE
KP=KMX
NOUT=4
RETURN
80 KP=K
IF(KP.NE.1) GO TO 120
NOUT=2
RETURN
120 IF(YR(5).GT.0.0) GO TO 130
NOUT=5
RETURN
130 IF(YR(5).LT.XMAX) RETURN
NOUT=6
RETURN
END
```

```
SUBROUTINE RESET(A,I)
DIMENSION A(4)
```

```
C
C      THIS SUBROUTINE RESETS THE VALUES OF A VARIABLE SO THAT THE
C      BEST ESTIMATE OF THE AVERAGE VALUE OF THE VARIABLE CAN BE USED IN
C      THE INTEGRATION STEP.
C      A(1) IS THE VALUE OF A AT POINT 1 FOR THE INTEGRATION STEP.
C      A(2) IS THE AVERAGE VALUE OF A OVER THE INTEGRATION STEP.
C      A(3) IS THE VALUE OF A AT POINT 2 USED IN THIS INTEGRATION STEP.
C      A(4) IS THE VALUE OF A AT POINT 2 CALCULATED AFTER THE INTEGRATION.
C
C      GO TO (10,20,30),I
C
C      FOR I=1, SET UP ARRAYS.
C
C      10 DO 15 J=2,4
C      15 A(J)=A(1)
C      RETURN
C
C      FOR I=2, RESET AVERAGE VALUE OF A, A(2), AND REMEMBER THE LATEST
C      VALUE OF A AT THE END OF THE STEP.
C
C      20 A(2)=(A(4)+A(1))/2.0
C      A(3)=A(4)
C      RETURN
C
C      FOR I=3, SYSTEM HAS CONVERGED. ESTIMATE THE AVERAGE VALUES BY
C      LINEARLY EXTENDING THE VALUES DETERMINED IN THE PREVIOUS POINTS.
C
C      30 A(2)=1.5*A(4)-0.5*A(1)
C      A(3)=2.0*A(4)-A(1)
C      A(1)=A(4)
C      A(4)=A(3)
C      RETURN
C      END
```

The function routine RUNGE has been removed from the published form of this report to protect the copyright of the authors of Reference 5.

```
SUBROUTINE RESTCO(VN,VT,ETA)
```

```
  DIMENSION ETA(2),A(10),B(10)
```

```
  DATA A/1.8190,5.1171,-49.2529,131.03528,-174.4249,117.8723,
```

```
1  -24.47885,-16.7297,11.2300,-1.979996/
```

```
  DATA B/5.7215,-41.8808,178.1685,-424.3892,572.7631,-406.6625,
```

```
1  87.1429,70.6511,-50.4982,9.6767/
```

```
  DATA IN THIS SUBROUTINE CORRESPONDS TO VELOCITIES IN FT/SEC.
```

```
  MATERIAL TYPICAL OF ALUMINUM AND SILICON PARTICLES.
```

```
  ETA(1) IS THE NORMAL RESTITUTION COEFFICIENT.
```

```
  ETA(2) IS THE TANGENTIAL RESTITUTION COEFFICIENT.
```

```
  BETA=ATAN2(VN,VT)
```

```
  V=SQRT(VN**2+VT**2)
```

```
  PHIONE=V/250.00
```

```
  PHITWO= A(1)*BETA+A(2)*BETA**2+A(3)*BETA**3+A(4)*BETA**4
```

```
1  +A(5)*BETA**5+A(6)*BETA**6+A(7)*BETA**7+A(8)*BETA**8
```

```
2  +A(9)*BETA**9+A(10)*BETA**10
```

```
  PHI=PHIONE*PHITWO
```

```
  IF(PHI.GT.0.90) PHI=0.90
```

```
  ETA(2)=1.000-PHI
```

```
  PSIONE=1.0000-EXP(-V/36.000)
```

```
  PSITWO= B(1)*BETA+B(2)*BETA**2+B(3)*BETA**3+B(4)*BETA**4
```

```
1  +B(5)*BETA**5+B(6)*BETA**6+B(7)*BETA**7+B(8)*BETA**8
```

```
1  +B(9)*BETA**9+B(10)*BETA**10
```

```
  PSI=PSIONE*PSITWO
```

```
  IF(PHI.GT.0.900) PSI=0.900
```

```
  ETA(1)=1.000-PSI
```

```
  RETURN
```

```
  END
```

```
SUBROUTINE RNJMBR(RENOLD,DGFC,CD)
```

```
  IF(ABS(RENOLD).LT.1.0E-12) RENOLD=1.0E-12
```

```
  IF(RENOLD.LT.1.0) GO TO 26
```

```
  IF((RENOLD.GE.1.0).AND.(RENOLD.LT.1.0E3)) GO TO 27
```

```
  CD=DGFC*0.4
```

```
  RETURN
```

```
26 CD=DGFC*(4.5+24.0/RENOLD)
```

```
  RETURN
```

```
27 ARE=ALOG(RENOLD)
```

```
  CD=(28.5-24.0*ARE+9.0682*ARE**2-1.7713*ARE**3+0.1718*ARE**4
```

```
1  -0.0065*ARE**5)*DGFC
```

```
  RETURN
```

```
  END
```

```
SUBROUTINE POLATE(P,Z,A,RP,ZP,IP,KP,AP,DD)
DIMENSION R(21,21),Z(21,21),A(21,21),D(2),DD(2)
DO 10 I=1,2
IA=IP+1-I
IF(ABS(Z(IA,KP-1)-Z(IA,KP)).LT.1.0E-12) GO TO 5
IF(ABS(R(IA,KP)-R(IA,KP-1)).LT.1.0E-12) GO TO 6
AM=(R(IA,KP)-R(IA,KP-1))/(Z(IA,KP)-Z(IA,KP-1))
B1=R(IA,KP-1)-AM*Z(IA,KP-1)
B2=RP+ZP/AM
ZA=(B2-B1)*AM/(AM**2+1.0)
RA=B2-ZA/AM
GO TO 10
5 RA=RP
ZA=Z(IA,KP-1)
GO TO 10
6 RA=R(IA,KP-1)
ZA=ZP
10 D(I)=SQRT((RA-RP)**2+(ZA-ZP)**2)
DT=D(1)+D(2)
AA=(D(1)*A(IP,KP-1)+D(2)*A(IP-1,KP-1))/DT
AB=(D(1)*A(IP,KP)+D(2)*A(IP-1,KP))/DT
DO 20 K=1,2
KA=KP-K+1
RC=(D(1)*R(IP-1,KA)+D(2)*R(IP,KA))/DT
ZC=(D(1)*Z(IP-1,KA)+D(2)*Z(IP,KA))/DT
20 DD(K)=SQRT((RP-RC)**2+(ZP-ZC)**2)
DT=DD(1)+DD(2)
AP=(DD(1)*AA+DD(2)*AB)/DT
RETURN
END
```

```

SUBROUTINE BOUNCE(A,H,C,P,PP,H,T,ETA,NFIX,PB,VP)
DIMENSION A(3),B(3),C(3),P(3),PP(3),V(3),AB(3),AC(3),G(3,3),J(5),
1 PB(3)
DIMENSION GS(3,3),VP(3),UN(3),PN(3)
DIMENSION ETA(2)
DIMENSION PNP(3),VN(4),VT(4)

C
NFIX=1
DO 10 I=1,3
V(I)=(PP(I)-P(I))/H
AB(I)=B(I)-A(I)
10 AC(I)=C(I)-A(I)
VPP=SQRT(V(1)**2+V(2)**2+V(3)**2)
C DETERMINE UNIT NORMAL TO SURFACE
C
UN(1)=AB(2)*AC(3)-AB(3)*AC(2)
UN(2)=AB(3)*AC(1)-AB(1)*AC(3)
UN(3)=AB(1)*AC(2)-AB(2)*AC(1)
CMAG= SQRT(UN(1)**2+UN(2)**2+UN(3)**2)
IF( ABS(CMAG).GT.1.0E-12) GO TO 20
NFIX=0
WRITE(6,1000)
1000 FORMAT(47H BOUNCE HAS ZERO UNIT VECTOR DESCRIBING SURFACE)
RETURN
20 DO 30 I=1,3
30 UN(I)=UN(I)/CMAG

C
C DETERMINE THE INTERSECTION POINT, PB, OF THE PLANE AND TRAJECTORY.
C
DETA=UN(1)*V(1)**2+UN(2)*V(1)*V(2)+UN(3)*V(1)*V(3)
J(1)=UN(1)*A(1)+UN(2)*A(2)+UN(3)*A(3)
D(2)=V(2)*P(1)-V(1)*P(2)
J(3)=V(3)*P(1)-V(1)*P(3)
DO 40 J=1,3
40 G(I,J)=UN(J)
G(2,1)= V(2)
G(2,2)=-V(1)
G(2,3)=0.0
G(3,1)= V(3)
G(3,2)=0.0
G(3,3)=-V(1)

C
C IF DETERMINANT EQUALS ZERO, GO TO 90
C
IF( ABS(DETA).LE.1.0E-12) GO TO 90
DO 70 K=1,3
DO 50 I=1,3
DO 50 J=1,3
50 GS(I,J)=G(I,J)
DO 60 I=1,3
60 GS(I,K)= D(I)
PB(K)=GS(1,1)*GS(2,2)*GS(3,3)+GS(1,2)*GS(2,3)*GS(3,1)+GS(1,3)
1 *GS(2,1)*GS(3,2)-GS(3,1)*GS(2,2)*GS(1,3)-GS(3,2)*GS(2,3)*GS(1,1)

```

```

2 -GS(3,3)*GS(2,1)*GS(1,2)
70 PB(K)=PB(K)/DETA
GO TO 100

```

C
C
C

```

IF DETERMINANT EQUALS ZERO, POINT P IS ON SURFACE, P EQUALS PB

```

```

80 PB(1)=P(1)
D(4)=D(1)-UN(1)*P(1)
D(5)=V(3)*P(2)-V(2)*P(3)
DETA=-UN(2)*V(2)-UN(3)*V(3)
IF( ABS(DETA).LT.1.0E-12) GO TO 85
PB(2)=(-D(4)*V(2)-D(5)*UN(3))/DETA
PB(3)=(UN(2)*D(5)-V(3)*D(4))/DETA
GO TO 100

```

```

85 IF( ABS(V(3)).GT.1.0E-12) GO TO 90
PB(2)=P(2)
PB(3)=A(3)
GO TO 100

```

```

90 PB(2)=A(2)
PB(3)=P(3)

```

```

100 CONTINUE
DPP= SQRT((PP(1)-P(1))**2+(PP(2)-P(2))**2+(PP(3)-P(3))**2)
DPB= SQRT((PB(1)-P(1))**2+(PB(2)-P(2))**2+(PB(3)-P(3))**2)
DPPB=SQRT((PP(1)-PB(1))**2+(PP(2)-PB(2))**2+(PP(3)-PB(3))**2)
IF((DPPB.LT.DPP).AND.(DPB.LT.DPP)) GO TO 103
NFIX=2
RETURN

```

```

103 CONTINUE
IF(DPB.LT.(0.5*DPP)) GO TO 180

```

C
C
C
C

```

DETERMINE THE INTERSECTION POINT, PN, OF THE SURFACE NORMAL THRU P
IF DETERMINANT EQUALS ZERO, GO TO 140

```

```

DETA=UN(1)**3+UN(1)*UN(2)**2+UN(1)*UN(3)**2
IF( ABS(DETA).LE.1.0E-12) GO TO 140
D(2)=P(1)*UN(2)-P(2)*UN(1)
D(3)=P(1)*UN(3)-P(3)*UN(1)
G(2,1)= UN(2)
G(2,2)=-UN(1)
G(2,3)=0.0
G(3,1)= UN(3)
G(3,2)=0.0
G(3,3)=-UN(1)
DO 130 K=1,3
DO 110 I=1,3
DO 110 J=1,3

```

```

110 GS(I,J)=G(I,J)
DO 120 I=1,3

```

```

120 GS(I,K)=D(I)
PN(K)=GS(1,1)*GS(2,2)*GS(3,3)+GS(1,2)*GS(2,3)*GS(3,1)+GS(1,3)
1 *GS(2,1)*GS(3,2)-GS(3,1)*GS(2,2)*GS(1,3)-GS(3,2)*GS(2,3)*GS(1,1)
2 -GS(3,3)*GS(2,1)*GS(1,2)
130 PN(K)=PN(K)/DETA

```



```

      GO TO 160
140  PN(1)=P(1)
      D(4)=D(1)
      J(5)=UN(3)*P(2)-UN(2)*P(3)
      DELTA=-UN(2)**2-JN(3)**2
      PN(2)=(-D(4)*UN(2)-D(5)*UN(3))/DELTA
      PN(3)=(UN(2)*J(5)-UN(3)*D(4))/DELTA
160  CONTINUE
C
C   DETERMINE PORTION OF TIME SEGMENT USED TO TRAVEL FROM P TO PB.
C
      DTIME=DPB/VPP
      IF(DTIME.LT.H) GO TO 163
      WRITE(6,1010)
1010  FORMAT(24H DTIME IS GREATER THAN H)
      VJUT=J
      RETURN
C
C   EXTENT LINE PN-PB THE PROPER DISTANCE TO FIND PNP.
C   THEN EXTENT A LINE NORMAL TO THE SURFACE FROM PNP TO GET THE POINT
C   AFTER BOUNCE, PP.
C   FIND VELOCITY COORDINATES BASED ON PP, PB AND TIME REMAINING IN
C   SEGMENT.
C
163  DO 165 I=1,3
      VT(I)=(PB(I)-PN(I))/DTIME
165  VN(I)=(PN(I)-P(I))/DTIME
      VT(4)= SQRT(VT(1)**2+VT(2)**2+VT(3)**2)
      VN(4)= SQRT(VN(1)**2+VN(2)**2+VN(3)**2)
      CALL RESTCO(VN(4),VT(4),ETA)
      DO 170 I=1,3
      PNP(I)=PB(I)+ETA(2)*VT(I)*(H-DTIME)
      PP(I)=PNP(I)-ETA(1)*VN(I)*(H-DTIME)
170  VP(I)=(PP(I)-PB(I))/(H-DTIME)
      T=T+H
      RETURN
C
C   IF POINT P LIES ON SURFACE, USE POINT PP TO DETERMINE AFTER
C   BOUNCE STATF.
C
180  CONTINUE
      DELTA=UN(1)**3+JN(1)*UN(2)**2+UN(1)*UN(3)**2
      IF(ABS(DELTA).LE.1.0E-12) GO TO 220
      D(2)=PP(1)*UN(2)-PP(2)*UN(1)
      D(3)=PP(1)*UN(3)-PP(3)*UN(1)
      G(2,1)= UN(2)
      G(2,2)=-JN(1)
      G(2,3)=0.0
      G(3,1)=UN(3)
      G(3,2)=0.0
      G(3,3)=-UN(1)
      DO 210 K=1,3
      DO 190 I=1,3

```

```

DO 190 J=1,3
190 GS(I,J)=G(I,J)
DO 200 I=1,3
200 GS(I,K)=D(I)
    PN(K)=GS(1,1)*GS(2,2)*GS(3,3)+GS(1,2)*GS(2,3)*GS(3,1)+GS(1,3)
    1 *GS(2,1)*GS(3,2)-GS(3,1)*GS(2,2)*GS(1,3)-GS(3,2)*GS(2,3)*GS(1,1)
    2 -GS(3,3)*GS(2,1)*GS(1,2)
210 PN(K)=PN(K)/DETA
    GO TO 240
220 PV(1)=PP(1)
    D(4)=D(1)
    D(5)=UN(3)*PP(2)-UN(2)*PP(3)
    DETA=-UN(2)*UN(2)-UN(3)*UN(3)
    PN(2)=(-D(4)*UN(2)-D(5)*UN(3))/DETA
    PN(3)=(UN(2)*D(5)-UN(3)*D(4))/DETA
240 CONTINUE
    
```

C
C DETERMINE PORTION OF TIME SEGMENT REMAINING FOR TRAVEL FROM PB TO PP.
C

```

    DTIME=H-DPB/VPP
    IF(DTIME.GT.1.0E-12) GO TO 245
    WRITE(6,1020)
1020 FORMAT(21H DTIME LESS THAN ZERO)
    NFIX=0
    RETURN
    
```

C
C DETERMINE THE PROPER DISTANCE ALONG PN-PB TO FIND PNP.
C THEN EXTEND A LINE NORMAL TO THE SURFACE FROM PNP TO GET THE POINT
C AFTER BOUNCE, PP.
C FIND VELOCITY COORDINATES BASED ON PP, PB, AND THE TIME H.
C

```

245 DO 250 I=1,3
    VT(I)=(PN(I)-PB(I))/DTIME
250 VV(I)=(PP(I)-PN(I))/DTIME
    VT(4)=SQRT(VT(1)**2+VT(2)**2+VT(3)**2)
    VV(4)=SQRT(VV(1)**2+VV(2)**2+VV(3)**2)
    CALL RESTCO(VV(4),VT(4),ETA)
    DO 260 I=1,3
    PNP(I)=PB(I)+ETA(2)*VT(I)*DTIME
    PP(I)=PNP(I)-ETA(1)*VV(I)*DTIME
260 VP(I)=(PP(I)-PB(I))/DTIME
    T=T+H
    RETURN
    END
    
```

Example

The example case presented here used the ft., lbm., second system of units. The gas flow conditions correspond to inlet stagnation conditions of standard sea level air. In the output, the R, THETA, V, and BETA arrays have been combined onto one page. The nozzle blades lie between radii of 0.274 ft., and 0.321 ft. The velocities are expressed in terms of V/V_{cr} , as can be seen from the output array.

The particle used in the example has a specific gravity of 3 and a diameter of approximately 24 microns. Initially, the particle has a velocity that is equal to one half the gas velocity. The output for the particle indicates that the particle enters passage 26, but the angular position has been corrected to correspond to the proper position in the passage for which the data on velocity and velocity direction apply.

The trajectory data illustrates a bounce off the pressure surface of the blade, in this case the iteration scheme failed to converge in 100 steps. This can be corrected by making the time step smaller.

The following pages contain a computer code sheet with the data arranged in the proper columns, and the output for this example.

STATOR Example Case for NASA Report																																																																															
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63	64	65	66	67	68	69	70	71	72	73	74	75	76	77	78	79	80
RADIAL STATOR CASE FOR THESIS AND NASA REPORT																																																																															
10	5	29	/																																																																												
	1.4	518.7	0.0748	1715.48	/																																																																										
		0.106E-4	492.0	198.2	1.0	/																																																																									
	0.0264	12.41	/																																																																												
10	0.1E-5	/																																																																													
	0.321	0.321	0.321	0.321	0.321	/																																																																									
.	/																																																																									
	0.274	0.274	0.274	0.274	0.274	/																																																																									
	-0.41	-3.60	-6.50	-9.00	-12.81	/																																																																									
.	/																																																																									
	15.00	12.00	9.50	7.00	4.75	/																																																																									
	0.142	0.139	0.124	0.107	0.074	/																																																																									
.	/																																																																									
	0.786	0.786	0.786	0.786	0.786	/																																																																									
	-12.00	20.00	31.50	39.00	32.00	/																																																																									
.	/																																																																									
	79.00	77.50	74.80	73.20	72.50	/																																																																									
EXAMPLE CASE FOR NASA REPORT V-PART/V-GAS = 50%.																																																																															
		0.774E-4	187.2	/																																																																											
	0.3209	-107.84	328.97	205.95	0.0132	0.0	/																																																																								
Additional particle trajectories require a Card Set 2 arrangement here.																																																																															
These sets are stacked one after the other.																																																																															

CARD SET 1

CARD SET 2

CARD SET 2

RADIAL STATOR CASE FOR THESIS AND NASA REPORT

FICH FIELD DATA

NR	NPX	NR		
10	5	29		
GAMMA	TIP	RMCIP	RGAS	
1.400000	510.699991	0.074800	1715.479980	
VISRFF	TREF	TSLT	DGFC	
0.1060E-04	492.000000	198.199997	1.000000	
TMAX	FIRST			
0.026400	12.410000			
NSTEP	F			
10	0.10E-05			

ORIGINAL PAGE IS
OF POOR QUALITY

RIT.K)	0.321000	0.321000	0.321000	0.321000	0.321000
RIT.K)	0.319000	0.319000	0.319000	0.319000	0.319000
RIT.K)	0.316000	0.316000	0.316000	0.316000	0.316000
RIT.K)	0.312000	0.312000	0.312000	0.312000	0.312000
RIT.K)	0.308000	0.308000	0.308000	0.308000	0.308000
RIT.K)	0.304000	0.304000	0.304000	0.304000	0.304000
RIT.K)	0.297500	0.297500	0.297500	0.297500	0.297500
RIT.K)	0.291000	0.291000	0.291000	0.291000	0.291000
RIT.K)	0.282500	0.282500	0.282500	0.282500	0.282500
RIT.K)	0.274000	0.274000	0.274000	0.274000	0.274000

TFTAIT.K)	-0.410000	-3.559999	-6.500000	-5.000000	-12.813999
TFTAIT.K)	-0.750000	-3.200000	-6.000000	-6.759999	-11.580000
TFTAIT.K)	-1.160000	-3.059999	-5.759999	-8.099999	-11.000000
TFTAIT.K)	-1.160000	-2.900000	-5.059999	-7.500000	-10.330000
TFTAIT.K)	-0.750000	-2.200000	-4.259999	-6.700000	-9.419999
TFTAIT.K)	-0.330000	-1.559999	-3.559999	-5.500000	-8.419999
TFTAIT.K)	0.920000	0.0	-2.000000	-4.000000	-7.500000
TFTAIT.K)	3.099999	2.000000	0.0	-2.000000	-4.160000
TFTAIT.K)	7.900000	6.000000	4.000000	1.500000	-0.160000
TFTAIT.K)	15.000000	12.000000	9.500000	7.000000	4.750000

VIT.K)	0.142000	0.135000	0.124000	0.107000	0.074000
VIT.K)	0.152000	0.146000	0.140000	0.113000	0.098000
VIT.K)	0.214000	0.173000	0.154000	0.126000	0.103000
VIT.K)	0.285000	0.211000	0.182000	0.146000	0.107000
VIT.K)	0.353000	0.283000	0.207000	0.174000	0.123000
VIT.K)	0.418000	0.335000	0.255000	0.215000	0.163000
VIT.K)	0.559000	0.448000	0.373000	0.276000	0.231000
VIT.K)	0.604000	0.582000	0.455000	0.404000	0.306000
VIT.K)	0.676000	0.634000	0.624000	0.581000	0.535000
VIT.K)	0.786000	0.786000	0.786000	0.786000	0.786000

PFTAIT.K)	-12.000000	20.000000	21.500000	35.000000	32.000000
PFTAIT.K)	-41.799999	20.199999	24.500000	40.000000	44.199999
PFTAIT.K)	-20.000000	57.500000	31.500000	42.000000	43.500000
PFTAIT.K)	14.500000	20.659999	42.000000	45.599999	48.000000
PFTAIT.K)	25.500000	28.000000	45.000000	48.500000	52.000000
PFTAIT.K)	38.799999	45.059999	48.000000	51.199999	55.000000
PFTAIT.K)	53.000000	54.000000	55.000000	56.000000	59.799999
PFTAIT.K)	67.000000	61.000000	61.000000	62.000000	64.199999
PFTAIT.K)	72.500000	71.000000	65.000000	66.000000	68.500000
PFTAIT.K)	79.000000	77.500000	74.799999	73.199999	72.500000

ORIGINAL PAGE IS
OF POOR QUALITY

FIAP 0.7740F-04 RMPF 107.195957

VR(1) 0.370900 VR(2) -107.895956 VR(3) 378.569571 VR(4) 205.949957 VR(5) 0.013200 VR(6) 0.0

PARTICLE ENTERS PASSAGE 26 INLET ANGLE CORRECTED TO -6.158586

SIMILARITY PARAMETERS. DELTA = 1.0125 T/L = 0.6520E-02 RECF = 0.1957E 03

STPF	T	VF(1)	VF(2)	VF(3)	VF(4)	VF(5)	VF(6)	RENCLD
0	0.0	0.370900	-107.895956	378.569571	205.949957	0.013200	0.0	
10	0.10F-04	0.31582	-107.72	-6.155	205.95	0.01320	0.0	0.3972E C1
20	0.20F-04	0.21874	-107.61	-5.561	208.93	0.01320	0.00	0.1700E G2
30	0.30F-04	0.21767	-107.53	-5.541	210.66	0.01320	0.00	0.1356E G2
40	0.40F-04	0.31659	-107.44	-5.719	212.32	0.01320	0.00	0.1065E G2
50	0.50F-04	0.31552	-107.40	-5.597	214.28	0.01320	0.00	0.3201E G2
60	0.60F-04	0.31445	-107.40	-5.474	216.57	0.01320	0.00	0.2725E G2
70	0.70F-04	0.31337	-107.37	-5.349	218.69	0.01320	0.00	0.2239E G2
80	0.80F-04	0.31230	-107.30	-5.222	220.65	0.01320	0.00	0.1757E G2
90	0.90F-04	0.31122	-107.26	-5.096	223.45	0.01320	0.00	0.4311E G2
100	0.10F-03	0.31015	-107.42	-4.967	226.39	0.01320	0.00	0.3843E G2
110	0.11F-03	0.30908	-107.44	-4.837	229.13	0.01320	0.00	0.3376E G2
120	0.12F-03	0.30800	-107.47	-4.705	231.68	0.01320	0.00	0.2916E G2
130	0.13F-03	0.30692	-107.67	-4.571	235.98	0.01320	0.00	0.6057E G2
140	0.14F-03	0.30585	-107.65	-4.434	239.98	0.01320	0.00	0.5336E G2
150	0.15F-03	0.30477	-107.56	-4.296	243.59	0.01320	0.00	0.4587E G2
160	0.16F-03	0.30369	-108.16	-4.155	248.17	0.01320	0.00	0.1164E G3
170	0.17F-03	0.30260	-108.80	-4.011	256.17	0.01320	0.00	0.1039E G3
180	0.18F-03	0.30151	-109.20	-3.862	263.29	0.01320	0.00	0.9151E C2
190	0.19F-03	0.30042	-109.68	-3.705	269.67	0.01320	0.00	0.8051E C2
200	0.20F-03	0.29932	-109.54	-3.553	275.23	0.01320	0.00	0.6577E G2
210	0.21F-03	0.29822	-110.05	-3.394	280.20	0.01320	0.00	0.5572E G2
220	0.22F-03	0.29712	-110.47	-3.232	287.62	0.01320	0.00	0.1556E C3
230	0.23F-03	0.29601	-111.14	-3.062	299.64	0.01320	0.00	0.1356E C3
240	0.24F-03	0.29485	-111.71	-2.888	310.35	0.01320	0.00	0.1236E G3
250	0.25F-03	0.29377	-112.12	-2.708	319.79	0.01320	0.00	0.1075E C3
260	0.26F-03	0.29265	-112.38	-2.522	328.01	0.01320	0.00	0.9176E G2
270	0.27F-03	0.29153	-112.45	-2.332	335.14	0.01320	0.00	0.7763E G2
280	0.28F-03	0.29040	-112.58	-2.131	348.96	0.01320	0.00	0.2103E G3
290	0.29F-03	0.28926	-113.73	-1.922	367.65	0.01320	0.00	0.1935E C3
300	0.30F-03	0.28812	-114.32	-1.716	384.69	0.01320	0.00	0.1756E C3
310	0.31F-03	0.28698	-114.72	-1.491	400.05	0.01320	0.00	0.1566E G3
320	0.32F-03	0.28583	-114.54	-1.258	413.68	0.01320	0.00	0.1365E G3
330	0.33F-03	0.28468	-114.55	-1.018	425.53	0.01320	0.00	0.1153E C3
340	0.34F-03	0.28354	-13.31	-0.778	417.21	0.01320	-0.02	0.1200E G3
350	0.35F-03	0.28240	-14.12	-0.537	424.36	0.01320	-0.02	0.1204E G3
360	0.36F-03	0.28127	-14.56	-0.292	431.59	0.01320	-0.02	0.1214E G3
370	0.37F-03	0.28013	-15.75	-0.043	438.91	0.01320	-0.02	0.1224E G3
380	0.38F-03	0.27900	-16.63	0.211	446.32	0.01320	-0.02	0.1232E G3
390	0.39F-03	0.27786	-17.49	0.469	453.81	0.01320	-0.02	0.1242E G3
400	0.40F-03	0.27672	-18.35	0.721	461.38	0.01320	-0.02	0.1252E G3
410	0.41F-03	0.27558	-19.22	0.968	469.04	0.01320	-0.02	0.1262E G3
420	0.42F-03	0.27444	-20.49	1.265	483.81	0.01320	-0.02	0.2890E G3
430	0.43F-03	0.27329	-22.39	1.554	511.38	0.01320	-0.02	0.2835E G3
440	0.44F-03	0.27215	-23.29	1.871	524.95	0.01320	-0.02	0.2860E G3

ROTOR - Particles in the Rotor

This program calculates the trajectory of a particle in a radial inflow turbine rotor. The gas flow solution is based on the quasi-orthogonal method, of Reference 9. The program that was given in Reference 9 is modified to provide required output on data cards. To avoid confusion, a listing of the modified program is included here as Part A of the Program Listing.

Method

The flow diagram of the program is given in Figure 23. The results of the fluid solutions are stored in arrays that specify the gas velocity vectors at the intersection points of the quasi-orthogonals and streamlines. With the fluid solution arrays input, the program goes to a RUNGE-KUTTA technique to integrate the three-dimensional equations of motion of the particle.

The integration of the equations of motion over one time increment, is first carried out using the velocity and the gas properties at point A to determine the new particle location B. The program then calls ALOCAT and POLATE to determine the gas velocity components at B_1 . A corrected gas velocity components are calculated from B_1 with the mean values at A and B_1 . The program then integrates the equations of motion again starting from A to find the corrected particle location B_2 . ALOCAT and POLATE are called again to give the gas velocity components at B_2 , and these velocity components are compared to the corresponding values at B_1 , if the difference is large, the previous iteration is repeated. Once the iteration has converged, the trajectory to point B has been determined and this point is used as the initial point for the next time increment.

The subroutine ALOCAT is used to determine the subscripts of the grid points that surround the particle. Figure 15 shows a typical particle within a set of quasi-orthogonals and streamlines. The subroutine returns the values IP and KP that locate the particle within a particular grid. The subroutine also returns JP, which is the next higher subscript in the XT and THTA arrays. If the particle

is no longer within the boundaries of the flow field, the subroutine returns NOUT which is a code specifying where the particle has gone out of the flow field.

The subroutine POLATE interpolates the value of any variable whose values are known at four grid points surrounding the particle. Referring to Figure 15, the subroutine first calculates the distances D(1) and D(2), and based on these distances, determines a weighted average of the variable at two locations on adjacent streamlines. These values are AA and AB. Then the subroutine determines DD(1) and DD(2) and uses these distances to get the weighted average of AA and AB at the position occupied by the particle.

The subroutine RBCH considers the particles that rebound from the casing or the hub. It is called whenever the particle position B is outside the casing or the hub boundaries. The subroutine returns to the previous position, and linearly extends the trajectory over one time increment, with the bounce occurring at some portion of the time segment. The subroutine writes "BOUNCE OFF SURFACE (NOUT) ..." and prints the location of the bounce.

The subroutine RBBB considers the case where the particle rebounds from the blade surfaces. The procedure is the same as RBCH.

The subroutine RNUMBR is used to determine the drag coefficients based on a curve fit of the drag versus Reynolds Number data. Equations 20 are used, and Figure 3 demonstrates the fit of these equations to data.

The subroutine RUNGE uses a fourth order method to integrate a system of simultaneous first order ordinary differential equations across one time step. Reference 5 explains this subroutine in more detail.

Input

The input cards take the following format. Units consistent with the quasi-orthogonal program as given in Reference 9 must be used.

The first group of input cards are the punched output cards from the quasi-orthogonal program of Reference 9. These cards are punched in the correct format when the code BCDP in the quasi-orthogonal program is set equal to 2. An example of the input data is included with the example case presented after the program listing.

Following these, the data sets corresponding to the particle trajectories are

TITLE	(18A4)
VISREF, TREF, TSUT, DGFC	(E20.5, 3F10.3)
RHOP, DIAP, H, TMAX, ETA-N, ETA-T	(F10.2, 3E10.4, 2F10.3)
YR(I), I = 1, 6	(6F10.3)
NSTEP	(I5)

Multiple sets of this group may be stacked together for cases of more than one particle. The input variables are defined below.

TITLE - The first card is reproduced at the top of the first page of output for each particle. Any statement in columns 2 through 72 will be reproduced.

VISREF - Reference viscosity corresponding to TREF. Used in Sutherland's Law. (lbm/ft sec).

TREF - Reference temperature corresponding to VISREF. Used in Sutherland's Law. ($^{\circ}$ R).

TSUT - Constant used in Sutherland's Law. (198.6° R).

DGFC - Drag factor. The drag coefficient based on Reynold's Number is multiplied by DGFC. Except in very special cases, this should be 1.0.

RHOP - Particle density. (lbm/ft³).

DIAP - Particle diameter. (Ft).

H - Time increment used in the integration process. (Sec).

TMAX - Program stops if time exceeds TMAX. (Sec).

ETA-N - Normal restitution coefficients.

ETA-T - Tangential restitution coefficients.

YR(1) - Initial radial position of the particle. (Ft).

YR(2) - Initial radial velocity component of the particle. (Ft/Sec).

- YR(3) - Initial angular position of the particle. (Radians).
- YR(4) - Initial angular velocity of the particle. (Sec^{-1}).
- YR(5) - Initial axial velocity of the particle. (Ft).
- YR(6) - Initial axial velocity of the particle. (Ft/Sec).
- NSTEP - The program prints out trajectory information every NSTEP time increments.

Output

An example listing of typical output is included after the program listing. This program output can be divided into several groups.

Output Group A.

This set of output is a reprint of some of the data that is transferred from the quasi-orthogonal solution.

Output Group B.

This set of output is concerned with the particle trajectory. The first part is an echo check of the data cards corresponding to a particle. Such data checks are useful in correcting key punch mistakes on the input cards.

After initializing the variables, the program calculates and prints several similarity parameters that are useful in relating this particle to other particles having similar trajectories. The quantities that are printed are explained below.

- STEP - A count of each of the integration steps.
- T - Time. (Sec).
- YR(1) ... YR(6) - Position and velocity components of the particle with respect to the rotor.
- ABS - The angular position of the particle with respect to the absolute reference frame.
- REYNOLD - Reynolds Number of the particle at this location.

When the particle leaves either the exit or inlet of the flow field, the program prints the last data information and includes the last value of STEP and M. The last M is the count of the number of iterations required for the solution of particle location to converge. If M is greater than 100, the program truncates prematurely.

C
C
C
C
C

USE OF ARBITRARY QUASI-ORTHOGONALS FOR CALCULATING FLOW
DISTRIBUTION IN THE MERIDIONAL PLANE OF A TURBOMACHINE.
THEODORE KATSANIS NASA TECHNICAL NOTE D-2546 DEC. 1964

COMMON SRW

```

DIMENSION AL(21,21),BETA(21,21),CAL(21,21),CBETA(21,21),
1CURV(21,21),DN(21,21),PRS(21,21),R(21,21),Z(21,21),SM(21,21),
2SA(21,21),SB(21,21),SC(21,21),SD(21,21),SAL(21,21),SBETA(21,21),
3TN(21,21),TT(21,21),WA(21,21),WTR(21,21)
DIMENSION AB(21),AC(21),AD(21),BA(21),DELBT(21),DRDM(21),
1DTR(21),DTRZ(21),DWRDM(21),DWRDM(21),RH(21),RS(21),ZH(21),ZS(21),
2THTA(21),WTF(21),XR(21),XT(21),XZ(21)
INTEGER RUNO,TYPE,BCDP,SRW

```

RUNO = 0

10 READ(5,1010) MX,KMX,MR,MZ,W,WT,XN,GAM,AR

ITNO = 1

RUNO = RUNO + 1

WRITE(6,1020) RUNO

WRITE(6,1010) MX,KMX,MR,MZ,W,WT,XN,GAM,AR

READ(5,1010) TYPE,BCDP,SRW,NULL,TEMP,ALM,RHO,TOLER,PLOSS

WRITE(6,1010) TYPE,BCDP,SRW,NULL,TEMP,ALM,RHO,TOLER,PLOSS

READ(5,1010) MHTA,NPRT,ITER,NULL,SFACT,ZSPLIT,BETIN,RB,CORFAC

WRITE(6,1010) MHTA,NPRT,ITER,NULL,SFACT,ZSPLIT,BETIN,RB,CORFAC

READ(5,1011) WTOLER

WRITE(6,1011) WTOLER

READ(5,1030) (ZS(I),I=1,MX)

WRITE(6,1030) (ZS(I),I=1,MX)

READ(5,1030) (ZH(I),I=1,MX)

WRITE(6,1030) (ZH(I),I=1,MX)

READ(5,1030) (RS(I),I=1,MX)

WRITE(6,1030) (RS(I),I=1,MX)

READ(5,1030) (RH(I),I=1,MX)

WRITE(6,1030) (RH(I),I=1,MX)

DO 20 I = 1,MX

ZS(I)=ZS(I)/12.

ZH(I)=ZH(I)/12.

RS(I)=RS(I)/12.

20 RH(I)=RH(I)/12.

IF(TYPE.NE.0) GO TO 40

WA(1,1)=WT/RHO/(ZS(1)-ZH(1))/3.14/(RS(1)+RH(1))

DO 30 I = 1,MX

DN(I,KMX)=SQRT((ZS(I)-ZH(I))**2+(RS(I)-RH(I))**2)

DO 30 K = 1,KMX

DN(I,K) = FLOAT(K-1)/FLOAT(KMX-1)*DN(I,KMX)

WA(I,K) = WA(1,1)

Z(I,K)=DN(I,K)/DN(I,KMX)*(ZS(I)-ZH(I))+ZH(I)

30 R(I,K)=DN(I,K)/DN(I,KMX)*(RS(I)-RH(I))+RH(I)

GO TO 50

40 IF(TYPE.NE.1) GO TO 145

DO 45 I=1,MX

READ(5,7010) (DN(I,K),K=1,KMX)

READ(5,7010) (WA(I,K),K=1,KMX)

```

READ(5,7010) ( Z(I,K),K=1,KMX)
READ(5,7010) ( R(I,K),K=1,KMX)
45 CONTINUE
WRITE(6,1040)
50 READ (5,1030)(THTA(I),I=1,MTHTA)
WRITE (6,1030)(THTA(I),I=1,MTHTA)
READ (5,1030)(XT(I),I=1,MTHTA)
WRITE (6,1030)(XT(I),I=1,MTHTA)
DO 60 K=1,MR
READ (5,1030)(TN(I,K),I=1,MZ)
60 WRITE (6,1030)(TN(I,K),I=1,MZ)
READ (5,1030)(XZ(I),I=1,MZ)
WRITE (6,1030)(XZ(I),I=1,MZ)
READ (5,1030)(XR(I),I=1,MR)
WRITE (6,1030)(XR(I),I=1,MR)

```

```

C
C END OF INPUT STATEMENTS.
C
C

```

```

C SCALING - CHANGE INCHES TO FEET AND PSI TO LB/SQ.FT,
C INITIALIZE, CALCULATE CONSTANTS.
C

```

```

70 DO 90 K = 1,MR
DO 80 I = 1,MZ
80 TN(I,K) = TN(I,K)/12.0
90 XR(K) = XR(K)/12.0
DO 100 I = 1,MZ
100 XZ(I) = XZ(I)/12.0
DO 110 K = 1,KMX
110 SM(1,K) = 0.0
BA(1) = 0.0
DO 120 K = 2,KMX
120 BA(K)=FLOAT(K-1)*WT/FLOAT(KMX-1)
DO 130 I = 1,MX
130 DN(I,1) = 0.0
DO 140 I = 1,MTHTA
140 XT(I) = XT(I)/12.0
ROOT=SQRT(2.0)
145 CONTINUE
DO 146 I=1,MX
DO 146 K=1,KMX
146 WTR(I,K)=0.0
TOLER = TOLER/12.0
RB = RB/12.0
ZSPLIT = ZSPLIT/12.0
PLOSS = PLOSS*144.0
CI = SQRT(GAM*AR*TEMP)
WRITE (6,1050) CI
KMXM1 = KMX-1
CP=AR*GAM/(GAM-1.)
EXPON = 1.0/(GAM-1.0)
BETIN = -BETIN/57.29577
RINLET = (RS(1)+RH(1))/2.0

```

```

CEF = SIN(BETIN)/COS(BETIN)/RINLET/(RINLET-RB)**2
ERROR = 10000.0

```

```

C
C BEGINNING OF LOOP FOR ITERATIONS.

```

```

C
150 IF(ITER.EQ.0) WRITE(6,1060) ITNO
    IF(ITER.EQ.0) WRITE(6,1070)
    ERROR1 = ERROR
    ERROR = 0.0

```

```

C
C START OF CALCULATIONS OF PARAMETERS.

```

```

C
DO 230 K = 1,KMX
DO 160 I = 1,MX
AB(I) = (Z(I,K)-R(I,K))/ROOT
160 AC(I) = (Z(I,K)+R(I,K))/ROOT
CALL SPLINE(AB,AC,MX,AL(1,K),CURV(1,K))
DO 170 I = 1,MX
CURV(I,K) = CURV(I,K)/(1.+AL(I,K)**2)**1.5
AL(I,K) = ATAN(AL(I,K))-0.785398
CAL(I,K) = COS(AL(I,K))
170 SAL(I,K) = SIN(AL(I,K))
DO 180 I = 2,MX
180 SM(I,K)=SM(I-1,K)+SQRT((Z(I,K)-Z(I-1,K))**2+(R(I,K)-R(I-1,K))**2)
190 CALL SPLDER(XT(1),THTA(1),MTHTA,Z(1,K),MX,DTDZ(1))
DO 220 I = 1,MX
CALL LININT(Z(I,K),R(I,K),XZ,XR,TN,21,21,T)
IF(R(I,K).LE.RB) GO TO 200
DTDR(I)=CEF*(R(I,K)-RB)**2
GO TO 210
200 DTDR(I) = 0.0
210 TQ = R(I,K)*DTDR(I)
    TP = R(I,K)*DTDZ(I)
    TT(I,K) = T*SQRT(1.+TP*TP)
    BETA(I,K) = ATAN(TP*CAL(I,K)+TQ*SAL(I,K))
    SBETA(I,K)=SIN(BETA(I,K))
    CBETA(I,K)=COS(BETA(I,K))
    SA(I,K)=CBETA(I,K)**2*CAL(I,K)*CURV(I,K)-SBETA(I,K)**2/F(I,K)
1+SA(I,K)*CBETA(I,K)*SBETA(I,K)*DTDR(I)
    SC(I,K)=-SAL(I,K)*CBETA(I,K)**2*CURV(I,K)+SAL(I,K)*CBETA(I,K)
1*SBETA(I,K)*DTDZ(I)
    AB(I)=WA(I,K)*CBETA(I,K)
220 AC(I)=WA(I,K)*SBETA(I,K)
CALL SPLINE(SM(1,K),AB,MX,DWMDM,AD)
CALL SPLINE(SM(1,K),AC,MX,DWTDN,AD)
IF((ITER.LE.0).AND.(MOD(K-1,NPRT).EQ.0)) WRITE(6,1080) K
DO 230 I = 1,MX
SB(I,K)=SAL(I,K)*CBETA(I,K)*DWMDM(I)-2.*W*SBETA(I,K)+DTDR(I)
1*R(I,K)*CBETA(I,K)*(DWTDM(I)+2.*W*SAL(I,K))
SD(I,K)=CAL(I,K)*CBETA(I,K)*DWMDM(I)+DTDZ(I)*R(I,K)*CBETA(I,K)
1*(DWTDM(I)+2.*W*SAL(I,K))
IF((ITER.GT.0).OR.(MOD(K-1,NPRT).NE.0)) GO TO 230
A = AL(I,K)*57.29577

```

B = SM(I,K)*12.

E = TT(I,K)*12.

G = BETA(I,K)*57.29577

WRITE(6,1090) A,CURV(I,K),B,G,E,SA(I,K),SB(I,K),SC(I,K),SD(I,K)

230 CONTINUE

C
C END OF LOOP - PARAMETER CALCULATION.

C
C CALCULATE BLADE SURFACE VELOCITIES. (AFTER CONVERGENCE.)

IF(ITER.NE.0) GO TO 260

DO 250 K = 1,KMX

CALL SPLINE(SM(1,K),TT(1,K),MX,DELBTA,AC)

A=XN

DO 240 I = 1,MX

240 AB(I)=(R(I,K)*W+WA(I,K)*SBETA(I,K))*(6.283186*R(I,K)/A-TT(I,K))

CALL SPLINE(SM(1,K),AB,MX,DRDM,AC)

IF(SFACT.LE.1.0) GO TO 245

A = SFACT*XN

DO 244 I = 1,MX

244 AB(I)=(R(I,K)*W+WA(I,K)*SBETA(I,K))*(6.283186*R(I,K)/A-TT(I,K))

CALL SPLINE(SM(1,K),AB,MX,AD,AC)

245 DO 250 I = 1,MX

BETAD=BETA(I,K)-DELBTA(I)/2.

BETAT = BETAD+DELBTA(I)

COSBD = COS(BETAD)

COSBT = COS(BETAT)

IF(Z(I,K).LT.ZSPLIT) DRDM(I) = AD(I)

WTR(I,K)=COSBD*COSBT/(COSBD+COSBT)*(2.*WA(I,K)/COSBD+R(I,K)*W
1*(BETAD-BETAT)/CBETA(I,K)**2+DRDM(I))

250 CONTINUE

C
C END OF BLADE SURFACE VELOCITY CALCULATIONS.

C
C START CALCULATION OF WEIGHT FLOW VS. DISTANCE FROM HUB.

260 DO 370 I = 1,MX

IND = 1

DO 270 K = 1,KMX

270 AC(K)=DN(I,K)

GO TO 290

280 WA(I,1)=0.5*WA(I,1)

290 DO 300 K = 2,KMX

J = K-1

HR = R(I,K)-R(I,J)

HZ = Z(I,K)-Z(I,J)

WAS=WA(I,J)*(1.+SA(I,J)*HR+SC(I,J)*HZ)+SB(I,J)*HR+SD(I,J)*HZ

WASS=WA(I,J)+WAS*(SA(I,K)*HR+SC(I,K)*HZ)+SB(I,K)*HR+SD(I,K)*HZ

300 WA(I,K)=(WAS+WASS)/2.

310 DO 340 K = 1,KMX

TIP=1.-(WA(I,K)**2+2.*W*ALM-(W*R(I,K))**2)/2./CP/TEMP

```

IF(TIP.LT.0.0) GO TO 280
TPP1P=1.-(2.*W*ALM-(W*R(I,K))**2)/2./CP/TEMP
DENSTY=TIP**EXPON*RHO-(TIP/TPP1P)**EXPON*PLOSS/AR/TPP1P/TEMP
I*32.17*SM(I,K)/SM(MX,K)
PRS(I,K)=DENSTY*AR*TIP*TEMP/32.17/144.
IF(ZS(I).LE.ZH(I)) GO TO 320
PSI=ATAN((RS(I)-RH(I))/(ZS(I)-ZH(I)))-1.5708
GO TO 330
320 PSI=ATAN((ZH(I)-ZS(I))/(RS(I)-RH(I)))
330 WTHRU = WA(I,K)*CBETA(I,K)*COS(PHI-AL(I,K))
A = XN
IF(Z(I,K).LT.ZSPLIT) A = SFACT*XN
C=6.283186*R(I,K)-A*TT(I,K)
340 AD(K)=DENSTY*WTHRU*C
CALL INTGRL(AC(1),AD(1),KMX,WTFL(1))
IF(ABS(WT-WTFL(KMX)).LE.WTOLER) GO TO 350
CALL CONTIN(WA(I,1),WTFL(KMX),IND,I,WT)
IF(IND.NE.6) GO TO 290
350 CALL SPLINT(WTFL,AC,KMX,BA,KMX,AB)
DO 360 K = 1,KMX
DELTA=ABS(AB(K)-DN(I,K))
DN(I,K)=(1.-CORFAC)*DN(I,K)+CORFAC*AB(K)
360 IF(DELTA.GT.ERROR) ERROR = DELTA
370 CONTINUE

```

C
C END OF LOOP - WEIGHT FLOW CALCULATION.

C
C
C CALCULATE STREAMLINE COORDINATES FOR NEXT ITERATION.

C

```

DO 380 K = 2,KMXM1
DO 380 I = 1,MX
Z(I,K)=DN(I,K)/DN(I,KMX)*(ZS(I)-ZH(I))+ZH(I)
380 R(I,K)=DN(I,K)/DN(I,KMX)*(RS(I)-RH(I))+RH(I)
IF((ERROR.GE.ERROR1).OR.(ERROR.LE.TOLER)) ITER=ITER-1
IF(ITER.GT.0) GO TO 410
WRITE(6,1100)
DO 400 K = 1,KMX,NPRT
WRITE (6,1080) K
DO 390 I = 1,MX
AB(I) = (Z(I,K)-R(I,K))/ROOT
390 AC(I) = (Z(I,K)+R(I,K))/ROOT
CALL SPLINE(AB,AC,MX,AL(1,K),CURV(1,K))
DO 400 I = 1,MX
CURV(I,K)=CURV(I,K)/(1.+AL(I,K)**2)**1.5
A=DN(I,K)*12.
B=Z(I,K)*12.
D=R(I,K)*12.
400 WRITE (6,1110) A,B,D,WA(I,K),PRS(I,K),WTR(I,K),CURV(I,K)
WRITE (6,1130)
410 A =ERROR*12.
WRITE (6,1120) ITNO,A
ITNO = ITNO+1

```



```

IF(ITER.GE.0) GO TO 150
IF(BCDP.NE.1) GO TO 415
DO 414 I=1,MX
PUNCH 7010, (DN(I,K),K=1,KMX)
PUNCH 7010, (WA(I,K),K=1,KMX)
PUNCH 7010, ( Z(I,K),K=1,KMX)
PUNCH 7010, ( R(I,K),K=1,KMX)

```

```
414 CONTINUE
```

```
415 IF(BCDP.NE.2) GO TO 10
```

```

C
C
C
C
C

```

```

CARD SET FOR READING AND WRITING BETWEEN QUASI-ORTHOGONAL PROGRAM
AND TRAJECTORIES PROGRAM. WRITE SET.

```

```

PUNCH 7000, MX,KMX,MHTA
PUNCH 7010, GAM,TEMP,RHO,AR,SFACT
PUNCH 7010, ZSPLIT,W,XN,ALM,PLOSS
DO 7777 I=1,MX
PUNCH 7010, ( R(I,K),K=1,KMX)
PUNCH 7010, ( Z(I,K),K=1,KMX)
PUNCH 7010, ( WA(I,K),K=1,KMX)
DO 777 K=1,KMX

```

```
777 AL(I,K)=ATAN(AL(I,K))-0.7853982
```

```
PUNCH 7010, ( AL(I,K),K=1,KMX)
```

```
PUNCH 7010, (BETA(I,K),K=1,KMX)
```

```
PUNCH 7010, ( SM(I,K),K=1,KMX)
```

```
7777 CONTINUE
```

```
PUNCH 7010, (THTA(I),I=1,MHTA)
```

```
PUNCH 7010, ( XT(I),I=1,MHTA)
```

```
7000 FORMAT(7I10)
```

```
7010 FORMAT(5E14.6)
```

```

C
C
C
C
C
C
C

```

```
420 GO TO 10
```

```
END OF CALCULATION OF NEXT STREAMLINE COORDINATES.
```

```
FORMAT STATEMENTS.
```

```
1010 FORMAT(4I5,5F10.4)
```

```
1011 FORMAT(F10.5)
```

```
1020 FORMAT(1H1,7HRUN NO.,I3,10X,23HINPUT DATA CARD LISTING)
```

```
1030 FORMAT(7F10.4)
```

```
1040 FORMAT(1H ,10X,25H BCD CARDS FOR DN,WA,Z,R.)
```

```
1050 FORMAT(1H1,4X,31HSTAG. SPEED OF SOUND AT INLET =,F9.2)
```

```
1060 FORMAT(1H0,/,5X,13HITERATION NO.,I3)
```

```
1070 FORMAT(1H ,6X,2HAL,12X,2HRC,12X,2HSM,12X,4HBETA,10X,2HTT,12X,2HSA,
1 12X,2HSB,12X,2HSC,12X,2HSD)
```

```
1080 FORMAT(1H ,2X,10HSTREAMLINE,I3)
```

```
1090 FORMAT(9F14.6)
```

```
1100 FORMAT(1H1,9X,2HDN,18X,1HZ,19X,1HR,19X,2HWA,18X,3HPRS,16X,3HWTR,
1 14X,2HRC)
```

```
1110 FORMAT(1H ,6F19.6,F18.6)
```

```
1120 FORMAT(1H ,4X,13HITERATION NO.,I3,10X,24HMAX. STREAMLINE CHANGE =,
```

```
1 F10.6)
```

```
1130 FORMAT(1H1)
```

```

C
C END OF FORMAT STATEMENTS.

```

```
END
```

INTEGFR RUNGE

DIMENSION R(21,21),Z(21,21),WA(21,21),AL(21,21),RFTA(21,21),

1 TT(21,21),THTA(21),XT(21)

— DIMENSION STATE(18),YR(6),YRS(6),WR(4),WU(4),WZ(4),PHOA(4),

1 TEMPA(4),VISTAR(4),FTA(2),DD(2),AP(2),AZ(2)

DIMENSION SM(21,21),FR(6),RA(2),ZA(2),THETA(2)

C

C READ QUASI-ORTHOGONAL RESULTS - WRITE SIGNIFICANT PARTS.

C

CALL BAREAD(MX,KMX,MTHTA)

WRITE(6,4040) MX,KMX,MTHTA

CALL BDRFAD(GAMMA,TEMP,RHO)

CALL BDRFAD(RGAS,SFACT,ZSPLIT)

CALL BDRFAD(W,XN,ALM)

CALL BDRFAD(PLOSS,ANULL,BNULL)

WRITE(6,4050) GAMMA,TEMP,RHO,RGAS,SFACT,ZSPLIT,W

WRITE(6,4090) XN,ALM,PLOSS

CALL BCREAD(R)

CALL BCREAD(Z)

CALL BCREAD(WA)

CALL BCREAD(AL)

CALL BCREAD(RFTA)

CALL BCREAD(SM)

CALL BBREAD(THTA)

CALL BBRFAD(XT)

WRITE(6,4070)

DO 10 I=1,MX

10 WRITE(6,4060) R(I,1),Z(I,1),R(I,KMX),Z(I,KMX)

WRITE(6,5050) (XT(I),I=1,MTHTA)

WRITE(6,5040) (THTA(I),I=1,MTHTA)

C

C READ PARTICLE DATA

C

605 READ(5,3000) (STATE(I),I=1,18)

WRITE(6,4000) (STATE(I),I=1,18)

READ(5,3010) VISPRF,TREF,TSUT,DGFC

WRITE(6,4010) VISPRF,TREF,TSUT,DGFC

READ(5,3020) RHOP,DIAP,H,TMAX,ETA(1),ETA(2)

WRITE(6,4020) RHOP,DIAP,H,TMAX,ETA(1),ETA(2)

READ(5,3030) (YR(I),I=1,6)

WRITE(6,4030) (YR(I),I=1,6)

READ(5,5010) NSTEP

WRITE(6,5020) NSTEP

C

C INITIALIZE.

C

T=0.0

L=0

WRITE(6,5000) L,T,(YR(I),I=1,6)

FXPON=1.0/(GAMMA-1.0)

CP=GAMMA*RGAS/(GAMMA-1.0)

RPART=DIAP/2.0

N=6

```

M=1
NTIME=NSTEP
TS=T
C
C DETERMINE AIR VELOCITIES AND PROPERTIES AT PARTICLE LOCATION.
C SET UP TO START TRACE, INITIALIZE.
C
CALL ALCCAT(R,Z,MX,KMX,YR,THTA,XT,MHTA,XN,ZSPLIT,IP,JP,KP,NOUT)
IF(NOUT.EQ.0) GO TO 606
WRITE(6,4080)
GO TO 605
C
606 CALL POLATE(R,Z,WA,YR(1),YR(5),IP,KP,WAP,DD)
CALL POLATE(R,Z,BETA,YR(1),YR(5),IP,KP,BETAP,DD)
CALL POLATE(R,Z,AL,YR(1),YR(5),IP,KP,ALPP,DD)
CALL POLATE(R,Z,SM,YR(1),YR(5),IP,KP,SMP,DD)
WR(1)=WAP*COS(BETAP)*SIN(ALPP)
WU(1)=WAP*SIN(BETAP)
WZ(1)=WAP*COS(BETAP)*COS(ALPP)
SMT=(DD(1)*SM(MX,KP)+DD(2)*SM(MX,KP-1))/(DD(1)+DD(2))
TIP=1.0-(WAP**2+2.*W*ALM-(W*YR(1))**2)/2.0/CP/TEMP
TPPIP=1.0-(2.*W*ALM-(W*YR(1))**2)/2.0/CP/TEMP
RHOA(1)=TIP**EXPON*RHO-(TIP/TPPIP)**EXPON*PLCSS/RGAS/TPPIP/TEMP
L *32.17*SMP/SMT
TEMPA(1)=TIP*TEMP
VISTAR(1)=VISRFF*((TEMPA(1)/TRFF)**1.5)*(TRFF+TSUT)/(TEMPA(1)
+TSUT)
C
C INITIALIZE FOR FIRST STEP.
C
DO 610 I=1,4
WR(I)=WR(1)
WU(I)=WU(1)
WZ(I)=WZ(1)
RHOA(I)=RHOA(1)
TEMPA(I)=TEMPA(1)
610 VISTAR(I)=VISTAR(1)
DO 620 I=1,N
620 YRS(I)=YR(I)
C
C INTEGRATE USING RUNGE-KUTTA METHOD
C
625 VDIFF=SQRT((WP(2)-YR(2))**2+(WU(2)-YR(1)*YR(4))**2
+ (WZ(2)-YR(6))**2)
RENOLD=PHOA(2)*VDIFF*DIAP/VISTAR(2)
CALL RNUMPR(RENOLD,CGFC,CD)
RCOIN=RHOA(2)*CD/PHOP/RPART/2.3333
630 IF(RUNGE(N,YR,FR,T,H).NE.1) GO TO 640
FR(1)=YR(2)
FR(2)=YR(1)*YR(4)**2+2.0*YR(1)*W*YR(4)+YR(1)*W**2
+RCOIN*VDIFF*(WR(2)-YR(2))
FR(3)=YR(4)
FR(4)=-2.0*YR(2)*YR(4)/YR(1)-2.0*W*YR(2)/YR(1)

```

C-2

```

1  +BCON*VDIFF*(WU(2)-YR(1)*YR(4))/YP(1)
FR(5)=YR(6)
FR(6)=BCON*VDIFF*(WZ(2)-YR(6))
GO TO 630
640 CONTINUE
C
C DETERMINE IF WALL INTERACTION OCCURRED.
C
CALL ALCCAT(R,Z,MX,KMX,YR,THTA,XT,MTHTA,XN,ZSPLIT,IP,JP,KP,NOUT)
IF(NOUT.EQ.0) GO TO 700
IF(NOUT.EQ.1) GO TO 780
IF(NOUT.EQ.3) GO TO 780
DD 645 I=1,6
645 YR(I)=YRS(I)
T=TS
IF(NOUT.NE.2) GO TO 650
RA(1)=R(IP,1)
RA(2)=R(IP-1,1)
ZA(1)=Z(IP,1)
ZA(2)=Z(IP-1,1)
CALL RBCH(YR,RA,ZA,T,H,ETA,NCUT)
GO TO 750
650 IF(NOUT.NE.4) GO TO 660
RA(1)=R(IP,KMX)
RA(2)=R(IP-1,KMX)
ZA(1)=Z(IP,KMX)
ZA(2)=Z(IP-1,KMX)
CALL RBCH(YR,RA,ZA,T,H,ETA,NOUT)
GO TO 750
660 THETA(1)=THTA(JP)
THETA(2)=THTA(JP-1)
ZA(1)=XT(JP)
ZA(2)=XT(JP-1)
IF(YP(5).GE.ZSPLIT) DTHET=3.1415927/XN
IF(YP(5).LT.ZSPLIT) DTHET=1.5707963/XN
CALL PBBH(YR,THETA,ZA,T,H,ETA,NOUT,DTHET)
GO TO 750
C
C DETERMINE AIR VALUES AT NEW LOCATION
C
700 CALL POLATE(R,Z,WA ,YR(1),YR(5),IP,KP,WAP ,DD)
CALL PCLATE(R,Z,BETA ,YR(1),YR(5),IP,KP,BETAP ,DD)
CALL POLATE(R,Z,AL ,YR(1),YR(5),IP,KP,ALPP ,DD)
CALL PCLATE(R,Z,SM ,YR(1),YR(5),IP,KP,SMP ,DD)
WR(4)=WAP*COS(BETAP)*SIN(ALPP)
WU(4)=WAP*SIN(BETAP)
WZ(4)=WAP*COS(BETAP)*COS(ALPP)
SMT=(DD(1)*SM(MX,KP)+DD(2)*SM(MX,KP-1))/(DD(1)+DD(2))
TIP=1.0-(WAP**2+2.*W*ALM-(W*YR(1))**2)/2.0/CP/TEMP
TPPIP=1.0-(2.*W*ALM-(W*YR(1))**2)/2.0/CP/TEMP
RHOA(4)=TIP**FXPON*RHO-(TIP/TPPIP)**FXPON*PLESS/RGAS/TPPIP/TEMP
1 *32.17*SMP/SMT
TEMPA(4)=TIP*TEMP

```

```
VISTAR(4)=VISPEF*((TFMPA(4)/TRFF)**1.5)*(TRFF+TSUT)/(TFMPA(4)
1 +TSUT)
```

C
C
C
C

```
TEST AIR VALUES USED, IF INCORRECT, RESET INTEGRATION VALUES AND GO
625. IF CORRECT, GO TO 750.
```

```
IF((ABS(WR(4)-WR(3)).LT.1.0E-4).AND.(ABS(WU(4)-WU(3)).LT.1.0E-4)
1 .AND.(ABS(WZ(4)-WZ(3)).LT.1.0E-4)) GO TO 750
```

```
WR(2)=(WR(4)+WR(1))/2.0
```

```
WU(2)=(WU(4)+WU(1))/2.0
```

```
WZ(2)=(WZ(4)+WZ(1))/2.0
```

```
WR(3)=WR(4)
```

```
WU(3)=WU(4)
```

```
WZ(3)=WZ(4)
```

```
RHOA(2)=(RHOA(4)+RHOA(1))/2.0
```

```
TFMPA(2)=(TFMPA(4)+TFMPA(1))/2.0
```

```
VISTAR(2)=(VISTAR(4)+VISTAR(1))/2.0
```

```
VISTAR(2)=(VISTAR(4)+VISTAR(1))/2.0
```

```
T=TS
```

```
DO 710 I=1,N
```

```
710 Y(I)=YPS(I)
```

```
IF(M.GT.100) GO TO 800
```

```
M=M+1
```

```
GO TO 625
```

C
C
C

```
COMPLETE INTEGRATION STEP. IF REQUIRED, WRITE OUTPUT. GO TO 625
```

```
750 M=1
```

```
L=L+1
```

```
TS=T
```

```
DO 760 I=1,N
```

```
760 YPS(I)=YP(I)
```

```
WP(2)=1.5*WR(4)-0.5*WR(1)
```

```
WR(3)=2.0*WR(4)-WR(1)
```

```
WP(1)=WR(4)
```

```
WR(4)=WR(3)
```

```
WU(2)=1.5*WU(4)-0.5*WU(1)
```

```
WU(3)=2.0*WU(4)-WU(1)
```

```
WU(1)=WU(4)
```

```
WU(4)=WU(3)
```

```
WZ(2)=1.5*WZ(4)-0.5*WZ(1)
```

```
WZ(3)=2.0*WZ(4)-WZ(1)
```

```
WZ(1)=WZ(4)
```

```
WZ(4)=WZ(3)
```

```
RHOA(2)=1.5*RHOA(4)-0.5*RHOA(1)
```

```
RHOA(3)=2.0*RHOA(4)-RHOA(1)
```

```
RHOA(1)=RHOA(4)
```

```
RHOA(4)=RHOA(3)
```

```
TFMPA(2)=1.5*TFMPA(4)-0.5*TFMPA(1)
```

```
TFMPA(3)=2.0*TFMPA(4)-TFMPA(1)
```

```
TFMPA(1)=TFMPA(4)
```

```
TFMPA(4)=TFMPA(3)
```

```
VISTAR(2)=1.5*VISTAR(4)-0.5*VISTAR(1)
```

```

VISTAR(3)=2.0*VISTAR(4)-VISTAR(1)
VISTAR(1)=VISTAR(4)
VISTAR(4)=VISTAR(3)
NTIME=NTIME-1
780 IF((NOUT.EQ.1).OR.(NOUT.EQ.3).OR.(T.GT.TMAX)) GO TO 800
IF(NTIME.GT.0) GO TO 625
WRITE(6,2050) L,T,(YR(I),I=1,6),RENOLD
NTIME=NSTEP-1
IF(L.GT.1) NTIME=NSTEP
GO TO 625
800 CONTINUE
WRITE(6,2050) L,T,(YR(I),I=1,6),RENOLD
WRITE(6,2060) M,L
805 CONTINUE
GO TO 605
2050 FORMAT(1H ,I10,F15.6,7F15.5)
2060 FORMAT(1H0,2I10)
3000 FORMAT(18A4)
3010 FORMAT(F20.4,3F10.4)
3020 FORMAT(F10.4,F10.3,F10.3,3F10.4)
3030 FORMAT(6F10.5)
4000 FORMAT(1H1,18A4)
4010 FORMAT(1H0,13X,6HVISPEF,11X,4HTRFF,11X,4HTSOI,11X,4HDFGC,/,
1 (F20.4,3F15.4))
4020 FORMAT(1H0,10X,4HFMOP,11X,4HDIAP,14X,1HH,11X,4HTMAX,10X,5HETA-N,
1 10X,5HETA-T,/, (F15.4,2F15.3,3F15.4))
4030 FORMAT(1H0, 9X,5HYR(1),10X,5HYR(2),10X,5HYR(3),10X,5HYR(4),10X,
1 5HYR(5),10X,5HYR(6),/, (6F15.5))
4040 FORMAT(1H1,44HIMPORTANT DATA FROM QUASI-ORTHOGONAL PROGRAM,/,
1 8X,2HMX,7X,3HKMX,5X,5HMTHTA,/, (3I10))
4050 FORMAT(1H0, 9X,5HGAMMA,11X,4HTEMP,12X,3HPHO,11X,4HFGAS,10X,
1 5HSFACT, 9X,6HZSPLIT,14X,1HW,/, (7F15.4))
4060 FORMAT(1H ,2F15.6,5X,2F15.6)
4070 FORMAT(1H0,15HHUR COORDINATES,27X,15HTIP COORDINATES,/,7X,1HR,
1 14X,1HZ,19X,1HR,14X,1HZ)
4080 FORMAT(46HCPARTICLE CUT OF PASSAGE AT FIRST POINT GIVEN.)
4090 FORMAT(1H0,12X,2HXN,12X,3HALM,10X,5HPLOSS,/,
1 (3F15.4))
5000 FORMAT(1H0,5X,4HSTEP,14X,1HT,10X,5HYR(1),10X,5HYR(2),10X,5HYR(3),
1 10X,5HYR(4),10X,5HYR(5),10X,5HYR(6),9X,6HRENOLD,
1 /,(I11,E15.6,6F15.5))
5010 FORMAT(I5)
5020 FORMAT(17HOPPRINT DATA EVERY,17,2X,7HSTEP(S))
5040 FORMAT(12HOTHETA ARRAY,/, (6F15.5))
5050 FORMAT(9HOXT ARRAY,/, (6F15.5))
END

```

```

SUBROUTINE ALUCAT(R,Z,MX,KMX,YR,THTA,XT,MTA,XN,ZST,IP,JP,KP,NOUT
DIMENSION R(21,21),Z(21,21),PC(21),ZC(21),YR(6),THTA(21),XT(21)
MHTA=MTA
ZSPLIT=ZST
NOUT=0
DO 20 I=1,MX
IF(ABS(Z(I,1)-Z(I,KMX)).LT.1.0E-12) GO TO 10
A=(R(I,1)-R(I,KMX))/(Z(I,1)-Z(I,KMX))
R=R(I,1)-A*Z(I,1)
RTEST=A*YR(5)+R
IF(RTEST.LE.YR(1)) GO TO 30
GO TO 20
10 IF(Z(I,1).GE.YR(5)) GO TO 30
20 CONTINUE
30 IP=I
IF(IP.NE.1) GO TO 40
NOUT=1
RETURN
40 IF(IP.NE.MX) GO TO 50
IF(Z(MX,1).GT.YR(5)) GO TO 50
NOUT=3
RETURN
50 CONTINUE
DO 70 K=1,KMX
IF(ABS(Z(IP,K)-Z(IP-1,K)).LT.1.0E-12) GO TO 60
A=(R(IP-1,K)-R(IP,K))/(Z(IP-1,K)-Z(IP,K))
B=R(IP,K)-A*Z(IP,K)
RTEST=A*YR(5)+B
IF(YR(1).LE.RTEST) GO TO 80
GO TO 70
60 IF(Z(IP,K).GE.YR(5)) GO TO 80
70 CONTINUE
80 KP=K
IF(KP.NE.1) GO TO 90
NOUT=2
RETURN
90 IF(KP.NE.KMX) GO TO 120
IF(ABS(Z(IP,KP)-Z(IP-1,KP)).LT.1.0E-12) GO TO 110
IF(YR(1).LT.RTEST) GO TO 120
NOUT=4
RETURN
110 IF(YR(5).LE.Z(IP,KP)) NOUT=4
RETURN
120 CONTINUE
DO 130 J=1,MHTA
IF(XT(J).GE.YR(5)) GO TO 140
130 CONTINUE
140 JP=J
A=(THTA(JP-1)-THTA(JP))/(XT(JP-1)-XT(JP))
R=THTA(JP)-A*XT(JP)
THTA=YR(5)+A*R
IF(YR(5).GE.ZSPLIT) DTHTA=3.1415927/XN
IF(YR(5).LT.ZSPLIT) DTHTA=1.5707963/XN

```

```

THET1=DTHTYA+THTR
THET2=-DTHTA+THTR
IF(YR(3).LT.THET1) GO TO 150
NDUT=5
RETURN
150 IF(YR(3).GT.THET2) GO TO 160
NDUT=6
160 RETURN
END

```

```

SUBROUTINE POLATE(R,Z,A,RP,ZP,IP,KP,AP,DD)
DIMENSION P(21,21),Z(21,21),A(21,21),D(2),DD(2)
DO 10 I=1,2
IA=IP+I-1
IF(ABS(Z(IA,KP-1))-Z(IA,KP)).LT.1.0E-12) GO TO 5
IF(ABS(R(IA,KP)-R(IA,KP-1)).LT.1.0E-12) GO TO 6
AM=(P(IA,KP)-R(IA,KP-1))/(Z(IA,KP)-Z(IA,KP-1))
B1=R(IA,KP-1)-AM*Z(IA,KP-1)
B2=RP+ZP/AM
ZA=(B2-B1)*AM/(AM**2+1.0)
RA=B2-ZA/AM
GO TO 10
5 RA=RP
ZA=Z(IA,KP-1)
GO TO 10
6 RA=P(IA,KP-1)
ZA=ZP
10 D(I)=SQRT((RA-RP)**2+(ZA-ZP)**2)
DT=D(1)+D(2)
AA=(D(1)*A(IP,KP-1)+D(2)*A(IP-1,KP-1))/DT
AB=(D(1)*A(IP,KP)+D(2)*A(IP-1,KP))/DT
DO 20 K=1,2
KA=KP-K+1
RC=(D(1)*R(IP,KA)+D(2)*R(IP-1,KA))/DT
ZC=(D(1)*Z(IP,KA)+D(2)*Z(IP-1,KA))/DT
20 DD(K)=SQRT((RP-RC)**2+(ZP-ZC)**2)
DT=DD(1)+DD(2)
AP=(DD(1)*AA+DD(2)*AB)/DT
RETURN
END

```



```

SUBROUTINE SPLINT (X,Y,N,Z,MAX,YINT)
DIMENSION X(50),Y(50),S(50),A(50),R(50),C(50),F(50),W(50),SR(50),
LG(50),FM(50),Z(50),YINT(50)
COMMON Q
INTEGER Q
DO 10 I=2,N
S(I)=X(I)-X(I-1)
10 CONTINUE
ND=N-1
DO 20 I=2,ND
A(I)=S(I)/6.0
R(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)
A(N)=-.5
R(1)=1.0
R(N)=1.0
C(1)=-.5
F(1)=0.0
F(N)=0.0
W(1)=F(1)
SR(1)=C(1)/W(1)
G(1)=0.0
DO 30 I=2,N
W(I)=R(I)-A(I)*SR(I-1)
SR(I)=C(I)/W(I)
30 G(I)=(F(I)-A(I)*G(I-1))/W(I)
FM(N)=G(N)
DO 40 I=2,N
K=N+1-I
40 FM(K)=G(K)-SR(K)*FM(K+1)
DO 50 I=1,MAX
K=2
IF(Z(I)-X(1)) 60,50,70
50 YINT(I)=Y(1)
GO TO 90
60 IF(Z(I).LT.(1.1*X(1)-.1*X(2)))WRITE (6,1000)Z(I)
GO TO 35
1000 FORMAT (17H OUT OF RANGE Z #F10.6)
65 IF(Z(I).GT.(1.1*X(N)-.1*X(N-1))) WRITE (6,1000)Z(I)
K=N
GO TO 85
70 IF(Z(I)-X(K)) 85,75,80
75 YINT(I)=Y(K)
GO TO 90
80 K=K+1
IF(K=N) 70,70,65
85 YINT(I) = FM(K-1)*(X(K)-Z(I))*3/6./S(K)+FM(K)*(Z(I)-X(K-1))*3/6.
1/S(K)+(Y(K)/S(K)-FM(K)*S(K)/6.)*(Z(I)-X(K-1))+(Y(K-1)/S(K)-FM(K-1)
2*S(K)/6.)*(X(K)-Z(I))
90 CONTINUE
MAX = MAX0(N,MAX)
IF(Q.EQ.16) WRITE(6,1010) N,MAX,(X(I),Y(I),Z(I),YINT(I),I=1,MAX)

```

```

SUBROUTINE RBCH(YR,R,Z,T,H,FTA,NOUT)
DIMENSION YR(6),R(2),Z(2),FTA(2)
IF( ABS(Z(2)-Z(1)).LT.1.0E-12) GO TO 100
IF( ABS(R(2)-R(1)).LT.1.0E-12) GO TO 200
SM=(R(2)-R(1))/(Z(2)-Z(1))
IF( ABS(YR(6)).LE.1.0E-12) GO TO 10
IF( ABS(YR(7)).LE.1.0E-12) GO TO 20
PM=YR(2)/YR(6)
ZR=(Z(1)*SM-YR(5)*PM+YR(1)-R(1))/(SM-PM)
RR=SM*(ZR-Z(1))+R(1)
GO TO 30
10 ZR=YR(5)
RR=SM*(ZR-Z(1))+R(1)
GO TO 30
20 RR=YR(1)
ZR=(RR-R(1))/SM+Z(1)
GO TO 30
100 IF( ABS(YR(2)).LT.1.0E-12) GO TO 110
PM=YR(2)/YR(6)
RR=PM*(Z(1)-YR(5))+YR(1)
ZR=Z(1)
GO TO 30
110 RR=YR(1)
ZR=Z(1)
GO TO 30
200 IF( ABS(YR(6)).LT.1.0E-12) GO TO 250
PM=YR(2)/YR(6)
RR=R(1)
ZR=(RR-YR(1))/PM+YR(5)
GO TO 30
250 RR=R(1)
ZR=YR(5)
30 DRP= SQRT((RR-YR(1))**2+(ZR-YR(5))**2)
VM= SQRT(YR(2)**2+YR(6)**2)
DT=H-DRP/VM
TB=YR(3)+YR(4)*DRP/VM
WRITE(6,1000) NOUT,RR,TR,ZR
GAMMA= ATAN2((R(2)-R(1)),(Z(2)-Z(1))) -1.5707963
ALPHA=3.1415927+ ATAN2(YR(2),YR(6))
VN=VM* COS(GAMMA-ALPHA)
VTR=VM* SIN(GAMMA-ALPHA)
VNP=-VN*FTA(1)
VTRP=VTR*FTA(2)
YR(4)=YR(4)*FTA(2)
VMP= SQRT(VNP**2+VTRP**2)
BETA= ATAN2(VTRP,-VNP)
YR(2)= VMP* SIN(GAMMA+BETA)
YR(6)= VMP* COS(GAMMA+BETA)
YR(1)=RR+YR(2)*DT
YR(3)=TB+YR(4)*DT
YR(5)=ZR+YR(6)*DT
T=T+H
RETURN

```

```

SUBROUTINE RBRB(YR,THET,Z,T,H,FTA,NDUT,DTHT)
DIMENSION YR(6),THET(2),Z(2),FTA(2)
DIMENSION Y(2)
IF(NDUT.NE.5) GO TO 4
DO 4 I=1,2
4 Y(I)=(THET(I)+DTHT)*YR(1)
5 IF(NDUT.NE.6) GO TO 7
DO 5 I=1,2
6 Y(I)=(THET(I)-DTHT)*YR(1)
7 CONTINUE
IF(ABS(Z(2)-Z(1)).LT.1.0E-12) GO TO 10
IF(ABS(Y(2)-Y(1)).LT.1.0E-12) GO TO 20
SM=(Y(2)-Y(1))/(Z(2)-Z(1))
IF(ABS(YR(4)).LT.1.0E-12) GO TO 30
IF(ABS(YR(6)).LT.1.0E-12) GO TO 40
PM=YR(4)*YR(1)/YR(6)
ZR=(YR(3)*YR(1)-PM*YR(5)-Y(1)+SM*Z(1))/(SM-PM)
YR=SM*(ZR-Z(1))+Y(1)
GO TO 100
10 IF(ABS(YR(4)).LT.1.0E-12) GO TO 15
ZR=Z(1)
PM=YR(4)*YR(1)/YR(6)
YR=PM*(Z(1)-YR(5))+YR(1)*YR(3)
GO TO 100
15 ZR=Z(1)
YR=YR(1)*YR(3)
GO TO 100
20 IF(ABS(YR(6)).LT.1.0E-12) GO TO 25
YR=Y(1)
PM=YR(4)*YR(1)/YR(6)
ZR=(YR-YR(1)*YR(3)+PM*YR(5))/PM
GO TO 100
25 YR=Y(1)
ZR=YR(5)
GO TO 100
30 YR=YR(1)*YR(3)
ZR=(YR-Y(1)+SM*Z(1))/SM
GO TO 100
40 ZR=YR(5)
YR=SM*(ZR-Z(1))+Y(1)
100 CONTINUE
DBP=SQRT((YR-YR(1)*YR(3))**2+(ZR-YR(5))**2)
VBR=SQRT((YR(4)*YR(1))**2+YR(6)**2)
DT1=DBP/VBR
DT=H-DT1
T3=YR/YR(1)
RH=YR(2)*DT1+YR(1)
WRITE(6,1000) NDUT,RR,TR,ZR
GAMMA= ATAN2((Y(2)-Y(1)),(Z(2)-Z(1))) -1.5707963
ALPHA=3.1415927-ATAN2((YR(1)*YR(4)),YR(6))
VN=VBR* COS(GAMMA-ALPHA)
VTR=VBR* SIN(GAMMA-ALPHA)
VJ2=-VN*FTA(1)

```

```

VTRP=VTR*ETA(2)
YR(2)=YR(2)*ETA(2)
VBBP=SQRT(VNP**2+VTRP**2)
BETA= ATAN2(VTRP,-VNP)
YR(4)= VBBP* SIN(GAMMA+BETA)/RB
YR(6)= VBBP* COS(GAMMA+BETA)
YR(1)=RB+YR(2)*DT
YR(3)=TB+YR(4)*DT
YR(5)=ZB+YR(6)*DT
T=T+H
RETURN
1000 FORMAT(29H PARTICLE BOUNCED OFF SURFACE,15,2X,16H(RB,TB,ZB) ARE (,
1 3F15.5,1H))
END

```

The function routine RUNGE has been removed from the published form of this report to protect the copyright of the authors of Reference 5.

```

SUBROUTINE RNUMBR(RENOLD,DGFC,CD)
IF( ABS(RENOLD).LT.1.0E-12) RENOLD=1.0E-12
IF(RENOLD.LT.1.0) GO TO 26
IF((RENOLD.GE.1.0).AND.(RENOLD.LT.1.0E3)) GO TO 27
CD=DGFC*0.4
RETURN
26 CD=DGFC*(4.5+24.0/RENOLD)
RETURN
27 ARE=ALOG(RENOLD)
CD=(28.5-24.0*ARE+9.0682*ARE**2-1.7713*ARE**3+0.1718*ARE**4
1 -0.0065*ARE**5)*DGFC
RETURN
END

```

```

SUBROUTINE RADUMP(MX,KMX,MHTA)
WRITE(7,100) MX,KMX,MHTA
100 FORMAT(3I5)
RETURN
END

```

```

SUBROUTINE RARFAD(MX,KMX,MHTA)
READ(5,100) MX,KMX,MHTA
100 FORMAT(3I5)
RETURN
END

```

```

SUBROUTINE BRDUMP(X)
DIMENSION X(21)
DO 10 K=1,19,3
10 WRITE(7,100) X(K),X(K+1),X(K+2),K
100 FORMAT(3E20.8,12X,2HBB,13)
RETURN
END

```

```

SUBROUTINE BRREAD(X)
DIMENSION X(21)
DO 10 K=1,19,3
10 READ(5,100) X(K),X(K+1),X(K+2)
100 FORMAT(3E20.8)
RETURN
END

```

```

SUBROUTINE BCDUMP(X)
DIMENSION X(21,21)
DO 20 I=1,21,1
DO 10 K=1,19,3
10 WRITE(7,100) X(I,K),X(I,K+1),X(I,K+2),I,K
20 CONTINUE
100 FORMAT(3E20.8,12X,2HRC,2I3)
RETURN
END

```

```

SUBROUTINE BCREAD(X)
DIMENSION X(21,21)
DO 20 I=1,21,1
DO 10 K=1,19,3
10 READ(5,100) X(I,K),X(I,K+1),X(I,K+2)
20 CONTINUE
100 FORMAT(3E20.8)
RETURN
END

```

```

SUBROUTINE BRDUMP(A,B,C)
WRITE(7,100) A,B,C
100 FORMAT(3E20.8)
RETURN
END

```

```

SUBROUTINE BRREAD(A,B,C)
READ(5,100) A,B,C
100 FORMAT(3E20.8)
RETURN
END

```

**ORIGINAL PAGE IS
OF POOR QUALITY**

Example (Part A)

The example presented here uses the ft., lbm., second system of units. The rotor in this case has an inlet radius of 2.961 inches (7.52 cm). The gas flow is that corresponding to a gas with inlet temperature of 2660°R and density of 0.1277 lbm/ft³.

A more detailed description of the input data format is contained in Reference 9. Only the output from the computer run is included on the following pages.

The solution reveals no large variations in the radius of curvature and smoothly accelerating gas velocities from inlet to exit.

Inputting the variable BCDP as described in Reference 9 with a value of 2 will cause a set of data cards to be punched. These data cards are used as the first part of the input for the particle trajectory program.

RUN NO. 1 INPUT DATA CARD LISTING

10	11	17	13	5390.0000	0.9860	13.0000	1.4000	1715.0000
0	0	0	0	592.0000	155.0000	0.1941	0.0010	2.5000
13	2	2	1	1.0000	-1.0000	-35.0000	1.7500	0.1000
0.00001								
0.3000	0.3400	0.3948	0.4810	0.5412	0.6193	0.7080		
0.8130	0.9100	1.0000						
0.0	-0.0270	-0.0530	-0.0540	0.0090	0.1370	0.4100		
0.7300	0.9100	1.0400						
2.2500	2.0520	1.8610	1.6800	1.6000	1.5341	1.4960		
1.4785	1.4751	1.4750						
2.2500	2.0180	1.7630	1.4120	1.2080	1.0010	0.7790		
0.6800	0.6750	0.6750						
0.0	0.0	0.0	-0.0000	-0.0020	-0.0000	-0.0240		
-0.0517	-0.0572	-0.1032	-0.2487	-0.3512	-0.4600			
-0.1000	0.0	0.1000	0.2000	0.3000	0.4000	0.5000		
0.6000	0.7000	0.8000	0.9000	1.0000	1.1000			
0.3000	0.3500	0.3220	0.2900	0.2600	0.2300	0.2010		
0.1750	0.1500	0.1280	0.1080	0.0920	0.0790			
0.3750	0.3450	0.3110	0.2800	0.2500	0.2200	0.1910		
0.1650	0.1400	0.1190	0.1000	0.0850	0.0740			
0.3650	0.3300	0.3000	0.2700	0.2400	0.2100	0.1810		
0.1550	0.1310	0.1100	0.0920	0.0780	0.0660			
0.3550	0.3210	0.2900	0.2590	0.2290	0.2000	0.1710		
0.1460	0.1210	0.1010	0.0850	0.0730	0.0630			
0.3450	0.3100	0.2800	0.2490	0.2190	0.1900	0.1610		
0.1360	0.1130	0.0930	0.0790	0.0690	0.0600			
0.3300	0.3000	0.2690	0.2380	0.2080	0.1790	0.1500		
0.1270	0.1040	0.0860	0.0730	0.0630	0.0550			
0.3200	0.2900	0.2580	0.2270	0.1970	0.1680	0.1400		
0.1170	0.0950	0.0780	0.0680	0.0590	0.0520			
0.3100	0.2790	0.2470	0.2150	0.1860	0.1570	0.1300		
0.1070	0.0870	0.0730	0.0620	0.0550	0.0490			
0.3000	0.2680	0.2360	0.2040	0.1750	0.1470	0.1200		
0.0980	0.0790	0.0680	0.0580	0.0510	0.0450			
0.2900	0.2570	0.2240	0.1930	0.1640	0.1360	0.1100		
0.0880	0.0730	0.0620	0.0540	0.0470	0.0420			
0.2810	0.2460	0.2130	0.1820	0.1530	0.1240	0.1000		
0.0770	0.0670	0.0570	0.0500	0.0440	0.0390			
0.2730	0.2230	0.2020	0.1700	0.1410	0.1130	0.0890		
0.0700	0.0610	0.0520	0.4600	0.0430	0.0370			
0.1940	0.1940	0.1900	0.1590	0.1300	0.1020	0.0790		
0.0600	0.0	0.0	0.0	0.0	0.0			
0.1660	0.1660	0.1660	0.1480	0.1180	0.0910	0.0690		
0.2500	0.0	0.0	0.0	0.0	0.0			
0.1370	0.1370	0.1370	0.1370	0.1060	0.0800	0.0590		
0.0400	0.0	0.0	0.0	0.0	0.0			
0.1090	0.1090	0.1090	0.1090	0.0930	0.0700	0.0500		
0.0300	0.0	0.0	0.0	0.0	0.0			
0.0800	0.0800	0.0800	0.0800	0.0800	0.0600	0.0400		
0.0200	0.0	0.0	0.0	0.0	0.0			
-0.1000	0.0	0.0000	0.2000	0.3000	0.4000	0.5000		
0.6000	0.7000	0.8000	0.9000	1.0000	1.1000			
0.6500	0.7500	0.8500	0.9500	1.0500	1.1500	1.2500		
1.3500	1.4500	1.5500	1.6500	1.7500	1.8500	1.9500		
2.0500	2.1500	2.2500						

107

STAG. SPEED OF SOUND AT INLET = 1192.22

ITERATION NO. 1	MAX. STREAMLINE CHANGE = 0.222071
ITERATION NO. 2	MAX. STREAMLINE CHANGE = 0.208135
ITERATION NO. 3	MAX. STREAMLINE CHANGE = 0.164157
ITERATION NO. 4	MAX. STREAMLINE CHANGE = 0.161030
ITERATION NO. 5	MAX. STREAMLINE CHANGE = 0.141867
ITERATION NO. 6	MAX. STREAMLINE CHANGE = 0.124375
ITERATION NO. 7	MAX. STREAMLINE CHANGE = 0.108968
ITERATION NO. 8	MAX. STREAMLINE CHANGE = 0.095628
ITERATION NO. 9	MAX. STREAMLINE CHANGE = 0.084726
ITERATION NO. 10	MAX. STREAMLINE CHANGE = 0.073936
ITERATION NO. 11	MAX. STREAMLINE CHANGE = 0.065144
ITERATION NO. 12	MAX. STREAMLINE CHANGE = 0.057467
ITERATION NO. 13	MAX. STREAMLINE CHANGE = 0.050730
ITERATION NO. 14	MAX. STREAMLINE CHANGE = 0.044813
ITERATION NO. 15	MAX. STREAMLINE CHANGE = 0.039612
ITERATION NO. 16	MAX. STREAMLINE CHANGE = 0.035022
ITERATION NO. 17	MAX. STREAMLINE CHANGE = 0.030981
ITERATION NO. 18	MAX. STREAMLINE CHANGE = 0.027411
ITERATION NO. 19	MAX. STREAMLINE CHANGE = 0.024260
ITERATION NO. 20	MAX. STREAMLINE CHANGE = 0.021477
ITERATION NO. 21	MAX. STREAMLINE CHANGE = 0.019018
ITERATION NO. 22	MAX. STREAMLINE CHANGE = 0.016839
ITERATION NO. 23	MAX. STREAMLINE CHANGE = 0.014918
ITERATION NO. 24	MAX. STREAMLINE CHANGE = 0.013239
ITERATION NO. 25	MAX. STREAMLINE CHANGE = 0.011765
ITERATION NO. 26	MAX. STREAMLINE CHANGE = 0.010454
ITERATION NO. 27	MAX. STREAMLINE CHANGE = 0.009275
ITERATION NO. 28	MAX. STREAMLINE CHANGE = 0.008263
ITERATION NO. 29	MAX. STREAMLINE CHANGE = 0.007350
ITERATION NO. 30	MAX. STREAMLINE CHANGE = 0.006537
ITERATION NO. 31	MAX. STREAMLINE CHANGE = 0.005815
ITERATION NO. 32	MAX. STREAMLINE CHANGE = 0.005170
ITERATION NO. 33	MAX. STREAMLINE CHANGE = 0.004608
ITERATION NO. 34	MAX. STREAMLINE CHANGE = 0.004102
ITERATION NO. 35	MAX. STREAMLINE CHANGE = 0.003652
ITERATION NO. 36	MAX. STREAMLINE CHANGE = 0.003251
ITERATION NO. 37	MAX. STREAMLINE CHANGE = 0.002896
ITERATION NO. 38	MAX. STREAMLINE CHANGE = 0.002591
ITERATION NO. 39	MAX. STREAMLINE CHANGE = 0.002298
ITERATION NO. 40	MAX. STREAMLINE CHANGE = 0.002048
ITERATION NO. 41	MAX. STREAMLINE CHANGE = 0.001824
ITERATION NO. 42	MAX. STREAMLINE CHANGE = 0.001628
ITERATION NO. 43	MAX. STREAMLINE CHANGE = 0.001449
ITERATION NO. 44	MAX. STREAMLINE CHANGE = 0.001293
ITERATION NO. 45	MAX. STREAMLINE CHANGE = 0.001152
ITERATION NO. 46	MAX. STREAMLINE CHANGE = 0.001026
ITERATION NO. 47	MAX. STREAMLINE CHANGE = 0.000916

108

ORIGINAL PAGE IS
OF POOR QUALITY

DN	Z	R	AA	PAR	WTR	PC
STREAMLINE 1						
0.0	0.0	2.250000	295.632083	28.948502	0.0	0.206545
0.0	-0.027000	2.018000	244.252365	26.055986	0.0	3.418553
0.0	-0.053000	1.763000	224.791565	23.130295	0.0	2.373647
0.0	-0.054000	1.411999	176.907639	20.036743	0.0	17.658295
0.0	0.009000	1.207999	183.239822	18.452895	0.0	13.869154
0.0	0.137000	1.000999	186.012283	17.094681	0.0	14.145462
0.0	0.411000	0.778999	193.714355	15.786360	0.0	11.166971
0.0	0.730000	0.680000	248.374222	14.876602	0.0	14.036775
0.0	0.910000	0.675000	399.253906	13.774944	0.0	-2.11723
0.0	1.039999	0.675000	537.193115	12.531340	0.0	-0.055512
STREAMLINE 3						
0.060683	0.060684	2.250000	296.517578	28.940155	0.0	1.552913
0.080352	0.053011	2.023412	248.089966	26.072495	0.0	3.637551
0.18628	0.063082	1.733403	240.976563	23.196121	0.0	9.014014
0.209881	0.133653	1.506001	236.871826	20.346039	0.0	16.129425
0.250463	0.217105	1.361281	248.646240	19.011536	0.0	15.952235
0.305679	0.342212	1.227825	270.109154	17.807739	0.0	13.123368
0.340734	0.540776	1.093552	309.077148	16.529480	0.0	8.132901
0.315767	0.762647	0.994385	409.825195	15.038394	0.0	5.953679
0.271172	0.910000	0.940192	509.27020	13.836711	0.0	5.602612
0.241698	1.027919	0.916604	596.654053	12.715893	0.0	3.030144
STREAMLINE 5						
0.121951	0.121953	2.250000	298.513184	28.921249	0.0	2.357697
0.157672	0.130002	2.032544	256.038330	26.069077	0.0	5.717117
0.221103	0.162993	1.810263	262.155762	23.215240	0.0	10.274740
0.344822	0.254303	1.586439	285.877197	20.447647	0.0	15.918599
0.406078	0.335959	1.448825	310.122803	19.164154	0.0	16.075500
0.460506	0.445950	1.342490	342.539795	17.975664	0.0	13.792950
0.500131	0.671954	1.247349	397.685547	16.637630	0.0	9.973216
0.484804	0.780124	1.162223	506.184082	14.985494	0.0	7.670432
0.440295	0.910000	1.121312	593.957275	13.717205	0.0	6.937749
0.420351	1.019007	1.039484	655.230469	12.794467	0.0	3.184549
STREAMLINE 7						
0.160830	0.160833	2.250000	301.141846	28.896210	0.0	2.523566
0.231808	0.203823	2.039333	265.339619	26.044357	0.0	6.033831
0.310030	0.249865	1.829290	285.497061	23.193777	0.0	10.567771
0.447203	0.345842	1.612293	335.270936	20.390320	0.0	15.847062
0.511862	0.421132	1.511561	375.252686	19.052689	0.0	17.242813
0.567918	0.510312	1.422142	425.136963	17.756897	0.0	16.933450
0.611972	0.644880	1.344129	496.145264	16.333649	0.0	14.253599
0.610006	0.793069	1.285758	594.472656	14.713057	0.0	11.125129
0.583000	0.910000	1.258323	666.619375	13.561134	0.0	6.894609
0.562545	1.011756	1.239857	711.296387	12.801081	0.0	3.117519
STREAMLINE 9						
0.240468	0.240472	2.250000	304.130371	28.867447	0.0	2.336857
0.302183	0.273896	2.045975	277.515381	26.006424	0.0	5.366113
0.380520	0.320542	1.840351	311.437061	23.096755	0.0	10.363472
0.529647	0.419555	1.649219	389.917480	20.163773	0.0	16.187617
0.554129	0.487371	1.500353	452.623291	18.635101	0.0	20.007431
0.650769	0.573596	1.483581	528.607422	17.065231	0.0	23.723633
0.700519	0.671865	1.425900	605.048340	15.604924	0.0	21.373779
0.712856	0.803303	1.383061	680.498291	14.257373	0.0	14.370979
0.690210	0.910000	1.373243	728.133789	13.408404	0.0	9.760336
0.690256	1.005528	1.304419	761.512207	12.798480	0.0	2.652925
STREAMLINE 11						
0.29995	0.300000	2.250000	307.447510	28.835251	0.0	2.002020
0.368506	0.340000	2.051998	289.208008	25.956879	0.0	4.448371
0.456393	0.394807	1.863998	347.774414	22.947128	0.0	9.891227
0.59870	0.481000	1.679998	453.198672	19.734543	0.0	16.634583

607

0.060463	0.541200	1.577998	551.020996	17.784127	0.7	25.383874
0.718853	0.619339	1.534079	670.338838	15.613330	0.0	30.459975
0.776429	0.708000	1.495998	727.275635	14.445056	0.0	29.738525
0.802772	0.813000	1.478498	759.715088	13.736666	0.7	15.735647
0.800066	0.910000	1.475098	777.707520	13.326412	0.0	2.621365
0.800469	1.000000	1.474998	804.683594	12.826401	0.0	1.255404

110

ITERATION NO. 49

	AL	RL	SM	BETA	TT	SA	SB	SC	SD
STREAMLINE 1									
-96.791061	0.264545	0.0	-34.911610	0.031000	-0.016314	4747.835938	0.137425	0.358299	
-96.445129	0.418553	0.233566	-13.164653	0.146281	-0.045529	1000.076663	0.403030	27.377579	
-91.303146	2.071640	0.489988	-0.021095	0.219230	-0.118752	1494.835938	2.365304	125.162001	
-81.547180	17.650295	0.840890	-0.000131	0.289255	2.595669	452.732422	17.466476	-67.108231	
-65.287501	10.867164	1.056396	0.073368	0.291358	4.563654	-422.876465	9.873979	193.090963	
-50.600610	14.145462	1.297773	-0.071968	0.273430	8.478398	-109.570862	10.930734	117.837570	
-28.514648	11.100991	1.649643	-4.242935	0.214962	9.674347	856.984619	5.255947	408.039053	
-5.167764	14.006795	1.984607	-22.761575	0.152448	9.221153	3764.851563	0.833668	7254.128906	
0.328677	-0.110723	2.164676	-33.047679	0.124522	-5.359133	5979.031250	0.002682	12124.316406	
-0.022912	-0.055512	2.294676	-37.600525	0.107411	-6.653266	6574.746094	-0.002661	13236.978125	
STREAMLINE 3									
-93.117676	1.552913	0.0	-34.760129	0.090000	-0.056727	4694.649438	1.041507	-3.471580	
-90.430634	3.607551	0.224719	-13.426351	0.144130	-0.026160	853.124023	3.680257	-0.351072	
-33.693914	9.011018	0.461741	-0.187078	0.202734	0.989301	308.530729	8.959467	-33.619497	
-16.571472	10.129425	0.753329	-0.058870	0.218700	6.443112	-340.042236	14.799632	178.280319	
-53.511100	15.452236	0.920084	-0.675334	0.208830	9.483495	-849.320557	12.821884	899.535371	
-43.637040	13.123368	1.103010	-2.979316	0.187637	9.909909	-363.689453	8.495051	1654.796875	
-28.451492	8.132471	1.342635	-14.068773	0.152189	6.079705	1788.532459	3.294352	4383.519531	
-20.309250	5.953609	1.585842	-32.615182	0.127046	0.454260	5048.824219	0.168118	9177.019750	
-15.628822	6.602612	1.740787	-41.269531	0.113726	-1.925783	6435.250000	-0.538733	10480.882813	
-12.377753	3.033144	1.862358	-45.235260	0.103878	-5.134829	7030.019531	-1.173935	10658.195313	
STREAMLINE 5									
-89.346924	2.357699	0.0	-34.399130	0.080000	0.017973	4510.097656	1.576772	6.821116	
-85.310944	5.717117	0.217644	-11.395299	0.141123	0.448847	477.819824	5.472179	54.345159	
-76.494675	13.274740	0.442354	-0.067580	0.174996	2.311893	-678.640625	10.009554	245.771451	
-60.754181	15.518599	0.702721	-1.036365	0.175554	7.772088	-1268.121094	13.880455	1187.100586	
-49.684494	16.075500	0.845901	-2.810857	0.165531	10.355851	-1159.979980	12.204514	2174.252930	
-33.660919	13.752950	0.998938	-3.611643	0.148094	10.328422	213.088379	8.263069	3841.041016	
-23.082184	9.970216	1.185081	-21.618515	0.128791	6.289762	3145.394531	3.355909	7227.346875	
-19.899139	7.670432	1.379328	-38.252792	0.115591	0.489895	6314.210938	0.177331	10574.556257	
-15.149429	6.933749	1.515944	-46.189529	0.108504	-2.366839	7585.464944	-0.640830	9550.392198	
-12.483317	3.164549	1.628169	-50.091349	0.102236	-5.169444	8142.582331	-1.144661	7938.335938	
STREAMLINE 7									
-85.563324	2.525566	0.0	-34.968750	0.080011	0.127869	4211.714844	1.648020	72.231430	
-81.395660	6.009341	0.211368	-12.066625	0.137025	0.854680	31.317627	5.648438	288.062988	
-73.105225	10.067771	0.426956	-1.469376	0.146942	3.067115	-1609.209717	10.098417	915.695313	
-53.110614	15.847062	0.669221	-2.822482	0.144654	8.021630	-2249.380127	13.400957	2750.246582	
-48.146820	17.242813	0.789701	-6.852255	0.136819	11.228038	-1505.800781	12.534456	4618.347656	
-37.309860	16.833450	0.921819	-14.509953	0.124902	12.018642	902.051514	9.159025	6884.910156	
-26.175706	14.253599	1.075753	-28.155437	0.114753	7.955796	4713.289291	3.910559	9694.574217	
-16.544713	11.125129	1.229660	-42.659570	0.107173	1.484939	7437.203125	0.461253	10342.351563	
-11.476128	6.804609	1.350070	-49.896286	0.104682	-2.813398	8357.382913	-0.571172	8024.203125	
-9.063053	3.117519	1.453334	-53.657333	0.099668	-5.198359	8771.852331	-0.829669	5809.210938	
STREAMLINE 9									
-82.304672	2.336857	0.0	-34.990158	0.080081	0.194861	3877.222412	1.442108	198.072571	
-73.601242	5.366113	0.206844	-12.909193	0.114779	0.992031	-462.827637	4.920492	653.158691	
-71.359677	10.003492	0.413476	-2.781227	0.123475	3.346933	-2632.400635	9.754119	1845.010985	
-57.334656	16.187607	0.631187	-5.975125	0.121622	8.563377	-3490.174590	13.356611	5399.914063	
-47.626007	20.0037431	0.742176	-11.343126	0.117098	12.665788	-2411.885986	13.882699	8586.516719	
-35.326230	23.722633	0.858424	-20.683314	0.110698	15.919282	2309.855957	11.307494	10965.921875	
-22.290340	21.303009	0.973460	-34.048294	0.105247	10.893934	6522.996094	0.465796	10803.105469	
-11.312394	14.370979	1.108619	-46.285843	0.103303	2.216461	8268.464844	0.443390	7934.511719	
-6.344591	5.760336	1.216086	-52.737564	0.100098	-3.436270	8768.257813	-0.382075	5619.281250	

-4.467793	2.652925	1.312021	-56.309677	0.077732	-5.274943	9051.046875	-0.410312	4511.976394
STREAMLINE 11								
-79.737608	2.302020	3.0	-35.144261	0.083277	0.197741	3641.653564	1.112099	355.649678
-76.358673	4.446370	3.202031	-14.161762	0.095963	0.915480	-1085.672412	3.921233	1195.637851
-70.267807	9.091227	0.400707	-4.722431	0.103803	3.282253	-3716.677666	9.157797	3377.983398
-57.715134	16.634583	0.601185	-7.341342	0.105953	8.462729	-5657.800781	13.394559	9199.488281
-47.657852	25.363804	0.701305	-15.723335	0.134203	15.290913	-4649.582031	16.779678	15753.722656
-31.560181	38.455976	0.803493	-27.270671	0.101182	24.249039	4225.871794	14.894911	15374.472344
-15.618239	29.736525	0.900328	-39.563538	0.099291	13.767314	8392.257813	3.848630	7444.877625
-4.485056	15.735647	1.006476	-49.197632	0.099745	2.047854	8514.763625	0.160632	3137.242188
-0.525846	2.621365	1.103536	-54.857422	0.097805	-4.571162	8839.652344	-0.041954	2781.229248
0.373310	1.255404	1.193536	-58.255966	0.095035	-5.536105	9161.582031	0.029017	4238.095938

112

DN	Z	R	WA	PVS	WTP	PC
STREAMLINE 1						
0.0	0.0	2.250000	295.632180	28.948502	-22.249878	1.274549
0.0	-0.053000	1.763000	224.791565	23.130295	-306.866599	2.373660
0.0	-0.054000	1.411999	176.912857	20.036713	-117.234940	17.658295
0.0	0.137000	1.033919	186.016708	17.094650	115.757858	14.145462
0.0	0.410000	0.778999	153.728287	15.786307	197.654394	11.164971
0.0	0.733000	0.633000	248.374222	14.976602	232.853397	14.956745
0.0	0.910000	0.675000	399.277832	13.774780	344.622803	-0.110723
0.0	1.034999	0.675000	537.226807	12.531041	458.083984	-0.055512
STREAMLINE 3						
0.060684	0.060685	2.250000	296.518066	28.940155	-27.389908	1.553555
0.080363	0.053022	2.025413	248.088745	26.072510	-221.903879	3.609227
0.110059	0.063108	1.763000	247.981291	23.196152	-217.159188	9.714427
0.209921	0.133698	1.505022	236.892212	20.346100	-3.155060	16.130051
0.258524	0.217154	1.361318	248.677002	19.011597	97.359497	15.951024
0.305977	0.342257	1.227876	270.160137	17.807861	177.968536	13.122724
0.349803	0.548803	1.093717	309.125000	16.529510	200.767746	8.130458
0.315625	0.762653	0.995143	409.868438	15.038340	283.413574	5.951674
0.271216	0.910000	0.945227	509.308594	13.806561	471.695768	6.673375
0.241932	1.027917	0.916639	596.692627	12.715644	499.987305	3.039512
STREAMLINE 5						
0.120453	0.120955	2.250000	298.514404	28.921249	-53.446077	2.358177
0.157609	0.137019	2.032345	256.040283	26.069077	-156.200637	5.718139
0.221138	0.163028	1.810275	262.165203	23.215225	-119.649353	10.275573
0.349881	0.254356	1.566464	285.907715	20.447632	45.509430	15.917777
0.406146	0.336014	1.448866	310.162842	19.164154	127.808746	16.073877
0.463579	0.445999	1.342564	342.588867	17.975464	170.034058	13.792167
0.503204	0.601983	1.209916	399.747070	16.637604	190.386154	9.969821
0.484874	0.789132	1.162292	576.234863	14.985426	318.764474	7.668449
0.446358	0.910000	1.121375	594.001221	13.717064	464.719971	6.928862
0.420404	1.019005	1.094894	655.275879	12.794203	561.632324	3.183752
STREAMLINE 7						
0.180832	0.180835	2.250000	301.143799	28.896210	-83.179711	2.523822
0.231824	0.205839	2.037304	266.363770	26.044327	-74.280319	6.004385
0.310063	0.249898	1.829287	285.500000	23.183716	-33.257859	10.569247
0.447254	0.345888	1.612316	335.303467	20.390213	48.339349	15.845944
0.511519	0.421179	1.511596	375.299072	19.052551	132.395637	17.242065
0.567479	0.518053	1.422168	425.196289	17.756668	143.375839	16.833572
0.612034	0.644924	1.344183	476.278984	16.333389	195.629913	14.253939
0.640069	0.743076	1.286421	594.514648	14.712975	374.632813	11.127616
0.563058	0.910000	1.203081	666.639404	13.561130	532.299316	6.801944
0.565976	1.011753	1.239911	711.337891	12.802856	625.729736	3.116441
STREAMLINE 9						
0.240469	0.240474	2.250000	304.132813	28.867447	31.448715	2.937012
0.302191	0.273937	2.045875	277.520752	26.076363	-19.822937	5.366443
0.389540	0.322562	1.846355	311.503732	23.096664	-21.733327	13.383789
0.529678	0.415882	1.649232	387.949918	20.163606	40.653183	16.187210
0.594164	0.487399	1.503373	452.657715	18.634973	83.171738	20.975753
0.653806	0.573621	1.433633	528.053076	17.064893	106.458579	23.726624
0.700597	0.670880	1.425934	605.083984	15.604748	232.032059	21.301834
0.712893	0.803706	1.337098	680.514648	14.257471	466.917725	14.377267
0.696251	0.910000	1.373279	729.140625	13.409529	607.758496	5.759739
0.690207	1.005527	1.364450	761.561504	12.798284	680.014160	2.652539
STREAMLINE 11						
0.259494	0.259499	2.250000	317.459195	28.835220	138.615967	2.052920
0.368506	0.360000	2.051998	289.211426	25.956833	26.414413	4.448370
0.456393	0.394800	1.860998	340.782715	22.947037	-24.681442	9.691227
0.596370	0.481000	1.679998	453.137859	19.734634	19.779663	16.634583

113

0.660983	0.541200	1.591993	551.003174	17.784256	-13.863796	29.383814
0.710643	0.619300	1.536077	670.272217	15.613315	77.112167	38.459976
0.776428	0.708777	1.495998	727.249047	14.445449	359.711670	29.738525
0.802712	0.813000	1.476443	759.687500	13.735049	597.143311	15.735647
0.800066	0.910000	1.475098	777.696777	13.326551	680.907959	2.621365
0.809968	1.001000	1.474998	804.698975	12.826185	718.401123	1.255434

114

Example (Part B)

The example presented here uses the ft., lbm., second system of units. The rotor in this case has an inlet radius of 0.26475 ft. (7.52 cm). The gas flow is that corresponding to a gas with inlet temperature of 2660°R and density of 0.1277 lb/ft³.

The particle studied has specific gravity of 2 and a diameter corresponding to about 10 microns. The velocity is initially in the same direction as the gas flow solution with a magnitude of one half the gas velocity. The particle bounces off the shroud surface before passing through the turbine.

The following pages contain a computer code sheet with the data arranged in the proper columns, and the output for this example. The block indicated as that punched by the Quasi-Orthogonal program is approximately a box of cards that is required to transfer the Quasi-Orthogonal information to this program.

ROTOR Example Case for NASA Report

1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80

Data set punched by Quasi-Orthogonal Program.

EXAMPLE FOR NASA RADIAL TURBINE V-PART/V-GAS = 50%.

	0.329E-6	492.0	198.6	1.0		
187.2	0.328E-4	0.1E-5	1.0E0	1.0	1.0	
0.24674	-305.5	-0.2590	-80.250	0.01375	0.0	
10						

Additional particle trajectories require a Card Set 2 arrangement here.
 These sets are stacked one after the other.

CARD SET 1

CARD SET 2

CARD SET 2

IMPORTANT DATA FROM QUASI-STATIC PROGRAM

MM 10 MM 21 MTS TA 10

RAMPA TMRP RBC RGAS SPACT ZSPLIT b
 1.3900 2460.0200 0.1277 1715.0000 1.0000 -0.0033 7330.0000

MM 12.0000 AIP 484.0000 PLCS 255.3556

MEP COORDINATES

R
 0.246750
 0.277942
 0.211000
 0.141492
 0.171117
 0.129298
 0.062900
 0.072700
 0.044099
 0.062900

TIP COORDINATES

R Z
 0.246750 0.026483
 0.225900 0.027483
 0.212400 0.025983
 0.195983 0.036400
 0.182442 0.045817
 0.170900 0.059150
 0.143000 0.075900
 0.127000 0.093900
 0.104150 0.110233
 0.125833 0.125000

X1 ARRAY

0.0 0.01200 0.02500 0.03500 0.05000 0.06500
 0.09000 0.10000 0.11000 0.12500

Y-STA ARRAY

0.0 0.0 0.0 -0.00066 -0.00741 -0.01760
 -0.07070 -0.17720 -0.25160 -0.38320

ORIGINAL PART IN
 OF POOR QUALITY

EXAMPLE FOR NASA RAPID TURNING V-PART/V-GAS = 50%

VTORFF 0.9960F-06	TRFF 452.0000	TSLT 198.6000	OGFC 1.0000
RMPF 2.0000	F18P 0.229F-04	F 0.100F-05	TMAX 1.0000
ETA-N 1.0000	ETA-T 1.0000	YR(1) 0.24674	YR(2) -0.25500
YR(3) -0.25500	YR(4) -0.25000	YR(5) 0.01375	YR(6) 0.0

PRINT DATA EVERY 10 STEPS

STEP	Y	YR(1)	YR(2)	YR(3)	YR(4)	YR(5)	YR(6)	RENOLD
0	0.0	0.24674	-305.80000	-0.25500	-80.25000	0.01375	0.0	
10	0.999999F-05	0.24244	-473.25395	-0.26650	-158.01053	0.01512	197.88184	568.81372
PARTICLE BOUNCED OFF SURFACE		6 (PP,TR,2B) AFE I		0.23868	-0.26180	0.01625		
20	0.999999F-04	0.23737	-523.15552	-0.26144	91.87747	0.01657	106.42526	675.89355
30	0.999999F-04	0.23215	-515.37500	-0.26166	-105.81550	0.01736	58.58961	532.99731
PARTICLE BOUNCED OFF SURFACE		6 (PP,TR,7B) AFE I		0.23160	-0.26180	0.01742		
PARTICLE BOUNCED OFF SURFACE		6 (PP,TR,7B) AFE I		0.22125	-0.26180	0.01761		
40	0.999995F-04	0.22706	-500.74172	-0.26176	95.76056	0.01782	35.60748	536.09888
50	0.999997F-04	0.22151	-514.41016	-0.26075	119.60564	0.01858	100.59963	545.00684
60	0.999991F-04	0.21562	-556.73975	-0.25940	140.53918	0.01962	132.56960	513.66211
70	0.999988F-04	0.20977	-602.44141	-0.25793	158.59013	0.02070	153.81693	625.21880
80	0.999986F-04	0.20387	-612.23730	-0.25520	333.42663	0.02261	210.18492	564.37915
90	0.999984F-04	0.19752	-616.78198	-0.25155	375.06860	0.02472	205.51757	522.54712
100	0.999982F-04	0.19141	-556.64205	-0.24771	419.00806	0.02700	251.27173	522.74585
110	0.109998F-03	0.18556	-560.15259	-0.24304	505.07764	0.03024	344.62891	485.57739
120	0.119998F-03	0.17970	-554.71118	-0.23775	540.29257	0.03365	328.19409	528.56460
130	0.129998F-03	0.17431	-486.02441	-0.23411	301.82593	0.03759	407.57153	420.81763
140	0.139997F-03	0.16970	-460.05347	-0.23108	302.96655	0.04268	523.16162	458.98584
150	0.149997F-03	0.16500	-485.40381	-0.22802	325.90161	0.04837	527.48901	455.85156
160	0.159997F-03	0.16023	-412.67188	-0.22795	-535.23145	0.05393	622.55811	537.72119
170	0.169997F-03	0.15563	-349.56396	-0.23446	-684.70974	0.06042	665.54932	287.07422
180	0.179996F-03	0.15218	-345.52285	-0.24094	-585.80200	0.06713	672.39111	348.02432
190	0.189996F-03	0.14750	-278.52894	-0.25277	-2210.40854	0.07403	729.54468	564.73730
200	0.199996F-03	0.14230	-248.72145	-0.27708	-2512.53174	0.08148	754.64673	172.18631
210	0.209996F-03	0.14484	-248.38708	-0.30157	-2338.92525	0.08905	757.38306	236.39944
220	0.219996F-03	0.14272	-182.62236	-0.33764	-4399.33203	0.09680	750.50073	376.36108
230	0.229996F-03	0.14045	-115.25328	-0.38300	-4575.14063	0.10474	757.69580	73.77063
240	0.239996F-03	0.13921	-127.42239	-0.43256	-5745.90234	0.11274	801.88745	471.13086
250	0.249996F-03	0.13804	-121.78424	-0.49234	-6215.82422	0.12075	806.32080	159.53368
255	0.259992F-03	0.13721	-121.57803	-0.53087	-6276.99605	0.12963	807.72144	63.04197

1 255

118

ORIGINAL PAGE IS
OF POOR QUALITY

VANPY - Particles in a Turning Vortex

This program calculates the trajectory of a particle in an inward turning vortex after first calculating the flow through such geometry.

The fluid flow solution is based on the principal of constant, $\lambda = V_u r$ along the streamlines, although the values λ , may be different for different streamlines. The variation in the radius of curvature of the streamlines between hub and shroud is taken to be linear. The original program was supplied by NASA Lewis Research Center, but was modified such that the input of data is easier and the velocity signs are consistent with those used in the solution of the particle trajectories.

Method

Figure 24 is a flow diagram of the program VANPY. The results of the fluid solution are stored in arrays that specify the velocity vectors of the gas at the intersection points of the orthogonals and streamlines. With the fluid solution completed, the program uses a Runge-Kutta technique to integrate the three-dimensional equations of motion of the particle.

The integration of the equations of motion over one time increment is first carried out using the gas velocity and the properties of the gas at point A to determine the new particle location B. The program then calls BLOCAT and POLATE to determine the gas velocity components at B_1 . A corrected gas velocity components are calculated from B_1 with the mean values at A and B_1 . The program then integrates the equations of motion again starting from A to find the corrected particle location B_2 . BLOCAT and POLATE are called again to give the gas velocity components at B_2 , and these velocity components are compared to the corresponding values at B_1 , if the difference is large, the previous iteration is repeated. Once the iteration has converged, the trajectory to point B has been determined and this point is used as the initial point for the next time increment.

The subroutine CONTIN is used to iterate for the hub surface velocity, the iteration technique is based on the mass flow calculated from the fluid solution.

The subroutine PABC calculates the coefficients of the parabola $y = ax^2 + bx + c$, passing through three given points. This subroutine is used in the fluid solution.

The subroutine FNTGRL performs the integration of a function $F(l)$ over equal size increments. This subroutine is used in the fluid solution.

The subroutine SPLINT interpolates points along a spline fit curve that lie between the points specifying the curve.

The subroutine RNUMBR is used to determine the drag coefficient based on a curve fit of the drag versus Reynolds Number data. Equations 20 and Figure 3 demonstrate the fit of these equations to the data.

The subroutine RUNGE uses a fourth order method to integrate a system of simultaneous first order ordinary differential equations across one time step. Reference 5 explains this subroutine in more detail.

The subroutine BLOCAT is used to determine the subscripts of the grid points that surround the location of the particle. Figure 15 shows the particle within a typical set of orthogonals and streamlines. The subroutine returns the values of IP and KP that locate the particle within a particular grid. If the particle is no longer within the boundaries of the flow field, the subroutine returns NOUT, which is a code specifying where the particle has moved out of bounds.

The subroutine POLATE interpolates the value of a variable whose values are specified at the four grid points surrounding the particle. Referring to Figure 15, the subroutine first determines the distances $D(1)$ and $D(2)$, and based on these distances, determines a weighted average of the variable at two locations on adjacent streamlines. These values are AA and AB. Then the subroutine determines $DD(1)$ and $DD(2)$ and uses these distances to get the weighted average of AA and AB at the position occupied by the particle.

The subroutine RBCH is used when the particle rebounds from the case or the hub. It is called whenever the particle position is outside the case or hub boundaries. The subroutine returns to the previous position, and linearly extends the trajectory over one time increment, with the bounce occurring at some portion of the time segment. The subroutine writes "BOUNCE OFF SURFACE (NOUT) ..." and prints the location of the bounce.

Input

The input cards take the following format. Any consistent system of units can be used. An example of the input data is included with the example case presented after the program listing. The data cards cover two main groups;

Group A

MX, KMX, Z1, Z2, R1, R2	(215, 4F10.5)
GAMMA, CP, TIP, RHOIP, WRFL	(5F10.5)
RC(2, 1) ... RC(2, KMX)	(8F10.5)
AL(1, 1) ... AL(MX, 1)	(3F10.5)
LAMBDA(1, 1) ... LAMBOA(1, KMX)	(8F10.5)

Group B

TITLE	(18A4)
VISREF, TREF, TSUT, DGFC	(D20.5, 3F10.3)
RHOP, DIAP, H, TMAX, ETA-N, ETA-T	(F10.2, 3D10.4, 2F10.3)
YR(1), YR(2), YR(3), YR(4), YR(5), YR(6)	(6F10.3)
NSTEP	(I5)

Data Group A.

These cards specify the parameters of the gas flow solution. If only these input cards were fed in the input, the fluid solution only will be obtained. Figure 25 indicates with more clarity some of the terms listed below.

- MX - Number of orthogonals.
- KMX - Number of streamlines.
- Z1 - Axial location of first orthogonal. (Length)
- Z2 - Axial location of the circle center. (Length)

- R1 - Radial location of last orthogonal. (Length)
- R2 - Radial location of the circle center. (Length)
- GAMMA - Ratio of specific heats.
- CP - Specific heat at constant pressure. (C_p , g) (Length²/Time² - Degrees Abs)
- TIP - Inlet stagnation temperature. (Degrees Abs.)
- RHOIP - Inlet stagnation density. (Mass/Length³)
- WTFL - Fluid mass flow. (Mass/Time)
- RC Array - Radius of curvature of streamlines 1 through KMX at orthogonals 2 through (MX-1). (Length). The radius of curvature of the streamlines along orthogonal 1 and MX are set equal to 10,000 by the program.
- AL Array - The angle α corresponding to the flow directions at the orthogonals 1 through MX, this angle is negative in the direction indicated in Figure 25. (Degrees)
- LAMBDA Array - The term $\lambda = rV_u$ for each streamline from 1 through KMX. (Length²/Time)

Data Group B.

These cards specify the variables that are used in determining the trajectory. For studies involving more than one particle, sets of input cards for different particles can be stacked together.

- TITLE - The first card is reproduced at the top of the first page of output for each particle. Any statement in columns 2 through 72 will be reproduced.
- VISREF - Reference viscosity corresponding to TREF. Used in Sutherland's Law. (Mass/Length Time)
- TREF - Reference temperature corresponding to VISREF. Used in Sutherland's Law. (Degrees Abs.)
- TSUT - Constant used in Sutherland's Law. Either 198.6°R or 110.0°K.
- DGFC - Drag factor. The drag coefficient based on Reynold's Number is multiplied by DGFC. Except in very special cases, this should be 1.0.

- RHOP - Particle density. (Mass/Length³)
- DIAP - Particle diameter. (Length)
- H - Time increment used in integration process. (Time)
- TMAX - Program stops if time exceeds TMAX. (Time)
- ETA-N - Normal restitution coefficient.
- ETA-T - Tangential restitution coefficient.
- YR(1) - Initial radial position of the particle. (Length)
- YR(2) - Initial radial velocity component of the particle.
(Length/Time)
- YR(3) - Initial angular position of the particle.
- YR(4) - Initial angular velocity of the particle. (Time⁻¹)
- YR(5) - Initial axial position of the particle. (Length)
- YR(6) - Initial axial velocity of the particle. (Length/Time)
- NSTEP - The program prints out trajectory information every
NSTEP time increments.

Output

An example listing of typical output is included after the program listing. This output is divided into several output groups.

Output Group A.

This set of output is an echo check of the input data and includes the calculated critical velocity of the gas. Such data checks are useful in correcting key punch mistakes on the data cards. The variables listed in this group were explained under Input.

Output Group B.

This set of output concerns the solution of the gas flow through the turning vortex. A new page is used for the printed output of different orthogonals. The R and Z arrays locate the intersection points of the orthogonals and the streamlines. The RC, ALPHA and LAMBDA arrays are explained under Input. The rest of the output formats are explained below.

- WTFLES - Estimated weight flow when iteration procedure stops.
(Mass/Time)

BETA Array - Angle between the meridional velocity component and the velocity vector, at the intersection of the orthogonals and streamlines. (Degrees)

V/V_{cr} Array - Dimensionless speed at the intersection point of the orthogonals and streamlines.

Output Group C.

This set of output corresponds to the particle trajectories. The first part is an echo check of the input cards corresponding to Data Set B. After initializing the variables, the program calculates and prints several similarity parameters to relate particles having similar trajectories. The variables that are printed are explained below.

DELTA - The characteristic length based on the critical density. (Length)

TAU - The particle time constant based on the viscosity at critical flow conditions. (Time)

RECR - The Reynolds Number of a particle whose velocity is equal to half the gas critical velocity.

After the similarity parameters are printed, trajectory information is printed every NSTEP time increments. This information includes the following terms.

L - Number count of the time increments.

T - Time

YR(1) - Position and velocity components of the particles.
... YR(6)

RENOLD - Reynold's Number of the particle at this location.

When the particle passes out of the flow field through the inlet or exit, the program always prints the trajectory information for the last point, which should be just outside the boundaries. Following this the value of the number of iterations, M, is printed. If the iteration procedure for the particle location B has not converged, the value of M will be 101, and a smaller time increment is recommended.

PARTICLE THROUGH A TURNING VORTEX

720626

THIS PROGRAM CALCULATES THE PARTICLE TRAJECTORY IN AN INWARD TURNING VORTEX, AFTER FIRST CALCULATING THE FLOW THROUGH SUCH A GEOMETRY. THE FLUIDS SOLUTION IS BASED ON A STREAMLINE CURVATURE METHOD IN WHICH A LINEAR VARIATION IN THE RADII OF CURVATURE OF THE STREAMLINES IS ASSUMED BETWEEN THE HUB AND THE SHROUD. THE HUB AND THE SHROUD CONTOURS ARE CIRCULAR AND THUS HAVE CONSTANT RADIUS OF CURVATURE. SEE THE PROGRAM DESCRIPTION FOR THE INPUT DATA RELATING TO THE FLUID SOLUTION.

FROM THE INITIAL POINT, GIVEN AS INPJT, THE PROGRAM USES A RUNGA-KUTTA METHOD TO INTEGRATE THE PARTICLE EQUATIONS OF MOTION TO DETERMINE THE LOCATION OF THE PARTICLE BASED ON A GIVEN TIME STEP. THE FLUIDS SOLUTION, AT GIVEN GRID POINTS IS USED TO CALCULATE THE AERODYNAMIC DRAG ON THE PARTICLE. SEE THE PROGRAM DESCRIPTION FOR THE INPUT DATA AND FORMAT RELATING TO THE TRAJECTORY SOLUTION.

W. B. CLEVENGER, JR.

```

IMPLICIT REAL*8 (A-H,O-Z)
REAL*8 LAMBDA
INTEGER RUNGE
DIMENSION V(21,21),BETA(21,21),R(21,21),RC(21,21),AL(21,21),
1  ANS(21  ),F(21,21),LAMBDA(21,21),RD(21),Z(21,21)
DIMENSION F1(21)
DIMENSION STATE(18),YR(6),YRS(6),VR(4),VU(4),VZ(4),RHOA(4),
1  TEMPA(4),VISTAR(4),ETA(4)
DIMENSION FR(6),RA(2),ZA(2)
PI=3.14159
CONV=PI/180.
JZ=1
READ(5,1000) MX,KMX,Z1,Z2,R1,R2
WRITE(6,2130) MX,KMX,Z1,Z2,R1,R2
READ(5,1005) GAMMA,CP,TIP,RHOIP,WTFI
WRITE(6,2140) GAMMA,CP,TIP,RHOIP,WTFI
READ(5,1010)      (RC(2,K),K=1,KMX)
WRITE(6,2170) KMX,(RC(2,K),K=1,KMX)
READ(5,1010)      (AL(I,1),I=1,MX)
WRITE(6,2160) MX,(AL(I,1),I=1,MX)
READ(5,1010)      (LAMBDA(1,K),K=1,KMX)
WRITE(6,2150)KMX,(LAMBDA(1,K),K=1,KMX)
DO 6 K=1,KMX
DO 6 I=1,MX
IF(I.EQ.1) RC(I,K)=-10000.
IF(I.EQ.MX) RC(I,K)=-10000.
IF((I.GT.1).AND.(I.LT.MX)) RC(I,K)=RC(2,K)
AL(I,K)=AL(I,1)
LAMBDA(I,K)=LAMBDA(1,K)
IF(I.EQ.1) R(I,K)=R2-RC(2,K)
IF(I.EQ.MX) R(I,K)=R1
IF((I.GT.1).AND.(I.LT.MX)) R(I,K)=R2-RC(2,K)*DCOS(AL(I,K)*CONV)
IF(I.EQ.1) Z(I,K)=Z1
IF(I.EQ.MX) Z(I,K)=Z2-RC(2,K)
IF((I.GT.1).AND.(I.LT.MX)) Z(I,K)=Z2+RC(2,K)*DSIN(AL(I,K)*CONV)

```

6 CONTINUE

DO 100 I=1, MX

WRITE(6,2060) I

WRITE(6,2010)

WRITE(6,2000) (R(I,K), K=1, KMX)

WRITE(6,2040)

WRITE(6,2000) (Z(I,K), K=1, KMX)

WRITE(6,2020)

WRITE(6,2000) (RC(I,K), K=1, KMX)

WRITE(6,2030)

WRITE(6,2000) (AL(I,K), K=1, KMX)

WRITE(6,2050)

WRITE(6,2000) (LAMBDA(I,K), K=1, KMX)

XLENTH=DSQRT((R(I,KMX)-R(I,1))**2+(Z(I,KMX)-Z(I,1))**2)

VCR=DSQRT(2.0*CP*TIP*(GAMMA-1.0)/(GAMMA+1.0))

IND=1

NN=KMX-1

DELTAN=XLENTH/D=LOAT(KMX-1)

VM=WTFL/RHOIP/XLENTH/PI/(R(I,1)+R(I,KMX))

VEST=DSQRT(VM**2+(LAMBDA(I,1)/R(I,1))**2)

DELTAY=VEST/20.

10 V(I,1)=VEST

IF((V(I,1)**2).GT.2.0*CP*TIP) GO TO 45

RHO=RHOIP*(1.-(V(I,1)**2)/(2.0*CP*TIP))**(1./(GAMMA-1.))

F(I,1)=RHO*VM*R(I,1)

VA=LAMBDA(I,1)/R(I,1)

A=1./RC(I,1)

B=-(VA**2)*(A+DCOS(AL(I,1)*CONV)/R(I,1))

DO 20 K=1, NN

V1STAR=V(I,K)+(A*V(I,K)+B/V(I,K))*DELTAN

A=1./RC(I,K+1)

VA=LAMBDA(I,K+1)/R(I,K+1)

B=-(VA**2)*(A+DCOS(AL(I,K+1)*CONV)/R(I,K+1))

V2STAR=V(I,K)+(A*V1STAR+B/V1STAR)*DELTAN

V(I,K+1)=(V1STAR+V2STAR)/2.

VM2=V(I,K+1)**2-VA**2

VM=0.0

IF(VM2.GT.0.0) VM=DSQRT(VM2)

IF((V(I,K+1)**2).GT.2.0*CP*TIP) GO TO 46

RHO=RHOIP*(1.-(V(I,K+1)**2)/(2.0*CP*TIP))**(1./(GAMMA-1.))

20 F(I,K+1)=RHO*VM*R(I,K+1)

DO 30 K=1, KMX

30 F1(K)=F(I,K)

CALL FNTGRL(DELTAN, F1, KMX, ANS)

WTFLES=ANS(KMX)*2.*PI

IF(DABS(WTFLES-WTFL).LE..00001 *WTFL) GO TO 50

CALL CONTIN(VEST, WTFLES, IND, JZ, WTFL, DELTAY)

IF(IND.LT.10) GO TO 10

IF(IND.NE.10) GO TO 40

WRITE(6,2110) WTFLES

GO TO 50

40 WRITE(6,2070)

GO TO 100

```

45 WRITE(6,2080) V(I,1)
   GO TO 100
46 WRITE(6,2081) V(I,K+1)
   GO TO 100
50 DO 60 K=1,KMX
   BETA(I,K)=DARSIN(LAMBDA(I,K)/V(I,K)/R(I,K))/CONV
60 V(I,K)=V(I,K)/VCR
   WRITE(6,2120) WTFLES
   WRITE(6,2090) (BETA(I,K),K=1,KMX)
   WRITE(6,2100) (V(I,K),K=1,KMX)
100 CONTINUE
105 READ(5,3000) (STATE(I),I=1,18)
   WRITE(6,4000)(STATE(I),I=1,18)
   READ(5,3010) VISREF,TREF,TSUT,RAIR,DGFC
   WRITE(6,4010) VISREF,TREF,TSUT,RAIR,DGFC
   READ(5,3020) RHOP,DIAP,H,TMAX,ETA(1),ETA(2)
   WRITE(6,4020) RHOP,DIAP,H,TMAX,ETA(1),ETA(2)
   T=0.0
   READ(5,3030) (YR(I),I=1,6)
   WRITE(6,4030)(YR(I),I=1,6)
   READ(5,3080) NSTEP
   WRITE(6,4080) NSTEP
   RHOCR=RHOIP*(2.0/(GAMMA+1.0))*(1.0/(GAMMA-1.0))
   DELTA=RHOP*DIAP/RHOCR/0.3
   TCR=(2.0/(GAMMA+1.0))*TIP
   VISCR=VISREF*((TCR/TREF)**1.5)*(TREF+TSUT)/(TCR+TSUT)
   TAU=RHOP*DIAP**2/18.0/VISCR
   RECR=DIAP*RHOCR*VCR/VISCR/2.0
   WRITE(6,5020) DELTA,TAU,RECR
   WRITE(6,4090)
   DO 110 I=1,MX
   DO 110 K=1,KMX
   AL(I,K)=AL(I,K)*CONV
   BETA(I,K)=BETA(I,K)*CONV
   V(I,K)=V(I,K)*VCR
110 CONTINUE
C
C DETERMINE AIR VELOCITIES AND PROPERTIES AT PARTICLE LOCATION.
C SET UP TO START TRACE AND INITIALIZE.
C
   CALL BLOCAT(R,Z,MX,KMX,YR,IP,KP,NOUT)
   IF(NOUT.EQ.0) GO TO 130
   WRITE(6,4040) NOUT
   GO TO 105
130 CALL POLATE(R,Z,V,YR(1),YR(5),IP,KP,VP,DD)
   CALL POLATE(R,Z,BETA,YR(1),YR(5),IP,KP,BETAP,DD)
   CALL POLATE(R,Z,AL,YR(1),YR(5),IP,KP,ALPP,DD)
   VR(1)=VP*DCOS(BETAP)*DSIN(ALPP)
   VJ(1)=VP*DSIN(BETAP)
   VZ(1)=VP*DCOS(BETAP)*DCOS(ALPP)
   TEMPA(1)=TIP*(1.-VP**2/CP/2./TIP)
   RHOA(1)=RHOIP*(TEMPA(1)/TIP)**(1./GAMMA-1.)
   VISTAR(1)=VISREF*(TEMPA(1)/TREF)**1.5*(TREF+TSUT)/(TEMPA(1)+TSUT)

```

INITIALIZE FOR FIRST STEP.

RPART=DIAP/2.

V=6

M=1

VTIME=-1

T=0.0

TS=T

DO 610 I=1,4

VR(I)=VR(1)

VU(I)=VU(1)

VZ(I)=VZ(1)

RHOA(I)=RHOA(1)

TEMPA(I)=TEMPA(1)

610 VI STAR(I)=VI STAR(1)

DO 620 I=1,N

620 YRS(I)=YR(I)

INTEGRATE USING RUNGE-KUTTA METHOD.

625 VDIFF=DSQRT((VR(2)-YR(2))**2+(VU(2)-YR(1)*YR(4))**2+(VZ(2)-YR(6))
1 **2)

RENOLD=RHOA(2)*VDIFF*DIAP/VI STAR(2)

CALL RNUMBR(RENOLD,DGFC,CD)

BCON=RHOA(2)*CD/RHOP/RPART/2.33333

630 IF(RUNGE(N,YR,FR,T,H).NE.1) GO TO 640

FR(1)=YR(2)

FR(2)=YR(1)*YR(4)**2+BCON*VDIFF*(VR(2)-YR(2))

FR(3)=YR(4)

FR(4)=-2.0*YR(2)*YR(4)/YR(1)+BCON*VDIFF*(VU(2)-YR(1)*YR(4))/YR(1)

FR(5)=YR(6)

FR(6)=BCON*VDIFF*(VZ(2)-YR(6))

GO TO 630

640 CONTINUE

DETERMINE IF WALL INTERACTION OCCURRED.

CALL BLOCAT(R,Z,MX,KMX,YR,IP,KP,NOUT)

IF(NOUT.EQ.0) GO TO 700

IF((NOUT.EQ.1).OR.(NOUT.EQ.3)) GO TO 780

DO 645 I=1,6

645 YR(I)=YRS(I)

T=TS

IF(NOUT.NE.2) GO TO 650

RA(1)=R(IP,1)

RA(2)=R(IP-1,1)

ZA(1)=Z(IP,1)

ZA(2)=Z(IP-1,1)

WRITE(6,5000) NOUT

CALL RBCH(YR,RA,ZA,T,H,ETA,NOUT)

GO TO 750

650 CONTINUE

```

RA(1)=R(IP,KMX)
RA(2)=R(IP-1,KMX)
ZA(1)=Z(IP,KMX)
ZA(2)=Z(IP-1,KMX)
WRITE(6,5000) NOUT
CALL RRCH(YR,RA,ZA,T,H,FTA,NOUT)
GO TO 750

```

DETERMINE AIR VALUES AT NEW LOCATION.

```

700 CALL POLATE(R,Z,V,YR(1),YP(5),IP,KP,VP,DD)
CALL POLATE(R,Z,BETA,YR(1),YP(5),IP,KP,BETAP,DD)
CALL POLATE(R,Z,AL,YR(1),YP(5),IP,KP,ALPP,DD)
VR(4)=VP*DCOS(BETAP)*DSIN(ALPP)
VU(4)=VP*DSIN(BETAP)
VZ(4)=VP*DCOS(BETAP)*DCOS(ALPP)
TEMPA(4)=1.0-(VP**2)/2.0/CP/TIP
RHOA(4)=RHOIP*(TEMPA(4)/TIP)**(1.0/(GAMMA-1.0))
VISTAR(4)=VISREF*(TEMPA(4)/TRFF)**1.5*(TRFF+TSUT)/(TEMPA(4)+TSUT)

```

TEST AIR VALUES USED, IF INCORRECT, RESET INTEGRATION VALUES AND GO TO 750.

```

IF((DABS(VR(4)-VR(3)).LT.1.0E-4).AND.(DABS(VU(4)-VU(3)).LT.1.0E-4)
1 .AND.(DABS(VZ(4)-VZ(3)).LT.1.0E-4)) GO TO 750

```

```

VR(2)=(VR(4)+VR(1))/2.0
VU(2)=(VU(4)+VU(1))/2.0
VZ(2)=(VZ(4)+VZ(1))/2.0
VR(3)=VR(4)
VU(3)=VU(4)
VZ(3)=VZ(4)
RHOA(2)=(RHOA(4)+RHOA(1))/2.0
TEMPA(2)=(TEMPA(4)+TEMPA(1))/2.0
VISTAR(2)=(VISTAR(4)+VISTAR(1))/2.0
T=TS
DO 710 I=1,N
710 YR(I)=YRS(I)
IF(M.GT.100) GO TO 800
M=M+1
GO TO 625

```

COMPLETE INTEGRATION STEP, IF REQUIRED, WRITE OUTPUT.

```

750 M=1
L=L+1
TS=T
DO 760 I=1,N
760 YRS(I)=YR(I)
VR(2)=1.5*VR(4)-0.5*VR(1)
VR(3)=2.0*VR(4)-VR(1)
VR(1)=VR(4)
VR(4)=VR(3)
VU(2)=1.5*VU(4)-0.5*VU(1)

```

```

VJ(3)=2.0*VU(4)-VU(1)
VU(1)=VU(4)
VU(4)=VU(3)
VZ(2)=1.5*VZ(4)-0.5*VZ(1)
VZ(3)=2.0*VZ(4)-VZ(4)
VZ(1)=VZ(4)
VZ(4)=VZ(3)
RHJA(2)=1.5*RHOA(4)-0.5*RHOA(1)
RHOA(3)=2.0*RHJA(4)-RHOA(1)
RHOA(1)=RHOA(4)
RHJA(4)=RHOA(3)
TEMPA(2)=1.5*TEMPA(4)-0.5*TEMPA(1)
TEMPA(3)=2.0*TEMPA(4)-TEMPA(1)
TEMPA(1)=TEMPA(4)
TEMPA(4)=TEMPA(3)
VISTAR(2)=1.5*VISTAR(4)-0.5*VISTAR(1)
VISTAR(3)=2.0*VISTAR(4)-VISTAR(1)
VISTAR(1)=VISTAR(4)
VISTAR(4)=VISTAR(3)
NTIME=NTIME-1
780 IF((NOUT.EQ.1).OR.(NOUT.EQ.3).OR.(T.GT.TMAX)) GO TO 800
IF(NTIME.GT.0) GO TO 625
WRITE(6,4050) T,(YR(I),I=1,6),RENOLD
VTIME=NSTEP-1
GO TO 625
800 CONTINUE
WRITE(6,4060) T,(YR(I),I=1,6),RENOLD
WRITE(6,4070) M
805 CONTINUE
GO TO 105

C
C
C
FORMAT STATEMENTS
1000 FORMAT(2I5,6F10.5)
1005 FJRMAT(8F10.5)
1010 FORMAT(8F10.5)
1015 FJRMAT(8E10.4)
2000 FJRMAT(1H ,8G16.7)
2010 FJRMAT(9H R ARRAY)
2020 FJRMAT(9HORC ARRAY)
2030 FJRMAT(12HOALPHA ARRAY)
2040 FJRMAT(8HOZ ARRAY)
2050 FJRMAT(7HOLAMBDA)
2060 FJRMAT(1H1,25H THE ORTHOGONAL NUMBER IS,13///)
2070 FORMAT(46HOA SOLUTION CANNOT BE OBTAINED AT THIS STATION)
2080 FJRMAT(1H0,43H AFTER 10, V(I,1) .GT. 2. CP*CP. E(I,1) =,F15.2)
2081 FORMAT(1H0,49H DJ 20 LOOP, V(I,K+1) .GT. 2.*CP*TIP. V(I,K+1) =,
1 F15.2)
2090 FJRMAT(11HOBETA ARRAY/(1X,8G16.7))
2100 FJRMAT(8HOV ARRAY/(1X,8G16.7))
2110 FORMAT(42HOTHE PASSAGE IS CHOKED WITH A MASS FLOW OF,G16.7)
2120 FJRMAT(7H0WTFLES/(1X,8G16.7))
2130 FJRMAT(1H1,114 INPUT DATA,/,8X,2HMX,7X,3HKMX, 8X,2HZ1,13X,2HZ2,

```

```

1 13X,2HR1,13X,2HR2,/, (2F10,4F15.5)
2140 FORMAT(140,10X,5HGAMMA,13X,2HCP,12X,3HTIP,10X,5HRHOIP,11X,
1 4HWTFL,/, (F15.4,2F15.3,2E15.5))
2150 FORMAT(1H0,5X,14HLAMBDA, K = 1.,13,/, (8G15.5))
2160 FORMAT(1H0,5X,13HALPHA, I = 1.,13,/, (8G15.5))
2170 FORMAT(1H0,5X,13HRC, K = 1.,13,/, (8G15.5))
3000 FORMAT(18A4)
3010 FORMAT(E20.5,4F10.3)
3020 FORMAT(F10.2,3F10.4,2F10.3)
3030 FORMAT(6F10.3)
3080 FORMAT(15)
4000 FORMAT(1H1,18A4)
4010 FORMAT(1H0,13X,5HVISREF,11X,4HTREF,11X,4HTSUT,11X,4HRAIR,11X,
1 4HOGFC,/, (E20.5,4F15.3))
4020 FORMAT(1H0,10X,4HRHOP,11X,4HDIAP,14X,1HH,11X,4HTMAX,10X,5HETA-N,
1 10X,5HETA-T,/, (F15.2,3E15.4,2F15.3))
4030 FORMAT(1H0, 9X,5HYR(1),10X,5HYR(2),10X,5HYR(3),10X,5HYR(4),10X,
1 5HYR(5),10X,5HYR(6),/(6F15.3))
4040 FORMAT(1H0,62H PARTICLE NOT IN PASSAGE AT FIRST POINT GIVEN, GO TO
1 NEXT CASE)
4050 FORMAT(1H ,E15.2,7F15.4)
4060 FORMAT(1H0,E15.2,7F15.4)
4070 FORMAT(140,4H M =,110)
4080 FORMAT(17HOPRINT DATA EVERY,17,2X,7HSTEP(S))
4090 FORMAT(1H0,13X,1HT,10X,5HYR(1),10X,5HYR(2),10X,5HYR(3),10X,
1 5HYR(4),10X,5HYR(5),10X,5HYR(6),9X,6HRENOID)
5000 FORMAT(29H PARTICLE BOUNCED OFF SURFACE,15)
5020 FORMAT(32HOSIMILARITY PARAMETERS. DELTA =,F10.4,5X,5HTAU =,E12.4,
1 5X,6HRECR =,E12.4)
END

```

```

SUBROUTINE PARC(X,Y,A,B,C)
IMPLICIT REAL*8 (A-H,C-Z)

```

```

SUBROUTINE PARC CALCULATES COEFFICIENTS A, B, C OF THE PARABOLA
Y = A*X**2 + B*X + C. PASSING THROUGH THREE GIVEN X,Y POINTS

```

```

DIMENSION X(3),Y(3)
C1=X(2)-X(1)
C2=(Y(2)-Y(1))/(X(2)-X(1))
A=(C1*C2-Y(3)+Y(1))/C1/(X(2)-X(3))
B=C2-(X(1)+X(2))*A
C=Y(1)-X(1)*B-X(1)**2*A
RETURN
END

```

The function routine RUNGE has been removed from the published form of this report to protect the copyright of the authors of Reference 5.

```

SURROUTINE CONTIN(XEST,YCALC,IND,JZ,YGIV,XDEL)
IMPLICIT REAL*8 (A-H,O-Z)
DIMENSION X(3),Y(3)
NCALL=NCALL+1
IF(IND.NE.1.AND.NCALL.GT.50) GO TO 160
GO TO (10,30,40,50,60,80,130),IND

```

C*****FIRST CALL

```

10 NCALL=1
IF(YCALC.GT.YGIV.AND.JZ.EQ.1) GO TO 20
IND=2
Y(1)=YCALC
X(1)=XEST
XEST=XEST+XDEL
RETURN

```

```

20 IND=3
Y(3)=YCALC
X(3)=XEST
XEST=XEST-XDEL
RETURN

```

C*****SECOND CALL

```

30 IND=4
Y(2)=YCALC
X(2)=XEST
XEST=XEST+XDEL
RETURN

```

```

40 IND=5
Y(2)=YCALC
X(2)=XEST
XEST=XEST-XDEL
RETURN

```

C*****THIRD OR LATER CALL - FIND SUBSONIC OR SUPERSONIC SOLUTION

```

50 Y(3)=YCALC
X(3)=XEST
GO TO 70
60 Y(1)=YCALC
X(1)=XEST
70 IF(YGIV.LT.DMINV(Y(1),Y(2),Y(3))) GO TO (90,95),JZ
75 IND=6

```

```

CALL PARC(X,Y,APA,BPB,CPC)
DISCR=BPB**2-4.*APA*(CPC-YGIV)
IF(DISCR.LT.0.0) GO TO 110
IF(DABS(400.*APA*(CPC-YGIV)).LF.BPB**2) GO TO 78
XEST=-BPB-DSIGN(DSQRT(DISCR),APA)
IF(JZ.EQ.2.AND.APA.LT.0.0) XEST=-BPB-DSQRT(DISCR)
XEST=XEST/2./APA
GO TO 79

```

```

78 ACB2=APA/BPB*(CPC-YGIV)/BPB
IF(DABS(ACB2).LE.1.E-8) ACB2=0.0
XEST=-(CPC-YGIV)/BPB*(1.+ACB2+2.*ACB2**2)
79 IF(XEST.GT.X(3)) GO TO 95
IF(XEST.LT.X(1)) GO TO 90
RETURN

```

C*****FOURTH OR LATER CALL - (NOT CHOKED)


```

80 IF(XEST.GT.X(3)) GO TO 50
   IF(XFST.LT.X(1)) GO TO 60
   Y(2)=YCALC
   X(2)=XEST
   GO TO 70
C*****THIRD OR LATER CALL / SOLUTION EXISTS,
C*****BJT RIGHT OR LEFT SHIFT REQUIRED.
90 IND=5
C*****LEFT SHIFT
   Y(3)=Y(2)
   X(3)=X(2)
   Y(2)=Y(1)
   X(2)=X(1)
   XEST=X(1)-XDEL
   RETURN
95 IND=4
C*****RIGHT SHIFT
   Y(1)=Y(2)
   X(1)=X(2)
   Y(2)=Y(3)
   X(2)=X(3)
   XEST=X(3)+XDEL
   RETURN
C*****THIRD OR LATER CALL - APPEARS TO BE CHOKED
110 XEST=-8PB/2./APA
    IND=7
    IF(X(1).LE.XEST.AND.XEST.LE.X(3)) RETURN
    IF(XFST.LT.X(1)) GO TO 90
    GO TO 95
C*****FOURTH OR LATER CALL - PROBABLY CHOKED
130 IF(YCALC.GE.YGIV) GO TO 80
    IND=10
    RETURN
C*****NO SOLUTION FOUND IN 50 ITERATIONS
160 IND=11
    RETURN
    END

```

```

SUBROUTINE FNTGRL(DX,F,N,ANS)
IMPLICIT REAL*8 (A-H,O-Z)
DIMENSION F(50),ANS(50)
DO 10 I=1,N
IF(I.EQ.1) ANS(1)=0.0
IF(I.EQ.2) ANS(2)=DX*(F(1)+F(2))/2.0
IF(I.EQ.3) ANS(3)=DX*(F(1)+4.0*F(2)+F(3))/3.0
IF(I.EQ.4) ANS(4)=3.0*DX*(F(1)+3.0*F(2)+3.0*F(3)+F(4))/8.0
IF(I.GT.4) ANS(I)=ANS(I-2)+DX*(F(I-2)+4.0*F(I-1)+F(I))/3.0
10 CONTINUE
RETURN
END

```

```
SUBROUTINE RLOCAT(R,7,MX,KMX,YR,IP,KP,NOUT)
  IMPLICIT REAL*8 (A-H,O-Z)
  DIMENSION R(21,21),Z(21,21),YR(6)
  NOUT=0
  DO 20 I=1,MX
    IF(DABS(Z(I,1)-Z(I,KMX)).LT.1.0E-12) GO TO 10
    A=(R(I,1)-R(I,KMX))/(Z(I,1)-Z(I,KMX))
    R=R(I,1)-A*Z(I,1)
    RTEST=A*YR(5)+R
    IF(RTEST.LE.YR(1)) GO TO 30
    GO TO 20
  10 IF(Z(I,1).GE.YR(5)) GO TO 30
  20 CONTINUE
  30 IP=I
    IF(IP.NE.1) GO TO 40
    NOUT=1
    RETURN
  40 IF(IP.NE.MX) GO TO 50
    IF(R(MX,1).LT.YR(1)) GO TO 50
    NOUT=3
    RETURN
  50 CONTINUE
    DO 70 K=1,KMX
      IF(DABS(Z(IP,K)-Z(IP-1,K)).LT.1.0E-12) GO TO 60
      A=(R(IP-1,K)-R(IP,K))/(Z(IP-1,K)-Z(IP,K))
      B=R(IP,K)-A*Z(IP,K)
      RTEST=A*YR(5)+B
      IF(YR(1).LE.RTEST) GO TO 80
      GO TO 70
    60 IF(Z(IP,K).GE.YR(5)) GO TO 80
    70 CONTINUE
    80 KP=K
      IF(KP.NE.1) GO TO 90
      NOUT=2
      RETURN
    90 IF(KP.NE.KMX) GO TO 100
      IF(DABS(Z(IP,KP)-Z(IP-1,KP)).LT.1.0E-12) GO TO 110
      IF(YR(1).LT.RTEST) GO TO 100
      NOUT=4
    100 RETURN
    110 IF(YR(5).GE.Z(IP,KP)) NOUT=4
      RETURN
  END
```

```

SUBROUTINE RBCH(YR,R,Z,T,H,ETA,NOUT)
IMPLICIT REAL*8 (A-H,O-Z)
DIMENSION YR(5),R(2),Z(2),ETA(2)
IF(DABS(Z(2)-Z(1)).LT.1.0E-12) GO TO 100
IF(DABS(R(2)-R(1)).LT.1.0E-12) GO TO 200
SM=(R(2)-R(1))/(Z(2)-Z(1))
IF(DABS(YR(6)).LE.1.0E-12) GO TO 10
IF(DABS(YR(2)).LE.1.0E-12) GO TO 20
PM=YR(2)/YR(6)
ZB=(Z(1)*SM-YR(5)*PM+YR(1)-R(1))/(SM-PM)
RB=SM*(ZB-Z(1))+R(1)
GO TO 30
10 ZB=YR(5)
RB=SM*(ZB-Z(1))+R(1)
GO TO 30
20 RB=YR(1)
ZB=(RB-R(1))/SM+Z(1)
GO TO 30
100 IF(DABS(YR(2)).LT.1.0E-12) GO TO 110
PM=YR(2)/YR(6)
RB=PM*(Z(1)-YR(5))+YR(1)
ZB=Z(1)
GO TO 30
110 RB=YR(1)
ZB=Z(1)
GO TO 30
200 IF(DABS(YR(6)).LT.1.0E-12) GO TO 250
PM=YR(2)/YR(6)
RB=R(1)
ZB=(RB-YR(1))/PM+YR(5)
GO TO 30
250 RB=R(1)
ZB=YR(5)
30 DBP=DSQRT((RB-YR(1))**2+(ZB-YR(5))**2)
VM=DSQRT(YR(2)**2+YR(6)**2)
DT=H-DBP/VM
TB=YR(3)+YR(4)*DBP/VM
GAMMA=DATAN2((R(2)-R(1)),(Z(2)-Z(1))) -1.5707963
ALPHA=DATAN2(YR(2),YR(6))
VN=VM*DCOS(GAMMA-ALPHA)
VTR=VM*DSIN(GAMMA-ALPHA)
VNP=-VN*ETA(1)
VTRP=VTR*ETA(2)
YR(4)=YR(4)*ETA(2)
VMP=DSQRT(VNP**2+VTRP**2)
BETA=DATAN2(VTRP,-VNP)
YR(2)=-VMP*DSIN(GAMMA+BETA)
YR(6)=-VMP*DCOS(GAMMA+BETA)
YR(1)=RB+YR(2)*DT
YR(3)=TB+YR(4)*DT
YR(5)=ZB+YR(5)*DT
T=T+H
RETURN

```

END

```

SUBROUTINE SPLINT (X,Y,N,Z,MAX,YINT)
  IMPLICIT REAL*8 (A-H,O-Z)
  DIMENSION X(50),Y(50),S(50),A(50),B(50),C(50),F(50),W(50),SB(50),
  1G(50),EM(50),Z(50),YINT(50)
  COMMON Q
  INTEGER Q
  DO 10 I=2,N
  S(I)=X(I)-X(I-1)
10 CONTINUE
  NO=N-1
  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)
  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 30 I=2,N
  W(I)=B(I)-A(I)*SB(I-1)
  SB(I)=C(I)/W(I)
30 G(I)=(F(I)-A(I)*G(I-1))/W(I)
  EM(N)=G(N)
  DO 40 I=2,N
  K=N+1-I
40 EM(K)=G(K)-SB(K)*EM(K+1)
  DO 90 I=1,MAX
  K=2
  IF(Z(I)-X(1)) 60,50,70
50 YINT(I)=Y(1)
  GO TO 90
60 IF(Z(I).LT.(1.1*X(1)-.1*X(2)))WRITE (6,1000)Z(I)
  GO TO 85
1000 FORMAT (17H OUT OF RANGE Z #F10.6)
65 IF(Z(I).GT.(1.1*X(N)-.1*X(N-1))) WRITE (6,1000)Z(I)
  K=N
  GO TO 85
70 IF(Z(I)-X(K)) 85,75,80
75 YINT(I)=Y(K)
  GO TO 90
80 K=K+1
  IF(K=N) 70,70,65
85 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))
90 CONTINUE
  MAX = MAX0(N,MAX)

```

Example

The example case presented here used the ft., slug, second system of units. The gas flow conditions correspond to inlet stagnation conditions of standard sea level air. The output that describes the mass flow through the turning vortex is contained on the first 5 pages of the output here. The output variables are the gas angle with respect to the meridional plane, β , and the velocity in terms of V/V_{cr} .

The particle used in the example has a specific gravity of 3 and a diameter of approximately 24 microns. Initially the particle has a velocity in the tangential direction with a velocity of 1000 rad/sec. The trajectory data indicates that this particle moves outward until it strikes the outer surface, where it bounces. The bounce drives the particle back into the inlet of the turning vortex.

The following pages contain a computer code sheet with the data arranged in the proper columns, and the output for this example.


```

IF(O.EQ.16) WRITE(6,1010) N,MAX,(X(I),Y(I),Z(I),YINT(I),I=1,MAX)
1010 FORMAT (2X21HNO. OF POINTS GIVEN #,I3,30H, NO. OF INTERPOLATED POI
INTS #,I3,/10X5HX 15X5HY 12X11HX-INTERPOL.9X11HY-INTERPOL./14
2E20.8))
100 RETURN
END

```

0460
0470

```

SUBROUTINE POLATE(R,Z,A,RP,ZP,IP,KP,AP,DD)
IMPLICIT REAL*8 (A-H,O-Z)
DIMENSION R(21,21),Z(21,21),A(21,21),D(2),DD(2)
DO 10 I=1,2
IA=IP+1-I
IF(DABS(Z(IA,KP-1)-Z(IA,KP))).LT.1.0E-12) GO TO 5
IF(DABS(R(IA,KP)-R(IA,KP-1))).LT.1.0E-12) GO TO 6
AM=(R(IA,KP)-R(IA,KP-1))/(Z(IA,KP)-Z(IA,KP-1))
B1=R(IA,KP-1)-AM*Z(IA,KP-1)
B2=RP+ZP/AM
ZA=(B2-B1)*AM/(AM**2+1.0)
RA=B2-ZA/AM
GO TO 10
5 RA=RP
ZA=Z(IA,KP-1)
GO TO 10
6 RA=R(IA,KP-1)
ZA=ZP
10 D(I)=DSQRT((RA-RP)**2+(ZA-ZP)**2)
DT=D(1)+D(2)
AA=(D(1)*A(IP,KP-1)+D(2)*A(IP-1,KP-1))/DT
AB=(D(1)*A(IP,KP)+D(2)*A(IP-1,KP))/DT
DO 20 K=1,2
KA=KP-K+1
RC=(D(1)*R(IP,KA)+D(2)*R(IP-1,KA))/DT
ZC=(D(1)*Z(IP,KA)+D(2)*Z(IP-1,KA))/DT
20 DJ(K)=DSQRT((RP-RC)**2+(ZP-ZC)**2)
DT=DD(1)+DD(2)
AP=(DD(1)*AA+DD(2)*AB)/DT
RETURN
END

```

```

SUBROUTINE RNUMBR(RENOLD,DGFC,CD)
IMPLICIT REAL*8 (A-H,O-Z)
IF(DABS(RENOLD).LT.1.0E-12) RENOLD=1.0E-12
IF(RENOLD.LT.1.0) GO TO 26
IF((RENOLD.GE.1.0).AND.(RENOLD.LT.1.0E3)) GO TO 27
CD=DGFC*0.4
RETURN
26 CD=DGFC*(4.5+24.0/RENOLD)
RETURN
27 ARE=DLOG(RENOLD)
CD=(28.5-24.0*ARE+9.0682*ARE**2-1.7713*ARE**3+0.1718*ARE**4
1 -0.0065*ARE**5)*DGFC
RETURN
END

```

REPORT DATE	MM	DD	71	72	R1	R2
10	7	0.0	0.90250	0.28500	0.28750	

GAMMA	CP	TIP	RHC10	WTPL
1.4000	6012.400	518.700	0.226000-02	0.156000-01

RC. K = 1. 7
 -0.146600-01 -0.211470-01 -0.258120-01 -0.302500-01 -0.347500-01 -0.392500-01 -0.433330-01

SIGMA. I = 1. 10
 0.0 -4.0000 -18.500 -30.000 -40.500 -50.260 -62.780 -75.000
 -86.000 -90.000

LAMBDA. K = 1. 7
 214.60 214.77 214.94 215.11 215.28 215.45 215.60

THE CORRELATIONAL NUMBER IS 1

0 ARRAY C.7041660	0.7506670	0.3133330	0.3177500	0.3222500	0.3267500	0.3308330
2 ARRAY 0.0	0.0	0.0	0.0	0.0	0.0	0.0
01 ARRAY -10000.00	-10000.00	-10000.00	-10000.00	-10000.00	-10000.00	-10000.00
SIGMA ARRAY 0.0	0.0	0.0	0.0	0.0	0.0	0.0
1 ARRAY 214.6000	214.7700	214.9400	215.1100	215.2800	215.4500	215.6000
MTIPS 0.1599040-01						
0216 ARRAY 76.88497	76.89091	76.88843	76.72123	76.69327	76.66794	76.63442
0 ARRAY 0.7109197	0.7008106	0.6913817	0.6822178	0.6733004	0.6646227	0.6561600

THE CORRELATIONAL NUMBER IS 2

0 ARRAY 0.7041676	0.7095928	0.3132424	0.3176431	0.3221281	0.3266129	0.3306810
2 ARRAY 0.38469730-02	0.42712070-02	0.46616470-02	0.50312520-02	0.54078020-02	0.57843520-02	0.61260000-02
01 ARRAY -0.16666000-01	-0.21167000-01	-0.25833000-01	-0.30250000-01	-0.34750000-01	-0.39250000-01	-0.43333000-01
SIGMA ARRAY -4.800000	-4.800000	-4.800000	-4.800000	-4.800000	-4.800000	-4.800000
1 ARRAY 214.6000	214.7700	214.9400	215.1100	215.2800	215.4500	215.6000
MTIPS 0.15800000-01						
0216 ARRAY 77.81085	77.86740	77.89427	76.74401	76.36662	75.67638	81.16005
0 ARRAY 0.7109197	0.7102707	0.6980822	0.6823811	0.6652185	0.6476288	0.6471542

THE CATHODICAL NUMBER IS 3

B ARRAY 0.3013048	0.3075777	0.3119980	0.3161868	0.3204543	0.3247217	0.3285937
F ARRAY 0.77881940-02	0.92163200-02	0.10696420-01	0.12090460-01	0.13524320-01	0.14954190-01	0.16249750-01
GF ARRAY -0.16666000-01	-0.21167000-01	-0.25833000-01	-0.30250000-01	-0.34750000-01	-0.39250000-01	-0.43333000-01
ALPHA ARRAY -18.50000	-18.50000	-18.50000	-18.50000	-18.50000	-18.50000	-18.50000
L ARRAY 214.4000	214.7700	214.9400	215.1100	215.2800	215.4500	215.6000
WTPIFS 0.19999990-01						
BETA ARRAY 67.74031	71.81264	74.56205	76.72264	78.35637	79.67541	81.14913
V ARRAY 0.7498319	0.7708777	0.7889763	0.8055867	0.8227395	0.8414598	0.8512845

THE CATHODICAL NUMBER IS 4

B ARRAY 0.3019972	0.3058312	0.3098720	0.3136973	0.3175944	0.3214915	0.3250275
F ARRAY 0.10892960-01	0.13082490-01	0.15414490-01	0.17624980-01	0.19874980-01	0.22124980-01	0.24166480-01
GF ARRAY -0.16666000-01	-0.21167000-01	-0.25833000-01	-0.30250000-01	-0.34750000-01	-0.39250000-01	-0.43333000-01
BIGMA ARRAY -30.00000	-30.00000	-30.00000	-30.00000	-30.00000	-30.00000	-30.00000
L ARRAY 214.4000	214.7700	214.9400	215.1100	215.2800	215.4500	215.6000
WTPIFS 0.19999990-01						
BETA ARRAY 67.40406	71.71750	74.50976	76.68524	78.33824	79.67337	81.12980
V ARRAY 0.7919741	0.7999700	0.7899777	0.6911342	0.6788411	0.6681101	0.6584458

ORIGINAL PAGE IS
OF POOR QUALITY

142

THE ORTHOGONAL NUMBER IS 5

R ARRAY C.9000971	0.3074997	0.3070260	0.3103446	0.3137655	0.3171673	0.3202934
Z ARRAY 0.19411400-01	0.16358250-01	0.15412400-01	0.22305890-01	0.25252220-01	0.28198550-01	0.30871840-01
BC ARRAY -0.14444000-01	-0.21167000-01	-0.25833000-01	-0.30250000-01	-0.34750000-01	-0.39250000-01	-0.43333000-01
BIGMA ARRAY -40.50000	-40.50000	-40.50000	-40.90000	-40.50000	-40.50000	-40.50000
SIGMA 214.6000	214.7700	214.9400	215.1100	215.2800	215.4500	215.6000
WTPIFC 0.19999500-01						
OPTA ARRAY 67.41767	71.58450	74.42750	76.43336	78.31270	79.66985	81.10263
V ARRAY 0.7591107	0.7219070	0.7127509	0.6987056	0.6871874	0.6772267	0.6683304

THE ORTHOGONAL NUMBER IS 6

R ARRAY 0.3001947	0.3010772	0.3040152	0.3068390	0.3097159	0.3125927	0.3152030
Z ARRAY 0.19315370-01	0.18776470-01	0.22264360-01	0.25760820-01	0.29221110-01	0.32681400-01	0.35821030-01
BC ARRAY -0.14444000-01	-0.21167000-01	-0.25833000-01	-0.30250000-01	-0.34750000-01	-0.39250000-01	-0.43333000-01
BIGMA ARRAY -50.26000	-50.26000	-50.26000	-50.26000	-50.26000	-50.26000	-50.26000
SIGMA 214.6000	214.7700	214.9400	215.1100	215.2800	215.4500	215.6000
WTPIFC 0.19999570-01						
OPTA ARRAY 67.21461	71.44451	74.34251	76.57611	78.28482	79.66507	81.07221
V ARRAY 0.7256517	0.7201147	0.7201501	0.7069619	0.6942451	0.6871478	0.6758954

ORIGINAL PAGE IS
OF POOR QUALITY

THE CRYSTAL NUMBER IS 7

B ARRAY	0.2448842	0.2565725	0.2589456	0.3009019	0.3028961	0.3048906	0.3066597
Z ARRAY	0.17440870-01	0.21475520-01	0.25658520-01	0.29618700-01	0.33652820-01	0.37687070-01	0.41347420-01
BC ARRAY	-0.16666000-01	-0.21167000-01	-0.25833000-01	-0.30250000-01	-0.34750000-01	-0.39250000-01	-0.43333000-01
SIGMA ARRAY	-63.70000	-63.70000	-63.70000	-63.70000	-63.70000	-63.70000	-63.70000
L SPREAD	214.8000	214.7700	214.9400	215.1100	215.2800	215.4500	215.6000
NTPIFS	0.15555550-01						
BETA ARRAY	66.89462	71.19474	74.15022	76.47379	78.23152	79.65428	81.01676
V ARRAY	0.7762187	0.7455475	0.7229023	0.7211558	0.7120555	0.7045307	0.6980300

THE CRYSTAL NUMBER IS 8

B ARRAY	0.2518175	0.2929785	0.2541261	0.2553293	0.2564960	0.2976587	0.2927155
Z ARRAY	0.18998110-01	0.22945740-01	0.27452750-01	0.31719240-01	0.36065510-01	0.40412570-01	0.44356450-01
BC ARRAY	-0.16666000-01	-0.21167000-01	-0.25833000-01	-0.30250000-01	-0.34750000-01	-0.39250000-01	-0.43333000-01
SIGMA ARRAY	-75.00000	-75.00000	-75.00000	-75.00000	-75.00000	-75.00000	-75.00000
L SPREAD	214.8000	214.7700	214.9400	215.1100	215.2800	215.4500	215.6000
NTPIFS	0.15555550-01						
BETA ARRAY	66.90581	70.54655	74.03871	76.37034	78.17692	79.64061	80.95943
V ARRAY	0.7806575	0.7608274	0.7453253	0.7350864	0.7275774	0.7216794	0.7168810

144

THE CRITICAL NUMBER IS 9

B ARRAY	0.2886313	0.2886447	0.2886571	0.2891311	0.2893793	0.2896227	0.2898435
Z ARRAY	0.19141610-01	0.23636020-01	0.28295200-01	0.32705730-01	0.37199150-01	0.41692560-01	0.45765590-01
BC ARRAY	-0.18866600-01	-0.21167000-01	-0.25833000-01	-0.30250000-01	-0.34750000-01	-0.39250000-01	-0.43230000-01
ALPHA ARRAY	-86.50000	-86.50000	-86.50000	-86.90000	-86.50000	-86.50000	-86.50000
I ARRAY	214.6700	214.7700	214.9400	215.1100	215.2800	215.4500	215.6000
WTWFS	0.15999910-01						
BETA ARRAY	75.10000	75.65531	75.85773	76.24603	76.10645	76.42079	80.88506
V ARRAY	0.7497961	0.7734400	0.7596654	0.7512209	0.7456455	0.7417505	0.7388870

THE CRITICAL NUMBER IS 10

B ARRAY	0.2850000	0.2850000	0.2850000	0.2850000	0.2850000	0.2850000	0.2850000
Z ARRAY	0.15188000-01	0.23647000-01	0.28333000-01	0.32750000-01	0.37250000-01	0.41750000-01	0.45823000-01
BC ARRAY	-10000.00	-10000.00	-10000.00	-10000.00	-10000.00	-10000.00	-10000.00
ALPHA ARRAY	-90.00000	-90.00000	-90.00000	-90.00000	-90.00000	-90.00000	-90.00000
I ARRAY	214.7000	214.7700	214.9400	215.1100	215.2800	215.4500	215.6000
WTWFS	0.15999910-01						
BETA ARRAY	75.97777	75.69550	75.87821	76.05980	76.24350	76.42950	76.59580
V ARRAY	0.7627417	0.7627417	0.7627416	0.7627416	0.7627416	0.7627415	0.7627415

EXAMPLE - TEST CASE

SIREFP	TRFF	TSST	RAIN	CGPC		
0.10000E-04	497.000	150.400	1710.400	1.000		
RMCP	RTAP	P	TRAX	ETA-N	ETA-T	
107.70	0.7760E-04	0.1000D-04	0.1000D-01	1.000	1.000	
YR(1)	YR(2)	YR(3)	YR(4)	YR(5)	YR(6)	
0.705	0.0	0.0	1000.000	0.001	0.0	

POINT DATA PUPRY 10 STEP(5)

STABILITY PARAMETERS. DELTA = 33.7500 TZU = 0.6554D-02 RECR = 0.5931D 01

T	YR(1)	YR(2)	YR(3)	YR(4)	YR(5)	YR(6)	RENOLD
0.100E-04	0.7750	2.5506	0.010	592.1252	0.0010	1.3275	14.0939
0.100E-03	0.7049	30.1415	0.0995	508.4179	0.0011	1.3310	0.0062
0.100E-01	0.7104	54.8850	0.1075	502.6432	0.0012	1.3346	0.0061
1.000E-01	0.7167	71.8210	0.2725	526.0020	0.0014	1.3380	0.0060
0.100E-01	0.7251	105.4130	0.3530	576.5317	0.0015	1.3414	0.0055
PARTICIP PRINCIP OFF SURFACE 4							
0.400E-01	0.7263	-110.3254	0.4213	567.0217	0.0014	-4.4910	0.0058
0.500E-01	0.7174	-87.2740	0.5116	516.3578	0.0010	-4.4876	0.0055
0.600E-01	0.7106	-67.4573	0.5959	556.6074	0.0006	-4.4841	0.0061
0.700E-01	0.7067	-26.2602	0.6834	584.6141	0.0002	-4.4806	0.0062
0.700E-01	0.7040	-21.3034	0.7329	592.6300	-0.0000	-4.4786	0.0063

P = 1

DISCUSSION OF RESULTS

The programs that are included here illustrate the general complexity of the problem of determining particle trajectories in a turbomachine. In tracing the particles all the way through a radial inflow turbine, the constantly changing nature of the flow requires that each region of the flow be considered individually. These programs are used to study the particle trajectories through each of these regions.

The programs have been developed over a three year period and some of the programs have subroutines that were stepping stones to the more complex routines that are explained in the section entitled, "General Numerical Techniques". Users of these programs might consider improvements by using the sophisticated subroutines instead, particularly the programs that do not allow variable particle restitution coefficients. The subroutine RESTCO, explained previously, can be used to describe in general the restitution coefficients. Experience has shown that the use of constant restitution coefficients less than 1.0 causes the particles to come to rest.

Several other programs, specifically the SCRL2D and STATOR programs could be improved with the addition of more realistic boundaries. In the scroll program, the solution of the gas flow is one-dimensional and the particle trajectories are two-dimensional. A better solution of the gas flow in this region might provide a slightly different particle trajectory pattern. In the stator program, the hub to shroud distance has been assumed constant, although in most real turbines, this distance is a function of the radius. Inclusion of this factor might lead to some interesting results in the study of particle trajectories in the radial turbine. These programs could be the basis of a very large program that could use a Monte Carlo Technique to study erosion rates from turbine internal surfaces.

CONCLUSIONS

This report has presented the computer programs that have been used to study the trajectories of particles in the radial inflow turbine. These programs can be used to investigate the trajectories of particles in radial inflow turbines, and provide information concerning the locations where particles strike the surfaces. This information can be used to predict the areas most subjected to erosion damage, in radial inflow turbines.

REFERENCES

1. Clevenger, W.B., and Tabakoff, W., "Erosion in Radial Inflow Turbines - Volume I: Erosive Particle Trajectory Similarity," NASA CR-134589, Lewis Research Center, 1974.
2. Clevenger, W.B., and Tabakoff, W., "Erosion in Radial Inflow Turbines - Volume II: Balance of Centrifugal and Radial Drag Forces on Erosive Particles," NASA CR-134616, Lewis Research Center, 1974.
3. Clevenger, W.B., and Tabakoff, W., "Erosion in Radial Inflow Turbines - Volume III: Trajectories of Erosive Particles in Radial Inflow Turbines," NASA CR- , Lewis Research Center, 1974.
4. Clevenger, W.B., and Tabakoff, W., "Erosion in Radial Inflow Turbines - Volume IV: Erosion Rates on Internal Surfaces," NASA CR-134677, Lewis Research Center, 1974.
5. Carnahan, B., Luther, H., and Wilkes, J.O., Applied Numerical Methods, Wiley and Sons, 1969.
6. Grant, G., and Tabakoff, W., "Erosion Prediction in Turbomachinery Due to Environmental Solid Particles," AIAA Paper No. 74-16, 1974.
7. Ball, R., and Tabakoff, W., "An Experimental Investigation of the Erosion Characteristics of 410 Stainless Steel and 6Al-4V Titanium," Report No. 73-40, Department of Aerospace Engineering, University of Cincinnati, 1973.
8. Durham, Franklin P., Aircraft Jet Powerplants, Prentice-Hall Inc., Englewood Cliffs, N.J., 1951.
9. Katsanis, T., "Use of Arbitrary Quasi-Orthogonals for Calculating Flow Distribution in the Meridional Plane of a Turbomachine," NASA TN D-2546, 1964.

Table 1. Coefficients of Polynomial, a_j ,
Describing Variation of
Tangential Restitution Coef-
ficient with Incidence Angle*

Coefficient of Number	Value of Coefficient
1	1.819024
2	5.117122
3	- 49.252858
4	131.035284
5	-174.424912
6	117.872348
7	- 24.478854
8	- 16.729740
9	11.230019
10	- 1.979996

*Angles expressed in radians

Table 2. Coefficients of Polynomial
Describing Variation of
Normal Restitution Coefficient
with Incidence Angle*

Coefficient of Number	Value of Coefficient
1	5.721469
2	- 41.880784
3	178.168464
4	-424.388179
5	572.763055
6	-406.662526
7	87.142750
8	70.651124
9	- 50.498187
10	9.676744

*Angles expressed in radians

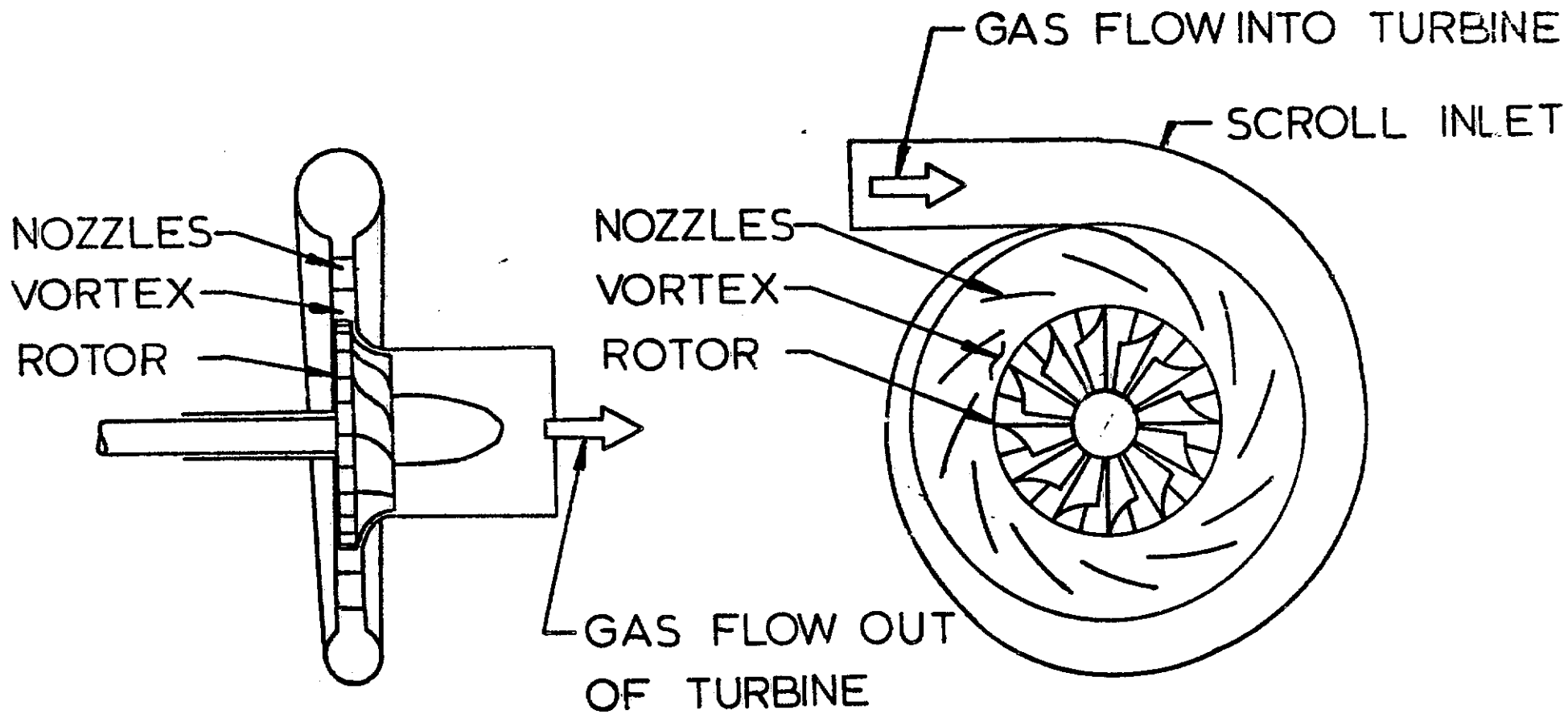


FIGURE 1. SCHEMATIC OF TYPICAL RADIAL INFLOW TURBINE

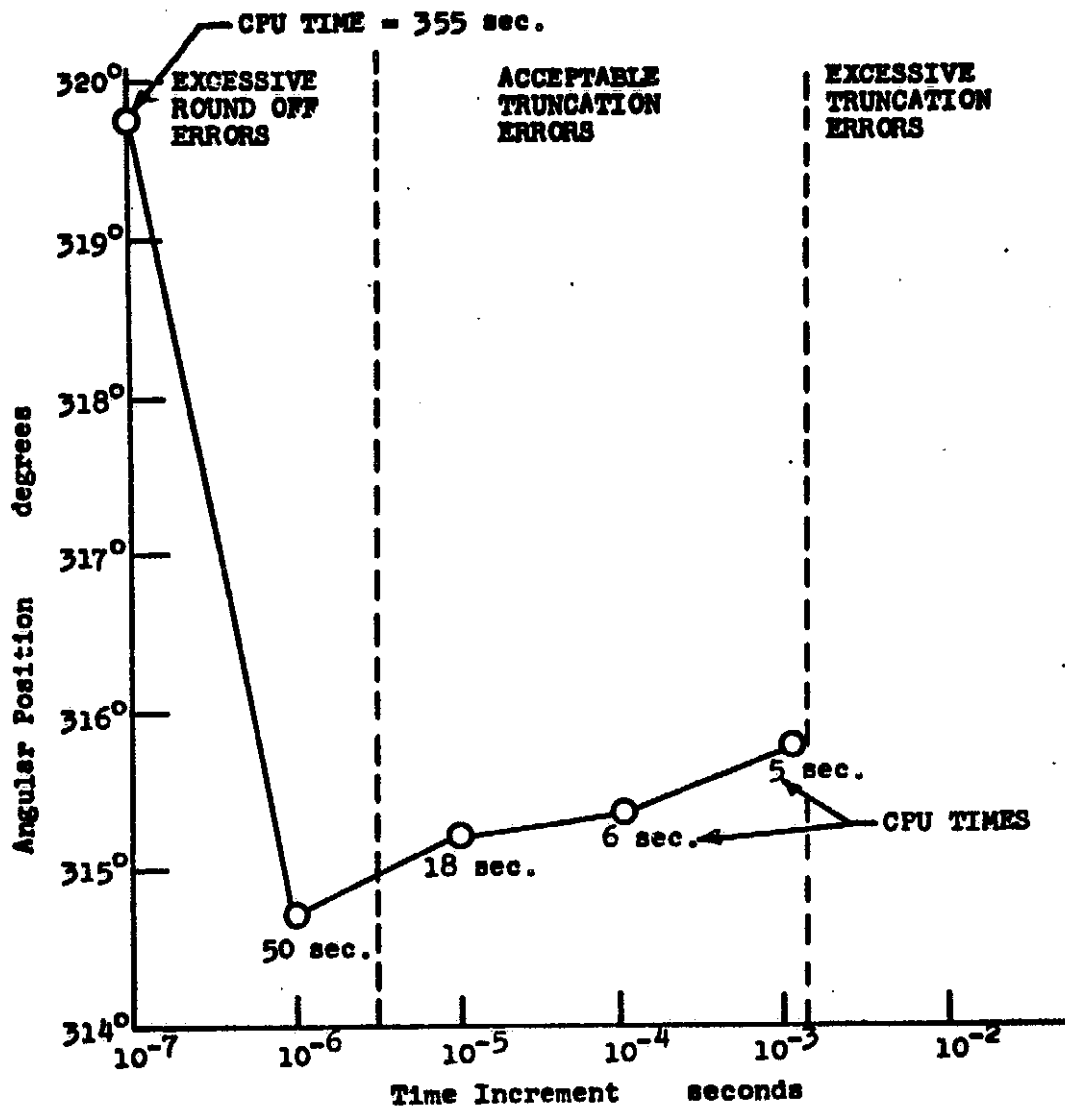


FIGURE 2. ROUND OFF AND TRUNCATION ERRORS INHERENT IN THE NUMERICAL PROCEDURE

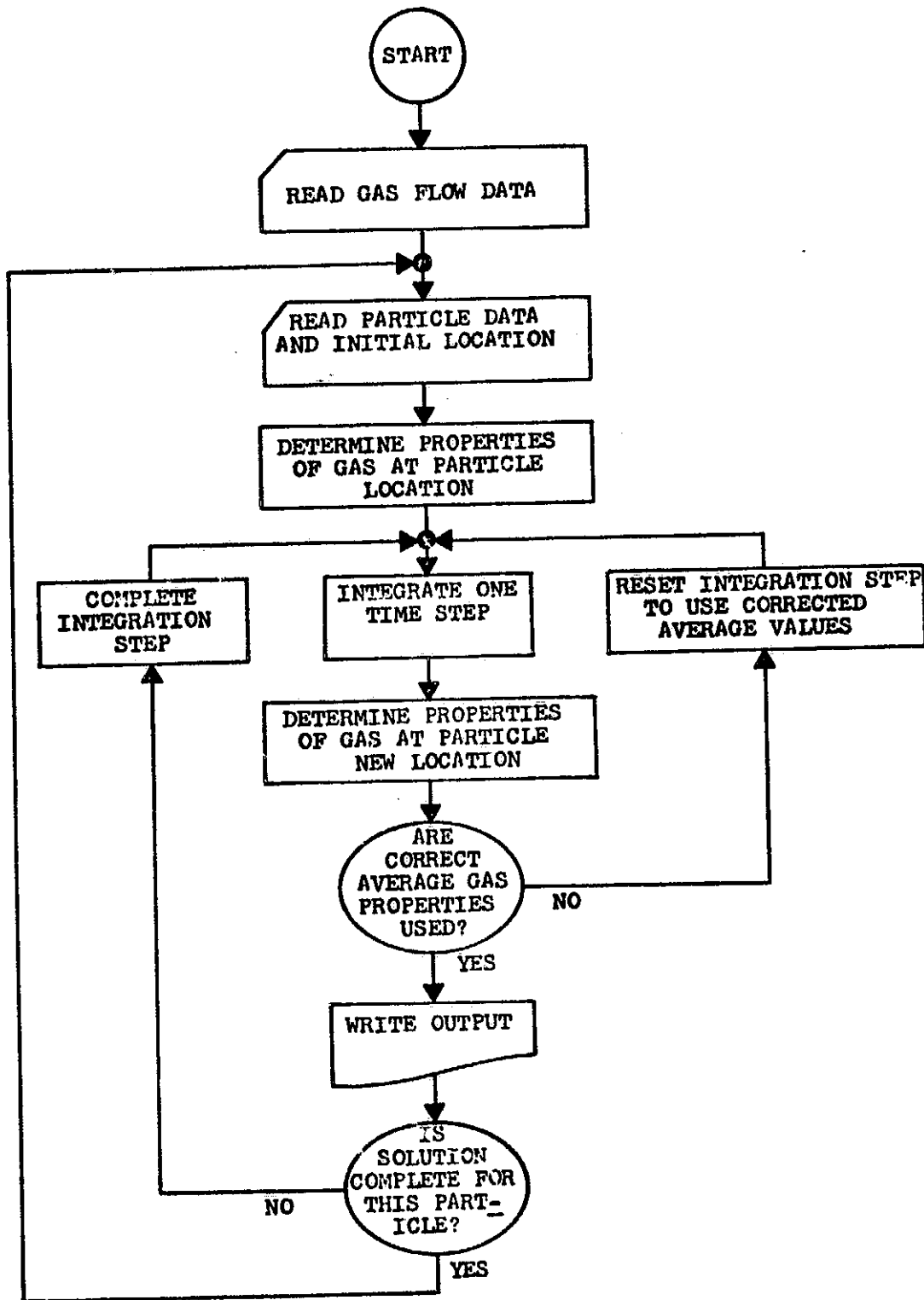


FIGURE 3. TYPICAL FLOW CHART

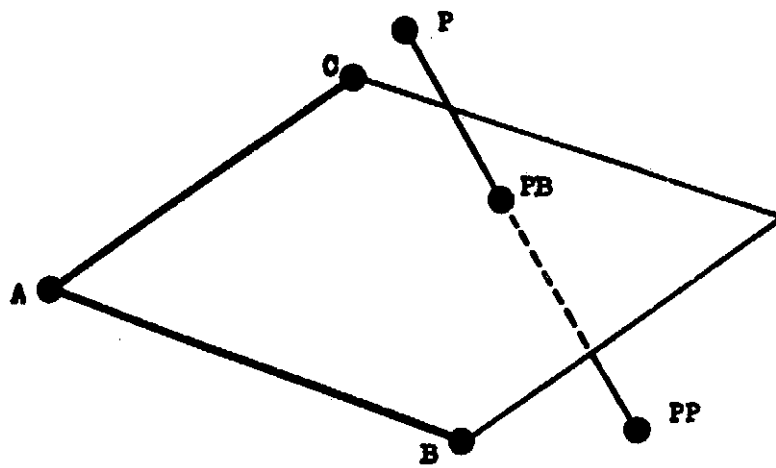


FIGURE 4. INTERSECTION OF TRAJECTORY WITH SURFACE

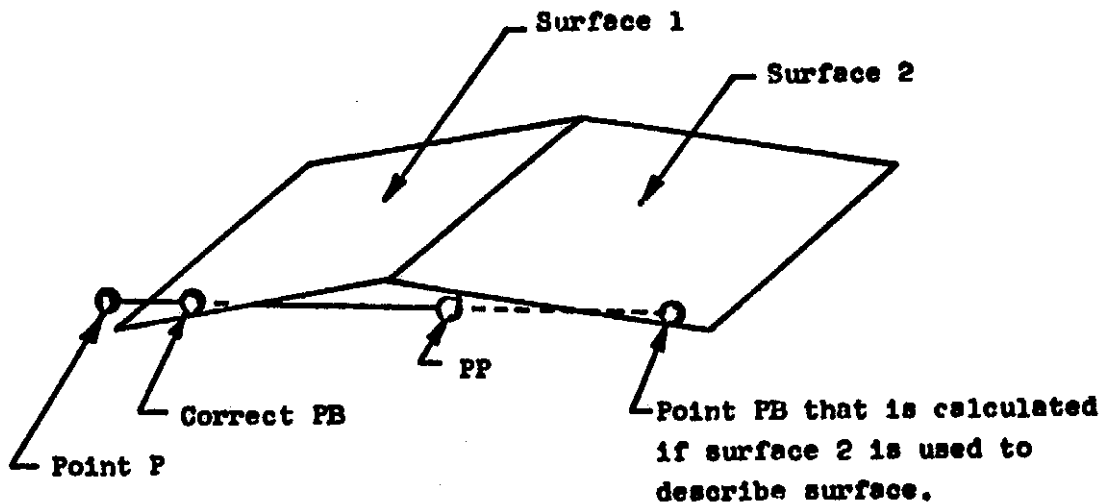


FIGURE 5. FIRST TYPE OF BOUNCE NEAR SURFACE NODES

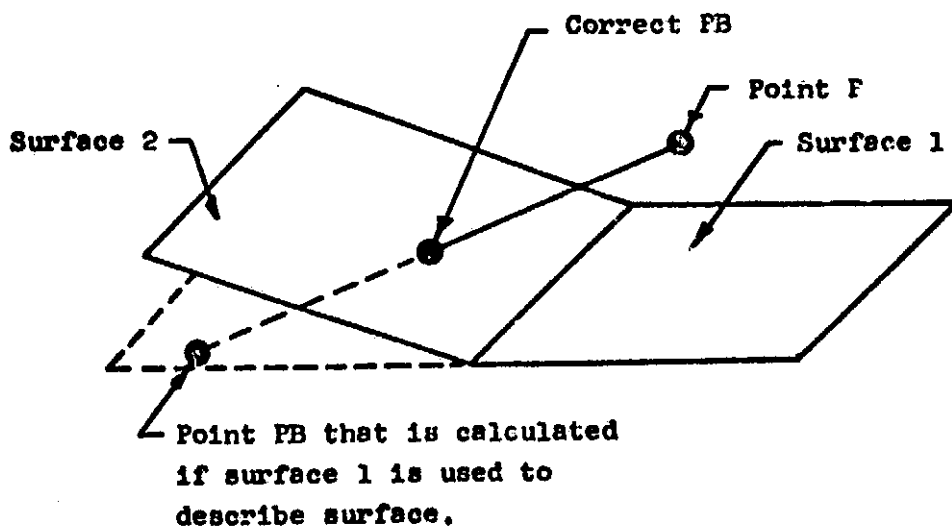


FIGURE 6. SECOND TYPE OF BOUNCE NEAR SURFACE NODES

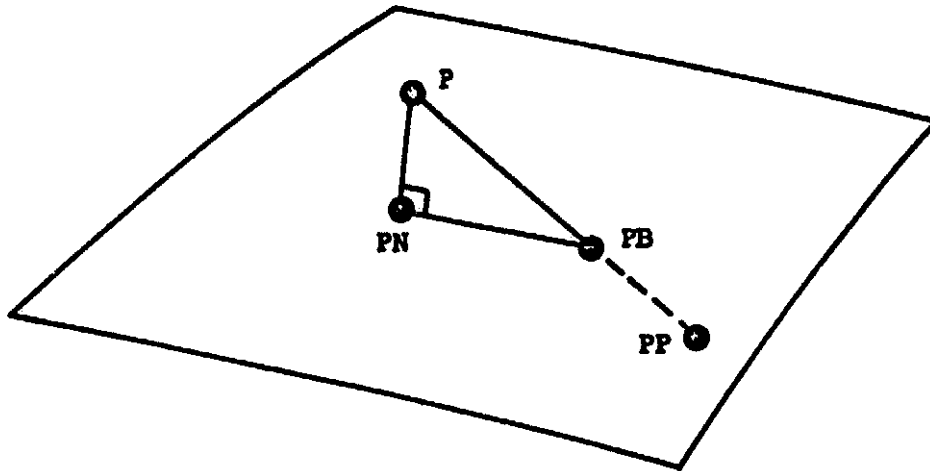


FIGURE 7. METHOD ONE POINTS

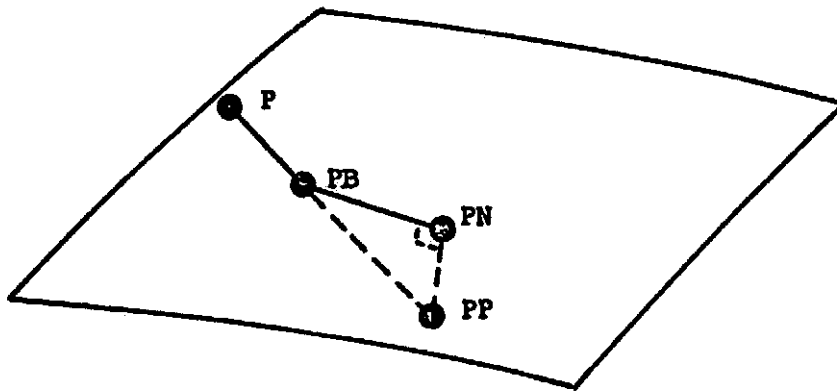


FIGURE 8. METHOD TWO POINTS

Material:
 Surface - 2024 Aluminum
 Particle - SiO₂

○ Data given in Reference 6
 for incidence velocity = 76.2 m/sec.

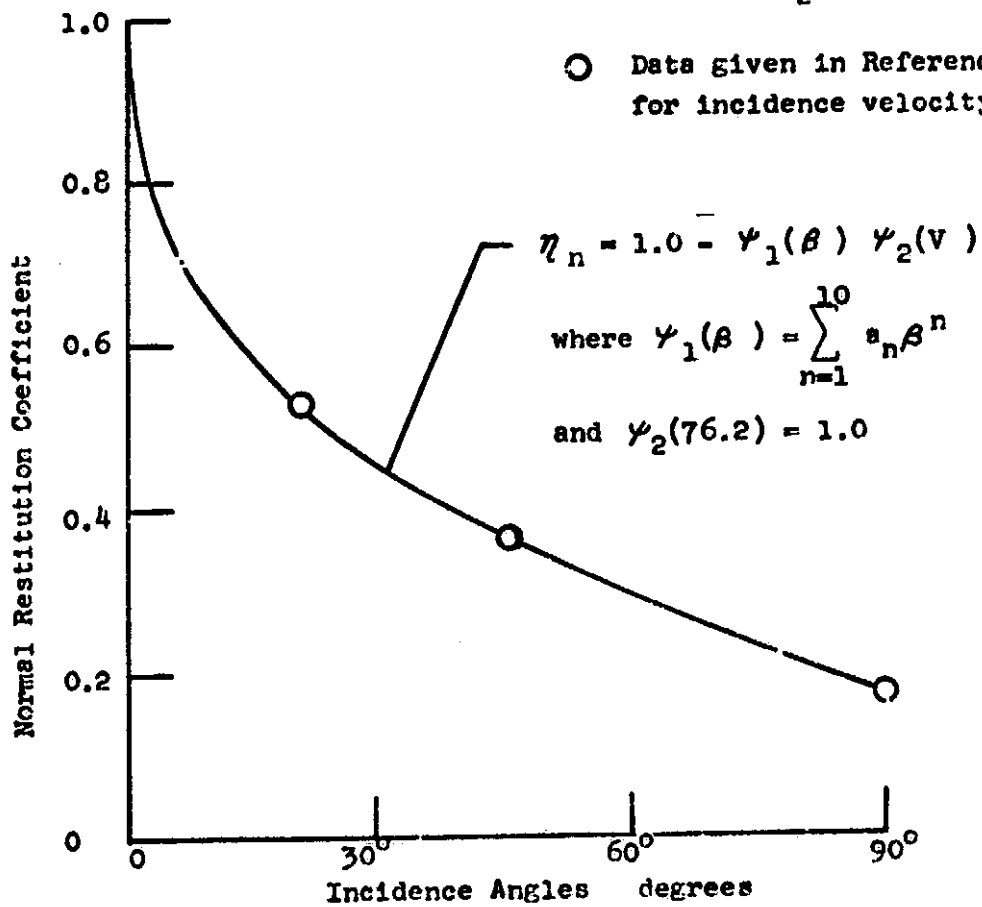


FIGURE 9. INFLUENCE OF INCIDENCE ANGLE ON NORMAL RESTITUTION COEFFICIENT

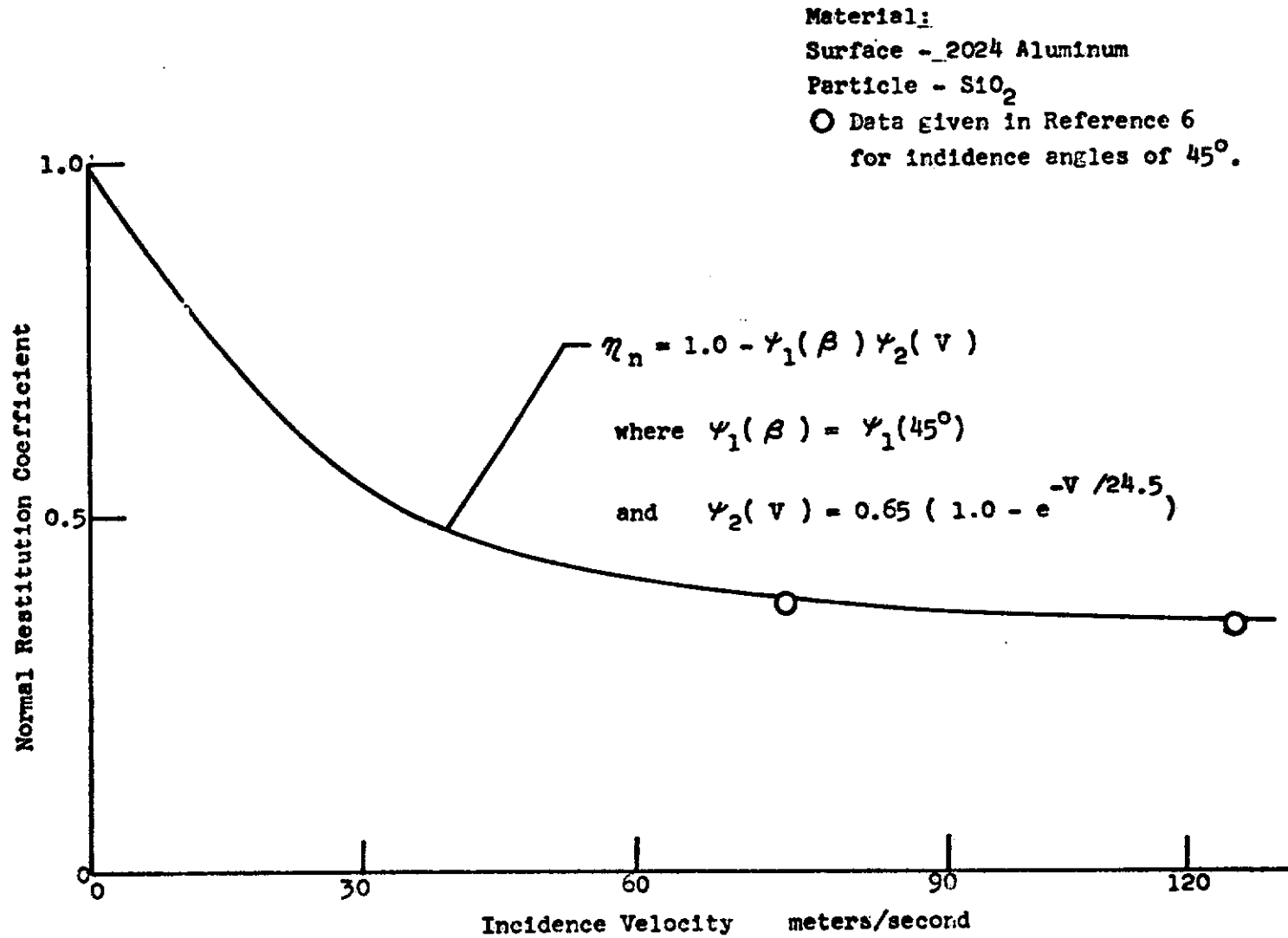


FIGURE 10. INFLUENCE OF INCIDENCE VELOCITY ON NORMAL RESTITUTION COEFFICIENT

Material:
Surface - 2024 Aluminum
Particle - SiO₂

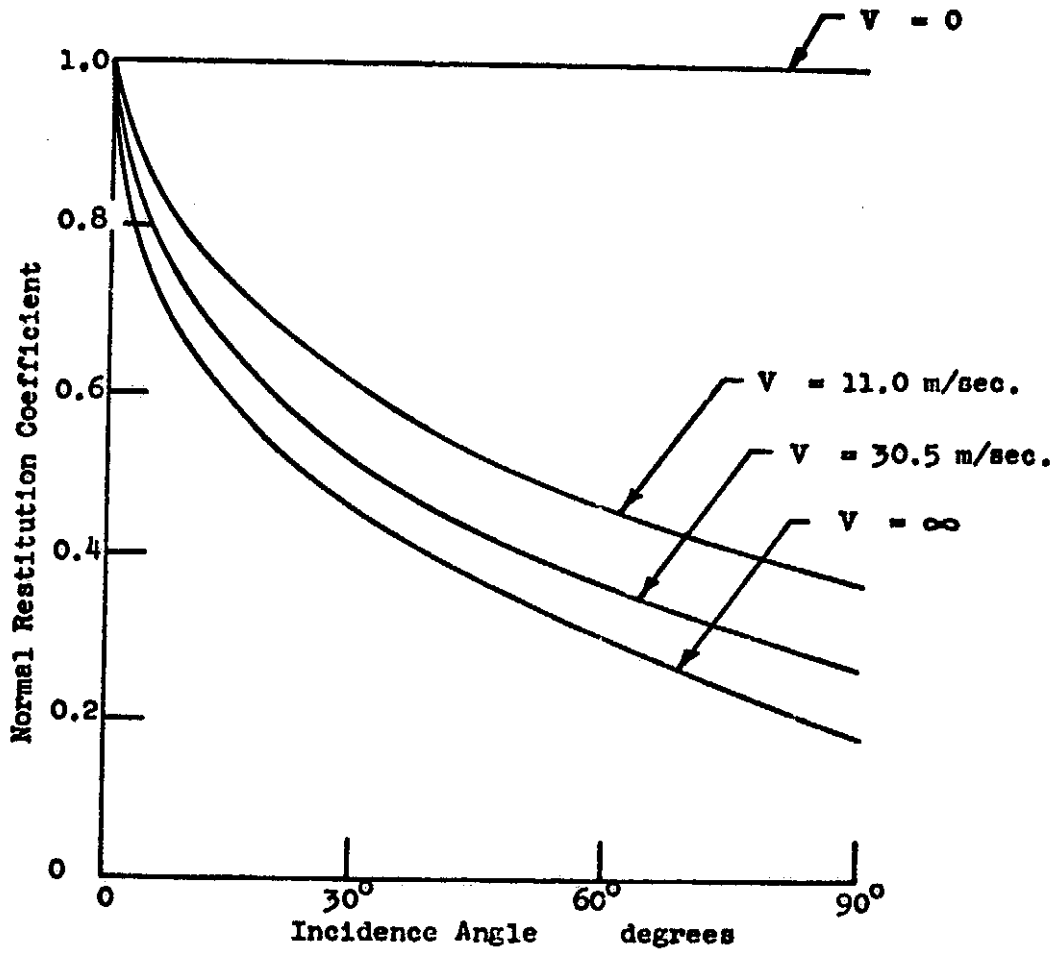


FIGURE 11. NORMAL RESTITUTION COEFFICIENT

Material:
 Surface - 2024 Aluminum
 Particle - SiO₂
 ○ Data given in Reference 6
 for incidence velocity = 76.2 m/sec.

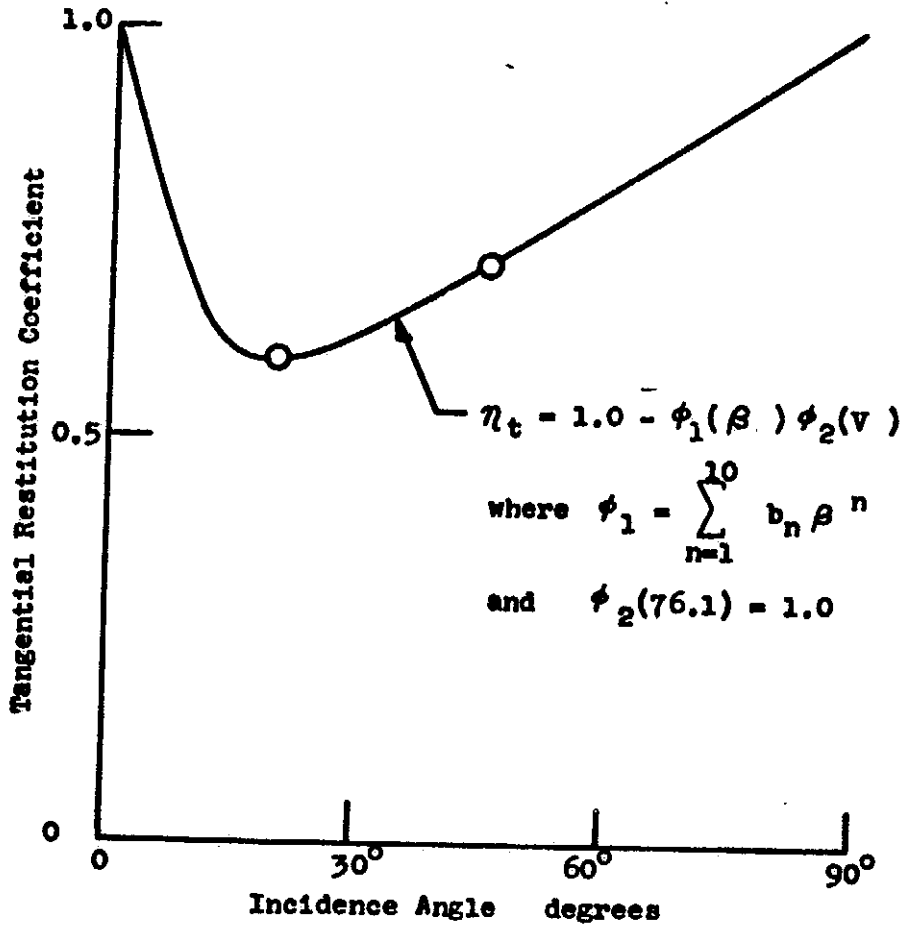


FIGURE 12. INFLUENCE OF INCIDENCE ANGLE ON TANGENTIAL RESTITUTION COEFFICIENT

Material:

Surface - 2024 Aluminum

Particle - SiO₂

○ Data given in Reference 6
for incidence angles of 45°.

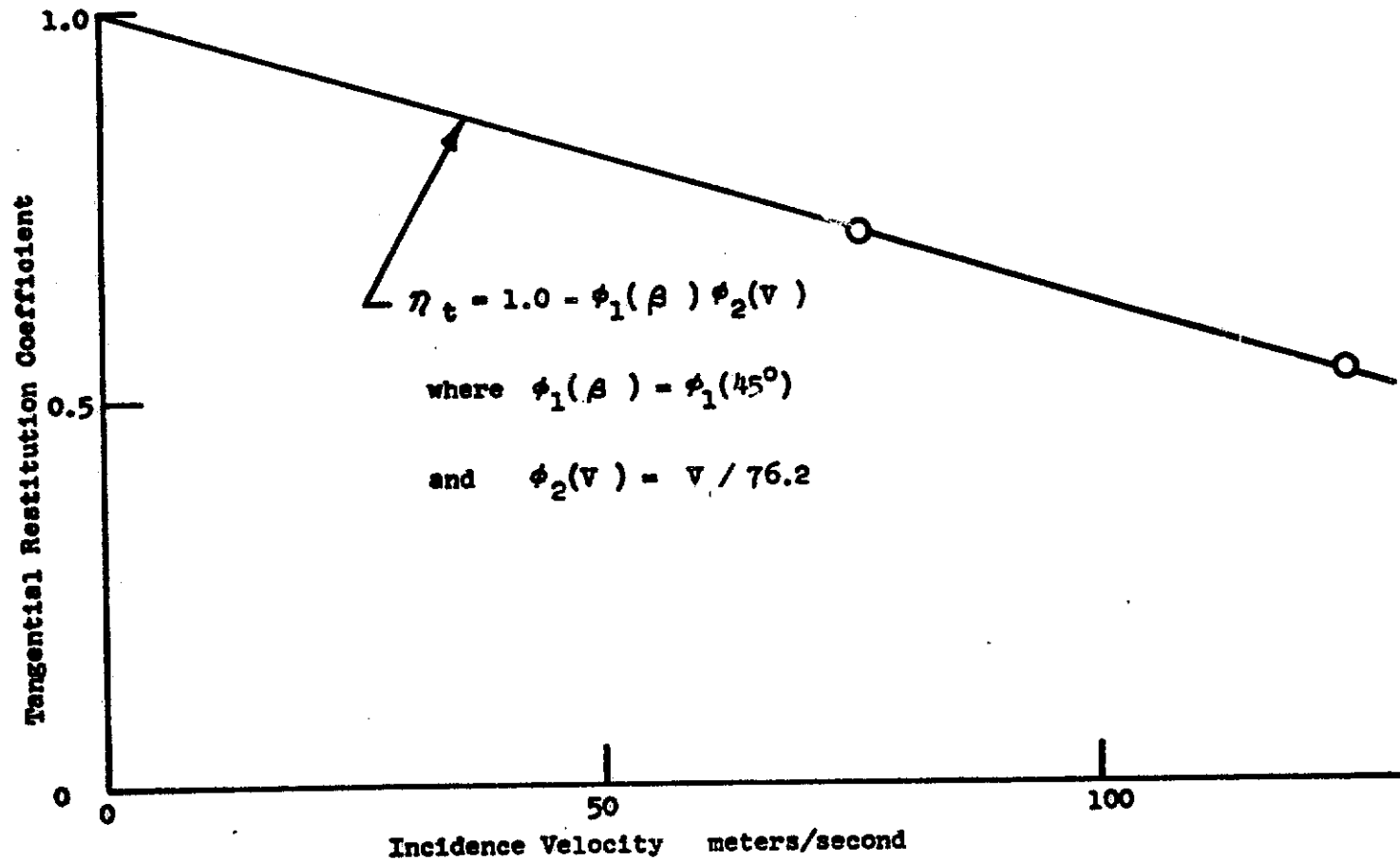


FIGURE 13. INFLUENCE OF INCIDENCE VELOCITY ON TANGENTIAL RESTITUTION COEFFICIENT

Material:
Surface - 2024 Aluminum
Particle - SiO₂

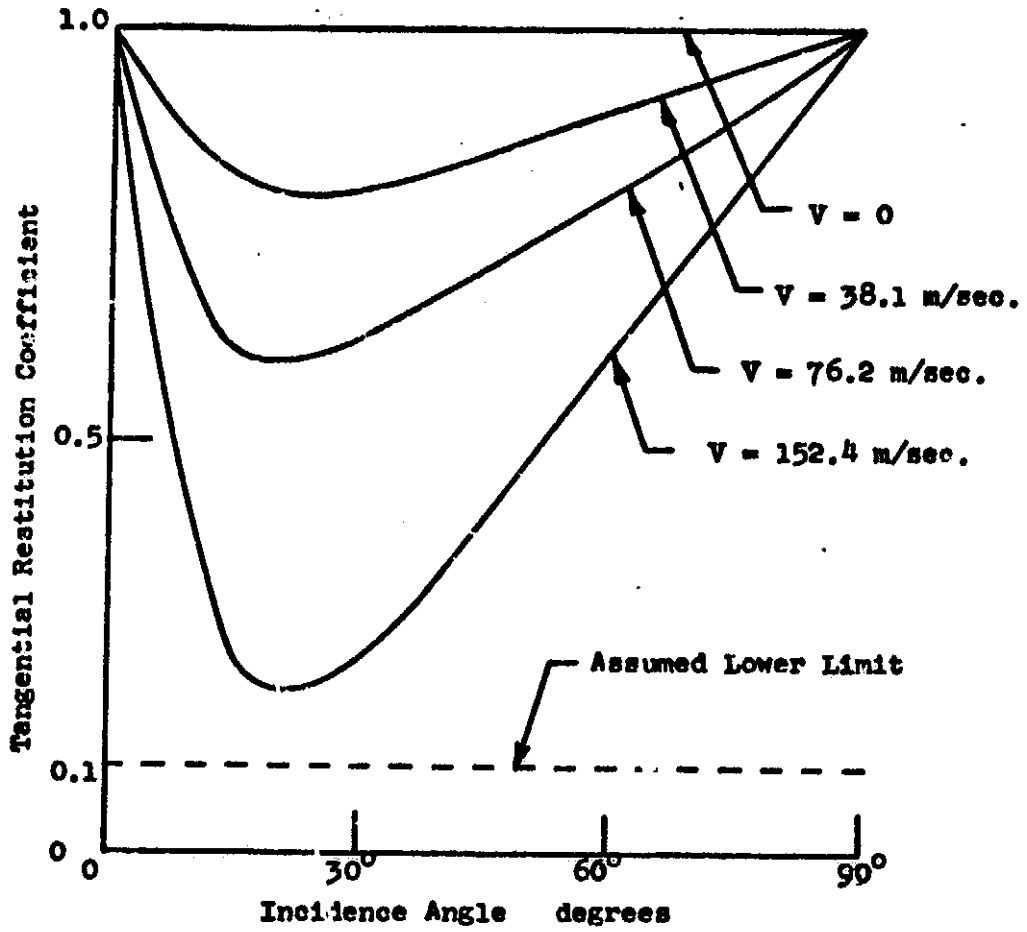


FIGURE 14. TANGENTIAL RESTITUTION COEFFICIENT

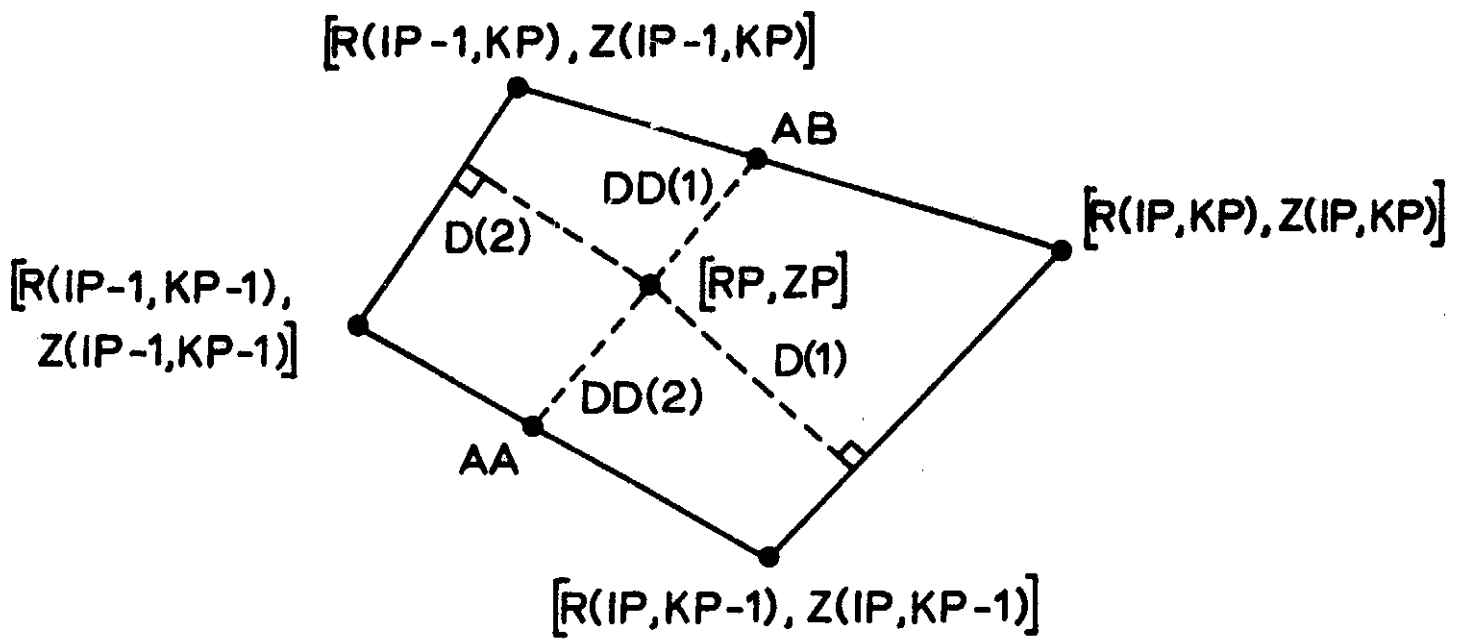


FIGURE 15. TYPICAL GRID CONFIGURATION USED IN POLATE

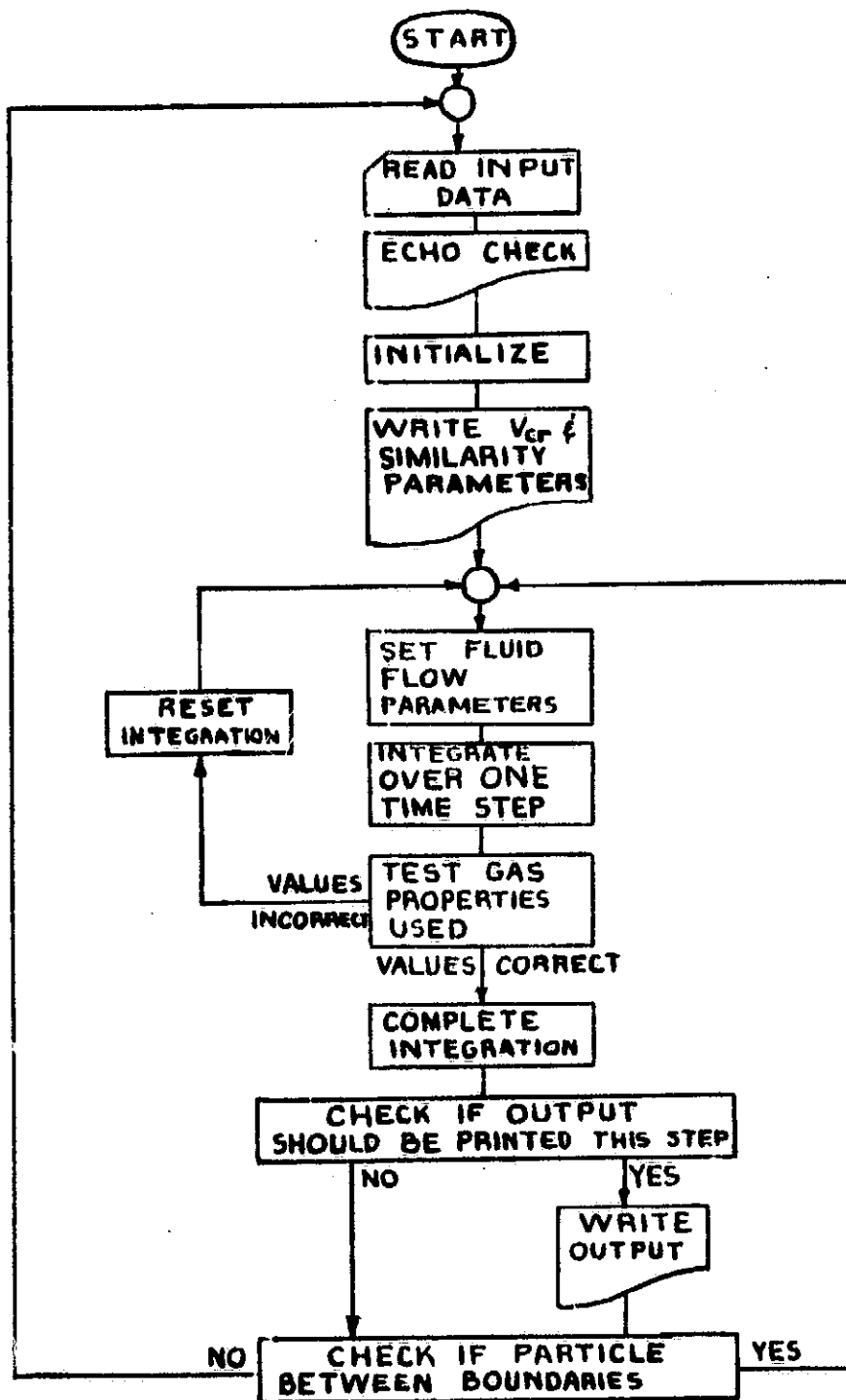


FIGURE 17. PARDIM FLOW CHART

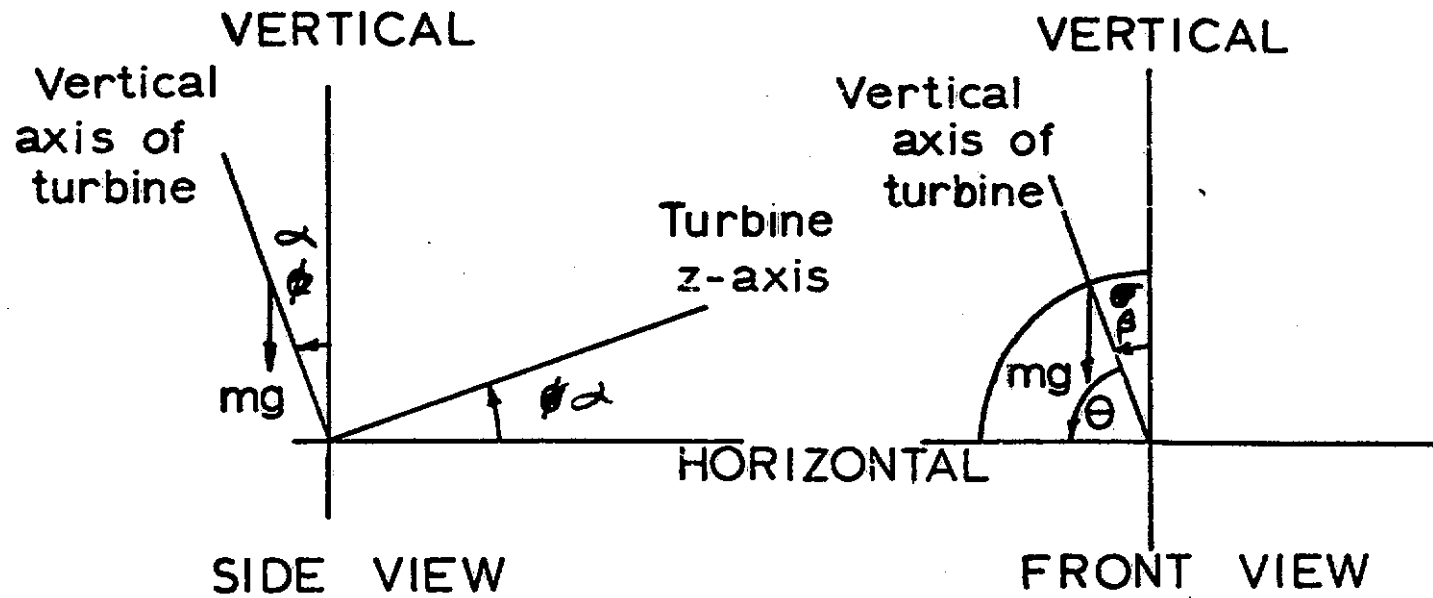


FIGURE 18. ORIENTATION COORDINATES

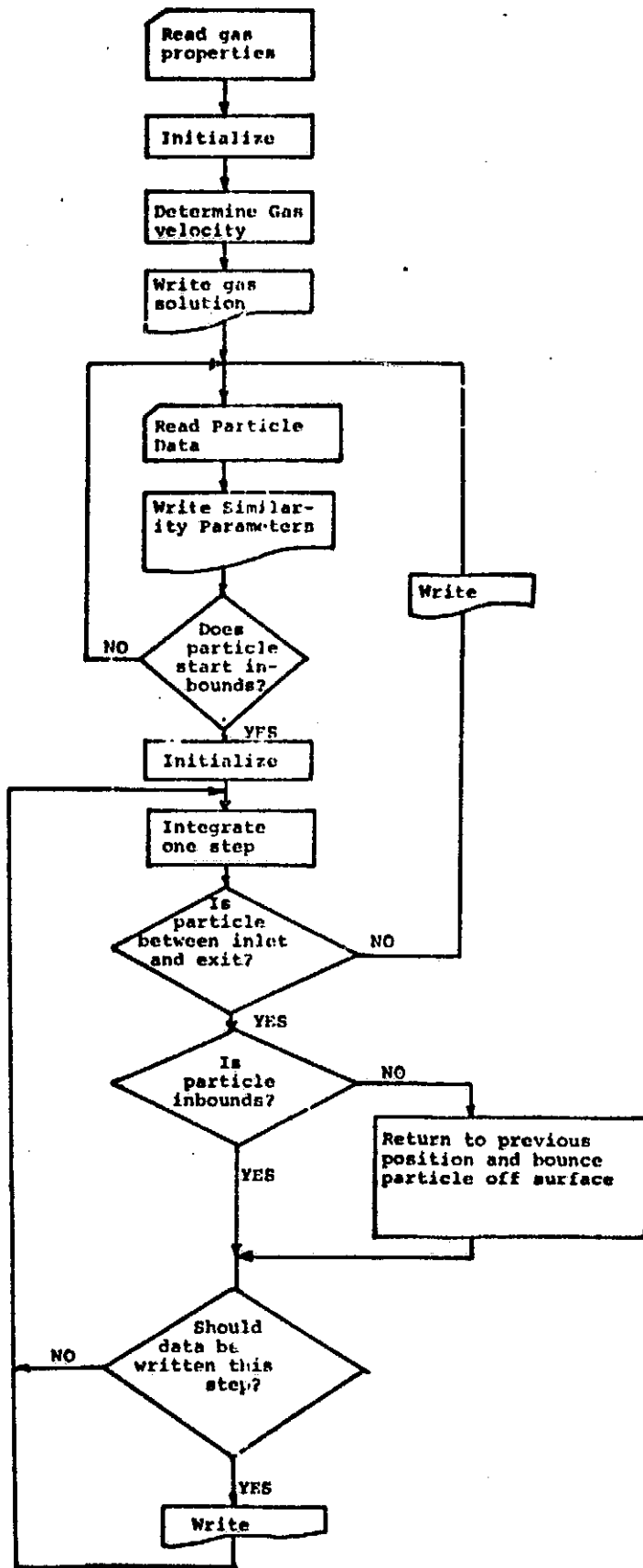


FIGURE 19. TWO-DIMENSIONAL SCROLL FLOW CHART

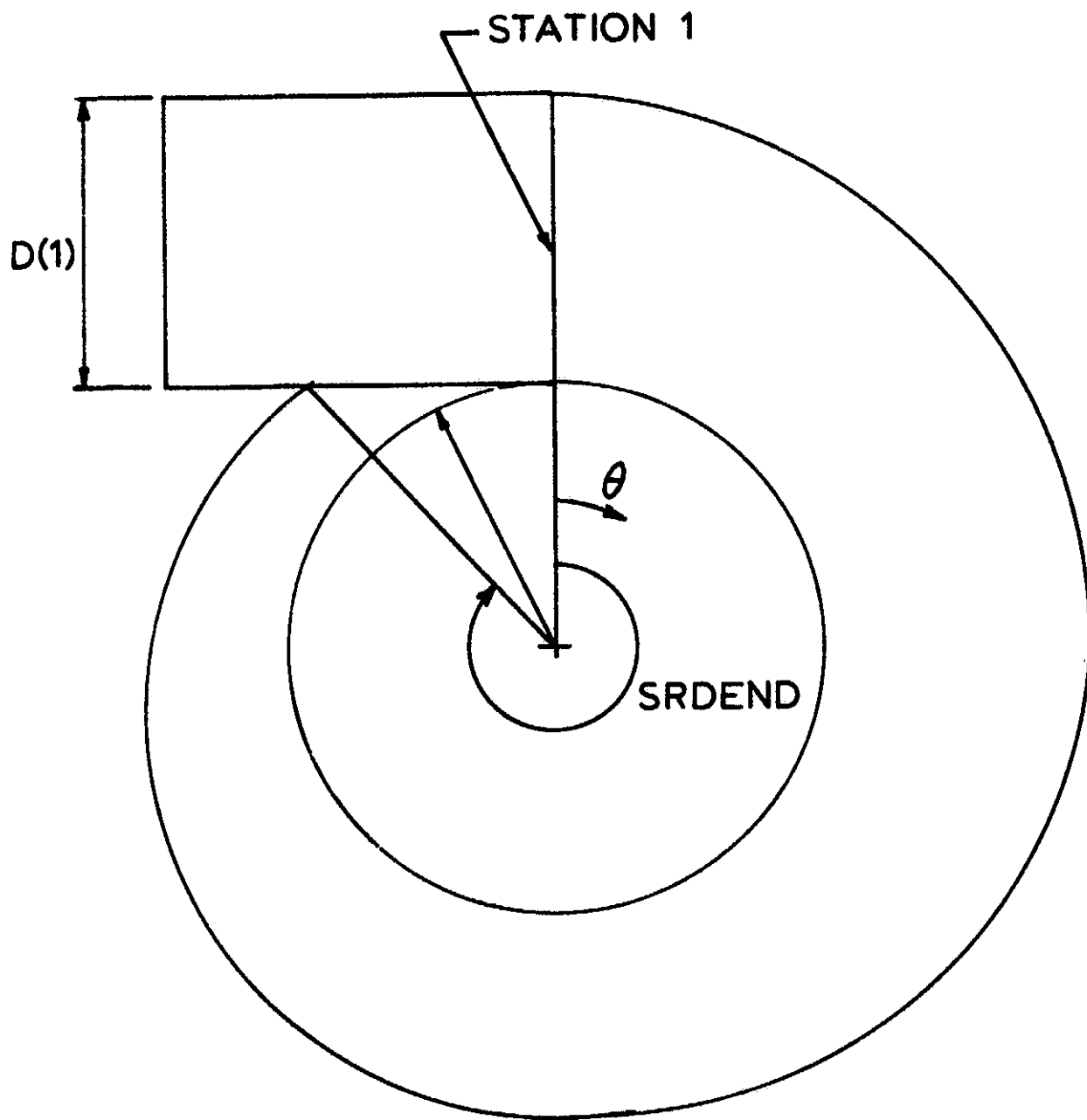


FIGURE 20. SCROLL GEOMETRY

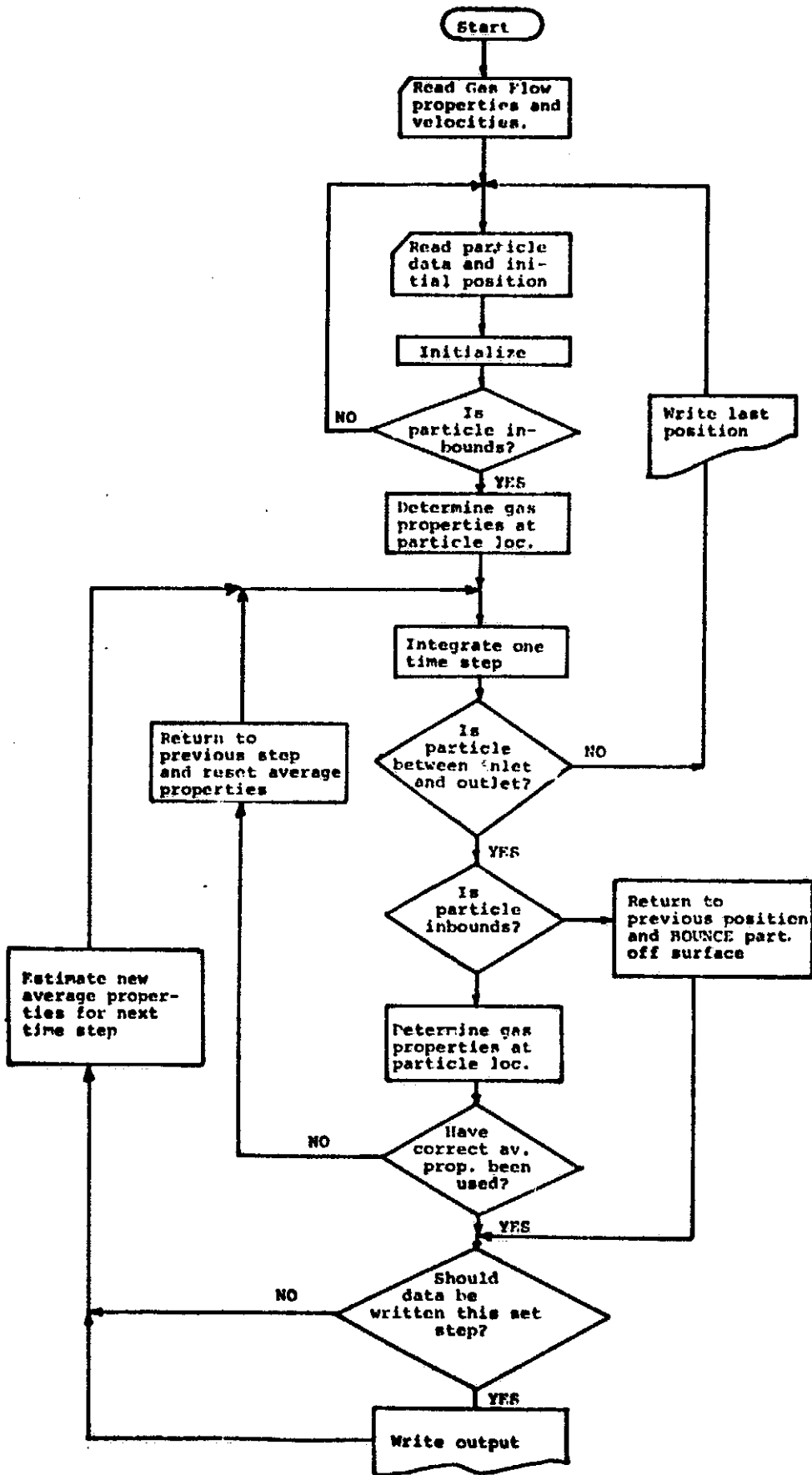
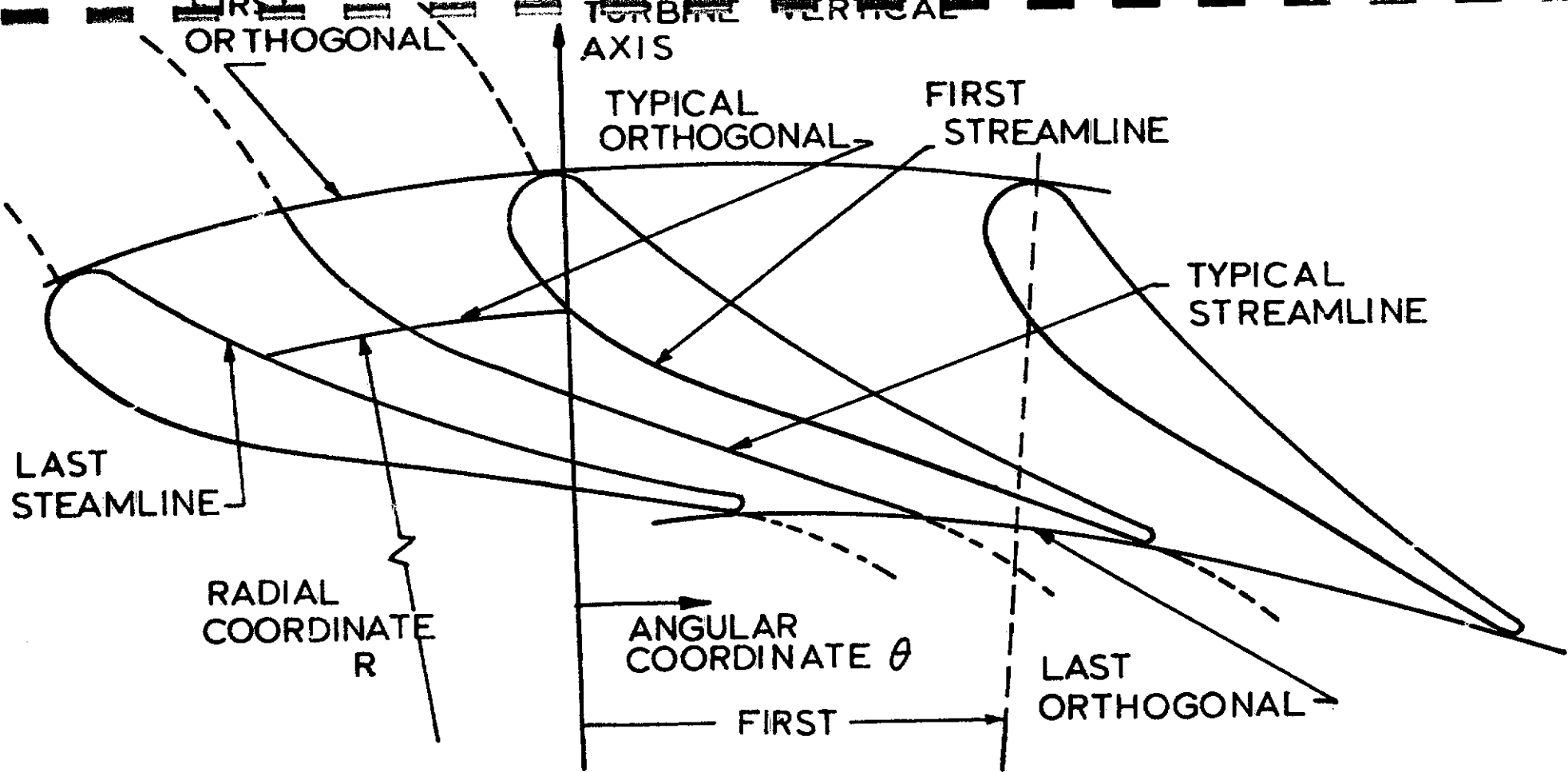


FIGURE 21. FLOW CHART FOR STATOR



NOTE: Data must be input from first to last orthogonal and from first to last streamline.

FIGURE 22. NOZZLE GEOMETRY AND COORDINATE SYSTEM USED IN STATOR PROGRAM

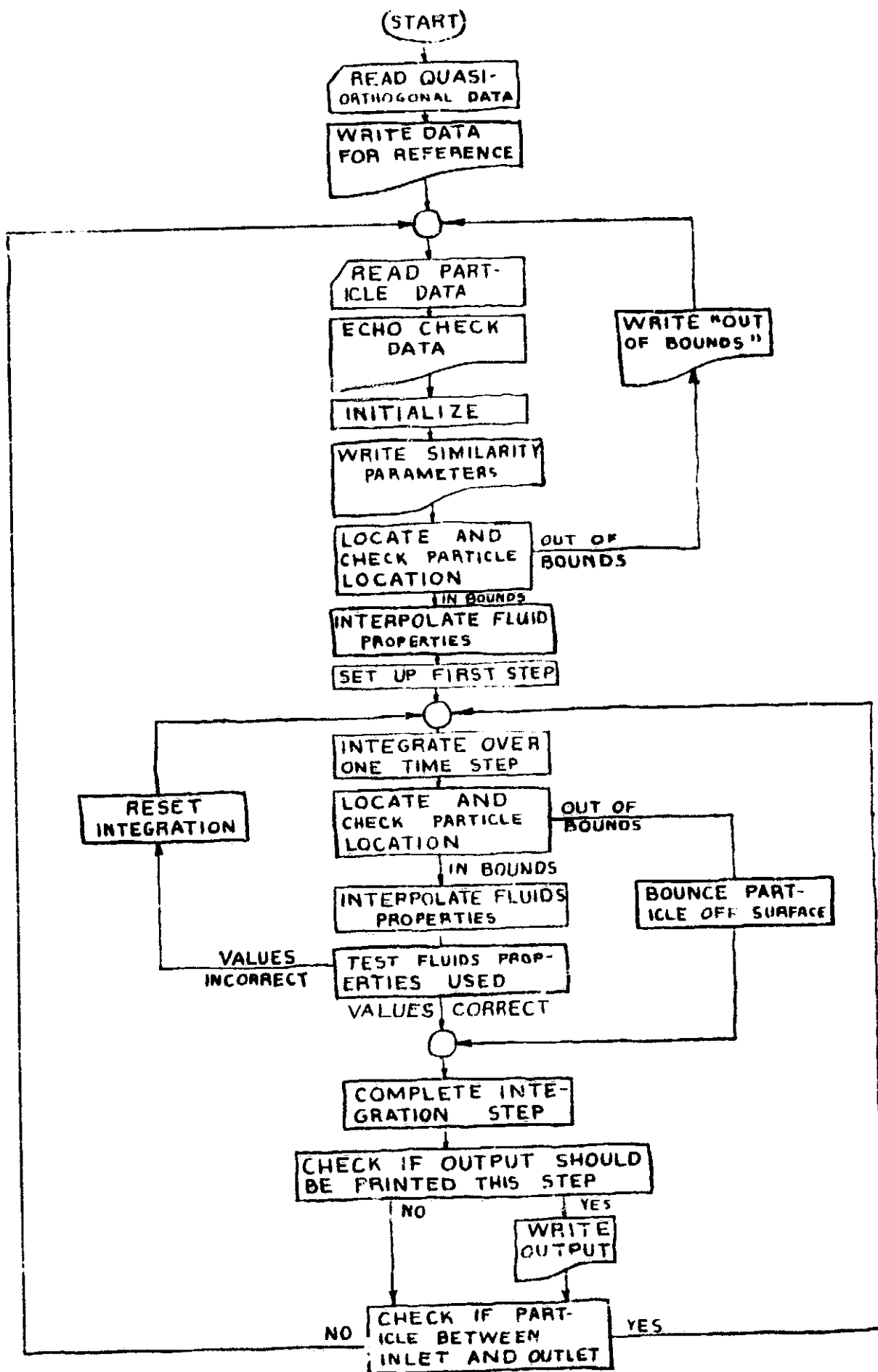


FIGURE 23. ROTOR FLOW DIAGRAM

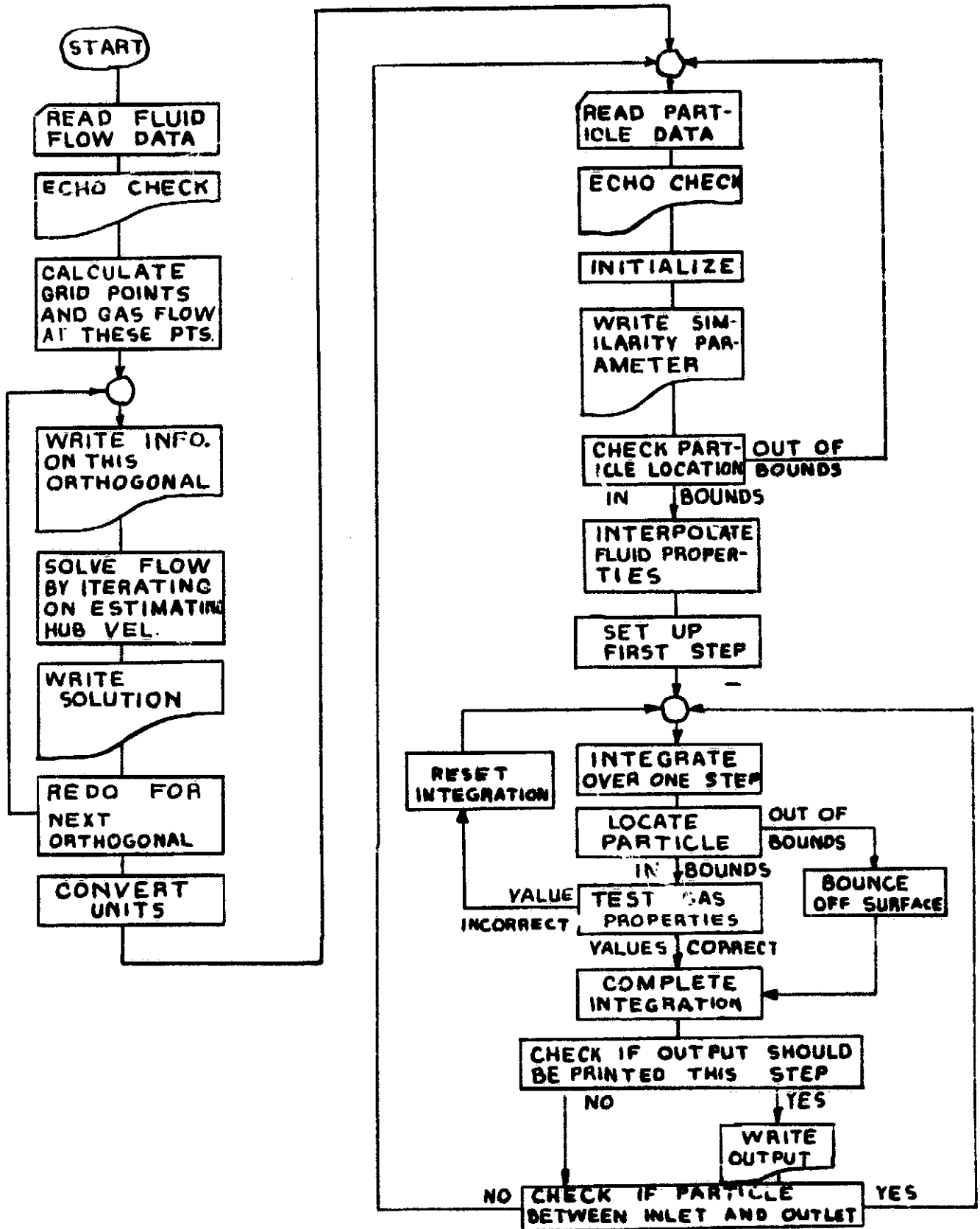


FIGURE 24. VANPY FLOW DIAGRAM

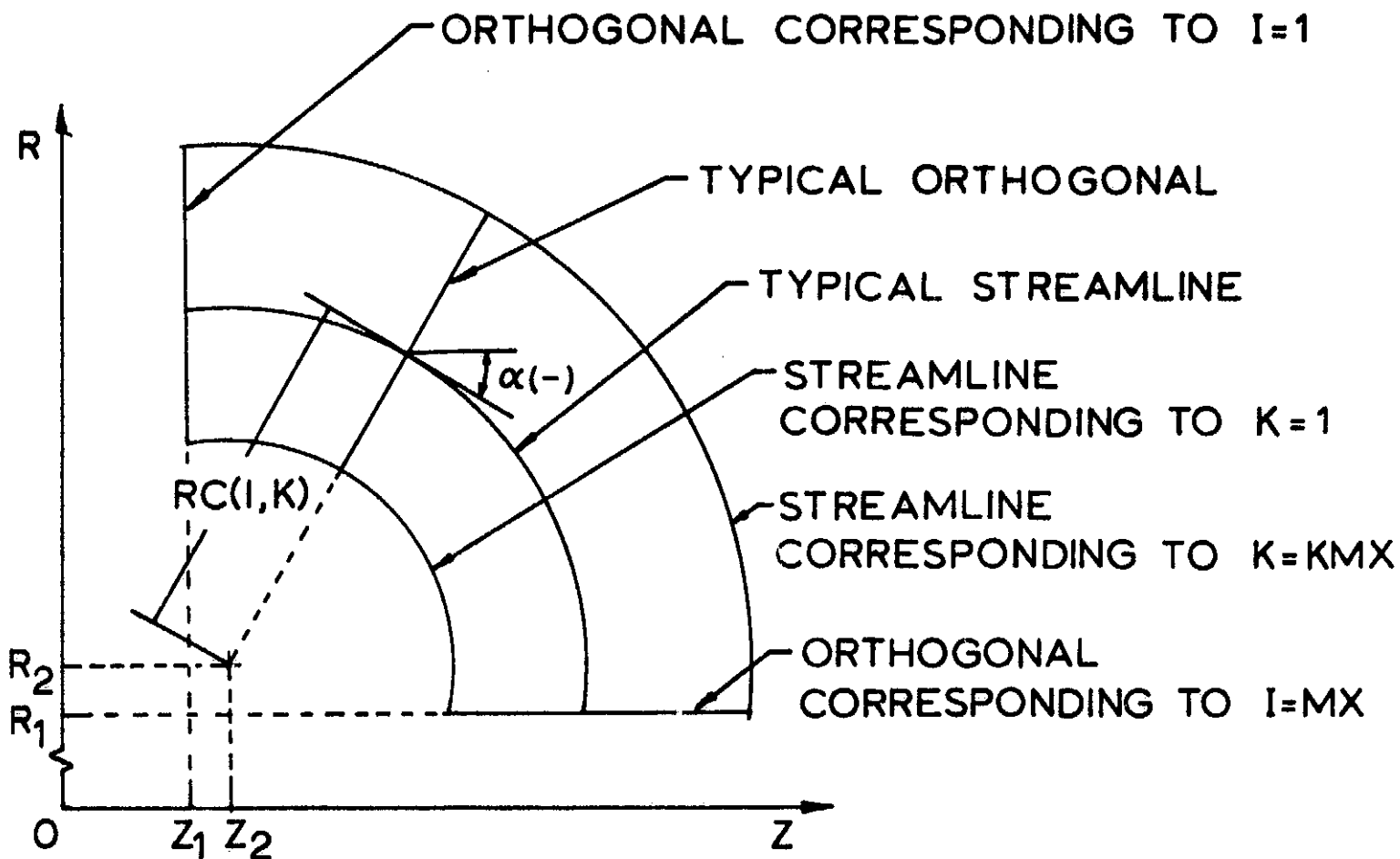


FIGURE 25. COORDINATE SYSTEM USED IN VANPY COMPUTER PROGRAM