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

By:
W.B. Clevenger, Jr.
and
W. Tabakoff

Supported by:
NATIONAL AERONAUTICS AND SPACE ADMINISTRATION
Lewis Research Center
Contract NGR 36-004-055
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.
# TABLE OF CONTENTS

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

\[
\begin{align*}
Y_1 &= \dot{r} \\
Y_2 &= \frac{\delta r}{\delta t} \\
Y_3 &= \theta \\
Y_4 &= \frac{\delta \theta}{\delta t} \\
Y_5 &= z \\
Y_6 &= \frac{\delta z}{\delta t}
\end{align*}
\]

(1)

The equations of motion can then be expressed as

\[
\begin{align*}
\frac{\delta Y_1}{\delta t} &= Y_2 \\
\frac{\delta Y_2}{\delta t} &= Y_1 Y_4^2 + 2Y_1 Y_4 \dot{\omega} + Y_1 \omega^2 + B|v|v_x - g \cos(\theta - \varphi) \\
\frac{\delta Y_3}{\delta t} &= Y_4
\end{align*}
\]
\[
\frac{\delta y_4}{\delta t} = \frac{1}{y_1} [-2y_2y_4 - 2\omega_2 + B|v|v_\theta + g \sin(\theta+\sigma)]
\]

\[
\frac{\delta y_5}{\delta t} = y_6
\]

\[
\frac{\delta y_6}{\delta t} = B|v|v_z - g \sin\phi
\]  \tag{2}

Where \( B = \frac{1}{2} \frac{C_D A}{m} = \frac{3}{4} \rho \frac{C_D}{D_p} \) for spherical particles. The terms \( V_r, V_\theta \) and \( V_z \) represent the velocity of the gas relative to the moving particle.

\[
V_r = w_r - y_2
\]

\[
V_\theta = w_\theta - y_1y_4
\]

\[
V_z = w_z - y_6
\]  \tag{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

\[
y_{i+1,j} = y_{i,j} + \frac{h}{6} (k_{1j} + 2k_{2j} + 2k_{3j} + k_{4j})
\]  \tag{4}

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

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

\[
k_{2j} = f_j(x_i + \frac{1}{2} h, y_i + \frac{1}{2} h k_{1j})
\]
The values of the function \( f_j \) corresponding to the equations of motion (2) are:

\[
\begin{align*}
    f_1 &= v_2 \\
    f_2 &= y_1 y_4^2 + 2y_1 y_4 \omega + y_1 \omega^2 + B|V|V_x - g \cos(\theta - \sigma) \\
    f_3 &= y_4 \\
    f_4 &= \frac{1}{y_1} [-2y_2 y_4 - 2\omega y_2 + B|V|V_0 + g \sin(\theta + \sigma)] \\
    f_5 &= y_6 \\
    f_6 &= B|V|V_z - g \sin \phi
\end{align*}
\]

In Equation (4), \( h \) represents the time increment between integration steps. In most of the work that is presented here, a value of \( 1 \times 10^{-5} \) 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

\[ \delta = \frac{10}{3} \frac{\rho P D_p}{\rho g} \]  

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

\[ \tau = \frac{P^2}{18 \mu g} \]  

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

\[ \text{Re}_{cr} = \frac{\rho g^* (V_{cr}/2.) D_p}{\mu g^*} \]  

Although this term was retained in the programs, experience has shown that is 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 $P$ is the last point along the trajectory that is still in bounds and point $PP$ 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:

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

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 $\vec{A}$, $\vec{B}$, and $\vec{C}$ within the subroutine. The subroutine then determines vectors $\vec{AB}$ and $\vec{AC}$ which lie in the surface, where
\[ \overline{AB} = \overline{B} - \overline{A} \]  
\[ \overline{AC} = \overline{C} - \overline{A} \]  

Next the subroutine determines the unit vector normal to the surface by calculating the cross product of \( \overline{AB} \) and \( \overline{AC} \), and then dividing by the magnitude of the resulting vector. Indicated below is the equation form of this operation.

\[
\overline{AB} \times \overline{AC} = \begin{vmatrix}
\mathbf{i} & \mathbf{j} & \mathbf{k} \\
AB_1 & AB_2 & AB_3 \\
AC_1 & AC_2 & AC_3
\end{vmatrix} 
\]

\[
= (AB_2 AC_3 - AB_3 AC_2) \mathbf{i} + (AB_3 AC_1 - AB_1 AC_3) \mathbf{j} + (AB_1 AC_2 - AB_2 AC_1) \mathbf{k} 
\]  

\[ \overline{UN} = \frac{\overline{AB} \times \overline{AC}}{|\overline{AB} \times \overline{AC}|} = UN_1 \mathbf{i} + UN_2 \mathbf{j} + UN_3 \mathbf{k} \]  

If \( |\overline{AB} \times \overline{AC}| = 0 \), the subroutine returns \( NFIX = 0 \). This corresponds to a case where points \( \overline{A}, \overline{B}, \) and \( \overline{C} \) 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, \( \overline{PB} \), where the particle bounces off the surface. This is determined by solving simultaneously the equations of the plane given by the unit vector \( \overline{UN} \) and the point \( \overline{A} \) and the straight line connecting the points \( \overline{P} \) and \( \overline{PP} \) of the trajectory. The equation of the plane is

\[ UN_1(x-A_1) + UN_2(y-A_2) + UN_3(z-A_3) = 0 \]  

\[ \text{(16)} \]
The equations for the line trajectory are

\[ \frac{x - P_1}{V_1} = \frac{y - P_2}{V_2} = \frac{z - P_3}{V_3} \]  

(17)

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

\[
\begin{align*}
UN_1 x + UN_2 y + UN_3 z &= D_1 \\
V_2 x - V_1 y &= D_2 \\
V_3 x - V_1 z &= D_3
\end{align*}
\]

(18)

where

\[
D_1 = UN_1 A_1 + UN_2 A_2 + UN_3 A_3
\]

\[
D_2 = V_2 P_1 - V_1 P_2
\]

\[
D_3 = V_3 P_1 - V_1 P_3
\]

(19)

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

\[
\begin{vmatrix}
UN_1 & UN_2 & UN_3 \\
V_2 & -V_1 & 0 \\
V_3 & 0 & -V_1
\end{vmatrix} = UN_1 V_1^2 + UN_2 V_1 V_2 + UN_3 V_1 V_3
\]

(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

$$UN_2 y + UN_3 z = D_4$$
$$V_3 y - V_2 z = D_5$$

where

$$D_4 = D_1 - UN_1 \cdot P_1$$
$$D_5 = V_3 P_2 - V_2 P_3$$

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

$$(UN_2 V_2 + UN_3 V_3) = 0$$

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 $PB$ and $P$, $PP$ and $P$, and $PB$ and $PP$. These distances can be expressed in equation form as

$$DPB = |PB - P|$$
$$DPP = |PP - P|$$
$$DPPB = |PB - PP|$$
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 \( DPP = |\overline{PP} - \overline{P}| \) is calculated and the subroutine checks to insure if it is greater than the distance \( DPPB = |\overline{PP} - \overline{PB}| \). 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 it bounces from the surface. Method 1 is used when \( DPB > \frac{1}{2} DPP \). 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 \( \overline{PN} \) which is the normal projection of the point \( \overline{P} \) 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 \( \overline{P} \) 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:

\[
\begin{align*}
UN_1 x + UN_2 y + UN_3 z &= D_1 \\
UN_2 x - UN_1 y &= D_2 \\
UN_3 x - UN_1 a &= D_3
\end{align*} \quad (26)
\]
where

\[ D_1 = UN_1 A_1 + UN_2 A_2 + UN_3 A_3 \]
\[ D_2 = UN_2 P_1 - UN_1 P_2 \]
\[ D_3 = UN_3 P_1 - UN_1 P_3 \]  \hspace{1cm} (27)

The solution of this system of equations gives the desired position vector \( \vec{F}N \). 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

\[
\begin{vmatrix}
UN_1 & UN_2 & UN_3 \\
UN_2 & -UN_1 & 0 \\
UN_3 & 0 & -UN_1
\end{vmatrix} = UN_1 (UN^2 + UN_2^2 + UN_3^2) \]  \hspace{1cm} (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

\[ UN_2 y + UN_3 z = D_4 \]
\[ UN_3 y - UN_2 z = D_5 \]  \hspace{1cm} (29)

where

\[ D_4 = D_1 - UN_1 P_1 = D_1 \]
\[ D_5 = UN_3 P_2 - UN_2 P_3 \]  \hspace{1cm} (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 \( \vec{P} \) to \( \vec{FB} \) is given by

\[ DTIME = DPE/VPP \]  \hspace{1cm} (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 \( \bar{P}N \) and \( \bar{P}B \):

\[
\bar{P}B = \bar{P}N + \bar{v}(At)
\]  

(32)

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

\[
v_{t1} = \frac{(PB_1 - PN_1)}{DTIME}
\]

\[
v_{t2} = \frac{(PB_2 - PN_2)}{DTIME}
\]

\[
v_{t3} = \frac{(PB_3 - PN_3)}{DTIME}
\]

(33)

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

\[
v_{n1} = \frac{(PN_1 - P_1)}{DTIME}
\]

\[
v_{n2} = \frac{(PN_2 - P_2)}{DTIME}
\]

\[
v_{n3} = \frac{(PN_3 - P_3)}{DTIME}
\]

(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 \( \bar{P}B \).

\[
\bar{v}_{P} = ETA(1) \bar{v}_{t} + ETA(2) \bar{v}_{n}
\]

(35)

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

\[
T = T + DTIME
\]

(36)
Method 2 is used when $\text{DPB} < \frac{1}{2} \text{DPP}$. In this case, the particle travels over a very short distance along its trajectory between $\vec{P}$ and $\vec{PP}$ 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, $\vec{PN}$ is the normal projection of the point $PP$ onto the surface. This point $\vec{PN}$ 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 $\vec{P}$ occurred in the previous equations, the coordinates of $\vec{PP}$ now occur. Next the portion of the time increment that would be used to travel from $\vec{PB}$ to $\vec{PP}$ is determined using

$$D\text{TIME} = H - \text{DPB}/VPP \quad (37)$$

The points $\vec{PP}$, $\vec{PB}$, and $\vec{PN}$ are used to resolve the velocity into the tangential and normal components, which are given by

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

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

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

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

The following pages contain a listing of the subroutine BOUNCE.
SUBROUTINE BOUNCE(A, B, C, P, PP, H, T, ETA, NFIX, PR, VP)
1 PR(3)
DIMENSION GS(3, 3), VP(3), UN(3), PH(3)
DIMENSION ETA(2)
DIMENSION NMP(3), VN(4), VT(4)
C
NFI X= 1
10 IF I = 1, 3
V(1) = (PP(1) - P(1)) / H
A(1) = A(1) - A(1)
10 AC(I) = C(I) - A(I)
VPP = SQRT(V(1)**2 + V(2)**2 + V(3)**2)
C DETERMINING 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) > 1.0E-12) GO TO 20
NFi X = 0
WRITE(6, 1000)
1000 FORMAT(47H BOUNCE HAS ZERO UNIT VECTOR DESCIBING SURFACE)
RETURN
20 N = 30 I = 1, 3
30 UN(I) = UN(I) / CMAG
C
C DETERMINING THE INTERSECTION POINT, PR, OF THE PLANE AND TRAJECTORY.
C
DETA = UN(1) * V(1)**2 + UN(2) * V(2) * V(2) + UN(3) * V(3) * V(3)
D(I) = 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
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) = C(I, J)
60 DO 60 I = 1, 3
60 GS(I, K) = N(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 P(k)=PR(K)/NFTA
GO TO 100

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

80 P(1)=P(1)
   P(1)=P(1)-UN(1)*P(1)
   P(5)=V(3)*P(2)-V(2)*P(3)
   NFTA=-UN(2)*V(2)-UN(3)*V(3)
   IF (ABS(NFTA).LT.1.0E-12) GO TO 85
   P(2)=(-N(4)*V(2)-N(5)*UN(3))/NFTA
   P(3)=(UN(2)*N(5)-V(3)*N(4))/NFTA
   GO TO 100
85 IF (ABS(V(3)).GT.1.0E-12) GO TO 90
   P(2)=P(2)
   P(3)=A(3)
   GO TO 100
90 P(2)=A(2)
   P(3)=P(3)
110 CONTINUE
   NPP=SQRT((P(1)-P(1))**2+(P(2)-P(2))**2+(P(3)-P(3))**2)
   NPP=SQRT((P(1)-P(1))**2+(P(2)-P(2))**2+(P(3)-P(3))**2)
   NPPB=SQRT((P(1)-P(1))**2+(P(2)-P(2))**2+(P(3)-P(3))**2)
   IF ((NPPB.LT.0.5*NPP).AND.((DDP,L.T.DPP))) GO TO 103
   X=2
   Y=UN
103 CONTINUE
   IF (NPP.LT.0.5*NPP)) GO TO 180
   DETERMINING THE INTERSECTION POINT, PN, OF THE SURFACE NORMAL THRU P

C IF DETERMINANT EQUALS ZERO, GO TO 140

C NFTA=UN(1)**2+UN(1)*UN(2)**2+UN(1)*UN(3)**2
   IF (ABS(NFTA).LT.1.0E-12) GO TO 140
   N(2)=P(1)*UN(2)-P(2)*UN(1)
   N(3)=P(1)*UN(3)-P(3)*UN(1)
   N(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)
   III=130 K=1,3
   III=110 I=1,3
   III=110 J=1,3
110 GS(I,J)=C(I,J)
   III=120 I=1,3
120 GS(I,K)=P(I)
   P(N(K))=GS(I,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(2,3)*GS(1,2)*GS(2,3)*GS(1,3)
   2 -GS(3,3)*GS(2,1)+GS(1,2)
130 PN(K)=PN(K)/NFTA

ORIGINAL PAGE IS OF POOR QUALITY
GO TO 160
140 PN(1)=P(1)
   N(1)=N(1)
   N(5)=UN(3)*P(2)-UN(2)*P(3)
   DELTA=UN(2)**2-UN(3)**2
   P(2)=(-N(4)*UN(2)-D(5)*UN(3))/DELTA
   P(3)=(UN(2)*N(5)-UN(3)*N(4))/DELTA
160 CONTINUE
C DETERMINE PORTION OF TIME SEGMENT USED TO TRAVEL FROM P TO PB.
C
C DTIME=PR/VPP
162 IF(DTIME,LT,0) GO TO 163
   DTIME=(6,1010)
161 IF(DTIME/(24*DTIME) IS GREATER THAN H)
   HUNIT=1
   RETURN
C EXTEND LINE PN-PB THROUGH 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 I I=1,3
   VT(1)=(PP(1)-PN(1))/DTIME
165 VN(1)=(P(1)-P(1))/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 FSTCC(VN(4),VT(4),FTA)
170 I=I+1
   PNP(1)=PR(I)+FTA(I)*VT(I)*(H-DTIME)
   PP(1)=PR(I)-FTA(I)*VN(I)*(H-DTIME)
170 VP(I)=(PP(I)-PB(I))/(H-DTIME)
   T=T+4
   RETURN
C IF POINT P LIES ON SURFACE, USE PP POINT TO DETERMINE AFTER
C BOUNCE STATE.
C
180 CONTINUE
   DELTA=UN(1)**2+UN(1)*UN(2)**2+UN(1)*UN(3)**2
   IF(ABS(Delta).LT.1.0F-12) GO TO 220
   N(2)=PP(1)*UN(2)-PP(2)*UN(1)
   N(3)=PP(1)*UN(3)-PP(3)*UN(1)
   G(2,1)= UN(2)
   G(2,2)= UN(1)
   G(2,3)= C.0
   G(3,1)= UN(3)
   G(3,2)= C.0
   G(3,3)= UN(1)
   N= X10 K=1,3
   IF 1A) I=1,3

ORIGINAL PAGE IS OF POOR QUALITY
**VG LEVEL 21**

**POUNCE**

**DATE = 75182**

16/47/45

```
190  GS(1, J) = (1, J)
200  T = 1, 3
210  PS(K) = PS(1) / DETA
      GO TO 240
220  PN(1) = PP(1)
      U(1) = UN(1)
      U(2) = UN(2) - UN(1) * U(1)
      NETA = - UN(2) * U(1) - UN(3) * U(1)
      PN(2) = (- U(1) * U(2) - U(1) * U(3)) / NETA
      PN(3) = U(1) * U(1) / NETA
      CONTINUE
      C DETERMINE PORTION OF TIME SEGMENT REMAINING FOR TRAVEL FROM PP TO PP.
      DTIMF = H - PP / VPP
      IF (DTIMF > 1.0F-12) GO TO 245
      DTIMF = 1.0F-12
      1020  FORMAT (2H DTIMF LESS THAN ZERO)\n         NEXIT = 0
         RETURN
      C DETERMINE THE PROPER DISTANCE ALONG PP-PB TO FIND PNP.
      THEN EXTEND A LINE NORMAL TO THE SURFACE FROM PNP TO GET THE POINT
      AFTER POUNCE, PP.
      C FIND VELOCITY COORDINATES BASED ON PP, PB, AND THE TIME H.
      245  DS 250  I = 1, 3
         VT(1) = (PN(I) - PR(I)) / DTIMF
         VN(1) = (PP(I) - PN(I)) / DTIMF
         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 PESTORM(VN(4), VT(4), FTA)
      DO 260  I = 1, 3
      PNP(I) = PR(I) + FTA(I) * VT(I) * DTIMF
         PP(I) = PNP(I) - FTA(I) * VN(I) * DTIMF
      260  VP(I) = (PP(I) - PR(I)) / DTIMF
      T = T + H
         RETURN
      END
```
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

\[ \eta_N(\beta, V) = 1.0 - \psi_1(\beta) \psi_2(V) \]
\[ \eta_T(\beta, V) = 1.0 - \phi_1(\beta) \phi_2(V) \]  

(41)

In this expression, \( \eta_N \) and \( \eta_T \) are the normal and tangential restitution coefficients and \( \psi_1, \psi_2, \phi_1, \phi_2 \) 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 \( \psi_2 \) 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 $\psi_2$, the value of $\psi_1$ at $45^\circ$ was used. A value of $\psi_2$ equal to one was taken for incidence velocity of 76.2 meters/sec. The following expression was developed to express $\psi_2$:

$$\psi_2 = 0.65(1.0 - e^{-V/24.5})$$  \hspace{1cm} (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 experiment 1 data points that were used, it was assumed that the tangential restitution coefficient was equal to one for incidence angles of $0^\circ$ and $90^\circ$. 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 RISTCO, 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), B(10)
DATA A/1.0, 1.4, 0.5, 1.17, -49.25, 2, 1.31, 0.35, 2, 1.17, 0.872, 3, 4, 7.885, 16, 7297, 11, 2300, -1.979, 996, 1/
DATA B/5.72, 15, -41.8, 808, 17, 8, 168, 4, -424.3, 82, 572, 7, 631, -406, 662, 5, 7.14, 2, 70, 551, -50, 498, 2, 9, 676, 7/
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.

ETA = ETA**2(VN, VT)
V = SQRT(VN**2 + VT**2)
PHIKE = V/250.0
PHI = V/250.0
PHI = PHI!
A(1) = ETA(1) + ETA(2)
A(2) = ETA(1) + ETA(2)
A(3) = ETA(1) + ETA(2)
A(4) = ETA(1) + ETA(2)
A(5) = ETA(1) + ETA(2)
A(6) = ETA(1) + ETA(2)
A(7) = ETA(1) + ETA(2)
A(8) = ETA(1) + ETA(2)
A(9) = ETA(1) + ETA(2)
A(10) = ETA(1) + ETA(2)
B(1) = ETA(1) + ETA(2)
B(2) = ETA(1) + ETA(2)
B(3) = ETA(1) + ETA(2)
B(4) = ETA(1) + ETA(2)
B(5) = ETA(1) + ETA(2)
B(6) = ETA(1) + ETA(2)
B(7) = ETA(1) + ETA(2)
B(8) = ETA(1) + ETA(2)
B(9) = ETA(1) + ETA(2)
B(10) = ETA(1) + ETA(2)
C = PHI
D = PHI
E = PHI
F = PHI
G = PHI
H = PHI
PHI = PHI
IF (C GT .90) PHI = .90
IF (D GT .90) PHI = .90
IF (E GT .90) PHI = .90
IF (F GT .90) PHI = .90
IF (G GT .90) PHI = .90
IF (H GT .90) PHI = .90

24
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:

\[
C_D = 4.5 + \frac{24}{Re} \quad \text{for } Re < 1.0
\]

\[
C_D = 28.5 - 24.0 \log(Re) + 9.0682 \log^2(Re) - 1.7713 \log^3(Re) + 0.1718 \log^4(Re) - 0.0065 \log^5(Re)
\]

\[
\text{for } 1.0 < Re < 3000
\]

\[
C_D = 0.4 \quad \text{for } 3000 < Re < 2.5 \times 10^{-5} \quad (43)
\]

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 PNUMAP(PFNOLO, NGFC, CD)

IF (ABS(PFNOLO) .LT. 1.0E-12) PFNOLO = 1.0E-12

IF (PFNOLO .LT. 1.0) GO TO 26

IF ((PFNOLO .GE. 1.0) .AND. (PFNOLO .LE. 1.0E3)) GO TO 27

CD = NGFC * 0.4
RETURN

26 CD = NGFC * (4.5 + 24.0 / PFNOLO)
RETURN

27 ARF = ALLC(PFNOLO)

CD = (24.0 - 24.0 * ARF + 9.0682 * ARF**2 - 1.7713 * ARF**3 + 0.1718 * ARF**4
  - 0.0365 * ARF**5) * NGFC
RETURN
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-θ 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, $r$, $\theta$ 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, $\frac{W}{2\pi D} = \rho V_r r = \text{constant}$, 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 = 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 (110, 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. (198.5°R or 110°K).

**GAM** - Ratio of specific heats.

**RGAS** - Gas constant. (Length^2/Time^2 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. (Mass/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, \( \theta \). (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).
RENOls - 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 $\alpha_1 = \arctan \left( \frac{VRIN}{VUIN} \right)$ (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),YPS(6),VZ(4),TEMP(2),PHO(2)
DIMENSION STATE(18)
INTEGER RNGS
READ(5,1000) DANGLE,TMAX,RMIN,RMAX,ANGMAX,ZMAX
WRITE(6,2000) DANGLE,TMAX,RMIN,RMAX,ANGMAX,ZMAX
READ(5,1100) VISREFF,TRFF,TSUT,AM,PGAS,DFGC
WRITE(6,2100) VISREFF,TRFF,TSUT,AM,PGAS,DFGC
READ(5,1200) RIN,VPIN,VWIN,VZIN,TT,RHOT
WRITE(6,2200) RIN,VPIN,VWIN,VZIN,TT,RHOT
READ(5,1300) ALPHA,RETA,GRAVITY
WRITE(6,2300) ALPHA,RETA,GRAVITY
IF(ABS(VWIN).LT.1.0F-10) VWIN=1.0F-10
TALPH=VPIN/VUTN
SALPH=SIN(ALPHA/57.29578)
RETA=RETA/57.29578
C/VR=1.0/(GAM-1.0)
CP/VR=GAM/(GAM-1.0)
AMC=(GAM-1.0)/((GAM+1.0)
VCR=SQRT(2.0*AM*GAM*TT/(GAM+1.0))
WRITE(6,2400) VCR
DO 205 NUM=1,100
READ(5,1400) (STATE(I),I=1,18)
WRITE(6,2700) (STATE(I),I=1,18)
READ(5,1300) (YR(I),I=1,16)
WRITE(6,2300) (YR(I),I=1,16)
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=RHCRF*(2.0/(GAM+1.0)**(1.0)
DELTA=RHOP*DIA/RHOCR/0.3
TT=TT*2.0/(GAM +1.0)
VCR=VISREFF*(((TR/TREF)**1.5)*(TREF+TSUT)/(TREF+TSUT))
TAU=RHOP*DIA**2/18.0/VISCR
RF=0.1*RHOCR*VCR/VISCR/2.0
WRITE(6,2900) DELTA,TAU,REC
WRITE(6,2410) TEMP(2)=TT
RHOP(2)=RHOT
YR(3)=YR(3)/57.29577
VR(1)=VRIN
VU(1)=VWIN
VZ(1)=VWIN
N=6
L=-1
M=1
T=0.0

34
ANGLE=YR(3)
TS=T
I=10
VR(I)=VR(1)
VZ(I)=VZ(1)
10 VU(I)=VU(1)
20 NAT=1
20 YPS(I)=VR(I)
30 CALL VOPTFX(L,VRIN,Vuin,VZIN,RIN,TEMP,RHO,GAM,RGAS)
L=L+1
VISTAR=VISOFF*(((TEMP(1)/TREF)**1.5)*TPFF+TSUT)/((TEMP(1)+TSUT)
C INTEGRATE USING RUNGE-KUTTA METHOD
25 VI0FF=SQRT(((VR(2)-YR(2))**2+(VU(2)-YR(1)*YR(4))**2
1 +(VZ(2)-YR(6))**2)
REN0LD=RHC(1)*VIOFF*DIA/VISTAR
CALL RNUMPR(RENO,LO,CO)
RCON=RHC(1)*CO/RHOP/DIA/1.66667
30 IF(RUNGFN,YK,FR,T,H1,NE,1) GO TO 40
FR(1) = YR(2)
FR(2) = VR(1)*YR(4)**2+PON*VI0FF*(VR(2)-YR(2))
1 -GRAVITY*CONS(YR(3)+BETA)
FR(3) = YR(4)
FR(4) = 2.0*YR(2)*YR(4)/YR(1)+RC0N*VI0FF
1 *(VU(2)-YR(1)*YR(4))/YR(1)
2 +GRAVITY*SIN(YR(3)+BETA)
FR(5) = YP(6)
FR(6) = 3.0*V10FF*(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)
1F(L.EQ.10000) GO TO 205
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.1.0F-4).AND.(ABS(VU(4)-VU(3)),LT.1.0F-4)
1 .AND.(ABS(VZ(4)-VZ(3)),LT.1.0F-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)=VP(4)
VU(3)=VU(4)
VZ(3)=VZ(4)
T=TS
45 IF(NGT.100) GO TO 200
M=M+1
GO TO 25
C COMPLETE INTEGRATION STEP, IF REQ'D, WRITE OUTPUT, GO TO 30
DO \nM=1 \nL=L+1 \nTS=T \n
THETA=VR(3)*57.29578 \n\n60 \nVR(1)=VR(1) \nVR(2)=1.5*VR(4)-0.5*VR(1) \nVR(3)=2.0*VR(4)-VR(1) \nVR(4)=VR(3) \n\nVU(2)=1.5*VU(4)-0.5*VU(1) \nVU(3)=2.0*VU(4)-VU(1) \nVU(4)=VU(3) \n\nVZ(2)=1.5*VZ(4)-0.5*VZ(1) \nVZ(3)=2.0*VZ(4)-VZ(1) \nVZ(4)=VZ(3) \n\nVISTAD=VISTAD*(((TEMP(1)/TPFF)**1.5)*(TEFF+TSUT))/(TEMP(1)+TSUT) \n\nIF((THETA.GT.TMAX).OR.(VR(1).LT.MIN).OR.(VR(1).GT.ZMAX)) GO TO 200 \nIF((THETA-ANGLE).LE.0.0) GO TO 30 \n\nRTD=VR(4) \nSLINF=EXP(VR(3)*TALPH) \nVRIN=VR(1)/KIN \nWRITE(6,2500) T,VR(1),VR(2),THETA,RTD,VR(5),VR(6),RENCLD,ROIN, \n1 \nSLINF \nANGLE=ANGLE+DANGLE \nGO TO 25 \n\n200 CONTINUE \nRTD=VR(4) \nSLINF=EXP(VR(3)*TALPH) \nVRIN=VR(1)/KIN \nWRITE(6,2600) T,VR(1),VR(2),THETA,RTD,VR(5),VR(6),RENULD,ROIN, \n1 \nSLINF \nWRITE(6,2800) M,L \n\n205 CONTINUE \n1000 FORMAT(7F10.4) \n1100 FORMAT(F20.5,F10.4) \n1200 FORMAT(5F10.4,F10.4) \n1300 FORMAT(6F10.6) \n1400 FORMAT(1HA4) \n1500 FORMAT(6F10.4) \n1510 FORMAT(110,F10.3) \n2000 FORMAT(1HO,7X,7HO-ANGLE,10X,5HT-MAX,10X,5HR-MIN,10X,5HM-MAX,6X, \n1 \n2100 FORMAT(1HO,5X,14HFF, VISCOSITY,6X,9HRFF, TEMPP,2X,13HSOTHERLANDS T \n1 \n2110 FORMAT(1HO,10X,4HRFF,12X,3HP1A,F,6F15.4)) \n2120 FORMAT(1HO,9X,5HSSTEP,14X,1HH,/,(15,F15.4)) \n2200 FORMAT(1HO,11X,3HPIN,11X,4HPVIN,11X,4HVUTN,11X,4HVZIN,13X,2HTT, \n1 \n36
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 VORTEX(L, VR, VU, VZ, POS, T, RHO, GAM, VGAS)

DIMENSION T(7), RHO(2)

M=0
IF(L.GE.0) GO TO 10
N=0
CVR=1.0/(GAM-1.0)
ABC=(GAM-1.0)/(GAM+1.0)
VCR=SQRT(2.0*VGAS*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**C.VOR)
CU=VU*POS
CH=-V*POS
C=V*POS
RETURN
10 IF(M.GT.100) GO TO 20
V*P=VR
VUP=VU
TOTT=1.0-ABC*((VU**2+VR**2+VZ**2)/(VCR**2))
IF(TOTT.LE.0.0) GO TO 15

V=CU*POS
VR=VCR*((1.0/TOTT)**C.VOR)/POS

M=M+1
IF((ABS(VUP-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**C.VOR)

A=VCR*SQRT((VU**2+VR**2+VZ**2)/(GAM*VGAS*T(1)))
IF(A.MAT.GT.1.0) GO TO 30
GO TO 40
15 WRITE(6,1200) POS
L=100
RETURN
20 WRITE(6,1000) M, VR, VPP, VU, VUP, TOTT
GO TO 40
30 IF(N.EQ.0) WRITE(6,1100)
N=1
40 CONTINUE
1000 FORMAT(110,5E15.8)
1100 FORMAT(140,11HCHKFED FLOW)
RETURN
END
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.
### PARDIM Example Case for NASA Report

<table>
<thead>
<tr>
<th>CARD</th>
<th>SET</th>
<th>V-PART/V-GAS</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.0</td>
<td>0.0264</td>
<td>0.2467, 0.2742, 369.0, 0.0264</td>
</tr>
<tr>
<td>0.106E-4</td>
<td>0.175.48</td>
<td>492.0, 198.2, 1.4</td>
</tr>
<tr>
<td>0.2741</td>
<td>-184.0</td>
<td>780.0, 0.0, 518.7, 0.764E-1</td>
</tr>
<tr>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
</tbody>
</table>

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

<table>
<thead>
<tr>
<th>CARD</th>
<th>SET</th>
<th>V-PART/V-GAS</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.2741</td>
<td>-92.0</td>
<td>0.0, 1422.84, 0.001, 0.0</td>
</tr>
<tr>
<td>187.2</td>
<td>0.7761E-3</td>
<td>0.0</td>
</tr>
<tr>
<td>10</td>
<td>0.1E-4</td>
<td></td>
</tr>
</tbody>
</table>

Additional particle trajectories require a Card Set 2 arrangement here. These sets are stacked one after the other.
<table>
<thead>
<tr>
<th>P-ANGLE</th>
<th>T-MIN</th>
<th>R-MIN</th>
<th>R-MAX</th>
<th>MAX-ANGLE</th>
<th>Z-MAX</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.0000</td>
<td>0.0032</td>
<td>0.2467</td>
<td>0.2742</td>
<td>360.0000</td>
<td>0.6264</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>REF. VISCOSITY</th>
<th>REF. TEMP</th>
<th>SURF THERMAL T</th>
<th>GAMMA</th>
<th>GAS CONST</th>
<th>CRGT FACTOR</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.10600E-04</td>
<td>492.0000</td>
<td>132.2000</td>
<td>1.4000</td>
<td>1715.4000</td>
<td>1.0000</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>PIN</th>
<th>VMIN</th>
<th>VLIN</th>
<th>VZIN</th>
<th>TT</th>
<th>R-FCT</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.2741</td>
<td>-1.040000</td>
<td>786.0000</td>
<td>0.0</td>
<td>518.7000</td>
<td>0.7640E-01</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>ALPHA</th>
<th>REF. ACC. OF GRAV.</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.0</td>
<td>0.0</td>
</tr>
</tbody>
</table>

**CRITICAL VELOCITY = 101.8826**
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 \( r, \theta \) 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, RNUMBR, 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:

\[
\begin{align*}
\text{GAM, RGAS, WTFL, RHOIP, TIP} & \quad (5F10.6) \\
D(1), A(1), REXIT, SRDEND & \quad (4F10.6)
\end{align*}
\]

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
VISREF, TREF, TSUT, DGFC
RHOP, DIAP, H, TMAX, ETA(1), ETA(2)
YR(I), I = 1, 6
ALPHA, BETA, GRAVITY
NSTEP
```

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 \((\text{Length}^2/\text{Time}^2 \text{ Degree Abs.})\)
- **WTFL** - Mass flow rate of gas \((\text{Mass}/\text{Time})\)
- **RHOIP** - Gas inlet stagnation density \((\text{Mass}/\text{Length}^3)\)
- **TIP** - Gas inlet stagnation temperature \((\text{Degree Abs.})\)
- **D(1)** - Dimension of the first station as indicated in Figure 20. \((\text{Length})\)
- **A(1)** - Area of the first station. \((\text{Length}^2)\)
- **REXIT** - Radial distance that determines exit location. See Figure 20. \((\text{Length})\)
- **SRDEND** - Angular location of the last portion of the regular scroll contour before suppression of the scroll begins. See Figure 20. \((\text{Degrees})\)
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/Length^3)

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)

GRAVTY - 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 CHl, where

\[
CHl = \arctan \left( \frac{D(1)}{2\pi \times \text{REXIT}} \right)
\]

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:

\begin{itemize}
  \item V - Gas velocity (Length/Time)
  \item VR - Radial Component of gas velocity (Length/Time)
  \item VU - Tangential component of gas velocity (Length/Time)
  \item CHl - Angle between V and VU. (Degrees)
  \item VCR - Critical Velocity. (Length/Time)
  \item RHO - Gas density (Mass/Length³)
  \item TEMP - Gas temperature (Degrees Abs.)
\end{itemize}
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.

DELT A - 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)

RENO LD - 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,WTFL,RHOIP,TIP
WRITE(6,2010) GAM,RGAS,WTFL,RHOIP,TIP
READ(5,1020) D(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)*1.0.E-12) D(I)=D(I-1)*SQR(A(I)/A(I-1))
  IF(A(I)*1.0.E-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=2,NSTART,37
    AVG=360.0-THETA(I,2)
25  R(I,2)=REXIT*SQR(1.0+TAN(ANG/57.29577)**2)
    R(I,1)=REXIT
    R(I,2)=REXIT+D(I)
    THETA(I,1)=0.0
    THETA(I,2)=0.0
    EXPON=1.0/(GAM-1.0)
    VCR=SQR(2.0*GAM*RGAS*TIP/(GAM+1.0))
    CIRCUM=2.0*3.1415927*REXIT
    CHI=ATAN2(D(I),CIRCUM)

DETERMINE GAS VELOCITY

M=1
  VU=WTFL/RHOIP/A(1)
  VR=-VU*TAN(CHI)
  VEST=SQR(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-WTFL).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=WTFL/RHO/A(1)
    VR=-VU*TAN(CHI)
    VEST=SQR(VR**2+VU**2)/VCR
    GO TO 30
I/O

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

PRINT OUT SOLUTION.

CHIW=CHI*57.29578
WRITE(6,2030) V,VV,VR,VU,CHIW,VCR,RHO,TEMP
DO 50 I=1,37
   WRITE(6,2040) I,THETA(I,1),THETA(I,2),R(I,1),R(I,2)
   DO 60 K=1,2
      WRITE(6,2040) I,THETA(I,K),57.29578
   60 END
   WRITE(6,2040) I,THETA(I,1)/57.29578
   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) 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)
   YR(3)=YR(3)/57.29578
   READ(5,1000) ALPHA,BETA,GRAVITY
   WRITE(6,3090) ALPHA,BETA,GRAVITY
   READ(5,3080) NSTEP
   WRITE(6,4080) NSTEP
   RHOCR=RHOP*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+TSJ)/(TCR+TSUT)
   TAU=RHOP*DIAP**2.180/VISCR
   RECRI=DIAP/RHOCR*VCR/VISCR/2.0
   WRITE(6,5020) DELTA,TAU,RECRI
   WRITE(6,4090)

CHECK THAT PARTICLE STARTS INBOUNDS.

NOUT=0
   RW=REXIT+D(1)*SORT(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
R PART = 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
VISSTAR = VISREF * (TREF + TSUT) / (T + TSUT)

INTEGRATE USING RUNGE-KUTTA METHOD.

625 IF(YR(3) .LT. SREND) 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)
RENOLO = RHO * VDIFF * DIAP / VISSTAR
CALL RNUMLO(RENOLO, DGFC, CD)
BCON = RHO / CD / RHO / 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) - GRAVITY * 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 + GRAVITY * SIN(YR(3) + BETA) / YR(1)
FR(5) = YR(6)
FR(6) = BCON * VDIFF * (VZ - YR(6)) - GRAVITY * 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) .LT. 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 * DIAP * SQRT(1.0 - YR(3) / 6.2831854)
IF(YR(1) .GE. RW) NOUT = 7
IF(YR(3) .LT. SREND) GO TO 650
IF(NOUT .EQ. 0) GO TO 700
LEVEL 21       MAIN

DATE = 75045   15/17/25

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)=RWS*COS(YR(3))
B(2)=RWS*SIN(YR(3))
B(3)=YR(5)
GO TO 660

650 XTEST=YR(1)*COS(YR(3))
IF(XTEST.LT.REXIT) GO TO 700
NOUT=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)*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(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)*COS(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=RWS
DO 760 I=1,N

760 YRS(I)=YR(I)
VTIME=NTIME-1

780 IF((NOUT.EQ.1).OR.(NOUT.EQ.3).OR.(NOUT.EQ.8).OR.(T.GT.TMAX))
1 GO TO 800

1
IIF((ABS(RY(1)) .LT. 1.0E-4) .AND. (ABS(RY(4)) .LT. 1.0E-4)) GO TO 800
1 (ABS(RY(6)) .LT. 1.0E-4) GO TO 800

IF(NTIME.GT.0) GO TO 625

NTIME=STEP-1
IF(L.GT.1) NTIME=STEP

RY3=RY(3)*57.29578
WRITE(6,4050) L,T,RY(1),RY(2),RY(3),RY(4),RY(5),RY(6),RENVOL
GO TO 625

800 CONTINUE
RY3=RY(3)*57.29578
WRITE(6,4060) L,T,RY(1),RY(2),RY(3),RY(4),RY(5),RY(6),RENVOL
GO TO 105

810 WRITE(6,2050)

C FORMAT STATEMENTS
C
1300 FORMAT(7F10.4)
1010 FORMAT(5F10.6)
1020 FORMAT(4F10.6)
2010 FORMAT(1HL,5X,21HINLET SCROLL SOLUTION,//,12X,3HNGAM,11X,4HRGAS,
1 11X,4HWTFL,10X,5HRHDIP,12X,3HTIP,/,(5F15.6))
2020 FORMAT(1HO,10X,4HD(1),11X,4HAI(1),10X,5HPEXIT,9X,6HSPDEND,
1 /,(4F15.6))
2030 FORMAT(1HO,13X,1HV,13X,2HVR,13X,2HUV,12X,3HCH1,12X,3HVRP,13X,
1 3HRD,11X,4HTFMP,/,(7F15.6),/,-11H ORTHOGONAL,5X,10HTHETA(I,1),
2 5X,10HTHETA(I,2),9X,6HR(I,1),"X,6HR(I,2))
2040 FORMAT(1HL,2F15.4,2F15.6)
2050 FORMAT(37HOGAS FLOW SOLUTION FAILED TO CONVERGE)
2060 FORMAT(1HO,7E15.8)
3000 FORMAT(184)
3310 FORMAT(F20.5,3F10.3)
3020 FORMAT(F10.2,3F10.4,2F10.3)
3330 FORMAT(6F10.3)
3080 FORMAT(15)
3090 FORMAT(1HO,9X,5HALPHA,11X,4HBETA,8X,7HGRAVITY,/,(3F15.4))
4000 FORMAT(1HL,184)
4100 FORMAT(1HO,13X,5HVIRREF,11X,4HTKFF,11X,4HTSUT,11X,4HOGEC,/,
1 (E20.5,3F15.3))
4020 FORMAT(1HO,13X,4HRHCP,11X,4HDIA,14X,1H4,11X,4HTMAX,10X,5HETAT-
1 10X,5HETAT-7,/,(F15.2,3F15.4,2F15.3))
4030 FORMAT(1HO, 9X,5HYR(1),10X,5HYR(2),10X,5HYR(3),10X,5HYR(4),
1 10X,5HYR(5),10X,5HYR(6),/,(6F15.6))
4040 FORMAT(1HO,62H PARTICLE NOT IN PASSAGE AT FIRST POINT GIVEN, GO TO
1 NEXT CASE)
4050 FORMAT(1H,15,E10.2,7F15.5)
4060 FORMAT(1HO,15,E10.2,7F15.5)
4080 FORMAT(17H0PRINT DATA EVERY,17,2X,7HSTEPS(5))
4090 FORMAT(1HO,3X, 1HL, 9X,1HT,10X,5HYR(1),10X,5HYR(2),10X,5HYR(3),
1 10X,5HYR(4),10X,5HYR(5),10X,5HYR(6),9X,6HRENL)
5000 FORMAT(1H,15,E10.2,6F15.5)
5020 FORMAT(32HOSIMILARITY PARAMETERS. DELTA =,F10.4,5X,5HTAU =,F12.4,
1 6X,6HRREC =,E12.4)
5030 FORMAT(1H 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 RUNJMBR(RENOLD, DGFC, CD)
I=1
IF(ABS(RENOLD).LT.1.0E-12) RENOLD=1.0E-12
IF(RENOLD.GT.1.0) GO TO 26
IF(RENOLD.LE.1.0) 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
-0.0065*ARE**5)*DGFC
RETURN
END
DATE = 75045 15/17/25

SUBROUTINE BOUNCE(A,R,C,P,PP,H,T,ETA,NFIX,PB,VP)


1 PB(3)

DIMENSION GS(3,3),VP(3),UN(3),PN(3)

DIMENSION ETA(3)

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).LT.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=UN(1)*V(1)*V(2)*V(3)+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(J,1)=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

70 GS(I,K)=G(I,K)

DO 60 J=1,3

60 GS(I,K)=D(J)

PB(K)=GS(1,K)*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
LEVEL 21  
BOUNCE  
DATE = 75045  
15/17/25  

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

80 PB(1)=P(1)
     DETR=-UN(2)*V(2)-UN(3)*V(3)
     IF ABS(DETR).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))/DETR
     PB(3)=(UN(2)*D(5)-V(3)*D(4))/DETR
     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 J=1,3
     DO 100 I=1,3
     110 GS(I,J)=G(I,J)
     DO 120 I=1,3
     120 GS(I,K)=G(I,K)
     PN(K)=GS(I,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)
     GO TO 160
     150 PN(1)=A(1)
     PN(2)=P(2)
     PN(3)=P(3)
     160 CONTINUE
LEVEL 21

DATE = 75045 15/17/25

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

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

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

DO 170 I=1,3
   S = (PB(I) - PN(I))/DTIME
   PNP = PB(I) + ETA(I)*S*(H - DTIME)
   SN = (PN(I) - P(I))/DTIME
   PP(I) = PNP - ETA(I)*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.
**SCRL2D** Example Case for NASA Report

<table>
<thead>
<tr>
<th>Set 1</th>
<th>Set 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.4 1715.48</td>
<td>0.000001</td>
</tr>
<tr>
<td>0.4 0.08</td>
<td>0.0</td>
</tr>
<tr>
<td>0.3615 316.175</td>
<td>518.7</td>
</tr>
</tbody>
</table>

**Example Case for NASA Report** V-PART/V-GAS = 50%

<table>
<thead>
<tr>
<th>Particles</th>
<th>Card Set 2</th>
<th>Card Set 1</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>0.10E-4 492.0</td>
<td>0.000001 518.7</td>
</tr>
<tr>
<td>187.2</td>
<td>0.388E-4 0.1E-4</td>
<td>0.000001 518.7</td>
</tr>
<tr>
<td>0.5615</td>
<td>0.0 0.000001</td>
<td>0.000001 518.7</td>
</tr>
<tr>
<td>0.0</td>
<td>0.0 32.174</td>
<td>32.174</td>
</tr>
</tbody>
</table>

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

<table>
<thead>
<tr>
<th>GAM</th>
<th>RGAS</th>
<th>WTPF</th>
<th>RHOIP</th>
<th>TIP</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.400000</td>
<td>1715.479980</td>
<td>0.486000</td>
<td>0.076400</td>
<td>518.699951</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>D(1)</th>
<th>A(1)</th>
<th>REXIT</th>
<th>SRDEND</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.400000</td>
<td>0.080000</td>
<td>0.361500</td>
<td>316.174805</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>V</th>
<th>VR</th>
<th>VU</th>
<th>CHI</th>
<th>VCR</th>
<th>RHO</th>
<th>TEMP</th>
</tr>
</thead>
<tbody>
<tr>
<td>80.950897</td>
<td>-14.039830</td>
<td>79.724136</td>
<td>9.987672</td>
<td>1018.882568</td>
<td>0.076199</td>
<td>518.154053</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>ORTHOGONAL</th>
<th>Theta(I,1)</th>
<th>Theta(I,2)</th>
<th>R(I,1)</th>
<th>R(I,2)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.0</td>
<td>0.0</td>
<td>0.361500</td>
<td>0.761500</td>
</tr>
<tr>
<td>2</td>
<td>10.00000</td>
<td>10.00000</td>
<td>0.361500</td>
<td>0.755905</td>
</tr>
<tr>
<td>3</td>
<td>20.00000</td>
<td>20.00000</td>
<td>0.361500</td>
<td>0.750230</td>
</tr>
<tr>
<td>4</td>
<td>30.00000</td>
<td>30.00000</td>
<td>0.361500</td>
<td>0.744471</td>
</tr>
<tr>
<td>5</td>
<td>40.00000</td>
<td>40.00000</td>
<td>0.361500</td>
<td>0.738623</td>
</tr>
<tr>
<td>6</td>
<td>50.00000</td>
<td>50.00000</td>
<td>0.361500</td>
<td>0.732684</td>
</tr>
<tr>
<td>7</td>
<td>60.00000</td>
<td>60.00000</td>
<td>0.361500</td>
<td>0.726648</td>
</tr>
<tr>
<td>8</td>
<td>70.00000</td>
<td>70.00000</td>
<td>0.361500</td>
<td>0.720510</td>
</tr>
<tr>
<td>9</td>
<td>80.00000</td>
<td>80.00000</td>
<td>0.361500</td>
<td>0.714266</td>
</tr>
<tr>
<td>10</td>
<td>90.00000</td>
<td>90.00000</td>
<td>0.361500</td>
<td>0.707910</td>
</tr>
<tr>
<td>11</td>
<td>100.0000</td>
<td>100.0000</td>
<td>0.361500</td>
<td>0.701434</td>
</tr>
<tr>
<td>12</td>
<td>110.0000</td>
<td>110.0000</td>
<td>0.361500</td>
<td>0.694833</td>
</tr>
<tr>
<td>13</td>
<td>120.0000</td>
<td>120.0000</td>
<td>0.361500</td>
<td>0.688098</td>
</tr>
<tr>
<td>14</td>
<td>130.0000</td>
<td>130.0000</td>
<td>0.361500</td>
<td>0.681221</td>
</tr>
<tr>
<td>15</td>
<td>140.0000</td>
<td>140.0000</td>
<td>0.361500</td>
<td>0.674194</td>
</tr>
<tr>
<td>16</td>
<td>150.0000</td>
<td>150.0000</td>
<td>0.361500</td>
<td>0.667004</td>
</tr>
<tr>
<td>17</td>
<td>160.0000</td>
<td>160.0000</td>
<td>0.361500</td>
<td>0.659641</td>
</tr>
<tr>
<td>18</td>
<td>170.0000</td>
<td>170.0000</td>
<td>0.361500</td>
<td>0.652092</td>
</tr>
<tr>
<td>19</td>
<td>180.0000</td>
<td>180.0000</td>
<td>0.361500</td>
<td>0.644342</td>
</tr>
<tr>
<td>20</td>
<td>190.0000</td>
<td>190.0000</td>
<td>0.361500</td>
<td>0.636373</td>
</tr>
<tr>
<td>21</td>
<td>200.0000</td>
<td>200.0000</td>
<td>0.361500</td>
<td>0.628166</td>
</tr>
<tr>
<td>22</td>
<td>210.0000</td>
<td>210.0000</td>
<td>0.361500</td>
<td>0.619698</td>
</tr>
<tr>
<td>23</td>
<td>220.0000</td>
<td>220.0000</td>
<td>0.361500</td>
<td>0.610943</td>
</tr>
<tr>
<td>24</td>
<td>230.0000</td>
<td>230.0000</td>
<td>0.361500</td>
<td>0.601869</td>
</tr>
<tr>
<td>25</td>
<td>240.0000</td>
<td>240.0000</td>
<td>0.361500</td>
<td>0.592439</td>
</tr>
<tr>
<td>26</td>
<td>250.0000</td>
<td>250.0000</td>
<td>0.361500</td>
<td>0.582607</td>
</tr>
<tr>
<td>27</td>
<td>260.0000</td>
<td>260.0000</td>
<td>0.361500</td>
<td>0.572317</td>
</tr>
<tr>
<td>28</td>
<td>270.0000</td>
<td>270.0000</td>
<td>0.361500</td>
<td>0.561498</td>
</tr>
<tr>
<td>29</td>
<td>280.0000</td>
<td>280.0000</td>
<td>0.361500</td>
<td>0.550060</td>
</tr>
<tr>
<td>30</td>
<td>290.0000</td>
<td>290.0000</td>
<td>0.361500</td>
<td>0.537882</td>
</tr>
<tr>
<td>31</td>
<td>300.0000</td>
<td>300.0000</td>
<td>0.361500</td>
<td>0.524797</td>
</tr>
<tr>
<td>32</td>
<td>310.0000</td>
<td>310.0000</td>
<td>0.361500</td>
<td>0.510569</td>
</tr>
<tr>
<td>33</td>
<td>320.0000</td>
<td>320.0000</td>
<td>0.361500</td>
<td>0.471905</td>
</tr>
<tr>
<td>34</td>
<td>330.0000</td>
<td>330.0000</td>
<td>0.361500</td>
<td>0.417424</td>
</tr>
<tr>
<td>35</td>
<td>340.0000</td>
<td>340.0000</td>
<td>0.361500</td>
<td>0.384700</td>
</tr>
<tr>
<td>36</td>
<td>350.0000</td>
<td>350.0000</td>
<td>0.361500</td>
<td>0.367077</td>
</tr>
<tr>
<td>37</td>
<td>360.0000</td>
<td>360.0000</td>
<td>0.361500</td>
<td>0.361500</td>
</tr>
</tbody>
</table>
### Example Case for NASA Report V-Part V-Gass - 508

<table>
<thead>
<tr>
<th>VISREF</th>
<th>TREF</th>
<th>TSUT</th>
<th>DGFC</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.10600F-04</td>
<td>492.000</td>
<td>198.200</td>
<td>1.000</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>RHOP</th>
<th>DIAP</th>
<th>H</th>
<th>TMAX</th>
<th>ETA-N</th>
<th>ETA-T</th>
</tr>
</thead>
<tbody>
<tr>
<td>187.20</td>
<td>0.3880F-04</td>
<td>0.1000E+04</td>
<td>0.1000E-05</td>
<td>1.000</td>
<td>1.000</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>YR(1)</th>
<th>YR(2)</th>
<th>YR(3)</th>
<th>YR(4)</th>
<th>YR(5)</th>
<th>YR(6)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.561500</td>
<td>0.0</td>
<td>0.000100</td>
<td>51.500000</td>
<td>0.0</td>
<td>0.0</td>
</tr>
</tbody>
</table>

**Alpha**

<table>
<thead>
<tr>
<th>Beta</th>
<th>Gravity</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.0</td>
<td>32.1740</td>
</tr>
</tbody>
</table>

**Print Data Every 100 Step(s)**

**Similarity Parameters**

- **Delta** = 0.4999
- **Tau** = 0.1638E-02
- **Recr** = 0.1002E 03

<table>
<thead>
<tr>
<th>I</th>
<th>T</th>
<th>YR(1)</th>
<th>YR(2)</th>
<th>YR(3)</th>
<th>YR(4)</th>
<th>YR(5)</th>
<th>YR(6)</th>
<th>REHOLD</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0.0</td>
<td>0.56150</td>
<td>0.0</td>
<td>0.00010</td>
<td>51.50000</td>
<td>0.0</td>
<td>0.0</td>
<td>4.15392</td>
</tr>
</tbody>
</table>

**Example**

<table>
<thead>
<tr>
<th>CASE</th>
<th>ETA-N</th>
<th>ETA-T</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.56150</td>
<td>0.0</td>
<td>0.00010</td>
</tr>
</tbody>
</table>

**Similarity**

- **Delta** = 0.4999
- **Recr** = 0.1002E 03

**Similarity**

- **Delta** = 0.4999
- **Recr** = 0.1002E 03

**Example**

<table>
<thead>
<tr>
<th>CASE</th>
<th>ETA-N</th>
<th>ETA-T</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.56150</td>
<td>0.0</td>
<td>0.00010</td>
</tr>
</tbody>
</table>

**Example**

<table>
<thead>
<tr>
<th>CASE</th>
<th>ETA-N</th>
<th>ETA-T</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.56150</td>
<td>0.0</td>
<td>0.00010</td>
</tr>
</tbody>
</table>
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 \( r, \theta, z \) 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:

\[
\text{TITLE} \\
\text{MX, KMX, NB}
\]
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/Length$^3$)
RGAS - The gas constant. (Length$^2$/Time$^2$ 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 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.
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), TEMP(4), RHOA(4), VSTAR(4),
2 YRS(6), FR(6), A(3), R(3), C(3), P(3), PP(3), PB(3), VP(3)

READ FLOW FIELD DATA

READ(5,1030) (STATE(I), I=1,18)
WRITE(6,2030) (STATE(I), I=1,18)
READ(5,1010) MX, KM, NB
WRITE(6,2010) MX, KM, NB
READ(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) NSMP, H
WRITE(6,2040) NSMP, H
WRITE(6,2130)
DO 10 I=1, MX
READ(5,1020) (R(I,K), K=1, KM)
10 WRITE(6,2013) (R(I,K), K=1, KM)
WRITE(6,2130)
DO 20 I=1, MX
READ(5,1020) (THETA(I,K), K=1, KM)
20 WRITE(6,2014) (THETA(I,K), K=1, KM)
WRITE(6,2130)
DO 30 I=1, MX
READ(5,1020) (V(I,K), K=1, KM)
30 WRITE(6,2015) (V(I,K), K=1, KM)
WRITE(6,2130)
DO 40 I=1, MX
READ(5,1020) (BETA(I,K), K=1, KM)
40 WRITE(6,2016) (BETA(I,K), K=1, KM)
VCR=SORT(1.0*GAMMA*RGAS*TIP/(GAMMA+1.0))
FIRST=FIRST/57.29577
DO 60 I=1, MX
DO 60 K=1, KM
THETA(I,K)=THETA(I,K)/57.29577
BETA(I,K)=BETA(I,K)/57.29577
60 V(I,K)=V(I,K)*VCR

READ PARTICLE DATA

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

\[ YR(3) = YR(3)/57.29577 \]
\[ DO 65 I = 1, NB \]
\[ TTEST = FIRST + 6.283186 * FLOAT(I-1)/FLOAT(NB) \]
\[ IF (TTEST > YR(3)) GO TO 66 \]
\[ 65 CONTINUE \]
\[ 66 J = I - 1 \]
\[ IPASS = J \]
\[ YR(3) = YR(3) - FIRST - 6.283186 * FLOAT(J)/FLOAT(NB) \]
\[ YR3 = YR(3)/57.29577 \]
\[ WRITE(6,2140) J, YR3 \]
\[ EXPON = 1.0/(\text{GAMMA}-1.0) \]
\[ RHO=\text{RHO}^ {(\text{GAMMA}+1.0)/**EXPO} \]
\[ DELTA=\text{RHO}^ {\text{DIAP}/\text{RHO}^ {\text{RCR}/0.3} \}
\[ TCR=TIP^ {2.0/(\text{GAMMA}+1.0)} \]
\[ VISCR=\text{VISREF}^{((\text{TCR}/\text{TREF})**1.5)*((\text{TREF}+\text{TSUT})/(\text{TCR}+\text{TSUT}))} \]
\[ TAU=\text{RHO}^ {\text{DIAP}/2.0/\text{VISCR}} \]
\[ RECR=\text{DIAP}^ {\text{RHO}^ {\text{RCR}}/\text{VCR}/\text{VISCR}^ {2.0} \}
\[ WRITE(6,2050) \text{DELTA, TAU, RECR} \]
\[ T=0.0 \]
\[ L=0 \]
\[ YR3 = YR(3)/57.29577 \]
\[ WRITE(6,2060) L, T, YR(1), YR(2), YR3, YR(4), YR(5), YR(6) \]
\[ CP=\text{GAMMA}^ {\text{RGAS}/(\text{GAMMA}-1.3)} \]
\[ \text{RPart} = \text{DIAP}^ {2.0} \]
\[ N=6 \]
\[ M=1 \]
\[ NTIME = \text{NSTEP} \]
\[ TS=T \]
\[ ALPHA=-90.0/57.29577 \]
\[ TRPS=\text{TREF}+\text{TSUT} \]

DETERMINE AIR VELOCITIES AND PROPERTIES AT PARTICLE LOCATION

CALL DLOCAT(R, THETA, ZMAX, MX, KMX, YR, IP, KP, NOUT)
T=(NOUT, E0.0) 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, BETA, DD)
\[ V(1)=VPP^ {\text{COS}(\text{BETA})^ {\text{SIN}(\text{ALPHA})} \]
\[ VU(1)=VPP^ {\text{SIN}(\text{BETA})} \]
\[ VZ(1)=VPP^ {\text{COS}(\text{BETA})^ {\text{COS}(\text{ALPHA})} \]
\[ TEMP(1)=TIP^ {1.0-(\text{GAMMA}-1.0)/\text{GAMMA}^ {1.0}+1.0}\text{VPP}/\text{VCR}^ {2.0} \]
\[ RHA(1)=\text{RHO}^ {\text{TIP}^ {\text{TEMP}(1)/\text{TIP}^ {**EXPO} \}
\[ VISTA(1)=\text{VISREF}^{((\text{TEMP}(1)/\text{TREF})+1.5)*((\text{TRPS}))/\text{TEMP}(1)+\text{TSUT}) \]
CALL RESET(VR, 1)
CALL RESET(VU, 1)
CALL RESET(VZ, 1)
CALL RESET(TEMP, 1)
CALL RESET(RHA, 1)
CALL RSET(VISTAR, I)
DO 90 I=1,N
90 YRS(I)=YR(I)

C INTEGRATE USING RUNGE-KUTTA METHOD
C
C
100 VDIFF=Sqrt(((VR(2)-YR(2))^2+((VU(2)-YR(1))*YR(4))^2
1 +((VZ(2)-YR(6))^2)
RENDL=RHODA(2)*VDIFF*DIAP/VISTAR(2)
CALL RNUMBR(RENDL, DGF,C,CD)
BCON=RHODA(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)-YP(2))
FR(3)=YR(4)
FR(4)=-2.*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 JETERNINE IF WALL INTERACTION OCCURRED.
C
C
CALL DLOCAT(R, THETA, ZMAX, MX, KMX, YR, IP, KP, NOUT)
IF(NOUT.EQ.O) GO TO 150
IF(NOUT.EQ.1) OR (NOUT.EQ.3) GO TO 200
GO TO 125

124 P=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)=A(3)
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
Determine air values at new location.

150 CALL POLATE(R, TET, V, YR(1), YR(3), IP, KP, VPP, DD)
   CALL POLATE(R, TET, RETA, YR(1), YR(3), IP, KP, RETAP, DD)
   YR(4) = VPP * COS(RETAP) * SIN(ALPHA)
   VU(4) = VPP * SIN(BF Tap)
   VZ(4) = VPP * COS(RETAP) * COS(ALPHA)
   TEMP(4) = TIP * (1.0 - ((GAMMA-1.0)/(GAMMA+1.0)) * (VPP/VCR)**2)
   RHO(4) = RHO(1) * (TEMP(4)/TIP)**EXPO
   VISTAR(4) = VISREF * (((TEMP(4)/TREF)**1.5)*(TRPS)/(TEMP(4)+TSUT))

Test air values use), if incorrect, reset integration values and
   go to 100. If correct, go to 170.

   I = (ABS(YR(4) - YR(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(TEMP, 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

Completed integration step.

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(TEMP, 3)
   CALL RESET(RHOA, 3)
   CALL RESET(VISTAR, 3)
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 3) GO TO 100
YR3 = YR(3) * 57.29577
WRITE(6,2100) L,T,YR(1),YR(2),YR3,YR(4),YR(5),YR(6),RENOLO
NTIME = NSTEP-1
IF(L.GT.0) NTIME = NSTEP
GO TO 100

200 CONTINUE
YR(3) = YR(3) * 2831853/FLOAT(NR)*FLOAT(IPASS)
WRITE(6,2110)
WRITE(6,2100) L,T,YR(1),YR(2),YR3,YR(4),YR(5),YR(6),RENOLO
WRITE(6,2120) M
GO TO 50

C FORMAT STATEMENTS.

1010 FORMAT(14I5)
1020 FORMAT(8F10.6)
1030 FORMAT(18A4)
1040 FORMAT(E20.4,3F10.6)
1050 FORMAT(I5,E10.2)
2010 FORMAT(1MO,15HFLOW FIELD DATA, //, 8X, 2HMX, 7X, 3HMX, 8X, 24NB, //, 1 (3110))
2011 FORMAT(1HO, 9X, 5GAMMA, 12X, 3HTIP, 10X, 5HRH1P, 11X, 4HRGAS, //, (4F15.6))
2012 FORMAT(1HO, 10X, 4HZMAX, 13X, 5HFRST, //, 2(F15.6))
2013 FORMAT(1H, 4X, 6HRH(I,K), (8F15.6))
2014 FORMAT(1H, 10HT-EITA(I,K), (8F15.6))
2015 FORMAT(1H, 4X, 6HVIN(I,K), (8F15.6))
2016 FORMAT(1H, 1X, 9HET(I,K), (8F15.6))
2030 FORMAT(1HL, 18A4)
2031 FORMAT(1HO, 13X, 6HVISREF, 11X, 4HTREF, 11X, 4HTSUT, 11X, 4HOGFC, //, 1 (F20.4, 3F15.6))
2032 FORMAT(1HO, 15X, 4HDIAPI, 11X, 4HRH1P, //, (E20.4, F15.6))
2033 FORMAT(1HO, 9X, 5HYR(1), 10X, 5HYR(2), 10X, 5HYR(3), 10X, 5HYR(4), 10X, 5HYR(5), 10X, 5HYR(6), //, (6F15.6))
2040 FORMAT(1HO, 4X, 54NSTEP, 14X, 1HF, //, (110, E15.2))
2050 FORMAT(32HO, SIMILARITY PARAMETERS. DELTA = F10.4, 5X, 5HSFMJ = E12.4, 1 5X, 6HREC = E12.4)
2060 FORMAT(1HO, 6X, 4HSTEP, 9X, 1HT, 5X, 5HYR(1), 5X, 5HYR(2), 5X, 5HYR(3), 5X, 1 5HYR(4), 5X, 5HYR(5), 5X, 5HYR(6), //, 4X, 6HRENOLO, //)
2070 FORMAT(1HO, 43HPARTICLE OUT OF BOUNDS AT FIRST POINT GIVEN)
2080 FORMAT(11HBOUNCE OFF, 14, 6X, F10.5, 10X, F10.3, 10X, F10.5)
2110 FORMAT(1HO)
2120 FORMAT(3HOM =, I4)
2130 FORMAT(1HL)
2140 FORMAT(24HOPARTICLE ENTERS PASSAGE, 15, 5X, 24HINLET ANGLE CORRECTED

STOP
END
SUBROUTINE DLCAT(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=1
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)

THIS SUBROUTINE RESETS THE VALUES OF A VARIABLE SO THAT THE
BEST ESTIMATE OF THE AVERAGE VALUE OF THE VARIABLE CAN BE USED IN
THE INTEGRATION STEP.
A(1) IS THE VALUE OF A AT POINT 1 FOR THE INTEGRATION STEP.
A(2) IS THE AVERAGE VALUE OF A OVER THE INTEGRATION STEP.
A(3) IS THE VALUE OF A AT POINT 2 USED IN THIS INTEGRATION STEP.
A(4) IS THE VALUE OF A AT POINT 2 CALCULATED AFTER THE INTEGRATION.

GO TO (10, 20, 30), I

FOR I=1, SET UP ARRAYS.
10 DO 15 J=2, 4
15 A(J)=A(1)
RETURN

FOR I=2, RESET AVERAGE VALUE OF A, A(2), AND REMEMBER THE LATEST
VALUE OF A AT THE END OF THE STEP.
20 A(2)=(A(4)+A(1))/2.
A(3)=A(4)
RETURN

FOR I=3, SYSTEM HAS CONVERGED. ESTIMATE THE AVERAGE VALUES BY
LINEARLY EXTENDING THE VALUES DETERMINED IN THE PREVIOUS POINTS.
30 A(2)=1.5*A(4)-0.5*A(1)
A(3)=2.0*A(4)-A(1)
A(1)=A(4)
A(4)=A(3)
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.
SUBROUTINE RESTCO(VN, VT, ETA)
DIMENSION ETA(2), A(10), B(10)
DATA A(1), A(2), A(3), A(4), A(5), A(6), A(7), A(8), A(9), A(10)
1, -24.47885, -15.7297, 11.2303, -1.979996
DATA B(1), B(2), B(3), B(4), B(5), B(6), B(7), B(8), B(9), B(10)
1, 87.1428, 70.6511, -50.4987, 9.67677

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.

ETA = ATAN2(VN, VT)
V = SQRT(VN**2 + VT**2)
PHIONE = V / 250.00
P + ITWO = A(1) * ETA + A(2) * BETA + A(3) * ETA**2 + A(4) * ETA**3
1 + A(5) * ETA**5 + A(6) * BETA**6 + A(7) * ETA**7 + A(8) * BETA**8
2 + A(9) * BETA**9 + A(10) * BETA**10

PHI = PHI + P * PHI + PHI + PHI
1 + PHI = 1.000 - PHI
PHI = 1.000 - PHI
ETA(2) = 1.000 - ETA
PSIONE = 1.0000 - EXP(-V/36.000)
PSITWO = B(1) * ETA + B(2) * BETA + B(3) * ETA**2 + B(4) * ETA**3
1 + B(5) * ETA**5 + B(6) * BETA**6 + B(7) * ETA**7 + B(8) * ETA**8
1 + B(9) * ETA**9 + B(10) * ETA**10
PSI = PSIONE + PSI + TWO
1 + PSI = 1.000 - PSI
ETA(1) = 1.000 - PSI
RETURN
END

SUBROUTINE RNJVAR(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 .LE. 1.0E3) GO TO 27
CD = DGFC * 0.4
RETURN
26 CD = DGFC * (4.5 + 24.0 / RENOLD)
RETURN
27 CAR = ALOG(RENOLD)
CC = (28.5 - 24.0 * CAR + 9.0682 * CAR**2 - 1.7713 * CAR**3 + 0.1718 * CAR**4
1 + 0.0665 * CAR**5) / DGFC
RETURN
END
SUBROUTINE POLATE(P, Z, A, AP, ZP, IP, KP, AP, DD)
DIMENSION R(21, 21), Z(21, 21), A(21, 21), D(2), DD(2)

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 = (R(IA, KP) - R(IA, KP - 1)) / (Z(IA, KP) - Z(IA, KP - 1))
A1 = R(IA, KP - 1) - AM * Z(IA, KP - 1)
R2 = RP + ZP / AM
ZA = (B2 - B1) * AM / (AM**2 + 1.0)
RA = R2 - ZA / AM
GO TO 10
5 QA = 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)**2 + D(2)**2 + A(IP, KP - 1)) / DT
AB = (D(1)**2 + A(IP, KP)**2 + D(2)**2) / DT
DO 20 K = 1, 2
KA = KP - K + 1
RC = (D(1) + D(IP - 1, KA)**2 + R(IP, KA)) / DT
ZC = (D(1)**2 + D(IP - 1, KA)**2 + Z(IP, KA)) / DT

20 DD(K) = SQRT((RP - RC)**2 + (ZP - ZC)**2)
DT = DD(1) + DD(2)
AP = (DD(1)**2 + DD(2)**2 + AB) / DT
RETURN
END
SUBROUTINE BOUNCE(A,H,C,P,PP,H1,ETA,NFIX,PR,VP)
1 PB(3),
DIMENSION GS(3,3),VP(3),UN(3),PN(3)
DIMENSION ETA(2)
DIMENSION PN(3),VN(4),VT(4)
C
C NFIX=1
C 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(2)+UN(3)*V(1)*V(3)
C(1)=UN(1)*A(1)+UN(2)*A(2)+UN(3)*A(3)
D(2)=V(2)**2-V(1)*P(2)
D(3)=V(3)**2-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)
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

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

80 PH(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)*P(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
DPB= 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)
DPBR=SQRT((PP(1)-PB(1))**2+(PP(2)-PB(2))**2+(PP(3)-PB(3))**2)
IF(DPBR.LT.DPB) GO TO 103

103 CONTINUE
IF(DPB.LT.0.5*DPB) 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
C DETA=JN(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)= JN(3)
G(3,2)=0.0
G(3,3)=UN(1)
DO 130 K=1,3
DO 110 J=1,3
DO 110 I=1,3
110 GS(I,J)=G(I,J)
DO 120 I=1,3
DO 120 J=1,3
120 GS(I,J)=0.0
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)
     D(5)=UN(3)*P(2)-UN(2)*P(3)
     DET=UN(2)**2-UN(3)**2
     P(1)=(-D(4)*UN(2)-D(5)*UN(3))/DE
     P(3)=(UN(2)*D(5)-UN(3)*D(4))/DE
160 CONTINUE

C DETERMINE PORTION OF TIME SEGMENT USED TO TRAVEL FROM P TO PB.
C
C DTIME=DB/VPP
C IF(DTIME.LT.H) GO TO 163
WRITE(6,1010)
1010 FORMAT(24H DTIME IS GREATER THAN H)
NJUT=1
RETURN

C EXTENT 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)=(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(I)*VT(I)*(H-DTIME)
     PP(I)=PNP(I)-ETA(I)*VN(I)*(H-DTIME)
170 VP(I)=(PP(I)-PN(I))/(H-DTIME)
     T=T+H
     RETURN
C IF POINT P LIES ON SURFACE, USE POINT PP TO DETERMINE AFTER
C BOUNCE STATF.
C
180 CONTINUE
     DET=UN(1)**2+UN(2)**2+UN(3)**2
     IF(ABS(DET).LE.1.E-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
210 DO 190 I=1,3

77
DO 190 I=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 PN(I)=PP(I)
     D(4)=D(1)
     D(5)=UN(3)*PN(2)-UN(2)*PN(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

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

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

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

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)*VN(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

**Radial Stator Case for Thesis and NASA Report**

<table>
<thead>
<tr>
<th>Card Set 1</th>
<th>Card Set 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Radial Stator Case for Thesis and NASA Report</td>
<td>Example Case for NASA Report V-Part/V-Case = 50%</td>
</tr>
<tr>
<td>10 5 29</td>
<td>0.774E-4 187.2</td>
</tr>
<tr>
<td>1.4 518.7 0.0748 1715.48</td>
<td>0.3209 -107.84 328.97 205.95 0.0132 0.0</td>
</tr>
<tr>
<td>0.106E-4 492.0 198.2 1.0</td>
<td>Additional particle trajectories require a Card Set 2 arrangement here.</td>
</tr>
<tr>
<td>0.0264</td>
<td>These sets are stacked one after the other.</td>
</tr>
<tr>
<td>10 0.1E-5</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>-------</td>
<td>-------</td>
</tr>
<tr>
<td>GAMMA</td>
<td>TIP</td>
</tr>
<tr>
<td>1.400000</td>
<td>1.1889951</td>
</tr>
<tr>
<td>VISRF</td>
<td>TREF</td>
</tr>
<tr>
<td>0.1060E-04</td>
<td>462.000000</td>
</tr>
<tr>
<td>TMAX</td>
<td>FIRST</td>
</tr>
<tr>
<td>0.074400</td>
<td>12.410000</td>
</tr>
<tr>
<td>NSTEP</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>0.10E-05</td>
</tr>
<tr>
<td>STEPS</td>
<td>VF1</td>
</tr>
<tr>
<td>-------</td>
<td>-----</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0.5</td>
</tr>
<tr>
<td>2</td>
<td>0.5</td>
</tr>
<tr>
<td>3</td>
<td>0.5</td>
</tr>
<tr>
<td>4</td>
<td>0.5</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>VF11</td>
<td>0</td>
</tr>
<tr>
<td>VF12</td>
<td>0</td>
</tr>
<tr>
<td>VF13</td>
<td>0</td>
</tr>
<tr>
<td>VF14</td>
<td>0</td>
</tr>
<tr>
<td>VF15</td>
<td>0</td>
</tr>
<tr>
<td>VF16</td>
<td>0</td>
</tr>
</tbody>
</table>

**Variables:**
- **VF**: Variable Factors
- **VF1** to **VF16**: Specific variable factors for each step.
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 RBBD 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 BCOP 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
VISREF, TREF, TSUT, DGFC
RHOP, DIAp, H, TMAX, ETA-N, ETA-T
YR(I), I = 1, 6
NSSTEP
```

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. (°R).

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

**DGFC** - Drag factor. The drag coefficient based on Reynolds'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.
REYNOLDS - 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.
USE OF ARBITRARY QUASI-ORTHOGONALS FOR CALCULATING FLOW DISTRIBUTION IN THE MERIDIONAL PLANE OF A TURBOMACHINE.

THEODORE KATSAKIS 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),SA(21,21),SBETA(21,21),
3TH(21,21),TT(21,21),WA(21,21),WTR(21,21),
DIMENSION AB(21),AC(21),AD(21),BA(21),DELBTA(21),DORDM(21),
1DTOR(21),DTZ(21),DWSDM(21),DWTDM(21),RH(21),RS(21),ZH(21),ZS(21),
2TH(21),WTFL(21),XR(21),XT(21),XZ(21),
INTEGER RUNO,TYPE,BCOP,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,BCOP,SRW,NULL,TEMP,ALM,RHO,TOLER,PLOSS
WRITE(6,1010) TYPE,BCOP,SRW,NULL,TEMP,ALM,RHO,TOLER,PLOSS
READ(5,1010) MTHTA,NPRT,ITER,NULL,SMFACT,ZSPLIT,ETIN,RB,CORFAC
WRITE(6,1010) MTHTA,NPRT,ITER,NULL,SMFACT,ZSPLIT,ETIN,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.
RH(I)=RH(I)/12.
20 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(I,1)
      Z(I,K)=DN(I,K)/DN(I,KMX)*(ZS(I)-ZH(I))+ZH(I)
   30   CONTINUE
   DO 40 I=1,MX
      R(I,1)=DN(I,1)/DN(I,KMX)*(RS(I)-RH(I))+RH(I)
   40   CONTINUE
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)
50 CONTINUE
```
IV G LEVEL 21

MAIN

DATE = 75155

11/27/18

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)
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)

END OF INPUT STATEMENTS.

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

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(I,K) = 0.0
BA(I) = 0.0
DO 120 K = 1,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(144.0/GAM*AR*TEMP)
WRITE (6,1050) CI
KMXM1 = KMX-1
CP = AR/GAM/(GAM-1.0)
EXPON = 1.0/(GAM-1.0)
BETIN = -BETIN/57.29577
RINLET = (RS(1)+RH(1))/2.0
```
G LEVEL 21  

DATE = 75155  

11/27/18

CEF = SIN(BETIN)/COS(BETIN)/RINLET/(RINLET-RB)**2
ERROR = 100000.0

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 START OF CALCULATIONS OF PARAMETERS.
C
DO 230 K = 1,KMX
DO 160 I = 1,MAX
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(I,K),CURV(I,K))
DO 170 I = 1,MAX
CURV(I,K) = CURV(I,K)/(1.+AL(I,K)**2)**1.5
AL(I,K) = ATAN(AL(I,K))-0.785398
CALL(1,K) = COS(AL(I,K))
170 SAL(I,K) = SIN(AL(I,K))
DO 180 I = 2,MAX
SAL(I,K) = SAL(I-1,K)+SQR((Z(I,K)-Z(I-1,K))**2+(R(I,K)-R(I-1,K))**2)
190 CALL SPLINEXRT(I,THTA(I),MTHTA,Z-I,K,MX,DTDZ(I))
DO 220 I = 1,MAX
CALL LININT(Z(I,K),R(I,K),XZ,XR,TN,21,21,T)
IF(R(I,K).LE.RB) GO TO 200
DTR(I) = CEF*(R(I,K)-RB)**2
GO TO 210
200 DTR(I) = 0.0
210 TQ = R(I,K)*DTR(I)
TP = R(I,K)*DTDZ(I)
TT(I,K) = T*SQR(TP+TP)
BETA(I,K) = ATAN(TP*CAL(I,K)+TQ*SAL(I,K))
SBET(I,K) = SIN(BET(I,K))
CBET(I,K) = COS(BET(I,K))
SA(I,K) = CBET(I,K)**2*CAL(I,K)*CURV(I,K)-SBET(I,K)**2/F(I,K)
1*SAL(I,K)*CBET(I,K)*SBET(I,K)*DTR(I)
SC(I,K) = SAL(I,K)*CBET(I,K)**2*CURV(I,K)+SAL(I,K)*SBET(I,K)
1*SAL(I,K)*DTR(I)
AB(I) = WA(I,K)*SBET(I,K)
220 AC(I) = WA(I,K)*SBET(I,K)
CALL SPLINE(SM(I,K),AB,MX,DWDM,AD)
CALL SPLINE(SM(I,K),AC,MX,DWTDM,AD)
IF(ITER.LE.0) AND (MOD(K-1,NEPRT).EQ.0)) WRITE(6,1080) K
DO 230 I = 1,MAX
SB(I,K) = SAL(I,K)*CBET(I,K)*DWDM(I)-2.*W*SBET(I,K)+DTPD(I)
1*R(I,K)*CBET(I,K)*DWT(I)**2.*W*SBET(I,K)
1*CAL(I,K)*CBET(I,K)*DWDM(I)+DTDZ(I)*R(I,K)*CBET(I,K)
1*(DWT(I)+2.*W*SBET(I,K))
IF((ITER.GT.0).OR. (MOD(K-1,NEPRT).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 CALCULATE BLADE SURFACE VELOCITIES. (AFTER CONVERGENCE.)

IF(ITER.NE.0) GO TO 260
DO 250 K = 1,KMX
CALL SPLINE(SM(I,K),TT(I,K),MX,DELBTA,AC)
A=XN
DO 240 I = 1,MX
240 AB(I)=RI(I,K)*W+WA(I,K)*SBETA(I,K)*(6.283186*R(I,K)/A-TT(I,K))
CALL SPLINE(SM(I,K),AB,MX,DRDM,AC)
IF(ISFACT.LE.1.0) GO TO 245
A = SFACT*XN
DO 244 I = 1,MX
244 ABI=IRI I,KI*W+WA(I,K)*SBETA(I,K)**16.2B3186*RI I,K)/A-TTII,KII
CALL SPLINE(SMI1,KI,ABI,MX,AO,AC)
245 00 250 I = 1,-MX
BETAO=BETA(I,K)-DELBTA(I)/2.
BETAT = BETA+DELBTA(I)
COSB = COS(BETAO)
COST = COS(BETAT)
IF(Z(I,K)*LT.ZSPLIT) DRDM(I) = AD(I)
WTR(I,K)=COSB*COST/(COSB+COST)*(2.*WA(I,K)/COSB+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.

C 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,J)-Z(I,K)
WAS=WA(I,J)*SA(I,J)*HR+SC(I,J)*HZ+SB(I,J)*HR+SD(I,J)*HZ
WASS=WA(I,J)*SA(I,K)*HR+SC(I,K)*HZ+SB(I,K)*HR+SD(I,K)=HZ
300 WA(I,K)=(WASS+WAS)/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=(Z**2)*X*W**2*AR/(AR*TIP)**2)*2./CP/TEMP
DENSX=TIP**X*EXPON*AR/CP/TEMP**EXPON*PLOSS/AR/TIP1P/TEMP
1*32.17*SMX/(SMX**2)*K
PRS(I,K)=DENSX*AR*TIP*TEMP/32.*17/144.
IF(ZS(I)-ZS(I)-Z(I)/ZS(I)-Z(I))-1.5708
GO TO 320
320 PSI=ATAN((RS(I)-RH(I))/ZS(I)-ZH(I))
GO TO 330
330 WTHRU = WA(I,K)*COS(PSI-AL(I,K))
A = XN
IF(Z(I,K).LT.ZSPLIT) A = SFCT*XN
C=6.283186*RS(I,K)-A*TT(I,K)
340 AD(K)=DENSX*WTHRU*C
CALL INTEGRAL(AC(I),AD(I),KMX,WTFL(I))
IF(ABS(WTFL(KMX)).LE.WTOLER) GO TO 350
CALL CONTINUE(WA(I,1),WTFL(KMX),IND,1,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 END OF LOOP - WEIGHT FLOW CALCULATION.

C CALCULATE STREAMLINE COORDINATES FOR NEXT ITERATION.
C
DO 380 K = 2,KMX1
DO 380 I = 1,KMX
Z(I,K)=DN(I,K)/DN(I,KMX)*(ZS(I)-ZH(I))*Z(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))/(2.*ROOT
390 AC(I) = (Z(I,K)+R(I,K))/(2.*ROOT
CALL SPLINE(AB,AC,MX,AL(I,K),CURV(I,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 (6,1100)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, (CN(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 CARDS SET FOR READING AND WRITING BETWEEN QUASI-ORTHOGONAL PROGRAM
C AND TRAJECTORIES PROGRAM. WRITE SET.
C
PUNCH 7000, MX,KMX,MTHTA
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 7777 K=1,KMX
7777 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)
CONTINUE
7000 FORMAT(7110)
7010 FORMAT(5E14.6)

C END OF CALCULATION OF NEXT STREAMLINE COORDINATES.
C
C FORMAT STATEMENTS.
C
1010 FORMAT(I5,5F10.4)
1011 FORMAT(F10.5)
1020 FORMAT(1H1,THRUN NO.,I3,10X,23INPUT 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(1H0,7X,13HITERATION NO.,I3)
1070 FORMAT(1H ,6X2HAL,12X,ZHRC,12X,2HS,12X,4HBT,10X,2HTT,12X,2HSA,
1 12X,2HSS,12X,2HSC,12X,2HSO)
1080 FORMAT(1H ,2X,10HSTREAMLINE,13)
1090 FORMAT(9F14.6)
1100 FORMAT(1H1,9X,2HDN,18X,1HZ,19X,1HR,19X,2HWA,18X,3HMP,16X,3HTR,
1 14X,2HTC)
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 END OF FORMAT STATEMENTS.
END
INTEGER RUNGF
DIMENSION R(21,21),Z(21,21),W(21,21),AL(21,21),RFTA(21,21),
L TT(21,21),THTA(21),XT(21)
DIMENSION STATE(18),YR(6),YRS(6),WR(4),WU(4),WZ(4),PHGA(4),
L TEMPA(4),VISTAP(4),FTA(2),DD(2),AP(2),AZ(2)
DIMENSION SM(21,21),FR(6),RA(2),ZA(2),THFTA(7)
C
C READ QUASI-ORTHOGONAL RESULTS - WRITE SIGNIFICANT PARTS.
C
CALL 3ARREAD(MX,KMX,MTHTA)
WRITE(6,4040) MX,KMX,MTHTA
CALL RORFAD(GAMMA,TEMP,RHO)
CALL RORFAD(RGAS,SFAC,TSPLIT)
CALL RORFAD(W,XN,ALM)
CALL RORFAD(PLOSS,ANULL,ANull) 
WRITE(6,4050) GAMMA,TEMP,RHO,RGAS,SFAC,TSPLIT,W
WRITE(6,4090) XN,ALM,PLOSS
CALL RCRREAD(R)
CALL RCRREAD(Z)
CALL RCRREAD(WA)
CALL RCRREAD(WL)
CALL RCRREAD(RFTA)
CALL RCRREAD(S4)
CALL RRREAD(THTA)
CALL 3PRFAD(XT)
WRITE(6,4070)
N=10 T=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) VISPRE,TREF,TSET,OGFC
WRITE(6,4010) VISPRE,TREF,TSUT,OGFC
READ(5,3020) RHOP,DIAP,H,TMAX,FTA(1),FTA(2)
WRITE(6,4020) RHOP,DIAP,H,TMAX,FTA(1),FTA(2)
READ(5,3030) (YR(I),I=1,6)
WRITE(6,4030) (YR(I),I=1,6)
READ(5,5010) NSTFP
WRITE(6,5020) NSTFP
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
Determine air velocities and properties at particle location.

Set up to start trace, initialize.

CALL ALCCAT(R,Z,MX,KMX,YR,THTA,XT,MTHTA,XN,ZSPLIT,IP,JP,KP,NOUT)
IF(NOUT.EQ.0) GO TO 606
WPTF(6,4080)
GO TO 605

606 CALL POLATE(R,Z,WA,YR(I),YR(5),IP,KP,WAP,DD)
CALL POLATE(R,Z,REFAT,YR(I),YR(5),IP,KP,HTAP,DD)
CALL POLATE(R,Z,SM,YR(I),YR(5),IP,KP,ALPP,DD)
WP(I)=WAP*COS(BFTAP)*SIN(ALPP)
WU(I)=WAP*SIN(BFTAP)
WZ(I)=WAP*COS(BFTAP)*COS(ALPP)
SMT=(SM(I)+SM(MX,KP)+DD(2)*SM(MX,KP-1))/DD(1)+DD(2)
TIP=1.0-(WAP**2+2.0*W*ALM-(W*YR(I))**2)/2.0/CP/TFMP
PPIP=1.0-(2.0*W*ALM-(W*YR(I))**2)/2.0/CP/TFMP
RHOA(I)=TIP**(EXPON*RHO-(TIP/TPPIP)**EXPON*PLCSS/RGAS/TPPIP/TFMP
1 =32.17*SM/PMT
TFMPA(I)=TIP*TFMP
VISTAR(I)=VISTARF*((TFMPA(I)/TRFF)**1.5)*(TRFF+TSUT)/(TFMPA(I)
1 +TSUT)

Initialize for first step.

610 I=1,4
WP(I)=WR(I)
WU(I)=WU(I)
WZ(I)=WZ(I)
RHOA(I)=RHOA(I)
TFMPA(I)=TFMPA(I)
610 VISTAR(I)=VISTAR(I)
620 YRS(I)=YR(I)

Integrate using Runge-Kutta method.

625 VDIFF=SQRT((WP(2)-YR(2))**2+(WU(2)-YR(1))**4)**2
1 +(WZ(2)-YR(6))**2)
RENOLO=RHOA(2)*VDIFF*DIAP/VISTAR(2)
ALL RNUM=RENOLO,CFGC,CD)
HCUN=RHOA(2)*CD/PHOP/RPART/2.33333
630 IF(RUNGF(N,YR,FR,T,H),NE.1) GO TO 640
FR(1)=YR(2)
FP(2)=YR(1)*YR(4)**2+2.0*YR(1)*W*YR(4)+YR(1)**2
1 +ACUN*VDIFF*(WR(2)-YR(2))
FR(3)=YR(4)
FP(4)=-2.0*YR(2)*YR(4)/YR(1)-2.0*W*YR(2)/YR(1)
1 +ACON*VDDFF*(WJ(2)-YR(1))*YR(4))/YR(1)
FR(5)=YR(6)
FR(6)=ACON*VDDFF*(WZ(2)-YR(6))
GO TO 630

640 CONTINUE

C DETERMINE IF WALL INTERACTION OCCURRED.

C
CALL ALCCAT(R,Z,MX,KMX,YR,THTA,XT,MTHTA,XN,ZSPLIT,IP,JP,KP,NOUT)
IF(NUIT.EQ.0) GO TO 700
IF(NUIT.EQ.1) GO TO 780
IF(NUIT.EQ.3) GO TO 780
I=1,6

645 YR(1)=YRS(1)
T=TS
IF(NOUT.NF.2) GO TO 650
PA(1)=R(IP,1)
PA(2)=R(IP-1,1)
ZA(1)=Z(IP,1)
ZA(2)=Z(IP-1,1)
CALL PBCH(YR,PA,ZA,T,H,ETA,NOUT)
GO TO 750

650 IF(NOUT.NF.4) GO TO 660
PA(1)=P(IP,KMX)
PA(2)=R(IP-1,KMX)
ZA(1)=Z(IP,KMX)
ZA(2)=Z(IP-1,KMX)
CALL PBCH(YR,PA,ZA,T,H,ETA,NOUT)
GO TO 750

660 THETA(1)=THTA(IP)
THETA(2)=THTA(IP-1)
ZA(1)=XT(IP)
ZA(2)=XT(IP-1)
IF(YR(5).GE.ZSPLIT) OTHET=3.1415927/XN
IF(YR(5).LT.ZSPLIT) OTHET=1.5707963/XN
CALL PBCH(YR,THETA,ZA,T,H,ETA,NOUT,OTHET)
GO TO 750

C DETERMINE AIR VALUES AT NEW LOCATION

C
700 CALL POLATE(R,Z,WAPER,YR(1),YR(5),IP,KP,WAP,nn)
CALL POLATE(R,Z,RETA,YR(1),YR(5),IP,KP,BFAP,nn)
CALL POLATE(R,Z,AL,YR(1),YR(5),IP,KP,ALPP,nn)
CALL POLATE(R,Z,SM,YR(1),YR(5),IP,KP,SMPP,nn)
WR(4)=WAP*COS(BFAP)*SIN(ALPP)
WU(4)=WAP*SIN(BFAP)
WZ(4)=WAP*COS(RETA)*COS(ALPP)
SM=(SM(1)+SM(2)*SM(MX,KP-1)/(SM(1)+SM(2))
TIP=1.0-(WAP**2+2.*W*ALM-(W*YR(1))**2)/2.0/CP/Temp
TPPI=1.0-(2.*W*ALM-(W*YR(1))**2)/2.0/CP/Temp
PHIA(4)=TIP*EXP(PHIN*RHO)-(TIP/TPPI)**EXP((PI*CSS/RGAS/TPPI/Temp
1 =32.17*SM/SMT
3=TA14(4)=TIP*Temp
VISTAR(4)=VISPFF*(((TFMPA(4)/TRFF)**1.5)*(TRFF+TSUT)/TFMPA(4)
1 *TSUT)

C TEST AIR VALUES USED, IF INCORRECT, RESET INTEGRATION VALUES AND GO
C 625. IF CORRECT, GO TO 750.
C
IF((ABS(WR(4)-WR(3)).LT.1.0F-4).AND.(ABS(WU(4)-WU(3)).LT.1.0F-4)
1 .AND.(ABS(WZ(4)-WZ(3)).LT.1.0F-4)) GO TO 750

WR(2)=(WR(4)+WR(1))/2.0
WU(2)=(WU(4)+WU(2))/2.0
WZ(2)=(WZ(4)+WZ(1))/2.0
WR(3)=WR(4)
WU(3)=WU(4)
WZ(3)=WZ(4)
RHIA(2)=(RHIA(4)+RHIA(1))/2.0
TFMPA(2)=(TFMPA(4)+TFMPA(1))/2.0
VISTAR(2)=(VISTAR(4)+VISTAR(2))/2.0
VISTAR(2)=(VISTAR(4)+VISTAR(1))/2.0
T=T

710 Y=Y+1
IF(Y.GT.100) GO TO 800
Y=Y+1
GO TO 625

C COMPLETE INTEGRATION STEP. IF REQUIRED, WRITE OUTPUT. GO TO 625
C
750 M=1
L=L+1
TS=T
GO TO 760, 1=1,N

760 YPS(1)=YP(1)
WP(2)=1.5*WR(4)-0.5*WR(1)
WR(3)=2.0*WP(4)-WP(1)
WP(1)=WR(4)
WP(4)=WP(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)
RHIA(2)=1.5*RHIA(4)-0.5*RHIA(1)
RHIA(3)=2.0*RHIA(4)-RHIA(1)
PHUA(1)=PHUA(4)
PHUA(4)=PHUA(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)

97
IVG LEVEL 21  MAIN  

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

NTIME=NTIME-1
780 IF((NOUT.FC).EQ.1).OR.(NOUT.EQ.3).OR.(T.GT.TMAX)) GO TO 800
IF(NTIME.EQ.10) GO TO 625
WRITE(6,2050) L,T,(YR(1),I=1,6),RFNOLD
NTIME=NSTEP-1
IF(L.EQ.1) 4TIME=NSTEP
GO TO 625
805 CONTINUE
WRITE(6,2050) L,T,(YR(1),I=1,6),RFNOLD
GO TO 605
2050 FORMAT(1H1,I10,F15.6,7F15.5)
2060 FORMAT(1H0,2I10)
3070 FORMAT(1HA4)
3080 FORMAT(F20.4,3F10.4)
3090 FORMAT(F10.4,F10.3,F10.3,3F10.4)
3100 FORMAT(6F10.5)
4000 FORMAT(1HI,18A4)
4010 FORMAT(1H0,13X,6HV1SPREF,11X,4HTREF,11X,4HTS,11X,4HNGFC,/,1)
4020 FORMAT(1H0,10X,4HDIAP,11X,4HDIAP,14X,1HM,11X,4HTMAX,10X,5MXTA-N,1
  10X,5MXTA-N,/, (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),/, (F15.5)
4040 FORMAT(1H0,44I IMPORTANT DATA FROM QUASI-CARTESIAN PROGRAM,/,1)
  1X,3HMX,7X,3HKMX,5X,5HTMA-TA,/, (3110)
4050 FORMAT(1H0,9X,5HGAMMA,11X,4HTEMP,12X,3HPHC,11X,4HRGAS,10X,1)
  5HGFACT, 9X,5HZSPLIT,14X,1HM,/, (7F15.4))
4060 FORMAT(1H0,2F15.5,5X,2F15.5)
4070 FORMAT(1H0,15HMX COORDINATES,27X,15HTIP COORDINATES,/,7X,1HR,1
  1X,1HM,1X,1HR,1X,1HZ)
4080 FORMAT(46H PARTICLE CUT OF PASSAGE AT FIRST POINT GIVEN)
4090 FORMAT(1H0,12X,7HXM,12X,3HLAM,10X,5HPLS,/,1)
  (3F15.4)
5010 FORMAT(1H0,5X,5HSTFP,14X,1HT,10X,5HYR(1),10X,5HYR(2),10X,5HYR(3),1
  1X,5HYR(4),1X,5HYR(5),10X,5HYR(6),9X,6HREFNOLD,1)
  (1F15.5,6F15.5)
5010 FORMAT(15)
5020 FORMAT(17H PRINT DATA EVERY 17*2X,7HSTFP(/))
5040 FORMAT(12H0THETA ARRAY,/(6F15.5))
5050 FORMAT(4HOXT ARRAY,/(6F15.5))
END
```
DIMENSION R(21,21),Z(21,21),P(21),J(21),THETA(21),XT(21)
MTA=MTA
/SP=MTA*7ST
NOUT=0
ON 10 I=1,MAX
IF(AHS(Z(I,1)-Z(I,KM),X,LS,THETA,XT)) LT.1,OF-12) GO TO 10
A=R(I,1)-K(1,LS)/Z(I,1)-Z(I,LK)
B=R(I,1)-K(I,LS)/Z(I,1)-Z(I,LK)
IF(AHSST,LS,THETA,XT)) GO TO 30
C1) TO 20
10 IF(Z(I,1),THETA,XT) GO TO 30
C2) CONTINUE
20 IF(T=1)
IF(T=1,THETA,XT) GO TO 40
NOUT=1
RETURN
40 IF(T=1,THETA,XT) GO TO 50
IF(Z(THETA,XT)) GO TO 50
NOUT=3
RETURN
50 CONTINUE
ON 70 K=1,MAX
IF(AHS(Z(IP,K)-Z(IP-1,K)),LS,THETA,XT)) LT.1,OF-12) GO TO 60
A=IP(IP-1,K)-P(IP,K)/Z(IP-1,K)-Z(IP,K)
B=R(IP,K)-A*Z(IP,K)
IF(AHSST,LS,THETA,XT)) GO TO 80
GO TO 70
60 IF(Z(IP,K),LS,THETA,XT)) GO TO 80
GO TO 70
70 CONTINUE
80 IF(T=1,THETA,XT) GO TO 90
NOUT=2
RETURN
90 IF(T=1,THETA,XT) GO TO 120
IF(AHS(Z(IP,K)-Z(IP-1,K)),LS,THETA,XT)) LT.1,OF-12) GO TO 110
IF(YK(1),LS,THETA,XT)) GO TO 120
NOUT=4
RETURN
110 IF(YK(1),LS,THETA,XT)) NOUT=4
RETURN
120 CONTINUE
ON 130 J=1,MTA
IF(XT(J),LS,THETA,XT)) GO TO 140
130 CONTINUE
140 IF(J=P)
A=THETA(JP-1)-THETA(JP)/(XT(JP-1)-XT(JP))
B=THETA(JP)-A*XT(JP)
THETA=THETA(JP)+A*B
IF(YK(1),LS,THETA,XT)) DT THETA=3.1415927/XN
IF(YK(J),LS,THETA,XT)) DT THETA=1.5707963/XN
99
7

1

I

I

RETURN

150 IF (YP(3).GT.THET2) GO TO 160

RETURN

END

SUBROUTINE POLATF(R,Z,AR,AP,ZP,IR,IP,KP,DP,DD)

DIMENSION P(21,21),Z(21,21),A(21,21),D(2),DD(2)

DO 10 J=1,2

1a=IP+1-1

IF(ABS(Z(IA,KP-1)-Z(IA,KP)).LT.1.0F-12) GO TO 5

IF(ABS(R(IA,KP)-R(IA,KP-1)).LT.1.0F-12) GO TO 6

AM=(P(IA,KP)-R(IA,KP-1))/(Z(IA,KP)-Z(IA,KP-1))

A1=R(IA,KP-1)-AM*Z(IA,KP-1)

A2=RP+ZP/AM

ZA=(A2-AM)*Z(I1-AM*(AM**2+1.0))

KA=NP-ZA/AM

5 GO TO 10

6 PA=RP

ZA=Z(IA,KP-1)

GO TO 10

6 KA=NP+1

ZA=ZP

10 N(1)=SORT((RA-PP)**2+(ZA-ZP)**2)

DT=O(1)+D(2)

A2=(O(1)*A(IP,KP-1)+D(2)*A(IP-1,KP-1))/DT

A3=(O(1)*A(IP,KP)+D(2)*A(IP-1,KP))/DT

N 20 K=1,2

KA=KP-K+1

PC=O(1)*R(IP,KP-1)+D(2)*R(IP-1,KP-1))/DT

ZC=O(1)*Z(IP,KP)+D(2)*Z(IP-1,KP))/DT

20 N0(K)=Sort((RP-RC)**2+(ZP-ZC)**2)

DT=O(1)+D(2)

AP=(O(1)*AA+D(2)*AB)/DT

END
SUBROUTINE SPLINT (X,Y,N,Z,MAX,YINT)
INTEGER N
DIMENSION X(50),Y(50),S(50),A(50),B(50),C(50),F(50),W(50),Z(50),
G(50),FM(50),Z(50),YINT(50)
COMMON Q

10 CONTINUE
N=1

20 I=2,N
A(I)=S(I)/6.0
B(I)=(S(I)+S(I+1))/3.0
C(I)=S(I+1)/6.0

50 F(I)=(Y(I+1)-Y(I))/S(I+1)-(Y(I)-Y(I-1))/S(I)
A(I)=.5
B(I)=1.0
C(I)=.5
F(I)=1.0
F(M)=0.0
A(I)=S(I)
S(I)=C(I)/W(I)
G(I)=0.0

30 I=2,N
K=K+1-1
40 F(K)=C(K)-S(K)*FM(K+1)
N=1
K=2
1F(7(I)-X(I)) 60,50,70
50 YINT(I)=Y(I)
G0 TO 90
60 IF(7(I).LT.1.1*X(1)-1*X(2))WRITE (6,1000)Z(I)
G0 TO 95
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
G0 TO 85
70 IF(Z(I)-X(K)) 85,75,80
75 YINT(I)=Y(K)
G0 TO 90
80 K=K+1
1F(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
MIX=MIXO(N,MAX)
1F(J.16) WRITE (6,1010) N,MAX,(X(I),Y(I),Z(I),YINT(I)),I=1,MIX}

101
SHEFIELD INP RACH(YR,P,Z,T,H,FTA,NOT)  
N=0: S=0  
IF (ABS(YR(2)) - LT.1.0F-12) GO TO 100  
IF (ABS(YR(1)) - LT.1.0F-12) GO TO 200  
S = (Y(2) - R(1))/((Z(2) - Z(1)))  
IF (ABS(YR(6)) - LT.1.0F-12) GO TO 10  
IF (ABS(YR(2)) - LT.1.0F-12) GO TO 20  
P=R*Y(2)/YR(6)  
Z=R*(Z(1) - YR(5))/PM + YR(1) - R(1)  
GO TO 30  
10  P=*YR(5)  
P=R*SM - (Z - Z(1)) + P(1)  
GO TO 30  
20  P=R*YR(1)  
Z=(P - R(1))/SM + P(1)  
GO TO 30  
100 IF (ABS(YR(1)) - LT.1.0F-12) GO TO 110  
P=R*YR(2)/YR(6)  
Z=R*Z(1)  
GO TO 30  
110 P=R*YR(1)  
Z=R*Z(1)  
GO TO 30  
200 P=R*P(1)  
P=R*P(1)  
Z=(P*YR(1)) + PM - YR(5)  
GO TO 30  
250 P=R*P(1)  
Z=R*YR(5)  
GO TO 30  
30  P*=SQR((RH - YR(1))**2 + (ZH - YR(5))**2)  
V= SQR((YR(2)**2 + YR(6)**2)  
T=H - 0.0P/VM  
T=R - YR(3) + YR(4)*CNP/VM  
ITC = (1000) NOT RP, TR, ZR  
GAMMA = ATAN2(YR(2) - R(1), (Z(2) - Z(1))) - 1.5707963  
ALPHA = 3.1415927 + ATAN2(YR(2), YR(6))  
V=VM*COS(GAMMA - ALPHA)  
VTR = VM*SIN(GAMMA - ALPHA)  
VTR = VM*F/T(1)  
VTR = VTR*FTA(1)  
Y(4) - YR(4)*FTA(2)  
Y(4) = SQR(VTR**2 + VTRP*2)  
FTA = ATAN2(VTRP - VNP)  
YR(2) = VTR* SIN(GAMMA + BFTA)  
YR(6) = VTR* COS(GAMMA + BFTA)  
YR(1) = RH + YR(2)*DT  
YR(3) = TR + YR(4)*DT  
YR(5) = ZH + YR(6)*DT  
T = T + H  
*FT=NUM  

102
IV C LEVEL 21                  DATE = 75182      16/55/51

SIMOUTTAEF       PARH(YR,THET,ZTH,FTRA,NJOT,DIHTHT)
DIMENSION YR(4),THET(2),Z(2),FTA(2)
DIMENSION Y(2)
IF(NJUT.*5) GO TO 6
4 1 = 1,2
4 Y(1) = (THET(1)+DTHET) * YR(1)
5 IF(NJUT.*6) GO TO 7
6 1 = 1,2
6 Y(1) = (THET(1)-DTHET) * YR(1)
7 CONTINUE
8 IF(AABS(Z(2)-Z(1)) LT .1 OF -12) GO TO 10
9 IF(AABS(Y(2)-Y(1)) LT .1 OF -12) GO TO 20
S = Y(2) - Y(1) / (Z(2) - Z(1))
10 IF(AABS(YR(4)) LT .1 OF -12) GO TO 30
11 IF(AABS(YR(6)) LT .1 OF -12) GO TO 40
PA = YR(4) * YR(1) / YR(6)
12 ZR = YR(4) * YR(1) / (PM*YR(5) - Y(1) + SM*Z(1)) / (SM - PM)
13 Y = SM* (ZR - Z(1)) + Y(1)
14 GO TO 10C
15 IF(AABS(YR(6)) LT .1 OF -12) GO TO 15
16 ZR = Z(1)
17 PA = YR(4) * YR(1) / YR(6)
18 Y = PM* (Z(1) - YR(5)) + YR(4) * YR(3)
19 GO TO 100
20 IF(AABS(YR(6)) LT .1 OF -12) GO TO 25
21 Y = Y(1)
22 PA = YR(4) * YR(1) / YR(6)
23 ZR = (YR(1) + YR(6))/PM
24 GO TO 100
25 Y = Y(1)
26 ZR = YR(5)
27 GO TO 100
30 YR = YR(1) * YR(3)
31 ZR = (YR - Y(1) + SM*Z(1)) / SM
32 GO TO 100
40 ZR = YR(5)
41 YR = SM* (ZR - Z(1)) + Y(1)
42 CONTINUE
43 PA = S*T( (YR - YR(1) * YR(3) ) * (ZR - YR(5) ) )
44 VY = S*T( (YR(4) * YR(1) ) * (YR(6) ) )
45 PT1 = S*P/VY
46 PT = H - PT1
47 T3 = YR / VY(1)
48 PH = YP(2) + PT1 + YR(1)
49 WRITE(6,1000) NJOT,YR,TR,ZR
GAMMA = ATAN2((YR(2) - Y(1)),(Z(2) - Z(1))) - 1.5707963
ALPHA = (YR(1)*YR(4) + YR(6))/2
VW = YR* CTS(GAMMA - ALPHA)
VT = YR* SINT(GAMMA - ALPHA)
VWD = -VA*FTA(1)
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 R8DUMP(MX,KMX,MTHTA)
WRITE(7,100) MX,KMX,MTHTA
100 FORMAT(3I5)
RETURN
END

SUBROUTINE R8READ(MX,KMX,MTHTA)
READ(5,100) MX,KMX,MTHTA
100 FORMAT(3I5)
RETURN
END

SUBROUTINE BCDUMP(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 BREAD(X)
DIMENSION X(21)
DO 10 K=1,19,3
10 READ(5,100) X(K),X(K+1),X(K+2)
100 FORMAT(3F20.8)
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,2HBB,2I3)
RETURN
END

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

SUBROUTINE PCRREAD(A,B,C)
READ(5,100) A,B,C
100 FORMAT(3E20.8)
RETURN
END
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.
<table>
<thead>
<tr>
<th>Iteration</th>
<th>Max. Streamline Change</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.222071</td>
</tr>
<tr>
<td>2</td>
<td>0.208135</td>
</tr>
<tr>
<td>3</td>
<td>0.16157</td>
</tr>
<tr>
<td>4</td>
<td>0.16130</td>
</tr>
<tr>
<td>5</td>
<td>0.16167</td>
</tr>
<tr>
<td>6</td>
<td>0.12432</td>
</tr>
<tr>
<td>7</td>
<td>0.10966</td>
</tr>
<tr>
<td>8</td>
<td>0.075628</td>
</tr>
<tr>
<td>9</td>
<td>0.089526</td>
</tr>
<tr>
<td>10</td>
<td>0.073936</td>
</tr>
<tr>
<td>11</td>
<td>0.065144</td>
</tr>
<tr>
<td>12</td>
<td>0.053730</td>
</tr>
<tr>
<td>13</td>
<td>0.04813</td>
</tr>
<tr>
<td>14</td>
<td>0.039612</td>
</tr>
<tr>
<td>15</td>
<td>0.035022</td>
</tr>
<tr>
<td>16</td>
<td>0.030901</td>
</tr>
<tr>
<td>17</td>
<td>0.027411</td>
</tr>
<tr>
<td>18</td>
<td>0.021477</td>
</tr>
<tr>
<td>19</td>
<td>0.019018</td>
</tr>
<tr>
<td>20</td>
<td>0.016639</td>
</tr>
<tr>
<td>21</td>
<td>0.014918</td>
</tr>
<tr>
<td>22</td>
<td>0.013239</td>
</tr>
<tr>
<td>23</td>
<td>0.011765</td>
</tr>
<tr>
<td>24</td>
<td>0.010545</td>
</tr>
<tr>
<td>25</td>
<td>0.009235</td>
</tr>
<tr>
<td>26</td>
<td>0.008263</td>
</tr>
<tr>
<td>27</td>
<td>0.007320</td>
</tr>
<tr>
<td>28</td>
<td>0.006539</td>
</tr>
<tr>
<td>29</td>
<td>0.005915</td>
</tr>
<tr>
<td>30</td>
<td>0.004620</td>
</tr>
<tr>
<td>31</td>
<td>0.003608</td>
</tr>
<tr>
<td>32</td>
<td>0.002612</td>
</tr>
<tr>
<td>33</td>
<td>0.001828</td>
</tr>
<tr>
<td>34</td>
<td>0.001026</td>
</tr>
<tr>
<td>35</td>
<td>0.000918</td>
</tr>
<tr>
<td>36</td>
<td>0.000628</td>
</tr>
<tr>
<td>37</td>
<td>0.00149</td>
</tr>
<tr>
<td>38</td>
<td>0.00129</td>
</tr>
<tr>
<td>39</td>
<td>0.00112</td>
</tr>
<tr>
<td>40</td>
<td>0.001026</td>
</tr>
<tr>
<td>41</td>
<td>0.000918</td>
</tr>
<tr>
<td>42</td>
<td>0.000628</td>
</tr>
<tr>
<td>43</td>
<td>0.00149</td>
</tr>
<tr>
<td>44</td>
<td>0.00129</td>
</tr>
<tr>
<td>45</td>
<td>0.00112</td>
</tr>
<tr>
<td>46</td>
<td>0.001026</td>
</tr>
<tr>
<td>47</td>
<td>0.000918</td>
</tr>
</tbody>
</table>

Stage: Speed of sound at inlet = 1.17 x 10^4
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°F 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

Data set punched by Quasi-Orthogonal Program.

<table>
<thead>
<tr>
<th>Example for NASA Radial Turbine V-PART/V-GAS = 50%</th>
</tr>
</thead>
<tbody>
<tr>
<td>SET 1</td>
</tr>
<tr>
<td><strong>Card 1</strong></td>
</tr>
<tr>
<td>1234567891011121314151617181920212223242526272829303132333435363738394041424344454647484950515253545556575859606162636465666768697071727374757677787980</td>
</tr>
</tbody>
</table>

**Card 2**

<table>
<thead>
<tr>
<th>SET 2</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Card 2</strong></td>
</tr>
<tr>
<td>1234567891011121314151617181920212223242526272829303132333435363738394041424344454647484950515253545556575859606162636465666768697071727374757677787980</td>
</tr>
</tbody>
</table>

**Card 3**

<table>
<thead>
<tr>
<th>SET 2</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Card 3</strong></td>
</tr>
<tr>
<td>1234567891011121314151617181920212223242526272829303132333435363738394041424344454647484950515253545556575859606162636465666768697071727374757677787980</td>
</tr>
</tbody>
</table>

Additionally, particle trajectories require a Card Set 2 arrangement here.

These sets are stacked one after the other.
<table>
<thead>
<tr>
<th>STIFF</th>
<th>Y</th>
<th>YR1(1)</th>
<th>YR2(1)</th>
<th>YR3(1)</th>
<th>YR4(1)</th>
<th>YR5(1)</th>
<th>YR6(1)</th>
<th>PENGOL</th>
<th>0.3000</th>
<th>0.4000</th>
<th>0.5000</th>
<th>PARTIAL PONTIFICATION AREA</th>
<th>PARTIAL PONTIFICATION AREA</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.0</td>
<td>0.2796</td>
<td>0.7976</td>
<td>-0.2076</td>
<td>96.7406</td>
<td>0.2172</td>
<td>0.2119</td>
<td>-0.2169</td>
<td>0.2360</td>
<td>0.2360</td>
<td>0.2360</td>
<td>0.2360</td>
<td>0.2360</td>
<td>0.2360</td>
</tr>
<tr>
<td>0.1</td>
<td>0.2796</td>
<td>0.7976</td>
<td>-0.2076</td>
<td>96.7406</td>
<td>0.2172</td>
<td>0.2119</td>
<td>-0.2169</td>
<td>0.2360</td>
<td>0.2360</td>
<td>0.2360</td>
<td>0.2360</td>
<td>0.2360</td>
<td>0.2360</td>
</tr>
<tr>
<td>0.2</td>
<td>0.2796</td>
<td>0.7976</td>
<td>-0.2076</td>
<td>96.7406</td>
<td>0.2172</td>
<td>0.2119</td>
<td>-0.2169</td>
<td>0.2360</td>
<td>0.2360</td>
<td>0.2360</td>
<td>0.2360</td>
<td>0.2360</td>
<td>0.2360</td>
</tr>
<tr>
<td>0.3</td>
<td>0.2796</td>
<td>0.7976</td>
<td>-0.2076</td>
<td>96.7406</td>
<td>0.2172</td>
<td>0.2119</td>
<td>-0.2169</td>
<td>0.2360</td>
<td>0.2360</td>
<td>0.2360</td>
<td>0.2360</td>
<td>0.2360</td>
<td>0.2360</td>
</tr>
<tr>
<td>0.4</td>
<td>0.2796</td>
<td>0.7976</td>
<td>-0.2076</td>
<td>96.7406</td>
<td>0.2172</td>
<td>0.2119</td>
<td>-0.2169</td>
<td>0.2360</td>
<td>0.2360</td>
<td>0.2360</td>
<td>0.2360</td>
<td>0.2360</td>
<td>0.2360</td>
</tr>
<tr>
<td>0.5</td>
<td>0.2796</td>
<td>0.7976</td>
<td>-0.2076</td>
<td>96.7406</td>
<td>0.2172</td>
<td>0.2119</td>
<td>-0.2169</td>
<td>0.2360</td>
<td>0.2360</td>
<td>0.2360</td>
<td>0.2360</td>
<td>0.2360</td>
<td>0.2360</td>
</tr>
</tbody>
</table>

**NOTES:**
- The table indicates values for different parameters, likely related to a specific scientific or engineering application.
- The data appears to be part of a larger set of measurements or calculations, possibly for a study or experiment.
- The units and specific context of the data are not immediately clear from the image alone.
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, \( \lambda = \frac{V}{r} \) along the streamlines, although the values \( \lambda \) 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 \( F(1) \) 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) (9F10.5)
LAMBDAB(1, 1) ... LAMBOA (1, KMX) (8F10.5)

**Group B**

TITLE (18A4)
VISREF, TREF, TSUT, DGFC (D20.5, 3F10.3)
RHOIP, 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. \((C_p g) \text{ (Length}^2/\text{Time}^2 - \text{Degrees Abs})\)
TIP - Inlet stagnation temperature. (Degrees Abs.)
RHOIP - Inlet stagnation density. (Mass/Length^3)
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 \(\alpha\) 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. (Length^2/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^oR or 110.0^oK.
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.

WTPLES    - 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)

\( \frac{V}{V_{cr}} \) 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

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 radius 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 input, the program uses a Runge-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. Cleveenger, 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),XR(6),YR(6),VR(4),VU(4),VZ(4),RHOA(4),
1 TEMPA(4),VISTAR(4),ETA(4)
DIMENSION F(6),RIZ(2),ZAI(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,WTFL
WRITE(6,2140) GAMMA,CP,TIP,RHOIP,WTFL
READ(5,1010) (RC(2,K),K=1,KMX)
WRITE(6,2170) KM,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(I,K),K=1,KMX)
WRITE(6,2150) KM,(LAMBDA(I,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(I,K)
IF(I.EQ.1) R(1,K)=R2-RC(2,K)
IF(I.EQ.MX) R(1,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(1,K)=Z1
IF(I.EQ.MX) Z(1,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)
LEVEL 21  MAIN  DATE = 75045  15/27/44

6 CONTINUE
DO 100 I=1,MAX
WRITE(6,2060) I
WRITE(6,2010)
WRITE(6,2000) (R(I,K),K=1,KMAX)
WRITE(6,2040)
WRITE(6,2000) (Z(I,K),K=1,KMAX)
WRITE(6,2020)
WRITE(6,2000) (RC(I,K),K=1,KMAX)
WRITE(6,2030)
WRITE(6,2000) (AL(I,K),K=1,KMAX)
WRITE(6,2050)
WRITE(6,2000) (LAMBD(A(I,K),K=1,KMAX)
XLENTH=DSORT((R(I,KMAX)-R(I,1))**2+(Z(I,KMAX)-Z(I,1))**2)
VCR=DSORT(2.0*CP*TIP*(GAMMA-1.0)/(GAMMA+1.0))
IND=1
NN=KMX-1
DELTAN=XLENTH/2=FLOAT(KMX-1)
VM=RHOIP/RHO/RLENTH/PI/(R(I,1)*R(I,KMAX))
VEST=DSORT(VM**2+(LAMBD(A(I,1)/R(I,1))**2)
DELTAN=VEST/20.*
10 V(I,1)=VEST
IF((V(1,1)**2).GT.2.0*CP*TIP) GO TO 45
RHO=RHOIP*(1.0-(V(1,1)**2)/(2.0*CP*TIP))**((1.0/(GAMMA-1.0)))
F(I,1)=RHO*VM*R(I,1)
VA=LAMBD(A(I,1)/R(I,1)
A=1./3C(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=LAMBD(A(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.*O
IF(VM2.GT.0.0) VM=DSORT(VM2)
IF((V(I,K+1)**2).GT.2.0*CP*TIP) GO TO 46
RHO=RHOIP*(1.0-(V(I,K+1)**2)/(2.0*CP*TIP))**((1.0/(GAMMA-1.0)))
20 F(I,K+1)=RHO*V4*R(I,K+1)
DO 30 K=1,KMX
30 F(I,K)=F(I,K)
CALL FNTGRL(DELTAN,F1,KMX,ANS)
WTFLS=ANS(KMX)*2.*PI
IF(DABS(WTFLS-2*PI.LT.0.00001*WTFL) GO TO 50
CALL CONTINUE(VEST,WTFLS,IND,JZ,WTFL,DELTAN)
IF(IND.LT.10) GO TO 10
IF(IND.NE.10)GO TO 40
WRITE(6,2110)WTFLS
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(LAMBD(A(I,K)/V(I,K)/R(I,K))/CONV
60 V(I,K)=V(I,K)/VCR
   WRITE(6,2120) NANGLES
   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) RHOIP,DIAP,H,TMAX,ETA(1),ETA(2)
   WRITE(6,4020) RHOIP,DIAP,H,TMAX,ETA(1),ETA(2)
   T=0.0
   READ(5,3030) (VY(I),I=1,6)
   WRITE(6,4030)(VY(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=RHOIP*DIAP/R-HOCR/O.3
   TCR=(2.0/(GAMMA+1.0))*TIP
   VISCR=VISREF*((TCR/TREF)**1.5)*((TREF+TSJT)/(TCR+TSUT))
   TAU=RHOIP*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(1,K)*CONV
   BETA(I,K)=BETA(I,K)*CONV
   V(I,K)=V(I,K)*VCR
110 CONTINUE

Determine air velocities and properties at particle location.
Set up to start trace and initialize.

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,VR(1),VR(5),IP,KP,VP,DD)
CALL POLATE(R,Z,BETA,VR(1),VR(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)
VR(1)=VP*DSIN(BETAP)
VZ(1)=VP*DCOS(BETAP)*DCOS(ALPP)
TEMPA(1)=TIP*(1.0-VP**2/CP/2.0/TIP)
RHOA(1)=RHOIP*(TEMPA(1)/TIP)**(1.0/GAMMA-1.0)
VISTAR(1)=VISREF*(TEMPA(1)/TREF)**1.5*(TREF+TSUT)/(TEMPA(1)+TSUT)
INITIALIZE FOR FIRST STEP.

\[ \text{RPART} = \text{DIAP}/2 \]
\[ V = 6 \]
\[ M = 1 \]
\[ \text{TIME} = -1 \]
\[ T = 0.0 \]
\[ TS = T \]

DO 610 \( i = 1, 4 \)
\[ VR(1) = VR(1) \]
\[ VU(1) = VU(1) \]
\[ VZ(1) = VZ(1) \]
\[ \text{RHOA}(1) = \text{RHOA}(1) \]
\[ \text{TEMPA}(1) = \text{TEMPA}(1) \]

610 \( \text{VISTAR}(1) = \text{VISTAR}(1) \)

DO 620 \( i = 1, N \)

620 \( \text{YRS}(1) = \text{YR}(1) \)

INTEGRATE USING RUNGE-KUTTA METHOD.

625 \[ \text{DIFF} = \text{DSORT}((\text{VR}(2) - \text{YR}(2))**2 + (\text{VU}(2) - \text{YR}(1))**2 + (\text{VZ}(2) - \text{YR}(4))**2) \]

\[ \text{RE} = \text{YR}(1)*\text{YR}(2) + \text{YR}(1)*\text{YR}(5) + \text{YR}(4)*\text{YR}(6) \]

\[ \text{RE} = \text{YR}(1)*\text{YR}(2) + \text{YR}(1)*\text{YR}(5) + \text{YR}(4)*\text{YR}(6) \]

IF \( \text{RUNGE}(N, \text{YR}, FR, T, H), \text{NE} = 1 \) GO TO 640

\[ FR(1) = VR(2) \]
\[ FR(2) = VR(1)*YR(4)**2 + BCON*\text{DIFF}*(VR(2) - YR(2)) \]
\[ FR(3) = YR(4) \]
\[ FR(4) = -2.0*YR(2)*YR(4)/YR(1) + BCON*\text{DIFF}*(VU(2) - YR(1))**2/YR(1) \]
\[ FR(5) = YR(6) \]
\[ FR(6) = BCON*\text{DIFF}*(VZ(2) - YR(6)) \]

GO TO 630

640 CONTINUE

DETERMINE IF WALL INTERACTION OCCURRED.

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

IF (NOUT.EQ.0) GO TO 700
IF (NOUT.EQ.1) GO TO 780

645 \( \text{VR}(1) = \text{YRS}(1) \)
\[ T = TS \]

IF (NOUT.EQ.2) GO TO 650

\[ \text{RA}(1) = \text{R}(IP, 1) \]
\[ \text{RA}(2) = \text{R}(IP, 1) \]
\[ \text{ZA}(1) = Z(IP, 1) \]
\[ \text{ZA}(2) = Z(IP, 1) \]

WRITE(6, '5000') NOUT

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

GO TO 750

650 CONTINUE
LEVEL 21  MAIN  DATE = 75045  15/27/44

RA(1)=R(IP,KMX)
RA(2)=R(IP-1,KMX)
ZA(1)=7(IP,KMX)
ZA(2)=2(IP-1,KMX)
WRITE(6,5000) NOUT
CALL RACH(VR,ZA,TA,T,H,FTA,NOUT)
GO TO 750

DETERMINE AIR VALUES AT NEW LOCATION.

700 CALL POLATE(R,Z, VR(1),YP(5),IP,KP,VP,DD)
CALL POLATE(R,Z,BETA,VR(1),YP(5),IP,KP,BETAP,DD)
CALL POLATE(R,Z, AL,VR(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)=RHOA(1)*(TEMPA(4)/TIP)**(1.0/(GAMMA-1.0))
VISTAR(4)=VISTAR(1)*(TEMPA(4)/TREF)**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)) 1,2,3
2,VR(2)=(VR(4)+VR(1))/2.0
3,VU(2)=(VU(4)+VU(1))/2.0
1,VZ(2)=(VZ(4)+VZ(2))/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(2))/2.0
VISTAR(2)=(VISTAR(4)+VISTAR(1))/2.0
T=TS
DO 710 1=1,N
710 YR(1)=YRS(1)
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*VP(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)
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)
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(TIME.GT.0) GO TO 625
WRITE(6,4050) T, (YR(I), I=1,6), RENOLD
WRITE(NSTEP-1,1)
GO TO 625
800 CONTINUE
WRITE(6,4060) T, (YR(I), I=1,6), RENOLD
WRITE(6,4070) M
805 CONTINUE
GO TO 105

FORMAT STATEMENTS

1000 FORMAT(215,6F10.5)
1005 FORMAT(8F10.5)
1010 FORMAT(8F10.5)
1015 FORMAT(8F10.5)
2000 FORMAT(1H ,8G16.7)
2010 FORMAT(9H R ARRAY)
2020 FORMAT(9H ORC ARRAY)
2030 FORMAT(12HOALPHA ARRAY)
2040 FORMAT(8HOZ ARRAY)
2050 FORMAT(7HOLAMDA)
2060 FORMAT(1H1,25H THE ORTHOgonal NUMBER IS,13///)
2070 FORMAT(46HOA SOLUTION CANNOT BE OBTAINED AT THIS STATION)
2080 FORMAT(1HO,63H AFTER 10, V(I,1) .GT. 2. * CP*CIP, E(I,1) =,F15.2)
2081 FORMAT(1HO,49H N0 20 LOOP, V(I,K+1) .GT. 2. * CP*TIP, V(I,K+1) =, 1 F15.2)
2090 FORMAT(11HORETA ARRAY/(1X,8G16.7))
2100 FORMAT(8HOV ARRAY/(1X,8G16.7))
2110 FORMAT(42HOTHE PASSAGE IS CHOKED WITH A MASS FLOW OF,8G16.7)
2120 FORMAT(7HOWTFLUES/(1X,8G16.7))
2130 FORMAT(1H1,11H INPUT DATA,/,8X,2HMX,7X,3HKMX, 8X,2HZL,13X,24Z2,
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 CONTIN(XFST, 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. 0.0) GO TO (90, 95), JZ
75 IND = 6
CALL PARCC(X, Y, APA, RPR, CPC)
DISCR = BPB**2 - APA*(CPC - YGIV)
IF (DISCR .LT. 0.0) GO TO 110
IF (DABS(400.0*APA*(CPC - YGIV)) .LT. BPB**2) GO TO 78
XEST = -RPR*SIGN(DSQR(DISCR), APA)
IF (JZ .EQ. 2 .AND. APA .LT. 0.0) XEST = -RPR*DSQR(DISCR)
XEST = XEST/2./APA
GO TO 79
78 ACR2 = APA/BPB*(CPC - YGIV)/BPB
IF (DABS(ACB2) .LT. 1.0E-8) ACB2 = 0.0
XEST = -(CPC - YGIV)/BPB*(1.0 + ACB2**2) .* ACR2**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)
**LEVEL 21**

```fortran
10 IF(EXEST.GT.X(3)) GO TO 50
   IF(EXEST.LT.X(1)) GO TO 60
   Y(2)=YCALC
   X(2)=XEST
   GO TO 70
C******THIRD OR LATER CALL / SOLUTION EXISTS,
C******BUT 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 - APPFARS TO BE CHOKEK
110 XEST=-BFB/2./APA
   IND=7
   IF(X(1).LE.XEST.AND.XEST.LE.X(3)) RETURN
   GO TO 90
130 IF(YCALC.GE.YGIV) GO TO 80
   IND=10
   RETURN
C******NO SOLUTION FOUND IN 50 ITERATIONS
160 IND=11
   RETURN
END
```

**SURROGATE**

```fortran
SURROGATE 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.EQ.4) ANS(I)=ANS(I-2)+DX*(F(I-2)+4.0*F(I-1)+F(I))/3.
10 CONTINUE
   RETURN
END
```
SUBROUTINE RJCAT(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(I, 1) = A*Z(I, 1)
RTEST = A*YR(5) + B
IF (RTEST .LE. YR(1)) GO TO 30
GO TO 20
10 IF (Z(1, 1) .GE. YR(5)) GO TO 30
20 CONTINUE
30 IP = 1
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 RCH(YR,R,Z,T,ETA,NOUT)
IMPLICIT REAL*8 (A-H,O-Z)
DIMENSION YR(5),R(7),Z(2),ETA(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
SN=(R(2)-R(1)) / (Z(2)-Z(1))
IF(ABS(YR(6)) .LE. 1.0E-12) GO TO 10
IF(ABS(YR(2)) .LE. 1.0E-12) GO TO 20
PM=YR(2)/YR(6)
ZB=(Z(1)*SM-YR(5)*PH+YR(1)-R(1))/(SM-PM)
R3=SN*(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(ABS(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(ABS(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),
G(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

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

B(N)=1.0

C(N)=-.5

F(N)=0.0

W(N)=0.0

SB(I)=C(I)/W(I)

G(I)=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

DO 30 I=1,MAX

K=2

IF(Z(I)-X(I)) 60,50,70

50 YINT(I)=Y(I)

GO TO 90

60 IF(Z(I).LT.(1.1*X(I)-.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(I)-.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.NE.70,70,65

85 YINT(I) = EM(K-1)*(X(K)-Z(I))**3/6*SB(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)-EM(K-1)

2*S(K)/6.)*X(K)-Z(I))

90 CONTINUE

MXA = 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, $\beta$, 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.
### EX: Example Case for NASA Report

<table>
<thead>
<tr>
<th>Card Group</th>
<th>Group A</th>
<th>Group B</th>
<th>Group C</th>
<th>Group D</th>
<th>Group E</th>
<th>Group F</th>
</tr>
</thead>
<tbody>
<tr>
<td>10.000</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>17.3</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>492.0</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>198.6</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>1716.48</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>215.15</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>215.28</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>215.45</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>215.60</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>0.0001</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>0.001</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
</tbody>
</table>

**Additional Note:**

These sets are stacked one after the other.

Card Group B here.
```fortran
SUBROUTINE POLOATE(R,Z,A,RP,ZP,IP,KP,AP,OO)
IMPLICIT REAL*8 (A-H,O-Z)
DIMENSION RI21,21..Z1Z1,211,AI21.21"ODI2.
DO 10 I=1,2
IA=IP+1-1
IF(DABSZ(IA,KP-1))..LT.1.0E-12) GO TO 5
IF(DABSZ(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)=DSORT((RA-RP)**2+(ZA-ZP)**2)
DT=D(I)*D(2)
AA=(D(I)*A(IP,KP-1)+D(2)*A(IP-1,KP-1))/DT
AB=(D(I)*A(IP,KP)+D(2)*A(IP-1,KP))/DT
DO 20 K=1,2
KA=KP-K+1
RC=(D(I)*R(IP,KA)+D(2)*R(IP-1,KA))/DT
ZC=(D(I)*Z(IP,KA)+D(2)*Z(IP-1,KA))/DT
20 CD(K)=DSORT((RP-RC)**2+(ZP-ZC)**2)
DT=DD(I)+DD(2)
AP=(DD(I)*AA+DD(2)*AB)/DT
RETURN
END
```

```fortran
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=(128.5-24.0*ARE+9.0682*ARE**2-1.7713*ARE**3+0.1718*ARE**4
1 -0.0065*ARE**5)*DGFC
RETURN
END
```
<table>
<thead>
<tr>
<th>Input Data</th>
<th>( m )</th>
<th>( \kappa )</th>
<th>( t )</th>
<th>( z_1 )</th>
<th>( z_2 )</th>
<th>( A_1 )</th>
<th>( A_2 )</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1.4000</td>
<td>912.400</td>
<td>516.700</td>
<td>0.226000-02</td>
<td>0.156000-01</td>
<td></td>
<td></td>
</tr>
<tr>
<td>( R_k ) ( n = 1 )</td>
<td>7</td>
<td>-0.184600-01</td>
<td>-0.211670-01</td>
<td>-0.237500-01</td>
<td>-0.347500-01</td>
<td>-0.392500-01</td>
<td>-0.433500-01</td>
</tr>
<tr>
<td>( R_k ) ( n = 1 )</td>
<td>10</td>
<td>-0.4000</td>
<td>-0.5000</td>
<td>-0.6000</td>
<td>-0.7000</td>
<td>-0.8000</td>
<td>-0.9000</td>
</tr>
<tr>
<td>( R_k ) ( n = 1 )</td>
<td>7</td>
<td>214.60</td>
<td>214.77</td>
<td>215.11</td>
<td>215.28</td>
<td>215.45</td>
<td>215.60</td>
</tr>
</tbody>
</table>
The mathematical content of the image is a table, which appears to contain data related to some kind of numerical analysis or scientific calculation. The table contains various columns, each with a header and several rows of data. The exact nature and purpose of the table are not clear from the image alone.

To provide a meaningful transcription, I would need to know the specific context or field this data pertains to. However, I can present the table as it appears in the image:

```
<table>
<thead>
<tr>
<th>Array</th>
<th>Data</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 AARRAY</td>
<td>0.3174710</td>
</tr>
<tr>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>BC ARRAY</td>
<td>10000.00</td>
</tr>
<tr>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>V ARRAY</td>
<td>0.7001470</td>
</tr>
<tr>
<td>0.0</td>
<td>0.0</td>
</tr>
</tbody>
</table>

The mathematical number is 1.
```

This table appears to list various arrays and their corresponding data, possibly in a scientific context. Further analysis or context would be required to interpret these numbers accurately.
The number of boxes is 2.

| X Array | 0.707573 | 0.315459 | 0.320594 | 0.3247217 | 0.328537 |
| Y Array | 0.726720 | 0.321000 | 0.331520 | 0.349320 | 0.361000 |
| Z Array | 0.713100 | 0.325000 | 0.334900 | 0.342500 | 0.350000 |
| A Array | 0.710100 | 0.320000 | 0.330000 | 0.337500 | 0.345000 |
| B Array | 0.707573 | 0.320594 | 0.3247217 | 0.328537 | 0.332100 |
| C Array | 0.726720 | 0.331520 | 0.349320 | 0.361000 | 0.372500 |
| D Array | 0.713100 | 0.342500 | 0.350000 | 0.357500 | 0.365000 |
| E Array | 0.710100 | 0.350000 | 0.357500 | 0.365000 | 0.372500 |
| F Array | 0.707573 | 0.361000 | 0.372500 | 0.381000 | 0.389537 |
| G Array | 0.726720 | 0.381000 | 0.390537 | 0.399000 | 0.407500 |
| H Array | 0.713100 | 0.407500 | 0.416000 | 0.424500 | 0.432500 |
| I Array | 0.710100 | 0.432500 | 0.441000 | 0.449500 | 0.457500 |
| J Array | 0.707573 | 0.457500 | 0.466000 | 0.474500 | 0.482500 |
| K Array | 0.726720 | 0.482500 | 0.491000 | 0.500000 | 0.508537 |
| L Array | 0.713100 | 0.508537 | 0.517000 | 0.525500 | 0.534000 |
| M Array | 0.710100 | 0.534000 | 0.542500 | 0.551000 | 0.559537 |
| N Array | 0.707573 | 0.559537 | 0.568000 | 0.576500 | 0.585037 |
| O Array | 0.726720 | 0.585037 | 0.593500 | 0.602000 | 0.610537 |
| P Array | 0.713100 | 0.610537 | 0.619000 | 0.627500 | 0.636037 |
| Q Array | 0.710100 | 0.636037 | 0.644500 | 0.653000 | 0.661537 |
| R Array | 0.707573 | 0.661537 | 0.670000 | 0.678500 | 0.687037 |
| S Array | 0.726720 | 0.687037 | 0.695500 | 0.704000 | 0.712537 |
| T Array | 0.713100 | 0.712537 | 0.721000 | 0.729500 | 0.738037 |
| U Array | 0.710100 | 0.738037 | 0.746500 | 0.755000 | 0.763537 |
| V Array | 0.707573 | 0.763537 | 0.772000 | 0.780500 | 0.789037 |

The number of rows is 11.

The number of columns is 2.
<table>
<thead>
<tr>
<th>SPRAY</th>
<th>0.4543437</th>
<th>0.3635971</th>
<th>0.7092690</th>
<th>0.3103446</th>
<th>0.3171673</th>
<th>0.3202594</th>
</tr>
</thead>
<tbody>
<tr>
<td>2 SPRAY</td>
<td>0.15441000-01</td>
<td>0.16395949-01</td>
<td>0.15412500-01</td>
<td>0.22301890-01</td>
<td>0.25252220-01</td>
<td>0.26148550-01</td>
</tr>
<tr>
<td>WC ARRAY</td>
<td>-0.19489000-01</td>
<td>-0.71676330-01</td>
<td>-0.30260600-01</td>
<td>-0.37390600-01</td>
<td>-0.39280000-01</td>
<td>-0.43333600-01</td>
</tr>
<tr>
<td>REFRA ARRAY</td>
<td>-46.490000</td>
<td>-46.500000</td>
<td>-46.900000</td>
<td>-46.500000</td>
<td>-46.500000</td>
<td>-46.500000</td>
</tr>
<tr>
<td>WTFS</td>
<td>0.29494477-01</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SEA PARRAY</td>
<td>71.980456</td>
<td>74.427346</td>
<td>76.633366</td>
<td>78.312760</td>
<td>79.469065</td>
<td>81.102630</td>
</tr>
<tr>
<td>T PARRAY</td>
<td>0.72190700</td>
<td>0.71275090</td>
<td>0.69870560</td>
<td>0.64711874</td>
<td>0.67222670</td>
<td>0.64833040</td>
</tr>
</tbody>
</table>

**The Corrected Page IS 4**
The central number is 7

<table>
<thead>
<tr>
<th>U ARRAY</th>
<th>0.24449469</th>
<th>0.26484426</th>
<th>0.300497</th>
<th>0.3028948</th>
<th>0.3065968</th>
<th>0.3066297</th>
</tr>
</thead>
<tbody>
<tr>
<td>7 ARRAY</td>
<td>0.24409420-01</td>
<td>0.24349742-01</td>
<td>0.29614280-01</td>
<td>0.23352880-01</td>
<td>0.27689070-01</td>
<td>0.41347420-01</td>
</tr>
<tr>
<td>H ARRAY</td>
<td>0.24664600-01</td>
<td>0.27116720-01</td>
<td>0.24733560-01</td>
<td>0.24750000-01</td>
<td>0.39250000-01</td>
<td>0.43353000-01</td>
</tr>
<tr>
<td>IHPA ARRAY</td>
<td>-0.69300000</td>
<td>-0.63700000</td>
<td>-0.63700000</td>
<td>-0.63700000</td>
<td>-0.63700000</td>
<td>-0.63700000</td>
</tr>
</tbody>
</table>

The central number is 8

<table>
<thead>
<tr>
<th>U ARRAY</th>
<th>0.7495457</th>
<th>0.7595313</th>
<th>0.7219958</th>
<th>0.7120555</th>
<th>0.7045307</th>
<th>0.6940300</th>
</tr>
</thead>
<tbody>
<tr>
<td>7 ARRAY</td>
<td>0.7491149</td>
<td>0.24441861</td>
<td>0.2453293</td>
<td>0.2464490</td>
<td>0.2465857</td>
<td>0.2467153</td>
</tr>
<tr>
<td>H ARRAY</td>
<td>0.73908140-01</td>
<td>0.73859790-01</td>
<td>0.738192-01</td>
<td>0.7386510-01</td>
<td>0.73912570-01</td>
<td>0.73958260-01</td>
</tr>
<tr>
<td>H ARRAY</td>
<td>0.74664600-01</td>
<td>0.74613660-01</td>
<td>0.74565000-01</td>
<td>0.74517500-01</td>
<td>0.74570000-01</td>
<td>0.74590000-01</td>
</tr>
<tr>
<td>IHPA ARRAY</td>
<td>-75.00000</td>
<td>-75.00000</td>
<td>-75.00000</td>
<td>-75.00000</td>
<td>-75.00000</td>
<td>-75.00000</td>
</tr>
</tbody>
</table>

The central number is 9

<p>| U ARRAY | 0.7864519 | 0.7973131 | 0.7550864 | 0.7275774 | 0.7216794 | 0.7149810 |</p>
<table>
<thead>
<tr>
<th>ARRAY</th>
<th>0.2866447</th>
<th>0.288671</th>
<th>0.289133</th>
<th>0.289372</th>
<th>0.289627</th>
<th>0.289845</th>
</tr>
</thead>
<tbody>
<tr>
<td>F ARRAY</td>
<td>0.4162620-01</td>
<td>0.4245200-01</td>
<td>0.4270570-01</td>
<td>0.42799150-01</td>
<td>0.43142560-01</td>
<td>0.45765590-01</td>
</tr>
<tr>
<td>FC ARRAY</td>
<td>0.28699000-01</td>
<td>0.29117000-01</td>
<td>0.30250000-01</td>
<td>0.34750000-01</td>
<td>0.39250000-01</td>
<td>0.43333000-01</td>
</tr>
<tr>
<td>AREA</td>
<td>0.2860000</td>
<td>0.2860000</td>
<td>0.2860000</td>
<td>0.2860000</td>
<td>0.2860000</td>
<td>0.2860000</td>
</tr>
<tr>
<td>F ARRAY</td>
<td>0.7146200</td>
<td>0.7146200</td>
<td>0.7151100</td>
<td>0.7151100</td>
<td>0.7154500</td>
<td>0.7154500</td>
</tr>
<tr>
<td>40015</td>
<td>0.7344456</td>
<td>0.7356484</td>
<td>0.7351220</td>
<td>0.7456455</td>
<td>0.7417505</td>
<td>0.7388870</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>ARRAY</th>
<th>0.2840000</th>
<th>0.2850000</th>
<th>0.2850000</th>
<th>0.2850000</th>
<th>0.2850000</th>
<th>0.2850000</th>
</tr>
</thead>
<tbody>
<tr>
<td>F ARRAY</td>
<td>0.4184600-01</td>
<td>0.42837000-01</td>
<td>0.42750000-01</td>
<td>0.43750000-01</td>
<td>0.44750000-01</td>
<td>0.45873000-01</td>
</tr>
<tr>
<td>FC ARRAY</td>
<td>0.10000.00</td>
<td>0.10000.00</td>
<td>0.10000.00</td>
<td>0.10000.00</td>
<td>0.10000.00</td>
<td>0.10000.00</td>
</tr>
<tr>
<td>AREA</td>
<td>0.2840000</td>
<td>0.2840000</td>
<td>0.2840000</td>
<td>0.2840000</td>
<td>0.2840000</td>
<td>0.2840000</td>
</tr>
<tr>
<td>F ARRAY</td>
<td>0.7146200</td>
<td>0.7146200</td>
<td>0.7151100</td>
<td>0.7151100</td>
<td>0.7154500</td>
<td>0.7154500</td>
</tr>
<tr>
<td>40015</td>
<td>0.7344456</td>
<td>0.7356484</td>
<td>0.7351220</td>
<td>0.7456455</td>
<td>0.7417505</td>
<td>0.7388870</td>
</tr>
<tr>
<td>STRING</td>
<td>RPM</td>
<td>TSPF</td>
<td>TSLT</td>
<td>TAIL</td>
<td>CGFC</td>
<td></td>
</tr>
<tr>
<td>--------</td>
<td>-----</td>
<td>------</td>
<td>------</td>
<td>------</td>
<td>------</td>
<td></td>
</tr>
<tr>
<td>ETA-N</td>
<td>ETA-T</td>
<td>ETA-N</td>
<td>ETA-T</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>VAR1</td>
<td>VAR2</td>
<td>VAR3</td>
<td>VAR4</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**SOLUTION**

<table>
<thead>
<tr>
<th>STRING</th>
<th>RPM</th>
<th>TSPF</th>
<th>TSLT</th>
<th>TAIL</th>
<th>CGFC</th>
</tr>
</thead>
<tbody>
<tr>
<td>ETA-N</td>
<td>ETA-T</td>
<td>ETA-N</td>
<td>ETA-T</td>
<td></td>
<td></td>
</tr>
<tr>
<td>VAR1</td>
<td>VAR2</td>
<td>VAR3</td>
<td>VAR4</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**SOLUTION**

<table>
<thead>
<tr>
<th>STRING</th>
<th>RPM</th>
<th>TSPF</th>
<th>TSLT</th>
<th>TAIL</th>
<th>CGFC</th>
</tr>
</thead>
<tbody>
<tr>
<td>ETA-N</td>
<td>ETA-T</td>
<td>ETA-N</td>
<td>ETA-T</td>
<td></td>
<td></td>
</tr>
<tr>
<td>VAR1</td>
<td>VAR2</td>
<td>VAR3</td>
<td>VAR4</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**SOLUTION**

<table>
<thead>
<tr>
<th>STRING</th>
<th>RPM</th>
<th>TSPF</th>
<th>TSLT</th>
<th>TAIL</th>
<th>CGFC</th>
</tr>
</thead>
<tbody>
<tr>
<td>ETA-N</td>
<td>ETA-T</td>
<td>ETA-N</td>
<td>ETA-T</td>
<td></td>
<td></td>
</tr>
<tr>
<td>VAR1</td>
<td>VAR2</td>
<td>VAR3</td>
<td>VAR4</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**SOLUTION**

<table>
<thead>
<tr>
<th>STRING</th>
<th>RPM</th>
<th>TSPF</th>
<th>TSLT</th>
<th>TAIL</th>
<th>CGFC</th>
</tr>
</thead>
<tbody>
<tr>
<td>ETA-N</td>
<td>ETA-T</td>
<td>ETA-N</td>
<td>ETA-T</td>
<td></td>
<td></td>
</tr>
<tr>
<td>VAR1</td>
<td>VAR2</td>
<td>VAR3</td>
<td>VAR4</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**SOLUTION**

<table>
<thead>
<tr>
<th>STRING</th>
<th>RPM</th>
<th>TSPF</th>
<th>TSLT</th>
<th>TAIL</th>
<th>CGFC</th>
</tr>
</thead>
<tbody>
<tr>
<td>ETA-N</td>
<td>ETA-T</td>
<td>ETA-N</td>
<td>ETA-T</td>
<td></td>
<td></td>
</tr>
<tr>
<td>VAR1</td>
<td>VAR2</td>
<td>VAR3</td>
<td>VAR4</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**SOLUTION**

<table>
<thead>
<tr>
<th>STRING</th>
<th>RPM</th>
<th>TSPF</th>
<th>TSLT</th>
<th>TAIL</th>
<th>CGFC</th>
</tr>
</thead>
<tbody>
<tr>
<td>ETA-N</td>
<td>ETA-T</td>
<td>ETA-N</td>
<td>ETA-T</td>
<td></td>
<td></td>
</tr>
<tr>
<td>VAR1</td>
<td>VAR2</td>
<td>VAR3</td>
<td>VAR4</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**SOLUTION**

<table>
<thead>
<tr>
<th>STRING</th>
<th>RPM</th>
<th>TSPF</th>
<th>TSLT</th>
<th>TAIL</th>
<th>CGFC</th>
</tr>
</thead>
<tbody>
<tr>
<td>ETA-N</td>
<td>ETA-T</td>
<td>ETA-N</td>
<td>ETA-T</td>
<td></td>
<td></td>
</tr>
<tr>
<td>VAR1</td>
<td>VAR2</td>
<td>VAR3</td>
<td>VAR4</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**SOLUTION**

<table>
<thead>
<tr>
<th>STRING</th>
<th>RPM</th>
<th>TSPF</th>
<th>TSLT</th>
<th>TAIL</th>
<th>CGFC</th>
</tr>
</thead>
<tbody>
<tr>
<td>ETA-N</td>
<td>ETA-T</td>
<td>ETA-N</td>
<td>ETA-T</td>
<td></td>
<td></td>
</tr>
<tr>
<td>VAR1</td>
<td>VAR2</td>
<td>VAR3</td>
<td>VAR4</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**SOLUTION**

<table>
<thead>
<tr>
<th>STRING</th>
<th>RPM</th>
<th>TSPF</th>
<th>TSLT</th>
<th>TAIL</th>
<th>CGFC</th>
</tr>
</thead>
<tbody>
<tr>
<td>ETA-N</td>
<td>ETA-T</td>
<td>ETA-N</td>
<td>ETA-T</td>
<td></td>
<td></td>
</tr>
<tr>
<td>VAR1</td>
<td>VAR2</td>
<td>VAR3</td>
<td>VAR4</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**SOLUTION**

<table>
<thead>
<tr>
<th>STRING</th>
<th>RPM</th>
<th>TSPF</th>
<th>TSLT</th>
<th>TAIL</th>
<th>CGFC</th>
</tr>
</thead>
<tbody>
<tr>
<td>ETA-N</td>
<td>ETA-T</td>
<td>ETA-N</td>
<td>ETA-T</td>
<td></td>
<td></td>
</tr>
<tr>
<td>VAR1</td>
<td>VAR2</td>
<td>VAR3</td>
<td>VAR4</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**SOLUTION**

<table>
<thead>
<tr>
<th>STRING</th>
<th>RPM</th>
<th>TSPF</th>
<th>TSLT</th>
<th>TAIL</th>
<th>CGFC</th>
</tr>
</thead>
<tbody>
<tr>
<td>ETA-N</td>
<td>ETA-T</td>
<td>ETA-N</td>
<td>ETA-T</td>
<td></td>
<td></td>
</tr>
<tr>
<td>VAR1</td>
<td>VAR2</td>
<td>VAR3</td>
<td>VAR4</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**SOLUTION**

<table>
<thead>
<tr>
<th>STRING</th>
<th>RPM</th>
<th>TSPF</th>
<th>TSLT</th>
<th>TAIL</th>
<th>CGFC</th>
</tr>
</thead>
<tbody>
<tr>
<td>ETA-N</td>
<td>ETA-T</td>
<td>ETA-N</td>
<td>ETA-T</td>
<td></td>
<td></td>
</tr>
<tr>
<td>VAR1</td>
<td>VAR2</td>
<td>VAR3</td>
<td>VAR4</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**SOLUTION**

<table>
<thead>
<tr>
<th>STRING</th>
<th>RPM</th>
<th>TSPF</th>
<th>TSLT</th>
<th>TAIL</th>
<th>CGFC</th>
</tr>
</thead>
<tbody>
<tr>
<td>ETA-N</td>
<td>ETA-T</td>
<td>ETA-N</td>
<td>ETA-T</td>
<td></td>
<td></td>
</tr>
<tr>
<td>VAR1</td>
<td>VAR2</td>
<td>VAR3</td>
<td>VAR4</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**SOLUTION**

<table>
<thead>
<tr>
<th>STRING</th>
<th>RPM</th>
<th>TSPF</th>
<th>TSLT</th>
<th>TAIL</th>
<th>CGFC</th>
</tr>
</thead>
<tbody>
<tr>
<td>ETA-N</td>
<td>ETA-T</td>
<td>ETA-N</td>
<td>ETA-T</td>
<td></td>
<td></td>
</tr>
<tr>
<td>VAR1</td>
<td>VAR2</td>
<td>VAR3</td>
<td>VAR4</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**SOLUTION**

<table>
<thead>
<tr>
<th>STRING</th>
<th>RPM</th>
<th>TSPF</th>
<th>TSLT</th>
<th>TAIL</th>
<th>CGFC</th>
</tr>
</thead>
<tbody>
<tr>
<td>ETA-N</td>
<td>ETA-T</td>
<td>ETA-N</td>
<td>ETA-T</td>
<td></td>
<td></td>
</tr>
<tr>
<td>VAR1</td>
<td>VAR2</td>
<td>VAR3</td>
<td>VAR4</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**SOLUTION**

<table>
<thead>
<tr>
<th>STRING</th>
<th>RPM</th>
<th>TSPF</th>
<th>TSLT</th>
<th>TAIL</th>
<th>CGFC</th>
</tr>
</thead>
<tbody>
<tr>
<td>ETA-N</td>
<td>ETA-T</td>
<td>ETA-N</td>
<td>ETA-T</td>
<td></td>
<td></td>
</tr>
<tr>
<td>VAR1</td>
<td>VAR2</td>
<td>VAR3</td>
<td>VAR4</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**SOLUTION**

<table>
<thead>
<tr>
<th>STRING</th>
<th>RPM</th>
<th>TSPF</th>
<th>TSLT</th>
<th>TAIL</th>
<th>CGFC</th>
</tr>
</thead>
<tbody>
<tr>
<td>ETA-N</td>
<td>ETA-T</td>
<td>ETA-N</td>
<td>ETA-T</td>
<td></td>
<td></td>
</tr>
<tr>
<td>VAR1</td>
<td>VAR2</td>
<td>VAR3</td>
<td>VAR4</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**SOLUTION**

<table>
<thead>
<tr>
<th>STRING</th>
<th>RPM</th>
<th>TSPF</th>
<th>TSLT</th>
<th>TAIL</th>
<th>CGFC</th>
</tr>
</thead>
<tbody>
<tr>
<td>ETA-N</td>
<td>ETA-T</td>
<td>ETA-N</td>
<td>ETA-T</td>
<td></td>
<td></td>
</tr>
<tr>
<td>VAR1</td>
<td>VAR2</td>
<td>VAR3</td>
<td>VAR4</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**SOLUTION**

<table>
<thead>
<tr>
<th>STRING</th>
<th>RPM</th>
<th>TSPF</th>
<th>TSLT</th>
<th>TAIL</th>
<th>CGFC</th>
</tr>
</thead>
<tbody>
<tr>
<td>ETA-N</td>
<td>ETA-T</td>
<td>ETA-N</td>
<td>ETA-T</td>
<td></td>
<td></td>
</tr>
<tr>
<td>VAR1</td>
<td>VAR2</td>
<td>VAR3</td>
<td>VAR4</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**SOLUTION**

<table>
<thead>
<tr>
<th>STRING</th>
<th>RPM</th>
<th>TSPF</th>
<th>TSLT</th>
<th>TAIL</th>
<th>CGFC</th>
</tr>
</thead>
<tbody>
<tr>
<td>ETA-N</td>
<td>ETA-T</td>
<td>ETA-N</td>
<td>ETA-T</td>
<td></td>
<td></td>
</tr>
<tr>
<td>VAR1</td>
<td>VAR2</td>
<td>VAR3</td>
<td>VAR4</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**SOLUTION**

<table>
<thead>
<tr>
<th>STRING</th>
<th>RPM</th>
<th>TSPF</th>
<th>TSLT</th>
<th>TAIL</th>
<th>CGFC</th>
</tr>
</thead>
<tbody>
<tr>
<td>ETA-N</td>
<td>ETA-T</td>
<td>ETA-N</td>
<td>ETA-T</td>
<td></td>
<td></td>
</tr>
<tr>
<td>VAR1</td>
<td>VAR2</td>
<td>VAR3</td>
<td>VAR4</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**SOLUTION**

<table>
<thead>
<tr>
<th>STRING</th>
<th>RPM</th>
<th>TSPF</th>
<th>TSLT</th>
<th>TAIL</th>
<th>CGFC</th>
</tr>
</thead>
<tbody>
<tr>
<td>ETA-N</td>
<td>ETA-T</td>
<td>ETA-N</td>
<td>ETA-T</td>
<td></td>
<td></td>
</tr>
<tr>
<td>VAR1</td>
<td>VAR2</td>
<td>VAR3</td>
<td>VAR4</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**SOLUTION**

<table>
<thead>
<tr>
<th>STRING</th>
<th>RPM</th>
<th>TSPF</th>
<th>TSLT</th>
<th>TAIL</th>
<th>CGFC</th>
</tr>
</thead>
<tbody>
<tr>
<td>ETA-N</td>
<td>ETA-T</td>
<td>ETA-N</td>
<td>ETA-T</td>
<td></td>
<td></td>
</tr>
<tr>
<td>VAR1</td>
<td>VAR2</td>
<td>VAR3</td>
<td>VAR4</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**SOLUTION**

<table>
<thead>
<tr>
<th>STRING</th>
<th>RPM</th>
<th>TSPF</th>
<th>TSLT</th>
<th>TAIL</th>
<th>CGFC</th>
</tr>
</thead>
<tbody>
<tr>
<td>ETA-N</td>
<td>ETA-T</td>
<td>ETA-N</td>
<td>ETA-T</td>
<td></td>
<td></td>
</tr>
<tr>
<td>VAR1</td>
<td>VAR2</td>
<td>VAR3</td>
<td>VAR4</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**SOLUTION**

<table>
<thead>
<tr>
<th>STRING</th>
<th>RPM</th>
<th>TSPF</th>
<th>TSLT</th>
<th>TAIL</th>
<th>CGFC</th>
</tr>
</thead>
<tbody>
<tr>
<td>ETA-N</td>
<td>ETA-T</td>
<td>ETA-N</td>
<td>ETA-T</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
| VAR1   | VAR2  | VAR3 ...
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


Table 1. Coefficients of Polynomial, $a_i$, Describing Variation of Tangential Restitution Coefficient with Incidence Angle*

<table>
<thead>
<tr>
<th>Coefficient of Number</th>
<th>Value of Coefficient</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1.819024</td>
</tr>
<tr>
<td>2</td>
<td>5.117122</td>
</tr>
<tr>
<td>3</td>
<td>-49.252858</td>
</tr>
<tr>
<td>4</td>
<td>131.035284</td>
</tr>
<tr>
<td>5</td>
<td>-174.424912</td>
</tr>
<tr>
<td>6</td>
<td>117.872348</td>
</tr>
<tr>
<td>7</td>
<td>-24.478854</td>
</tr>
<tr>
<td>8</td>
<td>-16.729740</td>
</tr>
<tr>
<td>9</td>
<td>11.230019</td>
</tr>
<tr>
<td>10</td>
<td>-1.979996</td>
</tr>
</tbody>
</table>

*Angles expressed in radians

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

<table>
<thead>
<tr>
<th>Coefficient of Number</th>
<th>Value of Coefficient</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>5.721469</td>
</tr>
<tr>
<td>2</td>
<td>-41.880784</td>
</tr>
<tr>
<td>3</td>
<td>178.168464</td>
</tr>
<tr>
<td>4</td>
<td>-424.388179</td>
</tr>
<tr>
<td>5</td>
<td>572.763055</td>
</tr>
<tr>
<td>6</td>
<td>-406.662526</td>
</tr>
<tr>
<td>7</td>
<td>87.142750</td>
</tr>
<tr>
<td>8</td>
<td>70.651124</td>
</tr>
<tr>
<td>9</td>
<td>-50.498187</td>
</tr>
<tr>
<td>10</td>
<td>9.676744</td>
</tr>
</tbody>
</table>

*Angles expressed in radians
FIGURE 1. SCHEMATIC OF TYPICAL RADIAL INFLOW TURBINE
FIGURE 2. ROUND OFF AND TRUNCATION ERRORS INHERENT IN THE NUMERICAL PROCEDURE
READ GAS FLOW DATA

READ PARTICLE DATA AND INITIAL LOCATION

DETERMINE PROPERTIES OF GAS AT PARTICLE LOCATION

COMPLETE INTEGRATION STEP

INTEGRATE ONE TIME STEP

RESET INTEGRATION STEP TO USE CORRECTED AVERAGE VALUES

DETERMINE PROPERTIES OF GAS AT PARTICLE NEW LOCATION

ARE CORRECT AVERAGE GAS PROPERTIES USED?

WRITE OUTPUT

IS SOLUTION COMPLETE FOR THIS PARTICLE?

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.

\[ \eta_n = 1.0 - \varphi_1(\theta) \varphi_2(v) \]

where \[ \varphi_1(\theta) = \sum_{n=1}^{10} a_n \theta^n \]
and \[ \varphi_2(76.2) = 1.0 \]

**FIGURE 9. INFLUENCE OF INCIDENCE ANGLE ON NORMAL RESTITUTION COEFFICIENT**
Material:
Surface - 2024 Aluminum
Particle - SiO₂

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

\[ \eta_n = 1.0 - \varphi_1(\beta) \varphi_2(\nu) \]
where \( \varphi_1(\beta) = \varphi_1(45°) \)
and \( \varphi_2(\nu) = 0.65 \left( 1.0 - e^{-\nu/24.5} \right) \)

**Figure 10. Influence of Incidence Velocity on Normal Restitution Coefficient**
Material:
Surface - 2024 Aluminum
Particle - SiO₂

Figure 11. Normal Restitution Coefficient

FIGURE 11. NORMAL RESTITUTION COEFFICIENT
Material:
Surface = 2024 Aluminum
Particle = SiO₂
Data given in Reference 6
for incidence velocity = 76.2 m/sec.

\[ n_t = 1.0 - \phi_1(\beta) \phi_2(v) \]
where \[ \phi_1 = \sum_{n=1}^{10} b_n \beta^n \]
and \[ \phi_2(76.1) = 1.0 \]

**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°.

\[ \eta_t = 1.0 - \phi_1(\beta) \phi_2(v) \]
where \[ \phi_1(\beta) = \phi_1(45°) \]
and \[ \phi_2(v) = \frac{v}{76.2} \]

**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 16. COORDINATE SYSTEM AND TYPICAL GAS VELOCITY COMPONENTS
FIGURE 17. PARDIM FLOW CHART
FIGURE 18. ORIENTATION COORDINATES
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
READ FLUID FLOW DATA
ECHO CHECK
CALCULATE GRID POINTS AND GAS FLOW AT THESE PTS.
WRITE INFO. ON THIS ORTHOGONAL
WRITE SOLUTION
REDO FOR NEXT ORTHOGONAL
CONVERT UNITS

READ PARTICLE DATA
ECHO CHECK
INITIALIZE
WRITE SIMILARITY PARAMETER
CHECK PARTICLE LOCATION OUT OF BOUNDS IN BOUNDS
INTERPOLATE FLUID PROPERTIES
SET UP FIRST STEP

INTEGRATE OVER ONE STEP
LOCATE PARTICLE IN BOUNDS
VALUE INCORECT
TEST GAS PROPERTIES VALUES CORRECT
COMPLETE INTEGRATION

CHECK IF OUTPUT SHOULD BE PRINTED THIS STEP
NO
YES
WRITE OUTPUT

NO CHECK IF PARTICLE BETWEEN INLET AND OUTLET
YES

FIGURE 24. VANPY FLOW DIAGRAM

172
FIGURE 25. COORDINATE SYSTEM USED IN VANPY COMPUTER PROGRAM