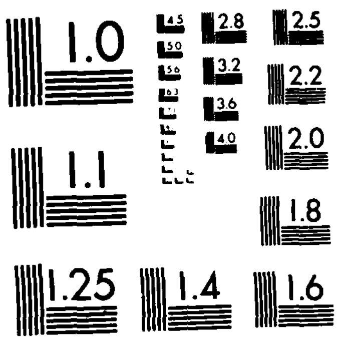
AD-A126 404 COMPUTER MODELING OF TIME-DEPENDENT RIME ICING IN THE ATMOSPHERE(U) COLD REGIONS RESEARCH AND ENGINEERING LAB HANOVER NH E P LOZOWSKI ET AL. JAN 83 CRREL-83-2 1/1

UNCLASSIFIED F/G 8/12 NL

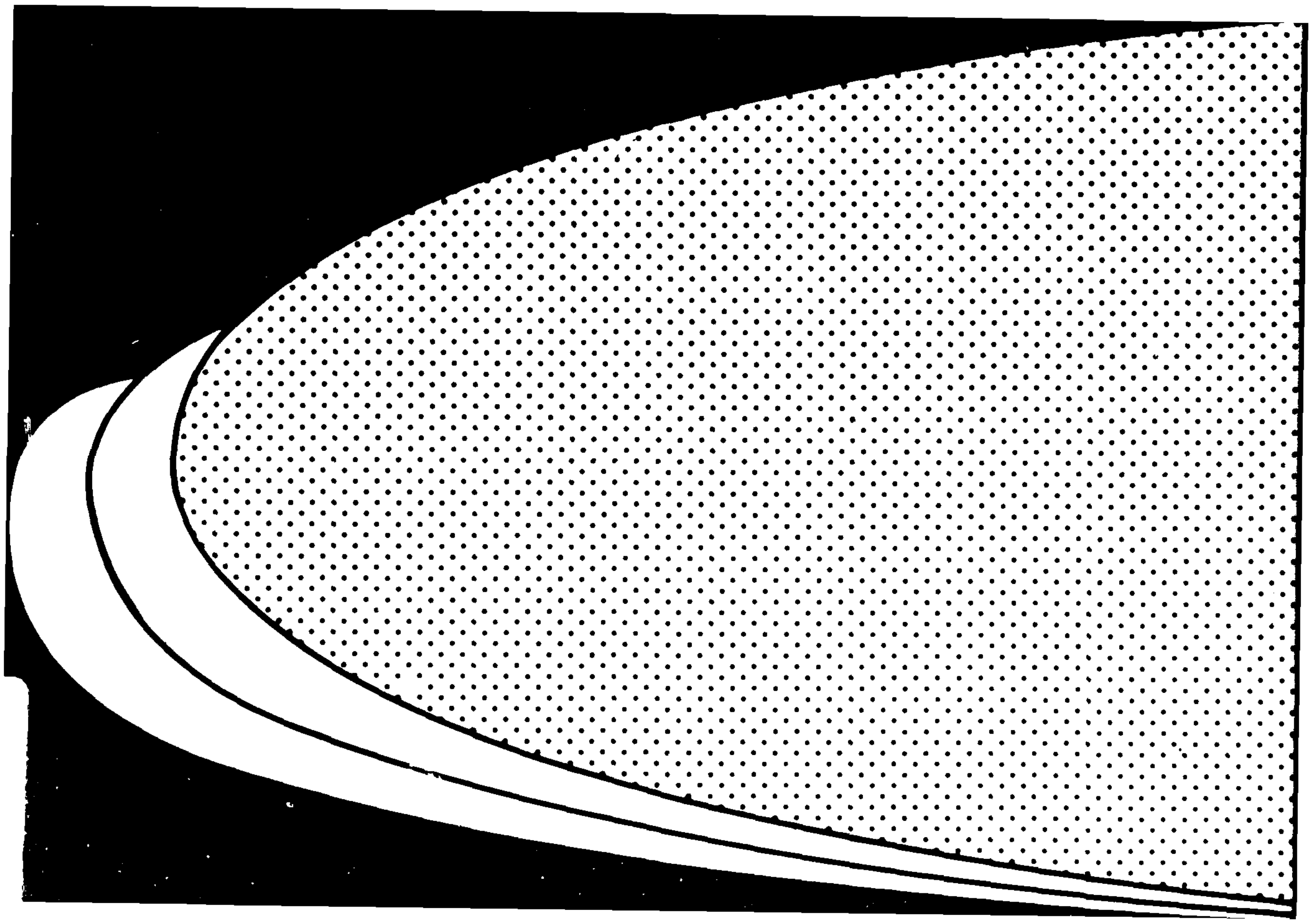
END

FILMED

DTIC



MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A



CRREL Report 83-2

January 1983

Computer modeling of time-dependent rime icing in the atmosphere

Edward P. Lozowski and Myron M. Oleskiw

Prepared for
HEADQUARTERS SPACE DIVISION
UNITED STATES AIR FORCE

Unclassified

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER CRREL Report 83-2	2. GOVT ACCESSION NO. AD-A126404	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) COMPUTER MODELING OF TIME-DEPENDENT RIME ICING IN THE ATMOSPHERE		5. TYPE OF REPORT & PERIOD COVERED 6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) Edward P. Lozowski and Myron M. Oleskiw		8. CONTRACT OR GRANT NUMBER(s)
9. PERFORMING ORGANIZATION NAME AND ADDRESS University of Alberta Edmonton, Alberta, Canada		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS DA Project 4A161102AT24, Task C/E1, Work Unit 002
11. CONTROLLING OFFICE NAME AND ADDRESS HQ Space Division, USAF Box 92960, Worldwide Postal Ctr. Los Angeles, California 90009		12. REPORT DATE January 1983 13. NUMBER OF PAGES 81
14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office) U.S. Army Cold Regions Research and Engineering Laboratory 72 Lyme Rd. Hanover, New Hampshire 03755		15. SECURITY CLASS. (of this report) Unclassified 15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Airfoils
Computerized simulation
Ice formation
Rime ice

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

A numerical model of rime ice accretion on an arbitrary two-dimensional airfoil is presented. The physics of the model are described and results are presented that demonstrate, by comparison with other theoretical data and experimental data, that the model predictions are believable. Results are also presented that illustrate the capability of the model to handle time-dependent rime ice accretion, taking into account the feedback between the ice accretion and the airflow and droplet trajectory fields.

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

Unclassified

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

PREFACE

This report was prepared by Dr. Edward P. Lozowski, Professor of Meteorology, and Myron M. Oleskiw, Ph.D. Candidate in Meteorology, of the University of Alberta in Edmonton, Alberta, Canada. The project was contracted under DA Project 4A161102AT24, *Adhesion and Physics of Ice,* Task C/E1, Work Unit 002.

The report was technically reviewed by Michael G. Ferrick and Jacqueline A. Richter of CRREL. Several individuals have contributed directly or indirectly to the success of the research. In particular, the authors wish to thank J. Stallabrass for his advice, assistance and inspiration; Professor D.J. Marsden for his assistance with the potential flow calculations; S.F. Ackley, who monitored the work and saw that it was brought to fruition; T. Cansdale and M. Bragg, whose work along parallel lines has stimulated the authors and helped them to gain confidence in the validity of the present model; L. Smith, who persevered in typing through several versions of the manuscript; and G. Lester and his staff, who capably prepared the illustrations.

CONTENTS

ILLUSTRATIONS

TABLES

NOMENCLATURE

C	cylinder diameter (m)
C_i	the control point on the ith line segment (control element), approximating the airfoil surface
C_D	drag coefficient (dimensionless)
E	total collision efficiency (%)
E_m	maximum local collision efficiency (%)
$\bar{g}$	acceleration due to gravity (m s^{-2})
K	Langmuir inertia parameter (dimensionless)
L	nondimensional distance along the surface of the accretion, starting at the nose (dimensionless)
N	number of line segments (control elements) approximating the airfoil surface
P	any point in the airstream
P	air pressure (Pa)
r	distance between a point on a control element and any point in the airstream (m)
$r(L)$	local radius of curvature of the accretion or substrate at distance L from the nose (m)
r_d	droplet radius (m)
$R(L)$	icing flux at distance L from the nose (kg m^{-2} s^{-1})
Re	Reynolds number of the droplet (dimensionless)
S_j	any point on the jth control element
T	air temperature (K)
t	time (s)
t_A	total accretion time for a layer (s)
u	x component of airspeed (m s^{-1})
v	y component of airspeed (m s^{-1})
v_x	x component of droplet impact speed (m s^{-1})
v_y	y component of droplet impact speed (m s^{-1})
V_∞	freestream airspeed (m s^{-1})
$\bar{V}_a$	vector air velocity (m s^{-1})
$\bar{V}_d$	vector droplet velocity (m s^{-1})
w	liquid water content of cloud (kg m^{-3})
x	x-coordinate (m)
X	nondimensional x-coordinate = x/C (dimensionless)
$\bar{X}_d$	nondimensional droplet position vector (dimensionless)
y	y-coordinate (m)
Y	nondimensional y-coordinate = y/C (dimensionless)

α	angle of attack of airfoil chord relative to freestream direction (radians)
β	local collision efficiency (%)
β_0	maximum local collision efficiency (%)
γ	vorticity density along a control element (s^{-1} m^{-1})
ϕ	nondimensional impingement parameter (dimensionless)
μ_a	dynamic viscosity of airstream (kg m^{-1} s^{-1})
υ_a	kinematic viscosity of airstream (m^2 s^{-1})
ϱ_a	density of airstream (kg m^{-3})
ϱ_d	density of a water droplet (kg m^{-3})
ϱ_i	density of accreted ice (kg m^{-3})
θ_m	maximum angle of impingement on cylinder (i.e. maximum accretion extent) (radians)
τ	time (s)
ψ	stream function (m^2 s^{-1})

COMPUTER MODELING OF TIME-DEPENDENT RIME ICING IN THE ATMOSPHERE

Edward P. Lozowski and Myron M. Oleskiw

INTRODUCTION

The literature on the subject of icing is very extensive, and we do not intend to review it here. Instead, we will mention simply that the present work arose chiefly as a result of two earlier investigations into icing, one at the U.S. Army Cold Regions Research and Engineering Laboratory and the other at the National Research Council of Canada in Ottawa. The first of these studies was reported by Ackley and Templeton (1979), while the second was described by Lozowski et al. (1979). Both were computer-simulated models of ice accretion on a cylinder. The first included time-dependent effects but ignored runback, and the second ignored time dependence but allowed for the thermodynamics of runback.

Although cylinder icing models are of intrinsic importance for understanding powerline icing, for example, their geometry is not appropriate for the study of airfoil icing. Airfoil icing has been a subject of renewed interest in recent years, in part because of a need to certify helicopters and general aviation aircraft for flight in IFR (instrument flight rules) icing conditions. The limited power available on such aircraft and the new materials used in airfoil construction demand that deicing or anti-icing equipment be carefully designed for maximum efficiency. Although the design of such equipment requires wind-tunnel and ultimately field testing, computer simulation models are considered to be an important tool in the design process (Rosen and Potash 1981).

The objective of the present work is therefore to develop and test a computer simulation model for airfoil icing. This report describes a model that permits simulation of the time-dependent growth of ice without runback on an arbitrary, two-dimensional airfoil. In developing the model, a great deal of effort has gone into carefully specifying the assumptions made and into testing the individual components of the model. We are confident that within the framework of assumptions of the model, the icing accretions that it predicts are believable. Because of the effort required to develop and test the present model, it has not been possible in the time available to make the model completely general. Consequently, we have not, for example, incorporated accretion thermodynamics into the model nor taken into account rotation effects, such as could be found on helicopter rotor blades. These are developments for the future. Nevertheless, the model as it stands should be very useful for estimating the icing rate and shape on airfoils when the accretion is dry (i.e. no runback) and when rotation effects (or any other three-dimensional effects) may be ignored. During the course of this work, two opportunties arose to make presentations of the progress to date to audiences of cloud physicists and aerodynamicists. These presentations are summarized in Oleskiw and Lozowski (1980) and Lozowski and Oleskiw (1981).

METHODOLOGY

The modeling of airfoil icing may be separated into two distinct aspects. The first is the impingement of supercooled water droplets on the airfoil surface, and the second is the mechanics and thermodynamics of the resulting accretion. The present study deals exclusively with the first aspect. This is sufficient for the investigation of the dry accretion process, in which the heat transfer is great enough that all of the impinging water droplets freeze at their point of impact. This restriction is analogous to that made by Ackley and Templeton (1979) in their model of icing cylinders. Cansdale and McNaughtan (1977) and Lozowski et al. (1979) also considered the case of wet accretion on cylinders, in which some impacting water remains unfrozen and is blown back along the icing surface. The extension of these wet accretion models to airfoils requires an ability to calculate the heat transfer of iced airfoils. We have not had the time or funds to do this, either theoretically or by experiment, under the present contract.

The computer algorithm for simulating the dry accretion process may be broken down into the following steps:

1. Determining the potential flow stream function field around an arbitrary two-dimensional airfoil in crossflow.
2. Determining the incompressible velocity field around the airfoil.
3. Calculating droplet trajectories and points of impact.
4. Determining the airfoil collision efficiency as a function of surface position for specified values of freestream airspeed, droplet size, and airfoil angle of attack.
5. Calculating the spatial distribution of icing during a short time interval, under the dry accretion assumption.
6. Determining the accretion shape and mass.
7. Calculating the new airfoil shape as modified by the ice accretion.
8. Repeating steps one to seven as often as desired to obtain the growth of the accretion as a function of time.

In the detailed descriptions that follow, we deal with each of these steps in turn.

Potential flow around an arbitrary airfoil

There are numerous potential flow codes available that permit the determination of the stream function field around an arbitrary two-dimensional airfoil. These can be broadly classified into two groups: a) conformal transformation techniques, e.g. Theodorson and Garrick (1932), and b) surface singularity methods, e.g. Hess and Smith (1967). The particular technique chosen to address the icing problem should have the following characteristics. First, it should be particularly efficient in terms of computer time, because of the large number of air velocity calculations required to determine droplet trajectories. Secondly, it should be capable of handling the changes to the airfoil profile due to the ice accretion. In this latter connection, the computer code must be capable of accepting a specification of the airfoil in terms of surface coordinates and, moreover, it should not be too sensitive to small errors in the specified airfoil coordinates.

In keeping with these considerations, we chose the method described by Kennedy and Marsden (1976). This is one of the so-called "surface singularity" or "panel" methods. It is thought to be the simplest available and provides exceptional accuracy for little computing effort. For the purpose of calculating the potential flow, the airfoil surface is approximated by N straight-line segments or "panels," labeled S_j, $j = 1,2, \ldots, N$. A constant, but unknown, vorticity density $\gamma(S_j)$ is distributed along each panel or control element. If this airfoil model is immersed in a uniform stream of unit velocity (nondimensionalized), at an angle of attack α, the stream function at any point external to the airfoil $P(x,y)$ is, according to elementary potential flow theory, given by:

$$\psi(x,y) = y\cos\alpha - x\sin\alpha - \frac{1}{2\pi}\sum_{j=1}^{N} \int_{S_j} \gamma(S_j) \ln r(P,S_j)\, dS_j \tag{1}$$

where $r(P,S_j)$ is the distance between point P and any point on the element S_j.

To solve for the unknown $\gamma(S_j)$ for each panel, eq 1 is applied at a control point, C_i, i = 1,2, ... , N, on each panel. Imposing the boundary condition, that the stream function be a constant along the airfoil surface (i.e. at each control point), and the Kutta condition, that the surface streamline leave the trailing edge smoothly, leads to a set of linear algebraic equations for the unknown vorticity densities. These matrix equations are solved in the usual way, and eq 1 then allows the determination of the stream function anywhere in the potential flow. The airspeed at any point can then be determined by differentiation of eq 1.

Other investigators (Bragg and Gregorek 1981) have adopted the conformal transformation approach, while still others (McComber and Touzot 1981) have solved Poisson's equation for the stream function using finite element methods. We believe, however, that the present method yields greater accuracy and spatial resolution for a similar computing effort.

Incompressible velocity field

The air velocity components may be calculated at any desired point in the airstream by differentiating the stream function; that is, by approximating the equations:

$$u = +\frac{\partial\psi}{\partial y} \qquad v = -\frac{\partial\psi}{\partial y} \tag{2}$$

with finite differences. This is done with a space increment, Δx or Δy, equal to the diameter of a cloud droplet. Thus it is possible to obtain a very accurate estimate of the airspeed at the position of the droplet's center of mass, wherever that happens to be. We believe that this approach is more accurate than that used by some other investigators (e.g. Cansdale 1980, private communication), who determine the airstream velocity field initially at a fixed array of grid points. When the air velocity at the droplet position is desired, an interpolation procedure among the grid point values is applied. Our approach of evaluating the air velocity as needed at precise points along the droplet trajectory, rather than by interpolation, is made possible by the economy with which ψ can be calculated using the Kennedy-Marsden approach.

Droplet trajectory equation

Pearcey and Hill (1956) have expressed the equation of motion of a spherical droplet of fixed mass in an accelerated air flow as:

$$\frac{d\bar{X}_d}{dt} = \bar{V}_d \tag{3}$$

$$\frac{d\bar{V}_d}{dt} = \frac{2(\varrho_d - \varrho_a)}{(2\varrho_d + \varrho_a)}\bar{g} \qquad \text{(Buoyancy term)}$$

$$- \frac{3C_D\varrho_a}{4r_d(2\varrho_d + \varrho_a)} |\bar{V}_d - \bar{V}_a|(\bar{V}_d - \bar{V}_a) \qquad \text{(Drag term)}$$

$$- \frac{9\varrho_a}{(2\varrho_d + \varrho_a)r_d} \sqrt{\frac{\upsilon_a}{\pi}} \int_{-\infty}^{t} \frac{d\bar{V}_d}{d\tau} \frac{d\tau}{\sqrt{t-\tau}} \qquad \text{(History term)} \tag{4}$$

where $\bar{X}_d(x_d,y_d)$ = droplet position vector
$\bar{V}_d(u_d,v_d)$ = droplet velocity vector
$\bar{V}_a(u_a,v_a)$ = air velocity vector
$\bar{g}$ = gravitational acceleration
C_D = steady-state droplet drag coefficient
ϱ_a = air density
ϱ_d = droplet density
r_d = droplet radius
ν_a = kinematic viscosity of the air
t = time.

The first term on the righthand side of eq 4 is the net buoyancy of the droplet in air. The second term is the steady drag, and the third is known as the history term (because of the time integral over the entire droplet history). The first two terms are probably in need of no explanation, although the gravitational term is frequently ignored in icing calculations (e.g. Langmuir and Blodgett 1946). The significance of the history term, however, may not be so apparent. It is essentially a correction to the drag term, which is necessary when the drag coefficient used in the second term is the steady-state value, appropriate for nonaccelerating droplets. For a given relative velocity between the droplet and the airstream, the true value of C_D is smaller for a drop that is accelerating with respect to the flow than for one that is not accelerating (i.e. one that is in equilibrium or steady state). This may be thought of as a phase lag effect, due to the finite rate of vorticity diffusion, which requires a certain time for the droplet to reach equilibrium with the airstream. Because of the large droplet acceleration that occurs in certain icing situations, we felt it important to examine the effects of the history term on the calculation of the droplet trajectories. Consequently, comparisons have been made between results calculated without the history term (referred to as the steady-state drag formulation) and those calculated with the history term included (referred to as the non-steady-state formulation). It should be noted that eq 4 also incorporates the effects of the droplet's induced mass resulting from the momentum it imparts to the air as it accelerates.

The formulation used to determine the steady-state drag coefficient as a function of droplet Reynolds number is given below:

1. $\mathrm{Re} < 0.01 \qquad C_D = 24/\mathrm{Re}_d$

2. $0.01 \le \mathrm{Re} \le 5 \qquad C_D = 24/\mathrm{Re}_d + 2.2$ (5)

3. $5 < \mathrm{Re} < 5000 \qquad C_D = 0.2924(1 + 9.06\ \mathrm{Re}_d^{-0.5})^2.$

The droplet Reynolds number is defined by:

$$\mathrm{Re}_d = \frac{2r_d\varrho_a}{\mu_a}\,|\bar{V}_d - \bar{V}_a|$$

where $\bar{V}_d$ and $\bar{V}_a$ are respectively the droplet and air velocity vectors, and μ_a is the dynamic viscosity of the air. The second formulation is from Sartor and Abbott (1975), while the third is given by Abraham (1970).

Computational procedure for trajectories

Equations 3 and 4 in component form yield four equations that, for the steady-state formulation, are numerically integrated using a fourth-order Runge-Kutta-Fehlberg method (Lapidus and Seinfeld 1971, Burden et al. 1978). This procedure permits the time step to be adjusted continuously for optimum speed of computation given a specified degree of accuracy required.

When the history term is incorporated into eq 4, it becomes a Voltera integro-differential equation of the second kind. The method of solution we used in this case is essentially the same as that used for the steady-state case, with the additional provision that the history term is approximated by a combined numerical and analytical technique. With this scheme, the integral is approximated by a finite sum at full time steps of the Runge-Kutta-Fehlberg method (for example, at τ and $\tau+\Delta\tau$). At intermediate time steps, however (between τ and $\tau+\Delta\tau$), the value of the integral is approximated by the extrapolation of a Legendre polynomial fitted to the previous values of the integral at full time steps.

Determining the point of impact

The droplet is assumed to have impinged upon the airfoil if any part of it contacts the airfoil surface. Thus, close to the airfoil, the finite size of the droplet is taken into account. This is particularly important for those trajectories just within the envelope of colliding trajectories, where the angle of incidence from the normal to the airfoil surface is close to 90°.

Calculation of collision efficiencies

To determine the local collision efficiency, β, at any point on the airfoil surface, use is made of the relation:

$$\beta(L) = \frac{dY}{dL} \cos\alpha$$

where Y is the ordinate at the starting point of a particular trajectory, L is the distance along the airfoil surface between the nose and the impact point of the same trajectory, and α is the angle of attack. By calculating the trajectories for between 10 and 20 droplets, Y may be plotted as a function of L, and the derivative taken to obtain β. These latter operations are in fact performed numerically using a cubic spline fit to the point values on the graph of Y vs L. The resulting local collision efficiencies may be plotted as a function of L (e.g. Fig. 2b) or of the corresponding abscissa X (e.g. Fig. 3a).

Accreting an ice layer

In the model, it is assumed that the ice growth on a particular small segment of the airfoil surface is oriented perpendicular to the surface. According to Lozowski et al. (1979), the accretion thickness is then given by the equation:

$$h(L) = \frac{2R(L)t_A}{\varrho_i} \bigg/ \left(1 + \sqrt{1 + \frac{2R(L)t_A}{\varrho_i r(L)}}\right) \tag{7}$$

where

$$R(L) = V_\infty w \beta(L) \tag{8}$$

is the icing flux with V_∞ the freestream velocity and w the liquid water content of the airstream, t_A is the period of accretion, ϱ_i the assumed ice density (890 kg m⁻³), and r the radius of curvature of the airfoil surface.

In the results presented here, we assume that time interval t_A is sufficiently small that the second term under the root in the denominator may be ignored.

By plotting the accretion thickness as a function of distance along the airfoil surface from the nose, it is possible to determine a new airfoil surface shape after it has iced for the specified time interval. The entire procedure can now be repeated, using the new iced airfoil surface to determine a new stream function, new droplet traject. .es, and ultimately a second accretion layer. By continuing in this manner, it is possible to build up a substantial ice accretion on the airfoil.

Determining the accuracy of the flow field

The accuracy of the Kennedy-Marsden technique was tested by comparing its predicted stream function for potential flow around a cylinder with the known analytic solution. Using 50 control elements, the error in ψ is at most 0.1% near the cylinder. It falls to below 0.01% at distances from the surface exceeding about four cylinder diameters.

We have also made a comparison between the corresponding velocity fields. In one such test for example, using a cylinder diameter of 0.15 m, an air pressure of 78.5 kPa, an air temperature of $-10\,°C$, and a freestream velocity of 114.3 m s^{-1}, the velocity field of the analytic solution was compared with that provided by the model using 38 control elements. At a distance of five diameters upstream, the air velocities differed by less than 10^{-2}%. Very close to the cylinder they were as high as 1 to 2%. However, the effect of these airstream velocity errors on the computed droplet collision efficiencies was found to be much less than 1%.

Determining the accuracy of the trajectories

To establish some confidence in the trajectories themselves, it was decided to make a comparison with two cases considered by Langmuir and Blodgett (1946). The two cases were chosen to check our method of trajectory calculation for both high and low collision efficiencies. For both cases, Langmuir's ϕ parameter was chosen to be 10^4. This is given by:

$$\phi = 9\,\frac{\varrho_a^2 C V_\infty}{\mu_a \varrho_d} \tag{9}$$

where ϱ_d = droplet density
ϱ_a = air density
μ_a = dynamic viscosity of air
C = cylinder diameter
V_∞ = freestream speed.

ϕ is a nondimensional impingement parameter. Large values of ϕ imply a large radius of curvature of the streamlines, and vice versa. Langmuir's K parameter was 36.0 in the first case and 1.0 in the second. K is given by the expression:

$$K = \frac{4\varrho_d r_d^2 V_\infty}{9\mu_a C} \tag{10}$$

where r_d is the droplet radius. K is the nondimensional inertia parameter. It is the ratio of the droplet's projectile range under Stokes' law to the radius of the cylinder. If K is small, the droplets tend to follow the streamlines, and, hence, the collision efficiency tends to be low.

In these cases, as in all the experiments considered in this report, the droplets were introduced into the airstream five chord lengths upstream of the nose of the target with a Reynolds number Re_d of 0.001. Ideally, the droplet trajectory integration should begin infinitely far upstream with the droplets having the same velocity as the air (i.e. $Re_d = 0$), but for computational reasons this is impractical. Tests indicate that the trajectory errors caused by this imperfect initial condition are smaller than the numerical integration errors.

The parameters chosen for the two cases considered are given in Table 1. Table 1 also presents a comparison between our results and those of Langmuir and Blodgett (1946). The

Table 1. Icing on a cylinder, present calculations (rows 2, 4, 5) vs Langmuir and Blodgett (rows 1, 3).

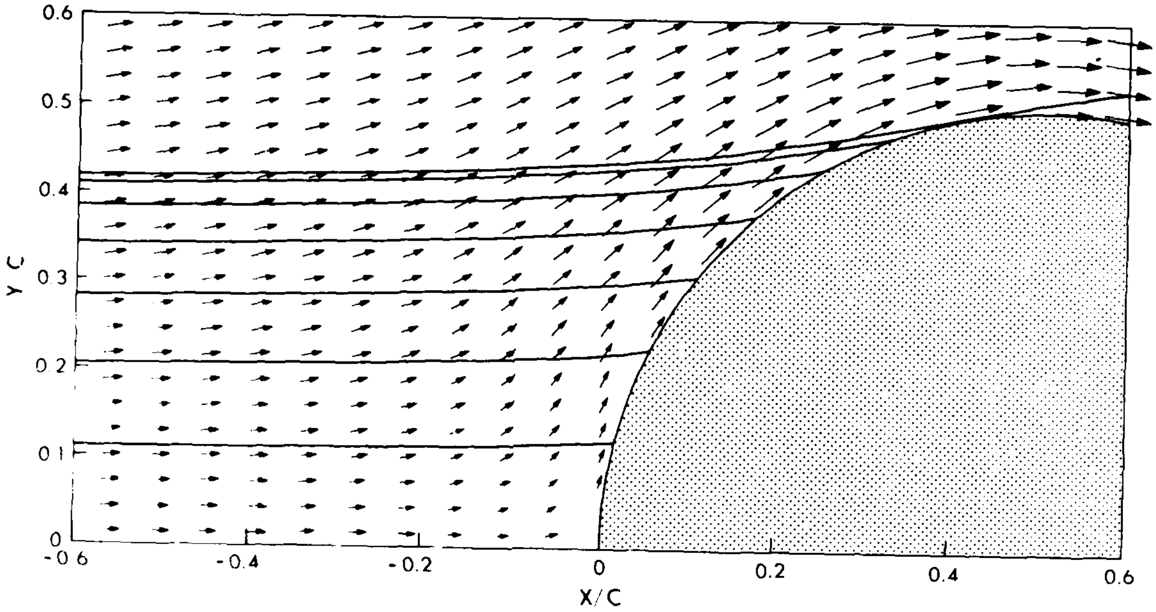
ϕ	K	T (°C)	P (kPa)	ρ_d (kg m^{-3})	C (m)	V_∞ (m s^{-1})	r_d (μm)	v_x	v_y	E_m (%)	β_0 (%)	θ_m (deg.)	History term
10,000	36	—	—	—	—	—	—	1.056	0.193	81.9	88.5	79.8	No (L&B)
10,000	36	-10	78.5	999.15	0.15	114.3	42.1	1.056	0.196	81.4	89.8	79.5	No (Model)
10,000	1	—	—	—	—	—	—	0.494	0.725	15.6	34.8	34.2	No (L&B)
10,000	1	-10	78.5	999.15	0.15	114.3	7.02	0.477	0.650	17.0	37.6	35.6	No (Model)
10,000	1	-10	78.5	999.15	0.15	114.3	7.02	0.527	0.662	18.7	39.1	38.9	Yes (Model)

symbols v_x and v_y denote the droplet impact velocity components in the x- and y-directions, respectively. θ_m denotes the maximum angle of droplet impingement from the forward stagnation point. Our model results for v_x, v_y, E_m, β_0 and θ_m (given in rows 2 and 4) compare favorably with those of Langmuir and Blodgett (given in rows 1 and 3). The discrepancies, which are larger for the case with the smaller inertia parameter K, are quite acceptable, if one recognizes that the Langmuir and Blodgett data should not be viewed as an absolute standard.

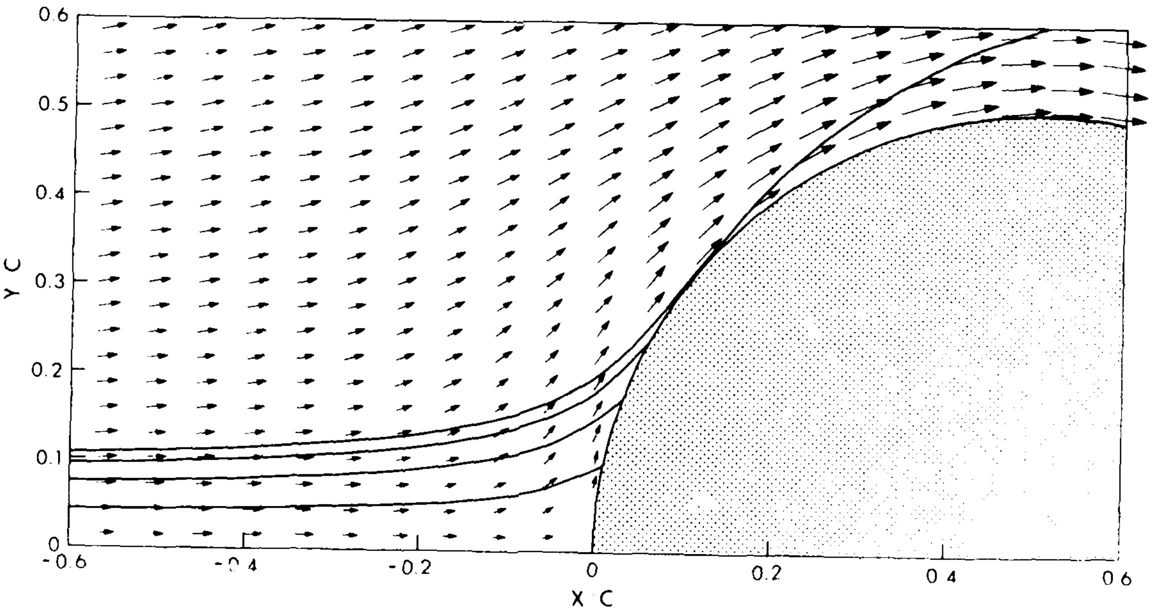
Figure 1 shows the flow field (indicated by velocity vectors) and the droplet trajectories (indicated by solid curves) for flow about the cylinder in the two cases just considered. It is interesting to note the much higher curvature of the trajectories for the less massive drops and the larger shadow zone between the grazing trajectory and the cylinder. One might also speculate on the collisions that could occur between the small and large drops, because of the way the smaller ones track across the trajectories of the large ones. It is hard to see, however, how such collisions might have any significant effect on the icing.

Figure 2 displays the collision efficiency for the two cases as a function of the nondimensional distances along the cylinder surface from the stagnation point (that is, actual distance L divided by cylinder diameter C). Negative values of L/C lie below the stagnation point, while positive values lie above it. Figure 2a is almost a cosine curve, a result that would occur if the droplet trajectories were straight lines. The slight "kinks" in Figure 2b are artificial and arise from the numerical spline fitting procedure. The overall collision efficiency is equal to the total area under the curves.

Figure 3 is similar to Figure 2, except that the abscissa is now X/C, where X is the projection of the arc length L onto the x-axis. The effect of this change in abscissa is to "squeeze" the curves in towards the origin. This squeezing is greatest near the origin, so that the most apparent effect is to sharpen the peak in the curves. Although it is somewhat redundant to present collision efficiencies as functions of X/C and L/C for a cylinder, the difference is more meaningful for airfoils, as some of the historical papers prefer X/C and others prefer L/C. For airfoils, a plot of collision efficiency vs L/C provides more resolution near the nose.

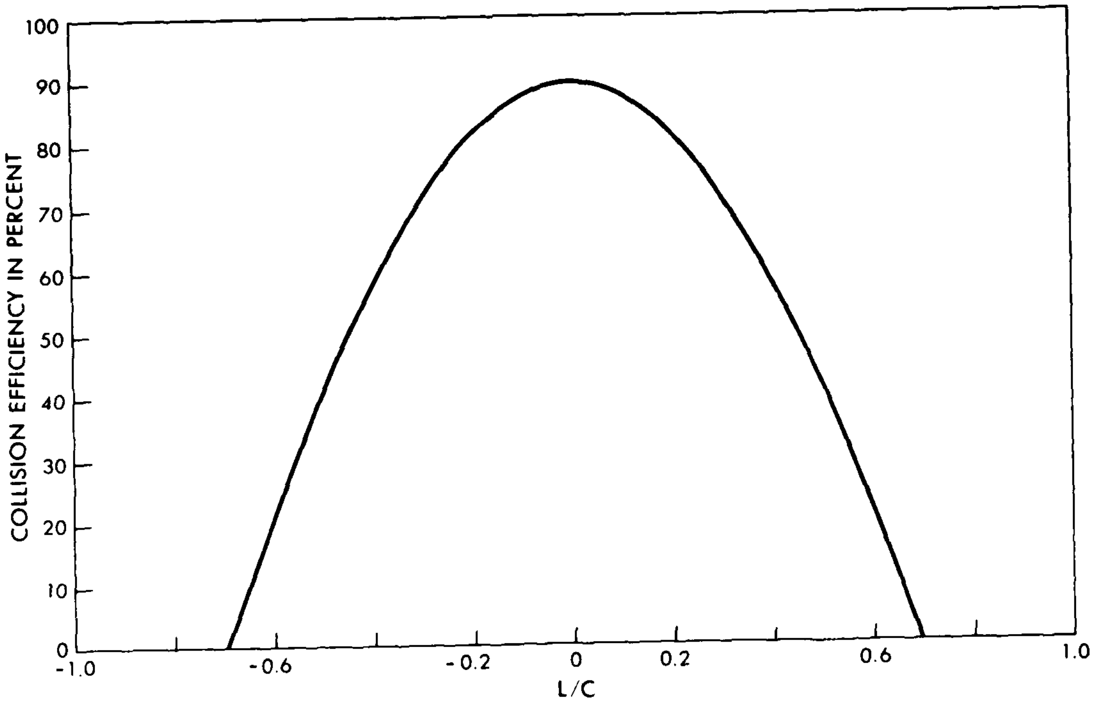


a. C = *15 cm;* V_∞ = *114.3 m s*$^{-1}$; r_d = *42.1 μm.*

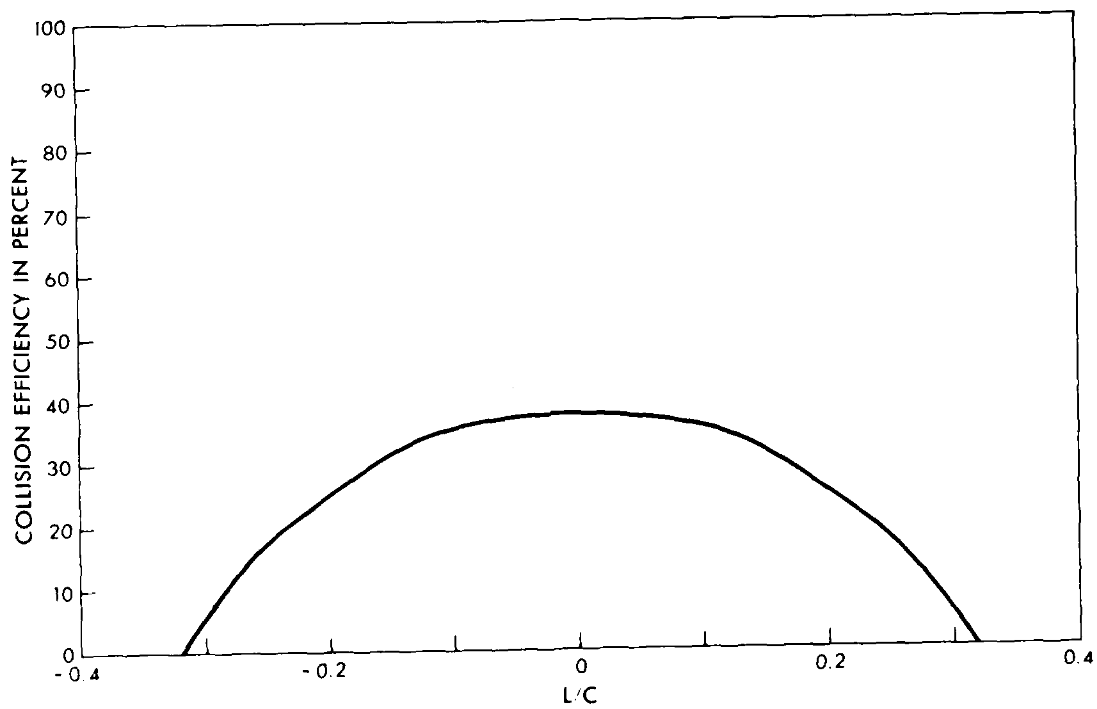


b. C = *15 cm;* V_∞ = *114.3 m s*$^{-1}$; r_d = *7.0 μm.*

Figure 1. Trajectories about a cylinder.

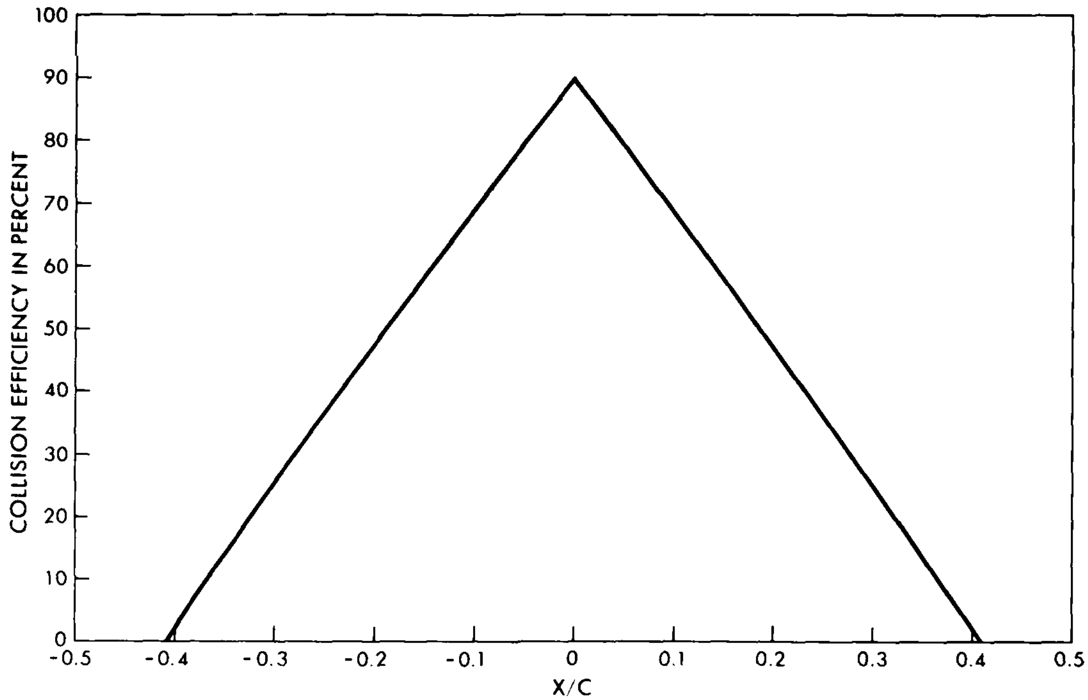


a. C = *15 cm;* V_∞ = *114.3 m s*$^{-1}$*;* r_d = *42.1 µm.*

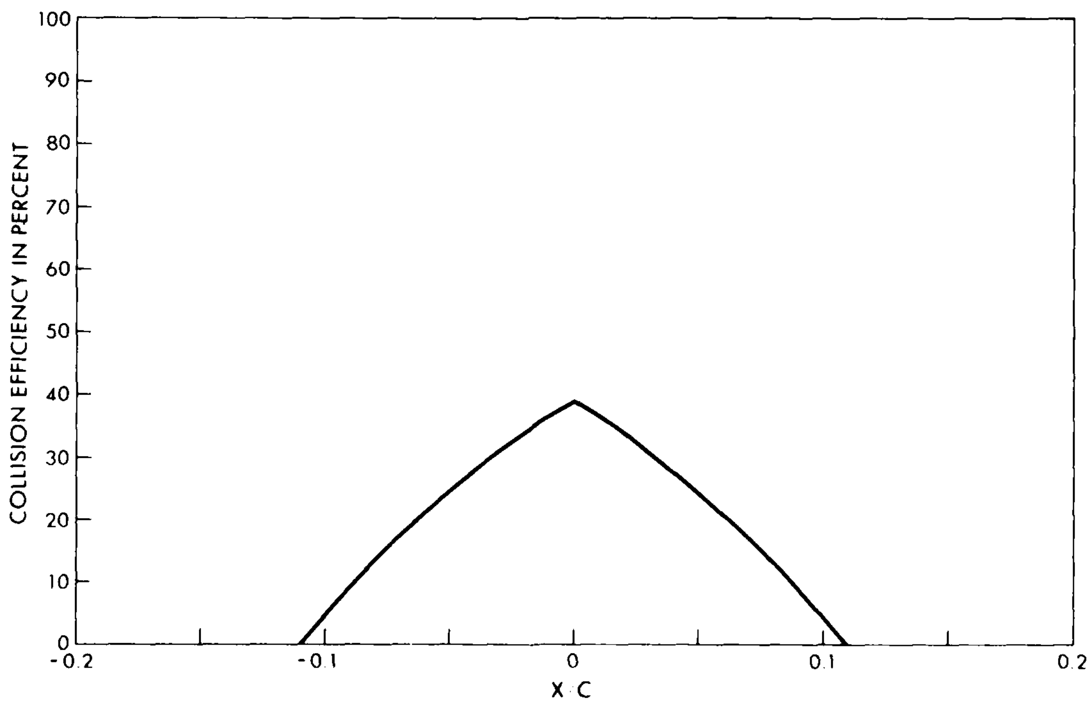


b. C = *15 cm;* V_∞ = *114.3 m s*$^{-1}$*;* r_d = *7.0 µm.*

Figure 2. Collision efficiency vs length along cylinder surface.



a. C = *15 cm;* V_∞ = *114.3 m* s^{-1}*;* r_d = *42.1 μm.*



b. C = *15 cm;* V_∞ = *114.3 m* s^{-1}*;* r_d = *7.0 μm.*

Figure 3. Collision efficiency vs distance along chord.

RESULTS AND DISCUSSION

Comparing results with and without the history term

Rows 4 and 5 of Table 1 present results of numerical experiments run respectively without and including the history term. All the other conditions of the experiment are identical. The droplet trajetories calculated with the history term are less influenced by the rapid changes in the airflow just before the collision, and so they tend to travel along straighter paths than those whose trajectories ignore the history term. This is explained by the fact that the history term acts to reduce the droplet accele ation, because it takes into account the finite rate at which vorticity is shed from the accelerating droplet. The net result is that in this case, ignoring the history term reduces the maximum impingement angle, θ_m, by 3.3°, reduces the stagnation line collection efficiency, β_0, by 1.5%, and reduces the overall collection efficiency, E_m, by 1.7% (about 10% of the actual value). The particular case used to study the influence of the history term was chosen to give a large effect. In most cases, the effects would probably be less than those indicated here, suggesting that the term may be ignored without severely affecting the accuracy of the calculations.

Collision efficiency of NACA 0015 airfoil at 8° attack angle

Thus far, the computational icing experiments have been limited to cylinders. Let us now consider the case of a NACA 0015 airfoil at an attack angle of 8°. The chord length is 16.9 cm, the freestream speed 30.5 m s^{-1}, and the droplet diameter 20 μm. The history term is not included in the computation. Figure 4 illustrates the resulting airflow (indicated by velocity vectors) and droplet trajectories (indicated by solid curves) for this case. It should be kept in mind that the flow region depicted in the figure is only a small portion of the total flow considered. In addition, the coordinate system is fixed to the airfoil so that the flow appears to be impinging upwards in a horizontally oriented airfoil. In fact, the entire figure should be rotated clockwise by 8°.

The droplet trajectories clearly indicate the asymmetry of the impingement above and below the stagnation point, when the airfoil is not at 0° attack. This is generally reflected in different icing characteristics above and below the stagnation line. Figure 5, which depicts the local collision efficiency, also illustrates this asymmetry. Negative values of L/C lie below the airfoil nose and positive values above it. The collision efficiency is a maximum close to (though not necessarily at) the stagnation line. The overall collision efficiency for this case is 50.1% and the maximum is 74.4% at a distance $L/C = -0.009$ below the nose. A comparison of Figure 5 has been made with the results of Bragg (private communication), who has also recently investigated this problem (see, for example, Bragg and Gregorek 1981). The differences between the two sets of computed results are generally negligible, in the sense that experiments would not likely be of sufficient accuracy to allow one to choose between the two.

Time-dependent accretion on NACA 0015 airfoil at 8° attack angle

The collision efficiencies plotted in Figure 5 are those for the airfoil surface itself at the onset of icing. Once a significant accretion has built up on the airfoil, the collision efficiencies change, and this change affects the subsequent development of the accretion. This feedback process between the accretion and the airflow and droplet trajectories goes on continuously in nature. We decided to simulate the continuous process in a step-wise fashion. Thus, using the computed initial collision efficiencies, we estimate the profile of the ice accretion after a finite, but small, time interval. We then use this new airfoil profile (including the already accreted ice layer) to determine a new flow field, droplet trajectories, and collision efficiencies. After that, a new increment to the accretion is once again calculated, and the entire process is repeated for as long a total period as desired. Other authors (e.g. Lozowski et

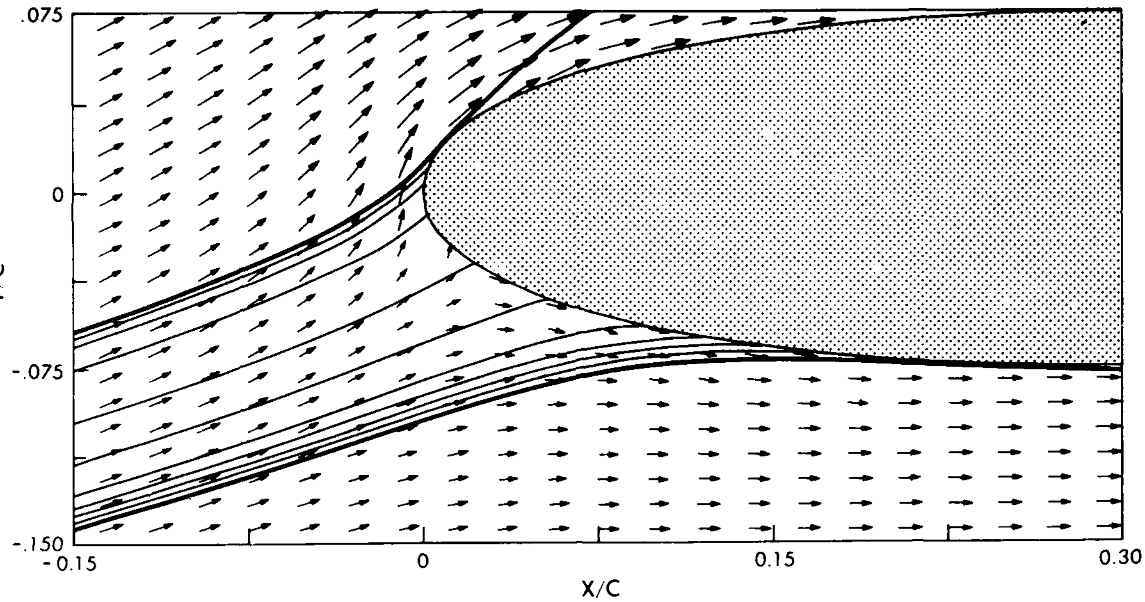


Figure 4. Trajectories about a NACA 0015 airfoil at 8° attack angle. C = *16.9 cm;* V_∞ = *30.5 m* s^{-1}*;* r_d = *10 µm.*

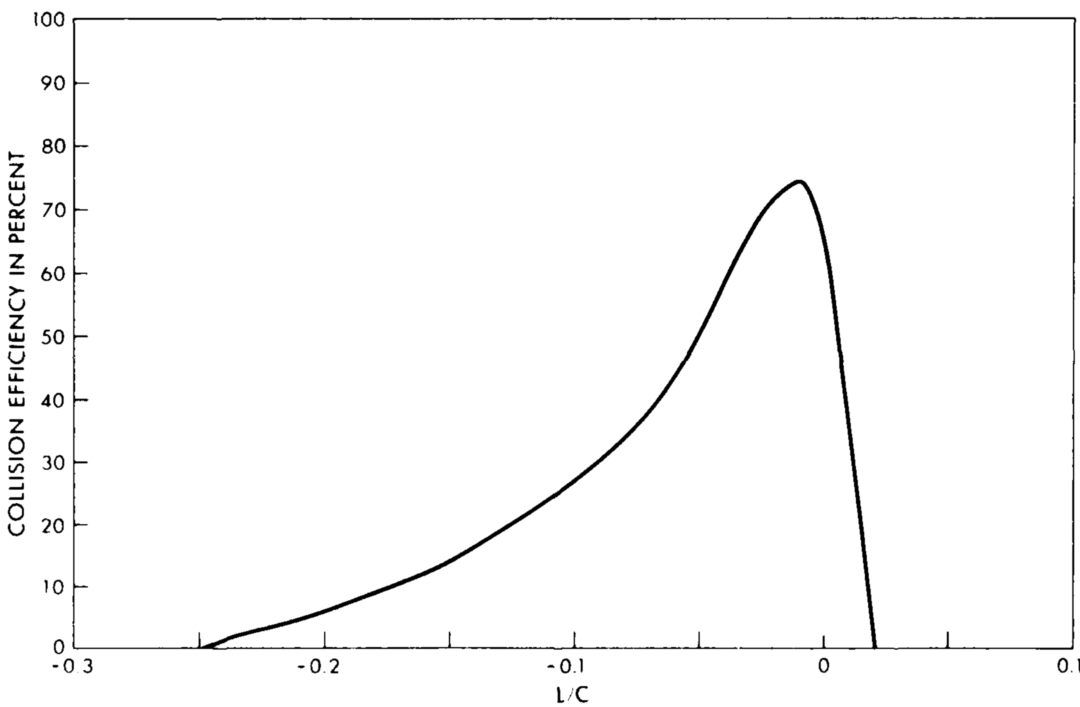


Figure 5. Collision efficiency vs length along airfoil surface (NACA 0015 airfoil at 8° attack angle). C = *16.9 cm;* V_∞ = *30.5 m* s^{-1}*;* r_d = *10 µm.*

al. 1979) have not taken this feedback process into account, but instead have used the initial collision efficiencies to try to extrapolate the growth of the accretion over substantial periods of time. In this section we compare these two procedures: viz. single-step accretion vs multi-step with feedback. We also make a comparison with an experimental accretion grown in the NRC high-speed icing tunnel.

Figure 6 shows the airflow and the droplet trajectories for a NACA 0015 airfoil at an 8° angle of attack, but the conditions are somewhat different from those of Figure 4. In the present case, the chord length is 21.3 cm, the freestream speed 61 m s^{-1}, and the droplet diameter 20 μm. The history term is not included in this simulation. The solid line in Figure 7 shows the initial local collision efficiency β vs the nondimensional length L/C along the airfoil surface. Based on these values of β, and assuming a cloud liquid water content of 0.4 g m^{-3} and an accretion density of 890 kg m^{-3}, the accretion growth in a single step over 2.5 minutes was calculated. The modified airfoil profile, with the ice accretion attached, was then used as a basis for calculating a new airflow and new droplet trajectories. From these, a new determination was made of the local collision efficiency after 2.5 minutes of icing. This is indicated as the dashed curve in Figure 7 (L now being measured from the nose of the accretion rather than from the nose of the airfoil). Although the differences between the solid and dashed curves are not striking, it is clear that there are some. In particular, E_m falls with time from 58.2% to 56.5% while β_0, the maximum collision efficiency, actually rises from 75.5% to 78.9%. Although the comparison is difficult to interpret because L/C has a slightly different meaning in each case (although C itself remains the chord length of the basic airfoil), it is apparent that the collision efficiency distribution has narrowed and become more peaked as a result of the change in the airflow caused by the first 2.5 minutes of accretion.

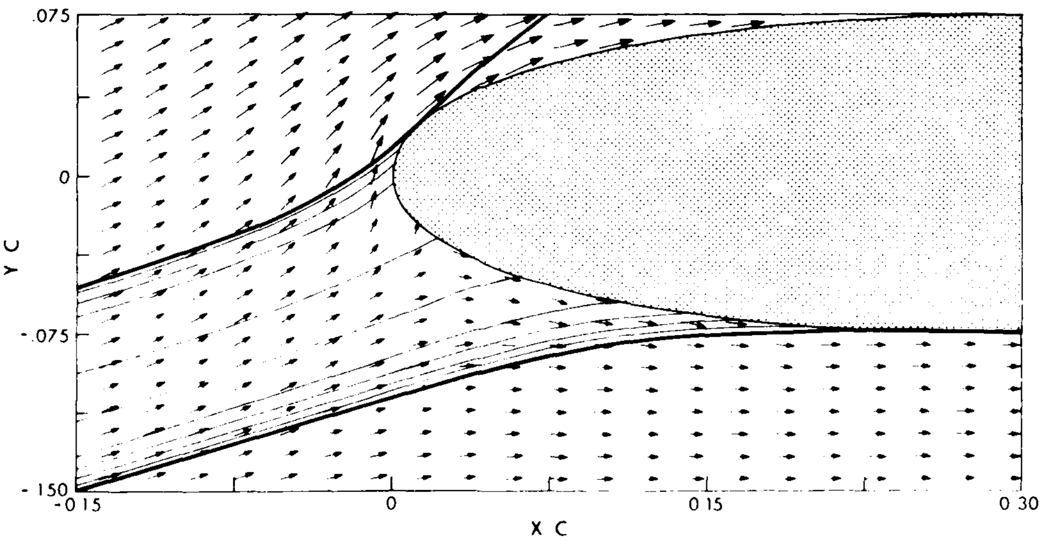


Figure 6. Trajectories about a NACA 0015 airfoil at 8° attack angle. C = 21.3 cm; V_∞ = 61 m s^{-1}; r_d = 10 μm.

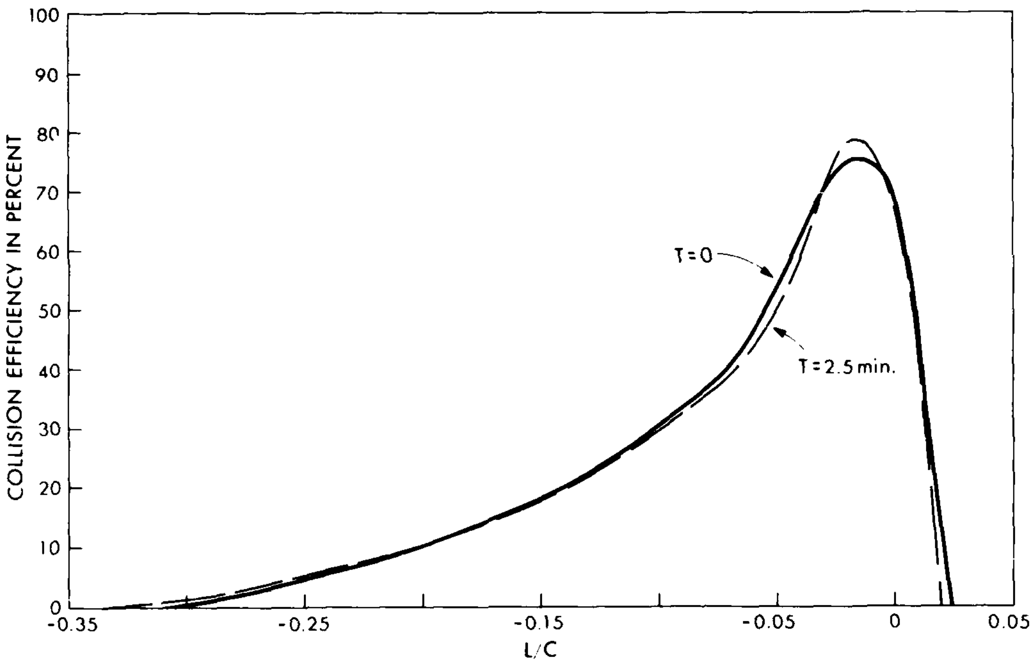


Figure 7. Collision efficiency vs length along airfoil surface (NACA 0015 airfoil at 8° attack angle). C = 21.3 cm; V_∞ = 61 m s⁻¹; r_d = 10 μm.

The decrease in the collision efficiency that occurs with time on the lower surface near the nose is illustrated in Figure 8, which shows the accretion profiles after the addition of two successive 2.5-minute accretions. This two-step accretion shape is compared in Figure 9 with a single-step 5-minute accretion, calculated using only the initial collision efficiencies. The single-step accretion model overestimates the growth above and below the nose and underestimates it at the nose itself. Although the differences between the single and multistep models may seem relatively minor over this period of time, they will be much more significant over longer periods, the multistep method giving much more realistic results.

The shape of an experimental accretion profile made under similar conditions in the NRC high-speed icing tunnel (Stallabrass and Lozowski 1978) is also shown in Figure 9. The experimental and theoretical results are not perfectly comparable because a droplet size (approximately equal to the medium volume diameter of the tunnel droplet spectrum) was employed in the model. The general agreement is quite encouraging, though one gets the impression that the model accretion occurs too low on the airfoil relative to the experimental one. This discrepancy may be the result of a bias error in the model. On the other hand, it may have to do with the way the experimental profiles were measured. The experimental profiles were obtained by making an impression in plasticene and then photographing their outline against the outline of the airfoil. Inaccurate registration of the airfoil outline and that of the plasticene mold may have displaced the experimental profile upwards from where it should be. Only further experiments, with an improved technique for measuring the experimental profile, can resolve which is the correct explanation.

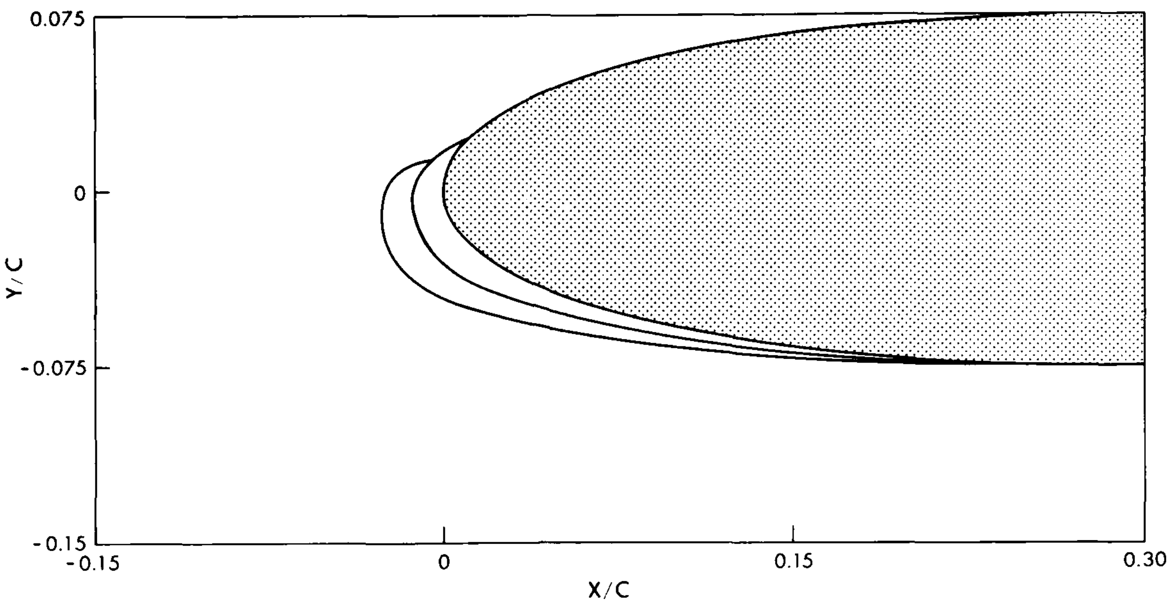


Figure 8. Accretion after 2.5 min and 5 min on NACA 0015 airfoil at 8° attack angle. C = *21.3 cm;* V_∞ = *61 m* s^{-1}*;* r_d = *10 μm;* LWC = *0.4 g* m^{-3}.

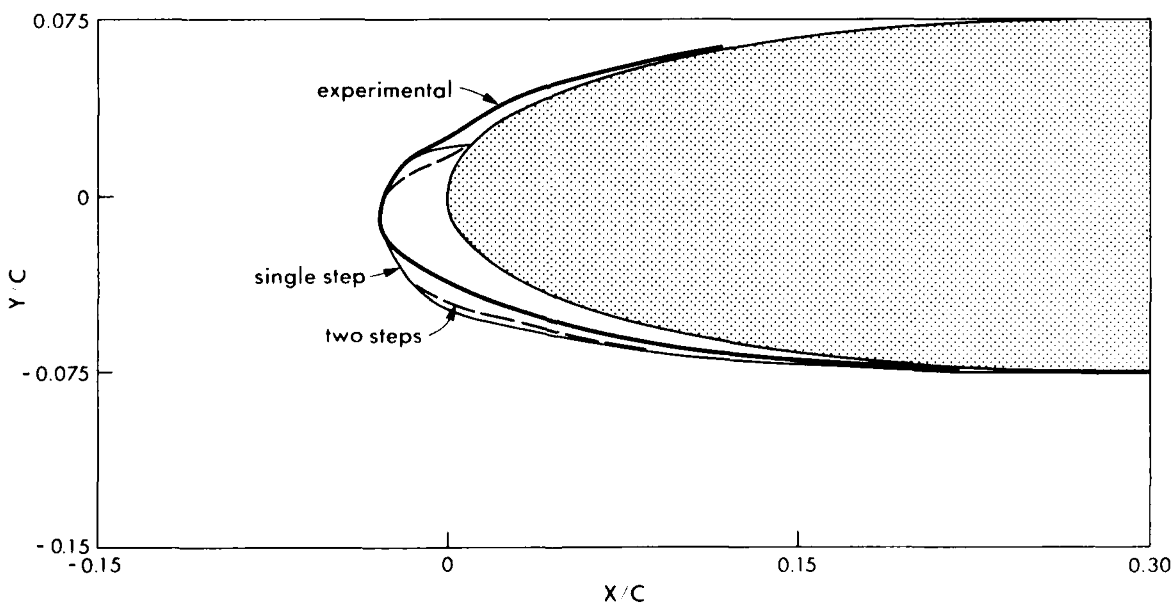


Figure 9. Accretion after 5 min in a single step (solid), two steps (dashed), and experimentally (bold) (NACA 0015 airfoil at 8° attack angle). C = *21.3 cm;* V_∞ = *61 m* s^{-1}*;* r_d = *10 μm;* LWC = *0.4 g* m^{-3}.

Time-dependent accretion on NACA 0015 airfoil at 0° attack angle

In this section we demonstrate that the multistep accretion process can be continued for as many as five steps, after each of which a new potential flow and new droplet trajectories are calculated. We have not as yet attempted to increase the number of steps beyond five, although we see no reason why, in principle, this could not be done. Figure 10 illustrates the airflow and droplet trajectories at the initial time before ice accretion begins. The airfoil chord length is 21.3 cm, the freestream velocity 91.5 m s^{-1}, and the droplet diameter 20 μm. To be rigorous, the history term was included in the trajectory calculations for this simulation, although generally speaking its effect may not be large.

Figure 11 shows the collision efficiency at the initial time and after 3 minutes of ice accretion. The local collision efficiency increases with time near the nose and diminishes with time farther back along the airfoil surface. Table 2 shows that the overall collision efficiency decreases with time, while the collision efficiency at the nose increases with time.

The result of this effect on the accretion itself is shown in Figure 12, where we see that the accretion tends to become more "pointed" with time, and that the growth rate at the nose in the model increases with time. This result seems reasonable inasmuch as the effect of the accretion is to decrease the local radius of curvature at the nose, thereby enhancing the collision efficiency and increasing the growth rate. Unfortunately, no time-dependent experimental growth rate measurements are available to confirm this result. Figure 13 compares the resulting accretion for the multistep approach with that obtained using a single 5-minute step. The single-step accretion slightly underestimates the growth at the nose and greatly overestimates it farther back.

An experimental ice accretion profile grown under similar conditions in the NRC high-speed icing tunnel is included in Figure 13 for comparison. Although the agreement is good at the nose, a substantial difference occurs farther back. We suspect that this is due to the growth of low-density feathery rime in the experimental case. Because we assume an ice density of 890 kg m^{-3} in the model, feathery rime growth is not taken into account.

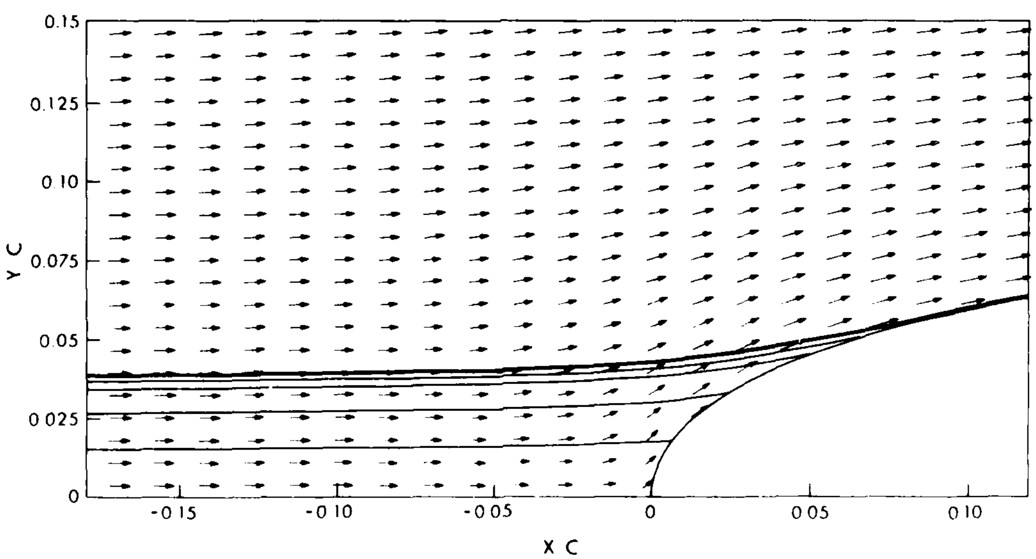


Figure 10. Trajectories about a NACA 0015 airfoil at 0° attack angle. C = 21.3 cm; V_∞ = 91.5 m s^{-1}; r_d = 10 μm; LWC = 0.4 g m^{-3}.

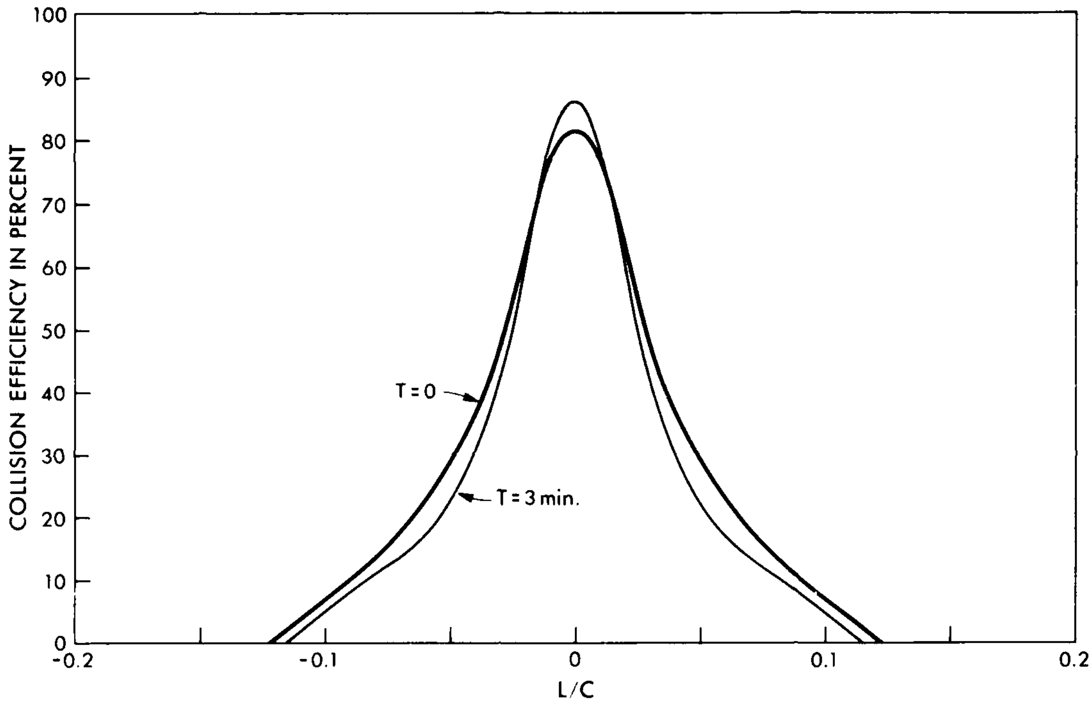


Figure 11. Collision efficiency vs length along airfoil surface (NACA 0015 airfoil at 0° attack angle). C = *21.3 cm;* V_∞ = *91.5 m* s^{-1}; r_d = *10 μm;* LWC = *0.4 g* m^{-3}.

Table 2. Collision efficiencies as a function of time for the case of Figure 10.

Time (min)	E_m (%)	β_0 (%)
0	49.1	81.8
1	47.7	83.8
2	46.1	85.5
3	44.7	86.3
4	43.3	87.1

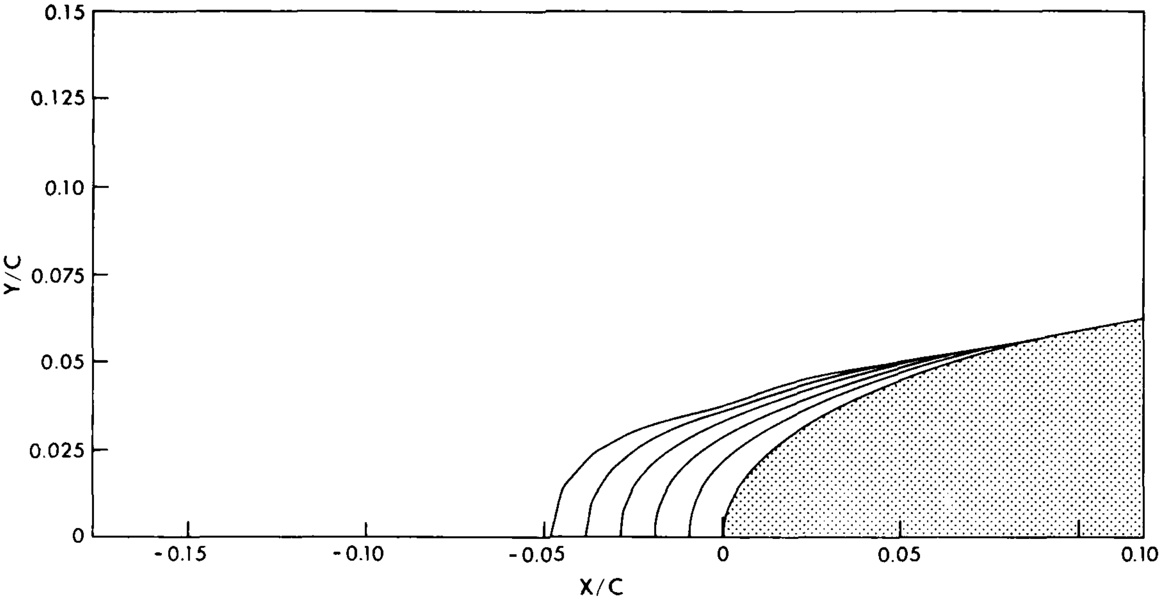


Figure 12. Accretion after 1 through 5 min on a NACA 0015 airfoil at 0° attack angle. C = *21.3 cm;* V_∞ = *91.5 m* s^{-1}*;* r_d = *10* μm*;* LWC = *0.4 g* m^{-3}.

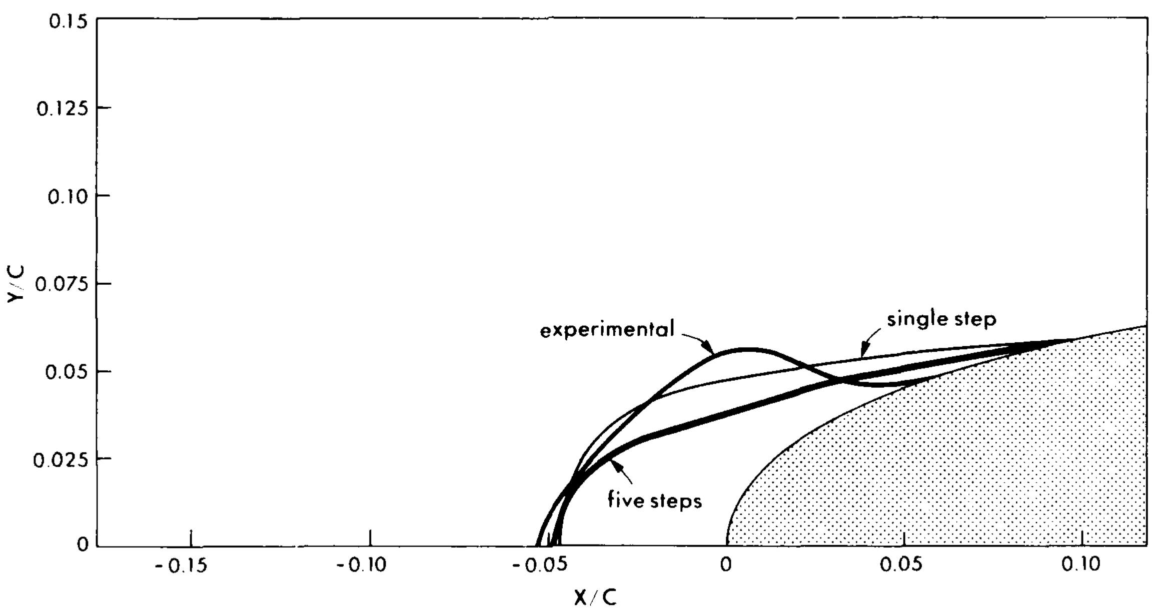


Figure 13. Accretion after 5 min in a single step (solid), five steps (boldest), and experimentally (bold) (NACA 0015 airfoil at 0° attack angle). C = *21.3 cm;* V_∞ = *91.5 m* s^{-1}*;* r_d = *10* μm*;* LWC = *0.4 g* m^{-3}.

CONCLUSIONS AND RECOMMENDATIONS

1. The principal accomplishment of the work performed under the present contract has been the development and testing of a computational simulation model of rime icing on arbitrary two-dimensional airfoils. The computer code for the model is presented in Appendix C. The program is annotated so that it should be possible for scientists elsewhere to run the program, check the present results, and develop new results for their own applications. Should any difficulties be encountered in the implementation of the model program, the authors will be pleased to offer their advice and assistance.

2. Most of the model runs described in the present report have been performed to test the accuracy of the components of the model. The potential flow field was tested against the known analytic solution for a circular cylinder, and was found to behave acceptably (stream function errors of 0.1% or less, which give rise to collision efficiency errors of less than 0.5%). The accuracy of the droplet trajectories was determined by comparing the model-predicted collision efficiencies with those computed by Langmuir and Blodgett (1946). Relative agreement to better than 10% was found even in the worst case. Finally, the ice accretion profiles themselves were tested against two experimental accretions, and, while the agreement was not exact, it was encouraging as to the model's simulation capabilities.

3. The other model runs presented in this report were performed either to test the importance of the history term in the droplet equations of motion or to compare a single-step vs a multistep accretion process. Although these tests were not exhaustive, they did indicate that omitting the history term did not have a dramatic effect on the results. The biggest effect of the history term occurred in cases with low collision efficiencies. The tests also showed that the accretion profiles predicted by the single-step and multistep processes are different, and that the difference increases with the duration of the accretion. As a result of these tests, and because in principal the multistep accretion procedure better simulates what is happening in nature, we recommend that the single-step approach be used only for brief accretion durations. Thus, for example, a single-step model might be quite useful for helicopter deicing calculations. On the other hand, the multistep method would be preferable for simulations of powerline icing where the duration may be hours or days.

4. Within the scope of the present contract, it has not been possible to use the model to investigate the effects of various parameters on the shape and development of the accretion. We recommend, however, that such studies be undertaken with the model. Although the model is presently limited to simulating rime accretions, there are many questions that it can be used to investigate. What is the effect of airfoil size and shape on the accretion? What would be the effect on the accretion of using a realistic cloud droplet distribution (see, for example, Ackley and Templeton 1979) rather than a single droplet size? What parameters should be simulated to properly model ice accretions at a reduced scale? How is the accretion changed if the airfoil attack angle and the airstream speed oscillate as they would for a helicopter rotor blade? Such questions and many others could and should be profitably considered using the present icing model.

LITERATURE CITED

Abraham, F.F. (1970) Functional dependence of the drag coefficient of a sphere on Reynolds number. *Physics of Fluids,* **13:** 2194–2195.

Ackley, S.F. and M.K. Templeton (1979) Computer modeling of atmospheric ice accretion. U.S. Army Cold Regions Research and Engineering Laboratory, CRREL Report 79-4. ADA 068582.

Bragg, M.B. and G.M. Gregorek (1981) An analytical approach to airfoil icing. AIAA 19th Aerospace Sciences Meeting, St. Louis, 12-15 January 1981. New York: American Institute of Aeronautics and Astronautics, AIAA-81-0403.

Burden, R.L., J.D. Faires and A.C. Reynolds (1978) *Numerical Analysis.* Boston: Prindle, Weber & Schmidt, 579 pp.

Cansdale, J.T. and I.I. McNaughtan (1977) Calculation of surface temperature accretion rate in a mixed water droplet/ice crystal cloud. Royal Aircraft Establishment, Technical Report 77090, 24 pp.

Crowe, C.T., J.A. Nicholls and R.B. Morrison (1963) Drag coefficients of inert and burning particles accelerating in gas streams. In *Ninth Symposium on Combustion.* New York: Academic Press, pp. 395–405.

Hess, J.L. and A.M.O. Smith (1967) Calculation of potential flow about arbitrary bodies. *Progress in Aeronautical Science,* **8.**

Kennedy, J.L. and D.J. Marsden (1976) Potential flow velocity distributions on multi-component airfoil sections. *Canadian Aeronautics and Space Journal,* **22:** 243–256.

Langmuir, I. and K.B. Blodgett (1946) A mathematical investigation of water droplet trajectories. In *Collected Works of Irving Langmuir.* New York: Pergamon Press, **10:** 348-393.

Lapidus, L. and J.H. Seinfeld (1971) *Numerical Solutions of Ordinary Differential Equations.* New York: Academic Press, 299 pp.

Lozowski, E.P. and M.M. Oleskiw (1981) Computer simulation of airfoil icing without runback. AIAA 19th Aerospace Sciences Meeting, St. Louis, 12-15 January 1981. New York: American Institute of Aeronautics and Astronautics, AIAA-81-0402.

Lozowski, E.P., J.R. Stallabrass and P.F. Hearty (1979) The icing of an unheated nonrotating cylinder in liquid water droplet-ice crystal clouds. National Research Council of Canada, Report LTR-LT-96, 109 pp.

McComber, P. and G. Touzot (1981) Calculation of the impingement of cloud droplets on a cylinder by the finite-element method. *Journal of the Atmospheric Sciences,* **38:** 1027-1036.

Oleskiw, M.M. and E.P. Lozowski (1980) Helicopter rotor blade icing: A numerical simulation. In *Proceedings of the 8th International Conference on Cloud Physics, Clermont-Ferrand, France, 15-19 July 1980.* International Commission on Cloud Physics.

Pearcey, T. and G.W. Hill (1956) The accelerated motion of droplets and bubbles. *Australian Journal of Physics,* **9:** 19-30.

Rosen, K.M. and M.L. Potash (1981) 40 years of helicopter ice protection experience at Sikorsky Aircraft. AIAA 19th Aerospace Sciences Meeting, St. Louis, 12-15 January 1981. New York: American Institute of Aeronautics and Astronautics, AIAA-81-0407.

Sartor, J.D. and C.E. Abbott (1975) Prediction and measurement of the accelerated motion of water drops in air. *Journal of Applied Meteorology,* **14:** 232-239.

Stallabrass, J.R. and E.P. Lozowski (1978) Ice shapes on cylinders and rotor blades. Paper presented at NATO Panel X Symposium on Helicopter Icing, November 1978, London.

Theodorson, T. and I.E. Garrick (1932) General potential theory of arbitrary wing sections. NACA Report 452. Washington, D.C.: National Advisory Committee on Aeronautics.

APPENDIX A: SAMPLE INPUT

This appendix contains a sample of the parameters that must be input to the program along with examples of their typical values. These parameters are read by the program from input device 4 (see, for example, program line 186).

```
NEF,NEB,NIF,
  9, 11,  3,
ALPHA,TYPE,THICK,XMIN,XMAX,YMIN,YMAX,XZ,YZ,ANAL,
  0.0,   1,100.0,-0.6, 0.6, 0.0, 0.6,43,43,   1,
PLTFAC,TRJPLA,YOL,CEL,CEX,ICEPLA,LYRMAX,CETOL,   ICE,
   1.0,     1,  1,  1,  1,     1,     1,  0.3,  0.05,
 UINF,    C, PINF, TINF,   RD,        A1,       A2,
114.3, 0.15, 78.5,-10.0, 42.1,      0.D0,  6.25D-2,
CDS,TRJPRA,PRINTI,PLOTI,PRINTO,CRIT,BETAO,
  1,     1,    25,    50,    10, 1.0, 0.89,
NTRAJU,NTRAJL,AT,BOTH,EQN,PC,  DTS,    EPS,ACN,
     6,     0, 1,   0,  1, 2,.06D0, 1.D-6,  0,
XO,
-5.0,
YO,...  FIRST LAYER
0.4065,
YO,...  SECOND LAYER
0.0,
YO,..   THIRD LAYER
0.0,
YO,...  FOURTH LAYER
0.0,
YO,. .  FIFTH LAYER
0.0,
END OF FILE
```

APPENDIX B: SAMPLE OUTPUT

This appendix contains a sample of the printed output produced by the program starting with the sample input values given in Appendix A. This output is written to output device 7. The trajectories calculated correspond to those of Figure 1. The collision efficiency data (on the last page) correspond to Figures 3 and 5.

```
FOR EQN. SOLN. IER=  0

THE POTENTIAL FLOW LIFT COEFFICIENT IS  0.00000
```

CONTROL PT.	X COORD.	Y COORD.	SFC. VEL.
1	0.00380	0.04341	-0.17408
2	0.01887	0.12892	-0.51694
3	0.04857	0.21050	-0.84409
4	0.09198	0.28570	-1.14558
5	0.14779	0.35221	-1.41223
6	0.21430	0.40802	-1.63591
7	0.28949	0.45143	-1.80976
8	0.37108	0.48113	-1.92846
9	0.45659	0.49620	-1.98401
10	0.53911	0.49692	-1.99819
11	0.61636	0.48469	-1.94391
12	0.69075	0.46052	-1.84660
13	0.76044	0.42501	-1.70398
14	0.82372	0.37903	-1.51956
15	0.87903	0.32372	-1.29778
16	0.92501	0.26044	-1.04407
17	0.96052	0.19075	-0.76468
18	0.98469	0.11636	-0.46647
19	0.99692	0.03911	-0.15678
20	0.99692	-0.03911	0.15678
21	0.98469	-0.11636	0.46647
22	0.96052	-0.19075	0.76468
23	0.92501	-0.26044	1.04407
24	0.87903	-0.32372	1.29778
25	0.82372	-0.37903	1.51956
26	0.76044	-0.42501	1.70398
27	0.69075	-0.46052	1.84660
28	0.61636	-0.48469	1.94391
29	0.53911	-0.49692	1.99819
30	0.45659	-0.49620	1.98401
31	0.37108	-0.48113	1.92846
32	0.28949	-0.45143	1.80976
33	0.21430	-0.40802	1.63591
34	0.14779	-0.35221	1.41223
35	0.09198	-0.28570	1.14558
36	0.04857	-0.21050	0.84409
37	0.01887	-0.12892	0.51694
38	0.00380	-0.04341	0.17408
39	1.00003	0.0	-0.00000

TRAJECTORY STARTING POSITION IS X= -5.00 YO= 0.40700

STEP	TIME	DTS	XDS	YDS	PSI	UAS	UDS	VAS	VDS	RED	ACCN/MOD	HIST/RHS	USTAB	VSTAB
2	0.06	0.1080	-4.94048	0.40707	0.40366	0.99174	0.99192	0.00124	0.00120	0.10834	0.00000	0.0		
3	0.17	0.1944	-4.83336	0.40720	0.40363	0.99141	0.99192	0.00132	0.00120	0.30974	0.00000	0.0		
4	0.36	0.3499	-4.64053	0.40744	0.40348	0.99077	0.99191	0.00147	0.00121	0.70110	0.00000	0.0		
5	0.71	0.5161	-4.29345	0.40786	0.40300	0.98941	0.99187	0.00181	0.00121	1.51575	0.00000	0.0		
6	1.23	0.6160	-3.78156	0.40849	0.40183	0.98681	0.99174	0.00254	0.00125	3.06155	0.00000	0.0		
7	1.84	0.5682	-3.17072	0.40929	0.39972	0.98223	0.99140	0.00401	0.00134	5.73218	0.00000	0.0		
8	2.41	0.5974	-2.60755	0.41010	0.39542	0.97557	0.99076	0.00656	0.00154	9.59751	0.00000	0.0		
9	3.01	0.5889	-2.01605	0.41114	0.38640	0.96375	0.98946	0.01217	0.00201	16.58722	0.00000	0.0		
10	3.60	0.4623	-1.43403	0.41262	0.37090	0.94199	0.98684	0.02593	0.00320	30.17147	0.00000	0.0		
11	4.06	0.2998	-0.97867	0.41459	0.35116	0.90999	0.98256	0.05477	0.00573	52.54998	0.00002	0.0		
12	4.36	0.2366	-0.68478	0.41682	0.32385	0.87717	0.97743	0.09862	0.00973	80.39044	0.00004	0.0		
13	4.60	0.1772	-0.45424	0.41983	0.29071	0.84551	0.97091	0.16853	0.01651	118.23268	0.00009	0.0		
14	4.77	0.1310	-0.28281	0.42355	0.25485	0.82837	0.96402	0.26251	0.02651	163.31494	0.00020	0.0		
15	4.91	0.1196	-0.15692	0.42780	0.21027	0.83644	0.95804	0.36970	0.03938	211.18507	0.00034	0.0		
16	5.03	0.0846	-0.04263	0.43355	0.17077	0.88622	0.95325	0.50224	0.05820	269.43021	0.00072	0.0		
17	5.11	0.0772	0.03789	0.43924	0.12934	0.96888	0.95214	0.61034	0.07743	319.88825	0.00104	0.0		
18	5.19	0.0575	0.11153	0.44607	0.09650	1.09732	0.95518	0.70329	0.10023	371.73509	0.00178	0.0		
20	5.28	0.0246	0.20179	0.45703	0.06219	1.33382	0.96871	0.76342	0.13400	436.57149	0.00527	0.0		
23	5.34	0.0166	0.26306	0.46619	0.03511	1.53938	0.98686	0.72567	0.15802	475.26932	0.00862	0.0		
29	5.41	0.0044	0.32795	0.47719	0.01462	1.78278	1.01427	0.63857	0.18036	536.82450	0.03760	0.0		
39	5.46	0.0079	0.37972	0.48659	0.00501	1.83917	1.04124	0.46137	0.19194	505.28922	0.01786	0.0		
78	5.52	0.0040	0.43886	0.49756	0.00659	1.94067	1.07571	0.20471	0.19835	518.96558	0.03485	0.0		
86	5.56	0.0051	0.49214	0.50723	0.01875	1.98619	1.10427	0.06653	0.19658	534.84820	0.02742	0.0		
96	5.61	0.0083	0.54719	0.51670	0.04404	1.90441	1.13258	-0.16026	0.18761	507.935'9	0.01477	0.0		
101	5.65	0.0084	0.59496	0.52431	0.07079	1.82285	1.15305	-0.31023	0.17586	496.53490	0.01354	0.0		
102	5.66	0.0087	0.60464	0.52577										

CLOSEST APPROACH IS Y= 0.00008 NO. OF STEPS REQUIRED=102 PSI= 0.071

TRAJECTORY STARTING POSITION IS X= -5.00 YO= 0.40692

STEP	TIME	DTS	XDS	YDS	PSI	UAS	UDS	VAS	VDS	RED	ACCN/MOD	HIST/RHS	USTAB	VSTAB
2	0.06	0.1080	-4.94048	0.40699	0.40358	0.99174	0.99192	0.00124	0.00120	0.10834	0.00000	0.0		
3	0.17	0.1944	-4.83336	0.40712	0.40355	0.99141	0.99192	0.00132	0.00120	0.30974	0.00000	0.0		
4	0.36	0.3499	-4.64053	0.40735	0.40340	0.99077	0.99191	0.00147	0.00121	0.70110	0.00000	0.0		
5	0.71	0.5161	-4.29345	0.40778	0.40292	0.98941	0.99187	0.00181	0.00121	1.51576	0.00000	0.0		
6	1.23	0.6160	-3.78156	0.40841	0.40175	0.98681	0.99174	0.00254	0.00125	3.06156	0.00000	0.0		
7	1.84	0.5682	-3.17073	0.40920	0.39964	0.98223	0.99140	0.00401	0.00134	5.73218	0.00000	0.0		
8	2.41	0.5974	-2.60756	0.41002	0.39534	0.97557	0.99076	0.00656	0.00154	9.59750	0.00000	0.0		
9	3.01	0.5889	-2.01606	0.41106	0.38632	0.96375	0.98946	0.01217	0.00201	16.58716	0.00000	0.0		
10	3.60	0.4623	-1.43405	0.41253	0.37083	0.94198	0.98684	0.02593	0.00320	30.17128	0.00000	0.0		
11	4.06	0.2998	-0.97869	0.41450	0.35109	0.90998	0.98256	0.05476	0.00573	52.55051	0.00002	0.0		
12	4.36	0.2366	-0.68481	0.41674	0.32378	0.87715	0.97743	0.09860	0.00973	80.39062	0.00004	0.0		
13	4.60	0.1772	-0.45428	0.41975	0.29064	0.84548	0.97091	0.16850	0.01651	118.23119	0.00009	0.0		
14	4.77	0.1310	-0.28283	0.42347	0.25479	0.82832	0.96402	0.26249	0.02651	163.32065	0.00020	0.0		
15	4.91	0.1196	-0.15696	0.42771	0.21021	0.83635	0.95803	0.36968	0.03937	211.19303	0.00034	0.0		
16	5.03	0.0845	-0.04268	0.43346	0.17072	0.88608	0.95324	0.50224	0.05819	269.44352	0.00072	0.0		
17	5.11	0.0772	0.03784	0.43916	0.12928	0.96869	0.95212	0.61038	0.07743	319.90951	0.00104	0.0		

18	5.19	0.0576	0.11147	0.44598	0.09641	1.09709	0.95515	0.70340	0.10023	371.76847	0.00178	0.0
20	5.28	0.0246	0.20178	0.45695	0.06212	1.33375	0.96869	0.76365	0.13402	436.66320	0.00528	0.0
23	5.34	0.0167	0.26304	0.46611	0.03500	1.53940	0.98683	0.72594	0.15806	475.38563	0.00862	0.0
29	5.41	0.0044	0.32792	0.47711	0.01450	1.78308	1.01425	0.63926	0.18041	537.18119	0.03760	0.0
39	5.46	0.0079	0.37959	0.48650	0.00489	1.83901	1.04118	0.46170	0.19198	505.29246	0.01788	0.0
82	5.52	0.0041	0.43959	0.49762	0.00660	1.94009	1.07614	0.20299	0.19842	518.35470	0.03386	0.0
90	5.56	0.0054	0.49215	0.50716	0.01873	1.98654	1.10431	0.06672	0.19665	535.01908	0.02598	0.0
100	5.62	0.0084	0.55014	0.51713	0.04547	1.90153	1.13398	-0.16891	0.18706	507.62302	0.01453	0.0
105	5.66	0.0084	0.59813	0.52474	0.07265	1.81539	1.15431	-0.31826	0.17502	494.87569	0.01336	0.0
106	5.67	0.0089	0.60788	0.52620								

CLOSEST APPROACH IS Y= 0.00001 NO. OF STEPS REQUIRED=106 PSI= 0.073

TRAJECTORY STARTING POSITION IS X= -5.00 YO= 0.40690

STEP	TIME	DTS	XDS	YDS	PSI	UAS	UDS	VAS	VDS	RED	ACCN/MOD	HIST/RHS	USTAB	VSTAB
2	0.06	0.1080	-4.94048	0.40698	0.40357	0.99174	0.99192	0.00124	0.00120	0.10834	0.00000	0.0		
3	0.17	0.1944	-4.83336	0.40711	0.40353	0.99141	0.99192	0.00132	0.00120	0.30974	0.00000	0.0		
4	0.36	0.3499	-4.64053	0.40734	0.40338	0.99077	0.99191	0.00147	0.00121	0.70110	0.00000	0.0		
5	0.71	0.5161	-4.29345	0.40776	0.40291	0.98941	0.99187	0.00181	0.00121	1.51576	0.00000	0.0		
6	1.23	0.6160	-3.78156	0.40840	0.40173	0.98681	0.99174	0.00254	0.00125	3.06156	0.00000	0.0		
7	1.84	0.5682	-3.17073	0.40919	0.39963	0.98223	0.99140	0.00401	0.00134	5.73218	0.00000	0.0		
8	2.41	0.5974	-2.60756	0.41000	0.39533	0.97557	0.99076	0.00656	0.00154	9.59750	0.00000	0.0		
9	3.01	0.5889	-2.01607	0.41104	0.38630	0.96375	0.98946	0.01217	0.00201	16.58715	0.00000	0.0		
10	3.60	0.4623	-1.43405	0.41252	0.37081	0.94198	0.98684	0.02593	0.00320	30.17124	0.00000	0.0		
11	4.06	0.2998	-0.97869	0.41449	0.35108	0.90998	0.98256	0.05476	0.00573	52.55060	0.00002	0.0		
12	4.36	0.2366	-0.68481	0.41673	0.32377	0.87715	0.97743	0.09860	0.00973	80.39064	0.00004	0.0		
13	4.60	0.1772	-0.45429	0.41973	0.29063	0.84548	0.97091	0.16849	0.01651	118.23090	0.00009	0.0		
14	4.77	0.1310	-0.28284	0.42345	0.25478	0.82830	0.96402	0.26248	0.02651	163.32170	0.00020	0.0		
15	4.91	0.1196	-0.15696	0.42770	0.21020	0.83633	0.95803	0.36967	0.03937	211.19449	0.00034	0.0		
16	5.03	0.0845	-0.04269	0.43345	0.17071	0.88605	0.95324	0.50223	0.05819	269.44597	0.00072	0.0		
17	5.11	0.0772	0.03783	0.43914	0.12927	0.96866	0.95211	0.61038	0.07743	319.91342	0.00104	0.0		
18	5.19	0.0576	0.11146	0.44597	0.09639	1.09705	0.95515	0.70341	0.10023	371.77462	0.00178	0.0		
20	5.28	0.0246	0.20178	0.45694	0.06210	1.33374	0.96868	0.76370	0.13403	436.68007	0.00528	0.0		
23	5.34	0.0167	0.26303	0.46610	0.03498	1.53940	0.98683	0.72599	0.15807	475.40724	0.00862	0.0		
29	5.41	0.0044	0.32792	0.47709	0.01447	1.78314	1.01424	0.63938	0.18042	537.24778	0.03760	0.0		
39	5.46	0.0079	0.37956	0.48648	0.00487	1.83899	1.04117	0.46176	0.19199	505.29383	0.01789	0.0		
43	5.48	0.0024	0.40621	0.49142	0.00294	1.93025	1.05555	0.44102	0.19640	544.93309	0.06463	0.0		
44	5.49	0.0022	0.40878	0.49189										

COLLISION COORDS: X= 0.4087804 Y= 0.4916086 L= 0.6936905 NO. OF STEPS REQUIRED= 44

TRAJECTORY STARTING POSITION IS X= -5.00 YO= 0.40650

STEP	TIME	DTS	XDS	YDS	PSI	UAS	UDS	VAS	VDS	RED	ACCN/MOD	HIST/RHS	USTAB	VSTAB
2	0.06	0.1080	-4.94048	0.40657	0.40317	0.99174	0.99192	0.00124	0.00120	0.10834	0.00000	0.0		
3	0.17	0.1944	-4.83336	0.40670	0.40313	0.99141	0.99192	0.00132	0.00120	0.30975	0.00000	0.0		
4	0.36	0.3499	-4.64053	0.40694	0.40298	0.99077	0.99191	0.00147	0.00120	0.70112	0.00000	0.0		
5	0.71	0.5161	-4.29345	0.40736	0.40251	0.98941	0.99187	0.00181	0.00121	1.51579	0.00000	0.0		
6	1.23	0.6160	-3.78157	0.40799	0.40134	0.98681	0.99174	0.00254	0.00125	3.06159	0.00000	0.0		
7	1.84	0.5682	-3.17075	0.40878	0.39923	0.98223	0.99140	0.00401	0.00134	5.73219	0.00000	0.0		
8	2.41	0.5973	-2.60761	0.40959	0.39493	0.97557	0.99076	0.00655	0.00154	9.59744	0.00000	0.0		
9	3.01	0.5889	-2.01614	0.41063	0.38592	0.96375	0.98946	0.01216	0.00201	16.58688	0.00000	0.0		
10	3.60	0.4624	-1.43416	0.41211	0.37044	0.94197	0.98684	0.02590	0.00319	30.17029	0.00000	0.0		
11	4.06	0.2997	-0.97876	0.41408	0.35073	0.90994	0.98255	0.05472	0.00573	52.55323	0.00002	0.0		
12	4.36	0.2365	-0.68494	0.41631	0.32345	0.87708	0.97742	0.09852	0.00972	80.39154	0.00004	0.0		
13	4.60	0.1773	-0.45450	0.41931	0.29032	0.84533	0.97091	0.16834	0.01649	118.22351	0.00009	0.0		
14	4.77	0.1309	-0.28297	0.42303	0.25449	0.82802	0.96400	0.26237	0.02649	163.35026	0.00020	0.0		
15	4.91	0.1196	-0.15714	0.42727	0.20994	0.83587	0.95800	0.36956	0.03935	211.23430	0.00034	0.0		
16	5.03	0.0845	-0.04291	0.43302	0.17044	0.88534	0.95318	0.50222	0.05816	269.51244	0.00072	0.0		
17	5.11	0.0772	0.03758	0.43871	0.12899	0.96771	0.95202	0.61056	0.07739	320.01936	0.00104	0.0		
18	5.19	0.0578	0.11118	0.44553	0.09592	1.09590	0.95502	0.70392	0.10021	371.94083	0.00177	0.0		
20	5.28	0.0243	0.20173	0.45654	0.06174	1.33335	0.96856	0.76485	0.13417	437.12689	0.00536	0.0		
23	5.35	0.0181	0.26655	0.46628	0.03219	1.54963	0.98802	0.72176	0.15959	476.75558	0.00793	0.0		
31	5.42	0.0034	0.33606	0.47819	0.01198	1.80826	1.01861	0.58066	0.18313	530.41222	0.04695	0.0		
40	5.46	0.0079	0.38339	0.48684	0.00383	1.84628	1.04317	0.45726	0.19297	507.26400	0.01799	0.0		
41	5.47	0.0077	0.39166	0.48837										

COLLISION COORDS: X= 0.3906382 Y= 0.4878934 L= 0.6751708 NO. OF STEPS REQUIRED= 41

TRAJECTORY STARTING POSITION IS X= -5.00 YO= 0.39860

STEP	TIME	DTS	XDS	YDS	PSI	UAS	UDS	VAS	VDS	RED	ACCN/MOD	HIST/RHS	USTAB	VSTAB
2	0.06	0.1080	-4.94049	0.39867	0.39533	0.99174	0.99191	0.00122	0.00118	0.10837	0.00000	0.0		
3	0.17	0.1944	-4.83336	0.39880	0.39530	0.99141	0.99191	0.00129	0.00118	0.30985	0.00000	0.0		
4	0.36	0.3499	-4.64053	0.39903	0.39515	0.99076	0.99190	0.00144	0.00118	0.70135	0.00000	0.0		
5	0.71	0.5159	-4.29345	0.39944	0.39469	0.98941	0.99186	0.00178	0.00119	1.51633	0.00000	0.0		
6	1.23	0.6157	-3.78177	0.40006	0.39353	0.98679	0.99174	0.00249	0.00122	3.06220	0.00000	0.0		
7	1.84	0.5679	-3.17129	0.40084	0.39147	0.98221	0.99140	0.00393	0.00131	5.73231	0.00000	0.0		
8	2.41	0.5968	-2.60848	0.40163	0.38726	0.97553	0.99075	0.00643	0.00151	9.59651	0.00000	0.0		
9	3.01	0.5882	-2.01751	0.40265	0.37843	0.96367	0.98946	0.01192	0.00197	16.58181	0.00000	0.0		
10	3.60	0.4633	-1.43618	0.40410	0.36318	0.94178	0.98683	0.02541	0.00313	30.15217	0.00000	0.0		
11	4.06	0.2985	-0.97984	0.40603	0.34387	0.90924	0.98250	0.05385	0.00563	52.62109	0.00002	0.0		
12	4.36	0.2350	-0.68730	0.40822	0.31720	0.87568	0.97734	0.09693	0.00954	80.42689	0.00004	0.0		
13	4.59	0.1789	-0.45837	0.41114	0.28415	0.84253	0.97075	0.16555	0.01616	118.11761	0.00009	0.0		
14	4.77	0.1300	-0.28527	0.41483	0.24880	0.82241	0.96357	0.26018	0.02614	163.97835	0.00021	0.0		
15	4.90	0.1188	-0.16040	0.41899	0.20464	0.82691	0.95730	0.36744	0.03886	212.09573	0.00034	0.0		
16	5.02	0.0840	-0.04703	0.42463	0.16522	0.87143	0.95199	0.50184	0.05755	270.90499	0.00073	0.0		
17	5.11	0.0768	0.03284	0.43023	0.12349	0.94927	0.95027	0.61380	0.07682	322.17732	0.00107	0.0		
19	5.23	0.0358	0.15352	0.44241	0.07321	1.18894	0.95745	0.76378	0.11772	411.75067	0.00337	0.0		
22	5.31	0.0153	0.23201	0.45328	0.03874	1.43228	0.97438	0.79274	0.15001	473.46729	0.00957	0.0		
27	5.38	0.0125	0.29798	0.46420	0.00945	1.63349	0.99896	0.70691	0.17571	496.49409	0.01200	0.0		
32	5.41	0.0010	0.32607	0.46925	0.00260	1.76769	1.01193	0.75359	0.18592	567.09926	0.17484	0.0		
33	5.41	0.0010	0.32713	0.46944										

COLLISION COORDS: X= 0.3271172 Y= 0.4691605 L= 0.6088966 NO. OF STEPS REQUIRED= 33

TRAJECTORY STARTING POSITION IS X= -5.00 YO= 0.37369

STEP	TIME	DTS	XDS	YDS	PSI	UAS	UDS	VAS	VDS	RED	ACCN/MOD	HIST/RHS	USTAB	VSTAB
2	0.06	0.1080	-4.94049	0.37375	0.37062	0.99172	0.99190	0.00114	0.00111	0.10848	0.00000	0.0		
3	0.17	0.1944	-4.83336	0.37387	0.37059	0.99139	0.99190	0.00121	0.00111	0.31015	0.00000	0.0		
4	0.36	0.3499	-4.64054	0.37409	0.37045	0.99074	0.99189	0.00135	0.00111	0.70205	0.00000	0.0		
5	0.71	0.5153	-4.29346	0.37448	0.37002	0.98938	0.99185	0.00167	0.00112	1.51798	0.00000	0.0		
6	1.23	0.6146	-3.78236	0.37506	0.36894	0.98675	0.99172	0.00234	0.00115	3.06404	0.00000	0.0		
7	1.84	0.5668	-3.17290	0.37579	0.36700	0.98214	0.99138	0.00369	0.00123	5.73272	0.00000	0.0		
8	2.41	0.5953	-2.61111	0.37653	0.36306	0.97542	0.99073	0.00604	0.00142	9.59375	0.00000	0.0		
9	3.00	0.5863	-2.02164	0.37749	0.35479	0.96345	0.98943	0.01119	0.00185	16.56669	0.00000	0.0		
10	3.59	0.4669	-1.44225	0.37884	0.34028	0.94119	0.98679	0.02385	0.00293	30.09834	0.00000	0.0		
11	4.06	0.2946	-0.98239	0.38067	0.32218	0.90704	0.98234	0.05111	0.00531	52.87910	0.00002	0.0		
12	4.35	0.2309	-0.69367	0.38270	0.29732	0.87133	0.97706	0.09194	0.00898	80.62649	0.00004	0.0		
13	4.58	0.1851	-0.46883	0.38540	0.26437	0.83388	0.97026	0.15697	0.01516	118.04472	0.00009	0.0		
14	4.77	0.1270	-0.28996	0.38903	0.23049	0.80462	0.96214	0.25397	0.02516	166.66587	0.00022	0.0		
15	4.90	0.1039	-0.16825	0.39294	0.19277	0.79896	0.95502	0.36123	0.03746	215.64175	0.00041	0.0		
16	5.00	0.0947	-0.06936	0.39760	0.14852	0.82251	0.94890	0.48360	0.05345	268.98880	0.00065	0.0		
17	5.09	0.0856	0.02024	0.40363	0.09994	0.89175	0.94473	0.62263	0.07517	329.99604	0.00101	0.0		
19	5.21	0.0306	0.13153	0.41469	0.06031	1.09396	0.94715	0.80219	0.11523	421.46840	0.00420	0.0		
22	5.28	0.0189	0.20099	0.42424	0.02130	1.30710	0.95839	0.85747	0.14733	474.66329	0.00804	0.0		
29	5.33	0.0004	0.24719	0.43184	0.00255	1.46229	0.97154	0.97069	0.17051	563.18460	0.53418	0.0		
30	5.33	0.0005	0.24755	0.43190										

COLLISION COORDS: X= 0.2476601 Y= 0.4316533 L= 0.5209184 NO. OF STEPS REQUIRED= 30

TRAJECTORY STARTING POSITION IS X= -5.00 YO= 0.33217

STEP	TIME	DTS	XDS	YDS	PSI	UAS	UDS	VAS	VDS	RED	ACCN/MOD	HIST/RHS	USTAB	VSTAB
2	0.06	0.1080	-4.94049	0.33223	0.32944	0.99170	0.99187	0.00102	0.00099	0.10864	0.00000	0.0		
3	0.17	0.1944	-4.83337	0.33233	0.32941	0.99136	0.99187	0.00108	0.00099	0.31061	0.00000	0.0		
4	0.36	0.3499	-4.64055	0.33252	0.32929	0.99071	0.99186	0.00121	0.00099	0.70313	0.00000	0.0		
5	0.71	0.5145	-4.29348	0.33287	0.32890	0.98934	0.99182	0.00149	0.00099	1.52050	0.00000	0.0		
6	1.23	0.6131	-3.78325	0.33339	0.32794	0.98670	0.99170	0.00208	0.00102	3.06686	0.00000	0.0		
7	1.84	0.5653	-3.17535	0.33403	0.32622	0.98205	0.99135	0.00329	0.00110	5.73343	0.00000	0.0		
8	2.41	0.5931	-2.61510	0.33470	0.32272	0.97526	0.99070	0.00538	0.00126	9.58974	0.00000	0.0		
9	3.00	0.5834	-2.02787	0.33554	0.31539	0.96310	0.98939	0.00997	0.00165	16.54437	0.00000	0.0		
10	3.58	0.4750	-1.45139	0.33674	0.30209	0.94029	0.98673	0.02124	0.00261	30.01918	0.00000	0.0		
11	4.06	0.2887	-0.98370	0.33840	0.28593	0.90331	0.98205	0.04652	0.00479	53.46506	0.00002	0.0		
12	4.35	0.2259	-0.70087	0.34020	0.26374	0.86419	0.97657	0.08365	0.00808	81.25274	0.00004	0.0		
13	4.57	0.1768	-0.48100	0.34257	0.23502	0.81989	0.96937	0.14324	0.01361	118.71162	0.00009	0.0		
14	4.75	0.1311	-0.31038	0.34566	0.20250	0.77842	0.96066	0.23101	0.02233	166.22120	0.00021	0.0		
15	4.88	0.1002	-0.18497	0.34930	0.16777	0.75317	0.95174	0.34011	0.03418	218.82518	0.00043	0.0		
16	4.98	0.0915	-0.08998	0.35341	0.12592	0.75209	0.94357	0.46307	0.04903	273.69081	0.00070	0.0		
17	5.07	0.0603	-0.00404	0.35878	0.09199	0.78741	0.93602	0.61316	0.06982	337.96534	0.00152	0.0		
19	5.17	0.0289	0.08889	0.36733	0.04743	0.90933	0.93090	0.81276	0.10429	425.25790	0.00467	0.0		
23	5.25	0.0068	0.16403	0.37725	0.00500	1.07507	0.93419	0.97649	0.14335	506.95171	0.02650	0.0		
26	5.26	0.0014	0.17197	0.37849	0.00249	1.08304	0.93504	1.01900	0.14830	529.88968	0.13443	0.0		
27	5.26	0.0016	0.17331	0.37870										

COLLISION COORDS: X= 0.1732322 Y= 0.3784479 L= 0.4293008 NO. OF STEPS REQUIRED= 27

TRAJECTORY STARTING POSITION IS X= -5.00 YO= 0.27404

STEP	TIME	DTS	XDS	YDS	PSI	UAS	UDS	VAS	VDS	RED	ACCN/MOD	HIST/RHS	USTAB	VSTAB
2	0.06	0.1080	-4.94049	0.27409	0.27178	0.99167	0.99185	0.00084	0.00081	0.10883	0.00000	0.0		
3	0.17	0.1944	-4.83337	0.27417	0.27176	0.99133	0.99184	0.00089	0.00082	0.31116	0.00000	0.0		
4	0.36	0.3499	-4.64056	0.27433	0.27166	0.99067	0.99183	0.00100	0.00082	0.70443	0.00000	0.0		
5	0.71	0.5134	-4.29350	0.27462	0.27134	0.98929	0.99180	0.00123	0.00082	1.52354	0.00000	0.0		
6	1.23	0.6112	-3.78431	0.27505	0.27055	0.98663	0.99167	0.00172	0.00084	3.07029	0.00000	0.0		
7	1.84	0.5635	-3.17826	0.27558	0.26910	0.98194	0.99132	0.00272	0.00091	5.73441	0.00000	0.0		
8	2.40	0.5904	-2.61985	0.27612	0.26624	0.97506	0.99067	0.00445	0.00104	9.58524	0.00000	0.0		
9	2.99	0.5799	-2.03528	0.27682	0.26022	0.96269	0.98935	0.00824	0.00136	16.51864	0.00000	0.0		
10	3.57	0.4840	-1.46220	0.27780	0.24835	0.93921	0.98665	0.01756	0.00215	29.92838	0.00000	0.0		
11	4.05	0.2856	-0.98565	0.27921	0.23521	0.89867	0.98169	0.03948	0.00402	54.15967	0.00002	0.0		
12	4.34	0.2218	-0.70600	0.28071	0.21646	0.85459	0.97587	0.07156	0.00682	82.48561	0.00004	0.0		
13	4.56	0.1673	-0.49033	0.28268	0.19239	0.80103	0.96806	0.12364	0.01153	120.69358	0.00010	0.0		
14	4.73	0.1528	-0.32914	0.28514	0.15855	0.74515	0.95857	0.19875	0.01870	167.52814	0.00018	0.0		
15	4.88	0.0917	-0.18358	0.28887	0.12813	0.68555	0.94540	0.32347	0.03158	234.46619	0.00054	0.0		
16	4.97	0.0841	-0.09735	0.29234	0.09072	0.65532	0.93468	0.44351	0.04515	291.91448	0.00086	0.0		
18	5.10	0.0411	0.02149	0.30005	0.04045	0.66113	0.91646	0.70127	0.07869	403.72246	0.00312	0.0		
25	5.20	0.0013	0.10637	0.30922	0.00211	0.71762	0.90413	0.94695	0.12084	508.11663	0.14815	0.0		
26	5.20	0.0020	0.10754	0.30937										

COLLISION COORDS: X= 0.1069864 Y= 0.3090961 L= 0.3332461 NO. OF STEPS REQUIRED= 26

TRAJECTORY STARTING POSITION IS X= -5.00 YO= 0.19930

STEP	TIME	DTS	XDS	YDS	PSI	UAS	UDS	VAS	VDS	RED	ACCN/MOD	HIST/RHS	USTAB	VSTAB
2	0.06	0.1080	-4.94049	0.19934	0.19766	0.99164	0.99182	0.00061	0.00059	0.10902	0.00000	0.0		
3	0.17	0.1944	-4.83337	0.19940	0.19764	0.99130	0.99181	0.00065	0.00059	0.31172	0.00000	0.0		
4	0.36	0.3499	-4.64057	0.19952	0.19757	0.99064	0.99181	0.00073	0.00060	0.70574	0.00000	0.0		
5	0.71	0.5124	-4.29352	0.19972	0.19733	0.98924	0.99177	0.00090	0.00060	1.52660	0.00000	0.0		
6	1.22	0.6094	-3.78537	0.20004	0.19676	0.98655	0.99164	0.00126	0.00062	3.07378	0.00000	0.0		
7	1.83	0.5616	-3.18115	0.20042	0.19573	0.98182	0.99129	0.00199	0.00066	5.73553	0.00000	0.0		
8	2.40	0.5878	-2.62457	0.20082	0.19363	0.97486	0.99063	0.00325	0.00076	9.58106	0.00000	0.0		
9	2.98	0.5766	-2.04263	0.20132	0.18927	0.96227	0.98930	0.00601	0.00099	16.49397	0.00000	0.0		
10	3.56	0.4673	-1.47287	0.20203	0.18136	0.93813	0.98658	0.01281	0.00156	29.84171	0.00000	0.0		
11	4.03	0.3061	-1.01284	0.20301	0.17066	0.89712	0.98172	0.02811	0.00285	52.97420	0.00002	0.0		
12	4.33	0.2212	-0.71325	0.20417	0.15646	0.84488	0.97521	0.05372	0.00505	83.46889	0.00004	0.0		
13	4.55	0.1639	-0.49842	0.20564	0.13873	0.78034	0.96661	0.09447	0.00864	123.04846	0.00010	0.0		
14	4.72	0.1298	-0.34083	0.20745	0.11651	0.70674	0.95589	0.15406	0.01413	171.44332	0.00022	0.0		
15	4.85	0.1185	-0.21752	0.20977	0.08517	0.62541	0.94256	0.23945	0.02243	230.56234	0.00041	0.0		
16	4.97	0.0598	-0.10687	0.21318	0.06313	0.53141	0.92413	0.37548	0.03666	311.18950	0.00137	0.0		
18	5.07	0.0343	-0.01071	0.21817	0.02415	0.44399	0.90017	0.58226	0.06053	415.80209	0.00409	0.0		
22	5.14	0.0025	0.05207	0.22331	0.00084	0.40174	0.87958	0.75434	0.08705	492.41797	0.07811	0.0		
23	5.15	0.0040	0.05427	0.22353										

COLLISION COORDS: X= 0.0525900 Y= 0.2232139 L= 0.2314103 NO. OF STEPS REQUIRED= 23

TRAJECTORY STARTING POSITION IS X= -5.00 YO= 0.10795

STEP	TIME	DTS	XDS	YDS	PSI	UAS	UDS	VAS	VDS	RED	ACCN/MOD	HIST/RHS	USTAB	VSTAB
2	0.06	0.1080	-4.94049	0.10797	0.10706	0.99161	0.99179	0.00033	0.00032	0.10918	0.00000	0.0		
3	0.17	0.1944	-4.83338	0.10801	0.10705	0.99127	0.99179	0.00035	0.00032	0.31216	0.00000	0.0		
4	0.36	0.3499	-4.64058	0.10807	0.10701	0.99061	0.99178	0.00040	0.00032	0.70678	0.00000	0.0		
5	0.71	0.5116	-4.29354	0.10818	0.10689	0.98920	0.99174	0.00049	0.00033	1.52904	0.00000	0.0		
6	1.22	0.6080	-3.78620	0.10835	0.10658	0.98650	0.99161	0.00068	0.00033	3.07658	0.00000	0.0		
7	1.83	0.5602	-3.18343	0.10856	0.10602	0.98173	0.99126	0.00108	0.00036	5.73652	0.00000	0.0		
8	2.39	0.5857	-2.62828	0.10878	0.10488	0.97470	0.99060	0.00176	0.00041	9.57799	0.00000	0.0		
9	2.98	0.5740	-2.04839	0.10905	0.10252	0.96194	0.98927	0.00326	0.00054	16.47521	0.00000	0.0		
10	3.55	0.4524	-1.48121	0.10943	0.09842	0.93726	0.98652	0.00695	0.00085	29.77600	0.00000	0.0		
11	4.00	0.2940	-1.03589	0.10994	0.09303	0.89626	0.98178	0.01493	0.00151	51.93415	0.00002	0.0		
12	4.30	0.2259	-0.74806	0.11052	0.08546	0.84418	0.97556	0.02781	0.00260	80.26203	0.00004	0.0		
13	4.52	0.1754	-0.52863	0.11129	0.07522	0.77326	0.96674	0.04964	0.00448	119.20374	0.00009	0.0		
14	4.70	0.1369	-0.35998	0.11232	0.06197	0.68076	0.95459	0.08483	0.00762	170.69586	0.00021	0.0		
15	4.84	0.1073	-0.23031	0.11367	0.04563	0.56671	0.93855	0.13820	0.01260	235.47969	0.00047	0.0		
16	4.94	0.0661	-0.13059	0.11538	0.03126	0.43462	0.91836	0.21404	0.02007	312.69240	0.00127	0.0		
18	5.04	0.0278	-0.04022	0.11797	0.01338	0.26180	0.88879	0.33976	0.03322	418.72403	0.00526	0.0		
23	5.11	0.0028	0.01358	0.12041	-0.00002	0.15997	0.86289	0.46208	0.04713	489.73179	0.07123	0.0		
24	5.11	0.0045	0.01603	0.12054										

COLLISION COORDS: X= 0.0146779 Y= 0.1203817 L= 0.1215945 NO. OF STEPS REQUIRED= 24

BETAO (MAX LOCAL CE) IS 89.8% AT A DISTANCE OF 0.0 FROM THE NOSE
THE TOTAL COLLISION EFFICIENCY IS 81.4%

ENDPT. X COORD.	Y COORD.	DIST. FROM NOSE	COLL. EFF.
-0.0	-0.0	0.0	0.8980
0.00048	0.02181	0.02181	0.8970
0.00190	0.04358	0.04365	0.8941
0.00428	0.06526	0.06547	0.8891
0.00760	0.08682	0.08729	0.8823
0.01185	0.10822	0.10911	0.8734
0.01704	0.12941	0.13093	0.8626
0.02314	0.15035	0.15274	0.8498
0.03015	0.17101	0.17456	0.8352
0.03806	0.19134	0.19638	0.8187
0.04685	0.21131	0.21819	0.8002
0.05649	0.23087	0.24001	0.7799
0.06699	0.25000	0.26183	0.7578
0.07830	0.26865	0.28364	0.7340
0.09042	0.28679	0.30546	0.7084
0.10332	0.30438	0.32727	0.6811
0.11698	0.32139	0.34909	0.6521
0.13136	0.33780	0.37091	0.6215
0.14645	0.35355	0.39272	0.5894
0.16220	0.36864	0.41454	0.5557
0.17861	0.38302	0.43636	0.5204
0.19562	0.39668	0.45817	0.4837
0.21321	0.40958	0.47999	0.4456
0.23135	0.42170	0.50181	0.4061
0.25000	0.43301	0.52362	0.3652
0.26913	0.44351	0.54544	0.3229
0.28869	0.45315	0.56726	0.2793
0.30866	0.46194	0.58907	0.2344
0.32899	0.46985	0.61089	0.1881
0.34965	0.47686	0.63271	0.1405
0.37059	0.48296	0.65452	0.0915
0.39178	0.48815	0.67634	0.0411
0.41318	0.49240	0.69816	0.0

THE ACCRETED AREA FOR LAYER 1 IS 0.09271
THE ACCUMULATED ACCRETED AREA IS 0.09271

APPENDIX C: PROGRAM LISTING

This appendix contains the program listing as written in Fortran. The program listing has been carefully annotated. However, should difficulties be encountered in attempting to run the program as listed, the authors are prepared to offer advice and assistance.

```
C
C WRITTEN BY: M. OLESKIW  ON:790526  LAST MODIFIED:801228
C
C CALCULATE POTENTIAL FLOW ABOUT AN ARBITRARILY SHAPED AEROFOIL;
C   CALCULATE A SERIES OF DROPLET TRAJECTORIES AND
C   DETERMINE THE COLLISION LOCATIONS; FIND THE RESULTING COLLISION
C   EFFICIENCY AND ACCRETE A LAYER OF ICE.
C   REPEAT THE PROCESS FOR A PREDETERMINED NUMBER OF STEPS.
C
C INTERNAL SUBROUTINES:
C    COORDS: CALCULATE THE UPPER AND LOWER SFC. COORDINATES
C      OF THE AEROFOIL.
C    POT1: SOLVE FOR SFC. VORTEX DENSITY ON 1 ELEMENT AEROFOIL
C      IN POTENTIAL FLOW, GIVEN COORDINATES OF AEROFOIL SFC.
C    STRMFN: CALCULATE STREAMFUNCTION ON A GRID ABOUT AN AEROFOIL
C      SECTION GIVEN THE SFC. VORTICITY DENSITY ON THE AEROFOIL
C      AND PLOT THE FLOW USING VELOCITY VECTORS.
C    AIRPLT: PLOTS AEROFOIL OUTLINE WITHIN WINDOW
C    SFC:CALCULATE Y VALUES AND THE LENGTH FROM THE NOSE ON THE
C      SFC. OF THE AEROFOIL BY A CUBIC SPLINE INTERPOLATION.
C    SFCLEN:CALCULATES THE LENGTH ALONG A SEGMENT OF A CUBIC SPLINE.
C    CE:CALCULATE AND PLOT COLLISION EFFICIENCY OF ARBITRARY
C      AEROFOIL BY DETERMINING A SET OF IMPACTING TRAJECTORIES.
C    PLTSZ:DETERMINE PARAMETERS NECESSARY FOR SCALING OF A PLOT
C      AND ITS AXES.
C    ICING:CALCULATE AMOUNT OF ACCRETION AND DETERMINE A NEW SET
C      OF AEROFOIL SFC. ELEMENT ENDPOINTS AFTER DETERMINING THE
C      AEROFOIL NOSE LOCATION.
C    GROWTH:PLOTS SUCCESSIVE AEROFOIL OUTLINES WITHIN VIEW WINDOW.
C    TRAJEC: CALCULATES TRAJECTORIES OF DROPLETS IN POTENTIAL FLOW
C            ABOUT AN AEROFOIL.
C    ACCN:CALCULATES RHS OF NON-DIMENSIONAL EQNS. OF MOTION.
C    AIRVEL:CALCULATES THE AIR VELOCITY COMPONENTS AT A GIVEN
C      LOCATION.
C    DRAG:CALCULATES THE REYNOLDS NUMBER AND DRAG COEFFICIENT
C      OF THE DROPLET AT ANY STEP ALONG ITS TRAJECTORY.
C    HIST:DETERMINES VALUE OF INTEGRAL IN HISTORY TERM.
C    RKF4: THE RUNGE-KUTTA-FEHLBERG 4TH ORDER ODE INTEGRATION TECHNIQUE
C    RK4: THE RUNGE-KUTTA 4TH ORDER ODE INTEGRATION TECHNIQUE.
C    PC4: THE PREDICTOR-CORRECTOR 4TH ORDER ODE INTEGRATION TECHNIQUE
C INTERNAL FUNCTIONS:
C    NSURF:CALCULATES THE UNROTATED X VALUE OF A POINT ON THE
C      ACCRETED AEROFOIL SFC. BASED UPON THE COLLISION EFFICIENCY,
C      DIRECTION OF GROWTH, AND OLD AEROFOIL (ROTATED) SFC. POSITION
C
C EXTERNAL SUBROUTINES:
C IMSL: (INTERNATIONAL MATHEMATICAL AND SCIENTIFIC LIBRARY)
C   LEQT1F:SOLVES SYSTEM OF EQNS.
C   ICSICU:CUBIC SPLINE INTERPOLATION
C   ZXGSN:GOLDEN SECTION SEARCH METHOD FOR FINDING FN. MINIMUM.
C
C SSPLIB: (IBM SUPPLIED SCIENTIFIC SUBROUTINE LIBRARY)
C   DELI1:INCOMPLETE ELLIPTIC INTEGRAL OF THE FIRST KIND.
C   DELI2:INCOMPLETE ELLIPTIC INTEGRAL OF THE SECOND KIND.
C   DCEL1:COMPLETE ELLIPTIC INTEGRAL OF THE FIRST KIND.
C   DCEL2:COMPLETE ELLIPTIC INTEGRAL OF THE SECOND KIND.
C
C INPUT/OUTPUT DEVICE ASSIGNMENTS:
C   3:DATA READ BY SUBPROGRAM PLTSZ TO SCALE PLOTS.
```

```
C   4 PROGRAM INPUT PARAMETERS (DESCRIBED BELOW).
C   5:INPUT CRT DEVICE FOR CONTROL OF PROGRAM.
C   6:OUTPUT CRT DEVICE FOR MONITORING OF PROGRAM.
C   7:OUTPUT HARDCOPY DEVICE FOR PRINTED OUTPUT.
C   9:OUTPUT FILE FOR STORAGE OF PLOT DESCRIPTION (CALCOMP FORMAT).
C
C PROGRAM INPUT PARAMETERS:
C   TO BE READ IN FROM INPUT DEVICE 4.  EACH GROUP OF PARAMETERS
C   IS TO BE READ FROM THE SAME LINE (CARD) USING THE SPECIFIED
C   FORMAT.  EACH DATA LINE PRECEDED BY A DESCRIPTIVE REMINDER LINE.
C   SEE EXAMPLE FOR DETAILS.
C
C    NEF=NO. OF ELEMENT ENDPTS. ON FRONT HALF OF AEROFOIL (I4)
C    NEB=NO. OF ELEMENT ENDPTS. ON BACK HALF OF AEROFOIL
C        (INCLUDES THE MIDPOINT ENDPT. (AT THETA=90)) (I4)
C    NIF=NO. OF SPLINE ENDPTS/ELEMENT ENDPT. (FRONT HALF) (I4)
C
C    ALPHA=ANGLE OF ATTACK IN DEGREES (F6.0)
C    TYPE=AEROFOIL TYPE (I5)
C        -1:ANALYTICAL CYLINDER
C         0:NACA RAZOR
C         1:CYLINDER (VORTEX SHEETS)
C    THICK=THICKNESS OF AEROFOIL IN PERCENT (F6.0)
C    XMIN=
C    XMAX= VIEWPORT SIZE IN X (2F5.0)
C    YMIN=
C    YMAX= VIEWPORT SIZE IN Y (2F5.0)
C    XZ= VELOCITY VECTOR GRID SIZE IN X (I3)
C    YZ= VELOCITY VECTOR GRID SIZE IN Y (I3)
C    ANAL=0:ESTIMATE SEGMENT LENGTH BY NUMERICAL APPROXIMATION. (I5)
C         1:DETERMINE SEGMENT LENGTH BY ANALYTICAL METHOD.
C
C    PLTFAC= PLOT REDUCTION OR EXPANSION FACTOR FOR ALL PLOTS (F7.2)
C    TRJPLA=PLOT TRAJECTORIES (0 OR 1) (I7)
C    YOL=PLOT THE YO VS L GRAPH (0, 1, OR 2) (2 PLOTS AT HALF PAGE SIZE) (I4)
C    CEL=PLOT THE CE VS L GRAPH (0, 1, OR 2) (2 PLOTS AT HALF PAGE SIZE) (I4)
C    CEX=PLOT THE CE VS X GRAPH (0, 1, OR 2) (2 PLOTS AT HALF PAGE SIZE) (I4)
C    ICEPLA=PLOT AEROFOIL AND ICE LAYERS (0 OR 1) (I7)
C    LYRMAX=MAX. NUMBER OF LAYERS TO ACCRETE (I7)
C    CETOL=CRITERION (FOR CHANGE IN CE BETWEEN ENDPTS.) TO DETERMINE
C          WHETHER OR NOT TO CREATE NEW ENDPTS. (F6.2)
C    ICE=FRACTION OF CHORD LENGTH TO BE ACCRETED PER LAYER ASSUMING
C        A COLLISION EFFICIENCY OF 100% (F7.2)
C
C    UINF=FREESTREAM VELOCITY (M/S) (F6.0)
C    C=CHORD LENGTH (M) (F6.0)
C    PINF=FREESTREAM PRESSURE (KPA) (F6.0)
C    TINF=FREESTREAM TEMPERATURE (C) (F6.0)
C    RD=DROPLET RADIUS (MICROMETERS) (F6.0)
C    A1=
C    B1=PARAMETERS FOR PREDICTOR-CORRECTOR FORMULAE (2D10.0)
C
C    CDS:DRAG COEFFICIENT FORMULATION: (I4)
C       =0 ABRAHAM (1970)
C       =1:RE < 0.01: STOKE'S DRAG
C          0.01 < RE < 5: SARTOR AND ABBOTT (1975)
C          RE > 5: ABRAHAM (1970)
C       =2:LANGMUIR AND BLODGETT (1945)
C    TRJPRA=PRINT TRAJCTORY INFO (0 OR 1) (I7)
C    PRINTI=NO. OF STEPS AT WHICH TO PRINT TRAJECTORY INFO
C           WITHIN VIEWPORT. (I7)
C    PLOTI=NO. OF STEPS AT WHICH TO PLOT TRAJECTORY WITHIN VIEWPORT.
C          (I6)
C    PRINTO=NO. OF STEPS AT WHICH TO PRINT TRAJECTORY INFO
C           OUTSIDE VIEWPORT. (I7)
C    CRIT=CRITERION (EXPRESSED AS % OF DROPLET RADIUS) USED
C         TO INDICATE SUFFICIENTLY CLOSE DROPLET APPROACH
C         TO DENOTE COLLISION (F5.0)
C    BETAO=ESTIMATED LOCAL COLLISION EFFICIENCY AT STAGNATION PT.
C          (F6.0)
C
C    NTRAJU=MANUAL MODE: NO. OF TRAJECTORIES PRINTED/PLOTTED (I7)
C          =AUTO MODE: NO. OF TRAJECTORIES DESIRED ON UPPER SFC.
C    NTRAJL=AUTO MODE: NO. OF TRAJECTORIES DESIRED ON LOWER SFC.(I7)
```

```
C     AT=O:START TRAJECTORIES AS SPECIFIED BY INPUT TERMINAL. (I3)
C         1:AUTOMATICALLY DETERMINE TRAJECTORY STARTING POINTS
C           AFTER FIRST ONE FOR EACH SFC.
C     BOTH=O: SYMMETRICAL AEROFOIL AT O DEGREES ATTACK -
C              CALCULATE TRAJECTORIES FOR UPPER SFC. ONLY.
C           1: CALCULATE TRAJECTORIES FOR BOTH SFCS. (I5)
C     EQN=O: EQN. OF MOTION INCLUDES TERMS A AND B (NO INDUCED
C             MASS OR BUOYANCY) (I4)
C          1: EQN. OF MOTION INCLUDES TERMS APRIME AND BPRIME
C          2: EQN. OF MOTION INCLUDES TERMS APRIME, BPRIME, AND
C             CPRIME (HISTORY TERM)
C     PC=INTEGRATE BY RUNGE-KUTTA (O) OR PREDICTOR-CORRECTOR (1)
C        (AFTER FIRST 3 INTERVALS) OR RUNGE-KUTTA-FEHLBERG (2) (I3)
C     DTS=NON-DIM. INITIAL TIME STEP (F6.O)
C     EPS= FOR ODE INTEGRATION TECHNIQUE RUNGE-KUTTA-FEHLBERG:
C           LOCAL ERROR DIVIDED BY LOCAL STEP SIZE. (D8.O)
C     ACN=O:DROPLET INITIAL VELOCITY VECTOR SLIGHTLY GREATER THAN
C            THAT OF THE AIR AT THAT POINT. (I4)
C         1:DROPLET INITIAL ACCELERATION WEIGHTED BY CHANGE IN
C            POTENTIAL FLOW FIELD.
C
C     XO=X (UPSTREAM) COORD. FOR TRAJECTORY STARTING PTS. (F10.O)
C
C     YO=Y (OFF AXIS) COORDS FOR TRAJECTORY STARTING POINTS. (F10.O)
C        INPUT ONE FOR EACH SFC. (AUTO-TRAJECTORY MODE), OR FOR ALL
C        THE TRAJECTORIES DESIRED OTHERWISE.
C
5      FORMAT(/,3I4)
10     FORMAT(/,F6.O,I5,F6.O,4F5.O,2I3,I5)
20     FORMAT(/,F7.2,I7,3I4,2I7,F6.2,F7.2)
30     FORMAT('OTHE ACCRETED AREA FOR LAYER',I3,' IS',F10.5,/,
      .' THE ACCUMULATED ACCRETED AREA IS',F10.5)
C
       DOUBLE PRECISION ALPHA, XE(101),YE(101),LEN,YNNUR,XNNLR,
      .PI,X,DFLOAT,LU(101),LL(101),XS,CETOL,ICE,YNNLR,ACCRU,ACCRL,
      .XNP,YNP,XURTLP,XLRTLP,ACCR,ACCRT,INTU,INTL,XNNUR,
      .XU(101),YU(101),XL(101),YL(101),THICK,S30,C30,
      .XLR(101),YLR(101),DSQRT,XN,YN,BPARU(4),BPARL(4),CU(100,3),
      .CL(100,3),XUR(101),YUR(101),ALPHAR,THETA,INTUP,INTLP
C
       REAL XMAX,XMIN,YMIN,YMAX,PLTFAC
       INTEGER I,J,TYPE,XZ,YZ,TRJPLA,NCOU,NCOL,IERU,IERL,LYRM1,
      .PLT,LAYER,LYRMAX,NCOL1   L,YOL,ICEPLA,AT,BOTH,FAIL,ANAL,
      .ATYPE,IABS,IU(51),IL(51),NEB,NEF,NIF,NIFP1,CEX,II,IJ,NEU,NEL
C
       COMMON ALPHAR,PI/AERO1/XE,YE/NOSE/XN,YN/FOIL/XUR,YUR,
      .XLR,YLR/LG/LU,LL/LA/ANAL/AERO3/NCOU,NCOL/ROTP/C30,S30
      ./GRID/XMIN,XMAX,YMIN,YMAX,XZ,YZ/SFCS/XU,YU,XL,YL
      ./SPLINE/CU,CL/AERO4/NEU,NEL/ENDS/IU,IL
      ./NNOSE/XNP,YNP,XURTLP,XLRTLP
C
C INPUT PARAMETERS:
       READ(4,5)NEF,NEB,NIF
       READ(4,10)ALPHA,TYPE,THICK,XMIN,XMAX,YMIN,YMAX,XZ,YZ,ANAL
       READ(4,20)PLTFAC,TRJPLA,YOL,CEL,CEX,ICEPLA,LYRMAX,CETOL,ICE
       PI=3.14159265358979324
C INITIALIZE PARAMETERS
       ALPHAR=ALPHA*PI/1.8D2
       NCOU=NEF+NEB
       NCOL=NCOU
       XN=O.ODO
       YN=O.ODO
       ACCRT=O.DO
       NIFP1=NIF+1
       IJ=1
       ATYPE=IABS(TYPE)
C
C CALCULATE AEROFOIL COORDS.
C UPPER AND LOWER COORDS  FOR LEFT HALF OF AEROFOIL
          DO 110 I=1,NEF
          IU(I)=IJ
          IL(I)=IJ
            DO 140 J=1,NIFP1
            THETA=PI/2.DO*DFLOAT((I-1)*NIFP1+J-1)/DFLOAT(NEF*NIFP1)
```

```
                  CALL COORDS(TYPE,THICK,THETA,X,YU(IJ),YL(IJ))
                  XU(IJ)=X
                  XL(IJ)=X
                  IJ=IJ+1
      140         CONTINUE
      110       CONTINUE
C UPPER AND LOWER COORDS. FOR RIGHT HALF OF AEROFOIL.
                DO 150 I=1,NEB
                THETA=PI/2.DO*(1.DO+DFLOAT(I-1)/DFLOAT(NEB-1))
                CALL COORDS(TYPE,THICK,THETA,X,YU(IJ),YL(IJ))
                XU(IJ)=X
                XL(IJ)=X
                IU(NEF+I)=IJ
                IL(NEF+I)=IJ
                IJ=IJ+1
      150       CONTINUE
              NEU=IJ-1
              NEL=NEU
              LAYER=1
C
C TRANSFORM THESE COORDS. TO ONE VECTOR OF LENGTH NCOU+NCOL-1
C    IN CLOCKWISE ORDER, WITH XE(1)=XE(NCOL+NCOU-1) - THE LEADING PT.
      100       DO 102 I=1,NCOU
                II=IU(I)
                XE(I)=XU(II)
                YE(I)=YU(II)
      102       CONTINUE
              NCOL1=NCOL-1
                DO 104 I=1,NCOL1
                J=NCOU+NCOL-I
                II=IL(I)
                XE(J)=XL(II)
                YE(J)=YL(II)
      104       CONTINUE
C
C ROTATE UPPER & LOWER SFCS. BY 30 DEG. ABOUT NOSE IN ORDER
C    TO FIT CUBIC SPLINES.
C    - SEE KENNEDY & MARSDEN (1976)
C DO NOT ROTATE IF AEROFOIL IS A CYLINDER.
              IF(ATYPE.EQ.1)GOTO 200
              S30=5.D-1
              C30=DSQRT(3.DO)/2.DO
              GOTO 210
      200     S30=0.DO
              C30=1.DO
      210       DO 320 I=1,NEU
                XUR(I)=(XU(I)-XU(1))*C30+(YU(I)-YU(1))*S30
                YUR(I)=(YU(I)-YU(1))*C30-(XU(I)-XU(1))*S30
      320       CONTINUE
                DO 330 I=1,NEL
                XLR(I)=(XL(I)-XL(1))*C30-(YL(I)-YL(1))*S30
                YLR(I)=(YL(I)-YL(1))*C30+(XL(I)-XL(1))*S30
      330       CONTINUE
C
C SET PARAMETERS FOR SPLINE FITTING
              IF(ATYPE.EQ.1)GOTO 220
              BPARU(1)=1.DO
              BPARU(2)=6.DO/(XUR(2)-XUR(1))*((YUR(2)-YUR(1))/(XUR(2)-XUR
              (1))-DSQRT(3.DO))
              BPARU(3)=0.DO
              BPARU(4)=0.DO
              BPARL(1)=1.DO
              BPARL(2)=6.DO/(XLR(2)-XLR(1))*((YLR(2)-YLR(1))/(XLR(2)-
              XLR(1))+DSQRT(3.DO))
              BPARL(3)=0.DO
              BPARL(4)=0.DO
              GOTO 230
      220     BPARU(1)=0.DO
              BPARU(2)=0.DO
              BPARU(3)=0.DO
              BPARU(4)=0.DO
              BPARL(1)=0.DO
              BPARL(2)=0.DO
              BPARL(3)=0.DO
```

```
      BPARL(4)=0.D0
C FIT CUBIC SPLINES TO EACH SFC.
230   CALL ICSICU(XUR,YUR,NEU,BPARU,CU,100,IERU)
      CALL ICSICU(XLR,YLR,NEL,BPARL,CL,100,IERL)
C
C CALCULATE INTEGRAL OF UPPER AND LOWER SFC. PROFILES.
C FIND THE LENGTHS FROM THE NOSE TO VARIOUS ENDPTS.
      LU(1)=0.D0
      LL(1)=0.D0
      INTU=0.D0
      INTL=0.D0
        DO 340 I=2,NEU
        XS=XUR(I)-XUR(I-1)
        CALL SFCLEN(XS,LEN,CU(I-1,3),CU(I-1,2),CU(I-1,1))
        LU(I)=LU(I-1)+LEN
        INTU=INTU+(((CU(I-1,3)*XS/4.D0+CU(I-1,2)/3.D0)*XS
        +CU(I-1,1)/2.D0)*XS+YUR(I-1))*XS
340     CONTINUE
        DO 350 I=2,NEL
        XS=XLR(I)-XLR(I-1)
        CALL SFCLEN(XS,LEN,CL(I-1,3),CL(I-1,2),CL(I-1,1))
        LL(I)=LL(I-1)+LEN
        INTL=INTL+(((CL(I-1,3)*XS/4.D0+CL(I-1,2)/3.D0)*XS
        +CL(I-1,1)/2.D0)*XS+YLR(I-1))*XS
350     CONTINUE
      IF(LAYER.EQ.1)GOTO 400
      XNNUR=(XN-XNP)*C30+(YN-YNP)*S30
      YNNUR=(YN-YNP)*C30-(XN-XNP)*S30
C ACCRETION AREA FOR UPPER LAYER.
      ACCRU=INTU-INTUP+YNNUR*XURTLP-XNNUR*YNNUR/2.D0
      IF(BOTH.EQ.1)GOTO 410
      ACCR=2.D0*ACCRU
      GOTO 420
410   XNNLR=(XN-XNP)*C30-(YN-YNP)*S30
      YNNLR=(YN-YNP)*C30+(XN-XNP)*S30
C ACCRETION AREA FOR LOWER LAYER
      ACCRL=INTLP-INTL-YNNLR*XLRTLP+XNNLR*YNNLR/2.D0
      ACCR=ACCRU+ACCRL
420   ACCRT=ACCRT+ACCR
      LYRM1=LAYER-1
      WRITE(6,30)LYRM1,ACCR,ACCRT
      WRITE(7,30)LYRM1,ACCR,ACCRT
400   INTUP=INTU
      INTLP=INTL
C
      IF(LAYER.GT.LYRMAX.AND.ICEPLA.EQ.1)GOTO 121
      IF(LAYER.GT.LYRMAX.AND.ICEPLA.EQ.2)GOTO  130
      IF(TYPE.GE.0)CALL POT1
      PLT=TRJPLA+YOL+CEL+CEX+ICEPLA
      IF(PLT.EQ.0)GOTO 120
      IF(LAYER.GT.1)GOTO 125
C
C OPEN PLOTTING
      CALL PLOTS
      CALL METRIC(1)
      CALL ORGEP(5.0,5.0,5.0)
      CALL FACTOR(PLTFAC)
C
125   IF(TRJPLA.EQ.0)GOTO 121
C PLOT AEROFOIL OUTLINE AND VELOCITY VECTORS
130   CALL STRMEN(TYPE)
121   CALL AIRPLT(LAYER,TRJPLA,LYRMAX)
      IF(LAYER.GT.LYRMAX)GOTO 370
C
C CALCULATE DROPLET TRAJECTORIES
120   IF(LAYER.EQ.1)CALL TRAJEC(TYPE,TRJPLA,THICK,AT,BOTH)
      IF(LAYER.GT.1)CALL TRAJEK
      IF(AT.EQ.0)GOTO 360
C
C CALCULATE COLLISION EFFICIENCY
      CALL CE(YOL,CEL,CEX,PLTFAC,THICK,LAYER)
C
C ACCRETE ICE AND FIND NEW AEROFOIL SHAPE
      CALL ICING(EFTOL,ICE,BOTH,FAIL)
```

```
      IF(LAYER.EQ.LYRMAX.AND.ICEPLA.EQ.O)GOTO 360
      IF(FAIL.EQ.1)GOTO 360
      LAYER=LAYER+1
      GOTO 100
C
C PLOT SUCCESSIVE AEROFOIL SHAPES ON ONE PLOT.
370   CALL GROWTH(ICEPLA,LYRMAX,PLTFAC,TRJPLA)
360   IF(PLT.NE.O)CALL PLOT(O.,O.,999)
      STOP
      END
C
C
      SUBROUTINE COORDS(TYPE,T,THETA,X,YU,YL)
C
C WRITTEN BY: M. OLESKIW ON:790928 LAST MODIFIED:801022
C
C CALCULATE THE UPPER AND LOWER SFC. COORDINATES OF THE AEROFOIL.
C
      DOUBLE PRECISION X,YU,YL,DSQRT,B,C,T,THETA,DCOS,EIM2,EIM1,
     .EI,DABS,A,B,DSIN,XI,ETA,E,TC
C
      INTEGER TYPE,ATYPE,IABS
C
      COMMON /JOUK1/A,B,EI
C
C IN   TYPE=AEROFOIL TYPE
C IN   T=AEROFOIL THICKNESS IN PERCENT
C IN   THETA=ANGLE FROM X AXIS
C OUT X=X-COORD. OF AEROFOIL SFC.
C OUT YU=
C OUT YL= UPPER & LOWER Y-COORDS. OF AEROFOIL SFC.
C
C
      ATYPE=IABS(TYPE)
      IF(ATYPE.EQ.1)GOTO 101
C
C CALCULATE THE SHAPE OF A NACA AEROFOIL MODIFIED TO HAVE A RAZOR-LIKE
C TRAILING EDGE BY REMOVING A LINEARLY INCREASING AMOUNT
C FROM X=0.3 TO X=1.0
C REF  GREGORY, N. & P.G. WILBY (1973), A.R.C. PAPER #1261
C        ABBOTT, I.H. & A.E. VON DOENHOFF (1959), THEORY OF WING SECTIONS,
C        TL 672 A12 1959, P113 & 321
C
C CALCULATE AEROFOIL X & Y COORDS. FOR EACH SFC.
      X=(1.DO-DCOS(THETA))/2.DO
      B=O.2969DO*DSQRT(X)-O.126DO*X-O.3516DO*X*X
      C=O.2843DO*X**3-O.1015DO*X**4
      YU=T/O.2D2*(B+C)
      IF(X.GT.O.3DO)YU=YU-(X-O.3DO)*2.1D-3*T/O.7DO/O.2D2
      IF(X.GT.O.9999999999)YU=O.DO
      YL=-YU
      RETURN
C
C CALCULATE THE X & Y COORDS. OF A CYLINDER
101   X=(1.DO-DCOS(THETA))/2.DO
      YU=DSQRT(O.25DO-(X-O.5DO)*(X-O.5DO))
      IF(X.GT.O.9999999999)YU=O.DO
      YL=-YU
      RETURN
C
C
C
      SUBROUTINE POT1
C
C WRITTEN BY: M. OLESKIW ON:781129 LAST MODIFIED:801227
C
C SOLVE FOR SURFACE VORTEX DENSITY ON 1 ELEMENT AEROFOIL IN POTENTIAL
C FLOW, GIVEN COORDS. OF AEROFOIL SURFACE
C REF  KENNEDY, J.L. & D.J. MARSDEN (1976), CAN. AERO. & SPACE JOUR.,
C        V22, #5, P243-256
C SUBROUTINE  LEQT1F OF *IMSLDPLIB  LINEAR EQN. SOLN., FULL STORAGE
C MODE, SPACE ECONOMIZER SOLN
C
      DOUBLE PRECISION XE(101),YE(101),XC(101),YC(101),R(101),
```

```
     .DATAN,DABS,DSIGN,DLOG,SI(100),CO(100),PI,CL,
     .K(101,101),WKAREA(101),D(100),XT,YT,DE,DELTA,
     .DXC,DYC,B,A,R1S,R2S,R3S,T3,T1,T2,ALPHAR,DCOS,DSIN,DSQRT
C
      INTEGER N,N1,J,J1,IDGT,IER,I,NCOU,NCOU1,NCOL,JJ
C
      COMMON ALPHAR,PI/AERO1/XE,YE/AERO3/NCOU,NCOL/AERO2/XC,YC,R,D,SI,CO
C
 10   FORMAT('OFOR EQN. SOLN. IER=',I3,/,
     .'OTHE POTENTIAL FLOW LIFT COEFFICIENT IS',F9.5)
 20   FORMAT('OCONTROL PT.  X COORD.  Y COORD.  SFC. VEL.')
 30   FORMAT(' ',I6,5X,2F10.5,F11.5)
C
      NCOU1=NCOU-1
      N=NCOU1+NCOL-1
      N1=N+1
C
C CALC. ELEMENT LENGTHS (D) AND CONTROL POINTS (XC,YC)
C XE(1)=XE(2*NCO-1)=XE(N1)=LEADING PT. X COORD.
        DO 110 J=1,N
        J1=J+1
        XC(J)=(XE(J)+XE(J1))*0.5DO
        YC(J)=(YE(J)+YE(J1))*0.5DO
        D(J)=DSQRT((XE(J1)-XE(J))**2+(YE(J1)-YE(J))**2)
 110    CONTINUE
C
C FIND TRAILING POINT COORDS. XC(N1),YC(N1): FIG.5
      XT=XE(NCOU)-(XC(NCOU1)+XC(NCOU))*0.5DO
      YT=YE(NCOU)-(YC(NCOU1)+YC(NCOU))*0.5DO
      XC(N1)=XE(NCOU)+1.D-2*XT
      YC(N1)=YE(NCOU)+1.D-2*YT
C
C FORM MATRICES K AND R: EQNS. 9 & 10
C DO FOR EACH SFC. ELEMENT J (COLUMN OF K) AND ROW OF R
        DO 120 J=1,N1
        R(J)=YC(J)*DCOS(ALPHAR)-XC(J)*DSIN(ALPHAR)
        IF(J.EQ.N1)GO TO 140
        J1=J+1
        DE=D(J)
C CALCULATE ANGLE OF ELEMENT TO X-AXIS.
        CO(J)=(XE(J1)-XE(J))/DE
        SI(J)=(YE(J1)-YE(J))/DE
        DELTA=DE/2.DO
 140      DO 130 I=1,N1
          IF(J.EQ.N1)GO TO 150
C FIND DISTANCE BETWEEN CONTROL PTS. I AND J.
          DXC=XC(I)-XC(J)
          DYC=YC(I)-YC(J)
C CALCULATE COMPONENTS OF EQN. 9 AND FIG 2
          B=DXC*CO(J)+DYC*SI(J)
          A=DYC*CO(J)-DXC*SI(J)
          R1S=A*A+(B+DELTA)*(B+DELTA)
          R2S=A*A+(B-DELTA)*(B-DELTA)
          R3S=A*A+B*B-DELTA*DELTA
          IF(R3S.LT.1.D-30)GO TO 160
          T3=DATAN(2.DO*A*DELTA/R3S)
          GO TO 170
 160      IF(DABS(A).LT.1.D-30)GO TO 180
          T3=DATAN((B+DELTA)/A)-DATAN((B-DELTA)/A)
          GO TO 170
 180      T3=DSIGN(PI,A)
 170      T1=(B+DELTA)*DLOG(R1S)
          T2=(B-DELTA)*DLOG(R2S)
          K(I,J)=(T1-T2+2.DO*A*T3-4.DO*DELTA)/4.DO/PI
          GO TO 130
C FOR LAST COLUMN OF K
 150      K(I,J)=1.DO
 130      CONTINUE
 120    CONTINUE
      IDGT=8
      CALL LEQT1F(K,1,N1,101,R,IDGT,WKAREA,IER)
C ON OUTPUT, THE SOLN. IS IN R
C
C CALCULATE THE LIFT COEFFICIENT.
```

```
      CL=0.D0
        DO 200 JJ=1,N
        CL=CL-2.D0*R(JJ)*D(JJ)
200     CONTINUE
      WRITE(6,10) IER,CL
      WRITE(7,10) IER,CL
      WRITE(7,20)
C OUTPUT AEROFOIL COORDS. AND SFC. VELOCITY.
        DO 210 JJ=1,N1
        WRITE(7,30)JJ,XC(JJ),YC(JJ),R(JJ)
210     CONTINUE
      RETURN
      END
C
C
      SUBROUTINE STRMFN(TYPE)
C
C WRITTEN BY: M. OLESKIW  ON: 800222  LAST MODIFIED: 801229
C
C CALCULATE STREAMFUNCTION ON A GRID ABOUT AN AEROFOIL SECTION
C   GIVEN THE SFC. VORTICITY DENSITY ON THE AEROFOIL AND PLOT THE
C   FLOW USING VELOCITY VECTORS.
C REF: KENNEDY, J.L. & D.F. MARSDEN (1976), CAN. AERO. & SPACE JOUR.
C       V 22, #5, PP 243-256
C
      DOUBLE PRECISION ALPHAR,XE(101),YE(101),XC(101),YC(101),GAMMA(101)
     .,D(100),SI(100),CO(100),DBLE,YUP1,YLP1,YU,YL,ZZ,DEN,PJKE,PJKA,DD,
     .PID
C
      REAL PSI(3721),K(101),DELTA,PI,ALPHAS,SNGL,SCO,SSI,X,Y,DXC,DYC,
     .XMIN,XMAX,YMIN,YMAX,B,A,R1S,R2S,T3,ATAN,SIGN,T1,T2,
     .R,ABS,LOG,FLOAT,SIN,COS,R3S,DX,DY,DPX,DPY,XPAGE,YPAGE,
     .XTIP,YTIP,XP1,YP1,YM1,U,V,AHL,AHLEN,SQRT
C
      INTEGER XZ,YZ,TYPE,J,I,M,XZ1,YZ1,F,N,NCOU,NCOL,L,II
C
      COMMON ALPHAR,PID/AERO1/XE,YE/AERO3/NCOU,NCOL/AERO2/XC,YC,GAMMA,D,
     .SI,CO
     ./GRID/XMIN,XMAX,YMIN,YMAX,XZ,YZ/SRCH/DD,II
C
C IN  TYPE=AEROFOIL TYPE.
C
C PLOT BOUNDARIES
      CALL NEWPEN(1)
      CALL ORIGIN(999,21.0,10.5,5.0,5.0)
      CALL AX2EP(3.5,3,2,0,0.9)
      CALL AXIS2(0.,0.,'X/C',-3,21.,0.,XMIN,(XMAX-XMIN)/21.,3.5)
      CALL AXIS2(21.,0.,' ',-1,-10.5,90.,0.,0.,1.75)
      CALL AX2EP(1.75,3,3,0,1.1)
      CALL AXIS2(0.,0.,'Y/C',3,10.5,90.,YMIN,(YMAX-YMIN)/10.5,-1.75)
      CALL AXIS2(0.,10.5,' ',1,-21.,0.,XMIN,(XMAX-XMIN)/21.,3.5)
C CHANGE TO SECOND PEN
      CALL NEWPEN(2)
      N=NCOU+NCOL-2
      PI=SNGL(PID)
C ALPHAR=ANGLE OF ATTACK IN RADIANS
      ALPHAS=SNGL(ALPHAR)
C
C CALCULATE STRMFN. ON GRID.
        DO 120 J=1,XZ
        X=XMIN+FLOAT(J-1)/FLOAT(XZ-1)*(XMAX-XMIN)
          DO 130 I=1,YZ
C PSI IS STORED IN VECTOR FORM BY COLUMNS.
          M=(J-1)*YZ+I
          Y=YMAX-FLOAT(I-1)/FLOAT(YZ-1)*(YMAX-YMIN)
          PSI(M)=0.0
          IF(TYPE.EQ.-1)GOTO 135
            DO 140 L=1,N
C FIND DISTANCE BETWEEN CONTROL PT. L AND GRID PT. I,J.
            DXC=X-SNGL(XC(L))
            DYC=Y-SNGL(YC(L))
C CALCULATE COMPONENTS OF EQN. 9 AND FIG. 2
            DELTA=SNGL(D(L))/2.0
            SCO=SNGL(CO(L))
```

```
                 SSI=SNGL(SI(L))
                 B=DXC*SCO+DYC*SSI
                 A=DYC*SCO-DXC*SSI
                 R1S=A*A+(B+DELTA)*(B+DELTA)
                 R2S=A*A+(B-DELTA)*(B-DELTA)
                 R3S=A*A+B*B-DELTA*DELTA
                 IF(R3S.LT.1.E-30)GO TO 160
                 T3=ATAN(2.0*A*DELTA/R3S)
                 GO TO 170
 160             IF(ABS(A).LT.1.E-30)GO TO 180
                 T3=ATAN((B+DELTA)/A)-ATAN((B-DELTA)/A)
                 GO TO 170
 180             T3=SIGN(PI,A)
 170             T1=(B+DELTA)*LOG(R1S)
                 T2=(B-DELTA)*LOG(R2S)
                 K(L)=(T1-T2+2.0*A*T3-4.0*DELTA)/4.0/PI
                 PSI(M)=PSI(M)-SNGL(GAMMA(L))*K(L)
 140             CONTINUE
               R=Y*COS(ALPHAS)-X*SIN(ALPHAS)
C ASSURE THAT PSI ON AEROFOIL = O.
               PSI(M)=PSI(M)+R-SNGL(GAMMA(N+1))
               GOTO 130
C
C STREAMFN. FOR A CYLINDER.
 135           DEN=(X-5.D-1)**2+Y*Y
               IF(DEN.LT.1.D-70)GOTO 136
               PSI(M)=Y-Y/4.DO/((X-5.D-1)**2+Y*Y)
 136           PSI(M)=O.DO
 130           CONTINUE
 120         CONTINUE
C
          XZ1=XZ-1
          YZ1=YZ-1
          F=O
          II=1
C
             DO 200 J=2,XZ1,2
             DX=(XMAX-XMIN)/FLOAT(XZ1)
             X=XMIN+FLOAT(J-1)*DX
             DPX=21./FLOAT(XZ1)
C ARROWHEAD TAIL IN FRAME COORDS.
             XPAGE=FLOAT(J-1)*DPX
             XP1=X+DX
C CHECK IF CENTERED DIFFERENCING IS OK
             IF(XP1.LE.SNGL(XE(1)))GOTO 220
             CALL SFC(DBLE(XP1),YUP1,1,O,ZZ)
             CALL SFC(DBLE(XP1),YLP1,O,O,ZZ)
             F=F+1
             IF(X.LE.SNGL(XE(1)))GOTO 220
             CALL SFC(DBLE(X),YU,1,O,ZZ)
             CALL SFC(DBLE(X),YL,O,O,ZZ)
             F=F+1
C
C DO FOR EACH COLUMN OF ARROWHEAD TAILS
 220           DO 210 I=2,YZ1,2
               DPY=10.5/FLOAT(YZ1)
               DY=(YMAX-YMIN)/FLOAT(YZ1)
               Y=YMAX-FLOAT(I-1)*DY
C ARROWHEAD TAIL IN FRAME COORDS.
               YPAGE=10.5-FLOAT(I-1)*DPY
               M=(J-1)*YZ+I
               IF(F.LE.1)GOTO 230
               YP1=Y-DY
               YM1=Y+DY
C IS CENTERED DIFFERENCING OK?
               IF(YP1.GE.SNGL(YU).OR.YM1.LE.SNGL(YL))GOTO 230
               IF(Y.GE.SNGL(YU))GOTO 250
C CHECK FOR LOCATION WITHIN AEROFOIL
               IF(Y.GT.SNGL(YL))GOTO 210
C FORWARD DIFFERENCING IN Y
               U=(PSI(M)-PSI(M+1))/DY
               GOTO 240
C BACKWARD DIFFERENCING IN Y
 250           U=(PSI(M-1)-PSI(M))/DY
```

```
            GOTO 240
C CENTERED DIFFERENCING IN Y
230         U=(PSI(M-1)-PSI(M+1))/2.0/DY
240         IF(F.EQ.0)GOTO 260
C IS CENTERED DIFFERENCING OK?
            IF(Y.GE.SNGL(YUP1).OR.Y.LE.SNGL(YLP1))GOTO 260
C BACKWARD DIFFERENCING IN X
            V=(PSI((J-2)*YZ+I)-PSI((J-1)*YZ+I))/DX
            GOTO 270
C CENTERED DIFFERENCING IN X
260         V=(PSI((J-2)*YZ+I)-PSI(J*YZ+I))/2.0/DX
C ARROWHEAD TIP
270         XTIP=XPAGE+U*DPX
            YTIP=YPAGE+V*DPX
            AHL=SQRT(U*U+V*V)
C ARROWHEAD LENGTH
            AHLEN=0.25*AHL*DPX
            CALL AROHD(XPAGE,YPAGE,XTIP,YTIP,AHLEN,0,16)
210         CONTINUE
200       CONTINUE
        RETURN
        END
C
C
        SUBROUTINE AIRPLT(LAYER,TRJPLA,LYRMAX)
C
C WRITTEN BY: M. OLESKIW  ON:800607  LAST MODIFIED: 801022
C
C PLOTS OUTLINE OF AEROFOIL WITHIN VIEW WINDOW
C
        DOUBLE PRECISION XU(101),YU(101),DD,XL(101),YL(101),
       .XE(101),YE(101)
C
        REAL XMIN,XMAX,YMIN,YMAX,SNGL,XP,YP,XPT(104),
       .YPT(104),XPE(103),YPE(103),XGR(104,10),YGR(104,10),
       .XGRE(103,10),YGRE(103,10),XPP,YPP
C
        INTEGER NCOU,NCOL,XZ,YZ,NCOB,IE,IP,I,J,NCOB1,I,II,
       .IT(10),LAYER,ITT,TRJPLA,IPB,LYRMAX,ITE(10),ITTE,
       .IEL,NEL,NEU,NELM2
C
        COMMON /GRID/XMIN,XMAX,YMIN,YMAX,XZ,YZ/GROW/XGR,YGR,
       .XGRE,YGRE,ITE,IT/AERO1/XE,YE/AERO3/NCOU,NCOL/SRCH/DD,II
       ./SFCS/XU,YU,XL,YL/AERO4/NEU,NEL
C
C IN  LAYER=INDEX OF ACCRETION LAYER
C IN  TRJPLA=PLOT TRAJECTORIES (0 OR 1)
C IN  LYRMAX=INDEX OF FINAL ACCRETION LAYER
C
        NELM2=NEL-2
        NCOB=NCOU+NCOL-1
        NCOB1=NCOB-1
        IP=0
        IE=0
C
C FOR THE UPPER SFC.:
          DO 700 J=1,NEU
          XP=SNGL(XU(J))
          YP=SNGL(YU(J))
          IF(YP.GE.YMAX)GOTO 720
          IF(XP.GE.XMAX)GOTO 730
          IP=IP+1
          XPT(IP)=XP
          YPT(IP)=YP
700       CONTINUE
        GOTO 740
720     IF(IP.GT.0)GOTO 750
        XPT(IP+1)=XP
        YPT(IP+1)=YMAX
        GOTO 760
C OUT ALONG THE TOP EDGE
750     XPT(IP+1)=(XP-XPT(IP))/(YP-YPT(IP))*(YMAX-YPT(IP))+XPT(IP)
        YPT(IP+1)=YMAX
C UPPER RIGHT CORNER
```

```
 760  IP=IP+2
      XPT(IP)=XMAX
      YPT(IP)=YMAX
      GOTO 740
C OUT ALONG THE RIGHT EDGE
 730  XPT(IP+1)=XMAX
      YPT(IP+1)=(YP-YPT(IP))/(XP-XPT(IP))*(XMAX-XPT(IP))+YPT(IP)
      IP=IP+1
C
C FOR THE LOWER SFC.:
 740  IPB=IP
      IEL=0
        DO 800 J=1,NELM2
        XP=SNGL(XL(NEL-J))
        YP=SNGL(YL(NEL-J))
        IF(XP.GE.XMAX.OR.YP.LE.YMIN)GOTO 820
        IF(J.EQ.1)GOTO 830
        IF(XPP.LE.XMAX.AND.YPP.GE.YMIN)GOTO 830
        IF(YPP.LE.YMIN)GOTO 840
C IN ON THE RIGHT EDGE
        IP=IP+1
        XPT(IP)=XMAX
        YPT(IP)=(YP-YPP)/(XP-XPP)*(XMAX-XPP)+YPP
        GOTO 820
C IN ON THE BOTTOM EDGE
 840    XPT(IP+1)=XMAX
        YPT(IP+1)=YMIN
        IP=IP+2
        XPT(IP)=(XP-XPP)/(YP-YPP)*(YMIN-YPP)+XPP
        YPT(IP)=YMIN
        GOTO 820
C ADD ANOTHER POINT WITHIN WINDOW.
 830    IP=IP+1
        XPT(IP)=XP
        YPT(IP)=YP
 820    XPP=XP
        YPP=YP
 800    CONTINUE
      IF(IP.NE.IPB)GOTO 850
      IP=IP+1
      XPT(IP)=XMAX
      YPT(IP)=YMIN
C
C ADD PARAMETERS NECESSARY FOR PLOTTING
 850  XPT(IP+1)=XPT(1)
      YPT(IP+1)=YPT(1)
      XPT(IP+2)=XMIN
      YPT(IP+2)=YMIN
        DO 200 I=1,NCOB1
        XP=SNGL(XE(I))
        YP=SNGL(YE(I))
        IF(XP.GT.XMAX)GOTO 200
        IF(YP.GT.YMAX.OR.YP.LT.YMIN)GOTO 200
        IE=IE+1
        XPE(IE)=XP
        YPE(IE)=YP
 200    CONTINUE
      XPE(IE+1)=XMIN
      YPE(IE+1)=YMIN
      XPT(IP+3)=(XMAX-XMIN)/21.0
      XPE(IE+2)=(XMAX-XMIN)/21.0
      YPT(IP+3)=(YMAX-YMIN)/10.5
      YPE(IE+2)=(YMAX-YMIN)/10.5
      IT(LAYER)=IP+3
      ITT=IP+3
C
C THESE ARE THE AEROFOIL OUTLINE POINTS TO BE PLOTTED WITHIN THE WINDOW
        DO 400 I=1,ITT
        XGR(I,LAYER)=XPT(I)
        YGR(I,LAYER)=YPT(I)
 400    CONTINUE
      ITE(LAYER)=IE+2
      ITTE=IE+2
C THESE ARE THE AEROFOIL ELEMENT ENDPTS  WITHIN THE WINDOW
```

```
         DO 450 I=1,ITTE
         XGRE(I,LAYER)=XPE(I)
         YGRE(I,LAYER)=YPE(I)
450      CONTINUE
       IF(TRJPLA.EQ.O.OR.LAYER.GT.LYRMAX)GOTO 500
       CALL NEWPEN(3)
       CALL LINE(XPT,YPT,IP+1,1,0,0)
       CALL LINEP(0.1)
       CALL LINE(XPE,YPE,IE,1,-1,0)
500    RETURN
       END
C
C
       DOUBLE PRECISION FUNCTION NSURF(XROT)
C
C WRITTEN BY: M. OLESKIW  ON. 800905  LAST MODIFIED: 801022
C
C CALCULATES THE UNROTATED X VALUE OF A POINT ON THE ACCRETED AEROFOIL
C    SURFACE BASED UPON THE COLLISION EFFICIENCY, DIRECTION OF
C    GROWTH, AND OLD AEROFOIL (ROTATED) SFC. POSITION.
C
       DOUBLE PRECISION XUR(101),YUR(101),CU(100,3),XLR(101),YLR(101),
      .CL(100,3),C30,S30,XROT,D,LENG,LEN,LU(101),LL(101),
      .L(51),YO(51),CEE(50,3),XLRT,YLRT,XN,YN,DD,SLP,K,XLRN,YLRN,
      .DSIGN,DSQRT,ICE,NSURFY,CE
C
       INTEGER J,RUN,I,ICT,ICU,ICL,NEU,NEL,NEL1
C
       COMMON /FOIL/XUR,YUR,XLR,YLR/SPLINE/CU,CL/ROTP/C30,S30
      ./IND/NSURFY,ICE,I,J,RUN/LG/LU,LL/COL/L,YO,ICT,ICU,ICL/EFF/CEE
      ./NOSE/XN,YN/AERO4/NEU,NEL
C
C  IN XROT=ROTATED X POSITION ON LOWER AEROFOIL SFC.
C
10     FORMAT('OOUT OF BOUNDS IN SEARCHING FOR AEROFOIL',
      .'OR CE SPLINES IN NSURF')
C
       IF(J.LT.1)J=1
       RUN=RUN+1
       NEL1=NEL-1
C
C FIND THE APPROPRIATE AEROFOIL SPLINE SEGMENT
120    IF(XROT.GT.XLR(J))GOTO 105
       J=J-1
       IF(J.EQ.0)GOTO 600
       GOTO 120
105    IF(XROT.LE.XLR(J+1))GOTO 110
       J=J+1
       IF(J.LE.NEL1)GOTO 105
       GOTO 600
110    D=XROT-XLR(J)
C FIND LENGTH ALONG SFC. FROM NOSE TO THIS POINT.
       CALL SFCLEN(D,LENG,CL(J,3),CL(J,2),CL(J,1))
       LEN=LL(J)+LENG
C ROTATED COORDS.
       XLRT=XROT
       YLRT=YLR(J)+((CL(J,3)*D+CL(J,2))*D+CL(J,1))*D
C
C FIND THE APPROPRIATE CE VS L SPLINE SEGMENT
       IF(I.LT.1)I=1
220    IF(-LEN.GT.L(I))GOTO 205
       I=I-1
       IF(I.EQ.0)GOTO 200
       GOTO 220
205    IF(-LEN.LE.L(I+1))GOTO 210
       I=I+1
       IF(I.LE.ICL)GOTO 205
       GOTO 600
C CE EQUALS 0 - NEW AND OLD SFCS. THE SAME.
200    NSURF=XLRT*C30+YLRT*S30+XN
       NSURFY=-XLRT*S30+YLRT*C30+YN
       RETURN
C
C CALCULATE THE COLLISION EFFICIENCY.
```

```
210   DD=-LEN-L(I)
      CE=(3.DO*CEE(I,3)*DD+2.DO*CEE(I,2))*DD+CEE(I,1)
C FIND AEROFOIL SLOPE
      SLP=(3.DO*CL(J,3)*D+2.DO*CL(J,2))*D+CL(J,1)
      K=-1.DO/SLP
C
C NEW SURFACE COORDS:
      XLRN=XLRT-DSIGN(DSQRT(ICE*ICE*CE*CE/(1.DO+K*K)),K)
      YLRN=YLRT+K*(XLRN-XLRT)
      NSURF=XLRN*C30+YLRN*S30+XN
      NSURFY=-XLRN*S30+YLRN*C30+YN
      RETURN
600   WRITE(6,10)
      WRITE(7,10)
      RETURN
      END
C
C
      SUBROUTINE SFC(X,Y,S,L,LEN)
C
C WRITTEN BY: M. OLESKIW  ON:800623  LAST MODIFIED:801102
C
C CALCULATES Y VALUES AND THE LENGTH FROM THE NOSE
C   ON THE SFC. OF THE AEROFOIL BY A CUBIC SPLINE INTERPOLATION
C
      DOUBLE PRECISION XN,YN,XUR(101),YUR(101),CU(100,3),CL(100,3),
     .XLR(101),YLR(101),XB,DELTA,DELTAP,DABS
     ..S30,C30,XR,YR,X,Y,LU(101),LL(101),LEN,LENG,D
C
      INTEGER S,L,I,JU,JL,NEU1,NEU,NEL1,NEL
C
      COMMON /NOSE/XN,YN/LG/LU,LL/FOIL/XUR,YUR,XLR,YLR/SPLINE/CU,CL
     ./ROTP/C30,S30/AERO4/NEU,NEL/SRCH/D,I
C
C IN  X=POINT AT WHICH Y VALUE IS TO BE CALCULATED
C OUT Y=SFC. POSITION ON SPLINE
C IN  S=0:LOWER SFC.
C       1:UPPER SFC.
C IN  L=1:FIND LENGTH ALONG AEROFOIL SFC. FROM NOSE TO (X,Y)
C OUT LEN=LENGTH ALONG AEROFOIL SFC. FROM NOSE TO (X,Y)
C
10    FORMAT('0OUT OF BOUNDS ON SEARCHING FOR SFC. POSITION ',
     .'IN ROUTINE SFC')
C
      JU=1
      JL=1
C ROTATED X COORD.
      XR=(X-XN)*C30
      IF(S.EQ.0)GOTO 150
C
C FOR THE UPPER SFC.
      NEU1=NEU-1
      IF(XR.GT.0.DO)GOTO 120
      IF(XR.LT.0.DO)GOTO 600
      Y=YN
      LEN=0.DO
      RETURN
C FIND THE APPROPRIATE SPLINE SEGMENT.
120   IF(XR.GT.XUR(I))GOTO 105
      I=I-1
      IF(I.EQ.0)GOTO 600
      GOTO 120
105   IF(XR.LE.XUR(I+1))GOTO 110
      I=I+1
      IF(I.LE.NEU1)GOTO 105
      GOTO 600
110   D=XR-XUR(I)
C ROTATED Y COORD.
      YR=((CU(I,3)*D+CU(I,2))*D+CU(I,1))*D+YUR(I)
      XB=XR*C30-YR*S30+XN
      DELTA=X-XB
      IF(DABS(DELTA).LE.1.D-10)GOTO 400
      DELTAP=-C30+S30*((3.DO*CU(I,3)*D+2.DO*CU(I,2))*D+CU(I,1))
      XR=XR-DELTA/DELTAP
```

```
      JU=JU+1
      GOTO 120
C
C UNROTATED Y COORD.
400   Y=YR*C30+YN+XR*S30
      IF(L.EQ.0)GOTO 300
C FIND THE SEGMENT LENGTH.
      CALL SFCLEN(D,LENG,CU(I,3),CU(I,2),CU(I,1))
      LEN=LU(I)+LENG
      GOTO 300
C
C FOR THE LOWER SFC.:
150   NEL1=NEL-1
C FIND THE APPROPRIATE SFC. SPLINE SEGMENT.
220   IF(XR.GT.XLR(I))GOTO 205
      I=I-1
      IF(I.EQ.0)GOTO 600
      GOTO 220
205   IF(XR.LE.XLR(I+1))GOTO 210
      I=I+1
      IF(I.LE.NEL1)GOTO 205
      GOTO 600
210   D=XR-XLR(I)
C ROTATED Y COORD.
      YR=((CL(I,3)*D+CL(I,2))*D+CL(I,1))*D+YLR(I)
      XB=XR*C30+YR*S30+XN
      DELTA=X-XB
      IF(DABS(DELTA).LE.1.D-10)GOTO 500
      DELTAP=-C30-S30*((3.D0*CL(I,3)*D+2.D0*CL(I,2))*D+CL(I,1))
      XR=XR-DELTA/DELTAP
      JL=JL+1
      GOTO 220
C
C UNROTATED Y COORD.
500   Y=-XR*S30+YR*C30+YN
      IF(L.EQ.0)GOTO 300
C FIND THE SEGMENT LENGTH.
      CALL SFCLEN(D,LENG,CL(I,3),CL(I,2),CL(I,1))
      LEN=LL(I)+LENG
300   RETURN
C
600   WRITE(6,10)
      WRITE(7,10)
      RETURN
      END
C
C
      SUBROUTINE SFCLEN(D,L,A,B,C)
C
C WRITTEN BY: M. OLESKIW  ON:800525  LAST MODIFIED:800902
C
C CALCULATES THE LENGTH ALONG A SEGMENT OF THE CUBIC SPLINE FIT OF THE
C   AEROFOIL SFC.
C
C REF:DOUG S. PHILLIPS (1980)
C
      DOUBLE PRECISION II,NU,E,F,DSQRT,DELTA,G,A,B,C,D,L,
     .T1,T2,T3,T4,NU1,ANU1,DABS,NUO,ANUO,K,E2,F2,E3,F3,EO2,FO2,
     .EO3,FO3,XO,X1,CK,FO,EO,F1,E1,YP,DISTP,DIST,DFLOAT,Y,
     .DLOG,DSIGN
C
      INTEGER IER,I,ANAL
C
      COMMON /LA/ANAL
C
C IN  D=ROTATED X COORDINATE OF POINT FROM BEGINNING OF SEGMENT
C         OF INTEREST TO WHICH THE LENGTH IS TO BE FOUND.
C OUT L=SEGMENT LENGTH
C IN  A=
C IN  B=
C IN  C= SPLINE PARAMETERS FOR SECTION OF INTEREST
C
```

```
      II(NU,E,F)=NU/3.DO*DSQRT(1.DO+(DELTA+NU*NU)**2)*
     .(1.DO+2.DO*DELTA*G*G/(1.DO+NU*NU*G*G))
     .+((1.DO+DELTA*G*G)*F-2.DO*DELTA*G*G*E)/3.DO/G**3
C
      IF(ANAL.EQ.O)GOTO 200
      IF(A.NE.O.DO)GOTO 100
      IF(B.NE.O.DO)GOTO 110
C
C A AND B EQUAL TO O
      L=D*DSQRT(1.DO+C*C)
      RETURN
C
C A EQUAL O, B NOT EQUAL O
110   T1=(2.DO*B*D+C)*DSQRT(1.DO+(2.DO*B*D+C)**2)
      T2=C*DSQRT(1.DO+C*C)
      T3=DLOG((2.DO*B*D+C)+DSQRT(1.DO+(2.DO*B*D+C)**2))
      T4=DLOG(C+DSQRT(1.DO+C*C))
      L=(T1-T2+T3-T4)/4.DO/B
      RETURN
C
C A NOT EQUAL O
100   NU1=DSQRT(3.DO*DABS(A))*(D+B/3.DO/A)
      ANU1=DABS(NU1)
      NUO=B/3.DO/A*DSQRT(3.DO*DABS(A))
      ANUO=DABS(NUO)
      DELTA=(C-B*B/3.DO/A)*DSIGN(1.DO,A)
      G=1.DO/(1.DO+DELTA*DELTA)**O.25DO
      K=DSQRT(5.D-1-DELTA*G*G/2.DO)
      E2=O.DO
      F2=O.DO
      EO2=O.DO
      FO2=O.DO
      XO=2.DO*G*ANUO/(1.DO-ANUO*ANUO*G*G)
      X1=2.DO*G*ANU1/(1.DO-ANU1*ANU1*G*G)
      CK=DSQRT(1.DO-K*K)
      IF(ANU1.EQ.1.DO/G)GOTO 120
      IF(ANU1.GT.1.DO/G)GOTO 130
C
C ZETA LESS THAN PI/2
      CALL DELI1(F1,X1,CK)
      CALL DELI2(E1,X1,CK,1.DO,CK*CK)
      GOTO 140
C
C ZETA GREATER THAN PI/2
130   CALL DELI1(F2,-X1,CK)
      CALL DELI2(E2,-X1,CK,1.DO,CK*CK)
C
C ZETA EQUALS PI/2
120   CALL DCEL1(F3,K,IER)
      CALL DCEL2(E3,K,1.DO,CK*CK,IER)
      F1=2.DO*F3-F2
      E1=2.DO*E3-E2
140   IF(ANUO.EQ.1.DO/G)GOTO 150
      IF(ANUO.GT.1.DO/G)GOTO 160
C
C ZETA LESS THAN PI/2
      CALL DELI1(FO,XO,CK)
      CALL DELI2(EO,XO,CK,1.DO,CK*CK)
      GOTO 170
C
C ZETA GREATER THAN PI/2
160   CALL DELI1(FO2,-XO,CK)
      CALL DELI2(EO2,-XO,CK,1.DO,CK*CK)
C
C ZETA EQUALS PI/2
150   CALL DCEL1(FO3,K,IER)
      CALL DCEL2(EO3,K,1.DO,CK*CK,IER)
      FO=2.DO*FO3-FO2
      EO=2.DO*EO3-EO2
170   L=(DSIGN(1.DO,NU1)*II(ANU1,E1,F1)-DSIGN(1.DO,NUO)*II(ANUO,EO,FO))
     ./DSQRT(3.DO*DABS(A))
      RETURN
C
C NON-ANALYTICAL (APPROXIMATE) SFC LENGTH DETERMINATION
```

```
 200    L=0.D0
        YP=0.D0
        DISTP=0.D0
          DO 210 I=1,25
          DIST=D*DFLOAT(I)/25.D0
          Y=((A*DIST+B)*DIST+C)*DIST
          L=L+DSQRT((DIST-DISTP)**2+(Y-YP)**2)
          YP=Y
          DISTP=DIST
 210      CONTINUE
        RETURN
        END
 C
 C
        SUBROUTINE CE(YOL,CEL,CEX,PLTFAC,THICK,LAYER)
 C
 C WRITTEN BY: M. OLESKIW  ON:800622  LAST MODIFIED:801227
 C
 C CALCULATE AND PLOT COLLISION EFFICIENCY OF ARBITRARY AEROFOIL
 C    GIVEN A SET OF IMPACTING TRAJECTORIES
 C
        DOUBLE PRECISION D,L(51),YO(51),BPAR(4),CEE(50,3),THICK,
       .PN,P,DIST,SLP,SSLP,DABS,CET,ALPHAR,DCOS,CEMAX,PNI,ZZ,DCOS,
       .LU(101),LL(101),XU(101),XL(101),YU(101),YL(101),Y,DBLE
 C
        REAL LPMIN,YOPMIN,LRG,YORG,SNGL,FACT(2),LP(103),FLOAT,
       .YOP(103),CEP(203),XPAR(4,10),YPAR(4,10),LS(53),YOS(53),
       .PLTFAC,XP(203),XPMIN,CEPMIN,X,XLF,XRG,COS
 C
        INTEGER CEL,F,I,ICT,IER,IRX,IRY,PX,PY,YOL,ICU,ICL,J,CEX,II,
       .KK,KL,KU,LAYER,NEU,NEL,CO,IJ
 C
        COMMON ALPHAR/COL/L,YO,ICT,ICU,ICL/EFF/CEE/PLTPRM/XPAR,YPAR
       ./CEM/P/LG/LU,LL/SFCS/XU,YU,XL,YL/SRCH/D,IJ/AERO4/NEU,NEL
 C
 C IN   YOL=PLOT YO VS L GRAPH(0 OR 1)
 C IN   CEL=PLOT CE VS L GRAPH (0 OR 1).
 C IN   CEX=PLOT CE VS X GRAPH (0 OR 1).
 C IN   PLTFAC=FACTOR FOR SCALING ALL PLOTS.
 C IN   THICK=AEROFOIL THICKNESS IN %.
 C IN   LAYER=INDEX OF ACCRETION LAYER.
 C
 10     FORMAT('-BETAO (MAX LOCAL CE) IS',F7.1,'% AT A DISTANCE OF',
       .F10.3,' FROM THE NOSE',/,' THE TOTAL COLLISION EFFICIENCY IS',
       .F7.1,'%')
 20     FORMAT('-FAILURE TO CONVERGE UPON MAX CE')
 C
        FACT(1)=1.0
        FACT(2)=0.7
 C CUBIC SPLINE END PARAMETERS
        BPAR(1)=1.D0
        BPAR(2)=6.D0*(YO(2)-YO(1))/(L(2)-L(1))**2
        BPAR(3)=1.D0
        BPAR(4)=-6.D0*(YO(ICT)-YO(ICT-1))/(L(ICT)-L(ICT-1))**2
 C CREATE SINGLE PRECISION VERSIONS OF L AND YO IN LS AND YOS
          DO 130 I=1,ICT
          LS(I)=SNGL(L(I))
 130      CONTINUE
          DO 140 I=1,ICT
          YOS(I)=SNGL(YO(I))
 140      CONTINUE
 C FIT CUBIC SPLINE TO YO VS L CURVE
        CALL ICSICU(L,YO,ICT,BPAR,CEE,50,IER)
 C
 C FIND BETAO (MAX VALUE OF LOCAL CE)
        PNI=0.D0
 540    PN=PNI
        J=0
 520    P=PN
 C FIND YO VS L SLOPE AND CE VS L SLOPE
        I=1
        IF(P.LT.L(1))P=L(1)
 500    IF(P.LT.L(I+1))GOTO 510
        I=I+1
```

```
      IF(I.LT.ICT)GOTO 500
      P=L(ICT)
510   DIST=P-L(I)
      SSLP=6.DO*CEE(I,3)
      SLP=6.DO*CEE(I,3)*DIST+2.DO*CEE(I,2)
      IF(DABS(SSLP-O.DO).LT.1.D-10)GOTO 512
C THE NEWTON-RAPHSON METHOD
      PN=P-SLP/SSLP
      J=J+1
      IF(J.LT.100)GOTO 530
      IF(PNI.LE.-4.D-2)GOTO 550
      PNI=PNI-1.D-2
      GOTO 540
550   WRITE(6,20)
      WRITE(7,20)
      GOTO 560
530   IF(DABS(PN-P).GT.1.D-5)GOTO 520
C
C FIND THE TOTAL AND MAX. COLLISION EFFICIENCY.
512   CET=(YO(ICT)-YO(1))*DCOS(ALPHAR)/THICK*1.D4
      CEMAX=((3.DO*CEE(I,3)*DIST+2.DO*CEE(I,2))*DIST+CEE(I,1))*1.D2
     .*DCOS(ALPHAR)
      WRITE(6,10)CEMAX,P,CET
      WRITE(7,10)CEMAX,P,CET
C
C DETERMINE PLOTTING PARAMETERS.
560   IF(LAYER.EQ.1)CALL PLTSZ(LS(1),LS(ICT),YOS(1),YOS(ICT),
     .LPMIN,YOPMIN,PX,PY,IRX,IRY)
      IF(LAYER.GT.1)CALL PLTSZE(LS(1),LS(ICT),YOS(1),YOS(ICT),
     .LPMIN,YOPMIN,PX,PY,IRX,IRY)
      LS(ICT+1)=LPMIN
      LS(ICT+2)=XPAR(4,IRX)/10.0**PX
      CALL NEWPEN(1)
      IF(YOL.EQ.O)GOTO 200
C
C PLOT THE YO VS L GRAPH.
      YOS(ICT+1)=YOPMIN
      YOS(ICT+2)=YPAR(4,IRY)/10.0**PY
      CALL FACTOR(FACT(YOL)*PLTFAC)
      CALL ORIGIN(999,20.0,13.0,5.0,5.0)
      CALL AX2EP(XPAR(3,IRX),3,1+PX,0,1.0)
      CALL AXIS2(0.0,0.0,'L/C',-3,XPAR(2,IRX),0.0,LPMIN,XPAR(4,IRX)/
     .10.0**PX,XPAR(3,IRX))
      CALL AXIS2(XPAR(2,IRX),0.0,' ',-1,-YPAR(2,IRY),90.0,1.0,1.0,YPAR(3
     .,IRY))
      CALL AX2EP(YPAR(3,IRY),3,1+PY,0,1.1)
      CALL AXIS2(0.0,0.0,'YO/C',4,YPAR(2,IRY),90.0,YOPMIN,YPAR(4,IRY)/
     .10.0**PY,-YPAR(3,IRY))
      CALL AXIS2(0.0,YPAR(2,IRY),' ',1,-XPAR(2,IRX),0.0,1.,1.,XPAR(3,IRX
     .))
C PLOT THE YO VS L POINTS
      CALL LINEP(0.15)
      CALL LINE(LS,YOS,ICT,1,-1,0)
      F=1
      LRG=LS(ICT)-LS(1)
      YORG=YOS(ICT)-YOS(1)
        DO 100 I=1,101
        LP(I)=LS(1)+FLOAT(I-1)/100.0*LRG
120     IF(LP(I).LE.LS(F+1))GOTO 110
        F=F+1
        GOTO 120
110     D=LP(I)-LS(F)
        YOP(I)=SNGL(((CEE(F,3)*D+CEE(F,2))*D+CEE(F,1))*D)+YOS(F)
100     CONTINUE
      YOP(102)=YOS(ICT+1)
      YOP(103)=YOS(ICT+2)
      LP(102)=LS(ICT+1)
      LP(103)=LS(ICT+2)
C PLOT THE YO VS L LINE
      CALL LINE(LP,YOP,101,1,0,1)
C
C PLOT THE CE VS L GRAPH.
200   IF(CEL.EQ.O)GOTO 300
      CALL FACTOR(FACT(CEL)*PLTFAC)
```

```
      CALL ORIGIN(999,20.0,13.0,5.0,5.0)
      CALL AX2EP(XPAR(3,IRX),3,1+PX,0,1.0)
      CALL AXIS2(0.0,0.0,'L/C',-3,XPAR(2,IRX),0.0,LPMIN,XPAR(4,IRX)/
     .10.0**PX,XPAR(3,IRX))
      CALL AXIS2(XPAR(2,IRX),0.0,' ',-1,-YPAR(2,10),90.0,0.0,1.0,YPAR(3,
     .10))
      CALL AX2EP(YPAR(3,10),3,0,0,1.1)
      CALL AXIS2(0.0,0.0,'COLLISION EFFICIENCY IN %',25,YPAR(2,10),
     .90.0,0.0,YPAR(4,10)*10.0,-YPAR(3,10))
      CALL AXIS2(0.0,YPAR(2,10),' ',1,-20.0,0.0,1.,1.,XPAR(3,IRX))
      LRG=LS(ICT)-LS(1)
      F=1
C DETERMINE PLOTTING VALUES OF CE.
        DO 210 I=1,101
        LP(I)=LS(1)+FLOAT(I-1)/100.0*LRG
230     IF(LP(I).LE.LS(F+1))GOTO 220
        F=F+1
        GOTO 230
220     D=LP(I)-LS(F)
        CEP(I)=SNGL((3.DO*CEE(F,3)*D+2.DO*CEE(F,2))*D+CEE(F,1))*100.0
        *COS(SNGL(ALPHAR))
210     CONTINUE
      LP(102)=LS(ICT+1)
      LP(103)=LS(ICT+2)
      CEP(102)=0.0
      CEP(103)=YPAR(4,10)*10.0
C PLOT THE CE VS L LINE.
      CALL LINE(LP,CEP,101,1,0,1)
300   IF(CEX.EQ.0.OR.LAYER.GT.1)GOTO 400
        DO 310 KL=1,NEL
C
C PLOT THE CE VS X GRAPH.
        IF(-LL(KL).LE.L(1))GOTO 320
310     CONTINUE
320     DO 330 KU=1,NEU
        IF(LU(KU).GT.L(ICT))GOTO 340
330     CONTINUE
340   XRG=SNGL(XL(KL)+XU(KU))
      XLF=SNGL(-XU(KU))
      CO=0
      II=ICT-1
        DO 350 KK=1,201
        X=XLF+XRG/200.*FLOAT(KK-1)
        XP(KK)=X
C DETERMINE VALUE OF L FOR EACH X.
        IF(X.GT.0.)GOTO 360
        CALL SFC(DBLE(-X),Y,1,1,ZZ)
        GOTO 370
360     CALL SFC(DBLE(X),Y,0,1,ZZ)
        ZZ=-ZZ
370     IF(CO.EQ.1)GOTO 380
        IF(ZZ.GT.L(ICT))GOTO 380
        IF(ZZ.GT.L(II))GOTO 410
        II=II-1
        IF(II.EQ.0)GOTO 390
        GOTO 370
390     CO=1
380     CEP(KK)=0.0
        GOTO 350
410     D=ZZ-L(II)
        CEP(KK)=SNGL((3.DO*CEE(II,3)*D+2.DO*CEE(II,2))*D+CEE(II,1))*100.
        *COS(SNGL(ALPHAR))
350     CONTINUE
C DETERMINE THE PLOTTING PARAMETERS.
      CALL PLTSZE(XP(1),XP(201),0.DO,99.9,XPMIN,CEPMIN,PX,PY,IRX,IRY)
      XP(202)=XPMIN
      XP(203)=XPAR(4,IRX)/10.0**PX
      CEP(202)=0.0
      CEP(203)=YPAR(4,10)*10.0
C PLOT CE VS X AXES
      CALL FACTOR(FACT(CEX)*PLTFAC)
      CALL ORIGIN(999,20.0,13.0,5.0,5.0)
      CALL AX2EP(XPAR(3,IRX),3,1+PX,0,1.0)
      CALL AXIS2(0.0,0.0,'X/C',-3,XPAR(2,IRX),0.0,XPMIN,XPAR(4,IRX)
```

```
     ./10.0**PX,XPAR(3,IRX))
      CALL AXIS2(XPAR(2,IRX),0.0,' ',-1,-YPAR(2,10),90.0,0.0,1.0,YPAR(3,
     .10))
      CALL AX2EP(YPAR(3,10),3,0,0,1,1)
      CALL AXIS2(0.0,0.0,'COLLISION EFFICIENCY IN %',25,YPAR(2,10),
     .90.0,0.0,YPAR(4,10)*10.0,-YPAR(3,10))
      CALL AXIS2(0.0,YPAR(2,10),' ',1,-20.0,0.0,1.,1.,XPAR(3,IRX))
C PLOT THE CE VS X LINE.
      CALL LINE(XP,CEP,201,1,0,1)
400   RETURN
      END
C
C
      SUBROUTINE PLTSZ(XMIN,XMAX,YMIN,YMAX,XL,YB,PX,PY,IRX,IRY)
C
C WRITTEN BY: M. OLESKIW  ON:800627  LAST MODIFIED:801018
C
C DETERMINE PARAMETERS NECESSARY FOR SCALING OF A PLOT AND ITS AXES
C
      REAL XPAR(4,10),YPAR(4,10),XD,FLOAT,AINT,XMIN,XMAX,
     .XL,YD,YMIN,YMAX,YB,DX,DY,XR,YT
C
      INTEGER PX,PNX,PY,PNY,I,J,IX,IRX,INT,IY,IRY,IFIX
C
      COMMON/PLTPRM/XPAR,YPAR
C
C IN  XMIN=
C IN  XMAX=
C IN  YMIN=
C IN  YMAX=
C OUT XL=LEFT EDGE OF PLOT
C OUT YB=BOTTOM EDGE OF PLOT
C OUT PX=POWER OF TEN IN X-AXIS RANGE
C OUT PY=POWER OF TEN IN Y-AXIS RANGE
C OUT IRX=MIN. LENGTH OF X AXIS.
C OUT IRY=MIN. LENGTH OF Y AXIS.
C
10    FORMAT(8F10.0)
C
C READ IN PLOTTING PARAMETERS
         DO 101 I=2,10
         READ(3,10)(XPAR(J,I),J=1,4),(YPAR(J,I),J=1,4)
101      CONTINUE
C
      ENTRY PLTSZE(XMIN,XMAX,YMIN,YMAX,XL,YB,PX,PY,IRX,IRY)
      PNX=0
      PNY=0
C
C DETERMINE THE PLOTTING RANGE OF THE X VARIABLE
100   PX=PNX
      XD=(XMAX-XMIN)*10.0**PX
      IF(XD.GT.10.0)PNX=PNX-1
      IF(XD.LT.1.00001)PNX=PNX+1
      IF(PNX.NE.PX)GOTO 100
C PX GIVES 1/(POWER OF TEN) OF THE X VARIABLE PLOTTING RANGE
      IX=1
120   IRX=INT(XD)+IX
      DX=FLOAT(IRX)/10.0**PX/XPAR(1,IRX)
C SET THE X VALUE AT THE LEFT GRAPH EDGE
      IF(XMIN.LT.0)XL=AINT(XMIN/DX-1.0)*DX
      IF(XMIN.GE.0)XL=AINT(XMIN/DX)*DX
      XR=XL+XPAR(1,IRX)*DX
      IF(XR.GE.XMAX.AND.IRX.NE.3.AND.
     .IRX.NE.6.AND.IRX.NE.7.AND.IRX.NE.9)GOTO 105
      IX=IX+1
      GOTO 120
105   IF(IFIX((XR-XMAX)/DX).LE.IFIX((XMIN-XL)/DX))GOTO 110
C CENTRE THE PLOT.
      XL=XL-DX
      XR=XR-DX
      GOTO 105
C
C DETERMINE THE PLOTTING RANGE OF THE Y VARIABLE
110   PY=PNY
```

```
      YD=(YMAX-YMIN)*10.0**PY
      IF(YD.GT.9.99999)PNY=PNY-1
      IF(YD.LT.1.0)PNY=PNY+1
      IF(PNY.NE.PY)GOTO 110
C PY GIVES 1/(POWER OF TEN) OF THE Y VARIABLE PLOTTING RANGE
      IY=1
130   IRY=INT(YD)+IY
      DY=FLOAT(IRY)/10.0**PY/YPAR(1,IRY)
C SET THE Y VALUE AT THE BOTTOM OF THE GRAPH
      IF(YMIN.LT.0.0)YB=AINT(YMIN/DY-1.0)*DY
      IF(YMIN.GE.0.0)YB=AINT(YMIN/DY)*DY
      YT=YB+YPAR(1,IRY)*DY
      IF(YT.GE.YMAX)GOTO 135
      IY=2
      GOTO 130
135   IF(IFIX((YT-YMAX)/DY).LE.IFIX((YMIN-YB)/DY))GOTO 140
C CENTRE THE PLOT
      YB=YB-DY
      YT=YT-DY
      GOTO 135
140   RETURN
      END
C
C
      SUBROUTINE ICING(CETOL,ICE,BOTH,FAIL)
C
C WRITTEN BY: M. OLESKIW  ON:800713  LAST MODIFIED:801227
C
C CALCULATE AMOUNT OF ACCRETION AND DETERMINE A NEW SET OF AEROFOIL
C    SURFACE ELEMENT ENDPOINTS AFTER DETERMINING THE AEROFOIL
C    NOSE LOCATION.
C
      DOUBLE PRECISION XN,YN,XNN,YNN,XUR(101),YUR(101),
     .CU(100,3),CL(100,3),XLR(101),YLR(101),L(51),YO(51),
     .D,CEE(50,3),CETOL,K,DSIGN,XLRN(101),YLRN(101),
     .S30,C30,NSURF,XURN(101),YURN(101),CEU(101),CEL(101),D1,D2,
     .XUT(101),XLT(101),XU(101),YU(101),XL(101),YL(101),
     .YUT(101),YLT(101),DABS,DSQRT,LU(101),LL(101),ICE,
     .PP,TOL,LE,RE,ICEE,ALPHAR,DCOS
      DOUBLE PRECISION NSURFY,XRMIN,XNP,YNP,XURTLP,XLRTLP
      INTEGER BOTH,J,NCOU,NCOL,NCOUN,NCOLN,ICT,ICU,ICL,I,IER,NOAC,ONCE,
     .IM,IUS,ILS,IK,FAIL,RUN,NEU,NEL,IU(51),IL(51),IUN(51),ILN(51),
     .I1,I2,J1,J2,KK,KL,LLL,IXU(101),IXL(101),IZU(101),IZL(101),IUU,ILL
C
      COMMON ALPHAR/AERO3/NCOU,NCOL/NOSE/XN,YN/FOIL/XUR,YUR,
     .XLR,YLR/ROTP/C30,S30/CEM/PP/IND/NSURFY,ICEE,I,J,RUN/AERO4/NEU,
     .NEL/COL/L,YO,ICT,ICU,ICL/EFF/CEE/SFCS/XU,YU,XL,YL/LG/LU,LL
     ./SPLINE/CU,CL/ENDS/IU,IL/NNOSE/XNP,YNP,XURTLP,XLRTLP
C
      EXTERNAL NSURF
C
C IN  CETOL=CRITERION FOR DETERMINING THE NEED FOR NEW CONTROL
C             SEGMENT ENDPOINTS.
C IN  ICE=MAX. THICKNESS OF ICE ACCRETION (ASSUMING CE=100%).
C IN  BOTH=TRAJECTORIES FOR BOTH SFCS  (0 OR 1)
C OUT FAIL=FAILURE INDICATOR.
C
10    FORMAT('-FAILURE TO CONVERGE TO NEW NOSE POSITION')
20    FORMAT('1ENDPT  X COORD   Y COORD   DIST FROM NOSE  COLL. EFF.')
30    FORMAT(' ',F14.5,F10.5,F17.5,F12.4)
40    FORMAT(' ')
C
      XURTLP=XUR(NEU)
      XLRTLP=XLR(NEL)
      J=ICL
      NOAC=0
      ONCE=0
C
C FOR THE UPPER SFC.
        DO 100 I=1,NEU
        IF(NOAC.EQ.1)GOTO 115
C DETERMINE THE APPROPRIATE CE VS L SEGMENT
110     IF(LU(I).LE.L(J+1))GOTO 120
        J=J+1
```

```
      IF(J.LT.ICT)GOTO 110
      NOAC=1
C NO ACCRETION REGION ON TOP SFC.
115   CEU(I)=0.D0
      XURN(I)=XUR(I)
      YURN(I)=YUR(I)
      GOTO 100
120   D=LU(I)-L(J)
      CEU(I)=((3.D0*CEE(J,3)*D+2.D0*CEE(J,2))*D+CEE(J,1))*DCOS(ALPHAR)
      IF(DABS(CU(I,1)).LT.1.D-20)GOTO 150
      K=-1.D0/CU(I,1)
C
C NEW ENDPTS.:
      XURN(I)=XUR(I)+DSIGN(DSQRT(ICE*ICE*CEU(I)*CEU(I)/(1.D0+K*K)),K)
      YURN(I)=YUR(I)+K*(XURN(I)-XUR(I))
      GOTO 100
C GROWTH IN Y AXIS DIRECTION
150   XURN(I)=XUR(I)
      YURN(I)=YUR(I)+CEU(I)*ICE
100   CONTINUE
      DO 160 I=1,NCOU
      IUN(I)=IU(I)
160   CONTINUE
     NCOUN=NCOU
C
C CHECK FOR NEED OF CREATING NEW CONTROL ENDPTS. ON UPPER SFC.
      DO 300 I=2,NCOU
      IF(ONCE.EQ.0)GOTO 335
      ONCE=0
      GOTO 300
335   I1=IUN(I)
      I2=IUN(I-1)
      IF(I1.EQ.I2+1)GOTO 300
      IF(CEU(I2).EQ.0.D0)GOTO 390
C CHECK FOR ZERO CE BETWEEN CONTROL ENDPTS.
      IF(CEU(I1).EQ.0.D0)GOTO 330
C CHECK FOR RAPID CHANGE IN CE
325   IF(DABS(CEU(I1)-CEU(I2)).LT.CETOL)GOTO 315
      J1=I2+1
      J2=I1
        DO 320 J=J1,J2
        IF(DABS(CEU(J)-CEU(J-1)).GE.CETOL/1.2D0)GOTO 350
320     CONTINUE
      GOTO 360
330   J1=I2+1
      J2=I1
        DO 340 J=J1,J2
        IF(CEU(J).EQ.0.D0.AND.CEU(J-1).GE.CETOL/2.D0)GOTO 350
340     CONTINUE
      GOTO 325
350   KK=J-1
      IF(J.EQ.J1)KK=J
      GOTO 370
C CHECK IF DISTANCE BETWEEN CONTROL ENDPTS. IS INCREASING SUBSTANTIALLY
315   D1=DSQRT((XUR(I1)-XUR(I2))**2+(YUR(I1)-YUR(I2))**2)
      D2=DSQRT((XURN(I1)-XURN(I2))**2+(YURN(I1)-YURN(I2))**2)
      IF(D2.LT.1.25D0*D1)GOTO 300
360   KK=(I1+I2)/2
      ONCE=1
370   KL=NCOUN-I+1
C
C SHIFT INDICES OF CONTROL ENDPTS. TO MAKE ROOM FOR A NEW ONE.
        DO 380 LLL=1,KL
        IUN(NCOUN+2-LLL)=IUN(NCOUN+1-LLL)
380     CONTINUE
      NCOUN=NCOUN+1
      IUN(I)=KK
300   CONTINUE
390   J=1
      DO 170 I=1,NEU
      IF(IUN(J).EQ.I)GOTO 180
      IXU(I)=0
      GOTO 170
180   IXU(I)=1
```

```
          J=J+1
  170     CONTINUE
        WRITE(7,20)
          DO 190 I=1,NEU
          WRITE(7,30)XU(I),YU(I),LU(I),CEU(I)
          IF(CEU(I).EQ.O.DO)GOTO 195
  190     CONTINUE
  195   IF(BOTH.EQ.O)GOTO 590
        J=ICL+1
        NOAC=O
C
C FOR THE LOWER SFC.:
          DO 200 I=1,NEL
          IF(NOAC.EQ.1)GOTO 215
C DETERMINE THE APPROPRIATE CE VS L SEGMENT.
  210     IF(-LL(I).GT.L(J))GOTO 220
          J=J-1
          IF(J.GT.O)GOTO 210
          NOAC=1
C NO ACCRETION REGION ON LOWER SFC.
  215     CEL(I)=O.DO
          XLRN(I)=XLR(I)
          YLRN(I)=YLR(I)
          GOTO 200
  220     D=-LL(I)-L(J)
          CEL(I)=((3.DO*CEE(J,3)*D+2.DO*CEE(J,2))*D+CEE(J,1))*DCOS(ALPHAR)
          IF(DABS(CL(I,1)).LT.1.D-20)GOTO 250
          K=-1.DO/CL(I,1)
C
C NEW ENDPTS.:
          XLRN(I)=XLR(I)-DSIGN(DSQRT(ICE*ICE*CEL(I)*CEL(I)/(1.DO+K*K)),K)
          YLRN(I)=YLR(I)+K*(XLRN(I)-XLR(I))
          GOTO 200
C GROWTH IN Y AXIS DIRECTION
  250     XLRN(I)=XLR(I)
          YLRN(I)=YLR(I)-CEL(I)*ICE
  200     CONTINUE
          DO 260 I=1,NCOL
          ILN(I)=IL(I)
  260     CONTINUE
        NCOLN=NCOL
        ONCE=O
C
C CHECK FOR NEED OF CREATING NEW CONTROL ENDPTS. ON LOWER SFC.
          DO 400 I=2,NCOL
          IF(ONCE.EQ.O)GOTO 435
          ONCE=O
          GOTO 400
  435     I1=ILN(I)
          I2=ILN(I-1)
          IF(I1.EQ.I2+1)GOTO 400
          IF(CEL(I2).EQ.O.DO)GOTO 905
C CHECK FOR ZERO CE BETWEEN CONTROL ENDPTS.
          IF(CEL(I1).EQ.O.DO)GOTO 430
C CHECK FOR RAPID CHANGE IN CE.
  425     IF(DABS(CEL(I1)-CEL(I2)).LT.CETOL)GOTO 415
          J1=I2+1
          J2=I1
            DO 420 J=J1,J2
            IF(DABS(CEL(J)-CEL(J-1)).GE.CETOL/1.2DO)GOTO 450
  420       CONTINUE
          GOTO 460
  430     J1=I2+1
          J2=I1
            DO 440 J=J1,J2
            IF(CEL(J).EQ.O.DO.AND.CEL(J-1).GE.CETOL/2.DO)GOTO 450
  440       CONTINUE
          GOTO 425
  450     KK=J-1
          IF(J.EQ.J1)KK=J
          GOTO 470
C CHECK IF DISTANCE BETWEEN CONTROL ENDPTS. IS INCREASING SUBSTANTIALLY.
  415     D1=DSQRT((XLR(I1)-XLR(I2))**2+(YLR(I1)-YLR(I2))**2)
          D2=DSQRT((XLRN(I1)-XLRN(I2))**2+(YLRN(I1)-YLRN(I2))**2)
```

```
            IF(D2.LT.1.25DO*D1)GOTO 400
      460   KK=(I1+I2)/2
            ONCE=1
      470   KL=NCOLN-I+1
C
C SHIFT INDICES OF CONTROL ENDPTS. TO MAKE ROOM FOR A NEW ONE.
              DO 480 LLL=1,KL
               ILN(NCOLN+2-LLL)=ILN(NCOLN+1-LLL)
      480   CONTINUE
            NCOLN=NCOLN+1
            ILN(I)=KK
      400   CONTINUE
      905  J=1
            DO 270 I=1,NEL
            IF(ILN(J).EQ.I)GOTO 280
            IXL(I)=0
            GOTO 270
      280   IXL(I)=1
            J=J+1
      270   CONTINUE
           WRITE(7,40)
            DO 230 I=1,NEL
            WRITE(7,30)XL(I),YL(I),LL(I),CEL(I)
            IF(CEL(I).EQ.O.DO)GOTO 900
      230   CONTINUE
           GOTO 900
C
C UPPER & LOWER SFCS. MIRROR IMAGES; NOSE STAYS ON THE X-AXIS.
      590   DO 595 I=1,NEU
            XLRN(I)=XURN(I)
            YLRN(I)=-YURN(I)
            IXL(I)=IXU(I)
      595   CONTINUE
           GOTO 930
C
C FIND NEW NOSE LOCATION USING THE GOLDEN SECTION SEARCH METHOD
C   OF DETERMINING THE MIN. VALUE OF THE NEW SURFACE X-COORD.
      900  ICEE=ICE
           RUN=0
           J=1
           I=1
            DO 910 KK=1,NCOL
            IF(LL(KK).GE.-PP)GOTO 920
      910   CONTINUE
      920  TOL=1.D-5
           FAIL=0
           LE=1.D-10
           RE=XLR(KK)
           CALL ZXGSN(NSURF,LE,RE,TOL,XRMIN,IER)
           IF(IER.LT.129.OR.IER.GT.132)GOTO 950
           FAIL=1
           WRITE(6,10)
           WRITE(7,10)
           GOTO 720
C NEW NOSE COORDS.:
      950  YNN=NSURFY
           XNN=NSURF(XRMIN)
C
C DE-ROTATE NEW UPPER & LOWER SFCS. ABOUT PREVIOUS NOSE POSITION
      930   DO 500 I=1,NEU
            XUT(I)=XURN(I)*C30-YURN(I)*S30+XN
            YUT(I)=XURN(I)*S30+YURN(I)*C30+YN
      500   CONTINUE
            DO 510 I=1,NEL
            XLT(I)=XLRN(I)*C30+YLRN(I)*S30+XN
            YLT(I)=-XLRN(I)*S30+YLRN(I)*C30+YN
      510   CONTINUE
           IF(BOTH.EQ.1)GOTO 520
           XNN=XUT(1)
           YNN=YUT(1)
           IM=1
      520  XU(1)=XNN
           XL(1)=XNN
           YU(1)=YNN
```

```
      YL(1)=YNN
      IUU=1
      ILL=1
      IF(BOTH.EQ.0)GOTO 625
C
C SEE IF ANY LOWER SFC. ENDPTS. ARE ABOVE THE NEW NOSE POSITION
C   & THUS BELONG ON THE UPPER SFC.
        DO 610 IM=1,NEL
        IF(DABS(YLT(IM)-YNN).LT.1.D-4)GOTO 620
        IF(YLT(IM).LT.YNN)GOTO 630
610     CONTINUE
620   IF(IM.GT.2)GOTO 640
      IF(IM.EQ.2)GOTO 650
C SAME NOSE INDEX
625   IUS=2
      ILS=2
      GOTO 665
C NEW NOSE IS NEAR FIRST ENDPT. BELOW PREVIOUS NOSE
650   IUS=1
      ILS=3
      GOTO 665
C NEW NOSE IS NEAR SECOND OR GREATER ENDPT. BELOW PREVIOUS NOSE
640   IK=IM-2
        DO 670 I=1,IK
        IUU=IUU+1
        XU(IUU)=XLT(IM-I)
        YU(IUU)=YLT(IM-I)
        IZU(IUU)=IXL(IM-I)
670     CONTINUE
      IUS=1
      ILS=IM+1
665   IZU(1)=1
      IZL(1)=1
      GOTO 660
630   IF(IM.GT.2)GOTO 680
C NEW NOSE IS BETWEEN FIRST & SECOND ENDPTS. ON LOWER SFC.
      IUS=1
      ILS=2
      GOTO 666
C NEW NOSE IS BELOW SECOND ENDPT. ON LOWER SFC.
680   IK=IM-2
        DO 690 I=1,IK
        IUU=IUU+1
        XU(IUU)=XLT(IM-I)
        YU(IUU)=YLT(IM-I)
        IZU(IUU)=IXL(IM-I)
690     CONTINUE
      IUS=1
      ILS=IM
666   IZU(1)=1
      IZL(1)=1
660     DO 700 I=IUS,NEU
        IUU=IUU+1
        XU(IUU)=XUT(I)
        YU(IUU)=YUT(I)
        IZU(IUU)=IXU(I)
        IF(I.EQ.IUS.AND.IUU.LT.3)IZU(IUU)=0
700     CONTINUE
        DO 710 I=ILS,NEL
        ILL=ILL+1
        XL(ILL)=XLT(I)
        YL(ILL)=YLT(I)
        IZL(ILL)=IXL(I)
710     CONTINUE
      NEU=IUU
      NEL=ILL
      XNP=XN
      YNP=YN
      XN=XNN
      YN=YNN
      IUU=1
        DO 730 I=1,NEU
        IF(IZU(I).EQ.0)GOTO 730
        IU(IUU)=I
```

```
        IUU=IUU+1
  730   CONTINUE
      ILL=1
        DO 740 I=1,NEL
        IF(IZL(I).EQ.0)GOTO 740
        IL(ILL)=I
        ILL=ILL+1
  740   CONTINUE
      NCOU=IUU-1
      NCOL=ILL-1
  720 RETURN
      END
C
C
      SUBROUTINE GROWTH(ICEPLA,LYRMAX,PLTFAC,TRJPLA)
C
C WRITTEN BY: M. OLESKIW  ON:800713  LAST MODIFIED:801022
C
C PLOTS SUCCESSIVE AEROFOIL OUTLINES WITHIN VIEW WINDOW
C
      REAL XGR(104,10),YGR(104,10),PLTFAC,XMIN,XMAX,YMIN,YMAX,
     .XPLT(104),YPLT(104),XGRE(103,10),YGRE(103,10),XPLTE(101),
     .YPLTE(101)
C
      INTEGER IT(10),XZ,YZ,LYRMAX,ICEPLA,ITT,I,J,TRJPLA,LYRM1,
     .ITE(10),ITTE
C
      COMMON/GROW/XGR,YGR,XGRE,YGRE,ITE,IT/GRID/XMIN,XMAX,YMIN,YMAX,XZ,
     .YZ
C
C IN  ICEPLA=PLOT ACCRETION OUTLINE. (0 OR 1)
C IN  LYRMAX=NO. OF LAYERS TO BE ACCRETED.
C IN  PLTFAC=PLOT EXPANSION/REDUCTION FACTOR.
C IN  TRJPLA=PLOT TRAJECTORIES. (0 OR 1)
C
      IF(ICEPLA.EQ.2)GOTO 120
C DRAW AXES.
      CALL NEWPEN(1)
      CALL ORIGIN(999,21.0,10.5,5.0,5.0)
      CALL AX2EP(3.5,3,2,0,0.9)
      CALL AXIS2(0.,0.,'X/C',-3,21.,0.,XMIN,(XMAX-XMIN)/21.,3.5)
      CALL AXIS2(21.,0.,' ',-1,-10.5,90.,0.,0.,1.75)
      CALL AX2EP(1.75,3,3,0,1.1)
      CALL AXIS2(0.,0.,'Y/C',3,10.5,90.,YMIN,(YMAX-YMIN)/10.5,-1.75)
      CALL AXIS2(0.,10.5,' ',1,-21.,0.,XMIN,(XMAX-XMIN)/21.,3.5)
      LYRM1=LYRMAX+1
  120   DO 100 I=1,LYRM1
        ITT=IT(I)
        ITTE=ITE(I)
          DO 110 J=1,ITT
          XPLT(J)=XGR(J,I)
          YPLT(J)=YGR(J,I)
  110     CONTINUE
          DO 210 J=1,ITTE
          XPLTE(J)=XGRE(J,I)
          YPLTE(J)=YGRE(J,I)
  210     CONTINUE
        CALL NEWPEN(3)
C DRAW ACCRETION OUTLINES.
        CALL LINE(XPLT,YPLT,IT(I)-2,1,0,0)
        CALL LINEP(0.1)
C PLOT CONTROL SEGMENT ENDPTS.
        CALL LINE(XPLTE,YPLTE,ITE(I)-2,1,-1,0)
  100   CONTINUE
      RETURN
      END
C
C
      SUBROUTINE TRAJEC(TYPE,TRJPLA,THICK,AT,BOTH)
C
C WRITTEN BY: M. OLESKIW  ON:790526  LAST MODIFIED:801227
C
C CALCULATE TRAJECTORIES OF DROPLETS IN POTENTIAL FLOW
C ABOUT AN AEROFOIL
```

```
C
      DOUBLE PRECISION DFLOAT,UINF,C,RD,CD,GS,RDS,RHOA,RHOD,NUS,
     .MU,DTS(6),DEL,XP(7),YP(7),WDSREL,DBLE,HF,UST,VST,EPS,
     .CC1,CC2,C3,C4,C5,C6,C7,C8,C9,C10,C11,C12,C13,C14,C15,C16,
     .C17,C18,C19,C20,C21,C22,C23,C24,F,TSLOPE,HFP,ADD,
     .XL,YL,COORD,CLAP,XCDUPR,YCDUPR,YCAUPR,XCDLPR,YCDLPR,
     .YCALPR,PIM1,PIM2,FPIM1,FPIM2,XCOLL,YCOLL,DABS,DSIGN,
     .LT(51,2),CLAPP,K,K1,LG,LP1,LTH,TOL,XN,YN,YOG,XI
      DOUBLE PRECISION PSI(7),DUADX,DVADY,DMIN1,DTSS,L(51),YO(51),
     .YCDU,YCDL,XCDU,XCDL,ZZ,YCAU,YCAL,UAS(6),VAS(6),RED(6),
     .LEN,PRDSTI,PRDSTO,PLDSTI,DIST,TS(500),YUS1,YLS1,YUS2,YLS2,UVAT,
     .DSQRT,PINF,TINF,CRIT,XO,USLOPE,LSLOPE,YOT(25,2),DDD
      DOUBLE PRECISION XDS(6),UDS(6),AN(2,6),YDS(6),TTLACN,VPSQ,NA,
     .VDS(6),HT(2,6),AO,A1,A2,BO,B1,B2,B3,E5B,CO,C1,C2,
     .DM1,DO,D1,D2,E5,UPI,UCI,VPI,VCI,XPI,XCI,YPI,YCI,ER1,ER2,
     .PRD,PLD,BETAO,YCG,THICK,CLAPPP,SLP,YOTUX,YOTLX,LINT
C
      REAL XMIN,XMAX,YMIN,YMAX,SNGL,X,Y,XDSP(150),YDSP(150),YPREV,
     .XPREV
C
      INTEGER I,CDS,XZ,YZ,IJ,IK,TRJEND,SMASH,ALMOST,AT,BOTH,ACN,
     .GRAZE,IC,ICL,ICT,ICU,IG,IU,NT,PLOTI,UX,LX,II,ICLL,III,
     .TRJPRA,TRJPLA,PRINTI,PRINTO,NTRAJU,NTRAJL,TYPE,TYPE2,
     .IM4,IM3,IM2,IM1,IO,IP1,ITEMP,EQN,PC,INT,EQ
C
      COMMON /EQNMN/GS,RHOA,RHOD,RDS,NUS,HF
     ./AIR/XP,YP,DEL,PSI,TYPE2/REL/UAS,VAS,RED,CD
     ./GRID/XMIN,XMAX,YMIN,YMAX,XZ,YZ
     ./PV/XDS,YDS,UDS,VDS/INTEG/AN,HT
     ./PCM/AO,A1,A2,BO,B1,B2,B3,CO,C1,C2,DM1,DO,D1,D2,
     .UPI,UCI,VPI,VCI,ER1,ER2,XPI,XCI,YPI,YCI,UST,VST
     ./LOC/TS,DTS,I,IM4,IM3,IM2,IM1,IO,IP1
      COMMON /RKFM/CC1,CC2,C3,C4,C5,C6,C7,C8,C9,C10,C11,C12,C13,C14,
     .C15,C16,C17,C18,C19,C20,C21,C22,C23,C24
     ./COL/L,YO,ICT,ICU,ICL/NOSE/XN,YN/SRCH/DDD,III
C
C IN  TYPE=AEROFOIL TYPE.
C IN  TRJPLA=PLOT DROPLET TRAJECTORIES. (O OR 1)
C IN  THICK=AEROFOIL THICKNESS IN %.
C IN  AT=AUTO TRAJECTORY MODE (O OR 1)
C IN  BOTH=CALCULATE TRAJECTORIES TO COLLIDE ON BOTH SFCS. (O OR 1)
C
10    FORMAT(/5F6.0,2D10.2)
20    FORMAT(/I4,2I7,I6,I7,F5.0,F6.0)
25    FORMAT(/2I7,I3,I5,I4,I3,F6.0,D8.0,I4)
30    FORMAT(/30F10.0)
40    FORMAT('OSTEP',T7,'TIME',T15,'DTS',T22,'XDS',T31,'YDS',T40,'PSI',
     .T49,'UAS ,T58,'UDS',T67,'VAS',T76,'VDS',T86,'RED',T94,
     .'ACCN/MOD HIST/RHS',T114,'USTAB',T123,'VSTAB')
50    FORMAT(' ',I4,F6.2,F7.4,7F9.5,F10.5,4F9.5)
60    FORMAT('OCLOSEST APPROACH IS Y=',F10.5,'  NO. OF STEPS REQUIRED=',
     .I3,'  PSI=',F8.3)
70    FORMAT('1TRAJECTORY STARTING POSITION IS X=',
     .F6.2,'  YO=',F9.5)
75    FORMAT('-TRAJECTORY STARTING POSITION IS X=',
     .F6.2,'  YO=',F9.5)
80    FORMAT('OCOLLISION COORDS: X=',F10.7,'  Y=',F10.7,'  L=',F10.7,
     .'  NO. OF STEPS REQUIRED=',I3)
90    FORMAT('OFIRST TRAJECTORY HIT AEROFOIL')
95    FORMAT('OUNEXPECTED AEROFOIL MISS')
96    FORMAT('OYO?')
97    FORMAT(F10.0)
C
C STATEMENT FUNCTION TO CALCULATE DISTANCE BETWEEN
C   AEROFOIL SLOPE AND TRAJECTORY.
      F(X)=TSLOPE*(X XL)+YL-COORD
C
C INPUT PARAMFTERS
      READ(4,10)UINF,C,PINF,TINF,RD,A1,A2
      READ(4,20)CDS,TRJPRA,PRINTI,PLOTI,PRINTO,CRIT,BETAO
      READ(4,25)NTRAJU,NTRAJL,AT,BOTH,EQN,PC,DTSS,EPS,ACN
      READ(4,30)XO
C CHECK FOR AUTO-TRAJECTORY MODE
      IF(AT.EQ.1)GOTO 700
```

```
      NT=51
      GOTO 710
C CHECK TO SEE IF COLLISION EFFICIENCIES ARE TO BE CALCULATED
C    FOR BOTH SFCS.
700   NT=BOTH+1
C
C NON-DIMENSIONAL VIEWPORT DIAGONAL LENGTH
710   LEN=DSQRT(DBLE((XMAX-XMIN)**2+(YMAX-YMIN)**2))
C PRINT LENGTH INTERVAL WITHIN VIEWPORT
      PRDSTI=LEN/DFLOAT(PRINTI)
C PLOT LENGTH INTERVAL WITHIN VIEWPORT
      PLDSTI=LEN/DFLOAT(PLOTI)
C PRINT LENGTH INTERVAL TO LEFT OF VIEWPORT
      PRDSTO=LEN/DFLOAT(PRINTO)
C NON-DIMENSIONAL ACCN. OF GRAVITY
      GS=O.DO*C/UINF/UINF
C NON-DIMENSIONAL DROPLET RADIUS
      RDS=RD*1.D-6/C
      DEL=RDS
C AIR DENSITY
      RHOA=PINF*1.D3/287.04DO/(TINF+273.16DO)
C WATER DENSITY  REF: LIST - SMT
      RHOD=999.15DO
C DYNAMIC VISCOSITY OF AIR  REF: LOZOWSKI ET AL. (1979)
      MU=1.718D-5+5.1D-8*TINF
C NON-DIMENSIONAL KINEMATIC VISCOSITY OF AIR:
      NUS=MU/RHOA/C/UINF
      TOL=1.D-5*THICK
      TYPE2=TYPE
      IF(PC.NE.2)GOTO 420
C
C DETERMINE PARAMETERS FOR RUNGE-KUTTA-FEHLBERG METHOD.
      CC1=.25DO
      CC2=3.DO/32.DO
      C3=9.DO/32.DO
      C4=1932.DO/2197.DO
      C5=72.D2/2197.DO
      C6=7296.DO/2197.DO
      C7=439.DO/216.DO
      C8=8.DO
      C9=3680.DO/513.DO
      C10=845.DO/4104.DO
      C11=8.DO/27.DO
      C12=2.DO
      C13=3544.DO/2565.DO
      C14=1859.DO/4104.DO
      C15=11.DO/40.DO
      C16=25.DO/216.DO
      C17=1408.DO/2565.DO
      C18=2197.DO/4104.DO
      C19=.2DO
      C20=16.DO/135.DO
      C21=6656.DO/12825.DO
      C22=28561.DO/56430.DO
      C23=9.DO/50.DO
      C24=2.DO/55.DO
      GOTO 400
420   IF(PC.NE.1)GOTO 400
C
C DETERMINE PARAMETERS FOR PREDICTOR-CORRECTOR METHOD.
      AO=1.DO-A1-A2
      BO=(55.DO+9.DO*A1+8.DO*A2)/24.DO
      B1=(-59.DO+19.DO*A1+32.DO*A2)/24.DO
      B2=(37.DO-5.DO*A1+8.DO*A2)/24.DO
      B3=(-9.DO+A1)/24.DO
      E5B=(251.DO-19.DO*A1-8.DO*A2)/6.DO
      C1=A1
      C2=A2
      CO=1.DO-C1-C2
      DM1=(9.DO-C1)/24.DO
      DO=(19.DO+13.DO*C1+8.DO*C2)/24.DO
      D1=(-5.DO+13.DO*C1+32.DO*C2)/24.DO
      D2=(1.DO-C1+8.DO*C2)/24.DO
      E5=(-19.DO+11.DO*C1-8.DO*C2)/6.DO
```

```
          ER1=E5B/(E5B-E5)
          ER2=E5/(E5B-E5)
C
C FOR EACH TRAJECTORY (OR TRAJECTORY SET):
          ENTRY TRAJEK
400       IF(AT.EQ.O)GOTO 390
          READ(4,30)(YO(I),I=1,NT)
390         DO 200 IJ=1,NT
            IF(AT.EQ.1)GOTO 395
            WRITE(6,96)
            READ(5,97)YO(IJ)
            IF(DABS(YO(IJ)).LT.1.D-10)GOTO 690
395         IG=1
            GRAZE=1
            IC=1
            INT=O
            K1=1.DO
            K=O.85DO
            YDS(1)=YO(IJ)
C SET COUNTERS
405         IM4=2
            IM3=3
            IM2=4
            IM1=5
            IO=6
            IP1=1
C
C DROPLET AT INITIAL POSITION
            XDS(1)=XO
            WRITE(6,75) XDS(1),YDS(1)
            WRITE(7,70) XDS(1),YDS(1)
            IF(PC.NE.1)GOTO 410
C
C SET PREVIOUS PREDICTOR-CORRECTOR VALUES TO O.
            XPI=O.DO
            XCI=O.DO
            YPI=O.DO
            YCI=O.DO
            UPI=O.DO
            UCI=O.DO
            VPI=O.DO
            VCI=O.DO
410         IF(ACN.EQ.1)GOTO 415
C
C SET DROPLET TRAVELLING WITH JUST SLIGHTLY GREATER VELOCITY
C  THAN AIR (RED=O.001)
            CALL AIRVEL(XDS(1),YDS(1),UAS(1),VAS(1),4)
C CALCULATE TOTAL AIR VELOCITY.
            UVAT=DSQRT(UAS(1)*UAS(1)+VAS(1)*VAS(1))
C CALCULATE TOTAL STARTING RELATIVE VELOCITY.
            WDSREL=1.D-3*NUS/2.DO/RDS
C CALCULATE INITIAL DROPLET VELOCITY
            UDS(1)=UAS(1)*(1.DO+WDSREL/UVAT)
            VDS(1)=VAS(1)*(1.DO+WDSREL/UVAT)
            GOTO 416
C SET GRID FOR INITIAL DROPLET VELOCITY CALCULATIONS
C
415         XP(6)=XDS(1)+2.DO*RDS
            XP(7)=XDS(1)+2.DO*RDS
            YP(6)=YDS(1)+RDS
            YP(7)=YDS(1)-RDS
            CALL AIRVEL(XDS(1),YDS(1),UAS(1),VAS(1),7)
C CALCULATE DUA/DX
            DUADX=(PSI(6)+PSI(4)-PSI(7)-PSI(3))/4.DO/RDS/RDS
C CALCULATE DVA/DY
            DVADY=(PSI(3)+PSI(7)-PSI(6)-PSI(4))/4.DO/RDS/RDS
C TOTAL POTENTIAL FLOW ACCELERATIVE TERM
            UVAT-DSQRT(DUADX*DUADX+DVADY*DVADY)
C CALCULATE TOTAL STARTING RELATIVE VELOCITY
            WDSREL=1.D-3*NUS/2.DO/RDS
C ASSURE STARTING RED=O.001 WEIGHTED BY POTENTIAL FLOW
C   ACCELERATIVE COMPONENTS.
            UDS(1)=UAS(1)-DUADX/UVAT*WDSREL
            VDS(1)=VAS(1)-DVADY/UVAT*WDSREL
```

```
416     CALL DRAG(UDS(1),VDS(1),UAS(1),VAS(1),CDS,RED(1),CD)
        HT(1,1)=0.D0
        HT(2,1)=0.D0
C CALCULATE STARTING ACCELERATIONS:
        IF(EQN.EQ.0)GOTO 417
        EQ=1
        GOTO 418
417     EQ=0
418     CALL ACCN(UDS(1),VDS(1),UAS(1),VAS(1),RED(1),CD,EQ,0.D0,0)
        IF(TRJPRA.EQ.1)WRITE(7,40)
        III=1
        I=0
        IK=0
        TRJEND=0
        ALMOST=0
        DTS(1)=DTSS
        CLAP=1.D1
        XCDL=0.D0
        XCDU=0.D0
        YCAL=0.D0
        YCAU=0.D0
        YCDL=0.D0
        YCDU=0.D0
        SMASH=0
        TS(1)=0.D0
        PLD=0.D0
C
100     PRD=0.D0
        IF(PLD.LT.PLDSTI)GOTO 105
102     PLD=0.D0
C
C INCREMENT INDICES
105     ITEMP=IM4
        IM4=IM3
        IM3=IM2
        IM2=IM1
        IM1=IO
        IO=IP1
        IP1=ITEMP
        I=I+1
        HFP=HF
C
C INTEGRATE EQNS. OF MOTION
        IF(PC.EQ.2)CALL RKF4(EQN,CDS,EPS)
        IF(I.GE.4.AND.PC.EQ.1)CALL PC4(EQN,CDS)
        IF(I.LT.4.AND.PC.EQ.1.OR.PC.EQ.0)CALL RK4(EQN,CDS)
C
C CALCULATE DISTANCE SINCE LAST PRINT/PLOT OF DROPLET POSITION
        DIST=DSQRT((XDS(IP1)-XDS(IO))**2+(YDS(IP1)-YDS(IO))**2)
        PRD=PRD+DIST
        X=SNGL(XDS(IP1))
        XPREV=SNGL(XDS(IO))
        IF(X.GT.XMIN)GOTO 190
        IF(PRD.GE.PRDSTO)GOTO 230
        GOTO 105
190     Y=SNGL(YDS(IP1))
        YPREV=SNGL(YDS(IO))
C CHECK FOR OUT-OF-BOUNDS.
        IF(Y.GE.YMAX)GOTO 211
        IF(BOTH.EQ.0)GOTO 191
        IF(Y.LT.YMIN.AND.YPREV.GT.YMIN)GOTO 212
191     IF(X.GE.XMAX)GOTO 213
        PLD=PLD+DIST
        IF(X.GE.SNGL(XN).AND.X.LE.1.0)GOTO 240
        IF(IK.EQ.0.AND.TRJPLA.EQ.1)GOTO 226
        IF(PLD.GE.PLDSTI)GOTO 220
        IF(PRD.GE.PRDSTI)GOTO 230
        GOTO 105
C
C HOW CLOSE IS DROPLET TO AEROFOIL?
C COUNT NUMBER OF STEPS PAST NOSE.
240     ALMOST=ALMOST+1
        IF(YDS(IP1).LT.YN)GOTO 310
        IF(YDS(IO).GT.YN)GOTO 320
```

```
      TSLOPE=(YDS(IP1)-YDS(IO))/(XDS(IP1)-XDS(IO))
      IF(DABS(TSLOPE-O.DO).LT.1.D-70)GOTO 320
      XI=(YN-YDS(IO))/TSLOPE+XDS(IO)+RDS
      IF(XI.GT.XN.AND.XI.LT.XDS(IP1)+RDS/1.D6)GOTO 330
C
C FOR UPPER SFC.:
320   CALL SFC(XDS(IP1),YUS1,1,O,ZZ)
      CALL SFC(XDS(IP1)+RDS,YUS2,1,O,ZZ)
      USLOPE=DSQRT(RDS*RDS+(YUS2-YUS1)**2)
C PREVIOUS CLOSEST APPROACH X AND Y COORDS.
      XCDUPR=XCDU
      YCDUPR=YCDU
      YCAUPR=YCAU
C CALCULATE DROPLET Y COORD. OF CLOSEST APPROACH
      YCDU=YDS(IP1)-RDS*RDS/USLOPE
C CALCULATE DROPLET X COORD. OF CLOSEST APPROACH
      XCDU=XDS(IP1)+RDS*(YUS2-YUS1)/USLOPE
C CALCULATE AEROFOIL X AND Y COORDS. OF CLOSEST APPROACH
      CALL SFC(XCDU,YCAU,1,O,ZZ)
C STORE CLOSEST APPROACH VALUE
      CLAP=DMIN1(CLAP,(YCDU-YCAU))
C CHECK FOR DROPLET-AEROFOIL 'COLLISION'
      IF((YCDU-YCAU).LE.RDS*CRIT/1.D2)GOTO 505
      IF(PLD.GE.PLDSTI)GOTO 220
      IF(PRD.GE.PRDSTI)GOTO 230
      GOTO 105
C
C COLLISION FLAGGED:
505   SMASH=1
      IF(YCDU.GT.YCAU)GOTO 520
      IF(ALMOST.EQ.1)GOTO 500
      XL=XCDUPR
      YL=YCDUPR
      TSLOPE=(YCDU-YCDUPR)/(XCDU-XCDUPR)
      PIM2=XDS(IO)
      CALL SFC(PIM2,COORD,1,O,ZZ)
      FPIM2=F(PIM2)
      PIM1=XDS(IO)+RDS
      CALL SFC(PIM1,COORD,1,O,ZZ)
      FPIM1=F(PIM1)
      GOTO 510
C NEAR NOSE COLLISION
500   XL=XDS(IO)+RDS
      YL=YDS(IO)
      TSLOPE=(YDS(IP1)-YL)/(XDS(IP1)+RDS-XL)
      IF(DABS(TSLOPE-O.DO).LT.1.D-70)GOTO 507
      PIM2=XN
      COORD=YN
      FPIM2=F(PIM2)
      PIM1=XDS(IP1)
      COORD=YUS1
      FPIM1=F(PIM1)
      GOTO 510
507   XCOLL=XN
      YCOLL=YN
      LTH=O.DO
      GOTO 210
C AN 'ALMOST' COLLISION
520   XCOLL=XDS(IP1)
      CALL SFC(XCOLL,YCOLL,1,1,LTH)
      GOTO 210
C
C ITERATE TO COLLISION LOCATION USING SECANT METHOD.
510   XCOLL=PIM1-FPIM1*(PIM1-PIM2)/(FPIM1-FPIM2)
      IF(XCOLL.GT.XN)GOTO 511
      XCOLL=XN
      COORD=YN
      GOTO 512
511   CALL SFC(XCOLL,COORD,1,O,ZZ)
512   PIM2=PIM1
      FPIM2=FPIM1
      PIM1=XCOLL
      FPIM1=F(XCOLL)
      IF(DABS(FPIM1).GT.RDS*CRIT/1.D2)GOTO 510
```

```
            CALL SFC(XCOLL,YCOLL,1.1,LTH)
            GOTO 210
  310       IF(YDS(IO).LT.YN)GOTO 330
            TSLOPE=(YDS(IP1)-YDS(IO))/(XDS(IP1)-XDS(IO))
            IF(DABS(TSLOPE-O.DO).LT.1.D-70)GOTO 330
            XI=(YN-YDS(IO))/TSLOPE+XDS(IO)+RDS
            IF(XI.GT.XN.AND.XI.LT.XDS(IP1)+RDS/1.D6)GOTO 320
C
C FOR LOWER SFC.:
  330       CALL SFC(XDS(IP1),YLS1,O.O,ZZ)
            CALL SFC(XDS(IP1)+RDS,YLS2,O.O,ZZ)
            LSLOPE=DSQRT(RDS*RDS+(YLS2-YLS1)**2)
C PREVIOUS CLOSEST APPROACH X AND Y COORDS
            XCDLPR=XCDL
            YCDLPR=YCDL
            YCALPR=YCAL
C CALCULATE DROPLET Y COORD. OF CLOSEST APPROACH
            YCDL=YDS(IP1)+RDS*RDS/LSLOPE
C CALCULATE DROPLET X COORD. OF CLOSEST APPROACH
            XCDL=XDS(IP1)+RDS*(YLS1-YLS2)/LSLOPE
C CALCULATE AEROFOIL X AND Y COORDS. OF CLOSEST APPROACH
            CALL SFC(XCDL,YCAL,O.O,ZZ)
C STORE CLOSEST APPROACH VALUE
            CLAP=DSIGN(CLAP,-1.DO)
            CLAP=DMAX1(CLAP,(YCDL-YCAL))
C CHECK FOR DROPLET-AEROFOIL 'COLLISION'
            IF((YCAL-YCDL).LE.RDS*CRIT/1.D2)GOTO 504
            IF(PLD.GE.PLDSTI)GOTO 220
            IF(PRD.GE.PRDSTI)GOTO 230
            GOTO 105
C
C COLLISION FLAGGED
  504       SMASH=1
            IF(YCAL.GT.YCDL)GOTO 570
            IF(ALMOST.EQ.1)GOTO 550
            XL=XCDLPR
            YL=YCDLPR
            TSLOPE   (CDL-YCDLPR)/(XCDL-XCDLPR)
            PIM2=XDS(IO)
            CALL SFC(PIM2,COORD,O.O,ZZ)
            FPIM2=F(PIM2)
            PIM1=XDS(IO)+RDS
            CALL SFC(PIM1,COORD,O.O,ZZ)
            FPIM1=F(PIM1)
            GOTO 560
C NEAR NOSE COLLISION
  550       XL=XDS(IO)+RDS
            YL=YDS(IO)
            TSLOPE=(YDS(IP1)-YL)/(XDS(IP1)+RDS-XL)
            IF(DABS(TSLOPE-O.DO).LT.1.D-70)GOTO 556
            PIM2=XN
            COORD=YN
            FPIM2=F(PIM2)
            PIM1=XDS(IP1)
            COORD=YLS1
            FPIM1=F(PIM1)
            GOTO 560
  556       XCOLL=XN
            YCOLL=YN
            LTH=O.DO
            GOTO 210
C AN 'ALMOST' COLLISION
  570       XCOLL=XDS(IP1)
            CALL SFC(XCOLL,YCOLL,O.1,LTH)
            GOTO 210
C
C ITERATE TO COLLISION LOCATION USING SECANT METHOD.
  560       XCOLL=PIM1-FPIM1*(PIM1-PIM2)/(FPIM1-FPIM2)
            IF(XCOLL.GT.XN)GOTO 561
            XCOLL=XN
            COORD=YN
            GOTO 562
  561       CALL SFC(XCOLL,COORD,O.O,ZZ)
  562       PIM2=PIM1
```

```
          FPIM2=FPIM1
          PIM1=XCOLL
          FPIM1=F(XCOLL)
          IF(DABS(FPIM1).GT.RDS*CRIT/1.D2)GOTO 560
          CALL SFC(XCOLL,YCOLL,O,1,LTH)
C
C END OF TRAJECTORY FLAGGED: COLLISION
210       TRJEND=1
          IF(IK.EQ.O.OR.TRJPLA.EQ.O)GOTO 230
          IK=IK+1
          XDSP(IK)=SNGL(XCOLL)
          YDSP(IK)=SNGL(YCOLL)
          GOTO 230
C END OF TRAJECTORY FLAGGED: EXCEEDED YMAX
211       TRJEND=1
          IF(IK.EQ.O.OR.TRJPLA.EQ.O)GOTO 230
          IK=IK+1
          XDSP(IK)=(X-XPREV)/(Y-YPREV)*(YMAX-YPREV)+XPREV
          YDSP(IK)=YMAX
          GOTO 230
C END OF TRAJECTORY FLAGGED: EXCEEDED YMIN
212       TRJEND=1
          IF(IK.EQ.O.OR.TRJPLA.EQ.O)GOTO 230
          IK=IK+1
          XDSP(IK)=(X-XPREV)/(Y-YPREV)*(YMIN-YPREV)+XPREV
          YDSP(IK)=YMIN
          GOTO 230
C END OF TRAJECTORY FLAGGED; EXCEEDED XMAX
213       TRJEND=1
          IF(IK.EQ.O.OR.TRJPLA.EQ.O)GOTO 230
          IK=IK+1
          XDSP(IK)=XMAX
          YDSP(IK)=(Y-YPREV)/(X-XPREV)*(XMAX-XPREV)+YPREV
          GOTO 230
C
C STORE PLOT COORDINATES FOR FIRST POINT WITHIN WINDOW
226       IK=IK+1
          XDSP(IK)=XMIN
          YDSP(IK)=(Y-YPREV)/(X-XPREV)*(XMIN-XPREV)+YPREV
          GOTO 230
C STORE COORDS FOR LATER PLOTTING
220       IF(TRJPLA.EQ.O)GOTO 230
          IK=IK+1
          XDSP(IK)=SNGL(XDS(IO))
          YDSP(IK)=SNGL(YDS(IO))
230       IF(TRJPRA.EQ.O.AND.TRJEND.EQ.O)GOTO 100
          IF(PRD.LT.PRDSTI.AND.TRJEND.EQ.O)GOTO 102
C
C PRINT INTERVAL EXCEEDED
          TTLACN=DSQRT(AN(1,IO)*AN(1,IO)+AN(2,IO)*AN(2,IO))
          VPSQ=UDS(IO)*UDS(IO)+VDS(IO)*VDS(IO)
          NA=RDS*TTLACN/DTS(IO)/VPSQ
          IF(TRJPRA.EQ.O)GOTO 181
C
C PRINT TRAJECTORY INFO
          IF(PC.EQ.1.AND.I.GT.4)GOTO 235
          WRITE(7,50)I,TS(I),DTS(IO),XDS(IO),YDS(IO),PSI(5),UAS(IO),
     .    UDS(IO),VAS(IO),VDS(IO),RED(IO),NA,HFP
          IF(TRJEND.EQ.O)GOTO 100
          GOTO 225
235       WRITE(7,50)I,TS(I),DTS(IO),XDS(IO),YDS(IO),PSI(5),UAS(IO),
     .    UDS(IO),VAS(IO),VDS(IO),RED(IO),NA,HFP,UST,VST
          IF(TRJEND.EQ.O)GOTO 100
225       I=I+1
          WRITE(7,50)I,TS(I),DTS(IP1),X,Y
181       IF(TRJPLA.EQ.O)GOTO 180
C
C PLOT TRAJECTORIES:
          XDSP(IK+1)=XMIN
          XDSP(IK+2)=(XMAX-XMIN)/21.O
          YDSP(IK+1)=YMIN
          YDSP(IK+2)=(YMAX-YMIN)/10.5
          CALL LINE(XDSP,YDSP,IK,1,O,O)
180       IF(SMASH.EQ.1)GOTO 195
```

```
              WRITE(6,60)CLAP,I,PSI(5)
              WRITE(7,60)CLAP,I,PSI(5)
              GOTO 196
      195     WRITE(6,80)XCOLL,YCOLL,LTH,I
              WRITE(7,80)XCOLL,YCOLL,LTH,I
      196     IF(AT.EQ.0)GOTO 200
              IF(GRAZE.EQ.0)GOTO 630
              IF(SMASH.EQ.1)GOTO 610
      C
      C ITERATE TOWARD THE GRAZING TRAJECTORY
              IF(IG.EQ.1)GOTO 600
              IF(DABS(CLAP).LE.TOL)K=K1
      C FIND FIND NEW YO POSITION BY USING THE SECANT METHOD TO ESTIMATE THE
      C    LOCATION OF YO AT GRAZING
              SLP=(YOT(IG,IJ)-YOT(IG-1,IJ))/(CLAP-CLAPP)
              IF(DABS(SLP).LT.1.2DO.OR.IG.LT.3)GOTO 340
              SLP=(YOT(IG,IJ)-YOT(IG-2,IJ))/(CLAP-CLAPPP)
              K=K1
      340     YOT(IG+1,IJ)=YOT(IG,IJ)-K*CLAP*SLP
      C SET PREVIOUS CLOSEST APPROACH
              CLAPPP=CLAPP
              CLAPP=CLAP
              IG=IG+1
              YDS(1)=YOT(IG,IJ)
              GOTO 405
      C AFTER FIRST MISSING TRAJECTORY, ESTIMATE NEW YO VIA CLAP
      600     YOT(1,IJ)=YO(IJ)
              IF(DABS(CLAP).LE.TOL)K=K1
              YOT(2,IJ)=YOT(1,IJ)-K*CLAP
              CLAPP=CLAP
              IG=2
              YDS(1)=YOT(2,IJ)
              GOTO 405
      C
      C THIS IS THE GRAZING TRAJECTORY
      610     IF(IG.GT.1)GOTO 625
              WRITE(6,90)
      C ADJUST FIRST TRAJECTORY TO BE A NEAR MISS
              IF(IJ.EQ.2)GOTO 605
              YO(1)=YO(1)+5.D-4
              GOTO 606
      605     YO(1)=YO(1)-5.D-4
      606     YDS(1)=YO(1)
              GOTO 405
      625     GRAZE=0
              YOG=YOT(IG,IJ)
              YOT(1,IJ)=YOG
              YCG=YCOLL
              LG=LTH
              LP1=LG
      C
      C THESE ARE COLLIDING TRAJECTORIES
      630     IF(SMASH.EQ.1)GOTO 635
              WRITE(6,95)
              IC=IC-1
              GOTO 640
      635     IF(IC.EQ.1)GOTO 800
              IF(DABS(DSIGN(YCOLL-YN,YCG-YN)-YCOLL+YN).GT.1.D-10)GOTO 645
              IF(INT.EQ.0)GOTO 810
              INT=0
              LT(IC,IJ)=LTH
              IC=IC+1
              GOTO 820
      810     IF(LT(IC-1,IJ)-LTH.LE.1.35DO*LINT)GOTO 800
              LT(IC+1,IJ)=LTH
              YOT(IC+1,IJ)=YOT(IC,IJ)
              INT=1
              YOT(IC,IJ)=0.6DO*YOT(IC-1,IJ)+0.4DO*YOT(IC,IJ)
              YDS(1)=YOT(IC,IJ)
              GOTO 405
      800     LT(IC,IJ)=LTH
      820     IF(IC.GT.1)GOTO 633
              IF(BOTH.EQ.0)GOTO 631
              IF(IJ.EQ.2)GOTO 632
```

```
C ESTIMATED INTERVAL IN L BETWEEN COLLISIONS.
          LINT=LG/(DFLOAT(NTRAJU)+1.DO)
          ADD=-O.5DO
          GOTO 633
632       LINT=LG/(DFLOAT(NTRAJL)+1.DO)
          ADD=-O.5DO
          GOTO 633
631       LINT=LG/(DFLOAT(NTRAJU)+O.5DO)
          ADD=O.5DO
633       LP1=LP1-LINT
          IF(IC.EQ.1)LP1=LP1-ADD*LINT
          IF(DABS(DSIGN(LP1,LTH)-LP1).GT.1.D-1O.OR.DABS(LP1).LT.1.D-1O)
          GOTO 640
          IC=IC+1
C
C ESTIMATE NEW YO TO SPREAD POINTS EVENLY ALONG CE VS L CURVE
          IF (BOTH.EQ.1)GOTO 620
          YOT(IC,IJ)=2.DO*YOG/LG*LP1*(1.DO-LP1/2.DO/LG)
          YDS(1)=YOT(IC,IJ)
          GOTO 405
620       IF(IJ.EQ.2)GOTO 850
          YOT(IC,1)=BETAO*LP1*(1.DO-LP1/2.DO/LG)
          +YOG-BETAO*LG/2.DO
          GOTO 860
850       YOT(IC,2)=-BETAO*LP1*(1.DO-LP1/2.DO/LG)
          +YOG+BETAO*LG/2.DO
          IF(YOT(IC,2).LT.YOT(ICU,1))GOTO 860
          IC=IC-1
          GOTO 640
860       YDS(1)=YOT(IC,IJ)
          GOTO 405
640       IF(IJ.EQ.1)ICU=IC
          IF(IJ.EQ.2)ICL=IC
          GOTO 200
645       LT(IC,IJ)=LTH
          IF(IJ.EQ.2)GOTO 646
          ICU=IC-1
          UX=1
          GOTO 200
646       ICL=IC-1
          LX=1
200       CONTINUE
C
C TRANSFER COLLISION INFO TO SINGLE MONOTONICALLY INCREASING
C   (IN L) VECTORS
        IF(BOTH.EQ.1)GOTO 660
        IF(DABS(LT(ICU,1)-O.DO).GT.1.D-4)GOTO 651
        ICU=ICU-1
651     YO(ICU+1)=O.DO
        L(ICU+1)=O.DO
          DO 650 I=1,ICU
          IU=2*ICU+2-I
          YO(I)=-YOT(I,1)
          YO(IU)=YOT(I,1)
          L(I)=-LT(I,1)
          L(IU)=LT(I,1)
650       CONTINUE
        ICT=2*ICU+1
        ICL=ICU+1
        GOTO 690
660     IF(UX.EQ.1)YOTUX=YOT(ICU+1,1)
        IF(LX.EQ.1)YOTLX=YOT(ICL+1,2)
        II=O
          DO 67O I=1,ICL
          IF(UX.NE.1)GOTO 665
          IF(YOTUX.GE.YOT(I,2))GOTO 665
          IF(DABS(YOTUX-YOT(I,2)).LT.1.D-5)GOTO 666
          IF(DABS(YOTUX-YOT(I-1,2)).LT.1.D-5)GOTO 666
          II=II+1
          YO(II)=YOTUX
          L(II)=-LT(ICU+1,1)
666       UX=O
665       II=II+1
          YO(II)=YOT(I,2)
```

```
        L(II)=-LT(I,2)
 670    CONTINUE
      IF(UX.NE.1)GOTO 667
      IF(DABS(YOTUX-YOT(ICL,2)).LT.1.D-5)GOTO  667
      II=II+1
      YO(II)=YOTUX
      L(II)=-LT(ICU+1,1)
 667  ICLL=ICL
      ICL=II
        DO 680 I=1,ICU
        IU=ICU+1-I
        IF(LX.NE.1)GOTO 675
        IF(YOTLX.GE.YOT(IU,1))GOTO 675
        IF(DABS(YOTLX-YOT(IU,1)).LT.1.D-5)GOTO 676
        IF(DABS(YOTLX-YOT(IU+1,1)).LT.1.D-5)GOTO 676
        II=II+1
        YO(II)=YOTLX
        L(II)=LT(ICLL+1,2)
 676    LX=0
 675    II=II+1
        YO(II)=YOT(IU,1)
        L(II)=LT(IU,1)
 680    CONTINUE
      ICT=II
      ICU=ICT-ICL
 690  RETURN
      END
C
C
      SUBROUTINE ACCN(UD,VD,UA,VA,RED,CD,EQN,T,G)
C
C WRITTEN BY: M. OLESKIW  ON: 801216  LAST MODIFIED:801223
C
C CALCULATES RHS OF NON-DIMENSIONAL EQNS. OF MOTION
C
      DOUBLE PRECISION RELVEL,RED,NUS,RDS,APU,APV,BPU,BPV
     .,AN(2,6),HF,HX,HY,HT(2,6),DSQRT,AU,AV,BU,BV,RHOA,
     .RHOD,GS,ALPHAR,PI,CD,UD,VD,UA,VA,TS(500),DTS(6),T
C
      INTEGER EQN,G,I,IM4,IM3,IM2,IM1,IO,IP1
C
      COMMON ALPHAR,PI/EQNMN/GS,RHOA,RHOD,RDS,NUS,HF
     ./INTEG/AN,HT/LOC/TS,DTS,I,IM4,IM3,IM2,IM1,IO,IP1
C
C IN  UD=
C IN  VD=DROPLET VELOCITY COMPONENTS.
C IN  UA=
C IN  VA=AIR VELOCITY COMPONENTS.
C IN  RED=RELATIVE MOTION REYNOLDS NO.
C IN  CD=DRAG COEFFICIENT.
C IN  EQN=PARAMETER TO DETERMINE TERMS USED IN EQN. OF MOTION.
C IN  T=TIME AT THIS TIME STEP.
C IN  G=0:EXTRAPOLATE HISTORY TERM SEQUENCE.
C IN    1:CALCULATE NEW HISTORY TERM VALUE.
C
      RELVEL=RED*NUS/RDS/2.D0
      IF(EQN.EQ.0)GOTO 100
C
C FIRST TWO TERMS IN EQN. OF MOTION INCLUDING GRAVITATION AND
C   STEADY STATE DRAG. (INCLUDES BUOYANCY AND INDUCED MASS EFFECTS)
      APU=2.D0*(RHOD-RHOA)/(2.D0*RHOD+RHOA)*GS*DSIN(ALPHAR)
      APV=2.D0*(RHOD-RHOA)/(2.D0*RHOD+RHOA)*GS*DCOS(ALPHAR)
      BPU=0.75D0*CD*RHOA/RDS/(2.D0*RHOD+RHOA)
     .*(UD-UA)*RELVEL
      BPV=0.75D0*CD*RHOA/RDS/(2.D0*RHOD+RHOA)
     .*(VD-VA)*RELVEL
      AN(1,IP1)=APU-BPU
      AN(2,IP1)=-APV-BPV
      IF(EQN.EQ.2)GOTO 300
      HF=0.D0
      RETURN
C
C THIRD (HISTORY) TERM FOR SHEDDING OF VORTICITY
 300  CALL HIST(T,G)
```

```
      HX=-9.DO*RHOA/(2.DO*RHOD+RHOA)/RDS*DSQRT(NUS/PI)*HT(1,IP1)
      HY=-9.DO*RHOA/(2.DO*RHOD+RHOA)/RDS*DSQRT(NUS/PI)*HT(2,IP1)
      AN(1,IP1)=AN(1,IP1)+HX
      AN(2,IP1)=AN(2,IP1)+HY
      IF(G.EQ.O)RETURN
      HF=DSQRT((HX*HX+HY*HY)/((APU-BPU)**2+(APV+BPV)**2))
      RETURN
C
C FIRST TWO TERMS IN EQN. OF MOTION WITHOUT BUOYANCY AND INDUCED MASS
100   AU=GS*DSIN(ALPHAR)
      AV=GS*DCOS(ALPHAR)
      BU=0.375DO*RHOA/RHOD*CD/RDS*(UD-UA)*RELVEL
      BV=0.375DO*RHOA/RHOD*CD/RDS*(VD-VA)*RELVEL
      AN(1,IP1)=AU-BU
      AN(2,IP1)=-AV-BV
      HF=0.DO
      RETURN
      END
C
C
      SUBROUTINE AIRVEL(X,Y,UAS,VAS,NP)
C
C WRITTEN BY: M. OLESKIW  ON:800222  LAST MODIFIED:801216
C
C CALCULATES THE AIR VELOCITY COMPONENTS AT A GIVEN LOCATION
C
      DOUBLE PRECISION X,Y,UAS,VAS,XP(7),YP(7),XC(101),YC(101)
     .,DEL,GAMMA(101),D(100),K(101),PI,PJKA,PJKE,
     .SI(100),CO(100),PSI(7),DXC,DYC,DELTA,A,B,R1S,R2S,
     .R3S,DATAN,T3,DABS,DSIGN,ALPHAR,T1,T2,DLOG,R,DCOS,DSIN
C
      INTEGER L,NP,J,NCOU,NCOL,N,TYPE
C
      COMMON ALPHAR,PI/AERO3/NCOU,NCOL/AERO2/XC,YC,GAMMA,D,SI,CO
     ./AIR/XP,YP,DEL,PSI,TYPE
C
C IN  X=
C IN  Y=COORDS. AT WHICH AIR VELOCITY IS TO BE DETERMINED.
C OUT UAS=
C OUT VAS=COMPONENTS OF AIR VELOCITY.
C IN  NP=NUMBER OF POINTS AT WHICH TO CALCULATE PSI.
C
      N=NCOU+NCOL-2
C SET GRID FOR AIR VELOCITY CALCULATIONS
      XP(1)=X+DEL
      XP(2)=X-DEL
      XP(3)=X
      XP(4)=X
      XP(5)=X
      YP(1)=Y
      YP(2)=Y
      YP(3)=Y+DEL
      YP(4)=Y-DEL
      YP(5)=Y
        DO 110 J=1,NP
        IF(TYPE.EQ.-1)GOTO 115
        PSI(J)=0.0
          DO 120 L=1,N
C FIND DISTANCE BETWEEN CONTROL PT. L AND GRID PT. I,J.
          DXC=XP(J)-XC(L)
          DYC=YP(J)-YC(L)
C CALCULATE COMPONENTS OF EQN. 9 AND FIG. 2
          DELTA=D(L)/2.DO
          B=DXC*CO(L)+DYC*SI(L)
          A=DYC*CO(L)-DXC*SI(L)
          R1S=A*A+(B+DELTA)*(B+DELTA)
          R2S=A*A+(B-DELTA)*(B-DELTA)
          R3S=A*A+B*B-DELTA*DELTA
          IF(R3S.LT.1.D-30)GO TO 130
          T3=DATAN(2.DO*A*DELTA/R3S)
          GO TO 140
130       IF(DABS(A).LT.1.D-30)GO TO 150
          T3=DATAN((B+DELTA)/A)-DATAN((B-DELTA)/A)
          GO TO 140
```

```
150        T3=DSIGN(PI,A)
140        T1=(B+DELTA)*DLOG(R1S)
           T2=(B-DELTA)*DLOG(R2S)
           K(L)=(T1-T2+2.DO*A*T3-4.DO*DELTA)/4.DO/PI
           PSI(J)=PSI(J)-GAMMA(L)*K(L)
120        CONTINUE
         R=YP(J)*DCOS(ALPHAR)-XP(J)*DSIN(ALPHAR)
C ASSURE THAT PSI ON AEROFOIL = O.
         PSI(J)=PSI(J)+R-GAMMA(N+1)
         GOTO 110
115      PSI(J)=YP(J)-YP(J)/4.DO/((XP(J)-5.D-1)**2+YP(J)*YP(J))
110      CONTINUE
C
C CALCULATE AIRSPEED FROM STREAMFN.
       UAS=(PSI(3)-PSI(4))/2.DO/DEL
       VAS=(PSI(2)-PSI(1))/2.DO/DEL
       RETURN
       END
C
C
       SUBROUTINE DRAG(UDS,VDS,UAS,VAS,CDS,RED,CD)
C
C WRITTEN BY: M. OLESKIW  ON:800222  LAST MODIFIED:801216
C
C CALCULATES THE REYNOLDS NUMBER AND DRAG COEFFICIENT OF THE DROPLET
C
       DOUBLE PRECISION DSQRT,UDS,VDS,UAS,VAS,RED,CD,
      .GS,RHOA,RHOD,RDS,NUS,HF
C
       INTEGER CDS
C
       COMMON /EQNMN/GS,RHOA,RHOD,RDS,NUS,HF
C
C IN  UDS=
C IN  VDS=DROPLET VELOCITY COMPONENTS.
C IN  UAS=
C IN  VAS=AIR VELOCITY COMPONENTS.
C IN  CDS=PARAMETER TO DETERMINE DRAG COEFFICIENT FORMULATION.
C OUT RED=RELATIVE MOTION REYNOLDS NO.
C OUT CD=DRAG COEFFICIENT.
C
       RED=DSQRT((UDS-UAS)**2+(VDS-VAS)**2)*2.DO*RDS/NUS
       IF(CDS.EQ.2)GOTO 300
       IF(CDS.EQ.1.AND.RED.LE.5.DO)GOTO 100
C
C STEADY STATE DRAG COEFFICIENT OF DROPLET FOR RED < 5000
C   ABRAHAM (1970)
       CD=0.2924DO*(1+9.06DO/DSQRT(RED))**2
       RETURN
100    IF(RED.GE.1.D-2)GOTO 200
C
C STEADY STATE STOKE'S DRAG FOR RED < 0.01
       CD=24.DO/RED
       RETURN
C
C STEADY STATE DRAG COEFFICIENT FOR 0.01 < RED < 5 - SARTOR
C   AND ABBOTT (1975)
200    CD=24.DO/RED+2.2DO
       RETURN
C
C STEADY STATE DRAG COEFFICIENT - LANGMUIR & BLODGETT (1945)
300    CD=24.DO/RED+4.73DO/RED**O.37DO+6.24D-3*RED**O.38DO
       RETURN
       END
C
C
       SUBROUTINE HIST(T,G)
C
C WRITTEN BY: M. OLESKIW  ON:801216  LAST MODIFIED:801222
C
C DETERMINES VALUE OF INTEGRAL IN HISTORY TERM FOR U COMPONENT EQN.
C REF: BURDEN, R.L., J.D. FAIRES, & A.C. REYNOLDS (1978)
C       NUMERICAL ANALYSIS  P. 90   QA 297.B84
C
```

```
      DOUBLE PRECISION TAU3,TAU2,TAU1,TAUO,P11,P1O,P21,P22,P2O,
     .P33,P32,P31,P3O,TO,T1,T2,T3,TS(5OO),FO,F1,F2,F3,DSQRT,DTS(6),
     .HT(2,6),T,A,B,C,D,F,AN(2,6),P(2,745),Z2,Z33,Z32,Z31,Z3O,AA,BB
C
      INTEGER J,I,L,FF,E,MOD,JI,JJ,G,I,IM4,IM3,IM2,IM1,IO,IP1
C
      COMMON /LOC/TS,DTS,I,IM4,IM3,IM2,IM1,IO,IP1/INTEG/AN,HT
C
C IN  T=TIME AT PRESENT TIME STEP.
C IN  G=O:EXTRAPOLATE HISTORY TERM SEQUENCE.
C IN    1:CALCULATE NEW HISTORY TERM.
C
C STATEMENT FNS. TO EVALUATE PORTIONS OF THE INTEGRAL.
      TAU3(A,B)=((5.DO*A**3+6.DO*A*A*T+8.DO*A*T*T+16.DO*T**3)
     .*DSQRT(T-A)-(5.DO*B**3+6.DO*B*B*T+8.DO*B*T*T+16.DO*T**3)
     .*DSQRT(T-B))*2.DO/35.DO
      TAU2(A,B)=((3.DO*A*A+4.DO*A*T+8.DO*T*T)*DSQRT(T-A)
     .-(3.DO*B*B+4.DO*B*T+8.DO*T*T)*DSQRT(T-B))*2.DO/15.DO
      TAU1(A,B)=((2.DO*T+A)*DSQRT(T-A)-(2.DO*T+B)*DSQRT(T-B))*2.DO/3.DO
      TAUO(A,B)=2.DO*(DSQRT(T-A)-DSQRT(T-B))
C
C STATEMENT FNS. TO FIND THE TERMS OF THE LAGRANGE POLY. FIT.
      P11(TO)=(F1-FO)/(T1-TO)
      P1O(TO)=(FO*T1-F1*TO)/(T1-TO)
      Z2(A,B,C,F)=F/(A-B)/(A-C)
      P22(TO)=Z2(TO,T1,T2,FO)+Z2(T1,TO,T2,F1)+Z2(T2,TO,T1,F2)
      P21(TO)=-(T1+T2)*Z2(TO,T1,T2,FO)-(TO+T2)*Z2(T1,TO,T2,F1)
     .-(TO+T1)*Z2(T2,TO,T1,F2)
      P2O(TO)=T1*T2*Z2(TO,T1,T2,FO)+TO*T2*Z2(T1,TO,T2,F1)
     .+TO*T1*Z2(T2,TO,T1,F2)
      Z33(A,B,C,D,F)=F/(A-B)/(A-C)/(A-D)
      P33(TO)=Z33(TO,T1,T2,T3,FO)+Z33(T1,TO,T2,T3,F1)
     .+Z33(T2,TO,T1,T3,F2)+Z33(T3,TO,T1,T2,F3)
      Z32(A,B,C,D,F)=-(B+C+D)*F/(A-B)/(A-C)/(A-D)
      P32(TO)=Z32(TO,T1,T2,T3,FO)+Z32(T1,TO,T2,T3,F1)
     .+Z32(T2,TO,T1,T3,F2)+Z32(T3,TO,T1,T2,F3)
      Z31(A,B,C,D,F)=(B*C+B*D+C*D)*F/(A-B)/(A-C)/(A-D)
      P31(TO)=Z31(TO,T1,T2,T3,FO)+Z31(T1,TO,T2,T3,F1)
     .+Z31(T2,TO,T1,T3,F2)+Z31(T3,TO,T1,T2,F3)
      Z3O(A,B,C,D,F)=-B*C*D*F/(A-B)/(A-C)/(A-D)
      P3O(TO)=Z3O(TO,T1,T2,T3,FO)+Z3O(T1,TO,T2,T3,F1)
     .+Z3O(T2,TO,T1,T3,F2)+Z3O(T3,TO,T1,T2,F3)
C
      IF(G.EQ.1)GOTO 2OO
C EXTRAPOLATION OF HISTORY TERM SEQUENCE
      GOTO(14O,12O,1OO),I
      TO=TS(I-3)
      T1=TS(I-2)
      T2=TS(I-1)
      T3=TS(I).
        DO 11O J=1,2
        FO=HT(J,IM3)
        F1=HT(J,IM2)
        F2=HT(J,IM1)
        F3=HT(J,IO)
        HT(J,IP1)=P33(TO)*T**3+P32(TO)*T*T+P31(TO)*T+P3O(TO)
11O     CONTINUE
      RETURN
C
1OO   TO=TS(1)
      T1=TS(2)
      T2=TS(3)
        DO 13O J=1,2
        FO=HT(J,IM2)
        F1=HT(J,IM1)
        F2=HT(J,IO)
        HT(J,IP1)=P22(TO)*T*T+P21(TO)*T+P2O(TO)
13O     CONTINUE
      RETURN
C
12O   TO=TS(1)
      T1=TS(2)
        DO 15O J=1,2
        FO=HT(J,IM1)
```

```
            F1=HT(J,IO)
            HT(J,IP1)=P11(TO)*T+P10(TO)
 150        CONTINUE
          RETURN
C
 140      HT(1,IP1)=0.DO
          HT(2,IP1)=0.DO
          RETURN
C
 200      L=(I-4)/2*3+1
          HT(1,IP1)=0.DO
          HT(2,IP1)=0.DO
          GOTO(400,500,600,700),I
          FF=MOD(I,2)
          E=I-5+FF
C EVALUATE INTEGRAL UP TO LAST SEVERAL INTERVALS
            DO 210 J=1,E,2
            AA=TS(J)
            BB=TS(J+2)
            JI=(J-1)/2*3+1
              DO 220 JJ=1,2
              HT(JJ,IP1)=HT(JJ,IP1)+P(JJ,JI)*TAU2(AA,BB)
     .        +P(JJ,JI+1)*TAU1(AA,BB)+P(JJ,JI+2)*TAUO(AA,BB)
 220          CONTINUE
 210        CONTINUE
          IF(FF.EQ.1)GOTO 600
C EVALUATE INTEGRAL FOR LAST 4 INTERVALS
C    USING TWO INTERVAL PAIRS (FOR I EVEN)
 700      TO=TS(I-3)
          T1=TS(I-2)
          T2=TS(I-1)
            DO 710 J=1,2
            FO=AN(J,IM3)
            F1=AN(J,IM2)
            F2=AN(J,IM1)
C FIT A 2ND ORDER LAGRANGE POLYNOMIAL
            P(J,L)=P22(TO)
            P(J,L+1)=P21(TO)
            P(J,L+2)=P20(TO)
            HT(J,IP1)=HT(J,IP1)+P(J,L)*TAU2(TO,T2)+P(J,L+1)*TAU1(TO,T2)
     .      +P(J,L+2)*TAUO(TO,T2)
 710        CONTINUE
C FOR THE SECOND PAIR OF THE SET
C    (OR FOR THE VERY FIRST PAIR OF INTERVALS)
 500      TO=TS(I-1)
          T1=TS(I)
          T2=TS(I+1)
            DO 720 J=1,2
            FO=AN(J,IM1)
            F1=AN(J,IO)
            F2=AN(J,IP1)
            HT(J,IP1)=HT(J,IP1)+P22(TO)*TAU2(TO,T2)+P21(TO)*TAU1(TO,T2)
     .      +P20(TO)*TAUO(TO,T2)
 720        CONTINUE
          RETURN
C
C EVALUATE INTEGRAL FOR LAST 3 INTERVALS (FOR I ODD)
 600      TO=TS(I-2)
          T1=TS(I-1)
          T2=TS(I)
          T3=TS(I+1)
            DO 610 J=1,2
            FO=AN(J,IM2)
            F1=AN(J,IM1)
            F2=AN(J,IO)
            F3=AN(J,IP1)
            HT(J,IP1)=HT(J,IP1)+P33(TO)*TAU3(TO,T3)+P32(TO)*TAU2(TO,T3)
     .      +P31(TO)*TAU1(TO,T3)+P30(TO)*TAUO(TO,T3)
 610        CONTINUE
          RETURN
C
C EVALUATE INTEGRAL FOR THE FIRST INTERVAL
 400      TO=TS(1)
          T1=TS(2)
```

```
         DO 410 J=1,2
         FO=AN(J,IO)
         F1=AN(J,IP1)
         HT(J,IP1)=HT(J,IP1)+P11(TO)*TAU1(TO,T1)+P10(TO)*TAUO(TO,T1)
  410    CONTINUE
      RETURN
      END
C
C
      SUBROUTINE RKF4(EQN,CDS,EPS)
C
C WRITTEN BY: M. OLESKIW  ON: 800227  LAST MODIFIED:801227
C
C INTEGRATE THE DROPLET EQNS. OF MOTION (IN X AND Y) USING
C   THE 4TH ORDER RUNGE-KUTTA-FEHLBERG TECHNIQUE.
C REF: BURDEN, R.L., J.D. FAIRES, & A.C. REYNOLDS (1978),
C      NUMERICAL ANALYSIS, P.254,  QA 297.B84
C
      DOUBLE PRECISION EPS,XDS(6),UDS(6),AN(2,6),YDS(6),
     .VDS(6),HT(2,6),DTS(6),UAS(6),VAS(6),RED(6),CD,RE,
     .K1,K2,K3,K4,K5,K6,L1,L2,L3,L4,L5,L6,M1,M2,M3,M4,M5,M6,
     .N1,N2,N3,N4,N5,N6,UA,VA,RMAX,DMAX1,DMIN,DMIN1,XDEL,YDEL,
     .UDEL,VDEL,DABS,XR,YR,UR,VR,XT,YT,UT,VT,
     .XD,YD,UD,VD,CC1,CC2,C3,C4,C5,C6,C7,C8,C9,C10,C11,C12,C13,
     .C14,C15,C16,C17,C18,C19,C20,C21,C22,C23,C24,TS(500)
      DOUBLE PRECISION DMINP
C
      INTEGER EQN,CDS,I,IM4,IM3,IM2,IM1,IO,IP1,K
C
      COMMON /PV/XDS,YDS,UDS,VDS/INTEG/AN,HT
     ./LOC/TS,DTS,I,IM4,IM3,IM2,IM1,IO,IP1
     ./REL/UAS,VAS,RED,CD
     ./RKFM/CC1,CC2,C3,C4,C5,C6,C7,C8,C9,C10,C11,C12,C13,C14,
     .C15,C16,C17,C18,C19,C20,C21,C22,C23,C24
C
C IN  EQN=DENOTES PORTION OF TOTAL SYSTEM OF EQUATIONS TO BE SOLVED.
C IN  CDS=TYPE OF DRAG COEFFICIENT TO BE USED.
C IN  EPS=LOCAL ERROR PARAMETER.
C
      IF(DABS(DMINP).LT.1.D-70)DMINP=1.01DO
  100 TS(I+1)=TS(I)+DTS(IO)
      K1=DTS(IO)*UDS(IO)
      L1=DTS(IO)*VDS(IO)
      M1=DTS(IO)*AN(1,IO)
      N1=DTS(IO)*AN(2,IO)
      XD=XDS(IO)+CC1*K1
      YD=YDS(IO)+CC1*L1
      UD=UDS(IO)+CC1*M1
      VD=VDS(IO)+CC1*N1
      CALL AIRVEL(XD,YD,UA,VA,4)
      CALL DRAG(UD,VD,UA,VA,CDS,RE,CD)
C
      K2=DTS(IO)*UD
      L2=DTS(IO)*VD
      CALL ACCN(UD,VD,UA,VA,RE,CD,EQN,TS(I)+DTS(IO)/4.DO,O)
      M2=DTS(IO)*AN(1,IP1)
      N2=DTS(IO)*AN(2,IP1)
      XD=XDS(IO)+CC2*K1+C3*K2
      YD=YDS(IO)+CC2*L1+C3*L2
      UD=UDS(IO)+CC2*M1+C3*M2
      VD=VDS(IO)+CC2*N1+C3*N2
      CALL AIRVEL(XD,YD,UA,VA,4)
      CALL DRAG(UD,VD,UA,VA,CDS,RE,CD)
C
      K3=DTS(IO)*UD
      L3=DTS(IO)*VD
      CALL ACCN(UD,VD,UA,VA,RE,CD,EQN,TS(I)+DTS(IO)*3.75D-1,O)
      M3=DTS(IO)*AN(1,IP1)
      N3=DTS(IO)*AN(2,IP1)
      XD=XDS(IO)+C4*K1-C5*K2+C6*K3
      YD=YDS(IO)+C4*L1-C5*L2+C6*L3
      UD=UDS(IO)+C4*M1-C5*M2+C6*M3
      VD=VDS(IO)+C4*N1-C5*N2+C6*N3
      CALL AIRVEL(XD,YD,UA,VA,4)
```

```
      CALL DRAG(UD,VD,UA,VA,CDS,RE,CD)
C
      K4=DTS(IO)*UD
      L4=DTS(IO)*VD
      CALL ACCN(UD,VD,UA,VA,RE,CD,EQN,TS(I)+12.DO/13.DO
     .*DTS(IO),O)
      M4=DTS(IO)*AN(1,IP1)
      N4=DTS(IO)*AN(2,IP1)
      XD=XDS(IO)+C7*K1-C8*K2+C9*K3-C10*K4
      YD=YDS(IO)+C7*L1-C8*L2+C9*L3-C10*L4
      UD=UDS(IO)+C7*M1-C8*M2+C9*M3-C10*M4
      VD=VDS(IO)+C7*N1-C8*N2+C9*N3-C10*N4
      CALL AIRVEL(XD,YD,UA,VA,4)
      CALL DRAG(UD,VD,UA,VA,CDS,RE,CD)
C
      K5=DTS(IO)*UD
      L5=DTS(IO)*VD
      CALL ACCN(UD,VD,UA,VA,RE,CD,EQN,TS(I+1),O)
      M5=DTS(IO)*AN(1,IP1)
      N5=DTS(IO)*AN(2,IP1)
      XD=XDS(IO)-C11*K1+C12*K2-C13*K3+C14*K4-C15*K5
      YD=YDS(IO)-C11*L1+C12*L2-C13*L3+C14*L4-C15*L5
      UD=UDS(IO)-C11*M1+C12*M2-C13*M3+C14*M4-C15*M5
      VD=VDS(IO)-C11*N1+C12*N2-C13*N3+C14*N4-C15*N5
      CALL AIRVEL(XD,YD,UA,VA,4)
      CALL DRAG(UD,VD,UA,VA,CDS,RE,CD)
C
      K6=DTS(IO)*UD
      L6=DTS(IO)*VD
      CALL ACCN(UD,VD,UA,VA,RE,CD,EQN,TS(I)+DTS(IO)/2.DO,O)
      M6=DTS(IO)*AN(1,IP1)
      N6=DTS(IO)*AN(2,IP1)
C
C NEW DROPLET POSITION AT I+1
      XDS(IP1)=XDS(IO)+C16*K1+C17*K3+C18*K4-C19*K5
      YDS(IP1)=YDS(IO)+C16*L1+C17*L3+C18*L4-C19*L5
C NEW DROPLET VELOCITY AT I+1
      UDS(IP1)=UDS(IO)+C16*M1+C17*M3+C18*M4-C19*M5
      VDS(IP1)=VDS(IO)+C16*N1+C17*N3+C18*N4-C19*N5
C
C 5TH ORDER ESTIMATE OF POSITION AND VELOCITY
      XT=XDS(IO)+C20*K1+C21*K3+C22*K4-C23*K5+C24*K6
      YT=YDS(IO)+C20*L1+C21*L3+C22*L4-C23*L5+C24*L6
      UT=UDS(IO)+C20*M1+C21*M3+C22*M4-C23*M5+C24*M6
      VT=VDS(IO)+C20*N1+C21*N3+C22*N4-C23*N5+C24*N6
C
C DETERMINE DEFFERENCES IN 4TH AND 5TH ORDER ESTIMATES.
      XR=(XT-XDS(IP1))/DTS(IO)
      IF(DABS(XR).LT.1.D-70)XR=1.D-70
      YR=(YT-YDS(IP1))/DTS(IO)
      IF(DABS(YR).LT.1.D-70)YR=1.D-70
      UR=(UT-UDS(IP1))/DTS(IO)
      IF(DABS(UR).LT.1.D-70)UR=1.D-70
      VR=(VT-VDS(IP1))/DTS(IO)
      IF(DABS(VR).LT.1.D-70)VR=1.D-70
C CALCULATE STEP SIZE ADJUSTING FACTORS.
      XDEL=(EPS/DABS(XR))**.25DO
      YDEL=(EPS/DABS(YR))**.25DO
      UDEL=(EPS/DABS(UR))**.25DO
      VDEL=(EPS/DABS(VR))**.25DO
C ADJUST FOR LEAST PRECISE EQN.
      DMIN=DMIN1(XDEL,YDEL,UDEL,VDEL)
      RMAX=DMAX1(DABS(XR),DABS(YR),DABS(UR),DABS(VR))
      K=IO
      IF(RMAX.LE.EPS)K=IP1
      IF(DMINP.LT.1.DO)GOTO 200
      IF(DMIN.LT.1.DO)DTS(K)=0.9DO*DMIN*DTS(IO)
      IF(DMIN.GE.11.DO)DTS(K)=1.8DO*DTS(IO)
      IF(DMIN.GE.1.DO.AND.DMIN.LT.11.DO)DTS(K)=((DMIN-1.DO)/10.DO+1.DO)
     .*DTS(IO)*0.9DO
      GOTO 210
200   IF(DMIN.LE.0.5DO)DTS(K)=0.5DO*DTS(IO)
      IF(DMIN.GT.1.DO)DTS(K)=((DMIN-1.DO)/10.DO+1.DO)*0.9DO*DTS(IO)
      IF(DMIN.GT.0.5DO.AND.DMIN.LE.1.DO)DTS(K)=DMIN*0.9DO*DTS(IO)
```

```
  210 DMINP=DMIN
      IF(RMAX.GT.EPS)GOTO 100
C
C NEW ACCELERATIONS AT I+1
      CALL AIRVEL(XDS(IP1),YDS(IP1),UAS(IP1),VAS(IP1),5)
      CALL DRAG(UDS(IP1),VDS(IP1),UAS(IP1),VAS(IP1),CDS,RED(IP1),CD)
      CALL ACCN(UDS(IP1),VDS(IP1),UAS(IP1),VAS(IP1),RED(IP1),
     .CD,EQN,TS(I+1),0)
      IF(EQN.NE.2)RETURN
      CALL ACCN(UDS(IP1),VDS(IP1),UAS(IP1),VAS(IP1),RED(IP1),
     .CD,EQN,TS(I+1),1)
      RETURN
      END
C
C
      SUBROUTINE RK4(EQN,CDS)
C
C WRITTEN BY: M. OLESKIW  ON: 790926  LAST MODIFIED:801223
C
C INTEGRATE THE DROPLET EQNS  OF MOTION (IN X AND Y) USING THE 4TH
C   ORDER RUNGE-KUTTA TECHNIQUE.
C REF: BURDEN,R.L., J.D. FAIRES, & A.C. REYNOLDS (1978), NUMERICAL
C       ANALYSIS  P. 281   QA 297.B84
C
      DOUBLE PRECISION K1,L1,K2,L2,K3,L3,K4,L4,DTS(6),XDS(6),UDS(6),
     .YDS(6),VDS(6),AN(2,6),HT(2,6),
     .M1,M2,M3,M4,N1,N2,N3,N4,U1,U2,U3,V1,V2,V3,CD,RE,RED(6),
     .VAS(6),UAS(6),TS(500)
C
      INTEGER I,EQN,IM4,IM3,IM2,IM1,IO,IP1,CDS
C
      COMMON /INTEG/AN,HT/PV/XDS,YDS,UDS,VDS
     ./LOC/TS,DTS,I,IM4,IM3,IM2,IM1,IO,IP1
     ./REL/UAS,VAS,RED,CD
C
C IN  DTS=NON-DIMENSIONAL TIME STEP
C IN  I= PRESENT INDEX OF VECTORS XDS,UDS,...
C IN  EQN= CHOICE OF TERMS USED IN RHS OF ODE
C
      TS(I+1)=TS(I)+DTS(IO)
      K1=DTS(IO)*UDS(IO)
      L1=DTS(IO)*VDS(IO)
      M1=DTS(IO)*AN(1,IO)
      N1=DTS(IO)*AN(2,IO)
      CALL AIRVEL(XDS(IO)+K1/2.D0,YDS(IO)+L1/2.D0,U1,V1,4)
      CALL DRAG(UDS(IO)+M1/2.D0,VDS(IO)+N1/2.D0,U1,V1,CDS,RE,CD)
C
      K2=DTS(IO)*(UDS(IO)+M1/2.D0)
      L2=DTS(IO)*(VDS(IO)+N1/2.D0)
      CALL ACCN(UDS(IO)+M1/2.D0,VDS(IO)+N1/2.D0,U1,V1,RE,CD,EQN,
     .TS(I),0)
      M2=DTS(IO)*AN(1,IP1)
      N2=DTS(IO)*AN(2,IP1)
      CALL AIRVEL(XDS(IO)+K2/2.D0,YDS(IO)+L2/2.D0,U2,V2,4)
      CALL DRAG(UDS(IO)+M1/2.D0,VDS(IO)+N1/2.D0,U2,V2,CDS,RE,CD)
C
      K3=DTS(IO)*(UDS(IO)+M2/2.D0)
      L3=DTS(IO)*(VDS(IO)+N2/2.D0)
      CALL ACCN(UDS(IO)+M2/2.D0,VDS(IO)+N2/2.D0,U2,V2,RE,CD,EQN,
     .TS(I)+DTS(IO)/2.D0,0)
      M3=DTS(IO)*AN(1,IP1)
      N3=DTS(IO)*AN(2,IP1)
      CALL AIRVEL(XDS(IO)+K3,YDS(IO)+L3,U3,V3,4)
      CALL DRAG(UDS(IO)+M3,VDS(IO)+N3,U3,V3,CDS,RE,CD)
C
      K4=DTS(IO)*(UDS(IO)+M3)
      L4=DTS(IO)*(VDS(IO)+N3)
      CALL ACCN(UDS(IO)+M3,VDS(IO)+N3,U3,V3,RE,CD,EQN,
     .TS(I)+DTS(IO)/2.D0,0)
      M4=DTS(IO)*AN(1,IP1)
      N4=DTS(IO)*AN(2,IP1)
C
C NEW DROPLET POSITION AT I+1
      XDS(IP1)=XDS(IO)+(K1+2.D0*K2+2.D0*K3+K4)/6.D0
```

```
      YDS(IP1)=YDS(IO)+(L1+2.DO*L2+2.DO*L3+L4)/6.DO
C NEW VELOCITIES AT I+1
      UDS(IP1)=UDS(IO)+(M1+2.DO*M2+2.DO*M3+M4)/6.DO
      VDS(IP1)=VDS(IO)+(N1+2.DO*N2+2.DO*N3+N4)/6.DO
C NEW ACCELERATIONS AT I+1
      CALL AIRVEL(XDS(IP1),YDS(IP1),UAS(IP1),VAS(IP1),5)
      CALL DRAG(UDS(IP1),VDS(IP1),UAS(IP1),VAS(IP1),CDS,RED(IP1),CD)
      CALL ACCN(UDS(IP1),VDS(IP1),UAS(IP1),VAS(IP1),RED(IP1),CD,EQN,
     .TS(I+1),O)
      DTS(IP1)=DTS(IO)
      IF(EQN.NE.2)RETURN
C
      CALL ACCN(UDS(IP1),VDS(IP1),UAS(IP1),VAS(IP1),RED(IP1),CD,EQN,
     .TS(I+1),1)
      RETURN
      END
C
C
      SUBROUTINE PC4(EQN,CDS)
C
C WRITTEN BY: M. OLESKIW  ON: 800122 LAST MODIFIED: 801223
C
C INTEGRATE EQNS. OF MOTION USING THE 4TH ORDER PREDICTOR-
C   CORRECTOR METHOD
C REF: BURDEN, R.L., J.D. FAIRES, & A.C. REYNOLDS (1978),
C       NUMERICAL ANALYSIS    QA 297.B84     P.266
C       HAMMING, R.W. (1973), NUMERICAL METHODS FOR SCIENTISTS &
C       ENGINEERS, 2ND ED.,    QA 297.H28    CHAPS. 22 & 23
C
      DOUBLE PRECISION XDS(6),UDS(6),AN(2,6),HT(2,6),YDS(6),
     .VDS(6),AO,A1,A2,BO,B1,B2,B3,
     .CO,C1,C2,DM1,DO,D1,D2,UPI,UCI,VPI,VCI,MUAS,MVAS,
     .PUDS,DTS(6),PVDS,MUDS,MVDS,CUDS,CVDS,UDSP1,VDSP1
     .,FMU,FMV,UST,VST,ER1,ER2,PXDS,PYDS,MXDS,MYDS,CXDS,CYDS
     .,UAS(6),VAS(6),RED(6),XPI,XCI,YPI,YCI,RE,CD,TS(500)
C
      INTEGER I,EQN,IM4,IM3,IM2,IM1,IO,IP1,CDS
C
      COMMON/INTEG/AN,HT/PV/XDS,YDS,UDS,VDS
     ./PCM/AO,A1,A2,BO,B1,B2,B3,CO,C1,C2,DM1,DO,D1,D2,
     .UPI,UCI,VPI,VCI,ER1,ER2,XPI,XCI,YPI,YCI,UST,VST
     ./LOC/TS,DTS,I,IM4,IM3,IM2,IM1,IO,IP1
     ./REL/UAS,VAS,RED,CD
C
C  IN EQN= CHOICE OF TERMS USED IN RHS OF ODE
C IN  CDS=TYPE OF DRAG COEFFICIENT TO BE USED.
C
      TS(I+1)=TS(I)+DTS(IO)
C
C THE PREDICTOR
      PXDS=AO*XDS(IO)+A1*XDS(IM1)+A2*XDS(IM2)
     .+DTS(IO)*(BO*UDS(IO)+B1*UDS(IM1)+B2*UDS(IM2)+B3*UDS(IM3))
      PYDS=AO*YDS(IO)+A1*YDS(IM1)+A2*YDS(IM2)
     .+DTS(IO)*(BO*VDS(IO)+B1*VDS(IM1)+B2*VDS(IM2)+B3*VDS(IM3))
      PUDS=AO*UDS(IO)+A1*UDS(IM1)+A2*UDS(IM2)
     .+DTS(IO)*(BO*AN(1,IO)+B1*AN(1,IM1)+B2*AN(1,IM2)+B3*AN(1,IM3))
      PVDS=AO*VDS(IO)+A1*VDS(IM1)+A2*VDS(IM2)
     .+DTS(IO)*(BO*AN(2,IO)+B1*AN(2,IM1)+B2*AN(2,IM2)+B3*AN(2,IM3))
C
C MODIFICATION OF THE PREDICTOR
      MXDS=PXDS-ER1*(XPI-XCI)
      MYDS=PYDS-ER1*(YPI-YCI)
      MUDS=PUDS-ER1*(UPI-UCI)
      MVDS=PVDS-ER1*(VPI-VCI)
      CALL AIRVEL(MXDS,MYDS,MUAS,MVAS,4)
      CALL DRAG(MUDS,MVDS,MUAS,MVAS,CDS,RE,CD)
      CALL ACCN(MUDS,MVDS,MUAS,MVAS,RE,CD,EQN,TS(I+1),O)
      FMU=AN(1,IP1)
      FMV=AN(2,IP1)
C
C THE CORRECTOR
      CXDS=CO*XDS(IO)+C1*XDS(IM1)+C2*XDS(IM2)
     +DTS(IO)*(DM1*MUDS+DO*UDS(IO)+D1*UDS(IM1)+D2*UDS(IM2))
      CYDS=CO*YDS(IO)+C1*YDS(IM1)+C2*YDS(IM2)
```

```
     .+DTS(IO)*(DM1*MVDS+DO*VDS(IO)+D1*VDS(IM1)+D2*VDS(IM2))
      CUDS=CO*UDS(IO)+C1*UDS(IM1)+C2*UDS(IM2)
     .+DTS(IO)*(DM1*FMU+DO*AN(1,IO)+D1*AN(1,IM1)+D2*AN(1,IM2))
      CVDS=CO*VDS(IO)+C1*VDS(IM1)+C2*VDS(IM2)
     .+DTS(IO)*(DM1*FMV+DO*AN(2,IO)+D1*AN(2,IM1)+D2*AN(2,IM2))
C
C FINAL VALUES
      XDS(IP1)=CXDS+ER2*(PXDS-CXDS)
      YDS(IP1)=CYDS+ER2*(PYDS-CYDS)
      UDS(IP1)=CUDS+ER2*(PUDS-CUDS)
      VDS(IP1)=CVDS+ER2*(PVDS-CVDS)
C
C NEW VALUES FOR ACCELERATION AT I+1
      CALL AIRVEL(XDS(IP1),YDS(IP1),UAS(IP1),VAS(IP1),5)
      CALL DRAG(UDS(IP1),VDS(IP1),UAS(IP1),VAS(IP1),CDS,RED(IP1),CD)
      CALL ACCN(UDS(IP1),VDS(IP1),UAS(IP1),VAS(IP1),RED(IP1),CD,EQN,
     .TS(I+1),0)
      UDSP1=AN(1,IP1)
      VDSP1=AN(2,IP1)
      IF(EQN.NE.2)GOTO 100
      CALL ACCN(UDS(IP1),VDS(IP1),UAS(IP1),VAS(IP1),RED(IP1),CD,EQN,
     .TS(I+1),1)
C
C CALCULATE STABILITY INDICES
100   UST=(FMU-UDSP1)/(MUDS-UDS(IP1))
      VST=(FMV-VDSP1)/(MVDS-VDS(IP1))
      XPI=PXDS
      XCI=CXDS
      YPI=PYDS
      YCI=CYDS
      UPI=PUDS
      UCI=CUDS
      VPI=PVDS
      VCI=CVDS
      DTS(IP1)=DTS(IO)
      RETURN
      END
END OF FILE
```

A facsimile catalog card in Library of Congress MARC format is reproduced below.

Lozowski, Edward P.

Computer modeling of time-dependent rime icing in the atmosphere / by Edward P. Lozowski and Myron M. Oleskiw. Hanover, N.H.: U.S. Cold Regions Research and Engineering Laboratory; Springfield, Va.: available from National Technical Information Service, 1983.

vi, 31 p., illus.; 28 cm. (CRREL Report 83-2.)

Bibliography: p. 20.

1. Airfoils. 2. Computerized simulation. 3. Ice formation. 4. Rime ice. I. Oleskiw, Myron M. II. United States. Army. Corps of Engineers. III. Cold Regions Research and Engineering Laboratory, Hanover, New Hampshire. IV. Series: CRREL Report 83-2.

4-

DT