DOE/ER/03077-273

Courant Mathematics and
Computing Laboratory

U.S. Department of Energy

The Method of Complex Characteristics for Design of Transonic Blade Sections

M. R. Bledsoe

Research and Development Report

Supported by the Applied Mathematical Sciences
subprogram of the Office of Energy Research,
U.S. Dept. of Energy under Contract DE-AC02-76ER03077;
National Science Foundation Grant No. DMS-8320430;
and NASA-Ames Research Center Grant No. NAG 2-345

Mathematics and Computers

June 1986

NEW YORK UNIVERSITY

UNCLASSIFIED

Courant Mathematics and Computing Laboratory

New York University

DOE/ER/03077-273
UC-32
Mathematics and Computers

THE METHOD OF COMPLEX CHARACTERISTICS FOR DESIGN OF TRANSONIC BLADE SECTIONS

M. R. Bledsoe

June 1986

Supported by the Applied Mathematical Sciences subprogram of the Office of Energy Research, U. S. Department of Energy under Contract No. DE-AC02-76ER03077; National Science Foundation Grant No. DMS-8320430; and NASA-Ames Research Center Grant No. NAG 2-345

UNCLASSIFIED

DISCLAIMER

Printed in U.S.A.

Available from

National Technical Information Service
U.S. Department of Commerce
5285 Port Royal Road
Springfield, VA 22161

PREFACE

A variety of computational methods have been developed to obtain shockless or near shockless flow past two-dimensional airfoils. Our approach has been the method of complex characteristics, which determines smooth solutions to the transonic flow equations based on an input speed distribution. The approach is to find solutions of the partial differential equation

$$(c^2-u^2)\,\Phi_{xx} - 2uv\,\Phi_{xy} + (c^2-v^2)\,\Phi_{yy} = 0$$

by the method of complex characteristics. Here Φ is the velocity potential, $\nabla\Phi = (u,v)$, and c is the local speed of sound. Our method consists of noting that the coefficients of the equation are analytic, so that we can use analytic continuation, conformal mapping, and a spectral method in the hodograph plane to determine the flow.

After complex extension we obtain canonical equations for Φ and for the stream function Ψ as well as an explicit map from the hodograph plane to complex characteristic coordinates. In the subsonic case, a new coordinate system is defined in which the flow region corresponds to the interior of an ellipse. We construct special solutions of the flow equations in these coordinates by solving characteristic initial value problems in the ellipse with initial data defined by the complete system of Chebyshev polynomials. The condition $\Psi = 0$ on the boundary of the ellipse is used to determine the series representation of Φ and Ψ. The map from the ellipse to the complex flow coordinates is found from data specifying the speed q as a function of the arc length s. The transonic problem for shockless flow becomes well posed after

appropriate modifications of this procedure. The nonlinearity of the problem is handled by an iterative method that determines the boundary value problem in the ellipse and the map function in sequence.

We have implemented this method as a computer code to design two-dimensional cascades and isolated wing sections. A particular feature of this approach concerns the design of compressor blades with high solidity. We have been able to obtain gap-to-chord ratios as low as .42 with the computer program presented here.

The first portion of this report is devoted to mathematical theory. After a brief introduction to the problem in Chapter 1, Chapter 2 presents general results from fluid mechanics. Chapter 3 is an account of the method of complex characteristics, including a description of the particular spaces and coordinates, conformal transformations, and numerical procedures that are used. The remainder of the report concerns the operation of the computer program COMPRES. Chapter 4 presents examples of blade sections designed with the code, and Chapter 5 is a manual for users of our program. The glossaries presented in Chapter 6 provide additional information which may be helpful to users. Finally, in Chapter 9 we present a listing of the computer program in Fortran, including numerous comment cards.

I would like to make a few acknowledgements. Professor Paul Garabedian suggested this project and has guided me with untiring patience. Dr. Frances Bauer and Dr. Jose Sanz have contributed to various stages of the work. Finally, I would like to thank my family for their support.

TABLE OF CONTENTS

LIST OF FIGURES

I. INTRODUCTION

1.1 The physical problem

In recent years the development of numerical techniques for computing solutions to the differential equations of transonic aerodynamics has made useful mathematical models available to aeronautical engineers engaged in the study of physical problems. These models can replace costly and difficult wind tunnel experiments, especially in the case of two-dimensional flow. Thus they have fostered the rapid and efficient development of supercritical wing technology.

When attempting to model a physical system numerically, the mathematical formulation which results may have a significance which is independent of the original purpose of the investigation. Still, knowledge of the physical problem guides the researcher in approaching the solution and interpreting the results. The motivation for our problem involves the flow of air through an arrangement of rotating axial blades. These may be compressors which are designed to increase the pressure and density of the fluid passing through them, or turbines which lower these quantities and increase the speed. Such systems are a basic component of jet engines, and hence are of crucial importance to modern aerodynamics. In addition, we will consider the case of an isolated wing. When the maximum speed past these bodies exceeds sonic speed, the wave drag caused by shock waves in the supersonic flow

region must be minimized. We wish to design blades for which this low drag occurs.

Figure 1 shows a plot of the lift coefficient C_L and of the drag coefficient C_D versus the free stream Mach number M_∞ for a typical wing section [17]. The drag coefficient does not increase just above the critical Mach number M_{CR} because the shock wave is weak, but at the drag rise Mach number M_D there is a jump in the size of the supersonic zone and drag rise begins. The lift coefficient grows between M_D and the value of M_∞ at which the boundary layer separates. In this interval it can be seen that $(M_\infty L)/D$ will have a maximum. This ratio, where L is the lift and D is the drag, can be used as a measure of airfoil efficiency. At the maximum M_∞ will be subsonic while speeds are supersonic along a portion of the airfoil, so that the flow will be transonic.

The proper design of the profile can delay drag rise to higher Mach numbers, resulting in maximum efficiency at greater speed. In particular, airfoils which admit shockless flows at certain operating conditions delay drag rise at nearby conditions. Hence we consider the problem of determining a series of blades which allow shockless transonic flow at particular operating conditions. We can expect such systems to admit flows with minimal drag for a certain range of boundary values.

We turn now to the simplifications necessary to generate a mathematical model of compressor and turbine flow. First, let us consider the geometry of the problem. The systems of blades are mounted on a rotating cylinder in a gas flowing in the direction of the axis. A logarithmic transformation will map this configuration into a

series of blades which are stacked vertically. If the z-axis lies in the direction of the span of the blades, the (x,y)-plane becomes orthogonal to them. Let us suppose that the flow lies in the (x,y)-plane, which is a widely used approximation. We thus arrive at the problem of plane compressible flow past a periodic array of airfoils in cascade.

We assume that the airfoils are streamlined so that viscous effects are confined to the immediate vicinity of the profile. In this case the flow outside the boundary layer can be found using the partial differential equations describing inviscid fluid motion. This solution can be then used to calculate a boundary layer correction from which the profile may be obtained. We also assume that time dependent effects are negligible, so that the flow is steady. As a result, our mathematical problem concerns the determination of the steady shockless flow of an inviscid compressible fluid past an individual airfoil or a cascade or airfoils in the (x,y) plane.

1.2 The theory of shockless flow

The mathematical theory of the transonic flow equations is difficult, and few theorems on the existence and uniqueness of solutions have been proven. Whenever the flow is not entirely subsonic, smooth solutions to the equations are exceptional and cannot usually be expected. Generally, the solution contains shocks, and shock conditions and an entropy inequality must be satisfied. Numerical studies of the equations, however, suggest that there may exist a unique weak solution to the direct problem of flow past a given profile. Any smooth solution to the equations would then coincide with this unique weak solution.

The fact that solutions of the transonic equations are not in general smooth was demonstrated in 1956 by Morawetz [14] following a decade of controversy. At that time, however, the physical significance of shockless flows could not be evaluated. Then in the 1960s Pearcey [19] performed experiments that exhibited nearly shockless flow past transonic airfoils having a suction pressure peak near the leading edge. This was followed by Whitcomb's discovery of the supercritical wing [27], which has a large supersonic zone and minimal boundary layer separation. Later Spee and Uijlenhoet [24] did wind tunnel tests of a symmetric shockless airfoil designed by Nieuwland [18] using the hodograph method. They obtained essentially shockless flow that agreed with the mathematical predictions. These developments confirmed the existence of nearly shockless flows in nature and established their effectiveness in reducing drag.

Further advances in transonic flow theory resulted from the development of finite difference schemes to solve the partial differential equations of motion. Most important was the method of Murman and Cole [15], which captures shocks in the supersonic region through the use of an artificial viscosity term obtained by retarding difference operators in the direction of the flow. Such schemes are now used routinely to analyze flows in two and three dimensions past wings and wing body combinations [3]. They show that drag and shock strength vary continuously with the shape of the profile and with operating conditions. Moreover, the solutions confirm that profiles which are shockless at given design conditions exhibit weaker shocks at off-design conditions than do other airfoils.

In the light of these developments the genuine significance of shockless flow and the importance of methods for computing supercritical airfoils are recognized. Such methods are generally inverse methods in which the profile is found from the solution to the flow equations as the locus of points where the stream function vanishes. Our purpose here will be to describe one such method, the method of complex characteristics [2,4].

1.3 Design of supercritical airfoils

We review briefly methods of solving the transonic flow equations that result in solutions which are smooth or have weak shocks. Because arbitrary boundary condtions will not yield such solutions, these are design methods in which the profile is determined in the course of the computation. The reader is referred to the literature for a general survey of techniques for finding transonic flows [17].

Nieuwland [18] developed a technique for computing shockless airfoils in which the stream function is determined as a linear combination of solutions found by separation of variables in the hodograph plane. His work resulted in the first realistic symmetric shockless airfoils, whose flow properties were then verified by the wind tunnel tests of Spee and Uijlenhoet [24].

Some methods [5,11] are based on the small disturbance equation. The solution is expanded in a parameter describing the thickness of the airfoil, where the zero order approximation is a slit. A nonlinear differential equation results for the first order terms of the expansion. The desired pressure distribution along the profile is given as a boundary condition, and actual coordinates are determined from the resulting velocity components. This method has been extended to the full 3-dimensional problem [1].

Further methods solve the inverse problem for the full potential equation with a free boundary. Carlson [6] uses a prescribed distribution to obtain Dirichlet boundary conditions for the velocity potential in Cartesian coordinates. Tranen [26] uses the NYU analysis

code [3] to iterate between design and analysis computations. At each cycle the prescribed pressure distribution is modified to achieve convergence.

McFadden [13] has also used the NYU analysis code to design airfoils with weak shocks by an iterative method which uses a prescribed speed distribution q(s) to approximate the conformal map from the airfoil to the unit circle. The solution computed by the analysis code is then used to improve the approximation. This method has been extended to three dimensions and applied to construct swept wings with low levels of wave drag in transonic flow [10].

Sobieczky, Fung and Seebass [23] find shockless flow in two and three dimensions by introducing a fictitious gas law. When the flow becomes supersonic, the equation of state is changed to make the equation for the velocity potential remain elliptic. Standard difference schemes are applied, and a correct solution emerges in the subsonic flow region, but not in the supersonic zone. Consequently the correct hyperbolic equations must be solved there along real characteristics extending from the sonic line. The body is determined by tracing stream lines. This technique is similar to one exploited by Shiffman in 1952 to prove existence theorems [22].

The method of complex characteristics developed by Bauer, Garabedian and Korn [2,4] solves the equations in the hodograph plane by extending all variables into the complex domain, where the notion of type is no longer significant. The subsonic flow region is mapped into a unit circle in the complex domain and a series of characteristic initial value problems are solved there. Smooth solutions are found by prescribing a relationship on the circumference of the circle between

the speed q of the flow and the arc length s along the airfoil. In the complex domain this leads to a well posed boundary value problem even for transonic flow. The profile is obtained after the solution is determined in the real supersonic zone. This method has the advantage that it can be used to design a cascade of shockless airfoils. Since the present work is an extension of this method, it will be described in detail in Chapter 3.

Sanz [21] has modified the method of complex characteristics by introducing a new coordinate transformation so that the supersonic flow region maps into an ellipse. This allows for the computation of flows through cascades of compressor airfoils with gap-to-chord ratios G/C as low as 0.5, which was beyond the scope of the previous code. Our new version of the design code also computes a solution using elliptic coordinates. In addition, modifications of the paths of integration and of the quantities computed along them allow solutions to be obtained for a wider range of boundary data.

II. MATHEMATICAL BACKGROUND

2.1 The differential equations of transonic flow

We shall be concerned with the partial differential equations governing the two-dimensional steady flow of a compressible inviscid fluid [7]. These are

$$(\rho u)_x + (\rho v)_y = 0$$

$$\rho u u_x + \rho v u_y + p_x = 0$$

$$\rho u v_x + \rho v v_y + p_y = 0$$

$$u S_x + v S_y = 0$$

where x and y are the rectangular coordinates in the physical plane, u and v are the components of velocity, ρ is the density, p is the pressure, and S is the entropy. We assume an equation of state of the form $p = A \rho^{\gamma}$, where A is a known function of S and γ is the gas constant. We denote by q the speed of the flow, by c the local speed of sound, and by $M = q/c$ the Mach number.

Transonic solutions of the equations of motion are weak solutions containing shock waves on which only integral forms of the differential equations are satisfied. Across these curves there are jump conditions which specify conservation of mass, momentum and energy, and there is an entropy inequality which assures that the shock is compressive. This jump in entropy is third order in the shock strength. Since the flow past wing sections is uniform far from the profile, the entropy is

constant there, and it will remain constant in the region where the flow is continuous. Because we are concerned with shockless flows, we may therefore assume that the entropy is constant everywhere and that the flow is isentropic throughout. For flows containing shocks, such an assumption is equivalent to using alternate shock conditions in which the normal component of momentum is not conserved. The jump in this quantity can be considered an approximation of the wave drag that the shock exerts on the airfoil.

In the isentropic case the flow is irrotational and satisfies the equation

$$u_y - v_x = 0$$

We have then a velocity potential $\Phi(x,y)$ satisfying

$$\Phi_x = u \ , \ \Phi_y = v$$

and a stream function $\Psi(x,y)$ such that

$$\Psi_x = -\rho v \ , \ \Psi_y = \rho u$$

These relations comprise a pair of generalized Cauchy-Riemann equations for Φ and Ψ with coefficients defined by Bernoulli's law

$$\frac{q^2}{2} + \frac{c^2}{\gamma - 1} = \frac{1}{2} \frac{\gamma + 1}{\gamma - 1} c_*^2$$

where c_* is the critical speed at which $M^2 = 1$. These equations are

elliptic when $M^2 < 1$ and hyperbolic when $M^2 > 1$. Alternatively we can obtain the single second order differential equation

$$(c^2-u^2)\,\Phi_{xx} - 2uv\,\Phi_{xy} + (c^2-v^2)\,\Phi_{yy} = 0$$

for Φ.

These nonlinear formulations are convenient for many purposes, including finite difference methods for capturing shocks. For design or inverse problems, however, it is more natural to apply the hodograph transformation, in which the dependent and independent variables are interchanged and a linear system of equations results. For such a hodograph formulation, boundary conditions are based on some property of the flow, and the profile is then determined from the solution as the streamline $\Psi = 0$.

One version of the hodograph flow equations is given by

$$x_v - y_u = 0$$

$$(c^2-u^2)\,y_v + uv(x_v + y_u) + (c^2-v^2)\,x_u = 0$$

Here the Jacobian $J = x_u y_v - x_v y_u$ must be nonzero in order to represent a physically reasonable flow. In the subsonic region J can vanish only at isolated points, as is seen from the expression

$$-(c^2-u^2)\,J = (c^2-u^2)\,x_v^2 + 2uv\,x_u\,x_v + (c^2-v^2)\,x_u^2\ .$$

For supersonic flow J can be nontrivially zero along a curve in the (u,v)-plane whose image in the physical plane is a limiting line along

which different branches of the solutions $u(x,y)$ and $v(x,y)$ are joined [7]. Although such curves cannot be allowed to occur in the flow field, they may exist in the region of the (x,y)-plane enclosed by the profile. Once a hodograph solution has been found, x and y may be determined along the profile $\Psi = 0$ from the expression

$$x + iy = \int \frac{d\Phi + i\,\frac{d\Psi}{\rho}}{u - iv}$$

2.2 Characteristic coordinates and canonical equations

An equivalent formulation of the system in the hodograph plane is given by the Chaplygin equations

$$\Phi_\theta = \frac{q}{\rho}\Psi_q \quad , \quad \Phi_q = \frac{M^2-1}{\rho q}\Psi_\theta$$

where θ is the angle of the flow. For the hyperbolic case $M^2 > 1$, these can be transformed by introducing characteristic directions in the (q,θ)-plane. These comprise a curvilinear net of characteristic coordinates (ξ,η) that are constant on the two independent sets of real characteristics. In the (u,v)-plane they describe epicycloids cusping at the sonic line $M^2 = 1$. Under this coordinate change we obtain two equations, in each of which Φ and Ψ are differentiated in only one characteristic direction. This formulation allows for the solution of characteristic initial value problems in which compatible initial data for Φ and Ψ are given along two characteristics $\xi = \xi_c$ and $\eta = \eta_c$. The solution is found in a quadrant bounded by these lines.

In the elliptic case $M^2 < 1$, the characteristic directions are complex and conjugate. Hence to apply characteristic coordinates to transonic flow problems, we continue all dependent and independent variables analytically into the complex domain. This is possible because the system for Φ and Ψ is analytic in all arguments. Complex characteristic coordinates $\xi(q,\theta)$ and $\eta(q,\theta)$ then exist everywhere and complex analytic solutions Φ and Ψ to characteristic initial value

problems can be found throughout the four-dimensional complex (ξ,η)-space.

Defining $r = \int \frac{\sqrt{1-M^2}}{q} dq$ we may introduce the characteristic coordinates $\xi = \theta + ir$, $\eta = \theta - ir$. The system of canonical equations for the characteristics and for Φ and Ψ then becomes

$$\theta_\xi = \frac{i\sqrt{1-M^2}}{q} q_\xi \quad , \quad \theta_\eta = \frac{-i\sqrt{1-M^2}}{q} q_\eta$$

$$\Phi_\xi = \frac{i\sqrt{1-M^2}}{\rho} \Psi_\xi \quad , \quad \Phi_\eta = \frac{-i\sqrt{1-M^2}}{\rho} \Psi_\eta$$

These characteristic coordinates are not unique. If ξ and η are characteristic coordinates and if $\tilde{\xi} = f(\xi)$ and $\tilde{\eta} = g(\eta)$, where f and g are complex analytic functions, then $\tilde{\xi}$ and $\tilde{\eta}$ are also characteristic coordinates. We shall denote by (s,t) the particular set of characteristic coordinates given by

$$s = h(q)\, e^{-i\theta} \quad , \quad t = h(q)\, e^{i\theta}$$

where $h = e^r$.

The method of complex characteristics breaks down on the two-dimensional sonic surface, resulting in a singularity of the characteristic coordinate transformation. The difference scheme used to obtain solutions becomes singular at points where $M^2 = 1$ and is ill-conditioned at nearby points. Hence, all computational grids must be constructed to avoid this surface.

Although all variables have been extended into the complex domain, we are only interested in solutions Φ and Ψ in the real (q,θ)-plane. Hence we wish to know where this plane lies in the (s,t)-coordinate system. First, let us state a simple result:

LEMMA. We have $s = \bar{t}$ if and only if h and θ are real.

PROOF. If $s = \bar{t}$, then $h\, e^{-i\theta} = \bar{h}\, e^{-i\bar{\theta}}$, so that $h/\bar{h} = e^{-i(\bar{\theta}-\theta)}$. Because $|h/\bar{h}|=1$ and $\bar{\theta} - \theta$ is imaginary, it follows that $\bar{\theta} - \theta$ vanishes, so that θ is real. In addition, $h/\bar{h} = 1$, and therefore h is also real. The proof in the opposite direction is immediate.

For the subsonic case, the quantity h is real when q is real so that the real subsonic domain lies in the plane $s = \bar{t}$. For $M^2 > 1$, however, h is not real, so the real supersonic domain lies outside of the plane $s = \bar{t}$ on the complex surface defined by $|s| = |t| = |h(c_*)|$. This surface attaches to the plane $s = \bar{t}$ along the curve in (s,t)-space which is the image of the sonic line.

A major problem with the hodograph method is that the flow region in the velocity plane is unknown and may have a complicated geometry. Instead of using complex characteristic coordinates (s,t), we may perform a conformal mapping which eliminates this difficulty. By the Riemann mapping theorem, a second set of characteristic coordinates ξ and η can be determined by sending a fixed region in the plane $\xi = \bar{\eta}$ to the subsonic flow region in the plane $s = \bar{t}$. If we require that the transformation has the form $s = f(\xi)$, $t = f(\bar{\eta})$, the plane $s = \bar{t}$ will correspond to the plane $\xi = \bar{\eta}$. The new coordinates will be found by solving a nonlinear boundary value problem for f.

Because the solution to be constructed lies in the four-dimensional space of two complex variables, characteristic initial value problems for Φ and Ψ are solved by imposing initial data for Ψ on two characteristic planes $\xi = \xi_C$ and $\eta = \eta_C$. Let this initial data be given by

$$\Psi(\xi_C,\eta) = F_1(\eta) , \quad \Psi(\xi,\eta_C) = F_2(\xi)$$

where the analytic functions F_1 and F_2 satisfy the compatibility condition $F_1(\eta_C) = F_2(\xi_C)$. Then Φ can be determined on the characteristic initial planes from the canonical equations. In order to find the solution at a point (ξ,η) of complex space, paths of integration originating at the point (ξ_C,η_C) and terminating at (ξ_C,η) and (ξ,η_C) must be drawn in the two initial planes. The solution can then be found from the initial data by constructing a grid bounded by the two paths and applying a finite difference scheme.

The existence and uniqueness of solutions to characteristic initial value problems are proven by formulating the system for Φ and Ψ as a pair of integral equations. Setting $\tau = \frac{i\sqrt{1-M^2}}{\rho}$, the integral form of the canonical equations becomes

$$\Phi(\xi,\eta) = \Phi(\xi_C,\eta) + \int_{\xi_C}^{\xi} \tau(\hat{\xi},\eta)\, \Psi_\xi(\hat{\xi},\eta)\, d\hat{\xi}$$

$$\Phi(\xi,\eta) = \Phi(\xi,\eta_C) - \int_{\eta_C}^{\eta} \tau(\xi,\hat{\eta})\, \Psi_\eta(\xi,\hat{\eta})\, d\hat{\eta}$$

Integrating by parts we obtain

$$\Phi(\xi,\eta) = \Phi(\xi_C,\eta) + \tau(\xi,\eta)\,\Psi(\xi,\eta) - \tau(\xi_C,\eta)\,\Psi(\xi_C,\eta) - \int_{\xi_C}^{\xi} \tau_\xi(\hat{\xi},\eta)\,\Psi(\hat{\xi},\eta)\,d\hat{\xi}$$

$$\Phi(\xi,\eta) = \Phi(\xi,\eta_C) - \tau(\xi,\eta)\,\Psi(\xi,\eta) + \tau(\xi,\eta_C)\,\Psi(\xi,\eta_C) + \int_{\eta_C}^{\eta} \tau_\eta(\xi,\hat{\eta})\,\Psi(\xi,\hat{\eta})\,d\hat{\eta}$$

These can be combined to form a single integral equation for Ψ,

$$\Psi(\xi,\eta) = \frac{1}{2(\tau(\xi,\eta))}\left[\Phi(\xi,\eta_C) - \Phi(\xi_C,\eta) + \tau(\xi,\eta_C)\,\Psi(\xi,\eta_C) + \tau(\xi_C,\eta)\,\Psi(\xi_C,\eta)\right.$$

$$\left. + \int_{\eta_C}^{\eta} \tau_\eta(\xi,\hat{\eta})\,\Psi(\xi,\hat{\eta})\,d\hat{\eta} + \int_{\xi_C}^{\xi} \tau_\xi(\hat{\xi},\eta)\,\Psi(\hat{\xi},\eta)\,d\hat{\xi}\right]$$

The solution is found by Picard iteration. The iteration is initialized by setting

$$\Psi^{(o)}(\xi,\eta) = F_2(\xi) + F_1(\eta) - F_1(\eta_C)$$

and determining $\Phi^{(o)}$ from the differential equations. Using standard techniques, it is then shown that each pair of iterates $\Phi^{(n)}$ and $\Psi^{(n)}$ satisfies the data on the characteristic initial planes and that the iterative scheme converges uniformly in some neighborhood of (ξ_C,η_C) to a unique solution. Each pair of iterates is analytic so that by Cauchy's integral theorem, the integrals in the above expression for Ψ are path independent. As a result the solution is analytic and independent of the particular pair of paths of integration chosen to connect the point (ξ_C,η_C) with the points (ξ,η_C) and (ξ_C,η).

2.3 The reflection principle in two complex variables

We review here results concerning analytic functions of two complex variables which we shall use in solving the canonical equations by the method of complex characteristics.

DEFINITION. An analytic function $F(\xi,\eta)$ of two complex variables is called a real function if F is real in the plane $\xi = \bar{\eta}$.

Such functions become analogous to real functions in the real (x,y)-plane if one makes the substitution $z = x + iy$, $\bar{z} = x-iy$ and uses the complex coordinates $\xi = z$ and $\eta = \bar{z}$. The Schwarz reflection principle for these functions may be stated as follows:

THEOREM. $F(\xi,\eta)$ is a real function in the above sense if and only if $F(\xi,\eta) = \overline{F(\bar{\eta},\bar{\xi})}$.

PROOF. Let

$$I(\xi,\eta) = F(\xi,\eta) - \overline{F(\bar{\eta},\bar{\xi})}$$

If F is a real function we have $F(\xi,\eta) = \overline{F(\bar{\eta},\bar{\xi})}$ in the plane $\xi = \bar{\eta}$ by hypothesis, so I vanishes there. Since I is an analytic function of the two complex variables ξ and η, it must be identically zero. The converse is immediate.

We shall choose our complex characteristic coordinates so that the physical flow region lies in the plane $\xi = \bar{\eta}$ for subsonic flow, and we shall solve characteristic initial value problems to obtain Φ and Ψ

there. Because of the reflection principle, the computation of real solutions to characteristic initial value problems will simplify in a manner to be described. Hence we should like the solutions to the canonical equations to be linear combinations of real functions. We must therefore know how to set up characteristic initial value problems which will result in real functions. We shall show that for subsonic flow the solutions of our system for Φ and Ψ are real when the characteristic initial data have an appropriate symmetry property.

Let us choose characteristic initial planes $\xi = \xi_C$ and $\eta = \eta_C$ through the real point $\xi_C = \overline{\eta_C}$. The symmetry property required of the initial data is given by

$$\Psi(\xi,\eta_C) = F(\xi) \quad , \quad \Psi(\xi_C,\eta) = \overline{F(\bar{\eta})}$$

where F is an analytic function which is real at (ξ_C,η_C). Note that for subsonic flow $i\tau(\xi,\eta)$ is a real function, and hence $\tau(\xi,\eta) = \overline{-\tau(\bar{\eta},\bar{\xi})}$. Moreover, since

$$\frac{d}{d\eta}\,\overline{F(\bar{\eta})} = \overline{\frac{d}{/d\bar{\eta}/}\,F(\bar{\eta})}$$

we have from the integral form of the canonical equations

$$\Phi(\xi_C,\eta) = \overline{\Phi(\bar{\eta},\eta_C)}$$

so that Φ has the same symmetry property on the initial characteristics. We now prove the following

THEOREM. Let $\Phi(\xi,\eta)$, $\Psi(\xi,\eta)$ be a solution of a characteristic initial value problem for the flow equations with data satisfying the symmetry requirements

$$\Phi(\xi_C,\eta) = \overline{\Phi(\bar{\eta},\eta_C)} \quad , \quad \Psi(\xi_C,\eta) = \overline{\Psi(\bar{\eta},\eta_C)}$$

Then $\Phi(\xi,\eta)$ and $\Psi(\xi,\eta)$ are real functions.

PROOF. Observe that $\Phi(\xi,\eta)$, $\Psi(\xi,\eta)$ and $\overline{\Phi(\bar{\eta},\bar{\xi})}$, $\overline{\Psi(\bar{\eta},\bar{\xi})}$ satisfy the same differential equations with the same characteristic initial values. Hence they must be equal because of the uniqueness theorem for the solution.

2.4 Boundary conditions

There are both direct and inverse approaches to finding the flow past an airfoil, and different boundary conditions are associated with each problem. In the direct problem the coordinates x and y of the airfoil are specified as functions of the arc length s measured from the trailing edge on the lower surface to the trailing edge on the upper surface. The frame of reference is chosen so that the cascade of airfoils is at rest. The inlet speed at infinity is given, and defining the stagger angle β as the angle between the cascade of blades and the vertical, the blades are oriented so that $\beta = 0$. The Mach number at some fixed speed q must also be specified, which determines c_*. Since the flow must be tangential to the boundary, the stream function vanishes there,

$$\Psi(x(s), y(s)) = 0$$

The circulation Γ will be determined by the Kutta-Joukowski condition asserting that the velocity must be finite at the trailing edge. This formulation of the problem can be solved by standard finite difference methods [3], and results in weak solutions which generally contain a number of shocks in the supersonic zone.

In this report, we will use the inverse approach to the problem. This results in a free boundary problem in which the profile will be determined as the locus of points where $\Psi = 0$. The Mach number at a fixed speed is given, and the coordinate system is again rotated so

that $\beta = 0$. But now we prescribe, instead of the shape of the profile, the speed distribution $q(s)$ to be achieved along the boundary. When $\max q(s) > c_*$, the flow will be transonic and the flow equations will be of mixed type. In prescribing q, we determine

$$\Phi(s) = \int q(s)\, ds$$

as a function of s. We can also find the circulation Γ by integrating q around the entire profile. For transonic flow, this problem is not well posed, so we cannot expect that an appropriate profile will always be found having everywhere the prescribed pressure distribution. In order to design shockless airfoils by this inverse method, we develop a way of modifying the prescribed data in the supersonic zone to obtain smooth flow past a physically reasonable airfoil.

III. THE METHOD OF COMPLEX CHARACTERISTICS

3.1 Conformal mapping

The method of complex characteristics is concerned with analytic solutions of the canonical system

$$\Phi_\xi = \tau\Psi_\xi \quad , \quad \Phi_\eta = -\tau\Psi_\eta$$

$$\theta_\xi = \frac{i\sqrt{1-M^2}}{q} q_\xi \quad , \quad \theta_\eta = \frac{-i\sqrt{1-M^2}}{q} q_\eta$$

We shall be interested in the case where the boundary values of Ψ and q are assigned on an ellipse in the ξ-plane. The solution of the boundary value problem will be found by a spectral method. In the transonic case the boundary data must be modified to make the problem well posed. In our exposition we shall first treat the subsonic case and then present modifications which allow us to find shockless transonic flows.

All quantities have been extended into the complex domain by analytic continuation. Using the conjugate characteristic coordinates presented in Section 2.2,

$$s = h(q)e^{-i\theta} \quad , \quad t = h(q)\, e^{i\theta}$$

the real subsonic flow region corresponds to the plane $s = \bar{t}$. In order

that Φ and Ψ be real-valued in this region the reflection principle of Section 2.3 can be used to construct solutions to the canonical equations which are real functions. However, it will be more convenient in practice to use the fact that for arbitrary complex solutions Φ and Ψ, $\mathrm{Re}(\Phi)$ and $\mathrm{Re}(\Psi)$ satisfy the flow equations over the real domain.

Let us introduce a conformal map

$$s = \frac{\xi - \xi_0}{k\xi - \xi_0} e^{f(\xi)} \quad , \qquad t = \frac{\eta - \eta_0}{\overline{k\eta} - \eta_0} \overline{e^{f(\eta)}}$$

sending a fixed domain in the plane $\xi = \overline{\eta}$ into the flow region in the plane $s = \overline{t}$. Here $\xi_0 = \overline{\eta_0}$ is the point on the ellipse boundary corresponding to stagnation, and k is a constant of absolute value less than 1 which is introduced to improve convergence. The fixed domain should possess a complete system of analytic functions which can be used to represent the flow. In an earlier version of the method the unit circle $|\xi| < 1$ was mapped onto the flow region and analytic functions were represented as power series [4]. We now replace the unit circle by an ellipse with foci at ± 1, where the Chebyshev polynomials form a complete orthogonal system. The ellipse can be identified with a circular ring in the z-plane under the transformation

$$\xi = \frac{1}{2}(z + \frac{1}{z})$$

identifying the straight line between the foci with the unit circle,

and an arbitrary point ξ with the pair of points z and $\frac{1}{z}$. Then the Chebyshev polynomials are expressed as

$$T_n(\xi) = \frac{1}{2}(z^n + \frac{1}{z^n})$$

Consider the Laurent series for a function in the ring taking identical values at z and $1/z$. This leads us to represent the analytic function $f(\xi)$ in the form

$$f(\xi) = \sum_{n=0}^{\infty} a_n T_n(\xi)$$

If $Re(f)$ is known at equally spaced points on the circle $|z| = R$, $R<1$, the coefficients of a truncated Chebyshev expansion for f within the ring $R<|z|<1/R$ can be determined using the fast Fourier transform. The boundary values of $Re(f)$ depend both on the assigned data for the speed q as a function of arc length and on the solution Φ in the ellipse. This will be discussed in more detail in the following section.

3.2 Solutions in the ellipse

In the ellipse in the ξ-plane or in the hodograph plane, there are points corresponding to the velocity at infinity. For a cascade let ξ_A denote the point in the plane $\xi = \overline{\eta}$ corresponding to the exit velocity and let ξ_B denote the point corresponding to the inlet velocity. At ξ_A we have a sink and at ξ_B a source, so that Φ and Ψ are singular there. A branch cut connects these points (cf. Figure 2) and the flow region is represented by an infinite-sheeted Riemann surface. The ellipse has been chosen for the design of compressors with low gap-to-chord ratios because it becomes necessary to locate ξ_A and ξ_B near the boundary. This leads to difficulties in constructing paths of integration unless the transformation to the hodograph plane redistributes mesh points appropriately.

The stream function Ψ possesses singularities at ξ_A and ξ_B which are derived from the fundamental solution of the linear partial differential equations in the hodograph plane [9]. For a cascade we have

$$\Psi(\xi,\eta) = (\Psi_1(\xi,\eta) + i\Psi_A)\log(\xi-\xi_A) + (\Psi_2(\xi,\eta) + i\Psi_B)\log(\xi-\xi_B) + \Psi_3(\xi,\eta)$$

where the Ψ_j are real analytic functions which are regular at ξ_A and ξ_B and Ψ_A and Ψ_B are real constants. Similarly,

$$\Phi(\xi,\eta) = (\Phi_1(\xi,\eta) + i\Phi_A)\log(\xi-\xi_A) + (\Phi_2(\xi,\eta) + i\Phi_B)\log(\xi-\xi_B) + \Phi_3(\xi,\eta)$$

We wish to construct characteristic initial value problems for the functions Ψ_j and Φ_j and to determine appropriate initial data.

Applying the canonical equations and equating the coefficients of singular terms, we obtain the following differential equations:

$$\Phi_{1\xi} = \tau\Psi_{1\xi} \quad , \quad \Phi_{1\eta} = -\tau\Psi_{1\eta}$$

$$\Phi_{2\xi} = \tau\Psi_{2\xi} \quad , \quad \Phi_{2\eta} = -\tau\Psi_{2\eta}$$

$$\Phi_{3\xi} = \tau\Psi_{3\xi} - \left[\frac{(\Phi_1+i\Phi_A)-\tau(\Psi_1+i\Psi_A)}{\xi-\xi_A} + \frac{(\Phi_2+i\Phi_B)-\tau(\Psi_2+i\Psi_B)}{\xi-\xi_B}\right], \quad \Phi_{3\eta} = -\tau\Psi_{3\eta}$$

In order that the first inhomogeneous term above be regular, the restriction $\Phi_1+i\Phi_A = \tau(\Psi_1+i\Psi_A)$ is imposed on the characteristic plane $\xi = \xi_A$. This together with the differential equations will determine Φ_1 and Ψ_1 on the plane $\xi = \xi_A$ up to the real constants Φ_A and Ψ_A. If, moreover, we wish Φ_1 and Ψ_1 to be real functions, we may impose symmetric data on the plane $\eta = \eta_A$, where $\xi_A = \overline{\eta_A}$, thus obtaining a characteristic initial value problem for Φ_1 and Ψ_1. Similarly, we obtain a characteristic initial value problem for real functions Φ_2 and Ψ_2 with symmetric initial data on the planes $\xi = \xi_B$ and $\eta = \eta_B$, where $\xi_B = \overline{\eta_B}$. The constants Φ_A, Φ_B, Ψ_A and Ψ_B may be found from the speed distribution q(s) and the properties of a solution to the flow equations because of the following relations:

$$1)\ \Phi_A + \Phi_B = \Gamma$$

$$2)\ \Psi_A + \Psi_B = 0$$

$$3)\ \frac{d\Phi}{d\xi}\Big|_{\xi=\xi_0} = 0$$

$$4)\ \Phi(\text{TAIL}) - \Phi(\xi_0) = \Phi_M$$

where Φ_M is a constant which is determined by the data q(s).

The solution to the canonical equations is found by the spectral method. Φ and Ψ are each represented as a linear combination of a complete set of special solutions to the homogeneous canonical equations whose coefficients are determined by solving a boundary value problem for Ψ. The characteristic initial data for these special solutions is defined by the complete system of the Chebyshev polynomials in the ellipse and is imposed on an initial plane $\xi = \xi_C$ at some distance from ξ_A and ξ_B. On the second initial plane $\eta = \eta_C$, where $\xi_C = \overline{\eta_C}$, corresponding symmetric data will be given. If these special solutions are represented by $\Phi_3^{(n)}$ and $\Psi_3^{(n)}$, we express Φ_3 and Ψ_3 as the convergent series of real functions

$$\Phi_3(\xi,\eta) = \Phi_3^{(o)}(\xi,\eta) + \sum_{n=1}^{\infty} c_n \Phi_3^{(n)}(\xi,\eta)$$

$$\Psi_3(\xi,\eta) = \Psi_3^{(o)}(\xi,\eta) + \sum_{n=1}^{\infty} c_n \Psi_3^{(n)}(\xi,\eta)$$

where $\Phi_3^{(o)}$ and $\Psi^{(o)}$ satisfy the inhomogeneous equations for Φ_3 and Ψ_3. On both the initial planes $\xi = \xi_C$ and $\eta = \eta_C$ we have

$$\Phi_3^{(o)}(\xi,\eta) = 0$$

On the plane $\xi = \xi_C$ the initial data for the homogeneous special solutions is given by

$$\Phi_3^{(2n+1)}(\xi_C,\eta) + i\,\Phi_3^{(2n)}(\xi_C,\eta) = T_n(\xi_C) + \overline{T_n(\bar{\eta})}$$

When all of the component functions have been determined, the real coefficients c_n of a truncated expansion for Ψ_3 are found numerically by interpolating to satisfy the condition $\mathrm{Re}[\Psi(\xi,\bar{\xi})] = 0$ on the boundary of the ellipse. This matrix problem is well conditioned when equally spaced points are used on the circle in the z-plane corresponding to the ellipse boundary, using the relation between the ellipse and the annular ring given by $\xi = \frac{1}{2}(z + \frac{1}{z})$.

For an isolated airfoil Ψ is expressed as

$$\Psi(\xi,\eta) = (\Psi_1(\xi,\eta) + i\Psi_A)\log(\xi - \xi_A) + (\Psi_2(\xi,\eta) + i\Psi_B)\frac{1}{\xi - \xi_A} + \Psi_3(\xi,\eta)$$

and a similar expression holds for Φ. Characteristic initial value problems for the component functions are then determined as for a cascade.

The initial data q(s) determine a relation $q = q(\Phi)$ which can be used to find $q(\xi)$ once $\Phi(\xi)$ is known. We use this to determine the map from the (ξ,η)-plane to the (s,t)-plane. Since

$$f(\xi) = \log s - \log\left(\frac{\xi - \xi_0}{k\xi - \xi_0}\right),$$

for subsonic flow we have

$$\text{Re } f(\xi) = \log h(q) - \log \left|\frac{\xi - \xi_0}{k\xi - \xi_0}\right|$$

from which we can determine the coefficients a_n of the Chebyshev expansion for f.

We have determined the map function f from the solution in the ellipse and yet f is required in order to solve the canonical equations for Φ and Ψ. This is a nonlinear boundary value problem in which Re(f) and Re(Ψ) are prescribed simultaneously. By using the spectral method we have decomposed this into a coupled pair of problems for the map function f and the stream function Ψ which can be solved by iteration. We first find the incompressible solution in the ellipse, which is independent of f. After f is determined using the fast Fourier transform, Φ and Ψ are found in the ellipse as solutions of the canonical system with boundary condition Re(Ψ) = 0. From this solution a new determination of f is made and the iteration is repeated. The process converges rapidly to an acceptable answer that defines a cascade of airfoils whose pressure distribution has been prescribed.

3.3 The transonic case

When the Mach number of the flow exceeds one locally, $M^2 > 1$, the corresponding points in the hodograph plane no longer map into the real plane of symmetry $\xi = \bar{\eta}$ in the complex domain. The map function f will map the subsonic flow region into a portion of the ellipse in the plane $\xi = \bar{\eta}$ that is bounded by the sonic line $M^2 = 1$. In the remainder of the ellipse q and θ are complex, and the real supersonic flow region is a surface extending into complex (ξ,η)-space. Our procedure in this case is to solve a singular boundary value problem in the ellipse and then locate the real supersonic arc of the profile $\mathrm{Re}(\Psi) = 0$ afterwards by tracing real characteristics.

In order to solve the boundary value problem for the stream function Ψ, we prescribe an artificial boundary condition on the portion of the ellipse beyond the sonic surface $M^2 = 1$. Along the part of the ellipse boundary corresponding to real subsonic flow, we still impose the condition $\mathrm{Re}(\Psi) = 0$. On the rest of the boundary we let

$$\mathrm{Re}\ \{\Psi(\xi,\bar{\xi})\} + K_3\ I_m\ \{\Psi(\xi,\bar{\xi})\} = 0$$

where the constant K_3 is determined empirically. A similar boundary value problem for the Tricomi equation has been shown to be well posed by Jose Sanz [21]. Moreover, empirical data on the condition number of the matrix of linear equations determining the coefficients c_n of the Chebyshev expansion for Ψ indicate that this boundary value problem is well posed.

Further difficulties involve the determination of the region in the complex domain corresponding to the real supersonic zone and the solution of the flow equations there. The real characteristics in the supersonic zone cannot be determined by crossing the sonic line directly because the canonical equations fail when $M^2 = 1$. However, since the locus of points $M^2 = 1$ is a two-dimensional surface in the four-dimensional complex domain, it is possible to reach these characteristics by deforming the paths of integration appropriately. These paths are constructed to circumvent the sonic surface $M^2 = 1$ while staying on the proper branch of $\sqrt{1-M^2}$. They have been described in detail elsewhere [25]. Once the canonical equations have been solved along the real characteristics, the supersonic portion of the profile is determined by locating the points where $Re(\Psi)=0$.

Finally, a method must be devised for determining the map function from (ξ,η)-space to (s,t)-space. This will still be based upon the data for q given along the airfoil in the real physical plane. We compute the values of

$$Re(f) = \int \frac{\sqrt{1-M^2}}{q}\, dq - \log \left|\frac{\xi - \xi_0}{k\xi - \xi_0}\right|$$

on the subsonic arc of the ellipse as before. Elsewhere on the ellipse boundary we assign to Re(f) the value

$$Re(f) = \log h(c_*) + \{K_1 \left[\int_{c_*}^{q} \left|\sqrt{1-M^2}\right| \frac{dq}{q}\right]^{K_2}\} - \log \left|\frac{\xi - \xi_0}{k\xi - \xi_0}\right|$$

where the constants K_1 and K_2 are again determined empirically. The resulting solutions of the canonical partial differential equations define a shockless airfoil whose pressure distribution may not match the prescribed data for q along the supersonic arc of the profile.

3.4 Numerical procedures

We turn now to the numerical procedures we shall use to find a solution to our problem. In Section 3.2, the functions Φ and Ψ were expressed as a series of solutions to particular characteristic initial value problems. In order to solve these in the complex domain we must construct paths of integration in the ξ-plane and the η-plane. We place mesh points ξ_j on a path lying in the ξ-plane and determine mesh points η_k on a corresponding path in the η-plane. These paths will lie in complex (ξ,η)-space in the appropriate initial planes, which are determined below. On the resulting grid of points (ξ_j,η_k), we apply a finite difference approximation to the canonical system of differential equations for Φ and Ψ.

The characteristic initial value problems formulated in Section 3.2 for the pair of functions Φ_1 and Ψ_1 require initial data on the planes $\xi = \xi_A$ and $\eta = \eta_A$. To determine Φ_2 and Ψ_2, we require data on the planes $\xi = \xi_B$ and $\eta = \eta_B$. Here (ξ_A,η_A) and (ξ_B,η_B) are the points in the plane $\xi = \overline{\eta}$ corresponding to the exit and inlet velocities, respectively. For the remaining series of canonical equations, initial data must be imposed on a plane ξ_C and on a similar plane η_C. These must be located at some distance from ξ_A and ξ_B since the inhomogeneous equations for Φ_3 and Ψ_3 are singular at these points. We shall choose these initial planes such that $\xi_C = \overline{\eta_C}$.

Figure 2 shows the ellipse in the ξ-plane with the points ξ_A, ξ_B and ξ_C. Initial paths are represented there, as is the sonic locus, which is the set of points in the ξ-plane corresponding to the sonic

line in the plane $\xi = \bar{\eta}$. The figure is also a representation of the $\bar{\eta}$ plane, so that the conjugate of the initial paths in the η-plane are shown.

In the ξ-plane the paths begin at the point ξ_A and pass through ξ_B and ξ_C so that all initial planes are represented in the resulting grid. To solve the boundary value problem for the canonical system in the ellipse, we construct a series of paths there, each of which terminate on an arc of the ellipse boundary. These paths must be constructed to circumvent the sonic surface $M^2(\xi,\eta)=1$, since the characteristic equations fail there. Expressing this surface as the locus of points

$$\{(\xi,\eta): \eta = g(\xi),\ g \text{ an analytic function}\},$$

we note that $\bar{\xi} = g(\xi)$ on the sonic locus in the ξ-plane. Hence for nearby points on the sonic surface, $\bar{\eta}$ is the reflection of ξ across that locus. Points on paths in the ξ and η-planes resulting in a point on the sonic surface are depicted in Figure 3. In general, the presence of points on the sonic surface will appear in the ellipse as a pair of reflected points on initial paths in the ξ-plane and the $\bar{\eta}$-plane.

We shall distinguish between 3 types of paths: subsonic, transonic and supersonic. Subsonic paths lie entirely in the portion of the ellipse corresponding to the subsonic flow region. Transonic paths cross the sonic locus into the region of the ellipse which does not represent a real flow. Both of these types of paths terminate along a portion of the ellipse boundary. The solutions on the boundary are used to obtain a series representation for the flow and to determine

the map function from (ξ,η)-space to (s,t)-space. The supersonic paths determine the solution along characteristics in the real supersonic flow region.

In the subsonic case, the paths in the ξ-plane and the η-plane are chosen to be conjugate, so that $\xi_k = \overline{\eta_k}$. Since Figure 2 represents both the ξ-plane and the $\bar{\eta}$-plane, these are represented there by a single path. Let us prescribe symmetric data on the conjugate characteristic initial planes. Then by the theorem of Section 2.3, the resulting solution is a real function, so that the solution at the diagonal points (ξ_k,η_k) is real. Moreover, the solution at a point (ξ_j,η_k) above the diagonal is the conjugate of the solution at the point (ξ_k,η_j) below the diagonal. This halves the number of computations which are required. The computational grid for subsonic paths is shown in Figure 4.

Transonic paths are required when the arc of the ellipse to be covered contains points beyond the sonic line, which will not lie in the real flow region. In this situation, conjugate paths like those used in the subsonic flow region would result in grid points on the sonic surface $M^2 = 1$. Moreover, symmetric data on conjugate initial paths will no longer result in solutions which are real functions throughout the computational grid, since the coefficient τ is not a real function across the sonic line. Therefore, points on the grid above the diagonal would be computed even for conjugate paths. Hence we construct a pair of paths in the ξ-plane and the η-plane that terminate on the ellipse boundary but are not conjugates of one another. These transonic paths each consist of two arcs, together with a portion of the ellipse boundary. The arcs are constructed to prevent

the appearance of the reflected points described above, which would give rise to points on the sonic surface. In addition, the paths in the ξ-plane and the $\bar{\eta}$-plane traverse the ellipse in opposite directions in order to avoid the sonic surface. Points on the boundary of the ellipse in the plane $\xi = \bar{\eta}$ lie in the resulting grid, as shown in Figure 5. As the transonic paths wind around the sonic locus, the value of $\sqrt{1-M^2}$ is adjusted to remain on the correct branch of the solution. Thus valid flows result even in marginal cases.

The real supersonic zone must be found by constructing paths for which the resulting grid twists around the two-dimensional sonic surface $M^2 = 1$ to reach points in the real supersonic flow region. These points turn out to be of the form (ξ_j,η_k) where both $(\xi_j,\bar{\xi}_j)$ and $(\bar{\eta}_k,\eta_k)$ lie on the sonic line. This is to be expected since a real supersonic characteristic will contain a point on the sonic line, at which the characteristic will cusp. Therefore the initial paths are chosen so that both the ξ-characteristics and the η-characteristics of the resulting computational grid intersect the sonic line and a triangle of real supersonic points is obtained, as shown in Figure 6. The solution cannot be determined at points on the grid beyond the sonic line due to the singularity of the characteristic equations there. Hence the paths cover the sonic loci in opposite directions so that the real supersonic zone is encountered before reaching the sonic line. Since the characteristic transformation is multivalued, the manner in which the grid wraps around the sonic locus determines the branch of the solution. A correct branch yielding the solution in the real supersonic zone has been found empirically by careful choice of the supersonic paths of integration [4,24], as depicted in Figure 2.

The finite difference equations have the characteristic form

$$\Phi_{j,k} - \Phi_{j-1,k} = \frac{\tau_{j,k} + \tau_{j-1,k}}{2} [\Psi_{j,k} - \Psi_{j-1,k}]$$

$$\Phi_{j,k} - \Phi_{j,k-1} = - \frac{\tau_{j,k} + \tau_{j,k-1}}{2} [\Psi_{j,k} - \Psi_{j,k-1}]$$

and are second order accurate away from the sonic surface, where they become singular. The coefficients τ are determined from the values of u and v, which can be found using the conformal map from (ξ, η)-space to (s,t)-space. In practice, however, it is more convenient to solve the characteristic equations

$$v_{j,k} - v_{j-1,k} = \frac{\lambda^+_{j,k} + \lambda^+_{j-1,k}}{2} [u_{j,k} - u_{j-1,k}]$$

$$v_{j,k} - v_{j,k-1} = \frac{\lambda^-_{j,k} + \lambda^-_{j,k-1}}{2} [u_{j,k} - u_{j,k-1}]$$

where

$$\lambda^{\pm} = \frac{uv \pm \sqrt{c^2(q^2 - c^2)}}{q^2 - v^2}$$

These may by solved by the predictor corrector difference method of Massau [8] which has been discussed elsewhere [?]. A third order accurate method may also be used which performs the calculations with two different grid sizes and uses Richardson extrapolation.

3.5 Determination of the profile

The solutions Φ and Ψ of the canonical equations may be substituted into the formula

$$x + iy = \int \frac{d\Phi + i d\Psi/\rho}{qe^{-i\theta}}$$

to calculate the shape of the profile $\Psi = 0$. For subsonic and transonic paths this integration is performed on paths in the computational grid terminating on the boundary of the ellipse (Figures 4 5). In the real supersonic zone x and y are also calculated along real supersonic characteristics (Figure 6). These can then be plotted to obtain information about the behavior of the flow. Cusping of the characteristics indicates the presence of a limiting line, which corresponds to the appearance of a shock wave. In order for a smooth profile to result, care must be taken to stay on the right branch of the logarithms which occur in the singular solutions.

This profile is a streamline for inviscid flow, however, while we wish to obtain a model for viscous flow past a blade section. Due to viscous effects, physical flows will contain a boundary layer at the surface of the profile. The thickness of this layer will be greater on the upper than the lower surface, which will reduce the angle of attack and cause a loss of circulation. Flow on the upper surface of the airfoil is therefore retarded, so that the supersonic zone diminishes and shockless flow may not occur. Moreover, if the boundary layer

separates before the last few percent of chord, a large wake results and the shockless regime is lost.

To overcome these difficulties, we use a boundary layer correction to modify the profile and estimate the point at which separation may be predicted. The profile is obtained by subtracting from the blade which generates the inviscid solution along the streamline $\Psi = 0$ a displacement thickness δ^* determined by the method of Nash and McDonald [16]. This is based on numerical integration of the von Karman momentum equation

$$\frac{d\theta^*}{ds} + (2 + H - M^2) \frac{dq}{ds} \frac{\theta^*}{q} = \frac{\tau}{\rho q^2}$$

where θ^* is the momentum thickness, $H = \frac{\delta^*}{\theta^*}$, τ is the skin friction, and s is the arc length. M^2 and q are given by the inviscid solution and H and τ are determined by semi-empirical formulas. Integration is initiated at transition points (x_R, y_R) which are prescribed for the upper and lower surfaces of the profile. The displacement of the laminar boundary layer before transition is small and can be neglected. Separation is predicted when the Nash-MacDonald parameter $SEP = \frac{-\theta^*}{q} \frac{dq}{ds}$ exceeds 0.04. Usually the input speed distribution is chosen so that SEP is near 0.03 near the trailing edge of the upper surface.

IV. COMPUTATIONAL RESULTS

4.1 Comparison with the Korn code

In this chapter we present the results of runs obtained with the new version of the design code. We will also discuss the effect of a low gap-to-chord ratio on the flow and on the geometry of the resulting cascade.

We first present a cascade designed with the new code and a similar case obtained with the previous code [4]. Both result from 3 iterations NI of the map function and 128 functions in the series representing Φ and Ψ. The turning angle of the flows is 45^0 and the inlet and exit Mach numbers are .77 and .44, respectively. The gap-to-chord ratio obtained in both of these cases is .81. In the run with the new code, the ellipse parameter EP is set to .6. The other input parameters, including the speed distribution of Figure 7, are identical in the two runs except for the location of the singularities ξ_A and ξ_B. Because of the difference in the conformal mappings used by the two codes, a change in the position of the these singularities was required in order to obtain a similar design from both runs.

In Figure 11 the difficulties of the old code are illustrated. Here the singularities ξ_A and ξ_B are at the edge of the unit circle and can be moved no further, so that this gap-to-chord ratio represents the limits of the old code. Moreover, it can be seen from the resulting blade in Figure 12, which is plotted without interpolating between data points, that the effect of having the singularities so close to the boundary in the unit circle is poor resolution near the stagnation

point and at the tail. These difficulties have been corrected in the new code. In Figure 9 we see that the use of an ellipse of medium eccentricity allows us to obtain the desired gap-to-chord ratio easily. By moving the stagnation point counterclockwise along the boundary of the ellipse, we have been able to keep XIA and XIB near the foci, which decreases the distance between the blades and improves the geometry of the paths relative to the sonic surface.

A compressor designed with an earlier version of the previous code has been tested in a cascade wind tunnel with successful results. Moreover, close agreement has been found between the results of the design code and the N.Y.U. analysis code, the latter of which has been favorably compared with experimental results in a number of cases. The agreement between the new and old design codes allows us to refer to these tests to conclude that the new code effectively designs physically reasonable shockless airfoils.

4.2 Two new compressors

We next present a cascase of stator airfoils having a low gap-to-chord ratio which was designed with the new version of the code. This case was obtained with two cycles of the map function and 32 Chebyshev coefficients. The inlet and exit Mach numbers for the flow are 0.78 and 0.45, the turning angle is 47^0, and the peak value of the Mach number on the upper surface is 1.2. The gap-to-chord ratio of 0.42 was achieved by using a ratio of minor to major axes of 0.25, and placing the singularities ξ_A and ξ_B near the foci and at a substantial distance from one another, as shown in Figure 14. The distance between the upper and lower pressure distributions, as shown in Figure 13, decreases with the gap-to-chord ratio so that this cascade has less lift than the preceding example.

The distribution of points around the ellipse in Figure 13 shows acceptable resolution of the profile at the stagnation point and the tail. This results from a property of the mapping which also makes the solution highly sensitive to small changes in the location of parameters in the ellipse. A slight modification in the placement of ξ_B, or moving ξ_0 by a single mesh point, has a great effect on the position of the sonic line, which rotates along the ellipse boundary in the same direction as ξ_0. In Figure 14 it can be seen that for a highly eccentric ellipse the sonic locus extends into the interior of the ellipse so that the transonic and supersonic paths must be placed at some distance from this curve in order to avoid points on the sonic surface $M^2 = 1$.

As our final example we have a cascade for which the flow is supersonic on both the upper and lower surfaces. As shown in Figure 16, the speed distributions along the upper and lower surfaces of the blades are similar and remain relatively high at the rear of the blades in order to obtain a realistic trailing edge. This case was obtained with two iterations of the map function and 32 Chebyshev coefficients. The ratio of minor to major axes in the ellipse is 0.3, and the gap-to-chord ratio is 0.54. The inlet and exit Mach numbers for this case are .72 and .53. Because supersonic arcs of the profile are straight lines, a profile with substantial supersonic zones will have less camber, so that the turning angle of the resulting flow is only 22^0.

Figure 3 shows the hodograph plane with a pair of sonic loci representing the two sonic surfaces. In this case it was more difficult to construct paths which avoid the sonic surface. This reflects the fact that in the physical flow, choking will occur if the two supersonic regions come too close to one another, so that smooth flows will not be possible. The extent of the difficulty is reflected in the fact that slight modifications of parameters in the ellipse from those which are used here caused a point on the line joining ξ_A and ξ_B to generate a point on the sonic surface. Hence the limitations of the present code do not arise from inadequate resolution or the proximity of the singularities to the boundary, but from the increased interaction between points on the sonic surface and points required in order to obtain a solution.

4.3 Input files for additional cases

In this section we present input files for an isolated airfoil and a turbine cascadewith accompanying graphics 19-23 in Chapter VI. The airfoil shown in Figure 19 was designed by Jose Sanz and is a modification of the original supercritical wing section developed empirically by Whitcomb. Tape 3, the input speed distribution, is presented, as well as the additional input parameters contained in Tape 7. Graphical output include the input and output pressure distributions along the blade (Figure 19), and the plot of the ellipse (Figure 20). In this case we have used an ellipse of lower eccentricity, as the difficulties encountered designing cascades of airfoils do not arise.

Turbines in transonic flow will generally be expected to have supersonic exit speeds, while in our design code the exit velocity is constrained to be subsonic. Nonetheless, we present a case designed by Jose Sanz which is useful as a first approximation. The speed distribution typical of a turbine is presented in Figure 21, and the paths of integration are shown in Figure 22. The resulting cascade of Figure 23 has a larger turning angle and greater supersonic zone than the previous turbines developed with our method. The input files which generate these cases are presented on the following pages.

WHITCOMB WING SECTION
INPUT SPEED DISTRIBUTION

CARD	S-INPUT	S-USED	Q-INPUT
1	.011174	0.000000	-.838000
2	.098920	.087217	-.765226
3	.246873	.234278	-.859370
4	.390006	.376549	-1.054257
5	.535817	.521481	-1.128083
6	.730869	.715358	-1.137690
7	.876067	.859681	-1.141049
8	.915000	.898379	-1.142000
9	.950000	.933168	-1.130000
10	.986233	.969183	-.983997
11	1.002148	.985002	-.777272
12	1.016000	.998770	-.280000
13	1.027870	1.010569	.341939
14	1.047740	1.030319	1.130473
15	1.063555	1.046039	1.365332
16	1.090000	1.072324	1.468000
17	1.130000	1.112083	1.475000
18	1.170427	1.152267	1.467623
19	1.220562	1.202100	1.459237
20	1.315641	1.296606	1.447927
21	1.460661	1.440751	1.433724
22	1.750807	1.729149	1.194273
23	1.894000	1.871479	.972500
24	2.023300	2.000000	.838000

FLOW PARAMETERS

RUN = -15 ; IPLT = 82

NI = 3 ; NP = 8

NFC = 32 ; NCAS = 2

NPTS = 201 ; NOSE = 0

MACH = .765 ; RN = .80E+07

EP = .80 ; GAMMA = 1.40

RATC = .60 ; CONE = .50

CTWO = 1.00 ; CTHR = 1.00

TRANU = .28 ; TRANL = .28

GRID = .12 ; MRP = -2

XIA = .100 , -.150 ; XIB = .100 , -.150

XIC = -.050 , -.200

INPUT SPEED FOR TURBINE CASE

CARD	S-INPUT	S-USED	Q-INPUT
1	.208000	0.000000	-.836500
2	.360000	.132002	-.584200
3	.630439	.366860	-.285000
4	.850000	.557534	-.172500
5	1.002470	.689944	-.160000
6	1.048000	.729484	-.157500
7	1.083000	.759879	-.136250
8	1.114033	.786829	-.081455
9	1.146000	.814590	.037500
10	1.178000	.842380	.264000
11	1.210000	.870170	.576000
12	1.290000	.939644	.980000
13	1.421000	1.053409	1.208000
14	1.551000	1.166305	1.352500
15	1.660000	1.260965	1.465000
16	1.740000	1.330439	1.530000
17	1.850000	1.425967	1.560000
18	1.970930	1.530986	1.450000
19	2.150384	1.686830	1.147500
20	2.310262	1.825673	.970000
21	2.510999	2.000000	.836500

FLOW PARAMETERS

RUN = -273 ; IPLT = 12

NI = 1 ; NP = 8

NFC = 64 ; NCAS = 3

NPTS = 201 ; NOSE = 10

MACH = .755 ; RN = .10E+07

EP = .60 ; GAMMA = 1.40

RATC = .60 ; CONE = .50

CTWO = 1.00 ; CTHR = 1.00

TRANU = .30 ; TRANL = .50

GRID = .10 ; MRP = -1

XIA = 1.000 , .020 ; XIB =-1.080 , .200

XIC =-1.110 , .140

4.4 Sample run of the code

In this section we present the printed output of a sample run of the code. The acoompanying graphics appear as Figures 7-10. The first page of printed output consists of the input speed distribution from TAPE3. On the second page are found the input parameters entered on TAPE7, follwed by the inlet and exit Mach numbers, the residual difference in the value of the potential function computed at each cycle, and the values of DX and DY after each iteration. The next page lists the points close to the sonic surface which have been encountered on the computational grids. The sign of the stream function along the real characteristics is then shown, so that the supersonic arc of the profile may be determined as the curve where this function vanishes. From this plot the presence of limiting lines can be inferred. Finally the output data for the resulting cascade is printed, including the coordinates of the airfoil before and after the boundary layer correction and the corresponding Mach numbers. An explanation of the output parameters listed here is given in Section 7.2.

The first plot comprises a plot of the input speed distribution (Figure 7). This is followed by a plot of the pressure (or Mach) distribution and the resulting blade section, which may be plotted with supersonic characteristics (Figure 8). There follows a plot of the complex hodograph plane (Figure 9), and finally, a plot of the isolated airfoil or of NCAS airfoils in cascade (Figure 10). Variations in the types of plotted output depend on the value of IPLT, the plot control

parameter. These options are described in Section 7.1. A negative value of IPLT disables the graphics.

BEGIN EXECUTION OF DESIGN RUN -44

TAPE3=INPUT

CARD	S-INPUT	S-USED	Q-INPUT
1	.010600	0.000000	-.691000
2	.176500	.157488	-.608992
3	.504488	.468846	-.562715
4	.751306	.703150	-.720804
5	.820327	.768671	-.788132
6	.907617	.851535	-.876813
7	.936237	.878703	-.905578
8	.964321	.905363	-.922484
9	.985000	.924994	-.925500
10	1.002000	.941132	-.890000
11	1.010010	.948736	-.804954
12	1.017042	.955411	-.595487
13	1.026123	.964031	0.000000
14	1.034320	.971813	.572142
15	1.041325	.978463	.996473
16	1.052891	.989443	1.280000
17	1.069407	1.005121	1.560000
18	1.079932	1.015113	1.620000
19	1.090457	1.025104	1.650000
20	1.115771	1.049134	1.680000
21	1.141084	1.073164	1.695000
22	1.166397	1.097193	1.700000
23	1.191710	1.121223	1.700000
24	1.256062	1.182312	1.700000
25	1.303599	1.227439	1.700000
26	1.352135	1.273514	1.691000
27	1.396689	1.315809	1.670000
28	1.441242	1.358102	1.620000
29	1.517326	1.430329	1.405000
30	1.593410	1.502555	1.210000
31	1.669493	1.574781	1.050000
32	1.786404	1.685763	.900000
33	1.902314	1.795796	.785000
34	2.117425	2.000000	.691000

SUBSONIC CASCADE DESIGN RUN 44 86/06/12.

TAPE7-INPUT

RUN =	-44 ;	IPLT =	74
NI =	3 ;	NP =	8
NFC =	64 ;	NCAS =	2
NPTS =	251 ;	NOSE =	12
MACH =	.645 ;	RN =	.10E+07
EP =	.60 ;	GAMMA =	1.40
RATC =	0.00 ;	CONE =	.30
CTWO =	.50 ;	CTHR =	1.00
TRANU =	.35 ;	TRANL =	.15
GRID =	.10 ;	MRP =	1

XIA = 1.080 , 0.000 ; XIB =-1.060 , .015

XIC =-1.100 , .016

CYCLE	MINLET	MEXIT	RESIDUAL	DX	DY
1	.748	.418	.190E+00	.12072	.07484
2	.764	.430	.752E-01	.03061	.04154
3	.763	.435	.256E-01	-.00721	.04121

LONGEST SUBSONIC PATH HAS 79 POINTS

SUBSONIC CP TIME IS 1221.3 SECONDS

LONGEST SUPERSONIC PATH HAS 121 POINTS

SUPERSONIC CP TIME IS 40.1 SECONDS

LISTING OF POINTS ENCOUNTERED NEAR THE SONIC SURFACE

MODE	XI		ETA		ROOTOLD	-	ROOT	
3	-.884	-.530	-.884	.530	.072	.180	.030	.149
3	-.478	-.693	-.478	.693	.060	.197	.017	.179
3	-.304	-.728	-.304	.728	.093	.172	.061	.121
-1	-1.175	.255	-1.175	-.255	.207	.033	.185	.038
-1	-1.199	.040	-1.199	-.040	.246	-.003	.192	-.009
-1	-1.195	.007	-1.195	-.007	.236	-.001	.185	-.007
-1	-1.188	-.025	-1.188	.025	.227	.000	.178	-.003
-1	-1.179	-.056	-1.179	.056	.220	.002	.172	.000
-1	-1.168	-.087	-1.168	.087	.213	.004	.165	.003
-1	-1.155	-.117	-1.155	.117	.208	.006	.160	.006
-1	-1.141	-.146	-1.141	.146	.204	.007	.156	.007
-1	-1.124	-.175	-1.124	.175	.201	.007	.152	.008
-1	-1.105	-.204	-1.105	.204	.199	.007	.150	.009
-1	-1.083	-.233	-1.083	.233	.198	.007	.148	.008
-1	-1.060	-.261	-1.060	.261	.197	.007	.147	.008
-1	-1.034	-.289	-1.034	.289	.197	.006	.146	.007
-1	-1.006	-.316	-1.006	.316	.197	.006	.146	.006
-1	-.976	-.344	-.976	.344	.197	.005	.146	.004
-1	-.943	-.371	-.943	.371	.198	.004	.147	.003
-1	-.909	-.397	-.909	.397	.199	.004	.147	.002
-1	-.872	-.423	-.872	.423	.200	.003	.148	.001
-1	-.833	-.449	-.833	.449	.202	.003	.149	.000
-1	-.792	-.474	-.792	.474	.203	.002	.150	-.001
-1	-.749	-.499	-.749	.499	.205	.002	.151	-.001
-1	-.704	-.523	-.704	.523	.207	.002	.153	-.002
-1	-.657	-.547	-.657	.547	.209	.001	.154	-.002
-1	-.609	-.571	-.609	.571	.211	.001	.156	-.003
-1	-.560	-.595	-.560	.595	.214	.001	.157	-.003
-1	-.509	-.620	-.509	.620	.216	.001	.159	-.003
-1	-.456	-.645	-.456	.645	.218	.002	.158	-.003
-1	-.404	-.672	-.404	.672	.218	.003	.158	-.000
-1	-.353	-.700	-.353	.700	.214	.003	.154	.000
-1	-.304	-.727	-.304	.727	.144	.055	.114	.041

POSITIVE AND NEGATIVE VALUES OF STREAM FUNCTION AT SUPERSONIC POINTS

```
N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N
N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N N P P
N N N N N N N N N N N N N N N N N N N N N N N N N N N N N P P P P
N N N N N N N N N N N N N N N N N N N N N N N N N N N P P P P P
N N N N N N N N N N N N N N N N N N N N N N N N N P P P P P
N N N N N N N N N N N N N N N N N N N N N N N N P P P P P
N N N N N N N N N N N N N N N N N N N N N N N P P P P P P
N N N N N N N N N N N N N N N N N N N N N N P P P P P P P
N N N N N N N N N N N N N N N N N N N N N N P P P P P P P
N N N N N N N N N N N N N N N N N N N N N P P P P P P P
N N N N N N N N N N N N N N N N N N N N P P P P P P P P
N N N N N N N N N N N N N N N N N N N P P P P P P P P
N N N N N N N N N N N N N N N N N N P P P P P P P P P
N N N N N N N N N N N N N N N N N P P P P P P P P P
N N N N N N N N N N N N N N N N P P P P P P P P P P
N N N N N N N N N N N N N N N P P P P P P P P P P
N N N N N N N N N N N N N N P P P P P P P P P P P
N N N N N N N N N N N N N P P P P P P P P P P P
N N N N N N N N N N N N P P P P P P P P P P P P
N N N N N N N N N N N P P P P P P P P P P P P
N N N N N N N N N N P P P P P P P P P P P P
N N N N N N N N N P P P P P P P P P P P P
N N N N N N N N P P P P P P P P P P P P P
N N N N N N N P P P P P P P P P P P P P
N N N N N N P P P P P P P P P P P P P
N N N N N P P P P P P P P P P P P P
N N N N N P P P P P P P P P P P P
N N N N P P P P P P P P P P P P
N N N N P P P P P P P P P P P
N N N N P P P P P P P P P P
N N N P P P P P P P P P P
N N N P P P P P P P P P
N N N P P P P P P P P
N N P P P P P P P P
N N P P P P P P P
N P P P P P P P
N P P P P P P
N P P P P P
N P P P P
N P P P
N P P
N P
N
```

THICKNESS/CHORD= .1348

COEFFICIENT OF LIFT=1.3386

DIFFUSION FACTOR= .680

GAP/CHORD= .810

INLET MACH NUMBER= .763 INLET FLOW ANGLE= 42.10

EXIT MACH NUMBER= .435 EXIT FLOW ANGLE= 87.22

TURNING ANGLE= 45.12

DX=-.0077 DY= .0438

COORDINATES FROM LOWER SURFACE TAIL TO UPPER SURFACE TAIL

X	Y	ANG	K	M	THETA	SEP	XS	YS
.9558	.2916	.9	2.64	.4356	.00216	-.00220	.9558	.2944
.9554	.2916	.9	.98	.4354	.00216	-.00220	.9554	.2944
.9498	.2915	.6	.46	.4328	.00219	-.00210	.9498	.2943
.9377	.2914	.2	.22	.4281	.00223	-.00185	.9377	.2943
.9202	.2914	.0	.05	.4218	.00229	-.00182	.9202	.2944
.8986	.2914	-.0	-.08	.4145	.00236	-.00181	.8986	.2945
.8743	.2914	.1	-.20	.4065	.00244	-.00182	.8743	.2946
.8479	.2913	.4	-.30	.3983	.00253	-.00183	.8479	.2946
.8203	.2910	.8	-.40	.3902	.00262	-.00183	.8203	.2944
.7918	.2904	1.5	-.50	.3823	.00272	-.00182	.7917	.2939
.7627	.2894	2.3	-.61	.3747	.00282	-.00179	.7625	.2931
.7331	.2880	3.4	-.71	.3677	.00292	-.00171	.7329	.2917
.7033	.2859	4.6	-.81	.3613	.00301	-.00157	.7030	.2898
.6732	.2831	6.0	-.89	.3558	.00309	-.00136	.6728	.2871
.6427	.2789	7.6	-.98	.3514	.00314	-.00110	.6422	.2831
.6126	.2745	9.3	-1.06	.3480	.00317	-.00078	.6119	.2787
.5827	.2691	11.1	-1.11	.3460	.00317	-.00039	.5819	.2734
.5531	.2628	13.0	-1.16	.3454	.00313	.00005	.5520	.2671
.5239	.2555	15.0	-1.18	.3463	.00306	.00050	.5228	.2599
.4954	.2473	17.0	-1.18	.3488	.00294	.00095	.4940	.2516
.4677	.2382	19.0	-1.15	.3530	.00278	.00135	.4662	.2424
.4407	.2284	20.9	-1.10	.3588	.00259	.00168	.4392	.2325
.4149	.2181	22.7	-1.03	.3660	.00239	.00188	.4134	.2218
.3900	.2073	24.3	-.96	.3745	.00217	.00199	.3885	.2107
.3663	.1963	25.7	-.87	.3839	.00196	.00200	.3648	.1993
.3435	.1850	27.0	-.79	.3943	.00175	.00195	.3422	.1877
.3219	.1737	28.1	-.71	.4053	.00156	.00185	.3206	.1761
.3011	.1624	29.0	-.63	.4168	.00139	.00172	.3000	.1645
.2814	.1513	29.9	-.54	.4287	.00123	.00158	.2803	.1531
.2624	.1402	30.5	-.48	.4408	.00108	.00145	.2615	.1418
.2449	.1297	31.1	-.39	.4531	.00095	.00131	.2440	.1310
.2277	.1192	31.5	-.31	.4654	.00084	.00115	.2269	.1204

X	Y	ANG	K	M	THETA	SEP	XS	
.2112	.1090	31.9	-.23	.4777	.00073	.00101	.2106	.1101
.1954	.0992	32.1	-.16	.4897	.00064	.00087	.1949	.1001
.1804	.0897	32.3	-.09	.5014	.00056	.00075	.1799	.0905
.1659	.0805	32.4	-.03	.5127	.00048	.00064	.1654	.0812
.1521	.0717	32.4	.04	.5236	.00041	.00054	.1517	.0723
.1387	.0633	32.4	.12	.5342			.1387	.0633
.1261	.0553	32.3	.24	.5447	TRANSITION		.1261	.0553
.1138	.0476	32.1	.41	.5548			.1138	.0476
.1023	.0404	31.7	.64	.5647			.1023	.0404
.0911	.0336	31.3	.92	.5738			.0911	.0336
.0806	.0273	30.6	1.19	.5817			.0806	.0273
.0705	.0213	29.8	1.57	.5876			.0705	.0213
.0610	.0160	28.8	1.98	.5919			.0610	.0160
.0515	.0110	27.6	2.97	.5946			.0515	.0110
.0434	.0062	26.0	4.88	.5962			.0434	.0062
.0358	.0025	23.6	8.07	.5952			.0358	.0025
.0286	-.0003	20.1	13.06	.5869			.0286	-.0003
.0218	-.0024	14.8	22.64	.5620			.0218	-.0024
.0155	-.0037	6.5	38.63	.5090			.0155	-.0037
.0098	-.0037	-6.2	48.28	.3962			.0098	-.0037
.0043	-.0024	-21.9	59.53	.2177			.0043	-.0024
-.0018	.0050	-55.1	47.01	.2317			-.0018	.0050
-.0040	.0106	-71.4	37.48	.4814			-.0040	.0106
-.0049	.0184	-88.3	25.25	.7156			-.0049	.0184
-.0057	.0269	-100.7	16.77	.8666			-.0057	.0269
-.0024	.0440	-117.5	6.97	1.1966			-.0024	.0440
.0047	.0561	-123.1	3.55	1.2386			.0047	.0561
.0140	.0695	-126.4	1.93	1.2407			.0140	.0695
.0267	.0860	-128.7	1.40	1.2307			.0267	.0860
.0414	.1038	-130.6	1.15	1.2213			.0414	.1038
.0581	.1226	-132.2	1.03	1.2166			.0581	.1226
.0761	.1419	-133.8	1.01	1.2165			.0761	.1419
.0941	.1603	-135.3	1.04	1.2199			.0941	.1603
.1110	.1766	-136.7	1.10	1.2245			.1110	.1766
.1259	.1904	-138.0	1.18	1.2286			.1259	.1904
.1390	.2020	-139.1	1.27	1.2314			.1390	.2020
.1504	.2118	-140.2	1.36	1.2329			.1504	.2118
.1607	.2202	-141.3	1.45	1.2332			.1607	.2202
.1701	.2276	-142.3	1.54	1.2327			.1701	.2276
.1788	.2343	-143.2	1.62	1.2314			.1788	.2343
.1871	.2403	-144.2	1.69	1.2295			.1871	.2403
.1949	.2459	-145.1	1.77	1.2271			.1949	.2459
.2025	.2511	-146.1	1.84	1.2242			.2025	.2511
.2099	.2560	-147.0	1.90	1.2208			.2099	.2560
.2171	.2606	-147.9	1.97	1.2170			.2171	.2606
.2241	.2649	-148.8	2.04	1.2128			.2241	.2649
.2310	.2690	-149.8	2.11	1.2081			.2310	.2690
.2378	.2729	-150.7	2.19	1.2029			.2378	.2729
.2444	.2766	-151.7	2.27	1.1967			.2444	.2766
.2508	.2800	-152.6	2.35	1.1895			.2508	.2800

X	Y	ANG	K	M	THETA	SEP	XS	YS
.2571	.2832	-153.6	2.43	1.1810			.2571	.2832
.2633	.2862	-154.5	2.55	1.1709			.2633	.2862
.2692	.2889	-155.5	2.74	1.1590			.2692	.2889
.2748	.2914	-156.4	3.00	1.1453			.2748	.2914
.2800	.2937	-157.4	3.20	1.1294			.2800	.2937
.2849	.2956	-158.4	3.11	1.1094			.2849	.2956
.2897	.2975	-159.3	3.30	1.0842			.2897	.2975
.2943	.2992	-160.2	2.04	1.0527			.2943	.2992
.3110	.3045	-162.3	1.90	.9557			.3110	.3045
.3234	.3087	-163.7	1.82	.9268			.3234	.3087
.3363	.3123	-165.1	1.64	.8976	TRANSITION		.3363	.3123
.3504	.3159	-166.5	1.61	.8700			.3504	.3159
.3645	.3191	-167.8	1.51	.8436	.00032	.00064	.3646	.3185
.3791	.3221	-169.1	1.46	.8181	.00038	.00075	.3793	.3214
.3940	.3248	-170.4	1.35	.7930	.00045	.00088	.3941	.3240
.4093	.3272	-171.6	1.24	.7683	.00052	.00103	.4094	.3263
.4249	.3294	-172.7	1.10	.7439	.00060	.00117	.4251	.3283
.4410	.3313	-173.7	.96	.7203	.00069	.00130	.4412	.3301
.4575	.3330	-174.6	.83	.6979	.00079	.00141	.4576	.3316
.4744	.3345	-175.4	.72	.6771	.00089	.00148	.4745	.3329
.4916	.3358	-176.1	.65	.6581	.00100	.00153	.4917	.3340
.5092	.3368	-176.8	.61	.6409	.00111	.00156	.5093	.3349
.5271	.3378	-177.4	.58	.6252	.00123	.00160	.5272	.3356
.5454	.3385	-178.0	.57	.6106	.00135	.00165	.5455	.3361
.5641	.3390	-178.6	.54	.5968	.00148	.00173	.5641	.3364
.5836	.3396	-179.2	.54	.5834	.00162	.00187	.5836	.3367
.6028	.3398	-179.8	.49	.5701	.00177	.00209	.6028	.3365
.6225	.3397	-180.4	.43	.5567	.00194	.00232	.6225	.3361
.6425	.3395	-180.9	.34	.5432	.00212	.00256	.6425	.3354
.6630	.3391	-181.3	.25	.5299	.00233	.00278	.6629	.3346
.6840	.3386	-181.6	.15	.5169	.00256	.00297	.6838	.3335
.7054	.3380	-181.7	.05	.5044	.00282	.00311	.7053	.3322
.7273	.3373	-181.8	-.04	.4926	.00308	.00317	.7271	.3310
.7497	.3366	-181.8	-.12	.4818	.00336	.00316	.7495	.3298
.7724	.3360	-181.6	-.19	.4720	.00363	.00307	.7722	.3287
.7955	.3354	-181.3	-.24	.4634	.00390	.00290	.7953	.3278
.8187	.3349	-181.0	-.29	.4559	.00415	.00264	.8186	.3271
.8420	.3346	-180.6	-.32	.4497	.00438	.00230	.8419	.3267
.8648	.3344	-180.2	-.33	.4447	.00457	.00191	.8648	.3266
.8868	.3344	-179.8	-.33	.4411	.00472	.00148	.8868	.3267
.9070	.3345	-179.4	-.29	.4387	.00483	.00104	.9071	.3270
.9252	.3350	-179.1	-.24	.4373	.00490	.00066	.9253	.3276
9386	.3352	-178.9	-.03	.4366	.00495	.00048	.9387	.3279
.9465	.3353	-178.9	1.29	.4363	.00497	.00045	.9466	.3280
.9481	.3354	-179.1	0.00	.4356	.00497	.00045	.9483	.3280

RATC = .5342

PROFILE DRAG COEFFICIENT = .0051

LOSS COEFFICIENT = .0093

V. USER'S MANUAL

5.1 Input for Program COMPRES

This chapter is intended as a guide for those using Program COMPRES. It summarizes different code options and describes input and output parameters. Also discussed are modifications that may be required to achieve a good design. The Glossaries of input and output parameters in Chapter VII will also be helpful to new users of the code.

The code is written in Fortran 4 and runs on the Cyber 170 computer at the Courant Institute of Mathematical Sciences. A typical run at MRP = 1, NFC = 32, and NI = 2 requires 200K octal words of core storage and 12 minutes of CP time.

The input to the design program consists of two data files, TAPE3 and TAPE7. TAPE3 contains data specifying the speed q as a function of arc length s along the airfoil measured from the lower trailing edge to the upper trailing edge. The first record of the file contains NIN, the number of data points to be entered. The code uses different routines to interpolate this data, based on the sign of NIN. For NIN negative a distribution q(s) of points is computed by an exponential spline fitting routine as shown in Figure 16. When NIN is positive, a cubic spline interpolation routine is used as in the speed distribution of Figure 7, and a greater number of carefully chosen data points are required.

The exponential spline routine interpolates data according to the ordinary differential equation

$$-E\frac{d^4q}{ds^4} + \frac{d^2q}{ds^2} = G$$

where E, G and q must be specified for each value of s. E controls the oscillation at the data point and G is a tension parameter which weights the curve. Lowering E decreases the oscillation until the limiting case where E vanishes, when the resulting curve is quadratic. Positive values of G result in a convex speed distribution while negative values reverse the curvature. This interpolation routine has the advantage that a smooth speed distribution may be obtained from a small number of data points. It is useful when the speed distribution must be varied slightly in a series of runs for the purpose of achieving an optimal design.

TAPE7 contains all the other input parameters used in the code. A list of these is found in the Section 7.1, together with the default values which are used when no input value is given. TAPE7 parameters are entered in namelist format, as described in standard Fortran references. A card sequence for a typical cascade run is shown below.

Data Card 1:

```
{ $ONE NRN=10, NFC=64,MRP=-1,NI=3,NOSE=6,NCAS=3$ }
```

Data Card 2:

```
{ $ONE EP=.4, MACH=.64, RN=1.0 E+06$ }
```

Data Card 3:

```
{ $ONE XIA=(.92,.30), XIB=(-.86,-.25)$ }
```

Data Card 4:

```
{ $ONE IPLT=82$ }
```

Most TAPE7 parameters are easily chosen and need not be discussed. A valid flow is computed regardless of the number, NI, of iterations of the map function, although at least 3 iterations are usually required to achieve an accurate fit to the prescribed pressure distribution. The location of the constants XIA, XIB, and XIC requires special attention and will be discussed in Section 3.

MACH denotes the Mach number of the flow at q=1 and is used to determine c_*. Incompressible flows can be found by setting MACH to .001. Subsonic flows result from runs with low values of MACH. To obtain transonic flows, MACH must be chosen so that the supersonic region is of adequate size, since the code is most sensitive near the surface $M^2 = 1$. Raising MACH affects the resulting design by increasing the size of the supersonic zone and the free stream Mach number and decreasing the thickness-to-chord ratio and the vertical separation at the trailing edge. It should have little effect on the lift or the horizontal separation at the tail. Too large a value of MACH results in grids which intersect the sonic surface. In this case the characteristic equations become singular and the correct solution will not be found.

The factor RATC is used to improve the convergence of the series used in the map from the ellipse to the hodograph plane. This map from (ξ,η)-space to the (s,t)-coordinates of hodograph space is given by

$$s = \frac{\xi - \xi_0}{\text{RATC}*\xi - \xi_0} \exp f(\xi) \quad , \quad t = \frac{\eta - \eta_0}{\text{RATC}*\eta - \eta_0} \exp \overline{\overline{f(\eta)}}$$

where ξ_0 is the stagnation point and f is a series of Chebyshev

polynomials. RATC must be of absolute value less than one to place the singularity outside of the ellipse and near the stagnation point. Generally, RATC is determined by the code in subroutine CYCLO, but for low values of the ellipse parameter the user may want to adjust this quantity. A lower value of RATC is indicated when the resulting pressure distribution oscillates near the stagnation point.

5.2 The speed distribution

The speed distribution of a cascade of lifting airfoils in transonic flow is shown in Figure 7. On the upper surface the speed rises from stagnation to supersonic values, which extend in a horizontal arc along the first 25-40% of chord and then fall to a Stratford distribution near the tail. Along the lower surface the speeds are assigned negative values and are slower than those of the upper surface. In Figure 7 the lower surface speeds are entirely subsonic, although supersonic speeds may be attained, as shown in Figure 16. In this case the lower surface speed distribution more closely resembles the upper distribution.

When designing a profile it will be necessary to modify the input speed distribution to achieve given specifications. The lift of the blades may be raised by increasing the area between the Mach distribution on the upper and lower surfaces, which is shown in Figure 13. The leading edge radius and the thickness-to-chord ratio are decreased by raising $q'(s)$ at stagnation. Separation is determined by the slope of the upper surface speed distribution near the trailing edge which must be adjusted to keep SEP constant and near .003 in order to prevent boundary layer separation.

The trailing edge of the profile should terminate in two parallel horizontal arcs which are perpendicular to the line joining their end points. For the cascade the thickness of the trailing edge should be approximately 2% of chord before the boundary layer correction. In order to achieve this, the speed can be modified to change DX, the

horizontal distance between the upper and lower endpoints, and the vertical distance DY. DX is adjusted by rescaling the arc length on the upper or lower surface of the airfoil. A large value of DX indicates an excess of upper surface arc length which will be diminished by decreasing s on the upper surface or increasing it on the lower surface. Small values of DX indicate a greater proportion of lower surface arc length and are adjusted by increasing s on the upper surface or decreasing it on the lower surface. DY increases with the value of q at the trailing edge.

When performing runs to obtain a desired geometry it is advisable to adjust the speed distribution in stages so that the effects of each change can be observed. Most of these changes have little effect on the gap-to-chord ratio, which depends primarily on the location of the singularities XIA and XIB.

5.3 The parameters in the ellipse

Solutions to the transonic flow equations having a given speed distribution q(s) are determined uniquely by specifying 3 real constants in the ellipse plane $\xi = \bar{\eta}$. In the case of the airfoil, these are the mesh point NOSE assigned to stagnation and the complex singularity XIA. In the case of the cascade both XIA and XIB, which correspond to the exit and inlet velocities respectively, are prescibed by the user, as well as the mesh point NOSE. Thus a total of 5 real constants are prescribed, which overdetermines the map from the (ξ,η)-to the (s,t)-plane. For this reason, the selection of these points will determine some physical property of the flow which we could otherwise prescribe. In our case this property is the gap and the stagger of the blades. If a particular gap-to-chord ratio and orientation of the cascade is desired, it must be obtained by a series of runs which adjust the parameters in the ellipse. We discuss here how to vary these parameters, and the ellipse parameter EP, in order to construct vertical arrays of thin compressor airfoils with gap-to-chord ratios near .5 and maximum Mach numbers near 1.2.

Lowering EP decreases the gap-to-chord ratio, as does increasing the distance between XIA and XIB. A small value of EP also increases the distribution of points in the physical plane corresponding to the ellipse boundary near the y-axis and makes the solution more sensitive to the placement of XIA, XIB, MACH and NOSE. The orientation of the blades is determined by the angle of the branch cut from XIA to XIB relative to the stagnation point. It is useful to keep XIA and XIB

close to the y-axis because of the increased resolution near the foci of the ellipse, and to vary NOSE instead. The effect of moving this mesh point in the clockwise direction for fixed XIB is to increase the camber of the blades and to rotate them counterclockwise. The location of XIC determines the initial planes for the series of regular solutions, but does not affect the resulting flow.

Since the problem to be solved is nonlinear, these parameters combine in a complicated fashion and a series of runs will be required in which each parameter is changed separately. We recommend here the following procedure for obtaining a good design. Given a speed distribution, determine the value of MACH which produces the desired Mach numbers along the profile. In the following runs, MACH may then be lowered slightly to avoid possible difficulties with the sonic surface. Next, place XIA and XIB at points which result in the proper gap-to-chord ratio, and then change NOSE to adjust the orientation and curvature of the blades. These steps may be repeated several times. Because the location of the stagnation point must be changed by a discrete amount while the singularities vary continuously, fine modifications are more easily obtained by moving the latter. MACH may then be increased to its desired value or until difficulties arise with the sonic surface. Finally, the speed distribution is changed to yield the desired thickness, closure and lift.

5.4 Difficulties with the code

We describe here certain difficulties which may arise in the operation of the code and methods for resolving them. The first problem concerns the location of points on the sonic surface $M^2 = 1$. At such points the characteristic equations fail and a correct solution will not be obtained. Hence the transonic and supersonic paths are constructed to avoid them. If such a point appears on the rectangular grids as the last point along a ξ-characteristic at which the solution is computed, the inaccuracy of the solution at the point will not affect the computation in the rest of the grid. However, if the sonic point occurs at an interior point of the grid, the resulting error will be transmitted to all other points on the characteristic.

Points on the sonic surface may be detected on the plot in Figure 3, which comprises the ellipse in the ξ-plane and the $\bar{\eta}$-plane with the sonic locus and the paths of integration. In the figure, points on transonic and supersonic paths which result in sonic points have been indicated. As was discussed in Section 3.4, a point on the sonic surface results from a pair of points on the initial paths in the planes $\xi = \xi_A$ and $\bar{\eta} = \bar{\eta}_A$ which are approximately the reflections of one another across the sonic surface. This rule of thumb will often allow the user to locate sonic points using the graphic of the ellipse (Figure 3).

Points near the sonic surface at which the Mach number is close to 1 may also result in numerical errors. For this reason, these points are listed in the printed output. If at a point (ξ, η) the quantity

$|\sqrt{1-M^2}|$ is less than ε, both the point and the root are printed in a table. This information may be used to modify the paths of integration in a manner described in the next section.

A second difficulty concerns the branch of $\sqrt{1-M^2}$. The branch assigned by the complex function CSQRT in the code places the root in the same half plane as the root of the preceding point on the path. For Mach numbers near 1, this may be incorrect and the wrong branch of the root will result along the entire characteristic. Numerical studies have shown that the branch of the root z should be such that $Im(z) > -Re(z)$. This might be expected since the correct branch of the root in the subsonic case is positive imaginary and in the supersonic case is positive real. If the root is computed to lie below this line the code prints the diagnostic

CORRECTION OF THE BRANCH AT XI = (,), ETA = (,)

and the opposite branch of the square root is chosen. This diagnostic occurs in LAMBDA and LAMBD, the subroutines which compute τ, λ_+ and λ_-, the coefficients for the canonical equations and the characteristic equations for u and v.

A third difficulty involves the branch of the logarithms in the singular solutions for Φ and Ψ. The paths are constructed by the code to traverse the singularities properly and to supply the necessary jumps when a path encircles a singularity. It is possible, however, for paths to cross branch cuts. The following diagnostics have been inserted to indicate this:

TRANSONIC PATH CROSSES CUT FROM XIA TO XIB

TRANSONIC PATH CROSSES CUT FROM XIB TO XIC

In the case of a subsonic or supersonic path, similar diagnostics appear.

A final difficulty concerns the location of the paths of integration. When MACH is high and the points XIB and XIC are close to the ellipse boundary, there is little room in which to construct paths around the singularity and still avoid points on the sonic surface. In some cases this may not be accomplished by the paths in the code. The user may then construct the paths manually by changing a few instructions in the code. This procedure is described in the following section.

5.5 Changing the paths of integration

The paths of integration are constructed by the subroutines GTPATH, SUPATH, and PATH. GTPATH determines the location of subsonic and transonic paths, while SUPATH is concerned with supersonic paths. Subroutine PATH actually constructs the paths of integration and is called by the two other subroutines.

Calls to PATH are of the form:

CALL PATH (RAD,TH1, TH2, K1, K2).

PATH will then construct an arc from XI(K1), which was previously the terminal point of the path, to a final point XI(K2). This arc is constructed as follows: let XIF be the point in the ellipse corresponding to the point RAD*(exp(i*TH1)) and let XIL be a second point in the ellipse corresponding to the point RAD*(exp(i*TH2)). PATH first constructs a straight line from XI(K1) to XIF. If XIF=XIL this completes the arc, so that XI(K2)=XIF. If not, a second arc is constructed along the boundary of the ellipse with foci ± 1 between the points XIF and XIL, terminating at XI(K2)=XIL.

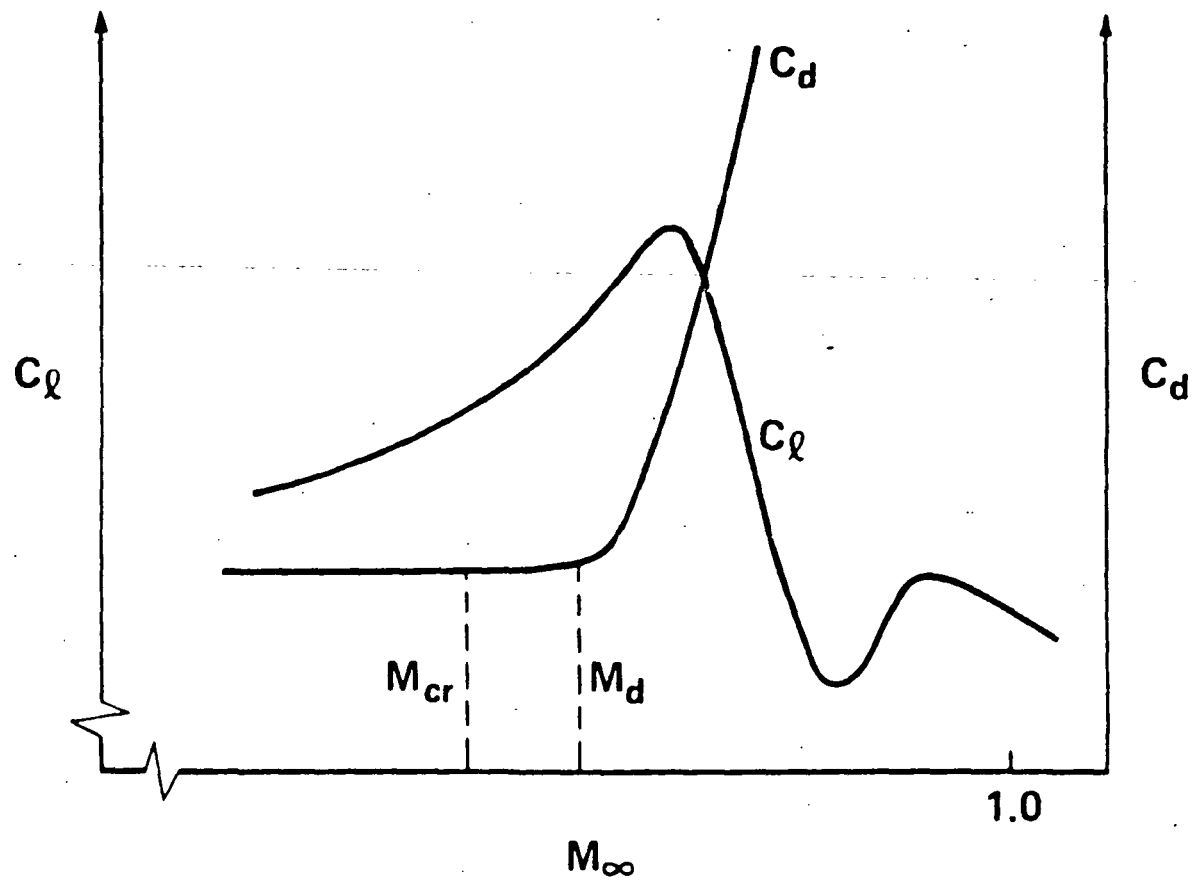
In Subroutine GTPATH a subsonic path of integration is first constructed consisting of an arc joining XIC to the ellipse boundary together with the required portion of the boundary. This is accomplished with a single call to PATH. GTPATH then tests the speed at all points of the path lying on the ellipse boundary. If the sonic locus is not encountered, the appropriate subsonic path has been constructed. If, however, points beyond the sonic line occur on the boundary, the subroutine replaces this subsonic path with a transonic

path. This is accomplished with two additional calls to PATH. The first consists of a straight line from XIC to a point in the subsonic region at some distance from the sonic line. The second call constructs a straight line to the ellipse boundary together with a portion of the boundary.

In order to alter a particular path, the proper parameters must be included in order to identify it. The paths are numbered from 1 to NP in the counterclockwise direction, beginning with the path containing the stagnation point. The parameter MODE denotes the path in question. Subsonic paths occur when the parameter KC is 0, while for transonic paths KC=1. Since a pair of paths must be constructed for each value of MODE in the transonic case, these are distinguished by the value of FAC. For FAC=1, the transonic η-path is constructed, which traverses the ellipse in the counterclockwise direction. When FAC=-1 the ξ-path is determined. This path travels the ellipse in the clockwise direction.

Figure 3 shows a path constructed by the code and its subsequent manual alteration. Note that in the first figure, reflected points are present on a pair of ξ- and η-paths in the ellipse. The update required in order to alter these paths is given below.

In subroutine SUPATH two sets of calls to PATH construct the supersonic paths. As shown in Figure 2, these paths wind around the sonic loci in order to achieve the right branch of the solution. In order to avoid points on the sonic surface, the paths first extend beyond the ellipse boundary for short arcs and then lie on the boundary. They then terminate along the sonic loci. The ξ-path, which is constructed first, traverses the sonic locus in the clockwise direction, while the $\bar{\eta}$-path traverses the same locus in the counterclockwise direction. These supersonic paths are each constructed using three calls to subroutine PATH, and changes in these calls will alter the placement of a supersonic path.



C_L : LIFT COEFFICIENT

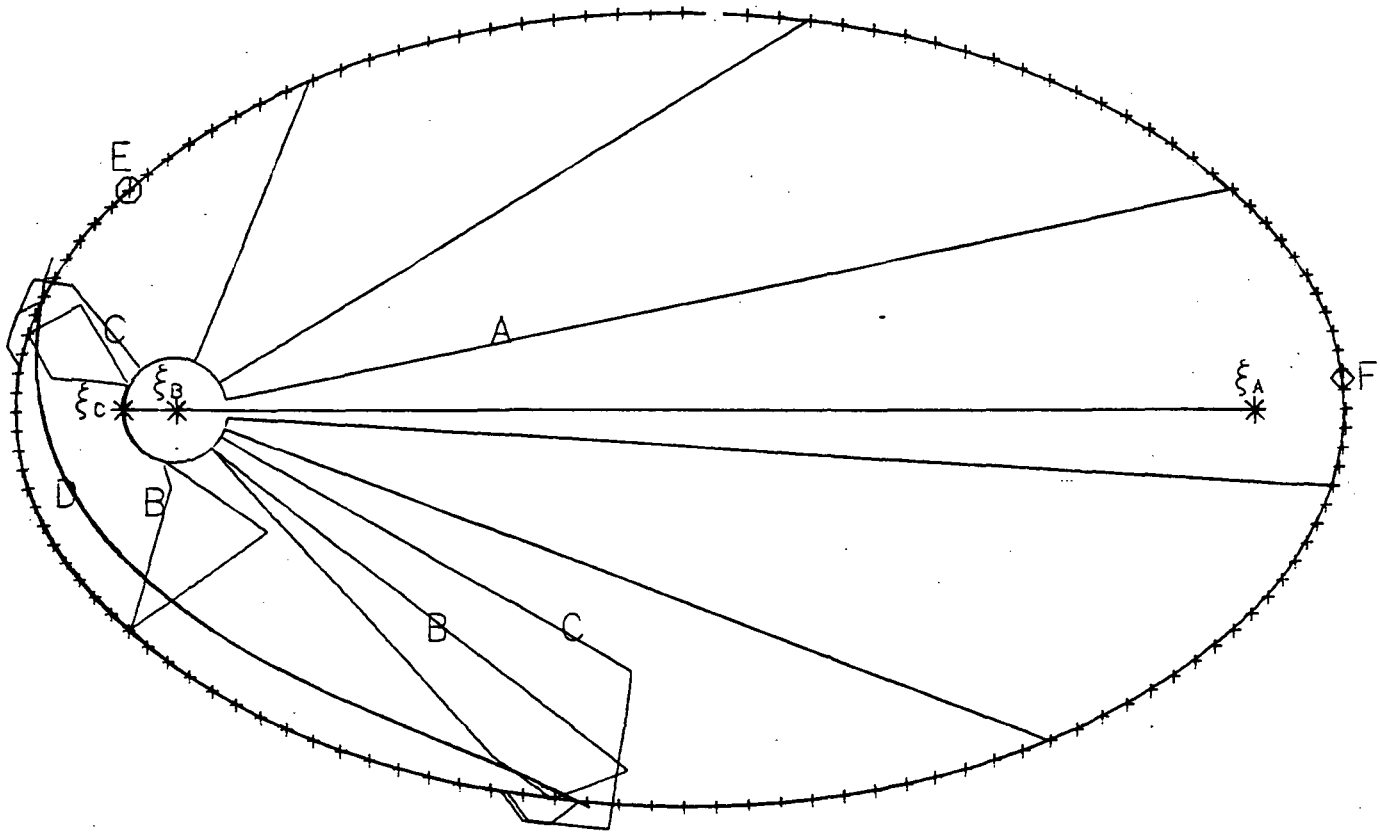
C_D : DRAG COEFFICIENT

M_∞ : FREE STREAM MACH NUMBER

M_{CR}: CRITICAL MACH NUMBER

M_D : DRAG RISE MACH NUMBER

FIG. 1. LIFT AND DRAG AT TRANSONIC SPEEDS



A SUBSONIC INTEGRATION PATH

B TRANSONIC INTEGRATION PATHS

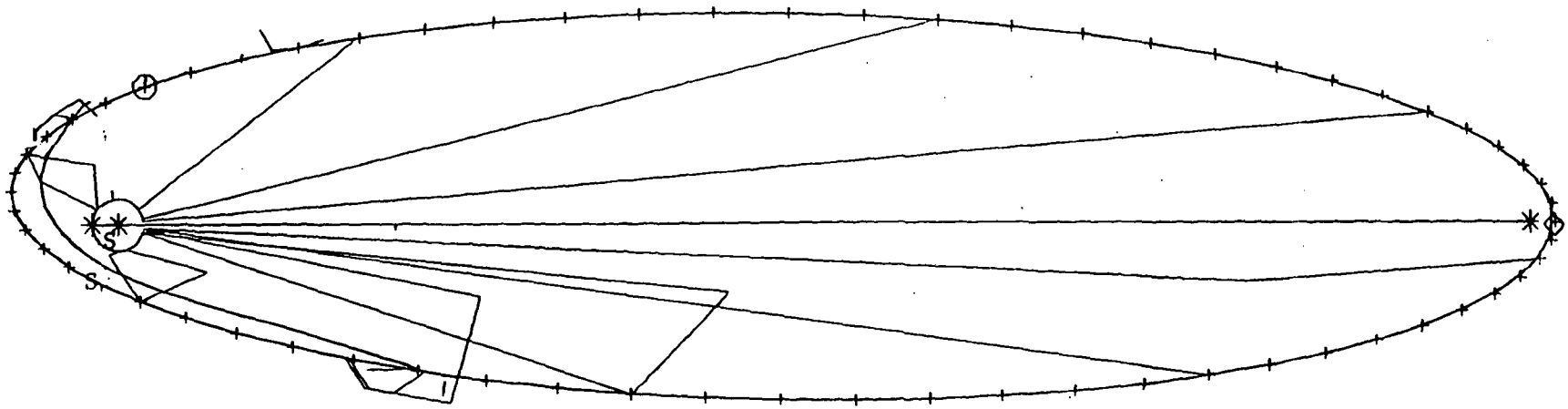
C SUPERSONIC INTEGRATION PATHS

D SONIC LOCUS/SONIC LINE

E STAGNATION POINT

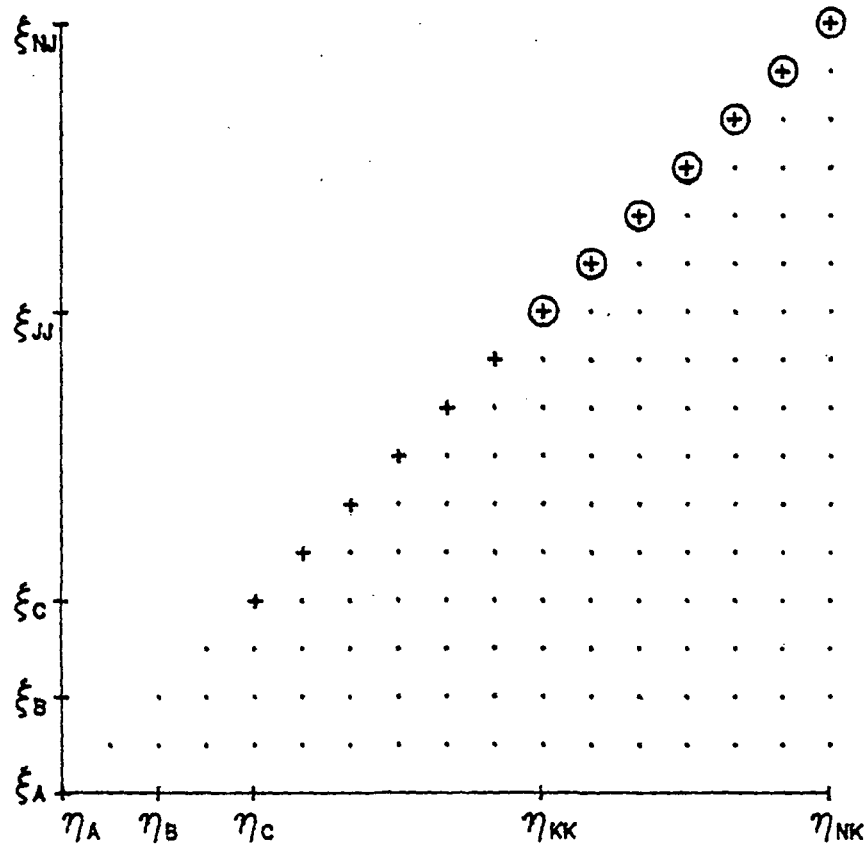
F TRAILING EDGE

FIG. 2. COMPLEX CHARACTERISTIC ξ–PLANE



POINTS LABELED "S" REPRESENT POINTS ON INITIAL PATHS WHICH RESULT IN A POINT ON THE SONIC SURFACE

FIG. 3. POINTS ON SONIC SURFACE

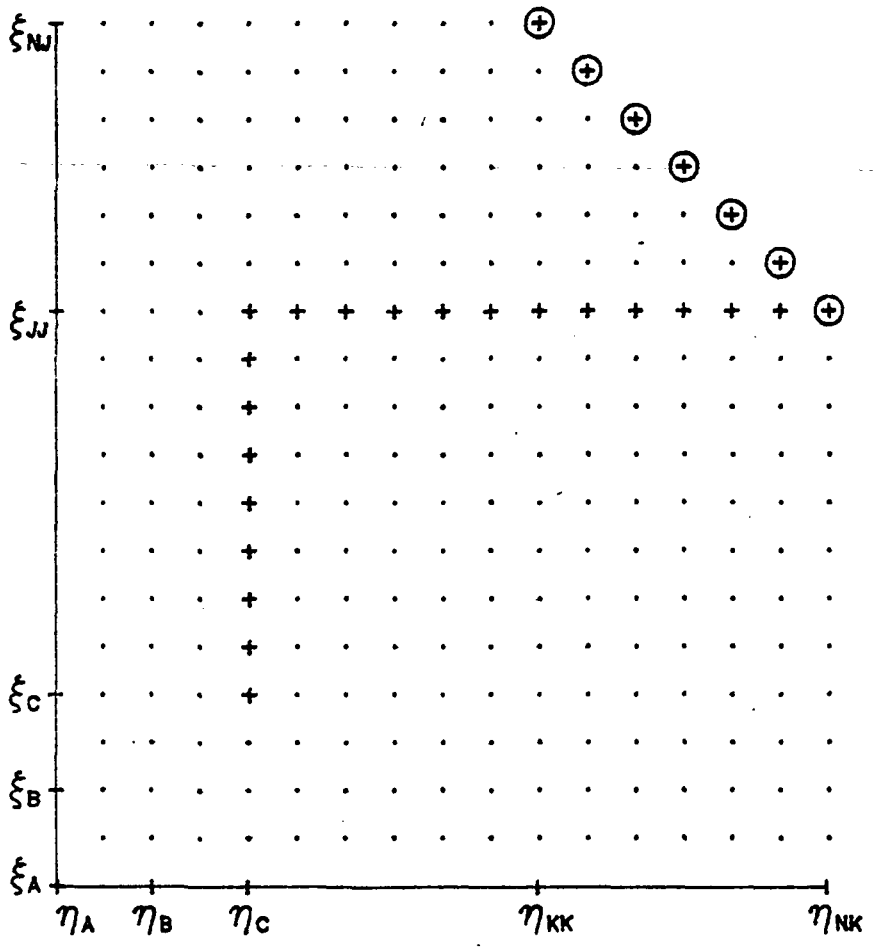


· POINT ON THE COMPUTATIONAL GRID

\+ POINT OF INTEGRATION FOR X AND Y

O POINT ON THE ELLIPSE BOUNDARY

FIG. 4. COMPUTATIONAL GRID FOR SUBSONIC PATH

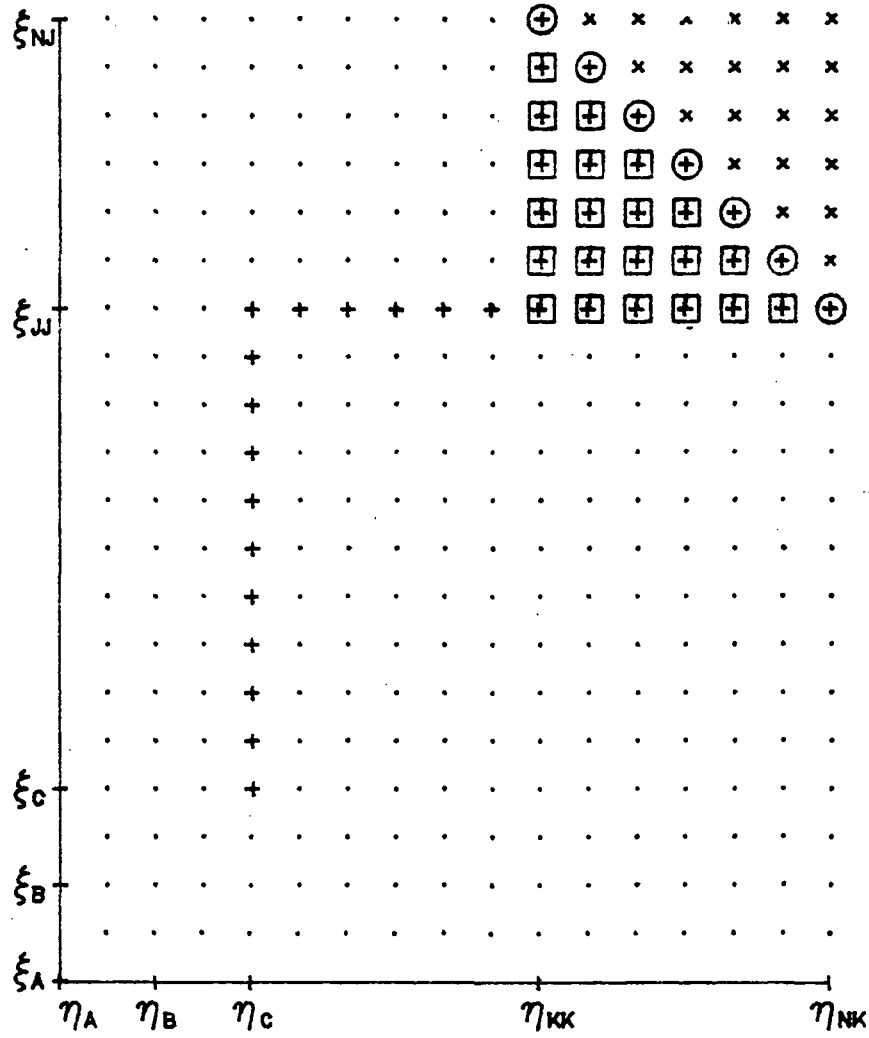


· POINT ON THE COMPUTATIONAL GRID

\+ POINT OF INTEGRATION FOR X AND Y

O POINT ON THE ELLIPSE BOUNDARY

FIG. 5. COMPUTATIONAL GRID FOR TRANSONIC PATH



· POINT ON THE COMPUTATIONAL GRID

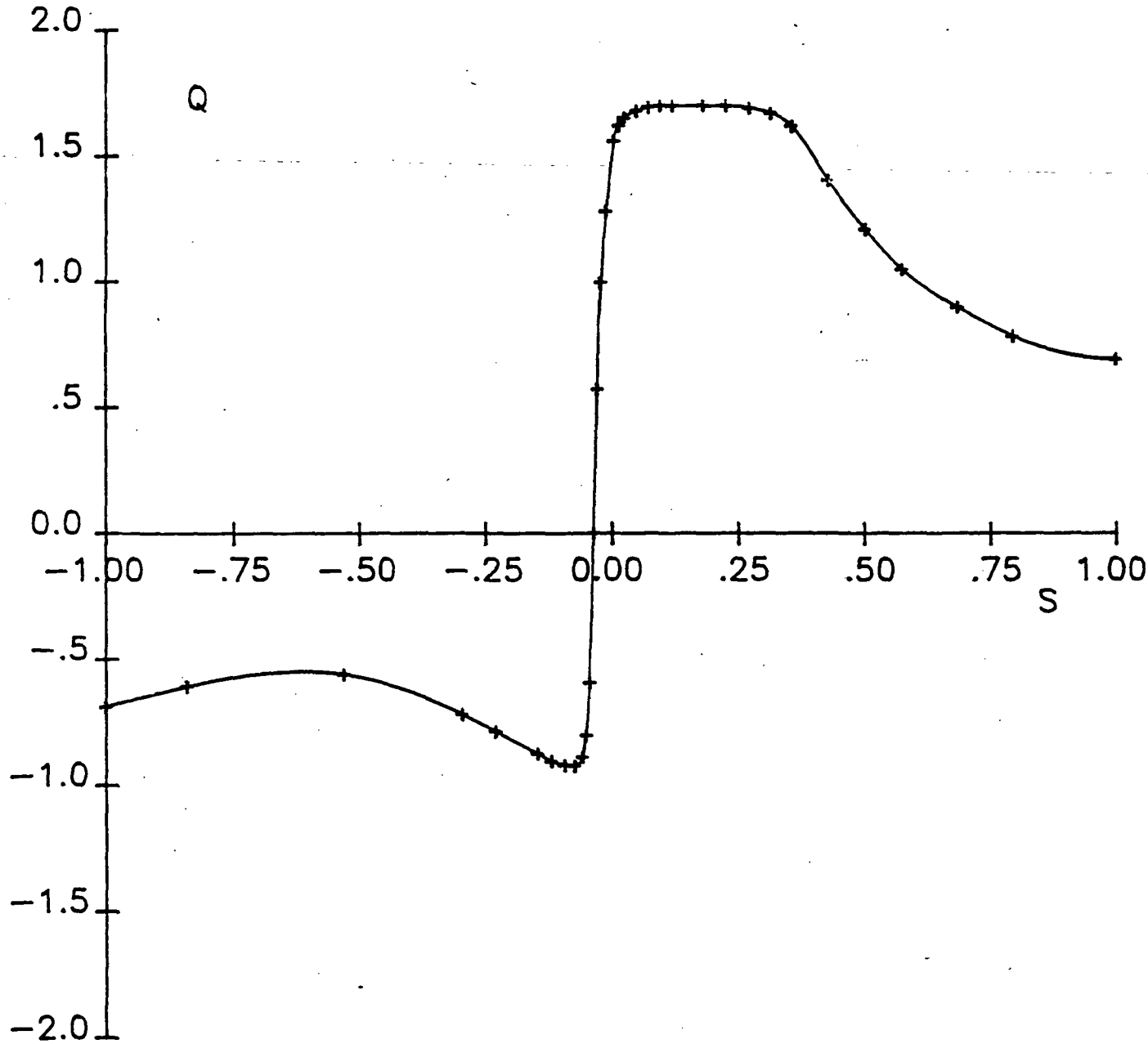
\+ POINT OF INTEGRATION FOR X AND Y

O POINT ON THE SONIC LINE

⊡ POINT IN THE REAL SUPERSONIC ZONE

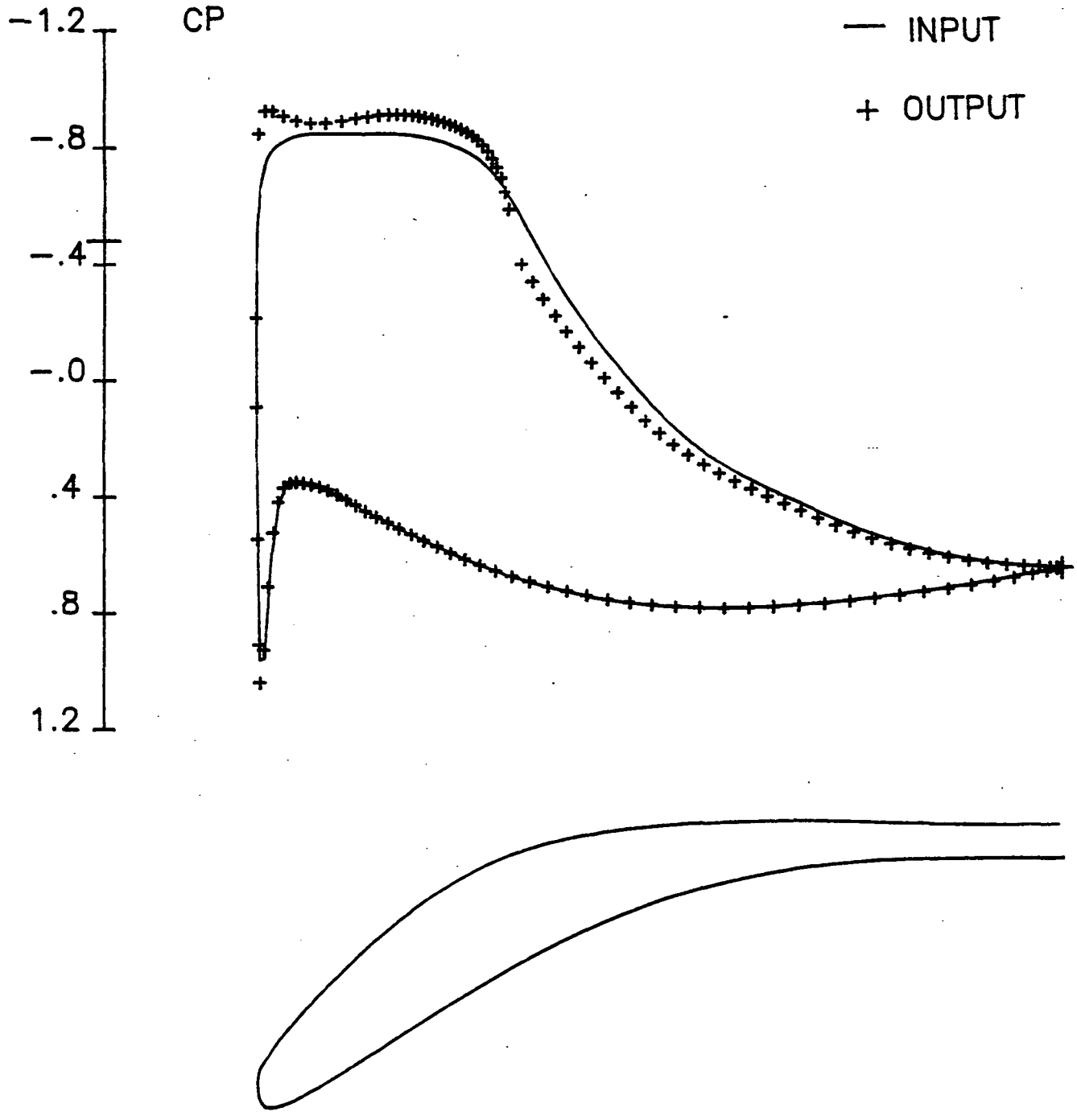
× POINT WHERE SOLUTION CANNOT BE COMPUTED

FIG. 6. COMPUTATIONAL GRID FOR SUPERSONIC PATH



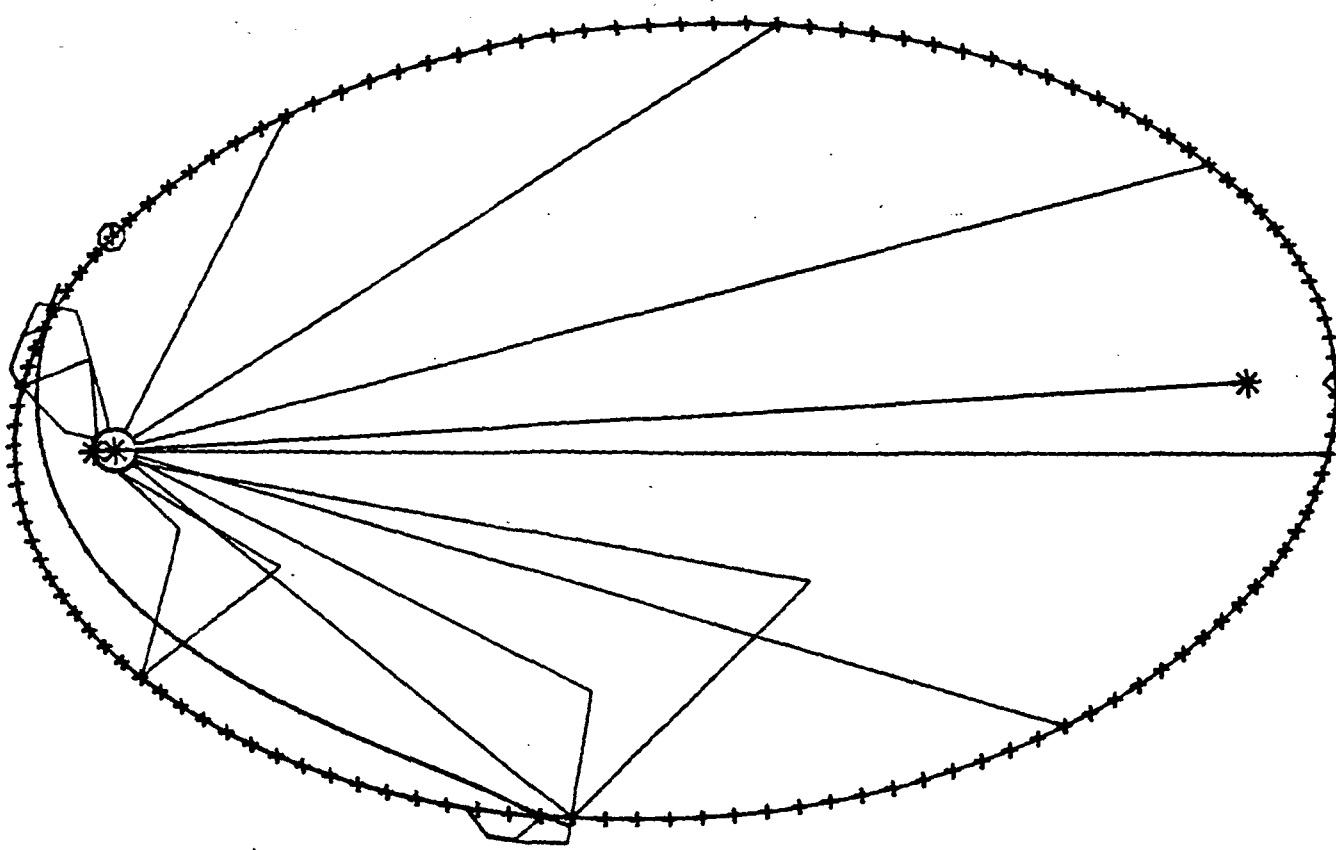
INPUT SPEED DISTRIBUTION

FIG. 7. INPUT Q(S) FOR CASCADE TEST CASE



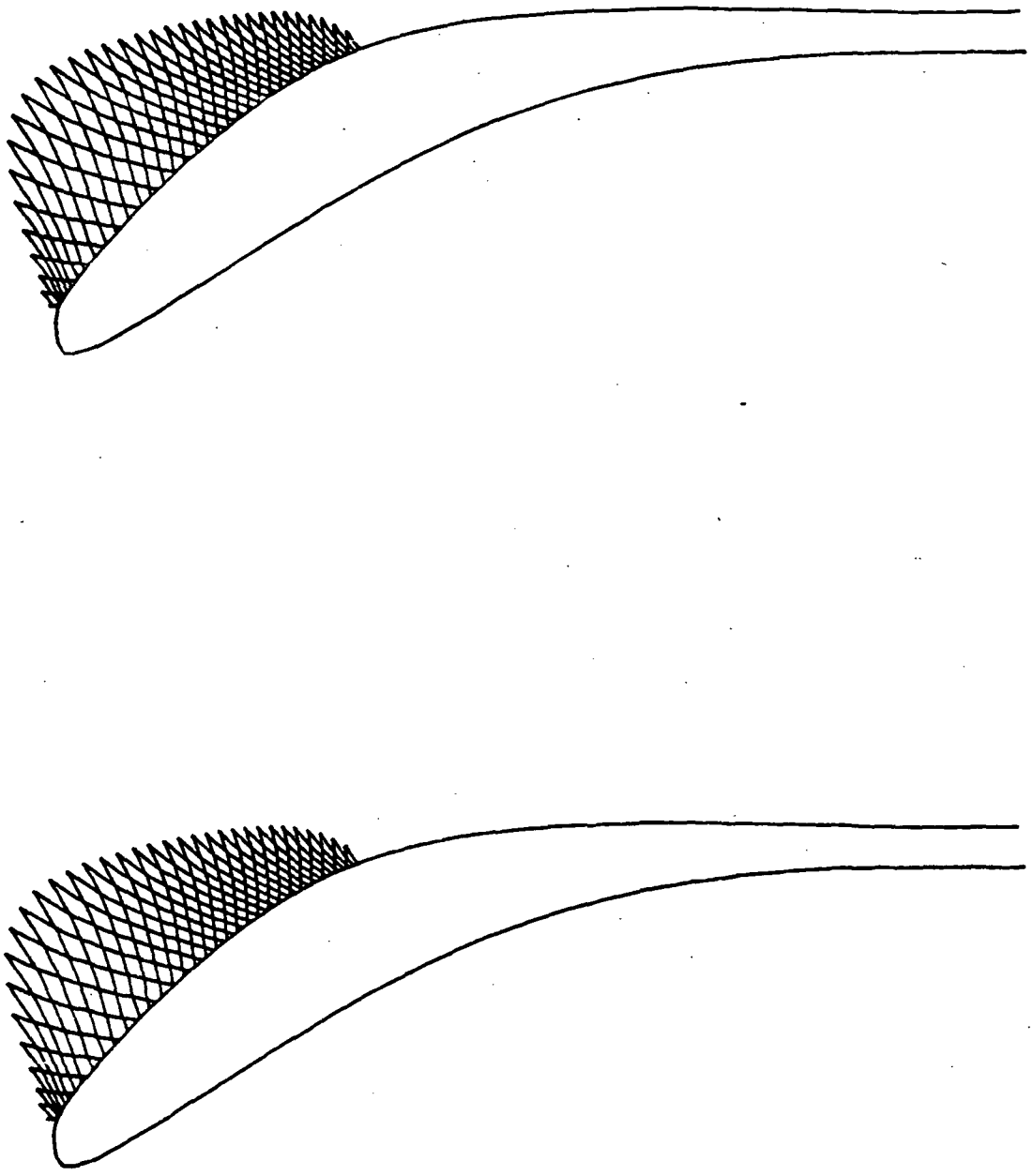
M1= .77 M2= .44 DEL TH= 45. G/C= .81

FIG. 8. TEST CASE AIRFOIL



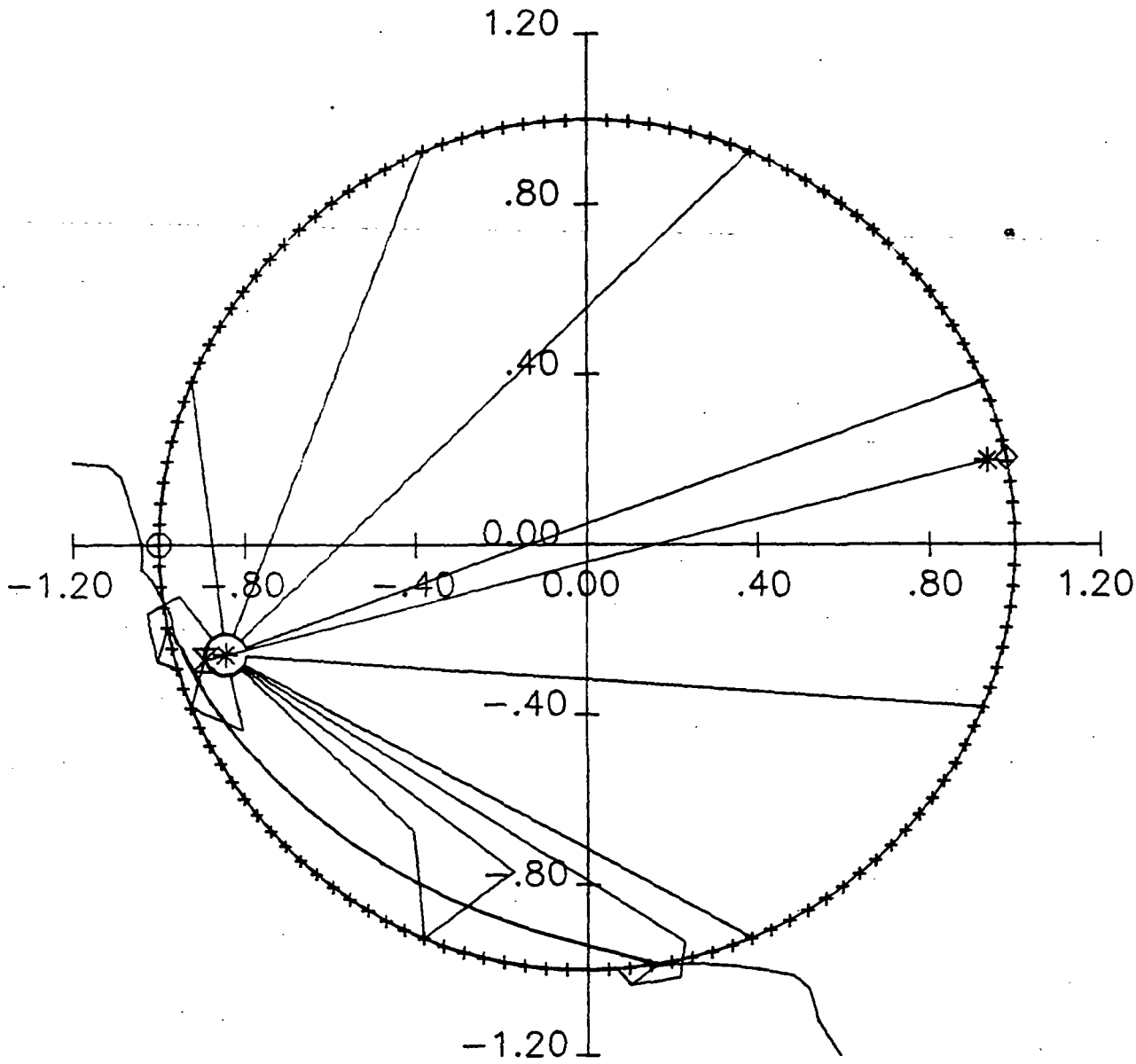
ξ_A = 1.08, 0.00 ξ_B =−1.06, .02 ξ_C =−1.10, .02

FIG. 9. HODOGRAPH PLANE FOR TEST CASE



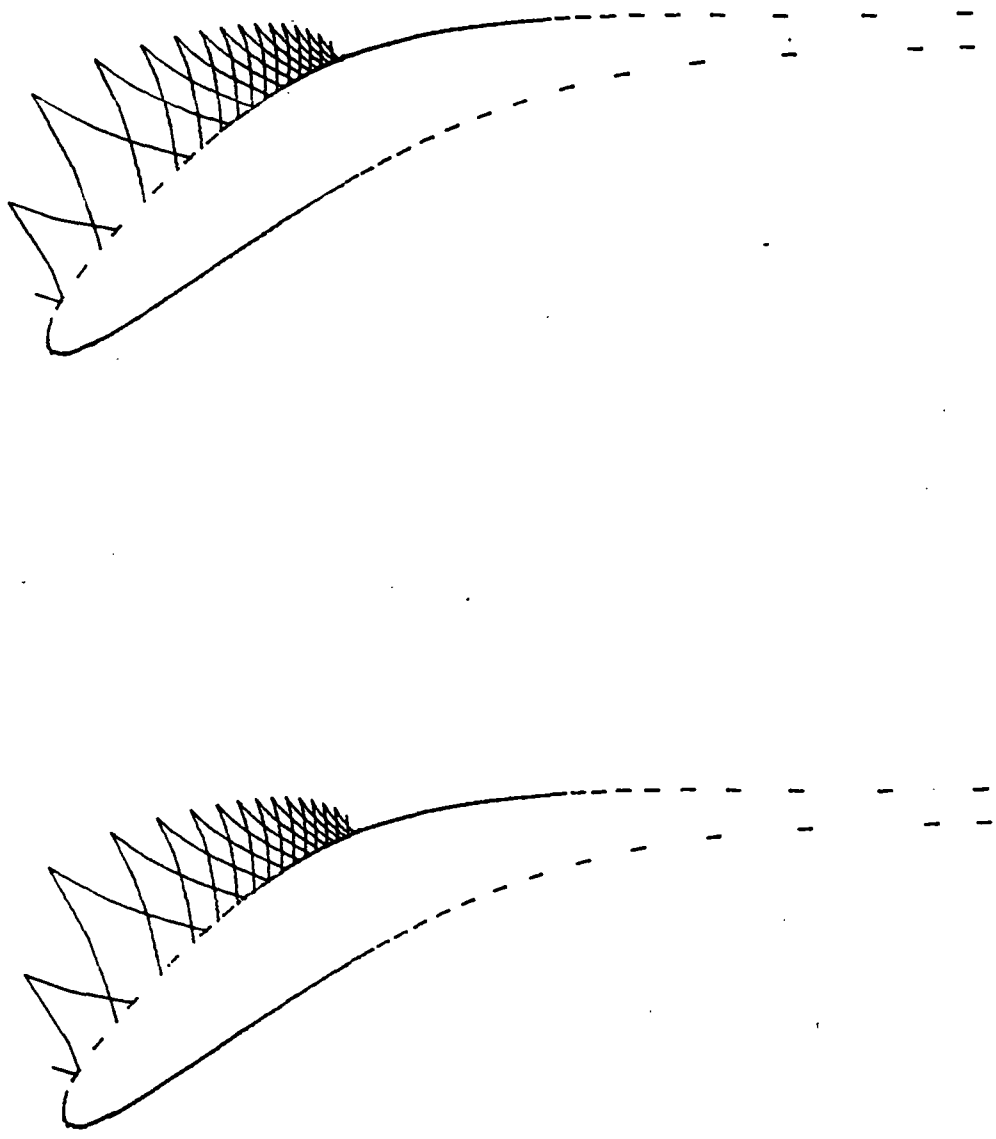
G/C = .807

FIG. 10. TEST CASE CASCADE OF AIRFOILS



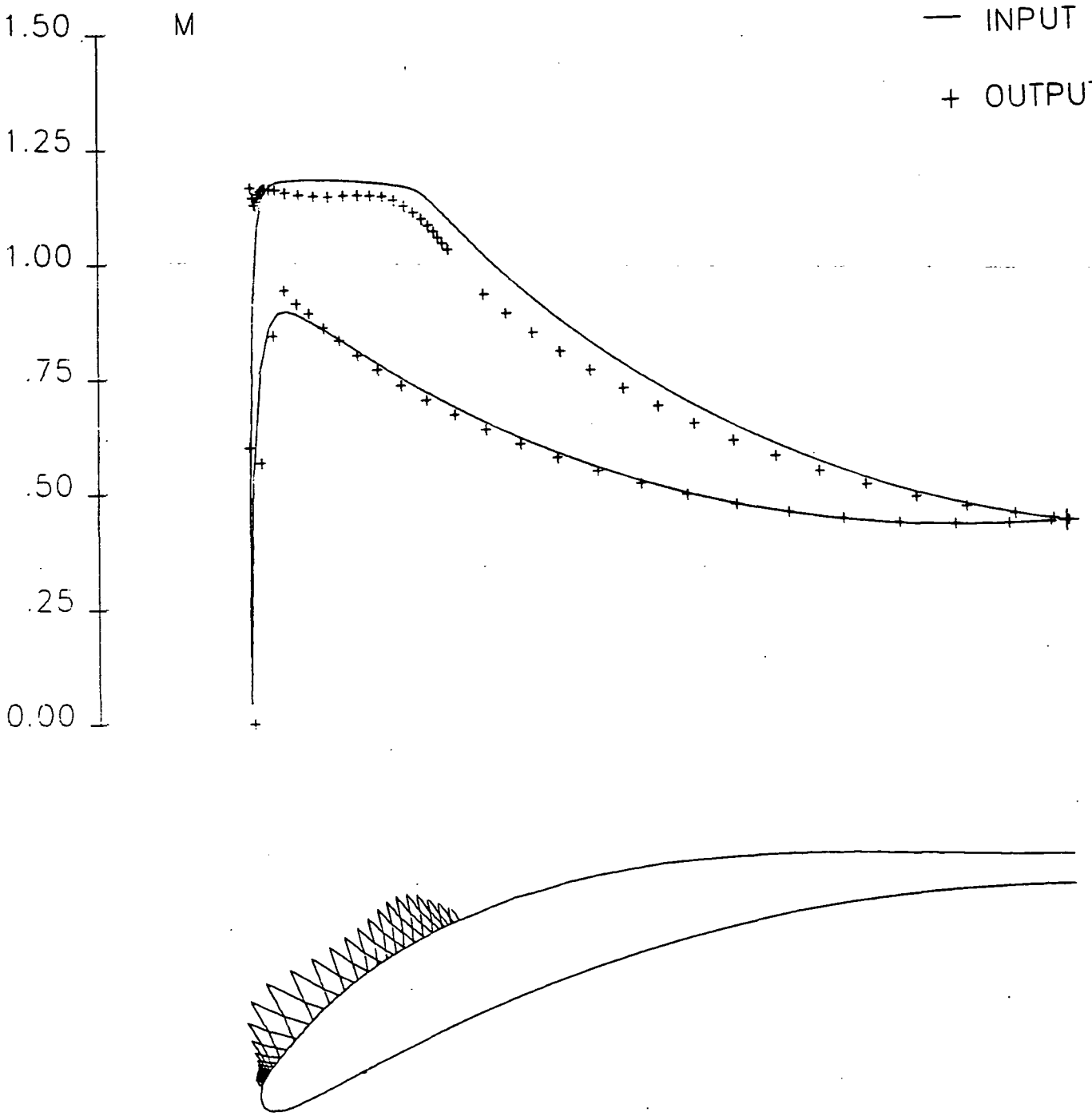
$\xi_A = .94, .20 \qquad \xi_B = -.85, -.26 \qquad \xi_C = -.89, -.27$

FIG. 11. HODOGRAPH PLANE FROM KORN CODE



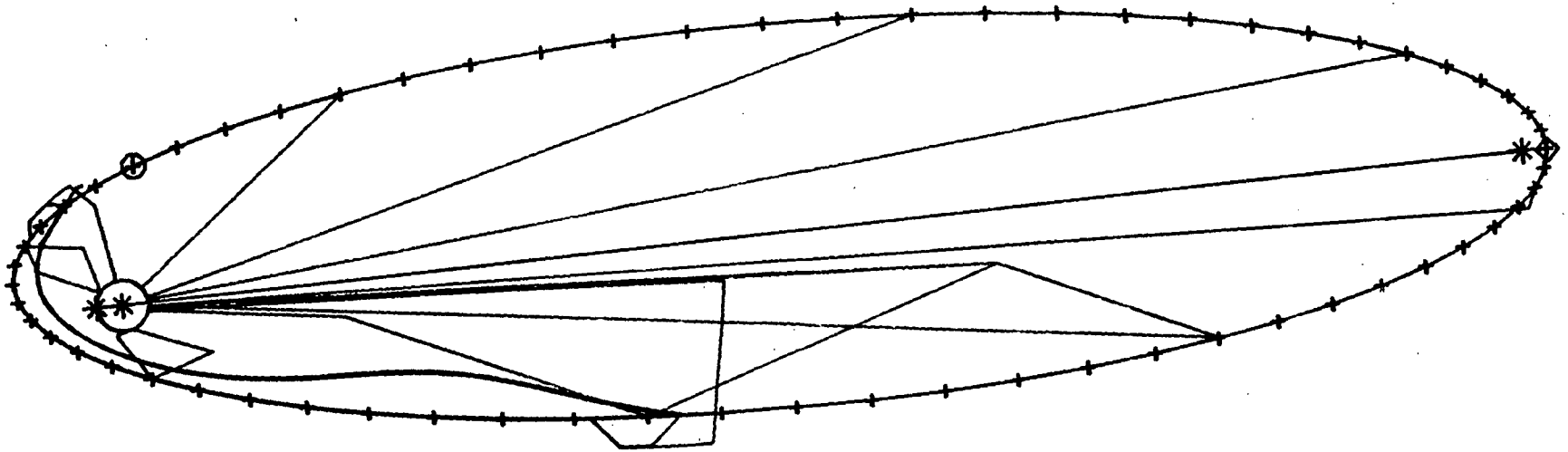
M1=.767 M2=.455 DEL TH= 44.81 G/C= .81

FIG. 12. TEST CASE CASCADE FROM KORN CODE



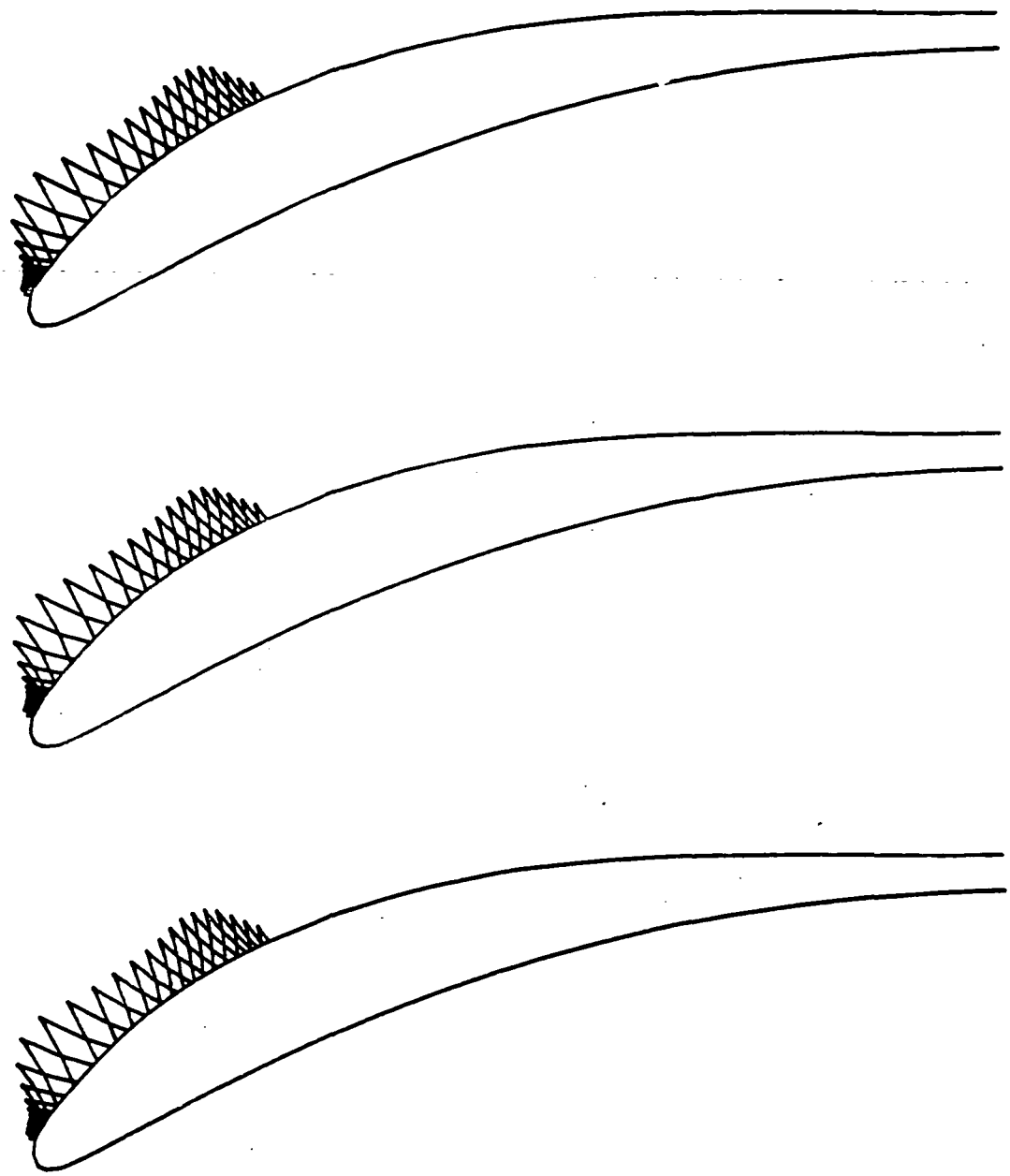
M1= .78 M2= .44 DEL TH= 47. G/C= .43

FIG. 13. LOW GAP-TO-CHORD COMPRESSOR AIRFOIL



$\xi_A = 1.00,\ 0.00 \qquad \xi_B = -.89,\ -.04 \qquad \xi_C = -.93,\ -.04$

FIG. 14. PATHS OF INTEGRATION FOR COMPRESSOR



G/C = .430

FIG. 15. LOW GAP–TO–CHORD COMPRESSOR CASCADE

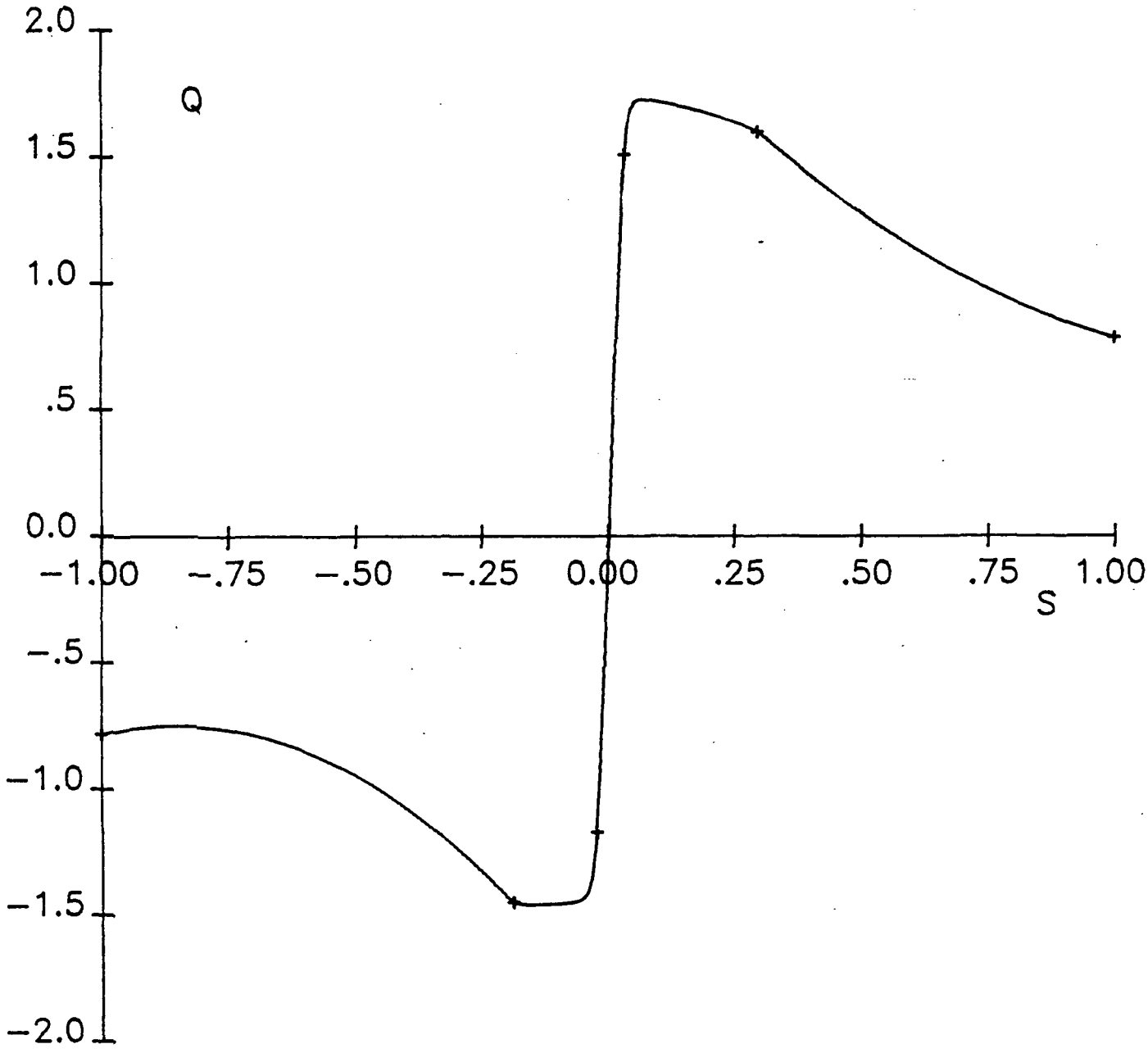
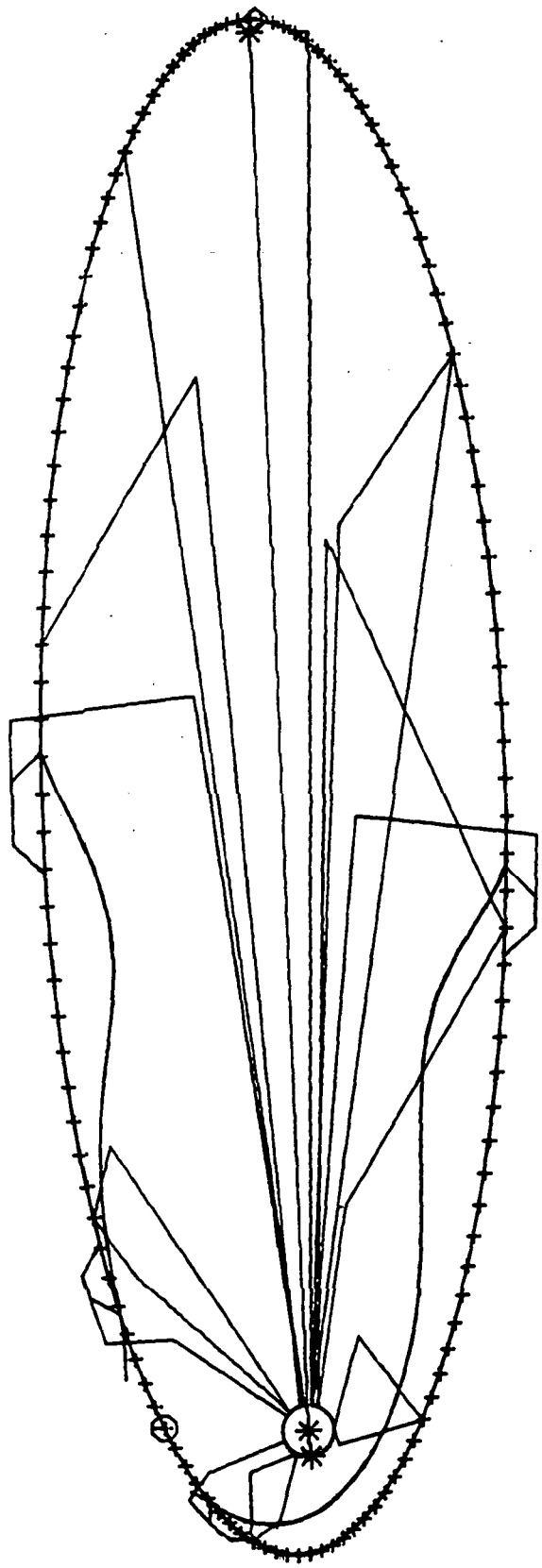
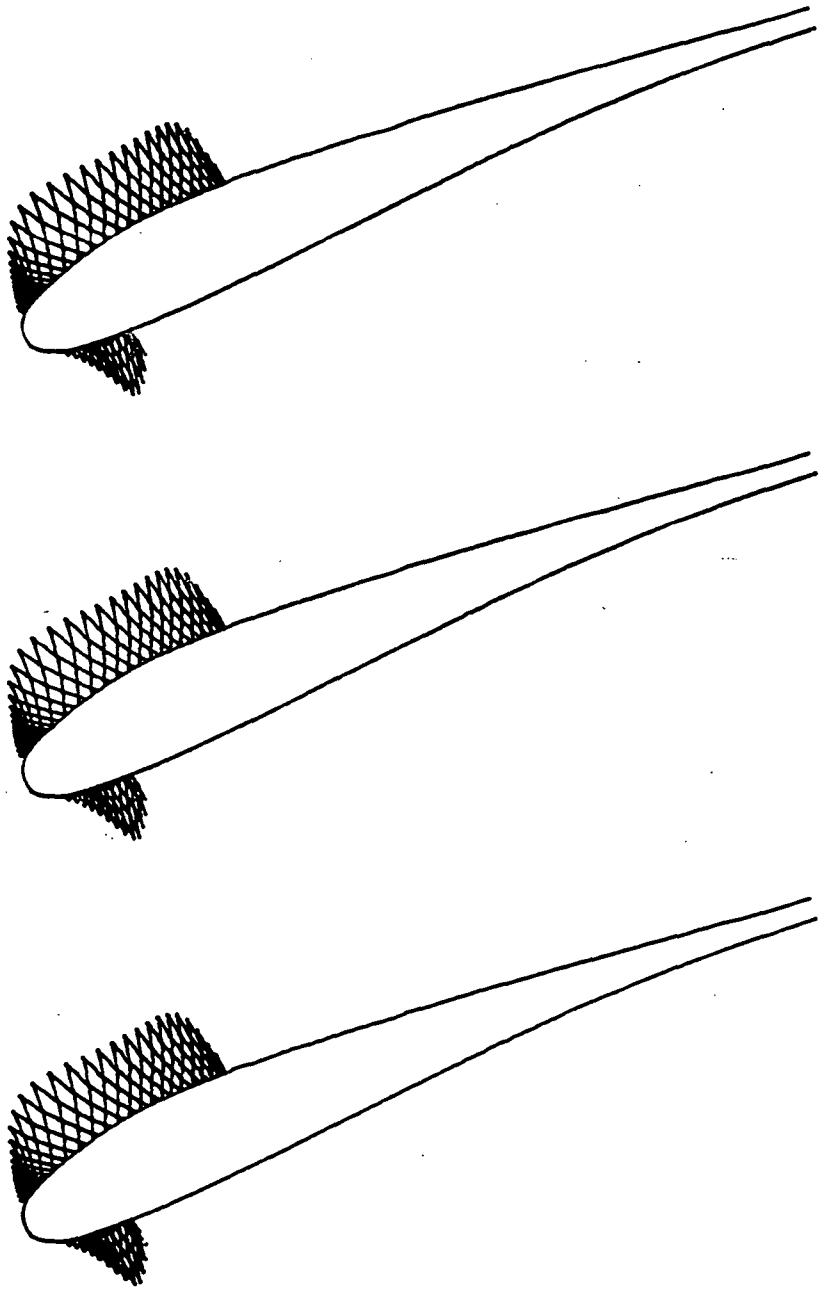


FIG. 16. INPUT FOR CASE WITH TWO SUPERSONIC ZONES



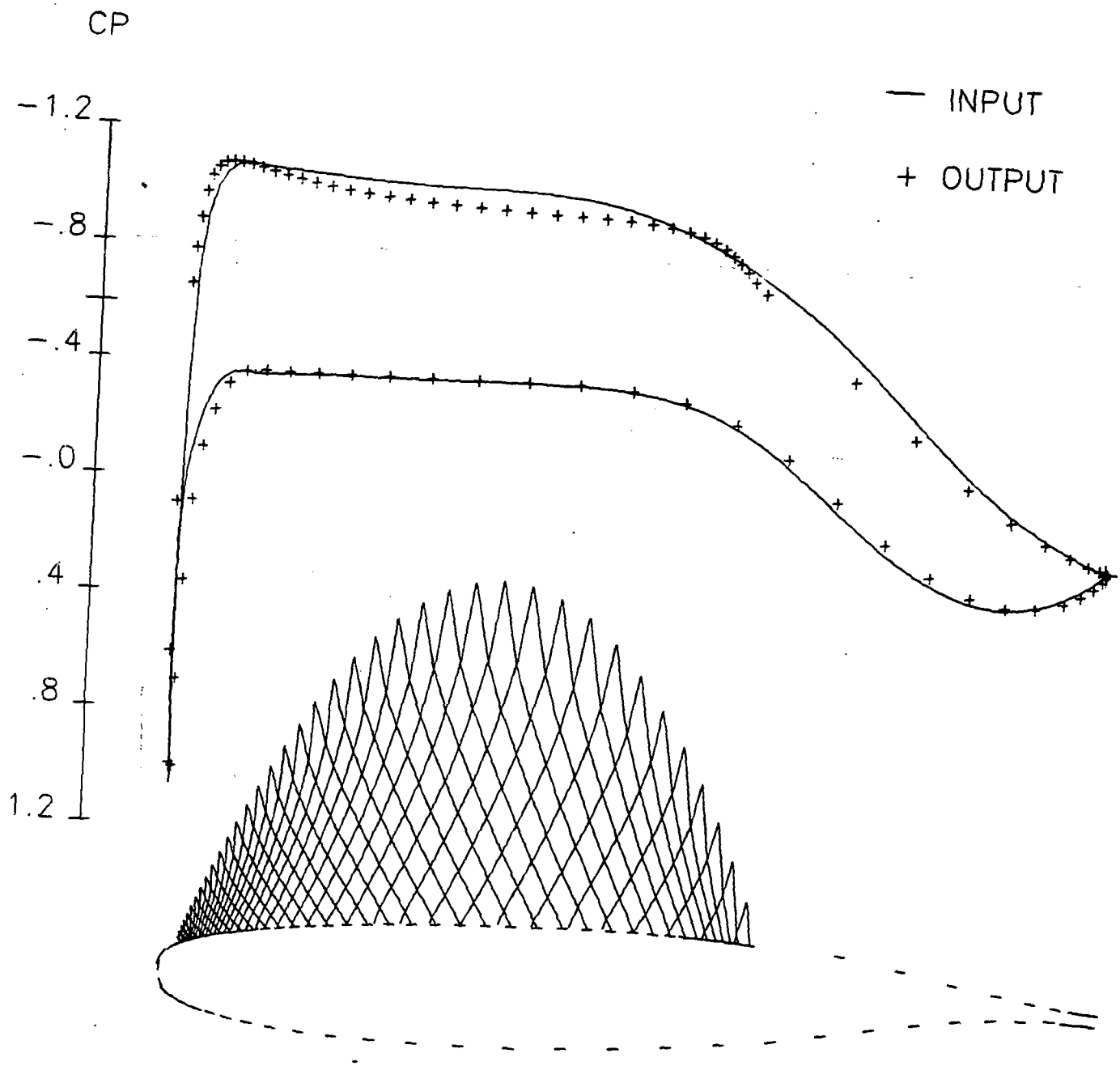
$\xi_A = 1.03, 0.00$ $\xi_B = -.88, -.02$ $\xi_C = -.92, -.02$

FIG. 17. HODOGRAPH PLANE WITH TWO SUPERSONIC ZONES



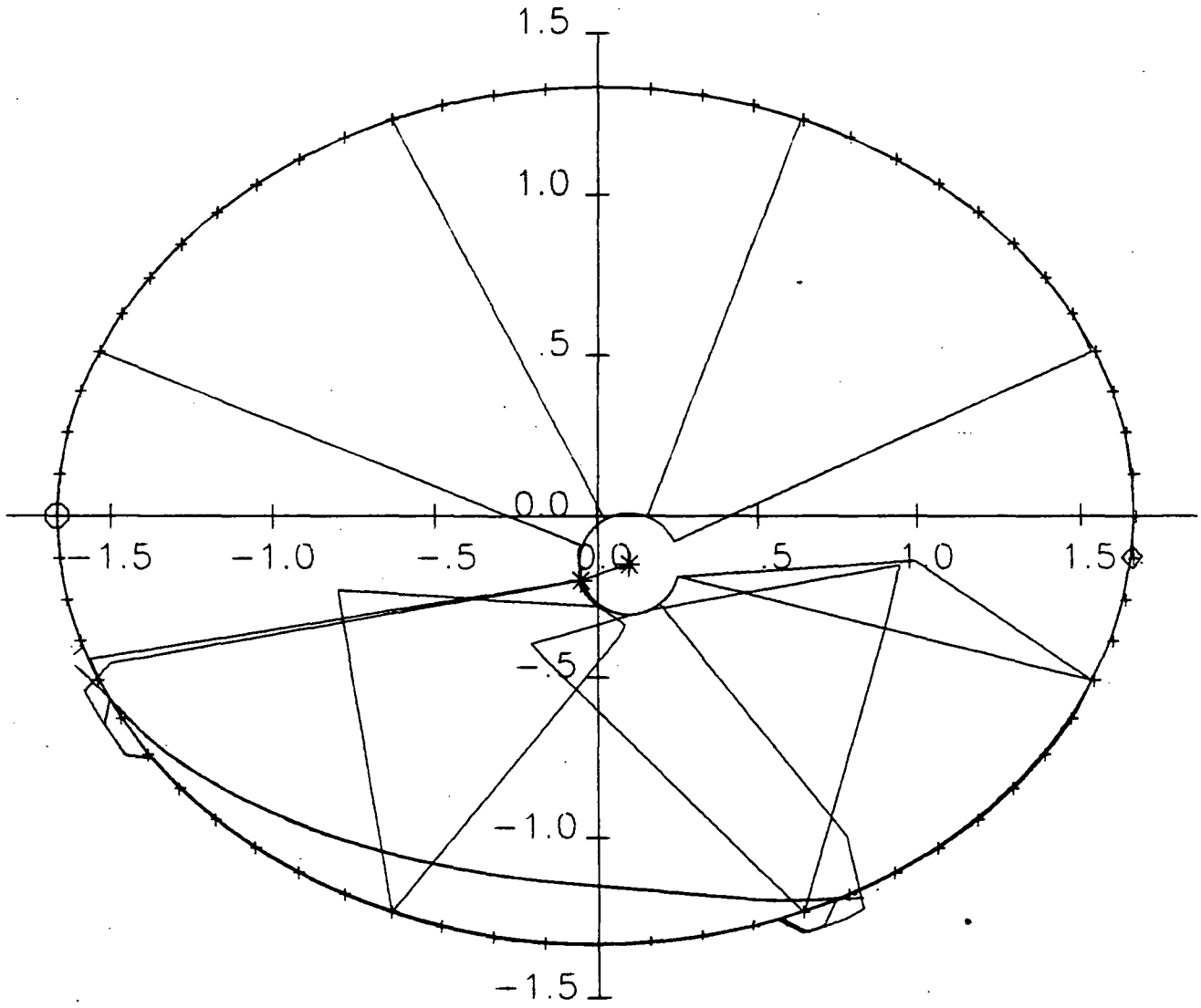
G/C = .543

FIG. 18. COMPRESSOR WITH PAIR OF SUPERSONIC ZONES



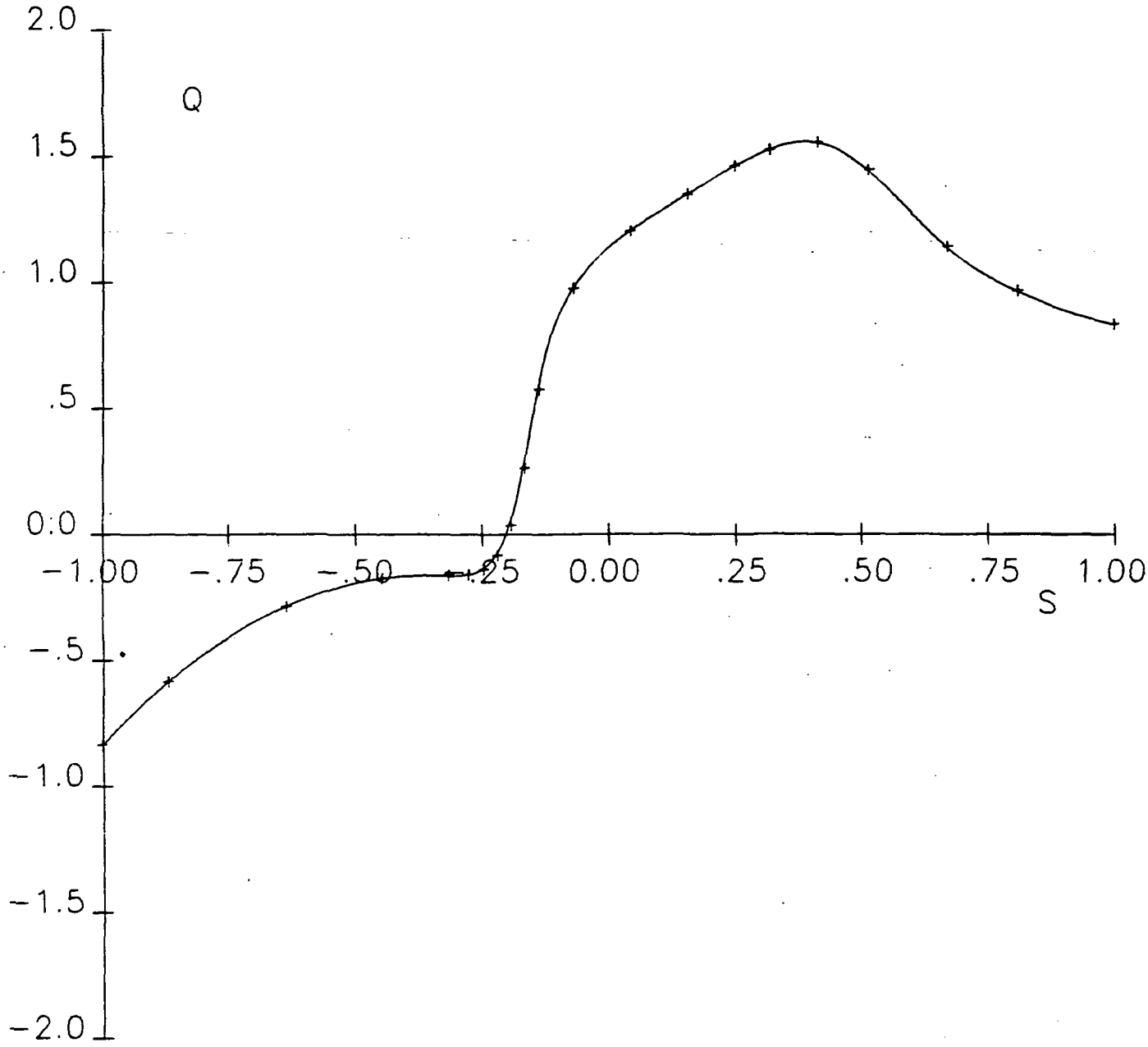
M= .750 CL= .52 DX= -.00 DY= .01 T/C= .13

FIG. 19. MODIFIED WHITCOMB WING SECTION



$\xi_A = .10, -.15 \qquad \xi_C = -.05, -.20$

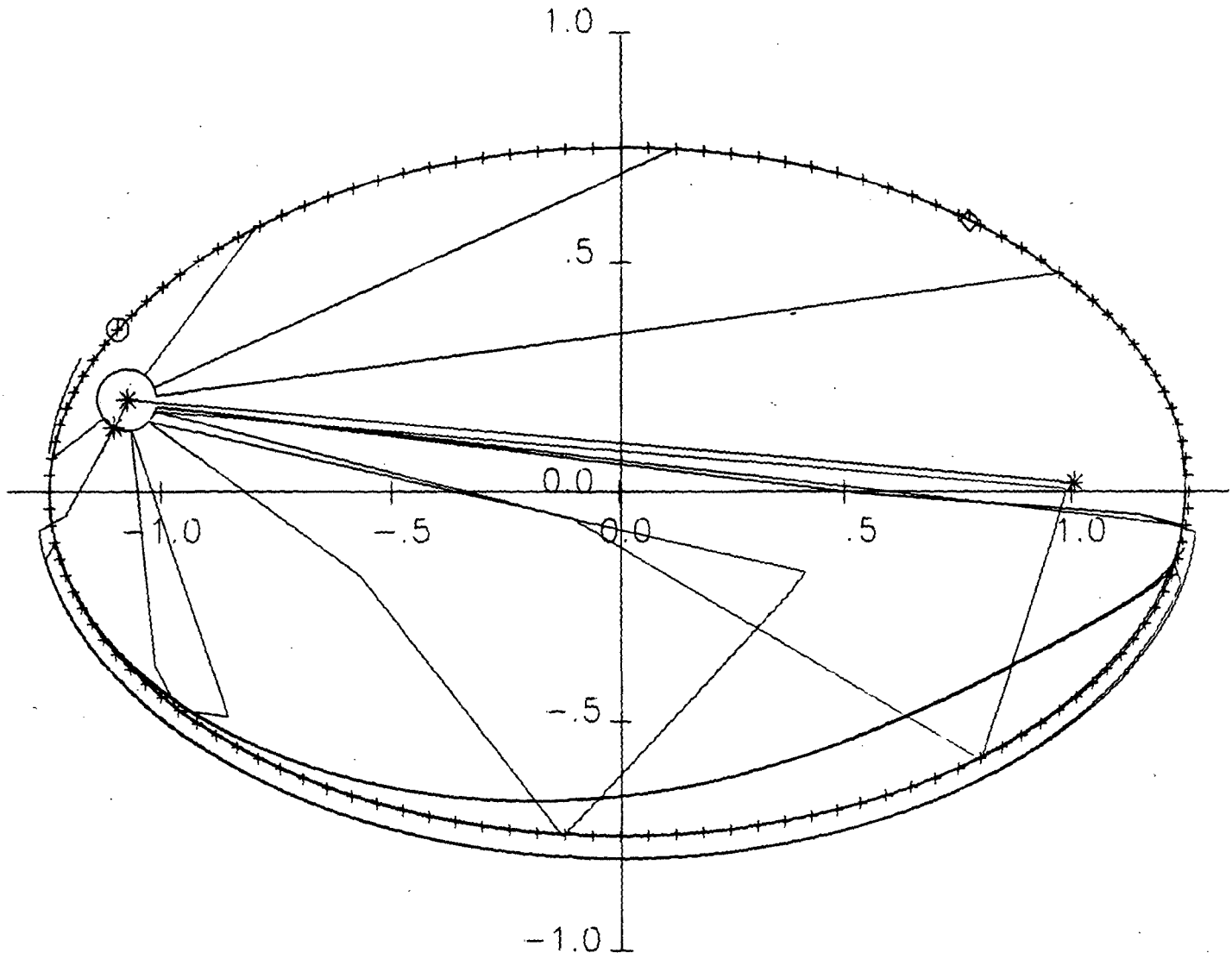
FIG. 20. HODOGRAPH PLANE FOR WHITCOMB EXAMPLE



INPUT SPEED DISTRIBUTION

FIG. 21. INPUT SPEED DISTRIBUTION FOR TURBINE

C-2



ξ_A = 1.00, .02 ξ_B = −1.08, .20 ξ_C = −1.11, .14

FIG. 22 PATHS OF INTEGRATION FOR TURBINE

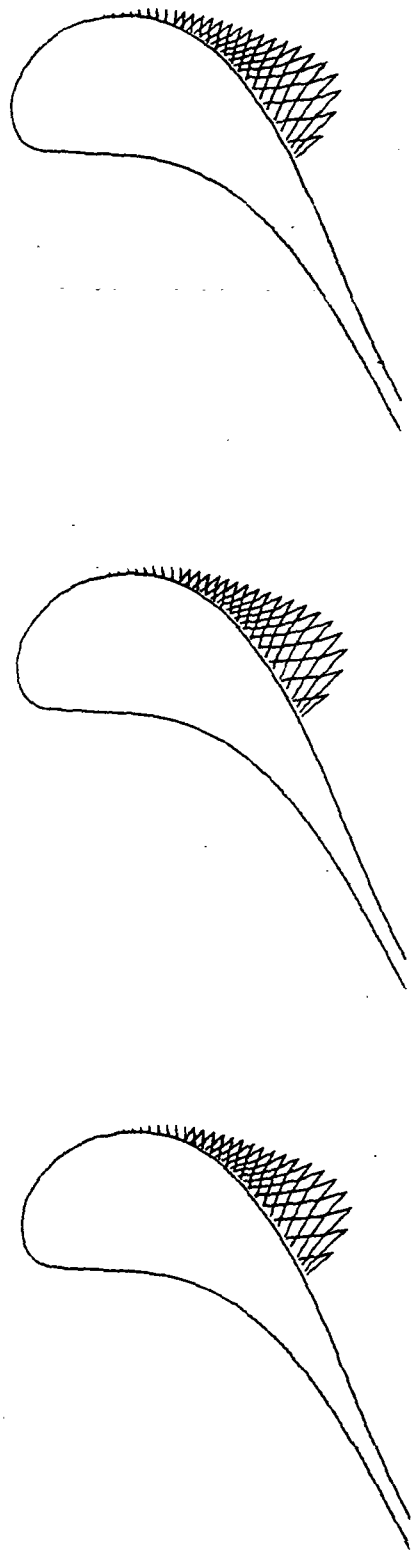


FIG. 23. CASCADE OF TURBINE BLADES

VII GLOSSARIES

7.1 Input parameters

CONE — Parameter used in the extension of the map function across the sonic line. Default = .5

CTWO — Parameter used in the extension of the map function across the sonic line. Default = 1.0

CTHR — Parameter used in the artificial boundary condition along the transonic arc of the ellipse. Default = 1.0

EP — Parameter denoting the shape of the ellipse. EP is the ratio of the minor ellipse axis to the major ellipse axis. Default = .8

GAMMA — The gas constant. Default = 1.4

GRID — Grid spacing parameter. GRID divided by MRP is the maximum mesh size. Default = .08

IPLT — A two digit integer which controls graphics. For negative IPLT, no plots are generated. The first digit controls the plot of the Mach or pressure distribution:

- 0 no plot generated
- 1 airfoil and Mach distribution
- 2 same as 1 but with airfoil higher on the page
- 3-4 same as 1-2 with characteristics
- 5-9 same as 0-4 but with pressure plot instead of Mach plot

The last digit controls the plot of the ξ-plane:

- 0 no plot generated
- 1-2 plots ellipse with sonic locus and paths of integration
- 3-4 same as 1-2 but without coordinate axes
- 5-9 same as 0-4 with contour curves q = constant

For IPLT = 0, only the input speed distribution and the resulting cascade of airfoils are plotted. Default = 34

MACH — The Mach number at q = 1, which is used to determine the critical speed. Raising MACH increases the size of the supersonic zone. Default = .75

MRP — Mesh refinement parameter. Controls the number of points on the computational grid, e.g. doubling it cuts mesh spacing in half. If NI > 1, mesh refinement is done only on the last iteration. MRP negative gives 3rd order accuracy by use of Richardson extrapolation. Default = 1

NCAS — The number of cascades in the cascade graphic. Default = 2

NI — The number of iterations for the map function from the ellipse to the hodograph plane. Default = 1

NFC — The number of Chebyshev coefficients in the expansion of the map function. Use it to define NF = 2*NFC, which is the number of functions in the series representing Φ and Ψ, and the number of nodes on the ellipse. Default = 32

NOSE — Parameter which locates the mesh point on the ellipse corresponding to stagnation. The argument of the stagnation point is Π-NOSE*(Π/NFC). Default = 0

NP — The number of paths of integration in the ellipse. NP must be an even integer that divides NFC. Default = 8

NPTS — The number of points in the spline defining the input speed distribution. Default = 201

NRN — Run number. A negative value of NRN results in white paper plots. Default = 1

RATC — The factor used to improve the convergence of the map function from the ellipse to the hodograph plane. If RATC = 0, it will be determined in subroutine CYCLQ. Default = 0

RN — Reynolds number. If RN = 0 no boundary layer correction is made. Default = 0

TRANU — The value of x (before rotation) at which the turbulent boundary layer integration on the upper surface starts. Default = .05

TRANL — The value of x (before rotation) at which the turbulent boundary layer integration on the lower surface starts. Default = .10

XIA — The point ξ_A in the ξ-plane corresponding to the exit velocity. For an isolated airfoil, set XIA = XIB. Default = (0.,0.)

XIB — The point ξ_B in the ξ-plane corresponding to the inlet velocity. For an isolated airfoil, set XIA = XIB. Default = (0.,0.)

XIC — The point ξ_C determining the characteristic initial plane for the inhomogeneous and regular solutions. The default value for the airfoil is

$$\xi_A - \frac{4 * GRID * (R1 - \xi_A)}{|R1 - \xi_A|}$$

For the cascade it is

$$\xi_B + \frac{\text{GRID} * (\xi_B - \xi_A)}{|\xi_B - \xi_A|}$$

7.2 Output parameters

ANG	Angle of the tangent to the flow along the airfoil.
C_L	Standard lift coefficient.
DEL TH	The change in flow angle from $x = -\infty$ to $x = +\infty$.
DIFFUSION FACTOR	Standard measure of performance of compressor blades.
DX	The horizontal displacement before boundary layer correction from the lower to the upper trailing edge.
DY	The vertical displacement before boundary layer correction from the lower to the upper trailing edge.
EXIT FLOW ANGLE	The clockwise angle between the vertical and the flow at $x = +\infty$.
G/C	Gap to chord ratio. The vertical distance between blades divided by chord length.
INLET FLOW ANGLE	The clockwise angle between the vertical and the flow at $x = -\infty$.
K	The curvature of the airfoil before the boundary layer correction.
LOSS COEFFICIENT	Loss coefficient computed from the boundary layer correction.
M	For the isolated airfoil, the Mach number at infinity.
M_1	Inlet Mach number. For the cascade, the Mach number at $x = -\infty$.
M_2	Exit Mach number. For the cascade, the Mach number at $x = +\infty$.
PROFILE DRAG COEFFICIENT	Drag coefficient computed from boundary layer correction.
SEP	Nash-Macdonald separation parameter. The boundary layer is predicted to separate for SEP > .004 .
T/C	Thickness to chord ratio.
THETA	Momentum thickness of the boundary layer.
TRANSITION	Point on the upper or lower surface at which the turbulent boundary layer correction begins.

X	x coordinate of the airfoil before the boundary layer correction.
XS	x coordinate of the airfoil after the boundary layer correction.
Y	y coordinate of the airfoil before the boundary layer correction.
YS	y coordinate of the airfoil after the boundary layer correction.

VIII. BIBLIOGRAPHY

1. BAILEY, F.R., and BALLHAUS, W.F. - Relaxation methods for transonic flow about wing-cylinder combinations and lifting swept wings, Lecture Notes in Physics, Vol 19, pp. 2-9, Springer-Verlag, New York 1972.

2. BAUER, F., GARABEDIAN, P., and KORN, D. - A Theory of Supercritical Wing Sections, with Computer Programs and Examples, Lecture Notes in Economics and Mathematical Systems, Vol. 66, Springer-Verlag, New York, 1972.

3. BAUER, F., GARABEDIAN, P., JAMESON, A., and KORN, D. - Supercritical Wing Sections II, a Handbook, Lecture Notes in Economics and Mathematical Systems, Vol, 108, Springer-Verlag, New York, 1975.

4. BAUER, F., GARABEDIAN, P., and KORN, D. - Supercritical Wing Sections III, Lecture Notes in Economics and Mathematical Systems, Vol. 150, Springer-Verlag, New York, 1977.

5. CARLSON, L.A. - Inverse transonic flow calculations using experimental pressure distributions, A.I.A.A. Journal, Vol. 12, pp. 571-572, 1974.

6. CARLSON, L.A. - Transonic airfoil analysis and design using Cartesian coordinates, Journal of Aircraft, Vol. 13, pp. 349-356, 1976.

7. COURANT, R., and FRIEDRICHS, K.O. - Supersonic Flow and Shock Waves, Interscience-Wiley, New York, 1948.

8. FORSYTHE, G.E., and WASOW, W.R. - Finite Difference Methods for Partial Differential Equations, Wiley, New York, 1960.

9. GARABEDIAN, P. - Partial Differential Equations, Interscience-Wiley, New York, 1964.

10. GARABEDIAN, P., and McFADDEN, G. - Computational fluid dynamics of airfoils and wings, Transonic, Shock and Multidimensional Flows: Advances in Scientific Computing, Ed. Meyer, R.F. Academic Press, New York, 1982.

11. LANGLEY, M.J. - Numerical methods for two-dimensional and axisymmetric transonic flows, ARA Memo 143, 1973.

12. McCARTIN, B. - Theory, computation and application of exponential splines, R&D Report DOE/ER/3077-171, Courant Inst. Math. Sci., New York University, 1981.

13. McFADDEN, G. - An artificial viscosity method for the design of supercritical airfoils, R&D Report C00-3077-158, Courant Inst. Math. Sci., New York Univ., 1979.

14. MORAWETZ, C.S. - On the nonexistence of continuous transonic flows past profiles I, Comm. Pure Appl. Math., Vol. 9, pp. 45-48, 1956.

15. MURMAN, E.M., and COLE, J.D. - Calculation of plane steady transonic flows, A.I.A.A. Journal, Vol. 9, pp. 114-121, 1972.

16. NASH, J.F., and MacDONALD, A.G.J. - The calculation of momentum thickness in a turbulent boundary layer at Mach numbers up to unity, Aeronautical Research Council C.P. No. 963, London, 1967.

17. NIEUWLAND, G.Y., and SPEE, B.M. - Transonic airfoils: recent developments in theory, experiment, and design, Annual Review of Fluid Mechanics, Vol. 5, pp. 119-150, 1973, Annual Reviews, Inc., Palo Alto, Calif., 1973.

18. NIEUWLAND, G.Y. - Transonic potential flow around a family of quasi-elliptical aerofoil sections, N.L.R. Report Tr.T172, 1967.

19. PEARCEY, H.H. - The aerodynamic design of section shapes for swept wings, Advances Aero. Sci., Vol. 3, p. 277-322, 1962.

20. SANZ, J. - Design of supercritical cascades with high solidity, A.I.A.A. Paper 82-10954, 1982.

21. SANZ, J. - A well posed boundary value problem in transonic gas dynamics, Comm. Pure Appl. Math., Vol. 31, pp. 671-679, 1978.

22. SHIFFMAN, M. - On the existence of subsonic flows of a compressible fluid, J. Ratl. Mech. Anal., Vol. 1, pp. 605-652, 1952.

23. SOBIECZKY, H., FUNG, K.Y., and SEEBASS, A.R. - A new method for designing shock-free transonic configurations, A.I.A.A. Paper 78-1114. 1978.

24. SPEE, B.M., and UIJLENHOET, R. - Experimental verification of shock-free transonic flow around quasi-elliptic aerofoil sections, N.L.R. Report MP 68003 U, 1969.

25. SWENSON, E. - Geometry of the complex characteristics in transonic flow, Comm. Pure Appl. Math., Vol 21, pp. 175-185, 1968.

26. TRANEN, T.L. - A rapid computer aided transonic airfoil design method, A.I.A.A. Paper 74-501, 1974.

27. WHITCOMB, R.T., and CLARK, L.R. - An airfoil shape for efficient flight at supercritical Mach numbers, NASA TM X-1109, 1965.

VII. LISTING OF THE CODE

C
C LISTING OF CODE COMPRES WITH COMMENT CARDS
C
C
C TABLE OF SUBROUTINES
C
C

```
C
C     47. FFORMS    FAST FOURIER TRANSFROM ROUTINE                  179
C
C     48. GRAPH     CONTROL ROUTINE FOR GRAPHICS                    182
C
C     83. PLOTQS    PLOT SPEED DISTRIBUTION                         183
C
C     50. LOCUS     PLOT COMPLEX HODOGRAPH PLANE                    186
C
C     51. CONTOR    PLOT CURVES Q=CONST                             188
C
C     52. PLPATH    PLOT PATHS OF INTEGRATION                       191
C
C     53. BDYPLT    PLOT ISOLATED AIRFOIL                           192
C
C     54. CASCAD    PLOT CASCADE OF AIRFOILS                        193
C
C     55. PLTCHR    PLOT SUPERSONIC CHARACTERISITICS                194
C
C     56. XYAXES    PLOT COORDINATES AXES                           195
C
C
C                          LIBRARY SUBROUTINES
C
C
C         FRAME     BEGIN NEW PAGE OF PLOT
C
C         PLOT      DRAW LINE BETWEEN TWO POINTS
C
C         PLOTS     INITIALIZE GRAPH PAPER PLOTS
C
C         PLOTSBL   INITIALIZE BLANK PAPER PLOTS
C
C         SYMBOL    DRAW SYMBOL ON PLOT
C
C
C                          INDEX OF TAPES
C
C         TAPE1      INTERNAL TAPE
C
C         TAPE3      INPUT TAPE FOR SPEED DISTRIBUTION
C
C         TAPE4      OUTPUT TAPE
C
C         TAPE7      INPUT TAPE FOR NAMELIST PARAMETERS
C
C         M1         INTERNAL TAPE
C
C         TAPE1 =M1 UNLESS MRP NEGATIVE.  IN THIS CASE, RICHARDSON
C         EXTRATPOLATION IS DONE AFTER THE LAST ITERATION, AND THEN
C         TAPE1 =TAPE3.
C
```

```
C         TAPE7 IS LATER USED TO STORE DATA IN SUPERSONIC REGION
C         NEEDED TO PLOT CHARACTERISTICS IN SUBROUTINE PLTCHR
C
C
      PROGRAM COMPRES(OUTPUT,TAPE4=OUTPUT,TAPE7,TAPE1,TAPE3)
C     THIS CODE COMPUTES SMOOTH SOLUTIONS TO THE TRANSONIC FLOW
C     EQUATIONS USING THE METHOD OF COMPLEX CHARACTERISTICS
C     THIS ROUTINE IS THE MAIN CONTROL PROGRAM
C
      COMMON /1/ NP,KBM,MODE,NF,NN,MM,NI,NX,IS,KC,LBM,JJ,KK,NB,NC,MRP,NJ
     1,KF,NBPS,NFC,NRN,NPTS,NT,IPLT,LOFF,KP,NOSE,NCAS
      COMMON /2/ PI,TP,GRID,TOL
      COMPLEX XIA,XIB,XIC,DZ
      COMMON /3/ R,R1,R2,RATC,PHMN,GAM,WTAIL,XIA,XIB,XIC,UA,UB,VA,VB,UC,
     1VC,TRANU,TRANL,CONE,CTWO,CTHR,EP,RN,SO,THNOS,DZ
      REAL MACHA,MACHB
      COMMON /7/ MACHA,MACHB,ANGLA,ANGLB
      REAL MACH
      COMMON /A/ GAMMA,MACH,RHOINF,EMU,QSTRSQ
      COMMON /G/ N1,N3,N4,N7,M1
      COMPLEX F1,F2,F3,S1,S2,S3,PHI,PSI,U,V,LAMDAP,LAMDAM,TAU,XI,ETA
      COMMON PHI(585,1),SKA(390),PSI(585,1),SKB(390),XI(585),ETA(585),U(
     1585),V(585),F1(585),S1(585),F2(585),S2(585),F3(585),S3(585),LAMDAP
     2(585),LAMDAM(585),TAU(585)
C     THE ABOVE COMMON IS CONSTRUCTED SO THAT ARRAYS HAVE DIMENSION
C     195 IN THE ELLIPSE WHERE 16 FUNCTIONS PHI AND PSI ARE COMPUTED
C     IN THE SUPERSONIC REGIONS ARRAYS HAVE DIMENSION 585 AND ONE PAIR
C     OF FUNCTIONS PHI AND PSI IS COMPUTED
      NAMELIST /ONE/ NRN,IPLT,NFC,XIA,XIB,EP,NP,MRP,NI,MACH,RN,GRID,GAMM
     1A,XIC,RATC,TRANU,TRANL,CONE,CTWO,CTHR,NPTS,NCAS,NOSE
      DATA N1/1/,N3/3/,N4/4/,N7/7/,LBM/195/,KBM/16/,NPTS/201/,NP/8/,MRP/
     11/,NI/1/,MACH/.75/,RN/0./,GRID/.08/,GAMMA/1.4/,GAM/0./,EP/.8/,XIA/
     2(.0,.0)/,XIB/(.0,.0)/,XIC/(0.,0.)/,NFC/32/,NOSE/0/,NCAS/2/,TRANU/.
     305/,TRANL/.10/,CONE/.5/,CTWO/1./,CTHR/1./,NRN/1/,RATC/0./
      DATA NPTMAX/600/,NFCMAX/64/
C     READ IN TAPE 7 PARAMETERS
      READ (N7,ONE)
C     CHECK THAT ARRAYS ARE PROPERLY DIMENSIONED
      IF (NPTS.LE.NPTMAX) GO TO 10
      WRITE (N4,240) NPTMAX
      CALL EXIT
   10 IF (NFC.LE.NFCMAX) GO TO 20
      WRITE (N4,250) NFCMAX*2
      CALL EXIT
C     INITIALIZE PARAMETERS REQUIRED IN CODE
   20 TOL=1.E-7
      PI=ACOS(-1.)
```

```
      TP=2.*PI
      NBPS=NFC+1
      NF=2*NFC
      DARG=TP/FLOAT(NF)
      THNOS=PI-FLOAT(NOSE)*DARG
      NT=NF+1
      R=SQRT((1.+EP)/(1.-EP))
      R1=.5*(R+1./R)
      R2=.5*(R-1./R)
      NNMAX=0
      IF (CABS(XIC).GT.TOL) GO TO 30
      XIC=XIA-4.*GRID*(R1-XIA)/CABS(R1-XIA)
      IF (CABS(XIA-XIB).GT.TOL) XIC=XIB+GRID*(XIB-XIA)/CABS(XIB-XIA)
   30 M1=N1
      REWIND N1
      REWIND N3
      REWIND N7
      NM1=8H DESIGN
      WRITE (N4,230) NM1,NRN
      WRITE (N4,180)
C     INITIALIZE PROBLEM
      CALL READQS
      CALL PAGE2
      MRPOLD=MRP
      MRP=1
      CALL PHIINC
      WRITE (N4,190)
      CALL SECOND (TIME)
C     ITERATE TO FIND SOLUTION IN THE ELLIPSE
      DO 100 ITER=1,NI
      IF (ITER.EQ.NI) MRP=IABS(MRPOLD)
C     FIND MAP FROM ELLIPSE TO HODOGRAPH PLANE
      CALL CYCLQ (ITER)
      CALL INIT (XI,ETA,U,V)
      NCP=NC+1
      CALL CONST (3)
   40 DO 80 MODE=1,NP
      KC=0
      CALL GTPATH (XI,NC+1,MM)
      JJ=KK
      NN=MM
      DO 50 J=NCP,NN
   50 ETA(J)=CONJG(XI(J))
      IF (KC.LE.0) GO TO 60
C     CONSTRUCT TRANSONIC PATH IN ETA PLANE
      NNMAX=MAX0(NNMAX,MM)
      CALL GTPATH (XI,NC+1,MM)
      KKL=KK
      KK=JJ
      JJ=KKL
   60 NNMAX=MAX0(NNMAX,MM)
      IS=0
```

```
C     COMPUTE SINGULAR TERMS AND NF-1 REGULAR TERMS
      DO 70 J=2,NF,KBM
      NX=MINO(KBM,NF+1-J)
      IS=IS+1
      NJ=J-2
   70 CALL SOLVE (NN)
   80 CONTINUE
      IF ((ITER.NE.NI).OR.(MRPOLD.NE.-MRP)) GO TO 90
C     RICHARDSON EXTRAPOLATION
      MRP=MRP+MRP
      N1=N3
      REWIND N1
      CALL INIT (XI,ETA,U,V)
      NCP=NC+1
      GO TO 40
   90 CALL AUTO2 (RES,DZ)
  100 WRITE (N4,200) ITER,MACHB,MACHA,RES,REAL(DZ),AIMAG(DZ)
      CALL SECOND (T1)
      TIME=T1-TIME
      MSG=10H SUBSONIC
      WRITE (N4,210) MSG,NNMAX
      WRITE (N4,220) MSG,TIME
C     FIND SOLUTION IN REAL SUPERSONIC ZONE
      CALL SECOND (TIME)
      MRP=IABS(MRPOLD)
      N1=M1
      REWIND N1
      READ (N1)
      READ (N1)
      READ (N1)
      READ (N1)
      READ (N1)
      NNMAX=0
      IS=1
      NX=1
  110 CALL INIT (XI,ETA,U,V)
      NCP=NC+1
      J=0
  120 KC=0
      J=J+1
      IF (J.GT.6) GO TO 160
      MODE=-J
      CALL SUPATH (J,XI,NC+1,MM)
      IF (MM.EQ.0) GO TO 140
      NNMAX=MAXO(NNMAX,MM)
      CALL SUPATH (J,ETA,NC+1,NN)
      DO 130 M=NCP,NN
  130 ETA(M)=CONJG(ETA(M))
      NNMAX=MAXO(NNMAX,NN)
      CALL SOLVE (NN)
      GO TO 120
  140 IF ((MRP.NE.-MRPOLD).OR.(J.EQ.1)) GO TO 150
```

```
C     RICHARDSON EXTRAPOLATION
      MRP=MRP+MRP
      N1=N3
      REWIND N1
      GO TO 110
  150 CONTINUE
      CALL SECOND (T1)
      TIME=T1-TIME
      MSG=10HSUPERSONIC
      WRITE (N4,210) MSG,NNMAX
      WRITE (N4,220) MSG,TIME
      WRITE (N1) WTAIL+TP+TOL,WTAIL+TP+TOL
      MRP=IABS(MRPOLD)
C     FIND WRITTEN AND GRAPHICAL OUTPUT
      CALL OUTPT (TC,GAP,GC)
      CALL GRAPH (TC,GAP,GC)
      CALL EXIT
  160 WRITE (N4,170)
      CALL EXIT
  170 FORMAT (///13X40H TOO MANY SUPERSONIC ZONES. CHANGE SPEED,13H DIST
     1RIBUTION)
  180 FORMAT (1H0/38X,11HTAPE3=INPUT/)
  190 FORMAT (////13X5HCYCLE,4X6HMINLET,5X5HMEXIT,5X8HRESIDUAL,8X2HDX,8X
     12HDY/)
  200 FORMAT (1H0,11X15,1X,2F10.3,E13.3,1X,2F10.5)
  210 FORMAT (///13X8HLONGEST ,A10,10H PATH HAS ,I3,7H POINTS)
  220 FORMAT (/13X,A10,11H CP TIME IS,F7.1,8H SECONDS)
  230 FORMAT (1H1/8X,19HBEGIN EXECUTION OF A8,4H RUN,I4)
  240 FORMAT (56H0****SPLINE OF INPUT SPEED DISTRIBUTION CAN HAVE AT MOS
     1T,I4,12H POINTS.****//41H****CHANGE NPTS OR DIMENSION OF ARRAYS IN
     210HCOMMON****)
  250 FORMAT (50H0****MAXIMUM DIMENSION OF ARRAYS AROUND ELLIPSE IS,I4,4
     1H****//48H****CHANGE NFC OR DIMENSION OF COMMON BLOCKS****)
      END


      SUBROUTINE PAGE2
C     PRINTS TAPE 7 PARAMETERS ON SECOND PAGE OF OUTPUT
      COMMON /1/ NP,KBM,MODE,NF,NN,MM,NI,NX,IS,KC,LBM,JJ,KK,NB,NC,MRP,NJ
     1,KF,NBPS,NFC,NRN,NPTS,NT,IPLT,LOFF,KP,NOSE,NCAS
      COMMON /2/ PI,TP,GRID,TOL
      COMPLEX XIA,XIB,XIC,DZ
      COMMON /3/ R,R1,R2,RATC,PHMN,GAM,WTAIL,XIA,XIB,XIC,UA,UB,VA,VB,UC,
     1VC,TRANU,TRANL,CONE,CTWO,CTHR,EP,RN,SO,THNOS,DZ
      REAL MACH
      COMMON /A/ GAMMA,MACH,RHOINF,EMU,QSTRSQ
```

```
      COMMON /G/ N1,N3,N4,N7,M1
      CALL DATE (IDATE)
      NRUN=IABS(NRN)
      NM1=7HCASCADE
      NM2=8H DESIGN
      IF (CABS(XIA-XIB).LT.TOL) NM1=7HAIRFOIL
      WRITE (N4,10) NM1,NM2,NRUN,IDATE
      NM1=8H   RUN =
      NM2=8H  IPLT =
      WRITE (N4,20) NM1,NRN,NM2,IPLT
      NM1=8H    NI =
      NM2=8H    NP =
      WRITE (N4,20) NM1,NI,NM2,NP
      NM1=8H   NFC =
      NM2=8H  NCAS =
      WRITE (N4,20) NM1,NFC,NM2,NCAS
      NM1=8H  NPTS =
      NM2=8H  NOSE =
      WRITE (N4,20) NM1,NPTS,NM2,NOSE
      NM1=8H  MACH =
      NM2=8H    RN =
      WRITE (N4,70) NM1,MACH,NM2,RN
      NM1=8H    EP =
      NM2=8H GAMMA =
      WRITE (N4,40) NM1,EP,NM2,GAMMA
      NM1=8H  RATC =
      NM2=8H  CONE =
      WRITE (N4,40) NM1,RATC,NM2,CONE
      NM1=8H  CTWO =
      NM2=8H  CTHR =
      WRITE (N4,40) NM1,CTWO,NM2,CTHR
      NM1=8H TRANU =
      NM2=8H TRANL =
      WRITE (N4,40) NM1,TRANU,NM2,TRANL
      NM1=8H  GRID =
      NM2=8H   MRP =
      WRITE (N4,30) NM1,GRID,NM2,MRP
      NM1=8H   XIA =
      NM2=8H   XIB =
      WRITE (N4,50) NM1,XIA,NM2,XIB
      NM1=8H   XIC =
      WRITE (N4,80) NM1,XIC
      WRITE (N4,60)
      RETURN
   10 FORMAT (1H1/15X,10H SUBSONIC ,A8,A9,3HRUNI5,17XA10///37X,12HTAPE7=
     1INPUT /)
   20 FORMAT (/17X,A8,2X,I4,11X1H;12XA8,2X,I4)
   30 FORMAT (/17X,A8,F6.2,11X1H;12XA8,2X,I4)
   40 FORMAT (/17X,A8,F6.2,11X1H;12XA8,F6.2)
   50 FORMAT (/9XA8,F6.3,2H ,F6.3,11X1H;4XA8,F6.3,2H ,F6.3)
   60 FORMAT (1HO)
   70 FORMAT (/17X,A8,F7.3,10X1H;12XA8,1X,E8.2)
```

```
   80 FORMAT (/27X,A8,F6.3,2H ,F6.3)
      END


      SUBROUTINE READQS
C     SCALES AND STORES THE INPUT SPEED DISTRIBUTION
      COMMON /1/ NP,KBM,MODE,NF,NN,MM,NI,NX,IS,KC,LBM,JJ,KK,NB,NC,MRP,NJ
     1,KF,NBPS,NFC,NRN,NPTS,NT,IPLT,LOFF,KP,NOSE,NCAS
      COMMON /2/ PI,TP,GRID,TOL
      COMPLEX XIA,XIB,XIC,DZ
      COMMON /3/ R,R1,R2,RATC,PHMN,GAM,WTAIL,XIA,XIB,XIC,UA,UB,VA,VB,UC,
     1VC,TRANU,TRANL,CONE,CTWO,CTHR,EP,RN,SO,THNOS,DZ
      COMMON /G/ N1,N3,N4,N7,M1
      COMMON SX(300),QI(300),SI(300),E(300),G(300),A(300),B(300),C(300),
     1D(300),ES(600),Q(600),PHI(600),FP(600),FPP(600),FPPP(600)
      DATA KK/0/,NINMAX/300/
      READ (N3,130) NIN
      IF (NIN.LT.0) KK=1
C     NIN.LT.0 IF EXPONENTIAL SPLINE IS USED
      NIN=IABS(NIN)
      IF (NIN.GT.NINMAX) GO TO 110
C     INPUT SPEED DISTRIBUTION IS FIRST PAGE OF OUTPUT
      IF (KK.EQ.0) GO TO 30
      DO 10 J=1,NIN
   10 READ (N3,140) SX(J),QI(J),E(J),G(J)
      ARCL=SX(NIN)-SX(1)
      FAC=2./ARCL
      CONST=-FAC*SX(1)
      WRITE (N4,160)
      SI(1)=0.
      DO 20 J=2,NIN
      SI(J)=FAC*SX(J)+CONST
   20 WRITE (N4,170) J-1,SX(J-1),SI(J-1),QI(J-1),E(J-1),G(J-1)
      WRITE (N4,170) NIN,SX(NIN),SI(NIN),QI(NIN)
      GO TO 60
   30 DO 40 J=1,NIN
   40 READ (N3,140) SX(J),QI(J)
      ARCL=SX(NIN)-SX(1)
      FAC=2./ARCL
      CONST=-FAC*SX(1)
      WRITE (N4,150)
      DO 50 J=1,NIN
      SI(J)=FAC*SX(J)+CONST
   50 WRITE (N4,180) J,SX(J),SI(J),QI(J)
   60 SI(NIN)=2.
      WRITE (N1) NRN,NIN,(SI(J),QI(J),J=1,NIN)
```

```
      DS=(SI(NIN)-SI(1))/FLOAT(NPTS-1)
C     FIND Q(S) AT EVENLY SPACED POINTS
      ES(1)=SI(1)
      DO 70 J=2,NPTS
   70 ES(J)=ES(J-1)+DS
      ES(NPTS)=SI(NIN)
      IF (KK.EQ.0) GO TO 80
      CALL SPTEN (NIN,SI,QI,E,G,A,B,C,D,3,0.,3,0.)
      CALL INTEN (NPTS,ES,Q,NIN,SI,A,B,C,D,E,G)
      GO TO 90
   80 CALL SPLIF (NIN,SI,QI,FP,FPP,FPPP,3,0.,3,0.)
      CALL INTPL (NPTS,ES,Q,SI,QI,FP,FPP,FPPP)
C     INTEGRATE Q(S) TO GET PHI(S)
   90 CALL SPLIF (NPTS,ES,Q,FP,FPP,PHI,3,0.,3,0.)
      GAM=PHI(NPTS)
C     SPLINE FIT Q AS A FUNCTION OF S
      CALL SPLIF (NPTS,ES,Q,FP,FPP,FPPP,3,0.,3,0.)
C     FIND S WHERE Q VANISHES
      CALL INTPLI (1,SO,0.,NPTS,ES,Q,FP,FPP,FPPP)
C     SPLINE FIT PHI AS A FUNCTION OF S
      CALL SPLIF (NPTS,ES,PHI,FP,FPP,FPPP,3,0.,3,0.)
C     FIND MINIMUM VALUE OF PHI
      CALL INTPL (1,SO,PHMN,ES,PHI,FP,FPP,FPPP)
C     FIND MONOTONIC PHI
      DO 100 I=1,NPTS
      VAL=AMAX1(0.,PHI(I)-PHMN)
  100 PHI(I)=SIGN(SQRT(VAL),ES(I)-SO)
      WRITE (N1) NPTS,SO,PHMN,GAM,(ES(J),Q(J),PHI(J),J=1,NPTS)
      RETURN
  110 WRITE (N4,120) NINMAX
      CALL EXIT
  120 FORMAT (15H0****MORE THAN ,I4,30H INPUT CARDS NOT PERMITTED****/32
     1H0***PROGRAM STOPPED IN READQS***)
  130 FORMAT (I5)
  140 FORMAT (4E20.8)
  150 FORMAT (1H0/21X,4HCARD,5X,7HS-INPUT,6X,6HS-USED,5X,7HQ-INPUT/)
  160 FORMAT (1H0/11X,4HCARD,5X,7HS-INPUT,5X,6HS-USED,5X,7HQ-INPUT,9X,1H
     1E,9X,1HG)
  170 FORMAT (6X,I9,5F12.6)
  180 FORMAT (16X,I9,5F12.6)
      END


      SUBROUTINE PHIINC
C     CALCULATES POTENTIAL PHII IN ELLIPSE FOR INCOMPRESSIBLE FLOW
      COMMON /1/ NP,KBM,MODE,NF,NN,MM,NI,NX,IS,KC,LBM,JJ,KK,NB,NC,MRP,NJ
```

```
     1,KF,NBPS,NFC,NRN,NPTS,NT,IPLT,LOFF,KP,NOSE,NCAS
      COMMON /2/ PI,TP,GRID,TOL
      COMPLEX XIA,XIB,XIC,DZ
      COMMON /3/ R,R1,R2,RATC,PHMN,GAM,WTAIL,XIA,XIB,XIC,UA,UB,VA,VB,UC,
     1VC,TRANU,TRANL,CONE,CTWO,CTHR,EP,RN,SO,THNOS,DZ
      COMMON /G/ N1,N3,N4,N7,M1
      COMPLEX CLOG,ROT,FAC1,FAC2
      COMPLEX BP,XI
      COMMON BP(65),XI(130),F1(130),F2(130),F3(130),BD1(130),BD2(130),BD
     13(130),BD4(130),FINT(130),ATRAN(130),TT(130),FP(130),FPP(130),FPPP
     2(130),GP(130),GPP(130),GPPP(130),HP(130),HPP(130),HPPP(130),ARG(13
     30),PHII(130)
      DARG=TP/FLOAT(NF)
      KN=NT-NOSE
      CF=GAM/TP
      DO 10 J=1,NT
      ARG(J)=FLOAT(J-1)*DARG-PI
   10 XI(J)=CMPLX(R1*COS(ARG(J)),R2*SIN(ARG(J)))
      ARG(NT)=PI
C     FIND SINGULAR TERMS
C     LOGARITHM HAS BRANCH CUT ALONG NEGATIVE REAL AXIS
      ROT=CMPLX(0.,TOL)
      ROT=CEXP(ROT)
      FAC1=-CONJG(XI(1)-XIA)*ROT/CABS(CONJG(XI(1)-XIA))
      IF (CABS(XIB-XIA).LT.TOL) GO TO 30
      FAC2=-CONJG(XI(1)-XIB)*ROT/CABS(CONJG(XI(1)-XIB))
      DO 20 J=1,NF
      ROT=CLOG((XI(J)-XIB)*FAC2)
      BD4(J)=AIMAG(ROT)
      BD3(J)=REAL(ROT)
      ROT=CLOG((XI(J)-XIA)*FAC1)
      BD2(J)=AIMAG(ROT)-BD4(J)
   20 BD1(J)=REAL(ROT)-BD3(J)
      GO TO 50
   30 DO 40 J=1,NF
      ROT=1./(XI(J)-XIA)
      BD1(J)=REAL(ROT)
      BD2(J)=AIMAG(ROT)
      ROT=CLOG((XI(J)-XIA)*FAC1)
      BD3(J)=REAL(ROT)
   40 BD4(J)=AIMAG(ROT)
C     FIND REGULAR TERMS
   50 CALL RING (NF,BD1,F1,BP,R)
      CALL RING (NF,BD2,F2,BP,R)
      CALL RING (NF,BD3,F3,BP,R)
      DO 60 J=1,NF
      F1(J)=-BD2(J)+F1(J)
      F2(J)=BD1(J)+F2(J)
   60 F3(J)=BD4(J)-F3(J)
      F1(NT)=F1(1)
      F2(NT)=F2(1)
      F3(NT)=F3(1)+TP
```

```
C     FIND REGULAR COEFFICIENTS OF FINT
      CALL PSPLIF (NT,ARG,F1,FP,FPP,FPPP,FINT)
      CALL PSPLIF (NT,ARG,F2,GP,GPP,GPPP,FINT)
      CALL PSPLIF (NT,ARG,F3,BD1,BD2,BD3,FINT)
      C1=FP(KN)
      C2=GP(KN)
      C3=BD1(KN)
      WTAIL=THNOS-PI
C     ITERATE AT MOST 20 TIMES TO FIND TAIL
      WELL=-PHMN
      DO 90 K=1,20
      CALL INTPL (1,WTAIL,CC1,ARG,F1,FP,FPP,FPPP)
      CALL INTPL (1,WTAIL,CC2,ARG,F2,GP,GPP,GPPP)
      CALL INTPL (1,WTAIL,CC3,ARG,F3,BD1,BD2,BD3)
      CC1=CC1-F1(KN)
      CC2=CC2-F2(KN)
      CC3=CC3-F3(KN)
      DET=1./(C2*CC1-CC2*C1)
      AF=DET*(CF*C3*CC2-CF*C2*CC3+C2*WELL)
      BF=DET*(-CF*C3*CC1+CF*C1*CC3-C1*WELL)
      DO 70 J=1,NT
      FINT(J)=0.
   70 TT(J)=AF*FP(J)+BF*GP(J)+CF*BD1(J)
      CALL SPLIF (NT,ARG,TT,HP,HPP,HPPP,3,0.,3,0.)
      WOLD=WTAIL
      NS=-NF
      CALL INTPLI (NS,ATRAN,FINT,NT,ARG,TT,HP,HPP,HPPP)
      WTAIL=ATRAN(1)
      IF (NS.LE.1) GO TO 90
      DO 80 LL=2,NS
   80 IF (ABS(ATRAN(LL)-WOLD).LT.ABS(WTAIL-WOLD)) WTAIL=ATRAN(LL)
   90 IF (ABS(WTAIL-WOLD).LT.TOL) GO TO 100
      WRITE (N4,120)
  100 CONST=AF*F1(KN)+BF*F2(KN)+CF*F3(KN)-GAM
      DO 110 J=1,NT
  110 PHII(J)=AF*F1(J)+BF*F2(J)+CF*F3(J)-CONST
      WRITE (N1) NT,(PHII(J),J=1,NT)
      RETURN
  120 FORMAT (37H ITERATION FOR WTAIL DID NOT CONVERGE)
      END


      SUBROUTINE CYCLQ (ITER)
C     FINDS MAP FUNCTION FROM ELLIPSE TO HODOGRAPH PLANE
      COMMON /1/ NP,KBM,MODE,NF,NN,MM,NI,NX,IS,KC,LBM,JJ,KK,NB,NC,MRP,NJ
     1,KF,NBPS,NFC,NRN,NPTS,NT,IPLT,LOFF,KP,NOSE,NCAS
```

```
      COMMON /2/ PI,TP,GRID,TOL
      COMPLEX XIA,XIB,XIC,DZ
      COMMON /3/ R,R1,R2,RATC,PHMN,GAM,WTAIL,XIA,XIB,XIC,UA,UB,VA,VB,UC,
     1VC,TRANU,TRANL,CONE,CTWO,CTHR,EP,RN,SO,THNOS,DZ
      COMPLEX ROOT,ROOT1
      COMMON /5/ ROOT,ROOT1
      COMPLEX BP,ETJ
      COMMON /8/ BP(65),ETJ(65)
      REAL MACH
      COMMON /A/ GAMMA,MACH,RHOINF,EMU,QSTRSQ
      COMMON /G/ N1,N3,N4,N7,M1
      COMPLEX H,HP,ROT
      COMPLEX XI
      COMMON XI(130),AX(130),SS(130),QI(130),QLG(130),PHII(130),FP(600),
     1FPP(600),FPPP(600),QP(600),QPP(600),QPPP(600),ES(600),Q(600),PHI(6
     200)
      DIMENSION QOLD(130)
      ROOT1=(1.,0.)
      IF (ITER.GT.1) GO TO 10
      DT=TP/FLOAT(NF)
      KN=NT-NOSE
      KNP=KN+1
      KNPP=KN+2
      IF (KNP.GT.NF) KNP=KNP-NF
      IF (KNPP.GT.NF) KNPP=KNPP-NF
      KNM=KN-1
      KNMM=KN-2
      IF (KN.GT.NF) KN=KN-NF
      GAMP=(GAMMA+1.)/2.
      GAMM=(GAMMA-1.)/2.
      EMUSQ=GAMM/GAMP
      GAM2=GAM/2.
      QSTRSQ=EMUSQ+1./(GAMP*MACH*MACH)
   10 REWIND N1
      READ (N1)
      READ (N1) NPTS,SO,PHMN,GAM,(ES(J),Q(J),PHI(J),J=1,NPTS)
      READ (N1) NT,(PHII(J),J=1,NT)
C     SPLINE FIT Q AS A FUNCTION OF S
      CALL SPLIF (NPTS,ES,Q,QP,QPP,QPPP,3,0.,3,0.)
C     SPLINE FIT Q AS A FUNCTION OF PHI
      CALL SPLIF (NPTS,PHI,Q,FP,FPP,FPPP,3,0.,3,0.)
C     PHII IS THE GUESS FOR PHI AT EVENLY SPACED POINTS IN THE RING
      DO 20 J=1,NT
      SS(J)=FLOAT(J-1)*DT-PI
      XI(J)=CMPLX(R1*COS(SS(J)),R2*SIN(SS(J)))
      FAC=SIGN(1.,WTAIL-SS(J))
      VAL=AMAX1(0.,PHII(J)-(1.-FAC)*GAM2)
      IF (SS(J).GT.THNOS) FAC=1.
      AX(J)=FAC*SQRT(VAL)
C     COMPUTE SPEED QI AT POINTS ON THE ELLIPSE
      CALL INTPL (1,AX(J),QI(J),PHI,Q,FP,FPP,FPPP)
   20 CONTINUE
```

```
      IF (RATC.NE.0.) GO TO 30
C     FIND DQ/DW AT THE STAGNATION POINT
      ROT=XI(KNP)/XI(KNM)
      DW=ATAN2(AIMAG(ROT),REAL(ROT))
      DQDW=EP*(QI(KNP)-QI(KNM))/DW
      CALL GETHSQ ((0.,0.),H,HP)
      RATC=1.-3./(ABS(DQDW)*SQRT(REAL(HP)))
C     GET COMPRESSIBLE ANALOG FOR LOG(Q)
   30 DO 40 J=1,NF
      IF (J.EQ.KN) GO TO 40
      QHOLD=QI(J)
      IF (ITER.EQ.1) GO TO 35
      RFAC=0.
      IF ((J.EQ.KNP).OR.(J.EQ.KNM)) RFAC=-.7
      AX(J)=ABS(QI(J)-QOLD(J))
      QI(J)=ABS(QI(J))+RFAC*AX(J)
   35 QOLD(J)=QHOLD
      CALL GETHSQ (CMPLX(QI(J)*QI(J),0.),H,HP)
      ROT=(XI(J)-XI(KN))/(RATC*XI(J)-XI(KN))
      RP=REAL(ROT)
      AIMP=AIMAG(ROT)
      FAC=RP*RP+AIMP*AIMP
      HIMG=ABS(AIMAG(H))**CTWO
      HAB=CABS(H)+HIMG*CONE
      QLG(J)=.5*ALOG(HAB/FAC)
   40 CONTINUE
      QLG(KN)=QLG(KNP)+QLG(KNM)-.5*(QLG(KNPP)+QLG(KNMM))
C     COMPUTE THE COEFFICIENTS FOR THE MAPPING FUNCTION
      CALL RING (NF,QLG,QP,BP,R)
      BP(1)=CMPLX(REAL(BP(1)),0.)
C     BP IS FILTERED IN SUBROUTINE RING
      RETURN
      END


      SUBROUTINE INIT (XI,ETA,U,V)
C     CONSTRUCTS THE PATH FROM XIA TO XIB TO XIC AND
C     FINDS VELOCITY AT XIA AND XIB FROM MAP FUNCTION
      COMPLEX S,H,SFROMX,XI(1),ETA(1),U(1),V(1)
      COMMON /1/ NP,KBM,MODE,NF,NN,MM,NI,NX,IS,KC,LBM,JJ,KK,NB,NC,MRP,NJ
     1,KF,NBPS,NFC,NRN,NPTS,NT,IPLT,LOFF,KP,NOSE,NCAS
      COMMON /2/ PI,TP,GRID,TOL
      COMPLEX XIA,XIB,XIC,DZ
      COMMON /3/ R,R1,R2,RATC,PHMN,GAM,WTAIL,XIA,XIB,XIC,UA,UB,VA,VB,UC,
     1VC,TRANU,TRANL,CONE,CTWO,CTHR,EP,RN,SO,THNOS,DZ
      COMPLEX ROOT,ROOT1
```

```
      COMMON /5/ ROOT,ROOT1
      REAL MACHA,MACHB
      COMMON /7/ MACHA,MACHB,ANGLA,ANGLB
      REAL MACH
      COMMON /A/ GAMMA,MACH,RHOINF,EMU,QSTRSQ
C     INITIALIZE BRANCH OF COMPLEX SQUARE ROOT
      ROOT=(0.,1.)
      ROOT1=(1.,0.)
      RAD=PI/180.
      GAMM=(GAMMA-1.)/2.
      COSQ=1./(MACH*MACH)+GAMM
C     LAY DOWN PATH FROM XIA TO XIB TO XIC IN XI-PLANE
      XI(1)=XIA
C     COMPUTE THE NUMBER OF POINTS FROM XIA TO XIB
      MM=CABS(XIA-XIB)/GRID+1.-TOL
      MM=MM*MRP
      NB=MM+1
      IF (NB.EQ.1) GO TO 20
      H=(XIB-XIA)/FLOAT(MM)
C     LAY DOWN GRID TO XIB
      DO 10 L=2,NB
   10 XI(L)=XI(L-1)+H
C     CONSTRUCT PATH FROM XIB TO XIC
   20 MM=CABS(XIC-XIB)/GRID+1.-TOL
      MM=MM*MRP*2
      NC=NB+MM
      H=(XIC-XIB)/FLOAT(MM)
      NBP=NB+1
      DO 30 L=NBP,NC
   30 XI(L)=XI(L-1)+H
C     LAY DOWN REFLECTED GRID IN THE ETA-PLANE
      DO 40 L=1,NC
   40 ETA(L)=CONJG(XI(L))
C     FIND MACH NUMBER AND ANGLE AT INLET
      S=SFROMX(XIB)
      U(NB)=1.
      V(NB)=0.
      CALL GETUV (S,CONJG(S),U(NB),V(NB))
      UB=REAL(U(NB))
      VB=REAL(V(NB))
      QSB=UB*UB+VB*VB
      MACHB=SQRT(QSB/(COSQ-GAMM*QSB))
      ANGLB=ATAN2(VB,UB)/RAD
C     FIND MACH NUMBER AND ANGLE AT EXIT
      U(1)=1.
      V(1)=0.
      S=SFROMX(XIA)
      CALL GETUV (S,CONJG(S),U(1),V(1))
      UA=REAL(U(1))
      VA=REAL(V(1))
      QSA=UA*UA+VA*VA
      MACHA=SQRT(QSA/(COSQ-GAMM*QSA))
```

```
      ANGLA=ATAN2(VA,UA)/RAD
C     INITIALIZE AT XIC
      U(NC)=1.
      V(NC)=0.
      S=SFROMX(XIC)
      CALL GETUV (S,CONJG(S),U(NC),V(NC))
      UC=REAL(U(NC))
      VC=REAL(V(NC))
      RETURN
      END


      SUBROUTINE GTPATH (XI,K1,K2)
C     SETS UP SUBSONIC PATHS IN COMPLEX CHARACTERISTIC PLANES
      COMMON /1/ NP,KBM,MODE,NF,NN,MM,NI,NX,IS,KC,LBM,JJ,KK,NB,NC,MRP,NJ
     1,KF,NBPS,NFC,NRN,NPTS,NT,IPLT,LOFF,KP,NOSE,NCAS
      COMMON /2/ PI,TP,GRID,TOL
      COMPLEX XIA,XIB,XIC,DZ
      COMMON /3/ R,R1,R2,RATC,PHMN,GAM,WTAIL,XIA,XIB,XIC,UA,UB,VA,VB,UC,
     1VC,TRANU,TRANL,CONE,CTWO,CTHR,EP,RN,SO,THNOS,DZ
      REAL MACH
      COMMON /A/ GAMMA,MACH,RHOINF,EMU,QSTRSQ
      COMMON /G/ N1,N3,N4,N7,M1
      COMPLEX H,POINT,S,SFROMX,XI(1)
      COMPLEX SIG,XIOFS
      DATA K2MAX/0/
      QCRIT=(GAMMA-1.)/(GAMMA+1.)+2./((GAMMA+1.)*MACH*MACH)
      CALL GETHSQ (CMPLX(QCRIT,0.),H,POINT)
      SCRIT=REAL(H)
      PIX=TP/FLOAT(NP)
      PIX2=PIX/2.
      TH=THNOS-TP-PIX2+FLOAT(MODE-1)*PIX
C     KC NEGATIVE IF WE ARE CONSTRUCTING SECOND TRANSONIC PATH
      IF (KC.GT.0) TH=TH+PIX
      FAC=1.
      IF (KC.GT.0) FAC=-1.
      TH1=TH
      TH2=TH+FAC*PIX
      CALL PATH (XI,R,TH1,TH2,K1,K2)
C     CHECK TO SEE IF WE HAVE CROSSED THE SONIC LINE
      DO 10 J=KK,K2
      S=SFROMX(XI(J))
   10 IF (REAL(S)*REAL(S)+AIMAG(S)*AIMAG(S).GE.SCRIT) GO TO 20
      GO TO 40
C     REFLECT POINT ACROSS THE SONIC LINE
   20 POINT=XIOFS(CONJG(SCRIT/SFROMX(XI(J))),XI(J))
```

```
      POINT=SIG(POINT)
      KC=1
C     ROTATE POINT TO AVOID SONIC SURFACE
      RAD=CABS(POINT)
      THF=ATAN2(AIMAG(POINT),REAL(POINT))
      SCL=RAD*RAD/(R*R)
      RAD=1.+SCL*(RAD-1.)
      THF=THF-.1*FAC/SCL
      IF (ABS(TH).GE.2.) GO TO 30
      FAC2=ABS(THF)/PI
      THF=THF-.2*FAC/FAC2
      RAD=1.+.7*FAC2*(RAD-1.)
   30 IF (ABS(THF-TH).GE.PI) TH=TH+SIGN(TP,THF-TH)
      IF (FAC*(THF-TH).GT.0.) GO TO 40
      CALL PATH (XI,RAD,THF,THF,KF,K2)
      CALL PATH (XI,R,TH1,TH2,K2+1,K2)
   40 K2MAX=MAX0(K2MAX,K2)
      IF (K2.LE.LBM) RETURN
      MSG=10HAUTOMATION
      WRITE (N4,50) MSG
      CALL EXIT
   50 FORMAT (////5X5H**** ,A10,23H PATH IS TOO LONG *****)
      END


      SUBROUTINE SUPATH (JP,XI,K1,K2)
C     DETERMINES SUPERSONIC PATHS OF INTEGRATION
      COMMON /1/ NP,KBM,MODE,NF,NN,MM,NI,NX,IS,KC,LBM,JJ,KK,NB,NC,MRP,NJ
     1,KF,NBPS,NFC,NRN,NPTS,NT,IPLT,LOFF,KP,NOSE,NCAS
      COMMON /2/ PI,TP,GRID,TOL
      COMPLEX XIA,XIB,XIC,DZ
      COMMON /3/ R,R1,R2,RATC,PHMN,GAM,WTAIL,XIA,XIB,XIC,UA,UB,VA,VB,UC,
     1VC,TRANU,TRANL,CONE,CTWO,CTHR,EP,RN,SO,THNOS,DZ
      REAL MACH
      COMMON /A/ GAMMA,MACH,RHOINF,EMU,QSTRSQ
      COMMON /G/ N1,N3,N4,N7,M1
      COMPLEX POINT,H,S,SFROMX,XIOFS,XI
      DIMENSION XI(1)
      IF (KC.EQ.0) GO TO 30
C     GET SECOND SUPERSONIC PATH
      RAD=CABS(CMPLX(R1*COS(THF),R2*SIN(THF)))
      FAC=AMIN1(RAD/R1,(.96*R-1.)/(R-1.))
      FAC=1.+FAC*(R-1.)
      CALL PATH (XI,FAC,THF-.07,THF-.07,K1,K2)
      IF (ABS(THF-THL).LT..30) GO TO 10
C     SUPERSONIC PATH LIES ON ELLIPSE BOUNDARY TO AVOID SONIC SURFACE
```

```
      CALL PATH (XI,1.04*R,THF-.04,THF+.12,K2+1,K2)
      CALL PATH (XI,R,THF+.16,THL-.12,K2+1,K2)
      CALL PATH (XI,1.04*R,THL-.08,THL-.04,K2+1,K2)
      GO TO 20
   10 CALL PATH (XI,1.04*R,THF-.04,THL-.04,K2+1,K2)
   20 CALL PATH (XI,R,THL,THF,K2+1,K2)
      GO TO 170
   30 CONTINUE
      IF (JP.NE.1) GO TO 40
      IM=TP/.04+1.-TOL
      DT=TP/FLOAT(IM)
      QCRIT=(GAMMA-1.)/(GAMMA+1.)+2./((GAMMA+1.)*MACH*MACH)
      CALL GETHSQ (CMPLX(QCRIT,0.),H,POINT)
      SCRIT=REAL(H)
      THFO=WTAIL
      L2=1
      SOLD=TOL
   40 DTHF=-DT
      DTHL=-DT
      THF=THFO
C     CHECK FOR SUPERSONIC POINT
      DO 50 L=L2,IM
      POINT=CMPLX(R1*COS(THF),R2*SIN(THF))
      S=SFROMX(POINT)
      SNEW=REAL(S)*REAL(S)+AIMAG(S)*AIMAG(S)
      IF (SNEW.GT.SCRIT) GO TO 60
      SOLD=SNEW
   50 THF=THF+DT
      K2=0
      RETURN
C     PERFORM NEWTON ITERATION TO FIND SONIC POINT
   60 L2=L+1
      THL=THF
      SLAS=SNEW
      DO 70 ICOUNT=1,20
      DTHF=-DTHF*(SNEW-SCRIT)/(SNEW-SOLD)
      IF (DTHF.LT.TOL) GO TO 90
      THF=THF-DTHF
      S=SFROMX(CMPLX(R1*COS(THF),R2*SIN(THF)))
      SOLD=SNEW
      SNEW=REAL(S)*REAL(S)+AIMAG(S)*AIMAG(S)
   70 IF (ABS(SNEW-SCRIT).GT.ABS(SOLD-SCRIT)) GO TO 80
   80 WRITE (N4,200)
C     SEARCH FOR SUBSONIC POINT
      SOLD=SLAS
   90 DO 100 L=L2,IM
      THL=THL+DT
      POINT=CMPLX(R1*COS(THL),R2*SIN(THL))
      S=SFROMX(POINT)
      SNEW=REAL(S)*REAL(S)+AIMAG(S)*AIMAG(S)
      IF (SNEW.LT.SCRIT) GO TO 110
  100 SOLD=SNEW
```

```
      WRITE (N4,210)
      CALL EXIT
C     FIND SONIC POINT
  110 L2=L
      THFO=THL+DT
      DO 120 ICOUNT=1,20
      DTHL=-DTHL*(SNEW-SCRIT)/(SNEW-SOLD)
      IF (DTHL.LT.TOL) GO TO 140
      THL=THL-DTHL
      SOLD=SNEW
      S=SFROMX(CMPLX(R1*COS(THL),R2*SIN(THL)))
      SNEW=REAL(S)*REAL(S)+AIMAG(S)*AIMAG(S)
  120 IF (ABS(SNEW-SCRIT).GT.ABS(SOLD-SCRIT)) GO TO 130
  130 WRITE (N4,200)
  140 CONTINUE
C     CONSTRUCT SUPERSONIC PATH IN XI-PLANE
      RAD=CABS(CMPLX(R1*COS(THL),R2*SIN(THL)))
      FAC=AMIN1(RAD/R1,(.96*R-1.)/(R-1.))
      FAC=1.+FAC*(R-1.)
      CALL PATH (XI,FAC,THL+.07,THL+.07,K1,K2)
      IF (ABS(THF-THL).LT..30) GO TO 150
      CALL PATH (XI,1.04*R,THL+.04,THL-.08,K2+1,K2)
      CALL PATH (XI,R,THL-.12,THF+.16,K2+1,K2)
      CALL PATH (XI,1.04*R,THF+.12,THF+.04,K2+1,K2)
      GO TO 160
  150 CALL PATH (XI,1.04*R,THL+.04,THF+.04,K2+1,K2)
  160 CALL PATH (XI,R,THF,THL,K2+1,K2)
      JJ=KK
      KC=1
      WRITE (N1) THF,THL
  170 CONTINUE
C     PASTE POINTS TO THE SONIC LOCUS
      KN=KK+1
      DO 180 J=KN,K2
      S=SFROMX(XI(J))
      ABSSQ=REAL(S)*REAL(S)+AIMAG(S)*AIMAG(S)
      XI(J)=XIOFS(S*SQRT(SCRIT/ABSSQ),XI(J-1))
  180 CONTINUE
      IF (K2.LE.3*LBM) RETURN
      MSG=10HSUPERSONIC
      WRITE (N4,190) MSG
      CALL EXIT
  190 FORMAT (////5X5H**** ,A10,23H PATH IS TOO LONG *****)
  200 FORMAT (///12X48H****NEWTON ITERATION FOR SONIC POINT DID NOT CON,
     112HRGE WELL****)
  210 FORMAT (///28H**SUPERSONIC SPEEDS AT TAIL.,29H CHANGE SPEED DISTRI
     1BUTION** )
      END
```

```
      SUBROUTINE PATH (XI,RAD,TH1,TH2,K1,LL)
C     CONSTRUCTS PATHS OF INTEGRATION
C     ENDPOINTS OF PATH ARE ENTERED AS VARIABLES IN THE RING
C     XIF IS THE POINT IN THE ELLIPSE HAVING RADIUS RAD AND ARGU-
C     MENT TH1 IN THE RING
C     PATH IS CONSTRUCTED FROM XIC AROUND XIB AND THEN TO XIF
C     IF PATH COMES TOO CLOSE TO XIA IT IS REPLACE BY A PAIR OF
C     PATHS
C     IF TH1 IS NOT EQUAL TO TH2 PATH CONTINUES IN ELLIPSE AND
C     CORRESPONDS TO CIRCULAR ARC IN RING PLANE OF RADIUS RAD
C     FROM TH1 TO TH2
C     CARE IS TAKEN TO AVOID CROSSING THE BRANCH CUT FROM XIA TO XIB
      COMMON /1/ NP,KBM,MODE,NF,NN,MM,NI,NX,IS,KC,LBM,JJ,KK,NB,NC,MRP,NJ
     1,KF,NBPS,NFC,NRN,NPTS,NT,IPLT,LOFF,KP,NOSE,NCAS
      COMMON /2/ PI,TP,GRID,TOL
      COMPLEX XIA,XIB,XIC,DZ
      COMMON /3/ R,R1,R2,RATC,PHMN,GAM,WTAIL,XIA,XIB,XIC,UA,UB,VA,VB,UC,
     1VC,TRANU,TRANL,CONE,CTWO,CTHR,EP,RN,SO,THNOS,DZ
      COMMON /G/ N1,N3,N4,N7,M1
      COMPLEX XI(1),ARC,DXI,ARC1,P1,XIF,ROT
      DATA ISW/0/
      DATA ALPH/0./
      IF (NB.EQ.1) GO TO 10
      IF (ISW.NE.0) GO TO 10
      ARC1=CONJG(XIB-XIA)*(XIC-XIB)/CABS(XIB-XIA)
      ALPH=ATAN2(AIMAG(ARC1),REAL(ARC1))
      ISW=1
   10 RA=.5*(RAD+1./RAD)
      RI=.5*(RAD-1./RAD)
      LL=K1-1
      P1=XI(LL)
      XIF=CMPLX(RA*COS(TH1),RI*SIN(TH1))
C     CHECK THAT XIF IS OUTSIDE OF CIRCLE AROUND XIB
      SCF=CABS((XIB-XIF)/(XIB-XIC))
      IF (SCF.LT.1.) XIF=XIB+1.2*(XIF-XIB)/SCF
      IF (LL.NE.NC) GO TO 30
      KF=NC+1
C     CONSTRUCT CIRCLE AROUND XIB
      ARC=XIB-XIA
      IF (NB.EQ.1) ARC=XIC-XIB
      ARC1=CONJG(XIF-XIB)*ARC/CABS(XIF-XIB)
      ARG=ATAN2(AIMAG(ARC1),REAL(ARC1))
      IF (ABS(ARG).GT.3.) ARG=SIGN(3.,ARG)
      ARG=ARG+ALPH
      IX=ABS(ARG)*32./TP
      IF (IX.EQ.0) GO TO 30
      DT=-ARG/FLOAT(IX*MRP)
```

```
      ROT=CMPLX(COS(DT),SIN(DT))
      LL=LL+IX*MRP
      DO 20 J=K1,LL
   20 XI(J)=XIB+(XI(J-1)-XIB)*ROT
      P1=XI(LL)
      KF=LL+1
   30 K1=LL+1
C     CONSTRUCT PATH FROM P1 TO XIF
      ARC=XIF-P1
      SCL=CABS(ARC)
      IX=SCL/GRID+1.-TOL
      DXI=ARC/FLOAT(IX*MRP)
      LL=LL+MRP*IX
      GRID2=GRID/2.
      DO 40 J=K1,LL
      XI(J)=XI(J-1)+DXI
   40 IF (CABS(XI(J)-XIA).LT.GRID) GO TO 50
      GO TO 80
C     PATH WILL CONSIST OF TWO ARCS TO AVOID XIA
   50 LL=LL-MRP*IX
      ARC1=(XI(J)-XIA)*CONJG(XIB-XIA)
      ARG=ATAN2(AIMAG(ARC1),REAL(ARC1))
      FAC=SIGN(1.,ARG)
      ARC1=FAC*CMPLX(0.,1.)
      P1=XIB+.8*(XIA-XIB)
      P1=P1+GRID*ARC1*(XIB-XIA)/CABS(XIB-XIA)
      ARC=P1-XI(LL)
      SCL=CABS(ARC)
      IX=SCL/GRID+1.-TOL
      DXI=ARC/FLOAT(IX*MRP)
      LL=LL+MRP*IX
      DO 60 J=K1,LL
   60 XI(J)=XI(J-1)+DXI
      K1=LL+1
      ARC=XIF-XI(LL)
      SCL=CABS(ARC)
      IX=SCL/GRID+1.-TOL
      DXI=ARC/FLOAT(IX*MRP)
      LL=LL+MRP*IX
      DO 70 J=K1,LL
   70 XI(J)=XI(J-1)+DXI
   80 IF (ABS(TH2-TH1).LT.TOL) RETURN
C     CONSTRUCT PATHS AROUND THE ELLIPSE
      KK=LL
      K1=LL+1
      IX=R1*TP/(GRID*FLOAT(NF))-TOL
      IX=MRP*(IX+1)*NF/NP
      IF (MODE.GT.0) GO TO 90
      IX=RAD*ABS(TH2-TH1)/GRID-TOL
      IX=(IX+1)*MRP
   90 DTH=(TH2-TH1)/FLOAT(IX)
      ANG=TH1
```

```
      LL=LL+IX
      DO 100 J=K1,LL
      ANG=ANG+DTH
  100 XI(J)=CMPLX(RA*COS(ANG),RI*SIN(ANG))
      RETURN
      END

      SUBROUTINE CONST (KS)
C     DETERMINES IMAGINARY CONSTANTS OF SINGULAR SOLUTIONS
C     KS=1 FOR SUPERSONIC PATHS
      COMMON /1/ NP,KBM,MODE,NF,NN,MM,NI,NX,IS,KC,LBM,JJ,KK,NB,NC,MRP,NJ
     1,KF,NBPS,NFC,NRN,NPTS,NT,IPLT,LOFF,KP,NOSE,NCAS
      COMPLEX XIA,XIB,XIC,DZ
      COMMON /3/ R,R1,R2,RATC,PHMN,GAM,WTAIL,XIA,XIB,XIC,UA,UB,VA,VB,UC,
     1VC,TRANU,TRANL,CONE,CTWO,CTHR,EP,RN,SO,THNOS,DZ
      COMMON /6/ FAIMG(3),FBIMG(3),SAIMG(3),SBIMG(3),XAIMG(3),XBIMG(3),Y
     1AIMG(3),YBIMG(3)
      REAL MACH
      COMMON /A/ GAMMA,MACH,RHOINF,EMU,QSTRSQ
      COMPLEX RH1,RH2,USIG,VSIG,FA,SA,VEL
      COMPLEX S,SP,X1,Y1,U,V,LAMDAP,LAMDAM,TAU,RHOSIG,S1,F1,I
      DATA I/(0.,1.)/
C     FIND LAMBDA AND TAU AT XI=XIA, ETA=CONJG(XIA)
      U=UA
      V=VA
      CALL LAMBDA (U,V,LAMDAP,LAMDAM,TAU,XIA,CONJG(XIA),1)
      QSA=UA*UA+VA*VA
      RHOA=RHO(QSA)
C     COMPUTE S AND DS/DXI AT XI = XIA
      SP=1.
      CALL STOFXI (XIA,S,SP)
      IF (NB.NE.1) GO TO 10
      GAMM=(GAMMA-1.)/2.
      CS=1./(MACH*MACH)+GAMM-GAMM*QSA
      VEL=CMPLX(UA,VA)
      CALL GETHSQ (CMPLX(QSA,0.),USIG,VSIG)
      USIG=CONJG(S)/(2.*REAL(VSIG)*VEL)
      VSIG=.5*I*VEL/CONJG(S)
      RHOSIG=-(UA*USIG+VA*VSIG)/CS
   10 DO 80 K=1,KS
      IF (KS.EQ.1) GO TO 50
      FAIMG(K)=0.
      SAIMG(K)=0.
      FBIMG(K)=0.
      SBIMG(K)=0.
```

```
      GO TO (20,30,40), K
   20 SAIMG(K)=-1./AIMAG(TAU)
      GO TO 50
   30 FAIMG(K)=1.
      GO TO 50
   40 FBIMG(K)=1.
   50 IF (NB.EQ.1) GO TO 60
      SBIMG(K)=-SAIMG(K)
      IF (KS.EQ.3) FBIMG(K)=FBIMG(K)-FAIMG(K)
C     FIND SINGULAR TERMS F1 AND S1 AT XI=XIA,ETA=CONJ(XIA)
   60 RH1=REAL(TAU)
      RH2=AIMAG(TAU)
      S1=(FAIMG(K)-RH1*SAIMG(K))/RH2
      F1=RH1*S1-RH2*SAIMG(K)
C     GET IMAGINARY CONSTANTS FOR X AND Y
      FA=CMPLX(REAL(F1),FAIMG(K))
      SA=CMPLX(REAL(S1),SAIMG(K))
      X1=(UA*FA-VA*(SA/RHOA))/QSA
      Y1=(UA*(SA/RHOA)+VA*FA)/QSA
      XAIMG(K)=AIMAG(X1)
      YAIMG(K)=AIMAG(Y1)
      IF (NB.NE.1) GO TO 70
      XX=FBIMG(K)+AIMAG(SP*(USIG*X1+VSIG*Y1))
      YY=AIMAG(SP*((USIG+UA*RHOSIG)*Y1-(VSIG+VA*RHOSIG)*X1))
      XAIMG(K)=(UA*XX-VA*YY)/QSA
      YAIMG(K)=(UA*YY+VA*XX)/QSA
C     GET IMAGINARY CONSTANTS XBIMG(K) AND YBIMG(K)
      XBIMG(K)=0.
      YBIMG(K)=-1.E-7
      GO TO 80
   70 QSB=UB*UB+VB*VB
      RHOB=RHO(QSB)
      XBIMG(K)=(UB*FBIMG(K)-VB*(SBIMG(K)/RHOB))/QSB
      YBIMG(K)=(UB*(SBIMG(K)/RHOB)+VB*FBIMG(K))/QSB
   80 CONTINUE
      RETURN
      END


      SUBROUTINE SOLVE (K2)
C     CONTROL ROUTINE FOR SOLUTION OF CHARACTERISTIC INITIAL
C     VALUE PROBLEM. GRID IS TRIANGULAR FOR SUBSONIC FLOW AND
C     RECTANGULAR FOR TRANSONIC AND SUPERSONIC FLOW
      COMMON /1/ NP,KBM,MODE,NF,NN,MM,NI,NX,IS,KC,LBM,JJ,KK,NB,NC,MRP,NJ
     1,KF,NBPS,NFC,NRN,NPTS,NT,IPLT,LOFF,KP,NOSE,NCAS
      COMPLEX XIA,XIB,XIC,DZ
```

```
      COMMON /3/ R,R1,R2,RATC,PHMN,GAM,WTAIL,XIA,XIB,XIC,UA,UB,VA,VB,UC,
     1VC,TRANU,TRANL,CONE,CTWO,CTHR,EP,RN,SO,THNOS,DZ
      COMPLEX ROOT,ROOT1
      COMMON /5/ ROOT,ROOT1
      COMPLEX F1,F2,F3,S1,S2,S3,PHI,PSI,U,V,LAMDAP,LAMDAM,TAU,XI,ETA
      COMMON PHI(585,1),SKA(390),PSI(585,1),SKB(390),XI(585),ETA(585),U(
     1585),V(585),F1(585),S1(585),F2(585),S2(585),F3(585),S3(585),LAMDAP
     2(585),LAMDAM(585),TAU(585)
      COMMON /2/ PI,TP,GRID,TOL
      COMMON /6/ FAIMG(3),FBIMG(3),SAIMG(3),SBIMG(3),XAIMG(3),XBIMG(3),Y
     1AIMG(3),YBIMG(3)
      COMMON /G/ N1,N3,N4,N7,M1
      COMMON /M/ SINGX,SINGY,SINGXO,SINGYO
      COMPLEX TEMP,ROOT2
      LOFF=-1
      IF (MODE.LE.0) LOFF=0
      K1=NC+1
      ROOT=(0.,1.)
C     IF IS.GT.3 SINGULAR TERMS HAVE ALL BEEN COMPUTED
      IF (IS.GT.3) GO TO 20
      IF (MODE.LT.0) GO TO 10
C     STORE JUMPS FOR USE IN READX
      FJUMP=FAIMG(IS)+FBIMG(IS)
      SJUMP=SAIMG(IS)+SBIMG(IS)
      XJUMP=XAIMG(IS)+XBIMG(IS)
      YJUMP=YAIMG(IS)+YBIMG(IS)
      WRITE (N1) FJUMP,SJUMP,XJUMP,YJUMP
   10 K1=2
      CALL GETFS1 (K2)
   20 IF (IS.GT.3) CALL GETPHI (K2,NX)
      IF (KC.GT.0) GO TO 70
C     SUBSONIC FLOW, TRIANGULAR GRID, CHARACTERISTIC INITIAL PATHS ARE
C     CONJUGATE, SO (XI(J),ETA(K))=CONJG(XI(K),ETA(J))
      DO 60 J=K1,K2
C     SET THE BRANCH FOR THE SQUARE ROOT IN LAMDA+ AND LAMDA-
      ROOT=(0.,1.)
C     GET VALUES OF FUNCTIONS AT XI(J),ETA(J-1) BY REFLECTION
      U(J-1)=CONJG(U(J))
      V(J-1)=CONJG(V(J))
      TEMP=CONJG(LAMDAP(J))
      LAMDAP(J-1)=CONJG(LAMDAM(J))
      LAMDAM(J-1)=TEMP
      TAU(J-1)=CONJG(-TAU(J))
      IF (IS.GT.3) GO TO 30
      F1(J-1)=CONJG(F1(J))
      S1(J-1)=CONJG(S1(J))
C  20 CHECK TO SEE IF WE ARE PAST XIC
      IF (J.LE.NB) GO TO 50
C     GET SINGULAR TERMS F2 AND S2 BY REFLECTION
      F2(J-1)=CONJG(F2(J))
      S2(J-1)=CONJG(S2(J))
      IF (J.LE.NC) GO TO 50
```

```
C     GET INHOMOGENEOUS TERMS F3 AND S3 BY REFLECTION
      F3(J-1)=CONJG(F3(J))
      S3(J-1)=CONJG(S3(J))
C     GET REGULAR TERMS PHI AND PSI BY REFLECTION
   30 DO 40 LX=1,NX
      PHI(J-1,LX)=CONJG(PHI(J,LX))
   40 PSI(J-1,LX+LOFF)=CONJG(PSI(J,LX+LOFF))
C     GET SOLUTION AT XI(J) FROM ETA(J) TO ETA(K2)
   50 CALL LINEJ (J,J,K2)
C     SAVE SOLUTION ALONG DIAGONAL POINTS
   60 IF (J.GE.NC) CALL SAVE (J,J,J)
      RETURN
C     SOLVE EQUATIONS ON RECTANGULAR GRID FOR TRANSONIC OR SUPERSONIC
C     PATHS
   70 ROOT=(0.,1.)
      ROOT1=(1.,0.)
      IR=R1*TP/GRID+1.-TOL
      IR=MRP*(1+IR/NF)
      IF (MODE.LT.0) IR=MRP
      KM=1+(NN-KK)/IR
      KO=K1-1
C     SOLVE DIFFERENTIAL EQUATIONS ON XI CHARACTERISTICS FROM KO TO JJ
      DO 80 J=K1,JJ
C     SAVE SOLUTION ALONG CHARACTERISTIC ETA(NC)
      IF (J.GT.NC) CALL GETPSI (NC,J-1)
C     FIND VALUES ON NEXT XI CHARACTERISTIC
      CALL LINIT (J,KO)
      ROOT2=ROOT
      CALL LINEJ (J,K1,NN)
   80 ROOT=ROOT2
      WRITE (N1) KC,KM
      LN=(MM-JJ)/IR+1
      IF (MODE.LT.0) WRITE (N1) LN
      LB=NN
      IF (MODE.LT.0) LB=KK
C     SAVE SOLUTION ON CHARACTERISTIC XI(JJ)
      DO 90 K=NC,LB
   90 CALL GETPSI (K,JJ)
      XX=SINGX
      YY=SINGY
      IF (MODE.LT.0) CALL SAVE (KK+1,NN,JJ)
      SINGX=XX
      SINGY=YY
      LN=MM-JJ
      DO 120 K=1,LN
      L=NN-K
      M=JJ+K
      CALL LINIT (M,KO)
      ROOT2=ROOT
      CALL LINEJ (M,K1,L)
      ROOT=ROOT2
      IF (MODE.LT.0) GO TO 100
```

```
C     SAVE SOLUTION ON ELLIPSE BOUNDARY FOR TRANSONIC PATHS
      CALL GETPSI (L,M)
      GO TO 110
  100 MK=(L-KK)/IR+1
C     SUPERSONIC PATH, SAVE SOLUTION ALONG REAL CHARACTERISTICS
      IF (MOD(M-JJ,IR).EQ.0) WRITE (N1) MK
      CALL GETPSI (KK,M)
      XX=SINGX
      YY=SINGY
      IF (L.GE.KK+1) CALL SAVE (KK+1,L,M)
      SINGX=XX
      SINGY=YY
  110 CONTINUE
  120 CONTINUE
      RETURN
      END


      SUBROUTINE LINEJ (J,K1,K2)
C     SOLVES DIFFERENCE EQUATIONS ALONG COMPLEX CHARACTERISTIC XI(J)
      COMMON /1/ NP,KBM,MODE,NF,NN,MM,NI,NX,IS,KC,LBM,JJ,KK,NB,NC,MRP,NJ
     1,KF,NBPS,NFC,NRN,NPTS,NT,IPLT,LOFF,KP,NOSE,NCAS
      COMMON /6/ FAIMG(3),FBIMG(3),SAIMG(3),SBIMG(3),XAIMG(3),XBIMG(3),Y
     1AIMG(3),YBIMG(3)
      COMMON /WWS/ W2,WS2,LAM2
      COMPLEX F1,F2,F3,S1,S2,S3,PHI,PSI,U,V,LAMDAP,LAMDAM,TAU,XI,ETA
      COMMON PHI(585,1),SKA(390),PSI(585,1),SKB(390),XI(585),ETA(585),U(
     1585),V(585),F1(585),S1(585),F2(585),S2(585),F3(585),S3(585),LAMDAP
     2(585),LAMDAM(585),TAU(585)
      COMPLEX T1,T2,WS3,W3,LAMF,L23,L13,WS13,W23,W1,W2,WS1,WS2,LAM1,LAM2
      COMPLEX DXI,XIH,U3,V3,LAMBP,LAMBM,TAUL,LAMBPH,LAMBMH,TAUP,TAUM,DTA
     1UI,PHINC,RH1,RH2,RH3,RH4,ETAH,GE,ETGE,EF,DY,I
      COMMON /T/ PHINC(20)
      DATA I/(0.,1.)/
      NEX=1
      IF (NB.EQ.1) NEX=2
      DXI=XI(J)-XI(J-1)
      XIH=.5*(XI(J)+XI(J-1))
      DO 130 L=K1,K2
      IF (MODE.LT.0) GO TO 10
C     SOLVE FOR U AND V TO FIRST ORDER ACCURACY
      RH1=V(L)-LAMDAP(L)*U(L)
      RH2=V(L-1)-LAMDAM(L-1)*U(L-1)
      U3=(RH1-RH2)/(LAMDAM(L-1)-LAMDAP(L))
      V3=RH1+LAMDAP(L)*U3
C     COMPUTE LAMDA+,LAMDA- TO FIRST ORDER ACCURACY
```

```
      CALL LAMBDA (U3,V3,LAMBP,LAMBM,TAUL,XI(J),ETA(L),-1)
C     COMPUTE MIDPOINT VALUES OF LAMBDA+ AND LAMBDA-
      LAMBPH=.5*(LAMBP+LAMDAP(L))
      LAMBMH=.5*(LAMBM+LAMDAM(L-1))
C     SOLVE FOR U AND V TO SECOND ORDER ACCURACY
      RH1=V(L)-LAMBPH*U(L)
      RH2=V(L-1)-LAMBMH*U(L-1)
      U(L)=(RH1-RH2)/(LAMBMH-LAMBPH)
      V(L)=RH1+LAMBPH*U(L)
C     FIND LAMBDA+ AND LAMBDA- AT XI(J),ETA(L)
      CALL LAMBDA (U(L),V(L),LAMDAP(L),LAMDAM(L),TAUL,XI(J),ETA(L),1)
      GO TO 20
C     SUPERSONIC PATH.  USE ALTERNATE METHOD TO COMPUTE U,V AND TAU
   10 WS1=U(L)+CMPLX(-AIMAG(V(L)),REAL(V(L)))
      W1=U(L)-CMPLX(-AIMAG(V(L)),REAL(V(L)))
      CALL LAMBD (W1,WS1,LAM1,TAUL,XI(J),ETA(L),-1)
      T1=LAM2/(W2*W2)
      T2=LAM1/(WS1*WS1)
      WS3=(-T1*(T2*WS1-W1)-(T1*W2-WS2))/(1.-T1*T2)
      W3=T2*(WS3-WS1)+W1
      CALL LAMBD (W3,WS3,LAMF,TAUL,XI(J),ETA(L),-1)
      L23=0.5*(LAM2+LAMF)
      L13=0.5*(LAM1+LAMF)
      WS13=0.5*(WS3+WS1)
      W23=0.5*(W2+W3)
      T1=L23/(W23*W23)
      T2=L13/(WS13*WS13)
      T1=(-T1*(T2*WS1-W1)-(T1*W2-WS2))/(1.-T1*T2)
      W2=T2*(T1-WS1)+W1
      WS2=T1
      CALL LAMBD (W2,WS2,LAM2,TAUL,XI(J),ETA(L),1)
      U(L)=(W2+WS2)/2.
      V(L)=(WS2-W2)/(2.*I)
   20 CONTINUE
C     COMPUTE MIDPOINT VALUE FOR TAU
      TAUP=.5*(TAU(L-1)+TAUL)
      TAUM=-.5*(TAU(L)+TAUL)
      TAU(L)=TAUL
      DTAUI=1./(TAUM-TAUP)
      IF (IS.GT.3) GO TO 40
C     FIRST COMPUTE F1 AND S1 AT XI(J),ETA(L)
      RH1=F1(L)+TAUM*S1(L)
      RH2=F1(L-1)+TAUP*S1(L-1)
      S1(L)=(RH1-RH2)*DTAUI
      F1(L)=RH1-TAUM*S1(L)
      IF ((J.LE.NB).OR.(L.LT.NB)) GO TO 130
C     NOW COMPUTE F2 AND S2
      RH3=F2(L)+TAUM*S2(L)
      RH4=F2(L-1)+TAUP*S2(L-1)
      IF (L.NE.NB) GO TO 30
C     INITIALIZE AT ETA(NB),J GREATER THAN NB
      RH4=I*(FBIMG(IS)+TAU(L)*SBIMG(IS))
```

```
      S2(L)=(RH3-RH4)/(TAUM-TAU(L))
      F2(L)=RH3-TAUM*S2(L)
      GO TO 130
   30 CONTINUE
C     FIND F2 AND S2 FOR J AND L GREATER THAN NB
      S2(L)=(RH3-RH4)*DTAUI
      F2(L)=RH3-TAUM*S2(L)
      IF ((J.LT.NC).OR.(L.LT.NC)) GO TO 130
      IF ((J.EQ.NC).AND.(L.EQ.NC)) GO TO 60
C     EF AND GE ARE INHOMOGENEOUS TERMS
C     F3 AND S3 ARE REAL FUNCTIONS FOR SUBSONIC AND TRANSONIC PATHS
      ETAH=.5*(ETA(L)+ETA(L-1))
      GE=.5*((RH2-I*(FAIMG(IS)+TAUP*SAIMG(IS)))/(ETA(1)-ETAH)**NEX+(RH4-
     1I*(FBIMG(IS)+TAUP*SBIMG(IS)))/(ETA(NB)-ETAH))
      IF (MODE.LT.0) GE=0.
      ETGE=(ETA(L)-ETA(L-1))*GE
      IF (J.EQ.NC) GO TO 80
      EF=((RH1+I*(FAIMG(IS)+TAUM*SAIMG(IS)))/(XI(1)-XIH)**NEX+(RH3+I*(FB
     1IMG(IS)+TAUM*SBIMG(IS)))/(XI(NB)-XIH))
      IF (MODE.GT.0) EF=.5*EF
   40 IF (L.EQ.NC) GO TO 100
C     COMPUTE SOLUTIONS FOR J AND L GREATER THAN NC
C     FIRST GET THE REGULAR TERMS PHI AND PSI
      DO 50 LX=1,NX
      RH1=PHI(L,LX)+TAUM*PSI(L,LX+LOFF)
      RH2=PHI(L-1,LX)+TAUP*PSI(L-1,LX+LOFF)
      PSI(L,LX+LOFF)=(RH1-RH2)*DTAUI
   50 PHI(L,LX)=RH1-TAUM*PSI(L,LX+LOFF)
      IF (IS.GT.3) GO TO 130
C     FIND F3 AND S3
      DY=DXI*EF
      RH1=F3(L)+TAUM*S3(L)+DY
      RH2=F3(L-1)+TAUP*S3(L-1)+ETGE
      S3(L)=(RH1-RH2)*DTAUI
      F3(L)=RH1-TAUM*S3(L)
      GO TO 130
C     FIND SOLUTIONS FOR J=NC AND L=NC
   60 DO 70 LX=1,NX
      PSI(NC,LX+LOFF)=0.
      CALL INITFN (ETA(NC),PHINC(LX),LX)
      PHI(NC,LX)=REAL(PHINC(LX))
   70 IF (MODE.LT.0) PHI(NC,LX)=PHINC(LX)
      F3(NC)=0.
      S3(NC)=0.
      GO TO 130
C     FIND SOLUTIONS FOR J=NC AND L GREATER THAN NC
   80 DO 90 LX=1,NX
      CALL INITFN (ETA(L),PHI(L,LX),LX)
      IF (MODE.LT.0) GO TO 90
      PHI(L,LX)=.5*(PHI(L,LX)+CONJG(PHINC(LX)))
   90 PSI(L,LX+LOFF)=PSI(L-1,LX+LOFF)-(PHI(L,LX)-PHI(L-1,LX))/TAUP
      F3(L)=0.
```

```
      S3(L)=S3(L-1)+ETGE/TAUP
      GO TO 130
C     FIND SOLUTIONS FOR J GREATER THAN NC AND L=NC
  100 IF (MODE.LT.0) GO TO 120
      DO 110 LX=1,NX
      CALL INITFN (CONJG(XI(J)),DY,LX)
      IF (MODE.GT.0) DY=.5*(PHINC(LX)+CONJG(DY))
      PSI(L,LX+LOFF)=PSI(L,LX+LOFF)-(DY-PHI(L,LX))/TAUM
  110 PHI(L,LX)=DY
  120 CONTINUE
      S3(L)=S3(L)+DXI*EF/TAUM
  130 CONTINUE
      IF (J.EQ.NB) CALL GETFS2 (K2)
      RETURN
      END


      SUBROUTINE GETFS1 (K2)
C     COMPUTE U AND V AND SINGULAR TERMS F1 AND S1 ON XI=XIA
C     FOR AIRFOIL ALSO DETERMINE F2 AND S2
      COMPLEX F1,F2,F3,S1,S2,S3,PHI,PSI,U,V,LAMDAP,LAMDAM,TAU,XI,ETA
      COMMON PHI(585,1),SKA(390),PSI(585,1),SKB(390),XI(585),ETA(585),U(
     1585),V(585),F1(585),S1(585),F2(585),S2(585),F3(585),S3(585),LAMDAP
     2(585),LAMDAM(585),TAU(585)
      COMMON /1/ NP,KBM,MODE,NF,NN,MM,NI,NX,IS,KC,LBM,JJ,KK,NB,NC,MRP,NJ
     1,KF,NBPS,NFC,NRN,NPTS,NT,IPLT,LOFF,KP,NOSE,NCAS
      COMPLEX XIA,XIB,XIC,DZ
      COMMON /3/ R,R1,R2,RATC,PHMN,GAM,WTAIL,XIA,XIB,XIC,UA,UB,VA,VB,UC,
     1VC,TRANU,TRANL,CONE,CTWO,CTHR,EP,RN,SO,THNOS,DZ
      COMMON /6/ FAIMG(3),FBIMG(3),SAIMG(3),SBIMG(3),XAIMG(3),XBIMG(3),Y
     1AIMG(3),YBIMG(3)
      REAL MACH
      COMMON /A/ GAMMA,MACH,RHOINF,EMU,QSTRSQ
      COMPLEX SOFXI,RH1,RH2,S,T,SP,I
      COMPLEX DTAUDX
      DATA I/(0.,1.)/
C     FIND LAMBDA AND TAU AT XI=XIA, ETA=CONJG(XIA)
C     COMPUTE S AT XI = XIA
      CALL STOFXI (XIA,S,SP)
      U(1)=UA
      V(1)=VA
      CALL LAMBDA (U(1),V(1),LAMDAP(1),LAMDAM(1),TAU(1),XIA,CONJG(XIA),1
     1)
C     FIND F1 AND S1 AT XI=XIA,ETA=CONJ(XIA)
      RH1=REAL(TAU(1))
      RH2=AIMAG(TAU(1))
```

```
      S1(1)=(FAIMG(IS)-RH1*SAIMG(IS))/RH2
      F1(1)=RH1*S1(1)-RH2*SAIMG(IS)
      IF (NB.NE.1) GO TO 10
      RH1=-1./TAU(1)
      RH2=DTAUDX(U(1),V(1))*(S1(1)+I*SAIMG(IS))*RH1
      F2(1)=(AIMAG(RH2)-SBIMG(IS)-REAL(RH1)*FBIMG(IS))/AIMAG(RH1)
      S2(1)=REAL(RH2)-REAL(RH1*(F2(1)+I*FBIMG(IS)))
   10 DO 20 L=2,K2
C     COMPUTE HODOGRAPH VARIABLE T(ETA(L))
      CALL STOFXI (CONJG(ETA(L)),SOFXI,0.)
      T=CONJG(SOFXI)
      U(L)=U(L-1)
      V(L)=V(L-1)
      CALL GETUV (S,T,U(L),V(L))
      CALL LAMBDA (U(L),V(L),LAMDAP(L),LAMDAM(L),TAU(L),XIA,ETA(L),1)
C     FIND F1 AND S1 ALONG XI=XIA AT ETA(L)
C     COMPUTE TAU+ AT MIDPOINT
      TAUP=.5*(TAU(L)+TAU(L-1))
      RH1=F1(L-1)+TAUP*S1(L-1)
      RH2=-I*(FAIMG(IS)-TAU(L)*SAIMG(IS))
      S1(L)=(RH1-RH2)/(TAUP+TAU(L))
      F1(L)=RH1-TAUP*S1(L)
      IF (NB.NE.1) GO TO 20
C     FOR AIRFOIL GET F2 AND S2
      RH1=F2(L-1)+TAUP*S2(L-1)
      RH2=-I*(FBIMG(IS)-TAU(L)*SBIMG(IS))
      RH2=RH2+DTAUDX(U(L),V(L))*(S1(L)+I*SAIMG(IS))
      S2(L)=(RH1-RH2)/(TAUP+TAU(L))
      F2(L)=RH1-TAUP*S2(L)
   20 CONTINUE
      RETURN
      END


      SUBROUTINE GETFS2 (K2)
C     COMPUTE SINGULAR TERMS F2 AND S2 ON XI=XIB
      COMMON /1/ NP,KBM,MODE,NF,NN,MM,NI,NX,IS,KC,LBM,JJ,KK,NB,NC,MRP,NJ
     1,KF,NBPS,NFC,NRN,NPTS,NT,IPLT,LOFF,KP,NOSE,NCAS
      COMMON /6/ FAIMG(3),FBIMG(3),SAIMG(3),SBIMG(3),XAIMG(3),XBIMG(3),Y
     1AIMG(3),YBIMG(3)
      COMPLEX F1,F2,F3,S1,S2,S3,PHI,PSI,U,V,LAMDAP,LAMDAM,TAU,XI,ETA
      COMMON PHI(585,1),SKA(390),PSI(585,1),SKB(390),XI(585),ETA(585),U(
     1585),V(585),F1(585),S1(585),F2(585),S2(585),F3(585),S3(585),LAMDAP
     2(585),LAMDAM(585),TAU(585)
      COMPLEX I,RH1,RH2,TAUP
      DATA I/(0.,1.)/
```

```
C     FIRST FIND F2 AND S2 AT XIB,CONJG(XIB)
      RH1=-1./TAU(NB)
      F2(NB)=(-SBIMG(IS)-REAL(RH1)*FBIMG(IS))/AIMAG(RH1)
      S2(NB)=-REAL(RH1*(F2(NB)+I*FBIMG(IS)))
C     NOW FIND F2,S2 ALONG INITIAL CHARACTERISTIC XI=XIB
      NBP=NB+1
      DO 10 L=NBP,K2
C     COMPUTE TAU+ AT MIDPOINT
      TAUP=.5*(TAU(L)+TAU(L-1))
      RH1=F2(L-1)+TAUP*S2(L-1)
      RH2=-I*(FBIMG(IS)-TAU(L)*SBIMG(IS))
      S2(L)=(RH1-RH2)/(TAUP+TAU(L))
   10 F2(L)=RH1-TAUP*S2(L)
      RETURN
      END


      SUBROUTINE GETPHI (K2,KX)
C     FIND U AND V AND REGULAR SOLUTIONS ON CHARACTERISTIC XIC
C     USED ONLY WHEN NO SINGULAR TERMS ARE COMPUTED
      COMPLEX F1,F2,F3,S1,S2,S3,PHI,PSI,U,V,LAMDAP,LAMDAM,TAU,XI,ETA
      COMMON PHI(585,1),SKA(390),PSI(585,1),SKB(390),XI(585),ETA(585),U(
     1585),V(585),F1(585),S1(585),F2(585),S2(585),F3(585),S3(585),LAMDAP
     2(585),LAMDAM(585),TAU(585)
      COMMON /1/ NP,KBM,MODE,NF,NN,MM,NI,NX,IS,KC,LBM,JJ,KK,NB,NC,MRP,NJ
     1,KF,NBPS,NFC,NRN,NPTS,NT,IPLT,LOFF,KP,NOSE,NCAS
      COMPLEX XIA,XIB,XIC,DZ
      COMMON /3/ R,R1,R2,RATC,PHMN,GAM,WTAIL,XIA,XIB,XIC,UA,UB,VA,VB,UC,
     1VC,TRANU,TRANL,CONE,CTWO,CTHR,EP,RN,SO,THNOS,DZ
      COMPLEX PHINC,S,T,SOFXI,TAUP
      DIMENSION PHINC(20)
      NCP=NC+1
C     CALCULATE COEFFICIENTS ON INITIAL CHARACTERISTIC XIC
C     FIND U AND V ON INTIAL CHARACTERISTIC
      U(NC)=UC
      V(NC)=VC
      CALL LAMBDA (U(NC),V(NC),LAMDAP(NC),LAMDAM(NC),TAU(NC),XIC,CONJG(X
     1IC),1)
C     INITIALIZATION OF PHI AND PSI
      DO 10 LX=1,KX
      CALL INITFN (ETA(NC),PHINC(LX),LX)
      PSI(NC,LX+LOFF)=0.
   10 PHI(NC,LX)=REAL(PHINC(LX))
C     COMPUTE S AT XI=XIA
      CALL STOFXI (XI(NC),S,0.)
C     FIND U,V,LAMBDA AND TAU ALONG XI=XIC
```

```
      DO 30 L=NCP,K2
C     COMPUTE HODOGRAPH VARIABLE T(ETA(L))
      CALL STOFXI ((CONJG(ETA(L))),SOFXI,0.)
      T=CONJG(SOFXI)
      U(L)=U(L-1)
      V(L)=V(L-1)
      CALL GETUV (S,T,U(L),V(L))
      CALL LAMBDA (U(L),V(L),LAMDAP(L),LAMDAM(L),TAU(L),XIC,ETA(L),1)
C     COMPUTE TAU+ AT MIDPOINT
      TAUP=.5*(TAU(L)+TAU(L-1))
      DO 20 LX=1,KX
      CALL INITFN (ETA(L),PHI(L,LX),LX)
      PHI(L,LX)=.5*(PHI(L,LX)+CONJG(PHINC(LX)))
   20 PSI(L,LX+LOFF)=PSI(L-1,LX+LOFF)-(PHI(L,LX)-PHI(L-1,LX))/TAUP
   30 CONTINUE
      RETURN
      END


      SUBROUTINE INITFN (ETA,Z,LJ)
C     COMPUTES DATA Z ON THE INITIAL CHARACTERISTIC XI = XIC
      COMMON /1/ NP,KBM,MODE,NF,NN,MM,NI,NX,IS,KC,LBM,JJ,KK,NB,NC,MRP,NJ
     1,KF,NBPS,NFC,NRN,NPTS,NT,IPLT,LOFF,KP,NOSE,NCAS
      COMPLEX XIA,XIB,XIC,DZ
      COMMON /3/ R,R1,R2,RATC,PHMN,GAM,WTAIL,XIA,XIB,XIC,UA,UB,VA,VB,UC,
     1VC,TRANU,TRANL,CONE,CTWO,CTHR,EP,RN,SO,THNOS,DZ
      COMPLEX BP,ETJ
      COMMON /8/ BP(65),ETJ(65)
      COMPLEX I,ETA,Z,SIG,TEMP
      EXTERNAL SIG
      DATA I/(0.,1.)/
      IF (MODE.LE.0) GO TO 10
C     INITIAL DATA FOR (LJ+NJ)TH TERM IS REAL OR IMAGINARY PART OF
C     LXTH CHEBYSHEV POLYNOMIAL
      LX=LJ+IABS(NJ)
      J=(LX+1)/2
      TEMP=1.
      IF (LX.NE.J+J) TEMP=-I
      Z=SIG(ETA)
      AJ=FLOAT(J)
      DEN=AJ*ALOG(R)
      Z=TEMP*(CEXP(AJ*CLOG(Z)-DEN)+CEXP(-AJ*CLOG(Z)-DEN))
      RETURN
C     INITIALIZE REGULAR TERM FOR SUPERSONIC PATH
   10 NF2=NF/2
      Z=SIG(CONJG(ETA))
```

```
      TEMP=0.
      DO 20 J=1,NF2
      AJ=FLOAT(J)
      DEN=AJ*ALOG(R)
   20 TEMP=TEMP+ETJ(J)*(CEXP(AJ*CLOG(Z)-DEN)+CEXP(-AJ*CLOG(Z)-DEN))
      Z=CONJG(TEMP)
      RETURN
      END


      SUBROUTINE LINIT (J1,K1)
C     INITIALIZES INTEGRATION ON NEW CHARACTERISTIC FOR RECTANGULAR GRID
      COMMON /1/ NP,KBM,MODE,NF,NN,MM,NI,NX,IS,KC,LBM,JJ,KK,NB,NC,MRP,NJ
     1,KF,NBPS,NFC,NRN,NPTS,NT,IPLT,LOFF,KP,NOSE,NCAS
      COMPLEX XIA,XIB,XIC,DZ
      COMMON /3/ R,R1,R2,RATC,PHMN,GAM,WTAIL,XIA,XIB,XIC,UA,UB,VA,VB,UC,
     1VC,TRANU,TRANL,CONE,CTWO,CTHR,EP,RN,SO,THNOS,DZ
      COMMON /6/ FAIMG(3),FBIMG(3),SAIMG(3),SBIMG(3),XAIMG(3),XBIMG(3),Y
     1AIMG(3),YBIMG(3)
      COMPLEX W2,WS2,LAM2
      COMMON /WWS/ W2,WS2,LAM2
      COMPLEX F1,F2,F3,S1,S2,S3,PHI,PSI,U,V,LAMDAP,LAMDAM,TAU,XI,ETA
      COMMON PHI(585,1),SKA(390),PSI(585,1),SKB(390),XI(585),ETA(585),U(
     1585),V(585),F1(585),S1(585),F2(585),S2(585),F3(585),S3(585),LAMDAP
     2(585),LAMDAM(585),TAU(585)
      COMPLEX I,S,T,RH1,RH2,SFROMX,TAUP,TAUM,DTAUDX,PHINC
      COMMON /T/ PHINC(20)
      DATA I/(0.,1.)/
C     COMPUTE S AT XI(J1)
      S=SFROMX(XI(J1))
C     COMPUTE T AT ETA(K1)
      T=CONJG(SFROMX(CONJG(ETA(K1))))
      CALL GETUV (S,T,U(K1),V(K1))
      CALL LAMBDA (U(K1),V(K1),LAMDAP(K1),LAMDAM(K1),TAUP,XI(J1),ETA(K1)
     1,1)
      IF (MODE.GT.0) GO TO 10
      WS2=U(K1)+CMPLX(-AIMAG(V(K1)),REAL(V(K1)))
      W2=U(K1)-CMPLX(-AIMAG(V(K1)),REAL(V(K1)))
      CALL LAMBD (W2,WS2,LAM2,TAU(K1),XI(J1),ETA(K1),-1)
   10 CONTINUE
C     COMPUTE TAU- AT THE MIDPOINT
      TAUM=-.5*(TAU(K1)+TAUP)
C     STORE NEW TAU+
      TAU(K1)=TAUP
      IF (K1.EQ.NC) RETURN
C     FIND F1 AND S1 AT (CONJG(XIA),XI(J1))
```

```
      RH1=F1(1)+TAUM*S1(1)
      RH2=I*(FAIMG(IS)+TAUP*SAIMG(IS))
      S1(1)=(RH1-RH2)/(TAUM-TAUP)
      F1(1)=RH1-TAUM*S1(1)
      IF (NB.NE.1) GO TO 20
C     FIND F2 AND S2 FOR AIRFOIL
      RH1=F2(1)+TAUM*S2(1)
      RH2=CONJG(DTAUDX(CONJG(U(1)),CONJG(V(1))))*(S1(1)-I*SAIMG(IS))
      RH2=I*(FBIMG(IS)+TAUP*SBIMG(IS))+RH2
      S2(1)=(RH1-RH2)/(TAUM-TAUP)
      F2(1)=RH1-TAUM*S2(1)
      RETURN
C     F2 AND S2 ARE INITIALIZED IN LINEJ FOR CASCADE
   20 F2(NB-1)=CMPLX(0.,1.)*FBIMG(IS)
      S2(NB-1)=CMPLX(0.,1.)*SBIMG(IS)
      RETURN
      END


      SUBROUTINE SAVE (K1,K2,K3)
C     IN REAL FLOW DOMAIN INTEGRATES TO FIND X AND Y AND SAVES THE
C     SOLUTIONS ON TAPE
      REAL LOG1,LOG2
      COMMON /1/ NP,KBM,MODE,NF,NN,MM,NI,NX,IS,KC,LBM,JJ,KK,NB,NC,MRP,NJ
     1,KF,NBPS,NFC,NRN,NPTS,NT,IPLT,LOFF,KP,NOSE,NCAS
      COMMON /2/ PI,TP,GRID,TOL
      COMPLEX XIA,XIB,XIC,DZ
      COMMON /3/ R,R1,R2,RATC,PHMN,GAM,WTAIL,XIA,XIB,XIC,UA,UB,VA,VB,UC,
     1VC,TRANU,TRANL,CONE,CTWO,CTHR,EP,RN,SO,THNOS,DZ
      COMMON /6/ FAIMG(3),FBIMG(3),SAIMG(3),SBIMG(3),XAIMG(3),XBIMG(3),Y
     1AIMG(3),YBIMG(3)
      COMPLEX BP,ETJ
      COMMON /8/ BP(65),ETJ(65)
      REAL MACH
      COMMON /A/ GAMMA,MACH,RHOINF,EMU,QSTRSQ
      COMMON /G/ N1,N3,N4,N7,M1
      COMMON /M/ SINGX,SINGY,SINGXO,SINGYO
      COMPLEX F1,F2,F3,S1,S2,S3,PHI,PSI,U,V,LAMDAP,LAMDAM,TAU,XI,ETA
      COMMON PHI(585,1),SKA(390),PSI(585,1),SKB(390),XI(585),ETA(585),U(
     1585),V(585),F1(585),S1(585),F2(585),S2(585),F3(585),S3(585),LAMDAP
     2(585),LAMDAM(585),TAU(585)
      COMPLEX TEMP,ROTATE
      COMMON /TT/ ROTATE,THETAP,THETA,UOLD,VOLD,SINGF,SINGS,PHIOLD(20),P
     1SIOLD(20),XOLD(20),YOLD(20),THLAST
      LOGICAL ISW
      IF (IS.GT.3) GO TO 50
```

```
      TEMP=CLOG((XI(K3)-XI(1))/(XI(K3)-XI(NB)))
      LOG1=REAL(TEMP)
      IF (K3.NE.NC) GO TO 20
C     INITIALIZE TERMS NEEDED FOR INTEGRATION
      UOLD=U(NC)
      VOLD=V(NC)
      SINGX=0.
      SINGY=0.
      DO 10 LX=1,NX
      XOLD(LX)=0.
   10 YOLD(LX)=0.
      IR=R1*TP/GRID+1.-TOL
      IR=MRP*(1+IR/NF)
      M=1+(NN-KK)/IR
      THETA=AIMAG(TEMP)
      THETAP=AIMAG(CLOG(XI(NC)-XI(NB)))
      ROTATE=CMPLX(COS(THETAP),-SIN(THETAP))
   20 ISW=.TRUE.
C     CHECK THAT WE ARE NOT CROSSING THE BRANCH CUT
      IF (MODE.LT.0) IR=MRP
      DTH=THETA-AIMAG(TEMP)
      THETA=AIMAG(TEMP)
      IF (ABS(DTH).LT.PI) GO TO 30
      WRITE (N4,160)
C     CHECK FOR JUMP IN BRANCH OF LOGARITHM
   30 TEMP=CLOG((XI(K3)-XI(NB))*ROTATE)
      LOG2=REAL(TEMP)
      THETAP=AIMAG(TEMP)+THETAP
      ROTATE=CMPLX(COS(THETAP),-SIN(THETAP))
      IF (MODE.LT.0) GO TO 40
      IF (K3.NE.JJ) GO TO 40
      IF (MODE.EQ.1) THLAST=THETAP
      DTH=THETAP-THLAST
      IF (ABS(DTH).LT.PI) GO TO 40
C     THIS PATH AND THE LAST FORM A CLOSED LOOP ABOUT XIB
C     SET ISW TO MAKE AN APPROPRIATE ADJUSTMENT FOR X AND Y INTEGRATION
      ISW=.FALSE.
      DTH=(PI+PI)*DTH/ABS(DTH)
      THETAP=THETAP-DTH
      FJUMP=FBIMG(IS)
      SJUMP=SBIMG(IS)
      XJUMP=XAIMG(IS)+XBIMG(IS)
      YJUMP=YAIMG(IS)+YBIMG(IS)
      IF (NB.EQ.1) GO TO 40
      FJUMP=FAIMG(IS)+FBIMG(IS)
      SJUMP=SAIMG(IS)+SBIMG(IS)
   40 THETA2=THETAP
      IF (K1.EQ.NN) THLAST=THETA2
      IF (NB.EQ.1) TEMP=1./(XI(K3)-XI(1))
C     FIND COMPONENTS OF PHI,PSI,X AND Y FROM ETA(K1) TO ETA(K2)
   50 DO 150 L=K1,K2
      UNEW=REAL(U(L))
```

```
      VNEW=REAL(V(L))
      UH=.5*(UOLD+UNEW)
      VH=.5*(VOLD+VNEW)
      QS=UH*UH+VH*VH
      IF (IS.GT.3) GO TO 100
      SINGFO=SINGF
      SINGSO=SINGS
      IF (NB.EQ.1) GO TO 60
C     COMPUTE SINGULAR TERMS FOR CASCADE
      SINGF=REAL(F1(L))*LOG1+(REAL(F1(L))+REAL(F2(L)))*LOG2-FAIMG(IS)*TH
     1ETA-(FAIMG(IS)+FBIMG(IS))*THETA2+REAL(F3(L))
      SINGS=REAL(S1(L))*LOG1+(REAL(S1(L))+REAL(S2(L)))*LOG2-SAIMG(IS)*TH
     1ETA-(SAIMG(IS)+SBIMG(IS))*THETA2+REAL(S3(L))
      GO TO 70
C     COMPUTE SINGULAR TERMS FOR ISOLATED AIRFOIL
   60 SINGF=REAL((F1(L)+CMPLX(0.,FAIMG(IS)))*TEMP)+REAL(F2(L))*LOG2-FBIM
     1G(IS)*THETA2+REAL(F3(L))
      SINGS=REAL((S1(L)+CMPLX(0.,SAIMG(IS)))*TEMP)+REAL(S2(L))*LOG2-SBIM
     1G(IS)*THETA2+REAL(S3(L))
   70 IF (MODE.GT.0) GO TO 80
      SINGF=SINGF+REAL(PHI(L,1))
      SINGS=SINGS+REAL(PSI(L,1))+ETJ(65)
   80 IF (L.EQ.NC) GO TO 100
      IF (ISW) GO TO 90
C     MODIFY THE SINGULAR PART OF X AND Y TO ACCOUNT FOR JUMP IN LOG
      SINGFO=SINGFO+DTH*FJUMP
      SINGSO=SINGSO+DTH*SJUMP
      SINGX=SINGX+DTH*XJUMP
      SINGY=SINGY+DTH*YJUMP
C     INTEGRATE TO OBTAIN SINGX AND SINGY
   90 DPHI=SINGF-SINGFO
      DPSI=(SINGS-SINGSO)/RHO(QS)
      SINGX=SINGX+(UH*DPHI-VH*DPSI)/QS
      SINGY=SINGY+(UH*DPSI+VH*DPHI)/QS
  100 DO 120 LX=1,NX
      IF (L.EQ.NC) GO TO 110
C     INTEGRATE TO GET REGULAR TERMS FOR X AND Y
      DPHI=REAL(PHI(L,LX))-PHIOLD(LX)
      DPSI=(REAL(PSI(L,LX+LOFF))-PSIOLD(LX))/RHO(QS)
      XOLD(LX)=XOLD(LX)+(UH*DPHI-VH*DPSI)/QS
      YOLD(LX)=YOLD(LX)+(UH*DPSI+VH*DPHI)/QS
  110 PSIOLD(LX)=PSI(L,LX+LOFF)
      PHIOLD(LX)=PHI(L,LX)
  120 CONTINUE
C     STORE SOLUTIONS ON TAPE1
      IF (L.LT.KK) GO TO 140
      IF ((MOD(K3-JJ,IR).NE.0).OR.(MOD(L-KK,IR).NE.0)) GO TO 140
      IF (MODE.LT.0) GO TO 130
      IF (L.EQ.KK) WRITE (N1) KC,M
      WRITE (N1) UNEW,VNEW,(PHIOLD(K),PSIOLD(K),XOLD(K),YOLD(K),K=1,NX),
     1SINGF,SINGS,SINGX,SINGY
      GO TO 140
```

```
  130 WRITE (N1) XI(K3),ETA(L),UNEW,VNEW,SINGF,SINGS,SINGX,SINGY
  140 UOLD=UNEW
  150 VOLD=VNEW
      RETURN
  160 FORMAT (//6X49H****SUBSONIC PATH CROSSES CUT FROM XIA TO XIB****)
      END


      SUBROUTINE GETPSI (L,J)
C     FOR COMPLEX U AND V INTEGRATES TO OBTAIN X AND Y AT XI(J),ETA(L)
C     AND SAVES THE SOLUTIONS ON TAPE
      COMMON /1/ NP,KBM,MODE,NF,NN,MM,NI,NX,IS,KC,LBM,JJ,KK,NB,NC,MRP,NJ
     1,KF,NBPS,NFC,NRN,NPTS,NT,IPLT,LOFF,KP,NOSE,NCAS
      COMMON /2/ PI,TP,GRID,TOL
      COMPLEX XIA,XIB,XIC,DZ
      COMMON /3/ R,R1,R2,RATC,PHMN,GAM,WTAIL,XIA,XIB,XIC,UA,UB,VA,VB,UC,
     1VC,TRANU,TRANL,CONE,CTWO,CTHR,EP,RN,SO,THNOS,DZ
      COMMON /6/ FAIMG(3),FBIMG(3),SAIMG(3),SBIMG(3),XAIMG(3),XBIMG(3),Y
     1AIMG(3),YBIMG(3)
      COMPLEX BP,ETJ
      COMMON /8/ BP(65),ETJ(65)
      REAL MACH
      COMMON /A/ GAMMA,MACH,RHOINF,EMU,QSTRSQ
      COMMON /G/ N1,N3,N4,N7,M1
      COMMON /M/ SINGX,SINGY,SINGXO,SINGYO
      COMPLEX F1,F2,F3,S1,S2,S3,PHI,PSI,U,V,LAMDAP,LAMDAM,TAU,XI,ETA
      COMMON PHI(585,1),SKA(390),PSI(585,1),SKB(390),XI(585),ETA(585),U(
     1585),V(585),F1(585),S1(585),F2(585),S2(585),F3(585),S3(585),LAMDAP
     2(585),LAMDAM(585),TAU(585)
      COMMON /TT/ ROTATE,THETAP,THETA,UOLD,VOLD,SINGF,SINGS,PHIOLD(20),P
     1SIOLD(20),XOLD(20),YOLD(20),THLAST
      COMPLEX LOG1,LOG2,I,CLOG,CSINGF,CSINGS,DPHI,DPSI,CU,CV,CPHI(30),CP
     1SI(30),CRHO,UH,VH,QS,CEXP,SINGFO,SINGSO,ROTATE
      COMPLEX F1L,F2L,S1L,S2L,ROT2,F3L,F4L,S3L,S4L
      DIMENSION NM1(2)
      DATA I/(0.,1.)/
      IF (J.NE.NC) GO TO 20
C     INITIALIZE QUANTITIES FOR INTEGRATION
      GAMM=(GAMMA-1.)/2.
      COSQ=1./(MACH*MACH)+GAMM
      GAMI=-1./(GAMMA-1.)
      RHOINF=(MACH*MACH)**GAMI
      NM1=10H TRANSONIC
      IR=R1*TP/GRID+1.-TOL
      IR=MRP*(1+IR/NF)
      SINGX=0.
```

```
      SINGY=0.
      XJUMP=XAIMG(IS)+XBIMG(IS)
      YJUMP=YAIMG(IS)+YBIMG(IS)
      DO 10 LX=1,NX
      XOLD(LX)=0.
   10 YOLD(LX)=0.
      THETA=AIMAG(CLOG((XI(NC)-XI(1))/(XI(NC)-XI(NB))))
      THETAP=AIMAG(CLOG(XI(NC)-XI(NB)))
      ROTATE=CMPLX(COS(THETAP),-SIN(THETAP))
      IF (MODE.EQ.1) THLAST=THETAP
      ALPHA=-THETA
      ALPHAP=-THETAP
      ROT2=CONJG(ROTATE)
      CU=U(NC)
      CV=V(NC)
      FAC=CTHR
      IF (MODE.GE.0) GO TO 20
      IR=MRP
      NM1=10HSUPERSONIC
      SINGX=SINGXO
      SINGY=SINGYO
      FAC=0.
C     COMPUTE THE COMPLEX VELOCITY AND THE COMPLEX DENSITY
   20 UH=.5*(CU+U(L))
      VH=.5*(CV+V(L))
      QS=UH*UH+VH*VH
      CRHO=CEXP(-GAMI*CLOG(COSQ-GAMM*QS))/RHOINF
      IF (IS.GT.3) GO TO 100
      ISW=0
C     CHECK THAT CUT BETWEEN XIA AND XIB IS NOT CROSSED
      LOG1=CLOG((XI(J)-XI(1))/(XI(J)-XI(NB)))
      DTH=THETA-AIMAG(LOG1)
      THETA=AIMAG(LOG1)
      IF (ABS(DTH).GE.PI) WRITE (N4,160) NM1
C     COMPUTE THE COMPLEX LOGARITHM AND ADJUST THE BRANCH
      IF (NB.EQ.1) LOG1=1./(XI(J)-XI(1))
      LOG2=CLOG((XI(J)-XI(NB))*ROTATE)+CMPLX(0.,THETAP)
      IF (MODE.EQ.1) THLAST=AIMAG(LOG2)
      THETAP=AIMAG(LOG2)
      ROTATE=CMPLX(COS(THETAP),-SIN(THETAP))
      IF (MODE.LT.0) GO TO 30
C     CHOOSE THE BRANCH WHOSE ARGUMENT IS CLOSEST TO THLAST
      DTH=THETAP-THLAST
      IF (ABS(DTH).LT.PI) GO TO 30
C     COMPUTE JUMPS IN ALL QUANTITIES
      DTH=(PI+PI)*DTH/ABS(DTH)
      THETAP=THETAP-DTH
      LOG2=LOG2-CMPLX(0.,DTH)
      ISW=1
      XJUMP=XBIMG(IS)+XAIMG(IS)
      YJUMP=YBIMG(IS)+YAIMG(IS)
      SINGX=SINGX+DTH*XJUMP
```

```
      SINGY=SINGY+DTH*YJUMP
      THLAST=THETAP
      FJUMP=FBIMG(IS)
      SJUMP=SBIMG(IS)
      IF (NB.EQ.1) GO TO 125
      FJUMP=FJUMP+FAIMG(IS)
      SJUMP=SJUMP+SAIMG(IS)
  125 SINGFO=SINGFO+DTH*FJUMP
      SINGSO=SINGSO+DTH*SJUMP
   30 IF (NB.NE.1) LOG1=LOG1+LOG2
      IF ((J.EQ.JJ).AND.(L.EQ.NN)) THLAST=THETAP
      F1L=F1(L)+I*FAIMG(IS)
      F2L=F2(L)+I*FBIMG(IS)
      S1L=S1(L)+I*SAIMG(IS)
      S2L=S2(L)+I*SBIMG(IS)
C     COMPUTE THE SINGULAR SOLUTIONS
      CSINGF=F1L*LOG1+F2L*LOG2+F3(L)
      CSINGS=S1L*LOG1+S2L*LOG2+S3(L)
      IF (MODE.LE.0) GO TO 70
C     INCLUDE ETA TERMS IN SINGULAR SOLUTIONS ON TRANSONIC PATHS
      IF (NB.EQ.1) GO TO 40
C     CHECK THE SECOND TRANSONIC PATH DOES NOT CROSS BRANCH CUT
      LOG1=CLOG((ETA(L)-ETA(1))/(ETA(L)-ETA(NB)))
      DTH=ALPHA-AIMAG(LOG1)
      ALPHA=AIMAG(LOG1)
      IF (ABS(DTH).GE.PI) WRITE (N4,160) NM1
      GO TO 50
   40 LOG1=1./(ETA(L)-ETA(1))
   50 LOG2=CLOG((ETA(L)-ETA(NB))*ROT2)+CMPLX(0.,ALPHAP)
      ALPHAP=AIMAG(LOG2)
      ROT2=CMPLX(COS(ALPHAP),-SIN(ALPHAP))
C     CHECK BRANCH OF SECOND TRANSONIC PATH
      DTH=ALPHAP+THLAST
      IF (ABS(DTH).LT.PI) GO TO 60
      DTH=(PI+PI)*DTH/ABS(DTH)
      ISW=1-ISW
      IF (ISW.NE.0) GO TO 60
      ALPHAP=ALPHAP-DTH
      LOG2=LOG2-CMPLX(0.,DTH)
   60 IF (NB.NE.1) LOG1=LOG1+LOG2
      F3L=F1(L)-I*FAIMG(IS)
      F4L=F2(L)-I*FBIMG(IS)
      S3L=S1(L)-I*SAIMG(IS)
      S4L=S2(L)-I*SBIMG(IS)
C     MODIFY SINGULAR SOLUTIONS ON TRANSONIC PATHS
      CSINGF=.5*(CSINGF+F3L*LOG1+F4L*LOG2+F3(L))
      CSINGS=.5*(CSINGS+S3L*LOG1+S4L*LOG2+S3(L))
      GO TO 80
   70 CSINGF=CSINGF+PHI(L,1)
      CSINGS=CSINGS+PSI(L,1)+ETJ(65)
   80 IF (J.EQ.NC) GO TO 90
      DPHI=CSINGF-SINGFO
```

```
      DPSI=(CSINGS-SINGSO)/CRHO
C     FIND SINGULAR X AND Y TERMS
      SINGX=SINGX+(UH*DPHI-VH*DPSI)/QS
      SINGY=SINGY+(UH*DPSI+VH*DPHI)/QS
   90 SINGFO=CSINGF
      SINGSO=CSINGS
      SINGF=REAL(CSINGF)
      SINGS=REAL(CSINGS)
      SINGS=SINGS+FAC*AIMAG(CSINGS)
  100 CU=U(L)
      UOLD=CU
      CV=V(L)
      VOLD=CV
C     FIND REGULAR X AND Y TERMS
      DO 120 LX=1,NX
      IF (J.EQ.NC) GO TO 110
      DPHI=PHI(L,LX)-CPHI(LX)
      DPSI=(PSI(L,LX+LOFF)-CPSI(LX))/CRHO
      XOLD(LX)=XOLD(LX)+(UH*DPHI-VH*DPSI)/QS
      YOLD(LX)=YOLD(LX)+(UH*DPSI+VH*DPHI)/QS
  110 CPHI(LX)=PHI(L,LX)
      CPSI(LX)=PSI(L,LX+LOFF)
      PSIOLD(LX)=CPSI(LX)
      PSIOLD(LX)=PSIOLD(LX)+FAC*AIMAG(PSI(L,LX+LOFF))
  120 PHIOLD(LX)=PHI(L,LX)
C     STORE SOLUTIONS ON TAPE
      IF (MODE.LT.0) GO TO 140
      IF ((J.EQ.JJ).AND.(L.EQ.NN)) GO TO 130
      IF ((J.GT.JJ).AND.(MOD(J-JJ,IR).EQ.0)) GO TO 130
      RETURN
  130 WRITE (N1) UOLD,VOLD,(PHIOLD(K),PSIOLD(K),XOLD(K),YOLD(K),K=1,NX),
     1SINGF,SINGS,SINGX,SINGY
      RETURN
  140 IF ((J.LT.JJ).OR.(L.LT.KK)) GO TO 150
      IF ((MOD(J-JJ,IR).NE.0).OR.(MOD(L-KK,IR).NE.0)) GO TO 150
      WRITE (N1) XI(J),ETA(L),UOLD,VOLD,SINGF,SINGS,SINGX,SINGY
  150 RETURN
  160 FORMAT (//6X4H****,A10,37H PATH CROSSES CUT FROM XIA TO XIB****)
      END


      SUBROUTINE AUTO2 (RES,DT)
C     SOLVES LINEAR EQUATIONS FOR COEFFICIENTS OF STREAM FUNCTION PSI
C     AND FINDS ALL SOLUTIONS AROUND THE ELLIPSE
      COMPLEX DT,ROTATE
      COMMON PHS(129,3),RHX(129,3),RHY(129,3),RHS(129,3),U(129),V(129)
```

```
      COMMON ARRAY(128,128)
      COMMON /1/ NP,KBM,MODE,NF,NN,MM,NI,NX,IS,KC,LBM,JJ,KK,NB,NC,MRP,NJ
     1,KF,NBPS,NFC,NRN,NPTS,NT,IPLT,LOFF,KP,NOSE,NCAS
      COMMON /2/ PI,TP,GRID,TOL
      COMPLEX XIA,XIB,XIC,DZ
      COMMON /3/ R,R1,R2,RATC,PHMN,GAM,WTAIL,XIA,XIB,XIC,UA,UB,VA,VB,UC,
     1VC,TRANU,TRANL,CONE,CTWO,CTHR,EP,RN,SO,THNOS,DZ
      COMMON /6/ FAIMG(3),FBIMG(3),SAIMG(3),SBIMG(3),XAIMG(3),XBIMG(3),Y
     1AIMG(3),YBIMG(3)
      REAL MACHA,MACHB
      COMMON /7/ MACHA,MACHB,ANGLA,ANGLB
      COMPLEX BP,ETJ
      COMMON /8/ BP(65),ETJ(65)
      REAL MACH
      COMMON /A/ GAMMA,MACH,RHOINF,EMU,QSTRSQ
      COMMON /G/ N1,N3,N4,N7,M1
      COMPLEX U1,V1,F1,S1,LAMDAP,LAMDAM,TAU
      NT=NF+1
C     READ IN PSI
      CALL READX(2)
      FAC=SQRT(2.*(1.-MACH*MACH)/FLOAT(NF))
      ROOT2=1./SQRT(2.)
      CON1=1./SQRT(FLOAT(NF))
      DO 30 K=1,NF
      DO 10 J=2,NF
   10 ARRAY(K,J)=FAC*ARRAY(K,J)
      DO 20 J=1,3
   20 RHS(K,J)=RHS(K,J)*FAC
      ARRAY(K,NF)=ARRAY(K,NF)*ROOT2
   30 ARRAY(K,1)=CON1
C     FIND COEFFICIENTS OF REGULAR SOLUTIONS
      CALL LEQ (ARRAY,RHS,NF,3,128,129,SX)
C     THE FOLLOWING FILTER IS USED FOR THE KORN AIRFOIL
C     IT MAY BE USEFUL FOR OTHER DESIGN CASES
C     FILTER THE COEFFICIENTS OF PSI, PHI, X AND Y
C     DO 35 K=3,NF
C     DO 35 M=1,3
C     IFAC=(K-1)/2
C     COFAC=(PI+PI)*FLOAT(IFAC)/FLOAT(NF)
C     FILT=SIN(COFAC)/COFAC
C  35 RHS(K,M)=FILT*RHS(K,M)
C     READ IN PHI
      CALL READX(1)
      DO 60 K=1,NF
      DO 40 J=1,3
   40 RHX(K,J)=FAC*RHX(K,J)
      DO 50 J=2,NF
   50 ARRAY(K,J)=FAC*ARRAY(K,J)
   60 ARRAY(K,NF)=ARRAY(K,NF)*ROOT2
      DO 90 M=1,3
      DO 80 K=1,NF
      SX=-RHX(K,M)
```

```
      DO 70 J=2,NF
   70 SX=SX+RHS(J,M)*ARRAY(K,J)
   80 PHS(K,M)=SX
      PHS(NF+1,M)=PHS(1,M)
      IF (M.NE.3) GO TO 90
      PHS(NF+1,3)=PHS(1,3)+(PI+PI)*FAC
   90 CONTINUE
C     READ IN Y
      CALL READX(4)
      DO 110 M=1,3
      DO 110 K=1,NF
      IF (M.EQ.1) ARRAY(K,NF)=ARRAY(K,NF)*ROOT2
      SX=-RHX(K,M)
      DO 100 J=2,NF
  100 SX=SX+RHS(J,M)*ARRAY(K,J)
      IF (K.EQ.1) RHY(NT,M)=FAC*(SX-RHX(NT,M)+RHX(1,M))
  110 RHY(K,M)=FAC*SX
C     READ IN X
      CALL READX(3)
      DO 130 M=1,3
      DO 130 K=1,NF
      IF (M.EQ.1) ARRAY(K,NF)=ARRAY(K,NF)*ROOT2
      SX=-RHX(K,M)
      DO 120 J=2,NF
  120 SX=SX+RHS(J,M)*ARRAY(K,J)
      IF (K.EQ.1) RHX(NT,M)=FAC*(SX-RHX(NT,M)+RHX(1,M))
  130 RHX(K,M)=FAC*SX
C     FIND COEFFICIENTS OF EXPANSION FOR PHII, X AND Y
      CALL GETABC (AF,BF,CF,FAC,RES,CL)
C     ARRAY RHS CONTAINS COEFFICIENTS OF REGULAR TERMS OF EXPANSION
      DO 140 K=1,NF
      RHS(K,1)=AF*RHS(K,1)+BF*RHS(K,2)+CF*RHS(K,3)
  140 CONTINUE
      NFH=NF/2
      DO 150 J=1,NFH
  150 ETJ(J)=CMPLX(RHS(2*J+1,1),RHS(2*J,1))
      ETJ(NFH)=CMPLX(0.,RHS(NF,1)*ROOT2)
      ETJ(65)=RHS(1,1)/SQRT(2.*(1.-MACH*MACH))
C     FIND IMAGINARY CONSTANTS NEEDED FOR SUPERSONIC SOLUTION
      U1=UA
      V1=VA
      CALL LAMBDA (U1,V1,LAMDAP,LAMDAM,TAU,XIA,CONJG(XIA),1)
      FAREAL=-AF
      FAIMG(1)=-BF
      FBIMG(1)=-CF
      SBIMG(1)=0.
      IF (NB.NE.1) FBIMG(1)=FBIMG(1)-FAIMG(1)
      F1=CMPLX(FAREAL,FAIMG(1))
      S1=F1/TAU
      SAIMG(1)=AIMAG(S1)
      IF (NB.NE.1) SBIMG(1)=-SAIMG(1)
      CALL CONST (1)
```

```
      ANGROT=ATAN2(-XBIMG(1),-YBIMG(1))
      ROTATE=CMPLX(COS(ANGROT),SIN(ANGROT))
      DT=-(PI+PI)*CMPLX(XAIMG(1)+XBIMG(1),YAIMG(1)+YBIMG(1))*ROTATE
C     COMPUTE DIFFUSION FACTOR
      DF=0.
      IF (NB.EQ.1) GO TO 160
      QSB=UB*UB+VB*VB
      QSA=UA*UA+VA*VA
      X1=VA*YBIMG(1)+UA*XBIMG(1)
      Y1=VB*YBIMG(1)+UB*XBIMG(1)
      TURN=X1-Y1
      DF=1.-SQRT(QSA/QSB)+PI*TURN/SQRT(QSB)
  160 WRITE (N1) MACH,CL,DF,XAIMG(1),XBIMG(1),YAIMG(1),YBIMG(1),ROTATE
      RETURN
      END


      SUBROUTINE READX (INDX)
C     RESULTS OF INTEGRATION COLLECTED IN ARRAY FOR USE IN AUTO2
C     INDX = 1 FOR PHI, INDX=2 FOR PSI, INDX=3 FOR X, INDX=4 FOR Y
      COMMON PHS(129,3),RHX(129,3),RHY(129,3),RHS(129,3),U(129),V(129)
      COMMON ARRAY(128,128)
      COMMON /1/ NP,KBM,MODE,NF,NN,MM,NI,NX,IS,KC,LBM,JJ,KK,NB,NC,MRP,NJ
     1,KF,NBPS,NFC,NRN,NPTS,NT,IPLT,LOFF,KP,NOSE,NCAS
      COMMON /2/ PI,TP,GRID,TOL
      COMPLEX XIA,XIB,XIC,DZ
      COMMON /3/ R,R1,R2,RATC,PHMN,GAM,WTAIL,XIA,XIB,XIC,UA,UB,VA,VB,UC,
     1VC,TRANU,TRANL,CONE,CTWO,CTHR,EP,RN,SO,THNOS,DZ
      COMMON /G/ N1,N3,N4,N7,M1
      DIMENSION DATA(4,20), SING(4), BPER(4,3)
      DIMENSION SXY(4,10), RXY(4,129)
      IF (M1.NE.N1) REWIND M1
      REWIND N1
      READ (M1)
      READ (M1)
      READ (M1)
      KX=0
      NS=NF/NP
      NOSE=NF/(2*NP)
      DO 160 L=1,NP
      MX=0
      DO 110 I=2,NF,KBM
      M=I-1
      MX=MX+1
      NX=MIN0(KBM,NF-M)
      IF (MX.GT.3) GO TO 20
```

```
      READ (N1) (SING(IK),IK=1,4)
      DO 10 IK=1,4
   10 BPER(IK,MX)=SING(IK)
   20 READ (N1) K1,K2
      IF (M1.EQ.N1) GO TO 30
      IF (MX.LE.3) READ (M1)
      READ (M1)
   30 INC=-1
      K=KX+(K2-1)+1
C     K1.GT.0 FOR TRANSONIC AND SUPERSONIC PATHS
C     READ POINTS IN REVERSE ORDER FOR THESE PATHS
      IF (K1.GT.0) GO TO 40
      INC=1
      K=KX
   40 DO 110 LL=1,K2
C     SKIP PAST FIRST POINT FOR EACH SUBSONIC PATH
      IF ((LL.EQ.1).AND.(K1.EQ.0)) GO TO 80
C     SKIP PAST LAST POINT ON TRANSONIC PATHS
      IF ((LL.EQ.K2).AND.(K1.GT.0.)) GO TO 80
      K=K+INC
      READ (N1) UK,VK,((DATA(IK,J),IK=1,4),J=1,NX),(SING(IK),IK=1,4)
      IF (MX.GT.3) GO TO 50
      U(K)=UK
      V(K)=VK
      RHX(K,MX)=SING(INDX)
   50 DO 60 J=1,NX
   60 ARRAY(K,J+M)=DATA(INDX,J)
      IF (N1.EQ.M1) GO TO 110
      READ (M1) UK,VK,((DATA(IK,J),IK=1,4),J=1,NX),(SING(IK),IK=1,4)
C     DO RICHARDSON EXTRAPOLATION
      DO 70 J=1,NX
   70 ARRAY(K,J+M)=(4.*ARRAY(K,J+M)-DATA(INDX,J))/3.
      IF (MX.GT.3) GO TO 110
      RHX(K,MX)=(4.*RHX(K,MX)-SING(INDX))/3.
      GO TO 110
   80 IF (L.LE.2) GO TO 90
      READ (N1)
      IF (N1.NE.M1) READ (M1)
      GO TO 110
   90 CONTINUE
      READ (N1) DUM,DUM,(DUM,DUM,RXY(2*L-1,J+M),RXY(2*L,J+M),J=1,NX),DUM
     1,DUM,SXY(2*L-1,MX),SXY(2*L,MX)
      IF (M1.EQ.N1) GO TO 110
      READ (M1) DUM,DUM,(DUM,DUM,DATA(3,J),DATA(4,J),J=1,NX),DUM,DUM,SIN
     1G(3),SING(4)
C     RICHARDSON EXTRAPOLATION FOR LEADING EDGE CORRECTION
      DO 100 J=1,NX
      RXY(2*L-1,J+M)=(4.*RXY(2*L-1,J+M)-DATA(3,J))/3.
  100 RXY(2*L,J+M)=(4.*RXY(2*L,J+M)-DATA(4,J))/3.
      SXY(2*L-1,MX)=(4.*SXY(2*L-1,MX)-SING(3))/3.
      SXY(2*L,MX)=(4.*SXY(2*L,MX)-SING(4))/3.
  110 CONTINUE
```

```
C     CORRECT X AND Y NEAR LEADING EDGE
      IF ((L.NE.2).OR.(INDX.LE.2)) GO TO 160
      DO 130 M=2,NF
      COR=RXY(INDX,M)-ARRAY(NS,M)
      ARRAY(NOSE,M)=ARRAY(NOSE,M)-.5*COR
      DO 120 KZ=NOSE,NS
  120 ARRAY(KZ,M)=ARRAY(KZ,M)+COR
  130 CONTINUE
      DO 150 M=1,3
      COR=SXY(INDX,M)-RHX(NS,M)
      RHX(NOSE,M)=RHX(NOSE,M)-.5*COR
      DO 140 KZ=NOSE,NS
  140 RHX(KZ,M)=RHX(KZ,M)+COR
  150 CONTINUE
  160 KX=KX+(K2-1)
      IF (INDX.NE.2) GO TO 180
      DO 170 M=1,3
      DO 170 K=1,NF
  170 RHS(K,M)=RHX(K,M)
      RETURN
  180 IF (INDX.EQ.1) GO TO 230
C     SECOND CORRECTION AT LEADING EDGE
      DO 200 M=1,3
      BJUMP=SXY(INDX-2,M)-TP*BPER(INDX,M)
      COR=RHX(NF,M)-BJUMP
      RHX(NOSE,M)=RHX(NOSE,M)-.5*COR
      DO 190 KZ=1,NOSE
  190 RHX(KZ,M)=RHX(KZ,M)+COR
  200 CONTINUE
      DO 220 M=2,NF
      COR=ARRAY(NF,M)-RXY(INDX-2,M)
      ARRAY(NOSE,M)=ARRAY(NOSE,M)-.5*COR
      DO 210 KZ=1,NOSE
  210 ARRAY(KZ,M)=ARRAY(KZ,M)+COR
  220 CONTINUE
  230 DO 240 M=1,3
  240 RHX(NF+1,M)=RHX(1,M)-(PI+PI)*BPER(INDX,M)
      RETURN
      END


      SUBROUTINE GETABC (AF,BF,CF,CX,DMAX,CL)
C     FINDS COEFFICIENTS OF PHII AND COMPUTES FLOW QUANTITIES ON
C     THE ELLIPSE
      COMPLEX XIBODY
      COMMON PHS(129,3),RHX(129,3),RHY(129,3),RHS(129,3),U(129),V(129)
```

```
      COMMON DPHI(130,3),PHIPP(130,3),PHIP3(130,3),PHINT(130),DPHII(130)
     1,PHIIPP(130),PHIIP3(130),PHIX(130),PHII(130),STAG(130),XIBODY(200)
     2,UBODY(200),VBODY(200),XBODY(200),YBODY(200),OM(130),AX(130),ZERO(
     3300)
      DIMENSION THBODY(300)
      COMMON /1/ NP,KBM,MODE,NF,NN,MM,NI,NX,IS,KC,LBM,JJ,KK,NB,NC,MRP,NJ
     1,KF,NBPS,NFC,NRN,NPTS,NT,IPLT,LOFF,KP,NOSE,NCAS
      COMMON /2/ PI,TP,GRID,TOL
      COMPLEX XIA,XIB,XIC,DZ
      COMMON /3/ R,R1,R2,RATC,PHMN,GAM,WTAIL,XIA,XIB,XIC,UA,UB,VA,VB,UC,
     1VC,TRANU,TRANL,CONE,CTWO,CTHR,EP,RN,SO,THNOS,DZ
      COMMON /6/ FAIMG(3),FBIMG(3),SAIMG(3),SBIMG(3),XAIMG(3),XBIMG(3),Y
     1AIMG(3),YBIMG(3)
      REAL MACHA,MACHB
      COMMON /7/ MACHA,MACHB,ANGLA,ANGLB
      COMPLEX BP,ETJ
      COMMON /8/ BP(65),ETJ(65)
      COMMON /G/ N1,N3,N4,N7,M1
      COMMON /M/ SINGX,SINGY,SINGXO,SINGYO
      DIMENSION PHIWT(3), DPDWO(3), PHIWO(3)
      DATA RELAX/.8/
      N1=M1
      REWIND N1
      READ (N1)
      READ (N1)
      READ (N1) NT,(PHII(J),J=1,NT)
C     CONSTRUCT ARRAY OF EVENLY SPACED POINTS AROUND RING
      DW=TP/FLOAT(NF)
      NOSE=NF/(NP+NP)
      OM(1)=-FLOAT(NOSE-1)*DW+THNOS-TP
      DO 10 K=1,NF
      OM(K+1)=OM(K)+DW
      ZERO(K)=0.
   10 CONTINUE
      DO 20 M=1,3
C     SPLINE FIT EACH OF THE SINGULAR SOLUTIONS
      CALL PSPLIF (NT,OM,PHS(1,M),DPHI(1,M),PHIPP(1,M),PHIP3(1,M),PHINT)
      PHIWO(M)=PHS(NOSE,M)
      DPDWO(M)=(PHS(NOSE+1,M)-PHS(NOSE-1,M))/(DW+DW)
   20 CONTINUE
      CF=GAM/(PHS(NT,3)-PHS(1,3))
      WELL=GAM-PHMN
C     WTAIL IS RING ANGLE CORRESPONDING TO THE TAIL
C     THE FIRST EQUATION REQUIRES DPHI/DW=0 AT STAGNATION
      A11=DPDWO(1)
      A12=DPDWO(2)
C     ITERATE AT MOST 20 TIMES TO FIND WTAIL
      DO 70 J=1,20
      DO 30 M=1,3
C     EVALUATE PHI(WTAIL) FOR EACH SINGULAR SOLUTION
      CALL INTPL (1,WTAIL,PHIWT(M),OM,PHS(1,M),DPHI(1,M),PHIPP(1,M),PHIP
     13(1,M))
```

```
   30 CONTINUE
C     FIX PHI AT TRAILING EDGE - PHI AT STAGNATION
      A21=PHIWT(1)-PHIWO(1)
      A22=PHIWT(2)-PHIWO(2)
      B2=-CF*(PHIWT(3)-PHIWO(3))+WELL
      B1=-CF*DPDWO(3)
      BF=(A11*B2-A21*B1)/(A11*A22-A21*A12)
      AF=(A22*B1-A12*B2)/(A11*A22-A21*A12)
      DO 40 K=1,NT
      PHIX(K)=AF*PHS(K,1)+BF*PHS(K,2)+CF*PHS(K,3)
      DPHII(K)=AF*DPHI(K,1)+BF*DPHI(K,2)+CF*DPHI(K,3)
      PHIIPP(K)=AF*PHIPP(K,1)+BF*PHIPP(K,2)+CF*PHIPP(K,3)
      PHIIP3(K)=AF*PHIP3(K,1)+BF*PHIP3(K,2)+CF*PHIP3(K,3)
   40 CONTINUE
C     FIND THE LOCATION OF THE TAIL
      NSTAG=-NF
      CALL INTPLI (NSTAG,STAG,ZERO,NT,OM,DPHII,PHIIPP,PHIIP3,ZERO)
      WOLD=WTAIL
      WTAIL=STAG(1)
      IF (NSTAG.LE.1) GO TO 60
      DO 50 LL=2,NSTAG
      IF (ABS(STAG(LL)-WOLD).LT.ABS(WTAIL-WOLD)) WTAIL=STAG(LL)
   50 CONTINUE
   60 IF (ABS(WTAIL-WOLD).LT.TOL) GO TO 80
   70 CONTINUE
      WRITE (N4,130) WTAIL
   80 DMAX=0.
C     PHIX IS THE VELOCITY DISTRIBUTION AROUND THE ELLIPSE
      DO 90 J=1,NT
   90 PHIX(J)=AF*PHS(J,1)+BF*PHS(J,2)+CF*PHS(J,3)
      KTAIL=(WTAIL+PI)/DW
      KX=ABS((OM(1)-WTAIL)/DW)+1
      JX=-(PI+OM(1))/DW+1
      JX=-(PI+OM(1))/DW+1.-TOL
      PHINOS=PHIX(NOSE)
      XNOSE=AF*RHX(NOSE,1)+BF*RHX(NOSE,2)+CF*RHX(NOSE,3)
      YNOSE=AF*RHY(NOSE,1)+BF*RHY(NOSE,2)+CF*RHY(NOSE,3)
      RFAC=RELAX-1.
      KM=1
      U(NT)=U(1)
      V(NT)=V(1)
      DELX=-.5*(AF*(RHX(NT,1)-RHX(1,1))+BF*(RHX(NT,2)-RHX(1,2))+CF*(RHX(
     1NT,3)-RHX(1,3)))
      DELY=-.5*(AF*(RHY(NT,1)-RHY(1,1))+BF*(RHY(NT,2)-RHY(1,2))+CF*(RHY(
     1NT,3)-RHY(1,3)))
      RAT=(OM(KX+1)-WTAIL)/DW
      XIBODY(1)=CMPLX(R1*COS(WTAIL),R2*SIN(WTAIL))
C     GET U AND V AT TRAILING EDGE BY LINEAR INTERPOLATION
      UBODY(1)=RAT*U(KX)+(1.-RAT)*U(KX+1)
      VBODY(1)=RAT*V(KX)+(1.-RAT)*V(KX+1)
      THBODY(1)=OM(KX)
C     FIND ALL QUANTITIES AROUND THE ELLIPSE
```

```
      DO 110 K=1,NF
      J=K+KX
C     XBODY,YBODY,UBODY,VBODY ARE STORED FROM TAIL TO TAIL
      IF (J.GT.NF) J=J-NF
      IF (J.NE.1) GO TO 100
      DELX=-DELX
      DELY=-DELY
      XNOSE=XNOSE+DELX
      YNOSE=YNOSE+DELY
  100 XBODY(K+1)=AF*RHX(J,1)+BF*RHX(J,2)+CF*RHX(J,3)
      YBODY(K+1)=AF*RHY(J,1)+BF*RHY(J,2)+CF*RHY(J,3)
      XBODY(K+1)=XBODY(K+1)+DELX
      YBODY(K+1)=YBODY(K+1)+DELY
      XIBODY(K+1)=CMPLX(R1*COS(OM(J)),R2*SIN(OM(J)))
      THBODY(K+1)=THBODY(K)+DW
      VBODY(K+1)=V(J)
      UBODY(K+1)=U(J)
C     PHII IS STORED FROM -PI TO PI
      J=JX+K
      IF (J.EQ.NT) PHINOS=PHINOS-GAM
      IF (J.GT.NF) J=J-NF
      PHIIK=PHIX(J)-PHINOS
      AX(K)=PHIIK-PHII(K)
      PHII(K)=PHIIK+RFAC*AX(K)
      IF (K.LT.KTAIL) PHII(K)=AMAX1(PHII(K),PHII(KM))
      KM=K
      IF (ABS(AX(K)).LT.DMAX) GO TO 110
      DMAX=ABS(AX(K))
  110 CONTINUE
      XIBODY(NT+1)=XIBODY(1)
      UBODY(NT+1)=UBODY(1)
      VBODY(NT+1)=VBODY(1)
      THBODY(1)=WTAIL
      THBODY(NT+1)=WTAIL+TP
C     FIND X AND Y AT THE TRAILING EDGE BY FOUR POINT INTERPOLATION
      B1=.5*(1.+RAT)*(2.-RAT)
      B2=RAT*(RAT-1.)/6.
      XX=RAT*(XBODY(NF+1)-DELX)+(1.-RAT)*(XBODY(2)+DELX)
      YY=(2.-RAT)*(XBODY(NF)-DELX)+(1.+RAT)*(XBODY(3)+DELX)
      XX=B1*XX+B2*YY
      XBODY(1)=XX-DELX
      XBODY(NT+1)=XX+DELX
      XX=RAT*(YBODY(NF+1)-DELY)+(1.-RAT)*(YBODY(2)+DELY)
      YY=(2.-RAT)*(YBODY(NF)-DELY)+(1.+RAT)*(YBODY(3)+DELY)
      YY=B1*XX+B2*YY
      YBODY(1)=YY-DELY
      YBODY(NT+1)=YY+DELY
      PHII(NT)=PHII(1)+GAM
      NTP=NT+1
      NOSE=1+KX-(NOSE-1)
      CL=(CF+CF)*(PHS(NF+1,3)-PHS(1,3))
      DO 120 K=1,NTP
```

```
      XBODY(K)=XBODY(K)-XNOSE
  120 YBODY(K)=YBODY(K)-YNOSE
      SINGXO=DELX-XNOSE
      SINGYO=DELY-YNOSE
      AF=AF*CX
      BF=BF*CX
      CF=CF*CX
      REWIND N1
      READ (N1)
      READ (N1)
      WRITE (N1) NT,(PHII(J),J=1,NT)
      WRITE (N1) WTAIL,NTP,(XIBODY(J),THBODY(J),UBODY(J),VBODY(J),XBODY(
     1J),YBODY(J),J=1,NTP),NBPS,(BP(J),J=1,NBPS)
      RETURN
  130 FORMAT (* WTAIL DID NOT CONVERGE, WTAIL= *,E20.10)
      END


      SUBROUTINE OUTPT (TC,GAP,GC)
C     ORGANIZES OUTPUT DATA AND WRITES ON OUTPUT TAPE 4
      REAL MAVE
      REAL MACHN,KAPPA
      COMPLEX XI,XIBODY,ROT,XIP,ETAP,ROOTA,ROOTB,ROTATE
      COMMON SEPR(300),QL(300),XI(300),XIBODY(300),UBODY(300),VBODY(300)
     1,XBODY(300),YBODY(300),H(300),THETA(300),ANGL(300),MACHN(300),KAPP
     2A(300),XREAL(300),YREAL(300),S(300)
      COMMON FP(300),FPP(300),FPPP(300),GP(300),GPP(300),GPPP(300)
      COMMON ILINE(300),JLINK(300),NPT(300),OM(300)
      COMMON /1/ NP,KBM,MODE,NF,NN,MM,NI,NX,IS,KC,LBM,JJ,KK,NB,NC,MRP,NJ
     1,KF,NBPS,NFC,NRN,NPTS,NT,IPLT,LOFF,KP,NOSE,NCAS
      COMMON /2/ PI,TP,GRID,TOL
      COMPLEX XIA,XIB,XIC,DZ
      COMMON /3/ R,R1,R2,RATC,PHMN,GAM,WTAIL,XIA,XIB,XIC,UA,UB,VA,VB,UC,
     1VC,TRANU,TRANL,CONE,CTWO,CTHR,EP,RN,SO,THNOS,DZ
      COMMON /6/ FAIMG(3),FBIMG(3),SAIMG(3),SBIMG(3),XAIMG(3),XBIMG(3),Y
     1AIMG(3),YBIMG(3)
      REAL MACHA,MACHB
      COMMON /7/ MACHA,MACHB,ANGLA,ANGLB
      REAL MACH
      COMMON /A/ GAMMA,MACH,RHOINF,EMU,QSTRSQ
      COMMON /G/ N1,N3,N4,N7,M1
      EMACH(QS)=SQRT(QS/(COSQ-((GAMMA-1.)/2.)*QS))
      RAD=180./PI
      GAMM=.5*(GAMMA-1.)
      COSQ=GAMM+1./(MACH*MACH)
      MAVE=.5*(ABS(MACHA)+ABS(MACHB))
```

```
      ANGROT=ATAN2(-XBIMG(1),-YBIMG(1))
C     READ IN POINTS NEAR SONIC SURFACE AND WRITE ON OUTPUT TAPE
      END FILE N7
      REWIND N7
      IF (EOF(N7).NE.0) GO TO 20
      WRITE (N4,280)
      WRITE (N4,360)
   10 READ (N7) K1,XIP,ETAP,ROOTA,ROOTB
      IF (EOF(N7).NE.0) GO TO 20
      WRITE (N4,370) K1,XIP,ETAP,ROOTA,ROOTB
      GO TO 10
   20 CONTINUE
C     NOW USE TAPE7 TO STORE REAL SUPERSONIC CHARACTERSITICS FOR PLOTS
      REWIND N7
      IF (N1.NE.M1) REWIND N1
      REWIND M1
      READ (M1)
      READ (M1)
      READ (M1)
      READ (M1) WTAIL,KP,(XIBODY(J),OM(J),UBODY(J),VBODY(J),XBODY(J),YBO
     1DY(J),J=1,KP),NBPS,(ROTATE,J=1,NBPS)
      READ (M1) MACH,CL,DF,XAIMG(1),XBIMG(1),YAIMG(1),YBIMG(1),ROTATE
      XMIN=XBODY(1)
      DO 121 K=2,KP
  121 XMIN=AMIN1(XMIN,XBODY(K))
      XMAX=.5*(XBODY(1)+XBODY(KP))
      SCALE=1./(XMAX-XMIN)
      COSA=REAL(ROTATE)
      SINA=AIMAG(ROTATE)
C     FIND SUPERSONIC BODY POINTS
      READ (N1) THF,THL
      IF (N1.NE.M1) READ (M1) DUM,DUM
      KL=KP
      DTH=PI/FLOAT(NF)
      DO 30 J=1,KP
      ILINE(J)=0
      IF (OM(J).LT.THF) GO TO 30
      JF=J
      IF ((OM(J)-THF).GT.DTH) JF=J-1
      CALL BDYPTS (JF,THF,THL,KL,KM)
   30 CONTINUE
C     CREATE A POINTER LIST TO LOCATE BODY POINTS
      JLINK(1)=1
      KP=KL
      DO 40 K=2,KP
      JLINK(K)=JLINK(K-1)+1
   40 IF (ILINE(JLINK(K)).GT.0) JLINK(K)=ILINE(JLINK(K))
C     COMPUTE OUTPUT DATA AND INITIALIZE FOR NASHMCDONALD
      DIS=0.
C     DROP STAGNATION POINT IN ALL OUTPUT DATA
      L=0
      DO 50 M=1,KP
```

```
      K=JLINK(M)
      QS=UBODY(K)*UBODY(K)+VBODY(K)*VBODY(K)
      EMN=EMACH(QS)
      IF (EMN.LT..005) GO TO 50
      L=L+1
      XREAL(L)=XBODY(K)*COSA-YBODY(K)*SINA
      YREAL(L)=XBODY(K)*SINA+YBODY(K)*COSA
      QL(L)=SQRT(QS)
      MACHN(L)=EMN
      ANGL(L)=ATAN2(UBODY(K)*SINA+VBODY(K)*COSA,UBODY(K)*COSA-VBODY(K)*S
     1INA)
      IF (L.EQ.1) GO TO 50
      DX=XREAL(L)-XREAL(L-1)
      DY=YREAL(L)-YREAL(L-1)
      DIS=DIS+SQRT(DX*DX+DY*DY)
   50 CONTINUE
      KP=L
C     SCALE TOTAL ARC LENGTH OF PROFILE TO 2
      SCLE=2./DIS
      SCLE=SCALE
      CTHR=SCLE
      DO 60 L=1,KP
      XREAL(L)=XREAL(L)*SCLE
   60 YREAL(L)=YREAL(L)*SCLE
      CL=SCLE*CL
      CL=2.*SCLE*GAM
C     DEFINE QUANTITIES FOR BOUNDARY LAYER CORRECTION
      KQMIN=1
      KQMAX=1
      QMIN=MACHN(1)
      QMAX=QMIN
      H(1)=0.
      THETA(1)=0.
      S(1)=0.
      KAPPA(KP)=0.
      ROT=CMPLX(XREAL(1),-YREAL(1))
      ROT=ROT/CABS(ROT)
C     XBODY,YBODY WILL BE ROTATED TO THE HORIZONTAL
      C1=REAL(ROT)
      S1=AIMAG(ROT)
      XBODY(1)=XREAL(1)*C1-YREAL(1)*S1
      YBODY(1)=XREAL(1)*S1+YREAL(1)*C1
      DO 70 L=2,KP
      H(L)=0.
      THETA(L)=0.
      SEPR(L)=0.
      IF (MACHN(L).GT.QMAX) KQMAX=L
      IF (MACHN(L).LT.QMIN) KQMIN=L
      QMIN=AMIN1(MACHN(L),QMIN)
      QMAX=AMAX1(MACHN(L),QMAX)
      DX=XREAL(L)-XREAL(L-1)
      DY=YREAL(L)-YREAL(L-1)
```

```
      IF (ANGL(L)-ANGL(L-1).GT.PI/2.) ANGL(L)=ANGL(L)-PI
      DTH=ANGL(L)-ANGL(L-1)
      FAC=1.
      IF (DTH.NE.0.) FAC=.5*DTH/SIN(.5*DTH)
      DS=FAC*SQRT(DX*DX+DY*DY)
      S(L)=S(L-1)+DS
      KAPPA(L-1)=-DTH/DS
      XBODY(L)=XREAL(L)*C1-YREAL(L)*S1
      YBODY(L)=XREAL(L)*S1+YREAL(L)*C1
   70 CONTINUE
      LTRU=KP+2
      LTRL=-1
      IF (RN.LE.0.) GO TO 120
C     NASHMC FINDS MOMENTUM THICKNESS OF BOUNDARY LAYER ON AIRFOIL
      L=KQMAX
      IF (TRANU.LT.0.) GO TO 90
      DO 80 L=KQMIN,KP
      IF (XREAL(L).GE.TRANU) GO TO 90
   80 CONTINUE
      WRITE (N4,290)
      GO TO 120
   90 XTRANS=XREAL(L)
      LTRU=L
      CALL NASHMC (L,KP,RN)
      IF (TRANL.GT.0.) XTRANS=TRANL
      DO 100 L=1,KQMIN
      IF (XREAL(L).LE.XTRANS) GO TO 110
  100 CONTINUE
      IF (RN.NE.0.) WRITE (N4,290)
  110 LTRL=L
      CALL NASHMC (LTRL,1,RN)
  120 CONTINUE
      LC=18
      IF (NB.EQ.1) LC=11
C     COMPUTE THICKNESS-CHORD RATIO
      CALL SPLIF (KP,S,XREAL,FP,FPP,FPPP,3,0.,3,0.)
      CALL SPLIF (KP,S,YREAL,GP,GPP,GPPP,3,0.,3,0.)
      DIS=0.
      DO 130 J=2,KP
      DX=XREAL(J)-XREAL(1)
      DY=YREAL(J)-YREAL(1)
      DNEW=SQRT(DX*DX+DY*DY)
      IF (DNEW.LT.DIS) GO TO 140
  130 DIS=DNEW
  140 CHRD=DIS
      CHRD=1.
      THK=0.
      DO 160 J=1,KP
      SN=S(KP)-S(J)
      CALL INTPL (1,SN,XN,S,XREAL,FP,FPP,FPPP)
      CALL INTPL (1,SN,YN,S,YREAL,GP,GPP,GPPP)
      XN=XN-XREAL(J)
```

```
      YN=YN-YREAL(J)
      THN=SQRT(XN*XN+YN*YN)
      IF (THN-THK) 160,160,150
  150 THK=THN
  160 CONTINUE
      TC=THK/CHRD
C     COMPUTE GAP-TO-CHORD RATIO
      GAP=SCLE*(PI+PI)*SQRT(XBIMG(1)*XBIMG(1)+YBIMG(1)*YBIMG(1))
      GC=GAP/CHRD
      DX=-SCLE*(PI+PI)*((XAIMG(1)+XBIMG(1))*COSA-(YAIMG(1)+YBIMG(1))*SIN
     1A)
      DY=-SCLE*(PI+PI)*((XAIMG(1)+XBIMG(1))*SINA+(YAIMG(1)+YBIMG(1))*COS
     1A)
      DZ=CMPLX(DX,DY)
C     WRITE OUTPUT DATA ONTO TAPE 4
      WRITE (N4,250) TC,CL
      EMA=ABS(MACHA)
      IF (NB.NE.1) GO TO 170
      WRITE (N4,270) EMA
      GO TO 180
  170 EMB=ABS(MACHB)
      IF (EMB.GT.EMA) WRITE (N4,220) DF,GC
      IF (EMB.LT.EMA) WRITE (N4,230) GAP
      THA=90.-ANGLA-RAD*ANGROT
      THB=90.-ANGLB-RAD*ANGROT
      TURN=THA-THB
      WRITE (N4,240) EMB,THB,EMA,THA,TURN
  180 WRITE (N4,260) DX,DY
      WRITE (N4,320)
      WRITE (N4,330)
      SFAC=2./(S(KP)-S(1))
      DO 210 L=1,KP
      U=QL(L)*COS(ANGL(L))
      V=QL(L)*SIN(ANGL(L))
      ANG=RAD*ANGL(L)
      DELS=THETA(L)*H(L)
      XS=XREAL(L)-DELS*SIN(ANGL(L))
      YS=YREAL(L)+DELS*COS(ANGL(L))
      IF ((L.LT.LTRL).OR.(L.GT.LTRU)) GO TO 190
      MSG=1H
      IF (MACHN(L).LT.TOL) MSG=10HSTAGNATION
      IF (MACHN(L).LT.TOL) GO TO 200
      IF ((L.EQ.LTRL+1).OR.(L.EQ.LTRU-1)) MSG=10HTRANSITION
      WRITE (N4,350) XREAL(L),YREAL(L),ANG,KAPPA(L),MACHN(L),MSG,XS,YS
      GO TO 200
  190 CONTINUE
      WRITE (N4,340) XREAL(L),YREAL(L),ANG,KAPPA(L),MACHN(L),THETA(L),SE
     1PR(L),XS,YS
  200 LC=LC+1
      S(L)=SFAC*S(L)-1.
      IF (LC.LE.55) GO TO 210
      LC=4
```

```
      WRITE (N4,280)
      WRITE (N4,330)
  210 CONTINUE
      WRITE (N4,380) RATC
      S(KP)=1.
      QS=U*U+V*V
      RT=(1.+GAMM*MAVE*MAVE)/(1.+GAMM*MACHN(1)*MACHN(1))
      FAC=.25/(1.+GAMM*MACHN(1)*MACHN(1))
      HBT=FAC*(H(KP)+1.)+1.
      HBB=FAC*(H(1)+.1)+1.
      CDF=2.*RT**3*(THETA(KP)*QS**HBT+THETA(1)*QS**HBB)
      IF (ABS(MACHA).LE.ABS(MACHB)) WRITE (N4,300) CDF
      END FILE N7
      REWIND M1
      READ (M1)
      READ (M1)
      WRITE (M1) KP,EMB,EMA,TURN,CL,DX,DY,ROTATE,(UBODY(J),VBODY(J),XBOD
     1Y(J),YBODY(J),XREAL(J),YREAL(J),S(J),QL(J),MACHN(J),J=1,KP)
      WRITE (M1) (ANGL(J),J=1,KP)
      IF (NB.EQ.1) RETURN
      COSTE=COS(ANGL(1))
      COSBET=COS(ANGROT+PI*(ANGLA+ANGLB)/360.)
      FAC=(COSBET/COSTE)*(1.+.125*CDF*(HBB+HBT)**2/(GAP*COSBET))
      CLOSS=(CDF+CDF)*FAC
      IF (ABS(MACHB).GT.ABS(MACHA)) WRITE (N4,310) CLOSS
      RETURN
  220 FORMAT (34X,17HDIFFUSION FACTOR=,F5.3//39X,10HGAP/CHORD=,F6.3/)
  230 FORMAT (39X,10HGAP/CHORD=,F6.3/)
  240 FORMAT (13X,18HINLET MACH NUMBER=,F6.3,15X,17HINLET FLOW ANGLE=,F7
     1.2//13X,18H EXIT MACH NUMBER=,F6.3,15X,17H EXIT FLOW ANGLE=,F7.2//
     235X,14HTURNING ANGLE=,F7.2/)
  250 FORMAT (1H1/36X,16HTHICKNESS/CHORD=,F6.4//32X,12HCOEFFICIENT ,8HOF
     1 LIFT=,F6.4/)
  260 FORMAT (34X,3HDX=,F6.4,6X,3HDY=,F6.4/)
  270 FORMAT (40X,12HMACH NUMBER=,F6.3/)
  280 FORMAT (1H1//)
  290 FORMAT (//52H****TRANSITION NOT FOUND--BOUNDARY LAYER SKIPPED****)
  300 FORMAT (1HO,10X,27HPROFILE DRAG COEFFICIENT = ,F6.4)
  310 FORMAT (1HO,10X,27H       LOSS COEFFICIENT = ,F6.4)
  320 FORMAT (13X,45HCOORDINATES FROM LOWER SURFACE TAIL TO UPPER ,12HSU
     1RFACE TAIL/)
  330 FORMAT (12X,1HX,7X,1HY,7X,3HANG,3X,1HK,7X,1HM,5X,5HTHETA,4X,3HSEP,
     15X,2HXS,6X,2HYS/)
  340 FORMAT (F15.4,F8.4,F8.1,F6.2,F8.4,2F8.5,F7.4,F8.4)
  350 FORMAT (F15.4,F8.4,F8.1,F6.2,F8.4,3X,A10,3X,F7.4,F8.4)
  360 FORMAT (12X,52HLISTING OF POINTS ENCOUNTERED NEAR THE SONIC SURFAC
     1E,////8X,4HMODE,7X,2HXI,13X,3HETA,11X,7HROOTOLD,11X4HROOT//)
  370 FORMAT (I12,8F8.3)
  380 FORMAT (1HO,10X,27H                    RATC = ,F6.4)
      END
```

```
      SUBROUTINE BDYPTS (JF,THF,THL,KL,MP)
C     ORGANIZES SOLUTION ON REAL SUPERSONIC CHARACTERISTICS
      COMMON /1/ NP,KBM,MODE,NF,NN,MM,NI,NX,IS,KC,LBM,JJ,KK,NB,NC,MRP,NJ
     1,KF,NBPS,NFC,NRN,NPTS,NT,IPLT,LOFF,KP,NOSE,NCAS
      COMMON /2/ PI,TP,GRID,TOL
      COMPLEX XIA,XIB,XIC,DZ
      COMMON /3/ R,R1,R2,RATC,PHMN,GAM,WTAIL,XIA,XIB,XIC,UA,UB,VA,VB,UC,
     1VC,TRANU,TRANL,CONE,CTWO,CTHR,EP,RN,SO,THNOS,DZ
      COMMON /G/ N1,N3,N4,N7,M1
      COMPLEX XI,XIBODY
      COMMON SEPR(300),QL(300),XI(300),XIBODY(300),UBODY(300),VBODY(300)
     1,XBODY(300),YBODY(300),H(300),THETA(300),ANGL(300),MACHN(300),KAPP
     2A(300),XREAL(300),YREAL(300),S(300)
      COMMON FP(300),FPP(300),FPPP(300),GP(300),GPP(300),GPPP(300)
      COMMON ILINE(300),JLINK(300),OM(300)
      DATA MM/0/
      DARG=TP/FLOAT(NT-1)
      DTH=THL-THF+TOL
C     MM IS TOTAL NUMBER OF SUPERSONIC XI CHARACTERISTICS WHICH HAVE
C     BEEN READ FROM TAPE
C     MP IS TOTAL NUMBER OF POINTS IN ARRAYS UBODY, VBODY
C     KL IS NUMBER OF POINTS DEFINING AIRFOIL
C     NDT IS THE NUMBER OF POINTS ON THE ELLIPSE ACROSS SONIC LINE
      IF (MM.EQ.0) MP=KP
      NDT=INT(DTH/DARG+1.75)
      ILINE(JF)=MP+1
C     FIND BODY POINTS
      READ (N1) KK,NN
      IR=NN/50+1
      IR=(NN-1)/IR+1
      WRITE (N7) IR
      IF (N1.NE.M1) READ (M1)
C     CHECK NN ROWS FOR BODYPOINTS
      WRITE (N4,40)
C     M IS NUMBER OF BODY POINTS ON THIS SUPERSONIC ARC
      M=0
      DO 10 J=1,NN
      M=M+1
      MM=MM+1
      READ (N1) K
      IF (N1.NE.M1) READ (M1)
C     INTERPOLATE TO FIND BODY POINT
      CALL INTERP (J,K,MP+M,MX,THF)
C     IF MX=0 NO BODY POINT HAS BEEN FOUND
      IF (MX.EQ.0) M=M-1
   10 IF (M.NE.1) ILINE(MP+M)=0
      MP=MP+M
```

```
      KL=KL+M-NDT
      IF (M.NE.0) GO TO 20
      ILINE(JF)=JF+NDT
      GO TO 30
   20 ILINE(MP+1)=JF+NDT
   30 CONTINUE
C     READ IN ARGUMENT OF NEXT SUPERSONIC PATH
      READ (N1) THF,THL
      IF (N1.NE.M1) READ (M1)
      RETURN
   40 FORMAT (1H1//45H   POSITIVE AND NEGATIVE VALUES OF THE STREAM,30H
     1FUNCTION AT SUPERSONIC POINTS///)
      END


      SUBROUTINE INTERP (J1,K1,K2,K3,THF)
C     LOCATES SUPERSONIC BODY POINTS BY QUADRATIC INTERPOLATION
C     THE BODY IS THE STREAMLINE PSI = 0
C     K3 IS ZERO WHEN THERE IS NO BODY POINT IN THIS INTERVAL
      COMMON /1/ NP,KBM,MODE,NF,NN,MM,NI,NX,IS,KC,LBM,JJ,KK,NB,NC,MRP,NJ
     1,KF,NBPS,NFC,NRN,NPTS,NT,IPLT,LOFF,KP,NOSE,NCAS
      COMMON /2/ PI,TP,GRID,TOL
      COMMON /G/ N1,N3,N4,N7,M1
      COMPLEX XI,XIBODY
      COMMON SEPR(300),QL(300),XI(300),XIBODY(300),UBODY(300),VBODY(300)
     1,XBODY(300),YBODY(300),H(300),THETA(300),ANGL(300),MACHN(300),KAPP
     2A(300),XREAL(300),YREAL(300),S(300)
      COMMON FP(300),FPP(300),FPPP(300),GP(300),GPP(300),GPPP(300)
      COMMON ILINE(300),JLINK(300),OM(300)
      COMPLEX XSI,ETA
      DIMENSION XSI(100), ETA(100), U(100), V(100), PSI(100), PHI(100),
     1X(100), Y(100)
      COMPLEX XI13,XI23,CDUM
      MRQ=NN/65+1
      IR=NN/50+1
      K3=0
      XB=0.
      XA=0.
C     READ IN POINTS ON REAL SUPERSONIC CHARACTERISTIC
C     CHECK FOR A SIGN CHANGE BETWEEN THE SECOND AND THIRD INTERVAL
      DO 10 L=1,K1
      I=K1+1-L
      READ (N1) XSI(I),ETA(I),U(I),V(I),PHI(I),PSI(I),X(I),Y(I)
      IF (N1.EQ.M1) GO TO 10
C     RICHARDSON EXTRAPOLATION
      READ (M1) CDUM,CDUM,U2,V2,DUM,PSI2,X2,Y2
```

```
      U(I)=(4.*U(I)-U2)/3.
      V(I)=(4.*V(I)-V2)/3.
      PSI(I)=(4.*PSI(I)-PSI2)/3.
      X(I)=(4.*X(I)-X2)/3.
      Y(I)=(4.*Y(I)-Y2)/3.
   10 CONTINUE
      DO 30 I=1,K1
      XC=PSI(I)
      IF (I.EQ.1) GO TO 20
      IF (XB*XC.LE.0) GO TO 40
   20 XA=XB
   30 XB=XC
C     NO BODY POINT ON THIS CHARACTERISTIC
      K3=0
      GO TO 60
   40 K3=I
      L=K3
      EA=XC-XB
      EB=XC-XA
      EC=XB-XA
C     DO LINEAR INTERPOLATION BETWEEN POINTS 2 AND 3
      U23=(U(L-1)*XC-U(L)*XB)/EA
      V23=(V(L-1)*XC-V(L)*XB)/EA
      X23=(X(L-1)*XC-X(L)*XB)/EA
      Y23=(Y(L-1)*XC-Y(L)*XB)/EA
      XI23=(XSI(L-1)*XC-XSI(L)*XB)/EA
      IF (EA*EC.LE.0.) GO TO 50
      IF ((ABS(XB).LT.TOL).OR.(ABS(XC).LT.TOL)) GO TO 50
C     DO LINEAR INTERPOLATION BETWEEN POINTS 1 AND 3
      U13=(U(L-2)*XC-U(L)*XA)/EB
      V13=(V(L-2)*XC-V(L)*XA)/EB
      X13=(X(L-2)*XC-X(L)*XA)/EB
      Y13=(Y(L-2)*XC-Y(L)*XA)/EB
      XI13=(XSI(L-2)*XC-XSI(L)*XA)/EB
C     DO QUADRATIC INTERPOLATION
      UBODY(K2)=(U13*XB-U23*XA)/EC
      VBODY(K2)=(V13*XB-V23*XA)/EC
      XBODY(K2)=(X13*XB-X23*XA)/EC
      YBODY(K2)=(Y13*XB-Y23*XA)/EC
      XIBODY(K2)=(CONJG(XI13)*XB-CONJG(XI23)*XA)/EC
      GO TO 60
C     IF PSI IS NOT MONOTONIC USE LINEAR INTERPOLATION
   50 UBODY(K2)=U23
      VBODY(K2)=V23
      XBODY(K2)=X23
      YBODY(K2)=Y23
      XIBODY(K2)=CONJG(XI23)
C     CONSTRUCT GRAPH OF SIGN OF PSI ON SUPERSONIC CHARACTERISTICS
   60 NCOUNT=0
      DO 70 L=1,K1,MRQ
      J=K1+1-L
      NCOUNT=NCOUNT+1
```

```
      N=1HP
      IF (PSI(J).LT.0.) N=1HN
   70 JLINK(NCOUNT)=N
      WRITE (N4,90) (JLINK(K),K=1,NCOUNT)
      IF (MOD(J1-1,IR).NE.0) GO TO 80
C     WRITE DATA ON TAPE 7 FOR USE IN PLTCHR
      KL=(K3-2)/IR+2
      IF (K3.EQ.0) KL=0
      WRITE (N7) KL
      IF (KL.EQ.0) GO TO 80
      K3L=K3-1
      WRITE (N7) (X(J),Y(J),J=1,K3L,IR)
      WRITE (N7) XBODY(K2),YBODY(K2)
   80 CONTINUE
      RETURN
   90 FORMAT (2X,65(1X,A1))
      END


      SUBROUTINE NASHMC (K1,K2,RN)
C     COMPUTES THE BOUNDARY LAYER CORRECTION
      REAL MACHN,KAPPA
      COMPLEX XI,XIBODY
      COMMON SEPR(300),QL(300),XI(300),XIBODY(300),UBODY(300),VBODY(300)
     1,XBODY(300),YBODY(300),H(300),THETA(300),ANGL(300),MACHN(300),KAPP
     2A(300),XREAL(300),YREAL(300),S(300)
      REAL MACH
      COMMON /A/ GAMMA,MACH,RHOINF,EMU,QSTRSQ
      REAL MACHA,MACHB
      COMMON /7/ MACHA,MACHB,ANGLA,ANGLB
      REAL MH,MHSQ,NU,MAVE
      DATA TR/.3424/,RTHD/320./,TE1/5.E-3/,TE2/5.E-5/,PIMIN/-1.5/,PIMAX/
     11.E4/
      GAMM=(GAMMA-1.)/2.
      GAM1=1./(GAMMA-1.)
      MAVE=.5*(ABS(MACHA)+ABS(MACHB))
      CSIINF=1.+GAMM*MAVE*MAVE
      INC=ISIGN(1,K2-K1)
      GE=6.5
      L=K1
      DSOLD=ABS(S(L)-S(L-INC))
   10 LP=L+INC
      MH=.5*(MACHN(L)+MACHN(LP))
      MHSQ=MH*MH
      CSIH=1.+GAMM*MHSQ
      DS=ABS(S(LP)-S(L))
```

```
      IF (LP.NE.K2) DQDS=(MACHN(LP)-MACHN(L))/(DS*MH*CSIH)
      T=CSIINF/CSIH
      RHOH=T**GAM1
      NU=T*(1.+TR)/(RHOH*(T+TR))
      RTH=RN*MH/(MAVE*NU)
      IF (L.NE.K1) GO TO 20
      THETAH=RTHO/RTH
      THT=THETAH
      THETA(L)=THETAH
   20 FC=1.0+.066*MHSQ-.008*MH*MHSQ
      FR=1.-.134*MHSQ+.027*MHSQ*MH
C     DO AT MOST 200 ITERATIONS
      DO 40 J=1,200
      RTAU=1./(FC*(2.4711*ALOG(FR*RTH*THETAH)+4.75)+1.5*GE+1724./(GE*GE+
     1200.)-16.87)
      TAU=RTAU*RTAU
      HB=1./(1.-GE*RTAU)
      HH=(HB+1.)*(1.+.178*MHSQ)-1.
      SEP=-THETAH*DQDS
      PIE=HH*SEP/TAU
      PIE=AMAX1(PIMIN,AMIN1(PIMAX,PIE))
      G=6.1*SQRT(PIE+1.81)-1.7
      T2=ABS(G-GE)/GE
      GE=G
      DT2=DT
      DT=(HH+2.-MHSQ)*SEP+TAU
      IF (J.EQ.1) GO TO 30
      TI=ABS((DT-DT2)/DT)
      IF ((TI.LT.TE2).AND.(T2.LT.TE1)) GO TO 50
   30 THETAH=THT+.5*DT*DS
   40 CONTINUE
   50 THETA(LP)=DT*DS+THT
      THETAH=THETA(LP)
      THT=THETA(LP)
      H(L)=(H(L)*DS+HH*DSOLD)/(DS+DSOLD)
      H(LP)=HH
      SEP=-THETAH*DQDS
      SEPR(L)=(SEPR(L)*DS+SEP*DSOLD)/(DS+DSOLD)
      SEPR(LP)=SEP
      DSOLD=DS
      L=LP
      IF (L.NE.K2) GO TO 10
      H(K2)=2.*H(K2)-H(K2-INC)
      H(K1)=0.
      SEPR(K2)=SEPR(K2)+(SEPR(K2)-SEPR(K2-INC))
      RETURN
      END
```

```
      SUBROUTINE LAMBDA (UL,VL,LAMDAP,LAMDAM,TAU,XI,ETA,IND)
C     COMPUTES COEFFICIENTS OF DIFFERENCE EQUATIONS GIVEN UL AND
C     VL ON SUBSONIC AND TRANSONIC GRIDS
      EXTERNAL CSQRT
      COMPLEX UL,VL,LAMDAP,LAMDAM,TAU,U,V,QS,CS,CLOG,CEXP,XI,ETA
      COMPLEX CSQRT,ROOTL
      COMMON /1/ NP,KBM,MODE,NF,NN,MM,NI,NX,IS,KC,LBM,JJ,KK,NB,NC,MRP,NJ
     1,KF,NBPS,NFC,NRN,NPTS,NT,IPLT,LOFF,KP,NOSE,NCAS
      COMPLEX ROOT,ROOT1
      COMMON /5/ ROOT,ROOT1
      REAL MACH
      COMMON /A/ GAMMA,MACH,RHOINF,EMU,QSTRSQ
      COMMON /G/ N1,N3,N4,N7,M1
      DATA EMOLD/0./,ROOT/(0.,1.)/
      IF (MACH.EQ.EMOLD) GO TO 10
      EMOLD=MACH
      GAMM=(GAMMA-1.)/2.
      COSQ=1./(MACH*MACH)+GAMM
      GAMI=-1./(GAMMA-1.)
      RHOINF=(MACH*MACH)**GAMI
      GAMI=GAMMA*GAMI
   10 U=UL
      V=VL
      QS=U*U+V*V
      CS=COSQ-GAMM*QS
      ROOTL=ROOT
      ROOT=CSQRT(QS/CS-1.,ROOT)
C     CHECK MAGNITUDE AND BRANCH OF ROOT
      IF ((IS.NE.1).OR.(IND.LT.0)) GO TO 20
      IF (CABS(ROOT).LE..2) WRITE (N7) MODE,XI,ETA,ROOTL,ROOT
   20 IF (REAL(ROOT).GE.-AIMAG(ROOT)) GO TO 30
      WRITE (N4,40) XI,ETA
      ROOT=-ROOT
   30 LAMDAP=(U*V+CS*ROOT)/(CS-V*V)
      LAMDAM=(U*V-CS*ROOT)/(CS-V*V)
      IF (IND.LT.0) RETURN
      TAU=CS*RHOINF*CEXP(GAMI*CLOG(CS))*ROOT
      RETURN
   40 FORMAT (//5X,32HCORRECTION OF THE BRANCH AT XI =F7.3,1H,,F7.3,8H
     1 ETA =F7.3,1H,,F7.3)
      END
```

```
      SUBROUTINE LAMBD (WL,WSL,LAM,TAU,XI,ETA,IND)
C     COMPUTES COEFFICIENTS OF DIFFERENCE EQUATIONS FROM WL AND WSL
C     FOR SUPERSONIC GRIDS
      COMMON /1/ NP,KBM,MODE,NF,NN,MM,NI,NX,IS,KC,LBM,JJ,KK,NB,NC,MRP,NJ
     1,KF,NBPS,NFC,NRN,NPTS,NT,IPLT,LOFF,KP,NOSE,NCAS
      COMPLEX ROOT,ROOT1
      COMMON /5/ ROOT,ROOT1
      REAL MACH
      COMMON /A/ GAMMA,MACH,RHOINF,EMU,QSTRSQ
      COMMON /G/ N1,N3,N4,N7,M1
      COMPLEX WL,WSL,LAM,TAU,W,WS,CS,QS,CSQRT,CLOG,CEXP,I,XI,ETA,ROOTL
      EXTERNAL CSQRT
      DATA EMOLD/0./,I/(0.,1.)/
      IF (MACH.EQ.EMOLD) GO TO 10
      EMOLD=MACH
      GAM=(GAMMA-1.)/2.
      COSQ=1./(MACH*MACH)+GAM
      GAMI=-1./(GAMMA-1.)
      RHOINF=(MACH*MACH)**GAMI
      GAMI=GAMMA*GAMI
   10 W=WL
      WS=WSL
      QS=W*WS
      CS=COSQ-GAM*QS
      ROOTL=ROOT
      ROOT=CSQRT(QS/CS-1.,ROOT)
C     CHECK MAGNITUDE AND BRANCH OF ROOT
      IF ((IS.NE.1).OR.(IND.LT.0)) GO TO 20
      IF (CABS(ROOT).LE..2) WRITE (N7) MODE,XI,ETA,ROOTL,ROOT
   20 IF (REAL(ROOT).GE.-AIMAG(ROOT)) GO TO 30
      WRITE (N4,40) XI,ETA
      ROOT=-ROOT
   30 LAM=2.*CS-QS-2.*I*CS*ROOT
      IF (IND.LT.0) RETURN
      TAU=RHOINF*CEXP(GAMI*CLOG(CS))*CS*ROOT
      RETURN
   40 FORMAT (//5X,32HCORRECTION OF THE BRANCH AT XI =F7.3,1H,,F7.3,8H
     1 ETA =F7.3,1H,,F7.3)
      END


      COMPLEX FUNCTION DTAUDX(UL,VL)
C     FINDS THE DERIVATIVE OF TAU AS A FUNCTION OF XI
      COMPLEX XIA,XIB,XIC,DZ
      COMMON /3/ R,R1,R2,RATC,PHMN,GAM,WTAIL,XIA,XIB,XIC,UA,UB,VA,VB,UC,
     1VC,TRANU,TRANL,CONE,CTWO,CTHR,EP,RN,SO,THNOS,DZ
      REAL MACH
      COMMON /A/ GAMMA,MACH,RHOINF,EMU,QSTRSQ
      COMPLEX UL,VL,S,CS,QS,CLOG,CEXP,DSDXI,CONST
      DATA EMOLD/0./
      IF (MACH.EQ.EMOLD) GO TO 10
```

```
      EMOLD=MACH
      GAMM=(GAMMA-1.)/2.
      COSQ=1./(MACH*MACH)+GAMM
      DSDXI=1.
      CALL STOFXI (XIA,S,DSDXI)
      GAMI=-1./(GAMMA-1.)
      RHOINF=(MACH*MACH)**GAMI
      GAMI=GAMMA*GAMI
      CONST=CMPLX(0.,-.25*(GAMMA+1.)*RHOINF)
   10 QS=UL*UL+VL*VL
      CS=COSQ-GAMM*QS
      DTAUDX=CONST*CEXP(GAMI*CLOG(CS))*QS*QS*DSDXI/(S*(QS-CS))
      RETURN
      END


      SUBROUTINE GETUV (S,T,U,V)
C     FINDS U AND V FROM HODOGRAPH COORDINATES S AND T
      COMMON /2/ PI,TP,GRID,TOL
      COMMON /G/ N1,N3,N4,N7,M1
      COMPLEX I,S,T,U,V,W,WSTR,HSQ,HSQPR,QS,QSNEW,ST,CSQRT
      EXTERNAL CSQRT
      DATA I/(0.,1.)/
      QS=U*U+V*V
      W=S
      WSTR=T
      ST=S*T
      IF (CABS(ST).GE.TOL) GO TO 10
      CALL GETHSQ ((0.,0.),HSQ,HSQPR)
      W=W/SQRT(REAL(HSQPR))
      WSTR=WSTR/SQRT(REAL(HSQPR))
      GO TO 50
C     DO AT MOST 20 NEWTON ITERATIONS
   10 DO 20 L=1,20
      CALL GETHSQ (QS,HSQ,HSQPR)
      QSNEW=QS-(HSQ-ST)/HSQPR
      IF (CABS(QSNEW-QS).LT.TOL) GO TO 30
   20 QS=QSNEW
      WRITE (N4,60) ST
      QSNEW=U*U+V*V
      QS=QSNEW
   30 IF (CABS(ST).LT..1) GO TO 40
      QS=QSNEW
      W=CSQRT(QS*(S/T),W)
      WSTR=W*(T/S)
      GO TO 50
```

```
   40 W=S*CSQRT(QS/HSQ,(1.,0.))
      WSTR=T*CSQRT(QS/HSQ,(1.,0.))
   50 U=.5*(W+WSTR)
      V=.5*I*(W-WSTR)
      RETURN
   60 FORMAT (42H NEWTON ITERATION DID NOT CONVERGE AT ST =,2E12.4)
      END


      SUBROUTINE GETHSQ (QS,HSQ,HSQPR)
C     COMPUTES HODOGRAPH QUANTITY H(QS)**2 AND ITS DERIVATIVE
C     FROM THE SPEED QS
      COMPLEX ROOT,ROOT1
      COMMON /5/ ROOT,ROOT1
      REAL MACH
      COMMON /A/ GAMMA,MACH,RHOINF,EMU,QSTRSQ
      COMPLEX QS,BPHI1,BPHI2,ROOT1,TEMP,HSQ,HSQPR,ROOT2,CSQRT,CS
      COMPLEX QSX
      EXTERNAL CSQRT
      DATA ROOT1/(1.,0.)/
      DATA EMOLD/0./
      IF (MACH.EQ.EMOLD) GO TO 10
      CONST=1.
      GAMM=(GAMMA-1.)/2.
      COSQ=1./(MACH*MACH)+GAMM
      QSTRSQ=(2./(GAMMA+1.))*COSQ
      EMU=SQRT((GAMMA-1.)/(GAMMA+1.))
      FAC=EMU/QSTRSQ
      QSX=QS
      QS=(1.,0.)
   10 CS=COSQ-GAMM*QS
      BPHI1=(CS+CS)/QSTRSQ-1.
      BPHI2=(CS+CS)-QS
      ROOT2=CSQRT((CS+CS)*(BPHI2-QS),ROOT1)
      ROOT1=FAC*ROOT2
      TEMP=CEXP(CLOG(BPHI1-ROOT1)/EMU)
      TEMP=CONST/(TEMP*(BPHI2+ROOT2))
C     TEMP IS H**2/QS
      HSQ=QS*TEMP
      IF (MACH.EQ.EMOLD) GO TO 20
      EMOLD=MACH
C     SET CONST SO THAT H(1) = 1
      CONST=1./REAL(HSQ)
      QS=QSX
      GO TO 10
C     COMPUTE THE DERIVATIVE OF HSQ WITH RESPECT TO QS
```

```
   20 HSQPR=TEMP*ROOT2/(CS+CS)
      RETURN
      END


      SUBROUTINE STOFXI (X,SOFXI,SOFXIP)
C     CALCULATES HODOGRAPH VARIABLE S FROM POINT XI IN ELLIPSE
C     S = ((XI+XO)/(RATC*XI+XO))*EXP(FN(XI))
      COMMON /1/ NP,KBM,MODE,NF,NN,MM,NI,NX,IS,KC,LBM,JJ,KK,NB,NC,MRP,NJ
     1,KF,NBPS,NFC,NRN,NPTS,NT,IPLT,LOFF,KP,NOSE,NCAS
      COMMON /2/ PI,TP,GRID,TOL
      COMPLEX XIA,XIB,XIC,DZ
      COMMON /3/ R,R1,R2,RATC,PHMN,GAM,WTAIL,XIA,XIB,XIC,UA,UB,VA,VB,UC,
     1VC,TRANU,TRANL,CONE,CTWO,CTHR,EP,RN,SO,THNOS,DZ
      COMPLEX BP,ETJ
      COMMON /8/ BP(65),ETJ(65)
      COMPLEX SIG
      COMPLEX XO
      EXTERNAL SIG
      COMPLEX F11,F22,S,SI,X,TEMP,SOFXI,SOFXIP,ROT,F,FP
      COMPLEX XOLD
      XO=-CMPLX(R1*COS(THNOS),R2*SIN(THNOS))
      XOLD=X
      L=1
C     EVALUATE FN(XI) AND ITS DERIVATIVE FP
   10 S=SIG(X)/R
      SI=1./(S*R*R)
      F11=BP(NBPS)*S+BP(NBPS-1)
      F22=BP(NBPS)*SI+BP(NBPS-1)
      J=NBPS-2
   20 F11=S*F11+BP(J)
      F22=SI*F22+BP(J)
      J=J-1
      IF (J.GE.1) GO TO 20
      F=.5*(F11+F22)
      GO TO (30,40,50), L
   30 S=CEXP(F)
      TEMP=1./(RATC*X+XO)
      S=(X+XO)*TEMP*S
      SOFXI=S
      IF (REAL(SOFXIP).EQ.0.) RETURN
C     COMPUTE DS/DXI
      TEMP=1./(X+XO)-RATC*TEMP
C     FIND D/DXI(F)
      ROT=CMPLX(0.,.01)
      X=X+ROT
```

```
      L=2
      GO TO 10
   40 FP=F
      X=X-2.*ROT
      L=3
      GO TO 10
   50 FP=(FP-F)/(2.*ROT)
      SOFXIP=(FP+TEMP)*SOFXI
      X=XOLD
      RETURN
      END


      COMPLEX FUNCTION SFROMX(X)
C     CALLS SUBROUTINE STOFXI
      COMPLEX X,SOFXI,SOFXIP
      SOFXIP=0.
      CALL STOFXI (X,SOFXI,SOFXIP)
      SFROMX=SOFXI
      RETURN
      END


      COMPLEX FUNCTION XIOFS(S,XIPAST)
C     FINDS ELLIPSE COORDINATE XI FROM HODOGRAPH COORDINATES
      COMPLEX XIPAST,XIOLD,XINEW,SFROMX,SOFXI,SOFXIP,S
      COMMON /1/ NP,KBM,MODE,NF,NN,MM,NI,NX,IS,KC,LBM,JJ,KK,NB,NC,MRP,NJ
     1,KF,NBPS,NFC,NRN,NPTS,NT,IPLT,LOFF,KP,NOSE,NCAS
      COMMON /2/ PI,TP,GRID,TOL
      COMMON /G/ N1,N3,N4,N7,M1
      XIOLD=XIPAST
C     DO AT MOST 20 NEWTON ITERATIONS
      DO 10 K=1,20
      SOFXIP=1.
      CALL STOFXI (XIOLD,SOFXI,SOFXIP)
      XINEW=XIOLD-(SOFXI-S)/SOFXIP
      IF (CABS(XINEW-XIOLD).LE.TOL) GO TO 20
      XIOLD=XINEW
   10 CONTINUE
      XINEW=SFROMX(XIOLD)-S
      WRITE (N4,30) S,XIOLD,XINEW
      XINEW=XIPAST
   20 XIOFS=XINEW
      RETURN
   30 FORMAT (/* NO CONVERGENCE AT S=*,2F6.3,5X3HXI=,2F6.3,6X,*S(XI)-S=*
     1,2F6.3/* TROUBLE CALCULATING SONIC LINE. CHANGE MACH *)
      END
```

```
      SUBROUTINE RING (NK,TS1,TS2,CC,R)
C     CALCULATES CHEBYSHEV COEFFICIENTS OF F IF REAL(F)=TS1
C     AROUND THE ELLIPSE AND FINDS IMAG(F)=TS2
      COMPLEX CC,I
      COMMON /2/ PI,TP,GRID,TOL
      DIMENSION TS1(1), TS2(1), CC(1), AX(130), BX(130), ARG(130)
      DATA I/(.0,1.)/
      DARG=TP/FLOAT(NK)
      NK2=NK/2
      NFCS=NK2+1
C     FIND CHEBYSHEV COEFFICIENTS GIVEN REAL PART OF FUNCTION
      DO 10 J=1,NK
      K=NK2+J
      IF (K.GT.NK) K=K-NK
   10 TS2(J)=TS1(K)
      CALL FOUCF (NK,TS2,CC,AX,BX)
      DO 20 J=1,NFCS
      E1=1.
      E2=EXP(FLOAT(2-2*J)*ALOG(R))
      C1=.5*(E1+E2)
      C2=.5*(E1-E2)
      T1=REAL(CC(J))/C1
      T2=AIMAG(CC(J))/C2
   20 CC(J)=CMPLX(T1,T2)
      CC(1)=CMPLX(REAL(CC(1)),0.)
C     FILTER THE CHEBYSHEV COEFFICIENTS
      FAC=TP/FLOAT(NK)
      DO 30 J=2,NFCS
      FAC1=SIN(FLOAT(J-1)*FAC)/(FLOAT(J-1)*FAC)
   30 CC(J)=FAC1*CC(J)
C     FIND CONJUGATE FUNCTION
      DO 50 J=1,NK
      L=NK2+J
      IF (L.GT.NK) L=L-NK
      ARG(J)=FLOAT(J-1)*DARG
      AX(J)=0.
      DO 40 K=1,NFCS
      E1=1.
      E2=EXP(FLOAT(2-2*K)*ALOG(R))
      C1=.5*(E1+E2)
      C2=.5*(E1-E2)
      T1=REAL(CC(K))*C2
      T2=AIMAG(CC(K))*C1
      TRM=T1*SIN((K-1)*ARG(J))+T2*COS((K-1)*ARG(J))
      AX(J)=AX(J)+TRM
   40 TS2(L)=AX(J)
   50 CONTINUE
```

```
      RETURN
      END


      COMPLEX FUNCTION SIG(X)
C     CALCULATES POINT SIG IN RING CORRESPONDING TO POINT X IN ELLIPSE
      COMPLEX X,I
      DATA I/(0.,1.)/
      RR=REAL(X)
      C=AIMAG(X)
      TOL=1.E-5
      IF ((ABS(C).GE.TOL).OR.(ABS(RR).GT.1.)) GO TO 10
      RAD=1.
      THT=ACOS(RR)
      GO TO 20
   10 CONTINUE
      T=RR*RR+C*C+1.
      ASQ=.5*(T+SQRT(T*T-4.*RR*RR))
      ASQ=AMAX1(ASQ,1.)
      B=SQRT(ASQ-1.)
      A=SQRT(ASQ)
      RAD=A+B
      FAC=AMAX1(ABS(RR/A),1.)
      VAR=RR/(A*FAC)
      THT=ACOS(VAR)
   20 CONTINUE
      IF (C.LT.0) THT=-THT
      SIG=CEXP(I*THT)*RAD
      RETURN
      END


      FUNCTION RHO (QS)
C     COMPUTES DENSITY AS A FUNCTION OF THE SQUARE OF THE SPEED
      REAL MACH
      COMMON /A/ GAMMA,MACH,RHOINF,EMU,QSTRSQ
      DATA EMOLD/0./
      IF (MACH.EQ.EMOLD) GO TO 10
      EMOLD=MACH
      GAMM=(GAMMA-1.)/2.
      COSQ=1./(MACH*MACH)+GAMM
      GAMI=-1./(GAMMA-1.)
      RHOINF=(MACH*MACH)**GAMI
   10 CS=COSQ-GAMM*QS
      RHO=CS**(-GAMI)/RHOINF
      RETURN
      END
```

```
      COMPLEX FUNCTION CSQRT(Z,BRANCH)
C     COMPUTES THE COMPLEX SQUARE ROOT OF Z WHOSE BRANCH CUT IS
C     A STRAIGHT LINE FROM THE ORIGIN PASSING THROUGH -BRANCH
      DIMENSION Z(2), BRANCH(2)
      REAL IMAGZ
      REALZ=Z(1)
      IMAGZ=Z(2)
      R=REALZ*REALZ+IMAGZ*IMAGZ
      Q=0.
      IF (R.EQ.0.) GO TO 40
      Q=SQRT(.5*(SQRT(R)+ABS(REALZ)))
      IF (REALZ.GE.0.) GO TO 10
      R=Q
      Q=.5*IMAGZ/R
      GO TO 20
   10 R=.5*IMAGZ/Q
   20 IF (Q*BRANCH(1)+R*BRANCH(2)) 30,40,40
   30 CSQRT=CMPLX(-Q,-R)
      RETURN
   40 CSQRT=CMPLX(Q,R)
      RETURN
      END


      SUBROUTINE LEQ (A,B,NEQS,NSOLNS,IA,IB,DET)
C     SOLVES A SYSTEM OF LINEAR EQUATIONS BY GAUSS ELIMINATION
      DIMENSION A(IA,IA), B(IB,IB)
      NSIZ=NEQS
      NBSIZ=NSOLNS
      DET=1.0
      DO 40 I=1,NSIZ
      BIG=A(I,1)
      IF (NSIZ.LE.1) GO TO 130
      DO 10 J=2,NSIZ
      IF (ABS(BIG).GE.ABS(A(I,J))) GO TO 10
      BIG=A(I,J)
   10 CONTINUE
      BG=1.0/BIG
      DO 20 J=1,NSIZ
   20 A(I,J)=A(I,J)*BG
      DO 30 J=1,NBSIZ
   30 B(I,J)=B(I,J)*BG
      DET=DET*BIG
   40 CONTINUE
      NUMSYS=NSIZ-1
      DO 120 I=1,NUMSYS
```

```
      NN=I+1
      BIG=A(I,I)
      NBGRW=I
      DO 50 J=NN,NSIZ
      IF (ABS(BIG).GE.ABS(A(J,I))) GO TO 50
      BIG=A(J,I)
      NBGRW=J
   50 CONTINUE
      BG=1.0/BIG
      IF (NBGRW.EQ.I) GO TO 80
      DO 60 J=I,NSIZ
      TEMP=A(NBGRW,J)
      A(NBGRW,J)=A(I,J)
   60 A(I,J)=TEMP
      DET=-DET
      DO 70 J=1,NBSIZ
      TEMP=B(NBGRW,J)
      B(NBGRW,J)=B(I,J)
   70 B(I,J)=TEMP
   80 DO 110 K=NN,NSIZ
      PMULT=-A(K,I)*BG
      IF (PMULT.EQ.0.) GO TO 110
      DO 90 J=NN,NSIZ
   90 A(K,J)=PMULT*A(I,J)+A(K,J)
      DO 100 L=1,NBSIZ
  100 B(K,L)=PMULT*B(I,L)+B(K,L)
  110 CONTINUE
  120 CONTINUE
  130 DO 170 NCOLB=1,NBSIZ
      DO 160 I=1,NSIZ
      NROW=NSIZ+1-I
      TEMP=0.0
      NXS=NSIZ-NROW
      IF (NXS.EQ.0) GO TO 150
      DO 140 K=1,NXS
      KK=NSIZ+1-K
  140 TEMP=TEMP+B(KK,NCOLB)*A(NROW,KK)
  150 B(NROW,NCOLB)=(B(NROW,NCOLB)-TEMP)/A(NROW,NROW)
  160 CONTINUE
  170 CONTINUE
      DO 180 I=1,NSIZ
  180 DET=DET*A(I,I)
      RETURN
      END
```

```
      SUBROUTINE SPLIF (NN,S,F,FP,FPP,FPPP,KM,VM,KN,VN)
C     GIVEN S AND F AT N CORRESPONDING POINTS COMPUTES A CUBIC SPLINE
C     THROUGH THESE POINTS SATISFYING AN END CONDITION IMPOSED ON
C     EITHER END.  FP,FPP,FPPP WILL BE THE FIRST,SECOND AND THIRD
C     DERIVATIVE RESPECTIVELY AT EACH POINT ON THE SPLINE
      DIMENSION NN(2), F(2), FP(2), FPP(2), FPPP(2), S(2)
      INDX(JJ)=JJ
      N=IABS(NN(1))
      NORM=1
      IF (NN(1).LT.0) NORM=0
      I=INDX(1)
      J=INDX(2)
      JJ=2
      DS=S(J)-S(I)
      D=DS
      IF (DS.EQ.0.) GO TO 120
      DF=(F(J)-F(I))/DS
      IF (IABS(KM)-2) 10,20,30
   10 U=.5
      V=3.*(DF-VM)/DS
      GO TO 50
   20 U=0.
      V=VM
      GO TO 50
   30 U=-1.
      V=-DS*VM
      GO TO 50
   40 I=J
      JJ=JJ+1
      J=INDX(JJ)
      DS=S(J)-S(I)
      IF (D*DS.LE.0.) GO TO 120
      DF=(F(J)-F(I))/DS
      B=1./(DS+DS+U)
      U=B*DS
      V=B*(6.*DF-V)
   50 FP(I)=U
      FPP(I)=V
      U=(2.-U)*DS
      V=6.*DF+DS*V
      IF (JJ.LT.N) GO TO 40
      IF (KN-2) 60,70,80
   60 V=(6.*VN-V)/U
      GO TO 90
   70 V=VN
      GO TO 90
   80 V=(DS*VN+FPP(I))/(1.+FP(I))
   90 B=V
      D=DS
  100 DS=S(J)-S(I)
      U=FPP(I)-FP(I)*V
      FPPP(I)=(V-U)/DS
```

```
      FPP(I)=U
      FP(I)=(F(J)-F(I))/DS-DS*(V+U+U)/6.
      V=U
      J=I
      JJ=JJ-1
      I=INDX(JJ-1)
      IF (JJ.GT.1) GO TO 100
      N=INDX(N)
      FPPP(N)=FPPP(N-1)
      FPP(N)=B
      FP(N)=DF+D*(FPP(N-1)+B+B)/6.
      IF (KM.GT.0) RETURN
      N=IABS(NN(1))
      FPPP(J)=0.
      V=FPP(J)
  110 I=J
      JJ=JJ+1
      J=INDX(JJ)
      DS=S(J)-S(I)
      U=FPP(J)
      FPPP(J)=FPPP(I)+.5*DS*(F(I)+F(J)-DS*DS*(U+V)/12.)
      V=U
      IF (JJ.LT.N) GO TO 110
      RETURN
  120 PRINT 130
      PRINT 140
      STOP
  130 FORMAT (47H0**** NON-MONOTONIC ABCISSA FOR SPLINE FIT ****)
  140 FORMAT (44H0*** PROGRAM STOPPED IN SUBROUTINE SPLIF ***)
      END


      SUBROUTINE PSPLIF (M,S,F,FP,FPP,FPPP,FINT)
C     PERIODIC SPLINE FITTING ROUTINE
      DIMENSION S(1), F(1), FP(1), FPP(1), FPPP(1), FINT(1)
      CON=0.
      IF (M.LT.0) CON=FINT(1)
      N=IABS(M)
      K=1
      I=1
      J=2
      DS=S(2)-S(1)
      D=DS
      IF (DS.EQ.0.) GO TO 50
      DF=6.*(F(2)-F(1))/DS
      U1=.5
```

```
      V1=DF/(DS+DS)
      U2=.5
      V2=(DF-6.)/(DS+DS)
      GO TO 20
   10 I=J
      J=J+K
      DS=S(J)-S(I)
      IF (D*DS.LE.0.) GO TO 50
      DF=6.*(F(J)-F(I))/DS
      B=1./(DS+DS+U1)
      U1=B*DS
      V1=B*(DF-V1)
      B=1./(DS+DS+U2)
      U2=B*DS
      V2=B*(DF-V2)
   20 FP(I)=U1
      FPP(I)=V1
      FPPP(I)=U2
      FINT(I)=V2
      U1=(2.-U1)*DS
      V1=DF+DS*V1
      U2=(2.-U2)*DS
      V2=DF+DS*V2
      IF (J.NE.N) GO TO 10
      V1=-V1/U1
      V2=(6.-V2)/U2
      B1=V1
      B2=V2
   30 DS=S(J)-S(I)
      DF=(F(J)-F(I))/DS
      U1=FPP(I)-FP(I)*V1
      FPP(I)=U1
      FP(I)=DF-DS*(V1+U1+U1)/6.
      V1=U1
      U2=FINT(I)-FPPP(I)*V2
      FINT(I)=U2
      FPPP(I)=DF-DS*(V2+U2+U2)/6.
      V2=U2
      J=I
      I=I-K
      IF (J.NE.1) GO TO 30
      X=-(FPP(1)-B1)/((FINT(1)-B2)-(FPP(1)-B1))
      FP(1)=X
      U=FPP(1)+X*(FINT(1)-FPP(1))
      FPP(1)=U
      FPP(N)=FPP(1)
      FINT(N)=FPP(N)
      FP(N)=FP(1)
      FPPP(N)=FP(N)
      FINT(1)=CON
   40 I=J
      J=J+K
```

```
      DS=S(J)-S(I)
      FP(J)=FP(J)+X*(FPPP(J)-FP(J))
      V=FPP(J)+X*(FINT(J)-FPP(J))
      FPP(J)=V
      FPPP(I)=(V-U)/DS
      FINT(J)=FINT(I)+.5*DS*(F(I)+F(J)-DS*DS*(U+V)/12.)
      U=V
      IF (J.NE.N) GO TO 40
      FPPP(N)=FPPP(1)
      RETURN
   50 PRINT 60
      PRINT 70
      STOP
   60 FORMAT (47HO**** NON-MONOTONIC ABCISSA FOR SPLINE FIT ****)
   70 FORMAT (45HO*** PROGRAM STOPPED IN SUBROUTINE PSPLIF ***)
      END


      SUBROUTINE INTPL (NX,SI,FI,S,F,FP,FPP,FPPP)
C     GIVEN S,F(S) AND THE FIRST THREE DERIVATIVES AT A SET OF POINTS
C     FIND FI(SI) AT THE NX VALUES OF SI BY EVALUATING THE TAYLOR SERIES
C     OBTAINED BY USING THE FIRST THREE DERIVATIVES
      DIMENSION SI(1), FI(1), S(1), F(1), FP(1), FPP(1), FPPP(1)
      DATA PT/.33333333333333/
      J=0
      DO 30 I=1,NX
      VAL=0.
      SS=SI(I)
   10 J=J+1
      TT=S(J)-SS
      IF (TT) 10,30,20
   20 J=MAXO(1,J-1)
      SS=SS-S(J)
      VAL=SS*(FP(J)+.5*SS*(FPP(J)+SS*PT*FPPP(J)))
   30 FI(I)=F(J)+VAL
      RETURN
      END
```

```
      SUBROUTINE INTPLI (MX,XI,FI,N,X,F,FP,FPP,FPPP)
C     INVERSE INTERPOLATION ROUTINE
C     XI(L) WILL SATISFY F(XI(L)) = FI(L)  FOR L = 1 TO ABS(MX)
C     F,FP,FPP,FPPP ARE THE FUNCTION AND DERIVATIVES AT THE X POINTS
      DIMENSION X(1), F(1), FP(1), FPP(1), FPPP(1), XI(1), FI(1)
      REAL NEW,LEFT
      DATA TOL/1.E-9/
      NX=IABS(MX)
      K=2
      L1=1
      NEW=X(1)
      FN=F(1)
      FVAL=FI(1)
      IF (ABS(F(1)-FI(1)).GT.TOL) GO TO 10
      L1=2
      XI(1)=X(1)
      IF (NX.EQ.1) RETURN
   10 DO 130 L=L1,NX
      IF ((FVAL.NE.FI(L)).OR.(L.EQ.1)) GO TO 20
      NEW=X(K)
      FN=F(K)
      IF (FP(K)*FP(K-1).GT.0.) GO TO 20
      ROOT=SQRT(FPP(K-1)**2-2.*FP(K-1)*FPPP(K-1))
      DX=-2.*FP(K-1)/(FPP(K-1)+SIGN(ROOT,FPP(K-1)))
      NEW=X(K-1)+DX
      FN=F(K-1)+DX*(FP(K-1)+DX*(.5*FPP(K-1)+DX*FPPP(K-1)/6.))
   20 FVAL=FI(L)
      SGN=F(K-1)-FVAL
      IF (NEW.GT.X(K-1)) SGN=FN-FVAL
      DO 40 J=K,N
      IF (FP(J)*FP(J-1).LE.0.) GO TO 30
      IF (SGN*(F(J)-FVAL).LE.0.) GO TO 60
      GO TO 40
   30 ROOT=SQRT(FPP(J-1)**2-2.*FP(J-1)*FPPP(J-1))
      DX=-2.*FP(J-1)/(FPP(J-1)+SIGN(ROOT,FPP(J-1)))
      RIGHT=X(J-1)+DX
      LEFT=AMAX1(X(J-1),NEW+TOL)
      IF (LEFT.GT.RIGHT) GO TO 40
      F2=.5*FPP(J-1)
      F3=FPPP(J-1)/6.
      FN=F(J-1)+DX*(FP(J-1)+DX*(F2+DX*F3))
      IF (SGN*(FN-FVAL).LE.0) GO TO 90
   40 CONTINUE
      IF (MX.GT.0) GO TO 50
      MX=L-1
      RETURN
   50 PRINT 140, L,FI(L)
      J=K
      GO TO 130
   60 OLD=AMAX1(X(J-1),NEW+TOL)
      F2=.5*FPP(J-1)
      F3=FPPP(J-1)/6.
```

```
      START=OLD
      DO 70 K=1,10
      DX=OLD-X(J-1)
      FPOLD=FP(J-1)+DX*(FPP(J-1)+.5*DX*FPPP(J-1))
      IF (ABS(FPOLD).LE.TOL) GO TO 80
      FN=F(J-1)+DX*(FP(J-1)+DX*(F2+DX*F3))
      NEW=OLD-(FN-FVAL)/FPOLD
      IF (NEW.LT.START) GO TO 80
      NEW=AMIN1(NEW,X(J))
      IF (ABS(NEW-OLD).LT.TOL) GO TO 120
   70 OLD=NEW
      CALL ABORT
   80 RIGHT=X(J)
      LEFT=OLD
      IF (SGN*(FN-FVAL).GT.0.) GO TO 90
      RIGHT=LEFT
      LEFT=XI(L-1)
      IF (L.EQ.1) LEFT=X(1)
   90 DO 110 K=1,50
      IF ((RIGHT-LEFT).LE.TOL) GO TO 120
      NEW=.5*(LEFT+RIGHT)
      DX=NEW-X(J-1)
      FN=F(J-1)+DX*(FP(J-1)+DX*(F2+DX*F3))
      IF ((FN-FVAL)*SGN.LE.0.) GO TO 100
      LEFT=NEW
      GO TO 110
  100 RIGHT=NEW
  110 CONTINUE
  120 XI(L)=NEW
  130 K=J
      MX=NX
      RETURN
  140 FORMAT (* TROUBLE AT *,I5,3X,E16.6)
      END


      SUBROUTINE SPTEN (N,S,F,E,G,A,B,C,D,KM,VM,KN,VN)
C     EXPONENTIAL SPLINE FITTING ROUTINE
      DIMENSION S(1), F(1), E(1), G(1), A(1), B(1), C(1), D(1)
      NM=N-1
      H=S(2)-S(1)
      HI=1./H
      SI=1./SINH(H/E(1))
      TI=SI*COSH(H/E(1))
      IF (KM-2) 10,20,30
   10 B(1)=E(1)*(E(1)*HI-TI)
```

```
      C(1)=E(1)*(SI-E(1)*HI)
      D(1)=VM+HI*(F(1)-F(2))+E(1)*G(1)*(SI-TI)+.5*H*G(1)
      GO TO 40
   20 B(1)=1.
      C(1)=0.
      D(1)=VM
      GO TO 40
   30 B(1)=-TI
      C(1)=SI
      D(1)=G(1)*(SI-TI)+VM*E(1)
   40 XX=1./B(1)
      D(1)=XX*D(1)
      DO 50 I=2,NM
      HM=H
      HIM=HI
      SIM=SI
      TIM=TI
      H=S(I+1)-S(I)
      HI=1./H
      SI=1./SINH(H/E(I))
      TI=SI*COSH(H/E(I))
      A(I)=E(I-1)*(SIM-HIM*E(I-1))
      B(I)=E(I)*(HI*E(I)-TI)+E(I-1)*(HIM*E(I-1)-TIM)
      C(I)=E(I)*(SI-HI*E(I))
      D(I)=HIM*(F(I)-F(I-1))+E(I-1)*G(I-1)*(SIM-TIM)+.5*HM*G(I-1)+HI*(F(
     1I)-F(I+1))+E(I)*G(I)*(SI-TI)+.5*H*G(I)
      C(I-1)=XX*C(I-1)
      XX=1./(B(I)-A(I)*C(I-1))
   50 D(I)=(D(I)-A(I)*D(I-1))*XX
      IF (KN-2) 60,70,80
   60 A(N)=E(NM)*(HI*E(NM)-SI)
      B(N)=E(NM)*(TI-HI*E(NM))
      D(N)=VN+HI*(F(NM)-F(N))-.5*H*G(NM)+E(NM)*G(NM)*(TI-SI)
      GO TO 90
   70 A(N)=0.
      B(N)=1.
      D(N)=VN
      GO TO 90
   80 A(N)=-SI/E(NM)
      B(N)=TI/E(NM)
      D(N)=VN+G(NM)*(TI-SI)/E(NM)
   90 C(NM)=XX*C(NM)
      XX=1./(B(N)-A(N)*C(NM))
      D(N)=(D(N)-A(N)*D(NM))*XX
      DO 100 J=2,N
      I=N+1-J
  100 D(I)=D(I)-C(I)*D(I+1)
      DI=D(1)
      DO 110 I=1,NM
      DIP=D(I+1)
      H=S(I+1)-S(I)
      SI=1./SINH(H/E(I))
```

```
      A(I)=F(I+1)-F(I)+(DI-DIP)*E(I)**2-.5*G(I)*H**2
      B(I)=F(I)+(G(I)-DI)*E(I)**2
      C(I)=SI*(DIP-G(I))*E(I)**2
      D(I)=SI*(DI-G(I))*E(I)**2
  110 DI=DIP
      RETURN
      END


      SUBROUTINE INTEN (NX,SI,FI,N,S,A,B,C,D,E,G)
C     INTERPOLATION ROUTINE FOR EXPONENTIAL SPLINE
      DIMENSION SI(1), FI(1), S(1), A(1), B(1), C(1), D(1), E(1), G(1)
      I=0
      J=1
      JP=J+1
   10 I=I+1
   20 IF (SI(I).LT.S(JP)) GO TO 30
      IF (JP.EQ.N) GO TO 30
      J=J+1
      JP=J+1
      GO TO 20
   30 HI=1./(S(JP)-S(J))
      EI=1./E(J)
      FI(I)=B(J)+HI*A(J)*(SI(I)-S(J))+C(J)*SINH(EI*(SI(I)-S(J)))+D(J)*SI
     1NH(EI*(S(JP)-SI(I)))+.5*G(J)*(SI(I)-S(J))**2
      IF (I.LT.NX) GO TO 10
      RETURN
      END


      SUBROUTINE FOUCF (N,G,X,A,B)
C     COMPUTE FOURIER COEFFICIENTS BY FAST FOURIER TRANSFORM
C     FROM DATA G AROUND THE UNIT CIRCLE
      COMPLEX G,EIV,QP,X,GK
      DIMENSION G(1), X(1), A(1), B(1)
      DATA PI/3.1415926/
      L=N/2
      V=PI/L
      EIV=CMPLX(COS(V),SIN(V))
      MX=-1
```

```
      CALL FFORMS (MX,L,G,A,B,X)
      GK=0.
      I=1
      DO 10 J=1,L,2
      X(J)=CMPLX(-B(I),A(I))
      X(J+1)=X(J)*EIV
   10 I=I+1
      K=L
      DO 20 J=1,L
      QP=GK-CONJG(G(J))
      GK=GK+CONJG(G(J))-QP*X(J)
      X(J)=.5*CONJG(GK)
      GK=G(K)
   20 K=K-1
      X(L+1)=CMPLX(-AIMAG(X(1)),0.)
      X(1)=CMPLX(REAL(X(1)),0.)
      RETURN
      END


      SUBROUTINE FFORMS (MX,NX,G,CN,SN,Y)
C     COMPLEX FAST FOURIER TRANSFORM
      DIMENSION CN(1), SN(1), G(1), Y(1)
      COMPLEX G,Y,EIX
      LOGICAL ISW,NR2
      DATA PI/3.1415926/
      N=IABS(NX)
      M=IABS(MX)
      IF ((N.LT.2).OR.(M.LT.1)) RETURN
      FAC=-ISIGN(1,NX)
      NM=N*M
      NH=NM/2
      NQM=NH/2
      NQ=1
      IF (4*(N/4).EQ.N) NQ=4
      NS=1
      NT=N
      NR=2
      IF (MX.GT.0) GO TO 30
      H=0.
      DH=(PI+PI)/FLOAT(N)
      NDM=(N+1)/2
      DO 10 J=1,NDM
      CN(J)=COS(H)
      SN(J)=SIN(H)
   10 H=H+DH
```

```
      K=N
      DO 20 J=1,NDM
      SN(J+NDM)=-SN(J)
      CN(K)=CN(J+1)
   20 K=K-1
   30 ISW=.TRUE.
      LA=NH-ISIGN(NQM,NX)
   40 NSKP=MIN0(2,NR-1)
      DO 50 K=NR,NT,NSKP
      IF (MOD(NT,K).EQ.0) GO TO 60
   50 CONTINUE
   60 ND=NT/K
      NS=NS*K
      NR=K
      NR2=.TRUE.
      IF (NR.NE.2) NR2=.FALSE.
      NDM=ND*M
      NTM=NT*M
      IF (NR2) GO TO 120
      NSX=NS
      NQ=1
      L=NDM
      IF (ISW) GO TO 90
      DO 70 J=1,NDM
   70 G(J)=Y(J)+Y(NDM+J)
      DO 80 KK=3,NR
      L=L+NDM
      DO 80 J=1,NDM
   80 G(J)=G(J)+Y(L+J)
      GO TO 160
   90 DO 100 J=1,NDM
  100 Y(J)=G(J)+G(NDM+J)
      DO 110 KK=3,NR
      L=L+NDM
      DO 110 J=1,NDM
  110 Y(J)=Y(J)+G(L+J)
      GO TO 160
  120 NSX=NS/NR
      IF (ISW) GO TO 140
      DO 130 J=1,NDM
      G(J)=Y(J)+Y(NDM+J)
  130 G(NH+J)=Y(J)-Y(NDM+J)
      GO TO 160
  140 DO 150 J=1,NDM
      Y(J)=G(J)+G(NDM+J)
  150 Y(NH+J)=G(J)-G(NDM+J)
      IF (NS.EQ.2) GO TO 320
  160 ID=ND
      IDM=NDM
      ITM=0
      NSQ=NS/NQ+1
      DO 310 I=2,NSX
```

```
      ITM=ITM+NTM
      IF (I.EQ.NSQ) GO TO 220
      EIX=CMPLX(CN(ID+1),FAC*SN(ID+1))
      IF (NR2) GO TO 260
      IF (ITM.GE.NM) ITM=ITM-NM
      LD=ITM+NDM
      MM=ID
      IF (ISW) GO TO 190
      DO 170 J=1,NDM
  170 G(IDM+J)=Y(ITM+J)+EIX*Y(LD+J)
      DO 180 KK=3,NR
      MM=MM+ID
      IF (MM.GE.N) MM=MM-N
      EIX=CMPLX(CN(MM+1),FAC*SN(MM+1))
      LD=LD+NDM
      DO 180 J=1,NDM
  180 G(IDM+J)=G(IDM+J)+EIX*Y(LD+J)
      GO TO 300
  190 DO 200 J=1,NDM
  200 Y(IDM+J)=G(ITM+J)+EIX*G(LD+J)
      DO 210 KK=3,NR
      MM=MM+ID
      IF (MM.GE.N) MM=MM-N
      EIX=CMPLX(CN(MM+1),FAC*SN(MM+1))
      LD=LD+NDM
      DO 210 J=1,NDM
  210 Y(IDM+J)=Y(IDM+J)+EIX*G(LD+J)
      GO TO 300
  220 LP=NH+ISIGN(NQM,NX)
      LD=NH+NDM
      IF (ISW) GO TO 240
      DO 230 J=1,NDM
      G(LA+J)=Y(NH+J)+CMPLX(AIMAG(Y(LD+J)),-REAL(Y(LD+J)))
  230 G(LP+J)=Y(NH+J)-CMPLX(AIMAG(Y(LD+J)),-REAL(Y(LD+J)))
      GO TO 300
  240 DO 250 J=1,NDM
      Y(LA+J)=G(NH+J)+CMPLX(AIMAG(G(LD+J)),-REAL(G(LD+J)))
  250 Y(LP+J)=G(NH+J)-CMPLX(AIMAG(G(LD+J)),-REAL(G(LD+J)))
      GO TO 300
  260 LP=IDM+NH
      LD=ITM+NDM
      IF (ISW) GO TO 280
      DO 270 J=1,NDM
      G(IDM+J)=Y(ITM+J)+EIX*Y(LD+J)
  270 G(LP+J)=Y(ITM+J)-EIX*Y(LD+J)
      GO TO 300
  280 DO 290 J=1,NDM
      Y(IDM+J)=G(ITM+J)+EIX*G(LD+J)
  290 Y(LP+J)=G(ITM+J)-EIX*G(LD+J)
  300 IDM=IDM+NDM
  310 ID=ID+ND
  320 NT=ND
```

```
      ISW=.NOT.ISW
      IF (ND.GT.1) GO TO 40
      MX=M
      IF (NX.GT.0) GO TO 340
      IF (ISW) RETURN
      DO 330 J=1,NM
  330 G(J)=Y(J)
      RETURN
  340 ENI=1./FLOAT(N)
      IF (ISW) GO TO 360
      DO 350 J=1,NM
  350 G(J)=ENI*Y(J)
      RETURN
  360 DO 370 J=1,NM
  370 G(J)=ENI*G(J)
      RETURN
      END


      SUBROUTINE GRAPH (TC,GAP,GC)
C     CONTROL ROUTINE FOR GRAPHICS
      COMPLEX ROTATE,CHAR
      REAL MACHN
      COMMON SI(300),QIN(300),SS(600),QQ(600),XX(600),UBODY(300),VBODY(3
     100),XBODY(300),YBODY(300),XREAL(300),YREAL(300),S(300),QL(300),MAC
     2HN(300),FP(300),FPP(300),FPPP(300)
      COMMON CHAR(50)
      COMMON /1/ NP,KBM,MODE,NF,NN,MM,NI,NX,IS,KC,LBM,JJ,KK,NB,NC,MRP,NJ
     1,KF,NBPS,NFC,NRN,NPTS,NT,IPLT,LOFF,KP,NOSE,NCAS
      COMMON /2/ PI,TP,GRID,TOL
      COMPLEX XIA,XIB,XIC,DZ
      COMMON /3/ R,R1,R2,RATC,PHMN,GAM,WTAIL,XIA,XIB,XIC,UA,UB,VA,VB,UC,
     1VC,TRANU,TRANL,CONE,CTWO,CTHR,EP,RN,SO,THNOS,DZ
      COMPLEX BP,ETJ
      COMMON /8/ BP(65),ETJ(65)
      COMMON /G/ N1,N3,N4,N7,M1
      DIMENSION TITLE(10), NAME(2), ID(4)
C     READ IN DATA NEEDED FOR GRAPHICAL OUTPUT
      N1=M1
      REWIND N1
      READ (N1) IDUM,NIN,(SI(J),QIN(J),J=1,NIN)
      READ (N1) NPTS,DUM,DUM,DUM,(SS(J),QQ(J),DUM,J=1,NPTS)
      READ (N1) KP,EMB,EMA,TURN,CL,DX,DY,ROTATE,(UBODY(J),VBODY(J),XBODY
     1(J),YBODY(J),XREAL(J),YREAL(J),S(J),QL(J),MACHN(J),J=1,KP)
      READ (N1)
C     WRITE ONTO TAPE1 FOR USE IN PROGRAM GRAPH
```

```
      WRITE (N1) NB,NC,WTAIL,TC,GAP,GC,DZ,NBPS,(BP(J),J=1,NBPS)
      REWIND N7
   10 READ (N7) K2
      WRITE (N1) K2
      IF (EOF(N7).NE.0) GO TO 30
      DO 20 J=1,K2
      READ (N7) NPT
      WRITE (N1) NPT
      IF (NPT.EQ.0) GO TO 20
      NM=NPT-1
      READ (N7) (CHAR(K),K=1,NM)
      READ (N7) CHAR(NPT)
      WRITE (N1) (CHAR(K),K=1,NPT)
   20 CONTINUE
      GO TO 10
   30 END FILE N1
      IF (IPLT.LT.0) RETURN
C     INITIATE PLOT
      NPG=12*(10+1)
      NRUN=IABS(NRN)
      ENCODE (30,60,ID) NRUN
      ID(4)=ID(4).AND..NOT.7777B
      IF (NRN.GT.0) CALL PLOTS (NPG,ID)
      IF (NRN.LT.0) CALL PLOTSBL (NPG,ID)
      ENCODE (60,70,TITLE) EMB,EMA,TURN,GC
      IF (NB.EQ.1) ENCODE (60,80,TITLE) EMA,CL,DX,DY,TC
      CALL PLOTQS (TITLE,NIN,CTHR,TC,ROTATE)
C     NOW PLOT HODOGRAPH PLANE
      CALL LOCUS
      IF (NB.NE.1) GO TO 40
      CALL BDYPLT (NIN,DZ,TC)
      GO TO 50
C     PLOT CASCADE
   40 CALL CASCAD (TITLE,TC,GC,GAP,CTHR,ROTATE)
   50 CALL PLOT (0.0,0.0,999)
      RETURN
   60 FORMAT (3HRUN,I5)
   70 FORMAT(3X3HM1=F5.3,3X4H M2=F5.3,3X7HDEL TH=F4.0,3X4HG/C=F4.2)
   80 FORMAT(2HM=,F5.3,3X3HCL=F5.2,3X3HDX=F5.2,3X3HDY=F5.2,3X4HT/C=F4.2)
      END


      SUBROUTINE PLOTQS (TITLE,NIN,CTHR,TC,ROT)
C     PLOTS INPUT SPEED AND PRESSURE OR MACH DISTRIBUTIONS
      COMPLEX CON,ROT
      REAL MACHN
```

```
      COMMON SI(300),QIN(300),SS(600),QQ(600),XX(600),UBODY(300),VBODY(3
     100),XBODY(300),YBODY(300),XREAL(300),YREAL(300),S(300),QL(300),MAC
     2HN(300),FP(300),FPP(300),FPPP(300)
      DIMENSION ANGL(300)
      COMMON/G/ N1,N3,N4,N7,M1
      COMMON /1/ NP,KBM,MODE,NF,NN,MM,NI,NX,IS,KC,LBM,JJ,KK,NB,NC,MRP,NJ
     1,KF,NBPS,NFC,NRN,NPTS,NT,IPLT,LOFF,KP,NOSE,NCAS
      REAL MACHA,MACHB
      COMMON /7/ MACHA,MACHB,ANGLA,ANGLB
      REAL MACH
      COMMON /A/ GAMMA,MACH,RHOINF,EMU,QSTRSQ
      DIMENSION TITLE(10), TITL2(10)
      EMACH(QS)=SQRT(QS/(COSQ-((GAMMA-1.)/2.)*QS))
      CP(QS)=HEADI*(RHO(QS)**GAMMA-PINF)
      PI=ACOS(-1.)
      RAD=180./PI
      GAMX=(GAMMA-1.)/2.
      COSQ=GAMX+1./(MACH*MACH)
      QSB=COSQ*(MACHB*MACHB)/(1.+GAMX*(MACHB*MACHB))
      PINF=RHO(QSB)**GAMMA
      HEADI=2./(GAMMA*(MACHB*MACHB)*PINF)
C     PLOT INPUT SPEED DISTRIBUTION Q(S)
      PLT=8.
      SCALE=PLT/2.
      SFI=SCALE/2.
      CALL PLOT (5.5,6.0,-3)
      IPEN=3
      DO 10 L=1,NPTS
      SS(L)=SS(L)-1.
      CALL PLOT (SCALE*SS(L),SFI*QQ(L),IPEN)
   10 IPEN=2
      DO 20 L=1,NIN
   20 CALL SYMBOL (SCALE*(SI(L)-1.),SFI*QIN(L),.07,3,0.,-1)
      NM1=6H INPUT
      ENCODE (30,100,TITL2) NM1
      CALL SYMBOL (-1.5,-5.0,.14,TITL2,0.,30)
      CALL XYAXES (-SCALE,0.,-1.,1.,.25,PLT,0.,6H(F5.2))
      CALL SYMBOL (SCALE-.5,-.5,.14,1HS,0.,1)
      CALL XYAXES (-SCALE,-SCALE,-2.,2.,.50,PLT,90.,6H(F5.1))
      CALL SYMBOL (-SCALE+.5,SCALE-.5,.14,1HQ,0.,1)
      CALL FRAME
C     PLOT MACH OR CP ALONG AIRFOIL
      IPL=MOD(IPLT/10,10)
      IF ((IPL.EQ.0).OR.(IPL.EQ.5)) RETURN
      CPOR=2.25
      EMOR=0.
      SMACH=3.
      SCP=-4.5/2.4
      SXR=AMIN1(4./TC,6.)
      CALL PLOT (-2.5,-.5,-3)
C     PLOT MACH NUMBER AT THE TAIL POINT
      QS=UBODY(1)*UBODY(1)+VBODY(1)*VBODY(1)
```

```
      IF (IPL.LT.5) EMN=SMACH*MACHN(1)+EMOR
      IF (IPL.GT.5) EMN=SCP*CP(QS)+CPOR
      CALL SYMBOL (SXR*XREAL(1),EMN,.14,3,0.,-1)
      CALL SPLIF (KP,S,XREAL,FP,FPP,FPPP,3,0.,3,0.)
      CALL INTPL (NPTS,SS,XX,S,XREAL,FP,FPP,FPPP)
      IPEN=3
      DO 30 L=1,NPTS
      QS=QQ(L)*QQ(L)
      IF (IPL.LT.5) EMN=SMACH*EMACH(QS)+EMOR
      IF (IPL.GT.5) EMN=SCP*CP(QS)+CPOR
      CALL PLOT (SXR*XX(L),EMN,IPEN)
   30 IPEN=2
C     PLOT OUTPUT VALUES OF MACH OR CP
      DO 40 L=1,KP
      QS=QL(L)*QL(L)
      IF (IPL.LT.5) EMN=EMACH(QS)*SMACH+EMOR
      IF (IPL.GT.5) EMN=SCP*CP(QS)+CPOR
   40 CALL SYMBOL (SXR*XREAL(L),EMN,.07,3,0.,-1)
      WORD=1HM
      IF (IPL.GT.5) WORD=2HCP
      CALL SYMBOL (-.5,4.5,.14,WORD,.0,3)
      CALL PLOT (5.1,4.7,3)
      CALL PLOT (5.35,4.7,2)
      WORD=6HINPUT
      CALL SYMBOL (5.5,4.6,.14,WORD,0.,6)
      WORD=8H+ OUTPUT
      CALL SYMBOL (5.15,4.1,.14,WORD,0.,8)
      IF (IPL.LE.5) GO TO 50
      CALL XYAXES (-1.,0.,1.2,-1.2,-.40,4.5,90.,6H(F5.1))
      CPCRIT=CP((COSQ+COSQ)/(GAMMA+1.))
      CALL PLOT (-1.1,SCP*CPCRIT+CPOR,3)
      CALL PLOT (-.90,SCP*CPCRIT+CPOR,2)
      GO TO 60
   50 CALL XYAXES (-1.,0.,0.,1.5,.25,4.5,90.,6H(F5.2))
C     PLOT BODY
   60 YMAX=0.
      YMIN=0.
      DO 70 J=1,KP
      YMAX=AMAX1(YMAX,YREAL(J))
   70 YMIN=AMIN1(YMIN,YREAL(J))
      YOR=2.5+.5*SXR*(YMAX-YMIN)
      IF ((IPL.EQ.2).OR.(IPL.EQ.4).OR.(IPL.EQ.7).OR.(IPL.EQ.9)) YOR=YOR-
     11.
      CALL PLOT (0.,-YOR,-3)
      REWIND N1
      READ(N1)
      READ(N1)
      READ(N1)
      READ(N1) (ANGL(J),J=1,KP)
      IPEN=3
      DO 80 L=1,KP
      IF (MACHN(L).LT..005) GO TO 80
```

```
      ANG=RAD*ANGL(L)+90.
      CALL SYMBOL(SXR*XREAL(L),SXR*YREAL(L),.07,13,ANG,-1)
   80 IPEN=2
      IF ((IPL.EQ.1).OR.(IPL.EQ.2).OR.(IPL.EQ.6).OR.(IPL.EQ.7)) GO TO 90
      FAC=SXR*CTHR
      CON=0.
      CALL PLTCHR (FAC,CON,ROT)
   90 CALL SYMBOL (-1.,YOR-4.5,.14,TITLE,0.,60)
C     RESTORE ORIGIN TO ORIGINAL POSITION
      CALL PLOT (-3.,YOR-5.5,-3)
      CALL FRAME
      RETURN
  100 FORMAT (A6,19H SPEED DISTRIBUTION)
      END


      SUBROUTINE LOCUS
C     PLOTS COMPLEX CHARACTERISTIC HODOGRAPH PLANE
      COMPLEX SB,HSQPR,XII,SFROMX,CSQRT,XIMIN,XIMAX
      DIMENSION CONL(20), TEMP(61), TITLE(12)
      REAL MACHN
      COMMON SI(300),QIN(300),SS(600),QQ(600),XX(600),UBODY(300),VBODY(3
     100),XBODY(300),YBODY(300),XREAL(300),YREAL(300),S(300),QL(300),MAC
     2HN(300),FP(300),FPP(300),FPPP(300)
      COMMON /1/ NP,KBM,MODE,NF,NN,MM,NI,NX,IS,KC,LBM,JJ,KK,NB,NC,MRP,NJ
     1,KF,NBPS,NFC,NRN,NPTS,NT,IPLT,LOFF,KP,NOSE,NCAS
      COMMON /2/ PI,TP,GRID,TOL
      COMPLEX XIA,XIB,XIC,DZ
      COMMON /3/ R,R1,R2,RATC,PHMN,GAM,WTAIL,XIA,XIB,XIC,UA,UB,VA,VB,UC,
     1VC,TRANU,TRANL,CONE,CTWO,CTHR,EP,RN,SO,THNOS,DZ
      REAL MACH
      COMMON /A/ GAMMA,MACH,RHOINF,EMU,QSTRSQ
      EXTERNAL SFROMX,CSQRT
      IPL=MOD(IPLT,10)
      IF ((IPL.EQ.0).OR.(IPL.EQ.5)) RETURN
      GAMM=(GAMMA-1.)/2.
      COSQ=GAMM+1./(MACH*MACH)
      XCEN=5.5
      YCEN=6.0
      SF=5.5/(R1+R2)
C     SPECIAL FORMAT STATEMENTS FOR GREEK LETTERS IN CAPTION VIA NBSLIB
      RA=REAL(XIA)
      RB=REAL(XIB)
      RC=REAL(XIC)
      AIMA=AIMAG(XIA)
      AIMB=AIMAG(XIB)
```

```
      AIMC=AIMAG(XIC)
      IF (CABS(XIA-XIB).LT.TOL) GO TO 10
      ENCODE (70,100,TITLE) RA,AIMA,RB,AIMB,RC,AIMC
      CALL SYMBOL (2.0,1.,.14,TITLE,0.,70)
      GO TO 20
   10 ENCODE (70,110,TITLE) RA,AIMA,RC,AIMC
      CALL SYMBOL (2.5,1.,.14,TITLE,0.,70)
C     SCALE AND CENTER PLOT
   20 YTOP=R2*SF
      YSZ=2.*YTOP
      XTOP=R1*SF
      XSZ=2.*XTOP
      IX=60
      JX=INT(R2*FLOAT(IX)/R1+TOL)
      CALL PLOT (XCEN-XTOP,YCEN-YTOP,-3)
      XIMIN=-CMPLX(R1,R2)
C     FIND LEVEL CURVES OF Q
      XIMAX=-XIMIN
      IND=1
      IF (IPL.LT.5) GO TO 40
      IND=20
   30 IF (COSQ.LE.8.) GO TO 40
      COSQ=.25*COSQ
      GO TO 30
   40 DO 50 J=1,IND
      EMX=FLOAT(IND+1-J)/FLOAT(IND)
      XII=COSQ*EMX*EMX/(1.+GAMM*EMX*EMX)
      CALL GETHSQ(XII,SB,HSQPR)
   50 CONL(IND+1-J)=REAL(SB)
      CALL CONTOR (IX,JX,IND,XSZ,YSZ,SFROMX,CONL,TEMP,XIMIN,XIMAX)
      CALL PLOT (XTOP,YTOP,-3)
      RA=SF*REAL(XIA)
      AIMA=SF*AIMAG(XIA)
      RB=SF*REAL(XIB)
      AIMB=SF*AIMAG(XIB)
      RC=SF*REAL(XIC)
      AIMC=SF*AIMAG(XIC)
      CALL SYMBOL (RA,AIMA,.14,11,0.,-1)
      CALL SYMBOL (RC,AIMC,.14,11,.0,-1)
      CALL SYMBOL (SF*R1*COS(WTAIL),SF*R2*SIN(WTAIL),.14,5,0.,-1)
      CALL SYMBOL (SF*R1*COS(THNOS),SF*R2*SIN(THNOS),.14,1,0.,-1)
      IPEN=3
      IF (NB.EQ.1) GO TO 60
      CALL SYMBOL (RB,AIMB,.14,11,0.,-1)
      CALL PLOT (RA,AIMA,IPEN)
      IPEN=2
   60 CALL PLOT (RB,AIMB,IPEN)
      CALL PLOT (RC,AIMC,2)
      IF ((IPL.EQ.3).OR.(IPL.EQ.4).OR.(IPL.EQ.8).OR.(IPL.EQ.9)) GO TO 70
      NAX=R1/.5
      AX=FLOAT(NAX+1)*.5
      CALL XYAXES (-SF*AX,0.,-AX,AX,.50,2.*SF*AX,0.,6H(F5.1))
```

```
      NAX=R2/.5
      AX=FLOAT(NAX+1)*.5
      CALL XYAXES (0.,-SF*AX,-AX,AX,.50,2.*SF*AX,90.,6H(F5.1))
C     PLOT PATHS OF INTEGRATION
   70 CALL PLPATH (NP,SF)
      IPEN=3
      DT=TP/NF
      DO 80 J=2,NT
      ANG=FLOAT(J-1)*DT
      CALL PLOT (SF*R1*COS(ANG),SF*R2*SIN(ANG),IPEN)
   80 IPEN=2
      DO 90 J=2,NT
      ANG=FLOAT(J-1)*DT
   90 CALL SYMBOL (SF*R1*COS(ANG),SF*R2*SIN(ANG),.07,3,0.,-1)
C     RESTORE ORIGIN TO ORIGINAL POSITION
      CALL PLOT (-XCEN,-YCEN,-3)
      CALL FRAME
      RETURN
  100 FORMAT (9H#A =,F5.2,1H,,F5.2,4X,9H#B =,F5.2,1H,,F5.2,4X,
     19H#C =,F5.2,1H,,F5.2)
  110 FORMAT (8X,9H#A =,F5.2,1H,,F5.2,4X,9H#C =,F5.2,1H,,F5.2)
      END


      SUBROUTINE CONTOR (M,N,NC,XS,YS,FN,CONL,TEMP,XIMIN,XIMAX)
C     PLOTS LEVEL CURVES OF ABSOLUTE VALUE OF FN
      COMPLEX FN,W,XIMIN,XIMAX
      COMPLEX S
      COMPLEX XIA,XIB,XIC,DZ
      COMMON /3/ R,R1,R2,RATC,PHMN,GAM,WTAIL,XIA,XIB,XIC,UA,UB,VA,VB,UC,
     1VC,TRANU,TRANL,CONE,CTWO,CTHR,EP,RN,SO,THNOS,DZ
      LOGICAL ISW,JSW,KSW
      DIMENSION CONL(1), TEMP(1)
      EQUIVALENCE (A,JA), (B,JB), (C,JC), (D,JD)
      DATA ISW/.TRUE./,TOL/.01/
      XS2=XS/2.
      YS2=YS/2.
      XSS=XS2*XS2
      YSS=YS2*YS2
      RDXI=(REAL(XIMAX)-REAL(XIMIN))/FLOAT(M)
      ADXI=(AIMAG(XIMAX)-AIMAG(XIMIN))/FLOAT(N)
      DX=XS/FLOAT(M)
      DY=YS/FLOAT(N)
      NCL=IABS(NC)
      MP=M+1
      DO 20 I=1,MP
```

```
      S=XIMIN+FLOAT(I-1)*RDXI
      SCHK=REAL(S)*REAL(S)/(R1*R1)+AIMAG(S)*AIMAG(S)/(R2*R2)
      IF (SCHK.LE.1.2) GO TO 10
      XR=(1.-AIMAG(S)*AIMAG(S)/(R2*R2))*SIGN(1.,REAL(S))
      S=CMPLX(XR,AIMAG(S))
   10 W=FN(S)
   20 TEMP(I)=REAL(W)*REAL(W)+AIMAG(W)*AIMAG(W)
      YP=0.
      DO 180 J=1,N
      XP=0.
      S=XIMIN+CMPLX(0.,FLOAT(J)*ADXI)
      SCHK=REAL(S)*REAL(S)/(R1*R1)+AIMAG(S)*AIMAG(S)/(R2*R2)
      IF (SCHK.LE.1.2) GO TO 30
      XR=(1.-AIMAG(S)*AIMAG(S)/(R2*R2))*SIGN(1.,REAL(S))
      S=CMPLX(XR,AIMAG(S))
   30 W=FN(S)
      TM=REAL(W)*REAL(W)+AIMAG(W)*AIMAG(W)
      YPP=YP+DY
      DO 170 I=1,M
      XPP=XP+DX
      S=XIMIN+CMPLX(FLOAT(I)*RDXI,FLOAT(J)*ADXI)
      SCHK=REAL(S)*REAL(S)/(R1*R1)+AIMAG(S)*AIMAG(S)/(R2*R2)
      IF (SCHK.LE.1.2) GO TO 40
      XR=(1.-AIMAG(S)*AIMAG(S)/(R2*R2))*SIGN(1.,REAL(S))
      S=CMPLX(XR,AIMAG(S))
   40 W=FN(S)
      TP=REAL(W)*REAL(W)+AIMAG(W)*AIMAG(W)
      TOP=AMAX1(TP,TM,TEMP(I),TEMP(I+1))
      BOT=AMIN1(TP,TM,TEMP(I),TEMP(I+1))
      IF (CONL(NCL).LT.BOT) GO TO 160
      DO 50 L1=1,NCL
      IF (CONL(L1).GT.BOT) GO TO 60
   50 CONTINUE
      GO TO 160
   60 DO 150 L=L1,NCL
      IF (CONL(L).GT.TOP) GO TO 160
      A=TEMP(I)-CONL(L)
      B=TEMP(I+1)-CONL(L)
      C=TM-CONL(L)
      D=TP-CONL(L)
      IA=ISIGN(1,JA)
      IB=ISIGN(2,JB)
      IC=ISIGN(4,JC)
      ID=ISIGN(8,JD)
      ITYP=8+(IA+IB+IC+ID+1)/2
      GO TO (150,70,80,90,100,110,120,130,130,120,110,100,90,80,70,150),
     1 ITYP
   70 X1=XP
      Y1=YP+DY*A/(A-C)
      X2=XP+DX*A/(A-B)
      Y2=YP
      GO TO 140
```

```
 80 X1=XP+DX*A/(A-B)
    Y1=YP
    X2=XPP
    Y2=YP+DY*B/(B-D)
    GO TO 140
 90 X1=XP
    Y1=YP+DY*A/(A-C)
    X2=XPP
    Y2=YP+DY*B/(B-D)
    GO TO 140
100 X1=XP
    Y1=YP+DY*A/(A-C)
    X2=XP+DX*C/(C-D)
    Y2=YPP
    GO TO 140
110 X1=XP+DX*A/(A-B)
    Y1=YP
    X2=XP+DX*C/(C-D)
    Y2=YPP
    GO TO 140
120 ISW=.FALSE.
    E=(A+D)+(B+C)
    RR=ABS(E/((A+D)-(B+C)))
    JSW=RR.LE.TOL
    IF (JSW) GO TO 90
    KSW=A*E.GT.0.
    IF (KSW) GO TO 80
130 X1=XP+DX*C/(C-D)
    Y1=YPP
    X2=XPP
    Y2=YP+DY*B/(B-D)
140 CALL PLOT (X1,Y1,3)
    PTST=(X2-XS2)*(X2-XS2)/XSS+(Y2-YS2)*(Y2-YS2)/YSS
    IF (PTST.LE.1.05) CALL PLOT (X2,Y2,2)
    IF (ISW) GO TO 150
    ISW=.TRUE.
    IF (JSW) GO TO 110
    IF (KSW) GO TO 100
    GO TO 70
150 CONTINUE
160 TEMP(I)=TM
    XP=XPP
    TM=TP
170 CONTINUE
    TEMP(M+1)=TP
    YP=YPP
180 CONTINUE
    RETURN
    END
```

```
      SUBROUTINE PLPATH (K1,SF)
C     PLOTS INTEGRATION PATHS IN THE ELLIPSE
      COMPLEX XI
      DIMENSION XI(300)
      COMMON /1/ NP,KBM,MODE,NF,NN,MM,NI,NX,IS,KC,LBM,JJ,KK,NB,NC,MRP,NJ
     1,KF,NBPS,NFC,NRN,NPTS,NT,IPLT,LOFF,KP,NOSE,NCAS
      COMPLEX XIA,XIB,XIC,DZ
      COMMON /3/ R,R1,R2,RATC,PHMN,GAM,WTAIL,XIA,XIB,XIC,UA,UB,VA,VB,UC,
     1VC,TRANU,TRANL,CONE,CTWO,CTHR,EP,RN,SO,THNOS,DZ
      XI(NC)=XIC
      DO 30 MODE=1,K1
C     PLOT SUBSONIC PATH OR FIRST TRANSONIC PATH
      KC=0
      CALL GTPATH (XI,NC+1,NN)
      IPEN=3
      DO 10 J=NC,NN
      CALL PLOT (SF*REAL(XI(J)),SF*AIMAG(XI(J)),IPEN)
   10 IPEN=2
      IF (KC.EQ.0) GO TO 30
C     PLOT TRANSONIC PATH
      CALL GTPATH (XI,NC+1,NN)
      IPEN=3
      DO 20 J=NC,NN
      CALL PLOT (SF*REAL(XI(J)),SF*AIMAG(XI(J)),IPEN)
   20 IPEN=2
   30 CONTINUE
C     PLOT SUPERSONIC PATH
      MODE=-12
      K=1
   40 KC=0
      CALL SUPATH (K,XI,NC+1,NN)
      IF (NN.EQ.0) GO TO 70
      IPEN=3
      DO 50 J=NC,NN
      CALL PLOT (SF*REAL(XI(J)),SF*AIMAG(XI(J)),IPEN)
   50 IPEN=2
      CALL SUPATH (K,XI,NC+1,NN)
      IPEN=3
      DO 60 J=NC,NN
      CALL PLOT (SF*REAL(XI(J)),SF*AIMAG(XI(J)),IPEN)
   60 IPEN=2
      K=K+1
      GO TO 40
   70 CONTINUE
      RETURN
      END
```

```
      SUBROUTINE BDYPLT (NIN,DZ,TC)
C     PLOTS PROFILE IN HORIZONTAL POSITION FOR ISOLATED AIRFOIL
      REAL MACHN
      COMMON SI(300),QIN(300),SS(600),QQ(600),XX(600),UBODY(300),VBODY(3
     100),XBODY(300),YBODY(300),XREAL(300),YREAL(300),S(300),QL(300),MAC
     2HN(300),FP(300),FPP(300),FPPP(300)
      COMMON /1/ NP,KBM,MODE,NF,NN,MM,NI,NX,IS,KC,LBM,JJ,KK,NB,NC,MRP,NJ
     1,KF,NBPS,NFC,NRN,NPTS,NT,IPLT,LOFF,KP,NOSE,NCAS
      REAL MACHA,MACHB
      COMMON /7/ MACHA,MACHB,ANGLA,ANGLB
      DIMENSION TITLE(10)
      COMPLEX DZ
      DIMENSION XT(500),ST(500),YT(500),YP(500),YPP(500),YPPP(500)
      DX=REAL(DZ)
      DY=AIMAG(DZ)
      SC=10./AMAX1(XBODY(1),XBODY(KP))
      ENCODE (50,30,TITLE) DX,DY,TC
      CALL SYMBOL (2.0,2.,.14,TITLE,0.,50)
      IF (NB.NE.1) ENCODE (50,40,TITLE) ANGLA,ANGLB
      IF (NB.EQ.1) ENCODE (50,50,TITLE) ANGLA
      CALL SYMBOL (2.0,1.,.14,TITLE,0.,50)
C     FIND ORIGIN FOR PLOT DEPENDING ON ORIENTATION OF AIRFOIL
      YMAX=0.
      YMIN=0.
      DO 10 J=1,KP
      YMAX=AMAX1(YMAX,YBODY(J))
   10 YMIN=AMIN1(YMIN,YBODY(J))
      XOR=1.
      YOR=6.-.5*SC*(YMAX-YMIN)
      CALL PLOT (XOR,YOR,-3)
      IPEN=3
      NPLT=201
      DS=2./FLOAT(NPLT-1)
      ST(1)=-1.
      DO 35 J=2,NPLT
   35 ST(J)=ST(J-1)+DS
      ST(NPLT)=1.
      CALL SPLIF(KP,S,XREAL,FP,FPP,FPPP,3,0.,3,0.)
      CALL INTPL(NPLT,ST,XT,S,XREAL,FP,FPP,FPPP)
      CALL SPLIF(KP,S,YREAL,FP,FPP,FPPP,3,0.,3,0.)
      CALL INTPL(NPLT,ST,YT,S,YREAL,FP,FPP,FPPP)
      DO 20 J=1,NPLT
      CALL PLOT(SC*XT(J),SC*YT(J),IPEN)
   20 IPEN=2
C     RETURN ORIGIN
      CALL PLOT (-XOR,-YOR,-3)
      RETURN
```

```
   30 FORMAT (5X5HDX = F5.3,5X5HDY = F5.3,5X6HT/C = F5.3)
   40 FORMAT (7X,9HANGLE1 = F7.2,5X,9HANGLE2 = F7.2)
   50 FORMAT (14X,18HANGLE OF ATTACK = F5.3)
      END


      SUBROUTINE CASCAD (TITLE,TC,GC,GAP,CTHR,ROT)
C     PLOTS CASCADE OF AIRFOILS
      COMMON /1/ NP,KBM,MODE,NF,NN,MM,NI,NX,IS,KC,LBM,JJ,KK,NB,NC,MRP,NJ
     1,KF,NBPS,NFC,NRN,NPTS,NT,IPLT,LOFF,KP,NOSE,NCAS
      REAL MACHN
      COMMON SI(300),QIN(300),SS(600),QQ(600),XX(600),UBODY(300),VBODY(3
     100),XBODY(300),YBODY(300),XREAL(300),YREAL(300),S(300),QL(300),MAC
     2HN(300),FP(300),FPP(300),FPPP(300)
      COMMON /2/ PI,TP,GRID,TOL
      COMPLEX ROT1,ROT
      DIMENSION TITLE(10)
      DIMENSION XT(500),ST(500),YT(500),YP(500),YPP(500),YPPP(500)
C     FIND SCALE FACTORS
      XFAC=AMAX1(XREAL(1),XREAL(KP))
      YMAX=0.
      YMIN=0.
      DO 10 J=1,KP
      YMAX=AMAX1(YMAX,YREAL(J))
   10 YMIN=AMIN1(YMIN,YREAL(J))
      YFAC=FLOAT(NCAS-1)*GAP+(YMAX-YMIN)
      SCF=AMIN1(8./XFAC,8./YFAC)
      FAC=SCF*CTHR
      XOR=5.5-.5*SCF*XFAC
      YOR=6.0-.5*SCF*YFAC
      ENCODE (25,40,TITLE) GC
      CALL SYMBOL (3.,1.,.14,TITLE,.0,25)
      CALL PLOT (XOR,YOR,-3)
      CON=-GAP
      DO 30 M=1,NCAS
      CON=CON+GAP
      IPEN=3
      NPLT=201
      DS=2./FLOAT(NPLT-1)
      ST(1)=-1.
      DO 35 J=2,NPLT
   35 ST(J)=ST(J-1)+DS
      ST(NPLT)=1.
      CALL SPLIF(KP,S,XREAL,FP,FPP,FPPP,3,0.,3,0.)
      CALL INTPL(NPLT,ST,XT,S,XREAL,FP,FPP,FPPP)
      CALL SPLIF(KP,S,YREAL,FP,FPP,FPPP,3,0.,3,0.)
```

```
      CALL INTPL(NPLT,ST,YT,S,YREAL,FP,FPP,FPPP)
      DO 20 J=1,NPLT
      CALL PLOT(SCF*XT(J),SCF*(YT(J)+CON),IPEN)
   20 IPEN=2
      ROT1=CMPLX(0.,SCF*CON)
      CALL PLTCHR (FAC,ROT1,ROT)
   30 CONTINUE
C     RESTORE ORIGIN TO ORIGINAL POSITION
      CALL PLOT (-XOR,-YOR,-3)
      RETURN
   40 FORMAT (14X6HG/C = F5.3)
      END


      SUBROUTINE PLTCHR (FAC,CONST,ROT)
C     PLOTS CHARACTERISTICS IN THE REAL SUPERSONIC ZONE
      COMMON /G/ N1,N3,N4,N7,M1
      REAL MACHN
      COMMON SI(300),QIN(300),SS(600),QQ(600),XX(600),UBODY(300),VBODY(3
     100),XBODY(300),YBODY(300),XREAL(300),YREAL(300),S(300),QL(300),MAC
     2HN(300),FP(300),FPP(300),FPPP(300)
      COMPLEX CHAR,ROT,CONST,SB,C1,C2,C3,C4,ROT1,ROT3
      COMMON CHAR(50,50)
      DIMENSION NPT(50)
      REWIND N7
   10 READ (N7) K2
      IF (EOF(N7).EQ.0) GO TO 20
      RETURN
C     READ IN CHARACTERISTIC POINTS
   20 DO 40 J=1,K2
      READ (N7) NPT(J)
      KL=NPT(J)
      IF (KL.EQ.0) GO TO 40
      KLM=KL-1
      READ (N7) (CHAR(J,K),K=1,KLM)
      READ (N7) CHAR(J,KL)
      IPEN=3
      DO 30 K=1,KL
      SB=ROT*FAC*CHAR(J,K)+CONST
      CALL PLOT (REAL(SB),AIMAG(SB),IPEN)
   30 IPEN=2
   40 CONTINUE
C     PLOT SECOND FAMILY OF CHARACTERISTICS
      DO 90 K=1,K2
      IPEN=3
      C1=CHAR(K,1)
```

```
      C2=C1
      DO 50 L=1,K
      J=K+1-L
      IF (NPT(J).LE.L) GO TO 50
      C2=CHAR(J,L)
      SB=FAC*ROT*CHAR(J,L)+CONST
      CALL PLOT (REAL(SB),AIMAG(SB),IPEN)
      IPEN=2
   50 CONTINUE
      IF (CABS(C1-C2).LE.1.E-05) GO TO 90
C     FIND BODY POINT FOR THIS CHARACTERISTIC
      ROT1=CONJG(C2-C1)/CABS(C2-C1)
      ANG4=1.
      ANG3=ANG4
      IK=0
      DO 70 J=1,K2
      IF (NPT(J).EQ.0) GO TO 70
      C3=CHAR(J,NPT(J))
      ROT3=(C3-C1)*ROT1
      ANG3=ATAN2(AIMAG(ROT3),REAL(ROT3))
      IF (IK.EQ.0) GO TO 60
      IF ((ANG3*ANG4.LE.0.).AND.(J.GT.1)) GO TO 80
   60 IK=IK+1
      C4=C3
   70 ANG4=ANG3
      GO TO 90
   80 ANG3=ABS(ANG3)
      ANG4=ABS(ANG4)
      SB=FAC*ROT*(ANG3*C4+ANG4*C3)/(ANG3+ANG4)+CONST
      CALL PLOT (REAL(SB),AIMAG(SB),IPEN)
   90 CONTINUE
      GO TO 10
      END


      SUBROUTINE XYAXES (X,Y,BOT,TOP,SCF,TOTL,ANGL,FORMAT)
C     PLOTS AND LABELS COORDINATE AXIS
      COMPLEX ZB,ZT,H,COR
      H=1.
      COR=(-.40,-.3)
      AN=0.
      IF (ABS(ANGL).NE.90.) GO TO 10
      H=(0.,1.)
      COR=(-.75,0.)
      AN=90.
   10 CALL PLOT (X+REAL(H*TOTL),Y+AIMAG(H*TOTL),3)
```

```
      CALL PLOT (X,Y,2)
      SCL=SCF*TOTL/(TOP-BOT)
      B=BOT
      SC=0.
      ZB=CMPLX(X,Y)
   20 ZT=ZB+COR
      CALL SYMBOL (REAL(ZB),AIMAG(ZB),.14,13,AN,-1)
      ENCODE (10,FORMAT,A) B
      CALL SYMBOL (REAL(ZT),AIMAG(ZT),.14,A,0.,5)
      SC=SC+SCL
      B=B+SCF
      ZB=ZB+H*SCL
      IF (SC.LE.TOTL) GO TO 20
      RETURN
      END
```