Thesis and Dissertation Collection

1988

Investigation of incompressible cascade flows using a viscous/inviscid interactive code.

Snir, Zeev.
Monterey, California. Naval Postgraduate School

http://hdl.handle.net/10945/23059

Downloaded from NPS Archive: Calhoun
INVESTIGATION OF INCOMPRESSIBLE CASCADE FLOWS USING A VISCOS/INVISCID INTERACTIVE CODE

by

Zeev Snir

December 1988

Approved for public release; distribution is unlimited.
INVESTIGATION OF INCOMPRESSIBLE CASCADE FLOWS USING A VISCOS-INVISCID INTERACTIVE CODE

Personal Author(s): Zeev Snir

Type of Report: Master's Thesis

Time Covered: From To

Date of Report (year, month, day): December 1988

Page Count: 135

Abstract:

A two-dimensional, incompressible viscous-inviscid interaction computer code, designed to compute cascade flows, was investigated. Comparison of the flow characteristics predicted by the code with experimentally available data indicates that the code predicts reasonably well flow parameters on lightly loaded cascades. However, the code fails to predict correctly the actual boundary layer development and the velocity distribution for highly loaded cascades. It is concluded that further improvement of the code is needed and recommendations are presented to achieve the required improvements.
Investigation of Incompressible Cascade Flows Using a Viscous/Inviscid Interactive Code

by

Zeev Snir
Major, Israeli Air Force
B.S., Technion, Haifa, Israel, 1977

Submitted in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE IN AERONAUTICAL ENGINEERING

from the

NAVAL POSTGRADUATE SCHOOL
December 1988
ABSTRACT

A two dimensional, incompressible viscous inviscid interaction computer code, designed to compute cascade flows, was investigated. Comparison of the flow characteristics predicted by the code with experimentally available data indicates that the code predicts reasonably well flow parameters on lightly loaded cascades. However, the code fails to predict correctly the actual boundary layer development and the velocity distribution for highly loaded cascades. It is concluded that further improvement of the code is needed and recommendations are presented to achieve the required improvements.
# TABLE OF CONTENTS

I. INTRODUCTION ........................................................................................................... 1

II. CASCADE FLOW PROBLEM FORMULATION ............................................................. 2
   A. INVISCID FLOW METHOD. ....................................................................................... 2
   B. VISCOS FLOW METHOD. ....................................................................................... 3
      1. Boundary Layer Theory ..................................................................................... 3
      2. Interactive Boundary Layer Method ................................................................. 6
      3. Interactive Model ............................................................................................. 8
      4. Turbulence Model ........................................................................................... 9
      5. Transition .................................................................................................... 12

III. DESCRIPTION OF THE COMPUTER CODE ............................................................ 13
   A. GENERAL STRUCTURE OF THE MAIN PROGRAM .............................................. 13
   B. DESCRIPTION OF THE SUBROUTINES ............................................................... 13
      1. Subroutine POTNL ......................................................................................... 13
      2. Subroutine CASBLP ....................................................................................... 14
      3. Subroutine COMPBL .................................................................................... 14
      4. Subroutine BL2D .......................................................................................... 14
      5. Subroutine MAIN2 ........................................................................................ 15
      6. Subroutine OUTPUT ..................................................................................... 15
      7. Subroutine TRANS ....................................................................................... 16
      8. Subroutine FILLUP ....................................................................................... 16
      9. Subroutine EDDY .......................................................................................... 17
     10. Subroutine INTL ............................................................................................ 17

IV. RESULTS AND DISCUSSION ..................................................................................... 18
   A. CD CASCADE ..................................................................................................... 19
      1. Transition location and intermittency distribution ........................................... 19
      2. External Velocity Distribution ........................................................................ 30
      3. Boundary Layer Thickness ............................................................................ 34
      4. Comparison to a Navier Stokes Code .............................................................. 42
LIST OF FIGURES

Figure 1. Controlled Diffusion cascade ........................................... 20
Figure 2. Shape factor comparison on the upper surface ....................... 21
Figure 3. Shape factor comparison on the lower surface ....................... 22
Figure 4. Shape factor on the upper surface without velocity smoothing. .... 24
Figure 5. Displacement thickness ................................................. 25
Figure 6. The effect of the intermittency model ................................ 26
Figure 7. Shape factor on the lower surface with transition input at 21% .... 27
Figure 8. Shape factor at $\beta = 46^\circ$ on the upper surface ............... 28
Figure 9. Shape factor at $\beta = 46^\circ$ on the lower surface. ............... 29
Figure 10. External velocity at $\beta = 40^\circ$ .................................. 31
Figure 11. External velocity at $\beta = 43.4^\circ$ ................................ 32
Figure 12. External velocity at $\beta = 46^\circ$ .................................. 33
Figure 13. Displacement thickness on the lower surface ($\beta = 40^\circ$) .... 35
Figure 14. Displacement thickness on the lower surface ($\beta = 43.4^\circ$) .... 36
Figure 15. Displacement thickness on the lower surface ($\beta = 46^\circ$) .... 37
Figure 16. Displacement thickness on the upper surface ($\beta = 40^\circ$) .... 38
Figure 17. Displacement thickness on the upper surface ($\beta = 43.4^\circ$) .... 39
Figure 18. Displacement thickness on the upper surface ($\beta = 46^\circ$) .... 40
Figure 19. The effect of sharp trailing edge. ................................... 41
Figure 20. The results of the N. S. code at $\beta = 40^\circ$ ....................... 43
Figure 21. The results of the N. S. code at $\beta = 43.4^\circ$ .................... 44
Figure 22. The results of the N. S. code at $\beta = 46^\circ$ ....................... 45
Figure 23. Pratt & Whitney cascade .............................................. 47
Figure 24. Comparison of pressure coefficient. ................................ 48
Figure 25. Displacement thickness comparison .................................. 49
Figure 26. Velocity profile at 96.8% chord on the upper surface. .......... 50
Figure 27. C4 Cascade ................................................................. 52
Figure 28. C4 cascade at $\beta = 34.1^\circ$ ........................................ 53
Figure 29. C4 cascade at $\beta = 36.3^\circ$ ........................................ 54
Figure 30. C4 cascade at $\beta = 45.6^\circ$ ........................................ 55
Figure 31. C4 cascade at $\beta = 47.7^\circ$ ........................................ 56
Figure 32. C4 cascade at $\beta = 45.6^\circ$ ........................................... 58
Figure 33. C4 cascade at $\beta = 47.7^\circ$ ........................................... 59
Figure 34. C4 cascade at $\beta = 34.1^\circ$ ........................................... 60
Figure 35. C4 cascade at $\beta = 36.3^\circ$ ........................................... 61
ACKNOWLEDGEMENTS

My sincere appreciation to Professor M. F. Platzer for his guidance and professional advice in conducting this thesis. I would also like to express my appreciation to Mr. A. Krainer for his help in running the computer code and for his constructive suggestions during the conduct of this thesis.
I. INTRODUCTION

The need for better and more efficient gas turbines requires the availability of cheap and reliable design tools for blades used in compressors and turbines. Computational methods are the preferred choice for such a design tool, considering the cost and complexity of wind tunnel experiments.

Among the computational methods available today, the logical choice seems to be a computer code that can solve directly the full Navier Stokes equations. However, given the state of the art in both algorithms and computer hardware, such Navier Stokes solvers are restricted only to supercomputers, and even then the computation time is quite long.

In order to enable fast and efficient computations, the viscous inviscid interaction code was developed by Cebeci [Ref. 1]. The approach used in this code is to solve the outer flow field using potential methods, and solving the boundary layer flow using a boundary layer method subject to an interaction law, that couples the inner and the outer flows. This interaction law is needed because classical boundary layer methods fail in areas of flow reversal and separation, which are very common in real life flows.

The viscous inviscid interaction code was originally developed, and successfully used, for flows about single airfoils. It was later adapted to cascade flows.

In this thesis the applicability of the code to cascades was investigated by comparing its results to experimental data. It was found that although the code can reasonably predict experimental results in some cases, it still needs improvements before it can be applied generally as a reliable design tool.

A major restriction in improving the code is the lack of a wide data base of appropriate experimental results. Some of the key elements in the code, like transition and turbulence modelling, are based on empirical correlations, and more detailed and accurate experiments should be performed, before a better understanding of these phenomena can be achieved.

In the following, the theoretical background of the code is presented in Chapter II, a description of the code in Chapter III, the results and discussions are presented in Chapter IV and the conclusions and recommendations in Chapter V. A listing of the computer program is given in Appendix A.
II. CASCADE FLOW PROBLEM FORMULATION

This chapter outlines the theoretical background of the viscous inviscid interactive method used in the computer code to investigate cascade flows. Only the major steps in the mathematical developments will be described here. A detailed description of the theory and the numerical methods is given by Cebeci and Bradshaw [Ref. 2] and by Krainer [Ref. 3] on which this chapter is based.

A. INVISCID FLOW METHOD

Inviscid flow is the first building block of the flow and is solved using the panel method. The incompressible two dimensional outer flow must satisfy the Laplace equation:

\[ \nabla^2 \Phi = 0 , \]

subject to the boundary conditions on the surface of the blade:

\[ \frac{\partial \Phi}{\partial n} = v_w , \]

where the commonly used boundary condition of zero normal velocity on the surface is replaced by a specified blowing velocity \( v_w \) to allow for the effect of the boundary layer on the outer flow.

In addition, the Kutta condition must be satisfied, in order to prevent the existence of discontinuous pressure distribution near the trailing edge of the blade.

Since the Laplace equation is linear, a solution to a complex flow field can be built by superposition of solutions of elementary flows. The elementary flows used in the panel method are the uniform parallel flow and flows about singularities (sources and vortices).

The panel method is based on replacing each blade by a distribution of sources and vortices on its surface. The surface is divided into a finite number of straight segments, called panels.

On each panel, a uniform source distribution and a uniform vorticity distribution is assumed. The source strength at each panel is set to satisfy the boundary condition at the midpoint of the panel (called the control point), and so, in general the source
strength will vary from panel to panel. The vorticity strength is assumed to be the same for all the panels and is set to satisfy the Kutta condition.

The cascade is defined as an infinite row of similar blades, each one modelled by panels of source and vortex distributions. The flow at each point is found by summing the contributions of all the singularities on all the blades, and the uniform parallel flow. A useful concept in dealing with such flows is the concept of influence coefficients. An influence coefficient is defined as the velocity at a point induced by a unit strength singularity placed at some other field point. The influence coefficients are a function of geometry and so can be computed for a given cascade and a given choice of panel geometry.

Using the influence coefficients, the normal and tangential velocities at each control point can be written as a function of the unknown source strength of each panel and the unknown vortex strength. Using the boundary conditions, by equating the normal velocity at each control point to the prescribed blowing velocity \( v_\text{b} \), and using the Kutta condition (which requires equal velocities on the upper and on the lower panels at the trailing edge), a system of linear equations is constructed.

By solving the system of equations, the strength of the sources and vortices is found, and the velocities (and the pressures) can be computed everywhere in the flow field.

The velocity distribution on the surface of the blade, computed by the panel method, is used as the boundary condition for the boundary layer flow calculations.

It should be noted that in the panel method as used in the present computer code, there is no modelling of the wake, and its effect on the flow field is ignored.

B. VISCIOUS FLOW METHOD

Viscous flow is the second building block of the cascade flow and it is applied to the thin boundary layer near the blade surface.

1. Boundary Layer Theory

The boundary layer concept, first suggested by L. Prandtl, assumes that the flow field can be divided into an outer flow where the viscous effects are negligible compared to inertia effects, and a thin layer close to the surface where the viscous effects cannot be neglected. The complete presentation of the boundary layer theory, and the development of the boundary layer equations, is given by Schlichting [Ref. 4].

Under the assumptions of two dimensional incompressible thin boundary layer flow, the Navier Stokes equations and the continuity equation reduce to:
\[
\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} = 0 , \\
u \frac{\partial u}{\partial x} + v \frac{\partial u}{\partial y} = u_e \frac{du_e}{dx} + v \frac{\partial}{\partial y} \left( b \frac{\partial u}{\partial y} \right),
\]

with the boundary conditions:

\[
y = 0 \quad u(x,0) = 0 , \quad v(x,0) = 0 , \\
y = y_e \quad u(x,y_e) = u_e(x) ,
\]

where \( b \) denotes \( 1 + \frac{v_e}{v} \).

Writing the velocity components in terms of a stream function \( \Psi \):

\[
u = \frac{\partial \psi}{\partial y}, \\
v = -\frac{\partial \psi}{\partial x}.
\]

This eliminates the continuity equation (which the stream function satisfies by definition).

Introducing the Falkner Skan transformation:

\[
\eta = \sqrt{\frac{u_e}{v_x^2}} y, \\
f(x,y) = \frac{1}{\sqrt{u_e v_x}} \psi(x,y),
\]

the momentum equation and the boundary conditions transform to:

\[
(bf''')' + m + \frac{1}{2} f'''' + m[1 - (f')^2] = \eta \left( f' \frac{\partial f'}{\partial x} - f'' \frac{\partial f}{\partial x} \right),
\]

\[
\eta = 0 \quad f'(x,0) = 0 , \quad f(x,y) = 0 ,
\]
\[ \eta = \eta_e \quad f''(x, \eta_e) = 1, \]

where \( m \) is defined by:

\[ m = \frac{x}{u_e} \frac{du_e}{dx}. \]

The third order differential equation can be reduced to a system of first order differential equations by introduction of two new variables \( U \) and \( V \):

\[ U = f', \]
\[ V = U', \]

\[ (bV)' + \frac{m+1}{2} fV + m(1-U^2) = x \left( U \frac{\partial U}{\partial x} - V \frac{\partial f}{\partial x} \right), \]

with the boundary conditions:

\[ \eta = 0 \quad U(x,0) = 0, f(x,0) = 0, \]
\[ \eta = \eta_e \quad U(x, \eta_e) = 1. \]

The next step is to use a finite difference approach to solve the equations. The box method is applied using central differencing in both the \( x \) and \( \eta \) directions, and satisfying the equations midway between nodes.

Applying the box method results in a system of nonlinear equations in the unknown variables (which are \( f \), \( U \) and \( V \) in each node along the \( \eta \) direction at the current \( x \) station).

In order to solve the nonlinear system the Newton iterative procedure is used, linearizing the equations first about the solution at the adjacent upstream station, and then about the preceding iteration. The linearization is performed by letting:

\[ f_i^{j, \kappa} = f_i^{j, \kappa-1} + \delta f_i^{j, \kappa}, \]
\[ U_i^{j, \kappa} = U_i^{j, \kappa-1} + \delta U_i^{j, \kappa}, \]
\[ V_i^{j, \kappa} = V_i^{j, \kappa-1} + \delta V_i^{j, \kappa}, \]
where:
- \( i \) denotes location in the \( x \) direction
- \( j \) denotes location in the \( y \) direction
- \( \kappa \) indicates the iteration counter

This linearization results in a system of linear equations for the unknown increments: \( \delta f_j^{*\kappa} \), \( \delta U_j^{*\kappa} \) and \( \delta V_j^{*\kappa} \).

This system of equations is solved repeatedly until the changes in the unknowns are small enough. Since the system is block tridiagonal, Keller’s block elimination method is used.

The method described so far, is a direct boundary layer method. It can be used as long as the flow does not separate. Whenever separation or flow reversal occurs, and a zero skin friction coefficient is encountered, the equations become singular and the calculations will break down.

2. Interactive Boundary Layer Method

The interactive boundary layer method is designed to overcome the difficulties encountered at regions of flow reversal and separations. In such areas the external velocity is substantially changed by the viscous effects and can no longer be considered as a known boundary condition for the boundary layer flow.

The general approach to the solution is the same as for the direct method but, since the outer flow is unknown, the velocity at the edge of the boundary layer is written as:

\[
\begin{align*}
    u(x, y) &= u_{\text{el}}(x) + \frac{1}{\pi} \int \frac{d}{d\xi} (u_{\text{el}}') \left( \frac{d\xi}{x - \xi} \right) \\
    &\text{where:} \\
    &1. \ u(x, y) \text{ is the total velocity at the edge of the boundary layer.} \\
    &2. \ u_{\text{el}}(x) \text{ is the velocity as computed by the inviscid method.} \\
    &3. \ \frac{1}{\pi} \int \frac{d}{d\xi} (u_{\text{el}}') \left( \frac{d\xi}{x - \xi} \right) \text{ is the Hilbert integral.}
\end{align*}
\]
The numerical solution of the boundary layer equations follows the same steps as for the direct method, but with some changes.

The transformation of the stream function and the \( y \) coordinate uses a constant velocity \( u_0 \) as a scaling factor, and a scaled velocity \( w \) is introduced:

\[
\eta = \sqrt{\frac{u_0}{v\xi}} \cdot y ,
\]

\[
f(x, \eta) = \frac{1}{\sqrt{u_0v\xi}} \cdot \psi(x, \xi) ,
\]

\[
w = \frac{u_e(x, \xi)}{u_0}.
\]

Using this transformation, the boundary layer equations become a system of first order differential equations:

\[
f' = U ,
\]

\[
U' = V ,
\]

\[
W' = 0 ,
\]

\[
(bV)' + \frac{1}{2} fV + xW \frac{\partial W}{\partial x} = x\left(U \frac{\partial U}{\partial x} - V \frac{\partial f}{\partial x}\right) ,
\]

with the boundary conditions:

\[
\eta = 0 \quad U(x,0) = 0 \quad f(x,0) = 0 ,
\]

\[
\eta = \eta_e \quad U(x, \eta_e) = W(x, \eta_e) ,
\]

\[
w(x, \eta_e) = \frac{u_e(x)}{u_0} + \frac{1}{\pi} \int \frac{d}{d\xi} \left(\sqrt{\frac{v\xi}{u_0}} [W(\xi, \eta_e) - f(\xi, \eta_e)]\right) \frac{d\xi}{x - \xi} .
\]

The finite difference box method is used to solve the equations, in the same way as it was used for the direct case, but with two additions:
1. In areas of flow reversal the term $u\partial u/\partial x$ is omitted to assure stable integration (the FLARE approximation).

2. The edge velocity, $W_j$ (where J denotes the edge station) which involves integration, is approximated by:

$$W_j^i = g_i + c_i(W_{ij}^i - f_j) ,$$

where $g_i$ and $c_i$ are obtained from the numerical approximation to the Hilbert integral (which will be presented in the next section).

By using central differencing to approximate the differential equations, a system of nonlinear algebraic equations is obtained for the unknown variables (which are $f_j$, $U_j$, $V_j$ and $W_j$). To solve the system of equations, the system is linearized by the Newton iterative procedure, and the resulting linear system is solved (for the new unknown variables which are the increments $\delta f_j^*$, $\delta U_j^*$, $\delta V_j^*$ and $\delta W_j^*$).

The solution of the system is repeated until the change in the increments is negligible compared to the preceding iteration, and the whole process is performed again at the next downstream station.

3. Interactive Model

The interactive model is used to couple the boundary layer to the external flow. It is needed in areas where strong interaction occurs, and both the boundary layer and the outer flow must be solved simultaneously. The interaction model provides the outer boundary condition to the boundary layer calculations by adding a correction term to the external velocity computed by the inviscid flow method.

The external velocity is assumed to consist of a potential flow term ($u_{el}(x)$) and a correction term due to viscous effects ($u_{ed}(x)$):

$$u_e(x) = u_{el}(x) + u_{ed}(x).$$

The viscous effect is obtained by a surface distribution of sources on the blade (a concept first suggested by Lighthill [Ref. 5]). The normal velocities at the surface of the blade, induced by these sources, displace the streamlines from the surface in the same way that the actual boundary layer displaces them:

$$\frac{d\delta^*(x)}{dx} = \frac{v(x, \delta^*)}{u_e(x)} ,$$

Where $v(x, \delta^*)$ is the normal velocity at the displaced surface.
Assuming that the surface can be approximated to be a flat plate, the normal velocity will be half the local source strength $\sigma(x)$. Assuming also that the inviscid velocity does not change across the boundary layer, the local source strength will be:

$$\frac{\sigma(x)}{2} = v(x, 0) = v(x, \delta^*) - \int_0^{\delta^*} \frac{\partial v}{\partial y} dy = \frac{d}{dx} (u_e \delta^*) .$$

The local horizontal velocity induced by the source distribution, is the correction term to the inviscid velocity, and can be represented by the Hilbert integral:

$$\frac{1}{\pi} \int_{x_0}^{x_b} \frac{\sigma(\xi)}{x - \xi} d\xi = \frac{1}{\pi} \int_{x_0}^{x_b} \frac{d}{d\xi} (u_e \delta^*) \frac{d\xi}{x - \xi} .$$

The integration is carried out on all the sources on the surface, since the horizontal velocity is influenced by all the sources.

The Hilbert integral is then approximated by a finite series:

$$\frac{1}{\pi} \int_{x_0}^{x_b} \frac{d}{d\xi} (u_e \delta^*) \frac{d\xi}{x - \xi} = \sum_{k=1}^{K} c_{ik} (u_e \delta^*)^k .$$

Where $c_{ik}$ is a matrix of interaction coefficients which are functions of the geometry only ($i$ denotes the chordwise position where $u_{e,i}$ is evaluated and $k$ is the location of the source which effects $u_{e,i}$).

Since the computation of $u_{e,i}$ involves values of $\delta^*$ downstream of the current $x$ location, which are not known yet, these terms are taken from the previous iteration using a relaxation formula.

4. Turbulence Model

The turbulence model used here is the algebraic eddy viscosity formulation of Cebeci and Smith [Ref. 6]. According to the model used in the present computer code, the eddy viscosity $v_i$ is defined by two different expressions, for the inner region and for the outer region:
\[ v_i = \begin{cases} 
0.4 y \left[ 1 - \exp\left( -\frac{y}{A} \right) \right] \left[ \frac{\partial u}{\partial y} \right] y_{ii} & \text{for } 0 \leq y \leq y_c, \\
\alpha \int_0^\infty (u_e - u) dy y_{ii} & \text{for } y_c \leq y \leq \delta.
\end{cases} \]

Where:
\[
A = \frac{26v}{\left( \nu \frac{\partial u}{\partial y} \right)^{1/2}} ,
\]
\[
\gamma = \frac{1}{1 + 5.5(y'/\delta)^6} ,
\]
\[
\alpha = \frac{0.0168}{1 - \beta \left[ \frac{\partial u' \partial x}{\partial u' \partial y} \right]^{2.5}} ,
\]
\[
\beta = \frac{6}{1 + 2R_T(2 - R_T)} \quad \text{for } R_T < 1 ,
\]
\[
\beta = \frac{1 + R_T}{R_T} \quad \text{for } R_T \geq 1 ,
\]
\[
R_T = \frac{\tau_w}{(-\bar{u}'\bar{v}')}_{\max} .
\]

The distance from the wall to the point between the two regions, \( y_c \), is chosen such that the viscosity will be continuous.
The intermittency factor, $\gamma_{tr}$, is defined by:

$$
\gamma_{tr} = 1 - \exp\left[-\frac{u_e^3}{G_v v^2} R_{xtr}^{-1.34} (x - x_{tr}) \int_{x_{tr}}^x \frac{d\xi}{u_e}\right].
$$

Where:
- $R_{xtr}$ is the Reynolds number based on external velocity and transition location.
- $G_v$ is an empirical constant, originally assigned the value 1200.

Cebeci and Bradshaw [Ref. 2, p.246] described a different expression for the variable $A$ in the inner region viscosity formula:

$$
A = \frac{26v}{(1 - 11.8 p^-)^{1/2} (v \frac{\partial u}{\partial y})_{max}^{1/2}}.
$$

Where:

$$
p^- = \frac{vu_e}{(v \frac{\partial u}{\partial y})^{3/2}} \frac{du_e}{dx}.
$$

This version of the turbulence model was not implemented in the original computer code. During the work on this thesis, the effect of the modified turbulence model was investigated.

A different intermittency distribution was implemented successfully by Rodi and Schonung [Ref. 7] for transition over separation bubbles. They used for $G_v$ the expression:

$$
G_v = \frac{100}{\exp(0.99Tu)}.
$$

Where $Tu$ is the turbulence level in the free flow. This intermittency model was also investigated during the work on this thesis.
5. Transition

The prediction of transition from laminar to turbulent flow is very difficult and has to rely on empirical correlations. The relation used here to predict the onset of transition is a combination of Michel’s method and the $e^9$ method, and is given by Cebeci and Bradshaw [Ref. 2, p. 153]:

$$R_{\theta_{ir}} = 1.174 \left( 1 + \frac{22400}{R_{e_{ir}}} \right) R_{e_{ir}}^{0.46}.$$  

Where:

1. $R_{\theta_{ir}}$ is the Reynolds number based on the momentum thickness at the onset of transition.
2. $R_{e_{ir}}$ is the Reynolds number based on $x$ at the onset of transition.

In the computer code, if a laminar separation is detected before transition occurs, the onset of transition is assumed at the point of laminar separation.
III. DESCRIPTION OF THE COMPUTER CODE

The computer code used here to investigate cascade flows was written by Cebeci, and is based on the numerical formulation that was outlined in the previous chapters. In this chapter the general structure and the major subroutines of the code will be described.

A. GENERAL STRUCTURE OF THE MAIN PROGRAM

The main program reads in the cascade data (blade coordinates, spacing and stagger angle), the flow data (inlet angle and Reynolds number), and transition parameters. The transition onset on each surface of the blade can be computed by the program, or can be input by the user. The intermittency parameter $G$ should be specified by the user.

The program then calls subroutine POTNL to compute the outer inviscid flow field for the first cycle. The output of subroutine POTNL is the external velocity distribution on the surface of the blades. This velocity distribution is then transferred to subroutine CASBLP, which calculate the boundary layer flow.

Subroutine CASBLP returns the displacement thickness distribution and the blowing velocity distribution on the blades to the main program. This data is then transferred back to subroutine POTNL to the next cycle of calculations.

The program repeats the cycles of calculations by calling the two subroutines, until the specified number of cycles is reached, or until a convergence criterion is satisfied.

B. DESCRIPTION OF THE SUBROUTINES

1. Subroutine POTNL

This subroutine solves the inviscid outer flow by using the panel method. The subroutine calculates the influence coefficients and calculates the velocities subject to the boundary conditions.

The velocities are evaluated on the displaced surface (the surface created by adding the displacement thickness to the original surface of the blade). The input to this subroutine includes the cascade geometry, the blowing velocity and the displacement thickness (for the first cycle both the displacement thickness and the blowing velocity are taken to be zero).
2. Subroutine CASBLP

This subroutine, called by the MAIN program, receives the blade geometry and the velocity distribution as input.

It transforms the x,y blade coordinates to the chordwise tangential coordinates and smooths the velocity data (during the work on this thesis it was found that smoothing the velocity data prevents the detection of the separation bubble near the leading edge, and therefore it was eliminated). The subroutine then calls subroutine COMPBL for further calculations.

3. Subroutine COMPBL

This subroutine finds the stagnation point and controls the generation of the boundary layer calculation grid for each surface (the grid starts at the stagnation point and includes 91 points in the chordwise direction for the upper surface and 71 points on the lower surface).

The subroutine then calls subroutine BL2D which calculates the boundary layer parameters for each surface (BL2D is called twice, first for the upper surface and then for the lower surface).

4. Subroutine BL2D

This subroutine computes the displacement thickness and the blowing velocity and returns them back to the calling subroutine (COMBL) in arrays compatible with the potential flow calculations (one array that contains all the points of the blade, first the lower surface starting at the trailing edge and proceeding forward, and then the upper surface, starting at the leading edge and proceeding backwards).

BL2D calls the following subroutines:

1. Subroutine INPUT which calculates the following:
   a. NS, the switching point between direct and interactive boundary layer calculations (this point is set at the first pressure peak when the blade is scanned from leading edge towards the trailing edge)
   b. NTR, transition location (only if the transition location is an input. Otherwise it is calculated by subroutine TRNS).
   c. GMTR, the distribution of the intermittency factor $\gamma_{tr}$.

   In addition this subroutine generates the boundary layer grid in the $\eta$ direction and the initial velocity profile, by calling subroutine INTL.

2. CALCIJ, calculates the $c_n$ coefficients used in the Hilbert integral approximation.

3. EDDY, calculates the eddy viscosity (called only after transition has been detected).
4. COEFTR, calculates the coefficients of the boundary layer finite difference equations in transformed form (for the direct method calculations).

5. SOLVE3, solves the linearized boundary layer equations for the F, U and V variables by computing the increments \( \delta F, \delta U \) and \( \delta V \).

The subroutine then checks the convergence of the Newton iterations and repeats the calculations if needed. If the subroutine detects flow separation or if it reaches the switching point NS, subroutine MAIN2 is called for the interactive method calculations. Otherwise, the subroutine proceeds to the next chordwise point of the grid (NX) and repeats the calculations.

5. Subroutine MAIN2

This subroutine calculates the boundary layer parameters by the interactive method. The subroutine performs the following steps:

1. It first calls subroutines JOINT and COMGI to compute the interaction coefficients.

2. In regions of laminar flow it calls the following subroutines:
   a. COEF, which calculates the coefficients of the boundary layer finite differences equations.
   b. SOLV4, solves for the variables F, U, V and W by computing the increments \( \delta F, \delta U, \delta V \) and \( \delta W \).
   c. TRANS, to check if the condition for transition is satisfied (it also checks for laminar separation and initiates transition at the point of laminar separation if it is detected).

The subroutine then checks for convergence of Newton iterations and repeats the calculations as needed.

3. In regions of turbulent flows the subroutine calls the following subroutines:
   a. EDDY, to compute the eddy viscosity parameter \( B = 1 + \nu_r/\nu \).
   b. COEF and SOLV4, the same as for laminar flow.

6. Subroutine OUTPUT

This subroutine computes the boundary layer parameters. It is called with a parameter "INDEX" which determines the type of calculations:

1. For INDEX = 1 the computations relates to transformed coordinates (direct boundary layer method) using the relations:

\[
\gamma_f = \frac{2 V(1.2) B(1,2)}{\sqrt{R}_{ex}}
\]
\[ V_w = u_e \sqrt{\frac{u_e}{x}} V(1,2), \]
\[ D = u_e \delta^* \sqrt{R_e}, \]
\[ \delta^* = \frac{x}{\sqrt{R_{ex}}} (\eta(NP) - F(NP)), \]
\[ \theta = \frac{x}{\sqrt{R_{ex}}} \sum_{j=2}^{NP} a_j U_j (1 - U_j). \]

Where \( V(1,2) \) and \( B(1,2) \) are the velocity gradient and the viscosity parameter at the surface, respectively, and \( \eta(NP) \) and \( F(NP) \) are \( \eta \) and \( F \) evaluated at the edge of the boundary layer.

2. For \( INDEX = 2 \) the subroutine calculates the boundary layer parameters for semi-transformed coordinates (interactive boundary layer method) using the relations:

\[ C_f = \frac{2 V(1,2) B(1,2)}{\sqrt{x} \sqrt{R_e} [W(NP)]^2}, \]
\[ V_w = \frac{V(1,2)}{\sqrt{x}}, \]
\[ u_e = U(NP), \]
\[ D = (U \eta - F) \sqrt{x}, \]
\[ \delta^* = \left( \eta - \frac{f}{U} \right) \sqrt{x}. \]

For \( NX > NTR \) (after transition has been detected), subroutine SMPSON is called (subroutine SMPSON calculates the coefficient of the outer region eddy viscosity). The subroutine then prints out the velocity profiles at the required stations.

7. Subroutine TRANS

This subroutine calculates the transition location based on the Michel criterion or based on laminar separation (whichever occurs first). If transition has been detected the intermittency distribution is calculated for all the remaining points of the surface.

8. Subroutine FILLUP

This subroutine increases the number of points in the boundary layer grid (in the \( \eta \) direction) as needed. It also fills up the arrays of \( F, U, B, W \) and \( V \) between the edge of the boundary layer to the end of the arrays (with \( V = 0, W, B \) and \( U \), with the last values that they had in the edge of the boundary layer and \( F \) as the integral of \( U \)).
9. **Subroutine EDDY**

   This subroutine calculates the eddy viscosity using the Cebeci-Smith two layer eddy viscosity formula. It receives the vectors $U,V$ and $\eta$ at a point and computes the viscosity vector $B$.

10. **Subroutine INTL**

   This subroutine generates the boundary layer grid in the $\eta$ direction. It sets the number of grid points and generates the initial velocity profile.
IV. RESULTS AND DISCUSSION

The viscous inviscid interaction code was run with several cascades on which experimental data is available. In order to enable a thorough comparison between experimental results and the computed results, a very detailed experimental data base is needed. The data should include measurements of the boundary layer development along the blade, velocity profiles along the boundary layer, transition location and distribution, flow separation, and external velocity distribution.

Unfortunately, very few cascade experiments has been performed, which obtained the required data with sufficient accuracy, due mostly to the lack of appropriate measurement equipment. Only recently, with the introduction of non-interfering methods like the Laser Doppler Velocimeter (LDV), the required data can be measured accurately.

Recently an experiment involving the investigation of a linear compressor cascade of Controlled Diffusion Blading (which will be referred here as the CD cascade) has been carried out by Elazar [Ref. 8]. Most of the work in the present thesis, involves comparison of the computer code results with Elazar’s experimental results.

Other cascades that were investigated are:

1. A shockless, supercritical airfoil cascade, designed in 1974 by Korn in cooperation with Pratt & Whitney Aircraft (referred here as the P & W cascade). The experimental results of the cascade were obtained from a report by Hobbs, Wagner, Dannenhoffer and Dring [Ref. 9].

2. Stator blade of a single stage axial compressor (referred here as the C4 cascade). The blade profile is the British C4 section (10% thickness) on a circular arc camber line. The experiment has been performed by Walker [Ref. 10]. The detailed boundary layer measurements are not presented in the report and were obtained directly from the author.

The code failed to run with two other cascades:

1. A highly loaded, double circular arc blade with a sharp leading edge and a sharp trailing edge, used in a compressor cascade that was investigated by Deutsch and Zierk [Ref. 11].

2. V2 double circular arc blade, highly loaded cascade. This cascade was investigated by Hoheisel and Seyb [Ref. 12].

In both cases the code calculated the potential flow successfully but failed in trying to compute the first cycle of the boundary layer calculations.
A. CD CASCADE

The experimental data for the CD cascade was obtained at $M = 0.25$, $R_e = 700000$ and at three inlet angles: 40° (the design condition), 43.4° and 46°. The spacing was 0.6 of the chord and the stagger angle 14.27°. A general layout of the cascade is shown in Figure 1 on page 20.

The following observations were concluded from the experiment:

1. A separation bubble exists near the leading edge on the upper surface at all the inlet angles. The bubble became larger at increased inlet angles.
2. Transition from laminar to turbulent flow occurred above the separation bubble (on the upper surface).
3. Transition on the lower surface occurred at midchord.
4. The boundary layer thickness on the upper surface increased with inlet angle, and reached a thickness of 15% chord at the highest inlet angle. The boundary layer thickness on the lower surface did not change significantly with inlet angle.
5. The turbulent boundary layer on both surfaces remained fully attached at all the inlet angles.

1. Transition location and intermittency distribution

The effects of the transition location and the intermittency factor were investigated. The code was first run with the transition location calculated by the code, and with several values of the intermittency factor $G_r$. It was found that the code did not run with $G_r = 1200$ (which is the value used usually for high Reynolds numbers). The highest value of $G_r$ with which the code run successfully was 900.

The code failed to predict the separation bubble on the upper surface, and predicted laminar separation at 78% chord on the lower surface (which did not occur in the experiment). Transition on the upper surface occurred at 41% chord (detected by Michel's criterion) and at 78% chord on the lower surface (at laminar separation).

The shape factor computed by the code was compared to the experimental results. As can be seen in Figure 2 on page 21 the shape factor as predicted by the code deviates substantially from the actual results, due mainly to the different transition location.

On the lower surface, as can be seen in Figure 3 on page 22 the shape factor deviates even more from the experimental results. In this figure the effect of changing the intermittency factor $G$, can be seen. For both the extreme values of $G$, 10 and 900, the computed shape factor curve is far from agreement with the actual results.
Figure 1. Controlled Diffusion cascade
Figure 2. Shape factor comparison on the upper surface: Transition computed by the code ($\beta = 40^\circ$)
Figure 3. Shape factor comparison on the lower surface: Transition computed by the code ($\beta = 40^\circ$).
Since the code did not predict the existence of the separation bubble on the upper surface, it was decided to try to eliminate the smoothing of the external velocity as computed by the potential flow subroutine (originally, the velocities were smoothed in subroutine CASBLP prior to boundary layer calculations). It was found that without smoothing the velocities a small separation bubble is predicted by the code at 4% of chord. The onset of transition is set by the code at the beginning of the separation bubble.

The code was run with unsmoothed velocities with two values of $G$, 10 and 900. The shape factor behavior can be seen in Figure 4 on page 24. Changing the value of $G$, did not change the shape of the curve much, and generally the shapes of the computed and the experimental curves look alike.

The elimination of the velocity smoothing in the code, also affects the thickness of the boundary layer. In Figure 5 on page 25 the displacement thickness is plotted for both cases (with and without the velocity smoothed). Without smoothing, the displacement thickness is much thicker, especially on the rear half of the blade, which is closer to the actual results.

The effect of changing the intermittency distribution to the one used by Rodi and Schonung [Ref. 7] was investigated. It was found, as can be seen in Figure 6 on page 26 that the effect of the new model is equivalent to using $G$, in the present model.

On the lower surface it was necessary to run the code with transition as input, to get reasonable results, as can be seen in Figure 7 on page 27 for transition input at 21% of chord.

At the off design conditions (inlet angles of 43.4° and 46°) a similar behavior of the transition has been observed, as can be seen for example in Figure 8 on page 28 for the upper surface and in Figure 9 on page 29 for the lower surface, both at $\beta = 46^\circ$. 
Figure 4. Shape factor on the upper surface without velocity smoothing.
Figure 5. Displacement thickness: The effect of velocity smoothing.
Figure 6. The effect of the intermittency model: Upper surface, $\beta = 40^\circ$
Figure 7. Shape factor on the lower surface with transition input at 21%.
Figure 8. Shape factor at $\beta = 46^\circ$ on the upper surface
Figure 9. Shape factor at $\beta = 46^\circ$ on the lower surface.
2. External Velocity Distribution

The external velocity distribution, computed by the code using the interaction law, was compared to the experimentally measured velocities. It was found that in general, the velocities measured experimentally, were higher than those computed by the code for all the inlet angles.

There are two possible sources to the discrepancy in the velocities:

1. The computer code calculates pure 2-D flows. In the experiment the flow was observed to accelerate due to the effect of the boundary layer on the side walls (a 3-D effect). This effect was calculated in the experiment and is referred to as the AVDR correction [Ref. 8, p. 43].

2. The flow accelerates due to the thickening of the boundary layer. Since the boundary layer as computed by the code is substantially thinner than the actual boundary layer (as will be discussed in the next section) the external velocities predicted by the code are smaller.

To compensate for the first error source, all the computed velocities were compared to the experimental velocities corrected by the AVDR correction. The comparison between the velocities can be seen in Figure 10 on page 31 for $\beta = 40^\circ$, in Figure 11 on page 32 for $\beta = 43.4^\circ$ and in Figure 12 on page 33 for $\beta = 46^\circ$.

It can be seen from the figures that the difference between the computed and the experimental velocities is larger on the lower surface. The reason might be the method with which the correction to the inviscid velocity is computed. The assumption on which the interaction law is based, is that only sources (representing the viscous effects) on the surface being considered, affect the local velocity. In reality, the boundary layer on both surfaces affects the local velocity (because the boundary layer developed on the upper surface of a blade, causes a velocity disturbance that is felt on the lower surface of the adjacent blade).

Since the boundary layer on the lower surface is much thinner, its effect on the velocity on the upper surface is much smaller than the effect of the upper surface boundary layer on the lower surface velocity.
Figure 10. External velocity at $\beta = 40^\circ$
Figure 11. External velocity at $\beta = 43.4^\circ$
Figure 12. External velocity at $\beta = 46^\circ$
3. **Boundary Layer Thickness**

The boundary layer thickness as computed by the code was compared with the experimental results by comparing the displacement thicknesses.

It was found that on the lower surface the computed and the actual displacement thickness agree quite well, as can be seen in Figure 13 on page 35 for $\beta = 40^\circ$, Figure 14 on page 36 for $\beta = 43.4^\circ$ and in Figure 15 on page 37 for $\beta = 46^\circ$.

On the upper surface the displacement thicknesses computed by the program are significantly thinner than those measured experimentally. The difference between the computed and the actual thickness increases along the blade and it increases with increased inlet angle. It was found that by using a different expression for the inner region eddy viscosity (as mentioned in chapter II), the displacement thickness can be increased, but the difference between the actual and the computed thickness is still substantial, especially at the higher inlet angles. Figure 16 on page 38, Figure 17 on page 39 and Figure 18 on page 40 shows the displacement thickness on the upper surface for the three inlet angles, with the original and the modified eddy viscosity models.

The large error in the prediction of the boundary layer thickness, can be the result of several reasons:

1. The transition model used in the code, sets the onset of transition at the first point of laminar separation. It causes rapid transition to turbulent flow which reattaches immediately, resulting in a very small separation bubble compared to the bubble observed in the experiment.

2. The turbulent model used in the code could be inaccurate. It was derived based on empirical data obtained in single airfoil experiments and not with cascades. In addition the present model does not include the effects of the free stream turbulence (that was relatively high in the experiment).

3. The boundary layer as measured in the experiment was quite thick, especially at the higher inlet angles (it reached 15% of the chord at $\beta = 46^\circ$). Such a thick boundary layer may violate the basic assumptions on which the boundary layer equations, and the interaction law, were based (especially when the spacing between the blades is small, 60% chord in this case).

It was suggested that one of the possible reasons to the inaccurate prediction of the boundary layer is the blunt trailing edge of the blade, that might cause difficulties in the computations. A modified blade, with a sharp trailing edge has been run, and the displacement thickness distribution can be seen in Figure 19 on page 41. As can be seen in the figure the sharp trailing edge affects only the boundary layer adjacent to the trailing edge, and therefore cannot provide an explanation to the difference between the actual and the computed displacement thickness.
Figure 13. Displacement thickness on the lower surface ($\beta = 40^\circ$)
Figure 14. Displacement thickness on the lower surface ($\beta = 43.4^\circ$)
Figure 15. Displacement thickness on the lower surface ($\beta = 46^\circ$)
Figure 16. Displacement thickness on the upper surface ($\beta = 40^\circ$)
Figure 17. Displacement thickness on the upper surface ($\beta = 43.4^\circ$)
Figure 18. Displacement thickness on the upper surface ($\beta = 46^\circ$)
Figure 19. The effect of sharp trailing edge.
4. Comparison to a Navier Stokes Code

A limited comparison of the experimental and the computed results with a Navier Stokes (N.S.) code calculations has been performed. The N.S. code has been developed and run by S. J. Shamroth of Scientific Research Associates inc. in cooperation with Pratt and Whitney Aircraft.

Since the N.S. code does not compute the displacement thickness, the velocity profiles near the surface of the blade were compared. The comparisons were made at 90% chord on the suction surface for all three inlet angles.

At the design point, $\beta = 40^\circ$, shown in Figure 20 on page 43, both the interactive code and the N.S. code failed to predict accurately the actual velocity profile. In this case the interactive code seems to yield somewhat better results than the N.S. code.

At the higher inlet angles, $\beta = 43.4^\circ$ and $\beta = 46^\circ$, shown in Figure 21 on page 44 and in Figure 22 on page 45 respectively, the N.S. calculations show significantly better agreement with the experimental results than the interactive code.

From these comparisons, it can be seen that the interactive code deviation from the actual results increases with increased inlet angle (increased loading of the cascade), whereas the N.S. code deviation seems to decrease with increased inlet angle.
Figure 20. The results of the N. S. code at $\beta = 40^\circ$
Figure 21. The results of the N. S. code at $\beta = 43.4^\circ$
Figure 22. The results of the N. S. code at $\beta = 46^\circ$
B.  P & W CASCADE

The experimental data for the P & W cascade was obtained at inlet flow angle of 52°, at M = 0.11, and Reynolds number of 478000. The cascade had a stagger angle of 15.75° and 0.7 spacing. A general layout of the cascade is shown in Figure 23 on page 47.

A comparison of the computed and the measured pressure coefficients on the blade is shown in Figure 24 on page 48. There is a good agreement between the computed and the measured $C_p$.

The displacement thickness was measured in the experiment only at 96.8% of chord. This measurement is compared to the computed results in Figure 25 on page 49. As can be seen, the computed and the measured data agree almost perfectly on the lower surface, and quite well on the upper surface. The difference observed on the upper surface is caused by the early prediction, by the code, of trailing edge separation, a short distance upstream of the actual location. This can also be observed when comparing the velocity profiles at that point, in Figure 26 on page 50. The computed velocity curve shows a small zone of reversed flow near the surface of the blade. This reversed flow could be the result of a too early prediction of trailing edge separation by the code, or it could have existed in the actual flow but not detected because of its size.
Figure 23. Pratt & Whitney cascade
Figure 24. Comparison of pressure coefficient.
Figure 25. Displacement thickness comparison: Experimental data shown at 96.8% chord.
Figure 26. Velocity profile at 96.8% chord on the upper surface.
C. C4 CASCADE

The C4 cascade has a stagger angle of 29.5°, a camber angle of 31.1° and spacing of 0.992. It has been tested at Reynolds numbers of about 200000, and inlet angles of 34.1° to 47.7° (which corresponds to incidence angles of -10.9° to 2.7°). The general layout of the cascade is shown in Figure 27 on page 52. A computer code that generates the coordinates of the blade and a summary of some experimental results are given in Appendix B.

The code was run with the intermittency constant $G_s = 10$. Higher values of $G_s$ (above 100) caused numerical problems in the code. The onset of transition was first taken at the point where it was observed in the experiment. At the lower inlet angle it seems that a better agreement with the experimental results can be obtained by delaying the onset of transition but trying to implement it resulted in numerical breakdown of the computation. At the higher inlet angles, better agreement with the experimental results was achieved by initiating the transition earlier (at 26% chord for $\beta = 45.6°$ and at 21% for $\beta = 47.7°$ as compared to 44% and 36% chord as observed in the experiment).

1. Displacement Thickness

Comparisons of the experimental data to the computed displacement thickness are shown in Figure 28 on page 53 for inlet angle of 34.1°, in Figure 29 on page 54 for inlet angle of 36.3°, in Figure 30 on page 55 for inlet angle of 45.6° and in Figure 31 on page 56 for inlet angle of 47.7°.

As can be seen in the figures, there is a good agreement between the actual and the computed results at the two lower angles ($\beta = 34.1°$ and $\beta = 36.3°$, in which the incidence angles were negative). At the two higher angles, $\beta = 45.6°$ and $\beta = 47.7°$ the computed results agree with the actual results up to about 70% chord, and then the displacement thickness predicted by the code becomes much thicker than the actual one.

The code predicted a large flow separation area starting at about 70% chord at the lower inlet angles, and at about 46% chord at the higher inlet angles. This flow separation was not observed in the experiment. The discrepancies between the computed and the actual results behind 60% to 70% chord can be explained by the inaccurate calculations by the code due to the large separated areas. When the code encounters separation, several approximations are made (like the FLARE approximation) based on the assumption that the separated area is small. When the separated area is large, these approximations may result in inaccurate prediction of the flow field.
Figure 27. C4 Cascade
Figure 28. C4 cascade at $\beta = 34.1^\circ$: Displacement thickness comparison with computed results.
Figure 29. C4 cascade at $\beta = 36.3^\circ$: Displacement thickness comparison with computed results.
Figure 30. C4 cascade at $\beta = 45.6^\circ$: Displacement thickness comparison with computed results ($G = 10$).
Figure 31. C4 cascade at $\beta = 47.7^\circ$: Displacement thickness comparison with computed results ($G = 10$).
2. External Velocity and Velocity Profiles Comparisons

A comparison of the external velocity on the upper surface of the blade is shown in Figure 32 on page 58 for inlet angle of 45.6° and in Figure 33 on page 59 for inlet angle of 47.7°. It can be seen that there is a good agreement between the experimental and the computed results up to about 80% chord. Near the trailing edge the computed results deviate from the experimental results due to the inaccuracy in the calculations of the displacement thickness.

A comparison of the velocity profiles in the boundary layer at 50% chord is shown in Figure 34 on page 60 for inlet angle of 34.1° and in Figure 35 on page 61 for inlet angle of 36.3°. The agreement between the calculated velocity profiles and the measured velocity profiles is very good.
Figure 32. C4 cascade at $\beta = 45.6^\circ$: External velocity distribution.
Figure 33. C4 cascade at $\beta = 47.7^\circ$: External velocity distribution.
Figure 34. C4 cascade at $\beta = 34.1^\circ$: Velocity profile at 50% chord.
Figure 35. C4 cascade at $\beta = 36.3^\circ$: Velocity profile at 50% chord.
V. CONCLUSIONS AND RECOMMENDATIONS

A. CONCLUSIONS

The interactive viscous inviscid computer code, has been investigated by comparing its predictions of boundary layer parameters to experimental data.

It has been found that the code yields reasonable results for lightly loaded cascades, but the prediction of the boundary layer thickness on the suction surface of highly loaded cascades deviates significantly from experimentally measured data. In two cases involving highly cambered cascade blades, with sharp leading edge, the code failed to run. It was also found that the prediction of external velocity distribution on highly loaded cascades was inaccurate.

The main reasons to the discrepancies in the prediction of the boundary layer thickness seem to be:

1. Inaccuracy in predicting flow parameters in regions of large flow separation (due to inadequate transition model and approximations made in calculating the flow in separated areas).
2. Inaccurate turbulence modelling.
3. Possible violation of the basic assumptions of the boundary layer theory in areas of very thick boundary layers.
4. The wake is not calculated by the code. The result is inaccurate flow prediction near the trailing edge.

The inaccuracy in the prediction of the external velocity distribution in highly loaded cascades is due to the interaction law, which does not account for the presence of adjacent blades.

B. RECOMMENDATIONS

The recommended steps in order to improve the code are:

1. Improving the interaction law by assuming a distribution of sources on the actual surface (instead of the assumption of a flat plate), letting the correction term to the external velocity vary across the boundary layer and distributing sources on the adjacent blade as well for better modelling of the boundary layer effect on the external velocity.
2. Changes to the derivation of the boundary layer equations should be investigated to allow a better treatment of thick boundary layers (like omitting the assumption of $\frac{\partial P}{\partial y} = 0$ across the boundary layer).

3. Different turbulence models should be investigated.

4. The wake should be included in the calculations.
APPENDIX A. COMPUTER CODE LISTING

C**********************************************************************INT001
C** VISCOUS-INVISCID INTERACTION PROGRAM FOR CASCADE FLOWS **********INT002
C**********************************************************************INT003
C
C VERSION 3.0
JANUARY 87
C
C THIS VISCOUS-INVISCID INTERACTION METHOD, CAPABLE OF COMPUTING BOTH
C SINGLE AIRFOIL AND CASCADE FLOWS, WAS DEVELOPED BY CEPECI AND
C COLLABORATEURS AT LONG BEACH STATE AND DOUGLAS AIRCRAFT COMPANY.
C THE CODE APPLIES TO INCOMPRESSIBLE, 2-DIMENSIONAL, STEADY FLOWS
C PAST LINEAR, ARBITRARILY STAGGERED CASCADES. THE METHODS BASIC
C INGREDIENTS INCLUDE
C 1. A FIRST ORDER PANEL METHOD TO SOLVE LAPLACE'S EQUATION,
C 2. A FINITE DIFFERENCE SCHEME TO SOLVE THE BOUNDARY LAYER EQUATIONS
C SUBJECT TO DIRECT OR INTERACTIVE BOUNDARY CONDITIONS,
C 3. A STRONG INTERACTION MODEL TO COUPLE VISCOUS AND INVISCID FLOW
C RESULTS, AND
C 4. A ZERO EQUATION, ALGEBRAIC TURBULENCE MODEL TO ESTIMATE
C TURBULENT SHEAR STRESSES.
C
C IN SUMMARY, THE CODE WILL PROVIDE, FOR ATTACHED AS WELL AS MODERATE-
C LY SEPARATED FLOWS PAST SINGLE AIRFOILS OR CASCADES, THE FOLLOWING
C 1. INVISCID AND VISCOUS PRESSURE DISTRIBUTIONS,
C 2. DISTRIBUTIONS OF
C   A. LOCAL SKIN FRICTION COEFFICIENT,
C   B. DISPLACEMENT AND MOMENTUM THICKNESS, AND
C 3. VELOCITY PROFILES ACROSS THE BOUNDARY LAYER.
C
C MODIFICATIONS SINCE VERSION 3.0:
C 1. PRECISE ASSIGNMENT OF BEGIN OF TRANSITION.
C 2. CORRECTION OF AN ERROR IN THE CALCULATION OF MOMENTUM THICKNESS.
C 3. ADDITIONAL PRINT OPTION: IP=-2 WILL PROVIDE AN INPUT FILE (UNIT
C NUMBER 12) FOR THE PLOTTING ROUTINE.
C
C COMMON /BLCO/ NX,NXT,NP,NTR,IT,INVRS,NS,IP
C COMMON/BLIN/ TITLE(20),XC(100),YC(100),ISG(100),DELS(100),
C XCTR,XTR,ISTRP,ICYCLE,ICYTL,XCTRS(2),TRFIND(2)
C COMMON/CASCADE/INLET,SP,SINGLE,ALPHAA,ALPHA1,STAG
C COMMON/TRN/ PGAMTR,OGAMA,RTHETB,RTRANB
C COMMON/PLT/ NVP(2),NXVP(20,2),ICC
C DIMENSION XO(100),YO(100),X(100),Y(100),VCOM(100),DLS(100),
C XS(100),YS(100),XSTGR(100),YSTGR(100),DBPP(100)
C DIMENSION CASEID(20),XCTRI(2),ITRI(2),NBL(2)
C LOGICAL SINGLE,TRFIND
C TRFIND(1)= .FALSE.
C TRFIND(2)= .FALSE.

64
ICASE = 0
READ(5,5,END=999) TITLE
FORMAT(20A4)
ICASE = ICASE + 1
REWIND 3
C READ (5,10)
FORMAT(1X)
READ (5,20) ITRI(1),ITRI(2),IRST,ICYTL,IP
FORMAT(16I5)
READ (5,20)
READ (5,25) INLET,ISTAG,ALPHAI,STAG,SP,PGAMTR,OMEGA
READ (5,27) RN,XCTRI(1),XCTRI(2),ALPHAA
FORMAT(4E10.0)
IF (IP.EQ.-2) THEN
  READ (5,20) NVP(1),NVP(2)
  IF (NVP(1).NE.0) READ (5,20) (NXVP(I,1),I=1,NVP(1))
  IF (NVP(2).NE.0) READ (5,20) (NXVP(I,2),I=1,NVP(2))
END IF
IF (ICASE.EQ.1) READ (5,20) N,NI
IREAD = 1
IBLOW = 1
SINGLE = .FALSE.
IF (SP.LE.0.0) SINGLE = .TRUE.
N = N - 1
NI = N + 1
IF (ICASE.GT.1) THEN
  NI = NISAVE
  N = N - 1
GOTO 53
END IF
IF (IREAD.EQ.1) GO TO 40
C READ (5,10)
READ (5,3C) (XO(I),YO(I),I=1,N+1)
FORMAT(2F10.0)
GO TO 50
C
G40 READ(5,10)
40 READ(5,45) (XO(I),I=1,N+1)
C READ(5,10)
READ(5,45) (YO(I),I=1,N+1)
FORMAT(6F10.0)
C
50 CONTINUE
IF (IP.EQ.-2) THEN
  WRITE(12,20) N+1,NVP(1),NVP(2),90,70,INLET
  IF (NVP(1).NE.0) WRITE(12,20) (NXVP(I,1),I=1,NVP(1))
  IF (NVP(2).NE.0) WRITE(12,20) (NXVP(I,2),I=1,NVP(2))
  IF (INLET.NE.1) WRITE(12,80) RN,ALPHAA
  IF (INLET.EQ.1) WRITE(12,80) RN,ALPHAI
  WRITE(12,82) (XO(I),I=1,N+1)
  WRITE(12,82) (YO(I),I=1,N+1)
FORMAT(2E15.5)
FORMAT(8F10.6)
END IF
C
NRITE = (NI+1)/2
IMIN = (NI-1)/2+1
IF((NI/2)=N1) IMIN = NI/2
CALL TRGRID (N1,XO,YO,NI,NRITE,0.5,IMIN,RAD,1,NXSS1)
NISAVE = NI
53 CONTINUE
ALPHAA = 0.0174533 * ALPHAA
ALPHAI = 0.0174533 * ALPHAI
STAG = 0.0174533 * STAG
C IF (INLET .EQ. 0) THEN
ALPHA = ALPHAA
ELSE
ALPHA = ALPHAI
END IF
C IF (ISTAG .NE. 0) THEN
CALL STAGR(N1,STAG,XO,YO,XSTGR,YSTGR)
ELSE
DO 55 I = 1 , N1
XSTGR(I) = XO(I)
YSTGR(I) = YO(I)
55 CONTINUE
END IF
C READ DATA FROM VISCOS CAL.
C ICYCLE = 0
CYCLE = ICYCLE + 1
C CALL POTNL(N1,IRST,ALPHA,CHORD,XO,YO,XSTGR,YSTGR,X,Y,DLS,VCOM,
+ DBPP)
IF (ICYCLE .GT. ICYTL) THEN
REWIND 3
WRITE (3)N1,(X0(I),YO(I),DLS(I),VN(I),DBPP(I),I=1,N1)
GOTO 1
END IF
C IF (ISTAG .NE. 0) THEN
DO 70 I = 1 , N1-1
X(I) = 0.5 * (XO(I)+XO(I+1))
Y(I) = 0.5 * (YO(I)+YO(I+1))
70 CONTINUE
END IF
C CALL CASBLP(N1,XO,YO,X,Y,XS,YS,DLS,VCOM,DBPP,RN
+ ,NBL,ITRI,XCTRI,TITLE)
GO TO 60
999 CONTINUE
STOP
END
C SUBROUTINE POTNL(N1,IRST,ALPHA,CHORD,XO,YO,XSTGR,YSTGR,X,Y,DLS,
+ VCOM,DBPP)
C COMMON/BLOW/VN(100)
C SIMPLE SOURCE POTENTIAL CODE

```
DIMENSION AOFF(100,100), BOFF(100,100), XP(100), YP(100), X(100),
+ S(100), C(100), D(100), VTAN(3,100), VNOR(3,100), R(3,100),
+ VGCOM(100), SIGCOM(100), CP(100), XO(100), YO(100),
+ VNC(100), D1(100), D2(100), D3(100), SO(100), SC(100),
+ XOFF(100), YOFF(100), T(3,100), VTCOM(100), VNCOM(100),
+ XS(100), YS(100), SOFF(100), COFF(100), XP(100), YP(100),
+ XSTGR(100), YSTGR(100), VT(3), VL(3), VUN(3), VLN(3), DBPP(100),
+ CPI(100), XOS(100), YOS(100), DBPPC(100)
```

REAL NUM1, NUM2
LOGICAL OFF, SINGLE
PI = 3.141592
CM = 0.0
N = N1 - 1
IF (ICYCLE .EQ. 1) THEN
  IF (IRST .EQ. 0) THEN
    DO 10 I=1,N1
    DLS(I) = 0.0
    VN (I) = 0.0
    DBPP(I)= 0.0
  10 CONTINUE
  ELSE
    DO 5 I = 1 , N1
    XOS(I) = XO(I)
    YOS(I) = YO(I)
  5 CONTINUE
  READ (3) NT,(XS(I),YS(I),DLSC(I),VNC(I),DBPPC(I),I=1,NT)
  XMIN = XS(1)
  DO 15 I = 2 , NT
    IF (XS(I) .GT. XMIN) GOTO 15
    XMIN = XS(I)
    IMIN = I
  15 CONTINUE
  DO 17 I = 1 , NT
    IF (I .LT. IMIN) GOTO 16
    XS(I) = XS(I) - XMIN
    GOTO 17
  16 XS(I) = XMIN - XS(I)
  17 CONTINUE
C
  XMIN = XOS(I)
  DO 20 I = 2 , N1
    IF (XOS(I) .GT. XMIN) GOTO 20
    XMIN = XOS(I)
    IMIN = I
  20 CONTINUE
  DO 22 I = 1 , N1
    IF (I .LT. IMIN) GOTO 21
    XOS(I) = XOS(I) - XMIN
```

67
CALL DIFF3(NT,XS,DLSC,D1,D2,D3,0)
CALL INTRP3(NT,XS,DLSC,D1,D2,D3,N1,XOS,DLS)
CALL AMEAN (1,N1,XOS,DLS,1)
CALL DIFF3 (NT,XS,VNC,D1,D2,D3,0)
CALL INTRP3(NT,XS,VNC,D1,D2,D3,N1,XOS,VN)
CALL AMEAN (1,N1,XOS,VN,1)
CALL DIFF3 (NT,XS,DBPPC,D1,D2,D3,0)
CALL INTRP3(NT,XS,DBPPC,D1,D2,D3,N1,XOS,DBPP)
CALL AMEAN (1,N1,XOS,DBPP,1)
END IF
END IF
DO 30 I=1,N1
XP(I) = XSTGR(I)
YP(I) = YSTGR(T)
CONTINUE
CALCULATE GEOMETRIC QUANTITIES
DO 100 J=1,N
VNC(J) = 0.5 * (VN(J) + VN(J+1))
X(J) = .5*(XP(J)+XP(J+1))
Y(J) = .5*(YP(J)+YP(J+1))
D(J) = SQRT((XP(J+1)-XP(J))**2 + (YP(J+1)-YP(J))**2)
C(J) = (XP(J+1)-XP(J))/D(J)
S(J) = (YP(J+1)-YP(J))/D(J)
100 CONTINUE
CALCULATE NORMAL AND TANGENTIAL MATRICES
CONTINUE
IF ( INLET .NE. 0 .AND. .NOT. SINGLE) THEN
SUM = D(1)
DO 35 J = 2, N
SUM = SUM + D(J)
35 CONTINUE
Q = 2.0 * PI * SUM / SP
ELSE
Q = 0.0
END IF
C
CALCULATE NORMAL AND TANGENTIAL MATRICES
CONTINUE
IF (SINGLE) THEN
IF ( .NOT. OFF) THEN
DO 120 I=1,N
DO 110 J=1,N
IF (J .EQ. I) GO TO 105
XX= (X(I)-X(J))*C(J) + (Y(I)-Y(J))*S(J)
YY=-(X(I)-X(J))*S(J) + (Y(I)-Y(J))*C(J)
UU= LOG(((XX+.5*D(J))**2+YY**2)/((XX-.5*D(J))**2+YY**2))
VV= 2.*ATAN2(YY*D(J), XX**2+YY**2-.5*D(J)**2)
SS= S(I)*C(J) - C(I)*S(J)
CC= C(I)*C(J) + S(I)*S(J)
A(I,J)= -UU*SS + VV*CC
B(I,J)= UU*CC + VV*SS
GO TO 110
105 A(I,J) = 6.2831853
B(I,J) = 0.0
110 CONTINUE
120 CONTINUE
ELSE
DO 140 I=1,N
DO 130 J=1,N
XX = (XOFF(I) - X(J)) * C(J) + (YOFF(I) - Y(J)) * S(J)
YY = -(XOFF(I) - X(J)) * S(J) + (YOFF(I) - Y(J)) * C(J)
U = LOG(((XX + 0.5*D(J))**2 + YY**2) / ((XX - 0.5*D(J))**2 + YY**2))
VV = 2.*ATAN2(YY*D(J), XX**2 + YY**2 - (0.5*D(J))**2)
SS = SOFF(I)*C(J) - COFF(I)*S(J)
CC = COFF(I)*C(J) + SOFF(I)*S(J)
AOFF(I,J) = UU*SS + VV*CC
BOFF(I,J) = UU*CC + VV*SS
CONTINUE
CONTINUE
END IF
ELSE
IF (.NOT. OFF) THEN
DO 50 I=1,N
DO 40 J=1,N
IF (J.EQ. I) GO TO 45
XX = (X(I) - X(J)) * C(J) + (Y(I) - Y(J)) * S(J)
YY = -(X(I) - X(J)) * S(J) + (Y(I) - Y(J)) * C(J)
DX1 = PI*(X(I) - XP(J)) / SP
DY1 = PI*(Y(I) - YP(J)) / SP
DX2 = PI*(X(I) - XP(J+1)) / SP
DY2 = PI*(Y(I) - YP(J+1)) / SP
R1SQ = (COSH(DX1))**2 - (COS(DY1))**2
R2SQ = (COSH(DX2))**2 - (COS(DY2))**2
UU = LOG(R1SQ/R2SQ)
NUM1 = DX1*COSH(DX1)*SIN(DY1) +
       DY1*SINH(DX1)*COS(DY1) +
       DX2*COSH(DX2)*SIN(DY2) +
       DY2*SINH(DX2)*COS(DY2) +
       DX2*COSH(DX2)*SIN(DY2) +
       DY2*SINH(DX2)*COS(DY2) +
       EXV = 2.0*ATAN2(NUM2,DNUM2) - 2.0*ATAN2(NUM1,DNUM1)
VV = 2.0*ATAN2(YY*D(J), XX**2 + YY**2 - (0.5*D(J))**2)
VV = VV + EXV
SS = S(I)*C(J) - C(I)*S(J)
CC = C(I)*C(J) + S(I)*S(J)
A(I,J) = -UU*SS + VV*CC
B(I,J) = UU*CC + VV*SS
GO TO 40
CONTINUE
CONTINUE
ELSE
DO 70 I=1,N
DO 60 J=1,N
XX = (XOFF(I) - X(J)) * C(J) + (YOFF(I) - Y(J)) * S(J)
YY = -(XOFF(I) - X(J)) * S(J) + (YOFF(I) - Y(J)) * C(J)
A(I,J) = 6.2831853
B(I,J) = 0.0
CONTINUE
CONTINUE
ELSE
DO 70 I=1,N
DO 60 J=1,N
XX = (XOFF(I) - X(J)) * C(J) + (YOFF(I) - Y(J)) * S(J)
YY = -(XOFF(I) - X(J)) * S(J) + (YOFF(I) - Y(J)) * C(J)
DX1 = PI *(XOFF(I)-XP(J)) / SP
DY1 = PI *(YOFF(I)-YP(J)) / SP
DX2 = PI *(XOFF(I)-XP(J+1)) / SP
DY2 = PI *(YOFF(I)-YP(J+1)) / SP
R1SQ = (COSH(DX1))**2 - (COS(DY1))**2
R2SQ = (COSH(DX2))**2 - (COS(DY2))**2
U = LOG(R1SQ/R2SQ)
NUM1 = DX1 * COSH(DX1) * SIN(DY1) -
+ DY1 * SINH(DX1) * COS(DY1)
DNUM1 = DX1 * SINH(DX1) * COS(DY1) +
+ DY1 * COSH(DX1) * SIN(DY1)
NUM2 = DX2 * COSH(DX2) * SIN(DY2) -
+ DY2 * SINH(DX2) * COS(DY2)
DNUM2 = DX2 * SINH(DX2) * COS(DY2) +
+ DY2 * COSH(DX2) * SIN(DY2)
EXV = 2.0 * ATAN2(NUM1,DNUM1) - 2.0 * ATAN2(NUM2,DNUM2)
VV = 2.*ATAN2(YY*D(J), XX**2+YY**2-(.5*D(J))**2)
VV = VV + EXV
SS = SOFF(I)*C(J) - COFF(I)*S(J)
CC = COFF(I)*C(J) + SOFF(I)*S(J)
AOFJ = -UU*SS + VV*CC
BOFJ = UU*CC + VV*SS
C NORMAL AND TANGENTIAL COMPONENTS OF FUNDAMENTAL SOLUTIONS
C
DO 160 I=1,N
SUMR= 0.
SUMT= 0.
IF ( .NOT. OFF ) THEN
R(1,I)= S(I)+VNC(I)/COS(ALPHA)
T(1,I) = C(I)
R(2,I)= -C(I)
T(2,I) = S(I)
DO 145 J=1,N
SUMR = SUMR + B(I,J)
SUMT = SUMT + A(I,J)
145 CONTINUE
ELSE
R(1,I) = SOFF(I)
T(1,I) = COFF(I)
R(2,I) =-COFF(I)
T(2,I) = SOFF(I)
DO 150 J=1,N
SUMR= SUMR + BOFJ
SUMT = SUMT + AOFJ
150 CONTINUE
END IF
R(3,I)= SUMR
T(3,I)= SUMT
160 CONTINUE
C IF ( OFF ) GO TO 275
C DECOMPOSITION OF MATRIX A
DO 230 I=1,N-1
DO 220 K=1+1,N
A(K,I) = A(K,I)/A(I,I)
DO 210 J=I+1,N
A(K,J) = A(K,J) - A(K,I)*A(I,J)
210 CONTINUE
220 CONTINUE
230 CONTINUE

C OPERATE ON FUNDAMENTAL RIGHT SIDES WITH LOWER TRIANGULAR
DO 270 K=1,3
DO 260 J=1,N-1
DO 250 I=J+1,N
R(K,I) = R(K,I) - A(I,J)*R(K,J)
250 CONTINUE
260 CONTINUE
270 CONTINUE
C BACK SOLUTION
DO 300 K=1,3
DO 290 I=N,1,-1
SUM = 0.
DO 280 J=N,1+1,-1
SUM = SUM + A(I,J)*SIG(K,J)
280 CONTINUE
290 CONTINUE
300 CONTINUE
OFF = .TRUE.
C ADD DISPLACE VERTICALLY TO THE BODY TO GENERATE
C DISPLACEMENT SURFACE
DO 305 I = 2, N
SOFF(I) = (S(I)*D(I-1)+S(I-1)*D(I))/(D(I-1)+D(I))
COFF(I) = (C(I)*D(I-1)+C(I-1)*D(I))/(D(I-1)+D(I))
305 CONTINUE

DO 306 I = 1 , N1
XPOFF(I) = XP(I) - SOFF(I) * DLS(I)
YPOFF(I) = YP(I) + COFF(I) * DLS(I)
306 CONTINUE

DO 307 I = 1 , N
XOFF(I) = 0.5 * (XPOFF(I) + XPOFF(I+1))
YOFF(I) = 0.5 * (YPOFF(I) + YPOFF(I+1))
DOFF = SQRT((XPOFF(I+1)-XPOFF(I))**2 +
          (YPOFF(I+1)-YPOFF(I))**2)
COFF(I) = (XPOFF(I+1)-XPOFF(I))/DOFF
SOFF(I) = (YPOFF(I+1)-YPOFF(I))/DOFF
307 CONTINUE
GO TO 102
C CALCULATION OF SURFACE VELOCITIES FOR THE FUNDAMENTAL SOLUTIONS
C
DO 330 K=1,3
DO 320 I=1,N
SUMT = T(K,I)
SUMN = -R(K,I)
DO 310 J=1,N
SUMT = SUMT + BOFF(I,J)*SIG(K,J)
SUMN = SUMN + AOFF(I,J)*SIG(K,J)
CONTINUE
VTAN(K,I) = SUMT
VNOR(K,I) = SUMN
CONTINUE
DOFF1 = SQRT((XPOFF(2)-XPOFF(1))**2+(YPOFF(2)-YPOFF(1))**2)
DOFF2 = SQRT((XPOFF(3)-XPOFF(2))**2+(YPOFF(3)-YPOFF(2))**2)
DOFFN = SQRT((XPOFF(N+1)-XPOFF(N))**2+(YPOFF(N+1)-YPOFF(N))**2)
DOFFN1 = SQRT((XPOFF(N)-XPOFF(N-l))**2+(YPOFF(N)-YPOFF(N-l))**2)
VUT(K) = VTAN(K,N) + DOFFN * (VTAN(K,N)-VTAN(K,N-1))/
          (DOFFN+DOFFN1)
VUN(K) = VNOR(K,N) + DOFFN * (VNOR(K,N)-VNOR(K,N-1))/
          (DOFFN+DOFFN1)
VLT(K) = VTAN(K,1) + DOFF1 * (VTAN(K,1)-VTAN(K,2))/
          (DOFF1+DOFF2)
VLN(K) = VNOR(K,1) + DOFF1 * (VNOR(K,1)-VNOR(K,2))/
          (DOFF1+DOFF2)
330 CONTINUE
C OUTPUT FUNDAMENTAL SOLUTIONS
IF (ICYCLE .EQ. 1 .OR. ICYCLE .GE. ICYTL-1 .OR. IP .GE. 0)
   WRITE(6,335) TITLE
335 FORMAT(1H1,///20A4//)
C DO 360 K=1,3
C WRITE (6,340) K
C340 FORMAT(///,1H ,'FUNDAMENTAL SOLUTION NUMBER ',I2/////)
C WRITE(6,345)
345 FORMAT(3X,'I' ,8X,'X',11X,'Y',10X,'VT',10X,'VN',8X,'SIG' ///)
C WRITE(6,375) 1, XP(1), YP(1), VLT(K), VN(K)
C DO 350 I=1,N
C WRITE(6,375) I , X(I), Y(I), VTAN(K,I), VNOR(K,I), SIG(K,I)
C350 CONTINUE
C WRITE(6,375) N1 , XP(N1), YP(N1), VUT(K), VUN(K)
C360 CONTINUE
C COMBINED FLOW AT ANGLE OF ATTACK
C
IF ( INLET .NE. 0 ) THEN
YYY = ((((VUT(3)+VLT(3))*TAN(ALPHAI)+(VUT(1)+VLT(1))*Q)/
        ((VUT(3)+VLT(3))-(VUT(2)+VLT(2))*Q))
        +
        (((VUT(3)+VLT(3))-(VUT(2)+VLT(2))*Q))
XXX = -((VUT(1)+VLT(1))+(VUT(2)+VLT(2))*TAN(ALPHAI))/
        +
        (((VUT(3)+VLT(3))-(VUT(2)+VLT(2))*Q))
ALPHA = ACOS((1.0/SQRT(1.0+YYY**2)))
COSAL = COS(ALPHA)
SINAL = SIN(ALPHA)
W = XXX/SQRT(1.0+YYY**2)
ELSE
COSAL = COS(ALPHA)
SINAL = SIN(ALPHA)
W =-((VLT(1)+VUT(1))*COSAL+(VLT(2)+VUT(2))*SINAL)/
       ((VLT(3)+VUT(3)))
END
C FORCE COEFFICIENT CALCULATION
SUM1 = 0.
SUMX = 0.
SUMY = 0.
DO 390 I=1,N
   SUM1 = SUM1 + D(I)
   SUMX = SUMX - VCOM(I)**2*S(I)*D(I)
   SUMY = SUMY + VCOM(I)**2*C(I)*D(I)
390 CONTINUE
C FIND MAN. CHORD LENGTH
XOMIN = XO(I)
DO 395 I = 2, N
   IF (XO(I) .GT. XOMIN) GOTO 395
   XOMIN = XO(I)
395 CONTINUE
CHORD = XO(N) - XOMIN
CL1 = SUM1*25.13.274**W/CHORD
CL2 = (SUM*)COSAL-SUMX*SINAL)/CHORD
CD = (SUMX*COSAL+SUMY*SINAL)/CHORD
C CALCULATING PARAMETERS FOR INLET VELOCITY AS MODULUS OF NOMORIZED VEL
IF (.NOT. SINGLE) THEN
   NUM1 = SIN(ALPHA)+CL1*CHORD/(4.0*SP)
   ALPHID = ATAN2(NUM1,COS(ALPHA))
   NUM1 = SIN(ALPHA)-CL1*CHORD/(4.0*SP)
   ALPHED = ATAN2(NUM1,COS(ALPHA))
   NUM1 = CL1*CHORD/(2.0*SP)**2*COS(ALPHA)
   DNUM1 = 1.0-(CL1*CHORD/(4.0*SP))**2
   DALPHA = ATAN2(NUM1,DNUM1)
   VEXIT = (TAN(ALPHID)-TAN(ALPHED))]*(2.0*SP/CHORD*COS(ALPHID))/CL1
   CLI = CL1**2
   UIOU = 1.0/UOU
   VEXIT = COS(ALPHA)/COS(ALPHED)
ELSE
   ALPHID = ALPHA
   ALPHED = ALPHA
   DALPHA = 0.0
   UOU = 1.0
   UIOU = 1.0
   CLI = CL1
   VEXIT = 1.0
END IF
FAC = 180.0/PI
ALPHID = ALPHID * FAC
ALPHED = ALPHED * FAC
DALPHA = DALPHA * FAC
ALPHAD = ALPHA * FAC
IF (ICYCLE .EQ. 1 .OR. ICYCLE .GE. ICYTL-1 .OR. IP .GE. 0) THEN
WRITE(6,370) ALPHAD ,ALPHID,ALPHED,DALPHA,UIOU,VEXIT
370 FORMAT (///,1H , 'COMBINED FLOW AT AVERAGE ANGLE OF ATTACK = ',
   + F8.3, 4X,'DEGREES',//,1H ,17X,'INLET ANGLE OF ',
   + 'ATTACK = ',F8.3,4X,'DEGREES',//,1H ,
   + 17X,'EXIT ANGLE = ',F8.3,4X,'DEGREES',//,1H ,17X,
   + 'TURNING ANGLE = ',F8.3,4X,'DEGREES',//,1H ,17X,
'INLET VEL = ', F10.6, 3X, 'EXIT VEL = ', F10.6, //)
WRITE(6, 363)
END IF
DO 380 I = 1, N
VTCOM(I) = VTAN(1, I) * COSAL + VTAN(2, I) * SINAL + W * VTAN(3, I)
VNCOM(I) = VNOI(1, I) * COSAL + VNOI(2, I) * SINAL + W * VNOI(3, I)
VCOM(I) = SQRT(VTCOM(I) ** 2 + VNCOM(I) ** 2)
IF (VTCOM(I) . LT. 0.0) VCOM(I) = - VCOM(I)
CP(I) = 1.0 - VCOM(I) ** 2
CPI(I) = 1.0 - (VCOM(I) * VCOM(I)) ** 2
SIGCOM(I) = SIG(1, I) * COSAL + SIG(2, I) * SINAL + W * SIG(3, I)
XP(I) = 0.5 * (XO(I) + XO(I+1))
YP(I) = 0.5 * (YO(I) + YO(I+1))
380 CONTINUE
IF (ICYCLE . EQ. 1 . OR. ICYCLE . GE. ICYTL-1 . OR. IP . GE. 0) THEN
WRITE (1, 374) ( XO(I), YO(I), XP(I), YP(I), CP(I), CPI(I), I = 1, N)
WRITE (6, 375) ( I, XO(I), YO(I), XP(I), YP(I), VTCOM(I), +
   VNCOM(I), VCOM(I), CP(I), CPI(I), I = 1, N)
374 FORMAT(6F10.4)
375 FORMAT(1X, I3, 9F12.4)
C WRITE (2) I, XO(I), YO(I), XP(I), YP(I), CP(I), CPI(I), I = 1, N)
WRITE (6, 385) N+1, XO(N+1), YO(N+1)
385 FORMAT(1X, I3, 2F12.4)
WRITE (6, 400) CHORD, CL1, CLI
400 FORMAT(//3X, 'CHORD = ', F10.5, 4X, 'CL(AVG) = ', F10.5, 4X,
   + 'CL(INLET) = ', F10.5)
END IF
420 FORMAT(/3X, 'X = ', 1HV, 6X, 2HSO, 10X, 2HSC, 9X, 3H VN, 9X, 3HVNC, 9X, 3HDLS, 8X,
   + 4HDLS)
430 FORMAT(15, 6E12.4)
RETURN
END
C
C DATA SET KBBCAMEAN AT LEVEL 001 AS OF 08/24/84
C DATA SET KBBCAMEAN AT LEVEL 003 AS OF 04/05/84
C
C SUBROUTINE AMEAN(NS, ND, X, Y, IT)
C
C SMOOTH DATA USING 3-PTS WEIGHTING FORMULA
C NS : STARTING POINT OF THE DATA TO BE SMOOTHED
C ND : END POINT OF THE DATA TO BE SMOOTHED
C X, Y : INDEPENDENT + DEPENDENT VARIABLES OF THE DATA
C TO BE SMOOTHED
C IT : CYCLES OF DATA SMOOTHING
C
C DIMENSION X(101), Y(101)
C
C NM = ND - NS
IF(NM . LT. 2 . OR. IT . LT. 1) RETURN
C
C NDM1 = ND - 1
NSPI = NS + 1
C
C INT05530
INT05540
INT05550
INT05560
INT05570
INT05580
INT05590
INT05600
INT05610
INT05620
INT05630
INT05640
INT05650
INT05660
INT05670
INT05680
INT05690
INT05700
INT05710
INT05720
INT05730
INT05740
INT05750
INT05760
INT05770
INT05780
INT05790
INT05800
INT05810
INT05820
INT05830
INT05840
INT05850
INT05860
INT05870
INT05880
INT05890
INT05900
INT05910
INT05920
INT05930
INT05940
INT05950
INT05960
INT05970
INT05980
INT05990
INT06000
INT06010
INT06020
INT06030
INT06040
INT06050
INT06060
INT06070
INT06080
74
DO 20 K=1,IT
DL1 = X(NSP1) - X(NS)
Y1 = Y(NS)
DO 10 I=NSP1,NDM1
DL2 = X(I + 1) - X(I)
Y2 = Y(I)
YM = (DL2 - Y1 + DL1 * Y(I+1))/(DL1 + DL2)
Y(I) = 0.5 * (Y2 + YM)
DL1 = DL2
Y1 = Y2
10 CONTINUE

20 CONTINUE
RETURN
END

DATA SET KCBCBLGRID AT LEVEL 001 AS OF 08/24/84
DATA SET KCBCBLGRID AT LEVEL 001 AS OF 08/24/84
DATA SET KCBCBLGRID AT LEVEL 004 AS OF 04/05/84
SUBROUTINE BLGRID(N,X,T,D1)
GENERATE B. L. X-WISE GRID USING MODIFIED COSINE DISTRIBUTION
DIMENSION X(101),T(101),D1(101)
DATA CRAD/57.2957795/, BPI/3.14159265/

NN = 2 * N - 1
EN = FLOAT((NN-1)/2)
THO = 10./CRAD
CTO1 = 1. + COS(THO)
DTH = (BPI - THO) / EN
FI = FLOAT(N - 2)
DO 10 I=N,NN
FI = 1.0 + FI
II = I - N + 1
XII = THO + FI * DTH
X(II) = (1.0 + COS(XII))/CTO1
10 CONTINUE
X1 = X(1)
XN = X(N)
CH = XN - X1
FN1 = FLOAT(N-1)
N10 = N/10
DO 20 I=1,N
T(I) = FLOAT(I-1)/FN1
X(I) = (X(I)-X1)/CH
20 CONTINUE
CALL SMFIT(N10,N,T,X,D1,N10)
CALL SMFIT(1,N,T,X,D1,2)
CALL AMEAN(1,N,T,X,N10)
RETURN
END

DATA SET KCBCBL2D AT LEVEL 001 AS OF 08/24/84
C DATA SET KCBCBL2D AT LEVEL 001 AS OF 08/24/84
C DATA SET KCBCBL2D AT LEVEL 012 AS OF 04/06/84
SUBROUTINE BL2D ( ITR, ISWPT, SURFID )
PROGRAM CALCULATES VISCOUS/INVISCID INTERACTION USING HILBERT INTEGRAL.
C
C
C COMMON /BLC0/ NX, NXT, NP, NPT, NTR, IT, INVRS, NS, IP
C COMMON /BLC1/ P(101, 2), U(101, 2), V(101, 2), W(101, 2), B(101, 2)
C COMMON /BLC2/ DELF(101), DELU(101), DELV(101), DELW(101)
C COMMON /BLC7/ C(100, 100), D(100), DB(100), DBP(100), UE0(100), GI
C COMMON /EDDY1/ RL, RX, SQRX, RXNTR, GMTR, GMTRS(100), ALFAS(100), +
C COMMON /SMRY/ W(100), ITP(100), ITP(100), ITP(100), DELS(100), CF(100), +
C COMMON /GTY/ X(101), UE(100), P1(100), P2(100), CEL, CELH
C COMMON /BONV/ ITMAX, EPSL, EPST, CONV
C COMMON /SAVE/ FS(101), US(101), VS(101), WS(101), BS(101)
C COMMON /BLIN/ TITLE(20), XC(100), YC(100), ISG(100), DELS(100), +
C COMMON /1SURF/ ISF
C COMMON /PLOT/ NS(4), NS(4), ICC
DIMENSION SURFID(4)

GENERATE B. L. GRIDS + SET INITIAL CONDITIONS

DO 5 I = 1, NXT
ALFAS(I) = 0.0
FFS(I) = 1.0
RTS(I) = 1.0
5 CONTINUE
CALL INPUT (ITR, ISWPT, SURFID)

CALCULATE HILBERT COEFS., C(I, J)

CALL CALCIJ(NXT, 0)

LOOP OF CALCULATIONS

NSS = NS
NXSPT = NXT + 1
C IF (ICYCLE .EQ. 1 ) NS = NXT + 1
NX = NX + 1
CEL = 0.5 * (X(NX) + X(NX-1)) / (X(NX) - X(NX-1))
CELH = 0.5 * CEL
IT = 0
RX = UE(NX) * X(NX) * RL
SQRX = SQRT(RX)
30 IT = IT + 1
IF(IT .LE. ITMAX) GO TO 40
NXM1 = NX - 1
CALL HEADER( TITLE, SURFID, ISTRP )
WRITE(6, 170 ) (M, X(M), CF(M), DELS(M), UE(M), P2(M), THT(M), +
D(M), ALFAS(M), ITP(M), NPSTR(M), M = 1, NXM1)
WRITE(6, 160 ) NX
CONTINUE 
IF(NX.GT.NTR) CALL EDDY
CALL COEFTR
CALL SOLV3
IF(V(1,2).GT.0.0) GOTO 60
C
C EXTRAPOLATE CALCULATED D FOR TURBULENT SEPARATION OR LAMINAR 
C SEPARATION FOR LAMINAR FLOW CALCULATION ONLY
C
CALL EXTRAP(NX,NXT,X,D)
NXM1 = NX - 1
CALL HEADER( TITLE,SURFID,ISTRP )
WRITE(6,170) (M,X(M),CF(M),DLS(M),UE(M),P2(M),THT(M),
+ D(M),ALFAS(M),ITP(M),NPSTR(M),M=1,NXM1)
WRITE(6,180)
WRITE(6,190) (M,X(M),D(M),M=NX,NXT)
GOTO 130
C
IF(NX.GT.NTR) GO TO 70
IF(ABS(DELV(1)).GT.EPSL) GO TO 30
GO TO 80
C
CONTINUE 
IF(ABS(DELV(1)/V(1,2)).GT.EPST) GO TO 30
C
CHECK FOR GROWTH
C
IF(NP.GE.NPT) GO TO 90
IF(ABS(V(NP,2);).LT.0.0005 .AND. ABS(1.0-U(NP-2,2))
+ .LT.0.0035) GOTO 90
CALL FILLUP(1)
IT = 1
GO TO 30
C
90 
CONTINUE
C
CALL FILLUP(2)
CALL OUTPUT(1)
IF(ITR.EQ.0 .OR. NX.GE.NTR) GOTO 100
IF(NX.LT.3 .OR. ITR.NE.3) GO TO 100
CALL TRNS(ICODE)
IF(ICODE.EQ.1) GOTO 20
100 IF(NX.NE.NSS) GOTO 120
C
STORE PROFILES AT THE STATION NS FOR INVERSE B. L.
C
CALCULATION
C
DO 110 J = 1 , NPT
FS(J) = F(J,2)
US(J) = U(J,2)
VS(J) = V(J,2)
WS(J) = W(J,2)
BS(J) = B(J,2)
110 CONTINUE
C
IF ( NX .LT. NSS ) GOTO 10
C
IF (ICYCLE.NE.1) GO TO 130
IF (NX.GE.NS) GOTO 130
IF (NX .LT. NXT) GO TO 10
CALL HEADER( TITLE,SURFID,ISTRP )
WRITE(6, 170 )(M,X(M),CF(M),DLS(M),UE(M),P2(M),THT(M),
+ D(M), ALFAS(M), ITP(M),NPSTR(M),M=1,NXT)
130 DO 140 I = 1 , NXT
140 DB(I) = D(I)
NS = NSS
NX = NS
NP = NPSTR(NX)
DO 150 J = 1 , NPT
F(J,2) = FS(J)
U(J,2) = US(J)
V(J,2) = VS(J)
W(J,2) = WS(J)
B(J,2) = BS(J)
150 CONTINUE
155 INVRS = NS + 1
C CALCULATION SHIFTS TO USING PHYSICAL COORDINATES
CALL MAIN2(ITR,ISWPT,SURFID)
C PASS DELTA-STAR BACK TO MAIN PROG.
DO 158 I = 1,NXT
DELS(I) = DLS(I)
158 CONTINUE
RETURN
** ITERATIONS EXCEEDED ITMAX AT NX = ',I5/
+ 1H , '' CALCULATIONS STOP. ** '
170 FORMAT(1H0,' ** SUMMARY OF STANDARD B. L. SOLUTIONS. **'/
+ 1H0,4X,2HNX,7X,1HX,12X,2HCF,11X,3HDLS,12X,2HUE,
+ 12X,2HNP,11X,3HTHT,13X,1HD,10X,4HALFA,6X,2HIT,2X,2HNP/INT0810
+ ( 1H ,3X,13,F10.5,2X,7E14.5,2I4))
180 FORMAT(1H0,3H FLOW SEPARATES. D IS EXTRAPOLATED/
+ 1H0,3X,3H NX,7X,1HX,13X,1HD/)
PI = PI*SQRT(RL)
IL1 = IL - 1
DO 65 I = 2, IL
E (1) = 0.
L = LO + I
DO 60 J = 2, IL
J1 = J - 1
K = J + LO
DX1 = X(L) - X(K)
DX2 = X(K) - X(K-1)
DX3 = X(L) - X(K-1)
IF ( J .EQ. I ) GO TO 30
IF ( J .EQ. (I+1) ) GO TO 40

C C J .NE. I OR I+1 C
C
E (2) = ( 1.0/DX2 ) * ALOG( ABS( DX3 / DX1 ) )
GO TO 50
C
C J .EQ. I
C
30 R1 = ( X(K+1)-X(K) ) / (X(K)+1-X(K-1) )
E (2) = ( R1 * ALOG( ABS( DX3 / (X(L)-X(K+1)) ) ) + 2.0 ) / DX2
GO TO 50

C
C J .EQ. I+1
C
40 R1 = ( X(K-1)-X(K-2) ) / ( X(K)-X(K-2) )
E (2) = ( R1 * ALOG( ABS( (X(L)-X(K-2))/DX1 ) ) - 2.0 ) / DX2
GO TO 50

C
C CONTINUE
C (J1,I) = ( E(1) - E(2) ) / PI
E(1) = E(2)

60 CONTINUE
E (2) = 0.
J1 = IL
K = K + 1
C (J1,I) = E(1) / PI
C
65 CONTINUE
C
RETURN
C
END

C DATA SET KCBCCOEF AT LEVEL 001 AS OF 08/24/84
C DATA SET KCBCCOEF AT LEVEL 001 AS OF 08/24/84
C DATA SET KCBCCOEF AT LEVEL 007 AS OF 04/05/84

SUBROUTINE COEF(GAMMA1,GAMMA2)
C
C CALCULATE COEFFS OF B. L. FINITE-DIFFERENCE EQS. IN
C SEMI-TRANSF VARIABLES( AFTER SWITCHING).
C
C COMMON /BLCO/ NX,NXT, NP, NTR, IT, INVRS, NS, IP
C COMMON /BLC1/ P(101,2), U(101,2), V(101,2), W(101,2), B(101,2)
C COMMON /BLC6/ S1(101), S2(101), S3(101), S4(101), S5(101), S6(101),
C S7(101), S8(101), R1(101), R2(101), R3(101), R4(101)
C COMMON /BLC7/ C(100,100), D(100), DB(100), DBP(100), UE0(100), GI
COMMON /GRD / ETA(101),DETA(101),A(101)
COMMON /GTY / X(101),UE(100),P1(100),P2(100),CEL,CELH
C
C P1H = 0.5 * P1(NX)
DO 100 J= 2,NP
FLARE = 1.0
FB = 0.5*(F(J,2) + F(J-1,2))
UB = 0.5*(U(J,2) + U(J-1,2))
FVB = 0.5*(F(J,2)*V(J,2)+F(J-1,2)*V(J-1,2))
IF(UB .LT. 0.0) FLARE = 0.0
VB = 0.5*(V(J,2) + V(J-1,2))
USB = 0.5*(U(J,2)**2 + U(J-1,2)**2)
WSB = 0.5*(W(J,2)**2 + W(J-1,2)**2)
DERBV = (B(J,2)*V(J,2) - B(J-1,2)*V(J-1,2))/DETA(J-1)
FB4 = 0.5*(F(J,1) + F(J-1,1))
VB4 = 0.5*(V(J,1) + V(J-1,1))
USB4 = 0.5*(U(J,1)**2 + U(J-1,1)**2)
WSB4 = 0.5*(W(J,1)**2 + W(J-1,1)**2)
FVB4 = 0.5*(F(J,1)*V(J,1)+F(J-1,1)*V(J-1,1))
IF(UB .LT. 0.0) FLARE = 0.0
S1(J) = CELH*(FB - FB4) + P1H*F(J,2) + B(J,2)/DETA(J-1)
S2(J) = CELH*(FB - FB4) + P1H*F(J-1,2) - B(J-1,2)/DETA(J-1)
S3(J) = CELH*(VB + VB4) + P1H*V(J,2)
S4(J) = CELH*(VB + VB4) + P1H*V(J-1,2)
S5(J) = -CEL*FLARE*U(J,2)
S6(J) = -CEL*FLARE*U(J-1,2)
S7(J) = CEL*W(J,2)
S8(J) = CEL*W(J-1,2)

CRB = -DERBV4 + CEL*WSB4 - CEL*FLARE*USB4 - P1(NX)*FVB4
R2(J) = CRB - (DERBV - CEL*FLARE*USB + CEL*(VB+VB4)*(FB-FB4) +
+ CEL*WSB + P1(NX)*FVB)
R1(J) = F(J-1,2) - F(J,2) + DETA(J-1)*UB
R3(J-1) = U(J-1,2) - U(J,2) + DETA(J-1)*VB
R4(J-1) = W(J-1,2) - W(J,2)
100 CONTINUE
C
C BOUNDARY CONDITIONS
C
R1(1) = 0.0
R2(1) = 0.0
R4(NP) = 0.0
IF(NX .GE. INVRS) GO TO 120
GAMMA1 = 0.0
GAMMA2 = 1.0
R3(NP) = 0.0
RETURN
120 CONTINUE
CII = C(NX,NX) * SQRT(X(NX))
GAMMA1 = 1.0
GAMMA2 = (1.0 - CII*ETA(NP))/CII
R3(NP) = (G1 + CII*(ETA(NP)*W(NP,2) - F(NP,2)) -W(NP,2))/CII
C
RETURN
SUBROUTINE COEFTR
CALCULATE COEFFS. OF B. L. FINITE-DIFFERENCE EQS.
IN TRANSFORMED VARIABLES (BEFORE SWITCHING).

DO 100 J = 2, NP
FB  = 0.5&(F(J,2) + F(J-1,2))
UB  = 0.5&(U(J,2) + U(J-1,2))
VB  = 0.5&(V(J,2) + V(J-1,2))
USB = 0.5&(U(J,2)**2 + U(J-1,2)**2)
DERBV = (B(J,2)*V(J,2) - B(J-1,2)*V(J-1,2))/DETA(J-1)
FVB = 0.5&(F(J,2)*V(J,2) + F(J-1,2)*V(J-1,2))
FVB4 = 0.5&(F(J,1)*V(J,1) + F(J-1,1)*V(J-1,1))
FB4 = 0.5&(F(J,1) + F(J-1,1))
VB4 = 0.5&(V(J,1) + V(J-1,1))
USB4 = 0.5&(U(J,1)**2 + U(J-1,1)**2)
DERBV4 = (B(J,1)*V(J,1) - B(J-1,1)*V(J-1,1))/DETA(J-1)
S1(J) = CELH&(FB-FB4) + 0.5*P1(NX)*F(J,2) + B(J,2)/DETA(J-1)
S2(J) = CELH&(FB-FB4) + 0.5*P1(NX)*F(J-1,2) - B(J-1,2)/
          DETA(J-1)
S3(J) = CELH&(VB + VB4) + 0.5*P1(NX)*V(J,2)
S4(J) = CELH&(VB + VB4) + 0.5*P1(NX)*V(J-1,2)
S5(J) = -(CEL+P2(NX))*U(J,2)
S6(J) = -(CEL+P2(NX))*U(J-1,2)
CLB = DERBV4 + P1(NX-1)*FVB4 - P2(NX-1)*USB4 + P2(NX-1)
CRB = -CLB - CELB*USB4 - P2(NX)
R2(J) = CRB - (DERBV + P1(NX)*FVB - (CEL+P2(NX))*USB + CEL*
          (VB + VB4)*(FB - FB4))
R1(J) = F(J-1,2) - F(J,2) + DETA(J-1)*UB
R3(J-1) = U(J-1,2) - U(J,2) + DETA(J-1)*VB
100 CONTINUE
RETURN
END
DATA SET KCBBCOMPBL AT LEVEL 010 AS OF 08/24/84

SUBROUTINE COMPBL(CASEID, XP, YP, VT, S, DLSP, DLS, DBPP, NBL, ITRI, XCTRI, RN, NT, ISWPT)
+

COMMON /BLCO/ NX, NXT, NP, NPT, NTR, IT, INVRS, NS, IP
COMMON /BLC7/ C(100, 100), D(100), DB(100), DBP(100), UEO(100), GI
COMMON/EDDY1/ RL, RX, SQRX, RXNTR, GMR(100),
+ ALFAS(100), FFS(100), RTS(100), IEDY, NXNPT
COMMON /GTY / X(IO1), UE(100), P1(100), P2(100), CEL, CELH
COMMON /BLIN/ TITLE(20), XC(100), YC(100), ISG(100), DELS(100),
+ XCTR, XTR, ISTRP, ICYCLE, ICYT, XCTRS(2), TRFIND(2)
COMMON /BLOW/ VN(100)
COMMON /ISURF/ ISF
COMMON/PLT/NVP(2), NXVP(20, 2), ICC

DIMENSION XP(100), DLSP(100), YP(100), VT(100), S(100),
+ DBPP(100), DLS(100), CASEID(20)
DIMENSION XIN(100, 2), YIN(100, 2), SI(100, 2), VIN(100, 2),
+ DIN(100, 2), DELSTR(100, 2), DDD(100, 2), UDD(100, 2)
DIMENSION XB(101), D1(101), D2(101), D3(101), IEND(2)
DIMENSION NBL(2), ITRI(2), XCTRI(2)

LOGICAL TRFIND

CHARACTER * 4 SURF(4), STITLE(2), SURFID(4)

DATA SURF / ' ' 'R SU', 'RFAC', 'E' / DATA STITLE / 'UPPER', 'LOWE' /

90 FORMAT ( 1H1, 5X, 'COMPUTING BOUNDARY LAYER USING HILBERT',
+ ' INTEGRAL', )
110 FORMAT ( 1H0, 6X, 'I', 9X, 'XP', 13X, 'YP', 13X, 'S', 14X, 'VT', 13X,
+ 'DBP' / )
112 FORMAT ( 1H0, 6X, 'I', 4X, 'II' 3X, 'IK', 7X, 'XIN', 12X, 'YIN',
+ 13X, 'SI', 12X, 'VIN', 12X, 'DIN' / )
120 FORMAT ( 1H , 5X, '5E15.6' )
122 FORMAT ( 1H , 3X, 5E15.6 )
130 FORMAT ( 1H0, 5X, 'STAGNATION POINT IS FOUND BETWEEN POINT NO. ' ,
+ 'I3', ' AND POINT NO. 'I3 / )
140 FORMAT ( 1H0, 5X, 'INTERPOLATED STAGNATION POINT VALUES' /
+ 1H0, 5X, 'S = ', 'E13.6', 2X, 'XP = ', 'E13.6', 2X, 'YP = ',
+ 'E13.6', 2X, 'DBP = ', 'E13.6', 2X, 'VT = 0.0' / )
150 FORMAT ( 1H0, 5X, 'TOTAL NUMBER OF UPPER SURFACE POINTS = ', '15, '2X,' AND AT LOWER SURFACE = ', 'I5 / )
160 FORMAT ( 1H0, 5X, 'UPPER SURFACE DATA' )
170 FORMAT ( 1H0, 5X, 'LOWER SURFACE DATA' )
180 FORMAT ( 1H0, 5X, 'UPPER SURFACE CALCULATIONS' / )
182 FORMAT ( 1H0, 5X, 'LOWER SURFACE CALCULATIONS' / )
190 FORMAT ( 1H0, 5X, 'RESULTS OF POINT REDISTRIBUTION' / )
200 FORMAT ( 1H0, 5X, 'TABLE OF DELTA-STARS' / ( 1H , 5X, '8E15.6' ) )
220 FORMAT ( 1H0, 5X, 'TABLE OF BLOWING-VEL.' / ( 1H , 5X, '8E15.6' ) )
210 FORMAT ( 1H0, 5X, 'NO CHANGE OF SIGN ON VT. CANNOT FIND STAG. PT.' )

----------------------------------- * -----------------------------------
READ ONE STRIP INPUT DATA FROM UNIT NO. 1. THE ORDER IS
FROM THE LOWER SURFACE T.E. TO THE UPPER SURFACE T.E.

DO 230 I = 1,20
  TITLE(I) = CASEID(I)
  CONTINUE
  RL = RN
DO 300 I = 1,NT
  DBP(I) = DBP(I)
  CONTINUE

PRINT THE INPUT DATA.

P2(I) = 1.0
WRITE ( 6,90 )
WRITE ( 6,110 )
DO 500 I = 1,NT
WRITE ( 6,120 ) I,XP(I),YP(I),S(I),VT(I),DBP(I)
  CONTINUE

FIND STAGNATION POINT

DO 600 I = 2,NT
  VPROD = VT(I) * VT(I-1)
  IF ( VPROD .GT. 0. ) GO TO 600
  IS = I
  IS1 = IS - 1
  GO TO 700

WRITE ( 6,210 )
STOP 1

INTERPOLATE S AT VT = 0.

WRITE ( 6,130 ) IS1,IS
700  DS = S(IS) - S(IS1)
     DV = VT(IS) - VT(IS1)
     SS = S(IS) - VT(IS1) * ( DS/DV )
     DBB = DBP(IS) - DBP(IS1)
     DX = XP(IS) - XP(IS1)
     DY = YP(IS) - YP(IS1)
     DS1 = SS - S(IS)
     DBS = DBP(IS) + DS1 * ( DBB/DS )
     XPS = XP(IS) + DS1 * ( DX /DS )
     YPS = YP(IS) + DS1 * ( DY /DS )
     WRITE ( 6,140 ) SS,XPS,YPS,DBS

IU IS THE TOTAL UPPER SURFACE POINTS. + STAG. PT.
IL IS THE TOTAL LOWER SURFACE POINTS + STAG. PT.

IU = NT - IS + 2
IL = IS
WRITE ( 6,150 ) IU,IL

83
GROUP THE DATA FOR EACH SURFACE. FIRST, UPPER.

DO 1200 L = 1, 2
GO TO (800, 900), L

L = 1 IS UPPER SURFACE
L = 2 IS LOWER SURFACE

800 M1 = IS
M2 = NT
IEND(L) = IU
GO TO 1000

900 M1 = 1
M2 = IL-1
IEND(L) = IL

1000 I = 1
XIN(I,L) = XPS
YIN(I,L) = YPS
SI (I, L) = 0.
DIN(I,L) = DBS
VIN(I,L) = 0.
IF ( IP .GE. 1 ) THEN
IF ( L .EQ. 1 ) WRITE ( 6, 160 )
IF ( L .EQ. 2 ) WRITE ( 6, 170 )
WRITE ( 6, 112 )
WRITE ( 6, 122 ) I, I, I, XIN(1,L), YIN(1,L), SI(1,L), VIN(1,L),
+ DIN(1,L)
END IF
DO 1100 II = M1, M2
I = I + 1
IK = II
IF ( L .EQ. 2 ) IK = IL - II
XIN(I,L) = XP(IK)
YIN(I,L) = YP(IK)
SI (I, L) = S(IK) - SS
IF ( L .EQ. 2 ) SI(I,L) = SS - S(IK)
VIN(I,L) = ABS(VT(IK))
DIN(I,L) = DBP(IK)
IF (IP .GE. 1) WRITE ( 6, 122 ) I, II, IK, XIN(I,L), YIN(I,L), SI(I,L),
+ VIN(I,L), DIN(I,L)

1100 CONTINUE

1200 CONTINUE

RE-DISTRIBUTE POINTS ON EACH SURFACE TO A DENSER NUMBER.

WRITE ( 6, 90 )
DO 2000 ISF = 1, 2
NN = IEND(ISF)
ITRI = ITRI(ISF)
NBL = NBL(ISF)
XCTR = XCTRI(ISF)
SURF (1) = STITLE(ISF)
ICC = 1
DO 1220 J = 1, 4
SURFID(J) = SURF(J)

1220 CONTINUE
C IF ( ISF .EQ. 1 ) WRITE ( 6,180 )
C IF ( ISF .EQ. 2 ) WRITE ( 6,182 )
SCALE = SI(NN,ISF)
C CALL BLGRID ( NXT,XB,D1,D2 )
C DO 1300 I = 1,NXT
1300 X ( I ) = XB(I) * SCALE
C INTERPOLATE S,VT,X,Y,D INTO THE NEW DISTRIBUTION.
C CALL SMFIT ( 1,NN,SI(1,ISF),VIN(1,ISF),D1,2 )
CALL SMFIT ( 1,NN,SI(1,ISF),DIN(1,ISF),D1,2 )
CALL INTRP3( NN,SI(1,ISF),VIN(1,ISF),D1,D2,D3,NXT,X,UE )
CALL INTRP3 ( NN,SI(1,ISF),DIN(1,ISF),D1,D2,D3,NXT,X,DB )
CALL INTRP3 ( NN,SI(1,ISF),XIN(1,ISF),D1,D2,D3,NXT,X,XC )
CALL INTRP3 ( NN,SI(1,ISF),VIN(1,ISF),D1,D2,D3,NXT,X,YC )
IF ( IP .GE. 1 ) THEN
  WRITE ( 6,190 )
  WRITE ( 6,110 )
DO 1320 I = 1,NXT
  WRITE ( 6,120 ) I,XC(I),YC(I),X(I),UE(I),DB(I)
1320 CONTINUE
END IF
C INPUT TO THE B.L. PROGRAM X,UE,DB,XC,YC ARE NOW DEFINED.
C DO 1350 I = 1,NXT
 DBP(I) = DB(I)
 D(I) = DB(I)
1350 CONTINUE
C CALL BL2D( ITR,ISWPT,SURFID)
C CALL DIFF3 ( NXT,X,DELS,D1,D2,D3,0 )
CALL INTRP3( NXT,X,DELS,D1,D2,D3,NN,SI(1,ISF),DELSTR(1,ISF) )
CALL DIFF3(NXT,X,D,D1,D2,D3,0)
CALL INTRP3(NXT,X,D,D1,D2,D3,NN,SI(1,ISF),DD(1,ISF))
CALL DIFF3(NN,SI(1,ISF),DD(1,ISF),DDD(1,ISF),D2,D3,0)
TRFIND(ISF) = .FALSE.
IF(ITR .EQ. 3 .AND. NTR .LE. NXT) THEN
 XCTRS(ISF) = XCTR
 TRFIND(ISF) = .TRUE.
END IF
C 2000 CONTINUE
C PUT THE TWO SURFACES DELTA- STARS BACK TO ONE STRIP
C DELSTR(1,2) = 0.5*(DELSTR(2,1)+DELSTR(2,2))
DELSTR(1,1) = DELSTR(1,2)
DD(1,2) = 0.5 *(DD(2,1) + DD(2,2))
DD(1,1) = DD(1,2)
DDD(1,2) = 0.5 *(DDD(2,1) + DDD(2,2) /SQRT(RL)
DDD(1,1) = DDD(1,2)
IL = IEND(2)
I = IL
DO 2100 II = 2, IL
I = I-1
DLS(I) = DELSTR(II,2)
VN(I) = DDD(II,2)/SQRT(RL)
DBPP(I) = DD(II,2)
2100 CONTINUE
C
I = IL-1
IU = IEND(1)
DO 2200 II=2, IU
I = I + 1
DLS(I) = DELSTR(II,1)
VN(I) = DDD(II,1)/SQRT(RL)
DBPP(I) = DD(II,1)
2200 CONTINUE
C
WRITE ( 6,200 ) ( DLS(I), I=1,NT)
C
WRITE ( 6,220 ) ( VN(I) , I=1,NT)
C
RETURN
END

DATA SET KCBCCOMPGI AT LEVEL 001 AS OF 08/24/84
DATA SET KCBCCOMPGI AT LEVEL 001 AS OF 08/24/84
DATA SET KCBCCOMPGI AT LEVEL 003 AS OF 08/24/84

SUBROUTINE COMPGI

CALCULATE GI

COMMON /BLCO/ NX,NXT,NP,NPT,NTR,IT,INVRS,NS,IP
COMMON /BLC7/ C(100,100),D(100),DB(100),DBP(100),UE0(100),GI

SUM2 = 0.0
N1 = NX - 1
N2 = NXT
DO 130 K = N1,N2
SUM2 = SUM2 + C(K,NX) * (D(K) - DBP(K))
180 CONTINUE
C
N1 = 2
N2 = NX-1
SUM1 = 0.
DO 260 K = N1,N2
SUM1 = SUM1 + C(K,NX) * (D(K) - DBP(K))
260 CONTINUE
C
GI = UE0(NX) + SUM1 + SUM2 - C(NX,NX) * DBP(NX)
C
RETURN
SUBROUTINE DIFF3 (N,X,FP,FPP,FPPP,IEND)

C DETERMINES THE DERIVATIVE OF THE INPUT FUNCTION AT THE INPUT PTS.
C DIMENSION X(101),F(101),FP(101),FPP(101),FPPP(101)

C FIRST DERIVATIVES USING WEIGHTED ANGLES
C
N1=N-1
DX=X(2)-X(1)
DF=F(2)-F(1)
ANG2= ATAN2(DF,DX)
DL2=DX

DO 10 I=2,N1
  ANG1=ANG2
  DL1=DL2
  I1=I+1
  DX=X(I1)-X(I)
  DF=F(I1)-F(I)
  ANG2= ATAN2(DF,DX)
  DL2=DX
  ANG=(DL2*ANG1+DL1*ANG2)/(DL1+DL2)
  FP(I)= TAN(ANG)
C
IF (I.NE.2) GO TO 10
  ANGI = ANG
  ANG1I = ANG1
  DLI = DL1
C
10 CONTINUE
  ANGF = ANG
  ANG2F = ANG2
  DLF = DL2
  IEND1 = IEND + 1
  GO TO (11,12,13), IEND1
C
IEND = 0 , EXTRAPOLATE FOR END VALUES
C
11 FP(1) = 2.*(F(2)-F(1))/DLI - FP(2)
  FP(N) = 2.*(F(N)-F(N1))/DLF - FP(N1)
  GO TO 20
C
IEND = 1, DERIVATIVES ARE CONTINUOUS ACROSS ENDS
C
12 ANG = (DLI*ANG2F + DLF*ANG1I) / (DLI + DLF)
  FP(1) = TAN(ANG)
  FP(N) = FP(1)
GO TO 20

C IEND = 2, IF FIRST DERIVATIVE .LT. 0.0 RESET TO ZERO
C
13 CONTINUE
FP(1) = 2.*(F(2)-F(1))/DLI - FP(2)
IF (FP (1) .LT. 0.0) FP (1) = 0.0
FP(N) = 2.*(F(N)-F(N1))/DLF - FP(N1)
C
C SECOND + THIRD DERIVATIVES USING CUBIC FITS
C
20 DO 30 I=2,N1
I1 = I - 1
I2 = I + 1
DX1 = X (I1) - X (I)
DX2 = X (I2) - X (I)
DF1 = 2.0 *((F(I1) - F(I)) / DX1 - FP(I)) / DX1
DF2 = 2.0 *((F(I2) - F(I)) / DX2 - FP(I)) / DX2
FPPP(I)= 3.0 *(DF1 - DF2) / (DX1 - DX2)
FPP (I)= DF1 - DX1 * FPPP (I) / 3.0
30 CONTINUE
FPPP(N)= FPPP (N1)
FPP (1)= FPP (2) + (X (1) - X (2)) * FPPP (2)
FPP (N)= FPP (N1) + (X (N) - X (N1)) * FPPP (N1)
C
RETURN
END
C
DATA SET KCBCEDDY AT LEVEL 001 AS OF 08/24/84
C DATA SET KCBCEDDY AT LEVEL 002 AS OF 08/24/84
C DATA SET KCBCEDDY AT LEVEL 003 AS OF 04/05/84
C
SUBROUTINE EDDY
C
C CALCULATE EDDY VISCOSITY USING C.S. TWO-LAYERS EDDY
C
VISCOSITY FORMULA
COMMON /BLCO/ NX,NXT,NP,NPT,NTR,IT,INVRS,NS,IP
COMMON /BLG1/ F(101,2),U(101,2),V(101,2),W(101,2),B(101,2)
COMMON/BLC7/ C(100,100),D(100),DB(100),DBP(100),UE0(100),GI
COMMON/EDDY1/ RL,RX,SQRX,RXNTR,GNTR,GNTRS(100),
+ ALFAS(100),FFS(100),RTS(100),IEDY,NXSPT
COMMON /GRD / ETA(101),DETA(101),A(101)
COMMON /GTY / X(101),UE(100),P1(100),P2(100),CEL,CELH
DIMENSION FINT(101)
C
JO=1
UDEL=0.995*U(NP,2)
DO 10 J=1,NP
IF(U(J,2).LT.UDEL) JJ=J
10 IF(U(J,2).LT.0.0) JO=J
EDEL=ETA(JJ)+(ETA(JJ+1)-ETA(JJ))/(U(JJ+1,2)-U(JJ,2))
+ *(UDEL-U(JJ,2))
DO 15 J=1,NP
ETADEL=ETA(J)/EDEL
C
RETURN
END
IF(ETADEL > 1.0) ETADEL = 1.0

FINT(J) = 1.0 / (1.0 + 5.5 * ETADEL ** 6)

CALL AMEAN(1, JJ, ETA, FINT, 2)

RU = RL

IF (IT.GT. 1) GO TO 20

ALFAS(NX) = ALFAS(NX-1)

FFS(NX) = FFS(NX-1)

RTS(NX) = RTS(NX-1)

C

GMTR = GMTR(NX)

IF (NX.LE. NS) RU = RL * UE(NX)

RL2 = SQRT(RU * X(NX))

RL4 = SQRT(RL2)

RL216 = 0.16 * RL2

VMAX = 0.5 * (V1,2) + V1,1

DO 30 J=2, NP

VM = 0.5 * (V(J,2) + V(J,1))

IF (VM.GT. VMAX) VMAX = VM

30 CONTINUE

IF (IEDY.EQ. 0) GO TO 35

IF (IT.LE. 1 .OR. GMTR .LT. 0.85 .OR. NX .LE. NTR+3) GO TO 35

C

MODIFY ALFA USING SIMPSON’S ARGUMENTS

C

CALL SIMPSON

35 ALFA = ALFAS(NX)

EDVO = ALFA * RL2 * GMTR * (U(NP,2)*ETA(NP) - F(NP,2))

DO 40 J=2, NP

JJ = J

YBA = RL4 * SQRT(VMAX) / 26.0 * ETA(J)

EL = 1.0

IF (YBA .LT. 10.0) EL = 1.0 - EXP(-YBA)

EDVI = RL216 * GMTR * (EL * ETA(J) ** 2 * ABS(V(J,2))

IF (EDVI.GT. EDVO) GO TO 70

B(J,2) = 1.0 + EDVI * FINT(J)

IF (B(J,2).LT. B(J-1,2)) B(J,2) = B(J-1,2)

40 CONTINUE

JM = 2

BJM = B(2,2)

DO 50 J=2, NP

IF (BJM.GT.B(J,2)) GO TO 50

JM = J

BJM = B(J,2)

50 CONTINUE

GOTO 90

70 DO 80 J=JJ, NP

80 B(J,2) = 1.0 + EDVO * FINT(J)

C

B(J,2) = 1.0 + EDVO

C

90 CONTINUE

B(1,2) = 1.0

C

JJ = 1

DO 100 J=2, NP

100 CONTINUE
IF(U(J,2) .LT. 0.0) JJ = J
100 CONTINUE
IF(JJ.EQ.1) GO TO 110
C
C IN THE SEPARATED REGION, EDGY VISCOSITY IS SET EQUAL TO
C THAT IN THE PREVIOUS STATION TO AVOID NUMERICAL TROUBLE
JJP3 = JJ + 3
JJP3 = MIN(JJP3, NP)
CALL AMEAN(1,JJP3,ETA,B(1,2),2)
110 CALL AMEAN(1,NP,ETA,B(1,2),1)
C
RETURN
END
DATA SET KCBCEXTRAP AT LEVEL 001 AS OF 08/24/84
DATA SET KCBCEXTRAP AT LEVEL 001 AS OF 08/24/84
DATA SET KCBCEXTRAP AT LEVEL 008 AS OF 02/13/84
SUBROUTINE EXTRAP(NX,NXTE,X,Y)
C
C EXTRAPOLATE DATA USING PARABOLIC FORMULA
DIMENSION X(101),Y(101)
C
Y1 = Y(NX-2)
Y2 = Y(NX-1)
X1 = X(NX-2)
X2 = X(NX-1)
X3 = X(NXTE)
XI = X1 - X3
X2 = X2 - X3
BB = (Y1-Y2)/(X1**2 - X2**2)
AA = Y1 - BB * X1**2
DO 10 I=NX,NXTE
X1 = X(I) - X3
Y(I) = AA + BB * X1**2
10 CONTINUE
RETURN
END
DATA SET KBCFILLUP AT LEVEL 001 AS OF 08/24/84
DATA SET KBCFILLUP AT LEVEL 001 AS OF 08/24/84
DATA SET KBCFILLUP AT LEVEL 007 AS OF 04/05/84
SUBROUTINE FILLUP(INDEX)
COMMON /BLCO/ NX,NX,NP,NP,NPT,NTR,IT,INVRS,NS,IP
COMMON /BLC1/ F(101,2),U(101,2),V(101,2),W(101,2),B(101,2)
COMMON /GRD/ ETA(101),DETA(101),A(101)
C
C IF(NP.GE.NPT) RETURN
IF(INDEX.EQ.2) GOTO 10
C
C DEFINE PROFILES FOR B. L. GROWTH
NP1 = NP + 1
NP = NP + 2
NP = MIN(NP, NPT)
NM = NP
GOTO 20
C
FILL UP PROFILES BEFORE MOVING TO THE NEXT STATION

10 NP1 = NP + 1
NM = NPT

20 DO 30 J=NP1,NM
F(J,2) = F(J-1,2) + DELTA(J-1)*U(J-1,2)
U(J,2) = U(J-1,2)
V(J,2) = 0.0
B(J,2) = B(J-1,2)
W(J,2) = W(J-1,2)
30 CONTINUE

RETURN
END

DATA SET KBCBHEADER AT LEVEL 001 AS OF 08/24/84
DATA SET KBCBHEADER AT LEVEL 001 AS OF 04/05/84
SUBROUTINE HEADER ( TITLE,SURFID,ISTRP )
COMMON /ISURF/ ISF

DIMENSION TITLE(20), SURFID(4)

10 FORMAT ( 1H1,20X,20A4 )
20 FORMAT ( 1H0,15X, 'BOUNDARY LAYER CALCULATION FOR ',
+ 'UPPER SURFACE ',10X,'ICYCLE=',I5 / 16X,71(1H-) / )
30 FORMAT ( 1H0,15X, 'BOUNDARY LAYER CALCULATION FOR ',
+ 'LOWER SURFACE ',10X,'ICYCLE=',I5 / 16X,71(1H-) / )

WRITE ( 6,10 ) TITLE
IF(ISF .EQ. 1) WRITE ( 6,20 ) ISTRP
IF(ISF .EQ. 2) WRITE ( 6,30 ) ISTRP

RETURN
END

DATA SET KCBCINPUT AT LEVEL 001 AS OF 08/24/84
DATA SET KCBCINPUT AT LEVEL 009 AS OF 08/24/84
SUBROUTINE INPUT(ITR,ISWPT,SURFID)
COMMON /BLCO/ NX,NXT,NP,NPT,NTR,IT,INVRS,NS,IP
COMMON /BLC1/ F(101,2),U(101,2),V(101,2),W(101,2),B(101,2)
COMMON /BLC2/ DELF(101),DELU(101),DELV(101),DELM(101)
COMMON /BLC7/ C(100,100),D(100),DB(100),DBP(100),UE0(100),GI
COMMON /BONV/ ITMAX,EPSL,EPST,CONV
COMMON /EDDY1/ RL,RX,SQRX,RXNTR,GNTR,GMTRS(100),
+ ALFAS(100),FFS(100),RTS(100),IEDY,NXSPT
COMMON /GTY/ X(101),UE(100),P1(100),P2(100),CEL,CHEL
COMMON /BLIN/ TITLE(20),XC(100),YC(100),ISG(100),DELS(100),
+ XCTR,XTR,ISTRP,ICYCLE,ICYTL,XCTRS(2),TRFIND(2)
COMMON /TRN/ PGAMTR,OMEGA,RTHETB,RTTRAN
COMMON /ISURF/ ISF
DIMENSION D1(100),D2(100),D3(100)
DIMENSION SURFID(4),XCS(100)
LOGICAL TRFIND

91
SEARCH FOR PRESSURE PEAK AS SWITCH POINT

UMAX = UE(1)
DO 50 I = 2, NXT
IF (UE(I) .LE. UMAX) GO TO 55
UMAX = UE(I)
50 CONTINUE
GO TO 60
55 NS = I - 4
60 IF (NS .GT. NSS) NS = NSS
IF (NS .LT. 3) NS = 3

CALCULATE THE PRESSURE PARAMETERS P1 + P2 FOR B. L. CALCULATION
 USING TRANSFORMED COORDINATES

CALL DIFF3 (NXT, X, UE, D1, D2, D3, 0)
DO 65 I = 2, NXT
P2(I) = X(I) * D1(I) / UE(I)
P1(I) = 0.5 * (1.0 + P2(I))
65 CONTINUE
P1(1) = 0.5 * (1.0 + P2(1))
XCMIN = X(I)
MIN = 1
DO 70 I = 1, NXT
IF(XCMIN .LT. X(I)) GOTO 70
XCMIN = XC(I)
MIN = I
70 CONTINUE
DO 80 I = 1, NXT
IF (I .GE. MIN) THEN
XCS(I) = XC(I)
ISG(I) = 1
ELSE
XCS(I) = -XCS(I)
ISG(I) = -1
END IF
80 CONTINUE
INVRS = NS + 1

SEARCH FOR TRANSITION LOCATION
ITRP1 = ITR + 1
GOTO (150, 95, 120, 150), ITRP1

TRANSITION LOCATION IS INPUT ( = XCTR)
95 DO 100 I = 1, NXT
IF(XCTR.LT.XCS(I)) GOTO 105
100 CONTINUE
TRANSITION LOCATION IS SET AT THE PRESSURE PEAK

TRANSITION LOCATION WILL BE CALCULATED BASED ON MICHEL CRITERION

TRANSITION LOCATION PROVISIONALLY FROM PREVIOUS CYCLE

CALCULATE GAMTR DISTRIBUTION
ALFAS(I) = 0.0168
GMTRS(I)= 1.0
ALFAS(NTR) = 0.0168
UEINTG = 0.0
U1 = 0.5/UETR
X1 = XTR
DO 230 I=NTR+1,NXT
U2 = 0.5/U(E(I))
X2 = X(I)
UEINTG = UEINTG+(U1+U2)*(X2-X1)
U1 = U2
X1 = X2
GG = GGFT*UEINTG*(X(I)-XTR)*UE(I)**3
IF(GG .GT. 10.0) GO TO 250
GMTRS(I) = 1.0-EXP(-GG)
230 CONTINUE
250 CONTINUE
C
C GENERATE B. L. ETA GRIDS + INITIAL VELOCITY PROFILES
C
CALL INTL(ETAE,DETA1,VGP)
DO 260 I=1,NXT
UEO(I) = UE(I)
260 CONTINUE
C
PRINT OUT INPUT DATA
C
IF (ICYCLE .GE. ICYTL-1 .OR. IP .GE. 0) THEN
CALL HEADER( TITLE,SURFID,ISTRP )
WRITE(6,1002) NXT,ITR,IP,NS,NTR,ISWPT
WRITE(6,1003) VGP,DETA1,RL,XCTR,OMEGA,PGAMTR
ELSE
IF (ISF.EQ.1) WRITE(6,1008) ICYCLE
IF (ISF.EQ.2) WRITE(6,1009) ICYCLE
ENDIF
IF (NTR.LT.NXT) THEN
IF (ITR.EQ.1) WRITE (6,1005) XCTR,XTR,NTR
IF (ITR.EQ.2) WRITE (6,1006) XCTR,XTR,NTR
IF (TRFIND(ISF)) WRITE(6,1007) XCTR,XTR,NTR
ENDIF
RETURN
C
2 FORMAT(20A4)
3 FORMAT(10I5)
4 FORMAT(6F10.0)
1001 FORMAT(1H1,20X,20A4)
1002 FORMAT(1H0,10H NXT = ,I5,7X,10H ITR = ,I5,7X/
+ 1H ,10H IP = ,I5,7X,10H NS = ,I5,7X/
+ 1H ,10H NTR = ,I5,7X,10H ISWPT = ,I5)
1003 FORMAT(1H0,10H VGP = ,E12.4,10H DETA1 = ,E12.4/
+ 1H ,10H RL = ,E12.4,10H XCTR = ,E12.4/
+ 1H ,10H OMEGA = ,E12.4,10H PGAMTR = ,E12.4)
1004 FORMAT(1H0,3X,2H NLX,1X,2H Y,11X,2HC,11X,2HYC,11X,2HE,11X,2HEP1,
+ 11X,2HEP2/(1H ,3X,I3,6E13.5))
1005 FORMAT(/3X, 'BEGIN OF TRANSITION IS BEING INPUT AT X/C = ', F8.4,4X)
S/C = ',F8.4,4X, 'NTR = ',I3/
1006 FORMAT(/3X,'BEGIN OF TRANSITION IS SET AT PRESSURE PEAK, X/C = ',
+ 'F8.4,4X,'S/C = ',F8.4,4X,'NTR = ',I3/
1007 FORMAT(/3X,'BEGIN OF TRANSITION IS PROVISIONALLY TAKEN FROM ',
+ 'PREVIOUS CYCLE: X/C = ',F8.4,4X,'S/C = ',F8.4,4X,'NTR = ',I3/
1008 FORMAT(/3X,'UPPER SURFACE CALCULATIONS OF CYCLE',I3)
1009 FORMAT(/3X,'LOWER SURFACE CALCULATIONS OF CYCLE',I3)
END
DATA SET KCBCINTL AT LEVEL 001 AS OF 08/24/84
SUBROUTINE INTL(ETAE,DETA1,VGP)
COMMON /BLC0/ NX,NXT,NP,NPT,NTR,IT,INVRS,NS,IP
COMMON /BLC1/ F(101,2),U(101,2),V(101,2),W(101,2),B(101,2)
COMMON /BLC2/ DELF(101),DELU(101),DELV(101),DELW(101)
COMMON /BLC6/ S1(101),S2(101),S3(101),S4(101),S5(101),S6(101),
+ S7(101),S8(101),R1(101),R2(101),R3(101),R4(101)
COMMON /BONV/ ITMAX,EPSL,EPST,CONV
COMMON /GRD/ ETA(101),DETA(101),A(101)
COMMON /GTY/ X(101),UE(100),P1(100),P2(100),CEL,CELH

GENERATE THE GRID

DETA(1) = DETA1
IF(VGP .LT. 1.0) VGP = 1.0
IF((VGP-1.0) .LE. 0.001) GO TO 10
NP = ALOG((ETAE/DETA(1))*(VGP-1.0)+1.0)/ALOG(VGP) + 1.001
GO TO 20
10 NP = ETA(E/DETA(1) + 1.001
20 IF(NP .LE. NPT) GO TO 30
WRITE(6, 150 )
STOP
30 ETA(1) = 0.0
DO 40 J=2,NPT
ETA(J) = ETA(J-1) + DETA(J-1)
DETA(J) = VGP*DETA(J-1)
A(J) = 0.5*DETA(J-1)
40 CONTINUE

GENERATE INITIAL VELOCITY PROFILE
80 DO 90 J=1,NP
ETA(B) = ETA(J)/ETA(NP)
ETA(B) = ETA(J)/ETA(NP)
F(J,2) = 0.25*ETA(NP)*ETA(2)*(3.0 - 0.5*ETA(2))
U(J,2) = 0.5*ETA(2)*(3.0 - ETA(2))
V(J,2) = 1.5*(1.0 - ETA(2))/ETA(NP)
B(J,2) = 1.0
W(J,2) = 1.0
90 CONTINUE
NX = 1
IT = 0
120 IF(IT .LT. ITMAX) GO TO 130
130
STOP

CONTINUE

C

DO 140 J = 2, NP
FB = 0.5*(F(J,2) + F(J-1,2))
UB = 0.5*(U(J,2) + U(J-1,2))
VB = 0.5*(V(J,2) + V(J-1,2))
USB = 0.5*(U(J,2)**2 + U(J-1,2)**2)
DERBV = (B(J,2)*V(J,2) - B(J-1,2)**2*V(J-1,2))/DETA(J-1)
FVB = 0.5*(F(J,2)*V(J,2) + F(J-1,2)**2*V(J-1,2))

C

S1(J) = 0.5*P1(NX)*F(J,2) + B(J,2)/DETA(J-1)
S2(J) = 0.5*P1(NX)*F(J-1,2) - B(J-1,2)/DETA(J-1)
S3(J) = 0.5*P1(NX)*V(J,2)
S4(J) = 0.5*P1(NX)*V(J-1,2)
S5(J) = -P2(NX)*U(J,2)
S6(J) = -P2(NX)*U(J-1,2)
CRB = -P2(NX)
R2(J) = CRB - (DERBV + P1(NX)*FVB - P2(NX)*USB)

C

R1(J) = F(J-1,2) - F(J,2) + DETA(J-1)*UB
R3(J-1)= U(J-1,2) - U(J,2) + DETA(J-1)*VB

CONTINUE

R1(1) = 0.0
R2(J) = 0.0
R3(NP) = 0.0
CALL SOLV3
IF(ABS(DELV(1)) .GT. EPSL ) GO TO 120
CALL FILLUP(2)
CALL OUTPUT(1)

C

RETURN

C

FORMAT(1H0,37HNP EXCEEDED NPT -- PROGRAM TERMINATED)
END

DATA SET KCBCINTRP3 AT LEVEL 001 AS OF 08/24/84
DATA SET KCBCINTRP3 AT LEVEL 001 AS OF 08/24/84
DATA SET KCBCINTRP3 AT LEVEL 003 AS OF 04/05/84
SUBROUTINE INTPR3 (N1,X1,F1,FP1,FPP1,FPPP1,N2,X2,F2)

C

CUBIC INTERPOLATION

C

GIVEN THE VALUES OF A FUNCTION (F1) AND ITS DERIVATIVES
AT N1 VALUES OF THE INDEPENDENT VARIABLE (X1)

C

FIND THE VALUES OF THE FUNCTION (F2)
AT N2 VALUES OF THE INDEPENDENT VARIABLE (X2)

C

X2 CAN BE IN ARBITRARY ORDER

C

DIMENSION X1(101),F1(101),FP1(101),FPP1(101),FPPP1(101),X2(101),
     + F2(101)
DATA EPS /1E-04/
C

96
JT = 2
DO 10 I = 1, N2
XM = X2(I)
DO 20 J = JT, N1
J1 = J - 1
IF (X1(J).GE.XM) GO TO 30
20 CONTINUE
J = N1
JT = J
DXX = X2(I) - X1(J1)
DXX2 = DXX * DXX / 2.
DXX3 = DXX2 * DXX / 3.
DXX = X2(I)
DXX2 = DXX * F2(I)
DXX3 = DXX2 * FPP1(J1) + DXX3 * FPPP1(J1)
10 CONTINUE
RETURN
END
DATA SET KBCJOIN AT LEVEL 001 AS OF 08/24/84
DATA SET KBCJOIN AT LEVEL 001 AS OF 08/24/84
DATA SET KBCJOIN AT LEVEL 001 AS OF 02/20/84
APPLICATION JOIN(INDEX)
COMMON /BLCO/ NX,NXT,NP,NPT,NTR,IT,INVRS,NS,IP
COMMON /BLG1/ F(101,2),U(101,2),V(101,2),W(101,2),B(101,2)
COMMON /BLG7/ C(100,100),D(100),DB(100),DBP(100),UEO(100),GI
COMMON /EDDY/ RL,RX,SQRX,RXNTR,GIDX,GMSTR(100)
+ ALFAS(100),FFS(100),RTS(100),IEDX,NXSPT
COMMON /GTY/ X(101),UE(100),P1(100),P2(100),CEL,CELH
COMMON /GRD/ ETA(101),DETA(101),A(101)
COMMON /SVE/ FS(101),US(101),VS(101),BS(101),WS(101)
COMMON /SHRY/ VW(100),ITP(100),ISL(100),DLS(100),CF(100),
+ THT(100),NPSTR(100)
C INDEX = 1 FOR THE FIRST SWEEP
C INDEX = 2 FOR SUBSEQUENT SWEEP
C CALL COMPGI
GII = C(NX,NX)
UES = GI / (1.0 - DLS(NX) * SQRT(RL) * CII)
IF(INDEX.EQ.1) GOTO 15
C C RETRIEVE PROFILES AT STATION NS FOR INVERSE B. L.
C CALCULATION
DO 10 J=1, NPT
F(J,2) = FS(J)
U(J,2) = US(J)
V(J,2) = VS(J)
W(J,2) = WS(J)
B(J,2) = BS(J)
10 CONTINUE
UES = UES/W(1,2)
SQS = 1.0
GOTO 30
15 CONTINUE
SQS = 1.0 / SQRT(UES)
DO 20 J=2, NPT
INT18350
INT18360
INT18370
INT18380
INT18390
INT18400
INT18410
INT18420
INT18430
INT18440
INT18450
INT18460
INT18470
INT18480
INT18490
INT18500
INT18510
INT18520
INT18530
INT18540
INT18550
INT18560
INT18570
INT18580
INT18590
INT18600
INT18610
INT18620
INT18630
INT18640
INT18650
INT18660
INT18670
INT18680
INT18690
INT18700
INT18710
INT18720
INT18730
INT18740
INT18750
INT18760
INT18770
INT18780
INT18790
INT18800
INT18810
INT18820
INT18830
INT18840
INT18850
INT18860
INT18870
INT18880
INT18890
INT18900
ETA(J) = ETA(J) * IQS
DETA(J-1) = ETA(J) - ETA(J-1)
A(J) = 0.5 * DETA(J-1)

20 CONTINUE
C
30 DO 60 J=1,NPT
U(J,2) = U(J,2)*UES
W(J,2) = UES * W(J,2)
F(J,2) = F(J,2)*IQS*UES
V(J,2) = V(J,2)/IQS*UES
60 CONTINUE
UE(NX) = W(1,2)
RX = UE(NX)*X(NX)*RL
SQRX = SQRD(RX)
C
C IF(NX.GT.NTR) CALL EDDY
CALL FILLUP(2)
IF(INDEX.EQ.2) GOTO 70
C
C STORE PROFILES AT STATION NS FOR THE NEXT SWEEP
DO 65 J=1,NPT
FS(J) = F(J,2)
US(J) = U(J,2)
VS(J) = V(J,2)
WS(J) = W(J,2)
BS(J) = B(J,2)
65 CONTINUE
DO 80 J=1,NPT
F(J,1) = F(J,2)
U(J,1) = U(J,2)
V(J,1) = V(J,2)
W(J,1) = W(J,2)
B(J,1) = B(J,2)
80 CONTINUE
RETURN
END

DATA SET KCBCMAIN AT LEVEL 005 AS OF 09/18/84
C
C PROGRAM MAIN
C
C SUBROUTINE CASBLP(K2,XP,YP,XMP,YMP,XS,YS,DLS,VC,DBPP
+ ,RN,NBL,ITRI,XCTRI,CASEID)
COMMON /BLCO/ NX,NXT,NP,NPT,NTR,IT,INVR,NS,IP
COMMON/BLIN/ TITLE(20),XC(100),YC(100),ISG(100),DELS(100),
+ XCTR,XTR,ISTRP,ICYCLE,ICYTL,XCTRS(2),TRFIND(2)
COMMON/EDDY1/RL,RX,SQRX,SNXTR,GLTR,GLTRS(100),
+ ALFAS(100),FFS(100),RTS(100),IEDY,NXSPT
COMMON/BLOW/ VN(100)
COMMON/TRN/ PGNTR,OMEGA,RTHETB,RTRANB
COMMON/PLTP/NVP(2),NXVP(20,2),ICC
DIMENSION CASEID( 20 ), XCTRI( 2 ), ITRI ( 2 )
DIMENSION XP ( 100 ), YP ( 100 ), XMP ( 100 )
DIMENSION YMP ( 100 ), VC ( 100 ), SM ( 100 )
DIMENSION XS ( 100 ), YS ( 100 ), NBL ( 2 )
DIMENSION VT ( 100 ), S ( 100 ), DLS ( 100 )
DIMENSION DLS ( 100 ), XO ( 100 ), YO ( 100 )
DIMENSION D1 (100), D2 (100), D3 (100)
DIMENSION XPS (100), YPS(100), DBPP(100)

C 10 FORMAT (20A4)
20 FORMAT (415)
30 FORMAT (315,3F10.0)
40 FORMAT (6F10.5)
100 FORMAT (1H1,5X,20A4)
110 FORMAT (1H0,6X,'CYCLE NO. = ',I5,
+ 4X,'JOB COMPLETED. ')
120 FORMAT (1H0,6X,'FINISHED TOTAL NUMBER OF CYCLES = ',I5,
+ 4X,'JOB COMPLETED. ')
130 FORMAT (1H0,'THE UPDATED DISPLACEMENT SURFACE',/1X,2HNX,
+ 9X,2HXP,12X,2HYP,11X,3HDLX,10X,4HDBPP,12X,2HVN)
140 FORMAT (I5,5E14.6)
150 FORMAT (1H0.6X,'READ IN CONTROL POINTS DATA',/1X,2HNX,9X,
+ 3HXMP,11X,3HYP,12X,2HSM,13X,2HVC)
160 FORMAT (I5,4E14.6)
170 FORMAT (1H0,6X,'INTERPOLATED ORIGINAL GEOMETRY DATA',/1X,
+ 2HNY,9X,2HXP,12X,2HYP,12X,1HS,14X,2HVT)
180 FORMAT (I5,4E14.6)

C GRANG(X1,X2,X3,Y1,Y2,Y3,X0)= (X0-X2)*(X0-X3)/(X1-X2)/(X1-X3)*Y1
+ +/(X0-X1)*(X0-X3)/(X2-X1)/(X2-X3)*Y2+(X0-X1)*(X0-X2)
+ /(X3-X1)/(X3-X2)*Y3
ISWPT = 1
IEDY = 1
NBL(1) = 91
NBL(2) = 71
C WRITE ( 6,100 ) CASEID
5 CONTINUE
ISTRP = ICYCLE
NN = K2 - 1
NT = K2
C WRITE ( 6,110 ) ICYCLE
C INTERPOLATE OUTPUT CONTROL POINTS TO ORIGINAL GEOMETRY
C DO 15 I = 1 , NT
XPS(I) = XP(I)
YPS(I) = YP(I)
15 CONTINUE
S(1) = 0.0
DO 50 I = 2 , NT
S(I) = S(I-1) + SQRT((XP(I)-XP(I-1))**2 +
+ (YP(I)-YP(I-1))**2)
50 CONTINUE
SM(1) = SQRT((XMP(1)-XP(1))**2 + (YMP(1)-YP(1))**2)
DO 60 I = 2 , NN
SM(I) = SM(I-1) + SQRT((XMP(I)-XMP(I-1))**2 +
+ (YMP(I)-YMP(I-1))**2)
60 CONTINUE
C CALL AMEAN(NN-10,NN,SM,VC,1)
CALL AMEAN(1,NN,SM,VC,1)
SNT = S(NT)
\[ SM(NT) = SM(NN) + \sqrt{((XMP(NN) - XP(NT))^2 + (YMP(NN) - YP(NT))^2)} \]
\[ SM(NN) = S(NT) / SM(NT) \]

**DO 65 I = 1, NN**

\[ SM(I) = SM(I) * SMNT \]

**CALL DIFF3(NN, SM, VC, D1, D2, D3, 0)**

**CALL INTRP3(NN, SM, VC, D1, D2, D3, NT, S, VT)**

**C PRINT OUT INPUT DATA**

**C**

WRITE(6,150)

**C**

WRITE(6,160) (I, XMP(I), YMP(I), SM(I), VC(I), I=1,NN)

**WRITE(6,170)**

**WRITE(6,180)**

**C SMOOTH THE CALCULATED DISPLACEMENT THINKNESS**

**C**

**CALL SMFIT(1, NT, S, DLS, D1, 2)**

**C ADD DISPLACEMENT THINKNESS ON THE ORIGINAL BODY**

**C**

**DO 70 I = 1, NT**

DLSP(I) = DLS(I)

**END IF**

**CALL SMFIT (1, NT, XS, YS, D1, 2)**

**C**

**DATA SET KCBCMAIN2 AT LEVEL 001 AS OF 08/24/84**

**C**

**DATA SET KCBCMAIN2 AT LEVEL 001 AS OF 08/24/84**

**C**

**DATA SET KCBCMAIN2 AT LEVEL 010 AS OF 04/06/84**

**SUBROUTINE MAIN2( ITR, ISWPT, SURFID)**

**COMMON /BLC0/ NX, NXT, NP, NPT, NTR, IT, INVRS, NS, IP**

**COMMON /BLC1/ F(101,2), U(101,2), V(101,2), W(101,2), B(101,2)**

**COMMON /BLC2/ DELF(101), DELU(101), DELV(101), DELW(101)**
COMMON /BLC7/  C(100,100),D(100),DB(100),DBP(100),UEO(100),GI
COMMON /BONV/  ITMAX, EPSL, EPSST, CONV
COMMON /EDDY1/  RL, RX, SQRX, RXNTR, GMTR, GMTRS(100)
    +  ALFAS(100), FFS(100), RTS(100), IEDY, NXSPT
COMMON /GY/   X(101), UE(100), P1(100), P2(100), CEL, CELH
COMMON /SMRY/  W(100), ITP(100), ISL(100), DLS(100), CF(100),
    +  THT(100), NPSTR(100)
COMMON /BLIN/  TITLE(20), XC(100), YC(100), ISG(100), DELS(100),
    +  XCTR, XTR, ISTRP,ICYCLE, ICYTL, XCTRS(2), TRFIND(2)
COMMON /BLC9/  UEB(IOO), CFS(IOO)
COMMON /TRN/  PGAMTR, OMEGA, RTHETB, RTRANB
COMMON /ISURF/ IST
COMMON /PLOT/ NVP(2), NXVP(20,2), ICC
DIMENSION SURFID(4), RTSS(11)
LOGICAL SMOOT, SEPART, HOMOPY
DATA RTSS/0.0,0.1,0.2,0.3,0.4,0.5,0.6,0.7,0.8,0.9,1.0/

C
GRANG(X1,X2,X3,Y1,Y2,Y3,X0) = (X0-X2)*(X0-X3)/(X1-X2)/(X1-X3)*Y1
    + (X0-X1)*(X0-X3)/(X2-X1)/(X2-X3)*Y2+(X0-X1)*(X0-X2)
    + /(X3-X1)/(X3-X2)*Y3
ISWP = 0
INDEX = 1
IGROWT = 2
NXSPT = NXT + 1
10 CALL JOIN(INDEX)
NXSTOP = NXT-1
IF (NS .GE. NTR) GOTO 15
15 ISWP = ISWP + 1
20 NX = NX + 1
HOMOPY = .FALSE.
25 CEL = 0.5*(X(NX)+X(NX-1))/(X(NX)-X(NX-1))
P1(NX) = 0.5
P2(NX) = 0.0
CELH = 0.5*CEL
30 IT = 0
CALL COMPGI
IGROW=1
70 IT = IT + 1
RX = UE(NX)*X(NX)*RL
SQRX = SQRT(RX)
C
IF(IT .LE. ITMAX) GO TO 80
IF(HOMOPY) GO TO 72
IRC = 1
RT = RTSS(IRC)
HOMOPY = .TRUE.
UEREF = UEO(NX-1)
UESAVE = UEO(NX)
UEO(NX) = RT*UESAVE+(1.0-RT)*UEREF
DO 61 J=1,NP
   F(J,2) = F(J,1)
   U(J,2) = U(J,1)
   V(J,2) = V(J,1)
   W(J,2) = W(J,1)
   RX = UE(NX)*X(NX)*RL
   SQRX = SQRT(RX)
C
101
B(J,2) = B(J,1)

61 CONTINUE
GO TO 30

C
72 NXSTOP = NX - 1
CALL AMEAN(NS,NXSTOP,X,CF,1)
C CALL AMEAN(NS,NXSTOP,X,VW,1)
C CALL HEADER(TITLE,SURFID,ISTRP)
WRITE (6, 250) ISWP
WRITE (6, 260) (M,XC(M),X(M),CF(M),DLS(M),THT(M),UE(M),
+ UE0(M),D(M),DB(M),GMTRS(M),ITP(M),NPSTR(M),M=1,NXSTOP)
WRITE(6, 270) NX
STOP

80 CONTINUE
IF(NX .GT. NTR) GOTO 100
C
C LAMINAR FLOW CALCULATION
C
CALL COEF(GAMMA1,GAMMA2)
CALL SOLV4(GAMMA1,GAMMA2)
UE(NX) = U(NP,2)
IF(ABS(DELV(1)) .GT. EPSL) GO TO 70
C
C CHECK ON LAMINAR FLOW SEPARATION. IF SEPARATION OCCURS, ASSIGN BEGIN
C OF TRANSITION TO THAT POINT AND RECOMPUTE THE CURRENT STATION NX
C
IF(V(1,2).GT.0.0.OR. ITR.NE.3) GOTO 110
CALL TRNS(ICODE)
GOTO 25
C
C TURBULENT FLOW CALCULATION
C
100 CONTINUE
CALL EDDY
CALL COEF(GAMMA1,GAMMA2)
CALL SOLV4(GAMMA1,GAMMA2)
UE(NX) = U(NP,2)
VM = AMAX1(V(1,2),1.0)
IF(ABS(DELV(1)/VM) .GT. EPST) GO TO 70
110 CONTINUE
C
C CHECK FOR B. L. GROWTH
C
IF(NP.GE.NP7) GOTO 120
IF(ABS(V(NP,2)) .LT. 0.0005 .AND. ABS(1.0-U(NP-2,2)/U(NP,2))
+ .LT. 0.0035 .OR. IGROW.GT.IGROWT) GOTO 120
CALL FILLUP(1)
IGROW=IGROW+1
IT = 1
GO TO 70
C
120 CONTINUE
CALL FILLUP(2)
CALL OUTPUT(2)
IF(NX.GE.NTR .OR. ITR.EQ.0) GOTO 150
IF(NX.LT.3 .OR. ITR.NE.3) GOTO 150
C
C CALCULATE TRANSITION LOCATION USING MICHEL METHOD
C
CALL TRNS(ICODE)
IF(ICODE.EQ.0) GOTO 150
C
C TRANSITION OCCURS BASED ON MICHEL CRITERION AT STATION NX
C
CALCULATE TRANSITION LOCATION USING MICHEL METHOD

IT = 0
IGROW = 1
GOTO 70

150 CONTINUE
IF(.NOT. HOMOPY ) GOTO 154
IF( RT .GT. 0.9999) GOTO 154
IRC = IRC +1
RT = RTSS(IRC)
UEO(NX) =RT*UESAVE + (1.0-RT)*UEREF
GOTO 30

154 CONTINUE
IF(NX .LT. NXSTOP) GOTO 20
C
C THE B. L. CALCULATION FOR THE CURRENT SWEEP IS COMPLETED.
C
CHECK FOR THE CONVERGENCE AND , IT NOT, MOVE TO THE NEXT
C SWEEP.
C
C 160 CONTINUE
D(NXT) = GRANG(X(NXT-3),X(NXT-2),X(NXT-1),D(NXT-3),D(NXT-2),
+ D(NXT-1),X(NXT))
DLS(NXT)= GRANG(X(NXT-3),X(NXT-2),X(NXT-1),DLS(NXT-3),DLS(NXT-2),
+ DLS(NXT-1),X(NXT))
UE(NXT) = GRANG(X(NXT-3),X(NXT-2),X(NXT-1),UE(NXT-3),
+ UE(NXT-2),UE(NXT-1),X(NXT))
DO 165 I = 1, NXSTOP
CFS(I) = CF(I)
CALL AMEAN(NS,NXSTOP,X,CF,1)
CALL AMEAN(NS,NXSTOP,X,VW,1)
CALL HEADER( TITLE,SURFID,ISTRP )
IF(ICYCLE .LT. ICYTL-1 .AND. IP .LT. 0)GO TO 170
WRITE (6, 250 ) ISWP
WRITE (6, 262 ) 1,XC(1),X(1),CF(1),DLS(1),THT(1),UE(1),
+ UEO(1),0.0,GMTRS(1),ITP(1),NPSTR(1)
WRITE (6, 264 ) (M,XC(M),X(M),CF(M),DLS(M),THT(M),UE(M),
+ UEO(M),DLS(M)/THT(M),GMTRS(M),ITP(M),NPSTR(M),M=2,NXSTOP)
IF (((ICYCLE.EQ.ICYTL).AND.(IP .EQ. -2).AND.(NVP(ISF).NE.0))) THEN
WRITE(12,800) NS+1,NTR,XCTR
WRITE(12,810) (XC(M),X(M),CF(M),GMTRS(M),ISG(M),
+ M=1,NXSTOP)
2 FORMAT(2I5,F10.6)
810 FORMAT(5E15.5,I5)
END IF
170 CONTINUE
C
DMAX = D(1)
DDMAX = ABS(D(1) - DB(1))
DO 180 I = 2,NXT
DMAX = AMAX1( DMAX,D(I) )
DD = ABS(D(I) - DB(I))
DDMAX = AMAX1( DDMAX,DD )
180 CONTINUE
IF ( ABS( DDMAX / DMAX ) .LE. 0.0050 ) RETURN

C UPDATE D FOR THE NEXT SWEEP
C
IF (ISWP .GT. 1) GO TO 195
DO 190 I = NS , NXT
190 D(I) = D(I) * (1.0 + OMEGA*(UE(I)/UE0(I) - 1.0))
GO TO 205
195 IF (ISWP .EQ. 2) GOTO 205
DO 200 I = NS , NXT
200 D(I) = D(I) * (1.0 + OMEGA*(UE(I)/UEB(I) - 1.0))
205 IF(ISWP = GE. ISWPT ) RETURN
NX = NS
NP = NPSTR(NX)
INDEX = 2
DO 210 I= 1,NXT
DB(I) = D(I)
UB(I) = UE(I)
210 CONTINUE
GOTO 10

C END

250 FORMAT(1HO,' *** SUMMARY OF INVERSE BOUNDARY LAYER SOLUTIONS. ***',/122540
+ 1HO,4X,'ISWP = ',I3)/
260 FORMAT(1HO,4X,2HNNX,5X,3HC/C,9X,1HX,9X,2HC,8X,
+ 3HDL,8X,3HTHT,9X,2HUE, 8X,3HUE0,10X,1HD,9X,2HDB,3X,
+ 4HGMR,4X,2HIT,1X,2HNP/(1H ,3X,I3,F10.5,8E11.4,F8.4,2I3))
262 FORMAT(1HO,4X,2HNNX,6X,3HC/C,11X,1HX,10X,2HC,9X,
+ 3HDL,9X,3HTHT,10X,2HUE,9X,3HUE0,11X,1HH,8X,
+ 4HGMR,4X,2HIT,1X,2HNP/(1H ,3X,I3,9E12.4,2I3))
264 FORMAT(1HO,4X,I3,9E12.4,2I3)
270 FORMAT(1HO,' *** ITERATIONS EXCEEDED ITMAX AT NX = ',IS,' ***',/
+ 1HO,' *** CALCULATIONS STOP. ***')
END

C DATA SET KCBCOUTPUT AT LEVEL 001 AS OF 08/24/84
C DATA SET KCBCOUTPUT AT LEVEL 001 AS OF 08/24/84
C DATA SET KCBCOUTPUT AT LEVEL 002 AS OF 02/22/84
SUBROUTINE OUTPUT(INDEX)
COMMON /BLC0/ NX,NXT,NP,NPT,NTR,IT,INVRS,NS,IP
COMMON/BLIN/ TITLE(20),XC(100),YC(100),ISG(100),DELS(100),
+ XCTR,XTR,ISTRP,ICYCLE,ICYTL,XCTRS(2),TRFIND(2)
COMMON /BLC1/ F(101,2),U(101,2),V(101,2),W(101,2),B(101,2)
COMMON /BLC7/ C(100,100),D(100),DB(100),DBP(100),UE0(100),GI
COMMON/EDDY1/ RL,RX,SQRX,RNXTR,GNTR,GMTRS(100)
+ ALFAS(100),FFS(100),RTS(100),IEDY,NXSPT
COMMON /GTY/ X(101),UE(100),P1(100),P2(100),CEL,CELH
COMMON /GRD/ ETA(101),DETA(101),A(101)
COMMON /SMRY/ VW(100),ITP(100),ISL(100),DLS(100),CF(100),THT(100)
+ NPSTR(100)
COMMON /ISURF/ ISF
COMMON/PL0T/NVP(2),NXVP(20,2),ICC

ITP(NX) = IT
NPSTR(NX)=NP
IF(NX.GT.1) GOTO 5
DLS(NX)= 0.0
VW(NX) = 0.0
D(NX) = 0.0
THT(NX)= 0.0
CF(NX) = 0.0
VW(NX) = 0.0
GOTO 150
5 GOTO (10,100,200), INDEX

CALCULATE B. L. PARAMETERS FOR TRANSFORMED COORDINATES
10 CONTINUE
CF(NX) = 2.0 * V(1,2) * B(1,2)/SQRX
VW(NX) = UE(NX) * SQRT(UE(NX)/X(NX)) * V(1,2)
DLS(NX)= X(NX)/SQRX * (ETA(NP)-F(NP,2))
D(NX) = UE(NX) * DLS(NX) * SQRT(RL)
U1 = U(1,2) * (1.0 -U(1,2))
SUM = 0.0
DO 20 J=2,NP
U2 = U(J,2) * (1.0 -U(J,2))
SUM = SUM + A(J) * (U1 + U2)
U1 = U2
20 CONTINUE
THT(NX)= X(NX)/SQRX * SUM
GOTO 150

CALCULATE B. L. PARAMETERS FOR SEMI-TRANSF COORDINATES
100 CONTINUE
SQXC = SQRT(X(NX))
SQRL = SQRT(RL)
CF(NX) = 2.0 * V(1,2) * B(1,2)/(SQXC*SQRL*W(NP,2)**2)
VW(NX) = V(1,2) / SQXC
UE(NX) = U(NP,2)
RX = RL * UE(NX) * X(NX)
DLS(NX)= (ETA(NP)-F(NP,2)/U(NP,2))/SQRL*SQXC
SUM = 0.0
U1 = U(1,2)/U(NP,2)*(1.0 -U(1,2)/U(NP,2))
DO 120 J=2,NP
U2 = U(J,2)/U(NP,2)*(1.0 -U(J,2)/U(NP,2))
SUM = SUM + A(J) * (U1 + U2)
U1 = U2
120 CONTINUE
THT(NX)= SUM / SQRL * SQXC
D(NX) = (U(NP,2)*ETA(NP)-F(NP,2)) * SQXC
150 IF (NX .GE. NXT) GO TO 160
IF (Iedy.EQ. 0 .OR. NX .LE. NTR+2) GO TO 160

MODIFY ALFA USING SIMPSON'S ARGUMENTS
CALL SMPSON
DO 175 J=1,NPT
F(J,1) = F(J,2)
U(J,1) = U(J,2)
V(J,1) = V(J,2)
W(J,1) = W(J,2)
B(J,1) = B(J,2)
CONTINUE
IF (((IP.LE.0).AND.((IP.NE.-2).OR.(ICYCLE.LT.ICYTL)))) RETURN
PRINT OUT VELOCITY PROFILES
IF (NX.EQ.1) GOTO 210
IF (NX.LE.NS) THEN
   FAC1 = SQRT(X(NX)/RL/UE(NX))
   FAC2 = 1.0
ELSE
   FAC1 = SQRT(X(NX)/RL)
   FAC2 = 1.0/UE(NX)
ENDIF
NPM1 = NP -1
WRITE(6,4001) NX,X(NX)
WRITE(6,4000)
WRITE(6,4100) (J, ETA(J), F(J,2), U(J,2), V(J,2), W(J,2), B(J,2),
+       ETA(J)*FAC1, U(J,2)*FAC2, J=1,NPM1,3)
WRITE(6,4100) NP, ETA(NP), F(NP,2), U(NP,2), V(NP,2), W(NP,2), B(NP,2),
+       ETA(NP)*FAC1, U(NP,2)*FAC2
IF (IP.NE.-2) RETURN
IF (((NXVP(ICC,ISF).NE.NX).OR.(ICC.GT.NVP(ISF))) RETURN
WRITE(12,4200) NP
WRITE(12,4300) (ETA(J), J=1,NP)
WRITE(12,4300) (U(J,2), J=1,NP)
ICC = ICC+1
RETURN
FORMAT(/1HO,'NX =',15, 'S/C =',F10.5)
FORMAT(1H0,2H J,9X,3HETA,15X,1HF,13X,1HU,13X,1HV,13X,1HW,13X,1HB,
+       13X,3HY/C,10X,4HU/UE)
FORMAT(1H,13,E14.5,2X,5E14.5,2X,2E14.5)
FORMAT(15)
FORMAT(8F10.6)
END
DATA SET KCBCSMFIT AT LEVEL 001 AS OF 08/24/84
DATA SET KCBCSMFIT AT LEVEL 001 AS OF 08/24/84
DATA SET KCBCSMFIT AT LEVEL 001 AS OF 08/15/83
SUBROUTINE SMFIT(NS,ND,X,Q,D,KS)
THIS SUBROUTINE SMOOthes DATA, Q, USING FIVE-POINT FORMULA.
NS : BEGINNING POINT? ND : END POINT
X : INDEPENDENT COOrdinate? D : WORKING STORAGE
Q : VARIABLE TO BE SMOOthed
KS : NO OF SMOOTHING
DIMENSION X(101),Q(101),D(101)
SMT5(Q1,Q2,Q3,Q4,Q5 ) = 0.0625*(10.0*Q3+4.0*(Q2+Q4)-Q1-Q5)
SMT3(Q1,Q2,Q3,X1,X2,X3) = 0.5*(Q2+(Q1*ABS(X3-X2)+Q3*ABS(X2-X1)))
DATA AT LEVEL 001 AS OF 08/24/84
DATA AT LEVEL 005 AS OF 02/21/84

SUBROUTINE SOLV3

COMMON /BLC0/ NX,NXT,NP,NPT,NTR,IT,INVRS,NS,IP
COMMON /BLC1/ F(101,2),U(101,2),V(101,2),W(101,2),B(101,2)
COMMON /BLC2/ DELF(101),DELU(101),DELV(101),DELW(101)
COMMON /BLC6/ S1(101),S2(101),S3(101),S4(101),S5(101),S6(101),
            S7(101),S8(101),R1(101),R2(101),R3(101),R4(101)
COMMON /GRD/ ETA(101),DETA(101),A(101)
COMMON /BLCB/ A11(101),A12(101),A13(101),A14(101),
            A21(101),A22(101),A23(101),A24(101)

C
IF(KS.LE.0) RETURN
NSP1 = NS+1
NSP2 = NS+2
NDM1 = ND-1
NDM2 = ND-2

C
NDIF = ND-NS+1
IF( NDIF .LT. 3 ) RETURN
IF( NDIF .LT. 5 ) GO TO 200

C
DO 100 K=1,KS
   D(NS+1)= SMT3(Q(NS),Q(NS+1),Q(NS+2),X(NS),X(NS+1),X(NS+2))
   D(ND-1)= SMT3(Q(ND-2),Q(ND-1),Q(ND),X(ND-2),X(ND-1),X(ND))
   DO 20 I=NSP2,NDM2
      D(I) = SMT5(Q(I-2),Q(I-1),Q(I),Q(I+1),Q(I+2))
   CONTINUE
   DO 40 I=NS-1,NDM1
      Q(I) = D(I)
   CONTINUE
RETURN

C
DO 200 K = 1,KS
   DO 220 I = NSP1,NDM1
      D(I) = SMT3(Q(I-1),Q(I),Q(I+1),X(I-1),X(I),X(I+1))
   CONTINUE
   DO 250 I = NSP1,NDM1
      Q(I) = D(I)
   CONTINUE
RETURN

C
DATA SET KCBCSOLV3 AT LEVEL 001 AS OF 08/24/84
DATA SET KCBCSOLV3 AT LEVEL 001 AS OF 08/24/84
DATA SET KCBCSOLV3 AT LEVEL 005 AS OF 02/21/84
G21 = S4(J)  
G23 = -S2(J)/A(J)  
G22 = G23 + S6(J)  

**C** FORWARD SWEEP  
**C**  
DO 500 J = 2, NP  
IF(J .EQ. 2) GO TO 100  
DEN = (A13(J-1) * A21(J-1) - A23(J-1) * A11(J-1))  
  + (A12(J-1) * A21(J-1) - A22(J-1) * A11(J-1))  
DEN1 = A22(J-1) * A(J) - A23(J-1)  
G11 = (A(J) * A13(J-1) + G11 * (A12(J-1) * A(J) - A13(J-1))) / DEN  
G12 = -(A(J) * A(J) + G11 - K12(J-1) * A(J) - A13(J-1)) / DEN1  
G13 = (G11 * A13(J-1) + G12 * A23(J-1)) / A(J)  
G21 = (S2(J) * A21(J-1) - S4(J) * A23(J-1) + A(J) * A22(J-1)) / DEN  
G22 = (-S2(J) + S6(J) * A(J) - G21 * (A(J) * A12(J-1) - A13(J-1))) / DEN1  
G23 = G21 * A12(J-1) + G22 * A22(J-1) - S6(J)  

100  
A11(J) = 1.0  
A12(J) = -A(J) * G13  
A13(J) = A(J) * G13  
A21(J) = S3(J)  
A22(J) = S5(J) - G23  
A23(J) = S1(J) + A(J) * G23  
R1(J) = R1(J) - G11 * R1(J-1) + G12 * R2(J-1) + G13 * R3(J-1)  
R2(J) = R2(J) - G21 * R1(J-1) + G22 * R2(J-1) + G23 * R3(J-1)  
CONTINUE  
500  

**C** BACKWARD SWEEP  
**C**  
DELU(NP) = R3(NP)  
E1 = R1(NP) - A12(NP) * DELU(NP)  
E2 = R2(NP) - A22(NP) * DELU(NP)  
DELV(NP) = (E2 * A11(NP) - E1 * A21(NP)) / (A23(NP) * A11(NP) - A13(NP))  
  + A21(NP))  
DELF(NP) = (E1 - A13(NP) * DELV(NP)) / A11(NP)  
J = NP  
600  
J = J - 1  
E3 = R3(J) - DELU(J+1) + A(J+1) * DELV(J+1)  
DEN2 = A21(J) * A12(J) * A(J) - A21(J) * A13(J) + A(J+1) * A22(J) * A11(J)  
  + A23(J) * A11(J)  
DELV(J) = (A11(J) * (R2(J) + E3 * A22(J)) - A21(J) * R1(J) + E3 * A21(J) * A12(J)) / DEN2  
  + DELF(J)  
DELU(J) = -A(J+1) * DELV(J) + E3  
DELF(J) = (R1(J) - A12(J) * DELU(J) - A13(J) * DELV(J)) / A11(J)  
IF(J .GT. 1) GO TO 600  
CONTINUE  
500  

**C**  
DO 700 J = 1, NP  
F(J, 2) = F(J, 2) + DELF(J)  
U(J, 2) = U(J, 2) + DELU(J)  
V(J, 2) = V(J, 2) + DELV(J)  

700 CONTINUE  
U(1, 2) = 0.0  
CALL EDGCHK(NP, ETA, F(1, 2), U(1, 2), V(1, 2))  
RETURN  

**C**
DATA SET KCBCS0LV4 AT LEVEL 001 AS OF 08/24/84
DATA SET KCBCS0LV4 AT LEVEL 001 AS OF 08/24/84
DATA SET KCBCS0LV4 AT LEVEL 001 AS OF 02/21/84

SUBROUTINE SOLV4(GAMMA1,GAMMA2)

COMMON /BLCO/ NX,NXT,NP,NPT,NTR,IT,INVRS,NS,IP
COMMON /BLC1/ F(101,2),U(101,2),V(101,2),W(101,2),B(101,2)
COMMON /BLC2/ DELF(101),DELV(101),DELW(101)
COMMON /BLC6/ S1(101),S2(101),S3(101),S4(101),S5(101),S6(101),
               S7(101),S8(101),R1(101),R2(101),R3(101),R4(101)
COMMON /GRD/ ETA(101),DETA(101),A(101)
COMMON /BLCB/ All(101),A12(101),A13(101),A14(101),
               A21(101),A22(101),A23(101),A24(101)

All(1) = 1.0
A12(1) = 0.0
A13(1) = 0.0
A14(1) = 0.0
A21(1) = 0.0
A22(1) = 1.0
A23(1) = 0.0
A24(1) = 0.0

DO 10 J = 2,NP
AA1 = A13(J-1)-A(J)*A12(J-1)
AA2 = A23(J-1)-A(J)*A22(J-1)
AA3 = S2(J)-A(J)*S6(J)
DET = AA2*A11(J-1)-AA1*A21(J-1)
AJS = A(J)**2
G11 = -(AA2+A21(J-1)*AJS)/DET
G12 = (A11(J-1)*AJS+AA1)/DET
G13 = A12(J-1)*G11+A22(J-1)*G12+A(J)
G14 = A14(J-1)*G11+A24(J-1)*G12
G21 = (S4(J)*AA2-A21(J-1)*AA3)/DET
G22 = (A11(J-1)*AA3-S4(J)*AA1)/DET
G23 = A12(J-1)*G21+A22(J-1)*G22-S6(J)
G24 = A14(J-1)*G21+A24(J-1)*G22-S8(J)

A11(J) = G11
A12(J) = -(G12+G13)
A13(J) = A(J)*G13
A14(J) = -G14
A21(J) = S3(J)
A22(J) = S5(J)-G23
A23(J) = S1(J)+A(J)*G23
A24(J) = S7(J)-G24
R1(J) = R1(J) - G11*R1(J-1)-G12*R2(J-1)-R3(J-1)*G13
     + -G14*R4(J-1)
R2(J) = R2(J) - G21*R1(J-1)-G22*R2(J-1)-R3(J-1)*G23
     + -G24*R4(J-1)

10 CONTINUE

BACKWARD SWEEP
 J = NP
G1 = GAMMA1/GAMMA2
R3(J) = R3(J)/GAMMA2
R1(J) = R1(J) - A12(J) * (R4(J) + R3(J)) - A14(J) * R3(J)
R2(J) = R2(J) - A22(J) * (R4(J) + R3(J)) - A24(J) * R3(J)
C1 = A11(J) - G1 * (A12(J) + A14(J))
C2 = A21(J) - G1 * (A22(J) + A24(J))
DET = C1 * A23(J) - C2 * A13(J)
DELF(J) = (R1(J) * A23(J) - R2(J) * A13(J)) / DET
DELV(J) = (C1 * R2(J) - C2 * R1(J)) / DET
DELU(J) = R3(J) - G1 * DELF(J)
DELV(J) = R4(J) + DELW(J)

20
J = J - 1
CC1 = DELU(J+1) - R3(J) - A(J+1) * DELV(J+1)
CC2 = DELW(J+1) - R4(J)
CC3 = A13(J) - A(J+1) * A12(J)
CC4 = R1(J) - A12(J) * CC1 - A14(J) * CC2
CC5 = R2(J) - A22(J) * CC1 - A24(J) * CC2
DENO = A11(J) * CC5 - A21(J) * CC3
DELF(J) = (CC4 * CC5 - CC3 * CC6) / DENO
DELV(J) = (All(J) * CC6 - A21(J) * CC4) / DENO
DELU(J) = CC1 - A(J+1) * DELV(J)

IF(J .GE. DO) GO TO 20
DO 30 J = 1, NP
F(J, 2) = F(J, 2) + DELF(J)
U(J, 2) = U(J, 2) + DELU(J)
V(J, 2) = V(J, 2) + DELV(J)
W(J, 2) = W(J, 2) + DELW(J)
30 CONTINUE
U(1, 2) = 0.0
CALL EDGCHK(NP, ETA, F(1, 2), U(1, 2), V(1, 2))
RETURN
END

DATA SET KCBCTRNS AT LEVEL 001 AS OF 08/24/84
DATA SET KCBCTRNS AT LEVEL 001 AS OF 08/24/84
DATA SET KCBCTRNS AT LEVEL 005 AS OF 03/13/84

SUBROUTINE TRNS(ICODE)

COMMON /BLCO/ NX, NXT, NP, NPT, NTR, IT, INVRS, NS, IP
COMMON /BLG1/ F(101, 2), U(101, 2), V(101, 2), W(101, 2), B(101, 2)
COMMON /BLIN/ TITLE(20), XC(100), YC(100), ISG(100), DELS(100),
+ XCTR, XTR, ISTRP, ICYCLE, ICYTL, XCTRS(2), TRFIND(2)
COMMON /EDDY1/ RL, RX, SQRX, RXNTR, GMTR, GMTRS(100)
+ ALFAS(100), FFS(100), RTS(100), TEDY, NXSPT
COMMON /GRD/ ETA(101), DETA(101), A(101)
COMMON /GTY/ X(101), UE(100), P1(100), P2(100), CEL, CELH
COMMON/TRN/ PGAMTR, OMEGA, RTHETB, TRTRAN

100 FORMAT(/3X, 'BEGIN OF TRANSITION HAS BEEN DETECTED BY MICHEL'S',
110 FORMAT(/3X, 'BEGIN OF TRANSITION IS ASSUMED AT THE POINT OF',
ICODE = 0
INT25610 INT25620 INT25630 INT25640 INT25650 INT25660 INT25670 INT25680 INT25690
INT25710 INT25720 INT25730 INT25740 INT25750 INT25760
INT25770 INT25780 INT25790 INT25800 INT25810
INT25820 INT25830 INT25840 INT25850 INT25860
INT25870 INT25880 INT25890 INT25900 INT25910
INT25920 INT25930 INT25940 INT25950 INT25960
INT25970 INT25980 INT25990 INT26000 INT26010
INT26020 INT26030 INT26040 INT26050 INT26060
INT26070 INT26080 INT26090 INT26100 INT26110
INT26120 INT26130 INT26140 INT26150 INT26160
ISEP = 1

C
_IF(V(1,2).LT.0.0) THEN
   **TRANSITION PROCESS HAS BEGUN DUE TO LAMINAR SEPARATION**
   FAC = V(1,1)/(V(1,1)-V(1,2))
   GOTO 20
_END IF
C
C **CHECK MICHEL'S TRANSITION CRITERION**
ISEP = 0
SUM = 0.0
F1 = U(1,2)/U(NP,2)*(1.0-U(1,2)/U(NP,2))
DO 10 J=2,NP
   F2 = U(J,2)/U(NP,2)*(1.0-U(J,2)/U(NP,2))
   SUM = SUM + (F1 + F2)*A(J)
F1 = F2
10 CONTINUE
CONV = SQRT(RL/X(NX))
IF(NX.LE.NS) CONV = SQRT(RX)/X(NX)
THETA = SUM / CONV
RTHETA = RL*UE(NX)*THETA
RTRAN = 1.174*(1.0+22400.0/RX)*RX**0.46
IF(RTHETA.LT.RTRAN) THEN
   RTHETB = RTHETA
   RTRANB = RTRAN
   RETURN
END IF
C
C **TRANSITION PROCESS HAS BEGUN BECAUSE OF MICHEL'S CRITERION**
   FAC = (RTHETB-RTRANB)/(RTRAN-RTRANB-RTHETA+RTHETB)
C
C **COMPUTE EXACT LOCATION OF TRANSITION BEGIN**
20 NTR = NX-1
   NTR1 = NTR + 1
   XCTR = XC(NX-1) + FAC*(XC(NX)-XC(NX-1))
   XTR = X(NX-1) + FAC*(X(NX)-X(NX-1))
   UETR = UE(NX-1) + FAC*(UE(NX)-UE(NX-1))
   IF ( ISEP.EQ.0 ) WRITE (6,100) XCTR,XTR,NTR
   IF ( ISEP.EQ.1 ) WRITE (6,110) XCTR,XTR,NTR
   ICODE = 1
C
C **CALCULATE INTERMITTENCY DISTRIBUTION**
   RXNTR = XTR*UETR*RL
   GGFT = RL**2/PGAMTR/RXNTR**1.34*UETR**3
   DO 30 I=NTR1,NXT
      ALFAS(I) = 0.0168
      GMTRS(I) = 1.0
   30 CONTINUE
   ALFAS(NTR) = 0.0168
   UEINTG = 0.0
   U1 = 0.5/UETR
   X1 = XTR
   DO 40 I=NTR1,NXT
      U2 = 0.5/UE(I)
      X2 = X(I)
      UEINTG = UEINTG+(U1+U2)*(X2-X1)
INT26170
INT26180
INT26190
INT26200
INT26210
INT26220
INT26230
INT26240
INT26250
INT26260
INT26270
INT26280
INT26290
INT26300
INT26310
INT26320
INT26330
INT26340
INT26350
INT26360
INT26370
INT26380
INT26390
INT26400
INT26410
INT26420
INT26430
INT26440
INT26450
INT26460
INT26470
INT26480
INT26490
INT26500
INT26510
INT26520
INT26530
INT26540
INT26550
INT26560
INT26570
INT26580
INT26590
INT26600
INT26610
INT26620
INT26630
INT26640
INT26650
INT26660
INT26670
INT26680
INT26690
INT26700
INT26710
INT26720

111
U1 = U2
X1 = X2
GG = GGFT*UEINTG*(X(I)-XTR)
IF(GG.GT.10.0) GOTO 50
GMTRS(I) = 1.0-EXP(-GG)
40 CONTINUE
C
C *** RESET FINITE DIFFERENCE CALCULATIONS ***
50 DO 60 J=1,NPT
F(J,2) = F(J,1)
U(J,2) = U(J,1)
V(J,2) = V(J,1)
B(J,2) = B(J,1)
W(J,2) = W(J,1)
60 CONTINUE
RETURN
END
C
C SUBROUTINE EDGCHK(NP, ETA, F, U, V)
C
DIMENSION ETA(101), F(101), U(101), V(101)
C
JS = NP - 3
NPM1 = NP - 1
DO 10 J=JS, NPM1
JJ = J
IF(U(J).GE.U(NP).OR.V(J).LT.0.0) GOTO 20
10 CONTINUE
RETURN
JS = JJ - 1
IF(JS.GT.(NP-2)) JS = NP-2
CALL AMEAN(JS, NP, ETA, U, 1)
CALL AMEAN(JS, NP, ETA, F, 1)
DETAP = ETA(JS) -ETA(JS-1)
VJP = (U(JS)-U(JS-1))/DETAP
DO 30 J=JS,NPM1
DETAM = ETA(J+1)-ETA(J)
VJM = (U(J+1)-U(J))/DETAM
V(J) = (VJM*DETAP + VJP*DETAM)/(DETAP+DETAM)
VJP = VJM
DETAP = DETAM
30 CONTINUE
V(NP) = -V(NP-1) + 2.0*VJP
RETURN
C
C >>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
C NOTES: (FOR CHANGING FROM THE ORIGINAL PROGRAM)
1. 'EDDY' HAS BEEN MODIFIED BY ADDING 'FINT'.
2. SUBROUTINE 'EDGCHK' HAS BEEN ADDED.
3. GROWTH LIMIT HAS BEEN ADDED FOR 2 IN 'MAIN2'.
C
C >>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
END
SUBROUTINE SIMPSON

COMMON /BLCO/ NX,NXT,NP,NPT,NTR,IT,INVRS,NS,IP
COMMON /BLG1/ F(101,2),U(101,2),V(101,2),W(101,2),B(101,2)
COMMON /BLG7/ C(100,100),D(100),DB(100),DBP(100),UE0(100),GI
COMMON/EDDY1/ RL,EX,SRX,SRX1,GMTR,GMTS(100)
+ ,ALFAS(100),FFS(100),IESY,NXSPT
COMMON /GTY / X(101),UE(100),P1(100),P2(100),CEL,CELH
COMMON /GRD / ETA(101),DETA(101),A(101)
DIMENSION CRD(100),D(100),DB(100),DBP(100)
+ ,UE0(100),GI
RL2 = SQRT(RU)
RR = DUDX/VNXM/RL2

C
C ------------------------------
C
C STEP 1: CALCULATE (DU/DX)/(DU/DY)
C IF(NX.LT.NXSPT) GOTO 10
C
C IN THE SEPARATED REGION, ALFA SET TO BE CONSTANT
C ALFAS(NX)= ALFAS
C RETURN
C(-----
C
C 10 CONTINUE
C IF(V(1,2).GT. 0.0) GOTO 20
C
C SEPARATION OCCURS. ALFA SET TO BE THE PREVIOUS ITERATED VALUE
C ALFAS = ALFAS(NX)
C NXSPT = NX
C RETURN
C(-----
C
C MODIFY OUTER EDDY BASED ON SIMPSON SUGGESTION
TM = 0.0
JM = 1
DO 30 J=2,NP
TS = (B(J,2)-1.0)* V(J,2)
IF(TS.LT.TM) GOTO 30
TM = TS
JM = J
30 CONTINUE
VNXM = 0.5*(V(JM,2)+V(JM,1))
IF (NX.LE. NS) GOTO 35
DUDX = (U(JM,2)-U(JM,1)) / (X(NX)-X(NX-1))
GO TO 38
35 DUDX = CEL*(U(JM,2)-U(JM,1))+P2(NX)*U(JM,2)+0.5*ETA(JM)*
+ VNXM*(P2(NX)-1.0)
38 RU = RL
IF(NX.LE.NS) RU = RL * UE0(NX) * X(NX)
RL2 = SQRT(RU)
RR = DUDX/VNXM/RL2

C
C STEP 2: CALCULATE (UU - VV)/UV
VNXM = 0.5*(V(1,2)+V(1,1))
RT = VNXM/TM
C PRINT'(3X,2I5,3F10.3)',NX,JM,VNXM,TM,RT
IF (RT .LT. 0.0) RT = 0.0
IF(RT.GT.1.0) GOTO 60
CR = 6.0 /(1.0 + 2.0 * RT*(2.0 -RT))
C CR = 2.0
C DO 40 I=2,12
C IF(RT.LT.RTD(I)) GOTO 50
C 40 CONTINUE
C GOTO 70
C 50 CR = CRD(I-1)+(CRD(I)-CRD(I-1))*(RT-RTD(I-1))/(RTD(I)-RTD(I-1))
GOTO 70
C 60 CR = (1.0 + RT ) /RT
C C STEP 3 : CALCULATE FF
70 FR = CR * RR
IF(FR .GT.0.35) FR = 0.35
IF (FR .LT. -0.8) FR = -0.8
FFS(NX)=(FFS(NX) + (1.0 -FR))/ 2.0
RTS(NX)= RT
ALFAS(NX)= 0.0168/FFS(NX)**2.5
RETURN
C C------
END
C SUBROUTINE XSPACE(NI,NRITE,XII,XLLT,RAD,NL1,NR1)
DIMENSION XII(200),T(200)
DATA PI/3.14159265358979/
RAD = PI
NLEFT=NI-1-NRITE
NR4=NRITE/2
IF((NRITE/2*2).NE. NRITE) NR4=(NRITE+1)/2
NL1=NR4+1
NL2=NR4+NLEFT
NR1=NL2+1
NR2=NI
PI2=0.5*PI
RAD2=(PI-RAD)/2.0+PI2
RAD3=RAD2+RAD
SRT =RAD2/FLOAT(NR4)
SRT2=SRT
IF((NRITE/2*2).NE. NRITE) SRT2=RAD2/FLOAT(NR4-1)
SLT = RAD/FLOAT(NLEFT)
DO 10 I=1,NR4
10 XII(I)=0.5*(1.0+COS(FLOAT(I-1)*SRT))
DO 20 I=NL1,NL2
20 XII(I)=0.5+XLLT*COS(FLOAT(I-NL1)*SLT+RAD2)
DO 30 I=NR1,NR2
30 XII(I)=0.5*(1.0+COS(FLOAT(I-NR1)*SRT2+RAD3))
NA=(NI+1)/2
IF((NI/2*2) .EQ. NI) NA=NI/2+1
FN1=FLOAT(NA-1)
FN2=FN1
IF((NI/2*2) .EQ. NI) FN2=FLOAT(NA-2)
DO 40 I=1,NA
40 T(I)=FLOAT(NA-I)/FN1
CALL AMEAN(1, NA, T, XII, 1)
XDIF = XII(1) - XII(2)
IF(XDIF .LT. 0.004) THEN
  DO 45 I = 2, 5
    XII(I) = XII(I-1) - XDIF*3.0
  CONTINUE
CALL AMEAN(2, NA, T, XII, 10)
END IF
DO 50 I = NA, NI
  XI(I) = XII(I-1) - XDIF*3.0
  CONTINUE
CALL AMEAN(2, NA, T, XII, 10)
END

DO 45 I = 2, 5
  XI(I) = XII(I-1) - XDIF*3.0
CONTINUE

SUBROUTINE TRGRID (NL, XO, YO, NI, NR, XLLT, N10, RAD, ID, NXSS)

  THIS SUB. IS TO REGRID SPACING NEAR TRAILING-EDGE

  DIMENSION XO(200), YO(200), XI(200), YI(200), D1(200), D2(200), D3(200),
  + XOO(200), YOO(200), XII(200), YII(200), WX(200), WY(200),
  + WX1(200), WY1(200), T(200)

  N20 = N10
  IF((NL/2*2) .EQ. 0) N10 = N10 + 1
  IF((NI-(N1/2)*2) .NE. 0) N1I = (NI-1)/2+1
  IF((NI-(N1/2)*2) .EQ. 0) N1I = NI/2
  N2I = N1I
  IF((NI/2*2) .EQ. 0) N2I = N1I + 1

  CALL XSPACE(NI, NR, XI, XLLT, RAD, NL1, NR1)
  PRINT *, 'NR= ', NR, 'XLLT= ', XLLT

  WRITE (6, 290)
  WRITE (6, 300) (XO(I), I=1, NL)
  WRITE (6, 298)
  WRITE (6, 300) (YO(I), I=1, NL)

  IF(ID .EQ. 2) THEN
    DO 60 I = NL1, NXSS
      YI(I) = YO(I)
    CONTINUE
    XI(I) = XI(I)
    NXST = NXSS + 8
    XM1 = (XI(NXST) - XI(NXSS))/8.0
    XM2 = (XI(NR1) - XI(NXSS))/(NR1 - NXSS)
    DO 62 I = NXSS, NR1-1
      XI(I) = (XI(I) + XM1)/2.0
    CONTINUE
    TX(I) = (XI(I) + XM2)/2.0
    DO 65 I = NXSS, NXST
      T(I) = FLOAT(I - NXSS)*XM1 + XI(NXSS)
    CALL AMEAN(NXSS, NXST, T, XI, 8)
    DO 68 I = NXSS, NR1
      T(I) = FLOAT(I - NXSS)*XM2 + XI(NXSS)
      CALL AMEAN(NXSS, NR1, T, XI, 12)
    CONTINUE
  ELSE
    NXSS = N2I
    END IF

  FOR LOWER SURFACE
    DO 5 I = 1, N10

II = N10  - I + 1
WX(II) = XO(I)
WY(II) = YO(I)

CONTINUE
DO 7 I = 1 , N1I
II = N1I  - I + 1
WX(I) = XI(I)

CONTINUE
CALL DIFF3(N10,WX,WY,D1,D2,D3,0)
CALL INTRP3(N10,WX,WY,D1,D2,D3,N1I,WXI,WYI)

CONTINUE
DO 9 I = 1 , N1I
II = N1I  - I + 1
YI(II) = WYI(I)

CONTINUE
C FOR UPPER SURFACE
DO 10 I = 1 , N2O
II = N1  - N2O + I
XOO(I) = XO(II)
YOO(I) = YO(II)

CONTINUE
DO 20 I = 1 , N2I
II = N1  - N2I + I
WX(I) = XI(II)

CONTINUE
CALL DIFF3(N2O,XOO,YOO,D1,D2,D3,0)
CALL INTRP3(N20,XOO,YOO,D1,D2,D3,N2I,WXI,WYI)

CONTINUE
DO 25 I = 1 , N2I
II = N2I  - I + 1
XII(II) = WXI(I)
YII(II) = WYI(I)

CONTINUE
C COMBINE TWO SURFACES INTO ONE CIRCLE
C NN = N1  - N1O  - N2O

DO 30 I = 1 , NN

II = N1I + I

III = N10 + I

XII(II) = XO(III)

YII(II) = YO(III)

CONTINUE
DO 40 I = 1 , N2I
I1 = NXSS + I -1
I2 = N2I  - I + 1
XII(I1) = XII(I2)
YII(I1) = YII(I2)

CONTINUE
XI(I1) = XO(I)
XI(I1) = XO(N1)
YI(I1) = YO(I)
YI(I1) = YO(N1)

N1 = I1
DO 50 I = 1 , N1
XO(I) = XI(I)
YO(I) = YI(I)
50 CONTINUE
C
C WRITE (6, 295)
C WRITE (6, 300) (XO(I), I=1,N1)
C WRITE (6, 298)
C WRITE (6, 300) (YO(I), I=1,N1)
C
RETURN

100 FORMAT(7I5)
200 FORMAT(6F10.0)
290 FORMAT(//'ORIGINAL COORDINATES',/,'X/C')
295 FORMAT(//'INTERPLATED COORDINATES',/,'X/C')
298 FORMAT(6F10.6)
C
SUBROUTINE STAGR(N,STAG,XO,YO,XSTGR,YSTGR)
DIMENSION XO(100),YO(100),XSTGR(100),YSTGR(100),DS(100)
C
XOTE = 0.5*(XO(1)+XO(N))
YOTE = 0.5*(YO(1)+YO(N))
DS(1) = SQRT((XO(1)-XOTE)**2 + (YO(1)-YOTE)**2)
DSM = DS(1)
DO 10 I = 2, N
DS(I) = SQRT((XO(I)-XOTE)**2 + (YO(I)-YOTE)**2)
IF (DS(I) .LT. DSM) GOTO 10
IM = I
DSM = DS(I)
CONTINUE
C
YYY = YOTE-YO(IM)
XXX = XOTE-XO(IM)
IF (YYY .EQ. 0.0 .AND. XXX .EQ. 0.0) THEN
ANG = 0.0
ELSE
ANG = ATAN2(YYY,XXX)
END IF
ANG = ANG + STAG
C
COSAN = COS(ANG)
SINAN = SIN(ANG)
DO 20 I = 1, N
YY = YO(I)-YO(IM)
XX = XO(I)-XO(IM)
C
IF (YY .EQ. 0.0 .AND. XX .EQ. 0.0) THEN
ANGCO = 0.0
ELSE
ANGCO = ATAN2(YY,XX)
END IF
XSTGR(I) = XO(I)*COSAN + YO(I)*SINAN
YSTGR(I) = YO(I)*COSAN - XO(I)*SINAN
20 CONTINUE
RETURN
END
APPENDIX B. C4 CASCADE

A. EXPERIMENTAL RESULTS

The experimental results of the C4 cascade were obtained directly from professor G.J. Walker, University of Tasmania, Tasmania, Australia, who performed these experiments.

The results of the boundary layer measurements of the C4 cascade are given below at four inlet angles: 34.1°, 36.3°, 45.6°, and 47.7°. The Reynolds numbers, based on the chord and the upstream velocity, are 200000, 191000, 173000 and 171000 respectively. The results given in the following tables include the displacement thickness (δ*), the shape factor (H) and the local free stream velocity (UE).

Table 1. EXPERIMENTAL RESULTS AT INLET ANGLE OF 34.1°

<table>
<thead>
<tr>
<th>x/c</th>
<th>δ* [10^-3 FT]</th>
<th>H</th>
<th>UE [FT/SEC]</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.4</td>
<td>4.9</td>
<td>2.48</td>
<td>168.37</td>
</tr>
<tr>
<td>0.5</td>
<td>6.28</td>
<td>2.61</td>
<td>167.35</td>
</tr>
<tr>
<td>0.6</td>
<td>8.79</td>
<td>3.24</td>
<td>158.31</td>
</tr>
<tr>
<td>0.7</td>
<td>10.83</td>
<td>3.63</td>
<td>149.27</td>
</tr>
<tr>
<td>0.8</td>
<td>16.63</td>
<td>3.79</td>
<td>147.13</td>
</tr>
<tr>
<td>0.9</td>
<td>16.19</td>
<td>1.89</td>
<td>143.79</td>
</tr>
</tbody>
</table>

Table 2. EXPERIMENTAL RESULTS AT INLET ANGLE OF 36.3°

<table>
<thead>
<tr>
<th>x/c</th>
<th>δ* [10^-4 FT]</th>
<th>H</th>
<th>UE [FT/SEC]</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.4</td>
<td>5.43</td>
<td>2.55</td>
<td>161.63</td>
</tr>
<tr>
<td>0.5</td>
<td>7.09</td>
<td>2.70</td>
<td>157.59</td>
</tr>
<tr>
<td>0.6</td>
<td>10.3</td>
<td>3.34</td>
<td>148.78</td>
</tr>
<tr>
<td>0.7</td>
<td>12.63</td>
<td>3.78</td>
<td>139.87</td>
</tr>
<tr>
<td>0.8</td>
<td>14.84</td>
<td>2.78</td>
<td>135.01</td>
</tr>
<tr>
<td>0.9</td>
<td>16.43</td>
<td>1.76</td>
<td>133.23</td>
</tr>
</tbody>
</table>
### Table 3. EXPERIMENTAL RESULTS AT INLET ANGLE OF 45.6°

<table>
<thead>
<tr>
<th>x/c</th>
<th>$\delta^*$ [10^{-3} FT]</th>
<th>H</th>
<th>UE [FT/SEC]</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.4</td>
<td>8.08</td>
<td>2.58</td>
<td>137.88</td>
</tr>
<tr>
<td>0.5</td>
<td>9.83</td>
<td>2.41</td>
<td>133.70</td>
</tr>
<tr>
<td>0.6</td>
<td>12.35</td>
<td>2.33</td>
<td>122.18</td>
</tr>
<tr>
<td>0.7</td>
<td>12.98</td>
<td>1.97</td>
<td>114.93</td>
</tr>
<tr>
<td>0.8</td>
<td>19.44</td>
<td>1.90</td>
<td>111.77</td>
</tr>
<tr>
<td>0.9</td>
<td>27.69</td>
<td>1.92</td>
<td>109.26</td>
</tr>
</tbody>
</table>

### Table 4. EXPERIMENTAL RESULTS AT INLET ANGLE OF 47.7°

<table>
<thead>
<tr>
<th>x/c</th>
<th>$\delta^*$ [10^{-4} FT]</th>
<th>H</th>
<th>UE [FT/SEC]</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.4</td>
<td>8.87</td>
<td>2.24</td>
<td>130.64</td>
</tr>
<tr>
<td>0.5</td>
<td>10.27</td>
<td>2.19</td>
<td>124.76</td>
</tr>
<tr>
<td>0.6</td>
<td>14.31</td>
<td>2.08</td>
<td>116.86</td>
</tr>
<tr>
<td>0.7</td>
<td>16.45</td>
<td>1.87</td>
<td>106.72</td>
</tr>
<tr>
<td>0.8</td>
<td>24.16</td>
<td>1.82</td>
<td>103.75</td>
</tr>
<tr>
<td>0.9</td>
<td>36.00</td>
<td>2.01</td>
<td>102.18</td>
</tr>
</tbody>
</table>
The results of the measurements of the velocity profiles in the boundary layer at two inlet angles, 34.1° and 36.3° at 50% chord are given below.

<table>
<thead>
<tr>
<th>$\gamma$</th>
<th>$\beta = 36.3^\circ$</th>
<th>$\beta = 34.1^\circ$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>2.3</td>
<td>0.172</td>
<td>0.208</td>
</tr>
<tr>
<td>3.7</td>
<td>0.270</td>
<td>0.327</td>
</tr>
<tr>
<td>6.2</td>
<td>0.469</td>
<td>0.534</td>
</tr>
<tr>
<td>8.6</td>
<td>0.666</td>
<td>0.728</td>
</tr>
<tr>
<td>11.0</td>
<td>0.794</td>
<td>0.867</td>
</tr>
<tr>
<td>13.4</td>
<td>0.891</td>
<td>0.933</td>
</tr>
<tr>
<td>18.3</td>
<td>0.982</td>
<td>0.985</td>
</tr>
<tr>
<td>23.2</td>
<td>1.000</td>
<td>1.000</td>
</tr>
</tbody>
</table>

B. C4 CASCADE COORDINATES

```
DIMENSION X(0:100),XU(0:100),XL(0:100),YU(0:100),YL(0:100)          C4 00010
DATA A1,A2,A3,A4/0.15492,0.06563,0.2528,0.2811/                  C4 00020
DATA B1,B2,B3,B4/0.03866,0.07871,0.1467,0.0348/                 C4 00030
PI = ACOS(-1.0)                                                   C4 00040
C READ (5,800) NMAX                                               C4 00050
800 FORMAT (I5)                                                   C4 00060
NMAX=33                                                          C4 00070
C READ (5,810) (X(I),I=0,NMAX)                                    C4 00080
810 FORMAT (6F10.6)                                               C4 00090
DO 50 I=0,NMAX                                                    C4 00100
X(I) = (1.0-COS(PI*I/NMAX))/2.                                      C4 00110
50 CONTINUE                                                     C4 00120
DO 100 I=0,NMAX                                                  C4 00130
SRT = SQRT((0.5/SIN(PI/12))**2-(0.5-X(I))**2)                   C4 00140
YC = -0.5/TAN(PI/12) + SRT                                        C4 00150
DY = ATAN((0.5-X(I))/SRT)                                        C4 00160
IF (X(I).LT.0.3) THEN                                             C4 00170
  YT = A1*SQRT(X(I)) - A2*X(I) - A3*X(I)**2 + A4*X(I)**3          C4 00180
ELSE                                                             C4 00190
  YT = B1 + B2*X(I) - B3*X(I)**2 + B4*X(I)**3                     C4 00200
END IF                                                           C4 00210
```
\[ YU(I) = YC + \cos(DY) \times YT \]
\[ YL(I) = YC - \cos(DY) \times YT \]
\[ XU(I) = X(I) - \sin(DY) \times YT \]
\[ XL(I) = X(I) + \sin(DY) \times YT \]

CONTINUE

\[
\text{WRITE } (6,900) (I,X(I),XU(I),YU(I),XL(I),YL(I),I=0,NMAX)
\]

\[
\text{WRITE } (1,910) (XL(I),I=NMAX,0,-1),(XU(I),I=1,NMAX)
\]

\[
\text{WRITE } (1,910) (YL(I),I=NMAX,0,-1),(YU(I),I=1,NMAX)
\]

STOP
END
LIST OF REFERENCES


<table>
<thead>
<tr>
<th>No.</th>
<th>Copies</th>
<th>Name and Address</th>
</tr>
</thead>
</table>
| 1.  | 2      | Defense Technical Information Center  
Cameron Station  
Alexandria, VA 22304-6145 |
| 2.  | 2      | Library, Code 0142  
Naval Postgraduate School  
Monterey, CA 93943-5002 |
| 3.  | 1      | Chairman, Department of Aeronautics, Code 67  
Naval Postgraduate School  
Monterey, CA 93943-5000 |
| 4.  | 1      | Professor T. Cebeci  
Staff director  
Research & Technology  
McDonnell Douglas Corporation  
3855 Lakewood Boulevard  
Long Beach, CA 90846 |
| 5.  | 12     | Professor M. F. Platzer  
Department of Aeronautics, Code 67P1  
Naval Postgraduate School  
Monterey, CA 93943-5000 |
| 6.  | 1      | Mr. T. S. Momiyama  
Director of Aircraft Division  
Naval Air System Command, Code 931  
Washington D.C. 20361 |
| 7.  | 1      | Mr. G. Derderian  
Project Manager for Propulsion, Aircraft Division  
Naval Air System Command, Code 931  
Washington D.C. 20361 |
| 8.  | 1      | Professor G. J. Walker  
Department of Civil and Mechanical Engineering  
University of Tasmania  
Hobart, Tasmania, Australia |
| 9.  | 1      | Professor R. P. Shreeve  
Turbopropulsion Laboratory, Code 67Sf  
Naval Postgraduate School  
Monterey, CA 93943-5000 |
10. Professor D. W. Netzer  
   Department of Aeronautics, Code 67Nt  
   Naval Postgraduate School  
   Monterey, CA 93943-5000

11. Dr. Stephen J. Shamroth  
   Scientific Research Association, inc.  
   50 Nye Road, P.O. Box 1058  
   Clastonbury, CO 06033

12. Dr. Robert J. Simoneau  
   Chief of Heat Transfer Branch, Mail Stop 5-11  
   NASA Lewis Research Center  
   21000 Brookpark Road  
   Cleveland, OH 44135

13. Mr. James O'Brien  
   Heat Transfer Branch, Mail Stop 5-11  
   NASA Lewis Research Center  
   21000 Brookpark Road  
   Cleveland, OH 44135

14. Moshe Horev  
   Israel Defense Force  
   I.D.F., P.O. Box 02158  
   Israel

15. Defense and Armed Forces Attache  
   Embassy of Israel  
   3514 International Drive, N.W.  
   Washington, D.C. 20008

16. Isaak Mostov  
   SMC 2752  
   Naval Postgraduate School  
   Monterey, CA 93943-5000

17. Zeev Snir  
   Israel Defense Force  
   I.D.F., P.O. Box 02158  
   Israel
Investigation of incompressible cascade flows using a viscous/inviscid interactive code.