# CONTRACTOR REPORT

8024

SAND86—7182
Unlimited Release
UC—60

RS-8232-2/ 65763
C. 1

# User's Manual for ADAM (Advanced Dynamic Airfoil Model)

8232-2//065763
00000001 -

J. W. Oler, J. H. Strickland, B. J. Im
Department of Mechanical Engineering
Texas Tech University
Lubbock, Tx 79409

Prepared by Sandia National Laboratories Albuquerque, New Mexico 87185 and Livermore, California 94550 for the United States Department of Energy under Contract DE-AC04-76DP00789

Printed June 1987

1078404

Issued by Sandia National Laboratories, operated for the United States Department of Energy by Sandia Corporation.

**NOTICE:** This report was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government nor any agency thereof, nor any of their employees, nor any of their contractors, subcontractors, or their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise, does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government, any agency thereof or any of their contractors or subcontractors. The views and opinions expressed herein do not necessarily state or reflect those of the United States Government, any agency thereof or any of their contractors or subcontractors.

Printed in the United States of America
Available from
National Technical Information Service
U.S. Department of Commerce
5285 Port Royal Road
Springfield, VA 22161

NTIS price codes
Printed copy: A05
Microfiche copy: A01

Distribution
Category UC-60

SAND86-7182
Unlimited Release
Printed June 1987

# User's Manual for ADAM

# (Advanced Dynamic Airfoil Model)

J.W. Oler, J.H. Strickland, and B.J. Im
Department of Mechanical Engineering
Texas Tech University
Lubbock, TX 79409

Sandia Contract No. 52-3727

ABSTRACT

The computer code for an advanced dynamic airfoil model (ADAM) is described. The code is capable of calculating steady or unsteady flow over two-dimensional airfoils with allowances for boundary layer separation. Specific types of airfoil motions currently installed are steady rectilinear motion, impulsively started rectilinear motion, constant rate pitching, sinusoidal pitch oscillations, sinusoidal lateral plunging, and simulated Darrieus turbine motion. Other types of airfoil motion may be analyzed through simple modifications of a single subroutine. The code has a built-in capability to generate the geometric parameters for a cylinder, the NACA four-digit series of airfoils, and a NASA NLF-0416 laminar airfoil. Other types of airfoils are easily incorporated. The code ADAM is currently in a state of development. It is theoretically consistent and complete. However, further work is needed on the numerical implementation of the method. Details of the method and a description of the accuray limitations of the current numerical implementations may be found in A Numerical Model for Unsteady Two-Dimensional Flow Calculations with Flow Separation, SAND86-7183.

TABLE OF CONTENTS

NOMENCLATURE

1. INTRODUCTION ..... 1

2. MODEL DESCRIPTION ..... 2

    2.1 Theoretical Basis ..... 2

    2.2 Analysis Options ..... 7

3. PROGRAM DESCRIPTION ..... 12

    3.1 Input Requirements ..... 12

    3.2 Program Output ..... 15

    3.3 Subroutine Calling Sequence ..... 16

    3.4 Subroutine Functions ..... 17

    3.5 Common Block Variable Definitions ..... 19

APPENDIX - Program Listing ..... 23

## NOMENCLATURE

$A_{ij}$ - Normal influence velocity coefficient due to an airfoil vorticity element.

$A^t_{ik}$ - Normal influence velocity coefficient due to a wake element on the trailing edge wake.

$A^s_{il}$ - Normal influence velocity coefficient due to a wake element on the boundary layer separation surface.

$c$ - Airfoil chord length.

$C_f$ - Skin friction coefficient.

$\vec{n}$ - Normal unit vector at a potential source point.

$\vec{n}_{TE}$ - Normal unit vector on the nascent trailing edge wake element.

NE - Number of elements on the airfoil.

NS - Number of elements on the separated boundary layer wake surface.

NT - Number of elements on the trailing edge wake surface.

$P$ - Pressure.

$P_\infty$ - Freestream pressure.

$R$ - Magnitude of the vector connecting a potential source point to a field point.

$R$, $\tilde{R}$ - Dimensional and nondimensional radius of a Darrieus turbine.

$s$ - Surface coordinate along the airfoil surface.

$S$ - Surface of the airfoil.

$t$, $\tilde{t}$ - Dimensional and dimensionless time.

$\vec{u}$ - Total fluid velocity vector.

$\vec{u}_e$ - Edge velocity on the airfoil surface.

$\vec{u}_r$ - Relative velocity between the airfoil and the freestream.

$u^*_r$ - Relative velocity between a Darrieus turbine blade and the freestream.

$u_\infty$ - Freestream velocity magnitude.

$u_\infty^*$ - Freestream velocity in a Darrieus turbine analysis.

$W$ - Wake surfaces.

$X_a, Y_a, \tilde{X}_a, \tilde{Y}_a$ - Dimensional and dimensionless airfoil coordinates.

$Y_{max}, \tilde{Y}_{max}$ - Dimensional and dimensionless maximum periodic lateral displacement.

$\alpha$ - Angle of attack.

$\gamma$ - Surface vorticity strength.

$\Gamma_b, \Gamma_s, \Gamma_t, \Gamma_{net}$ - Circulation associated with vorticity on the airfoil, on the boundary layer separation wake, on the trailing edge wake, and in the complete flow field.

$\delta^*$ - Displacement thickness.

$\theta$ - Momentum thickness.
- Angle of attack parameter.

$\lambda$ - Darrieus tip-to-windspeed ratio.

$\mu$ - Source strength distribution on the airfoil surface.

$\nu$ - Surface unit normal at a field point.

$\pi$ - 3.14159...

$\rho$ - fluid density.

$\sigma$ - Doublet strength distribution on the airfoil surface.

$\phi$ - Disturbance potential.

$\omega, \tilde{\omega}$ - Dimensional and dimensionless angular rotation rate.

# 1. INTRODUCTION

To reduce the peak power output of a Darrieus turbine without adversely affecting its performance in the low and medium windspeed ranges, it is necessary to tailor the dynamic stall characteristics of its blade sections. Experimental investigations have shown that power regulation may be achieved through the design of blade sections which either passively exhibit the desired dynamic characteristics or have provisions for active boundary layer control. A numerical model provides an economical method for evaluating the potential usefulness of candidate airfoil sections. To fully characterize the performance of an airfoil, the model should be capable of predicting the aerodynamic loads on the airfoil in steady and unsteady motion with possible boundary layer separation.

The aerodynamic model ADAM, whose numerical code is described herein, is theoretically capable of performing these analysis functions. However, in its current state of development, predictions of separated flows cannot be reliably made due to problems associated with identifying the boundary layer separation point. Work in this area is continuing. A complete description of the theoretical basis of ADAM may be found in the final report for Sandia Contract No. 52-3727.

The objective for the current report is to provide a description of the execution and operation of the numerical code. In Chapter 2, a brief description of the theoretical basis of ADAM and the analysis options currently installed in the program are given. Chapter 3 contains descriptions of the input requirements, program output, program structure, subroutine functions, and definitions for the common block variables. A complete listing of the program is given in the appendix with the routines listed in alphabetic sequence. The complete FORTRAN 77 source for ADAM is available on magnetic tape and MS-DOS diskette.

## 2. MODEL DESCRIPTION

### 2.1 Theoretical Basis

The basic theoretical approach involves a coupled viscid/inviscid representation of the flow around an airfoil. Viscous effects are restricted to a thin boundary layer extending from the stagnation point near the leading edge of the airfoil to the trailing edge or possibly, to a boundary layer separation point. The inviscid flow field is modeled as an irrotational flow with the boundaries represented by surfaces of distributed potential singularity. These boundaries include wake surfaces extending from the trailing edge and any boundary layer separation points. Flow calculations are started from a state of uniform motion for steady, unseparated flows or impulsively for unsteady flows and flows with possible boundary layer separation. Solutions are achieved through an incremental time stepping process.

The boundary layer calculations are based upon a momentum integral approach for which the basic equation is

$$\frac{1}{u_e^2}\frac{\partial}{\partial t}(u_e\delta^*) + \frac{\partial\theta}{\partial s} + \frac{1}{u_e}\frac{\partial u_e}{\partial s}(2\theta + \delta^*) = \frac{C_f}{2} \tag{1}$$

where $u_e$, $\delta^*$, $\theta$, and $C_f$ are the edge velocity, momentum thickness, displacement thickness, and skin friction coefficient, respectively. The calculations begin at the stagnation point with initial conditions based upon wedge flow solutions. Laminar flow calculations are continued until transition is triggered by laminar separation and turbulent reattachment or through laminar instabilities. If complete laminar separation does not occur, turbulent boundary layer calculations are continued to the trailing edge or an intermittent turbulent boundary layer separation point.

The various flow regimes involved in the boundary layer calculations are illustrated in Fig.1.

For the irrotational flow calculations, the velocity vector at any point in the flow is given by

$$\vec{u} = \vec{u}_\infty + \nabla\phi \tag{2}$$

where $\vec{u}_\infty$ is the freestream velocity and $\phi$ is the disturbance potential associated with the presence of the airfoil and wake surfaces in the flow. The governing equation for $\phi$ is the well-known Laplace equation for which any solution may be written in the form

$$\phi = \frac{1}{2\pi}\int_{S+W} \sigma \frac{\partial}{\partial n}\left(\frac{1}{R}\right) dS + \frac{1}{2\pi}\int_S \frac{\mu}{R}\, dS \cdot \tag{3}$$

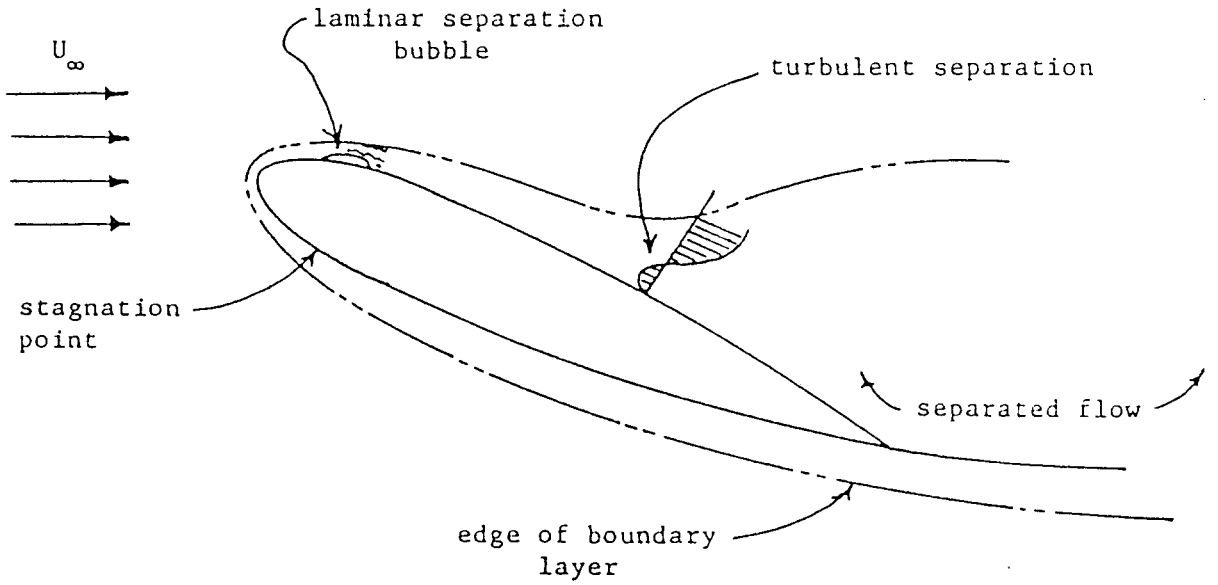


Fig. 1 Boundary Layer Flow Regimes

In Eq. 3, the potential field is given in terms of unknown doublet and source distributions on the airfoil and doublet distributions on the wake surfaces. These unknown distributions are determined through the application of boundary conditions and the development of a time dependent solution in which the wake properties are known at each solution instant.

The kinematic surface tangency condition requires that the normal relative velocity between the fluid and airfoil surface be zero. This boundary condition may be expressed as

$$\frac{\partial \phi}{\partial \nu} + \vec{u}_r \cdot \vec{\nu} = 0 \cdot \tag{4}$$

Applying Eq. 4 to the general expression for the disturbance potential given in Eq. 3 yields

$$\frac{1}{2\pi} \int_{S+W} \sigma \frac{\partial^2}{\partial \nu \partial n}\left(\frac{1}{R}\right) dS = -\vec{u}_r \cdot \vec{\nu} - \frac{1}{2\pi} \int_S \mu \frac{\partial}{\partial \nu}\left(\frac{1}{R}\right) dS \cdot \tag{5}$$

In actual application, the doublet distributions on the airfoil and wake surfaces are replaced by equivalent vorticity distributions where the local vorticity strength is equal to the gradient of the doublet strength.

Eq. 5 is made tractable for numerical solution by utilizing a panel representation of the airfoil and wake surfaces. On the airfoil, the vorticity and source distributions are piecewise linear. With the exception of the nascent wake element at the trailing edge (which has a linear vorticity distribution), the vorticity distributions on the wake surfaces are represented by lumped, discrete vortices. Boundary conditions on the airfoil are satisfied at the element centroids and the wake geometry is determined by tracking the motion of the discrete vortices.

Applying Eq. 5 to the NE element representation of the airfoil (shown in Fig. 2) with NT and NS wake elements on the trailing edge and separated wake surfaces, respectively, produces NE simultaneous equations, i.e.,

$$A_{ij}\gamma_j = -\vec{u}_r \cdot \vec{\nu}_i - A^t_{ik}\gamma^t_k - A^s_{il}\gamma^s_l - B_{ij}\mu_j \tag{6}$$

$$\begin{aligned} \text{for } i &= 1 \text{ to NE} \\ j &= 1 \text{ to NE} \\ k &= 1 \text{ to NT} \\ l &= 1 \text{ to NS.} \end{aligned}$$

To uniquely define the circulation around the airfoil, a Kutta condition must be applied. For steady flow, this requires that the wake vorticity be removed to infinity and that the net vorticity at the trailing edge be equal to zero, i.e.,

$$\gamma_1 = \gamma_{NE} \quad \text{and} \quad A^s_{ik}\gamma^s_k = A^t_{il}\gamma^t_l = 0 \cdot \tag{7}$$

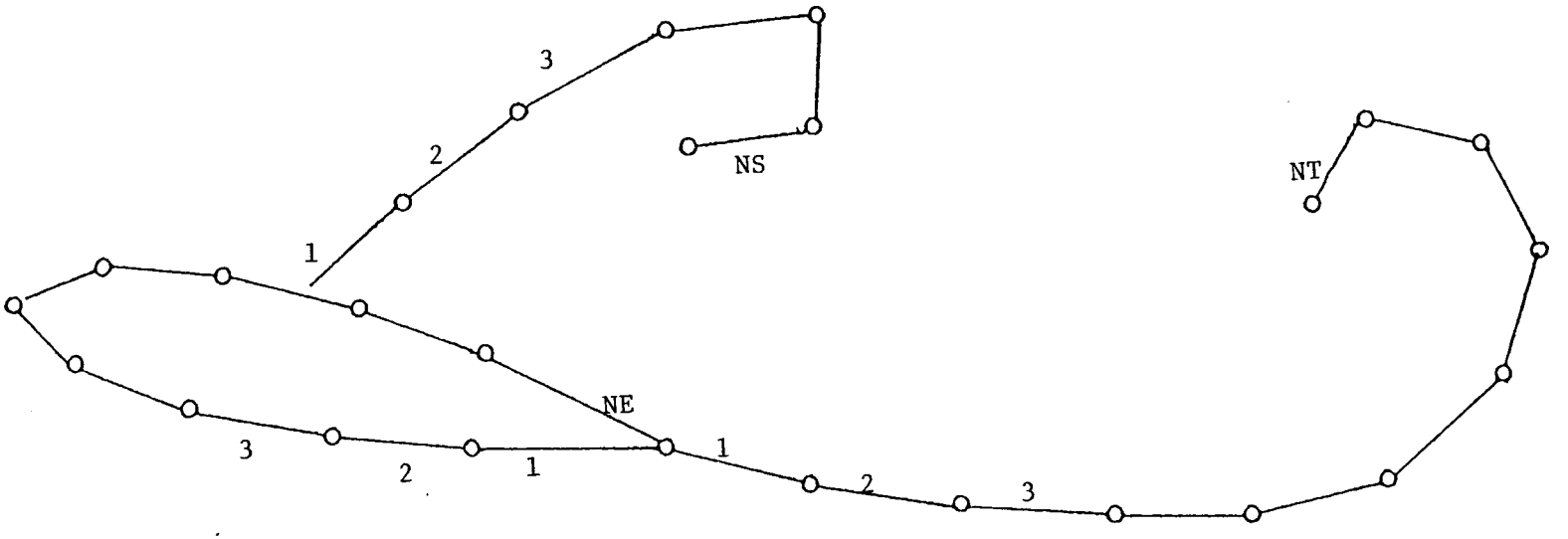


Fig. 2 Panel Representation of Airfoil and Wake System

For unsteady flows, the flow is required to come off of the trailing edge parallel to the trailing edge bisector. By establishing a control point on the nascent wake element, this condition may be enforced by requiring

$$\vec{u} \cdot \vec{n}_{TE} = 0 \cdot \tag{8}$$

If boundary layer separation occurs, vorticity will be injected into the wake surface emanating from the separation point at the rate

$$\frac{d\Gamma_s}{dt} = - \frac{u_e^2}{2} \tag{9}$$

where $u_e$ is the edge velocity at the separation point. The rate of vorticity shedding at the trailing edge is determined by applying the Kelvin-Helmholtz vorticity conservation theorem, i.e.,

$$\frac{d\Gamma_{net}}{dt} = \frac{d\Gamma_b}{dt} + \frac{d\Gamma_s}{dt} + \frac{d\Gamma_t}{dt} = 0 \tag{10}$$

where $\Gamma_b$, $\Gamma_s$, and $\Gamma_t$ are the vorticity on the airfoil, separation point wake and trailing edge wake, respectively.

A final boundary condition is chosen such that the local airfoil element source strength is equal to the negative of the normal velocity due to the relative velocity between the airfoil and freestream, i.e.,

$$\mu = -\vec{u}_r \cdot \vec{\nu} \cdot \tag{11}$$

The source strength at the trailing edge is equal to zero.

The above boundary conditions provide (2*NE+2) equations for (2*NE+NW+NS+1) unknown source and vortex strengths on the airfoil and wake surfaces. Solutions for steady flows are accomplished by solving a reduced set of equations in which there are no wake terms and the Kelvin-Helmholtz theorem is inherently satisfied. Solutions for unsteady flows are accomplished by starting the motion impulsively so that there are initially no wake surfaces. Thereafter, the wake surface at each time step is generated on the basis of the previous time step's solution so that the wake terms in Eq. 6 are known at all times and the number of unknowns is reduced by (NS+NW-1).

Once a solution for the source and vorticity strengths has been determined, the airloads can be calculated from the unsteady Bernoulli equation,

$$P = P_\infty - \rho \left(\frac{\partial\phi}{\partial t} + \frac{1}{2} u_e^2\right) \cdot \tag{12}$$

Behind a boundary layer separation point, consideration must be given to the change in the total head of the flow by an amount equal to the vorticity shedding rate.

## 2.2 Analysis Options

The current implementation of ADAM allows six basic types of solution:

* steady motion without boundary layer separation (Fig. 3)
* impulsively started rectilinear motion (Fig. 3)
* constant rate pitching motion (Fig. 4)
* sinusoidal pitch oscillations (Fig. 5)
* sinusoidal plunging motion (Fig. 6)
* Simulated Darrieus motion (Fig. 7).

In all solution types, the input and output parameters are nondimensionalized by the airfoil chord length and freestream velocity. For an airfoil in unsteady rectilinear motion, the time scale is nondimensionalized to yield

$$\tilde{t} = u_\infty t/c \ . \tag{13}$$

An airfoil undergoing constant rate pitching has its angle of attack determined by

$$\alpha = \theta_o + \omega t \tag{14}$$

and in nondimensional form as

$$\alpha = \theta_o + \frac{\omega c}{u_\infty} \frac{u_\infty t}{c} \tag{15}$$

or

$$\alpha = \theta_o + \tilde{\omega}\tilde{t} \ . \tag{16}$$

The corresponding relation for an airfoil in periodic pitch oscillations is

$$\alpha = \theta_o + \cos \frac{\omega c}{u_\infty} \frac{u_\infty t}{c} \tag{17}$$

or

$$\alpha = \theta_o + \cos \tilde{\omega}\tilde{t} \ . \tag{18}$$

For an airfoil in periodic plunging motion, the instantaneous lateral position is given in nondimensional form as

$$\frac{Y_a}{c} = \frac{Y_{max}}{c} \cos \frac{\omega c}{u_\infty} \frac{u_\infty t}{c} \tag{19}$$

or

$$\tilde{Y}_a = \tilde{Y}_{max} \cos \tilde{\omega}\tilde{t} \ . \tag{20}$$

A one-bladed Darrieus turbine is simulated by utilizing an unsteady, translational airfoil motion which produces the same angular orientation and relative velocity between the airfoil and fluid.

Referring to Fig. 7, the approximate instantaneous relative velocity between the freestream and a Darrieus turbine blade is given by

$$u_r^* \cong R\omega + u_\infty \cos \omega t \ . \tag{21}$$

An equivalent relative velocity can be generated in an unsteady translational airfoil motion by utilizing a freestream velocity equivalent to the blade-speed of the Darrieus turbine and an airfoil motion which produces a periodic relative velocity of magnitude equal to the actual freestream velocity, i.e., for the simulation,

$$u_r = u_\infty - u_a \tag{22}$$

where

$$u_\infty = R\omega$$

$$u_a = -u_\infty^* \cos \omega t \, .$$

Nondimensionalizing by the freestream velocity ($R\omega$ in the simulation) and the airfoil chord length yields

$$\frac{u_r}{R\omega} = 1 + \frac{u_\infty^*}{R\omega} \cos \frac{R\omega t}{c} \frac{c}{R} \tag{23}$$

or

$$\tilde{u}_r = 1 + \frac{1}{\lambda} \cos \tilde{t}/\tilde{R} \, . \tag{24}$$

The time dependent airfoil translational position is found by integrating the airfoil velocity expression which yields

$$X_a = -\frac{u_\infty^*}{\omega} \sin \omega t \, . \tag{25}$$

Nondimensionalizing by the freestream velocity (tip-speed) and chord length results in

$$\frac{X_a}{c} = -\frac{u_\infty^*}{R\omega} \frac{R}{c} \sin \frac{R\omega t}{c} \frac{c}{R} \tag{26}$$

or

$$\tilde{X}_a = \tilde{R}/\lambda \sin \tilde{t}/\tilde{R} \, .$$

For the Darrieus turbine as represented in the simulation, the angle of incidence between the relative velocity vector and the blade is

$$\alpha = \tan^{-1} \left\{ \frac{u_\infty^* \sin \omega t}{R\omega + u_\infty^* \cos \omega t} \right\} \, . \tag{27}$$

Nondimensionalizing yields

$$\alpha = \tan^{-1} \left\{ \frac{\sin \frac{R\omega t}{c} \frac{c}{R}}{\frac{R\omega}{u_\infty^*} + \cos \frac{R\omega t}{c} \frac{c}{R}} \right\} \tag{28}$$

or

$$\alpha = \tan^{-1} \left\{ \frac{\sin \tilde{t}/\tilde{R}}{\lambda + \cos \tilde{t}/\tilde{R}} \right\} \, . \tag{29}$$

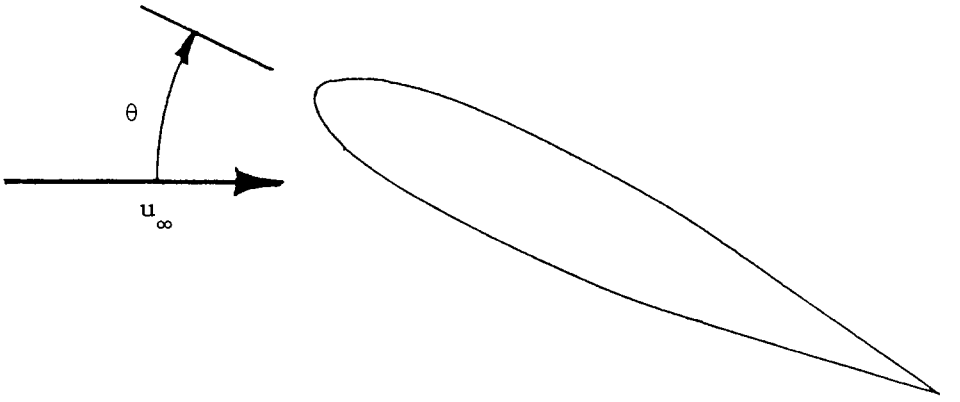


Fig. 3 Airfoil at Constant Angle of Attack

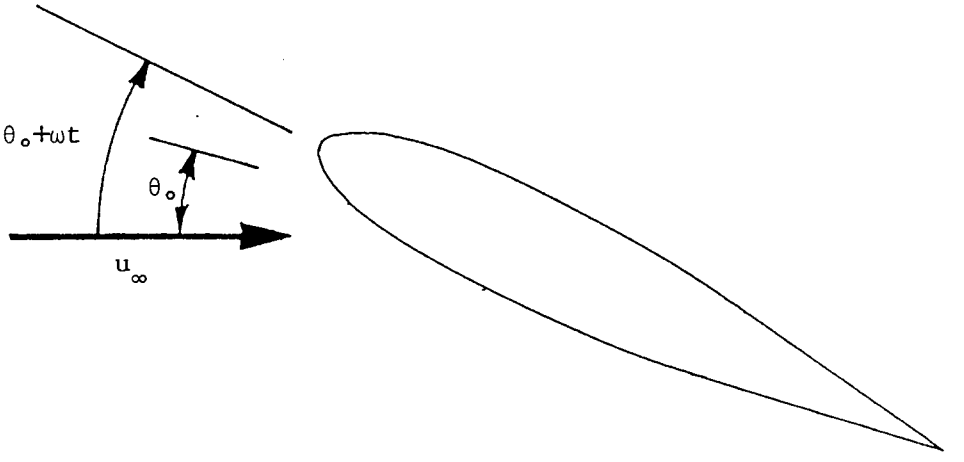


Fig. 4 Airfoil in Constant Rate Pitching

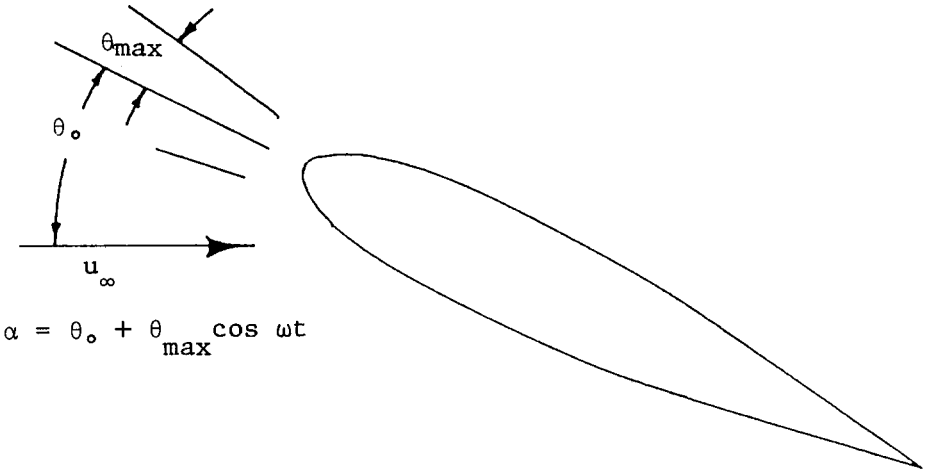


Fig. 5 Airfoil with Periodic Pitch Oscillations

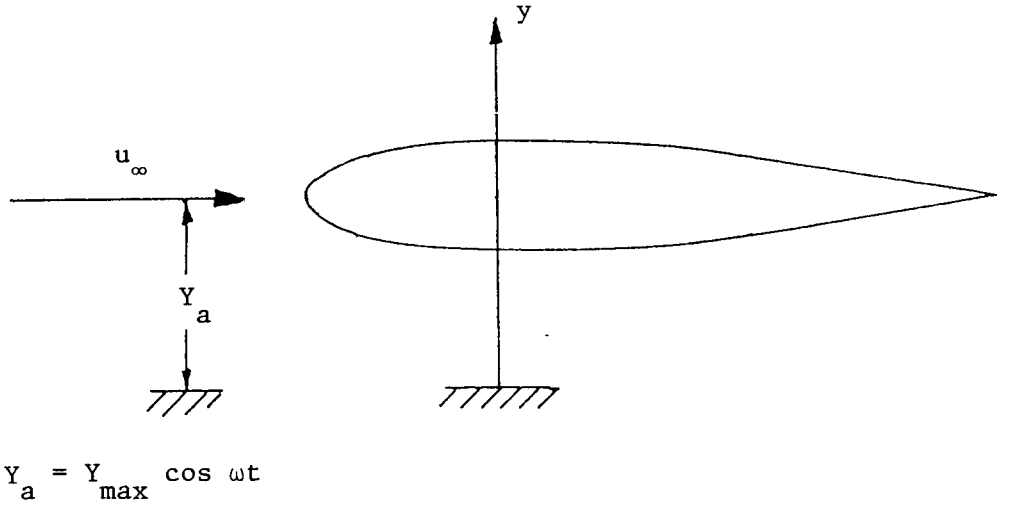


Fig. 6 Airfoil with Periodic Lateral Plunging Motion

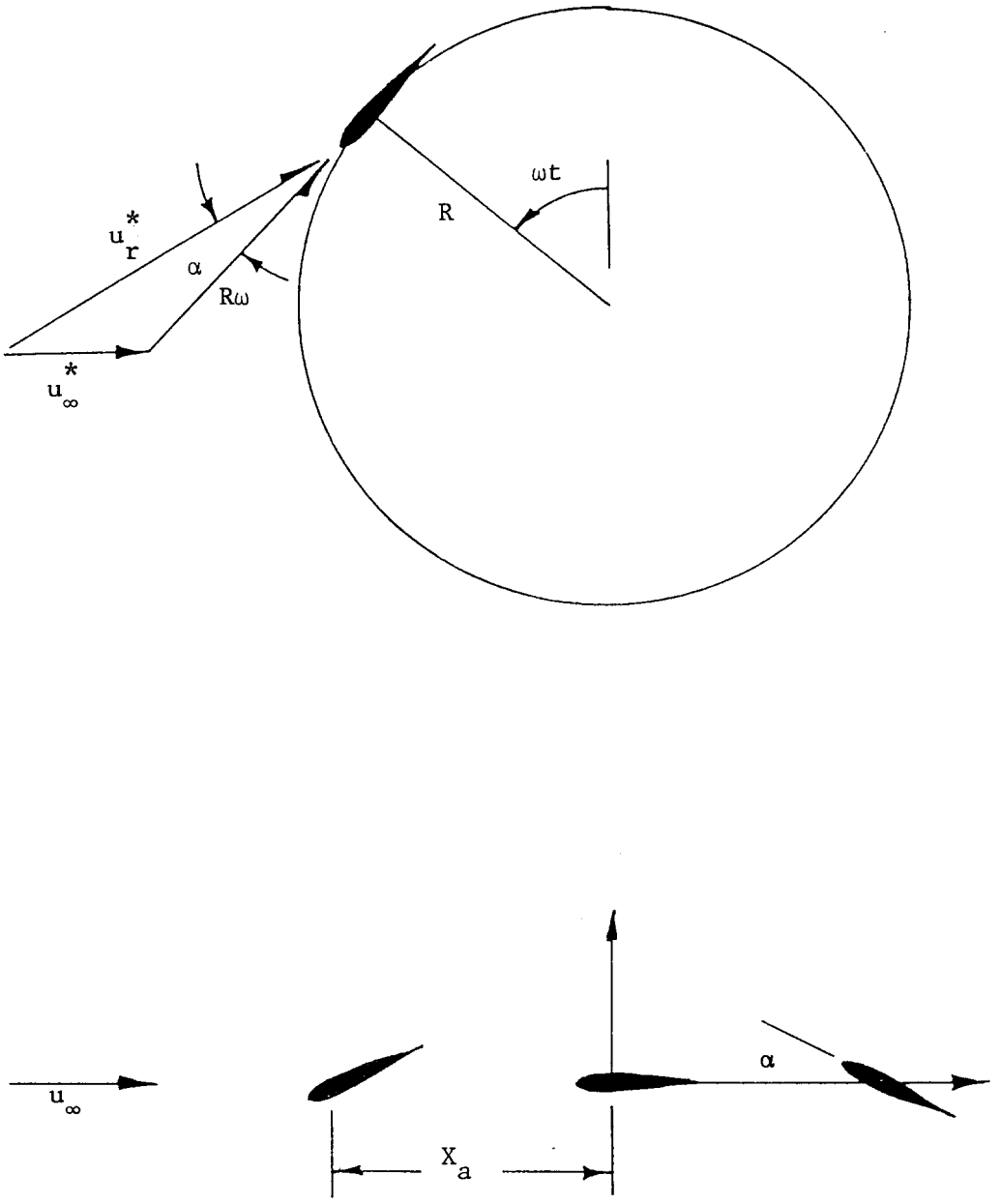


Fig. 7 Translational Simulation of the Blade Motion of a Darrieus Turbine

# 3. PROGRAM DESCRIPTION

This chapter contains adequate information to allow the preparation of data decks for ADAM, interpretation of output, and program revisions.

## 3.1 Input Requirements

The following is a list of the data images required to execute ADAM and a sample data file.

HEADER - format: 50H

Arbitrary program heading used to document analysis.

REY,STEADY,TECON - format: E11.3,1X,L1,F10.4

Reynolds number based on chord length and freestream velocity.
Logic variable used to select a steady or unsteady analysis.
Distance from the trailing edge to the control point of the nascent vortex given as a percent of chord.

NE,RX,RY,THETA,A1,SHIFT - format: I3,2F8.5,F7.5,A1,F8.5

Number of elements used to model airfoil surface.
Initial X and Y coordinates (nondimensionalized by the chord length) of the airfoil reference system with respect to the inertial reference system.
Initial angular orientation of the airfoil reference system with respect to the inertial reference system - equivalent to the angle of attack.
Alphanumeric variable which indicates that THETA was given in degrees (instead of radians) when A1='D'.
Chordwise position of the airfoil fixed reference frame.

BODY,ND0,ND1,ND2,ND3 - format: A4,1X,4I1

Airfoil type: NACA, CYLNder, laminar airfoil (default).
Four digit NACA airfoil designation - unnecessary for cylinder or laminar airfoil.

MCASE,THT0,A2,OMEGA,A3 - format: I5,F9.7,A1,F9.7,A1

Body motion designation: 1 - Impulsively started rectilinear motion.
2 - Sinusoidal oscillations in pitch.

3 - Sinusoidal lateral plunging motion.
4 - Darrieus turbine simulation.

Depending upon the body motion selected, the value input for THTO will specify the mean angle of attack for sinusoidal pitch oscillations or starting angle for a constant rate pitching motion.

Alphanumeric variable which indicates that THTO was given in degrees (instead of radians) when A2='D'.

The pitch rate for a constant rate pitching motion given in angular units per chord length of travel.

Alphanumeric variable which indicates that OMEGA was given in degrees (instead of radians) per chord length of travel when A3='D'.

THTMAX,A4,PLGMAX,FREQ - format: F9.7,A1,F10.7,F10.7

Oscillation amplitude (degrees) for sinusoidal pitch oscillations - no meaning for other body motions.

Alphanumeric variable which indicates that THTMAX was given in degrees when A1='D'.

Lateral plunging motion displacement amplitude (non-dimensionalized by the chord length) - no meaning for other body motions.

Frequency of either pitch or plunging oscillations in radians per chord length of travel.

RADIUS,RAMDA - format: 2F10.4

Radius (non-dimensionalized by chord length) of a simulated single bladed Darrieus turbine - no meaning for other body motions.

Tip-to-windspeed ratio for a Darrieus turbine - no meaning for other body motions.

XSTA(K), K=1, NE/2+1 - format: F8.5

Chordwise stations (non-dimensionalized by the chord) of element endpoints given in trailing to leading edge sequence with respect to a reference system at the leading edge of the airfoil. These card images are not read for the 'CYLN' body type.

PXTEM(K),PYTEM(K), K=1,62 - format: 2F10.5

Surface coordinates (given below) for an NLF - 0416 airfoil. The element endpoint coordinates used in the analysis will be the X stations specified in the previous card images and an interpolation from the NLF - 0416 data. These card images are not read for the 'NACA' and 'CYLN' body types.

```
001.00000 000.00000        <-------- PXTEM, PYTEM (NLF-0416)
   .99633    .00065
   .98520    .00242
   .96638    .00478
   .93978    .00696
   .90576    .00792
   .86536    .00667
```

| | |
|---|---|
| .82012 | .00278 |
| .77156 | -.00364 |
| .72107 | -.01221 |
| .67014 | -.02231 |
| .62019 | -.03250 |
| .57122 | -.04063 |
| .52175 | -.04614 |
| .47150 | -.05009 |
| .42127 | -.05291 |
| .37167 | -.05462 |
| .32326 | -.05529 |
| .27659 | -.05494 |
| .23218 | -.05357 |
| .19050 | -.05121 |
| .15200 | -.04787 |
| .11708 | -.04361 |
| .08609 | -.03847 |
| .05933 | -.03254 |
| .03708 | -.02594 |
| .01956 | -.01883 |
| .00709 | -.01154 |
| .00073 | -.00439 |
| 000.00000 | 000.00000 |
| .00049 | .00403 |
| .00509 | .01446 |
| .01393 | .02573 |
| .02687 | .03729 |
| .04383 | .04870 |
| .06471 | .05964 |
| .08936 | .06984 |
| .11761 | .07904 |
| .14925 | .08707 |
| .18404 | .09374 |
| .22169 | .09892 |
| .26187 | .10247 |
| .30422 | .10425 |
| .34839 | .10405 |
| .39438 | .10162 |
| .44227 | .09729 |
| .49172 | .09166 |
| .54204 | .08515 |
| .59256 | .07801 |
| .64262 | .07047 |
| .69155 | .06272 |
| .73872 | .05493 |
| .78350 | .04724 |
| .82530 | .03977 |
| .86357 | .03265 |
| .89779 | .02594 |
| .92749 | .01974 |
| .95224 | .01400 |
| .97197 | .00862 |
| .98686 | .00398 |
| .99656 | .00098 |
| 1.00000 | .00000 |

The following is a sample data file:

```
 IMPULSIVELY STARTED NACA 0012 AT 10 DEG. AOA
00010 0000.1000 0190                    <--- NSTEPS, DT, NWMAX
001.000D+06 F 0002.0000                 <--- REY, STEADY, TECON
028 0.00000 0.00000 010.00D 0.25000     <--- NE, RX, RY, THETA, SHIFT
NACA 0012                               <--- BODY, ND0, ND1, ND2, ND3
00001 000.0000D 000.0000R               <--- MCASE, THT0, OMEGA
000.00000D 0.0000000 0.0000000          <--- THTMAX, PLGMAX, FREQ
00000.0000 0000.0000                    <--- RADIUS, RAMDA
 1.0000                                 <--- XSTA
  .9600
  .9000
  .8200
  .7100
  .6000
  .4950
  .3900
  .2900
  .1900
  .1050
  .0520
  .0200
  .0050
  .0
```

## 3.2 Program Output

Each execution of ADAM provides the following basic elements of output:

INITIALIZATION

* echo of input data
* airfoil geometric parameters including:
  - airfoil ref. frame and surface coordinates of element endpoints and centroids
  - normal and tangential unit vector components at the element centroids
* source and vorticity influence coefficient matrices

INCREMENTAL TIME STEP SOLUTIONS

* airfoil position and velocity with respect to the inertial reference frame
* angular transformation matrix for coordinate transformations between the inertial and airfoil fixed reference frames
* separation data from the previous time step, if any
* breakdown of the contributions to the normal downwash on the airfoil
* source and vorticity distributions determined for the current time step

* coordinates and strengths of all wake vortices after convection to new locations
* total bound vorticity
* stagnation point location
* development of boundary layer parameters with locations of transition and separation
* pressure distribution
* integrated airloads

### 3.3 Subroutine Calling Sequence

The following provides a flow chart of the calling sequence for the subroutines in ADAM.

```
MAIN
      INPUT
            NACAGM
      GEOM
      COEFA
            VORTEX
      DECOMP
      COEFB
            SOURCE
      MOTION
      WAKVEL
            GETVEL
                  SOURCE
                  VORTEX
                  LMPVTX
      TEGEOM
            CROSS
                  NACAGM
      SPGEOM
            CROSS
                  NACAGM
      WAKINF
            VORTEX
            LMPVTX
      DECOMP
      DWASH
      SOLVE
            GETVEL
                  SOURCE
                  VORTEX
                  LMPVTX
      SEPCAL
            SPDATA
            BNDL
                  STAG
                  LAMBL
                        GRAD
                        RKM44
                              F
```

```
                                        TRANS
                                        TSEP
                TURBL
                      GRAD
                      RKM44
                            F
                                        TRANS
                                        TSEP
          DBNDL
                TURBL
                      DSTAG
                      DLAMBL
                      GRAD
                      RKM44
                            F
                                        TRANS
                                        TSEP
                DTURBL
                      DGRAD
                      DRKM44
                            DF
                                        DTRANS
                                        DTSEP
                DTURBL
                      DGRAD
                      DRKM44
                            DF
                                        DTRANS
                                        DTSEP
     CPDIST
```

### 3.4 Subroutine Functions

The following is a short description of the function of each subroutine in ADAM.

Subroutine Functions:

BNDL,DBNDL -Interface to potential flow routines and driver for boundary layer calculations on upper and lower sides of the airfoil, respectively.

COEFA - Calculates the normal induced velocity coefficient matrix associated with the airfoil surface vorticity distribution.

COEFB - Calculates the normal induced velocity coefficient matrix associated with the airfoil surface source distribution.

CPDIST - Calculates the pressure coefficient distribution on the airfoil and the integrated airloads.

COV - Calculates the center of vorticity of a linearly distributed vorticity element.

CROSS - Adjusts coordinates of separated wake vortices to maintain a minimum separation with airfoil surface.

DECOMP - Performs an LU decomposition of the influence coefficient

matrix A.

DWASH - Calculates the source and freestream induced normal downwash on the airfoil.

F,DF - Solves simultaneous equations involved in Runge-Kutta integration method.

GEOM - Evaluates the airfoil element geometric parameters.

GETVEL - Evaluates the total disturbance velocity at specified point.

GRAD,DGRAD - Evaluates the surface gradient of the edge velocity.

INPUT - Reads and interprets problem definition data set.

LAMBL,DLAMBL - Predicts the development of the laminar portion of the boundary layer on the upper and lower surfaces of the airfoil, respectively.

LMPVTX - Evaluates the velocity induced by a lumped vortex.

MAIN - Mainline routine which controls overall program execution.

MOTION - Calculates the instantaneous rectilinear and angular velocity of the body and evaluates the transformation matrix between the body-fixed and inertially-fixed reference frames.

NACAGM - Generates surface coordinates for NACA four digit airfoils.

RKM44 - Fourth order Runge-Kutta solution routine.

SEPCAL - Interface to boundary layer calculations, provides separation point properties for potential flow calculations.

SOLVE - Solves for unknown airfoil vorticity distribution.

SOURCE - Evaluates the velocity induced by an airfoil surface element.

SPDATA - Provides tabulated boundary layer separation point locations as a function of angle of attack.

SPGEOM - Computes new geometry for wake surface emanating from boundary layer separation point.

STAG,DSTAG - Determines the initial boundary layer characteristics at the stagnation point for the upper and lower surfaces, respectively.

TEGEOM - Computes new geometry for wake surface emanating from airfoil trailing edge.

TRANS,DTRANS - Tests for the transition from a laminar to a turbulent boundary layer on the upper and lower surfaces of the airfoil,

respectively.

TSEP,DTSEP - Tests for the intermittent separation of the turbulent boundary layer on the upper and lower surfaces of the airfoil, respectively.

TURBL,DTURBL - Predicts the development of the turbulent portion of the boundary layer on the upper and lower surfaces of the airfoil, respectively.

VORTEX - Calculates the velocity induced by an airfoil vortex element.

WAKINF - Calculates the normal downwash on the airfoil due to the wake surfaces.

WAKVEL - Calculates the total induced disturbance velocity in the wake surfaces.

## 3.5 Common Block Variable Definitions

The critical variables involved in the calculations performed by ADAM are contained in fifteen common blocks. A short description of each of these variables is given below.

BLK0 : STEADY - Logic variable indicating a steady (true) or unsteady analysis.
USEP - Logic variable indicating that the boundary layer is separated from the upper surface of the airfoil.
DSEP - Logic variable indicating that the boundary layer is separated from the lower surface of the airfoil.

BLK1 : NSTEPS - Total number of time steps.
NT - Current time step.
NE - Number of elements on the airfoil surface.
ND0,ND1,ND2,ND3 - Four digit discriptor for a NACA series airfoil.
MCASE - Body motion case number.
IP - LU decomposition pivot array.

BLK2 : PI - 3.14159265358979267
TWOPI - 2 * PI
REY - Reynolds number based on chord length and freestream velocity.
RX,RY - Vector coordinates of the airfoil fixed reference frame with respect to the inertially fixed frame.
THETA - Angular orientation of the airfoil fixed reference frame with respect to the inertially fixed frame.
SHIFT - Chordwise distance between the leading edge of a NACA airfoil and the origin of the airfoil fixed reference frame.
DT - Time step duration.
TECON - Distance from trailing edge to nascent vortex control

point.

BLK3 : PX,PY - Array of element endpoint coordinates.
CX,CY - Array of element centroid coordinates.
S - Array of surface coordinates of the element centroids.
SP - Array of surface coordinates of the element endpoints.
DSBD - Array of element lengths.

BLK4 : ESX,ESY - Array of element tangential unit vectors.
ENX,ENY - Array of element normal unit vectors.
ETN - Array of angles between the element tangential unit vectors and the x axis of airfoil ref. frame.

BLK5 : A - Surface vorticity induced, normal velocity coefficient matrix.
B - Surface source induced, normal velocity coefficient matrix.
DW - Array of the total downwash on the airfoil elements due to the relative motion between the airfoil and freestream, vorticity distribution on the wake, and source distribution on the airfoil.
PERMA - Storage matrix for original normal induced velocity coefficient matrix.

BLK6: UBX,UBY - Velocity vector of the airfoil fixed ref. frame with respect to the inertial ref. frame.
OMEGA - Angular velocity of the airfoil fixed ref. frame with respect to the inertial ref. frame.
THTO - Median angle of attack for an airfoil oscillating in pitch
THTMAX - Amplitude of sinusoidal angle of attack oscillations.
FREQ - Frequency of angle of attack oscillations or lateral plunging motion.
PLGMAX - Amplitude of sinusoidal lateral plunging motion.
RAMDA - Tip-to-Windspeed ratio for a Darrieus turbine motion.
RADIUS - Darrieus turbine radius.

BLK7: URX,URY - Array of relative velocity vectors between the element centroids and the freestream velocity.
C - Array of the normal velocity induced at the element centroids due to the source distribution on the airfoil.
D - Array of the normal relative velocity component between the element centroids and the freestream velocity.
E - Array of the normal velocity induced at the element centroids due to the vorticity distribution on the wake surface extending from the trailing edge.
F - Array of the normal velocity induced at the element centroids due to the vorticity distribution on the wake surface extending from the boundary layer separation point.

BLK8: SIGMA - Array of the airfoil source distribution strength at the element centroids.
GAM - Array of the airfoil vorticity distribution strength at the element endpoints.

CP - Array of pressure coefficients at the element centroids.

BLK9: TANVEL - Array of the tangential component of the relative velocity between the airfoil and freestream at the element control points.
EDGVEL - Array of the total tangential velocity at the element control points.
GAMVEL - Array of the self-induced tangential velocity at the centroid of each element control point.

BLK10: CIRCU - Total bound vorticity on the airfoil surface.
STAGPT - Surface coordinate at which the EDGVEL is zero.
CONST - Undefined and not utilized.
T - Transformation matrix for inertial to airfoil fixed coordinate changes.

BLK11: NTEV - Number of wake elements in the wake surface extending from the trailing edge.
NSPV - Number of wake elements in the wake surface extending from the boundary layer separation point.
NWMAX - Oldest trailing edge wake element for which contributions to the disturbance velocity field are included.
NOSP - Number of airfoil element on which boundary layer separation occurred.
NSTAG - Number of airfoil element on which the stagnation point occurs.

BLK12: TEVX,TEVY - Array of inertial coordinates of the wake elements generated from the trailing edge.
TEVS - Array of strengths of the wake elements generated from the trailing edge.
DSTE - Length of the nascent wake element generated at the trailing edge.
TELS - Strength of the vorticity distribution at the upstream edge of the nascent trailing edge wake element.
TERS - Strength of the vorticity distribution at the downstream edge of the nascent trailing edge wake element.

BLK13: SPVX,SPVY - Array of inertial coordinates of wake elements generated at the boundary layer separation point.
SPVS - Array of strengths of the wake elements generated from the boundary layer separation point.
DSSP - Length of the nascent wake element generated at the boundary layer separation point.

BLK14: TE2X,TE2Y - Inertial coordinates of the downstream edge of the nascent wake element generated at the trailing edge.
SP2X,SP2Y - Inertial coordinates of the nascent vortex generated at the boundary layer separation point.
SPX, SPY - Coordinates of the boundary layer separation point with respect to the airfoil fixed ref. frame.

BLK15: UXTE,UYTE - Array of the inertial velocity vectors for the wake elements generated at the trailing edge.

UXSP,UYSP - Array of the inertial velocity vectors for the wake elements generated at the boundary layer separation point.

# APPENDIX

## Program Listing of Adam

```
      SUBROUTINE BNDL(USURF,S,NE,STAGPT,REC,SPPS,SEPRT,NT,DT)
C*****************************************C
C MAIN PROGRAM FOR BOUND CHECKOUT         C
C*****************************************C
      IMPLICIT REAL*8(A-H,O-Z)
      REAL*8 USURF(1),S(1),STAGPT,REC,SPPS
      LOGICAL SPRT,SEPRT
      COMMON/VEL/U(150),X(150),NS
      COMMON/FIN/RST,FT,AT,GT,HT,XNS,XNL,XNT,TURB,NTM
      COMMON/OLD/OAT(150),ODSL(150),OXNL,ONT,ONSEP,XSEP,OXSEP
      COMMON/BLI/HI,THEI,RTHI
      COMMON/PTN/SPRT
      DO 10 I=1,NE
      U(I)=USURF(I)
   10 X(I)=S(I)
      NS=NE
      XNS=STAGPT
      REL=REC
      ONSEP=0.D0
      OXSEP=X(NS)
      NTM=NT
      DELT=DT
      XSEP=SPPS
      SPRT=SEPRT
      OXNL=XNS
      CALL STAG(REL,HIL,THEIL)
      TURB=0.D0
      CALL LAMBL(*20,DELT,HIL,THEIL,REL)
      TURB=1.D0
      CALL TURBL(DELT,HI,THEI,REL)
      SPPS=XSEP
      SEPRT=SPRT
   20 RETURN
      END
```

```
      SUBROUTINE COEFA
C
C THIS ROUTINE IS TO COMPUTE COEFFICIENTS FOR THE NORMAL DOWNWASH
C CONTRIBUTION DUE TO VORTEX DISTRIBUTION
C
      IMPLICIT REAL*8 (A-H,O-Z)
      REAL*8 CCX(41),CCY(41),CDX(41),CDY(41),TEMPA(41)
      LOGICAL STEADY,USEP,DSEP
C
      COMMON/BLK0/STEADY,USEP,DSEP
      COMMON/BLK1/NSTEPS,NT,NE,ND0,ND1,ND2,ND3,MCASE,IP(40)
      COMMON/BLK2/PI,TWOPI,REY,RX,RY,THETA,SHIFT,DT,TECON
      COMMON/BLK3/PX(41),PY(41),CX(40),CY(40),S(40),SP(41),DSBD(40)
      COMMON/BLK4/ESX(41),ESY(41),ENX(40),ENY(40),ETN(40)
      COMMON/BLK5/A(40,40),B(40,40),DW(40),PERMA(40,40)
      COMMON/BLK6/UBX,UBY,OMEGA,THT0,THTMAX,FREQ,PLGMAX,RAMDA,RADIUS
      COMMON/BLK7/URX(40),URY(40),C(40),D(40),E(40),F(40)
      COMMON/BLK8/SIGMA(40),GAM(40),CP(40)
      COMMON/BLK9/TANVEL(40),EDGVEL(40),GAMVEL(40)
      COMMON/BLK10/CIRCU,STAGPT,CONST,T(2,2)
      COMMON/BLK11/NTEV,NSPV,NWMAX,NOSP,NSTAG
      COMMON/BLK12/TEVX(201),TEVY(201),TEVS(201),DSTE,TELS,TERS
      COMMON/BLK13/SPVX(201),SPVY(201),SPVS(201),DSSP
      COMMON/BLK14/TE2X,TE2Y,SP2X,SP2Y,SPPS,SPX,SPY
      COMMON/BLK15/UXTE(201),UYTE(201),UXSP(201),UYSP(201)
C
      DO 1050 I=1,NE+2
        IF(I.EQ.NE+1) GO TO 1050
C
C GET THE INDUCED VELOCITY FROM JTH SOURCE TO ITH FIELD WRT BFF
C
        DO 1000 J=1,NE
          CALL VORTEX(CX(J),CY(J),CX(I),CY(I),ESX(J),ESY(J),DSBD(J),
     &                VV1X,VV1Y,VV2X,VV2Y)
          CCX(J)=.5D0*VV1X-VV2X/DSBD(J)
          CCY(J)=.5D0*VV1Y-VV2Y/DSBD(J)
          CDX(J)=.5D0*VV1X+VV2X/DSBD(J)
          CDY(J)=.5D0*VV1Y+VV2Y/DSBD(J)
 1000   CONTINUE
C
C GET THE COEFFICIENTS "A"
C
        A(I,1)=ENX(I)*CCX(1)+ENY(I)*CCY(1)
        A(I,NE+1)=ENX(I)*CDX(NE)+ENY(I)*CDY(NE)
        DO 1010 J=2,NE
          A(I,J)=ENX(I)*(CDX(J-1)+CCX(J))+ENY(I)*(CDY(J-1)+CCY(J))
 1010   CONTINUE
 1050 CONTINUE
C
C BRANCHING FOR STEADY CASE
C
      IF(STEADY) GO TO 2000
C
      A(NE+1,1)=.5D0*DSBD(1)
      A(NE+1,NE+1)=.5D0*DSBD(NE)
      DO 1060 J=2,NE
        A(NE+1,J)=.5D0*(DSBD(J-1)+DSBD(J))
 1060 CONTINUE
C
C PRINT THE COEFFICIENTS "A"
```

```
C
      WRITE(6,6000)
      DO 1080 K=1,NE+2
        WRITE(6,6010) K,(A(K,L),L=1,NE+1)
 1080 CONTINUE
C
C  SAVE MATRIX A IN PERMANENT PLACE FOR LATER USE
C
      DO 1066 I=1,NE+2
        DO 1066 J=1,NE+1
          PERMA(I,J)=A(I,J)
 1066 CONTINUE
C
      RETURN
C
C  CASES FOR STEADY STATE
C
 2000 CONTINUE
      WRITE(6,6020)
      DO 2010 K=2,NE
        A(NE+1,K)=0.D0
 2010 CONTINUE
      A(NE+1,1)=1.D0
      A(NE+1,NE+1)=1.D0
C
      RETURN
C
 6000 FORMAT(///' ***** COEF. "A" FOR VORTEX DISTRIBUTION *****'//)
 6010 FORMAT(' ROW',I3,9(1X,E10.4,1X),50(/7X,9(1X,E10.4,1X)))
 6020 FORMAT(///' ********************************************',
     &         /' *************  STEADY  STATE   **************',
     &         /' ********************************************'/)
C
      END
```

```
      SUBROUTINE COEFB
C
C THIS ROUTINE IS TO COMPUTE THE COEFFICIENTS FOR NORMAL VELOCITY
C CONTRIBUTION DUE TO THE SOURCE DISTRIBUTION
C
      IMPLICIT REAL*8 (A-H,O-Z)
      REAL*8 CEX(41),CEY(41),CFX(41),CFY(41),CGX(41),CGY(41)
C
      COMMON/BLK0/STEADY,USEP,DSEP
      COMMON/BLK1/NSTEPS,NT,NE,ND0,ND1,ND2,ND3,MCASE,IP(40)
      COMMON/BLK2/PI,TWOPI,REY,RX,RY,THETA,SHIFT,DT,TECON
      COMMON/BLK3/PX(41),PY(41),CX(40),CY(40),S(40),SP(41),DSBD(40)
      COMMON/BLK4/ESX(41),ESY(41),ENX(40),ENY(40),ETN(40)
      COMMON/BLK5/A(40,40),B(40,40),DW(40),PERMA(40,40)
      COMMON/BLK6/UBX,UBY,OMEGA,THTO,THTMAX,FREQ,PLGMAX,RAMDA,RADIUS
      COMMON/BLK7/URX(40),URY(40),C(40),D(40),E(40),F(40)
      COMMON/BLK8/SIGMA(40),GAM(40),CP(40)
      COMMON/BLK9/TANVEL(40),EDGVEL(40),GAMVEL(40)
      COMMON/BLK10/CIRCU,STAGPT,CONST,T(2,2)
      COMMON/BLK11/NTEV,NSPV,NWMAX,NOSP,NSTAG
      COMMON/BLK12/TEVX(201),TEVY(201),TEVS(201),DSTE,TELS,TERS
      COMMON/BLK13/SPVX(201),SPVY(201),SPVS(201),DSSP
      COMMON/BLK14/TE2X,TE2Y,SP2X,SP2Y,SPPS,SPX,SPY
      COMMON/BLK15/UXTE(201),UYTE(201),UXSP(201),UYSP(201)
C
      DO 1050 I=1,NE+2
        IF(I.EQ.NE+1) GO TO 1050
C
C GET THE INDUCED VELOCITY FROM JTH SOURCE TO ITH FIELD WRT BFF
C
        DO 1000 J=1,NE
          TEMP1=2.D0/DSBD(1)
          TEMP2=2.D0/DSBD(NE)
          IF(J.NE.1) TEMP1=2.D0/(DSBD(J-1)+DSBD(J))
          IF(J.NE.NE) TEMP2=2.D0/(DSBD(J+1)+DSBD(J))
          CALL SOURCE(CX(J),CY(J),CX(I),CY(I),ESX(J),ESY(J),DSBD(J),
     &                VS1X,VS1Y,VSLX,VSLY,VSRX,VSRY)
          CEX(J)=-TEMP1*VSLX
          CEY(J)=-TEMP1*VSLY
          CFX(J)=VS1X+TEMP1*VSLX-TEMP2*VSRX
          CFY(J)=VS1Y+TEMP1*VSLY-TEMP2*VSRY
          CGX(J)=TEMP2*VSRX
          CGY(J)=TEMP2*VSRY
 1000   CONTINUE
C
C COMPUTE THE COEFFICIENTS "B"
C
        B(I,1)=ENX(I)*(CFX(1)+CEX(2))+ENY(I)*(CFY(1)+CEY(2))
        B(I,NE)=ENX(I)*(CGX(NE-1)+CFX(NE))+ENY(I)*(CGY(NE-1)+CFY(NE))
        DO 1010 J=2,NE-1
          B(I,J)=ENX(I)*(CGX(J-1)+CFX(J)+CEX(J+1))
     &          +ENY(I)*(CGY(J-1)+CFY(J)+CEY(J+1))
 1010   CONTINUE
C
 1050 CONTINUE
C
C PRINT THE COMPUTED COEFFICIENT "B"
C
      WRITE(6,6000)
      DO 1100 K=1,NE+2
```

```
        WRITE(6,6010) K,(B(K,L),L=1,NE)
 1100 CONTINUE
C
      RETURN
C
 6000 FORMAT(////' ***** COEF. "B" FOR SOURCE DISTRIBUTION *****'//)
 6010 FORMAT(' ROW',I3,9(1X,E10.4,1X),50(/7X,9(1X,E10.4,1X)))
C
      END
```

```
      REAL FUNCTION COV(SL,SR)
C
     IMPLICIT REAL*8 (A-H,O-Z)
C
      TEMP1=SL+SR
      COV=.5D0
      IF(TEMP1.EQ.0.D0) RETURN
      COV=(1.D0+SR/TEMP1)/3.D0
      RETURN
C
      END
```

```
      SUBROUTINE CPDIST
C
C THIS ROUTINE TO CALCULATE THE PRESSURE DISTRIBUTIONS
C
      IMPLICIT REAL*8 (A-H,O-Z)
      REAL*8 T1(40),T2(40),T3(40),T4(40),T5(40),T6(40)
      REAL*8 OLDVEL(40),DISVEL(40)
      LOGICAL STEADY,USEP,DSEP
C
      COMMON/BLK0/STEADY,USEP,DSEP
      COMMON/BLK1/NSTEPS,NT,NE,ND0,ND1,ND2,ND3,MCASE,IP(40)
      COMMON/BLK2/PI,TWOPI,REY,RX,RY,THETA,SHIFT,DT,TECON
      COMMON/BLK3/PX(41),PY(41),CX(40),CY(40),S(40),SP(41),DSBD(40)
      COMMON/BLK4/ESX(41),ESY(41),ENX(40),ENY(40),ETN(40)
      COMMON/BLK5/A(40,40),B(40,40),DW(40),PERMA(40,40)
      COMMON/BLK6/UBX,UBY,OMEGA,THT0,THTMAX,FREQ,PLGMAX,RAMDA,RADIUS
      COMMON/BLK7/URX(40),URY(40),C(40),D(40),E(40),F(40)
      COMMON/BLK8/SIGMA(40),GAM(40),CP(40)
      COMMON/BLK9/TANVEL(40),EDGVEL(40),GAMVEL(40)
      COMMON/BLK10/CIRCU,STAGPT,CONST,T(2,2)
      COMMON/BLK11/NTEV,NSPV,NWMAX,NOSP,NSTAG
      COMMON/BLK12/TEVX(201),TEVY(201),TEVS(201),DSTE,TELS,TERS
      COMMON/BLK13/SPVX(201),SPVY(201),SPVS(201),DSSP
      COMMON/BLK14/TE2X,TE2Y,SP2X,SP2Y,SPPS,SPX,SPY
      COMMON/BLK15/UXTE(201),UYTE(201),UXSP(201),UYSP(201)
C
C  COMPUTE THE PRESSURE DISTRIBUTION
C
      WRITE(6,6000)
      DO 1000  K=1,NE
        OLDVEL(K)=DISVEL(K)
        DISVEL(K)=EDGVEL(K)-TANVEL(K)
        T1(K)=(DISVEL(K)-OLDVEL(K))/DT
        IF(STEADY.OR.NT.EQ.0) T1(K)=0.D0
        T2(K)=DISVEL(K)*(URX(K)*ESX(K)+URY(K)*ESY(K))
        T3(K)=D(K)*D(K)
        T4(K)=DISVEL(K)*DISVEL(K)
        T5(K)=T2(K)-.5D0*T3(K)+.5D0*T4(K)
 1000 CONTINUE
C
      T1STG=T1(NSTAG-1)+(T1(NSTAG)-T1(NSTAG-1))*
     &      (STAGPT-S(NSTAG-1))/(S(NSTAG)-S(NSTAG-1))
      T5STG=T5(NSTAG-1)+(T5(NSTAG)-T5(NSTAG-1))*(STAGPT-S(NSTAG-1))/
     &       (S(NSTAG)-S(NSTAG-1))
C
      DO 1010 K=1,NE
        CP(K)=0.D0
        T5(K)=T5(K)-T5STG
 1010 CONTINUE
C
      T6(NSTAG)=.5D0*(T1STG+T1(NSTAG))*(S(NSTAG)-STAGPT)
      DO 1020 K=NSTAG+1,NE-1
        T6(K)=T6(K-1)+.5D0*(T1(K)+T1(K-1))*(S(K)-S(K-1))
 1020 CONTINUE
      T6(NE)=T6(NE-1)+.5D0*(T1(NE)+T1(NE-1))*(S(NE)-S(NE-1))+
     &       (SP(NE+1)-S(NE))*T1(NE)
C
      T6(NSTAG-1)=.5D0*(T1STG+T1(NSTAG-1))*(S(NSTAG-1)-STAGPT)
      DO 1030 K=NSTAG-2,2,-1
        T6(K)=T6(K+1)+.5D0*(T1(K)+T1(K+1))*(S(K)-S(K+1))
```

```
 1030 CONTINUE
      T6(1)=T6(2)+.5D0*(T1(1)+T1(2))*(S(1)-S(2))+(SP(1)-S(1))*T1(1)
C
      DO 1040 K=1,NE
        CP(K)=-2.D0*(T6(K)+T5(K))+1.D0
 1040 CONTINUE
C
      IF(.NOT.(USEP.OR.DSEP)) GO TO 200
      ADDCP=2.D0*SPVS(2)/DT
      IF(DSEP) GO TO 100
      IF(SPPS.LE.S(NOSP)) CP(NOSP)=CP(NOSP)+ADDCP
      DO 1045 K=NOSP+1,NE
        CP(K)=CP(K)+ADDCP
 1045 CONTINUE
      GO TO 200
  100 ADDCP=-ADDCP
      IF(SPPS.GE.S(NOSP)) CP(NOSP)=CP(NOSP)+ADDCP
      DO 1048 K=NOSP-1,1,-1
        CP(K)=CP(K)+ADDCP
 1048 CONTINUE
  200 CONTINUE
C
      WRITE(6,6010)(K,CX(K),CP(K),T1(K),T2(K),T3(K),T4(K),T5(K),
     &              T6(K),K=1,NE)
C
C  NOW GET THE INTEGRATED FORCE COEF.'S
C
      CN=0.
      CT=0.
      WRITE(6,6020)
      DO 1050 K=1,NE
        CN1=-CP(K)*ENY(K)
        CT1=-CP(K)*ENX(K)
        CL1=CT1*T(1,2)+CN1*T(2,2)
        CD1=CT1*T(1,1)+CN1*T(2,1)
        CN=CN+CN1*DSBD(K)
        CT=CT+CT1*DSBD(K)
        WRITE(6,6030) K,CN1,CT1,CL1,CD1,DSBD(K)
 1050 CONTINUE
      CL=CT*T(1,2)+CN*T(2,2)
      CD=CT*T(1,1)+CN*T(2,1)
C
C PRESSURE CALCULATIONS FOR DARRIEUS CASE
C
      IF(MCASE.NE.5) GO TO 300
      CD2=.00512D0*CL*CL+.006D0
      CN2=CD2*T(2,1)+CL*T(2,2)
      CT2=CD2*T(1,1)+CL*T(1,2)
      FN=CN2*RAMDA*RAMDA
      FT=CT2*RAMDA*RAMDA
      TEMP=DCOS(FLOAT(NT)*DT/RADIUS)
      FN=FN-TWOPI*(.25D0*RAMDA*RAMDA/RADIUS+.5D0*RAMDA*TEMP/RADIUS)
C
      TANG=THETA*180.D0/PI
      RANG=FLOAT(NT)*DT/RADIUS*180.D0/PI
C
      WRITE(6,6060) NT,CL,CN,FN,RANG,NT,CD,CT,FT,TANG
C
      RETURN
C
```

```
  300 CONTINUE
      TEMP=-THETA*180.D0/PI
      WRITE(6,6040) NT,CL,CN,ADDCP,TEMP,NT,CD,CT,TEMP
C
      RETURN
C
 6000 FORMAT(///' ***** CP DISTRIBUTIONS *****'//
     &  ' NO.',8X,'CX',12X,'CP',12X,'T1',12X,'T2',12X,'T3',12X,'T4'/)
 6010 FORMAT((1X,I3,3X,8(1X,D12.6,1X)))
 6020 FORMAT(/'  K',8X,'CN1',13X,'CT1',13X,'CL1',13X,'CD1',12X,'DSBD')
 6030 FORMAT(I4,5(3X,D13.6))
 6040 FORMAT(/' NT=',I3,' CL=',D11.4,' CN=',D11.4,' ADDCP=',F7.2,
     &         ' THETA=',F7.3,' DEG'/
     &         ' NT=',I3,' CD=',D11.4,' CT=',D11.4,' THETA=',F7.3,' DEG')
 6060 FORMAT(/' NT=',I3,' CL=',E11.4,' CN=',E11.4,' FN=',E11.4,
     &        ' RANG=',F8.3,' DEG'/
     &        ,' NT=',I3,' CD=',E11.4,' CT=',E11.4,' FT=',E11.4,
     &        ' TANG=',F8.3,' DEG'/)
C
      END
```

```
      SUBROUTINE CROSS(X,Y,T,RX,RY,SHIFT,I1,I2,I3,I4)
C
      IMPLICIT REAL*8 (A-H,O-Z)
      REAL*8 T(2,2)
C
C TRANSFORM THE X AND Y WRT B.F.F
C
      BX=X*T(1,1)+Y*T(1,2)-RX
      BY=X*T(2,1)+Y*T(2,2)-RY
C
C SPECIFY THE SECONDARY SURFACE DISTANCE FROM THE PHYSICAL SURFACE
C
      DIST=.02D0
C
      XLEFT=SHIFT
      XRITE=1.D0+SHIFT
      IF(BX.GE.XRITE.OR.BX.LE.XLEFT) RETURN
      YUP=.15D0/2.D0+DIST
      YLOW=-YUP
      IF(BY.GE.YUP.OR.BY.LE.YLOW) RETURN
C
C TEST THE CROSSING
C
      TX=BX-SHIFT
      CALL NACAGM(TX,AY,I1,I2,I3,I4,SHIFT)
      BX=TX
      AYU=AY+DIST
      AYL=-AYU
      IF(BY.GE.0.D0.AND.BY.LE.AYU) BY=AYU
      IF(BY.LT.0.D0.AND.BY.GE.AYL) BY=AYL
      IF(BY.GT.AYU.OR.BY.LT.AYL) RETURN
C
C TRANSFORM BX AND BY WRT I.F.F
C
      X=BX*T(1,1)+BY*T(2,1)+RX
      Y=BX*T(1,2)+BY*T(2,2)+RY
      WRITE(6,600) X,Y
  600 FORMAT(1X,'X = ',F8.5,2X,'Y = ',F8.5)
C
      RETURN
      END
```

```
      SUBROUTINE DBNDL(USURF,S,NE,STAGPT,REC,SPPS,SEPRT,NT,DT)
C*****************************************C
C MAIN PROGRAM FOR BOUND CHECKOUT          C
C*****************************************C
      IMPLICIT REAL*8(A-H,O-Z)
      REAL*8 USURF(1),S(1),STAGPT,REC,SPPS
      LOGICAL SPRT,SEPRT
      COMMON/DVEL/U(150),X(150),NS
      COMMON/DFIN/RST,FT,AT,GT,HT,XNS,XNL,XNT,TURB,NTM
      COMMON/DOLD/OAT(150),ODSL(150),OXNL,ONT,ONSEP,XSEP,OXSEP
      COMMON/DBLI/HI,THEI,RTHI
      COMMON/DPTN/SPRT
      DO 10 I=1,NE
      U(I)=USURF(I)
   10 X(I)=S(I)
      NS=NE
      XNS=STAGPT
      REL=REC
      ONSEP=0.D0
      OXSEP=X(NS)
      NTM=NT
      DELT=DT
      XSEP=SPPS
      SPRT=SEPRT
      OXNL=XNS
      CALL DSTAG(REL,HIL,THEIL)
      TURB=0.D0
      CALL DLAMBL(*20,DELT,HIL,THEIL,REL)
      TURB=1.D0
      CALL DTURBL(DELT,HI,THEI,REL)
      SPPS=XSEP
      SEPRT=SPRT
   20 RETURN
      END
```

```
      SUBROUTINE DECOMP
C
C THIS ROUTINE IS TO PERFORM LOWER-UPPER DECOMPOSITION
C
      IMPLICIT REAL*8 (A-H,O-Z)
      LOGICAL STEADY,USEP,DSEP
C
      COMMON/BLK0/STEADY,USEP,DSEP
      COMMON/BLK1/NSTEPS,NT,NE,ND0,ND1,ND2,ND3,MCASE,IP(40)
      COMMON/BLK2/PI,TWOPI,REY,RX,RY,THETA,SHIFT,DT,TECON
      COMMON/BLK3/PX(41),PY(41),CX(40),CY(40),S(40),SP(41),DSBD(40)
      COMMON/BLK4/ESX(41),ESY(41),ENX(40),ENY(40),ETN(40)
      COMMON/BLK5/A(40,40),B(40,40),DW(40),PERMA(40,40)
      COMMON/BLK6/UBX,UBY,OMEGA,THTO,THTMAX,FREQ,PLGMAX,RAMDA,RADIUS
      COMMON/BLK7/URX(40),URY(40),C(40),D(40),E(40),F(40)
      COMMON/BLK8/SIGMA(40),GAM(40),CP(40)
      COMMON/BLK9/TANVEL(40),EDGVEL(40),GAMVEL(40)
      COMMON/BLK10/CIRCU,STAGPT,CONST,T(2,2)
      COMMON/BLK11/NTEV,NSPV,NWMAX,NOSP,NSTAG
      COMMON/BLK12/TEVX(201),TEVY(201),TEVS(201),DSTE,TELS,TERS
      COMMON/BLK13/SPVX(201),SPVY(201),SPVS(201),DSSP
      COMMON/BLK14/TE2X,TE2Y,SP2X,SP2Y,SPPS,SPX,SPY
      COMMON/BLK15/UXTE(201),UYTE(201),UXSP(201),UYSP(201)
C
      MSIZE=NE+1
      IF(NT.GE.1) MSIZE=NE+2
C
      DO 1000 I=1,MSIZE
        IP(I)=I
 1000 CONTINUE
      DO 2000 K=1,MSIZE-1
        BIGA=0.D0
        DO 1100 I=K,MSIZE
          IF((BIGA-DABS(A(I,K))).GE.0.D0) GO TO 1100
          IPVT=I
          BIGA=DABS(A(I,K))
 1100   CONTINUE
        IF(BIGA.GT.0.D0) GO TO 10
        WRITE(6,6000)
        STOP
   10   CONTINUE
C
        DO 1200 J=1,MSIZE
          TEMP=A(K,J)
          A(K,J)=A(IPVT,J)
          A(IPVT,J)=TEMP
 1200   CONTINUE
        TEMP=IP(K)
        IP(K)=IP(IPVT)
        IP(IPVT)=TEMP
        DO 1300 I=K+1,MSIZE
          A(I,K)=A(I,K)/A(K,K)
          DO 1300 J=K+1,MSIZE
            A(I,J)=A(I,J)-A(I,K)*A(K,J)
 1300   CONTINUE
C
 2000 CONTINUE
C
      IF(A(MSIZE,MSIZE).NE.0.D0) GO TO 20
      WRITE(6,6000)
```

```
      STOP
   20 CONTINUE
C
      RETURN
C
 6000 FORMAT(///' PROGRAM EXECUTION STOPPED FOR DET(A)=0.',5X,20('?'))
C
      END
```

```
      SUBROUTINE DF(*,XN,Y,YP)
C*************************************C
      IMPLICIT REAL*8(A-H,O-Z)
      DIMENSION Y(2),YP(2)
      COMMON/DFIN/RST,FT,AT,GT,HT,XNS,XNL,XNT,TURB,NTM
      COMMON/DFOUT/VT,SH,CF2
      COMMON/DPTN/SPRT
      ONSEP=0.D0
      IF(TURB.EQ.1.D0) GO TO 35
      PG=-GT/AT
      IF(Y(1).LE.0.D0) Y(1)=1.0D-06
      Y(2)=.325D0+.130D0*Y(1)*RST*PG
      IF(Y(2).LE.0.255D0) Y(2)=0.255D0
      SH=.320D0+.922D0*Y(2)
      CALL DTRANS(*100,SH,RST,XN,XNS,XNT,Y,NTM)
      CF2=4.13D0*(.463D0-Y(2))/RST
      A=2.239D0-3.D0*SH
      B1=CF2-Y(1)*FT/(AT*AT)-(3.D0-2.D0*SH)*Y(1)*GT/AT-HT/AT
      YP(1)=B1/A
   30 CONTINUE
      GO TO 110
   35 VT=.44D0*DABS(1.D0-2.D0*Y(2))**.885D0*DABS(Y(2)/RST)**.115D0
      IF(Y(2).GT..5D0) VT=-VT
      SH=1.5D0*Y(2)+.179D0*VT+.321D0*VT*VT/Y(2)
      IF(SH.GE.0.9D0) SH=0.9D0
      CALL DTSEP(*100,SH,XN,Y,NTM)
      CF2=.1681D0*VT*DABS(VT)
      C1=1.5D0-.321D0*DABS(VT/Y(2))**2.D0+((.115D0-2.D0*Y(2))/Y(2))
     $*(.179D0+.642D0*VT/Y(2))*(.3D0+.4D0*Y(2))/DABS(RST)**.115D0
      C2=.115D0*VT*(.179D0+.642D0*VT/Y(2))
      A11=.977D0-SH+C2
      A12=-Y(1)*C1
      A22=-Y(1)/(Y(2)*(1.D0-Y(2)))
      DET=A11*A22-A12
      B1=CF2-Y(1)*FT/(AT*AT)-(2.95D0-2.D0*SH+C2)*Y(1)*GT/AT-HT/AT
      ENT=.0083D0/DABS(1.D0-Y(2))**2.5D0
      XN0=XN
      B2=ENT*Y(2)/(1.D0-Y(2))-Y(1)*GT/AT
      IF(Y(2).GT..415D0) GO TO 60
      GO TO 70
   60 IF(Y(2).LT..425D0) GO TO 80
   70 CONTINUE
      YP(1)=(B1*A22-B2*A12)/DET
      YP(2)=(B2*A11-B1)/DET
   80 CONTINUE
      GO TO 101
  100 RETURN 1
  101 CONTINUE
  110 CONTINUE
      RETURN
      END
```

```
      SUBROUTINE DGRAD(NX,DELT,DSL)
C*************************************C
      IMPLICIT REAL*8(A-H,O-Z)
      COMMON/DFIN/RST,FT,AT,GT,HT,XNS,XNL,XNT,TURB,NTM
      COMMON/DVEL/U(150),X(150),NS
      COMMON/DOLD/OAT(150),ODSL(150),OXNL,ONT,ONSEP,XSEP,OXSEP
      COMMON/DPTN/SPRT
      AT=(U(NX+1)+U(NX))/2.D0
      GT=(U(NX+1)-U(NX))/(X(NX+1)-X(NX))
      IF(NX.EQ.1) GO TO 10
      IF(NTM.EQ.1) GO TO 10
      IF(X(NX).LE.OXNL) GO TO 10
      IF(X(NX).LE.XNL) GO TO 10
      FT=(AT-OAT(NX))/DELT
      IF(X(NX).GE.OXSEP) GO TO 20
      IF(X(NX).GE.XSEP ) GO TO 20
      HT=(DSL-ODSL(NX))/DELT
      GO TO 20
   10 CONTINUE
      FT=0.D0
      HT=0.D0
   20 CONTINUE
      OAT(NX)=AT
      ODSL(NX)=DSL
      RETURN
      END
```

```
      SUBROUTINE DLAMBL(*,DELT,HI,THEI,REL)
C*****************************************************C
      IMPLICIT REAL*8(A-H,O-Z)
      COMMON/DFIN/RST,FT,AT,GT,HT,XNS,XNL,XNT,TURB,NTM
      COMMON/DFOUT/VT,SH,CF2
      COMMON/DVEL/U(150),X(150),NS
      COMMON/DOLD/OAT(150),ODSL(150),OXNL,ONT,ONSEP,XSEP,OXSEP
      COMMON/DPTN/SPRT
      DIMENSION Y(2),YP(2)
      DO 20 I=2,NS
      IF(X(I).GT.XNL) GO TO 10
      GO TO 20
   10 CONTINUE
      NXL=I-1
      GO TO 30
   20 CONTINUE
   30 CONTINUE
      UI=U(NXL)+(XNL-X(NXL))*(U(NXL+1)-U(NXL))/(X(NXL+1)-X(NXL))
      RTHI=UI*THEI*REL
      RSTI=HI*RTHI
      DSLI=HI*THEI
      DSDI=(.68D0*HI-1.D0)/(.922D0*HI)
      DLI=DSLI/DSDI
      CF2I=4.13D0*(.463D0-DSDI)/RSTI
C
      RST=RSTI
      Y(1)=DSLI
      Y(2)=DSDI
      OP=0.D0
      ONT=0.D0
      WRITE(6,40)
      NTEMP=NS-NXL+1
      WRITE(6,50) NTEMP,XNL,UI,DLI,DSLI,THEI,HI,CF2I,RTHI
      NS1=NS-1
      DO 60 NX=NXL,NS1
      X1=X(NX)
      X2=X(NX+1)
      IF(NX.EQ.NXL) X1=XNL
      CALL DGRAD(NX,DELT,Y(1))
      CALL DRKM44(*100,X1,X2,Y,1,1.0D-05)
      IF(Y(1).LE.0.D0) Y(1)=1.0D-06
      RST=U(NX+1)*Y(1)*RSTI/DSLI/UI
      CF2=4.13D0*(.463D0-Y(2))/RST
      DEL=Y(1)/Y(2)
      H=1.D0/(1.D0-SH)
      THE=Y(1)/H
      RTH=RST/H
      UE=U(NX+1)
      NXP1=NX+1
      NTEMP=NS-NXP1+1
      WRITE(6,50) NTEMP,X2,UE,DEL,Y(1),THE,H,CF2,RTH
   40 FORMAT(//1X,'NO.',7X,'X',9X,'U',8X,'DEL',7X,'DLST',6X,'THET',
     $7X,'H',8X,'CF2',7X,'RTH')
   50 FORMAT(1X,I3,4X,8E10.3)
   60 CONTINUE
      RETURN 1
  100 CONTINUE
      RETURN
      END
```

```
      SUBROUTINE DRKM44(*,T,TOUT,Y,NEQN,ABSERR)
C*************************************************C
      IMPLICIT REAL*8(A-H,O-Z)
      DIMENSION Y(150),Y1(150),Y0(150),YP(150),FE(5,150)
      COMMON/DPTN/SPRT
      H=(TOUT-T)/256.D0
      ABE=0.D0
      DT=0.D0
   17 CONTINUE
      DO 18 I=1,NEQN
   18 Y0(I)=Y(I)
      T0=T
      IF((ABE/ABSERR).GT.1.2D0)DT=.5D0*DT
      IF((ABE/ABSERR).LT.0.5D0)DT=2.D0*DT
      IF(ABE.EQ.0.D0)DT=H
      IF((T+DT).GT.TOUT)DT=TOUT-T
      CALL DF(*100,T,Y,YP)
      DO 19 I=1,NEQN
      FE(1,I)=DT*YP(I)
   19 Y(I)=Y0(I)+FE(1,I)/3.D0
      T=T0+DT/3.D0
      CALL DF(*100,T,Y,YP)
      DO 20 I=1,NEQN
      FE(2,I)=DT*YP(I)
   20 Y(I)=Y0(I)+(FE(1,I)+FE(2,I))/6.D0
      CALL DF(*100,T,Y,YP)
      DO 21 I=1,NEQN
      FE(3,I)=DT*YP(I)
   21 Y(I)=Y0(I)+(FE(1,I)+3.D0*FE(3,I))/8.D0
      T=T0+DT/2.D0
      CALL DF(*100,T,Y,YP)
      DO 22 I=1,NEQN
      FE(4,I)=DT*YP(I)
      Y(I)=Y0(I)+(FE(1,I)-3.D0*FE(3,I)+4.D0*FE(4,I))/2.D0
   22 Y1(I)=Y(I)
      T=T0+DT
      CALL DF(*100,T,Y,YP)
      ABE=0.D0
      DO 23 I=1,NEQN
      FE(5,I)=DT*YP(I)
      Y(I)=Y0(I)+(FE(1,I)+4.D0*FE(4,I)+FE(5,I))/6.D0
C<<<<<<<<< NOTE:THE FOLLOWING LIMITS Y(1) TO + VALUES>>>>>>>>>
      IF(Y(1).LE.0.D0) Y(1)=1.0D-06
   23 ABE=DMAX1(ABE,DABS(Y(I)-Y1(I)))
      IF(TOUT.NE.T)GO TO 17
      GO TO 101
  100 RETURN 1
  101 CONTINUE
      RETURN
      END
```

```
      SUBROUTINE DSTAG(REL,HI,THEI)
C*****************************************C
      IMPLICIT REAL*8(A-H,O-Z)
      COMMON/DVEL/U(150),X(150),NS
      COMMON/DFIN/RST,FT,AT,GT,HT,XNS,XNL,XNT,TURB,NTM
      COMMON/DPTN/SPRT
      DO 10 I=1,NS
      IF(X(I).GT.XNS) GO TO 20
   10 CONTINUE
   20 UI=U(I)
      XNL=X(I)
      XSL=XNL-XNS
      HI=2.216D0
      TI=.29234D0
      REXI=UI*XSL*REL
      THEI=TI*XSL/REXI**0.5D0
      RETURN
      END
```

```
      SUBROUTINE DTRANS(*,SH,RST,XN,XNS,XNT,Y,NTM)
C*********************************************************C
      IMPLICIT REAL*8(A-H,O-Z)
      COMMON/DOLD/OAT(150),ODSL(150),OXNL,ONT,ONSEP,XSEP,OXSEP
      COMMON/DBLI/HI,THEI,RTHI
      COMMON/DPTN/SPRT
      DIMENSION Y(2)
      IF(SH.GE.(1.D0-1.D0/3.9D0)) GO TO 10
      H=1.D0/(1.D0-SH)
      RTH=RST/H
      THE=Y(1)/H
      REX=RST*(XN-XNS)/Y(1)
      IF(REX.EQ.0.D0) GO TO 20
      RHS=1.174D0*(1.D0+22400.D0/REX)*DABS(REX)**0.46D0
      IF(RTH.GE.RHS) GO TO 10
      GO TO 20
   10 IF(DFLOAT(NTM).EQ.ONT) GO TO 20
      XNT=XN
      ONT=DFLOAT(NTM)
      WRITE(6,30) XNT
   30 FORMAT(4X,'THE TRANSITION POINT (XNT)=',E12.5)
      HI=H
   40 IF(HI.GT.2.76D0) HI=2.76D0
   50 CONTINUE
      THEI=THE
      RETURN 1
   20 CONTINUE
      RETURN
      END
```

```
      SUBROUTINE DTSEP(*,SH,XN,Y,NTM)
C*****************************************C
      IMPLICIT REAL*8(A-H,O-Z)
      LOGICAL SPRT
      COMMON/DOLD/OAT(150),ODSL(150),OXNL,ONT,ONSEP,XSEP,OXSEP
      COMMON/DPTN/SPRT
      DIMENSION Y(2)
      H=1.D0/(1.D0-SH)
      RHS=1.D0+1.D0/(1.D0-Y(2))
      IF(H.GE.RHS) GO TO 10
      GO TO 20
   10 IF(DFLOAT(NTM).EQ.ONSEP) GO TO 20
      OXSEP=XSEP
      XSEP =XN
      ONSEP=DFLOAT(NTM)
      SPRT = .TRUE.
      WRITE(6,30) XSEP,H,RHS
   30 FORMAT(4X,'INTERMITTENT SEP POINT (XSEP) =',E12.5,
     &       2X,'H =',E12.5,2X,'RHS =',E12.5)
      RETURN 1
   20 CONTINUE
      RETURN
      END
```

```
      SUBROUTINE DTURBL(DELT,HI,THEI,REL)
C*****************************************************C
      IMPLICIT REAL*8(A-H,O-Z)
      COMMON/DFIN/RST,FT,AT,GT,HT,XNS,XNL,XNT,TURB,NTM
      COMMON/DFOUT/VT,SH,CF2
      COMMON/DVEL/U(150),X(150),NS
      COMMON/DOLD/OAT(150),ODSL(150),OXNL,ONT,ONSEP,XSEP,OXSEP
      COMMON/DPTN/SPRT
      DIMENSION Y(2),YP(2)
      OXNT=OXT
      DO 20 I=2,NS
      IF(X(I).GT.XNT) GO TO 10
      GO TO 20
   10 CONTINUE
      NXT=I-1
      GO TO 30
   20 CONTINUE
   30 CONTINUE
      UI=U(NXT)+(XNT-X(NXT))*(U(NXT+1)-U(NXT))/(X(NXT+1)-X(NXT))
      RTHI=UI*THEI*REL
      RSTI=HI*RTHI
      DSLI=HI*THEI
      DSDI=(.9D0*HI-1.D0)/(1.3D0*HI)
      DLI=DSLI/DSDI
      CF2I=0.051D0*DABS(1.D0-2.D0*DSDI)**1.732D0/DABS(RSTI/DSDI)
     $**.268D0*DABS(1.D0-2.D0*DSDI)/(1.D0-2.D0*DSDI)
      RST=RSTI
      Y(1)=DSLI
      Y(2)=DSDI
      NTEMP=NS-NXT+1
      WRITE(6,50) NTEMP,XNT,UI,DLI,DSLI,THEI,HI,CF2I,RTHI
      NS1=NS-1
      DO 60 NX=NXT,NS1
      X1=X(NX)
      X2=X(NX+1)
      IF(NX.EQ.NXT) X1=XNT
      CALL DGRAD(NX,DELT,Y(1))
      CALL DRKM44(*100,X1,X2,Y,2,1.0D-05)
      RST=U(NX+1)*Y(1)*RSTI/DSLI/UI
      CF2=.1681D0*VT*DABS(VT)
      DEL=Y(1)/Y(2)
      H=1.D0/(1.D0-SH)
      THE=Y(1)/H
      RTH=RST/H
      UE=U(NX+1)
      NXP1=NX+1
      NTEMP=NS-NXP1+1
      WRITE(6,50) NTEMP,X2,UE,DEL,Y(1),THE,H,CF2,RTH
   50 FORMAT(1X,I3,4X,8E10.3)
   60 CONTINUE
  100 CONTINUE
      RETURN
      END
```

```
      SUBROUTINE DWASH
C
C THIS ROUTINE IS TO COMPUTE THE DOWN WASH DUE TO RELATIVE VELOCITY OF
C THE FREE STREAM AND SOURCE DISTRIBUTION ON THE SURFACE.
C
      IMPLICIT REAL*8 (A-H,O-Z)
      LOGICAL STEADY,USEP,DSEP
C
      COMMON/BLK0/STEADY,USEP,DSEP
      COMMON/BLK1/NSTEPS,NT,NE,ND0,ND1,ND2,ND3,MCASE,IP(40)
      COMMON/BLK2/PI,TWOPI,REY,RX,RY,THETA,SHIFT,DT,TECON
      COMMON/BLK3/PX(41),PY(41),CX(40),CY(40),S(40),SP(41),DSBD(40)
      COMMON/BLK4/ESX(41),ESY(41),ENX(40),ENY(40),ETN(40)
      COMMON/BLK5/A(40,40),B(40,40),DW(40),PERMA(40,40)
      COMMON/BLK6/UBX,UBY,OMEGA,THT0,THTMAX,FREQ,PLGMAX,RAMDA,RADIUS
      COMMON/BLK7/URX(40),URY(40),C(40),D(40),E(40),F(40)
      COMMON/BLK8/SIGMA(40),GAM(40),CP(40)
      COMMON/BLK9/TANVEL(40),EDGVEL(40),GAMVEL(40)
      COMMON/BLK10/CIRCU,STAGPT,CONST,T(2,2)
      COMMON/BLK11/NTEV,NSPV,NWMAX,NOSP,NSTAG
      COMMON/BLK12/TEVX(201),TEVY(201),TEVS(201),DSTE,TELS,TERS
      COMMON/BLK13/SPVX(201),SPVY(201),SPVS(201),DSSP
      COMMON/BLK14/TE2X,TE2Y,SP2X,SP2Y,SPPS,SPX,SPY
      COMMON/BLK15/UXTE(201),UYTE(201),UXSP(201),UYSP(201)
C
      MSIZE=NE+1
      IF(NT.GE.1) MSIZE=NE+2
C
C RELATIVE TRANSLATIONAL VELOCITY WRT BFF
C
      UX=T(1,1)*(1.D0-UBX)-T(1,2)*UBY
      UY=T(2,1)*(1.D0-UBX)-T(2,2)*UBY
C
C ADD THE VELOCITY DUE TO THE BODY ROTATION (BFF)
C
      DO 1000 I=1,NE
        URX(I)=UX+OMEGA*CY(I)
        URY(I)=UY-OMEGA*CX(I)
 1000 CONTINUE
      URX(NE+2)=UX
      URY(NE+2)=UY
C
C DOWN WASH DUE TO THE RELATIVE VELOCITY OF THE STREAM WRT ELEMENT'S
C MOTION
C
      DO 1020 I=1,MSIZE
        IF(I.EQ.NE+1) GO TO 1020
        D(I)=URX(I)*ENX(I)+URY(I)*ENY(I)
 1020 CONTINUE
C
C GET THE SOURCE DISTRIBUTION WHICH NEUTRALIZE THE NORMAL VELOCITY ON
C EACH ELEMENT
C
      DO 1040 I=1,NE
        TANVEL(I)=URX(I)*ESX(I)+URY(I)*ESY(I)
        SIGMA(I)=-D(I)
 1040 CONTINUE
C
C GET THE DOWN WASH FROM SOURCE DISTRIBUTION
C
```

```
      DO 1060 I=1,MSIZE
        C(I)=0.D0
        DO 1060 J=1,NE
          C(I)=C(I)+B(I,J)*SIGMA(J)
 1060 CONTINUE
C
      RETURN
C
      END
```

```
      SUBROUTINE F(*,XN,Y,YP)
C*************************************C
      IMPLICIT REAL*8(A-H,O-Z)
      DIMENSION Y(2),YP(2)
      COMMON/FIN/RST,FT,AT,GT,HT,XNS,XNL,XNT,TURB,NTM
      COMMON/FOUT/VT,SH,CF2
      COMMON/PTN/SPRT
      ONSEP=0.D0
      IF(TURB.EQ.1.D0) GO TO 35
      PG=-GT/AT
      IF(Y(1).LE.0.D0) Y(1)=1.0D-06
      Y(2)=.325D0+.130D0*Y(1)*RST*PG
      IF(Y(2).LE.0.255D0) Y(2)=0.255D0
      SH=.320D0+.922D0*Y(2)
      CALL TRANS(*100,SH,RST,XN,XNS,XNT,Y,NTM)
      CF2=4.13D0*(.463D0-Y(2))/RST
      A=2.239D0-3.D0*SH
      B1=CF2-Y(1)*FT/(AT*AT)-(3.D0-2.D0*SH)*Y(1)*GT/AT-HT/AT
      YP(1)=B1/A
   30 CONTINUE
      GO TO 110
   35 VT=.44D0*DABS(1.D0-2.D0*Y(2))**.885D0*DABS(Y(2)/RST)**.115D0
      IF(Y(2).GT..5D0) VT=-VT
      SH=1.5D0*Y(2)+.179D0*VT+.321D0*VT*VT/Y(2)
      IF(SH.GE.0.9D0) SH=0.9D0
      CALL TSEP(*100,SH,XN,Y,NTM)
      CF2=.1681D0*VT*DABS(VT)
      C1=1.5D0-.321D0*DABS(VT/Y(2))**2.D0+((.115D0-2.D0*Y(2))/Y(2))
     $*(.179D0+.642D0*VT/Y(2))*(.3D0+.4D0*Y(2))/DABS(RST)**.115D0
      C2=.115D0*VT*(.179D0+.642D0*VT/Y(2))
      A11=.977D0-SH+C2
      A12=-Y(1)*C1
      A22=-Y(1)/(Y(2)*(1.D0-Y(2)))
      DET=A11*A22-A12
      B1=CF2-Y(1)*FT/(AT*AT)-(2.95D0-2.D0*SH+C2)*Y(1)*GT/AT-HT/AT
      ENT=.0083D0/DABS(1.D0-Y(2))**2.5D0
      XNO=XN
      B2=ENT*Y(2)/(1.D0-Y(2))-Y(1)*GT/AT
      IF(Y(2).GT..415D0) GO TO 60
      GO TO 70
   60 IF(Y(2).LT..425D0) GO TO 80
   70 CONTINUE
      YP(1)=(B1*A22-B2*A12)/DET
      YP(2)=(B2*A11-B1)/DET
   80 CONTINUE
      GO TO 101
  100 RETURN 1
  101 CONTINUE
  110 CONTINUE
      RETURN
      END
```

```
      SUBROUTINE GEOM
C
C THIS ROUTINE IS TO COMPUTE THE GEOMETRIC VALUES FOR GIVEN AIRFOIL
C SHAPE
C
      IMPLICIT REAL*8 (A-H,O-Z)
C
      COMMON/BLK0/STEADY,USEP,DSEP
      COMMON/BLK1/NSTEPS,NT,NE,ND0,ND1,ND2,ND3,MCASE,IP(40)
      COMMON/BLK2/PI,TWOPI,REY,RX,RY,THETA,SHIFT,DT,TECON
      COMMON/BLK3/PX(41),PY(41),CX(40),CY(40),S(40),SP(41),DSBD(40)
      COMMON/BLK4/ESX(41),ESY(41),ENX(40),ENY(40),ETN(40)
      COMMON/BLK5/A(40,40),B(40,40),DW(40),PERMA(40,40)
      COMMON/BLK6/UBX,UBY,OMEGA,THT0,THTMAX,FREQ,PLGMAX,RAMDA,RADIUS
      COMMON/BLK7/URX(40),URY(40),C(40),D(40),E(40),F(40)
      COMMON/BLK8/SIGMA(40),GAM(40),CP(40)
      COMMON/BLK9/TANVEL(40),EDGVEL(40),GAMVEL(40)
      COMMON/BLK10/CIRCU,STAGPT,CONST,T(2,2)
      COMMON/BLK11/NTEV,NSPV,NWMAX,NOSP,NSTAG
      COMMON/BLK12/TEVX(201),TEVY(201),TEVS(201),DSTE,TELS,TERS
      COMMON/BLK13/SPVX(201),SPVY(201),SPVS(201),DSSP
      COMMON/BLK14/TE2X,TE2Y,SP2X,SP2Y,SPPS,SPX,SPY
      COMMON/BLK15/UXTE(201),UYTE(201),UXSP(201),UYSP(201)
C
      TEMPSP=0.D0
      SP(1)=0.D0
      TECON=TECON*.01D0
C
      DO 1000 K=1,NE
C
C GET THE CENTROID LOCATION OF ELEMENT WRT BBF
C
        CX(K)=.5D0*(PX(K)+PX(K+1))
        CY(K)=.5D0*(PY(K)+PY(K+1))
        XDIST=PX(K+1)-PX(K)
        YDIST=PY(K+1)-PY(K)
        TEMP=DSQRT(XDIST*XDIST+YDIST*YDIST)
C
C GET EACH ELEMENT SIZE
C
        DSBD(K)=TEMP
        TEMPSP=TEMPSP+TEMP
        TEMPS=TEMPSP-.5D0*TEMP
C
C GET THE TANGENTIAL UNIT VECTOR OF EACH ELEMENT
C
        ESX(K)=XDIST/TEMP
        ESY(K)=YDIST/TEMP
C
C GET THE TANGENT ANGLE WRT B.F.F.
C
        ETN(K)=DATAN2(ESY(K),ESX(K))
C
C GET THE NORMAL UNIT VECTOR
C
        ENX(K)=-ESY(K)
        ENY(K)= ESX(K)
C
C GET THE SURFACE DISTANCE FOR CONTROL POINTS
C
```

```
        S(K)=TEMPS
C
C GET THE SURFACE DISTANCE FOR BOUNDARY POINTS
C
        SP(K+1)=TEMPSP
 1000 CONTINUE
C
C GET THE TRAILING EDGE BISECTOR
C
      BISX=.5D0*(ESX(NE)-ESX(1))
      BISY=.5D0*(ESY(NE)-ESY(1))
      TEMP1=DSQRT(BISX*BISX+BISY*BISY)
      ESX(NE+2)=BISX/TEMP1
      ESY(NE+2)=BISY/TEMP1
      ENX(NE+2)=-ESY(NE+2)
      ENY(NE+2)= ESX(NE+2)
      CX(NE+2)=PX(1)+ESX(NE+2)*TECON
      CY(NE+2)=PY(1)+ESY(NE+2)*TECON
C
      WRITE(6,6000)
      WRITE(6,6010) (K,PX(K),PY(K),SP(K),K=1,NE+1)
      WRITE(6,6020)
      WRITE(6,6030) (K,CX(K),CY(K),S(K),ENX(K),ENY(K),ESX(K),
     &               ESY(K),K=1,NE)
      WRITE(6,6030) (K,CX(K),CY(K),S(K),ENX(K),ENY(K),ESX(K),
     &               ESY(K),K=NE+2,NE+2)
C
      RETURN
C
 6000 FORMAT(///16X,'ENDPOINTS'/13X,'PX',11X,'PY',11X,'SP')
 6010 FORMAT(1X,I5,2X,F10.4,2X,F10.4,2X,F10.5)
 6020 FORMAT(///10X,'CX',7X,'CY',10X,'S',9X,'ENX',6X,'ENY',8X,'ESX',
     &       6X,'ESY')
 6030 FORMAT((1X,I5,2X,2(F7.4,2X),2X,F7.4,4X,2(F7.4,2X),2X,2(F7.4,2X)))
      END
```

```
      SUBROUTINE GETVEL(POINTX,POINTY,VELX,VELY)
C
C THIS ROUTINE IS TO COMPUTE THE INDUCED NORMAL & TANGENTIAL VELOCITY
C DUE TO THE SOURCE & VORTEX DISTRIBUTIONS ON THE BODY & THE WAKE AT
C THE GIVEN POINT.  IT EXCLUDES THE INFLUENCE OF FREE STREAM AND THE
C MOTION OF BODY.
C
      IMPLICIT REAL*8 (A-H,O-Z)
      LOGICAL STEADY,USEP,DSEP
C
      COMMON/BLK0/STEADY,USEP,DSEP
      COMMON/BLK1/NSTEPS,NT,NE,ND0,ND1,ND2,ND3,MCASE,IP(40)
      COMMON/BLK2/PI,TWOPI,REY,RX,RY,THETA,SHIFT,DT,TECON
      COMMON/BLK3/PX(41),PY(41),CX(40),CY(40),S(40),SP(41),DSBD(40)
      COMMON/BLK4/ESX(41),ESY(41),ENX(40),ENY(40),ETN(40)
      COMMON/BLK5/A(40,40),B(40,40),DW(40),PERMA(40,40)
      COMMON/BLK6/UBX,UBY,OMEGA,THT0,THTMAX,FREQ,PLGMAX,RAMDA,RADIUS
      COMMON/BLK7/URX(40),URY(40),C(40),D(40),E(40),F(40)
      COMMON/BLK8/SIGMA(40),GAM(40),CP(40)
      COMMON/BLK9/TANVEL(40),EDGVEL(40),GAMVEL(40)
      COMMON/BLK10/CIRCU,STAGPT,CONST,T(2,2)
      COMMON/BLK11/NTEV,NSPV,NWMAX,NOSP,NSTAG
      COMMON/BLK12/TEVX(201),TEVY(201),TEVS(201),DSTE,TELS,TERS
      COMMON/BLK13/SPVX(201),SPVY(201),SPVS(201),DSSP
      COMMON/BLK14/TE2X,TE2Y,SP2X,SP2Y,SPPS,SPX,SPY
      COMMON/BLK15/UXTE(201),UYTE(201),UXSP(201),UYSP(201)
C
C INITIALIZE THE ARRAY
C
      VELX=0.D0
      VELY=0.D0
C
C-----------------------------------------------------------------
C INDUCED VELOCITY DUE TO THE SOURCE DISTRIBUTION
C
      AVELX=0.D0
      AVELY=0.D0
      DO 1000 J=1,NE
        CALL SOURCE(CX(J),CY(J),POINTX,POINTY,ESX(J),ESY(J),DSBD(J),
     &              VS1X,VS1Y,VSLX,VSLY,VSRX,VSRY)
        TEMP1=SIGMA(J)
        IF(J.EQ.1) TEMP2=2.D0*SIGMA(1)/DSBD(1)
        IF(J.NE.1) TEMP2=2.D0*(SIGMA(J)-SIGMA(J-1))/(DSBD(J)+DSBD(J-1))
        IF(J.EQ.NE) TEMP3=-2.D0*SIGMA(NE)/DSBD(NE)
        IF(J.NE.NE) TEMP3=2.D0*(SIGMA(J+1)-SIGMA(J))/(DSBD(J+1)+
     &                    DSBD(J))
        TEMP00=TEMP1*VS1X+TEMP2*VSLX+TEMP3*VSRX
        TEMP11=TEMP1*VS1Y+TEMP2*VSLY+TEMP3*VSRY
        AVELX=AVELX+TEMP00
        AVELY=AVELY+TEMP11
 1000 CONTINUE
C
      VELX=VELX+AVELX
      VELY=VELY+AVELY
C-----------------------------------------------------------------
C INDUCED VELOCITY DUE TO THE VORTEX DISTRIBUTION ON THE BODY
C
      AVELX=0.D0
      AVELY=0.D0
      DO 2000 J=1,NE
```

```
         CALL VORTEX(CX(J),CY(J),POINTX,POINTY,ESX(J),ESY(J),DSBD(J),
     &               VV1X,VV1Y,VV2X,VV2Y)
         TEMP1=.5D0*(GAM(J)+GAM(J+1))
         TEMP2=(GAM(J+1)-GAM(J))/DSBD(J)
         TEMP33=TEMP1*VV1X+TEMP2*VV2X
         TEMP44=TEMP1*VV1Y+TEMP2*VV2Y
         AVELX=AVELX+TEMP33
         AVELY=AVELY+TEMP44
 2000 CONTINUE
C
      VELX=VELX+AVELX
      VELY=VELY+AVELY
C-----------------------------------------------------------------------
C INDUCED VELOCITY DUE TO THE VORTEX DISTRIBUTION ON TE WAKE
C
      IF(NTEV.LE.0) GO TO 4000
      DISTX=TE2X-TEVX(1)
      DISTY=TE2Y-TEVY(1)
      TEMESX=DISTX/DSTE
      TEMESY=DISTY/DSTE
      CTEVX=.5D0*(TE2X+TEVX(1))
      CTEVY=.5D0*(TE2Y+TEVY(1))
      TEMP1=.5D0*(TELS+TERS)
      TEMP2=(TERS-TELS)/DSTE
      TEMPX=POINTX*T(1,1)+POINTY*T(2,1)
      TEMPY=POINTX*T(1,2)+POINTY*T(2,2)
      CALL VORTEX(CTEVX,CTEVY,TEMPX,TEMPY,TEMESX,TEMESY,
     &            DSTE,VV1X,VV1Y,VV2X,VV2Y)
      UWTEX=VV1X*TEMP1+VV2X*TEMP2
      UWTEY=VV1Y*TEMP1+VV2Y*TEMP2
      AVELX=UWTEX*T(1,1)+UWTEY*T(1,2)
      AVELY=UWTEX*T(2,1)+UWTEY*T(2,2)
C
      VELX=VELX+AVELX
      VELY=VELY+AVELY
C-----------------------------------------------------------------------
      IF(NTEV.LE.1) GO TO 4000
      UWTEX=0.D0
      UWTEY=0.D0
      DO 3000 I=3,NTEV+1
         CALL LMPVTX(2,I,TEVX(I),TEVY(I),TEMPX,TEMPY,VLX,VLY)
         UWTEX=UWTEX+VLX*TEVS(I)
         UWTEY=UWTEY+VLY*TEVS(I)
 3000 CONTINUE
      AVELX=UWTEX*T(1,1)+UWTEY*T(1,2)
      AVELY=UWTEX*T(2,1)+UWTEY*T(2,2)
C
      VELX=VELX+AVELX
      VELY=VELY+AVELY
 4000 CONTINUE
C-----------------------------------------------------------------------
C INDUCED VELOCITY DUE TO THE VORTEX DISTRIBUTION ON SP WAKE
C
      IF(NSPV.LT.1) GO TO 8000
      UWSPX=0.D0
      UWSPY=0.D0
      DO 7000 I=2,NSPV+1
         CALL LMPVTX(2,I,SPVX(I),SPVY(I),TEMPX,TEMPY,VLX,VLY)
         UWSPX=UWSPX+VLX*SPVS(I)
         UWSPY=UWSPY+VLY*SPVS(I)
```

```
 7000 CONTINUE
      AVELX=UWSPX*T(1,1)+UWSPY*T(1,2)
      AVELY=UWSPX*T(2,1)+UWSPY*T(2,2)
C
      VELX=VELX+AVELX
      VELY=VELY+AVELY
 8000 CONTINUE
C
C
      RETURN
C
      END
```

```
      SUBROUTINE GRAD(NX,DELT,DSL)
C*************************************C
      IMPLICIT REAL*8(A-H,O-Z)
      COMMON/FIN/RST,FT,AT,GT,HT,XNS,XNL,XNT,TURB,NTM
      COMMON/VEL/U(150),X(150),NS
      COMMON/OLD/OAT(150),ODSL(150),OXNL,ONT,ONSEP,XSEP,OXSEP
      COMMON/PTN/SPRT
      AT=(U(NX+1)+U(NX))/2.D0
      GT=(U(NX+1)-U(NX))/(X(NX+1)-X(NX))
      IF(NX.EQ.1) GO TO 10
      IF(NTM.EQ.1) GO TO 10
      IF(X(NX).LE.OXNL) GO TO 10
      IF(X(NX).LE.XNL) GO TO 10
      FT=(AT-OAT(NX))/DELT
      IF(X(NX).GE.OXSEP) GO TO 20
      IF(X(NX).GE.XSEP ) GO TO 20
      HT=(DSL-ODSL(NX))/DELT
      GO TO 20
   10 CONTINUE
      FT=0.D0
      HT=0.D0
   20 CONTINUE
      OAT(NX)=AT
      ODSL(NX)=DSL
      RETURN
      END
```

```
      SUBROUTINE INPUT
C
C  THIS ROUTINE IS TO READ THE DATA FROM DATA CARDS
C
      IMPLICIT REAL*8 (A-H,O-Z)
      REAL*8 XSTA(41),PXTEM(80),PYTEM(80)
      LOGICAL STEADY,USEP,DSEP
      CHARACTER A1,A2,A3,A4
      CHARACTER*4 BODY
C
      COMMON/BLK0/STEADY,USEP,DSEP
      COMMON/BLK1/NSTEPS,NT,NE,ND0,ND1,ND2,ND3,MCASE,IP(40)
      COMMON/BLK2/PI,TWOPI,REY,RX,RY,THETA,SHIFT,DT,TECON
      COMMON/BLK3/PX(41),PY(41),CX(40),CY(40),S(40),SP(41),DSBD(40)
      COMMON/BLK4/ESX(41),ESY(41),ENX(40),ENY(40),ETN(40)
      COMMON/BLK5/A(40,40),B(40,40),DW(40),PERMA(40,40)
      COMMON/BLK6/UBX,UBY,OMEGA,THTO,THTMAX,FREQ,PLGMAX,RAMDA,RADIUS
      COMMON/BLK7/URX(40),URY(40),C(40),D(40),E(40),F(40)
      COMMON/BLK8/SIGMA(40),GAM(40),CP(40)
      COMMON/BLK9/TANVEL(40),EDGVEL(40),GAMVEL(40)
      COMMON/BLK10/CIRCU,STAGPT,CONST,T(2,2)
      COMMON/BLK11/NTEV,NSPV,NWMAX,NOSP,NSTAG
      COMMON/BLK12/TEVX(201),TEVY(201),TEVS(201),DSTE,TELS,TERS
      COMMON/BLK13/SPVX(201),SPVY(201),SPVS(201),DSSP
      COMMON/BLK14/TE2X,TE2Y,SP2X,SP2Y,SPPS,SPX,SPY
      COMMON/BLK15/UXTE(201),UYTE(201),UXSP(201),UYSP(201)
C
      PI=3.14159265358979267
      TWOPI=2.D0*PI
      USEP=.FALSE.
      DSEP=.FALSE.
      NT=0
      NTEV=0
      NSPV=0
C
C PRINT THE PROGRAM TITLE
C
      WRITE(6,6000)
C
C READ AND PRINT THE JOB TITLE
C
      READ(5,5000)
      WRITE(6,5000)
C
C READ DATA
C
      READ(5,5010) NSTEPS,DT,NWMAX
      READ(5,5020) REY,STEADY,TECON
      READ(5,5030) NE,RX,RY,THETA,A1,SHIFT
      READ(5,5040) BODY,ND0,ND1,ND2,ND3
      READ(5,5070) MCASE,THTO,A2,OMEGA,A3,THTMAX,A4,PLGMAX,FREQ
      READ(5,5080) RADIUS,RAMDA
C
      THETA=-THETA
      THTO=-THTO
      OMEGA=-OMEGA
      THTMAX=-THTMAX
      SHIFT=-SHIFT
C
C CONVERT THE DEGREES TO RADIENS IF IT'S IN DEGREES.
```

```
C
      CFACTR=PI/180.D0
      IF(A1.EQ.'D') THETA=THETA*CFACTR
      IF(A2.EQ.'D') THTO=THTO*CFACTR
      IF(A3.EQ.'D') OMEGA=OMEGA*CFACTR
      IF(A4.EQ.'D') THTMAX=THTMAX*CFACTR
C
C GET THE SURFACE GEOMETRY FOR NACA SERIES AIRFOIL
C
      IF(BODY.NE.'NACA') GO TO 110
  100 CONTINUE
C
C READ X STATIONS WRT BFF
C
      READ(5,5050) (XSTA(K),K=1,NE/2+1)
      DO 1000 K=1,NE/2+1
        KK=NE+2-K
C
C CALL NACA DESCRIPTER FOR SURFACE NODE LOCATIONS
C
        CALL NACAGM(XSTA(K),PY(KK),ND0,ND1,ND2,ND3,SHIFT)
        PX(K)=XSTA(K)
        PX(KK)=PX(K)
        PY(K)=-PY(KK)
 1000 CONTINUE
      PY(1)=0.D0
      PY(NE+1)=0.D0
      GO TO 140
C
C GET THE SURFACE GEOMETRY FOR CYLINDER
C
  110 CONTINUE
      IF(BODY.NE.'CYLN') GO TO 130
  120 CONTINUE
      DO 1010 K=1,NE
        TEMP=-DBLE(K-1)*TWOPI/DBLE(NE)
        PX(K)=.5D0*DCOS(TEMP)+SHIFT
        PY(K)=.5D0*DSIN(TEMP)
 1010 CONTINUE
      PX(NE+1)=PX(1)
      PY(NE+1)=PY(1)
      GO TO 140
  130 CONTINUE
C
C  GET THE SURFACE GEOMETRY FOR LAMINAR AIRFOIL
C
      READ(5,5050) (XSTA(K),K=1,NE/2+1)
      NETEML=29
      NETEMU=32
      READ(5,7000)(PXTEM(I),PYTEM(I),I=1,NETEML+NETEMU+1)
 7000 FORMAT(2F10.5)
      DO 8000 I=1,NE/2+1
        DO 8002 J=1,NETEML+1
          IF(XSTA(I).GE.PXTEM(J)) GO TO 8100
 8002   CONTINUE
 8100   PX(I)=XSTA(I)
        PPTEM=(XSTA(I)-PXTEM(J-1))/(PXTEM(J)-PXTEM(J-1))
        PY(I)=PYTEM(J-1)+(PYTEM(J)-PYTEM(J-1))*PPTEM
 8000 CONTINUE
      DO 8300 I=2,NE/2
```

```
        DO 8004 J=NETEML+NETEMU+1,NETEML+1,-1
          IF(XSTA(I).GE.PXTEM(J)) GO TO 8102
 8004   CONTINUE
 8102   PX(NE+2-I)=XSTA(I)
        PPTEM=(XSTA(I)-PXTEM(J))/(PXTEM(J+1)-PXTEM(J))
        PY(NE+2-I)=PYTEM(J)+(PYTEM(J+1)-PYTEM(J))*PPTEM
 8300 CONTINUE
      PX(NE+1)=PX(1)
      PY(NE+1)=PY(1)
  140 CONTINUE
C
C PRINT THE INPUT DATA
C
      WRITE(6,6010) NSTEPS,DT
      WRITE(6,6020) REY,STEADY,TECON
      WRITE(6,6030) NE,RX,RY,THETA,SHIFT
      WRITE(6,6040) BODY,ND0,ND1,ND2,ND3
      WRITE(6,6050) MCASE,THT0,OMEGA,THTMAX,PLGMAX,FREQ,RADIUS,RAMDA
C
      RETURN
C
 6000 FORMAT('1',5(/),18X,'UNSTEADY VISCOUS AERODYNAMIC MODEL',3(/),
     &       25X,'TEXAS TECH UNIVERSITY'//)
 5000 FORMAT(50H                                                  )
 5010 FORMAT(I5,F10.4,I5)
 5020 FORMAT(E11.3,1X,L1,F10.4)
 5030 FORMAT(I3,2F8.5,F7.5,A1,F8.5)
 5040 FORMAT(A4,1X,4I1)
 5050 FORMAT(F8.5)
 5070 FORMAT(I5,F9.7,A1,F9.7,A1/F9.7,A1,F10.7,F10.7)
 5080 FORMAT(2F10.4)
 6010 FORMAT(////' NSTEPS =',I5/' DT =',F10.5)
 6020 FORMAT(' REY =',E11.3/' STEADY = ',L1/' TE CONTROL PT = ',F10.4,
     &       ' % CHORD')
 6030 FORMAT(' NE =',I5/' RX =',F10.5/' RY =',F10.5/' THETA =',
     &       F10.5/' SHIFT =',F10.5)
 6040 FORMAT(' BODY =',A5,2X,4I1)
 6050 FORMAT(' MCASE =',I5/' THT0 =',F10.5/' OMEGA =',F10.5/
     &       ' THTMAX =',F10.5/' PLGMAX =',F10.7/' FREQ =',F10.7/
     &       ' RADIUS =',F10.4/' RAMDA =',F10.4)
C
      END
```

```
      SUBROUTINE LAMBL(*,DELT,HI,THEI,REL)
C*****************************************************C
      IMPLICIT REAL*8(A-H,O-Z)
      COMMON/FIN/RST,FT,AT,GT,HT,XNS,XNL,XNT,TURB,NTM
      COMMON/FOUT/VT,SH,CF2
      COMMON/VEL/U(150),X(150),NS
      COMMON/OLD/OAT(150),ODSL(150),OXNL,ONT,ONSEP,XSEP,OXSEP
      COMMON/PTN/SPRT
      DIMENSION Y(2),YP(2)
      DO 20 I=2,NS
      IF(X(I).GT.XNL) GO TO 10
      GO TO 20
   10 CONTINUE
      NXL=I-1
      GO TO 30
   20 CONTINUE
   30 CONTINUE
      UI=U(NXL)+(XNL-X(NXL))*(U(NXL+1)-U(NXL))/(X(NXL+1)-X(NXL))
      RTHI=UI*THEI*REL
      RSTI=HI*RTHI
      DSLI=HI*THEI
      DSDI=(.68D0*HI-1.D0)/(.922D0*HI)
      DLI=DSLI/DSDI
      CF2I=4.13D0*(.463D0-DSDI)/RSTI
C
      RST=RSTI
      Y(1)=DSLI
      Y(2)=DSDI
      OP=0.D0
      ONT=0.D0
      WRITE(6,40)
      WRITE(6,50) NXL,XNL,UI,DLI,DSLI,THEI,HI,CF2I,RTHI
      NS1=NS-1
      DO 60 NX=NXL,NS1
      X1=X(NX)
      X2=X(NX+1)
      IF(NX.EQ.NXL) X1=XNL
      CALL GRAD(NX,DELT,Y(1))
      CALL RKM44(*100,X1,X2,Y,1,1.0D-05)
      IF(Y(1).LE.0.D0) Y(1)=1.0D-06
      RST=U(NX+1)*Y(1)*RSTI/DSLI/UI
      CF2=4.13D0*(.463D0-Y(2))/RST
      DEL=Y(1)/Y(2)
      H=1.D0/(1.D0-SH)
      THE=Y(1)/H
      RTH=RST/H
      UE=U(NX+1)
      NXP1=NX+1
      WRITE(6,50) NXP1,X2,UE,DEL,Y(1),THE,H,CF2,RTH
   40 FORMAT(//1X,'NO.',7X,'X',9X,'U',8X,'DEL',7X,'DLST',6X,'THET',
     $7X,'H',8X,'CF2',7X,'RTH')
   50 FORMAT(1X,I3,4X,8E10.3)
   60 CONTINUE
      RETURN 1
  100 CONTINUE
      RETURN
      END
```

```
      SUBROUTINE LMPVTX(KASE,LAGE,XS,YS,XF,YF,VELX,VELY)
C
C THIS ROUTINE COMPUTES INDUCED VELOCITY FROM A LUMPED VORTEX
C
      IMPLICIT REAL*8 (A-H,O-Z)
C
      TWOPI=2.D0*3.14159265358979267
      VELX=0.D0
      VELY=0.D0
      DISTX=XF-XS
      DISTY=YF-YS
      DIST2=DISTX*DISTX+DISTY*DISTY
      IF(DIST2.LT.1.D-06) RETURN
      DIST=DSQRT(DIST2)
      RCORE=.07D0*DSQRT(DFLOAT(LAGE-2))
C
      GO TO (100,200,300),KASE
C
  100 VEL=1.D0/(TWOPI*DIST2)
      GO TO 900
C
  200 IF(DIST.GE.RCORE) GO TO 220
      VEL=1.D0/(TWOPI*RCORE*RCORE)
      GO TO 900
  220 VEL=1.D0/(TWOPI*DIST2)
      GO TO 900
C
  300 TEMP=-DIST2/(RCORE*RCORE)
      EGIPT=0.D0
      IF(TEMP.GT.-40.D0) EGIPT=2.718281828D0**TEMP
      VEL=(1.D0-EGIPT)/(TWOPI*DIST2)
      GO TO 900
C
  900 VELX=-VEL*DISTY
      VELY= VEL*DISTX
C
      RETURN
      END
```

```
// JOB (BON$LJ,7056,9,,100,,,1),'ALUMP-OSCIL',REGION=1024K
// EXEC FORTVCLG
//FORT.SYSIN DD *
C
      IMPLICIT REAL*8 (A-H,O-Z)
      LOGICAL STEADY,USEP,DSEP,SPRT
C
      COMMON/BLK0/STEADY,USEP,DSEP
      COMMON/BLK1/NSTEPS,NT,NE,ND0,ND1,ND2,ND3,MCASE,IP(40)
      COMMON/BLK2/PI,TWOPI,REY,RX,RY,THETA,SHIFT,DT,TECON
      COMMON/BLK3/PX(41),PY(41),CX(40),CY(40),S(40),SP(41),DSBD(40)
      COMMON/BLK4/ESX(41),ESY(41),ENX(40),ENY(40),ETN(40)
      COMMON/BLK5/A(40,40),B(40,40),DW(40),PERMA(40,40)
      COMMON/BLK6/UBX,UBY,OMEGA,THT0,THTMAX,FREQ,PLGMAX,RAMDA,RADIUS
      COMMON/BLK7/URX(40),URY(40),C(40),D(40),E(40),F(40)
      COMMON/BLK8/SIGMA(40),GAM(40),CP(40)
      COMMON/BLK9/TANVEL(40),EDGVEL(40),GAMVEL(40)
      COMMON/BLK10/CIRCU,STAGPT,CONST,T(2,2)
      COMMON/BLK11/NTEV,NSPV,NWMAX,NOSP,NSTAG
      COMMON/BLK12/TEVX(201),TEVY(201),TEVS(201),DSTE,TELS,TERS
      COMMON/BLK13/SPVX(201),SPVY(201),SPVS(201),DSSP
      COMMON/BLK14/TE2X,TE2Y,SP2X,SP2Y,SPPS,SPX,SPY
      COMMON/BLK15/UXTE(201),UYTE(201),UXSP(201),UYSP(201)
C
C   **********  MAIN PROGRAM  **********
C
      CALL INPUT
C
      CALL GEOM
C
      CALL COEFA
C
      CALL DECOMP
C
      CALL COEFB
C
      NT=0
   10 CONTINUE
C
      CALL CPUTM(0,ITM)
      TIME=DBLE(NT)*DT
      WRITE(6,6000) NT,TIME
C
      CALL MOTION
C
      IF(NT.EQ.0) GO TO 100
C
      CALL WAKVEL
C
      CALL TEGEOM
C
      CALL SPGEOM
C
      CALL WAKINF
C
      CALL DECOMP
C
  100 CONTINUE
C
      CALL DWASH
```

```
C
      CALL SOLVE
C
      CALL SEPCAL
C
      CALL CPDIST
C
      CALL CPUTM(1,ITM)
      SEC=ITM*0.01D0
      WRITE(6,6100) SEC
      NT=NT+1
      IF(STEADY.OR.NT.GT.NSTEPS) STOP
      GO TO 10
C
 6000 FORMAT(////
     2 '1',9X,50H**************************************************
     3/,10X,17H************ STEP,I3,10H - TIME = ,F8.5,12H ***********

     4/,10X,50H************************************************** //)
 6100 FORMAT(///30('-'),' CPU TIME FOR THIS STEP = ',F5.2,' SEC.'///)
C
      END
```

```
      SUBROUTINE MOTION
C
C THIS ROUTINE IS USED TO CALCULATE THE MOTION OF BBF WRT IFF
C
      IMPLICIT REAL*8 (A-H,O-Z)
C
      COMMON/BLK0/STEADY,USEP,DSEP
      COMMON/BLK1/NSTEPS,NT,NE,ND0,ND1,ND2,ND3,MCASE,IP(40)
      COMMON/BLK2/PI,TWOPI,REY,RX,RY,THETA,SHIFT,DT,TECON
      COMMON/BLK3/PX(41),PY(41),CX(40),CY(40),S(40),SP(41),DSBD(40)
      COMMON/BLK4/ESX(41),ESY(41),ENX(40),ENY(40),ETN(40)
      COMMON/BLK5/A(40,40),B(40,40),DW(40),PERMA(40,40)
      COMMON/BLK6/UBX,UBY,OMEGA,THT0,THTMAX,FREQ,PLGMAX,RAMDA,RADIUS
      COMMON/BLK7/URX(40),URY(40),C(40),D(40),E(40),F(40)
      COMMON/BLK8/SIGMA(40),GAM(40),CP(40)
      COMMON/BLK9/TANVEL(40),EDGVEL(40),GAMVEL(40)
      COMMON/BLK10/CIRCU,STAGPT,CONST,T(2,2)
      COMMON/BLK11/NTEV,NSPV,NWMAX,NOSP,NSTAG
      COMMON/BLK12/TEVX(201),TEVY(201),TEVS(201),DSTE,TELS,TERS
      COMMON/BLK13/SPVX(201),SPVY(201),SPVS(201),DSSP
      COMMON/BLK14/TE2X,TE2Y,SP2X,SP2Y,SPPS,SPX,SPY
      COMMON/BLK15/UXTE(201),UYTE(201),UXSP(201),UYSP(201)
C
      GO TO (100,200,300,400,500),MCASE
C
C RECTILINEAR MOTION (IFF)
C
  100 CONTINUE
      UBX=0.D0
      UBY=0.D0
      OMEGA=0.D0
      WRITE(6,6000)
      GO TO 700
C
C OSCILLATION MOTION (IFF)
C
  200 CONTINUE
      UBX=0.D0
      UBY=0.D0
      THETA=THT0+THTMAX*DSIN(FREQ*DT*DBLE(NT)-.5D0*PI)
      OMEGA=FREQ*THTMAX*DCOS(FREQ*DT*DBLE(NT)-.5D0*PI)
      WRITE(6,6010)
      GO TO 700
C
C PLUNGING MOTION (IFF)
C
  300 CONTINUE
      RY=PLGMAX*DSIN(FREQ*TWOPI*DT*DBLE(NT))
      UBY=PLGMAX*FREQ*TWOPI*DCOS(FREQ*TWOPI*DT*DBLE(NT))
      UBX=0.D0
      WRITE(6,6020)
      GO TO 700
C
C  PITCHING MOTION
C
  400 CONTINUE
      IF(THETA.EQ.THTMAX) GO TO 700
      UBX=0.
      UBY=0.
      THETA=THT0+OMEGA*DT*DBLE(NT)
```

```
      IF(DABS(THETA).LE.DABS(THTMAX)) GO TO 33
      STOP
   33 CONTINUE
      ARM=SQRT(RX*RX+RY*RY)
      RX=-ARM*DSIN(THETA)
      RY=ARM*DCOS(THETA)
      WRITE(6,6022)
      GO TO 700
C
C  DARRIEUS MOTION
C
  500 CONTINUE
      TEMP1=FLOAT(NT)*DT/RADIUS
      TEMP2=DSIN(TEMP1)/RAMDA
      TEMP3=DCOS(TEMP1)/RAMDA
      TEMP4=TEMP2/(1.D0+TEMP3)
C
C  UBX,UBY.... + ; SAME DIRECTION TO THE FREE STREAM
C  THETA...... + ; OUT-WARD INCLINATION OF THE RELATIVE FLUID VEL.
C
      UBX=-TEMP3
      UBY=0.D0
      RX=-RADIUS*TEMP2
      RY=0.D0
      THETA=DATAN(TEMP4)
      OMEGA=(TEMP3+1.D0/(RAMDA*RAMDA))/
     &      (RADIUS*(1.D0+TEMP3)*(1.D0+TEMP3)*(1.D0+TEMP4*TEMP4))
      WRITE(6,6024)
C
C
  700 CONTINUE
      WRITE(6,6030)
      WRITE(6,6040) RX,RY,THETA,UBX,UBY,OMEGA,FREQ
C
C SET THE TRANSFORMATION MATRIX WHICH TRANSFORMS IFF TO BFF
C
      T(1,1)=DCOS(THETA)
      T(1,2)=DSIN(THETA)
      T(2,1)=-T(1,2)
      T(2,2)=T(1,1)
      WRITE(6,6050)
      WRITE(6,6060)(T(K,1),T(K,2),K=1,2)
C
      RETURN
C
 6000 FORMAT(///' ***** RECTILINEAR MOTION *****'/)
 6010 FORMAT(///' ***** OSCILLATING MOTION *****'/)
 6020 FORMAT(///' ***** PLUNGING MOTION *****'/)
 6022 FORMAT(///' ***** PITCHING MOTION *****'/)
 6024 FORMAT(///' ***** DARRIEUS MOTION *****'/)
 6030 FORMAT(/7X,'RX',9X,'RY',8X,'THETA',7X,'UBX',8X,
     &       'UBY',7X,'OMEGA',7X,'FREQ.')
 6040 FORMAT(7F11.5)
 6050 FORMAT(///' *** TRANSFORMATION MATRIX ***'//)
 6060 FORMAT(2(5X,E12.5))
C
      END
```

```
      SUBROUTINE NACAGM(X,Y,ND0,ND1,ND2,ND3,SHIFT)
C
      IMPLICIT REAL*8 (A-H,O-Z)
      REAL*8 MC,MT
C
      DATA C0,C1,C2,C3,C4,C5/.2969D0,-.126D0,-.3516D0,.2843D0,
     &     -.1015D0,1.1019D0/
C
C INTERPRET THE NACA DESCRIPTION
C
C   MC=MAX CAMBER
      MC=.01D0*ND0
C   PMC=POSITION OF MAX CAMBER
      PMC=0.1D0*ND1
C   MT=MAX THICKNESS
      MT=0.1D0*ND2+0.01D0*ND3
C   R=NOSE RADIUS
      R=C5*MT*MT
C  CALCULATE THE THICKNESS
C     IF(X .GE. R)  GO TO 20
C     YT=DSQRT(R**2-(R-X)**2)
C     GO TO 25
   20 YT=MT*(C0*DSQRT(X)+C1*X+C2*X**2+C3*X**3+C4*X**4)/0.2D0
   25 CONTINUE
C  CALCULATE CAMBER AND SLOPE
      IF(PMC.NE.0.D0.AND.MC.NE.0.D0)  GO TO 30
      YC=0.D0
      DYCDX=0.D0
      GO TO 45
   30 CONTINUE
      IF(X .GT. PMC)  GO TO 40
      YC=MC*(2.D0*PMC*X-X**2)/PMC**2
      DYCDX=MC*2*(PMC-X)/PMC**2
      GO TO 45
   40 YC=MC*((1.D0-2.D0*PMC)+2.D0*PMC*X-X**2)/(1.D0-PMC**2)
      DYCDX=MC*2.D0*(PMC-X)/(1.D0-PMC**2)
   45 CONTINUE
C COMPUTE THE SURFACE COORDINATES
      ANGLE=DATAN(DYCDX)
      DELX=YT*DSIN(ANGLE)
      DELY=YT*DCOS(ANGLE)
      X=(X+DELX)+SHIFT
      Y=-(YC-DELY)
C
      RETURN
C
      END
```

```
      SUBROUTINE RKM44(*,T,TOUT,Y,NEQN,ABSERR)
C*************************************************C
      IMPLICIT REAL*8(A-H,O-Z)
      DIMENSION Y(150),Y1(150),YO(150),YP(150),FE(5,150)
      COMMON/PTN/SPRT
      H=(TOUT-T)/256.DO
      ABE=0.DO
      DT=0.DO
   17 CONTINUE
      DO 18 I=1,NEQN
   18 YO(I)=Y(I)
      TO=T
      IF((ABE/ABSERR).GT.1.2DO)DT=.5DO*DT
      IF((ABE/ABSERR).LT.0.5DO)DT=2.DO*DT
      IF(ABE.EQ.0.DO)DT=H
      IF((T+DT).GT.TOUT)DT=TOUT-T
      CALL F(*100,T,Y,YP)
      DO 19 I=1,NEQN
      FE(1,I)=DT*YP(I)
   19 Y(I)=YO(I)+FE(1,I)/3.DO
      T=TO+DT/3.DO
      CALL F(*100,T,Y,YP)
      DO 20 I=1,NEQN
      FE(2,I)=DT*YP(I)
   20 Y(I)=YO(I)+(FE(1,I)+FE(2,I))/6.DO
      CALL F(*100,T,Y,YP)
      DO 21 I=1,NEQN
      FE(3,I)=DT*YP(I)
   21 Y(I)=YO(I)+(FE(1,I)+3.DO*FE(3,I))/8.DO
      T=TO+DT/2.DO
      CALL F(*100,T,Y,YP)
      DO 22 I=1,NEQN
      FE(4,I)=DT*YP(I)
      Y(I)=YO(I)+(FE(1,I)-3.DO*FE(3,I)+4.DO*FE(4,I))/2.DO
   22 Y1(I)=Y(I)
      T=TO+DT
      CALL F(*100,T,Y,YP)
      ABE=0.DO
      DO 23 I=1,NEQN
      FE(5,I)=DT*YP(I)
      Y(I)=YO(I)+(FE(1,I)+4.DO*FE(4,I)+FE(5,I))/6.DO
C<<<<<<<< NOTE:THE FOLLOWING LIMITS Y(1) TO + VALUES>>>>>>>>>
      IF(Y(1).LE.0.DO) Y(1)=1.0D-06
   23 ABE=DMAX1(ABE,DABS(Y(I)-Y1(I)))
      IF(TOUT.NE.T)GO TO 17
      GO TO 101
  100 RETURN 1
  101 CONTINUE
      RETURN
      END
```

```
      SUBROUTINE SEPCAL
C
C THIS ROUTINE IS TO CALCULATE THE SEPARATION POINT AND ITS SHEDDING
C VELOCITY.
C
      IMPLICIT REAL*8 (A-H,O-Z)
      LOGICAL STEADY,USEP,DSEP,DATASP
      REAL*8 VELTEM(40),STEMP(40)
C
      COMMON/BLK0/STEADY,USEP,DSEP
      COMMON/BLK1/NSTEPS,NT,NE,ND0,ND1,ND2,ND3,MCASE,IP(40)
      COMMON/BLK2/PI,TWOPI,REY,RX,RY,THETA,SHIFT,DT,TECON
      COMMON/BLK3/PX(41),PY(41),CX(40),CY(40),S(40),SP(41),DSBD(40)
      COMMON/BLK4/ESX(41),ESY(41),ENX(40),ENY(40),ETN(40)
      COMMON/BLK5/A(40,40),B(40,40),DW(40),PERMA(40,40)
      COMMON/BLK6/UBX,UBY,OMEGA,THTO,THTMAX,FREQ,PLGMAX,RAMDA,RADIUS
      COMMON/BLK7/URX(40),URY(40),C(40),D(40),E(40),F(40)
      COMMON/BLK8/SIGMA(40),GAM(40),CP(40)
      COMMON/BLK9/TANVEL(40),EDGVEL(40),GAMVEL(40)
      COMMON/BLK10/CIRCU,STAGPT,CONST,T(2,2)
      COMMON/BLK11/NTEV,NSPV,NWMAX,NOSP,NSTAG
      COMMON/BLK12/TEVX(201),TEVY(201),TEVS(201),DSTE,TELS,TERS
      COMMON/BLK13/SPVX(201),SPVY(201),SPVS(201),DSSP
      COMMON/BLK14/TE2X,TE2Y,SP2X,SP2Y,SPPS,SPX,SPY
      COMMON/BLK15/UXTE(201),UYTE(201),UXSP(201),UYSP(201)
C
      DATASP=.FALSE.
C
C SEPARATION POINT FROM THE STEADY SEPARATION DATA
C
      IF(.NOT.DATASP) GO TO 100
      CALL SPDATA(THETA,SPPS,SP(NE+1),USEP,DSEP)
      WRITE(6,6000) SPPS
      RETURN
C
C SEPARATION ON THE UPPER SIDE OF AIRFOIL
C
  100 CONTINUE
      CALL BNDL(EDGVEL,S,NE,STAGPT,REY,USPPS,USEP,NT,DT)
C
      IF(USPPS.LE.SP(NE/2+1)) USPPS=SP(NE/2+1)
      IF(USPPS.GE.SP(NE)) USEP=.FALSE.
      IF(USEP) WRITE(6,6010) USPPS
C
C SEPARATION ON THE LOWER SIDE OF AIRFOIL
C
      DO 1000 K=NE,1,-1
        VELTEM(K)=-EDGVEL(NE-K+1)
        STEMP(K)=SP(NE+1)-S(NE-K+1)
 1000 CONTINUE
      STGTEM=SP(NE+1)-STAGPT
C
C#### CALL DBNDL(VELTEM,STEMP,NE,STGTEM,REY,DSPPS,DSEP,NT,DT)
      IF(DSEP) DSPPS=SP(NE+1)-DSPPS
      IF(DSPPS.GT.SP(NE/2+1)) DSPPS=SP(NE/2+1)
      IF(DSPPS.LE.SP(2)) DSEP=.FALSE.
      IF(DSEP) WRITE(6,6020) DSPPS
C
      IF(USEP.AND.DSEP) GO TO 200
      IF(DSEP) SPPS=DSPPS
```

```
      IF(USEP) SPPS=USPPS
      RETURN
C
  200 CONTINUE
      IF(THETA.LE.0.D0) DSEP=.FALSE.
      IF(THETA.GT.0.D0) USEP=.FALSE.
      IF(DSEP) SPPS=DSPPS
      IF(USEP) SPPS=USPPS
      WRITE(6,6050)
      RETURN
C
 6000 FORMAT(/' STEADY SEPARATION POINT AT S =',F8.5/)
 6010 FORMAT(/' PREDICTED UPPER SEPARATION POINT AT S =',F8.5/)
 6020 FORMAT(/' PREDICTED LOWER SEPARATION POINT AT S =',F8.5/)
 6050 FORMAT(/////' EXECUTION STOPPED --- SEPARATION ON BOTH SIDES')
C
      END
```

```
      SUBROUTINE SOLVE
C
C THIS ROUTINE IS TO SOLVE THE SIMULTANEOUS EQUATIONS FOR THE SYSTEM
C
      IMPLICIT REAL*8 (A-H,O-Z)
      REAL*8 Y(41)
      LOGICAL STEADY,USEP,DSEP
C
      COMMON/BLK0/STEADY,USEP,DSEP
      COMMON/BLK1/NSTEPS,NT,NE,ND0,ND1,ND2,ND3,MCASE,IP(40)
      COMMON/BLK2/PI,TWOPI,REY,RX,RY,THETA,SHIFT,DT,TECON
      COMMON/BLK3/PX(41),PY(41),CX(40),CY(40),S(40),SP(41),DSBD(40)
      COMMON/BLK4/ESX(41),ESY(41),ENX(40),ENY(40),ETN(40)
      COMMON/BLK5/A(40,40),B(40,40),DW(40),PERMA(40,40)
      COMMON/BLK6/UBX,UBY,OMEGA,THT0,THTMAX,FREQ,PLGMAX,RAMDA,RADIUS
      COMMON/BLK7/URX(40),URY(40),C(40),D(40),E(40),F(40)
      COMMON/BLK8/SIGMA(40),GAM(40),CP(40)
      COMMON/BLK9/TANVEL(40),EDGVEL(40),GAMVEL(40)
      COMMON/BLK10/CIRCU,STAGPT,CONST,T(2,2)
      COMMON/BLK11/NTEV,NSPV,NWMAX,NOSP,NSTAG
      COMMON/BLK12/TEVX(201),TEVY(201),TEVS(201),DSTE,TELS,TERS
      COMMON/BLK13/SPVX(201),SPVY(201),SPVS(201),DSSP
      COMMON/BLK14/TE2X,TE2Y,SP2X,SP2Y,SPPS,SPX,SPY
      COMMON/BLK15/UXTE(201),UYTE(201),UXSP(201),UYSP(201)
C
      MSIZE=NE+1
      IF(NT.GE.1) MSIZE=NE+2
C
      DO 1000 K=1,MSIZE
        GAM(K)=0.D0
        DW(K)=-C(K)-D(K)-E(K)-F(K)
 1000 CONTINUE
C
      Y(1)=DW(IP(1))
      DO 1100 I=2,MSIZE
        Y(I)=DW(IP(I))
        DO 1100 J=1,I-1
          Y(I)=Y(I)-A(I,J)*Y(J)
 1100 CONTINUE
      GAM(MSIZE)=Y(MSIZE)/A(MSIZE,MSIZE)
      DO 1300 I=MSIZE-1,1,-1
        GAM(I)=Y(I)
        DO 1200 J=I+1,MSIZE
          GAM(I)=GAM(I)-A(I,J)*GAM(J)
 1200   CONTINUE
        GAM(I)=GAM(I)/A(I,I)
 1300 CONTINUE
C
C SAVE DISTURBANCE VELOCITY TO OLD PLACE AND GET THE NEW ONE DUE TO
C VORTEX DISTRIBUTION AND ADD TO RELATIVE MOTIONAL VELOCITY.
C
      TELS=GAM(1)+GAM(NE+1)
      TERS=GAM(NE+2)
      TEVS(2)=.5D0*DSTE*(TELS+TERS)
      DO 1400 I=1,NE+2
        IF(I.EQ.NE+1) GO TO 1400
        CALL GETVEL(CX(I),CY(I),TEMPVX,TEMPVY)
        EDGVEL(I)=(TEMPVX+URX(I))*ESX(I)+(TEMPVY+URY(I))*ESY(I)
 1400 CONTINUE
      DO 1420 I=1,NE
```

```
        GAMVEL(I)=TANVEL(I)-.5D0*(GAM(I)+GAM(I+1))
 1420 CONTINUE
C
      WRITE(6,6000)
      WRITE(6,6010)
      WRITE(6,6020) (K,C(K),D(K),E(K),F(K),DW(K),K=1,MSIZE)
      WRITE(6,6030)
      WRITE(6,6040) (K,SIGMA(K),GAM(K),TANVEL(K),EDGVEL(K),GAMVEL(K)
     &              ,K=1,NE+2)
C
      IF(NT.EQ.0) GO TO 1450
C
C  USE CENTER OF VORTICITY (COV)
C
      TEMP=COV(TELS,TERS)
      TEVX(2)=TEVX(1)+(TE2X-TEVX(1))*TEMP
      TEVY(2)=TEVY(1)+(TE2Y-TEVY(1))*TEMP
C
      WRITE(6,6100)
      WRITE(6,6110)
      K=1
      WRITE(6,6120) NT,K,TEVX(K),TEVY(K)
      IF(NTEV.GE.1) WRITE(6,6140)(NT,K,TEVX(K),TEVY(K),TEVS(K),
     &              K=2,NTEV+1)
C
      IF(NSPV.LE.0) GO TO 1450
C
C  USE CENTER OF VORTICITY (COV)
C
      TEMP=1.D0
      SPVX(2)=SPVX(1)+(SP2X-SPVX(1))*TEMP
      SPVY(2)=SPVY(1)+(SP2Y-SPVY(1))*TEMP
C
      WRITE(6,6115)
      K=1
      WRITE(6,6125) NT,K,SPVX(K),SPVY(K)
      IF(NSPV.GE.1) WRITE(6,6145)(NT,K,SPVX(K),SPVY(K),SPVS(K),
     &              K=2,NSPV+1)
 1450 CONTINUE
C
C GET THE BOUND CIRCULATION
C
      CIRCU=0.D0
      DO 1500 I=1,NE
        CIRCU=CIRCU+.5D0*DSBD(I)*(GAM(I)+GAM(I+1))
 1500 CONTINUE
      WRITE(6,6050) CIRCU
C
C FIND THE STAGNATION POINT WITH A LINEAR INTERPOLATION
C
      DO 1600 K=2,NE
        IF(EDGVEL(K).GE.0.D0.AND.EDGVEL(K-1).LE.0.D0) GO TO 100
 1600 CONTINUE
  100 CONTINUE
      NSTAG=K
      STAGPT = S(K-1)-EDGVEL(K-1)*(S(K)-S(K-1))/(EDGVEL(K)-
     &         EDGVEL(K-1))
      IF(STAGPT.LT.SP(NSTAG)) NSTAG=NSTAG-1
      WRITE(6,6060) STAGPT,NSTAG
C
```

```
      RETURN
C
 6000 FORMAT(///' ***** SURFACE DISTRIBUTION *****')
 6010 FORMAT(//4X,'ID',9X,'C',14X,'D',14X,'E',14X,'F',14X,'DW'/)
 6020 FORMAT((1X,I5,5(3X,F12.5)))
 6030 FORMAT(//4X,'ID',7X,'SIGMA',11X,'GAM',11X,'TANVEL',10X,'EDGVEL',
     &       10X,'GAMVEL'/)
 6040 FORMAT((1X,I5,5(3X,F12.5)))
 6050 FORMAT(///' TOTAL CIRCUALTION ON BODY =',E13.5//)
 6060 FORMAT(' STAGNATION POINT AT S =',F8.5,' ON',I3,'TH ELEMENT'/)
 6100 FORMAT(///33H ***** WAKE CHARACTERISTICS *****  //)
 6110 FORMAT(/' TE WAKE GEOMETRY'//3X,'NT  NO.',8X,'TEVX',10X,'TEVY'
     &       ,10X,'TEVS')
 6115 FORMAT(/' SP WAKE GEOMETRY'//3X,'NT  NO.',8X,'SPVX',10X,'SPVY'
     &       ,10X,'SPVS')
 6120 FORMAT(2I5,5X,E12.5,2X,E12.5,14X,' TTTTTTTTTT')
 6125 FORMAT(2I5,5X,E12.5,2X,E12.5,14X,' SSSSSSSSSS')
 6140 FORMAT(2I5,5X,E12.5,2X,E12.5,2X,E12.5,' TTTTTTTTTT')
 6145 FORMAT(2I5,5X,E12.5,2X,E12.5,2X,E12.5,' SSSSSSSSSS')
C
C
      END
```

```
      SUBROUTINE SOURCE(XS,YS,XF,YF,ESX,ESY,S,VS1X,VS1Y,VSLX,VSLY,
     &                  VSRX,VSRY)
C
      IMPLICIT REAL*8 (A-H,O-Z)
C
C THIS ROUTINE COMPUTES THE INDUCED VELOCITY DUE TO SOURCE DISTRIBUTION
C WRT BFF
C
      TWOPI=2.D0*3.14159265358979267
      XDIST=XF-XS
      YDIST=YF-YS
C
C GET THE TANGENTIAL DISTANCE XI AND THE NORMAL DISTANCE ETA FROM SOURCE
C POINT TO FIELD POINT WRT SOURCE EFF
C
      XI=XDIST*ESX+YDIST*ESY
      ETA=-XDIST*ESY+YDIST*ESX
      IF(DABS(XI).LE.1.D-03.AND.DABS(ETA).LE.1.D-03) GO TO 500
      TEMP1=XI+.5D0*S
      TEMP2=XI-.5D0*S
      ALP0=DATAN2(ETA,XI)
      ALP1=DATAN2(ETA,TEMP1)
      ALP2=DATAN2(ETA,TEMP2)
      ROSQ=XI*XI+ETA*ETA
      R1SQ=TEMP1*TEMP1+ETA*ETA
      R2SQ=TEMP2*TEMP2+ETA*ETA
      IF(R1SQ.LE.1.D-06) GO TO 300
      IF(R2SQ.LE.1.D-06) GO TO 400
      Z1=.5D0*DLOG(R1SQ/R2SQ)
      Z2=.5D0*DLOG(R1SQ/ROSQ)
      Z3=.5D0*DLOG(ROSQ/R2SQ)
      IF(DABS(ETA).LE.1.D-03) GO TO 100
      Z4=ALP2-ALP1
      Z5=ALP0-ALP1
      Z6=ALP2-ALP0
C
      S1X=Z1/TWOPI
      S1Y=Z4/TWOPI
      SLX=(XI*Z2-.5D0*S+ETA*Z5)/TWOPI
      SLY=(XI*Z5-ETA*Z2)/TWOPI
      SRX=(XI*Z3-.5D0*S+ETA*Z6)/TWOPI
      SRY=(XI*Z6-ETA*Z3)/TWOPI
      GO TO 200
C***** ETA EQUALS ZERO *****C
  100 CONTINUE
      S1X=Z1/TWOPI
      S1Y=0.D0
      IF(DABS(XI).LT.(.5D0*S)) S1Y=.5D0
      SLX=(XI*Z2-.5D0*S)/TWOPI
      SLY=0.D0
      SRX=(XI*Z3-.5D0*S)/TWOPI
      SRY=0.D0
      GO TO 200
C***** R1SQ EQUALS ZERO *****C
  300 CONTINUE
      S1X=-1.5D0
      SLX=.75D0
      Z3=.5D0*DLOG(ROSQ/R2SQ)
      SRX=(XI*Z3-.5D0*S)/TWOPI
      SRY=-ETA*Z3/TWOPI
```

```
      IF(ETA) 310,320,330
  310 S1Y=-.25D0
      SLY=.125D0
      GO TO 200
  320 S1Y=0.D0
      SLY=0.D0
      GO TO 200
  330 S1Y=.25D0
      SLY=-.125D0
      GO TO 200
C***** R2SQ EQUALS ZERO *****C
  400 CONTINUE
      S1X=1.5D0
      SRX=.75D0
      Z2=.5D0*DLOG(R1SQ/ROSQ)
      SLX=(XI*Z2-.5D0*S)/TWOPI
      SLY=-ETA*Z2/TWOPI
      IF(ETA) 410,420,430
  410 S1Y=-.25D0
      SRY=-.125D0
      GO TO 200
  420 S1Y=0.D0
      SRY=0.D0
      GO TO 200
  430 S1Y=.25D0
      SRY=.125D0
      GO TO 200
C***** ROSQ EQUALS ZERO *****C
  500 CONTINUE
      S1X=0.D0
      SLX=-.5D0*S/TWOPI
      SLY=0.D0
      SRX=SLX
      SRY=0.D0
C*NOTE* NEXT 1 LINE * SURFACE IS ASSUMED TO BE CONVEX ONLY
      IF(ETA) 530,530,530
  510 S1Y=-.5D0
      GO TO 200
  520 S1Y=0.D0
      GO TO 200
  530 S1Y=.5D0
      GO TO 200
C
  200 CONTINUE
C
C TRANSFORM THE VELOCITIES FROM SOURCE EFF TO BFF
C
      VS1X=S1X*ESX-S1Y*ESY
      VS1Y=S1X*ESY+S1Y*ESX
      VSLX=SLX*ESX-SLY*ESY
      VSLY=SLX*ESY+SLY*ESX
      VSRX=SRX*ESX-SRY*ESY
      VSRY=SRX*ESY+SRY*ESX
C
      RETURN
C
      END
```

```
      SUBROUTINE SPDATA(THETA,SPPS,STOTL,USEP,DSEP)
C
      REAL*8 SEP(22),ANG(22)
      LOGICAL USEP,DSEP
C
      DATA ANG/00.D0,02.D0,04.D0,06.D0,08.D0,10.D0,12.D0,14.D0,
     &         16.D0,18.D0,20.D0,22.D0,24.D0,26.D0,28.D0,30.D0,
     &         32.D0,34.D0,36.D0,38.D0,40.D0,90.D0/
C REY=42,900
      DATA SEP/2.05590D0,1.92356D0,1.79383D0,1.67403D0,1.45483D0,
     &         1.08252D0,1.07537D0,1.06481D0,1.05836D0,1.01985D0,
     &         1.05436D0,1.05436D0,1.05440D0,1.05440D0,1.05430D0,
     &         1.05430D0,1.04971D0,1.04626D0,1.04510D0,1.04512D0,
     &         1.04204D0,1.02795D0/
C REY=655,000
C     DATA SEP/2.05590D0,1.96924D0,1.94527D0,1.92629D0,1.83722D0,
C    &         1.72571D0,1.58010D0,1.46125D0,1.48790D0,1.19103D0,
C    &         1.16702D0,1.12196D0,1.11241D0,1.10185D0,1.10922D0,
C    &         1.09416D0,1.09384D0,1.09384D0,1.08107D0,1.07918D0,
C    &         1.02795D0/
C REY=1,270,000
C     DATA SEP/2.05590D0,1.98421D0,1.95806D0,1.93698D0,1.89373D0,
C    &         1.81057D0,1.68725D0,1.56899D0,1.45697D0,1.30551D0,
C    &         1.24677D0,1.18540D0,1.16489D0,1.12290D0,1.12620D0,
C    &         1.11488D0,1.11488D0,1.11064D0,1.10639D0,1.08197D0,
C    &         1.02795D0/
C
      DEG=DABS(THETA*180.D0/3.14159265358979267)
      USEP=.FALSE.
      DSEP=.FALSE.
      IF(DEG.LE.ANG(1)) RETURN
      IF(THETA.GT.0.D0) DSEP=.TRUE.
      IF(THETA.LE.0.D0) USEP=.TRUE.
      IMAX=22
      DO 100 I=2,IMAX
      IF(DEG.LT.ANG(I)) GO TO 110
  100 CONTINUE
      SPPS=SEP(IMAX)
      IF(DSEP) SPPS=STOTL-SPPS
      RETURN
  110 CONTINUE
      SPPS=SEP(I-1)+(SEP(I)-SEP(I-1))*(DEG-ANG(I-1))/(ANG(I)-ANG(I-1))
      IF(DSEP) SPPS=STOTL-SPPS
      RETURN
      END
```

```
      SUBROUTINE SPGEOM
C
C THIS ROUTINE COMPUTES THE NEW SP WAKE GEOMETRY
C
      IMPLICIT REAL*8 (A-H,O-Z)
      REAL*8 CEX(41),CEY(41),CFX(41),CFY(41),CGX(41),CGY(41)
      LOGICAL STEADY,USEP,DSEP
C
      COMMON/BLK0/STEADY,USEP,DSEP
      COMMON/BLK1/NSTEPS,NT,NE,ND0,ND1,ND2,ND3,MCASE,IP(40)
      COMMON/BLK2/PI,TWOPI,REY,RX,RY,THETA,SHIFT,DT,TECON
      COMMON/BLK3/PX(41),PY(41),CX(40),CY(40),S(40),SP(41),DSBD(40)
      COMMON/BLK4/ESX(41),ESY(41),ENX(40),ENY(40),ETN(40)
      COMMON/BLK5/A(40,40),B(40,40),DW(40),PERMA(40,40)
      COMMON/BLK6/UBX,UBY,OMEGA,THTO,THTMAX,FREQ,PLGMAX,RAMDA,RADIUS
      COMMON/BLK7/URX(40),URY(40),C(40),D(40),E(40),F(40)
      COMMON/BLK8/SIGMA(40),GAM(40),CP(40)
      COMMON/BLK9/TANVEL(40),EDGVEL(40),GAMVEL(40)
      COMMON/BLK10/CIRCU,STAGPT,CONST,T(2,2)
      COMMON/BLK11/NTEV,NSPV,NWMAX,NOSP,NSTAG
      COMMON/BLK12/TEVX(201),TEVY(201),TEVS(201),DSTE,TELS,TERS
      COMMON/BLK13/SPVX(201),SPVY(201),SPVS(201),DSSP
      COMMON/BLK14/TE2X,TE2Y,SP2X,SP2Y,SPPS,SPX,SPY
      COMMON/BLK15/UXTE(201),UYTE(201),UXSP(201),UYSP(201)
C
      IF(NSPV.EQ.0) GO TO 100
C
C  COMPUTE THE NEW SP WAKE GEOMETRY (IFF)
C
      IF(.NOT.(USEP.OR.DSEP)) GO TO 220
      DO 1100 K=NSPV+2,3,-1
        SPVX(K)=SPVX(K-1)+UXSP(K-1)*DT+RX
        SPVY(K)=SPVY(K-1)+UYSP(K-1)*DT+RY
        SPVS(K)=SPVS(K-1)
C
C  TEST THE CROSSING OF THE WAKE ELEMENT
C
      CALL CROSS(SPVX(K),SPVY(K),T,RX,RY,SHIFT,ND0,ND1,ND2,ND3)
 1100 CONTINUE
      GO TO 250
  220 CONTINUE
      DO 1150 K=NSPV+1,2,-1
        SPVX(K)=SPVX(K)+UXSP(K)*DT+RX
        SPVY(K)=SPVY(K)+UYSP(K)*DT+RY
 1150 CONTINUE
      RETURN
  250 CONTINUE
C
C  SHED NEW WAKE (IFF) & INCREASE SP WAKE ELEMENT NUMBER BY 1.
C
  100 CONTINUE
      IF(.NOT.(USEP.OR.DSEP)) RETURN
      NSPV=NSPV+1
      IF(NSPV.GT.NWMAX) NSPV=NWMAX
C
C FIND THE ELEMENT NUMBER WHERE THE SEPARATION OCCURS
C
      NOSP=0
      DO 1160 K=2,NE+1
        IF(SPPS.LT.SP(K)) GO TO 310
```

```
 1160 CONTINUE
  310 NOSP=K-1
C
C FIND SEPARATION POINT WRT BFF
C
      PRINT *,' '
      PRINT *,'NOSP =',NOSP,'  <----- ELEMENT NO. WHERE SEP. OCCUR'
C
C AVOID THE CASE WHEN THE SEPARATION OCCURS AT THE CENTER POINT
C
      TERR=4.D-3
      TEMP=SPPS-S(NOSP)
      IF(DABS(TEMP).LE.TERR) SPPS=S(NOSP)+TERR*TEMP/DABS(TEMP)
C
C LOCATION IN CENTER POINT-WISE
C
      IF(SPPS.GT.S(NOSP)) NTEMP=NOSP
      IF(SPPS.LT.S(NOSP)) NTEMP=NOSP-1
      RATIO1=(SPPS-S(NTEMP))/(S(NTEMP+1)-S(NTEMP))
C
C LOCATION IN BOUNDARY POINT-WISE
C
      RATIO=(SPPS-SP(NOSP))/(SP(NOSP+1)-SP(NOSP))
C
C SEPARATION POINT WRT BFF
C
      SPX=PX(NOSP)+RATIO*(PX(NOSP+1)-PX(NOSP))
      SPY=PY(NOSP)+RATIO*(PY(NOSP+1)-PY(NOSP))
C
C  TRANSFORM THE SEPARATION POINT WRT I.F.F.
C
      SPVX(1)=SPX*T(1,1)+SPY*T(2,1)+RX
      SPVY(1)=SPX*T(1,2)+SPY*T(2,2)+RY
C
C FIND THE SEPARATION SHOOTING VEL. AND ITS STRENGTH
C
      SEPVEL=EDGVEL(NTEMP)+RATIO1*(EDGVEL(NTEMP+1)-EDGVEL(NTEMP))
      SPVS(2)=-.5D0*SEPVEL*SEPVEL*DT
      DSSP=DABS(SEPVEL)*DT
C
C  TRANSFORM THE SEPARATION VORTEX VELOCITY WRT B.F.F.
C
      TANGLE=36.D0*PI/180.D0
      USP=SEPVEL*DCOS(TANGLE)
      VSP=DABS(SEPVEL)*DSIN(TANGLE)
      SEPDVX=USP*ESX(NOSP)-VSP*ESY(NOSP)
      SEPDVY=USP*ESY(NOSP)+VSP*ESX(NOSP)
      IF(SEPDVX.GE.0.D0) GO TO 444
      SEPDVX=0.D0
      SEPDVY=DABS(SEPVEL)
  444 CONTINUE
C
C  THE SEPARATION VORTEX VELOCITY WRT I.F.F.
C
      SPWUX=SEPDVX*T(1,1)+SEPDVY*T(2,1)+UBX
      SPWUY=SEPDVX*T(1,2)+SEPDVY*T(2,2)+UBY
C
C  NEW POSITION OF THE SEP WAKE WRT I.F.F.
C
      SP2X=SPWUX*DT+SPVX(1)
```

```
      SP2Y=SPWUY*DT+SPVY(1)
      TEMP=1.D0
      SPVX(2)=SPVX(1)+(SP2X-SPVX(1))*TEMP
      SPVY(2)=SPVY(1)+(SP2Y-SPVY(1))*TEMP
C
      WRITE(6,6000) SPPS,NT,SPX,SPY,SEPVEL,SEPDVX,
     &              SEPDVY,REY
C
      RETURN
C
 6000 FORMAT(//1X,'SEP POINT WRT SURF. COOR.=',F6.3,'  NT=',I4/
     &         1X,'SEP POINT WRT BODY F. FRAME =',2(F6.3,3X)/
     &         1X,'SEPARATION EDGE VELOCITY =',F8.5/
     &         1X,'SHOOTING VECTOR WRT B.F.F =',2(F8.5,3X)/
     &         1X,'REYNOLD NUMBER =',E15.5//)
C
      END
```

```
      SUBROUTINE STAG(REL,HI,THEI)
C*****************************************C
      IMPLICIT REAL*8(A-H,O-Z)
      COMMON/VEL/U(150),X(150),NS
      COMMON/FIN/RST,FT,AT,GT,HT,XNS,XNL,XNT,TURB,NTM
      COMMON/PTN/SPRT
      DO 10 I=1,NS
      IF(X(I).GT.XNS) GO TO 20
   10 CONTINUE
   20 UI=U(I)
      XNL=X(I)
      XSL=XNL-XNS
      HI=2.216D0
      TI=.29234D0
      REXI=UI*XSL*REL
      THEI=TI*XSL/REXI**0.5D0
      RETURN
      END
```

```
      SUBROUTINE TEGEOM
C
C THIS ROUTINE COMPUTES THE NEW TE & SP WAKE GEOMETRY
C
      IMPLICIT REAL*8 (A-H,O-Z)
C
      COMMON/BLK0/STEADY,USEP,DSEP
      COMMON/BLK1/NSTEPS,NT,NE,ND0,ND1,ND2,ND3,MCASE,IP(40)
      COMMON/BLK2/PI,TWOPI,REY,RX,RY,THETA,SHIFT,DT,TECON
      COMMON/BLK3/PX(41),PY(41),CX(40),CY(40),S(40),SP(41),DSBD(40)
      COMMON/BLK4/ESX(41),ESY(41),ENX(40),ENY(40),ETN(40)
      COMMON/BLK5/A(40,40),B(40,40),DW(40),PERMA(40,40)
      COMMON/BLK6/UBX,UBY,OMEGA,THT0,THTMAX,FREQ,PLGMAX,RAMDA,RADIUS
      COMMON/BLK7/URX(40),URY(40),C(40),D(40),E(40),F(40)
      COMMON/BLK8/SIGMA(40),GAM(40),CP(40)
      COMMON/BLK9/TANVEL(40),EDGVEL(40),GAMVEL(40)
      COMMON/BLK10/CIRCU,STAGPT,CONST,T(2,2)
      COMMON/BLK11/NTEV,NSPV,NWMAX,NOSP,NSTAG
      COMMON/BLK12/TEVX(201),TEVY(201),TEVS(201),DSTE,TELS,TERS
      COMMON/BLK13/SPVX(201),SPVY(201),SPVS(201),DSSP
      COMMON/BLK14/TE2X,TE2Y,SP2X,SP2Y,SPPS,SPX,SPY
      COMMON/BLK15/UXTE(201),UYTE(201),UXSP(201),UYSP(201)
C
      IF(NTEV.EQ.0) GO TO 100
C
C  COMPUTE THE NEW WAKE GEOMETRY (IFF)
C
      DO 1100 K=NTEV+2,3,-1
        TEVX(K)=TEVX(K-1)+UXTE(K-1)*DT+RX
        TEVY(K)=TEVY(K-1)+UYTE(K-1)*DT+RY
        TEVS(K)=TEVS(K-1)
C
C  TEST THE CROSSING OF THE WAKE ELEMENT
C
      CALL CROSS(TEVX(K),TEVY(K),T,RX,RY,SHIFT,ND0,ND1,ND2,ND3)
 1100 CONTINUE
C
C  SHED NEW WAKE (IFF) & INCREASE TE WAKE ELEMENT NUMBER BY 1.
C
  100 CONTINUE
      NTEV=NTEV+1
      IF(NTEV.GT.NWMAX) NTEV=NWMAX
      VELAVG=EDGVEL(NE+2)
C
      DSTE=VELAVG*DT
      TEVX(1)=PX(1)*T(1,1)+PY(1)*T(2,1)+RX
      TEVY(1)=PX(1)*T(1,2)+PY(1)*T(2,2)+RY
      TEMX=DSTE*ESX(NE+2)
      TEMY=DSTE*ESY(NE+2)
      TE2X=TEMX*T(1,1)+TEMY*T(2,1)+TEVX(1)-UBX*DT
      TE2Y=TEMX*T(1,2)+TEMY*T(2,2)+TEVY(1)-UBY*DT
C
      RETURN
C
      END
```

```
      SUBROUTINE TRANS(*,SH,RST,XN,XNS,XNT,Y,NTM)
C*******************************************************C
      IMPLICIT REAL*8(A-H,O-Z)
      COMMON/OLD/OAT(150),ODSL(150),OXNL,ONT,ONSEP,XSEP,OXSEP
      COMMON/BLI/HI,THEI,RTHI
      COMMON/PTN/SPRT
      DIMENSION Y(2)
      IF(SH.GE.(1.D0-1.D0/3.9D0)) GO TO 10
      H=1.D0/(1.D0-SH)
      RTH=RST/H
      THE=Y(1)/H
      REX=RST*(XN-XNS)/Y(1)
      IF(REX.EQ.0.D0) GO TO 20
      RHS=1.174D0*(1.D0+22400.D0/REX)*DABS(REX)**0.46D0
      IF(RTH.GE.RHS) GO TO 10
      GO TO 20
   10 IF(DFLOAT(NTM).EQ.ONT) GO TO 20
      XNT=XN
      ONT=DFLOAT(NTM)
      WRITE(6,30) XNT
   30 FORMAT(4X,'THE TRANSITION POINT (XNT)=',E12.5)
      HI=H
   40 IF(HI.GT.2.76D0) HI=2.76D0
   50 CONTINUE
      THEI=THE
      RETURN 1
   20 CONTINUE
      RETURN
      END
```

```
      SUBROUTINE TSEP(*,SH,XN,Y,NTM)
C*****************************************C
      IMPLICIT REAL*8(A-H,O-Z)
      LOGICAL SPRT
      COMMON/OLD/OAT(150),ODSL(150),OXNL,ONT,ONSEP,XSEP,OXSEP
      COMMON/PTN/SPRT
      DIMENSION Y(2)
      H=1.D0/(1.D0-SH)
      RHS=1.D0+1.D0/(1.D0-Y(2))
      IF(H.GE.RHS) GO TO 10
      GO TO 20
   10 IF(DFLOAT(NTM).EQ.ONSEP) GO TO 20
      OXSEP=XSEP
      XSEP =XN
      ONSEP=DFLOAT(NTM)
      SPRT = .TRUE.
      WRITE(6,30) XSEP,H,RHS
   30 FORMAT(4X,'INTERMITTENT SEP POINT (XSEP) =',E12.5,
     &       2X,'H =',E12.5,2X,'RHS =',E12.5)
      RETURN 1
   20 CONTINUE
      RETURN
      END
```

```
      SUBROUTINE TURBL(DELT,HI,THEI,REL)
C*****************************************************C
      IMPLICIT REAL*8(A-H,O-Z)
      COMMON/FIN/RST,FT,AT,GT,HT,XNS,XNL,XNT,TURB,NTM
      COMMON/FOUT/VT,SH,CF2
      COMMON/VEL/U(150),X(150),NS
      COMMON/OLD/OAT(150),ODSL(150),OXNL,ONT,ONSEP,XSEP,OXSEP
      COMMON/PTN/SPRT
      DIMENSION Y(2),YP(2)
      OXNT=OXT
      DO 20 I=2,NS
      IF(X(I).GT.XNT) GO TO 10
      GO TO 20
   10 CONTINUE
      NXT=I-1
      GO TO 30
   20 CONTINUE
   30 CONTINUE
      UI=U(NXT)+(XNT-X(NXT))*(U(NXT+1)-U(NXT))/(X(NXT+1)-X(NXT))
      RTHI=UI*THEI*REL
      RSTI=HI*RTHI
      DSLI=HI*THEI
      DSDI=(.9D0*HI-1.D0)/(1.3D0*HI)
      DLI=DSLI/DSDI
      CF2I=0.051D0*DABS(1.D0-2.D0*DSDI)**1.732D0/DABS(RSTI/DSDI)
     $**.268D0*DABS(1.D0-2.D0*DSDI)/(1.D0-2.D0*DSDI)
      RST=RSTI
      Y(1)=DSLI
      Y(2)=DSDI
      WRITE(6,50) NXT,XNT,UI,DLI,DSLI,THEI,HI,CF2I,RTHI
      NS1=NS-1
      DO 60 NX=NXT,NS1
      X1=X(NX)
      X2=X(NX+1)
      IF(NX.EQ.NXT) X1=XNT
      CALL GRAD(NX,DELT,Y(1))
      CALL RKM44(*100,X1,X2,Y,2,1.0D-05)
      RST=U(NX+1)*Y(1)*RSTI/DSLI/UI
      CF2=.1681D0*VT*DABS(VT)
      DEL=Y(1)/Y(2)
      H=1.D0/(1.D0-SH)
      THE=Y(1)/H
      RTH=RST/H
      UE=U(NX+1)
      NXP1=NX+1
      WRITE(6,50) NXP1,X2,UE,DEL,Y(1),THE,H,CF2,RTH
   50 FORMAT(1X,I3,4X,8E10.3)
   60 CONTINUE
  100 CONTINUE
      RETURN
      END
```

```
      SUBROUTINE VORTEX(XS,YS,XF,YF,ESX,ESY,S,VV1X,VV1Y,VV2X,VV2Y)
C
      IMPLICIT REAL*8 (A-H,O-Z)
C
C THIS ROUTINE COMPUTES THE INDUCED VELOCITY DUE TO VORTEX DISTRIBUTION
C WRT BFF
C
      TWOPI=2.D0*3.14159265358979267
      XDIST=XF-XS
      YDIST=YF-YS
      IF(S.LE..005D0) GO TO 700
C
C GET THE TANGENTIAL DISTANCE XI AND THE NORMAL DISTANCE ETA FROM SOURCE
C POINT TO FIELD POINT WRT SOURCE EFF
C
      XI=XDIST*ESX+YDIST*ESY
      ETA=-XDIST*ESY+YDIST*ESX
      IF(DABS(XI).LE.1.D-03.AND.DABS(ETA).LE.1.D-03) GO TO 500
      TEMP1=XI+.5D0*S
      TEMP2=XI-.5D0*S
      ALP1=DATAN2(ETA,TEMP1)
      ALP2=DATAN2(ETA,TEMP2)
      R1SQ=TEMP1*TEMP1+ETA*ETA
      R2SQ=TEMP2*TEMP2+ETA*ETA
      IF(R1SQ.LE.1.D-06) GO TO 300
      IF(R2SQ.LE.1.D-06) GO TO 400
      Z2=.5D0*DLOG(R1SQ/R2SQ)
      IF(DABS(ETA).LE.1.D-03) GO TO 100
      Z1=ALP1-ALP2
C
      V1X=Z1/TWOPI
      V1Y=Z2/TWOPI
      V2X=(XI*Z1+ETA*Z2)/TWOPI
      V2Y=(XI*Z2-S-ETA*Z1)/TWOPI
      GO TO 200
C***** ETA EQUALS ZERO *****C
  100 CONTINUE
      V1X=0.D0
      V1Y=Z2/TWOPI
      V2X=0.D0
      V2Y=(XI*Z2-S)/TWOPI
      GO TO 200
C***** R1SQ EQUALS ZERO *****C
  300 CONTINUE
      V1Y=-1.5D0
      V2Y=.75D0
      IF(ETA) 310,320,330
  310 V1X=.25D0
      V2X=-.125D0
      GO TO 200
  320 V1X=0.D0
      V2X=0.D0
      GO TO 200
  330 V1X=-.25D0
      V2X=.125D0
      GO TO 200
C***** R2SQ EQUALS ZERO *****C
  400 CONTINUE
      V1Y=1.5D0
      V2Y=.75D0
```

```
      IF(ETA) 410,420,430
  410 V1X=.25D0
      V2X=.125D0
      GO TO 200
  420 V1X=0.D0
      V2X=0.D0
      GO TO 200
  430 V1X=-.25D0
      V2X=-.125D0
      GO TO 200
C***** ROSQ EQUALS ZERO *****C
  500 CONTINUE
      V1Y=0.D0
      V2X=0.D0
      V2Y=-S/TWOPI
C*NOTE* NEXT 1 LINE * SURFACE IS ASSUMED TO BE CONVEX ONLY
      IF(ETA) 530,530,530
  510 V1X=.5D0
      GO TO 200
  520 V1X=0.D0
      GO TO 200
  530 V1X=-.5D0
      GO TO 200
C
  200 CONTINUE
C
C TRANSFORM THE VELOCITIES FROM SOURCE EFF TO BFF
C
      VV1X=V1X*ESX-V1Y*ESY
      VV1Y=V1X*ESY+V1Y*ESX
      VV2X=V2X*ESX-V2Y*ESY
      VV2Y=V2X*ESY+V2Y*ESX
C
      RETURN
C
  700 CONTINUE
      RDIST=XDIST*XDIST+YDIST*YDIST
      VV1X=-YDIST*S/(TWOPI*RDIST)
      VV1Y= XDIST*S/(TWOPI*RDIST)
      VV2X=0.D0
      VV2Y=0.D0
C
      RETURN
C
      END
```

```
      SUBROUTINE WAKINF
C
C THIS ROUTINE CALCULATES THE CONTRIBUTION OF THE TE AND SP-WAKE TO
C THE NORMAL DOWNWASH AT THE CENTROID OF BODY SURFACE ELEMENT
C
      IMPLICIT REAL*8 (A-H,O-Z)
      REAL*8 TEMADD(40),SPMADD(40)
      LOGICAL STEADY,USEP,DSEP
C
      COMMON/BLK0/STEADY,USEP,DSEP
      COMMON/BLK1/NSTEPS,NT,NE,ND0,ND1,ND2,ND3,MCASE,IP(40)
      COMMON/BLK2/PI,TWOPI,REY,RX,RY,THETA,SHIFT,DT,TECON
      COMMON/BLK3/PX(41),PY(41),CX(40),CY(40),S(40),SP(41),DSBD(40)
      COMMON/BLK4/ESX(41),ESY(41),ENX(40),ENY(40),ETN(40)
      COMMON/BLK5/A(40,40),B(40,40),DW(40),PERMA(40,40)
      COMMON/BLK6/UBX,UBY,OMEGA,THT0,THTMAX,FREQ,PLGMAX,RAMDA,RADIUS
      COMMON/BLK7/URX(40),URY(40),C(40),D(40),E(40),F(40)
      COMMON/BLK8/SIGMA(40),GAM(40),CP(40)
      COMMON/BLK9/TANVEL(40),EDGVEL(40),GAMVEL(40)
      COMMON/BLK10/CIRCU,STAGPT,CONST,T(2,2)
      COMMON/BLK11/NTEV,NSPV,NWMAX,NOSP,NSTAG
      COMMON/BLK12/TEVX(201),TEVY(201),TEVS(201),DSTE,TELS,TERS
      COMMON/BLK13/SPVX(201),SPVY(201),SPVS(201),DSSP
      COMMON/BLK14/TE2X,TE2Y,SP2X,SP2Y,SPPS,SPX,SPY
      COMMON/BLK15/UXTE(201),UYTE(201),UXSP(201),UYSP(201)
C
      MSIZE=NE+2
C
C INITIALIZE THE DOWNWASH ARRAY
C
      DO 1000 I=1,MSIZE
        E(I)=0.D0
        F(I)=0.D0
 1000 CONTINUE
C
C DOWNWASH DUE TO THE FIRST TE WAKE ELEMENT
C
      TEMVX=.5D0*(TE2X+TEVX(1))
      TEMVY=.5D0*(TE2Y+TEVY(1))
      CTEVX=TEMVX*T(1,1)+TEMVY*T(1,2)-RX
      CTEVY=TEMVX*T(2,1)+TEMVY*T(2,2)-RY
      DO 1010 I=1,MSIZE
        IF(I.EQ.NE+1) GO TO 1010
        CALL VORTEX(CTEVX,CTEVY,CX(I),CY(I),ESX(NE+2),ESY(NE+2),
     &              DSTE,VV1X,VV1Y,VV2X,VV2Y)
        CCX=.5D0*VV1X-VV2X/DSTE
        CCY=.5D0*VV1Y-VV2Y/DSTE
        CDX=.5D0*VV1X+VV2X/DSTE
        CDY=.5D0*VV1Y+VV2Y/DSTE
        TEMADD(I)=CCX*ENX(I)+CCY*ENY(I)
        A(I,NE+2)=CDX*ENX(I)+CDY*ENY(I)
 1010 CONTINUE
      TEMADD(NE+1)=.5D0*DSTE
      A(NE+1,NE+2)=.5D0*DSTE
C
C  DOWNWASH FROM THE LUMPED TE WAKE
C
      IF(NTEV.LE.1) GO TO 2000
      DO 1030 J=3,NTEV+1
        TEVXB=TEVX(J)*T(1,1)+TEVY(J)*T(1,2)-RX
```

```
        TEVYB=TEVX(J)*T(2,1)+TEVY(J)*T(2,2)-RY
        DO 1020 I=1,MSIZE
          IF(I.EQ.NE+1) GO TO 1020
          CALL LMPVTX(2,J,TEVXB,TEVYB,CX(I),CY(I),VELX,VELY)
          E(I)=E(I)+(VELX*ENX(I)+VELY*ENY(I))*TEVS(J)
 1020   CONTINUE
        E(NE+1)=E(NE+1)+TEVS(J)
 1030 CONTINUE
C
 2000 CONTINUE
C-----------------------------------------------------------
C  DOWNWASH FROM THE LUMPED SP WAKE
C
 2025 CONTINUE
      IF(NSPV.LT.1) GO TO 3000
      DO 2030 J=2,NSPV+1
        SPVXB=SPVX(J)*T(1,1)+SPVY(J)*T(1,2)-RX
        SPVYB=SPVX(J)*T(2,1)+SPVY(J)*T(2,2)-RY
        DO 2040 I=1,MSIZE
          IF(I.EQ.NE+1) GO TO 2040
          CALL LMPVTX(2,J,SPVXB,SPVYB,CX(I),CY(I),VELX,VELY)
          F(I)=F(I)+(VELX*ENX(I)+VELY*ENY(I))*SPVS(J)
 2040   CONTINUE
        F(NE+1)=F(NE+1)+SPVS(J)
 2030 CONTINUE
C
 3000 CONTINUE
C-----------------------------------------------------------
C SETUP THE NEW MATRIX A
C
      DO 3110 I=1,MSIZE
        DO 3120 J=2,NE
 3120     A(I,J)=PERMA(I,J)
        A(I,1)=PERMA(I,1)+TEMADD(I)
        A(I,NE+1)=PERMA(I,NE+1)+TEMADD(I)
 3110 CONTINUE
C
 4000 CONTINUE
C
 7000 CONTINUE
 7010 CONTINUE
C
      RETURN
C
 6000 FORMAT(' ROW',I3,9(1X,E10.4,1X),50(/7X,9(1X,E10.4,1X)))
C
      END
```

```
      SUBROUTINE WAKVEL
C
C THIS ROUTINE IS TO COMPUTE THE INDUCED VELOCITIES OF THE WAKE ELEMENTS
C DUE TO THE ENTIRE SINGULARITIES IN THE FLOW FIELD.
C
      IMPLICIT REAL*8 (A-H,O-Z)
C
      COMMON/BLK0/STEADY,USEP,DSEP
      COMMON/BLK1/NSTEPS,NT,NE,ND0,ND1,ND2,ND3,MCASE,IP(40)
      COMMON/BLK2/PI,TWOPI,REY,RX,RY,THETA,SHIFT,DT,TECON
      COMMON/BLK3/PX(41),PY(41),CX(40),CY(40),S(40),SP(41),DSBD(40)
      COMMON/BLK4/ESX(41),ESY(41),ENX(40),ENY(40),ETN(40)
      COMMON/BLK5/A(40,40),B(40,40),DW(40),PERMA(40,40)
      COMMON/BLK6/UBX,UBY,OMEGA,THT0,THTMAX,FREQ,PLGMAX,RAMDA,RADIUS
      COMMON/BLK7/URX(40),URY(40),C(40),D(40),E(40),F(40)
      COMMON/BLK8/SIGMA(40),GAM(40),CP(40)
      COMMON/BLK9/TANVEL(40),EDGVEL(40),GAMVEL(40)
      COMMON/BLK10/CIRCU,STAGPT,CONST,T(2,2)
      COMMON/BLK11/NTEV,NSPV,NWMAX,NOSP,NSTAG
      COMMON/BLK12/TEVX(201),TEVY(201),TEVS(201),DSTE,TELS,TERS
      COMMON/BLK13/SPVX(201),SPVY(201),SPVS(201),DSSP
      COMMON/BLK14/TE2X,TE2Y,SP2X,SP2Y,SPPS,SPX,SPY
      COMMON/BLK15/UXTE(201),UYTE(201),UXSP(201),UYSP(201)
C
C  GET THE INDUCED VELOCITY DUE TO THE SINGULARITY DISTRIBUTION IN THE
C  FIELD
C
      IF(NTEV.LE.0) GO TO 1010
      DO 1000 K=2,NTEV+1
        TEVXB=TEVX(K)*T(1,1)+TEVY(K)*T(1,2)-RX
        TEVYB=TEVX(K)*T(2,1)+TEVY(K)*T(2,2)-RY
        CALL GETVEL(TEVXB,TEVYB,TEMPU,TEMPV)
        TEMPU=TEMPU+T(1,1)*(1.D0-UBX)-T(1,2)*UBY+OMEGA*TEVYB
        TEMPV=TEMPV+T(2,1)*(1.D0-UBX)-T(2,2)*UBY-OMEGA*TEVXB
        UXTE(K)=TEMPU*T(1,1)+TEMPV*T(2,1)
        UYTE(K)=TEMPU*T(1,2)+TEMPV*T(2,2)
 1000 CONTINUE
 1010 CONTINUE
C
C GET THE INDUCED VELOCITY DUE TO THE SINGULARITY DISTRIBUTION IN THE
C FIELD
C
      IF(NSPV.LE.0) GO TO 1110
      DO 1100 K=2,NSPV+1
        SPVXB=SPVX(K)*T(1,1)+SPVY(K)*T(1,2)-RX
        SPVYB=SPVX(K)*T(2,1)+SPVY(K)*T(2,2)-RY
        CALL GETVEL(SPVXB,SPVYB,TEMPU,TEMPV)
        TEMPU=TEMPU+T(1,1)*(1.D0-UBX)-T(1,2)*UBY+OMEGA*SPVYB
        TEMPV=TEMPV+T(2,1)*(1.D0-UBX)-T(2,2)*UBY-OMEGA*SPVXB
        UXSP(K)=TEMPU*T(1,1)+TEMPV*T(2,1)
        UYSP(K)=TEMPU*T(1,2)+TEMPV*T(2,2)
 1100 CONTINUE
 1110 CONTINUE
C
      RETURN
C
      END
```

DISTRIBUTION:

Alcoa Technical Center (5)
Aluminum Company of America
Alcoa Center, PA 15069
Attn: D. K. Ai
J. T. Huang
J. R. Jombock
M. Klingensmith
J. L. Prohaska

Alternative Sources of Energy
Milaca, MN 56353
Attn: L. Stoiaken

Amarillo College
Amarillo, TX 79100
Attn: E. Gilmore

American Wind Energy Association
1017 King Street
Alexandria, VA 22314

Arizona State University
University Library
Tempe, AZ 85281
Attn: M. E. Beecher

Dr. A. S. Barker
Trinity Western
7600 Glover Road
Langley, BC
CANADA V3A 4R9

Battelle-Pacific Northwest Laboratory
P.O. Box 999
Richland, WA 99352
Attn: L. Wendell

Bechtel Group, Inc.
P.O. Box 3965
San Francisco, CA 94119
Attn: B. Lessley

Dr. George Bergeles
Dept. of Mechanical Engineering
National Technical University
42, Patission Street
Athens, GREECE

Bonneville Power Administration
P.O. Box 3621
Portland, OR 97208
Attn: N. Butler

Burns & Roe, Inc.
800 Kinderkamack Road
Oradell, NJ 07649
Attn: G. A. Fontana

Canadian Standards Association
178 Rexdale Blvd.
Rexdale, Ontario, M9W 1R3
CANADA
Attn: T. Watson

Mark Chappel
Division of Energy
National Research Council
of Canada
Montreal Road
Ottawa, Ontario
CANADA K1A OR6

Professor V. A. L. Chasteau
School of Engineering
University of Auckland
Private Bag
Auckland, NEW ZEALAND

Colorado State University
Dept. of Civil Engineering
Fort Collins, CO 80521
Attn: R. N. Meroney

Commonwealth Electric Co.
Box 368
Vineyard Haven, MA 02568
Attn: D. W. Dunham

Gale B. Curtis
Curtis Associates
3089 Oro Blanco Drive
Colorado Springs, CO 80917

M. M. Curvin
11169 Loop Road
Soddy Daisy, TN 37379

Department of Economic Planning
and Development
Barrett Building
Cheyenne, WY 82002
Attn: G. N. Monsson

Otto de Vries
National Aerospace Laboratory
Anthony Fokkerweg 2
Amsterdam 1017
THE NETHERLANDS

DOE/ALO
Albuquerque, NM 87115
Attn: G. P. Tennyson

DOE/ALO
Energy Technology Liaison Office
NGD
Albuquerque, NM 87115
Attn: Capt. J. L. Hanson, USAF

DOE Headquarters (20)
Wind/Oceans Technologies Division
1000 Independence Avenue
Washington, DC 20585
Attn: L. J. Rogers
P. R. Goldman

J. B. Dragt
Nederlands Energy Research Foundation
(E.C.N.)
Physics Department
Westerduinweg 3 Petten (nh)
THE NETHERLANDS

Dynergy Systems Corporation
821 West L Street
Los Banos, CA 93635
Attn: C. Fagundes

Electric Power Research Institute
3412 Hillview Avenue
Palo Alto, CA 94304
Attn: E. Demeo
F. Goodman

Dr. Norman E. Farb
10705 Providence Drive
Villa Park, CA 92667

Alcir de Faro Orlando
Pontificia Universidade Catolica-PUC/Rj
Mechanical Engineering Department
R. Marques de S. Vicente 225
Rio de Janeiro, BRAZIL

FloWind Corporation (2)
1183 Quarry Lane
Pleasonton, CA 94566
Attn: L. Schienbein
B. Im

A. D. Garrad
Garrad Hasson
10 Northampton Square
London EC1M 5PA
UNITED KINGDOM

Gates Learjet
Mid-Continent Airport
P.O. Box 7707
Wichita, KS 67277
Attn: G. D. Park

H. Gerardin
Mechanical Engineering Department
Faculty of Sciences and Engineering
Universite Laval-Quebec, G1K 7P4
CANADA

R. T. Griffiths
University College of Swansea
Dept. of Mechanical Engineering
Singleton Park
Swansea, SA2 8PP
UNITED KINGDOM

Helion, Inc.
Box 445
Brownsville, CA 95919
Attn: J. Park, President

Indal Technologies, Inc. (2)
3570 Hawkestone Road
Mississauga, Ontario
CANADA L5C 2V8
Attn: D. Malcolm
C. Wood

Institut de Recherche d'Hydro-Quebec
1800, Montee Ste-Julie
Varennes, Quebec, JOL 2P.O.
CANADA
Attn: Gaston Beaulieu
Bernard Masse

Iowa State University
Agricultural Engineering, Room 213
Ames, IA 50010
Attn: L. H. Soderholm

K. Jackson
West Wind Industries
P.O. Box 1705
Davis, CA 95617

M. Jackson
McAllester Financial
1816 Summit
W. Lafayette, IN 47906

Kaiser Aluminum and Chemical
Sales, Inc.
14200 Cottage Grove Avenue
Dolton, IL 60419
Attn: A. A. Hagman

Kaiser Aluminum and Chemical
Sales, Inc.
6177 Sunol Blvd.
P.O. Box 877
Pleasonton, CA 94566
Attn: D. D. Doerr

Kansas State University
Electrical Engineering Department
Manhattan, KS 66506
Attn: Dr. G. L. Johnson

R. E. Kelland
The College of Trades and Technology
P.O. Box 1693
Prince Philip Drive
St. John's, Newfoundland, A1C 5P7
CANADA

KW Control Systems, Inc.
RD#4, Box 914C
South Plank Road
Middletown, NY 10940
Attn: R. H. Klein

Kalman Nagy Lehoczky
Cort Adelers GT. 30
Oslo 2, NORWAY

L. K. Liljergren
1260 S.E. Walnut #5
Tustin, CA 92680

L. Liljidahl
Building 005, Room 304
Barc-West
Beltsville, MD 20705

Olle Ljungstrom
FFA, The Aeronautical Research
Institute
Box 11021
S-16111 Bromma, SWEDEN

Robert Lynette
R. Lynette & Assoc., Inc.
15921 SE 46th Way
Bellevue, WA 98006

Massachusetts Institute of Technology
77 Massachusetts Avenue
Cambridge, MA 02139
Attn: Professor N. D. Ham
W. L. Harris, Aero/Astro Dept.

H. S. Matsuda
Composite Materials Laboratory
Pioneering R&D Laboratories
Toray Industries, Inc.
Sonoyama, Otsu, Shiga, JAPAN 520

G. M. McNerney
US Wind Power
160 Wheeler Road
Burlington, MA 01803

Michigan State University
Division of Engineering Research
East Lansing, MI 48825
Attn: O. Krauss

Napier College of Commerce and
Technology
Tutor Librarian, Technology Faculty
Colinton Road
Edinburgh, EH10 5DT
ENGLAND

National Rural Electric
Cooperative Assn
1800 Massachusetts Avenue, NW
Washington, DC 20036
Attn: Wilson Prichett, III

Natural Power, Inc.
New Boston, NH 03070
Attn: Leander Nichols

Northwestern University
Dept. of Civil Engineering
Evanston, IL 60201
Attn: R. A. Parmalee

Ohio State University
Aeronautical and Astronautical Dept.
2070 Neil Avenue
Columbus, OH 43210
Attn: Professor G. Gregorek

Oklahoma State University
Mechanical Engineering Dept.
Stillwater, OK 76074
Attn: D. K. McLaughlin

Oregon State University
Mechanical Engineering Dept.
Corvallis, OR 97331
Attn: R. E. Wilson

Pacific Gas & Electric Co.
3400 Crow Canyon Road
San Ramon, CA 94583
Attn: T. Hillesland

Ion Paraschivoiu
Department of Mechanical Engineering
Ecole Polytecnique
CP 6079
Succursale A
Montreal H3C 3A7
CANADA

Jacques Plante
Hydro Quebec
Place Dupuis Ile etage
855 est rue Ste-Catherine
Montreal, Quebec
CANADA H2L 4P5

The Power Company, Inc.
P.O. Box 221
Genesee Depot, WI 53217
Attn: A. A. Nedd

Power Technologies, Inc.
P.O. Box 1058
Schenectady, NY 12301-1058
Attn: Eric N. Hinrichsen

Public Service Co. of New Hampshire
1000 Elm Street
Manchester, NH 03105
Attn: D. L. C. Frederick

Public Service Company of New Mexico
P.O. Box 2267
Albuquerque, NM 87103
Attn: M. Lechner

RANN, Inc.
260 Sheridan Ave., Suite 414
Palo Alto, CA 94306
Attn: A. J. Eggers, Jr.

Dr. R. Ganesh Rajagopalan, Asst. Prof.
Aerospace Engineering Department
Iowa State University
404 Town Engineering Bldg.
Ames, IA 50011

The Resources Agency
Department of Water Resources
Energy Division
P.O. Box 388
Sacramento, CA 95802
Attn: R. G. Ferreira

Reynolds Metals Company
Mill Products Division
6601 West Broad Street
Richmond, VA 23261
Attn: G. E. Lennox

R. G. Richards
Atlantic Wind Test Site
P.O. Box 189
Tignish P.E.I., COB 2BO
CANADA

Riso National Laboratory
Postbox 49
DK-4000 Roskilde
DENMARK
Attn: Troels Friis Pedersen
Helge Petersen

A. Robb
Memorial University of Newfoundland
Faculty of Engineering and Applied Sciences
St. John's Newfoundland, A1C 5S7
CANADA

Dr. Ing. Hans Ruscheweyh
Institut fur Leichbau
Technische Hochschule Aachen
Wullnerstrasse 7
FEDERAL REPUBLIC OF GERMANY

Beatrice de Saint Louvent
Establissement d'Etudes et de Recherches
Meteorologigues
77 Rue de Serves
92106 Boulogne-Billancourt Cedex
FRANCE

Gwen Schreiner
Librarian
National Atomic Museum
Albuquerque, NM 87185

Arnan Seginer
Professor of Aerodynamics
Technion-Israel Institute of Technology
Department of Aeronautical Engineering
Haifa
ISRAEL

Mr. Farrell Smith Seiler, Editor
Wind Energy Abstracts
P.O. Box 3870
Bozeman, MT 59772-3870

David Sharpe
Dept. of Aeronautical Engineering
Queen Mary College
Mile End Road
London, E1 4NS
UNITED KINGDOM

Kent Smith
Instituto Technologico Costa Rico
Apartado 159 Cartago
COSTA RICA

Solar Energy Research Institute
1617 Cole Boulevard
Golden, CO 80401
Attn: R. W. Thresher

Bent Sorenson
Roskilde University Center
Energy Group, Bldg. 17.2
IMFUFA
P.O. Box 260
DK-400 Roskilde
DENMARK

Peter South
ADECON
32 Rivalda Road
Weston, Ontario, M9M 2M3
CANADA

Southern California Edison
Research & Development Dept., Room 497
P.O. Box 800
Rosemead, CA 91770
Attn: R. L. Scheffler

G. Stacey
The University of Reading
Department of Engineering
Whiteknights, Reading, RG6 2AY
ENGLAND

Stanford University
Dept. of Aeronautics and Astronautics Mechanical Engineering
Stanford, CA 94305
Attn: Holt Ashley

Dr. Derek Taylor
Alternative Energy Group
Walton Hall
Open University
Milton Keynes, MK7 6AA
UNITED KINGDOM

R. J. Templin (3)
Low Speed Aerodynamics Laboratory
NRC-National Aeronautical Establishment
Montreal Road
Ottawa, Ontario, K1A 0R6
CANADA

Texas Tech University (2)
Mechanical Engineering Dept.
P.O. Box 4289
Lubbock, TX 79409
Attn: J. W. Oler

K. J. Touryan
Moriah Research
6200 Plateau Dr.
Englewood, CO 80111

Tulane University
Dept. of Mechanical Engineering
New Orleans, LA 70018
Attn: R. G. Watts

Tumac Industries, Inc.
650 Ford Street
Colorado Springs, CO 80915
Attn: J. R. McConnell

J. M. Turner
Terrestrial Energy Technology
Program Office
Energy Conversion Branch
Aerospace Power Division/
Aero Propulsion Lab
Air Force Systems Command (AFSC)
Wright-Patterson AFB, OH 45433

United Engineers and Constructors, Inc.
P.O. Box 8223
Philadelphia, PA 19101
Attn: A. J. Karalis

Universal Data Systems
5000 Bradford Drive
Huntsville, AL 35805
Attn: C. W. Dodd

University of California
Institute of Geophysics
and Planetary Physics
Riverside, CA 92521
Attn: Dr. P. J. Baum

University of Colorado
Dept. of Aerospace Engineering Sciences
Boulder, CO 80309
Attn: J. D. Fock, Jr.

University of Massachusetts
Mechanical and Aerospace
Engineering Dept.
Amherst, MA 01003
Attn: Dr. D. E. Cromack

University of New Mexico
New Mexico Engineering
Research Institute
Campus P.O. Box 25
Albuquerque, NM 87131
Attn: G. G. Leigh

University of Oklahoma
Aero Engineering Department
Norman, OK 73069
Attn: K. Bergey

University of Sherbrooke
Faculty of Applied Science
Sherbrooke, Quebec, J1K 2R1
CANADA
Attn: A. Laneville
P. Vittecoq

The University of Tennessee
Dept. of Electrical Engineering
Knoxville, TN 37916
Attn: T. W. Reddoch

USDA, Agricultural Research Service
Southwest Great Plains Research Center
Bushland, TX 79012
Attn: Dr. R. N. Clark

Utah Power and Light Co.
51 East Main Street
P.O. Box 277
American Fork, UT 84003
Attn: K. R. Rasmussen

W. A. Vachon
W. A. Vachon & Associates
P.O. Box 149
Manchester, MA 01944

Dirk Vandenberghe
State Univ. of Ghent
St. Pietersniewstraat 41
9000 Ghent
BELGIUM

Washington State University
Dept. of Electrical Engineering
Pullman, WA 99163
Attn: F. K. Bechtel

West Texas State University
Government Depository Library
Number 613
Canyon, TX 79015

West Texas State University
Department of Physics
P.O. Box 248
Canyon, TX 79016
Attn: V. Nelson

West Virginia University
Dept. of Aero Engineering
1062 Kountz Avenue
Morgantown, WV 26505
Attn: R. Walters

D. Westlind
Central Lincoln People's Utility District
2129 North Coast Highway
Newport, OR 97365-1795

Wichita State University
Aero Engineering Department (2)
Wichita, KS 67208
Attn: M. Snyder
W. Wentz

Wind Power Digest
P.O. Box 700
Bascom, OH 44809
Attn: Michael Evans

Wisconsin Division of State Energy
8th Floor
101 South Webster Street
Madison, WI 53702
Attn: Wind Program Manager

| | |
|---|---|
| 1520 | C. W. Petersson |
| 1522 | R. C. Reuter, Jr. |
| 1523 | J. H. Biffle |
| 1524 | A. K. Miller |
| 1524 | D. W. Lobitz |
| 1550 | R. C. Maydew |
| 1556 | G. F. Homicz |
| 2525 | R. P. Clark |
| 3141-1 | S. A. Landenberger (5) |
| 3151 | W. L. Garner (3) |
| 3154-3 | C. H. Dalin (28) For DOE/OSTI (Unlimited Release) |
| 3160 | J. E. Mitchell (15) |
| 3161 | P. S. Wilson |
| 6000 | D. L. Hartley |
| 6200 | V. L. Dugan |
| 6220 | D. G. Schueler |
| 6225 | H. M. Dodd (50) |
| 6225 | T. D. Ashwill |
| 6225 | D. E. Berg |
| 6225 | T. C. Bryant |
| 6225 | L. R. Gallo |
| 6225 | P. C. Klimas |
| 6225 | S. D. Nicolaysen |
| 6225 | D. S. Oscar |
| 6225 | M. E. Ralph |
| 6225 | D. C. Reda |
| 6225 | M. A. Rumsey |
| 6225 | L. L. Schluter |
| 6225 | W. A. Stephenson |
| 6225 | H. J. Sutherland |
| 7111 | J. W. Reed |
| 7544 | D. R. Schafer |
| 7544 | T. G. Carne |
| 7544 | J. Lauffer |
| 8024 | P. W. Dean |
| 9100 | R. G. Clem |
| 9122 | T. M. Leonard |