1989-12

Computer graphics adaptation of several aerodynamic prediction programs

MacAllister, Craig M.
Monterey, California. Naval Postgraduate School

https://hdl.handle.net/10945/25766

This publication is a work of the U.S. Government as defined in Title 17, United States Code, Section 101. Copyright protection is not available for this work in the United States.
COMPUTER GRAPHICS ADAPTATION
OF SEVERAL AERODYNAMIC
PREDICTION PROGRAMS

by

Craig M. MacAllister

December 1989

Thesis Advisor: J. V. Healey

Approved for public release: distribution is unlimited
Computer Graphics Adaptation of Several Aerodynamic Prediction Programs

Craig M. MacAllister

This thesis describes the adaptation of six computer programs on the Micro VAX/2000/CAD/CAE workstation. Three of the programs, NEW_DOUBLE, NEW_PANEL, and NEW_VOR, were originally transferred to the Aeronautical Engineering VAX System Server by LCDR John Campbell. Two of the programs (SUB and SUPER), both vortex lattice method programs, were placed in the VAX system by Mr. Rich Margason of the Langley Research Center. The sixth program, a viscous interaction program was transferred/adapted to the VAX system by the author of this report. Extensive modifications were made to these programs to enhance their user interface. In addition, each program has been adapted to provide interactive graphical/printed output. Furthermore, program NEW_DOUBLE was modified to accept any arbitrary symmetrical shaped body. Lastly, NEW_PANEL was altered to interface with the viscous interaction program in which boundary layer characteristics were determined. All user inputs in NEW_DOUBLE, NEW_PANEL and NEW_VOR were also backed up with interactive checking routines. The programs were intended to be used by aeronautics/astronautics engineering students in basic and advanced courses in aerodynamics.
Approved for public release; distribution is unlimited.

Computer Graphics Adaptation of Several Aerodynamic Prediction Programs

by

Craig M. MacAllister
Captain, United States Army
B.S., United States Military Academy, 1979

Submitted in partial fulfillment of the requirements for the degree of

AERONAUTICAL ENGINEER

from the

NAVAL POSTGRADUATE SCHOOL
December 1989
ABSTRACT

This thesis describes the modification of six computer programs on the Micro VAX/2000/CAD/CAE workstation. Three of the programs, NEW_DOUBLE, NEW_PANEL, and NEW_VOR, were originally transferred to the Aeronautical Engineering VAX System Server by LCDR John Campbell. Two of the programs (SUB and SUPER), both vortex lattice method programs, were placed in the VAX system by Mr. Rich Margason of the Langley Research Center. None of the above five programs had any graphics facility. The sixth program, a viscous interaction program was transferred/adapted to the VAX system by the author of this report. Extensive modifications were subsequently made to these programs to enhance their user interface. In addition, all the programs have been adapted to provide interactive graphical/printed output. Furthermore, program NEW_DOUBLE was modified to accept any arbitrary symmetrical shaped body. Lastly, NEW_PANEL was altered to interface with a viscous interaction effects program in which the boundary layer characteristics are determined. All user inputs in NEW_DOUBLE, NEW_PANEL and NEW_VOR were backed up with interactive checking routines. The programs are intended to be used by aeronautics/astronautics engineering students in basic and advanced courses in aerodynamics.
THESIS DISCLAIMER

The reader is cautioned that computer programs developed in this research may not have been exercised for all cases of interest. While every effort has been made, within the time available, to ensure that the programs are free of computational and logic errors, they cannot be considered validated. Any application of these programs without additional verification is at the risk of the user.
# TABLE OF CONTENTS

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

II. PROGRAM ADAPTATION ............................................................................................................ 3
   A. INTRODUCTION .................................................................................................................. 3
   B. PROGRAM NEW_DOUBLE ADAPTATIONS ................................................................. 4
   C. PROGRAM NEW_PANEL ADAPTATIONS ..................................................................... 6
   D. PROGRAM NEW_VOR ADAPTATIONS ........................................................................... 8
   E. PROGRAM “SUB” ADAPTATIONS ................................................................................. 9
   F. PROGRAM “SUPER” ADAPTATIONS ........................................................................... 11

III. SOLUTION FOR THE TWO-DIMENSIONAL INCOMPRESSIBLE
     LAMINAR AND TURBULENT BOUNDARY LAYER PROBLEM -------------- 13
   A. INTRODUCTION ............................................................................................................. 13
   B. NUMERICAL SOLUTION BASIS .................................................................................. 14
   C. COMPUTER PROGRAM NUMERICAL SOLUTION ...................................................... 14

IV. USER’S MANUAL ....................................................................................................................... 17

V. PROGRAM COMPUTER CODES ................................................................................................ 18

VI. RESULTS AND RECOMMENDATIONS ..................................................................................... 19

APPENDIX A. PROGRAM NEW_DOUBLE USER’S MANUAL -------------------------- 21
APPENDIX B. PROGRAM NEW_PANEL USER’S MANUAL ----------------------------- 35
APPENDIX C. PROGRAM NEW_VOR USER’S MANUAL ------------------------------- 51
APPENDIX D. PROGRAM SUB USER’S MANUAL ............................................................... 61
APPENDIX E. PROGRAM SUPER USER’S MANUAL ......................................................... 80
APPENDIX F. PROGRAM NEW_DOUBLE COMPUTER CODE ---------------------------- 95
APPENDIX G. PROGRAM NEW_PANEL COMPUTER CODE -------------- 120
APPENDIX H. PROGRAM NEW_VOR COMPUTER CODE --------------- 163
APPENDIX I. PROGRAM SUB COMPUTER CODE ------------------ 177
APPENDIX J. PROGRAM SUPER COMPUTER CODE ----------------- 218
LIST OF REFERENCES ---------------------------------------- 251
INITIAL DISTRIBUTION LIST ------------------------------- 253
<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>Rectangular Net Orientation, Keller's Box</td>
<td>15</td>
</tr>
<tr>
<td>2.</td>
<td>Initial Screen for Program NEW_DOUBLE</td>
<td>24</td>
</tr>
<tr>
<td>3.</td>
<td>Endpoint Determination Method Selection Screen</td>
<td>25</td>
</tr>
<tr>
<td>4.</td>
<td>Plotting Options Screen</td>
<td>28</td>
</tr>
<tr>
<td>5.</td>
<td>Symmetric Shape Data Point Input Screen</td>
<td>30</td>
</tr>
<tr>
<td>6.</td>
<td>Doublet Strength Distribution</td>
<td>32</td>
</tr>
<tr>
<td>7.</td>
<td>Cp Distribution</td>
<td>33</td>
</tr>
<tr>
<td>8.</td>
<td>Airfoil Shape</td>
<td>34</td>
</tr>
<tr>
<td>9.</td>
<td>Initial Screen for Program NEW_PANEL</td>
<td>39</td>
</tr>
<tr>
<td>10.</td>
<td>Screen Showing Data for Computer Generated Airfoil</td>
<td>41</td>
</tr>
<tr>
<td>11.</td>
<td>Printing Option Screen</td>
<td>41</td>
</tr>
<tr>
<td>12.</td>
<td>Graphical Selection Screen</td>
<td>42</td>
</tr>
<tr>
<td>13.</td>
<td>Menu for Surface Coordinate Data Entry Method</td>
<td>43</td>
</tr>
<tr>
<td>14.</td>
<td>Menu for Viscous Data Input Option</td>
<td>45</td>
</tr>
<tr>
<td>15.</td>
<td>Viscous Data Output File Option Screen</td>
<td>46</td>
</tr>
<tr>
<td>16.</td>
<td>Cp Distribution</td>
<td>48</td>
</tr>
<tr>
<td>17.</td>
<td>Body Shape</td>
<td>49</td>
</tr>
<tr>
<td>18.</td>
<td>Cl &amp; Cm c/4 vs. Alpha</td>
<td>50</td>
</tr>
<tr>
<td>19.</td>
<td>Initial Screen for Program NEW_VOR</td>
<td>54</td>
</tr>
<tr>
<td>20.</td>
<td>Data Review/Correction Screen</td>
<td>55</td>
</tr>
</tbody>
</table>
21. Graphical Selection Screen ................................................. 56
22. Cl vs. Y ........................................................................... 58
23. Cd vs. Y ........................................................................... 59
24. Cl vs. Cd .......................................................................... 60
25. Initial Screen for Program SUB ........................................ 68
26. Printing Determination Screen ........................................... 68
27. Output File Designation Screen ........................................ 69
28. Plot Determination Screen ............................................... 70
29. Induced Drag Coeff. vs. 2Y/B ........................................... 74
30. LE Thrust Coeff. vs. 2Y/B ................................................ 75
31. Suction Coeff. vs. 2Y/B .................................................... 76
32. Span Load Coeff. vs. 2Y/B ................................................ 77
33. Coeff. of Lift Ratio vs. 2Y/B ............................................. 78
34. Delta Cp vs. X c/4 ............................................................ 79
35. Initial Screen for Program SUPER ..................................... 86
36. Printing Determination Screen ........................................... 86
37. Output File Designation Screen ........................................ 87
38. Plot Determination Screen ............................................... 88
39. Spanwise Cp Distribution ................................................ 91
40. Chordwise Cp Distribution .............................................. 92
41. Drag Polar ....................................................................... 93
42. Spanwise Lift Distribution ............................................... 94
ACKNOWLEDGEMENTS

First and foremost, I would like to thank the members of my family for their unwavering support of my thesis efforts. Secondly, I would like to thank Professor Val Healey, who provided me with the topic for this thesis and who helped me to understand the underlying principles of computational fluid dynamics. Additionally, I would like to thank both Professor Healey and Professor Platzer for affording me several opportunities to have their students test and help in the development of these computer programs. The comments of their students greatly assisted me in improving the user interface characteristics of these programs. Furthermore, I would like to thank Mr. Edward Ward, Ms. Donna Byrch and Mrs. Karen Yates for transferring my FORTRAN files from my VAX lab account to my IBM mainframe account. Their efforts greatly facilitated the final editing of my programs. Lastly, a special note of thanks goes to Mr. Tony Cricelli and Mr. Dave Marco for the benefit of their technical expertise in using the VAX system.
I. INTRODUCTION

Incorporated in this thesis are six FORTRAN programs which have been extensively modified to improve their user interface and to enhance their output capabilities. Three of the programs, NEW_PANEL, NEW_DOUBLE, and NEW_VOR were originally transferred to the Aeronautical Engineering CAD/CAE Lab by LCDR John Campbell. Two of the programs, SUB and SUPER, vortex lattice programs, were developed by NASA AMES and transferred to the CAD/CAE Lab by Mr. Rich Margason. The sixth program, written by Dr. Cebeci, was transferred/adapted by the author of this thesis.

The main focus of this thesis was to adapt the above mentioned programs with a graphic facility. The graphics base program was developed by Mr. Dave Marco of the Mechanical Engineering Department, Naval Postgraduate School. The base program is simply a compilation of FORTRAN subroutines (similar to the popular graphics program DISSPLA) which can be called to effect 2-D graphics. As such, the graphics produced by the aforementioned Computational Fluid Dynamics (CFD) programs are limited to two dimensions. All of the plots generated by the respective program adaptations are effected interactively with little or no user input.

Additionally, the respective program modifications were primarily intended to enhance the user interface with the CFD programs and, in a sense, effectively streamline their use. Specifically, adaptations such as input checking, interactive menus, automatic sorting routines, and backup output data file creation were integrated into each program. To supplement this thesis
objective, a concise User's Manual was created to provide the aeronautical engineering student a program reference guide. The user's manual, in fact, details all functional aspects of these programs and provides computational examples supplemented with graphical/tabular results. The User's Manual was geared to the student with little or no experience with the VAX system.

Furthermore, the programs NEW_DOUBLE and NEW_PANEL each received specific adaptations which were not originally within the scope of this thesis. Specifically, the NEW_DOUBLE program, a line doublet distribution program, was adapted to input any symmetrical body. Originally, the NEW_DOUBLE program could only consider an elliptic or a one-family symmetrical airfoil-like shape; this family is described by the equation:

\[ Y(x) = A \sqrt{\frac{x}{c}} (c-x) \]

The NEW PANEL program was adapted to interactively process a coefficient of pressure (Cp) distribution to determine boundary layer characteristics. The source of the Cp distribution to be analyzed can be either from the NEW PANEL program itself or an arbitrary Cp distribution which can be entered from an input data file.

Thesis results and recommendations for future work are given.
II. PROGRAM ADAPTATION

A. INTRODUCTION

Computer programs can always be improved or enhanced. New programming techniques coupled with improved programming languages provide a limitless number of programming innovations which can be initiated. Additionally, as technological advances occur in the realm of computer hardware, it is without question that software development will also be enhanced.

The first major modification made to all the programs considered in this thesis was to totally restructure and streamline each program to facilitate editing and compiling. As each program was modified, it became excessively large; the main program coupled with all of its subroutines to include the new graphics subroutines invariably exceeded the buffer size of the VAX 2000 Workstation. The base program and each of its subroutines were placed in a separate FORTRAN file. Each FORTRAN subroutine was then compiled as a single entity to create its own object file. The object files of the subroutines were then consolidated into a library file specifically named to support its base program. Prior to running a particular program, the base program object file, the library of subroutine object files and the object file for the graphics program were linked together to create a single executable file.

All of the programs, with the exception of the viscous effects program, were adapted to output 2-D plots. As mentioned in the introduction, the graphics program was created by Mr. Dave Marco and, in its present state, is called DLIB. Again, DLIB is a compilation of graphics subroutines. The version
of DLIB used in this thesis is called DISL and represents a small alteration of the original to enhance its interface with the graphics subroutines. Specifically, the DLIB requirement to enter the command of "CONTINUE" once a graphics image is presented on the screen was deleted. The original intent of this requirement was to ensure that the graphics image would not be erased from the screen until the user decided to continue. However, the graphics image created by a FORTRAN subroutine will not disappear until the program execution is terminated. Thus, this command requirement (CONTINUE) was not needed and presented a point of confusion for program users.

The procedure to effect the graphical plots in each program differed to a degree. The principal difference lay in the method in which data arrays were read into the graphics subroutine. However, the use of backup data files was common to all programs. These files not only facilitated the development of the graphics but also acted as a data checking vehicle. The use of common blocks to transfer data arrays between subroutines was kept to a minimum for simplicity's sake. Automatic scaling of data is not standardized across all of the CFD programs. Varying techniques were needed to preclude data array distortion. However, each graphics subroutine contains a FORTRAN "CALL" statement to effect the automatic scaling routine which determined the maximum and minimum values of the data arrays to be plotted.

B. PROGRAM NEW_DOUBLE ADAPTATIONS

The purpose of the NEW_DOUBLE program is to determine the piecewise constant doublet strength, \( m(t) \), for a line doublet distribution of an elliptic or symmetrical airfoil-like shapes at zero angle of attack. The points \( t_i \), represent the location of the doublets along the chord or line of symmetry. They are
concentrated near the ends of the distribution, using a cosine spacing method, where the variation of the doublet strength is expected to be most rapid. The point \( t_1 \) corresponds to \( x_s \) and \( t_N \) corresponds to the endpoint \( x_f \).

The stream function can be calculated from the doublet strength distribution. From the stream function, the velocity components and the pressure coefficients are then calculated. The surface shape is defined by \( y = Y(x) \) and the solution must satisfy zero velocity conditions at the leading and trailing edge stagnation points.

In addition to adding graphics subroutines to this program, NEW_DOUBLE was adapted to analyze any symmetric shape. The user is first required to interactively enter the respective data points for the top portion of the symmetric shape. Once all of the points have been entered, the program will allow the user to correct any mistakes he or she may have made while entering the data. Using a spline routine, the intermediate points along the symmetric shape can be obtained readily to facilitate program processing [Ref. 1]. In brief, the spline routine created a continuous function between each adjacent data point.

The NEW_DOUBLE graphics subroutines (GRAPH1, GRAPH2, GRAPH3) presented several unique features. First of all, the automatic scaling routines are specific to the particular type of plot to be created. Specifically, three automatic scaling routines were created: FIX, SCALER, SCALER2. Each of these routines determined the maximum and the minimum value of the specific array to be plotted. Another unique feature common to the NEW_DOUBLE graphics subroutines is that they were designed to read the data arrays to be plotted from dummy data files which were established in the computational
subroutines. This technique facilitated the data checking capability of the program and ensured accurate plots. Lastly, all of the plots were created to produce explanatory remarks to enhance the user's capability to relate the program inputs to the graphical outputs. The graphics subroutines included common blocks which enabled the plots to display user interactive inputs such as the thickness ratio, the maximum thickness, and the number of intervals specified. Typical plots obtained from the NEW_DOUBLE program are provided in Appendix A. [Ref. 2 and 3]

C. PROGRAM NEW_PANEL ADAPTATIONS

The purpose of the original PANEL program was to provide an analysis of the aerodynamics of NACA four-digit airfoils and airfoils of the NACA 230XX family using the source panel method. The program has been modified to accept arbitrary airfoil surface coordinate input and is limited to single-element airfoils. The solution is determined for conditions of incompressible and inviscid uniform free-stream flow. The very small coefficient of drag provided in the results is due to numerical round-off error. Furthermore, NEW_PANEL has also been adapted to analyze viscous effects. When considering the viscous analysis loop of the program, it is important to understand that the Cebeci program adaptation is sensitive to flow separation on the airfoil. Boundary layer thickness and other boundary layer characteristics are computed and outputted into a tabular format.

The most dramatic modification made to the NEW_PANEL program was the adaptation of the program to consider viscous effects. The first step in making this modification was to transfer the Cebeci program to a CAD/CAE lab account. The original version of this program was provided by Dr. M. F.
Platzer, Aeronautics/Astronautics Department, Naval Postgraduate School (NPS). The program, written in FORTRAN, had already been adapted for an IBM PC but the user interface with this program was extremely poor and formal instructions on its use did not exist. The program was subsequently manually transferred to a VAX lab account. After an inordinate amount of error checking, the program was validated against a report offered by Dr. Platzer. The Cebeci program was then modified to enhance its user interface by incorporating interactive input requests to include printing options and input source selection. In addition, a common “bubble-sort” FORTRAN routine was added to the Cebeci program to automatically determine the stagnation point on the airfoil [Ref. 4]. The original version of this program required the user to specify this point. Furthermore, the user is required to specify the point at which laminar-turbulent transition occurs on both the top and the bottom of the airfoil as well as the stream-flow Reynolds number. Finally, the Cebeci program was fully integrated with the NEW PANEL program. The Cp distribution created by the NEW PANEL method was then interactively sorted, scaled, and inputted into the Cebeci program. This program, as noted above, then computed and outputted the respective boundary-layer characteristics. In addition to the Cp distribution created by NEW PANEL program, the user can also enter any arbitrary Cp distribution from a data file called BL2D.DAT. This last option allows the user to in effect, conduct viscous effects calculations while not being limited by the program restrictions of NEW PANEL.

The NEW PANEL graphics subroutines (GRAF1, GRAF2) presented several unique features. First of all, the automatic scaling routines are specific to the particular type of plot to be created. Specifically, two automatic scaling
routines were created: FORM1, FORM2. Each of these routines determined the maximum and the minimum value of the specific array to be plotted. Like NEW_DOUBLE, NEW_PANEL graphics subroutines were designed to read the data arrays to be plotted from dummy data files which were established in the computational subroutines. Again, all of the plots were created to produce explanatory remarks to enhance the user's capability to relate program inputs to graphical outputs. The graphics subroutines included common blocks which enabled the plots to display user interactive inputs such as the number of panels, the angle of attack, and the NACA airfoil number. Another unique feature of the NEW_PANEL graphics subroutines is the addition of the capability to produce two graphs which were not within the scope of the original NEW_PANEL program. Specifically, the relationships of Cm c/4 versus angle of attack and Cl versus angle of attack can be plotted. This adaptation was realized by causing the NEW_PANEL program to perform the NEW_PANEL analysis at 2 degree increments in angle of attack from -8 degrees to 16 degrees. Typical plots obtained from the NEW_PANEL program are provided in Appendix B. [Refs. 2 and 3]

D. PROGRAM NEW_VOR ADAPTATIONS

The purpose of the NEW_VOR program is to provide an application of the vortex lattice method for the determination of the lift distribution of a flat rectangular wing. This method is based on a distribution of discrete horseshoe vortices over a wing surface that has been divided into a finite number of panels. A system of linear equations is developed for the vortex strengths on the panels and solved by matrix methods.
In addition to adapting the NEW_VOR program for printing/graphics options, this program was also modified with a unique subroutine to effect the automatic scaling function. Rather than creating a separate/unique scaling subroutine for each graphical output, a single subroutine (called MAXMIN) was developed to sort the designated array. The MAXMIN subroutine was designed to output the maximum and minimum value in the array, and the particular array in ascending order. In that the array to be plotted was outputted in ascending order, it was necessary to establish a dummy array in each respective graphics subroutine which would be plotted. Otherwise, the plots would invariably ascend from left to right. Again, each plot was adapted to present user inputs. Specifically, the values for aspect ratio and angle of attack are displayed. Typical plots obtained from the NEW_VOR program are provided in Appendix C. [Ref. 2]

E. PROGRAM "SUB" ADAPTATIONS

The SUB program has been adapted from a National Aeronautics and Space Administration (NASA) FORTRAN program which has been used considerably at the Langley Research Center and in industry. The results have shown good correlation with experimental results. SUB has subsequently been revised to enhance its ease of use and its ability to present accurate graphical results. This particular program has also undergone extensive student evaluations. An AE 2035 class of 14 students thoroughly tested and evaluated the majority of the functions which this particular program offers. As a result of their findings, numerous modifications were made to the program SUB as will be detailed below.
The purpose of the SUB program is to estimate the subsonic aerodynamic characteristics of complex planforms. The program represents a lifting planform with a vortex lattice. A relatively complex planform may be analyzed using up to 24 line segments on a semispan. Additionally, these line segments may have an outboard variable-sweep panel or they may have several dihedral angles across the span. Furthermore, two planforms may be used together to represent a combination of wings and tails or wing, bodies, and tails.

The SUB graphics subroutines (GRAPH1, GRAPH2, GRAPH3) present several unique features. First of all, automatic scaling is again effected by the MAXMIN routine. Secondly, dummy data files were established in the computational subroutines and subsequently read in each graphics subroutine. The use of common blocks was kept to a minimum. The coefficient of pressure data provided by the SUB program lends itself readily to 3-D graphics. However, in the absence of a 3-D graphics program in the CAD/CAE Lab, the program was modified to locate the data at the user specified planform position. Specifically, a sorting routine was developed to allow the user to specify a particular spanwise position on the planform to analyze the Cp distribution across the chord of the planform. In order to create this sorting routine, it was necessary to adapt the data output to the finite difference nodal network. This was simply done by realizing the constant spacing distances between the nodal points (stations). Typical plots obtained from the SUB program are provided in Appendix D. [Ref. 5]

Lastly, the SUB program was modified to provide the user the opportunity to copy the output data file into an alternate data file so that his or her results would be saved for further analysis. Subsequent runs of the program
could then be made without losing the results already determined. This modification was effected through interactively allowing the user to select an alternate data file from a list of four files.

F. PROGRAM "SUPER" ADAPTATIONS

The SUPER program has also been adapted from a National Aeronautics and Space Administration (NASA) FORTRAN program and has been used considerably at the Langley Research Center. The use of this program is confined to the supersonic flow regime. In addition, the linearized supersonic lifting surface theory, used in this program, applies to wings having negligible thickness. SUPER has subsequently been revised to enhance its ease of use and its ability to present accurate graphical plots. These graphical representations have been verified with NASA reports as referenced below.

The purpose of the SUPER program is to estimate the supersonic aerodynamic characteristics of complex planforms. Linearized supersonic lifting surface theory is employed to calculate the aerodynamic characteristics of a warped wing of arbitrary planform. The program calculates lifting pressure distribution for the warped wing at fixed attitude and the pressure distribution (per degree angle of attack) for a corresponding flat wing. These two pressure distributions are combined by superposition principles and integrated over the wing surface to obtain the variation of aerodynamic characteristics with changes in angle of attack.

Similar to the case of program SUB, complex sorting routines were developed to allow the user to specify the respective chordwise or spanwise position on the supersonic planform which would be analyzed for plotting purposes. The coefficient of pressure data provided by the SUPER program also lends
itself quite readily to 3-D graphics. Again, in the absence of a 3-D graphics program, the program was modified to locate the data at the user specified planform position. In order to create these sorting routines it was necessary to adapt the data output to the finite difference nodal network. Like program SUB, this was done by realizing the constant spacing distances between the nodal points and subsequently sorting the data accordingly to isolate the requested data. Typical plots obtained from the SUPER program are provided in Appendix E. [Refs. 6 and 7]

Another modification made to the SUPER program concerns its output data file. The output data file for the SUPER program is extremely long. The great length of this file was of negligible utility to the common user. A new output data file was created within the text of the program which simply outputted the input data and the aerodynamic results. The tabular coefficient of pressure data was not incorporated into the output file. However, the full output data file with complete Cp data is still written to a file called "OUTER.DAT". The abbreviated output file (OUTFILE.DAT) greatly facilitates the printing of the output file for the user.

Lastly, like SUB, the SUPER program was modified to provide the user the opportunity to copy the output data file into an alternate data file so that his or her results would be saved for further analysis. The user is given the opportunity to interactively select an alternate data file name in which he or she can store their computational results.
III. SOLUTION FOR THE TWO-DIMENSIONAL INCOMPRESSIBLE LAMINAR AND TURBULENT BOUNDARY LAYER PROBLEM

A. INTRODUCTION

This section relates the numerical methods employed to solve the two dimensional incompressible laminar and turbulent boundary layer problem as conceived by Dr. T. Cebeci and Dr. H. B. Keller. As discussed earlier, this particular boundary-layer solution method was modified and imbedded into the NEW_PANEL program. The intent of this section is to provide a brief synopsis of their problem solution, not a detailed account. The development of the specific theoretical basis/computer code development of the Cebeci program is not within the scope of this thesis [Ref. 8]. In order to use Dr. Cebeci’s method, it is necessary to input the potential flow solution over a section shape. In particular, the Cp distribution or the velocity distribution is required. Such information is obtained quite readily through the execution of the NEW_PANEL program. In fact, the Cp distribution is interactively sorted and inputted to the Cebeci program upon the user’s request. In addition, one of the functional capabilities of the NEW_PANEL program is to input an arbitrary velocity distribution. Furthermore, the Cebeci program version provided by Dr. M. F. Platzer was further revised to determine the coefficients for skin friction drag and form drag from the computed boundary-layer characteristics. This additional capability was transferred from the original version which is currently available for use on the IBM mainframe at the Naval Postgraduate School, account 4632P.
B. NUMERICAL SOLUTION BASIS

This program uses a finite-difference method to solve the partial differential equation obtained by using the Falkner-Skan transformation of the general boundary layer equations. Both laminar and turbulent flows may be analyzed in that an eddy-viscosity concept has been incorporated into the program which allows the momentum equation for turbulent flows to be written in the same form as a laminar flow. Dr. Cebeci's method is valid except upon the evolution of flow separation. The boundary layer separation point corresponds to the vanishing of the wall shear force at that point. Dr. Cebeci [Ref. 8] states, "if the wall shear vanishes at some x-location during the solution procedure, the solutions break down and convergence cannot be obtained. This is sometimes referred to as the singular behavior of the wall shear close to the separation point." Close inspection of the boundary layer results provided by the NEW_PANEL program is advised in order to ensure that the results are in fact valid. Extremely large values of displacement or momentum thickness indicate flow separation on the shape being analyzed. As an additional note, the program is limited to two dimensions in that negligible transverse curvature has been assumed.

C. COMPUTER PROGRAM NUMERICAL SOLUTION

There exist several methods to solve the boundary-layer equations. The finite difference method used in this program was developed by Dr. H. B. Keller [Ref. 9]. Keller's box method has been used extensively to solve the boundary-layer equations. The first requirement to be effected before the Keller method can be employed is to rewrite the governing equations as a first order system. The resulting first-order equations are subsequently
approximated on an arbitrary rectangular net. The finite-difference equations evolve from "centered-difference" derivatives and averaged at the midpoints of the net rectangle. Figure 1 represents the orientation of the net rectangle. The respective nodal points are determined by:

\[ E_0 = 0, \quad E_n = E_{n-1} + k_n, \quad n = 1,2,3...N \]
\[ n_0 = 0, \quad n_j = n_{j-1} + h_j, \quad j = 1,2,3...J \]

Figure 1. Rectangular Net Orientation, Keller's Box

Solutions of the finite difference equations yield a truncation error of the second order. The difference equations are subsequently linearized by Newton's method [Ref. 9]. Finally, the equations are solved by a block-elimination method [Ref. 8].

The computer program has been broken down into several separate subroutine programs labeled as CIB, COEF, BL, EDDY, SOLV3, OUTPUT, and DRAG. Subroutine CIB is called from the NEW_PANEL main program which in turn
calls the remaining subroutines in order to determine the requisite boundary-layer characteristics. The flow transition point (laminar to turbulent) is interactively inputted by the user for both the upper and lower surfaces of the airfoil. The user is also prompted to enter the chord-based Reynolds number.
IV. USER'S MANUAL

In order to facilitate the use of the programs addressed in this thesis, a User's Manual was created which expanded upon the one made by LCDR John Campbell [Ref. 1]. Appendices A through E of this thesis contain the text portions of the User's Manual as well as representative graphical outputs. The User's Manual in its final form is approximately 150 pages long, excluding section and sample problem dividers. The bulk of the User's Manual consists of the output data files, the input data files (if appropriate) and the graphical outputs for each sample problem referenced in the User's Manual. These output files/plots served as the primary basis for validating the graphical plots obtained by each respective program. As an additional note, the User's Manual was not included in this thesis in its entirety due to its extreme length.
V. PROGRAM COMPUTER CODES

Appendices F through J contain the complete source codes of the programs discussed in this thesis. These codes are in their final form. However, it should be noted that each subroutine/main program has been written and presented as a stand-alone FORTRAN program in that each program can be compiled individually. As noted earlier, once each subroutine was compiled, an "object" file was created by the computer. This "object" file was then placed in its respective program library (DOUBLIB, PANLIB, VORLIB, SUBLIB, SUPLIB). Prior to running the particular program, the library of "object" files was linked with the object file of the main program (NEW_DOUBLE, NEW_PANEL, NEW_VOR, SUB, SUPER) and with the object file of the graphics program (DISL). Linking the files in this manner created a single executable file for each program.

There exists several lines of FORTRAN code in each individual program which are currently not executable (comment lines). Some of these comment lines will facilitate future modifications; others represent routines which were specifically incorporated for data checking; and the remainder are simply comments to clarify the executable statements. Rather than deleting these lines as being extraneous, they were left in the program to facilitate future modifications/program maintenance. These routines are marked appropriately within the program.
VI. RESULTS AND RECOMMENDATIONS

The objectives of this thesis, as originally conceived, have been realized. Five FORTRAN programs (NEW_DOUBLE, NEW_PANEL, NEW_VOR, SUB, SUPER) have been modified to interactively supply graphical representation of the respective computational results. In fact, SUB and SUPER were added to the list of programs to be modified well into the thesis research process. In addition, NEW_DOUBLE can now analyze any symmetrical shape. Data checking routines were also added to NEW_DOUBLE to enhance data input procedures. Furthermore, the user interface capabilities of each program were significantly improved, especially for SUB and SUPER. Lastly, each program was adapted to provide the user the capability to interactively print the computational results or the plots developed.

To enhance the utility of these programs, a concise User's Manual was developed. This manual fully describes each program to include program and input restrictions. In addition, numerous sample problems were integrated into the manual to demonstrate to the user the various capabilities of each program. For each sample problem, detailed instructions are given on how to use the program properly. Furthermore, the output data files and graphical plots created by each sample problem are also included in the User's Manual.

The validation of the graphical results was achieved through several sources; in particular, the books by Ira H. Abbott and A.E. VonDoenhoff [Ref. 10] and by John D. Anderson, Jr. [Ref. 11], were used to check the plots generated by NEW_DOUBLE and NEW_PANEL. LCDR J.A. Campbell's thesis
results were used to validate the NEW_VOR plots. In order to check the SUB and SUPER graphs, the NASA publications detailing each respective program were used [Refs. 5, 6, and 7]. In all cases, the graphical representations produced by these programs were qualitatively and quantitatively correct. No arbitrary adjustments were made to the graphics subroutines to "fit" the data to the respective validating source.

Modifications and further adjustments can always be made to a computer program to either enhance or expand its capabilities. As stated earlier, the tabular output of data in SUB and SUPER readily lends itself to 3-D graphics display. At such time that the Aeronautics/Astronautics Department CAD/CAE lab receives a 3-D general graphics package, such as DISSPLA, SUB and SUPER can easily be adapted to produce 3-D plots. The graphics subroutines, as they are currently written, use call statements identical to those used with a DISSPLA package. Furthermore, the data generation required for the respective 3-D plots has already been programmed into the graphics subroutines. An additional modification would be to adapt the Cebeci program output to produce graphical results. Furthermore, the Cebeci program could also be modified to solve a variety of problems including 2-D flows with heat and mass transfer, slot injection as well as axisymmetric flows. Lastly, the programs SUB and SUPER could be adapted to interactively accept the data inputs from the console rather than requiring the user to create an input data file. However, the inputs to both programs can be rather long and detailed in the analysis of a complex planform.
APPENDIX A
PROGRAM NEW_DOUBLE USER'S MANUAL

USER'S GUIDE CONTENTS

I. INTRODUCTION .......................................................... 22
II. ASSUMPTIONS AND LIMITATIONS .................................. 22
III. INPUT DESCRIPTION .................................................... 23
IV. SAMPLE PROBLEMS ..................................................... 23
V. STARTING THE PROGRAM .............................................. 24
VI. SAMPLE GRAPHICAL OUTPUTS ....................................... 25
I. INTRODUCTION

The purpose of the NEW_DOUBLE program is to determine the piecewise constant doublet strength $m(t)$ for a line doublet distribution of an elliptic or airfoil-like shape at zero angle of attack. The points $t_i$, represent the location of the doublets along the chord or line of symmetry. They are concentrated near the ends of the distribution, using a cosine spacing method, where the variation of the doublet strength is expected to be most rapid. The point $t_1$ corresponds to $x_s$ and $t_N$ corresponds to the endpoint $x_f$.

The stream function can be calculated from the doublet strength distribution. From the stream function, the velocity components and the pressure coefficients may be calculated. The surface shape is defined by $y=Y(x)$ and the solution must satisfy the zero velocity conditions at the leading and trailing edge stagnation points.

II. ASSUMPTIONS AND LIMITATIONS

The approach taken to develop this method of solution assumes that the doublet strength functions are both piecewise-constant along the chord. It is also important to remember that this solution is valid for incompressible and inviscid uniform freestream flow. Since the bodies under investigation are (two dimensional) symmetrical and at zero angle of attack, there is no lift nor induced drag produced. In addition, there is no drag since we are considering an inviscid fluid and no separation is allowed for.
III. INPUT DESCRIPTION

There are very few input values required for this simple program. Their description and program variable names are listed below.

NTYPE—Type of body shape; elliptic, a single-family airfoil-like, given by
\[ Y(x) = A \sqrt{\frac{x}{c}} (c-x) \], or symmetric.

TAU—Thicknness ratio. (Maximum thickness/chord)

XMAXY—Chordwise location of the point of maximum thickness. (Airfoil only)

N—Number of intervals. \(2 \leq N \leq 100\)

XS—Doublet distribution starting point.

XF—Doublet distribution ending point.

NXTOL—Exponent value used to generate the convergence criterion XTOL.

NFTOL—Exponent value used to generate the convergence criterion FTOL.

XTOL—X location tolerance.

FTOL—Function tolerance.

X-X Coordinate of the symmetric shape airfoil surface.

Y-Y Coordinate of the symmetric shape airfoil surface.

IV. SAMPLE PROBLEMS

A few sample problems will illustrate the use of the NEW_DOUBLE program. The first problem will use an ellipse of thickness ratio 0.1. The second problem will analyze an airfoil-like shape with a thickness ratio of 0.12 and a chordwise location of maximum thickness of 0.30. Finally, the third problem will analyze a symmetric shape.
V. STARTING THE PROGRAM

Begin with the screen showing the DCL prompt, which looks like this:

$  

Next, ensure that the program is in your directory by typing:

DIR [Return]

and viewing the files for NEW_DOUBLE.EXE.

To run the program, type:

RUN NEW_DOUBLE [Return]

The program will start and the screen should look similar to that shown in Figure 2.

PROGRAM NEW_DOUBLE: VERSION 3 : 4 OCTOBER 89

DOUBLET DISTRIBUTION METHOD IS USED TO DETERMINE INCOMPRESSIBLE FLOW AROUND AN ELLIPSE, SYMMETRICAL AIRFOIL OR ARBITRARY-SYMMETRIC SHAPE AT ZERO ANGLE OF ATTACK

PROGRAM ASSUMES A NONDIMENSIONAL CHORD, THAT IS, THE VALID RANGE OF $X$ IS FROM 0 TO 1.

ENTER TYPE OF BODY SHAPE DESIRED:
1) ELLIPTIC
2) SYMMETRICAL AIRFOIL-LIKE OR
3) ARBITRARY SYMMETRIC SHAPE

ENTER 1, 2, OR 3.

NOTE THAT OPTION 3 WILL REQUIRE MANUALLY INPUTTING DATA POINTS FOR THE UPPER SIDE OF THE RESPECTIVE BODY

Figure 2. Initial Screen for Program NEW_DOUBLE
VI. SAMPLE GRAPHICAL OUTPUTS

A. SAMPLE PROBLEM ONE

For the elliptic case, respond to the initial screen request by entering:

1 (Return)

Respond to the request for the thickness ratio by entering:

0.1 (Return)

Now enter the number of intervals you desire the doublet distribution to have by entering:

10 [Return]

The screen should now look like that shown in Figure 3.

WHICH METHOD DO YOU WISH TO USE TO DETERMINE THE DOUBLET DISTRIBUTION ENDPOINTS?

1) PROGRAM INTERVAL-HALVING SUBROUTINE TO ITERATE

2) MANUAL ITERATION BY THE USER

3) RETURN TO START

4) EXIT PROGRAM

ENTER 1,2,3 OR 4

Figure 3. Endpoint Determination Method Selection Screen
Respond to the question by entering:

1 [Return]

If you should desire to enter your own values, enter 2.

The next values you will be required to enter are for the X location tolerance and the stagnation point velocity function tolerance. It is recommended that values of 10E-6 (0.000001) be used. The maximum number of iterations should be set at a value of at least 20 when using such small tolerances. Additionally, if you desire to use, for example, 10 intervals, you should use 10E-4 so as to achieve a small velocity magnitude at the stagnation points.

The output parameter entry has only to do with the interval halving subroutine. Unless you are having problems with the program or are interested in the convergence of the solution, it is recommended that this value be set to zero (0).

Following entry of the output parameter, the program begins the solution process. It returns with UO and U1, the values for the X velocity component at the leading and trailing stagnation points respectively and the values for XS and XF, the beginning and ending points of the line doublet distribution. If the values for UO and U1 are sufficiently close to zero, say less than 10E-3 (0.001), then enter:

Y [Return]

If you desire more accuracy, enter:

N [Return]
and then reenter the tolerance and maximum iteration values. Responding with a (Y) will cause the program to proceed to the output stage. Values will be printed to the screen and to the following data files:

- **DUBLET.DAT**: DOUBLET STRENGTH DISTRIBUTION
- **SHAPE.DAT**: BODY SURFACE COORDINATES
- **PRESSURE.DAT**: SURFACE PRESSURE DISTRIBUTION

You will be asked for the number of pressure coefficient output points you desire. This number is independent of the number of intervals of the line doublet distribution. It affects only the number of output data points and not the accuracy of the solution. After entering the number of Cp output points, pressure distribution data will be displayed to your screen. The program now asks if you want to print the results (Y/N). Enter your response and select the respective file which you want to print from a tabulated listing. However, be aware that you must have already logged onto the KELLY terminal to print anything, or be at a terminal which is connected to a printer.

You will now be asked if you want to graph the results (Y/N). If you respond affirmatively, the screen will look similar to Figure 4.

Once you have selected your plotting option and the respective plot has appeared on your screen (on the KELLY terminal screen if you are printing items) you will be asked if you would like a print of the plot (Y/N). Answer accordingly and continue with the program.

You will now be asked if you would like to make another run. Enter:

1 [Return]
WHICH OF THE FOLLOWING DATA FILES DO YOU WANT TO GRAPH?
1) DUBLET.DAT
2) PRESSURE.DAT
3) SHAPE.DAT
4) NONE
INPUT OPTION NO. (1,2,3 OR 4)

Figure 4. Plotting Options Screen

B. SAMPLE PROBLEM TWO

Sample problem two will work through the airfoil-like shape case and the user will supply the values of XS and XF. The user may experiment with manual iteration, however to save space this sample will use previously determined satisfactory values of XS and XF for the initial guess.

You should now be back at the initial screen and it should look like Figure 2. For the airfoil-like case enter:

2 [Return]

Respond to the request for the thickness ratio by entering:

.12 [Return]
For the chordwise location of maximum thickness, enter:

.30 [Return]

Now enter the number of intervals you desire the doublet distribution to have by entering:

10 [Return]

The next step is to select the method for the determination of the endpoints for the doublet distribution. The screen should look like Figure 3. This time respond to the question by entering:

2 [Return]

For the doublet distribution starting point, XS, enter

.0082129128 [Return]

For the doublet distribution ending point, XF, enter

.9994138 [Return]

As with the previous example, the program now begins the solution process. It returns with U0 and U1, the values for the X velocity component at the stagnation points. It also echoes back the values entered for XS and XF. If the returned values for U0 and U1 are sufficiently close to zero, then enter:

Y [Return]

This response will cause the program to proceed to the output stage. Values will be printed to the screen and to the data files.
Enter the number of pressure coefficient output points you desire. You are reminded that this number is independent of the number of intervals of the line doublet distribution and it does not affect the accuracy of the solution.

Again, you will be afforded the opportunity to print and graph the results as in sample problems one.

The program now asks if you want to make another run. Enter:

1 [Return]

C. SAMPLE PROBLEM THREE

Sample Problem Three provides an example of arbitrary shape analysis. You should now be back at the initial screen and it should look like Figure 2. For the symmetric shape case enter:

3 [Return]

Once you have entered this response, your screen should look similar to Figure 5.

```
HOW MANY UPPER PROFILE DATA POINTS DO YOU DESIRE? (ENTER A NUMBER BETWEEN 3 AND 100)

BE AWARE THAT THE LEADING EDGE OF YOUR DESIRED SHAPE HAS BEEN PROGRAMMED TO BE AT THE ORIGIN AND THAT YOUR TRAILING EDGE IS AT (1,0). SCALE YOUR SHAPE/OBJECT ACCORDINGLY.
```

Figure 5. Symmetric Shape Data Point Input Screen
Enter the number of points you wish to use to describe your symmetric shape. You will then be given the opportunity to enter each point. Once you have entered all of your surface points, the program will ask if you want to check your input data. You may then make any corrections as necessary. When you have finished correcting your data, enter N to the question asking you if you have any 'input data corrections.' The program will then proceed as described in example problems one and two.

This completes the sample problems for the NEW_DOUBLE program. Representative graphical outputs created by these sample runs are listed in Figures 6 through 8. Since the bodies analyzed by this program are symmetrical with respect to the x axis, only the upper surface body shape coordinates and pressure coefficients are output. For this reason, the piecewise constant doublet strength M(I) is divided by two to indicate the portion affecting the upper surface.
Figure 6. Doublet Strength Distribution

- Symmetric Airfoil Doublet Distribution
- Thickness Ratio (\( \tau \)) = 0.12
- Maximum Thickness at \( x = 0.30 \)
- Number of intervals used = 10
ELLiptical AirfoiL Cp Distributions

THICKNESS RATIO (TAU) = 0.10
NUMBER OF INTERVALS USED = 20

Figure 7. Cp Distribution
Figure 8. Airfoil Shape

Upper Surface

• = Airfoil Shape

Symmetric Airfoil Shape

Thickness Ratio (tau) = 0.12

Maximum thickness at X = 0.30

Number of intervals used = 10
APPENDIX B
PROGRAM NEW_PANEL USER'S MANUAL
USER'S GUIDE CONTENTS

I. INTRODUCTION ................................................................. 36
II. ASSUMPTIONS AND LIMITATIONS ...................................... 36
III. INPUT DESCRIPTIONS ...................................................... 36
IV. INPUT RESTRICTIONS ...................................................... 37
V. OUTPUT VARIABLES FOR VISCIOUS RESULTS ..................... 37
VI. SAMPLE PROBLEMS ......................................................... 38
VII. STARTING THE PROGRAM ................................................ 38
VIII. SAMPLE GRAPHICAL OUTPUTS ........................................ 39
I. INTRODUCTION

The purpose of the NEW_PANEL program is to provide an analysis of the aerodynamics of NACA four-digit airfoils and airfoils of the NACA 230XX family using the panel method. This program has been modified to accept arbitrary airfoil surface coordinate input. NEW_PANEL has also been adapted to analyze viscous effects.

II. ASSUMPTIONS AND LIMITATIONS

This program is limited to single-element airfoils. The solution is determined for conditions of incompressible and inviscid irrotational flow. The coefficient of drag provided in the results is due to numerical round-off error. When considering the viscous analysis loop of the program, it is important that you understand that the Cebeci eddy-viscosity program adaptation is sensitive to flow separation on the airfoil. Boundary layer thickness and other boundary layer characteristics will be computed. It is advised that viscous analysis be limited to small angles of attack and to relatively slender airfoils.

III. INPUT DESCRIPTION

As with the NEW_DOUBLE program, there are very few input values required for this simple program. Their description and program variable names are listed below.

NUPPER - Number of nodes on the upper surface.
NLOWER - Number of nodes on the lower surface.
X(1),Y(1) - Surface coordinates.

These may be entered from the keyboard, from a data file, or from data statements. The program is capable of generating an approximation for airfoils of the NACA XXXX and 230XX series.
ALPHA — Angle of attack. (Angle between the chord and the freestream velocity.)

RL — Chord Reynolds' Number

XCTRI(1) - Flow transition point from laminar to turbulent flow on the top of the airfoil

XCTRI(2) - Flow transition point from laminar to turbulent flow on the bottom of the airfoil.

IV. INPUT RESTRICTIONS

The program, as written, is limited to 100 total surface nodes. This may be modified by changing the size of the arrays; however, only a very complex surface should require that many values to accurately define the surface. If that is the case, a more sophisticated program should be considered for the investigation. As mentioned above, the computer generated approximations to airfoil shapes are limited to the NACA XXXX and 230 XX series. The program will accept values for ALPHA up to 90 degrees, but the user is cautioned that since separation can exist for angles of attack as low as 5°-10°, results for values above about 10° may be suspect.

V. OUTPUT VARIABLES FOR VISCOUS RESULTS

XC - Airfoil Coordinate (Abcissa)

S - Station Location

VW - Wall Shear

CF - Coefficient of Friction

DLS - Displacement Thickness

THT - Momentum Thickness
VI. SAMPLE PROBLEMS

A few sample problems will illustrate the use of the NEW_PANEL program. The first run will be done using an approximation to a NACA 0012 airfoil which is generated by the program using the information associated with each digit in the NACA number. The second run will analyze a NASA LS(1)-0013 airfoil using a set of data statements containing the airfoil surface coordinates. These statements have been inserted into the proper location in the program already. The last sample problem will re-analyze the LS(1)-0013 airfoil but now viscous effects will be included.

VII. STARTING THE PROGRAM

Begin with the screen showing the DCL prompt, which looks like this:

\$

Next, ensure that the program is in your directory by typing:

**DIR [Return]**

and viewing the files for NEW_PANEL.EXE. To run the program, type:

**RUN NEW_PANEL [Return]**

The program will start and the screen should look similar to what is shown in Figure 9.
PROGRAM NEW_PANEL

SMITH-HESS (DOUGLAS) PANEL METHOD FOR A SINGLE-ELEMENT LIFTING AIRFOIL IN TWO-DIMENSIONAL INCOMPRESSIBLE FLOW

DO YOU WISH TO:

1) USE AIRFOIL SURFACE COORDINATE DATA VALUES.

2) HAVE COMPUTER GENERATE AN APPROXIMATION FOR NACA XXXX OR 230XX AIRFOIL SECTION.

3) QUIT THE PROGRAM

ENTER 1, 2, OR 3

Figure 9. Initial Screen for Program NEW_PANEL

VIII. SAMPLE GRAPHICAL OUTPUTS

A. SAMPLE PROBLEM ONE

For the first case we will have the computer generate an approximation for the shape of a NACA 0012 airfoil, consisting of 20 surface panels, using an algorithm contained in subroutine NACA45. The angle of attack of the onset flow will be six degrees. To use the approximation method, enter:

2 [Return]

Respond to the request for the number of surface data points by entering:

20 [Return]
Confirm the number of surface data points you desire by entering:

1 [Return]

Although the program will allow a different number of upper and lower surface data points, it is recommended that you try and keep them equal. An unequal number of nodes yields trailing-edge panels of unequal length, which lowers the accuracy of the approximation of the Kutta condition. Respond to this question by entering:

1 [Return]

The next question asks for the NACA number of the airfoil you are considering. For this case we will look at the NACA 0012, so enter:

0012 [Return]

The screen should now look like similar to Figure 10.

The program is now ready to perform its calculations. The final piece of information required is the angle of attack, ALPHA. For this case, respond to the question by entering:

6 [Return]

Following entry of the angle of attack, the program begins the solution process. Values scroll up the screen and are simultaneously being written to the data files. You should now see a screen similar to the one shown in Figure 11.

Should you select to print the results, you will be given the option to print both of the data files or just the one you want. Once you have finished printing the results, you will be asked if you want to graph the results. Respond affirmatively and the screen should then look similar to Figure 12.
ENTER NUMBER OF SURFACE DATA POINTS DESIRED

20

NUMBER OF SURFACE DATA POINTS TO BE GERATED = 20
IS THIS VALUE CORRECT? (YES=1, NO=2)

1

ARE THE NUMBER OF UPPER AND LOWER SURFACE DATA POINTS (NODES) EQUAL? (YES=1, NO=2)

1

INPUT NACA NUMBER, ANY FOUR DIGIT OR 230XX SERIES

0012

INPUT ALPHA IN DEGREES

Figure 10. Screen Showing Data for Computer Generated Airfoil

PROGRAM NEW_PANEL RESULTS HAVE BEEN WRITTEN TO FILES:
PBODY.DAT : BODY SURFACE COORDINATES
PPRESS.DAT : SURFACE PRESSURE DISTRIBUTION
WOULD YOU LIKE TO PRINT THE RESULTS (Y/N)?

Figure 11. Printing Option Screen
WHICH OF THE FOLLOWING DATA OUTPUTS DO YOU WANT TO PLOT?

1) PPRESS.DAT (CP DISTRIBUTION)
2) PBODY.DAT (AIRFOIL SHAPE)
3) CL VS. ANGLE OF ATTACK & CM C/4 VS. ANGLE OF ATTACK
4) NONE

INPUT OPTION NO. (1,2,3 OR 4)

Figure 12. Graphical Selection Screen

Once the selected plot is displayed on your screen (screen KELLY if you are printing) you will be given the option of printing the plot. Again, you must have already used the "set host kelly" command to print items. If you elect not to print the graphical output you screen will again look similar to Figure 12. Selecting option 4 (NONE) will exit you from the graphing loop. You will now be asked to analyze the viscous effects for the airfoil. Respond negatively by entering:

N [Return]

A new screen will be presented and the program now asks if you want to make another run. Enter:

1 [Return]
B. SAMPLE PROBLEM TWO

This time the sample problem will examine a NASA LS (1)-0013 whose coordinates have been entered as data statements in the program. You should now be back at the initial screen and it should look like Figure 9. Since you will be using actual airfoil coordinate data values, enter:

1 [Return]

The screen shown in Figure 13 now presents you with the three choices available for entering the airfoil surface coordinate data values. You will be using the data statements, so enter:

3 [Return]

DO YOU WISH TO ENTER THE SURFACE COORDINATE VALUES:

1) FROM A DATA FILE.
2) FROM THE KEYBOARD.
3) USING DATA STATEMENTS ALREADY ENTERED IN THE MAIN PROGRAM. **NOTE** THIS REQUIRES THAT PROGRAM BE MODIFIED IN ADVANCE BY MOVING DATA STATEMENTS TO THE CORRECT LOCATION.

ENTER 1, 2, OR 3. (FOR PREVIOUS MENU ENTER 4)

Figure 13. Menu for Surface Coordinate Data Entry Method
The number of data points has been entered via the data statements, therefore you are not asked that question for this case. For the angle of attack, again enter:

6 [Return]

As you saw in the previous example, values scroll up the screen. The program will again allow you to print or graph the respective results as before. Additionally, you will again be asked if you want to analyze viscous effects. Respond accordingly to exercise the required program options. Finally, the program will ask if you want to make another run. Enter:

1 [Return]

C. SAMPLE PROBLEM THREE

As noted earlier, this sample problem will again analyze the LS(1)-0013 airfoil but with viscous effects. You should now be back at the initial screen and it should look like Figure 9. Since you will be using airfoil surface coordinate data values enter:

1 [Return]

The screen should again look like Figure 13. Again enter the response 3 in that data statements will again be used.

3 [Return]

For the angle of attack response enter:

0 [Return]
As you have seen in the two previous examples, values scroll up the screen. The program will again allow you to print and/or graph the respective results as before. When asked if you would like to analyze the viscous effects for this airfoil enter:

Y [Return]

The screen should now look similar to Figure 14.

The first option (1) is used to input an arbitrary external velocity profile. The external velocity values at each respective point were obtained from the expression: \( \text{SQRT}(1-C_p) \). To input the \( C_p \) distribution just created for the LS(1)-0013 airfoil enter:

2 [Return]

---

**VISCOUS BOUNDARY LAYER ANALYSIS**

*** INPUT DATA OPTION ***

WHAT INPUT SOURCE WOULD YOU LIKE TO USE ?

1) DATA FILE "BL2D.DAT" OR
2) NEW_PANEL CP DISTRIBUTION JUST CREATED
3) QUIT PROGRAM

ENTER 1, 2, OR 3

---

Figure 14. Menu for Viscous Data Input Option
You will now be asked to enter the flow Reynold's Number. Enter:

6000000 [Return]

Now you will be asked to enter the respective nondimensionalized values of XCRIT(1) and XCRIT(2). Again these values correspond to the point along the chord of the airfoil at which flow transition from laminar to turbulent occurs for the top and bottom of the airfoil, respectively. Enter .3 for both values. To avoid flow separation, these value should be greater than 0.15 for analysis at angles of attack in excess of approximately 5°.

The program will now begin to process the input data and determine the boundary layer characteristics. Upon completion of the computations the screen should look similar to Figure 15.

Remember that in order to print the results you must be logged onto a computer terminal which is connected to a printer. Enter the following command to print the boundary layer results:

Y [Return]

Once you have finished the viscous flow analysis process, the program will again ask you if would like to make another run of NEW_PANEL. Enter the following command to exit the NEW_PANEL program:

2 [Return]

This completes the sample problems for the NEW_PANEL program. Representative graphical outputs created by these sample runs are shown in Figures 16 through 18.
READING THE DATA...
INPUT OF DATA COMPLETE.
BOUNDARY LAYER COMPUTATIONS IN PROGRESS...
BOUNDARY LAYER COMPUTATIONS IN PROGRESS...

THE BOUNDARY LAYER RESULTS HAVE BEEN
WRITTEN TO FILE "BL2D.OUT"

WOULD YOU LIKE TO PRINT THESE RESULTS?

Figure 15. Viscous Data Output File Option Screen
NACA AIRFOIL 12
NUMBER OF PANELS USED = 20
ANGLE OF ATTACK = 6.00

CP DISTRIBUTION
• = LOWER AND UPPER AIRFOIL POINTS

Figure 16. Cp Distribution
NUMBER OF PANELS USED = 28
ANGLE OF ATTACK = 6.00

Figure 17. Body Shape
MACA AIRFOIL 12
NUMBER OF PANELS USED = 20

- = CL VS. ANGLE OF ATTACK
○ = CM C/4 VS. ANGLE OF ATTACK
□ = ZERO LINE-REFERENCE ONLY

Figure 18. Cl & Cm c/4 vs. Alpha
APPENDIX C
PROGRAM NEW_VOR USER'S MANUAL

USER'S GUIDE CONTENTS

I. INTRODUCTION ................................................................. 52
II. ASSUMPTIONS AND LIMITATIONS ........................................ 52
III. INPUT DESCRIPTION ......................................................... 52
IV. INPUT RESTRICTIONS ....................................................... 53
V. SAMPLE PROBLEMS ............................................................ 53
VI. STARTING THE PROGRAM ................................................... 53
VII. SAMPLE GRAPHICAL OUTPUTS .......................................... 54
I. INTRODUCTION

The purpose of the NEW_VOR program is to provide an application of the vortex lattice method for the determination of the lift distribution of a flat rectangular plate. This method is based on a distribution of discrete horseshoe vortices over a wing surface that has been divided into a finite number of panels. A system of linear equations is developed for the vortex strengths on the panels and solved by matrix methods.

II. ASSUMPTIONS AND LIMITATIONS

This program is limited to flat rectangular wings which it divides into panels, using a uniform grid. Additionally, the uniform grid spacing method incorporates an enhancement whereby the panels do not extend to the wing tips, but only to a distance of $d/4$ from the tips. The value of $d$ is the spanwise width of a wing panel.

The solution is determined for conditions of incompressible and inviscid irrotational flow. Since we are considering an inviscid fluid, the coefficient of drag provided in the results is an accumulation of numerical errors. This program is intended to be used for the analysis of flat rectangular wings with low aspect ratio.

III. INPUT DESCRIPTION

There are very few input values required for this simple program. Their description and program variable names are listed below.

AR—Aspect ratio of the wing. $(\text{Span})^2/\text{Area or Span/Chord.}$
NX,NY—Number of vortices in the X and Y directions.
ALPHA—Angle of attack. (Angle between the chord and the freestream velocity.)
IV. INPUT RESTRICTIONS

The program, as written, is limited to 350 total surface vortices. This may be modified by changing the size of the arrays, however for the wings that this program was intended to analyze, this should be sufficient. The program will accept values for ALPHA up to 45 degrees, but, as noted previously with program NEW_PANEL, the user is cautioned that values above about 10° may result in output data which in incorrect.

V. SAMPLE PROBLEMS

Two sample problems will be used to illustrate the use of the NEW_VOR program. The first run will use a flat rectangular wing with an aspect ratio of two. The lattice will be created by placing three vortices on the wing in the X direction and five vortices on the wing in the Y direction. The vortices will be distributed using the Uniform Grid spacing method and the wing will be set at an angle of attack (alpha) of six degrees. The second run will use the same wing, but with five vortices on the wing in the X direction and 10 vortices on the wing in the Y direction.

VI. STARTING THE PROGRAM

Begin with the screen showing the DCL prompt, which looks like this:

\$  

Next, ensure that the program is in your directory by typing:

```
DIR [Return]
```

and viewing the files for NEW_VOR.EXE.

To run the program, type:

```
RUN NEW_VOR [Return]
```
VII. SAMPLE GRAPHICAL OUTPUTS

A. SAMPLE PROBLEM ONE

The program will start and the screen should look similar to what is shown in Figure 19.

```
PROGRAM VORLAT : VERSION 5 : 10 OCTOBER 89
VOXEL-LATTICE METHOD USED TO DETERMINE SPANWISE
LIFT DISTRIBUTION FOR A FLAT RECTANGULAR WING

ENTER THE ASPECT RATIO?
```

Figure 19. Initial Screen for Program NEW_VOR

Respond to the request for the aspect ratio by entering:

```
2 [Return]
```

Respond to the request for the number of vortices by entering:

```
3,5 [Return]
```

Finally, enter the angle of attack in degrees:

```
6 [Return]
```

The screen is then cleared and you will be presented with what is shown in Figure 20. If your display agrees with this, respond to the question by entering:

```
1 [Return]
```
THE CURRENT VALUES ARE:
1) ASPECT RATIO .................. = 2.000000
2) NUMBER OF VORTICES (NX, NY) = 3, 5
3) ANGLE OF ATTACK (DEGREES) = 6.000000

THE CALCULATED PARAMETERS ARE:

\[ \Delta x = 0.3333333 \]
\[ \Delta y = 0.1904762 \]

NUMBER OF EQUATIONS TO SOLVE = 15
ARE THESE VALUES CORRECT? (YES=1, NO=2)

Figure 20. Data Review/Correction Screen

If you should desire to change any values, enter 2, and you will be asked which value you want to correct and the new desired value. Following entry of the correct values and a positive response, the program begins the solution process. It returns with the coefficients of lift and drag at the indicated spanwise positions, as well as the chordwise center of pressure for those positions. Overall values for the coefficients of lift, drag, induced drag and moment about the leading edge are calculated and then printed out near the bottom of the screen. Don't worry if you miss some of the values as they scroll up on the screen. All the values are printed to both the screen and to the data file (VORLAT4.DAT).

The program now asks if you want to print the results. Entering an affirmative response of 'Y' will print the output file VORLAT4.DAT.

The program will now ask if you want to graph the results. Enter: 

Y [Return]
Your screen should now look similar to Figure 21.

WHICH OF THE FOLLOWING RELATIONSHIPS
DO YOU WANT TO GRAPH?

1) CL VS. Y
2) CD VS. Y
3) CL VS. CD
4) NONE

INPUT OPTION NO. (1,2,3 OR 4)

Figure 21. Graphical Selection Screen

Enter a desired plot selection and compare your one plot to the sample output plot at the end of this section. There should not be any difference. You will also be asked if you would like a print of the respective plot. Upon entering:

N [Return]

your screen should once again be similar to Figure 21. Enter a response of 4 to exit the graphing loop.

B. SAMPLE PROBLEM TWO

The program now asks if you want to make another run. Enter:

1 [Return]

You should now be back at the data review/correction screen and it should look like Figure 20.
Now run the same wing, but change the number of vortices to 5 and 10. Enter:

2 [Return]

You want to change the number of vortices, so enter

2 [Return]

Respond to the request for the number of vortices by entering:

5,10 [Return]

The screen is automatically updated and you will see that the number of vortices has changed. As in the previous example, responding with a '1' causes the program to proceed to the output stage. The solution will be printed to the screen and appended to the data file which contains the data from the prior run. Again you will be afforded the opportunity to print and graph the results as in Sample Problem One. Respond accordingly...the output file/graphical plots for all plotting selections are enclosed at the end of this section.

The program now asks if you want to make another run. The session is finished, so enter

2 [Return]

This completes the sample problems for the NEW_VOR program. Figures 22 through 24 give representative graphical outputs created by these sample problems. To create these plots, five vortices across the wing chord (NX) and 10 vortices across the span (NY) were used.
Figure 22. Cl vs. Y
FLAT RECTANGULAR WING
ASPECT RATIO (AR) = 2.00
ANGLE OF ATTACK = 6.00 DEGREES

Figure 23. Cd vs. Y
2-D PLOT

\( \bullet = \text{CL/CD VALUES} \)

FLAT RECTANGULAR WING

\( \text{ASPECT RATIO (AR)} = 2.00 \)

\( \text{ANGLE OF ATTACK} = 6.00 \text{ DEG} \)

Figure 24. Cl vs. Cd
APPENDIX D
PROGRAM SUB USER'S MANUAL

USER'S GUIDE CONTENTS

I. INTRODUCTION ......................................................... 62
II. ASSUMPTIONS AND LIMITATIONS ............................... 62
III. INPUT DESCRIPTION .................................................. 63
IV. SAMPLE PROBLEMS .................................................... 67
V. STARTING THE PROGRAM ............................................ 67
VI. SAMPLE GRAPHICAL OUTPUTS ..................................... 68
I. INTRODUCTION

The SUB program has been adapted from a National Aeronautics and Space Administration (NASA) FORTRAN program and has been used considerably at the Langley Research Center. Additionally, this particular program has also been used in industry and the results have shown good correlation with experimental values. SUB has subsequently been revised to enhance its ease of use and its ability to present accurate graphical results.

The purpose of the SUB program is to estimate the subsonic aerodynamic characteristics of complex planforms. The program represents a lifting planform with a vortex lattice. A relatively complex planform may be analyzed by creating the planform with up to 24 line segments on a semispan. Additionally, these line segments may have an outboard variable-sweep panel or they may have several dihedral angles across the span. Furthermore, two planforms may be used together to represent a combination of wings and tails or wing, bodies, and tails.

II. ASSUMPTIONS AND LIMITATIONS

The use of this program is confined to the subsonic flow regime. Additionally, the planform is in steady, uniform, inviscid, incompressible, attached flow conditions.

Certain restrictions must also be kept in mind when using this program. Three specific restrictions apply to all planforms analyzed: 1) Only a total of two planforms may be specified; 2) The maximum number of horseshoe vortices on the left side must be limited to 120. When two planforms are specified, the sum total of the vortices is limited to 120. Within this limit, the number of horseshoe vortices in any chordwise row may vary from 1 to 20 and
the number of chordwise rows may vary from 1 to 50, and 3) The left side of the planform must be described with less than 24 line segments.

Additionally, there are also three limitations which must be applied to variable-sweep planforms: 1) There should always be a fixed-sweep panel between the root chord and the outboard variable-sweep panel; 2) The pivot cannot be canted from the vertical, and 3) Dihedral considerations cannot be programmed for the variable-sweep panel or at the intersection of this panel with the fixed portion of the wing.

Finally, there exists three limitations when considering planforms which have nonzero dihedral angles or to two planforms which do not lie in the same plane: 1) The variation in local chord must be continuous from the tip chord to the root chord of each planform specified; 2) The number of horseshoe vortices in each chordwise row must be at least two, and 3) The number of horseshoe vortices must be constant over the semispan of each planform.

III. INPUT DESCRIPTION

There are relatively few input values required for this program. Their description and program variable names are listed below. The user's first task before running this program will be to create an input data file corresponding to the respective planform to be analyzed and the desired program specifications. Each line of the input file is detailed explicitly.
A. GROUP ONE DATA

Line 1

**PLAN**  Number of planforms for the configuration.
**TOTAL**  Number of sets of group two data (normally one).
**CREF**  Reference chord of the configuration (greater than zero).
**SREF**  Reference area of the configuration (greater than zero).

Line 2

**AAN(IT)**  Number of line segments used to describe the left half of the planform.
**XS(IT)**  X location of the pivot; use 0 on a fixed wing.
**YS(IT)**  Y location of the pivot; use 0 on a fixed wing.
**RTCDHT(IT)**  Vertical distance of the particular planform being read in with respect to the wing root chord height; use 0.

** The next series of input data lines are used to describe each line segment which was used to specify the planform shape. In other words, if one has used five line segments to describe his or her planform, the next five lines will describe each line segment respectively. The first break-point is located at the intersection of the left wing leading edge with the root chord. They are numbered in increasing order for each intersection of lines in a counterclockwise direction. The input variables for each of these lines is as follows: **

**Lines 3-Whatever**

**XREG(I,IT)**  X location of the ith breakpoint.
**YREG(I,IT)**  Y location of the ith breakpoint.
**D IH(I,IT)**  Dihedral angle (degrees) in y-z plane of line from breakpoint; positive upward.
AMCD  The move code. (This input indicates whether or not the line segment in question is on a movable panel. Use 1 for a line which is fixed or 2 for a line which is movable.

B.  GROUP TWO DATA

Depending upon planform specifications, group two data could consist of three sections. The first section is always included. The second section is to be used if the number of chordwise horseshoe vortices varies across the semispan. One reason to vary the number of chordwise vortices across the semispan would be the need to analyze specific portions of the wing which may experience great pressure gradients i.e. at the intersection point of the fixed and the movable planforms for a variable sweep wing. The third section is used when the wing has twist and/or camber distribution and may consist of up to 15 lines, depending upon the number of horseshoe vortices.

Line 1 (Section One)

CONFIG  An arbitrary configuration number (up to four digits)—user's choice.

SCW  The number of chordwise horseshoe vortices to be used to represent the wing; a maximum value of 20 may be used. If the user desires that the number of chordwise vortices vary across the semi span, enter 0. Entering zero will require the use of section two of Group Two data. The SCW = 0 option can only be used on wings without dihedral and for coplanar wing-tail configurations.

VIC  The number of spanwise rows at which chordwise horseshoe vortices will be TBLSCW(I) cannot exceed 120.

MACH  Mach number. A value other than 0 will cause the Prandtl-Glauert compressibility factor to be applied. Regardless, the Mach number should be less than the critical Mach number.
**CLDES** Desired Lift Coefficient. Used to obtain the span load distribution at a particular lift coefficient. If this aspect is not required enter 1. Enter 11 for drag polar data.

**PTEST** If the damping-in-roll parameter is desired, enter 1.

**QTEST** If CLq or Cmq stability derivatives are desired, enter 1. However, PTEST and QTEST cannot both be done in the same program run.

**TWIST(1)** Twist code for the first planform. Enter 0 for no twist. Enter 1 if the planform has twist and provide data in section three.

**SA(1)** Variable sweep angle for the first planform. Specify the leading-edge sweep angle (degrees) for the first movable line adjacent to the fixed portion of the planform. For a fixed planform, this quantity may be omitted.

**TWIST(2)** Twist code for the second planform.**

**SA(2)** Variable sweep angle for the second planform.**

**Obviously, these inputs may be omitted if there is only one planform.

**Line 2 (Section Two)**

Again, section two is to be used if SCW was set to 0 thus allowing for the number of chordwise horseshoe vortices to vary across the semispan.

**STA** Total number of spanwise rows of horseshoe vortices per semispan. This input sets the number of values to be read into TBLSCW(I)—next input.

**Lines 3-Whatever**

**TBLSCW(I)** Number of horseshoe vortices in each row starting at the row near the tip of the first planform and proceeding to the row near the root. If a second planform has been specified, the table of chordwise rows concludes with the number of vortices specified for the second planform (see Example B for format).
Section Three

Again, section three is to be used if the planform has twist.

ALP(NV) Local angle of attack in radians. Refer to Example Three (3)

FORMAT Refer to the sample input data files on how to properly format the input data files. Failure to follow these examples implicitly will result in a "data read error".

IV. SAMPLE PROBLEMS

Three sample problems have been included in this user's guide section. The first two problems analyze a fixed planform without a variable-sweep panel. The first problem simply uses four (4) spanwise vortices while the second problem uses 40 spanwise vortices. The second problem demonstrates the benefit of using extra horseshoe vortices to enhance data representation. The last problem is an example of a rather complex planform. This particular wing has variable chordwise vortices across seven (7) spanwise rows and has twist incorporated into the wing. Additionally, this wing is described using 14 line segments.

V. STARTING THE PROGRAM

Begin with the screen showing the DCL prompt, which looks like this:

$ 

Next, enter the following command:

SET DEF [.SUB]

Now, enter the command to run the program:

RUN SUB
The program will start and the screen should look similar to what is shown in Figure 25.

![Figure 25. Initial Screen for Program SUB](image)

**VI. SAMPLE GRAPHIC OUTPUTS**

**A. EXAMPLE PROBLEM 1**

Enter the name of the input data file.

**A9WS60.DAT [Return]**

Once the program has finished its data tabulations, your screen should be similar to Figure 26.

![Figure 26. Printing Determination Screen](image)
Respond negatively to this request by typing:

N [Return]

Respond affirmatively to the request to copy the output data file (OUTFILE.DAT) to another file by typing:

Y [Return]

A screen similar to Figure 27 will then appear which lists the file choices possible for copying.

<table>
<thead>
<tr>
<th>WHAT NAME WOULD YOU LIKE FOR THE OUTPUT FILE?</th>
</tr>
</thead>
<tbody>
<tr>
<td>1) VIGILANTE.DAT</td>
</tr>
<tr>
<td>2) CORSAIR.DAT</td>
</tr>
<tr>
<td>3) HAWKEYE.DAT</td>
</tr>
<tr>
<td>4) SKYHAWK.DAT</td>
</tr>
</tbody>
</table>

Figure 27. Output File Designation Screen

Select from the designated list of file names your choice.

1,2,3, OR 4 [Return]

Respond affirmatively to the request to graph the results by typing:

Y [Return]

The screen should now look like Figure 28.
WHICH OF THE FOLLOWING RELATIONSHIPS DO YOU WANT PLOTTED?

1) INDUCED DRAG COEFF VS. 2Y/B
2) LE EDGE THRUST COEFF VS. 2Y/B
3) SUCTION COEFF VS. 2Y/B
4) SPAN LOAD COEFF VS. 2Y/B
5) CL RATIO VS 2Y/B
6) NONE

INPUT OPTION NO. (1, 2, 3, 4, 5, OR 6)

Figure 28. Plot Determination Screen

Select from the designated list of graphical relationships your choice.

1, 2, 3, 4, OR 5 [Return]

The requested plot will momentarily appear on your screen. If you have remoted your terminal to terminal "KELLY" for printing purposes your plot will come up on the "KELLY" monitor. Compare your plot with the example plots corresponding to EXAMPLE 1; it should be the same.

Respond negatively to the request to print the plot by typing:

N [Return]
The user will again be given the opportunity to graph another relationship (Figure 28 will be presented). Respond with the 6th choice to exit the graphing loop. Enter:

6 [Return]

The program now asks if you want to make another run. Enter

1 [Return]

**B. EXAMPLE PROBLEM 2**

The screen should again look like Figure 25.

Enter the name of the input data file.

**E9WS60.DAT [Return]**

Respond negatively to the request for a printed copy of the output file by typing:

**N [Return]**

Respond negatively to the request to copy the output data file (OUTFILE.DAT) to another file by typing:

**N [Return]**

Respond affirmatively to the request to graph the results by typing:

**Y [Return]**

Again, Figure 28 will appear on your screen with a listing of the available plotting routines.

Select from the list your plotting choice.

**1,2,3,4, OR 5 [Return]**
The requested plot will momentarily appear on your screen. Again, if you have remoted your terminal to terminal "KELLY" for printing purposes your plot will come up on the "KELLY" monitor. Compare your plot with the example plots corresponding to EXAMPLE 2; it should be the same.

Respond negatively to the request to print the plot by typing:

N [Return]

The user will again be given the opportunity to graph another relationship. Respond with the 6th choice to exit the graphing loop. Enter:

6 [Return]

Respond affirmatively to the request to perform another run of program by typing:

1 [Return] **

**Entering a "2" would exit the user from the program.

C. EXAMPLE PROBLEM 3

Enter the name of the input data file.

B9WS60.DAT [Return]

Respond negatively to the program request to print OUTFILE.DAT. Enter:

N [Return]

Respond negatively to the request to copy the output data file (OUTFILE.DAT) to another file by typing:

N [Return]
Respond affirmatively to the request to graph the results by typing:

Y [Return]

Select from the designated list of graphical relationships your choice.

1, 2, 3, 4, OR 5 [Return]

The requested plot will then appear on your screen and you will be asked if you want to print the plot. Compare your plot with the example plots corresponding to EXAMPLE 3; it should be the same.

Respond negatively to the request to print the plot by typing:

N [Return]

The user will again be given the opportunity to graph another relationship. Respond with the 6th choice to exit the graphing loop. Enter:

6 [Return]

Respond negatively to the request to perform another run of program by typing:

2 [Return]

This completes the sample problems for the SUB program. Graphical output examples created by these sample runs are shown in Figures 29 through 34. The first five plots were generated from the analysis of a wing with an aspect ratio of nine and a leading edge sweep angle of 60°. The last plot (Figure 34) was produced from the analysis of a rather complex planform [Ref. 6], which had seven rows of spanwise vortices with nine vortices across the chord at horseshoe vortex Number 3.
CONTRIBUTION TO TOTAL COEFF.

• = INDUCED DRAG COEFFICIENT

Figure 29. Induced Drag Coeff. vs. 2Y/B
CONTRIBUTION TO TOTAL COEFF.

• = LE THRUST COEFFICIENT

Figure 30. LE Thrust Coeff. vs. 2Y/B
CONTRIBUTION TO TOTAL COEFF.

• = SUCTION COEFFICIENT

Figure 31. Suction Coeff. vs. 2Y/B
COEFFICIENT OF LIFT (WING) = 1.0

Figure 32. Span Load Coeff. vs. 2Y/B
Figure 33. Coeff. of Lift Ratio vs. 2Y/B
HORSESHOE VORTEX -- NUMBER 3

COEFFICIENT OF LIFT (WING) = 1.0

Figure 34. Delta Cp vs. X c/4
APPENDIX E
PROGRAM SUPER USER'S MANUAL
USER'S GUIDE CONTENTS

I. INTRODUCTION ................................................. 81
II. ASSUMPTIONS AND LIMITATIONS ............................. 81
III. INPUT DESCRIPTION ........................................ 82
IV. SAMPLE PROBLEMS ........................................... 85
V. STARTING THE PROGRAM ...................................... 85
VI. SAMPLE GRAPHICAL OUTPUTS ................................. 86
I. INTRODUCTION

The SUPER program has been adapted from a National Aeronautics and Space Administration (NASA) FORTRAN program and has been used considerably at the Langley Research Center. Additionally, this particular program has also been used in industry. The results have shown good correlation with experimental results. SUPER has subsequently been revised to enhance its ease of use and its ability to present accurate graphical results.

The purpose of the SUPER program is to estimate the supersonic aerodynamic characteristics of complex planforms. Linearized supersonic lifting surface theory is employed to calculate the aerodynamic characteristics of a warped wing of arbitrary planform. The program calculates lifting pressure distribution for the chordwise warped wing at fixed attitude and the pressure distribution (per degree angle of attack) for a corresponding flat wing. These two pressure distributions are combined by superposition principles and integrated over the wing surface to obtain the variation of aerodynamic characteristics with changes in angle of attack.

II. ASSUMPTIONS AND LIMITATIONS

The use of this program is confined to the supersonic flow regime. In addition, the linearized supersonic lifting surface theory, used in this program, applies to wings having negligible thickness.

There exist two specific limitations which must be considered when entering the respective input data values. The number of semispan grid elements is limited to 100 or 47.5*B*SPAN/XMAX. The relative increase in semispan grid elements will increase the computational time of the program. Additionally, the number of percent chord values is limited to 26. Lastly,
there are a few other input restrictions which need to be referenced when creating your input data file. The next section delineates each respective input and declares any restrictions.

III. INPUT DESCRIPTION

There are relatively few input values required for this program. Their description and program variable names are listed below. The user's first task before running this program will be to create an input data file corresponding to the respective planform to be analyzed and the desired program specifications.

LINE 1: $INPT1
Type line as indicated. This lines cues the program for input of data.

LINE 2: XM =
Mach Number of freestream.

LINE 3: NOM =
Number of additional Mach Numbers other than XM (NOM≤5).

LINE 4: NOPCT =
Number of percent chord values for TZORD input (NOPCT≤26).

LINE 5: TPCT =
Table of percent chord values, corresponding to NOPCT, in increasing order from 0 to 100.

LINE 6: JBYMAX =
Number of spanwise stations at which TZORD is to be specified (JBYMAX≤51).
LINE 7:  TYB2  
Table of semispan fractions, corresponding to JBYMAX, in increasing order from 0 to 1.0.

LINE 8:  TZORD  
Zc coordinates of right-hand wing panel corresponding to TYB2 and TPCT. All values of Zc at a given semispan station entered in order according to TPCT, 26 values required to fill a table column. You must enter 26 values per column although only NOPCT values are used. After the first column is filled, repeat with other TYB2 stations, proceeding to right-hand wing tip.

LINE 9:  REFAR  
Wing reference area.

LINE 10:  SPAN  
Total wing span.

LINE 11:  XLEO  
X coordinate of wing leading edge of y=0.

LINE 12:  XTEO  
X coordinate of wing trailing edge at y=0.

LINE 13:  XMAX  
Largest value of x in wing definition.

LINE 14:  XO  
Distance from some arbitrary location to wing apex. XO=0, if you are considering the wing only. This term is used in locating streamwise lift distribution with respect to XO rather than wing apex.

LINE 15:  TYPEX  
= 0. Input TTXLE and TXTE tables.
= 1. Input NLEX, NTEX and tables of TBLEX, TBLEY, TBTEX, TBTEY.
LINE 16: TXLE =
Table of wing leading edge x coordinates at successive values of
y=((SPAN/2)/NON)*N where N = 1, 2, 3, ...NON. (Omit if TYPEX = 1.)

LINE 17: TXTE =
Table of wing trailing edge x coordinates specified at same values of y as TXLE.
(Omit if TYPEX = 1.)

LINE 18: NLEX =
Number of leading edge (x;y) points to be input (NLEX<15). (Omit if TYPEX = 0.)

LINE 19: NTEX =
Number of trailing edge (x,y) points to be input (NTEX<15). (Omit if TYPEX = 0.)

LINE 20: TBLEX =
Table of NLEX leading edge x values(spanwise, root to tip).(Omit if TYPEX =0.)

LINE 21: TBLEY =
Table of NLEX leading edge y values(spanwise, root to tip).(Omit if TYPEX =0.)

LINE 22: TBTEX =
Table of NTEX trailing edge x values(spanwise, root to tip).(Omit if TYPEX =0.)

LINE 23: TBTEY =
Table of NTEX trailing edge y values(spanwise, root to tip).(Omit if TYPEX =0.)

LINE 24: CBAR =
Reference length used for pitching moment coefficient.

LINE 25: XMREF =
X distance from X=0. locating pitching moment center.
LINE 26:   \texttt{NON} =
Number of semispan grid elements selected to represent the wing. (NON\leq50 or 
NON\leq47.5\ast B\ast\text{SPAN}/XMAX (whichever value is less).

LINE 27:   \texttt{$END$}
Line statement ends input of data.

\textbf{IV. SAMPLE PROBLEMS}

Two sample problems have been included in this user's guide section. Both consider the same planform shape, but the input method of the planform shape is different. Only one set of plots exists in the sample problems output file section in that the two sets of plots are identical.

\textbf{V. STARTING THE PROGRAM}

Begin with the screen showing the DCL prompt, which looks like this:
\begin{verbatim}
$
\end{verbatim}

Next, enter the following command:
\begin{verbatim}
SET DEF [.SUPER] [Return]
\end{verbatim}

Now, enter the command to run the program:
\begin{verbatim}
RUN SUPER [Return]
\end{verbatim}

The program will start and the screen should look similar to what is shown is Figure 35.
VI. SAMPLE GRAPHICAL OUTPUTS

A. EXAMPLE PROBLEM 1

Enter the name of the input data file.

SSVL1.DAT [Return]

Once the program has finished its data tabulations, your screen should be similar to Figure 36.

Figure 36. Printing Determination Screen

Respond negatively to print request by typing:

N [Return]
Respond affirmatively to the request to copy the output data file (OUTFILE.DAT) to another file by typing:

Y [Return]

A screen similar to Figure 37 will then appear which lists the file choices possible for copying.

<table>
<thead>
<tr>
<th>WHAT NAME WOULD YOU LIKE FOR THE OUTPUT FILE?</th>
</tr>
</thead>
<tbody>
<tr>
<td>1) TOMCAT.DAT</td>
</tr>
<tr>
<td>2) PHANTOM.DAT</td>
</tr>
<tr>
<td>3) INTRUDER.DAT</td>
</tr>
<tr>
<td>4) CRUSADOR.DAT</td>
</tr>
</tbody>
</table>

ENTER 1, 2, 3 OR 4

Figure 37. Output File Designation Screen

Select from the designated list of file names your choice.

1,2,3, OR 4 [Return]

Respond affirmatively to the request to graph the results by typing:

Y [Return]

The screen should now look like Figure 38.
WHICH OF THE FOLLOWING RELATIONSHIPS DO YOU WANT PLOTTED?

1) SPANWISE PRESSURE DISTRIBUTION
2) CHORDWISE PRESSURE DISTRIBUTION
3) DRAG POLAR (CL VS. CD)
4) STREAMWISE LIFT DISTRIBUTION
5) SPANWISE LIFT DISTRIBUTION
6) NONE

INPUT OPTION NO. (1, 2, 3, 4, 5 OR 6)

Figure 38. Plot Determination Screen

Select from the designated list of graphical relationships your choice.

1,2,3,4, OR 5 [Return]

The requested plot will then appear on your screen and you will be asked if you want to print the plot. If you have remoted your terminal to terminal "KELLY" for printing purposes your plot will come up on the "KELLY" monitor. Compare your plot with the example plots corresponding to SAMPLE # 1; it should be the same.

Respond negatively to the request to print the plot by typing:

N [Return]
The user will again be given the opportunity to graph another relationship (Figure 22 will be presented). Respond with the 6th choice to exit the graphing loop. Enter:

6 [Return]

The program now asks if you want to make another run. Enter

1 [Return]

B. EXAMPLE PROBLEM 2

The screen should again look like Figure 35.

Enter the name of the input data file.

EXPROB2.DAT [Return]

Respond negatively to the request for a printed copy of the output file by typing:

N [Return]

Respond negatively to the request to copy the output data file (OUTFILE.DAT) to another file by typing:

N [Return]

Respond affirmatively to the request to graph the results by typing:

Y [Return]

Again, Figure 38 will appear on your screen with a listing of the available plotting routines.
Select from the list your plotting choice.

1, 2, 3, 4, OR 5 [Return]

The requested plot will appear on your screen. Again, if you have remoted your terminal to terminal "KELLY" for printing purposes your plot will come up on the "KELLY" monitor. Compare your plot with the example plots corresponding to SAMPLE#2; it should be the same.

Respond negatively to the request to print the plot by typing:

N [Return]

The user will again be given the opportunity to graph another relationship. Respond with the 6th choice to exit the graphing loop. Enter:

6 [Return]

Respond negatively to the request to perform another run of program by typing:

2 [Return]

This completes the sample problems for the SUPER program. Graphical output examples created by these sample runs are shown in Figures 39 through 42. These plots were created from the analysis of a B2 Bomber planform at a Mach of 1.2. The span used was 200 feet with a planform reference area of 8260.4 ft². Thirty semispan grid elements were used to represent the wing.
CHORDAL FRACTION \( \frac{X}{L} = 0.690 \)

Figure 39. Spanwise Cp Distribution
Figure 40. Chordwise Cp Distribution
Figure 41. Drag Polar
Figure 42. Spanwise Lift Distribution
APPENDIX F. PROGRAM NEW_DOUBLE COMPUTER CODE

PROGRAM NEW_DOUBLE

*** MODIFIED FOR USE ON THE MICROVAX/2000 BY J.A. CAMPBELL (JUL 88)
UPDATES MADE BY C.M. MACALLISTER JAN-JUL 89 (CMM)

INCOMPRESSIBLE AERODYNAMICS OF SYMMETRIC AIRFOIL
AT ZERO ANGLE OF ATTACK BY LINE DOUBLET DISTRIBUTION

ORIGINAL IBM MAINFRAME PROGRAM WAS ADAPTED FROM JACK MORAN'S BOOK
'AN INTRODUCTION TO THEORETICAL AND COMPUTATIONAL AERODYNAMICS',
WILEY AND SONS, NEW YORK 1984. THE LISTING IS FOUND ON PAGE 75.

PROGRAM FLEXIBILITY AND USER INTERFACE WAS REVISED FOR
PROFESSOR J.V. HEALEY BY JOHN CAMPBELL.
ADDITIONAL PROGRAM UPDATES TO INCLUDE DOUBLET USE FOR ANY
ARBITRARY 2-D SHAPE, PRINTING ROUTINES, PROCESSING CORRECTIONS,
AND GRAPHICAL ANALYSIS WERE MADE BY CRAIG MACALLISTER IN
JAN-JUL 1989. (CMM)

CHARACTER*1 IANS,PRINT,GRAPH,PLOT1,PLOT2,
+PLOT3,CHECK,CORRECT
INTEGER NANS,DATPO,PRINTOPT,GRAPHOPT
REAL*4 T(100),M(100),XS,XF
REAL XX,CP
INTEGER N,R,NPRINT
COMMON /GRAPH/XX,CP,NPRINT
COMMON /MAIN/ T,M,N,XS,XF
COMMON /GRAPHER/GRAPHOPT,XMAXY
COMMON /FCN/AX,TAU,NTYPE
COMMON /DATA/COORX(101),COORY(101)
COMMON /PROB/DATPO
DIMENSION NUM(100)
REAL MPLOT

OPEN FILE FOR DOUBLET STRENGTH DISTRIBUTION OUTPUT
OPEN (UNIT=11,
2 FILE= 'DUBLET.DAT',
2 ORGANIZATION= 'SEQUENTIAL',
2 ACCESS= 'SEQUENTIAL',
2 RECORDDTYPE = 'VARIABLE',
2 FORM= 'FORMATTED',
2 STATUS= 'UNKNOWN')

OPEN FILE FOR BODY SHAPE OUTPUT
OPEN (UNIT=12,
2 FILE= 'SHAPE.DAT',
2 ORGANIZATION= 'SEQUENTIAL',
2 ACCESS= 'SEQUENTIAL',
2 RECORDDTYPE = 'VARIABLE',

95
OPEN FILE FOR BODY SURFACE PRESSURE DISTRIBUTION OUTPUT
OPEN (UNIT=13, 
FILE= 'PRESSURE.DAT', 
ORGANIZATION= 'SEQUENTIAL', 
ACCESS= 'SEQUENTIAL', 
RECORDTYPE= 'VARIABLE', 
FORM= 'FORMATTED', 
STATUS= 'UNKNOWN')

OPEN ANOTHER FILE FOR BODY SURFACE PRESSURE DISTRIBUTION OUTPUT
OPEN (UNIT=14, 
FILE= 'PRESS.DAT', 
ORGANIZATION= 'SEQUENTIAL', 
ACCESS= 'SEQUENTIAL', 
RECORDTYPE= 'VARIABLE', 
FORM= 'FORMATTED', 
STATUS= 'UNKNOWN')

OPEN ANOTHER FILE FOR BODY SHAPE OUTPUT
OPEN (UNIT=15, 
FILE= 'SHAPEBODY.DAT', 
ORGANIZATION= 'SEQUENTIAL', 
ACCESS= 'SEQUENTIAL', 
RECORDTYPE= 'VARIABLE', 
FORM= 'FORMATTED', 
STATUS= 'UNKNOWN')

CALL LIBRARY ROUTINE TO CLEAR THE SCREEN, THE PRINT HEADER
5 CONTINUE
CALL CLRSCRN
PRINT *
PRINT *, ' PROGRAM DUBLET : VERSION 3 : 4 OCTOBER 89 '
PRINT *
PRINT *, ' DOUBLET DISTRIBUTION METHOD IS USED TO DETERMINE' 
PRINT *, ' INCOMPRESSIBLE AERODYNAMICS OF AN ELLIPSE, SYMMETRICAL' 
PRINT *, ' AIRFOIL OR ARBITRARY SYMMETRIC SHAPE AT ZERO ANGLE' 
PRINT *, ' OF ATTACK'
PRINT *, ' 
PRINT *, ' PROGRAM ASSUMES A NONDIMENSIONAL CHORD, THAT IS,' 
PRINT *, ' THE VALID RANGE OF X IS FROM 0 TO 1.'
PRINT *
10 PRINT *, ' ENTER TYPE OF BODY SHAPE DESIRED: '
PRINT *, ' 1) ELLIPTIC'
PRINT *, ' 2) SYMMETRICAL AIRFOIL-LIKE OR'
PRINT *, ' 3) ARBITRARY SYMMETRIC SHAPE'
PRINT *, ' ENTER 1, 2, OR 3.'
PRINT *, ' 
PRINT *, ' NOTE THAT OPTION 3 WILL REQUIRE MANUALLY INPUTTING DATA'
PRINT *, ' POINTS FOR THE UPPER SIDE OF THE RESPECTIVE BODY'
15 READ (5,*) NTYPEx
IF (NTYPE .LT. 1 .OR. NTYPE .GT. 3) THEN
PRINT *, ' INVALID ENTRY. ENTER 1, 2, OR 3.'
GO TO 15
END IF
IF (NTYPE .EQ. 3) THEN
CALL CLRSCRN
PRINT *, 'HOW MANY UPPER PROFILE DATA POINTS DO'
PRINT *, 'YOU DESIRE? (ENTER A NUMBER BETWEEN 3 AND 100)'
PRINT *, '
PRINT *, 'BE AWARE THAT THE LEADING EDGE OF YOUR DESIRED'
PRINT *, 'SHAPE HAS BEEN PROGRAMMED TO BE AT THE ORIGIN'
PRINT *, 'AND THAT YOUR TRAILING EDGE IS AT (1,0). SCALE'
PRINT *, 'YOUR SHAPE/OBJECT ACCORDINGLY.'
PRINT *, '17 READ (5,*) DATPO
IF (DATPO .LT. 3 .OR. DATPO .GT. 100) THEN
PRINT *, 'INVALID ENTRY. ENTER A NUMBER BETWEEN'
PRINT *, 'THREE(3) AND 100 INCLUSIVE.'
GO TO 17
END IF
DO 26 R = 1, DATPO
COORX(1) = 0.0
COORX(DATPO+2) = 1.0
WRITE (5,27) R
27 FORMAT (1X,'ENTER X(',I2,')')
READ (5,*) COORX(R+1)
COORY(1) = 0.0
COORY(DATPO+2) = 0.0
WRITE (5,28) R
28 FORMAT (1X,'ENTER Y(',I2,')')
READ (5,*) COORY(R+1)
26 CONTINUE
PRINT *, '
PRINT *, 'WOULD YOU LIKE TO CHECK YOUR SURFACE DATA POINTS? '
PRINT *, ' (Y/N)'
READ 1002, CHECK
IF (CHECK .EQ. 'Y'. OR. CHECK .EQ. 'y') THEN
313 CALL CLRSCRN
DO 65 I = 1, DATPO+2
WRITE(5,29) I,COORX(I),COORY(I)
29 FORMAT (5X,I3,3X,F8.4,3X,F8.4,/) 
65 CONTINUE
PRINT *, 'WOULD YOU LIKE TO MAKE ANY CORRECTIONS?'
PRINT *, ' (Y/N)'
PRINT *, '
READ 1002, CORRECT
IF (CORRECT .EQ. 'Y'. OR. CORRECT .EQ. 'y') THEN
PRINT *, '
PRINT *, 'WHICH DATA POINT WOULD YOU LIKE TO CORRECT?'
NUMBERS = DATPO + 2
WRITE (5,30) NUMBERS
30 FORMAT (5X,'ENTER A NUMBER 1 THRU',I4,' INCLUSIVE')
312 READ (5,*) NUMCOR
IF (NUMCOR .LT. 1. OR. NUMCOR .GT. NUMBERS) THEN
PRINT *, 'INVALID ENTRY '
WRITE (5,30) NUMBERS
PRINT *, '
GO TO 312
ENDIF
WRITE (5,27)NUMCOR
READ(5,*)COORX(NUMCOR)
WRITE (5,28)NUMCOR
READ(5,*)COORY(NUMCOR)
GO TO 313
ENDIF
ENDIF
GO TO 70
ENDIF
PRINT *, ' ENTER THICKNESS RATIO (TAU).'</READ (5,*) TAU
IF (NTYPE .GT. 1) THEN
PRINT *
PRINT *, ' ENTER THE NONDIMENSIONAL X LOCATION OF MAXIMUM', + ' THICKNESS.'
READ (5,*) XMAXY
IF (XMAXY .GT. 0.5) THEN
PRINT *, ' THE PROGRAM CONSIDERS THE ONSET FLOW TO BE' + ' APPROACHING FROM THE LEFT. THEREFORE, THE' + ' X LOCATION OF MAXIMUM THICKNESS MUST BE < 0.5.' + ' ==> PLEASE REENTER.'
GO TO 20
END IF
AX = (.5 * TAU)/(SQRT(XMAXY)*(1 - XMAXY))
END IF
C
C INPUT NUMBER OF INTERVALS N
C
70 CALL CLRSCRN
PRINT *
PRINT *, ' ENTER NUMBER OF INTERVALS DESIRED. N ='
71 READ (5,*) N
PRINT *
IF(N .LT. 2 .OR. N .GT. 100) THEN
WRITE(6,21) N
PRINT *, ' A MINIMUM OF TWO INTERVALS AND A MAXIMUM OF' + ' 100 IS ALLOWED. ==> PLEASE REENTER.'
GO TO 71
END IF
21 FORMAT(1X,5X,'NUMBER OF INTERVALS REQUESTED =',I3)
C
C ASK USER FOR AUTOMATIC OR MANUAL DETERMINATION OF ENDPOINTS.
80 CONTINUE
CALL CLRSCRN
PRINT *
PRINT *, ' WHICH METHOD DO YOU WISH TO USE TO DETERMINE THE' + ' DOUBLET DISTRIBUTION ENDPOINTS? '
PRINT *, ' 1) PROGRAM INTERVAL HALVING SUBROUTINE TO ITERATE.' + ' 2) MANUAL ITERATION BY THE USER.' + ' 3) RETURN TO START' + ' 4) EXIT PROGRAM' + ' ENTER 1,2,3 OR 4'
PRINT *
READ (5,*) NMETH
IF (NMETH .LT. 1 .OR. NMETH .GT. 4) THEN
C C MANUALLY DETERMINE ENDPOINTS OF SOURCE DISTRIBUTION XS, XF

100 CONTINUE
CALL CLRSCRN
PRINT *
PRINT *, ' ROUTINE FOR MANUAL DETERMINATION OF ENDPOINTs'
PRINT *, '--------------------------------------'
PRINT *, ' ENTER THE DOUBLET DISTRIBUTION STARTING POINT, XS.'
PRINT *, ' (XS SHOULD BE APPTIMATELY ONE HALF OF'
PRINT *, ' THE NONDIMENSIONAL LEADING EDGE RADIUS.)'
READ (5,* ) XS
PRINT *
PRINT *, ' ENTER THE DOUBLET DISTRIBUTION ENDING POINT, XF.'
PRINT *, ' (XF SHOULD BE APPTIMATELY ONE MINUS HALF'
PRINT *, ' OF THE NONDIMENSIONAL TRAILING EDGE RADIUS.)'
READ (5,* ) XF
PRINT *
PRINT *
CALL FINDM (T,M,N,XS, XF)
CALL PRESS(0.0,U0,CP0)
CALL PRESS(1.0,U1,CP1)
GO TO 150

120 CONTINUE
CALL CLRSCRN
PRINT *
PRINT *, ' INTERVAL HALVING ROUTINE FOR DETERMINATION OF'
PRINT *, ' DOUBLET DISTRIBUTION ENDPOINTS'
PRINT *, '--------------------------------------'
C C ENTER THE PARAMETERS REQUIRED BY THE INTERVAL HALVING METHOD
C WHICH IS USED TO OBTAIN THE PROPER LOCATIONS FOR XS AND XF.
PRINT *, ' ENTER THE INTEGER EXPONENT FOR THE X TOLERANCE, NXTOL.'
PRINT *, ' EXAMPLE: A VALUE OF 4, GIVES A TOLERANCE OF 0.0001.'
READ (5,* ) NXTOL
PRINT *
PRINT *, ' ENTER THE INTEGER EXPONENT FOR THE FUNCTION ',
PRINT *, '& TOLERANCE, NFTOL.'
PRINT *, ' (SAME IDEA AS NXTOL; 5 YIELDS FTOL = 0.00001).'
READ (5,* ) NFTOL
PRINT *
PRINT *, ' ENTER THE MAXIMUM NUMBER OF ITERATIONS, MAXIT, TO '
PRINT *, ' LOCATE XS AND XF. (FOR NFTOL = 6, SUGGEST 35-40)
READ (5,* ) MAXIT
PRINT *
PRINT *, ' ENTER THE OUTPUT PARAMETER, IOUT.'
PRINT *, 'IOUT = 0 TO SUPPRESS ALL ITERATION RELATED OUTPUT'
PRINT *, ' 1 TO OUTPUT FINAL RESULTS ONLY'
PRINT *, ' 2 TO OUTPUT DETAILS FOR EACH ITERATION'
READ (5,*) IOUT
CALL INTHV (NXTOL,NFTOL,NTYPE,MAXIT,IOUT,UO,U1)
C  RUN THROUGH PROCESS AGAIN WITH FINAL VALUES OBTAINED BY ITERATION
CALL FINDM (T,N,N,XS,XF)
CALL PRESS(0.0,UO,CPO)
CALL PRESS(1.0,U1,CP1)
C 150 PRINT *, 'U AT X = 0 =',U0,'  XS =',XS
PRINT *, 'U AT X = 1 =',U1,'  XF =',XF
PRINT *, 'THESE VALUES FOR U SHOULD BE NEAR ZERO.'
PRINT *, 'DO YOU ACCEPT THESE RESULTS (Y/N)'
READ 1000, IANS
IF (IANS .EQ. 'N') THEN
  PRINT *, 'CORRECTION LINE NO. 1'
  GO TO (120,100) NMETH
ELSE
  GO TO 152
END IF
C  C  OUTPUT RESULTS
C 152 PRINT 1010
WRITE (11,1012)
M(N+1) = 0.0
DO 200 I = 1,N+1
MPLOT = REAL(M(I)*3.1415926585)
PRINT 1040, T(I),MPLOT
200 WRITE (11,1040) T(I),MPLOT
CLOSE (UNIT=11)
PRINT 1020
WRITE (12,1020)
IF (NTYPE .LE. 2) THEN
  DO 210 I = 1,N
    XX = .5*(T(I) + T(I+1))
    YY = Y(XX)
    PRINT 1040, XX,YY
    WRITE (15,1040) XX,YY
  210 XX = 1.0
    YY = 0.0
    WRITE (15,1040) XX,YY
ENDIF
IF (NTYPE .EQ. 3) THEN
  DO 211 I = 1,DATPO+2
    XX = COORX(I)
    YY = COORY(I)
    PRINT 1040, XX,YY
    WRITE (15,1040) XX,YY
  211 XX = 1.0
    YY = 0.0
    WRITE (15,1040) XX,YY
ENDIF
CLOSE (UNIT=12)
CLOSE (UNIT=15)
PRINT 1030
READ (5,*) NPRINT
IF (NPRINT .LT. 2) THEN
   PRINT *, ' YOU MUST ENTER A MINIMUM OF 2. PLEASE REENTER.'
   GO TO 212
END IF
WRITE (13,1032)
DO 220 I = 1, NPRINT
   XX = (I-1)/FLOAT(NPRINT-1)
   CALL PRESS(XX,U,CP)
   PRINT 1040, XX, CP
WRITE (14,1040) XX, CP
220 WRITE (13,1040) XX, CP
CLOSE (UNIT = 13)
CLOSE (UNIT=14)
C CALL LIBRARY ROUTINE TO CLEAR THE SCREEN, THEN PRINT HEADER
CALL CLRSCRN
PRINT *, ' PROGRAM DUBLET RESULTS HAVE BEEN WRITTEN TO FILES:'
PRINT *
PRINT *, ' DUBLET.DAT : DOUBLET STRENGTH DISTRIBUTION'
PRINT *, ' SHAPE.DAT : BODY SURFACE COORDINATES'
PRINT *, ' PRESSURE.DAT: SURFACE PRESSURE DISTRIBUTION'
PRINT *
PRINT *, 'WOULD YOU LIKE TO PRINT THE RESULTS (Y/N)?'
PRINT *
READ 1002, PRINT
IF (PRINT.EQ.'Y'.OR.PRINT.EQ.'y')THEN
   PRINT *, 'WHICH OF THE FOLLOWING FILES DO YOU WANT?'
   PRINT *
   PRINT *, ' 1) DUBLET.DAT'
   PRINT *, ' 2) PRESSURE.DAT'
   PRINT *, ' 3) SHAPE.DAT'
   PRINT *, ' OR 4) ALL THREE FILES'
   PRINT *
   PRINT *, 'INPUT OPTION NO.(1,2,3, OR 4)' 12
READ 1006, PRINTOPT
IF (PRINTOPT .LT. 1 .OR. PRINTOPT .GT. 4) THEN
   PRINT *, 'INVALID ENTRY, ENTER INTEGER BETWEEN'
   PRINT *, 'ONE(1) AND FOUR(4).'
   PRINT *, '
   GO TO 12
ENDIF
ENDIF
IF (PRINTOPT .EQ. 1) THEN
   CALL LIB$SPAWN('PRINT DUBLET.DAT')
ENDIF
IF (PRINTOPT .EQ. 2) THEN
   CALL LIB$SPAWN('PRINT PRESSURE.DAT')
ENDIF
IF (PRINTOPT .EQ. 3) THEN
   CALL LIB$SPAWN('PRINT SHAPE.DAT')
ENDIF
IF (PRINTOPT .EQ. 4) THEN
CALL LIB$SPAWN('PRINT DUBLET.DAT, PRESSURE.DAT, SHAPE.DAT')
ENDIF
PRINT *
PRINT *
PRINT *, 'WOULD YOU LIKE TO GRAPH THE RESULTS (Y/N)'
PRINT *
READ 1002, GRAPH
IF (GRAPH.EQ. 'Y' .OR. GRAPH.EQ. 'y') THEN
PRINT *
PRINT *, 'WHICH OF THE FOLLOWING DATA FILES'
PRINT *, 'DO YOU WANT TO GRAPH?'
PRINT *, '1) DUBLET.DAT'
PRINT *, '2) PRESSURE.DAT'
PRINT *, '3) SHAPE.DAT'
PRINT *, '4) NONE'
PRINT *
PRINT *, 'INPUT OPTION NO. (1, 2, 3 OR 4)'
READ 1006, GRAPHOPT
IF (GRAPHOPT .LT. 1 .OR. GRAPHOPT .GT. 4) THEN
PRINT *, 'INVALID ENTRY, ENTER INTEGER BETWEEN'
PRINT *, 'ONE(1) AND FOUR(4).'
PRINT *, '
GO TO 616
ENDIF
IF (GRAPHOPT .EQ. 1) THEN
CALL GRAPH1(NTYPE, XMAXY, TAU)
ENDIF
GET A HARDCOPY OF THIS GRAPHIC
CALL LIB$SPAWN('RENDER/DEVICE=LA210/DRAFT_QUALITY/PAPER_'
+SIZE=A Pl.UIS')
CALL LIB$SPAWN('CONTINUE')
PRINT *, '
PRINT *, 'WOULD YOU LIKE A PRINT OF THIS PLOT? (Y/N)'
PRINT *
READ 1002, PLOT1
IF (PLOT1.EQ. 'Y' .OR. PLOT1.EQ. 'y') THEN
CALL LIB$SPAWN('PRINT P1.REN')
ENDIF
GO TO 46
ENDIF
IF (GRAPHOPT .EQ. 2) THEN
CALL GRAPH2(NTYPE, XMAXY, NPRINT, TAU, N)
CALL LIB$SPAWN('RENDER/DEVICE=LA210/DRAFT_QUALITY/PAPER_'
+SIZE=A P2.UIS')
PRINT *, '
CALL LIB$SPAWN('CONTINUE')
PRINT *, 'WOULD YOU LIKE A PRINT OF THIS PLOT? (Y/N)'
PRINT *
READ 1002, PLOT2
IF (PLOT2.EQ. 'Y' .OR. PLOT2.EQ. 'y') THEN
CALL LIB$SPAWN('PRINT P2.REN')
ENDIF
GO TO 46
ENDIF
IF (GRAPHOPT .EQ. 3) THEN
CALL GRAPH3(NTYPE, XMAXY, N, TAU, DATPO)
CALL LIB$SPAWN('RENDER/DEVICE=LA210/DRAFT_QUALITY/PAPER_SIZE=A3_UIS')
PRINT *, ''
CALL LIB$SPAWN('CONTINUE')
PRINT *, 'WOULD YOU LIKE A PRINT OF THIS PLOT? (Y/N)'
READ 1002, PLOT3
IF (PLOT3.EQ.'Y'.OR.PLOT3.EQ.'Y')THEN
  CALL LIB$SPAWN('PRINT P3.REN')
ENDIF
GO TO 46
ENDIF
IF (GRAPHOPT.EQ.4) THEN
  GO TO 64
ENDIF
ENDIF
C OPTION TO MAKE ANOTHER RUN
64
  PRINT *
  PRINT *, 'DO YOU WISH TO: '
  PRINT *, '1) MAKE ANOTHER RUN OR'
  PRINT *, '2) END THIS SESSION'
  PRINT *, 'ENTER 1 OR 2.'
  PRINT *
  CALL QUERY (NANS)
  CALL CLRSCRN
  IF (NANS.EQ.1) GO TO 10
999 STOP
1000 FORMAT(A1)
1002 FORMAT(A1)
1006 FORMAT(I1)
1010 FORMAT(/, 'DOUBLET STRENGTH DISTRIBUTION',/,
  + 'M = M(I) FOR T(I) .LT. T .LT. T(I+1)',/,
  + '5X,'T(I)',5X,'M(I)/2',/)
1012 FORMAT(/,9X, 'DOUBLET STRENGTH DISTRIBUTION',/,
  + '14X,'T(I)',5X,'M(I)/2',/)
1020 FORMAT(/,9X, 'BODY SHAPE - UPPER SURFACE',/,
  + '15X,'X',9X,'Y',/)
1030 FORMAT(/, 'BODY SURFACE PRESSURE DISTRIBUTION',/,
  + '6X,'X',8X,'CP',/,'INPUT NUMBER OF PRESSURE COEFFICIENT',
  + 'OUTPUT POINTS')
1032 FORMAT(/,9X, 'BODY SURFACE PRESSURE DISTRIBUTION',/,
  + '16X,'X',8X,'CP',/)
1040 FORMAT(9X,2F10.4)
END

SUBROUTINE CLRSCRN
C LIBRARY ROUTINE TO CLEAR THE SCREEN.

C ISTAT = LIB$ERASE_PAGE (1,1)
RETURN
END

C SUBROUTINE QUERY(NANS)
C ROUTINE TO TRAP ERRORS CAUSED BY IMPROPER RESPONSES TO QUESTIONS.
C THE COMPUTER GENERATES AND ERROR WHEN A CHARACTER IS SUPPLIED TO
C A QUESTION EXPECTING AN INTEGER OR REAL VALUE.

NQTEST=0
1 CONTINUE
IF (NQTEST .GT. 0) THEN
  PRINT *, ' CHARACTER VALUES ARE NOT VALID.'
  PRINT *, ' PLEASE ENTER AN INTEGER VALUE.'
END IF
NQTEST = NQTEST + 1
READ (5,*,ERR=1)NANS
RETURN
END

SUBROUTINE FINDM (T,M,N,XS,XF)

C FIND DOUBLET STRENGTH TO MEET
C FLOW TANGENCY CONDITION

REAL*4 T(100),M(100),XS,XF
INTEGER N,R
COMMON /COF/ A(101,111),NEQNS
PI = 3.1415926585
NP = N + 1
DO 100 I = 1,NP
C COSINE SPACING SCHEME FROM XS TO XF
FRAC = .5*(1. - COS(PI*(I-1)/FLOAT(N)))
100 T(I) = XS + (XF - XS)*FRAC
C
SET UP LINEAR SYSTEM OF EQUATIONS

DO 210 I = 1,N
XI = .5*(T(I) + T(I+1))
YI = Y(XI)
FAC1 = ATAN2(T(I) - XI,YI)
DO 200 J = 1,N
FAC2 = ATAN2(T(J+1) - XI,YI)
A(I,J) = (FAC2 - FAC1)/YI
FAC1 = FAC2
200 CONTINUE
A(I,NP) = 1.0
210 CONTINUE
C
SOLVE FOR DOUBLET STRENGTH

NEQNS = N
CALL GAUSS(1)
DO 300 I = 1,N
300 M(I) = A(I,NP)
RETURN
END

SUBROUTINE FIX(VALMAX,VALMIN)
C ARRAY = THE ARRAY WHICH IS BEING SORTED INTO ASCENDING ORDER
C NUMBER= THE NUMBER OF ELEMENTS IN THE ARRAY TO BE SORTED
C VALMAX= LARGEST VALUE IN THE ARRAY
C VALMIN= SMALLEST VALUE IN THE ARRAY
REAL VALMAX, VALMIN
INTEGER NUMBER
LOGICAL SORTED
COMMON /JACKEL/YPLOT, NP
DIMENSION ARRAY(100), YPLOT(100)
SORTED = .FALSE.
NUMBER = NP
DO 20 I = 1, NUMBER
   ARRAY(I) = YPLOT(I)
20 CONTINUE
30 IF (.NOT. SORTED) THEN
   SORTED = .TRUE.
   DO 40 I = 1, NUMBER - 1
      IF (ARRAY(I) .GT. ARRAY(I+1)) THEN
         VALUE = ARRAY(I)
         ARRAY(I) = ARRAY(I+1)
         ARRAY(I+1) = VALUE
         SORTED = .FALSE.
      ENDIF
40 CONTINUE
ENDIF
GO TO 30
END

VALMAX = ARRAY(NUMBER)
VALMIN = ARRAY(1)

* THE FOLLOWING FILE IS CREATED TO CHECK THE VALIDITY OF THIS ROUTINE
OPEN (UNIT=26, FILE='ARRAY3.DAT', STATUS='NEW')
DO 45 I = 1, NUMBER
   WRITE (17, 55) ARRAY(I)
45 CONTINUE
WRITE (17, 65) VALMAX, VALMIN, NUMBER
55 FORMAT (1X, E11.4)
65 FORMAT (/, 1X, 'VALMAX = ', F6.4, '/', 'VALMIN = ', E11.4, '/', 'NUMBER = ', I3)
CLOSE (UNIT=26)
RETURN
END

SUBROUTINE GAUSS (NRHS)

C SOLUTION OF LINEAR ALGEBRAIC SYSTEM BY
C GAUSS ELIMINATION WITH PARTIAL PIVOTING
C
C A       = COEFFICIENT MATRIX
C NEQNS    = NUMBER OF EQUATIONS
C NRHS     = NUMBER OF RIGHT HAND SIDES
C
C RIGHT-HAND SIDES AND SOLUTIONS STORED IN
C COLUMNS NEQNS+1 THRU NEQNS+NRHS OF 'A
C
COMMON DX, DY, AR, PI
COMMON /COF/ A(350, 351), NEQNS
NP = NEQNS + 1
NTOT = NEQNS + NRHS

C GAUSS REDUCTION

105
DO 150  I = 2,NEQNS
   -- SEARCH FOR LARGEST ENTRY IN (I-1)TH COLUMN
   -- ON OR BELOW MAIN DIAGONAL
   IM     = I - 1
   IMAX   = IM
   AMAX   = ABS(A(IM,IM))
   DO 110  J = I,NEQNS
      IF (AMAX .GE. ABS(A(J,IM))) GO TO 110
      IMAX  = J
      AMAX  = ABS(A(J,IM))
   110 CONTINUE
   -- SWITCH (I-1)TH AND IMAXTH EQUATIONS
   IF (IMAX .NE. IM) GO TO 140
   DO 130  J = IM,NTOT
      TEMP  = A(IM,J)
      A(IM,J) = A(IMAX,J)
      A(IMAX,J) = TEMP
   130 CONTINUE
  ELIMINATE (I-1)TH UNKNOWN FROM
  ITH THRU (NEQNS)TH EQUATIONS
  DO 150  J = I,NEQNS
      R = A(J,IM)/A(IM,IM)
      DO 150  K = I,NTOT
         A(J,K) = A(J,K) - R*A(IM,K)
   150 CONTINUE
  BACK SUBSTITUTION
  DO 220  K = NP,NTOT
      A(NEQNS,K) = A(NEQNS,K)/A(NEQNS,NEQNS)
         DO 210  L = 2,NEQNS
            I     = NEQNS + 1 - L
            IP    = I + 1
         DO 200  J = IP,NEQNS
            A(I,K) = A(I,K) - A(I,J)*A(J,K)
      200 CONTINUE
      A(I,K) = A(I,K)/A(I,I)
   220 CONTINUE
RETURN
END

SUBROUTINE GRAPH1(NTYPE,XMAXY,TAU)

DEFINE IPACK ARRAY FOR LEGEND
INTEGER*4 IPACK(35)
REAL*4 T(100),M(100),XS, XF, TAU, XMAXY, MIN, MAX
INTEGER N, R, NTYPE, NP
COMMON /MAIN/T,M,N,XS,XF
COMMON /JACKEL/YPLOT,NP
CHARACTER*40 L1
DIMENSION YPLOT(100)

DEFINE AND ASSIGN LEGEND CHARACTER STRINGS
L1 = 'DOUBLET STRENGTH$'
C INITIALIZE THE GRAPHICS SYSTEM
   CALL INIT
C LABEL X AND Y AXES USING SELF COUNTING STRINGS
   CALL XNAME('X',1)
   CALL YNAME('STRENGTH',8)
C DEFINE PLOT AREA TO BE 6 INCHES BY 8 INCHES
   CALL AREA2D(6.0,8.0)
C DEFINE HEADING LABEL
   CALL HEADING('DOUBLET STRENGTH DISTRIBUTION$',-100,2.,1)
C PLOT ADDITIONAL TICK MARKS
   CALL XTICKS(1)
   CALL YTICKS(1)
C PACK LEGEND LABELS INTO ARRAY IPACK
   CALL LINES(L1,IPACK,1)
C COSINE SPACING SCHEME FROM XS TO XF
   PI = 3.1415926585
   NP = N + 1
   DO 100 I = 1,NP
      FRACT = .5*(1. - COS(PI*(I-1)/FLOAT(N)))
   100 T(I) = XS + (XF - XS)*FRACT
C CREATE THE RESPECTIVE VALUES FOR YPLOT
   DO 207 I = 1,NP+1
      YPLOT(I) = REAL(M(I)*3.1415926585)
   207 CONTINUE
C SET UP AXIS
   CALL GRAF(0.,2.1,(MIN-.1),.05,(MAX+.2))
C FRAME THE SUBPLOT AREA
   CALL FRAME
C PLOT DUBLET STRENGTH DATA WITH A THICK LINE AND MARKER 15
   CALL MARKER(15)
   CALL THKCVR(.04)
   CALL CURVE(T,YPLOT,NP,1)
C PLOT MESSAGES
   IF (NTYPE.EQ.1) THEN
      CALL MESSAG('ELLIPTICAL AIRFOIL DOUBLET DISTRIBUTION$',100,+
                  .5,6.)
      CALL MESSAG('THICKNESS RATIO (TAU) = $',100,.5,5.5)
      CALL REALNO(TAU,2,4.,5.5)
      CALL MESSAG('NUMBER OF INTERVALS USED = $',100,.5,5.)
      CALL INTNO(N,'ABUT','ABUT')
   ENDIF
   IF (NTYPE.EQ.2) THEN
      CALL MESSAG('SYMMETRIC AIRFOIL DOUBLET DISTRIBUTION$',100,+
                  .75,2.5)
      CALL MESSAG('THICKNESS RATIO (TAU) = $',100,.75,2.)
      CALL REALNO(TAU,2,4.,2.)
      CALL MESSAG('MAXIMUM THICKNESS AT X = $',100,.75,1.5)
      CALL REALNO(XMAXY,2,4.1,1.5)
      CALL MESSAG('NUMBER OF INTERVALS USED = $',100,.75,1.)
      CALL INTNO(N,'ABUT','ABUT')
   ENDIF
   IF (NTYPE.EQ.3) THEN
      CALL MESSAG('ARBITRARY SHAPE DOUBLET DISTRIBUTION$
                  + ,100,.75,1.5)
CALL MESSAG('NUMBER OF INTERVALS USED = $',100,.,75,1.)
CALL INTNO(N,'ABUT','ABUT')
ENDIF
C PLOT LEGEND
CALL MYLEGN('DOUBLET STRENGTH$',100)
C PLOT LEGEND
CALL LEGEND(IPACK,1,3.0,7.0)
C END PLOT
CALL ENDPLO)
C CREATE GRAPHICS METAFILE P1.UIS
CALL METAFL(1)
C TERMINATE PLOT AT END OF PLOTTING SESSION
CALL DONEPL
RETURN
END
SUBROUTINE GRAPH2(NTYPE,XMAXY,NPRINT,TAU,N)
C
C DEFINE IPACK ARRAY FOR LEGEND
INTEGER*4 IPACK(35)
INTEGER NUM,NPRINT,NTYPE,N
REAL XX(100),CP(100),MAX,MIN,TAU,XMAXY
CHARACTERS LI
COMMON /ABLE/CP,NUM
C READ ELEMENTS OF UNIT 14 INTO ARRAYS TO PLOT
OPEN(UNIT=14,FILE='PRESS.DAT',STATUS='OLD')
DO 25 I = 1,NPRINT
READ (14,*)XX(I),CP(I)
25 CONTINUE
NUM = NPRINT
CLOSE(UNIT=14)
CALL SCALER2(MAX,MIN)
C DEFINE AND ASSIGN LEGEND CHARACTER STRINGS
L1 = 'CP DISTRIBUTION$
C INITIALIZE THE GRAPHICS SYSTEM
CALL INIT
C LABEL X AND Y AXES USING SELF COUNTING STRINGS
CALL XNAME('X$',100)
CALL YNAME('CP$',100)
C DEFINE PLOT AREA TO BE 6 INCHES BY 8 INCHES
CALL AREA2D(6.0,8.0)
C DEFINE HEADING LABEL
CALL HEADIN('CP DISTRIBUTION$',-100,2.,1)
C PLOT ADDITIONAL TICK MARKS
CALL XTICKS(1)
CALL YTICKS(1)
C PACK LEGEND LABELS INTO ARRAY IPACK
CALL LINES(L1,IPACK,1)
C SET UP AXIS
CALL GRAF(0.0,0.2,1.0,(MIN-.1),((MAX-MIN)/5.),MAX+.1))
C FRAME THE SUBPLOT AREA
CALL FRAME
C PLOT PRESSURE DISTRIBUTION DATA WITH A THICK LINE AND MARKER 15
CALL MARKER(15)
CALL THKCRV(.04)

108
CALL CURVE(XX,CP,NUM,1)

C PLOT MESSAGES
IF (NTYPE.EQ. 1) THEN
  CALL MESSAG('ELLIPtical AIRFOIL CP DISTRIBUTION$',100,
+ .75,4.)
  CALL MESSAG('THICKNESS RATIO (TAU) =$',100,.75,3.5)
  CALL REALNO(TAU,2,4.,3.5)
  CALL MESSAG('NUMBER OF INTERVALS USED =$',100,.75,3.0)
  CALL INTNO(N,'ABUT','ABUT')
ENDIF
IF (NTYPE.EQ. 2) THEN
  CALL MESSAG('SYMMETRIC AIRFOIL CP DISTRIBUTION$',100,
+ .75,6.0)
  CALL MESSAG('THICKNESS RATIO (TAU) =$',100,.75,5.5)
  CALL REALNO(TAU,2,4.1,5.5)
  CALL MESSAG('MAXIMUM THICKNESS AT X =$',100,.75,5.)
  CALL REALNO(XMAXY,2,4.1,5.)
  CALL MESSAG('NUMBER OF INTERVALS USED =$',100,.75,4.5)
  CALL INTNO(N,'ABUT','ABUT')
ENDIF
IF (NTYPE.EQ. 3) THEN
  CALL MESSAG('ARBITRARY SHAPE CP DISTRIBUTION$'
+ .75,5.5)
  CALL MESSAG('NUMBER OF INTERVALS USED =$',100,.75,5.0)
  CALL INTNO(N,'ABUT','ABUT')
ENDIF

C CHANGE LEGEND NAME TO "CP DISTRIBUTION"
CALL MYLEGNC'CP DISTRIBUTION',100)

C PLOT LEGEND
CALL LEGEND(IPACK,1,2.0,7.0)
C END PLOT
CALL ENDPL(0)

C CREATE GRAPHICS METAFILE P2.UIS
CALL METAFL(2)

C TERMINATE PLOT AT END OF PLOTTING SESSION
CALL DONEPL
RETURN
END

SUBROUTINE GRAPH3(NTYPE,XMAXY,N,TAU,DATPO)

C DEFINE IPACK ARRAY FOR LEGEND
INTEGER*4 IPACK(35)
INTEGER NUM,NTYPE,N,DATPO
REAL XX(100),YY(100),MAX,MIN,TAU,XMAY
CHARACTER*40 LI
COMMON /JACK/YY,NUM

C READ ELEMENTS OF UNIT 15 INTO ARRAYS TO PLOTh
OPEN(UNIT=15,FILE='SHAPEBODY.DAT',STATUS='OLD')
IF (NTYPE.LE. 2) THEN
  XX(1) = 0.0
  YY(1) = 0.0
  DO 25 I = 2,N+2
    READ(15,*)XX(I),YY(I)
    25 CONTINUE
  XX(N+3) = 1.0

109
YY(N+3) = 0.0
NUM = N + 3
ENDIF
IF (NTYPE .EQ. 3) THEN
DO 35 I = 1,DATPO+2
    READ(15,*)XX(I),YY(I)
35 CONTINUE
NUM = DATPO +2
ENDIF
CLOSE(UNIT=15)
C CALL SCALER TO FIND THE MAX AND MIN VALUES OF THE Y ORDINATE ARRAY
   CALL SCALER(MAX,MIN)
C DEFINE AND ASSIGN LEGEND CHARACTER STRINGS
   L1 = 'AIRFOIL SHAPE$'
C INITIALIZE THE GRAPHICS SYSTEM
   CALL INIT
C LABEL X AND Y AXES USING SELF COUNTING STRINGS
   CALL XNAME('X$',100)
   CALL YNAME('Y$',100)
C DEFINE PLOT AREA TO BE 6 INCHES BY 8 INCHES
   CALL AREA2D(6.0,8.0)
C DEFINE HEADING LABEL
   CALL HEADINC('AIRFOIL SHAPE$','-100,2.,1)
C PLOT ADDITIONAL TICK MARKS
   CALL XTICKS(1)
   CALL YTICKS(1)
C PACK LEGEND LABELS INTO ARRAY IPACK
   CALL LINES(L1,IPACK,1)
C SET UP AXIS
   CALL GRAF(0.,2.,1.,0.,((MAX-MIN+.4)/5.), (MAX+.4))
C FRAME THE SUBPLOT AREA
   CALL FRAME
C PLOT PRESSURE DISTRIBUTION DATA WITH A THICK LINE AND MARKER 15
   CALL MARKER(15)
   CALL THKCRV(.04)
   CALL CURVE(XX,YY,NUM,1)
C PLOT MESSAGES
   IF (NTYPE.EQ.1) THEN
     CALL MESSAG('ELLIPTICAL AIRFOIL SHAPE$','100,' + 1.,5.)
     CALL MESSAG('THICKNESS RATIO (TAU) = $',100,1.,4.5)
     CALL REALNO(TAU,2,4.5,4.5)
     CALL MESSAG('NUMBER OF INTERVALS USED = $',100,1.,4.0)
     CALL INTNO(N,'ABUT','ABUT')
   ENDIF
   IF (NTYPE.EQ.2) THEN
     CALL MESSAG('SYMMETRIC AIRFOIL SHAPE$','100,' + 1.,5.0)
     CALL MESSAG('THICKNESS RATIO (TAU) = $',100,1.,4.5)
     CALL REALNO(TAU,2,4.3,4.5)
     CALL MESSAG('MAXIMUM THICKNESS AT X = $',100,1.,4.)
     CALL REALNO(XMAXY,2,4.4,4.)
     CALL MESSAG('NUMBER OF INTERVALS USED = $',100,1.,3.5)
     CALL INTNO(N,'ABUT','ABUT')
   ENDIF
   IF (NTYPE.EQ.3) THEN
SUBROUTINE INTHV(NXTOL,NFTOL,NTYPE,MAXIT,IOUT,U0,U1)
C
COMMON /MAIN/T,M,N,XS,XF
DIMENSION T(100),M(100)
C
SUBROUTINE TO FIND THE ROOTS OF f(x) = 0 USING THE
INTERVAL HALVING METHOD
C
IN THE PARAMETER LIST THE USER MUST PROVIDE:
C
NXTOL = EXPONENT FOR X TOLERANCE VALUE
NFTOL = EXPONENT FOR FUNCTION TOLERANCE VALUE
NTYPE = SHAPE TYPE; ELLIPTICAL OR AIRFOIL
MAXIT = MAXIMUM NUMBER OF ITERATIONS
IOUT = 0 TO SUPPRESS ALL OUTPUT (TO DEVICE IW)
1 TO OUTPUT FINAL RESULTS ONLY
2 TO OUTPUT DETAILS FOR EACH ITERATION
C
THE SUBROUTINE CALCULATES:
XS, XF = TWO INITIAL GUESSES, GIVEN N
C
THE SUBROUTINE RETURNS:
U0, U1 = CURRENT VELOCITY VALUES WHEN TERMINATION OCCURRED
IEXIT = 1, 2, 3, 4 OR 7 (SEE FORMAT STATEMENTS 1 - 4 & 7)
C
IW = 5
XTOL = 10.**(-NXTOL)
FTOL = 10.**(-NFTOL)
C
CALCULATE INITIAL GUESS FOR XS AND XF, GIVEN N
XS = 1. / FLOAT(N + 1)
XSPREV = 10.**(-6)
XF = 1. - XS
XFPREV = 1. - XSPREV
C
SET X VALUES FOR LEADING AND TRAILING EDGES FOR SUBROUTINE PRESS
XLE = 0.0
XTE = 1.0
C
ITERATE TO DETERMINE THE PROPER LOCATION FOR XF
C
FIRST CHECK TO SEE THAT f(XF) & f(XFPREV) DIFFER IN SIGN
SO THAT THE METHOD WILL CONVERGE.
EVALUATE PREVIOUS X VALUE
CALL FINDM (T,M,N,XS,XFPREV)
CALL PRESS (XTE,U1,CP)
YFPREV = U1
IF (IOUT .GT. 1) WRITE (IW,5) XFPREV, YPREV, XF, YF
IF (YFPREV*YF .GT. 0.0) THEN
  I = -2
  PRINT 201
  RETURN
END IF

EVALUATE INITIAL GUESS FOR X VALUE
CALL FINDM (T,M,N,XS,XF)
CALL PRESS (XTE,U1,CP)
YF = U1
IF (IOUT .GT. 1) WRITE (IW,5) XFPREV, YPREV, XF, YF
IF (YFPREV*YF .GT. 0.0) THEN
  I = -2
  PRINT 201
  RETURN
END IF

COMPUTE SEQUENCE OF POINTS CONVERGING TO THE ROOT
IEXIT = 1
DO 10 K=1, MAXIT
  XR = (XFPREV + XF)/2.0
C FOR THE ELLIPTIC CASE XS AND XF WILL BE EQUIDISTANT FROM THE EDGES.
  IF (NTYPE .LT. 2) THEN
    XS = ABS (1. - XR)
  END IF
  CALL FINDM (T,M,N,XS,XR)
  CALL PRESS (XTE,U1,CP)
  Y = U1
C CHECK ON STOPPING CRITERIA

  DELTAXF = XFPREV-XR
  XERR = ABS(XFPREV-XR)/2.0
  IF (IOUT .GT. 1) WRITE (IW,6) K,XR,Y,DELTAXF
  IF (Y .EQ. 0.0) IEXIT = 2
  IF (ABS(Y) .LE. FTOL) IEXIT = 3
  IF (XERR .LE. XTOL) IEXIT = 7
  IF (IEXIT .GT. 1) GO TO 20
  IF (Y*YFPREV .GT. 0.0) THEN
    XFPREV = XR
    YFPREV = Y
  ELSE
    XF = XR
    YF = Y
  END IF
10 CONTINUE
C THE MAXIMUM ITERATIONS HAS BEEN EXCEEDED, WITHOUT FINDING A ROOT.
  IEXIT = 4
20 IF (IOUT .EQ. 0) GO TO 30
  IF (IEXIT .EQ. 1 ) WRITE (IW,1) XR
  IF (IEXIT .EQ. 2 ) WRITE (IW,2) XR
  IF (IEXIT .EQ. 3 ) WRITE (IW,3) XR, NUMSIG
  IF (IEXIT .EQ. 4 ) WRITE (IW,4) MAXIT
30 CONTINUE
C FOR THE ELLIPTIC CASE XS AND XF ARE DETERMINED, SO GO BACK.
C
  IF (NTYPE .LT. 2) THEN
    CALL FINDM (T,M,N,XS,XF)
CALL PRESS (XLE,U0,CP)
GO TO 90
END IF

C NOW DO THE SAME FOR XS
PRINT *,'VALUE OBTAINED FOR XF',XF
PRINT *,'-- WORKING ON XS.'

C EVALUATE PREVIOUS X VALUE
CALL FINDM (T,M,N,XSPREV, XF)
CALL PRESS (XLE,U0,CP)
YSPREV = U0

C EVALUATE INITIAL GUESS FOR X VALUE
CALL FINDM (T,M,N,XS, XF)
CALL PRESS (XLE,U0,CP)
YS = U0
IF (IOUT.GT.1) WRITE (IW,5) XSPREV, YSPREV, XS, YS
IF (YSPREV*YS.GT.0.0) THEN
I = -2
PRINT 201
RETURN
END IF

C

C COMPUTE SEQUENCE OF POINTS CONVERGING TO THE ROOT
IEIXIT = 1
DO 40 K=1, MAXIT
XR = (XSPREV + XS)/2.0
CALL FINDM (T,M,N,XR, XF)
CALL PRESS (XLE,U0,CP)
Y = U0

C CHECK ON STOPPING CRITERIA
C
DELTA XS = XSPREV-XR
XERR = ABS(XSPREV-XR)/2.0
IF (IOUT.GT.1) WRITE (IW,6) K,XR,Y,DELTA XS
IF (Y.EQ.0.0) IEXIT = 2
IF (ABS(Y).LE.FTOL) IEXIT = 3
IF (XERR.LE.XTOL) IEXIT = 7
IF (IEIXIT.GT.1) GO TO 50
IF (Y*YSPREV.GT.0.0) THEN
XSPREV = XR
YSPREV = Y
ELSE
XS = XR
YS = Y
END IF
40 CONTINUE

C THE MAXIMUM ITERATIONS HAS BEEN EXCEEDED, WITHOUT FINDING A ROOT.
IEIXIT = 4
50 IF (IOUT.EQ.0) RETURN
IF (IEIXIT.EQ.1) WRITE (IW,1) XR
IF (IEIXIT.EQ.2) WRITE (IW,2) XR
IF (IEIXIT.EQ.3) WRITE (IW,3) XR, NUMSIG
IF (IEIXIT.EQ.4) WRITE (IW,4) MAXIT
IF (IEIXIT.EQ.7) WRITE (IW,7) XR, XTOL
90 RETURN

C******************************************************************************

C
THIS SHOULD RETURN WITH U0 NEAR ZERO AND A GOOD VALUE OF XS.
1 FORMAT('OSLOPE = 0 WHEN X =',G12.7,'. ITERATION DISCONTINUED. ')
2 FORMAT('OCOMPUTED F( ',G12.7,' ) IS 0. ITERATION DISCONTINUED. ')
3 FORMAT('OROOT: ',G12.7,'. APPEARS TO BE ACCURATE TO ',I1,' S. ')
4 FORMAT('ODESIREO ACCURACY IS NOT EVIDENT IN ',I3,' ITERATIONS. ')
5 FORMAT('OHALVING METHOD: Xc-1!, XC0! ARE INITIAL GUESSES. ',/,
   '0 k',4X,'X = Xk',7X,'Y = F(X)',7X,'X-XPREV',/,
   '-1 ',G12.7,E12.5,'/ ',0 ' ',G12.7,E12.5)
6 FORMAT(I3,3X,G12.7,E12.5,E15.5)
7 FORMAT('OX LOCATION: ',G12.7,' IS WITHIN X TOLERANCE OF ',E12.5)
201 FORMAT('OFUNCTION HAS THE SAME SIGN AT BOTH INITIAL POSITIONS. '
   '/ ','O THE BUILT-IN ITERATION SCHEME WILL NOT WORK, THEREFORE'
   '/ ','OYOU MUST SELECT THE ENDPOINTS MANUALLY. ')
END

SUBROUTINE PRESS(X,U,CP)

FIND PRESSURE COEFFICIENT CP AT (X,Y(X))

COMMON /MAIN/T,M,N,XS, XF
DIMENSION T(100),M(100)
REAL M
YB = Y(X)
U = 1.0
V = 0.0
VF1 = 1./((T(1) - X)**2 + YB*YB)
UF1 = (T(1) - X)*VF1
DO 100 J = 1,N
VF2 = 1./((T(J+1) - X)**2 + YB*YB)
UF2 = (T(J+1) - X)*VF2
U = U + M(J)*(UF2 - UF1)
V = V - M(J)*YB*(VF2 - VF1)
VF1 = VF2
100 UF1 = UF2
CP = 1.0 - U*U - V*V
RETURN
END

FUNCTION Y(X)

COMMON /FCN/ AX,TAU,NTYPE
COMMON /DATA/COORX(101),COORY(101)
COMMON /PROB/DATPO
DIMENSION FDP(101)
REAL FIRST,LATER,UP,DOWN,ARC
INTEGER N,DATPO

ORDINATE OF BODY CONTOUR

IF (NTYPE .EQ. 1) THEN
  PROVIDE BODY ORDINATES FOR AN ELLIPSE OF THICKNESS RATIO TAU
  (CHORD HAS BEEN NONDIMENSIONALIZED, C=1.0)
  TO REDUCE THE NUMBER OF VARIABLES PASSED IN THE FUNCTION
STATEMENT, THE DUMMY VARIABLE AX PASSES TAU FOR THE ELLIPSOID CASE AND THE COEFFICIENT AX(TAU,XMAXY) FOR THE SYMMETRICAL AIRFOIL-LIKE CASE.

\[ Y = \tau \times \sqrt{x \times (1 - x)} \]

ELSEIF (NTYPE .EQ. 2) THEN

PROVIDE BODY ORDINATES FOR A SYMMETRIC AIRFOIL-LIKE SHAPE
(CHOORD HAS BEEN NONDIMENSIONALIZED, C=1.0)

\[ Y = ax \times \sqrt{x} \times (1 - x) \]

ELSE

PROVIDE BODY ORDINATES FOR AN ARBITRARY BODY. TO DETERMINE THESE POINTS A CUBIC SPLINE INTERPOLATION SUBROUTINE WAS ADDED TO PROGRAM DUBLET.

N = DATPO + 2
XX = X
CALL SPLINE(N,COORX,COORY,FDP)
CALL SPEVAL(N,COORX,COORY,FDP,XX,F)
Y = F
ENDIF
RETURN
END

SUBROUTINE SCALER(VALMAX,VALMIN)

ARRAY = THE ARRAY WHICH IS BEING SORTED INTO ASCENDING ORDER
NUMBER = THE NUMBER OF ELEMENTS IN THE ARRAY TO BE SORTED
VALMAX = LARGEST VALUE IN THE ARRAY
VALMIN = SMALLEST VALUE IN THE ARRAY
REAL VALMAX,VALMIN
INTEGER NUMBER
LOGICAL SORTED
COMMON /JACK/YY.NUM
DIMENSION ARRAY(100),YY(100)
SORTED = .FALSE.
NUMBER = NUM
DO 20 I = 1,NUMBER
 ARRAY(I) = YY(I)
20 CONTINUE
30 IF (.NOT.SORTED) THEN
 SORTED = .TRUE.
 DO 40 I = 1,NUMBER - 1
 IF(ARRAY(I).GT.ARRAY(I+1)) THEN
 VALUE = ARRAY(I)
 ARRAY(I) = ARRAY(I+1)
 ARRAY(I+1) = VALUE
 SORTED = .FALSE.
 ENDIF
40 CONTINUE
GO TO 30
ENDIF
VALMAX = ARRAY(NUMBER)
VALMIN = ARRAY(1)

* THE FOLLOWING FILE IS CREATED TO CHECK THE VALIDITY OF THIS ROUTINE
OPEN (UNIT=16,FILE='ARRAY.DAT',STATUS='NEW')
DO 45 I = 1,NUMBER
    WRITE (16,55)ARRAY(I)
45  CONTINUE
WRITE (16,65)VALMAX,VALMIN,NUMBER
55  FORMAT(1X,E11.4)
65  FORMAT(/,1X,'VALMAX = ',F6.4,/, 'VALMIN = ',E11.4,/, 'NUMBER = ',I3)
CLOSE (UNIT=16)
RETURN
END
SUBROUTINE SCALER2(VALMAX, VALMIN)

C
C ARRAY = THE ARRAY WHICH IS BEING SORTED INTO ASCENDING ORDER
C NUMBER= THE NUMBER OF ELEMENTS IN THE ARRAY TO BE SORTED
C VALMAX= LARGEST VALUE IN THE ARRAY
C VALMIN= SMALLEST VALUE IN THE ARRAY
REAL VALMAX, VALMIN
INTEGER NUMBER
LOGICAL SORTED
COMMON /ABLE/CP, NUM
DIMENSION ARRAY(100), CP(100)
SORTED = .FALSE.
NUMBER = NUM
DO 20 I = 1, NUMBER
    ARRAY(I) = CP(I)
20  CONTINUE
30  IF (.NOT. SORTED) THEN
    SORTED = .TRUE.
    DO 40 I = 1, NUMBER - 1
        IF (ARRAY(I).GT. ARRAY(I+1)) THEN
            VALUE = ARRAY(I)
            ARRAY(I) = ARRAY(I+1)
            ARRAY(I+1) = VALUE
            SORTED = .FALSE.
        ENDIF
    40  CONTINUE
    GO TO 30
ENDIF
40  CONTINUE
END

* THE FOLLOWING FILE IS CREATED TO CHECK THE VALIDITY OF THIS ROUTINE
OPEN (UNIT=17,FILE='ARRAY2.DAT',STATUS='NEW')
DO 45 I = 1, NUMBER
    WRITE (17,55)ARRAY(I)
45  CONTINUE
WRITE (17,65)VALMAX, VALMIN, NUMBER
55  FORMAT(1X,E11.4)
65  FORMAT(/,1X,'VALMAX = ',F6.4,/, 'VALMIN = ',E11.4,/, 'NUMBER = ',I3)
CLOSE (UNIT=17)
RETURN
END
SUBROUTINE SPEVAL(N,COORX,COORY,FDP,XX,F)

C
C THIS SUBROUTINE EVALUATES THE CUBIC SPLINE GIVEN
C THE DERIVATIVES COMPUTED BY SUBROUTINE SPLINE.
C THE INPUT PARAMETERS N,X,Y,FDP HAVE THE SAME
C MEANING AS IN SPLINE.
C XX = VALUE OF INDEPENDENT VARIABLE FOR WHICH
C AN INTERPOLATED VALUE IS REQUESTED
C F = THE INTERPOLATED RESULT
DIMENSION COORX(101),COORY(101),FDP(101)

C THE FIRST STEP IS TO FIND THE PROPER INTERVAL
NM1 = N - 1
DO 1 =1,NM1
IF (XX.LE.COORX(I+1)) GO TO 10
1 CONTINUE

C NOW EVALUATE THE CUBIC

10 DXM = XX - COORX(I)
DXP = COORX(I+1) - XX
DEL = COORX(I+1) - COORX(I)
F = FDP(I)*DXP*(DXP*DXP/DEL - DEL)/6.
1 +FDP(I+1)*DXM*(DXM*DXM/DEL - DEL)/6.
2 +COORY(I)*DXP/DEL + COORY(I+1)*DXM/DEL
RETURN
END

SUBROUTINE SPLINE(N,COORX,COORY,FDP)

C
C THIS SUBROUTINE COMPUTES THE SECOND DERIVATIVES NEEDED
C IN CUBIC SPLINE INTERPOLATION. THE INPUT DATA ARE:
C N = NUMBER OF DATA POINTS
C COORX = ARRAY CONTAINING THE VALUES OF THE INDEPENDENT VARIABLE
C (ASSUMED TO BE ASCENDING ORDER)
C COORY = ARRAY CONTAINING THE VALUES OF THE FUNCTION AT THE
C DATA POINTS GIVEN IN THE COORX ARRAY
DIMENSION COORX(101),COORY(101),A(101),B(101)
DIMENSION C(101),R(101),FDP(101)
ALAMDA = 1
NM2 = N - 2
NM1 = N - 1
C(1) = COORX(2) - COORX(1)
DO 1 =2,NM1
C(I) = COORX(I+1) - COORX(I)
A(I) = C(I-1)
B(I) = 2.*(A(I) + C(I))
R(I) = 6.*((COORY(I+1)-COORY(I))/C(I)-(COORY(I)
+ -COORY(I-1))/C(I-1))
1 CONTINUE
B(2) = B(2) + ALAMDA * C(1)
B(NM1) = B(NM1) + ALAMDA * C(NM1)
DO 2 =3,NM1
T = A(I)/B(I-1)
B(I) = B(I) - T * C(I-1)
R(I) = R(I) - T * R(I-1)
2 CONTINUE
FDP(NM1) = R(NM1)/B(NM1)
DO 3 =2,NM2

117
NMI = N - I
FDP(NMI) = (R(NMI) - C(NMI) * FDP(NMI+1))/B(NMI)

3 CONTINUE
FDP(1) = ALAMDA * FDP(2)
FDP(N) = ALAMDA * FDP(NM1)
C DESIRED DERIVATIVES HAVE NOW BEEN DETERMINED
C RETURN TO MAIN PROGRAM
RETURN
END

FUNCTION YREF(XNUM)
COMMON /LEROY/NUMERAL
COMMON /BRAVO/NUMPTS
COMMON /CHARLIE/NO
COMMON /FLAGGER/FIGURE
DIMENSION FDP(101),XX(101),YY(101)
DIMENSION XPOINT(101),YPOINT(101),XPOIN(101),YPOIN(101)
NO = NUMPTS
C READ IN THE CURRENT SHAPE OF THE AIRFOIL
C IF(FIGURE.EQ.2)NUMERAL=NUMERAL-2
OPEN(UNIT=15,FILE='BODYSHAPE.DAT',STATUS='OLD')
XX(1) = 0.0
YY(1) = 0.0
DO 30 I = 2,NUMERAL+1
   READ (15,*) XX(I),YY(I)
30 CONTINUE
XX(NUMERAL+2) = 1.
YY(NUMERAL+2) = 0.
CLOSE(UNIT=15)
C PROVIDE BODY ORDINATES FOR AN ARBITRARY. TO DETERMINE
C THESE POINTS A CUBIC SPLINE INTERPOLATION SUBROUTINE WAS ADDED
C TO PROGRAM NEW_PANEL.
C THE AIRFOIL SHAPE IS BEING SPLIT INTO UPPER AND LOWER SURFACES AND
C THEN FORMATTED TO BE USED WITH THE SPLINE/SPEVAL ROUTINES.
C NOB = INT(NUMPTS/2)+1
DO I=1,INT(NUMPTS/2)+1
   DUMMY=XX(I)
   DUM =YY(I)
   XPOINT(I)=DUMMY
   YPOINT(I)=DUM
END DO
DO I=INT(NUMPTS/2)+2,NUMPTS
   DUMM=XX(I)
   DU =YY(I)
   XPOIN(I)=DUMM
   YPOIN(I)=DU
END DO
XPOIN(NUMPTS+1)=1.
YPOIN(NUMPTS+1)=0.
CALL SORTNUM(XPOINT,YPOINT,NOB)
CALL SORTNUM(XPOIN,YPOIN,NOB-1)

C UPPER SURFACE Y COORDINATE DETERMINATION
C
IF (XNUM.GT.0.) THEN
  N = INT(NUMPTS/2)+1
  XPT = XNUM
  CALL SPLINE(N,XPOINT,YPOINT,FDP)
  CALL SPEVAL(N,XPOINT,YPOINT,FDP,XPT,F)
  YREF = F
ENDIF

C LOWER SURFACE Y COORDINATE DETERMINATION
C
IF (XNUM.LT.0.) THEN
  N = INT(NUMPTS/2)
  XPT = XNUM
  CALL SPLINE(N,XPOIN,YPOIN,FDP)
  CALL SPEVAL(N,XPOIN,YPOIN,FDP,XPT,F)
  YREF = F
ENDIF
RETURN
END
APPENDIX G. PROGRAM NEW_PANEL COMPUTER CODE

PROGRAM NEW_PANEL

*** MODIFIED FOR USE ON THE MICROVAX/2000 BY J.A. CAMPBELL (JUL 88)
UPDATES MADE BY C.M. MACALLISTER JAN-NOV 89 (CMM)

PROGRAM NEW_PANEL

SMITH-HESS (DOUGLAS) PANEL METHOD
FOR SINGLE-ELEMENT LIFTING AIRFOIL
IN TWO-DIMENSIONAL INCOMPRESSIBLE FLOW

ORIGINAL IBM MAINFRAME PROGRAM WAS ADAPTED FROM JACK MORAN'S BOOK
'AN INTRODUCTION TO THEORETICAL AND COMPUTATIONAL AERODYNAMICS',
WILEY AND SONS, NEW YORK 1984. THE LISTING IS FOUND ON PAGE 118.

PROGRAM FLEXIBILITY AND USER INTERFACE WAS REVISED FOR
PROFESSOR J.V. HEALEY BY JOHN CAMPBELL. APRIL 1988.
ADDITIONAL PROGRAM UPDATES TO INCLUDE PRINTING ROUTINES,
PROCESSING CORRECTIONS, GRAPHICAL ANALYSIS, AND VISCous
INTERACTION ADAPTATION WERE MADE BY CRAIG MACALLISTER
IN JAN-NOV 1989. (CMM)

THE VISCous INTERACTION ADAPTATION WAS REALIZED USING A
FORTRAN PROGRAM DEVELOPED BY DR. T. CEBECI AND DR. H.
B. KELLER. THE ORIGINAL VERSION OF THIS PROGRAM IS
CURRENTLY AVAILABLE FOR USE ON THE IBM MAINFRAME AT THE
NAVAL POSTGRADUATE SCHOOL, ACCOUNT 4632P. IN ORDER TO
USE DR. CEBECI'S PROGRAM IT IS NECESSARY TO INPUT THE
POTENTIAL FLOW SOLUTION OVER A SECTION SHAPE. IN PARTICULAR, THE CP DISTRIBUTION OR THE VELOCITY DISTRIBUTION OVER
AS SURFACE IS REQUIRED. SUCH INFORMATION IS OBTAINED QUITE
READILY THROUGH THE EXECUTION OF THE NEW_PANEL PROGRAM. IN
FACT, THE CP DISTRIBUTION CREATED BY NEW_PANEL IS INTER-
ACTIVELY ADAPTED, SORTED AND INPUTTED TO THE CEBECI PROGRAM
UPON THE USER'S REQUEST.

THIS PROGRAM PROVIDES THE BODY COORDINATES AND THE SURFACE
PRESSURE DISTRIBUTION ABOUT A SINGLE ELEMENT LIFTING AIRFOIL
IN TWO-DIMENSIONAL FLOW.

ESTIMATED VALUES FOR LIFT COEFFICIENT AND THE MOMENT COEFFICIENT
ABOUT THE LEADING EDGE AND QUARTER CHORD ARE DETERMINED FROM THE
PRESSURE COEFFICIENTS OF EACH PANEL.

YOU MAY PROVIDE ACTUAL AIRFOIL SURFACE COORDINATE DATA VALUES OR
HAVE THE COMPUTER GENERATE AN APPROXIMATION FOR THE COORDINATES
OF A NACA XXXX OR 230XX AIRFOIL SECTION.

IF YOU DESIRE TO ENTER THE SURFACE COORDINATE VALUES, SEVERAL
OPTIONS ARE AVAILABLE. YOU MAY ENTER THEM 1) FROM A DATA FILE,
2) FROM THE KEYBOARD OR 3) USING DATA STATEMENTS ALREADY ENTERED
AT THE END OF THE MAIN PROGRAM LISTING.

IF INPUTTING YOUR OWN DATA, REMEMBER TO START AT THE TRAILING EDGE
(X/C = 1.0), AND WORK TOWARDS THE LEADING EDGE, ENTERING THE LOWER
SIDE FIRST, FOLLOWED BY THE UPPER SURFACE. DO NOT ENTER THE
TRAILING EDGE TWICE. TRY TO ENTER A SUFFICIENT NUMBER OF POINTS
NEAR THE NOSE FOR GOOD RESOLUTION.

*** NOTE: TO SATISFY THE KUTTA CONDITION, X VALUES FOR POINTS
2 AND N MUST BE THE SAME. THIS ENSURES THAT THE LAST
PANELS, UPPER AND LOWER, ARE OF EQUAL SIZE.

CD IS JUST AN INDICATOR OF NUMERICAL ACCURACY OF THIS
PROGRAM. VALUE OF CD SHOULD BE NEAR ZERO.

IF USING DATA STMTS OR AN INPUT FILE, REMEMBER THE NUMBER
OF DATA VALUES AS YOU WILL BE ASKED FOR THIS BY THE PROGRAM.

USE OF THE DATA STATEMENTS REQUIRES THAT PROGRAM BE
MODIFIED IN ADVANCE BY MOVING THEM TO THE CORRECT LOCATION.

******************************************
INTEGER NANS, FLAG, FIGURE
REAL LIFTA, MOMENTA, MENTA
DIMENSION XX(101), YCOORD(101), DLS(101), THT(101)
DIMENSION ANGLE(13), CLA(13), CMA(13), CMB(13), YY(101), XREF(101)
DIMENSION YDAT(101)

*** NOTE: IF YOU CHANGE SIZE OF X AND Y, CHANGE N BELOW ALSO|   ***

CHARACTER*1 PRINT, GRAPH, PLOT1, PLOT2, PLOT3, PLOT4, PLOT5
CHARACTER*1 VISCOS, PRINTER
INTEGER PRINTOPT, GRAPHOPT, THICKOPT, VISOPT
DATA X, Y /101*0., 101*0. /
COMMON /EXTRA/LIFTA, MOMENTA, MENTA
COMMON /DATA/X(101), Y(101)
COMMON /BOD/ NODTOT, COSTHE(100), SINTHE(100), NFLAG
COMMON /PAR/ NACA, TAU, EPSMAX, PTMAX
COMMON /FLAGGER/ FIGURE
COMMON /FINAL/ FLAG, XREF, YCORD
COMMON /COF/ A(101, 111), KUTTA
COMMON /BRAVO/ NUMPTS
COMMON /NUM/ PI, PI2INV
COMMON /CPD/ GP(100)
COMMON /PEN/ CLA, CMA, ANGLE, CMB

*** FOLLOWING DATA IS FOR THE NASA LS(1)-0013 AIRFOIL ***
DATA NUPPER, NLOWER /14, 14/
DATA (X(I), I=1, 28) /1.0, .90, .80, .70, .60, .50, .40, .30, .20, .10,
  1 0.07535, 0.05, 0.0247, 0.01255, 0.0, 0.01301, 0.02505, 0.04993, 0.07498,
  2 0.10, .20, .30, .40, .50, .60, .70, .80, .90/

*** NOTE: VALUE FOR TRAILING EDGE IS SET TO 0.000 VS ACT THICKNESS *
DATA (Y(I), I=1, 28) /0.00000, -.01165, -.02654, -.04196, -.05459,
  1 -.06209, -.06453, -.06316, -.05755, -.04543, -.04070, -.03462,
  1 -.02612, -.01938, 0.0, .01892, .02583, .03465, .04075, .04541,
**PI = 3.1415926585**

*** MAKE SURE THAT N CORRESPONDS TO THE SIZE OF X AND Y DIMENSION ***

N = 100
FLAG = 1

OPEN FILE FOR BODY SURFACE COORDINATE OUTPUT

OPEN (UNIT=11,
  2       FILE= 'PBODY.DAT',
  2       ORGANIZATION= 'SEQUENTIAL',
  2       ACCESS= 'SEQUENTIAL',
  2       RECORDTYPE= 'VARIABLE',
  2       FORM= 'FORMATTED',
  2       STATUS= 'NEW')

OPEN FILE FOR PRESSURE COEFFICIENT OUTPUT

OPEN (UNIT=12,
  2       FILE= 'PPRESS.DAT',
  2       ORGANIZATION= 'SEQUENTIAL',
  2       ACCESS= 'SEQUENTIAL',
  2       RECORDTYPE= 'VARIABLE',
  2       FORM= 'FORMATTED',
  2       STATUS= 'NEW')

OPEN ANOTHER FILE FOR BODY SURFACE PRESSURE DISTRIBUTION OUTPUT

OPEN (UNIT=14,
  2       FILE= 'PRESSER.DAT',
  2       ORGANIZATION= 'SEQUENTIAL',
  2       ACCESS= 'SEQUENTIAL',
  2       RECORDTYPE= 'VARIABLE',
  2       FORM= 'FORMATTED',
  2       STATUS= 'NEW')

OPEN ANOTHER FILE FOR BODY SHAPE OUTPUT

OPEN (UNIT=15,
  2       FILE= 'BODYSHAPE.DAT',
  2       ORGANIZATION= 'SEQUENTIAL',
  2       ACCESS= 'SEQUENTIAL',
  2       RECORDTYPE= 'VARIABLE',
  2       FORM= 'FORMATTED',
  2       STATUS= 'NEW')

OPEN(UNIT=46,FILE='ME4.DAT',STATUS='NEW')

DO I= 1,NUMPTS
  WRITE(46,999)X(I),Y(I)
END DO
CLOSE(UNIT=46)

CALL INDATA(X,Y,N,NLOWER,NUPPER)
IF (FLAG.EQ.2) THEN
  NLOWER = NUMPTS/2
  N = NUMPTS
  NUPPER = NUMPTS/2
  OPEN(UNIT=43,FILE='ME1.DAT',STATUS='NEW')
  DO I= 1,NUMPTS
WRITE(43,999)X(I),Y(I)
END DO
CLOSE(UNIT=43)
ENDIF
CALL SETUP(X,Y,N,NLOWER,NUPPER)

CHECK THE INPUT OF THE AIRFOIL SHAPE DATA (OPTIONAL)
OPEN (66,FILE='MAKE.DAT',STATUS='NEW')
DO I = 1,NUMPTS
  WRITE(66,999)X(I),Y(I)
END DO
999 FORMAT(1X,F8.4,2X,F8.4)
CLOSE (UNIT=66)

IF (FLAG.EQ.2) GO TO 908
100 PRINT 1000
READ (5,*) ALPHA

IF (ALPHA .GT. 90.) GO TO 200

AOA = ALPHA
COSALF = COS(ALPHA*PI/180.)
SINALF = SIN(ALPHA*PI/180.)
CALL COFISH(SINALF,COSALF,X,Y,N,NLOWER,NUPPER)
CALL GAUSS(1)
OPEN(UNIT=45,FILE='ME3.DAT',STATUS='NEW')
DO I= 1,NUMPTS
  WRITE(45,999)X(I),Y(I)
END DO
CLOSE(UNIT=45)
CALL VELDIS(SINALF,COSALF,X,Y,N,NLOWER,NUPPER,ALPHA)
CALL FANDM(SINALF,COSALF,X,Y,N,NLOWER,NUPPER)
200 CONTINUE
CLOSE(UNIT=11)
CLOSE(UNIT=12)
CLOSE(UNIT=14)
CLOSE(UNIT=15)
908 IF (FLAG.EQ.2) THEN
  CALL COFISH(SINALF,COSALF,X,Y,N,NLOWER,NUPPER)
  CALL GAUSS(1)
  OPEN(UNIT=44,FILE='ME2.DAT',STATUS='NEW')
  DO I= 1,NUMPTS
    WRITE(44,999)X(I),Y(I)
  END DO
  CLOSE(UNIT=44)
  CALL VELDIS(SINALF,COSALF,X,Y,N,NLOWER,NUPPER,ALPHA)
  CALL FANDM(SINALF,COSALF,X,Y,N,NLOWER,NUPPER)
  CLOSE(UNIT=11)
  CLOSE(UNIT=12)
  CLOSE(UNIT=14)
  CLOSE(UNIT=15)
ENDIF
CALL LIBRARY ROUTINE TO CLEAR THE SCREEN, THEN PRINT HEADER
CALL CLRSCRN
PRINT *
PROGRAM PANEL RESULTS HAVE BEEN WRITTEN TO FILES:

PBODY.DAT : BODY SURFACE COORDINATES
PPRESS.DAT : SURFACE PRESSURE DISTRIBUTION

"WOULD YOU LIKE TO PRINT THE RESULTS (Y/N)?"

"WHICH OF THE FOLLOWING FILES DO YOU WANT?"

1) PBODY.DAT
2) PPRESS.DAT
3) BOTH FILES

"INPUT OPTION NO. (1, 2, OR 3)"

"WOULD YOU LIKE TO GRAPH THE RESULTS (Y/N)?"

"WHICH OF THE FOLLOWING DATA OUTPUTS DO YOU WANT TO PLOT?"

1) PPRESS.DAT (CP DISTRIBUTION)
2) PBODY.DAT (AIRFOIL SHAPE)
3) CL VS. ANGLE OF ATTACK & CM C/4 VS. ANGLE OF ATTACK
4) NONE

"INPUT OPTION NO. (1, 2, 3 OR 4)"
PRINT *, 'INVALID ENTRY, ENTER INTEGER BETWEEN'
PRINT *, 'ONE(1) AND FOUR(4),'
PRINT *, ''
GO TO 65
ENDIF
IF (GRAPHOPT .EQ. 1) THEN
CALL GRAF1
C GET A HARDCOPY OF THIS GRAPHIC
CALL LIB$SPAWN('RENDER/DEVICE=LA210/DRAFT_QUALITY/PAPER+
+SIZE=A1.PUIS')
PRINT *, '',
PRINT *, 'WOULD YOU LIKE A PRINT OF THIS PLOT? (Y/N)'
PRINT *, '',
READ 1002, PLOT1
IF (PLOT1.EQ.'Y'.OR.PLOT1.EQ.'y')THEN
  CALL LIB$SPAWN('PRINT P1.REN')
ENDIF
GO TO 46
ENDIF
IF (GRAPHOPT .EQ. 2) THEN
CALL GRAF2
CALL LIB$SPAWN('RENDER/DEVICE=LA210/DRAFT_QUALITY/PAPER+
+SIZE=A2.PUIS')
PRINT *, '',
PRINT *, 'WOULD YOU LIKE A PRINT OF THIS PLOT? (Y/N)'
PRINT *, '',
READ 1002, PLOT2
IF (PLOT2.EQ.'Y'.OR.PLOT2.EQ.'y')THEN
  CALL LIB$SPAWN('PRINT P2.REN')
ENDIF
GO TO 46
ENDIF
IF (GRAPHOPT .EQ. 3) THEN
C ******************************************
OPEN (UNIT=11,
  2 FILE= 'PBODY.DAT',
  2 ORGANIZATION= 'SEQUENTIAL',
  2 ACCESS= 'SEQUENTIAL',
  2 RECORDTYPE= 'VARIABLE',
  2 FORM= 'FORMATTED',
  2 STATUS= 'UNKNOWN')
OPEN (UNIT=12,
  2 FILE= 'PPRESS.DAT',
  2 ORGANIZATION= 'SEQUENTIAL',
  2 ACCESS= 'SEQUENTIAL',
  2 RECORDTYPE= 'VARIABLE',
  2 FORM= 'FORMATTED',
  2 STATUS= 'UNKNOWN')
OPEN (UNIT=14,
  2 FILE= 'PRESSER.DAT',
  2 ORGANIZATION= 'SEQUENTIAL',
  2 ACCESS= 'SEQUENTIAL',
  2 RECORDTYPE= 'VARIABLE',
  2 FORM= 'FORMATTED',
  2 STATUS= 'UNKNOWN')
OPEN (UNIT=15,
FILE= 'BODYSHAPE.DAT',
ORGANIZATION= 'SEQUENTIAL',
ACCESS= 'SEQUENTIAL',
RECORDTYPE= 'VARIABLE',
FORM= 'FORMATTED',
STATUS= 'UNKNOWN')

DO 45 I = 1,13
    ALPHA = Angle(I)
    COSELF = COS(ALPHA*PI/180.)
    SINALF = SIN(ALPHA*PI/180.)
    CALL COFISH(SINALF,COSALF,X,Y,N,NLOWER,NUPPER)
    CALL GAUSS(1)
    CALL VELDIS(SINALF,COSALF,X,Y,N,NLOWER,NUPPER,ALPHA)
    CALL FANDM(SINALF,COSALF,X,Y,N,NLOWER,NUPPER)
    CLA(I) = LIFTA
    CMA(I) = MOMENTA
    CHB(I) = MENTA
    CLOSE(UNIT=11)
    CLOSE(UNIT=12)
    CLOSE(UNIT=14)
    CLOSE(UNIT=15)

45 CONTINUE

ENDIF

IF (GRAPHOPT .EQ. 3)THEN
    CALL GRAF3
C GET A HARDCOPY OF THIS GRAPHIC
    CALL LIB$SPAWN('RENDER/DEVICE=LA210/DRAFT_QUALITY/PAPER _ +SIZE=A P3.UIS')
    PRINT *, ' ' PRINT *, 'WOULD YOU LIKE A PRINT OF THIS PLOT? (Y/N)'
    PRINT *, ' ' READ 1002, PLOT3
    IF (PLOT3.EQ. 'Y'.OR. PLOT3.EQ. 'y')THEN
        CALL LIB$SPAWN('PRINT P3.REN')
    ENDIF
GO TO 46
ENDIF

IF (GRAPHOPT .EQ. 4 .AND. FLAG .EQ. 1) GO TO 68
IF (GRAPHOPT .EQ. 4 .AND. FLAG .EQ. 2) GO TO 64
ENDIF

IF (FLAG.EQ. 2)GO TO 64

68 CALL CLRSCRN
    PRINT *, ' ' PRINT *, 'WOULD YOU LIKE TO ANALYZE VISCOUS EFFECTS'
    PRINT *, ' ' FOR THIS AIRFOIL (Y/N) ?'
    PRINT *, ' ' READ 1002, VISCOUS
    IF (VISCOUS.EQ. 'Y'.OR. VISCOUS.EQ. 'y')THEN
        FLAG = 2
        FIGURE = 2
        PRINT *, ' ' VISCOS BOUNDARY LAYER ANALYSIS'
        PRINT *, ' ' *** INPUT DATA OPTION ***'
        PRINT *, ' ' WHAT INPUT SOURCE WOULD YOU LIKE TO USE?'
PRINT *, ' ', 1) DATA FILE "BL2D.DAT" OR '  
PRINT *, '  2) NEW_PANEL CP DISTRIBUTION JUST CREATED'  
PRINT *, '  3) QUIT PROGRAM'  
PRINT *, ' ', ENTER 1, 2, OR 3'  
PRINT *, ' ', ENTER 1, 2, OR 3'  
READ *, VISOPT  
IF (VISOPT.LT.1.OR.VISOPT.GT.3)THEN  
   PRINT *, '  INVALID ENTRY| TRY AGAIN|'  
   PRINT *, '  ENTER 1, 2, OR 3'  
   GO TO 42  
ENDIF  
IF (VISOPT.EQ.3)GO TO 1111  
CALL CIB(VISOPT)  
PRINT *, '  THE BOUNDARY LAYER RESULTS HAVE BEEN'  
PRINT *, '  WRITTEN TO FILE "BL2D.OUT"'  
PRINT *, '  WOULD YOU LIKE TO PRINT THESE RESULTS?'  
PRINT *, '  '  
READ 1002, PRINTER  
IF (PRINTER.EQ.'Y'.OR.PRINTER.EQ.'y')THEN  
   CALL LIB$SPAWN('PRINT BL2D.OUT')  
ENDIF  
IF (VISOPT.EQ.1)GO TO 64  
  
THE FOLLOWING "GO TO" STATEMENT WAS PUT IN TO CIRCUMVENT  
PROCESSING THE PANEL METHOD AGAIN.  
  
GO TO 64  
  
NUMPTS = NUMPTS-2  
OPEN(UNIT=63,FILE='VISC.DAT',STATUS='UNKNOWN')  
READ(63,920)XP,DLSN,THTN  
READ(63,920)XP,DLSN,THTN  
READ(63,920)XP,DLSN,THTN  
DO I=1,INT(NUMPTS/2)  
   READ(63,920)XREF(I),DLS(I),THT(I)  
   XNUM = XREF(I)  
   YYY = YREF(XNUM)  
   YCORD(I)=YYY  
END DO  
DO I=(INT(NUMPTS/2)+1),NUMPTS  
   READ(63,920)XREF(I),DLS(I),THT(I)  
   XNUM = XREF(I)  
   YYY = YREF(XNUM)  
   YCORD(I) = '-YYY'  
END DO  
CLOSE(UNIT=63)  
CHECKING THE DATA COORDINATES TO SEE IF IN PROPER ORDER  
OPEN (UNIT=77,FILE='RET.DAT',STATUS='NEW')  
DO I=1,NUMPTS  
   WRITE (77,991)XREF(I),YCORD(I)  
C991   FORMAT(1X,F8.4,2X,F8.4)  
END DO  
CLOSE (UNIT=77)
PRINT *, ' WHICH TYPE OF BOUNDARY LAYER THICKNESS'  
PRINT *, ' WOULD YOU LIKE TO ANALYZE'  
PRINT *, ' 1) DISPLACEMENT THICKNESS'  
PRINT *, ' 2) MOMENTUM THICKNESS'  
PRINT *, ' 3) NONE--QUIT VISCOUS ANALYSIS'  
PRINT *,  
READ 1006, THICKOPT  
IF (THICKOPT .LT. 1 .OR. THICKOPT .GT. 3) THEN  
PRINT *, ' INVALID ENTRY, ENTER INTEGER BETWEEN'  
PRINT *, ' ONE(1) AND THREE(3).'  
PRINT *,  
GO TO 69  
ENDIF  
IF (THICKOPT.EQ.1) THEN  
DO I=1,NUMPTS  
THICK = DLS(I)  
IF (YCORD(I).LT.0.)YCOR = YCORD(I)-THICK  
IF (YCORD(I).GE.0.)YCOR = YCORD(I)+THICK  
YCORD(I) = YCOR  
END DO  
ENDIF  
IF (THICKOPT.EQ.2) THEN  
DO I=1,NUMPTS  
THICK = THT(I)  
IF (YCORD(I).LT.0.)YCOR = YCORD(I)-ABS(THICK)  
IF (YCORD(I).GE.0.)YCOR = YCORD(I)+ABS(THICK)  
YCORD(I) = YCOR  
END DO  
ENDIF  
IF (THICKOPT.EQ.3) GO TO 64  
GO TO 60  
ENDIF  
64 CALL CLRSCRN  
OPTION TO MAKE ANOTHER RUN  
PRINT *  
PRINT *, ' DO YOU WISH TO:'  
PRINT *, ' 1) MAKE ANOTHER RUN OR'  
PRINT *, ' 2) END THIS SESSION'  
PRINT *, ' ENTER 1 OR 2.'  
CALL QUERY (NANS)  
IF (NANS .EQ. 1) GO TO 59  
STOP  
1111 FORMAT(1X,F8.4,2E11.4)  
930 FORMAT(1X,F8.4,F11.6)  
1000 FORMAT(///,,' INPUT ALPHA IN DEGREES ')  
1002 FORMAT(A1)  
1006 FORMAT(I1)  
END  
C ******************************************************  
C DATA VALUES FOR VARIOUS AIRFOILS. TO USE, REMOVE COMMENTS  
C AND PLACE AFTER COMMON CARDS IN MAIN PROGRAM.  
C ******************************************************
*** FOLLOWING DATA IS FOR THE NACA 0006 AIRFOIL ***
DATA NUPPER, NLOWER /14,14/
DATA (X(I),I=1,28)/1.0, 90, 80, 70, 60, 50, 40, 30, 20, 10,
1 0.075, 0.05, 0.025, 0.0125, 0.0, 0.0125, 0.025, 0.05, 0.075,
2 0.10, 20, 30, 40, 50, 60, 70, 80, 90/
*** NOTE: VALUE FOR TRAILING EDGE IS SET TO 0.000 VS ACT THICKNESS *
DATA (Y(I),I=1,28)/-0.0063, -0.0724, -0.1312, -0.1832, -0.2282,
1 -0.2647, -0.2902, -0.3001, -0.2869, -0.2341, 0.0, 0.2341, 0.2869,
2 0.3001, 0.2902, 0.2647, 0.2282, 0.1832, 0.1312, 0.0724/

*** FOLLOWING DATA IS FOR THE NACA 0012 AIRFOIL ***
DATA NUPPER, NLOWER /14,14/
DATA (X(I),I=1,28)/1.0, 90, 80, 70, 60, 50, 40, 30, 20, 10,
1 0.075, 0.05, 0.025, 0.0125, 0.0, 0.0125, 0.025, 0.05, 0.075,
2 0.10, 20, 30, 40, 50, 60, 70, 80, 90/
*** NOTE: VALUE FOR TRAILING EDGE IS SET TO 0.000 VS ACT THICKNESS *
DATA (Y(I),I=1,28)/0.00000, -0.01448, -0.02623, -0.03664, -0.04563,
1 -0.05294, -0.05803, -0.06002, -0.05737, -0.04683, -0.04200, -0.03555,
1 -0.02615, -0.01894, 0.0, 0.01894, 0.02615, 0.03555, 0.04200, 0.04683,
2 0.05737, 0.06002, 0.05803, 0.05294, 0.06002, 0.06453, 0.06307, 0.05604/

*** FOLLOWING DATA IS FOR THE NASA LS(1)-0013 AIRFOIL ***
DATA NUPPER, NLOWER /14,14/
DATA (X(I),I=1,28)/1.0, 90, 80, 70, 60, 50, 40, 30, 20, 10,
1 0.07535, 0.05, 0.0247, 0.01255, 0.0, 0.01301, 0.02505, 0.04993, 0.07498,
2 0.10, 20, 30, 40, 50, 60, 70, 80, 90/
*** NOTE: VALUE FOR TRAILING EDGE IS SET TO 0.000 VS ACT THICKNESS *
DATA (Y(I),I=1,28)/0.00000, -0.01165, -0.02654, -0.04196, -0.05459,
1 -0.06209, -0.06453, -0.06316, -0.05755, -0.04543, -0.04070, -0.03462,
1 -0.02612, -0.01938, 0.0, 0.01892, 0.02583, 0.03465, 0.04075, 0.04541,
2 0.05750, 0.06307, 0.06432, 0.06203, 0.05446, 0.04183, 0.02638, 0.01172/

* USER INSTRUCTIONS FOR MANUAL DATA ENTRY: *

(1) UPON CUE ENTER THE TOTAL NUMBER OF AIRFOIL DATA POINTS. DO NOT COUNT THE LEADING OR TRAILING EDGE TWICE.
NOTE: ARRAYS ARE DIMENSIONED TO 100, THIS IS, THEREBY THE LIMITING NUMBER OF DATA POINTS THAT CAN BE ENTERED WITHOUT HAVING TO REDIMENSION THE PROGRAMS ARRAYS.

(2) ENTER X COORDINATES AS MANY TO A LINE AS DESIRED. THE PROGRAM WILL ALLOW FOR CORRECTION IF ANY ERRORS ARE MADE. A TABLE OF X COORDINATES IS DISPLAYED FOR THE USER TO CHECK HIS INPUT.

(3) ENTER Y COORDINATES AS MANY TO A LINE AS DESIRED. THE PROGRAM WILL ALLOW FOR CORRECTION IF ANY ERRORS ARE MADE. A TABLE OF Y COORDINATES IS DISPLAYED FOR THE USER TO CHECK HIS INPUT.

(4) PROGRAM ALLOWS FOR AS MANY RUNS AS THE USER DESIRES SIMPLY FOLLOW CUING SEQUENCE.
SUBROUTINE BL

C

COMMON /BLC2/ NX,NXT,NP,NPT,NTR,IT,ISF
COMMON /BLC3/ X(100),UE(100),P1(100),P2(100),GMTR(100)
COMMON /BLC7/ ETA(101),DETA(101),A(101)
COMMON /BLC8/ F(101,2),U(101,2),V(101,2),B(101,2)
COMMON /BLC6/ DELF(101),DELU(101),DELV(101)

C

NX =0
ITMAX = 10
IGROWT = 2
EPSL = .0001
EPST = .01
NPT = 101

C ETA-GRID NETWORK
C

ETAE = 8.
VGP = 1.1
DETA(1) = .01
NP = ALOG((ETAE/DETA(1))*(VGP-1.)*1.)/ALOG(VGP) +1.001
ETA(1) = 0.
DO 10 J = 2,NPT
   ETA(J) = ETA(J-1) + DETA(J-1)
   DETA(J) = VGP*DETA(J-1)
   A(J) = .5*DETA(J-1)
10  CONTINUE

C INITIAL LAMINAR VELOCITY PROFILE
C

DO 20 J = 1,NP
   ETAB = ETA(J)/ETA(NP)
   ETAB2 = ETAB**2
   F(J,2) = .25*ETA(NP)*ETAB2*3. - .5*ETAB2)
   U(J,2) = .5*ETAB*(3. - ETAB2)
   V(J,2) = 1.5 *(1. - ETAB2)/ETA(NP)
   B(J,2) = 1.
20  CONTINUE

C

1 NX = NX+1
   IGROW = 0
   IT = 0
C

5 IT = IT+1
C
   WRITE(*,*)IT
   IF (IT.GT. ITMAX) GO TO 101
   IF(NX.GE.NTR) CALL EDDY
      CALL COEF
      CALL SOLV3
C
   CHECK FOR CONVERGENCE
   IF (NX .LT. NTR) THEN
IF (ABS(DELV(1)).GT. EPSL) GO TO 5
ELSE
    IF (ABS(DELV(1)/V(1,2)).GT. EPST) GO TO 5
ENDIF

PROFILES FOR GROWTH

DO 30 J = NP+1,NPT
    F(J,2) = F(J-1,2) + DELTA(J-1)*U(J-1,2)
    U(J,2) = U(J-1,2)
    V(J,2) = 0.
    B(J,2) = B(J-1,2)
30 CONTINUE

CHECK FOR GROWTH

IF (ABS(V(NP,2)).GT. .0005 .OR. ABS(1. - U(NP-2,2)/U(NP,2)) + .GT. .005) THEN
    NP = NP + 2
    IGROW = IGROW + 1
    IF (NP .LE. NPT .AND. IGROW .LE. IGROWT) THEN
        IT = 0
        GO TO 5
    ENDIF
ENDIF

CALL OUTPUT
IF (NX .LT. NXT) GO TO 1
RETURN
END

SUBROUTINE BODY(Z,SIGN,XI,YI)

RETURN COORDINATES OF POINT ON THE BODY SURFACE

Z = NODE-SPACING PARAMETER
X,Y = CARTESIAN COORDINATES
SIGN = +1. FOR UPPER SURFACE
-1. FOR LOWER SURFACE

COMMON /PAR/ NACA,TAU,EPSMAX,PTMAX
IF (SIGN .LT. 0.0)  Z = 1. - Z
CALL NACA45(Z,THICK,CAMBER,BETA)
XI = Z - SIGN*THICK*SIN(BETA)
YI = CAMBER + SIGN*THICK*COS(BETA)
RETURN
END

SUBROUTINE NACA45(Z,THICK,CAMBER,BETA)

COMMON /PAR/ NACA,TAU,EPSMAX,PTMAX

EVALUATE THICKNESS AND CAMBER
FOR NACA 4- OR 5-DIGIT AIRFOIL
THICK = 0.0
IF (Z.LT. 1.E-10) GO TO 100
THICK = 5.*TAU*(0.2969*SQRT(Z) - Z*(0.126 + Z*(0.3537
+ - Z*(0.2843 - Z*(0.1015))))
100 IF (EPSMAX.EQ. 0.0) GO TO 130
IF (NACA.GT. 9999) GO TO 140
IF (Z.GT. PTMAX) GO TO 110
CAMBER = EPSMAX/PTMAX/PTMAX*(2.*PTMAX - Z)*Z
DCAMDX = 2.*EPSMAX/PTMAX/PTMAX*(PTMAX - Z)
GO TO 120
DCAMDX = 2.*EPSMAX/(1. -PTMAX)**2*(PTMAX - Z)
120 BETA = ATAN(DCAMDX)
RETURN
130 CAMBER = 0.0
BETA = 0.0
RETURN
140 IF (Z.GT. PTMAX) GO TO 150
W = Z/PTMAX
CAMBER = EPSMAX*W*((W - 3.)*W + 3. - PTMAX)
DCAMDX = EPSMAX*3.*W*(1. - W)/PTMAX
GO TO 120
150 CAMBER = EPSMAX*(1. - Z)
DCAMDX = - EPSMAX
GO TO 120
END

PROGRAM CEBECI

THIS PROGRAM REPRESENTS AN ADAPTATION OF A VISCOUS
BOUNDARY LAYER PROGRAM CURRENTLY ON THE IBM 3033. GIVEN A
COEFFICIENT OF PRESSURE DISTRIBUTION ABOUT AN AIRFOIL/WING,
THIS PROGRAM WILL DETERMINE THE RELATIVE BOUNDARY LAYER
THICKNESS ALONG THE CHORD AND THE COEFFICIENT OF FRICTION
AT THE SAME POSITION.

SUBROUTINE CIB(OPTION)

COMMON /BLCO/ RL,NBL(2),XCTRI(2)
COMMON /BLC1/ ITR,XCTR,XC(100),YC(100)
COMMON /BLC2/ NX,NXT,NP,NPT,NTR,IT,ISF
COMMON /BLC3/ X(100),UE(100),P1(100),P2(100),GMTR(100)
COMMON /BLCS/ DLS(100),WV(100),CF(100),THT(100)
COMMON /VISCIOUS/XCORD,YCOR,CPDAT
COMMON /BRAVO/NUMPTS
COMMON /FRIC/DISKIN,DFORM
DIMENSION NXTSF(2),XI(200),YI(200),VEI(200),VEL(200)
DIMENSION XCORD(100),YCOR(100),CPDAT(100)
INTEGER OPTION

OPEN (UNIT=9,FILE='BL2D.DAT',STATUS='UNKNOWN')
OPEN (UNIT=8,FILE='BL2D.OUT',STATUS='UNKNOWN')
OPEN (UNIT=62,FILE='VIS.DAT',STATUS='UNKNOWN')
OPEN (UNIT=63,FILE='VISC.DAT',STATUS='UNKNOWN')

UNIT=62 IS A CHECKING FILE
IF (OPTION.EQ.2)THEN
DO I = 1,NUMPTS
  VEI(I)=SQRT(1-CPDAT(I))
END DO

WRITE(62,777)(VEI(I),I=1,NUMPTS)
C 777 FORMAT(1X,F10.5)
NUMBER = 1
DUMMY = VEI(1)
DO I = 2,NUMPTS
  IF (DUMMY.GT.VEI(I))THEN
    DUMMY=VEI(I)
    NUMBER=I
  ENDDIF
END DO
IS = NUMBER
CALL CLRSCRN
PRINT *, 'ENTER THE TRANSITION POINT ON THE ' 
PRINT *, 'UPPER SURFACE(E.G. 0.3)'
PRINT *, 'ENTER THE TRANSITION POINT ON THE ' 
PRINT *, 'LOWER SURFACE(E.G. 0.3)'
READ *, XCTRI(1)
READ *, XCTRI(2)
WRITE(6,*) 'READING THE DATA...'
C CHECK INPUT DATA BY WRITING IT INTO UNIT = 62
18 FORMAT(1X,3F10.5)
19 FORMAT(1X,F10.1,2F10.5)
WRITE(62,19)RL,XCTRI(1),XCTRI(2)
WRITE(62,10)NUMPTS,IS
NI = NUMPTS
NOT = 1
NAY = 1
66 IF (NOT.LE.NUMPTS)THEN
  WRITE(62,18)XCORD(NOT),YCOR(NOT),VEI(NOT)
  XI(NAY) = XCORD(NOT)
  YI(NAY) = YCOR(NOT)
  VEI(NAY)= SQRT(1-CPDAT(NOT))
  VEL(NAY)= SQRT(1-CPDAT(NOT))
  NAY = NAY + 1
  NOT = NOT + 1
  GO TO 66
ENDIF
C CHECK DATA
C OPEN (UNIT=87,FILE='CHECK1.DAT',STATUS='NEW')
C DO I=1,NUMPTS
  WRITE (87,991)XCORD(I),YCOR(I),VEI(I),XI(I),YI(I)
C 991 FORMAT(1X,5F8.4)
C END DO
C CLOSE (UNIT=87)
C
DO I=1,INT(NUMPTS/2)
   CAN1 = XCORD(I)
   CAN2 = YCOR(I)
   CAN3 = VEL(I)
   XI(I+1)=CAN1
   YI(I+1)=CAN2
   VEI(I+1)=CAN3
END DO
DO I=INT(NUMPTS/2)+1,NUMPTS
   CAN1 = XCORD(I)
   CAN2 = YCOR(I)
   CAN3 = VEL(I)
   XI(I+2)=CAN1
   YI(I+2)=CAN2
   VEI(I+2)=CAN3
END DO
XI(1)=1.
YI(1)=0.
VEI(1)=.97*VEI(1)
XI(NUMPTS/2+2)=0.
YI(NUMPTS/2+2)=0.
VEI(NUMPTS/2+2)=(VEI(NUMPTS/2+1)+VEI(NUMPTS/2+3))/2.
XI(NUMPTS+3)=1.000
YI(NUMPTS+3)=0.000
VEI(NUMPTS+3)=.98*VEI(NUMPTS+2)
NUMPTS = NUMPTS+3
NI = NUMPTS
C CHECK DATA
C PRINT *,NUMPTS
C PRINT *,NI
C OPEN (UNIT=88,FILE='CHECK2.DAT',STATUS='NEW')
C DO I=1,NUMPTS
C   WRITE (88,996)XI(I),YI(I),VEI(I)
C 996 FORMAT(1X,3F8.4)
C END DO
ELSE
   WRITE(6,*) 'READING THE DATA...'
   READ (9,15) RL,XCTRI(1),XCTRI(2)
   READ (9,10) NI,IS
   READ (9,15) (XI(I),YI(I),VEI(I),I=1,NI)
ENDIF
DO I=1,NI
   WRITE(62,18)XI(I),YI(I),VEI(I)
END DO
65 WRITE (6,*) 'INPUT OF DATA COMPLETE.'
WRITE (8,90) RL,XCTRI(1),XCTRI(2)
NXTSF(1) = NI - IS + 1
NXTSF(2) = IS
C DATA FOR EACH SURFACE
C DO 200 ISF = 1,2
   NXT = NXTSF(ISF)
   GO TO (201,202), ISF
C
C UPPER SURFACE

201 II = IS -1
DO 211 I = 1,NXT
   II = II+1
   XC(I) = XI(II)
   YC(I) = YI(II)
   UE(I) = VEI(II)
211 CONTINUE
   GO TO 300

C LOWER SURFACE

202 II = IS + 1
DO 212 I = 1,NXT
   II = II - 1
   XC(I) = XI(II)
   YC(I) = YI(II)
   UE(I) = VEI(II)
212 CONTINUE

300 X(1) = 0.
DO 301 I = 2,NXT
   X(I) = X(I-1)+SQRT((XC(I)-XC(I-1))**2+(YC(I)-YC(I-1))**2)
301 CONTINUE

C TRANSITION LOCATION

DO 320 I = 1,NXT
   GMTR(I) = 0.
   IF (XC(I) .GE. XCTRI(ISF)) GO TO 321
320 CONTINUE

321 NTR = I
   PGAMTR = 1200.
   RXNTR = X(NTR-1) * UE(NTR-1) * RL
   CLOSE (UNIT=68)
   GGFT = RL*RL/RXNTR**1.34*UE(NTR-1)*UE(NTR-1)*UE(NTR-1)
   UEINTG = 0.
   U1 = .5/UE(NTR-1)/PGAMTR
   DO 322 I = NTR,NXT
      U2 = .5/UE(I)/PGAMTR
      UEINTG = UEINTG + (U1 + U2)*(X(I)-X(I-1))
      U1 = U2
      GG = GGFT*UEINTG*(X(I)-X(NTR-1))
      IF (GG .GT. 10.) GO TO 323
   GMTR(I) = 1. - EXP(-GG)
322 CONTINUE

323 DO 324 II = I,NXT
324 GMTR(II) = 1.

C PRESSURE GRADIENT PARAMETERS

DX = X(2) - X(1)
DUE = UE(2) - UE(1)
ANG2 = ATAN2(DUE,DX)
DL2 = DX
DO 331 I = 2,NXT-1
ANG1 = ANG2
DL1 = DL2
DX = X(I+1) - X(I)
DUE = UE(I+1) - UE(I)
ANG2 = ATAN2(DUE,DX)
DL2 = DX
ANG = (DL2*ANG1+DL1*ANG2)/(DL1+DL2)
P2(I) = TAN(ANG)

331 CONTINUE
P2(NXT) = 2.*DUE/DL2 - P2(NXT-1)
DO 330 I = 2,NXT
   P2(I) = X(I) * P2(I) /UE(I)
P1(I) = .5 * (1. + P2(I))
330 CONTINUE
P2(1) = 1.
P1(1) = .5 * (1. + P2(1))

C BOUNDARY LAYER CALCULATION

C WRITE(6,*) 'BOUNDARY LAYER COMPUTATIONS IN PROGRESS,..
CALL BL
C INSERTED ABS FOR CHECKING PURPOSES ONLY
WRITE (8,910) ISF,(I,XC(I),X(I),VW(I),CF(I),DLS(I),THT(I),
   + I=1,NXT)
WRITE(63,920)(XC(I),ABS(DLS(I)),ABS(THT(I)),I=1,NXT)

200 CONTINUE
CALL DRAG
WRITE(8,103)DSKIN,DFORM
C
CLOSE(UNIT = 8)
CLOSE(UNIT = 9)
CLOSE(UNIT = 62)
CLOSE(UNIT = 63)

10 FORMAT(2I5)
15 FORMAT(3F10.0)
90 FORMAT(///5X,'RL='',E12.5,5X,'XCTRI(1)='',F8.3,5X,
   + 'XCTR(2)='',F8.3)
103 FORMAT(///,6X,'TOTAL SKIN DRAG = ','F10.6,
   + ',6X,'TOTAL FORM DRAG = ','F10.6)
910 FORMAT(///2X,'*** SUMMARY OF BOUNDARY LAYER SOLUTIONS OF ISF = '
   + ',I2,///2X,'NX',4X,'XC',8X,'S',8X,'VW',8X,'CF',8X,'DLS',8X,'THT'
   + ',/I5,2F8.4,4E11.4))
920 FORMAT(1X,F8.4,2E11.4)
RETURN
END

SUBROUTINE CLRSCRN
C
C LIBRARY ROUTINE TO CLEAR THE SCREEN.
C
ISTAT = LIB$ERASE_PAGE (1,1)
RETURN
END

SUBROUTINE QUERY(NANS)
ROUTINE TO TRAP ERRORS CAUSED BY IMPROPER RESPONSES TO QUESTIONS.
THE COMPUTER GENERATES AND ERROR WHEN A CHARACTER IS SUPPLIED TO
A QUESTION EXPECTING AN INTEGER OR REAL VALUE.

NQTEST=0
1 CONTINUE
IF (NQTEST .GT. 0) THEN
   PRINT *, ' CHARACTER VALUES ARE NOT VALID. '
   PRINT *, ' PLEASE ENTER AN INTEGER VALUE. '
END IF
NQTEST = NQTEST + 1
READ (5,*,ERR=1)NANS
RETURN
END

SUBROUTINE COEF

COMMON /BLC2/ NX,NXT,NP,NPT,NTR,IT,ISF
COMMON /BLC3/ X(100),UE(100),P1(100),P2(100),GMTR(100)
COMMON /BLC7/ ETA(101),DETA(101),A(101)
COMMON /BLC8/ F(101,2),U(101,2),V(101,2),B(101,2)
COMMON /BLC9/ S1(101),S2(101),S3(101),S4(101),S5(101),
   +S6(101),S7(101),S8(101),R1(101),R2(101),R3(101),R4(101)

P1H = .5*P1(NX)
IF (NX .EQ. 1) THEN
   CEL = 0.
   CELH = 0.
   DO 5 J=1,NP
      F(J,1) = 0.
      U(J,1) = 0.
      V(J,1) = 0.
      B(J,1) = 0.
   CONTINUE
ELSE
   CEL = .5 * (X(NX)+X(NX-1))/(X(NX)-X(NX-1))
   CELH = .5 * CEL
ENDIF

DO 100 J = 2,NP

CURRENT STATION

FB = .5*(F(J,2) + F(J-1,2))
UB = .5*(U(J,2) + U(J-1,2))
FVB = .5*(F(J,2)*V(J,2)+F(J-1,2)*V(J-1,2))
VB = .5*(V(J,2) + V(J-1,2))
USB = .5*(U(J,2)**2 + U(J-1,2)**2)
DERBV = (B(J,2)*V(J,2) - B(J-1,2)*V(J-1,2))/DETA(J-1)

PREVIOUS STATION

CFB = .5*(F(J,1) + F(J-1,1))
CUB = .5*(U(J,1) + U(J-1,1))
CVB = .5*(V(J,1) + V(J-1,1))
CUSB = .5*(U(J,1)**2 + U(J-1,1)**2)
CFVB = .5*(F(J,1)*V(J,1)+F(J-1,1)*V(J-1,1))
CDERBV = (B(J,1)*V(J,1) - B(J-1,1)*V(J-1,1))/DETA(J-1)

S- COEFFICIENTS

S1(J) = CELH*(F(J,2) - CFVB) + P1H*F(J,2) +
        +B(J,2)/DETA(J-1)
S2(J) = CELH*(F(J-1,2) - CFVB) + P1H*F(J-1,2) -
        +B(J-1,2)/DETA(J-1)
S3(J) = CELH*(V(J,2) + CVB) + P1H*V(J,2) +
S4(J) = CELH*(V(J-1,2) + CVB) + P1H*V(J-1,2)
S5(J) = -(CEL+P2(NX))*U(J,2)
S6(J) = -(CEL+P2(NX))*U(J-1,2)

R- COEFFICIENTS

IF (NX .EQ. 1) THEN
    CRB = -P2(NX)
    R2(J) = CRB - (DERBV + P1(NX) * FVB - P2(NX)*USB)
ELSE
    CLB = CDERBV + P1(NX-1)*CFVB - P2(NX-1)*CUSB +
    + P2(NX-1)
    CRB = -CLB - CEL*CUSB - P2(NX)
    R2(J) = CRB - (DERBV + P1(NX)*FVB - (CEL+P2(NX))*
    + USB + CEL*(FVB + CVB*FB - VB*CFB - CFVB))
ENDIF
R1(J) = F(J-1,2) - F(J,2) + DETA(J-1)*UB
R3(J-1) = U(J-1,2) - U(J,2) + DETA(J-1)*VB
100 CONTINUE

BOUNDARY CONDITIONS

R1(1) = 0.
R2(1) = 0.
R3(NP) = 0.

RETURN
END

SUBROUTINE COFISH(SINALF,COSALF,X,Y,N,NLOWER,NUPPER)

SET COEFFICIENTS OF LINEAR SYSTEM

REAL X(N),Y(N)
COMMON /BOD/ NODTOT,COSTHE(100),SINTHE(100),NFLAG
COMMON /COF/ A(101,111),KUTTA
COMMON /NUM/ PI,PI2INV
KUTTA = NODTOT + 1

INITIALIZE COEFFICIENTS

DO 90 J = 1,KUTTA
90 A(KUTTA,J) = 0.0

SET VN = 0 AT MID-POINT OF I-TH PANEL
DO 120 I = 1,NODTOT
XMID = .5*(X(I) + X(I+1))
YMID = .5*(Y(I) + Y(I+1))
A(I,KUTTA) = 0.0

C

-- FIND CONTRIBUTION OF J-TH PANEL

DO 110 J = 1,NODTOT
FLOG = 0.0
FTAN = PI
IF (J .EQ. I) GO TO 100
DXJ = XMID - X(J)
DXJP = XMID - X(J+1)
DYJ = YMID - Y(J)
DYJP = YMID - Y(J+1)
FLOG = .5*ALOG((DXJP*DXJP+DYJP*DYJP)/(DXJ*DXJ+DYJ*DYJ))
FTAN = ATAN2(DYJP*DXJ-DXJP*DYJ,DXJP*DXJ+DYJ*DYJ)
CTMTJ = COSTHE(I)*COSTHE(J) + SINTHE(I)*SINTHE(J)
STMTJ = SINTHE(I)*COSTHE(J) - COSTHE(I)*SINTHE(J)
A(I,J) = PI2INV*(FTAN*CTMTJ + FLOG*STMTJ)
B = PI2INV*(FLOG*CTMTJ - FTAN*STMTJ)
A(I,KUTTA) = A(I,KUTTA) + B
IF ((I .GT. 1) .AND. (I .LT. NODTOT)) GO TO 110

C

-- IF I-TH PANEL TOUCHES TRAILING EDGE,
ADD CONTRIBUTION TO KUTTA CONDITION

A(KUTTA,J) = A(KUTTA,J) - B
A(KUTTA,KUTTA) = A(KUTTA,KUTTA) + A(I,J)
110 CONTINUE

C

FILL IN KNOWN SIDES

A(I,KUTTA+1) = SINTHE(I)*COSALF - COSTHE(I)*SINALF
120 CONTINUE

A(KUTTA,KUTTA+1) = -(COSTHE(1) + COSTHE(NODTOT))*COSALF
+ -(SINTHE(1) + SINTHE(NODTOT))*SINALF
RETURN
END

SUBROUTINE DRAG

THE PURPOSE OF THIS SUBROUTINE IS TO CALCULATE THE TOTAL
FORM DRAG AND THE TOTAL SKIN DRAG GIVEN THE BOUNDARY LAYER
CHARACTERISTICS FROM SUBROUTINE CIB.

COMMON /BLC1/ ITR,XCTR,XC(100),YC(100)
COMMON /BLC2/ NX,NXT,NP,NPT,NTR,IT,ISF
COMMON /BLC3/ X(100),UE(100),P1(100),P2(100),GMTR(100)
COMMON /BLCS/ DLS(100),VW(100),CF(100),THT(100)
COMMON /FRIC/ DSKIN,DFORM
DSKIN = 0.
T1 = 0.
DO I=2,NXT
  T2 = CF(I)*UE(I)**2
  DSKIN = DSKIN + .5*(T1+T2)*(XC(I)-XC(I-1))
  T1 = T2
139
END
HTE = DLS(NXT)/THT(NXT)
DFORM = 2. * THT(NXT)*UE(NXT)**( .5*(5. + HTE))
RETURN
END

SUBROUTINE EDDY

C
COMMON /BLCO/ RL,NBL(2),XCTRI(2)
COMMON /BLC2/ NX,NXT,NP,NPT,NTR,IT,ISF
COMMON /BLC3/ X(100),UE(100),P1(100),P2(100),GMTR(100)
COMMON /BLC7/ ETA(101),ETAE(101),A(101)
COMMON /BLC8/ F(101,2),U(101,2),V(101,2),B(101,2)
DIMENSION EDVI(101)

C
RL2 = SQRT(RL*UE(NX)*X(NX))
RL4 = SQRT(RL2)
RL216 = .16 * RL2

C
ALFA = .0168
EDVO = ALFA*RL2*GMTR(NX)*(U(NP,2)*ETA(NP)-F(NP,2))
EDVI(1) = 0.
YBAJ = RL4*SQRT(ABS(V(1,2)))/26.
DO 70 J = 2,NP
JJ = J
YBA = YBAJ*ETA(J)
EL = 1.
IF(YBA .LT. 10.) EL = 1. - EXP(-YBA)
EDVI(J) = RL216*GMTR(NX)*(EL*ETA(J))**2 * ABS(V(J,2))
IF (EDVI(J) .GT. EDVO) GO TO 90
IF (EDVI(J) .LE. EDVI(J-1)) EDVI(J) = EDVI(J-1)
B(J,2) = 1. + EDVI(J)
70 CONTINUE
90 DO 100 JJ=J,NPT
100 B(JJ,2) = 1. + EDVO
B(1,2) = 1.

C
RETURN
END

SUBROUTINE FANDM(SINALF,COSALF,X,Y,N,NLOWER,NUPPER)

C
REAL X(N),Y(N)
REAL LIFTA,MOMENTA,MENTA
COMMON /EXTRA/LIFTA,MOMENTA,MENTA
COMMON /BOD/ NODTOT,COSTHE(100),SINTHE(100),NFLAG
COMMON /CPD/ CP(100)
CFX = 0.0
CFY = 0.0
CM = 0.0
CMC4 = 0.0
DO 100 I = 1,NODTOT
XMID = .5*(X(I) + X(I+1))
YMID = .5*(Y(I) + Y(I+1))
100 CONTINUE
DX = X(I+1) - X(I)
DY = Y(I+1) - Y(I)
CFX = CFX + CP(I)*DY
CFY = CFY - CP(I)*DX
CM = CM + CP(I)*(DX*Xmid + DY*Ymid)
CMC4 = CMC4 + CP(I)*(DX*(Xmid-0.25) + DY*Ymid)

CONTINUE
CD = CFX*COSALF + CFY*SINALF
CL = CFY*COSALF - CFX*SINALF
LIFTA = CL
MENTA = CM
MOMENTA = CMC4
PRINT 1000, CD,CL,CM,CMC4
WRITE (12,1000) CD,CL,CM,CMC4

FORMAT('///',10X,'CD=',F8.5,',',CL=',F8.5,///,10X,'CM=',F8.5,',CMC4=',F8.5)
RETURN
END

SUBROUTINE GAUSS (NRHS)

SOLUTION OF LINEAR ALGEBRAIC SYSTEM BY
GAUSS ELIMINATION WITH PARTIAL PIVOTING

A = COEFFICIENT MATRIX
NEQNS = NUMBER OF EQUATIONS
NRHS = NUMBER OF RIGHT HAND SIDES

RIGHT-HAND SIDES AND SOLUTIONS STORED IN
COLUMNS NEQNS+1 THRU NEQNS+NRHS OF A

COMMON DX,DY,AR,PI
COMMON /COF/ A(350,351),NEQNS
NP = NEQNS + 1
NTOT = NEQNS + NRHS

DO 150 I = 2,NEQNS

-- SEARCH FOR LARGEST ENTRY IN (I-1)TH COLUMN
ON OR BELOW MAIN DIAGONAL

IM = I - 1
IMAX = IM
AMAX = ABS(A(IM,IM))
DO 110 J = I,NEQNS
  IF (AMAX .GE. ABS(A(J,IM))) GO TO 110
  IMAX = J
  AMAX = ABS(A(J,IM))
110 CONTINUE

-- SWITCH (I-1)TH AND IMAXTH EQUATIONS

IF (IMAX .NE. IM) GO TO 140
DO 130 J = IM,NTOT
TEMP = A(IM,J)
A(IM,J) = A(IMAX,J)
A(IMAX,J) = TEMP

CONTINUE

ELIMINATE (I-1)TH UNKNOWN FROM
ITH THRU (NEQNS)TH EQUATIONS

DO 150 J = I,NEQNS
R = A(J,IM)/A(IM,IM)
DO 150 K = I,NTOT
A(J,K) = A(J,K) - R*A(IM,K)

BACK SUBSTITUTION

DO 220 K = NP,NTOT
A(NEQNS,K) = A(NEQNS,K)/A(NEQNS,NEQNS)
DO 210 L = 2,NEQNS
I = NEQNS + 1 - L
IP = I + 1
DO 200 J = IP,NEQNS
A(I,K) = A(I,K) - A(I,J)*A(J,K)
210 A(I,K) = A(I,K)/A(I,I)
220 CONTINUE
RETURN
END

SUBROUTINE GRAF1

DEFINE IPACK ARRAY FOR LEGEND
INTEGER*4 IPACK(35)
INTEGER NUM
REAL XX(100),CP(100),MAX,MIN,AIR
CHARACTER*40 LI
COMMON /ABLE/NUM
COMMON /BOD/ NODTOT,COSTHE(100),SINTHE(100),NFLAG
COMMON /BRAVO/NUMPTS
COMMON /PAR/ NACA,TAU,EPSMAX,PTMAX
COMMON /CHARLIE/ AIR
COMMON /CRAIG/CP

READ ELEMENTS OF UNIT 14 INTO ARRAYS TO PLOT
OPEN(UNIT=14,FILE='PRESSER.DAT',STATUS='OLD')
DO 25 I = 1,NUM
READ(14,*),XX(I),CP(I)
25 CONTINUE
CLOSE(UNIT=14)
CALL FORM1(MAX,MIN)

DEFINE AND ASSIGN LEGEND CHARACTER STRINGS
L1 = 'LOWER AND UPPER AIRFOIL POINTS$

INITIALIZE THE GRAPHICS SYSTEM
CALL INIT

LABEL X AND Y AXES USING SELF COUNTING STRINGS
CALL XNAME('X$',100)
CALL YNAME('CP$',100)

DEFINE PLOT AREA TO BE 6 INCHES BY 8 INCHES
CALL AREA2D(6.0,8.0)
C DEFINE HEADING LABEL
   CALL HEADING('CP DISTRIBUTION',100,2,1)
C PLOT ADDITIONAL TICK MARKS
   CALL XTIKS(1)
   CALL YTIKS(1)
C PACK LEGEND LABELS INTO ARRAY IPACK
   CALL LINES(L1,IPACK,1)
C SET UP AXIS
   CALL GRAF(0.0,0.2,1.0,(MIN-.1),((MAX-MIN)/5.),(MAX+.1))
C FRAME THE SUBPLOT AREA
   CALL FRAME
C PLOT PRESSURE DISTRIBUTION DATA WITH A THICK LINE AND MARKER 15
   CALL MARKER(15)
   CALL THKCRV(.04)
   CALL CURVE(XX,CP,NUM,1)
C PRINT MESSAGES
   IF (NFLAG.EQ.1) GO TO 58
      CALL MESSAGE('NACA AIRFOIL',100,2,7.0)
      CALL INTNO('NACA','ABUT','ABUT')
   58 CALL MESSAGE('NUMBER OF PANELS USED = ',100,2,6.5)
      CALL INTNO(NUMPTS,'ABUT','ABUT')
      CALL MESSAGE('ANGLE OF ATTACK = ',100,2,6.0)
      CALL REALNO(AIR,2,4.75,6.0)
C CHANGE LEGEND NAME TO "CP DISTRIBUTION"
   CALL MYLEGN('CP DISTRIBUTION',100)
C PLOT LEGEND
   CALL LEGEND(IPACK,1,.75,1)
C END PLOT
   CALL ENDP(0)
C CREATE GRAPHICS METAFILE P1.UIS
   CALL METAFL(1)
C TERMINATE PLOT AT END OF PLOTTING SESSION
   CALL DONEPL
   RETURN
END

SUBROUTINE GRAF2
C C DEFINE IPACK ARRAY FOR LEGEND
   INTEGER*4 IPACK(35)
   INTEGER NUMERAL, FIGURE
   REAL XX(101), YY(101), MAX, MIN
   CHARACTER*40 L1
   COMMON /LEROY/NUMERAL
   COMMON /BOD/ NODTOT, COSTHE(100), SINTHE(100), NFLAG
   COMMON /BRAVO/NUMPTS
   COMMON /PAR/ NACA, TAU, EPSMAX, PTMAX
   COMMON /CHARLIE/AIR
   COMMON /PIN/XX, YY
   COMMON /FLAGGER/FIGURE
C READ ELEMENTS OF UNIT 15 INTO ARRAYS TO PLOT
   OPEN(UNIT=15, FILE='BODYSHAPE.DAT', STATUS='OLD')
   IF (FIGURE.EQ.2) GO TO 31
   XX(1) = 0.0
   YY(1) = 0.0
DO 30 I = 2, NUMERAL + 1
   READ (15, *) XX(I), YY(I)
30   CONTINUE
XX(NUMERAL + 2) = 1.0
YY(NUMERAL + 2) = 0.0
NUMERAL = NUMERAL + 2
31   IF (FIGURE.EQ.2) THEN
      READ (15, *) XERR, YERR
      DO I = 1, NUMERAL - 2
         READ (15, *) XX(I), YY(I)
      END DO
   ENDIF
CLOSE(UNIT=15)
C CALL SCALER TO FIND THE MAX AND MIN VALUES OF THE Y ORDIANTE ARRAY
   CALL FORM2(MAX, MIN)
C DEFINE AND ASSIGN LEGEND CHARACTER STRINGS
   L1 = 'AIRFOIL SHAPE$'
C INITIALIZE THE GRAPHICS SYSTEM
   CALL INIT
C LABEL X AND Y AXES USING SELF COUNTING STRINGS
   CALL XNAME('X$', 100)
   CALL YNAME('Y$', 100)
C DEFINE PLOT AREA TO BE 6 INCHES BY 8 INCHES
   CALL AREA2D(6.0, 8.0)
C DEFINE HEADING LABEL
   CALL HEADIN('BODY SHAPE$', -100, 2.1)
C PLOT ADDITIONAL TICK MARKS
   CALL XTIMES(1)
   CALL YTICKS(1)
C PACK LEGEND LABELS INTO ARRAY IPACK
   CALL LINES(L1, IPACK, 1)
C SET UP AXIS
   CALL GRAP(0.0, 0.2, 1.0, -.5, 2.5)
C FRAME THE SUBPLOT AREA
   CALL FRAME
C PLOT PRESSURE DISTRIBUTION DATA WITH A THICK LINE AND MARKER 15
   CALL MARKER(15)
   CALL THKCRV(.04)
   CALL CURVE(XX, YY, NUMERAL, 1)
C PRINT MESSAGES
   IF (NFLAG.EQ.1) GO TO 58
      CALL MESSAGE('NACA AIRFOIL $', 100, 2.7.0)
      CALL INTNO(NACA, 'ABUT', 'ABUT')
   58   CALL MESSAGE('NUMBER OF PANELS USED = $', 100, 2.6.5)
      CALL INTNO(NUMPTS, 'ABUT', 'ABUT')
      CALL MESSAGE('ANGLE OF ATTACK = $', 100, 2.6.0)
      CALL REALNO(AIR, 2.4.75, 6.0)
C CHANGE LEGEND NAME TO "UPPER AND LOWER SURFACES"
   CALL MYLEGN('UPPER SURFACE AND LOWER SURFACES$', 100)
C PLOT LEGEND
   CALL LEGEND(IPACK, 1.75, 1.0)
C END PLOT
   CALL ENDPL(0)
C CREATE GRAPHICS METAFILE P2.UIS
   CALL METAFL(2)
C TERMINATE PLOT AT END OF PLOTTING SESSION
CALL DONEPL
RETURN
END

SUBROUTINE GRAF3

C DEFINE IPACK ARRAY FOR LEGEND
INTEGER*4 IPACK(35)
REAL ANGLE(13),CLA(13),CMA(13),MAX,MIN
CHARACTER*40 L1,L2,L3
COMMON /PEN/ CLA,CMA,ANGLE
COMMON /BOD/ NODTOT,COSTHE(100),SINTHE(100),NFLAG
COMMON /BRAVO/NUMPTS
COMMON /PAR/ NACA,TAU,EPSMAX,PTMAX
DIMENSION Y(3),X(3)
CALL MAXMIN(CLA,13,MAX,MIN)
CALL MAXMIN(CMA,13,VALMAX,VALMIN)

C DEFINE AND ASSIGN LEGEND CHARACTER STRINGS
L1 = 'CL VS. ANGLE OF ATTACK$
L2 = 'CM C/4 VS. ANGLE OF ATTACK$
L3 = 'ZERO LINE-REFERENCE ONLY$
C INITIALIZE THE GRAPHICS SYSTEM
CALL INIT
C LABEL X AND Y AXES USING SELF COUNTING STRINGS
CALL XNAME('ANGLE OF ATTACKS',100)
CALL YNAME('MOMENT(C/4) & LIFT COEFFICIENTS',100)
C DEFINE PLOT AREA TO BE 6 INCHES BY 8 INCHES
CALL AREA2D(6.0,8.0)
C DEFINE HEADING LABEL
CALL HEADIN('CL & CM C/4 VS. ALPHA',-100,2.,1)
C PLOT ADDITIONAL TICK MARKS
CALL XTICKS(1)
CALL YTICKS(1)
C PACK LEGEND LABELS INTO ARRAY IPACK
CALL LINES(L1,IPACK,1)
CALL LINES(L2,IPACK,2)
CALL LINES(L3,IPACK,3)
C SETUP AXIS
CALL GRAF(-8.4,16.,(MIN-.5),((MAX-MIN)/5.),((MAX+.5))
C FRAME THE SUBPLOT AREA
CALL FRAME
C PLOT DATA WITH A THICK LINE AND MARKER 15
CALL MARKER(15)
CALL THKCRV(.04)
CALL CURVE(ANGLE,CLA,13,1)
CALL MARKER(16)
CALL RESET('THKCRV')
CALL DASH
CALL CURVE(ANGLE,CMA,13,1)
C ZERO LINE - REFERENCE ONLY
X(1) = -8.
X(2) = 2.
X(3) = 15.9
Y(1) = 0.
Y(2) = 0.
Y(3) = 0.
CALL MARKER(17)
CALL RESET('THKCRV')
CALL DOT
CALL CURVE(X,Y,3,1)

C PRINT MESSAGES
IF (NFLAG.EQ.1)GO TO 58
   CALL MESSAG('NACA AIRFOIL $',100,1.5,8.7)
   CALL INTNO(NACA,'ABUT','ABUT')
58   CALL MESSAG('NUMBER OF PANELS USED = $',100,1.5,8.3)
   CALL INTNO(NUMPTS,'ABUT','ABUT')
C CHANGE LEGEND NAME TO " "
   CALL MYLEGN(' $',100)
C PLOT LEGEND
   CALL LEGEND(IPACK,1,.75,6.5)
C END PLOT
   CALL ENDPL(0)
C CREATE GRAPHICS METAFILE P3.UIS
   CALL METAFL(3)
C TERMINATE PLOT AT END OF PLOTTING SESSION
   CALL DONEPL
   RETURN
END

SUBROUTINE GRAF4

C DEFINE IPACK ARRAY FOR LEGEND
   INTEGER*4 IPACK(35)
   REAL ANGLE(13),CLA(13),CMA(13),MAX,MIN
   CHARACTER*40 L1
   COMMON /PEN/ CLA,CMA,ANGLE
   COMMON /BRAVO/NUMPTS
   COMMON /PAR/ NACA,TAU,EPSMAX,PTMAX
   CALL MAXMIN(CMA,13,MAX,MIN)
C DEFINE AND ASSIGN LEGEND CHARACTER STRINGS
   L1 = 'CM C/4 VS. ANGLE OF ATTACK$
C INITIALIZE THE GRAPHICS SYSTEM
   CALL INIT
C LABEL X AND Y AXES USING SELF COUNTING STRINGS
   CALL XNAME('ANGLE OF ATTACK$,100)
   CALL YNAME('MOMENT COEFFICIENT (C/4)$',100)
C DEFINE PLOT AREA TO BE 6 INCHES BY 8 INCHES
   CALL AREA2D(6.0,8.0)
C DEFINE HEADING LABEL
   CALL HEADIN('CM C/4 VS. ALPHA$',-100,2.,1)
C PLOT ADDITIONAL TICK MARKS
   CALL XTI KCS(1)
   CALL YTI KCS(1)
C PACK LEGEND LABELS INTO ARRAY IPACK
   CALL LINES(L1,IPACK,1)
C SET UP AXIS
   CALL GRAF(-10.,4.,18.,(MIN-.01),((MAX-MIN)/2.),(MAX+.01))
C FRAME THE SUBPLOT AREA
   CALL FRAME
C PLOT PRESSURE DISTRIBUTION DATA WITH A THICK LINE AND MARKER 15
   CALL MARKER(15)
   CALL THKCRV(.04)

146
CALL CURVE(ANGLE,CMA,13,1)
C PRINT MESSAGES
IF (NFLAG.EQ.1)GO TO 58
CALL MSG('NACA AIRFOIL $',100,2.,7.0)
CALL INTNO(NACA,'ABUT','ABUT')
58 CALL MSG('NUMBER OF PANELS USED = $',100,2.,6.5)
CALL INTNO(NUMPTS,'ABUT','ABUT')
C CHANGE LEGEND NAME TO ""
CALL MYLEGN(' $',100)
C PLOT LEGEND
CALL LEGEND(IPACK,1,75,5)
C END PLOT
CALL ENDP(0)
C CREATE GRAPHICS METAFILE P4.UIS
CALL METAFL(4)
C TERMINATE PLOT AT END OF PLOTTING SESSION
CALL DONEPL
RETURN
END

SUBROUTINE GRAF5
C C DEFINE IPACK ARRAY FOR LEGEND
INTEGER*4 IPACK(35)
REAL ANGLE(13),CLA(13),CMA(13),CMB(13),MAX,MIN
CHARACTER*40 L1
COMMON /PEN/ CLA,CMA,ANGLE,CMB
COMMON /BRAVO/NUMPTS
COMMON /PAR/ NACA,TAU,EPSMAX,PTMAX
CALL MAXMIN(CMB,13,MAX,MIN)
C DEFINE AND ASSIGN LEGEND CHARACTER STRINGS
L1 = 'CM VS. ANGLE OF ATTACK$'
C INITIALIZE THE GRAPHICS SYSTEM
CALL INIT
C LABEL X AND Y AXES USING SELF COUNTING STRINGS
CALL XNAME('ANGLE OF ATTACK$',100)
CALL YNAME('MOMENT COEFFICIENT$',100)
C DEFINE PLOT AREA TO BE 6 INCHES BY 8 INCHES
CALL AREA2D(6.0,8.0)
C DEFINE HEADING LABEL
CALL HEADIN('CM VS. ALPHAS$','-100,2.,1)
C PLOT ADDITIONAL TICK MARKS
CALL XTICKS(1)
CALL YTICKS(1)
C PACK LEGEND LABELS INTO ARRAY IPACK
CALL LINES(L1,IPACK,1)
C SET UP AXIS
CALL GRAF(-10.,4.,18.),(MIN-.2),.2,(MAX+.2))
C FRAME THE SUBPLOT AREA
CALL FRAME
C PLOT PRESSURE DISTRIBUTION DATA WITH A THICK LINE AND MARKER 15
CALL MARKER(15)
CALL THKCRV(.04)
CALL CURVE(ANGLE,CMB,13,1)
C PRINT MESSAGES
IF (NFLAG.EQ.1)GO TO 58
CALL MESSAG('NACA AIRFOIL $',100,2.,7.0)
CALL INTNO(NACA,'ABUT','ABUT')
58 CALL MESSAG('NUMBER OF PANELS USED = $',100,2.,6.5)
CALL INTNO(NUMPTS,'ABUT','ABUT')
C CHANGE LEGEND NAME TO " "
CALL MYLEGN(' $',100)
C PLOT LEGEND
CALL LEGEND(IPACK,1.,75.,5)
C END PLOT
CALL ENDPLO(0)
C CREATE GRAPHICS METAFILE P5.UIS
CALL METAFL(5)
C TERMINATE PLOT AT END OF PLOTTING SESSION
CALL DONEPL
RETURN
END

SUBROUTINE INDATA(X,Y,N,NLOWER,NUPPER)
C
C SET PARAMETERS OF BODY SHAPE
C FLOW SITUATION, AND NODE DISTRIBUTION
C
C USER MUST INPUT
NLOWER = NUMBER OF NODES ON LOWER SURFACE
NUPPER = NUMBER OF NODES ON UPPER SURFACE
PLUS DATA ON BODY AND SUBROUTINE BODY
C
REAL X(N),Y(N)
INTEGER NUMPTS,I,STATUS,IFLAG
CHARACTER*20 INFILE
INTEGER*2 INFILE_SIZE
INTEGER FLAG,INFIS
LOGICAL EXIST
COMMON /FINAL/FLAG,XREF,YCORD
COMMON /BOD/ NODTOT,COSTHE(100),SINTHE(100),NFLAG
COMMON /BRAVO/NUMPTS
COMMON /CHARLIE/NO
COMMON /PAR/ NACA,TAU,EPSMAX,PTMAX
DIMENSION XREF(101),YCORD(101),Y1(101),X1(101),Y2(101)
+X2(101)
IF (FLAG.EQ.2) THEN
NUPPER=NUMPTS/2
NLOWER=NUMPTS/2
GO TO 909
ENDIF
C CALL LIBRARY ROUTINE TO CLEAR THE SCREEN, THEN PRINT HEADER
5 CALL CLRSCRN
PRINT *
PRINT *
PRINT *, ' PROGRAM PANEL '  
PRINT *
PRINT *, ' SMITH-HESS (DOUGLAS) PANEL METHOD'
PRINT *, ' FOR A SINGLE-ELEMENT LIFTING AIRFOIL'
PRINT *, ' IN TWO-DIMENSIONAL INCOMPRESSIBLE FLOW'
PRINT *
PRINT *, ' DO YOU WISH TO:  '
1) USE AIRFOIL SURFACE COORDINATE DATA VALUES.
2) HAVE COMPUTER GENERATE AN APPROXIMATION
   FOR A NACA XXXX OR 230XX AIRFOIL SECTION.
3) QUIT THE PROGRAM.
READ (5,*) NFLAG
GO TO (10,50,999) NFLAG

C ROUTINE TO INPUT SHAPE FROM DATA FILE, KEYBOARD OR DATA STMTS **
10 CALL CLRSCRN

PRINT *, 'DO YOU WISH TO ENTER THE SURFACE COORDINATE VALUES:
             1) FROM A DATA FILE.'
PRINT *, '2) FROM THE KEYBOARD.'
PRINT *, '3) USING DATA STATEMENTS ALREADY ENTERED'
PRINT *, 'IN THE MAIN PROGRAM. ** NOTE ** THIS REQUIRES'
PRINT *, 'THAT PROGRAM BE MODIFIED IN ADVANCE BY MOVING'
PRINT *, 'DATA STATEMENTS TO THE CORRECT LOCATION.'
PRINT *, 'ENTER 1, 2, OR 3. (FOR PREVIOUS MENU ENTER 4)'

12 READ (5,*) IFLAG
IF (IFLAG .EQ. 4) GO TO 5
IF (IFLAG .LT. 1 .OR. IFLAG .GT. 3) THEN
    PRINT *, 'INVALID ENTRY. ENTER 1, 2, OR 3.'
    GO TO 12
END IF
IF (IFLAG .EQ. 1) GO TO 110
IF (IFLAG .EQ. 3) THEN
    NUMPTS=28
    GO TO 100
ENDIF

C CUE THE USER TO ENTER THE NUMBER OF DATA POINTS (NUMPTS)
15 CALL CLRSCRN

PRINT *, 'ENTER NUMBER OF DATA POINTS'
READ *, NUMPTS

C CHECK THE INPUT
PRINT *, 'NUMBER OF DATA POINTS TO BE ENTERED =',NUMPTS
PRINT *
PRINT *, 'IS THIS VALUE CORRECT? (YES=1, NO=2)
READ *, M1
IF (M1 .GT. 1) GO TO 15
CALL NODES(NUMPTS,NLOWER,NUPPER)

C SEND CONTROL TO DATA FILE OR KEYBOARD ENTRY ROUTINE ****
110 GO TO (20,30,100) IFLAG

C DATA FILE READ ROUTINE

C LIB$GET_INPUT IS A VAX LIBRARY ROUTINE. IT MAY BE REPLACED BY AN
C EQUIVALENT WRITE/READ TO GET THE FILENAME INTO THE PROGRAM.

20 PRINT *, '
PRINT *, 'NOTICE--YOU CAN NOW ENTER ANY FILE NAME,'
PRINT *, 'NOT JUST "INFILE.DAT"
PRINT *, '
STATUS = LIB$GET_INPUT (INFILE, | THE INPUT FILE
    , | PROMPT
    , | FILENAME SIZE

C CHECK TO SEE IF THE FILE EXISTS BEFORE TRYING TO ACCESS IT
IF (INFILE .EQ. '999') GO TO 5
INQUIRE (FILE = INFILE (1:INFILE_SIZE), EXIST = EXIST)
IF (.NOT. EXIST) THEN
  PRINT *
  PRINT *, ' THAT FILE NAME DOES NOT EXIST.'
  PRINT *, ' (ENTER 999 TO RETURN TO MENU).'
  PRINT *
  GO TO 20
END IF

C OPEN FILE FOR SURFACE COORDINATE INPUT
OPEN (UNIT=13,  
  FILE= INFILE,  
  ORGANIZATION= 'SEQUENTIAL',  
  ACCESS= 'SEQUENTIAL',  
  RECORDTYPE= 'VARIABLE',  
  FORM= 'FORMATTED',  
  STATUS= 'OLD')
PRINT *, '
PRINT *, ' HOW MANY DATA POINTS ARE IN YOUR INPUT FILE?
PRINT *, '
READ *, INFIS
NUMPTS = INFIS

C PRINT *, INFILE_SIZE
DO 25  I = 1,INFIS
   READ (13,*) X(I),Y(I)
PRINT 1010, X(I),Y(I)
25 CONTINUE
1010 FORMAT(F10.4,F10.4)
GO TO 100

C ***** ROUTINE TO ENTER DATA FROM THE KEYBOARD *****
30 CALL INPUT(X,Y,NUMPTS)
GO TO 100

C ***** ROUTINE TO CALCULATE SHAPE, GIVEN NACA NUMBER *****
50 CALL CLRSCRN
PRINT *
PRINT *, ' ENTER NUMBER OF SURFACE DATA POINTS DESIRED'
READ *, NUMPTS

***** ECHO CHECK THE INPUT
CALL CLRSCRN
PRINT *
PRINT *, ' NUMBER OF SURFACE DATA POINTS TO BE GENERATED =',NUMPTS
PRINT *
PRINT *, ' IS THIS VALUE CORRECT? (YES=1, NO=2)'  
READ *, M1  
IF (M1 .GT. 1) GO TO 50  
CALL NODES(NUMPTS,NLOWER,NUPPER)
PRINT *
PRINT *, ' INPUT NACA NUMBER, ANY FOUR-DIGIT OR 230XX SERIES'
READ (5,*) NACA
IEPS = NACA/1000
IPTMAX = NACA/100 - 10*IEPS
ITAU = NACA - 1000*IEPS - 100*IPTMAX
EPSMAX = IEPS*0.01
PTMAX = IPTMAX*0.1
TAU = ITAU*0.01
IF (IEPS .LT. 10) RETURN
PTMAX = 0.2025
EPSMAX = 2.6595*PTMAX**3
909 IF (FLAG.EQ.2) THEN
  C PRINT *, NUMPTS
  C CHECK DATA
  C OPEN(UNIT=69,FILE='TIM.DAT',STATUS='NEW')
  C DO I = 1, NUMPTS
     WRITE(69,978)XREF(I), YCORD(I)
  C 978 FORMAT(1X,F8.4,F12.6)
END DO
  X(1)=1.
  Y(1)=0.
  DO I = 2, NUMPTS+1
     DUMMY = XREF(I)
     DUM = YCORD(I)
     X(I) = DUMMY
     Y(I) = DUM
  END DO
  DO I = 1, INT(NUMPTS/2)+1
     UUU=X(I)
     VVV=Y(I)
     X1(I)=UUU
     Y1(I)=VVV
  END DO
  CALL SORTER(X1,Y1,INT(NUMPTS/2)+1)
  DO I = INT(NUMPTS/2)+1, NUMPTS+1
     UUU=X(I)
     VVV=Y(I)
     X2(I)=UUU
     Y2(I)=VVV
  END DO
  CALL SORTER(X2,Y2,NUMPTS/2+1)
  DO I = INT(NUMPTS/2)+1
     DDD=X1(I)
     X(I)=DDD
     EEE=Y1(I)
     Y(I)=EEE
  END DO
  DO I = INT(NUMPTS/2)+2, NUMPTS+1
     DDD=X2(I)
     X(I)=DDD
     EEE=Y2(I)
     Y(I)=EEE
  END DO
END IF
100 RETURN
999 STOP
END

SUBROUTINE INPUT(A,B,N)
C
  INTEGER N, I
  DIMENSION A(N), B(N)
C  CUE THE USER TO INPUT X VALUES
10 PRINT *, 'ENTER X VALUES AS MANY PER LINE AS DESIRED'
   READ *, (A(I), I = 1,N)
C ECHO CHECK THE INPUT
PRINT 20, N
20 FORMAT (/1X,'TABLE OF', I3,' X VALUES: '/1X,21('='))
   PRINT 30, (A(I), I=1,N)
30 FORMAT (1X,3F10.6)
   PRINT *, 'ARE THE VALUES CORRECT? (YES=1, NO=2)'
   READ *, J1
   IF (J1.GT. 1) GO TO 10
C CUE THE USER TO INPUT Y VALUES
35 PRINT *, 'ENTER Y VALUES AS MANY PER LINE AS DESIRED'
   READ *, (B(J), J=1,N)
C ECHO CHECK THE INPUT
PRINT 40, N
40 FORMAT (/1X,'TABLE OF', I3,' Y VALUES: '/1X,21('='))
   PRINT 30, (B(J), J=1,N)
   PRINT *, 'ARE THE VALUES CORRECT? (YES=1, NO=2)'
   READ *, K1
   IF (K1.GT. 1) GO TO 35
RETURN
END

SUBROUTINE MAXMIN(ARRAY,NY,VALMAX,VALMIN)
C
C ARRAY = THE ARRAY WHICH IS BEING SORTED INTO ASCENDING ORDER
C NUMBER= THE NUMBER OF ELEMENTS IN THE ARRAY TO BE SORTED
C VALMAX= LARGEST VALUE IN THE ARRAY
C VALMIN= SMALLEST VALUE IN THE ARRAY
REAL VALMAX,VALMIN
INTEGER NUMBER
LOGICAL SORTED
DIMENSION ARRAY(100)
SORTED = .FALSE.
NUMBER = NY
30 IF (.NOT.SORTED) THEN
   SORTED = .TRUE.
   DO 40 I = 1,NUMBER - 1
     IF(ARRAY(I).GT.ARRAY(I+1))THEN
       VALUE = ARRAY(I)
       ARRAY(I) = ARRAY(I+1)
       ARRAY(I+1) = VALUE
       SORTED = .FALSE.
     ENDIF
   40 CONTINUE
   GO TO 30
ENDIF
VALMAX = ARRAY(NUMBER)
VALMIN = ARRAY(1)
RETURN
END

SUBROUTINE NODES(NUMPTS,NLOWER,NUPPER)
C
C **** CALCULATE NLOWER AND NUPPER FOR LATER USE ***
PRINT *
PRINT *, ' ARE THE NUMBER OF UPPER AND LOWER SURFACE'
PRINT *, ' DATA POINTS(NODES) EQUAL? (YES=1, NO=2)'
READ *, M1
IF (M1 .EQ. 1) THEN
  NLOWER = NUMPTS/2
  NUPPER = NLOWER
ELSE
  CALL CLRSCRN
  PRINT *
  PRINT *, ' TOTAL NUMBER OF SURFACE POINTS =', NUMPTS
  PRINT *
  PRINT *, ' INPUT NUMBER OF LOWER SURFACE POINTS, NLOWER'
  READ (5,*), NLOWER
  PRINT *, ' INPUT NUMBER OF UPPER SURFACE POINTS, NUPPER'
  READ (5,*), NUPPER
  NTEST = NLOWER + NUPPER
  IF (NTEST .NE. NUMPTS) THEN
    PRINT *, ' OKAY, TRY IT AGAIN EINSTEIN. REMEMBER ADDITION?'
    PRINT *, ' NLOWER + NUPPER MUST EQUAL', NUMPTS
    GO TO 20
  END IF
END IF
RETURN
END

SUBROUTINE OUTPUT

C

COMMON /BLCO/ RL,NBL(2), XCTRI(2)
COMMON /BLC2/ NX,NXT,NP,NPT,NTR,IT,ISF
COMMON /BLC3/ X(100),UE(100),P1(100),P2(100),GMTR(100)
COMMON /BLC7/ ETA(101),DETA(101),A(101)
COMMON /BLC8/ F(101,2),U(101,2),V(101,2),B(101,2)
COMMON /BLCS/ DLS(100),VW(100),CF(100),THT(100)

C

IF (NX.EQ.1) THEN
  DLS(NX) = 0.
  THT(NX) = 0.
  CF(NX) = 0.
  VW(NX) = V(1,2)
ELSE
  SQRX = SQRT(UE(NX)*X(NX)*RL)
  CF(NX) = 2. * V(1,2) * B(1,2) /SQRX
  VW(NX) = V(1,2)
  DLS(NX) = X(NX)/SQRX * (ETA(NP)-F(NP,2))
  U1 = U(1,2) * (1.-U(1,2))
  SUM = 0.
  DO 20 J = 2,NP
    U2 = U(J,2) * (1. - U(J,2))
    SUM = SUM + A(J) * (U1 + U2)
    U1 = U2
  20 CONTINUE
  THT(NX) = X(NX)/SQRX * SUM
ENDIF
SHIFT PROFILES FOR THE NEXT STATION

DO 175 J = 1, NPT
   F(J,1) = F(J,2)
   U(J,1) = U(J,2)
   V(J,1) = V(J,2)
   B(J,1) = B(J,2)
175 CONTINUE

RETURN
END

SUBROUTINE SETUP(X,Y,N,NLOWER,NUPPER)

REAL X(N), Y(N)
REAL PI
INTEGER FIGURE
COMMON /BOD/ NODTOT, COSTHE(100), SINTHE(100), NFLAG
COMMON /NUM/ PI, PI2INV
COMMON /LEROY/ NUMERAL
COMMON /FLAGGER/ FIGURE
COMMON /BRAVO/ NUMPTS
DIMENSION XE(100), YE(100)
DATA PI/3.1415926585/
PI2INV = 1./(2.*PI)
C NZERO = 31
C YMULT = 200

SET COORDINATES OF NODES ON BODY SURFACE

IF(FIGURE.EQ.2)THEN
   OPEN(UNIT=43, FILE='ME1.DAT', STATUS='OLD')
   READ(43,999) XERR, YERR
   DO I = 1, NUMPTS-2
      PRINT *, NUMPTS
      READ(43,999) XE(I), YE(I)
      CAM= XE(I)
      CAN= YE(I)
      X(I)= CAM
      Y(I)= CAN
   END DO
   CLOSE(UNIT=43)
ENDIF

WRITE (11,1000)
NPOINT = NLOWER
SIGN = -1.
NSTART = 0
DO 110 NSURF = 1, 2
   DO 100 N = 1, NPOINT
      FRACT = FLOAT(N-1)/FLOAT(NPOINT)
      Z = .5*(1. - COS(PI*FRACT))
      I = NSTART + N
      IF (NFLAG .EQ. 1. OR FIGURE .EQ. 2) GO TO 90
      CALL BODY(Z, SIGN, X(I), Y(I))
90  WRITE (11,1010) X(I), Y(I)
WRITE (15,1010) X(I),Y(I)
C PRINT 1010, X(I),Y(I)
100 CONTINUE
NPOINT = NUPPER
SIGN = 1.0
NSTART = NLOWER
110 CONTINUE
NUMERAL = NPOINT*2
NODTOT = NLOWER + NUPPER
X(NODTOT+1) = X(1)
Y(NODTOT+1) = Y(1)
C C SET SLOPES OF PANELS
C
DO 200 I = 1,NODTOT
DX = X(I+1) - X(I)
DY = Y(I+1) - Y(I)
DIST = SQRT(DX*DX + DY*DY)
SINTHE(I) = DY/DIST
COSTHE(I) = DX/DIST
200 CONTINUE
999 FORMAT(1X,F8.4,2X,F8.4)
1000 FORMAT(///,11X,' BODY SHAPE',///,12X,'X',9X,'Y',/)
1010 FORMAT(5X,F10.4,F10.4)
RETURN
END

SUBROUTINE SOLV3
C
COMMON /BLC2/ NX,NXT,NP,NPT,NTR,IT,ISF
COMMON /BLC7/ ETA(101),DETA(101),A(101)
COMMON /BLC8/ F(101,2),U(101,2),V(101,2),B(101,2)
COMMON /BLC9/ S1(101),S2(101),S3(101),S4(101),S5(101),
+S6(101),S7(101),S8(101),R1(101),R2(101),R3(101),R4(101)
COMMON /BLC6/ DELF(101),DELU(101),DELV(101)
DIMENSION A11(101),A12(101),A13(101),A14(101),
+A21(101),A22(101),A23(101),A24(101)
A11(1) = 1.
A12(1) = 0.
A13(1) = 0.
A21(1) = 0.
A22(1) = 1.
A23(1) = 0.
G11 = -1.
G12 = -A(2)
G13 = 0.
G21 = S4(2)
G23 = -S2(2)/A(2)
G22 = G23 + S6(2)
A11(2) = 1.
A12(2) = -A(2) - G13
A13(2) = A(2) * G13
A21(2) = S3(2)
A22(2) = S5(2) - G23
A23(2) = S1(2) + A(2) * G23
R1(2) = R1(2) - (G11*R1(1)+G12*R2(1)+G13*R3(1))
R2(2) = R2(2) - (G21*R1(1)+G22*R2(1)+G23*R3(1))

FORWARD SWEEP

DO 500 J=2,NP
   DEN = (A13(J-1)*A21(J-1)-A23(J-1)*A11(J-1)-A(J)*
         (A12(J-1)*A21(J-1)-A22(J-1)*A11(J-1))
   DEN1 = A22(J-1)*A(J)-A23(J-1)
   G11 = (A23(J-1)+A(J)*G11*(A(J)*A12(J-1)-A13(J-1)))/DEN
   G12 = -(A(J)*A(J)+G11*(A12(J-1)*A(J)-A13(J-1)))/DEN1
   G13 = (G11*A13(J-1)+G12*A23(J-1))/A(J)
   PRINT *, S2(J)
   PRINT *, A22(J-1)
   PRINT *, A(J)
   PRINT *, S4(J)
   PRINT *, A21(J-1)
   PRINT *, S6(J)
   PRINT *, NP
   PRINT *, J
   G21 = (S2(J)*A21(J-1)-S4(J)*A23(J-1)+A(J)*(S4(J)*
         A22(J-1)-S6(J)*A21(J-1)))/DEN
   G22 = (-S2(J)+S6(J)*A(J)-G21*(A(J)*A12(J-1)-
         A13(J-1)))/DEN1
   G23 = G21*A12(J-1)+G22*A22(J-1)-S6(J)
A11(J) = 1.
A12(J) = -A(J)-G13
A13(J) = A(J)*G13
A21(J) = S3(J)
A22(J) = S5(J) - G23
A23(J) = S1(J) + A(J) * G23
R1(J) = R1(J) - (G11*R1(J-1)+G12*R2(J-1)+G13*R3(J-1))
R2(J) = R2(J) - (G21*R1(J-1)+G22*R2(J-1)+G23*R3(J-1))

BACKWARD SWEEP

DELU(NP) = R3(NP)
E1 = R1(NP) - A12(NP)*DELU(NP)
E2 = R2(NP) - A22(NP)*DELU(NP)
   DELV(NP) = (E2*A11(NP)-E1*A21(NP))/(A23(NP)*A11(NP)-
            +A13(NP)*A21(NP))
DELF(NP) = (E1-A13(NP)*DELV(NP))/A11(NP)
DO 600 J = NP-1,1,-1
   E3 = R3(J)-DELU(J+1)+A(J+1)*DELV(J+1)
   DEN2 = A21(J)*A12(J)*A(J+1)-A21(J)*A13(J)+A(J+1)*
           +A22(J)*A11(J)+A23(J)*A11(J)
   DELV(J) = (A11(J)*R2(J)+E3*A22(J))-A21(J)*R1(J)-
            +E3*A21(J)*A12(J))/DEN2
   DELU(J) = -A(J+1) * DELV(J) - E3
   DELF(J) = (R1(J)-A12(J)*DELU(J)-A13(J)*DELV(J))/A11(J)

600 CONTINUE

DO 700 J = 1,NP
   F(J,2) = F(J,2) + DELF(J)
U(J,2) = U(J,2) + DELU(J)
V(J,2) = V(J,2) + DELV(J)
700 CONTINUE
U(1,2) = 0.
C
RETURN
END

SUBROUTINE SORTER(ARRAY, CARRY, NY)
C
C ARRAY = THE ARRAY WHICH IS BEING SORTED INTO ASCENDING ORDER
C CARRY = THE ARRAY VARIABLES STAYING WITH EACH RESP. ARRAY VAR. ABOVE
C NY = THE NUMBER OF ELEMENTS IN THE ARRAY TO BE SORTED
C
LOGICAL SORTED
DIMENSION ARRAY(101), CARRY(101)
SORTED = .FALSE.
NUMBER = NY
30 IF (.NOT. SORTED) THEN
    SORTED = .TRUE.
    DO 40 I = 1, NUMBER - 1
        IF(ARRAY(I).LT.ARRAY(I+1))THEN
            VALUE = ARRAY(I)
            VAL = CARRY(I)
            ARRAY(I) = ARRAY(I+1)
            CARRY(I) = CARRY(I+1)
            ARRAY(I+1) = VALUE
            CARRY(I+1) = VAL
            SORTED = .FALSE.
        ENDIF
    40 CONTINUE
    GO TO 30
ENDIF
RETURN
END

SUBROUTINE SORTNUM(ARRAY, CARRY, NY)
C
C ARRAY = THE ARRAY WHICH IS BEING SORTED INTO ASCENDING ORDER
C CARRY = THE ARRAY VARIABLES STAYING WITH EACH RESP. ARRAY VAR. ABOVE
C NY = THE NUMBER OF ELEMENTS IN THE ARRAY TO BE SORTED
C
LOGICAL SORTED
DIMENSION ARRAY(101), CARRY(101)
SORTED = .FALSE.
NUMBER = NY
30 IF (.NOT. SORTED) THEN
    SORTED = .TRUE.
    DO 40 I = 1, NUMBER - 1
        IF(ARRAY(I).GT.ARRAY(I+1))THEN
            VALUE = ARRAY(I)
            VAL = CARRY(I)
            ARRAY(I) = ARRAY(I+1)
            CARRY(I) = CARRY(I+1)
            ARRAY(I+1) = VALUE
            CARRY(I+1) = VAL
            SORTED = .FALSE.
        ENDIF
    40 CONTINUE
    GO TO 30
ENDIF
RETURN
END
CONTINUE
GO TO 30
ENDIF
RETURN
END

SUBROUTINE SPEVAL(N,COORX,COORY,FDP,XX,F)

C
THIS SUBROUTINE EVALUATES THE CUBIC SPLINE GIVEN
THE DERIVATIVES COMPUTED BY SUBROUTINE SPLINE.
THE INPUT PARAMETERS N,X,Y,FDP HAVE THE SAME
MEANING AS IN SPLINE.
XX = VALUE OF INDEPENDENT VARIABLE FOR WHICH
AN INTERPOLATED VALUE IS REQUESTED
F = THE INTERPOLATED RESULT
DIMENSION COORX(101),COORY(101),FDP(101)
THE FIRST STEP IS TO FIND THE PROPER INTERVAL
NM1 = N -1
DO 1 I=1,NM1
IF (XX.LE.COORX(I+1)) GO TO 10
1 CONTINUE
C
NOW EVALUATE THE CUBIC
10 DXM = XX - COORX(I)
DXP = COORX(I+1) - XX
DEL = COORX(I+1) - COORX(I)
F = FDP(I)*DXP*(DXP*DXP/DEL - DEL)/6.
1 +FDP(I+1)*DXM*(DXM*DXM/DEL - DEL)/6.
2 +COORY(I)*DXP/DEL + COORY(I+1)*DXM/DEL
RETURN
END

SUBROUTINE SPLINE (N,COORX,COORY,FDP)

C
THIS SUBROUTINE COMPUTES THE SECOND DERIVATIVES NEEDED
IN CUBIC SPLINE INTERPOLATION. THE INPUT DATA ARE:
N = NUMBER OF DATA POINTS
COORX = ARRAY CONTAINING THE VALUES OF THE INDEPENDENT VARIABLE
(ASSUMED TO BE ASCENDING ORDER)
COORY = ARRAY CONTAINING THE VALUES OF THE FUNCTION AT THE
DATA POINTS GIVEN IN THE COORX ARRAY
DIMENSION COORX(101),COORY(101),A(101),B(101)
DIMENSION C(101),R(101),FDP(101)
ALAMDA = 1
NM2 = N - 2
NM1 = N - 1
C(1) = COORX(2) - COORX(1)
DO 1 I=2,NM1
C(I) = COORX(I+1) - COORX(I)
A(I) = C(I-1)
B(I) = 2.*(A(I) + C(I))
R(I) = 6.*((COORY(I+1)-COORY(I))/C(I)-(COORY(I)
+ -COORY(I-1))/C(I-1))
1 CONTINUE
B(2) = B(2) + ALAMDA * C(1)
B(NM1) = B(NM1) + ALAMDA * C(NM1)
DO 2 I=3,NM1

\[ T = \frac{A(I)}{B(I-1)} \]
\[ B(I) = B(I) - T \times C(I-1) \]
\[ R(I) = R(I) - T \times R(I-1) \]

2 CONTINUE
\[ FDP(NM1) = \frac{R(NM1)}{B(NM1)} \]
DO 3 I=2,NM2
NMI = N - I
\[ FDP(NMI) = \frac{(R(NMI) - C(NMI) \times FDP(NMI+1))/B(NMI)} \]
3 CONTINUE

FDP(1) = \(\text{ALAMDA} \times FDP(2)\)
FDP(N) = \(\text{ALAMDA} \times FDP(NM1)\)

C DESIRED DERIVATIVES HAVE NOW BEEN DETERMINED
C RETURN TO MAIN PROGRAM
RETURN
END

SUBROUTINE VELDI(SINALF,COSALF,X,Y,N,NLOWER,NUPPER,ALPHA)
C
C COMPUTE AND PRINT OUT PRESSURE DISTRIBUTION
C
REAL X(N),Y(N)
INTEGER FLAG
COMMON /FINAL/FLAG,XREF,YCORD
COMMON /BOD/ NODTOT,COSTHE(100),SINTHE(100),NFLAG
COMMON /COF/ A(101,111),KUTTA
COMMON /CHARLIE/AIR
COMMON /ABLE/NUM
COMMON /CPD/ CP(100)
COMMON /VISCOUS/XCORD,YCOR,CPDAT
COMMON /NUM/ PI,PI2INV
COMMON /SKAL/ NZERO,YMULT
DIMENSION Q(150)
DIMENSION XCORD(100),YCOR(100),CPDAT(100)
DATA XCORD,YCOR,CPDAT /100*0.,100*0.,100*0./
C IF (FLAG.EQ.2)THEN
C DO I= 1,NODTOT
C WRITE(47,999)X(I),Y(I)
C END DO
C WRITE(47,*)NODTOT
C WRITE(47,*)NUM
C NODTOT = NODTOT-2
C CLOSE(UNIT=47)
C ENDIF
C YMULT = 20.0
PRINT 1000, ALPHA
WRITE (12,1000) ALPHA
AIR = ALPHA
PRINT 1005
WRITE (12,1005)
C
C RETRIEVE SOLUTION FROM A-MATRIX
C
DO 50 I = 1,NODTOT
50 Q(I) = A(I,KUTTA+1)
GAMMA = A(KUTTA,KUTTA+1)

FIND VTAND CP AT MID-POINT OF I-TH PANEL

DO 130 I = 1,NODTOT
XMID = .5*(X(I) + X(I+1))
YMID = .5*(Y(I) + Y(I+1))
XCORD(I)= XMID
YCOR(I) = YMID
VTANG = COSALF*COSTHE(I) + SINALF*SINTHE(I)

ADD CONTRIBUTION OF J-TH PANEL

DO 120 J = 1,NODTOT
FLOG = 0.0
FTAN = PI
IF (J .EQ. I) GO TO 100
DXJ = XMID - X(J)
DXJP = XMID - X(J+1)
DYJ = YMID - Y(J)
DYJP = YMID - Y(J+1)
FLOG = .5*ALOG((DXJP*DXJP+DYJP*DYJP)/(DXJ*DXJ+DYJ*DYJ))
FTAN = ATAN2(DYJP*DXJ-DXJP*DYJ,DXJP*DXJ+DYJP*DYJ)

CTIMTJ = COSTHE(I)*COSTHE(J) + SINTHE(I)*SINTHE(J)
STIMTJ = SINTHE(I)*COSTHE(J) - COSTHE(I)*SINTHE(J)
AA = PI2INV*(FTAN*CTIMTJ + FLOG*STIMTJ)
B = PI2INV*(FLOG*CTIMTJ - FTAN*STIMTJ)
VTANG = VTANG - B*Q(J) + GAMMA*AA

CONTINUE
CP(I) = 1. - VTANG*VTANG
CPDAT(I)=CP(I)
PRINT 1010, XMID,CP(I)
WRITE (12,1010) XMID,CP(I)
WRITE (14,1010) XMID,CP(I)

CONTINUE
NUM = NODTOT
CLOSE (UNIT=14)

FORMATT(1X,F8.4,2X,F8.4)

FORMAT(1X,'ANGLE OF ATTACK IN DEGREES = ',F8.3,/)  
1005 FORMAT(1X,'PRESSURE DISTRIBUTION',/,,14X,'X',9X,'CP',/)  
1010 FORMAT(10X,F10.4,F10.4)
RETURN
END

FUNCTION YREF(XNUM)
COMMON /LEROY/NUMERAL
COMMON /BRAVO/NUMPTS
COMMON /CHARLIE/NO
COMMON /FLAGGER/FIGURE
DIMENSION FDP(101),XX(101),YY(101)
DIMENSION XPOINT(101),YPOINT(101),XPOIN(101),YPOIN(101)
NO = NUMPTS

READ IN THE CURRENT SHAPE OF THE AIRFOIL

IF(FIGURE.EQ.2)NUMERAL=NUMERAL-2

160
OPEN(UNIT=15,FILE='BODYSHAPE.DAT',STATUS='OLD')
XX(1) = 0.0
YY(1) = 0.0
DO 30 I = 2,NUMERAL+1
   READ (15,*) XX(I),YY(I)
30    CONTINUE
XX(NUMERAL+2) = 1.
YY(NUMERAL+2) = 0.
CLOSE(UNIT=15)

CHECK THE INPUT OF THE AIRFOIL SHAPE DATA(OPTIONAL)
OPEN (66,FILE='MAKE.DAT',STATUS='NEW')
DO I = 1,NUMPTS
   WRITE(66,999)XX(I),YY(I)
END DO
999 FORMAT(1X,F8.4,2X,F8.4)
CLOSE (UNIT=66)

PROVIDE BODY ORDINATES FOR A SYMMETRIC BODY. TO DETERMINE
   THESE POINTS A CUBIC SPLINE INTERPOLATION SUBROUTINE WAS ADDED
   TO PROGRAM NEW_PANEL.

THE AIRFOIL SHAPE IS BEING SPLIT INTO UPPER AND LOWER SURFACES AND
   THEN FORMATTED FOR USE WITH THE SPLINE/SPEVAL Routines.

NOB = INT(NUMPTS/2)+1
DO I=1,INT(NUMPTS/2)+1
   DUMMY=XX(I)
   DUM  =YY(I)
   XPOINT(I)=DUMMY
   YPOINT(I)=DUM
END DO
DO I=INT(NUMPTS/2)+2,NUMPTS
   DUMM=XX(I)
   DU   =YY(I)
   XPOINT(I)=DUMM
   YPOINT(I)=DU
END DO
XPOINT(NUMPTS+1)=1.
YPOINT(NUMPTS+1)=0.
CALL SORTNUM(XPOINT,YPOINT,NOB)
CALL SORTNUM(XPOINT,YPOINT,NOB-1)

UPPER SURFACE Y COORDINATE DETERMINATION
IF (XNUM.GT.0.)THEN
   N = INT(NUMPTS/2)+1
   XPT = XNUM
   CALL SPLINE(N,XPOINT,YPOINT,FDP)
   CALL SPEVAL(N,XPOINT,YPOINT,FDP,XPT,F)
   YREF = F
ENDIF

LOWER SURFACE Y COORDINATE DETERMINATION
IF (XNUM.LT.0.)THEN
   N = INT(NUMPTS/2)
   XPT = XNUM
   CALL SPLINE(N,XPOIN,YPOIN,FDP)
   CALL SPEVAL(N,XPOIN,YPOIN,FDP,XPT,F)
   YREF = F
ENDIF
RETURN
END
APPENDIX H. PROGRAM NEW_VOR COMPUTER CODE

PROGRAM NEW_VOR

*** MODIFIED FOR USE ON THE MICROVAX/2000 BY J. A. CAMPBELL (JUL 88)
*** MODIFIED FOR GRAPHICAL OUTPUT AND/OR PRINTING OPTIONS BY C.M.
MACCALLISTER (AUG 89) FINAL UPDATES MADE 10 OCT 89 - (CMN)

CURRENT VERSION IS VERSION 5 INCORPORATING THE ABOVE NOTED CHANGES
DONE BY CRAIG MACCALLISTER FOR PROFESSOR J. V. HEALEY.

ORIGINAL IBM MAINFRAME PROGRAM WAS ADAPTED FROM JACK MORAN'S BOOK
'AN INTRODUCTION TO THEORETICAL AND COMPUTATIONAL AERODYNAMICS',
WILEY AND SONS, NEW YORK 1984. THE LISTING IS FOUND ON PAGE 151.

SIGNIFICANT UPGRADES HAVE BEEN IMPLEMENTED IN VERSION 5 WITH
RESPECT TO EASE OF OPERATION AND ERROR CORRECTION.

CHARACTER*1 PRINT, GRAPH, PLT1, PLT2, + PLT3
INTEGER NANS, GRAPHOPT, IFLAG
REAL ALPHA
DIMENSION GAM(350)
COMMON DX, DY, AR, PI, IOPT, NX, NY
COMMON /COUNTER/MANY
COMMON /ASPECT/RATIO
COMMON /COF/A(350,351), NEQNS
PI = 3.1415926585
NPASS = 1

FOLLOWING LINES FOR OUTPUT FILES ADDED BY J. A. CAMPBELL (JUL88)

OPEN FILE FOR COEFFICIENT OUTPUT
OPEN (UNIT=11,
  2 FILE= 'VORLAT4.DAT',
  2 ORGANIZATION= 'SEQUENTIAL',
  2 ACCESS= 'SEQUENTIAL',
  2 RECORDTYPE= 'VARIABLE',
  2 FORM= 'FORMATTED',
  2 STATUS= 'UNKNOWN')

OPEN (UNIT=12,
  2 FILE= 'VORLAT5.DAT',
  2 ORGANIZATION= 'SEQUENTIAL',
  2 ACCESS= 'SEQUENTIAL',
  2 RECORDTYPE= 'VARIABLE',
  2 FORM= 'FORMATTED',
  2 STATUS= 'UNKNOWN')

OPEN (UNIT=13,
  2 FILE= 'VORLAT6.DAT',
  2 ORGANIZATION= 'SEQUENTIAL',
  2 ACCESS= 'SEQUENTIAL',
  2 RECORDTYPE= 'VARIABLE',
  2 FORM= 'FORMATTED',

163
C

OPEN (UNIT=14,
2 FILE= 'VORLAT7.DAT',
2 ORGANIZATION= 'SEQUENTIAL',
2 ACCESS= 'SEQUENTIAL',
2 RECORDTYPE= 'VARIABLE',
2 FORM= 'FORMATTED',
2 STATUS= 'UNKNOWN')

C

C INPUT ASPECT RATIO (AR), NUMBERS OF VORTICES
C IN X- AND Y- DIRECTIONS (NX, NY) AND
C ANGLE OF ATTACK IN DEGREES (ALPHA)

C CALL LIBRARY ROUTINE TO CLEAR THE SCREEN, THE PRINT HEADER
CALL CLRSCRN
PRINT *
PRINT *, ' PROGRAM VORLAT : VERSION 5 : 10 OCTOBER 89 '
PRINT *
PRINT *, ' VORTEX-LATTICE METHOD USED TO DETERMINE SPANWISE'
PRINT *, ' LIFT DISTRIBUTION FOR A FLAT RECTANGULAR WING'
PRINT *
PRINT *
C

10 PRINT *, ' ENTER THE ASPECT RATIO? '
READ *, AR
RATIO = AR
IF (NPASS .GT. 1) GO TO 70
30 PRINT *, ' INPUT THE NUMBER OF VORTICES, IN THE X AND Y DIRECTIONS + (NX, NY)',
32 READ *, NX, NY
IF ((NX*NY) .GT. 350) THEN
   PRINT *, ' NX * NY MUST BE LESS THAN OR EQUAL TO 350.'
   PRINT *, ' PLEASE REENTER.'
   GO TO 32
END IF
MANY = NY
IF (NPASS .GT. 1) GO TO 70
50 PRINT *, ' WHAT IS THE ANGLE OF ATTACK IN DEGREES? '
52 READ *, ALPHA
IF (ALPHA .EQ. 0.) THEN
   PRINT *, ' ALPHA MUST BE GREATER THAN ZERO. PLEASE REENTER.'
   GO TO 52
ELSE IF (ALPHA .GT. 45.) THEN
   PRINT *, ' ALPHA MUST BE LESS THAN 45. PLEASE REENTER.'
   GO TO 52
END IF
IF (NPASS .GT. 1) GO TO 72
C 60 PRINT *, ' ENTER GRID SPACING OPTION (1 OR 2): (1) UNIFORM',
C + ', (2) COSINE'
C READ *, IOPT
IOPT = 1
NPASS = NPASS + 1
C
**** MAKE CALCULATIONS AND ECHO CHECK THE INPUT
C

164
\[ DX = 1. / \text{FLOAT}(NX) \]
\[ DY = \frac{\text{AR}}{(2. * \text{NY} + .5)} \]
\[ \text{NEQNS} = NX * \text{NY} \]

C
call library routine to clear the screen

72 call clrscrn

C
print *, ' the current values are: '
print *
print *, ' 1) aspect ratio . . . . . . . =', AR
print *, ' 2) number of vortices (NX, NY) =', NX, NY
print *, ' 3) angle of attack (degrees) =', ALPHA
print *, ' 4) grid spacing: (1) uniform, (2) cosine =', IOPT
print *
print *, ' the calculated parameters are: '
print *
if (IOPT .eq. 1) then
print *
print *, ' delta x =', DX
print *, ' delta y =', DY
else
print *, ' since cosine spacing was chosen, '
print *, ' delta x and delta y are variable. '
en end if
print *
print *, ' number of equations to solve =', NEQNS
print *
print *, ' are these values correct? (yes=1, no=2)'

75 call query (NANS)

iflag = nans
if (iflag .lt. 1 .or. iflag .gt. 2) then
print *, ' invalid entry. enter 1 or 2. '
go to 75
end if
if (iflag .eq. 1) go to 90
C

print *, ' which value do you wish to correct? '
print *
80 print *, ' enter 1, 2, 3 or 4'
call query (NANS)
iflag = nans
if (iflag .gt. 3) then
print *, ' invalid entry. enter 1, 2, 3 or 4. '
go to 80
end if
C

**** send control back to obtain correct data ****
go to (10, 30, 50) iflag
C ** change grid type ****
c if (iopt .eq. 1) then
C
iopt = 2
C
else
C
iopt = 1
C
end if
C
Go to 72
C
C
90 cosalf = cos(alpha*pi/180.)
SINALF = SIN(ALPHA*PI/180.)

INFORM OPERATOR THAT PROCESSING HAS STARTED
WRITE (6,1003)

SET COEFFICIENTS OF EQUATIONS FOR VORTEX STRENGTHS
DO 100 I = 1, NY
    DO 100 J = 1, NX
        IJ = (I - 1)*NX + J
        A(IJ,NEQNS + 1) = SINALF
    DO 100 K = 1, NY
    DO 100 L = 1, NX
        KL = (K-1)*NX + L
        CALL DNWASH (I,J,K,L,A(KL,IJ),1)
100 CONTINUE

SOLVE FOR VORTEX STRENGTHS
CALL GAUSS (1)
DO 200 I = 1, NY
    DO 200 J = 1, NX
        IJ = (I-1)*NX + J
    200 GAM(IJ) = A(IJ,NEQNS+1)

PRINT OUT HEADINGS FOR DATA
IF (IOPT.EQ. 1) WRITE (11,1000) NX,NY,AR,ALPHA
IF (IOPT.EQ. 2) WRITE (11,1001) NX,NY,AR,ALPHA
WRITE (6,1005)
WRITE (11,1005)

INITIALIZE TOTAL FORCE AND MOMENT COEFFICIENTS

CMT = 0.0
CDT = 0.0
CLT = 0.0

COMPUTE FORCE AND MOMENT COEFFICIENTS
Y = 0.00
CL = 0.00
CD = 0.00
XCP= .25
WRITE(12,1010) Y,CL,CD,XCP
WRITE(13,1010) Y,CL,CD,XCP
WRITE(14,1010) Y,CL,CD,XCP
DO 320 I = 1, NY
    CX = 0.0
    CZ = 0.0
    CM = 0.0
320 CONTINUE
DO 310 J = 1, NX
    IJ = (I-1)*NX + J
    W = 0.0
    DO 300 K = 1, NY
DO 300 L = 1,NX
   KL = (K-1)*NX + L
   CALL DNWASH(K,L,I,J,DELW,2)
   W = W + DELW*GAM(KL)
CONTINUE

300

CX = CX + GAM(IJ)*(W - SINALF)*2.
CZ = CZ + GAM(IJ)*COSALF*2.
IF (ILOPT .EQ. 1) THEN
   CM = CM - GAM(IJ)*DX*(J - .75)*COSALF*2.
ELSE
   CM = CM - GAM(IJ)*(FCOS(J,NX)+0.25*(FCOS(J+1,NX)
      - FCOS(J,NX)))*COSALF*2.
END IF

310 CONTINUE

CL = CZ*COSALF - CX*SINALF
CD = CZ*SINALF + CX*COSALF
IF (ILOPT .EQ. 1) THEN
   CLT = CLT + CL*DY*2./AR
   CDT = CDT + CD*DY*2./AR
   CMT = CMT + CM*2.*DY/AR
ELSE
   DELY = (0.5*AR - 0.25*DY)*(FSIN(I+1,NY) - FSIN(I,NY))
   DELY = (0.5*AR - 0.25*DY)*(FCOS(I+1,NY) - FCOS(I,NY))
   CLT = CLT + CL*DELY*2./AR
   CDT = CDT + CD*DELY*2./AR
   CMT = CMT + CM*DELY*2./AR
END IF

XCP = -CM/CL
IF (ILOPT .EQ. 1) THEN
   Y = (I-.5)*DY
ELSE
   CCC
      Y = (0.5*AR - 0.25*DY)*0.5*(FSIN(I,NY) + FSIN(I+1,NY))
      Y = (0.5*AR - 0.25*DY)*(FCOS(I,NY) +
         0.5*(FCOS(I+1,NY) - FCOS(I,NY)))
   END IF
WRITE(6,1010) Y,CL,CD,XCP
WRITE(11,1010) Y,CL,CD,XCP
WRITE(12,1010) Y,CL,CD,XCP
WRITE(13,1010) Y,CL,CD,XCP
WRITE(14,1010) Y,CL,CD,XCP

320 CONTINUE

XCP = -CMT/CLT
CDOCL2 = CDT/CLT**2
WRITE(6,1020) CLT,CDT,CDOCL2,CMT,XCP
WRITE(11,1020) CLT,CDT,CDOCL2,CMT,XCP
CLOSE(UNIT=11)
CLOSE(UNIT=12)
CLOSE(UNIT=13)
CLOSE(UNIT=14)

PRINT *
PRINT *, ' THE COEFFICIENT OUTPUT DATA FOR LIFT, DRAG AND'
PRINT *, ' PRESSURE HAS BEEN WRITTEN TO FILE VORLAT4.DAT.'
PRINT *
PRINT *, ' WOULD YOU LIKE TO PRINT THE RESULTS (Y/N),'
PRINT *
READ 1002, PRINT
IF (PRINT.EQ.'Y'.OR.PRINT.EQ.'y')THEN
   CALL LIB$SPAWN('PRINT VORLAT4.DAT')
ENDIF
PRINT *
PRINT *, 'WOULD YOU LIKE TO GRAPH THE RESULTS (Y/N)'
PRINT *
READ 1002, GRAPH
IF (GRAPH.EQ.'Y'.OR.GRAPH.EQ.'y')THEN
   PRINT *
   PRINT *, 'WHICH OF THE FOLLOWING RELATIONSHIPS'
   PRINT *, 'DO YOU WANT TO GRAPH?'
   PRINT *, '* 1) CL VS. Y'
   PRINT *, '  2) CD VS. Y'
   PRINT *, '  3) CL VS. CD'
   PRINT *, '  4) NONE'
   PRINT *
   PRINT *, 'INPUT OPTION NO.(1,2,3 OR 4)'
   READ 1006, GRAPHOPT
   IF (GRAPHOPT.LT.1.OR.GRAPHOPT.GT.4) THEN
      PRINT *, 'INVALID ENTRY, ENTER INTEGER BETWEEN'
      PRINT *, 'ONE(1) AND FOUR(4).'
      PRINT *, '
      GO TO 65
   ENDIF
   IF (GRAPHOPT.EQ.1) THEN
      CALL PLOT1(ALPHA)
      CALL LIB$SPAWN('RENDER/DEVICE=LA210/DRAFT_QUALITY/PAPER_+
                      SIZE=A P1.UIS')
      PRINT *, '
      PRINT *, 'WOULD YOU LIKE A PRINT OF THIS PLOT? (Y/N)'
      PRINT *
      READ 1002, PLT1
      IF (PLT1.EQ.'Y'.OR.PLT1.EQ.'y')THEN
         CALL LIB$SPAWN('PRINT P1.REN')
      ENDIF
      GO TO 46
   ENDIF
   IF (GRAPHOPT.EQ.2) THEN
      CALL PLOT2(ALPHA)
      CALL LIB$SPAWN('RENDER/DEVICE=LA210/DRAFT_QUALITY/PAPER_+
                      SIZE=A P2.UIS')
      PRINT *, '
      PRINT *, 'WOULD YOU LIKE A PRINT OF THIS PLOT? (Y/N)'
      PRINT *
      READ 1002, PLT2
      IF (PLT2.EQ.'Y'.OR.PLT2.EQ.'y')THEN
         CALL LIB$SPAWN('PRINT P2.REN')
      ENDIF
      GO TO 46
   ENDIF
   IF (GRAPHOPT.EQ.3) THEN
      CALL PLOT3(ALPHA)
      CALL LIB$SPAWN('RENDER/DEVICE=LA210/DRAFT_QUALITY/PAPER_
+SIZE=A P3.UIS')
PRINT *, ', ',
CALL LIB$SPAWN('CONTINUE')
PRINT *, 'WOULD YOU LIKE A PRINT OF THIS PLOT? (Y/N)'
PRINT *,', '
READ 1002, PLT3
IF (PLT3.EQ..'Y'.OR.PLT3.EQ..'y')THEN
   CALL LIB$SPAWN('PRINT P3.REN')
ENDIF
GO TO 46
ENDIF
IF (GRAPHOPT .EQ. 4) THEN
   GO TO 64
ENDIF
ENDIF
C  OPTION TO MAKE ANOTHER RUN
64 PRINT *
   PRINT *, 'DO YOU WISH TO: ',
   PRINT *, '1) MAKE ANOTHER RUN OR'
   PRINT *, '2) END THIS SESSION'
   PRINT *, 'ENTER 1 OR 2.'
   PRINT *
   CALL QUERY (NANS)
   CALL CLRSCRN
   IF (NANS .EQ. 1) GO TO 72
STOP
1000 FORMAT(/,10X,'** UNIFORM GRID SPACING **',/,,10X,
      + NX= ',I2,' NY= ',I2,' ASPECT RATIO = ',F5.2,
      &/,16X,'ANGLE OF ATTACK = ',F5.2)
1001 FORMAT(/,10X,'** COSINE GRID SPACING **',/,,10X,
      + NX= ',I2,' NY= ',I2,' ASPECT RATIO = ',F5.2,
      +/,16X,'ANGLE OF ATTACK = ',F5.2)
1002 FORMAT(A1)
1006 FORMAT(I1)
1003 FORMAT(/,'PROCESSING BEGINS....',/)
1005 FORMAT(/,,10X,' Y CL(Y) CD(Y) XCP(Y)',/)
1010 FORMAT(10X,F6.3,3F10.5)
1020 FORMAT(/,,10X,' CL = ',F12.5/,10X,' CD = ',F14.7/,10X,
      + CD/CL2 = ',F7.4/,10X,' CMLE = ',F11.6/,10X,' XCP = ',F11.5)
END

SUBROUTINE CLRSCRN

C LIBRARY ROUTINE TO CLEAR THE SCREEN.
C
ISTAT = LIB$ERASE_PAGE (1,1)
RETURN
END

C SUBROUTINE QUERY(NANS)
C
C ROUTINE TO TRAP ERRORS CAUSED BY IMPROPER RESPONSES TO QUESTIONS.
C THE COMPUTER GENERATES AN ERROR WHEN A CHARACTER IS SUPPLIED TO
C A QUESTION EXPECTING AN INTEGER OR REAL VALUE.
C
NQTEST=0
CONTINUE
IF (NQTEST .GT. 0) THEN
   PRINT *, ' CHARACTER VALUES ARE NOT VALID.'
   PRINT *, ' PLEASE ENTER AN INTEGER VALUE.'
END IF
NQTEST = NQTEST + 1
READ (5,*,ERR=1)NANS
RETURN
END

SUBROUTINE DNWASH(I,J,K,L,W,IND)

COMMON DX,DY,AR,PI,IOPT,NX,NY

C
C COMPUTE DOWNWASH ON PANEL CENTERED AT (L-.5)DX,(K-.5)DY
C DUE TO VORTICES AT PANELS CENTERED AT (J-.5)DX,+(I-.5)DY

C
IF (IOPT .EQ. 2) GO TO 50
XA = DX*(J - .75)
YA = DY*(I - 1)
YB = DY*I
IF (IND .EQ. 1) XP = DX*(L - .25)
IF (IND .EQ. 2) XP = DX*(L - .75)
YP = DY*(K-.5)
GO TO 60

C THE FOLLOWING LINES HANDLE THE COSINE SPACING SCHEME
C FAC IS THE HALF SPAN MINUS A 1/4 LATTICE WIDTH INSET.

50 FAC = 0.5*AR - 0.25*DY
XA = FCOS(J,NX) + 0.25*(FCOS(J+1,NX) - FCOS(J,NX))
CCC YA = FAC * FSIN(I-1,NY)
CCC YB = FAC * FSIN(I,NY)
YB = FAC * FCOS(I+1,NY)
YP = FAC*0.5*(FSIN(K,NY) + FSIN(K-1,NY))
CCC YP = FAC*(FCOS(K,NY) + 0.5*(FCOS(K+1,NY) - FCOS(K,NY)))

60 W = WHV(XP,YP,XA,YA) - WHV(XP,YP,XA, YB)
+ - WHV(XP,YP,XA,-YA) + WHV(XP,YP,XA,-YB)
W = W*.25/3.1415926585
RETURN
END

FUNCTION WHV(X1,Y1,X2,Y2)
   IF (X1 .EQ. X2) GO TO 100
   WHV = (1. + SQRT(( X1-X2)**2 + (Y1-Y2)**2)/(X1 - X2))
+ /(Y1 - Y2)
   RETURN
100 WHV = 1./(Y1 - Y2)
RETURN
END

C THIS RETURNS THE NONDIMENSIONAL X COORD OF EACH SECTION BOUNDARY
C
FUNCTION FCOS(I,N)
PI = 3.1415926585
FRACT = FLOAT(I-1)/FLOAT(N)
FCOS = 0.5 * (1. - COS(PI*FRACT))
RETURN
END

THIS RETURNS THE NONDIMENSIONAL Y COORD OF EACH SECTION BOUNDARY
THIS WAS INTENDED TO IMPLEMENT THE SIN-LAW LATTICE SPACING SCHEME
REFERRED TO BY GARY HOUGH, JOU. OF ACFT., MAY 1973, VOL. 10, NO. 5

FUNCTION FSIN(I,N)
PI = 3.1415926585
FRACT = FLOAT(I)/FLOAT(N)
FSIN = (SIN(.5*PI*FRACT))
RETURN
END

SUBROUTINE GAUSS (NRHS)

SOLUTION OF LINEAR ALGEBRAIC SYSTEM BY
GAUSS ELIMINATION WITH PARTIAL PIVOTING

A     = COEFFICIENT MATRIX
NEQNS = NUMBER OF EQUATIONS
NRHS  = NUMBER OF RIGHT HAND SIDES

RIGHT-HAND SIDES AND SOLUTIONS STORED IN
COLUMNS NEQNS+1 THRU NEQNS+NRHS OF A

COMMON DX, DY, AR, PI
COMMON /COF/ A(350, 351), NEQNS
NP    = NEQNS + 1
NTOT  = NEQNS + NRHS

GAUSS REDUCTION

DO 150 I = 2, NEQNS

-- SEARCH FOR LARGEST ENTRY IN (I-1)TH COLUMN
ON OR BELOW MAIN DIAGONAL

IM     = I - 1
IMAX   = IM
AMAX   = ABS(A(IM, IM))
DO 110 J = I, NEQNS
    IF (AMAX .GE. ABS(A(J, IM))) GO TO 110
    IMAX = J
    AMAX = ABS(A(J, IM))
110 CONTINUE

-- SWITCH (I-1)TH AND IMAXTH EQUATIONS

IF (IMAX .NE. IM) GO TO 140
DO 130 J = IM, NTOT
    TEMP = A(IM, J)
    A(IM, J) = A(IMAX, J)
130 CONTINUE
A(IMAX,J) = TEMP
CONTINUE

C ELIMINATE (I-1)TH UNKNOWN FROM ITH THRU (NEQNS)TH EQUATIONS

DO 150 J = I,NEQNS
    R = A(J,IM)/A(IM,IM)
    DO 150 K = I,NTOT
    A(J,K) = A(J,K) - R*A(IM,K)
150
BACK SUBSTITUTION

DO 220 K = NP,NTOT
    A(NEQNS,K) = A(NEQNS,K)/A(NEQNS,NEQNS)
    DO 210 L = 2,NEQNS
        I = NEQNS + 1 - L
        IP = I + 1
        DO 200 J = IP,NEQNS
    200 A(I,K) = A(I,K) - A(I,J)*A(J,K)
    210 A(I,K) = A(I,K)/A(I,I)
220 CONTINUE
RETURN
END

SUBROUTINE MAXMIN(ARRAY,NY,VALMAX,VALMIN)

ARRAY = THE ARRAY WHICH IS BEING SORTED INTO ASCENDING ORDER
NUMBER= THE NUMBER OF ELEMENTS IN THE ARRAY TO BE SORTED
VALMAX= LARGEST VALUE IN THE ARRAY
VALMIN= SMALLEST VALUE IN THE ARRAY
REAL VALMAX,VALMIN
INTEGER NUMBER
LOGICAL SORTED
DIMENSION ARRAY(IOO)
SORTED = .FALSE.
NUMBER = NY

30 IF (.NOT.SORTED) THEN
    SORTED = .TRUE.
    DO 40 I = 1,NUMBER - 1
        IF(ARRAY(I).GT.ARRAY(I+1))THEN
            VALUE = ARRAY(I)
            ARRAY(I) = ARRAY(I+1)
            ARRAY(I+1) = VALUE
            SORTED = .FALSE.
        ENDIF
    40 CONTINUE
GO TO 30
ENDIF
VALMAX = ARRAY(NUMBER)
VALMIN = ARRAY(1)
RETURN
END
SUBROUTINE PLOT1(ALPHA)

DEFINE IPACK ARRAY FOR LEGEND
INTEGER*4 IPACK(35)
INTEGER NUMBER
REAL YY(100),CD(100),CL(100),XCP(100)
REAL BA,MAX,MIN,AR
CHARACTER*40 L1
COMMON /COUNTER/MANY
COMMON /ASPECT/RATIO
DIMENSION YY1(100),CL1(100)

READ ELEMENTS OF UNIT 12 INTO ARRAYS TO PLOT
NUMBER = MANY
BA=ALPHA
OPEN(UNIT=12,FILE='VORLAT5.DAT',STATUS='OLD')
DO 25 I = 1,MANY+1
   READ (12,*)YY(I),CL(I),CD(I),XCP(I)
   DUM = YY(I)
   DUMM = CL(I)
   YY1(I)=DUM
   CL1(I)=DUMM
25 CONTINUE
CLOSE(UNIT=12)
CALL MAXMIN(CL1,MANY+1,MAX,MIN)
CALL MAXMIN(YY1,MANY+1,VALMAX,VALMIN)

DEFINE AND ASSIGN LEGEND CHARACTER STRINGS
L1 = 'CL VALUES$'

INITIALIZE THE GRAPHICS SYSTEM
CALL INIT

LABEL X AND Y AXES USING SELF COUNTING STRINGS
CALL XNAME('Y$',100)
CALL YNAME('CL$',100)

DEFINE PLOT AREA TO BE 6 INCHES BY 8 INCHES
CALL AREA2D(6.0,8.0)

DEFINE HEADING LABEL
CALL HEADING('CL VS. Y$',-100,2.0,1)

PLOT ADDITIONAL TICK MARKS
CALL XTICKS(1)
CALL YTICKS(1)

PACK LEGEND LABELS INTO ARRAY IPACK
CALL LINES(L1,IPACK,1)

SET UP AXES
CALL GRAP(0.,((VALMAX-VALMIN)/5.),VALMAX.1),0.,
         ((MAX-MIN)/2.),(MAX+1))

FRAME THE SUBPLOT AREA
CALL FRAME

PLOT PRESSURE DISTRIBUTION DATA WITH A THICK LINE AND MARKER 15
CALL MARKER(15)
CALL THKCRV(.04)
CALL CURVE(YY,CL,NUMBER+1,1)

PLOT MESSAGES
CALL MESSAG('FLAT RECTANGULAR WING$',100,
             +.75,1.5)
CALL MESSAG('ASPECT RATIO(AR) =$',100,.75,1.)
CALL REALNO(RATIO,2,3.5,1.)
CALL MESSAG('ANGLE OF ATTACK =$',100,.75,.5)
CALL REALNO(BA,2,3.5,.5)
C CHANGE LEGEND NAME TO "2-D PLOT"
   CALL MYLEGN('2-D PLOT$',100)
C PLOT LEGEND
   CALL LEGEND(IPACK,1,2.0,7.0)
C END PLOT
   CALL ENDPL(0)
C CREATE GRAPHICS METAFILE P1.UIS
   CALL METAFL(1)
C TERMINATE PLOT AT END OF PLOTTING SESSION
   CALL DONEPL
   RETURN
END

SUBROUTINE PLOT2(ALPHA)

C DEFINE IPACK ARRAY FOR LEGEND
INTEGER*4 IPACK(35)
INTEGER NUM,MANY
REAL YY(100),CD(100),CL(100),XCP(100)
REAL BAD,MAX,MIN,ALPHA,AR
CHARACTER*40 LI
COMMON /COUNTER/MANY
COMMON /ASPECT/RATIO
DIMENSION YY1(100),CD1(100)

C READ ELEMENTS OF UNIT 13 INTO ARRAYS TO PLOT
NUM = MANY
OPEN(UNIT=13,FILE='VORLAT6.DAT',STATUS='UNKNOWN')
DO 25 I = 1,MANY+1
   READ (13,*)YY(I),CL(I),CD(I),XCP(I)
   DUM = YY(I)
   DUMM = CD(I)
   YY1(I)=DUM
   CD1(I)=DUMM
25 CONTINUE
CLOSE(UNIT=13)
CALL MAXMIN(CD1,MANY+1,MAX,MIN)
CALL MAXMIN(YY1,MANY+1,VALMAX,VALMIN)
C DEFINE AND ASSIGN LEGEND CHARACTER STRINGS
LI = 'CD VALUES$
C INITIALIZE THE GRAPHICS SYSTEM
   CALL INIT
C LABEL X AND Y AXES USING SELF COUNTING STRINGS
   CALL XNAME('Y$',100)
   CALL YNAME('CD$',100)
C DEFINE PLOT AREA TO BE 6 INCHES BY 8 INCHES
   CALL AREA2D(6.0,8.0)
C DEFINE HEADING LABEL
   CALL HEADIN('CD VS. Y$','-100,2.0,1')
C PLOT ADDITIONAL TICK MARKS
   CALL XTICKS(1)
   CALL YTICKS(1)
C PACK LEGEND LABELS INTO ARRAY IPACK
   CALL LINES(LI,IPACK,1)
C SET UP AXIS
   CALL GRAF(0.,((VALMAX-VALMIN)/5.),(VALMAX+.1),0.,
+ ((MAX-MIN)/3.,(MAX+.001))
C FRAME THE SUBPLOT AREA
   CALL FRAME
C PLOT PRESSURE DISTRIBUTION DATA WITH A THICK LINE AND MARKER 15
   CALL MARKER(15)
   CALL THKCRV(.04)
   CALL CURVE(YY,CD,NUM+1,1)
C PLOT MESSAGES
   CALL MESSAG('FLAT RECTANGULAR WING$',100,
+ .75,1.5)
   CALL MESSAG('ASPECT RATIO(AR) = $',100,.75,1.)
   CALL REALNO(RATIO,2,3.5,1.)
   CALL MESSAG('ANGLE OF ATTACK = $',100,.75,.5)
   CALL REALNO(ALPHA,2,3.5,.5)
C CHANGE LEGEND NAME TO "2-D PLOT"
   CALL MYLEGN('2-D PLOT$',100)
C PLOT LEGEND
   CALL LEGEND(IPACK,1,2.0,7.0)
C END PLOT
   CALL ENDPL(0)
C CREATE GRAPHICS METAFILE P2.UIS
   CALL METAFL(2)
C TERMINATE PLOT AT END OF PLOTTING SESSION
   CALL DONEPL
   RETURN
END

SUBROUTINE PLOT3(ALPHA)
C C DEFINE IPACK ARRAY FOR LEGEND
   INTEGER*4 IPACK(35)
   INTEGER NUMB
   REAL YY(100),CD(100),CL(100),XCP(100)
   REAL MAXY,MINY,MAX,MIN,ALPHA,AR,BED
   CHARACTER*40 L1
   COMMON /COUNTER/MANY
   COMMON /ASPECT/RATIO
C READ ELEMENTS OF UNIT 14 INTO ARRAYS TO PLOT
   NUMB = MANY
   OPEN(UNIT=14,FILE='VORLAT7.DAT',STATUS='OLD')
   DO 25 I = 1,MANY+1
      READ (14,*)YY(I),CL(I),CD(I),XCP(I)
25   CONTINUE
   CLOSE(UNIT=14)
   CALL MAXMIN(CL,MANY+1,MAX,MIN)
   CALL MAXMIN(CD,MANY+1,MAXY,MINY)
C DEFINE AND ASSIGN LEGEND CHARACTER STRINGS
   L1 = 'CL/CD VALUES$
C INITIALIZE THE GRAPHICS SYSTEM
   CALL INIT
C LABEL X AND Y AXES USING SELF COUNTING STRINGS
   CALL XNAME('CD$',100)
   CALL YNAME('CL$',100)
C DEFINE PLOT AREA TO BE 6 INCHES BY 8 INCHES
   CALL AREA2D(6.0,8.0)
C DEFINE HEADING LABEL
CALL HEADIN('CL VS. CDS',-100,2.,1)
C PLOT ADDITIONAL TICK MARKS
   CALL XTICKS(1)
   CALL YTICKS(1)
C PACK LEGEND LABELS INTO ARRAY IPACK
   CALL LINES(L1,IPACK,1)
C SET UP AXIS
   CALL GRAF(0.,((MAXY-MINY)/5.),MAXY+.001),
   +0.,((MAX-MIN)/5),(MAX+.01))
C FRAME THE SUBPLOT AREA
   CALL FRAME
C PLOT PRESSURE DISTRIBUTION DATA WITH A THICK LINE AND MARKER 15
   CALL MARKER(15)
   CALL THKCRV(.04)
   CALL CURVE(CD,CL,NUMB,1)
C PLOT MESSAGES
   CALL MESSAG('FLAT RECTANGULAR WING$',100,
   + 1.75,1.5)
   CALL MESSAG('ASPECT RATIO(AR) = $',100,1.75,1.)
   CALL REALNO(RATIO,2,4.5,1.)
   CALL MESSAG('ANGLE OF ATTACK = $',100,1.75,.5)
   CALL REALNO(ALPHA,2,4.5,.5)
C CHANGE LEGEND NAME TO "2-D PLOT"
   CALL MYLEGN('2-D PLOT$',100)
C PLOT LEGEND
   CALL LEGEND(IPACK,1,2.0,7.0)
C END PLOT
   CALL ENDPL(0)
C CREATE GRAPHICS METAFILE P3.UIS
   CALL METAFL(3)
C TERMINATE PLOT AT END OF PLOTTING SESSION
   CALL DONEPL
   RETURN
END

END
APPENDIX I. PROGRAM SUB COMPUTER CODE

PROGRAM SUB

*** MODIFIED FOR USE ON THE MICROVAX/2000 BY R. MARGASON.
*** MODIFIED FOR GRAPHICAL OUTPUT AND/OR PRINTING OPTIONS BY C. M.
MACALLISTER (AUB 89) FINAL UPDATES MADE OCT 89 - (CMM).

THE SUB PROGRAM HAS BEEN ADAPTED FROM A NATIONAL AERONAUTICS AND
SPACE ADMINISTRATION (NASA) FORTRAN PROGRAM AND HAS BEEN USED CON-
SIDERABLY AT THE LANGLEY RESEARCH CENTER. THE PURPOSE OF THE SUB
PROGRAM IS TO ESTIMATE THE SUBSONIC AERODYNAMIC CHARACTERISTICS
OF COMPLEX PLANFORMS. THE PROGRAM REPRESENTS A LIFTING PLANFORM
WITH A VORTEX LATTICE. A RELATIVELY COMPLEX PLANFORM MAY BE
ANALYZED BY CREATING THE PLANFORM WITH UP TO 24 LINE SEGMENTS ON
A SEMISPAWN. ADDITIONALLY, THOSE LINE SEGMENTS MAY HAVE AN OUT-
BOARD VARIABLE-SWEEP PANEL OR THEY MAY HAVE SEVERAL DIHEDRAL ANGLES
ACROSS THE SPAN. FURTHERMORE, TWO PLANFORMS MAY BE USED TOGETHER
TO REPRESENT A COMBINATION OF WINGS AND TAILS OR WING, BODIES, AND
TAILS. THE USE OF THIS PROGRAM IS CONFINED TO THE SUBSONIC FLOW
REGIME. ADDITIONALLY, THE PLANFORM IS IN STEADY, IRROTATIONAL,
INVISCID, INCOMPRESSIBLE, ATTACHED FLOW CONDITIONS.

CHARACTER*20 CASEFN, OUTFIL
INTEGER GRAPHOPT,OUTER,LSTA,NSTA,METH
CHARACTER*1 PRINT,GRAPH,COPY,PLOT1,PLOT2,PLOT3
CHARACTER*1 PLOT4,PLOT5,PLOT6
REAL MACH
COMMON/SHIP/VIC,SCW
COMMON/ALL/ BOT,M,BETA,PTEST,QTEST,TBLSCW(50),Q(300),PN(300),
1 PV(300),ALP(300),S(300),PSI(300),PHI(300),ZH(50)
COMMON/TOTHRE/ CIR(300,2),SECTRST(50)
COMMON/ONETHRE/TWIST(2),CREF,SREF,CAVE,CLDES,TRUE,AR,ARTRUE,
1 RTCDHT(2),CONFIG,NSSWSV(2),MSV(2),KBOT,PLAN,IPLAN,MACH
2 ,SSWWA(50)
COMMON/MAINONE/ICODEOF,TOTAL,AAN(2),XS(2),YS(2),KFCTS(2)
1 ,XREG(25,2),YREG(25,2),AREG(25,2),DIH(25,2),MCD(25,2)
2 ,XX (25,2),YY (25,2),AS (25,2),TTWD(25,2),MMCD(25,2)
3 ,AN(2),ZZ (25,2)
7 FORMAT( //10X,16,'HORSESHOE VORTICES LAYOUT, THIS IS MORE THAN
1 THE 300 MAXIMUM. THIS CONFIGURATION IS ABORTED. ')
8 FORMAT( // 10X, 16,' ROWS OF HORSESHOE VORTICES LAYOUT. THIS I
1S MORE THAN THE 50 MAXIMUM. THIS CONFIGURATION IS ABORTED. ')
9 FORMAT( //10X, 'PLANFORM',16,' HAS',16,
1 ' BREAKPOINTS. THE MAXIMUM DIMENSIONED IS 25. THE CONFIGURATION I
2S ABORTED. ')
100 FORMAT(A20)
101 FORMAT( ///, START OF A NEW CASE, CASE FILE NAME IS ', A20//)
102 FORMAT( /// , THE OUTPUT FILE NAME IS "OUTFILE.DAT", ' //)

VOXETE LATTICE AERODYNAMIC COMPUTATION
NASA-LRC PROGRAM NO. A2794

METH = INT(VIC)
NMAX = 300
ICODEOF = 0
TOTAL = 0

C INPUT FILE NAME OF THE CASE TO BE RUN

1 WRITE(*,*) '', WRITE(*,*) ' PROGRAM SUB - SUBSONIC VORTEX LATTICE ANALYSIS'
   PRINT *, ' ', WRITE(*,*) ' ENTER INPUT DATA FILE NAME ' WRITE(*,*) 'USE LAST.END AS DATA FILE NAME TO STOP THE PROGRAM'
   PRINT *, ' ', READ(*,100) CASEFN
   IF (CASEFN.EQ.'LAST.END') GO TO 999
   IF (CASEFN.EQ.'last.end') GO TO 999
   OPEN(28,FILE=CASEFN,STATUS='OLD')

C CREATE FILES WHICH WILL BE USED TO PLOT THE RESULTS

C OPEN FILE FOR SPANWISE PRESSURE DISTRIBUTION OUTPUT
   OPEN (UNIT=11,
      FILE='AERO1.DAT',
      ORGANIZATION='SEQUENTIAL',
      ACCESS='SEQUENTIAL',
      RECORDTYPE='VARIABLE',
      FORM='FORMATTED',
      STATUS='UNKNOWN')

C OPEN FILE FOR DRAG POLAR OUTPUT
   OPEN (UNIT=12,
      FILE='AERO2.DAT',
      ORGANIZATION='SEQUENTIAL',
      ACCESS='SEQUENTIAL',
      RECORDTYPE='VARIABLE',
      FORM='FORMATTED',
      STATUS='UNKNOWN')

C OPEN FILE FOR CP OUTPUT
   OPEN (UNIT=13,
      FILE='AERO3.DAT',
      ORGANIZATION='SEQUENTIAL',
      ACCESS='SEQUENTIAL',
      RECORDTYPE='VARIABLE',
      FORM='FORMATTED',
      STATUS='UNKNOWN')

C OPEN (29, FILE ='OUTFILE.DAT', STATUS = 'NEW' )

C WRITE(29,101) CASEFN WRITE(29,102)

C 11 CALL GEOM
   IF(ICODEOF.GT.0) GO TO 99
   IF(M.GT.NMAX) GO TO 2
   NSW = NSSWSV(1) + NSSWSV(2)
   IF ( NSW.GT.50 ) GO TO 4
   ITSV = 0
DO 10 IT=1,IPLAN
IF ( AN(IT).LE.25. ) GO TO 10
WRITE (29,9) IT,AN(IT)
ITSV = 1
10 CONTINUE
IF (ITSV.GT.0) GO TO 5
GO TO 3
4 WRITE (29,8) NSW
GO TO 5
2 WRITE(29,7) M
GO TO 5
3 CALL MATX
CALL AERO
5 TOTAL=TOTAL-1.
IF ( TOTAL.GT.0. ) GO TO 11
99 CLOSE(UNIT=28)
CLOSE(UNIT=29)
PRINT *
PRINT *, ' PROGRAM RESULTS HAVE BEEN WRITTEN TO THE FILE'
PRINT *, ' OUTFILE.DAT.'
PRINT *, 'WOULD YOU LIKE A PRINTED COPY OF THIS OUTPUT FILE?'
PRINT *, ' YES OR NO (Y/N)'
PRINT *
READ 1002, PRINT
1002 FORMAT(A1)
IF (PRINT.EQ.'Y')THEN
   CALL LIB$SPAWN('PRINT OUTFILE.DAT')
ENDIF
PRINT *
PRINT *, 'WOULD YOU LIKE THE OUTPUT FILE COPIED TO ANOTHER'
PRINT *, ' FILE FOR FUTURE REFERENCE (Y/N) ? '
PRINT *
READ 1002,COPY
IF (COPY .EQ.'Y') THEN
   PRINT *, 'WHAT NAME WOULD YOU LIKE FOR THE OUTPUT FILE?'
   PRINT *
   PRINT *, ' 1) VIGILANTE.DAT'
   PRINT *, ' 2) CORSAIR.DAT'
   PRINT *, ' 3) HAWKEYE.DAT'
   PRINT *, ' 4) SKYHAWK.DAT'
   PRINT *
   PRINT *, 'ENTER 1,2,3 OR 4 '
69 READ 1006, OUTER
IF (OUTER.LT.1 OR OUTER.GT.4) THEN
   PRINT *, 'INVALID ENTRY, ENTER INTEGER BETWEEN'
   PRINT *, ' ONE(1) AND FOUR(4).'
   PRINT *
   GO TO 69
ENDIF
IF (OUTER.EQ.1) CALL LIB$SPAWN('COPY OUTFILE.DAT VIGILANTE.DAT')
IF (OUTER.EQ.2) CALL LIB$SPAWN('COPY OUTFILE.DAT CORSAIR.DAT')
IF (OUTER.EQ.3) CALL LIB$SPAWN('COPY OUTFILE.DAT HAWKEYE.DAT')
IF (OUTER.EQ.4) CALL LIB$SPAWN('COPY OUTFILE.DAT SKYHAWK.DAT')
ENDIF
PRINT *, 'WOULD YOU LIKE TO GRAPH THE RESULTS (Y/N) ?'
PRINT *

179
Which of the following relationships do you want plotted?

1) Induced drag coeff vs. 2Y/B
2) LE edge thrust coeff vs. 2Y/B
3) Suction coeff vs. 2Y/B
4) Span load coeff vs. 2Y/B
5) CL ratio vs. 2Y/B
6) Delta CP vs. X c/4
7) None

Input option no. (1, 2, 3, 4, 5, 6 or 7)
IF (GRAPHOPT .EQ. 3) THEN
  CALL GRAPH3
ENDIF
GO TO 41
ENDIF

IF (GRAPHOPT .EQ. 4) THEN
  CALL GRAPH4
ENDIF
GO TO 41
ENDIF

IF (GRAPHOPT .EQ. 5) THEN
  CALL GRAPH5
ENDIF
GO TO 41
ENDIF

C *************
IF (GRAPHOPT .EQ. 6) THEN
  PRINT *, 'THE SELECTED NUMBER OF HORSESHOE VORTICES HAVE'
  PRINT *, 'BEEN EVENLY SPACED ACROSS THE SEMISPAN AND THE'
  PRINT *, 'FIRST VORTEX IS NEAR THE WING TIP.'
  PRINT *, '
  PRINT *, 'AT WHICH HORSESHOE VORTEX WOULD YOU LIKE TO'
  PRINT *, 'SEE THE CHORDWISE DELTA CP DISTRIBUTION?'
  PRINT 922, VIC
REMEMBER THAT THE VORTICES ARE SPREAD EVENLY ACROSS THE SEMISSPAN AND THE 1ST VORTEX IS NEAR THE WING TIP.

PRINT *, 'GRAPHICS BEING CREATED'
CALL GRAPH6(NUMVOR)

C GET A HARDCOPY OF THIS GRAPHIC
CALL LIB$SPAWN('RENDER/DEVICE=LA210/DRAFT_QUALITY/PAPER_SIZE=A P6.UIS')
CALL LIB$SPAWN('CONTINUE')
PRINT *, 'WOULD YOU LIKE A PRINT OF THIS PLOT? (Y/N)'
READ 1002, PLOT6
IF (PLOT6.EQ. 'Y') THEN
    CALL LIB$SPAWN('PRINT P6. REN')
ENDIF
GO TO 41
ENDIF
ENDIF
END

OPTION TO MAKE ANOTHER RUN

PRINT *
PRINT *, 'DO YOU WISH TO :
PRINT *, '  1) MAKE ANOTHER RUN OR'
PRINT *, '  2) END THIS SESSION'
PRINT *, ' ENTER 1 OR 2.'
PRINT *
CALL QUERY (NANS)
CALL CLRSCRN
CLOSE (UNIT = 11)
CLOSE (UNIT = 12)
CLOSE (UNIT = 13)
IF (NANS .EQ. 1) GO TO 1

STOP

SUBROUTINE AERO

REAL MACH, UOU(300,2), VOU(300,2), YCP(2), YY(2), FU(2), FV(2),
1 XTLLEG(60), CHLPT(300,2), CLCC(300,2), YTLLEG(50), SDLT(50), CLA(2), SUM(2 2), AC(2), CH(2,50), CCAV(2,50), CLCL(2,50), CP(120), FW(2)
3, DIFCIRS(25), YLEGSV(25), ZLEGSV(25), CLPT(300,2), CLPB(300,2)
COMMON/ALL/ BOT, M, BETA, PTEST, QTEST, TBLSCW(50), Q(300), PN(300),
1 PV(300), ALP(300), S(300), PSI(300), PHI(300), ZH(50)
COMMON/TOTHRE/ CIR(300,2), SECTRST(50)
COMMON/ONETHRE/TWIST(2), CREF, SREF, CAVE, CLDES, STRUE, AR, ARTRUE,
1 RTCDHT(2), CONFIG, NSSWSV(2), MSV(2), KBOT, PLAN, IPLAN, MACH
2 , SSWA(50)
COMMON/PLT1/NSSW
COMMON/THRECDI/SLOAD(3,50)
COMMON/INSUB23/APS1, APHI , XX , YYY, ZZ , SNN, TOLCSQ
CHARACTER*8 HEAD
1 FORMAT (/ 12X, 'SECOND PLANFORM HORSESHOE VORTEX DESCRIPTIONS' /)
2 FORMAT ( / //58X,16HAERODYNAMIC DATA, //54X, 'CONFIGURATION
INO.', , F7.0 / )
3 FORMAT(///18X,'COMPLETE CONFIGURATION',31X,'WING-BODY CHARACTERIST
ICS',/ 64X, 'LIFT', 9X, 'INDUCED DRAG (FAR FIELD SOLUTION)',/ 2 16X, A8, 'CL COMPUTED ALPHA',19X, 'CL(WB)',7X, 'CDI AT CL(WB)',
3 4X,15HCDI/(CL(WB)**2),/ 88X,12H(1/(PI*AR) =,F8.5, ')
4 FORMAT (11X,2F15.5,15X,3F15.5)
5 FORMAT(///4X,11H REF. CHORD,6X,25HC AVERAGE TRUE WING AREA,3X,13
1HREF WING AREA,9X,3HB/2, 8X,7HREF. AR,8X,7TRUE AR,4X,11H MACH NUMB
2ER/) 8 FORMAT(8F15.5)
11 FORMAT (/// 47X, 'COMPLETE CONFIGURATION CHARACTERISTICS',/ 1 36X, 'CL ALPHA',8X, 'CL(TWIST) ALPHA AT CL=0 Y CP CM/CL
2 CMO', / 27X, 'PER RADIUS PER DEGREE', / 24X,7F12.5 )
12 FORMAT(///25X,'ADDITIONAL LOADING AT',/23X, 'L(TOTAL)/(Q**S(TRUE)) =
11.0',/57X, 'LOAD DUE ADD. LOAD AT BASIC LOAD',2X, 'SPAN LOAD AT
2 SL COEF FROM', '/ STATION',6X, '2Y/B',9X, 'S L COEF.',4X, 'CL RATIO'
3,4X, 'C RATIO',7X, 'TO TWIST CL=',F9.5,3X, 'AT CL=0',5X, 'DESIRED CL
4 CHORD BD VOR')
13 FORMAT (/// 47X, 'CONTRIBUTION OF THE SECOND PLANFORM TO SPAN LOAD D
1ISTRIBUTION' /)
15 FORMAT(4X, I4, F12.5, 5X, 3F12.5, 3X, 3F12.5, 3X, 2F12.5)
16 FORMAT (1H)
18 FORMAT(///55X,21HTHIS CASE IS FINISHED)
20 FORMAT(///5X,'DELTA CP TERMS FROM LE TIP TO TE TIP THEN INBOARD
1 ENDING WITH THE TE OF ROOT CHORD')
21 FORMAT (///54X, 'CMQ AND CLQ ARE COMPUTED')
22 FORMAT(///38X, 'STATIC LONGITUDINAL AERODYNAMIC COEFFICIENTS ARE COMPUT
1UTED')
23 FORMAT (/// 59X, 'CLP IS COMPUTED')
24 FORMAT(8F15.5)
25 FORMAT (/20X,'X',11X,'X',11X,'Y',11X,'Z',12X,'S',5X,'C/4 SWEEP',4X
1,'DIHEDRAL',2X, 'LOCAL ALPHA',2X, 'DELTA CP AT DESIRED /
2 19X,'C/4',9X, '3C/4',42X, 'ANGLE',7X, 'ANGLE',4X, 'IN RADIANS',4X,
3 'CL =',F10.5 /)
303 FORMAT(12X,9F12.5)
1013 FORMAT(///47X, 'CONTRIBUTION OF THE SECOND PLANFORM TO THE CHORD OR D
1RAG FORCE')
1070 FORMAT (/// 30X, 'INDUCED DRAG, LEADING EDGE THRUST AND SUCTION
1 COEFFICIENT CHARACTERISTICS',/ 2 34X, 'COMPUTED AT ONE RADIANS OF ATTACK FROM A NEAR FIELD S
UTION',/// 58X, 'SECTION COEFFICIENTS',12X, 'CONTRIBUTIONS TO TOTAL COEF.' ,/ 5 92X, 'FROM EACH SPANWISE ROW' ,/ 6 38X, 'L. E. SWEEP' ,/
7 15X, 'STATION',9X, '2Y/B',5X, 'ANGLE',5X, 'CDII C/2B',5X, 'CT C/2B' ,
PART 3 - COMPUTE OUTPUT TERMS

RAD = 57.29578
TWST = TWIST(1) + TWIST(2)
ALREF = 1

THE TOLERANCE SET AT THIS POINT IN THE PROGRAM MAY NEED TO BE
CHANGED FOR COMPUTERS OTHER THAN THE CDC 6000 SERIES

TOLC = .0100*BOT
TOLCSQ = TOLC**2
QINF = 1.
NSSW = NSSWSV(1) + NSSWSV(2)
IF(RTCDHT(1).NE.RTCDHT(2)) GO TO 794
SUMPHI = 0
DO 801 J=1,NSSW
801 SUMPHI = SUMPHI + ABS(PHI(J))
IF(SUMPHI.EQ.0.) GO TO 921

PART 3 - SECTION 1
COMPUTE LIFT AND PITCHING MOMENT FOR WINGS WITH DIHEDRAL

GEOMETRY FOR TIP TRAILING LEGS

794 CPM(1) = 0
CPM(2) = 0
YCP(1) = 0
YCP(2) = 0
IM = 0
CLT = 0
CLNT = 0
NSSW1 = 0
NSSW2 = NSSWSV(1)
NSSW3 = NSSWSV(1)
L = 1
NSCW = MSV(1) / NSSWSV(1)
GO TO 798
796 NSSW1 = NSSWSV(1)
NSSW2 = NSSW
NSSW3 = NSSWSV(2)
L=NSSWSV(1)+1
NSCW = MSV(2) / NSSWSV(2)
798 I = IM + 1
J = IM + 2
IUU=2
DIFFCR1=0.
APHI=ATAN(PHI(I))
TLX1=PN(I)-S(I)*TAN(PSI(I))
TLX2=PN(J)-S(J)*TAN(PSI(J))
CLFTLG=TLX1-TLX2
XTLEG(1)=TLX1/2.+TLX2/2.
YLEG=Q(I)-S(I)*COS(APHI)
IF(NSSW1.EQ.0) YLEGSV( 1)=YLEG
ZLEG=ZH(I)-S(I)*SIN(APHI)
IF(NSSW1.EQ.0) ZLEGSV( 1)=ZLEG
IF(NSSW1.EQ.NSSWSV(1)) GO TO 850
GO TO 852
850 DO 5050 IT=1,L
IF((ABS(YLEGSV(IT)-YLEG).LT.TOLC).AND.(ABS(ZLEGSV(IT)-ZLEG).LT.TOLC)) DIFFCR1=DIFCIRS(IT)
5050 CONTINUE
852 DO 802 NV=2,NSCW
NVT=NV-1
802 XTLEG(NV)=XTLEG(NVT)-CLFTLG
NCTL=0
NA =1
NB =NSCW
803 DO 823 NV=NA,NB
VOU(NV,1)= 0
VOU(NV,2)= 0
UOU(NV,1)= 0
UOU(NV,2)= 0.
DO 809 NN=1,M
IZ=(NN-1)/NSCW+1
APHI=ATAN(PHI(IZ))
APSI=PSI(NN)
XX=XTLEG(NV)-PN(NN)
YY(1)=YLEG-Q(NN)
YY(2)=YLEG+Q(NN)
ZZ=ZLEG-ZH(IZ)
SNN = S(NN)
C
DO 822 I=1,2
YYY = YY(I)
CALL INFSUB (BOT,FU(I),FV(I),FW(I) )
APHI=-APHI
APSI=-APSI
822 CONTINUE
C
9001 DO 809 IXX=1,2
UOU(NV,IXX)=UOU(NV,IXX)+((FU(1)+FU(2))*CIR(NN,IXX))/12.566371
809 VOU(NV,IXX)=VOU(NV,IXX)+((FV(1)+FV(2))*CIR(NN,IXX))/12.566371
823 CONTINUE
NCTL=NCTL+1
IF (NCTL-2) 810,811,812
C
C GEOMETRY FOR SPANWISE BOUND VORTICES

810 NA=NSCW+1
NB=2*NSCW
JA=IM*NSCW+1
YLEG=Q(JA)
ZLEG=ZH(IM+1)
DO 818 J=1,NSCW
   JK=IM*NSCW+J
   NV=J+NSCW
818 XTLLEG(NV)=PN(JK)
   GO TO 803

C GEOMETRY ALONG RIGHT TRAILING LEGS

811 NA=2*NSCW+1
NB=3*NSCW
DIFFCR2=0.
JK=IM*NSCW+1
APHI=ATAN(PHI(IM+1))
YLEG=Q(JK)+S(JK)*COS(APHI)
IF(NSSW1.EQ.0) YLEGSV(IUU)=YLEG
ZLEG=ZH(IM+1)+S(JK)*SIN(APHI)
IF(NSSW1.EQ.0) ZLEGSV(IUU)=ZLEG
TLX1=PN(JK)+S(JK)*TAN(PSI(JK))
   JK=JK+1
TLX2=PN(JK)+S(JK)*TAN(PSI(JK))
CRTTLG=TLX1-TLX2
XTLEG(NA)=TLX1/2.+TLX2/2.
NAA=NA+1
IF(NSSW1.EQ.NSSWSV(1)) GO TO 851
   GO TO 853
851 DO 5051 IT=1,L
   IF(ABS(YLEGSV(IT)-YLEG).LT.TOLC).AND.(ABS(ZLEGSV(IT)-ZLEG).LT.TOLC) DIFFCR2=DIFFCRS(IT)
5051 CONTINUE
853 DO 819 NV=NAA,NB
   NVT=NV-1
819 XTLLEG(NV)=XTLEG(NVT)-CRTTLG
   GO TO 803

C COMPUTE LIFT AND PITCHING MOMENT FOR EACH ELEMENTAL PANEL

812 YY(1)=0
   YY(2)=0
   IF ( IM.NE.NSSW1 ) GO TO 834
   DO 835 IXX=1,2
      DIFFCIR=DIFFCR1
      DO 835 NPOS=1,NSCW
         DIFFCIR=DIFFCIR+CIR(NPOS,IXX)
      CON=1.
      MORT = MORT + 1
      IF (NPOS.EQ.NSCW) CON=.75
      CHLFT(NPOS,IXX)=CHLFTLG*CON*DIFFCIR*VOU(NPOS,IXX)*(2./SREF)
      CLPT(NPOS,IXX)=CHLFT(NPOS,IXX)*(Q(NPOS)-S(NPOS))*(2.)
5051 CONTINUE

835 CONTINUE
IF(NSSW1.EQ.0) DIFCIRS(1)=DIFCIR

834 DO 815 IXX=1,2
    DIFCIR=DIFFCR2
    DO 815 NPOS=1,NSCW
    JK=IM*NSCW+NPOS
    JL=(IM+1)*NSCW+NPOS
    JM=NSCW+NPOS
    JN=2*NSCW+NPOS
    IF (IM.EQ.(NSSW2-1)) GO TO 836
    DIFCIR=DIFCIR+CIR(JL,IXX)-CIR(JK,IXX)
836 CON=1.
    IF (NPOS.EQ.NSCW) CON=.75
    CHLFT(JL,IXX)=CRTTLG*CON*DIFCIR*V0U(JN,IXX)*(2./SREF)
    CLCC(JK,IXX)=(2./SREF)*CIR(JK,IXX)*2.*S(JK)*COS(APHI)*((1.-V0U(JM,1IXX)+V0U(JM,IXX)*TAN(PSI(JK)))
    CLPB(JK,IXX)=CLCC(JK,IXX)*Q(JK)*2.
    CLPT(JL,IXX)=CHLFT(JL,IXX)*Q(JK)+S(JK)*2.
    CPM(IXX)=CPM(IXX)+(CLCC(JK,IXX)*XTLEG(JM)*BETA+CHLFT(JK,IXX)*XTLEG1(NPOS)*BETA)*2./SREF
    YCP(IXX)=YCP(IXX)+(CLCC(JK,IXX)*Q(JK)+CHLFT(JK,IXX)*(Q(JK)-S(JK)*1COS(APHI)))/BOT
815 CONTINUE
    IF(NSSW1.EQ.0) DIFCIRS(IUU)=DIFCIR
    CLT=CLT+YY(1)
    CLNT=CLNT+YY(2)
    IM=IM+1
    IF(NSSW1.EQ.0) IUU=IM+2
    IF(IM.EQ.NSSWSV(1)) CLWNGT=CLT
    IF(IM.EQ.NSSWSV(1)) CLWING=CLNT
    IF(IM.GE.NSSW2) GO TO 816
    NCTL=1
    DO 817 IXX=1,2
        DO 817 NV=1,NSCW
            NY=NV+2*NSCW
            XTLEG(NV)=XTLEG(NY)
817 VOU(NV,IXX)=VOU(NY,IXX)
    GO TO 810
C
     SUM LIFT AND PITCHING MOMENT FOR ENTIRE WING
C
816 YY(1)=CLT*SREF/STRUE
    YY(2)=CLNT*SREF/STRUE
    NUP=NSSW3+1
    YTLEG(NUP)=0.
    XTLEG(NUP)=0
    IND=1
    IF (TWST.EQ.0.) IND=2
    DO 837 IXX=IND,2
        DO 837 JSSW=IND,NSSW2
            SLOAD(IXX,JSSW)=0
            SLDT(JSSW)=0
            APHI=ATAN(PHI(JSSW))
            JL=(JSSW-1)*NSCW+1
            K=JSSW-L+1
    820 YTLEG(K)=Q(JL)-S(JL)*COS(APHI)
DO 837 INC=1,NSCW
DO 838 JNS=L,NSSW2
JK=(JNS-1)*NSCW+INC
K=JNS-L+1

838 XTLEG( K )=CHLFT(JK,IXX)
DO 837 INS=L,NSSW2
JK=(INS-1)*NSCW+INC
APHI=ATAN(PHI(INS))
CALL FTLUP( Q(JK),CHTLF,+1,NUP,YTLEG,XTLEG)
T= SREF/(2.*S(JK)*COS(APHI)*CAVE)
SLDT(INS)=SLDT(INS)+CHTLF*T
CLCC(JK,IXX) = (CLCC(JK,IXX) + CHTLF ) * T

837 SLOAD(IXX,INS)=SLOAD(IXX,INS)+ CLCC(JK,IXX)
IF (IM.NE.NSSW) GO TO 796
CLA(2)=CLNT /ALREF
CMCL=CPM(2)/CLNT
CMO=CPM(1)-CMCL*CLT
YCP(2)=YCP(2)/(CLNT/2.)
DO 840 I=1,NSSW
SLDT(I)=SLDT(I)/YY(2)
IF (TWST .EQ. 0.) SLOAD(I,1)=0.
IF (TWST .NE.0.) SLOAD(I,1)=SLOAD(I,1)/YY(1)

840 SLOAD(2,I) = SLOAD(2,I)/YY(2)
CRL=0.
DO 860 IAM=1,M
860 CRL=CRL+CLPB(IAM,2)+CLPT(IAM,2)
CLP=CRL/(.08725*2.*BOT)
GO TO 903

C
C PART 3 - SECTION 2
C COMPUTE LIFT AND PITCHING MOMENT FOR WINGS WITHOUT DIHEDRAL
C
921 DO 901 NV=1,2
SUM(NV)=0
DO 901 I=1,M
SUM(NV)=SUM(NV)+CIR(I,NV)*S(I)
IF (NV.EQ.1.AND.I.EQ.MSV(1)) CLWNGT = SUM(1)*8. / SREF
IF (NV.EQ.2.AND.I.EQ.MSV(1)) CLWING = SUM(2)*8. / SREF
901 CONTINUE
CLT = 8.* SUM(1)/SREF
CLNT = 8.* SUM(2)/SREF
IF (KBOT.EQ.1) GO TO 800
CLWNGT = CLT - CLWNGT
CLWING = CLNT- CLWING

800 CRL = 0.
DO 905 I=1,M
CRL=CRL+(Q(I)*CIR(I,2)*2.*S(I))*2.
CLCC(I,1)=CIR(I,1)*2./CAVE
905 CLCC(I,2)=CIR(I,2)*2./CAVE

C
C COMPUTE CLP
C
CLP= CRL/(SREF*BOT*0.08725)
CLA(2)=CLNT
DO 922 IXX=1,2
SA = 0
SB = 0
SC = 0.
I = 0
DO 920 JSSW=1,NSSW
SLDT(JSSW)=0
SLOAD(IXX,JSSW)=0
NSCW = TBLSCW(JSSW)
DO 920 JSCW=1,NSCW
IF(TWST .EQ. 0. .AND. IXX.EQ.1) GO TO 930
I = I + 1
SA=SA+CIR(I,IXX)*S(I)
SB=SB+CIR(I,IXX)*Q(I)*S(I)
SC=SC+CIR(I,IXX)*PN(I)*S(I)*BETA
SLOAD(IXX,JSSW) = SLOAD(IXX,JSSW)+(BOT*CIR(I,IXX))/(2.*SUM(IXX))
GO TO 920
930 SLOAD(1,JSSW)=0.
920 CONTINUE
IF(TWST .EQ. 0. .AND. IXX.EQ.1) GO TO 932
YCP(IXX)=SB/(SA*BOT)
AC(IXX)=SC/(SA*CREF)
GO TO 922
932 YCP(1)=0
AC(1)=0.
922 CONTINUE
CMCL=AC(2)
CMO=(AC(1)-AC(2))*CLT

C PART 3 - SECTION 3
C COMPUTE AND PRINT FINAL OUTPUT DATA FOR ALL WINGS
C

903 DO 902 IXX=1,2
JN = 0
DO 902 JSSW=1,NSSW
CH (IXX,JSSW)=0
NSCW = TBLSCW(JSSW)
DO 904 JSCW=1,NSCW
JN = JN + 1
CH (IXX,JSSW)=(-2.0)*(PV(JN)-PN(JN))*BETA+CH (IXX,JSSW)
904 CONTINUE
CCAV(IXX,JSSW)=CH(IXX,JSSW)/CAVE
CLCL(IXX,JSSW)=SLOAD(IXX,JSSW)/CCAV(IXX,JSSW)
902 CONTINUE
CLD=CLDES
IF(CLDES.EQ.11) CLD=1.
DO 1020 I=1,M
CP(I) = (CLCC(I,1)+CLCC(I,2)*(CLD-CLT)/CLNT)*CAVE/(2.*(PN(I)-1)
PV(I) ) * BETA )
1020 CONTINUE
WRITE (29,4) CONFIG
IF ( PTEST.NE.0. ) WRITE (29,23)
IF ( QTEST.NE.0. ) WRITE (29,21)
IF ( PTEST.EQ.0. .AND. QTEST.EQ.0. ) WRITE (29,22)
WRITE(29,25) CLD
HEAD = ' DESIRED'
IF (CLDES.EQ.11. ) HEAD = ' IEND =11

189
IF(CLDES.NE.11.) IEND=1
DO 5000 IUTK=1,IEND
IF(IEND.EQ.11.) CLDES=(FLOAT(IUTK)-1.)/10.
IF(CLDES.EQ.0.) CLDES=-.1
NR = 0
MORT = MORT + 1
DO 3006 NV=1,NSSW
NSCW = TBLSCW(NV)
NP = NR + 1
NR = NR + NSCW
PHIPR = ATAN(PHI(NV)) * RAD
SLOAD(3,NV)=0.
IF (NV.EQ.(NSSWSV(1)+1).AND. IUTK.EQ.1) WRITE (29,1)
METH = METH + 1
DO 3006 I=NP,NR
IF (IUTK.GT.1) GO TO 3006
PNPR = PN(I) * BETA
PVPR = PV(I) * BETA
PSIPR = PSI(I)* RAD
WRITE(29,303) PNPR,PVPR,Q(I),ZH(NV),S(I),PSIPR,PHIPR,ALP(I),CP(I)
WRITE(13,603) METH,MORT,PNPR,PVPR,Q(I),CP(I)
WRITE(29,303)
C
SLOAD(3,NV)=SLOAD(3,NV)+CLCC(I,2)*CLDES/CLNT+CLCC(I,1)-CLCC(I,2)*C
C
IF(IUTK.GT.1) GO TO 3007
WRITE(29,7)
WRITE(29,8) CREF,CAVE,STRUE,SREF,BOT,AR,ARTRUE,MACH
C
CONTINUE
C
IF(PTEST.NE.0.)WRITE(29,4445) CLP
IF(PTEST.NE.0.) GO TO 4444
C
COMPUTE CMQ,CLQ
C
CMQ=2.*CMCL*CLNT/(0.08725*CREF)
CLQ=2.*CLNT/(0.08725*CREF)
IF(QTEST.NE.0.) WRITE(29,4446) CMQ,CLQ
IF(QTEST.NE.0.) GO TO 4444
C
COMPUTE INDUCED DRAG
C
NSV=NSSWSV(1)+1
MTOT=MSV(1)+1
IF(KBOT.EQ.1) GO TO 1001
NSV=NSV+NSSWSV(2)
MTOT=MTOT+MSV(2)
1001 CALL CDICLS (AR,ARTRUE,NSSWSV(KBOT),MTOT,NSV,CDI,CDIT)
CLAPD=CLA(2)/57.29578
ALPO=-(CLT/CLA(2))*57.29578
ALPD=CLDES/CLAPD+ALPO
ALPW=1./CLAPD
CLWB=CLWING*ALPD/57.29578+CLWNGT
CDIWB = CDI /(CLWB*CLWB)

190
IF (IUTK.EQ. 1) WRITE (29, 5) HEAD, CDIT

5000 WRITE (29, 6) CLDES, ALPD, CLWB, CDI, CDIWB
WRITE(29, 11) CLA(2), CLAPD, CLT, ALPO, YCP(2), CMCL, CMO
WRITE(29, 12) CLT

NR = 0
J = 0
DO 1004 NV=1, NSSW
BCLCC = 0
BADLAE= 0
BASLD = 0.
NSCW = TBLSCW(NV)
NP = NR + 1
NR = NR + NSCW
DO 1002 I=NP,NR
ADLAE=CLCC(I, 2)*CLT/CTLN
BSLD=CLCC(I, 1)-ADLAE
BCLCC=BCLCC+CLCC(1, 1)
BADLAE=BADLAE+ADLAE
BASLD=BASLD+BSLD
JMETH=METH+1
1002 CONTINUE
J = J + NSCW
YQ = Q(J) / BOT
IF (NV.EQ.(NSSWSV(1)+1)) WRITE(29, 13)
WRITE(12, 15) NV,-(YQ),SLOAD(2,NV), CLCL(2,NV), CCAV(2,NV),
+ BCLCC, BADLAE, BASLD, SLOAD(3, NV), SLDT(NV)
1004 WRITE(29, 15) NV,YQ, SLOAD(2, NV), CLCL(2, NV), CCAV(2, NV), BCLCC, BADLAE, 1
BASLD, SLOAD(3, NV), SLDT(NV)
WRITE (29, 1070)
CTHRUST = 0
CSUCT = 0.
CDRAG = 0.
NN = 1
DO 1050 NV=1, NSSW
SSCTRST = SECTRST(NV) / (4. * BOT)
SSCDRAG = SLOAD (2, NV) * CAVE * SREF * CLA(2) / (STRUE * 4. * BOT)
1 - SSCTRST
CSSWWA = COS (ATAN (SSWWA(NV)))
SSCSUCT = SSCTRST / CSSWWA
IF (NV.EQ. 1) GO TO 1060
NN = NN + TBLSCW(NV - 1)
1050 PHIPR = ATAN (PHI(NV))
CDRAGS = SSCDRAG*4. *BOT*2. *S(NN)*COS(PHIPR)/SREF
CDRAG = CDRAG + 2.0 * CDRAGS
CTHRUSS = SECTRST(NV)*2. *S(NN)*COS(PHIPR) / SREF
CTHRUST = CTHRUST + 2.0 * CTHRUSS
CSUCTS = CTHRUST / CSSWWA
CSUCT = CSUCT + 2.0 * CSUCTS
SWALE = ATAN(SSWWA(NV)) / RAD
YQ = Q(NN) / BOT
IF(NV.EQ.(NSSWSV(1)+1)) WRITE(29, 1013)
WRITE(11, 1071) NV, YQ, SWALE, SSCDRAG, SSCTRST, SSCSUCT,
+ CDRAGS, CTHRUSS, CSUCTS
1060 WRITE(29, 1071) NV, YQ, SWALE, SSCDRAG, SSCTRST, SSCSUCT, CDRAGS, CTHRUSS, 1
CSUCTS
CDRAGP = CDRAG / (CLA(2)*CLA(2))
WRITE(29,1072) CDRAGP, CTHRUST, CSUCT

4444 WRITE(29,18)
METH = 99
MORT = 0
PNPR = 0.00
PVPR = 0.00
Q(NR+1) = 0.00
CP(NR+1) = 0.00
WRITE(13,603) METH, MORT, PNPR, PVPR, Q(NR+1), CP(NR+1)
CLOSE(UNIT = 11)
CLOSE(UNIT = 12)
CLOSE(UNIT = 13)
WRITE(29,16)
RETURN
END

SUBROUTINE AMATINV(A,N,B,M,DETERM, IPIVOT, INDEX, NMAX, ISCALE)

C********** DOCUMENT DATE 08-01-68 SUBROUTINE REVISED 08-01-68 *********
C
C MATRIX INVERSION WITH ACCOMPANYING SOLUTION OF LINEAR EQUATIONS
C
DIMENSION IPIVOT(N), A(NMAX,N), B(NMAX,M), INDEX(NMAX,2)
EQUIVALENCE (IROW, JROW), (ICOLUM, JCOLUMN), (AMAX, T, SWAP)

C
C INITIALIZATION
C
5 ISCALE=0
6 R1=10.0**35
7 R2=1.0/R1
10 DETERM=1.0
15 DO 20 J=1,N
20 IPIVOT(J)=0
30 DO 550 I=1,N

C
C SEARCH FOR PIVOT ELEMENT
C
40 AMAX=0.0
45 DO 105 J=1,N
50 IF (IPIVOT(J)-1) 60, 105, 60
60 DO 100 K=1,N
70 IF (IPIVOT(K)-1) 80, 100, 740
80 IF (ABS(AMAX)-ABS(A(J,K))) 85, 100, 100
85 IROW=J
90 ICOLUMN=K
95 AMAX=A(J,K)
100 CONTINUE
105 CONTINUE
IF (AMAX) 110, 106, 110
106 DETERM=0.0
ISCALE=0
GO TO 740
110 IPIVOT(ICOLUMN)=IPIVOT(ICOLUMN)+1

C
INTERCHANGE ROWS TO PUT PIVOT ELEMENT ON DIAGONAL

130 IF (IROW-ICOLUM) 140, 260, 140
140 DETERM=-DETERM
150 DO 200 L=1,N
160 SWAP=A(IROW,L)
170 A(IROW,L)=A(ICOLUMN,L)
200 A(ICOLUMN,L)=SWAP
205 IF(M) 260, 260, 210
210 DO 250 L=1,M
220 SWAP=B(IROW,L)
230 B(IROW,L)=B(ICOLUMN,L)
250 B(ICOLUMN,L)=SWAP
260 INDEX(I,1)=IROW
270 INDEX(I,2)=ICOLUM
310 PIVOT=A(ICOLUMN,ICOLUMN)
    IF (PIVOT) 1000,106,1000

SCALE THE DETERMINANT

1000 PIVOTI=PIVOT
1005 IF(ABS(DETERM)-R1)1030,1010,1010
1010 DETERM=DETERM/R1
    ISCALE=ISCALE+1
1020 DETERM=DETERM/R1
    ISCALE=ISCALE+1
    GO TO 1060
1030 IF(ABS(DETERM)-R2)1040,1040,1060
1040 DETERM=DETERM*R1
    ISCALE=ISCALE-1
1050 DETERM=DETERM*R1
    ISCALE=ISCALE-1
1060 IF(ABS(PIVOTI)-R1)1090,1070,1080
1070 PIVOTI=PIVOTI/R1
    ISCALE=ISCALE+1
1080 PIVOTI=PIVOTI/R1
    ISCALE=ISCALE+1
    GO TO 320
1090 IF(ABS(PIVOTI)-R2)2000,2000,320
2000 PIVOTI=PIVOTI*R1
    ISCALE=ISCALE-1
2010 PIVOTI=PIVOTI*R1
    ISCALE=ISCALE-1
320 DETERM=DETERM*PIVOTI

DIVIDE PIVOT ROW BY PIVOT ELEMENT

330 A(ICOLUMN,ICOLUMN)=1.0
340 DO 350 L=1,N
350 A(ICOLUMN,L)=A(ICOLUMN,L)/PIVOT
355 IF(M) 380, 380, 360
360 DO 370 L=1,M
370 B(ICOLUM,L) = B(ICOLUM,L) / PIVOT

C REDUCE NON-PIVOT ROWS

380 DO 550 L1 = 1, N
390 IF(L1 - ICOLUM) 400, 550, 400
400 T = A(L1, ICOLUM)
410 A(L1, ICOLUM) = 0.0
420 DO 450 L = 1, N
430 A(L1, L) = A(L1, L) - A(ICOLUM, L) * T
440 CONTINUE
450 CONTINUE

C INTERCHANGE COLUMNS

500 DO 710 I = 1, N
510 L = N + 1 - I
520 IF (INDEX(L, 1) - INDEX(L, 2)) 630, 710, 630
530 JR0W = INDEX(L, 1)
540 JCOLUMN = INDEX(L, 2)
550 DO 705 K = 1, N
560 SWAP = A(K, JROW)
570 A(K, JROW) = A(K, JCOLUMN)
580 A(K, JCOLUMN) = SWAP
590 CONTINUE
600 CONTINUE
610 CONTINUE
620 CONTINUE
630 CONTINUE
640 CONTINUE
650 CONTINUE
660 CONTINUE
670 CONTINUE
680 CONTINUE

SUBROUTINE CDICLS (AR, ARTRUE, ISEMSP, MTOT, NSV, CDI, CDIT)

DIMENSION ETAN(51), GAMPR(51, 1), ETA(41), GAMMA(41), VE(41), B(41),
1 FN(41, 41)
COMMON/ALL/ BOT, M, BETA, PTEST, QTEST, TBLSCW(50), Q(300), PN(300),
1 CASEFN, PV(300), ALP(300), S(300), PSI(300), PHI(300), ZH(50)
COMMON/THRECDI/SLOAD(3, 50)
CHARACTER*20 CASEFN
DO 15 I = 1, 41
DO 15 J = 1, 41
15 FN(I, J) = 0
SPAN = 2. * BOT
CAVB = SPAN / ARTRUE
PI = 3.14159265E+01
NST = ISEMSP + 1
NN = MTOT
DO 101 N = 1, ISEMSP
NM = NSV - N
NSCW = TBLSCW(NM)
NN = NN - NSCW
ETAN(N) = ASIN(-Q(NN) * 2. / SPAN)
GAMPR(N, 1) = SLOAD(3, NM) * CAVB / (2. * SPAN)
101 CONTINUE
ETAN(NST) = PI / 2.
GAMPR(NST, 1) = 0
DO 7 NP= 1,41
ANP=NP
7 ETA(NP)= (ANP-21.)*PI/42.

C

DO 102 JK=21,41
CALL FTLUP(ETA(JK),GAMMA(JK),1,NST,ETAN,GAMPR)
102 CONTINUE

DO 600 NY=22,41
ETA(NY)=SIN(ETA(NY))
NR=42-NY
ETA(NR)=-ETA(NY)
600 GAMMA(NR)=GAMMA(NY)

DO 589 NU=21,41
ANU=NU
DO 14 N=1,41
AN=N
NNUD=IABS(N-NU)
VE(N)=COS(((AN-21.)*PI)/42.)
IF(NNUD.NE.0) GO TO 9
B(N)=(42.)/(4.*COS(((ANU-21.)*PI)/42.))
GO TO 14
9 IF(MOD(NNUD,2).EQ.0) GO TO 12
B(N)=VE(N)/((42.)*(ETA(N)-ETA(NU))*2)
GO TO 14
12 B(N)=0.0
14 CONTINUE

DO 589 NP=21,41
NUST =IABS(NU-21)
IF(NUST.EQ.0) GO TO 589
IF(MOD(NUST,2).EQ.0) GO TO 589
NPST=IABS(NP-20)
IF(MOD(NPST,2).EQ.0) GO TO 589
NPNUD=IABS(NP-NU)
IF(NPNUD.EQ.0) GO TO 589
IF(MOD(NPNUD,2).EQ.0) GO TO 589
FVN(NU,NP)=2.0*B(NP)/21.*COS((ANU-21.)*PI/42.)
IT=42-NU
ITT=42-NP
FVN(NU,ITT)=2.0*B(ITT)/21.*COS((ANU-21.)*PI/42.)
FVN(IT,NP)=FVN(NU,ITT)
FVN(IT,ITT)=FVN(NU,NP)
589 CONTINUE

C
CCC=0.0
DO 10 N=1,41
10 CCC=CCC+(GAMMA(N)*GAMMA(N))
CCD=0.0
DO 11 NUP=1,41
11 CCC=CCC+GAMMA(N)*GAMMA(N))
CCD=CCD-2.0*FVN(NUP,N)*(GAMMA(NUP)*GAMMA(N))
11 CONTINUE
CDI=PI*AR/4.*(CCC+CCD)
CDIT=1./(PI*AR)
RETURN
END
SUBROUTINE CLRSCRN
C
C LIBRARY ROUTINE TO CLEAR THE SCREEN.
C
ISTAT = LIB$ERASE_PAGE (1,1)
RETURN
END

SUBROUTINE FTLUP (X,Y,M,N,VARI,VARD)
C
C********** DOCUMENT DATE 09-12-69 SUBROUTINE REVISED 07-07-69 **********
C* MODIFICATION OF LIBRARY INTERPOLATION SUBROUTINE FTLUP
DIMENSION VARI(1),VARD(1),V(3),YY(2)
DIMENSION II(43)
C*
C* INITIALIZE ALL INTERVAL POINTERS TO -1.0 FOR MONOTONICITY CHECK
DATA (II(J),J=1,43)/43*-1/
MA=IABS(M)
C*
C* ASSIGN INTERVAL POINTER FOR GIVEN VARI TABLE
C* THE SAME POINTER WILL BE USED ON A GIVEN VARI TABLE EVERY TIME
LI = 1
I=II(LI)
IF (I.GE.0) GO TO 10
IF (N.LT.2) GO TO 10
C*
C*MONOTONICITY CHECK
IF (VARI(2)-VARI(1)) 1,1,3
C* ERROR IN MONOTONICITY
2 K = 1
WRITE(29,102)J,K,(VARI(J),J=1,N),(VARD(J),J=1,N)
102 FORMAT (1H1,' TABLE BELOW OUT OF ORDER FOR FTLUP AT POSITION ',1,I5,' X TABLE IS STORED IN LOCATION ',06,//(8G15.8))
STOP
C* MONOTONIC DECREASING
1 DO 5 J=2,N
5 CONTINUE
GO TO 10
C* MONOTONIC INCREASING
3 DO 6 J=2,N
6 CONTINUE
C*INTERPOLATION
10 IF (I.LE.0) I=1
IF (I.GE.N) I=N-1
IF (N.LE.1) GO TO 8
IF (MA.NE.0) GO TO 99
C* ZERO ORDER
8 Y=VARD(1)
GO TO 800
C* LOCATE I INTERVAL (X(I).LE.X.LT.X(I+1))
99 IF (((VARI(I)-X)*(VARI(I+1)-X)) 61,61,40
C* IN GIVES DIRECTION FOR SEARCH OF INTERVALS
196
IN=SIGN(1.0,(VARI(I+1)-VARI(I))*(X-VARI(I)))

IF X OUTSIDE ENDPOINTS, EXTRAPOLATE FROM END INTERVAL

IF ((I+IN).LE.0) GO TO 61
IF ((I+IN).GE.N) GO TO 61
I=I+IN
IF ((VARI(I)-X)*(VARI(I+1)-X)) 61,61,41

IF (MA.EQ.2) GO TO 200

FIRST ORDER

Y=(VARD(I)*(VARI(I+1)-X)-VARD(I+1)*(VARI(I)-X))/(VARI(I+1)-VARI(I))

GO TO 800

SECOND ORDER

200 IF (N.EQ.2) GO TO 2
IF (I.EQ.(N-1)) GO TO 209
IF (I.EQ.1) GO TO 201

PICK THIRD POINT

SK= VARI(I+1)-VARI(I)
IF (((SK*(X-VARI(I-1))).LT.(SK*(VARI(I+2)-X))) GO TO 209

L=I
GO TO 702

L=I-1

702 V(1)=VARI(L)-X
V(2)=VARI(L+1)-X
V(3)=VARI(L+2)-X
YY(1)=(VARD(L)*V(2)-VARD(L+1)*V(1))/(VARI(L+1)-VARI(L))
YY(2)=(VARD(L+1)*V(3)-VARD(L+2)*V(2))/(VARI(L+2)-VARI(L+1))
Y=(YY(1)*V(3)-YY(2)*V(1))/(VARI(L+2)-VARI(L))

II(LI)=I
RETURN
END

SUBROUTINE GEOM

DIMENSION XREF(25),YREF(25),SAR(25),A(25),RSAR(25),X(25),Y(25),
1 BOTS(2),SA(2),VBORD(51),SPY(50,2),KFX(2),IYL(50,2),
2 IYT(50,2)
COMMON/SHIP/VIC,SCW
COMMON/ALL/ BOT,M,BETA,PTEST,QTEST,TBLSCW(50),Q(300),PN(300),
1 PV(300),ALP(300),S(300),PSI(300),PHI(300),ZH(50)
COMMON/ONETHRE/TWIST(2),CREF,SREF,CAVE,CLDES,STRUE,AR,ARTRUE,
1 RTCDHT(2),CONFIG,NSSWSV(2),MSV(2),KBOT,PLAN,IPLAN,MACH
2 ,SSWVA(50)
COMMON/MAINONE/ICODEOF,TOTAL,AAN(2),XS(2),YS(2),KFCTS(2)
1 ,XREG(25,2),YREG(25,2),AREG(25,2),DIH(25,2),MCD(25,2)
2 ,XX (25,2),YY (25,2),AS (25,2),TTWD(25,2),MMCD(25,2)
3 ,AN(2),ZZ (25,2)
REAL MACH
CHARACTER*10 PRTCON

1 FORMAT ( /63X,'GEOMETRY DATA' )
2 FORMAT (///45X,A10,'REFERENCE PLANFORM HAS',I3,'CURVES',//
1 12X,'ROOT CHORD HEIGHT =',F12.5,4X,'VARIABLE SWEEP
2 PIVOT POSITION',4X,'X(S) =',F12.5,5X,'Y(S) =',F12.5,46X,
3 'BREAK POINTS FOR THE REFERENCE PLANFORM' / )
3 FORMAT (8F10.4)
4 FORMAT (8F15.5)
5 FORMAT ( // 47X, 'CONFIGURATION NO.', F8.0 / )
6 FORMAT (2F12.5, 2E12.5, F12.5)
7 FORMAT ( // 36X, I4, 44H HORSESHOE VORTICES ON LEFT HALF OF THE WING/36X, I4, 10H CHORDWISE, 21X, I4, 4.9H SPANWISE// )
9 FORMAT (20X, I5, 3F12.5, 2F14.5, 16)
10 FORMAT ( //40X, 'CURVE', 13, 'IS SWEEPED', F12.5, 'DEGREES ON PLANFORM', F7.0)
11 FORMAT ( ///41X, *END OF FILE ENCOUNTERED AFTER CONFIGURATION' F7.0)
12 FORMAT ( ///18X, 'THE FIRST VARIABLE SWEEP CURVE SPECIFIED (K =', I3, ' AND QTEST =', F5.1, ' ) DOES NOT HAVE AN M CODE OF 2 FOR PLANFORM', I4)
13 FORMAT (26X, I5, 2F12.5, 2F16.5, 4X, I4)
14 FORMAT ( ///10X, 'ERROR - PROGRAM CANNOT PROCESS PTEST =', F5.1, ' AND QTEST =', F5.1 )
15 FORMAT ( ///10X, 'SECOND PLANFORM BREAK POINTS' / )
16 FORMAT ( ///25X, 'BREAKPOINT LOCATED SPANWISE AT', F11.5, 3X, 20HH1AS BEEN ADJUSTED TO', F9.5 /// )
17 FORMAT ( ///27X, 'TABLE OF HORSESHOES IN EACH CHORDWISE ROW (FROM TI TO ROOT BEGINNING WITH FIRST PLANFORM)' //, 25F5.0, //, 25F5.0 )
18 FORMAT ( ///37X, I5, 'HORSESHOES USED ON THE LEFT HALF OF THE CONFIGURATION', //, 50X, 'PLANFORM TOTAL SPANWISE' / )
19 FORMAT (52X, I4, 10X, I3, 11X, I4)

PART ONE - GEOMETRY COMPUTATION

SECTION ONE - INPUT OF REFERENCE WING POSITION

RTCDHT(1)=0
RTCDHT(2)=0.
YTOL = 1. E-10
AZY = 1. E+13
PIT = 1.5707963
RAD = 57.29578
IF (TOTAL.GT.0.) GO TO 80

SET PLAN EQUAL TO 1. FOR A WING ALONE COMPUTAION - EVEN FOR A VARIABLE SWEEP WING
SET PLAN EQUAL TO 2. FOR A WING - TAIL COMBINATION
SET TOTAL EQUAL TO THE NUMBER OF SETS
OF GROUP TWO DATA PROVIDED

40 READ (28,3,END=1006) PLAN,TOTAL,CREF,SREF
   IPLAN =PLAN
SET AAN(IT) EQUAL TO THE MAXIMUM NUMBER OF CURVES REQUIRED TO
DEFINE THE PLANFORM PERIMETER OF THE (IT) PLANFORM.

SET RTCDHT(IT) EQUAL TO THE ROOT CHORD HEIGHT OF THE LIFTING
SURFACE (IT), WHOSE PERIMETER POINTS ARE BEING READ IN, WITH
RESPECT TO THE WING ROOT CHORD HEIGHT.

WRITE (29,1)
DO 58 IT = 1, IPLAN
READ (28,3) AAN(IT), XS(IT), YS(IT), RTCDHT(IT)
N    = AAN(IT)
N1    = N + 1
MAK   = 0
IF (IPLAN.EQ.1)     PRTCON = '     '
IF (IPLAN.EQ.2 .AND. IT.EQ.1 ) PRTCON = '     FIRST'
IF (IPLAN.EQ.2 .AND. IT.EQ.2 ) PRTCON = '     SECOND'
WRITE (29,2) PRTCON, N, RTCDHT(IT), XS(IT), YS(IT)
WRITE(29,17)
DO 59 I = 1, N1
READ (28,3) XREG(I,IT), YREG(I,IT), DIH(I,IT), AMCD
MCD(I,IT) = AMCD
IF (I.EQ.1) GO TO 59
IF (MAK.NE.0 .OR. MCD(I-1,IT).NE.2) GO TO 49
MAK = I-1
49 IF (ABS(YREG(I-1,IT)-YREG(I,IT)).LT.YTOL) GO TO 50
AREG(I-1,IT) = (XREG(I-1,IT)-XREG(I,IT))/(YREG(I-1,IT)-YREG(I,IT))
ASWP = ATAN (AREG(I-1,IT)) * RAD
GO TO 51
50 YREG(I,IT) = YREG(I-1,IT)
AREG(I-1,IT) = AZY
ASWP     = 90.
51 J    = I - 1

WRITE PLANFORM PERIMETER POINTS AND ANGLES

WRITE(29,14) J, XREG(J,IT), YREG(J,IT), ASWP, DIH(J,IT), MCD(J,IT)
DIH(J,IT) = TAN(DIH(J,IT)/RAD)
59 CONTINUE
KFCTS(IT) = MAK
WRITE(29,14) N1, XREG(N1,IT), YREG(N1,IT)
58 CONTINUE

PART 1 - SECTION 2
READ GROUP 2 DATA AND COMPUTE DESIRED WING POSITION

SCW MUST NOT BE SET EQUAL TO ZERO OR ONE WHEN THE WING HAS DIHEDRAL

SET SA(1), SA(2) EQUAL TO THE SWEEP ANGLE, IN DEGREES, FOR THE FIRST
CURVE(S) THAT CAN CHANGE SWEEP FOR EACH PLANFORM

IF A PARTICULAR VALUE OF CL IS DESIRED AT WHICH THE LOADINGS ARE
TO BE COMPUTED, SET CLDES EQUAL TO THIS VALUE
SET CLDES EQUAL TO 11. FOR A DRAG POLAR AT CL VALUES OF -.1 TO 1.0
C IF PTEST IS SET EQUAL TO ONE THE PROGRAM WILL COMPUTE CLP
C IF QTEST IS SET EQUAL TO ONE THE PROGRAM WILL COMPUTE CMQ AND CLQ
C DO NOT SET BOTH PTEST AND QTEST TO ONE FOR A SINGLE CONFIGURATION
C
C SET TWIST(1) OR TWIST(2) EQUAL TO 0. FOR A FLAT PLANFORM AND TO 1.
C FOR A PLANFORM THAT HAS TWIST AND/OR CAMBER
C
80 READ(28,13,END=1006)CONFIG,SCW,VIC,MACH,CLDES,
1PTEST,QTEST,TWIST(1),SA(1),TWIST(2),SA(2)
WRITE(29,5) CONFIG
82 IF ( PTEST.NE.0 .AND. QTEST.NE.0. ) GO TO 1008
IF (SCW.EQ.0.) GO TO 76
DO 74 I=1,50
74 TBLSCW(I) = SCW
GO TO 78
76 READ (28,3) STA
NSTA = STA
READ (28,3) (TBLSCW(I),TBLSCW(I+1),TBLSCW(I+2),TBLSCW(I+3)
1,TBLSCW(I+4),TBLSCW(I+5),TBLSCW(I+6),TBLSCW(I+7),
2 I = 1,NSTA,8)
78 DO 100 IT = 1,IPLAN
N = AAN(IT)
N1 = N + 1
DO 83 I=1,N
XREF(I) = XREG(I,IT)
YREF(I) = YREG(I,IT)
A(I) = AREG(I,IT)
RSAR(I) = ATAN(A(I))
IF (A(I).EQ.AZY) RSAR(I) = PIT
83 CONTINUE
XREF(N1) = XREG(N1,IT)
YREF(N1) = YREG(N1,IT)
IF ( KFCTS(IT) .GT. 0 ) GO TO 79
K = 1
SA(IT) = RSAR(1) * RAD
GO TO 77
79 K = KFCTS(IT)
77 WRITE (29,10) K,SA(IT),IT
SB = SA(IT)/RAD
IF ( ABS( SB - RSAR(K) ).GT. (.1/RAD) ) GO TO 111
C REFERENCE PLANFORM COORDINATES ARE STORED UNCHANGED FOR WINGS
C WITHOUT CHANGE IN SWEEP
C
DO 113 I=1,N
X(I)=XREF(I)
Y(I)=YREF(I)
IF (RSAR(I) .EQ. PIT ) GO TO 114
A(I)=TAN(RSAR(I))
GO TO 113
114 A(I)=AZY
113 SAR(I)=RSAR(I)
X(N1)=XREF(N1)
Y(N1)=YREF(N1)
GO TO 103
C
C CHANGES IN WING SWEEP ARE MADE HERE

C
111 IF (MCD(K,IT).NE.2)  GO TO 1007
  KA=K-1
  DO 81 I=1,KA
  X(I)=XREF(I)
  Y(I)=YREF(I)
81 SAR(I)=RSAR(I)
C DETERMINE LEADING EDGE INTERSECTION BETWEEN FIXED AND VARIABLE
C SWEEP WING SECTIONS
  SAR(K) = SB
  A(K) = TAN(SB)
  SAI = SB-RSAR(K)
  X(K+1)=X(IT)+(XREF(K+1)-X(IT))*COS(SAI)+(YREF(K+1)-Y(IT))
  Y(K+1)=Y(IT)+(YREF(K+1)-Y(IT))*COS(SAI)-(XREF(K+1)-X(IT))
  IF ((ABS(SB-SAR(K-1)).LT.(.1/RAD))
    GO TO 84
  Y(K)=X(K)+X(K-1)-A(K)*Y(K+1)+A(K-1)*Y(K-1)
  Y(K)=Y(K)/((A(K+1)-A(K))
  X(K)=A(K)*X(K-1)-A(K-1)*X(K+1)+A(K-1)*A(K)*(Y(K+1)-Y(K-1))
  X(K)=X(K)/(A(K)-A(K-1))
  GO TO 85
C ELIMINATE EXTRANEOUS BREAKPOINTS
84 X(K)=XREF(K-1)
  Y(K)=YREF(K-1)
  SAR(K) = SAR(K-1)
  K=K+1
C SWEEP THE BREAKPOINTS ON THE VARIABLE SWEEP PANEL
  (IT ALSO KEEPS SWEEP ANGLES IN FIRST OR FOURTH QUADRANTS)
86 K=K+1
  SAR(K-1)=SAR(K-1)+RSAR(K-1)
99 IF (SAR(K-1).LE.PIT)  GO TO 102
  SAR(K-1)=SAR(K-1)-3.1415927
  GO TO 99
102 IF (SAR(K-1).GE.(-PIT))  GO TO 106
  SAR(K-1)=SAR(K-1)+3.1415927
  GO TO 102
106 IF((SAR(K-1)).LT.0) GO TO 108
  IF (SAR(K-1) - PIT)  90,87,87
  IF (SAR(K-1) + PIT)  89,89,90
  A(K-1)=AZY
  GO TO 91
87 A(K-1)=AZY
  GO TO 91
90 A(K-1)=TAN(SAR(K-1))
91 KK = MCD(K,IT)
  GO TO (93,92),KK
92 Y(K)=Y(IT)+(YREF(K-YS(IT))*COS(SAI)-(XREF(K)-XS(IT))
  1 *SIN(SAI)
  X(K)=X(IT)+(XREF(K)-XS(IT))*COS(SAI)+(YREF(K)-YS(IT))
  1 *SIN(SAI)
  GO TO 86
C DETERMINE THE TRAILING EDGE INTERSECTION
C BETWEEN FIXED AND VARIABLE SWEEP WING SECTIONS
93 IF (ABS(RSAR(K)-SAR(K-1)).LT.(.1/RAD)) GO TO 96
  Y(K)=XREF(K+1)-X(K-1)-A(K)*YREF(K+1)+A(K-1)*Y(K-1)
  Y(K)=Y(K)/(A(K-1)-A(K))
\[ X(K) = A(K) * X(K-1) - A(K-1) * XREF(K+1) / (A(K) - A(K-1)) \]

\[ X(K) = X(K) / (A(K) - A(K-1)) \]

\[ Y(K) = YREF(K+1) \]

\[ K = K + 1 \]

C STORE REFERENCE PLANFORM COORDINATES ON INBOARD FIXED TRAILING EDGE

DO 98 I = K, N1
   X(I) = XREF(I)
   Y(I) = YREF(I)
98 SAR(I-1) = RSAR(I-1)

DO 101 I = 1, N
   XX(I, IT) = X(I)
   YY(I, IT) = Y(I)
   MMCD(I, IT) = MCD(I, IT)
   TTWD(I, IT) = DIH(I, IT)
101 AS (I, IT) = A(I)
   XX(N1, IT) = X(N1)
   YY(N1, IT) = Y(N1)
   AN(IT) = AAN(IT)
100 CONTINUE

C LINE UP BREAKPOINTS AMONG PLANFORMS

299 BOTSV(1) = 0
   BOTSV(2) = 0.
   WRITE (29, 16)
   DO 180 IT = 1, IPLAN
      NIT = AN(IT) + 1
      DO 178 IIT = 1, IPLAN
         IF (IIT .EQ. IT) GO TO 178
         NITT = AN(IIT) + 1
      176 CONTINUE
      DO 170 JP = 1, NIT
         IF (YY(JP, IT) .LT. YY(I, IIT)) GO TO 168
      170 CONTINUE
      GO TO 176
   168 JPSV = JP
      IND = NIT - (JPSV - 1)
      DO 172 JP = 1, IND
         K2 = NIT - JP + 2
         K1 = NIT - JP + 1
         XX(K2, IT) = XX(K1, IT)
         YY(K2, IT) = YY(K1, IT)
         MMCD(K2, IT) = MMCD(K1, IT)
         AS(K2, IT) = AS(K1, IT)
      172 TTWD(K2, IT) = TTWD(K1, IT)
      YY(JPSV, IT) = YY(I, IIT)
      AS(JPSV, IT) = AS(JPSV-1, IT)
      TTWD(JPSV, IT) = TTWD(JPSV-1, IT)
      XX(JPSV, IT) = (YY(JPSV, IT) - YY(JPSV-1, IT)) * AS(JPSV-1, IT)

202
1 + XX(JPSV-1,IT)
MMCD(JPSV,IT) = MMCD(JPSV-1,IT)
AN(IT) = AN(IT) + 1.
NIT = NIT + 1
176 CONTINUE
178 CONTINUE

C SEQUENCE WING COORDINATES FROM TIP TO ROOT

C
N1 = AN(IT)+ 1.
DO 203 I=1,N1
203 Q(I) = YY(I,IT)
DO 208 J=1,N1
HIGH = 1.
DO 205 I=1,N1
IF (( Q(I)-HIGH).GE.0. ) GO TO 205
HIGH = Q(I)
IH = I
205 CONTINUE
IF (J.NE.1) GO TO 206
BOTS(1) = HIGH
KFX(IT) = IH
206 Q(IH) = 1.
SPY(J,IT) = HIGH
IF (IH.GT.KFX(IT)) GO TO 209
IYL(J,IT) = 1
IYT(J,IT) = 0
GO TO 208
209 IYL(J,IT) = 0
IYT(J,IT) = 1
208 CONTINUE
180 CONTINUE

C SELECT MAXIMUM B/2 AS THE WING SEMISPAN

C
KBOT = 1
IF (BOTS(1).GE.BOTS(2)) KBOT = 2
BOT = BOTS(KBOT)

C COMPUTE NOMINAL HORSSEHOE VORTEX WIDTH ALONG WING SURFACE

C
TSPAN = 0
ISAVE = KFX(KBOT) - 1
I  = KFX(KBOT) - 2
216 IF (I.EQ.0) GO TO 217
IF(TTWD(I,KBOT).EQ.TTWD(ISAVE,KBOT)) GO TO 218
217 CTWD = COS( ATAN(TTWD(ISAVE,KBOT) ) )
TLGTH = (YY(ISAVE+1,KBOT) - YY(I+1,KBOT) ) / CTWD
TSPAN = TSPAN + TLGTH
IF (I.EQ.0) GO TO 219
ISAVE = I.
218 I  = I  -1
GO TO 216
219 VI = TSPAN / VIC
VSTOL = VI / 2
ELIMINATE PLANFORM BREAKPOINTS WHICH ARE WITHIN \((B/2)/2000\) UNITS LATERALLY

DO 220 IT = 1, IPLAN
N = AN(IT)
N1 = N + 1
DO 220 J = 1, N
AA = ABS(SPY(J, IT) - SPY(J+1, IT))
IF (AA.EQ.0. OR. AA.GT.ABS(TSPAN/2000.)) GO TO 220
IF (AA.GT.YTOL) WRITE(6,19) SPY(J+1, IT), SPY(J, IT)
DO 222 I = 1, N1
IF (YY(I, IT).NE.SPY(J+1, IT)) GO TO 222
YY(I, IT) = SPY(J, IT)
222 CONTINUE
SPY(J+1, IT) = SPY(J, IT)
220 CONTINUE

COMPUTE Z COORDINATES

DO 236 IT = 1, IPLAN
JM = AN(IT) + 1.
N1 = AN(IT) + 1.
DO 230 JZ = 1, N1
230 ZZ(JZ, IT) = RTCDHT(IT)
JZ = 1
232 JZ = JZ + 1
IF (JZ.GT.KFX(IT)) GO TO 234
ZZ(JZ, IT) = ZZ(JZ-1, IT) + (YY(JZ, IT) - YY(JZ-1, IT)) * TTWD(JZ-1, IT)
GO TO 232
234 JM = JM - 1
IF (JM.EQ.KFX(IT)) GO TO 236
ZZ(JM, IT) = ZZ(JM+1, IT) + (YY(JM, IT) - YY(JM+1, IT)) * TTWD(JM, IT)
GO TO 234
236 CONTINUE

WRITE PLANFORM PERIMETER POINTS ACTUALLY USED IN THE COMPUTATIONS

WRITE (29,8)
DO 240 IT = 1, IPLAN
N = AN(IT)
N1 = N + 1
IF (IT.EQ.2) WRITE (29,18)
DO 238 KK = 1, N
TOUT = ATAN (TIWD(KK, IT)) * RAD
AOUT = ATAN(AS(KK, IT)) * RAD
IF (AS(KK, IT).EQ.AZY) AOUT = 90.
WRITE (29,9) KK, XX(KK, IT), YY(KK, IT), ZZ(KK, IT), AOUT, TOUT, MMCD(KK, IT)
238 CONTINUE
WRITE (29,9) N1, XX(N1, IT), YY(N1, IT), ZZ(N1, IT)
240 CONTINUE

PART ONE - SECTION THREE - LAY OUT YAWED HORSESHOE VORTICES

STRUE = 0.
NSSWSV(1) = 0
NNSWSV(2) = 0
MSV(1) = 0
MSV(2) = 0
700 DO 722 IT=1,IPLAN
   N1 = AN(IT) + 1.
   I = 0
   J = 1
   YIN = BOTSV(IT)
   ILE = KFX(IT)
   ITE = KFX(IT)
   C DETERMINE SPANWISE BORDERS OF HORSESHOE VORTICES
   701 IXL = 0
   IXT = 0
   I = I + 1
   IF(YIN.GE.(SPY(J,IT)+VSTOL)) GO TO 703
   C BORDER IS WITHIN VORTEX SPACING TOLERANCE (VSTOL) OF BREAKPOINT
   C THEREFORE USE THE NEXT BREAKPOINT INBOARD FOR THE BORDER
   VBORD(I) = YIN
   GO TO 707
   C USE NOMINAL VORTEX SPACING TO DETERMINE THE BORDER
   703 VBORD(I) = SPY(J,IT)
   C COMPUTE SUBSCRIPTS ILE AND ITE TO INDICATE WHICH
   C BREAKPOINTS ARE ADJACENT AND WHETHER THEY ARE ON THE WING LEADING
   C EDGE OR THE TRAILING EDGE
   715 IF (J.GE.N1) GO TO 706
   IF (SPY(J,IT).NE.SPY(J+1,IT)) GO TO 706
   IXL = IXL + IYL(J,IT)
   IXT = IXT + IYT(J,IT)
   J = J + 1
   GO TO 715
   706 YIN = SPY(J,IT)
   IXL = IXL + IYL(J,IT)
   IXT = IXT + IYT(J,IT)
   J = J + 1
   707 CPHI = COS ( ATAN ( TTWD(ILE,IT) ) )
   IPHI = ILE - IXL
   IF ( J.GE.N1 ) IPHI = 1
   YIN = YIN - V1*COS ( ATAN ( TTWD(IPHI,IT) ) )
   IF ( I.NE.1) GO TO 709
   708 ILE = ILE - IXL
   ITE = ITE + IXT
   GO TO 701
   C COMPUTE COORDINATES FOR CHORDWISE ROW OF HORSESHOE VORTICES
   709 YQ = ( VBORD(I-1) + VBORD(I) ) / 2.
   HW = ( VBORD(I) - VBORD(I-1))/ 2.
   IM1 = I - 1 + NNSWSV(1)
   ZH(IM1) = ZZ(ILE,IT) + ( YQ - YY(ILE,IT) )*TTWD(ILE,IT)
   PHI(IM1) = TTWD(ILE,IT)
   SSSWA(IM1) = AS(ILE,IT)
   XLE = XX(ILE,IT) + AS(ILE,IT) *(YQ - YY(ILE,IT) )
   XTE = XX(ITE,IT) + AS(ITE,IT) *(YQ - YY(ITE,IT) )
   XLOCAL = ( XLE - XTE ) / TBLSCW(IM1)
   C C COMPUTE WING AREA PROJECTED TO THE X - Y PLANE
   C

205


```c
C
STRUE = STRUE + XLOCAL * TBLSCW(IM1) * (HW * 2.) * 2.

C
NSCW = TBLSCW(IM1)
DO 720 JCW=1,NSCW
AJCW = JCW - 1
XLEL = XLE - AJCW * XLOCAL
NTS = JCW + MSV(1) + MSV(2)
PN(NTS) = XLEL - .25 * XLOCAL
PV(NTS) = XLEL - .75 * XLOCAL
PSI(NTS) = ((XLE - PN(NTS)) + AS(ITE, IT) + (PN(NTS) - XTE) * AS(ILO, IT)) / (XLE - XTE) * CPHI
S(NTS) = HW / CPHI
Q(NTS) = YQ
720 CONTINUE
MSV(IT) = MSV(IT) + NSCW
C
TEST TO DETERMINE WHEN WING ROOT (Y=0) IS REACHED
IF ( VBORD(I) .LT. -0. ) GO TO 708

C
NSSWSV(IT) = I - 1
722 CONTINUE
M = MSV(1) + MSV(2)
C
COMPUTE ASPECT RAT10 AND AVERAGE CHORD
C
BOT = - BOT
AR = 4. * BOT * BOT / SREF
ARTRUE = 4. * BOT * BOT / STRUE
CAVE = STRUE / (2. * BOT)
BETA = (1. - MACH * MACH) ** .5
NVTWO = 0
DO 354 IT=1,IPLAN
NVONE = 1 + (IT-1) * MSV(1)
NVTWO = NVTWO + MSV(IT)
IF (TWIST(IT) .LE. 0. ) GO TO 350
READ(28,3) (ALP(NV),ALP(NV+1),ALP(NV+2),ALP(NV+3),ALP(NV+4),ALP(NV+5),ALP(NV+6),ALP(NV+7),NV=NVONE,NVTWO,8)
GO TO 354
350 DO 351 NV = NVONE, NVTWO
351 ALP(NV) = 0.
354 CONTINUE
WRITE (29,24) M
WRITE (29,25) (IT, MSV(IT), NSSWSV(IT), IT=1,IPLAN)
IF (SCW.NE.O.) WRITE (29,20) SCW
IF (SCW.EQ.O.) WRITE (29,22) (TBLSCW(I), I=1,NSTA)
C
APPLY PRANDTL-GLAUERT CORRECTION
C
DO 360 NV = 1,M
PSI(NV) = ATAN(PSI(NV)/BETA)
PN (NV) = PN(NV) / BETA
360 PV (NV) = PV(NV) / BETA
RETURN
1006 ICODEOF = 1
WRITE(29,11) CONFIG
RETURN
```
SUBROUTINE GRAPH1

C
C DEFINE IPACK ARRAY FOR LEGEND
INTEGER*4 IPACK(35)
INTEGER NV(100),NSSW
REAL YQ(100),SWALE(100),SSCDRAG(100),SSCTRST(100),
+ SSCSUCT(100),CDRAGS(100),CTHRUSS(100),CSUCTS(100)
REAL MAX,MIN,VALMAX,VALMIN
CHARACTER*40 LI
COMMON /PLT1/NSSW
DIMENSION CDRAGS1(100),YQ1(100)

C READ ELEMENTS OF UNIT 11 INTO ARRAYS TO PLOT
OPEN(UNIT=11,FILE='AER01.DAT',STATUS='OLD')
DO 25 I = 1,NSSW
  READ(11,1071)NV(I),YQ(I),SWALE(I),SSCDRAG(I),SSCTRST(I),
  + SSCSUCT(I),CDRAGS(I),CTHRUSS(I),CSUCTS(I)
  DUM = YQ(I)
  DUMM = CDRAGS(I)
  YQ1(I) = DUM
  CDRAGS1(I) = DUMM
1071 FORMAT (10X,I10,5X,8F12.5)
25 CONTINUE
CLOSE (UNIT = 11)
CALL MAXMIN(YQ1,NSSW,VALMAX,VALMIN)
CALL MAXMIN(CDRAGS1,NSSW,MAX,MIN)
C DEFINE AND ASSIGN LEGEND CHARACTER STRINGS
L1 = 'INDUCED DRAG COEFFICIENT$
C INITIALIZE THE GRAPHICS SYSTEM
CALL INIT
C LABEL X AND Y AXES USING SELF COUNTING STRINGS
CALL XNAME('2Y/B$',100)
CALL YNAME('INDUCED DRAG COEFFICIENT$',100)
C DEFINE PLOT AREA TO BE 6 INCHES BY 8 INCHES
CALL AREA2D(6.0,8.0)
C DEFINE HEADING LABEL
CALL HEADINC('INDUCED DRAG COEFF. VS. 2Y/B$','-100,1.8,1')
C PLOT ADDITIONAL TICK MARKS
CALL XTICKS(1)
CALL YTICKS(1)
C PACK LEGEND LABELS INTO ARRAY IPACK
CALL LINES(L1,IPACK,1)
C SET UP AXIS
CALL GRAF(0.,2.1.,(MIN-.005),((MAX-MIN)/5.),((MAX+.005))
C FRAME THE SUBPLOT AREA
CALL FRAME
CALL MARKER(15)
CALL THKCRV(.04)
CALL CURVE(YQ,CDRAGS,NSSW,1)
C CHANGE LEGEND NAME TO "CONTRIBUTION TO TOTAL COEFF."
   CALL MYLEGN('CONTRIBUTION TO TOTAL COEFF.$',100)
C PLOT LEGEND
   CALL LEGEND(IPACK,2,1.2,7.25)
C END PLOT
   CALL ENDPLO(0)
C CREATE GRAPHICS METAFILE P1.UIS
   CALL METAFL(1)
C TERMINATE PLOT AT END OF PLOTTING SESSION
   CALL DONEPL
   RETURN
END

SUBROUTINE GRAPH2
C
C DEFINE IPACK ARRAY FOR LEGEND
INTEGER*4 IPACK(35)
INTEGER NV(100),NSSW
REAL YQ(100),SWALE(100),SSCDRAG(100),SSCTRST(100),
     + SSCSUCT(100),CDRAGS(100),CTHRUSS(100),CSUCTS(100)
REAL MAX,MIN,VALMAX,VALMIN
CHARACTER*40 LI
COMMON /PLT1/NSSW
DIMENSION CTHRUSS1(100),YQ1(100)
C READ ELEMENTS OF UNIT 11 INTO ARRAYS TO PLOT
OPEN(UNIT=11,FILE='AERO1.DAT',STATUS='OLD')
DO 25 I = 1,NSSW
   READ(11,1071)NV(I),YQ(I),SWALE(I),SSCDRAG(I),SSCTRST(I),
     + SSCSUCT(I),CDRAGS(I),CTHRUSS(I),CSUCTS(I)
   DUM = YQ(I)
   DUMM= CTHRUSS(I)
   YQ1(I) = DUM
   CTHRUSS1(I) = DUMM
1071 FORMAT (10X,I10,5X,8F12.5)
25 CONTINUE
CLOSE (UNIT = 11)
CALL MAXMIN(YQ1,NSSW,VALMAX,VALMIN)
CALL MAXMIN(CTHRUSS1,NSSW,MAX,MIN)
C DEFINE AND ASSIGN LEGEND CHARACTER STRINGS
   L1 = 'LE THRUST COEFFICIENT$'
C . INITIALIZE THE GRAPHICS SYSTEM
   CALL INIT
C LABEL X AND Y AXES USING SELF COUNTING STRINGS
   CALL XNAME('2Y/B$,100)
   CALL YNAME('LE THRUST COEFFICIENTS',100)
C DEFINE PLOT AREA TO BE 6 INCHES BY 8 INCHES
   CALL AREA2D(6.0,8.0)
C DEFINE HEADING LABEL
   CALL HEADIN('LE THRUST COEFF. VS. 2Y/B$','-100,1.8,1)
C PLOT ADDITIONAL TICK MARKS
   CALL XTICKS(1)
   CALL YTICKS(1)
C PACK LEGEND LABELS INTO ARRAY IPACK
   CALL LINES(L1,IPACK,1)
C SET UP AXIS
CALL GRAF(0. ,2.1.,(MIN-.005),((MAX-MIN)/5.), (MAX+.005))
C FRAME THE SUBPLOT AREA
CALL FRAME
C PLOT PRESSURE DISTRIBUTION DATA WITH A THICK LINE AND MARKER 15
CALL MARKER(15)
CALL THKCRV(.04)
CALL CURVE(YQ,CTHRUSS,NSSW,1)
C CHANGE LEGEND NAME TO "CONTRIBUTION TO TOAL COEFF."
CALL MYLEGN('CONTRIBUTION TO TOTAL COEFF. $',100)
C PLOT LEGEND
CALL LEGEND(IPACK,2,1.2,7.25)
C END PLOT
CALL ENDPLO(0)
C CREATE GRAPHICS METAFILE P2.UIS
CALL METAFL(2)
C TERMINATE PLOT AT END OF PLOTTING SESSION
CALL DONEPL
RETURN
END

SUBROUTINE GRAPH3
C C DEFINE IPACK ARRAY FOR LEGEND
INTEGER*4 IPACK(35)
INTEGER NV(100),NSSW
REAL YQ(100),SWALE(100),SSCDRAG(100),SSCTRST(100),
+ SSCSUCT(100),CDRAGS(100),CTHRUSS(100),CSUCTS(100)
REAL MAX,MIN,VALMAX,VALMIN
CHARACTER*40 LI
COMMON /PLT1/NSSW
DIMENSION CSUCTS1(100),YQ1(100)
C READ ELEMENTS OF UNIT 11 INTO ARRAYS TO PLOT
OPEN(UNIT=11,FILE='AERO1.DAT',STATUS='OLD')
DO 25 I = 1,NSSW
   READ(11,1071)NV(I),YQ(I),SWALE(I),SSCDRAG(I),SSCTRST(I),
+ SSCSUCT(I),CDRAGS(I),CTHRUSS(I),CSUCTS(I)
   DUM = YQ(I)
   DUMM = CSUCTS(I)
   YQ1(I) = DUM
   CSUCTS1(I) = DUMM
1071 FORMAT (10X,I10,5X,8F12.5)
25 CONTINUE
CLOSE (UNIT = 11)
CALL MAXMIN(YQ1,NSSW,VALMAX,VALMIN)
CALL MAXMIN(CSUCTS1,NSSW,MAX,MIN)
C DEFINE AND ASSIGN LEGEND CHARACTER STRINGS
LI = 'SUCTION COEFFICIENTS'
C INITIALIZE THE GRAPHICS SYSTEM
CALL INIT
C LABEL X AND Y AXES USING SELF COUNTING STRINGS
CALL XNAME('2Y/B$',100)
CALL YNAME('SUCTION COEFFICIENTS$',100)
C DEFINE PLOT AREA TO BE 6 INCHES BY 8 INCHES
CALL AREA2D(6.0,8.0)
C DEFINE HEADING LABEL
CALL HEADIN('SUCTION COEFF. VS. 2Y/B$','-100,1.8,1')
C PLOT ADDITIONAL TICK MARKS
CALL XTICKS(1)
CALL YTICKS(1)
C PACK LEGEND LABELS INTO ARRAY IPACK
CALL LINES(L1,IPACK,1)
C SET UP AXIS
CALL GRAF(0.,2.1.,MIN,(ABS((MAX-MIN)/5.)),(ABS(MAX+.05)))
C FRAME THE SUBPLOT AREA
CALL FRAME
C PLOT PRESSURE DISTRIBUTION DATA WITH A THICK LINE AND MARKER 15
CALL MARKER(15)
CALL THKCRV(.04)
CALL CURVE(YQ,CSUCTS,NSSW,1)
C CHANGE LEGEND NAME TO "CONTRIBUTION TO TOTAL COEFF"
CALL MYLEGNC('CONTRIBUTION TO TOTAL COEFF.$',100)
C PLOT LEGEND
CALL LEGEND(IPACK,2,1.2,7.25)
C END PLOT
CALL ENDPL(0)
C CREATE GRAPHICS METAFILE P3.UIS
CALL METAFL(3)
C TERMINATE PLOT AT END OF PLOTTING SESSION
CALL DONEPL
RETURN
END

SUBROUTINE GRAPH4

C DEFINE IPACK ARRAY FOR LEGEND
INTEGER*4 IPACK(35)
INTEGER NV(100),NSSW
REAL YQ(100),SLOAD(100),CLCL(100),CCAV(100),
+ BCLCC(100),BADLAE(100),BASLD(100),SLDT(100)
REAL MAX,MIN,VALMAX,VALMIN
CHARACTER*40 LI
COMMON /PLT1/NSSW
DIMENSION SLOAD1(100),YQ1(100)
C READ ELEMENTS OF UNIT 12 INTO ARRAYS TO PLOT
OPEN(UNIT=12,FILE='AERO2.DAT',STATUS='OLD')
DO 25 I = 1,NSSW
    READ(12,15)NV(I),YQ(I),SLOAD(I),CLCL(I),CCAV(I),
+        BCLCC(I),BADLAE(I),BASLD(I),SLOAD(I),SLDT(I)
    DUM = YQ(I)
    DUMM=SLOAD(I)
    YQ1(I)=DUM
    SLOAD1(I)=DUMM
15 FORMAT(4X,I4,F12.5,5X,3F12.5,3X,3F12.5,3X,2F12.5)
25 CONTINUE
CLOSE (UNIT = 12)
CALL MAXMIN(YQ1,NSSW,VALMAX,VALMIN)
CALL MAXMIN(SLOAD1,NSSW,MAX,MIN)
C DEFINE AND ASSIGN LEGEND CHARACTER STRINGS
L1 = 'SPAN LOAD COEFFICIENTS'
C INITIALIZE THE GRAPHICS SYSTEM

210
CALL INIT
C LABEL X AND Y AXES USING SELF COUNTING STRINGS
   CALL XNAME('2Y/B$',100)
   CALL YNAME('SPAN LOAD COEFFICIENT$',100)
C DEFINE PLOT AREA TO BE 6 INCHES BY 8 INCHES
   CALL AREA2D(6.0,8.0)
C DEFINE HEADING LABEL
   CALL HEADIN('SPAN LOAD COEFF. VS. 2Y/B$',-100,1.8,1)
C PLOT ADDITIONAL TICK MARKS
   CALL XTICKS(1)
   CALL YTICKS(1)
C PACK LEGEND LABELS INTO ARRAY IPACK
   CALL LINES(L1,IPACK,1)
C SET UP AXIS
   CALL GRAF(0.,2.,1.,MIN,(ABS((MAX-MIN)/5.)),(ABS(MAX+.05)))
C FRAME THE SUBPLOT AREA
C PLOT PRESSURE DISTRIBUTION DATA WITH A THICK LINE AND MARKER 15
   CALL MARKER(15)
   CALL THKCRV(.04)
   CALL CURVE(YQ,SLOAD,NSSW,1)
C CHANGE LEGEND NAME TO "COEFFICIENT OF LIFT(WING) = 1.0"
   CALL MYLEGN('COEFFICIENT OF LIFT(WING) = 1.0$',100)
C PLOT LEGEND
   CALL LEGEND(IPACK,2,1.2,7.25)
C END PLOT
C CREATE GRAPHICS METAFILE P4.UIS
   CALL METAFL(4)
C TERMINATE PLOT AT END OF PLOTTING SESSION
   CALL DONEPL
RETURN
END

SUBROUTINE GRAPH5
C C DEFINE IPACK ARRAY FOR LEGEND
   INTEGER*4 IPACK(35)
   INTEGER NV(100),NSSW
   REAL YQ(100),SLOAD(100),CLCL(100),CCAV(100),
      + BCLCC(100),BADLAE(100),BASLD(100),SLDT(100)
   REAL MAX,MIN,VALMAX,VALMIN
   CHARACTER*40 L1
   COMMON /PLT1/NSSW
   DIMENSION CLCL1(100),YQ1(100)
C READ ELEMENTS OF UNIT 12 INTO ARRAYS TO PLOT
   OPEN(UNIT=12,FILE='AERO2.DAT',STATUS='OLD')
   DO 25 I = 1,NSSW
      READ(12,15)NV(I),YQ(I),SLOAD(I),CLCL(I),CCAV(I),
         + BCLCC(I),BADLAE(I),BASLD(I),SLOAD(I),SLDT(I)
      DUM = YQ(I)
      DUMM = CLCL(I)
      YQ1(I)=DUM
      CLCL1(I)=DUMM
   15 FORMAT(4X,I4,F12.5,5X,3F12.5,3X,3F12.5,3X,2F12.5)
   25 CONTINUE
CLOSE (UNIT = 12)
CALL MAXMIN(YQ1,NSSW,VALMAX,VALMIN)
CALL MAXMIN(CLCL1,NSSW,MAX,MIN)
C DEFINE AND ASSIGN LEGEND CHARACTER STRINGS
   L1 = ' COEFFICIENT OF LIFT RATIO$'
C INITIALIZE THE GRAPHICS SYSTEM
CALL INIT
C LABEL X AND Y AXES USING SELF COUNTING STRINGS
   CALL XNAME('2Y/B$',100)
   CALL YNAME('COEFFICIENT OF LIFT RATIO (SECTION/WING)$',100)
C DEFINE PLOT AREA TO BE 6 INCHES BY 8 INCHES
   CALL AREA2D(6.0,8.0)
C DEFINE HEADING LABEL
   CALL HEADIN('COEFF. OF LIFT RATIO VS. 2Y/B$,-100,1.8,1')
C PLOT ADDITIONAL TICK MARKS
   CALL XTICKS(1)
   CALL YTICKS(1)
C PACK LEGEND LABELS INTO ARRAY IPACK
   CALL LINES(L1,IPACK,1)
C SET UP AXIS
   CALL GRAF(0.,2,1.,MIN,(ABS((MAX-MIN)/5.)),(ABS(MAX+.05)))
C FRAME THE SUBPLOT AREA
   CALL FRAME
C PLOT PRESSURE DISTRIBUTION DATA WITH A THICK LINE AND MARKER 15
   CALL MARKER(15)
   CALL THKCRV(.04)
   CALL CURVE(YQ,CLCL,NSSW,1)
C CHANGE LEGEND NAME TO "COEFFICIENT OF LIFT(WING) = 1.0"
   CALL MYLEGNC('COEFFICIENT OF LIFT(WING) = 1.0$',100)
C PLOT LEGEND
   CALL LEGEND(IPACK,2,1.2,7.25)
C END PLOT
   CALL ENDPL(0)
C CREATE GRAPHICS METAFILE P5.UIS
   CALL METAFL(5)
C TERMINATE PLOT AT END OF PLOTTING SESSION
   CALL DONEPL
   RETURN
   END

SUBROUTINE GRAPH6(NUMVOR)

C DEFINE IPACK ARRAY FOR LEGEND
INT INTEGER*4 IPACK(35)
INTEGER NUMVOR,METH,MORT,INC,MANY,COUNT
REAL PNPRS(120),PVPRS(120),QS(120),CPS(120)
REAL MAX,MIN,VALMAX,VALMIN
CHARACTER*40 L1
COMMON/SHIP/VIC,SCW
DIMENSION CPS1(120),PNPRS1(120)
MANY = INT(SCW)
603 FORMAT(1X,2I3,4F12.5)
C READ ELEMENTS OF UNIT 13 INTO ARRAYS TO PLOT
   OPEN(UNIT=13,FILE='AERO3.DAT',STATUS='OLD')
   INC = 1
   COUNT = 0
14 READ(13,603) METH,MORT,PNPR,PVPR,Q,CP
IF (METH.EQ.NUMVOR) THEN
  PNPRS(INC) = PNPR
  PVPRS(INC) = PVPR
  QS(INC) = Q
  CPS(INC) = CP
  INC = INC + 1
  COUNT = COUNT + 1
  GO TO 14
ELSEIF(METH.EQ.99) THEN
  GO TO 15
ELSE
  GO TO 14
ENDIF
15 PRINT *, ' '
CLOSE (UNIT = 13)
DO I = 1,COUNT
  DUM=CPS(I)
  DUMM=PNPRS(I)
  CPS1(I)=DUM
  PNPRS1(I)=DUMM
END DO
CALL MAXMIN(PNPRS1,COUNT,VALMAX,VALMIN)
call maxmin(cps1,count,max,min)
C DEFINE AND ASSIGN LEGEND CHARACTER STRINGS
L1 = ' DELTA CP VS. X C/4 $'
C INITIALIZE THE GRAPHICS SYSTEM
CALL INIT
C LABEL X AND Y AXES USING SELF COUNTING STRINGS
CALL XNAME('X C/4$',100)
CALL YNAME('COEFFICIENT OF PRESSURE CHANGE$',100)
C DEFINE PLOT AREA TO BE 6 INCHES BY 8 INCHES
CALL AREA2D(6.0,8.0)
C DEFINE HEADING LABEL
CALL HEADIN('DELTA CP VS. X C/4$',-100,1.8,1)
C PLOT ADDITIONAL TICK MARKS
CALL XTICKS(1)
CALL YTICKS(1)
C PACK LEGEND LABELS INTO ARRAY IPACK
CALL LINES(L1,IPACK,1)
C SET UP AXIS
  CALL GRAF(((VALMIN-.05),((VALMAX-VALMIN)/2.5),(VALMAX+.05),
+ (MIN-.1),((MAX-MIN)/5.),(MAX+.1))
C FRAME THE SUBPLOT AREA
CALL FRAME
C PLOT PRESSURE DISTRIBUTION DATA WITH A THICK LINE AND MARKER 15
CALL MARKER(15)
CALL THKCRV(.04)
CALL CURVE(PNPRS,CPS,COUNT,1)
C PLOT MESSAGE
  CALL MESSAG('HORSESHOE VORTEX -- NUMBER $',100,1.,8.25)
  CALL INTNO(NUMVOR,'ABUT','ABUT')
C CHANGE LEGEND NAME TO "COEFFICIENT OF LIFT(WING) = 1.0"
  CALL MYLEGNC('COEFFICIENT OF LIFT(WING) = 1.0$',100)
C PLOT LEGEND
CALL LEGEND(IPACK, 2, 1.2, 7.25)
CALL ENDPL(0)
CREATE GRAPHICS METAFILE P6.UIS
CALL METAFL(6)
TERMINATE PLOT AT END OF PLOTTING SESSION
CALL DONEPL
RETURN
END

SUBROUTINE INFSUB (BOT, FUI, FVI, FWI)

COMMON/INSUB23/PSII, APHII, XXX, YYY, ZZZ, SNN, TOLRNC
FC = COS(PSII)
FS = SIN(PSII)
FT = FS/FC
FPC = COS(APHII)
FPS = SIN(APHII)
FPT = FPS/FPC
F1 = XXX + SNN * FT * FPC
F2 = YYY + SNN * FPC
F3 = ZZZ + SNN * FPS
F4 = XXX - SNN * FT * FPC
F5 = YYY - SNN * FPC
F6 = ZZZ - SNN * FPS
FFA = (XXX**2 + (YYY * FPS)**2 + FPC**2 * ((YYY * FT)**2 + (ZZZ / FC)**2 - 2. * XXX * YYY * FT) - 2. * ZZZ * FPC * (YYY * FPS + XXX * FT * FPS))
FFB = (F1 * F1 + F2 * F2 + F3 * F3)**.5
FFC = (F4 * F4 + F5 * F5 + F6 * F6)**.5
FFD = F5 * F5 + F6 * F6
FFE = F2 * F2 + F3 * F3
FFF = (F1 * FPC * FT + F2 * FPC + F3 * FPS) / FFB - (F4 * FPC * FT + F5 * FPC + F6 * FPS) / FFC

THE TOLERANCE SET AT THIS POINT IN THE PROGRAM MAY NEED TO BE CHANGED FOR COMPUTERS OTHER THAN THE CDC 6000 SERIES

IF(ABS(FFA).LT.(BOT*15.E-5)**2) GO TO 262
FUONE = (ZZZ * FPC - YYY * FPS) * FFF / FFA
FVONE = (XXX * FPS - ZZZ * FT * FPC) * FFF / FFA
FWONE = (YYY * FT - XXX) * FFF / FFA * FPC
GO TO 265
262 FUONE = 0
FVONE = 0
FWONE = 0.
265 IF(ABS(FFD).LT.TOLRNC) GO TO 263
FVTWO = F6 * (1. - F4 / FFC) / FFD
FWTWO = - F5 * (1. - F4 / FFC) / FFD
GO TO 266
263 FVTWO = 0
FWTWO = 0.
266 IF(ABS(FFE).LT.TOLRNC) GO TO 264
FVTHRE = - F3 * (1. - F1 / FFB) / FFE
FWTHRE=F2*(1.-F1/FFB)/FFE

GO TO 267
264 FVTHRE=0
FWTHRE=0.
267 FUI=FUONE
FVI=FVONE+FVTWO+FVTHRE
FWI=FUONE+FWTWO+FWTHRE
RETURN
END

SUBROUTINE MATX

DIMENSION YY(2),FU(2),FV(2),FW(2),FVN(300,300),IPIVOT(300),
1 INDEX(300,2)
COMMON/ALL/ BOT,M,BETA,PTEST,QTEST,TBLSCW(50),Q(300),PN(300),
1 PV(300),ALP(300),S(300),PSI(300),PHI(300),ZH(50)
COMMON/TOTHERE/ CIR(300,2),SECTRST(50)
COMMON/INSUB23/APSI,APHI,XX,YYY,ZZ,SNN,TOLC

PART 2 - COMPUTE CIRCULATION TERMS

FPI = 12.5663704
IM = 2
NMAX = 300

THE TOLERANCE SET AT THIS POINT IN THE PROGRAM MAY NEED TO BE
CHANGED FOR COMPUTERS OTHER THAN THE CDC 6000 SERIES

TOLC=(BOT*15.E-05)**2
DO 6667 NUU=1,NMAX
DO 6667 NUT=1,NMAX
FVN(NUU,NUT)=0.
6667 CONTINUE
DO 308 NV=1,M
CIR(NV,1)= 12.5663704 * ALP(NV)
CIR(NV,2)= 12.5663704
IF (PTEST.NE.0.) CIR(NV,2) = -1.0964155 * Q(NV) / BOT
IF (QTEST.NE.0.) CIR(NV,2) = -1.0964155 * PV(NV) *BETA
308 CONTINUE
IZZ=1
NNV=TBLSCW(IZZ)
DO 314 NV=1,M
IZ=1
NNN=TBLSCW(IZ)
DO 316 NN=1,M
APHI = 'ATAN(PHI(IZ))
APSI = PSI(NN)
XX=PV(NV)-PN(NN)
YY(1)=Q(NV)-Q(NN)
YY(2) = Q(NV) + Q(NN)
ZZ = ZH(IZZ) - ZH(IZ)
SNN = S(NN)
DO 261 I = 1, 2
YYY = YY(I)
CALL INFSUB (BOT, FU(I), FV(I), FW(I))
APHI = -APHI
APSI = -APSI
261 CONTINUE
IF (PTEST. NE. 0.) GO TO 342
FVN(NV, NN) = FW(1) - FV(1) * PHI(IZ) + FW(2) - FV(2) * PHI(IZ)
GO TO 312
342 FVN(NV, NN) = FW(1) - FV(1) * PHI(IZ) - FW(2) + FV(2) * PHI(IZ)
312 IF (NN. LT. NNN . OR. NN. EQ. M ) GO TO 316
IZ = IZ + 1
NNN = NNN + TBLSCW(IZ)
316 CONTINUE
IF (NV. LT. NNV . OR. NV. EQ. M ) GO TO 314
IZZ = IZZ + 1
NNV = NNV + TBLSCW(IZZ)
314 CONTINUE
CALL AMATINV(FVN, M, CIR, IM, DETERM, IPIVOT, INDEX, NMAX, ISCALE)
IZZA = IZZ
DO 320 NZ = 1, IZZA
320 SECTRST(NZ) = 0.
IZZ = 1
NNV = TBLSCW(IZZ)
DO 614 NV = 1, M
IZ = 1
NNN = TBLSCW(IZ)
VELIN = 0.
DO 616 NN = 1, M
APHI = ATAN(PHI(IZ))
APSI = PSI(NN)
XX = PN(NV) - PN(NN)
YY(1) = Q(NV) - Q(NN)
YY(2) = Q(NV) + Q(NN)
ZZ = ZH(IZZ) - ZH(IZ)
SNN = S(NN)
DO 661 I = 1, 2
YYY = YY(I)
CALL INFSUB (BOT, FU(I), FV(I), FW(I))
APHI = -APHI
APSI = -APSI
661 CONTINUE
VELIN = ((FW(1) + FW(2)) - (FV(1) + FW(2)) * TAN(APHI)) * CIR(NN, 2)
1 / FPI + VELIN
IF (NN. LT. NNN . OR. NN. EQ. M ) GO TO 616
IZ = IZ + 1
NNN = NNN + TBLSCW(IZ)
616 CONTINUE
CTCP = - (VELIN - 1.) * 2. * CIR(NV, 2)
SECTRST(IZZ) = SECTRST(IZZ) + CTCP
IF (NV. LT. NNV . OR. NV. EQ. M ) GO TO 614
IZZ = IZZ + 1
NNV = NNV + TBLSCW(IZZ)
SUBROUTINE QUERY(NANS)

C ROUTINE TO TRAP ERRORS CAUSED BY IMPROPER RESPONSES TO QUESTIONS.
C THE COMPUTER GENERATES AN ERROR WHEN A CHARACTER IS SUPPLIED TO
C A QUESTION EXPECTING AN INTEGER OR REAL VALUE.

C

NQTEST=0

1 CONTINUE
   IF (NQTEST .GT. 0) THEN
      PRINT *, 'CHARACTER VALUES ARE NOT VALID.'
      PRINT *, 'PLEASE ENTER AN INTEGER VALUE.'
   END IF
   NQTEST = NQTEST + 1
   READ (5,*,ERR=1)NANS
RETURN
END
APPENDIX J. PROGRAM SUPER COMPUTER CODE

PROGRAM SUPER

*** MODIFIED FOR USE ON THE MICROVAX/2000 BY R. MARGASON.
*** MODIFIED FOR GRAPHICAL OUTPUT AND/OR PRINTING OPTIONS BY C.M.
MACALLISTER (AUG 89). FINAL UPDATES MADE, NOV 89 - (CMM).

THE SUPER PROGRAM HAS BEEN ADAPTED FROM A NATIONAL AERONAUTICS
AND SPACE ADMINISTRATION (NASA) FORTRAN PROGRAM AND HAS BEEN
USED CONSIDERABLY AT THE LANGLEY RESEARCH CENTER AND IN INDUSTRY.
THE PURPOSE OF THE SUPER PROGRAM IS TO ESTIMATE THE SUPERSONIC
AERODYNAMIC CHARACTERISTICS OF COMPLEX PLANFORMS. LINEARIZED
SUPERSONIC LIFTING SURFACE THEORY IS EMPLOYED TO CALCULATE THE
AERODYNAMIC CHARACTERISTICS OF A WARPED WING OF AN ARBITRARY
PLANFORM. THE USE OF THIS PROGRAM IS CONFINED TO THE SUPERSONIC
FLOW REGIME. IN ADDITION, THE LINEARIZED SUPERSONIC LIFTING
SURFACE THEORY APPLIES TO WINGS HAVING NEGLIGIBLE THICKNESS AND
ESSENTIALLY PLANAR CAMBER SURFACES.

DIMENSION DUMB(6891)
INTEGER GRAPHOPT, OUTER, LSTA, NSTA
REAL SPAFRACT, CHOFRACT
CHARACTER*1 PRINT, GRAPH, COPY, PLOT1, PLOT2, PLOT3
CHARACTER*1 PLOT4, PLOT5
CHARACTER*20 CASEFN, OUTFIL
COMMON TYB2(51), TZORD(26,51), JBYMAX, NON, NOPCT, RATIO, XLEO, XTEO,
1TPCT(26), TXLE(50), TXTE(50), DZDX, XMAX, CBAR, TDZDX(105,51), XM, NOM,
2TMACH(5), TZSKAL, REFAR, SPAN, XO, PI, CNPOD, CAPOD, TCNPOD(5), TCAPOD(5)
COMMON FDZDX, XLEOF, TXLEF(50), NFLAP2, NFLAP1, XMREF
COMMON TYPEX, NLEX, NTEX, TBLEX(15), TBLEY(15), TBTEX(15), TBTEY(15)
COMMON IDENT(8)
COMMON /SPAN/ SPAFRACT
COMMON /CHORD/ CHOFRACT
COMMON /PLOT1/LSTA
COMMON /HEALEY/RCL9, RCL9F
EQUIVALENCE (DUMB(1), TYB2(1))
NAMELIST /INPT1/TYB2, TZORD, JBYMAX, NON, NOPCT, XLEO, XTEO, TPCT,
1TXLE, TXTE, XMAX, CBAR, XM, NOM, TMACH, TZSKAL, REFAR, SPAN, XO, CNPOD, CAPOD,
1TCNPOD, TCAPOD, FDZDX, XLEOF, TXLEF, NFLAP2, NFLAP1,
1TYPEX, NTEX, NLEX, TBLEX, TBLEY, TBTEX, TBTEY, XMREF

100 FORMAT (A20)
101 FORMAT (/// ' START OF A NEW CASE, CASE FILE NAME IS ', A20///)
102 FORMAT ( ' THE OUTPUT FILE NAME IS "OUTFILE.DAT" ',//)

CREATE FILES WHICH WILL BE USED TO PLOT THE RESULTS

OPEN FILE FOR SPANWISE PRESSURE DISTRIBUTION OUTPUT
OPEN (UNIT=11,
2 FILE= 'SPWPD.DAT',
2 ORGANIZATION= 'SEQUENTIAL',
2 ACCESS= 'SEQUENTIAL',
2 RECORDTYPE= 'VARIABLE',
2 FORM= 'FORMATTED',

218
WRITE (62,101) CASEFN
WRITE (62,102)
READ (25,3,END=99) IDENT
C
4 DUMB(KAK)=0.0
FDZDX=0.0
PI=3.1415926
RATIO=1.
TYPEX = 0.0
TZSKAL=0
CNPOD=0.0
CAPOD=0.0
TSEC1 = SECNDS(0.0)
READ (25,INPT1)
WRITE (26,201)
WRITE (62,201)
C
PRINT 201
201 FORMAT(1H1//31X,60H*****LINEARIZED THEORY SUPersonic WING ANALYSIS
$  PROGRAM****/46X,28HLANGLEY PROGRAM NUMBER A4410)
1 WRITE (26,200) IDENT
WRITE (62,200) IDENT
200 FORMAT (1//1X,8A10/)//)
WRITE(26,7)
WRITE(62,7)
7 FORMAT(2X///39X,41H***COMPLETE INPUT DATA,NAMElIST FORMAT***)
WRITE(26,INPT1)
WRITE(62,INPT1)
CALL P916AF
TIME = SECNDS(TSEC1)
C
PRINT 8,TIME
WRITE(26,8)TIME
WRITE(62,8)TIME
8 FORMAT(2X///15X,29HCENTRAL PROCESSING UNIT TIME=F12.3,5H SEC.)
WRITE (26,202)
WRITE (62,202)
C
PRINT 202
202 FORMAT (1H1)
CLOSE (UNIT = 25)
CLOSE (UNIT = 26)
CLOSE (UNIT = 62)
PRINT *
PRINT *, ' PROGRAM RESULTS HAVE BEEN WRITTEN TO THE FILE'
PRINT *, 'OUTFILE.DAT.'
PRINT *, 'WOULD YOU LIKE A PRINTED COPY OF THIS OUTPUT FILE?'
PRINT *, 'YES OR NO (Y/N)'
PRINT *
READ 1002, PRINT
1002 FORMAT(A1)
IF (PRINT.EQ.'Y'.OR.PRINT.EQ.'y')THEN
   CALL LIB$SPAWN('PRINT OUTFILE.DAT')
ENDIF
PRINT *
PRINT *, 'WOULD YOU LIKE THE OUTPUT FILE COPIED TO ANOTHER'
FILE FOR FUTURE REFERENCE (Y/N) ?

READ 1002,COPY
IF (COPY .EQ. 'Y'. OR. COPY.EQ. 'y') THEN
    PRINT *, 'WHAT NAME WOULD YOU LIKE FOR THE OUTPUT FILE?'
    PRINT *, '1) TOMCAT.DAT'
    PRINT *, '2) PHANTOM.DAT'
    PRINT *, '3) INTRUDER.DAT'
    PRINT *, '4) CRUSADOR.DAT'
    PRINT *, 'ENTER 1,2,3 OR 4'
ENDIF
READ 1006, OUTER
IF (OUTER .LT. 1 .OR. OUTER .GT. 4) THEN
    PRINT *, 'INVALID ENTRY, ENTER INTEGER BETWEEN'
    PRINT *, 'ONE(1) AND FOUR(4).'
    GO TO 69
ENDIF
IF (OUTER.EQ. 1) CALL LIB$SPAWN('COPY OUTFILE.DAT TOMCAT.DAT')
IF (OUTER.EQ. 2) CALL LIB$SPAWN('COPY OUTFILE.DAT PHANTOM.DAT')
IF (OUTER.EQ. 3) CALL LIB$SPAWN('COPY OUTFILE.DAT INTRUDER.DAT')
IF (OUTER.EQ. 4) CALL LIB$SPAWN('COPY OUTFILE.DAT CRUSADOR.DAT')
PRINT *, 'WOULD YOU LIKE TO GRAPH THE RESULTS (Y/N),'
PRINT *
READ 1002, GRAPH
IF (GRAPH .EQ. 'Y'. OR. GRAPH.EQ. 'y') THEN
PRINT *, ' WHICH OF THE FOLLOWING RELATIONSHIPS'
PRINT *, ' DO YOU WANT PLOTTED?'
PRINT *
PRINT *, '1) SPANWISE PRESSURE DISTRIBUTION'
PRINT *, '2) CHORDWISE PRESSURE DISTRIBUTION'
PRINT *, '3) DRAG POLAR (CL VS. CD)'
PRINT *, '4) STREAMWISE LIFT DISTRIBUTION'
PRINT *, '5) SPANWISE LIFT DISTRIBUTION'
PRINT *, '6) NONE'
PRINT *
PRINT *, 'INPUT OPTION NO. (1,2,3,4,5 OR 6)'
READ 1006, GRAPHOPT
IF (GRAPHOPT .LT. 1 .OR. GRAPHOPT .GT. 6) THEN
    PRINT *, 'INVALID ENTRY, ENTER INTEGER BETWEEN'
    PRINT *, 'ONE(1) AND SIX(6).'
    GO TO 42
ENDIF
C ***************
IF (GRAPHOPT .EQ. 1) THEN
PRINT *, 'AT WHAT CHORDAL FRACTION(X/L) WOULD YOU LIKE TO'
PRINT *, 'SEE THE SPANWISE PRESSURE DISTRIBUTION?'
PRINT *, 'ENTER DECIMAL FRACTION (E.G. .25)'
READ 1008, CHOFRACT
FORMAT(F8.6)
IF (CHOFRACT .LT. 0. .OR. CHOFRAC T.GT.1.) THEN
PRINT *, ' '
PRINT *, 'INVALID ENTRY. TRY AGAIN'
PRINT *, 'PLEASE ENTER DECIMAL NUMERAL (E.G. .25)'
GO TO 67
ENDIF
LSTA = INT(CHOFRACT/.023333333)
CALL GRAPH1(LSTA)
C GET A HARDCOPY OF THIS GRAPHIC
CALL LIB$SPAWN('RENDER/DEVICE=LA210/DRAFT_QUALITY/PAPER_+
+SIZE=A P1.UIS')
CALL LIB$SPAWN('CONTINUE')
PRINT *, '
PRINT *, 'WOULD YOU LIKE A PRINT OF THIS PLOT? (Y/N)'
PRINT *, '
READ 1002, PLOT1
IF (PLOT1.EQ.'Y'.OR.PLOT1.EQ.'y')THEN
  CALL LIB$SPAWN('PRINT P1.REN')
ENDIF
GO TO 41
ENDIF
IF (GRAPHOPT.EQ.2) THEN
  PRINT *, '
  PRINT *, 'AT WHAT HALF SPAN FRACTION(Y/B/2) WOULD YOU LIKE TO'
  PRINT *, '
  PRINT *, 'SEE THE CHORDAL PRESSURE DISTRIBUTION?'
  PRINT *, '
  PRINT *, 'ENTER DECIMAL FRACTION (E.G. .25)'
  68 READ 1008, SPAFRAC
  IF (SPAFRACT.LT.0. OR SPAFRACT.GT.1.) THEN
    PRINT *, '
    PRINT *, 'INVALID ENTRY. TRY AGAIN'
    PRINT *, '
  GO TO 68
  ENDIF
  NSTA = INT(SPAFRAC/.033333333)
  CALL GRAPH2(NSTA)
C GET A HARDCOPY OF THIS GRAPHIC
  CALL LIB$SPAWN('RENDER/DEVICE=LA210/DRAFT_QUALITY/PAPER_+
  +SIZE=A P2.UIS')
  CALL LIB$SPAWN('CONTINUE')
  PRINT *, '
  PRINT *, 'WOULD YOU LIKE A PRINT OF THIS PLOT? (Y/N)'
  PRINT *, '
  READ 1002, PLOT2
  IF (PLOT2.EQ.'Y'.OR.PLOT2.EQ.'y')THEN
    CALL LIB$SPAWN('PRINT P2.REN')
 ENDIF
  GO TO 41
  ENDIF
C ***************
  IF (GRAPHOPT.EQ.3) THEN
    CALL GRAPH3
  ENDIF
C GET A HARDCOPY OF THIS GRAPHIC
  CALL LIB$SPAWN('RENDER/DEVICE=LA210/DRAFT_QUALITY/PAPER_+
  +SIZE=A P3.UIS')
  CALL LIB$SPAWN('CONTINUE')
  PRINT *, '
  PRINT *, 'WOULD YOU LIKE A PRINT OF THIS PLOT? (Y/N)'

PRINT *, ' ',
READ 1002, PLOT3
IF (PLOT3.EQ. 'Y'. OR. PLOT3.EQ. 'y')THEN
  CALL LIB$SPAWN('PRINT P3.REN')
ENDIF
GO TO 41
ENDIF
C *****************
  IF (GRAPHOPT .EQ. 4) THEN
    CALL GRAPH4
C GET A HARDCOPY OF THIS GRAPHIC
    CALL LIB$SPAWN('RENDER/DEVICE=LA210/DRAFT_QUALITY/PAPER_+
+SIZE=A P4.UIS')
    CALL LIB$SPAWN('CONTINUE')
    PRINT *, ' ', 'WOULD YOU LIKE A PRINT OF THIS PLOT? (Y/N)'
    PRINT *, ' ',
    READ 1002, PLOT4
    IF (PLOT4.EQ. 'Y'. OR. PLOT4.EQ. 'y')THEN
      CALL LIB$SPAWN('PRINT P4.REN')
    ENDIF
    GO TO 41
  ENDIF
C *****************
  IF (GRAPHOPT .EQ. 5) THEN
    CALL GRAPH5
C GET A HARDCOPY OF THIS GRAPHIC
    CALL LIB$SPAWN('RENDER/DEVICE=LA210/DRAFT_QUALITY/PAPER_+
+SIZE=A P5.UIS')
    CALL LIB$SPAWN('CONTINUE')
    PRINT *, ' ', 'WOULD YOU LIKE A PRINT OF THIS PLOT? (Y/N)'
    PRINT *, ' ',
    READ 1002, PLOT5
    IF (PLOT5.EQ. 'Y'. OR. PLOT5.EQ. 'y')THEN
      CALL LIB$SPAWN('PRINT P5.REN')
    ENDIF
    GO TO 41
  ENDIF
C *****************
  IF (GRAPHOPT .EQ. 6) THEN
    GO TO 192
  ENDIF
ENDIF
ENDIF
1006 FORMAT(II)
C ****************** OPTION TO MAKE ANOTHER RUN ******************
192 PRINT *
  PRINT *, 'DO YOU WISH TO : ',
  PRINT *, '   1) MAKE ANOTHER RUN OR'
  PRINT *, '   2) END THIS SESSION'
  PRINT *, ' ENTER 1 OR 2 .'  
  PRINT *
  CALL QUERY (NANS)
  CALL CLRSCRN
  CLOSE (UNIT = 11)
  CLOSE (UNIT = 12)
CLOSE (UNIT = 13)
CLOSE (UNIT = 14)
CLOSE (UNIT = 15)
CLOSE (UNIT = 38)
IF (NANS .EQ. 1) GO TO 2
99 STOP
END

SUBROUTINE CLRSCRN

C LIBRARY ROUTINE TO CLEAR THE SCREEN.
C
ISTAT = LIB$ERASE_PAGE (1,1)
RETURN
END

SUBROUTINE GRAPH1(LSTA)

C DEFINE IPACK ARRAY FOR LEGEND
INTEGER*4 IPACK(35)
INTEGER LSTA,COUNT,INC
REAL LSOP(100),JBYS(100),YBO2(100),XPRINT(100),TBCPF(100),
+ TBCP(100)
REAL MAX,MIN,VALMAX,VALMIN,MAXY,MINY,CHOFRAC
CHARACTER*40 L1,L2
COMMON /CHORD/CHOFRACT
DIMENSION YARRY(100),CP1ARRY(100),CP2ARRY(100)
C READ ELEMENTS OF UNIT 11 INTO ARRAYS TO PLOT
OPEN(UNIT=11,FILE='SPWPD.DAT',STATUS='OLD')
COUNT = 0
62 READ(11,*)JBYS(1),YBO2(1),XPRINT(1),TBCPF(1),TBCP(1)
5201 FORMAT(20X,I3,10X,F7.4,8X,F7.4,10X,E12.4,11X,E12.4)
IF (JBYS(1) .EQ. 0) THEN
COUNT = COUNT + 1
ENDIF
IF (COUNT .LT. LSTA) THEN
GO TO 62
ELSE
INC = 2
47 READ(11,*)JBYS(INC),YBO2(INC),XPRINT(INC)
+ ,TBCPF(INC),TBCP(INC)
IF (JBYS(INC) .NE. 0) THEN
INC = INC + 1
GO TO 47
ENDIF
48 ENDIF
CLOSE(UNIT = 11)
C *************** CHECKING OUT DATA INPUT ***************
OPEN(UNIT=32,FILE='PPP.DAT',STATUS='UNKNOWN')
DO 60 I = 1,INC
WRITE (32,*)JBYS(I),YBO2(I),XPRINT(I),TBCPF(I),TBCP(I)
60 CONTINUE
CLOSE (UNIT=32)
C *************** CHECKING OUT DATA INPUT ***************
OPEN(UNIT=32,FILE='PPP.DAT',STATUS='UNKNOWN')
DO I = 1,INC-1

224
READ(32,*),JBYS(I),YBO2(I),XPRINT(I),TBCPF(I),TBCP(I)
XYZ=YBO2(I)
STP=TBCPF(I)
STU=TBCP(I)
YARRY(I)=XYZ
CP1ARRY(I)=STP
CP2ARRY(I)=STU
END
DO
CLOSE (UNIT=32)
CALL MAXMIN(TBCPF,INC,VALMAX,VALMIN)
CALL MAXMIN(TBCP,INC,MMAX,MIN)
CALL MAXMIN(YBO2,INC,MAY,MINY)
C DEFINE AND ASSIGN LEGEND CHARACTER STRINGS
L1 = 'CP - FLAT WING$
L2 = 'CP - CAMBERED WING$
C INITIALIZE THE GRAPHICS SYSTEM
CALL INIT
C LABEL X AND Y AXES USING SELF COUNTING STRINGS
CALL XNAME('Y/B/2$',100)
CALL YNAME('COEFFICIENT OF PRESSURES$',100)
C DEFINE PLOT AREA TO BE 6 INCHES BY 8 INCHES
CALL AREA2D(6.0,8.0)
C DEFINE HEADING LABEL
CALL HEADIN('SPANWISE CP DISTRIBUTION?',-100,1.8,1)
C PLOT ADDITIONAL TICK MARKS
CALL XTIICKS(1)
CALL YTIICKS(1)
C PACK LEGEND LABELS INTO ARRAY IPACK
CALL LINES(L1,IPACK,1)
CALL LINES(L2,IPACK,2)
C SET UP AXIS
CALL GRAF(0.,((MAY-MINY)/5.),MAY+1.0.,((VALMAX
+ -VALMIN)/5.),VALMAX+.1))
C FRAME THE SUBPLOT AREA
CALL FRAME
C PLOT PRESSURE DISTRIBUTION DATA WITH A THICK LINE AND MARKER 15
CALL MARKER(15)
CALL THKCRV(.04)
CALL CURVE(YARRY,CP1ARRY,INC-1,1)
CALL MARKER(16)
CALL RESET('THKCRV')
CALL DASH
CALL CURVE(YARRY,CP2ARRY,INC-1,1)
C
C PLOT MESSAGE
C
CALL MESSAG('CHORDAL FRACTION(X/L) = $',100,1.,8.25)
CALL REALNO(CHOFRACT,3,4.25,8.25)
C CHANGE LEGEND NAME TO "CP DISTRIBUTION"
CALL MYLEGN('CP CURVES?',100)
C PLOT LEGEND
CALL LEGEND(IPACK,2,1.2,6.8)
C END PLOT
CALL ENDPL(0)
C CREATE GRAPHICS METAFILE P1.UIS
CALL METAFL(1)
C TERMINATE PLOT AT END OF PLOTTING SESSION
CALL DONEPL
RETURN
END

SUBROUTINE GRAPH2(NSTA)
C
C DEFINE IPACK ARRAY FOR LEGEND
INTEGER*4 IPACK(35)
INTEGER JBYS(100)
INTEGER NSTA,INC,N
REAL YBO2(100),XPRINT(100),TBCPF(100),TBCP(100)
REAL Y,X,CPF,CPC
REAL MAX,MIN,VALMAX,VALMIN,MAXY,MINY,SPAFRAC
CHARACTER*40 L1,L2
COMMON /SPAN/SPAFRACT
DIMENSION XARRY(100),CP1ARRY(100),CP2ARRY(100)
C READ ELEMENTS OF UNIT 11 INTO ARRAYS TO PLOT
OPEN(UNIT=11,FILE='SPWPD.DAT',STATUS='OLD')
INC = 1
14 READ(11,5201)N,Y,X,CPF,CPC
5201 FORMAT(20X,I3,10X,F7.4,8X,F7.4,10X,E12.4,11X,E12.4)
IF (N.EQ.NSTA) THEN
   JBYS(INC) = N
   YBO2(INC) = Y
   XPRINT(INC) = X
   TBCPF(INC) = CPF
   TBCP(INC) = CPC
   INC = INC + 1
   GO TO 14
ELSEIF(N.EQ.99) THEN
   GO TO 15
ELSE
   GO TO 14
ENDIF
15 PRINT *,
CLOSE(UNIT = 11)
C ********** CHECKING OUT DATA INPUT **********
OPEN(UNIT=33,FILE='PP.DAT',STATUS='UNKNOWN')
DO 60 I = 1,INC-1 
   WRITE (33,5201)JBYS(I),YBO2(I),XPRINT(I),TBCPF(I),TBCP(I)
60 CONTINUE
CLOSE(UNIT = 33)
C ******************************************
OPEN(UNIT=33,FILE='PP.DAT',STATUS='UNKNOWN')
DO I = 1,INC-1 
   READ(33,5201)JBYS(I),YBO2(I),XPRINT(I),TBCPF(I),TBCP(I)
   XYZ=XPRINT(I)
   STP=TBCPF(I)
   STU=TBCP(I)
   XARRY(I)=XYZ
   CP1ARRY(I)=STP
   CP2ARRY(I)=STU
END DO
CLOSE (UNIT = 33)
CALL MAXMIN(TBCPF, INC, VALMAX, VALMIN)
CALL MAXMIN(TBCP, INC, MAX, MIN)
CALL MAXMIN(XPRINT, INC, MAXY, MINY)

C DEFINE AND ASSIGN LEGEND CHARACTER STRINGS
L1 = 'CP - FLAT WINGS'
L2 = 'CP - CAMBERED WINGS'

C INITIALIZE THE GRAPHICS SYSTEM
CALL INIT

C LABEL X AND Y AXES USING SELF COUNTING STRINGS
  CALL XNAME('(X-XLE)/CS',100)
  CALL YNAME('COEFFICIENT OF PRESSURE',100)

C DEFINE PLOT AREA TO BE 6 INCHES BY 8 INCHES
CALL AREA2D(6.0,8.0)

C DEFINE HEADING LABEL
  CALL HEADINC('CHORDWISE CP DISTRIBUTION',-100,1.8,1)

C PLOT ADDITIONAL TICK MARKS
  CALL XTICKS(1)
  CALL YTICKS(1)

C PACK LEGEND LABELS INTO ARRAY IPACK
  CALL LINES(L1,IPACK,1)
  CALL LINES(L2,IPACK,2)

C SET UP AXIS
  CALL GRAF(0.,1.,1.8,(VALMAX-VALMIN)/5.),(VALMAX+.1))

C FRAME THE SUBPLOT AREA
  CALL FRAME

C PLOT PRESSURE DISTRIBUTION DATA WITH A THICK LINE AND MARKER 15
  CALL MARKER(15)
  CALL THKCRV(.04)
  CALL CURVE(XARRY,CP1ARRY,INC-1,1)
  CALL MARKER(16)
  CALL RESET('THKCRV')
  CALL DASH
  CALL CURVE(XARRY,CP2ARRY,INC-1,1)

C PLOT MESSAGE

  CALL MESSAG('SPAN FRACTION(Y/B/2) = $',100,1.8.25)
  CALL REALNO(SPAFRACT,3,4.25,8.25)

C CREATE LEGEND NAME OF "CP CURVES"
  CALL MYLEGN('CP CURVES',100)

C PLOT LEGEND
  CALL LEGEND(IPACK,2,1.2,6.8)

C END PLOT
  CALL ENDPLO()

C CREATE GRAPHICS METAFILE P2.UIS
  CALL METAFL(2)

C TERMINATE PLOT AT END OF PLOTTING SESSION
  CALL DONEPL
  RETURN

SUBROUTINE GRAPH3

C DEFINE IPACK ARRAY FOR LEGEND
INTEGER*4 IPACK(35)
REAL XCL(20),DGF(20),TOTAL(20),RDGF(20),RTOTAL(20)
REAL MAX,MIN,VALMAX,VALMIN
CHARACTER*40 L1,L2
DIMENSION DRAGER(20),TOTA(20)

C READ ELEMENTS OF UNIT 12 INTO ARRAYS TO PLOT
OPEN(UNIT=12,FILE='DRAGPO.DAT',STATUS='OLD')
DO 25 I = 1,20
   READ(12,3529)XCL(I),DGF(I),TOTAL(I),RDGF(I),RTOTAL(I)
3529 FORMAT(5X,F12.6,2X,F12.6,8X,F12.6,15X,F12.6,10X,F12.6)
25 CONTINUE
CLOSE (UNIT = 12)
DO I = 1,20
   XYZ=DGF(I)
   STU=TOTAL(I)
   DRAGER(I)=XYZ
   TOTA(I)=STU
END DO
CALL MAXMIN(DGF,20,VALMAX,VALMIN)
CALL MAXMIN(TOTAL,20,MAX,MIN)

C DEFINE AND ASSIGN LEGEND CHARACTER STRINGS
L1 = 'DRAG POLAR - FLAT WING$
L2 = 'DRAG POLAR - CAMBERED WING$

C INITIALIZE THE GRAPHICS SYSTEM
CALL INIT
C LABEL X AND Y AXES USING SELF COUNTING STRINGS
   CALL XNAME('CD$',100)
   CALL YNAME('CL$',100)

C DEFINE PLOT AREA TO BE 6 INCHES BY 8 INCHES
CALL AREA2D(6.0,8.0)

C DEFINE HEADING LABEL
   CALL HEADIN('DRAG POLARS',-100,2.,1)

C PLOT ADDITIONAL TICK MARKS
   CALL XTICKS(1)
   CALL YTICKS(1)

C PACK LEGEND LABELS INTO ARRAY IPACK
   CALL LINES(L1,IPACK,1)
   CALL LINES(L2,IPACK,2)

C SET UP AXIS
   CALL GRAF(0.0,0.1,.5,0.,((MAX-MIN)/5.),((MAX+.01))

C FRAME THE SUBPLOT AREA
   CALL FRAME

C PLOT PRESSURE DISTRIBUTION DATA WITH A THICK LINE AND MARKER 15
   CALL MARKER(15)
   CALL THKCRV(.04)
   CALL CURVE(XCL,DRAGER,20,1)
   CALL MARKER(16)
   CALL RESET('THKCRV')
   CALL DASH
   CALL CURVE(XCL,TOTA,20,1)

C CHANGE LEGEND NAME TO "CP DISTRIBUTION"
   CALL MYLEGN('DRAG POLAR CURVES$',100)

C PLOT LEGEND
   CALL LEGEND(IPACK,2,1.2,6.5)

C END PLOT
   CALL ENDPL(0)

C CREATE GRAPHICS METAFILE P3.UIS
CALL METAFL(3)
C TERMINATE PLOT AT END OF PLOTTING SESSION
CALL DONEPL
RETURN
END

SUBROUTINE GRAPH4
C
C DEFINE IPACK ARRAY FOR LEGEND
INTEGER*4 IPACK(35)
INTEGER LAFT
REAL XOL(100),XAIRP(100),CLIFTF(100),CLIFT(100)
REAL MAX,MIN,VALMAX,VALMIN
COMMON /PLOT4/LAFT
CHARACTER*40 L1,L2
DIMENSION CL1ARRY(100),CL2ARRY(100)
C READ ELEMENTS OF UNIT 13 INTO ARRAYS TO PLOT
OPEN(UNIT=13,FILE='SWLD.DAT',STATUS='OLD')
DO 25 I = 1,LAFT-1
   READ(13,862)XOL(I),XAIRP(I),CLIFTF(I),CLIFT(I)
862 FORMAT(12X,F8.5,2X,F8.3,3X,F9.6,26X,F9.6)
25 CONTINUE
CLOSE (UNIT = 13)
C CHECKING DATA INPUT
OPEN(UNIT=24,FILE='CHECK.DAT',STATUS='UNKNOWN')
DO 35 I = 1,LAFT-1
   WRITE(24,862)XOL(I),XAIRP(I),CLIFTF(I),CLIFT(I)
35 CONTINUE
CLOSE(UNIT=24)
C
OPEN(UNIT=24,FILE='CHECK.DAT',STATUS='UNKNOWN')
DO I = 1,LAFT-1
   READ(24,862)XOL(I),XAIRP(I),CLIFTF(I),CLIFT(I)
   XYZ=CLIFTF(I)
   STP=CLIFT(I)
   CL1ARRY(I)=XYZ
   CL2ARRY(I)=STP
END DO
CLOSE (UNIT = 24)
CALL MAXMIN(CLIFTF,LAFT-1,VALMAX,VALMIN)
CALL MAXMIN(CLIFT,LAFT-1,MAX,MIN)
C DEFINE AND ASSIGN LEGEND CHARACTER STRINGS
   L1 = 'LIFT FRACTION - FLAT WINGS'
   L2 = 'LIFT FRACTION - CAMBERED WINGS'
C INITIALIZE THE GRAPHICS SYSTEM
CALL INIT
C LABEL X AND Y AXES USING SELF COUNTING STRINGS
   CALL XNAME('X/XMAX$',100)
   CALL YNAME('LIFT FRACTION$',100)
C DEFINE PLOT AREA TO BE 6 INCHES BY 8 INCHES
   CALL AREA2D(6.0,8.0)
C DEFINE HEADING LABEL
   CALL HEADIN('STREAMWISE LIFT DISTRIBUTION$',-100,1.5,1)
C PLOT ADDITIONAL TICK MARKS
   CALL XTICKS(1)
CALL YTICKS(1)

C PACK LEGEND LABELS INTO ARRAY IPACK
   CALL LINES(L1,IPACK,1)
   CALL LINES(L2,IPACK,2)
C SET UP AXIS
   CALL GRAF(0.0,0.2,1.,0.,((VALMAX-VALMIN)/5.), (VALMAX+.005))
C FRAME THE SUBPLOT AREA
   CALL FRAME
C PLOT PRESSURE DISTRIBUTION DATA WITH A THICK LINE AND MARKER 15
   CALL MARKER(15)
   CALL THKCRV(.04)
   CALL CURVE(XOL,CL1ARRY,LAFT-1,1)
   CALL MARKER(16)
   CALL RESET('THKCRV')
   CALL DASH
   CALL CURVE(XOL,CL2ARRY,LAFT-1,1)
C CHANGE LEGEND NAME TO "LIFT FRACTION CURVES"
   CALL MYLEGN('LIFT FRACTION CURVES',100)
C PLOT LEGEND
   CALL LEGEND(IPACK,2,1.2,6.8)
C END PLOT
   CALL ENDPL(0)
C CREATE GRAPHICS METAFILE P4.UIS
   CALL METAFL(4)
C TERMINATE PLOT AT END OF PLOTTING SESSION
   CALL DONEPL
   RETURN
END

SUBROUTINE GRAPH5

C DEFINE IPACK ARRAY FOR LEGEND
   INTEGER*4 IPACK(35)
   INTEGER NR
   REAL YB2(100),SLFF(100),SLIFT(100),SDRAG(100)
   REAL MAX,MIN,VALMAX,VALMIN
   COMMON /PLOT5/NR
   CHARACTER*40 L1,L2
   DIMENSION XL1ARRY(100),XL2ARRY(100)
C READ ELEMENTS OF UNIT 14 INTO ARRAYS TO PLOT
   OPEN(UNIT=14,FILE='SPWLD.DAT',STATUS='OLD')
   DO 45 I = 1,31
      READ(14,865)YB2(I),SLFF(I),SLFF(I),SLIFT(I),SDRAG(I)
      865 FORMAT(17X,F10.6,5X,F11.6,3X,F11.6,10X,F11.6,3X,F11.6)
   CONTINUE
   CLOSE (UNIT = 14)
   DO I = 1,31
      XYZ=SLFF(I)
      STP=SLIFT(I)
      XL1ARRY(I)=XYZ
      XL2ARRY(I)=STP
   END DO
C OPEN(UNIT=25,FILE='CHECKER.DAT',STATUS='UNKNOWN')
C DO 35 I = 1,31
C WRITE(25,865)YB2(I),SLFF(I),SLFF(I),SLIFT(I),SDRAG(I)
CONTINUE

CLOSE(UNIT=25)
CALL MAXMIN(SLFF,31,VALMAX,VALMIN)
CALL MAXMIN(SLIFT,31,MAX,MIN)

DEFINE AND ASSIGN LEGEND CHARACTER STRINGS
L1 = 'LIFT FRACTION - FLAT WINGS'
L2 = 'LIFT FRACTION - CAMBERED WINGS'

INITIALIZE THE GRAPHICS SYSTEM
CALL INIT

LABEL X AND Y AXES USING SELF COUNTING STRINGS
CALL XNAME('Y/B/2$',100)
CALL YNAME('LIFT FRACTION?',100)

DEFINE PLOT AREA TO BE 6 INCHES BY 8 INCHES
CALL AREA2D(6.0,8.0)

DEFINE HEADING LABEL
CALL HEADIN('SPANWISE LIFT DISTRIBUTION?',-100,1.3,1)

PLOT ADDITIONAL TICK MARKS
CALL XTICKS(1)
CALL YTICKS(1)

PACK LEGEND LABELS INTO ARRAY IPACK
CALL LINES(L1,IPACK,1)
CALL LINES(L2,IPACK,2)

SET UP AXIS
CALL GRAF(0.0,0.0,2.1,0.0,((VALMAX-VALMIN)/5.),(VALMAX+.005))

FRAME THE SUBPLOT AREA
CALL FRAME

PLOT PRESSURE DISTRIBUTION DATA WITH A THICK LINE AND MARKER 15
CALL MARKER(15)
CALL THKCRV(.04)
CALL CURVE(YB2,XL1ARRY,31,1)
CALL MARKER(16)
CALL RESET('THKCRV')
CALL DASH
CALL CURVE(YB2,XL2ARRY,31,1)

CHANGE LEGEND NAME TO "LIFT FRACTION CURVES"
CALL MYLEGN('LIFT FRACTION CURVES?',100)

PLOT LEGEND
CALL LEGEND(IPACK,2,1.2,6.8)

END PLOT
CALL ENDPL(0)

CREATE GRAPHICS METAFILE P5.UIS
CALL METAFL(5)

TERMINATE PLOT AT END OF PLOTTING SESSION
CALL DONEPL
RETURN

END

SUBROUTINE MAXMIN(ARRAY,NY,VALMAX,VALMIN)

ARRAY = THE ARRAY WHICH IS BEING SORTED INTO ASCENDING ORDER
NUMBER = THE NUMBER OF ELEMENTS IN THE ARRAY TO BE SORTED
VALMAX = LARGEST VALUE IN THE ARRAY
VALMIN = SMALLEST VALUE IN THE ARRAY
REAL VALMAX,VALMIN
INTEGER NUMBER
LOGICAL SORTED
DIMENSION ARRAY(IOO)
SORTED = .FALSE.
NUMBER = NY
30 IF (.NOT.SORTED) THEN
   SORTED = .TRUE.
   DO 40 I = 1,NUMBER - 1
      IF(ARRAY(I).GT.ARRAY(I+1)) THEN
         VALUE = ARRAY(I)
         ARRAY(I) = ARRAY(I+1)
         ARRAY(I+1) = VALUE
         SORTED = .FALSE.
      ENDIF
   40 CONTINUE
END IF
VALMAX = ARRAY(NUMBER)
VALMIN = ARRAY(1)
RETURN
END

SUBROUTINE P916AF

C TO COMPUTE CL, ETC, FOR FLAT WING AT ARBITRARY
C PLANFORM WITH CP UNKNOWN (CONSTANT DZDX)
C WING DEFINITION IS BY TABLE LOOK-UP, WITH XLE AND
C XTE STORED AS A FUNCTION OF BETAY, USING THE GRID
C SYSTEM OF THE CAMBER SURFACE COMPUTING PROGRAM
DIMENSION TBCP(105,51),TBCPF(105,51)
DIMENSION DCP(51,2),DCPF(51,2),PHI(51,3),PHIF(51,3)
DIMENSION TSUM(51),TSUF(51)
REAL K1,K2,K3
INTEGER LAFT,CONSTANT
DIMENSION TCLIFT(100,2),TSLIFT(51,2)
DIMENSION TSDRAG(51)
COMMON TYB2(51),TZORD(26,51),JBYMAX,NOPCT,RATIO,XLEO,XTEO,
1 TPCT(26),TXLE(50),TXTE(50),DZDX,XMAX,CBAR,TZDX(105,51),XM,NOM,
2 TMAECH(5),TZSKAL,REFAR,SPAN,XO,PI,CNPQD,CPQD,TCNPQD(5),TCQPOD(5)
COMMON FDZDX,XLEOF,TXLEF(50),NFLAP2,NFLAP1,XMREF
COMMON TYPEX,NLEX,NTEX,TBLEX(15),TBLEY(15),TBTEX(15),TBTEY(15)
COMMON IDENT(8)
COMMON /PLOT4/LAFT
COMMON /PLOTS/NR
COMMON /HEALEY/RCL9,RCL9F
XMREFG=XMREF
DO 900 IX=1,100
DO 900 IY=1,51
TBCP(IX,IY)=0.0
900 TBCPF(IX,IY)=0.0
DO 901 NZ=1,100
TCLIFT(NZ,1)=0.0
901 TCLIFT(NZ,2)=0.0
DO 902 IZ=1,51
TSDRAG(IZ)=0.
902 TSLIFT(IZ,1)=0.0
NSF=0

232
REFAR=REFAR/2.0
ALAIR=XMAX
N0NP1=N0N+1
FNON=FLOAT(NON)
WRITE (26,742)
WRITE (62,742)
C PRINT 742
742 FORMAT(2X///40X,39H***INPUT DATA IN AIRPLANE DIMENSIONS***)
WRITE(26,729)IDENT
WRITE(62,729)IDENT
C PRINT 729,IDENT
729 FORMAT(2X/8A10/)
WRITE(26,726)XM,NON,JBYMAX,NOPCT
WRITE(62,726)XM,NON,JBYMAX,NOPCT
C PRINT 726,XM,NON,JBYMAX,NOPCT
726 FORMAT(2X/20X,2HM=F8.4,10X,4HNON=I4,12X,7HJBYMAX=I4,9X,6HNOPCT=I4)
WRITE(26,727)XM,XLEO,XTEO,SPAN,REFAR,CBAR,XMREFO
WRITE(62,727)XM,XLEO,XTEO,SPAN,REFAR,CBAR,XMREFO
C PRINT 727,XM,XLEO,XTEO,SPAN,REFAR,CBAR,XMREFO
727 FORMAT(2X/20X,8HXMAX=F9.4,3X,5HXLEO=F9.4,6X,5HXTEO=F9.4,5X,5HSP$AN=F9.4/20X,8HREFAR/2=F9.4,3X,5HCBAR=F9.4,6X,6HXMREF=F9.4)
WRITE (26,728)
WRITE (62,728)
C PRINT 728
728 FORMAT(2X///50X,4HTPCT,22X,4HTYB2/) IF (JBYMAX .LT. NOPCT) GO TO 731
KLAST=NOPCT
GO TO 732
731 KLAST=JBYMAX
732 DO 750 KPRINT=1,KLAST
C PRINT 747,KPRINT,TPCT(KPRINT),KPRINT,TYB2(KPRINT)
750 WRITE (26,747)KPRINT,TPCT(KPRINT),KPRINT,TYB2(KPRINT)
WRITE (62,747)KPRINT,TPCT(KPRINT),KPRINT,TYB2(KPRINT)
747 FORMAT(35X,I5,5X,F11.5,4X,I5,5X,F11.5)
KLAST=KLAST+1
IF (JBYMAX .EQ. NOPCT) GO TO 736
IF (JBYMAX .LT. NOPCT) GO TO 734
DO 735 KPRINT=KLAST,JBYMAX
C PRINT 745,KPRINT,TYB2(KPRINT)
735 WRITE (26,745)KPRINT,TYB2(KPRINT)
WRITE (62,745)KPRINT,TYB2(KPRINT)
745 FORMAT(62X,I5,4X,F11.5)
GO TO 736
734 DO 737 KPRINT=KLAST,NOPCT
C PRINT 746,KPRINT,TPCT(KPRINT)
737 WRITE (26,746)KPRINT,TPCT(KPRINT)
WRITE (62,746)KPRINT,TPCT(KPRINT)
746 FORMAT(35X,I5,5X,F11.5)
736 CONTINUE
610 IF(TYPEX)612,602,612
612 WRITE(26,613)
WRITE(62,613)
C PRINT 613
613 FORMAT(2X///50X,5HTBLEX,22X,5HTBLEY/)
614 FORMAT(45X,F11.5,16X,F11.5)
615 FORMAT(2X///50X,5HTBTEX,22X,5HTBTEY/)
DO 617 I=1,NLEX
C PRINT 614,TBLEX(I),TBLEY(I)
617 WRITE(26,614)TBLEX(I),TBLEY(I)
WRITE(62,614)TBLEX(I),TBLEY(I)
WRITE(26,615)
WRITE(62,615)
C PRINT 615
DO 618 I=1,NTEX
C PRINT 614,TBTEX(I),TBTEY(I)
618 WRITE(26,614)TBTEX(I),TBTEY(I)
WRITE(62,614)TBTEX(I),TBTEY(I)
KC=1
KB=1
YACC =0.0
YDEL= SPAN/(2.*(FLOAT(NON)))
DO 660 N=1,NON
YACC=YACC + YDEL
630 DELXL = TBLEX(KB +1) - TBLEX(KB)
DELYL= TBLEY(KB+1)-TBLEY(KB)
IF(TBLEY(KB+1)-YACC)634,644,644
634 IF(N-NON)636,638,638
636 KB=KB+1
637 GO TO 630
638 TXLE(N) = TBLEX(NLEX)
TXTE(N) = TBTEX(NTEX)
GO TO 660
644 TXLE(N) =TBLEX(KB)+(DELXL/DELYL)*(YACC-TBLEY(KB))
646 DELXT = TBTEX(KC+1) - TBTEX(KC)
DELYT = TBTEY(KC+1) -TBTEY(KC)
IF(TBTEY(KC+1)-YACC)650,652,652
650 KC = KC+1
GO TO 646
652 TXTE(N) = TBTEX(KC)+(DELXT/DELYT)*(YACC-TBTEY(KC))
660 CONTINUE
602 CONTINUE
WRITE (26,752)
WRITE (62,752)
C PRINT 752
752 FORMAT(2X//34X,1HN15X,4HTXLE23X,4HTXTE/)
DO 760 J=1,NON
C PRINT 753,J,TXLE(J),TXTE(J)
760 WRITE (26,753) J,TXLE(J),TXTE(J)
WRITE (62,753) J,TXLE(J),TXTE(J)
753 FORMAT(30X,I5,11X,F11.6,16X,F11.6)
WRITE (26,769)
C PRINT 769
769 FORMAT(2X//20X,24HTABLE OF ORDINATES,TZORD//20X,4HNOTE/23X,81HFOR $ EACH PERCENT CHORD VALUE PRINTED BELOW , THE TABLE OF ORDINATES W $HICH FOLLOWS/20X,67HCORRESPONDS TO SPAN POSITIONS GIVEN IN THE TA $BLE OF SPAN-FRACTIONS//53X,31HTABLE OF SPAN-FRACTIONS , Y/B/2)
WRITE(26,770)(TYB2(J),J=1,JBYMAX)
C PRINT 770,(TYB2(J),J=1,JBYMAX)
WRITE(26,774)
C PRINT 774
774 FORMAT(2X//5X,13HPERCENT CHORD,45X,11HZ-ORDINATES)
DO 771 J=1,NOPCT
WRITE(26,772)TPCT(J)
C PRINT 772,TPCT(J)
772 FORMAT(2X,/,8X,F7.3)
C PRINT 770,(TZORD(J,K),K=1,JBYMAX)
771 WRITE (26,770) (TZORD(J,K),K=1,JBYMAX)
770 FORMAT(19X,10(1X,F9.4))
NR=NON+100
DZDXF=-.01746
MUPR=NOM+l
DO 790 JDO=1,MUPR
BETA=SQRT(XM**2-1.0)
SF=FLOAT(NON)/(BETA*SPAN/2.0)
IF(TZSKAL)1113,1112,1113
1113 RESKAL=TZSKAL
TZSKAL=0
GO TO 1111
1112 IF(JDO-1)704,705,704
704 BETPRE=BETA
XM=TMACH(JDO-1)
CNPOD=TCNPOD(JDO-1)
CAPOD=TCAPOD(JDO-1)
WRITE(26,706)XM
C PRINT 706, XM
706 FORMAT(1H1,40X,30H***WING RESCALED FOR MACH NO. F8.5///)
BETA=SQRT(XM**2-1.0)
SF=FLOAT(NON)/(BETA*SPAN/2.0)
RESKAL=BETPRE/BETA
RATIO=RATIO/RESKAL
XLEO=XLEO*RESKAL
XTEO=XTEO*RESKAL
XMAX=XMAX*RESKAL
CBAR=CBAR*RESKAL
XMREF=XMREF*RESKAL
DO 720 JSKAL=1,NON
TXLE(JSKAL)=TXLE(JSKAL)*RESKAL
720 TXTE(JSKAL)=TXTE(JSKAL)*RESKAL
1111 DO 730 KSAL=1, JBYMAX
DO 730 NSKAL=1,NOPCT
730 TZORD(NSKAL,KSAL)=TZORD(NSKAL,KSAL)*RESKAL
705 IF (NSF .NE. 0) GO TO 739
DO 60 KSFR=1,NON
TXLE(KSF)=TXLE(KSF)*SF
60 TXTE(KSF)=TXTE(KSF)*SF
DO 61 KSFC=1, JBYMAX
DO 61 KSFR=1,NOPCT
61 TZORD(KSFR,KSFC)=TZORD(KSFR,KSFC)*SF
CBAR=CBAR*SF
XLEO=XLEO*SF
XTEO=XTEO*SF
XMAX=XMAX*SF
XMREF=XMREF*SF
NSF=1
739 WRITE (26,62)
C PRINT 62
62 FORMAT(2X///40X,38H***INPUT DATA IN PROGRAM DIMENSIONS***)
235
WRITE(26,729) IDENT
C PRINT 729, IDENT
WRITE(26,726) XM, NON, JBYMAX, NOPCT
C PRINT 726, XM, NON, JBYMAX, NOPCT
WRITE(26,63) XMAX, XLEO, XTEO, NON, CBAR, XMREF
C PRINT 63, XMAX, XLEO, XTEO, NON, CBAR, XMREF
63 FORMAT(2X/20X,8HXMAX = F9.4, 3X, 5HXLEO = F9.4, 6X, 5HXTEO = F9.4, 5X, 5HS$AN = 15/40X, 5HCBAR = F9.4, 6X, 6HXMRREF = F9.4)
WRITE(26,752)
C PRINT 752
C PRINT 753, JSF, TXLE(JSF), TXTE(JSF)
64 WRITE (26,753) JSF, TXLE(JSF), TXTE(JSF)
WRITE (26,769)
C PRINT 769
WRITE(26,770)(TYB2(J), J=1, JBYMAX)
C PRINT 770, (TYB2(J), J=1, JBYMAX)
WRITE(26,774)
C PRINT 774
DO 77 JSF=1, NON
WRITE(26,753) JSF, TXLE(JSF), TXTE(JSF)
77 CONTINUE
C IF (XMAX .LE. 100.) GO TO 4001
WRITE (26,4000) XMAX, XM
C PRINT 4000, XMAX, XM
4000 FORMAT (/68H SORRY XMAX CAN NOT EXCEED 100. PROGRAM WILL CONTINUE 1 TO NEXT CASE. /15X, 5HXM = E16.8, 10X, 5HMCH = E16.8//)
GO TO 790
4001 CALL SLOPE
IF (FDZDX .EQ. 0.0) GO TO 7373
KI=0
70 IF(NFLAP1)73,71,73
71 XLEOF=XLEOF*SF
LEOF = INT(XLEOF + 1.0)
LTEOF5 = INT(XTEO+5.0)
DO 72 L=LEOF,LTEOF5
72 TDZDX(L,1) = FDZDX
JFLAPS = 2
JFLAP = 1
GO TO 74
73 JFLAPS = NFLAP1 + 1
JFLAP = NFLAP1
74 DO 77 I = JFLAP, NFLAP2
KI=KI+1
XLEF=TXLEF(KI)*SF
LEF = INT(XLEF + 1.0)
XTE=TXTE(I)
LTEF5 = INT(XTE + 5.0)
DO 76 L = LEF, LTEF5
76 TDZDX(L,JFLAPS)= FDZDX
JFLAPS = JFLAPS + 1
77 CONTINUE
78 FORMAT(1H0///, 5X, 34HFLAP OPTION INCLUDED, FLAP SLOPE =F11.6//)
WRITE(26,78)FDZDX

LMAX=INT(XMAX+1.)
DRAG=0
ALIFT=0
PMOM=0
ALIFTF=0.
PMOMF=0.
DFOC=0.
BETA=SQRT(XM**2-1.0)
B4=4.0/BETA
PI1=1.0/PI
NL=200-NR

C
XMU=0.5
DO 5024 IY=1,51
TSUM(IY)=0.
5024 TSUMF(IY)=0.
C
WRITE(26,5202)
5202 FORMAT(2X///37X,46H***CALCULATED LIFTING PRESSURE DISTRIBUTION***)
C
WRITE(26,729)IDENT
C
WRITE(26,726)XM,NON,JBYMAX,NOPCT
9 DO 110 LSOP1=1,LMAX
C
LSTAR=LSOP1
DO 5000 LSOP2=1,2
IF(LSOP2.EQ.2)LSTAR=LSOP1+1
IF(LSOP2.EQ.1)GO TO 10
DO 5025 IY=1,51
TSUM (IY)=0.
5025 TSUMF(IY)=0.
10 DO 100 NSTAR=100,NR
JBY=NSTAR-100
IF(JBY)12,11,12
11 XSTE=XTEO
XSLE=XLEO
GO TO 13
12 XSLE=TXLE(JBY)
XSTE=TXTE(JBY)
13 LSLE=INT(XSLE+1.0)
IF(LSLE-LSTAR)15,15,100
15 LSTE=INT(XSTE+1.0)
LSTE4=LSTE
IF(LSTAR-LSTE4)17,17,100
17 SUM=0.
SUMF=0.
IF(LSOP2.EQ.2)GO TO 5026
SUM =TSUM (JBY+1)
SUMF=TSUMF(JBY+1)
5026 CONTINUE
IF(LSTAR-1)18,56,18
18 DO 55 N=NL,NR
NDELTN=NSTAR-N
NDIFF=IABS(NDELTN)
LMACH=LSTAR-NDIFF
JBY=IABS(N-100)
IF(JBY)38,37,38
37 XLE=XLEO
XTE=XTEO
GO TO 39
38 XLE=TXLE(JBY)
XTE=TXTE(JBY)
39 LLE=INT(XLE+1.0)
LTE=INT(XTE+1.)
   IF(LLE-LMACH)45,45,55
45 DELTN1=FLOAT(NDELTN)+.5
DELTN2=DELTN1-1.0
LAST=LMACH
   IF(LTE.LE.LMACH)LAST=LTE
   LSTART=LLE
   IF(LSOP2.EQ.2)GO TO 5027
   LSTART=LSTAR-1
   IF(LSTART.GT.LAST)GO TO 55
5027 CONTINUE
   DO 54 LVAR=LSTART,LAST
   NS1=LSTAR-LVAR
   DELTL=FLOAT(NS1)+.5
   SQDL=DELTL**2
   TERM1=(SQRT(SQDL-DELTN1**2))/DELTN1
   TERM2=(SQRT(SQDL-DELTN2**2))/DELTN2
   R=(TERM2-TERM1)/DELTL
   IF(LLE.EQ.LTE)GO TO 5021
   IF(LVAR.EQ.LLE)GO TO 48
   IF(LVAR.EQ.LTE)GO TO 5022
   GO TO 51
5021 R=R*(XTE-XLE)
   GO TO 51
5022 ATE=XTE-FLOAT(LTE-1)
   R=R*ATE
   GO TO 51
48 ALE=FLOAT(LLE)-XLE
   R=ALE*R
51 JCP=JBY+1
   CPF=TBCPF(LVAR,JCP)
   SUMF=SUMF+R*CPF
   CP=TBCP(LVAR,JCP)
   SUM=SUM+R*CP
   IF(LSOP2.EQ.1)GO TO 54
   IF(LVAR.GT.(LSTAR-2))GO TO 54
   TSUM (JBYS+1)=TSUM (JBYS+1)+R*CP
   TSUMF(JBYS+1)=TSUMF(JBYS+1)+R*CPF
54 CONTINUE
55 CONTINUE
56 JCP=JBYS+1
   DZDX=TDZDX(LSTAR,JCP)
   CPAFT=-B4*DZDX+PI1*SUM
   CPAFTF=-B4*DZDXF+PI1*SUMF
   DCP(JCP,LSOP2)=CPAFT
   DCPF(JCP,LSOP2)=CPAFTF
   TBCP(LSTAR,JCP)=CPAFT
   TBCPF(LSTAR,JCP)=CPAFTF
100 CONTINUE
5000 CONTINUE
C
WRITE(26,5200)LSOP1
5200 FORMAT(2X//15X,6HLSSTAR=I4/5X,5HNSTAR,10X,5HY/B/2,9X,9H(X-XLE)/C,7
$X,13HCP(FLAT WING),17X,17HCP(CAMBERED WING))
5008 CONTINUE
C
DO 5020 NSTAR=100,NRC
JBYS=NSTAR-100
YB02=FLOAT(JBYS)/FNON
JCPS=JBYS+1
LSTAR=LSOP1
IF(JBYS)5002,5001,5002
5001 XSLE=XLEO
XSTE=XTEO
GO TO 5003
5002 XSLE=TXLE(JBYS)
XSTE=TXTE(JBYS)
5003 LSL=INT(XSLE+1.)
LSTE=INT(XSTE+1.)
IF(LSTAR.LT.LSLE)GO TO 5020
IF(LSTAR.GT.LSTE)GO TO 5020
IF(LSTAR.EQ.LSTE)GO TO 5023
IF(LSTAR.EQ.LSLE)GO TO 5005
5005 PHI(JCPS,3)=0.0
PHIF(JCPS,3)=0.0
A1=FLOAT(LSTAR)-XSLE
5006 A1=1.0
GO TO 5007
5007 CONTINUE
A2=1.
PHI(JCPS,1)=PHI(JCPS,3)-.25*DCP(JCPS,1)*A1
PHIF(JCPS,1)=PHIF(JCPS,3)-.25*DCPF(JCPS,1)*A1
PHI(JCPS,2)=PHI(JCPS,1)-.25*DCP(JCPS,2)*A2
PHIF(JCPS,2)=PHIF(JCPS,1)-.25*DCPF(JCPS,2)*A2
ABSA1=ABS(1.-A1)
IF(ABSA1.GT.0.001)GO TO 5010
K1=XMU
K2=.5*(1.-XMU)
K3=K2
GO TO 5011
5010 K1=XMU
K3=(1.-XMU)/(A1+1.)
K2=A1*K3
5011 PHI3=K2*PHI(JCPS,2)+K1*PHI(JCPS,1)+K3*PHI(JCPS,3)
PHIF3=K2*PHIF(JCPS,2)+K1*PHIF(JCPS,1)+K3*PHIF(JCPS,3)
TBCP(LSTAR,JCPS)=-4.0*(PHI3-PHI(JCPS,3))/A1
TBCPF(LSTAR,JCPS)=-4.0*(PHIF3-PHIF(JCPS,3))/A1
C
5023 CONTINUE
CHORD=XSTE-XSLE
IF(CHORD.LT.0.001)GO TO 5009
IF(LSTAR.EQ.LSTE)GO TO 5009
XPRINT=(FLOAT(LSTAR)-XSLE)/CHORD
GO TO 5012
5009  XPRINT=1.0
5012  CONTINUE
     WRITE(26,5201)JBYS,YBO2,XPRINT,TBCPF(LSTAR,JCPS),TBCP(LSTAR,JCPS)
     WRITE(11,5201)JBYS,YBO2,XPRINT,TBCPF(LSTAR,JCPS),TBCP(LSTAR,JCPS)
5201  FORMAT(20X,I3,10X,F7.4,8X,F7.4,10X,E12.4,11X,E12.4)
     IF(LSTAR.EQ.LSTE)GO TO 5020
     C
     PHI(JCPS,3)=PHI3
     PHIF(JCPS,3)=PHI3F
     5020  CONTINUE
     110  CONTINUE
     AREA9=0.
     ALIFT9=0
     DRAG9=0
     PMOM9=0
     DFOC9=0.
     ALFF9=0.
     PMOM9F=0.
     LAFT=LMAX
     DO 355 LIFT=1,LAFT
     TCLIFT(LIFT,1)=0.
     355  TCLIFT(LIFT,2)=0.
     DO 400 N=100,NR
     JBY=N-100
     JCP=JBY+1
     IF(JBY)301,300,301
     300  XLE=XLEO
     XTE=XTEO
     GO TO 305
     301  XLE=TXLE(JBY)
     XTE=TXTE(JBY)
     305  LLE=INT(XLE+1.0)
     LTE=INT(XTE+1.0)
     ALE=FLOAT(LLE)-XLE+.5
     TSLIFT(JCP,1)=0.
     TSLIFT(JCP,2)=0.
     2001  FORMAT(1HO//)
     KWIT=0
     DO 370 LSTAR=LLE,LTE
        **********
        *** FORCE INTEGRATION USING ***
        ***CP=.75*TBCP(LSTAR,JCP) + .25*TBCP(LSTAR+1,JCP)***
        ***DZDX=.75*TDZDX(LSTAR) + .25*TDZDX(LSTAR-1)***
        ***CP AND DZDX CONSTANT BETWEEN BLOCK CENTERS ***
        **********
        IF(KWIT.NE.0)GO TO 370
        IF(LLE.EQ.LTE)GO TO 407
        IF(LLE.EQ.(LTE-1))GO TO 408
        IF(LSTAR.EQ.LLE)GO TO 410
        IF(LSTAR.GE.(LTE-1))GO TO 411
        GO TO 413
        407  ABLOCK=XTE-XLE
        CP9 =TBCP (LSTAR,JCP)
        CP9F=TBCPF(LSTAR,JCP)
        XLS=.5*(XLE + XTE)
        DZDX=TDZDX(LSTAR,JCP)
GO TO 414
408 XCHECK=FLOAT(LTE)-.5
IF(XTE.LE.XCHECK)GO TO 415
IF(LSTAR.EQ.LLE)GO TO 410
409 ABLOCK=XTE-FLOAT(LTE)+.5
XLS=.5*(XTE+XCHECK)
GO TO 414
410 ABLOCK=ALE
CP9 = 75*TBCP (LSTAR,JCP) + .25*TBCP (LSTAR+1,JCP)
CP9F = 75*TBCP F(LSTAR,JCP) + .25*TBCP F(LSTAR+1,JCP)
XLS = .5*(FLOAT(LLE)+XLE)
DZDX=TDZDX(LSTAR,JCP)
GO TO 325
411 XCHECK=FLOAT(LTE)-.5
IF(XTE.LE.XCHECK)GO TO 412
IF(LSTAR.EQ.LTE)GO TO 416
GO TO 413
412 ABLOCK=XTE-FLOAT(LTE)+1.5
CP9 = 75*TBCP (LSTAR,JCP) + .25*TBCP (LSTAR+1,JCP)
CP9F = 75*TBCP F(LSTAR,JCP) + .25*TBCP F(LSTAR+1,JCP)
DZDX=TDZDX(LSTAR,JCP)+.25*TDZDX(LSTAR-1,JCP)
XLS=0.5*(XTE+FLOAT(LTE)-1.5)
GO TO 414
413 ABLOCK=1.0
CP9 = 75*TBCP (LSTAR,JCP) + .25*TBCP (LSTAR+1,JCP)
CP9F = 75*TBCP F(LSTAR,JCP) + .25*TBCP F(LSTAR+1,JCP)
DZDX=TDZDX(LSTAR,JCP)+.25*TDZDX(LSTAR-1,JCP)
XLS=FLOAT(LSTAR)
GO TO 325
415 ABLOCK=XTE-XLE
CP9 = 75*TBCP (LSTAR,JCP) + .25*TBCP (LSTAR+1,JCP)
CP9F = 75*TBCP F(LSTAR,JCP) + .25*TBCP F(LSTAR+1,JCP)
DZDX=TDZDX(LSTAR,JCP)+.25*TDZDX(LSTAR-1,JCP)
XLS=0.5*(XLE+XTE)
GO TO 414
416 ABLOCK=XTE-FLOAT(LTE)+.5
XLS=.5*(XTE+XCHECK)
GO TO 414
414 KWIT=1
325 IF(JBY.EQ.0)ABLOCK=ABLOCK*.5
IF(N.EQ.NR)ABLOCK=ABLOCK*.5
AREA9=AREA9+ABLOCK
340 CONTINUE
FORCE=CP9*ABLOCK
FORCEF=CP9F*ABLOCK
ALIFT9=ALIFT9+FORCE
ALFF9=ALFF9+FORCEF
DRAG9=DRAG9-FORCEF*DZDX
DFOC9=DFOC9-FORCEF*DZDX
PMOM9=PMOM9-FORCEF*XLS
PMOM9F=PMOM9F-FORCEF*XLS
TCLIFT(LSTAR,1)=TCLIFT(LSTAR,1)+FORCE
TCLIFT(LSTAR,2)=TCLIFT(LSTAR,2)+FORCEF
TSDRAG(JCP)=TSDRAG(JCP)-FORCE*DZDX
TSLIFT(JCP,1)=TSLIFT(JCP,1)+FORCE
TSLIFT(JCP,2)=TSLIFT(JCP,2)+FORCEF
370 CONTINUE
400 CONTINUE
   SAREA9=AREA9/BETA
   WRITE(26,3511)
   WRITE(62,3511)
C PRINT 3511
3511 FORMAT(2X///,30X,61H*****CALCULATED WING OVERALL AERODYNAMIC CHARA
$TERISTICS*****/) WRITE(26,729) IDENT
WRITE(62,729) IDENT
C PRINT 729, IDENT
WRITE(26,726) XM, NON, JBYMAX, NOPCT
WRITE(62,726) XM, NON, JBYMAX, NOPCT
C PRINT 726, XM, NON, JBYMAX, NOPCT
WRITE(26,3543)
WRITE(62,3543)
C PRINT 3543
3543 FORMAT(2X///,30X,12HPROGRAM AREA, 35X, 14HREFERENCE AREA)
WRITE(26,3544)
WRITE(62,3544)
C PRINT 3544
3544 FORMAT(2X/23X,9HFLAT WING, 10X, 13HCAMBERED WING, 14X, 9HFLAT WING, 12X, 13HCAMBERED WING)
3545 FORMAT(1X/12X, 2HCL, 1X, E17.8, 6X, E17.8, 6X, E17.8, 8X, E17.8)
3547 FORMAT(1X/12X, 2HC, 1X, E17.8, 6X, E17.8, 6X, E17.8, 8X, E17.8)
3549 FORMAT(1X/12X, 2HC, 1X, E17.8, 6X, E17.8, 6X, E17.8, 8X, E17.8)
3551 FORMAT(1X/12X, 4HAREA, 14X, E17.8, 32X, E17.8)
CHANGE = SAREA9/(SF**2*REFAR)
CL9= ALIFT9/AREA9
CD9= DRAG9/AREA9
CMAP9= PMOM9/(AREA9 * CBAR)
CL9F= ALFF9/AREA9
CD9F= -CL9F * DZDXF
CMAP9F = PMOM9F/(AREA9*CBAR)
RCMP9F = CMAP9F * CHANGE
RCMAP9 = CMAP9 * CHANGE
RCL9= CL9 * CHANGE
RCD9= CD9 * CHANGE
RCL9F = CL9F * CHANGE
RCD9F = CD9F * CHANGE
WRITE(26,3545)CL9F, CL9, RCL9F, RCL9
WRITE(62,3545)CL9F, CL9, RCL9F, RCL9
C PRINT 3545, CL9F, CL9, RCL9F, RCL9
WRITE(26,3547) CD9F, CD9, RCD9F, RCD9
WRITE(62,3547) CD9F, CD9, RCD9F, RCD9
C PRINT 3547, CD9F, CD9, RCD9F, RCD9
WRITE(26,3549) CMAP9F, CMAP9, RCMP9F, RCMP9
WRITE(62,3549) CMAP9F, CMAP9, RCMP9F, RCMP9
C PRINT 3549, CMAP9F, CMAP9, RCMP9F, RCMP9
WRITE(26,3551) SAREA9, REFAR
WRITE(62,3551) SAREA9, REFAR
C PRINT 3551, SAREA9, REFAR
CDFOC9 = DFOC9/AREA9
CDCOF9 = -CL9 * DZDXF
CDINT = (CDFOC9 +CDFOC9)/CL9F
CDOCL2 = -DZDXF/CL9F

242
RCDCL2 = CDOCL2/CHANGE
XCL = -.02

3525 FORMAT(2X//12X,18HPOLAR,PROGRAM AREA,7X,4HCD =F10.6,3H + F10.6,6H( $ CL -F10.6,3H) +F10.6,6H( CL -F10.6,4H)**2)
WRITE(26,3525)CD9,CDINT,CL9,CDOCL2,CL9
WRITE(62,3525)CD9,CDINT,CL9,CDOCL2,CL9

C PRINT 3525,CD9,CDINT,CL9,CDOCL2,CL9

3526 FORMAT(2X//12X,20HPOLAR,REFERENCE AREA5X,4HCD =F10.6,3H + F10.6,6H( $ CL -F10.6,3H) +F10.6,6H( CL -F10.6,4H)**2)
C PRINT 3526,RC9,CDINT,RCL9,RCDC2,RCL9
WRITE (26,3526)RC9,CDINT,RCL9,RCDC2,RCL9
WRITE (62,3526)RC9,CDINT,RCL9,RCDC2,RCL9
WRITE(26,3543)
WRITE(62,3543)
C PRINT 3543
WRITE(26,3528)
WRITE(62,3528)
C PRINT 3528

3528 FORMAT(2X//12X,2HCL,7X,12HCD,FLAT WING,6X,16HCD,CAMBERED WING,13X, $12HCD,FLAT WING,8X,16HCD,CAMBERED WING)
DO 3530 KCL = 1,20
XCL = XCL +.02
DELTCL = XCL - CL9
DELC2 = DELTCL **2
XINT = CDINT * DELTCL
XFLAT = CDOCL2 * DELCL2
TOTAL = CD9 + XINT + XFLAT
DGF = CDOCL2 * (XCL**2)
RDELCL = XCL - RCL9
RDCL2 = RDELCL **2
RXINT = CDINT * RDELCL
RXFLAT = RCDC2 * RDCL2
RTOTAL = RCD9 + RXINT + RXFLAT
RDGF = RCDC2 *(XCL**2)

3529 FORMAT(5X,F12.6,2X,F12.6,8X,F12.6,15X,F12.6,10X,F12.6)
C PRINT 3529,XCL,DGF,TOTAL,RDFD,RTOTAL
WRITE (12,3529)XCL,DGF,TOTAL,RDFD,RTOTAL
WRITE (62,3529)XCL,DGF,TOTAL,RDFD,RTOTAL

3530 WRITE (26,3529)XCL,DGF,TOTAL,RDFD,RTOTAL

3532 FORMAT(2X//12X,39HTRANSFORMED POLAR,REFERENCE AREA, CD =F10.6, $3H + .F10.6,6H (CL =F10.6,4H)**2)
CONSTANT = 99
SST = 0.0000
STT = 0.0000
STU = 0.0000E-00
STV = 0.0000E-00
WRITE(11,5201)CONSTANT,SST,STT,STU,STV
CLOSE(UNIT=11)
CLOSE(UNIT=12)
RCLMNT = RCL9 - (CDINT/(2.*RCDC2))
RCDMNT = RCD9 - (CDINT**2)/(4.*RCDCL2)
WRITE (26,3532)RCLMNT,RCDC2,RCLMNT
WRITE (62,3532)RCDMNT,RCDC2,RCLMNT
C PRINT 3532,RCDMNT,RCDC2,RCLMNT
XM=PRIM=XMREF/CBAR
X1XMP=-RCMP9F/RCL9F

243
X2XMP = X1XMP - XMPRIM
CMO = RCMAP9 + (X1XMP * RCL9)
WRITE(26,3400) XMREFO, CMO, X2XMP
WRITE(62,3400) XMREFO, CMO, X2XMP
C
PRINT 3400, XMREFO, CMO, X2XMP
3400 FORMAT(2X/12X, 28HMOMENT COEFFICIENT ABOUT X = , F10.5, 11H, CM = , $, F12.6, 3H - , F12.6, 5H \* CL)
WRITE(26, 3401) RCL9, RCL9F
WRITE(62, 3401) RCL9, RCL9F
C
PRINT 3401, RCL9, RCL9F
3401 FORMAT(2X/12X, 18HLIFT CURVE, CL = , F10.5, 3H + , F10.5, 8H \* ALPHA)
IF(CNPOD) 880, 890, 880
880
RCDMNP = RCD9 + CAPOD
RCINTP = (CDINT + CNPOD * RCDCL2) / (1. - CAPOD \* RCDCL2)
RCDC2P = RCDCL2 / ((1. - CAPOD \* RCDCL2)**2)
3533 FORMAT(1HO/, 5X, 52HPOLAR INCLUDING POD INTERFERENCE EFFECTS (REF. AREA))
WRITE (26, 3533)
WRITE (62, 3533)
CLOINT = CNPOD + RCL9
3534 FORMAT(1HO/10X, 4HCD = F10.6, 2H + F10.6, 6H (CL - F10.6, 4H) + F10.6, 6H 1(CL - F10.6, 4H)**2)
3535 FORMAT(1HO/10X, 7HCNPOD = F12.6, 15X, 7HCAPOD = F12.6)
WRITE (26, 3535) CNPOD, CAPOD
WRITE (26, 3536)
WRITE (62, 3535) CNPOD, CAPOD
WRITE (26, 3536)
WRITE (62, 3536)
3536 FORMAT(1HO/10X, 15X, 2HCD)
XCL = -.02
DO 888 KCLI = 1, 20
XCL = XCL + .02
DCLINT = XCL - CLOINT
DCLINT2 = DCLINT**2
CDPINT = RCDMNP + (RCINTP * DCLINT) + (RCDC2P * DCLINT2)
3537 FORMAT(8X, F12.6, 5X, F12.6)
WRITE (62, 3537) XCL, CDPINT
888 WRITE (26, 3537) XCL, CDPINT
3538 FORMAT(///5X, 64HTRANSFORMED POLAR INCLUDING POD INTERFERENCE EFFECT 1S (REF. AREA))
WRITE(26, 3538)
WRITE(62, 3538)
CDMIN = RCD9 + CAPOD - (1. / (4. * RCDCL2)) * ((CDINT + CNPOD * RCDCL2)**2)
CKINT = RCDCL2 / ((1. - CAPOD * RCDCL2)**2)
CLCDMN = RCL9 + CNPOD - (1. / (2. * RCDCL2)) * (1. - CAPOD * RCDCL2) * (CDINT + (CNPOD 1* RCDCL2))
WRITE (26, 3532) CDMIN, CKINT, CLCDMN
WRITE (62, 3532) CDMIN, CKINT, CLCDMN
C
C
***WRITE STREAMWISE LIFT DISTRIBUTION**
C
890 WRITE (26, 860)
WRITE (62, 860)
860 FORMAT(2X//46X, 28HSTREAMWISE LIFT DISTRIBUTION, //39X, 9HFLAT WING, 
$25X, 13HCAMBERED WING, //36X, 4HLIFT, 31X, 4HLIFT

244
.

$25X, 6HCAMBER, /15X, 6HX/XMAX, 4X, 6X + XO, 3X, 8HFRACTION, 5X, $9HSUMMATION, 13X, 8HFRACTION, 5X, 9HSUMMATION, 10X, 4HAREA)
SUML=0.0
SOF=0.0
KWIT=0
DO 870 LEN=1, LAFT
IF(KWIT.EQ.1) GO TO 870
XOL=FLOAT(LEN)/XMAX
XOLM=(FLOAT(LEN)+.5)/XMAX
IF(XOLM.LT.1.0) GO TO 871
KWIT=1
XOLM=1.
IF(XOL.GT.1.) XOL=1.
871 CONTINUE
XAIRP=XO+XOL*ALAIR
XAIRPm=XO+XOLm*ALAIR
IF(ALIFT9.EQ.0.) CLIFT=TCLIFT(LEN,1)
IF(ALIFT9.EQ.0.) GO TO 852
CLIFT=TCLIFT(LEN,1)/ALIFT9
852 IF(ALFF9.EQ.0.) CLIFF=1
IF(ALFF9.EQ.0.) GO TO 854
CLIFF=TCLIFT(LEN,2)/ALFF9
854
SUML=SUML+CLIFT
SOF=SOF+CLIFF
CAMAREA=BETA/2.0*RCL9*2.0*REFAR*(SUML-SOF)
WRITE(62,862) XOL, XAIRP, CLIFT, CLIFT
WRITE(62,8625) XOLM, XAIRPM, SOF, SUML, CAMAREA
WRITE(26,862) XOL, XAIRF, CLIFT, CLIFT
WRITE(13,862) XOL, XAIRP, CLIFT, CLIFT
870 WRITE(26,8625) XOLM, XAIRPM, SOF, SUML, CAMAREA
862 FORMAT(12X,F8.5,2X,F8.3,3X,F9.6,26X,F9.6)
8625 FORMAT(12X,F8.5,2X,F8.3,16X,F9.6,26X,F9.6,F11.6)
CLOSE(UNIT=13)

***WRITE SPANWISE LIFT DISTRIBUTION***

WRITE (62,863)
WRITE (26,863)
863 FORMAT(2X//42X,36HSPANWISE LIFT AND DRAG DISTRIBUTIONS///,39X, $9HFLAT WING, 25X, 13HCAMBERED WING, /, 36X, 4HLIFT, 10X, 4HDRAG, 17X, $4HLIFT10X, 4HDRAG, /, 20X, 5HY/B/2, 9X, 8HFRACTION, 6X, $8HFRACTION, 13X, 8HFRACTION, 6X, 8HFRACTION)
BY=-1.0
BYTIP=FLOAT(NON)
JCP=0
FLNON=FLOAT(NON)
DO 875 NSPAN=100, NR
BY=BY+1.0
JCP=JCP+1
YB2=BY/BYTIP
IF(ALIFT9.EQ.0.) SLIFT=TSLIFT(JCP,1)
IF(ALIFT9.EQ.0.) GO TO 840
SLIFT=TSLIFT(JCP,1)/ALIFT9
840 IF(ALFF9.EQ.0.) SLFF=TSLIFT(JCP,2)
IF(ALFF9.EQ.0.) GO TO 842
SLFF=TSLIFT(JCP,2)/ALFF9

245
IF(DRAG9. EQ. 0.) SDRAG = TSDRAG(JCP)
IF(DRAG9. EQ. 0.) GO TO 875
SDRAG = TSDRAG(JCP)/DRAG9
WRITE(14, 865) YB2, SLFF, SLFF, SLIFT, SDRAG
WRITE(62, 865) YB2, SLFF, SLFF, SLIFT, SDRAG
WRITE(26, 865) YB2, SLFF, SLFF, SLIFT, SDRAG

875 CONTINUE
865 FORMAT(17X, F10.6, 5X, F11.6, 3X, F11.6, 10X, F11.6, 3X, F11.6)
CLOSE(UNIT = 14)
IF(ALFF9. EQ. 0.) THEN
  WRITE(26, 3519)
  WRITE(62, 3519)
ENDIF
IF(ALIFT9. EQ. 0.) THEN
  WRITE(26, 3520)
  WRITE(62, 3520)
ENDIF
IF(DRAG9. EQ. 0.) THEN
  WRITE(26, 3521)
  WRITE(62, 3521)
ENDIF

3519 FORMAT(2X/15X, 56H SINCE CL AND CD FOR FLAT WING ARE ZERO THE DISTRIBUTIONS/9X, 26H ARE NOT IN FRACTIONAL FORM)
3520 FORMAT(2X/15X, 56H SINCE CL FOR CAMBERED WING IS ZERO THE LIFT DISTRIBUTION/9X, 25H IS NOT IN FRACTIONAL FORM)
3521 FORMAT(2X/15X, 56H SINCE CD FOR CAMBERED WING IS ZERO THE DRAG DISTRIBUTION/9X, 25H IS NOT IN FRACTIONAL FORM)

790 CONTINUE
500 RETURN
END
SUBROUTINE QUERY(NANS)
C
C ROUTINE TO TRAP ERRORS CAUSED BY IMPROPER RESPONSES TO QUESTIONS.
C THE COMPUTER GENERATES AN ERROR WHEN A CHARACTER IS SUPPLIED TO
C A QUESTION EXPECTING AN INTEGER OR REAL VALUE.
C
NQTEST = 0
1 CONTINUE
IF (NQTEST .GT. 0) THEN
  PRINT *, ' CHARACTER VALUES ARE NOT VALID. '
  PRINT *, ' PLEASE ENTER AN INTEGER VALUE. '
END IF
NQTEST = NQTEST + 1
READ (5, *, ERR=1) NANS
RETURN
END
SUBROUTINE SLOPE
C
C TO OBTAIN THE STREAMWISE SLOPE, DZDX, IN ALL BLOCKS OF A WING
C PLANFORM GRID FOR A CAMBERED WING SURFACE - LINEAR INTERPOLATION
C BETWEEN INPUT POINTS
COMMON TYB2(51), TZORD(26, 51), JBYMAX, NON, NOPCT, RATIO, XLEO, XTEO,
1 TPCT(26), TXLE(50), TXTE(50), DZDX, XMAX, CBAR, TDX(105, 51), XM, NOM,
2 TMACH(5), TZSKAL, REFAR, SPAN, XO, PI, CNPOD, CAPOD, TCNPOD(5), TCAPOD(5)
COMMON FDZDX, XLEOF, TXLEF(50), NFLAP2, NFLAP1, ZMAX, IDDI
DIMENSION TFY(26,51), TCHORD(26,3), BYNON(51), ZZMAX(51), IN(2), IIXM(4), IYM(3), TZZ(26)

DO 1 KF=1, 26
DO 1 NF=1, 51
1 TFY(KF, NF)=0.0
DO 2 KC=1, 26
DO 2 NC=1, 3
2 TCHORD(KC, NC)=0.0
MAX=NON+1
FNON=FLOAT(NON)

C SPANWISE INTERPOLATION ALONG CONSTANT PERCENT CHORD LINES

C DO 130 JSPAN=1, NOPCT
PCC=TPCT(JSPAN)
Y1=0.0
Y2=FNON*TYB2(JSPAN)
FY1=TZORD(JSPAN, 1)
FY2=TZORD(JSPAN, 2)
C2=(FY1-FY2)/(Y1-Y2)
C1=0.5*(FY1+FY2-C2*(Y1+Y2))
JCOL=0
JUPR=INT(Y2)+1
DO 120 KOL=1, JUPR
JCOL=JCOL+1
BY=FLOAT(JCOL)-1.0
FY=C1+C2*BY
FYR*FY*RATIO
120 TFY(JSPAN, JCOL)=FY
JBY1=1
JLAST=JBYMAX-2
DO 130 JCU=1, JLAST
JBY1=JBY1+1
JBY2=JBY1+1
Y1=FNON*TYB2(JBY1)
Y2=FNON*TYB2(JBY2)
FY1=TZORD(JSPAN, JBY1)
FY2=TZORD(JSPAN, JBY2)
C2=(FY1-FY2)/(Y1-Y2)
C1=0.5*(FY1+FY2-C2*(Y1+Y2))
IF(JCU-JLAST)122, 121, 122
121 Y2=FNON+0.5
122 BYTEST=BY+1.0
124 JCOL=JCOL+1
BY=BYTEST
ZFY=C1+C2*BY
ZFYR*ZFY*RATIO
TFY(JSPAN, JCOL)=ZFY
GO TO 122
130 CONTINUE
C WRITE(26,902)
902 FORMAT(2X///41X, 38H***PROGRAM GENERATED GEOMETRIC DATA***/)
C

247
CHORDWISE INTERPOLATION ALONG CONSTANT SPANWISE N VALUES.
SURFACE SLOPES DZDX CALCULATED.

DO 160 JCOL=1,MAX
    JBY=JCOL-1
    YBO2=FLOAT(JBY)/FNON
WRITE(26,1000)JBY,YBO2
1000 FORMAT(2X//15X,2HN=I3,5X,29HSNAP STATION, Y/B/2 = N/NON =F8.5/22X
    $,1HL,9X,1HX,7X,5HX-XLE,7X,1HZ,1HX,7X,5HX-XLE,5X,4HDZDX/)
    IF(JBY)132,131,132
131 XLE=XLEO
    XTE=XTEO
    GO TO 133
132 XLE=TXLE(JBY)
    XTE=TXTE(JBY)
    GO TO 133
133 LLE=INT(XLE)+1
    LTE=INT(XTE)+1
    CHORD=XTE-XLE
    IF(LLE-LTE)134,152,134
134
    PCT1=TPCT(1)*.01
    PCT2=TPCT(2)*0.01
    Y1=PCT1*CHORD
    Y2=PCT2*CHORD
    FY1=TFY(1,JCOL)
    FY2=TFY(2,JCOL)
    C2=(FY1-FY2)/(Y1-Y2)
    C1=0.5*(FY1+FY2-C2*(Y1+Y2))
    TCHORD(1,1)=C1
    TCHORD(1,2)=C2
    TCHORD(1,3)=Y2+XLE
    JX1=1
    LAST=N0PCT-2
    DO 136 LINK=1,LAST
        JX1=JX1+1
        JX2=JX1+1
        PCT1=TPCT(JX1)*0.01
        PCT2=TPCT(JX2)*0.01
        Y1=PCT1*CHORD
        Y2=PCT2*CHORD
        FY1=TFY(JX1,JCOL)
        FY2=TFY(JX2,JCOL)
        C2=(FY1-FY2)/(Y1-Y2)
        C1=0.5*(FY1+FY2-C2*(Y1+Y2))
        JROW=LINK+1
        TCHORD(JROW,1)=C1
        TCHORD(JROW,2)=C2
        TCHORD(JROW,3)=Y2+XLE
    136 CONTINUE

    JROW=1
    KWIT=0
    DO 157 L=LLE,LTE
        XVAR=FLOAT(L)+0.5
        IF(L.GE.LTE-1) GO TO 142
        GO TO 139
142 IF(L.LE.EQ.LTE-1)GO TO 143
    GO TO 138

248
143 XVPRE=XLE
   ZPRE=TFY(1,JCOL)
   Z=TFY(NOPCT,JCOL)
   KWIT=1
   X1=XVPRE
   GO TO 140
138 XVPRE=XVAR-1.0
   ZPRE=Z
   Z=TFY(NOPCT,JCOL)
   KWIT=1
   X1=XVPRE
   GO TO 140
139 XTEST=TCHORD(JROW,3)
   IF(XTEST-XVAR)141,145,145
141 JROW=JROW+1
   GO TO 139
145 C1=TCHORD(JROW,1)
   C2=TCHORD(JROW,2)
   IF(L-LLE)147,146,147
146 ZPRE=TFY(1,JCOL)
   XLGTH=XVAR-XLE
   X1=XLE
   GO TO 148
147 ZPRE=Z
   X1=FLOAT(L)-.5
148 XPM=XVAR-XLE
   Z=C1+C2*XPM
140 DZDX=Z-ZPRE
   IF(KWIT)151,150,151
150 IF(L.EQ.LLE) GO TO 149
   GO TO 155
149 DZDX=DZDX/XLGTH
   GO TO 155
151 ATE=XTE-XVPRE
   IF(ATE)169,168,169
168 DZDX=0.0
   GO TO 155
169 DZDX=DZDX/ATE
155 TDZDX(L,JCOL)=DZDX
   XP1=X1-XLE
   X2=FLOAT(L)
   XP2=X2-XLE
   ZPRINT=ZPRE
   C WRITE(26,1001)L,X1,XP1,ZPRINT,X2,XP2,DZDX
905 IF(KWIT)405,157,405
405 DO 406 LAD=1,6
406 TDZDX(L+LAD,JCOL)=DZDX
   L6=L+6
   GO TO 160
157 CONTINUE
152 IF(CHORD.LE..001) GO TO 153
   GO TO 154
153 DZDX=0.0
   GO TO 156
154 ZLE=TFY(1,JCOL)
ZTE = TFY(NOPCT, JCOL)
DZDX = (ZTE - ZLE) / CHORD

156 TDZDX(LLE, JCOL) = DZDX
L = LLE
X1 = XLE
XP1 = X1 - XLE
ZPRINT = TFY(1, JCOL)
X2 = XTE
XP2 = X2 - XLE

CWRITE(26, 1001) L, X1, XP1, ZPRINT, X2, XP2, DZDX
DO 408 LAD = 1, 5
408 TDZDX(LLE + LAD, JCOL) = DZDX
L5 = LLE + 5
160 CONTINUE
500 RETURN
END
LIST OF REFERENCES


<table>
<thead>
<tr>
<th>No.</th>
<th>DISTRIBUTION LIST</th>
<th>No. Copies</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>Defense Technical Information Center&lt;br&gt;Cameron Station&lt;br&gt;Alexandria, VA 22304-6145</td>
<td>2</td>
</tr>
<tr>
<td>2.</td>
<td>Superintendent&lt;br&gt;Attn: Library, Code 1424&lt;br&gt;Naval Postgraduate School&lt;br&gt;Monterey, CA 93943-5000</td>
<td>2</td>
</tr>
<tr>
<td>3.</td>
<td>E. R. Wood, Chairman, Code 67&lt;br&gt;Department of Aeronautics and Astronautics&lt;br&gt;Naval Postgraduate School&lt;br&gt;Monterey, CA 93943-5000</td>
<td>1</td>
</tr>
<tr>
<td>4.</td>
<td>Professor J. V. Healey, Code 67He&lt;br&gt;Department of Aeronautics and Astronautics&lt;br&gt;Naval Postgraduate School&lt;br&gt;Monterey, CA 93943-5000</td>
<td>2</td>
</tr>
<tr>
<td>5.</td>
<td>Professor M. F. Platzer, Code 67Pl&lt;br&gt;Department of Aeronautics and Astronautics&lt;br&gt;Naval Postgraduate School&lt;br&gt;Monterey, CA 93943-5000</td>
<td>2</td>
</tr>
<tr>
<td>6.</td>
<td>Professor M. S. Chandrasekhar, Code 67Ch&lt;br&gt;Department of Aeronautics and Astronautics&lt;br&gt;Naval Postgraduate School&lt;br&gt;Monterey, CA 93943-5000</td>
<td>1</td>
</tr>
<tr>
<td>7.</td>
<td>Professor D. J. Collins, Code 67 Co&lt;br&gt;Department of Aeronautics and Astronautics&lt;br&gt;Naval Postgraduate School&lt;br&gt;Monterey, CA 93943-5000</td>
<td>1</td>
</tr>
<tr>
<td>8.</td>
<td>Professor S. K. Hebbar, Code 67 Hb&lt;br&gt;Department of Aeronautics and Astronautics&lt;br&gt;Naval Postgraduate School&lt;br&gt;Monterey, CA 93943-5000</td>
<td>1</td>
</tr>
</tbody>
</table>
9. Professor R. M. Howard, Code 67Ho
Department of Aeronautics and Astronautics
Naval Postgraduate School
Monterey, CA 93943-5000

10. Professor J. A. Miller, Code 67 Mo
Department of Aeronautics and Astronautics
Naval Postgraduate School
Monterey, CA 93943-5000

11. Professor D. W. Netzer, Code 67 Nt
Department of Aeronautics and Astronautics
Naval Postgraduate School
Monterey, CA 93943-5000

12. Professor L. V. Schmidt, Code 67 Sc
Department of Aeronautics and Astronautics
Naval Postgraduate School
Monterey, CA 93943-5000

13. Professor R. P. Shreeve, Code 67Sf
Department of Aeronautics and Astronautics
Naval Postgraduate School
Monterey, CA 93943-5000

14. Mr. E. Ward, Code 67 Ew
Department of Aeronautics and Astronautics
Naval Postgraduate School
Monterey, CA 93943-5000

15. Mr. A. Cricelli, Code 67 Cr
Department of Aeronautics and Astronautics
Naval Postgraduate School
Monterey, CA 93943-5000

16. Commander
U. S. Army Materiel Command
ATTN: AMCDE-O
5001 Eisenhower Avenue
Alexandria, VA 22333-0001

17. Commandant
U.S. Army Air Defense Artillery School
ATTN: ATSA-AC-FP
Ft. Bliss, TX 79916-7004

254
18. Director
   Ballistics Research Laboratory
   Aberdeen Proving Ground, MD  21005-5066

19. CPT Craig M. MacAllister
    14123 Misty Meadow
    Houston, TX  77079
Computer graphics adaptation of several aero-dynamic prediction programs.