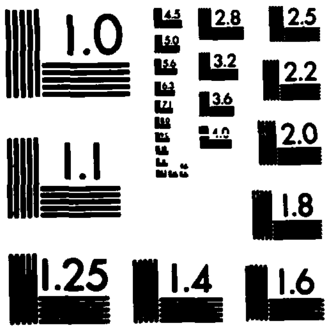
AD-A123 285 A MICROCOMPUTER-BASED METHOD FOR PREDICTING VISCOUS FLOW ABOUT A SHIP HULL(U) DCW INDUSTRIES INC STUDIO CITY CA D C WILCOX DEC 82 DCW-R-28-01 ONR-CR062-735-1F 1/2

UNCLASSIFIED N00014-82-C-0432 F/G 9/2 NL



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

(12)

AD A123285

OFFICE OF NAVAL RESEARCH

ONR

USN

A MICROCOMPUTER-BASED METHOD FOR
PREDICTING VISCOUS FLOW
ABOUT A SHIP HULL

DCW INDUSTRIES

DTIC
ELECTE
JAN 12 1983
A

4367 TROOST AVENUE, STUDIO CITY, CALIFORNIA 91604

REPORT ONR-CR062-735-1F

OFFICE OF NAVAL RESEARCH

A MICROCOMPUTER-BASED METHOD FOR PREDICTING VISCOUS FLOW ABOUT A SHIP HULL

David C. Wilcox

DCW INDUSTRIES, INC.
Studio City
California
91604

CONTRACT N00014-82-C-0432
ONR TASK 062-735

December 1982

FINAL REPORT

PREPARED FOR THE

OFFICE OF NAVAL RESEARCH ● 800 N. QUINCY ST. ● ARLINGTON ● VA ● 22217

UNCLASSIFIED
SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER ONR-CR062-735-1F	2. GOVT ACCESSION NO. AD-A123285	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) A MICROCOMPUTER-BASED METHOD FOR PREDICTING VISCOUS FLOW ABOUT A SHIP HULL		5. TYPE OF REPORT & PERIOD COVERED FINAL REPORT 20 MAY 1982 - 15 NOV 1982
		6. PERFORMING ORG. REPORT NUMBER DCW-R-28-01
7. AUTHOR(s) DAVID C. WILCOX		8. CONTRACT OR GRANT NUMBER(s) N00014-82-C-0432
9. PERFORMING ORGANIZATION NAME AND ADDRESS DCW INDUSTRIES, INC. 4367 Troost Avenue Studio City, CA 91604		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS NR 062-735
11. CONTROLLING OFFICE NAME AND ADDRESS Office of Naval Research Code 430, 800 North Quincy Street Arlington, Virginia 22217		12. REPORT DATE DECEMBER 1982
		13. NUMBER OF PAGES 97
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office)		15. SECURITY CLASS. (of this report) UNCLASSIFIED
		15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for Public Release, Distribution Unlimited.

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

18. SUPPLEMENTARY NOTES

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

SHIP HYDRODYNAMICS, THREE-DIMENSIONAL BOUNDARY LAYERS, MICROCOMPUTERS

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

A microcomputer-based computational method has been devised for numerically simulating viscous flow past an arbitary ship hull. The method uses (a) thin-ship theory to compute hull-surface velocities and (b) a three-dimensional boundary-layer program which embodies the Wilcox-Rubesin two-equation turbulence model to compute boundary-layer development. The method makes provision for including displacement effects.

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

UNCLASSIFIED
SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

The computational package, developed in the form of seven program modules, has been developed for use by a ship-hull designer in an interactive mode. All program modules can be run on a conventional Z-80 based microcomputer although run times are too lengthy for practical design applications. With a 16-bit microprocesser, execution time for a complete computation should be 3 to 4 hours.

CONTENTS

1. INTRODUCTION

During the past five years the microcomputer has developed into a powerful computational aid to the engineer. Currently, Z80-based microcomputers are capable of performing most day-to-day numerical-fluid-mechanics tasks such as data preparation/reduction, two-dimensional boundary-layer computations with very fine finite-difference grids, and many other practical computations including, on relatively crude finite-difference grids, time-averaged Navier-Stokes solutions. With the exception of the latter example, these computations usually can be done as fast on a microcomputer as running on a scientific mainframe computer, if you include the time required for transmitting the input data and final printout via a terminal interface.

This is true because of the overall real time spent using a mainframe computer with the microcomputer as a terminal, typically only 5% is devoted to the actual execution of the program. The other 95% consists of signing on, transmitting input data, and eventually transmitting the final printout. Obviously the time distribution varies depending on the overall execution time and the amount of print taken. However, our experience over a two-year period has shown a 5%-95% split to be quite representative for numerical-fluid-mechanics applications.

On the basis of the great power of modern microcomputers, this project has been undertaken with the overall aim of developing a microcomputer-based computational procedure for predicting viscous flow about a ship hull. The theoretical foundation of the procedure is (a) classical thin-ship theory[1] for the inviscid velocity distribution on the ship hull and (b) a three-dimensional boundary-layer program, EDDY3, which incorporates the Wilcox-Rubesin[2] two-equation turbulence model. In a straightforward manner, the computational procedure corrects for boundary-layer displacement effects. With the exception of the boundary-layer computation, all phases of the procedure are accomplished on a TRS-80 Model II (Z80-based) microcomputer. While the boundary-layer computations could also be done on the TRS-80, run times are too long (30 hours) to render such runs useful in a design environment.

With the exception of a plotting routine (SPLOT) and an interpolation routine (INTERP), all programs have been developed in FORTRAN (ANSI-66) and, with trivial modifications, can thus be run on virtually all scientific computers.

In Section 2 we first present an overview of the computational procedure. Next we describe the various programs including input and output (program listings are located in the Appendix). Then, we present results of two sample computations.

Section 3 summarizes and discusses results of the project.

2. ANALYSIS

2.1 OVERVIEW OF THE COMPUTATIONAL PROCEDURE

In devising the computational procedure, our plan has been to create a method which lays the foundation for a ship designer to effect design modifications in an interactive mode. Because we are creating the foundation, the procedure has been developed in several discrete modules each of which performs a straightforward task.

Our starting point is a disk file containing the three-dimensional hull coordinates from bow to stern. This file presumably has been created by the designer. As illustrated in Figure 1, we first use program PRMESH to generate a preliminary orthogonal, rectangular mesh for use in computing the inviscid velocities on the hull. Before proceeding to computation of the hull velocities, program SPLOT can be used to visually inspect the coordinates via the micro-computer video display. This step provides an opportunity to spot obvious input data errors. Once we are confident the input data are satisfactorily smooth, we use program SHIP to compute the inviscid velocities on the ship-hull surface. This completes the inviscid phase of the computational procedure.

The second (viscous) phase of the procedure begins with program SHPMSH which generates a nonorthogonal finite-difference grid for use in the 3-D boundary-layer computation, including all required geodesic curvatures and metrics. Then, using program VELOC we interpolate the inviscid velocities onto the nonorthogoral grid. At this point, all required input data for program EDDY3 have been prepared. The final step in the viscous phase is execution of program EDDY3 which computes boundary-layer development over the hull.

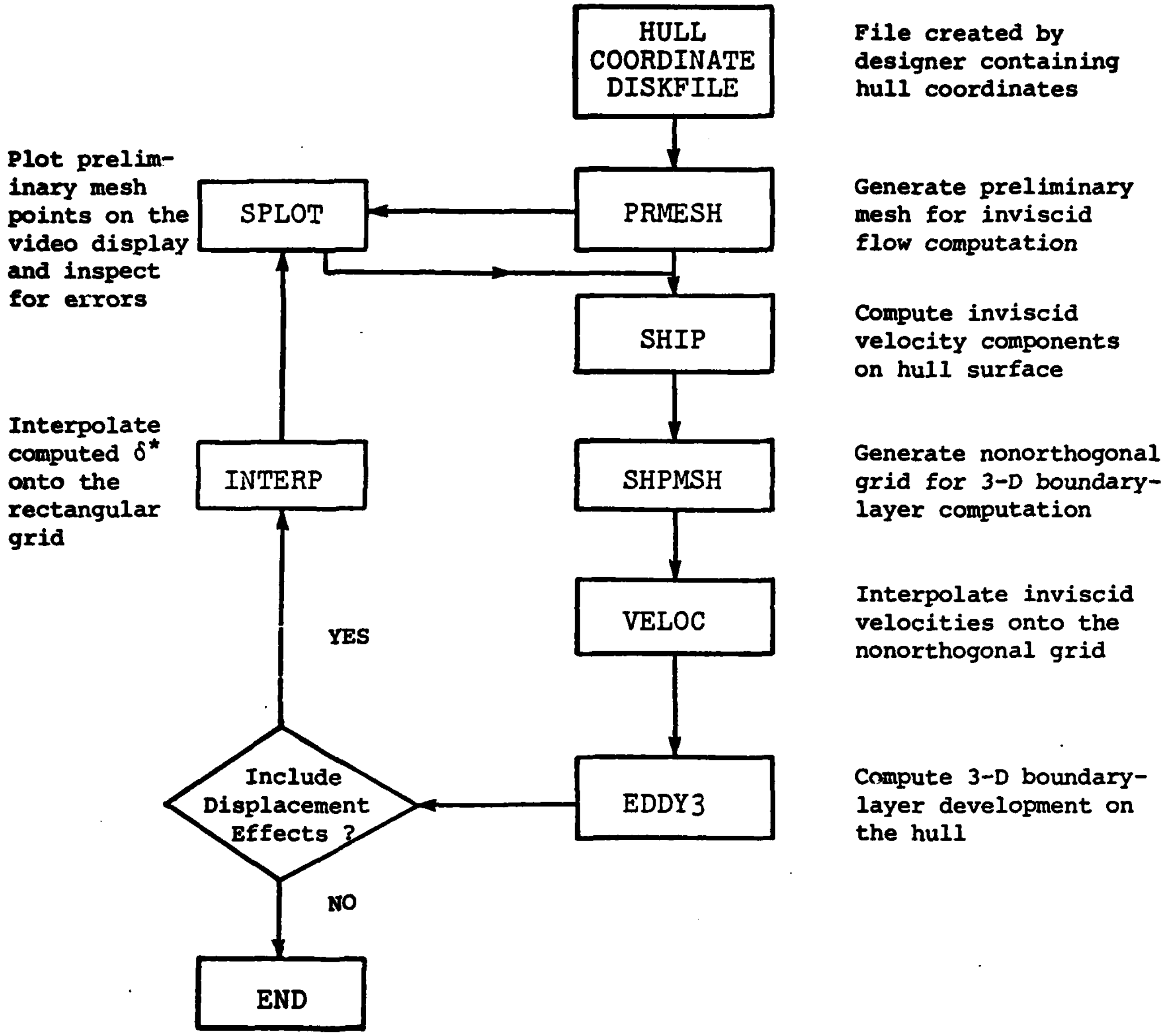


Figure 1. Schematic representation of the computational procedure.

Upon completion of the viscous phase, the designer can opt to include effects of boundary-layer displacement. Should this option be selected, a disk file is available from EDDY3 which containes the computed displacement thicknesses on the nonorthogonal grid. Program INTERP interpolates the contents of this file onto the rectangular grid used by program SHIP. The complete computational sequence is then repeated.

Certainly visual inspection of the inviscid velocities, nonorthogonal grid coordinates, final output from EDDY3, etc. would be helpful, if not essential, in a design application. Simple modifications to SPLOT, for example, would provide the vehicle for such inspection. Even more powerful microcomputer graphics techniques are available for both video and hard-copy displays and there seems little point to going to great lengths at this point to rival existing software. Thus, we have not devised the needed modifications because of the preliminary nature of this project which has been, to some extent, a feasibility study. Nevertheless, the capability for inspection of the input data has been included to give an indication of the practicability of a microcomputer-based approach.

2.2 PROGRAM INPUT/OUTPUT

We turn now to the input and output of the seven program modules. The Appendix contains complete listings of each program. In the interest of clarity, for each program, we reiterate its overall purpose below.

2.2.1 Program PRMESH

<u>The purpose of this program</u> is to generate a rectangular finite-difference grid for use in computing the inviscid velocity components.

<u>Program input data</u> consists of two disk files, viz, MESH and PRMESHIN. The first of these two files contains the coordinates $\bar{x}$, $\bar{y}$, $\bar{z}$ of

the ship (Figure 2), where the coordinates are most conveniently supplied in nondimensional form (normalized, for example, by hull length, L). Note that flow past the hul' is toward positive $\bar{x}$, positive $\bar{y}$ is to starboard, and $\bar{z}$ increases as we move from keel to waterline.

In disk file MESH, the coordinates are presented on constant-$\bar{x}$ sections with each line giving two pairs of coordinates. The format is (3F10.5,1X,3F10.5). Note that the space between coordinate pairs (the 1X) is included for consistancy with the input data format specified for the 1980 ITTC/SSPA Workshop on Ship Bound y Layers. Thus, a few typical lines of data assume the followir form:

SAMPLE 'MESH" INPUT DATA

$\bar{x}$	$\bar{y}$	$\bar{z}$	$\bar{x}$	$\bar{y}$	$\bar{z}$
-0.40000	-0.13152	-0.07847	-0.40000	-0.13425	-0.06825
-0.40000	-0.13603	-0.05375	-0.40000	-0.13668	-0.03680
-0.40000	-0.13705	-0.01872	-0.40000	-0.13738	0.00000
-0.35000	0.00000	-0.11808	-0.35000	-0.00690	-0.11803
-0.35000	-0.01498	-0.11790	-0.35000	-0.02537	-0.11767
-0.35000	-0.03952	-0.11732	-0.35000	-0.05698	-0.11668

Note that points on each section are input from keel (where $\bar{y}$ = 0) to waterline (where $\bar{z}$ = 0). In its current form, program PRMESH will accept data on up to 50 sections with a maximum of 20 ($\bar{y}$,$\bar{z}$) pairs per section. One restriction on the input data is that an even number of ($\bar{y}$,$\bar{z}$) pairs must be prescribed. Note also that the input coordinates must define the ship hull from bow to stern; a partial mesh is insufficient.

The second input disk file, PRMESHIN, contains the following data:

IMAX = Number of sections (constant $\bar{x}$) in the computed rectangular mesh (cannot exceed 41).

JMAX = Number of points per section in the computed rectangular mesh (cannot exceed 20).

NXTL = Number of sections in the input mesh of disk file MESH (cannot exceed 50).

NZTL = Number of ($\bar{y}$,$\bar{z}$) pairs per section in the input mesh of disk file MESH (cannot exceed 20).

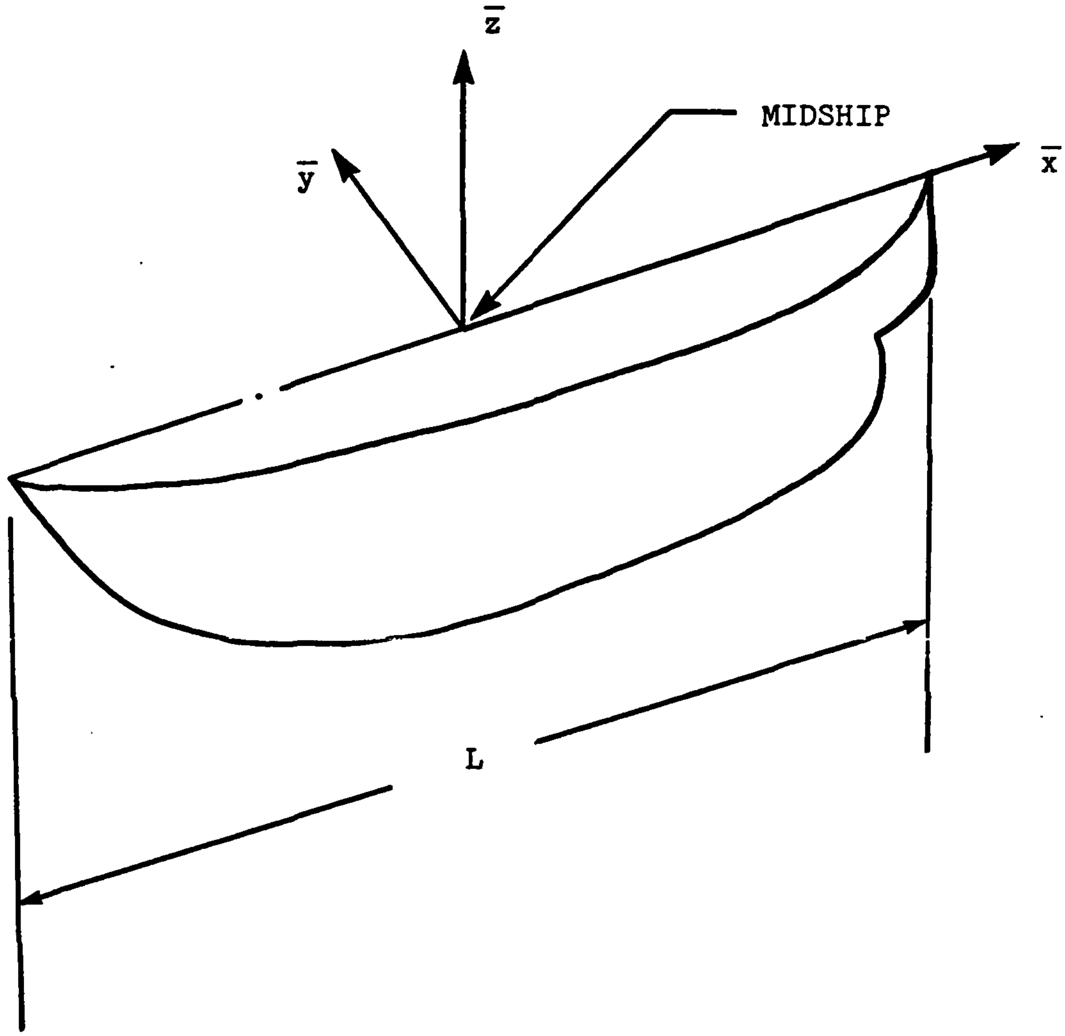


Figure 2. Schematic of ship-hull coordinate system.

X = Specified streamwise ($\bar{x}$-coordinate) array for the computed rectangular mesh (a total of IMAX values).

Z = Specified vertical ($\bar{z}$-coordinate) array for the computed rectangular mesh (a total of JMAX values).

The input format for this disk file has been designed to resemble NAMELIST input. For IMAX, JMAX, NXTL and NZTL, the format is (1X,3A4,I4) while for X and Z it is (1X,3A4,E13.6). A typical input file is shown below:

SAMPLE 'PRMESHIN' INPUT DATA

```
IMAX      =     5
JMAX      =     4
NXTL      =    40
NZTL      =    20
X(1)      = -0.500000E 00
X(2)      = -0.250000E 00
X(3)      =  0.000000E 00
X(4)      =  0.250000E 00
X(5)      =  0.500000E 00
Z(1)      =  0.000000E 00
Z(2)      = -0.030000E 00
Z(3)      = -0.065000E 00
Z(4)      = -0.100000E 00
```

As illustrated, the (1X3A4) part of the format permits input of the variable name. Unlike NAMELIST input, all variables must be input and the order of input cannot be mixed. An exception to this is that only IMAX (JMAX) values of X(Z) must be input; the unused parts of these arrays needn't be defined.

Program output is (1) a printed listing of input disk file PRMESHIN and (2) a disk file named MESHPR containing the coordinates of the computed rectangular mesh. Inspection of the latter can be accomplished with program SPLOT described in the next Subsection.

2.2.2 Program SPLOT

The purpose of program SPLOT is to permit section-by-section visual inspection of the rectangular mesh generated by program PRMESH.

Such an inspection will aid in identifying smoothness of the mesh as well as input data errors.

The program is written in BASIC, makes use of special features of the TRS-80 Model II, and cannot be readily adapted to another machine. However, for reasons noted in the Introduction, it has been included for demonstration purposes and is not essential in the overall computational procedure. Input to the program consists of (1) prompted requests for IMAX and JMAX and (2) disk file MESHPR. Also, as the visual inspection proceeds, additional prompted requests are made as will be discussed below.

All input and output appears on the video display. Upon execution of SPLOT, the program will print "IMAX=?". In response, the value of IMAX used in running PRMESH must be entered. Next the message "JMAX=?" is displayed. Again, the value used in running PRMESH must be entered.

The program will now proceed section by section. On each section it will print the streamwise section number and the value of $\bar{x}$ on the section. This will be followed by a printout identifying the maximum and minimum values of $\bar{y}$ and $\bar{z}$ on the section. The program then requests the appropriate plotting range, viz, the desired values of $\bar{y}_{min}$, $\bar{y}_{max}$, $\bar{z}_{min}$, $\bar{z}_{max}$. After the first section has been investigated, the program first asks if the same plotting range used for the previous plot is desired. Entering "Y" obviates the need to reenter $\bar{y}_{min}$, etc. Entering "N" permits changing the plotting range. This feature has been included to permit viewing all sections on the same scale.

The plot for a given section will be displayed until a character is entered. The following options are available:

C = Continue inspection on this section;

N = Next section - proceed to same;

For any other ASCII character, the run terminates. The "C" option permits changing the plotting range. This might be done to expand the scale in a particular region for closer inspection. Note that if this is done, the entered $\bar{y}_{min}$, etc. become the new plotting range for all following sections. Hence, returning to the original plotting range requires the appropriate input once the user proceeds to the next section. The "C" option can be used as many times as desired on each section.

As a safeguard against incorrect entries, the program asks if "ANY ERRORS ?" have been made each time $\bar{y}_{min}$, etc. are entered. If no plotting-range errors have been made, enter "N" and the program continues. If an error has been made, enter "Y" and the correct values can then be typed in. Finally, any points which are off scale will be indicated in the lefthand margin of the plot.

When the user is satisfied the mesh is sufficiently smooth and free of errors, he/she is ready to proceed to computation of the inviscid flow. This end is accomplished with the program described in the next subsection.

2.2.3 Program SHIP

The purpose of program SHIP is to compute the inviscid velocity components on the surface of the ship hull. As an auxiliary output, the program computes the wave drag.

Input to program SHIP consists of two disk files, viz: (1) file MESHPR created by program PRMESH; and (2) disk file SHIPIN. The latter file has again been created in a format similar to NAMELIST. The input data are:

- FR = Froude number defined by $F_r = U/\sqrt{gL}$ where U is free-stream velocity, g is gravitational acceleration and L is the length scale used to nondimensionalize the ship coordinates $\bar{x},\bar{y},\bar{z}$.
- IMAX = Number of sections (constant $\bar{x}$) in the rectangular mesh (same value used in disk file PRMESHIN - cannot exceed 41);
- JMAX = Number of points per section in the rectangular mesh (same value used in disk file PRMESHIN - cannot exceed 10);

MINT2 = Number of points used in Chebyshev quadrature to evaluate an integral whose dimensionless wave-number range is 1 to ∞ (cannot exceed 80);

MINT3 = Number of points used in Chebyshev quadrature to evaluate an integral whose dimensionless wave-number range is zero to 1 (cannot exceed 80);

NZOT = Print flag. For zero the cell geometry and velocity components are printed. Otherwise, only the wave drag is printed.

As with disk file PRMESHIN of Subsection 2.2.1, the format is (1X,3A4,E13.6) for FR and (1X,3A4,I4) for all integer input quantities. A typical input file is as follows

SAMPLE 'SHIPIN' INPUT DATA

```
FR      =  2.500000E-01
IMAX    =    5
JMAX    =    4
MINT2   =   20
MINT3   =   20
NZOT    =    0
```

Note that the maximum value of JMAX allowed has been limited to 10 to keep execution times on a TRS-80 Model II within practical limits.

<u>Output from program SHIP</u> consists of: (1) video messages indicating when each of three stages of the computation is completed; (2) printed output depending on the value assigned to input flag NZOT; and (3) a disk file named UEWE containing mesh coordinates and computed velocity components.

<u>The first part of the printout</u> is diskfile SHIPIN. This provides an immediate check that input data have been correctly entered.

<u>Printed messages during the computation</u> appear on the video display upon completion of calls to subroutines VELOC1, VELOC2 and VELOC3. The message after completing a call to VELOCn is "VELOCn COMPUTATIONS

COMPLETE". These messages permit the user to monitor the progress of the run.

The next part of the printout depends upon the value assigned to NZOT. If NZOT is zero the program prints: the three coordinates of the rectangular cell centroids; the computed dimensionless velocity components u/U, v/U, w/U; and the dimensionless total velocity q/U. The print proceeds section by section including incremental wave drag on the section and total wave drag computed up to the section.

The final part of the printout is the total dimensionless wave drag (RWAVE) defined in terms of the drag R by

$$RWAVE = \frac{R}{\frac{1}{2}\rho U^2 L^2} \qquad (1)$$

where ρ is density, while U and L are as defined above.

Finally, diskfile UEWE is written for use by program VELOC described in Subsection 2.2.5. Upon completion of this phase of the computation, we have determined the inviscid velocity components on an orthogonal rectangular mesh.

2.2.4 Program SHPMSH

The purpose of this program is to generate a nonorthogonal finite-difference mesh for use by the three-dimensional boundary-layer program EDDY3 (Subsection 2.2.6). The program generates least-squares cubic-spline fits[4] to accurately and smoothly compute all required geodesic curvatures and metric coefficients.

Input data for SHPMSH consists of two diskfiles, viz, MESH and NAMLIST. Diskfile MESH is the original file containing the ship-hull coordinates. In the interest of economy, this file needn't cover the entire hull. Rather, it need only cover the part of the hull over which the boundary layer is to be computed. Diskfile

NAMLIST contains the following input data:

ISYM = Symmetry flag. To force mesh symmetry ($d\bar{y}/d\bar{z}=0$) at the waterline use ISYM=0. Otherwise the program automatically infers the appropriate value of $d\bar{y}/d\bar{z}$ at the waterline.

IU1 = Logical unit number for input diskfile MESH.

IU2 = Logical unit number for temporary diskfile SCRATCH1.

IU3 = Logical unit number for temporary diskfile SCRATCH2.

IU4 = Logical unit number for output diskfile GEOMETRY.

IU5 = Logical unit number for final printout.

IMAX = Number of sections in the input mesh of diskfile MESH (cannot exceed 40).

JMAX = Number of ($\bar{y},\bar{z}$) pairs per section in the input mesh of diskfile MESH (cannot exceed 40).

NXTL = Number of sections in the computed nonorthogonal mesh (cannot exceed 40).

NZTL = Number of points, per section in the nonorthogonal mesh (cannot exceed 21).

JFZ = Number of key points used for the least-squares cubic-spline fit in the girthwise direction (cannot exceed 30).

LZ = Array of indices of the key points to be used in the girthwise direction.

JFX = Number of key points used for the least-square cubic-spline fit in the streamwise direction (cannot exceed 30).

LX = Array of indices of the key points to be used in the streamwise direction.

PSIZ = Nondimensional distance (measured from the bow) of the most upstream section in the nonorthogonal mesh.

XSTART = Value of $\bar{x}$ at the most upstream section in the nonorthogonal mesh.

XEND = Value of $\bar{x}$ at the most downstream section in the nonorthogonal mesh.

DX = Array of streamwise stepsizes, $\Delta\bar{x}_i$, for the nonorthogonal mesh (cannot exceed 10).

NX = Array of indices indicating the number of times $\Delta\bar{x}_i$ is repeated, viz, the mesh will have $\Delta\bar{x}$=DX(1) for the first NX(1) stations, $\Delta\bar{x}$=DX(2) for the next NX(2) Stations, etc.

DZ = Array of girthwise stopsizes, $\Delta\bar{z}_j$, for the non-orthogonal mesh (cannot exceed 10). Note that the mesh is constructed so that $\bar{z}$=0 on the keel and $\bar{z}$=1 on the waterline.

NZ = Array of indices indicating the number of times $\Delta\bar{z}_j$ is repeated.

As in programs PRMESH and SHIP, the format for integer quantities is (1X,3A4,I4) while floating point variables use (1X,3A4,E13.6). Sample input is as follows:

SAMPLE 'NAMLIST' INPUT DATA

ISYM	=	1	DX(5)	=	0.000000E 00
IU1	=	7	DX(6)	=	0.000000E 00
IU2	=	8	DX(7)	=	0.000000E 00
IU3	=	9	DX(8)	=	0.000000E 00
IU4	=	10	DX(9)	=	0.000000E 00
IU5	=	2	DX(10)	=	0.000000E 00
IMAX	=	30	NX(1)	=	5
JMAX	=	18	NX(2)	=	2
NXTL	=	15	NX(3)	=	7
NZTL	=	11	NX(4)	=	0
JFZ	=	8	NX(5)	=	0
LZ(1)	=	1	NX(6)	=	0
LZ(2)	=	3	NX(7)	=	0
LZ(3)	=	5	NX(8)	=	0
LZ(4)	=	7	NX(9)	=	0
LZ(5)	=	10	NX(10)	=	0
LZ(6)	=	12	ZSTART	=	0.000000E 00
LZ(7)	=	15	ZEND	=	1.000000E 00
LZ(8)	=	18	DZ(1)	=	1.000000E-01
JFX	=	12	DZ(2)	=	0.000000E 00
LX(1)	=	1	DZ(3)	=	0.000000E 00
LX(2)	=	3	DZ(4)	=	0.000000E 00
LX(3)	=	6	DZ(5)	=	0.000000E 00
LX(4)	=	9	DZ(6)	=	0.000000E 00
LX(5)	=	12	DZ(7)	=	0.000000E 00
LX(6)	=	15	DZ(8)	=	0.000000E 00
LX(7)	=	18	DZ(9)	=	0.000000E 00
LX(8)	=	21	DZ(10)	=	0.000000E 00
LX(9)	=	24	NZ(1)	=	10
LX(10)	=	26	NZ(2)	=	0
LX(11)	=	28	NZ(3)	=	0
LX(12)	=	30	NZ(4)	=	0
PSIZ	=	3.000000E-01	NZ(5)	=	0
XSTART	=	-7.000000E-01	NZ(6)	=	0
XEND	=	9.000000E-01	NZ(7)	=	0
DX(1)	=	1.000000E-01	NZ(8)	=	0
DX(2)	=	1.500000E-01	NZ(9)	=	0
DX(3)	=	1.000000E-01	NZ(10)	=	0
DX(4)	=	0.000000E 00			

Output from program SHPMSH consists of: video display messages indicating when various phases of the computation are complete; printed output of computed quantities; and diskfile GEOMETRY for use by programs VELOC (Subsection 2.2.5) and EDDY3 (Subsection 2.2.6).

The first printed output is a listing of diskfile NAMLIST which permits checking for input data errors.

Three messages are then printed on the video display during the run to permit monitoring the progress of the run. The messages are:

1. "All Section Data Smoothed and Differentiated";
2. "Geometrical Parameters Computed for J=j";
3. "d(Theta)/dz Computed for All Sections".

The second message is repeated for each girthwise station where j varies from 1 to JMAX.

The next output is the computed quantities along each girthwise station. This information is routed to the printer if IU5=2 or saved on disk provided IU5 is greater than or equal to 7. The data is saved on diskfile FORTu5/DAT where "u5" is the numerical value of IU5 when IU5≥7. This option permits reiterating to include displacement effects via program INTERP (Subsection 2.2.7).

The final output is diskfile GEOMETRY which is used by programs VELOC and EDDY3.

2.2.5 Program VELOC

The purpose of this program is to interpolate the inviscid velocities onto the nonorthogonal coordinate mesh for use in the three-dimensional boundary-layer computation.

Input data for this program consists of three diskfiles, viz, NAMVEL, UEWE and GEOMETRY. Diskfiles UEWE and GEOMETRY are created by programs SHIP and SHPMSH, respectively. Diskfile NAMVEL contains the following input data:

ISYM = Symmetry flag. To treat the waterline as a symmetry plane, use ISYM=0. Otherwise, the program will automatically infer velocities at the free surface.

IU1 = Logical unit number for input diskfile UEWE.

IU2 = Not used.

IU3 = Logical unit number for output diskfile VELOCITY.

IU4 = Logical unit number for input diskfile GEOMETRY.

IU5 = Logical unit number for final printout.

IMAX = Number of sections in the input mesh of diskfile UEWE (cannot exceed 40).

JMAX = Number of points per section in the input mesh of diskfile UEWE (cannot exceed 19).

NXTL = Number of sections in the nonorthogonal mesh of diskfile GEOMETRY (cannot exceed 40).

NZTL = Number of points per section in the nonorthogonal mesh of diskfile GEOMETRY (cannot exceed 21).

JFZ = Number of key points for the least-squares cubic-spline fit in the girthwise direction (cannot exceed 30).

LZ = Array of indices of the key points to be used in the girthwise direction.

JFX = Number of key points used for the least-squares cubic-spline fit in the streamwise direction (cannot exceed 30).

LX = Array of indices of the key points to be used in the streamwise direction.

Note that because program SHIP does not provide keel or waterline velocities, this program computes both. In so doing, the program replaces JMAX by JMAX+2. Also, JFZ, JFX, LZ and LX needn't be the same as used in program SHPMSH. As in all similar input data files of preceding programs, the format is (1X,3A4,I4) for integer input quantities and (1X,3A4,E13.6) for floating point data. A typical listing of diskfile NAMVEL follows.

SAMPLE 'NAMVEL' INPUT DATA

ISYM	=	1	LZ(6)	=	13
IU1	=	7	LZ(7)	=	16
IU2	=	8	LZ(8)	=	19
IU3	=	9	JFX	=	12
IU4	=	10	LX(1)	=	1
IU5	=	2	LX(2)	=	3
IMAX	=	31	LX(3)	=	6
JMAX	=	17	LX(4)	=	9
NXTL	=	15	LX(5)	=	12
NZTL	=	11	LX(6)	=	15
JFZ	=	8	LX(7)	=	18
LZ(1)	=	1	LX(8)	=	21
LZ(2)	=	3	LX(9)	=	24
LZ(3)	=	5	LX(10)	=	27
LZ(4)	=	7	LX(11)	=	29
LZ(5)	=	10	LX(12)	=	31

Program output consists of: video messages indicating when specific phases of the computation are complete; printed output listing interpolated nonorthogonal velocity components; and diskfile VELOCITY for use by program EDDY3.

The first output is a printout of input diskfile NAMVEL for inspection to insure no input data errors have been made.

The next output phase consists of the following messages on the video display:

1. "Section Data Smoothed & Differentiated for I=i";
2. "All Section Data Smoothed & Differentiated".

The first message is displayed for each section where i ranges from 1 to NXTL.

The program then provides printed output for all computed velocity components and for $\partial\overline{w}/\partial\overline{z}$ on the keel and waterline.

Finally, diskfile VELOCITY is written for use by program EDDY3.

2.2.6 Program EDDY3

The purpose of this program is to compute three-dimensional boundary-layer development on the ship hull. While all of the other programs listed in the Appendix are shown in their forms appropriate to the TRS-80 Model II microcomputer, the listing of EDDY3 is appropriate to a UNIVAC 1108 computer. Required changes for CDC or IBM computers are confined to SUBROUTINE STORIT and are included via Comment lines.

Input data for program EDDY3 consists of: diskfile GEOMETRY; diskfile VELOCITY; three NAMELIST files, viz, NAME, DATA and STRT; and two lines of card-type data. Diskfiles GEOMETRY and VELOCITY are created by programs SHPMSH and VELOC, respectively. The rest of the input is described below.

The first line of input is read from unit TAPEIN which has a default value of 5. This input is the title of the case to be run which can be up to 54 characters long (format 9A6).

The next segment of the input is also read from unit TAPEIN and is in NAMELIST format. The data are as follows:

NAMELIST/NAME/: This namelist includes primary parameters defining the grid, print logic, etc.

DETA = Array of $\Delta\eta$ values; only the $\Delta\eta$ nearest the surface need be input. Note that η is the conventional Levy-Lees-type normal coordinate.

EPS = Convergence criterion for turbulence-model equations (DEFAULT=.01).

EPST = Convergence criterion for crossflow momentum equation (DEFAULT=.01).

EPSV = Convergence criterion for streamwise momentum equation (DEFAULT=.01).

ETAE = Boundary-layer edge value for η at initial station (DEFAULT=8).

ICHORD = Not used (DEFAULT=1).

IFLOW = Flag indicating crossflow integration direction. IFLOW=1 for integration from keel to waterline and IFLOW=2 for integration from waterline to keel (DEFAULT=1).

IFPRNT = Print flag. IFPRNT=0 deletes printout of contents of diskfiles GEOMETRY and VELOCITY. Otherwise the contents of these two input files are printed (DEFAULT=0).

IPX = Intervals at which profiles are printed in streamwise direction, e.g., IPX=10 means print every tenth step.

IPZ = Intervals at which profiles are printed in girthwise direction.

ISPAN = Not used (DEFAULT=1).

ITMAX = Maximum number of iterations allowed (DEFAULT=20).

NPT = Maximum number of mesh points allowed normal to hull (must not exceed 101... DEFAULT=101).

NTR = Laminar/turbulent flow flag. To start from laminar profiles use NTR=0. Otherwise, start from turbulent profiles (DEFAULT=1).

NXSTRT = Index of initial section. (must be 1).

NXT = Index of final section to which computation proceeds.

NZSTRT = Index of initial girthwise line (Must be 1).

NZT = Index of final girthwise line to which computation proceeds.

TAPEIN = Logical unit number for subsequent NAMELIST input (DEFAULT=5).

TAPEOT = Logical unit number for printed output (DEFAULT=6).

TAPEGP = Logical unit number for input diskfile GEOMETRY (DEFAULT=16).

TAPEPF = Logical unit number for temporary diskfile-this is a very large file (DEFAULT=17).

TAPEDT = Logical unit number for temporary diskfile (DEFAULT=18).

TAPEVL = Logical unit number for input diskfile VELOCITY (DEFAULT=19).

VGP = Geometric progression ratio for normal grid.

ISHORT = Flag for slightly abbreviated print. ISHORT=0 deletes details of convergence parameters. Otherwise all details are given (DEFAULT=0).

RFTRB = Relaxation coefficient for turbulence model equations (DEFAULT=.8).

RFVEL = Relaxation coefficient for momentum equations (DEFAULT=1).

NAMELIST/DATA/: This namelist defines the various turbulence-model closure coefficients.

SALFA = The closure coefficient α (DEFAULT=1.11111).

SALFAS = The closure coefficient α^* (DEFAULT=.3).

SBETA = The closure coefficient β (DEFAULT=.15).

SBETAS = The closure coefficient β^* (DEFAULT=.09).

SLAMDA = The closure coefficient λ (DEFAULT=.091).

SSIGMA = The closure coefficient σ (DEFAULT=.5).

SSGMAS = The closure coefficient σ^* (DEFAULT=.5)

USTOP = The value of $u^+=u/u_\tau$ below which turbulent dissipation rate is analytically prescribed (DEFAULT=4).

ZIOTAE = Dimensionless value of turbulent mixing energy at the boundary-layer edge (DEFAULT=$3.75\cdot10^{-5}$).

ZIOTAL = Ratio of turbulent length scale to boundary layer thickness (DEFAULT=.09).

RW2 = The closure coefficeint R_ω (DEFAULT=4.).

NAMELIST/STRT/: This namelist provides input parameters needed to define turbulent starting profiles.

ALAMI = Array of values at each girthwise station for ratio of turbulent mixing energy to the corresponding flat-plate value.

CFXI = Array of streamwise skin friction, $C_{f_x}=\tau_{w_x}/(\frac{1}{2}\rho U^2)$, at each girthwise station.

CFZI = Array of crossflow skin friction, $C_{f_z}=\tau_{w_z}/(\frac{1}{2}\rho U^2)$, at each girthwise station.

DELTAI = Array of boundary-layer thickness at each girthwise station.

THETXI = Array of streamwise momentum thickness at each girthwise station.

THETZI = Array of crossflow momentum thickness at each girthwise station (NOTE: This array must be the same as THETXI in the present form of the program.).

The final input is a line defining the freestream velocity, static pressure and kinematic viscosity. These quantities are dimensional and the format is (3E12.4).

Printed program output is mostly self explanatory and will thus be described only briefly here. The first part of the printout lists all input parameters. Then, on every section, key integral parameters such as c_{f_x}, c_{f_z}, δ_x^*, δ_z^*, θ_x, θ_z are given at all girthwise stations. At the specified IPX and IPZ intervals, velocity and turbulence parameter profiles are printed.

The program also writes a special diskfile (use logical unit number 9) containing computed displacement thickness. This diskfile is used in program INTERP (next Subsection) to allow repeating the computation with displacement effects included.

2.2.7 Program INTERP

The purpose of this program is to interpolate the computed displacement thickness distribution onto the orthogonal rectangular coordinate system used by program SHIP. The whole computational procedure can then be repeated to account for displacement effects.

The program is written in BASIC and uses an extremely efficient MACHINE LANGUAGE interpolation routine referred to as USR1 in the program.

The first two inputs to the program are prompted by the video messages "NXTL=?" and "NZTL=?"; the values of NXTL and NZTL used in running EDDY3 must be entered. The next input is prompted by the message "FILE NAME FOR NONORTHOGONAL MESH ?"; the response is the name given to the print file generated by program SHPMSH, viz, FORTu5/DAT. The option is included to enter the file name in case the user opts to rename the file after running program SHPMSH. The final input is prompted by the messages "IMAX=?" and "JMAX=?"; the values of IMAX and JMAX used in running program PRMESH must be used.

Program output is a revised diskfile MESHPR containing the modified ship-hull coordinates. The original MESHPR is renamed MESHPR/OLD. The contents of the new MESHPR diskfile can be examined using program SPLOT (Subsection 2.2.2). Then, the entire computational sequence is repeated to determine the effect of including displacement thickness in the inviscid-flow computation.

2.3 APPLICATIONS

In this section we exercise the computational package to exhibit accuracy and capabilities for two test cases. We first compute the inviscid flow past a very simple hull investigated by Michell[1]. Then, we do both inviscid and viscous computations for the SSPA 720 ship hull[3].

2.3.1 Michell's Example

Our first test case is the sinusoidal hull form investigated by Michell. The hull coordinates are defined by

$$\bar{y} = -\ .02\left[1 + \cos(2\pi\bar{x})\right]\left[1 + \cos(10\pi\bar{z})\right] \qquad (2)$$

Michell found that for a Froude number based on hull length of 0.25, the wave-drag coefficient is

$$C_R \equiv \frac{R}{\frac{1}{2}\rho U^2 L^2} \doteq 0.60\cdot 10^{-4} \qquad (3)$$

In our computations we use 10 equally-spaced mesh points in the vertical direction. The number of points used in the streamwise direction has been varied to determine the number needed to achieve acceptable engineering accuracy. Again using equally-spaced points, computations have been made for 11, 21 and 41 points. Table 1 summarizes total wave-drag coefficient.

Table 1. Predicted Wave Drag

No. of Streamwise Mesh Points	$10^4 C_R$
11	0.49
21	0.52
41	0.56
Michell	0.60

Figure 3 provides an interesting view of the rate of convergence of the method. The figure shows the local drag coefficient as a function of distance along the hull. Note that the peak values near $2x/L = -0.25$ and $2x/L = +0.25$ are almost an order-of-magnitude larger than C_R.

As a test of the sensitivity to vertical-mesh-point spacing, we have repeated the 11-streamwise-point computation with only 6 points in the vertical direction. The predicted C_R is 1.3% lower than the value listed in Table 1.

Based on results obtained in these computations, accuracy acceptable for preliminary design (10%) can be obtained on a mesh of about 15 streamwise points by 6 vertical points. Such a run takes about 45 minutes on a TRS-80 Model II microcomputer.

2.3.2 SSPA Model 720

Turning now to the SSPA Model 720, we exercise both the inviscid and the viscous segments of the computational package. However, for reasons which become evident below, we do not exercise the displacement-thickness correction phase for this hull.

Figures 4 and 5 show computed streamwise velocity distribution near the free surface and along the keel, respectively. In all cases

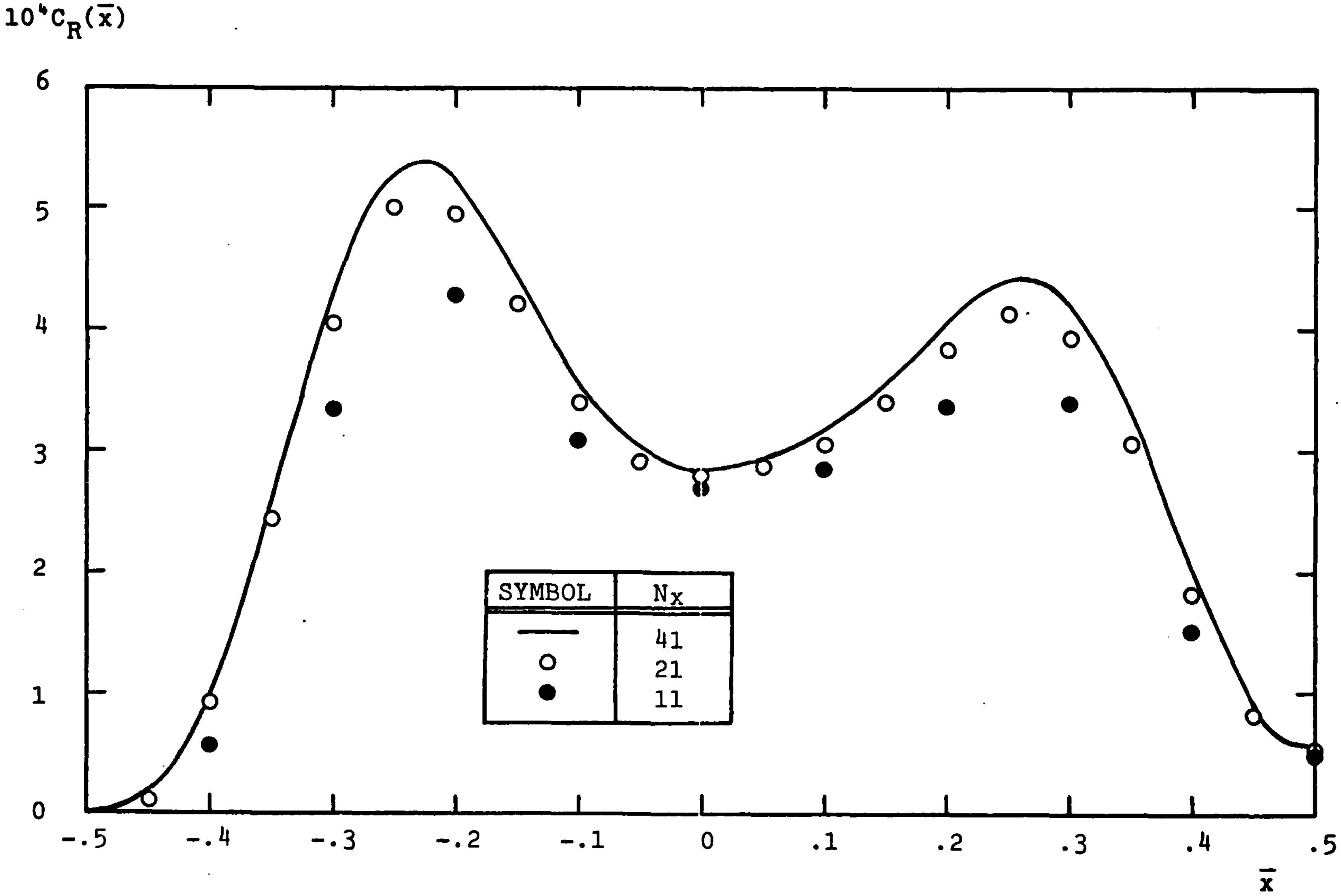


Figure 3. Variation of predicted wave-drag distribution with streamwise mesh-point number, N_x.

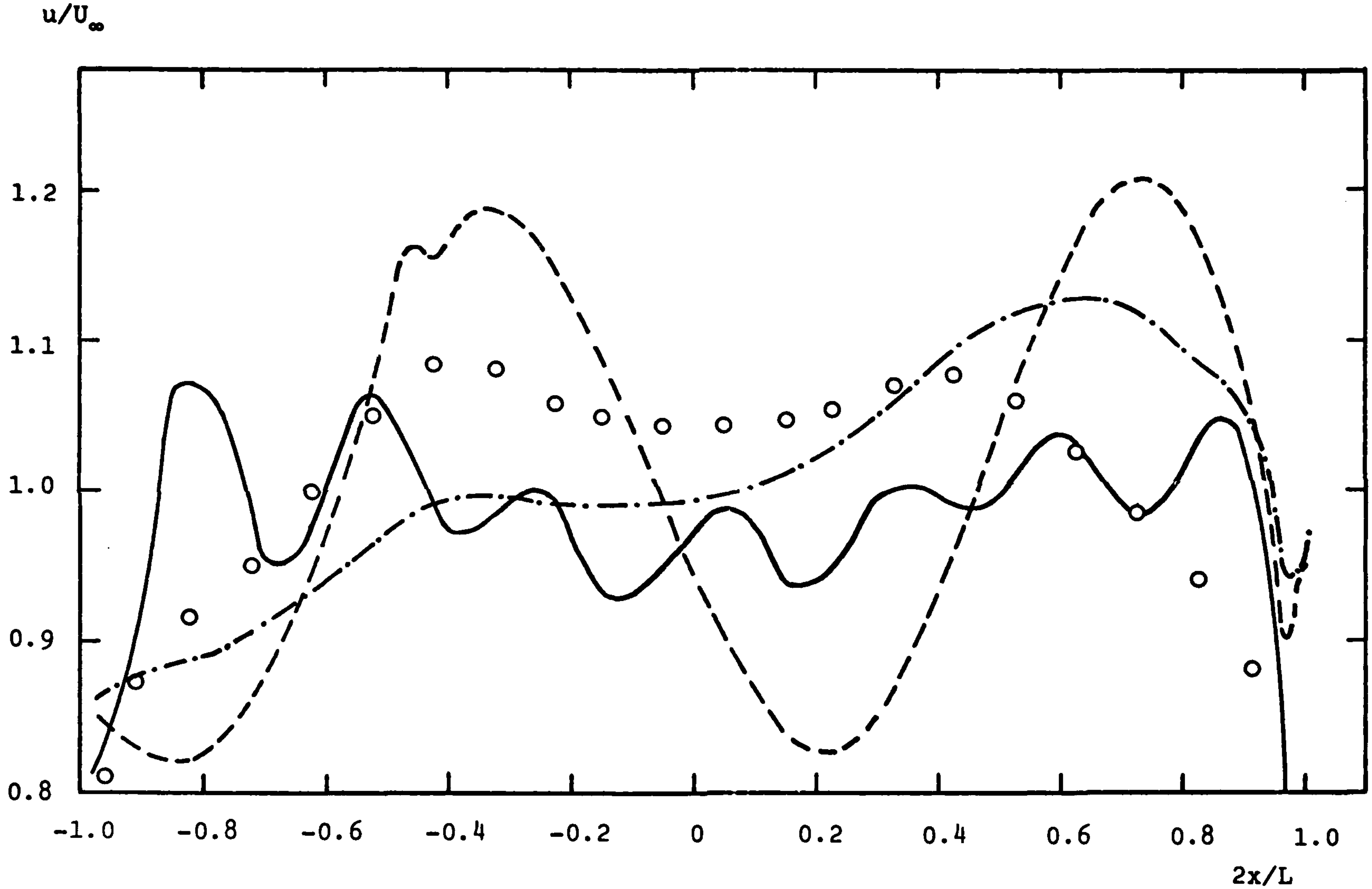


Figure 4. Computed velocity distribution on a line near the waterline (2z/L=-0.025): ———, Fr=0.15; – – – –, Fr=0.30; —·—, Fr=0.60; o , Douglas-Neumann.

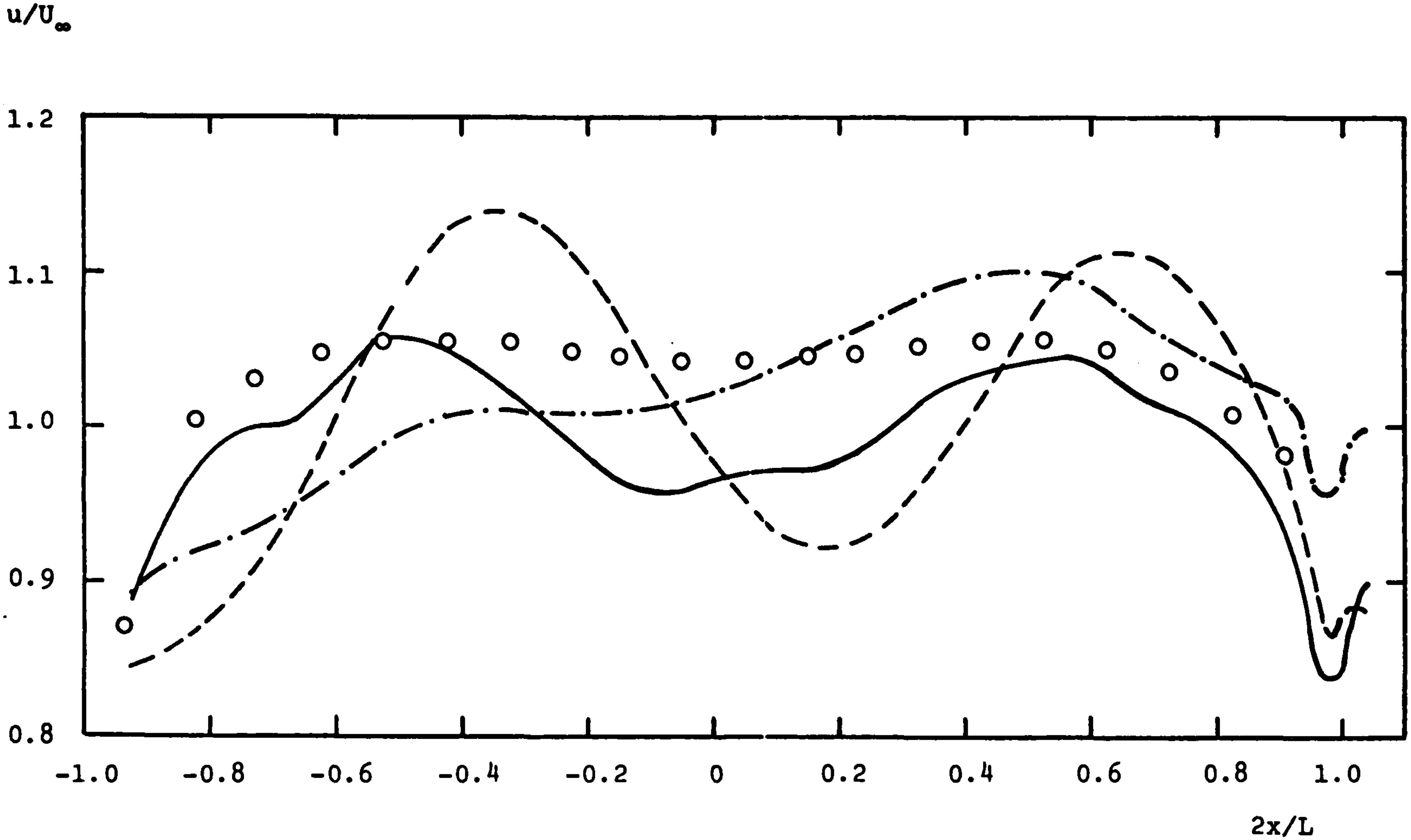


Figure 5. Computed velocity distribution along the keel (2z/L=0): ——— , Fr=0.15; – – – –, Fr=0.30; —·—, Fr=0.60; ○ , Douglas-Neumann.

the finite-difference grid consists of 41 points in the stream-wise direction and 11 points girthwise. Each figure shows velocity distributions for Froude numbers based on total hull length of 0.15, 0.30 and 0.60. Also shown in the figures are corresponding velocity distributions obtained from a Douglas-Neumann[5] "double-hull" computation.

Near the waterline the Fr = 0.15 distribution is quite oscillatory, a feature characteristic of thin-ship theory in the limit of small Froude number. In the region between 2x/L = -0.30 and 2x/L = 0.20, the Fr = 0.30 distribution has a strong adverse gradient which almost certainly would lead to boundary-layer separation. For Fr = 0.60, the near-waterline distribution corresponds to flow acceleration over the first 70% of the hull. In contrast to all of the thin--ship-theory predictions, the "double-hull" computation indicates a more-or-less symmetrical distribution with nearly constant velocity over the central 40% of the hull. Figure 5 indicates similar trends along the keel with the exception that no oscillations appear for the Fr = 0.15 computation.

Because the Fr = 0.60 distributions appear to be the least likely to cause numerical difficulty in running EDDY3, we have selected Fr = 0.60 as most appropriate for the test case (the actual value is not available in our literature on this hull).

Figure 6 compares computed and measured momentum thickness, θ. The figure also shows computed results obtained using Douglas-Neumann velocities. Lines A, B and C lie along the keel, about 15% of the way between keel and waterline, and about midway between keel and waterline, respectively. As shown, the most notable difference is the rapid increase in θ along Lines A and B downstream of 2x/L = 0.20. We predict boundary-layer separation at 2x/L = 0.36 over a significant portion of the hull. Although this result at first appears inconsistent with the velocity distributions of Figures 4 and 5, there is no contradiction. The predicted crossflow velocities are quite large ($w/U \sim 0.25$) and along streamlines we indeed predict relatively large adverse gradients approaching 2x/L = 0.36.

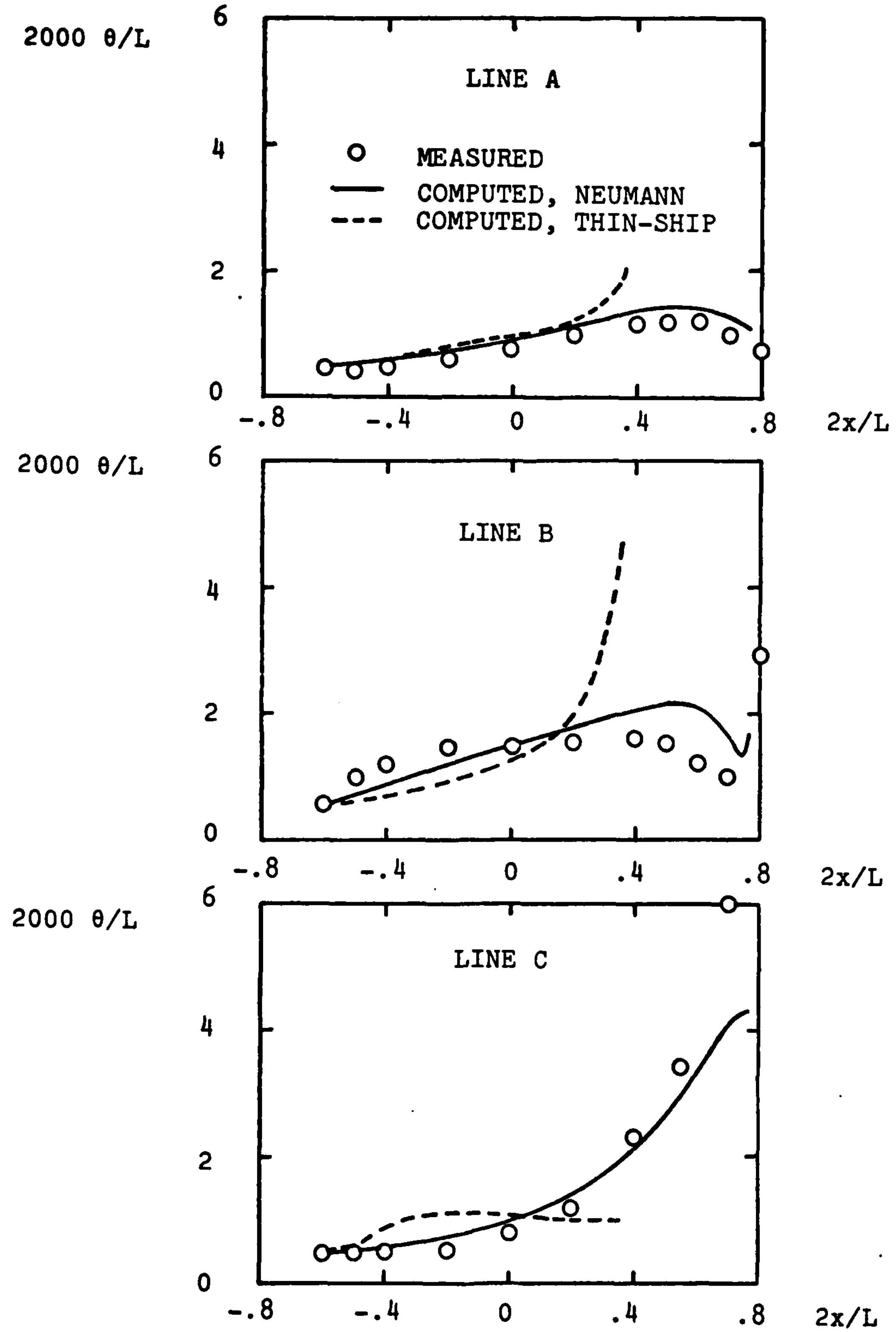


Figure 6. Comparison of computed and measured momentum thickness for the SSPA Model 720.

Because the numerical predictions fail to cover such a large region of the hull, there is little point in reiterating to account for displacement effects at this time.

3. DISCUSSION

The foundation has been developed for a microcomputer-based interactive design procedure for general ship hulls. Typical computing time for all but the boundary-layer computation are of the order of 3 to 4 hours on a TRS-80 Model II microcomputer. Using a 16-bit microprocesser, these computing times would be reduced to less than a half hour. Typically, EDDY3 runs in about 20 minutes on a UNIVAC 1108 which translates to about 3 hours on a 16-bit microcomputer. Thus, the computational package constitutes a practical "numerical towing tank" which can be used to effect and test design modifications in a single workday.

On the one hand, the weakest link in the computational package is program SHIP which determines the inviscid velocities. Based on classical thin-ship theory, its limitations are well known. On the other hand, program EDDY3 embodies one of the most tested and validated turbulence models available. The model has even been shown to accurately predict properties of "thick" boundary layers typical of those near the stern of a ship hull[6].

Research in the immediate future should focus upon improving the ability to predict the inviscid velocities. The formulation of Noblesse and Dagan[7], which amount to a coordinate transformation using the thin-ship velocity potential, should be incorporated in program SHIP. Also, normal pressure gradient in the boundary-layer computation should be accounted for to provide accurate predictions when the boundary layer becomes "thick". These two steps alone would yield a computational package which could be immediately applied by serious hull designers. To further aid the designer, a master program should be devised to coordinate input data preparation for the various program modules. Additionally, available graphics packages should be adapted to the computational procedure to permit more-detailed display and examination of calculated mesh and flow details.

Long range research topics should include: (a) numerical studies to aid development of a more-accurate integral-method treatment of the boundary layers, and (b) development of an approximate engineering-oriented procedure for treating hulls on which the boundary layer separates. The modular structure of the computational package developed in this project is readily adaptable to such advanced developments.

REFERENCES

1. Michell, J.H., "The Wave-Resistance of a Ship", Proc Roy Soc, Phil Mag (5), Vol 45, pp 106-123 (1898).

2. Wilcox, D.C. and Rubesin, M.W., "Progress in Turbulence Modeling for Complex Flow Fields Including Effects of Compressibility," NASA TP 1517 (1980).

3. Larsson, L., "SSPA-ITTC Workshop on Ship Boundary Layers" (1980).

4. Wilcox, D.C., "User's Manual for PROGRAM SMOOTH: A Least-Squares Cubic-Spline Fit Routine," DCW Industries Report DCW-R-16-02 (1976).

5. Hess,J.L. and Smith, A.M.O., "Calculation of Potential Flow About Arbitrary Bodies," Prog Aero Sci, Vol 8, Pergammon Press, Oxford and New York (1966).

6. Wilcox, D.C., "Boundary-Layer Development on Ship Hulls," DCW Industries Report DCW-R-26-01 (1983).

7. Noblesse, F. and Dagan, G., "Nonlinear Ship Wave Theories by Continuous Mapping," Institute of Hydraulic Research, Univ of Iowa (1975).

APPENDIX: PROGRAM LISTINGS

This Appendix contains listings of the seven program modules which constitute the computational package. In the listings, the number zero appears with a slash to distinguish it from the letter "O". The various programs are located on the following pages:

```
      PROGRAM PRMESH
C    ************************************************************
C    CREATE RECTANGULAR MESH FOR USE IN PROGRAM SHIP
C    ************************************************************
      DIMENSION XB(50),Y(41,20),YB(50,20),YI(50,20),ZB(20)
      COMMON/DATA/  X(41),Z(20)
      COMMON/NDATA/ IMAX,JMAX,NXTL,NZTL
      CALL NAMIN
      NZTLM=NZTL-1
      DO 10 I=1,NXTL
      READ(9,100) XB(I),YB(I,NZTL),ZB(NZTL),YB(I,NZTLM),ZB(NZTLM)
      DO 1 J=3,NZTLM,2
      JJ=NZTL-J+1
      JM=J-1
 1    READ(9,110) YB(I,JJ),ZB(JJ),YB(I,JM),ZB(JM)
      JN=1
      JS=2
      ZN=ZB(1)
      ZS=ZB(2)
      DO 5 J=1,JMAX
 2    IF(Z(J).LT.ZS.OR.ZN.EQ.ZS) GO TO 3
      YI(I,J)=YB(I,JN)+(YB(I,JS)-YB(I,JN))*(Z(J)-ZN)/(ZS-ZN)
      GO TO 5
 3    JN=JN+1
      IF(JN.GE.NZTL) GO TO 4
      JS=JS+1
      ZN=ZB(JN)
      ZS=ZB(JS)
      GO TO 2
 4    YI(I,J)=0.
 5    CONTINUE
 10   CONTINUE
      IW=1
      IE=2
      XW=XB(1)
      XE=XB(2)
      DO 15 I=1,IMAX
 11   IF(X(I).GT.XE.OR.XW.EQ.XE) GO TO 13
      DO 12 J=1,JMAX
 12   Y(I,J)=YI(IW,J)+(YI(IE,J)-YI(IW,J))*(X(I)-XW)/(XE-XW)
      GO TO 15
 13   IW=IW+1
      IE=IW+1
      XW=XB(IW)
      XE=XB(IE)
      GO TO 11
 15   CONTINUE
      DO 20 I=1,IMAX
 20   WRITE(8,200) X(I)
      DO 30 J=1,JMAX
 30   WRITE(8,200) Z(J)
      DO 40 I=1,IMAX
      DO 40 J=1,JMAX
 40   WRITE(8,200) Y(I,J)
 100  FORMAT(3F10.5,11X,2F10.5)
 110  FORMAT(10X,2F10.5,11X,2F10.5)
 200  FORMAT(1PE13.6)
      END
      SUBROUTINE NAMIN
C    ************************************************************
```

```
C   READ INPUT DATA FILE
C   ********************
C   DESCRIPTION OF 'PRMESHIN' INPUT DATA:
C      IMAX...NUMBER OF POINTS IN STREAMWISE DIRECTION: OUTPUT
C      JMAX...NUMBER OF POINTS IN  VERTICAL  DIRECTION: OUTPUT
C      NXTL...NUMBER OF POINTS IN STREAMWISE DIRECTION: INPUT
C      NZTL...NUMBER OF POINTS IN  VERTICAL  DIRECTION: INPUT
C      X(I)...SPECIFIED STREAMWISE COORDINATE ARRAY: OUTPUT
C      Z(J)...SPECIFIED  VERTICAL  COORDINATE ARRAY: OUTPUT
C   ************************************************************
       COMMON/DATA/  X(41),Z(20)
       COMMON/NDATA/ N(4)
       CALL OPEN(7,'PRMESHIN ',1)
       CALL OPEN(8,'MESHPR ',1)
       CALL OPEN(9,'MESH ',1)
       DO 1 I=1,4
       N(I) = 0
 1     CALL NAMLST(7,2,N(I),DUMMY)
       IMAX=N(1)
       JMAX=N(2)
       DO 2 I=1,IMAX
 2     CALL NAMLST(7,2,1,X(I))
       DO 3 J=1,JMAX
 3     CALL NAMLST(7,2,1,Z(J))
       RETURN
       END
       SUBROUTINE NAMLST(II,IO,N,X)
C   ************************************************************
C   READ VARIABLE NAME AND VALUE
C   ************************************************************
       DIMENSION A(3)
       IF(N.NE.0) GO TO 1
       READ(II,2) A(1),A(2),A(3),N
       WRITE(IO,2) A(1),A(2),A(3),N
       RETURN
 1     READ(II,3) A(1),A(2),A(3),X
       WRITE(IO,3) A(1),A(2),A(3),X
       RETURN
 2     FORMAT(1X,3A4,I4)
 3     FORMAT(1X,3A4,1PE13.6)
       END
```

```
10 REM ***********************************************************
20 REM **********  PLOTTING ROUTINE FOR SHIP SECTIONS **********
30 REM **********    DISK FILE 'MESHPR' CREATED BY    **********
40 REM **********              PROGRAM PRMESH          **********
50 REM ***********************************************************
60 DIM X(41),Y(20),Z(20)
70 NZ=70
80 NY=20
90 NL=9
100 IC=0
110 OPEN "I", 1, "MESHPR"
120 INPUT "IMAX ="; IM
130 INPUT "JMAX ="; JM
140 FOR I=1 TO IM
150 INPUT#1, X(I)
160 NEXT I
170 FOR J=1 TO JM
180 INPUT#1, Z(J)
190 NEXT J
200 IC=IC+1
210 IF IC>IM THEN 1600
220 FOR J=1 TO JM
230 INPUT#1, Y(J)
240 Y(J)=-Y(J)
250 NEXT J
260 GOSUB 620
270 CLS
280 PRINT "I ="; IC; "        X(I) ="; X(IC)
290 PRINT:PRINT
300 PRINT "ZMIN = "; ZL; "    ZMAX = "; ZU
310 PRINT
320 PRINT "YMIN = "; YL; "    YMAX = "; YU
330 PRINT:PRINT
340 PRINT "SPECIFY PLOTTING RANGE"
350 PRINT
360 IF IC=1 THEN 430
370 PRINT "SAME RANGE ?"
380 Q$=INKEY$
390 IF Q$="" THEN 380
400 IF LEFT$(Q$,1)<>"Y" THEN 430
410 GOSUB 1550
420 GOTO 510
430 INPUT "ZMIN ="; ZL
440 INPUT "ZMAX ="; ZU
450 INPUT "YMIN ="; YL
460 INPUT "YMAX ="; YU
470 GOSUB 1490
480 PRINT
490 INPUT "ANY ERRORS"; Q$
500 IF LEFT$(Q$,1)="Y" THEN 340
510 CLS
520 GOSUB 770
530 GOSUB 1060
540 GOSUB 1130
550 A$=INKEY$
560 IF A$="" THEN 550
570 IF A$="C" THEN 260
580 IF A$="N" THEN 200
590 END
```

```
600 REM **********  FIND MIN & MAX VALUES  **********
610 REM --- MIN VALUE ---
620 ZL=1.E20
630 YL=1.E20
640 FOR I=1 TO JM
650 IF Z(I)<ZL THEN ZL=Z(I)
660 IF Y(I)<YL THEN YL=Y(I)
670 NEXT I
680 REM --- MAX VALUE ---
690 ZU=ZL
700 YU=YL
710 FOR I=1 TO JM
720 IF Z(I)>ZU THEN ZU=Z(I)
730 IF Y(I)>YU THEN YU=Y(I)
740 NEXT I
750 RETURN
760 REM **********  DRAW AXES AND TIC MARKS  **********
770 ZZ=ZU-ZL
780 YY=YU-YL
790 REM --- Y-AXIS TIC MARKS ---
800 IY%=1
810 FOR I=1 TO 5
820 PRINT@(IY%,NL-2), CHR$(95);
830 PRINT@(IY%,NL-1), CHR$(95)
840 IY%=IY%+5
850 NEXT I
860 REM --- DRAW THE Y-AXIS ---
870 IY%=2
880 II=NY+1
890 FOR I=IY% TO II
900 PRINT@(I,NL), CHR$(156)
910 NEXT I
920 REM --- Z-AXIS TIC MARKS ---
930 II=NZ+NL
940 IS=NZ/5
950 FOR I=NL TO II STEP IS
960 PRINT@(NY+2,I), CHR$(156);
970 NEXT I
980 REM --- DRAW THE Z-AXIS ---
990 II=NZ+NL-1
1000 NN=NL+1
1010 FOR I=NN TO II
1020 PRINT@(NY+1,I), CHR$(95)
1030 NEXT I
1040 RETURN
1050 REM ********** LABEL AXES **********
1060 PRINT@(0,0) , YU;
1070 PRINT@(NY+1,0) , YL;
1080 PRINT@(NY+3,NL-2) , ZL;
1090 PRINT@(NY+3,NZ+NL-6) , ZU;
1100 PRINT@(0,NZ/2+NL-4), "X ="; X(IC);
1110 RETURN
1120 REM **********  PLOT THE POINTS  **********
1130 ID=0
1140 FOR I=1 TO JM
1150 C=NL+INT(NZ*(Z(I)-ZL)/ZZ+.5)
1160 R=NY+1-INT(NY*(Y(I)-YL)/YY)
1170 IF C>(NZ+NL) THEN 1270
1180 IF C<NL THEN 1270
```

```
1190 IF R>(NY+1) THEN 1270
1200 IF R<1 THEN 1270
1210 GOSUB 1360
1220 IF GC=160 THEN 1240
1230 PRINT@(R,C), CHR$(GC);
1240 PRINT@(R+1,C), CHR$(26)+CHR$(GC);
1250 PRINT CHR$(25);
1260 GOTO 1280
1270 ID=ID+1
1280 NEXT I
1290 PRINT@(7,1), ID;
1300 PRINT@(8,0), "POINTS";
1310 PRINT@(9,1), "OFF";
1320 PRINT@(10,0), "SCALE";
1330 PRINT CHR$(02);
1340 RETURN
1350 REM ********** INTERPOLATE WITHIN BOX **********
1360 RR=R-(NY+1-NY*(Y(I)-YL)/YY)
1370 GC=154
1380 IF RR<.2 THEN RETURN
1390 GC=155
1400 IF RR<.4 THEN RETURN
1410 R=R-1
1420 GC=160
1430 IF RR<.6 THEN RETURN
1440 GC=152
1450 IF RR<.8 THEN RETURN
1460 GC=153
1470 RETURN
1480 REM **********  SAVE PLOTTING RANGES  **********
1490 XL=ZL
1500 XU=ZU
1510 WL=YL
1520 WU=YU
1530 RETURN
1540 REM **********  RETRIEVE PLOTTING RANGES  **********
1550 ZL=XL
1560 ZU=XU
1570 YL=WL
1580 YU=WU
1590 RETURN
1600 END
```

```
      PROGRAM SHIP
C     ******************************************************************
C     THIS PROGRAM USES LINEAR SHIP THEORY TO COMPUTE THE WAVE DRAG
C     AND SURFACE VELOCITY COMPONENTS FOR AN ARBITRARY VESSEL.
C     ******************************************************************
      COMMON/COORD/ XB(41),YB(41,10),ZB(10),FB(40,9)
      COMMON/DATA/  FR,IMAX,JMAX,MINT2,MINT3,NZOT,IMAXM,JMAXM
      COMMON/QUADRA/AM(80),RSAV(80),TSAV(80),WM
      COMMON/VELO/  U(40,9),W(40,9)
      CALL NAMIN
      FR2R=1./FR**2
      DO 1 I=1,IMAX
      READ(7,100) XB(I)
    1 XB(I)=FR2R*XB(I)
      DO 2 J=1,JMAX
      READ(7,100) ZB(J)
    2 ZB(J)=FR2R*ZB(J)
      DO 3 I=1,IMAX
      DO 3 J=1,JMAX
      READ(7,100) YB(I,J)
    3 YB(I,J)=FR2R*YB(I,J)
      CALL GEOM(IMAX,JMAX)
      RWAVE=0.
      DO 4 I=1,IMAXM
      DO 4 J=1,JMAXM
      U(I,J)=0.
    4 W(I,J)=0.
      CALL VELOC1
      WRITE(1,510)
      CALL VELOC2
      WRITE(1,520)
      CALL VELOC3
      WRITE(1,530)
      DO 10 I=1,IMAXM
      CALL DRAG(DRWAVE,I)
      RWAVE=RWAVE+DRWAVE
      IF(NZOT.EQ.0) CALL OUTPUT(I)
      WRITE(2,400) DRWAVE,RWAVE
   10 CONTINUE
      WRITE(2,200) RWAVE
      WRITE(1,200) RWAVE
  100 FORMAT(E13.6)
  200 FORMAT(//////,1X18HCOMPUTED WAVE DRAG///4X7HRWAVE =,1PE12.4)
  400 FORMAT(//,1X,'INCREMENTAL WAVE DRAG ON THIS SECTION IS ',1PE13.4/
     *           ,1X,'TOTAL DRAG COMPUTED UP TO THIS SECTION IS',E13.4)
  510 FORMAT(1X,'VELOC1 COMPUTATIONS COMPLETE')
  520 FORMAT(1X,'VELOC2 COMPUTATIONS COMPLETE')
  530 FORMAT(1X,'VELOC3 COMPUTATIONS COMPLETE')
      END
      FUNCTION ACOS(X)
C     ******************************************************************
C     THIS SUBROUTINE COMPUTES THE ARC COSINE
C     ******************************************************************
      DATA PH/1.5707963268/
      ACOS=PH-ASIN(X)
      RETURN
      END
      FUNCTION ASIN(X)
C     ******************************************************************
C     THIS SUBROUTINE COMPUTES THE ARC SINE
```

```
C     ******************************************************************
      DATA A0,A1,A2/1.5707963050,-.2145988016,.0889789874/
      DATA A3,A4,A5/-.0501743046,.0308918810,-.0170881256/
      DATA A6,A7,PH/.0066700901,-.0012624911,1.5707963268/
      Y=ABS(X)
      ASIN=PH-SQRT(1.-Y)*(A0+Y*(A1+Y*(A2+Y*(A3+Y*(A4+Y*(A5
     *                              +Y*(A6+Y*A7)))))))
      IF(X.LT.0.) ASIN=-ASIN
      RETURN
      END
      SUBROUTINE DRAG(DRWAVE,I)
C     ******************************************************************
C     THIS SUBROUTINE COMPUTES THE WAVE DRAG
C     ******************************************************************
      COMMON/COORD/ XB(41),YB(41,10),ZB(10),FB(40,9)
      COMMON/DATA/  FR,IMAX,JMAX,MINT2,MINT3,NZOT,IMAXM,JMAXM
      COMMON/VELO/  U(40,9),W(40,9)
      DRWAVE=0.
      DX=XB(I+1)-XB(I)
      DO 10 J=2,JMAX
      DZ=ZB(J-1)-ZB(J)
   10 DRWAVE=DRWAVE+U(I,J-1)*FB(I,J-1)*DX*DZ
      DRWAVE=4.*DRWAVE*FR**4
      RETURN
      END
      SUBROUTINE GEOM(IMAX,JMAX)
C     ******************************************************************
C     THIS SUBROUTINE COMPUTES CELL SLOPE
C     ******************************************************************
      COMMON/COORD/ XB(41),YB(41,10),ZB(10),FB(40,9)
      DO 10 I=2,IMAX
      DX=XB(I)-XB(I-1)
      DO 10 J=2,JMAX
   10 FB(I-1,J-1)=.5*(YB(I,J)+YB(I,J-1)-YB(I-1,J)-YB(I-1,J-1))/DX
      RETURN
      END
      SUBROUTINE INTK0(G,Z,RMU)
C     ******************************************************************
C     THIS SUBROUTINE APPROXIMATES THE BESSEL FUNCTION INTEGRAL WITH A
C     GROUP OF CLOSED FORM ANALYTICAL FUNCTIONS.
C     ******************************************************************
      ZMU=Z*RMU
      IF(ZMU.LE.0) GO TO 20
      IF(ZMU.LE..5) G=ZMU*(1.-ALOG(ZMU))
      IF(ZMU.GT..5) G=1.2*SQRT(ZMU)
      ZMU2=ZMU*RMU
      IF(Z.LT.3.) GO TO 40
      XI=3./Z
      GMAX=1.+.571*(1.-XI)*(.37+.63*EXP(-10.*XI))
      IF(GMAX.LT.G) GO TO 10
      RETURN
   10 G=GMAX
      IF(Z.GE.10.) GO TO 80
      GO TO 60
C -------- MU*Z IS ZERO --------
   20 G=0.
      RETURN
C ----- Z BETWEEN 0 AND 3  -----
   40 RMUT=1.+1.114*Z**(-.6666667)
      IF(RMU.GE.RMUT) GO TO 55
```

```
      IF(Z.GE.1) GO TO 45
      ZP25=Z**.25
      GM=.83*ZP25
      RN=.62+1.68*ZP25
      GO TO 50
   45 GM=.83+(Z-1.)*(.12-.02*(Z-1.))
      RN=2.+.3*Z
   50 G2=GM/(1.+.25*ZMU2**RN)
      GO TO 100
   55 G=SQRT(1.5707963/ZMU)*EXP(-ZMU)/(RMU-1.)
      RETURN
C ----- Z BETWEEN 3 AND 10 -----
   60 RMUT=1.+1.114*Z**(-.6666667)
      IF(RMU.GE.RMUT) GO TO 65
      XI=.1*(Z-3.)
      A=1.38+XI*(.76-.54*XI)
      RM=.68+XI*(.53-.41*XI)
      G2=A*EXP(-RM*ZMU2)
      GO TO 100
   65 G=SQRT(1.5707963/ZMU)*EXP(-ZMU)/(RMU-1.)
      RETURN
C ----- Z GREATER THAN 10  -----
   80 G2=0.
      IF(RMU.LT.1.) G2=(1.571+ASIN(RMU))*EXP(-ZMU2)
     *                   /SQRT(1.-RMU*RMU)
  100 IF(G2.LT.G) G=G2
      RETURN
      END
      SUBROUTINE NAMIN
C   ****************************************************
C   READ INPUT DATA FILE
C   ********************
C   DESCRIPTION OF 'SHIPIN' INPUT DATA:
C      FR.....FROUDE NUMBER = U/SQRT(G*L)
C      IMAX...NUMBER OF MESH POINTS IN STREAMWISE DIRECTION
C      JMAX...NUMBER OF MESH POINTS IN  VERTICAL  DIRECTION
C      MINT2..NUMBER OF POINTS IN QUADRATURE...INFINITE INTEGRAL
C      MINT3..NUMBER OF POINTS IN QUADRATURE.....FINITE INTEGRAL
C      NZOT...0 TO PRINT CELL GEOMETRY AND VELOCITY COMPONENTS
C             ...OTHERWISE PRINT ONLY THE WAVE DRAG
C   ****************************************************
      COMMON/DATA/ FR,N(5),IMAXM,JMAXM
      CALL OPEN(7,'MESHPR ',1)
      CALL OPEN(8,'SHIPIN ',1)
      CALL OPEN(9,'UEWE ',1)
      CALL NAMLST(8,2,1,FR)
      DO 1 I=1,5
      N(I) = 0
    1 CALL NAMLST(8,2,N(I),DUMMY)
      IMAXM=N(1)-1
      JMAXM=N(2)-1
      RETURN
      END
      SUBROUTINE NAMLST(II,IO,N,X)
C   ****************************************************
C   READ VARIABLE NAME AND VALUE
C   ****************************************************
      DIMENSION A(3)
      IF(N.NE.0) GO TO 1
      READ(II,2) A(1),A(2),A(3),N
```

```
      WRITE(IO,2) A(1),A(2),A(3),N
      RETURN
    1 READ(II,3) A(1),A(2),A(3),X
      WRITE(IO,3) A(1),A(2),A(3),X
      RETURN
    2 FORMAT(1X,3A4,I4)
    3 FORMAT(1X,3A4,1PE13.6)
      END
      SUBROUTINE OUTPUT(I)
C   ********************************************************************
C   THIS SUBROUTINE PRINTS SURFACE VELOCITY COMPONENTS AND CREATES DISK
C   FILE 'UEWE' FOR PROGRAM VELOC
C   ********************************************************************
      COMMON/COORD/ XB(41),YB(41,10),ZB(10),FB(40,9)
      COMMON/DATA/  FR,IMAX,JMAX,MINT2,MINT3,NZOT,IMAXM,JMAXM
      COMMON/VELO/  U(40,9),W(40,9)
      FR2=FR*FR
      XI=.5*FR2*(XB(I)+XB(I+1))
      WRITE(2,100) XI
      DO 1 J=1,JMAXM
      JJ=JMAX-J
      ZETA=.5*FR2*(ZB(JJ)+ZB(JJ+1))
      ETA=.5*FR2*(YB(I,JJ)+YB(I,JJ+1))
      UP=1.+U(I,JJ)
      VP=FB(I,JJ)
      WP=W(I,JJ)
      WRITE(2,200) J,ZETA,UP,VP,WP
      Q=SQRT(UP*UP+VP*VP+WP*WP)
      UP=UP/Q
      VP=VP/Q
      WP=WP/Q
    1 WRITE(9,1000) XI,ETA,ZETA,UP,VP,WP,Q
  100 FORMAT(/,1X38HCOMPUTED VELOCITY COMPONENTS ALONG X =,1PE12.4//
     *     4X1HJ,7X4HZ(J),11X4HU(J),11X4HV(J),11X4HW(J)/)
  200 FORMAT(I5,1P4E15.4)
 1000 FORMAT(3F10.5,4F10.6)
      RETURN
      END
      SUBROUTINE QUAD(M)
C   ********************************************************************
C   THIS SUBROUTINE COMPUTES QUADRATURE WEIGHTS AND ABSCISSAS
C   ********************************************************************
      COMMON/DATA/  FR,IMAX,JMAX,MINT2,MINT3,NZOT,IMAXM,JMAXM
      COMMON/QUADRA/AM(80),RSAV(80),TSAV(80),WM
      DATA  PI/3.14159265/,CON/.2026423/
C   EQUAL WEIGHT CHEBYSHEV INTEGRATION FOR DU3 AND DW3
      IF(M.EQ.2) GO TO 15
      COEF=PI/(4.*FLOAT(MINT3))
      WM=2.*CON*COEF
      DO 10 I=1,MINT3
      RI=FLOAT(I)
      AM(I)=COS((2.*RI-1.)*COEF)
      RSAV(I)=SQRT(1.-AM(I)**2)
   10 TSAV(I)=ACOS(AM(I))+PI
      RETURN
C   EQUAL WEIGHT CHEBYSHEV INTEGRATION FOR DU2 AND DW2
   15 COEF=PI/(4.*FLOAT(MINT2))
      WM=2.*CON*COEF
      DO 20 I=1,MINT2
      RI=FLOAT(I)
```

```
      AM(I)=COS((2.*RI-1.)*COEF)
      EM2=AM(I)**2
      RSAV(I)=SQRT(1.-EM2)/AM(I)
      ARGLOG=1./AM(I)-RSAV(I)
   20 TSAV(I)=ALOG(ARGLOG)
      RETURN
      END
      SUBROUTINE VELOC1
C    *******************************************************************
C    THIS SUBROUTINE COMPUTES THE CONTRIBUTION OF THE SOURCE-LIKE TERMS
C    TO THE SURFACE VELOCITY COMPONENTS
C    *******************************************************************
      COMMON/COORD/ XB(41),YB(41,10),ZB(10),FB(40,9)
      COMMON/DATA/  FR,IMAX,JMAX,MINT2,MINT3,NZOT,IMAXM,JMAXM
      COMMON/VELO/  U(40,9),W(40,9)
      DATA PI/3.14159265/
C    --------- INITIALIZATION ----------
      DO 40 I=1,IMAXM
      XI=.5*(XB(I)+XB(I+1))
      DO 40 J=1,JMAXM
      ZETA=.5*(ZB(J)+ZB(J+1))
      UINT=0.
      WINT=0.
      DO 30 II=1,IMAXM
      DO 30 JJ=1,JMAXM
      XP=XB(II)-XI
      XM=XB(II+1)-XI
      ZP=-(ZETA+ZB(JJ+1))
      ZM=-(ZETA+ZB(JJ))
      Z1P=ZB(JJ+1)-ZETA
      Z1M=ZB(JJ)-ZETA
      RA=SQRT(XP*XP+ZP*ZP)
      RB=SQRT(XM*XM+ZP*ZP)
      RC=SQRT(XM*XM+ZM*ZM)
      RD=SQRT(XP*XP+ZM*ZM)
      R1A=SQRT(XP*XP+Z1P*Z1P)
      R1B=SQRT(XM*XM+Z1P*Z1P)
      R1C=SQRT(XM*XM+Z1M*Z1M)
      R1D=SQRT(XP*XP+Z1M*Z1M)
      UINT=UINT+ALOG((R1A-Z1P)*(R1C-Z1M)*(RD+ZM)*(RB+ZP)
     *    /((R1B-Z1P)*(R1D-Z1M)*(RC+ZM)*(RA+ZP)))*FB(II,JJ)
      WINT=WINT+ALOG((R1A-XP)*(R1C-XM)*(RD-XP)*(RB-XM)
     *     /((R1D-XP)*(R1B-XM)*(RC-XM)*(RA-XP)))*FB(II,JJ)
   30 CONTINUE
      U(I,J)=U(I,J)+.5*UINT/PI
      W(I,J)=W(I,J)+.5*WINT/PI
   40 CONTINUE
      RETURN
      END
      SUBROUTINE VELOC2
C    *******************************************************************
C    THIS SUBROUTINE COMPUTES THE CONTRIBUTION OF THE INFINITE INTEGRAL
C    TO THE SURFACE VELOCITY COMPONENTS
C    *******************************************************************
      COMMON/COORD/ XB(41),YB(41,10),ZB(10),FB(40,9)
      COMMON/DATA/  FR,IMAX,JMAX,MINT2,MINT3,NZOT,IMAXM,JMAXM
      COMMON/INTG/  TCOS(41),TSIN(41),TEXP(9,10),TBES(9,10)
      COMMON/QUADRA/AM(80),RSAV(80),TSAV(80),WM
      COMMON/VELO/  U(40,9),W(40,9)
      DATA PI/3.14159265/
```

```
C  --------- INITIALIZATION -----------
       CALL QUAD(2)
       DO 100 M=1,MINT2
       EM=AM(M)
       EM2=EM*EM
       DO 10 J=1,JMAXM
       ZETA=.5*(ZB(J)+ZB(J+1))
       DO 10 JJ=1,JMAX
       ZED=-(ZB(JJ)+ZETA)
       EMR=1./EM
       CALL INTK0(G,ZED,EMR)
       TBES(J,JJ)=G*RSAV(M)
   10  TEXP(J,JJ)=EXP(-ZED/EM2)
C  ----- COMPUTE FOURIER INTEGRAL -----
       DO 40 I=1,IMAXM
       XI=.5*(XB(I)+XB(I+1))
       DO 20 II=1,IMAX
       TCOS(II)=COS((XB(II)-XI)/EM)
   20  TSIN(II)=SIN((XB(II)-XI)/EM)
       DO 40 J=1,JMAXM
       UINT=0.
       WINT=0.
       DO 30 II=1,IMAXM
       COSXOM=TCOS(II)-TCOS(II+1)
       SINXOM=TSIN(II)-TSIN(II+1)
       DO 30 JJ=1,JMAXM
       C=(TEXP(J,JJ+1)-TEXP(J,JJ))*FB(II,JJ)
       D=(TBES(J,JJ+1)-TBES(J,JJ))*FB(II,JJ)
       UINT=UINT+C*(PI*SINXOM+TSAV(M)*COSXOM)-D*COSXOM
   30  WINT=WINT+(C*(PI*COSXOM-TSAV(M)*SINXOM)-D*SINXOM)/EM
       U(I,J)=U(I,J)+WM*UINT
       W(I,J)=W(I,J)+WM*WINT
   40  CONTINUE
  100  CONTINUE
       RETURN
       END
       SUBROUTINE VELOC3
C  ****************************************************************
C  THIS SUBROUTINE COMPUTES THE CONTRIBUTION OF THE FINITE INTEGRAL
C  TO THE SURFACE VELOCITY COMPONENTS
C  ****************************************************************
       COMMON/COORD/  XB(41),YB(41,10),ZB(10),FB(40,9)
       COMMON/DATA/   FR,IMAX,JMAX,MINT2,MINT3,NZOT,IMAXM,JMAXM
       COMMON/INTG/   TCOS(41),TSIN(41),TEXP(9,10),TBES(9,10)
       COMMON/QUADRA/AM(80),RSAV(80),TSAV(80),WM
       COMMON/VELO/   U(40,9),W(40,9)
C  --------- INITIALIZATION -----------
       CALL QUAD(3)
       DO 100 M=1,MINT3
       EM=AM(M)
       EM2=EM*EM
       DO 10 J=1,JMAXM
       ZETA=.5*(ZB(J)+ZB(J+1))
       DO 10 JJ=1,JMAX
       ZED=-(ZB(JJ)+ZETA)
       CALL INTK0(G,ZED,EM)
   10  TEXP(J,JJ)=TSAV(M)*EXP(-ZED*EM2)+RSAV(M)*G
C  ----- COMPUTE FOURIER INTEGRAL -----
       DO 40 I=1,IMAXM
       XI=.5*(XB(I)+XB(I+1))
```

```
      DO 20 II=1,IMAX
      TCOS(II)=COS((XB(II)-XI)*EM)
20    TSIN(II)=SIN((XB(II)-XI)*EM)
      DO 40 J=1,JMAXM
      UINT=0.
      WINT=0.
      DO 30 II=1,IMAXM
      COSXMU=TCOS(II)-TCOS(II+1)
      SINXMU=TSIN(II)-TSIN(II+1)
      DO 30 JJ=1,JMAXM
      C=(TEXP(J,JJ+1)-TEXP(J,JJ))*FB(II,JJ)
      UINT=UINT-C*COSXMU/EM
30    WINT=WINT+C*SINXMU
      U(I,J)=U(I,J)+WM*UINT
      W(I,J)=W(I,J)+WM*WINT
40    CONTINUE
100   CONTINUE
      RETURN
      END
```

```
      PROGRAM SHPMSH
C ******************************************************************
C THIS PROGRAM CALCULATES METRIC AND CURVATURE PARAMETERS FOR ARBITRAR
C SHIP HULLS AND WRITES A DISK FILE FOR USE BY PROGRAM EDDY3
C ******************************************************************
      COMMON/COORD/ Z(40)
      COMMON/COORDB/XB(40),YB(40),ZB(40)
      COMMON/INTRP/ XI(40),ZI(40)
      COMMON/GEOMP/ H1(40),H2(40),THETA(40,40),CK1(40),CK2(40),
     *              CK12(40),CK21(40),PSI(40)
      COMMON/TEMPZ/ DYBDZ(40),D2YBDZ(40),DZBDZ(40),D2ZBDZ(40)
      COMMON/TEMPX/ DYBDX(40),D2YBDX(40),DZBDX(40),D2ZBDX(40)
      COMMON/DTHET/ DTHDX(40),DTHDZ(40),THET(40),LT(30)
      COMMON/XDATA/ PSIZ,XSTART,XEND,DX(10),NX(10)
      COMMON/ZDATA/ ZSTART,ZEND,DZ(10),NZ(10)
      COMMON/NDATA/ IU1,IU2,IU3,IU4,IU5,IMAX,JMAX,NXTL,NZTL,JFZ,LZ(30)
      COMMON/MDATA/ JFX,LX(30)
      COMMON/LDATA/ ISYM,ISET
      CALL OPEN(6,'NAMLIST ',1)
      CALL NAMIN
      CALL OPEN(IU1,'MESH ',1)
      CALL OPEN(IU2,'SCRATCH1 ',12)
      CALL OPEN(IU3,'SCRATCH2 ',28)
      CALL OPEN(IU4,'GEOMETRY ',32)
C SET UP COMPUTATIONAL GRID
      CALL GRID
      JMAXM=JMAX-1
      DO 1 I=1,IMAX
      ISET=0
C READ SECTION COORDINATES
      READ(IU1,1000) XB(I),YB(1),ZB(1),YB(2),ZB(2)
      DO 2 J=3,JMAXM,2
      JP=J+1
    2 READ(IU1,1100) YB(J),ZB(J),YB(JP),ZB(JP)
C COMPUTE AND RESCALE LOCAL ARCLENGTH
      CALL ARC(DYDX1)
C SMOOTH SECTION DATA AND COMPUTE D/DZ
      IF(ISYM.EQ.1) ISET=1
      CALL SMOOTH(Z,YB,LZ,JFZ,JMAX,1,NZTL,IU2,ZI,0,DYDX1)
      CALL SMOOTH(Z,ZB,LZ,JFZ,JMAX,1,NZTL,IU2,ZI,0,0.)
    1 CONTINUE
      WRITE(1,200)
      REWIND IU2
      DO 6 J=1,NZTL
C READ SMOOTHED DATA FOR CONSTANT Z
      CALL READYZ(J)
C SMOOTH ALONG Z=CONSTANT AND COMPUTE D/DX
      CALL SMOOTH(XB,YB,LX,JFX,IMAX,0,NXTL,IU3,XI,1,DUMMY)
      CALL SMOOTH(XB,ZB,LX,JFX,IMAX,0,NXTL,IU3,XI,2,DUMMY)
      CALL SMOOTH(XB,DYBDZ,LX,JFX,IMAX,0,NXTL,IU3,XI,4,DUMMY)
      CALL SMOOTH(XB,DZBDZ,LX,JFX,IMAX,0,NXTL,IU3,XI,5,DUMMY)
      CALL SMOOTH(XB,D2YBDZ,LX,JFX,IMAX,0,NXTL,IU3,XI,6,DUMMY)
      CALL SMOOTH(XB,D2ZBDZ,LX,JFX,IMAX,0,NXTL,IU3,XI,7,DUMMY)
C COMPUTE H1,H2,THETA,K1,K2
      CALL GEOMET(J)
      WRITE(1,300) J
      REWIND IU2
    6 CONTINUE
C COMPUTE DERIVATIVES OF THETA ON EACH SECTION
      REWIND IU2
```

```
      JFT=NZTL/2+1
      LT(1)=1
      JUP=NZTL-1
      LL=1
      DO 8 J=2,JUP,2
      LL=LL+1
8     LT(LL)=J
      LT(JFT)=NZTL
      DO 9 I=1,NXTL
      DO 7 J=1,NZTL
7     THET(J)=THETA(I,J)
      CALL SMOOTH(ZI,THET,LT,JFT,NZTL,0,NZTL,IU2,ZI,8,DUMMY)
      CALL SMOOTH(ZI,DTHDZ,LT,JFT,NZTL,0,NZTL,IU2,ZI,0,DUMMY)
9     CONTINUE
      WRITE(1,400)
      REWIND IU2
      REWIND IU3
      JFT=NXTL/2+1
      LT(1)=1
      IUP=NXTL-1
      LL=1
      DO 13 I=2,IUP,2
      LL=LL+1
13    LT(LL)=I
      LT(JFT)=NXTL
      DO 12 J=1,NZTL
C  READ D(THETA)/DZ FROM DISK FILE FOR CONSTANT Z
      CALL READTH(J)
C  SMOOTH THETA ALONG Z=CONSTANT AND COMPUTE D/DX
      DO 10 I=1,NXTL
10    READ(IU3) YB(I),ZB(I),H1(I),H2(I),THET(I),CK1(I),CK2(I)
      CALL SMOOTH(XI,THET,LT,JFT,NXTL,0,NXTL,IU3,XI,3,DUMMY)
C  COMPUTE K12 AND K21 THEN WRITE FINAL DISK FILE
      WRITE(IU5,1200) ZI(J)
      PSI(1)=PSIZ+H1(1)*DX(1)
      DO 11 I=1,NXTL
      IF(I.LT.NXTL) PSI(I+1)=PSI(I)+.5*(H1(I)+H1(I+1))*(XI(I+1)-XI(I))
      FACT1=CK1(I)+DTHDX(I)/H1(I)
      FACT2=CK2(I)+DTHDZ(I)/H2(I)
      SINTH=SIN(THETA(I,J))
      COSTH=COS(THETA(I,J))
      CK12(I)=(-FACT1+COSTH*FACT2)/SINTH
      CK21(I)=(-FACT2+COSTH*FACT1)/SINTH
      WRITE(IU4) H1(I),H2(I),CK1(I),CK2(I),CK12(I),
     *                CK21(I),THETA(I,J),PSI(I)
      WRITE(IU5,100) I,XI(I),YB(I),ZB(I),H1(I),H2(I),THETA(I,J),
     *                     CK1(I),CK2(I),CK12(I),CK21(I),PSI(I)
11    CONTINUE
      REWIND IU2
12    CONTINUE
100   FORMAT(I4,5X,1P11E11.3)
200   FORMAT(1X,'All Section Data Smoothed & Differentiated'//)
300   FORMAT(1X,'Geometrical Parameters Computed for J = ',I3)
400   FORMAT(/1X,'d(Theta)/dz Computed for all Sections'/)
1000  FORMAT(3F10.5,11X,2F10.5)
1100  FORMAT(10X,2F10.5,11X,2F10.5)
1200  FORMAT(//1X,'Z = ',F7.3/3X1HI,10X2HXB,9X2HYB,9X2HZB,
     *                  9X2HH1,9X2HH2,8X5HTHETA,7X2HK1,9X2HK2,
     *                  9X3HK12,8X3HK21,8X3HPSI)
      END
```

```
      SUBROUTINE ANGLE(COSTH,SINTH,THETA)
C ******************************************************************
C THIS SUBROUTINE CALCULATES AN ANGLE IN EITHER THE FIRST OR SECOND
C QUADRANT GIVEN THE COSINE OF THE ANGLE
C ******************************************************************
      IF(COSTH.EQ.0.) GO TO 2
      IF(COSTH.GT.0.) GO TO 1
      SINPHI=-COSTH
      COSPHI=SQRT(1.-SINPHI**2)
      TANPHI=SINPHI/COSPHI
      PHI=ATAN(TANPHI)
      THETA=355./226.+PHI
      SINTH=COSPHI
      RETURN
    1 SINTH=SQRT(1.-COSTH**2)
      TANTH=SINTH/COSTH
      THETA=ATAN(TANTH)
      RETURN
    2 SINTH=1.
      THETA=355./226.
      RETURN
      END
      SUBROUTINE ARC(DYDX1)
C ******************************************************************
C THIS SUBROUTINE COMPUTES LOCAL ARCLENGTH AND RESCALES TO UNITY
C ******************************************************************
      COMMON/COORD/ Z(40)
      COMMON/COORDB/XB(40),YB(40),ZB(40)
      COMMON/NDATA/ IU1,IU2,IU3,IU4,IU5,IMAX,JMAX,NXTL,NZTL,JFZ,LZ(30)
      Z(1)=0.
      DO 1 J=2,JMAX
      JM=J-1
      DYB=YB(J)-YB(JM)
      DZB=ZB(J)-ZB(JM)
      DZ=SQRT(DYB*DYB+DZB*DZB)
    1 Z(J)=Z(JM)+DZ
      ZSCALE=Z(JMAX)
      DO 2 J=2,JMAX
    2 Z(J)=Z(J)/ZSCALE
      DYDX1=-ZSCALE
      RETURN
      END
      SUBROUTINE GEOMET(J)
C ******************************************************************
C THIS SUBROUTINE CALCULATES THE METRIC AND CURVATURE PARAMETERS
C ******************************************************************
      COMMON/COORDB/XB(40),YB(40),ZB(40)
      COMMON/GEOMP/ H1(40),H2(40),THETA(40,40),CK1(40),CK2(40),
     *              CK12(40),CK21(40),PSI(40)
      COMMON/TEMPZ/ DYBDZ(40),D2YBDZ(40),DZBDZ(40),D2ZBDZ(40)
      COMMON/TEMPX/ DYBDX(40),D2YBDX(40),DZBDX(40),D2ZBDX(40)
      COMMON/NDATA/ IU1,IU2,IU3,IU4,IU5,IMAX,JMAX,NXTL,NZTL,JFZ,LZ(30)
      DO 3 I=1,NXTL
      H1(I)=SQRT(1.+DZBDX(I)**2+DYBDX(I)**2)
      H2(I)=SQRT(DYBDZ(I)**2+DZBDZ(I)**2)
      COSTH=(DYBDZ(I)*DYBDX(I)+DZBDX(I)*DZBDZ(I))/(H1(I)*H2(I))
      CALL ANGLE(COSTH,SINTH,THETA(I,J))
      FACT1=DYBDZ(I)*DZBDX(I)-DYBDX(I)*DZBDZ(I)
      FACT2=DZBDX(I)*D2YBDX(I)-DYBDX(I)*D2ZBDX(I)
      FACT3=DYBDZ(I)*D2YBDX(I)+DZBDZ(I)*D2ZBDX(I)
```

```
      DENOM=H1(I)**4*H2(I)*SINTH
      CK1(I)=(FACT1*FACT2+FACT3)/DENOM
      FACT2=DZBDZ(I)*D2YBDZ(I)-DYBDZ(I)*D2ZBDZ(I)
      DENOM=H1(I)*H2(I)**4*SINTH
      CK2(I)=-FACT1*FACT2/DENOM
      WRITE(IU3) YB(I),ZB(I),H1(I),H2(I),THETA(I,J),CK1(I),CK2(I)
 3    CONTINUE
      RETURN
      END
      SUBROUTINE GRID
C *******************************************************************
C  THIS SUBROUTINE SETS UP THE COMPUTATIONAL GRID
C *******************************************************************
      COMMON/INTRP/ XI(40),ZI(40)
      COMMON/XDATA/ PSIZ,XSTART,XEND,DX(10),NX(10)
      COMMON/ZDATA/ ZSTART,ZEND,DZ(10),NZ(10)
      COMMON/NDATA/ IU1,IU2,IU3,IU4,IU5,IMAX,JMAX,NXTL,NZTL,JFZ,LZ(30)
      XI(1)=XSTART+DX(1)
      K=1
      MCUT=NX(1)
      NXTLM=NXTL-1
      DO 1 I=1,NXTLM
      IF(I.LE.MCUT) GO TO 1
      K=K+1
      MCUT=MCUT+NX(K)
 1    XI(I+1)=XI(I)+DX(K)
      IF(ABS(XI(NXTL)-XEND).GT..1*DX(K)) GO TO 7
      XI(NXTL)=XEND
      ZI(1)=ZSTART
      K=1
      MCUT=NZ(1)
      NZTLM=NZTL-1
      DO 3 I=1,NZTLM
      IF(I.LE.MCUT) GO TO 3
      K=K+1
      MCUT=MCUT+NZ(K)
 3    ZI(I+1)=ZI(I)+DZ(K)
      IF(ABS(ZI(NZTL)-ZEND).GT..1*DZ(K)) GO TO 8
      ZI(NZTL)=ZEND
      DO 4 I=1,NXTL
 4    WRITE(IU4) XI(I)
      DO 5 I=1,NZTL
 5    WRITE(IU4) ZI(I)
      RETURN
 7    WRITE(IU5,100) XI(NXTL),XEND
      STOP
 8    WRITE(IU5,200) ZI(NZTL),ZEND
      STOP
 100  FORMAT(/1X,'STEPSIZE ARRAY IMPLIES XEND =',1PE13.6,
     *' WHICH DOES NOT MATCH THE DESIRED VALUE...',E13.6)
 200  FORMAT(/1X,'STEPSIZE ARRAY IMPLIES ZEND =',1PE13.6,
     *' WHICH DOES NOT MATCH THE DESIRED VALUE...',E13.6)
      END
      SUBROUTINE INITL(X,Y,N,L,JF,ISTART,DYDX1)
C *******************************************************************
C  THIS SUBROUTINE SETS UP INITIAL VALUES FOR THE VARIOUS ARRAYS
C *******************************************************************
      DIMENSION X(40),Y(40),L(30)
      COMMON/QUAN/ A(30),B(30),PH0PH0(30),PH0PH1(30),
     *             PH0PS0(30),PH0PS1(30),PH1PH1(30),PH1PS0(30),
```

```
     *                 PH1PS1(30),P1(30),P2(30),P3(30),Q1(30),
     *                 Q2(30),Q3(30),XI(30),YPHI0(30),YPHI1(30),
     *                 H(30)
      COMMON/DIMN/ JFM
      COMMON/LDATA/ ISYM,ISET
      JFM=JF-1
      Y1=Y(1)
      YN=Y(N)
      IF(ISTART.NE.0) GO TO 2
      X32=X(3)-X(2)
      X31=X(3)-X(1)
      X21=X(2)-X(1)
      AA=-(X21+X31)/(X21*X31)
      BB=X31/(X21*X32)
      CC=-X21/(X31*X32)
      DYDX1=AA*Y(1)+BB*Y(2)+CC*Y(3)
    2 X32=X(N-2)-X(N-1)
      X31=X(N-2)-X(N)
      X21=X(N-1)-X(N)
      AA=-(X21+X31)/(X21*X31)
      BB=X31/(X21*X32)
      CC=-X21/(X31*X32)
      DYDXN=AA*Y(N)+BB*Y(N-1)+CC*Y(N-2)
      IF(ISET.EQ.0) GO TO 3
      DYDXN=0.
      YN=-(BB*Y(N-1)+CC*Y(N-2))/AA
    3 ISET=0
      P1(1)=Y1
      P2(1)=0.
      P3(1)=0.
      Q1(1)=DYDX1
      Q2(1)=0.
      Q3(1)=0.
      A(JF)=YN
      B(JF)=DYDXN
      DO 1 J=1,JF
      K=L(J)
      XI(J)=X(K)
      PH0PH0(J)=0.
      PH0PH1(J)=0.
      PH0PS0(J)=0.
      PH0PS1(J)=0.
      PH1PH1(J)=0.
      PH1PS0(J)=0.
      PH1PS1(J)=0.
      YPHI0 (J)=0.
      YPHI1 (J)=0.
    1 CONTINUE
      RETURN
      END
      SUBROUTINE INTERP(I,J,XX,IU2,IFLAG)
C   ********************************************************************
C   THIS SUBROUTINE USES THE SPLINE FIT TO INTERPOLATE THE ORIGINAL
C   DATA AND TO COMPUTE ITS FIRST AND SECOND DERIVATIVES
C   ********************************************************************
      COMMON/QUAN/ A(30),B(30),PH0PH0(30),PH0PH1(30),
     *                 PH0PS0(30),PH0PS1(30),PH1PH1(30),PH1PS0(30),
     *                 PH1PS1(30),P1(30),P2(30),P3(30),Q1(30),
     *                 Q2(30),Q3(30),XI(30),YPHI0(30),YPHI1(30),
     *                 H(30)
```

```
      COMMON/TEMPX/ DYBDX(40),D2YBDX(40),DZBDX(40),D2ZBDX(40)
      COMMON/TEMPZ/ DYBDZ(40),D2YBDZ(40),DZBDZ(40),D2ZBDZ(40)
      COMMON/COORDB/XB(40),YB(40),ZB(40)
      COMMON/DTHET/ DTHDX(40),DTHDZ(40),THET(40),LT(30)
      COMMON/DIMN/ JFM
      IF(XX.LE.XI(I+1)) GO TO 1
      I=I+1
 1    IP=I+1
      DX=(XX-XI(I))/H(I)
      DXP=(XI(IP)-XX)/H(I)
      PHI1=(3.-2.*DX)*DX**2
      PHI0=1.-PHI1
      PSI1=-H(I)*DXP*DX**2
      PSI0=H(I)*DX*DXP**2
      PHI1P=6.*DX*(1.-DX)/H(I)
      PHI0P=-PHI1P
      PSI1P=-DX*(2.*DXP-DX)
      PSI0P=DXP*(DXP-2.*DX)
      PH1PP=6.*(1.-2.*DX)/H(I)**2
      PH0PP=-PH1PP
      PS1PP=2.*(2.*DX-DXP)/H(I)
      PS0PP=2.*(DX-2.*DXP)/H(I)
      YY=A(I)*PHI0 +A(IP)*PHI1 +B(I)*PSI0 +B(IP)*PSI1
      YP=A(I)*PHI0P+A(IP)*PHI1P+B(I)*PSI0P+B(IP)*PSI1P
      Y2=A(I)*PH0PP+A(IP)*PH1PP+B(I)*PS0PP+B(IP)*PS1PP
      IF(IFLAG.GT.0) GO TO 2
      WRITE(IU2) YY,YP,Y2
      RETURN
 2    IF(IFLAG.GT.2) GO TO 4
      IF(IFLAG.EQ.2) GO TO 3
      YB(J)=YY
      DYBDX(J)=YP
      D2YBDX(J)=Y2
      RETURN
 3    ZB(J)=YY
      DZBDX(J)=YP
      D2ZBDX(J)=Y2
      RETURN
 4    IF(IFLAG.EQ.3) DTHDX(J)=YP
      IF(IFLAG.EQ.4) DYBDZ(J)=YY
      IF(IFLAG.EQ.5) DZBDZ(J)=YY
      IF(IFLAG.EQ.6) D2YBDZ(J)=YY
      IF(IFLAG.EQ.7) D2ZBDZ(J)=YY
      IF(IFLAG.EQ.8) DTHDZ(J)=YP
      RETURN
      END
      SUBROUTINE NAMIN
C   ******************************************************************
C   THIS SUBROUTINE COORDINATES READING OF INPUT DATA
C   ******************************************************************
      COMMON/NDATA/ N(40)
      COMMON/MDATA/ M(31)
      COMMON/LDATA/ ISYM,ISET
      COMMON/XDATA/ X(13),NX(10)
      COMMON/ZDATA/ Z(12),NZ(10)
      ISYM=0
      CALL NAMLST(6,2,ISYM,DUMMY)
      DO 1  I=1,10
      N(I)=0
 1    CALL NAMLST(6,2,N(I),DUMMY)
```

```
      IUP=10+N(10)
      DO 2 I=11,IUP
      N(I)=0
    2 CALL NAMLST(6,2,N(I),DUMMY)
      M(1)=0
      CALL NAMLST(6,2,M(1),DUMMY)
      IUP=1+M(1)
      DO 3 I=2,IUP
      M(I)=0
    3 CALL NAMLST(6,2,M(I),DUMMY)
      DO 4 I=1,13
    4 CALL NAMLST(6,2,1,X(I))
      DO 5 I=1,10
      NX(I)=0
    5 CALL NAMLST(6,2,NX(I),DUMMY)
      DO 6 I=1,12
    6 CALL NAMLST(6,2,1,Z(I))
      DO 7 I=1,10
      NZ(I)=0
    7 CALL NAMLST(6,2,NZ(I),DUMMY)
      RETURN
      END
      SUBROUTINE NAMLST(II,IO,N,X)
C ******************************************************************
C THIS SUBROUTINE READS INPUT DATA
C ******************************************************************
      DIMENSION A(3)
      IF(N .NE. 0)  GO TO 1
      READ (II,2)  A(1),A(2),A(3),N
      WRITE(IO,2)  A(1),A(2),A(3),N
      RETURN
    1 READ (II,3)  A(1),A(2),A(3),X
      WRITE(IO,3)  A(1),A(2),A(3),X
      RETURN
    2 FORMAT(1X,3A4,I4)
    3 FORMAT(1X,3A4,1PE13.6)
      END
      SUBROUTINE READTH(J)
C ******************************************************************
C THIS SUBROUTINE READS D(THETA)/DZ ON A SECTION FROM DISK
C ******************************************************************
      COMMON/DTHET/ DTHDX(40),DTHDZ(40),THET(40),LT(30)
      COMMON/NDATA/ IU1,IU2,IU3,IU4,IU5,IMAX,JMAX,NXTL,NZTL,JFZ,LZ(30)
      DO 3 I=1,NXTL
      DO 1 JJ=1,J
    1 READ(IU2) YY,D,D
      DTHDZ(I)=YY
      IF(J.EQ.NZTL) GO TO 3
      JP=J+1
      DO 2 JJ=JP,NZTL
    2 READ(IU2) D,D,D
    3 CONTINUE
      RETURN
      END
      SUBROUTINE READYZ(J)
C ******************************************************************
C THIS SUBROUTINE READS SMOOTHED SECTION DATA FROM DISK AND FILLS
C ARRAYS IN PREPARATION FOR SMOOTHING IN THE STREAMWISE DIRECTION
C ******************************************************************
      COMMON/COORDB/XB(40),YB(40),ZB(40)
```

```
      COMMON/TEMPZ/ DYBDZ(40),D2YBDZ(40),DZBDZ(40),D2ZBDZ(40)
      COMMON/NDATA/ IU1,IU2,IU3,IU4,IU5,IMAX,JMAX,NXTL,NZTL,JFZ,LZ(30)
      DO 6 I=1,IMAX
      DO 1 JJ=1,J
    1 READ(IU2) YY,YP,Y2
      YB(I)=YY
      DYBDZ(I)=YP
      D2YBDZ(I)=Y2
      IF(J.EQ.NZTL) GO TO 3
      JP=J+1
      DO 2 JJ=JP,NZTL
    2 READ(IU2) D,D,D
    3 CONTINUE
      DO 4 JJ=1,J
    4 READ(IU2) ZZ,ZP,Z2
      ZB(I)=ZZ
      DZBDZ(I)=ZP
      D2ZBDZ(I)=Z2
      IF(J.EQ.NZTL) GO TO 6
      DO 5 JJ=JP,NZTL
    5 READ(IU2) D,D,D
    6 CONTINUE
      RETURN
      END
      SUBROUTINE SMOOTH(X,Y,L,JF,N,ISTART,NN,IU2,XT,IFLAG,DYDX1)
C *******************************************************************
C THIS SUBROUTINE FITS A TABULATED FUNCTION WITH CUBIC SPLINES USING
C LEAST-SQUARES ERROR MINIMIZATION
C *******************************************************************
      DIMENSION X(40),Y(40),L(30),XT(40)
      COMMON/QUAN/ A(30),B(30),PH0PH0(30),PH0PH1(30),
     *             PH0PS0(30),PH0PS1(30),PH1PH1(30),PH1PS0(30),
     *             PH1PS1(30),P1(30),P2(30),P3(30),Q1(30),
     *             Q2(30),Q3(30),XI(30),YPHI0(30),YPHI1(30),
     *             H(30)
      COMMON/DIMN/ JFM
C INITIATE THE COMPUTATION
      CALL INITL(X,Y,N,L,JF,ISTART,DYDX1)
      CALL SUMUP(X,Y,N,L,JF)
C COMPUTE THE P'S AND Q'S
      DO 1 J=2,JFM
      M=J-1
C COEFFICIENTS IN THE ORIGINAL EQUATIONS
      A1=PH0PH1(M)
      B1=PH1PH1(M)+PH0PH0(J)
      C1=PH0PH1(J)
      D1=PH1PS0(M)
      E1=PH1PS1(M)+PH0PS0(J)
      F1=PH0PS1(J)
      G1=YPHI1 (M)+YPHI0 (J)
      A2=3./H(M)**2
      B2=3.*(1./H(J)**2-1./H(M)**2)
      C2=-3./H(J)**2
      D2=1./H(M)
      E2=2.*(1./H(J)+1./H(M))
      F2=1./H(J)
      G2=0.
C MODIFIED COEFFICIENTS
      B1S=B1+A1*P2(M)+D1*Q2(M)
      E1S=E1+A1*P3(M)+D1*Q3(M)
```

```
      G1S=G1-A1*P1(M)-D1*Q1(M)
      B2S=B2+A2*P2(M)+D2*Q2(M)
      E2S=E2+A2*P3(M)+D2*Q3(M)
      G2S=G2-A2*P1(M)-D2*Q1(M)
C  THE P'S AND Q'S
      DEL=1./(B1S*E2S-B2S*E1S)
      P1(J)=(E2S*G1S-E1S*G2S)*DEL
      P2(J)=(E1S*C2 -E2S*C1 )*DEL
      P3(J)=(E1S*F2 -E2S*F1 )*DEL
      Q1(J)=(B1S*G2S-B2S*G1S)*DEL
      Q2(J)=(B2S*C1 -B1S*C2 )*DEL
      Q3(J)=(B2S*F1 -B1S*F2 )*DEL
 1    CONTINUE
C  SOLVE FOR THE A'S AND B'S
      K=JFM
      KP=JF
      DO 2 J=1,JFM
      A(K)=P1(K)+P2(K)*A(KP)+P3(K)*B(KP)
      B(K)=Q1(K)+Q2(K)*A(KP)+Q3(K)*B(KP)
      KP=K
      K=K-1
 2    CONTINUE
C  INTERPOLATE TO DESIRED MESH
      I=1
      DO 3 J=1,NN
      CALL INTERP(I,J,XT(J),IU2,IFLAG)
 3    CONTINUE
      RETURN
      END
      SUBROUTINE SUMUP(X,Y,N,L,JF)
C  *****************************************************************
C  THIS SUBROUTINE PERFORMS SUMMING OPERATIONS ON THE CUBIC-SPLINE
C  WEIGHTING FUNCTIONS
C  *****************************************************************
      DIMENSION X(40),Y(40),L(30)
      COMMON/QUAN/ A(30),B(30),PH0PH0(30),PH0PH1(30),
     *             PH0PS0(30),PH0PS1(30),PH1PH1(30),PH1PS0(30),
     *             PH1PS1(30),P1(30),P2(30),P3(30),Q1(30),
     *             Q2(30),Q3(30),XI(30),YPHI0(30),YPHI1(30),
     *             H(30)
      COMMON/DIMN/ JFM
      DO 2 J=1,JFM
      H(J)=XI(J+1)-XI(J)
      KL=L(J)
      KU=L(J+1)
      DO 1 K=KL,KU
      DX=(X(K)-XI(J))/H(J)
      DXP=(XI(J+1)-X(K))/H(J)
      YY=Y(K)
      PHI1=(3.-2.*DX)*DX**2
      PHI0=1.-PHI1
      PSI1=-H(J)*DXP*DX**2
      PSI0= H(J)*DX*DXP**2
      PH0PH0(J)=PH0PH0(J)+PHI0*PHI0
      PH0PH1(J)=PH0PH1(J)+PHI0*PHI1
      PH0PS0(J)=PH0PS0(J)+PHI0*PSI0
      PH0PS1(J)=PH0PS1(J)+PHI0*PSI1
      PH1PH1(J)=PH1PH1(J)+PHI1*PHI1
      PH1PS0(J)=PH1PS0(J)+PHI1*PSI0
      PH1PS1(J)=PH1PS1(J)+PHI1*PSI1
```

```
      YPHI0 (J)=YPHI0 (J)+PHI0*YY
      YPHI1 (J)=YPHI1 (J)+PHI1*YY
1     CONTINUE
2     CONTINUE
      RETURN
      END
```

```
      PROGRAM VELOC
C ****************************************************************
C THIS PROGRAM SMOOTHS AND INTERPOLATES BOUNDARY LAYER EDGE VELOCITY
C COMPONENTS ONTO A RECTANGULAR GRID IN TRANSFORMED COORDINATES AND
C WRITES A DISK FILE FOR USE BY PROGRAM EDDY3
C ****************************************************************
      COMMON/COORD/ Z(40)
      COMMON/COORDB/XB(40),YB(40),ZB(40)
      COMMON/DERIV/ DYBDZ(40),DZBDZ(40),DWDZ1(40),DWDZN(40)
      COMMON/INTRP/ XI(40),ZI(40)
      COMMON/VEL/   UE(40),VE(40),WE(40)
      COMMON/VELI/  UEI(40,40),WEI(40,40)
      COMMON/NDATA/ IU1,IU2,IU3,IU4,IU5,IMAX,JMAX,NXTL,NZTL,JFZ,LZ(30)
      COMMON/MDATA/ JFX,LX(30)
      COMMON/LDATA/ ISYM,ISET
      CALL OPEN(6,'NAMVEL ',1)
      CALL NAMIN
      CALL OPEN(IU1,'UEWE ',1)
      CALL OPEN(IU3,'VELOCITY ',8)
      CALL OPEN(IU4,'GEOMETRY ',32)
      DO 1 I=1,NXTL
 1    READ(IU4) XI(I)
      DO 2 I=1,NZTL
 2    READ(IU4) ZI(I)
      JMAX=JMAX+2
      JMAXM=JMAX-1
      DO 4 I=1,IMAX
      ISET=0
C READ SECTION VELOCITIES
      READ(IU1,1000) XB(I),YB(2),ZB(2),CX,CY,CZ,Q
      UE(2)=Q*CX
      VE(2)=Q*CY
      WE(2)=Q*CZ
      DO 3 J=3,JMAXM
      READ(IU1,1100) YB(J),ZB(J),CX,CY,CZ,Q
      UE(J)=Q*CX
      VE(J)=Q*CY
 3    WE(J)=Q*CZ
C COMPUTE VELOCITIES ON KEEL AND FREE SURFACE
      CALL ENDS
C COMPUTE AND RESCALE LOCAL ARCLENGTH
      CALL ARC(DYDX1)
C COMPUTE TANGENTIAL VELOCITY ON SECTION
      IF(ISYM.EQ.1) ISET=1
      CALL SMOOTH(Z,YB,LZ,JFZ,JMAX,1,JMAX,Z,6,DYDX1,IDUM)
      CALL SMOOTH(Z,ZB,LZ,JFZ,JMAX,1,JMAX,Z,7,0.,IDUM)
      DO 10 J=1,JMAX
      H2=SQRT(DYBDZ(J)**2+DZBDZ(J)**2)
      Q2=UE(J)**2+VE(J)**2+WE(J)**2
      WE(J)=(VE(J)*DYBDZ(J)+WE(J)*DZBDZ(J))/H2
 10   UE(J)=SQRT(Q2-WE(J)**2)
C SMOOTH AND INTERPOLATE ORTHOGONAL VELOCITIES
      CALL SMOOTH(Z,UE,LZ,JFZ,JMAX,1,NZTL,ZI,0,0.,I)
      CALL SMOOTH(Z,WE,LZ,JFZ,JMAX,0,NZTL,ZI,1,DUMMY,I)
      WRITE(1,150) I
 4    CONTINUE
      WRITE(1,200)
C SMOOTH AND INTERPOLATE DATA ALONG Z=CONSTANT
      DO 6 J=1,NZTL
      DO 5 I=1,IMAX
```

```
      UE(I)=UEI(I,J)
    5 WE(I)=WEI(I,J)
      CALL SMOOTH(XB,UE,LX,JFX,IMAX,0,NXTL,XI,2,DUMMY,IDUM)
      CALL SMOOTH(XB,WE,LX,JFX,IMAX,0,NXTL,XI,3,DUMMY,IDUM)
C  COMPUTE NONORTHOGONAL VELOCITY COMPONENTS
      CALL ROTATE(J)
    6 CONTINUE
      CALL SMOOTH(XB,DWDZ1,LX,JFX,IMAX,0,NXTL,XI,4,DUMMY,IDUM)
      CALL SMOOTH(XB,DWDZN,LX,JFX,IMAX,0,NXTL,XI,5,DUMMY,IDUM)
C  WRITE DISK FILE FOR SMOOTHED DATA ALONG Z=CONSTANT
      DO 7 J=1,NZTL
      WRITE(IU5,1200) ZI(J)
      DO 8 I=1,NXTL
      WRITE(IU3) UEI(I,J),WEI(I,J)
    8 WRITE(IU5,100) I,XI(I),UEI(I,J),WEI(I,J)
    7 CONTINUE
      WRITE(IU5,250)
      DO 20 I=1,NXTL
      WRITE(IU3) DWDZ1(I),DWDZN(I)
   20 WRITE(IU5,100) I,XI(I),DWDZ1(I),DWDZN(I)
  100 FORMAT(I4,5X,1P3E11.3)
  150 FORMAT(1X,'Section Data Smoothed & Differentiated for I =',I3)
  200 FORMAT(1X,'All Section Data Smoothed & Differentiated'//)
  250 FORMAT(//3X1HI,10X2HXB,8X5HDWDZ1,7X5HDWDZN)
 1000 FORMAT(3F10.5,4F10.6)
 1100 FORMAT(10X,2F10.5,4F10.6)
 1200 FORMAT(//1X,'Z = ',F7.3/3X1HI,10X2HXB,9X2HUE,9X2HWE)
      END
      SUBROUTINE ARC(DYDX1)
C  ******************************************************************
C  THIS SUBROUTINE COMPUTES LOCAL ARCLENGTH AND RESCALES TO UNITY
C  ******************************************************************
      COMMON/COORD/ Z(40)
      COMMON/COORDB/XB(40),YB(40),ZB(40)
      COMMON/NDATA/ IU1,IU2,IU3,IU4,IU5,IMAX,JMAX,NXTL,NZTL,JFZ,LZ(30)
      Z(1)=0.
      DO 1 J=2,JMAX
      JM=J-1
      DYB=YB(J)-YB(JM)
      DZB=ZB(J)-ZB(JM)
      DZ=SQRT(DYB*DYB+DZB*DZB)
    1 Z(J)=Z(JM)+DZ
      ZSCALE=Z(JMAX)
      DO 2 J=2,JMAX
    2 Z(J)=Z(J)/ZSCALE
      DYDX1=-ZSCALE
      RETURN
      END
      SUBROUTINE ENDS
C  ******************************************************************
C  THIS SUBROUTINE COMPUTES VELOCITIES ON THE KEEL AND FREE SURFACE
C  ******************************************************************
      COMMON/COORDB/XB(40),YB(40),ZB(40)
      COMMON/VEL/   UE(40),VE(40),WE(40)
      COMMON/NDATA/ IU1,IU2,IU3,IU4,IU5,IMAX,JMAX,NXTL,NZTL,JFZ,LZ(30)
      Y2SQ=YB(2)**2
      Y3SQ=YB(3)**2
      D=1./(Y3SQ-Y2SQ)
      UE(1)=(Y3SQ*UE(2)-Y2SQ*UE(3))*D
      VE(1)=0.
```

```
      WE(1)=(Y3SQ*WE(2)-Y2SQ*WE(3))*D
      YB(1)=0.
      ZB(1)=(Y3SQ*ZB(2)-Y2SQ*ZB(3))*D
      JM1=JMAX-1
      JM2=JMAX-2
      ZNM1=ZB(JM1)
      ZNM2=ZB(JM2)
      D=1./(ZNM1-ZNM2)
      UE(JMAX)=(ZNM1*UE(JM2)-ZNM2*UE(JM1))*D
      VE(JMAX)=(ZNM1*VE(JM2)-ZNM2*VE(JM1))*D
      WE(JMAX)=0.
      YB(JMAX)=(ZNM1*YB(JM2)-ZNM2*YB(JM1))*D
      ZB(JMAX)=0.
      RETURN
      END
      SUBROUTINE INITL(X,Y,N,L,JF,ISTART,DYDX1)
C ******************************************************************
C THIS SUBROUTINE SETS UP INITIAL VALUES FOR THE VARIOUS ARRAYS
C ******************************************************************
      DIMENSION X(40),Y(40),L(30)
      COMMON/QUAN/ A(30),B(30),PH0PH0(30),PH0PH1(30),
     *             PH0PS0(30),PH0PS1(30),PH1PH1(30),PH1PS0(30),
     *             PH1PS1(30),P1(30),P2(30),P3(30),Q1(30),
     *             Q2(30),Q3(30),XI(30),YPHI0(30),YPHI1(30),
     *             H(30)
      COMMON/DIMN/ JFM,NFS
      COMMON/LDATA/ ISYM,ISET
      JFM=JF-1
      Y1=Y(1)
      YN=Y(N)
      IF(ISTART.NE.0) GO TO 2
      X32=X(3)-X(2)
      X31=X(3)-X(1)
      X21=X(2)-X(1)
      AA=-(X21+X31)/(X21*X31)
      BB=X31/(X21*X32)
      CC=-X21/(X31*X32)
      DYDX1=AA*Y(1)+BB*Y(2)+CC*Y(3)
    2 X32=X(N-2)-X(N-1)
      X31=X(N-2)-X(N)
      X21=X(N-1)-X(N)
      AA=-(X21+X31)/(X21*X31)
      BB=X31/(X21*X32)
      CC=-X21/(X31*X32)
      DYDXN=AA*Y(N)+BB*Y(N-1)+CC*Y(N-2)
      IF(ISET.EQ.0) GO TO 3
      DYDXN=0.
      YN=-(BB*Y(N-1)+CC*Y(N-2))/AA
    3 ISET=0
      P1(1)=Y1
      P2(1)=0.
      P3(1)=0.
      Q1(1)=DYDX1
      Q2(1)=0.
      Q3(1)=0.
      A(JF)=YN
      B(JF)=DYDXN
      DO 1 J=1,JF
      K=L(J)
      XI(J)=X(K)
```

```
      PH0PH0(J)=0.
      PH0PH1(J)=0.
      PH0PS0(J)=0.
      PH0PS1(J)=0.
      PH1PH1(J)=0.
      PH1PS0(J)=0.
      PH1PS1(J)=0.
      YPHI0 (J)=0.
      YPHI1 (J)=0.
    1 CONTINUE
      RETURN
      END
      SUBROUTINE INTERP(I,J,XX,IFLAG,II)
C *******************************************************************
C THIS SUBROUTINE USES THE SPLINE FIT TO INTERPOLATE THE ORIGINAL
C DATA AND TO COMPUTE ITS FIRST DERIVATIVE
C *******************************************************************
      COMMON/QUAN/ A(30),B(30),PH0PH0(30),PH0PH1(30),
     *             PH0PS0(30),PH0PS1(30),PH1PH1(30),PH1PS0(30),
     *             PH1PS1(30),P1(30),P2(30),P3(30),Q1(30),
     *             Q2(30),Q3(30),XI(30),YPHI0(30),YPHI1(30),
     *             H(30)
      COMMON/DERIV/ DYBDZ(40),DZBDZ(40),DWDZ1(40),DWDZN(40)
      COMMON/VEL/   UE(40),VE(40),WE(40)
      COMMON/VELI/  UEI(40,40),WEI(40,40)
      COMMON/DIMN/ JFM,NFS
      IF(XX.LE.XI(I+1)) GO TO 1
      I=I+1
    1 IP=I+1
      DX=(XX-XI(I))/H(I)
      DXP=(XI(IP)-XX)/H(I)
      IF(IFLAG.GT.5) GO TO 2
      PHI1=(3.-2.*DX)*DX**2
      PHI0=1.-PHI1
      PSI1=-H(I)*DXP*DX**2
      PSI0=H(I)*DX*DXP**2
      YY=A(I)*PHI0+A(IP)*PHI1+B(I)*PSI0+B(IP)*PSI1
      IF(IFLAG.EQ.0) UEI(II,J)=YY
      IF(IFLAG.NE.1) GO TO 3
      WEI(II,J)=YY
      IF(J.EQ.1.OR.J.EQ.NFS) GO TO 2
    3 IF(IFLAG.EQ.2) UE(J)=YY
      IF(IFLAG.EQ.3) WE(J)=YY
      IF(IFLAG.EQ.4) DWDZ1(J)=YY
      IF(IFLAG.EQ.5) DWDZN(J)=YY
      RETURN
    2 PHI1P=6.*DX*(1.-DX)/H(I)
      PHI0P=-PHI1P
      PSI1P=-DX*(2.*DXP-DX)
      PSI0P=DXP*(DXP-2.*DX)
      YP=A(I)*PHI0P+A(IP)*PHI1P+B(I)*PSI0P+B(IP)*PSI1P
      IF(IFLAG.EQ.1.AND.J.EQ.1) DWDZ1(II)=YP
      IF(IFLAG.EQ.1.AND.J.EQ.NFS) DWDZN(II)=YP
      IF(IFLAG.EQ.6) DYBDZ(J)=YP
      IF(IFLAG.EQ.7) DZBDZ(J)=YP
      RETURN
      END
      SUBROUTINE NAMIN
C *******************************************************************
C THIS SUBROUTINE COORDINATES READING OF INPUT DATA
```

```
      C ************************************************************************
            COMMON/NDATA/ N(40)
            COMMON/MDATA/ M(31)
            COMMON/LDATA/  ISYM,ISET
            ISYM=0
            CALL  NAMLST(6,2,ISYM,DUMMY)
            DO 1   I=1,10
            N(I)  =  0
       1    CALL  NAMLST(6,2,N(I),DUMMY)
            IUP=10+N(10)
            DO 2 I=11,IUP
            N(I)=0
       2    CALL  NAMLST(6,2,N(I),DUMMY)
            M(1)=0
            CALL  NAMLST(6,2,M(1),DUMMY)
            IUP=1+M(1)
            DO 3 I=2,IUP
            M(I)=0
       3    CALL  NAMLST(6,2,M(I),DUMMY)
            RETURN
            END
            SUBROUTINE  NAMLST(II,IO,N,X)
      C ************************************************************************
      C THIS SUBROUTINE READS INPUT DATA
      C ************************************************************************
            DIMENSION A(3)
            IF(N .NE. 0)   GO TO 1
            READ (II,2)   A(1),A(2),A(3),N
            WRITE(IO,2)   A(1),A(2),A(3),N
            RETURN
       1    READ (II,3)   A(1),A(2),A(3),X
            WRITE(IO,3)   A(1),A(2),A(3),X
            RETURN
       2    FORMAT(1X,3A4,I4)
       3    FORMAT(1X,3A4,1PE13.6)
            END
            SUBROUTINE  ROTATE(J)
      C ************************************************************************
      C THIS SUBROUTINE TRANSFORMS VELOCITY COMPONENTS TO THE NONORTHOGONAL
      C COORDINATE SYSTEM
      C ************************************************************************
            COMMON/VEL/    UE(40),VE(40),WE(40)
            COMMON/VELI/   UEI(40,40),WEI(40,40)
            COMMON/NDATA/ IU1,IU2,IU3,IU4,IU5,IMAX,JMAX,NXTL,NZTL,JFZ,LZ(30)
            DO 2 I=1,NXTL
            READ(IU4) H1,A,A,A,A,A,THETA,A
            CSCTH=1./SIN(THETA)
            COTTH=CSCTH*COS(THETA)
            UEI(I,J)=UE(I)-WE(I)*COTTH
            WEI(I,J)=WE(I)*CSCTH
            IF(J.EQ.1.OR.J.EQ.NZTL) WEI(I,J)=0.
       2    CONTINUE
            RETURN
            END
            SUBROUTINE  SMOOTH(X,Y,L,JF,N,ISTART,NN,XT,IFLAG,DYDX1,II)
      C ************************************************************************
      C THIS SUBROUTINE FITS A TABULATED FUNCTION WITH CUBIC SPLINES USING
      C LEAST-SQUARES ERROR MINIMIZATION
      C ************************************************************************
            DIMENSION  X(40),Y(40),L(30),XT(40)
```

```
      COMMON/QUAN/ A(30),B(30),PH0PH0(30),PH0PH1(30),
     *             PH0PS0(30),PH0PS1(30),PH1PH1(30),PH1PS0(30),
     *             PH1PS1(30),P1(30),P2(30),P3(30),Q1(30),
     *             Q2(30),Q3(30),XI(30),YPHI0(30),YPHI1(30),
     *             H(30)
      COMMON/DIMN/ JFM,NFS
C  INITIATE THE COMPUTATION
      CALL INITL(X,Y,N,L,JF,ISTART,DYDX1)
      CALL SUMUP(X,Y,N,L,JF)
C  COMPUTE THE P'S AND Q'S
      DO 1 J=2,JFM
      M=J-1
C  COEFFICIENTS IN THE ORIGINAL EQUATIONS
      A1=PH0PH1(M)
      B1=PH1PH1(M)+PH0PH0(J)
      C1=PH0PH1(J)
      D1=PH1PS0(M)
      E1=PH1PS1(M)+PH0PS0(J)
      F1=PH0PS1(J)
      G1=YPHI1  (M)+YPHI0  (J)
      A2=3./H(M)**2
      B2=3.*(1./H(J)**2-1./H(M)**2)
      C2=-3./H(J)**2
      D2=1./H(M)
      E2=2.*(1./H(J)+1./H(M))
      F2=1./H(J)
      G2=0.
C  MODIFIED COEFFICIENTS
      B1S=B1+A1*P2(M)+D1*Q2(M)
      E1S=E1+A1*P3(M)+D1*Q3(M)
      G1S=G1-A1*P1(M)-D1*Q1(M)
      B2S=B2+A2*P2(M)+D2*Q2(M)
      E2S=E2+A2*P3(M)+D2*Q3(M)
      G2S=G2-A2*P1(M)-D2*Q1(M)
C  THE P'S AND Q'S
      DEL=1./(B1S*E2S-B2S*E1S)
      P1(J)=(E2S*G1S-E1S*G2S)*DEL
      P2(J)=(E1S*C2 -E2S*C1 )*DEL
      P3(J)=(E1S*F2 -E2S*F1 )*DEL
      Q1(J)=(B1S*G2S-B2S*G1S)*DEL
      Q2(J)=(B2S*C1 -B1S*C2 )*DEL
      Q3(J)=(B2S*F1 -B1S*F2 )*DEL
    1 CONTINUE
C  SOLVE FOR THE A'S AND B'S
      K=JFM
      KP=JF
      DO 2 J=1,JFM
      A(K)=P1(K)+P2(K)*A(KP)+P3(K)*B(KP)
      B(K)=Q1(K)+Q2(K)*A(KP)+Q3(K)*B(KP)
      KP=K
      K=K-1
    2 CONTINUE
C  INTERPOLATE TO DESIRED MESH
      I=1
      NFS=NN
      DO 3 J=1,NN
      CALL INTERP(I,J,XT(J),IFLAG,II)
    3 CONTINUE
      RETURN
      END
```

```
      SUBROUTINE SUMUP(X,Y,N,L,JF)
C *******************************************************************
C THIS SUBROUTINE PERFORMS SUMMING OPERATIONS ON THE CUBIC-SPLINE
C WEIGHTING FUNCTIONS
C *******************************************************************
      DIMENSION X(40),Y(40),L(30)
      COMMON/QUAN/ A(30),B(30),PH0PH0(30),PH0PH1(30),
     *             PH0PS0(30),PH0PS1(30),PH1PH1(30),PH1PS0(30),
     *             PH1PS1(30),P1(30),P2(30),P3(30),Q1(30),
     *             Q2(30),Q3(30),XI(30),YPHI0(30),YPHI1(30),
     *             H(30)
      COMMON/DIMN/ JFM,NFS
      DO 2 J=1,JFM
      H(J)=XI(J+1)-XI(J)
      KL=L(J)
      KU=L(J+1)
      DO 1 K=KL,KU
      DX=(X(K)-XI(J))/H(J)
      DXP=(XI(J+1)-X(K))/H(J)
      YY=Y(K)
      PHI1=(3.-2.*DX)*DX**2
      PHI0=1.-PHI1
      PSI1=-H(J)*DXP*DX**2
      PSI0= H(J)*DX*DXP**2
      PH0PH0(J)=PH0PH0(J)+PHI0*PHI0
      PH0PH1(J)=PH0PH1(J)+PHI0*PHI1
      PH0PS0(J)=PH0PS0(J)+PHI0*PSI0
      PH0PS1(J)=PH0PS1(J)+PHI0*PSI1
      PH1PH1(J)=PH1PH1(J)+PHI1*PHI1
      PH1PS0(J)=PH1PS0(J)+PHI1*PSI0
      PH1PS1(J)=PH1PS1(J)+PHI1*PSI1
      YPHI0 (J)=YPHI0 (J)+PHI0*YY
      YPHI1 (J)=YPHI1 (J)+PHI1*YY
    1 CONTINUE
    2 CONTINUE
      RETURN
      END
```

```
C      PROGRAM EDDY3
C   ******************************************************************
C   THREE-DIMENSIONAL BOUNDARY LAYER PROGRAM FOR FLOW ABOUT SHIP HULLS
C        USING THE WILCOX-RUBESIN TWO-EQUATION MODEL OF TURBULENCE
C   ******************************************************************
       INTEGER     TAPEIN,TAPEOT,TAPEGP,TAPEPF,TAPEDT,TAPEVL
C      DIMENSION   INDEX1(1189),INDEX4(1189)
       COMMON/BLC0/ NX,NZ,NP,NXT,NZT,NTR,NPT,NXTL,NZTL,NXSTRT,
      *             NZSTRT,KC,IT,IFLOW,ICHORD,ISPAN,INTDIR
       COMMON/BLC3/ X(81),Z(41),DETA(101),ETA(101),ETAE,VGP,CEL,BEL1,BEL2
       COMMON/BLC5/ F(101,2,2),U(101,2,2),V(101),G(101,2,2),W(101,2,2),
      *             T(101),TPROF(620),TPCF(10)
       COMMON/BLC9/ NSEP(41),ICASE(41)
       COMMON/TURB/ E(101,2,2),WT(101),WT2(101,2,2),ELT(101)
       COMMON/STOR/ CON,EPS,ICONVE,ITMAX,WTEDG
       COMMON/TAPE/ TAPEIN,TAPEOT,TAPEGP,TAPEPF,TAPEDT,TAPEVL
       COMMON/IPRT/ IPZ,IPX,IPRINT,EPSV,EPST
       COMMON/CONV/ ICOUNE,ICOUNW,INEGE,INEGW,DELV,DELT
       COMMON/SVNM/ IPCF,IPRF
       DATA IPCF,IPRF/10,620/
C      CALL OPENMS(16,INDEX1,1189,0)
C      CALL OPENMS(17,INDEX4,1189,0)
C***   DISK FILES TAPEGP(16) AND TAPEPF(17) ARE RANDOM ACCESS FILES
C         TAPEGP...BODY GEOMETRY AND VELOCITY DISTRIBUTION
C         TAPEPF...PROFILES AND PRESSURES
C***   DISK FILE TAPEDT(18) IS CREATED IN PROGRAM SHPMSH
C***   DISK FILE TAPEVL(19) IS CREATED IN PROGRAM VELOC
C***   FILES TAPEIN(5) AND TAPEOT(6) ARE CONVENTIONAL READ AND WRITE
C - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
       CALL INPUT
       CALL GRID
       DO 10 I=1,NZTL
 10    NSEP(I)=0
       DO 20 K=1,2
       DO 20 J=1,NPT
       F(J,K,1)=0.
       U(J,K,1)=0.
       G(J,K,1)=0.
       W(J,K,1)=0.
       E(J,K,1)=0.
 20    WT2(J,K,1)=0.
       INTDIR=IFLOW
       IF(IFLOW.EQ.1) WRITE(TAPEOT,690)
       IF(IFLOW.EQ.2) WRITE(TAPEOT,680)
       NZ1=NZSTRT
       NZ2=NZT
       DO 500 NX=NXSTRT,NXT
       NZ=NZ1
       IF(IFLOW.EQ.2) NZ=NZ2
       CALL LOGIC(NZ1,NZ2)
 30    IF(NX.EQ.NXSTRT) CALL PROFIL(1)
       IF(NSEP(NZ).NE.0) GO TO 405
       IT=0
       ICONVE=0
       IPRINT=0
       IF((NZ.EQ.NZSTRT.OR.NZ.EQ.NZT).AND.MOD(NX,IPX).EQ.0) IPRINT=1
       IF(NX.EQ.NXT.AND.MOD(NZ,IPZ).EQ.0) IPRINT=1
       IF(MOD(NZ,IPZ).EQ.0.AND.MOD(NX,IPX).EQ.0) IPRINT=1
       KC=2
       IF(ICASE(NZ).EQ.4) KC=4
```

```
      IF(NZ.EQ.NZT) KC=ICHORD
      IF(NZ.EQ.NZSTRT) KC=1
      IF(IPRINT.EQ.0) GO TO 290
      WRITE(TAPEOT,620) NX,NZ,X(NX),Z(NZ)
      IF(KC.EQ.1) WRITE(TAPEOT,640)
      IF(KC.EQ.2) WRITE(TAPEOT,650)
      IF(KC.EQ.4) WRITE(TAPEOT,670)
290   IT=IT+1
      IF(IT.LE.ITMAX) GO TO 300
      WRITE(TAPEOT,630) NX,NZ
      GO TO 390
300   CALL EDDY
      CALL CGEN
      CALL SOLV6
      CALL SOLVEW
      IF(V(1).LT.0.) GO TO 380
      IF(ABS(DELV/(V(1)+.5*DELV)).GT.EPSV.OR.
     *          ABS(DELT).GT.EPST) GO TO 290
      IF(ICONVE.EQ.0) GO TO 290
      IF(NP.EQ.NPT) GO TO 390
      IF(U(NP-1,2,2).LT..999) GO TO 370
      GO TO 390
370   CALL PROFIL(3)
      IT=0
      GO TO 290
380   WRITE(TAPEOT,720) NX,NZ
      CALL SHIFT2(NZ1,NZ2)
      GO TO 405
390   CALL OUTPUT(1)
      IF(NP.EQ.NPT) GO TO 400
      CALL PROFIL(2)
400   CALL SHIFT(NZ1,NZ2)
405   IF(IFLOW.EQ.2) GO TO 407
      NZ=NZ+1
      IF(NZ.GT.NZ2) GO TO 410
      GO TO 30
407   NZ=NZ-1
      IF(NZ.LT.NZ1) GO TO 410
      GO TO 30
410   CALL OUTPUT(2)
500   CONTINUE
620   FORMAT(1H0,5H**** ,4HNX =,I3,5X4HNZ =,I3,5X3HX =,F10.3,5X3HZ =,
     *        F10.3)
630   FORMAT(1H0,18X32HITERATIONS EXCEED ITMAX FOR NX =,I4,5X4HNZ =,I4)
640   FORMAT(1H0,30H** SYMMETRY-PLANE EQUATIONS **)
650   FORMAT(1H0,45H** CHORDWISE INFINITE-SWEPT HULL EQUATIONS **)
670   FORMAT(1H0,18H** GENERAL CASE **)
680   FORMAT(//1H0,38X50H*****---- WATERLINE CALCULATIONS STARTED ----*
     ****//)
690   FORMAT(//1H0,38X52H*****----- KEEL-LINE CALCULATIONS STARTED -----
     ******//)
720   FORMAT(1H0,11X36H** BOUNDARY-LAYER SEPARATION AT NX =,I4,5X4HNZ =,
     *I4)
      END
```

```
      SUBROUTINE ARRYMV(LL,I,K)
      COMMON/BLC0/ NX,NZ,NP,NXT,NZT,NTR,NPT,NXTL,NZTL,NXSTRT,
     *             NZSTRT,KC,IT,IFLOW,ICHORD,ISPAN,INTDIR
      COMMON/BLC5/ F(101,2,2),U(101,2,2),V(101),G(101,2,2),W(101,2,2),
     *             T(101),TPROF(620),TPCF(10)
      COMMON/PRSC/ PS1(2,2),H1(2,2),H2(2,2),UE(2,2),
     *             WE(2,2),CK1(2,2),CK2(2,2),CK12(2,2),CK21(2,2),
     *             THETA(2,2),PFRS,UFRS,CNUFRS,UREF1,WNP
      COMMON/PRES/ P1(2,2),P2(2,2),P3(2,2),P4(2,2),P5(2,2),
     *             P6(2,2),P7(2,2),P8(2,2),P9(2,2),P10(2,2),P11(2,2),
     *             P12(2,2),P13(2,2),P14(2,2)
      COMMON/TURB/ E(101,2,2),WT(101),WT2(101,2,2),ELT(101)
      COMMON/UNIV/ J1,J2,J3,J4,J5,N1,N2,N3,N4,N5,N6,N7,N8,N9,
     *             N10,N11,N12,N13,N14,NPT2,NPT3,NPT4,NPT5,NPT6
C - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
      NPT2=NPT*2
      NPT3=NPT*3
      NPT4=NPT*4
      NPT5=NPT*5
      NPT6=NPT*6
      GO TO(10,30,90,50,70,85), LL
C   PUT FLOW ARRAYS IN TPROF IN PREPARATION FOR A DISK WRITE
 10   DO 20 J=1,NPT
      CALL INDEX(J,NPT)
      TPROF(J)=F(J,I,K)
      TPROF(J1)=U(J,I,K)
      TPROF(J2)=G(J,I,K)
      TPROF(J3)=W(J,I,K)
      TPROF(J4)=E(J,I,K)
 20   TPROF(J5)=WT2(J,I,K)
      TPROF(N1)=P1(I,K)
      TPROF(N2)=P2(I,K)
      TPROF(N3)=P3(I,K)
      TPROF(N4)=P4(I,K)
      TPROF(N5)=P5(I,K)
      TPROF(N6)=P6(I,K)
      TPROF(N7)=P7(I,K)
      TPROF(N8)=P8(I,K)
      TPROF(N9)=P9(I,K)
      TPROF(N10)=P10(I,K)
      TPROF(N11)=P11(I,K)
      TPROF(N12)=P12(I,K)
      TPROF(N13)=P13(I,K)
      TPROF(N14)=P14(I,K)
      RETURN
C   FILL FLOW ARRAYS AFTER A DISK READ
 30   DO 40 J=1,NPT
      CALL INDEX(J,NPT)
      F(J,I,K)=TPROF(J)
      U(J,I,K)=TPROF(J1)
      G(J,I,K)=TPROF(J2)
      W(J,I,K)=TPROF(J3)
      E(J,I,K)=TPROF(J4)
 40   WT2(J,I,K)=TPROF(J5)
      P1(I,K)=TPROF(N1)
      P2(I,K)=TPROF(N2)
      P3(I,K)=TPROF(N3)
      P4(I,K)=TPROF(N4)
      P5(I,K)=TPROF(N5)
      P6(I,K)=TPROF(N6)
```

```
      P7(I,K)=TPROF(N7)
      P8(I,K)=TPROF(N8)
      P9(I,K)=TPROF(N9)
      P10(I,K)=TPROF(N10)
      P11(I,K)=TPROF(N11)
      P12(I,K)=TPROF(N12)
      P13(I,K)=TPROF(N13)
      P14(I,K)=TPROF(N14)
      RETURN
C   SHIFT (K,K) FLOW ARRAYS TO (I,K)
 50   DO 60 J=1,NPT
      F(J,I,K)=F(J,K,K)
      U(J,I,K)=U(J,K,K)
      G(J,I,K)=G(J,K,K)
      W(J,I,K)=W(J,K,K)
      E(J,I,K)=E(J,K,K)
 60   WT2(J,I,K)=WT2(J,K,K)
      RETURN
C   SHIFT (I,K) FLOW ARRAYS TO (K,K)
 70   DO 80 J=1,NPT
      F(J,K,K)=F(J,I,K)
      U(J,K,K)=U(J,I,K)
      G(J,K,K)=G(J,I,K)
      W(J,K,K)=W(J,I,K)
      E(J,K,K)=E(J,I,K)
 80   WT2(J,K,K)=WT2(J,I,K)
      RETURN
C   SHIFT (I,I) FLOW ARRAYS TO (I,K)
 85   DO 88 J=1,NPT
      F(J,I,K)=F(J,I,I)
      U(J,I,K)=U(J,I,I)
      G(J,I,K)=G(J,I,I)
      W(J,I,K)=W(J,I,I)
      E(J,I,K)=E(J,I,I)
 88   WT2(J,I,K)=WT2(J,I,I)
      RETURN
C   FILL GEOMETRY ARRAYS AFTER A DISK READ
 90   H1(I,K)=TPCF(1)
      H2(I,K)=TPCF(2)
      CK1(I,K)=TPCF(3)
      CK2(I,K)=TPCF(4)
      CK12(I,K)=TPCF(5)
      CK21(I,K)=TPCF(6)
      THETA(I,K)=TPCF(7)
      PS1(I,K)=TPCF(8)
      UE(I,K)=TPCF(9)
      WE(I,K)=TPCF(10)
      RETURN
      END
```

```
      SUBROUTINE BELCEL
      COMMON/BLC0/ NX,NZ,NP,NXT,NZT,NTR,NPT,NXTL,NZTL,NXSTRT,
     *             NZSTRT,KC,IT,IFLOW,ICHORD,ISPAN,INTDIR
      COMMON/BLC3/ X(81),Z(41),DETA(101),ETA(101),ETAE,VGP,CEL,BEL1,BEL2
      COMMON/BLC5/ F(101,2,2),U(101,2,2),V(101),G(101,2,2),W(101,2,2),
     *             T(101),TPROF(620),TPCF(10)
      COMMON/PRES/ P1(2,2),P2(2,2),P3(2,2),P4(2,2),P5(2,2),
     *             P6(2,2),P7(2,2),P8(2,2),P9(2,2),P10(2,2),P11(2,2),
     *             P12(2,2),P13(2,2),P14(2,2)
C -----------------------------------------------------------------
      CALL PRESUR
      IF(KC.EQ.4) GO TO 20
      BEL1=0.
      BEL2=0.
      IF(NX.EQ.NXSTRT) GO TO 10
      CEL=P10(2,2)/(X(NX)-X(NX-1))
      RETURN
   10 CEL=P10(2,2)/(X(NX+1)-X(NX))
      IF(NTR.EQ.0) CEL=0.
      RETURN
   20 DELX=X(NX)-X(NX-1)
      IF(IFLOW.EQ.2) GO TO 30
      DELZ1=Z(NZ)-Z(NZ-1)
      DELZ2=Z(NZ+1)-Z(NZ)
      IF(NZ.GT.(NZSTRT+1)) GO TO 25
      DO 22 J=1,NPT
      G(J,2,1)=0.
   22 W(J,2,1)=0.
      GO TO 40
   25 IF(ICHORD.EQ.2.OR.NZ.LT.(NZT-1)) GO TO 40
      DO 27 J=1,NPT
      G(J,1,1)=0.
   27 W(J,1,1)=0.
      GO TO 40
   30 DELZ1=Z(NZ)-Z(NZ+1)
      DELZ2=Z(NZ-1)-Z(NZ)
      IF(ICHORD.EQ.2.OR.NZ.LT.(NZT-1)) GO TO 35
      DO 32 J=1,NPT
      G(J,2,1)=0.
   32 W(J,2,1)=0.
      GO TO 40
   35 IF(NZ.GT.(NZSTRT+1)) GO TO 40
      DO 37 J=1,NPT
      G(J,1,1)=0.
   37 W(J,1,1)=0.
   40 CEL=P10(2,2)/DELX
      BEL1=.5*P7(2,2)/DELZ1
      BEL2=.5*P7(2,2)/DELZ2
      RETURN
      END
```

```
      SUBROUTINE CGEN
      COMMON/BLC0/ NX,NZ,NP,NXT,NZT,NTR,NPT,NXTL,NZTL,NXSTRT,
     *             NZSTRT,KC,IT,IFLOW,ICHORD,ISPAN,INTDIR
      COMMON/BLC3/ X(81),Z(41),DETA(101),ETA(101),ETAE,VGP,CEL,BEL1,BEL
      COMMON/BLC5/ F(101,2,2),U(101,2,2),V(101),G(10 ,2,2),W(101,2,2),
     *             T(101),TPROF(620),TPCF(10)
      COMMON/PRSC/ PS1(2,2),H1(2,2),H2(2,2),UE(2,2),
     *             WE(2,2),CK1(2,2),CK2(2,2),CK12(2,2),CK21(2,2),
     *             THETA(2,2),PFRS,UFRS,CNUFRS,UREF1,WNP
      COMMON/BLC7/ PU(101),PW(101),QU(101),QW(101)
      COMMON/PRES/ P1(2,2),P2(2,2),P3(2,2),P4(2,2),P5(2,2),
     *             P6(2,2),P7(2,2),P8(2,2),P9(2,2),P10(2,2),P11(2,2),
     *             P12(2,2),P13(2,2),P14(2,2)
      COMMON/BLCE/ EDV(101)
      COMMON/BLCY/ Y1(101),Y3(101),W1,W2,W3,W4
C - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
      IF(IT.EQ.1) CALL BELCEL
      U(NP,2,2)=1.
      W(NP,2,2)=WE(2,2)/UREF1
      IF(KC.EQ.1) W(NP,2,2)=WNP
      PU(1)=0.
      PW(1)=0.
      QU(1)=0.
      QW(1)=0.
      NPM=NP-1
      BM=1.+.5*(EDV(1)+EDV(2))
      DO 50 J=2,NPM
      BP=1.+.5*(EDV(J)+EDV(J+1))
      UB=U(J,2,2)
      WB=W(J,2,2)
      E2=P1(2,2)*F(J,2,2)+P6(2,2)*G(J,2,2)
      DFB=CEL*(F(J,2,2)-F(J,1,2))
      DGB=BEL1*(G(J,2,2)-G(J,2,1))+BEL2*(G(J,1,1)-G(J,1,2))
      USB=UB*UB
      WSB=WB*WB
      DVB=(E2+DFB+DGB)/(DETA(J)+DETA(J-1))
      X1=-(P2(2,2)*UB+P5(2,2)*WB)
      X2=-(P4(2,2)*UB+P3(2,2)*WB)
      A1= BM*Y3(J)-DVB
      B1=-BP*Y1(J)-BM*Y3(J)+X1-(CEL*UB+BEL1*WB)
      C1= BP*Y1(J)+DVB
      D1=P8(2,2)*WSB-P11(2,2)-(CEL*UB+BEL2*WB)*U(J,1,2)
     *                 -BEL1*WB*U(J,2,1)+BEL2*WB*U(J,1,1)
      B2=-BP*Y1(J)-BM*Y3(J)+X2-(CEL*UB+BEL1*WB)
      D2=P9(2,2)*USB-P12(2,2)-(CEL*UB+BEL2*WB)*W(J,1,2)
     *                 -BEL1*WB*W(J,2,1)+BEL2*WB*W(J,1,1)
      B1S=B1+A1*QU(J-1)
      B2S=B2+A1*QW(J-1)
      D1S=D1-A1*PU(J-1)
      D2S=D2-A1*PW(J-1)
      PU(J)=D1S/B1S
      PW(J)=D2S/B2S
      QU(J)=-C1/B1S
      QW(J)=-C1/B2S
   50 BM=BP
      RETURN
      END
```

```
      SUBROUTINE EDDY
      COMMON/BLC0/ NX,NZ,NP,NXT,NZT,NTR,NPT,NXTL,NZTL,NXSTRT,
     *             NZSTRT,KC,IT,IFLOW,ICHORD,ISPAN,INTDIR
      COMMON/BLC5/ F(101,2,2),U(101,2,2),V(101),G(101,2,2),W(101,2,2),
     *             T(101),TPROF(620),TPCF(10)
      COMMON/PRSC/ PS1(2,2),H1(2,2),H2(2,2),UE(2,2),
     *             WE(2,2),CK1(2,2),CK2(2,2),CK12(2,2),CK21(2,2),
     *             THETA(2,2),PFRS,UFRS,CNUFRS,UREF1,WNP
      COMMON/BLCE/ EDV(101)
      COMMON/TURB/ E(101,2,2),WT(101),WT2(101,2,2),ELT(101)
      COMMON/STOR/ CON,EPS,ICONVE,ITMAX,WTEDG
C - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
      REY=UE(2,2)*PS1(2,2)/CNUFRS
      CEP=REY*(UREF1/UE(2,2))**2
      EDV(1)=0.
      ELT(1)=0.
      DO 54 J=2,NP
      IF(IT.EQ.1) WT(J)=SQRT(WT2(J,2,2))
      RET=CEP*E(J,2,2)/WT(J)
      EDV(J)=RET*(1.-CON*EXP(-RET))
   54 ELT(J)=SQRT(E(J,2,2))/WT(J)
      RETURN
      END
```

```
      SUBROUTINE GRID
      INTEGER TAPEIN,TAPEOT,TAPEGP,TAPEPF,TAPEDT,TAPEVL
      COMMON/BLC0/ NX,NZ,NP,NXT,NZT,NTR,NPT,NXTL,NZTL,NXSTRT,
     *             NZSTRT,KC,IT,IFLOW,ICHORD,ISPAN,INTDIR
      COMMON/BLC3/ X(81),Z(41),DETA(101),ETA(101),ETAE,VGP,CEL,BEL1,BEL
      COMMON/BLCY/ Y1(101),Y3(101),W1,W2,W3,W4
      COMMON/TAPE/ TAPEIN,TAPEOT,TAPEGP,TAPEPF,TAPEDT,TAPEVL
C - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
      ETA(1)=0.
      NP=ALOG((ETAE/DETA(1))*(VGP-1.)+1.)/ALOG(VGP)+1.0001
      IF(NP.LE.NPT) GO TO 30
      WRITE(TAPEOT,50)
      STOP 10
   30 DO 40 J=2,NPT
      DETA(J)=VGP*DETA(J-1)
      ETA(J)=ETA(J-1)+DETA(J-1)
      C1=2./(DETA(J)+DETA(J-1))
      Y1(J)=C1/DETA(J)
   40 Y3(J)=C1/DETA(J-1)
      ETAE=ETA(NP)
      W5=(1.+VGP)*(1.+VGP*(1.+VGP))*VGP**3
      DEN=1./(W5*DETA(1))
      W4=(1.+VGP)*DEN
      W3=(1.+VGP*(1.+VGP))**2*DEN
      W2=VGP*(1.+VGP)*W3
      W1=(1.+VGP+(1.+VGP*(1.+VGP))**2*(VGP*(1.+VGP)-1.))*DEN
      RETURN
   50 FORMAT(1H0,37HNP EXCEEDED NPT -- PROGRAM TERMINATED)
      END
```

```
      SUBROUTINE INDEX(J,NPT)
      COMMON/UNIV/ J1,J2,J3,J4,J5,N1,N2,N3,N4,N5,N6,N7,N8,N9,
     *             N10,N11,N12,N13,N14,NPT2,NPT3,NPT4,NPT5,NPT6
C -----------------------------------------------------------------
      J1=J+NPT
      J2=J+NPT2
      J3=J+NPT3
      J4=J+NPT4
      J5=J+NPT5
      IF(J.LT.NPT) RETURN
      N1=NPT6+1
      N2=N1+1
      N3=N2+1
      N4=N3+1
      N5=N4+1
      N6=N5+1
      N7=N6+1
      N8=N7+1
      N9=N8+1
      N10=N9+1
      N11=N10+1
      N12=N11+1
      N13=N12+1
      N14=N13+1
      RETURN
      END
```

```
      SUBROUTINE INPUT
      DIMENSION GMPKG(8),TITLE(9),APC(10)
      INTEGER TAPEIN,TAPEOT,TAPEGP,TAPEPF,TAPEDT,TAPEVL
      COMMON/BLC0/ NX,NZ,NP,NXT,NZT,NTR,NPT,NXTL,NZTL,NXSTRT,
     *             NZSTRT,KC,IT,IFLOW,ICHORD,ISPAN,INTDIR
      COMMON/BLC3/ X(81),Z(41),DETA(101),ETA(101),ETAE,VGP,CEL,BEL1,BEL
      COMMON/CHRD/ WNP1(81),WNPN(81)
      COMMON/PRSC/ PS1(2,2),H1(2,2),H2(2,2),UE(2,2),
     *             WE(2,2),CK1(2,2),CK2(2,2),CK12(2,2),CK21(2,2),
     *             THETA(2,2),PFRS,UFRS,CNUFRS,UREF1,WNP
      COMMON/TAPE/ TAPEIN,TAPEOT,TAPEGP,TAPEPF,TAPEDT,TAPEVL
      COMMON/IPRT/ IPZ,IPX,IPRINT,EPSV,EPST
      COMMON/STOR/ CON,EPS,ICONVE,ITMAX,WTEDG
      COMMON/CONS/ SALFA,SALFAS,SBETA,SBETAS,SLAMDA,SSIGMA,SSGMAS,USTOP,
     *             ZIOTAE,ZIOTAL,RW2
      COMMON/SVNM/ IPCF,IPRF
      COMMON/SHRT/ ISHORT
      COMMON/RELX/ RFTRB,RFVEL
      COMMON/INTG/ ALAMI(41),CFXI(41),CFZI(41),DELTAI(41),
     *             THETXI(41),THETZI(41)
      EQUIVALENCE (APC(1),GMPKG(1)),(APC(9),UEUF),(APC(10),WEUF)
      DATA   TAPEIN/5/,TAPEOT/6/,TAPEGP/16/,TAPEPF/17/,TAPEDT/18/,
     *       TAPEVL/19/,ISHORT/0/,RFTRB/.8/,RFVEL/1./,NPT/101/
      DATA   SALFA/1.11111/,SALFAS/.3/,SBETA/.15/,SBETAS/.09/,
     *       SLAMDA/.091/,SSIGMA/.5/,SSGMAS/.5/,USTOP/4./,
     *       ZIOTAE/3.75E-5/,ZIOTAL/.09/,RW2/4./
      DATA   EPS,EPSV,EPST/3*.01/,ETAE/8./,ICHORD/1/,IFLOW/1/,
     *       IFPRNT/0/,ISPAN/1/,ITMAX/20/,NTR/1/
      NAMELIST/NAME/    DETA   ,  EPS    ,  EPST   ,  EPSV   ,  ETAE   ,
     *                  ICHORD ,  IFLOW  ,  IFPRNT ,  IPX    ,  IPZ    ,
     *                  ISPAN  ,  ITMAX  ,  NPT    ,  NTR    ,  NXSTRT ,
     *                  NXT    ,  NZSTRT ,  NZT    ,  TAPEIN ,  TAPEOT ,
     *                  TAPEGP ,  TAPEPF ,  TAPEDT ,  TAPEVL ,  VGP    ,
     *                  ISHORT ,  RFTRB  ,  RFVEL
      NAMELIST/DATA/    SALFA  ,  SALFAS ,  SBETA  ,  SBETAS ,  SLAMDA ,
     *                  SSIGMA ,  SSGMAS ,  USTOP  ,  ZIOTAE ,  ZIOTAL ,
     *                  RW2
      NAMELIST/STRT/    ALAMI  ,  CFXI   ,  CFZI   ,  DELTAI ,  THETXI ,
     *                  THETZI
C - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
      READ(TAPEIN,120) TITLE
      READ(TAPEIN,NAME)
      READ(TAPEIN,DATA)
      READ(TAPEIN,STRT)
      READ(TAPEIN,170) UFRS,PFRS,CNUFRS
      READ(TAPEDT) NXTL,NZTL
      READ(TAPEDT) (X(I),I=1,NXTL)
      READ(TAPEDT) (Z(I),I=1,NZTL)
      CON=1.-SLAMDA**2
      IF(IFPRNT.EQ.1) WRITE(TAPEOT,210)
      NX=NXSTRT
      IF(IFLOW.EQ.1) NZ=NZSTRT
      IF(IFLOW.EQ.2) NZ=NZT
      UREF1=UFRS
      IPNTG=0
      DO 70 K=1,NZTL
      DO 60 I=1,NXTL
      IPNTG=IPNTG+1
      READ(TAPEDT) GMPKG
      READ(TAPEVL) UEUF,WEUF
```

```
      PE=PFRS
      UEUF=UEUF*UFRS
      WEUF=WEUF*UFRS
      IF(IFPRNT.EQ.1) WRITE(TAPEOT,220) K,I,GMPKG
      CALL STORIT(TAPEGP,APC,IPCF,IPNTG,0)
      IF(I.NE.NXSTRT) GO TO 60
      IF(K.NE.NZ) GO TO 60
      H1(2,2)=APC(1)
      H2(2,2)=APC(2)
      CK1(2,2)=APC(3)
      CK2(2,2)=APC(4)
      CK12(2,2)=APC(5)
      CK21(2,2)=APC(6)
      THETA(2,2)=APC(7)
      PS1(2,2)=APC(8)
      UE(2,2)=APC(9)
      WE(2,2)=APC(10)
 60   CONTINUE
 70   CONTINUE
      DO 75 I=1,NXTL
 75   READ(TAPEVL) WNP1(I),WNPN(I)
      IF(IFPRNT.EQ.0) GO TO 100
      REWIND TAPEVL
      WRITE(6,230)
      DO 90 K=1,NZTL
      WRITE(TAPEOT,240) Z(K)
      DO 80 I=1,NXTL
      READ(TAPEVL) UEUF,WEUF
      UEUF=UEUF*UFRS
      WEUF=WEUF*UFRS
      WRITE(TAPEOT,250) I,X(I),UEUF,WEUF
 80   CONTINUE
 90   CONTINUE
      WRITE(TAPEOT,252)
      DO 95 I=1,NXTL
 95   WRITE(TAPEOT,250) I,X(I),WNP1(I),WNPN(I)
      WRITE(TAPEOT,255)
 100  WRITE(TAPEOT,180)
      WRITE(TAPEOT,190) TITLE
      WRITE(TAPEOT,200) NXSTRT,NZSTRT,NXT,NZT,NTR,IFLOW,ICHORD,
     *                  IFPRNT,ISPAN,ITMAX,IPZ,IPX,VGP,ETAE,
     *                  DETA(1),UFRS,PFRS,CNUFRS,EPS,EPSV,EPST
      WRITE(TAPEOT,260) SALFA,SALFAS,SBETA,SBETAS,SSIGMA,SSGMAS,
     *                  SLAMDA,RW2,USTOP,ZIOTAE,ZIOTAL
      WRITE(TAPEOT,255)
      RETURN
 120  FORMAT(9A6)
 170  FORMAT(1P3E12.4)
 180  FORMAT(50X30HTHREE-D BOUNDARY-LAYER PROGRAM/1H ,
     *     47X36HFOR TURBULENT FLOW ABOUT A SHIP HULL)
 190  FORMAT(////40X,9A6)
 200  FORMAT(///1H0,32X8HNXSTRT= ,I3,15X8HNZSTRT= ,I3,15X8HNXT    = ,I3
     *          /1H0,32X8HNZT    = ,I3,15X8HNTR    = ,I3,15X8HIFLOW = ,I3
     *          /1H0,32X8HICHORD= ,I3,15X8HIFPRNT= ,I3,15X8HISPAN = ,I3
     *          /1H0,32X8HITMAX = ,I3,15X8HIPZ    = ,I3,15X8HIPX    = ,I3
     *    /1H0,32X7HVGP    =,1PE14.6,5X7HETAE  =,E14.6,5X7HDETA  =,E14.6
     *      /1H0,32X7HUFRS  =,E14.6,5X7HPFRS  =,E14.6,5X7HNUFRS =,E14.6
     *      /1H0,32X7HEPS   =,E14.6,5X7HEPSV  =,E14.6,5X7HEPST  =,E14.6)
 210  FORMAT(1X,25H** INPUT HULL GEOMETRY **//2X2HNZ,1X2HNX,5X2HH1,
     *    10X2HH2,10X2HK1,10X2HK2,10X3HK12,9X3HK21,8X5HTHETA,8X2HS1/)
```

```
220  FORMAT(1H ,2I3,1P8E12.4)
230  FORMAT(1H1,45H** EXTERNAL VELOCITY DISTRIBUTION (FT/SEC) **/
    *                          1H0,5X1HZ,13X1HX,13X2HUE,12X2HWE/)
240  FORMAT(1H ,1PE12.5)
250  FORMAT(1H ,7X,I3,2X,1P3E14.5)
252  FORMAT(//1H0,19X1HX,12X4HWNP1,11X4HWNPN/)
255  FORMAT(1H1)
260  FORMAT(/////45X48H**WILCOX-RUBESIN TWO-EQUATION TURBULENCE MODEL*
    *    ///45X29HTHE CLOSURE COEFFICIENTS ARE://47X8HALPHA   =,1PE12.3
    *      5X8HALPHA* =,E12.3//47X8HBETA    =,E12.3,5X8HBETA*  =,E12.3/
    *     47X8HSIGMA  =,E12.3,5X8HSIGMA* =,E12.3//47X8HLAMBDA =,E12.3,
    *      5X8HRW2    =,E12.3//
    *     /36X54HDISSIPATION RATE PRESCRIBED ANALYTICALLY UP TO UPLUS =
    *      E12.4///45X24HINITIAL EDGE CONDITIONS://
    *      47X8HIOTAE  =,E12.4,5X8HIOTAL  =,E12.4)
     END
```

```
      SUBROUTINE LOGIC(NZ1,NZ2)
      INTEGER TAPEIN,TAPEOT,TAPEGP,TAPEPF,TAPEDT,TAPEVL
      COMMON/BLC0/ NX,NZ,NP,NXT,NZT,NTR,NPT,NXTL,NZTL,NXSTRT,
     *             NZSTRT,KC,IT,IFLOW,ICHORD,ISPAN,INTDIR
      COMMON/BLC9/ NSEP(41),ICASE(41)
      COMMON/TAPE/ TAPEIN,TAPEOT,TAPEGP,TAPEPF,TAPEDT,TAPEVL
C - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
      NZ1P=NZ1+1
      NZ2M=NZ2-1
      IF(NX.NE.NXSTRT) GO TO 20
      DO 10 I=NZ1P,NZ2M
   10 ICASE(I)=1
      IF(IFLOW.EQ.2) GO TO 15
      ICASE(NZ1)=1
      ICASE(NZ2)=2
      RETURN
   15 ICASE(NZ1)=2
      ICASE(NZ2)=1
      RETURN
   20 IF(IFLOW.EQ.2) GO TO 40
      ICASE(NZ1)=3
      ICASE(NZ2)=5
      DO 30 I=NZ1P,NZ2M
      ICASE(I)=4
      IF(NSEP(I+1).NE.0) ICASE(I)=6
      IF(NSEP(I-1).NE.0) ICASE(I)=3
      IF(NSEP(I+1).NE.0.AND.NSEP(I-1).NE.0) GO TO 25
      GO TO 30
   25 IF(NSEP(I).NE.0) GO TO 30
      NSEP(I)=1
      WRITE(TAPEOT,100) I,NX
   30 CONTINUE
      RETURN
   40 ICASE(NZ2)=3
      ICASE(NZ1)=5
      DO 50 I=NZ1P,NZ2M
      ICASE(I)=4
      IF(NSEP(I-1).NE.0) ICASE(I)=6
      IF(NSEP(I+1).NE.0) ICASE(I)=3
      IF(NSEP(I+1).NE.0.AND.NSEP(I-1).NE.0) GO TO 45
      GO TO 50
   45 IF(NSEP(I).NE.0) GO TO 50
      NSEP(I)=1
      WRITE(TAPEOT,100) I,NX
   50 CONTINUE
      RETURN
  100 FORMAT(/9X26H** CALCULATIONS ALONG NZ =,I4,2X18HTERMINATED AT NX =
     *       ,I4/9X40H   BECAUSE OF MERGING SEPARATION REGIONS//)
      END
```

```
      SUBROUTINE OUTPUT(LL)
      DIMENSION Y(101),SUMTBL(10,41),CVC(101),BETA(101),USUSE(101)
      DIMENSION WPRNT(101),ELPRNT(101),SVC(101)
      INTEGER TAPEIN,TAPEOT,TAPEGP,TAPEPF,TAPEDT,TAPEVL
      COMMON/BLC0/ NX,NZ,NP,NXT,NZT,NTR,NPT,NXTL,NZTL,NXSTRT,
     *             NZSTRT,KC,IT,IFLOW,ICHORD,ISPAN,INTDIR
      COMMON/BLC3/ X(81),Z(41),DETA(101),ETA(101),ETAE,VGF,CEL,BEL1,BEL
      COMMON/BLC5/ F(101,2,2),U(101,2,2),V(101),G(101,2,2),W(101,2,2),
     *             T(101),TPROF(620),TPCF(10)
      COMMON/BLC9/ NSEP(41),ICASE(41)
      COMMON/PRSC/ PS1(2,2),H1(2,2),H2(2,2),UE(2,2),
     *             WE(2,2),CK1(2,2),CK2(2,2),CK12(2,2),CK21(2,2),
     *             THETA(2,2),PFRS,UFRS,CNUFRS,UREF1,WNP
      COMMON/BLCE/ EDV(101)
      COMMON/CONS/ SALFA,SALFAS,SBETA,SBETAS,SLAMDA,SSIGMA,SSGMAS,USTOP,
     *             ZIOTAE,ZIOTAL,RW2
      COMMON/STOR/ CON,EPS,ICONVE,ITMAX,WTEDG
      COMMON/TURB/ E(101,2,2),WT(101),WT2(101,2,2),ELT(101)
      COMMON/DERV/ DUDX,DWDX,DUDZ,DWDZ
      COMMON/IPRT/ IPZ,IPX,IPRINT,EPSV,EPST
      COMMON/TAPE/ TAPEIN,TAPEOT,TAPEGP,TAPEPF,TAPEDT,TAPEVL
C - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
      GO TO (10,220), LL
   10 USE=SQRT(UE(2,2)**2+WE(2,2)**2+2.*UE(2,2)*WE(2,2)*
     *         COS(THETA(2,2)))
      DUSEDS=((UE(2,2)+WE(2,2)*COS(THETA(2,2)))*(UE(2,2)/H1(2,2)*
     *         DUDX+WE(2,2)/H2(2,2)*DUDZ)+(WE(2,2)+UE(2,2)*
     *         COS(THETA(2,2)))*(UE(2,2)/H1(2,2)*DWDX+WE(2,2)/H2(2,2)*
     *         DWDZ)+UE(2,2)*WE(2,2)*SIN(THETA(2,2))*(UE(2,2)*
     *         ((CK12(2,2)+CK21(2,2)*COS(THETA(2,2)))/SIN(THETA(2,2))+
     *         CK1(2,2))+WE(2,2)*((CK21(2,2)+CK12(2,2)*COS(THETA(2,2)))/
     *         SIN(THETA(2,2))+CK2(2,2))))/USE**2
      URUE=UREF1/UE(2,2)
      US1=(UE(2,2)/USE)**2
      US2=(UREF1/USE)**2
      US3=2.*UE(2,2)*UREF1/USE**2*COS(THETA(2,2))
      B1=URUE*SIN(THETA(2,2))
      B2=URUE*COS(THETA(2,2))
      RX=UE(2,2)*PS1(2,2)/CNUFRS
      TRCMN=SQRT(CNUFRS*PS1(2,2)/UE(2,2))
      WSINTH=B1/URUE
      IF(KC.EQ.1) WSINTH=0.
      NPYES=0
      NPMO3=(NP-1)/3
      ENPMO3=(NP-1.)/3.
      DIFFNP=ENPMO3-NPMO3
      IF(DIFFNP.GT..1) NPYES=1
      DO 20 J=1,NP
   20 Y(J)=TRCMN*ETA(J)
      CVCT=UREF1/UE(2,2)*COS(THETA(2,2))
      CVCD=(1.+WE(2,2)/UE(2,2)*COS(THETA(2,2)))*URUE
      DO 80 J=1,NP
      USUSE(J)=SQRT(US1*U(J,2,2)**2+US2*W(J,2,2)**2+US3*U(J,2,2)*
     *           W(J,2,2))
      BETA(J)=0.
      IF(J.GT.1.AND.KC.NE.1)
     *  BETA(J)=57.29578*ATAN(B1*W(J,2,2)/(U(J,2,2)+B2*W(J,2,2)))
      SVC(J)=WSINTH*W(J,2,2)
   80 CVC(J)=(U(J,2,2)+W(J,2,2)*CVCT)/CVCD
      IF(KC.NE.1) BETA(1)=57.29578*ATAN(B1*T(1)/(V(1)+B2*T(1)))
```

```
      IF(IPRINT.EQ.0) GO TO 111
      WRITE(TAPEOT,280)
      WSCALE=US1/(US2*RX)
      ELSCAL=URUE*PS1(2,2)
      WPRNT(1)=9.99999E+37
      ELPRNT(1)=0.
      DO 84 J=2,NP
      WPRNT(J)=WT(J)*WSCALE
 84   ELPRNT(J)=ELT(J)*ELSCAL
      WRITE(TAPEOT,290) (J,Y(J),CVC(J),SVC(J),USUSE(J),BETA(J),
     *           E(J,2,2),WPRNT(J),ELPRNT(J),EDV(J),  J=1,NP,3)
      IF(NPYES.EQ.0) GO TO 111
      WRITE(TAPEOT,290) NP,Y(NP),CVC(NP),SVC(NP),USUSE(NP),BETA(NP),
     *                   E(NP,2,2),WPRNT(NP),ELPRNT(NP),EDV(NP)
 111  BETA(NP)=BETA(NP)/57.29578
      CTRM=2./SQRT(RX)
      UEUF=UE(2,2)/UFRS
      S1SQRX=PS1(2,2)/SQRT(RX)
      COSTH=COS(THETA(2,2))
      CTRM2=UE(2,2)*UREF1/UFRS**2*T(1)*COSTH
      CFC=CTRM*(UEUF**2*V(1)+CTRM2)
      CFN=CTRM*UE(2,2)*UREF1/UFRS**2*T(1)*SIN(THETA(2,2))
      CI2=0.
      CI3=0.
      CI4=0.
      WEUECT=WE(2,2)/UE(2,2)*COSTH
      URUECT=UREF1/UE(2,2)*COSTH
      C2=(U(1,2,2)+URUECT*W(1,2,2))**2
      C3=W(1,2,2)**2
      C4=U(1,2,2)**2
      DO 170 J=2,NP
      C22=(U(J,2,2)+URUECT*W(J,2,2))**2
      C33=W(J,2,2)**2
      C44=U(J,2,2)**2
      CI2=CI2+.5*(C2+C22)*DETA(J-1)
      CI3=CI3+.5*(C3+C33)*DETA(J-1)
      CI4=CI4+.5*(C4+C44)*DETA(J-1)
      C2=C22
      C3=C33
 170  C4=C44
      TNUM=F(NP,2,2)+URUECT*G(NP,2,2)
      TDEN=1.+WEUECT
      DLSTS=S1SQRX*(ETA(NP)-(F(NP,2,2)+UREF1/UE(2,2)*G(NP,2,2)
     *          *COSTH)/TDEN)
      THTAS=S1SQRX*(TNUM/TDEN-CI2/TDEN**2)
      IF(KC.GT.1) GO TO 180
      CFC=CTRM*UEUF**2*V(1)
      CFN=0.
      DLSTS=S1SQRX*(ETA(NP)-F(NP,2,2))
      THTAS=S1SQRX*(F(NP,2,2)-CI4)
 180  IF(ABS(WE(2,2)).GT.1.E-8) GO TO 200
      DLSTN=0.
      THTAN=0.
      GO TO 210
 200  DLSTN=S1SQRX*(ETA(NP)-UREF1/WE(2,2)*G(NP,2,2))
      THTAN=S1SQRX*UREF1/WE(2,2)*(G(NP,2,2)-UREF1/WE(2,2)*CI3)
 210  IF(IPRINT.EQ.1) WRITE(TAPEOT,300) CFC,DLSTS,THTAS,BETA(1),CFN,
     *   DLSTN,THTAN,RX,USE,DUSEDS
      SUMTBL(1,NZ)=V(1)
      SUMTBL(2,NZ)=T(1)
```

```
        SUMTBL(3,NZ)=CFC
        SUMTBL(4,NZ)=CFN
        SUMTBL(5,NZ)=BETA(1)
        SUMTBL(6,NZ)=DLSTS
        SUMTBL(7,NZ)=DLSTN
        SUMTBL(8,NZ)=THTAS
        SUMTBL(9,NZ)=THTAN
        SUMTBL(10,NZ)=RX
        RETURN
  220   WRITE(TAPEOT,310) X(NX)
        DO 240 I=NZSTRT,NZT
        IF(NSEP(I).EQ.0) GO TO 230
        WRITE(TAPEOT,340) I,Z(I)
        GO TO 240
  230   WRITE(TAPEOT,320) I,Z(I),(SUMTBL(J,I),J=1,10)
  240   CONTINUE
        WRITE(TAPEOT,330)
        RETURN
C - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
  280   FORMAT(1H0,2X1HJ,4X1HY,10X4HU/UR,9X4HW/UR,9X4HQ/QE,9X4HBETA,
     *          8X7HE/UR**2,6X5HOMEGA,9X2HEL,11X3HEPS)
  290   FORMAT(1H ,I3,1PE10.3,8E13.4)
  300   FORMAT(2X6HCFC   =,1PE14.6,3X6HDLSTS=,E14.6,3X6HTHTAS=,E14.6,
     *          3X6HBETA1=,E14.6/2X6HCFN   =,E14.6,3X6HDLSTN=,E14.6,
     *          3X6HTHTAN=,E14.6,3X6HRX    =,E14.6/2X6HQE    =,E14.6,
     *          3X6HDQEDX=,E14.6/1H0,62(2H*-))
  310   FORMAT(1H0,35X28H****-- SUMMARY TABLE FOR X =,1PE10.2,
     *          1X7H --****/1H0,3H NZ,5X1HZ,
     *          9X5HVWALL,6X5HTWALL,7X3HCFC,8X3HCFN,7X4HBETA,
     *          7X5HDLSTS,6X5HDLSTN,6X5HTHTAS,6X5HTHTAN,7X2HRX)
  320   FORMAT(1H ,I3,1PE10.3,10E11.3)
  330   FORMAT(1H0,62(2H*-)/)
  340   FORMAT(1H ,I3,1PE10.3,3X17H*** SEPARATED ***)
        END
```

```
      SUBROUTINE PRESUR
      INTEGER TAPEIN,TAPEOT,TAPEGP,TAPEPF,TAPEDT,TAPEVL
      DIMENSION APC(10)
      COMMON/BLC0/ NX,NZ,NP,NXT,NZT,NTR,NPT,NXTL,NZTL,NXSTRT,
     *             NZSTRT,KC,IT,IFLOW,ICHORD,ISPAN,INTDIR
      COMMON/BLC3/ X(81),Z(41),DETA(101),ETA(101),ETAE,VGP,CEL,BEL1,BEL2
      COMMON/CHRD/ WNP1(81),WNPN(81)
      COMMON/PRSC/ PS1(2,2),H1(2,2),H2(2,2),UE(2,2),
     *             WE(2,2),CK1(2,2),CK2(2,2),CK12(2,2),CK21(2,2),
     *             THETA(2,2),PFRS,UFRS,CNUFRS,UREF1,WNP
      COMMON/PRES/ P1(2,2),P2(2,2),P3(2,2),P4(2,2),P5(2,2),
     *             P6(2,2),P7(2,2),P8(2,2),P9(2,2),P10(2,2),P11(2,2),
     *             P12(2,2),P13(2,2),P14(2,2)
      COMMON/DERV/ DUDX,DWDX,DUDZ,DWDZ
      COMMON/TAPE/ TAPEIN,TAPEOT,TAPEGP,TAPEPF,TAPEDT,TAPEVL
      COMMON/IPRT/ IPZ,IPX,IPRINT,EPSV,EPST
      COMMON/SVNM/ IPCF,IPRF
C - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
      IF(NZ.EQ.NZT) GO TO 20
      IF(NZ.GT.NZSTRT) GO TO 10
      ICNT=(NZ-1)*NXTL+NX
      GO TO 30
   10 ICNT=(NZ-2)*NXTL+NX
      GO TO 30
   20 ICNT=(NZ-3)*NXTL+NX
   30 IPNT2=ICNT
      CALL STORIT(TAPEGP,APC,IPCF,IPNT2,1)
      H11=APC(1)
      H21=APC(2)
      THETA1=APC(7)
      S11=APC(8)
      UE1=APC(9)
      WE1=APC(10)
      ICNT=ICNT+NXTL
      IPNT2=ICNT
      CALL STORIT(TAPEGP,APC,IPCF,IPNT2,1)
      H12=APC(1)
      H22=APC(2)
      THETA2=APC(7)
      S12=APC(8)
      UE2=APC(9)
      WE2=APC(10)
      ICNT=ICNT+NXTL
      IPNT2=ICNT
      CALL STORIT(TAPEGP,APC,IPCF,IPNT2,1)
      H13=APC(1)
      H23=APC(2)
      THETA3=APC(7)
      S13=APC(8)
      UE3=APC(9)
      WE3=APC(10)
      IF(NZ.EQ.NZT) GO TO 60
      IF(NZ.GT.NZSTRT) GO TO 50
   40 Z1=Z(NZ)
      Z2=Z(NZ+1)
      Z3=Z(NZ+2)
      A1=(Z1-Z2)*(Z1-Z3)
      A2=(Z2-Z1)*(Z2-Z3)
      A3=(Z3-Z1)*(Z3-Z2)
      D1=(2.*Z1-Z3-Z2)/A1
```

```
      D2=(Z1-Z3)/A2
      D3=(Z1-Z2)/A3
      DWDZ=UREF1*WNP1(NX)
      GO TO 70
   50 Z1=Z(NZ-1)
      Z2=Z(NZ)
      Z3=Z(NZ+1)
      A1=(Z2-Z1)*(Z3-Z1)
      A2=(Z2-Z1)*(Z3-Z2)
      A3=(Z3-Z2)*(Z3-Z1)
      D1=-(Z3-Z2)/A1
      D2=(Z3-2.*Z2+Z1)/A2
      D3=(Z2-Z1)/A3
      DWDZ=D1*WE1+D2*WE2+D3*WE3
      GO TO 70
   60 Z1=Z(NZ-2)
      Z2=Z(NZ-1)
      Z3=Z(NZ)
      A1=(Z2-Z1)*(Z3-Z1)
      A2=(Z2-Z1)*(Z3-Z2)
      A3=(Z3-Z2)*(Z3-Z1)
      D1=(Z3-Z2)/A1
      D2=-(Z3-Z1)/A2
      D3=(2.*Z3-Z1-Z2)/A3
      DWDZ=UREF1*WNPN(NX)
   70 DUDZ=D1*UE1+D2*UE2+D3*UE3
      WNP=DWDZ/UREF1
      DH1DZ=D1*H11+D2*H12+D3*H13
      DTHDZ=D1*THETA1+D2*THETA2+D3*THETA3
      DS1DZ=D1*S11+D2*S12+D3*S13
      CPR1=SQRT(CNUFRS*UE1*S11)*H21*SIN(THETA1)*H11/H21
     *     *UREF1/UE1
      CPR2=SQRT(CNUFRS*UE2*S12)*H22*SIN(THETA2)*H12/H22
     *     *UREF1/UE2
      CPR3=SQRT(CNUFRS*UE3*S13)*H23*SIN(THETA3)*H13/H23
     *     *UREF1/UE3
      DCPDZ=D1*CPR1+D2*CPR2+D3*CPR3
      IF(NX.EQ.NXT) GO TO 120
      IF(NX.GT.NXSTRT) GO TO 110
  100 ICNT=(NZ-1)*NXTL+NX
      INX=1
      GO TO 130
  110 ICNT=(NZ-1)*NXTL+NX-1
      INX=2
      GO TO 130
  120 ICNT=(NZ-1)*NXTL+NX-2
      INX=3
  130 IPNT2=ICNT
      CALL STORIT(TAPEGP,APC,IPCF,IPNT2,1)
      H21=APC(2)
      THETA1=APC(7)
      S11=APC(8)
      UE1=APC(9)
      WE1=APC(10)
      ICNT=ICNT+1
      IPNT2=ICNT
      CALL STORIT(TAPEGP,APC,IPCF,IPNT2,1)
      H22=APC(2)
      THETA2=APC(7)
      S12=APC(8)
```

```
      UE2=APC(9)
      WE2=APC(10)
      ICNT=ICNT+1
      IPNT2=ICNT
      CALL STORIT(TAPEGP,APC,IPCF,IPNT2,1)
      H23=APC(2)
      THETA3=APC(7)
      S13=APC(8)
      UE3=APC(9)
      WE3=APC(10)
      IF(INX.EQ.3) GO TO 160
      IF(INX.EQ.2) GO TO 150
 140  X1=X(NX)
      X2=X(NX+1)
      X3=X(NX+2)
      A1=(X1-X2)*(X1-X3)
      A2=(X2-X1)*(X2-X3)
      A3=(X3-X1)*(X3-X2)
      D1=(2.*X1-X3-X2)/A1
      D2=(X1-X3)/A2
      D3=(X1-X2)/A3
      GO TO 170
 150  X1=X(NX-1)
      X2=X(NX)
      X3=X(NX+1)
      A1=(X2-X1)*(X3-X1)
      A2=(X2-X1)*(X3-X2)
      A3=(X3-X2)*(X3-X1)
      D1=-(X3-X2)/A1
      D2=(X3-2.*X2+X1)/A2
      D3=(X2-X1)/A3
      GO TO 170
 160  X1=X(NX-2)
      X2=X(NX-1)
      X3=X(NX)
      A1=(X2-X1)*(X3-X1)
      A2=(X2-X1)*(X3-X2)
      A3=(X3-X2)*(X3-X1)
      D1=(X3-X2)/A1
      D2=-(X3-X1)/A2
      D3=(2.*X3-X1-X2)/A3
 170  DUDX=D1*UE1+D2*UE2+D3*UE3
      IF(NX.EQ.NXSTRT) DUDX=(UE2-UE1)/(X2-X1)
      DBPDX=D1*SQRT(CNUFRS*UE1*S11)*H21*SIN(THETA1)+
     *       D2*SQRT(CNUFRS*UE2*S12)*H22*SIN(THETA2)+
     *       D3*SQRT(CNUFRS*UE3*S13)*H23*SIN(THETA3)
 190  DWDX=D1*WE1+D2*WE2+D3*WE3
      HTRM1=H21*SIN(THETA1)
      HTRM2=H22*SIN(THETA2)
      HTRM3=H23*SIN(THETA3)
      DHTRMX=D1*HTRM1+D2*HTRM2+D3*HTRM3
      COTTH=COTAN(THETA(2,2))
      SINTH=SIN(THETA(2,2))
      COSTH=COS(THETA(2,2))
      BPR=SQRT(CNUFRS*UE(2,2)*PS1(2,2))*H2(2,2)*SINTH
      H1H2=H1(2,2)/H2(2,2)
      IF(KC.EQ.4) GO TO 460
      DUDZ=0.
      DWDZ=0.
      DS1DZ=0.
```

```
      DCPDZ=0.
  460 DLOGS1=DS1DZ/PS1(2,2)
      P1(2,2)=.5*(PS1(2,2)/(H1(2,2)*UE(2,2))*DUDX+1.)+PS1(2,2)/
     *         (H1(2,2)*H2(2,2)*SINTH)*DHTRMX
      P2(2,2)=PS1(2,2)/(UE(2,2)*H1(2,2))*DUDX-PS1(2,2)*CK1(2,2)*COTTH
      P3(2,2)=-PS1(2,2)*COTTH*CK2(2,2)*UREF1/UE(2,2)
      P4(2,2)=PS1(2,2)*CK21(2,2)
      P5(2,2)=PS1(2,2)/H2(2,2)*UREF1/UE(2,2)**2*DUDZ+CK12(2,2)*
     *         PS1(2,2)*UREF1/UE(2,2)
      P6(2,2)=PS1(2,2)/(H1(2,2)*BPR)*DCPDZ
      P7(2,2)=PS1(2,2)/H2(2,2)*UREF1/UE(2,2)
      P8(2,2)=PS1(2,2)*CK2(2,2)/SINTH*(UREF1/UE(2,2))**2
      P9(2,2)=PS1(2,2)*CK1(2,2)/SINTH*UE(2,2)/UREF1
      P10(2,2)=PS1(2,2)/H1(2,2)
      P11(2,2)=PS1(2,2)*(1./(UE(2,2)*H1(2,2))*DUDX+WE(2,2)/(UE(2,2)
     *          **2*H2(2,2))*DUDZ-COTTH*CK1(2,2)+CK2(2,2)/SINTH*
     *          (WE(2,2)/UE(2,2))**2+CK12(2,2)*WE(2,2)/UE(2,2))
      P12(2,2)=PS1(2,2)/(UE(2,2)*UREF1)*(UE(2,2)/H1(2,2)*DWDX+
     *          WE(2,2)/H2(2,2)*DWDZ-COTTH*CK2(2,2)*WE(2,2)**2+
     *          CK1(2,2)/SINTH*UE(2,2)**2+CK21(2,2)*WE(2,2)*UE(2,2))
      P13(2,2)=1.-PS1(2,2)/(H1(2,2)*UE(2,2))*DUDX
      P14(2,2)=P7(2,2)*(DLOGS1-DUDZ/UE(2,2))
      IF(KC.NE.1) RETURN
      DWDZ=UREF1*WNP
      NX1=NX+1-INX
      NX2=NX1+1
      NX3=NX2+1
      IF(NZ.EQ.NZSTRT) D2WDZX=D1*WNP1(NX1)+D2*WNP1(NX2)+D3*WNP1(NX3)
      IF(NZ.EQ.NZT) D2WDZX=D1*WNPN(NX1)+D2*WNPN(NX2)+D3*WNPN(NX3)
      P3(2,2)=P7(2,2)
      P5(2,2)=0.
      P6(2,2)=P7(2,2)
      P8(2,2)=0.
      P9(2,2)=0.
      P12(2,2)=PS1(2,2)/UREF1*(UREF1*D2WDZX/H1(2,2)
     *         +DWDZ**2/(UE(2,2)*H2(2,2))+CK21(2,2)*DWDZ)
      RETURN
      END
```

```
      SUBROUTINE PROFIL(LL)
      COMMON/BLC0/ NX,NZ,NP,NXT,NZT,NTR,NPT,NXTL,NZTL,NXSTRT,
     *             NZSTRT,KC,IT,IFLOW,ICHORD,ISPAN,INTDIR
      COMMON/BLC3/ X(81),Z(41),DETA(101),ETA(101),ETAE,VGP,CEL,BEL1,BEL2
      COMMON/BLC5/ F(101,2,2),U(101,2,2),V(101),G(101,2,2),W(101,2,2),
     *             T(101),TPROF(620),TPCF(10)
      COMMON/BLCE/ EDV(101)
      COMMON/PRSC/ PS1(2,2),H1(2,2),H2(2,2),UE(2,2),
     *             WE(2,2),CK1(2,2),CK2(2,2),CK12(2,2),CK21(2,2),
     *             THETA(2,2),PFRS,UFRS,CNUFRS,UREF1,WNP
      COMMON/TURB/ E(101,2,2),WT(101),WT2(101,2,2),ELT(101)
      COMMON/STOR/ CON,EPS,ICONVE,ITMAX,WTEDG
      COMMON/CONS/ SALFA,SALFAS,SBETA,SBETAS,SLAMDA,SSIGMA,SSGMAS,USTOP,
     *             ZIOTAE,ZIOTA ,RW2
C - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
      GO TO (10,40,70), LL
   10 IF(NTR.EQ.0) GO TO 15
      CALL START
      GO TO 40
   15 V(1)=.332
      T(1)=.332
      WCON=20./SBETA
      REY=UE(2,2)*PS1(2,2)/CNUFRS
      WTEDG=UREF1/UE(2,2)*SQRT(REY*ZIOTAE/SALFAS)/(ZIOTAL*ETAE)
      C1=ETA(NP)*V(1)
      C2=3.-2.*C1
      C3=-2.+C1
      C1D2=C1/2.
      C3D4=C3/4.
      DO 20 J=1,NP
      ERAT=ETA(J)/ETA(NP)
      ERATC2=ERAT*C2
      ERATSQ=ERAT**2
      ERTSQ3=ERATSQ*C3
      F(J,2,2)=(C1D2+ERATC2/3.+C3D4*ERATSQ)*ERATSQ*ETA(NP)
      U(J,2,2)=(C1+ERATC2+ERTSQ3)*ERAT
      G(J,2,2)=F(J,2,2)
      W(J,2,2)=U(J,2,2)
      E(J,2,2)=0.
      IF(J.EQ.1) GO TO 20
      WT(J)=WCON/ETA(J)**2
      IF(WT(J).LT.WTEDG) WT(J)=WTEDG
      WT2(J,2,2)=WT(J)**2
   20 CONTINUE
      E(NP,2,2)=ZIOTAE
      WT2(1,2,2)=WT2(2,2,2)
   40 K=NP+1
      L=NPT
   50 DO 60 J=K,L
      F(J,2,2)=ETA(J)+F(K-1,2,2)-ETA(K-1)
      U(J,2,2)=1.
      W(J,2,2)=W(K-1,2,2)
      G(J,2,2)=W(K-1,2,2)*(ETA(J)-ETA(NP))+G(K-1,2,2)
      E(J,2,2)=E(K-1,2,2)
      WT(J)=WTEDG
   60 WT2(J,2,2)=WTEDG**2
      IF(LL.NE.1) RETURN
      CALL ARRYMV(4,1,2)
      RETURN
   70 K=NP+1
```

```
      NP=NP+2
      IF(NP.LE.NPT) GO TO 80
      NP=NPT
80    L=NP
      GO TO 50
      END
```

```
      SUBROUTINE SHIFT(NZ1,NZ2)
      INTEGER TAPEIN,TAPEOT,TAPEGP,TAPEPF,TAPEDT,TAPEVL
      COMMON/BLC0/ NX,NZ,NP,NXT,NZT,NTR,NPT,NXTL,NZTL,NXSTRT,
     *             NZSTRT,KC,IT,IFLOW,ICHORD,ISPAN,INTDIR
      COMMON/BLC5/ F(101,2,2),U(101,2,2),V(101),G(101,2,2),W(101,2,2),
     *             T(101),TPROF(620),TPCF(10)
      COMMON/BLC9/ NSEP(41),ICASE(41)
      COMMON/TAPE/ TAPEIN,TAPEOT,TAPEGP,TAPEPF,TAPEDT,TAPEVL
      COMMON/SVNM/ IPCF,IPRF
C - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
C   WRITE (2,2) FLOW ARRAYS ON DISK
      IPNT=NX+(NZ-1)*NXTL
      CALL ARRYMV(1,2,2)
      CALL STORIT(TAPEPF,TPROF,IPRF,IPNT,0)
      INC=3-2*IFLOW
      IGO=ICASE(NZ)
      GO TO (100,200,300,300,400,500), IGO
C   SET POINTERS FOR NX=NXSTRT EXCEPT AT END OF INITIAL LINE SWEEP
  100 IPNTG=NX+(NZ+INC-1)*NXTL
      GO TO 650
C   SET POINTERS FOR NX=NXSTRT AND END OF INITIAL LINE SWEEP
  200 IF(IFLOW.EQ.2) GO TO 210
      IPNT=NX+(NZ1-1)*NXTL
      IPNT11=NX+NZ1*NXTL
      IPNTG=NX+1+(NZ1-1)*NXTL
      GO TO 220
  210 IPNT=NX+(NZ2-1)*NXTL
      IPNT11=NX+(NZ2-2)*NXTL
      IPNTG=NX+1+(NZ2-1)*NXTL
  220 CALL STORIT(TAPEPF,TPROF,IPRF,IPNT,1)
      CALL ARRYMV(2,1,2)
      CALL ARRYMV(5,1,2)
      GO TO 600
C   SET POINTERS FOR GENERAL CASE AND BEGINNING OF LINE SWEEP
  300 CALL ARRYMV(6,1,2)
      IPNTG=NX+(NZ+INC-1)*NXTL
      NZNEXT=NZ+INC
      IF(ICASE(NZNEXT).NE.4) GO TO 650
      CALL ARRYMV(6,2,1)
      IPNT11=NX-1+(NZ+2*INC-1)*NXTL
      GO TO 600
C   GENERAL CASE...END OF LINE SWEEP
  400 IF(NX.EQ.NXT) RETURN
      IF(IFLOW.EQ.2) GO TO 410
      IPNT=NX+(NZ1-1)*NXTL
      IPNT11=NX+NZ1*NXTL
      IPNTG=NX+1+(NZ1-1)*NXTL
      GO TO 420
  410 IPNT=NX+(NZ2-1)*NXTL
      IPNT11=NX+(NZ2-2)*NXTL
      IPNTG=NX+1+(NZ2-1)*NXTL
  420 CALL STORIT(TAPEPF,TPROF,IPRF,IPNT,1)
      CALL ARRYMV(2,1,2)
      CALL ARRYMV(5,1,2)
      GO TO 600
C   END OF ATTACHED REGION
  500 NZN=NZ
  510 NZN=NZN+INC
      NZNP=NZN+INC
      IF(NSEP(NZNP).EQ.1) GO TO 510
```

```
      IPNT=NX-1+NZN*NXTL
      CALL STORIT(TAPEPF,TPROF,IPRF,IPNT,1)
      CALL ARRYMV(2,1,2)
      CALL ARRYMV(5,1,2)
      IPNT11=NX-1+(NZN+INC)*NXTL
      IPNTG=NX+NZN*NXTL
C  READ IN (1,1) FLOW ARRAYS AND (2,2) GEOMETRY PARAMETERS
 600  CALL STORIT(TAPEPF,TPROF,IPRF,IPNT11,1)
      CALL ARRYMV(2,1,1)
 650  CALL STORIT(TAPEGP,TPCF,IPCF,IPNTG,1)
      CALL ARRYMV(3,2,2)
      RETURN
      END
```

```
      SUBROUTINE SHIFT2(NZ1,NZ2)
      INTEGER TAPEIN,TAPEOT,TAPEGP,TAPEPF,TAPEDT,TAPEVL
      COMMON/BLC0/ NX,NZ,NP,NXT,NZT,NTR,NPT,NXTL,NZTL,NXSTRT,
     *             NZSTRT,KC,IT,IFLOW,ICHORD,ISPAN,INTDIR
      COMMON/BLC5/ F(101,2,2),U(101,2,2),V(101),G(101,2,2),W(101,2,2),
     *             T(101),TPROF(620),TPCF(10)
      COMMON/BLC9/ NSEP(41),ICASE(41)
      COMMON/TAPE/ TAPEIN,TAPEOT,TAPEGP,TAPEPF,TAPEDT,TAPEVL
      COMMON/SVNM/ IPCF,IPRF
C - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
      IF(IFLOW.EQ.2) GO TO 50
C  SEPARATION ON LOWER BOUNDARY...IFLOW=1
      IF(NZ.GT.NZ1) GO TO 10
 5    NZ1=NZ1+1
      NSEP(NZ1)=1
      IF(NSEP(NZ1+1).EQ.1) GO TO 5
      IPNT=NX-1+(NZ1-1)*NXTL
      CALL STORIT(TAPEPF,TPROF,IPRF,IPNT,1)
      CALL ARRYMV(2,1,2)
      IPNT11=NX-1+NZ1*NXTL
      IPNTG=NX+(NZ1-1)*NXTL
      ICASE(NZ1)=3
      GO TO 80
C  SEPARARION ON UPPER BOUNDARY...IFLOW=1
 10   IF(NZ.LT.NZ2) GO TO 20
 15   NZ2=NZ2-1
      NSEP(NZ2)=1
      IF(NSEP(NZ2-1).EQ.1) GO TO 15
      IF(NX.EQ.NXT) RETURN
      IPNT=NX+(NZ1-1)*NXTL
      CALL STORIT(TAPEPF,TPROF,IPRF,IPNT,1)
      CALL ARRYMV(2,1,2)
      IPNT11=NX+NZ1*NXTL
      IPNTG=NX+1+(NZ1-1)*NXTL
      GO TO 80
C  SEPARATION AT INTERIOR POINT...IFLOW=1
 20   NZN=NZ+1
      NSEP(NZN-1)=1
      IF(NSEP(NZN+1).EQ.1) GO TO 20
      IPNT=NX-1+(NZN-1)*NXTL
      CALL STORIT(TAPEPF,TPROF,IPRF,IPNT,1)
      CALL ARRYMV(2,1,2)
      IPNT11=NX-1+NZN*NXTL
      IPNTG=NX+(NZN-1)*NXTL
      ICASE(NZN)=3
      GO TO 80
C  SEPARARION ON UPPER BOUNDARY...IFLOW=2
 50   IF(NZ.LT.NZ2) GO TO 60
 55   NZ2=NZ2-1
      NSEP(NZ2)=1
      IF(NSEP(NZ2-1).EQ.1) GO TO 55
      IPNT=NX-1+(NZ2-1)*NXTL
      CALL STORIT(TAPEPF,TPROF,IPRF,IPNT,1)
      CALL ARRYMV(2,1,2)
      IPNT11=NX-1+(NZ2-2)*NXTL
      IPNTG=NX+(NZ2-1)*NXTL
      ICASE(NZ2)=3
      GO TO 80
C  SEPARATION ON LOWER BOUNDARY...IFLOW=2
 60   IF(NZ.GT.NZ1) GO TO 70
```

```
 65   NZ1=NZ1+1
      NSEP(NZ1)=1
      IF(NSEP(NZ1+1).EQ.1) GO TO 65
      IF(NX.EQ.NXT) RETURN
      IPNT=NX+(NZ2-1)*NXTL
      CALL STORIT(TAPEPF,TPROF,IPRF,IPNT,1)
      CALL ARRYMV(2,1,2)
      IPNT11=NX+(NZ2-2)*NXTL
      IPNTG=NX+1+(NZ2-1)*NXTL
      GO TO 80
C   SEPARATION AT INTERIOR POINT...IFLOW=2
 70   NZN=NZ-1
      NSEP(NZN+1)=1
      IF(NSEP(NZN-1).EQ.1) GO TO 70
      IPNT=NX-1+(NZN-1)*NXTL
      CALL STORIT(TAPEPF,TPROF,IPRF,IPNT,1)
      CALL ARRYMV(2,1,2)
      IPNT11=NX-1+(NZN-2)*NXTL
      IPNTG=NX+(NZN-1)*NXTL
      ICASE(NZN)=3
 80   CALL ARRYMV(5,1,2)
      CALL STORIT(TAPEPF,TPROF,IPRF,IPNT11,1)
      CALL ARRYMV(2,1,1)
      CALL STORIT(TAPEGP,TPCF,IPCF,IPNTG,1)
      CALL ARRYMV(3,2,2)
      RETURN
      END
```

```
      SUBROUTINE SOLVEW
      INTEGER TAPEIN,TAPEOT,TAPEGP,TAPEPF,TAPEDT,TAPEVL
      DIMENSION    ESAVE(101),WSAVE(101)
      COMMON/BLC0/ NX,NZ,NP,NXT,NZT,NTR,NPT,NXTL,NZTL,NXSTRT,
     *             NZSTRT,KC,IT,IFLOW,ICHORD,ISPAN,INTDIR
      COMMON/BLC3/ X(81),Z(41),DETA(101),ETA(101),ETAE,VGP,CEL,BEL1,BEL2
      COMMON/BLC5/ F(101,2,2),U(101,2,2),V(101),G(101,2,2),W(101,2,2),
     *             T(101),TPROF(620),TPCF(10)
      COMMON/PRES/ P1(2,2),P2(2,2),P3(2,2),P4(2,2),P5(2,2),
     *             P6(2,2),P7(2,2),P8(2,2),P9(2,2),P10(2,2),P11(2,2),
     *             P12(2,2),P13(2,2),P14(2,2)
      COMMON/PRSC/ PS1(2,2),H1(2,2),H2(2,2),UE(2,2),
     *             WE(2,2),CK1(2,2),CK2(2,2),CK12(2,2),CK21(2,2),
     *             THETA(2,2),PFRS,UFRS,CNUFRS,UREF1,WNP
      COMMON/TURB/ E(101,2,2),WT(101),WT2(101,2,2),ELT(101)
      COMMON/STOR/ CON,EPS,ICONVE,ITMAX,WTEDG
      COMMON/CONV/ ICOUNE,ICOUNW,INEGE,INEGW,DELV,DELT
      COMMON/CONS/ SALFA,SALFAS,SBETA,SBETAS,SLAMDA,SSIGMA,SSGMAS,USTOP,
     *             ZIOTAE,ZIOTAL,RW2
      COMMON/BLCE/ EDV(101)
      COMMON/BLC7/ PU(101),PW(101),QU(101),QW(101)
      COMMON/BLCY/ Y1(101),Y3(101),W1,W2,W3,W4
      COMMON/IPRT/ IPZ,IPX,IPRINT,EPSV,EPST
      COMMON/SHRT/ ISHORT
      COMMON/TAPE/ TAPEIN,TAPEOT,TAPEGP,TAPEPF,TAPEDT,TAPEVL
      COMMON/RELX/ RFTRB,RFVEL
C - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
      DO 1 J=2,NP
      IF(U(J,2,2).GE..999) GO TO 2
 1    CONTINUE
 2    NPL=J
      NPM=NPL-1
      U2=U(NPL,2,2)
      U1=U(NPM,2,2)
      ETADEL=ETA(NPM)+DETA(NPM)*(.999-U1)/(U2-U1)
      URUE=UREF1/UE(2,2)
      URUECT=2.*UREF1/UE(2,2)*COS(THETA(2,2))
      REY=UE(2,2)*PS1(2,2)/CNUFRS
      UTAU4=V(1)**2
      IF(KC.NE.1) UTAU4=UTAU4+(URUE*T(1))**2+URUECT*V(1)*T(1)
      WTEDG=SQRT(REY*ZIOTAE/SALFAS)/(ZIOTAL*ETADEL)
      UHAT=(UTAU4/REY)**.25
      JSTOP=1
      WCOEF=20./SBETA
      DO 25 J=2,NP
      UTOT=U(J,2,2)
      IF(KC.NE.1) UTOT=SQRT(UTOT**2+(URUE*W(J,2,2))**2
     *                       +URUECT*UTOT*W(J,2,2))
      UPLUS=UTOT/UHAT
      IF(UPLUS.GT.USTOP) GO TO 26
      WT(J)=WCOEF/ETA(J)**2
      WT2(J,2,2)=WT(J)**2
 25   JSTOP=JSTOP+1
 26   WT(1)=WT(2)
      WT2(1,2,2)=WT(2)**2
      E(1,2,2)=0.
      ICONVE=0
      PU(1)=0.
      QU(1)=0.
      PW(JSTOP)=WT2(JSTOP,2,2)
```

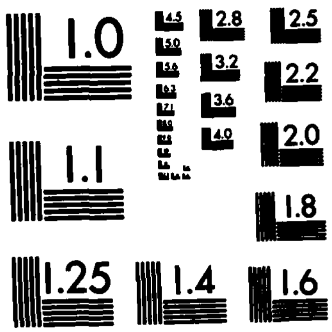
```
      QW(JSTOP)=0.
      E4M=1.+.5*SSGMAS*(EDV(1)+EDV(2))
      E5M=1.+.5*SSIGMA*(EDV(1)+EDV(2))
      DO 150 J=2,NPM
      ESAVE(J)=E(J,2,2)
      WSAVE(J)=WT2(J,2,2)
      E4P=1.+.5*SSGMAS*(EDV(J)+EDV(J+1))
      E5P=1.+.5*SSIGMA*(EDV(J)+EDV(J+1))
      VB=V(J)
      TB=T(J)
      FB=F(J,2,2)
      UB=U(J,2,2)
      GB=G(J,2,2)
      WB=W(J,2,2)
      EB=E(J,2,2)
      WTB=WT(J)
      E2=P1(2,2)*FB+P6(2,2)*GB
      DFB=CEL*(F(J,2,2)-F(J,1,2))
      DGB=BEL1*(G(J,2,2)-G(J,2,1))+BEL2*(G(J,1,1)-G(J,1,2))
      DVB=(E2+DFB+DGB)/(DETA(J)+DETA(J-1))
      RETB=URUE**2*REY*EB/WTB
      GAMS=1.-CON*EXP(-RETB)
      SHEAR2=VB*VB
      IF(KC.NE.1) SHEAR2=SHEAR2+(URUE*TB)**2+URUECT*TB*VB
      PRODE=GAMS*REY*SHEAR2/WTB
      DISSE=SBETAS*WTB
      ABE=.3
      PSIFIX=PRODE/DISSE-.7
      IF(PSIFIX.GT.ABE) ABE=PSIFIX
      X3=PRODE-(1.+ABE)*DISSE
      X7=-(CEL*UB+BEL2*WB)*E(J,1,2)-BEL1*WB*E(J,2,1)
     *                                 +BEL2*WB*E(J,1,1)
      A1= E4M*Y3(J)-DVB
      B1=-E4P*Y1(J)-E4M*Y3(J)+X3-(CEL*UB+BEL1*WB)
      C1= E4P*Y1(J)+DVB
      D1=X7-ABE*DISSE*EB
      B1S=B1+A1*QU(J-1)
      D1S=D1-A1*PU(J-1)
      PU(J)=D1S/B1S
      QU(J)=-C1/B1S
      IF(J.LE.JSTOP) GO TO 145
      WT2B=WT2(J,2,2)
      GAM=SALFA*(1.-CON*EXP(-RETB/RW2))/GAMS
      DLDY=(ELT(J+1)-ELT(J-1))/(DETA(J)+DETA(J-1))
      DLDY2=DLDY*DLDY
      PRODW=GAM*PRODE+X9
      SBETAT=SBETA+2.*SSIGMA*REY*DLDY2*URUE**2
      DISSW=SBETAT*WTB
      ABW=.3
      PSIFIX=PRODW/DISSW-.7
      IF(PSIFIX.GT.ABW) ABW=PSIFIX
      PSISTP=.5*PRODW/DISSW-.25
      IF(PSISTP.GT.ABW) ABW=PSISTP
      X6=PRODW-(1.+ABW)*DISSW
      X8=-(CEL*UB+BEL2*WB)*WT2(J,1,2)-BEL1*WB*WT2(J,2,1)
     *                                     +BEL2*WB*WT2(J,1,1)
      X9=2.*(P13(2,2)*UB+P14(2,2)*WB)
      A2= E5M*Y3(J)-DVB
      B2=-E5P*Y1(J)-E5M*Y3(J)+X6-(CEL*UB+BEL1*WB)
      C2= E5P*Y1(J)+DVB
```

```
      D2=X8-ABW*DISSW*WT2B
      B2S=B2+A2*QW(J-1)
      D2S=D2-A2*PW(J-1)
      PW(J)=D2S/B2S
      QW(J)=-C2/B2S
145   E4M=E4P
150   E5M=E5P
      WT2EDG=WTEDG*WTEDG
      E(NPL,2,2)=ZIOTAE
      WT2(NPL,2,2)=WT2EDG
      KON=NPM
      DO 170 J=2,NPM
      E(KON,2,2)=PU(KON)+QU(KON)*E(KON+1,2,2)
      IF(KON.LE.JSTOP) GO TO 170
      WT2(KON,2,2)=PW(KON)+QW(KON)*WT2(KON+1,2,2)
170   KON=KON-1
      ICOUNE=0
      INEGE=0
      DO 175 J=2,NPM
      IF(E(J,2,2).GE.0.) GO TO 176
      INEGE=INEGE+1
      E(J,2,2)=.5*ESAVE(J)
      IF(J.GE.(NPL-3)) E(J,2,2)=ZIOTAE
      GO TO 175
176   IF(ABS(E(J,2,2)-ESAVE(J)).GT.EPS*ESAVE(J)) ICOUNE=ICOUNE+1
175   E(J,2,2)=ESAVE(J)+RFTRB*(E(J,2,2)-ESAVE(J))
      ICOUNW=0
      INEGW=0
      DO 195 J=JSTOP,NPM
      IF(WT2(J,2,2).GT.0.) GO TO 196
      WT2(J,2,2)=.5*WSAVE(J)
      INEGW=INEGW+1
      GO TO 197
196   IF(ABS(WT2(J,2,2)-WSAVE(J)).GT.EPS*WSAVE(J)) ICOUNW=ICOUNW+1
      WT2(J,2,2)=WSAVE(J)+RFTRB*(WT2(J,2,2)-WSAVE(J))
197   WT(J)=SQRT(WT2(J,2,2))
195   CONTINUE
      ICOUNT=ICOUNW+ICOUNE+INEGW+INEGE
      IF(ICOUNT.LE.2) ICONVE=1
      NPP=NPL+1
      DO 200 J=NPP,NPT
      E(J,2,2)=E(NPL,2,2)
      WT(J)=WTEDG
200   WT2(J,2,2)=WT2EDG
      IF(ISHORT.EQ.1) RETURN
      IF(IPRINT.EQ.0) RETURN
      IF(IT.EQ.1) WRITE(TAPEOT,40)
      WRITE(TAPEOT,50) IT,V(1),DELV,T(1),DELT,INEGE,
     *                    INEGW,ICOUNE,ICOUNW
      RETURN
40    FORMAT(1H0,1X2HIT,5X7HV(WALL),7X7HDELV(1 ...
     *       7X5HINEGE,3X5HINEGW,3X6HICOUNE, ...
50    FORMAT(1H ,I3,1P4E14.5,4I8)
      END
```

AD-A123 285   A MICROCOMPUTER-BASED METHOD FOR PREDICTING VISCOUS FLOW ABOUT A SHIP HULL(U) DCW INDUSTRIES INC STUDIO CITY CA D C WILCOX DEC 82 DCW-R-28-01 ONR-CR062-735-1F   2/2

UNCLASSIFIED   N00014-82-C-0432   F/G 9/2   NL

END
FILMED
DTIC



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

```
      SUBROUTINE SOLV6
      DIMENSION USAVE(101),WSAVE(101)
      COMMON/BLC0/ NX,NZ,NP,NXT,NZT,NTR,NPT,NXTL,NZTL,NXSTRT,
     *             NZSTRT,KC,IT,IFLOW,ICHORD,ISPAN,INTDIR
      COMMON/BLC3/ X(81),Z(41),DETA(101),ETA(101),ETAE,VGP,CEL,BEL1,BEL
      COMMON/BLC5/ F(101,2,2),U(101,2,2),V(101),G(101,2,2),W(101,2,2),
     *             T(101),TPROF(620),TPCF(10)
      COMMON/BLC7/ PU(101),PW(101),QU(101),QW(101)
      COMMON/BLCY/ Y1(101),Y3(101),W1,W2,W3,W4
      COMMON/IPRT/ IPZ,IPX,IPRINT,EPSV,EPST
      COMMON/CONV/ ICOUNE,ICOUNW,INEGE,INEGW,DELV,DELT
      COMMON/RELX/ RFTRB,RFVEL
C - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
      VWSAVE=V(1)
      TWSAVE=T(1)
      NPM=NP-1
      KON=NPM
      DO 10 J=2,NPM
      USAVE(KON)=U(KON,2,2)
      WSAVE(KON)=W(KON,2,2)
      U(KON,2,2)=PU(KON)+QU(KON)*U(KON+1,2,2)
      W(KON,2,2)=PW(KON)+QW(KON)*W(KON+1,2,2)
   10 KON=KON-1
      U(1,2,2)=0.
      W(1,2,2)=0.
      DO 20 J=2,NPM
      U(J,2,2)=USAVE(J)+RFVEL*(U(J,2,2)-USAVE(J))
   20 W(J,2,2)=WSAVE(J)+RFVEL*(W(J,2,2)-WSAVE(J))
      F(1,2,2)=0.
      G(1,2,2)=0.
      CF1=0.
      CG1=0.
      DO 30 J=2,NP
      CF=U(J,2,2)
      CG=W(J,2,2)
      F(J,2,2)=F(J-1,2,2)+.5*(CF+CF1)*DETA(J-1)
      G(J,2,2)=G(J-1,2,2)+.5*(CG+CG1)*DETA(J-1)
      CF1=CF
   30 CG1=CG
      DO 35 J=2,NPM
      V(J)=(U(J+1,2,2)-U(J-1,2,2))/(DETA(J)+DETA(J-1))
   35 T(J)=(W(J+1,2,2)-W(J-1,2,2))/(DETA(J)+DETA(J-1))
      V(NP)=(1.-U(NPM,2,2))/DETA(NPM)
      T(NP)=(W(NP,2,2)-W(NPM,2,2))/DETA(NPM)
      V(1)=-W1*U(1,2,2)+W2*U(2,2,2)-W3*U(3,2,2)+W4*U(4,2,2)
      T(1)=-W1*W(1,2,2)+W2*W(2,2,2)-W3*W(3,2,2)+W4*W(4,2,2)
      DELV=V(1)-VWSAVE
      DELT=T(1)-TWSAVE
      RETURN
      END
```

```
      SUBROUTINE START
      INTEGER TAPEIN,TAPEOT,TAPEGP,TAPEPF,TAPEDT,TAPEVL
      DIMENSION YPLUS(101)
      COMMON/BLC0/ NX,NZ,NP,NXT,NZT,NTR,NPT,NXTL,NZTL,NXSTRT,
     *             NZSTRT,KC,IT,IFLOW,ICHORD,ISPAN,INTDIR
      COMMON/BLC3/ X(81),Z(41),DETA(101),ETA(101),ETAE,VGP,CEL,BEL1,BEL2
      COMMON/BLC5/ F(101,2,2),U(101,2,2),V(101),G(101,2,2),W(101,2,2),
     *             T(101),TPROF(620),TPCF(10)
      COMMON/PRSC/ PS1(2,2),H1(2,2),H2(2,2),UE(2,2),
     *             WE(2,2),CK1(2,2),CK2(2,2),CK12(2,2),CK21(2,2),
     *             THETA(2,2),PFRS,UFRS,CNUFRS,UREF1,WNP
      COMMON/BLCE/ EDV(101)
      COMMON/TURB/ E(101,2,2),WT(101),WT2(101,2,2),ELT(101)
      COMMON/STOR/ CON,EPS,ICONVE,ITMAX,WTEDG
      COMMON/TAPE/ TAPEIN,TAPEOT,TAPEGP,TAPEPF,TAPEDT,TAPEVL
      COMMON/CONS/ SALFA,SALFAS,SBETA,SBETAS,SLAMDA,SSIGMA,SSGMAS,USTOP,
     *             ZIOTAE,ZIOTAL,RW2
      COMMON/INTG/ ALAMI(41),CFXI(41),CFZI(41),DELTAI(41),
     *             THETXI(41),THETZI(41)
      DATA PI/3.14159265/,AKAPPA/.41/
C - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
      PS1(1,2)=PS1(2,2)-H1(2,2)*(X(NXSTRT+1)-X(NXSTRT))
      ALAMM=ALAMI(NZ)
      CFX=CFXI(NZ)
      CFZ=CFZI(NZ)
      DELTA=DELTAI(NZ)
      THETAX=THETXI(NZ)
      THETAZ=THETZI(NZ)
      RES=UE(2,2)*PS1(1,2)/CNUFRS
      REDELT=UE(2,2)*DELTA/CNUFRS
      ROOTRS=SQRT(RES)
      URUE=UREF1/UE(2,2)
      URUE2=URUE**2
      UTAU2=.5*(CFX+ABS(CFZ))
      UTAU=SQRT(UTAU2)
      DELTAP=UREF1*UTAU*DELTA/CNUFRS
      YSCALP=UTAU*URUE*ROOTRS
      COSTH=COS(THETA(2,2))
      SINTH=SIN(THETA(2,2))
      COTTH=COSTH/SINTH
      DO 10 J=1,NP
 10   YPLUS(J)=YSCALP*ETA(J)
C  VELOCITY PROFILES
      DOTXM3=DELTA/THETAX-3.
      DOTZM3=3.
      IF(ABS(THETAZ).GT.1.E-8) DOTZM3=DELTA/THETAZ-3.
      ENX=.5*(DOTXM3+SQRT(DOTXM3*DOTXM3-8.))
      ENZ=.5*(DOTZM3+SQRT(DOTZM3*DOTZM3-8.))
      SUBSCL=.5*ROOTRS*URUE2
      OTSCLX=(1.+WE(2,2)*COSTH/UE(2,2))*(ROOTRS/REDELT)**(1./ENX)
      OTSCLZ=(WE(2,2)*SINTH/UE(2,2))*(ROOTRS/REDELT)**(1./ENZ)
      V(1)=SUBSCL*CFX
      T(1)=SUBSCL*CFZ
      U(1,2,2)=0.
      W(1,2,2)=0.
      DO 20 J=2,NP
      IF(YPLUS(J).GE.DELTAP) GO TO 30
      U(J,2,2)=CFX*SUBSCL*ETA(J)
      W(J,2,2)=CFZ*SUBSCL*ETA(J)
      UOUT=OTSCLX*ETA(J)**(1./ENX)
```

```
      WOUT=OTSCLZ*ETA(J)**(1./ENZ)
      IF(U(J,2,2).GT.UOUT) U(J,2,2)=UOUT
   20 IF(ABS(W(J,2,2)).GT.ABS(WOUT)) W(J,2,2)=WOUT
   30 WSCALE=1./(URUE*SINTH)
      DO 40 J=2,NP
      IF(YPLUS(J).GE.DELTAP) GO TO 35
      U(J,2,2)=U(J,2,2)-W(J,2,2)*COTTH
      W(J,2,2)=WSCALE*W(J,2,2)
      GO TO 40
   35 U(J,2,2)=1.
      W(J,2,2)=WE(2,2)/UREF1
   40 CONTINUE
      V(1)=V(1)-T(1)*COTTH
      T(1)=WSCALE*T(1)
C  STREAMFUNCTION COMPONENTS
      F(1,2,2)=0.
      G(1,2,2)=0.
      CF1=0.
      CG1=0.
      DO 50 J=2,NP
      CF=U(J,2,2)
      CG=W(J,2,2)
      F(J,2,2)=F(J-1,2,2)+.5*(CF+CF1)*DETA(J-1)
      G(J,2,2)=G(J-1,2,2)+.5*(CG+CG1)*DETA(J-1)
      CF1=CF
   50 CG1=CG
C  TURBULENT MIXING ENERGY
      ESCALE=ALAMM*UTAU2/SALFAS
      EMATCH=ESCALE*(COS(5.*PI/DELTAP))**2
      DO 70 J=1,NP
      IF(YPLUS(J).GT.10.) GO TO 65
      E(J,2,2)=EMATCH*(.1*YPLUS(J))**4
      GO TO 70
   65 IF(YPLUS(J).GE.DELTAP) GO TO 68
      E(J,2,2)=ESCALE*(COS(.5*PI*YPLUS(J)/DELTAP))**2
      IF(E(J,2,2).GT.ZIOTAE) GO TO 70
   68 E(J,2,2)=ZIOTAE
   70 CONTINUE
C  TURBULENT DISSIPATION RATE
      YPLSUB=20.*SALFAS*AKAPPA/SBETA
      WSUB=(20./SBETA)**2
      WWAL=(ALAMM*YSCALP/(SALFAS*AKAPPA))**2
      WWAK=(PS1(1,2)*URUE/(ZIOTAL*DELTA))**2/SALFAS
      DO 80 J=2,NP
      IF(YPLUS(J).GT.YPLSUB) GO TO 75
      WT2(J,2,2)=WSUB/ETA(J)**4
      GO TO 80
   75 WTEST1=WWAL/ETA(J)**2
      WTEST2=WWAK*E(J,2,2)
      IF(WTEST1.LT.WTEST2.OR.YPLUS(J).GE.DELTAP) GO TO 78
      WT2(J,2,2)=WTEST1
      GO TO 80
   78 WT2(J,2,2)=WTEST2
   80 WT(J)=SQRT(WT2(J,2,2))
      WTEDG=WT(NP)
      WT(1)=WT(2)
      WT2(1,2,2)=WT(2)**2
      RETURN
      END
```

```
      SUBROUTINE STORIT(IUNIT,A,NDIM,IPNT,N)
C     ************************************************************
C     THIS SUBROUTINE COORDINATES TEMPORARY STORAGE AND RETREIVAL
C     OF DATA FROM DISK.  DATA ARE STORED AS A SERIES OF BLOCK
C     DATA IN THE A ARRAY WHOSE DIMENSION IS NDIM.  DURING A READ
C     FROM THE SUBROUTINE (N=1), THE FIRST (IPNT-1) BLOCKS ARE
C     SKIPPED AND THE A ARRAY CORRESPONDING TO IPNT IS READ FROM
C     DISK.  DATA ARE WRITTEN ON DISK WHEN N=0.
C     ************************************************************
      INTEGER TAPEIN,TAPEOT,TAPEGP,TAPEPF,TAPEDT,TAPEVL
      DIMENSION  A(620)
      COMMON/TAPE/ TAPEIN,TAPEOT,TAPEGP,TAPEPF,TAPEDT,TAPEVL
C     ******************
C     WRITE DATA ON DISK
C     ******************
C
      IF(N.NE.0) GO TO 10
C
C*****IBM OPTION*****
C     WRITE(IUNIT,IPNT)  A
C*****CDC 7600 OPTION*****
C     CALL WRITMS(IUNIT,A,NDIM,IPNT)
C*****UNIVAC 1108 OPTION*****
      INDEX=NDIM*(IPNT-1)
      CALL NTRAN(IUNIT,10)
      CALL NTRAN(IUNIT,6,INDEX)
      CALL NTRAN(IUNIT,1,NDIM,A,LFLAG)
 1    IF(LFLAG.EQ.-1) GO TO 1
      IF(LFLAG.LT.-1) GO TO 20
      RETURN
C
C     ******************
C     READ DATA FROM DISK
C     ******************
C
 10   CONTINUE
C
C*****IBM OPTION*****
C     READ (IUNIT,IPNT)  A
C*****CDC 7600 OPTION*****
C     CALL READMS(IUNIT,A,NDIM,IPNT)
C*****UNIVAC 1108 OPTION*****
      INDEX=NDIM*(IPNT-1)
      CALL NTRAN(IUNIT,10)
      CALL NTRAN(IUNIT,6,INDEX)
      CALL NTRAN(IUNIT,2,NDIM,A,LFLAG)
 11   IF(LFLAG.EQ.-1) GO TO 11
      IF(LFLAG.LT.-1) GO TO 20
      RETURN
C
 20   IF(N.EQ.0) WRITE(TAPEOT,5)
      IF(N.EQ.1) WRITE(TAPEOT,6)
      IF(LFLAG.EQ.-2) WRITE(TAPEOT,2) LFLAG
      IF(LFLAG.EQ.-3) WRITE(TAPEOT,3) LFLAG
      IF(LFLAG.EQ.-4) WRITE(TAPEOT,4) LFLAG
      STOP
C     ************************************************************
 2    FORMAT(1H ,1X7HLFLAG =,I3,5X23HEND OF FILE ENCOUNTERED)
 3    FORMAT(1H ,1X7HLFLAG =,I3,5X12HDEVICE ERROR)
 4    FORMAT(1H ,1X7HLFLAG =,I3,5X20HTRANSMISSION ABORTED)
```

```
5     FORMAT(1H1,29HWRITE ERROR DETECTED BY NTRAN)
6     FORMAT(1H1,28HREAD ERROR DETECTED BY NTRAN)
C   *********************************************************
      END
```

```
10 REM ****************************************************
20 REM *******   INTERPOLATION ROUTINE TO ACCOUNT    *******
30 REM *******            FOR EFFECTS OF             *******
40 REM *******       DISPLACEMENT THICKNESS          *******
50 REM ****************************************************
60 DEFUSR1=&H7D00
70 CLEAR 1000
80 DIM X(41),Z(21),DISP(41,21)
90 DIM ZB(41,21)
100 DIM XR(41),YR(41,11),ZR(11),YN(41,11)
110 OPEN "I",1,"DISPL"
120 INPUT "NXTL =";NX
130 INPUT "NZTL =";NZ
140 FOR I=1 TO NX
150 FOR J=1 TO NZ
160 INPUT#1, DISP(I,J)
170 NEXT J
180 NEXT I
190 CLOSE
200 INPUT "FILE NAME FOR NONORTHOGONAL MESH";F$
210 OPEN "I",1,F$
220 FOR J=1 TO NZ
230 INPUT#1, H$,Z(J)
240 FOR I=1 TO NX
250 INPUT#1, D,X(I),D,ZB(I,J),D,D,D,D,D,D,D,D
260 NEXT I
270 NEXT J
280 CLOSE
290 OPEN "I",1,"MESHPR"
300 INPUT "IMAX =";IM
310 INPUT "JMAX =";JM
320 FOR I=1 TO IM
330 INPUT#1, XR(I)
340 NEXT I
350 FOR J=1 TO JM
360 INPUT#1, ZR(J)
370 NEXT J
380 FOR I=1 TO IM
390 FOR J=1 TO JM
400 INPUT#1, YR(I,J)
410 NEXT J
420 NEXT I
430 CLOSE
440 NAME "MESHPR" AS "MESHPR/OLD"
450 YN=USR1(X,Z,DISP,ZB,XR,YR,ZR,NX,NZ,IM,JM)
460 OPEN "O",1,"MESHPR"
470 FOR I=1 TO IM
480 PRINT#1 USING "##.#########↑↑↑↑",XR(I)
490 NEXT I
500 FOR J=1 TO JM
510 PRINT#1 USING "##.#########↑↑↑↑",ZR(J)
520 NEXT J
530 FOR I=1 TO IM
540 FOR J=1 TO JM
550 PRINT#1 USING "##.#########↑↑↑↑",YN(I,J)
560 NEXT J
570 NEXT I
580 CLOSE
590 END
```

2-8

DT