NASA
CR
1926
pt.2
c.1

TECH LIBRARY KAFB, NM
0061900

NASA CONTRACTOR REPORT

NASA CR-1927

LOAN COPY: RETURN TO
AFWL (DOUL)
KIRTLAND AFB, N. M.

NASA CR-1927

FLOW FIELD ANALYSIS OF AIRCRAFT CONFIGURATIONS USING A NUMERICAL SOLUTION TO THE THREE-DIMENSIONAL UNIFIED SUPERSONIC/HYPERSONIC SMALL-DISTURBANCE EQUATIONS

Part II

by E. D'Sylva

Prepared by
THE BOEING COMPANY
Seattle, Wash.
for Langley Research Center

NATIONAL AERONAUTICS AND SPACE ADMINISTRATION • WASHINGTON, D. C. • FEBRUARY 1972

TECH LIBRARY KAFB, NM

0061016

1. Report No.	2. Government Accession No.	3. Recipient's Catalog No.
NASA CR-1927		

4. Title and Subtitle	5. Report Date
FLOW FIELD ANALYSIS OF AIRCRAFT CONFIGURATIONS USING A NUMERICAL SOLUTION TO THE THREE-DIMENSIONAL UNIFIED SUPERSONIC/HYPERSONIC SMALL-DISTURBANCE EQUATIONS. PART II	February 1972
	6. Performing Organization Code
7. Author(s)	8. Performing Organization Report No.
E. D' Sylva	D6-25124, Part II
	10. Work Unit No.
9. Performing Organization Name and Address	
The Boeing Company Commercial Airplane Group Seattle, Washington	11. Contract or Grant No. NAS1-9562
	13. Type of Report and Period Covered
12. Sponsoring Agency Name and Address	Contractor Report
National Aeronautics and Space Administration Washington, D.C. 20546	14. Sponsoring Agency Code

15. Supplementary Notes

16. Abstract

The unified small disturbance equations are numerically solved using the well-known Lax-Wendroff finite-difference technique. The method allows complete determination of the inviscid flow field and surface properties as long as the flow remains supersonic. Shock waves and other discontinuities are accounted for implicity in the numerical method. This technique has been programed for general application to the three-dimensional case. The validity of the method is demonstrated by calculations on (1) cones, (2) axisymmetric bodies, (3) lifting bodies, (4) delta wings, and (5) a conical wing/body combination. Part I contains the discussion of problem development and results of the study. Part II contains flow charts, subroutine descriptions, and a listing of the computer program.

1. Flow distribution
2. Aerodynamic configurations - Calculations
3. Finite difference techniques

18 Feb 72 wg

17. Key Words (Suggested by Author(s))		18. Distribution Statement
Flow fields	Inviscid flow	Unclassified - Unlimited
Finite difference	Steady flow	
Aircraft flows	Three-dimensional flow	
Flow analysis		

19. Security Classif. (of this report)	20. Security Classif. (of this page)	21. No. of Pages	22. Price*
Unclassified	Unclassified	300	$3.00

*For sale by the National Technical Information Service, Springfield, Virginia 22151

TABLE OF CONTENTS

		Page
1.0	INTRODUCTION	1
2.0	PROGRAM DESCRIPTION	2
3.0	PROGRAM DESIGN CONSIDERATIONS	4
4.0	PROGRAM FLOW CHARTS	6
5.0	DESCRIPTION OF SUBPROGRAMS	18
	5.1 Overlay (0,0) The Main Overlay	22
	5.2 Primary Overlay (1,0)	38
	5.3 Primary Overlay (2,0)	51
	5.4 Primary Overlay (3,0)	55
	5.5 Primary Overlay (33,0)	68
	5.6 Primary Overlay (4,0)	78
	5.7 Primary Overlay (5,0)	82
	5.8 Primary Overlay (6,0)	91
	5.9 Primary Overlay (7,0)	95
	5.10 Primary Overlay (10,0)	103
6.0	FILE DESCRIPTION	107
7.0	SAMPLE INPUT AND OUTPUT DATA	112
8.0	PROGRAM LISTING	122

FLOW FIELD ANALYSIS OF AIRCRAFT CONFIGURATIONS USING A NUMERICAL SOLUTION TO THE THREE-DIMENSIONAL UNIFIED SUPERSONIC/ HYPERSONIC SMALL-DISTURBANCE EQUATIONS - PART II

E. D'Sylva, Boeing Computer Services

1.0 INTRODUCTION

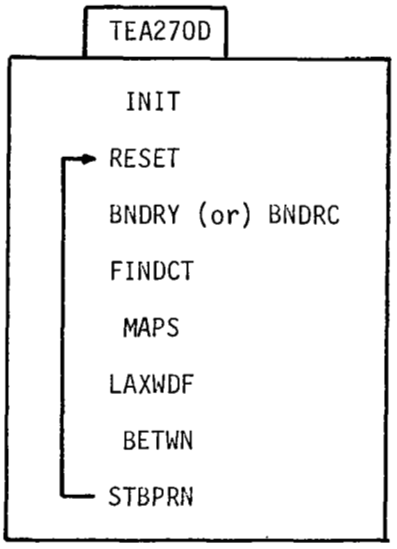
This part of the document (Part II) describes the digital computer program developed to implement a numerical solution to the unified supersonic/hypersonic small disturbance equations for flow about aircraft type configurations. In hypersonic small disturbance theory, the concept of equivalence of a three-dimensional steady flow to a two-dimensional unsteady flow allows the solution to be advanced successively from one cross-cut plane to an adjacent cross-cut plane in the free stream direction.

The numerical method used to update the flow field in a cross-cut plane is based on a Lax-Wendroff difference approximation to the flow equations with special treatment of points on and near the boundary contour. Boundary points are updated using either a quasi-one-dimensional method of characteristics (called BND3) or a conservative difference scheme derived from the Lax-Wendroff difference equations (called BND1). There are also field points near the boundary contour (called 'between' points) which are updated by interpolation. Details of the theoretical considerations underlying the method are presented in Part I of this document.

2.0 DESCRIPTION OF PROGRAM

The computer program is coded in CDC FØRTRAN 2.3 language for the Control Data Corporation 6400/6600 computers and the Scope 3.1 operating system.

To keep within the field constraint of 70,000 octal the program uses the overlay feature of the loader. The program is divided into ten subprograms consisting of a main overlay and nine primary overlays. Each overlay consists of a set of subroutines. The overlay structure is shown below:



TEA270D, the main overlay or overlay (0,0) calls the primary overlays. It contains basic subroutines used by several primary overlays.

INIT is primary overlay (1,0). It is the initialisation overlay and as its name suggests, it executes the tasks that need to be done once and for all at the commencement of a program run.

The next eight overlays from RESET to STBPRN are executed in a loop. Each cycle through the loop advances the flow field solution in the free stream direction.

RESET is primary overlay (2,0). It initialises the program within the time integration loop.

BNDRY (primary overlay (3,0)) or BNDRC (primary overlay (33,0)) updates the points on the boundary of the contour.

FINDCT (primary overlay (4,0)) interpolates for the cross-cut contours at the new time level.

MAPS, which is primary overlay (5,0) locates intersections of the Cartesian grid lines with the contours and generates maps of inside, 'between' and Lax-Wendroff points.

LAXWDF (primary overlay (6,0)) updates those points which can be updated by the Lax-Wendroff method.

BETWN (primary overlay (7,0)) controls the interpolation scheme for points near the boundary which are not Lax-Wendroff points.

STBPRN, which is primary overlay (10,0) completes the time step for the next update and prints out flow field data.

3.0 PROGRAM DESIGN CONSIDERATIONS

To achieve a computer program which would accept configurations requiring quite a large number of grid points for their analysis while still using only a limited amount of core, the program incorporates the following features.

A Use of overlays

B Extensive use of disc storage

C Partition of flow field into blocks

D Use of a single scratch array with dynamic storage allocation

The overlay feature is used to reduce the number of program instructions which reside in core at any one time, thereby freeing more core for the work area. The use of disc memory is essential to the storage of the vast amount of data required to represent the fluid state of all the grid points in the flow field. As all these data cannot be accommodated in core simultaneously, the flow field is partitioned into blocks. As the solution in the blocks are interdependent, sufficient overlap in information is provided. The partition of the flow field into blocks of optimum size is automatically executed by the program.

The key decision of the program design is the use of a single one-dimensioned scratch array and the avoidance of arrays with fixed dimensions. This scratch array is then used in the following way: The main program in each primary overlay is only used to call a number of subroutines which do the computation. But before each subroutine is called, the scratch array is cut up into various subsections and the addresses of these subsections are transferred to these subroutines.

The subroutines themselves have various arrays which may be multiply-dimensioned, but have adjustable dimensions. The values of these adjustable dimensions and the addresses of these arrays are brought in through the parameter list. In this way, dynamic storage allocation of the work area A is achieved.

The beneficial results of these design features are

(a) Arrays with fixed dimensions are wasteful of core, except when the biggest cases are being run. There are no such arrays in the program. Instead, the single scratch A is automatically partitioned into subsections in an optimum manner on every run. No alteration in dimension or other statements needs to be made from run to run.

(b) If the core constraint needs to be changed, it can be achieved by a single card change in the program. The card that needs to be changed is in the main overlay, namely,

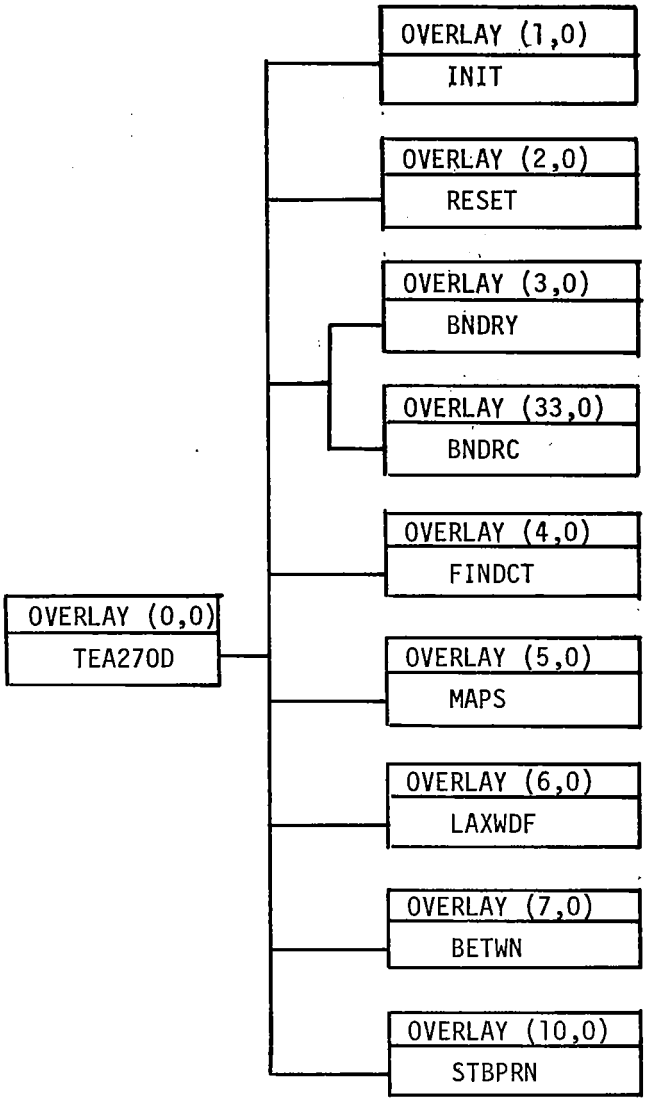
```
CØMMØN/CM999/NN,A(9000)  $NN = 9000.
```

For example, the length of the scratch array A could be changed to 20,000 by replacing the above card by

```
CØMMØN/CM999/NN,A(20000)  $NN = 20000
```

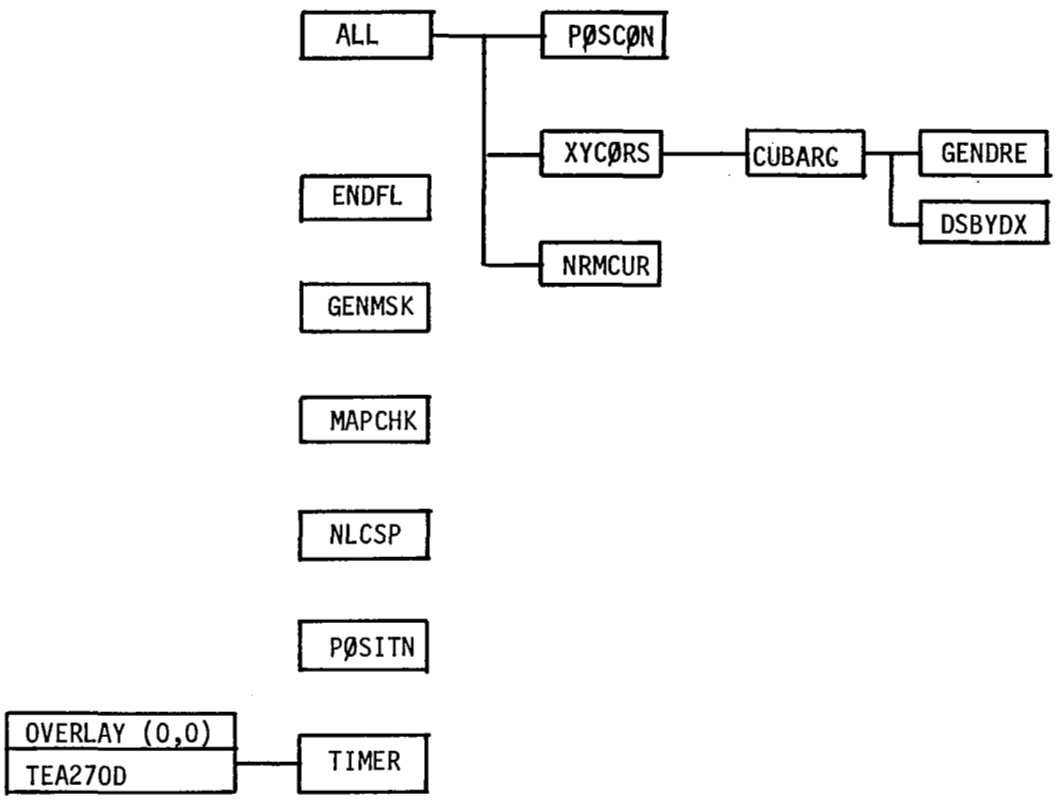
4.0 PROGRAM FLOW CHARTS

The program flow charts are listed in this section. Figure 1 is an overall flow chart of the overlays. Figures 2 to 11 are flow charts of the individual overlays.



There are no secondary overlays in the program.

FIGURE 1: MAIN AND PRIMARY OVERLAYS



Only TIMER is called by TEA27OD. The other subroutines are used by the primary overlays. All are retained in core.

FIGURE 2: SUBROUTINES IN MAIN OVERLAY

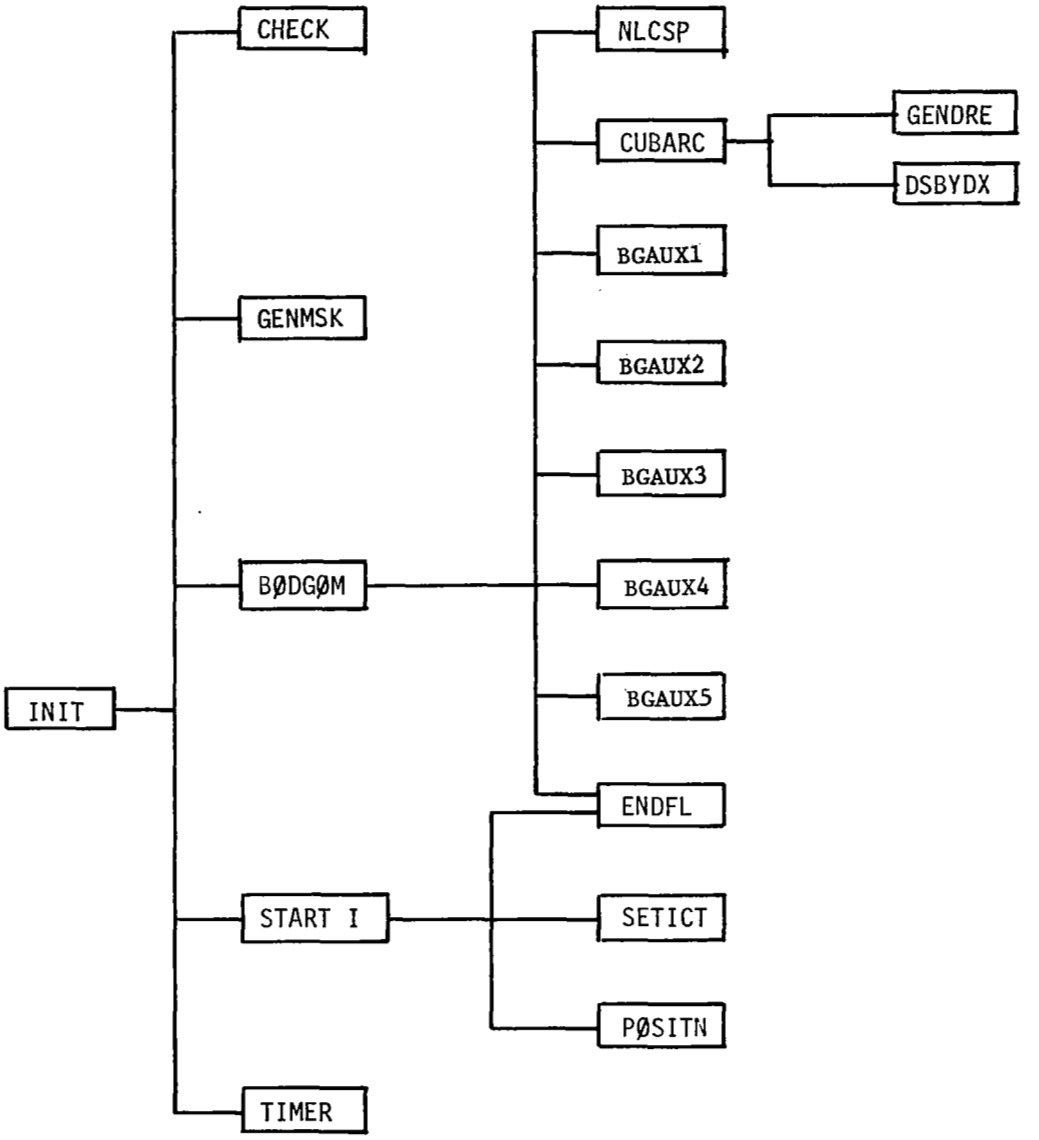


FIGURE 3: INIT PRIMARY OVERLAY (1,0)

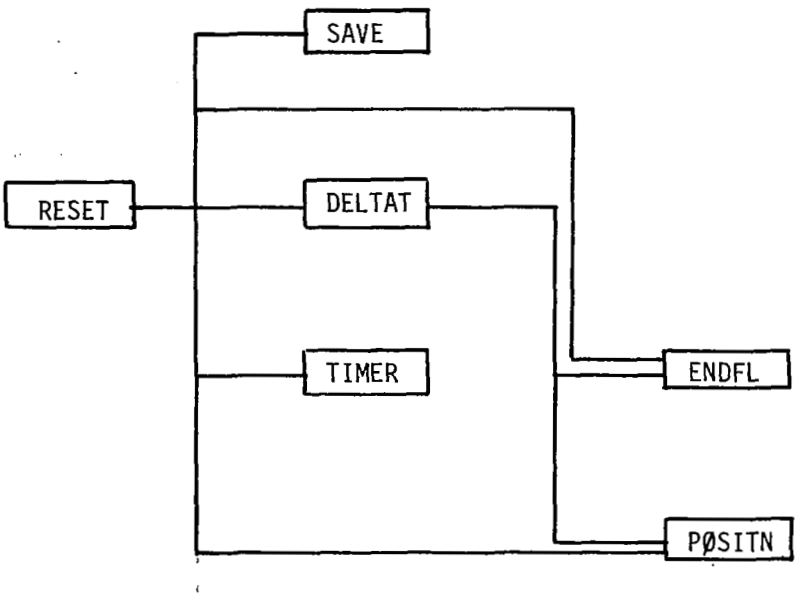


FIGURE 4: SUBROUTINES IN PRIMARY OVERLAY (2,0)

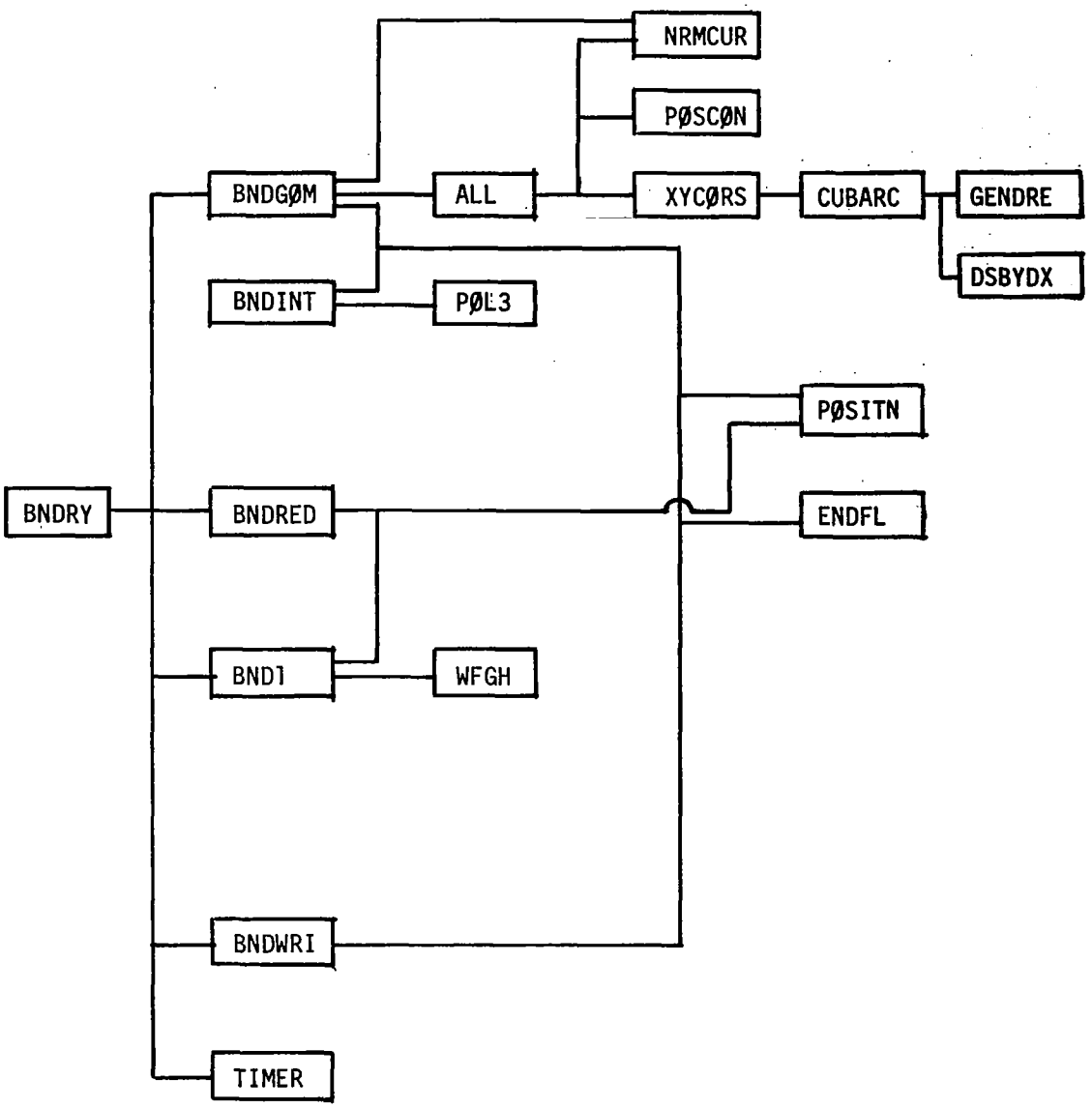


FIGURE 5: SUBROUTINES IN PRIMARY OVERLAY (3,0)

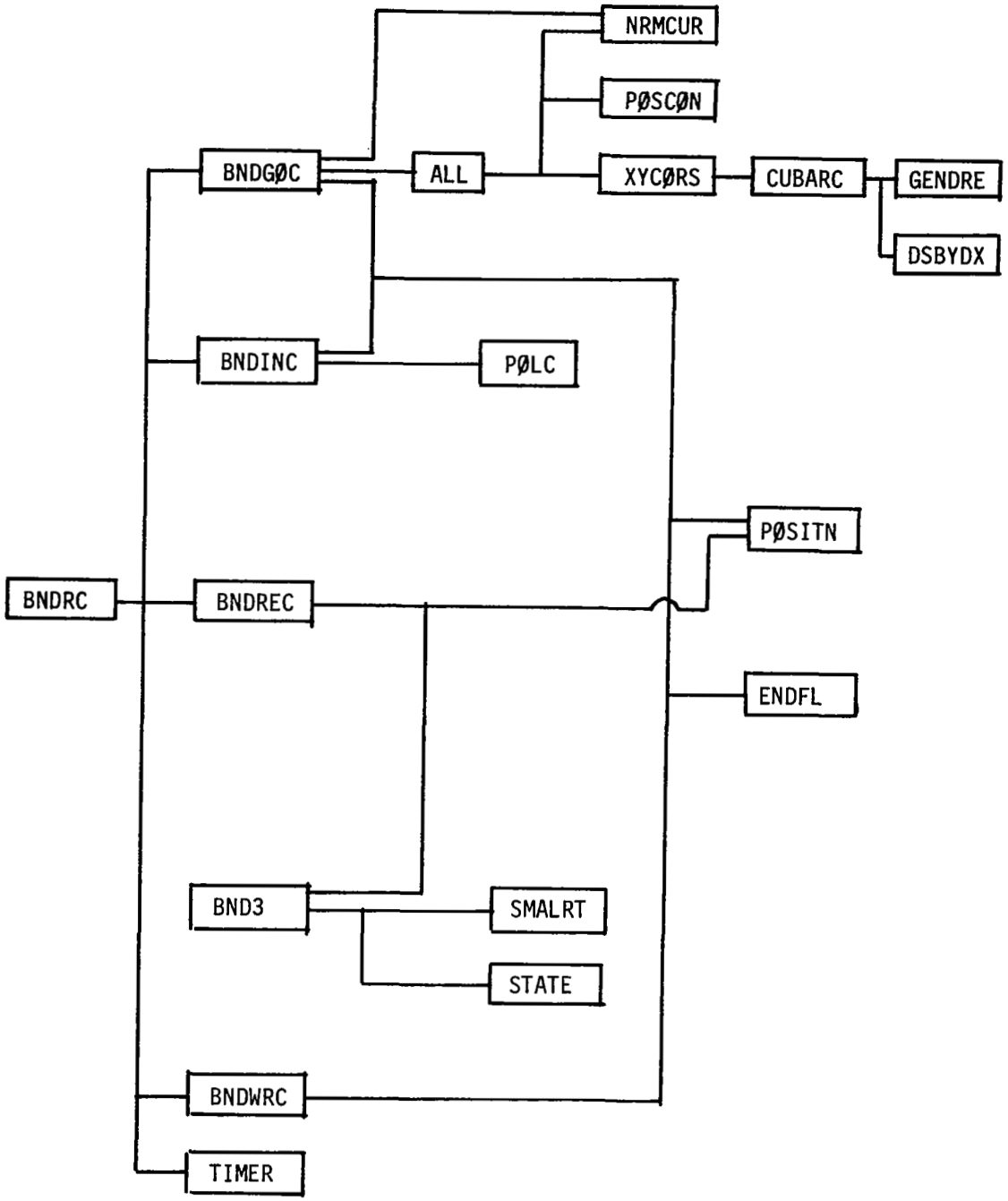


FIGURE 6: SUBROUTINES IN PRIMARY OVERLAY (33,0)

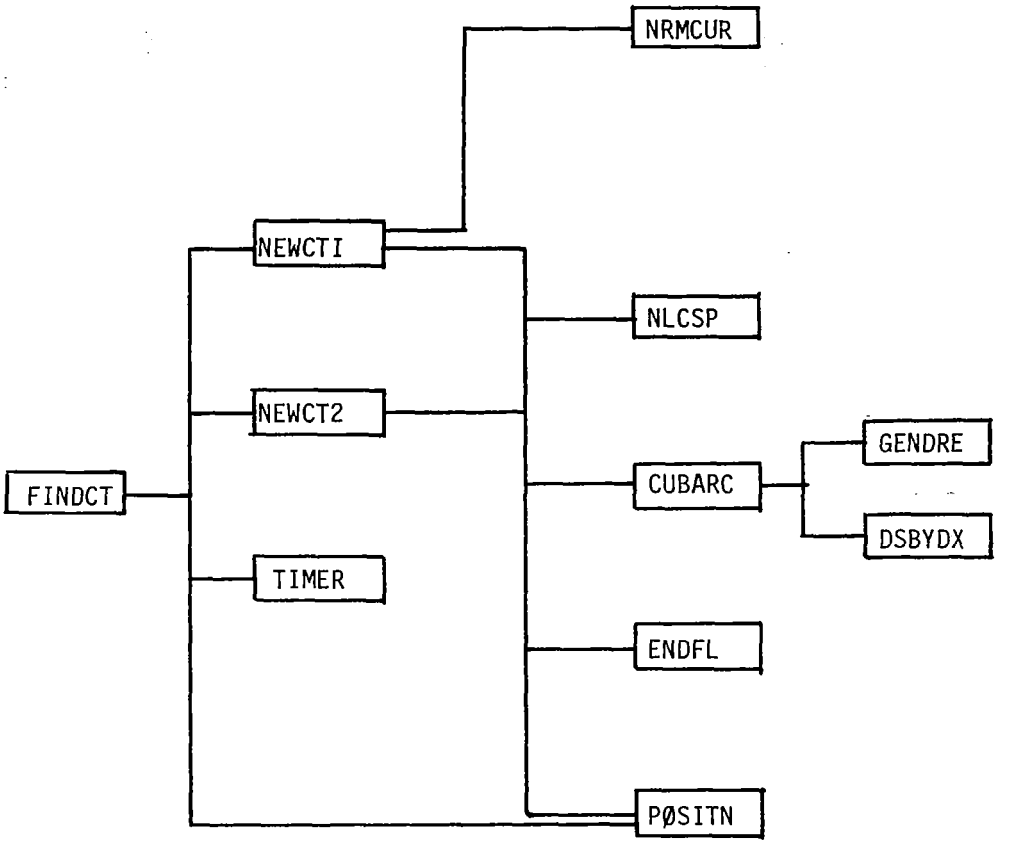


FIGURE 7: SUBROUTINES IN PRIMARY OVERLAY (4,0)

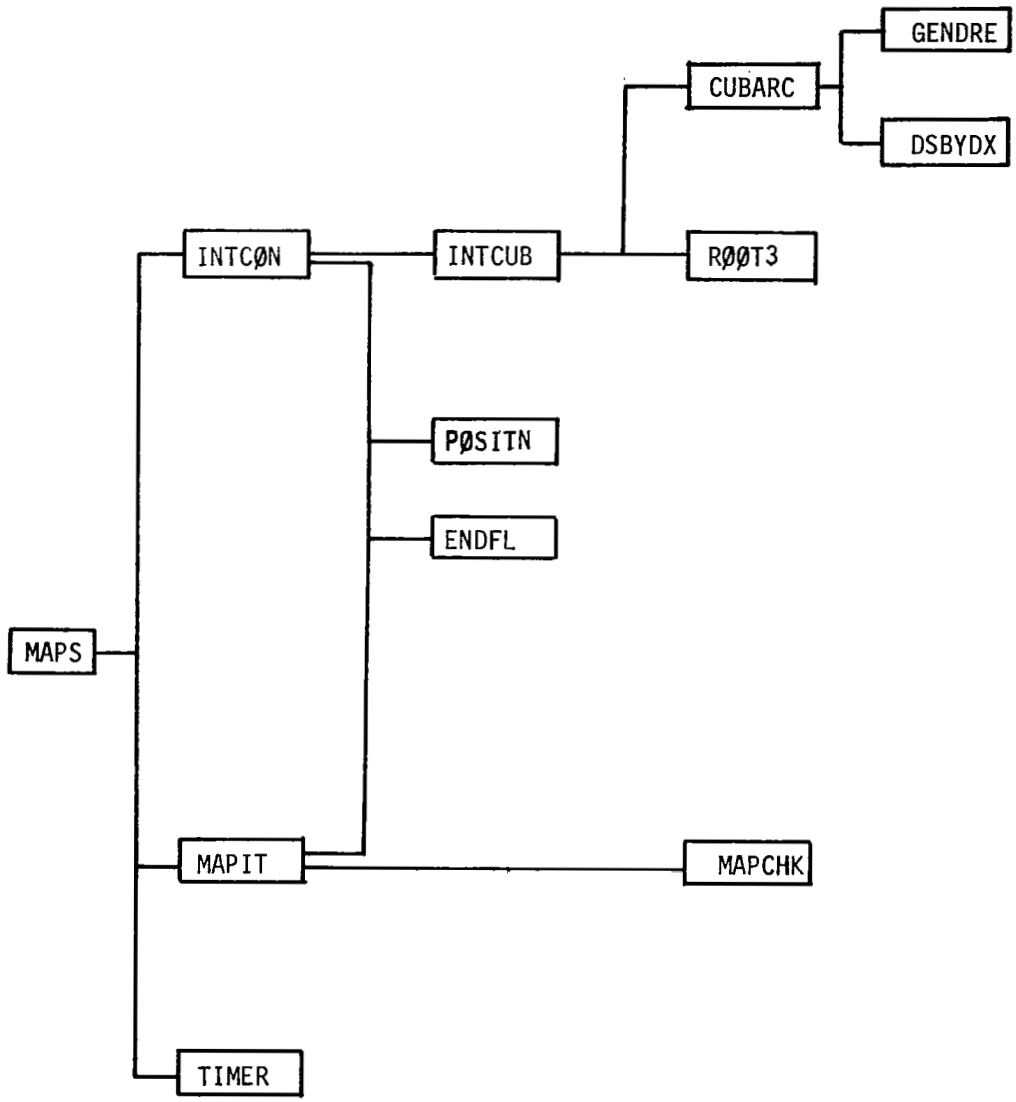


FIGURE 8: SUBROUTINES IN PRIMARY OVERLAY (5,0)

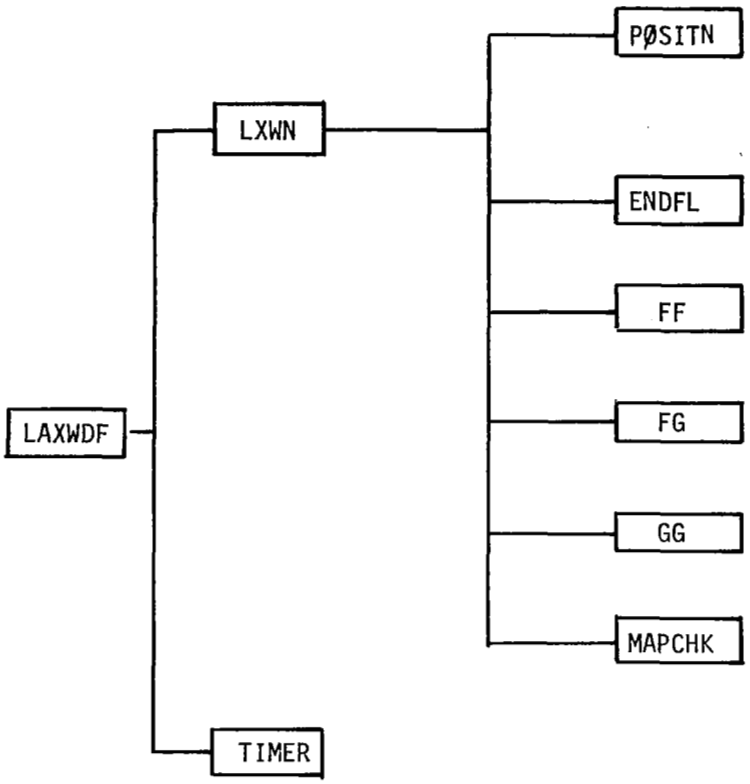


FIGURE 9: SUBROUTINE IN PRIMARY OVERLAY (6,0)

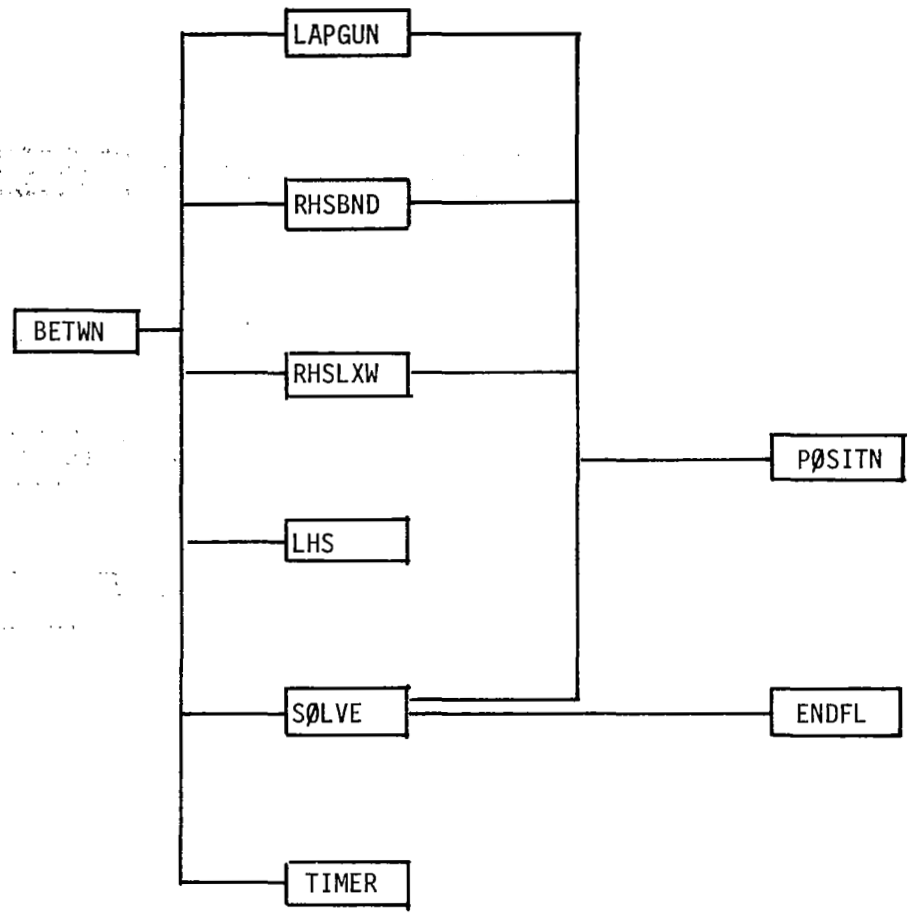


FIGURE 10: SUBROUTINES IN PRIMARY OVERLAY (7,0)

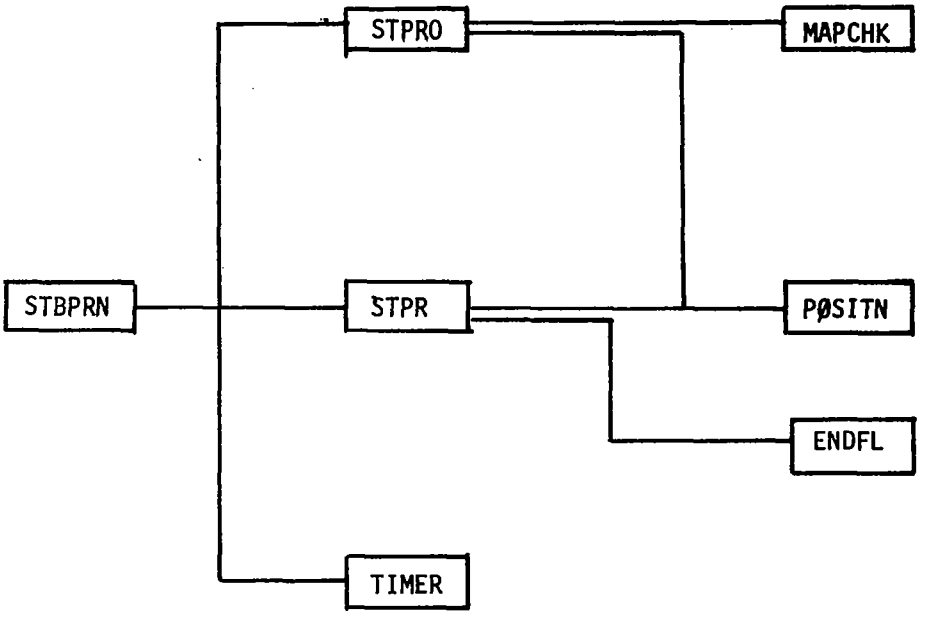


FIGURE 11: SUBROUTINE IN OVERLAY (10,0)

5.0 DESCRIPTION OF SUBPROGRAMS

Below is an index of subprogram descriptions.

5.1 Overlay (0,0) The Main Overlay

	Field Length (Octal)
TEA270D	Variable
ALL	171
CUBARC	108
DSBYDX	36
ENDFL	43
GENDRE	248
GENMSK	37
MAPCHK	45
NLCSP	1,053
NRMCUR	74
PØSCØN	108
PØSITN	124
TIMER	77
XYCØRS	255

5.2 Primary Overlay (1,0)

	Field Length (Octal)
INIT	1,668
CHECK	766
CØNESØL	272
TANWEDG	435
PATCH	558
SETICT	106
START1	133
START2	1,793

	BØDGOM	Variable
	BGAUX1	Variable
	BGAUX2	Variable
	BGAUX3	Variable
	BGAUX4	Variable
	BGAUX5	Variable

5.3 Primary Overlay (2,0)

	Field Length (Octal)
RESET	626
DELTAT	132
SAVE	501

5.4 Primary Overlay (3,0)

	Field Length (Octal)
BNDRY	500
BNDGØM	1,243
BNDINT	1,420
BNDRED	352
BND1	3,224
BNDWRI	1,143
PØL3	176
WFGH	231
BWTØW	104

5.5 Primary Overlay (33,0)

	Field Length (Octal)
BNDRC	517
BNDGØC	1,351
BNDINC	1,627

BNDREC	. . . . .	363
BND3	. . . . .	3,547
BNDWRC	. . . . .	134
PØLC	. . . . .	176
STATE	. . . . .	150
SMALRT	. . . . .	164

5.6 Primary Overlay (4,0) Field Length (Octal)

FINDCT	. . . . .	551
NEWCT1	. . . . .	1,535
NEWCT2	. . . . .	1,161

5.7 Primary Overlay (5,0) Field Length (Octal)

MAPS	. . . . .	263
INTCØN	. . . . .	1,572
INTCUB	. . . . .	337
MAPIT	. . . . .	1,174
RØØT3	. . . . .	501

5.8 Primary Overlay (6,0) Field Length (Octal)

LAXWDF	. . . . .	206
LXWN	. . . . .	3,462
FF	. . . . .	60
FG	. . . . .	66
GG	. . . . .	60

5.9 Primary Overlay (7,0)

	Field Length (Octal)
BETWN	234
LAPGUN	1,113
RHSBND	220
RHSLXW	737
LHS	363
SØLVE	334

5.10 Primary Overlay (10,0)

	Field Length (Octal)
STBPRN	150
STPR	1,517
STPRO	1,111

5.1 Overlay (0,0) Main Overlay

There is only one subprogram in this overlay namely TEA270D, which is described below.

SUBJECT: FØRTRAN IV program TEA270D

PURPOSE: To control the calling of primary overlays and provide communication among them.

METHØD: TEA270D calls the primary overlays. It contains a set of labeled common statements which contain information which is always in core and is available to all primary overlays.

USAGE: PROGRAM TEA270D (INPUT = 1002, ØUTPUT, TAPE5 = INPUT, TAPE6 = ØUTPUT, TAPE1 = 1002, TAPE2, TAPE3, TAPE4, TAPE7, TAPE10)

Input: TAPE1 = file on which geometry data for input contour is stored

TAPE2, TAPE3, TAPE4, TAPE7 = scratch files

TAPE10 = file used to save data for a restart.

SUBPRØGRAMS: TIMER

SUBJECT: FORTRAN IV subroutine ALL

PURPOSE: To compact the computation performed by subroutines PØSCØN, XYCØRS, and NRMCUR into one call statement.

METHOD: See description of subroutines PØSCØN, XYCØRS, NRMCUR.

SUBPROGRAMS: PØSCØN

XYCØRS

NRMCUR

SUBJECT: FØRTRAN IV subroutine CUBARC

PURPOSE: To compute the arc length of a cubic.

$y = C_1 X + C_2 X^2 + C_3 X^3$ from X = A to X = B

METHOD: The arc length is computed by supplying the numerical integrator GENDRE with the integrand

SQRT $(1 + (C1 + 2C_2 X + 3C_3 X^2)^2)$

SUBPROGRAMS CALLED:

DSBYDX

GENDRE

USAGE: CALL CUBARC (A,B,C1,C2,C3,EPS, ARC)

Input

A = lower limit of integration

B = upper limit of integration

C1, C2, C3 = coefficients of cubic

EPS = absolute tolerance arc length must meet

Output

ARC = the arc length from A to B of the given cubic

ERROR RETURNS: If the tolerance is not met, the parameter list is printed out, but processing continues.

SUBJECT: FØRTRAN IV subroutine DSBYDX

PURPOSE: Finds ds/dx for cubic $y = C_1 + C_2 X^2 + C_3 X^3$

METHOD: $ds/dx = SQRT (1 + (C_1 + 2C_2 X + 3C_3 X^2)^2)$

USAGE: CALL DSBYDX (X, FX)

Input X = value at which ds/dx is required

Output FX = ds/dx at X

SUBJECT: FØRTRAN IV subroutine ENDFL.

PURPOSE: To write an end of file mark and keep track of the present file position on the system file.

METHOD: An end of file mark is written on system file IØUNIT and the file position index IØFP is increased by one.

USAGE: CALL ENDFL (IØFP, IØUNIT)

Input IØFP = file position before ENDFL is called.

IØUNIT = system file which is being operated on.

Output IØFP = file position after ENDFL is called.

ERRØR RETURNS: If IØFP is not a positive integer, an error message is written and exit called.

SUBJECT: FØRTRAN IV subroutine GENDRE

PURPOSE: To evaluate an integral numerically using Gauss-Legendre quadrature.

METHOD: The integral over the whole range I1 and the integrals over the upper and lower half ranges (I2, I3) are computed using an 8 point Gauss-Legendre quadrature. If (I1-I2-I3) exceeds a given tolerance, the procedure is repeated with the 16 point formula. If the tolerance is still not satisfied, the error flag is set.

USAGE: CALL GENDRE (AUX, A, B, CØNTR, KØDE, FINTL, IRRØR)

Input AUX = name of auxiliary subroutine to evaluate the function at the required points.

A = lower limit of integral

B = upper limit of integral

CØNTR = tolerance to which integral is evaluated

KØDE = 0 if tolerance is absolute

= 1 if tolerance is relative

Output FINTL = value of integral

IRRØR = error code. See error returns.

SUBPROGRAMS: AUX (X, FX) returns function value FX at X

ERROR RETURNS: IRRØR = 1 if tolerance not met

= 0 if tolerance is met

SUBJECT: FØRTRAN IV subroutine GENMSK

PURPOSE: To generate bit patterns which are needed in overlay (5,0).

METHOD: The arrays MSK and MSKRGT each of length sixty are generated in this subroutine. MSK (I) has one in bit position I, zero elsewhere. MSKRGI (I) has one in bit positions up to and including I, zero elsewhere.

USAGE: CALL GENMSK

SUBJECT: FØRTRAN IV function MAPCHK

PURPOSE: To check if a particular bit in a given word has the value 0 or 1.

METHOD: This is achieved by using the masks MSK (I) which are generated in subroutine GENMSK.

USAGE: FUNCTIØN MAPCHK (MAP, KW, NX, J, I)

Input MAP is an array dimensioned MAP(KW,NX). It may be regarded as consisting of NX columns of bits, each column containing 60*KW bits.

Output If the J^{th} bit of the I^{th} column is zero MAPCHK is set to zero, otherwise it is non-zero.

SUBJECT: FØRTRAN IV subroutine NLCSP

PURPOSE: To curve fit a set of points with a curve which is continuous in slope and curvature. This curve consists of a chain of cubics with respect to local axis.

METHOD: Either the slope or the curvature at each of the two end points must be specified. The slope at each intermediate point is approximated by passing a circle through three adjacent points. With this slope estimate, the cubic in each segment is uniquely determined. As the cubics in general will not match in curvature at their end points, the slopes are perturbed using a Newton-Raphson type process to minimize the discrepancy in curvatures at the intermediate points until a specified tolerance is satisfied.

USAGE: CALL NLCSP (X, Y, N, N1, SS1, N2, SS2, TØL, D, CS,
SN, A1, A2, A3, A4, A5)

Input X = array of X coordinates of points
Y = array of Y coordinates of points
N = number of points
N1 = 0 of slope specified at first point
= 1 if curvature specified at first point
N2 = 0 if slope specified at last point
= 1 if curvature specified at last point
If N1 = 0, SS1 = angle in radians made by tangent at first point with the reference axis
= 1 SS1 = curvature at first point

If N2 = 0, SS2 = angle in radians made by tangent at last point with the reference axis

= 1, SS2 = curvature at last point

TØL = tolerance to which curvatures should match at intermediate points

A4, A5 scratch arrays of length N

Output

D = array of distances between points

CS = array of cosines of angles made by the line segments joining the points, and the X axis

SN = array of sines of angles made by the line segments joining the points, and the X axis

A1, A2, A3 arrays of the coefficients of X, X^2 and X^3 of the local cubics

ERROR RETURNS: If the curvature discrepancy tolerance is not met within (20 + N/2) iterations, exit is called after diagnostic printout.

RESTRICTIONS: The angle between two adjacent line segments must not exceed thirty degrees approximately.

SUBJECT: FØRTRAN IV subroutine NRMCUR

PURPOSE: To compute the normal direction and the curvature of a cubic at a point. The direction cosines of the normal are then transformed to principal axes.

SUBJECT: FØRTRAN IV subroutine PØSCØN

PURPOSE: To find the position of a point on a contour, given its arc length from the contour starting point.

METHOD: Since the arc length of each contour point is known, it can easily be determined between which two contour points the given point lies.

USAGE: CALL PØSCØN (N, S, ARC, J, SS, S1, S2, SGN)

Input N = number of points on the contour

S = array of arc lengths of the input contour points

ARC = arc length of required point

Output J the required point lies between input contour points J and (J + 1)

SS = arc distance of required point from contour point J

S1 = fractional arc distance of required point from point J

S2 = fractional arc distance of required point from point (J + 1)

SGN = +1 if required point is on contour

= -1 if required point is on reflected half of contour

SUBJECT: FØRTRAN IV subroutine PØSITN

PURPOSE: To position the pointer on a given system file at a given logical file.

METHOD: After determining the quickest way to get to the desired position, forward space file (FSF), backspace file (BSF) and REWIND are used to get there.

USAGE: CALL PØSITN (IFIL1, IFIL2, IØUNIT, IND)

Input IFIL1 = logical file in which system file is positioned

IFIL2 = logical file at the start of which pointer is to be positioned

IØUNIT = system file to be operated on

IND = is a reference number to identify the call if it is a failure. Also, if IND is positive the use of BSF is suppressed.

ERROR RETURNS: If the positioning cannot be achieved, the parameter list is printed out and exit called.

RESTRICTIONS: IFIL2 must be greater than zero.

SUBJECT: FØRTRAN IV subroutine TIMER

PURPOSE: Determines the CP time since program load.

METHOD: The CP time is obtained by calling the system routine SECØND.

USAGE: CALL TIMER (TMR, TIMA, TIMB, TIM)

Input

TMR = 1 for CP time

TIMA = Input CP reference time in secs

Output

TIMB = elapsed CP time in seconds since load

TIM = elapsed CP time since TIMA

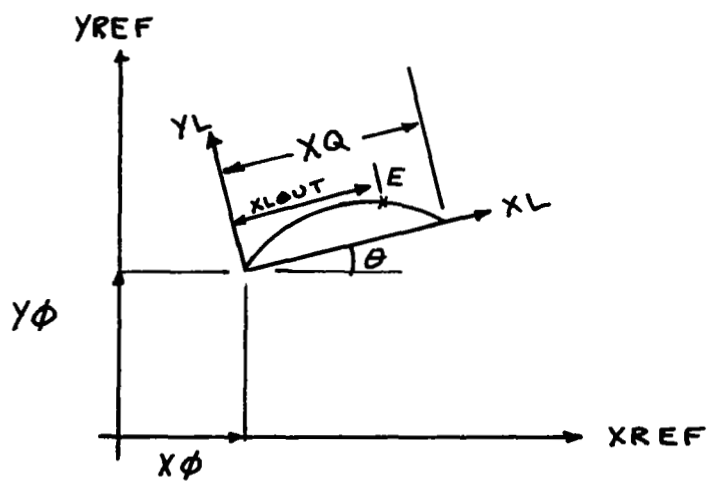
SUBPROGRAMS: SECØND

SUBJECT: FØRTRAN IV subroutine XYCØRS

PURPOSE: To compute the position of a point on a cubic corresponding to a given arc length.

METHOD: Given the cubic YL = C1* XL + C2* XL**2 + C3* XL**3 in the range (XP, XQ) it is required to find XLØUT such that S(XLØUT) = ARCIN where S(XL) is the arc length of the curve from XP to XL. The subroutine CUBARC is used to compute S(XL) and XLØUT is computed by the method of false position.

USAGE: CALL XYCØRS (XP, XQ, C1, X2, C3, IND, TØLS, TØLX, ARCIN, XLØUT, XØ, YØ, CS, SN, XREF, YREF)



Input XP = starting point of range of cubic in local axes XP

XQ = end point of range of cubic in local axes

C1, C2, C3 = coefficients of cubic

YL = C1* XL + C2* XL**2 + C3* XL**3

where XL, YL are in local axes

IND If IND=1, the arc length from XP to XLØUT is equal to ARCIN

If IND=2, the arc length from XLØUT to XQ is equal to ARCIN

TØLS TØLS *(XQ - XP) is the tolerance to which an arc length is computed by CUBARC

TØLX TOLX *(XQ - XP) is the tolerance to which XLØUT is computed

XØ X coordinate of starting point of cubic in reference axes

YØ Y coordinate of starting point of cubic in reference axes

CS CØSINE (Ø)

SN SIN (Ø)

Output

XLØUT local X coordinate of required point E for which S(XLØUT) = ARCIN if IND = 1 or S(XQ) - S(XLØUT) = ARCIN if IND = 2

XREF = X coordinate of point C with respect to reference axes

YREF = Y coordinate of point C with respect to reference axes

5.2 Primary Overlay (1,0)

Tasks which need to be done once only at the start of a program run are performed in this overlay. The program in the overlay is INIT which does some of the chores itself and the rest by calling subroutine CHECK, BØDGØM and START1 or START2.

SUBJECT: FØRTRAN IV program INIT

PURPOSE: To initialise a program run.

METHOD: Initialisation tasks are performed in INIT. Data are read in and printed out. Much used constants are computed once and for all. System files are initialised. CHECK is called to ensure that the work area provided is sufficient. BØDGØM is called to set up configuration geometry data on file ICT. START1 or START2 is called to begin the computations.

USAGE: PRØGRAM INIT

SUBPROGRAMS: CHECK
BØDGØM
START1
START2

SUBJECT: FØRTRAN IV subroutine CHECK

PURPOSE: Checks that the work area is sufficient to process the current case.

METHOD: The minimum size of the work area needed and the manner in which it is cut up is a function of NCTMX, NBTMX, NX, NY, KX and KY.

NCTMX = maximum number of points defining a contour
NBTMX = maximum number of 'between' points
NX = the number of vertical mesh lines
NY = the number of horizontal mesh lines
KX = maximum number of intersections of a vertical mesh line with a contour
KY = maximum number of intersections of a horizontal mesh line with a contour

The state variables for all the mesh points for the entire flow field cannot fit in core simultaneously. Therefore, the flow field is divided into subareas or equivalently the flow field mesh points are partitioned into blocks

A print out of the work area used by the various subroutines is produced so that the critical subroutines can be spotted.

USAGE: CALL CHECK

ERROR RETURNS: If the work area is too small to run the current case, a message is written and exit called.

SUBJECT: FØRTRAN IV subroutine CØNESØL

PURPOSE: To compute the initial flow field using the tangent cone method.

METHOD: The flow field properties around a cone of given semi-angle and zero angle of attack are read in by the subroutine. From these data W1, W2, W3, W4 are computed at equidestant points between the body and the shock wave. They are saved for use in subroutine STPRO, where values of W1, W2, W3, W4 are computed at any arbitrary point by interpolation. The initial cross-cut contour must be circular.

SUBJECT: FØRTRAN IV subroutine TANWEDG

PURPOSE: To compute the initial flow field for a flat delta wing using the tangent wedge method. The leading edges must be supersonic.

SUBJECT: FØRTRAN IV subroutine PATCH.

PURPOSE: To patch in a solution in the neighborhood of the leading edge of a sharp edged wing. The edge must be supersonic.

METHOD: The solution that is used is the exact solution for the flow over a wedge in supersonic flow.

SUBJECT: FØRTRAN IV subroutine SETICT

PURPOSE: To position the geometry file ICT pointer at the appropriate geometry file.

METHOD: The geometry files are successively scanned until the file is found which defines the configuration between T1 and T2; where $T1 \leq T \leq T2$, T being the input argument.

USAGE: CALL SETICT(T)

Input: T the input axial position.

ERROR RETURNS: If the required position T is not in the interval defined by the geometry files, an error message is printed out and exit called.

SUBJECT: FØRTRAN IV subroutine START1

PURPOSE: Initialises when a configuration analysis is starting from scratch.

METHOD: A card is read in giving the axial distance at which integration is to commence. The geometry file ICT is then positioned at the appropriate logical file position. The save tape ISV is initialised.

USAGE: CALL START1

SUBPROGRAMS: ENDFL
SETICT

SUBJECT: FØRTRAN IV subroutine START2

PURPOSE: Restarts the computation from data saved on a previous run.

METHOD: The data saved on tape ISV is transferred to scratch files ID2 and IL1 in prepration for the restart.

SUBPROGRAMS: ENDFL

PØSITN

SETICT

SUBJECT: FØRTRAN IV subroutine BØDGØM

PURPOSE: To process the geometry data input cards and produce a set of configuration cross-cut contours. Data corresponding to these contours are saved in a standard format on the geometry file ICT.

METHOD: There are several source decks available, all of them called BØDGØM. Each of these decks has a functional name such as CØNBØD ØR SLBDEL and generates cross cut contours in the standard format for the special body under consideration i.e., conical body or slab delta wing in the above cases. REDBØD is a version of BØDGØM which will read in arbritrary cross-cut contours from cards. The BØDGØMs are described in Section 7.4 of Part I of this report. They are:

CØNBØD

ELLBØD

SLBDEL

SLBDEL APPRØX

CØNE SLBDEL

REDBØD

The standard format for saving geometry data on system file ICT is now described.

The i^{th} file lies between the i^{th} and $(i+i)^{th}$ end of file mark. The i^{th} file contains information needed to generate contours between T(i) and T(i+i). Each file contains three records.

Record 1 has five words IFP, T1, T2, N, M

Record 2 contains arrays X1, Y1, X2, Y2, XD1, YD1, XD2, YD2 NPTA, NA, A1, A2, NPTB, NB, B1, B2. The first eight arrays are of length N, the last eight of length M.

Record 3 consists of an array S of length N

IFP = file number

T1, T2 = interval end points

N = number of points defining contour

M = number of curve segments defining the contour. Each segment has continuous slope and curvature and consists of a chain of cubic splines.

X1 = X coordinates of points defining contour at T1

Y1 = Y coordinates of points defining contour at T1

X2 = X coordinates of points on contour at T2 which correspond with points on T1

Y2 = Y coordinates of points on contour at T2 which correspond with points on T1

XD1 = first derivative in X direction of contour points at T1

YD1 = first derivative in Y direction of contour points at T1

XD2 = second derivative in X direction of contour points at T1

YD2 = second derivative in Y direction of contour points at T1

NPTA NPTA(M) is the contour point number of the starting point of the M^{th} curve segment.

NA If NA(M) = 1; the starting point of the M^{th} curve segment has specified slope, If its value is 2,

curvature is specified.

A1 If NA(M) = 1; A1(M) is the angle in radians made by the tangent at the starting point of the M^{th} curve segment of the contour at T1 with the X reference axis. If NA(M) = 2; it is the curvature at the starting point.

A2 The same as A1, but for the contour at T2.

NPTB NPTB(M) is the contour point number of the end point of the M^{th} curve segment.

NB If NB(M) = 1; the end point of the M^{th} curve segment has specified slope. If its value is 2; curvature is specified.

B1 If NB(M) = 1; B1(M) is the angle in radians made by the tangent at the end point of the M^{th} curve segment of the contour at T2 with the X reference axis. If NB(M) = 2; it is the curvature at the end point.

B2 The same as B1, but for the contour at T2.

SUBJECT: FØRTRAN IV subroutines BGAUX1, BGAUX2, BGAUX3, BGAUX4, BGAUX5

PURPOSE: To assist BØDGØM in setting out the geometry data.

METHOD: These are dummy subroutines in the program file. The user has to use these subroutine names if he requires auxiliary subroutines to set up the configuration geometry.

5.3 Primary Overlay (2,0)

In this overlay the initialisation that needs to be done at the start of a new integration step is performed. Most of the work is done in the program of the overlay namely RESET. DELTAT is called to compute the integration time step and subroutine SAVE is used to save the flow field information needed for a restart.

SUBJECT: FØRTRAN IV program RESET

PURPOSE: To initialise the computations at the start of an integration time step.

METHOD: Debug and print switches are set. Information is saved on TAPE10 if required. The program is stopped and the reason for exit printed out.

USAGE: Program RESET

SUBPROGRAMS: DELTAT
SAVE
TIMER
ENDFL
PØSITN

SUBJECT: FØRTRAN IV subroutine DELTAT

PURPOSE: To compute the size of the next integration time step.

METHOD: Let $DT(P) = \frac{DXYSN}{\sqrt{2*(a+w)}}$

where

DXYSN = AMINI (DX, DY, DS, DN)

a = speed of sound at point P

w = fluid speed at point P

Then DT = the minimum value of DT(P), where P ranges over all grid and boundary points.

The minimum value of DT(P) for the boundary points is DTBND, for the grid points is DTØUT, and for the free stream points is DTFRST. They have already been computed in other subprograms. DT is taken as the minimum of these three quantities. But of (T+DT) overshoots the upper limit T2 for the present geometry file, DT is set at (T2-T).

USAGE: Call DELTAT

SUBJECT: FØRTRAN IV subroutine SAVE

PURPOSE: To save on TAPE10 information needed to restart the program.

METHOD: Flow field and geometry data are transferred from scratch files ID2 and IL1 to save file ISV to provide for a restart.

USAGE: CALL SAVE (S, Y, D, CS, SN, C1, C2, C3, S, XD, YD, XDD, YDD, WC, IX, VX, IY, VY, NSX, NSY, MAPBNW, W1, W2, W3, W4, NPTA, NC, NX, NY, KX, KY, KW, NCURV)

The arguments up to NPTA are the arrays that are saved. The arguments from NC onward are the dimensions for these arrays.

SUBPROGRAMS: PØSITN

ENDFL

5.4 Primary Overlay (3,0)

In this overlay the boundary points are updated using a conservative finite difference scheme based on the Lax-Wendroff method. The program in the overlay namely BNDRY calls six principal subroutines to execute the computation BNDGØM, BNDINT, BNDRED, BND1, BNDWRI.

BNDGØM	concerns itself with geometry problems
BNDINT	interpolates for state variable values at boundary mesh points
BNDRED	brings together data for BND1
BND1	subroutine in which the update scheme is implemented
BNDWRI	controls printout of boundary point data

SUBJECT: FØRTRAN IV program BNDRY

PURPOSE: To control the flow within primary overlay (3,0)

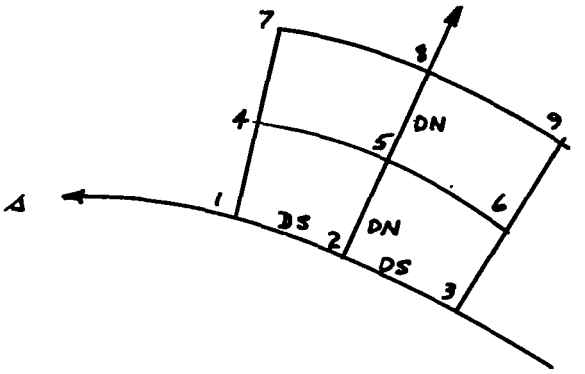
METHOD: It partitions the work array A for the principal subroutines of the overlay, and calls these subroutines to execute the task of updating boundary points.

USAGE: Program BNDRY

SUBJECT: FØRTRAN IV subroutine BNDGØM

PURPOSE: To compute geometric data required in this overlay.

METHOD:



The above figure shows the nine point grid associated with the boundary point being updated, namely point 2. The tasks performed in this subroutine are:

(a) The coordinates of points 1 and 3 are computed such that they are on the contour at an arc distance DS from 2. The values of W at points 1 and 3 are computed by interpolation between the values of W at the contour points.

(b) Normals to the contour at points 1, 2, 3 are erected and the coordinates of points 4, 5, 6, 7, 8, 9 are found.

(c) The W values, the cosine and sine of the normals and the curvature at points 1, 2, 3 are saved on disc.

(d) The coordinates of points 4,5,6,7,8,9 are saved on disc for subroutine BNDINT which will compute the W values at these points by interpolation.

(e) If any of the boundary grid points are to the left of the line of symmetry they are reflected across the line of symmetry.

The above operations are done for each of the NC contour points.

USAGE: CALL BØDGØM (A(N1), A(N2),.....A(N17), NP)

Input A(NI) = the addresses of storage areas used by the subroutine

NP a variable dimension used by the subroutine to dimension the arrays

SUBPROGRAMS: PØSITN

ALL

NRMCUR

ENDFL

ERROR RETURNS: A check is made on the number of boundary grid points. If incorrect, an error message is printed and exit called.

SUBJECT: FØRTRAN IV subroutine BNDINT

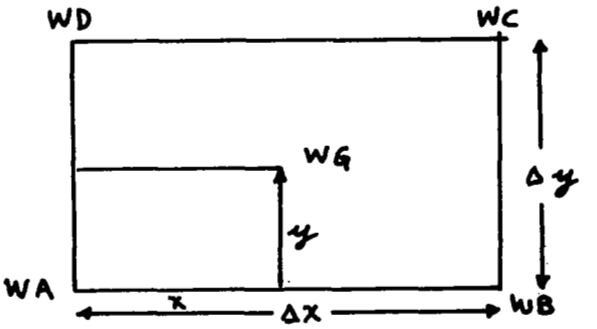
PURPOSE: To compute W at points 4,5,6,7,8,9 of the nine point boundary grid.

METHOD:

(a) The W values for a block of grid points are read in from disc file IL1.

(b) The coordinates of boundary grid points which are in the area covered by this block are read in one at a time from disc file ID2.

(c) If the point is surrounded by four regular mesh points, bilinear interpolation is performed as shown below.



Set X1 = x/DX

Y1 = y/Dy

X2 = 1 -X1

Y2 = 1-Y1

Then WG = (X1*Y1)*WA + (X2*Y1)*WB + (X2*Y2)*WC + (X1*Y2)*WD

(d) If the configuration contour intersects the rectangular grid surrounding the point, then PØL3 is used to estimate the value of WG. The method is described in PØL3.

(e) The interpolated values are saved in array WCP for use in the update subroutine BND1.

USAGE: CALL BNDINT (A(N1),.....A(N11), MM1,NP,KX,NX,KY,NY)

Input A(NI) = the addresses of storage areas used in the subroutine.

MMI, NP, KX, NX, KY, NY = variable dimensions used in BNDINT

SUBPROGRAMS: PØSITN

ENDFL

PØL3

ERROR RETURNS: If an error occurs, diagnostic print out is produced and the program exits.

SUBJECT: FØRTRAN IV subroutine BNDRED

PURPOSE: To get together in core the data required to update the contour boundary points.

METHOD: The required geometry and flow field data are read in from disc files.

USAGE: CALL BNDRED (A(N1), ...A(N8), NP)

Input A(NI) = addresses of storage areas used by the subroutine

NP = a variable dimension used in the subroutine to dimension various arrays in BNDRED.

SUBPROGRAMS: PØSITN

ENDFL

PØL3

SUBJECT: FØRTRAN IV subroutine BND1

PURPOSE: To update the flow field values on the boundary using a conservative finite difference scheme.

METHOD: The method is described in Section 7.1 of Part I of the document. BND1 implements these finite difference equations for the contour boundary points updating them one at a time. When BNDISC = 1., those contour boundary points which have discontinuities in slope or curvature are not updated by the method but by interpolation between adjacent points.

USAGE: CALL BND1 (A(N1), ... A(N12), NP, M)

Input A(NI) = addresses of storage areas used by the subroutine

NP, M = variable dimensions used in the subroutine to dimension various arrays in BND1.

SUBPROGRAMS: PØSITN

BWTØW

WFGH

SUBJECT: FØRTRAN IV subroutine BNDWRI

PURPOSE: To print out the state variables on the contour boundary.

METHOD: Pressure, temperature and local Mach number are computed. The state variables are printed out.

USAGE: CALL BNDWRI (A(NI), ... A(N14), NP)

Input A(NI) = addresses of storage areas used by the subroutine

NP = variable dimension used in the subroutine to dimension various arrays in BNDWRI

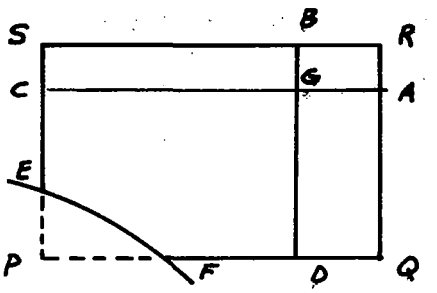
SUBPROGRAMS: PØSITN

ENDFL

SUBJECT: FØRTRAN IV subroutine PØL3

PURPOSE: To interpolate for the value of W at a point near the boundary contour.

METHOD:



PQ = SR = dx

PS = QR = dy

The value of W is known at the grid intersections Q, R, S and at E, F, the intersections of the grid lines with the contour. The point P is inside the contour.

Estimated values of W are obtained for point A by linear interpolation between Q and R, point B by linear interpolation between R and S, point C by linear interpolation between S and E, point D by linear interpolation between Q and F.

Then WG is taken as the mean of the linearly interpolated values between B and D, and A and C.

USAGE: CALL PØL3 (WQ, WR, WS, ZX, ZY, WH, WV, ZH, ZV, ANS)

Input	WQ	value of W at Q
	WR	value of W at R
	WS	value of W at S
	ZX	CG/dx on figure
	ZY	DG/dy on figure
	WH	value of W at F
	WV	value of W at E
	ZH	PF/dx on figure
	ZV	PE/dy on figure
Output	WANS	value of W at G

SUBJECT: FØRTRAN IV subroutine WFGH

PURPOSE: To transform W from Cartesian axes to body axis system and compute the vectors F, G, H.

METHOD: A mere implementation of algebraic formulas for W,F,G,H in Section 7.1 of Part I.

SUBJECT:	FØRTRAN IV subroutine BWTØW
PURPOSE:	Given W in moving body coordinates, to compute the corresponding values of W in the Cartesian frame.
METHOD:	A mere implementation of transformation equations in Section 7.1 of Part I.

5.5 Primary Overlay (33.0)

In this overlay the boundary points are updated using a quasi-one-dimensional method of characteristics. The program in the overlay namely BNDRC calls six principal subroutines to execute the computation: BNCGØC, BNDINC, BNDREC, BND3, BNDWRC

BNDGØC concerns itself with geometry problems

BNDINC interpolates for state variable values at boundary mesh points

BNDREC brings together data for BND3

BND3 implements the method of characteristics

BNDWRC controls printout of boundary point data

SUBJECT: FØRTRAN IV program BNDRC

PURPOSE: To control the flow within primary overlay (33,0)

METHOD: It partitions the work array A for the principal subroutines to execute the task of updating boundary points.

USAGE: Program BNDRC

SUBJECT: FØRTRAN IV subroutine BNDGØC

PURPOSE: To compute geometric data required in this overlay.

METHOD: The rest of this subroutine description is identical to that of BNDGØM.

SUBJECT: FØRTRAN IV subroutine BNDINC

PURPOSE: To compute W at the boundary grid points.

METHOD: The description of the subroutine is identical with that of BNDINT.

SUBJECT: FØRTRAN IV subroutine BNDREC

PURPOSE: To assemble in core the data required to update the contour boundary points.

METHOD: The description of this subroutine is identical with that of BNDRED .

SUBJECT: FØRTRAN IV subroutine BND3

PURPOSE: To update the flow field values on the boundary using the method of characteristics.

METHOD: The method is described in Section 7.2 of Part I of this document. BND3 implements these finite difference equations for the contour boundary points, updating them one at a time. When BNDISC = 1., these contour boundary points which have discontinuities in slope or curvature are not updated by the method but by interpolation between adjacent points.

USAGE: CALL BND3 (A(N1), ... A(N12), NP, M)

Input A(NI) = addresses of storage areas used by the subroutines

NP, M = variable dimensions used in the subroutine to dimension various arrays in BND1.

SUBPROGRAMS: PØSITN

SMALRT

STATE

SUBJECT: FØRTRAN IV subroutine BNDWRC

PURPOSE: To print out the state variables on the contour boundary.

METHOD: Pressure, temperature and local Mach number are computed. The state variables are printed out.

USAGE: CALL BNDWRC (A(N1), ... A(N13), NP)

Input A(NI) = addresses of storage areas used by the subroutine

NP = variable dimensions used in the subroutine to dimension various arrays in BNDWRI

SUBPROGRAMS: PØSITN

ENDFL

SUBJECT: FØRTRAN IV subroutine PØLC

PURPOSE: To interpolate for the value of W at a point near the boundary contour.

METHOD: See description of PØL3.

SUBJECT: FØRTRAN IV subroutine STATE

PURPOSE: To convert Cartesian values of W to local coordinate values and then compute pressure, speed of sound and entropy.

SUBJECT: FØRTRAN IV subroutine SMALRT

PURPOSE: To compute the smallest positive root of a quadratic

A*Z*Z + BB*Z + C = Q

METHOD: To eliminate cancellation errors the two roots are computed as

X1 = DD/A X2 = C/DD

where

DD = -B - SIGN (1.,B) * SQRT(DEL)

DEL = B*B - A*C

B = .5*BB

USAGE: CALL SMALRT (C, BB, A, X)

Input C,BB,A = coefficients of quadratic

Output X smallest positive root if any

ERROR RETURNS: If the roots are imaginary or there is no positive root, diagnostic print out is produced and exit called.

5.6 Primary Overlay (4,0)

In this overlay the cross-cut contour at the new time step is computed. It is generated by interpolation between two consecutive contours on the geometry tape. FINDCT is the overlay program. NEWCT1 is the subroutine called to do the interpolation, but when a new geometry file is needed, NEWCT2 is called.

SUBJECT: FØRTRAN IV program FINDCT

PURPOSE: To partition work array A.

METHOD: After array A has been cut up, the new contour is computed by calling NEWCT1 or NEWCT2.

USAGE: Program FINDCT

SUBJECT: FØRTRAN IV subroutine NEWCT1

PURPOSE: To interpolate for the cross-cut contour at time T

METHOD:

(a) The coordinates of the cross-cut contours at T1 and T2 together with their velocities and accelerations are read in from the geometry file ICT.

(b) The X-position of contour point I at time T is found by the formula

X = X1 + V1*(T-T1) + .5*A1*(T-T1)**2

where X = position of contour point I at time T

X1 = position of contour point I at time T1

V1 = velocity of contour point I at time T1

A1 = acceleration of contour point I at time T1

(c) The slope or curvature conditions at the end of the cubic spline segments are obtained by interpolation.

(d) Non-linear cubic splines are fitted to each curve segment.

(e) All these data for the contour at time T are saved in disc file ID2.

USAGE: CALL NEWCT1 (A(N1), ... A(N30), N, M)

Input A(NI) = the addresses of storage areas used by the subroutine

N, M available dimensions used by the subroutines to dimension the arrays.

SUBPROGRAMS: PØSITN

NRMCUR

NLCSP

ENDFL

CUBARC

SUBJECT: FØRTRAN IV subroutine NEWCT2

PURPOSE: To compute cross cut contour data at a value of T which is the starting point of a new geometry data file.

METHOD: The method is the same as in NEWCT1, except that no interpolation needs to be done since T coincides with T1, the starting point of a geometry data file.

But as T is the end of a geometry data file as well as the starting point of a new file the number of points on the two identical contours at T may be different. Therefore, the value of W at the new contour points is computed by interpolation between the old contour points at T.

USAGE: See NEWCT1

SUBPROGRAMS: NLCSP

ENDFL

CUBARC

PØSITN

5.7 Primary Overlay (5,0)

In this overlay the intersections of the grid lines with the contour are found in subroutine INTCØN. Using this information, maps of inside,' between' and Lax-Wendroff points are produced in subroutine MAPIT. The coordinating program in the overlay is MAPS.

SUBJECT: FØRTRAN IV subroutine MAPS

PURPOSE: To direct work in this overlay.

METHOD: Partitions work array A, calls principal subroutines INTCØN and MAPIT, and times the overlay.

SUBJECT: FØRTRAN IV subroutine INTCØN

PURPOSE: To find intersection of the grid lines with the contour.

METHOD: The cubic curves which comprise the contour are considered one at a time and their intersections with the NX vertical grid lines are found and saved in array VX. The intersections with the NY horizontal grid lines are saved in array VY.

SUBPROGRAMS: PØSITN

ENDFL

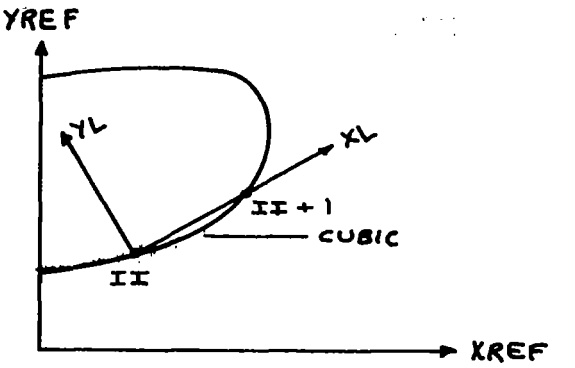
INTCUB

SUBJECT: FØRTRAN IV subroutine INTCUB

PURPOSE: To find the intersection of a grid line with a cubic defining the contour between two contour points.

METHOD: The cubic equation is set up and the polynomial equation solver RØØT3 is called.

USAGE: CALL INTCUB (II, XØ, YØ, D, CS, SN, C1, C2, C3, SS, A, B, C, TØLCF, TØLS, NR, XR, YR, NS)



Input II — cubic segment extends from contour point II to (II + 1)

XØ, YØ — reference coordinates of contour point II

D — chord distance from point II to (II + 1)

CS, SN — cosine, sine of angle made by XL with XREF

C1, C2, C3 — coefficients of cubic YL=C1*XL+C2*XL**2+C3*XL**3

SS — arc length of cubic from II to (II +1)

A, B, C — AX+BY+C = 0 is straight line in reference axes

TØLCF — tolerance used to suppress small coefficients of the cubic

TØLS — tolerance to which an arc length must be computed

NR number of intersections of cubic with given line

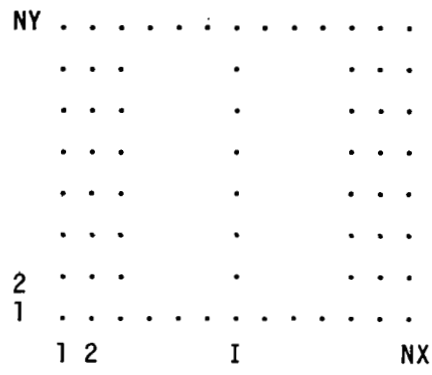
XR, YR arrays of length 3. They contain in reference axes the coordinates of NR intersection points.

NS is an array of length three. Each cell contains the information about the position of the intersection in the cross cut contour.

SUBJECT: FØRTRAN IV subroutine MAPIT

PURPOSE: To tag each grid point so that it can be identified as an outside point, or a 'between' point or a Lax-Wendroff point.

METHOD: Introduction:



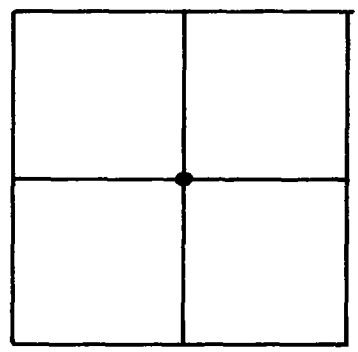
These are NX*NY grid intersection points. Let KW = 1 + (NY-1)/60. Consider the array MAP(KW, NX). This doubly dimensioned Fortran array may be regarded as a two dimensional array of bits, NX columns of 60*KW bits each. The first NY bits of the Ith column are assumed to correspond to the NY grid points in the Ith column.

Let a bit be assigned the value 1 if the corresponding grid point lies outside the contour and zero if it lies inside. Then we say that we have a map of outside points. In a similar way a map of Lax-Wendroff points is obtained if the bits corresponding to Lax-Wendroff points are set

to 1, the other bits being set to zero. The same method is applied to 'between' points. Thus we have a compact means of storing information about the grid.

(a) Using the contour grid intersection information, MAPØUT is generated. It is a map of the outside points for the present time step.

(b) A map of possible between points for the next time step (MAPBNW) is now generated. The bit corresponding to a grid point is set to unity if the contour intersects any of the sides of the four grid rectangles centered on the grid point



(c) The map (MAPBØL) of possible 'between' points for the previous time step is read in from disc. The map of between points (MAPBET) is the logical intersection of MAPBØL and MAPØUT.

(d) The map (MAPLXW) of Lax-Wendroff points is then produced. Those points which are not between points and lie outside the contour are Lax-Wendroff points.

(e) A list LISTB of between points is generated.
LISTB(NBT) = 10000*I+J, when between point number NBT is the intersection of the Ith vertical grid line and the Jth horizontal grid line.

(f) MAPØUT, MARBET, MAPLXW, and LISTB are saved on disc for use in subsequent overlays.

SUBJECT: FORTRAN IV subroutine RØØT3

PURPOSE: To find the roots of a polynomial of degree less than or equal to 3.

METHOD: Uses the Newton-Bairstow method. See for example Ellenberger, K. W.,"On Programming the Numerical Solution of of Polynomial Equations" CACM, Vol. 3, No. 12, 1960, p. 644.

USAGE: CALL RØØT3 (N, A, U, V, CØNV)

Input N degree of polynomial to be solved. $N \leq 3$

A one dimensional array of polynomial coefficients, A(1) being the coefficient of x^0. A must be of dimension N + 1.

Output U one dimensional array of real part of roots; dimension N

V one dimensional array of imaginary part of roots; dimension N

CØNV Final tolerance if solved,or error code.

ERROR RETURNS Error condition CØNV = -1. returned if A(N+1) = 0.

Error condition CØNV = -2. returned if tolerance E equals machine infinity.

Success condition CØNV = 0. indicating final tolerance used for convergence test.

5.8 Primary Overlay (6,0)

In this overlay the core of the computation is done. Most of the field points are updated using the Lax-Wendroff finite difference technique. The overlay program is LAXWDF. The principal subroutine is LXWN.

SUBJECT: FORTRAN IV program LAXWDF

PURPOSE: To partition the work array A for the principle subroutine LXWN and time the overlay.

SUBJECT: FØRTRAN IV subroutine LXWN

PURPOSE: To implement the Lax-Wendroff finite difference method.

METHOD: The equations for the Lax-Wendroff method are described in 3.2.2 of Part I.

The procedure may be broken up into the following steps.

(a) MAPLXW is read in from disc so that any grid point (I, J) can be checked to see if it is a Lax-Wendroff point and needs to be updated.

(b) The values of W at the previous time step are brought into core from disc file IL1 one block at a time. The Lax-Wendroff points in the block are updated.

(c) The computations are organized so that there is no unnecessary recomputation. Each updated block is saved on disc file IL2.

USAGE: CALL LXWN (A(N1), ... A(N14), MM3, ...NY)

Input A(NI) the addresses of storage areas used by the subroutine

MM3 .. NY variable dimensions used in LXWN to dimension the arrays

SUBPROGRAMS: PØSITN

MAPCHK

ENDFL

FF

FG

GG

SUBJECT: FØRTRAN IV subroutines FF, GG, FG

PURPOSE: These subroutines compute the vectors f only, g only, and both f and g from vector w.

METHOD: w, f, g are defined in Section 3.2.2 of Part I.

5.9 Primary Overlay (7,0)

In this overlay, simultaneous linear equations, Ax = B, to update the 'between' points are set up and solved. 'Between' points are those grid intersection points near the boundary which cannot be updated by the Lax-Wendroff method.

BETWN	is the program in the overlay
LAPGUN	sorts out the geometry associated with each 'between' point
RHSBND	computes contributions to the B matrix from the boundary contour
RHSLXW	computes contributions to the B matrix from the Lax-Wendroff points
LHS	computes the A matrix
SØLVE	solves the equations using Gauss-Seidel iteration

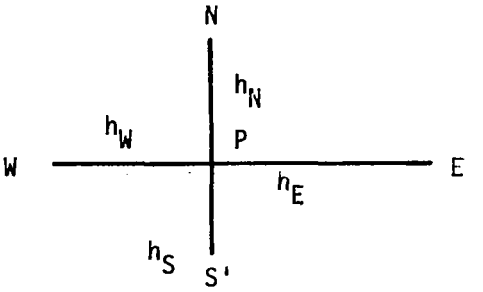
SUBJECT: FØRTRAN IV program BETWN

PURPOSE: This is the program in the overlay. It cuts up the work array A, calls the principle subroutines and times the overlay.

SUBJECT: FØRTRAN IV subroutine LAPGUN

PURPOSE: To compute the coefficients of the Laplace star associated with a between point.

METHOD:



If S, E, N, W are equispaced from P, then a good estimate for f_p is $.25*(f_S + f_E + f_N + f_W)$. This is equivalent to a discretisation of the diffusion equation. If the points are not equispaced from P, a discretisation of the differential equation for unequally spaced points is used.

$$\frac{\dfrac{f_E - f_P}{h_E} - \dfrac{f_P - f_W}{h_W}}{\dfrac{h_E + h_W}{2}} + \frac{\dfrac{f_N - f_P}{h_N} - \dfrac{f_P - f_S}{h_S}}{\dfrac{h_N + h_S}{2}} = 0$$

Simplifying

$$f_P = A_S f_S + A_E f_E + A_N f_N + A_W f_W = 0$$

$$A_S = \frac{1}{h_S\,(h_N + h_S)\,D}$$

$$A_E = \frac{1}{h_E\,(h_E + h_W)\,D}$$

$$A_N = \frac{1}{h_N\,(h_N + h_S)\,D}$$

$$A_W = \frac{1}{hW\ (h_E + h_W)\ D}$$

$$D = -\left(\frac{1}{h_E h_W} + \frac{1}{h_N h_S}\right)$$

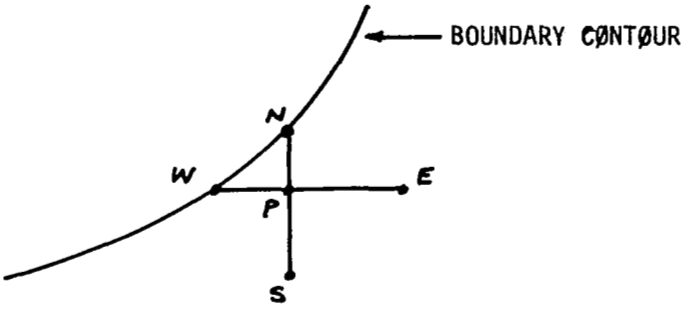
The list of 'between' points LISTB is read in from disc file ID1. The arms h_S, h_E, h_N, h_W are computed for each point using the contour intersection information which is read in from disc file ID2. A_S, A_E, A_N, A_W are computed and saved in the array ARM.

SUBPROGRAM: PØSITN

SUBJECT: FØRTRAN IV subroutine RHSBND

PURPOSE: To compute the contribution from the boundary contour to the right hand sides of the equations to update the between points.

METHOD:



The equation for the point P is

$f_P + A_S f_S + A_E f_E + A_N f_N + A_W f_W = 0$

W, N are points on the contour. f_N, f_W are found by interpolation between the contour points. Assume that S, E are also between points. The equation is rearranged to read

$f_P + A_S f_S + A_E f_E = - A_N f_N - A_W f_W$

$-A_N f_N - A_W f_W$ is the contribution from the boundary to the right hand side of the equation.

The subroutine RHSBND computes these contributions to the right hand side B which is a NBT*4 matrix where NBT are the number of unknowns. There are four right hand sides corresponding to the four independent variables in the partial differential equations namely W(1), W(2), W(3), W(4).

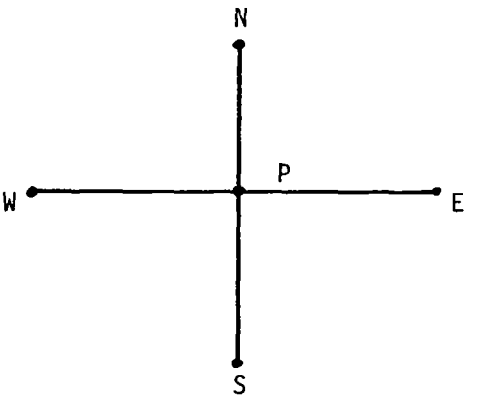
SUBPROGRAMS: PØSITN

ENDFL

SUBJECT: FØRTRAN IV subroutine RHSLXW

PURPOSE: To compute the contribution from the regular grid points to the right hand sides of the 'between' update equations.

METHOD:



The equation for the 'between' point P is

$f_P + A_S f_S + A_E f_E + A_N f_N + A_W f_W = 0.$

Assume S, W are also 'between' points. Then f_S, f_W are unknowns in the equations. If E, N are Lax-Wendroff points then f_E, f_N have already been computed. Therefore the equation is

$f_P + A_S f_S + A_W f_W = - A_E f_E - A_N f_N$

$-A_E f_E - A_N f_N$ is the contribution from the Lax-Wendroff points to the right hand side of the equation.

The subroutine RHSLXW computes these contributions to the right hand side B which is a NBT*4 matrix where NBT are the number of unknowns. There are four right hand sides corresponding to the four independent variables in the partial differential equations namely W(1), W(2), W(3), W(4).

SUBPROGRAMS: PØSITN

SUBJECT: FØRTRAN IV subroutine LHS

PURPOSE: To compute the matrix A for the linear equations AX = B which update the 'between' points.

METHOD: The diagonal elements of matrix A are all unity. Each 'between' point can be connected to at the most four other 'between' points. Therefore, the maximum number of non zero off-diagonal elements in a row is four. The matrix is saved in a compacted form in a NBT*4 array AA. An auxiliary NBT*4 array NN contains information about the interconnections among the 'between' points.

Here is an illustration. If the first two equations are

$$X_1 + .75\,X_{10} + .01\,X_{11} = 2.$$
$$X_2 + .60\,X_{10} + .15\,X_{15} + .05\,X_{21} = 3.5$$

Then AA = $\begin{bmatrix} .75 & .01 & 0 & 0 \\ .60 & .15 & .05 & 0 \\ & . & . & . \end{bmatrix}$

NN = $\begin{bmatrix} 10 & 11 & 0 & 0 \\ 10 & 15 & 21 & 0 \\ & . & . & . \end{bmatrix}$

SUBPROGRAMS: PØSITN

ENDFL

SUBJECT: FØRTRAN IV program SØLVE.

PURPOSE: To solve a set of simultaneous linear equations.

METHOD: Since the equations to be solved are sparse, an iterative method (Gauss-Seidel) is used. There are four sets of simultaneous linear equations as there are four right hand sides. The iterative process is stopped when

DELMX4 = TØLSØL * ELMX4

where

DELMX4 = maximum change in Wr between previous and present iteration

ELMX4 = maximum value of W4

TØLSØL = presently set at .0001 in subroutine INIT

It usually takes 3 to 10 iterations to attain this tolerance.

SUBPROGRAMS: PØSITN

ENDFL

5.10 Primary Overlay (10,0)

In this overlay the updated values of W for the 'between' points is merged with the flow field values for the regular grid points. The time step is computed to satisfy the Courant-Friedrichs stability criterion. The entire flow field values are printed out if required.

SUBJECT: FØRTRAN IV program STBPRN

PURPOSE: It is the program in the overlay. Cuts up work array A and calls STPRO or STPR.

SUBJECT: FØRTRAN IV subroutine STPR

PURPOSE: To compute the time step from the grid points and print out the flow field.

METHOD: The flow field data at the 'between' points is merged with the data for the Lax-Wendroff points. The flow field is printed out if IPRNT $\neq$ 0. The new time step for the grid, DTØUT, is computed according to the formula in subroutine DELTAT.

SUBJECT: FØRTRAN IV subroutine STPRØ

PURPOSE: Initialises the flow field at the start of a run.

METHOD: Sets the flow field values to free stream.

6.0 FILE DESCRIPTION

There are eight files used. Seven are disk files. The last file is used to save data required to restart the run. It is usually a tape. The table below summarizes the file information.

FILE NAME	FORTRAN NAME	BCD OR BINARY	DISK OR TAPE	COMMENTS
TAPE5	IR	BCD	Disk	Input
TAPE6	IW	BCD	Disk	Output
TAPE2	ID1, ID2	Binary	Disk	Scratch — Roles interchanged every time step
TAPE3	ID2, ID1	Binary	Disk	Scratch
TAPE4	IL1, IL2	Binary	Disk	Scratch
TAPE7	IL2, IL1	Binary	Disk	Scratch
TAPE1	ICT	Binary	Disk	Scratch — Contains configuration data
TAPE10	ISV	Binary	Tape	Save-tape — Data saved for restart

Each of the four files ID1, ID2, IL1, IL2 is a scratch file on disc containing binary data. There is one or more files on each of these scratch units. Each logical file consists of one or more records. Each record consists of a set of arrays. Four tables follow. Each is a sequential list of READ/WRITE statements addressed to the scratch files ID1, ID2, IL1, IL2. The I/Ø statements in overlay BNDRC are omitted from the list as they are almost identical with those in overlay BNDRY. For convenience the overlays are identified by the program name in the overlay.

OVERLAY	SUBROUTINE	READ/ WRITE	LOGICAL FILE	CONTENTS
RESET	RESET	WRITE	1	MAPBØL
BNDRY	BNDGØM	READ	2	XP, YP, DP, CSP, SNP, C1P, C2P, C3P, SP
		READ	4	WCP
	BNDINT	READ	5	IX, VX, IY, VY, NSX, NSY
		READ	4	WCP
		WRITE	4	(JPT, IPT, WCP) 6*NC records
	BNDRED	READ	3	XD, YD, XDD, YDD
		READ	4	(JPT, IPT, WCP) 6*NC records
	BND1	READ	2	S, NPTA
	BNDWRI	WRITE	1	WC
FINDCT	NEWCTI	READ	1	WC
MAPS	MAPIT	WRITE	1	MAPLWX
		WRITE	2	LISTB
LAXWND	LXWN	READ	1	MAPLXW
BETWN	LAPGUN	READ	2	LISTB
	RHSLXW	WRITE	3	SØL
STBPRN	STPR	READ	2	LISTB
		READ	3	SØL

TABLE 2 SEQUENTIAL I/O LIST FOR FILE ID1

OVERLAY	SUBROUTINE	READ/ WRITE	LOGICAL FILE	CONTENTS
RESET	RESET	WRITE	1	MAPBØL
BNDRY	BNDGØM	WRITE	2	WCP, CØSN, SINN, CUR, W4
			3	(J, I, X, Y) 6*NC records
	BNDINT	READ	3	(J, I, X, Y) 6*NC records
	BNDRED	READ	2	WCP, CØSN, SINN, CUR
	BNDWRI	READ	2	W4
FINDCT	NEWCTI	WRITE	2	X, Y, C, CS, SN, C1, C2, C3, S', S, NPTA
		WRITE	3	XD, YD, XDD, YDD
		WRITE	4	WC
MAPS	INTCØN	READ	2	X, Y, D, CS, SN, C1, C2, C3, S
		WRITE	5	IX, VX, IY, VY, NSX, NSY
	MAPIT	READ	1	MAPBØL
		WRITE	6	MAPBNW
LAXWND	LXWN			
BETWN	LAPGUN	READ	5	IX, VX, IY, VY, NSX, NSY
	RHSBND	READ	4	WC
STBPRN				

TABLE 3 SEQUENTIAL I/O LIST FOR FILE ID2

OVERLAY	SUBROUTINE	READ/ WRITE	LOGICAL FILE	CONTENTS
BNDRY	BNDINT	READ	1	W1, W2, W3, W4
LAXWND	LXWN	READ	1	W1, W2, W3, W4
STBPRN	STPR	WRITE	1	W1, W2, W3, W4

TABLE 4 SEQUENTIAL I/O LIST FOR FILE IL1

OVERLAY	SUBROUTINE	READ/ WRITE	LOGICAL FILE	CONTENTS
LAXWND	LXWN	WRITE	1	W1, W2, W3, W4
BETWN	RHSLXW	READ	1	W1, W2, W3, W4
STBPRN	STPR	READ	1	W1, W2, W3, W4

TABLE 5 SEQUENTIAL I/O LIST FOR FILE IL2

7.0 SAMPLE INPUT AND OUTPUT DATA

The input data preparation and printed output are described in the section on computer usage, that is Section 4 of Part I. The computer input and output data for a typical problem, namely a cone at an angle of attack, are given in the print out that follows.

CIRCULAR CONE HALF ANGLE = 10 DEG ANGLE OF ATTACK = 5 DEG MACH NO = 6 MAR 10, 1971

NX	NY	MDY	DX	DY	DS	DN
26	51	25	.010000	.010000	.010000	.010000

UNIFY	FSMN	GAMMA	ALFAD
1	6.0000	1.4000	5.0000

BND	BNDISC
1	-0

NCTMX	NBTMX	KX	KY
40	100	4	2

NPRNT	NBUG	NSAVE
10	-0	-0

TMX	NTMX
1.0000	3

IOCTL
0000001000

CORE STORAGE ANALYSIS

NCTMX	NBTMX	NX	NY	KX	KY	KW	NN
40	100	26	51	4	2	1	7000

SUBROUTINE	MINIMUM WORDS REQD	WORDS USED	WORDS UNUSED
BNDGOM	1680		5320
BNDINT	986	5882	1118
BNDRED	1960		5040
BND1	2280		4720
NEWCT	1160		5840
INTCON	360		6640
MAPIT	364		6636
LXWN	2015	6911	89
LAPGUN	900		6100
RHSBND	1460		5540
RHSLXW	1912	6808	192
LHS	1300		5700
SOLVE	1700		5300
STPR	704	5600	1400

KBLKS	MS	ML	MM
2	25	1	1275

```
CIRCULAR CONE  HALF ANGLE =  10 DEG
 CONE SEMI-ANGLE =    10.0000 DEGREES
      FILE        T1          T2          N
         1      .2000     1.0000          19
         2     1.0000     2.0000          37

 START1 CALLED

 STARTING VALUE OF T =              .200000

 GEOMETRY IOUNIT SET AT FILE POSITION      1

CP,PP, TIMES TO  $ INITIALISE $ ARE(SEC)        .10       1.66

 CP,PP TIMES FOR  $ RESET  $  ARE (SEC)         .00        .53      T =     .200000      NT =       0

 CP,PP TIMES FOR  $ BNDRY  $  ARE (SEC)         .00        .75      T =     .200000      NT =       0

 I,X,Y,C1,C2,C3,S ARE
    1         0.00000000      -.03526540          -.08748866     14.23236299      -.00000000          0.00000000
    2          .00612377      -.03472964          -.08748866     14.23236299       .00000000           .00615500
    3          .01206148      -.03313863          -.08748866     14.23236299       .00000000           .01230999
    4          .01763270      -.03054073          -.08748866     14.23236299      -.00000000           .01846499
    5          .02266816      -.02701486          -.08748866     14.23236299       .00000000           .02461999
    6          .02701486      -.02266816          -.08748866     14.23236299       .00000000           .03077498
    7          .03054073      -.01763270          -.08748866     14.23236299       .00000000           .03692998
    8          .03313863      -.01206148          -.08748866     14.23236299      -.00000000           .04308498
    9          .03472964      -.00612377          -.08748866     14.23236299       .00000000           .04923997
   10          .03526540      -.00000000          -.08748866     14.23236299      -.00000000           .05539497
   11          .03472964       .00612377          -.08748866     14.23236299       .00000000           .06154997
   12          .03313863       .01206148          -.08748866     14.23236299      -.00000000           .06770496
   13          .03054073       .01763270          -.08748866     14.23236299       .00000000           .07385996
   14          .02701486       .02266816          -.08748866     14.23236299      -.00000000           .08001496
   15          .02266816       .02701486          -.08748866     14.23236299       .00000000           .08616995
   16          .01763270       .03054073          -.08748866     14.23236299      -.00000000           .09232495
   17          .01206148       .03313863          -.08748866     14.23236299      -.00000000           .09847995
   18          .00612377       .03472964          -.08748866     14.23236299       .00000000           .10463494
   19         0.00000000       .03526540           0.00000000      0.00000000      0.00000000           .11078994

 CP,PP TIMES FOR  $ FINDCT $  ARE (SEC)         .10       1.85      T =     .200000      NT =       0

 CP,PP TIMES FOR  $ MAPS   $  ARE (SEC)         .09       1.57      T =     .200000      NT =       0

 CP,PP TIMES FOR  $ LAXWND $  ARE (SEC)         .01        .53      T =     .200000      NT =       0

 CP,PP TIMES FOR  $ BETWN  $  ARE (SEC)         .00        .65      T =     .200000      NT =       0

 CP,PP TIMES FOR  $ STBPRN $  ARE (SEC)         .29       1.23      T =     .200000      NT =       0

CP,PP TIME (SEC) FOR THIS TIME STEP        .50        7.10
                 TOTAL SINCE LOAD         5.46      124.10
T =     .200000
NT =       0

NEW TIME STEP BEGINS
**********************
```

NT	T	DT	DTFRST	DTBND	DTOUT	INDCT
1	.220475	.020475	.027565	.020475	.027565	0

CP,PP TIMES FOR $ RESET $ ARE (SEC) .02 3.19 T = .220475 NT = 1

I	WC1	WC2	WC3	WC4	PRES	TEMP	VTANG	MLOCL	CP	DW4BYDT	I
1	1.360039	.000000	-.131836	1.663288	1.553421	1.133614	.000000	.546263	.021961	17.514102	1
2	1.357104	.026641	-.129269	1.659199	1.548644	1.132624	.002792	.548303	.021772	17.314378	2
3	1.348427	.052078	-.121748	1.647133	1.534544	1.129685	.005411	.554364	.021212	16.725087	3
4	1.334378	.075251	-.109787	1.627687	1.511793	1.124903	.007701	.564283	.020309	15.775314	4
5	1.315552	.095354	-.094127	1.601813	1.481454	1.118444	.009534	.577825	.019105	14.511615	5
6	1.292722	.111876	-.075610	1.570762	1.444897	1.110532	.010824	.594718	.017655	12.995032	6
7	1.266799	.124578	-.055056	1.535993	1.403698	1.101439	.011532	.614673	.016020	11.296879	7
8	1.238773	.133416	-.033176	1.499070	1.359535	1.091475	.011670	.637365	.014267	9.493534	8
9	1.209662	.138447	-.010542	1.461545	1.314084	1.080970	.011292	.662400	.012464	7.660785	9
10	1.180458	.139743	.012383	1.424848	1.268925	1.070268	.010490	.689261	.010672	5.868465	10
11	1.152087	.137341	.035190	1.390199	1.225479	1.059706	.009379	.717266	.008948	4.176181	11
12	1.125373	.131243	.057446	1.358556	1.184959	1.049607	.008081	.745559	.007340	2.630702	12
13	1.101018	.121459	.078651	1.330600	1.148352	1.040264	.006707	.773136	.005887	1.265272	13
14	1.079599	.108062	.098216	1.306755	1.116423	1.031935	.005351	.798906	.004620	.100658	14
15	1.061562	.091256	.115486	1.287238	1.089731	1.024837	.004076	.821776	.003561	-.852543	15
16	1.047239	.071412	.129791	1.272120	1.068664	1.019145	.002914	.840734	.002725	-1.590959	16
17	1.036862	.049085	.140516	1.261380	1.053473	1.014989	.001865	.854925	.002122	-2.115464	17
18	1.030581	.024999	.147166	1.254969	1.044307	1.012460	.000908	.863706	.001758	-2.428606	18
19	1.028479	0.000000	.149420	1.252838	1.041244	1.011612	0.000000	.866678	.001637	-2.532688	19

I	WS1	WS2	WS3	WS4	PRES	TEMP	VTANG	MLOCL	CP	I
1	1.477365	.000000	-.260499	2.177095	1.747722	1.172178	.000000	.977177	.029671	1
2	1.472903	.041023	-.256486	2.168554	1.740217	1.170748	-.002810	.977898	.029374	2
3	1.459735	.080175	-.244728	2.143412	1.718118	1.166510	-.005729	.980065	.028497	3
4	1.438487	.115809	-.226021	2.103066	1.682628	1.159622	-.008840	.983687	.027088	4
5	1.410155	.146668	-.201518	2.049691	1.635632	1.150336	-.012182	.988762	.025223	5
6	1.376028	.171956	-.172537	1.986027	1.579523	1.138993	-.015726	.995245	.022997	6
7	1.337600	.191301	-.140366	1.915121	1.517005	1.126005	-.019371	1.003009	.020516	7
8	1.296466	.204644	-.106132	1.840065	1.450879	1.111838	-.022939	1.011797	.017892	8
9	1.254227	.212090	-.070751	1.763766	1.383842	1.096989	-.026189	1.021191	.015232	9
10	1.212394	.213778	-.034966	1.688783	1.318334	1.081965	-.028840	1.030615	.012632	10
11	1.172317	.209800	.000566	1.617250	1.256417	1.067256	-.030601	1.039392	.010175	11
12	1.135135	.200201	.035165	1.550881	1.199723	1.053317	-.031211	1.046662	.007926	12
13	1.101748	.185029	.068056	1.491019	1.149446	1.040547	-.030476	1.052521	.005930	13
14	1.072823	.164425	.098339	1.438723	1.106377	1.029279	-.028298	1.056149	.004221	14
15	1.048810	.138713	.125020	1.394832	1.070970	1.019773	-.024694	1.057878	.002816	15
16	1.029980	.108462	.147089	1.360010	1.043434	1.012220	-.019793	1.058161	.001724	16
17	1.016479	.074509	.163616	1.334771	1.023813	1.006748	-.013828	1.057648	.000945	17
18	1.008367	.037934	.173856	1.319480	1.012073	1.003437	-.007109	1.057006	.000479	18
19	1.005662	-.000000	.177325	1.314359	1.008167	1.002329	.000000	1.056732	.000324	19

CP,PP TIMES FOR $ BNDRY $ ARE (SEC) .88 3.98 T = .220475 NT = 1

I,X,Y,C1,C2,C3,S ARE

1	0.00000000	-.03887562	-.08748866	12.91065925	-.00000000	0.00000000
2	.00675068	-.03828501	-.08748866	12.91065925	.00000000	.00678510
3	.01329625	-.03653114	-.08748866	12.91065925	.00000000	.01357020
4	.01943781	-.03366728	-.08748866	12.91065925	.00000000	.02035531
5	.02498877	-.02978045	-.08748866	12.91065925	-.00000000	.02714041

6	.02978045	-.02498877	-.08748866	12.91065925	-.00000000	.03392551
7	.03366728	-.01943781	-.08748866	12.91065925	.00000000	.04071061
8	.03653114	-.01329625	-.08748866	12.91065925	-.00000000	.04749572
9	.03828501	-.00675068	-.08748866	12.91065925	.00000000	.05428082
10	.03887562	-.00000000	-.08748866	12.91065925	-.00000000	.06106592
11	.03828501	.00675068	-.08748866	12.91065925	.00000000	.06785102
12	.03653114	.01329625	-.08748866	12.91065925	-.00000000	.07463613
13	.03366728	.01943781	-.08748866	12.91065925	.00000000	.08142123
14	.02978045	.02498877	-.08748866	12.91065925	-.00000000	.08820633
15	.02498877	.02978045	-.08748866	12.91065925	-.00000000	.09499143
16	.01943781	.03366728	-.08748866	12.91065925	.00000000	.10177654
17	.01329625	.03653114	-.08748866	12.91065925	-.00000000	.10856164
18	.00675068	.03828501	-.08748866	12.91065925	.00000000	.11534674
19	0.00000000	.03887562	0.00000000	0.00000000	0.00000000	.12213184

CP,PP TIMES FOR $ FINDCT $ ARE (SEC) .10 1.87 T = .220475 NT = 1

CP,PP TIMES FOR $ MAPS $ ARE (SEC) .16 1.83 T = .220475 NT = 1

CP,PP TIMES FOR $ LAXWND $ ARE (SEC) .61 2.27 T = .220475 NT = 1

NO OF BETWEEN POINTS ARE 13

LISTB

10022 10030 20022 20030 30022 30030 40023 40029 50024 50025 50026 50027 50028

ITER,DELMX4,ELMX4,RATIO,TOLSOL ARE 5 .00003357 1.59620332 .00002103 .00010000

CP,PP TIMES FOR $ BETWN $ ARE (SEC) .28 2.32 T = .220475 NT = 1

CP,PP TIMES FOR $ STBPRN $ ARE (SEC) .44 2.12 T = .220475 NT = 1

CP,PP TIME (SEC) FOR THIS TIME STEP 2.50 17.58
TOTAL SINCE LOAD 7.96 141.68

T = .220475
NT = 1

NEW TIME STEP BEGINS
****

NT	T	DT	DTFRST	DTBND	DTOUT	INDCT
2	.242904	.022429	.027565	.022429	.022907	0

CP,PP TIMES FOR $ RESET $ ARE (SEC) .02 2.62 T = .242904 NT = 2

I	WC1	WC2	WC3	WC4	PRES	TEMP	VTANG	MLOCL	CP	DW4BYDT	I
1	1.758056	.000000	-.211326	2.466527	2.252373	1.265581	.000000	.641101	.049697	35.812649	1
2	1.733451	.044574	-.204420	2.426070	2.212268	1.260978	.004845	.644905	.048106	34.191176	2
3	1.679411	.083619	-.188154	2.335995	2.119711	1.247753	.008470	.658542	.044433	30.713071	3
4	1.665612	.119518	-.167033	2.311169	2.093337	1.242610	.012001	.663724	.043386	30.473251	4
5	1.653417	.152298	-.139323	2.288341	2.065407	1.235228	.016397	.673952	.042278	30.609042	5
6	1.619684	.173774	-.106046	2.227972	2.005138	1.224594	.018808	.681477	.039886	29.301376	6
7	1.543230	.185383	-.071935	2.099585	1.872725	1.201427	.019695	.705341	.034632	25.127691	7
8	1.486591	.191105	-.038375	2.004144	1.775213	1.183086	.019710	.723282	.030762	22.518873	8
9	1.450323	.193031	-.003593	1.942145	1.710002	1.168681	.020672	.738825	.028175	21.427663	9
10	1.407206	.186801	.029247	1.866425	1.635100	1.152420	.020784	.750974	.025202	19.687842	10

11	1.343893	.174640	.056467	1.762281	1.531383	1.131241	.018813	.770450	.021087	16.589348 11
12	1.272264	.160092	.079241	1.650750	1.416536	1.106631	.015490	.800802	.016529	13.027519 12
13	1.228053	.142762	.101252	1.580774	1.345929	1.090202	.013277	.818983	.013727	11.154099 13
14	1.190289	.122305	.120583	1.520420	1.285476	1.075059	.011557	.835000	.011328	9.526315 14
15	1.158308	.099925	.135695	1.469877	1.236172	1.063071	.009217	.846623	.009372	8.142986 15
16	1.118289	.076653	.147369	1.412100	1.175155	1.047685	.006529	.870730	.006951	6.241081 16
17	1.096385	.051986	.155995	1.379904	1.142185	1.039163	.004106	.882725	.005642	5.284409 17
18	1.090393	.026258	.161257	1.369123	1.132907	1.036534	.001965	.883035	.005274	5.089609 18
19	1.090294	-.000000	.162977	1.367655	1.132592	1.036343	.000000	.881013	.005262	5.119144 19

I	WS1	WS2	WS3	WS4	PRES	TEMP	VTANG	MLOCL	CP	I
1	2.071566	.000000	-.365273	3.403157	2.822591	1.342696	.000000	.913022	.072325	1
2	2.020139	.066220	-.350024	3.289297	2.721497	1.328022	.002195	.918123	.068313	2
3	1.916484	.120016	-.315933	3.060303	2.518426	1.296374	.002464	.929281	.060255	3
4	1.901867	.173194	-.287236	3.030081	2.491824	1.292685	.003351	.930684	.059199	4
5	1.910927	.221706	-.253821	3.052777	2.512308	1.297041	.003498	.929135	.060012	5
6	1.855094	.256057	-.203726	2.935021	2.408527	1.281455	.004597	.934902	.055894	6
7	1.732078	.266442	-.149333	2.674338	2.179256	1.243162	.002249	.948946	.046796	7
8	1.646546	.273278	-.098044	2.498466	2.025242	1.216348	.000811	.959281	.040684	8
9	1.619803	.280994	-.051195	2.448786	1.982518	1.210690	-.001002	.961525	.038989	9
10	1.562165	.275452	-.004721	2.337428	1.885915	1.194959	-.003022	.967960	.035155	10
11	1.459861	.254915	.036685	2.136281	1.710770	1.161419	-.005574	.982183	.028205	11
12	1.351329	.227432	.071808	1.926315	1.528483	1.122756	-.007629	.999386	.020972	12
13	1.294207	.203127	.104581	1.820740	1.437645	1.103662	-.008495	1.008220	.017767	13
14	1.254119	.178069	.131911	1.751776	1.378753	1.093051	-.010754	1.013809	.015030	14
15	1.200658	.144759	.154898	1.654198	1.295527	1.073886	-.009433	1.022378	.011727	15
16	1.145449	.109657	.169908	1.555215	1.211213	1.053591	-.008741	1.031970	.008381	16
17	1.112202	.073217	.182048	1.495904	1.161122	1.040985	-.005878	1.037502	.006394	17
18	1.111138	.037741	.192291	1.495525	1.161324	1.042188	-.003399	1.036520	.006402	18
19	1.111709	-.000000	.196024	1.497289	1.163089	1.043222	.000000	1.035814	.006472	19

CP,PP TIMES FOR $ BNDRY $ ARE (SEC) .81 3.93 T = .242904 NT = 2

I,X,Y,C1,C2,C3,S ARE

1	0.00000000	-.04283045	-.08748866	11.71853087	-.00000000	0.00000000
2	.00743743	-.04217976	-.08748866	11.71853087	0.00000000	.00747535
3	.01464888	-.04024746	-.08748866	11.71853087	.00000000	.01495070
4	.02141522	-.03709226	-.08748866	11.71853087	-.00000000	.02242606
5	.02753088	-.03281003	-.08748866	11.71853087	.00000000	.02990141
6	.03281003	-.02753088	-.08748866	11.71853087	.00000000	.03737676
7	.03709226	-.02141522	-.08748866	11.71853087	-.00000000	.04485211
8	.04024746	-.01464888	-.08748866	11.71853087	-.00000000	.05232747
9	.04217976	-.00743743	-.08748866	11.71853087	.00000000	.05980282
10	.04283045	-.00000000	-.08748866	11.71853087	-.00000000	.06727817
11	.04217976	.00743743	-.08748866	11.71853087	.00000000	.07475352
12	.04024746	.01464888	-.08748866	11.71853087	-.00000000	.08222887
13	.03709226	.02141522	-.08748866	11.71853087	-.00000000	.08970423
14	.03281003	.02753088	-.08748866	11.71853087	.00000000	.09717958
15	.02753088	.03281003	-.08748866	11.71853087	-.00000000	.10465493
16	.02141522	.03709226	-.08748866	11.71853087	.00000000	.11213028
17	.01464888	.04024746	-.08748866	11.71853087	-.00000000	.11960564
18	.00743743	.04217976	-.08748866	11.71853087	.00000000	.12708099
19	0.00000000	.04283045	0.00000000	0.00000000	0.00000000	.13455634

CP,PP TIMES FOR $ FINDCT $ ARE (SEC) .10 1.94 T = .242904 NT = 2

```
CP,PP TIMES FOR  $ MAPS    $  ARE (SEC)        .17       1.81      T =     .242904      NT =       2

CP,PP TIMES FOR  $ LAXWND $  ARE (SEC)        .64       2.27      T =     .242904      NT =       2

NO CF BETWEEN POINTS ARE        8

LISTB
  30022   30030   40022   4003C   50023   50024   50028   50029

ITER,DELMX4,ELMX4,RATIO,TOLSOL ARE     4        .00002923       2.12301865        .00001377        .00010000

CP,PP TIMES FOR  $ BETWN  $  ARE (SEC)        .27       2.36      T =     .242904      NT =       2

CP,PP TIMES FOR  $ STBPRN $  ARE (SEC)        .45       1.97      T =     .242904      NT =       2

CP,PP TIME (SEC) FOR THIS TIME STEP       2.45      16.91
                 TOTAL SINCE LOAD        10.41     158.59
T =     .242904
NT =       2

NEW TIME STEP BEGINS
**********************

        NT          T          DT      DTFRST      DTBND      DTOUT      INDCT
         3    .264850    .021946    .027565    .021946    .022718          0

CP,PP TIMES FOR  $ RESET  $  ARE (SEC)        .02       2.52      T =     .264850      NT =       3

 I         WC1          WC2          WC3          WC4         PRES         TEMP        VTANG        MLOCL           CP      DW4BYDT  I
 1    2.143098     .000000    -.273522     3.276797     2.989961     1.374216     .000000     .653242     .078967    36.920807  1
 2    2.124655     .061700    -.263236     3.238956     2.956117     1.370611     .007085     .652176     .077624    37.040035  2
 3    2.058297     .112720    -.235414     3.106089     2.832635     1.356279     .012343     .653317     .072724    35.090156  3
 4    1.917855     .151056    -.203385     2.863549     2.579452     1.326824     .015187     .688083     .062677    25.169739  4
 5    1.877617     .188036    -.168045     2.786429     2.496049     1.311851     .019187     .703590     .059367    22.695883  5
 6    1.828229     .216289    -.126199     2.696636     2.399372     1.295632     .023167     .722003     .055531    21.355188  6
 7    1.748425     .228260    -.081380     2.546183     2.251798     1.272340     .024967     .737250     .049675    20.349736  7
 8    1.737010     .235970    -.034655     2.506216     2.219155     1.262270     .027715     .733269     .048379    22.877421  8
 9    1.684947     .233313     .007841     2.405670     2.119814     1.243645     .028628     .745419     .044437    21.121010  9
10    1.620721     .222676     .046994     2.287714     2.002381     1.222115     .028995     .762118     .039777    19.196442 10
11    1.540547     .203482     .079569     2.141861     1.862141     1.196755     .027929     .777853     .034212    17.295974 11
12    1.444557     .178536     .102929     1.967864     1.699169     1.166004     .024685     .792691     .027745    14.449628 12
13    1.336474     .155587     .119970     1.790259     1.521707     1.130466     .019532     .829576     .020703     9.545389 13
14    1.275034     .131598     .137437     1.687435     1.420836     1.107526     .016229     .850842     .016700     7.610240 14
15    1.226087     .106079     .151936     1.607393     1.342450     1.089168     .013377     .868892     .013589     6.266073 15
16    1.178344     .079406     .160505     1.527004     1.267748     1.071240     .009747     .880979     .010625     5.235710 16
17    1.164642     .053026     .167337     1.498349     1.245895     1.065463     .006357     .876114     .009758     5.397071 17
18    1.155770     .026986     .172429     1.483692     1.231858     1.061744     .002913     .879293     .009201     5.220434 18
19    1.153767    -.000000     .174548     1.481030     1.228595     1.060815     .000000     .881311     .009071     5.166069 19

 I         WS1          WS2          WS3          WS4         PRES         TEMP        VTANG        MLOCL           CP  I
 1    2.599559     .000000    -.458372     4.624906     3.913774     1.479542     .000000     .869774     .115626  1
 2    2.573803     .092096    -.444593     4.574561     3.869350     1.477546     .005243     .870746     .113863  2
 3    2.467767     .167814    -.401981     4.334688     3.654900     1.456308     .008189     .877630     .105353  3
 4    2.207677     .210931    -.327713     3.705336     3.089130     1.377734     .008523     .902390     .082902  4
 5    2.150021     .261810    -.275204     3.561772     2.958526     1.355333     .011005     .910524     .077719  5
```

6	2.092913	.298522	-.218355	3.438177	2.849004	1.341251	.011762	.915543	.073373	6
7	1.984896	.315164	-.154102	3.206496	2.644517	1.313696	.012155	.925235	.065259	7
8	2.009004	.339641	-.102577	3.280562	2.714139	1.331875	.009843	.918150	.068021	8
9	1.926638	.337596	-.041760	3.098854	2.553413	1.307355	.009082	.926507	.061643	9
10	1.827795	.322290	.015828	2.885049	2.364697	1.277216	.008659	.937262	.054155	10
11	1.711698	.295263	.063581	2.646113	2.155942	1.244747	.006627	.948334	.045871	11
12	1.569764	.258859	.098076	2.360593	1.907420	1.202628	.002310	.964910	.036009	12
13	1.414634	.215937	.124863	2.048549	1.635162	1.146291	.000117	.988150	.025205	13
14	1.341815	.181181	.152158	1.907931	1.513348	1.119687	.000074	.999320	.020371	14
15	1.285728	.147212	.172421	1.807023	1.427105	1.102955	-.001509	1.007412	.016949	15
16	1.217249	.108981	.184918	1.682096	1.320069	1.078967	-.001579	1.018552	.012701	16
17	1.200218	.075972	.197561	1.655546	1.298005	1.076344	-.003183	1.019918	.011826	17
18	1.183297	.038720	.205038	1.625158	1.272120	1.070327	-.002136	1.022689	.010796	18
19	1.178125	-.000000	.207735	1.615729	1.264097	1.069358	.000000	1.023556	.010480	19

CP,PP TIMES FOR $ BNDRY $ ARE (SEC) .89 3.80 T = .264850 NT = 3

I,X,Y,C1,C2,C3,S ARE

1	0.00000000	-.04670015	-.08748866	10.74750172	-.00000000	0.00000000
2	.00810940	-.04599067	-.08748866	10.74750172	.00000000	.00815075
3	.01597239	-.04388378	-.08748866	10.74750172	.00000000	.01630149
4	.02335007	-.04044351	-.08748866	10.74750172	.00000000	.02445224
5	.03001828	-.03577439	-.08748866	10.74750172	.00000000	.03260298
6	.03577439	-.03001828	-.08748866	10.74750172	-.00000000	.04075373
7	.04044351	-.02335007	-.08748866	10.74750172	.00000000	.04890447
8	.04388378	-.01597239	-.08748866	10.74750172	-.00000000	.05705522
9	.04599067	-.00810940	-.08748866	10.74750172	.00000000	.06520596
10	.04670015	-.00000000	-.08748866	10.74750172	-.00000000	.07335671
11	.04599067	.00810940	-.08748866	10.74750172	.00000000	.08150745
12	.04388378	.01597239	-.08748866	10.74750172	.00000000	.08965820
13	.04044351	.02335007	-.08748866	10.74750172	-.00000000	.09780894
14	.03577439	.03001828	-.08748866	10.74750172	.00000000	.10595969
15	.03001828	.03577439	-.08748866	10.74750172	-.00000000	.11411043
16	.02335007	.04044351	-.08748866	10.74750172	.00000000	.12226118
17	.01597239	.04388378	-.08748866	10.74750172	-.00000000	.13041192
18	.00810940	.04599067	-.08748866	10.74750172	.00000000	.13856267
19	0.00000000	.04670015	0.00000000	0.00000000	0.00000000	.14671341

CP,PP TIMES FOR $ FINDCT $ ARE (SEC) .10 1.82 T = .264850 NT = 3

CP,PP TIMES FOR $ MAPS $ ARE (SEC) .17 1.81 T = .264850 NT = 3

CP,PP TIMES FOR $ LAXWND $ ARE (SEC) .65 2.06 T = .264850 NT = 3

NO CF BETWEEN POINTS ARE 17

LIST8

10021 10031 20021 20031 30021 30031 40022 40030 50022 50023 50029 50030 60024 60025 60026
60027 60028

ITER,DELMX4,ELMX4,RATIO,TOLSOL ARE 6 .00003788 2.67878442 .00001414 .00010000

CP,PP TIMES FOR $ BETWN $ ARE (SEC) .30 2.39 T = .264850 NT = 3

CP,PP TIMES FOR $ STBPRN $ ARE (SEC) .45 1.90 T = .264850 NT = 3

CP,PP TIME (SEC) FOR THIS TIME STEP 2.56 16.29

TOTAL SINCE LOAD 12.97 174.88

T = .264850
NT = 3

INFORMATION SAVED , T,NT,ISVFP ARE .264850 3 1

RUN TERMINATED BECAUSE NT.GT.NTMX 3 3

8.0 PROGRAM LISTING

The following pages contain a listing of the computer program.

```
      OVERLAY(OVER,0,0)
      PROGRAM TEA270D(INPUT=1002,OUTPUT,TAPE5=INPUT,TAPE6=OUTPUT,
     *                TAPE1=1002,TAPE2,TAPE3,TAPE4,TAPE7,TAPE10)

C UNIFIED SUPERSONIC-HYPERSONIC SMALL DISTURBANCE THEORY
C NASA CONTRACT NAS1-9562  PHASE 2

      COMMON / CM1    / TITLE(8),  DATE1,DATE2
      COMMON / CM2    / ZNX,ZNY,YZ,DX,DY,DS,DN,   RDX,RDY,RDS,RDN
      COMMON / CM3    / UNIFY,FSMN,GAMMA,ALFAC
      COMMON / CM3A   / CT1,CT2,CT3,CT4,CT5,CT6,CT7,CT8,CT9
      COMMON / CM3B   / ALFAR,W1FS,W2FS,W3FS,W4FS
      COMMON / CM4    / CF,DTMX, DXYSN, DTFRST,DTOUT,DTBND,DTSTB,DT
      COMMON / CM4A   / ICT,ICTFP,T1,T2,NC1,NCURV
      COMMON / CM4B   / TPREV,T,INDCT,NT,NTFRST
      COMMON / CM5    / BND,BNDISC,BNDPRN
      COMMON / CM6    / ARTVS,GAMB
      COMMON / CM7    / TOLS,TOLX,TOLCB,TOLCF,TOLSOL
      COMMON / CM8A   / NCTMX,NBTMX,NX,NY,KX,KY,KW,NBT,MM
      COMMON / CM8B   / MS,ML,KBLKS
      COMMON / CM10   / IOCTL,TMX,NTMX,TMR
      COMMON / CM10A  / TIMA(2),TIMB(2),TIMS(2),TIMT(2),TIM(2,8)
      COMMON / CM11   / IR,Iw
      COMMON / CM11A  / ISV,   ISVFP
      COMMON / CM11B  / ID1,ID2,IL1,IL2,   ID1FP,ID2FP,IL1FP,IL2FP
      COMMON / CM12   / IALZRS,IALWNS,MSK(60),MSKRGT(60)
      COMMON / CM13   / ISTART,ISTOP,ISAVE,NSAVE
      COMMON / CM14   / NPRNT,IPRNT,NBUG,IBUG
      COMMON / CM999  / NN,A(7000)   $   NN = 7000

      IR = 5
      Iw = 6
      TIMA(1) = TIMA(2) = 0.
      TMR = 2.

C PRINT OUT COMPILE TIMES
      CALL TIMER(TMR,TIMA,TIMB,TIMT)
      WRITE(Iw,6CCC) TIMT
```

```
C PRIMARY OVERLAY 1   *INIT*    INITIALISES THE PROGRAM
      CALL OVERLAY(4HOVER,1,0,0)



    2 CONTINUE
C PRIMARY OVERLAY 2   *RESET*   RESETS THE LOOP
      CALL OVERLAY(4HOVER,2,0,0)

C PRIMARY OVERLAY 3   *BNDRY*   COMPUTES *W* ON BOUNDARY
      IF(BND.EQ.1.)CALL OVERLAY(4HOVER,3 ,0,0)
      IF(BND.EQ.3.)CALL OVERLAY(4HOVER,27,0,0)

C PRIMARY OVERLAY 4   *FINDCT*  FINDS NEW CONTOUR
      CALL OVERLAY(4HOVER,4,0,0)

C PRIMARY OVERLAY 5   *MAPS*    DETERMINES OUTSIDE,LAXWND,BETWN POINTS
      CALL OVERLAY(4HOVER,5,0,0)

C PRIMARY OVERLAY 6   *LAXWND*  COMPUTES *W* IN FLOW FIELD
      CALL OVERLAY(4HOVER,6,0,0)

C PRIMARY OVERLAY 7   *BETWN*   SOLVES FOR *W* AT *BETWEEN* POINTS
      CALL OVERLAY(4HOVER,7,0,0)

C PRIMARY OVERLAY 8   *STBPRN*  FINDS *DTOUT* PRINTS *W*
      CALL OVERLAY(4HOVER,8,0,0)

      DO 90 I=1,2
      TIMT(I)=TIMB(I)-TIMS(I)
   90 TIMS(I)=TIMB(I)

      WRITE(Iw,6090) TIMT,TIMS,T,NT
      GO TO 2


 6000 FORMAT(1H1,*CP,PP TIMES TO COMPILE AND LOAD ARE (SEC)*2F10.2)
```

```
6090 FORMAT(1HO,*CP,PP TIME (SEC) FOR THIS TIME STEP* 2F10.2 /
     $        1X,*                    TOTAL SINCE LOAD  * 2F10.2 /
     $1X,*T = * F10.6 /1X,*NT = * I6 )

      END
```

```
      SUBROUTINE ALL (N,X,Y,D,CS,SN,C1,C2,C3,S,TOLS,TOLX,
     *                ARC,J,S1,S2,SGN,
     *                COSN,SINN,CURV,XREF,YREF )
      DIMENSION X(1),Y(1),D(1),CS(1),SN(1),C1(1),C2(1),C3(1),S(1)
      CALL POSCON ( N,S, ARC,    J,SS,S1,S2,SGN )
      CALL XYCORS ( 0.,D(J),C1(J),C2(J),C3(J),1,TOLS,TOLX,SS,XL,
     *             X(J),Y(J),CS(J),SN(J),XREF,YREF )
      CALL NRMCUR ( XL,CS(J),SN(J),C1(J),C2(J),C3(J),COSN,SINN,CURV)
      RETURN
      END
```

```
      SUBROUTINE CUBARC(A,B,C1,C2,C3,EPS,ARC)

C FINDS ARC LENGTH OF CUBIC ( C1*X + C2*X**2 + C3*X**3 )  FROM  A  TO  B

      COMMON / CUBCOF / C1Z,TWOC2,THRC3
      COMMON / CM11   / IR,IW
      EXTERNAL DSBYDX
      C1Z = C1
      TWOC2 = C2 + C2
      THRC3 = 3.*C3
      CALL GENDRE ( DSBYDX, A, B, EPS, 0, ARC, IERR )

      IF ( IERR.EQ.0 ) GO TO 99
      WRITE(IW,6) A,B,C1,C2,C3,EPS,ARC,IERR
    6 FORMAT(1HO,*LEGEND CANNOT MEET REQUIRED TOLERANCE*/,
     *1X,*A,B,C1,C2,C3,EPS,ARC,IERR ARE*/,
     *1X,5F25.15/,1X,2F25.15,I10)

   99 RETURN
      END
```

0020 CARDS

```
      SUBROUTINE DSBYDX ( X, FX )

C FINDS DS/DX FOR CUBIC  C1*X +C2*X**2 + C3*X**3

      COMMON / CUBCOF / C1,TWOC2,THRC3
      F = C1 + X*( TWOC2 + THRC3*X )
      FX = SQRT ( 1. + F*F )
      RETURN
      END
```

```
      SUBROUTINE ENDFL(ICFP,ICUNIT)
      COMMON / CM11   / IR,IW
      IF( IOFP.GE. 1) GO TO 10
      WRITE(IW,6) IOFP,ICUNIT
    6 FORMAT(1HO,* ERROR IN SUBROUTINE ENDFL   IOFP,IOUNIT ARE * 2I4 )
      CALL EXIT
   10 ENDFILE IOUNIT
      IOFP = IOFP + 1
      RETURN
      END
```

```
      SUBROUTINE GENMSK

C GENERATES MASKS

      COMMON / CM12    / IALZRS,IALWNS,MSK(60),MSKRGT(60)
      MSK(1) = 1
      DO 10 I=2,59
   10 MSK(I) = MSK(I-1)+MSK(I-1)

      MSKRGT(1) = 1
      DO 20 I=2,59
   20 MSKRGT(I) = MSK(I).OR.MSKRGT(I-1)

      MSK(60) = .NOT.MSKRGT(59)
      MSKRGT(60) = MSK(60).OR.MSKRGT(59)
      IALZRS = 0
      IALWNS = MSKRGT(60)

      RETURN
      END
```

```
      SUBROUTINE GENDKE (AUX,A,B,CONTR,KODE,FINTL,IRROR)                LEG 0010
      DIMENSION ARG(12),WGHT(12)                                        LEG 0160
      DATA ARG,WGHT/.960289856497536,.796666477413627,.525532409916329, LEG 0170
     C.183434642495650,.989400934991650,.944575023073233,              LEG 0180
     C.865631202387832,.755404408355003,.617876244402644,              LEG 0190
     C.458016777657227,.281603550779259,.095012509837637,              LEG 0200
     C.101228536290376,.222381034453374,.313706645877887,              LEG 0210
     C.362683783378362,.027152459411754,.062253523938647,              LEG 0220
     C.095158511682493,.124628971255534,.149595988816577,              LEG 0230
     C.169156519395003,.182603415044924,.189450610455068/              LEG 0240
      DIMENSION TOP(3),BOT(3),SUM(3)                                    LEG 0250
      TRAFO(Y)=(D+C)/2.+(D-C)*Y/2.                                      LEG 0310
      TOP(1)=B                                                          LEG 0370
      TOP(2)=(B+A)/2.                                                   LEG 0380
      TOP(3)=B                                                          LEG 0390
      BOT(1)=A                                                          LEG 0400
      BOT(2)=A                                                          LEG 0410
      BOT(3)=TOP(2)                                                     LEG 0420
      L1=1                                                              LEG 0480
      L2=4                                                              LEG 0490
10    DO 30 I=1,3                                                       LEG 0560
      C=BOT(I)                                                          LEG 0570
      D=TOP(I)                                                          LEG 0580
      SUM(I)=0.                                                         LEG 0590
      DO 20 L=L1,L2                                                     LEG 0600
      Y=-ARG(L)                                                         LEG 0650
      W=WGHT(L)                                                         LEG 0660
      X=TRAFO(Y)                                                        LEG 0670
      CALL AUX(X,FX)                                                    LEG 0680
      SUM(I)=SUM(I)+FX*W                                                LEG 0690
      Y=ARG(L)                                                          LEG 0760
      X=TRAFO(Y)                                                        LEG 0770
      CALL AUX(X,FX)                                                    LEG 0780
      SUM(I)=SUM(I)+FX*W                                                LEG 0790
20    CONTINUE                                                          LEG 0800
30    SUM(I)=SUM(I)/2.*(D-C)                                            LEG 0860
33    IF (KODE) 36,40,36                                                LEG 0930
36    IF(ABS ((SUM(1)-SUM(2)-SUM(3))/SUM(1))-CONTR) 70,70,50            LEG 1000
```

```
40    IF (ABS (SUM(1)-SUM(2)-SUM(3))-CONTR) 70,70,50
50    IF(L1-1) 60,55,60
55    L1=5
      L2=12
      GO TO 10
60    IRROR=1
      FINTL=SUM(1)
      RETURN
70    IRROR=0
      FINTL=SUM(L)
      RETURN
      END
```

```
LEG 1070
LEG 1080
LEG 1150
LEG 1160
LEG 1170
LEG 1230
LEG 1240
LEG 1250
LEG 1260
LEG 1270
LEG 1280

          0050 CARDS
```

```
      FUNCTION MAPCHK( MAP,Kw,NX,J,I )

C MAPCHK IS ZERO IF BIT POSITION (J,I) OF MAP IS ZERO

      DIMENSION MAP(Kw,NX)
      COMMON / CM12   / IALZRS,IALWNS,MSK(60),MSKRGT(60)
      J1 = J - 1
      IWD = 1 + J1/60
      IBIT = 60 - MOD(J1,60)
      MAPCHK = MSK(IBIT).AND.MAP(IWD,I)
      RETURN
      END
```

```
      SUBROUTINE NLCSP(X,Y,N,N1,SS1,N2,SS2,TOL,D,CS,SN,A1,A2,A3,A4,A5)
      DIMENSION X(1),Y(1),D(1),CS(1),SN(1),A1(1),A2(1),A3(1),A4(1),A5(1)

C     STATEMENT FUNCTIONS...

      TB(I)=(A1(I+1)+A2(I+1))/(1.-A1(I+1)*A2(I+1))
      CA(I)=-(4.*A2(I)+2.*TB(I))/(D(I)*SQRT((1.+A2(I)**2)**3))
      CB(I)= (2.*A2(I)+4.*TB(I))/(D(I)*SQRT((1.+TB(I)**2)**3))
C
      COMMON / CM11   / IR,IW
C     INITIALIZE...

      S1 = SS1
      S2 = SS2
      NM1=N-1
      DO 1 I=1,NM1
      D(I)=SQRT((X(I+1)-X(I))**2+(Y(I+1)-Y(I))**2)
      CS(I)=(X(I+1)-X(I))/D(I)
      SN(I)=(Y(I+1)-Y(I))/D(I)
      IF(I.EQ.1) GO TO 1
      SD=SN(I)*CS(I-1)-CS(I)*SN(I-1)
      CD=CS(I)*CS(I-1)+SN(I)*SN(I-1)
      A1(I)=SD/CD
      A2(I)=-SD/(1.+CD)
    1 CONTINUE
      A1(N)=0.
      A2(1)=0. $ D1=4./D(1) $ F1=0.5*D1
      IF(N1.EQ.2) GO TO 2
      S1 = TAN( S1 - ATAN2(SN(1),CS(1)))
      A2(1)=S1 $ D1=1. $F1=0.
    2 A2(N)=0. $ DN=4./D(NM1) $ EN=0.5*DN
      IF(N2.EQ.2)  GO TO 3
      S2 = TAN(S2  - ATAN2(SN(NM1),CS(NM1)))
      A2(N)=S2 $ DN=1. $ EN=0.
    3 CONTINUE
C
C     SOLVE FOR THE SLOPES USING A QUASI-NEWTON METHOD...
```

```
      ITMX=20+N/2
      DO 40 IT=1,ITMX
C     EVALUATE THE RESIDUAL VECTOR AND ITS NORM
      ER=0.
      DO 10 I=1,NM1
      IF(I.EQ.1) GO TO 8
      A3(I)=CB(I-1)-CA(I)
      RI=ABS(A3(I))
      GO TO 9
    8 A3(1)=A2(1)-S1
      IF(N1.EQ.2) A3(1)=S1-CA(1)
      RI=ABS(A3(1))
      A3(N)=A2(N)-S2
      IF(N2.EQ.2)A3(N)=CB(NM1)-S2
      RN=ABS(A3(N))
      IF(RN.GT.RI) RI=RN
    9 IF(RI.GT.ER) ER=RI
   10 CONTINUE
      IF(ER.LE.TOL) GO TO 50

C     SOLVE A TRI-DIAGONAL SET OF EQUATIONS FOR THE NEXT STEP

      A4(1)=F1/D1
      A5(1)=A3(1)/D1
      DO 20 I=2,N
      EI=EN $ DI=DN $ FI=0.
      IF(I.EQ.N) GO TO 19
      EI=2./D(I-1) $ FI=2./D(I) $ DI=2.*(EI+FI)
   19 T=DI-EI*A4(I-1)
      A4(I)=FI/T
      A5(I)=(A3(I)-EI*A5(I-1))/T
   20 CONTINUE
      A2(N)=A2(N)-A5(N)
      DO 30 J=2,N
      I=N+1-J
      A5(I)=A5(I)-A4(I)*A5(I+1)
      A2(I)=A2(I)-A5(I)
   30 CONTINUE
```

```
   40 CONTINUE
      WRITE(IW,100) IT,ER,(A2(I),I=1,N)
      WRITE(IW,101)(A3(I),I=1,N)
      WRITE(IW,150) ITMX
      CALL EXIT
   50 CONTINUE
C
C     COMPUTE THE SPLINE COEFFICIENTS...

      DO 60 I=1,NM1
      SA=A2(I)
      SB=TB(I)
      RDI = 1./D(I)
      A1(I)=SA
      A2(I)=-(2.*SA+SB)*RDI
      A3(I)= (SA+SB)*RDI*RDI
   60 CONTINUE
C
  100 FORMAT(22H1NUMBER OF ITERATIONS=I3/32H MAXIMUM DEVIATION IN CURVAT
     *URE=E11.4///26H THE SOLUTION VECTOR IS.../(5E24.10))
  101 FORMAT(//26H THE RESIDUAL VECTOR IS.../(5E24.10))
  150 FORMAT(61H1SUBROUTINE NLCSPL FAILED TO CONVERGE TO A CUBIC SPLINE
     *AFTER,I3,10HITERATIONS)
      RETURN
      END
```

```
      SUBROUTINE POSCON (N,S,  ARC, J,SS,S1,S2,SGN )
C FINDS POSITION ON CONTOUR OF POINT SPECIFIED BY ARC LENGTH *ARC*
      DIMENSION S(1)
      SGN = 1.
      IF(ARC.LT.0.)GO TO 220
      J = 1
  110 J = J + 1
      IF(J.GT.N ) GO TO 120
      IF(ARC.GT.S(J))  GO TO 110
      J = J - 1
      SS = ARC - S(J)
      GO TO 300
  120 SGN = - 1.
      J = N
      ARD = 2.*S(N) - ARC
  130 J = J - 1
      IF(ARD.LT.S(J))  GO TO 130
      SS = ARD - S(J)
      GO TO 300
  220 SGN = -1.
      J = 1
      ARD = -ARC
  230 J = J + 1
      IF(ARD.GT.S(J))  GO TO 230
      J = J - 1
      SS = ARD - S(J)
  300 S1 = SS/(S(J+1)-S(J))
      S2   = 1. - S1
      RETURN
      END
```

```
      SUBROUTINE POSITN(IFIL1,IFIL2,IOUNIT,IND)

C *IOUNIT* IS POSITIONED IN *IFIL1* ON ENTERING THIS SUBROUTINE
C *IOUNIT* IS POSITIONED AT START OF *IFIL2* ON LEAVING THIS SUBROUTINE
C IF *IND* IS POSITIVE *BSF* WILL NOT BE USED

      COMMON / CM11   / IR,IW
      IF(IFIL2.LE.0) GO TO 420
      IF(IFIL1.LE.0) GO TO 110
      IFILD = IFIL2 - IFIL1
      IF(IFILD) 100,200,300

  100 IF(IND.GE.0) GO TO 110
      CALL BSF(-IFILD,IOUNIT,IERR)
      GO TO 400

  110 REWIND IOUNIT
      CALL FSF(IFIL2,IOUNIT,IERR)
      GO TO 400

  200 IF(IND.GE.0) GO TO 110
      CALL BSF(1,IOUNIT,IERR)
      CALL FSF(1,IOUNIT,IERR)
      GO TO 400

  300 CALL FSF(IFILD,IOUNIT,IERR)
  400 IF(IERR.EQ.0) GO TO 500

  420 WRITE(IW,440) IFIL1,IFIL2,IOUNIT,IND,IERR
  440 FORMAT(1HO,*ERROR IN POSITN  IFIL1,IFIL2,IOUNIT,IND,IERR ARE * /
     *1X,5I10)
      CALL EXIT

  500 IFIL1 = IFIL2
      RETURN
      END
```

```
      SUBROUTINE TIMER(TMR,TIMA,TIMB,TIM)

C TIMB(1),TIMB(2) ARE CP,PP TIME ELAPSED SINCE LOAD IN SECONDS
C IF TMR.EQ.2 CP,PP TIMES ARE COMPUTED , OTHERWISE CP TIME ONLY

      DIMENSION TIMA(2),TIMB(2),TIM(2),BLOCK(3)
      DATA I1,I2,I3 / 7777B, 7777777700000B, 100000B /

      CALL SECOND(TIMB(1))
      TIM(1) = TIMB(1) - TIMA(1)
      IF( TMR.NE.2.) GO TO 10

      CALL CORE( BLOCK,19,2,1 )
      MILSEC = BLOCK(3).AND.I1
      ISEC =   BLOCK(3).AND.I2
      TIMB(2) = FLOAT(ISEC/I3) + .001*FLOAT(MILSEC)
      TIM(2) = TIMB(2) - TIMA(2)
      RETURN
   10 TIMB(2) = TIM(2) = 0.
      RETURN
      END
```

```
      SUBROUTINE XYCORS ( XP, XQ, C1, C2, C3,
     *                    INC,TOLS,TOLX,ARCIN,XLOUT,
     *                    XC, YC, CS, SN, XREF, YREF )

C FINDS X POSITION IN LOCAL AXES  AND (X,Y) IN REFERENCE AXES
C     CORRESPONDING TO ARC LENGTH .ARCIN.
C IF IND=1 *ARCIN* HAS BEEN MEASURED FROM *XP*
C IF IND=2 *ARCIN* HAS BEEN MEASURED FROM *XQ*

      COMMON / CM11   / IR,IW

      EPX = TOLX*(XQ-XP)
      EPS = TOLS*(XQ-XP)
      CALL CUBARC( XP,XQ,C1,C2,C3,EPS,STOT )
      X1 = XP
      X2 = XQ
      IF (IND.EQ.2) GO TO 5
      SA = ARCIN
      F1 = -SA
      F2 = STOT - SA
      GO TO 10
    5 SA = STOT - ARCIN
      F1 = -SA
      F2 = STOT - SA

   10 X3 = (X1*F2 - X2*F1)/(F2 - F1)
      IF ( ABS(X3-X1).LE.EPX ) GO TO 60
      IF ( ABS(X3-X2).LE.EPX ) GO TO 60
      CALL CUBARC ( XP, X3, C1, C2, C3, EPS, F3 )
      F3 = F3 - SA
      IF(F1*F3.GT.0.) GO TO 30
      X2 = X3
      F2 = F3
      GO TO 10

   30 IF(F2*F3.GT.0.) GO TO 50
      X1 = X3
      F1 = F3
```

```
      GO TO 10

   50 CONTINUE
      WRITE(IW,55) IND,XP,XQ,C1,C2,C3,TOLS,TOLX,ARCIN,X1,X2,X3,F1,F2,F3,
     *STCT
   55 FORMAT(1HO,* ERROR IN XYCORS * / 1X,I10 /(1X,5E25.10))
      CALL EXIT
   60 XLOUT = X3
      YLOUT = XLOUT*(C1 + XLOUT*(C2 + XLOUT*C3))
      XREF = XC + XLOUT*CS - YLOUT*SN
      YREF = YC + XLOUT*SN + YLOUT*CS
      RETURN
      END
```

```
      OVERLAY(OVER,1,0)
      PROGRAM INIT

C OVERLAY(1,0)  READS,PRINTS DATA. INITIALISES. SLICES ARRAY A

      COMMON / CM1    / TITLE(8),  DATE1,DATE2
      COMMON / CM2    / ZNX,ZNY,YZ,DX,DY,DS,DN,   RDX,RDY,RDS,RDN
      COMMON / CM2A   / MDY,AMDY
      COMMON / CM3    / UNIFY,FSMN,GAMMA,ALFAC
      COMMON / CM3A   / CT1,CT2,CT3,CT4,CT5,CT6,CT7,CT8,CT9
      COMMON / CM3B   / ALFAR,W1FS,W2FS,W3FS,W4FS
      COMMON / CM4    / CF,DTMX, DXYSN, DTFRST,DTOUT,DTBND,DTSTB,DT
      COMMON / CM4A   / ICT,ICTFP,T1,T2,NC1,NCURV
      COMMON / CM4B   / TPREV,T,INDCT,NT,NTFRST
      COMMON / CM5    / BND,BNDISC,BNDPRN
      COMMON / CM6    / ARTVS,GAMB
      COMMON / CM7    / TOLS,TOLX,TOLCB,TOLCF,TOLSOL
      COMMON / CM8A   / NCTMX,NBTMX,NX,NY,KX,KY,KW,NBT,MM
      COMMON / CM8B   / MS,ML,KBLKS
      COMMON / CM10   / IOCTL,TMX,NTMX,TMR
      COMMON / CM10A  / TIMA(2),TIMB(2),TIMS(2),TIMT(2),TIM(2,8)
      COMMON / CM11   / IR,IW
      COMMON / CM11A  / ISV,   ISVFP
      COMMON / CM11B  / ID1,ID2,IL1,IL2,   ID1FP,ID2FP,IL1FP,IL2FP
      COMMON / CM12   / IALZRS,IALWNS,MSK(60),MSKRGT(60)
      COMMON / CM13   / ISTART,ISTOP,ISAVE,NSAVE
      COMMON / CM14   / NPRNT,IPRNT,NBUG,IBUG
      COMMON / CM999  / NN,A(1)

      PI=3.14159265358979
      CALL DATE ( DATE1,DATE2 )

C READ IN DATA

      READ(IR,510) TITLE,
     2            ZNX,ZNY,AMDY,DX,DY,DS,DN,
     3              UNIFY,FSMN,GAMMA,ALFAC,
     5               BND,BNDISC,BNDPRN,
```

```
     8              ZNCTMX,ZNBTMX,ZKX,ZKY
      READ(IR,515) ZNPRNT,ZNBUG,ZNSAVE
      READ(IR,515) TMX,ZNTMX
      READ(IR,520) ICCTL
      CF = 1.
      CTMX = 9999.
      TCLS = .00005
      TCLX = .001
      TCLCB= .001
      TCLCF= 1.E-08
      TCLSCL = .001

C PRINT DATA

      WRITE(IW,610) TITLE,DATE1,DATE2,
     2            ZNX,ZNY,AMDY,DX,DY,DS,DN,
     3               UNIFY,FSMN,GAMMA,ALFAD,
     5                BND,BNDISC,
     8              ZNCTMX,ZNBTMX,ZKX,ZKY
      WRITE(IW,618) ZNPRNT,ZNBUG,ZNSAVE
      WRITE(IW,620) TMX,ZNTMX
      WRITE(IW,622) IUCTL

C COMPUTE SOME CONSTANTS

      IF(NBUG .EQ.0) NBUG =9999
      IF(NSAVE.EQ.0) NSAVE=9999
      IF(NPRNT.EQ.0) NPRNT=9999

      IF(UNIFY.LE.0.) GO TO 10

C UNIFIED THEORY USED

      B2 = FSMN*FSMN - 1.
      B  = SQRT(B2)
      GO TO 20
   10 CONTINUE
```

```
C UNIFIED THEORY NOT USED

      B = FSMN
      B2 = B*B
   20 G  = GAMMA
      CT1= 1./(G*B2)
      CT2= G - 1.
      CT3= -.5*G*CT2*B2
      CT4= 1./B
      CT5 = G
      CT6 = B
      CT7 = FSMN
      CT8 = 2./(G*B2)
      CT9 = (FSMN/B)**2
      RDX = 1./DX
      RDY = 1./DY
      RDS = 1./DS
      RDN = 1./DN
      DXYSN = AMIN1(DX,DY,DS,DN)/SQRT(2.)
      DXYSN = CF*DXYSN

C CONVERT TO INTEGERS

      NPRNT = INT(ZNPRNT)
      NBUG = INT(ZNBUG)
      NBTMX = INT(ZNBTMX)
      NCTMX = INT(ZNCTMX)
      NX = INT(ZNX)
      NY = INT(ZNY)
      KX    = INT(ZKX)
      KY    = INT(ZKY)
      KW = 1 + (NY-1)/60
      NTMX = INT(ZNTMX)
      NSAVE = INT(ZNSAVE)
      YZ = AMDY*DY
      MDY= INT(AMDY)

      CALL CHECK
```

```
      CALL GENMSK

C SET FREE STREAM VALUES

      W1FS = 1.
      W2FS = 0.
      ALFAR = ALFAD*PI/180.
      TANAL = TAN(ALFAR)
      W3FS  = TANAL
      W4FS  = 1. - CT3*(TANAL)**2
      RR = 1./W1FS
      U = RR*W2FS
      V = RR*W3FS
      VSQ = U*U + V*V
      VEL = SQRT(VSQ)
      TEMP = RR*W4FS + CT3*VSQ
      APLUSV = CT4*SQRT(TEMP) + VEL
      DTFRST = DXYSN/APLUSV

      ICT = 1
      IC1 = 2
      IC2 = 3
      IL1 = 4
      IL2 = 7
      ISV = 10

      REWIND IC1
      ENDFILE IC1
      IC1FP = 1

      REWIND IC2
      ENDFILE IC2
      IC2FP = 1

      REWIND IL1
      ENDFILE IL1
      IL1FP = 1
```

```
      REWIND IL2
      ENDFILE IL2
      IL2FP = 1


      NC = NCTMX
      N1 = 1
      N2 = N1 + NC
      N3 = N2 + NC
      N4 = N3 + NC
      N5 = N4 + NC
      N6 = N5 + NC
      N7 = N6 + NC
      N8 = N7 + NC
      N9 = N8 + NC
      N10= N9 + NC
      N11= N10+ NC
      N12= N11+ NC
      N13= N12+ NC
      N14= N13+ NC
      N15= N14+ NC
      N16= N15+ NC
      N17= N16+ NC
      N18= N17+ NC
      N19= N18+ NC
      N20= N19+ NC
      N21= N20+ NC
      N22= N21+ NC
      N23= N22+ NC
      N24= N23+ NC
      N25= N24+ NC
      CALL BODGOM(A(N1),A(N2),A(N3),A(N4),A(N5),A(N6),A(N7),A(N8),A(N9),
     *A(N10),A(N11),A(N12),A(N13),A(N14),A(N15),A(N16),A(N17),A(N18),
     *A(N19),A(N20),A(N21),A(N22),A(N23),A(N24),A(N25))

      ISAVE = -1
      ISTOP = -1
      READ(IR,515) ASTART
```

```
      ISTART =INT(ASTART)
      IF(ISTART.EQ.1) GO TO 100
      N8A= N8 + NC
      N9 = N1
      N10 = N2
      N11= N3
      N12= N4
      N13= N1
      N14= N1
      N15= N14 + NX
      N16= N15 + NX*KX
      N17= N16 + NY
      N18= N17 + NY*KY
      N19= N18 + NX*KX
      N20 = 1
      N21 = 1
      N22 = N21 + MM
      N23 = N22 + MM
      N24 = N23 + MM
      N25 = N24 + NC
      CALL START2 ( A(N1),A(N2),A(N3),A(N4),A(N5),A(N6),A(N7),A(N8),
     *              A(N8A),
     *              A(N9),A(N10),A(N11),A(N12),
     *              A(N13),
     *              A(N14),A(N15),A(N16),       A(N17),A(N18),A(N19),
     *              A(N2C),
     *              A(N21),A(N22),A(N23),A(N24),A(N25),
     *                 NX,NY,KX,KY,KW  )
      GO TO 110
  100 CALL START1
  110 CONTINUE

  510 FORMAT(8A10/7F10.0/4F10.0/3F10.0/4F10.0)
  515 FORMAT(7F10.0)
  520  FORMAT(C10)
  610 FORMAT(1H1,8A10,8X,2A10/1HO,
     2 8X2HNX 8X2HNY 8X3HMDY8X2HDX 8X2HDY 8X2HDS 8X2HDN /1X3F10.0,4F10.6
     2 /1HO,
```

```
     3 5X5HUNIFY 6X4HFSMN 5X5HGAMMA 5X5HALFAD /1X,F10.0,3F10.4 /1H0,
     57X3HBND 4X6HBNDISC /1X,2F10.0/1H0,
     8 5X5HNCTMX 5X5HNBTMX 8X2HKX 8X2HKY /1X,4F10.0 )
  618 FORMAT(1H0,5X5HNPRNT,6X4HNBUG 5X5HNSAVE/1X,3F10.0)
  620 FORMAT(1H0, 7X3HTMX 6X4HNTMX /1X,F10.4,F10.0)
  622 FORMAT(1H0, 5X5HICCTL /1X,O10 )

      CALL TIMER(TMR,TIMB,TIMS,TIMT)
      WRITE(IW,6010)TIMT
 6010 FORMAT(1H0,*CP,PP, TIMES TO  $ INITIALISE $ ARE(SEC) *,2F10.2)

      END
```

```
      SUBROUTINE CHECK

C CHECKS THAT *NN* IS LARGE ENOUGH TO SOLVE PRESENT CASE BEING

      COMMON / CM8A   / NCTMX,NBTMX,NX,NY,KX,KY,KW,NBT,MM
      COMMON / CM8B   / MS,ML,KBLKS
      COMMON / CM8C   / MSO,MLO,KBLKSO
      COMMON / CM11   / IR,IW
      COMMON / CM13   / ISTART,ISTOP,ISAVE,NSAVE
      COMMON / CM999  / NN,A(1)
      NC = NCTMX
      NB = NBTMX
      KK1 = KX*(NX+1) + KY*(NY+1)
      KK2 = KX*(2*NX+1) + KY*(2*NY+1)
      NY4 = 4*NY
      NY8  =  8*NY
      NY12 = 12*NY
      NY39 = 39*NY
      IBNDG  = 42*NC                        $     LBNDG = NN - IBNDG
      IBNDI  =  4*NC     + KK2  +NY8
      IBNDR  = 49*NC                        $     LBNDR = NN - IBNDR
      IBND1  = 57*NC                        $     LBND1 = NN - IBND1
      INWCT  = 29*NC                        $     LNWCT = NN - INWCT
      INTCN  =  9*NC                        $     LNTCN = NN - INTCN
      IMAPT  =  2*KW*NX + KK1 + NB          $     LMAPT = NN - IMAPT
      ILXWN  =    KW*NX + NY39
      ILPGN  =  9*NB                        $     LLPGN = NN - ILPGN
      IRHSB  = 13*NB     + 4*NC             $     LRHSB = NN - IRHSB
      IRHSL  = 13*NB     + NY12
      ILHSS  = 13*NB                        $     LLHSS = NN - ILHSS
      ISOLV  = 17*NB                        $     LSOLV = NN - ISOLV
      ISTPR  =  5*NB     + NY4
      MBNDI = (NN - IBNDI)/NY4 + 1
      MLXWN = (NN - ILXWN)/NY4 + 1
      MRHSL = (NN - IRHSL)/NY4 + 1
      MSTPR = (NN - ISTPR)/NY4 + 1
      MS = MINO(MBNDI,MLXWN,MRHSL,MSTPR)
      IF (MS.LT.1) MS = 1
```

```
      IF(ISTART.EQ.4) MS = MSO
      KBLKS = 1 + (NX-1)/MS
      ML    = NX - (KBLKS - 1)*MS
      IF(KBLKS.EQ.1) MS = ML
      MM = MS*NY
      KBNDI = IBNDI + NY4*(MS-1)
      KLXWN = ILXWN + NY4*(MS-1)
      KRHSL = IRHSL + NY4*(MS-1)
      KSTPR = ISTPR + NY4*(MS-1)
      LBNDI = NN - KBNDI
      LLXWN = NN - KLXWN
      LRHSL = NN - KRHSL
      LSTPR = NN - KSTPR
      LEAST = MIN0( LBNDG, LBNDI, LBNDR, LBND1,
     *              LNWCT, LNTCN, LMAPT, LLXWN,
     *              LLPGN, LRHSL, LRHSB, LLHSS, LSOLV, LSTPR )
      WRITE(IW,600)
      WRITE(IW,605) NC,NB,NX,NY,KX,KY,KW,NN
      WRITE(IW,630)
     *     IBNDG,                LBNDG,
     *     IBNDI,     KBNDI,     LBNDI,
     *     IBNDR,                LBNDR,
     *     IBND1,                LBND1,
     *     INWCT,                LNWCT,
     *     INTCN,                LNTCN,
     *     IMAPT,                LMAPT,
     *     ILXWN,     KLXWN,     LLXWN,
     *     ILPGN,                LLPGN,
     *     IRHSB,                LRHSB,
     *     IRHSL,     KRHSL,     LRHSL,
     *     ILHSS,                LLHSS,
     *     ISOLV,                LSOLV,
     *     ISTPR,     KSTPR,     LSTPR
      WRITE(IW,620) KBLKS,MS,ML,MM

      IF( LEAST.GE.0 ) GO TO 99
      NREQD = NN - LEAST
      WRITE(IW,640) NREQD
```

```
      CALL EXIT
  600 FORMAT(1H1,* CORE STORAGE ANALYSIS * )
  605 FORMAT(1HO, 5X5HNCTMX 5X5HNBTMX 8X2HNX 8X2HNY 8X2HKX 8X2HKY
     *            8X2HKW 8X2HNN /1X,8I10 )
  620 FORMAT(1HC,  5X5HKBLKS 8X2HMS 8X2HML 8X2HMM/ 1X,4I10 )
  630 FORMAT(1HO,10X10HSUBROUTINE,2X18HMINIMUM WORDS REQD,10X10HWORDS US
     *ED, 8X12HWORDS UNUSED /1X,
     *14X6HBNDCOM , I20,I40/1X
     *14X6HBNDINT , 3I20   /1X
     *14X6HBNDREC , I20,I40/1X
     *14X6HBND1   , I20,I40/1X
     *14X6HNEWCT  , I20,I40/1X
     *14X6HINTCON , I20,I40/1X
     *14X6HMAPIT  , I20,I40/1X
     *14X6HLXWN   , 3I20   /1X
     *14X6HLAPCON , I20,I40/1X
     *14X6HRHSBND , I20,I40/1X
     *14X6HRHSLXW , 3I20   /1X
     *14X6HLHS    , I20,I40/1X
     *14X6HSOLVE  , I20,I40/1X
     *14X6HSTPR   , 3I20       )
  640 FORMAT(1HO,* MINIMUM DIMENSION OF ARRAY $A$ SHOULD BE *,I10/
     *1X,* RUN ABORTED* )
   99 RETURN
      END
```

```
      SUBROUTINE CONESOL

C THIS SUBROUTINE ESTABLISHES A CONE AT ANGLE OF ATTACK SOLUTION

      COMMON / CM3     / UNITY,FSMN,GAMMA,ALFAD
      COMMON / CM11    / IR,IW
      COMMON / CM201   / ICONE,BK,FX,R1,CONC
      COMMON / CM202   / F1(21),F2(21),F3(21),X1(21),NTAB

      PI = 3.14159265358979

      READ(IR,515) BKD,FX,R1,ANTAB
  515 FORMAT(7F10.0)
      BK = BKD*PI/180.
      TBK=TAN(BK)
      NTAB = INT(ANTAB)

      XM = FSMN
      XM2= XM*XM
      G  = GAMMA
      B2 = XM2 - 1
      B  = SQRT(B2)
      CON = 2./(G+1.)
      CON = CON*(1. + (G-1.)*XM2/2.)
      CON = SQRT(CON)/XM
      CONA  = G*(2. + (G-1.)*XM2)/(G+1.)
      CONB= .5*G*(G-1.)*B2
      CONC = 1./(FX/TAN(BK) - 1.)
      CONC = CONC/R1

      DO 100 I=1,NTAB
      READ(IR,515) X1(I),F1(I),F2(I),F3(I)
      IF(I.EQ.1) FFF = F1(I)
      F1(I) = F1(I)*TBK/FFF
      W1 = (F3(I) - 1.)*B2/XM2 + 1.
      VR = F1(I)
      W23 = W1*VR
      PPINF = CONA*F2(I)
```

```
      PPINU = (PPINF - 1.)*B2/XM2 + 1.
      W4 = PPINU + CONB*W1*VR*VR
      F1(I) = W1
      F2(I) = W23
      F3(I) = W4
  100 CONTINUE

      WRITE(IW,610) BKD,R1
  610 FORMAT(1HO, * CONE SEMI ANGLE , RADIUS OF CONTOUR ARE *
     *    ./1X,2F20.8//)

      RETURN
      END
```

0051 CARDS

```
      SUBROUTINE TANWEDG

C COMPLTES TANCENT WEDGE INITIAL

      COMMON / CM3     / UNIFY,FSMN,GAMMA,ALFAD
      COMMON / CM3A    / CT1,CT2,CT3,CT4,CT5,CT6,CT7,CT8,CT9
      COMMON / CM11    / IR,IW
      COMMON / CM301   / IWEDG, HWED,XLWED,WEDL1,WEDL4

      FUN(B) = TANTHT - 2.*COS(B)*(XM2*SIN(B)**2 - 1.)/
     * (SIN(B)*(XM2*(GAMMA+COS(2.*B)) +  2. ) )

      PI=3.14159265358979
      PI2 = PI/2.
      TOLB= .0001*PI2
      READ(IR,510) ALAMD
      ALAMD = 90.- ALAMD
  510 FORMAT(7F10.0)


      ALAMR = PI*ALAMD/180.
      ALFAR = PI*ALFAD/180.
      TANALF= TAN(ALFAR)
      TANLAM= TAN(ALAMR)
      TANTHT = TANALF
      XM1 = FSMN
      XM2    = XM1*XM1


C COMPUTE BETA BY METHOD OF BISECTION

      DEL = PI/180.
      B1=DEL
      F1=FUN(B1)
      B2=DEL
    5 B2=B2+DEL
      F2=FUN(B2)
      IF(F1*F2.GT.0.)GO TO 5
```

```
      B1=B2-DEL
   10 F1 = FUN(B1)
      F2 = FUN(B2)
      B3 = .5*(B1+B2)
      IF(ABS(B1-B2).LT.TOLB) GO TO 70
      F3 = FUN(B3)
      IF((F1*F3).LE.0.) GO TO 50
      IF((F2*F3).LE.0.) GO TO 60
      WRITE(IW,610) B1,B2,B3,F1,F2,F3
  610 FORMAT(1HO,*  B1,B2,B3,F1,F2,F3  ARE  * /1X,6E20.6/1X,*ERROR IN
     * PATCH*)
      CALL EXIT
   50 F2 = F3
      B2 = B3
      GO TO 10
   60 F1 = F3
      B1 = B3
      GO TO 10


   70 BETA = B3
      SINB = SIN(BETA)
      SINB2= SINB**2
      G = GAMMA
      WL1 = (G+1.)*XM2*SINB2/((G-1.)*XM2*SINB2 + 2.)
      P2  = 1. + 2.*G*(XM2*SINB2 - 1.)/(G+1.)
      WEDL1 = WL1
      WEDL4 = P2
      READ(IR,510) T
      XLWED = T*TANLAM
      HWED  = T*TAN(BETA - ALFAR)

      Z=999.
      WRITE(IW,620) ALAMD,Z,Z,Z,
     *THET,BETA,Z,Z,
     *B1,B2,B3,Z,
     *F1,F2,F3,Z
  620 FORMAT(1H1,(/1X,4E20.6))
```

```
      RETURN
      END
```

0079 CARDS

```
      SUBROUTINE PATCH
C THIS SUBROUTINE COMPUTES FLOW FIELD VALUES AT THE WING LEADING EDGE

      COMMON / CM3     / UNIFY,FSMN,GAMMA,ALFAD
      COMMON / CM3A    / CT1,CT2,CT3,CT4,CT5,CT6,CT7,CT8,CT9
      COMMON / CM11    / IR,IW
      COMMON / CM101   / W1R,W2R,W3R,W4R,W1L,W2L,W3L,W4L
      COMMON / CM102   / NAS,NAE,NAN,NAW,TANLAM,TANBET
      COMMON / CM104   / NBOUND

      FUN(B) = TANTHT - 2.*COS(B)*(XM2*SIN(B)**2 - 1.)/
     * (SIN(B)*(XM2*(GAMMA+COS(2.*B)) +  2. ) )

      PI=3.14159265358979
      PI2 = PI/2.
      TOLB= .0001*PI2
      READ(IR,510) ALAMD
      ALAMD = 90.- ALAMD
      READ(IR,510) AS,AE,AN,AW,ABOUND
  510 FORMAT(7F10.0)

      NAS=AS
      NAE=AE
      NAN=AN
      NAW=AW
      NBOUND = ABOUND

      ALAMR = PI*ALAMD/180.
      ALFAR = PI*ALFAD/180.
      TANALF= TAN(ALFAR)
      TANLAM= TAN(ALAMR)
      TANTHT= TANALF/TANLAM
      THET  = ATAN(TANTHT)
      TTAL  = SQRT(TANALF**2 + TANLAM**2)
      XM1   = CT6*TTAL
      XM2   = XM1*XM1

C SET VALUES TO RIGHT OF SHOCK WAVE
```

```
      W1R = 1.
      W2R = 0.
      W3R = TANALF
      W4R = 1. - CT3*TANALF**2

C COMPUTE BETA BY METHOD OF BISECTION

      DEL = PI/180.
      B1=DEL
      F1=FUN(B1)
      B2=DEL
    5 B2=B2+DEL
      F2=FUN(B2)
      IF(F1*F2.GT.0.)GO TO 5
      B1=B2-DEL
   10 F1 = FUN(B1)
      F2 = FUN(B2)
      B3 = .5*(B1+B2)
      IF(ABS(B1-B2).LT.TOLB) GO TO 70
      F3 = FUN(B3)
      IF((F1*F3).LE.0.) GO TO 50
      IF((F2*F3).LE.0.) GO TO 60
      WRITE(IW,610) B1,B2,B3,F1,F2,F3
  610 FORMAT(1H0,*  B1,B2,B3,F1,F2,F3  ARE  * /1X,6E20.6/1X,*ERROR IN
     * PATCH*)
      CALL EXIT
   50 F2 = F3
      B2 = B3
      GO TO 10
   60 F1 = F3
      B1 = B3
      GO TO 10


   70 BETA = B3
      SINB = SIN(BETA)
      SINB2= SINB**2
```

```
      G = GAMMA
      WL1 = (G+1.)*XM2*SINB2/((G-1.)*XM2*SINB2 + 2.)
      P2  = 1. + 2.*G*(XM2*SINB2 - 1.)/(G+1.)
      Q = TANLAM - TTAL*COS(BETA)/COS(BETA-THET)
      W1L = WL1
      W2L = WL1*Q
      W3L = 0.
      W4L = P2 - CT3*WL1*Q**2

      TANBET = TAN(BETA-THET)
      Z=999.
      WRITE(IW,620) ALAMD,Z,Z,Z,
     *AS,AE,AN,AW,
     *THET,BETA,Z,Z,
     *B1,B2,B3,Z,
     *P1,P2,P3,Z,
     *W1R,W2R,W3R,W4R,
     *W1L,W2L,W3L,W4L
  620 FORMAT(1H1,(/1X,4E20.6))

      RETURN
      END
```


0098 CARDS

```
      SUBROUTINE SETICT(T)
C SETS ICTFP AT FILE FOR WHICH ( T.GE.T1 .AND. T.LT.T2 )
      COMMON / CM4A   / ICT,ICTFP,T1,T2,NC1,NCURV
      COMMON / CM11   / IR,IW
      CALL POSITN(ICTFP,1,ICT,111)
   10 READ(ICT) ICTFPP,T1,T2,NC1,NCURV
      IF(ICTFP.EQ.ICTFPP) GO TO 20
      WRITE(IW,610) ICTFP,ICTFPP
  610 FORMAT(1HO,*  ICT  OUT OF SYNC. ICTFP,ICTFPP ARE * 2I10/1HO,
     * * RUN ABORTED *)
      CALL EXIT
   20 IF (  T.GE.T1  .AND. T.LT.T2 ) GO TO 999
      CALL POSITN ( ICTFP,ICTFP+1,ICT,112)
      GO TO 10
  999 RETURN
      END
```

```
      SUBROUTINE START1
      COMMON / CM4A    / ICT,ICTFP,T1,T2,NC1,NCURV
      COMMON / CM4B    / TPREV,T,INDCT,NT,NTFRST
      COMMON / CM11    / IR,IW
      COMMON / CM11A   / ISV,    ISVFP
      COMMON / CM11B   / ID1,ID2,IL1,IL2,    ID1FP,ID2FP,IL1FP,IL2FP
      REWIND ISV
      ISVFP = 0

      READ(IR,510) T
      READ(IR,510) SVFPP
      ISVFPP = INT(SVFPP)
      IF(ISVFPP.GT.1) GO TO 10
      WRITE(ISV) DUM
      ENDFILE ISV
      ISVFP = 1
      GO TO 20
   10 CALL POSITN(ISVFP,ISVFPP,ISV,1111)
   20 CONTINUE

      NT=-1
      INDCT = 1
      NTFRST = NT
      TPREV = T
      CALL ENDFL(ID1FP,ID1)
      CALL SETICT(T)
      READ(ICT)
      WRITE(IW,610) T,ICTFP

  510 FORMAT(F10.0)
  610 FORMAT(1HO,* START1 CALLED * /
     *       1HO,* STARTING VALUE OF T = *,F20.6 /
     *       1HO,* GEOMETRY IOUNIT SET AT FILE POSITION * I5 )

      RETURN
      END
      SUBROUTINE START2 ( X,Y,D,CS,SN,C1,C2,C3,S,
     *                    XC,YC,XCC,YCC,
```

```
*                    WC,
*                    IX,VX,IY,VY,NSX,NSY,
*                    MAPBNW,
*                    W1,W2,W3,W4,
*               NPTA,
*                  NX,NY,KX,KY,KW )

      COMMON / CM4    / CF,DTMX, DXYSN, DTFRST,DTOUT,DTBND,DTSTB,DT
      COMMON / CM4A   / ICT,ICTFP,T1,T2,NC1,NCURV
      COMMON / CM4B   / TPREV,T,INDCT,NT,NTFRST
      COMMON / CM8B   / MS,ML,KBLKS
      COMMON / CM11   / IR,IW
      COMMON / CM11A  / ISV,   ISVFP
      COMMON / CM11B  / ID1,ID2,IL1,IL2,   ID1FP,ID2FP,IL1FP,IL2FP

      DIMENSION X(1),Y(1),D(1),CS(1),SN(1),C1(1),C2(1),C3(1),S(1),
     *          XD(1),YD(1),XCD(1),YCD(1),
     *          WC(4,1),
     *          IX(NX),VX(KX,NX),IY(NY),VY(KY,NY),NSX(KX,NX),NSY(KY,NY),
     *          MAPBNW(KW,NX),
     *          W1(1),W2(1),W3(1),W4(1)
      DIMENSION NPTA(1)

      REWIND ISV
      ISVFP = 0
      READ(IR,515) SVFPP
  515 FORMAT(7F10.0)
      ISVFPP = INT(SVFPP)
      CALL POSITN(ISVFP,ISVFPP,ISV,1001)

      READ(ISV) DTOUT,DTBND,ICTFP,T1,T2,NC1,NCURV,TPREV,T,INDCT,NT
      READ(ISV)
      CALL SETICT(T)
      READ(ICT)
      NTFRST = NT

      CALL POSITN(ID2FP,1,ID2,121)
      CALL ENDFL (ID2FP,ID2)
```

```
      NC = NC1
      READ (ISV) (X(I),I=1,NC),
     *           (Y(I),I=1,NC),
     *           (D(I),I=1,NC),
     *           (CS(I),I=1,NC),
     *           (SN(I),I=1,NC),
     *           (C1(I),I=1,NC),
     *           (C2(I),I=1,NC),
     *           (C3(I),I=1,NC),
     *           (S(I),I=1,NC)
      WRITE(ID2) (X(I),I=1,NC),
     *           (Y(I),I=1,NC),
     *           (D(I),I=1,NC),
     *           (CS(I),I=1,NC),
     *           (SN(I),I=1,NC),
     *           (C1(I),I=1,NC),
     *           (C2(I),I=1,NC),
     *           (C3(I),I=1,NC),
     *           (S(I),I=1,NC)
      READ(ISV) (S(I),I=1,NC),
     *          (NPTA(I),I=1,NCURV)
      WRITE(ID2)(S(I),I=1,NC),
     *         (NPTA(I),I=1,NCURV)
      CALL ENDFL(ID2FP,ID2)

      READ (ISV) (XD(I),I=1,NC),
     *           (YD(I),I=1,NC),
     *           (XDD(I),I=1,NC),
     *           (YDD(I),I=1,NC)
      WRITE(ID2) (XD(I), I=1,NC),
     *           (YD(I), I=1,NC),
     *           (XDD(I),I=1,NC),
     *           (YDD(I),I=1,NC)
      CALL ENDFL(ID2FP,ID2)

      READ (ISV) ((WC(J,I),J=1,4),I=1,NC)
      WRITE(ID2) ((WC(J,I),J=1,4),I=1,NC)
      CALL ENDFL (ID2FP,ID2)
```

```
      READ (ISV)  IX,VX,IY,VY,NSX,NSY
      WRITE(ID2)  IX,VX,IY,VY,NSX,NSY
      CALL ENDFL (ID2FP,ID2)

      READ (ISV)  MAPBNW
      WRITE(ID2)  MAPBNW
      CALL ENDFL (ID2FP,ID2)
      CALL POSITN(IL1FP,1,IL1,121)

      DO 190 KB=1,KBLKS
      M = MS
      IF(KB.EQ.KBLKS) M = ML
      MTOT = M*NY
      IF(KB.EQ.1) GO TO 120
      READ(ISV) (W1(K),K=1,NY),
     *          (W2(K),K=1,NY),
     *          (W3(K),K=1,NY),
     *          (W4(K),K=1,NY)
      WRITE(IL1)(W1(K),K=1,NY),
     *          (W2(K),K=1,NY),
     *          (W3(K),K=1,NY),
     *          (W4(K),K=1,NY)
  120 CONTINUE
      READ (ISV) (W1(K),K=1,MTOT),
     *           (W2(K),K=1,MTOT),
     *           (W3(K),K=1,MTOT),
     *           (W4(K),K=1,MTOT)
      WRITE(IL1) (W1(K),K=1,MTOT),
     *           (W2(K),K=1,MTOT),
     *           (W3(K),K=1,MTOT),
     *           (W4(K),K=1,MTOT)
      IF(KB.EQ.KBLKS) GO TO 190
      READ (ISV) (W1(K),K=1,NY),
     *           (W2(K),K=1,NY),
     *           (W3(K),K=1,NY),
     *           (W4(K),K=1,NY)
      WRITE(IL1) (W1(K),K=1,NY),
```

```
     *            (W2(K),K=1,NY),
     *            (W3(K),K=1,NY),
     *            (W4(K),K=1,NY)
  190 CONTINUE
      READ(IR,515) SVFPP
      ISVFPP = INT(SVFPP)
      CALL POSITN(ISVFP,ISVFPP,ISV,1002)
      RETURN
      END
```

```
      OVERLAY(OVER,2,0)
      PROGRAM RESET

      COMMON / CM4A    / ICT,ICTFP,T1,T2,NC1,NCURV
      COMMON / CM4B    / TPREV,T,INDCT,NT,NTFRST
      COMMON / CM8A    / NCTMX,NBTMX,NX,NY,KX,KY,KW,NBT,MM
      COMMON / CM1C    / ICCTL,TMX,NTMX,TMR
      COMMON / CM10A   / TIMA(2),TIMB(2),TIMS(2),TIMT(2),TIM(2,8)
      COMMON / CM11    / IR,IW
      COMMON / CM11B   / IC1,IC2,IL1,IL2,   IC1FP,IC2FP,IL1FP,IL2FP
      COMMON / CM13    / ISTART,ISTOP,ISAVE,NSAVE
      COMMON / CM14    / NPRNT,IPRNT,NBUG,IBUG
      COMMON / CM999   / NN,A(1)

      IF(MOD(NT,NSAVE).EQ.0)        ISAVE = 1
      IF(T.GE.TMX)                  ISAVE = ISTOP = 2
      IF(NT.GE.NTMX)                ISAVE = ISTOP = 3

      IF(NT.EQ.NTFRST) GO TO 10
      TT = FLOAT(ICCTL) - TIMB(1)
      TTT=TIMT(1)+TIM(1,6)
      IF(TTT.GE.TT)                 ISAVE = ISTOP = 4
   10 CONTINUE

      IF(NT.EQ.0)                   ISAVE = -1
      IF(NT.EQ.NTFRST)              ISAVE = -1
      IF(NSAVE.GE.9999)             ISAVE = -1

      IF(ISAVE.EQ.-1)  GO TO 15
      NC = NC1
      N1 = 1
      N2 = N1 + NC
      N3 = N2 + NC
      N4 = N3 + NC
      N5 = N4 + NC
      N6 = N5 + NC
      N7 = N6 + NC
      N8 = N7 + NC
```

```
      N8A= N8 + NC
      N9 = N1
      N10= N2
      N11= N3
      N12= N4
      N13= N1
      N14= N1
      N15= N14 + NX
      N16= N15 + NX*KX
      N17= N16 + NY
      N18= N17 + NY*KY
      N19= N18 + NX*KX
      N20= 1
      N21= 1
      N22 = N21 + MM
      N23 = N22 + MM
      N24 = N23 + MM
      N25 = N24 + NC
      CALL SAVE ( A(N1),A(N2),A(N3),A(N4),A(N5),A(N6),A(N7),A(N8),
     *              A(N8A),
     *            A(N9),A(N10),A(N11),A(N12),
     *            A(N13),
     *            A(N14),A(N15),A(N16),A(N17),A(N18),A(N19),
     *            A(N20),
     *              A(N21),A(N22),A(N23),A(N24),A(N25),
     *              NC,NX,NY,KX,KY,KW,NCURV )
   15 CONTINUE
      ISAVE = -1
      IF( ISTOP.EQ.2 ) WRITE(IW,610) T,TMX
      IF( ISTOP.EQ.3 ) WRITE(IW,620) NT,NTMX
      IF( ISTOP.EQ.4 ) WRITE(IW,630) TT,TTT
      IF( ISTOP.GT.0 ) CALL EXIT

C SET PRINT INDICES

      IPRNT = 0
      IBUG  = 0
      IF(MOD(NT+1,NPRNT).EQ.0) IPRNT= 1
```

```
      IF(MCD(NT+1,NBUG).EQ.0 ) IBUG  = 1
      IF((NT+1).EQ.0) IPRNT = 0
      IF((NT+1).EQ.0.AND.NBUG.NE.1) IBUG=0
      IF(NBUG.GE.9999) IBUG = 0
      IF(NPRNT.GE.9999) IPRNT=0


C RESET ID1,ID2
      IDSAVE = ID1
      ID1 = ID2
      ID2 = IDSAVE
      IFSAVE=ID1FP
      ID1FP=ID2FP
      ID2FP=IFSAVE
      IF(NT.EQ.-1) GO TO 50

C TRANSFER *MAPBCL* FROM *ID1* TO *ID2*
      NXKW = NX*KW
      CALL POSITN(ID1FP,6,ID1,211)
      READ(ID1) (A(I),I=1,NXKW)
      CALL POSITN(ID2FP,1,ID2,212)
      WRITE(ID2) (A(I),I=1,NXKW )
      CALL ENDFL(ID2FP,ID2)

   50 CONTINUE
      CALL DELTAT

  610 FORMAT(1HO,* RUN TERMINATED BECAUSE T.GT.TMX * 2F20.10)
  620 FORMAT(1HO,* RUN TERMINATED BECAUSE NT.GT.NTMX * 2I20 )
  630 FORMAT(1HO,* RUN TERMINATED BECAUSE THERE MAY NOT BE SUFFICIENT CP
     * TIME AVAILABLE TO COMPLETE NEXT TIME STEP * /1X,2F20.10)

      CALL TIMER(TMR,TIMS,TIMB,TIM(1,2))
      WRITE(IW,6090) TIM(1,2),TIM(2,2),T,NT
 6090 FORMAT(1HO,* CP,PP TIMES FOR  $ RESET  $  ARE (SEC) *,2F10.2,5X
     $*T = * F10.6, 4X * NT = * I6 )

      END
```

```
      SUBROUTINE DELTAT

C COMPUTES *DT*

      COMMON / CM4    / CF,DTMX, DXYSN, DTFRST,DTCUT,DTBND,DTSTB,DT
      COMMON / CM4A   / ICT,ICTFP,T1,T2,NC1,NCURV
      COMMON / CM4B   / TPREV,T,INDCT,NT,NTFRST
      COMMON / CM11   / IR,Ih
      COMMON / CM11B  / ID1,ID2,IL1,IL2,   ID1FP,ID2FP,IL1FP,IL2FP

      NT=NT+1
      IF(NT.EQ.0) GO TO 999
      TPREV = T
      DTSTB = AMIN1(DTFRST,DTCUT,DTBND,DTMX)
      DTPRV2 = T2 - TPREV
      IF(DTSTB.GE.(    DTPRV2)) GO TO 20
      IF(DTSTB.GE.(.75*DTPRV2)) GO TO 10

      DT = DTSTB
      T = TPREV + DT
      INDCT = 0
      GO TO 30

   10 DT = .5*DTPRV2
      T = TPREV + DT
      INDCT = 1
      GO TO 30

   20 DT = DTPRV2
      T=T2
      INDCT = 2

   30 CONTINUE
      WRITE(Ih,610) NT,T,DT,DTFRST,DTBND,DTCUT,INDCT
  610 FORMAT(1HO,*NEW TIME STEP BEGINS* / 1X,
     *        21H**********************, / 1HO,
     *8X2HNT 9X1HT  8X2HDT 4X6HDTFRST 5X5HDTBND 5X5HDTCUT 5X5HINDCT /1X,
     * I1C,5F10.6,I1C )
```

```
999 CONTINUE
    RETURN
    END
```

```
      SUBROUTINE SAVE ( X,Y,D,CS,SN,C1,C2,C3,S,
     *                  XD,YD,XDD,YDD,
     *                  WC,
     *                  IX,VX,IY,VY,NSX,NSY,
     *                  MAPBNW,
     *                  W1,W2,W3,W4,
     *              NPTA,
     *              NC,NX,NY,KX,KY,KW,NCRV     )

      COMMON / CM4    / CF,DTMX, DXYSN, DTFRST,DTOUT,DTBND,DTSTB,DT
      COMMON / CM4A   / ICT,ICTFP,T1,T2,NC1,NCURV
      COMMON / CM4B   / TPREV,T,INDCT,NT,NTFRST
      COMMON / CM8B   / MS,ML,KBLKS
      COMMON / CM11   / IR,IW
      COMMON / CM11A  / ISV,   ISVFP
      COMMON / CM11B  / ID1,ID2,IL1,IL2,   ID1FP,ID2FP,IL1FP,IL2FP

      DIMENSION X(NC),Y(NC),D(NC),CS(NC),SN(NC),C1(NC),C2(NC),C3(NC),
     *          S(NC),
     *          XD(NC),YD(NC),XDD(NC),YDD(NC),
     *          WC(4,NC),
     *          IX(NX),VX(KX,NX),IY(NY),VY(KY,NY),NSX(KX,NX),NSY(KY,NY),
     *          MAPBNW(KW,NX),
     *          W1(1),W2(1),W3(1),W4(1)
      DIMENSION NPTA(NCRV)


      WRITE(ISV) DTOUT,DTBND,ICTFP,T1,T2,NC1,NCURV,TPREV,T,INDCT,NT
      WRITE(ISV) MDYC,NXC,NYC,KBLKSO,MSC,MLO

      CALL POSITN(ID2FP,2,ID2,222 )
      READ(ID2) X,Y,D,CS,SN,C1,C2,C3,S
      WRITE(ISV)X,Y,D,CS,SN,C1,C2,C3,S
      READ(ID2) S,NPTA
      WRITE(ISV)S,NPTA

      CALL POSITN(ID2FP,3,ID2,223 )
      READ (ID2) XD,YD,XDD,YDD
```

```
      WRITE(ISV) XC,YC,XCC,YCC

      CALL POSITN(IC2FP,4,IC2,224 )
      READ (IC2) WC
      WRITE(ISV) WC

      CALL POSITN(IC2FP,5,IC2,225)
      READ (IC2) IX,VX,IY,VY,NSX,NSY
      WRITE(ISV) IX,VX,IY,VY,NSX,NSY

      CALL POSITN (IC2FP,6,IC2,226)
      READ (IC2) MAPBNW
      WRITE(ISV) MAPBNW

      CALL POSITN(IL1FP,1,IL1,227)

      DO 190 KB = 1,KBLKS
      M = MS
      IF(KB.EQ.KBLKS) M = ML
      MTOT = M*NY
      IF(MB.EQ.1) GO TO 120
      READ (IL1) (W1(K),K=1,NY),
     *           (W2(K),K=1,NY),
     *           (W3(K),K=1,NY),
     *           (W4(K),K=1,NY)
       WRITE(ISV) (W1(K),K=1,NY),
     *           (W2(K),K=1,NY),
     *           (W3(K),K=1,NY),
     *           (W4(K),K=1,NY)
  120 CONTINUE
      READ (IL1) (W1(K),K=1,MTOT),
     *           (W2(K),K=1,MTOT),
     *           (W3(K),K=1,MTOT),
     *           (W4(K),K=1,MTOT)
       WRITE(ISV) (W1(K),K=1,MTOT),
     *           (W2(K),K=1,MTOT),
     *           (W3(K),K=1,MTOT),
     *           (W4(K),K=1,MTOT)
```

```
  130 CONTINUE
      IF(KB.EQ.KBLKS) GO TO 190
      READ (IL1) (W1(K),K=1,NY),
     *           (W2(K),K=1,NY),
     *           (W3(K),K=1,NY),
     *           (W4(K),K=1,NY)
      WRITE(ISV) (W1(K),K=1,NY),
     *           (W2(K),K=1,NY),
     *           (W3(K),K=1,NY),
     *           (W4(K),K=1,NY)
  190 CONTINUE
      WRITE(IW,6000) T,NT,ISVFP
      CALL ENDFL(ISVFP,ISV)
 6000 FORMAT(1HO, * INFORMATION SAVED , T,NT,ISVFP ARE  * F20.6,2I10)
      RETURN
      END
```

```
      CVERLAY(CVER,3,0)
      PROGRAM BNDRY
      COMMON / CM4A   / ICT,ICTFP,T1,T2,NC1,NCURV
      COMMON / CM4B   / TPREV,T,INDCT,NT,NTFRST
      COMMON / CM5    / BND,BNDISC
      COMMON / CM8A   / NCTMX,NBTMX,NX,NY,KX,KY,KW,NBT,MM
      COMMON / CM10   / ICCTL,TMX,NTMX,TMR
      COMMON / CM10A  / TIMA(2),TIMB(2),TIMS(2),TIMT(2),TIM(2,8)
      COMMON / CM11   / IR,IW
      COMMON / CM999  / NN,A(1)
      IF(NT.EQ.0) GO TO 999
      MM1 = MM + NY
      NP = NC1
      N1 = 1
      N4 = N1 + 12*NP
      N5 = N4 + 3*NP
      N6 = N5 + 3*NP
      N7 = N6 + 3*NP
      N8 = N7 + 6*NP
      N9 = N8 + 6*NP
      N10 = N9 + NP
      N11 = N10 + NP
      N12 = N11 + NP
      N13 = N12 + NP
      N14 = N13 + NP
      N15 = N14 + NP
      N16 = N15 + NP
      N17 = N16 + NP

      CALL BNDGOM ( A(N1),A(N4),A(N5),A(N6),
     *              A(N7), A(N8),
     *              A(N9), A(N10), A(N11), A(N12), A(N13), A(N14),
     *              A(N15), A(N16), A(N17),   NP )

      N1 = 1
      N2 = N1 + 4*NP
      N3 = N2 + MM1
      N4 = N3 + MM1
```

```
      N5 = N4 + MM1
      N6 = N5 + MM1
      N7 = N6 + NX
      N8 = N7 + KX*NX
      N9 = N8 + NY
      N10 = N9 + KY*NY
      N11 = N10 +KX*NX

      CALL BNDINT ( A(N1), A(N2), A(N3),A(N4), A(N5), A(N6), A(N7),
     *              A(N8),A(N9),A(N10),A(N11), MM1,NP,KX,NX,KY,NY )

      N1 = 1
      N2 = N1 + 36*NP
      N3 = N2 + 3*NP
      N4 = N3 + 3*NP
      N5 = N4 + 3*NP
      N6 = N5 + NP
      N7 = N6 + NP
      N8 = N7 + NP

      CALL BNDRED ( A(N1), A(N2), A(N3), A(N4),
     *              A(N5), A(N6), A(N7), A(N8),  NP )

      N9 = N8 + NP
      N10 = N9 + 4*NP
      N11 = N10 + 4*NP
      N12 = N11 +   NP
      IF ( BND.EQ.1 )
     *CALL BND1    ( A(N1), A(N2), A(N3), A(N4),
     *               A(N5), A(N6), A(N7), A(N8),
     *               A(N9), A(N10), A(N11), A(N12), NP, NCURV )
      N13 = N10
      N12 = N9
      N11 = N3
      N10 = N2
      N1  = 1
      N2  = N1 + NP
      N3  = N2 + NP
```

```
      N4   = N3 + NP
      N5   = N4 + NP
      N6   = N5 + NP
      N7   = N6 + NP
      N8   = N7 + NP
      N9   = N8 + NP
      N14  = N9 + NP
      CALL BNDWRI ( A(N1), A(N2), A(N3), A(N4),
     *              A(N5), A(N6), A(N7), A(N8), A(N9),
     *         A(N10),A(N11),A(N12),A(N13),A(N14), NP )
  999 CONTINUE

      CALL TIMER(TMR,TIMB,TIMA,TIM(1,3))
      WRITE(IW,6090) TIM(1,3),TIM(2,3),T,NT
 6090 FORMAT(1HO,* CP,PP TIMES FOR  $ BNDRY  $  ARE (SEC) *,2F10.2,5X
     $*T = * F10.6, 4X * NT = * I6 )

      END
```

```
      SUBROUTINE BNDGCM ( WCP,COSN,SINN,CUR,
     *                    X,Y,
     *                    XP,YP,DP,CSP,SNP,C1P,C2P,C3P,SP,  NP )

C FINDS CO-ORDS OF 6*NP POINS.   SORTS THEM OUT.   STORES THEM FOR BNDINT
C FINDS (W) FOR 2*NP POINTS AND STORES THEM IN *WCP1,WCP3*
C COMPUTES COSN,SINN,CUR


      COMMON / CM2     / ZNX,ZNY,YZ,DX,DY,DS,DN,   RDX,RDY,RDS,RDN
      COMMON / CM7     / TOLS,TOLX,TOLCB,TOLCF,TOLSCL
      COMMON / CM8A    / NCTMX,NBTMX,NX,NY,KX,KY,KW,NBT,MM
      COMMON / CM8B    / MS,ML,KBLKS
      COMMON / CM11    / IR,IW
      COMMON / CM11B   / ID1,ID2,IL1,IL2,   ID1FP,ID2FP,IL1FP,IL2FP

      DIMENSION WCP(4,3,NP),
     *          XP(NP),YP(NP),DP(NP),CSP(NP),SNP(NP),
     *          C1P(NP),C2P(NP),C3P(NP),SP(NP),
     *          X(6,NP),Y(6,NP),
     *          COSN(3,NP),SINN(3,NP),CUR(3,NP)
      COMMON / BNDLCL / INDB(3)

C READ CONTP   IC1(2)

      CALL POSITN ( ID1FP,2,IC1,311 )
      READ(ID1) XP,YP,DP,CSP,SNP,C1P,C2P,C3P,SP
C READ WCP2    IC1(4)
      CALL POSITN ( ID1FP,4,IC1,312 )
      READ(ID1)(( WCP(K,2,I),K=1,4),I=1,NP)

C FIND X,Y CO-CRDS OF POINTS 5,8 CORRESPONDING TO EACH CONTOUR POINT

      DO 90 I=1,NP
      IF(I.EQ.NP) GO TO 10
      CALL NRMCUR(0.,CSP(I),SNP(I),C1P(I),C2P(I),C3P(I),CS,SN,CUR(2,I))
      IF(I.EQ.1) GO TO 20
      I1 = I - 1
```

```
      CALL NRMCUR( DP(I1),CSP(I1),SNP(I1),C1P(I1),C2P(I1),C3P(I1),
     * DUM,DUM,CURR)
      CUR(2,I) = .5*(CUR(2,I) + CURR )
      GO TO 20
   10 N = NP - 1
      CALL NRMCUR(DP(N),CSP(N),SNP(N),C1P(N),C2P(N),C3P(N),CS,SN,
     *            CUR(2,I))
      CS = 0.
      SN = 1.
      GO TO 30
   20 IF(I.NE.1) GO TO 30
      CS = 0.
      SN = -1.
   30 DNCOS = DN*CS
      DNSIN = DN*SN
      XX    = XP(I)
      YY    = YP(I)
      X(2,I) = XX + DNCOS
      X(5,I) = XX + 2.*DNCOS
      Y(2,I) = YY + DNSIN
      Y(5,I) = YY + 2.*DNSIN
      COSN(2,I) = CS
      SINN(2,I) = SN
   90 CONTINUE

C FIND CO-ORDS OF POINTS 4,7 CORRESPONDING TO EACH CONTOUR POINT
      DO 290 L=1,3
      GO TO (101,102,103),L
  101 CON = 1.
      JXY1= 1
      JXY2= 4
      INDB(1) = NP
      GO TO 110
  102 GO TO 290
  103 CON = -1.
      JXY1=3
      JXY2=6
      INDB(3)=1
```

```
  110 CONTINUE
      DO 190 I=1,NP
      ARC = SP(I) + CON*DS
      CALL ALL( NP,XP,YP,DP,CSP,SNP,C1P,C2P,C3P,SP,TOLS,TOLX,
     *          ARC, J,S1,S2,SGN,CS,SN,CURV,XX,YY )
      J1 = J + 1
      CUR(L,I) = CURV
      DNCOS = DN*CS
      DNSIN = DN*SN
      X(JXY1,I) = XX + DNCOS
      X(JXY2,I) = XX + 2.*DNCOS
      Y(JXY1,I) = YY + DNSIN
      Y(JXY2,I) = YY + 2.*DNSIN
      DO 150 K=1,4
  150 WCP(K,L,I) = S2*WCP(K,2,J) + S1*WCP(K,2,J1)
      IF(SGN.EQ.1.)GO TO 180
      CS = -CS
      WCP(2,L,I) = -WCP(2,L,I)
      IF(L.EQ.1) INDB(L)=MINO(INDB(L),I)
      IF(L.EQ.3) INDB(L)=MAXO(INDB(L),I)
  180 CONTINUE
      COSN(L,I) = CS
      SINN(L,I) = SN
  190 CONTINUE
  290 CONTINUE

      CALL POSITN ( IO2FP,2,IO2,313 )
      WRITE(IO2)(((WCP(K,J,I),K=1,4),I=1,NP),J=1,3),COSN,SINN,CUR
      WRITE(IO2)(WCP(4,2,I),I=1,NP)
      CALL ENDFL(IO2FP,IO2)
C SORT CUT POINTS AND WRITE (I,J,X,Y) ON IO2(3)
      NPTS = 0
      IBLKND = -1
      DO 700 KB = 1,KBLKS
      M = MS
      IF(K.EQ.KBLKS) M = ML
      KLEFT = (KB-1)*MS
      IF(KB.EQ.1) KLEFT = 1
```

```
      KRGHT = KB*MS
      XLEFT = FLOAT(KLEFT - 1)*DX
      XRGHT = FLOAT(KRGHT - 1)*DX
      DO 600 I=1,NP
      DO 600 J=1,6
      XX = X(J,I)
      IF(XX.LT.XLEFT) GO TO 600
      IF(XX.GE.XRGHT) GO TO 600
      JJ = J + 3
      NPTS = NPTS + 1
      WRITE(ID2) JJ,I,XX,Y(J,I)
  600 CONTINUE
      WRITE(ID2) IBLKND,IBLKND,IBLKND,IBLKND
  700 CONTINUE

      CALL ENDFL(ID2FP,ID2)
      NP6 = 6*NP
      IF(NPTS.EQ.NP6) GO TO 800
      WRITE(IW,750) NPTS,NP6
  750 FORMAT(1HO,*ERROR IN BNDGOM    NPTS,NP6 ARE* 2I6)
      CALL EXIT
  800 CONTINUE
      RETURN
      END
```

```
      SUBROUTINE BNDINT(WCP,W1,W2,W3,W4,IX,VX,IY,VY,NSX,NSY,
     *                  MM1,NP,KX,NX,KY,NY )

C THIS SUBROUTINE COMPUTES *W* AT POINTS 4,5,6,7,8,9 OF THE 9 POINT GRID
C    ASSOCIATED WITH EACH BOUNDARY POINT

C STATEMENT FUNCTIONS

      XMESH(I) = FLOAT(I-1)*DX
      YMESH(J) = FLOAT(J-1)*DY - YZ

      COMMON / CM2    / ZNX,ZNY,YZ,DX,DY,DS,DN,   RDX,RDY,RDS,RDN
      COMMON / CM8B   / MS,ML,KBLKS
      COMMON / CM11   / IR,IW
      COMMON / CM11B  / ID1,ID2,IL1,IL2,   ID1FP,ID2FP,IL1FP,IL2FP

      DIMENSION W1(MM1),W2(MM1),W3(MM1),W4(MM1),WCP(4,NP),
     *          IX(NX),VX(KX,NX),IY(NY),VY(KY,NY),NSX(KX,NX),NSY(KY,NY)
      DIMENSION WP(4),WQ(4),WR(4),WS(4),WH(4),WV(4),WANS(4)
      COMMON / BNDLCL / INDB(3)

C READ * IX,VX,IY,VY,NSX,NSY *  FROM ID1(5)

      CALL POSITN ( ID1FP,5,ID1,321 )
      READ(ID1) IX,VX,IY,VY,NSX,NSY

C POSITION *IL1* TO READ  *W1,W2,W3,W4*
C          *ID2* TO READ  *I,J,X,Y*
C          *ID1* TO WRITE *I,J, WANS(4)

      CALL POSITN ( ID2FP,3,ID2,322 )
      CALL POSITN ( IL1FP,1,IL1,323 )
      CALL POSITN ( ID1FP,4,ID1,324 )
      READ(ID1)WCP
      CALL POSITN(ID1FP,4,ID1,325)

C THIS IS THE DO LOOP OVER THE LAX-WND BLOCKS
```

```
      DO 900 KB=1,KBLKS

C READ * W1,W2,W3,W4 *

      M = MS
      IF ( KB.EQ.KBLKS ) M=ML
      K1 = 1
      K2 = NY
      K3 = K2 + 1
      K4 = (M + 1)*NY
      IF(KB.EQ.1) GO TO 2
      READ(IL1) (W1(I),I=K1,K2),
     *          (W2(I),I=K1,K2),
     *          (W3(I),I=K1,K2),
     *          (W4(I),I=K1,K2)
      CALL FSR(1,IL1,IERR)
    2 READ(IL1) (W1(I),I=K3,K4),
     *          (W2(I),I=K3,K4),
     *          (W3(I),I=K3,K4),
     *          (W4(I),I=K3,K4)
      I1 = (KB-1)*MS
C READ  IPT,JPT,X,Y FROM  ID2(3)

    5 READ(ID2) JPT,IPT,X,Y
      IF(IPT.LT.0) GO TO 900

C FIND LOWER LEFT HAND CORNER IP,JP OF CELL IN WHICH (X,Y) LIES

      IP = INT(X*RDX) + 1
      JP = INT((Y+YZ)*RDY) + 1
      KP = (IP - I1)*NY + JP
      KQ =KP + NY
      KR =KQ + 1
      KS =KP + 1
      INC = 0
      INDX = 0
   10 WP(1) = W1(KP)
```

```
      WP(2) = W2(KP)
      WP(3) = W3(KP)
      WP(4) = W4(KP)
      IF(WP(4).NE.-2.) GO TO 20
      IND = IND + 1
      INDX = 1
   20 WQ(1) = W1(KQ)
      WQ(2) = W2(KQ)
      WQ(3) = W3(KQ)
      WQ(4) = W4(KQ)
      IF(WQ(4).NE.-2.) GO TO 30
      IND = IND + 1
      INDX = 2
   30 WR(1) = W1(KR)
      WR(2) = W2(KR)
      WR(3) = W3(KR)
      WR(4) = W4(KR)
      IF(WR(4).NE.-2.) GO TO 40
      IND = IND + 1
      INDX = 3
   40 WS(1) = W1(KS)
      WS(2) = W2(KS)
      WS(3) = W3(KS)
      WS(4) = W4(KS)
      IF(WS(4).NE.-2.) GO TO 45
      IND = IND + 1
      INDX = 4
   45 IF (IND.LE.1 ) GO TO 60
      WRITE(IW,50)
   50 FORMAT(1HO,*ERROR IN BNDINT*)
      CALL EXIT
   60 IF(IND.EQ.1) GO TO 100
      ZX = (X - XMESH(IP))*RDX
      ZY = (Y - YMESH(JP))*RDY
      ZX1 = 1. - ZX
      ZY1 = 1. - ZY
      CP = ZX1*ZY1
      CQ = ZX * ZY1
```

```
      CR = ZX *ZY
      CS = ZX1*ZY
      DO 70 K=1,4
   70 WANS(K) = CP*WP(K) + CQ*WQ(K) + CR*WR(K) + CS*WS(K)
      GO TO 800

  100 XLO = XMESH(IP)
      XHI = XMESH(IP+1)
      YLO = YMESH(JP)
      YHI = YMESH(JP+1)
      ICORN = IP
      JCORN = JP
      IF( INDX.EQ.2 .OR. INDX.EQ.3) ICORN=IP+1
      IF(INDX.EQ.3 .OR. INDX.EQ.4) JCORN=JP+1
      XCORN = XMESH(ICORN)
      YCORN = YMESH(JCORN)
      ZX = RDX*ABS(X-XCORN)
      ZY = RDY*ABS(Y-YCORN)

C FIND HORIZONTAL INTERSECTION AND WH(K)

      KMAX = IY(JCORN)
      DO 110 K=1,KMAX
      XX = VY(K,JCORN)
      IF(XLO.GT.XX)GO TO 110
      IF(XHI.LT.XX)GO TO 110
      GO TO 120
  110 CONTINUE
      CALL EXIT

  120 NSYN = NSY(K,JCORN)
      IC = MOD(NSYN,10000)
      IC1 = IC + 1
      S1 = (1.E-08)*(FLOAT(NSYN/10000))
      S2 = 1. - S1
      DO 130 K=1,4
  130 WH(K) = S1*WCP(K,IC1) + S2*WCP(K,IC)
      ZH = RDX*ABS(XX-XCORN)
```

```
C FIND VERTICAL INTERSECTION AND WV(K)

      KMAX = IX(ICORN)
      DO 210 K=1,KMAX
      YY = VX(K,ICORN)
      IF(YLO.GT.YY) GO TO 210
      IF(YHI.LT.YY) GO TO 210
      GO TO 220
  210 CONTINUE
      CALL EXIT
  220 NSXN = NSX(K,ICORN)
      IC = MOD(NSXN,10000)
      IC1 = IC + 1
      S1 = (1.E-08)*(FLOAT(NSXN/10000))
      S2 = 1. - S1
      DO 230 K=1,4
  230 WV(K) = S1*WCP(K,IC1) +  S2*WCP(K,IC)
      ZV = RDY*ABS(YY-YCORN)

C CALL  PCL3 HERE

      IF(INDX.EQ.1)CALL PCL3(WQ,WR,WS,ZX,ZY,WH,WV,ZH,ZV,WANS)
      IF(INDX.EQ.2)CALL PCL3(WR,WS,WP,ZY,ZX,WV,WH,ZV,ZH,WANS)
      IF(INDX.EQ.3)CALL PCL3(WS,WP,WQ,ZX,ZY,WH,WV,ZH,ZV,WANS)
      IF(INDX.EQ.4)CALL PCL3(WP,WQ,WR,ZY,ZX,WV,WH,ZV,ZH,WANS)
  800 CONTINUE
      GO TO (900,900,900,
     *       810,899,830,
     *       810,899,830 ),JPT
  810 IF(IPT.GE.INDB(1)) GO TO 898
      GO TO 899
  830 IF(IPT.LE.INDB(3)) GO TO 898
      GO TO 899
  898 WANS(2) = - WANS(2)
  899 WRITE(IO1)  JPT,IPT,WANS
      GO TO 5
  900 CONTINUE
```

```
      CALL ENDFL(IC1FP,IC1)

      RETURN
      END
```

```
      SUBROUTINE BNDRED( WCPP,COSN,SINN,CUR,
     *                   XD,YD,XDD,YDD,  NP)

C FILLS IN ARRAYS IN READINESS FOR BOUNDARY UPDATE SCHEMES


      COMMON / CM11B  / ID1,ID2,IL1,IL2,   ID1FP,ID2FP,IL1FP,IL2FP

      DIMENSION WCPP(4,9,NP),COSN(3,NP),SINN(3,NP),CUR(3,NP)
     *         ,XD(NP),YD(NP),XDD(NP),YDD(NP)

      CALL POSITN( ID2FP,2,ID2,331 )
      READ(ID2) ((WCPP(K,1,I),K=1,4),I=1,NP),
     *          ((WCPP(K,2,I),K=1,4),I=1,NP),
     *          ((WCPP(K,3,I),K=1,4),I=1,NP),
     *          COSN,SINN,CUR

      CALL POSITN( ID1FP,3,ID1,332 )
      READ(ID1) XD,YD,XDD,YDD

      CALL POSITN ( ID1FP,4,ID1,333 )
   10 READ(ID1) J,I,WCPP(1,J,I),WCPP(2,J,I),WCPP(3,J,I),WCPP(4,J,I)
      IF(ECF,ID1) 20,10
   20 ID1FP = 5

      RETURN
      END
```

```
      SUBROUTINE BND1(WCP,COSN,SINN,CUR,
     *                XC,YC,XCC,YCC,
     *                WC,WS,S,NPTA,  NP, M )

C IMPLEMENTS BOUNDARY SCHEME 1

      COMMON / CM2    / ZNX,ZNY,YZ,DX,DY,DS,DN,   RDX,RDY,RDS,RDN
      COMMON / CM3A   / CT1,CT2,CT3,CT4,CT5,CT6,CT7,CT8
      COMMON / CM4    / CF,CTMX, DXYSN, DTFRST,DTCUT,DTBND,DTSTB,DT
      COMMON / CM5    / BND,BNDISC,BNDPRN
      COMMON / CM11   / IR,IW
      COMMON / CM11B  / ID1,ID2,IL1,IL2,   ID1FP,ID2FP,IL1FP,IL2FP
      DIMENSION WCP(4,9,NP),COSN(3,NP),SINN(3,NP),CUR(3,NP),
     *          XD(NP),YC(NP),XCC(NP),YCC(NP),
     *          WC(4,NP),WS(4,NP),
     *          W(4,9),F(4,9),G(4,9),H(4,9),
     *          WAH(4),WBH(4),WCH(4),WDH(4),WEH(4),WFH(4),W2H(4),W5H(4)
      DIMENSION S(NP),NPTA(M)

      IF(BNDISC.EQ.0.)GO TO 4
      CALL POSITN(ID1FP,2,ID1,331)
      READ(ID1)
      READ(ID1) S,NPTA
    4 CONTINUE
      CNAF = .25*RDN      $     CNAG =      RDS
      CNBF = .25*RDN      $     CNBG =      RDS
      CNCF =     RDN
      CNDF = .25*RDN
      CNEF =     RDN
      CNFF = .25*RDN
      CN5F = .50*RDN
      CN2F = .50*RDN      $     CN2G = .50*RDS

      DO 100 I=1,NP

      IF(BNDISC.EQ.0) GO TO 6
      DO 5 L=2,M
      IF(I.EQ.NPTA(L)) GO TO 100
```

```
    5 CONTINUE
    6 CONTINUE

      CS = COSN(2,I)
      SN = SINN(2,I)
      AL = 0.
      ALD = CS*XD(I) + SN*YD(I)
      ALDD = CS*XDD (I) + SN*YDD(I)
      AGD = CS*YD(I) - SN*XD(I)
      AGDD = CS*YDD(I) - SN*XDD(I)

      IF(BNDPRN.NE.0)
     *WRITE(IW,201) I,ALD,ALDD,AGD,AGDD,
     *               COSN(1,I),COSN(2,I),COSN(3,I),
     *               SINN(1,I),SINN(2,I),SINN(3,I),
     *               CUR(1,I),CUR(2,I),CUR(3,I)
  201 FORMAT(///* BOUNDARY POINT *,I4,/* ALD,ALDD,AGD,AGDD *,4E14.4,
     *          /* COSN1,COSN2,COSN3 *,3E14.4,
     *          /* SINN1,SINN2,SINN3 *,3E14.4,
     *          /* CUR1,CUR2,CUR3    *,3E14.4                )

      DO 10 J=1,9
      JM1 = J - 1
      J1 = 1 + MOD(JM1,3)
      ETA = DN*FLOAT(JM1/3)
      CALL WFGH(WCP(1,J,I),COSN(J1,I),SINN(J1,I),CUR(J1,I),
     *          ETA,AL,ALD,ALDD,W(1,J),F(1,J),G(1,J),H(1,J))
   10 CONTINUE

      IF(BNDPRN.NE.0)
     *WRITE(IW,202) ((W(K,J),F(K,J),G(K,J),H(K,J),K=1,4),J=1,9)
  202 FORMAT(/* W,F,G,H */(4E14.4))

      CUR2 = CUR(2,I)
      CNCG = .25*RDS/(1. +  0.5*DN*CUR2)
      CNCG =     RDS/(1. +  0.5*DN*(CUR2  +  CUR(3,I)))
      CNEG = .25*RDS/(1. +  1.5*DN*CUR2)
      CNFG =     RDS/(1. +  0.5*DN*(CUR2  +  CUR(1,I)))
```

```
      CN5G = .50*RDS/(1. +       DN*CUR2)
      DTH = .5*DT
      CSBAR = (CUR(1,I) - CUR(3,I))*.5*RDS
      CDN   = DTH*RDS*AGD

      DO 40 K=1,4
      W1 = W(K,1)   $  F1 = F(K,1)   $  G1 = G(K,1)   $  H1 = H(K,1)
      W2 = W(K,2)   $  F2 = F(K,2)   $  G2 = G(K,2)   $  H2 = H(K,2)
      W3 = W(K,3)   $  F3 = F(K,3)   $  G3 = G(K,3)   $  H3 = H(K,3)
      W4 = W(K,4)   $  F4 = F(K,4)   $  G4 = G(K,4)   $  H4 = H(K,4)
      W5 = W(K,5)   $  F5 = F(K,5)   $  G5 = G(K,5)   $  H5 = H(K,5)
      W6 = W(K,6)   $  F6 = F(K,6)   $  G6 = G(K,6)   $  H6 = H(K,6)
      W7 = W(K,7)   $  F7 = F(K,7)   $  G7 = G(K,7)   $  H7 = H(K,7)
      W8 = W(K,8)   $  F8 = F(K,8)   $  G8 = G(K,8)   $  H8 = H(K,8)
      W9 = W(K,9)   $  F9 = F(K,9)   $  G9 = G(K,9)   $  H9 = H(K,9)

      WCP15 = W(1,5)
      WCP25 = W(2,5)
      WCP35 = W(3,5)
      WCP45 = W(4,5)
      WCP12 = W(1,2)
      WCP32 = W(3,2)
      WCP42 = W(4,2)

      WFH(K)=.5*(W5+W4)-DTH*(CNFF*(F7+F8-F1-F2)+CNFG*(G4-G5)+.5*(H5+H4))
     *          + CDN*(W4-W5)
      WDH(K)=.5*(W5+W6)-DTH*(CNDF*(F8+F9-F2-F3)+CNDG*(G5-G6)+.5*(H5+H6))
     *          + CDN*(W5-W6)
      WEH(K)=.5*(W5+W8)-DTH*(CNEG*(G7+G4-G9-G6)+CNEF*(F8-F5)+.5*(H5+H8))
     *          + .25*CDN*(W7 + W4 - W9 - W6 )
      WCH(K)=.5*(W5+W2)-DTH*(CNCG*(G4+G1-G6-G3)+CNCF*(F5-F2)+.5*(H5+H2))
     *          + .25*CDN*(W4 + W1 - W3 - W6 )
      W5H(K)=   (W5   )-DTH*(CN5G*(G4   -G6   )+CN5F*(F8-F2)+   (H5   ))
     *          + .50*CDN*(W4 - W6)
      IF(K.EQ.2)GO TO 40
      WAH(K)=.5*(W2+W1)-DTH*(CNAF*(F1+F2+F4+F5)+CNAG*(G1-G2)+.5*(H2+H1))
     *          +     CDN*(W1 - W2)
      WBH(K)=.5*(W2+W3)-DTH*(CNBF*(F2+F3+F5+F6)+CNBG*(G2-G3)+.5*(H2+H3))
```

```
     *          +      CCN*(W2 - W3)
      W2H(K)=    (W2    )-DTH*(CN2F*(F2    +F5    )+CN2G*(G1-G3)+    (H2    ))
     *          + .50*CCN*(W1 - W3)
   40 CONTINUE
      W2H(2)=.5*WCH(2)

      IF(BNDPRN.NE.0)
     *WRITE(IW,203) (W2H(K),W5H(K),WAH(K),WBH(K),
     *               WCH(K),WDH(K),WEH(K),WFH(K),K=1,4)
  203 FORMAT(/* HALF TIME-STEP RESULTS */(8E14.4))

      ALH = ALD*DTH + .5*ALDD*DTH*DTH
      ALDH = ALD + ALDD*DTH
      AGH = AGD*DTH + .5*AGDD*DTH*DTH
      AGDH = AGD + AGDD*DTH
      CCNH = CT*AGDH*RDS
      CUR2 = CUR2 + CSBAR*AGDH

      W1=WCH(1)  $  W2=WCH(2)  $  W3=WCH(3)  $  W4=WCH(4)  $  RW1=1./W1
      W1=ABS(W1)
      FC1 = W2
      FC2 = CT1*(W1**CT2)*W4 + W2*W2*RW1
      FC3 = W2*W3*RW1
      FC4 = W2*W4*RW1

      W1=WEH(1)  $  W2=WEH(2)  $  W3=WEH(3)  $  W4=WEH(4)  $  RW1=1./W1
      W1=ABS(W1)
      FE1 = W2
      FE2 = CT1*(W1**CT2)*W4 + W2*W2*RW1
      FE3 = W2*W3*RW1
      FE4 = W2*W4*RW1

      W1=WAH(1)  $  W3=WAH(3)  $  W4=WAH(4)  $  RW1=1./W1
      WA1=WAH(1) $ WA2=WAH(2) $ WA3=WAH(3) $ WA4=WAH(4)
      W1=ABS(W1)
      GA1 = W3
      GA3 = CT1*(W1**CT2)*W4 + W3*W3*RW1
      GA4 = W3*W4*RW1
```

```
W1=WBH(1)  $  W3=WBH(3)  $  W4=WBH(4)  $  RW1=1./W1
WB1=WBH(1) $ WB2=WBH(2) $ WB3=WBH(3) $ WB4=WBH(4)
W1=ABS(W1)
GB1 = W3
GB3 = CT1*(W1**CT2)*W4 + W3*W3*RW1
GB4 = W3*W4*RW1

W1=WDH(1)  $  W2=WDH(2)  $  W3=WDH(3)  $  W4=WDH(4)  $  RW1=1./W1
WD1=WDH(1) $ WD2=WDH(2) $ WD3=WDH(3) $ WD4=WDH(4)
W1=ABS(W1)
GD1 = W3
GD2 = W2*W3*RW1
GD3 = CT1*(W1**CT2)*W4  + W3*W3*RW1
GD4 = W3*W4*RW1

W1=WFH(1)  $  W2=WFH(2)  $ W3=WFH(3)  $  W4=WFH(4)  $  RW1=1./W1
WF1=WFH(1) $ WF2=WFH(2) $ WF3=WFH(3) $ WF4=WFH(4)
W1=ABS(W1)
GF1 = W3
GF2 = W2*W3*RW1
GF3 = CT1*(W1**CT2)*W4 + W3*W3*RW1
GF4 = W3*W4*RW1


FAC5 = 1./(1. + CLR2*(ALH + DN))
FACT5 = FAC5*CLR2
UN = W5H(2)/W5H(1) + ALDH
H51 = FACT5*UN*(W5H(1))
H52 = FACT5*   (UN*W5H(2) - W5H(3)*W5H(3)/W5H(1)) + W5H(1)*ALDD
H53 = FACT5*UN*(2.*W5H(3))
H54 = FACT5*UN*(W5H(4))

FAC2 = 1./(1. + CLR2*ALH)
FACT2 = FAC2*CLR2
UN = W2H(2)/W2H(1) + ALDH
H21 = FACT2*UN*(W2H(1))
H23 = FACT2*UN*(2.*W2H(3))
```

```
      H24 = FACT2*UN*(W2H(4))
      DTHF = DT
      FACT2 = FAC2*RDS
      FACT5 = FAC5*RDS

      W51 = WCP15      -DTHF*((FE1-FC1)*RDN + FACT5*(GF1-GD1) + H51)
     *          + CONH*(WF1-WC1)
      W52 = WCP25      -DTHF*((FE2-FC2)*RDN + FACT5*(GF2-GD2) + H52)
     *          + CONH*(WF2-WC2)
      W53 = WCP35      -DTHF*((FE3-FC3)*RDN + FACT5*(GF3-GD3) + H53)
     *          + CONH*(WF3-WC3)
      W54 = WCP45      -DTHF*((FE4-FC4)*RDN +FACT5*(GF4-GD4) + H54)
     *          + CONH*(WF4-WC4)

      IF(BNDPRN.NE.0)
     *WRITE(6,205) ALH,ALDH,FC1,FC2,FC3,FC4,FE1,FE2,FE3,FE4,
     *                      GA1,GA2,GA3,GA4,GB1,GB2,GB3,GB4,
     *                      GD1,GD2,GD3,GD4,GF1,GF2,GF3,GF4,
     *                      H51,H52,H53,H54,H21,H22,H23,H24
  205 FORMAT(//* ALH,ALDH *,2E14.4,
     *       //* HALF TIME LEVEL F,G,H *,/(4E14.4) )

      WC(1,I) = WCP12      -DTHF*( FC1*RDN  + FACT2*(GA1-GB1) + H21)
     *          + CONH*(WA1-WB1)
      WC(2,I) = W52/3.
      WC(3,I) = WCP32      -DTHF*( FC3*RDN  + FACT2*(GA3-GB3) + H23)
     *          + CONH*(WA3-WB3)
      WC(4,I) = WCP42      -DTHF*( FC4*RDN  + FACT2*(GA4-GB4) + H24)
     *          + CONH*(WA4-WB4)

      IF(BNDPRN.NE.0)
     *WRITE(IW,204) (WC(K,I),K=1,4)
  204 FORMAT(/* FULL TIME-STEP RESULTS */4E14.4)

      WS(1,I) = .5*(3.*WC(1,I) - W51)
      WS(3,I) = .5*(3.*WC(3,I) - W53)
      WS(4,I) = .5*(3.*WC(4,I) - W54)
      WS(2,I) = .0
```

```
      ALD = ALD + DT*ALDD
      CALL BWTCW ( WC(1,I),WC(1,I),CS,SN,ALD )
      CALL BWTOW ( WS(1,I),WS(1,I),CS,SN,ALD )
  100 CONTINUE

      IF(BNDISC.EQ.0.)GO TO 410
      DO 400 I=1,NP
      DO 200 L=2,M
      IF(I.EQ.NPTA(L)) GO TO 300
  200 CONTINUE
      GO TO 400
  300 RR=1./(S(I+1)-S(I-1))
      RAT1=(S(I)-S(I-1))*RR
      RAT2=(S(I+1)-S(I))*RR
      DO 350 K=1,4
      WC(K,I) = RAT2*WC(K,I-1) + RAT1*WC(K,I+1)
  350 WS(K,I) = RAT2*WS(K,I-1) + RAT1*WS(K,I+1)
  400 CONTINUE
  410 CONTINUE

      RETURN
      END
```

```
      SUBROUTINE BNDWRI ( W1, W2, W3, W4,
     *                    PRES, TEMP, VSURF, AMLOCL, CP,
     *                COSN,SINN,   WC,WS,  WP4,  NP )
C PRINTS BOUNDARY VALUES
      COMMON / CM3A   / CT1,CT2,CT3,CT4,CT5,CT6,CT7,CT8,CT9
      COMMON / CM4    / CF,DTMX, DXYSN, DTFRST,DTOUT,DTBND,DTSTB,DT
      COMMON / CM11   / IR,IW
      COMMON / CM11B  / ID1,ID2,IL1,IL2,   ID1FP,ID2FP,IL1FP,IL2FP
      DIMENSION W1(NP), W2(NP), W3(NP), W4(NP),
     *          PRES(NP), TEMP(NP), VSURF(NP), AMLOCL(NP), CP(NP),
     *          COSN(3,NP), SINN(3,NP),   WC(4,NP), WS(4,NP)
      DIMENSION WP4(NP)
      RDT = 1./DT
      K = 1
      DMAX=0.

      CALL POSITN(ID2FP,2,IC2,353)
      READ(ID2)
      READ(ID2) WP4

   10 CONTINUE
      DO 100 I=1,NP
      IF ( K.EQ.2 ) GO TO 20
      RO = W1(I) = WC(1,I)
      RU = W2(I) = WC(2,I)
      RV = W3(I) = WC(3,I)
      E  = W4(I) = WC(4,I)
      WP4(I) = RDT*(W4(I) - WP4(I))
      GO TO 30
   20 RO = W1(I) = WS(1,I)
      RU = W2(I) = WS(2,I)
      RV = W3(I) = WS(3,I)
      E  = W4(I) = WS(4,I)
   30 CONTINUE
      RR = 1./RO
      U = RR*RU
      V = RR*RV
      VEL2 = U*U + V*V
```

```
      VEL = SQRT(VEL2)
      PRES(I) = E + CT3*RC*VEL2
      VSURF(I) = V*CCSN(2,I) - U*SINN(2,I)
      CP(I) = CT8*(PRES(I) - 1.)
      PRES(I) = 1. + CT9*(PRES(I)-1.)
      ROUN    = 1. + CT9*(W1(I)  -1.)
      TEMP(I) = PRES(I)/ROUN
      TTMMP = ABS(TEMP(I))
      AMLCCL(I) = CT7*VEL/SQRT(TTMMP)

      IF(K.EQ.2) GO TO 100

      IND=0
      IF(W1(I).LT.0.)    IND=1
      IF(PRES(I).LT.0.)  IND=1
      IF(TEMP(I).LT.0.)  IND=1
      IF(IND.EQ.0) GO TO 40
      RC = W1(I) = WC(1,I) = .5*(WC(1,I-1)+WC(1,I+1))
      RU = W2(I) = WC(2,I) = .5*(WC(2,I-1)+WC(2,I+1))
      RV = W3(I) = WC(3,I) = .5*(WC(3,I-1)+WC(3,I+1))
      E  = W4(I) = WC(4,I) = .5*(WC(4,I-1)+WC(4,I+1))
      GO TO 100
   40 CONTINUE
      TEMPP = E/RC + CT3*(RU*RU+RV*RV)/(RC*RC)
      IF(TEMPP.LT.0.) GO TO 100
      CNCW = CT4*SQRT(TEMPP) + VEL
      CMAX = AMAX1(CMAX,CNCW)

  100 CONTINUE
      CTRNC = DXYSN/CMAX
      IF(K.EQ.1) WRITE(IW,610)
      IF(K.EQ.2) WRITE(IW,620)
      IF(K.EQ.1)
     *WRITE(IW,631) ( (I,W1(I),W2(I),W3(I),W4(I),PRES(I),
     * TEMP(I),VSURF(I),AMLCCL(I),CP(I),WP4(I),I),I=1,NP)
      IF(K.EQ.2)
     *WRITE(IW,632) ( (I,W1(I),W2(I),W3(I),W4(I),PRES(I),
     * TEMP(I),VSURF(I),AMLCCL(I),CP(I)       ,I),I=1,NP)
```

```
      IF(K.EQ.2) GO TO 99
      K=2
      GO TO 10
  610 FORMAT(1HO, 2X1HI , 9X3HWC1 , 9X3HWC2 , 9X3HWC3 , 9X3HWC4 ,
     *8X4HPRES , 8X4HTEMP , 7X5HVTANG , 7X5HMLCCL , 10X2HCP ,
     * 5X7HDW4BYDT , 2X1HI )
  620 FORMAT(1HO,/3X1HI , 9X3HWS1 , 9X3HWS2 , 9X3HWS3 , 9X3HWS4 ,
     *8X4HPRES , 8X4HTEMP , 7X5HVTANG , 7X5HMLCCL , 10X2HCP , 2X1HI )
  631 FORMAT(1X,I3,10F12.6,I3)
  632 FORMAT(1X,I3, 9F12.6,I3)
   99 CONTINUE
      CALL POSITN(ID1FP,1,ID1,351)
      WRITE(ID1) WC
      CALL ENDFL(ID1FP,ID1)
      RETURN
      END
```

```
      SUBROUTINE PCL3(WC,WR,WS,ZX,ZY,WH,WV,ZH,ZV,WANS)
C INTERPOLATES FOR W AT POINT IN A CELL WHEN ONE POINT OF CELL IS INSIDE CONTOUR
      DIMENSION WQ(4),WR(4),WS(4),WA(4),WB(4),WC(4),WD(4),WANS(4)
      DIMENSION WH(4),WV(4)
      C1 = 1. - ZY
      C2 = 1. - ZX
      C3 =1./(1. - ZV)
      C4 =1./(1. - ZH)
      C6 =C1*C3
      C7 =(ZY-ZV)*C3
      C8 =C2*C4
      C9 =(ZX-ZH)*C4
      DO 10 K=1,4
      WA(K) = ZY*WR(K) + C1*WC(K)
      WB(K) = ZX*WR(K) + C2*WS(K)
      WC(K) = C6*WV(K) + C7*WS(K)
      WD(K) = C8*WH(K) + C9*WC(K)
   10 WANS(K) = .5*(C1*WD(K) + ZY*WB(K) + C2*WC(K) + ZX*WA(K))

      RETURN
      END
```

```
      SUBROUTINE WFGH(WW,CC,SS,CUR,ETA,AL,ALD,ALDD,W,F,G,H)

C COMPUTES W,F,G,H IN LOCAL CO-ORDS FROM W IN CARTESIAN CO-ORDS

      DIMENSION WW(4),W(4),F(4),G(4),H(4)
      COMMON / CM3A   / CT1,CT2,CT3,CT4,CT5,CT6,CT7,CT8
      W1 = WW(1)
      RU = WW(2)
      RV = WW(3)
      RW1 = 1./W1
      RUN =  RU*CC + RV*SS
      RUS = -RU*SS + RV*CC
      UN  = RUN*RW1
      US  = RUS*RW1
      UNP = UN - ALD
      RUNP = W1*UNP

      P = WW(4) + CT3*(RU*RU + RV*RV)*RW1
      PP = CT1*P
      S = P*(ABS(RW1))**CT5
      CN = CUR/(1. + CUR*(AL + ETA))
      F2 = RUNP*UNP
      F3 = RUNP*US
      G3 = RUS*US

      W(1) = W1
      W(2) = RUNP
      W(3) = RUS
      W(4) = W1*S
      F(1) = RUNP
      F(2) = PP + F2
      F(3) = F3
      F(4) = RUNP*S
      G(1) = RUS
      G(2) = F3
      G(3) = PP + G3
      G(4) = RUS*S
      H(1) = CN*RUN
```

```
      F(2) = CN*(RUN*LAP - G3) + W1*ALCC
      F(3) = CN*(2.*US*RUN)
      F(4) = CN*(RUN*S)
      RETURN
      END
```

```
      SUBROUTINE  BWTOW(BW,W,CC,SS,ALD)
C COMPUTES *W* IN FIXED CARTESIAN CO-ORDS , GIVEN *BW* IN MOVING BODY CO-ORDS
      COMMON / CM3A   / CT1,CT2,CT3,CT4,CT5,CT6,CT7,CT8
      DIMENSION BW(4),W(4)
      W1 = BW(1)
      RW1 = 1./W1
      UNP = RW1*BW(2)
      UN = UNP + ALD
      US = RW1*BW(3)
      U  = UN*CC - US*SS
      V  = UN*SS + US*CC
      P = BW(4)*(ABS(W1))**CT2
      W(1) = W1
      W(2) = W1*U
      W(3) = W1*V
      W(4) = P - CT3*W1*(U*U + V*V)
      RETURN
      END
```

```
      OVERLAY(OVER,33,0)
      PROGRAM BNDRC
      COMMON / CM4A   / ICT,ICTFP,T1,T2,NC1,NCURV
      COMMON / CM4B   / TPREV,T,INDCT,NT,NTFRST
      COMMON / CM8A   / NCTMX,NBTMX,NX,NY,KX,KY,KW,NBT,MM
      COMMON / CM10   / IOCTL,TMX,NTMX,TMR
      COMMON / CM10A  / TIMA(2),TIMB(2),TIMS(2),TIMT(2),TIM(2,8)
      COMMON / CM11   / IR,IW
      COMMON / CM999  / NN,A(1)
      IF(NT.EQ.0) GO TO 999
      MM1 = MM + NY
      NP = NC1
      N1 = 1
      N2 = N1 + 12*NP
      N3 = N2 +  3*NP
      N4 = N3 +  3*NP
      N5 = N4 +  3*NP
      N6 = N5 +  3*NP
      N7 = N6 +  6*NP
      N8 = N7 +  6*NP
      N9 = N8 +    NP
      N10 = N9 + NP
      N11 = N10 + NP
      N12 = N11 + NP
      N13 = N12 + NP
      N14 = N13 + NP
      N15 = N14 + NP
      N16 = N15 + NP
      N17 = N16 + NP
      N18 = N17 + NP

      CALL BNDGCC ( A(N1),A(N2),A(N3),A(N4),A(N5),
     *              A(N6),A(N7),A(N8),A(N9),
     *              A(N10),A(N11),A(N12),A(N13),A(N14),A(N15),
     *              A(N16),A(N17),A(N18),  NP )

      N1 = 1
      N2 = N1 + 4*NP
```

```
      N3 = N2 + MM1
      N4 = N3 + MM1
      N5 = N4 + MM1
      N6 = N5 + MM1
      N7 = N6 + NX
      N8 = N7 + KX*NX
      N9 = N8 + NY
      N10 = N9 + KY*NY
      N11 = N10 +KX*NX

      CALL BNDINC ( A(N1),A(N2),A(N3),A(N4),A(N5),A(N6),A(N7),
     *              A(N8),A(N9),A(N10),A(N11), MM1,NP,KX,NX,KY,NY )

      N1 = 1
      N2 = N1 + 36*NP
      N3 = N2 +  3*NP
      N4 = N3 +  3*NP
      N5 = N4 +  3*NP
      N6 = N5 + 3*NP
      N7 = N6 +   NP
      N8 = N7 +   NP
      N9 = N8 + NP
      CALL BNDREC ( A(N1),A(N2),A(N3),A(N4),A(N5),
     *         A(N6),A(N7),A(N8),A(N9)    , NP)

      N10 = N9 + NP
      N11 = N10 + 4*NP
      N12 = N11 + NP
      CALL BND3 (A(N1),A(N2),A(N3),A(N4),A(N5),
     *     A(N6),A(N7),A(N8),A(N9),
     *        A(N10),A(N11),A(N12)  ,  NP,NCURV )
      N12 = N10
      N11 = N3
      N10 = N2
      N1  = 1
      N2  = N1 + NP
      N3  = N2 + NP
      N4  = N3 + NP
```

```
      N5  = N4 + NP
      N6  = N5 + NP
      N7  = N6 + NP
      N8  = N7 + NP
      N9  = N8 + NP
      N13 = N9 + NP
      CALL BNDWRC( A(N1),A(N2),A(N3),A(N4),
     *             A(N5),A(N6),A(N7),A(N8),A(N9),
     *          A(N10),A(N11),A(N12),A(N13), NP)
  999 CONTINUE

      CALL TIMER(TMR,TIMB,TIMA,TIM(1,3))
      WRITE(IW,6090) TIM(1,3),TIM(2,3),T,NT
 6090 FORMAT(1HO,* CP,PP TIMES FOR  $ BNDRY  $  ARE (SEC) *,2F10.2,5X
     $*T = * F10.6, 4X * NT = * I6 )

      END
```

```
      SUBROUTINE BNDGOC( WCP, COSN,SINN,CUR,VDEL,
     *                   X,Y,XD,YD,
     *                   XP,YP,DP,CSP,SNP,C1P,C2P,C3P,SP,  NP )

C FINDS CO-ORDS OF 6*NP POINS.   SORTS THEM OUT.   STORES THEM FOR BNDINT
C FINDS (W) FOR 2*NP POINTS AND STORES THEM IN *WCP1,WCP3*
C COMPUTES COSN,SINN,CUR


      COMMON / CM2    / ZNX,ZNY,YZ,DX,DY,DS,DN,   RDX,RDY,RDS,RDN
      COMMON / CM7    / TOLS,TOLX,TOLCB,TOLCF,TOLSOL
      COMMON / CM8A   / NCTMX,NBTMX,NX,NY,KX,KY,KW,NBT,MM
      COMMON / CM8B   / MS,ML,KBLKS
      COMMON / CM11   / IR,IW
      COMMON / CM11B  / ID1,ID2,IL1,IL2,   ID1FP,ID2FP,IL1FP,IL2FP

      DIMENSION WCP(4,3,NP),VDEL(3,NP),
     *          X(6,NP),Y(6,NP),
     *          XD(NP),YD(NP),
     *          XP(NP),YP(NP),DP(NP),CSP(NP),SNP(NP),
     *          C1P(NP),C2P(NP),C3P(NP),SP(NP),
     *          COSN(3,NP),SINN(3,NP),CUR(3,NP)
      COMMON / BNDLCL / INDB(3)

C READ CONTP   ID1(2)

      CALL POSITN ( ID1FP,2,ID1,311 )
      READ(ID1) XP,YP,DP,CSP,SNP,C1P,C2P,C3P,SP
      CALL POSITN(ID1FP,3,ID1,3111)
      READ(ID1) XD,YD
C READ WCP2    ID1(4)
      CALL POSITN ( ID1FP,4,ID1,312 )
      READ(ID1) ((WCP(K,2,I),K=1,4),I=1,NP)

C FIND X,Y CO-ORDS OF POINTS 5,8 CORRESPONDING TO EACH CONTOUR POINT

      DO 9C I=1,NP
      IF(I.EQ.NP) GO TO 10
```

```
      CALL NRMCUR(0.,CSP(I),SNP(I),C1P(I),C2P(I),C3P(I),CS,SN,CUR(2,I))
      IF(I.EQ.1) GO TO 20
      I1 = I - 1
      CALL NRMCUR( DP(I1),CSP(I1),SNP(I1),C1P(I1),C2P(I1),C3P(I1),
     * DUM,DUM,CURR)
      CUR(2,I) = .5*(CUR(2,I) + CURR )
      GO TO 20
   10 N = NP - 1
      CALL NRMCUR(DP(N),CSP(N),SNP(N),C1P(N),C2P(N),C3P(N),CS,SN,
     *           CUR(2,I))
      CS = 0.
      SN = 1.
      GO TO 30
   20 IF(I.NE.1) GO TO 30
      CS = 0.
      SN = -1.
   30 DNCOS = DN*CS
      DNSIN = DN*SN
      XX    = XP(I)
      YY    = YP(I)
      X(2,I) = XX + DNCOS
      X(5,I) = XX + 2.*DNCOS
      Y(2,I) = YY + DNSIN
      Y(5,I) = YY + 2.*DNSIN
      COSN(2,I) = CS
      SINN(2,I) = SN
      VDEL(2,I) = XD(I)*CS + YD(I)*SN
   90 CONTINUE

      DO 290 L=1,3
      GO TO (101,102,103),L
  101 CON = 1.
      JXY1= 1
      JXY2= 4
      JV = 1
      INDB(1) = NP
      GO TO 110
  102 GO TO 290
```

```
  103 CON = -1.
      JXY1=3
      JXY2=6
      JV = 3
      INDB(3)=1
  110 CONTINUE
      DO 190 I=1,NP
      ARC = SP(I) + CON*DS
      CALL ALL( NP,XP,YP,DP,CSP,SNP,C1P,C2P,C3P,SP,TOLS,TOLX,
     *          ARC, J,S1,S2,SGN,CS,SN,CURV,XX,YY )
      J1 = J + 1
      CUR(L,I) = CURV
      DNCOS = DN*CS
      DNSIN = DN*SN
      X(JXY1,I) = XX + DNCOS
      X(JXY2,I) = XX + 2.*DNCOS
      Y(JXY1,I) = YY + DNSIN
      Y(JXY2,I) = YY + 2.*DNSIN
      DO 150 K=1,4
  150 WCP(K,L,I) = S2*WCP(K,2,J) + S1*WCP(K,2,J1)
      XDEL = S2*XD(J) + S1*XD(J1)
      YDEL = S2*YD(J) + S1*YD(J1)
      VDEL(JV,I) = XDEL*CS + YDEL*SN
      IF(SGN.EQ.1.)GO TO 180
      CS = -CS
      WCP(2,L,I) = -WCP(2,L,I)
      IF(L.EQ.1) INDB(L) = MINO(INDB(L),I)
      IF(L.EQ.3) INDB(L) = MAXO(INDB(L),I)
  180 CONTINUE
      COSN(L,I) = CS
      SINN(L,I) = SN
  190 CONTINUE
  290 CONTINUE

C WRITE WCP,COSN,SINN,CUR,VDEL
      CALL POSITN ( IO2FP,2,IO2,313 )
      WRITE(IO2) (((WCP(K,J,I),K=1,4),I=1,NP),J=1,3),COSN,SINN,CUR,VDEL
      WRITE(IO2)(WCP(4,2,I),I=1,NP)
```

```
      CALL ENDFL(IO2FP,IO2)

C SORT OUT POINTS AND WRITE (I,J,X,Y) ON IO2(3)
      NPTS = 0
      IBLKND = -1
      DO 700 KB = 1,KBLKS
      M = MS
      IF(K.EQ.KBLKS) M = ML
      KLEFT = (KB-1)*MS
      IF(KB.EQ.1) KLEFT = 1
      KRGHT = KB*MS
      XLEFT = FLOAT(KLEFT - 1)*DX
      XRGHT = FLOAT(KRGHT - 1)*DX
      DO 600 I=1,NP
      DO 600 J=1,6
      XX = X(J,I)
      IF(XX.LT.XLEFT) GO TO 600
      IF(XX.GE.XRGHT) GO TO 600
      JJ = J + 3
      NPTS = NPTS + 1
      WRITE(IO2) JJ,I,XX,Y(J,I)
  600 CONTINUE
      WRITE(IO2) IBLKND,IBLKND,IBLKND,IBLKND
  700 CONTINUE

      CALL ENDFL(IO2FP,IO2)
      NP6 = 6*NP
      IF(NPTS.EQ.NP6) GO TO 800
      WRITE(IW,750) NPTS,NP6
  750 FORMAT(1HO,* ERROR IN BNDGEO   NPTS,NP6 ARE * 2I6 )
      CALL EXIT
  800 CONTINUE
      RETURN
      END
```

```
      SUBROUTINE BNDINC(WCP,W1,W2,W3,W4,IX,VX,IY,VY,NSX,NSY,
     *                  MM1,NP,KX,NX,KY,NY )

C THIS SUBROUTINE COMPUTES *W* AT POINTS 4,5,6,7,8,9 OF THE 9 POINT GRID
C    ASSOCIATED WITH EACH BOUNDARY POINT

C STATEMENT FUNCTIONS

      XMESH(I) = FLOAT(I-1)*DX
      YMESH(J) = FLOAT(J-1)*DY - YZ

      COMMON / CM2    / ZNX,ZNY,YZ,DX,DY,DS,DN,   RDX,RDY,RDS,RDN
      COMMON / CM8B   / MS,ML,KBLKS
      COMMON / CM11   / IR,IW
      COMMON / CM11B  / ID1,ID2,IL1,IL2,   ID1FP,ID2FP,IL1FP,IL2FP

      DIMENSION W1(MM1),W2(MM1),W3(MM1),W4(MM1),WCP(4,NP),
     *          IX(NX),VX(KX,NX),IY(NY),VY(KY,NY),NSX(KX,NX),NSY(KY,NY)
      DIMENSION WP(4),WQ(4),WR(4),WS(4),WH(4),WV(4),WANS(4)
      COMMON / BNDLCL / INDB(3)

C READ * IX,VX,IY,VY,NSX,NSY *  FROM ID1(5)

      CALL POSITN ( ID1FP,5,ID1,321 )
      READ(ID1) IX,VX,IY,VY,NSX,NSY

C POSITION *IL1* TO READ  *W1,W2,W3,W4*
C          *ID2* TO READ  *I,J,X,Y*
C          *ID1* TO WRITE *I,J, WANS(4)

      CALL POSITN ( ID2FP,3,ID2,322 )
      CALL POSITN ( IL1FP,1,IL1,323 )
      CALL POSITN ( ID1FP,4,ID1,324 )
      READ(ID1)WCP
      CALL POSITN ( ID1FP,4,ID1,325)

C THIS IS THE CC LOOP OVER THE LAX-WND BLOCKS
```

```
      DO 900 KB=1,KBLKS

C READ * W1,W2,W3,W4 *

      M = MS
      IF ( KB.EQ.KBLKS ) M=ML
      K1 = 1
      K2 = NY
      K3 = K2 + 1
      K4 = (M + 1)*NY
      IF(KB.EQ.1) GO TO 2
      READ(IL1) (W1(I),I=K1,K2),
     *          (W2(I),I=K1,K2),
     *          (W3(I),I=K1,K2),
     *          (W4(I),I=K1,K2)
      CALL FSR(1,IL1,IERR)
    2 READ(IL1) (W1(I),I=K3,K4),
     *          (W2(I),I=K3,K4),
     *          (W3(I),I=K3,K4),
     *          (W4(I),I=K3,K4)
      I1 = (KB-1)*MS
C READ  IPT,JPT,X,Y FROM  IO2(3)

    5 READ(IO2) JPT,IPT,X,Y
      IF(IPT.LT.0) GO TO 900

C FIND LOWER LEFT HAND CORNER IP,JP OF CELL IN WHICH (X,Y) LIES

      IP = INT(X*RDX) + 1
      JP = INT((Y+YZ)*RDY) + 1
      KP = (IP - I1)*NY + JP
      KQ =KP + NY
      KR =KQ + 1
      KS =KP + 1
      IND = 0
      INDX = 0
   10 WP(1) = W1(KP)
```

```
      WP(2) = W2(KP)
      WP(3) = W3(KP)
      WP(4) = W4(KP)
      IF(WP(4).GE.0.) GO TO 20
      IND = IND + 1
      INDX = 1
   20 WQ(1) = W1(KQ)
      WQ(2) = W2(KQ)
      WQ(3) = W3(KQ)
      WQ(4) = W4(KQ)
      IF(WQ(4).GE.0.) GO TO 30
      IND = IND + 1
      INDX = 2
   30 WR(1) = W1(KR)
      WR(2) = W2(KR)
      WR(3) = W3(KR)
      WR(4) = W4(KR)
      IF(WR(4).GE.0.) GO TO 40
      IND = IND + 1
      INDX = 3
   40 WS(1) = W1(KS)
      WS(2) = W2(KS)
      WS(3) = W3(KS)
      WS(4) = W4(KS)
      IF(WS(4).GE.0.) GO TO 45
      IND = IND + 1
      INDX = 4
   45 IF (IND.LE.1 ) GO TO 60
      WRITE(IW,50)
   50 FORMAT(1HO,*ERROR IN BNDINT*)
      CALL EXIT
   60 IF(IND.EQ.1) GO TO 100
      ZX = (X - XMESH(IP))*RDX
      ZY = (Y - YMESH(JP))*RDY
      ZX1 = 1. - ZX
      ZY1 = 1. - ZY
      CP = ZX1*ZY1
      CQ = ZX * ZY1
```

```
      CR = ZX *ZY
      CS = ZX1*ZY
      DO 70 K=1,4
   70 WANS(K) = CP*WP(K) + CQ*WQ(K) + CR*WR(K) + CS*WS(K)
      GO TO 800

  100 XLO = XMESH(IP)
      XHI = XMESH(IP+1)
      YLO = YMESH(JP)
      YHI = YMESH(JP+1)
      ICORN = IP
      JCORN = JP
      IF( INDX.EQ.2 .OR. INDX.EQ.3) ICORN=IP+1
      IF(INDX.EQ.3 .OR. INDX.EQ.4) JCORN=JP+1
      XCORN = XMESH(ICORN)
      YCORN = YMESH(JCORN)
      ZX = RDX*ABS(X-XCORN)
      ZY = RDY*ABS(Y-YCORN)

C FIND HORIZONTAL INTERSECTION AND WH(K)

      KMAX = IY(JCORN)
      DO 110 K=1,KMAX
      XX = VY(K,JCORN)
      IF(XLO.GT.XX)GO TO 110
      IF(XHI.LT.XX)GO TO 110
      GO TO 120
  110 CONTINUE
      CALL EXIT

  120 NSYN = NSY(K,JCORN)
      IC = MOD(NSYN,10000)
      IC1 = IC + 1
      S1 = (1.E-08)*(FLOAT(NSYN/10000))
      S2 = 1. - S1
      DO 130 K=1,4
  130 WH(K) = S1*WCP(K,IC1) + S2*WCP(K,IC)
      ZH = RDX*ABS(XX-XCORN)
```

```
C FIND VERTICAL INTERSECTION AND WV(K)

      KMAX = IX(ICORN)
      DO 210 K=1,KMAX
      YY = VX(K,ICORN)
      IF(YLO.GT.YY) GO TO 210
      IF(YHI.LT.YY) GO TO 210
      GO TO 220
  210 CONTINUE
      CALL EXIT
  220 NSXN = NSX(K,ICORN)
      IC = MOD(NSXN,10000)
      IC1 = IC + 1
      S1 = (1.E-08)*(FLOAT(NSXN/10000))
      S2 = 1. - S1
      DO 230 K=1,4
  230 WV(K) = S1*WCP(K,IC1) +  S2*WCP(K,IC)
      ZV = RDY*ABS(YY-YCORN)

C CALL  PCLC HERE

      IF(INDX.EQ.1)CALL PCLC(WC,WR,WS,ZX,ZY,WH,WV,ZH,ZV,WANS)
      IF(INDX.EQ.2)CALL PCLC(WR,WS,WP,ZY,ZX,WV,WH,ZV,ZH,WANS)
      IF(INDX.EQ.3)CALL PCLC(WS,WP,WQ,ZX,ZY,WH,WV,ZH,ZV,WANS)
      IF(INDX.EQ.4)CALL PCLC(WP,WQ,WR,ZY,ZX,WV,WH,ZV,ZH,WANS)
  800 CONTINUE
      GO TO (900,900,900,
     *       810,899,830,
     *       810,899,830),JPT
  810 IF(IPT.GE.INDB(1)) GO TO 898
      GO TO 899
  830 IF(IPT.LE.INDB(3)) GO TO 898
      GO TO 899
  898 WANS(2) = - WANS(2)
  899 WRITE(IO1)  JPT,IPT,WANS
      GO TO 5
  900 CONTINUE
```

```
      CALL ENDFL(IC1FP,IC1)

      RETURN
      END
```

```
      SUBROUTINE BNDREC( WCPP,COSN,SINN,CUR,VDEL,
     *                   XC,YC,XCC,YCC,   NP )

C FILLS IN ARRAYS IN READINESS FOR BOUNDARY UPDATE SCHEMES

      COMMON / CM11B  / ID1,ID2,IL1,IL2,   ID1FP,ID2FP,IL1FP,IL2FP
      DIMENSION WCPP(4,5,NP),COSN(3,NP),SINN(3,NP),CUR(3,NP),VDEL(3,NP)
     *          ,XC(NP),YC(NP),XCC(NP),YCC(NP)

      CALL POSITN( ID2FP,2,ID2,331 )
      READ(ID2) ((WCPP(K,1,I),K=1,4),I=1,NP),
     *          ((WCPP(K,2,I),K=1,4),I=1,NP),
     *          ((WCPP(K,3,I),K=1,4),I=1,NP),
     *          COSN,SINN,CUR,VDEL

      CALL POSITN(ID1FP,3,ID1,332)
      READ(ID1) XC,YC,XCC,YCC

      CALL POSITN ( ID1FP,4,ID1,333 )
   10 READ(ID1) J,I,WCPP(1,J,I),WCPP(2,J,I),WCPP(3,J,I),WCPP(4,J,I)
      IF(ECF,ID1) 20,10
   20 ID1FP = 5

      RETURN
      END
```

```
      SUBROUTINE BND3( WCP,COSN,SINN,CUR,VDEL,
     *                 XD,YD,XDD,YDD,
     *                 WC,SS,NPTA,   NP,M )
C IMPLEMENTS METHOD OF CHARACTERISTICS
      COMMON / CM2    / ZNX,ZNY,YZ,DX,DY,DS,DN,   RDX,RDY,RDS,RDN
      COMMON / CM3A   / CT1,CT2,CT3,CT4,CT5,CT6,CT7,CT8
      COMMON / CM4    / CF,DTMX, DXYSN, DTFRST,DTOUT,DTBND,DTSTB,DT
      COMMON / CM5    / BND,BNDISC,BNDPRN
      COMMON / CM11   / IR,IW
      COMMON / CM11B  / ID1,ID2,IL1,IL2,   ID1FP,ID2FP,IL1FP,IL2FP
      DIMENSION  WCP(4,9,NP),COSN(3,NP),SINN(3,NP),CUR(3,NP),VDEL(3,NP)
      DIMENSION R(15),U(15),V(15),P(15),A(15),S(15),ALS(15)
      DIMENSION WC(4,NP)
      DIMENSION SS(NP),NPTA(M)
      DIMENSION XD(NP),YD(NP),XDD(NP),YDD(NP)

C STATEMENT FUNCTION
      CFUN(C1,C2,C3,Z)=C1+Z*(-1.5*C1+2.*C2-.5*C3+Z*(.5*(C1+C3)-C2))

      DTH = .5*DT
      RDT = 1./DT    $ DTS = DT/DS
      DN2 = 2.*DN    $ DS2 = 2.*DS
      CZ1=1./CT5 $ CZ2=2./CT2 $ CZ3=1./(CT5*CT2) $ CZ4=-.5*CT2
      CZ5=CT6*CT6  $  CZ6=1./CT2
      CZ7 = CT4*CT4
      CZ8 = 1./(CT2*CT6*CT6)

      IF(BNDISC.NE.1.) GO TO 4004
      CALL POSITN(ID1FP,2,ID1,3331)
      READ(ID1)
      READ(ID1) SS,NPTA
 4004 CONTINUE

C THIS IS THE DO LOOP OVER ALL THE BOUNDARY POINTS
      DO 900 I=1,NP
      IF(BNDISC.NE.1.)GO TO 4006
      DO 4005 L=2,M
      IF(I.EQ.NPTA(L)) GO TO 900
```

```
4005 CONTINUE
4006 CONTINUE

     CS = COSN(2,I)
     SN = SINN(2,I)
     AGD= CS*YD(I) - SN*XD(I)
     AGDD = CS*YDD(I) - SN*XDD(I)
     CDERV = (CUR(1,I) - CUR(3,I))*.5/DS
     AGDH = AGD + AGDD*DTH

     DO 9 J = 1,9
     GO TO (1,2,3,1,2,3,1,2,3),J
   1 JJ=1 $ GO TO 9
   2 JJ=2 $ GO TO 9
   3 JJ=3 $ GO TO 9
   4 JJ=4 $ GO TO 9
   5 JJ=5
   9 CALL STATE (  WCP(1,J,I),COSN(JJ,I),SINN(JJ,I),
    *            R(J),U(J),V(J),P(J),A(J),S(J),ALS(J))

     IF(BNDPRN.NE.0.)
    *WRITE(IW,6010)(R(K),U(K),V(K),P(K),A(K),S(K),ALS(K),K=1,15)

     U(10) = .5*(U(1)+U(2))
     R(10) = .5*(R(1)+R(2))
     V(10) = .5*(V(1)+V(2))
     S(10) = .5*(S(1)+S(2))
     A(10) = .5*(A(1)+A(2))

     U(11) = .5*(U(4)+U(5))
     R(11) = .5*(R(4)+R(5))
     V(11) = .5*(V(4)+V(5))
     S(11) = .5*(S(4)+S(5))
     A(11) = .5*(A(4)+A(5))

     U(12) = .5*(U(7)+U(8))
     R(12) = .5*(R(7)+R(8))
     V(12) = .5*(V(7)+V(8))
```

```
      S(12) = .5*(S(7)+S(8))
      A(12) = .5*(A(7)+A(8))

      U(13) = .5*(U(2)+U(3))
      R(13) = .5*(R(2)+R(3))
      V(13) = .5*(V(2)+V(3))
      S(13) = .5*(S(2)+S(3))
      A(13) = .5*(A(2)+A(3))

      U(14) = .5*(U(5)+U(6))
      R(14) = .5*(R(5)+R(6))
      V(14) = .5*(V(5)+V(6))
      S(14) = .5*(S(5)+S(6))
      A(14) = .5*(A(5)+A(6))

      U(15) = .5*(U(8)+U(9))
      R(15) = .5*(R(8)+R(9))
      V(15) = .5*(V(8)+V(9))
      S(15) = .5*(S(8)+S(9))
      A(15) = .5*(A(8)+A(9))

      P(10) = .5*(P(1)+P(2))
      P(13) = .5*(P(2)+P(3))

      ALS(10) =  .5*(ALS(1) + ALS(2))
      ALS(11) =  .5*(ALS(4) + ALS(5))
      ALS(12) =  .5*(ALS(7) + ALS(8))
      ALS(13) =  .5*(ALS(2) + ALS(3))
      ALS(14) =  .5*(ALS(5) + ALS(6))
      ALS(15) =  .5*(ALS(8) + ALS(9))

   50 CONTINUE


C COMPUTATIONS ON RAY A-C-E
      UA=U(10) $ RA=R(10) $ VA=V(10) $ SA=S(10) $ AA=A(10)
      UC=U(11) $ RC=R(11) $ VC=V(11) $ SC=S(11) $ AC=A(11)
      UE=U(12) $ RE=R(12) $ VE=V(12) $ SE=S(12) $ AE=A(12)
```

```
 PA = P(10)
 CA = .5*(CUR(1,I) + CUR(2,I))
 VDELA = .5*(VDEL(1,I) + VDEL(2,I))
 YDELA = .5*(V(10)+VDELA)*DT
 UDELAT = UA*(1. - CA*VA*DT)
*         -DTS*(.5*(U(1)**2 - U(2)**2)
*               + .5*(S(1)+S(2))*CZ8*(R(1)**CT2 - R(2)**CT2))
*    + AGD*DTS*(U(1) - U(2))
 SDELAT = SA - DTS*UA*(S(1)-S(2))
*    + AGD*DTS*(S(1) - S(2))
 GA = (YDELA         )*RDT -(VA - AA)
 GC = (YDELA -    DN )*RDT -(VC - AC)
 GE = (YDELA -    DN2)*RDT -(VE - AE)
 CAC = GA
 CA1 = -1.5*GA + 2.*GC -.5*GE
 CA2 = .5*(GA + GE) - GC
 CALL SMALRT(CAC,CA1,CA2,ZA)
 CNA2 = 1.                    $  CNA1 = CA
 CNC2 = 1./(1.+DN *CA)        $  CNC1 = CA*CNC2
 CNE2 = 1./(1.+DN2*CA)        $  CNE1 = CA*CNE2
 FA = CNA1*(UA*UA+AA*VA) + CNA2*(AA*(R(1)*U(1)-R(2)*U(2))/RA
*    -UA*(V(1)-V(2))+ AA*UA*CZ1*(ALS(1)-ALS(2)))*RDS
* + AGD*RDS*((V(1)-V(2))-AA*(R(1)-R(2))/RA-AA*CZ1*(ALS(1)-ALS(2)))
 FC = CNC1*(UC*UC+AC*VC) + CNC2*(AC*(R(4)*U(4)-R(5)*U(5))/RC
*    -UC*(V(4)-V(5))+AC*UC*CZ1*(ALS(4)-ALS(5)))*RDS
* + AGD*RDS*((V(4)-V(5))-AC*(R(4)-R(5))/RC-AC*CZ1*(ALS(4)-ALS(5)))
 FE = CNE1*(UE*UE+AE*VE) + CNE2*(AE*(R(7)*U(7)-R(8)*U(8))/RE
*    -UE*(V(7)-V(8))+ AE*UE*CZ1*(ALS(7)-ALS(8)))*RDS
* + AGD*RDS*((V(7)-V(8))-AE*(R(7)-R(8))/RE-AE*CZ1*(ALS(7)-ALS(8)))
 VBETAT = GFUN(VA,VC,VE,ZA)
 ABETAT = GFUN(AA,AC,AE,ZA)
 SBETAT = GFUN(SA,SC,SE,ZA)
 FYBETA = GFUN(FA,FC,FE,ZA)
 RHSA = DT*FYBETA - VDELA + VBETAT - CZ2*ABETAT
*      -CZ3*ABETAT*ALOG(SDELAT/SBETAT)
 ADELAT =  RHSA*CZ4
 TERM = CZ5*ADELAT**2
 RDELAT = (TERM/SDELAT)**CZ6
```

```
      PDELAT = TERM*RDELAT

      IF(BNDPRN.EQ.C.) GO TO 100
      WRITE(IW,6020) UDELAT,ADELAT,SDELAT,VBETAT,ABETAT,SBETAT
      WRITE(IW,6020) GA,GC,GE,FA,FC,FE
      WRITE(IW,6020) ZA,FYBETA
  100 CONTINUE

C COMPUTATIONS ON RAY B-D-F
      UB=U(13) $ RB=R(13) $ VB=V(13) $ SB=S(13) $ AB=A(13)
      UD=U(14) $ RD=R(14) $ VD=V(14) $ SD=S(14) $ AD=A(14)
      UF=U(15) $ RF=R(15) $ VF=V(15) $ SF=S(15) $ AF=A(15)
      PB =  P(13)
      CB = .5*(CUR(2,I) + CUR(3,I))
      VDELB = .5*(VDEL(2,I) + VDEL(3,I))
      YDELB = .5*(V(13)+VDELB)*DT
      UDELBT = UB*(1. - CB*VB*DT)
     *          -DTS*(.5*(U(2)**2 - U(3)**2)
     *                + .5*(S(2)+S(3))*CZ8*(R(2)**CT2 - R(3)**CT2) )
     *    + AGD*DTS*(U(2) - U(3))
      SDELBT = SB - DTS*UB*(S(2)-S(3))
     *    + AGD*DTS*(S(2) - S(3))
      GB = (YDELB         )*RDT -(VB - AB)
      GC = (YDELB -    DN )*RDT -(VD - AD)
      GF = (YDELB -    DN2)*RDT -(VF - AF)
      GBC = GB
      GB1 = -1.5*GB + 2.*GD - .5*GF
      GB2 = .5*(GB + GF) - GC
      CALL SMALRT(GBO,GB1,GB2,ZB)
      CNB2 = 1.                  $  CNB1 = CB
      CND2 = 1./(1.+DN *CB)      $  CND1 = CB*CND2
      CNF2 = 1./(1.+DN2*CB)      $  CNF1 = CB*CNF2
      FB = CNB1*(UB*UB+AB*VB) + CNB2*(AB*(R(2)*U(2)-R(3)*U(3))/RB
     *    -UB*(V(2)-V(3))+ AB*UB*CZ1*(ALS(2)-ALS(3)))*RDS
     * + AGD*RDS*((V(2)-V(3))-AB*(R(2)-R(3))/RB-AB*CZ1*(ALS(2)-ALS(3)))
      FD = CND1*(UD*UD+AD*VD) + CND2*(AD*(R(5)*U(5)-R(6)*U(6))/RD
     *    -UD*(V(5)-V(6))+ AD*UD*CZ1*(ALS(5)-ALS(6)))*RDS
     * + AGD*RDS*((V(5)-V(6))-AD*(R(5)-R(6))/RD-AD*CZ1*(ALS(5)-ALS(6)))
```

```
      FF = CNF1*(UF*UF+AF*VF) + CNF2*(AF*(R(8)*U(8)-R(9)*U(9))/RF
     *    -UF*(V(8)-V(9))+ AF*UF*CZ1*(ALS(8)-ALS(9)))*RDS
     * + AGD*RDS*((V(8)-V(9))-AF*(R(8)-R(9))/RF-AF*CZ1*(ALS(8)-ALS(9)))
      VBETBT = QFUN(VB,VC,VF,ZB)
      ABETBT = QFUN(AB,AC,AF,ZB)
      SBETBT = QFUN(SB,SC,SF,ZB)
      FYBETB = QFUN(FB,FD,FF,ZB)
      RHSB = DT*FYBETB - VDELB + VBETBT - CZ2*ABETBT
     *      -CZ3*ABETBT*ALOG(SDELBT/SBETBT)
      ADELBT = RHSB*CZ4
      TERM = CZ5*ADELBT**2
      RDELBT = (TERM/SDELBT)**CZ6
      PDELBT = TERM*RDELBT
      IF(BNDPRN.EQ.0.) GO TO 200
      WRITE(IW,6020) UDELBT,ADELBT,SDELBT,VBETBT,ABETBT,SBETBT
      WRITE(IW,6020) GB,GD,GF,FB,FD,FF
      WRITE(IW,6020) ZB,FYBETB
  200 CONTINUE


C COMPUTATION ON RAY 2-5-8
      C2 = CUR(2,I)
      CB     = C2 + AGD*CT*CDERV
      VDEL2 = VDEL(2,I)
      YDEL2 = .5*(V(2)+VDEL2)*DT
      RRR = 1./(1. + CB*YDEL2)
      UDEL2T = .5*(UDELAT + UDELBT)
      SDEL2T = .5*(SDELAT + SDELBT)
      RDEL2T = .5*(RDELAT + RDELBT)
      ADEL2T = .5*(ADELAT + ADELBT)
      TERM1 = U(2) - .5*DT*(C2*U(2)*V(2) + CB*UDEL2T*VDEL2*RRR)
      TERM2 = .5*(UDELAT**2 - UDELBT**2)
      TERM2 = TERM2*RRR
      TERM3 = .5*(SDELAT+SDELBT)*CZ8*(RDELAT**CT2 - RDELBT**CT2)
      TERM3 = TERM3*RRR
      TERM4 = .25*(U(1)**2 - U(3)**2)
      TERM5 = .5*S(2)*CZ8*(R(1)**CT2 - R(3)**CT2)
      UDEL2 =  TERM1 - .5*DTS*(TERM2 + TERM3 + TERM4 + TERM5 )
```

```
*     + .25*AGD*DTS*(U(1)-U(3)) + AGDH*DTS*(UDELAT-UDELBT)*.5
 SDEL2 = S(2) - .5*DTS*(U(2)*(S(1)-S(3))*.5 +
*         UDEL2T*(SDELAT - SDELBT)*RRR)
*     + .25*AGD*DTS*(S(1)-S(3)) + AGDH*DTS*(SDELAT-SDELBT)*.5
 TERM  = VDEL2 - ADEL2T
 G2 = (YDEL2          )*RDT - .5*(V(2) - A(2) + TERM )
 G5 = (YDEL2 -    DN )*RDT - .5*(V(5) - A(5) + TERM )
 G8 = (YDEL2 -    DN2)*RDT - .5*(V(8) - A(8) + TERM )
 C20  = G2
 C21  = -1.5*G2 + 2.*G5 - .5*G8
 C22  = .5*(G2 + G8) - G5
 CN22 = 1.                    $ CN21 = C2
 CALL SMALRT(C20,C21,C22,ZZ)
 CN52 = 1./(1.+DN * C2)    $ CN51 = C2*CN52
 CN82 = 1./(1. + 2.*DN*C2)     $   CN81 = C2*CN82
 F2 = CN21*(U(2)*U(2)+A(2)*V(2))+ CN22*(A(2)*(RA*UA-RB*UB)/R(2)
*     -U(2)*(VA-VB)+ A(2)*U(2)*CZ1*(ALS(10)-ALS(13)))*RDS
* + AGD*RDS*((VA-VB)-A(2)*(RA-RB)/R(2)-A(2)*CZ1*(ALS(10)-ALS(13)))
 F5 = CN51*(U(5)*U(5)+A(5)*V(5))+ CN52*(A(5)*(RC*UC-RD*UD)/R(5)
*     -U(5)*(VC-VD)+ A(5)*U(5)*CZ1*(ALS(11)-ALS(14)))*RDS
* + AGD*RDS*((VC-VD)-A(5)*(RC-RD)/R(5)-A(5)*CZ1*(ALS(11)-ALS(14)))
 F8 = CN81*(U(8)*U(8)+A(8)*V(8))+ CN82*(A(8)*(RE*UE-RF*UF)/R(8)
*     -U(8)*(VE-VF)+ A(8)*U(8)*CZ1*(ALS(12)-ALS(15)))*RDS
* + AGD*RDS*((VE-VF)-A(8)*(RE-RF)/R(8)-A(8)*CZ1*(ALS(12)-ALS(15)))
 VBET2  = QFUN(V(2),V(5),V(8),ZZ)
 ABET2  = QFUN(A(2),A(5),A(8),ZZ)
 SBET2  = QFUN(S(2),S(5),S(8),ZZ)
 FYBET2 = QFUN(F2  ,F5  ,F8  ,ZZ)
 RHS2 = FYBET2 + CB*RRR*(UDEL2T*UDEL2T + ADEL2T*VDEL2)
*       +RDS*RRR*( (ADEL2T/RDEL2T)*(RDELAT*UDELAT - RDELBT*UDELBT)
*        + CZ1*ADEL2T*UDEL2T*ALOG(SDELAT/SDELBT))
 RHS2 = .5*DT*RHS2 - VDEL2 + VBET2 - CZ2*ABET2 + .5*CZ3*(ABET2 +
*         ADEL2T)*ALOG(SBET2/SDEL2)
* -.5*AGDH*DT*ADEL2T*RDS*((RDELAT-RDELBT)/RDEL2T
* + CZ1*(SDELAT - SDELBT)/SDEL2T )
 ADEL2 = RHS2*CZ4
 TERM  = CZ5*ADEL2**2
 RDEL2 = (TERM/SDEL2)**CZ6
```

```
      PDEL2 =  TERM*RDEL2

      IF(BNDPRN.EQ.0.) GO TO 300
      WRITE(IW,6020)UDEL2,VDEL2,SDEL2,VBET2,ABET2,SBET2
      WRITE(IW,6020) G2,G5,G8,F2,F5,F8
      WRITE(IW,6020) ZZ,FYBET2
  300 CONTINUE

      WC(1,I) = RDEL2
      UX = VDEL2*COSN(2,I) - UDEL2*SINN(2,I)
      UY = VDEL2*SINN(2,I) + UDEL2*COSN(2,I)
      WC(2,I) = RDEL2*UX
      WC(3,I) = RDEL2*UY
      USQ     = UDEL2*UDEL2 + VDEL2*VDEL2
      WC(4,I) = PDEL2 - GT3*RDEL2*USQ

  900 CONTINUE

      IF(BNDISC.NE.1.) GO TO 4410
      DO 4400 I=1,NP
      DO 4200 L=2,M
      IF(I.EQ.NPTA(L)) GO TO 4300
 4200 CONTINUE
      GO TO 4400
 4300 RR=1./(SS(I+1)-SS(I-1))
      RAT1=(SS(I)-SS(I-1))*RR
      RAT2=(SS(I+1)-SS(I)) *RR
      DO 4350 K=1,4
 4350 WC(K,I) = RAT2*WC(K,I-1) + RAT1*WC(K,I+1)
 4400 CONTINUE
 4410 CONTINUE

 6010 FORMAT(1HO,*R,U,V,P,A,S,ALS */(1X,7F18.6))
 6020 FORMAT(1HO,7F18.6)
      RETURN
      END
```

```
      SUBROUTINE BNDWRC ( W1, W2, W3, W4,
     *                    PRES, TEMP, VSURF, AMLOCL, CP,
     *          COSN,SINN,WC,WP4, NP )
C PRINTS BOUNDARY VALUES
      COMMON / CM3A   / CT1,CT2,CT3,CT4,CT5,CT6,CT7,CT8,CT9
      COMMON / CM4    / CF,DTMX, DXYSN, DTFRST,DTOUT,DTBND,DTSTB,DT
      COMMON / CM11   / IR,IW
      COMMON / CM11B  / ID1,ID2,IL1,IL2,   ID1FP,ID2FP,IL1FP,IL2FP
      DIMENSION W1(NP), W2(NP), W3(NP), W4(NP),
     *          PRES(NP), TEMP(NP), VSURF(NP), AMLOCL(NP), CP(NP),
     *          COSN(3,NP), SINN(3,NP),  WC(4,NP)
      DIMENSION WP4(NP)
      RDT = 1./DT
      CMAX=0.
      CALL POSITN(ID2FP,2,ID2,353)
      READ(ID2)
      READ(ID2) WP4
   10 CONTINUE
      DO 100 I=1,NP
      RO = W1(I) = WC(1,I)
      RU = W2(I) = WC(2,I)
      RV = W3(I) = WC(3,I)
      E  = W4(I) = WC(4,I)
      WP4(I) = RDT*(W4(I) - WP4(I))
      RR = 1./RO
      U = RR*RU
      V = RR*RV
      VEL2 = U*U + V*V
      VEL = SQRT(VEL2)
      PRES(I) = E + CT3*RO*VEL2
      VSURF(I) = V*COSN(2,I) - U*SINN(2,I)
      CP(I) = CT8*(PRES(I) - 1.)
      PRES(I) = 1. + CT9*(PRES(I)-1.)
      ROUN    = 1. + CT9*(W1(I)  -1.)
      TEMP(I) = PRES(I)/ROUN
      TTMMP = ABS(TEMP(I))
      AMLOCL(I) = CT7*VEL/SQRT(TTMMP)
```

```
      IND=0
      IF(W1(I).LT.0.)     IND=1
      IF(PRES(I).LT.0.)   IND=1
      IF(TEMP(I).LT.0.)   IND=1
      IF(IND.EQ.0) GO TO 40
      RO = W1(I) = WC(1,I) = .5*(WC(1,I-1)+WC(1,I+1))
      RU = W2(I) = WC(2,I) = .5*(WC(2,I-1)+WC(2,I+1))
      RV = W3(I) = WC(3,I) = .5*(WC(3,I-1)+WC(3,I+1))
      E  = W4(I) = WC(4,I) = .5*(WC(4,I-1)+WC(4,I+1))
      GO TO 100
   40 CONTINUE
      TEMPP = E/RO + CT3*(RU*RU+RV*RV)/(RO*RO)
      IF(TEMPP.LT.0.) GO TO 100
      DNOW = CT4*SQRT(TEMPP) + VEL
      DMAX = AMAX1(DMAX,DNOW)

  100 CONTINUE
      DTBND = DXYSN/DMAX
      WRITE(IW,610)
      WRITE(IW,631) ( (I,W1(I),W2(I),W3(I),W4(I),PRES(I),
     * TEMP(I),VSURF(I),AMLOCL(I),CP(I),WP4(I),I),I=1,NP)
  610 FORMAT(1HO, 2X1HI , 9X3HWC1 , 9X3HWC2 , 9X3HWC3 , 9X3HWC4 ,
     *8X4HPRES , 8X4HTEMP , 7X5HVTANG , 7X5HMLOCL , 10X2HCP ,
     * 5X7HDW4BYDT , 2X1HI )
  631 FORMAT(1X,I3,10F12.6,I3)
   99 CONTINUE
      CALL POSITN(ID1FP,1,ID1,351)
      WRITE(ID1) WC
      CALL ENDFL(ID1FP,ID1)
      RETURN
      END
```

```
      SUBRCUTINE POLC(WQ,WR,WS,ZX,ZY,WH,WV,ZH,ZV,WANS)

C INTERPOLATES FOR W AT POINT IN A CELL WHEN ONE POINT OF CELL IS INSIDE CONTOUR

      DIMENSION WQ(4),WR(4),WS(4),WA(4),WB(4),WC(4),WD(4),WANS(4)
      DIMENSION WH(4),WV(4)
      C1 = 1. - ZY
      C2 = 1. - ZX
      C3 =1./(1. - ZV)
      C4 =1./(1. - ZH)
      C6 =C1*C3
      C7 =(ZY-ZV)*C3
      C8 =C2*C4
      C9 =(ZX-ZH)*C4
      DO 10 K=1,4
      WA(K) = ZY*WR(K) + C1*WQ(K)
      WB(K) = ZX*WR(K) + C2*WS(K)
      WC(K) = C6*WV(K) + C7*WS(K)
      WD(K) = C8*WH(K) + C9*WQ(K)
   10 WANS(K) = .5*(C1*WD(K) + ZY*WB(K) + C2*WC(K) + ZX*WA(K))

      RETURN
      END
```

```
      SUBROUTINE STATE( W,CS,SN,R,U,V,P,A,S,ALS)
C COMPUTES R,U,V,P,AS,ALS FROM CS,SN

      COMMON / CM3A   / CT1,CT2,CT3,CT4,CT5,CT6,CT7,CT8
      DIMENSION W(4)

      R  = W(1)
      RR = 1./R
      UX = RR*W(2)
      UY = RR*W(3)
      U  = -UX*SN + UY*CS
      V  =  UX*CS + UY*SN
      U2 = UX*UX + UY*UY
      P  = W(4) + CT3*U2*R
      T  = P/R
      A  = CT4*SQRT(T)
      S  = P/(R**CT5)
      ALS= ALOG(S)
      RETURN
      END
```

```
      SUBRCUTINE SMALRT( C,BB,A,X )
C FINDS SMALLEST POSITIVE RCOT OF A*Z*Z + B*Z + C = 0
      COMMCN / CM11   / IR,Iw
      B=.5*BB
      DEL = B*B - A*C
      IF(DEL.LT.0.) GC TC 10
      DD  = -B - SIGN(1.,B)*SQRT(DEL)
      IF(ABS(A).LT.1.E-1C)A=1.E-10
      X1=DD/A  $  X2=C/DD
      X = AMIN1(X1,X2)
      IF(X.GE.C.)GC TC 9C
      X = AMAX1(X1,X2)
      IF(X.LT.C.) GC TC 20
      GC TC 9C
   10 WRITE(Iw,11) A,BB,C,DEL
   11 FCRMAT(1HC,* QUADRATIC FRCM BND3 HAS IMAGINARY RCCTS*/
     *       1X ,* A,BB,C,DEL ARE * 4E20.8 )
      CALL EXIT
   20 WRITE(Iw,21) A,BB,C,X1,X2
   21 FCRMAT(1HC,* QUA4RATI3 6964 BND3 HAS BCTH RCCTS NEGATIVE*/
     *       1X ,* A,BB,C,X1,X2 ARE * 5E20.8 )
      CALL EXIT
   90 RETURN
      END
```

```
      OVERLAY(OVER,4,0)
      PROGRAM FINDCT

      COMMON / CM4A    / ICT,ICTFP,T1,T2,NC1,NCURV
      COMMON / CM4B    / TPREV,T,INDCT,NT,NTFRST
      COMMON / CM8A    / NCTMX,NBTMX,NX,NY,KX,KY,KW,NBT,MM
      COMMON / CM10    / ICCTL,TMX,NTMX,TMR
      COMMON / CM10A   / TIMA(2),TIMB(2),TIMS(2),TIMT(2),TIM(2,8)
      COMMON / CM11    / IR,IW
      COMMON / CM999   / NN,A(1)

      N = NCTMX
      N1 = 1
      N2 = N1 + N
      N3 = N2 + N
      N4 = N3 + N
      N5 = N4 + N
      N6 = N5 + N
      N7 = N6 + N
      N8 = N7 + N
      N9 = N8 + N
      N10 = N9 + N
      N11 = N10 + N
      N12 = N11 + N
      N13 = N12 + N
      N14 = N13 + N
      N15 = N14 + N
      N16 = N15 + N
      N17 = N16 + N
      N18 = N17 + N
      N19 = N18 + N
      N20 = N19 + N
      N21 = N20 + N
      N22 = N21 + N
      N23 = N22 + N
      N24 = N23 + N
      N25 = N24 + N
      N26 = N25 + N
```

```
      N27 = N26 + N

      IF ( INDCT.EQ.2 ) GO TO 200

      N28 = N27 + N
      N29 = N28 + N
      N30 = N29 + N

      CALL NEWCT1 ( A(N1), A(N2), A(N3), A(N4), A(N5),A(N6),A(N7),A(N8),
     *              A(N9), A(N10), A(N11), A(N12),
     *              A(N13), A(N14), A(N15), A(N16),
     *              A(N17), A(N18), A(N19), A(N20), A(N21), A(N22),
     *        A(N23),A(N24),A(N25),A(N26),A(N27),A(N28),A(N29),A(N30),
     *              NC1, NCURV )
      GO TO 999

  200 CONTINUE
      NP = NC1
      N27 = N26 + 4*N
      N27A = N27 + NP - 1
      N28 = N27 + N
      READ(ICT) (A(I),I=N27,N27A)
      CALL POSITN ( ICTFP,ICTFP+1,ICT,401 )
      READ(ICT) ICTFP,T1,T2,NC1,NCURV
      CALL NEWCT2 ( A(N1),A(N2),A(N3),A(N4),A(N5),A(N6),A(N7),A(N8),
     *              A(N9), A(N10), A(N11), A(N12),
     *              A(N13), A(N14), A(N15), A(N16),
     *              A(N17), A(N18), A(N19), A(N20), A(N21), A(N22),
     *              A(N23), A(N24), A(N25), A(N26), A(N27), A(N28),
     *              NC1,NCURV,NP )
  999 CONTINUE

      CALL TIMER(TMR,TIMA,TIMB,TIM(1,4))
      WRITE(IW,6090) TIM(1,4),TIM(2,4),T,NT
 6090 FORMAT(1HC,* CP,PP TIMES FOR  $ FINDCT $  ARE (SEC) *,2F10.2,5X
     $*T = * F10.6, 4X * NT = * I6 )

      END
```

```
      SUBROUTINE NEWCT1(X1,Y1,X2,Y2,X1D,Y1D,XDD,YDD,
     *                  NPTA,NA,A1,A2,
     *                  NPTB,NB,B1,B2,
     *                  X,Y,D,CS,SN,C1,C2,C3,S,XD,YD,SCR1,SCR2,WC,
     *                  N,M )

C FINDS THE NEW-CONTOUR WHEN IT IS NOT IDENTICAL WITH A READ IN CONTOUR

      COMMON / CM3A   / CT1,CT2,CT3,CT4,CT5,CT6,CT7,CT8
      COMMON / CM3B   / ALFAR,W1FS,W2FS,W3FS,W4FS
      COMMON / CM4    / CF,DTMX, DXYSN, DTFRST,DTOUT,DTBND,DTSTB,DT
      COMMON / CM4A   / ICT,ICTFP,T1,T2,NC1,NCURV
      COMMON / CM4B   / TPREV,T,INDCT,NT,NTFRST
      COMMON / CM7    / TOLS,TOLX,TOLCB,TOLCF,TOLSOL
      COMMON / CM11   / IR,IW
      COMMON / CM11B  / IC1,IC2,IL1,IL2,   IC1FP,IC2FP,IL1FP,IL2FP
      COMMON / CM13   / ISTART,ISTOP,ISAVE,NSAVE

      DIMENSION X1(N),Y1(N),X2(N),Y2(N),X1D(N),Y1D(N),XDD(N),YDD(N),
     *          NPTA(M),NA(M),A1(M),A2(M),
     *          NPTB(M),NB(M),B1(M),B2(M),
     *          X(N),Y(N),D(N),CS(N),SN(N),C1(N),C2(N),C3(N),S(N),
     *          XD(N),YD(N),SCR1(N),SCR2(N),WC(4,N)

      BACKSPACE ICT
      READ(ICT) X1,Y1,X2,Y2,X1D,Y1D,XDD,YDD,
     *          NPTA,NA,A1,A2,
     *          NPTB,NB,B1,B2
      X1(1)=X1(N)=X1D(1)=X1D(N)=XDD(1)=XDD(N)=0.

      DEL =  T - T1
      DEL2 = .5*DEL*DEL

      DO 10 I=1,N
      X(I) = X1(I) + X1D(I)*DEL + XDD(I)*DEL2
      Y(I) = Y1(I) + Y1D(I)*DEL + YDD(I)*DEL2
      XD(I) = X1D(I) + XDD(I)*DEL
```

```
      YC(I) = YIC(I) + YCC(I)*CEL
   10 CONTINUE

      R1 = (T-T1)/(T2-T1)
      R2 = (T2-T)/(T2-T1)

      DO 20 K=1,M
      KA = NPTA(K)
      KB = NPTB(K)
      NN = KB - KA + 1

C COMPUTE END CONDITIONS BY LINEAR INTERPOLATION BETWEEN VALUES AT CONTOURS 1,2

      VALA  = R2*A1(K) + R1*A2(K)
      VALB = R2*B1(K) + R1*B2(K)
      CALL NLCSP( X(KA),Y(KA),NN,NA(K),VALA,NB(K),VALB,TOLCB,
     *            D(KA),CS(KA),SN(KA),C1(KA),C2(KA),C3(KA),
     *            SCR1(KA),SCR2(KA) )
   20 CONTINUE
      D(N)=CS(N)=SN(N)=C1(N)=C2(N)=C3(N)=0.

C FILL IN ARRAY *S*

      N1 = N - 1
      S(1) = 0.
      DO 30 I=2,N
      I1 = I - 1
      CALL CUBARC ( 0.,D(I1),C1(I1),C2(I1),C3(I1),TOLS*D(I1),S(I))
   30 S(I) = S(I) + S(I1)

C WRITE CONTOUR INFORMATION ON IO(2),IO2(3)

      CALL POSITN( IO2FP,2,IO2,411)
      WRITE(IO2) X,Y,D,CS,SN,C1,C2,C3,S
      WRITE(IO2) S,NPTA
      CALL ENDFL(IO2FP,IO2)
      WRITE(IO2) XO,YO,XCC,YCC
      CALL ENDFL(IO2FP,IO2)
```

```
      IF(NT.EQ.0.AND.ISTART.EQ.1)GO TO 900
      CALL POSITN ( ID1FP,1,ID1,422 )
      READ(ID1) WC
  100 CONTINUE
      CALL POSITN ( ID2FP,4,ID2,423 )
      WRITE(ID2) WC
      CALL ENDFL(ID2FP,ID2)
      WRITE(IW,6000)
      WRITE(IW,6010) (I,X(I),Y(I),C1(I),C2(I),C3(I),S(I),I=1,N)
      RETURN

C SET *WC* WHEN NT = 0
  900 CONTINUE
      PFS=W4FS+CT3*(W2FS**2+W3FS**2)/W1FS
      DO 950 I=1,N
      IF(I.NE.1) GO TO 910
      SNN=-1.  $  CSN=0.
      GO TO 930
  910 IF(I.EQ.N) GO TO 920
      I1=I-1
      CALL NRMCUR(D(I1),CS(I1),SN(I1),C1(I1),C2(I1),C3(I1),CS1,SN1,DUM)
      CALL NRMCUR(0.,CS(I),SN(I),C1(I),C2(I),C3(I),CS2,SN2,DUM)
      TH1=ATAN2(SN1,CS1)  $  TH2=ATAN2(SN2,CS2)
      SNN=SIN(0.5*(TH1+TH2))
      CSN=COS(0.5*(TH1+TH2))
      GO TO 930
  920 SNN=1.  $  CSN=0.
  930 ALD=CSN*XD(I)+SNN*YD(I)
      WC(1,I) = W1FS
      WC(2,I) = W1FS*ALD*CSN
      WC(3,I) = W1FS*ALD*SNN
      WC(4,I) = PFS-CT3*W1FS*ALD*ALD
  950 CONTINUE
      RR = 1./WC(1,I)
      U = RR*WC(2,I)
      V = RR*WC(3,I)
      VSQ = U*U + V*V
      VEL = SQRT(VSQ)
```

```
      TEMP = RR*WC(4,I) + CT3*VSC
      APLUSV = CT4*SQRT(TEMP) + VEL
      DTBND = DXYSN/APLUSV
      GO TO 100
 6000 FORMAT(1H0,* I,X,Y,C1,C2,C3,S ARE * )
 6010 FORMAT(1X,I5,3X,2F15.8,4X,3F15.8,4X,F15.8)
      END
```

```
      SUBROUTINE NEWCT2 ( X1,Y1,X2,Y2,X1D,Y1D,XDD,YDD,
     *                    NPTA,NA,A1,A2,
     *                    NPTB,NB,B1,B2,
     *                    D,CS,SN,C1,C2,C3,SCR1,SCR2,
     *                    S,WC,  SP,WCQ,  N,M,NP )

C FINDS NEW CONTOUR WHEN  *INDCT* = 2

      COMMON / CM4A   / ICT,ICTFP,T1,T2,NC1,NCURV
      COMMON / CM4B   / TPREV,T,INDCT,NT,NTFRST
      COMMON / CM7    / TOLS,TOLX,TOLCB,TOLCF,TOLSOL
      COMMON / CM11   / IR,IW
      COMMON / CM11B  / IC1,IC2,IL1,IL2,   IC1FP,IC2FP,IL1FP,IL2FP


      DIMENSION X1(N),Y1(N),X2(N),Y2(N),X1D(N),Y1D(N),XDD(N),YDD(N),
     *          NPTA(M),NA(M),A1(M),A2(M),
     *          NPTB(M),NB(M),B1(M),B2(M),
     *          D(N),CS(N),SN(N),C1(N),C2(N),C3(N),SCR1(N),SCR2(N),
     *          S(N),WC(4,N),   SP(NP),WCQ(4,NP)
      READ(ICT) X1,Y1,X2,Y2,X1D,Y1D,XDD,YDD,
     *          NPTA,NA,A1,A2,
     *          NPTB,NB,B1,B2
      X1(1)=X1(N)=X2(1)=X2(N)=X1D(1)=X1D(N)=XDD(1)=XDD(N)=0.

C COMPUTE NON-LINEAR CUBIC SPLINES

      DO 20 K=1,M
      KA = NPTA(K)
      KB = NPTB(K)
      NN = KB - KA + 1
      CALL NLCSP ( X1(KA),Y1(KA),NN,NA(K),A1(K),NB(K),B1(K),TOLCB,
     *             D(KA),CS(KA),SN(KA),C1(KA),C2(KA),C3(KA),
     *             SCR1(KA),SCR2(KA) )
   20 CONTINUE
      D(N)=CS(N)=SN(N)=C1(N)=C2(N)=C3(N)=0.

C FILL IN ARC LENGTH ARRAY *S*
```

```
      N1 = N - 1
      S(1) = 0.
      DO 30 I=2,N
      I1 = I - 1
      CALL CUBARC ( 0.,D(I1),C1(I1),C2(I1),C3(I1),TOLS*D(I1),S(I))
   30 S(I) = S(I) + S(I1)

C WRITE CONTOUR INFORMATION ON ID2(2),ID2(3)

      CALL POSITN ( ID2FP,2,ID2,421 )
      WRITE(ID2) X1,Y1,D,CS,SN,C1,C2,C3,S
      WRITE(ID2) S,NPTA
      CALL ENDFL(ID2FP,ID2)
      WRITE(ID2) X1D,Y1D,XDD,YDD
      CALL ENDFL(ID2FP,ID2)

C READ WCC  FROM ID1(1)

      CALL POSITN ( ID1FP,1,ID1,422 )
      READ(ID1) WCC

C COMPUTE *WC* FROM *WCC*

      DO 40 K=1,4
      WC(K,1) = WCC(K,1)
      WC(K,N) = WCC(K,NP)
   40 CONTINUE

      N1 = N - 1
      DO 100 I=2,N1

      DO 60 J=2,NP
      IF(S(I).LE.SP(J)) GO TO 80
   60 CONTINUE

   80 J1 = J - 1
      RRDD = 1./(SP(J) - SP(J1))
```

```
      S1 = (S(I) - SP(J1))*RRCC
      S2 = (SP(J) - S(I))*RRCC
      DO 90 K=1,4
   90 WC(K,I) = S2*WCC(K,J1) + S1*WCC(K,J)
  100 CONTINUE

C SAVE *WC* ON ID2(4)

      CALL POSITN ( ID2FP,4,ID2,423 )
      WRITE(ID2) WC
      CALL ENDFL(ID2FP,ID2)

      WRITE(IW,6000)
      WRITE(IW,6010) (I,X1(I),Y1(I),C1(I),C2(I),C3(I),S(I),I=1,N)
 6000 FORMAT(1HO,* I,X,Y,C1,C2,C3,S ARE * )
 6010 FORMAT(1X,I5,3X,2F15.8,4X,3F15.8,4X,F15.8)

      RETURN
      END
```

```
      OVERLAY(OVER,5,0)
      PROGRAM MAPS

      COMMON / CM4A    / ICT,ICTFP,T1,T2,NC1,NCURV
      COMMON / CM4B    / TPREV,T,INDCT,NT,NTFRST
      COMMON / CM8A    / NCTMX,NBTMX,NX,NY,KX,KY,KW,NBT,MM
      COMMON / CM1C    / ICCTL,TMX,NTMX,TMR
      COMMON / CM10A   / TIMA(2),TIMB(2),TIMS(2),TIMT(2),TIM(2,8)
      COMMON / CM11    / IR,IW
      COMMON / CM999   / NN,A(1)

      NC = NC1
      N1 = 1
      N2 = N1 + KX*NX
      N3 = N2 + KY*NY
      N4 = N3 + NC
      N5 = N4 + NC
      N6 = N5 + NC
      N7 = N6 + NC
      N8 = N7 + NC
      N9 = N8 + NC
      N10 = N9 + NC
      N11 = N10 + NC
      N12 = MAXO( KX*NX + KY*NY + 9*NC , 2*KW*NX)
      N12 = N12 + 1
      N13 = N12 + NX
      N14 = N13 + KX*NX
      N15 = N14 + NY

      CALL INTCON (  A(N1), A(N2),
     *               A(N3),A(N4),A(N5),A(N6),A(N7),A(N8),A(N9),A(N10),
     *               A(N11),
     *               A(N12), A(N13), A(N14), A(N15),
     *               KX, NX, KY, NY,  NC )
      N2 = N1 + KW* NX

      CALL MAPIT ( A(N1), A(N2),   A(N12),A(N13),A(N14),A(N15),
     *             A(N1),
```

```
     *              A(N1), A(N2),
     *                     A(N2),
     *               KX,NX,KY,NY,KW,NBTMX,NBT )

      CALL TIMER(TMR,TIMB,TIMA,TIM(1,5))
      WRITE(IW,6090) TIM(1,5),TIM(2,5),T,NT
 6090 FORMAT(1HO,* CP,PP TIMES FOR  $ MAPS    $  ARE (SEC) *,2F10.2,5X
     $*T = * F10.6, 4X * NT = * I6 )

      END
```

```
      SUBROUTINE INTCON ( NSX,NSY,
     *                    X,Y,D,CS,SN,C1,C2,C3,S,
     *                    IX,VX,IY,VY,
     *                    KX,NX,KY,NY,  NC )

      COMMON / CM2     / ZNX,ZNY,YZ,DX,DY,DS,DN,   RDX,RDY,RDS,RDN
      COMMON / CM7     / TOLS,TOLX,TOLCB,TOLCF,TOLSOL
      COMMON / CM11    / IR,IW
      COMMON / CM11B   / ID1,ID2,IL1,IL2,   ID1FP,ID2FP,IL1FP,IL2FP
      COMMON / CM14    / NPRNT,IPRNT,NBUG,IBUG

      DIMENSION NSX(KX,NX), NSY(KY,NY),
     *          X(NC),Y(NC),D(NC),CS(NC),SN(NC),C1(NC),C2(NC),C3(NC),
     *          S(NC),
     *          IX(NX),VX(KX,NX),IY(NY),VY(KY,NY)
      DIMENSION X3(3),Y3(3),NS(3)
      DIMENSION VXY(20),NXY(20)

C READ *CONT* FROM  ID2(2)

      CALL POSITN ( ID2FP,2,ID2,511 )
      READ(ID2) X,Y,D,CS,SN,C1,C2,C3,S
      N1 = NC - 1

C ZERO OUT ARRAYS IX,VX,NSX

      DO 5 I=1,NX
      IX(I)=0
      DO 5 J=1,KX
      VX(J,I) = 0.
      NSX(J,I)= 0
    5 CONTINUE

C ZERO OUT ARRAYS IY,VY,NSY

      DO 7 I=1,NY
      IY(I)=0
```

```
      DO 7 J=1,KY
      VY(J,I) = 0.
      NSY(J,I)= 0
    7 CONTINUE

C IN FOLLOWING DO LOOP INTERSECTIONS OF CONT WITH VERT MESH LINES ARE FOUND

      DO 100 I=1,N1
      XC = X(I)
      XD = X(I+1)
      X1 = AMIN1(XC,XD)
      X2 = AMAX1(XC,XD)
      K1 = INT(X1*RDX) + 1
      K2 = INT(X2*RDX) + 1
      IF(K1.LT.1) K1=1
      IF(K2.GT.NX) K2=NX
      KM = (K1+K2)/2
      SS = S(I+1) - S(I)
      K = KM
      IND = 1
      GO TO 50
    8 IF( NR.EQ.0 .AND. IND.EQ.1 ) GO TO 20
      IF( NR.EQ.0 .AND. IND.EQ.2 ) GO TO 100
      IF( NR.NE.0 .AND. IND.EQ.1 ) GO TO 10
      IF( NR.NE.0 .AND. IND.EQ.2 ) GO TO 30
   10 K = K-1
      IF(K.GE.1)GO TO 50
      NR=0
      GO TO 8
   20 K = KM+1
      IF(K.GT.NX)GO TO 100
      IND = 2
      GO TO 50
   30 K = K+1
      IF(K.GT.NX)GO TO 100
C FIND INTERSECTION OF CUBIC SEGMENT *I* WITH X=(K-1)*DX

C FIND INTERSECTION OF CUBIC SEGMENT *I* WITH LINE X=(K-1)*DX , K FROM K1,K2
```

```
 50 XK = FLOAT(K-1)*DX
    CALL INTCUB(I,X(I),Y(I),D(I),CS(I),SN(I),C1(I),C2(I),C3(I),
   *            SS,1.,0.,-XK,TOLCF,TOLS,NR,X3,Y3,NS)
    IF(NR.EQ.0)GO TO 70
    II = IX(K)
    DO 60 N=1,NR
    II = II + 1
    VX(II,K) = Y3(N)
 60 NSX(II,K) = NS(N)
    IX(K) = IX(K) + NR
 70 CONTINUE
    GO TO 8
100 CONTINUE


    DO 200 I=1,N1
    YO = Y(I)
    YD = Y(I+1)
    Y1 = AMIN1(YO,YD) + YZ
    Y2 = AMAX1(YO,YD) + YZ
    K1 = INT(Y1*RDY) + 1
    K2 = INT(Y2*RDY) + 1
    IF(K1.LT.1) K1=1
    IF(K2.GT.NY) K2=NY
    KM = (K1+K2)/2
    SS=S(I+1)-S(I)
    K = KM
    IND = 1
    GO TO 150
108 IF( NR.EQ.0 .AND. IND.EQ.1 ) GO TO 120
    IF( NR.EQ.0 .AND. IND.EQ.2 ) GO TO 200
    IF( NR.NE.0 .AND. IND.EQ.1 ) GO TO 110
    IF( NR.NE.0 .AND. IND.EQ.2 ) GO TO 130
110 K = K-1
    IF(K.GE.1)GO TO 150
    NR=0
    GO TO 108
```

```
  120 K = KM+1
      IF(K.GT.NY) GC TC 200
      IND = 2
      GC TC 150
  130 K = K+1
      IF(K.GT.NY) GO TC 200

C     FIND INTERSECTICN CF CUBIC SEGMENT *I* WITH Y=(K-1)*DY-YZ

  150 YK = FLCAT(K-1)*DY - YZ
      CALL INTCUB(I,X(I),Y(I),D(I),CS(I),SN(I),C1(I),C2(I),C3(I),
     *            SS,0.,1.,-YK,TCLCF,TCLS,NR,X3,Y3,NS )
      IF(NR.EC.C) GC TC 170
      II = IY(K)
      DO 160 N=1,NR
      II = II + 1
      VY(II,K) = X3(N)
  160 NSY(II,K) = NS(N)
      IY(K) = IY(K) + NR
  170 CONTINUE
      GC TC 108
  200 CONTINUE

C REARRANGE EACH COLUMN CF VX,NSX IN ASCENDING ORDER OF CONTENTS OF VX

      DC 320 K=1,NX
      N = IX(K)
      IF(N.LE.1) GC TC 320
      DC 310 I=2,N
      DC 310 J=2,N
      IF(VX(J,K).GE.VX(J-1,K)) GC TO 310
      ELSAVE     = VX(J,K)
      VX(J,K)    = VX(J-1,K)
      VX(J-1,K) = ELSAVE
      NLSAVE      = NSX(J,K)
      NSX(J,K)    = NSX(J-1,K)
      NSX(J-1,K) = NLSAVE
  310 CONTINUE
```

```
  320 CONTINUE

C REARRANGE EACH COLUMN OF VY,NSY IN ASCENDING ORDER OF CONTENTS OF VY

      DO 420 K=1,NY
      N = IY(K)
      IF(N.LE.1) GO TO 420
      DO 410 I=2,N
      DO 410 J=2,N
      IF(VY(J,K).GE.VY(J-1,K)) GO TO 410
      ELSAVE     = VY(J,K)
      VY(J,K) = VY(J-1,K)
      VY(J-1,K) = ELSAVE
      NLSAVE      = NSY(J,K)
      NSY(J,K)    = NSY(J-1,K)
      NSY(J-1,K) = NLSAVE
  410 CONTINUE
  420 CONTINUE

      TOL=.001*(DX+DY)/2.

C WIPE OUT COINCIDENT POINTS IN *VX*
      DO 520 K=1,NX
      N=IX(K)
      IF(N.LE.1)GO TO 520
      VXY(1)=VX(1,K)
      NXY(1)=NSX(1,K)
      NN=1
      DO 510 J=2,N
      IF(N.EQ.2) GO TO 505
      IF(ABS(VX(J,K)-VX(J-1,K)).LE.TOL )GO TO 510
  505 CONTINUE
      NN=NN+1
      VXY(NN) = VX(J,K)
      NXY(NN) = NSX(J,K)
  510 CONTINUE
      IX(K) = NN
      DO 515 J=1,NN
```

```
      VX(J,K) = VXY(J)
      NSX(J,K)=NXY(J)
  515 CONTINUE
  520 CONTINUE

C WIPE OUT COINCIDENT POINTS IN *VY*

      DO 620 K=1,NY
      N=IY(K)
      IF(N.LE.1) GO TO 620
      VXY(1) = VY(1,K)
      NXY(1) = NSY(1,K)
      NN=1
      DO 610 J=2,N
      IF(ABS(VY(J,K)-VY(J-1,K)).LE.TOL ) GO TO 610
      NN = NN + 1
      VXY(NN) = VY(J,K)
      NXY(NN) = NSY(J,K)
  610 CONTINUE
      IY(K) = NN
      DO 615 J=1,NN
      VY(J,K) = VXY(J)
      NSY(J,K)=NXY(J)
  615 CONTINUE
  620 CONTINUE

C WRITE * IX,VX,IY,VY,NSX,NSY * ON ID2(5)

      CALL POSITN ( ID2FP,5,ID2,512 )
      WRITE(ID2) IX,VX,IY,VY,  NSX,NSY
      CALL ENDFL(ID2FP,ID2)

      IF(IBUG.EQ.0) GO TO 710
      WRITE(IW,6000)
      WRITE(IW,6010) IX
      WRITE(IW,6020) VX
      WRITE(IW,6030) NSX
      WRITE(IW,6040) IY
```

```
      WRITE(Iw,6020) VY
      WRITE(Iw,6030) NSY
  710 CONTINUE

 6000 FORMAT(1H0,* IX,VX,NSX,IY,VY,NSY*)
 6010 FORMAT(1X,5012)
 6020 FORMAT(1X,10F12.8)
 6030 FORMAT(1X,5120)
      RETURN
      END
```

```
      SUBROUTINE INTCUB(II,XC,YC,D,CS,SN,C1,C2,C3,SS,A,B,C,TOLCF,TOLS,
     *                  NR,XR,YR,NS)

      COMMON / CM11   / IR,IW
      DIMENSION XR(3),YR(3),XL(3),CF(4),RRL(3),RIM(3),NS(3)

C FIND COEFS OF LINE IN LOCAL COORDS

      AL =  A*CS + B*SN
      BL = -A*SN + B*CS
      CL =  A*XC + B*YC + C

C FIND COEFS OF CUBIC

      CF(1) = CL
      CF(2) = BL*C1 + AL
      CF(3) = BL*C2
      CF(4) = BL*C3
      CFMX=AMAX1(ABS(CF(1)),ABS(CF(2)),ABS(CF(3)),ABS(CF(4)))
      EPS=TOLCF*CFMX

C IF ANY COEF IS VERY SMALL COMPARED WITH OTHERS, SET TO ZERO

      NR=0
      DO 10 MC=1,4
   10 IF(ABS(CF(MC)).LT.EPS) CF(MC)=0.
      M = 3
      IF(CF(4).NE.0.)GO TO 20
      M = 2
      IF(CF(3).NE.0.) GO TO 20
      M = 1
      IF(CF(2).NE.0.) GO TO 20
      M = 0
      RETURN

   20 CALL ROOT3(M,CF,RRL,RIM,CONV)

      C1 = -.001*D
```

```
      D2 = 1.0D1*D
      E1 = .0DD1*D
      E2 = .9999*D
      DO 3D I=1,M
      IF(RIM(I).NE.D.) GO TO 3D
      RI = RRL(I)
      IF( RI.LT.D1 .OR. RI.GT.D2 ) GO TO 3D
      NR = NR + 1
      IF( RI.GE.D ) RI = D
      IF( RI.LE.0. )RI = 0.
      XL(NR) = RI
      CALL CUBARC (0.,XL(NR),C1,C2,C3,TOLS*SS,SI )
      SI = SI/SS
      IF(SI.GT.1.) SI=1.
      NS(NR) = 10000*INT(1.E+08*SI) + II
   3D CONTINUE

      DO 4D I=1,NR
      XX = XL(I)
      YY = XX*(C1 + XX*(C2 + XX*C3))
      XR(I) = XC + XX*CS - YY*SN
   4D YR(I) = YO + XX*SN + YY*CS

      RETURN
      END
```

```
      SUBRCUTINE MAPIT( MAPBNW,  MAPCUT,   IX,VX,IY,VY,
     *                  MAPBCL,
     *                  MAPBET,  MAPLXW,
     *                           LISTB,
     *              KX,NX,KY,NY,KW,NBTMX,NBT )

      DIMENSICN MAPBNW(KW,NX), MAPCUT(KW,NX), MAPBCL(KW,NX),
     *          MAPBET(KW,NX), MAPLXW(KW,NX),
     *          IX(NX), VX(KX,NX), IY(NY), VY(KY,NY), LISTB(NBTMX)

      COMMCN / CM2    / ZNX,ZNY,YZ,DX,DY,DS,DN,   RDX,RDY,RDS,RDN
      COMMCN / CM4B   / TPREV,T,INDCT,NT,NTFRST
      COMMCN / CM11   / IR,IW
      COMMCN / CM11B  / ID1,ID2,IL1,IL2,   ID1FP,ID2FP,IL1FP,IL2FP
      COMMCN / CM12   / IALZRS,IALWNS,MSK(60),MSKRGT(60)
      COMMCN / CM14   / NPRNT,IPRNT,NBUG,IBUG

C GENERATES *MAPCUT* FCR PRESENT CONTOUR *T*
C GENERATES *MAPBNW* FCR USE AT *T+DT*
C SAVES     *MAPBNW* CN ID2(6)
C READS IN  *MAPBCL* FROM ID2(1)
C           *MAPBET* = (MAPBCL).ANC.(MAPCUT)
C           *MAPLXW* = (.NCT.MAPBET).ANC.(MAPCLT)
C SAVES     *MAPLXW* ON ID1(1)
C FORMS     *LISTB * FRCM  *MAPBET*

C STATEMENT FUNCTICN
      XMESH(I) = FLCAT(I-1)*DX
      YMESH(J) = FLCAT(J-1)*DY - YZ

C SET *MAPCUT* = 1 , *MAPBNW * = 0

      DC 1C I=1,NX
      DC 1C J=1,KW
      MAPCLT(J,I) = IALWNS
      MAPBNW(J,I) = IALZRS
   10 CONTINUE
```

```
C GENERATE *MAPCUT*

      DO 100 I=1,NX
      JINT = IX(I)
      IF(JINT.EQ.0) GO TO 100
      DO 90 J=1,JINT
      YY = VX(J,I)
      N = INT((YY+YZ)*RDY) + 1
      YYY = YMESH(N)
      N = N + 1
      IF(YYY.EQ.YY) N = N - MCD(J,2)
      N1  = N - 1
      IBIT = 60 - MOD(N1,60)
      IWD = 1 + N1/60
      IL=MAPCUT(IWD,I)
      IM = MSKRGT(IBIT)
      LOM = IL.OR.IM
      LAM = IL.AND.IM
      NLAM = .NOT.LAM
      MAPCUT(IWD,I) = LOM.AND.NLAM
      IF(IWD.EQ.KW) GO TO 90
      IWD1 = IWD + 1
      DO 20 K=IWD1,KW
   20 MAPCUT(K,I) = .NOT.MAPCUT(K,I)
   90 CONTINUE
  100 CONTINUE

      IF(IBUG.EQ.0) GO TO 105
      WRITE(IW,6000)
      DO 101 I=1,NX
  101 WRITE(IW,6010)(MAPCUT(J,I),J=1,KW)
  105 CONTINUE

C GENERATE *MAPBNW*

      NY1 = NY - 1
      NX1 = NX - 1
      DO 200 I=1,NX
```

```
      JINT = IX(I)
      IF ( JINT.EQ.0 ) GO TO 200
      I1 = I - 1
      IF(I1.LT.1) I1 = 1
      I2 = I + 1

      DO 200 K=1,JINT
      YY = VX(K,I)
      J1 = INT((YY + YZ)*RDY) + 1
      J2 = J1 + 1
      DO 120 J=J1,J2
      JJ = J - 1
      IBIT = 60 - MOD(JJ,60)
      IWD  = 1 + JJ/60

      DO 120 II=I1,I2
      MAPBNW(IWD,II) = MAPBNW(IWD,II).OR.MSK(IBIT)
  120 CONTINUE

      IF ( YY.NE.YMESH(J1) ) GO TO 200
      JJ = (J1 - 1) - 1
      IBIT = 60 - MOD(JJ,60)
      IWD = 1 + JJ/60
      DO 130 II=I1,I2
      MAPBNW(IWD,II)= MAPBNW(IWD,II).OR.MSK(IBIT)
  130 CONTINUE
  200 CONTINUE


      DO 300 J=1,NY
      JINT = IY(J)
      IF ( JINT.EQ.0 ) GO TO 300
      J1 = J - 1
      J2 = J + 1
      DO 300 K=1,JINT
      XX = VY(K,J)
      I1 = INT(XX*RDX) + 1
      I2 = I1 + 1
```

```
      DO 220 JK = J1,J2
      JJ = JK - 1
      IBIT = 60 - MOD(JJ,60)
      IWD = 1 + JJ/60
      DO 220 I=I1,I2
      MAPBNW(IWD,I) = MAPBNW(IWD,I).OR.MSK(IBIT)
  220 CONTINUE

      IF(XX.NE.XMESH(I1)) GO TO 300
      IF(I1.LE.1) GO TO 300
      I1 = I1 - 1
      DO 230 JA=J1,J2
      JJ = JA - 1
      IBIT = 60 - MOD(JJ,60)
      IWD = 1 + JJ/60
      MAPBNW(IWD,I1)= MAPBNW(IWD,I1).OR.MSK(IBIT)
  230 CONTINUE
  300 CONTINUE

C STORE *MAPBNW* IN ID2(6)

      CALL POSITN ( ID2FP,6,ID2,521 )
      WRITE(ID2) MAPBNW
      CALL ENDFL(ID2FP,ID2)
      IF(NT.EQ.0)GO TO 500

C READ  *MAPBCL* FROM ID2(1)

      CALL POSITN ( ID2FP,1,ID2,522 )
      READ (ID2) MAPBCL

C GENERATE *MAPBET*

      DO 400 I=1,NX
      DO 400 J=1,KW
  400 MAPBET(J,I) = MAPBCL(J,I).AND.MAPCUT(J,I)
```

```
      IF(IBUG.EQ.0) GO TO 405
      WRITE(IW,6020)
      DO 401 I=1,NX
  401 WRITE(IW,6010)(MAPBET(J,I),J=1,KW)
  405 CONTINUE

C GENERATE *MAPLXW* AND STORE ON IC1(1)

      DO 500 I=1,NX
      DO 500 J=1,KW
  500 MAPLXW(J,I) = (.NOT.MAPBET(J,I)).AND.(MAPOUT(J,I))

C STORE *MAPLXW* ON IC1(1)

      CALL POSITN ( IC1FP,1,IC1,523 )
      WRITE(IC1) MAPLXW
      CALL ENDFL(IC1FP,IC1)

C     GENERATE *LISTB* FROM *MAPBET*

      DO 610 I=1,NBTMX
  610 LISTB(I) = 0
      NBT = 0
      DO 690 I=1,NX
      DO 690 J=1,NY
      IF(MAPCHK( MAPBET,KW,NX,J,I).EQ.0) GO TO 690
      NBT = NBT + 1
      LISTB(NBT) = 10000*I + J
  690 CONTINUE
      IF( NBT.LE.NBTMX ) GO TO 750
      WRITE(IW,700) NBT,NBTMX
  700 FORMAT(1HO,* NBT EXCEEDS NBTMX * / 1X,*NBT,NBTMX ARE* 216 /
     *       1X,* JOB ABORTED * )
      CALL EXIT
  750 CONTINUE

C WRITE *LISTB* ON IC1(2)
```

```
      CALL POSITN ( IO1FP,2,IO1,524 )
      WRITE(IO1) (LISTB(I),I=1,NBT)
      CALL ENDFL(IO1FP,IO1)
      RETURN
  900 CONTINUE
      CALL POSITN (IO1FP,1,IO1,525)
      WRITE(IO1) MAPOUT

 6000 FORMAT(1H0,* MAPOUT * )
 6010 FORMAT(1X,2020)
 6020 FORMAT(1H0, * MAPBET * )

      RETURN
      END
```

```
      SUBROUTINE RCOT3(N,A,U,V,CONV)
C FINDS ROOTS OF A POLYNOMIAL DEGREE.LE.3  USING NEWTON-BAIRSTOW
      DIMENSION A(4),U(3),V(3),H(5),B(5),C(5)                          00000090
      NC=N+1                                                           00000100
      IF (A(NC)) 1,98,1                                                00000110
    1 CONV=1.0E-20                                                     00000120
      DO 2 I=1,NC                                                      00000130
    2 H(I)=A(I)                                                        00000140
      P=0.0                                                            00000150
      Q=0.0                                                            00000160
      R=0.0                                                            00000170
      IREV=1                                                           00000180
    3 IF (H(1)) 6,4,6                                                  00000190
    4 NC=NC-1                                                          00000200
      V(NC)=0.0                                                        00000210
      U(NC)=0.0                                                        00000220
      DO 5 I=1,NC                                                      00000230
    5 H(I)=H(I+1)                                                      00000240
      GO TO 3                                                          00000250
    6 IF (NC-1) 7,100,7                                                00000260
    7 IF (NC-2) 9,8,9                                                  00000270
    8 R=-H(1)/H(2)                                                     00000280
      GO TO 50                                                         00000290
    9 IF (NC-3) 11,10,11                                               00000300
   10 P=H(2)/H(3)                                                      00000310
      Q=H(1)/H(3)                                                      00000320
      GO TO 70                                                         00000330
   11 IF (ABS(H(NC-1)/H(NC))-ABS(H(2)/H(1))) 12,19,19                  00000340
   12 IREV=-IREV                                                       00000350
      M=NC/2                                                           00000360
      DO 13 I=1,M                                                      00000370
      NL=NC+1-I                                                        00000380
      F=H(NL)                                                          00000390
      H(NL)=H(I)                                                       00000400
   13 H(I)=F                                                           00000410
      IF (Q) 15,14,15                                                  00000420
```

```
14    P=0.C                                                          C0CC0430
      GC TC 16                                                       C0CC0440
15    P=P/Q                                                          C0CC0450
      Q=1.C/Q                                                        C0CC0460
16    IF (R) 17,19,17                                                C0CC0470
17    R=1.C/R                                                        00CC0480
  19  E=1.E-13
      B(NC)=H(NC)                                                    00CC0500
      C(NC)=H(NC)                                                    0CC00510
      B(NC+1)=0.C                                                    00000520
      C(NC+1)=C.C                                                    C0CC0530
      NP=NC-1                                                        0CCC0540
  20  CC 37 J=1,100
      CC 21 I1=1,NP                                                  C0CC0560
      I=NC-I1                                                        00CC0570
      B(I)=H(I)+R*B(I+1)                                             00000580
21    C(I)=B(I)+R*C(I+1)                                             00000590
      IF (ABS(B(1)/H(1))-E) 5C,5C,22                                 C0CC0600
22    IF (C(2)) 24,23,24                                             00000610
23    R=R+1.0                                                        C0CC0620
      GC TC 25                                                       C0CC0630
24    R=R-B(1)/C(2)                                                  00000640
25    CC 3C I1=1,NP                                                  0CC00650
      I=NC-I1                                                        C0CC0660
      B(I)=H(I)-P*B(I+1)-Q*B(I+2)                                    00000670
3C    C(I)=B(I)-P*C(I+1)-Q*C(I+2)                                    C0CC0680
      IF (H(2))32,31,32                                              0CCC0690
  31  IF (ABS(B(2)/H(1))-E)33,33,34                                  C0CC0700
  32  IF (ABS(B(2)/H(2))-E)33,33,34                                  0CCC0710
  33  IF (ABS(B(1)/H(1))-E)7C,7C,34                                  CCCC0720
34    CBAR=C(2)-B(2)                                                 C0CC0730
      D=C(3)**2-CBAR*C(4)                                            0CCC0740
      IF (D)36,35,36                                                 C0CC075C
35    P=P-2.0                                                        0CC00760
      Q=Q*(Q+1.0)                                                    C0CC0770
      GC TC 37                                                       0CCC0780
36    P=P+(B(2)*C(3)-B(1)*C(4))/D                                    00CC0790
      Q=Q+(-B(2)*CBAR+B(1)*C(3))/D                                   0CC00800
```

```
37    CONTINUE                                                  CCCC0810
      E=E*10.0                                                  CCCC0820
      CALL OVERFL(NES)                                           CCC0822
      IF (NES.EQ.1) GO TO 99                                    CCCC0824
      IF (E-CONV) 20,20,40                                      CCCC0830
40    CONV=E                                                    CCCC0840
      GO TO 20                                                  CCCC0850
50    NC=NC-1                                                   CCCC0860
      V(NC)=0.0                                                 CCCC0870
      IF (IREV)51,52,52                                         CCCC0880
51    L(NC)=1.0/R                                               CCCC0890
      GO TO 53                                                  CCCC0900
52    L(NC)=R                                                   CCCC0910
53    DO 54 I=1,NC                                              CCCC0920
54    B(I)=B(I+1)                                               CCCC0930
      GO TO 6                                                   CCCC0940
70    NC=NC-2                                                   CCCC0950
      IF (IREV)71,72,72                                         CCCC0960
71    CP=1.0/C                                                  CCCC0970
      PP=P/(C*2.0)                                              CCCC0980
      GO TO 73                                                  CCCC0990
72    CP=C                                                      CCCC1000
      PP=P/2.0                                                  CCCC1010
73    F=(PP)**2-CP                                              CCCC1020
      IF (F)74,75,75                                            CCCC1030
74    L(NC+1)=-PP                                               CCCC1040
      L(NC)=-PP                                                 CCCC1050
      V(NC+1)=SQRT(-F)                                          CCCC1060
      V(NC)=-V(NC+1)                                            CCCC1070
      GO TO 76                                                  CCCC1080
75    L(NC+1)=-SIGN(ABS(PP)+SQRT(F),PP)                         CCCC1090
      V(NC+1)=0.0                                               CCCC1100
      L(NC)=CP/L(NC+1)                                          CCCC1110
      V(NC)=0.0                                                 CCCC1120
76    DO 77 I=1,NC                                              CCCC1130
77    B(I)=B(I+2)                                               CCCC1140
      GO TO 6                                                   CCCC1150
98    CONV=-1.0                                                 CCCC1160
```

```
      RETURN                                                    CCCC1165
99    CONV=-2.0                                                 CCCC1170
100   RETURN                                                    CCCC1180
      END
```

```
      OVERLAY(OVER,6,0)
      PROGRAM LAXWDF

      COMMON / CM4B   / TPREV,T,INDCT,NT,NTFRST
      COMMON / CM8A   / NCTMX,NBTMX,NX,NY,KX,KY,KW,NBT,MM
      COMMON / CM1C   / ICCTL,TMX,NTMX,TMR
      COMMON / CM10A  / TIMA(2),TIMB(2),TIMS(2),TIMT(2),TIM(2,8)
      COMMON / CM11   / IR,IW
      COMMON / CM999  / NN,A(1)

      IF(NT.EQ.0) GO TO 999
      MM3 =MM + 3*NY
      NY3 = 3*NY
      NY2 = 2*NY
      N1 = 1
      N2 = N1 + MM3
      N3 = N2 + MM3
      N4 = N3 + MM3
      N5 = N4 + MM3
      N6 = N5 + NY3
      N7 = N6 + NY3
      N8 = N7 + NY3
      N9 = N8 + NY3
      N10 = N9 + NY3
      N11 = N10 +NY2
      N12 = N11 + NY2
      N13 = N12 + NY2
      N14 = N13 + NY2
      CALL LXWN ( A(N1),A(N2),A(N3),A(N4),A(N5),A(N6),A(N7),A(N8),A(N9),
     *            A(N10),A(N11),A(N12),A(N13), A(N14),
     *            MM3,NY3,NY2,KW,NX,NY )
  999 CONTINUE

      CALL TIMER(TMR,TIMA,TIMB,TIM(1,6))
      WRITE(IW,6090) TIM(1,6),TIM(2,6),T,NT
 6090 FORMAT(1HC,* CP,PP TIMES FOR  $ LAXWND $  ARE (SEC) *,2F10.2,5X
     $*T = * F10.6, 4X * NT = * I6 )
```

END

```
      SUBROUTINE LXWN ( w1,w2,w3,w4, F2,F3,F4, G3,G4, HF1,HF2,HF3,HF4,
     *                  MAPLXw,   MM3,NY3,NY2,Kw,NX,NY )

C         LAX-WND STAR        FOR  WFHALF            FOR WGHALF         FOR WFINAL
C                                                    P   Q   R              QQ
C          3  6  9            C .     . R           .   .   .               .
C
C          2  5  8            B .  *  . Q               *               AA.   *   .BB
C
C          1  4  7            A .     . P           .   .   .               .
C                                                    A   B   C              PP


      COMMON / CM2    / ZNX,ZNY,YZ,DX,DY,DS,DN,   RDX,RDY,RDS,RDN
      COMMON / CM3B   / ALFAR,W1FS,W2FS,W3FS,W4FS
      COMMON / CM4    / CF,CTMX, DXYSN, DTFRST,DTOUT,DTBND,DTSTB,DT
      COMMON / CM6    / ARTVS,GAMB
      COMMON / CM8B   / MS,ML,KBLKS
      COMMON / CM11B  / ID1,ID2,IL1,IL2,   ID1FP,ID2FP,IL1FP,IL2FP

      DIMENSION  w1(MM3),w2(MM3),w3(MM3),w4(MM3),
     *           F2(NY3),F3(NY3),F4(NY3),  G3(NY3),G4(NY3),
     *           HF1(NY2),HF2(NY2),HF3(NY2),HF4(NY2),  MAPLXw(Kw,NX),
     *           HG1(2),HG2(2),HG3(2),HG4(2)

C STATEMENT FUNCTIONS

      WFHALF ( WB, WQ, FB, FQ, GA, GP, GQ, GR ) =
     *         .5*(WB + WQ) + D1*(FQ - FB) + D2*(GQ + GR - GA - GP)

      WGHALF ( WB, WQ, GB, GQ, FA, FP, FQ, FR ) =
     *         .5*(WB + WQ) + D3*(GQ - GB) + D4*(FQ + FR - FA - FP)

      WFINAL ( W, FAA, FBB, GPP, GQQ ) =
     *         W + D5*(FBB - FAA) + D6*(GQQ - GPP)

C SET INDICATORS
```

```
      DO 10 K=1,NY3
   10 F4(K) = -10.
      DO 20 K=1,NY2
   20 FF4(K) = -10.
      L1 = 0
      L2 = 1
      L3 = 2
      LF1 = 0
      LF2 = 1

C POSITION *IL1* AT IL1(1)  AND  *IL2* AT IL2(1)

      CALL POSITN ( IO1FP,1,IO1,611 )
C POSITION IO1 AT IO1(1)
      READ(IO1) MAPLXW
      CALL POSITN(IL1FP,1,IL1,612)
      CALL POSITN ( IL2FP,1,IL2,613 )
      D1 = - DT/(2.*DX)
      D2 = - DT/(8.*DY)
      D3 = - DT/(2.*DY)
      D4 = - DT/(8.*DX)
      D5 = - DT/(   DX)
      D6 = - DT/(   DY)
      NY1 = NY - 1

C THIS IS THE DO LOOP OVER THE BLOCKS

      DO 900 KB=1,KBLKS
      M = MS
      IF(KB.EQ.KBLKS) M=ML
      KR1 = NY + 1
      KR2 = 2*NY
      KR3 = KR2 + 1
      KR4 = NY*(M+2)
      KR5 = KR4 + 1
      KR6 = NY*(M+3)
      IF(KB.EQ.1) GO TO 30
      BACKSPACE IL1
```

```
      BACKSPACE IL1
      READ(IL1) (W1(K),K=KR1,KR2),
     *          (W2(K),K=KR1,KR2),
     *          (W3(K),K=KR1,KR2),
     *          (W4(K),K=KR1,KR2)
      CALL FSR ( 1,IL1,IERR )
   30 READ(IL1) (W1(K),K=KR3,KR4),
     *          (W2(K),K=KR3,KR4),
     *          (W3(K),K=KR3,KR4),
     *          (W4(K),K=KR3,KR4)
      IF(KB.EQ.KBLKS) GO TO 40
      CALL FSR ( 1,IL1,IERR )
      READ(IL1) (W1(K),K=KR5,KR6),
     *          (W2(K),K=KR5,KR6),
     *          (W3(K),K=KR5,KR6),
     *          (W4(K),K=KR5,KR6)
   40 CONTINUE

      W1(1) = W1(NY) = W1FS
      W2(1) = W2(NY) = W2FS
      W3(1) = W3(NY) = W3FS
      W4(1) = W4(NY) = W4FS

      IF( KB.NE.1 ) GO TO 46
C SET UP LEFT EDGE OF FIRST BLOCK
      J1 = 3*NY + 1
      J2 = 4*NY
      JA = 2*NY
      DO 45  J=J1,J2
      JJ = J - JA
      W1(JJ) =  W1(J)
      W2(JJ) = -W2(J)
      W3(JJ) =  W3(J)
      W4(JJ) =  W4(J)
   45 CONTINUE
   46 IF ( KB.NE.KBLKS ) GO TO 56
C SET UP RIGHT EDGE IF THIS IS LAST BLOCK
      J1 = NY*(M+2) + 1
```

```
      J2 = NY*(M+3)
      DO 55 J=J1,J2
      W1(J) = W1FS
      W2(J) = W2FS
      W3(J) = W3FS
      W4(J) = W4FS
   55 CONTINUE
   56 CONTINUE
C THIS IS THE DO LOOP OVER THE COLUMNS IN A BLOCK
      DO 800 I=1,M
      IM1 = I - 1
      IP1 = I + 1
C INTERCHANGE AUXILIARY STORAGE ARRAYS AND RESET *COMPUTE* INDICATOR
      LSAVE = L1
      L1 = L2
      L2 = L3
      L3 = LSAVE
      K1 = L3*NY + 1
      K2 = K1 + NY - 1
      DO 60 K=K1,K2
   60 F4(K) = -10.
      LSAVE = LF1
      LF1 = LF2
      LF2 = LSAVE
      K1 = LF2*NY + 1
      K2 = K1 + NY - 1
      DO 70 K=K1,K2
   70 FF4(K) = -10.
      LG1 = 1
      LG2 = 2
      FG4(LG1) = -10.
      FG4(LG2) = -10.
C THIS IS THE DO LOOP WITHIN A COLUMN
      DO 700 J=2,NY1
      IJN = IM1*NY + J
      LSAVE = LG1
      LG1 = LG2
      LG2 = LSAVE
```

```
      FG4(LG2) = -10.
C DETERMINE IF (I,J) IS A LAX-WND POINT
      II = (KB -1)*MS + I
      IF(MAPCHK ( MAPLXW, KW, NX, J, II) .NE. 0 ) GO TO 90
      W1(IJN) = W2(IJN)  = W3(IJN) = W4(IJN) = -2.
      GO TO 700
   90 CONTINUE
C FIND POSITION IN W ARRAYS OF THE NINE POINTS
      K5 = IP1*NY + J
      K4 = K5 - 1
      K6 = K5 + 1
      K2 = K5 - NY
      K1 = K2 - 1
      K3 = K2 + 1
      K8 = K5 + NY
      K7 = K8 - 1
      K9 = K8 + 1

C CHECK IF POINT IS A FREE STREAM POINT

      IF(W4(K1).NE.W4FS) GO TO 100
      IF(W4(K2).NE.W4FS) GO TO 100
      IF(W4(K3).NE.W4FS) GO TO 100
      IF(W4(K4).NE.W4FS) GO TO 100
      IF(W4(K5).NE.W4FS) GO TO 100
      IF(W4(K6).NE.W4FS) GO TO 100
      IF(W4(K7).NE.W4FS) GO TO 100
      IF(W4(K8).NE.W4FS) GO TO 100
      IF(W4(K9).NE.W4FS) GO TO 100
      W1(IJN) = W1FS
      W2(IJN) = W2FS
      W3(IJN) = W3FS
      W4(IJN) = W4FS
      GO TO 700
  100  CONTINUE

C FIND POSITIONS IN F,G ARRAYS OF NINE POINTS
```

```
      KK2 = L1*NY + J
      KK1 = KK2 - 1
      KK3 = KK2 + 1
      KK5 = L2*NY + J
      KK4 = KK5 - 1
      KK6 = KK5 + 1
      KK8 = L3*NY + J
      KK7 = KK8 - 1
      KK9 = KK8 + 1
      KF1 = LF1*NY + J
      KF2 = LF2*NY + J

  110 IF(F4(KK1).NE.-10.)GO TO 120
      CALL FG(W1(K1),W2(K1),W3(K1),W4(K1),
     *F2(KK1),F3(KK1),F4(KK1),G3(KK1),G4(KK1))

  120 IF(F4(KK2).NE.-10.)GO TO 130
      CALL FG(W1(K2),W2(K2),W3(K2),W4(K2),
     *F2(KK2),F3(KK2),F4(KK2),G3(KK2),G4(KK2))

  130 IF(F4(KK3).NE.-10.)GO TO 140
      CALL FG(W1(K3),W2(K3),W3(K3),W4(K3),
     *F2(KK3),F3(KK3),F4(KK3),G3(KK3),G4(KK3))

  140 IF(F4(KK4).NE.-10.)GO TO 150
      CALL FG(W1(K4),W2(K4),W3(K4),W4(K4),
     *F2(KK4),F3(KK4),F4(KK4),G3(KK4),G4(KK4))

  150 IF(F4(KK5).NE.-10.)GO TO 160
      CALL FG(W1(K5),W2(K5),W3(K5),W4(K5),
     *F2(KK5),F3(KK5),F4(KK5),G3(KK5),G4(KK5))

  160 IF(F4(KK6).NE.-10.)GO TO 170
      CALL FG(W1(K6),W2(K6),W3(K6),W4(K6),
     *F2(KK6),F3(KK6),F4(KK6),G3(KK6),G4(KK6))

  170 IF(F4(KK7).NE.-10.)GO TO 180
      CALL FG(W1(K7),W2(K7),W3(K7),W4(K7),
```

```
     *F2(KK7),F3(KK7),F4(KK7),G3(KK7),G4(KK7))

  180 IF(F4(KK8).NE.-10.)GO TO 190
      CALL FG(W1(K8),W2(K8),W3(K8),W4(K8),
     *F2(KK8),F3(KK8),F4(KK8),G3(KK8),G4(KK8))

  190 IF(F4(KK9).NE.-10.)GO TO 200
      CALL FG(W1(K9),W2(K9),W3(K9),W4(K9),
     *F2(KK9),F3(KK9),F4(KK9),G3(KK9),G4(KK9))
  200 CONTINUE

C.COMPLTE F AT HALF VALUES

      IF(HF4(KF1).NE.-10.)GO TO 220
      HW1=WFHALF(W1(K2),W1(K5),W2(K2),W2(K5),
     *           W3(K1),W3(K4),W3(K3),W3(K6))
      HW2=WFHALF(W2(K2),W2(K5),F2(KK2),F2(KK5),
     *           F3(KK1),F3(KK4),F3(KK3),F3(KK6))
      HW3=WFHALF(W3(K2),W3(K5),F3(KK2),F3(KK5),
     *           G3(KK1),G3(KK4),G3(KK3),G3(KK6))
      HW4=WFHALF(W4(K2),W4(K5),F4(KK2),F4(KK5),
     *           G4(KK1),G4(KK4),G4(KK3),G4(KK6))
      CALL FF(HW1,HW2,HW3,HW4,HF1(KF1),HF2(KF1),HF3(KF1),HF4(KF1))
  220 CONTINUE

      HW1=WFHALF(W1(K5),W1(K8),W2(K5),W2(K8),
     *           W3(K4),W3(K7),W3(K6),W3(K9))
      HW2=WFHALF(W2(K5),W2(K8),F2(KK5),F2(KK8),
     *           F3(KK4),F3(KK7),F3(KK6),F3(KK9))
      HW3=WFHALF(W3(K5),W3(K8),F3(KK5),F3(KK8),
     *           G3(KK4),G3(KK7),G3(KK6),G3(KK9))
      HW4=WFHALF(W4(K5),W4(K8),F4(KK5),F4(KK8),
     *           G4(KK4),G4(KK7),G4(KK6),G4(KK9))
      CALL FF(HW1,HW2,HW3,HW4,HF1(KF2),HF2(KF2),HF3(KF2),HF4(KF2))
  230 IF(HG4(LG1).NE.-10.)GO TO 240

      HW1=WGHALF(W1(K4),W1(K5),W3(K4),W3(K5),
     *           W2(K1),W2(K2),W2(K7),W2(K8))
```

```
      HW2=WGHALF(W2(K4),W2(K5),F3(KK4),F3(KK5),
     *           F2(KK1),F2(KK2),F2(KK7),F2(KK8))
      HW3=WGHALF(W3(K4),W3(K5),G3(KK4),G3(KK5),
     *           F3(KK1),F3(KK2),F3(KK7),F3(KK8))
      HW4=WGHALF(W4(K4),W4(K5),G4(KK4),G4(KK5),
     *           F4(KK1),F4(KK2),F4(KK7),F4(KK8))
      CALL GG(HW1,HW2,HW3,HW4,HG1(LG1),HG2(LG1),HG3(LG1),HG4(LG1))
  24C CONTINUE

      HW1=WGHALF(W1(K5),W1(K6),W3(K5),W3(K6),
     *           W2(K2),W2(K3),W2(K8),W2(K9))
      HW2=WGHALF(W2(K5),W2(K6),F3(KK5),F3(KK6),
     *           F2(KK2),F2(KK3),F2(KK8),F2(KK9))
      HW3=WGHALF(W3(K5),W3(K6),G3(KK5),G3(KK6),
     *           F3(KK2),F3(KK3),F3(KK8),F3(KK9))
      HW4=WGHALF(W4(K5),W4(K6),G4(KK5),G4(KK6),
     *           F4(KK2),F4(KK3),F4(KK8),F4(KK9))
      CALL GG(HW1,HW2,HW3,HW4,HG1(LG2),HG2(LG2),HG3(LG2),HG4(LG2))
      W1(IJN) = WFINAL(W1(K5),HF1(KF1),HF1(KF2),HG1(LG1),HG1(LG2))
      W2(IJN) = WFINAL(W2(K5),HF2(KF1),HF2(KF2),HG2(LG1),HG2(LG2))
      W3(IJN) = WFINAL(W3(K5),HF3(KF1),HF3(KF2),HG3(LG1),HG3(LG2))
      W4(IJN) = WFINAL(W4(K5),HF4(KF1),HF4(KF2),HG4(LG1),HG4(LG2))
  7CC CONTINUE
  8CC CONTINUE

C STORE RESULTS AWAY

      KW1 = 1
      KW2 = NY
      KW3 = 1
      KW4 = M*NY
      KW5 = (M-1)*NY + 1
      KW6 = M*NY
      IF(KB.EQ.1) GO TO 820
      WRITE(IL2) (W1(K),K=KW1,KW2),
     *           (W2(K),K=KW1,KW2),
     *           (W3(K),K=KW1,KW2),
     *           (W4(K),K=KW1,KW2)
```

```
820 CONTINUE
    WRITE(IL2) (W1(K),K=KW3,KW4),
   *           (W2(K),K=KW3,KW4),
   *           (W3(K),K=KW3,KW4),
   *           (W4(K),K=KW3,KW4)
    IF(KB.EQ.KBLKS) GO TO 900
    WRITE(IL2) (W1(K),K=KW5,KW6),
   *           (W2(K),K=KW5,KW6),
   *           (W3(K),K=KW5,KW6),
   *           (W4(K),K=KW5,KW6)
900 CONTINUE
    CALL ENDFL(IL2FP,IL2)

    RETURN
    END
```

```
      SUBROUTINE FG(W1,W2,W3,W4,F2,F3,F4,G3,G4)
      COMMON / CM3A    / CT1,CT2,CT3,CT4,CT5
      RW1=1./W1
      U  =W2*RW1
      V  =W3*RW1
      RU2=W2*U
      RV2=W3*V
      P  =W4+CT3*(RU2+RV2)
      CC =CT2*P+W4
      DD =CT1*P
      F2 =DD+RU2
      F3 =W2*V
      F4 =U*CC
      G3=DD+RV2
      G4 =V*CC
      RETURN
      END
```

```
      SUBROUTINE FF(h1,h2,h3,h4,F1,F2,F3,F4)
      COMMON / CM3A    / CT1,CT2,CT3,CT4,CT5
      Rh1=1./h1
      U   =h2*Rh1
      V   =h3*Rh1
      RU2=h2*U
      RV2=h3*V
      P   =h4+CT3*(RU2+RV2)
      F1 =h2
      F2 =CT1*P + RU2
      F3 =h2*V
      F4 =U*(CT2*P + h4)
      RETURN
      END
```

```
      SUBROUTINE GG(W1,W2,W3,W4,G1,G2,G3,G4)
      COMMON / CM3A    / CT1,CT2,CT3,CT4,CT5
      RW1=1./W1
      L   =W2*RW1
      V   =W3*RW1
      RL2=W2*L
      RV2=W3*V
      P   =W4+CT3*(RL2+RV2)
      G1  =W3
      G2  =W2*V
      G3  =CT1*P + RV2
      G4  =V*(CT2*P + W4)
      RETURN
      END
```

```
      OVERLAY(OVER,7,0)
      PROGRAM BETWN

      COMMON / CM4A    / ICT,ICTFP,T1,T2,NC1,NCURV
      COMMON / CM4B    / TPREV,T,INDCT,NT,NTFRST
      COMMON / CM8A    / NCTMX,NBTMX,NX,NY,KX,KY,KW,NBT,MM
      COMMON / CM10A   / TIMA(2),TIMB(2),TIMS(2),TIMT(2),TIM(2,8)
      COMMON / CM1C    / ICCTL,TMX,NTMX,TMR
      COMMON / CM11    / IR,IW
      COMMON / CM999   / NN,A(1)

      IF(NT.EQ.0) GO TO 999
      NC = NC1
      MM2 = MM + 2*NY
      N1 = 1
      N2 = N1 + NBT
      N3 = N2 + 4*NBT
      N4 = N3 + 4*NBT
      N5 = N4 + NX
      N6 = N5 + KX*NX
      N7 = N6 + NY
      N8 = N7 + KY*NY
      N9 = N8 + KX*NX
      CALL LAPGEN ( A(N1), A(N2), A(N3), A(N4), A(N5), A(N6), A(N7),
     *              A(N8), A(N9),   KX,NX,KY,NY, NBT )
      N5 = N4 + 4*NBT
      CALL RHSBND ( A(N1), A(N2), A(N3), A(N4), A(N5),
     *                            A(N3),   NBT, NC )
      N5 = N4 + MM2
      N6 = N5 + MM2
      N7 = N6 + MM2
      N8 = N4 + 4*NBT
      CALL RHSLXW ( A(N1), A(N2), A(N3), A(N4), A(N5), A(N6), A(N7),
     *              A(N8),  NBT,MM2,NY )
      CALL LHS ( A(N1), A(N2), A(N3), A(N4),  NBT )
      CALL SOLVE ( A(N1), A(N2), A(N3), A(N4), A(N8),   NBT )
  999 CONTINUE
```

```
      CALL TIMER(TMR,TIMB,TIMA,TIM(1,7))
      WRITE(IW,6090) TIM(1,7),TIM(2,7),T,NT
6090 FORMAT(1HO,* CP,PP TIMES FOR  $ BETWN  $  ARE (SEC) *,2F10.2,5X
     $*T = * F10.6, 4X * NT = * I6 )

      END
```

```
      SUBROUTINE LAPGUN(LISTB,ARM,NSZ,IX,VX,IY,VY,NSX,NSY,
     *                  KX,NX,KY,NY,NBT)

C THIS SUBROUTINE COMPUTES VALUES OF THE LAPLACE-GUNNESS STAR

      DIMENSION LISTB(NBT),ARM(4,NBT),NSZ(4,NBT),
     *          IX(NX),VX(KX,NX),IY(NY),VY(KY,NY),NSX(KX,NX),NSY(KY,NY)

      COMMON / CM11   / IR,IW
      COMMON / CM2    / ZNX,ZNY,YZ,DX,DY,DS,DN,   RDX,RDY,RDS,RDN
      COMMON / CM11B  / ID1,ID2,IL1,IL2,   ID1FP,ID2FP,IL1FP,IL2FP
      COMMON / CM14   / NPRNT,IPRNT,NBUG,IBUG

C STATEMENT FUNCTIONS
      XMESH(I) = FLOAT(I-1)*DX
      YMESH(J) = FLOAT(J-1)*DY - YZ

C SET NSZ,ARM

      DO 10 I=1,NBT
      NSZ(1,I) = NSZ(2,I) = NSZ(3,I) = NSZ(4,I) = 0
      ARM(1,I) = ARM(3,I) = DY
      ARM(2,I) = ARM(4,I) = DX
   10 CONTINUE

C THIS IS THE BIG DO LOOP

C READ *LISTB* FROM ID1(2)
C READ *IX,VX,IY,VY,NSX,NSY FROM ID2(5)
      CALL POSITN ( ID1FP,2,ID1,711 )
      READ(ID1) LISTB
      CALL POSITN ( ID2FP,5,ID2,712 )
      READ(ID2) IX,VX,IY,VY,NSX,NSY

      DO 900 N=1,NBT
      I = LISTB(N)/10000
      J = MOD(LISTB(N),10000)
```

```
C CHECK SOUTH AND NORTH ARMS

C NOTE NINT=IX(I) IS AN EVEN NUMBER

      NINT = IX(I)
      IF(NINT.EQ.0) GO TO 300
      Y = YMESH(J)
      Y1= YMESH(J-1)
      Y2= YMESH(J+1)

C CHECK SOUTH POINT

      DO 120 KOO = 2,NINT
      K = 2 + NINT - KOO
      YY = VX(K,I)
      IF(YY.GT.Y) GO TO 120
      IF(YY.LT.Y1)GO TO 120
      ARM(1,N) = Y - YY
      NSZ(1,N) = NSX(K,I)
      GO TO 200
  120 CONTINUE

C CHECK NORTH POINT

  200 NINT1 = NINT - 1
      DO 220 K=1,NINT1,2
      YY = VX(K,I)
      IF(YY.GT.Y2) GO TO 220
      IF(YY.LT.Y ) GO TO 220
      ARM(3,N) = (YY-Y)
      NSZ(3,N) = NSX(K,I)
      GO TO 300
  220 CONTINUE

C CHECK EAST AND WEST POINTS
  300 NINT = IY(J)
      IF( NINT.EQ.0 ) GO TO 700
      X = XMESH(I)
```

```
      X1= XMESH(I-1)
      X2= XMESH(I+1)

C CHECK EAST POINT
C IF NINT IS EVEN CHECK ODD POINTS  $  IF NINT IS ODD CHECK EVEN POINTS

      IF( MOD(NINT,2).EQ.0 ) GO TO 310
      IF( NINT.EQ.1 ) GO TO 400
      K1 = 2
      K2 = NINT - 1
      GO TO 315
  310 K1 = 1
      K2 = NINT - 1
  315 DO 320 K=K1,K2,2
      XX = VY(K,J)
      IF(XX.GT.X2) GO TO 320
      IF(XX.LT.X)  GO TO 320
      ARM(2,N) = (XX-X)
      NSZ(2,N) = NSY(K,J)
      GO TO 400
  320 CONTINUE

C CHECK WEST POINT

  400 IF(I.EQ.1) GO TO 600
C IF NINT IS EVEN CHECK EVEN POINTS  $  IF NINT IS ODD CHECK ODD POINTS
      DO 420 KOO = 1,NINT
      K = 1 + NINT - KOO
      XX = VY(K,J)
      IF(XX.GT.X) GO TO 420
      IF(XX.LT.X1)GO TO 420
      ARM(4,N) = (X-XX)
      NSZ(4,N) = NSY(K,J)
      GO TO 700
  420 CONTINUE
      GO TO 700

C WEST POINT WHEN I=1
```

```
  600 ARM(4,N) = ARM(2,N)
      NSZ(4,N) = NSZ(2,N)

C FORM LAPLACE-GUNNESS STAR

  700 HS = ARM(1,N)
      HE = ARM(2,N)
      HN = ARM(3,N)
      HW = ARM(4,N)
      TOL = .000001*(DX+DY)*.5
      IF(HS.GT.TOL) GO TO 710
      ARM(1,N) = -1.
      ARM(2,N) = 0.
      ARM(3,N) = 0.
      ARM(4,N) = 0.
      GO TO 900
  710 IF(HE.GT.TOL) GO TO 720
      ARM(2,N) = -1.
      ARM(3,N) = 0.
      ARM(4,N) = 0.
      ARM(1,N) = 0.
      GO TO 900
  720 IF(HN.GT.TOL) GO TO 730
      ARM(3,N) = -1.
      ARM(4,N) = 0.
      ARM(1,N) = 0.
      ARM(2,N) = 0.
      GO TO 900
  730 IF(HW.GT.TOL) GO TO 800
      ARM(4,N) = -1.
      ARM(1,N) = 0.
      ARM(2,N) = 0.
      ARM(3,N) = 0.
      GO TO 900
  800 REW = 1./(HE + HW)
      RSN = 1./(HS + HN)
      HEW = HE*HW
```

```
      HSN = HS*HN
      HH  = 1./(HEW + HSN)
      EW  = HH*REW*HSN
      SN  = HH*RSN*HEW
      ARP(1,N) = -SN*HN
      ARP(2,N) = - EW*HW
      ARP(3,N) = -SN*HS
      ARP(4,N) = -EW*HE
  900 CONTINUE

      WRITE(IW,6005) NBT
 6005 FORMAT(1HO,* NO OF BETWEEN POINTS ARE  * I6 )
      WRITE(IW,6010)
 6010 FORMAT(1HO, * LISTB * )
      WRITE(IW,6020) LISTB
 6020 FORMAT(1X,1518 )
      IF(IBUG.EQ.0) GO TO 910
      WRITE(IW,6030)
 6030 FORMAT(1HO, * ARP * )
      WRITE(IW,6040) ARP
 6040 FORMAT(1X,4F15.6 )
      WRITE(IW,6050)
 6050 FORMAT(1HO,* NSZ * )
      WRITE(IW,6060) NSZ
 6060 FORMAT(1X,4I20)
  910 CONTINUE

      RETURN
      END
```

```
      SUBROUTINE RHSBND( LISTB, ARM, NSZ, RHS, WC,
     *                                 RHT,
     *                   NBT,                    NC)
C COMPUTES CONTRIBUTION TO RHS FROM BOUNDARY
      DIMENSION  LISTB(NBT), ARM(4,NBT), NSZ(4,NBT),RHS(4,NBT),WC(4,NC),
     *                               RHT(4,NBT)

      COMMON / CM11   / IR,IW
      COMMON / CM11B  / IC1,IC2,IL1,IL2,   IC1FP,IC2FP,IL1FP,IL2FP

C ZERO OUT *ARM*

      DO 10 I=1,NBT
      DO 10 J=1,4
   10 RHS(J,I) = 0.

C READ *WC* FROM IC2(4)
      CALL POSITN( IC2FP,4,IC2,721 )
      READ(IC2) WC

      DO 200 N=1,NBT
      DO 100 K=1,4
      NN = NSZ(K,N)
      IF(NN.EQ.0) GO TO 100
      IP1 = MOD(NN,10000)
      S1 = (1.E-08)*FLOAT(NN/10000)
      S2 = 1. - S1
      IP2 = IP1 + 1
      AR = ARM(K,N)
      ARM(K,N) = -10.
      DO 50 L=1,4
   50 RHS(L,N) = RHS(L,N) - AR*(S1*WC(L,IP2) + S2*WC(L,IP1))
  100 CONTINUE
  200 CONTINUE

C MOVE *RHS* TO *RHT*
      DO 300 I=1,NBT
      DO 300 J=1,4
```

```
300 RHT(J,I) = RHS(J,I)

    RETURN
    END
```

```
      SUBROUTINE RHSLXW (LISTB, ARM, RHS, W1, W2, W3, W4,
     *                                    RHT,    NBT,MM2,NY )

C COMPUTES CONTRIBUTION TO RHS FROM LAX-WND POINTS

      DIMENSION  LISTB(NBT), ARM(4,NBT), RHS(4,NBT), RHT(4,NBT),
     *           W1(MM2),W2(MM2),W3(MM2),W4(MM2)

      COMMON / CM8B   / MS,ML,KBLKS
      COMMON / CM11B  / IC1,IC2,IL1,IL2,   IC1FP,IC2FP,IL1FP,IL2FP
      COMMON / CM11   / IR,IW
      COMMON / CM14   / NPRNT,IPRNT,NBUG,IBUG

C POSITION  *IL2*  AT IL2(1)
      CALL POSITN( IL2FP,1,IL2,731 )

      DO 500 KB=1,KBLKS
      M = MS
      IF(KB.EQ.KBLKS) M=ML
      KR1 = 1
      KR2 = NY
      KR3 = KR2 + 1
      KR4 = NY*(M+1)
      KR5 = KR4 + 1
      KR6 = NY*(M+2)
      IF(KB.EQ.1) GO TO 30
      BACKSPACE IL2
      BACKSPACE IL2
      READ(IL2) (W1(K),K=KR1,KR2),
     *          (W2(K),K=KR1,KR2),
     *          (W3(K),K=KR1,KR2),
     *          (W4(K),K=KR1,KR2)
      CALL FSR ( 1,IL2,IERR )
   30 READ(IL2) (W1(K),K=KR3,KR4),
     *          (W2(K),K=KR3,KR4),
     *          (W3(K),K=KR3,KR4),
     *          (W4(K),K=KR3,KR4)
      IF(KB.EQ.KBLKS) GO TO 40
```

```
      CALL FSR ( 1,IL2,IERR )
      READ(IL2) (W1(K),K=KR5,KR6),
     *          (W2(K),K=KR5,KR6),
     *          (W3(K),K=KR5,KR6),
     *          (W4(K),K=KR5,KR6)
   40 CONTINUE

C CHECK LISTB FOR BETWEEN POINTS FROM I1 TO I2

      I1 = (KB-1)*MS + 1
      I2 = I1 + M - 1
      DO 400 N=1,NBT
      I = LISTB(N)/10000
      IF(I.LT.I1) GO TO 400
      IF(I.GT.I2) GO TO 400
      J = MOD(LISTB(N),10000)
      DO 300 K=1,4
      IF(ARM(K,N).LE.-10.) GO TO 300
      IF(ARM(K,N).LE.-10.) GO TO 300
      GO TO (210,220,230,240),K
  210 II = I
      JJ = J - 1
      GO TO 250
  220 II = I + 1
      JJ = J
      GO TO 250
  230 II = I
      JJ = J + 1
      GO TO 250
  240 II = I - 1
      JJ = J
      IF( II.EQ.0 ) II=2
  250 CONTINUE
      L = (II - I1 +1)*NY  + JJ
      IF(W4(L).LT.0) GO TO 300
      LIS = 10000*II + JJ
      DO 280 NZ = 1,NBT
      IF(LIS.EQ.LISTB(NZ)) GO TO 300
```

```
  280 CONTINUE
      AR = ARM(K,N)
      ARM(K,N) = -20.
      RHS(1,N) = RHS(1,N) - AR*W1(L)
      RHS(2,N) = RHS(2,N) - AR*W2(L)
      RHS(3,N) = RHS(3,N) - AR*W3(L)
      RHS(4,N) = RHS(4,N) - AR*W4(L)
  300 CONTINUE
  400 CONTINUE
  500 CONTINUE

C MOVE *RHS* TO *RHT*

      DO 600 I=1,NBT
      DO 600 J=1,4
  600 RHT(J,I) = RHS(J,I)

      IF(IBUG.EQ.0) GO TO 650
      WRITE(IW,6850)
 6850 FORMAT(1HO,* RHS * )
      WRITE(IW,6900) RHS
 6900 FORMAT(1X,8F15.8)

  650 CONTINUE
      RETURN
      END
```

```
      SUBROUTINE LHS ( LISTB, ARM, AA, NN,  NBT )

C COMPUTES THE AA MATRIX

      COMMON / CM11   / IR,IW
      COMMON / CM14   / NPRNT,IPRNT,NBUG,IBUG
      DIMENSION LISTB(NBT),ARM(4,NBT),AA(4,NBT),NN(4,NBT)

C CLEAR ARRAYS *AA*, *NN*

      DO 10 I=1,NBT
      DO 10 J=1,4
      AA(J,I) = 0.
      NN(J,I) = 0
   10 CONTINUE

C THIS IS THE BIG DO LOOP OVER ALL THE BETWEEN POINTS
C     EACH PASS SETS UP ONE ROW OF THE *AA* MATRIX

      DO 400 L=1,NBT
      I = LISTB(L)/10000
      J = MOD(LISTB(L),10000)
      IPOS = 0

C IN THIS DO LOOP THE 4 NEIGHBOURING POINTS OF BETWEEN POINT *L* ARE CHECKED
C     TO ASCERTAIN IF THEY ARE BETWEEN POINTS


      DO 300 K=1,4
      IF(ARM(K,L).LE.-10.) GO TO 300
      GO TO (110,120,130,140),K
  110 LIS = 10000*I + (J-1)
      GO TO 150
  120 LIS = 10000*(I+1) + J
      GO TO 150
  130 LIS = 10000*I + (J+1)
      GO TO 150
  140 IF(I.EQ.1) GO TO 300
```

```
      LIS = 10000*(I-1) + J
  150 CONTINUE

C FIND THE POSITION OF LIS IN LISTB

      DO 200 M=1,NBT
      IF(LIS.EQ.LISTB(M)) GO TO 220
  200  CONTINUE
      WRITE(IW,210)
  210 FORMAT(1X,*ERROR IN AAMTRX*)
      CALL EXIT
  220 IPOS = IPOS + 1
      EL = ARM(K,L)
      IF(I.NE.1) GO TO 250
      IF(K.NE.2) GO TO 250
      ARM(4,L) = -30.
      EL = 2.*EL
  250 CONTINUE
      ARM(K,L) = -30.
      AA(IPOS,L) = EL
      NN(IPOS,L) = M
  300 CONTINUE
  400 CONTINUE

      IF(IBUG.EQ.0)GO TO 410
      WRITE(IW,6010)
 6010 FORMAT(1HO, * AA * )
      WRITE(IW,6020) AA
 6020 FORMAT(1X,4F20.8)
      WRITE(IW,6030)
 6030 FORMAT(1HO, * NN *)
      WRITE(IW,6035) NN
 6035 FORMAT(1X,4I20)
      WRITE(IW,6040)
 6040 FORMAT(1HO, * ARM *)
      WRITE(IW,6020) ARM
  410 CONTINUE
      RETURN
```

END

```
      SUBROUTINE SOLVE( LISTB, SOL, AA, NN, RHS,   NBT )

C SOLVES LINEAR EQNS FOR BETWEEN POINTS

      DIMENSION LISTB(NBT), SOL(4,NBT), AA(4,NBT), NN(4,NBT), RHS(4,NBT)
     *         ,SOLNEW(4)

      COMMON / CM7    / TOLS,TOLX,TOLCB,TOLCF,TOLSOL
      COMMON / CM11   / IR,IW
      COMMON / CM11B  / IC1,IC2,IL1,IL2,   IC1FP,IC2FP,IL1FP,IL2FP
      COMMON / CM14   / NPRNT,IPRNT,NBUG,IBUG
      TOL = TOLSOL

C SET STARTING SOLUTION

      DO 20 I=1,NBT
      DO 10 J=1,4
      SOL(J,I) = RHS(J,I)
   10 CONTINUE
      IF((LISTB(I)/10000).EQ.1) SOL(2,I) = 0.
   20 CONTINUE
      ITER = 0
   50 ITER = ITER + 1
      ELMX4 = 0.
      DELMX4 = 0.

C EACH PASS THROUGH FOLLOWING LOOP IS ONE ITERATION

      DO 400 I=1,NBT

      DO 100 J=1,4
  100 SOLNEW(J) = RHS(J,I)

      DO 300 K=1,4
      NNOW = NN(K,I)
      IF(NNOW.EQ.0) GO TO 320
      ANOW = AA(K,I)
```

```
      DO 200 J=1,4
  200 SOLNEW(J) = SOLNEW(J) - ANCW*SCL(J,NNCW)

  300 CONTINUE

  320 IF((LIST8(I)/10000).EQ.1) SOLNEW(2) = 0.
      DELMX4 = AMAX1(DELMX4,ABS(SOLNEW(4) - SCL(4,I)))
      ELMX4  = AMAX1(ELMX4 , SOLNEW(4))

      DO 340 J=1,4
  340 SCL(J,I) = SOLNEW(J)

  400 CONTINUE

      IF(ITER.GT.200) GO TO 450
      IF(DELMX4.GT.(TOL*ELMX4)) GO TO 50
  450 CONTINUE
      RAT = DELMX4/ELMX4
      WRITE(IW,6000) ITER,DELMX4,ELMX4, RAT,TOL
 6000  FORMAT(1HO,* ITER,DELMX4,ELMX4,RATIO,TOLSOL ARE * I4,4F16.8)

C STORE *SOL* ON ID1(1)

      CALL POSITN( ID1FP,3,ID1,741 )
      WRITE(ID1) SOL
      CALL ENDFL(ID1FP,ID1)

      IF(IBUG.EQ.0)GO TO 500
      WRITE(IW,6850)
      WRITE(IW,6900) SOL
 6850 FORMAT(1HO,* SOL * )
 6900 FORMAT(1X,8F15.8)
  500 CONTINUE

      RETURN
      END
```

```
      OVERLAY(OVER,10,0)
      PROGRAM STBPRN

      COMMON / CM4B    / TPREV,T,INDCT,NT,NTFRST
      COMMON / CM8A    / NCTMX,NBTMX,NX,NY,KX,KY,Kw,NBT,MM
      COMMON / CM10    / ICCTL,TMX,NTMX,TMR
      COMMON / CM10A   / TIMA(2),TIMB(2),TIMS(2),TIMT(2),TIM(2,8)
      COMMON / CM11    / IR,Iw
      COMMON / CM13    / ISTART,ISTOP,ISAVE,NSAVE
      COMMON / CM999   / NN,A(1)

      IF(NT.EQ.0.AND.ISTART.EQ.1)GO TO 900
      IF(NT.EQ.0.AND.ISTART.EQ.4)GO TO 999
      N1 = 1
      N2 = N1 + NBT
      N3 = N2 + 4*NBT
      N4 = N3 + MM
      N5 = N4 + MM
      N6 = N5 + MM
      CALL STPR ( A(N1), A(N2), A(N3), A(N4), A(N5), A(N6),  NBT,MM,NY )
      GO TO 999
  900 CONTINUE
      N1 = 1
      N2 = N1*Kw*NX
      N3 = N2 + MM
      N4 = N3 + MM
      N5 = N4 + MM
      CALL STPRO(A(N1),A(N2),A(N3),A(N4),A(N5),  NX,NY,MM,Kw)
  999 CONTINUE

      CALL TIMER(TMR,TIMA,TIMB,TIM(1,8))
      WRITE(Iw,6090) TIM(1,8),TIM(2,8),T,NT
 6090 FORMAT(1HO,* CP,PP TIMES FOR  $ STBPRN $  ARE (SEC) *,2F10.2,5X
     $*T = * F10.6, 4X * NT = * I6 )

      END
```

```
      SUBROUTINE STPRO(MAPOUT,W1,W2,W3,W4, NX,NY,MM,KW )

C THIS SUBROUTINE IS CALLED WHEN NT = 0

      COMMON / CM3B   / ALFAR,W1FS,W2FS,W3FS,W4FS
      COMMON / CM4    / CF,DTMX, DXYSQ, DTFRST,DTOUT,DTBND,DTSTB,DT
      COMMON / CM8B   / MS,ML,KBLKS
      COMMON / CM11   / IR,IW
      COMMON / CM11B  / ID1,ID2,IL1,IL2,   ID1FP,ID2FP,IL1FP,IL2FP
      COMMON / CM14   / NPRNT,IPRNT,NBUG,IBUG
      DIMENSION MAPOUT(KW,NX),W1(MM),W2(MM),W3(MM),W4(MM)
      CALL POSITN(ID1FP,1,ID1,821)
      READ(ID1) MAPOUT
      CALL POSITN(IL1FP,1,IL1,822)

      DO 500 KB=1,KBLKS
      M = MS
      IF(KB.EQ.KBLKS)M = ML
      MTOT = M*NY
      I1 = (KB-1)*MS
      DO 100 II=1,M
      I =II + I1
      IM1 = I1 - 1
      DO 100 J=1,NY
      K = IM1*NY + J
      IF(MAPCHK(MAPOUT,KW,NX,J,I).EQ.0) GO TO 90
      W1(K) = W1FS
      W2(K) = W2FS
      W3(K) = W3FS
      W4(K) = W4FS
      GO TO 100
   90 W1(K) = W2(K) = W3(K) = W4(K) = -2.
  100 CONTINUE

C PRINT OUT W1,W2,W3,W4
      IF(IPRNT.EQ.0)GO TO 250

      I1=(KB-1)*MS+1
```

```
      I2=I1+M-1
      WRITE(IW,6010)
      DO 210 I=I1,I2
      K = (I-I1)*NY
      K1 = K + 1
      K2 = K + NY
  210 WRITE(IW,6000)I,(W1(K),K=K1,K2)

      WRITE(IW,6020)
      DO 220 I=I1,I2
      K = (I-I1)*NY
      K1 = K + 1
      K2 = K + NY
  220 WRITE(IW,6000)I,(W2(K),K=K1,K2)

      WRITE(IW,6030)
      DO 230 I=I1,I2
      K = (I-I1)*NY
      K1 = K + 1
      K2 = K + NY
  230 WRITE(IW,6000)I,(W3(K),K=K1,K2)

      WRITE(IW,6040)
      DO 240 I=I1,I2
      K = (I-I1)*NY
      K1 = K + 1
      K2 = K + NY
  240 WRITE(IW,6000)I,(W4(K),K=K1,K2)
  250 CONTINUE

      IF(KE.EQ.1) GO TO 420
      K1 = 1
      K2 = NY
      WRITE(IL1) (W1(K),K=K1,K2),
     *           (W2(K),K=K1,K2),
     *           (W3(K),K=K1,K2),
     *           (W4(K),K=K1,K2)
  420 CONTINUE
```

```
      K1 = 1
      K2 = MTCT
      WRITE(IL1) (W1(K),K=K1,K2),
     *           (W2(K),K=K1,K2),
     *           (W3(K),K=K1,K2),
     *           (W4(K),K=K1,K2)
      K1 = MTCT-NY+1
      K2 = MTCT
      WRITE(IL1) (W1(K),K=K1,K2),
     *           (W2(K),K=K1,K2),
     *           (W3(K),K=K1,K2),
     *           (W4(K),K=K1,K2)

  300 CONTINUE
  500 CONTINUE
      CALL ENDFL(IL1FP,IL1)
      CTCLT = CTFRST


 6000 FORMAT(1X,I3,2X,15F8.4,/(6X,15F8.4))
 6010 FORMAT(1H0,*W1*/)
 6020 FORMAT(1H0,*W2*/)
 6030 FORMAT(1H0,*W3*/)
 6040 FORMAT(1H0,*W4*/)

      RETURN
      END
```

```
      SUBROUTINE STPR ( LISTB, SCL, W1,W2,W3,W4,  NBT,MM,NY )

C FILLS *W* ARRAYS WITH BETWEEN POINT VALUES.  PRINTS W.  FINDS DTOUT


      COMMON / CM3A   / CT1,CT2,CT3,CT4,CT5,CT6,CT7,CT8
      COMMON / CM3B   / ALFAR,W1FS,W2FS,W3FS,W4FS
      COMMON / CM4    / CF,CTMX, CXYSN, DTFRST,DTOUT,DTBND,DTSTB,DT
      COMMON / CM4B   / TPREV,T,INDCT,NT,NTFRST
      COMMON / CM8B   / MS,ML,KBLKS
      COMMON / CM11   / IR,IW
      COMMON / CM11B  / ID1,ID2,IL1,IL2,   IC1FP,IC2FP,IL1FP,IL2FP
      COMMON / CM14   / NPRNT,IPRNT,NBUG,IBUG

      DIMENSION LISTB(NBT),SCL(4,NBT),W1(MM),W2(MM),W3(MM),W4(MM)

C READ * LISTB,SCL * FROM IC1(2),IC1(3)

      CALL POSITN( IC1FP,2,IC1,811 )
      READ(ID1) LISTB

      CALL POSITN( IC1FP,3,IC1,812 )
      READ(ID1) SCL

C POSITION *IL2* AT IL2(1)
C POSITION *IL1* AT IL1(1)
      CALL POSITN( IL2FP,1,IL2,813 )
      CALL POSITN( IL1FP,1,IL1,814 )


      CMAX = 0.
      DO 500 KB=1,KBLKS
      M = MS
      IF(KB.EQ.KBLKS) M=ML
      MTOT = M*NY
      IF(KB.NE.1) CALL FSR(2,IL2,IERR)
      READ(IL2) (W1(I),I=1,MTOT),
     *          (W2(I),I=1,MTOT),
```

```
     *           (W3(I),I=1,MTCT),
     *           (W4(I),I=1,MTCT)

C INSERT BETWEEN POINTS

      I1 = (KB-1)*MS + 1
      I2 = I1 + M - 1

      DO 100 N=1,NBT
      II = LISTB(N)/10000
      J = MOD(LISTB(N),10000)
      IF(II.LT.I1)GO TO 100
      IF(II.GT.I2)GO TO 100
      I = II - I1
      K = I*NY + J
      W1(K) = SCL(1,N)
      W2(K) = SCL(2,N)
      W3(K) = SCL(3,N)
      W4(K) = SCL(4,N)
  100 CONTINUE

C PRINT OUT W1,W2,W3,W4
      IF(IPRNT.EQ.0) GO TO 250

      WRITE(IW,6010)
      DO 210 I=I1,I2
      K = (I-I1)*NY
      K1 = K + 1
      K2 = K + NY
  210 WRITE(IW,6000)I,(W1(K),K=K1,K2)

      WRITE(IW,6020)
      DO 220 I=I1,I2
      K = (I-I1)*NY
      K1 = K + 1
      K2 = K + NY
  220 WRITE(IW,6000)I,(W2(K),K=K1,K2)
```

```
      WRITE(IW,6030)
      DO 230 I=I1,I2
      K = (I-I1)*NY
      K1 = K + 1
      K2 = K + NY
  230 WRITE(IW,6000)I,(W3(K),K=K1,K2)

      WRITE(IW,6040)
      DO 240 I=I1,I2
      K = (I-I1)*NY
      K1 = K + 1
      K2 = K + NY
  240 WRITE(IW,6000)I,(W4(K),K=K1,K2)
  250 CONTINUE

C COMPUTE MAXIMUM VALUE OF (A+VEL) FROM THIS BLOCK

      DO 300 K=1,MTOT
      IF(W4(K).EQ.W4FS) GO TO 300
      IF(W4(K).EQ.-2.) GO TO 300
      RR = 1./W1(K)
      U = RR*W2(K)
      V = RR*W3(K)
      VSQ = U*U + V*V
      VEL = SQRT(VSQ)
      TEMP = RR*W4(K) + GT3*VSQ
      IF(TEMP.LT.0.) GO TO 270
      DNOW = GT4*SQRT(ABS(TEMP)) + VEL
      DMAX = AMAX1(DMAX,DNOW)
      GO TO 300

  270 CONTINUE
      KK1 = K-1
      KK2 = K+1
      KK3 = K - NY
      KK4 = K + NY
      WRITE(IW,271) KB,NY,K,W1(K),W2(K),W3(K),W4(K),TEMP
  271 FORMAT(1HO,* KB,NY,K,W1(K),W2(K),W3(K),W4(K),TEMP ARE *
```

```
     * /1X,3I4,5E20.6 )
      W1(K) = .25*(W1(KK1)+W1(KK2)+W1(KK3)+W1(KK4))
      W2(K) = .25*(W2(KK1)+W2(KK2)+W2(KK3)+W2(KK4))
      W3(K) = .25*(W3(KK1)+W3(KK2)+W3(KK3)+W3(KK4))
      W4(K) = .25*(W4(KK1)+W4(KK2)+W4(KK3)+W4(KK4))
      WRITE(IW,272) W1(K),W2(K),W3(K),W4(K)
  272 FORMAT(1HC * NEW W1(K),W2(K),W3(K),W4(K) ARE *
     * /13X, 4E20.6)

  300 CONTINUE

      IF(KB.EQ.1) GO TO 420
      K1 = 1
      K2 = NY
      WRITE(IL1) (W1(K),K=K1,K2),
     *           (W2(K),K=K1,K2),
     *           (W3(K),K=K1,K2),
     *           (W4(K),K=K1,K2)
  420 CONTINUE
      K1 = 1
      K2 = MTOT
      WRITE(IL1) (W1(K),K=K1,K2),
     *           (W2(K),K=K1,K2),
     *           (W3(K),K=K1,K2),
     *           (W4(K),K=K1,K2)
      K1 = MTOT-NY+1
      K2 = MTOT
      WRITE(IL1) (W1(K),K=K1,K2),
     *           (W2(K),K=K1,K2),
     *           (W3(K),K=K1,K2),
     *           (W4(K),K=K1,K2)

  500 CONTINUE
      CALL ENDFL(IL1FP,IL1)
      DTOLT = DXYSN/DMAX


 6000 FORMAT(1X,I3,2X,15F8.4,/(6X,15F8.4))
```

```
6010 FORMAT(1HC,*W1*/)
6020 FORMAT(1HO,*W2*/)
6030 FORMAT(1HO,*W3*/)
6040 FORMAT(1HC,*W4*/)

      RETURN
      END
```

5319 CARDS

* U. S. GOVERNMENT PRINTING OFFICE : 1972 720-390/320