SUPPLEMENT TO
NASA CR-1218

A NUMERICAL SOLUTION FOR THE MINIMUM INDUCED DRAG, AND THE CORRESPONDING LOADING, OF NONPLANAR WINGS

I - USER'S MANUAL FOR COMPUTER PROGRAM 55VD

II - DESCRIPTIVE MANUAL FOR COMPUTER PROGRAM 55VD

By J. L. Lundry

Distribution of this report is provided in the interest of information exchange. Responsibility for the contents resides in the author or organization that prepared it.

Prepared under Contract No. NAS 1-7484 by
THE McDONNELL DOUGLAS CORPORATION
Long Beach, Calif.

for Langley Research Center

NATIONAL AERONAUTICS AND SPACE ADMINISTRATION

For sale by the Clearinghouse for Federal Scientific and Technical Information
Springfield, Virginia 22151 – CFSTI price $3.00

A NUMERICAL SOLUTION FOR THE MINIMUM INDUCED DRAG, AND THE CORRESPONDING LOADING, OF NONPLANAR WINGS

I – USER'S MANUAL FOR COMPUTER PROGRAM 55VD

Issued by Originator as Report DAD-66898

A NUMERICAL SOLUTION FOR THE MINIMUM INDUCED DRAG, AND THE CORRESPONDING LOADING, OF NONPLANAR WINGS — USER'S MANUAL FOR COMPUTER PROGRAM 55VD

by J. L. Lundry

The McDonnell Douglas Corporation

SUMMARY

A numerical procedure has been developed for the accurate computation of the minimum induced drag, and the associated load distribution, of nonplanar wings. As is well known, the minimum induced drag may be determined by analyzing a potential problem in the Trefftz plane. The procedure solves the potential problem in an auxiliary mapping plane related to the physical plane by the Schwarz-Christoffel transformation for configurations with front views that may be approximated by straight line segments. Previously, the main computational difficulty had been the determination of the mapping constants. The procedure presented here offers a choice of two methods to determine these constants. By means of the electrostatic analogy to potential flow, the mapping constants can be measured with a small experimental error with an analog field plotter. The mapping derivative is then integrated numerically, and the constants are adjusted iteratively until the desired geometry is achieved to high order of accuracy.

Alternatively, the mapping constants can be determined without resort to analog experiment by evaluating them for a series of geometrically related configurations that ends with the desired configuration. The mapping constants for each member of the series are used as the equivalent of experimental values for the next member of the series, whose mapping constants in turn can then be determined with the iteration scheme employed in the experimental method. The series starts with the monoplane degenerate equivalent of the desired configuration, for which the mapping constants are known. This procedure is successful if each member of the series differs geometrically from its neighbors by a small amount.

Once the mapping constants are known, the minimum induced drag and the associated loading are determined by quadrature.

The procedure has been applied to seven nonplanar lifting configurations, resulting in the development of Computer Program 55VD in the FORTRAN IV language for use on an IBM 7094. Program 55VD has been converted to FORTRAN 2.0 for use on the Langley Research Center's CDC 6000 series digital computers. This report is a user's manual for the FORTRAN 2.0 version of Program 55VD.

INTRODUCTION

Munk's Theory

In Reference 1, Munk developed a theory for the minimum induced drag, and the associated loading, of arbitrary lifting configurations. All loadings are assumed light, so that velocity perturbations are small and the vortex wake in the Trefftz plane may be assumed undistorted. The loadings can be projected onto a plane normal to the free-stream velocity without changing the induced drag of the lifting system (Stagger Theorem). Munk's criterion for minimum induced drag is illustrated in Figure 1, and requires the induced velocity normal to the projected loadings to be proportional to the cosine of the angle of lateral inclination of the projected loadings. Munk further demonstrates that the loading to satisfy this criterion can be found by solving a potential flow problem about the wake in the Trefftz plane, in which the undisturbed flow is parallel to the downwash. The required loading is locally proportional to the potential difference across the wake and is normal to the wake.

Applications of Munk's Theory

Munk applied his theory to the monoplane, and obtained the classic result that a constant downwash across the span produces the minimum induced drag

$$D = \frac{L^2}{4 \pi q s^2} \tag{1}$$

and is given by an elliptical distribution of load. In References 2-6, Munk's theory is applied analytically to nonplanar configurations consisting either of combinations of a monoplane with vertical fences or a wing with part-span or full-span dihedral. In Reference 7, the electrostatic analogy to potential flow is exploited to determine experimentally loadings that satisfy Munk's criterion for complex nonplanar lifting configurations; the lift and the minimum induced drag are then evaluated numerically. Unfortunately, the numerical results of this conceptually powerful method can contain significant errors, as shown by Reference 8. Reference 8 solves the potential problem in an auxiliary mapping plane related to the real (Trefftz) plane by the Schwarz-Christoffel transformation for configurations with front views that may be approximated by straight line segments. Previously, the main computational difficulty with this approach had been the determination of the mapping constants. By means of the electrostatic analogy to potential flow, the mapping constants can be measured with a small experimental error with an analog field plotter. The transformation derivative is then integrated numerically, and the constants are adjusted iteratively until the desired geometry is obtained to high order of accuracy. Once the mapping constants are known, the minimum induced drag is determined by quadrature. A digital computer program was required to obtain the numerical results of Reference 8 for Configurations 1-5 of this report.

Reference 9 proposes extensions to Reference 8 and the digital computer program — principally:

1) Application of the numerical scheme of Reference 8 to Configurations 6 and 7 of this report.

2) Calculation of the loading to produce minimum induced drag for Configurations 1-7.

3) Development of an alternative method to determine the mapping constants so that the analog experiments could be eliminated from the method of Reference 8.

The proposed extensions to Reference 8 and to the digital computer program were funded under NASA Contract NAS-1-7484. This report is a user's manual for the extended digital computer program, which is described in detail in Reference 10. Reference 11 summarizes all work sponsored by the contract.

SYMBOLS

A,B,C,D,E,F,G	potentials in the real or physical plane (See Figure 3)
C_i	the i-th required geometry condition in the real plane
ΔC_i	error in C_i
D	minimum induced drag
I	the imaginary part of
L	lift
a,b,c,d,e,f,g	potentials in the auxiliary plane (See Figure 3)
$\bar{e}$	the square root of the sum of the squares of the errors in the required geometry conditions: $\bar{e} = \sqrt{\sum_{i=1}^{n} (\Delta C_i)^2}$
k	induced drag efficiency at minimum induced drag
ℓ	nondimensional length of fence or pylon
n	number of required geometry conditions C_i
p_j	the j-th mapping constant or potential
Δp_j	correction to p_j

q	dynamic pressure
s	semispan
z	complex variable in the real plane
Γ	dihedral angle in degrees
ζ	complex variable in the auxiliary plane
η	nondimensional semispan coordinate

ANALYSIS

The basic theory for the minimum induced drag of nonplanar lifting configurations is given in Reference 1 and is described in the INTRODUCTION. A potential problem is formulated about the vortex wake in the Trefftz plane. By the method of Reference 8, the problem is analyzed numerically in an auxiliary mapping plane by means of the Schwarz-Christoffel transformation. First, the constants of the Schwarz-Christoffel mapping must be determined. The minimum induced drag can then be obtained by quadrature in the form k. Finally, the required load distribution can be obtained by interpolation. The seven general configurations sketched in Figure 2 are analyzed by Computer Program 55VD.

The Mapping Constants

Two methods to determine the mapping constants have been included in program 55VD and are described here.

Electrostatic analog with iterative correction. This method is developed in Reference 8. By means of the electrostatic analogy to potential flow, the mapping constants can be measured experimentally with an analog field plotter, such as the one used by the author of Reference 7. The measured values of potential contain experimental errors, and are corrected by an iterative scheme that will be explained for Configuration 5 as an example. Figure 3 shows the Trefftz plane (also called real or physical plane) and the auxiliary mapping plane for Configuration 5. The Schwarz-Christoffel transformation between these planes is

$$\frac{dz}{d\zeta} = \frac{(\zeta - c)(\zeta - e)}{[(\zeta - a)(\zeta - b)(\zeta - g)]^{\frac{1}{2}}} (\zeta - d)^{\frac{\Gamma}{\pi} - \frac{1}{2}} (\zeta - f)^{-\frac{\Gamma}{\pi}} \qquad (2)$$

Given the experimental values of potential a, . . . , g, the geometry in the physical plane is evaluated by numerically integrating Equation (2).

For example, the length $\overline{AB}$ is

$$\overline{AB} = z_B - z_A = \int_a^b \frac{dz}{d\zeta}\, d\zeta \tag{3}$$

Each of the distances $\overline{AB}$, $\overline{BC}$, $\overline{CD}$, $\overline{DE}$, $\overline{EF}$, and $\overline{FG}$ can be evaluated in this way, and each of the integrals has either one or two integrable singularities at the end or ends of the integration interval. If an integral has two singularities, it is evaluated in two parts of equal range so that each numerical integral has at most one singularity. For example, the integral for the length $\overline{AB}$ has two singularities, and is evaluated as

$$\overline{AB} = \int_a^{\frac{1}{2}(a+b)} \frac{dz}{d\zeta}\, d\zeta + \int_{\frac{1}{2}(a+b)}^{b} \frac{dz}{d\zeta}\, d\zeta \tag{4}$$

The first integral is singular at a in the form

$$\int_a^{\frac{1}{2}(a+b)} \frac{Z(\zeta)}{(\zeta - a)^{\frac{1}{2}}}\, d\zeta \tag{5}$$

and can be rewritten in the standard way as

$$\int_a^{\frac{1}{2}(a+b)} \frac{Z(\zeta) - Z(a)}{(\zeta - a)^{\frac{1}{2}}}\, d\zeta + Z(a) \int_a^{\frac{1}{2}(a+b)} \frac{1}{(\zeta - a)^{\frac{1}{2}}}\, d\zeta$$

$$= \int_a^{\frac{1}{2}(a+b)} \frac{Z(\zeta) - Z(a)}{(\zeta - a)^{\frac{1}{2}}}\, d\zeta + 2Z(a)\, [\tfrac{1}{2}(b - a)]^{\frac{1}{2}} \tag{6}$$

The integrand $[Z(\zeta) - Z(a)]/[\zeta - a]^{\frac{1}{2}}$ is evaluated at up to fifty points on the integration interval that are spaced more closely near the singularity, and a modified Simpson rule is used to perform the quadrature.

In the example of Configuration 5, the mapping derivative contains seven unknown constants. Six of these constants are independent (one must

be fixed to define a coordinate origin) and determine the six lengths $\overline{AB}$, $\overline{BC}$, $\overline{CD}$, $\overline{DE}$, $\overline{EF}$, and $\overline{FG}$ in Figure 3. For practical reasons, the experimental values of the mapping constants are scaled linearly before they are used to evaluate the geometry in the real plane; this linear scaling changes only the absolute scale in the real plane, is equivalent to fixing one of the six independent mapping constants, and, for Configuration 5, is defined by setting b = 0 and e = 1. Five of the unknown mapping constants are used as the independent variables in a linear iteration scheme that is designed to satisfy the five required geometry conditions:

1) The fence must close ($\overline{BC} = \overline{CD}$).

2) The inboard wing must close ($\overline{AB} = \overline{FG}$).

3) The outboard wing must close ($\overline{DE} = \overline{EF}$).

4) The fence must have the proper length ($\overline{BC} = \ell[\overline{AB} + \overline{DE} \cos \Gamma]$).

5) The fence must have the proper semispan location ($\overline{AB} = \eta[\overline{AB} + \overline{DE} \cos \Gamma]$).

The initial values of the unknown mapping constants are measured experimentally with an analog field plotter. The corrections to the mapping constant Δp_j are computed from

$$- \Delta C_i = \frac{\partial C_i}{\partial p_j} \Delta p_j \tag{7}$$

where ΔC_i is the error in C_i, the i-th geometry requirement. The 5 x 5 correction matrix $\partial C_i/\partial p_j$ is evaluated numerically by perturbing p_j slightly, scaling the potentials, if necessary, and calculating the derivative as though C_i varies linearly with p_j. For Configuration 5, the potentials a, b, d, e, and g were selected from the seven unknown values of potential as the independent variables for the iteration. This simple iteration scheme converges rapidly for properly chosen independent variables. For some of the seven configurations, the set of mapping constants selected initially as independent variables did not give convergence, and other sets were chosen. However, convergence of the iteration scheme has been achieved for each of the seven configurations considered to date.

Successive solution scheme. Reference 9 proposed the development of a successive solution scheme to determine the unknown constants of the Schwarz-Christoffel mapping without resort to electrostatic analog experiments. The mapping constants are determined for a series of geometrically related configurations that ends with the desired configuration. The final values of the mapping constants for each member of the series are used as initial values for the next member of the series, whose mapping constants in turn can then be determined with the iteration scheme developed for the electrostatic analog method. The series is started with the monoplane

degenerate equivalent of the desired configuration, for which the mapping constants can be determined from the potential solution about a monoplane, given in Reference 12. This procedure to determine the mapping constants is generally successful if each member of the series differs geometrically from its neighbors by a small amount.

The Minimum Induced Drag

Once the mapping constants are known, the minimum induced drag is calculated in the terms of the efficiency k, where

$$D = \frac{L^2}{4 \pi s^2 q k} \tag{8}$$

If the expression for k is transformed into the auxiliary plane, Reference 8 shows that

$$k = \frac{1}{\pi s^2} I \int_a^g \zeta \frac{dz}{d\zeta} d\zeta \tag{9}$$

where I means "the imaginary part of". Again, the integration is performed numerically.

The Loading to Produce Minimum Induced Drag

Once the mapping constants are known, the potential is specified effectively in the physical plane as a function of geometric location. The required loading is proportional to the potential difference across each element and is obtained by interpolation in the tables resulting from the numerical integration that produces lengths in the real plane.

COMPUTER PROGRAM 55VD

Computer Program 55VD has been written in the FORTRAN IV language for an IBM 7094, and has been converted to the FORTRAN 2.0 language for use on the CDC 6000 series computers at the Langley Research Center. The main program and the twenty-four subroutines of Program 55VD are written on roughly 3700 FORTRAN source cards. Computing time per case varies from 0.1 minute to 3 minutes, depending on which of the seven configurations is being analyzed, the number of ordinates used in the evaluation of the geometry and drag numerical integrals, and the rate of convergence of the iteration for the mapping constants.

Overall Logic of Program 55VD

Figure 4 shows the overall logical flow of Computer Program 55VD. In several of the subroutines, internal logic is used to transfer to coding associated with the specific configuration being considered; such logic is not included in Figure 4. Basically, Program 55VD calculates just three types of results. First, it computes the mapping constants of the Schwarz-Christoffel transformation, either by the method of electrostatic analog with iterative correction, or by the successive solution scheme. The program then computes the induced drag efficiency parameter k. Finally, it computes the loading required to give the minimum induced drag.

Use and Operation of Program 55VD

Input Data. The input data for Program 55VD are specified by either four data cards or two data cards, depending on whether the program computes the Schwarz-Christoffel mapping constants by the electrostatic analog method (program operation mode = 1) or by the successive solution scheme (program operation mode = 2), respectively. The format for the four data input cards is illustrated in Figure 5. The first card is required for each case and contains four integer numbers. The integer in card column 2 is an execution termination flag, is 0 if other cases follow, and is 1 if the case is the last to be executed. The integer in card column 4 is the program operation mode, and is 1 or 2. The integer in card column 6 is the configuration number corresponding to the numbers given in Figure 2. The integer in card columns 9 and 10 is the number of ordinates in each of the Simpson rule numerical integrations for either geometry or drag and can vary from 10 to 50.

The second and third data input cards are required only if the program operation mode is 1. These two cards specify the initial values of the Schwarz-Christoffel mapping constants and contain 6 to 9 real numbers, depending on which of the seven configurations of Figure 2 is being analyzed. These initial values are scaled linearly by the program.

The fourth card (or second card, if program operation mode is 2) specifies the geometry in the real plane. From 3 to 5 real numbers are specified on this card. Table I defines the values required on this card for each of the seven configurations. Note that dummy input is required for some of the configurations.

Card columns 51-80 are not read on any of the four input data cards. These 30 card columns may be used for deck sequencing and deck identification.

Table I

Use of Fourth Data Input Card

Computer Program 55VD

Card Column / Configuration See Figure 2	1 - 10	11 - 20	21 - 30	31 - 40	41 - 50
1	η	(a)	ℓ	(b)	(b)
2	η_i	η_o	ℓ_i	ℓ_o	(b)
3	ℓ_H	(a)	s_H	(b)	(b)
4	η_i	η_o	ℓ_i	ℓ_o	(b)
5	η	(a)	ℓ	(a)	Γ Degrees
6	η_i	η_o	ℓ	(a)	Γ Degrees
7	ℓ_i	ℓ	ℓ	(b)	(b)

(a) Use a dummy value for this number, say 0.0.

(b) This number should not be entered.

Numerical Output of Program 55VD. An example of Program 55VD output is given in Figure 6. The first page of output gives the following information: the configuration number; the program operation mode; the number of ordinates used in each of the numerical integrations for geometry or drag; and, if program operation mode is 1, the unscaled (input) and scaled initial values of the Schwarz-Christoffel mapping constants.

The second page of output in Figure 6 first provides summary data for the iteration that determines the mapping constants. The first line of numerical data gives the initial values of the mapping constants, the errors in the required geometry conditions, and the square root of the sum of the squares of the errors. Below the first line of numerical output, the correction matrix, $\delta\sigma_i/\delta\rho_j$, is printed without a heading. Following the correction matrix are seven lines (the number varies from case to case) of numerical data identical in form with the first line. Each of the first six of these seven lines summarizes the numerical results of one cycle of the iteration. In this example, the iteration converges quite rapidly. The last line of this output repeats the results of the line above that has the smallest value of $\bar{e}$. For this case, the last line is identical to the line immediately above it. Below the last line of the iteration summary data, the final values of the mapping constants are printed in exponential format. The balance of the data on the second page of output describes the geometry specified by the final values of the mapping constants, gives the numerical result for the induced drag integral, and gives the minimum induced drag parameter k.

The third page of output in Figure 6 specifies the loading required to produce the minimum induced drag. The ratio of the gross load on the pylon or fence to the gross lift on the wing semispan is given first. Then, tabular data specify the required loading on both wing and pylon; for both surfaces, the local loading is nondimensionalized with respect to gross surface load. The elliptical span loading is also given for the wing. The output for each of the seven configurations follows this basic form.

The example of Figure 6 shows the form of output from Program 55VD when it is operated in mode 1. If the operation mode is 2, the unscaled and scaled mapping constants are not given on the first page of output. The subsequent pages of output describe the solution for each member of the series of configurations that are analyzed to arrive at the final configuration. With one exception, the output for each member of the series has the same form as the output when the program is operated in mode 1. The exception is the information on the first page of mode 1 output, which is given in abbreviated form only for the first member of the series.

For some cases, more iteration summary data may be given than in the example of Figure 6. If the convergence rate of the iteration is not sufficiently rapid, Program 55VD computes, prints, and uses a new correction matrix $\partial C_i/\partial p_j$. The format of each correction matrix is fixed. As many as four of these matrices may be computed and printed for a given case. Each matrix is printed just above the first line of iteration summary data that results from its use.

Error Messages. Program 55VD may print out one or more of several types of error messages. The input data is tested for validity by subroutine DATACK which prints appropriate error messages, and possibly terminates execution of a case. The subroutine terminates execution of the case if: the configuration number is less than 1 or greater than 7; the program operation mode parameter is neither 1 nor 2; the magnitude of the dihedral angle is greater than 90 degrees for configurations 5 and 6; the inboard span station is specified outboard of the outboard span station for configurations 2, 4, 6, or 7; or the input values of the mapping constants are not monotonically increasing as required by the theory. Subroutine DATACK also tests the input number of ordinates for each of the numerical integrals. If this number is less than 10, it is set to 10; if it is greater than 50, it is set to 50. If one of these events occurs, a message is printed to indicate the change.

The main program can print only one type of error message. If the successive solution scheme is being used to determine the mapping constants (program operation mode 2), execution of a case is terminated and a message is printed when the smallest value of $\bar{e}$ exceeds an arbitrarily specified value of 0.01 for any configuration in the series. Note that the final value of $\bar{e}$ is also the smallest. This test is included because the final values of the mapping constants are not very well known if $\bar{e}$ is large, and they will not likely serve adequately as initial values for the next member of the series of configurations.

Several tests are made in Program 55VD for division by zero and for overflow/underflow. If one of these abnormalities occurs, a message is printed that includes a location identification number. These tests are made for each case just before new input data is read in, just before and just after the iteration that determines the mapping constants, after the parameter k is calculated, and after the loading has been calculated. The identification numbers for these tests are 1, 2, 3, 4, and 5, respectively. Program execution continues after the message is written.

The numerical integration subroutines NUMINT and NUM each include provision for an error message if one of them is called with an invalid numerical integral number. This type of error occurs only with computing machine malfunction, and is therefore rarely seen. It does not terminate execution of a case.

Subroutines GEOM and DRAGI each include provision for an error message if the current values of the mapping constants are not monotonically increasing when the subroutines are called as required by the theory. This error is rarely encountered and does not terminate execution of a case.

Subroutine INVERT prints an error message if the correction matrix $\partial C_i/\partial p_j$ is singular. Execution of the case continues after the message is given.

Subroutine STORE prints an error message if the input calling parameter ICALL exceeds 10. This message is rarely seen, since a computing machine malfunction is necessary to produce it. Execution of the case continues after the message is given.

Subroutine INTERP prints an error message each time the interpolation for loading is unsuccessful or only partially successful. The message includes the values of several parameters from subroutine INTERP that aid in determining the reason for interpolation failure. This message is seen frequently if some of the auxiliary lifting surfaces have very small lengths or if $\bar{E}$ is not reduced to a small value by the iteration. In both cases, the loading is of little interest. Execution of the case continues after this message is given.

Subroutine ELOAD computes elliptical span loading as a function of nondimensional span location. If the specified location lies outside the semispan, it is printed as part of an error message. Execution is not halted by this type of error.

<u>Program OVERLAY Capability.</u> Computer Program 55VD does not require the use of OVERLAY on a CDC 6000 series computer or on a IBM 7094. Nevertheless, Program 55VD has been written to exploit the FORTRAN OVERLAY feature if it should become either necessary or desirable to do so. Figure 7 presents an OVERLAY structure for Program 55VD that saves considerable core storage, yet does not require excessive transfer of large subroutines into core storage.

Accuracy of Program 55VD Output

The accuracy of Program 55VD output can be expressed in terms of errors in the required geometry conditions C_i, errors in the value of k, and errors in the loading required to produce minimum induced drag. These errors arise from two sources. First, the integrals for induced drag and for distances in the physical plane are evaluated numerically with a modified Simpson rule subroutine that uses internal double precision arithmetic to improve accuracy. Nevertheless, Simpson rule quadrature provides exact results only if the function being integrated is a polynomial of third degree or less. The numerical integrals evaluated by Program 55VD have smooth integrands, but the integrands are not polynomials of third degree or less; thus the numerical integration introduces a small error into the integrals evaluated by Program 55VD.

The second source of error results from the arithmetic impossibility of reducing the ΔC_i to zero by iteration. Program 55VD nominally continues the iteration to determine the mapping constants until $\bar{e} \leq 3\times10^{-8}$. This limiting value of $\bar{e}$ can be compared with the semispans of the seven configurations which are of the order of 1 for typical cases. The mapping constants usually vary only in the eighth significant digit as $\bar{e} \rightarrow 3\times10^{-8}$ from above, providing some indication that they are known to the order of 10^{-8} if the effects of the first source of error are not considered. Both k and the required loading are proportional to the square root of quantities linear in the mapping constants. Errors in k and in the required loading should therefore be of the order of 10^{-4}, again without considering the effects of the first type of error.

A series of cases were analyzed with Program 55VD for Configurations 1-7 so that estimates could be made of the numerical errors in k and in the loading required to produce minimum induced drag. Ideally, the test geometry for each configuration would have been the degenerate case of the monoplane, for which $k = 1$ and the required loading is elliptical with span. However, one of the limitations of Program 55VD is the considerable difficulty in achieving a converging iteration when any of the lengths in the real plane are zero. This problem is discussed further in the Limitations section. The ideal monoplane geometry was approximated by setting each pylon or fence length to 10^{-5} semispans, which introduces an error in k of the order of 10^{-5}. Figure 8 presents k as a function of the number of integration ordinates in each of the numerical integrals. For each configuration, the value of k is brought nearer 1 by increasing the number of integration ordinates from 10 to 50. With 50 integration ordinates, the error in k is of the order of 10^{-4} or smaller for each of the seven configurations, although the errors in k for Configurations 3 and 7 are significantly larger than for the other configurations for reasons unknown.

Table II presents a comparison of elliptical span loading with the loading determined by Program 55VD for Configuration 1 as a near-monoplane. The maximum loading error is of the order of 10^{-4} with the exception of a single point near the wing tip. These results for Configuration 1 are typical of results obtained for the other configurations, including the loss of accuracy near the wing tip.

Table II

Comparison of Elliptical Span Loading With Program 55VD Loading for a Near-Monoplane

Configuration 1

$\eta = 0.60$ $\quad \ell = 10^{-5}$ $\quad$ 50 Integration Ordinates

Span Station Nondimensional	Program 55VD Loading	Elliptical Span Loading
0	1.273 2480	1.273 2395
0.10162769	1.266 6551	1.266 6473
0.20560790	1.246 0360	1.246 0360
0.30235275	1.213 6385	1.213 6469
0.40073262	1.166 5275	1.166 5356
0.50122220	1.101 7481	1.101 7578
0.60000000	1.018 5892	1.018 5916
0.71114510	0.895 1501	0.895 1450
0.80591056	0.753 8091	0.753 8053
0.90484013	0.541 7942	0.541 7976
0.98273646	0.236 5653	0.235 5633
0.99157844	0.164 8271	0.164 8941
0.99845841	0.070 3613	0.070 6689
0.99999999	0.000 0029	0.000 0000

Limitations of Program 55VD

In a very real sense, each case submitted to Program 55VD involves a numerical experiment in the convergence of the iteration for the unknown mapping constants. Convergence of the iteration is defined by the reduction of $\bar{e}$ below 3×10^{-8}. Because the correction matrix $\partial C_i/\partial p_j$ depends on the initial values of the mapping constants and is calculated numerically at least once for each case, the standard methods of proving convergence, some of which are discussed in Reference 13, cannot be applied. Therefore, it is not known, a priori, that the iteration will converge for any given case. In practice, the great majority of cases do have iterations that converge. When the electrostatic analog is used to obtain initial estimates of the mapping constants (program operation mode = 1), convergence has been obtained for nearly every case that has been run. When the successive solution scheme (program operation mode = 2) is used to determine the mapping constants, the iteration fails

frequently for cases having one or both of two geometric characteristics. The first of these characteristics applies only to Configurations 1, 2, 4, 5, and 6, and simply involves placing a vertical fence near the configuration centerline (η small). The crossflow velocities are small in this region, so that potential differences between ends of the fence so located are also small. Convergence of the iteration for the mapping constants is generally weakened in such cases — frequently to the point of failure.

The second geometric characteristic for which the iteration frequently fails when the program operation mode is 2 is found in each of the seven configurations in varying degree and consists simply of specifying one or more lengths in the real plane to be zero or near-zero. For Configuration 5, for example, specifying $\eta = 0.99$ with $\Gamma = 0$ requires both $\overline{DE}$ and $\overline{EF}$ to be small — 0.01s. Effectively, such a requirement also specifies the potential differences between d and e, and between e and f to be small, which was also noted for the first geometric characteristic. Experience with Configuration 1 suggests that this limitation is more pronounced for lengths oriented parallel to the spanwise direction than for lengths oriented normal to it. For example, if $\eta = 0.5$ for Configuration 1, the iteration for the mapping constants converges for ℓ as small as 10^{-5} (lengths $\overline{BC}$ and $\overline{CD} \rightarrow 0$). However, if $\ell = 0.04$, convergence is not obtained for $0.98 \lesssim \eta \lesssim 1.0$ (length $\overline{DE} \rightarrow 0$). This limitation of the iteration is believed to be basic in character, rather than numerical. When any length of one of the configurations becomes zero, a different configuration is formed that has fewer Schwarz-Christoffel mapping constants to be determined. For example, when length $\overline{DE}$ of Configuration 6 becomes zero, the resulting shape is identical to Configuration 5. Similarly, if the length of one of the vertical fences of either Configuration 2 or Configuration 4 is set to zero, the resulting shape is identical to Configuration 1. For Configurations 1, 2, 4, 5, and 6, the critical value of the fence spanwise coordinate η lies between 0.95 and 0.99 while the critical value of the fence length parameter is of the order of 10^{-5} to 10^{-7}; the iteration for the mapping constants frequently does not reduce $\bar{e}$ to 3×10^{-8} if η lies between 1 and the critical value, or if ℓ is smaller than the critical value. For Configuration 3, convergence has not been achieved for $s_H > 0.12$ if $\ell_H \leq 0.1$. For Configuration 7, convergence has been achieved only with considerably increasing difficulty as fence and end plate lengths are decreased; the smallest values of the geometry parameters for which convergence has been obtained with program operation mode = 2 are $\ell_i = \ell_o = 0.01$, $\ell = 0.02$. For this reason, the initial values of the mapping constants for Configuration 7 are not those of a monoplane, but rather for $\ell_i = \ell_o = 0.01$, $\ell = 0.04$.

These failures to obtain convergence of the iteration for the mapping constants have been observed principally when Program 55VD uses the successive solution scheme to determine the mapping constants.

CONCLUDING REMARKS

The introduction of Reference 8 points out that nonplanar lifting surfaces producing minimum induced drag can be analyzed in three steps:

(1) For a given configuration, determine the loading to minimize induced drag with a constraint on lift.

(2) Given (1), compute the minimum induced drag.

(3) Given (1), compute the wing geometry (camber and/or twist) to produce the minimum induced drag loading.

Computer Program 55VD calculates (1) and (2) for the seven configurations of Figure 2. Complimentary methods for treating (3) must be developed. Reference 14 outlines a nonplanar theory that can likely be adapted for this purpose.

```
000351          IF (POREQ .GT. POUT) POREQ = POUT                                55VD2890
000355          CALL SERIES                                                      55VD2900
000356          IF (LAST) GO TO 2                                                55VD2910
000360          IF (R1 .GT. 0.01) GO TO 3                                        55VD2920
000363          GO TO 402                                                        55VD2930
000363      500 CONTINUE                                                         55VD2940
          C                                                                      55VD2950
          C     WING WITH PYLON AND DIHEDRAL OUTBOARD OF PYLON                   55VD2960
          C                                                                      55VD2970
000363          A = 0.0                                                          55VD2980
000364          DUM1 = ETAIRQ                                                    55VD2990
000366          IF (DUM1 .GT. 0.98) DUM1 = 0.98                                  55VD3000
000371          D = 1.0 - SQRT(1.0 - DUM1**2)                                    55VD3010
000377          C = D - 0.01                                                     55VD3020
000401          B = C - 0.01                                                     55VD3030
000403          E = 1.0                                                          55VD3040
000404          F = 2.0 - D                                                      55VD3050
000406          G = 2.0                                                          55VD3060
000407          PIN = PIREQ                                                      55VD3070
000411          DIH = GAMMA                                                      55VD3080
000412          PIREQ = 0.0                                                      55VD3090
000413          GAMMA = 0.0                                                      55VD3100
000413      502 PIREQ = PIREQ + 0.03                                             55VD3110
000415          GAMMA = GAMMA + 3.0                                              55VD3120
000417          IF (PIREQ .LT. PIN .OR. GAMMA .LT. DIH) GO TO 503                55VD3130
000430          LAST = .TRUE.                                                    55VD3140
000431      503 IF (PIREQ .GT. PIN) PIREQ = PIN                                  55VD3150
000435          IF (GAMMA .GT. DIH) GAMMA = DIH                                  55VD3160
000441          CALL SERIES                                                      55VD3170
000442          IF (LAST) GO TO 2                                                55VD3180
000444          IF (R1 .GT. 0.01) GO TO 3                                        55VD3190
000447          GO TO 502                                                        55VD3200
000447      600 CONTINUE                                                         55VD3210
          C                                                                      55VD3220
          C     WING WITH INBOARD PYLON AND DIHEDRAL BREAK OUTBOARD OF PYLON     55VD3230
          C                                                                      55VD3240
000447          A = 0.0                                                          55VD3250
000450          DUM1 = ETAORQ                                                    55VD3260
000452          IF (DUM1 .GT. 0.98) DUM1 = 0.98                                  55VD3270
000455          E = 1.0 - SQRT(1.0 - DUM1**2)                                    55VD3280
000463          DUM2 = ETAIRQ                                                    55VD3290
000465          IF ((DUM2 < 0.02) .GT. DUM1) DUM2 = DUM1 - 0.02                  55VD3300
000472          D = 1.0 - SQRT(1.0 - DUM2**2)                                    55VD3310
000500          C = D - 0.01                                                     55VD3320
000502          B = C - 0.01                                                     55VD3330
000504          F = 1.0                                                          55VD3340
000505          G = 2.0 - E                                                      55VD3350
000507          H = 2.0                                                          55VD3360
000510          PIN = PIREQ                                                      55VD3370
000512          DIH = GAMMA                                                      55VD3380
000513          PIREQ = 0.0                                                      55VD3390
000514          GAMMA = 0.0                                                      55VD3400
000514      602 PIREQ = PIREQ + 0.03                                             55VD3410
000516          GAMMA = GAMMA + 3.0                                              55VD3420
000520          IF (PIREQ .LT. PIN .OR. GAMMA .LT. DIH) GO TO 603                55VD3430
000531          LAST = .TRUE.                                                    55VD3440
000532      603 IF (PIREQ .GT. PIN) PIREQ = PIN                                  55VD3450
000536          IF (GAMMA .GT. DIH) GAMMA = DIH                                  55VD3460
```

REFERENCES

1. Munk, Max M.: The Minimum Induced Drag of Aerofoils, NACA Report 121, 1921.

2. Mangler, W.: The Lift Distribution of Wings with End Plates, NACA TM No. 856, 1938 (Translated by J. Vanier from: Die Auftriebsverteilung am Traglulegel mit Endscheiben, Luftfahrtforschung, V. 14, No. 11, Nov. 20, 1937, pp 564-569).

3. Manger, W.: Lift Distribution on Wings with Lateral Plates, Douglas Aircraft Company Report No. LB-33497, Dec., 1966 (Translated by G. deMontalvo from Luftfahrtforschung, V. 16, 1939, pp 219-228).

4. Falkner, V. M., and Darwin, C.: The Design of Minimum Drag Tip Fins, A.R.C. R&M 2279, 1945.

5. Kuchemann, D., and Kettle, D. J.: The Effect of End Plates on Swept Wings, A.R.C. CP104, June, 1951.

6. Weber, J.: Theoretical Load Distribution on a Wing with Vertical Plates, A.R.C. R&M 2960, March, 1954.

7. Cone, C. D., Jr.: The Theory of Induced Lift and Minimum Induced Drag of Nonplanar Lifting Systems, NASA Technical Report R-139, 1962.

8. Lundry, J. L., and Lissaman, P.B.S.: A Numerical Solution for the Minimum Induced Drag of Nonplanar Wings, AIAA J. of Aircraft, V.5, No. 1, Jan-Feb, 1968.

9. Douglas Aircraft Company, Inc. Technical Proposal 3388T (Unsolicited), Load Distribution to Produce Minimum Induced Drag for Nonplanar Wings, March, 1967.

10. Lundry, J. L.: A Numerical Solution for the Minimum Induced Drag, and the Corresponding Loading, of Nonplanar Wings — Descriptive Manual for Computer Program 55VD, McDonnell-Douglas Corporation, Report No. DAD-63699, February 29, 1968.

11. Lundry, J. L.: A Numerical Solution for the Minimum Induced Drag, and the Corresponding Loading, of Nonplanar Wings — Final Report, McDonnell-Douglas Corporation, Report DAD-66900, March 29, 1968.

12. von Karman, Th. and Burgers, J. M.: General Aerodynamic Theory — Perfect Fluids, Division E, Vol. II, p. 138 of Aerodynamics Theory, W. F. Durand, Editor-in-Chief, Julius Springer 1934-36 and Dover Publications.

13. James, R. M.: A Review of Numerical Techniques Suitable for the Computation of General Inviscid Transonic Flows, Douglas Aircraft Company Report No. DAC-33978, December, 1967.

14. Ashley, H., and Landahl, M.: Aerodynamics of Wings and Bodies, Addison-Wesley, 1965, pp 208-220.

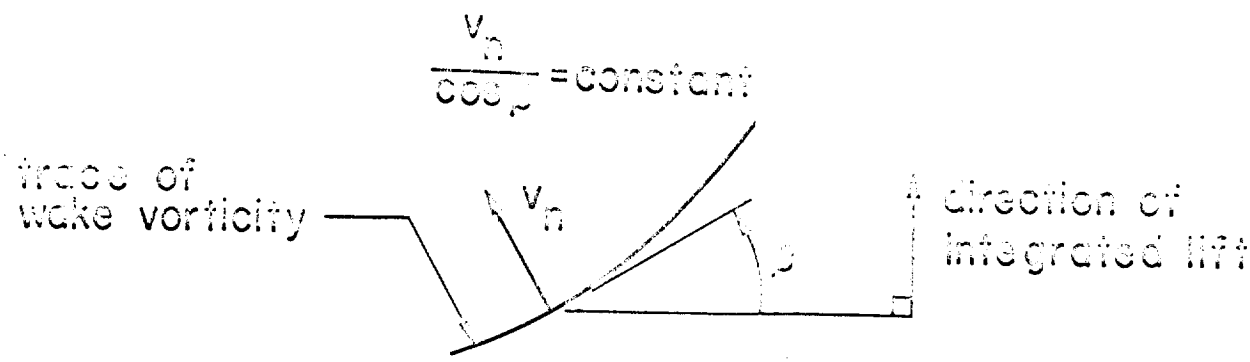
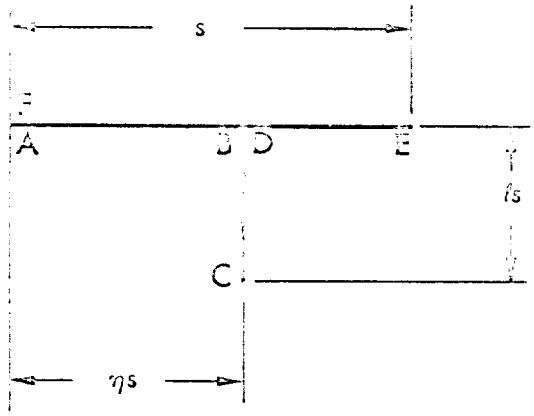
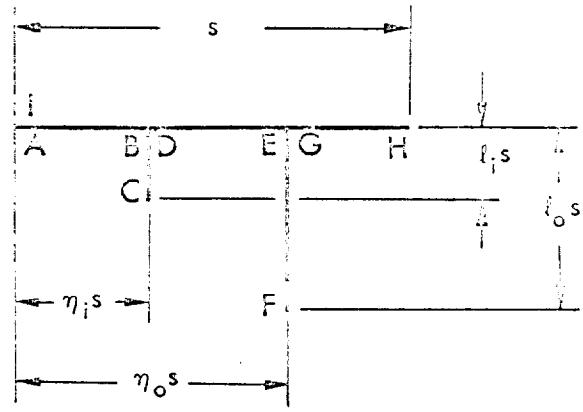


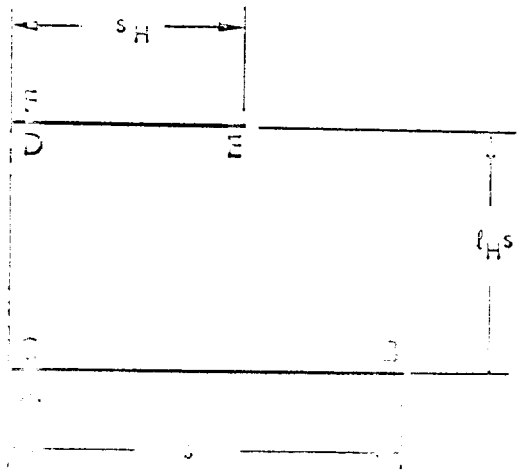
Figure 1. Munk's Minimum Induced Drag Criterion.



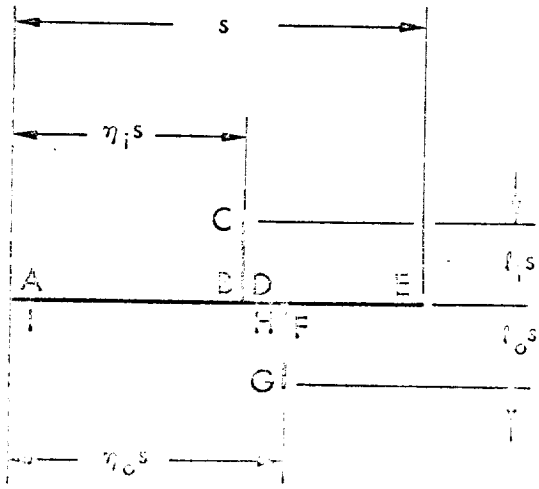
CONFIGURATION 1



CONFIGURATION 2



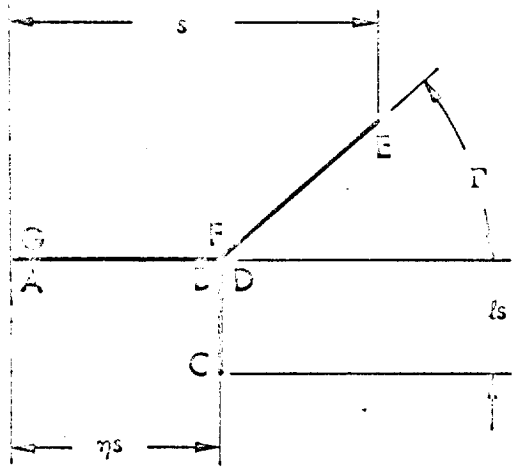
CONFIGURATION 3



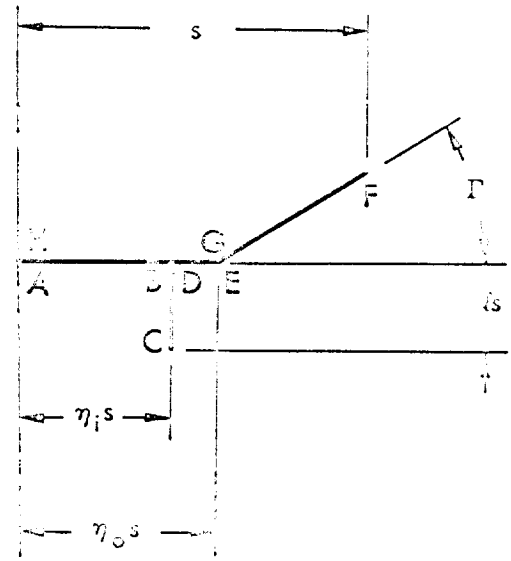
CONFIGURATION 4

Figure 2a. Configurations 1 - 4.

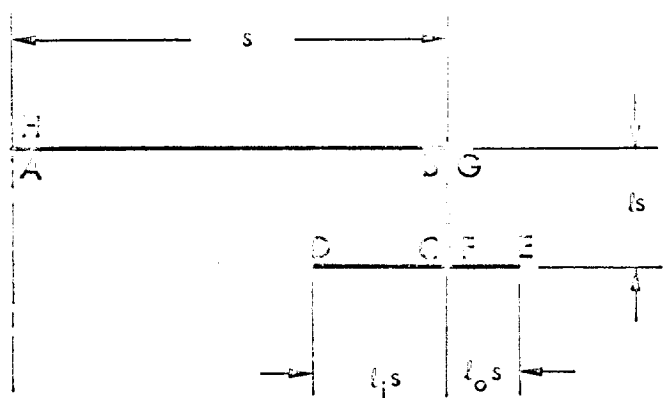
Figure 2. Configurations Analyzed by Program 55VD.



CONFIGURATION 5



CONFIGURATION 6



CONFIGURATION 7

Figure 2b. Configurations 5 - 7.

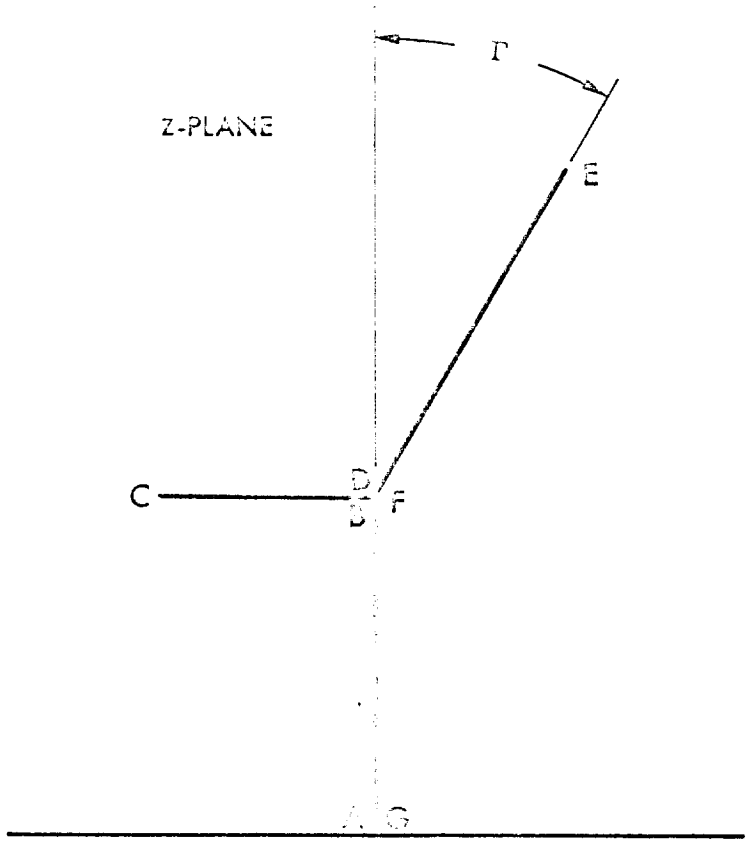


Figure 3a. Real Plane.

$$\frac{dZ}{d\zeta} = \frac{(\zeta-c)(\zeta-e)}{[(\zeta-a)(\zeta-b)(\zeta-g)]^{1/2}} (\zeta-d)^{\frac{\Gamma}{\pi}-\frac{1}{2}} (\zeta-f)^{-\frac{\Gamma}{\pi}}$$

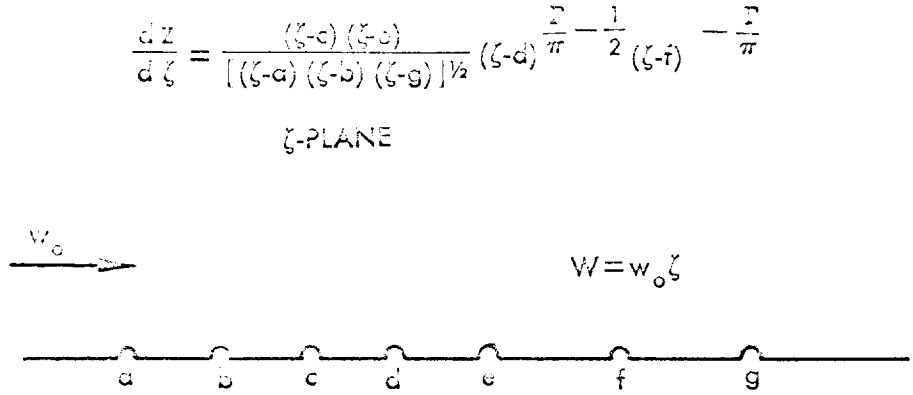


Figure 3b. Auxiliary Plane.

Figure 3. Mapping Planes for Configuration 5.

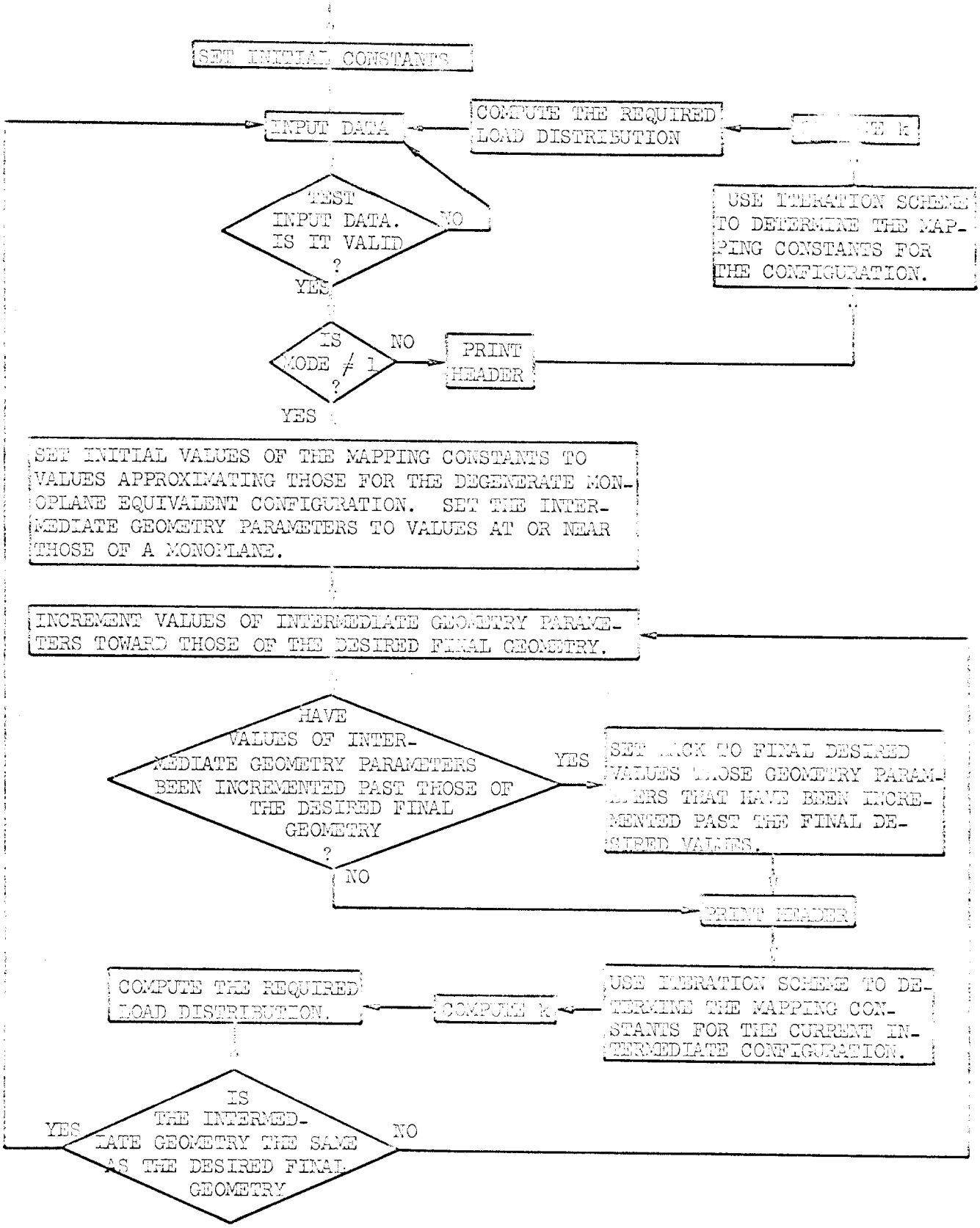
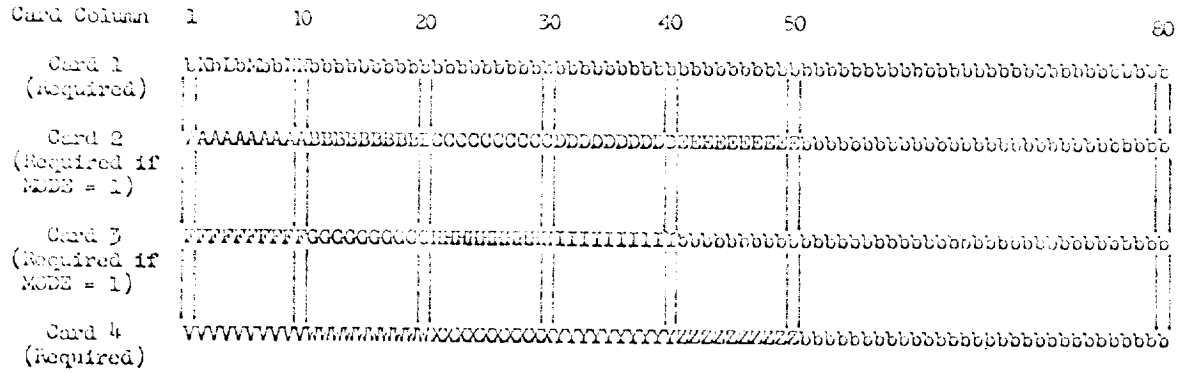


Figure 4. Overall Logical Flow of Computer Program 55VD.



K-N are integers; A-I, V-Z are floating point (real) numbers.

K Flag = 1 for the last case to be executed; Flag ≠ 1 for other cases.
L Program operation MODE number (values 1 or 2).
M Configuration number (values range from 1 to 7; see Figure 2).
N Number of numerical integration ordinates (values range from 10 to 50).

A Initial estimate of the potential at A (mapping constant).
B Initial estimate of the potential at B (mapping constant).
C Initial estimate of the potential at C (mapping constant).
D Initial estimate of the potential at D (mapping constant).
E Initial estimate of the potential at E (mapping constant).
F Initial estimate of the potential at F (mapping constant).
G Initial estimate of the potential at G (mapping constant).
H Initial estimate of the potential at H (mapping constant).
I Initial estimate of the potential at I (mapping constant).

V Geometric parameter; see Table I.
W Geometric parameter; see Table I.
X Geometric parameter; see Table I.
Y Geometric parameter; see Table I.
Z Geometric parameter; see Table I.

b Unused card column; leave these columns unpunched, except for cc. 51-80, which may be used for deck identification and sequencing.

Figure 5. Data Input Cards.

```
* * * * * * * *    NEW CASE DATA LOADED BY SUBROUTINE INPUT    * * * * * * *

                    CONFIGURATION   1

              PROGRAM OPERATION MODE NUMBER =   1

             NUMBER OF INTEGRATION STEPS =      15

INPUT POTENTIALS (CONSTANTS OF THE SCHWARZ-CHRISTOFFEL TRANSFORMATION)

                    A =   173.89999962
                    B =   196.50000000
                    C =   216.79999924
                    D =   242.50000000
                    E =   497.90000153
                    F =   823.00000000

          TRANSFORMED POTENTIALS (LINEARLY SCALED)

                    A =     0.00000000
                    B =     0.06975309
                    C =     0.13240740
                    D =     0.21172840
                    E =     1.00000000
                    F =     2.00339505
```

Figure 6. Program Output for Configuration 1 (Page 1).

WING WITH ONE PYLON (CONFIGURATION 1)

TRANSFORMATION CONSTANTS (INDEPENDENT VARIABLES)						ERRORS IN REQUIRED CONDITIONS				
A	B	C	D	E	F	PYLON CLOSURE	WING CLOSURE	PYLON LOCATION	PYLON LENGTH	SQUARE ROOT OF SUM OF SQR. ERRORS
0.0	0.06975300	0.13240740	0.21172840	1.0	2.00339505	-0.02015708	-0.03104809	0.00019069	0.00097906	0.03703059
-0.547	-1.593	-1.735	0.632							
4.148	-1.593	-1.946	-1.540							
1.987	-1.611	-1.091	0.567							
0.014	-1.547	-0.377	-0.097							
0.0	0.07313916	0.13229432	0.21466930	1.0	1.97653724	0.00016076	0.00099998	-0.00019194	0.00030635	0.00107541
-0.0	0.07335003	0.13250471	0.21473233	1.0	1.97734252	-0.00005205	-0.00005756	0.00001157	0.00000015	0.00007846
0.0	0.07335602	0.13250124	0.21473835	1.0	1.97728938	0.00000004	0.00000334	-0.00000085	0.00000130	0.00000380
-0.0	0.07335600	0.13250196	0.21473830	1.0	1.97728933	-0.00000021	-0.00000015	0.00000005	-0.00000000	0.00000028
0.0	0.07335602	0.13250196	0.21473833	1.0	1.97728913	-0.00000000	0.00000003	0.00000000	0.00000001	0.00000003
-0.0	0.07335602	0.13250197	0.21473833	1.0	1.97728914	-0.00000000	0.00000001	0.00000000	0.00000000	0.00000001
-0.0	0.07335602	0.13250197	0.21473833	1.0	1.97728914	-0.00000000	0.00000001	0.00000000	0.00000000	0.00000001

POTENTIALS IN EXPONENTIAL FORMAT

-0.00000000E-38 0.73356019E-01 0.13250197E 00 0.21473833E 00 0.10000000E 01 0.19772891E 01

CLOSURE INTEGRAL (PYLON) = -0.00000000
CLOSURE INTEGRAL (WING) = 0.00000001

PYLON SEMISPAN LOCATION = 0.50000000
PYLON LENGTH (SEMISPANS) = 0.125[illegible]0000

INDUCED DRAG INTEGRAL = 1.51209867

DISTANCE FROM A TO B = 0.48938697
DISTANCE FROM B TO C = -0.12234674
DISTANCE FROM C TO D = 0.12234674
DISTANCE FROM D TO E = 0.48938697
DISTANCE FROM E TO F = -0.97877393

LIFTING EFFICIENCY K = 1.00483663

Figure 6. Program Output for Configuration 1 (Page 2).

WING WITH ONE PYLON

RATIO OF LOADING ON PYLON TO LOAD ON WING = 0.00895395

WING SPANWISE COORDINATE	WING LOADING	ELLIPTICAL WING LOADING	PYLON COORDINATE	PYLON LOADING
0.00000000	1.27990375	1.27323949	0.00000000	1.27760035
0.00785242	1.27986737	1.27320023	0.04142986	1.27647349
0.01394202	1.27978893	1.27311572	0.07350609	1.27404992
0.02194368	1.27961907	1.27293290	0.11562882	1.26881345
0.03325770	1.27924955	1.27253515	0.17455553	1.25745764
0.04970680	1.27844205	1.27166557	0.25913838	1.23260236
0.07387982	1.27667461	1.26975991	0.37952775	1.17956047
0.10935263	1.27282895	1.26560390	0.54406353	1.08715430
0.15064506	1.26647694	1.25870916	0.71099733	0.99153217
0.18307467	1.26007468	1.25172047	0.81913509	0.72540996
0.21082138	1.25361966	1.24462292	0.89321870	0.56721438
0.23559976	1.24711202	1.23739801	0.94373438	0.41565274
0.25832753	1.24050489	1.23002242	0.97630009	0.26908094
0.27957898	1.23380792	1.22246613	0.99433713	0.12087613
0.29975913	1.22700888	1.21468946	**0.99999996**	**0.00000029**
0.29975913	1.22700888	1.21468946		
0.31918636	1.22008543	1.20663864		
0.33814248	1.21300463	1.19823922		
0.35691558	1.20567569	1.18937965		
0.37584868	1.19797234	1.17988691		
0.39543150	1.18980628	1.16946469		
0.41652872	1.18089484	1.15753074		
0.44121915	1.17051573	1.14260434		
0.46101549	1.16240960	1.12986263		
0.47399409	1.15727331	1.12112505		
0.48266789	1.15393926	1.11510754		
0.48858459	1.15171491	1.11092250		
0.49275314	1.15017313	1.10793371		
0.49592014	1.14901607	1.10564059		
0.50000000	1.14754303	1.10265774		
0.50000000	1.05602725	1.10265774		
0.51352063	1.05062258	1.09253784		
0.52406169	1.04588896	1.08439299		
0.53804071	1.03831708	1.07323757		
0.55818928	1.02610122	1.05642417		
0.58857374	1.00441071	1.02934223		
0.63600799	0.96364269	0.98253659		
0.71090107	0.88226352	0.89545917		
0.80067620	0.75399248	0.76279436		
0.86727338	0.62749662	0.63385901		
0.91755182	0.50077473	0.50620069		
0.95459890	0.37420251	0.37929065		
0.98011190	0.24714295	0.25266879		
0.99507128	0.11585662	0.12625733		
0.99999996	0.00000035	0.00000000		

Figure 6. Program Output for Configuration 1 (Page 3).

Deck name is followed by deck identification in parenthesis.

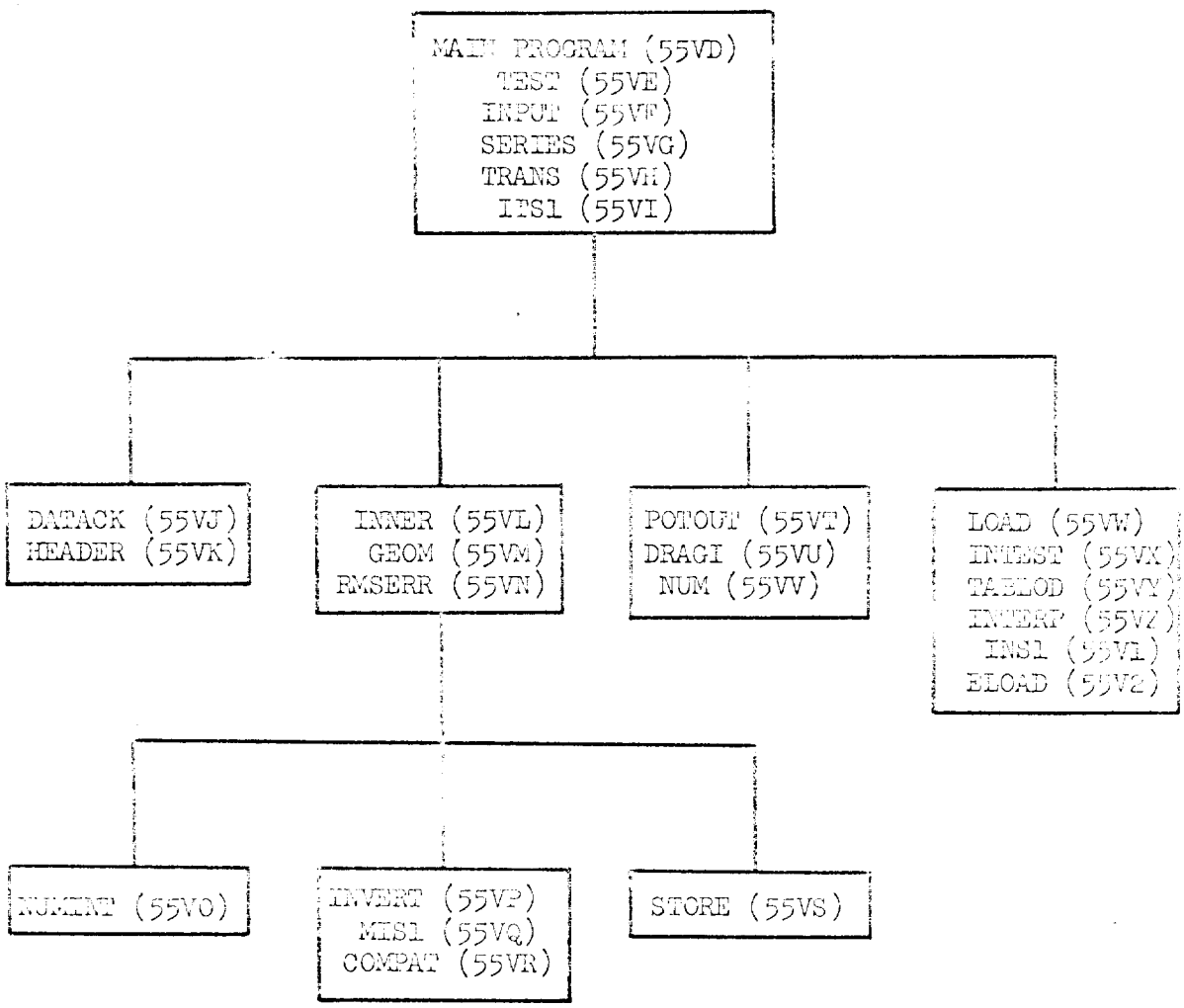


Figure 7. OVERLAY Structure for Program 55VD.

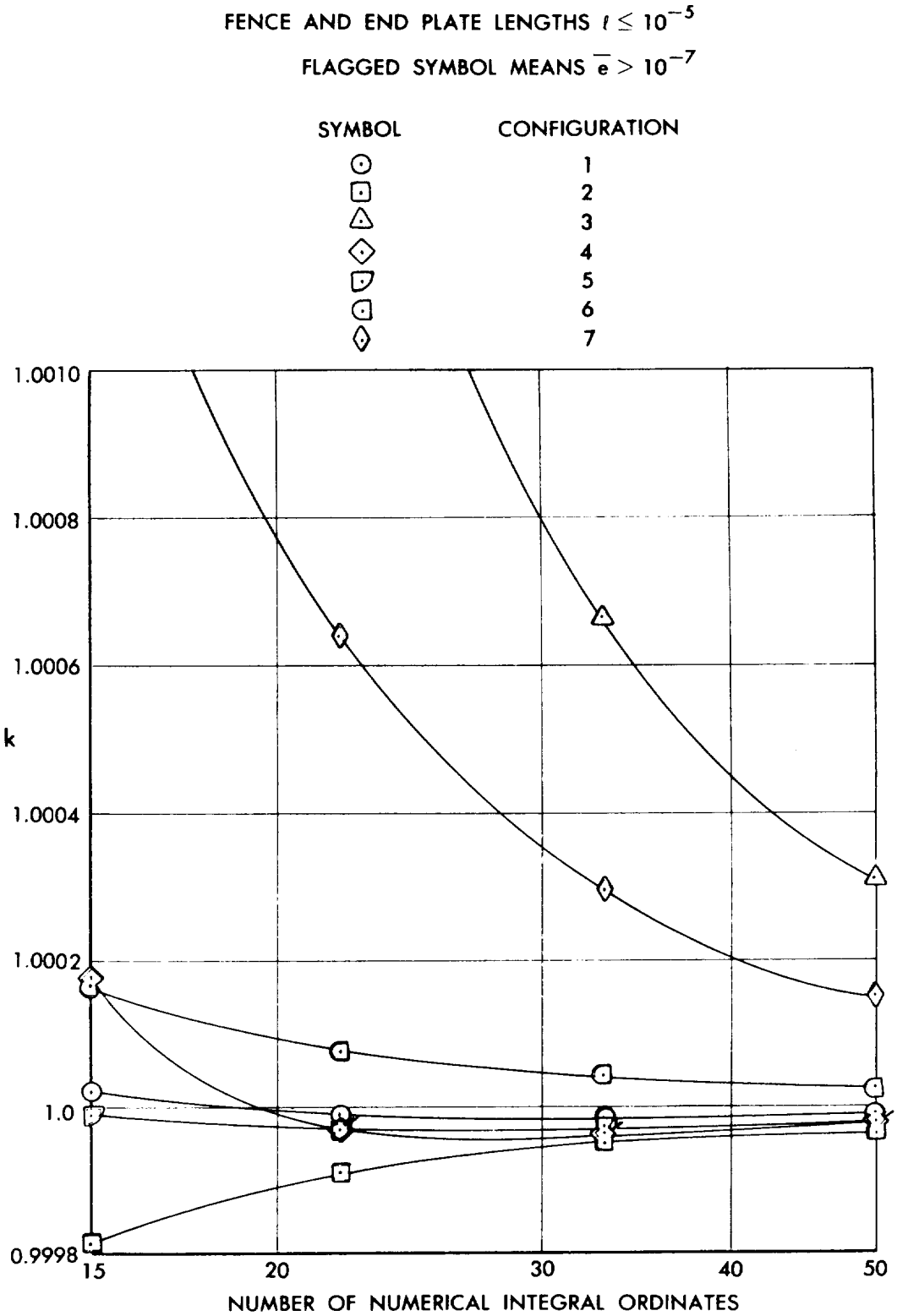


Figure 8. Induced Drag Efficiency at Minimum Induced Drag for Near-Monoplane Degenerate Configurations.

A NUMERICAL SOLUTION FOR THE MINIMUM INDUCED DRAG, AND THE CORRESPONDING LOADING, OF NONPLANAR WINGS

II - DESCRIPTIVE MANUAL FOR COMPUTER PROGRAM 55VD

Issued by Originator as Report DAC-66899

CONTENTS

	Page Number
SUMMARY	1
INTRODUCTION	2
Munk's Theory	2
Application of Munk's Theory	2
SYMBOLS	3
ANALYSIS	4
The Mapping Constants	4
Electrostatic analogy with iterative correction	4
Successive solution scheme	6
The Minimum Induced Drag	7
The Loading to Produce Minimum Induced Drag	7
COMPUTER PROGRAM 55VD	7
Input/Output Files	8
Detailed Description of Subroutines	8
Test Cases	283
Program OVERLAY Structure	378
REFERENCES	379
FIGURES	381

A NUMERICAL SOLUTION FOR THE MINIMUM INDUCED DRAG, AND THE CORRESPONDING LOADING, OF NONPLANAR WINGS — DESCRIPTIVE MANUAL FOR COMPUTER PROGRAM 55VD

by J. L. Lundry
The McDonnell Douglas Corporation

SUMMARY

A numerical procedure has been developed for the accurate computation of the minimum induced drag, and the associated load distribution, of nonplanar wings. As is well known, the minimum induced drag may be determined by analyzing a potential problem in the Trefftz plane. The procedure solves the potential problem in an auxiliary mapping plane related to the physical plane by the Schwarz-Christoffel transformation for configurations with front views that may be approximated by straight line segments. Previously, the main computational difficulty was the determination of the mapping constants. The procedure presented here offers a choice of two methods to determine these constants. By use of the electrostatic analogy to potential flow, the mapping constants are measured with an analog field plotter. The measured values contain small experimental errors. The mapping derivative is integrated numerically, and the constants are adjusted iteratively until the desired geometry is obtained to high order of accuracy.

Alternatively, the mapping constants can be determined without resorting to analog experimentation by evaluating them for a series of geometrically related configurations that ends with the desired configuration. The mapping constants for each member of the series are used as the equivalent of experimental values for the next member of the series whose mapping constants in turn can then be determined with the iteration scheme employed in the experimental method. The series starts with the monoplane degenerate equivalent of the desired configuration, for which the mapping constants are known. This procedure is successful if each member of the series differs geometrically from its neighbors by a small amount.

Once the mapping constants are known, the minimum induced drag and the associated loading are determined by quadrature.

The procedure has been applied to seven nonplanar lifting configurations, resulting in the development of Computer Program 55VD in the FORTRAN IV language for use on an IBM 7094. Program 55VD has been converted to FORTRAN 2.0 for use on the Langley Research Center's CDC 6000 series digital computers. This report describes in detail the FORTRAN 2.0 version of Program 55VD.

INTRODUCTION

Munk's Theory

In Reference 1, Munk developed a theory for the minimum induced drag, and the associated loading, of arbitrary lifting configurations. All loadings are assumed light, so that velocity perturbations are small and the vortex wake in the Trefftz plane may be assumed undistorted. The loadings can be projected onto a plane normal to the free-stream velocity without changing the induced drag of the lifting system (Stagger Theorem). Munk's criterion for minimum induced drag is illustrated in Figure 1, and requires the induced velocity normal to the projected loadings to be proportional to the cosine of the angle of lateral inclination of the projected loadings. Munk further demonstrates that the loading to satisfy this criterion can be found by solving a potential flow problem about the wake in the Trefftz plane, in which the undisturbed flow is parallel to the downwash. The required loading is locally proportional to the potential difference across the wake and is normal to the wake.

Applications of Munk's Theory

Munk applied his theory to the monoplane, and obtained the classic result that a constant downwash across the span produces the minimum induced drag

$$D = \frac{L^2}{4 \pi q s^2} \tag{1}$$

and is given by an elliptical distribution of load. In References 2-6, the theory is applied analytically to nonplanar configurations consisting either of combinations of a monoplane with vertical fences or a wing with part-span or full-span dihedral. In Reference 7, the electrostatic analogy to potential flow is exploited to determine experimentally loadings that satisfy Munk's criterion for complex nonplanar lifting configurations; the lift and the minimum induced drag are then evaluated numerically. Unfortunately, the numerical results of this method can contain significant errors, as shown by Reference 8. Reference 8 solves the potential problem in an auxiliary mapping plane related to the real (Trefftz) plane by the Schwarz-Christoffel transformation for configurations with front views that may be approximated by straight line segments. Previously, the main computational difficulty with this approach was the determination of the mapping constants. By means of the electrostatic analogy to potential flow, the mapping constants are measured with an analog field plotter. The measured values of the mapping constants include small experimental errors. The transformation derivative is integrated numerically, and the constants are adjusted iteratively until the desired geometry is obtained to high order of accuracy. Once the mapping constants are known, the minimum induced drag is determined by quadrature. A digital computer program was required to obtain the numerical results of Reference 8 for Configurations 1-5 (See Figure 2) of this report.

Reference 9 proposes extensions to Reference 8 and the digital computer program — principally:

1) Application of the numerical scheme of Reference 8 to Configurations 6 and 7 of this report.

2) Calculation of the loading to produce minimum induced drag for Configurations 1-7.

3) Development of an alternative method to determine the mapping constants so that the analog experiments could be eliminated from the method of Reference 8.

The proposed extensions to Reference 8 and to the digital computer program were funded under NASA Contract NAS1-7484. This report describes in detail the extended digital computer program. Reference 10 is a program user's manual. Reference 11 summarizes all work sponsored by the contract.

SYMBOLS

A,B,C,D, E, F, G	potentials in the real or physical plane (See Figure 3)
C_i	the i-th required geometry condition in the real plane
ΔC_i	error in C_i
D	minimum induced drag
I	the imaginary part of
L	lift
a,b,c,d, e,f,g	potentials in the auxiliary plane (See Figure 3)
$\bar{e}$	the square root of the sum of the squares of the errors in the required geometry conditions: $\bar{e} = \sqrt{\sum_{i=1}^{n} (\Delta C_i)^2}$
$\bar{f}_i$	abscissa spacing parameter (See Equations (10) and (11))
k	induced drag efficiency at minimum induced drag
ℓ	nondimensional length of fence or pylon
n	number of required geometry conditions C_i

p_j	the j-th mapping constant or potential (The potentials are numbered in alphabetical order.)
Δp_j	correction to p_j
q	dynamic pressure
s	semispan
z	complex variable in the real plane
Γ	dihedral angle in degrees
ζ	complex variable in the auxiliary plane
η	nondimensional semispan coordinate

ANALYSIS

The basic theory for the minimum induced drag of nonplanar lifting configurations is given in Reference 1 and is described in the INTRODUCTION. A potential problem is formulated about the vortex wake in the Trefftz plane. By the method of Reference 8, the problem is analyzed numerically in an auxiliary mapping plane by means of the Schwarz-Christoffel transformation. First, the constants of the Schwarz-Christoffel mapping must be determined. The minimum induced drag can then be obtained by quadrature in the form k. Finally, the required load distribution can be obtained by interpolation. The seven general configurations sketched in Figure 2 are analyzed by Computer Program 55VD.

The Mapping Constants

Two methods to determine the mapping constants have been included in Program 55VD and are described here.

Electrostatic analog with iterative correction. This method is developed in Reference 8. By means of the electrostatic analogy to potential flow, the mapping constants can be measured experimentally with an analog field plotter, such as the one used by the author of Reference 7. The measured values of potential contain experimental errors, and are corrected by an iterative scheme that will be explained for Configuration 5. Figure 3 shows the Trefftz plane (also called real or physical plane) and the auxiliary mapping plane for Configuration 5. The Schwarz-Christoffel transformation between these planes is

$$\frac{dz}{d\zeta} = \frac{(\zeta - c)(\zeta - e)}{[(\zeta - a)(\zeta - b)(\zeta - g)]^{1/2}} (\zeta - d)^{\frac{\Gamma}{\pi} - 1/2} (\zeta - f)^{-\frac{\Gamma}{\pi}} \qquad (2)$$

Given the experimental values of potential a, . . . , g, the geometry in the physical plane is evaluated by numerically integrating Equation (2). For example, the length $\overline{AB}$ is

$$\overline{AB} = z_B - z_A = \int_a^b \frac{dz}{d\zeta}\, d\zeta \tag{3}$$

Each of the distances $\overline{AB}$, $\overline{BC}$, $\overline{CD}$, $\overline{DE}$, $\overline{EF}$, and $\overline{FG}$ can be evaluated in this way, and each of the integrals has either one or two integrable singularities at the end or ends of the integration interval. If an integral has two singularities, it is evaluated in two parts of equal range so that each numerical integral has at most one singularity. For example, the integral for the length $\overline{AB}$ has singularities at a and b, and is evaluated as

$$\overline{AB} = \int_a^{\frac{1}{2}(a+b)} \frac{dz}{d\zeta}\, d\zeta + \int_{\frac{1}{2}(a+b)}^{b} \frac{dz}{d\zeta}\, d\zeta \tag{4}$$

The first integral is singular at a in the form

$$\int_a^{\frac{1}{2}(a+b)} \frac{Z(\zeta)}{(\zeta - a)^{\frac{1}{2}}}\, d\zeta \tag{5}$$

and can be rewritten in the standard way as

$$\int_a^{\frac{1}{2}(a+b)} \frac{Z(\zeta) - Z(a)}{(\zeta - a)^{\frac{1}{2}}}\, d\zeta + Z(a) \int_a^{\frac{1}{2}(a+b)} \frac{1}{(\zeta - a)^{\frac{1}{2}}}\, d\zeta$$

$$= \int_a^{\frac{1}{2}(a+b)} \frac{Z(\zeta) - Z(a)}{(\zeta - a)^{\frac{1}{2}}}\, d\zeta + 2Z(a)\, [\tfrac{1}{2}(b - a)]^{\frac{1}{2}} \tag{6}$$

The integrand $[Z(\zeta) - Z(a)]/[\zeta - a]^{\frac{1}{2}}$ is evaluated on the integration interval at up to fifty points that are spaced more closely near the singularity, and a modified Simpson rule is used to perform the quadrature

In the example of Configuration 5, the mapping derivative contains seven unknown constants. Six of these constants are independent (one must be fixed to define a coordinate origin) and determine the six lengths $\overline{AB}$, $\overline{BC}$, $\overline{CD}$, $\overline{DE}$, $\overline{EF}$, and $\overline{FG}$ in Figure 3. For practical reasons, the experimental values of the mapping constants are scaled linearly

before they are used to evaluate the geometry in the real plane; this linear scaling changes only the absolute scale in the real plane, is equivalent to fixing one of the six independent mapping constants, and, for Configuration 5, is defined by setting b = 0 and e = 1. Five of the unknown mapping constants are used as the independent variables in a linear iteration scheme that is designed to satisfy the five required geometry conditions:

1) The fence must close ($\overline{BC} = \overline{CD}$).

2) The inboard wing must close ($\overline{AB} = \overline{FG}$).

3) The outboard wing must close ($\overline{DE} = \overline{EF}$).

4) The fence must have the proper length ($\overline{BC} = \ell[\overline{AB} + \overline{DE} \cos \Gamma]$).

5) The fence must have the proper semispan location ($\overline{AB} = \eta[\overline{AB} + \overline{DE} \cos \Gamma]$).

The initial values of the unknown mapping constants are measured experimentally with an analog field plotter. The corrections to the mapping constant Δp_j are computed from

$$- \Delta C_i = \frac{\partial C_i}{\partial p_j} \Delta p_j \tag{7}$$

where ΔC_i is the error in C_i, the i-th geometry requirement. The 5x5 correction matrix $\partial C_i/\partial p_j$ is evaluated numerically by perturbing p_j slightly, scaling the potentials, if necessary, and calculating the derivative as though C_i varies linearly with p_j. For Configuration 5, the potentials a, b, d, e, and g were selected from the seven unknown values of potential as the independent variables for the iteration. This simple iteration scheme converges rapidly for properly chosen independent variables. For some of the seven configurations, the set of mapping constants selected initially as independent variables did not give convergence, and other sets were chosen. However, convergence of the iteration scheme has been achieved for each of the seven configurations considered to date.

Successive solution scheme. Reference 9 proposed the development of a successive solution scheme to determine the unknown constants of the Schwarz-Christoffel mapping without resort to electrostatic analog experiments. The mapping constants are determined for a series of geometrically related configurations that ends with the desired configuration. The final values of the mapping constants for each member of the series are used as initial values for the next member of the series, whose mapping constants in turn can then be determined with the iteration scheme developed for the electrostatic analog method. The series

is started with the monoplane degenerate equivalent of the desired configuration, for which the mapping constants can be determined from the potential solution about a monoplane, given in Reference 12. This procedure to determine the mapping constants is generally successful if each member of the series differs geometrically from its neighbors by a small amount.

The Minimum Induced Drag

Once the mapping constants are known, the minimum induced drag is calculated in the terms of the efficiency k, where

$$D = \frac{L^2}{4 \pi s^2 q k} \tag{8}$$

If the expression for k is transformed into the auxiliary plane, Reference 8 shows that

$$k = \frac{1}{\pi s^2} I \int_a^g \zeta \frac{dz}{d\zeta} d\zeta \tag{9}$$

where I means "the imaginary part of". Again, the integration is performed numerically.

The Loading to Produce Minimum Induced Drag

Once the mapping constants are known, the potential is specified effectively in the physical plane as a function of geometric location. The required loading is proportional to the potential difference across each element and is obtained by interpolation in the tables resulting from the numerical integration that produces lengths in the real plane.

COMPUTER PROGRAM 55VD

Computer Program 55VD has been written in the FORTRAN IV language for use on an IBM 7094 digital computer, and has been converted to FORTRAN 2.0 for use on the Langley Research Center's CDC 6000 series computers. Figure 4 gives the overall logic of Program 55VD. In several subroutines, internal logic is used to transfer to coding associated with the configuration being considered; such logic is not shown in Figure 4. Computing time per case varies from 0.1 minute to 3 minutes, depending on which of the seven configurations is being analyzed, the number of ordinates used in the evaluation of the geometry and drag numerical

integrals, and the rate of convergence of the iteration for the mapping constants. The main program and the twenty-four subroutines of Program 55VD are written on roughly 3700 FORTRAN source cards.

Input/Output Files

Only the standard files INPUT and OUTPUT are used by Program 55VD; these files are not described. Files INPUT and OUTPUT correspond to logical units 5 and 6, respectively, of the IBM 7094 FORTRAN system.

Detailed Description of Subroutines

For the main program and each of the twenty-four subroutines, this section of the report presents a description, a source deck listing, detailed flow charts, and a list of symbols. Each list of symbols contains five columns of information:

1) The first column is the symbol.

2) The second column contains one of the letters D,I,L, or R, denoting double precision, integer, logical, or real variable, respectively.

3) The third column contains one of the letters C,D, or U, denoting a common, dimensioned, or undimenioned variable, respectively. The hierachy is C,D, and U.

4) The fourth column describes the symbol.

5) The fifth column contains the name of the subroutine.

The flow charts have been constructed with the FLOTRAN computer program described briefly in Reference 13. This program accepts IBM 7094 FORTRAN source decks as input data and constructs flow charts with a Stromberg-Carlson SC-4020 automatic data plotting system. Only two of the FLOTRAN flow charts for Computer Program 55VD have been modified. Subroutines NUMINT and NUM each have one computed GO TO statement with 91 arguments; FLOTRAN can chart computed GO TO statements with a maximum of 30 arguments. The two computed GO TO statements have therefore been charted manually.

An index for the subroutine descriptive material follows.

Table I

Index for Subroutine Descriptive Material

Subroutine Name	Subroutine [illegible] Location	Page Number			
		Descr.	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]

A core map starts on page [illegible].

[illegible]

[illegible]

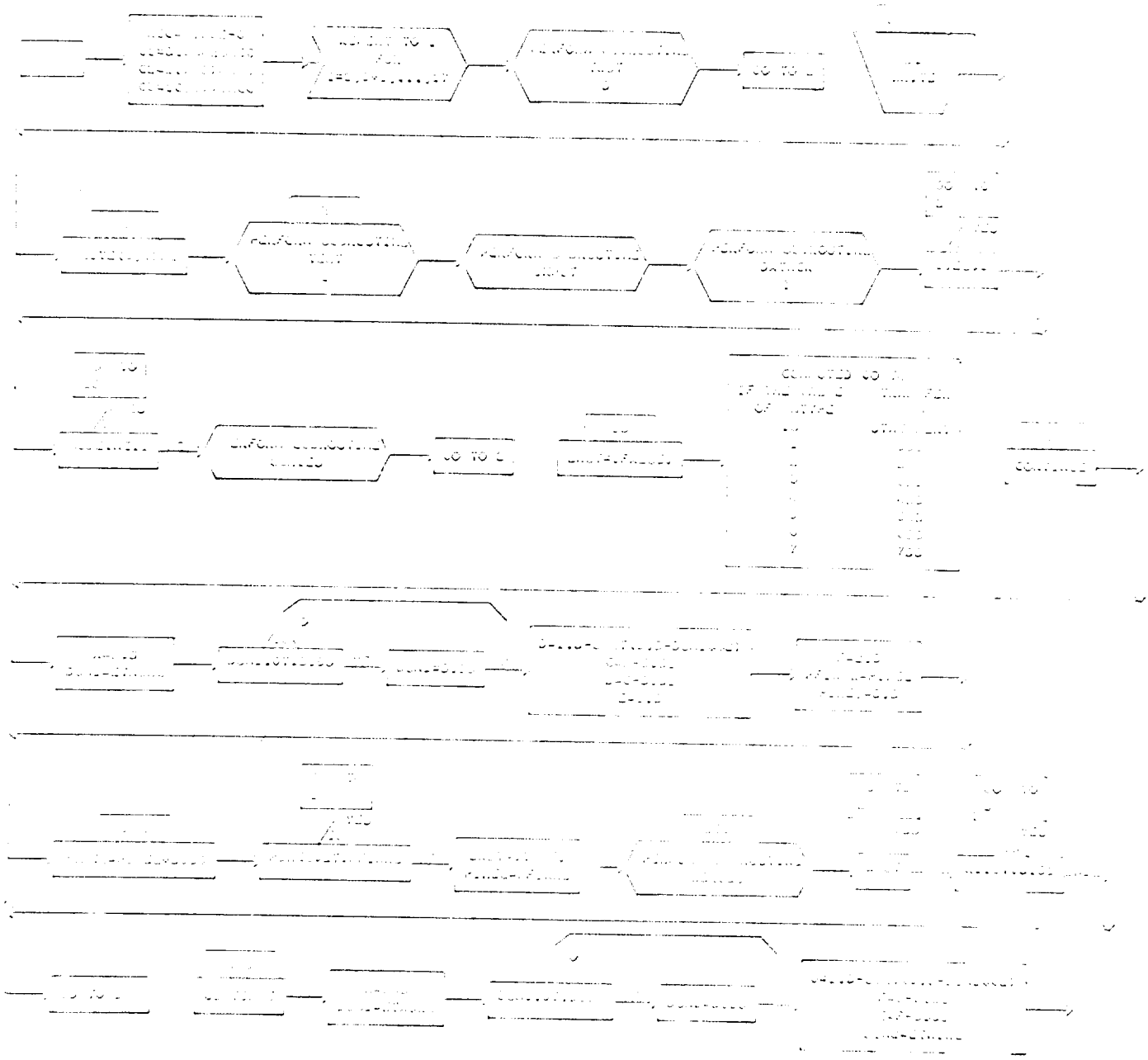
[illegible]

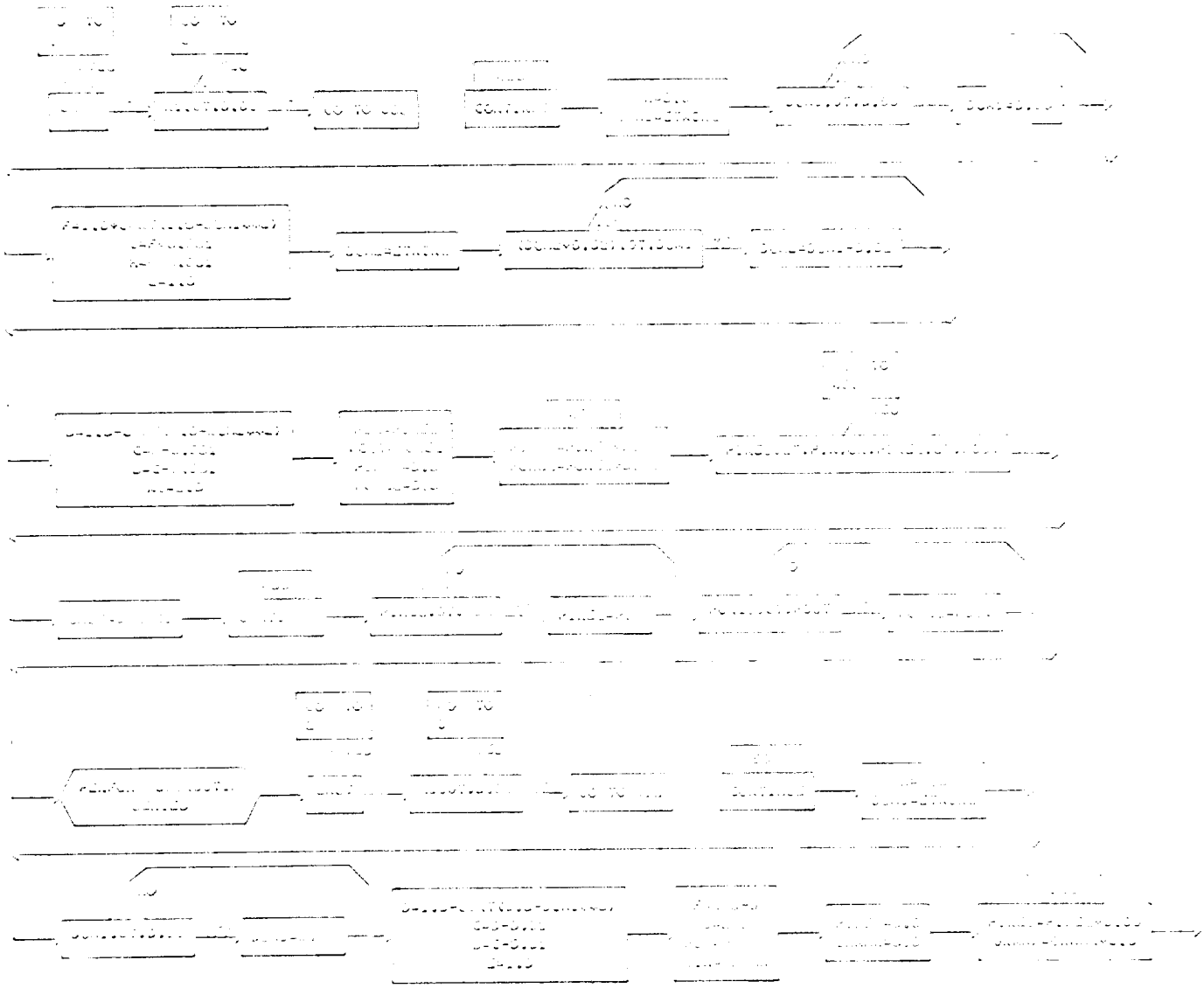
[illegible]

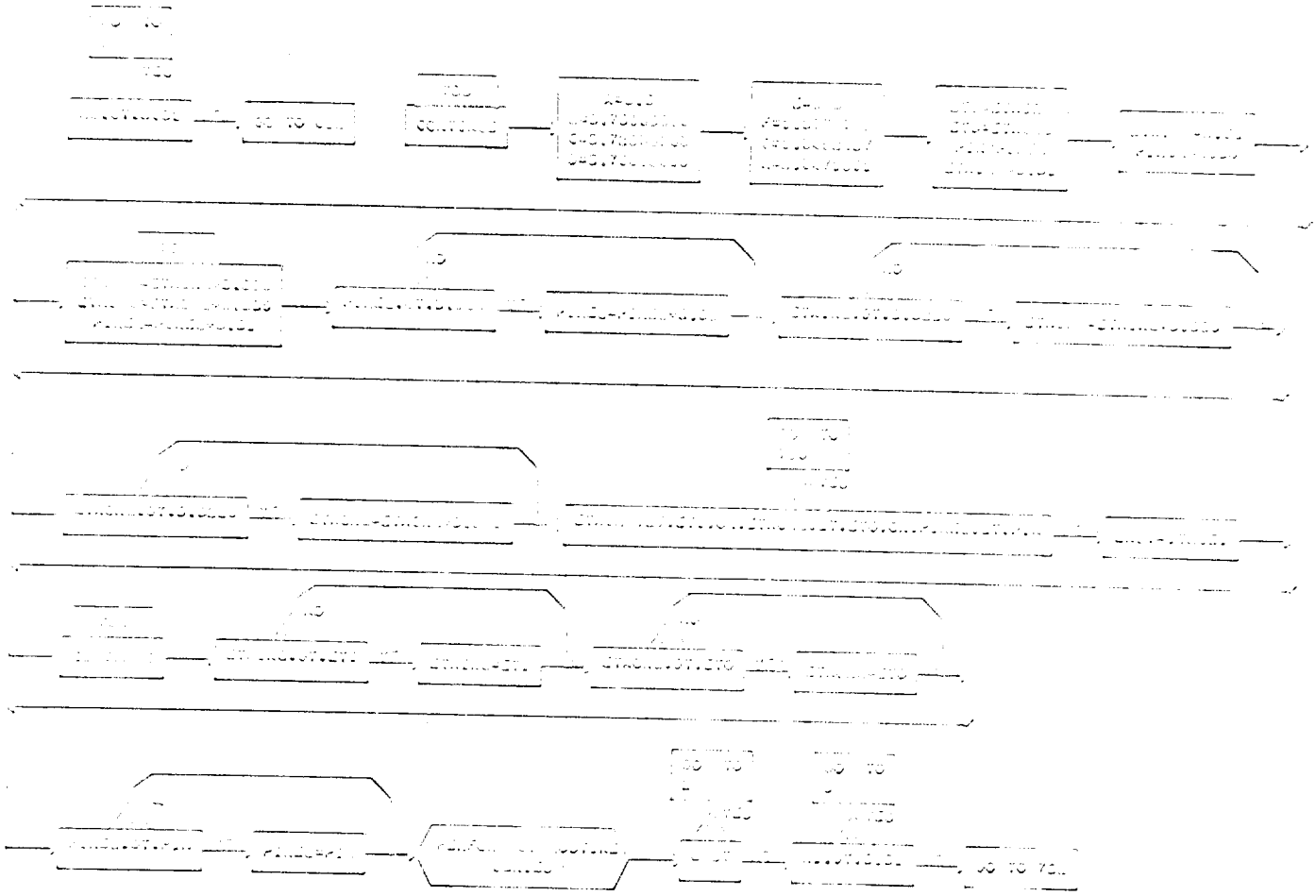
[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]

```
      [illegible]
      IF (FORCE .GT. FIN .OR. FORCE .GT. FOUT) GO TO [illegible]
      [illegible]
      [illegible] CONTINUE
C
C     WING WITH HORIZONTAL TAIL [illegible]
C
      [illegible]
      [illegible] CONTINUE
C
C     WING WITH [illegible] PYLON [illegible]
C
      [illegible]
```

```
      CALL SERIES
      IF (LAST) GO TO 2
      IF (RI .GT. 0.01) GO TO 5
      GO TO 702
  700 CONTINUE
C
C     WING WITH END-PLATED VERTICAL FENCE AT THE TIP
C     INITIAL VALUES OF POTENTIALS ARE GIVEN FOR VERTICAL FENCE LENGTH
C     OF 0.004 AND END PLATE LENGTHS OF 0.001
C
      A = [illegible]
      B = [illegible]
      C = [illegible]
      D = [illegible]
      E = [illegible]
      F = [illegible]
      G = [illegible]
      H = [illegible]
      [illegible]
      [illegible]
      [illegible]
      [illegible]
      [illegible]
      PINEQ = [illegible]
  702 ETAINC = ETAINC * [illegible]
      ETAONC = ETAONC * [illegible]
      PINEQ = PINEQ * 0.01
      IF (PINEQ .GT. 0.0[illegible]) PINEQ = PINEQ * [illegible]
      IF (ETAINC .GT. 0.0[illegible]) ETAINC = ETAINC * [illegible]
      IF (ETAONC .GT. 0.0[illegible]) ETAONC = ETAONC * [illegible]
      IF (ETAINC .LT. ETI .OR. ETAONC .LT. ETO .OR. PINEQ .LT. PIN)
     A     GO TO 705
      LAST = [illegible]
  705 IF (ETAINC .GT. ETI) ETAINC = ETI
      IF (ETAONC .GT. ETO) ETAONC = ETO
      IF (PINEQ .GT. PIN) PINEQ = PIN
      CALL SERIES
      IF (LAST) GO TO 2
      IF (RI .GT. 0.01) GO TO 5
      GO TO 702
      END
```







SYMBOLS USED IN SUBROUTINE MAIN (DECK IDENTIFICATION [illegible])

ORD	R	C	ORDINATE ARRAY FOR NUMERICAL INTEGRALS	MAIN
PFINAL	R	U	FINAL VALUE OF PFREQ	MAIN
PIN	R	U	FINAL VALUE OF PIREQ	MAIN
PIREQ	R	C	INBOARD LENGTH OF DISTANCE (NONDIMENSIONAL)	MAIN
POREQ	R	C	OUTBOARD LENGTH OF DISTANCE (NONDIMENSIONAL)	MAIN
POUT	R	U	FINAL VALUE OF POREQ	MAIN
RI	R	C	SQUARE ROOT OF SUM OF SQUARES OF ERRORS IN REQD CONDITIONS	MAIN
TAB	R	C	INTERPOLATION ARRAY FOR INDEPENDENT VARIABLE AND FUNCTION	MAIN
TABLE	R	C	INTEGRATED VALUES OF ORDINATES AS A FUNCTION OF ABSCISSA	MAIN

[illegible] This subroutine calls the [illegible] check and overflow/underflow subroutines of the [illegible] library. If any of these abnormalities are indicated, the subroutine prints a message that includes the location parameter I.

[illegible]

[illegible]

[illegible]

[illegible]

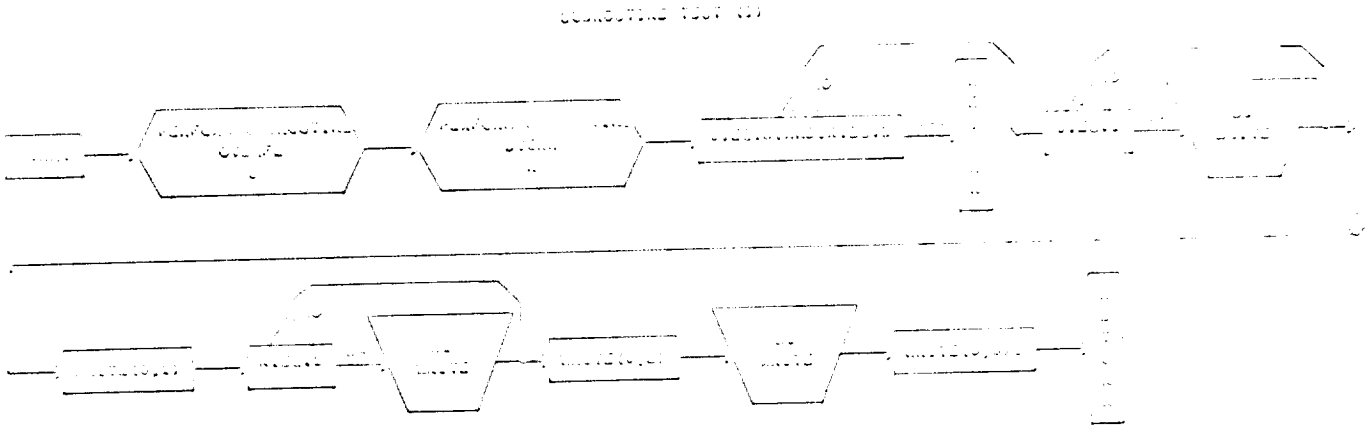
[illegible]

[illegible]

[illegible]

[illegible]

[illegible]



Subroutine INPUT (Pack 55MF). The input data for each case are read into the DATA array by subroutine INPUT, which prints a page header for each new case. If the program mode of operation is 1, four data cards are read. Two data cards are read if the program operation mode is 2. Reference 10 describes the use of the data cards in detail. Subroutine INPUT terminates execution after writing a termination message if the termination flag II is 1.

```
      SUBROUTINE INPUT
C
C     THIS SUBROUTINE READS INPUT DATA INTO THE B IN MODULE COMMON
C     BETWEEN THE FIRST ONCE THAT HAS BEEN EXECUTED. THIS SUBROUTINE
C     SHOULD NOT BE PLACED IN OVERLAYS
C
      COMMON /BLANK/ B(1000000), ORGIN(3), TRANSLX(3), NORMS(3,3)
     1 /INPT/ IT(3,3,4), SP(3), DP(3), DUP, DUMP, DUMY, DUMZ, TEMPX, DIF, DZ
     2 DZ, NS
C
      DIMENSION PTSDIV(3), ELEDIV(3), CTSDIV, MPTDIV, NDIV
C
      EQUIVALENCE
     1 (B(1), PTS), (B(2), ELE), (B(3), CTS), (B(4), MPT), (B(5), NPT),
     2 (B(6), NEL), (B(7), NCT), (B(8), NMT), (B(9), NOUT), (B(10), NTIT),
     3 (B(11), NCOM), (B(12), NPRINT), (B(13), NPUNCH), (B(14), NTIMELIMIT)
C
      DATA NOUT, NOUTX, DELTT / 6,6,0.0E-20 /
      DATA IT / 0 /
C
      DO 10 I = 1, 10
      READ(NOUTX,
      WRITE(NOUT,
      STOP 'ERROR IN SUBROUTINE INPUT'
C
      DO 20 I = 1, 10
      READ(NOUTX,
      WRITE(NOUT,
      STOP
C
      WRITE(NOUT,
C
      WRITE(NOUT,
C
      WRITE(NOUT,
C
      WRITE(NOUT,
C
C     WRITE(NOUT,
C     WRITE(NOUT,
C     WRITE(NOUT,
```

PROGRAM LENGTH
000100

FUNCTION ASSIGNMENTS

STATEMENT ASSIGNMENTS
[illegible]

BLOCK NAMES AND LENGTHS
[illegible]

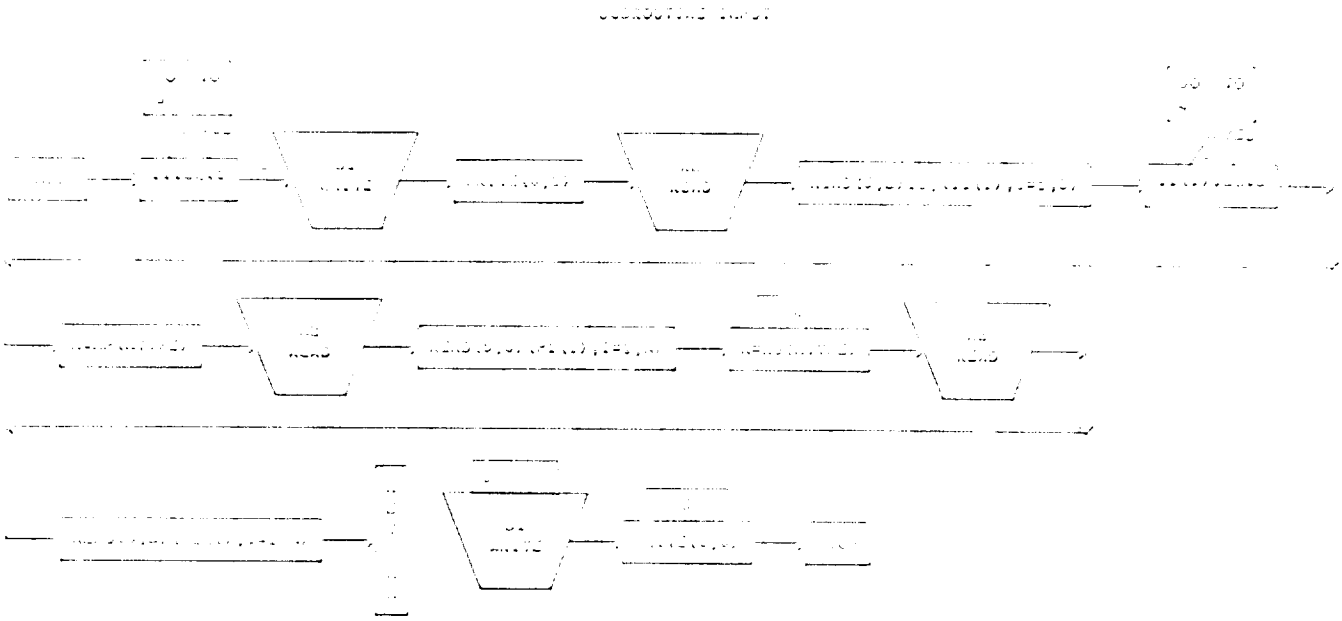
VARIABLE ASSIGNMENTS
[illegible]

LIST OF CONSTANTS
000071

LIST OF TEMPORARIES
000100

LIST OF INDIRECTS
000100

LOCATE COMPILER SPACE
045000



SYMBOLS USED IN SUBROUTINE INPUT (DECK MODIFICATION SDW)

A	R	C	POTENTIAL AT A (MAPPING CONSTANT)	INPUT
AA	R	C	ARRAY FOR RESULTS OF CONSTANT AND REAL INTEGRALS	INPUT
ABSC	R	C	ABSCISSA ARRAY FOR NUMERICAL INTEGRALS	INPUT
AI	R	C	POTENTIAL AT I (MAPPING CONSTANT)	INPUT
ALPHA	R	C	STORAGE ARRAY FOR POTENTIAL IN THE REAL (PHYSICAL) PLANE	INPUT
B	R	C	POTENTIAL AT B (MAPPING CONSTANT)	INPUT
C	R	C	POTENTIAL AT C (MAPPING CONSTANT)	INPUT
CA	R	C	COSINE OF DIHEDRAL ANGLE	INPUT
CCIM	R	C	FACTOR FOR IMAGINARY PART OF COMPLEX DISTANCE	INPUT
CCRR	R	C	FACTOR FOR REAL PART OF COMPLEX DISTANCE	INPUT
C1	R	C	CONSTANT THAT IS COMPUTER HARDWARE DEPENDENT	INPUT
C2	R	C	CONSTANT THAT IS COMPUTER HARDWARE DEPENDENT	INPUT
C3	R	C	CONSTANT THAT IS COMPUTER HARDWARE DEPENDENT	INPUT
D	R	C	POTENTIAL AT D (MAPPING CONSTANT)	INPUT
DATA	R	C	ARRAY FOR USER DATA	INPUT
E	R	C	POTENTIAL AT E (MAPPING CONSTANT)	INPUT
EIAIRC	R	C	INBOARD SPINDLE LOCATION (CENTIMETERSIONLY)	INPUT
EIAGRC	R	C	OUTBOARD SPAN SEP LOCATION (CENTIMETERSIONLY)	INPUT
F	R	C	POTENTIAL AT F (MAPPING CONSTANT)	INPUT
G	R	C	POTENTIAL AT G (MAPPING CONSTANT)	INPUT
GAMMA	R	C	DIHEDRAL ANGLE IN DEGREES	INPUT
GP	R	C	DIHEDRAL ANGLE DIVIDED BY PI	INPUT
GPH2	R	C	GP * 0.5	INPUT
GPP2	R	C	GP * 0.5	INPUT
GJ	R	C	CONSTANT THAT IS HIGHLY DEPENDENT UPON	INPUT
H	R	C	POTENTIAL AT H (MAPPING CONSTANT)	INPUT
I	I	U	DO LOOP INDEX	INPUT
IF	I	C	FLAG FOR FIRST CASE	INPUT
ILAST	I	C	FLAG FOR LAST INTEGRATION	INPUT
IKINT	I	C	NUMBER OF CONSTANT TO NUMERICAL INTEGRALS	INPUT
IJ	I	C	INDICES INTEG	INPUT
IOPC	I	C	CONTROL FOR OF OPERATIONS	INPUT
K	I	U	DO LOOP INDEX FOR NUMBER OF CASES	INPUT
KS	I	D	NUMBER OF CONSTANT OPERATIONS, IN PUBLIC LIBRARY INPUT	INPUT
LP	I	D	NUMBER OF POTENTIALS AS A FUNCTION OF INDEX	INPUT
MTYPE	I	C	CONFORMAL MAP TYPE	INPUT
OPCP	R	C	J.C - GP	INPUT
OPD	R	C	OPERATOR ARRAY FOR NUMERICAL RESULTS	INPUT

[illegible]

[illegible]	I	C	[illegible]
[illegible]	R	C	[illegible]
[illegible]	R	C	[illegible]
[illegible]	R	C	[illegible]
[illegible]	R	C	[illegible]
[illegible]	R	C	[illegible]

Subroutine SERIES (Pack 55W2). This subroutine serves as an executive control, and makes ten calls of other subroutines — TRANS, TEST, HEADER, INNER, TEST, POTOUT, DRAGT, TEST, LOAD, and TEST, in that order.

```
      SUBROUTINE SERIES
C     THIS SUBROUTINE CALLS A STANDARD SERIES OF SUBROUTINES
C
      [illegible]
      [illegible]
      [illegible]
C
      END [illegible]
      [illegible]
      [illegible]
      [illegible]
      [illegible]
      [illegible]
      [illegible]
C
      [illegible]
      [illegible]
      [illegible]
      [illegible]
      [illegible]
      [illegible]
      [illegible]
      [illegible]
      [illegible]
      [illegible]
      END
```

[illegible] GRAM LENGTH
[illegible]

FUNCTION ASSIGNMENTS

STATEMENT ASSIGNMENTS

BLOCK NAMES AND LENGTHS
[illegible] = [illegible]

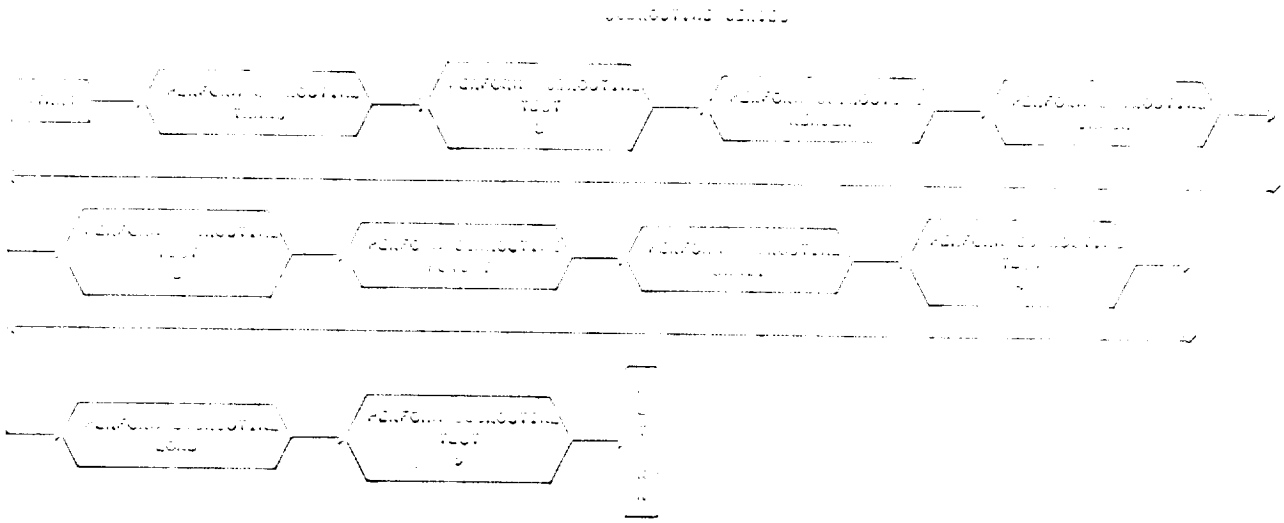
VARIABLE ASSIGNMENTS
[illegible] = [illegible] [illegible] = [illegible] [illegible] = [illegible] DATA = [illegible]
[illegible] = [illegible] [illegible] = [illegible] [illegible] = [illegible]

[illegible] OF CONSTANTS
[illegible]

[illegible] OF TEMPORARIES
[illegible]

[illegible] OF [illegible]
[illegible]

[illegible] COMPILER SPACE
[illegible]



SYMBOLS USED IN SUPERSONIC SERIES (DOUBLE-PRECISION SERIES)

A	R	C	POTENTIAL AT A (MAPPING CONSTANT)	SERIES
AA	R	C	ARRAY FOR RESULTS OF GEOMETRY AND DRAG INTEGRALS	SERIES
ABSC	R	C	ABSCISSA ARRAY FOR NUMERICAL INTEGRALS	SERIES
AI	R	C	POTENTIAL AT I (MAPPING CONSTANT)	SERIES
ATEAL	R	C	STORAGE ARRAY FOR POTENTIAL IN THE REAL (PHYSICAL) PLANE	SERIES
B	R	C	POTENTIAL AT B (MAPPING CONSTANT)	SERIES
C	R	C	POTENTIAL AT C (MAPPING CONSTANT)	SERIES
CG	R	C	CENTER OF PHYSICAL PLANE	SERIES
CGMI	R	C	FACTOR FOR IMAGINARY PART OF COMPLEX DISTANCE	SERIES
CGNR	R	C	FACTOR FOR REAL PART OF COMPLEX DISTANCE	SERIES
C1	R	C	CONSTANT THAT IS COMPUTER HARDWARE DEPENDENT	SERIES
C2	R	C	CONSTANT THAT IS COMPUTER HARDWARE DEPENDENT	SERIES
C3	R	C	CONSTANT THAT IS COMPUTER HARDWARE DEPENDENT	SERIES
D	R	C	POTENTIAL AT D (MAPPING CONSTANT)	SERIES
DATA	R	C	ARRAY FOR INPUT DATA	SERIES
E	R	C	POTENTIAL AT E (MAPPING CONSTANT)	SERIES
EYAFRC	R	C	INBOARD SPANWISE LOCATION (PROPORTIONAL)	SERIES
EYAOFC	R	C	OUTBOARD SPANWISE LOCATION (PROPORTIONAL)	SERIES
F	R	C	POTENTIAL AT F (MAPPING CONSTANT)	SERIES
G	R	C	POTENTIAL AT G (MAPPING CONSTANT)	SERIES
GAMMA	R	C	FREEDRAG FACTOR IN DRAG	SERIES
GP	R	C	GAMMA PLUS DIVIDED BY 2	SERIES
GPM2	R	C	GP - 0.5	SERIES
GPP2	R	C	GP + 0.5	SERIES
H	R	C	POTENTIAL AT H (MAPPING CONSTANT)	SERIES
HLAST	I	C	FLAG FOR LAST INTERVAL	SERIES
HLINT	I	C	NUMBER OF ORDINATES FOR INTEGRAL	SERIES
HOOS	I	C	PROGRAM MODE OF OPERATION	SERIES
MTYPE	I	C	CONFIGURATION TYPE	SERIES
OMGP	R	C	1.0 - GP	SERIES
ORD	R	C	ORDINATE ARRAY FOR NUMERICAL INTEGRALS	SERIES
PIRLG	R	C	INBOARD PHYSICAL COORDINATE (OUTER WINGPANEL)	SERIES
PLPLG	R	C	OUTBOARD PHYSICAL COORDINATE (OUTER WINGPANEL)	SERIES
RD	R	C	SCALE FOR A SET OF SPANWISE LOCATIONS IN PHYSICAL COORDINATES	SERIES
TAB	R	C	INTERPOLATION ARRAY FOR DIFFERENCE WEIGHT AND FORCE	SERIES
TABI	R	C	INTEGRATED VALUES OF ORDINATES AS A FUNCTION OF ABSCISSA	SERIES

[illegible]. This subroutine transforms the mapping constants with a linear scaling - the mapping constant is set to zero and another is set to one. The subroutine first determines the number of mapping constants and which the mapping constants are used to define the linear scaling by referring to the configuration number (See Figure 2). It then performs the linear scaling.

```
      SUBROUTINE TRANS
C
C     THIS SUBROUTINE SCALES LINEARLY THE CONSTANTS OF THE SCHWARZ-
C     CHRISTOFFEL TRANSFORMATION
C
      COMMON DATA(17), ZCOS(100), CHI(100), THETA(100), RADIUS(100), ANTIC
C     1 (100), TABZ(100), CP, SPHI, CPHI, SDV, CNSP, SCHV, SCHV, VEL0, CLV, CD
C     2 , CDV, N1
C
      EQUIVALENCE                    (DATA(1),                    N1),
C     1 (DATA(2),                    N2),                         (DATA(3),
C     2 (DATA(4),                    N3),                         (DATA(5),
C     3 (DATA(6),                    N4),                         (DATA(7),
C     4 (DATA(8),                    N5),                         (DATA(9),
C     5 (DATA(10),                   N6),                         (DATA(11),
C     6 (DATA(12),                   N7),                         (DATA(13),
C     7 (DATA(14),                   N8),                         (DATA(15),
C     8 (DATA(16),                   N9),                         (DATA(17),
C     9 (DATA(18),                   N10),                        (DATA(19),
C     A (DATA(20),                   N11),                        (DATA(21),
C     B (DATA(22),                   N12),                        (DATA(23),
C     C (DATA(24),                   N13),                        (DATA(25),
C     D (DATA(26),                   N14),                        (DATA(27),
C     E (DATA(28),                   N15),                        (DATA(29),
C     F (DATA(30),                   N16),                        (DATA(31),
C     G (DATA(32),                   N17),                        (DATA(33),
C     H (DATA(34),                   N18),                        (DATA(35),
C     I (DATA(36),                   N19),                        (DATA(37),
C     J (DATA(38),                   N20),                        (DATA(39),
C     K (DATA(40),                   N21),                        (DATA(41),
C     L (DATA(42),                   N22),                        (DATA(43),
C     M (DATA(44),                   N23),                        (DATA(45),
C     N (DATA(46),                   N24),                        (DATA(47),
C     O (DATA(48),                   N25),                        (DATA(49),
C     P (DATA(50),                   N26),                        (DATA(51),
C     Q (DATA(52),                   N27),                        (DATA(53),
C     R (DATA(54),                   N28),                        (DATA(55),
C     S (DATA(56),                   N29),                        (DATA(57),
C     T (DATA(58),                   N30),                        (DATA(59),
C     U (DATA(60),                   N31),                        (DATA(61),
C     V (DATA(62),                   N32),                        (DATA(63),
C     W (DATA(64),                   N33),                        (DATA(65),
C     X (DATA(66),                   N34),                        (DATA(67),
C     Y (DATA(68),                   N35),                        (DATA(69),
C     Z (DATA(70),                   N36),                        (DATA(71),
C
C     CONTAINS DATA ARRAY NUMBER FOR POTENTIAL TRANSFORMED TO THE RING
C     VELOCITY FOR POTENTIAL TRANSFORMED TO CIRCULARITY AND FOR TOTAL
C     NUMBER OF POTENTIALS INTO
C
      DELTA = 1
      DO 10 I = 1, 10, 20, 30, 40, 50, 60, 70, 80, NTVPL
    1 DELTA = DELTA * I
    2 DELTA = DELTA * I
    3 DELTA = DELTA * I
    4 DELTA = DELTA * I
    5 TONE = 5
      DO 10 I = 10, 50, 70, 90, 110, 130, 150, 170, 190, NTVPL
    6 TONE = TONE * I
    7 TONE = TONE * I
      DO 10 I = 110, 210, 310, 410, 510, 610, 710, 810, NTVPL
    8 N = N * I
    9 N = N * I
   10 N = N * I
   11 VELNO = DATA(VELNO)
      SCLV = DATA(SCLV) * VELNO
C
C     SCALE POTENTIALS (CONSTANTS OF SCHWARZ-CHRISTOFFEL TRANSFORMATION)
C
      DO 12 I=1,NTVPL
   12 DATA(I) = DATA(I) * VELNO / SCLV
      RETURN
      END
```

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

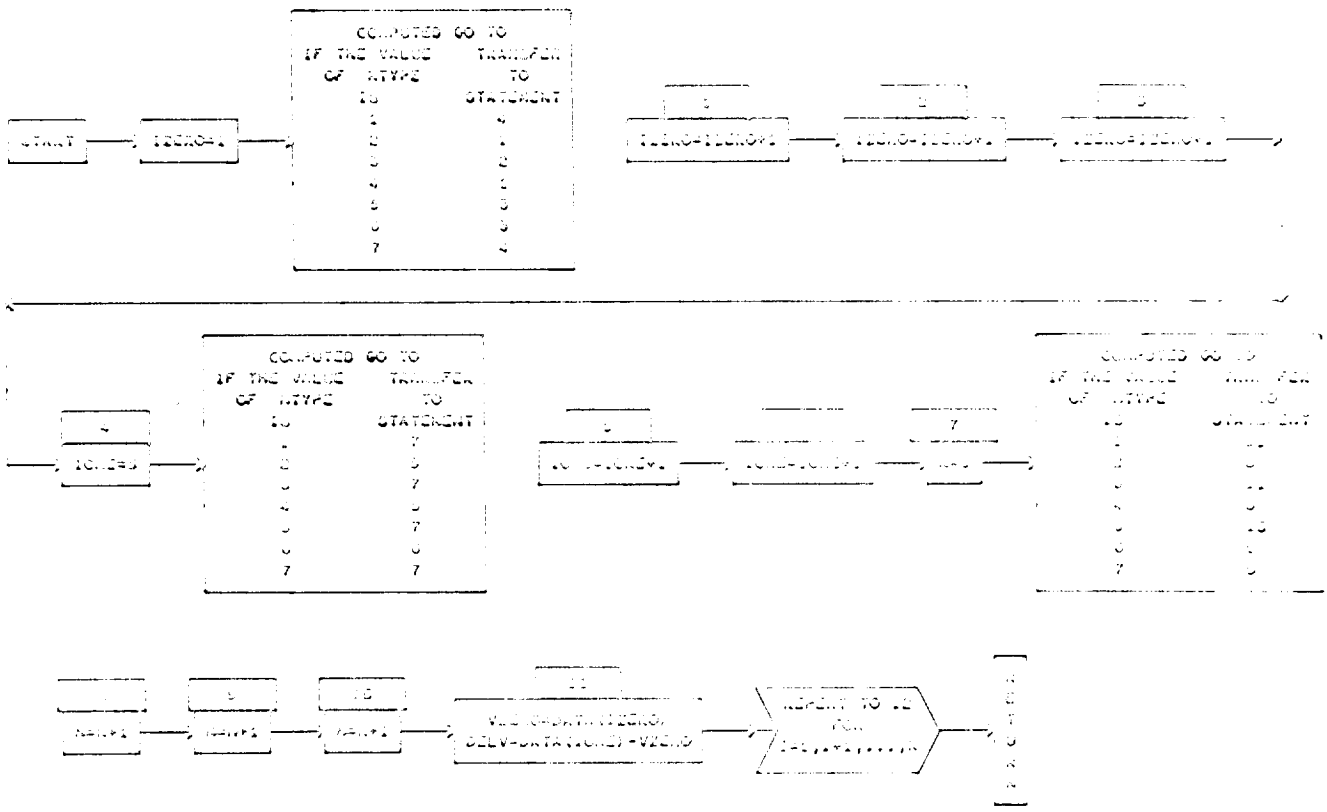
[illegible]

[illegible]

[illegible]

[illegible]

SUBROUTINE THANG



SYMBOLS USED IN SUBROUTINE TRANS (FROM FORTRAN SOURCE)

A	R	C	POTENTIAL AT A (MAPPING CONSTANT)	TRANS
AA	R	C	ARRAY FOR RESULTS OF CONSTANT AND DATA DISTORTIONS	TRANS
ABSC	R	C	ABSCISSA ARRAY FOR EQUIPOTENTIAL INTERCEPTS	TRANS
A1	R	C	POTENTIAL AT A1 (MAPPING CONSTANT)	TRANS
ALGAM	R	C	STORAGE ARRAY FOR POTENTIALS IN THE REAL (PHYSICAL) PLANE	TRANS
B	R	C	POTENTIAL AT B (MAPPING CONSTANT)	TRANS
C	R	C	POTENTIAL AT C (MAPPING CONSTANT)	TRANS
CC	R	C	COSINE OF DIHEDRAL ANGLE	TRANS
CCRI	R	C	FACTOR FOR IMAGINARY PART OF COMPLEX DISTANCE	TRANS
CCRR	R	C	FACTOR FOR REAL PART OF COMPLEX DISTANCE	TRANS
C1	R	C	CONSTANT THAT IS COMPUTER HARDWARE DEPENDENT	TRANS
C2	R	C	CONSTANT THAT IS COMPUTER HARDWARE DEPENDENT	TRANS
C3	R	C	CONSTANT THAT IS COMPUTER HARDWARE DEPENDENT	TRANS
D	R	C	POTENTIAL AT D (MAPPING CONSTANT)	TRANS
DATA	R	C	ARRAY FOR PLOT DATA	TRANS
DELV	R	C	POTENTIAL INCREMENT BEFORE STARTING (BASELINE DETAILERS)	TRANS
E	R	C	POTENTIAL AT E (MAPPING CONSTANT)	TRANS
ETAIRC	R	C	IMPRINT STARTING LOCATION (ONE DIMENSIONAL)	TRANS
ETAOUT	R	C	OUTPUT STARTING LOCATION (ONE DIMENSIONAL)	TRANS
F	R	C	POTENTIAL AT F (MAPPING CONSTANT)	TRANS
G	R	C	POTENTIAL AT G (MAPPING CONSTANT)	TRANS
GAMMA	R	C	DIHEDRAL ANGLE IN DEGREES	TRANS
GP	R	C	DIMENSIONLESS DISTANCE BETWEEN	TRANS
GPM2	R	C	GP - 0.5	TRANS
GPP2	R	C	GP + 0.5	TRANS
H	R	C	POTENTIAL AT H (MAPPING CONSTANT)	TRANS
ILAST	I	C	FLAG FOR LAST ITERATION	TRANS
ININ1	I	C	NUMBER OF ORDINATES IN NUMERICAL INTEGRATION	TRANS
ICNE	I	C	NUMBER OF POTENTIAL IN DATA ARRAY TO BE SCALED TO ONE	TRANS
IZERO	I	C	NUMBER OF POTENTIAL IN DATA ARRAY TO BE SCALED TO ZERO	TRANS
MODE	I	C	FLOWAT MODE OF OPERATION	TRANS
N	I	C	NUMBER OF POTENTIALS (MAPPING CONSTANTS)	TRANS
NTYPE	I	C	CONFIGURATION TYPE	TRANS
OPGP	R	C	1.0 - GP	TRANS
ORD	R	C	ORDINATE ARRAY FOR NUMERICAL INTEGRATION	TRANS
PIREC	R	C	INBOARD LENGTH OR DISTANCE (ONE DIMENSIONAL)	TRANS
POREC	R	C	OUTBOARD LENGTH OR DISTANCE (ONE DIMENSIONAL)	TRANS
RJ	R	C	SQUARE ROOT OF SUM OF SQUARES OF ERRORS IN PLOT COORDINATES	TRANS

SYMBOLS USED IN SUBROUTINE TRANS (DECK IDENTIFICATION, 55VH)

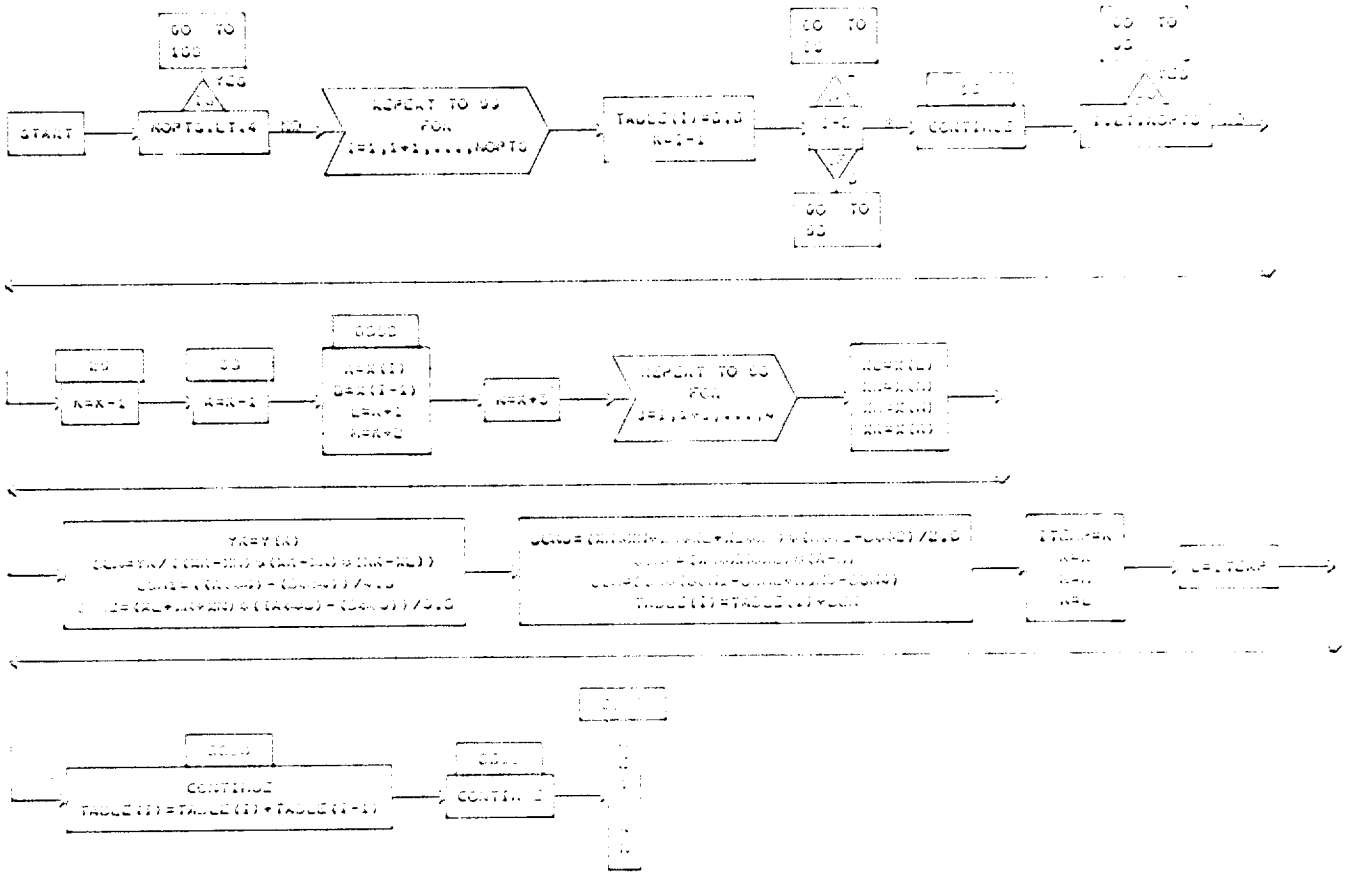
TAB	R	C	INTERPOLATION ARRAY FOR INDEPENDENT VARIABLE AND FUNCTION	TRANS
TABLE	R	C	INTEGRATED VALUES OF ORDINATES AS A FUNCTION OF ABSCISSA	TRANS
VZERO	R	U	DATA(IZERO) BEFORE SCALING	TRANS

Subroutine ITS1 (X, Y, TABLE, NOPTS) (Deck 55VT). This subroutine was completed on August 21, 1963 by G. E. Short of the Scientific Computer Programming Group for the Douglas Aircraft Company's set of library subroutines. Given a table of abscissae X and a corresponding table of ordinates Y, the subroutine uses a modified Simpson rule to evaluate the integral at each abscissa and stores the result in TABLE. NOPTS is the number of abscissae, ordinates, and integrals, and must be four or greater. The abscissa spacing need not be uniform.

```
      SUBROUTINE ITSI(X,Y, TABLE, NOPTS)
C
C     THIS SUBROUTINE WAS COMPLETED ON 8/11/68 BY G. E. SMOOT FOR THE
C     DOUGLAS SET OF STANDARD LIBRARY ROUTINES. GIVEN A MONOTONE SET
C     OF INDEPENDENT VARIABLE X WITH UNIQUE SPACING, THE SUBROUTINE
C     USES A MODIFIED SIMPSON RULE TO NUMERICALLY INTEGRATE THE
C     FUNCTION Y(X) AND STORE THE RESULT, STATION BY STATION, IN
C     TABLE(X).
C
      DOUBLE PRECISION A , B , SUM , SUM1 , SUM2 , SUM3 , SUM4 , XL ,
     1 XK , XM , XN , YK
      DIMENSION X(300),Y(300),TABLE(300)
C
      IF ( NOPTS .LT. 4 )  GO TO 100
      DO 99  I = 1, NOPTS , 1
      TABLE(I) = 0.0
      K = I - 1
      IF( I - 2 ) 99, 50 , 15
   15 IF( I .LT. NOPTS )  GO TO 30
   25 K = K - 1
   30 K = K + 1
 0050 A = X(I)
      B =  X(I -1)
      L = K + 1
      M = K + 2
      N = K + 3
      DO 98 J = 1,4,1
      XL = X(L)
      XM = X(M)
      XN = X(N)
      XK = X(K)
      YK = Y(K)
      SUM =  YK / (( XK - XM ) * (XK - XN) * (XK - XL) )
      SUM1 =  ( (A**4) - (B**4))/4.
      SUM2 = ( XL * XM * XN ) * ((A**2 - B**2) - (B**3)) / 3.0
      SUM3 = (XM* XN  + XM* XL  + XL*XN ) * (A**2 - B**2 ) /2.0
      SUM4 =  (XM* XN  + XL) * (A*B)
      SUM  = SUM* (SUM1 - SUM2 + SUM3 - SUM4)
      TABLE(I) = TABLE(I) + SUM
      ITEMP = K
      K = N
      N = M
      M = L
      L = ITEMP
 0098 CONTINUE
      TABLE(I) = TABLE(I) +  TABLE(I-1)
 0099 CONTINUE
 0100 RETURN
      END
```

```
SUBPROGRAM LENGTH
000301

FUNCTION ASSIGNMENTS

STATEMENT ASSIGNMENTS
[illegible]  -  [illegible]   [illegible]  -  [illegible]   [illegible]  -  [illegible]   [illegible]  -  [illegible]
[illegible]  -  [illegible]   [illegible]  -  [illegible]

BLOCK NAMES AND LENGTHS

VARIABLE ASSIGNMENTS
[illegible]  -  [illegible]   [illegible]  -  [illegible]   [illegible]  -  [illegible]   STEMP  -  [illegible]
[illegible]  -  [illegible]   K  -  [illegible]   [illegible]  -  [illegible]   [illegible]  -  [illegible]
[illegible]  -  [illegible]   SUM  -  [illegible]   SUM1  -  [illegible]   SUM2  -  [illegible]
SUM3  -  [illegible]   SUM4  -  [illegible]   XK  -  [illegible]   XL  -  [illegible]
XM  -  [illegible]   XN  -  [illegible]   YK  -  [illegible]

START OF CONSTANTS
[illegible]

START OF TEMPORARIES
[illegible]

START OF INDIRECTS
[illegible]

UNUSED COMPILER SPACE
042100
```

SUBROUTINE ITGI(X,Y, TABLE, NOPTO)



SYMBOLS USED IN SUBROUTINE ITS1 (BLOCK DESCRIPTION DATA)

A	D	U	DUMMY VARIABLE	ITS1
B	D	U	DUMMY VARIABLE	ITS1
I	I	U	DO LOOP INDEX	ITS1
ITEMP	I	U	TEMPORARY STORAGE VARIABLE FOR K	ITS1
K	I	U	COUNTER FOR ORDINATES	ITS1
L	I	U	COUNTER FOR ORDINATES	ITS1
M	I	U	COUNTER FOR ORDINATES	ITS1
N	I	U	COUNTER FOR ORDINATES	ITS1
NOPTS	I	U	NUMBER OF ORDINATES IN Y	ITS1
SUM	D	U	INTERMEDIATE SUM	ITS1
SUM1	D	U	INTERMEDIATE SUM	ITS1
SUM2	D	U	INTERMEDIATE SUM	ITS1
SUM3	D	U	INTERMEDIATE SUM	ITS1
SUM4	D	U	INTERMEDIATE SUM	ITS1
TABLE	R	D	INTEGRAL ARRAY	ITS1
X	R	D	ABSCISSA ARRAY	ITS1
XK	D	U	INTERMEDIATE ABSCISSA	ITS1
XL	D	U	INTERMEDIATE ABSCISSA	ITS1
XM	D	U	INTERMEDIATE ABSCISSA	ITS1
XN	D	U	INTERMEDIATE ABSCISSA	ITS1
Y	R	D	ORDINATE ARRAY	ITS1
YK	D	U	DUMMY ORDINATE	ITS1

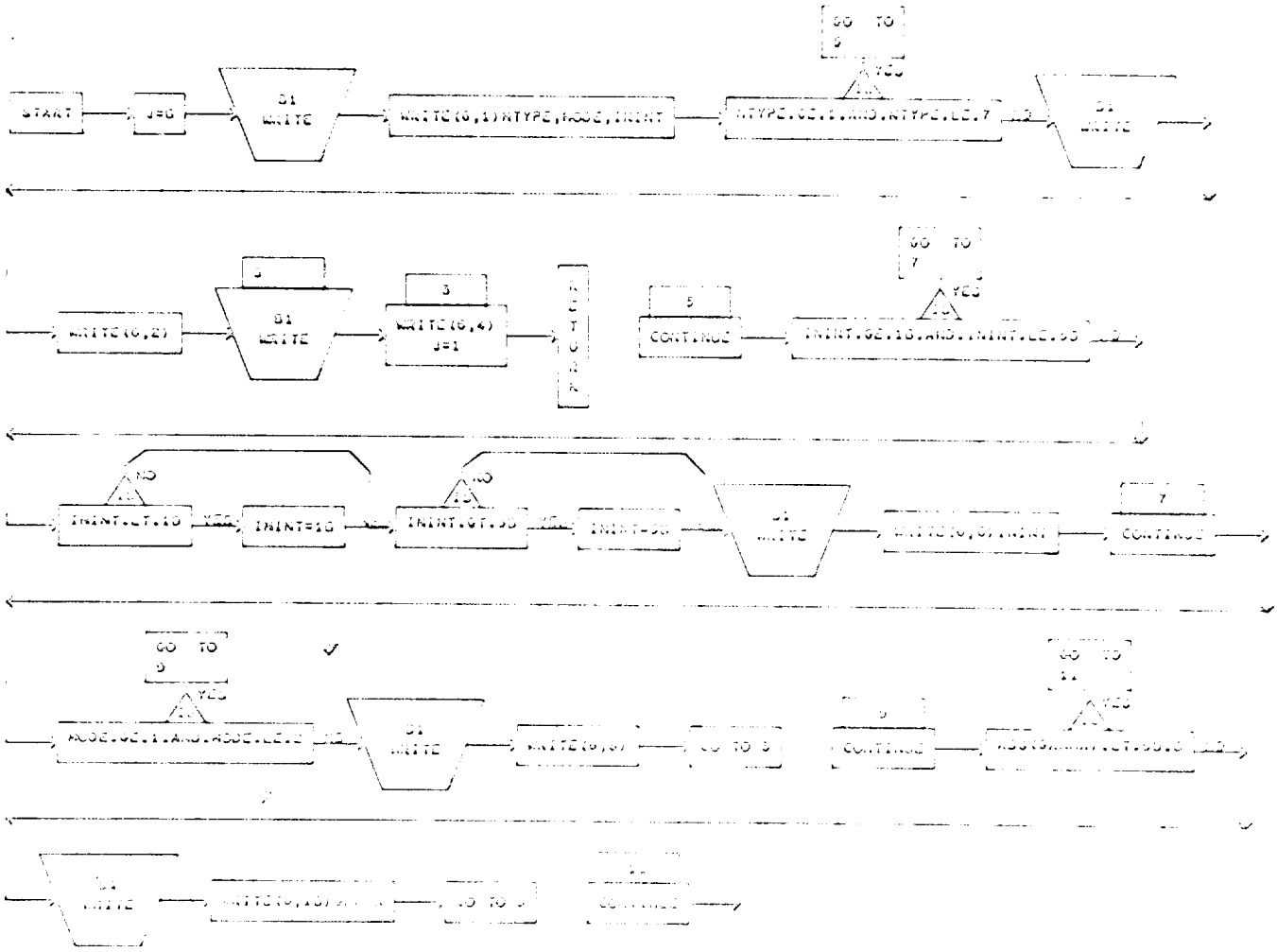
[illegible] The input data read into the [illegible] array by subroutine [illegible] are tested for validity by this subroutine. Subroutine DATACK prints an appropriate error message and sets the failure parameter 0 to 1, terminating execution of the case in the main program, if the input data satisfy any of the following conditions: the configuration number NTYPE is less than 1 or greater than 7; the program operation parameter MODE is neither 1 nor 2; the magnitude of the dihedral angle GAMMA is greater than 90 degrees; the inboard span station ETAIRQ is greater than the outboard station ETAORQ for configurations 2, 4, 6, or 7; or the mapping constants are not monotonically increasing as required by the theory. Subroutine DATACK also tests the number of ordinates ININT. If ININT is less than 10, it is set to 10; if it is greater than 50, it is set to 50. If one of these two events occurs, a message is printed to indicate the change that has been made. If MODE = 1, subroutine DATACK writes the initial values of the mapping constants before they are scaled, calls the scaling routine TRANS, and writes the scaled values of the mapping constants.

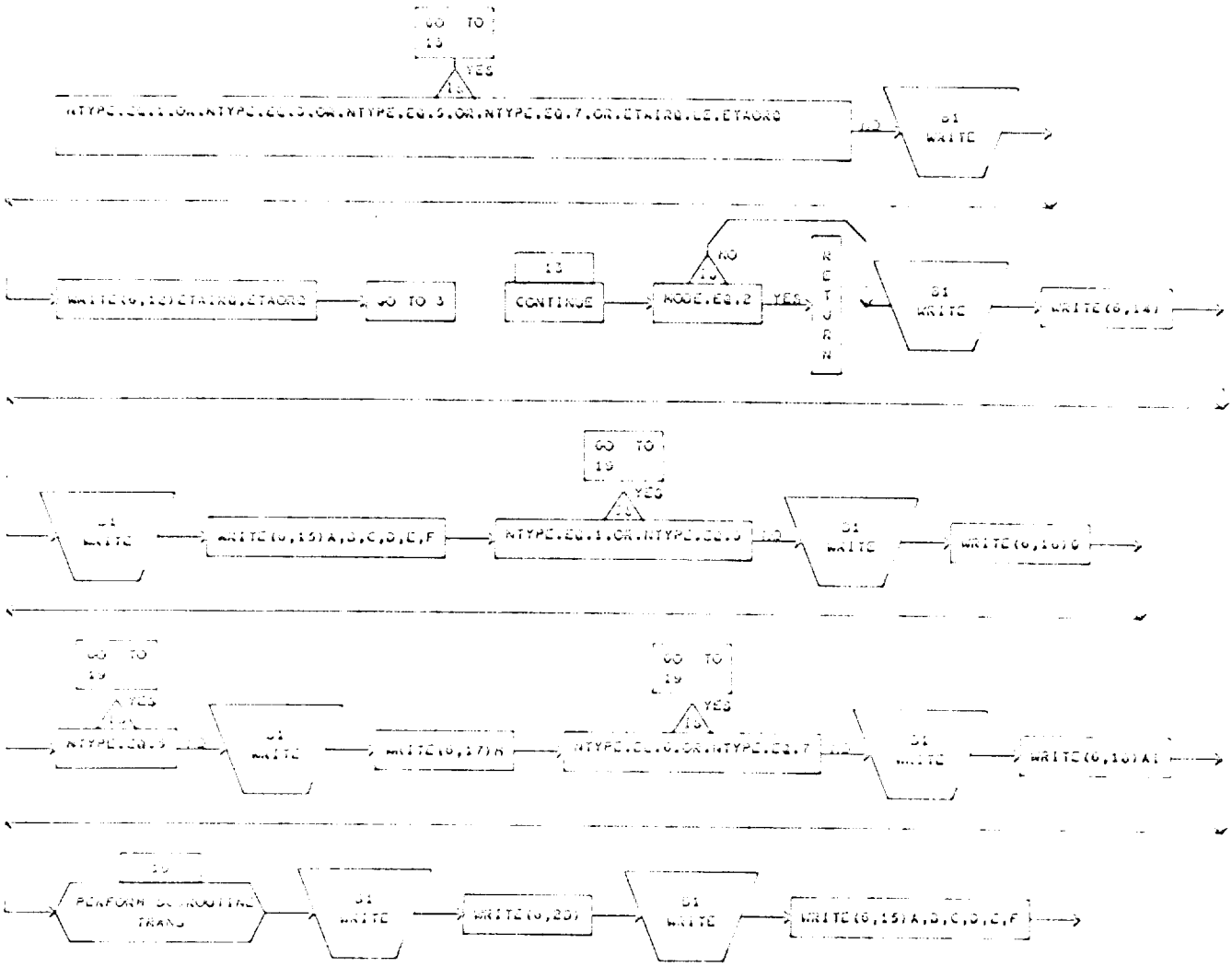
```
      SUBROUTINE DATACK(J)
C
C     THIS SUBROUTINE TESTS INPUT DATA AND PRINTS INPUT AND TRANSFORMED
C     POTENTIALS IF PROGRAM MODE IS 1.
C
000003 COMMON DATA(17), ABSC(50), ORD(50), TABLE(50), ALGAD(1000), AA(16)
     A , TAB(101), GP, GPM2, GPP2, CG, OMGP, CONR, CONI, ILAST, C1, C2
     B , C3, K1
C
000005 EQUIVALENCE              (DATA( 1),       A) , (DATA( 2),       B)
     A , (DATA( 3),       C) , (DATA( 4),       D) , (DATA( 5),       E)
     B , (DATA( 6),       F) , (DATA( 7),       G) , (DATA( 8),       H)
     C , (DATA( 9),      A1) , (DATA(10),    MODE) , (DATA(11),   NTYPE)
     D , (DATA(12),   ININT) , (DATA(13), ETAIRQ) , (DATA(14), ETAORQ)
     E , (DATA(15),   PIREQ) , (DATA(16),   POREQ) , (DATA(17),   GAMMA)
C
000010 J = 0
000011 WRITE (6,1) NTYPE, MODE, ININT
000013 1 FORMAT (1H0, 50X, 15H CONFIGURATION  , I5// 61X, 35H PROGRAM OPERA
     1TION MODE NUMBER =  , I5 // 40X, 31H NUMBER OF INTEGRATION STEPS =
     2   , I7 //)
000015 IF (NTYPE .GE. 1 .AND. NTYPE .LE. 7) GO TO 5
000018 WRITE (6,2)
000021 2 FORMAT (1H0, 39X, 40H CONFIGURATION NUMBER (NTYPE) NOT VALID  /)
000022 3 WRITE (6,4)
000025 4 FORMAT (1H0, 44X, 40H THIS CASE REJECTED BY SUBROUTINE DATACK  /)
000026 J = 1
000027 RETURN
000028 5 IF (ININT .GE. 10 .AND. ININT .LE. 50) GO TO 7
000031 IF (ININT .LT. 10) ININT = 10
000034 IF (ININT .GT. 50) ININT = 50
000037 WRITE (6,6) ININT
000039 6 FORMAT (1H0, 24X, 60H SUBROUTINE DATACK HAS CHANGED THE NUMBER OF
     1INTEGRATION STEPS TO  , I5 //)
000042 7 IF (MODE .GE. 1 .AND. MODE .LE. 2) GO TO 9
000045 WRITE (6,8)
000047 8 FORMAT (1H0, 30X, 41H PROGRAM OPERATION MODE NUMBER NOT VALID /)
000050 GO TO 3
000051 9 IF (ABS(GAMMA) .LT. 90.0) GO TO 11
000055 WRITE (6,10) GAMMA
000060 10 FORMAT (1H0, 37X, 18H DIHEDRAL ANGLE = , F14.7, 12H (NOT VALID)/)
000063 GO TO 3
000067 11 IF (NTYPE .EQ. 1 .OR. NTYPE .EQ. 3 .OR. NTYPE .EQ. 5 .OR. NTYPE
     A .EQ. 7 .OR. ETAIRQ .LE. ETAORQ) GO TO 13
000071 WRITE (6,12) ETAIRQ, ETAORQ
000100 12 FORMAT (1H0, 35X, 24H INBOARD SPAN STATION =  , F14.8 / 35X,
     A 24H OUTBOARD SPAN STATION =  , F14.8 // 42X, 35H THESE STATIONS A
     B NOT COMPATIBLE  //)
000103 GO TO 3
000105 13 IF (MODE .EQ. 2) RETURN
000108 WRITE (6,14)
000111 14 FORMAT (1H0, 23X, 72H INPUT POTENTIALS (CONSTANTS OF THE SCHWARZ-C
     1HRISTOFFEL TRANSFORMATION)  //)
000112 WRITE (6,15) A,B,C,D,E,F
000211 15 FORMAT (1H0,33X,3HA =,F15.8/51X,3HB =,F15.8/51X,3HC =,F15.8/51X,
     A 3HD =,F15.8/51X,3HE =,F15.8/51X,3HF =,F15.8)
000211 IF (NTYPE .EQ. 1 .OR. NTYPE .EQ. 3) GO TO 19
```

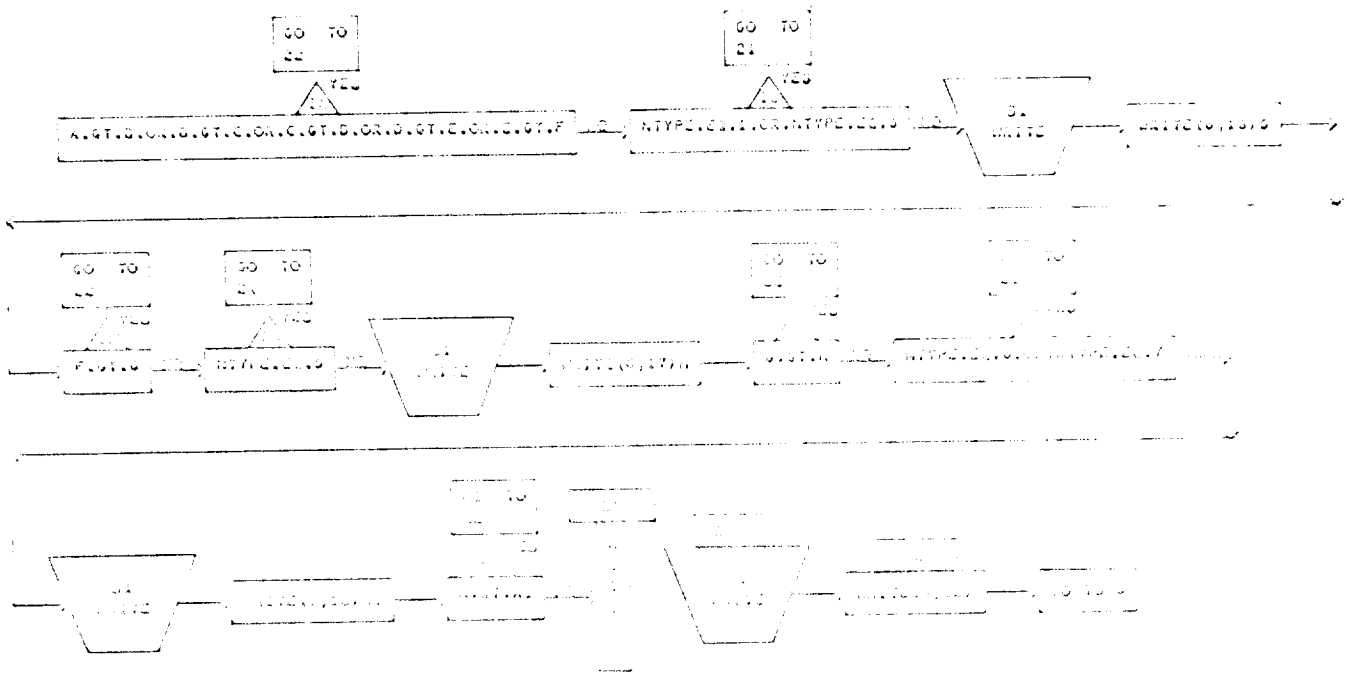
```
      WRITE (6,16) G
   16 FORMAT (1H , 50X,3HG =,F15.6)
      IF (NTYPE .EQ. 5) GO TO 19
      WRITE (6,17) H
   17 FORMAT (1H , 50X,3HH =,F15.6)
      IF (NTYPE .EQ. 6 .OR. NTYPE .EQ. 7) GO TO 19
      WRITE (6,18) AI
   18 FORMAT (1H , 50X,3HI =,F15.6)
   19 CALL TRANS
      WRITE (6,20)
   20 FORMAT (1H0, 30X, 41H TRANSFORMED POTENTIALS (LINEARLY SCALED) /)
      WRITE (6,15) A,B,C,D,E,F
      IF (A .GT. B .OR. B .GT. C .OR. C .GT. D .OR. D .GT. E .OR. E
     A      .GT. F) GO TO 22
      IF (NTYPE .EQ. 1 .OR. NTYPE .EQ. 3) GO TO 21
      WRITE (6,16) G
      IF (F .GT. G) GO TO 22
      IF (NTYPE .EQ. 5) GO TO 21
      WRITE (6,17) H
      IF (G .GT. H) GO TO 22
      IF (NTYPE .EQ. 6 .OR. NTYPE .EQ. 7) GO TO 21
      WRITE (6,18) AI
      IF (H .GT.AI) GO TO 22
   21 RETURN
   22 WRITE (6,23)
   23 FORMAT (1H0// 10X,110H INPUT VALUES OF POTENTIALS (CONSTANTS OF TRANSFORMATION
     AE SCHWARZ-CHRISTOFFEL TRANSFORMATION) ARE NOT MONOTONE INCREASING,
     B / 10X, 27H AS REQUIRED BY THE THEORY. /)
      GO TO 3
      END
```

```
SUBPROGRAM LENGTH
000042

FUNCTION ASSIGNMENTS

STATEMENT ASSIGNMENTS
        =  [illegible]  [illegible]  =  [illegible]  [illegible]  =  [illegible]  [illegible]  =  [illegible]
        =  [illegible]  [illegible]  =  [illegible]  [illegible]  =  [illegible]  [illegible]  =  [illegible]
        =  [illegible]  [illegible]  =  [illegible]  [illegible]  =  [illegible]  [illegible]  =  [illegible]
        =  [illegible]  [illegible]  =  [illegible]  [illegible]  =  [illegible]  [illegible]  =  [illegible]
        =  [illegible]  [illegible]  =  [illegible]  [illegible]  =  [illegible]  [illegible]  =  [illegible]
L1      =  000412  L2      =  000415  L3      =  [illegible]

BLOCK NAMES AND LENGTHS
        =  002420

VARIABLE ASSIGNMENTS
A       =  000008C01 AA      =  002217C01 ABSC    =  000021C01 AI      =  [illegible]
ALGND   =  000247C01 B       =  000001C01 C       =  000002C01 D       =  [illegible]
DATA    =  000000C01 E       =  000004C01 ETAING  =  000014C01 ETAORG  =  [illegible]
F       =  000005C01 G       =  000006C01 GAMMA   =  000020C01 H       =  [illegible]
ININT   =  000012C01 MODE    =  000011C01 NTYPE   =  000012C01 ORD     =  [illegible]
TAB     =  002257C01 TABLE   =  000165C01

START OF CONSTANTS
000425

START OF TEMPORARIES
000027

START OF INDIRECTS

          [illegible] SPACE
```

SUBROUTINE DATACK(J)




GO TO 13
YES
NTYPE.EQ.1.OR.NTYPE.EQ.5.OR.NTYPE.EQ.7.OR.ETAIRG.LE.ETAORG
NO
WRITE
WRITE(6,12)ETAIRG,ETAORG
GO TO 3
13
CONTINUE
MODE.EQ.2
NO
YES
RETURN
WRITE
WRITE(6,14)
WRITE
WRITE(6,15)A,B,C,D,E,F
GO TO 19
YES
NTYPE.EQ.1.OR.NTYPE.EQ.3
NO
WRITE
WRITE(6,16)G
GO TO 19
YES
NTYPE.EQ.3
NO
WRITE
WRITE(6,17)H
GO TO 19
YES
NTYPE.EQ.6.OR.NTYPE.EQ.7
NO
WRITE
WRITE(6,18)AI
PERFORM SUBROUTINE TRANS
WRITE
WRITE(6,20)
WRITE
WRITE(6,15)A,B,C,D,E,F



SYMBOLS USED IN SUBROUTINE DATACK (DECK IDENTIFICATION 55V2)

A	R	C	POTENTIAL AT A (MAPPING CONSTANT)	D	K
AA	R	C	ARRAY FOR RESULTS OF GEOMETRY AND DRAG INTEGRALS	D	K
ABSC	R	C	ABSCISSA ARRAY FOR NUMERICAL INTEGRALS	D	K
AI	R	C	POTENTIAL AT I (MAPPING CONSTANT)	D	K
ALOAD	R	C	STORAGE ARRAY FOR POTENTIAL IN THE REAL (PHYSICAL) PLANE	D	K
B	R	C	POTENTIAL AT B (MAPPING CONSTANT)	D	K
C	R	C	POTENTIAL AT C (MAPPING CONSTANT)	D	K
CG	R	C	COSINE OF DIHEDRAL ANGLE	D	K
CONI	R	C	FACTOR FOR IMAJINARY PART OF COMPLEX DISTANCE	D	K
CONR	R	C	FACTOR FOR REAL PART OF COMPLEX DISTANCE	D	K
C1	R	C	CONSTANT THAT IS COMPUTER HARDWARE DEPENDENT	D	K
C2	R	C	CONSTANT THAT IS COMPUTER HARDWARE DEPENDENT	D	K
C3	R	C	CONSTANT THAT IS COMPUTER HARDWARE DEPENDENT	D	K
D	R	C	POTENTIAL AT D (MAPPING CONSTANT)	D	K
DATA	R	C	ARRAY FOR INPUT DATA	D	K
E	R	C	POTENTIAL AT E (MAPPING CONSTANT)	D	K
ETAIRQ	R	C	INBOARD SPANWISE LOCATION (NONDIMENSIONAL)	D	K
ETAORQ	R	C	OUTBOARD SPANWISE LOCATION (NONDIMENSIONAL)	D	K
F	R	C	POTENTIAL AT F (MAPPING CONSTANT)	D	K
G	R	C	POTENTIAL AT G (MAPPING CONSTANT)	D	K
GAMMA	R	C	DIHEDRAL ANGLE IN DEGREES	D	K
GP	R	C	DIHEDRAL ANGLE DIVIDED BY PI	D	K
GPM2	R	C	GP - 0.5	D	K
GPP2	R	C	GP + 0.5	D	K
H	R	C	POTENTIAL AT H (MAPPING CONSTANT)	D	K
ILAST	I	C	FLAG FOR LAST ITERATION	D	K
ININT	I	C	NUMBER OF ORDINATES IN NUMERICAL INTEGRALS	D	K
J	I	U	ERROR FLAG TO HALT EXECUTION OF CASE	D	K
MODE	I	C	PROGRAM MODE OF OPERATION	D	K
NTYPE	I	C	CONFIGURATION TYPE	D	K
OMGP	R	C	1.0 - GP	D	K
ORD	R	C	ORDINATE ARRAY FOR NUMERICAL INTEGRALS	D	K
PIREQ	R	C	INBOARD LENGTH OR DISTANCE (NONDIMENSIONAL)	D	K
POREQ	R	C	OUTBOARD LENGTH OR DISTANCE (NONDIMENSIONAL)	D	K
R1	R	C	SQUARE ROOT OF SUM OF SQUARES OF ERRORS IN REQD CONDITIONS	D	K
TAB	R	C	INTERPOLATION ARRAY FOR INDEPENDENT VARIABLE AND FUNCTION	D	K
TABLE	R	C	INTEGRATED VALUES OF ORDINATES AS A FUNCTION OF ABSCISSA	D	K

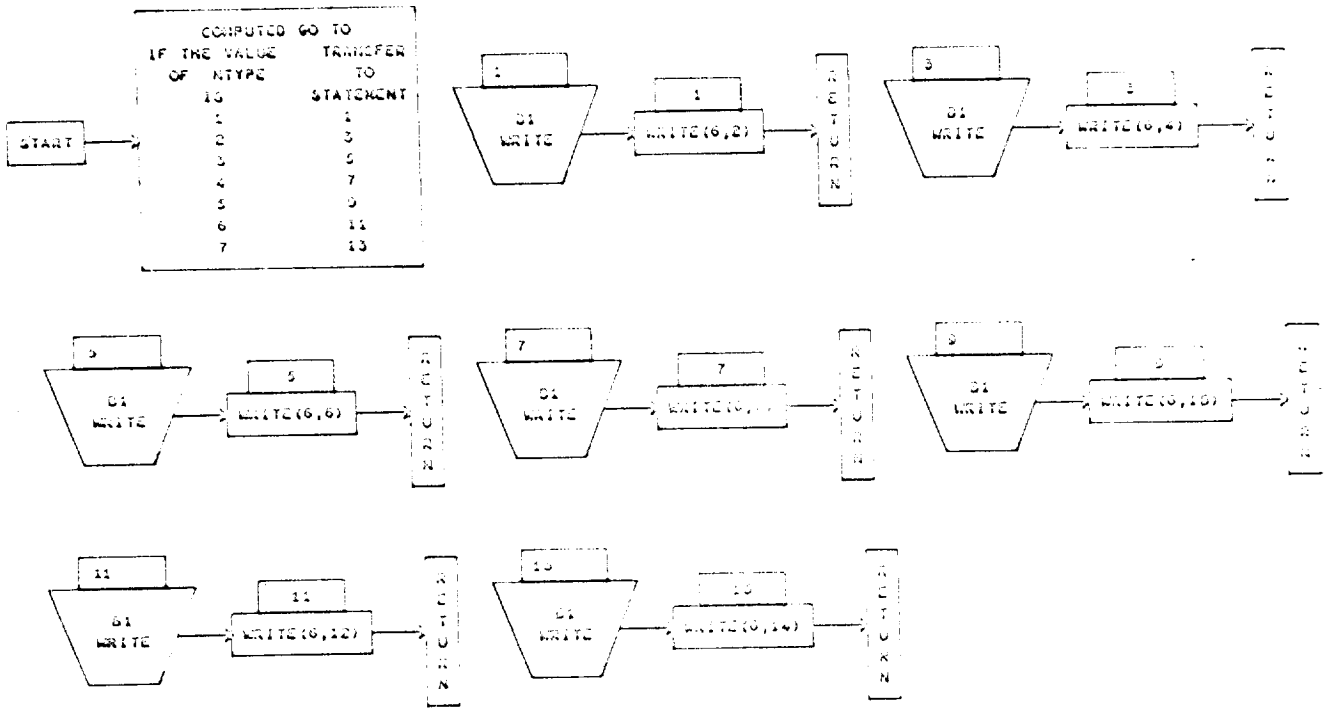
Subroutine HEADER (Deck 55UK). A column header is printed by this subroutine for the data that summarize the iteration for the mapping constants.

```
      SUBROUTINE HEADER
C
C     THIS SUBROUTINE PRINTS A HEADER FOR THE ITERATION INDEPENDENT
C     VARIABLES AND THE DEPENDENT REQUIRED CONDITIONS.
C
      COMMON DATA(17), ADOC(50), ORD(50), TABLE(50), ALOAD(1500), AA(10)
     A , TAU(10), GP, GPH2, GPP2, CG, GMGP, CONR, CONI, ILAST, C1, C2
     B , CS, R1
C
      EQUIVALENCE              (DATA( 1),       A) , (DATA( 2),       B)
     A , (DATA( 3),       C) , (DATA( 4),       D) , (DATA( 5),       E)
     B , (DATA( 6),       F) , (DATA( 7),       G) , (DATA( 8),       H)
     C , (DATA( 9),      AC) , (DATA(10),    MODE) , (DATA(11),   NTYPE)
     D , (DATA(12),   INIAT) , (DATA(13), ETAIRC) , (DATA(14), ETAORC)
     E , (DATA(15),   PIREQ) , (DATA(16),   POREQ) , (DATA(17),   GAMMA)
C
C     TRANSFER TO OUTPUT STATEMENT FOR SPECIFIC CONFIGURATION
C
      GO TO (1, 3, 5, 7, 9, 11, 13) , NTYPE
    1 WRITE (6,2)
    2 FORMAT (1H1, 45X, 59H WING WITH ONE PYLON (CONFIGURATION 1)  //
     A  65X, 31H ERRORS IN REQUIRED CONDITIONS  /
     B / 4X, 49H TRANSFORMATION CONSTANTS (INDEPENDENT VARIABLES)  ,
     C  65X, 13H SQUARE ROOT  / 70X, 6H PYLON , 7X, 6H WING , 21(6X,
     D  5HPYLON ) , 6X, 10H OF SUM OF / 4X, 1HA, 8X, 1HB, 12X, 1HC, 12X
     E  , 1HD, 8X, 1HE, 8X, 1HF, 8X, 21(6X, 7HCLOSURE ) , 8X, 8HLOCATION
     F  , 6X, 6HLENGTH , 5X, 11HSQR. ERRORS  //)
      RETURN
    3 WRITE (6,4)
    4 FORMAT (1H1, 45X, 45H WING WITH TWO PYLONS BELOW (CONFIGURATION 2)
     A // 65X, 31H ERRORS IN REQUIRED CONDITIONS  /
     B    / 5X, 59H TRANSFORMATION CONSTANTS (INDEPENDENT VARIABLES) ,
     C 4X, 17H INBOARD OUTBOARD , 9X, 21(1H  INBOARD OUTBOARD ) ,
     D 11HSQUARE ROOT  / 60X, 6H PYLON , 2X, 6H PYLON , 3X, 6H WING ,
     E  6X, 5H, 6H PYLON ) , 11H  OF SUM OF  / 5X, 1HA, 6X, 1HB, 6X, 1HC
     F  4X, 1HD, 5X, 1HE, 6X, 1HF, 6X, 1HG, 5X, 1HH, 6X, 1HI, 2X,
     G 5HH  CLOSURE ) , 21( 6H LOCATION ) , 21( 6H  LENGTH  ) ,
     H  11HSQR. ERRORS  //)
      RETURN
    5 WRITE (6,6)
    6 FORMAT (1H1, 55X, 65H WING WITH HORIZONTAL TAIL (BIPLANE CASE, CONS
     AFIGURATION 3)  // 65X, 31H ERRORS IN REQUIRED CONDITIONS  //
     B  7HA, 5X, 5HA (HORIZONTAL )                                  / 6X,
     C 4HA TRANSFORMATION CONSTANTS (INDEPENDENT VARIABLES)  , 19X,
     D  5X, 4HA(TAIL) , 6X, 11HSQUARE ROOT  / 73X, 4HWING , 5X,
     E  5X, 10H(2ND WING) )  , 4X,  5HOF SUM OF / 8X, 1HA, 12X, 1HB,
     F  6X, 1HC, 6X, 1HD, 6X, 1HE, 6X, 1HF, 6X, 21( 6X, 7HCLOSURE ) ,
     G  7X, 6HWEIGHT , 7X, 6HLENGTH , 5X, 11HSQR. ERRORS  //)
      RETURN
    7 WRITE (6,8)
    8 FORMAT (1H1, 55X, 65H WING WITH ONE PYLON ABOVE AND ONE PYLON BELO
     AW (CONFIGURATION 4)  // 65X, 31H ERRORS IN REQUIRED CONDITIONS //
     B                          5X, 49H TRANSFORMATION CONSTANTS (INDEPEN
     CDENT VARIABLES)  , 6X, 7HINBOARD , 1X, 8HOUTBOARD, 9X, 21( 2X,
     D  7HINBOARD, 1X, 8HOUTBOARD ) , 11HSQUARE ROOT / 57X, 21(4X, 5HPYLO
     EN ) , 4X, 5HWING  , 11(4X,5HPYLON ) , 2X, 9HOF SUM OF  / 5X, 1HA,
     F 6X, 1HB, 6X, 1HC, 4X, 1HD, 5X, 1HE, 6X, 1HF, 4X, 1HG, 5X, 1HH,
```

```
      G 6X, 1HI, 2X, 3( 2X, 7HCLOSURE ) ,        2( 1X, 8HLOCATION ) ,
      H     2 (9H  LENGTH  ) ,    11HSQR. ERRORS  //)
000040   RETURN
000041 9 WRITE (6,10)
000045 10 FORMAT (1H1, 15X, 99H WING WITH BREAK STATION, PYLON AT THE BREAK,
      A AND DIHEDRAL OUTBOARD OF THE BREAK (CONFIGURATION 5)  // 65X,
      B  31H ERRORS IN REQUIRED CONDITIONS  //
      C  7X, 49H TRANSFORMATION CONSTANTS (INDEPENDENT VARIABLES)  , 9X,
      D  8H INBOARD , 14X, 9H OUTBOARD , 13X, 5HPYLON , 5X, 11HSQUARE ROO
      ET  / 61X, 2( 7X, 4HWING, 6X, 5HPYLON ) , 5X, 8HSPANWISE , 4X,
      F  9HOF SUM OF  / 6X, 1HA, 7X, 1HB, 6X, 1HC, 10X, 1HD, 7X, 1HE,
      G  6X, 1HF, 10X, 1HG, 3X, 3( 4X, 7HCLOSURE ) , 5X, 6HLENGTH , 4X,
      H  8HLOCATION  , 3X, 11HSQR. ERRORS //)
000045   RETURN
000046 11 WRITE (6,12)
000052 12 FORMAT (1H1,  4X,124H WING WITH TWO BREAK STATIONS, PYLON AT THE
      ANBOARD STATION, AND DIHEDRAL OUTBOARD OF THE OUTBOARD STATION (CON
      FIGURATION 6)  // 65X, 31H ERRORS IN REQUIRED CONDITIONS  //
      C                          9X, 49H TRANSFORMATION CONSTANTS (INDEPEND
      DENT VARIABLES)  ,  9X, 7HINBOARD , 10X, 6HOUTBOARD , 10X,
      E   8HDIHEDRAL , 11X, 11HSQUARE ROOT / 64X, 2(4X, 4HWING, 5X,
      F  5HPYLON ) , 4X, 5HBREAK, 4X, 5HPYLON, 4X, 9HOF SUM OF / 6X, 1HA
      G  , 6X, 1HB, 6X, 1HC, 8X, 1HD, 6X, 1HE, 6X, 1HF, 6X, 1HG, 6X, 1HH
      H  , 3X, 3( 2X, 7HCLOSURE ) ,      2(4X, 5HLOC.  ) , 2X, 6HLENGTH ,
      I  3X, 11HSQR. ERRORS  //)
000052   RETURN
000053 13 WRITE (6,14)
000057 14 FORMAT (1H1, 30X, 69H WING WITH AN END-PLATED VERTICAL FENCE AT THE
      AE TIP (CONFIGURATION 7)  // 65X, 30H ERRORS IN REQUIRED CONDITIONS
      B  //                          9X, 49H TRANSFORMATION CONSTANTS (INDE
      CEPENDENT VARIABLES)  , 17X, 8HVERTICAL , 4X, 3HEND , 3X, 8HVERTICA
      DL , 2X, 7HINBOARD, 1X, 6HOUTBOARD , 2X, 11HSQUARE ROOT  / 65X,
      E  4HWING , 5X, 5HFENCE , 4X, 5HPLATE , 4X, 5HFENCE , 2 ( 4X,
      F  5HE. P. ) , 4X, 9HOF SUM OF  / 4X, 1HA, 6X, 1HB, 6X, 1HC, 6X,
      G  1HD, 6X, 1HE, 6X, 1HF, 6X, 1HG, 6X, 1HH, 3X, 3(2X, 7HCLOSURE ),
      H       3( 2X, 7HLENGTH  ) , 2X, 11HSQR. ERRORS  //)
000057   RETURN
000060   END
```

```
SUBPROGRAM LENGTH
000070

FUNCTION ASSIGNMENTS

STATEMENT ASSIGNMENTS
1      -  000010    2      -  000004    3      -  000022    4      -  000141
5      -  000027    6      -  [illegible]    7      -  000034    8      -  000310
9      -  000041    10     -  [illegible]    11     -  000046    12     -  000002
13     -  000033    14     -  000001

BLOCK NAMES AND LENGTHS
       -  002420

VARIABLE ASSIGNMENTS
[illegible]  -  [illegible] ABSC   -  [illegible] ALOAD  -  [illegible] DATA   -  [illegible]
NTYPE  -  [illegible] GND    -  [illegible] TAB    -  [illegible] TABLE  -  [illegible]

START OF CONSTANTS
000002

START OF TEMPORARIES
000074

START OF INDIRECTS
000070

UNUSED COMPILER SPACE
041000
```

SUBROUTINE HEADER



SYMBOLS USED IN SUBROUTINE HEADER (DECK IDENTIFICATION 55V0)

A	R	C	POTENTIAL AT A (MAPPING CONSTANT)	HEADER
AA	R	C	ARRAY FOR RESULTS OF GEOMETRY AND DRAG INTEGRALS	HEADER
ABSC	R	C	ABSCISSA ARRAY FOR NUMERICAL INTEGRALS	HEADER
AI	R	C	POTENTIAL AT I (MAPPING CONSTANT)	HEADER
ALCAD	R	C	STORAGE ARRAY FOR POTENTIAL IN THE REAL (PHYSICAL) PLANE	HEADER
B	R	C	POTENTIAL AT B (MAPPING CONSTANT)	HEADER
C	R	C	POTENTIAL AT C (MAPPING CONSTANT)	HEADER
CG	R	C	COSINE OF DIHEDRAL ANGLE	HEADER
CCNI	R	C	FACTOR FOR IMAJINARY PART OF COMPLEX DISTANCE	HEADER
CCNR	R	C	FACTOR FOR REAL PART OF COMPLEX DISTANCE	HEADER
C1	R	C	CONSTANT THAT IS COMPUTER HARDWARE DEPENDENT	HEADER
C2	R	C	CONSTANT THAT IS COMPUTER HARDWARE DEPENDENT	HEADER
C3	R	C	CONSTANT THAT IS COMPUTER HARDWARE DEPENDENT	HEADER
D	R	C	POTENTIAL AT D (MAPPING CONSTANT)	HEADER
DATA	R	C	ARRAY FOR INPUT DATA	HEADER
E	R	C	POTENTIAL AT E (MAPPING CONSTANT)	HEADER
ETAIBD	R	C	INBOARD SPANWISE LOCATION (NONDIMENSIONAL)	HEADER
ETAOBD	R	C	OUTBOARD SPANWISE LOCATION (NONDIMENSIONAL)	HEADER
F	R	C	POTENTIAL AT F (MAPPING CONSTANT)	HEADER
G	R	C	POTENTIAL AT G (MAPPING CONSTANT)	HEADER
GAMMA	R	C	DIHEDRAL ANGLE IN DEGREES	HEADER
GP	R	C	DIHEDRAL ANGLE DIVIDED BY PI	HEADER
GPM2	R	C	GP - 0.5	HEADER
GPP2	R	C	GP + 0.5	HEADER
H	R	C	POTENTIAL AT H (MAPPING CONSTANT)	HEADER
ILAST	I	C	FLAG FOR LAST ITERATION	HEADER
ININT	I	C	NUMBER OF ORDINATES IN NUMERICAL INTEGRALS	HEADER
MODE	I	C	PROGRAM MODE OF OPERATION	HEADER
NTYPE	I	C	CONFIGURATION TYPE	HEADER
OMGP	R	C	1.0 - GP	HEADER
ORD	R	C	ORDINATE ARRAY FOR NUMERICAL INTEGRALS	HEADER
PIREQ	R	C	INBOARD LENGTH OR DISTANCE (NONDIMENSIONAL)	HEADER
POREQ	R	C	OUTBOARD LENGTH OR DISTANCE (NONDIMENSIONAL)	HEADER
R1	R	C	SQUARE ROOT OF SUM OF SQUARES OF ERRORS IN REQD CONDITIONS	HEADER
TAB	R	C	INTERPOLATION ARRAY FOR INDEPENDENT VARIABLE AND FUNCTION	HEADER
TABLE	R	C	INTEGRATED VALUES OF ORDINATES AS A FUNCTION OF ABSCISSA	HEADER

[illegible] for [illegible] parameter. The origin of each cycle of the iteration is a statement CALL GEOM (A1), which places the errors in the required conditions in the A1 array for the current values of the mapping constants. The error parameter $\bar{e}$ is computed and the iteration cycle summary data is printed. If the iteration completion parameter ILAST is 1, return is made to the executive control subroutine SERIES. If ILAST is not 1, the current value of $\bar{e}$ is tested for convergence ($\bar{e} \leq 3 \times 10^{-8}$). The parameter ILAST is set to 1 if convergence has occurred and transfer is made to the origin of the cycle. If convergence has not occurred, the current values of the mapping constants are temporarily stored in the P1 array. The correction matrix calculation parameter JUMP is set to 1 (calculate a new correction matrix $\partial C_i/\partial p_j$) if any of the following conditions are satisfied: the magnitude of the error in any of the required geometry conditions exceeds a maximum value — either 0.03 or 0.04, depending on the configuration; the parameter ICOUNT is 0; or the current value of the error parameter $\bar{e}$ (R1) is greater than a definite fraction of the value (R2) from the preceding cycle and ICOUNT is greater than 4. The fraction is either 0.5 or 0.7, depending on the configuration. The current value of $\bar{e}$ is compared with the smallest value (RBEST) from preceding cycles; if R1 is less than RBEST, RBEST is set to R1, and the current values of the mapping constants are stored in the array BEST. The counter for the number of iteration cycles ICOUNT is incremented, and the counter for the number of influence matrices MCOUNT is incremented if JUMP is equal to 1. If ICOUNT > 40 or MCOUNT > 4, the mapping constants are set to the values producing the smallest $\bar{e}$ from the preceding cycles, ILAST is set to 1, and transfer is made to the origin of the iteration cycle. If neither the ICOUNT nor the MCOUNT conditions are satisfied, the influence matrix calculation parameter JUMP is tested for 0. If JUMP is not 0, a new influence matrix $\partial C_i/\partial p_j$ is calculated and stored in both CHAN and DUMB arrays. If JUMP is 0, the influence matrix from a preceding cycle that is stored in array DUMB is transferred into array CHAN, and the corrections Δp_j for the mapping constants are calculated by subroutine INVERT. The mapping constants are corrected ($p_{j+1} = p_j + \Delta p_j$), are made monitonically increasing by subroutine COMPAT, and are scaled linearly by subroutine TRANS. Transfer is then made to the origin of the iteration cycle.

```
      SUBROUTINE INNER                                                  55VL0010
C                                                                       55VL0020
C     THIS SUBROUTINE OPERATES THE        ITERATION FOR THE UNKNOWN     55VL0030
C     INDEPENDENT PARAMETERS, THE SCHWARZ-CHRISTOFFEL TRANSFORMATION    55VL0040
C     MAPPING CONSTANTS.  IT CORRESPONDS TO THE MAIN PROGRAM DECK OF    55VL0050
C     PROGRAM 55VC.                                                     55VL0060
C                                                                       55VL0070
      DIMENSION          A1(10), ABSERR(4), DATA(7), A4(10), A5(10), A6(10)55VL0080
     1 , A7(10), A8(10), CHARA(7), CHAR(7), P1(6), BEST(6)              55VL0090
C                                                                       55VL0100
      COMMON DATA(7), ABSERR, CHARA, TWO, THREE, ZERO, ONE, SMALL, ONE1, 55VL0110
     1 , THREE1, OP, CHAR, CHAR1, SIG, ONEP, ONEM, CONST, ICNOT, CTEST1, 55VL0120
     2 , SIG1, SIG2, N1                                                 55VL0130
C                                                                       55VL0140
      EQUIVALENCE                (DATA(1), A1)       , (DATA(2), A2)    55VL0150
     1 , (DATA(3), A3)       , (DATA(4), A4)       , (DATA(5), A5)     55VL0160
     2 , (DATA(6), A6)       , (DATA(7), A7)       , (CHARA(1), A8)    55VL0170
     3 , (CHARA(2), ALPHA)   , (CHARA(3), BETA)    , (CHARA(4), GAMMA) 55VL0180
     4 , (CHARA(5), ETAONE)  , (CHARA(6), ETATWO)  , (CHARA(7), ETATHR) 55VL0190
     5 , (DATA(1), PIONE)    , (DATA(2), PITWO)    , (DATA(3), PITHR)  55VL0200
     6 , (DATA(4), PIFOR)    , (DATA(5), PIFIV)    , (DATA(6), PISIX)  55VL0210
     7 , (DATA(7), PISEV)    , (CHARA(1), CHARA)                       55VL0220
      EQUIVALENCE (A4(1), P1)                                           55VL0230
C                                                                       55VL0240
C     THE FOLLOWING EQUIVALENCE STATEMENTS ARE ADDED TO STORE STORAGE   55VL0250
C                                                                       55VL0260
      EQUIVALENCE                (A6(1), A1)       , (A6(2), A2)       55VL0270
     1 , (A6(3), A3)       , (A6(4), A4)       , (A6(5), A5)           55VL0280
     2 , (A6(6), A6)       , (A6(7), A7)       , (A6(8), A8)           55VL0290
     3 , (A6(9), ABSERR)   , (A6(10), N1)                              55VL0300
C                                                                       55VL0310
      DATA P1, SIGMA / 3.1415926535897932384626433832795 /              55VL0320
C                                                                       55VL0330
      ICNOT = 0                                                         55VL0340
      ICOUNT = 0                                                        55VL0350
      ICOUNT = 0                                                        55VL0360
C                                                                       55VL0370
C     GO BACK TO COMPUTE FOR UNSCALED COORDINATES                       55VL0380
C                                                                       55VL0390
      GO TO 11                                                          55VL0400
C                                                                       55VL0410
      WRITE A1 ON THE PLOTTER OR VERTICAL PLANE                         55VL0420
C                                                                       55VL0430
   11 CALL SPUT (A1)                                                    55VL0440
      R1 = ABSERR(4)                                                    55VL0450
      WRITE (6,101) (DATA(I),I=1,6) , (A1(J),J=1,4) , R1                55VL0460
  101 FORMAT (1H , F5.1, 3F13.6, F5.1, F13.6, 3X, 4F13.6, F14.6)        55VL0470
      IF (ILAST .NE. 1) GO TO 10                                        55VL0480
      A1(3) = A1(3) + ETAONE                                            55VL0490
      A1(4) = A1(4) + PIONE                                             55VL0500
      RETURN                                                            55VL0510
   10 IF (N1 .GT. CTEST1) GO TO 13                                      55VL0520
      ILAST = 1                                                         55VL0530
      GO TO 11                                                          55VL0540
   13 DO 14 I=1,6                                                       55VL0550
      IF (ICOUNT .EQ. 0) BEST(I) = DATA(I)                              55VL0560
   14 P1(I) = DATA(I)                                                   55VL0570
      JUMP = 0
```

```
      DO 19 I=1,4
      IF (ABS(A1(I)) .GT. 0.03) JUMP = 1
   19 CH1(I) = (-A1(I))
      IF (ICOUNT .EQ. 0) RBEST = R1
      IF (R1 .GE. RBEST) GO TO 103
      DO 102 I=1,6
  102 BEST(I) = DATA(I)
      RBEST = R1
  103 IF (ICOUNT .EQ. 0) JUMP = 1
      IF (ICOUNT .GT.  4 .AND. R1 .GT. (0.5 * R2)) JUMP = 1
      R2 = R1
      ICOUNT = ICOUNT + 1
      IF (JUMP .EQ. 1) MCOUNT = MCOUNT + 1
      IF (ICOUNT .LE. 40 .AND. MCOUNT .LE. 4) GO TO 131
      DO 121 I=1,6
  121 DATA(I) = BEST(I)
      ILAST = 1
      GO TO 11
  131 IF (JUMP .EQ. 0) GO TO 20
      DELA = 0.02 * (B - A)
      A = A - DELA
      CALL TRANS
      CALL GEOM (A2)
      DO 15 I=1,6
   15 DATA(I) = P1(I)
      B = B - DELA
      CALL GEOM (A3)
      B = B + DELA
      DELD = 0.02 * (E - D)
      D = D + DELD
      CALL GEOM(A4)
      D = D - DELD
      DELF = 0.02 * (F - E)
      F = F + DELF
      CALL GEOM(A5)
      F = F - DELF
      DO 16 I=1,4
      CHAN(I,1) = (A2(I) - A1(I)) / (-DELA)
      CHAN(I,2) = (A3(I) - A1(I)) / (-DELA)
      CHAN(I,3) = (A4(I) - A1(I)) / DELD
      CHAN(I,4) = (A5(I) - A1(I)) / DELF
      DO 16 J=1,4
   16 DUMB(I,J) = CHAN(I,J)
      GO TO 22
   20 DO 21 I=1,4
      DO 21 J=1,4
   21 CHAN(I,J) = DUMB(I,J)
   22 CALL INVERT (JUMP, 4)
      A = A + CH1(1)
      B = B + CH1(2)
      D = D + CH1(3)
      F = F + CH1(4)
      CALL COMPAT
      CALL TRANS
      GO TO 11
C
C     WING WITH TWO PYLONS OR VERTICAL FENCES
C
```

```
000[illegible]    31 CALL GEOM (A1)                                          55VL1100
000344               R1 = RSSQRR(7)                                          55VL1110
000347               WRITE (6, 29) (DATA(I),I=1,9) , (A1(J),J=1,7) , R1      55VL1120
000370            29 FORMAT (1H , 3F7.3, 2(F4.1, 2F7.3), 7F9.5, F11.6)       55VL1130
000376               IF (ILAST .NE. 1) GO TO 27                              55VL1200
000372               A1(4) = A1(4) + ETAIRQ                                  55VL1210
000374               A1(5) = A1(5) + ETAORQ                                  55VL1220
000376               A1(6) = A1(6) + PIREQ                                   55VL1230
000400               A1(7) = A1(7) + POREQ                                   55VL1240
000402               RETURN                                                  55VL1250
000403            27 IF (R1 .GT. GTEST1) GO TO 34                            55VL1260
000407               ILAST = 1                                               55VL1270
000410               GO TO 31                                                55VL1280
000410            34 DO 35 I=1,9                                             55VL1290
000411               IF (ICOUNT .EQ. 0) DEST(I) = DATA(I)                    55VL1300
000416            35 P1(I) = DATA(I)                                         55VL1310
000423               JUMP = 0                                                55VL1320
000424               DO 39 I=1,7                                             55VL1330
000426               IF (ABS(A1(I)) .GT. 0.03) JUMP = 1                      55VL1340
000433            39 CH1(I) = (-A1(I))                                       55VL1350
000441               IF (ICOUNT .EQ. 0) RBEST = R1                           55VL1360
000443               IF (R1 .GE. RBEST) GO TO 105                            55VL1370
000446               DO 104 I=1,9                                            55VL1380
000447           104 DEST(I) = DATA(I)                                       55VL1390
000454               RBEST = R1                                              55VL1400
000455           105 IF (ICOUNT .EQ. 0) JUMP = 1                             55VL1410
000460               IF (ICOUNT .GT.  4 .AND. R1 .GT. (0.7 * R2)) JUMP = 1   55VL1420
000474               R2 = R1                                                 55VL1430
000475               ICOUNT = ICOUNT + 1                                     55VL1440
000476               IF (JUMP .EQ. 1) MCOUNT = MCOUNT + 1                    55VL1450
000501               IF (ICOUNT .LE. 40 .AND. MCOUNT .LE. 4) GO TO 132       55VL1460
000510               DO 122 I=1,9                                            55VL1470
000[illegible]   122 DATA(I) = BEST(I)                                       55VL1480
000521               ILAST = 1                                               55VL1490
000522               GO TO 31                                                55VL1500
000523           132 IF (JUMP .EQ. 0) GO TO 40                               55VL1510
000[illegible]       DELA = 0.02 * (B - A)                                   55VL1520
0[illegible]         A = A + DELA                                            55VL1530
0[illegible]         CALL GEOM (A2)                                          55VL1540
000[illegible]       A = A - DELA                                            55VL1550
000[illegible]       B = B - DELA                                            55VL1560
000[illegible]       CALL GEOM (A3)                                          55VL1570
000[illegible]       B = B + DELA                                            55VL1580
000[illegible]       DELD = 0.02 * (D - C)                                   55VL1590
000[illegible]       D = D - DELD                                            55VL1600
000[illegible]       CALL TRANS                                              55VL1610
000[illegible]       CALL GEOM (A4)                                          55VL1620
000[illegible]       DO 36 I=1,9                                             55VL1630
0[illegible]      36 DATA(I) = P1(I)                                         55VL1640
000[illegible]       DELE = 0.02 * (F - E)                                   55VL1650
000[illegible]       E = E + DELE                                            55VL1660
000[illegible]       CALL GEOM (A5)                                          55VL1670
000[illegible]       E = E - DELE                                            55VL1680
000[illegible]       DELF = 0.02 * (G - F)                                   55VL1690
000[illegible]       F = F + DELF                                            55VL1700
000575               CALL GEOM (A6)                                          55VL1710
000[illegible]       F = F - DELF                                            55VL1720
000577               G = G + DELF                                            55VL1730
```

```
000001          CALL TRANS
000002          CALL GEOM (A7)
000004          DO 37 I=1,9
000006       37 DATA(I) = A1(I)
000015          DELH = 0.02 * (A1 - H)
000016          H = H + DELH
000037          CALL GEOM (A0)
000062          H = H - DELH
000081          DO 38 I=1,7
000025          CHAN(I,1) = (A2(I) - A1(I)) / DELH
000033          CHAN(I,2) = (A3(I) - A1(I)) / (-DELH)
000037          CHAN(I,3) = (A4(I) - A1(I)) / (-DELH)
000045          CHAN(I,4) = (A5(I) - A1(I)) / DELH
000052          CHAN(I,5) = (A6(I) - A1(I)) / DELH
000056          CHAN(I,6) = (A7(I) - A1(I)) / DELH
000065          CHAN(I,7) = (A8(I) - A1(I)) / DELH
000067          DO 39 J=1,7
000070       39 DUMB(I,J) = CHAN(I,J)
000074          GO TO 42
000084       40 DO 41 I=1,7
000086          DO 41 J=1,7
000087       41 CHAN(I,J) = DUMB(I,J)
000088       42 CALL INVERT (COMP, 7)
000090          A = A + CH1(1)
000097          B = B + CH1(2)
000101          D = D + CH1(3)
000102          E = E + CH1(4)
000106          F = F + CH1(5)
000107          G = G + CH1(6)
000108          H = H + CH1(7)
000109          CALL COMPUT
000110          CALL TRANS
000112          GO TO 31
C
C       WRG WITH HORIZONTAL TAIL (COMPLANAR). THIS CODE WAS SUGGESTED BY
C       E. S. ROTOSKAY OF THE GOODYEAR AIRCRAFT COMPANY, INC.
C
000116       49 CALL GEOM (A1)
000117          R1 = ABS(A1(1))
000122          WRITE (6,102) (A1(I), I=1,6), R1
000124      102 FORMAT (1H , 2F15.4, 4F15.4, F15.4, 3X, 4F15.4, F14.4)
000126          IF (ILAST .NE. 1) GO TO 52
000130          A1(3) = A1(3) + 2*PIRAD
000140          A1(4) = A1(4) + PIRAD
000142          RETURN
000143       52 IF (R1 .GT. GTEST1) GO TO 54
000147          ILAST = 1
000150          GO TO 49
000150       54 DO 55 I=1,6
000152          IF (ICOUNT .EQ. 0) BEST(I) = DATA(I)
000156       55 P1(I) = DATA(I)
000165          JUMP = 0
000166          DO 58 I=1,4
000170          IF (ABS(A1(I)) .GT. 0.03) JUMP = 1
000173       58 CH1(I) = (-A1(I))
000181          IF (ICOUNT .EQ. 0) RBEST = R1
000183          IF (R1 .GE. RBEST) GO TO 107
000186          DO 106 I=1,6
```

```
  100 BEST(I) = DATA(I)
      BEST = R1
  107 IF (ICOUNT .EQ. 0) JUMP = 1
      IF (ICOUNT .GT. 4 .AND. R1 .GT. (0.5 * R2)) JUMP = 1
      R2 = R1
      ICOUNT = ICOUNT + 1
      IF (JUMP .EQ. 1) MCOUNT = MCOUNT + 1
      IF (ICOUNT .LE. 40 .AND. MCOUNT .LE. 4) GO TO 133
      DO 123 I=1,6
  123 DATA(I) = BEST(I)
      ILAST = 1
      GO TO 49
  133 IF (JUMP .EQ. 0) GO TO 59
      DELA = 0.02 * (A - B)
      B = B + DELA
      CALL GEOM (A2)
      B = B - DELA
      DELD = 0.02 * (D - C)
      D = D - DELD
      CALL GEOM (A3)
      D = D + DELD
      DELE = 0.02 * (E - D)
      E = E + DELE
      CALL TRANS
      CALL GEOM (A4)
      DO 62 I=1,6
   62 DATA(I) = P1(I)
      F = F + DELD
      CALL GEOM (A5)
      DO 56 I=1,6
   56 DATA(I) = P1(I)
      DO 57 I=1,6
      CHAN(I,1) = (A1(I) - A1(I)) / DELD
      CHAN(I,2) = (A5(I) - A1(I)) / (DELD)
      CHAN(I,3) = (A4(I) - A1(I)) / DELE
      CHAN(I,4) = (A5(I) - A1(I)) / DELD
      DO 57 J=1,4
   57 DUMB(I,J) = CHAN(I,J)
      GO TO 61
   59 DO 60 I=1,4
      DO 60 J=1,4
   60 CHAN(I,J) = DUMB(I,J)
   61 CALL INVERT (JUMP, 4)
      B = B + CH1(1)
      D = D + CH1(2)
      E = E + CH1(3)
      F = F + CH1(4)
      CALL COMPUT
      CALL TRANS
      GO TO 49
C
C     WING WITH PYLON ON TOP AND ON BOTTOM.  THIS CASE WAS SUGGESTED BY
C     DR. RICHARD WHITCOMB OF NASA LANGLEY RESEARCH CENTER.
C
   69 C     GEOM (A1)
      R1 = RSSERR(7)
      WRITE (0,502) (DATA(I),I=1,9) , (A1(J),J=1,7) , R1
  502 FORMAT (1H , 3F7.3, 2(F4.1, 2F7.3), 7F9.5, F11.8)
```

```
      IF (ILAST .NE. 1) GO TO 72
      A1(4) = A1(4) + ETHORG
      A1(5) = A1(5) + ETHORG
      A1(6) = A1(6) + PIREG
      A1(7) = A1(7) + PORED
      RETURN
   72 IF (R1 .GT. GTEST1) GO TO 74
      ILAST = 1
      GO TO 69
   74 DO 75 I=1,9
      IF (ICOUNT .EQ. 0) BEST(I) = DATA(I)
   75 P1(I) = DATA(I)
      JUMP = 0
      DO 76 I=1,7
      IF (ABS(A1(I)) .GT. 0.03) JUMP = 1
   76 CH1(I) = (-A1(I))
      IF (ICOUNT .EQ. 0) RBEST = R1
      IF (R1 .GE. RBEST) GO TO 109
      DO 108 I=1,9
  108 BEST(I) = DATA(I)
      RBEST = R1
  109 IF (ICOUNT .EQ. 0) JUMP = 1
      IF (ICOUNT .GT. 4 .AND. R1 .GT. (0.5 * R2)) JUMP = 1
      R2 = R1
      ICOUNT = ICOUNT + 1
      IF (JUMP .EQ. 1) MCOUNT = MCOUNT + 1
      IF (ICOUNT .LE. 40 .AND. MCOUNT .LE. 4) GO TO 134
      DO 124 I=1,9
  124 DATA(I) = BEST(I)
      ILAST = 1
      GO TO 69
  134 IF (JUMP .EQ. 0) GO TO 79
      DELA = 0.01 * (B - A)
      A = A + DELA
      CALL GEOM (A2)
      A = A - DELA
      B = B - DELA
      CALL GEOM (A3)
      B = B + DELA
      DELC = 0.02 * (C - D)
      C = C - DELC
      CALL GEOM (A4)
      C = C + DELC
      DELD = 0.02 * (E - D)
      E = E - DELD
      CALL GEOM (A5)
      E = E + DELD
      DELG = 0.02 * (H - G)
      G = G + DELG
      CALL TRANS
      CALL GEOM (A6)
      DO 76 I=1,9
   76 DATA(I) = P1(I)
      DELH = 0.02 * (A1 - H)
      H = H + DELH
      CALL GEOM (A7)
      H = H - DELH
      A1 = A1 + DELH
```

```
      DO 110 I=1,7
  110 BEST(I) = DATA(I)
      RBEST = R1
  111 IF (ICOUNT .EQ. 0) JUMP = 1
      IF (ICOUNT .GT.  4 .AND. R1 .GT. (0.5 * R2)) JUMP = 1
      R2 = R1
      ICOUNT = ICOUNT + 1
      IF (JUMP .EQ. 1) MCOUNT = MCOUNT + 1
      IF (ICOUNT .LE. 40 .AND. MCOUNT .LE. 4) GO TO 135
      DO 125 I=1,7
  125 DATA(I) = BEST(I)
      ILAST = 1
      GO TO 80
  135 IF (JUMP .EQ. 0) GO TO 95
      DELA = 0.01 * (B - A)
      A = A - DELA
      CALL GEOM (A2)
      A = A + DELA
      B = B - DELA
      CALL TRANS
      CALL GEOM (A3)
      DO 83 I=1,7
   83 DATA(I) = P1(I)
      DELC = 0.01 * (D - C)
      D = D - DELC
      CALL GEOM (A4)
      D = D + DELC
      DELE = 0.01 * (F - E)
      E = E + DELE
      CALL TRANS
      CALL GEOM (A5)
      DO 93 I=1,7
   93 DATA(I) = P1(I)
      DELG = 0.01 * (G - F)
      G = G + DELG
      CALL GEOM (A6)
      G = G - DELG
      DO 94 I=1,5
      CHAN(I,1) = (A2(I) - A1(I)) / (-DELA)
      CHAN(I,2) = (A3(I) - A1(I)) / (-DELA)
      CHAN(I,3) = (A4(I) - A1(I)) / (-DELC)
      CHAN(I,4) = (A5(I) - A1(I)) / DELE
      CHAN(I,5) = (A6(I) - A1(I)) / DELG
      DO 94 J=1,5
   94 SUMD(I,J) = CHAN(I,J)
      GO TO 112
   95 DO 114 I=1,5
      DO 114 J=1,5
  114 CHAN(I,J) = SUMD(I,J)
  112 CALL INVERT (JUMP, 5)
      A = A + CH1(1)
      B = B + CH1(2)
      D = D + CH1(3)
      E = E + CH1(4)
      G = G + CH1(5)
      CALL COMPAT
      CALL TRANS
      GO TO 80
```

```
C
C     USING A TWO TERM FOURIER AND EXPONENTIAL DISCRETIZATION, THIS CODE WAS
C     DEVELOPED BY CHARLES HARRIS OF NASA LANGLEY RESEARCH CENTER.
C
  100 [illegible] = GAMMA / 180.0
      [illegible] = [illegible] * [illegible]
      [illegible] = [illegible] * [illegible]
      CG = COS (DEGRAD * GAMMA)
      CGNT = CG
      SGNR = SIN(DEGRAD * GAMMA)
  101 CALL GEOM [illegible]
      [illegible] = ASSEM [illegible]
      [illegible] (DATA(I), I=1,[illegible]) * [illegible] [illegible] [illegible]
  102 FORMAT (1H [illegible] F9.6, F9.6, F9.6, F9.6, F9.6, F9.6, F9.6, F9.6)
      IF (DELAST EQ. 1) GO TO 104
      A(1) = A(1) * STRETC
      A(2) = A(2) * STRETC
      A(3) = A(3) * PIREC
      [illegible]
  104 IF (DI .GT. DTEST) GO TO 105
      DELAST = 1
      GO TO 101
  105 [illegible]
      IF ([illegible] .EQ. 0) DCST(I) = DATA(I)
  106 P(I) = DATA(I)
      [illegible] = 1
      DO 107 I=1,[illegible]
      IF (DCST(I) .GT. DTEST) COMP = 1
  107 DATA(I) = (DATA(I))
      IF (DCOUNT .EQ. 0) DELAST = 0
      IF (COMP .EQ. DELST) GO TO 109
      DO 108 I=1,[illegible]
  108 DCST(I) = DATA(I)
      DELST = COMP
  109 IF (DCOUNT .GT. 0) COMP = 1
      IF (DCOUNT .GT. 0 .AND. DI .GT. [illegible]) COMP = 1
      [illegible] = 0
      DCOUNT = DCOUNT + 1
      IF (DCOUNT .EQ. 1) MCOUNT = MCOUNT + 1
      IF (DCOUNT .EQ. 40 .AND. MCOUNT .EQ. 47) GO TO 101
      DO 110 I=1,[illegible]
  110 DATA(I) = DCST(I)
      DELAST = 1
      GO TO 101
  201 IF (COMP .EQ. 0) GO TO 103
      DELA = 0.01 * (B - A)
      A = A + DELA
      CALL GFOR (A,D)
      A = A - DELA
      B = B - DELA
      CALL [illegible]
      CALL GFOR (A,D)
      GO TO 101
  102 DATA(I) = P(I)
      DELD = 0.01 * (B - D)
      D = D + DELD
      CALL GFOR (A,D)
      D = D - DELD
```

```
      E = E - DELD
      CALL GEOM (A5)
      E = E + DELD
      DELF = 0.01 * (G - F)
      F = F + DELF
      CALL TRANS
      CALL GEOM (A6)
      DO 103 I=1,6
  103 DATA(I) = P1(I)
      C = C - DELF
      CALL GEOM (A7)
      C = C + DELF
      DO 104 I=1,6
      CHAN(I,1) = (A1(I) - A1(I)) / DELF
      CHAN(I,2) = (A2(I) - A1(I)) / DELF
      CHAN(I,3) = (A3(I) - A1(I)) / DELF
      CHAN(I,4) = (A4(I) - A1(I)) / DELF
      CHAN(I,5) = (A5(I) - A1(I)) / DELF
      CHAN(I,6) = (A7(I) - A1(I)) / (-DELF)
      DO 104 J=1,6
  104 DUMB(I,J) = CHAN(I,J)
      GO TO 107
  105 DO 106 I=1,6
      DO 106 J=1,6
  106 CHAN(I,J) = DUMB(I,J)
  107 CALL INVERT (JUMP, 6)
      A = A + CH1(1)
      B = B + CH1(2)
      D = D + CH1(3)
      E = E + CH1(4)
      F = F + CH1(5)
      G = G + CH1(6)
      CALL COMPAT
      CALL TRANS
      GO TO 151
C
C     WING WITH END-PLATED VERTICAL FENCE AT THE TIP.  THIS CASE WAS
C     REQUESTED BY JAMES BLACKWELL OF NASA LANGLEY RESEARCH CENTER.
C
  171 CALL GEOM (A1)
      R1 = RSSERR(6)
      WRITE (6,172) (DATA(I),I=1,6) , (R1,J),J=1,6) , R1
  172 FORMAT (1H , F5.1, 3F9.5, F5.1, 9F9.5, F15.6)
      IF (ILAST .NE. 1) GO TO 174
      A1(4) = A1(4) + PIREQ
      A1(5) = A1(5) + ETAIRQ
      A1(6) = A1(6) + ETAORQ
      RETURN
  174 IF (R1 .GT. QTEST1) GO TO 175
      ILAST = 1
      GO TO 171
  175 DO 176 I=1,6
      IF (ICOUNT .EQ. 0) DEST(I) = DATA(I)
  176 P1(I) = DATA(I)
      JUMP = 0
      DO 177 I=1,6
      IF (ABS(A1(I)) .GT. 0.04 .AND. ((MODE .EQ. 2) .OR. (MODE .EQ. 1
     A    .AND. ICOUNT .GT. 4))) JUMP = 1
```

```
177 CH1(I) = (-A1(I))
    IF (ICOUNT .EQ. 0) GOTO 178
    IF (R1 .GE. RDEST) GO TO 179
    GO 178 I=1,6
179 RDEST(I) = DATA(I)
    RDEST = R1
178 IF (ICOUNT .EQ. 0) JUMP = 1
    IF (ICOUNT .GE. 4 .AND. R1 .GT. (0.5 * RDT) JUMP = 1
    R0 = R1
    ICOUNT = ICOUNT + 1
    IF (JUMP .EQ. 1) ACOUNT = ACOUNT + 1
    IF (ICOUNT .GE. 40 .AND. ACOUNT .EQ. 0) GO TO 101
    DO 180 I=1,6
180 DATA(I) = RDEST(I)
    LAST = 1
    GO TO 171
101 IF (JUMP .EQ. 0) GO TO 185
    DELA = 0.01 * (B - A)
    B = B - DELA
    CALL GFOM (A2)
    C = B + DELA
    DELC = 0.05 * (C - D)
    C = C - DELC
    CALL GFOM (A3)
    C = C + DELC
    DELD = 0.01 * (E - D)
    D = D + DELD
    CALL GFOM (A4)
    D = D - DELD
    DELF = 0.01 * (G - F)
    F = F + DELF
    CALL GFOM (A5)
    F = F - DELF
    DELH = 0.01 * (H - G)
    G = G + DELH
    CALL GFOM (A6)
    G = G - DELH
    H = H + DELH
    CALL GFOM (A7)
    H = H - DELH
    DO 104 I=1,6
    CHAN(I,1) = (A1(I) - A0(I)) / (NVALUE
    CHAN(I,2) = (A2(I) - A1(I)) / (NVALUE
    CHAN(I,3) = (A3(I) - A2(I)) / DELA
    CHAN(I,4) = (A4(I) - A3(I)) / DELF
    CHAN(I,5) = (A5(I) - A4(I)) / DELH
    CHAN(I,6) = (A6(I) - A5(I)) / DELH
    CHAN(I,7) = (A7(I) - A6(I)) / DELH
    DO 104 CH=1,6
104 DCH(I,CH) = CHAN(I,CH)
    GO TO 107
105 DO 106 I=1,6
    DO 106 CH=1,6
106 CHAN(I,CH) = DCH(I,CH)
107 CALL INVERT (COMP, D)
    C = B * CH(1)
    C = C + CH(2)
    C = D * CH(3)
    C = G * CH(5)
```

```
F = F + CH1(6)
H = H + CH1(6)
CALL COMPAT
CALL TRANS
GO TO 17.
END
```

[illegible] PROGRAM LENGTH
[illegible]

FUNCTION ASSIGNMENTS

STATEMENT ASSIGNMENTS

[illegible]	[illegible]	15	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	000272	[illegible]	[illegible]	19	[illegible]	24	[illegible]
[illegible]	[illegible]	22	000325	27	[illegible]	29	003277
[illegible]	[illegible]	34	[illegible]	[illegible]	[illegible]	36	[illegible]
37	[illegible]	38	000570	[illegible]	000455	40	[illegible]
[illegible]	[illegible]	42	[illegible]	[illegible]	000746	42	[illegible]
[illegible]	[illegible]	53	[illegible]	[illegible]	001171	57	[illegible]
[illegible]	[illegible]	59	[illegible]	[illegible]	[illegible]	[illegible]	001257
[illegible]	[illegible]	69	[illegible]	72	001135	74	001342
[illegible]	[illegible]	[illegible]	[illegible]	77	001014	78	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	85	[illegible]
[illegible]	001072	[illegible]	001715	[illegible]	001752	90	001757
[illegible]	[illegible]	92	002032	[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	105	[illegible]
[illegible]	[illegible]	105	[illegible]	[illegible]	[illegible]	107	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	001407	[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	001125	[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]	152	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]	157	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	002740	[illegible]	[illegible]	178	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	003220	187	[illegible]	502	[illegible]
[illegible]	[illegible]	[illegible]	003342				

BLOCK NAMES AND LENGTHS
= [illegible]

VARIABLE ASSIGNMENTS

[illegible]	[illegible]	[illegible]	[illegible]	ADSC	[illegible]	A1	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]	A2	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	A5	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	C	[illegible]
[illegible]	[illegible]	CHAR	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	000472	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	F	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]	H	[illegible]	I	[illegible]
ICOUNT	[illegible]	ILAST	[illegible]	[illegible]	[illegible]	JUMP	[illegible]
[illegible]	[illegible]	MODE	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	PI	[illegible]	PIREG	[illegible]	POREG	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	002417001	R2	[illegible]
[illegible]	[illegible]	TABLE	[illegible]				

START OF CONSTANTS

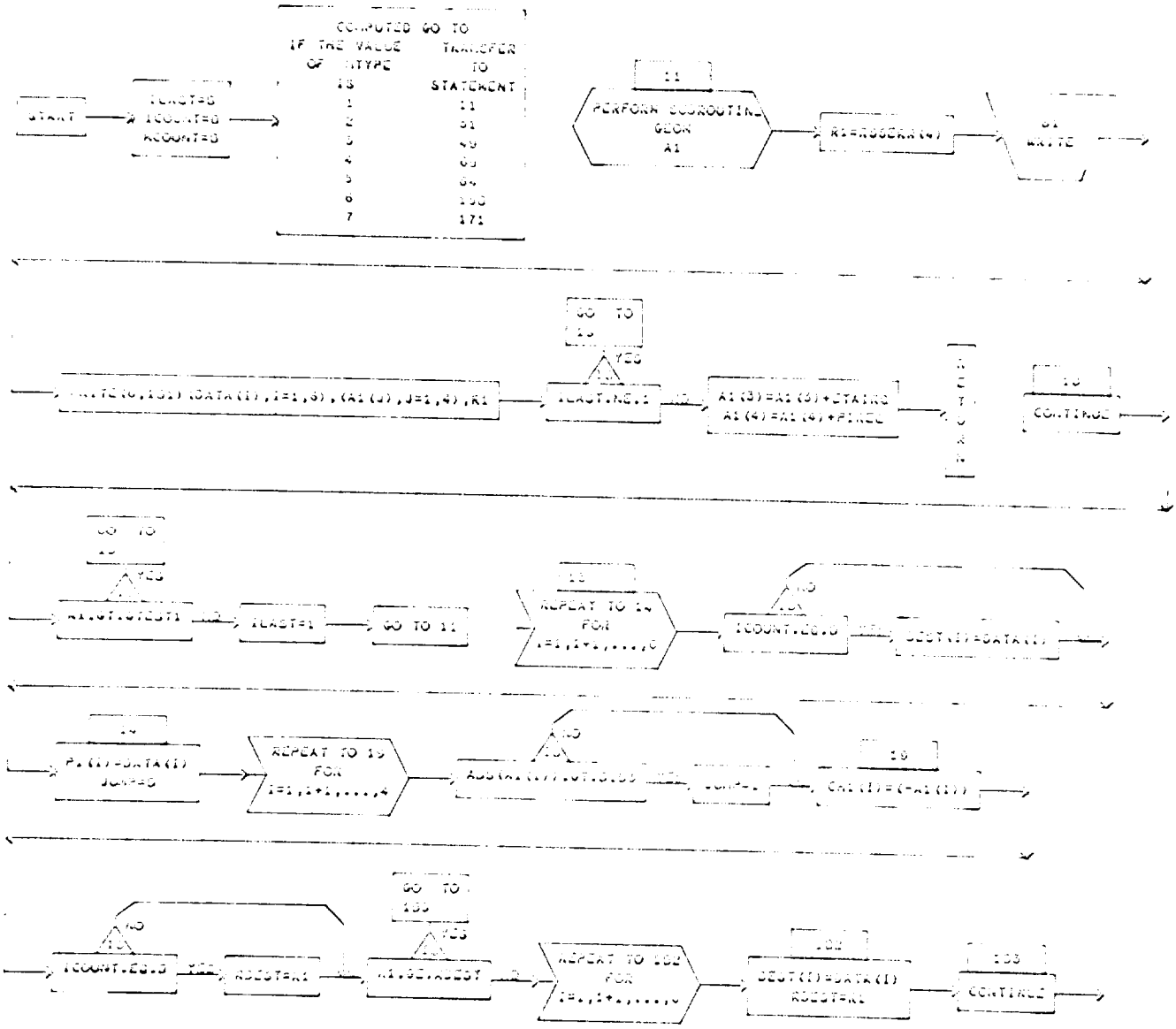
000037

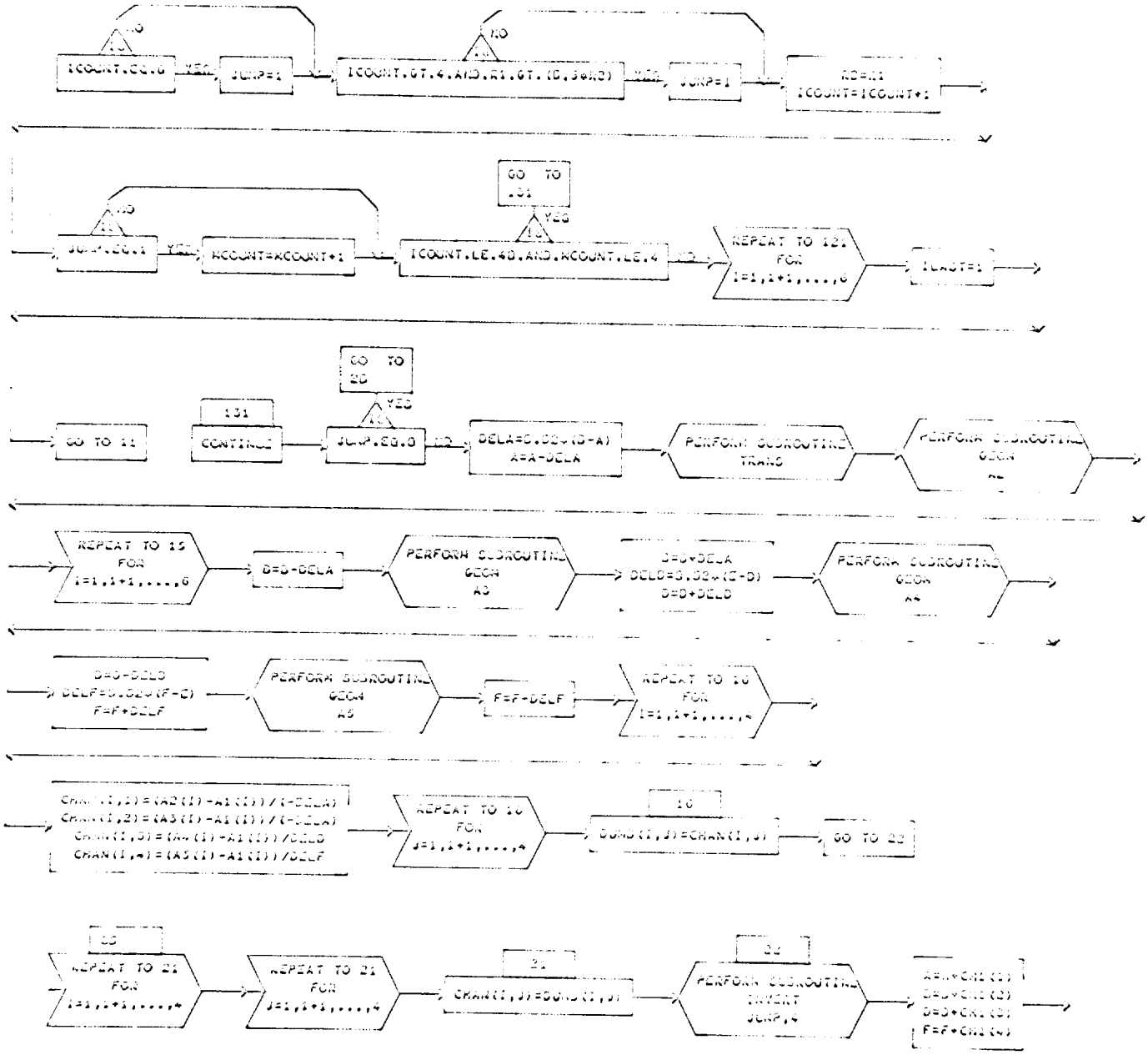
START OF TEMPORARIES
000404

START OF INDIRECTS
000402

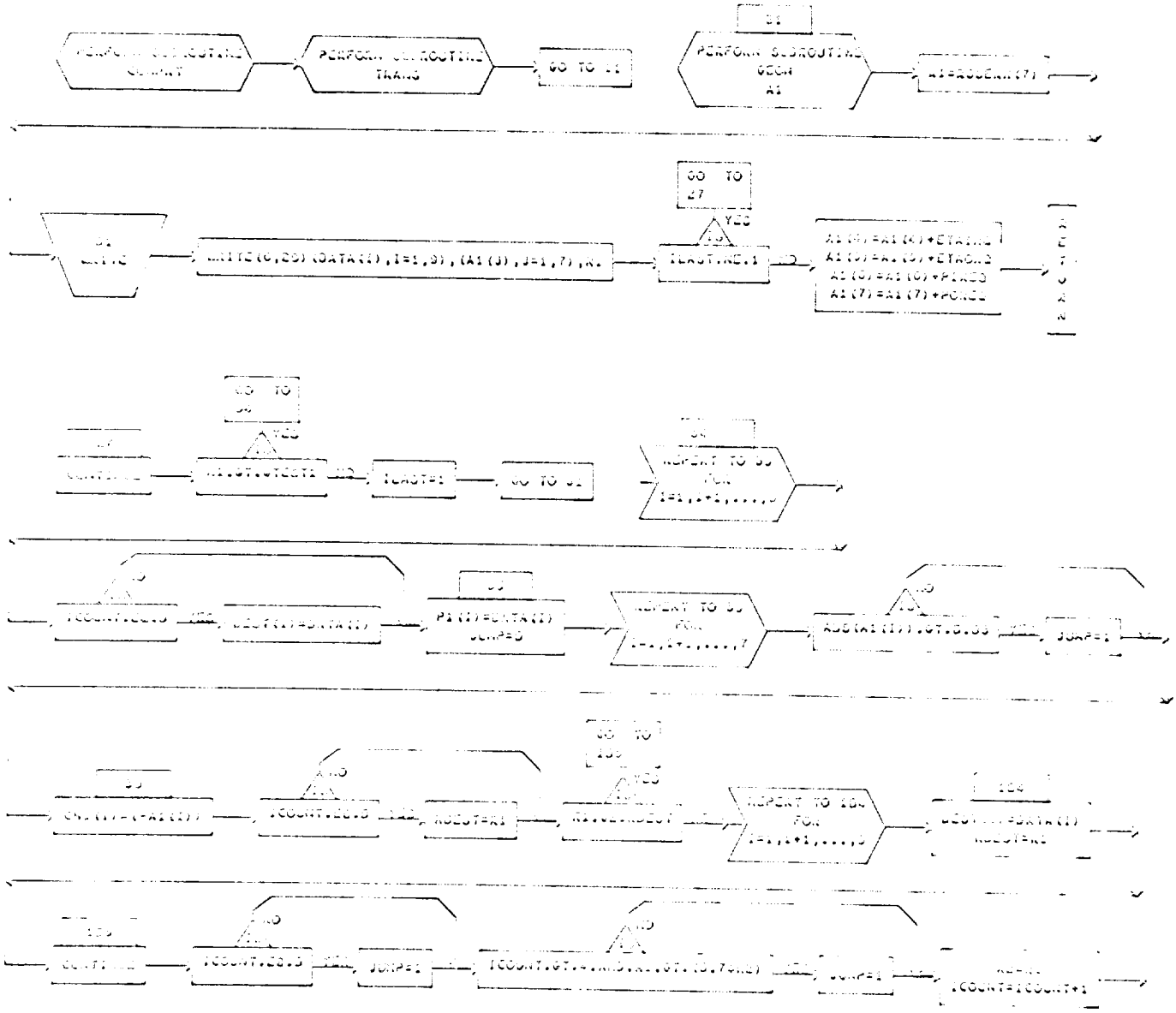
UNUSED COMPILER SPACE
001000

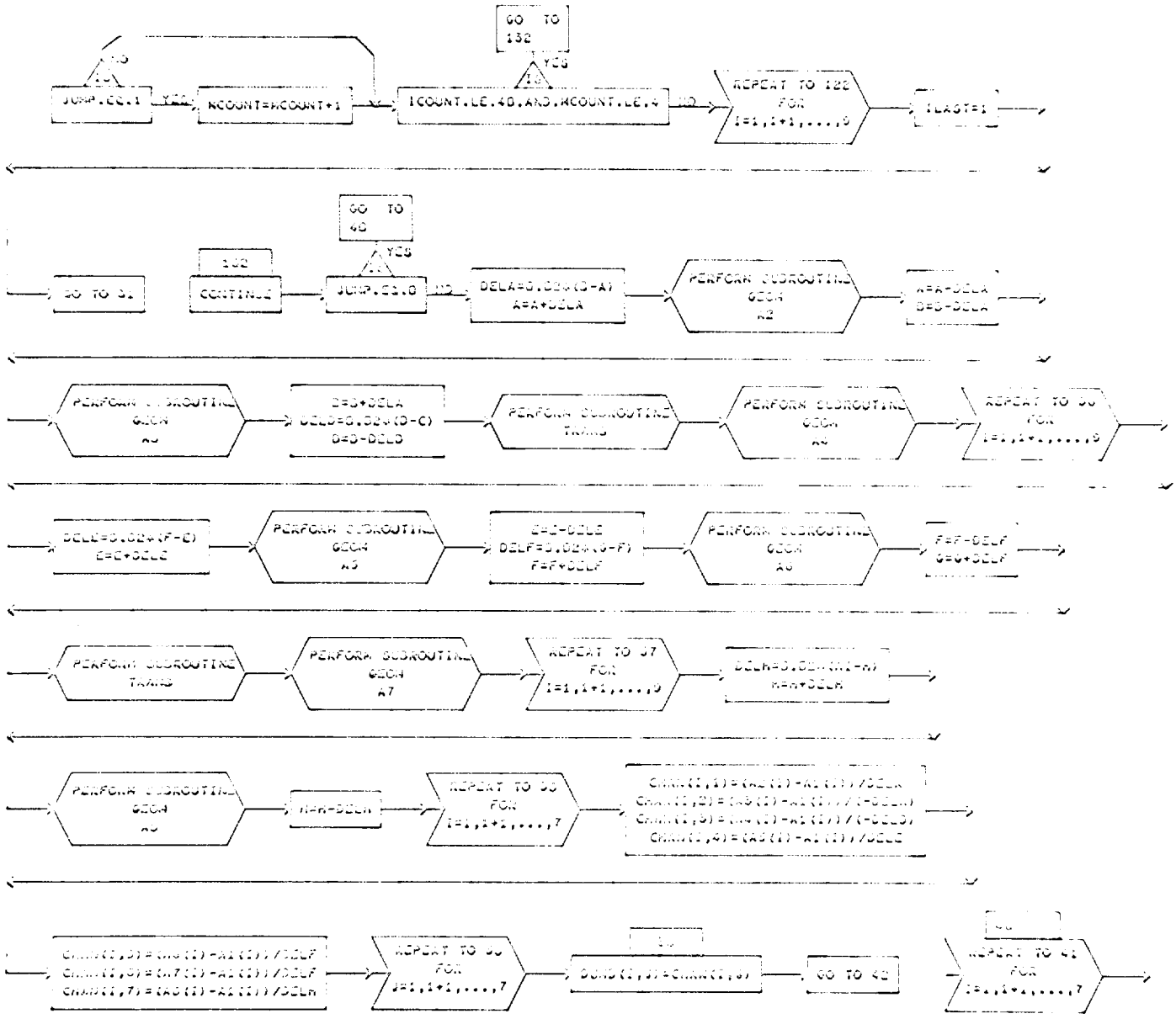
SUBROUTINE INNER

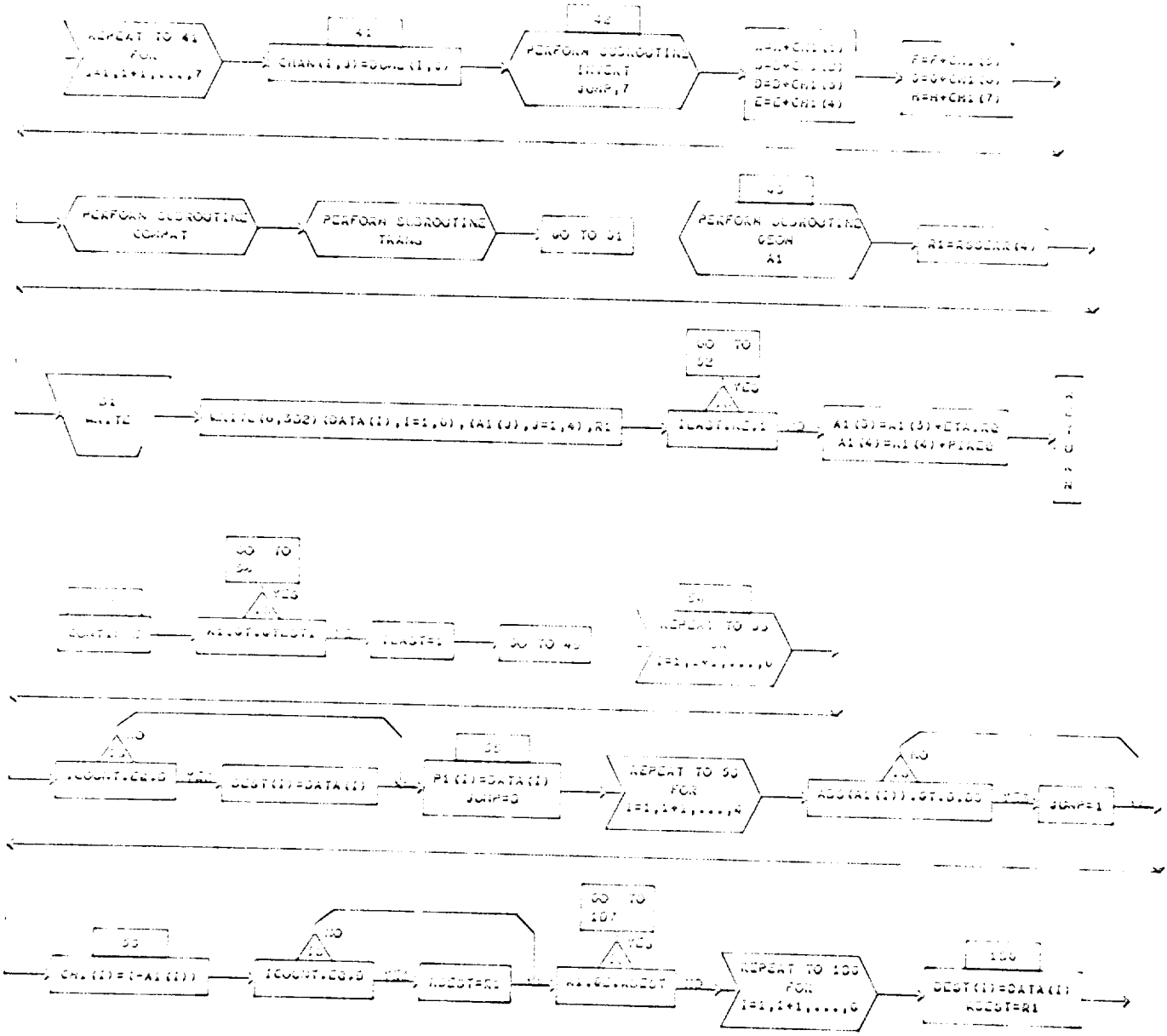


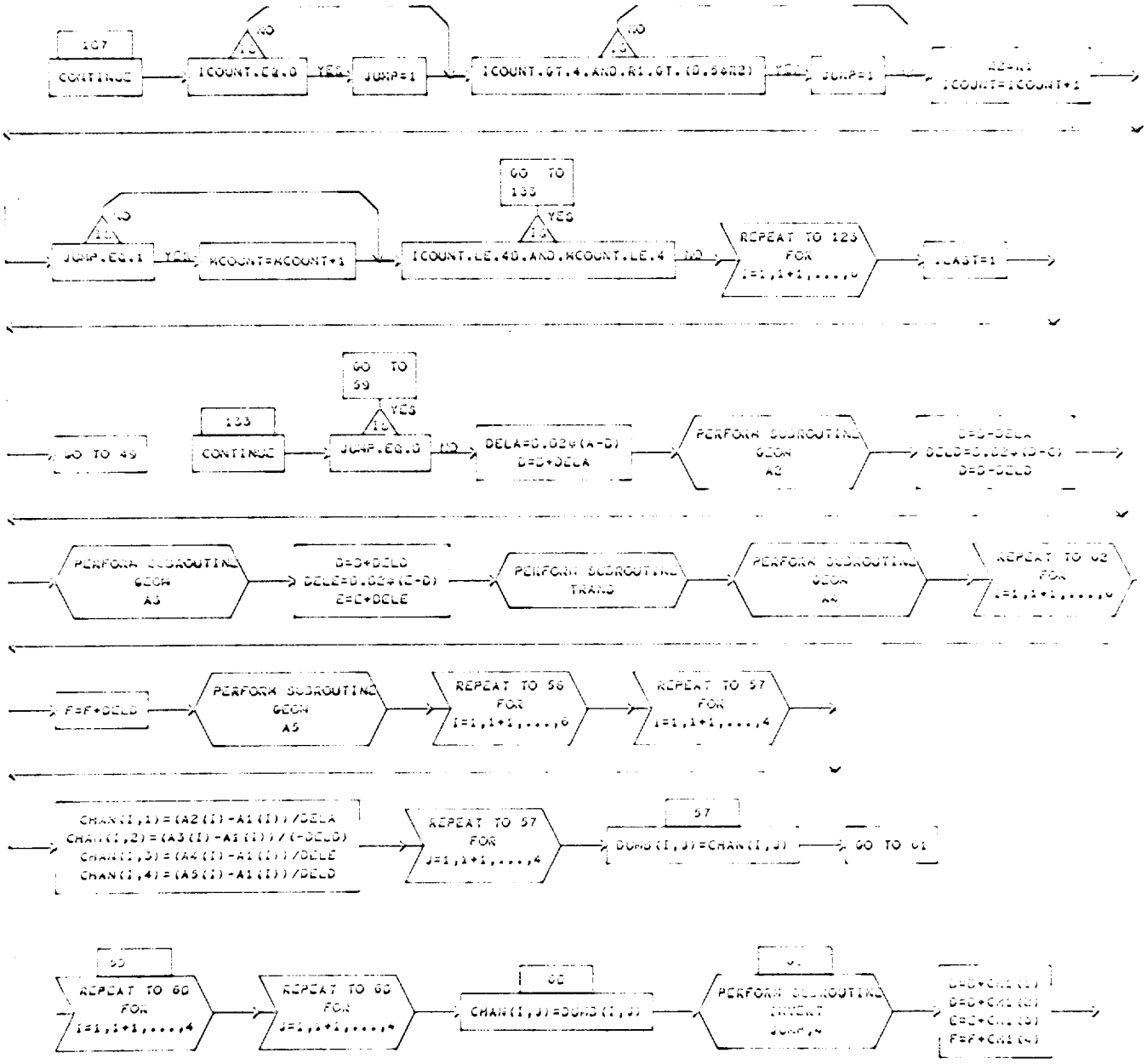


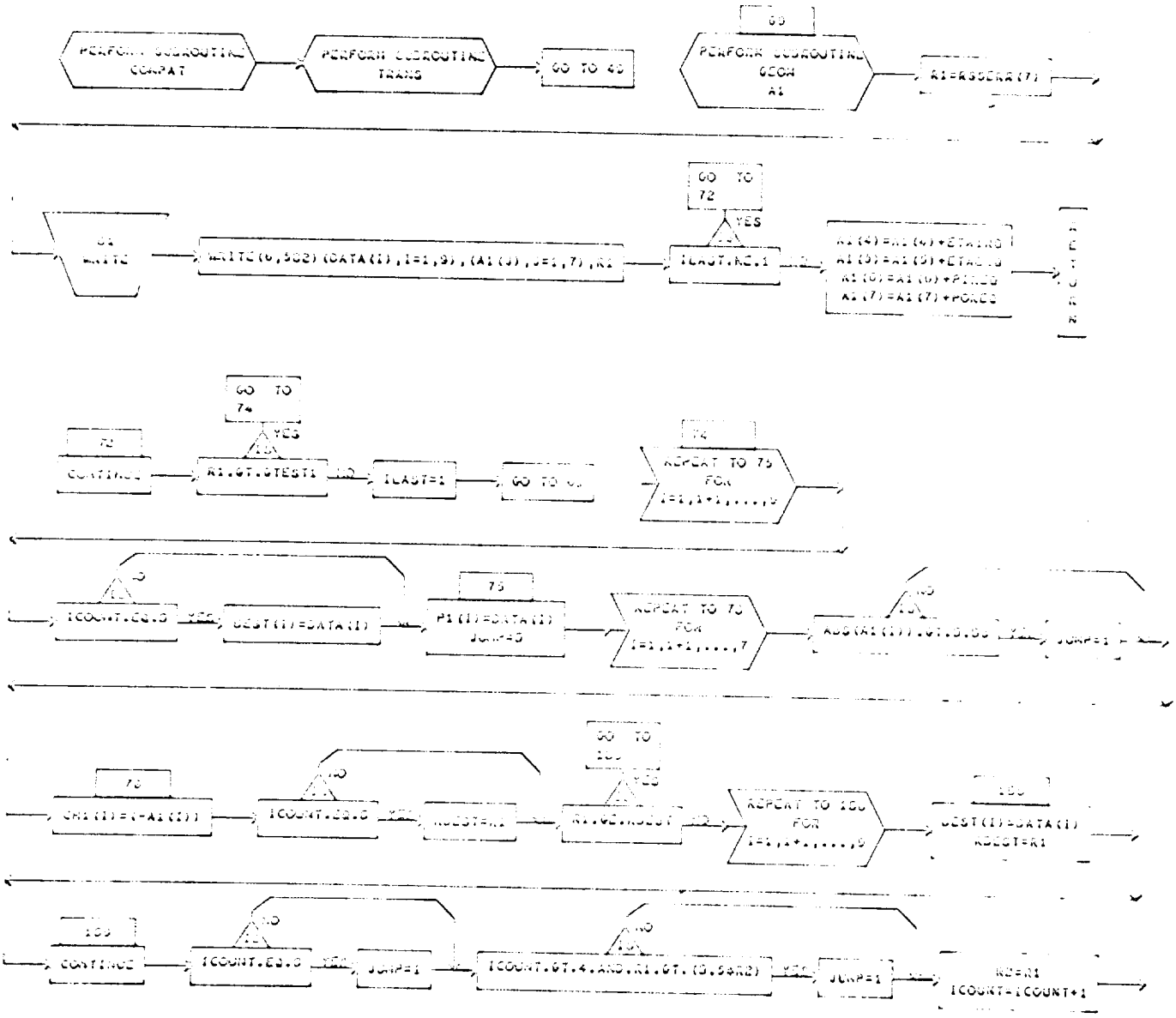
ICOUNT.EQ.0
YES
JUMP=1
ICOUNT.GT.4.AND.R1.GT.(D.5*R2)
JUMP=1
R2=R1
ICOUNT=ICOUNT+1
JUMP.EQ.1
WCOUNT=WCOUNT+1
GO TO 131
ICOUNT.LE.40.AND.WCOUNT.LE.4
REPEAT TO 121 FOR I=1,I+1,...,6
ILAST=1
GO TO 11
131
CONTINUE
GO TO 20
JUMP.EQ.0
DELA=0.02*(B-A)
A=A-DELA
PERFORM SUBROUTINE TRANS
PERFORM SUBROUTINE GEOM
REPEAT TO 15 FOR I=1,I+1,...,6
B=B-DELA
PERFORM SUBROUTINE GEOM A3
B=B+DELA
DELD=0.02*(E-D)
D=D+DELD
PERFORM SUBROUTINE GEOM A4
B=B-DELB
DELF=0.02*(F-E)
F=F+DELF
PERFORM SUBROUTINE GEOM A5
F=F-DELF
REPEAT TO 16 FOR I=1,I+1,...,4
CHAN(I,1)=(A2(I)-A1(I))/(-DELA)
CHAN(I,2)=(A3(I)-A1(I))/(-DELA)
CHAN(I,3)=(A4(I)-A1(I))/DELD
CHAN(I,4)=(A5(I)-A1(I))/DELF
REPEAT TO 16 FOR J=1,I+1,...,4
16
DUMB(I,J)=CHAN(I,J)
GO TO 22
REPEAT TO 21 FOR I=1,I+1,...,4
REPEAT TO 21 FOR J=1,I+1,...,4
21
CHAN(I,J)=DUMB(I,J)
22
PERFORM SUBROUTINE INVERT JUMP,4
A=A+CH1(1)
B=B+CH1(2)
D=D+CH1(3)
F=F+CH1(4)

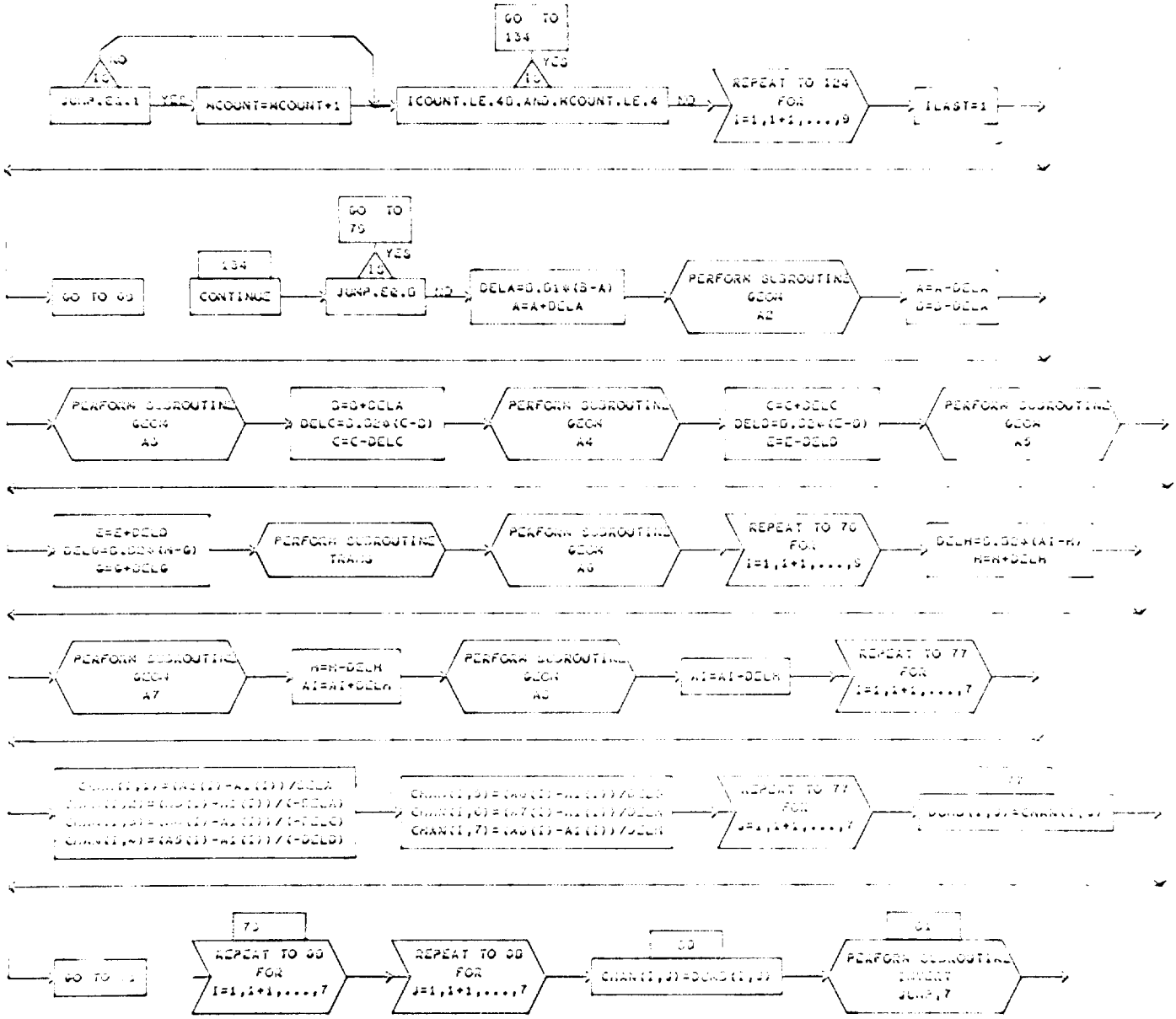

PERFORM SUBROUTINE TRANS
GO TO 11
PERFORM SUBROUTINE GEOM A1
GO TO 27
YES
ILAST.NE.1
GO TO 31
REPEAT TO 33 FOR
ILAST=1
GO TO 31
REPEAT TO 33 FOR
JUMP=1
GO TO 135
YES
REPEAT TO 134 FOR
JUMP=1
JUMP=1
ICOUNT=ICOUNT+1

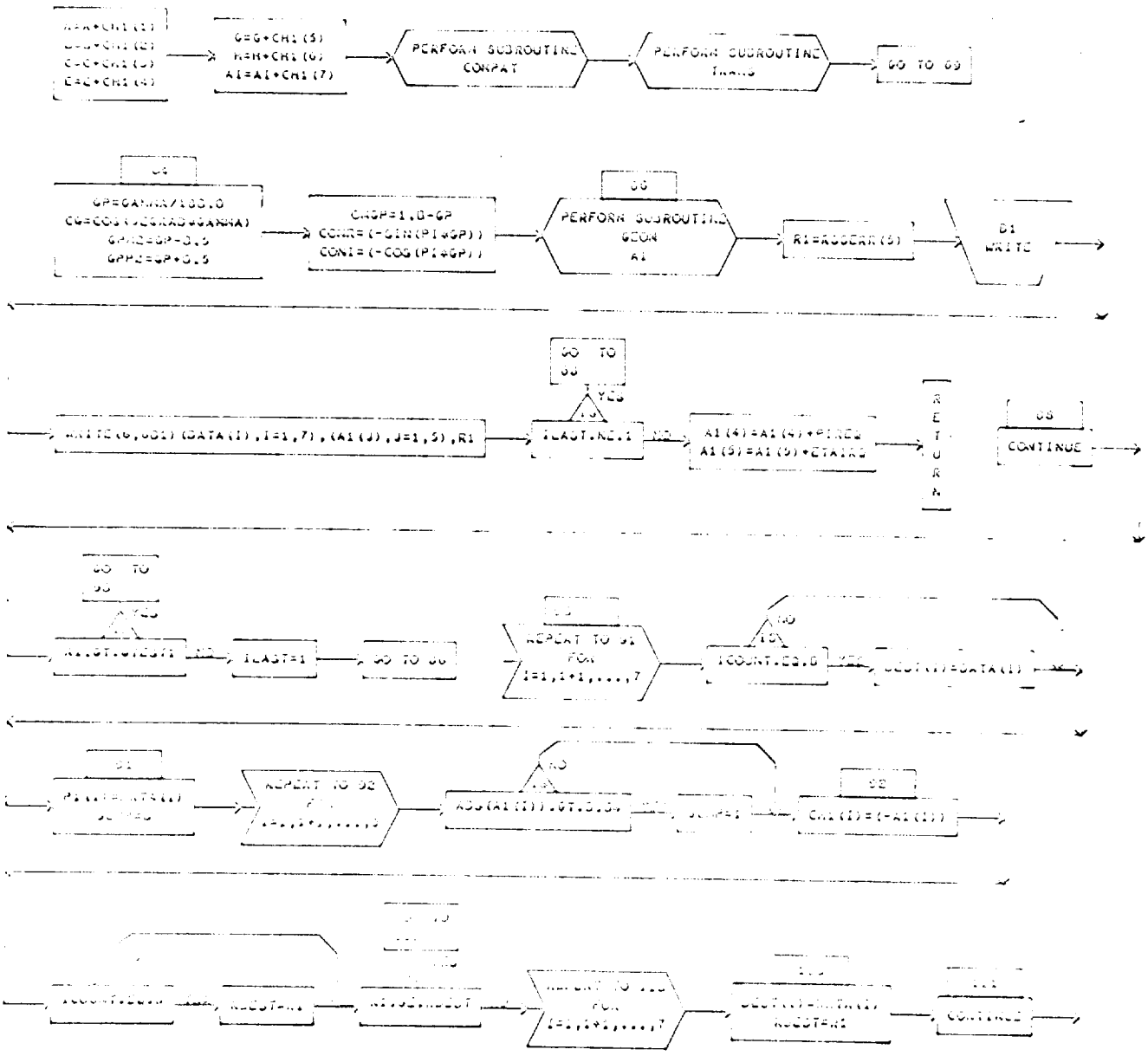

GO TO 132
YES
NCOUNT=NCOUNT+1
ICOUNT.LE.40.AND.NCOUNT.LE.4
REPEAT TO 122 FOR I=1,1+1,...,9
ILAST=1
GO TO 40
YES
CONTINUE
DELA=0.024(B-A)
A=A+DELA
PERFORM SUBROUTINE GEOM A2
PERFORM SUBROUTINE GEOM A3
B=B+DELA
DELD=0.024(D-C)
D=D-DELD
PERFORM SUBROUTINE TRANS
PERFORM SUBROUTINE GEOM A4
REPEAT TO 36 FOR I=1,1+1,...,9
DELE=0.024(F-E)
E=E+DELE
PERFORM SUBROUTINE GEOM A5
E=E-DELE
DELF=0.024(G-F)
F=F+DELF
PERFORM SUBROUTINE GEOM A6
F=F-DELF
G=G+DELF
PERFORM SUBROUTINE TRANS
PERFORM SUBROUTINE GEOM A7
REPEAT TO 37 FOR I=1,1+1,...,9
PERFORM SUBROUTINE GEOM A8
H=H-DELH
REPEAT TO 38 FOR I=1,1+1,...,7
GO TO 42
REPEAT TO 41 FOR I=1,1+1,...,7

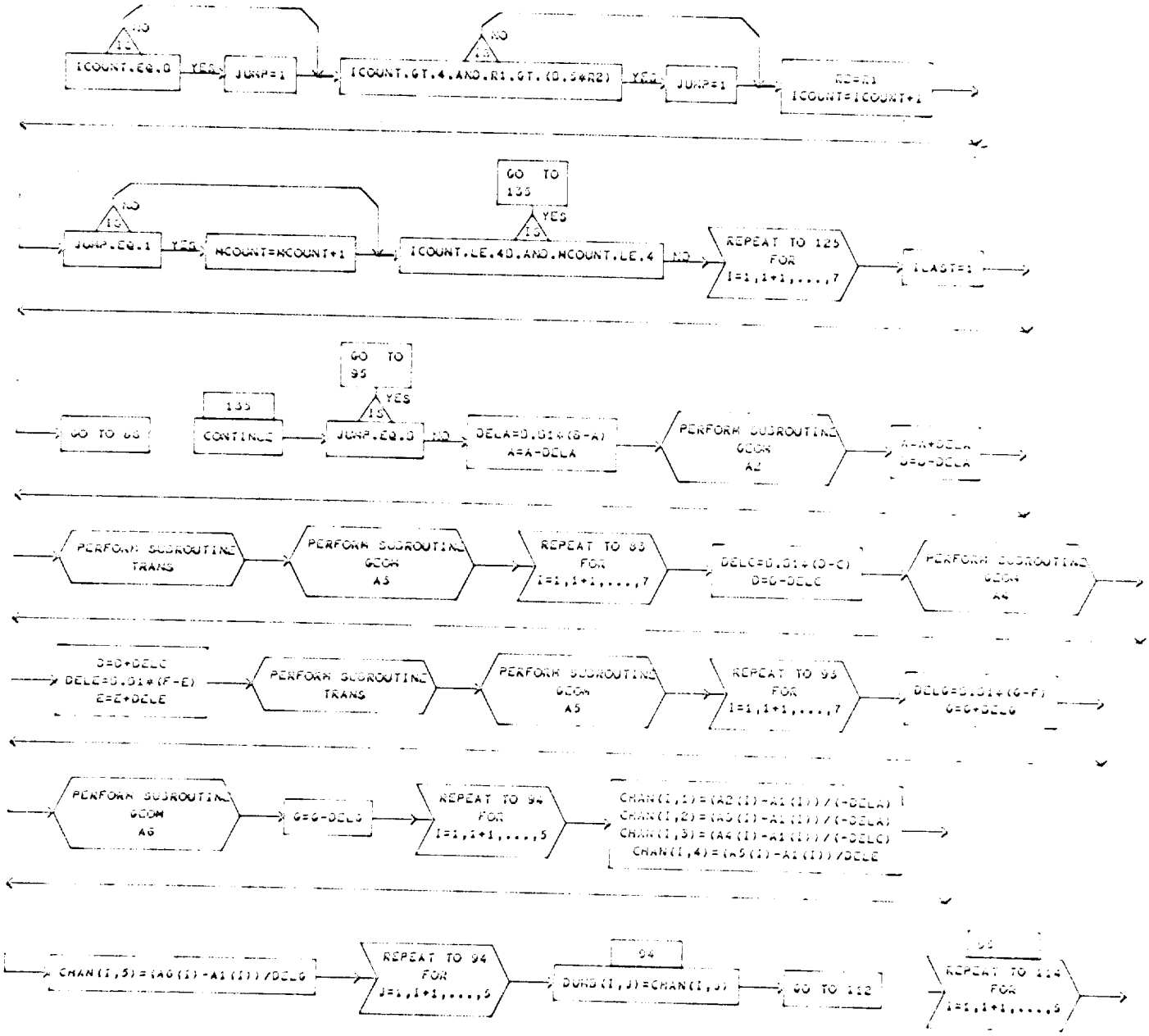

REPEAT TO 41
FOR
41
PERFORM SUBROUTINE
INVERT
JUMP,7
PERFORM SUBROUTINE
COMPAT
PERFORM SUBROUTINE
TRANS
GO TO 51
PERFORM SUBROUTINE
GEOM
A1
GO TO
52
YES
GO TO
YES
GO TO 45
REPEAT TO 53
NO
DEST(I)=DATA(I)
P1(I)=DATA(I)
JUMP=3
REPEAT TO 53
FOR
NO
JUMP=1
GO TO
107
YES
REPEAT TO 105
FOR
DEST(I)=DATA(I)
RDEST=R1

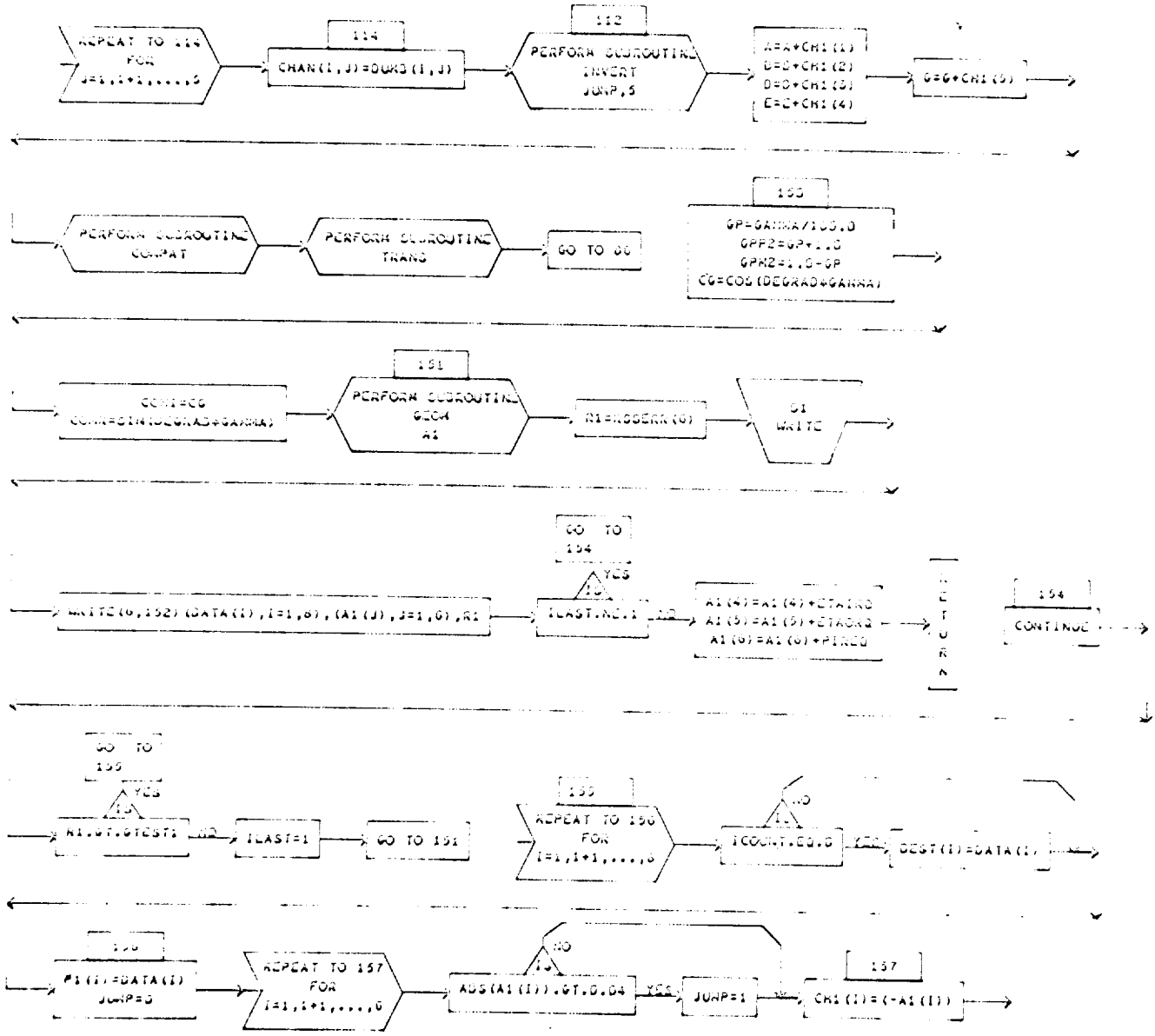

107
CONTINUE
ICOUNT.EQ.0
YES
NO
JUMP=1
ICOUNT.GT.4.AND.R1.GT.(0.5*R2)
JUMP=1
ICOUNT=ICOUNT+1
JUMP.EQ.1
KCOUNT=KCOUNT+1
ICOUNT.LE.40.AND.KCOUNT.LE.4
GO TO 133
REPEAT TO 125 FOR I=1,I+1,...,6
LAST=1
GO TO 49
133
CONTINUE
JUMP.EQ.0
GO TO 59
DELA=0.02*(A-B)
B=B+DELA
PERFORM SUBROUTINE GEOM A2
B=B-DELA
DELB=0.02*(B-C)
B=B-DELB
PERFORM SUBROUTINE GEOM A3
B=B+DELB
DELE=0.02*(C-D)
E=C+DELE
PERFORM SUBROUTINE TRANS
PERFORM SUBROUTINE GEOM A4
REPEAT TO 62 FOR I=1,I+1,...,6
F=F+DELD
PERFORM SUBROUTINE GEOM A5
REPEAT TO 56 FOR I=1,I+1,...,6
REPEAT TO 57 FOR I=1,I+1,...,4
CHAN(I,1)=(A2(I)-A1(I))/DELA
CHAN(I,2)=(A3(I)-A1(I))/(-DELB)
CHAN(I,3)=(A4(I)-A1(I))/DELE
CHAN(I,4)=(A5(I)-A1(I))/DELD
REPEAT TO 57 FOR J=1,I+1,...,4
57
DUMB(I,J)=CHAN(I,J)
GO TO 61
55
REPEAT TO 60 FOR I=1,I+1,...,4
REPEAT TO 60 FOR J=1,I+1,...,4
60
CHAN(I,J)=DUMB(I,J)
PERFORM SUBROUTINE INVERT JUMP,4
B=B+CH1(1)
D=D+CH1(2)
E=E+CH1(3)
F=F+CH1(4)

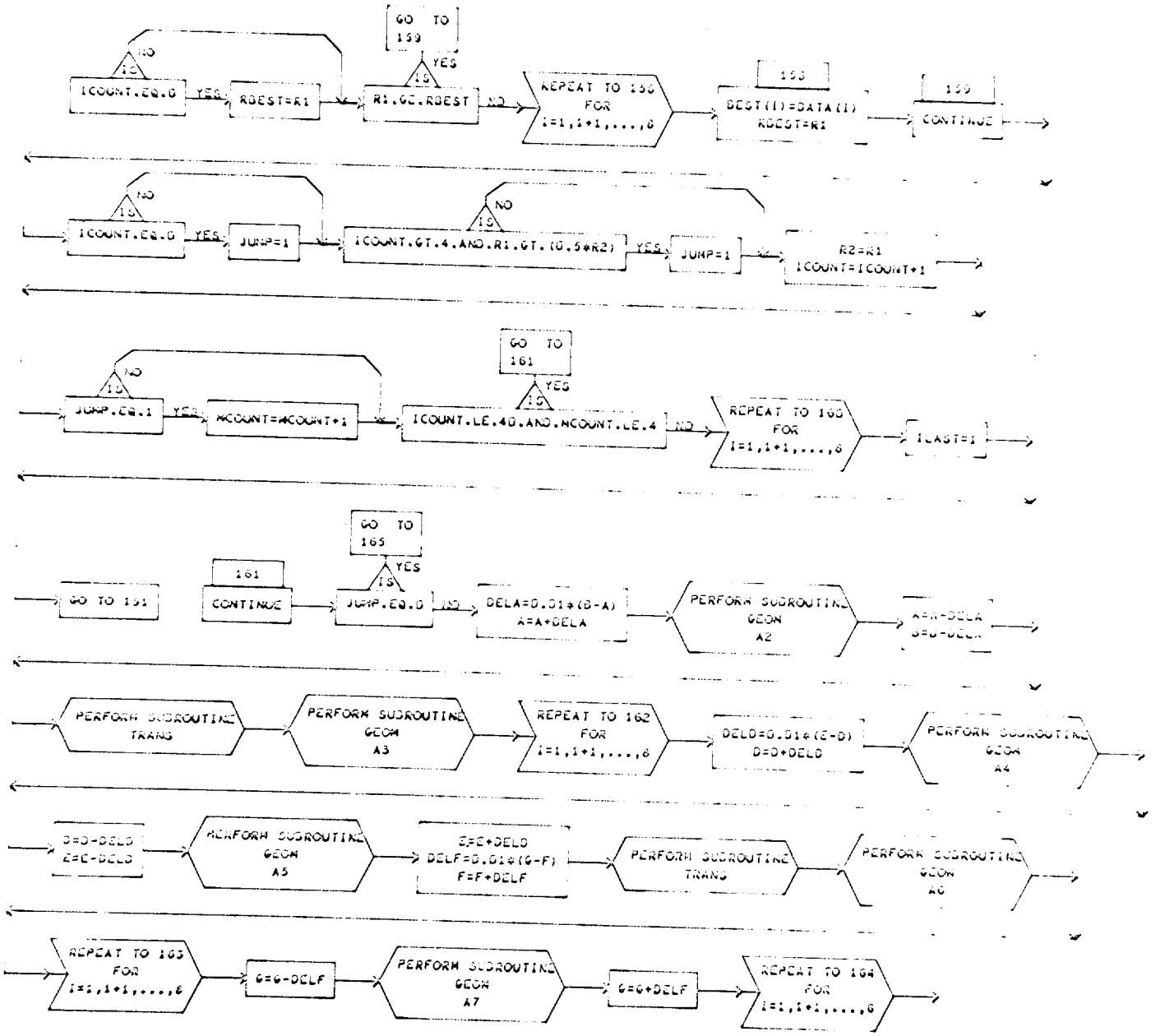

PERFORM SUBROUTINE COMPAT
PERFORM SUBROUTINE TRANS
GO TO 40
PERFORM SUBROUTINE GEOM A1
R1=RSSERR(7)
GO TO 72
YES
ILAST.NE.1
RETURN
GO TO 74
YES
R1.GT.GTEST1
ILAST=1
REPEAT TO 75 FOR I=1,1+1,...,9
ICOUNT.EQ.0
BEST(I)=DATA(I)
P1(I)=DATA(I)
REPEAT TO 75 FOR I=1,1+1,...,7
JUMP=1
CH1(I)=(-A1(I))
ICOUNT.EQ.0
GO TO
YES
REPEAT TO 180 FOR I=1,1+1,...,9
BEST(I)=DATA(I)
RBEST=R1
CONTINUE
ICOUNT.EQ.0
JUMP=1
ICOUNT.GT.4.AND.R1.GT.(0.5*R2)
JUMP=1
R2=R1
ICOUNT=ICOUNT+1

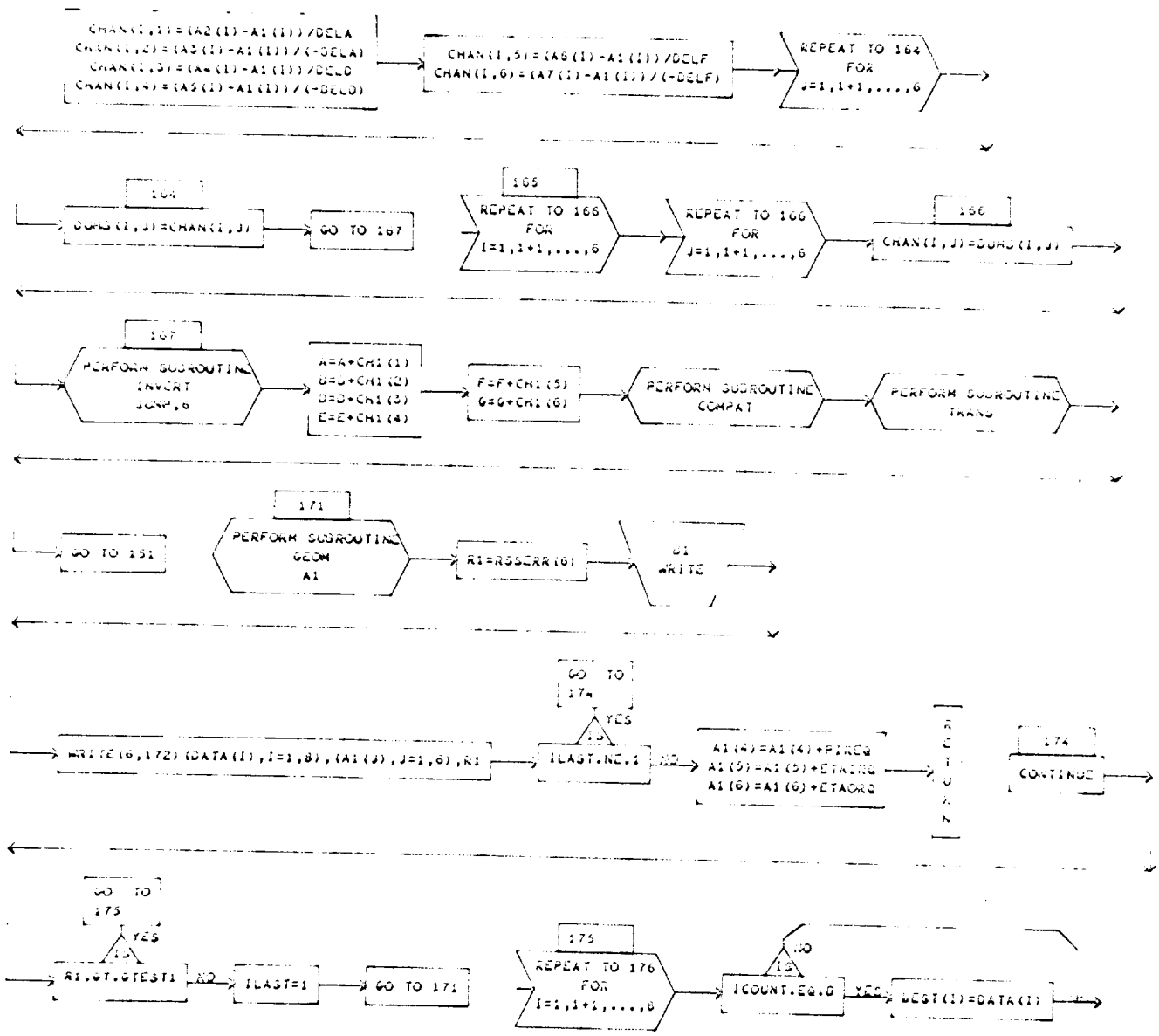

GO TO 134
YES
NO
JUMP.EQ.1
YES
NCOUNT=NCOUNT+1
ICOUNT.LE.40.AND.NCOUNT.LE.4
NO
REPEAT TO 124 FOR I=1,1+1,...,9
ILAST=1
GO TO 75
YES
134
CONTINUE
JUMP.EQ.0
NO
DELA=0.01*(B-A)
A=A+DELA
PERFORM SUBROUTINE GEOM A2
PERFORM SUBROUTINE GEOM A3
B=B+DELA
DELC=0.02*(C-B)
C=C-DELC
PERFORM SUBROUTINE GEOM A4
C=C+DELC
DELD=0.02*(E-D)
E=E-DELD
PERFORM SUBROUTINE GEOM A5
E=E+DELD
PERFORM SUBROUTINE TRANS
PERFORM SUBROUTINE GEOM A6
REPEAT TO 76 FOR I=1,1+1,...,9
PERFORM SUBROUTINE GEOM A7
H=H-DELH
AI=AI+DELH
PERFORM SUBROUTINE GEOM A8
AI=AI-DELH
REPEAT TO 77 FOR I=1,1+1,...,7
REPEAT TO 77 FOR J=1,1+1,...,7
REPEAT TO 88 FOR I=1,1+1,...,7
REPEAT TO 88 FOR J=1,1+1,...,7
PERFORM SUBROUTINE INVERT JUMP,7

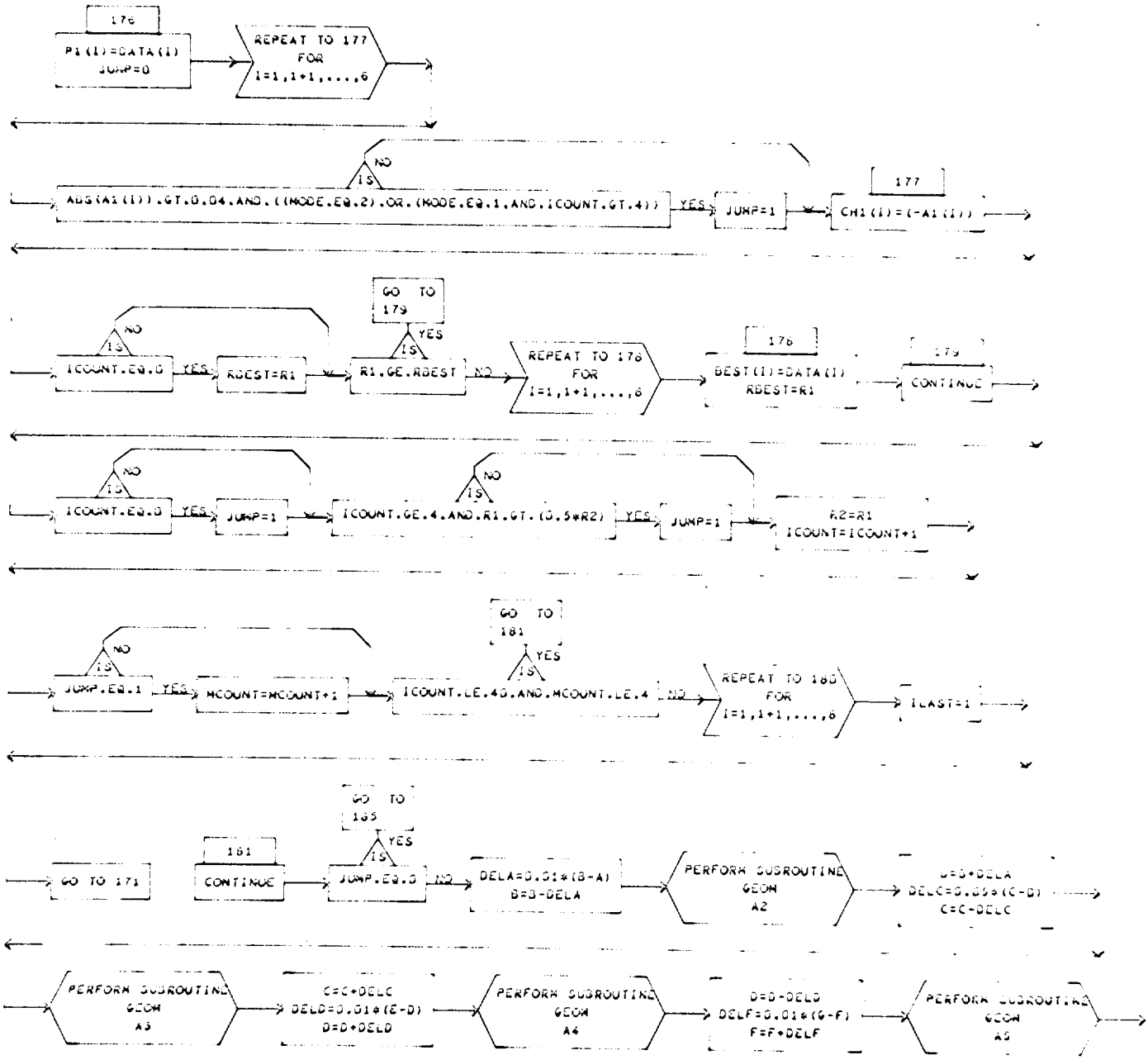

G=G+CH1(5)
H=H+CH1(6)
AI=AI+CH1(7)
PERFORM SUBROUTINE
CONPAT
GO TO 69
PERFORM SUBROUTINE
GEON
AI
WRITE
GO TO
YES
ILAST.NE.1
A1(4)=A1(4)+PIREQ
A1(5)=A1(5)+ETAIRQ
RETURN
CONTINUE
ILAST=1
REPEAT TO 91
FOR
I=1,1+1,...,7
ICOUNT.EQ.0
REPEAT TO 92
CH1(I)=(-A1(I))
CONTINUE

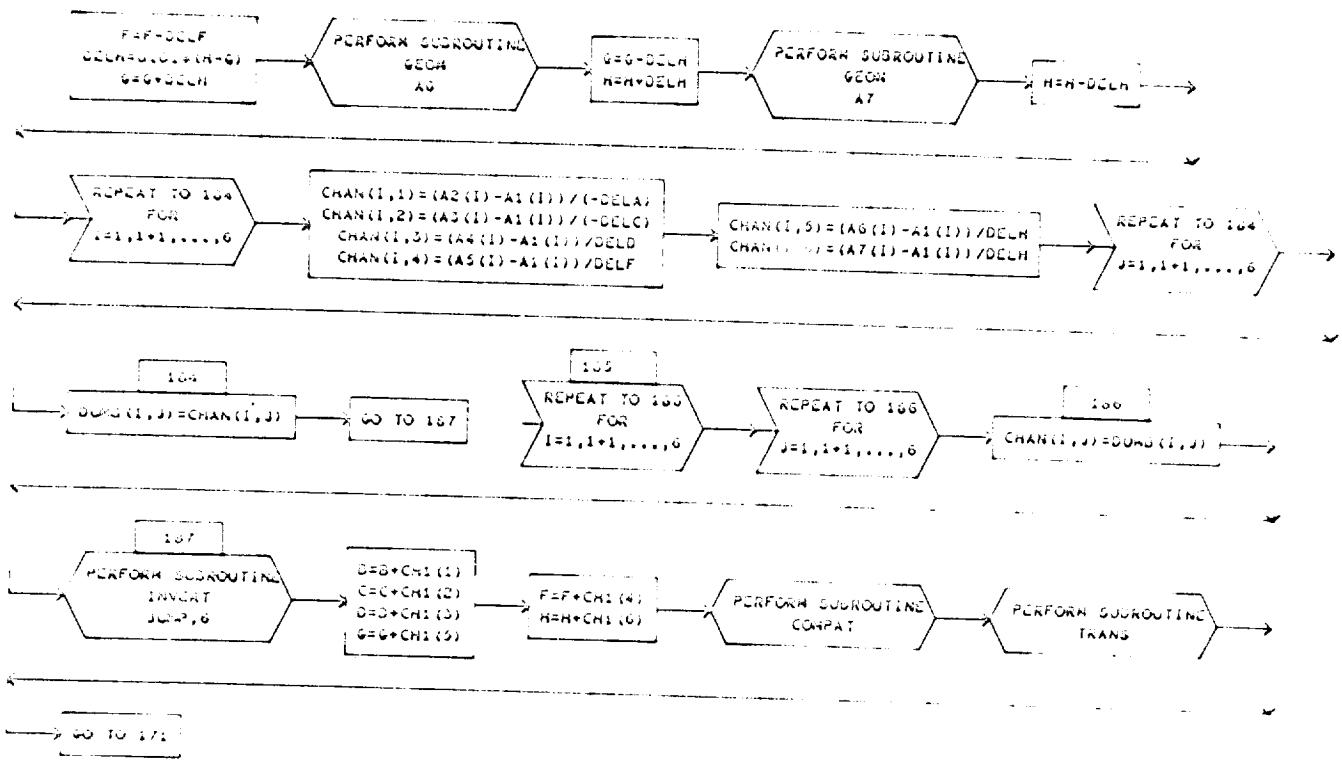

ICOUNT.EQ.0
YES
NO
JUMP=1
ICOUNT.GT.4.AND.R1.GT.(0.5*R2)
YES
NO
JUMP=1
R2=R1
ICOUNT=ICOUNT+1
JUMP.EQ.1
YES
NO
NCOUNT=NCOUNT+1
ICOUNT.LE.40.AND.NCOUNT.LE.4
YES
NO
GO TO 135
REPEAT TO 125 FOR I=1,1+1,...,7
ICAST=1
GO TO 65
135
CONTINUE
JUMP.EQ.0
YES
NO
GO TO 95
DELA=0.01*(B-A)
A=A-DELA
PERFORM SUBROUTINE GCOM A2
A=A+DELA
B=B-DELA
PERFORM SUBROUTINE TRANS
PERFORM SUBROUTINE GCOM A3
REPEAT TO 85 FOR I=1,1+1,...,7
DELC=0.01*(D-C)
D=D-DELC
PERFORM SUBROUTINE GCOM A4
D=D+DELC
DELE=0.01*(F-E)
E=E+DELE
PERFORM SUBROUTINE TRANS
PERFORM SUBROUTINE GCOM A5
REPEAT TO 95 FOR I=1,1+1,...,7
DELG=0.01*(G-F)
G=G+DELG
PERFORM SUBROUTINE GCOM A6
G=G-DELG
REPEAT TO 94 FOR I=1,1+1,...,5
CHAN(I,1)=(A2(I)-A1(I))/(-DELA)
CHAN(I,2)=(A3(I)-A1(I))/(-DELA)
CHAN(I,3)=(A4(I)-A1(I))/(-DELC)
CHAN(I,4)=(A5(I)-A1(I))/DELE
CHAN(I,5)=(A6(I)-A1(I))/DELG
REPEAT TO 94 FOR J=1,1+1,...,5
94
DUMB(I,J)=CHAN(I,J)
GO TO 112
95
REPEAT TO 114 FOR I=1,1+1,...,5


REPEAT TO 114 FOR J=1,1+1,...,5
114
CHAN(I,J)=DUMB(I,J)
112
PERFORM SUBROUTINE INVERT JUMP,5
A=A+CH1(1)
B=B+CH1(2)
D=D+CH1(3)
E=E+CH1(4)
G=G+CH1(5)
PERFORM SUBROUTINE COMPAT
PERFORM SUBROUTINE TRANS
GO TO 86
153
GP=GAMMA/100.0
GPP2=GP+1.0
GPM2=1.0-GP
CG=COS(DEGRAD*GAMMA)
151
PERFORM SUBROUTINE GEOM A1
R1=RSSERR(6)
D1 WRITE
GO TO 154
YES
ILAST.NE.1
NO
A1(4)=A1(4)+ETAIRQ
A1(5)=A1(5)+ETAORQ
A1(6)=A1(6)+PIREQ
RETURN
154
CONTINUE
GO TO 155
YES
R1.GT.GTEST1
NO
ILAST=1
GO TO 151
155
REPEAT TO 156 FOR I=1,1+1,...,6
NO
ICOUNT.EQ.0
YES
DEST(I)=DATA(I)
156
P1(I)=DATA(I)
JUMP=0
REPEAT TO 157 FOR I=1,1+1,...,6
NO
ABS(A1(I)).GT.0.04
YES
JUMP=1
157
CH1(I)=(-A1(I))


GO TO 159
NO
IS
YES
ICOUNT.EQ.0
YES
RBEST=R1
R1.GE.RBEST
NO
REPEAT TO 160 FOR I=1,1+1,...,8
160
DEST(I)=DATA(I)
RDEST=R1
159
CONTINUE
NO
IS
ICOUNT.EQ.0
YES
JUMP=1
NO
IS
ICOUNT.GT.4.AND.R1.GT.(0.5*R2)
YES
JUMP=1
R2=R1
ICOUNT=ICOUNT+1
GO TO 161
NO
IS
YES
IS
JUMP.EQ.1
YES
NCOUNT=NCOUNT+1
ICOUNT.LE.40.AND.NCOUNT.LE.4
NO
REPEAT TO 163 FOR I=1,1+1,...,8
ICAST=1
GO TO 161
161
CONTINUE
GO TO 165
YES
IS
JUMP.EQ.0
NO
DELA=0.01*(B-A)
A=A+DELA
PERFORM SUBROUTINE GEOM A2
A=A-DELA
B=B-DELA
PERFORM SUBROUTINE TRANS
PERFORM SUBROUTINE GEOM A3
REPEAT TO 162 FOR I=1,1+1,...,8
DELD=0.01*(E-D)
D=D+DELD
PERFORM SUBROUTINE GEOM A4
D=D-DELD
E=E-DELD
PERFORM SUBROUTINE GEOM A5
E=E+DELD
DELF=0.01*(G-F)
F=F+DELF
PERFORM SUBROUTINE TRANS
PERFORM SUBROUTINE GEOM A6
REPEAT TO 163 FOR I=1,1+1,...,8
G=G-DELF
PERFORM SUBROUTINE GEOM A7
G=G+DELF
REPEAT TO 164 FOR I=1,1+1,...,8


CHAN(I,1)=(A2(I)-A1(I))/DELA
CHAN(I,2)=(A3(I)-A1(I))/(-DELA)
CHAN(I,3)=(A4(I)-A1(I))/DELD
CHAN(I,4)=(A5(I)-A1(I))/(-DELD)
CHAN(I,5)=(A6(I)-A1(I))/DELF
CHAN(I,6)=(A7(I)-A1(I))/(-DELF)
REPEAT TO 164 FOR J=1,1+1,...,6
164
DCHS(I,J)=CHAN(I,J)
GO TO 167
165
REPEAT TO 166 FOR I=1,1+1,...,6
REPEAT TO 166 FOR J=1,1+1,...,6
166
CHAN(I,J)=DCHS(I,J)
167
PERFORM SUBROUTINE INVCRT JUMP,6
A=A+CH1(1)
B=B+CH1(2)
D=D+CH1(3)
E=E+CH1(4)
F=F+CH1(5)
G=G+CH1(6)
PERFORM SUBROUTINE COMPAT
PERFORM SUBROUTINE TRANS
GO TO 151
171
PERFORM SUBROUTINE GEOM A1
R1=RSSERR(6)
01 WRITE
WRITE(6,172)(DATA(I),I=1,8),(A1(J),J=1,6),R1
GO TO 174
YES
ILAST.NE.1
NO
A1(4)=A1(4)+PIREQ
A1(5)=A1(5)+ETAIRQ
A1(6)=A1(6)+ETAORQ
RETURN
174
CONTINUE
GO TO 175
YES
R1.GT.GTEST1
NO
ILAST=1
GO TO 171
175
REPEAT TO 176 FOR I=1,1+1,...,8
NO
ICOUNT.EQ.0
YES
BEST(I)=DATA(I)


176
P1(I)=DATA(I)
JUMP=0
REPEAT TO 177
FOR
I=1,I+1,...,6
ABS(A1(I)).GT.0.04.AND.((MODE.EQ.2).OR.(MODE.EQ.1.AND.ICOUNT.GT.4))
IS
NO
YES
JUMP=1
177
CH1(I)=(-A1(I))
ICOUNT.EQ.0
RBEST=R1
R1.GE.RBEST
GO TO
179
REPEAT TO 178
FOR
I=1,I+1,...,6
176
BEST(I)=DATA(I)
RBEST=R1
179
CONTINUE
JUMP=1
ICOUNT.GE.4.AND.R1.GT.(0.5*R2)
R2=R1
ICOUNT=ICOUNT+1
JUMP.EQ.1
MCOUNT=MCOUNT+1
ICOUNT.LE.40.AND.MCOUNT.LE.4
GO TO
181
REPEAT TO 180
FOR
I=1,I+1,...,6
ILAST=1
GO TO 171
181
CONTINUE
JUMP.EQ.0
GO TO
185
DELA=0.01*(B-A)
B=B-DELA
PERFORM SUBROUTINE
GEOM
A2
B=B+DELA
DELC=0.05*(C-B)
C=C-DELC
PERFORM SUBROUTINE
GEOM
A3
C=C+DELC
DELD=0.01*(E-D)
D=D+DELD
PERFORM SUBROUTINE
GEOM
A4
D=D-DELD
DELF=0.01*(G-F)
F=F+DELF
PERFORM SUBROUTINE
GEOM
A5


F=F+DELF
DELH=DELG/(N-6)
G=G+DELH
PERFORM SUBROUTINE
GEOM
A6
G=G-DELH
H=H+DELH
PERFORM SUBROUTINE
GEOM
A7
H=H-DELH
REPEAT TO 104
FOR
I=1,1+1,...,6
CHAN(I,1)=(A2(I)-A1(I))/(-DELA)
CHAN(I,2)=(A3(I)-A1(I))/(-DELC)
CHAN(I,3)=(A4(I)-A1(I))/DELD
CHAN(I,4)=(A5(I)-A1(I))/DELF
CHAN(I,5)=(A6(I)-A1(I))/DELH
CHAN(I,6)=(A7(I)-A1(I))/DELH
REPEAT TO 104
FOR
J=1,1+1,...,6
104
DUMB(I,J)=CHAN(I,J)
GO TO 107
105
REPEAT TO 106
FOR
I=1,1+1,...,6
REPEAT TO 106
FOR
J=1,1+1,...,6
106
CHAN(I,J)=DUMB(I,J)
107
PERFORM SUBROUTINE
INVCAT
JUMP,6
B=B+CH1(1)
C=C+CH1(2)
D=D+CH1(3)
G=G+CH1(5)
F=F+CH1(4)
H=H+CH1(6)
PERFORM SUBROUTINE
COMPAT
PERFORM SUBROUTINE
TRANS
GO TO 171

SYMBOLS USED IN SUBROUTINE INNER (DECK IDENTIFICATION 55V5)

A	R	C	POTENTIAL AT A (MAPPING CONSTANT)	INNER
AA	R	C	ARRAY FOR RESULTS OF GEOMETRY AND DRAG INTEGRALS	INNER
ABSC	R	C	ABSCISSA ARRAY FOR NUMERICAL INTEGRALS	INNER
AI	R	C	POTENTIAL AT I (MAPPING CONSTANT)	INNER
ALOAD	R	C	STORAGE ARRAY FOR POTENTIAL IN THE REAL (PHYSICAL) PLANE	INNER
A1	R	U	ANSWER ARRAY FROM SUBROUTINE GEOM	INNER
A2	R	C	ANSWER ARRAY FROM SUBROUTINE GEOM	INNER
A3	R	C	ANSWER ARRAY FROM SUBROUTINE GEOM	INNER
A4	R	C	ANSWER ARRAY FROM SUBROUTINE GEOM	INNER
A5	R	C	ANSWER ARRAY FROM SUBROUTINE GEOM	INNER
A6	R	C	ANSWER ARRAY FROM SUBROUTINE GEOM	INNER
A7	R	C	ANSWER ARRAY FROM SUBROUTINE GEOM	INNER
A8	R	C	ANSWER ARRAY FROM SUBROUTINE GEOM	INNER
B	R	C	POTENTIAL AT B (MAPPING CONSTANT)	INNER
BEST	R	C	VALUES OF POTENTIALS GIVING LOWEST ERROR IN REQD CONDITIONS	INNER
C	R	C	POTENTIAL AT C (MAPPING CONSTANT)	INNER
CG	R	C	COSINE OF DIHEDRAL ANGLE	INNER
CHAN	R	C	THE CORRECTION MATRIX DC-SUB-I / DP-SUB-J	INNER
CHI	R	C	ERROR ARRAY FOR THE REQUIRED GEOMETRY CONDITIONS	INNER
CONI	R	C	FACTOR FOR IMAJINARY PART OF COMPLEX DISTANCE	INNER
CONR	R	C	FACTOR FOR REAL PART OF COMPLEX DISTANCE	INNER
C1	R	C	CONSTANT THAT IS COMPUTER HARDWARE DEPENDENT	INNER
C3	R	C	CONSTANT THAT IS COMPUTER HARDWARE DEPENDENT	INNER
D	R	C	POTENTIAL AT D (MAPPING CONSTANT)	INNER
DATA	R	C	ARRAY FOR INPUT DATA	INNER
DEGRAD	R	U	CONVERSION FACTOR DEGREES TO RADIANS	INNER
DELA	R	U	POTENTIAL INCREMENT NEAR A	INNER
DELC	R	U	POTENTIAL INCREMENT NEAR C	INNER
DELD	R	U	POTENTIAL INCREMENT NEAR D	INNER
DELE	R	U	POTENTIAL INCREMENT NEAR E	INNER
DELF	R	U	POTENTIAL INCREMENT NEAR F	INNER
DELG	R	U	POTENTIAL INCREMENT NEAR G	INNER
DELH	R	U	POTENTIAL INCREMENT NEAR H	INNER
DUMB	R	C	TEMPORARY STORAGE FOR THE CORRECTION MATRIX	INNER
E	R	C	POTENTIAL AT E (MAPPING CONSTANT)	INNER
ETAIRQ	R	C	INBOARD SPANWISE LOCATION (NONDIMENSIONAL)	INNER
ETAORQ	R	C	OUTBOARD SPANWISE LOCATION (NONDIMENSIONAL)	INNER
F	R	C	POTENTIAL AT F (MAPPING CONSTANT)	INNER

SYMBOLS USED IN SUBROUTINE INNER (DECK IDENTIFICATION 55YL)

G	R	C	POTENTIAL AT G (MAPPING CONSTANT)	INNER
GAMMA	R	C	DIHEDRAL ANGLE IN DEGREES	INNER
GP	R	C	DIHEDRAL ANGLE DIVIDED BY PI	INNER
GPM2	R	C	GP - 0.5	INNER
GPP2	R	C	GP + 0.5	INNER
GTEST1	R	C	TEST VALUE FOR SQUARE ROOT OF SUM OF SQUARES OF ERRORS	INNER
H	R	C	POTENTIAL AT H (MAPPING CONSTANT)	INNER
I	I	U	DO LOOP COUNTER	INNER
ICOUNT	I	U	COUNTER FOR NUMBER OF ITERATIONS FOR A GIVEN CONFIGURATION	INNER
ILAST	I	C	FLAG FOR LAST ITERATION	INNER
ININT	I	C	NUMBER OF ORDINATES IN NUMERICAL INTEGRALS	INNER
JUMP	I	U	FLAG TO INDICATE CALCULATION OF NEW INFLUENCE MATRIX	INNER
MCOUNT	I	U	COUNTER FOR THE NUMBER OF INFLUENCE MATRICES PER CASE	INNER
MODE	I	C	PROGRAM MODE OF OPERATION	INNER
NTYPE	I	C	CONFIGURATION TYPE	INNER
OMGP	R	C	1.0 - GP	INNER
ORD	R	C	ORDINATE ARRAY FOR NUMERICAL INTEGRALS	INNER
PI	R	U	RATIO OF CIRCUMFERENCE OF A CIRCLE TO ITS DIAMETER	INNER
PIREQ	R	C	INBOARD LENGTH OR DISTANCE (NONDIMENSIONAL)	INNER
POREQ	R	C	OUTBOARD LENGTH OR DISTANCE (NONDIMENSIONAL)	INNER
P1	R	C	TEMPORARY STORAGE FOR THE CURRENT VALUES OF THE POTENTIALS	INNER
RBEST	R	U	LOWEST VALUE OF SQUARE ROOT OF SUM OF SQUARES OF ERRORS	INNER
R1	R	U	CURRENT VALUE OF SQUARE ROOT OF SUM OF SQUARES OF ERRORS	INNER
R2	R	U	LAST CYCLES VALUE OF SQUARE ROOT OF SUM OF SQUARES OF ERRORS	INNER
TAB	R	C	INTERPOLATION ARRAY FOR INDEPENDENT VARIABLE AND FUNCTION	INNER
TABLE	R	C	INTEGRATED VALUES OF ORDINATES AS A FUNCTION OF ABSCISSA	INNER

Subroutine GEOM (ANS) (Deck 55VM). This subroutine evaluates the lengths $\overline{AB}$, $\overline{BC}$, . . . in the physical plane (See Figure 2) and computes the errors in the required geometry. If the iteration cycle is the last one (parameter ILAST = 1), subroutine STORE is called to store data in the ALOAD array from the numerical integrals that produce lengths in the physical plane. These data are used in the interpolation to obtain the loading to give minimum induced drag, and are stored as potential, less a quantity corresponding to the second term on the right side of Equation 6, as a function of geometric location in the real plane.

```
      SUBROUTINE GEOM (ANS)                                             55VM0010
C                                                                       55VM0020
C     THIS SUBROUTINE EVALUATES GEOMETRY IN THE TREFFTZ PLANE           55VM0030
C                                                                       55VM0040
C                                                                       55VM0050
000000        ... ANS(16)                                               55VM0060
C                                                                       55VM0070
000000        COMMON DATA(17), ABSC(50), ORD(50), TABLE(50), ALOAD(1000), AA(16)55VM0080
     A , TAB(101), GP, GPM2, GPP2, CG, OMGP, CONR, CONI, ILAST, C2, C1  55VM0090
     B , C3, R1                                                         55VM0100
C                                                                       55VM0110
000000        EQUIVALENCE              (DATA( 1),       A) , (DATA( 2),       B)  55VM0120
     A , (DATA( 3),       C) , (DATA( 4),       D) , (DATA( 5),       E)  55VM0130
     B , (DATA( 6),       F) , (DATA( 7),       G) , (DATA( 8),       H)  55VM0140
     C , (DATA( 9),      AI) , (DATA(10),    MODE) , (DATA(11),   NTYPE)  55VM0150
     D , (DATA(12),   ININT) , (DATA(13), ETAIRO) , (DATA(14), ETAORO)  55VM0160
     E , (DATA(15),   PIREO) , (DATA(16),   POREO) , (DATA(17),   GAMMA)  55VM0170
C                                                                       55VM0180
000000        DATA ZERO, ONE / 0.0, 1.0 /                               55VM0190
C                                                                       55VM0200
000000        NINT = ININT                                              55VM0210
000005        GO TO 6                                                   55VM0220
000005      4 WRITE (6,5) (DATA(I),I=1,9)                               55VM0230
000017      5 FORMAT (1H0, 5X, 37HTHIS PASS REJECTED BY SUBROUTINE GEOM  / 4X,  55VM0240
     A  9E14.5//)                                                       55VM0250
000017        RETURN                                                    55VM0260
000020      6 ITEST = 1                                                 55VM0270
C                                                                       55VM0280
C     TRANSFER TO CODING FOR SPECIFIC CONFIGURATION                     55VM0290
C                                                                       55VM0300
000021        GO TO ( 7, 11, 15, 19, 21, 23, 25) , NTYPE                55VM0310
000035      7 CONTINUE                                                  55VM0320
C                                                                       55VM0330
C     WING WITH ONE PYLON                                               55VM0340
C                                                                       55VM0350
000035        IF (B .GE. 0.0 .AND. C .GE. B .AND. D .GE. C .AND. D .LE. 1.0  55VM0360
     A      .AND. F .GE. 1.0) ITEST = 0                                 55VM0370
000045        IF (ITEST .EQ. 1) GO TO 4                                 55VM0380
C                                                                       55VM0390
C     INTEGRATION OF TRANSFORMATION SLOPE TO GIVE TREFFTZ PLANE GEOMETRY55VM0400
C                                                                       55VM0410
C     INTEGRATION FROM A TO B IN TWO PARTS                              55VM0420
C                                                                       55VM0430
000047        ALPHA = 0.5 * (A + D)                                     55VM0440
000072        CALL NUMINT (ALPHA,     A,     A, NINT, 4, AB1, GT1)      55VM0450
000101        IF (ILAST .EQ. 1) CALL STORE (GT1,     A, 1)              55VM0460
000107        IF(ABS(ALPHA-A).GT.C1)    AB1 =    AB1 + 2.0 * GT1 * SQRT(ALPHA-A)  55VM0470
000123        CALL NUMINT (     B,ALPHA,     B, NINT, 5, AB2, GT2)      55VM0480
000132        IF (ILAST .EQ. 1) CALL STORE (GT2,     B, 2)              55VM0490
000140        IF(ABS(B-ALPHA).GT.C1)    AB2 =    AB2 + 2.0 * GT2 * SQRT(B-ALPHA)  55VM0500
000154        ADINT = AB1 + AB2                                         55VM0510
000156        CALL NUMINT (     C,     D,     D, NINT, 6, BCINT, GT )   55VM0520
000165        IF (ILAST .EQ. 1) CALL STORE (GT,     B, 3)               55VM0530
000173        IF(ABS(C - B  ).GT.C1) BCINT = BCINT + 2.0 * GT  * SQRT(C  -  B)  55VM0540
000207        BCINT = (-BCINT)                                          55VM0550
000210        CALL NUMINT (D, C, D, NINT, 7, CDINT, GT)                 55VM0560
000217        IF (ILAST .EQ. 1) CALL STORE (GT ,     D, 4)              55VM0570
000225        IF(ABS(  D - C).GT.C1) CDINT = CDINT + 2.0 * GT  * SQRT(  D - C)  55VM0580
```

```
000241          CDINT = (-CDINT)                                                55VM0580
000242          CALL NUMINT (ONE, D, D, NINT, 8, DEINT, GT)                     55VM0590
000251          IF (ILAST .EQ. 1) CALL STORE (GT ,    D, 5)                     55VM0600
000257          IF(ABS(1.0 - D).GT.C1) DEINT = DEINT + 2.0 * GT  * SQRT(1.0 - D) 55VM0610
000273          DEINT = (-DEINT)                                                55VM0620
000274          CALL NUMINT (F, ONE, F, NINT, 9, EFINT, GT)                     55VM0630
000303          IF (ILAST .EQ. 1) CALL STORE (GT ,    F, 6)                     55VM0640
000311          IF(ABS(F - 1.0).GT.C1) EFINT = EFINT + 2.0 * GT  * SQRT(F - 1.0) 55VM0650
000325          EFINT = (-EFINT)                                                55VM0660
000326          B2BAR = 0.5 * (ABINT + DEINT - EFINT)                           55VM0670
000332          ANS( 1) = (BCINT + CDINT) / B2BAR                               55VM0680
000336          ANS( 2) = (ABINT + DEINT + EFINT) / B2BAR                       55VM0690
000341          ANS( 3) = ABINY / B2BAR - ETAIRQ                                55VM0700
000345          ANS( 4) = 0.5 * (CDINT - BCINT) / B2BAR - PIRFQ                 55VM0710
000350          ANS( 6) = ABINT                                                 55VM0720
000352          ANS( 7) = BCINT                                                 55VM0730
000353          ANS( 8) = CDINT                                                 55VM0740
000354          ANS( 9) = DEINT                                                 55VM0750
000356          ANS(10) = EFINT                                                 55VM0760
000357          RETURN                                                          55VM0770
000360       11 CONTINUE                                                        55VM0780
        C                                                                       55VM0790
        C       WING WITH TWO PYLONS                                            55VM0800
        C                                                                       55VM0810
000360          IF (A .LE. B .AND. B .LE. C .AND. C .LE. 0.0 .AND. D .GE. 0.0   55VM0820
               A   .AND. F .GE. E .AND. F .LE. 1.0 .AND. H .GE. 1.0 .AND. AI .GE. 55VM0830
               B   H) ITEST = 0                                                 55VM0840
000430          IF (ITEST .EQ. 1) GO TO 4                                       55VM0850
        C                                                                       55VM0860
        C       INTEGRATION OF TRANSFORMATION SLOPE TO GIVE TREFFTZ PLANE GEOMETRY55VM0870
        C                                                                       55VM0880
000432          ALPHA = 0.5 * (A + B)                                           55VM0890
000435          CALL NUMINT (ALPHA, A, A, NINT, 15, AB1, GT1)                   55VM0900
000444          IF (ILAST .EQ. 1) CALL STORE (GT1,    A, 1)                     55VM0910
000452          CALL NUMINT (B, ALPHA, B, NINT, 16, AB2, GT2)                   55VM0920
000461          IF (ILAST .EQ. 1) CALL STORE (GT2,    B, 2)                     55VM0930
000467          IF(ABS(ALPHA-A).GT.C1)   AB1 =   AB1 + 2.0 * GT1 * SQRT(ALPHA-A) 55VM0940
000503          IF(ABS(B-ALPHA).GT.C1)   AB2 =   AB2 + 2.0 * GT2 * SQRT(B-ALPHA) 55VM0950
000517          ABINT = (-AB1) - AB2                                            55VM0960
000522          CALL NUMINT (C, B, B, NINT, 17, BCINT, GT)                      55VM0970
000530          IF (ILAST .EQ. 1) CALL STORE (GT ,    B, 3)                     55VM0980
000536          IF(ABS(  C - B).GT.C1) BCINT = BCINT + 2.0 * GT  * SQRT(  C - B) 55VM0990
000552          CALL NUMINT (ZERO, C, ZERO, NINT, 18, CDINT, GT)                55VM1000
000561          IF (ILAST .EQ. 1) CALL STORE (GT , ZERO, 4)                     55VM1010
000567          IF(ABS(    - C).GT.C1) CDINT = CDINT + 2.0 * GT  * SQRT(    - C) 55VM1020
000601          E2 = E / 2.0                                                    55VM1030
000603          CALL NUMINT (E2, ZERO, ZERO, NINT, 19, DE1, GT1)                55VM1040
000612          IF (ILAST .EQ. 1) CALL STORE (GT1, ZERO, 5)                     55VM1050
000620          CALL NUMINT (E, E2, E, NINT, 20, DE2, GT2)                      55VM1060
000627          IF (ILAST .EQ. 1) CALL STORE (GT2,    E, 6)                     55VM1070
000635          IF(ABS(     E2).GT.C1)   DE1 =   DE1 + 2.0 * GT1 * SQRT(     E2) 55VM1080
000646          IF(ABS(     E2).GT.C1)   DE2 =   DE2 + 2.0 * GT2 * SQRT(     E2) 55VM1090
000657          DEINT = DE1 + DE2                                               55VM1100
000661          CALL NUMINT (F, E, E, NINT, 21, EFINT, GT)                      55VM1110
000670          IF (ILAST .EQ. 1) CALL STORE (GT ,    E, 7)                     55VM1120
000676          IF(ABS(  F - E).GT.C1) EFINT = EFINT + 2.0 * GT  * SQRT(  F - E) 55VM1130
000712          EFINT = (-EFINT)                                                55VM1140
000713          CALL NUMINT (ONE, F, ONE, NINT, 22, FGINT, GT)                  55VM1150
```

```
      IF (ILAST .EQ. 1) CALL STORE (GT ,   ONE, 8)
      IF(ABS(1.0 - F).GT.C1) FGINT = FGINT + 2.0 * GT  * SQRT(1.0 - F)
      FGINT = (-FGINT)
      CALL NUMINT (H, ONE, ONE, NINT, 23, GHINT, GT)
      IF (ILAST .EQ. 1) CALL STORE (GT ,   ONE, 9)
      IF(ABS(H - 1.0).GT.C1) GHINT = GHINT + 2.0 * GT  * SQRT(H - 1.0)
      GHINT = (-GHINT)
      CALL NUMINT (AI, H, AI, NINT, 24, HIINT, GT)
      IF (ILAST .EQ. 1) CALL STORE (GT ,   AI,10)
      IF(ABS( AI - H).GT.C1) HIINT = HIINT + 2.0 * GT  * SQRT( AI - H)
      HIINT = (-HIINT)
      B2BAR = 0.5 * (ABINT + DEINT + GHINT - HIINT)
      ANS( 1) = (BCINT + CDINT) / B2BAR
      ANS( 2) = (EFINT + FGINT) / B2BAR
      ANS( 3) = (ABINT + DEINT + GHINT + HIINT) / B2BAR
      ANS( 4) = ABINT / B2BAR -ETAIRQ
      ANS( 5) = 1.0 - GHINT / B2BAR - ETAORQ
      ANS( 6) = 0.5 * (CDINT - BCINT) / B2BAR - PIRFQ
      ANS( 7) = 0.5 * (FGINT - EFINT) / B2BAR - PORFQ
      ANS( 9) = ABINT
      ANS(10) = BCINT
      ANS(11) = CDINT
      ANS(12) = DEINT
      ANS(13) = EFINT
      ANS(14) = FGINT
      ANS(15) = GHINT
      ANS(16) = HIINT
      RETURN
   15 CONTINUE
C
C     WING WITH HORIZONTAL
C
      IF (A .LE. H .AND. B .LE. 0.0 .AND. D .GT. 0.0 .AND. D .LE. 1.0
     A    .AND. F .GT. 1.0) ITEST = 0
      IF (ITEST .EQ. 1) GO TO 4
C
C     INTEGRATION OF TRANSFORMATION SLOPE TO GIVE TREFFTZ PLANE GEOMETRY
      CALL NUMINT (B, A, A, NINT, 26, ABINT, GT)
      IF (ILAST .EQ. 1) CALL STORE (GT ,   A, 1)
      IF(ABS(  B - A).GT.C1) ABINT = ABINT + 2.0 * GT  * SQRT(  B - A)
      CALL NUMINT (ZERO, B, ZERO, NINT, 29, BCINT, GT)
      IF (ILAST .EQ. 1) CALL STORE (GT , ZERO, 2)
      IF(ABS(    - B).GT.C1) BCINT = BCINT + 2.0 * GT  * SQRT(    - B)
      D2 = D / 2.0
      CALL NUMINT (D2, ZERO, ZERO, NINT, 30, CD1, GT1)
      IF (ILAST .EQ. 1) CALL STORE (GT1, ZERO, 3)
      CALL NUMINT (D, D2, D, NINT, 31, CD2, GT2)
      IF (ILAST .EQ. 1) CALL STORE (GT2,   D, 4)
      IF(ABS(     D2).GT.C1)   CD1 =   CD1 + 2.0 * GT1 * SQRT(     D2)
      IF(ABS(     D2).GT.C1)   CD2 =   CD2 + 2.0 * GT2 * SQRT(     D2)
      CDINT = (-CD1) - CD2
      CALL NUMINT (ONE, D, D, NINT, 32, DEINT, GT)
      IF (ILAST .EQ. 1) CALL STORE (GT ,   D, 5)
      IF(ABS(1.0 - D).GT.C1) DEINT = DEINT + 2.0 * GT  * SQRT(1.0 - D)
      DEINT = (-DEINT)
      CALL NUMINT (F, ONE, F, NINT, 33, EFINT, GT)
      IF (ILAST .EQ. 1) CALL STORE (GT ,   F, 6)
```

```
      IF(ABS(F - 0.5).GT.0.1) EFINT = EFINT + 0.0 * GT  * SQRT(F * 1.0)
      EFINT = 1-EFINT
      DELAR = 0.0 * (BBINT + BCINT)
      ANS( 1) = (BBINT + BCINT) / DELAR
      ANS( 2) = (BCINT + EFINT) / DELAR
      ANS( 3) = BCINT / DELAR - DTAING
      ANS( 4) = 0.0 * (BCINT - EFINT) / DELAR - PIAFG
      ANS( 5) = BBINT
      ANS( 6) = BCINT
      ANS( 7) = BCINT
      ANS( 8) = BCINT
      ANS( 9) = BCINT
      ANS(10) = EFINT
      RETURN
   10 CONTINUE
C
C     WING WITH PYLON ON TOP AND ON BOTTOM
C
      IF (A .LE. 0 .AND. B .LE. 0 .LE. C .AND. C .LE. 0.0 .AND. E .GE. 0.0
     ..AND. E .LE. F .AND. F .LE. 1.0 .AND. A .GE. 1.0 .AND.
     .   B .LE. A) ITEST = 0
      IF (ITEST .EQ. 1) GO TO 4
C
C     INTEGRATION OF TRANSFORMATION SLOPE TO GIVE TRAPPTY PLANE GEOMETRY
C
      ALPHA = 0.5 * (A + B)
      CALL NUMINT (ALPHA, AA,  B, MINTS, AA,     AA1, GT1)
      IF (ITEST .EQ. 1) CALL STORE (GT1,     AA, 1)
      CALL NUMINT (AA, ALPHA, B, MINTS, AA,     AA2, GT2)
      IF (ITEST .EQ. 1) CALL STORE (GT2,     AA, 2)
      IF(ABS(ALPHA-A).GT.0.1)    AA1 =    AA1 * 2.0 * GT1 * SQRT(ALPHA-A)
      IF(ABS(A-ALPHA).GT.0.1)    AA2 =    AA2 * 2.0 * GT2 * SQRT(A-ALPHA)
      AAINT = (AA1) - AA2
      CALL NUMINT (0, AA,     B, MINTS, AA,  BBINT, GT 1)
      IF (ITEST .EQ. 1) CALL STORE (GT,     AA, 3)
      IF(ABS(C-0).      .GT.0.1) BBINT = BBINT * 2.0 * GT  * SQRT(C-A)
      CALL NUMINT (ZERO,C,ZERO, MINTS, AA,  BCINT, GT 1)
      IF (ITEST .EQ. 1) CALL STORE (GT,     ZERO, 4)
      IF(ABS(F-0)       .GT.0.1) BCINT = BCINT * 2.0 * GT  * SQRT(F)
      CALL NUMINT (ZERO,ZERO, F, MINTS, AA,  CFINT, GT 1)
      IF (ITEST .EQ. 1) CALL STORE (GT,     ZERO, 5)
      IF(ABS(F-1)       .GT.0.1) CFINT = CFINT * 2.0 * GT  * SQRT(F-1)
      CALL NUMINT (ONE, F,     F, MINTS, AA,  FGINT, GT 1)
      IF (ITEST .EQ. 1) CALL STORE (GT,     F, 6)
      IF(ABS(G-F)       .GT.0.1) FGINT = FGINT * 2.0 * GT  * SQRT(G-F)
      FGINT = (1-FGINT)
      CALL NUMINT (AA, ONE,    B, MINTS, AA,  GAINT, GT 1)
      IF (ITEST .EQ. 1) CALL STORE (GT,     AA, 7)
      IF(ABS(A-G)       .GT.0.1) GAINT = GAINT * 2.0 * GT  * SQRT(A-G)
      GAINT = (1-GAINT)
      ALPHA = 0.5 * (A + B)
      CALL NUMINT (ALPHA,     B, MINTS, AA,     AA1, GT1)
      IF (ITEST .EQ. 1)       STORE (GT1,     AA, 1)
      CALL NUMINT (A,ALPHA,A, MINTS, AA,     AA2, GT2)
      IF (ITEST .EQ. 1) CALL STORE (GT2,     AA,2)
      IF(ABS(ALPHA-A).GT.0.1)    AA1 =    AA1 * 2.0 * GT1 * SQRT(ALPHA-A)
```

```
      IF(ABS(ALPHAW).LT.DT)  AW2 = AW2 * COS * DT2 * SQRT(ALPHAW)
      FINT = (FINT) * AW2
      WW = WW * COS * (WWINT * SZINT * EPINT - HZINT)
      WWDT(1) = (SCINT * CSINT) / WWW
      WWDT(2) = (FINT * CRINT) / WWW
      WWDT(3) = WWINT * SZINT * EPINT * WZINT / WWWR
      WWDT(4) = WWINT / WWWR - ETAREQ
      WWDT(5) = (WZINT) / WWWR - ETAREQ
      WWDT(6) = 0.0 * (CDINT - FCINT) / WWWR - FDREQ
      WWDT(7) = 0.0 * (FDINT - GAINT) / WWWR - FGREQ
      WWDT(8) = WZINT
      WWDT(9) = RZINT
      WWDT(10) = CZINT
      WWDT(11) = DZINT
      WWDT(12) = EZINT
      WWDT(13) = FZINT
      WWDT(14) = GZINT
      WWDT(15) = HZINT
      RETURN
   21 CONTINUE

C
C     WING WITH PYLON AND OUTBOARD DIAGONAL
C
      IF (W.LEE.0.0AND.B.LEE.100.0AND.C.LEE.0.0AND.D.LEE.0.0AND
     A.E.LEE.0.0AND.E.LEE.100.0AND.F.LEE.100.0AND.G.LEE.F
     B.AND.H.EQ.0
      IF (ETA.GT.LEE) GO TO 4
C
C     INTEGRATION OF TRANSFORMATION SLOPE TO GIVE TRAFFIC-PLANE DISPLACEMENT
C
      ALPHA = 0.0 * 1.0 * D1
      CALL NUMINT (ALPHAW, AW, W, W, ALPHA, DIV, ALPHA, DT)
      IF (ALPHAT.LEE.0) CALL STORE (DT, ALPHAW)
      CALL NUMINT (ALPHAW, ALPHAW, ALPHAW, ALPHAW, DIV, ALPHA, DT)
      IF (ALPHAT.LEE.0) CALL STORE (DT, ALPHAW)
      IF(ABS(ALPHAW).LT.DT) AW2 = AW2 * COS * DT2 * SQRT(ALPHAW)
      WWINT = AW2 * WW
      CALL NUMINT (DV, ZERO, ZERO, WWINT, DIV, WWINT, DT)
      IF (ALPHAT.LEE.0) CALL STORE (DT, WWINT)
      IF(ABS(WWINT).LT.DT) WCINT = WCINT * COS * DT * ABS(WWINT)
      WCINT = (WCINT)
      CALL NUMINT (DV, DV, DV, WCINT, DIV, WCINT, DT)
      IF (ALPHAT.LEE.0) CALL STORE (DT, WCINT)
      IF(ABS(WCINT).LT.DT) DZINT = DZINT * DT * (ORP-D) ** OPP2 / OPP2
      DZINT = (DZINT)
      CALL NUMINT (DV, DV, DV, DZINT, DIV, (DZINT), DT)
      IF (ALPHAT.LEE.0) CALL STORE (DT, DZINT)
      IF(ABS(DZINT).LT.DT) EPINT = EPINT * DT * (ORP-D) ** OPP2 / OPP2
      EPINT = DZINT * DCNT
      DCNT1 = DZINT * DCNT
      CALL NUMINT (F, ORGP, F, F, DIV, DIV, EPINT, DT)
      IF (ALPHAT.LEE.0) CALL STORE (DT, F, F)
      IF(ABS(F-DIV).LT.DT) EPINT = EPINT * DT * (F-ORGP) ** ORGP / ORGP
      EPINT = DCNT * EPINT
      EPINT1 = CINT * EPINT
      ALPHA = DIV * (F * D)
      CALL NUMINT (ALPHAW, F, F, ALPHA, DIV, FDIV, DT)
```

```
      IF (ILAST .EQ. 1) CALL STORE (GT1,      ,Y, D)
      CALL NGAINT (GX, ALPHAX, 0, NLAMX, LGX, FG2, GT2)
      IF (ILAST .EQ. 1) CALL STORE (GT2,      ,X, D)
      IF(ABS(ALPHAX-A).GT.C) FG1 = FG1 * GT1 * (ALPHAX-A) * GX / GXP
      IF(ABS(G-ALPHAX).GT.C) FG2 = FG2 * 2.0 * GT2 * SQRT(G - ALPHAX)
      FGINT = (-FG1) - FG2
      FPIAR = CG * EFINT - FGINT
      ANS( 1) =  AGINT * FGINT
      ANS( 2) =  BGINT * CGINT
      ANS( 3) =  DGINT * EFINT
      ANS( 4) = CGINT / DZBAR - PIREG
      ANS( 5) = AGINT / BZBAR - ZTAIRG
      ANS( 6) = AGINT
      ANS( 7) = FGINT
      ANS( 8) = BGINT
      ANS( 9) = CGINT
      ANS(10) = DGINT
      ANS(11) = DZINTR
      ANS(12) = DZINTI
      ANS(13) = EFINT
      ANS(14) = EFINTR
      ANS(15) = EFINTI
      ANS(16) = FGINT
      RETURN
   20 CONTINUE
C
C     WING WITH INBOARD PYLON AND OUTBOARD DIAEDRAL
C
      IF ( A .LT. 0.0 .AND. C .GT. 0.0 .AND. D .GE. C .AND. C .GTE. D
     *   .AND. E .LE. 1.0 .AND. G .GE. 1.0 .AND. A .GE. 0. ITEST = 0
      IF (ITEST .EQ. 1) GO TO 4
C
C     INTEGRATION OF TRANSFORMATION SLOPE TO GIVE TAIFFTY PLANE CLOSURAY
C
C     INTEGRATION FROM A TO B IN TWO PARTS
C
      ALPHA = 0.5 * (A + B)
      CALL NGAINT (ALPHA,    0.,    0., NLAMX, LGX, AG1,    , GT1)
      IF (ILAST .EQ. 1) CALL STORE (GT1,      ,X, D)
      IF (ABS(ALPHA-A).GT.C) AG1    = AG1    * 2.0 * GT1 * SQRT(ALPHA-A)
      CALL NGAINT ( ZERO,ALPHA,  ZERO, NLAMX, LGX, AG2    , GT2)
      IF (ILAST .EQ. 1) CALL STORE (GT2,  ZERO, D)
      IF (ABS( -ALPHA).GT.C) AG2    = AG2    * 2.0 * GT2 * SQRT( -ALPHA)
      AGINT = AG1 + AG2
      CALL NGAINT (    C, ZERO,  ZERO, NLAMX, LGX, BGINT, GT )
      IF (ILAST .EQ. 1) CALL STORE (GT ,  ZERO, D)
      IF (ABS(      C).GT.C) BGINT = BGINT * 2.0 * GT  * SQRT(      C)
      BGINT = (-BGINT)
      CALL NGAINT (    D,    C,    D, NLAMX, LGX, CGINT, GT )
      IF (ILAST .EQ. 1) CALL STORE (GT ,      D, D)
      IF (ABS(D  -  C).GT.C) CGINT = CGINT * 2.0 * GT  * SQRT(D  -  C)
      CGINT = (-CGINT)
      ALPHA = 0.5 * (D + E)
      CALL NGAINT (ALPHA,    D,    E, NLAMX, LGX, DZ1    , GT1)
      IF (ILAST .EQ. 1) CALL STORE (GT ,      D, D)
      IF (ABS(ALPHA-D).GT.C) DZ1    = DZ1    * 2.0 * GT1 * SQRT(ALPHA-D)
      CALL NGAINT (    E,ALPHA,    E, NLAMX, LGX, DZ2    , GT2)
      IF (ILAST .EQ. 1) CALL STORE (GT2,      E, D)
      IF (ABS(E-ALPHA).GT.C) DZ2    = DZ2    * GT2 * (E-ALPHA)**CPP2/CPP2
```

```
      [illegible]
      CALL [illegible] (GN2, [illegible], [illegible], [illegible], 75, EFINT, GT )
      IF (ILAST .EQ. 1) CALL STORE (GT [illegible], 2, 7)
      [illegible] = [illegible] * EFINT * GT * (GN2 - [illegible])**CPP2/CPP2
      EFINT = (1-EFINT)
      CALL [illegible] ([illegible], GN2, [illegible], [illegible], 76, FOINT, GT )
      IF (ILAST .EQ. 1) CALL STORE (GT [illegible], 2, 8)
      [illegible] = [illegible] * FOINT * GT * (G - GN2)**CPA2/CPA2
      FOINT = (1-FOINT)
      ALPHA = 0.5 * (G + A)
      CALL [illegible] (ALPHA, [illegible], [illegible], [illegible], 77, GA1, GT1)
      IF (ILAST .EQ. 1) CALL STORE (GT1, [illegible], 9)
      IF (ABS(ALPHA-G).GT.C1) GA1 = GA1 * GT1 * (ALPHA-G)**CPA2/CPA2
      CALL [illegible] (A, ALPHA, [illegible], [illegible], 78, GA2, GT2)
      IF (ILAST .EQ. 1) CALL STORE (GT2, [illegible], 10)
      IF (ABS(A-ALPHA).GT.C1) GA2 = GA2 * 200 * GT2 * SQRT(A-ALPHA)
      GAINT = (1-GA1) - GA2
      [illegible] = CG * EFINT * GAINT
      [illegible] = [illegible] * EFINT * GAINT
      [illegible] = [illegible] * GAINT
      [illegible] = [illegible] * FOINT
      [illegible] = [illegible] / [illegible] - [illegible]
      [illegible] = [illegible] / [illegible] - [illegible]
      [illegible] = [illegible] / [illegible] - [illegible]
      [illegible]
      [illegible]
      [illegible]
      [illegible]
      [illegible]
      [illegible]
      [illegible]
      [illegible]
      [illegible]
      [illegible]
   25 CONTINUE
C
C     [illegible] WITH CHANGEABLE VERTICAL FLANGE AT THE TIP
C
      IF (N .EQ. [illegible] .AND. [illegible] .EQ. [illegible] .AND. [illegible] .EQ. 1)
     * [illegible] .EQ. [illegible] .AND. [illegible] .EQ. [illegible] .AND. [illegible] .EQ. 1)
      IF (ILAST .EQ. 1) GO TO 4
      ALPHA = 0.5 * (1 + G)
      CALL [illegible] (ALPHA, [illegible], [illegible], [illegible], 80, GA1, GT1)
      IF (ILAST .EQ. 1) CALL STORE (GT1, [illegible], 1)
      IF (ABS(ALPHA-1).GT.C1) GA1 = GA1 * 200 * GT1 * SQRT(ALPHA-1)
      CALL [illegible] (G, ALPHA, [illegible], [illegible], 81, GA2, GT2)
      IF (ILAST .EQ. 1) CALL STORE (GT2, [illegible], 2)
      IF (ABS(G-ALPHA).GT.C1) GA2 = GA2 * 200 * GT2 * SQRT(G-ALPHA)
      GAINT = GA1 + GA2
      ALPHA = 0.5 * (B + C)
      CALL [illegible] (ALPHA, [illegible], [illegible], [illegible], 82, GC1, GT1)
      IF (ILAST .EQ. 1) CALL STORE (GT1, [illegible], 3)
      IF (ABS(ALPHA-B).GT.C1) GC1 = GC1 * 200 * GT1 * SQRT(ALPHA-B)
      CALL [illegible] (B, ALPHA, [illegible], [illegible], 83, GC2, GT2)
      IF (ILAST .EQ. 1) CALL STORE (GT2, [illegible], 4)
      IF (ABS(C-ALPHA).GT.C1) GC2 = GC2 * 200 * GT2 * SQRT(C-ALPHA)
      GCINT = GC1 + GC2
      GCINT = (1-GCINT)
      CALL [illegible] (G, [illegible], [illegible], [illegible], 84, GDINT, GT )
      IF (ILAST .EQ. 1) CALL STORE (GT , [illegible], 5)
```

```
      IF (ABS(D  -  C).GT.C) CDINT = CDINT * 2.0 * GT  * SQRT(D  -  C)
      CDINT = (-CDINT)
      CALL NUMINT (     E,     D,     E, NINT, 88, DEINT, GT )
      IF (ILAST .EQ. 1) CALL STORE ( GT ,     E, 8)
      DEINT = (-DEINT)
      CALL NUMINT (     F,     E,     F, NINT, 89, EFINT, GT )
      IF (ILAST .EQ. 1) CALL STORE (GT ,     F, 7)
      IF (ABS(F  -  E).GT.C) EFINT = EFINT * 2.0 * GT * SQRT(F  -  E)
      EFINT = (-EFINT)
      CALL NUMINT (     G,     F,     F, NINT, 90, FGINT, GT )
      IF (ILAST .EQ. 1) CALL STORE (GT ,     F, 6)
      IF (ABS(G  -  F).GT.C) FGINT = FGINT * 2.0 * GT  * SQRT(G  -  F)
      CALL NUMINT (     H,     G,     H, NINT, 91, GHINT, GT )
      IF (ILAST .EQ. 1) CALL STORE (GT ,     H, 5)
      IF (ABS(H  -  G).GT.C) GHINT = GHINT * 2.0 * GT  * SQRT(H  -  G)
      GHINT = (-GHINT)
      SQAR = 0.5 * (ABINT - GHINT)
      ANS( 1) = (ABINT + GHINT) / SQAR
      ANS( 2) = (BCINT + FGINT) / SQAR
      ANS( 3) = (CDINT + DEINT + EFINT) / SQAR
      ANS( 4) = 0.5 * (FGINT - BCINT) / SQAR - PIRFO
      ANS( 5) = (-BCINT) / SQAR - ETAIRO
      ANS( 6) = (-EFINT) / SQAR - ETAORO
      ANS( 8) = ABINT
      ANS( 9) = BCINT
      ANS(10) = CDINT
      ANS(11) = DEINT
      ANS(12) = EFINT
      ANS(13) = FGINT
      ANS(14) = GHINT
      RETURN
      END
```

PROGRAM LENGTH
[illegible]

FUNCTION ASSIGNMENTS

STATEMENT ASSIGNMENTS

[illegible]	-	[illegible]	5	-	003774	6	-	[illegible]	7	-	000335
[illegible]	-	[illegible]	15	-	[illegible]	19	-	001415	21	-	002144
22	-	002004	25	-	003317						

BLOCK NAMES AND LENGTHS
- [illegible]

VARIABLE ASSIGNMENTS

[illegible]	-	000000C01	AA	-	[illegible]C01	ADINT	-	004170	ASSC	-	[illegible]C01
[illegible]	-	[illegible]	AA2	-	[illegible]	[illegible]	-	[illegible]C01	ALOAD	-	[illegible]C01
[illegible]	-	[illegible]	[illegible]	-	[illegible]C01	BCINT	-	[illegible]	[illegible]	-	004226
[illegible]	-	004226	[illegible]	-	[illegible]	.C	-	[illegible]C01	CCINT	-	[illegible]
[illegible]	-	[illegible]	CC2	-	[illegible]	CG	-	[illegible]C01	COLI	-	[illegible]C01
[illegible]	-	[illegible]C01	[illegible]	-	[illegible]C01	D	-	[illegible]C01	DATA	-	[illegible]C01
[illegible]	-	[illegible]	[illegible]	-	[illegible]	DIINTR	-	[illegible]	[illegible]	-	[illegible]
[illegible]	-	[illegible]	[illegible]	-	[illegible]	I	-	[illegible]C01	[illegible]	-	[illegible]
[illegible]	-	[illegible]	[illegible]	-	[illegible]	[illegible]	-	[illegible]C01	[illegible]	-	[illegible]C01
	-	[illegible]	7	-	[illegible]C01	[illegible]	-	[illegible]	[illegible]	-	[illegible]
[illegible]	-	[illegible]	[illegible]	-	[illegible]C01	[illegible]	-	[illegible]	[illegible]	-	[illegible]
[illegible]	-	[illegible]	[illegible]	-	[illegible]C01	GPP2	-	[illegible]C01	[illegible]	-	[illegible]
[illegible]	-	[illegible]	[illegible]	-	[illegible]	[illegible]	-	[illegible]C01	[illegible]	-	[illegible]
[illegible]	-	[illegible]	[illegible]	-	[illegible]	[illegible]	-	[illegible]	[illegible]	-	[illegible]C01
[illegible]	-	[illegible]C01	[illegible]	-	[illegible]	[illegible]	-	[illegible]	NTYPE	-	[illegible]
[illegible]	-	[illegible]C01	[illegible]	-	[illegible]	[illegible]	-	[illegible]C01	PIREO	-	[illegible]C01
[illegible]	-	[illegible]C01	TAU	-	002257C01	TABLE	-	[illegible]C01	ZERO	-	004101

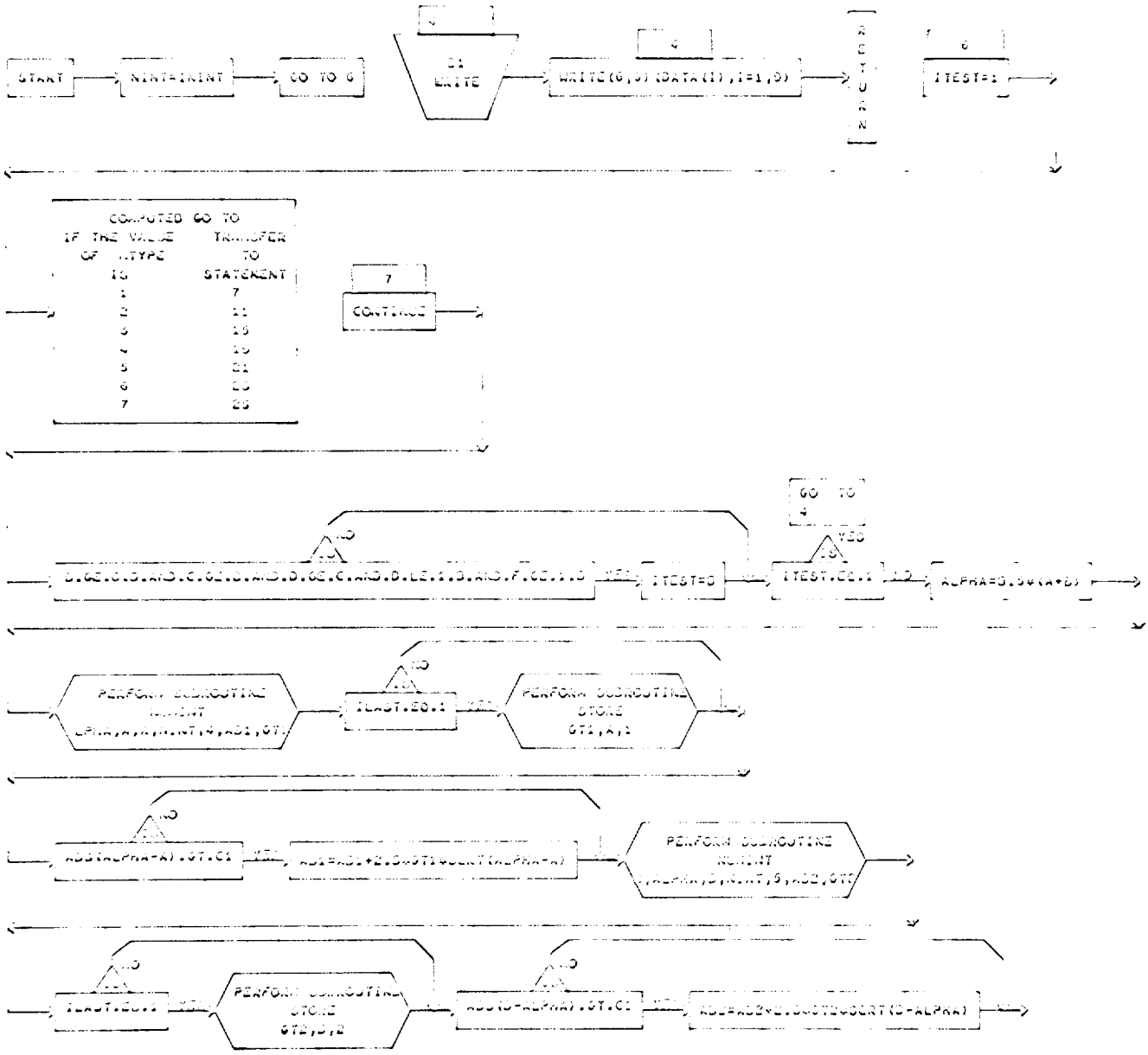
START OF CONSTANTS
[illegible]

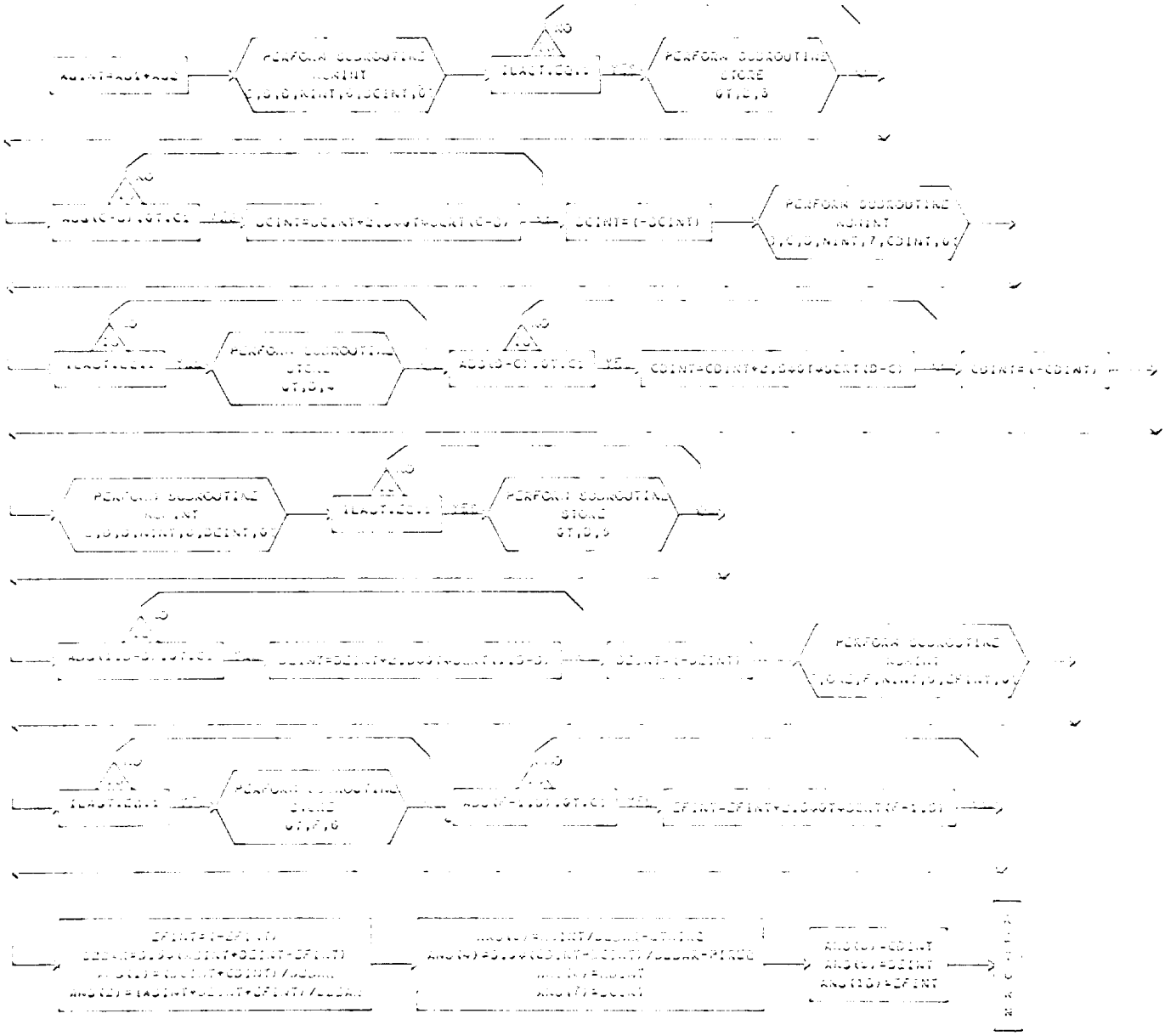
START OF TEMPORARIES
[illegible]

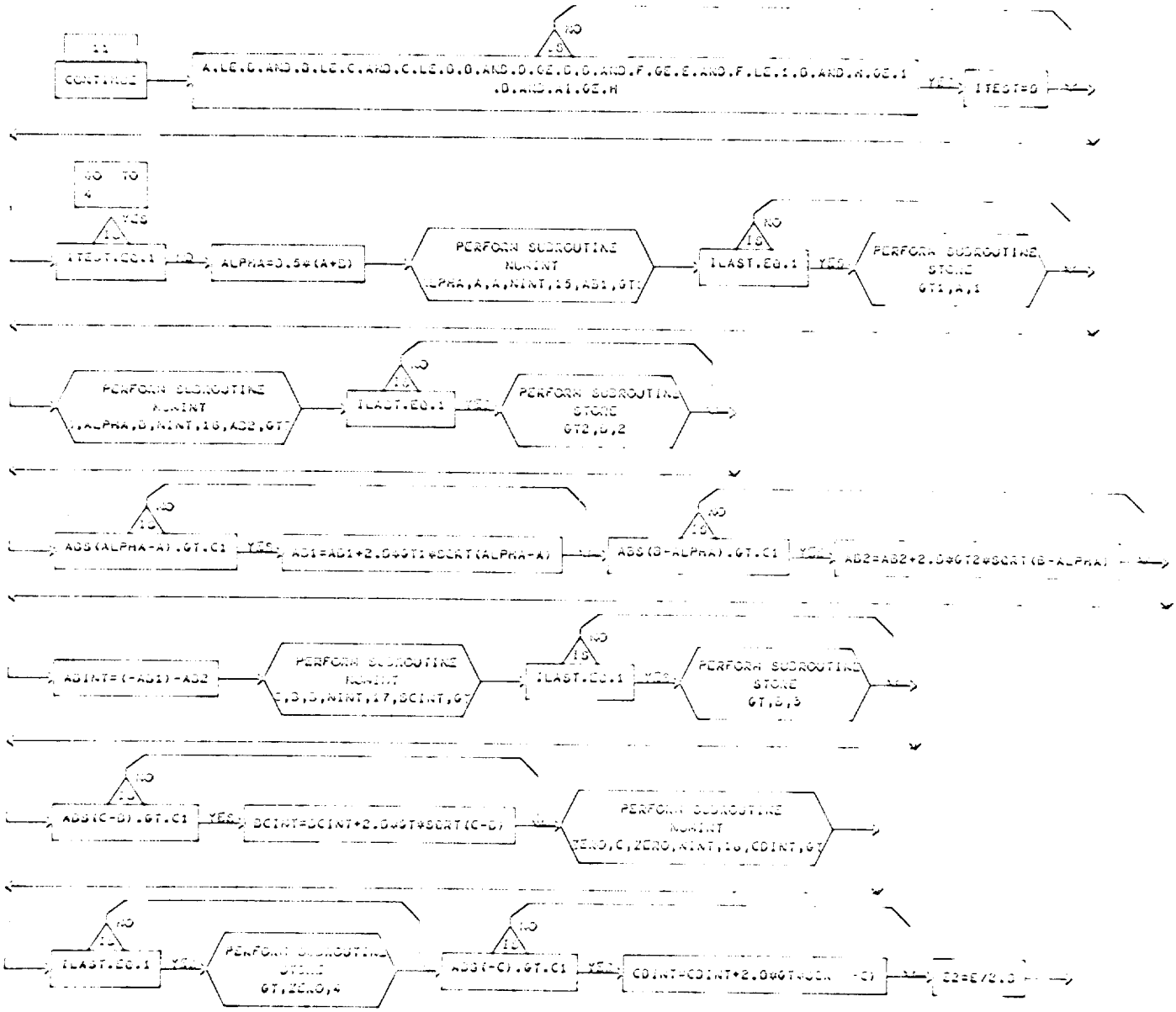
START OF INDIRECTS
[illegible]

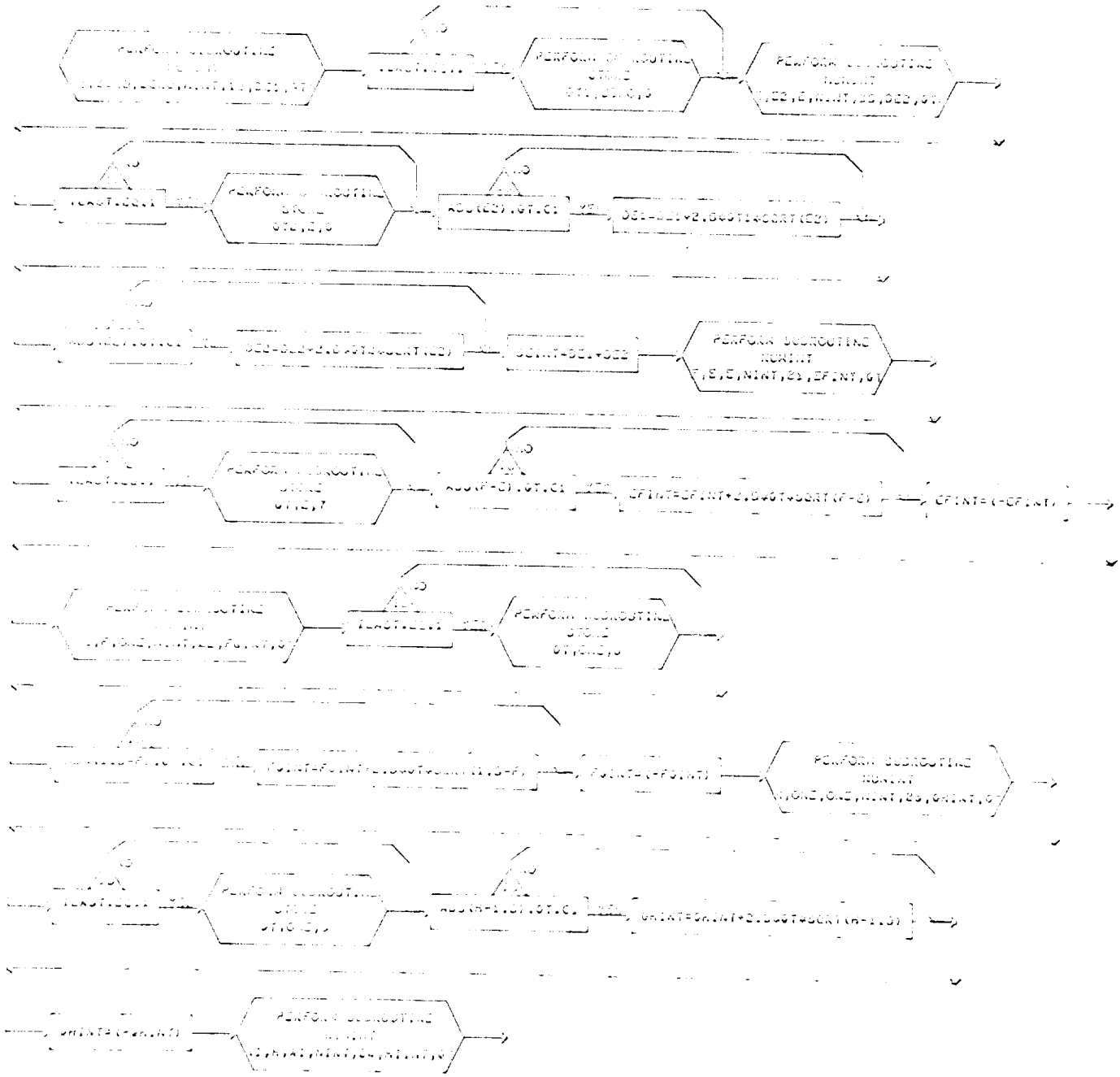
UNUSED COMPILER SPACE
[illegible]

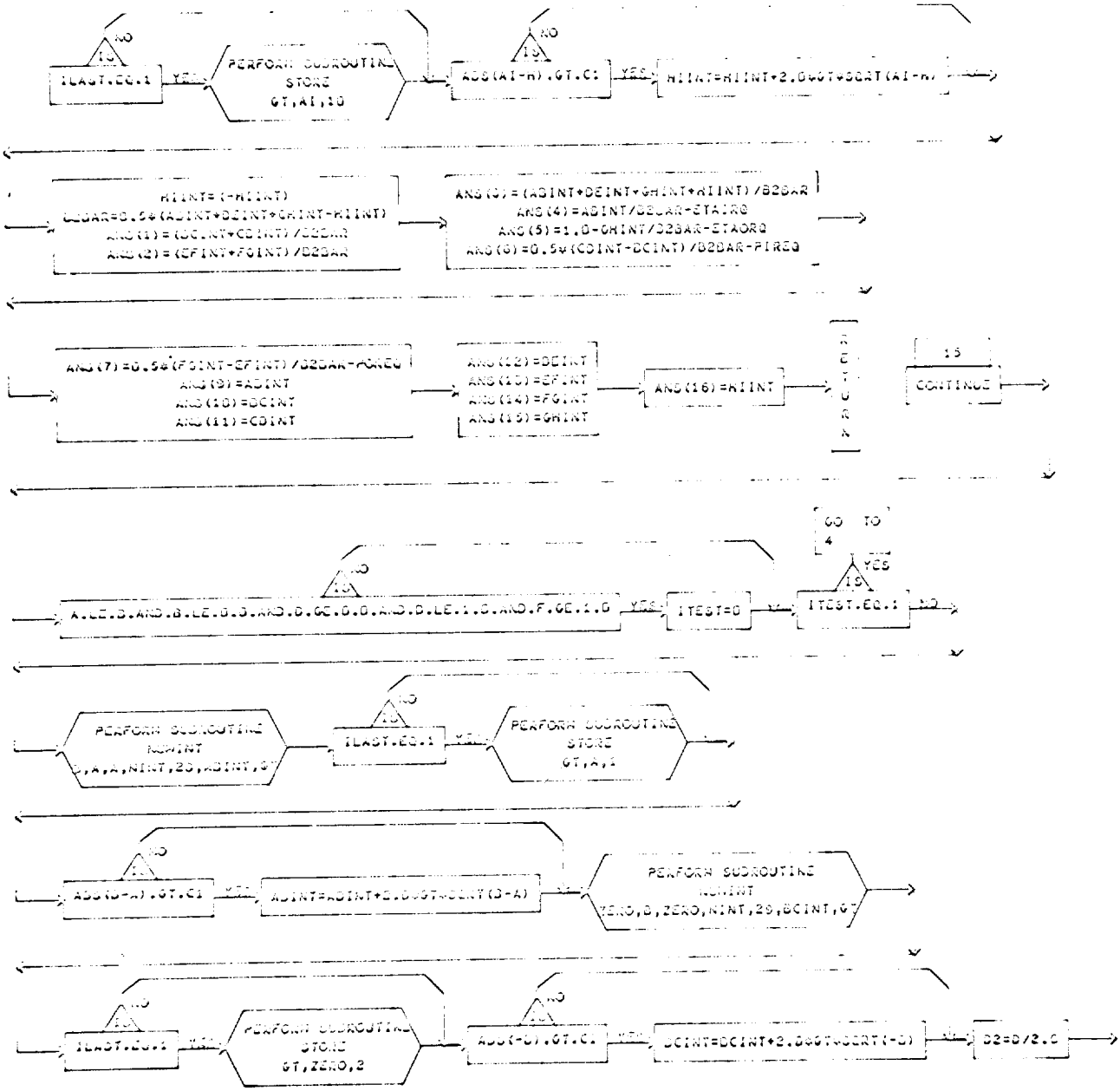
SUBROUTINE GEOM (A,B)

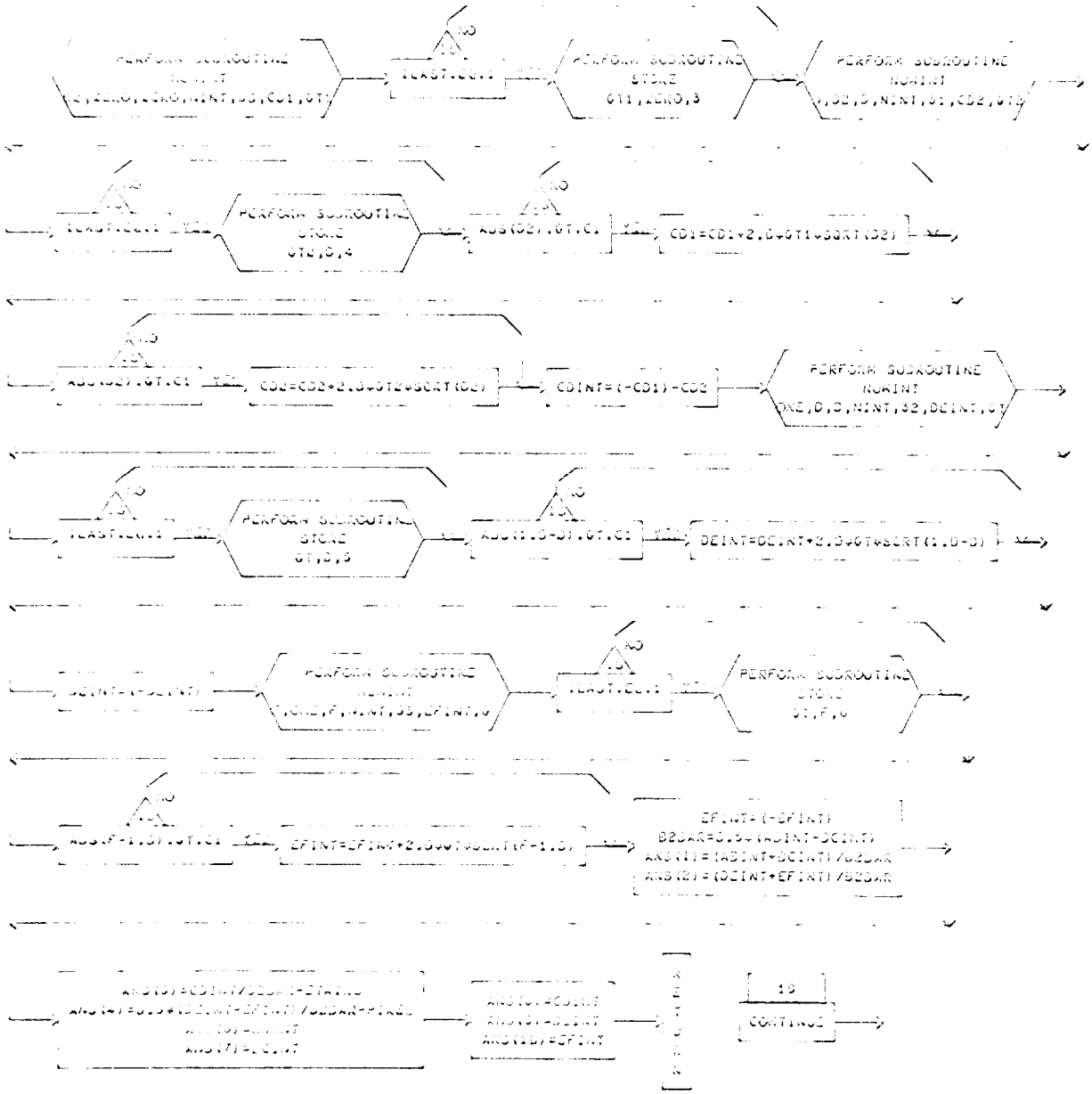


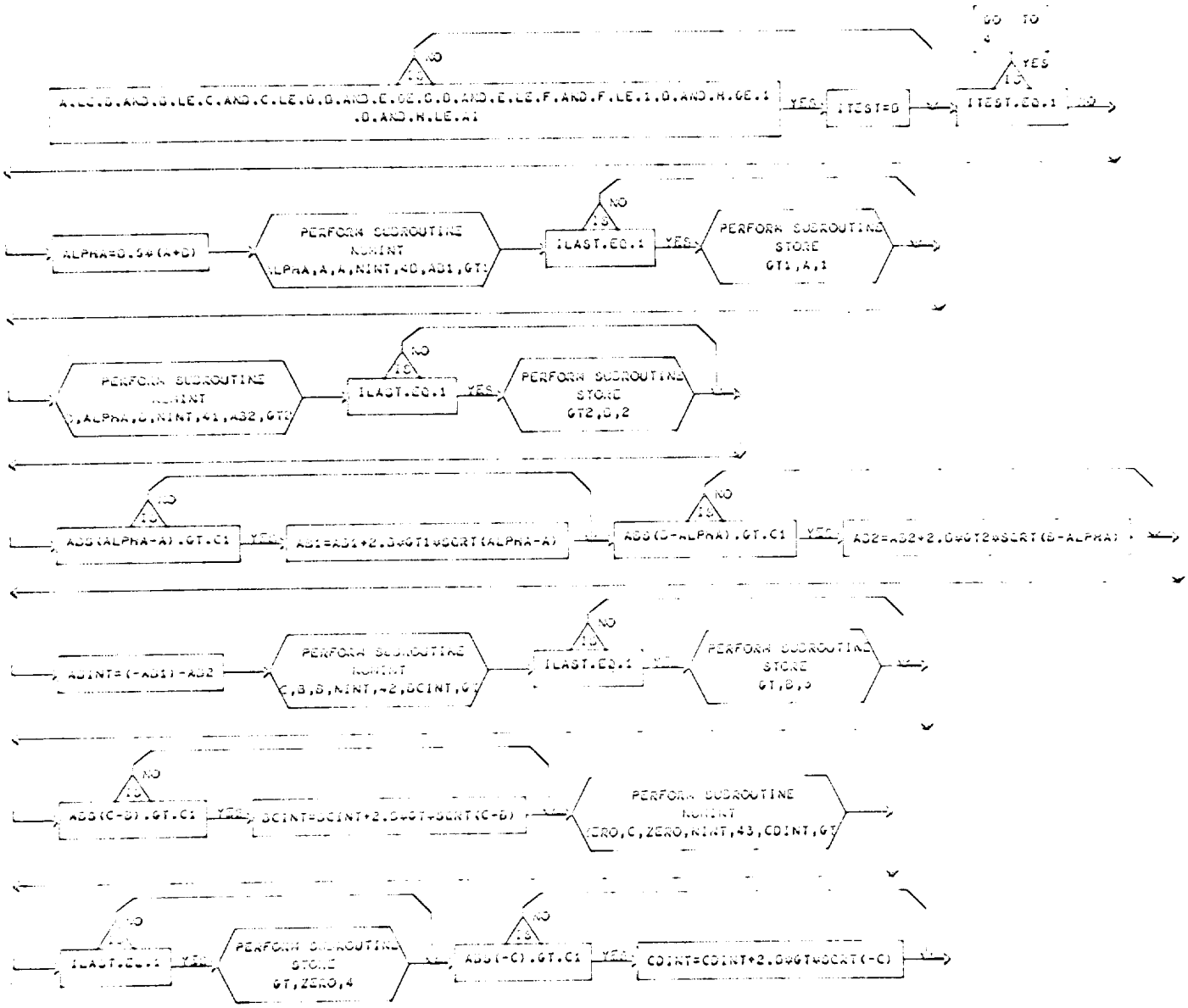

PERFORM SUBROUTINE
STORE
PERFORM SUBROUTINE
STORE
PERFORM SUBROUTINE
STORE
PERFORM SUBROUTINE
STORE
PERFORM SUBROUTINE
STORE
NO
YES

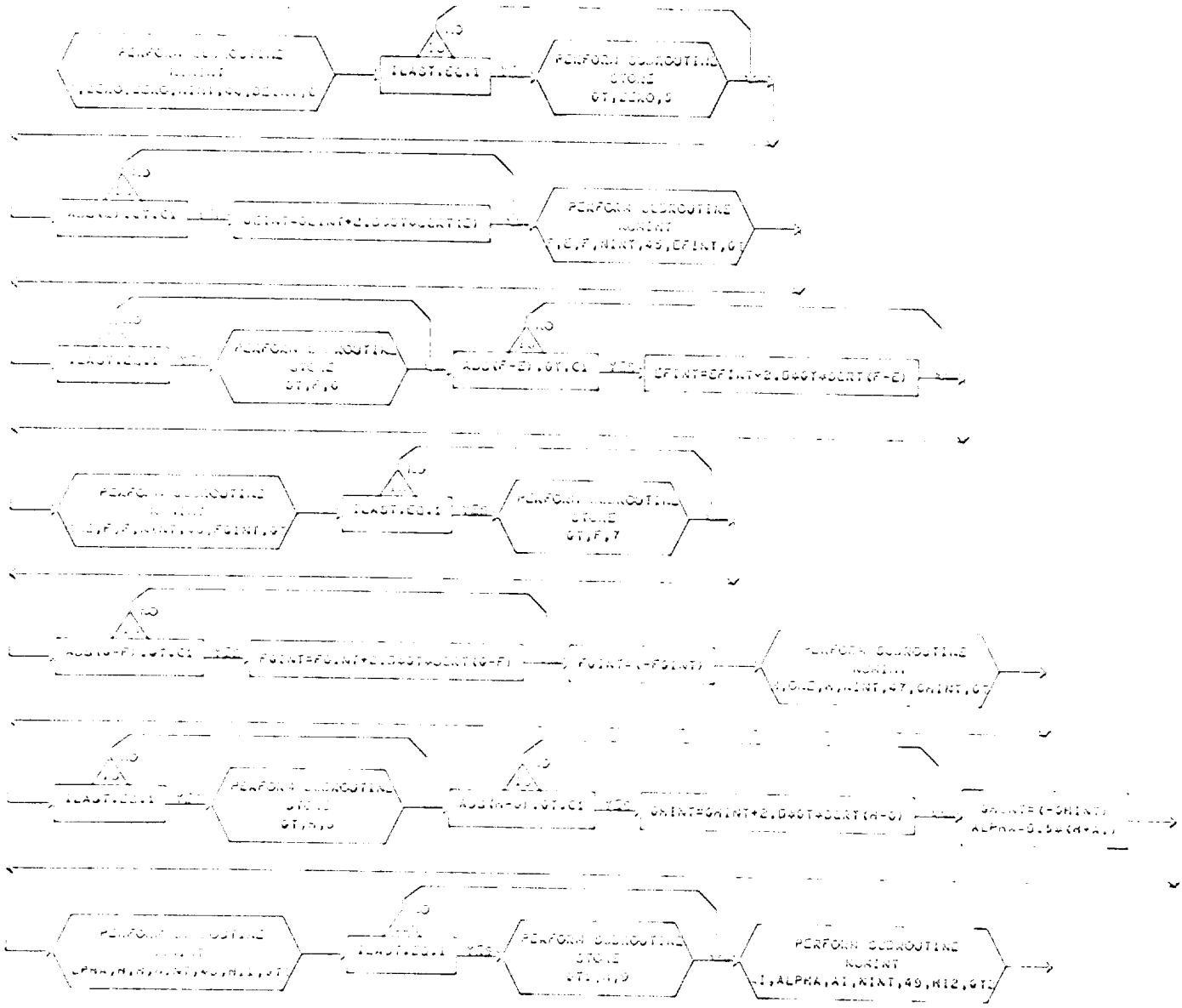

11
CONTINUE
A.LE.0.AND.B.LE.0.AND.C.LE.0.0.AND.D.GE.0.0.AND.F.GE.E.AND.F.LE.1.0.AND.A.GE.1
.0.AND.A1.GE.H
YES
ITEST=0
NO
NO TO
4
YES
ITEST.EQ.1
ALPHA=0.5*(A+D)
PERFORM SUBROUTINE
NUMINT
ALPHA,A,A,NINT,15,AB1,GT
ILAST.EQ.1
PERFORM SUBROUTINE
STORE
GT1,A,1
PERFORM SUBROUTINE
NUMINT
,ALPHA,B,NINT,16,AB2,GT
ILAST.EQ.1
PERFORM SUBROUTINE
STORE
GT2,B,2
ABS(ALPHA-A).GT.C1
AB1=AB1+2.0*GT1*SQRT(ALPHA-A)
ABS(B-ALPHA).GT.C1
AB2=AB2+2.0*GT2*SQRT(B-ALPHA)
ABINT=(-AB1)-AB2
PERFORM SUBROUTINE
NUMINT
E,D,D,NINT,17,BCINT,G
ILAST.EQ.1
PERFORM SUBROUTINE
STORE
GT,D,3
ABS(C-B).GT.C1
BCINT=BCINT+2.0*GT*SQRT(C-B)
PERFORM SUBROUTINE
NUMINT
ZERO,C,ZERO,NINT,18,CDINT,G
ILAST.EQ.1
PERFORM SUBROUTINE
STORE
GT,ZERO,4
ABS(-C).GT.C1
CDINT=CDINT+2.0*GT*SQR
-C)
C2=E/2.0

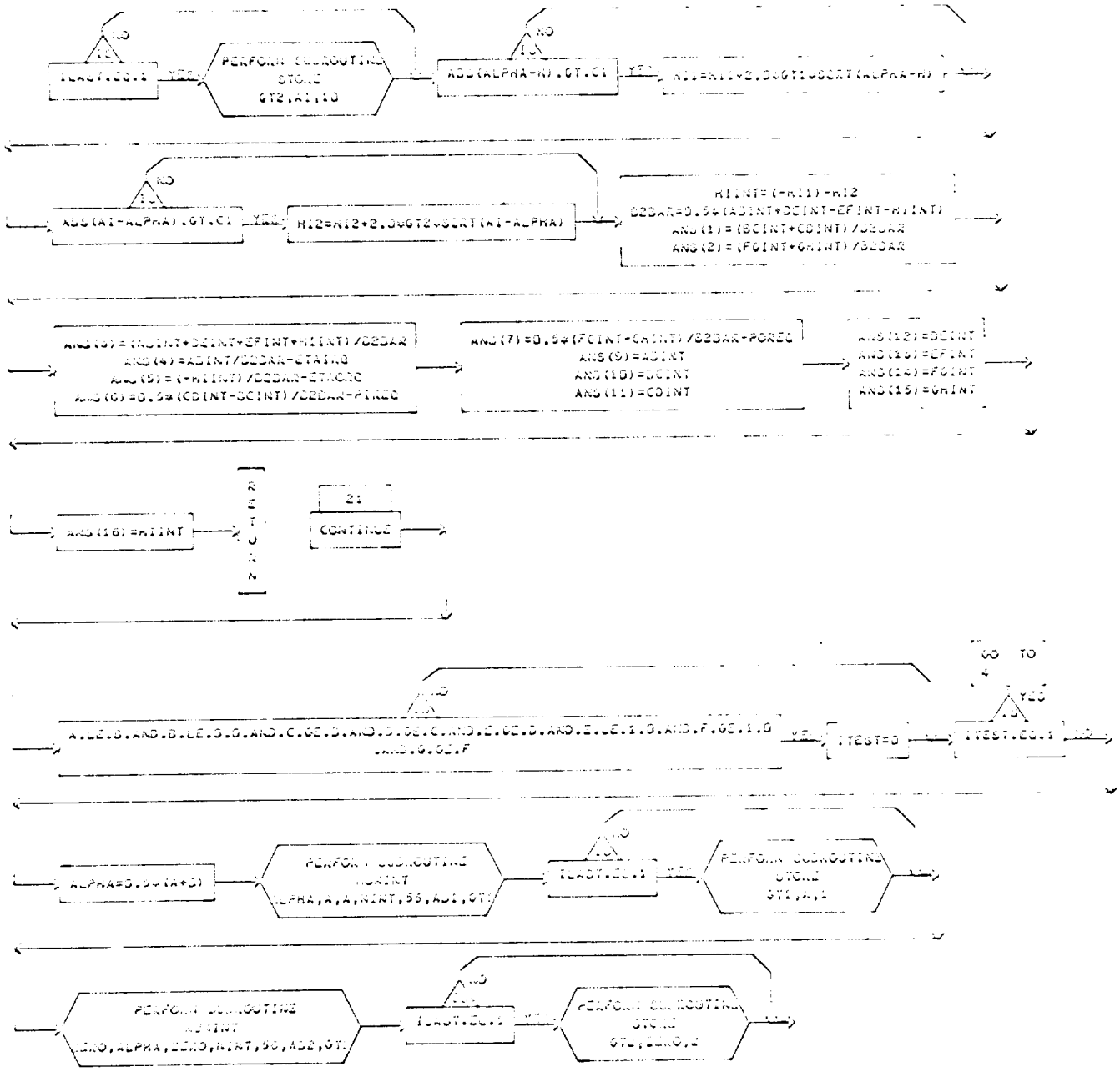


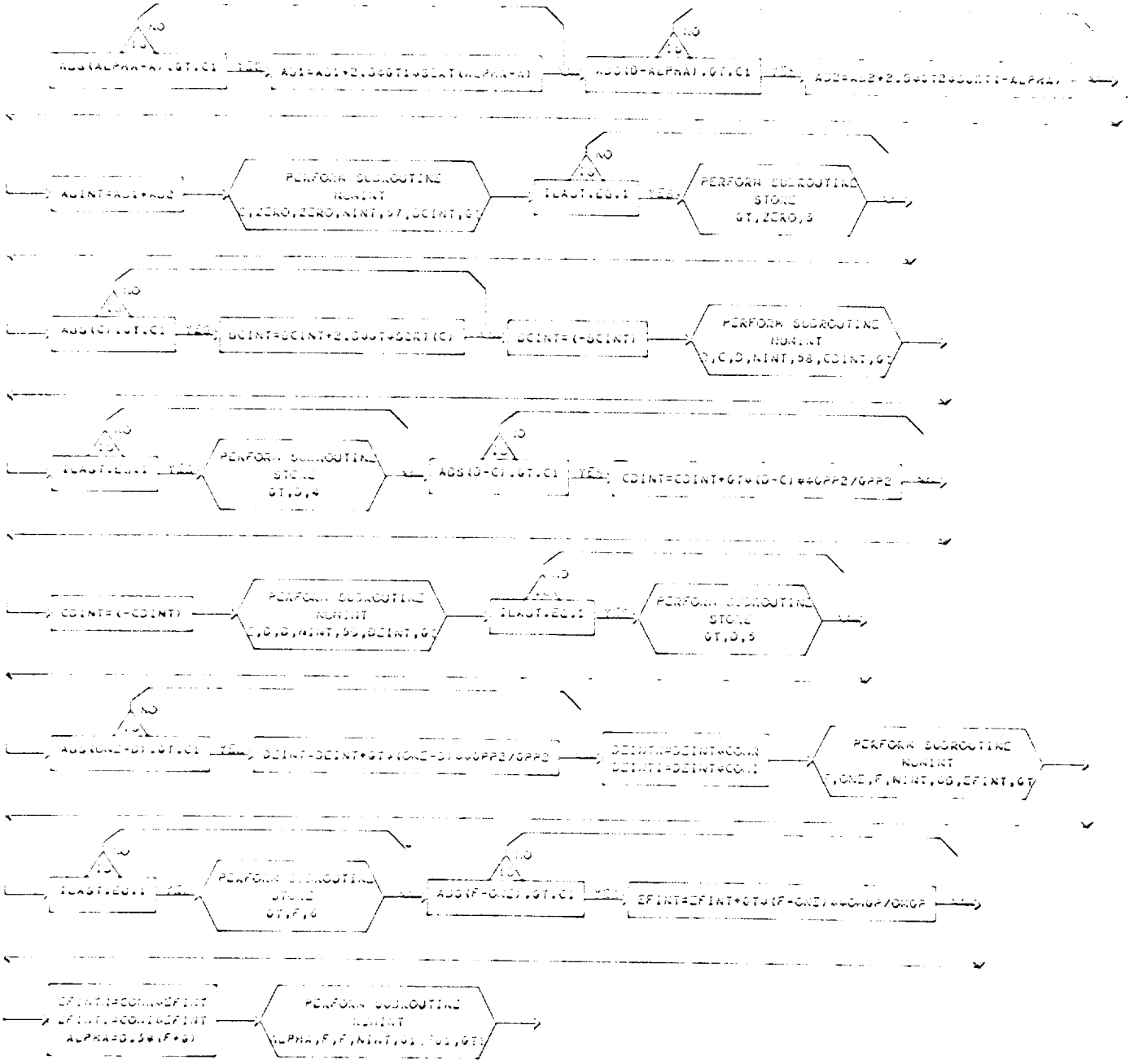

ILAST.EQ.1
PERFORM SUBROUTINE STORE GT,AI,10
ABS(AI-H).GT.C1
HIINT=HIINT+2.0*GT*SQRT(AI-H)
HIINT=(-HIINT)
B2BAR=0.5*(ABINT+DEINT+GHINT-HIINT)
ANS(1)=(BCINT+CDINT)/B2BAR
ANS(2)=(EFINT+FGINT)/B2BAR
ANS(3)=(ABINT+DEINT+GHINT+HIINT)/B2BAR
ANS(4)=ABINT/B2BAR-ETAIRQ
ANS(5)=1.0-GHINT/B2BAR-ETAORQ
ANS(6)=0.5*(CDINT-BCINT)/B2BAR-PIREQ
ANS(7)=0.5*(FGINT-EFINT)/B2BAR-POREQ
ANS(9)=ABINT
ANS(10)=BCINT
ANS(11)=CDINT
ANS(12)=DEINT
ANS(13)=EFINT
ANS(14)=FGINT
ANS(15)=GHINT
ANS(16)=HIINT
RETURN
15
CONTINUE
GO TO 4
A.LE.B.AND.B.LE.0.0.AND.D.GE.0.0.AND.D.LE.1.0.AND.F.GE.1.0
ITEST=0
ITEST.EQ.1
PERFORM SUBROUTINE NOWINT
ILAST.EQ.1
PERFORM SUBROUTINE STORE GT,A,1
ABS(B-A).GT.C1
ABINT=ABINT+2.0*GT*SQRT(B-A)
PERFORM SUBROUTINE NOWINT
ZERO,B,ZERO,NINT,29,BCINT,GT
ILAST.EQ.1
PERFORM SUBROUTINE STORE GT,ZERO,2
ABS(-B).GT.C1
BCINT=BCINT+2.0*GT*SQRT(-B)
B2=B/2.0

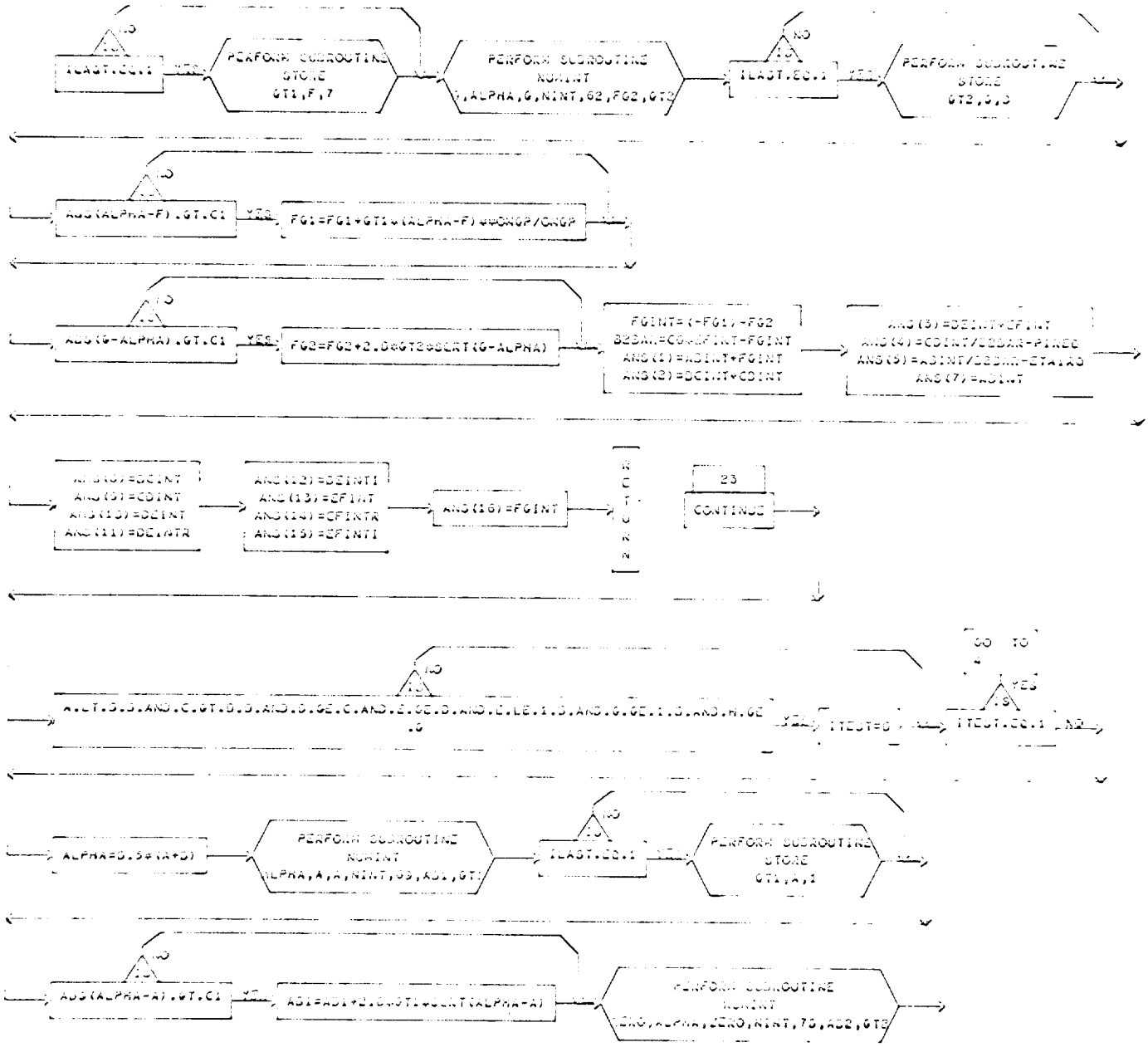

PERFORM SUBROUTINE
NUMINT
PERFORM SUBROUTINE
STORE
GT1,ZERO,3
PERFORM SUBROUTINE
NUMINT
PERFORM SUBROUTINE
STORE
GT2,D,4
CD1=CD1+2.D*GT1*SQRT(D2)
CD2=CD2+2.D*GT2*SQRT(D2)
CDINT=(-CD1)-CD2
PERFORM SUBROUTINE
NUMINT
PERFORM SUBROUTINE
STORE
DEINT=DEINT+2.D*GT*SQRT(1.D-D)
PERFORM SUBROUTINE
PERFORM SUBROUTINE
EFINT=EFINT+2.D*GT*SQRT(F-1.D)
EFINT=(-EFINT)
ANS(1)=(ABINT+DEINT)/B2BAR
ANS(2)=(DEINT+EFINT)/B2BAR
ANS(3)=CDINT/B2BAR-ETAIRS
ANS(4)=DEINT/B2BAR-PIRES
ANS(5)=CDINT
ANS(6)=DEINT
ANS(7)=EFINT
RETURN
19
CONTINUE

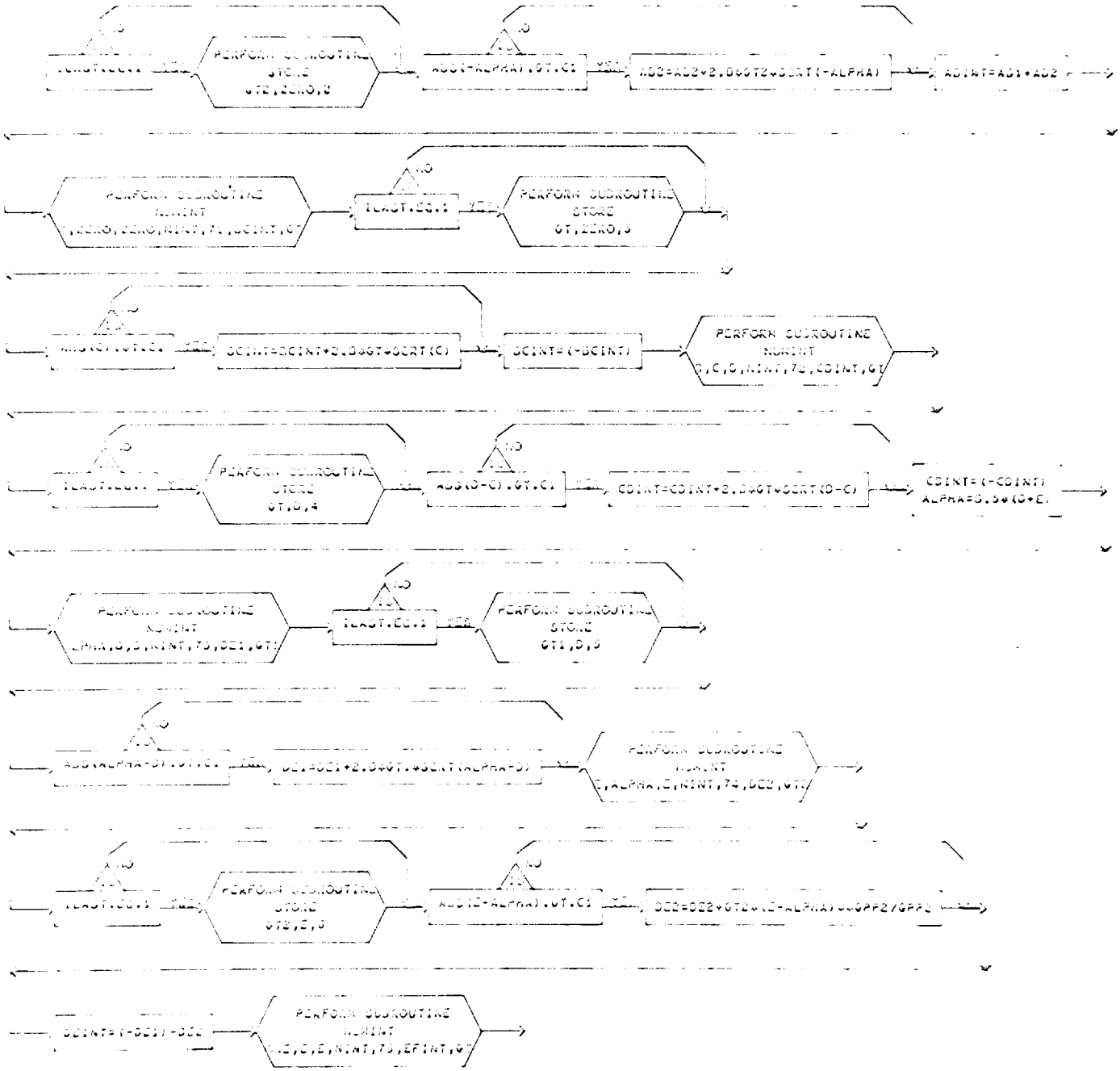

GO TO
4
YES
NO
IS
A.LE.B.AND.B.LE.C.AND.C.LE.D.B.AND.E.GE.C.B.AND.E.LE.F.AND.F.LE.1.B.AND.H.GE.1
.B.AND.H.LE.A1
YES
ITEST=B
ITEST.EQ.1
ALPHA=0.5*(A+B)
PERFORM SUBROUTINE
NUMINT
ALPHA,A,A,NINT,40,AB1,GT1
ILAST.EQ.1
PERFORM SUBROUTINE
STORE
GT1,A,1
PERFORM SUBROUTINE
NUMINT
B,ALPHA,B,NINT,41,AB2,GT2
ILAST.EQ.1
PERFORM SUBROUTINE
STORE
GT2,B,2
ABS(ALPHA-A).GT.C1
AB1=AB1+2.B*GT1*SQRT(ALPHA-A)
ABS(B-ALPHA).GT.C1
AB2=AB2+2.B*GT2*SQRT(B-ALPHA)
ABINT=(-AB1)-AB2
PERFORM SUBROUTINE
NUMINT
C,B,B,NINT,42,BCINT,GT
ILAST.EQ.1
PERFORM SUBROUTINE
STORE
GT,B,3
ABS(C-B).GT.C1
BCINT=BCINT+2.B*GT*SQRT(C-B)
PERFORM SUBROUTINE
NUMINT
ZERO,C,ZERO,NINT,43,CDINT,GT
ILAST.EQ.1
PERFORM SUBROUTINE
STORE
GT,ZERO,4
ABS(-C).GT.C1
CDINT=CDINT+2.B*GT*SQRT(-C)

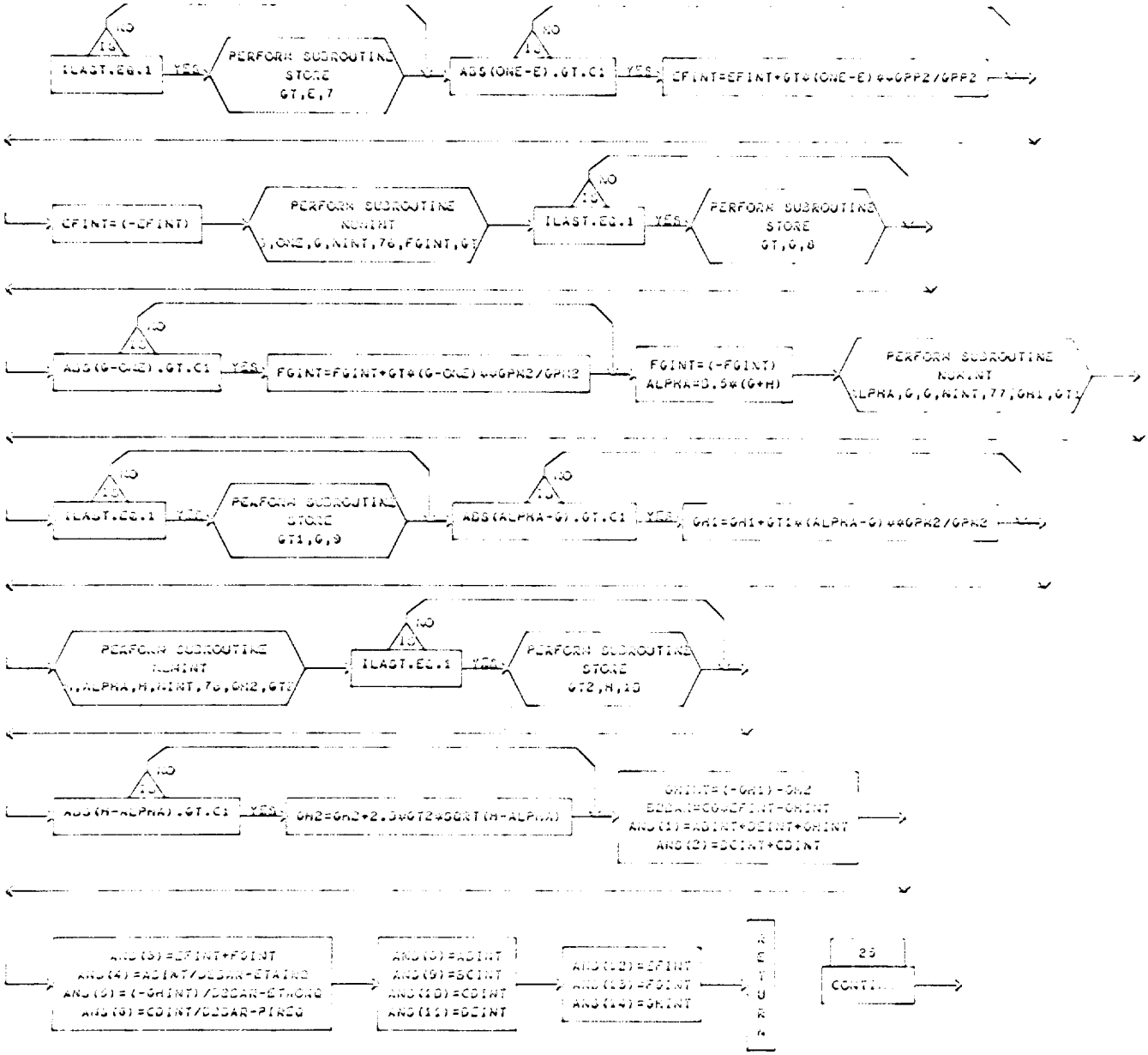


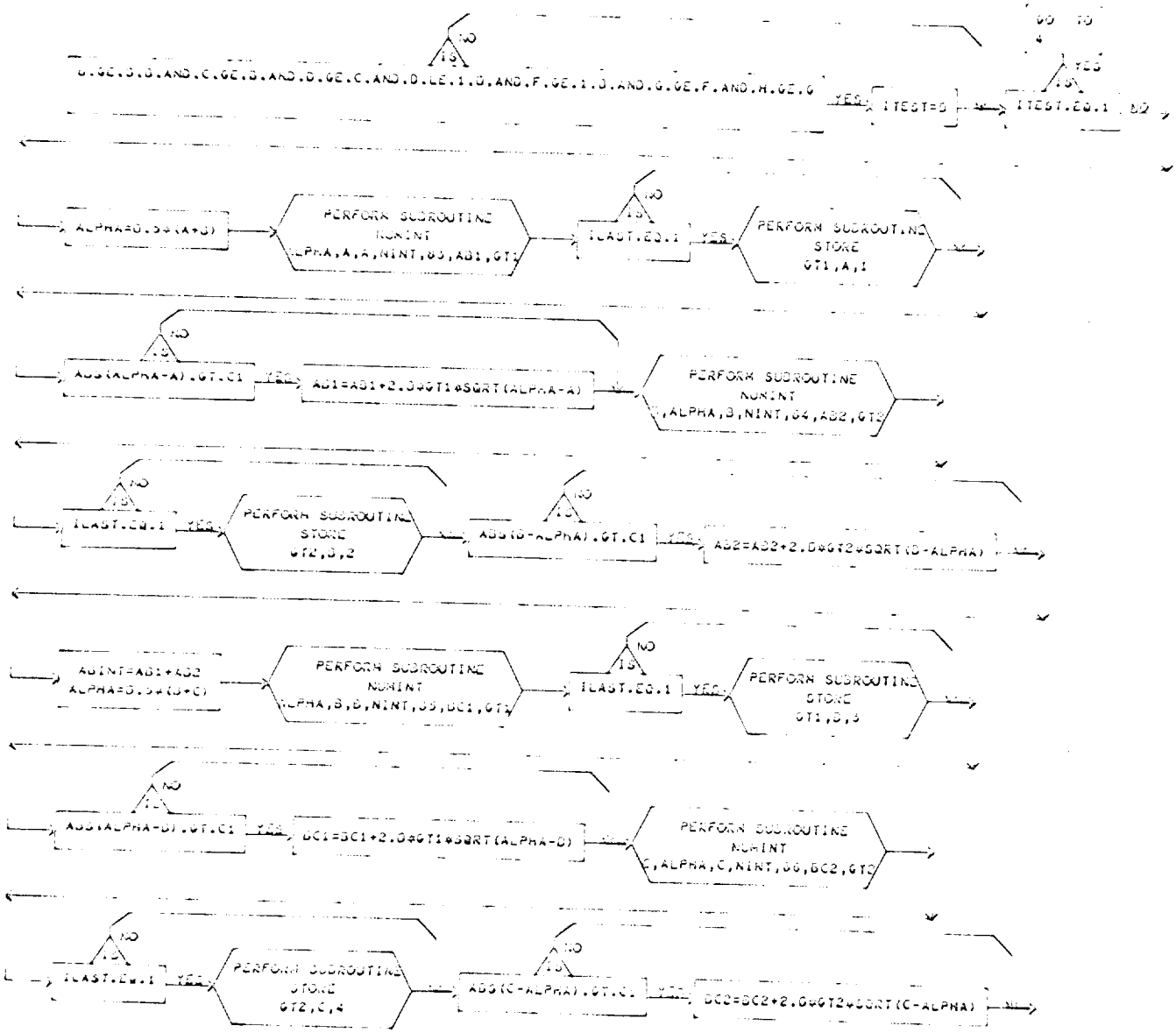

ILAST.EQ.1
PERFORM SUBROUTINE STORE GT2,AI,10
ABS(ALPHA-H).GT.C1
HI1=HI1+2.0*GT1*SQRT(ALPHA-H)
ABS(AI-ALPHA).GT.C1
HI2=HI2+2.0*GT2*SQRT(AI-ALPHA)
HIINT=(-HI1)-HI2
B2BAR=0.5*(ABINT+DEINT-EFINT-HIINT)
ANS(1)=(BCINT+CDINT)/B2BAR
ANS(2)=(FGINT+GHINT)/B2BAR
ANS(3)=(ABINT+DEINT+EFINT+HIINT)/B2BAR
ANS(4)=ABINT/B2BAR-ETAIRQ
ANS(5)=(-HIINT)/B2BAR-ETAIRQ
ANS(6)=0.5*(CDINT-BCINT)/B2BAR-PIREQ
ANS(7)=0.5*(FGINT-GHINT)/B2BAR-POREQ
ANS(9)=ABINT
ANS(10)=BCINT
ANS(11)=CDINT
ANS(12)=DEINT
ANS(13)=EFINT
ANS(14)=FGINT
ANS(15)=GHINT
ANS(16)=HIINT
RETURN
21
CONTINUE
A.LE.B.AND.B.LE.B.0.AND.C.GE.B.AND.D.GE.C.AND.E.GE.D.AND.E.LE.1.0.AND.F.GE.1.0 .AND.G.GE.F
ITEST=0
ITEST.EQ.1
GO TO 4
YES
ALPHA=0.5*(A+B)
PERFORM SUBROUTINE NUMINT ALPHA,A,A,NINT,55,AB1,GT1
ILAST.EQ.1
PERFORM SUBROUTINE STORE GT1,A,1
PERFORM SUBROUTINE NUMINT ZERO,ALPHA,ZERO,NINT,56,AB2,GT2
ILAST.EQ.1
PERFORM SUBROUTINE STORE GT2,ZERO,2

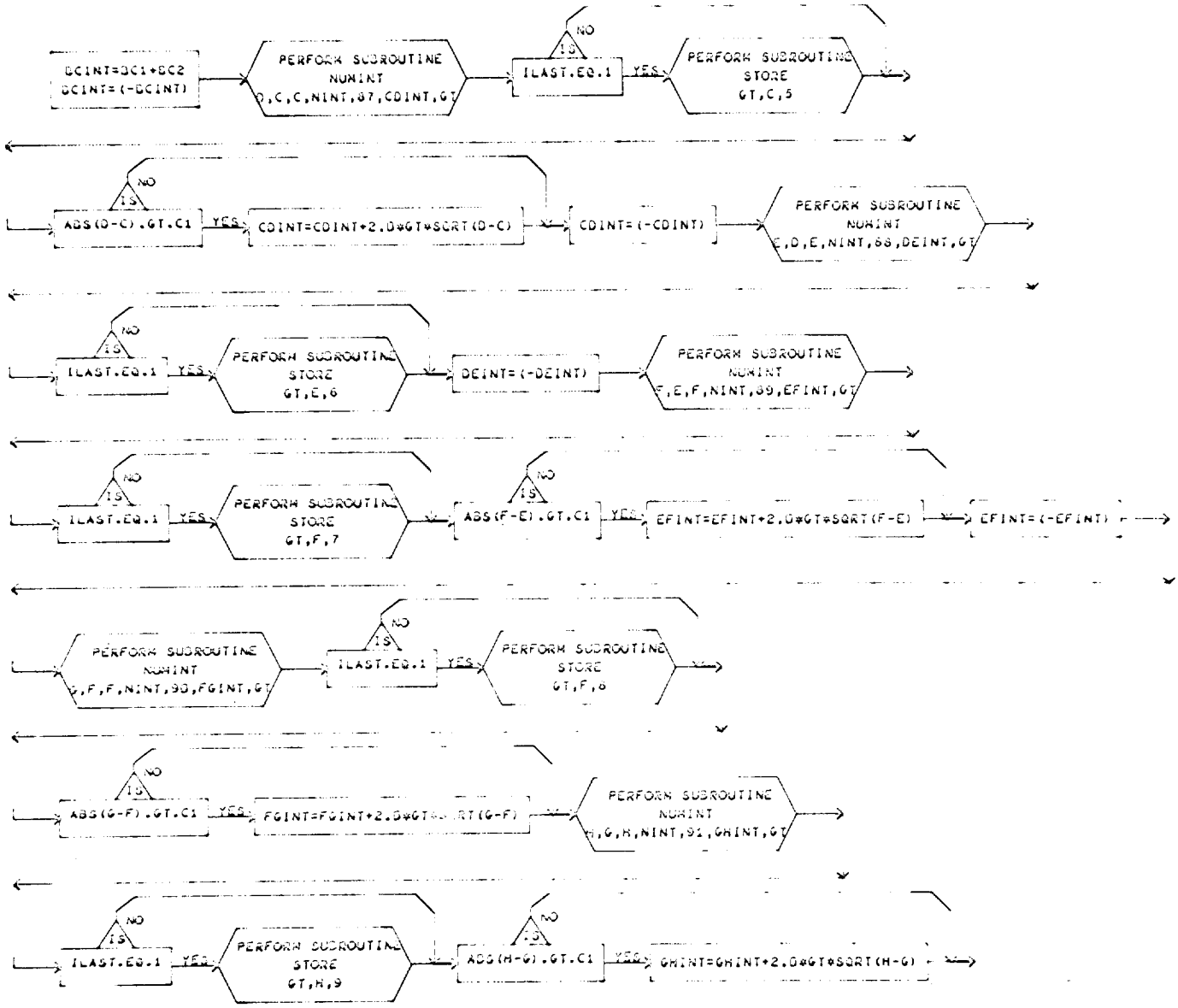

PERFORM SUBROUTINE
NUMINT
PERFORM SUBROUTINE
STORE
GT,ZERO,3
PERFORM SUBROUTINE
NUMINT
PERFORM SUBROUTINE
STORE
GT,D,4
PERFORM SUBROUTINE
STORE
GT,D,5
PERFORM SUBROUTINE
NUMINT
PERFORM SUBROUTINE
NUMINT
PERFORM SUBROUTINE
NUMINT
PERFORM SUBROUTINE
NUMINT

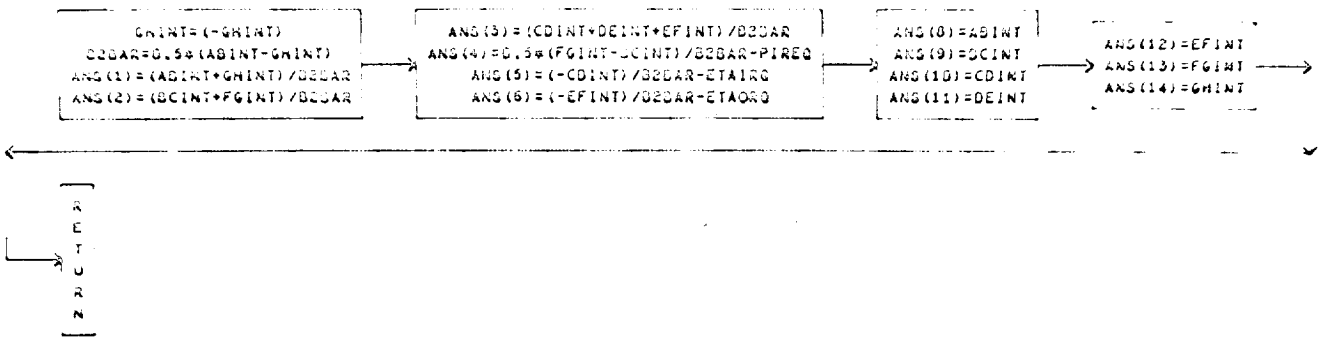

PERFORM SUBROUTINE STORE GT1,F,7
PERFORM SUBROUTINE NUMINT
ILAST.EQ.1
YES
NO
ABS(ALPHA-F).GT.C1
FG1=FG1+GT1*(ALPHA-F)**ONGP/ONGP
ABS(G-ALPHA).GT.C1
FG2=FG2+2.D*GT2*SQRT(G-ALPHA)
FGINT=(-FG1)-FG2
ANS(1)=ABINT+FGINT
ANS(2)=DCINT+CDINT
ANS(16)=FGINT
RETURN
23
CONTINUE
ITEST=0
ITEST.EQ.1
GO TO 4
ALPHA=0.5*(A+B)
PERFORM SUBROUTINE NUMINT
ILAST.EQ.1
PERFORM SUBROUTINE STORE GT1,A,1
ABS(ALPHA-A).GT.C1
PERFORM SUBROUTINE NUMINT


PERFORM SUBROUTINE
STORE
GT2,ZERO,2
ABS(-ALPHA).GT.C1
AB2=AB2+2.D*GT2*SQRT(-ALPHA)
ABINT=AB1+AB2
PERFORM SUBROUTINE
NUMINT
ILAST.EQ.1
PERFORM SUBROUTINE
STORE
GT,ZERO,3
BCINT=BCINT+2.D*GT*SQRT(C)
BCINT=(-BCINT)
PERFORM SUBROUTINE
NUMINT
PERFORM SUBROUTINE
STORE
GT,D,4
ABS(D-C).GT.C1
CDINT=CDINT+2.D*GT*SQRT(D-C)
CDINT=(-CDINT)
ALPHA=0.5*(D+E)
PERFORM SUBROUTINE
NUMINT
ILAST.EQ.1
PERFORM SUBROUTINE
STORE
GT1,D,5
PERFORM SUBROUTINE
NUMINT
PERFORM SUBROUTINE
STORE
GT2,E,6
ABS(E-ALPHA).GT.C1
DE2=DE2+GT2*(E-ALPHA)**GPP2/GPP2
DEINT=(-DE1)-DE2
PERFORM SUBROUTINE
NUMINT


ILAST.EQ.1
YES
NO
PERFORM SUBROUTINE STORE GT,E,7
ABS(ONE-E).GT.C1
EFINT=EFINT+GT*(ONE-E)**GPP2/GPP2
EFINT=(-EFINT)
PERFORM SUBROUTINE NUMINT
ILAST.EQ.1
PERFORM SUBROUTINE STORE GT,G,8
ABS(G-ONE).GT.C1
FGINT=FGINT+GT*(G-ONE)**GPN2/GPN2
FGINT=(-FGINT)
ALPHA=0.5*(G+H)
PERFORM SUBROUTINE NUMINT
ILAST.EQ.1
PERFORM SUBROUTINE STORE GT1,G,9
ABS(ALPHA-G).GT.C1
GH1=GH1+GT1*(ALPHA-G)**GPN2/GPN2
PERFORM SUBROUTINE NUMINT
ILAST.EQ.1
PERFORM SUBROUTINE STORE GT2,H,10
ABS(H-ALPHA).GT.C1
GH2=GH2+2.0*GT2*SQRT(H-ALPHA)
GHINT=(-GH1)-GH2
ANS(1)=ADINT+DEINT+GHINT
ANS(2)=BCINT+CDINT
ANS(3)=EFINT+FGINT
ANS(4)=ADINT/D2BAR-ETAIRQ
ANS(5)=(-GHINT)/D2BAR-ETAORQ
ANS(6)=CDINT/D2BAR-PIREQ
ANS(8)=ADINT
ANS(9)=BCINT
ANS(10)=CDINT
ANS(11)=DEINT
ANS(12)=EFINT
ANS(13)=FGINT
ANS(14)=GHINT
RETURN
25
CONTI...


B.GE.0.0.AND.C.GE.0.AND.D.GE.C.AND.D.LE.1.0.AND.F.GE.1.0.AND.G.GE.F.AND.H.GE.G
YES
ITEST=0
ITEST.EQ.1
ALPHA=0.5*(A+B)
PERFORM SUBROUTINE NONINT
ILAST.EQ.1
PERFORM SUBROUTINE STORE GT1,A,1
ABS(ALPHA-A).GT.C1
AB1=AB1+2.0*GT1*SQRT(ALPHA-A)
PERFORM SUBROUTINE NONINT
ILAST.EQ.1
PERFORM SUBROUTINE STORE GT2,B,2
ABS(B-ALPHA).GT.C1
AB2=AB2+2.0*GT2*SQRT(B-ALPHA)
ABINT=AB1+AB2
ALPHA=0.5*(B+C)
PERFORM SUBROUTINE NONINT
ILAST.EQ.1
PERFORM SUBROUTINE STORE GT1,B,3
ABS(ALPHA-B).GT.C1
BC1=BC1+2.0*GT1*SQRT(ALPHA-B)
PERFORM SUBROUTINE NONINT
ILAST.EQ.1
PERFORM SUBROUTINE STORE GT2,C,4
ABS(C-ALPHA).GT.C1
BC2=BC2+2.0*GT2*SQRT(C-ALPHA)


DCINT=DC1+DC2
DCINT=(-DCINT)
PERFORM SUBROUTINE
NUMINT
D,C,C,NINT,87,CDINT,GT
ILAST.EQ.1
IS
NO
YES
PERFORM SUBROUTINE
STORE
GT,C,5
ABS(D-C).GT.C1
CDINT=CDINT+2.D*GT*SQRT(D-C)
CDINT=(-CDINT)
PERFORM SUBROUTINE
NUMINT
E,D,E,NINT,88,DEINT,GT
PERFORM SUBROUTINE
STORE
GT,E,6
DEINT=(-DEINT)
PERFORM SUBROUTINE
NUMINT
F,E,F,NINT,89,EFINT,GT
PERFORM SUBROUTINE
STORE
GT,F,7
ABS(F-E).GT.C1
EFINT=EFINT+2.D*GT*SQRT(F-E)
EFINT=(-EFINT)
PERFORM SUBROUTINE
NUMINT
G,F,F,NINT,90,FGINT,GT
PERFORM SUBROUTINE
STORE
GT,F,8
ABS(G-F).GT.C1
PERFORM SUBROUTINE
NUMINT
H,G,H,NINT,91,GHINT,GT
PERFORM SUBROUTINE
STORE
GT,H,9
ABS(H-G).GT.C1
GHINT=GHINT+2.D*GT*SQRT(H-G)


GHINT=(-GHINT)
B2DAR=0.5*(ABINT-GHINT)
ANS(1)=(ABINT+GHINT)/B2DAR
ANS(2)=(BCINT+FGINT)/B2DAR
ANS(3)=(CDINT+DEINT+EFINT)/B2DAR
ANS(4)=0.5*(FGINT-BCINT)/B2DAR-PIREQ
ANS(5)=(-CDINT)/B2DAR-ETA1RQ
ANS(6)=(-EFINT)/B2DAR-ETA0RQ
ANS(8)=ABINT
ANS(9)=BCINT
ANS(10)=CDINT
ANS(11)=DEINT
ANS(12)=EFINT
ANS(13)=FGINT
ANS(14)=GHINT
RETURN

SYMBOLS USED IN SUBROUTINE GEOM (DECK IDENTIFICATION 55VM)

A	R	C	POTENTIAL AT A (MAPPING CONSTANT)	GEOM
AA	R	C	ARRAY FOR RESULTS OF GEOMETRY AND DRAG INTEGRALS	GEOM
ABINT	R	U	INTEGRAL FROM A TO B	GEOM
ABSC	R	C	ABSCISSA ARRAY FOR NUMERICAL INTEGRALS	GEOM
AB1	R	U	FIRST HALF OF INTEGRAL FROM A TO B	GEOM
AB2	R	U	SECOND HALF OF INTEGRAL FROM A TO B	GEOM
AI	R	C	POTENTIAL AT I (MAPPING CONSTANT)	GEOM
ALOAD	R	C	STORAGE ARRAY FOR POTENTIAL IN THE REAL (PHYSICAL) PLANE	GEOM
ALPHA	R	U	TEMPORARY STORAGE VARIABLE	GEOM
ANS	R	D	ARRAY FOR ANSWERS FROM NUMERICAL INTEGRALS	GEOM
B	R	C	POTENTIAL AT B (MAPPING CONSTANT)	GEOM
BCINT	R	U	INTEGRAL FROM B TO C	GEOM
BC1	R	U	FIRST HALF OF INTEGRAL FROM B TO C	GEOM
BC2	R	U	SECOND HALF OF INTEGRAL FROM B TO C	GEOM
B2BAR	R	U	AVERAGE SEMISPAN	GEOM
C	R	C	POTENTIAL AT C (MAPPING CONSTANT)	GEOM
CDINT	R	U	INTEGRAL FROM C TO D	GEOM
CD1	R	U	FIRST HALF OF INTEGRAL FROM C TO D	GEOM
CD2	R	U	SECOND HALF OF INTEGRAL FROM C TO D	GEOM
CG	R	C	COSINE OF DIHEDRAL ANGLE	GEOM
CONI	R	C	FACTOR FOR IMAJINARY PART OF COMPLEX DISTANCE	GEOM
CONR	R	C	FACTOR FOR REAL PART OF COMPLEX DISTANCE	GEOM
C1	R	C	CONSTANT THAT IS COMPUTER HARDWARE DEPENDENT	GEOM
C2	R	C	CONSTANT THAT IS COMPUTER HARDWARE DEPENDENT	GEOM
C3	R	C	CONSTANT THAT IS COMPUTER HARDWARE DEPENDENT	GEOM
D	R	C	POTENTIAL AT D (MAPPING CONSTANT)	GEOM
DATA	R	C	ARRAY FOR INPUT DATA	GEOM
DEINT	R	U	INTEGRAL FROM D TO E	GEOM
DEINTR	R	U	REAL PART OF COMPLEX DISTANCE FROM D TO E	GEOM
DEINTI	R	U	IMAJINARY PART OF COMPLEX DISTANCE FROM D TO E	GEOM
DE1	R	U	FIRST HALF OF INTEGRAL FROM D TO E	GEOM
DE2	R	U	SECOND HALF OF INTEGRAL FROM D TO E	GEOM
D2	R	U	INTERMEDIATE CALCULATION VARIABLE	GEOM
E	R	C	POTENTIAL AT E (MAPPING CONSTANT)	GEOM
EFINT	R	U	INTEGRAL FROM E TO F	GEOM
EFINTR	R	U	REAL PART OF COMPLEX DISTANCE FROM E TO F	GEOM
EFINTI	R	U	IMAJINARY PART OF COMPLEX DISTANCE FROM E TO F	GEOM
ETAIRO	R	C	INBOARD SPANWISE LOCATION (NONDIMENSIONAL)	GEOM

SYMBOLS USED IN SUBROUTINE GEOM (DECK IDENTIFICATION 55V11)

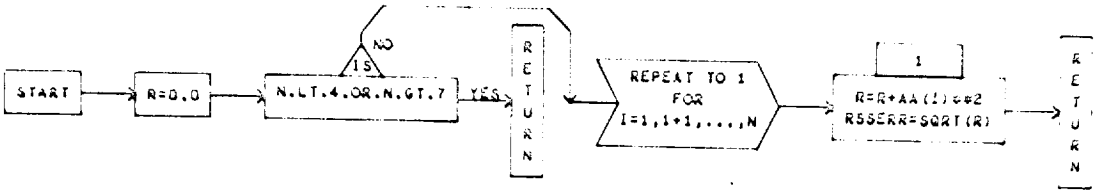
ETAOBC	R	C	OUTBOARD SPANWISE LOCATION (NONDIMENSIONAL)	GEOM
EZ	R	U	INTERMEDIATE CALCULATION VARIABLE	GEOM
F	R	C	POTENTIAL AT F (MAPPING CONSTANT)	GEOM
FGINT	R	U	INTEGRAL FROM F TO G	GEOM
FG1	R	U	FIRST HALF OF INTEGRAL FROM F TO G	GEOM
FG2	R	U	SECOND HALF OF INTEGRAL FROM F TO G	GEOM
G	R	C	POTENTIAL AT G (MAPPING CONSTANT)	GEOM
GAMMA	R	C	DIHEDRAL ANGLE IN DEGREES	GEOM
GHINT	R	U	INTEGRAL FROM G TO H	GEOM
GH1	R	U	FIRST HALF OF INTEGRAL FROM G TO H	GEOM
GH2	R	U	SECOND HALF OF INTEGRAL FROM G TO H	GEOM
GP	R	C	DIHEDRAL ANGLE DIVIDED BY PI	GEOM
GPM2	R	C	GP - 0.5	GEOM
GPP2	R	C	GP + 0.5	GEOM
GT	R	U	CONSTANT OF REMOVABLE SINGULARITY	GEOM
GT1	R	U	CONSTANT OF REMOVABLE SINGULARITY - FIRST HALF OF INTEGRAL	GEOM
GT2	R	U	CONSTANT OF REMOVABLE SINGULARITY - SECOND HALF OF INTEGRAL	GEOM
H	R	C	POTENTIAL AT H (MAPPING CONSTANT)	GEOM
HIINT	R	U	INTEGRAL FROM H TO I	GEOM
HI1	R	U	FIRST HALF OF INTEGRAL FROM H TO I	GEOM
HI2	R	U	SECOND HALF OF INTEGRAL FROM H TO I	GEOM
ILAST	I	C	FLAG FOR LAST ITERATION	GEOM
ININT	I	C	NUMBER OF ORDINATES IN NUMERICAL INTEGRALS	GEOM
ITEST	I	U	TEST FLAG FOR MONOTONE INCREASING POTENTIALS (A SMALLEST)	GEOM
MODE	I	C	PROGRAM MODE OF OPERATION	GEOM
NINT	I	U	DUMMY INTEGER VARIABLE	GEOM
NTYPE	I	C	CONFIGURATION TYPE	GEOM
OMGP	R	C	1.0 - GP	GEOM
ONE	R	U	1.0	GEOM
ORD	R	C	ORDINATE ARRAY FOR NUMERICAL INTEGRALS	GEOM
PIREQ	R	C	INBOARD LENGTH OR DISTANCE (NONDIMENSIONAL)	GEOM
POREQ	R	C	OUTBOARD LENGTH OR DISTANCE (NONDIMENSIONAL)	GEOM
R1	R	C	SQUARE ROOT OF SUM OF SQUARES OF ERRORS IN REQD CONDITIONS	GEOM
TAB	R	C	INTERPOLATION ARRAY FOR INDEPENDENT VARIABLE AND FUNCTION	GEOM
TABLE	R	C	INTEGRATED VALUES OF ORDINATES AS A FUNCTION OF ABSCISSA	GEOM
ZERO	R	U	0.0	GEOM

Subroutine RSSERR (N) (Deck 55VN). This subroutine computes the error parameter $\bar{e}$ if the calling parameter N is greater than 3 or less than 8; otherwise, no calculation is made after $\bar{e}$ is set to zero.

```
              FUNCTION RSSERR(N)                                        55VN0010
        C                                                               55VN0020
        C     THIS ARITHMETIC FUNCTION COMPUTES THE SQUARE ROOT OF THE SUM OF  55VN0030
        C     THE SQUARES OF THE ERRORS IN THE REQUIRED GEOMETRY CONDITIONS    55VN0040
        C                                                               55VN0050
000003        COMMON DATA(17), ABSC(50), ORD(50), TABLE(50), ALOAD(1000), AA(16)55VN0060
             A , TAB(101), GP, GPM2, GPP2, CG, OMGP, CONR, CONI, ILAST, C1, C2  55VN0070
             B , C3, R1                                                 55VN0080
        C                                                               55VN0090
000003        EQUIVALENCE              (DATA( 1),      A) , (DATA( 2),      B)  55VN0100
             A , (DATA( 3),      C) , (DATA( 4),      D) , (DATA( 5),      E)  55VN0110
             B , (DATA( 6),      F) , (DATA( 7),      G) , (DATA( 8),      H)  55VN0120
             C , (DATA( 9),     AI) , (DATA(10),   MODE) , (DATA(11),  NTYPE)  55VN0130
             D , (DATA(12),  ININT) , (DATA(13), ETAIRQ) , (DATA(14), ETAORQ)  55VN0140
             E , (DATA(15),  PIREQ) , (DATA(16),  POREQ) , (DATA(17),  GAMMA)  55VN0150
        C                                                               55VN0160
        C     TEST FOR VALID N                                          55VN0170
        C                                                               55VN0180
000003        R = 0.0                                                   55VN0190
000004        IF (N .LT. 4 .OR. N .GT. 7) RETURN                        55VN0200
000014        DO 1 I=1,N                                                55VN0210
000018      1 R = R + AA(I) ** 2                                        55VN0220
000022        RSSERR = SQRT(R)                                          55VN0230
000025        RETURN                                                    55VN0240
000025        END                                                       55VN0250
```

```
SUBPROGRAM LENGTH
000041

FUNCTION ASSIGNMENTS

STATEMENT ASSIGNMENTS
1       -  000016

BLOCK NAMES AND LENGTHS
        -  002420

VARIABLE ASSIGNMENTS
AA      -  002217C01 ABSC    -  000021C01 ALOAD  -  000247C01 DATA   -  000000C01
I       -  000040    ORD     -  000103C01 R      -  000037    RSSERR -  000036
TAB     -  002237C01 TABLE   -  000165C01

START OF CONSTANTS
000027

START OF TEMPORARIES
000030

START OF INDIRECTS
000035

UNUSED COMPILER SPACE
043400
```

FUNCTION RSSERR(N)



SYMBOLS USED IN SUBROUTINE RSSERR (DECK IDENTIFICATION 55VN)

A	R	C	POTENTIAL AT A (MAPPING CONSTANT)	RSSERR
AA	R	C	ARRAY FOR RESULTS OF GEOMETRY AND DRAG INTEGRALS	RSSERR
ABSC	R	C	ABSCISSA ARRAY FOR NUMERICAL INTEGRALS	RSSERR
AI	R	C	POTENTIAL AT I (MAPPING CONSTANT)	RSSERR
ALOAD	R	C	STORAGE ARRAY FOR POTENTIAL IN THE REAL (PHYSICAL) PLANE	RSSERR
B	R	C	POTENTIAL AT B (MAPPING CONSTANT)	RSSERR
C	R	C	POTENTIAL AT C (MAPPING CONSTANT)	RSSERR
CG	R	C	COSINE OF DIHEDRAL ANGLE	RSSERR
CONI	R	C	FACTOR FOR IMAJINARY PART OF COMPLEX DISTANCE	RSSERR
CONR	R	C	FACTOR FOR REAL PART OF COMPLEX DISTANCE	RSSERR
C1	R	C	CONSTANT THAT IS COMPUTER HARDWARE DEPENDENT	RSSERR
C2	R	C	CONSTANT THAT IS COMPUTER HARDWARE DEPENDENT	RSSERR
C3	R	C	CONSTANT THAT IS COMPUTER HARDWARE DEPENDENT	RSSERR
D	R	C	POTENTIAL AT D (MAPPING CONSTANT)	RSSERR
DATA	R	C	ARRAY FOR INPUT DATA	RSSERR
E	R	C	POTENTIAL AT E (MAPPING CONSTANT)	RSSERR
ETAIBQ	R	C	INBOARD SPANWISE LOCATION (NONDIMENSIONAL)	RSSERR
ETAOBQ	R	C	OUTBOARD SPANWISE LOCATION (NONDIMENSIONAL)	RSSERR
F	R	C	POTENTIAL AT F (MAPPING CONSTANT)	RSSERR
G	R	C	POTENTIAL AT G (MAPPING CONSTANT)	RSSERR
GAMMA	R	C	DIHEDRAL ANGLE IN DEGREES	RSSERR
GP	R	C	DIHEDRAL ANGLE DIVIDED BY PI	RSSERR
GPM2	R	C	GP - 0.5	RSSERR
GPP2	R	C	GP + 0.5	RSSERR
H	R	C	POTENTIAL AT H (MAPPING CONSTANT)	RSSERR
ILAST	I	C	FLAG FOR LAST ITERATION	RSSERR
ININT	I	C	NUMBER OF ORDINATES IN NUMERICAL INTEGRALS	RSSERR
MODE	I	C	PROGRAM MODE OF OPERATION	RSSERR
N	I	U	NUMBER OF REQUIRED GEOMETRY CONDITIONS	RSSERR
NTYPE	I	C	CONFIGURATION TYPE	RSSERR
OMGP	R	C	1.0 - GP	RSSERR
ORD	R	C	ORDINATE ARRAY FOR NUMERICAL INTEGRALS	RSSERR
PIREQ	R	C	INBOARD LENGTH OR DISTANCE (NONDIMENSIONAL)	RSSERR
POREQ	R	C	OUTBOARD LENGTH OR DISTANCE (NONDIMENSIONAL)	RSSERR
R	R	U	TEMPORARY CALCULATION VARIABLE	RSSERR
RSSERR	R	C	SQUARE ROOT OF SUM OF SQUARES OF ERRORS IN REQD CONDITIONS	RSSERR
R1	R	C	SQUARE ROOT OF SUM OF SQUARES OF ERRORS IN REQD CONDITIONS	RSSERR
TAB	R	C	INTERPOLATION ARRAY FOR INDEPENDENT VARIABLE AND FUNCTION	RSSERR

SYMBOLS USED IN SUBROUTINE RSSERR (DECK IDENTIFICATION 55VN)

TABLE	R	C	INTEGRATED VALUES OF ORDINATES AS A FUNCTION OF ABSCISSA	RSSERR

Subroutine NUMINT (UP, ALOW, SING, NPOINT, NTYPE, ANSWER, GT) (Deck 55V0). The numerical integrals for geometry are evaluated by this subroutine which sets ANSWER to the result of the quadrature. The integral of interest is identified by the value of the parameter NTYPE. The integration interval ranges from ALOW to UP, and has a singularity SING at either UP or ALOW. The abscissae for the numerical integration are calculated so that they are spaced more closely near the singularity. If the singularity is at the lower end of the integration interval, the abscissa spacing is given by

$$\Delta\zeta_i = \frac{\bar{f}_i \left(\zeta_u - \zeta_\ell + \sum_{j=1}^{i-1} \Delta\zeta_j \right)}{n - i + 1} \tag{10}$$

where ζ_u = UP, ζ_ℓ = ALOW, n = NPOINT - 1, and $\bar{f}$ is given by

$$\bar{f}_{i+1} = 2\,\bar{f}_i \tag{11}$$

subject to the limitation $\bar{f} \leq 1$, with $\bar{f}_1 = 0.01$. An analogous spacing of abscissae is computed by subroutine NUMINT if the singularity is at the upper limit of the interval.

```
                SUBROUTINE NUMINT (UP,ALOW,SING,NPOINT ,NTYPE,ANSWER,GT)          55V00010
          C                                                                       55V00020
          C     THIS SUBROUTINE SETS UP NUMERICAL INTEGRALS FOR GEOMETRY IN THE   55V00030
          C    TREFFTZ PLANE.                                                     55V00040
          C                                                                       55V00050
000010          COMMON DATA(17), ABSC(50), ORD(50), TABLE(50), ALOAD(1000), AA(16)55V00060
                A , TAB(101), GP, GPM2, GPP2, CG, OMGP, CONR, CONI, ILAST, C2, C1 55V00070
                B , C3, R1                                                        55V00080
          C                                                                       55V00090
000010          EQUIVALENCE              (DATA( 1),       A) , (DATA( 2),       B) 55V00100
                A , (DATA( 3),       C) , (DATA( 4),       D) , (DATA( 5),       E) 55V00110
                G , (DATA( 6),       F) , (DATA( 7),       G) , (DATA( 8),       H) 55V00120
                C , (DATA( 9),      AI) , (DATA(10),    MODE) , (DATA(11), NOUMMY)  55V00130
                D , (DATA(12),   ININT) , (DATA(13), ETAIRQ) , (DATA(14), ETAORQ)  55V00140
                E , (DATA(15),   PIREQ) , (DATA(16),  POREQ) , (DATA(17),  GAMMA)  55V00150
          C                                                                       55V00160
          C     ARITHMETIC STATEMENT FUNCTIONS                                    55V00170
          C                                                                       55V00180
          C     HIGHER LEVEL FUNCTIONS FOR THE CASE NTYPE = 5                     55V00190
          C                                                                       55V00200
000010          QC(Z) = (Z-C)*(Z-1.0)/SQRT(G-Z)*(F-Z)**(-GP)                      55V00210
000041          QD(Z) = (Z-C)*(Z-1.0)/SQRT((Z-A)*Z*(G-Z))                         55V00220
000063          QE(Z) = (Z-C)*(Z-1.0)/SQRT((Z-A)*Z)*(Z-D)**GPM2                   55V00230
          C                                                                       55V00240
          C     HIGHER LEVEL FUNCTIONS FOR CASE 6                                 55V00250
          C                                                                       55V00260
000112          QH(Z) = (Z-C)*(Z-1.0)/SQRT((D-Z)*(H-Z))*((E-Z)/(G-Z))**GP         55V00270
000144          QI(Z) = (Z-C)*(Z-1.0)/SQRT((Z-A)*Z*(H-Z))/(G-Z)**GP               55V00280
000175          QJ(Z) = (Z-C)*(Z-1.0)/SQRT((Z-A)*Z*(Z-D))*(Z-F)**GP               55V00290
          C                                                                       55V00300
          C     HIGHER LEVEL FUNCTIONS FOR THE CASE NTYPE = 7                     55V00310
          C                                                                       55V00320
000226          QM(Z) = (Z-D)*(Z-1.0)*SQRT((G-Z)/((F-Z)*(H-Z)))                   55V00330
000251          QN(Z) = (Z-D)*(Z-1.0)/SQRT(Z*(Z-B)*(Z-C))                         55V00340
          C                                                                       55V00350
000273          GAA(Z) = (Z-C)*(Z-1.0)/SQRT((B-Z)*(D-Z)*(F-Z))                    55V00360
000320          GAB(Z) = (Z-C)*(Z-1.0)/SQRT(Z*(D-Z)*(F-Z))                        55V00370
000343          GDD(Z) = (Z-C)*(Z-1.0)/SQRT(Z*(Z-B)*(F-Z))                        55V00380
000365          GFF(Z) = (Z-C)*(Z-1.0)/SQRT(Z*(Z-B)*(Z-D))                        55V00390
000407          G2A(Z) = (Z-C)*(Z-F)*(Z-H)/SQRT ((B-Z)*Z*(Z-F)*(Z-1.0)*(Z-AI))    55V00400
000441          G3A(Z) = (Z-C)*(Z-F)*(Z-H)/SQRT ((Z-A)*Z*(Z-F)*(Z-1.0)*(Z-AI))    55V00410
000473          GD2(Z) = (Z-C)*(Z-F)*(Z-H)/SQRT ((Z-A)*(Z-B)*(E-Z)*(Z-1.0)*(Z-AI))55V00420
000527          GE2(Z) = (Z-C)*(Z-F)*(Z-H)/SQRT ((Z-A)*(Z-B)*Z*(Z-1.0)*(Z-AI))    55V00430
000561          G11(Z) = (Z-C)*(Z-F)*(Z-H)/SQRT ((Z-A)*(Z-B)*Z*(Z-E)*(AI-Z))     55V00440
000613          GI2(Z) = (Z-C)*(Z-F)*(Z-H)/SQRT ((Z-A)*(Z-B)*Z*(Z-E)*(Z-1.0))    55V00450
000665          GAH(Z) = (Z-B)*(Z-1.0)/SQRT (Z*(Z-D)*(F-Z))                       55V00460
000607          GCH(Z) = (Z-B)*(Z-1.0)/SQRT ((Z-A)*(Z-D)*(Z-F))                   55V00470
000713          GDH(Z) = (Z-B)*(Z-1.0)/SQRT ((Z-A)*Z*(F-Z))                       55V00480
000736          GFH(Z) = (Z-B)*(Z-1.0)/SQRT ((Z-A)*Z*(Z-D))                       55V00490
000760          GYA(Z) = (Z-C)*(Z-E)*(Z-1.0)/SQRT ((Z-B)*Z*(F-Z)*(H-Z)*(AI-Z))    55V00500
001013          GYB(Z) = (Z-C)*(Z-E)*(Z-1.0)/SQRT ((A-Z)*Z*(F-Z)*(H-Z)*(AI-Z))    55V00510
001050          GYC(Z) = (Z-C)*(Z-E)*(Z-1.0)/SQRT ((A-Z)*Z*(F-Z)*(H-Z)*(AI-Z))    55V00520
001105          GYD(Z) = (Z-C)*(Z-E)*(Z-1.0)/SQRT ((Z-A)*(Z-B)*(F-Z)*(H-Z)*(AI-Z))55V00530
001141          GYF(Z) = (Z-C)*(Z-E)*(Z-1.0)/SQRT ((Z-A)*(Z-B)*Z*(H-Z)*(AI-Z))    55V00540
001173          GYH(Z) = (Z-C)*(Z-E)*(Z-1.0)/SQRT ((Z-A)*(Z-B)*Z*(Z-F)*(AI-Z))    55V00550
001225          GYI(Z) = (Z-C)*(Z-E)*(Z-1.0)/SQRT ((Z-A)*(Z-B)*Z*(Z-F)*(Z-H))     55V00560
001257          GWA(Z) = QC(Z)/SQRT(-Z)*(D-Z)**GPM2                               55V00570
```

```
001303        GWB(Z) = QC(Z)/SQRT(Z-A)*(D-Z)**GPM2                          55V00580
001330        GWC(Z) = QC(Z)/SQRT((Z-A)*Z)                                  55V00590
001347        GWD(Z) = QD(Z)*(F-Z)**(-GP)                                   55V00600
001363        GWE(Z) = QD(Z)*(Z-D)**GPM2                                    55V00610
001376        GWF(Z) = QE(Z)/SQRT(G-Z)                                      55V00620
001415        GWG(Z) = QE(Z)*(Z-F)**(-GP)                                   55V00630
001431        GUA(Z) = QH(Z)/SQRT(-Z)                                       55V00640
001447        GUB(Z) = QH(Z)/SQRT(Z-A)                                      55V00650
001466        GUC(Z) = QI(Z)/(G-Z)**GP                                      55V00660
001501        GUD(Z) = QI(Z)*(E-Z)**GP                                      55V00670
001514        GUE(Z) = QI(Z)/SQRT(Z-D)                                      55V00680
001533        GUF(Z) = QJ(Z)/SQRT(H-Z)                                      55V00690
001552        GUG(Z) = QJ(Z)/(Z-G)**GP                                      55V00700
001565        GSA(Z) = QM(Z)/SQRT((B-Z)*(C-Z))                              55V00710
001605        GSB(Z) = QM(Z)/SQRT(Z*(C-Z))                                  55V00720
001624        GSC(Z) = QM(Z)/SQRT(Z*(Z-B))                                  55V00730
001643        GSD(Z) = QN(Z)*SQRT((G-Z)/(H-Z))                              55V00740
001663        GSE(Z) = QN(Z)*SQRT((Z-G)/(Z-F))                              55V00750
        C                                                                   55V00760
                                                                            55V00770
001704        DUM = (UP - ALOW) / FLOAT (NPOINT - 1)                        55V00780
001710        IF (DUM .LT. C1) GO TO 501                                    55V00790
001712        IF (NTYPE .EQ.  4) GT = GAA(SING)                             55V00800
001716        IF (NTYPE .EQ.  5) GT = GAB(SING)                             55V00810
001722        IF (NTYPE .EQ.  6) GT = GAB(SING)                             55V00820
001726        IF (NTYPE .EQ.  7) GT = GDD(SING)                             55V00830
001732        IF (NTYPE .EQ.  8) GT = GDD(SING)                             55V00840
001736        IF (NTYPE .EQ.  9) GT = GFF(SING)                             55V00850
001742        IF (NTYPE .EQ. 15) GT = G2A(SING)                             55V00860
001746        IF (NTYPE .EQ. 16) GT = G3A(SING)                             55V00870
001752        IF (NTYPE .EQ. 17) GT = G3A(SING)                             55V00880
001756        IF (NTYPE .EQ. 18) GT = GD2(SING)                             55V00890
001762        IF (NTYPE .EQ. 19) GT = GD2(SING)                             55V00900
001766        IF (NTYPE .EQ. 20) GT = GE2(SING)                             55V00910
001772        IF (NTYPE .EQ. 21) GT = GE2(SING)                             55V00920
001776        IF (NTYPE .EQ. 22) GT = G11(SING)                             55V00930
002002        IF (NTYPE .EQ. 23) GT = G11(SING)                             55V00940
002006        IF (NTYPE .EQ. 24) GT = GT2(SING)                             55V00950
002012        IF (NTYPE .EQ. 28) GT = GAH(SING)                             55V00960
002016        IF (NTYPE .EQ. 29) GT = GCH(SING)                             55V00970
002022        IF (NTYPE .EQ. 30) GT = GCH(SING)                             55V00980
002026        IF (NTYPE .EQ. 31) GT = GDH(SING)                             55V00990
002032        IF (NTYPE .EQ. 32) GT = GDH(SING)                             55V01000
002036        IF (NTYPE .EQ. 33) GT = GFH(SING)                             55V01010
002042        IF (NTYPE .EQ. 40) GT = GYA(SING)                             55V01020
002046        IF (NTYPE .EQ. 41) GT = GYB(SING)                             55V01030
002052        IF (NTYPE .EQ. 42) GT = GYC(SING)                             55V01040
002056        IF (NTYPE .EQ. 43) GT = GYD(SING)                             55V01050
002062        IF (NTYPE .EQ. 44) GT = GYD(SING)                             55V01060
002066        IF (NTYPE .EQ. 45) GT = GYF(SING)                             55V01070
002072        IF (NTYPE .EQ. 46) GT = GYF(SING)                             55V01080
002076        IF (NTYPE .EQ. 47) GT = GYH(SING)                             55V01090
002102        IF (NTYPE .EQ. 48) GT = GYH(SING)                             55V01100
002106        IF (NTYPE .EQ. 49) GT = GYI(SING)                             55V01110
002112        IF (NTYPE .EQ. 55) GT = GWA(SING)                             55V01120
002116        IF (NTYPE .EQ. 56) GT = GWB(SING)                             55V01130
002122        IF (NTYPE .EQ. 57) GT = GWB(SING)                             55V01140
002126        IF (NTYPE .EQ. 58) GT = GWC(SING)                             55V01150
002132        IF (NTYPE .EQ. 59) GT = GWD(SING)
```

```
002136          IF (NTYPE .EQ. 60) GT = GWE(SING)                                   55V01160
002142          IF (NTYPE .EQ. 61) GT = GWF(SING)                                   55V01170
002146          IF (NTYPE .EQ. 62) GT = GWG(SING)                                   55V01180
002152          IF (NTYPE .EQ. 69) GT = GUA(SING)                                   55V01190
002156          IF (NTYPE .EQ. 70) GT = GUB(SING)                                   55V01200
002162          IF (NTYPE .EQ. 71) GT = GUB(SING)                                   55V01210
002166          IF (NTYPE .EQ. 72) GT = GUC(SING)                                   55V01220
002172          IF (NTYPE .EQ. 73) GT = GUD(SING)                                   55V01230
002176          IF (NTYPE .EQ. 74) GT = GUE(SING)                                   55V01240
002202          IF (NTYPE .EQ. 75) GT = GUE(SING)                                   55V01250
002206          IF (NTYPE .EQ. 76) GT = GUF(SING)                                   55V01260
002212          IF (NTYPE .EQ. 77) GT = GUF(SING)                                   55V01270
002216          IF (NTYPE .EQ. 78) GT = GUG(SING)                                   55V01280
002222          IF (NTYPE .EQ. 83) GT = GSA(SING)                                   55V01290
002226          IF (NTYPE .EQ. 84) GT = GSB(SING)                                   55V01300
002232          IF (NTYPE .EQ. 85) GT = GSB(SING)                                   55V01310
002236          IF (NTYPE .EQ. 86) GT = GSC(SING)                                   55V01320
002242          IF (NTYPE .EQ. 87) GT = GSC(SING)                                   55V01330
002246          IF (NTYPE .EQ. 88) GT = 0.0                                         55V01340
002251          IF (NTYPE .EQ. 89) GT = GSD(SING)                                   55V01350
002255          IF (NTYPE .EQ. 90) GT = GSD(SING)                                   55V01360
002261          IF (NTYPE .EQ. 91) GT = GSE(SING)                                   55V01370
002265          FF2 = 0.005                                                         55V01380
002267          FF  = FF2 + FF2                                                     55V01390
002270          IF ((SING - ALOW) .GT. (UP - SING)) GO TO 401                       55V01400
002275          ABSC(1) = ALOW                                                      55V01410
002275          DO 400 I=2,NPOINT                                                   55V01420
002277          ABSC(I) = ABSC(I-1) + FF * (UP - ABSC(I-1)) / FLOAT(NPOINT-I+1)     55V01430
002310          FF = FF + FF                                                        55V01440
002311          IF (FF .GT. 1.0) FF = 1.0                                           55V01450
002314      400 CONTINUE                                                            55V01460
002317          GO TO 403                                                           55V01470
002317      401 ABSC(NPOINT) = UP                                                   55V01480
002322          DO 402 I=2,NPOINT                                                   55V01490
002323          J = NPOINT -I + 1                                                   55V01500
002325          K = J + 1                                                           55V01510
002327          ABSC(J) = ABSC(K) - FF * (ABSC(K) - ALOW) / FLOAT(J)                55V01520
002337          FF = FF + FF                                                        55V01530
002340          IF (FF .GT. 1.0) FF = 1.0                                           55V01540
002343      402 CONTINUE                                                            55V01550
002346      403 DO 500 I=1,NPOINT                                                   55V01560
002350          AB = ABSC(I)                                                        55V01570
002352          IF (ABS(AB - SING) .LT. FF2 * DUM) GO TO 499                        55V01580
002357          GO TO (1,1,1,4,5,6,7,8,9,1,1,1,1,1,15,16,17,18,19,20,21,22,23,24,   55V01590
               A 1,1,1,28,29,30,31,35,33,1,1,1,1,1,1,40,41,42,43,44,45,46,47,48,   55V01600
               B 49,1,1,1,1,1,55,56,57,58,59,60,61,62,1,1,1,1,1,1,69,70,71,72,73,  55V01610
               C 74,75,76,77,78,1,1,1,1,83,84,85,86,87,88,89,90,91                 55V01620
               Z    ) , NTYPE                                                      55V01630
002516        1 WRITE (6,2)  NTYPE                                                 55V01640
002524        2 FORMAT (1H0, 39H SUBROUTINE NUMINT CALLED WITH NTYPE =  ,I7//)     55V01650
002524          GO TO 501                                                           55V01660
002530        4 ORD(I) = (GAA(AB) - GT) / SQRT(AB)                                 55V01670
002541          GO TO 500                                                           55V01680
002545        5 ORD(I) = (GAB(AB) - GT) / SQRT(B - AB)                             55V01690
002560          GO TO 500                                                           55V01700
002564        6 ORD(I) = (GAB(AB) - GT) / SQRT(AB - B)                             55V01710
002577          GO TO 500                                                           55V01720
002603        7 ORD(I) = (GDD(AB) - GT) / SQRT (D - AB)                            55V01730
```

```
002616          GO TO 500                                                   55V01740
002622        8 ORD(I) = (GDD(AB) - GT) / SQRT (AB - D)                     55V01750
002635          GO TO 500                                                   55V01760
002641        9 ORD(I) = (GFF(AB) - GT) / SQRT (F - AB)                     55V01770
002654          GO TO 500                                                   55V01780
002660       15 ORD(I) = (G2A(AB) - GT) / SQRT (AB - A)                     55V01790
002673          GO TO 500                                                   55V01800
002677       16 ORD(I) = (G3A(AB) - GT) / SQRT (B - AB)                     55V01810
002712          GO TO 500                                                   55V01820
002716       17 ORD(I) = (G3A(AB) - GT) / SQRT (AB - B)                     55V01830
002731          GO TO 500                                                   55V01840
002735       18 ORD(I) = (GD2(AB) - GT) / SQRT (-AB)                        55V01850
002747          GO TO 500                                                   55V01860
002753       19 ORD(I) = (GD2(AB) - GT) / SQRT (AB)                         55V01870
002764          GO TO 500                                                   55V01880
002770       20 ORD(I) = (GE2(AB) - GT) / SQRT (E - AB)                     55V01890
003003          GO TO 500                                                   55V01900
003007       21 ORD(I) = (GE2(AB) - GT) / SQRT (AB - E)                     55V01910
003022          GO TO 500                                                   55V01920
003026       22 ORD(I) = (G11(AB) - GT) / SQRT (1.0 - AB)                   55V01930
003041          GO TO 500                                                   55V01940
003045       23 ORD(I) = (G11(AB) - GT) / SQRT (AB - 1.0)                   55V01950
003060          GO TO 500                                                   55V01960
003064       24 ORD(I) = (GI2(AB) - GT) / SQRT (AI - AB)                    55V01970
003077          GO TO 500                                                   55V01980
003103       28 ORD(I) = (GAH(AB) - GT) / SQRT (AB - A)                     55V01990
003116          GO TO 500                                                   55V02000
003122       29 ORD(I) = (GCH(AB) - GT) / SQRT (-AB)                        55V02010
003134          GO TO 500                                                   55V02020
003140       30 ORD(I) = (GCH(AB) - GT) / SQRT (AB)                         55V02030
003151          GO TO 500                                                   55V02040
003155       31 ORD(I) = (GDH(AB) - GT) / SQRT (D - AB)                     55V02050
003170          GO TO 500                                                   55V02060
003174       32 ORD(I) = (GDH(AB) - GT) / SQRT (AB - D)                     55V02070
003207          GO TO 500                                                   55V02080
003213       33 ORD(I) = (GFH(AB) - GT) / SQRT (F - AB)                     55V02090
003226          GO TO 500                                                   55V02100
003232       40 ORD(I) = (GYA(AB) - GT) / SQRT (AB - A)                     55V02110
003245          GO TO 500                                                   55V02120
003251       41 ORD(I) = (GYB(AB) - GT) / SQRT (B - AB)                     55V02130
003264          GO TO 500                                                   55V02140
003270       42 ORD(I) = (GYC(AB) - GT) / SQRT (AB - B)                     55V02150
003303          GO TO 500                                                   55V02160
003307       43 ORD(I) = (GYD(AB) - GT) / SQRT (-AB)                        55V02170
003321          GO TO 500                                                   55V02180
003325       44 ORD(I) = (GYD(AB) - GT) / SQRT (AB)                         55V02190
003336          GO TO 500                                                   55V02200
003342       45 ORD(I) = (GYF(AB) - GT) / SQRT (F - AB)                     55V02210
003355          GO TO 500                                                   55V02220
003361       46 ORD(I) = (GYF(AB) - GT) / SQRT (AB - F)                     55V02230
003374          GO TO 500                                                   55V02240
003400       47 ORD(I) = (GYH(AB) - GT) / SQRT (H - AB)                     55V02250
003413          GO TO 500                                                   55V02260
003417       48 ORD(I) = (GYH(AB) - GT) / SQRT (AB - H)                     55V02270
003432          GO TO 500                                                   55V02280
003436       49 ORD(I) = (GYI(AB) - GT) / SQRT (AI - AB)                    55V02290
003451          GO TO 500                                                   55V02300
003455       55 ORD(I) = (GWA(AB)- GT) / SQRT (AB - A)                      55V02310
```

```
003470          GO TO 500
003474       56 ORD(I) = (GWB(AB)- GT) / SQRT (-AB)
003506          GO TO 500
003512       57 ORD(I) = (GWB(AB)- GT) / SQRT (AB)
003523          GO TO 500
003527       58 ORD(I) = (GWC(AB)- GT) * (D - AB) ** GPM2
003543          GO TO 500
003543       59 ORD(I) = (GWD(AB)- GT) * (AB - D) ** GPM2
003557          GO TO 500
003557       60 ORD(I) = (GWE(AB)- GT) * (F - AB) ** (-GP)
003574          GO TO 500
003574       61 ORD(I) = (GWF(AB)- GT) * (AB - F) ** (-GP)
003611          GO TO 500
003611       62 ORD(I) = (GWG(AB)- GT) / SQRT (G - AB)
003624          GO TO 500
003630       69 ORD(I) = (GUA(AB) - GT) / SQRT(AB - A)
003643          GO TO 500
003647       70 ORD(I) = (GUB(AB) - GT) / SQRT(-AB)
003661          GO TO 500
003665       71 ORD(I) = (GUB(AB) - GT) / SQRT(AB)
003676          GO TO 500
003702       72 ORD(I) = (GUC(AB) - GT) / SQRT(D - AB)
003715          GO TO 500
003721       73 ORD(I) = (GUD(AB) - GT) / SQRT(AB - D)
003734          GO TO 500
003740       74 ORD(I) = (GUE(AB) - GT) * (E - AB) ** GP
003754          GO TO 500
003754       75 ORD(I) = (GUE(AB) - GT) * (AB - E) ** GP
003770          GO TO 500
003770       76 ORD(I) = (GUF(AB) - GT) / (G - AB) ** GP
004004          GO TO 500
004004       77 ORD(I) = (GUF(AB) - GT) / (AB - G) ** GP
004020          GO TO 500
004020       78 ORD(I) = (GUG(AB) - GT) / SQRT(H - AB)
004033          GO TO 500
004037       83 ORD(I) = (GSA(AB) - GT) / SQRT(AB)
004050          GO TO 500
004054       84 ORD(I) = (GSB(AB) - GT) / SQRT(B - AB)
004067          GO TO 500
004073       85 ORD(I) = (GSB(AB) - GT) / SQRT(AB - B)
004106          GO TO 500
004112       86 ORD(I) = (GSC(AB) - GT) / SQRT(C - AB)
004125          GO TO 500
004131       87 ORD(I) = (GSC(AB) - GT) / SQRT(AB - C)
004144          GO TO 500
004150       88 ORD(I) = GSC(AB) / SQRT(AB-C)
004162          GO TO 500
004166       89 ORD(I) = (GSD(AB) - GT) / SQRT(F - AB)
004201          GO TO 500
004205       90 ORD(I) = (GSD(AB) - GT) / SQRT(AB - F)
004220          GO TO 500
004224       91 ORD(I) = (GSE(AB) - GT) / SQRT(H - AB)
004237          GO TO 500
004243      499 ORD(I) = 0.0
004245      500 CONTINUE
004250          DUMMY = ABSC(1)
004251          DO 502 I=1,NPOINT
004253      502 ABSC(I) = ABSC(I) - DUMMY
```

```
004260          CALL ITS1 (ABSC, ORD, TABLE, NPOINT)                    55V02900
004262          ANSWER = TABLE(NPOINT)                                  55V02910
004270          IF (ILAST .EQ. 0) RETURN                                55V02920
004272          DO 503 I=1,NPOINT                                       55V02930
004274      503 ABSC(I) = ABSC(I) + DUMMY                               55V02940
004301          RETURN                                                  55V02950
004301      501 ANSWER = 0.0                                            55V02960
004302          RETURN                                                  55V02970
004302          END                                                     55V02980
```

SUBPROGRAM LENGTH
006515

FUNCTION ASSIGNMENTS

GAA	-	000274	GAB	-	000321	GAH	-	000646	GCH	-	000670
GDD	-	000344	GDH	-	000714	GD2	-	000474	GE2	-	000530
GFF	-	000366	GFH	-	000737	GI2	-	000614	GSA	-	001566
GSH	-	001606	GSC	-	001625	GSD	-	001644	GSE	-	001664
GUA	-	001432	GUB	-	001450	GUC	-	001467	GUD	-	001502
GUE	-	001515	GUF	-	001534	GUG	-	001553	GWA	-	001260
GWB	-	001304	GWC	-	001331	GWD	-	001350	GWE	-	001364
GWF	-	001377	GWG	-	001416	GYA	-	000761	GYB	-	001014
GYC	-	001051	GYD	-	001106	GYF	-	001142	GYH	-	001174
GYI	-	001226	G11	-	000562	G2A	-	000410	G3A	-	000442
QC	-	000012	QD	-	000042	QE	-	000064	QH	-	000113
QI	-	000145	QJ	-	000176	QM	-	000227	QN	-	000252

STATEMENT ASSIGNMENTS

1	-	002516	2	-	004310	4	-	002530	5	-	002545
6	-	002564	7	-	002603	8	-	002622	9	-	002641
15	-	002660	16	-	002677	17	-	002716	18	-	002735
19	-	002753	20	-	002770	21	-	003007	22	-	003026
23	-	003045	24	-	003064	28	-	003103	29	-	003122
30	-	003140	31	-	003155	32	-	003174	33	-	003213
40	-	003232	41	-	003251	42	-	003270	43	-	003307
44	-	003325	45	-	003342	46	-	003361	47	-	003400
48	-	003417	49	-	003436	55	-	003455	56	-	003474
57	-	003512	58	-	003527	59	-	003543	60	-	003557
61	-	003574	62	-	003611	69	-	003630	70	-	003647
71	-	003665	72	-	003702	73	-	003721	74	-	003740
75	-	003754	76	-	003770	77	-	004004	78	-	004020
83	-	004037	84	-	004054	85	-	004073	86	-	004112
87	-	004131	88	-	004150	89	-	004166	90	-	004205
91	-	004224	400	-	002314	401	-	002317	402	-	002343
403	-	002346	499	-	004243	500	-	004245	501	-	004301
502	-	004254	503	-	004275						

BLOCK NAMES AND LENGTHS
- 002420

VARIABLE ASSIGNMENTS

A	-	000000C01	AA	-	002217C01	AB	-	006513	ABSC	-	000021C01
AI	-	000010C01	ALOAD	-	000247C01	B	-	000001C01	C	-	000002C01
C1	-	002415C01	D	-	000003C01	DATA	-	000000C01	DUM	-	006505
DUMMY	-	006514	E	-	000004C01	F	-	000005C01	FF	-	006507
FF2	-	006506	G	-	000006C01	GP	-	002404C01	GPM2	-	002405C01
GT	-	000000	H	-	000007C01	I	-	006510	ILAST	-	002413C01
J	-	006511	K	-	006512	ORD	-	000103C01	TAB	-	002237C01
TABLE	-	000165C01									

START OF CONSTANTS
004304

START OF TEMPORARIES
004317

START OF INDIRECTS

000410

UNUSED COMPILER SPACE
027200

```
SUBROUTINE NUMINT (UP,ALOW,SING,NPOINT ,NTYPE,ANSWER,GT)
```

START

```
QC(Z)=(Z-C)*(Z-1.0)/SQRT(G-Z)*(F-Z)**(-GP)
QD(Z)=(Z-C)*(Z-1.0)/SQRT((Z-A)*Z*(G-Z))
QE(Z)=(Z-C)*(Z-1.0)/SQRT((Z-A)*Z)*(Z-D)**GPM2
QH(Z)=(Z-C)*(Z-1.0)/SQRT((D-Z)*(H-Z))*((E-Z)/(G-Z))**GP
```

```
QI(Z)=(Z-C)*(Z-1.0)/SQRT((Z-A)*Z*(H-Z))/(G-Z)**GP
QJ(Z)=(Z-C)*(Z-1.0)/SQRT((Z-A)*Z*(Z-D))*(Z-E)**GP
QM(Z)=(Z-D)*(Z-1.0)*SQRT((G-Z)/((F-Z)*(H-Z)))
QN(Z)=(Z-D)*(Z-1.0)/SQRT(Z*(Z-B)*(Z-C))
```

```
GAA(Z)=(Z-C)*(Z-1.0)/SQRT((B-Z)*(D-Z)*(F-Z))
GAB(Z)=(Z-C)*(Z-1.0)/SQRT(Z*(D-Z)*(F-Z))
GDD(Z)=(Z-C)*(Z-1.0)/SQRT(Z*(Z-B)*(F-Z))
GFF(Z)=(Z-C)*(Z-1.0)/SQRT(Z*(Z-B)*(Z-D))
```

```
G2A(Z)=(Z-C)*(Z-F)*(Z-H)/SQRT((B-Z)*Z*(Z-E)*(Z-1.0)*(Z-AI))
G3A(Z)=(Z-C)*(Z-F)*(Z-H)/SQRT((Z-A)*Z*(Z-E)*(Z-1.0)*(Z-AI))
GD2(Z)=(Z-C)*(Z-F)*(Z-H)/SQRT((Z-A)*(Z-B)*(E-Z)*(Z-1.0)*(Z-AI))
GE2(Z)=(Z-C)*(Z-F)*(Z-H)/SQRT((Z-A)*(Z-B)*Z*(Z-1.0)*(Z-AI))
```

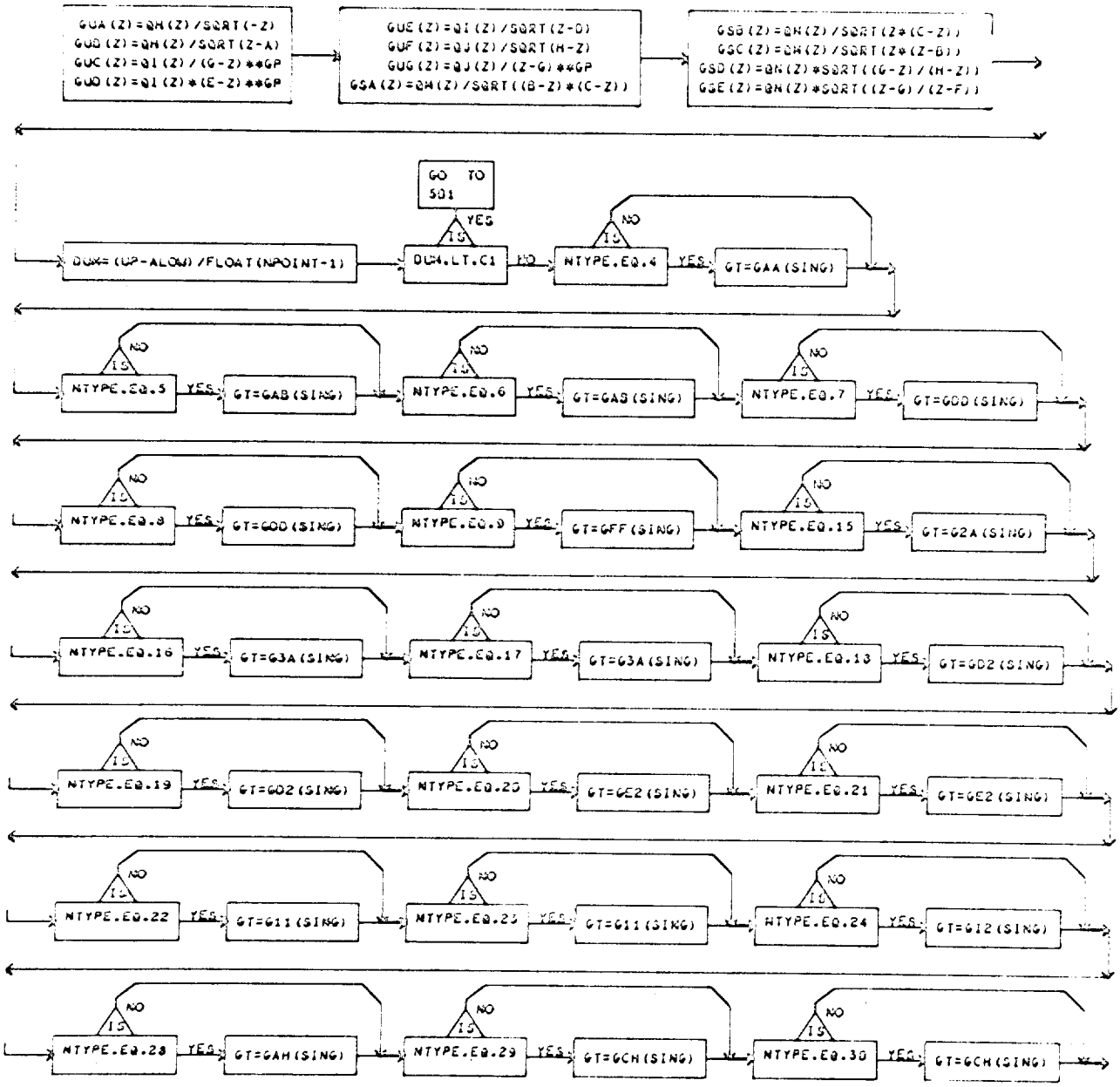
```
G11(Z)=(Z-C)*(Z-F)*(Z-H)/SQRT((Z-A)*(Z-B)*Z*(Z-E)*(AI-Z))
G12(Z)=(Z-C)*(Z-F)*(Z-H)/SQRT((Z-A)*(Z-B)*Z*(Z-E)*(Z-1.0))
GAH(Z)=(Z-B)*(Z-1.0)/SQRT(Z*(Z-D)*(F-Z))
GCH(Z)=(Z-B)*(Z-1.0)/SQRT((Z-A)*(Z-D)*(Z-F))
```

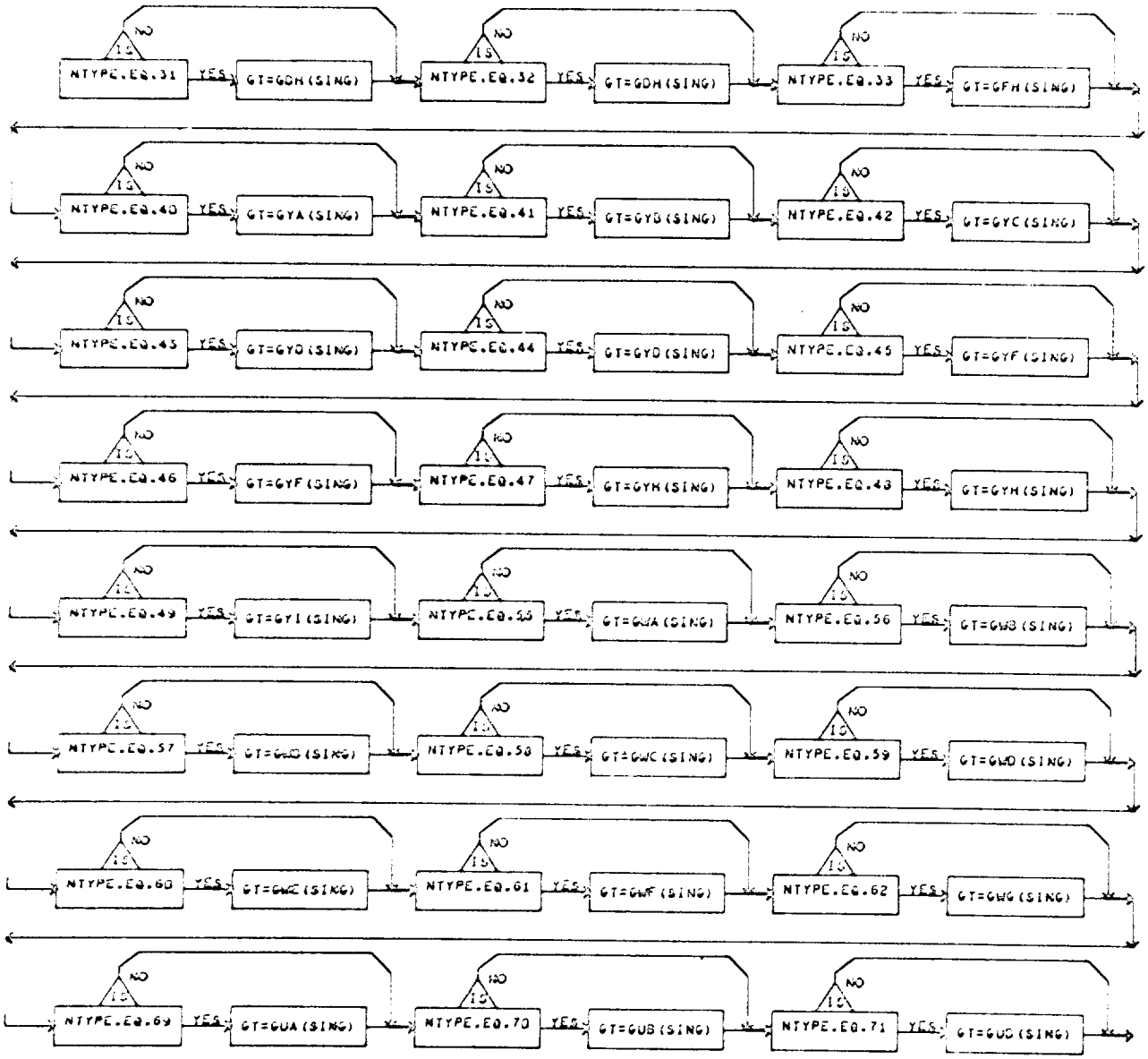
```
GDH(Z)=(Z-B)*(Z-1.0)/SQRT((Z-A)*Z*(F-Z))
GFH(Z)=(Z-B)*(Z-1.0)/SQRT((Z-A)*Z*(Z-D))
GYA(Z)=(Z-C)*(Z-E)*(Z-1.0)/SQRT((Z-B)*Z*(F-Z)*(H-Z)*(AI-Z))
GYB(Z)=(Z-C)*(Z-E)*(Z-1.0)/SQRT((A-Z)*Z*(F-Z)*(H-Z)*(AI-Z))
```

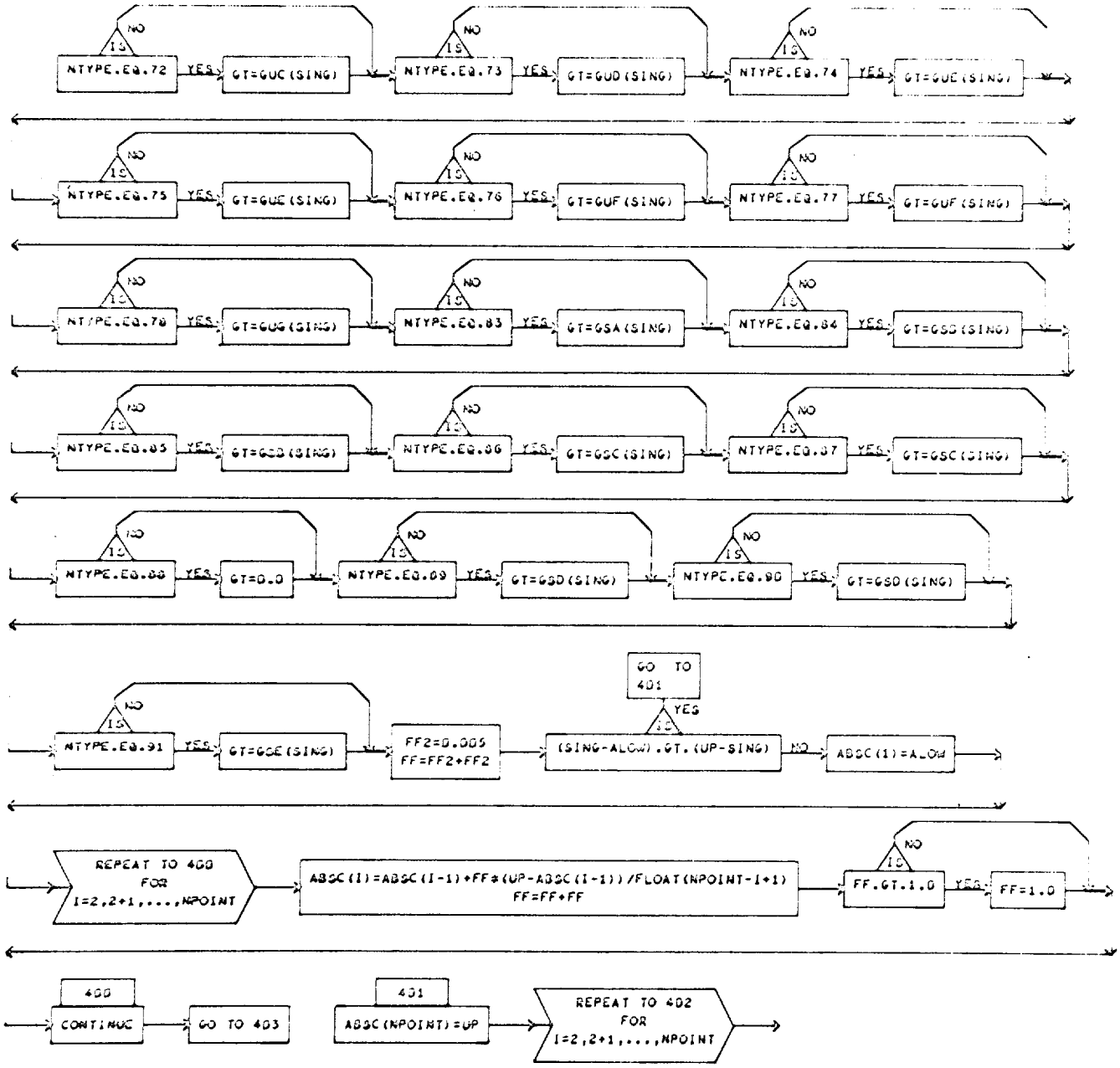
```
GYC(Z)=(Z-C)*(Z-E)*(Z-1.0)/SQRT((A-Z)*Z*(F-Z)*(H-Z)*(AI-Z))
GYD(Z)=(Z-C)*(Z-E)*(Z-1.0)/SQRT((Z-A)*(Z-B)*(F-Z)*(H-Z)*(AI-Z))
GYF(Z)=(Z-C)*(Z-E)*(Z-1.0)/SQRT((Z-A)*(Z-B)*Z*(H-Z)*(AI-Z))
GYH(Z)=(Z-C)*(Z-E)*(Z-1.0)/SQRT((Z-A)*(Z-B)*Z*(Z-F)*(AI-Z))
```

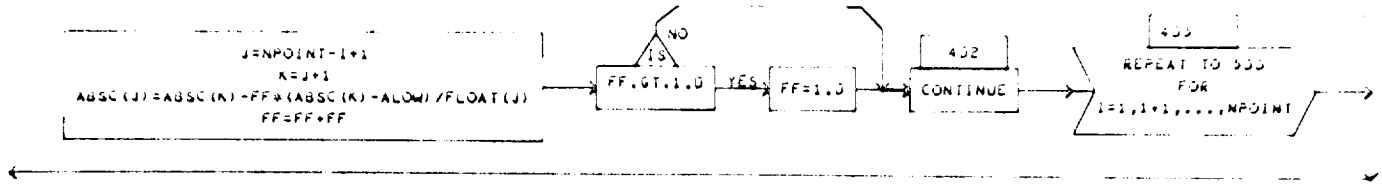
```
GYI(Z)=(Z-C)*(Z-E)*(Z-1.0)/SQRT((Z-A)*(Z-B)*Z*(Z-F)*(Z-H))
GWA(Z)=QC(Z)/SQRT(Z-A)*(D-Z)**GPM2
GWB(Z)=QC(Z)/SQRT(Z-A)*(D-Z)**GPM2
GWC(Z)=QC(Z)/SQRT((Z-A)*Z)
```

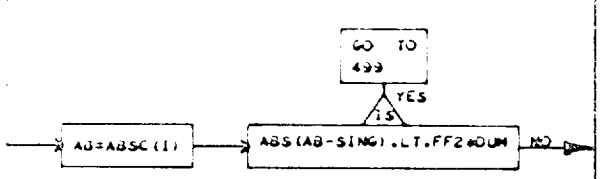
```
GWD(Z)=QD(Z)*(F-Z)**(-GP)
GWE(Z)=QD(Z)*(Z-D)**GPM2
GWF(Z)=QE(Z)/SQRT(G-Z)
GWG(Z)=QE(Z)*(Z-F)**(-GP)
```


GUA(Z)=QH(Z)/SQRT(-Z)
GUB(Z)=QH(Z)/SQRT(Z-A)
GUC(Z)=QI(Z)/(G-Z)**GP
GUD(Z)=QI(Z)*(E-Z)**GP
GUE(Z)=QI(Z)/SQRT(Z-D)
GUF(Z)=QJ(Z)/SQRT(H-Z)
GUG(Z)=QJ(Z)/(Z-G)**GP
GSA(Z)=QH(Z)/SQRT((B-Z)*(C-Z))
GSB(Z)=QH(Z)/SQRT(Z*(C-Z))
GSC(Z)=QH(Z)/SQRT(Z*(Z-B))
GSD(Z)=QH(Z)*SQRT((G-Z)/(H-Z))
GSE(Z)=QH(Z)*SQRT((Z-G)/(Z-F))
GO TO 501
DUX=(UP-ALOW)/FLOAT(NPOINT-1)
DUX.LT.C1
NTYPE.EQ.4
GT=GAA(SING)
NTYPE.EQ.5
GT=GAB(SING)
NTYPE.EQ.6
GT=GAB(SING)
NTYPE.EQ.7
GT=GOD(SING)
NTYPE.EQ.8
GT=GOD(SING)
NTYPE.EQ.9
GT=GFF(SING)
NTYPE.EQ.15
GT=G2A(SING)
NTYPE.EQ.16
GT=G3A(SING)
NTYPE.EQ.17
GT=G3A(SING)
NTYPE.EQ.18
GT=GD2(SING)
NTYPE.EQ.19
GT=GD2(SING)
NTYPE.EQ.20
GT=GE2(SING)
NTYPE.EQ.21
GT=GE2(SING)
NTYPE.EQ.22
GT=G11(SING)
NTYPE.EQ.23
GT=G11(SING)
NTYPE.EQ.24
GT=G12(SING)
NTYPE.EQ.28
GT=GAH(SING)
NTYPE.EQ.29
GT=GCH(SING)
NTYPE.EQ.30
GT=GCH(SING)
YES
NO


NTYPE.EQ.31
YES
NO
IS
GT=GDH(SING)
NTYPE.EQ.32
GT=GDH(SING)
NTYPE.EQ.33
GT=GFH(SING)
NTYPE.EQ.40
GT=GYA(SING)
NTYPE.EQ.41
GT=GYB(SING)
NTYPE.EQ.42
GT=GYC(SING)
NTYPE.EQ.43
GT=GYD(SING)
NTYPE.EQ.44
GT=GYD(SING)
NTYPE.EQ.45
GT=GYF(SING)
NTYPE.EQ.46
GT=GYF(SING)
NTYPE.EQ.47
GT=GYH(SING)
NTYPE.EQ.48
GT=GYH(SING)
NTYPE.EQ.49
GT=GYI(SING)
NTYPE.EQ.55
GT=GWA(SING)
NTYPE.EQ.56
GT=GWB(SING)
NTYPE.EQ.57
GT=GWD(SING)
NTYPE.EQ.58
GT=GWC(SING)
NTYPE.EQ.59
GT=GWD(SING)
NTYPE.EQ.60
GT=GWE(SING)
NTYPE.EQ.61
GT=GWF(SING)
NTYPE.EQ.62
GT=GWG(SING)
NTYPE.EQ.69
GT=GUA(SING)
NTYPE.EQ.70
GT=GUB(SING)
NTYPE.EQ.71
GT=GUD(SING)


NTYPE.EQ.72
YES
GT=GUC(SING)
NO
NTYPE.EQ.73
YES
GT=GUD(SING)
NO
NTYPE.EQ.74
YES
GT=GUE(SING)
NO
NTYPE.EQ.75
YES
GT=GUE(SING)
NO
NTYPE.EQ.76
YES
GT=GUF(SING)
NO
NTYPE.EQ.77
YES
GT=GUF(SING)
NO
NTYPE.EQ.78
YES
GT=GUG(SING)
NO
NTYPE.EQ.83
YES
GT=GSA(SING)
NO
NTYPE.EQ.84
YES
GT=GSB(SING)
NO
NTYPE.EQ.85
YES
GT=GSB(SING)
NO
NTYPE.EQ.86
YES
GT=GSC(SING)
NO
NTYPE.EQ.87
YES
GT=GSC(SING)
NO
NTYPE.EQ.88
YES
GT=0.0
NO
NTYPE.EQ.89
YES
GT=GSD(SING)
NO
NTYPE.EQ.90
YES
GT=GSD(SING)
NO
NTYPE.EQ.91
YES
GT=GSE(SING)
FF2=0.005
FF=FF2+FF2
(SING-ALOW).GT.(UP-SING)
YES
GO TO 401
NO
ABSC(1)=ALOW
REPEAT TO 400
FOR
I=2,2+1,...,NPOINT
ABSC(I)=ABSC(I-1)+FF*(UP-ABSC(I-1))/FLOAT(NPOINT-I+1)
FF=FF+FF
FF.GT.1.0
YES
FF=1.0
NO
400
CONTINUE
GO TO 403
401
ABSC(NPOINT)=UP
REPEAT TO 402
FOR
I=2,2+1,...,NPOINT

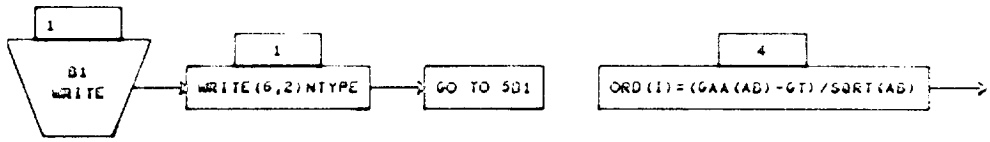


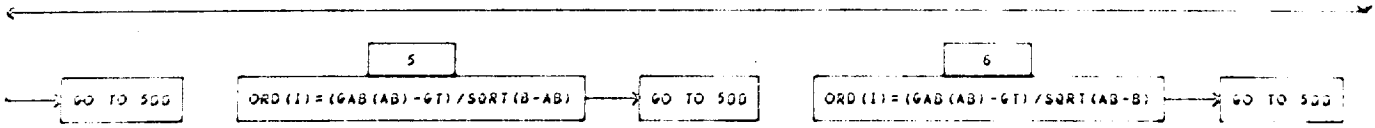


COMPUTED GO TO

IF THE VALUE OF NTYPE IS	TRANSFER TO STATEMENT
1	1
2	1
3	1
4	4
5	5
6	6
7	7
8	8
9	9
10	1
11	1
12	1
13	1
14	1
15	15
16	16
17	17
18	18
19	19
20	20
21	21
22	22
23	23
24	24
25	1
26	1
27	1
28	28
29	29
30	30
31	31
32	32
33	33
34	1
35	1
36	1
37	1
38	1
39	1
40	40
41	41
42	42
43	43
44	44

45	V	V	V 45
46			46
47			47
48			48
49			49
50			1
51			1
52			1
53			1
54			1
55			55
56			56
57			57
58			58
59			59
60			60
61			61
62			62
63			1
64			1
65			1
66			1
67			1
68			1
69			69
70			70
71			71
72			72
73			73
74			74
75			75
76			76
77			77
78			78
79			1
80			1
81			1
82			1
83			83
84			84
85			85
86			86
87			87
88			88
89			89
90			90
91			91





7

```
ORD(I)=(GDD(AB)-GT)/SQRT(D-AB)
```

→ GO TO 500

8

```
ORD(I)=(GDD(AB)-GT)/SQRT(AB-D)
```

→ GO TO 500

9

```
ORD(I)=(GFF(AB)-GT)/SQRT(F-AB)
```

→ GO TO 500

15

```
ORD(I)=(G2A(AB)-GT)/SQRT(AB-A)
```

→ GO TO 500

16

```
ORD(I)=(G3A(AB)-GT)/SQRT(B-AB)
```

→ GO TO 500

17

```
ORD(I)=(G3A(AB)-GT)/SQRT(AB-B)
```

→ GO TO 500

18

```
ORD(I)=(GD2(AB)-GT)/SQRT(-AB)
```

→ GO TO 500

19

```
ORD(I)=(GD2(AB)-GT)/SQRT(AB)
```

→ GO TO 500

20

```
ORD(I)=(GE2(AB)-GT)/SQRT(E-AB)
```

→ GO TO 500

21

```
ORD(I)=(GE2(AB)-GT)/SQRT(AB-E)
```

→ GO TO 500

22

```
ORD(I)=(G11(AB)-GT)/SQRT(1.0-AB)
```

→ GO TO 500

23

```
ORD(I)=(G11(AB)-GT)/SQRT(AB-1.0)
```

→ GO TO 500

24

```
ORD(I)=(G12(AB)-GT)/SQRT(A1-AB)
```

→ GO TO 500

25

```
ORD(I)=(GAH(AB)-GT)/SQRT(AB-A)
```

→ GO TO 500

29

```
ORD(I)=(GCH(AB)-GT)/SQRT(-AB)
```

→ GO TO 500

30

```
ORD(I)=(GCH(AB)-GT)/SQRT(AB)
```

→ GO TO 500

31

```
ORD(I)=(GDH(AB)-GT)/SQRT(D-AB)
```

→ GO TO 500

32

```
ORD(I)=(GDH(AB)-GT)/SQRT(AB-D)
```

→ GO TO 500

```
33
ORD(I)=(GFH(AB)-GT)/SQRT(F-AB)  ->  GO TO 500

40
ORD(I)=(GYA(AB)-GT)/SQRT(AB-A)  ->  GO TO 500

41
ORD(I)=(GYB(AB)-GT)/SQRT(B-AB)  ->  GO TO 500

42
ORD(I)=(GYC(AB)-GT)/SQRT(AB-B)  ->  GO TO 500

43
ORD(I)=(GYD(AB)-GT)/SQRT(-AB)  ->  GO TO 500

44
ORD(I)=(GYD(AB)-GT)/SQRT(AB)  ->  GO TO 500

45
ORD(I)=(GYF(AB)-GT)/SQRT(F-AB)  ->  GO TO 500

46
ORD(I)=(GYF(AB)-GT)/SQRT(AB-F)  ->  GO TO 500

47
ORD(I)=(GYH(AB)-GT)/SQRT(H-AB)  ->  GO TO 500

48
ORD(I)=(GYH(AB)-GT)/SQRT(AB-H)  ->  GO TO 500

49
ORD(I)=(GYI(AB)-GT)/SQRT(AI-AB)  ->  GO TO 500

55
ORD(I)=(GWA(AB)-GT)/SQRT(AB-A)  ->  GO TO 500

56
ORD(I)=(GWD(AB)-GT)/SQRT(-AB)  ->  GO TO 500

57
ORD(I)=(GWD(AB)-GT)/SQRT(AB)  ->  GO TO 500

58
ORD(I)=(GWC(AB)-GT)*(D-AB)**GPK2  ->  GO TO 500

59
ORD(I)=(GWD(AB)-GT)*(AB-D)**GPK2  ->  GO TO 500

60
ORD(I)=(GWE(AB)-GT)*(F-AB)**(-GP)  ->  GO TO 500

61
ORD(I)=(GWF(AB)-GT)*(AB-F)**(-GP)  ->  GO TO 500
```

62

```
ORD(I)=(GWG(AB)-GT)/SQRT(G-AB)
```

→ GO TO 500

69

```
ORD(I)=(GUA(AB)-GT)/SQRT(AB-A)
```

→ GO TO 500

70

```
ORD(I)=(GUB(AB)-GT)/SQRT(-AB)
```

→ GO TO 500

71

```
ORD(I)=(GUB(AB)-GT)/SQRT(AB)
```

→ GO TO 500

72

```
ORD(I)=(GUC(AB)-GT)/SQRT(D-AB)
```

→ GO TO 500

73

```
ORD(I)=(GUD(AB)-GT)/SQRT(AB-D)
```

→ GO TO 500

74

```
ORD(I)=(GUE(AB)-GT)*(E-AB)**GP
```

→ GO TO 500

75

```
ORD(I)=(GUE(AB)-GT)*(AB-E)**GP
```

→ GO TO 500

76

```
ORD(I)=(GUF(AB)-GT)/(G-AB)**GP
```

→ GO TO 500

77

```
ORD(I)=(GUF(AB)-GT)/(AB-G)**GP
```

→ GO TO 500

78

```
ORD(I)=(GUG(AB)-GT)/SQRT(H-AB)
```

→ GO TO 500

83

```
ORD(I)=(GSA(AB)-GT)/SQRT(AB)
```

→ GO TO 500

84

```
ORD(I)=(GSB(AB)-GT)/SQRT(B-AB)
```

→ GO TO 500

85

```
ORD(I)=(GSB(AB)-GT)/SQRT(AB-B)
```

→ GO TO 500

86

```
ORD(I)=(GSC(AB)-GT)/SQRT(C-AB)
```

→ GO TO 500

87

```
ORD(I)=(GSC(AB)-GT)/SQRT(AB-C)
```

→ GO TO 500

88

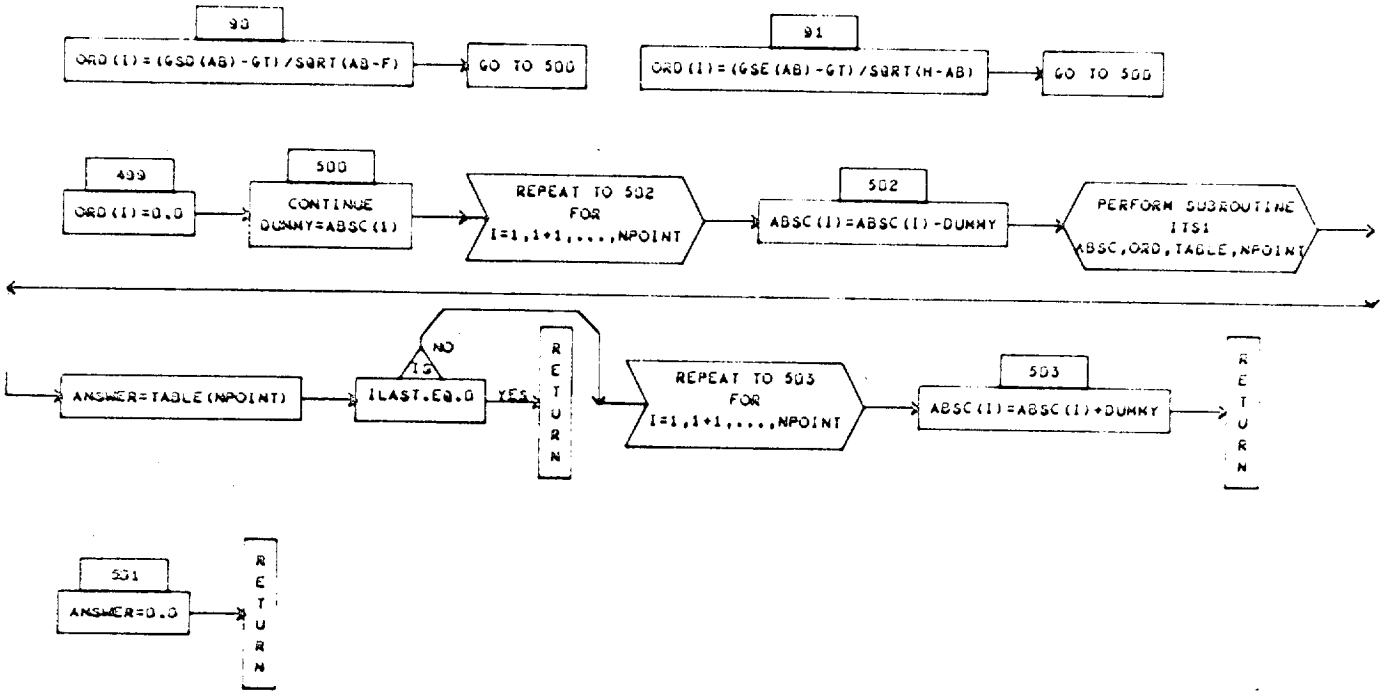
```
ORD(I)=GSC(AB)/SQRT(AB-C)
```

→ GO TO 500

89

```
ORD(I)=(GSD(AB)-GT)/SQRT(F-AB)
```

→ GO TO 500


93
ORD(I)=(GSD(AB)-GT)/SQRT(AB-F)
GO TO 500
91
ORD(I)=(GSE(AB)-GT)/SQRT(H-AB)
GO TO 500
499
ORD(I)=0.0
500
CONTINUE
DUMMY=ABSC(1)
REPEAT TO 502
FOR
I=1,1+1,...,NPOINT
502
ABSC(I)=ABSC(I)-DUMMY
PERFORM SUBROUTINE
ITS1
ABSC,ORD,TABLE,NPOINT
ANSWER=TABLE(NPOINT)
ILAST.EQ.0
NO
YES
RETURN
REPEAT TO 503
FOR
I=1,1+1,...,NPOINT
503
ABSC(I)=ABSC(I)+DUMMY
RETURN
501
ANSWER=0.0
RETURN

SYMBOLS USED IN SUBROUTINE NUMINT (DECK IDENTIFICATION 55V0)

A	R	C	POTENTIAL AT A (MAPPING CONSTANT)	NUMINT
AA	R	C	ARRAY FOR RESULTS OF GEOMETRY AND DRAG INTEGRALS	NUMINT
AB	R	U	CURRENT VALUE OF ABSCISSA	NUMINT
ABSC	R	C	ABSCISSA ARRAY FOR NUMERICAL INTEGRALS	NUMINT
AI	R	C	POTENTIAL AT I (MAPPING CONSTANT)	NUMINT
ALOAD	R	C	STORAGE ARRAY FOR POTENTIAL IN THE REAL (PHYSICAL) PLANE	NUMINT
ALOW	R	U	POTENTIAL AT THE LOWER INTEGRATION LIMIT	NUMINT
ANSWER	R	U	RESULT OF NUMERICAL INTEGRATION	NUMINT
B	R	C	POTENTIAL AT B (MAPPING CONSTANT)	NUMINT
C	R	C	POTENTIAL AT C (MAPPING CONSTANT)	NUMINT
CG	R	C	COSINE OF DIHEDRAL ANGLE	NUMINT
CONI	R	C	FACTOR FOR IMAJINARY PART OF COMPLEX DISTANCE	NUMINT
CONR	R	C	FACTOR FOR REAL PART OF COMPLEX DISTANCE	NUMINT
C1	R	C	CONSTANT THAT IS COMPUTER HARDWARE DEPENDENT	NUMINT
C2	R	C	CONSTANT THAT IS COMPUTER HARDWARE DEPENDENT	NUMINT
C3	R	C	CONSTANT THAT IS COMPUTER HARDWARE DEPENDENT	NUMINT
D	R	C	POTENTIAL AT D (MAPPING CONSTANT)	NUMINT
DATA	R	C	ARRAY FOR INPUT DATA	NUMINT
DUM	R	U	INTERMEDIATE CALCULATION VARIABLE	NUMINT
DUMMY	R	U	TEMPORARY CALCULATION VARIABLE	NUMINT
E	R	C	POTENTIAL AT E (MAPPING CONSTANT)	NUMINT
ETAIRQ	R	C	INBOARD SPANWISE LOCATION (NONDIMENSIONAL)	NUMINT
ETAORQ	R	C	OUTBOARD SPANWISE LOCATION (NONDIMENSIONAL)	NUMINT
F	R	C	POTENTIAL AT F (MAPPING CONSTANT)	NUMINT
FF	R	U	INTERMEDIATE CALCULATION VARIABLE	NUMINT
FF2	R	U	INTERMEDIATE CALCULATION VARIABLE	NUMINT
G	R	C	POTENTIAL AT G (MAPPING CONSTANT)	NUMINT
GAMMA	R	C	DIHEDRAL ANGLE IN DEGREES	NUMINT
GP	R	C	DIHEDRAL ANGLE DIVIDED BY PI	NUMINT
GPM2	R	C	GP - 0.5	NUMINT
GPP2	R	C	GP + 0.5	NUMINT
GT	R	U	CONSTANT OF REMOVABLE SINGULARITY	NUMINT
H	R	C	POTENTIAL AT H (MAPPING CONSTANT)	NUMINT
I	I	U	DO LOOP INDEX	NUMINT
ILAST	I	C	FLAG FOR LAST ITERATION	NUMINT
ININT	I	C	NUMBER OF ORDINATES IN NUMERICAL INTEGRALS	NUMINT
J	I	U	INTERMEDIATE CALCULATION VARIABLE	NUMINT
K	I	U	INTERMEDIATE CALCULATION VARIABLE	NUMINT

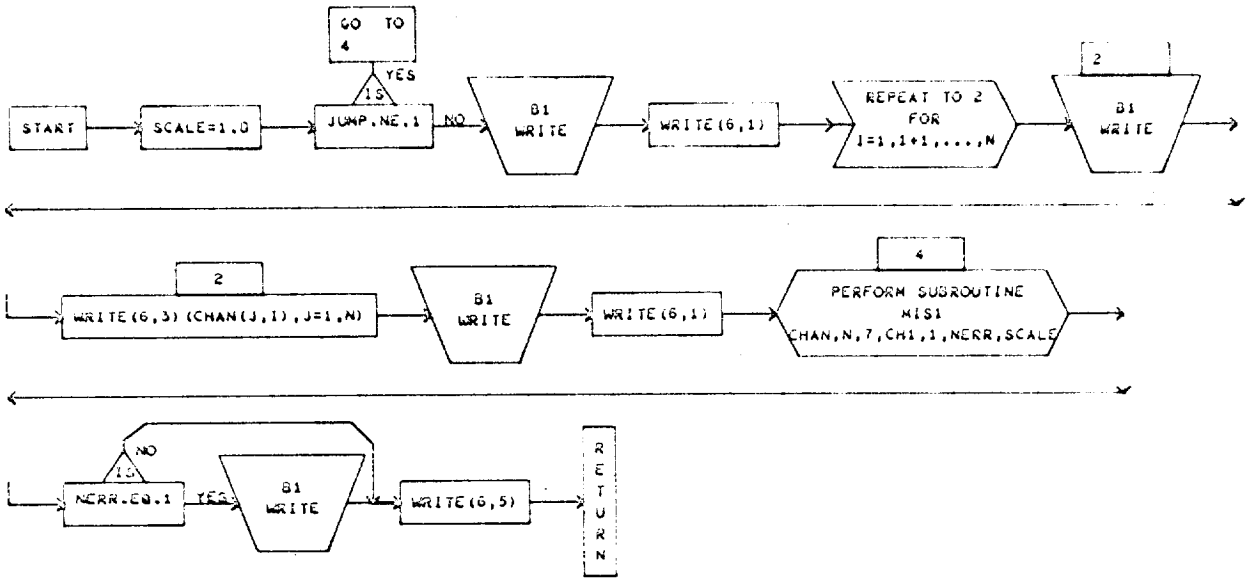
SYMBOLS USED IN SUBROUTINE NUMINT (DECK IDENTIFICATION 55V0)

MODE	I	C	PROGRAM MODE OF OPERATION	NUMINT
NDUMMY	I	U	UNUSED DUMMY VARIABLE	NUMINT
NPOINT	R	U	NUMBER OF ORDINATES IN NUMERICAL INTEGRAL	NUMINT
NTYPE	R	U	NUMERICAL INTEGRAL TYPE (INTEGRAL NUMBER)	NUMINT
OMGP	R	C	1.0 - GP	NUMINT
ORD	R	C	ORDINATE ARRAY FOR NUMERICAL INTEGRALS	NUMINT
PIREQ	R	C	INBOARD LENGTH OR DISTANCE (NONDIMENSIONAL)	NUMINT
POREQ	R	C	OUTBOARD LENGTH OR DISTANCE (NONDIMENSIONAL)	NUMINT
R1	R	C	SQUARE ROOT OF SUM OF SQUARES OF ERRORS IN REQD CONDITIONS	NUMINT
SING	R	U	POTENTIAL AT THE SINGULARITY	NUMINT
TAB	R	C	INTERPOLATION ARRAY FOR INDEPENDENT VARIABLE AND FUNCTION	NUMINT
TABLE	R	C	INTEGRATED VALUES OF ORDINATES AS A FUNCTION OF ABSCISSA	NUMINT
UP	R	U	POTENTIAL AT THE UPPER INTEGRATION LIMIT	NUMINT

Subroutine INVERT (JUMP, N) (Deck 5BV2). If a new influence matrix has been calculated (JUMP = 1), subroutine INVERT writes the new matrix just after it has been calculated. Subroutine INVERT then uses subroutine MISI to compute corrections to the current values of the mapping constants. An error message is written if the influence matrix is singular.

```
                SUBROUTINE INVERT (JUMP, N)                                     55VP0010
          C                                                                     55VP0020
          C     THIS SUBROUTINE PRINTS EACH NEW INFLUENCE MATRIX JUST AFTER IT IS 55VP0030
          C     CALCULATED, AND THEN CALLS THE STANDARD DOUGLAS SUBROUTINE THAT 55VP0040
          C     SOLVES LINEAR, SIMULTANEOUS EQUATIONS (SUBROUTINE MIS1).        55VP0050
          C                                                                     55VP0060
000004          COMMON DATA(17), ABSC(50), ORD(50), TABLE(50), ALOAD(1000), AA(16)55VP0070
               A , TAB(101), GP, GPM2, GPP2, CG, OMGP, CONR, CONI, ILAST, C1, C2 55VP0080
               B , C3, R1                                                       55VP0090
          C                                                                     55VP0100
000004          DIMENSION CHAN(7,7), CH1(7)                                     55VP0110
          C                                                                     55VP0120
000004          EQUIVALENCE (ALOAD(113), CHAN) , (ALOAD(163), CH1)              55VP0130
          C                                                                     55VP0140
000004          SCALE = 1.0                                                     55VP0150
000005          IF (JUMP .NE. 1) GO TO 4                                        55VP0160
000007          WRITE (6,1)                                                     55VP0170
000012        1 FORMAT (1H )                                                    55VP0180
000012          DO 2 I=1,N                                                      55VP0190
000015        2 WRITE (6,3) (CHAN(J,I),J=1,N)                                   55VP0200
000040        3 FORMAT (3X, 7F8.3)                                              55VP0210
000040          WRITE (6,1)                                                     55VP0220
000043        4 CALL MIS1 (CHAN, N, 7, CH1, 1, NERR, SCALE)                     55VP0230
000053          IF (NERR .EQ. 1) WRITE (6,5)                                    55VP0240
000063        5 FORMAT (1H0, 3X, 20H MATRIX IS SINGULAR  //)                    55VP0250
000063          RETURN                                                          55VP0260
000064          END                                                             55VP0270
```

```
SUBPROGRAM LENGTH
000115

FUNCTION ASSIGNMENTS

STATEMENT ASSIGNMENTS
1       -  000071    3       -  000073    4       -  000043    5       -  000100

BLOCK NAMES AND LENGTHS
        -  002420

VARIABLE ASSIGNMENTS
AA      -  002217C01 ABSC    -  000021C01 ALOAD   -  000247C01 CHAN    -  000427C01
CH1     -  000511C01 DATA    -  000000C01 I       -  000112    J       -  000113
NERR    -  000114    ORD     -  000103C01 SCALE   -  000111    TAB     -  002237C01
TABLE   -  000165C01

START OF CONSTANTS
000066

START OF TEMPORARIES
000105

START OF INDIRECTS
000111

UNUSED COMPILER SPACE
043388
```


GO TO 4
YES
IS
START
SCALE=1.0
JUMP.NE.1
NO
B1 WRITE
WRITE(6,1)
REPEAT TO 2 FOR I=1,I+1,...,N
2
B1 WRITE
2
WRITE(6,3) (CHAN(J,I),J=1,N)
B1 WRITE
WRITE(6,1)
4
PERFORM SUBROUTINE MIS1 CHAN,N,7,CH1,1,NERR,SCALE
NO
IS
NERR.EQ.1
YES
B1 WRITE
WRITE(6,5)
RETURN

SYMBOLS USED IN SUBROUTINE INVERT (DECK IDENTIFICATION 55V)

AA	R	C	ARRAY FOR RESULTS OF GEOMETRY AND DRAG INTEGRALS	INVERT
ABSC	R	C	ABSCISSA ARRAY FOR NUMERICAL INTEGRALS	INVERT
ALOAD	R	C	STORAGE ARRAY FOR POTENTIAL IN THE REAL (PHYSICAL) PLANE	INVERT
CG	R	C	COSINE OF DIHEDRAL ANGLE	INVERT
CHAN	R	C	INFLUENCE MATRIX DC-SUB-I / DP-SUB-J	INVERT
CHI	R	C	ERRORS IN REQUIRED CONDITIONS	INVERT
CONI	R	C	FACTOR FOR IMAGINARY PART OF COMPLEX DISTANCE	INVERT
CONR	R	C	FACTOR FOR REAL PART OF COMPLEX DISTANCE	INVERT
C1	R	C	CONSTANT THAT IS COMPUTER HARDWARE DEPENDENT	INVERT
C2	R	C	CONSTANT THAT IS COMPUTER HARDWARE DEPENDENT	INVERT
C3	R	C	CONSTANT THAT IS COMPUTER HARDWARE DEPENDENT	INVERT
DATA	R	C	ARRAY FOR INPUT DATA	INVERT
GP	R	C	DIHEDRAL ANGLE DIVIDED BY PI	INVERT
GPM2	R	C	GP - 0.5	INVERT
GPP2	R	C	GP + 0.5	INVERT
I	I	U	DC LOOP INDEX	INVERT
ILAST	I	C	FLAG FOR LAST ITERATION	INVERT
J	I	U	DC LOOP INDEX	INVERT
JUMP	I	U	FLAG TO PRINT OUT INFLUENCE MATRIX	INVERT
N	I	U	SIZE OF INFLUENCE MATRIX	INVERT
NERR	I	U	ERROR FLAG FOR SINGULAR MATRIX	INVERT
OMGP	R	C	1.0 - GP	INVERT
ORD	R	C	ORDINATE ARRAY FOR NUMERICAL INTEGRALS	INVERT
R1	R	C	SQUARE ROOT OF SUM OF SQUARES OF ERRORS IN REQD CONDITIONS	INVERT
SCALE	R	U	MATRIX SCALING FACTOR (SET TO ONE)	INVERT
TAB	R	C	INTERPOLATION ARRAY FOR INDEPENDENT VARIABLE AND FUNCTION	INVERT
TABLE	R	C	INTEGRATED VALUES OF ORDINATES AS A FUNCTION OF ABSCISSA	INVERT

Subroutine MIS1 (A,N, NDD, B, M, NERR, D) (Deck 55VQ). W. W. Wagner of the Scientific Computer Programming Group completed this subroutine on August 13, 1963 for the Douglas Aircraft Company's subroutine library. The matrix equation AC = B is solved for C. A is transformed to a unit matrix through row transformations that are applied to both A and B. This operation transforms B into C. The error flag NERR is 1 if A is singular and 0 if A is not singular.

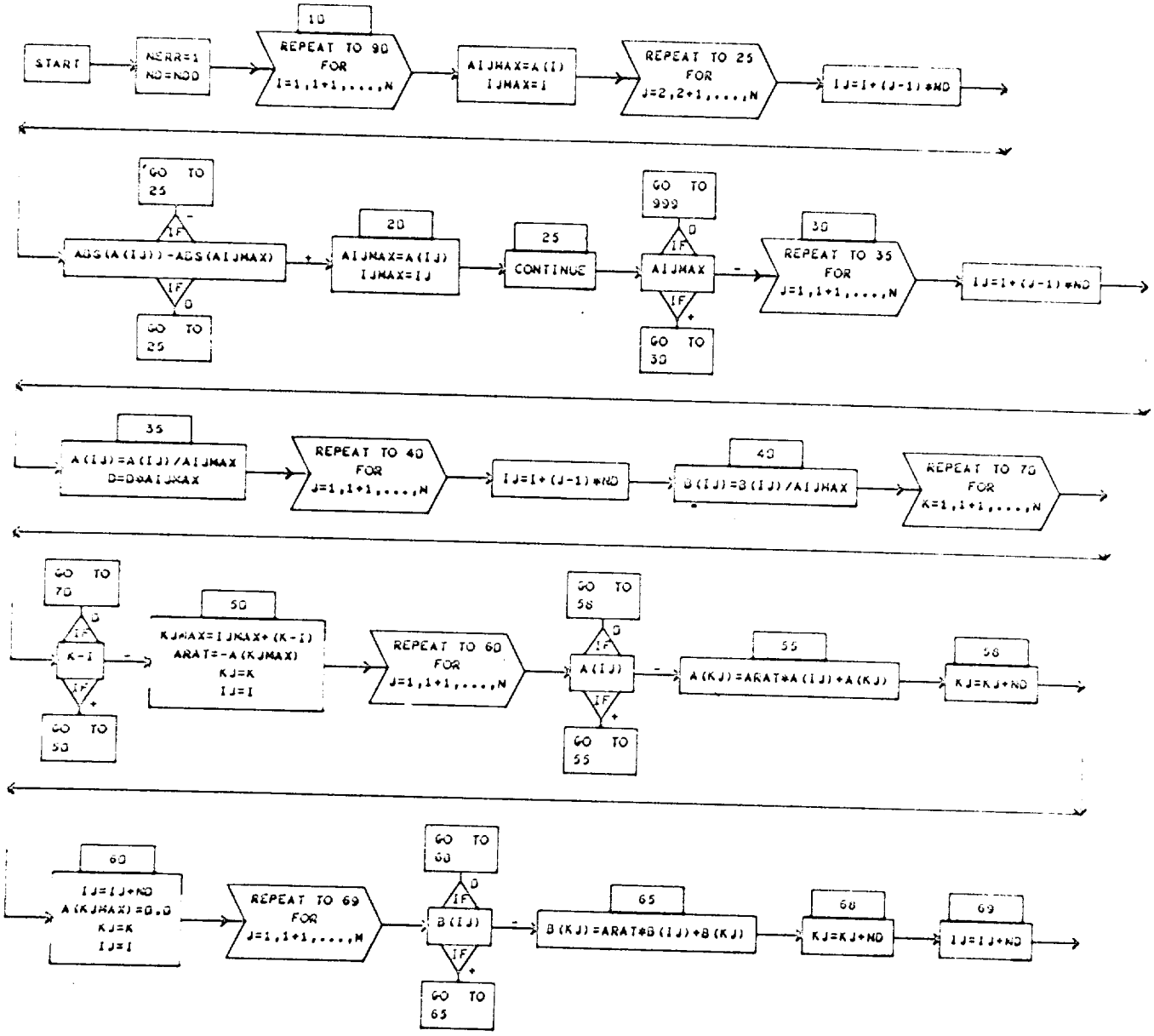
```
                SUBROUTINE MIS1 ( A, N, NDD, B, M, NERR, D    )
          C
          C     THIS SUBROUTINE WAS COMPLETED BY W. W. WAGNER ON 8/13/63 FOR THE
          C     DOUGLAS SUBROUTINE LIBRARY.
          C
          C     INVERT A MATRIX AND/OR SOLVE SIMULTANEOUS EQUATIONS
          C     INPUT,
          C     A = NAME OF INPUT MATRIX, DIMENSIONED A(ND,ND)
          C     B = NAME OF OUTPUT MATRIX,  DIMENSIONED B(ND,MD)
          C         ( B(ND) IS ALLOWED IF M=1 )
          C     D = SCALE FACTOR FOR VALUE OF DETERMINANT  (=1., FOR NO SCALING)
          C     OUTPUT,
          C     A(I,J) = GARBAGE
          C     B(I,J) = A-INVERSE * B
          C     D = SCALED VALUE OF DETERMINANT
          C     ERROR =  0--OK,  1--A IS SINGULAR
          C
000010          DIMENSION  A(1), B(1)
000010          EQUIVALENCE  (I,FI), (K,FK)
000010          NERR =  1
000011          ND = NDD
          C
          C     START REDUCTION OF MATRIX A
          C
000012       10 DO 90  I=1,N
          C
          C     SEARCH FOR MAXIMUM ELEMENT IN ITH ROW OF A-MATRIX
          C
000014          AIJMAX = A(I)
000016          IJMAX = I
000017          DO 25  J=2,N
000020          IJ = I + (J-1)*ND
000024          IF  ( ABS ( A(IJ) )  -  ABS  ( AIJMAX ) )   25,25,20
000032       20 AIJMAX = A(IJ)
000035          IJMAX = IJ
000036       25 CONTINUE
          C
          C     IF AIJMAX IS ZERO, THE MATRIX IS SINGULAR
          C
000041          IF  (AIJMAX)  30,999,30
          C
          C     NORMALIZE ITH ROW BY AIJMAX = IJMAX ELEMENT OF ITH ROW
          C
000042       30 DO 35  J=1,N
000044          IJ = I + (J-1)*ND
000050       35 A(IJ) = A(IJ)/AIJMAX
000055          D = D * AIJMAX
          C
          C     NORMALIZE ITH ROW OF B
          C
000056          DO 40  J=1,M
000060          IJ = I + (J-1)*ND
000064       40 B(IJ) = B(IJ)/AIJMAX
          C
          C     USE ROW TRANSFORMATIONS TO GET ZEROS ABOVE AND BELOW THE IJMAX
          C     ELEMENT OF THE ITH ROW OF A.  APPLY SAME ROW TRANSFORMATIONS
          C     TO THE B MATRIX.
```

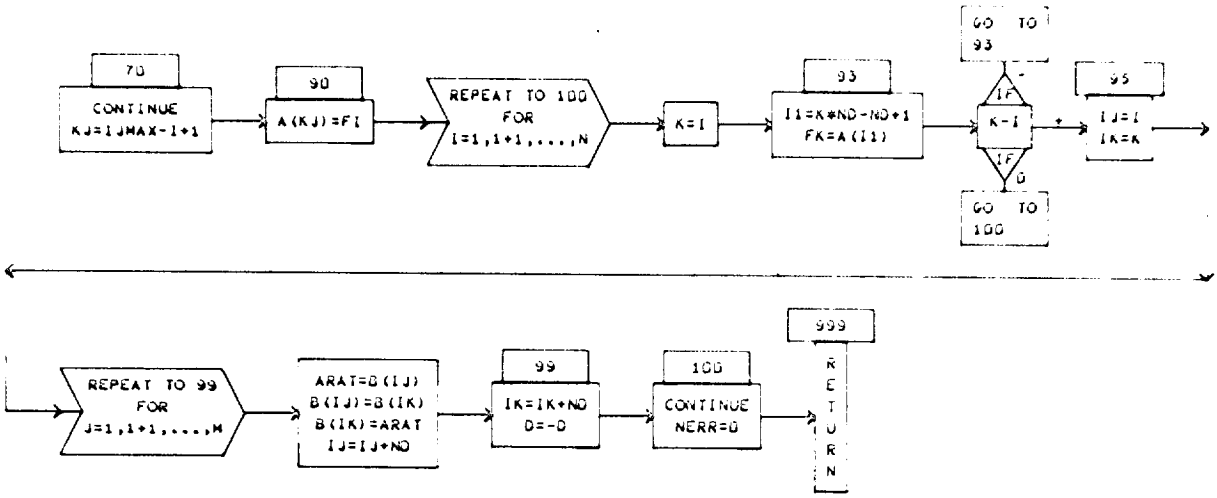
```
      C                                                                 55VQ0580
000071      DO 70  K=1,N                                                55VQ0590
000072      IF  (K-I)  50,70,50                                         55VQ0600
000074   50 KJMAX = IJMAX + (K-I)                                       55VQ0610
000077      ARAT  =  -A(KJMAX)                                          55VQ0620
000101      KJ = K                                                      55VQ0630
000103      IJ = I                                                      55VQ0640
000104      DO 60   J=1,N                                               55VQ0650
000106      IF  (A(IJ))  55,58,55                                       55VQ0660
000110   55 A(KJ) = ARAT*A(IJ) + A(KJ)                                  55VQ0670
000115   58 KJ = KJ + ND                                                55VQ0680
000117   60 IJ = IJ + ND                                                55VQ0690
000122      A(KJMAX) = 0.0                                              55VQ0700
000124      KJ = K                                                      55VQ0710
000125      IJ = I                                                      55VQ0720
000126      DO 69 J=1,M                                                 55VQ0730
000127      IF (B(IJ))  65,68,65                                        55VQ0740
000131   65 B(KJ) = ARAT*B(IJ) + B(KJ)                                  55VQ0750
000136   68 KJ = KJ + ND                                                55VQ0760
000140   69 IJ = IJ + ND                                                55VQ0770
000143   70 CONTINUE                                                    55VQ0780
000146      KJ = IJMAX - I+1                                            55VQ0790
      C                                                                 55VQ0800
      C     STORE ROW COUNTER (I) IN TOP ELEMENT OF IJMAX COLUMN.  THUS, 55VQ0810
      C     THE TOP ROW OF A WILL CONTAIN THE LOC OF THE PIVOT (UNITY)  55VQ0820
      C     ELEMENT OF EACH COLUMN (AFTER REDUCTION).                   55VQ0830
      C                                                                 55VQ0840
000150   90 A(KJ) = FI                                                  55VQ0850
      C                                                                 55VQ0860
      C                  THIS STORES INTEGER I IN TOP ROW OF A          55VQ0870
      C     THE REDUCTION OF A IS NOW COMPLETE.  PERFORM ROW INTERCHANGES 55VQ0880
      C     AS INDICATED IN THE FIRST ROW OF A.                         55VQ0890
      C                                                                 55VQ0900
000154      DO 100  I=1,N                                               55VQ0910
000156      K = I                                                       55VQ0920
000157   93 I1 = K*ND - ND + 1                                          55VQ0930
000163      FK = A(I1)                                                  55VQ0940
      C                                                                 55VQ0950
      C                  THIS PUTS THE INTEGER VALUE IN A INTO K        55VQ0960
      C                                                                 55VQ0970
000165      IF (K-I)  93,100,95                                         55VQ0980
      C                                                                 55VQ0990
      C     IF K=A(I1) IS LESS THAN I, THEN THAT ROW HAS ALREADY BEEN   55VQ1000
      C     INVOLVED IN AN INTERCHANGE, AND WE USE K=A(K) UNTIL WE GET  55VQ1010
      C     A VALUE OF K GREATER THAN I  (CORRESPONDING TO A ROW STORED 55VQ1020
      C     BELOW THE ITH ROW.  (CLEAR AS MUD)                          55VQ1030
      C                                                                 55VQ1040
000170   95 IJ = I                                                      55VQ1050
000172      IK = K                                                      55VQ1060
000173      DO 99  J=1,M                                                55VQ1070
000175      ARAT  =  B(IJ)                                              55VQ1080
000200      B(IJ) = B(IK)                                               55VQ1090
000203      B(IK)  =  ARAT                                              55VQ1100
000205      IJ = IJ + ND                                                55VQ1110
000206   99 IK = IK + ND                                                55VQ1120
      C                                                                 55VQ1130
      C     KEEP TRACK OF SIGN OF DETETMINANT                           55VQ1140
      C                                                                 55VQ1150
```

```
000211          D   =  -D                                          55VQ1160
000212      100 CONTINUE                                            55VQ1170
000215          NERR  =   0                                         55VQ1180
000215      999 RETURN                                              55VQ1190
000216          END                                                 55VQ1200
```

```
SUBPROGRAM LENGTH
000251

FUNCTION ASSIGNMENTS

STATEMENT ASSIGNMENTS
10      -  000012    20      -  000032    25      -  000036    30      -  000042
50      -  000074    55      -  000110    58      -  000115    65      -  000131
68      -  000136    70      -  000143    93      -  000157    95      -  000170
100     -  000212    999     -  000215

BLOCK NAMES AND LENGTHS

VARIABLE ASSIGNMENTS
AIJMAX  -  000240    ARAT    -  000245    D       -  000000    FI      -  000235
FK      -  000236    I       -  000235    IJ      -  000243    IJMAX   -  000241
IK      -  000250    I1      -  000247    J       -  000242    K       -  000236
KJ      -  000246    KJMAX   -  000244    ND      -  000237

START OF CONSTANTS
000220

START OF TEMPORARIES
000221

START OF INDIRECTS
000231

UNUSED COMPILER SPACE
042600
```

SUBROUTINE MIS1 (A, N, NDD, B, M, NERR, D)




70
CONTINUE
KJ=IJMAX-I+1
90
A(KJ)=FI
REPEAT TO 100
FOR
I=1,I+1,...,N
K=I
93
I1=K*ND-ND+1
FK=A(I1)
GO TO
93
IF
K-I
-
+
IF
0
GO TO
100
95
IJ=I
IK=K
REPEAT TO 99
FOR
J=1,J+1,...,M
ARAT=B(IJ)
B(IJ)=B(IK)
B(IK)=ARAT
IJ=IJ+ND
99
IK=IK+ND
D=-D
100
CONTINUE
NERR=0
999
RETURN

SYMBOLS USED IN SUBROUTINE MIS1 (DECK IDENTIFICATION 55V0)

A	R	D	INPUT MATRIX	
AIJMAX	R	U	MAXIMUM OF AIJ	MIS1
ARAT	R	U	CALCULATION VARIABLE	MIS1
B	R	D	OUTPUT MATRIX	MIS1
D	R	U	SCALE FACTOR FOR DETERMINANT OF MATRIX	MIS1
FI	R	U	CALCULATION VARIABLE	MIS1
FK	R	U	CALCULATION VARIABLE	MIS1
I	I	U	DO LOOP INDEX	MIS1
IJ	I	U	CALCULATION VARIABLE	MIS1
IJMAX	I	U	IJ FOR MAXIMUM OF AIJ	MIS1
IK	I	U	CALCULATION VARIABLE	MIS1
I1	I	U	CALCULATION VARIABLE	MIS1
J	I	U	DO LOOP INDEX	MIS1
K	I	U	CALCULATION VARIABLE	MIS1
KJ	I	U	CALCULATION VARIABLE	MIS1
KJMAX	I	U	CALCULATION VARIABLE	MIS1
M	I	U	NUMBER OF COLUMNS IN OUTPUT MATRIX	MIS1
N	I	U	ORDER OF MATRIX A	MIS1
ND	I	U	CALCULATION VARIABLE	MIS1
NDD	I	U	MAXIMUM ORDER OF MATRIX A	MIS1
NERR	I	U	ERROR FLAG FOR SINGULAR MATRIX	MIS1
				MIS1

Subroutine COMPAT (Deck 55VR). The mapping constants (or potentials) are forced to monotonically increasing status by this subroutine according to the algorithim:

$$\text{If} \quad p_{j+1} > p_j \ , \quad p_{j+1} = p_{j+1} \tag{12}$$

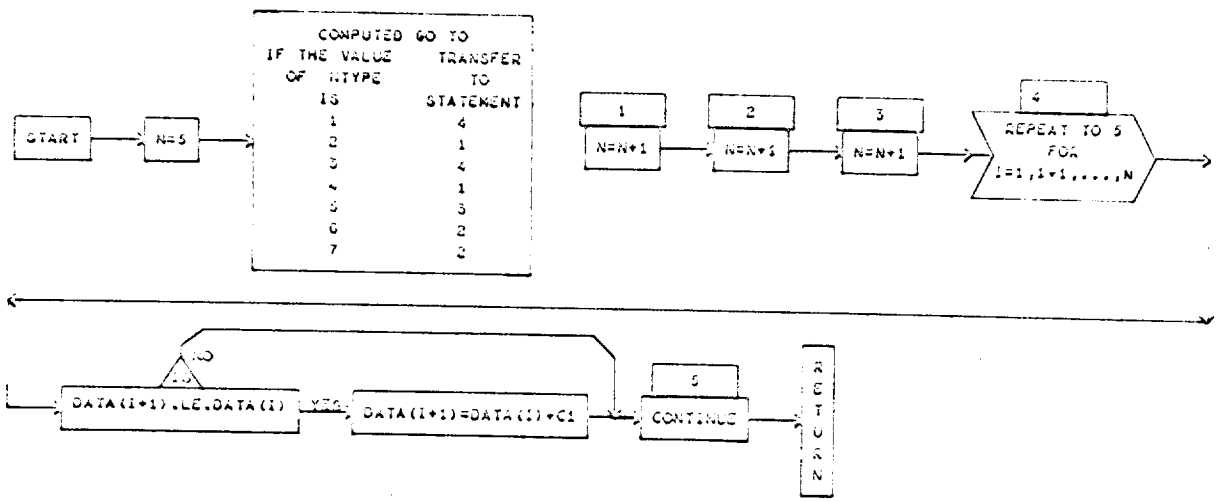
$$\text{If} \quad p_{j+1} \leq p_j \ , \quad p_{j+1} = p_j + \text{constant} \tag{13}$$

The value of the constant is computing machine dependent and is very small.

```
      SUBROUTINE COMPAT
C
C     THIS SUBROUTINE ASSURES THAT THE ASSUMED POTENTIALS ARE MONOTONE
C     INCREASING AS REQUIRED BY THE THEORY.
000002
      COMMON DATA(17), ADSC(50), ORD(50), TABLE(50), ALOAD(1000), AA(16)
     A , TAB(101), GP, GPM2, GPP2, CG, OAGP, CONR, CONI, ILAST, C1, C2
     B , C3, R1
C
000002
      EQUIVALENCE
     A , (DATA( 3),       C) , (DATA( 1),       A) , (DATA( 2),       B)
     B , (DATA( 6),       F) , (DATA( 4),       D) , (DATA( 5),       E)
     C , (DATA( 9),      AI) , (DATA( 7),       G) , (DATA( 8),       H)
     D , (DATA(12),   ININT) , (DATA(10),    MODE) , (DATA(11),   NTYPE)
     E , (DATA(15),   PIREQ) , (DATA(13),  ETAIRQ) , (DATA(14),  ETACRQ)
C                            , (DATA(16),  POREQ) , (DATA(17),   GAMMA)
000002
000003
000010
000020
000022
000024
000026
000036
000040
000040
      N = 5
      GO TO ( 4, 1, 4, 1, 3, 2, 2) , NTYPE
    1 N = N + 1
    2 N = N + 1
    3 N = N + 1
    4 DO 5 I=1,N
      IF (DATA(I+1) .LE. DATA(I)) DATA(I+1) = DATA(I) + C1
    5 CONTINUE
      RETURN
      END
```

```
SUBPROGRAM LENGTH
000053

FUNCTION ASSIGNMENTS

STATEMENT ASSIGNMENTS
1       -  000016    2       -  000020    3       -  000022    4       -  000024
5       -  000035

BLOCK NAMES AND LENGTHS
        -  002420

VARIABLE ASSIGNMENTS
AA      -  002217C01 ACC     -  000021C01 ALOAD   -  000247C01 C1      -  002414C01
DATA    -  000000C01 I       -  000052    N       -  000051    NTYPE   -  000012C01
ORD     -  000103C01 TAB     -  002237C01 TABLE   -  000165C01

START OF CONSTANTS
000042

START OF TEMPORARIES
000043

START OF INDIRECTS
000045

UNUSED COMPILER SPACE
043300
```

SUBROUTINE COMPAT



SYMBOLS USED IN SUBROUTINE COMPAT (DECK IDENTIFICATION 99V.)

A	R	C	POTENTIAL AT A (MAPPING CONSTANT)	COMPAT
AA	R	C	ARRAY FOR RESULTS OF GEOMETRY AND DRAG INTEGRALS	COMPAT
ABSC	R	C	ABSCISSA ARRAY FOR NUMERICAL INTEGRALS	COMPAT
AI	R	C	POTENTIAL AT I (MAPPING CONSTANT)	COMPAT
ALOAD	R	C	STORAGE ARRAY FOR POTENTIAL IN THE REAL (PHYSICAL) PLANE	COMPAT
B	R	C	POTENTIAL AT B (MAPPING CONSTANT)	COMPAT
C	R	C	POTENTIAL AT C (MAPPING CONSTANT)	COMPAT
CG	R	C	COSINE OF DIHEDRAL ANGLE	COMPAT
CONI	R	C	FACTOR FOR IMAGINARY PART OF COMPLEX DISTANCE	COMPAT
CONR	R	C	FACTOR FOR REAL PART OF COMPLEX DISTANCE	COMPAT
C1	R	C	CONSTANT THAT IS COMPUTER HARDWARE DEPENDENT	COMPAT
C2	R	C	CONSTANT THAT IS COMPUTER HARDWARE DEPENDENT	COMPAT
C3	R	C	CONSTANT THAT IS COMPUTER HARDWARE DEPENDENT	COMPAT
D	R	C	POTENTIAL AT D (MAPPING CONSTANT)	COMPAT
DATA	R	C	ARRAY FOR INPUT DATA	COMPAT
E	R	C	POTENTIAL AT E (MAPPING CONSTANT)	COMPAT
ETAIRC	R	C	INBOARD SPANWISE LOCATION (NONDIMENSIONAL)	COMPAT
ETAORC	R	C	OUTBOARD SPANWISE LOCATION (NONDIMENSIONAL)	COMPAT
F	R	C	POTENTIAL AT F (MAPPING CONSTANT)	COMPAT
G	R	C	POTENTIAL AT G (MAPPING CONSTANT)	COMPAT
GAMMA	R	C	DIHEDRAL ANGLE IN DEGREES	COMPAT
GP	R	C	DIHEDRAL ANGLE DIVIDED BY PI	COMPAT
GPM2	R	C	GP - 0.5	COMPAT
GPP2	R	C	GP + 0.5	COMPAT
H	R	C	POTENTIAL AT H (MAPPING CONSTANT)	COMPAT
ILAST	I	C	FLAG FOR LAST ITERATION	COMPAT
ININT	I	C	NUMBER OF ORDINATES IN NUMERICAL INTEGRALS	COMPAT
MODE	I	C	PROGRAM MODE OF OPERATION	COMPAT
N	I	U	NUMBER OF POTENTIALS (MAPPING CONSTANTS)	COMPAT
NTYPE	I	C	CONFIGURATION TYPE	COMPAT
OMGP	R	C	1.0 - GP	COMPAT
ORD	R	C	ORDINATE ARRAY FOR NUMERICAL INTEGRALS	COMPAT
PIREQ	R	C	INBOARD LENGTH OR DISTANCE (NONDIMENSIONAL)	COMPAT
POREQ	R	C	OUTBOARD LENGTH OR DISTANCE (NONDIMENSIONAL)	COMPAT
R1	R	C	SQUARE ROOT OF SUM OF SQUARES OF ERRORS IN REQD CONDITIONS	COMPAT
TAB	R	C	INTERPOLATION ARRAY FOR INDEPENDENT VARIABLE AND FUNCTION	COMPAT
TABLE	R	C	INTEGRATED VALUES OF ORDINATES AS A FUNCTION OF ABSCISSA	COMPAT

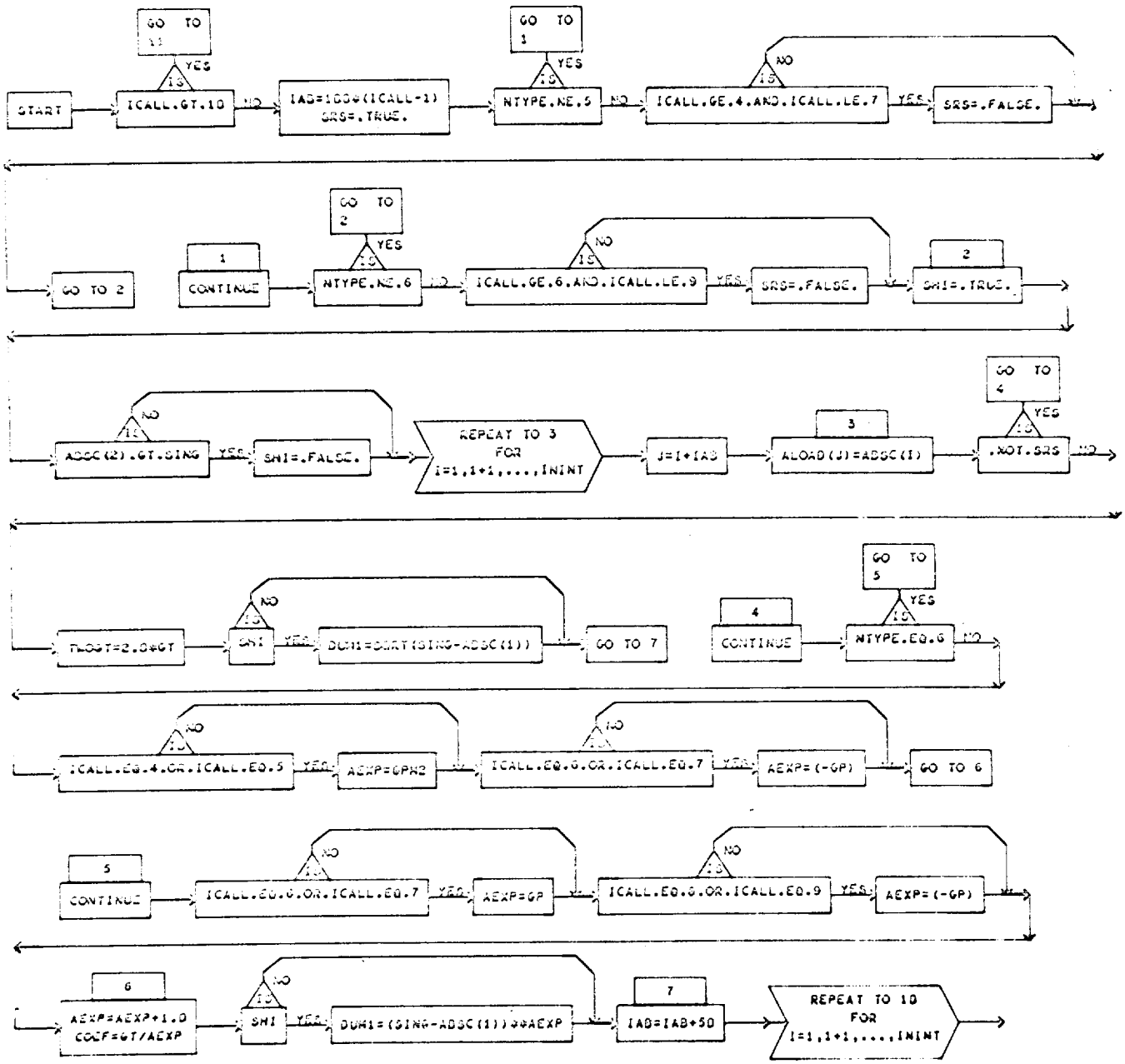
Subroutine STORE (GT, STNG, ICALL) (Deck 55VS). The tabular results from the numerical integration for distances in the real plane are transformed, except for sign and scale zero, into tables of potential versus coordinate in the real plane. A term analogous to the second term on the right side of Equation (6) is added to each of the ordinates before they are stored in the ALOAD array. The abscissae are stored in ALOAD without change. The first hundred locations in the ALOAD array are assigned to the tabular results for the first numerical integration, the second hundred to results for the second numerical integration, etc.

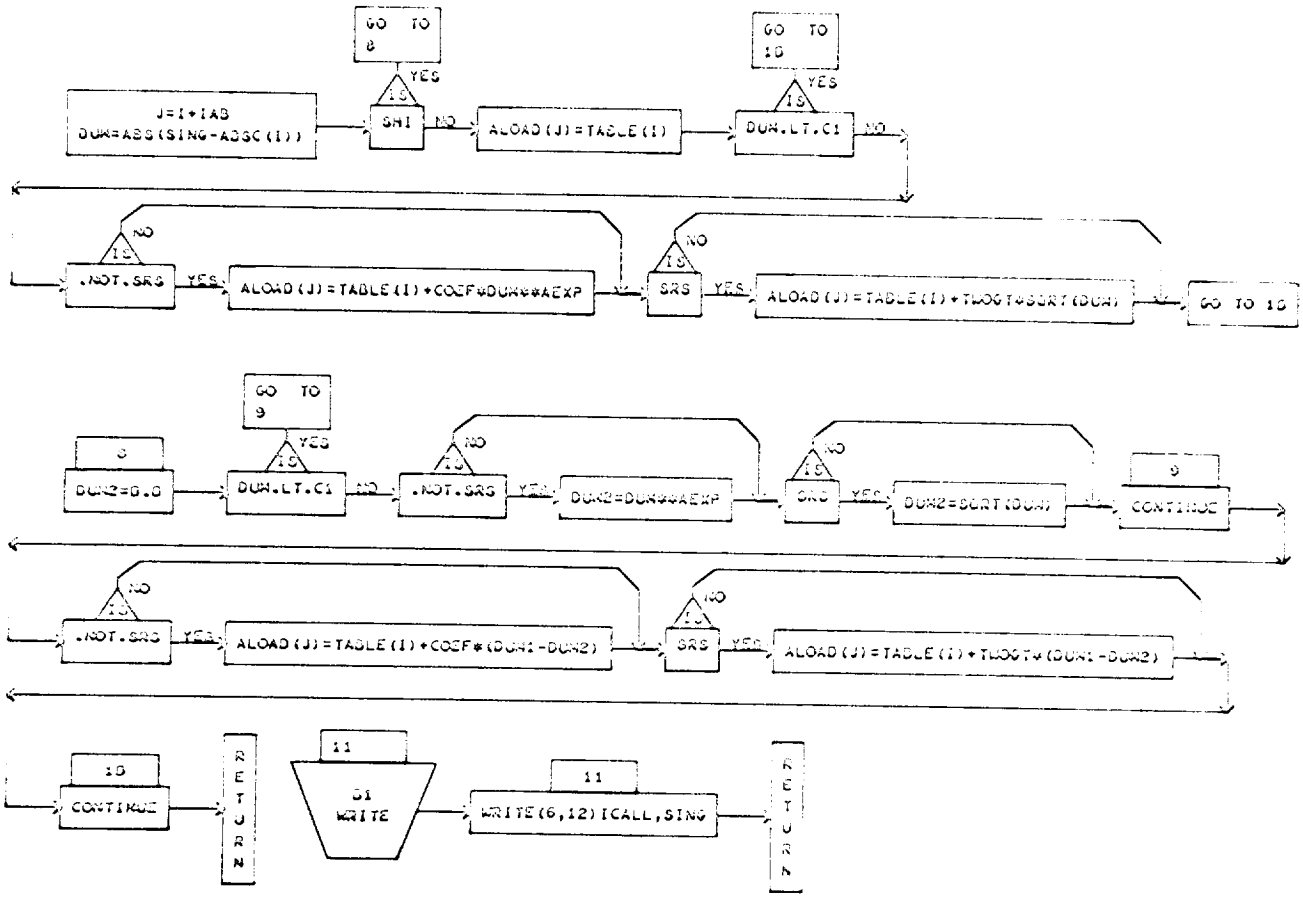
```
                SUBROUTINE STORE (GT,SING,ICALL)
          C
          C     THIS SUBROUTINE STORES ZETA (POTENTIAL) VERSUS ABSCISSA (Z-COORD-
          C     INATE) IN THE ALOAD ARRAY SO THAT LOAD DISTRIBUTION CAN BE CALCU-
          C     LATED FROM THE DATA IN THE ALOAD ARRAY ONLY
          C
000005          COMMON DATA(17), ABSC(50), ORD(50), TABLE(50), ALOAD(1000), AA(16)
               A , TAB(101), GP, GPM2, GPP2, CG, OMGP, CONR, CONI, ILAST, C1, C2
               B , C3, R1
          C
000005          EQUIVALENCE              (DATA( 1),       A) , (DATA( 2),       B)
               A , (DATA( 3),       C) , (DATA( 4),       D) , (DATA( 5),       E)
               B , (DATA( 6),       F) , (DATA( 7),       G) , (DATA( 8),       H)
               C , (DATA( 9),      AI) , (DATA(10),    MODE) , (DATA(11),  NTYPE)
               D , (DATA(12),  ININT) , (DATA(13), ETAIRO) , (DATA(14), ETAORO)
               E , (DATA(15),  PIREQ) , (DATA(16),  POREQ) , (DATA(17),  GAMMA)
          C
000005          LOGICAL SRS, SHI
          C
000005          IF (ICALL .GT. 10) GO TO 11
000010          IAB = 100 * (ICALL - 1)
000012          SRS = .TRUE.
000013          IF (NTYPE .NE. 5) GO TO 1
000015          IF (ICALL .GE. 4 .AND. ICALL .LE. 7) SRS = .FALSE.
000027          GO TO 2
000030        1 IF (NTYPE .NE. 6) GO TO 2
000032          IF (ICALL .GE. 6 .AND. ICALL .LE. 9) SRS = .FALSE.
000043        2 SHI = .TRUE.
000044          IF (ABSC(2) .GT. SING) SHI = .FALSE.
000050          DO 3 I=1,ININT
000052          J = I + IAB
000054        3 ALOAD(J) = ABSC(I)
000061          IF (.NOT. SRS) GO TO 4
000062          TWOGT = 2.0 * GT
000064          IF (SHI) DUM1 = SQRT(SING - ABSC(1))
000073          GO TO 7
000074        4 IF (NTYPE .EQ. 6) GO TO 5
000076          IF (ICALL .EQ. 4 .OR. ICALL .EQ. 5) AEXP = GPM2
000106          IF (ICALL .EQ. 6 .OR. ICALL .EQ. 7) AEXP = (-GP)
000116          GO TO 6
000117        5 IF (ICALL .EQ. 6 .OR. ICALL .EQ. 7) AEXP = GP
000127          IF (ICALL .EQ. 8 .OR. ICALL .EQ. 9) AEXP = (-GP)
000137        6 AEXP = AEXP + 1.0
000141          COEF = GT / AEXP
000142          IF (SHI) DUM1 = (SING - ABSC(1)) ** AEXP
000150        7 IAB = IAB + 50
          C
          C     ENTER CALCULATION AND STORAGE LOOP
          C
000152          DO 10 I=1,ININT
000153          J = I + IAB
000155          DUM = ABS(SING - ABSC(I))
000160          IF (SHI) GO TO 8
000161          ALOAD(J) = TABLE(I)
000165          IF (DUM .LT. C1) GO TO 10
000167          IF (.NOT. SRS) ALOAD(J) = TABLE(I) + COEF * DUM ** AEXP
000201          IF (       SRS) ALOAD(J) = TABLE(I) + TWOGT * SQRT(DUM)
```

```
000213          GO TO 10
000214        6 DUM2 = 0.0
000215          IF (DUM .LT. C1) GO TO 9                                    55VS0580
000220          IF (.NOT. SRS) DUM2 = DUM ** AEXP                           55VS0590
000225          IF (       SRS) DUM2 = SQRT(DUM)                            55VS0600
000230        9 IF (.NOT. SRS) ALOAD(J) = TABLE(I) * COEF * (DUM1 - DUM2)   55VS0610
000243          IF (       SRS) ALOAD(J) = TABLE(I) * TWOGT * (DUM1 - DUM2) 55VS0620
000255       10 CONTINUE                                                    55VS0630
000256          RETURN                                                      55VS0640
000256       11 WRITE (6,12) ICALL, SING                                    55VS0650
000266       12 FORMAT (1H0, 2X, 31H *** SUBROUTINE STORE  ICALL =  ,I7,5X,F20.8/)55VS0660
000266          RETURN                                                      55VS0670
000267          END                                                         55VS0680
                                                                            55VS0690
                                                                            55VS0700
```

```
SUBPROGRAM LENGTH
000337

FUNCTION ASSIGNMENTS

STATEMENT ASSIGNMENTS
1       -  000050    3       -  000043    4       -  000074    5       -  000117
6       -  000157    7       -  000150    8       -  000214    9       -  000233
10      -  000253    11      -  000256    12      -  000275

BLOCK NAMES AND LENGTHS
        -  002420

VARIABLE ASSIGNMENTS
AA      -  002217C01 ARSC    -  000021C01 AEXP    -  000333    ALOAD   -  000047C01
COEF    -  000334    C1      -  000414C01 DATA    -  000300C01 DOM     -  000333
DOM1    -  000332    DOM2    -  000330    EP      -  000404C01 GPA2    -  002400C01
I       -  000327    IAB     -  000326    ININT   -  000013C01 J       -  000330
MTYPE   -  000012C01 ORD     -  000103C01 SAI     -  000325    SAS     -  000324
TAB     -  002237C01 TABLE   -  000165C01 TWOGT   -  000331

START OF CONSTANTS
000271

START OF TEMPORARIES
000305

START OF INDIRECTS
000315

UNUSED COMPILER SPACE
042300
```

SUBROUTINE STORE (GT,SING,ICALL)




GO TO 8
YES
J=I+IAB
DUM=ABS(SING-ABSC(I))
IS
SNI
NO
ALOAD(J)=TABLE(I)
GO TO 10
YES
IS
DUM.LT.C1
NO
NO
IS
.NOT.SRS
YES
ALOAD(J)=TABLE(I)+COEF*DUM**AEXP
NO
IS
SRS
YES
ALOAD(J)=TABLE(I)+TLOGT*SQRT(DUM)
GO TO 10
GO TO 9
YES
8
DUM2=0.0
IS
DUM.LT.C1
NO
NO
IS
.NOT.SRS
YES
DUM2=DUM**AEXP
NO
IS
SRS
YES
DUM2=SQRT(DUM)
9
CONTINUE
NO
IS
.NOT.SRS
YES
ALOAD(J)=TABLE(I)+COEF*(DUM1-DUM2)
NO
IS
SRS
YES
ALOAD(J)=TABLE(I)+TLOGT*(DUM1-DUM2)
10
CONTINUE
RETURN
11
DI
WRITE
11
WRITE(6,12)ICALL,SING
RETURN

SYMBOLS USED IN SUBROUTINE STORE (DECK IDENTIFICATION 5LVS)

A	R	C	POTENTIAL AT A (MAPPING CONSTANT)	STORE
AA	R	C	ARRAY FOR RESULTS OF GEOMETRY AND DRAG INTEGRALS	STORE
ABSC	R	C	ABSCISSA ARRAY FOR NUMERICAL INTEGRALS	STORE
AEXP	R	U	EXPONENT FOR ORDINATE CALCULATION	STORE
AI	R	C	POTENTIAL AT I (MAPPING CONSTANT)	STORE
ALOAD	R	C	STORAGE ARRAY FOR POTENTIAL IN THE REAL (PHYSICAL) PLANE	STORE
B	R	C	POTENTIAL AT B (MAPPING CONSTANT)	STORE
C	R	C	POTENTIAL AT C (MAPPING CONSTANT)	STORE
CG	R	C	COSINE OF DIHEDRAL ANGLE	STORE
COEF	R	U	CALCULATION VARIABLE	STORE
CONI	R	C	FACTOR FOR IMAGINARY PART OF COMPLEX DISTANCE	STORE
CONR	R	C	FACTOR FOR REAL PART OF COMPLEX DISTANCE	STORE
C1	R	C	CONSTANT THAT IS COMPUTER HARDWARE DEPENDENT	STORE
C2	R	C	CONSTANT THAT IS COMPUTER HARDWARE DEPENDENT	STORE
C3	R	C	CONSTANT THAT IS COMPUTER HARDWARE DEPENDENT	STORE
D	R	C	POTENTIAL AT D (MAPPING CONSTANT)	STORE
DATA	R	C	ARRAY FOR INPUT DATA	STORE
DUM	R	U	CALCULATION VARIABLE	STORE
DUM1	R	U	CALCULATION VARIABLE	STORE
DUM2	R	U	CALCULATION VARIABLE	STORE
E	R	C	POTENTIAL AT E (MAPPING CONSTANT)	STORE
ETAIRQ	R	C	INBOARD SPANWISE LOCATION (NONDIMENSIONAL)	STORE
ETAORG	R	C	OUTBOARD SPANWISE LOCATION (NONDIMENSIONAL)	STORE
F	R	C	POTENTIAL AT F (MAPPING CONSTANT)	STORE
G	R	C	POTENTIAL AT G (MAPPING CONSTANT)	STORE
GAMMA	R	C	DIHEDRAL ANGLE IN DEGREES	STORE
GP	R	C	DIHEDRAL ANGLE DIVIDED BY PI	STORE
GPM2	R	C	GP - 0.5	STORE
GPP2	R	C	GP + 0.5	STORE
GT	R	U	CONSTANT OF REMOVABLE SINGULARITY	STORE
H	R	C	POTENTIAL AT H (MAPPING CONSTANT)	STORE
I	I	U	DO LOOP INDEX	STORE
IAB	I	U	CALCULATION VARIABLE	STORE
ICALL	I	U	REFERENCE NUMBER FOR NUMERICAL INTEGRAL	STORE
ILAST	I	C	FLAG FOR LAST ITERATION	STORE
ININT	I	C	NUMBER OF ORDINATES IN NUMERICAL INTEGRALS	STORE
J	I	U	CALCULATION VARIABLE	STORE
MODE	I	C	PROGRAM MODE OF OPERATION	STORE
				STORE

SYMBOLS USED IN SUBROUTINE STORE (DECK IDENTIFICATION SSVS)

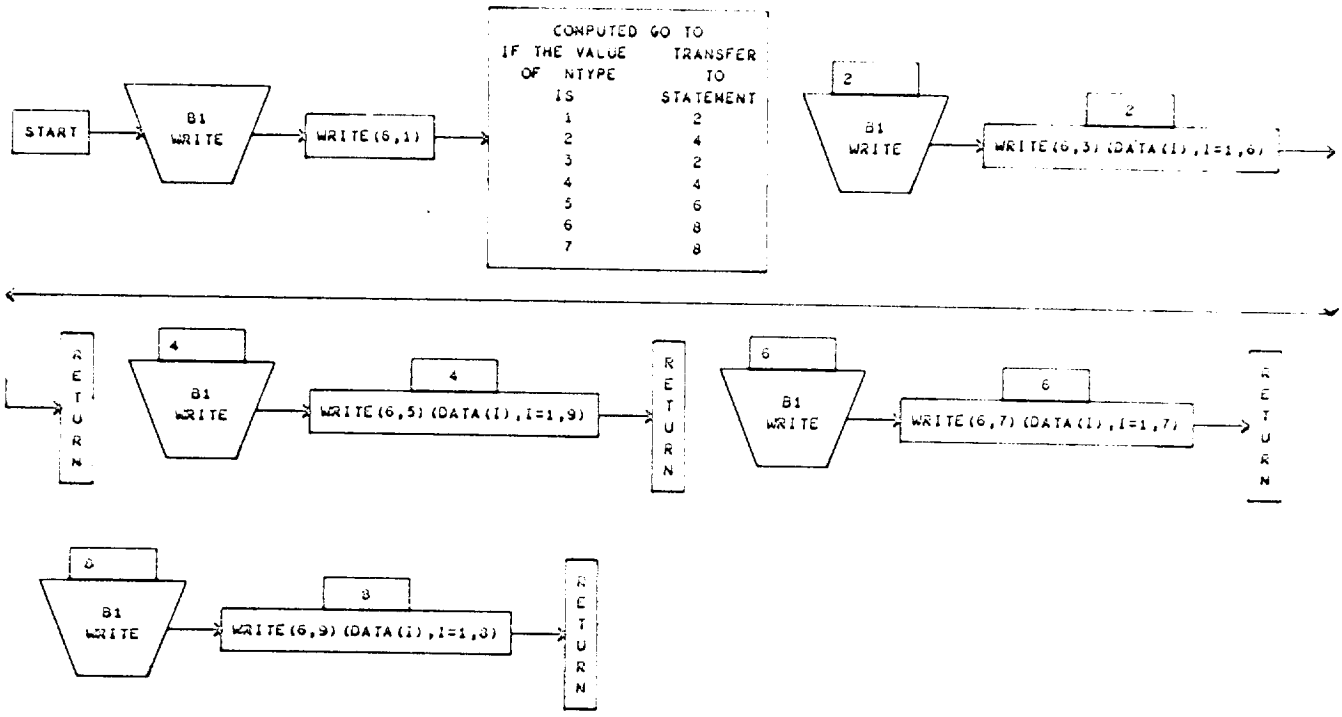
NTYPE	I	C	CONFIGURATION TYPE	STORE
CMGP	R	C	1.0 - GP	STORE
ORD	R	C	ORDINATE ARRAY FOR NUMERICAL INTEGRALS	STORE
PIREQ	R	C	INBOARD LENGTH OR DISTANCE (NONDIMENSIONAL)	STORE
POREQ	R	C	OUTBOARD LENGTH OR DISTANCE (NONDIMENSIONAL)	STORE
R1	R	C	SQUARE ROOT OF SUM OF SQUARES OF ERRORS IN REQD CONDITIONS	STORE
SHI	L	U	LOGICAL FLAG FOR SINGULARITY AT UPPER INTEGRATION LIMIT	STORE
SING	R	U	POTENTIAL AT THE SINGULARITY	STORE
SRS	L	U	LOGICAL FLAG FOR SQUARE ROOT SINGULARITY	STORE
TAB	R	C	INTERPOLATION ARRAY FOR INDEPENDENT VARIABLE AND FUNCTION	STORE
TABLI	R	C	INTEGRATED VALUES OF ORDINATES AS A FUNCTION OF ABSCISSA	STORE
TLOGT	R	U	TWICE THE CONSTANT OF THE REMOVABLE SINGULARITY	STORE

Subroutine POTOUT (Deck 55VT). The potentials (mapping constants) are written in exponential format by this subroutine.

```
                SUBROUTINE POTOUT
          C
          C     THIS SUBROUTINE WRITES OUT THE CONSTANTS OF THE SCHWARZ-
          C     CHRISTOFFEL TRANSFORMATION IN EXPONENTIAL FORMAT
          C
000002          COMMON DATA(17), ABSC(50), ORD(50), TABLE(50), ALOAD(1000), AA(16)
               A , TAB(101), GP, GPM2, GPP2, CG, OMGP, CONR, CONI, ILAST, C1, C2
               B , C3, R1
          C
000002          EQUIVALENCE                (DATA( 1),       A) , (DATA( 2),       B)
               A , (DATA( 3),       C) , (DATA( 4),       D) , (DATA( 5),       E)
               B , (DATA( 6),       F) , (DATA( 7),       G) , (DATA( 8),       H)
               C , (DATA( 9),      AI) , (DATA(10),    MODE) , (DATA(11),   NTYPE)
               D , (DATA(12),   ININT) , (DATA(13), ETAIRQ) , (DATA(14), ETAORQ)
               E , (DATA(15),   PIREQ) , (DATA(16),  POREQ) , (DATA(17),   GAMMA)
          C
000002          WRITE (6,1)
000006        1 FORMAT (1H0, 37X, 40H TRANSFORMATION CONSTANTS IN EXPONENTIAL FORM
               AAT //)
000006          GO TO (2, 4, 2, 4, 6, 8, 8) , NTYPE
000021        2 WRITE (6,3) (DATA(I),I=1,6)
000033        3 FORMAT (1H , 4X, 6E20.8 //)
000033          RETURN
000034        4 WRITE (6,5) (DATA(I),I=1,9)
000046        5 FORMAT (1H , 3E16.8, 2( E9.1, 2E16.8) //)
000046          RETURN
000047        6 WRITE (6,7) (DATA(I),I=1,7)
000061        7 FORMAT (1H , 1X, 7E18.8 //)
000061          RETURN
000062        8 WRITE (6,9) (DATA(I),I=1,8)
000074        9 FORMAT (1H , 1X, 8E16.8 //)
000074          RETURN
000075          END
```

```
SUBPROGRAM LENGTH
000136

FUNCTION ASSIGNMENTS

STATEMENT ASSIGNMENTS
1       -  000101    2       -  000021    3       -  000111    4       -  000034
5       -  000115    6       -  000047    7       -  000122    8       -  000062
9       -  000126

BLOCK NAMES AND LENGTHS
        -  002420

VARIABLE ASSIGNMENTS
AA      -  002217C01 ABSC    -  000021C01 ALOAD   -  000247C01 DATA    -  000000C01
I       -  000135    NTYPE   -  000012C01 ORD     -  000103C01 TAB     -  002237C01
TABLE   -  000165C01

START OF CONSTANTS
000077

START OF TEMPORARIES
000132

START OF INDIRECTS
000135

UNUSED COMPILER SPACE
043100
```

SUBROUTINE POTOUT



SYMBOLS USED IN SUBROUTINE POTOUT (DECK IDENTIFICATION 55[illegible]T)

A	R	C	POTENTIAL AT A (MAPPING CONSTANT)	POTOUT
AA	R	C	ARRAY FOR RESULTS OF GEOMETRY AND DRAG INTEGRALS	POTOUT
ABSC	R	C	ABSCISSA ARRAY FOR NUMERICAL INTEGRALS	POTOUT
AI	R	C	POTENTIAL AT I (MAPPING CONSTANT)	POTOUT
ALOAD	R	C	STORAGE ARRAY FOR POTENTIAL IN THE REAL (PHYSICAL) PLANE	POTOUT
B	R	C	POTENTIAL AT B (MAPPING CONSTANT)	POTOUT
C	R	C	POTENTIAL AT C (MAPPING CONSTANT)	POTOUT
CG	R	C	COSINE OF DIHEDRAL ANGLE	POTOUT
CONI	R	C	FACTOR FOR IMAJINARY PART OF COMPLEX DISTANCE	POTOUT
CONR	R	C	FACTOR FOR REAL PART OF COMPLEX DISTANCE	POTOUT
C1	R	C	CONSTANT THAT IS COMPUTER HARDWARE DEPENDENT	POTOUT
C2	R	C	CONSTANT THAT IS COMPUTER HARDWARE DEPENDENT	POTOUT
C3	R	C	CONSTANT THAT IS COMPUTER HARDWARE DEPENDENT	POTOUT
D	R	C	POTENTIAL AT D (MAPPING CONSTANT)	POTOUT
DATA	R	C	ARRAY FOR INPUT DATA	POTOUT
E	R	C	POTENTIAL AT E (MAPPING CONSTANT)	POTOUT
ETAIRD	R	C	INBOARD SPANWISE LOCATION (NONDIMENSIONAL)	POTOUT
ETAORD	R	C	OUTBOARD SPANWISE LOCATION (NONDIMENSIONAL)	POTOUT
F	R	C	POTENTIAL AT F (MAPPING CONSTANT)	POTOUT
G	R	C	POTENTIAL AT G (MAPPING CONSTANT)	POTOUT
GAMMA	R	C	DIHEDRAL ANGLE IN DEGREES	POTOUT
GP	R	C	DIHEDRAL ANGLE DIVIDED BY PI	POTOUT
GPM2	R	C	GP - 0.5	POTOUT
GPP2	R	C	GP + 0.5	POTOUT
H	R	C	POTENTIAL AT H (MAPPING CONSTANT)	POTOUT
ILAST	I	C	FLAG FOR LAST ITERATION	POTOUT
ININT	I	C	NUMBER OF ORDINATES IN NUMERICAL INTEGRALS	POTOUT
MODE	I	C	PROGRAM MODE OF OPERATION	POTOUT
NTYPE	I	C	CONFIGURATION TYPE	POTOUT
OMGP	R	C	1.0 - GP	POTOUT
ORD	R	C	ORDINATE ARRAY FOR NUMERICAL INTEGRALS	POTOUT
PIREQ	R	C	INBOARD LENGTH OR DISTANCE (NONDIMENSIONAL)	POTOUT
POREQ	R	C	OUTBOARD LENGTH OR DISTANCE (NONDIMENSIONAL)	POTOUT
R1	R	C	SQUARE ROOT OF SUM OF SQUARES OF ERRORS IN REQD CONDITIONS	POTOUT
TAB	R	C	INTERPOLATION ARRAY FOR INDEPENDENT VARIABLE AND FUNCTION	POTOUT
TABLE	R	C	INTEGRATED VALUES OF ORDINATES AS A FUNCTION OF ABSCISSA	POTOUT

Subroutine DRAGI (Deck 55VU). This subroutine evaluates induced drag by quadrature, and writes out a summary of both geometry and induced drag results. The numerical integral for k (Equation 9) is divided into regions identical to those used to evaluate geometry in the real plane. The induced drag integrals between limits that correspond to vertical section of the wake are zero, since Munk's criterion (Figure 1) specifies zero velocity normal to such surfaces.

```
                SUBROUTINE DRAGI
          C
          C     THIS SUBROUTINE EVALUATES INDUCED DRAG AFTER CONFIGURATION
          C     GEOMETRY IS ESTABLISHED
          C
000002          DIMENSION A1(16)
          C
000002          COMMON DATA(17), ABSC(50), ORD(50), TABLE(50), ALOAD(1000), AA(16)
               A , TAB(101), GP, GPM2, GPP2, CG, OMGP, CONR, CON1, ILAST, C2, C1
               B , C3, R1
          C
000002          EQUIVALENCE                (DATA( 1),       A) , (DATA( 2),       B)
               A , (DATA( 3),       C) , (DATA( 4),       D) , (DATA( 5),       E)
               B , (DATA( 6),       F) , (DATA( 7),       G) , (DATA( 8),       H)
               C , (DATA( 9),      AI) , (DATA(10),    MODE) , (DATA(11),   NTYPE)
               D , (DATA(12),   ININT) , (DATA(13), ETAIRQ) , (DATA(14), ETAORQ)
               E , (DATA(15),   PIREQ) , (DATA(16),  POREQ) , (DATA(17),  GAMMA)
000002          EQUIVALENCE (AA,A1)
          C
000002          DATA ZERO, ONE, TWOBPI / 0.0, 1.0, 0.63661977 /
          C
000002          NINT = ININT
000004          GO TO 4
000004        1 WRITE (6,2) (DATA(I),I=1,9)
000016        2 FORMAT (1H0, 2X, 40H SUBROUTINE DRAGI.  THIS PASS REJECTED.  /
               A  2X, 9E13.4 )
000016          RETURN
000017        4 ITEST = 1
          C
          C     TRANSFER TO CODING FOR SPECIFIC CONFIGURATION
          C
000020          GO TO ( 5, 6, 7, 8, 9, 10, 11) , NTYPE
000033        5 CONTINUE
          C
          C     WING WITH ONE PYLON
          C
000033          IF (B .GE. 0.0 .AND. C .GE. B .AND. D .GE. C .AND. D .LE. 1.0
               A   .AND. F .GE. 1.0) ITEST = 0
000063          IF (ITEST .EQ. 1) GO TO 1
          C
          C     INTEGRAL FROM A TO B
          C
000065          CALL NUM     (     B,     A,     B, NINT, 1, ABINT, GT )
000074          IF(ABS(B   -   A).GT.C1) ABINT = ABINT + 2.0 * GT  * SQRT(B   -   A)
000110          CALL NUM     (  ONE,     D,     D, NINT, 2, DEINT, GT )
000117          IF(ABS(ONE - D).GT.C1) DEINT = DEINT + 2.0 * GT *  SQRT(ONE - D)
000133          CALL NUM     (  .   F,  ONE,     F, NINT, 3, EFINT, GT )
000142          IF(ABS(F - ONE).GT.C1) EFINT = EFINT + 2.0 * GT  * SQRT(F - ONE)
000156          A1(5) = DEINT + EFINT - ABINT
000161          EFF = TWOBPI * A1(5) / (A1(10)) ** 2
000164          WRITE (6,12) (A1(I) , I=1,10) , EFF
000200       12 FORMAT (1H / 38X, 28H CLOSURE INTEGRAL (PYLON) =          ,F16.8
               A           / 39X, 27H CLOSURE INTEGRAL (WING) =          ,F16.8
               B          // 39X, 27H PYLON SEMISPAN LOCATION =          ,F16.8
               C           / 38X, 28H PYLON LENGTH (SEMISPANS) =          ,F16.8
               D          // 41X, 25H INDUCED DRAG INTEGRAL =          ,F16.8
               E          // 42X, 24H DISTANCE FROM A TO B =          ,F16.8
```

```
                F            / 42X, 24H DISTANCE FROM B TO C =              ,F16.8
                G            / 42X, 24H DISTANCE FROM C TO D =              ,F16.8
                H            / 42X, 24H DISTANCE FROM D TO E =              ,F16.8
                I            / 42X, 24H DISTANCE FROM E TO F =              ,F16.8
                J           // 42X, 24H LIFTING EFFICIENCY K =              ,F16.8
                S //)
000200           RETURN
000201         6 CONTINUE
          C
          C      WING WITH TWO PYLONS
          C
000201           IF (A .LE. B .AND. B .LE. C .AND. C .LE. 0.0 .AND. E .GE. 0.0
                A  .AND. F .GE. E .AND. F .LE. 1.0 .AND. H .GE. 1.0 .AND. AI .GE.
                B   H)  ITEST = 0
000251           IF (ITEST .EQ. 1) GO TO 1
          C
          C      INTEGRAL FROM A TO B IS EVALUATED IN TWO PARTS
          C
000253           ALPHA = 0.5 * (A + B)
000256           CALL NUM     (ALPHA, A, A, NINT, 10, AB1, GT1)
000265           CALL NUM     (B, ALPHA, B, NINT, 11, AB2, GT2)
000274           IF(ABS(ALPHA-A).GT.C1)    AB1 =    AB1 + 2.0 * GT1 * SQRT(ALPHA-A)
000310           IF(ABS(B-ALPHA).GT.C1)    AB2 =    AB2 + 2.0 * GT2 * SQRT(B-ALPHA)
000324           ABINT = AB1 + AB2
000326           CALL NUM     (E, ZERO, E, NINT, 12, DEINT, GT)
000335           IF(ABS(       E).GT.C1) DEINT = DEINT + 2.0 * GT  * SQRT(       E)
000346           CALL NUM     (H, ONE, ONE, NINT, 13, GHINT, GT)
000355           IF(ABS(H - 1.0).GT.C1) GHINT = GHINT + 2.0 * GT  * SQRT(H - 1.0)
000371           CALL NUM     (AI, H, AI, NINT, 14, HIINT, GT)
000400           IF(ABS(AI - H).GT.C1) HIINT = HIINT + 2.0 * GT  * SQRT(AI -  H)
000414           A1(8) = GHINT + HIINT - ABINT - DEINT
000420           EFF = TWOBPI * A1(8) / (A1(16)) ** 2
000423           WRITE (6,33) (A1(I) , I=1,16) , EFF
000437        33 FORMAT (1H / 34X, 35H CLOSURE INTEGRAL(INBOARD PYLON) =     ,F16.8
                A            / 33X, 36H CLOSURE INTEGRAL(OUTBOARD PYLON) =   ,F16.8
                B            / 43X, 26H CLOSURE INTEGRAL(WING) =             ,F16.8
                C           // 34X, 35H INBOARD PYLON SEMISPAN LOCATION =    ,F16.8
                D            / 34X, 35HOUTBOARD PYLON SEMISPAN LOCATION =    ,F16.8
                E            / 33X, 36H INBOARD PYLON LENGTH (SEMISPANS) =   ,F16.8
                F            / 33X, 36HOUTBOARD PYLON LENGTH (SEMISPANS) =   ,F16.8
                G           // 44X, 25H INDUCED DRAG INTEGRAL =              ,F16.8
                H           // 45X, 24H DISTANCE FROM A TO B =               ,F16.8
                I            / 45X, 24H DISTANCE FROM B TO C =               ,F16.8
                J            / 45X, 24H DISTANCE FROM C TO D =               ,F16.8
                K            / 45X, 24H DISTANCE FROM D TO E =               ,F16.8
                L            / 45X, 24H DISTANCE FROM E TO F =               ,F16.8
                M            / 45X, 24H DISTANCE FROM F TO G =               ,F16.8
                N            / 45X, 24H DISTANCE FROM G TO H =               ,F16.8
                O            / 45X, 24H DISTANCE FROM H TO I =               ,F16.8
                P           // 45X, 24H LIFTING EFFICIENCY K =               ,F16.8
                S  //)
000437           RETURN
000440         7 CONTINUE
          C
          C      WING WITH HORIZONTAL
          C
000440           IF (A .LE. B .AND. B .LE. 0.0 .AND. D .GE. 0.0 .AND. D .LE. 1.0
                A   .AND. F .GE. 1.0) ITEST = 0
```

```
000471          IF (ITEST .EQ. 1) GO TO 1                                       55VU1160
000473          CALL NUM     (ZERO, A, A, NINT, 25, ACINT, GT)                  55VU1170
000502          IF(ABS(     - A).GT.C1) ACINT = ACINT + 2.0 * GT  * SQRT(     - A)  55VU1180
000514          CALL NUM     (ONE, D, D, NINT, 26, DEINT, GT)                   55VU1190
000523          IF(ABS(1.0 - D).GT.C1) DEINT = DEINT + 2.0 * GT  * SQRT(1.0 - D)  55VU1200
000537          CALL NUM     (F, ONE, F, NINT, 27, EFINT, GT)                   55VU1210
000556          IF(ABS(F - 1.0).GT.C1) EFINT = EFINT + 2.0 * GT  * SQRT(F - 1.0)  55VU1220
000562          A1(5) = ACINT + DEINT + EFINT                                   55VU1230
000565          EFF = TWOBPI * A1(5) / (A1(6)) ** 2                             55VU1240
000570          WRITE (6,53) (A1(I) , I=1,10) , EFF                             55VU1250
000604       53 FORMAT(1H0/42X, 27H CLOSURE INTEGRAL (WING) =                 ,F15.8 55VU1260
             A            /36X, 33H CLOSURE INTEGRAL (HORIZONTAL) =          ,F15.8 55VU1270
             B           //35X, 34H HORIZONTAL LOCATION ABOVE WING =         ,F15.8 55VU1280
             C            /34X, 35H HORIZONTAL SEMISPAN  / WING B/2 =       ,F15.8 55VU1290
             D            /44X, 25H INDUCED DRAG INTEGRAL =                 ,F15.8 55VU1300
             E           //45X, 24H DISTANCE FROM A TO B =                   ,F15.8 55VU1310
             F            /45X, 24H DISTANCE FROM B TO C =                  ,F15.8 55VU1320
             G            /45X, 24H DISTANCE FROM C TO D =                  ,F15.8 55VU1330
             H            /45X, 24H DISTANCE FROM D TO E =                  ,F15.8 55VU1340
             I            /45X, 24H DISTANCE FROM E TO F =                  ,F15.8 55VU1350
             J           //45X, 24H LIFTING EFFICIENCY K =                   ,F15.8 55VU1360
             S //)                                                            55VU1370
000604          RETURN                                                          55VU1380
000605        8 CONTINUE                                                        55VU1390
        C                                                                       55VU1400
        C       WING WITH PYLON ON TOP AND ON BOTTOM                            55VU1410
        C                                                                       55VU1420
000605          IF (A. LE. B .AND. B .LE. C .AND. C .LE. 0.0 .AND. E .GE. 0.0   55VU1430
             A  .AND. E .LE. F .AND. F .LE. 1.0 .AND. H .GE. 1.0 .AND. H .LE.   55VU1440
             B    AI)  ITEST = 0                                              55VU1450
000655          IF (ITEST .EQ. 1) GO TO 1                                       55VU1460
000657          ALPHA = 0.5 * (A + B)                                           55VU1470
000662          CALL NUM     (ALPHA, A, A, NINT, 34, AB1, GT1)                  55VU1480
000671          CALL NUM     (B, ALPHA, B, NINT, 35, AB2, GT2)                  55VU1490
000700          IF(ABS(ALPHA-A).GT.C1)   AB1 =   AB1 + 2.0 * GT1 * SQRT(ALPHA-A)  55VU1500
000714          IF(ABS(B-ALPHA).GT.C1)   AB2 =   AB2 + 2.0 * GT2 * SQRT(B-ALPHA)  55VU1510
000730          ABINT = AB1 + AB2                                               55VU1520
000732          CALL NUM     (F, ZERO, F, NINT, 36, DFINT, GT)                  55VU1530
000741          IF(ABS(      F).GT.C1) DFINT = DFINT + 2.0 * GT  * SQRT(F)      55VU1540
000752          ALPHA = 0.5 * (H + AI)                                          55VU1550
000755          CALL NUM     (ALPHA, H, H, NINT, 38, HI1, GT1)                  55VU1560
000764          CALL NUM     (AI,ALPHA,AI, NINT, 39, HI2, GT2)                  55VU1570
000773          IF(ABS(ALPHA-H).GT.C1)   HI1 = HI1 + 2.0 * GT1 * SQRT(ALPHA-H)  55VU1580
001007          IF(ABS(AI-ALPHA).GT.C1)  HI2 = HI2 + 2.0 * GT2 * SQRT(AI-ALPHA)  55VU1590
001023          HIINT = HI1 + HI2                                               55VU1600
001025          A1(8) = HIINT - ABINT - DFINT                                   55VU1610
001030          EFF = TWOBPI * A1(8) / (A1(9) + A1(12)) ** 2                    55VU1620
001034          WRITE (6,73) (A1(I) , I=1,16) , EFF                             55VU1630
001050       73 FORMAT(1H /35X, 34H CLOSURE INTEGRAL (LOWER PYLON) =          ,F16.8 55VU1640
             A            /35X, 34H CLOSURE INTEGRAL (UPPER PYLON) =          ,F16.8 55VU1650
             B            /42X, 27H CLOSURE INTEGRAL (WING) =                 ,F16.8 55VU1660
             C           //33X, 36H  INBOARD PYLON SEMISPAN LOCATION =        ,F16.8 55VU1670
             D            /33X, 36H OUTBOARD PYLON SEMISPAN LOCATION =       ,F16.8 55VU1680
             E            /32X, 37H  INBOARD PYLON LENGTH (SEMISPANS) =      ,F16.8 55VU1690
             F            /32X, 37H OUTBOARD PYLON LENGTH (SEMISPANS) =      ,F16.8 55VU1700
             G           //44X, 25H INDUCED DRAG INTEGRAL =                  ,F16.8 55VU1710
             H           //45X, 24H DISTANCE FROM A TO B =                   ,F16.8 55VU1720
             I            /45X, 24H DISTANCE FROM B TO C =                  ,F16.8 55VU1730
```

```
          J          /45X, 24H DISTANCE FROM C TO D =                    ,F16.8 55VU1760
          K          /45X, 24H DISTANCE FROM D TO E =                    ,F16.8 55VU1770
          L          /45X, 24H DISTANCE FROM E TO F =                    ,F16.8 55VU1780
          M          /45X, 24H DISTANCE FROM F TO G =                    ,F16.8 55VU1770
          N          /45X, 24H DISTANCE FROM G TO H =                    ,F16.8 55VU1780
          O          /45X, 24H DISTANCE FROM H TO I =                    ,F16.8 55VU1790
          P         //45X, 24H LIFTING EFFICIENCY K =                    ,F16.8 55VU1800
          S //)                                                                 55VU1810
001050     RETURN                                                               55VU1820
001051   9 CONTINUE                                                             55VU1830
      C                                                                         55VU1840
      C    WING WITH PYLON BELOW AND DIHEDRAL                                   55VU1850
      C                                                                         55VU1860
001051     CALL NUM     (ZERO, A, A, NINT, 50, ABINT, GT)                       55VU1870
001060     IF(ABS(-A).GT.C1) ABINT = ABINT + 2.0 * GT * SQRT(-A)                55VU1880
001072     CALL NUM     (ONE, D, D, NINT, 51, DEINT, GT)                        55VU1890
001101     IF(ABS(1.0-D).GT.C1) DEINT = DEINT + GT * (1.0 - D) ** GPP2 / GPP2 55VU1900
001115     DEINT = DEINT * CONI                                                 55VU1910
001117     CALL NUM     (F, ONE, F, NINT, 52, EFINT, GT)                        55VU1920
001125     IF(ABS(F-1.0).GT.C1) EFINT = EFINT + GT*(F-1.0)**OMGP/OMGP           55VU1930
001141     EFINT = EFINT * CONI                                                 55VU1940
001143     ALPHA = 0.5 * (F + G)                                                55VU1950
001146     CALL NUM     (ALPHA, F, F, NINT, 53, FG1, GT1)                       55VU1960
001154     CALL NUM     (G, ALPHA, G, NINT, 54, FG2, GT2)                       55VU1970
001163     IF(ABS(ALPHA-F).GT.C1) FG1 = FG1 + GT1* (ALPHA-F) ** OMGP / OMGP    55VU1980
001177     IF(ABS(G-ALPHA).GT.C1) FG2 = FG2 + 2.0 * GT2 * SQRT(G-ALPHA)         55VU1990
001213     FGINT = FG1 + FG2                                                    55VU2000
001215     A1(6) = ABINT - DEINT - EFINT + FGINT                                55VU2010
001221     FFF = TWOPI * A1(6) / (A1(7) + A1(12)) ** 2                          55VU2020
001225     WRITE (6,89) (A1(I),I=1,16) , GAMMA , FFF                            55VU2030
001243  89 FORMAT (1H0/ 33X, 36H CLOSURE INTEGRAL ( INBOARD WING) =     ,F16.8 55VU2040
          A            / 41X, 28H CLOSURE INTEGRAL (PYLON) =            ,F16.8 55VU2050
          B            / 33X, 36H CLOSURE INTEGRAL (OUTBOARD WING) =    ,F16.8 55VU2060
          C           // 39X, 30H PYLON LENGTH    (SEMISPANS) =         ,F16.8 55VU2070
          D            / 39X, 30H PYLON LOCATION (SEMISPANS) =          ,F16.8 55VU2080
          E           // 44X, 25H INDUCED DRAG INTEGRAL =                ,F16.8 55VU2090
          F           // 45X, 24H DISTANCE FROM A TO B =                 ,F16.8 55VU2100
          G            / 45X, 24H DISTANCE FROM B TO C =                ,F16.8 55VU2110
          H            / 45X, 24H DISTANCE FROM C TO D =                ,F16.8 55VU2120
          I            / 45X, 24H DISTANCE FROM D TO E =                ,F16.8 55VU2130
          J            / 45X, 24H  REAL PART OF D TO E =                ,F16.8 55VU2140
          K            / 45X, 24H IMAJ. PART OF D TO E =                ,F16.8 55VU2150
          L            / 45X, 24H DISTANCE FROM E TO F =                ,F16.8 55VU2160
          M            / 45X, 24H  REAL PART OF E TO F =                ,F16.8 55VU2170
          N            / 45X, 24H IMAJ. PART OF E TO F =                ,F16.8 55VU2180
          O            / 45X, 24H DISTANCE FROM F TO G =                ,F16.8 55VU2190
          P           // 38X, 31H OUTBOARD DIHEDRAL (DEGREES) =         ,F16.8 55VU2200
          Q           // 45X, 24H LIFTING EFFICIENCY K =                 ,F16.8 55VU2210
          S //)                                                                 55VU2220
001243     RETURN                                                               55VU2230
001244  10 CONTINUE                                                             55VU2240
      C                                                                         55VU2250
      C    WING WITH INBOARD PYLON AND OUTBOARD DIHEDRAL                        55VU2260
      C                                                                         55VU2270
001244     IF ( A .LE. 0.0 .AND. C .GE. 0.0 .AND. D .GE. C .AND. E .GT. D       55VU2280
         A   .AND. E .LE. 1.0 .AND. G .GE. 1.0 .AND. H .GE. G) ITEST = 0        55VU2290
001306     IF (ITEST .EQ. 1) GO TO 1                                            55VU2300
001310     CALL NUM     ( ZERO,     A,     A, NINT, 63, ABINT, GT )             55VU2310
```

```
001317        IF(ABS(      -A).GT. C1) ABINT = ABINT + 2.0 * GT  * SQRT(      -A) 55VU2320
001331        CALL NUM     (    E,    D,    D, NINT, 64, DEINT, GT )             55VU2330
001340        IF(ABS(E  -  D).GT. C1) DEINT = DEINT + 2.0 * GT  * SQRT(E  -  D) 55VU2340
001354        CALL NUM     (  ONE,    E,    E, NINT, 65, EFINT, GT )             55VU2350
001363        IF(ABS(ONE - E).GT. C1) EFINT = EFINT + GT * (ONE -E)**GPP2 /GPP2 55VU2360
001377        EFINT = EFINT * CG                                                 55VU2370
001401        CALL NUM     (    G,  ONE,    G, NINT, 66, FGINT, GT )             55VU2380
001407        IF(ABS(G - ONE).GT. C1) FGINT = FGINT + GT * (G -ONE)**GPM2 /GPM2 55VU2390
001423        ALPHA = 0.5 * (G + H)                                              55VU2400
001426        FGINT = FGINT * CG                                                 55VU2410
001430        CALL NUM     (ALPHA,    G,    G, NINT, 67, GH1  , GT1)             55VU2420
001436        IF(ABS(ALPHA-G).GT. C1) GH1   = GH1   + GT1* (ALPHA-G)**GPM2/GPM2 55VU2430
001452        CALL NUM     (    H,ALPHA,    H, NINT, 68, GH2  , GT2)             55VU2440
001461        IF(ABS(H-ALPHA).GT. C1) GH2   = GH2   + 2.0 * GT2 * SQRT(H-ALPHA) 55VU2450
001475        GHINT = GH1 + GH2                                                  55VU2460
001477        A1(7) = ABINT + DEINT + EFINT + FGINT + GHINT                      55VU2470
001504        EFF = TWOBPI * A1(7) / (A1(13) * CG + A1(14)) ** 2                 55VU2480
001511        WRITE (6,91) (A1(I),I=1,14) , GAMMA, EFF                           55VU2490
001526     91 FORMAT (1H0/35X, 35H CLOSURE INTEGRAL (INBOARD WING) =      ,F16.8 55VU2500
             A            /42X, 28H CLOSURE INTEGRAL (PYLON) =            ,F16.8 55VU2510
             B            /34X, 36H CLOSURE INTEGRAL (OUTBOARD WING) =    ,F16.8 55VU2520
             C           //41X, 29H PYLON LOCATION (SEMISPAN) =            ,F16.8 55VU2530
             D            /32X, 38H DIHEDRAL BREAK LOCATION (SEMISPAN) =  ,F16.8 55VU2540
             E            /42X, 28H PYLON LENGTH (SEMISPANS) =            ,F16.8 55VU2550
             F           //45X, 25H INDUCED DRAG INTEGRAL =                ,F16.8 55VU2560
             G           //46X, 24H DISTANCE FROM A TO B =                 ,F16.8 55VU2570
             H            /46X, 24H DISTANCE FROM B TO C =                ,F16.8 55VU2580
             I            /46X, 24H DISTANCE FROM C TO D =                ,F16.8 55VU2590
             J            /46X, 24H DISTANCE FROM D TO E =                ,F16.8 55VU2600
             K            /46X, 24H DISTANCE FROM E TO F =                ,F16.8 55VU2610
             L            /46X, 24H DISTANCE FROM F TO G =                ,F16.8 55VU2620
             M            /46X, 24H DISTANCE FROM G TO H =                ,F16.8 55VU2630
             N           //42X, 28H DIHEDRAL ANGLE (DEGREES) =            ,F16.8 55VU2640
             O           //46X, 24H LIFTING EFFICIENCY K =                 ,F16.8 55VU2650
             S //)                                                                55VU2660
001526        RETURN                                                             55VU2670
001527     11 CONTINUE                                                           55VU2680
        C                                                                        55VU2690
        C     WING WITH END-PLATED VERTICAL FENCE AT THE TIP                     55VU2700
        C                                                                        55VU2710
001527        IF (B .GE. 0.0 .AND. C .GE.  B  .AND. D .GE. C .AND. D .LE. 1.0    55VU2720
             A  .AND. F .GE. 1.0 .AND. G .GE. F .AND. H .GE. G) ITEST = 0        55VU2730
001571        IF (ITEST .EQ. 1) GO TO 1                                          55VU2740
001573        CALL NUM     (    B,    A,    B, NINT, 79, ABINT, GT )             55VU2750
001602        IF (ABS(B  -  A).GT.C1) ABINT = ABINT + 2.0 * GT  * SQRT(B  -  A) 55VU2760
001616        CALL NUM     (  ONE,    C,    C, NINT, 80, CEINT, GT )             55VU2770
001625        IF (ABS(ONE - C).GT.C1) CEINT = CEINT + 2.0 * GT * SQRT(ONE - C)   55VU2780
001641        CALL NUM     (    F,  ONE,    F, NINT, 81, EFINT, GT )             55VU2790
001650        IF (ABS(F - ONE).GT.C1) EFINT = EFINT + 2.0 * GT * SQRT(F - ONE)   55VU2800
001664        CALL NUM     (    H,    G,    H, NINT, 82, GHINT, GT )             55VU2810
001673        IF (ABS(H  -  G).GT.C1) GHINT = GHINT + 2.0 * GT * SQRT(H  -  G)   55VU2820
001707        A1(7) = CEINT + EFINT + GHINT - ABINT                              55VU2830
001713        EFF = TWOBPI * A1(7) / (A1(8)) ** 2                                55VU2840
001716        WRITE (6,93) (A1(I),I=1,14) , EFF                                  55VU2850
001732     93 FORMAT (1H0/45X, 25H WING CLOSURE INTEGRAL =                 ,F16.8 55VU2860
             A            /35X, 35H VERTICAL FENCE CLOSURE INTEGRAL =     ,F16.8 55VU2870
             B            /40X, 30H END-PLATE CLOSURE INTEGRAL =          ,F16.8 55VU2880
             C           //45X, 25H VERTICAL FENCE LENGTH =                 ,F16.8 55VU2890
```

```
                 /42X, 28H INBOARD END-PLATE LENGTH =                  ,F16.8 55VU2900
D                /41X, 29H OUTBOARD END-PLATE LENGTH =                 ,F16.8 55VU2910
E               //45X, 25H INDUCED DRAG INTEGRAL =                     ,F16.8 55VU2920
F               //46X, 24H DISTANCE FROM A TO B =                      ,F16.8 55VU2930
G                /46X, 24H DISTANCE FROM B TO C =                      ,F16.8 55VU2940
H                /46X, 24H DISTANCE FROM C TO D =                      ,F16.8 55VU2950
I                /46X, 24H DISTANCE FROM D TO E =                      ,F16.8 55VU2960
J                /46X, 24H DISTANCE FROM E TO F =                      ,F16.8 55VU2970
K                /46X, 24H DISTANCE FROM F TO G =                      ,F16.8 55VU2980
L                /46X, 24H DISTANCE FROM G TO H =                      ,F16.8 55VU2990
M                /46X, 24H DISTANCE FROM G TO H =                      ,F16.8 55VU3000
N               //46X, 24H LIFTING EFFICIENCY K =                                55VU3010
S //)                                                                            55VU3020
001732          RETURN                                                           55VU3030
001733          END
```

SUBPROGRAM LENGTH
003365

FUNCTION ASSIGNMENTS

STATEMENT ASSIGNMENTS

1	-	000004	2	-	001737	4	-	000017	5	-	000033
6	-	000201	7	-	000440	8	-	000605	9	-	001051
10	-	001244	11	-	001527	12	-	001755	33	-	002075
53	-	002262	73	-	002401	89	-	002570	91	-	002766
93	-	003145									

BLOCK NAMES AND LENGTHS
- 002420

VARIABLE ASSIGNMENTS

A	-	000000C01	AA	-	002217C01	ABINT	-	003337	ABSC	-	000021C01
AB1	-	003345	AB2	-	003347	ACINT	-	003353	AI	-	000010C01
ALOAD	-	000247C01	ALPHA	-	003344	A1	-	002217C01	B	-	000001C01
C	-	000002C01	CEINT	-	003364	CG	-	002407C01	CONI	-	002412C01
C1	-	002415C01	D	-	000003C01	DATA	-	000000C01	DEINT	-	003341
DFINT	-	003354	E	-	000004C01	EFF	-	003343	FFINT	-	003342
F	-	000005C01	FGINT	-	003361	FG1	-	003357	FG2	-	003360
G	-	000006C01	GAMMA	-	000020C01	GHINT	-	003351	GH1	-	003362
GH2	-	003363	GPH2	-	002405C01	GPP2	-	002406C01	GT	-	003340
GT1	-	003346	GT2	-	003350	H	-	000007C01	HIINT	-	003352
HI1	-	003355	HI2	-	003356	I	-	003335	ININT	-	000013C01
ITEST	-	003336	NINT	-	003334	NTYPE	-	000012C01	OMGP	-	002410C01
ONE	-	003332	ORD	-	000103C01	TAB	-	002237C01	TABLE	-	000165C01
TWOBPI	-	003333	ZERO	-	003331						

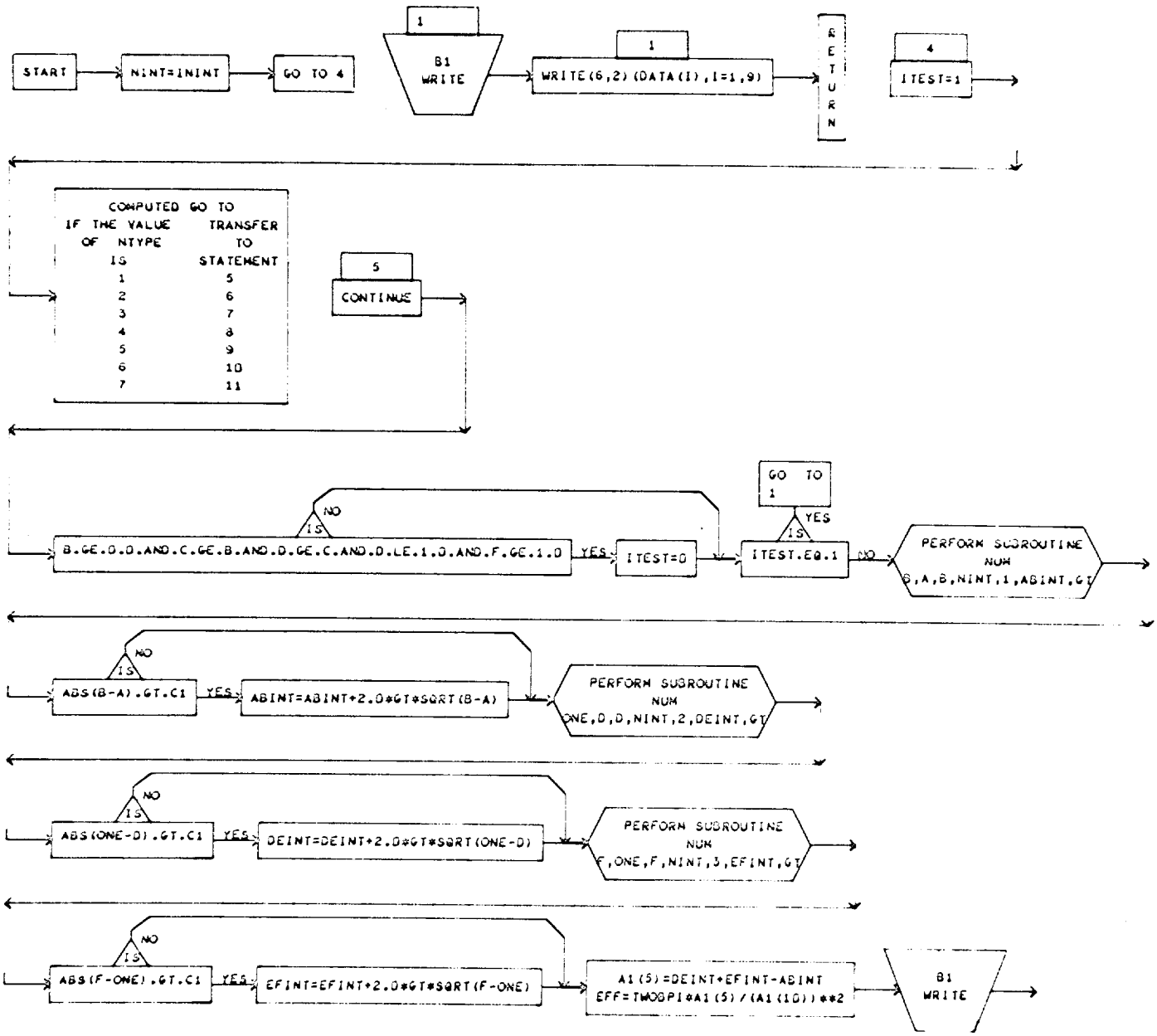
START OF CONSTANTS
001735

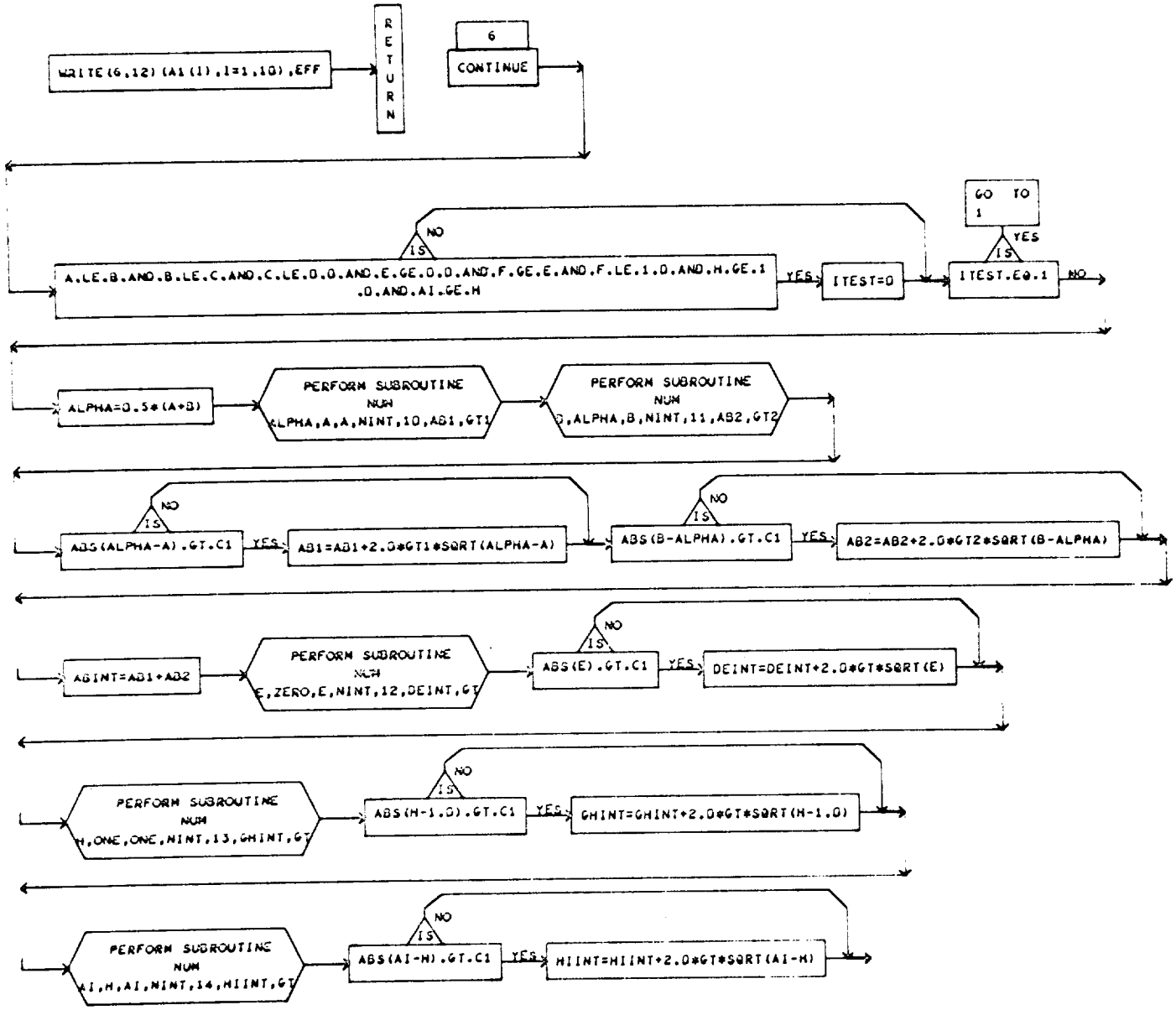
START OF TEMPORARIES
003311

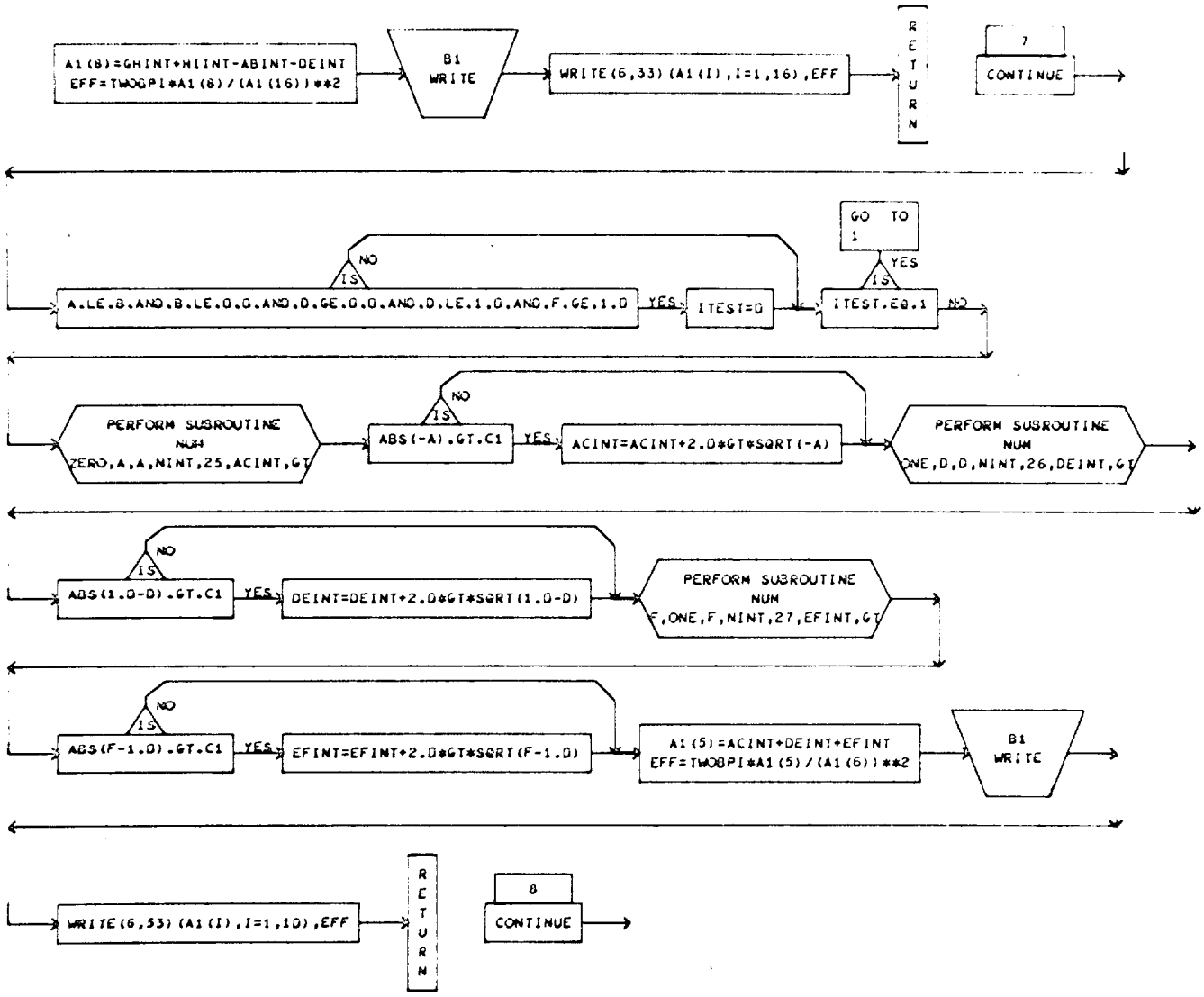
START OF INDIRECTS
003331

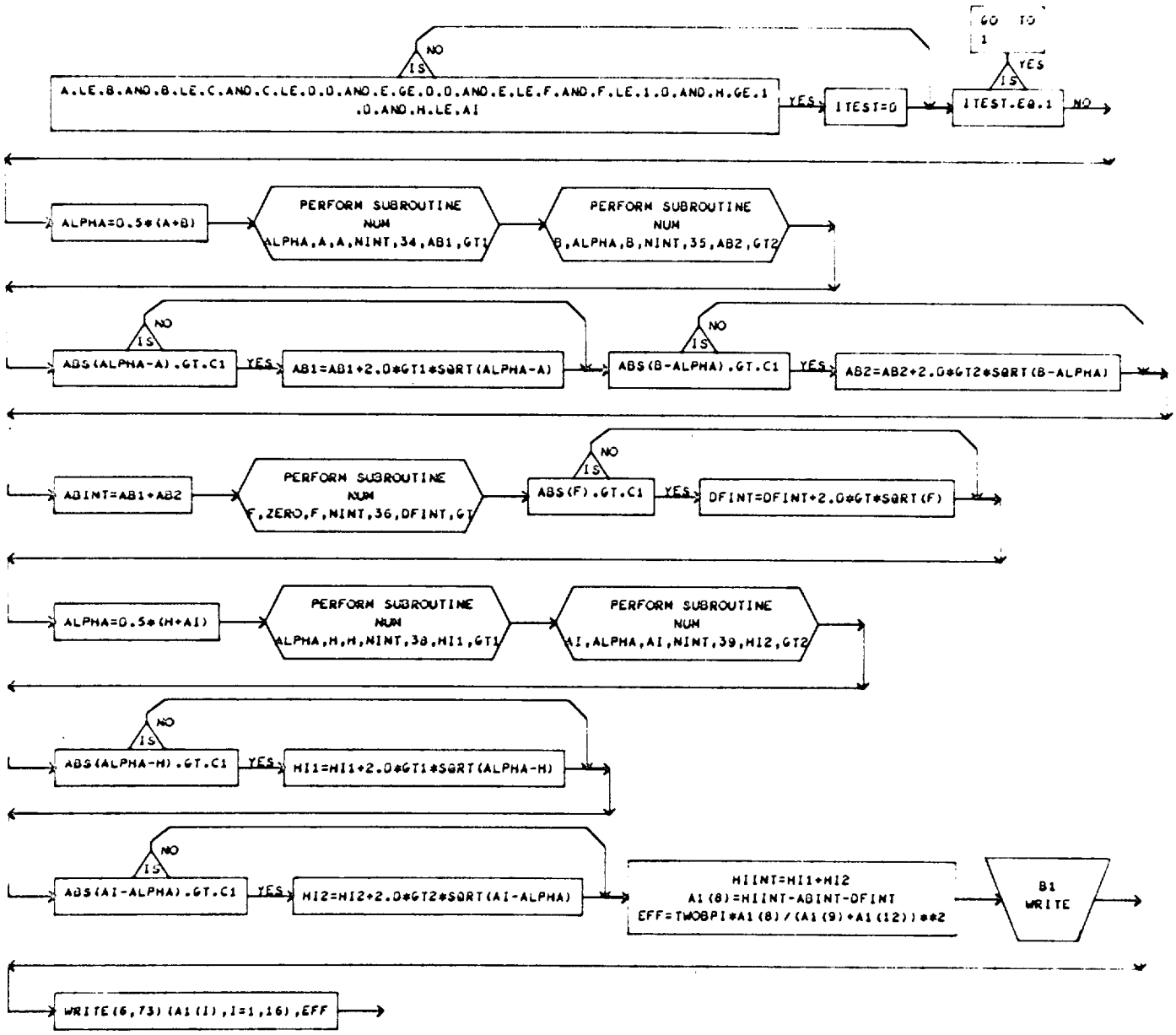
UNUSED COMPILER SPACE
033300

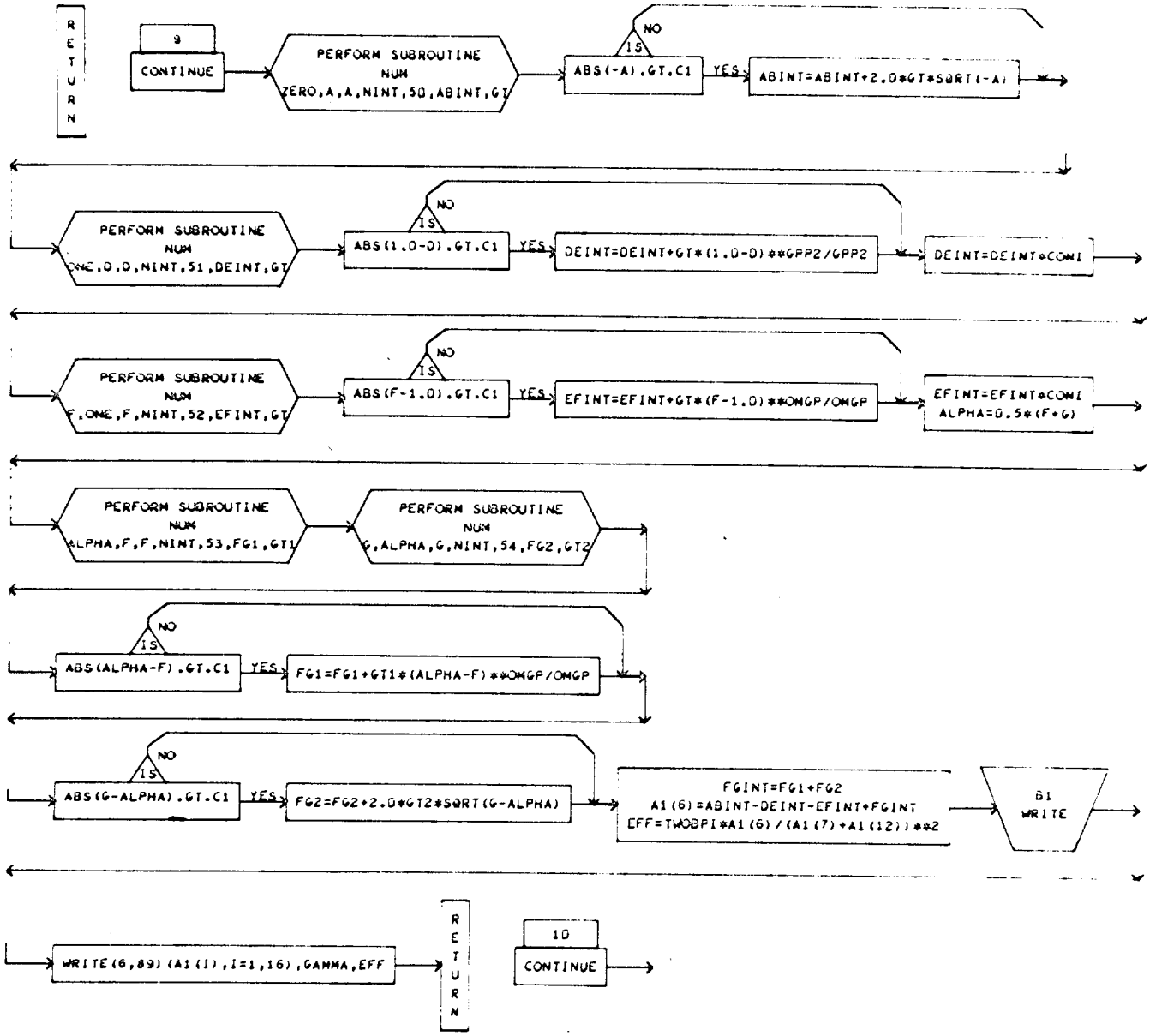
SUBROUTINE DRAG1

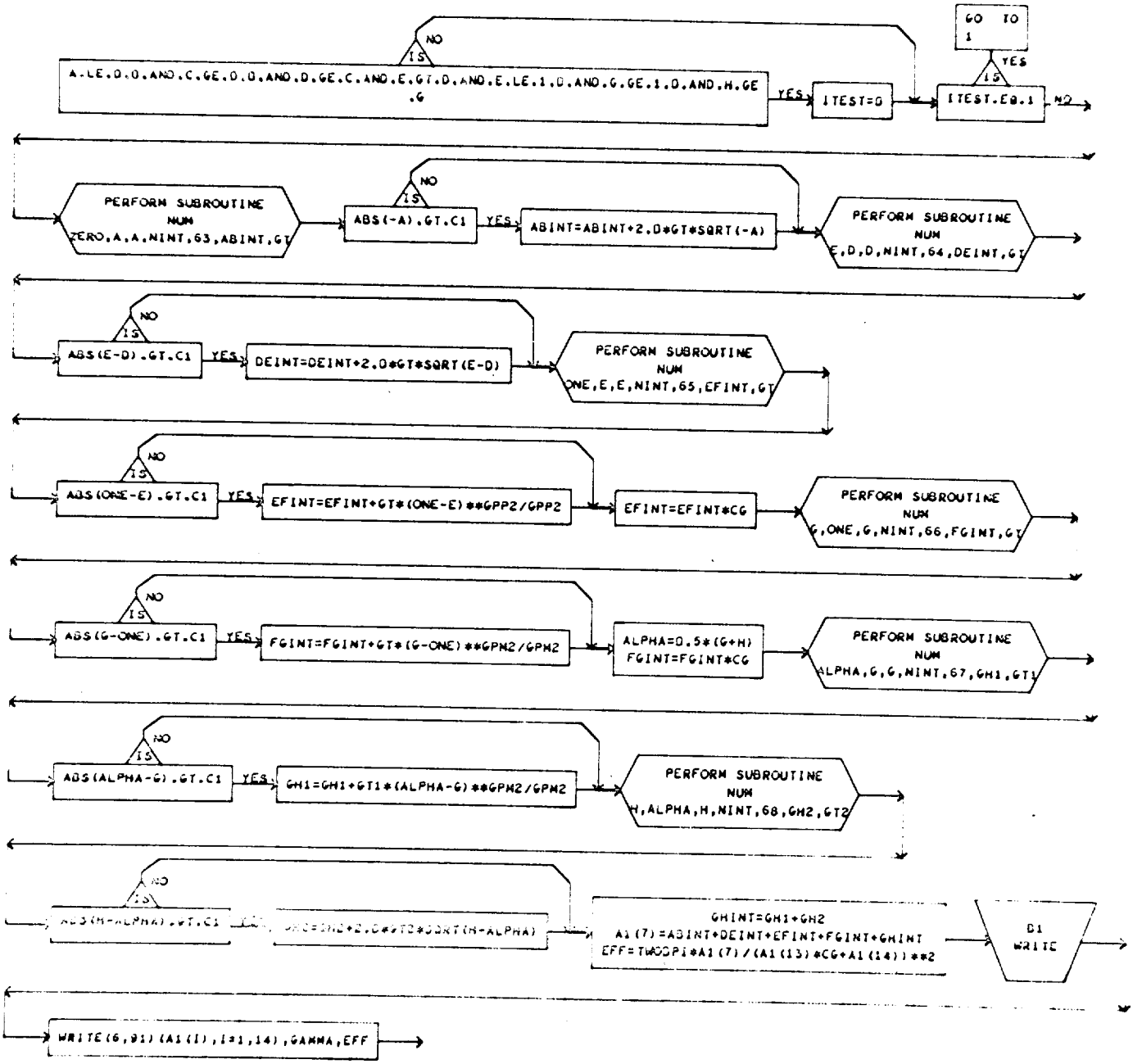


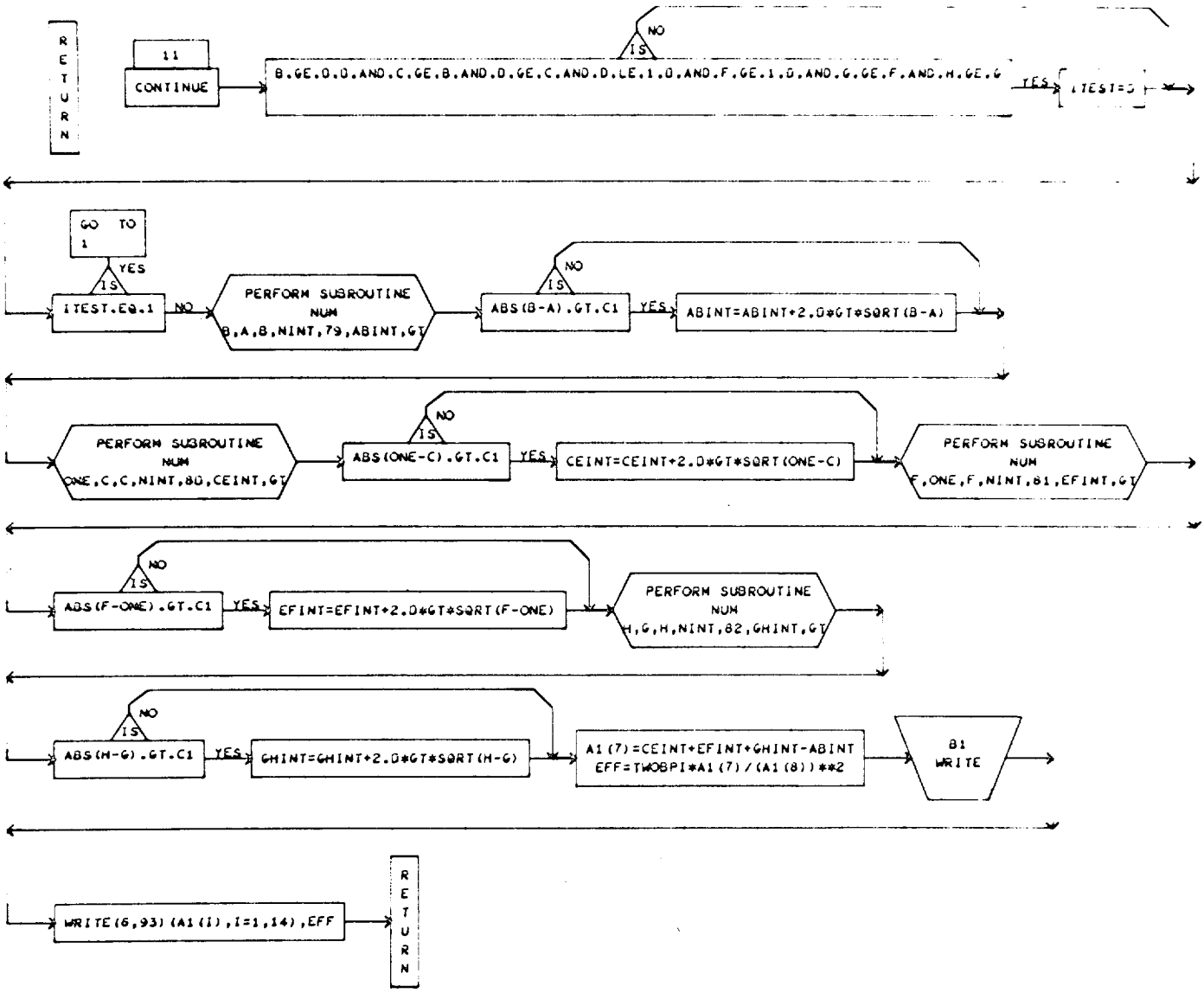

WRITE(6,12)(A1(I),I=1,10),EFF
RETURN
6
CONTINUE
A.LE.B.AND.B.LE.C.AND.C.LE.D.0.AND.E.GE.0.0.AND.F.GE.E.AND.F.LE.1.0.AND.H.GE.1.0.AND.AI.GE.H
IS
NO
YES
ITEST=0
GO TO 1
ITEST.EQ.1
ALPHA=0.5*(A+B)
PERFORM SUBROUTINE NUM
ALPHA,A,A,NINT,10,AB1,GT1
PERFORM SUBROUTINE NUM
ALPHA,B,NINT,11,AB2,GT2
ABS(ALPHA-A).GT.C1
AB1=AB1+2.0*GT1*SQRT(ALPHA-A)
ABS(B-ALPHA).GT.C1
AB2=AB2+2.0*GT2*SQRT(B-ALPHA)
ABINT=AB1+AB2
PERFORM SUBROUTINE NUM
ZERO,E,NINT,12,DEINT,GT
ABS(E).GT.C1
DEINT=DEINT+2.0*GT*SQRT(E)
PERFORM SUBROUTINE NUM
H,ONE,ONE,NINT,13,GHINT,GT
ABS(H-1.0).GT.C1
GHINT=GHINT+2.0*GT*SQRT(H-1.0)
PERFORM SUBROUTINE NUM
AI,H,AI,NINT,14,HIINT,GT
ABS(AI-H).GT.C1
HIINT=HIINT+2.0*GT*SQRT(AI-H)


A1(8)=GHINT+HIINT-ABINT-DEINT
EFF=TWOOPI*A1(8)/(A1(16))**2
B1
WRITE
WRITE(6,33)(A1(I),I=1,16),EFF
RETURN
7
CONTINUE
GO TO
1
NO
IS
YES
A.LE.B.AND.B.LE.0.0.AND.D.GE.0.0.AND.D.LE.1.0.AND.F.GE.1.0
ITEST=0
ITEST.EQ.1
PERFORM SUBROUTINE
NUM
ZERO,A,A,NINT,25,ACINT,GT
ABS(-A).GT.C1
ACINT=ACINT+2.0*GT*SQRT(-A)
PERFORM SUBROUTINE
NUM
ONE,D,D,NINT,26,DEINT,GT
ABS(1.0-D).GT.C1
DEINT=DEINT+2.0*GT*SQRT(1.0-D)
PERFORM SUBROUTINE
NUM
F,ONE,F,NINT,27,EFINT,GT
ABS(F-1.0).GT.C1
EFINT=EFINT+2.0*GT*SQRT(F-1.0)
A1(5)=ACINT+DEINT+EFINT
EFF=TWOOPI*A1(5)/(A1(6))**2
B1
WRITE
WRITE(6,53)(A1(I),I=1,10),EFF
RETURN
8
CONTINUE


GO TO 1
IS
NO
YES
A.LE.B.AND.B.LE.C.AND.C.LE.D.0.AND.E.GE.0.0.AND.E.LE.F.AND.F.LE.1.0.AND.H.GE.1.0.AND.H.LE.AI
ITEST=0
ITEST.EQ.1
ALPHA=0.5*(A+B)
PERFORM SUBROUTINE NUM ALPHA,A,A,NINT,34,AB1,GT1
PERFORM SUBROUTINE NUM B,ALPHA,B,NINT,35,AB2,GT2
ABS(ALPHA-A).GT.C1
AB1=AB1+2.0*GT1*SQRT(ALPHA-A)
ABS(B-ALPHA).GT.C1
AB2=AB2+2.0*GT2*SQRT(B-ALPHA)
ABINT=AB1+AB2
PERFORM SUBROUTINE NUM F,ZERO,F,NINT,36,DFINT,GT
ABS(F).GT.C1
DFINT=DFINT+2.0*GT*SQRT(F)
ALPHA=0.5*(H+AI)
PERFORM SUBROUTINE NUM ALPHA,H,H,NINT,38,HI1,GT1
PERFORM SUBROUTINE NUM AI,ALPHA,AI,NINT,39,HI2,GT2
ABS(ALPHA-H).GT.C1
HI1=HI1+2.0*GT1*SQRT(ALPHA-H)
ABS(AI-ALPHA).GT.C1
HI2=HI2+2.0*GT2*SQRT(AI-ALPHA)
HIINT=HI1+HI2
A1(8)=HIINT-ABINT-DFINT
EFF=TWOBPI*A1(8)/(A1(9)+A1(12))**2
B1 WRITE
WRITE(6,73)(A1(I),I=1,16),EFF


RETURN
9
CONTINUE
PERFORM SUBROUTINE
NUM
ZERO,A,A,NINT,50,ABINT,GT
ABS(-A).GT.C1
IS
NO
YES
ABINT=ABINT+2.D*GT*SQRT(-A)
PERFORM SUBROUTINE
NUM
ONE,D,D,NINT,51,DEINT,GT
ABS(1.D-D).GT.C1
DEINT=DEINT+GT*(1.D-D)**GPP2/GPP2
DEINT=DEINT*CONI
PERFORM SUBROUTINE
NUM
F,ONE,F,NINT,52,EFINT,GT
ABS(F-1.D).GT.C1
EFINT=EFINT+GT*(F-1.D)**OMGP/OMGP
EFINT=EFINT*CONI
ALPHA=0.5*(F+G)
PERFORM SUBROUTINE
NUM
ALPHA,F,F,NINT,53,FG1,GT1
PERFORM SUBROUTINE
NUM
G,ALPHA,G,NINT,54,FG2,GT2
ABS(ALPHA-F).GT.C1
FG1=FG1+GT1*(ALPHA-F)**OMGP/OMGP
ABS(G-ALPHA).GT.C1
FG2=FG2+2.D*GT2*SQRT(G-ALPHA)
FGINT=FG1+FG2
A1(6)=ABINT-DEINT-EFINT+FGINT
EFF=TWOBPI*A1(6)/(A1(7)+A1(12))**2
61
WRITE
WRITE(6,89)(A1(I),I=1,16),GAMMA,EFF
RETURN
10
CONTINUE


A.LE.0.0.AND.C.GE.0.0.AND.D.GE.C.AND.E.GT.D.AND.E.LE.1.0.AND.G.GE.1.0.AND.H.GE.G
IS
NO
YES
ITEST=0
ITEST.EQ.1
GO TO 1
PERFORM SUBROUTINE NUM ZERO,A,A,NINT,63,ABINT,GT
ABS(-A).GT.C1
ABINT=ABINT+2.0*GT*SQRT(-A)
PERFORM SUBROUTINE NUM E,D,D,NINT,64,DEINT,GT
ABS(E-D).GT.C1
DEINT=DEINT+2.0*GT*SQRT(E-D)
PERFORM SUBROUTINE NUM ONE,E,E,NINT,65,EFINT,GT
ABS(ONE-E).GT.C1
EFINT=EFINT+GT*(ONE-E)**GPP2/GPP2
EFINT=EFINT*CG
PERFORM SUBROUTINE NUM G,ONE,G,NINT,66,FGINT,GT
ABS(G-ONE).GT.C1
FGINT=FGINT+GT*(G-ONE)**GPM2/GPM2
ALPHA=0.5*(G+H)
FGINT=FGINT*CG
PERFORM SUBROUTINE NUM ALPHA,G,G,NINT,67,GH1,GT1
ABS(ALPHA-G).GT.C1
GH1=GH1+GT1*(ALPHA-G)**GPM2/GPM2
PERFORM SUBROUTINE NUM H,ALPHA,H,NINT,68,GH2,GT2
GHINT=GH1+GH2
A1(7)=ABINT+DEINT+EFINT+FGINT+GHINT
EFF=TWOOPI*A1(7)/(A1(13)*CG+A1(14))**2
D1 WRITE
WRITE(6,91)(A1(I),I=1,14),GAMMA,EFF


RETURN
11
CONTINUE
B.GE.0.0.AND.C.GE.B.AND.D.GE.C.AND.D.LE.1.0.AND.F.GE.1.0.AND.G.GE.F.AND.H.GE.G
IS
NO
YES
ITEST=3
GO TO 1
YES
IS
ITEST.EQ.1
NO
PERFORM SUBROUTINE NUM
B,A,B,NINT,79,ABINT,GT
ABS(B-A).GT.C1
ABINT=ABINT+2.0*GT*SQRT(B-A)
PERFORM SUBROUTINE NUM
ONE,C,C,NINT,80,CEINT,GT
ABS(ONE-C).GT.C1
CEINT=CEINT+2.0*GT*SQRT(ONE-C)
PERFORM SUBROUTINE NUM
F,ONE,F,NINT,81,EFINT,GT
ABS(F-ONE).GT.C1
EFINT=EFINT+2.0*GT*SQRT(F-ONE)
PERFORM SUBROUTINE NUM
H,G,H,NINT,82,GHINT,GT
ABS(H-G).GT.C1
GHINT=GHINT+2.0*GT*SQRT(H-G)
A1(7)=CEINT+EFINT+GHINT-ABINT
EFF=TWOBPI*A1(7)/(A1(8))**2
B1
WRITE
WRITE(6,93)(A1(I),I=1,14),EFF
RETURN

SYMBOLS USED IN SUBROUTINE DRAGI (DECK IDENTIFICATION 55V0)

A	R	C	POTENTIAL AT A (MAPPING CONSTANT)	DRAGI
AA	R	C	ARRAY FOR RESULTS OF GEOMETRY AND DRAG INTEGRALS	DRAGI
ABINT	R	U	INTEGRAL FROM A TO B	DRAGI
ABSC	R	C	ABSCISSA ARRAY FOR NUMERICAL INTEGRALS	DRAGI
AB1	R	U	FIRST HALF OF INTEGRAL FROM A TO B	DRAGI
AB2	R	U	SECOND HALF OF INTEGRAL FROM A TO B	DRAGI
ACINT	R	U	INTEGRAL FROM A TO C	DRAGI
AI	R	C	POTENTIAL AT I (MAPPING CONSTANT)	DRAGI
ALOAD	R	C	STORAGE ARRAY FOR POTENTIAL IN THE REAL (PHYSICAL) PLANE	DRAGI
ALPHA	R	U	CALCULATION VARIABLE	DRAGI
A1	R	C	ANSWER MATRIX FOR DRAG INTEGRAL	DRAGI
B	R	C	POTENTIAL AT B (MAPPING CONSTANT)	DRAGI
C	R	C	POTENTIAL AT C (MAPPING CONSTANT)	DRAGI
CEINT	R	U	INTEGRAL FROM C TO E	DRAGI
CG	R	C	COSINE OF DIHEDRAL ANGLE	DRAGI
CONI	R	C	FACTOR FOR IMAJINARY PART OF COMPLEX DISTANCE	DRAGI
CONR	R	C	FACTOR FOR REAL PART OF COMPLEX DISTANCE	DRAGI
C1	R	C	CONSTANT THAT IS COMPUTER HARDWARE DEPENDENT	DRAGI
C2	R	C	CONSTANT THAT IS COMPUTER HARDWARE DEPENDENT	DRAGI
C3	R	C	CONSTANT THAT IS COMPUTER HARDWARE DEPENDENT	DRAGI
D	R	C	POTENTIAL AT D (MAPPING CONSTANT)	DRAGI
DATA	R	C	ARRAY FOR INPUT DATA	DRAGI
DEINT	R	U	INTEGRAL FROM D TO E	DRAGI
DFINT	R	U	INTEGRAL FROM D TO F	DRAGI
E	R	C	POTENTIAL AT E (MAPPING CONSTANT)	DRAGI
EFF	R	U	EFFICIENCY OF INDUCED DRAG (NOTATION K IN THEORY)	DRAGI
EFINT	R	U	INTEGRAL FROM E TO F	DRAGI
ETAIRG	R	C	INBOARD SPANWISE LOCATION (NONDIMENSIONAL)	DRAGI
ETAORG	R	C	OUTBOARD SPANWISE LOCATION (NONDIMENSIONAL)	DRAGI
F	R	C	POTENTIAL AT F (MAPPING CONSTANT)	DRAGI
FGINT	R	U	INTEGRAL FROM F TO G	DRAGI
FG1	R	U	FIRST HALF OF INTEGRAL FROM F TO G	DRAGI
FG2	R	U	SECOND HALF OF INTEGRAL FROM F TO G	DRAGI
G	R	C	POTENTIAL AT G (MAPPING CONSTANT)	DRAGI
GAMMA	R	C	DIHEDRAL ANGLE IN DEGREES	DRAGI
GHINT	R	U	INTEGRAL FROM G TO H	DRAGI
GH1	R	U	FIRST HALF OF INTEGRAL FROM G TO H	DRAGI
GH2	R	U	SECOND HALF OF INTEGRAL FROM G TO H	DRAGI

SYMBOLS USED IN SUBROUTINE DRAGI (DECK IDENTIFICATION 55VU)

GP	R	C	DIHEDRAL ANGLE DIVIDED BY PI	DRAGI
GPM2	R	C	GP - 0.5	DRAGI
GPP2	R	C	GP + 0.5	DRAGI
GT	R	U	CONSTANT OF REMOVABLE SINGULARITY	DRAGI
GT1	R	U	CONSTANT OF REMOVABLE SINGULARITY - FIRST HALF OF INT.	DRAGI
GT2	R	U	CONSTANT OF REMOVABLE SINGULARITY - SECOND HALF OF INT.	DRAGI
H	R	C	POTENTIAL AT H (MAPPING CONSTANT)	DRAGI
HIINT	R	U	INTEGRAL FROM H TO I	DRAGI
HI1	R	U	FIRST HALF OF INTEGRAL FROM H TO I	DRAGI
HI2	R	U	SECOND HALF OF INTEGRAL FROM H TO I	DRAGI
ILAST	I	C	FLAG FOR LAST ITERATION	DRAGI
ININT	I	C	NUMBER OF ORDINATES IN NUMERICAL INTEGRALS	DRAGI
ITEST	I	U	FLAG TO INDICATE MONOTONE INCREASING POTENTIALS	DRAGI
MODE	I	C	PROGRAM MODE OF OPERATION	DRAGI
NINT	I	U	CALCULATION VARIABLE	DRAGI
NTYPE	I	C	CONFIGURATION TYPE	DRAGI
OMGP	R	C	1.0 - GP	DRAGI
ONE	R	U	1.0	DRAGI
ORD	R	C	ORDINATE ARRAY FOR NUMERICAL INTEGRALS	DRAGI
PIREQ	R	C	INBOARD LENGTH OR DISTANCE (NONDIMENSIONAL)	DRAGI
POREQ	R	C	OUTBOARD LENGTH OR DISTANCE (NONDIMENSIONAL)	DRAGI
R1	R	C	SQUARE ROOT OF SUM OF SQUARES OF ERRORS IN REQD CONDITIONS	DRAGI
TAB	R	C	INTERPOLATION ARRAY FOR INDEPENDENT VARIABLE AND FUNCTION	DRAGI
TABLE	R	C	INTEGRATED VALUES OF ORDINATES AS A FUNCTION OF ABSCISSA	DRAGI
TWOBPI	R	U	2.0 DIVIDED BY PI	DRAGI

Subroutine NUM (UP, ALOW, SING, NPOINT, NTYPE, ANSWER, GT)(Deck 55VV). Subroutine NUM is called by subroutine DRAGI to perform quadratures just as subroutine NUMINT is called by subroutine GEOM. The description of subroutine NUMINT applies, word for word, to subroutine NUM.

```
                SUBROUTINE NUM     (UP,ALOW,SING,NPOINT,NTYPE,ANSWER,GT)
          C
          C     THIS SUBROUTINE SETS UP NUMERICAL INTEGRATION TO EVALUATE INDUCED
          C     DRAG, USING SIMPSON S RULE SUBROUTINE.
          C
000010          COMMON DATA(17), ABSC(50), ORD(50), TABLE(50), ALOAD(1000), AA(16)
                A , TAB(101), GP, GPM2, GPP2, CG, OMGP, CONR, CONI, ILAST, C2, C1
                B , C3, R1
          C
000010          EQUIVALENCE                (DATA( 1),       A) , (DATA( 2),       B)
                A , (DATA( 3),       C) , (DATA( 4),       D) , (DATA( 5),       E)
                B , (DATA( 6),       F) , (DATA( 7),       G) , (DATA( 8),       H)
                C , (DATA( 9),      AI) , (DATA(10),    MODE) , (DATA(11), NDUMMY)
                D , (DATA(12),   ININT) , (DATA(13), ETAIRQ) , (DATA(14), ETAORQ)
                E , (DATA(15),   PIREQ) , (DATA(16),  POREQ) , (DATA(17),  GAMMA)
          C
          C     ARITHMETIC STATEMENT FUNCTIONS
          C
          C     HIGHER LEVEL FUNCTIONS FOR CASE  NTYPE = 5
          C
000010          QA(Z) = (Z-C)*(Z-1.0)*SQRT(Z/((Z-A)*(G-Z)))
000033          QB(Z) = (Z-C)*(Z-1.0)*SQRT(Z/(Z-A))*(Z-D)**GPM2
          C
          C     HIGHER LEVEL FUNCTIONS FOR CASE 6
          C
000062          QF(Z) = (Z-C)*(Z-1.0)/SQRT(H-Z)*((E-Z)/(G-Z))**GP
000112          QG(Z) = (Z-C)*(Z-1.0)*SQRT(Z/((Z-A)*(Z-D)))
          C
          C     HIGHER LEVEL FUNCTIONS FOR THE CASE NTYPE = 7
          C
000134          QK(Z) = (Z-D)*(Z-1.0)*SQRT(Z*(G-Z)/((F-Z)*(H-Z)))
000157          QL(Z) = (Z-D)*(Z-1.0)*SQRT(Z/((Z-B)*(Z-C)))
          C
000201          GA(Z) = (Z-C)*(Z-1.0)*SQRT(Z/((D-Z)*(F-Z)))
000223          GD(Z) = (Z-C)*(Z-1.0)*SQRT(Z/((Z-B)*(F-Z)))
000245          GF(Z) = (Z-C)*(Z-1.0)*SQRT(Z/((Z-B)*(Z-D)))
000270          GA2(Z) = (Z-C)*(Z-F)*(Z-H)*SQRT (Z/((B-Z)*(Z-E)*(Z-1.0)*(Z-AI)))
000324          GA3(Z) = (Z-C)*(Z-F)*(Z-H)*SQRT (Z/((Z-A)*(Z-F)*(Z-1.0)*(Z-AI)))
000356          GEE(Z) = (Z-C)*(Z-F)*(Z-H)*SQRT (Z/((Z-A)*(Z-B)*(Z-1.0)*(Z-AI)))
000410          GGG(Z) = (Z-C)*(Z-F)*(Z-H)*SQRT (Z/((Z-A)*(Z-B)*(F-  Z)*(Z-AI)))
000442          GII(Z) = (Z-C)*(Z-F)*(Z-H)*SQRT (Z/((Z-A)*(Z-B)*(Z-E)*(Z-1.0)))
000474          GHA(Z) = (Z-B)*(Z-1.0)*SQRT (Z/((Z-D)*(F-Z)))
000516          GHD(Z) = (Z-B)*(Z-1.0)*SQRT (Z/((Z-A)*(F-Z)))
000540          GHF(Z) = (Z-B)*(Z-1.0)*SQRT (Z/((Z-A)*(Z-D)))
000562          GZA(Z) = (Z-C)*(Z-E)*(Z-1.0)*SQRT (Z/((Z-B)*(F-Z)*(H-Z)*(AI-Z)))
000614          GZH(Z) = (Z-C)*(Z-E)*(Z-1.0)*SQRT (Z/((A-Z)*(F-Z)*(H-Z)*(AI-Z)))
000646          GZF(Z) = (Z-C)*(Z-E)*(Z-1.0)*SQRT (Z/((Z-A)*(Z-B)*(H-Z)*(AI-Z)))
000700          GZH(Z) = (Z-C)*(Z-E)*(Z-1.0)*SQRT (Z/((Z-A)*(Z-B)*(Z-F)*(AI-Z)))
000732          GZI(Z) = (Z-C)*(Z-E)*(Z-1.0)*SQRT (Z/((Z-A)*(Z-B)*(Z-F)*(Z-H)))
000765          GXA(Z) = (Z-C)*(Z-1.0)*SQRT(Z/(Z-G))*(D-Z)**GPM2*(F-Z)**(-GP)
001023          GXB(Z) = QA(Z)*(F-Z)**(-GP)
001037          GXC(Z) = QA(Z)*(Z-D)**GPM2
001052          GXD(Z) = QB(Z)/SQRT(G-Z)
001071          GXE(Z) = QB(Z)*(Z-F)**(-GP)
001105          GVA(Z) = QF(Z)*SQRT(Z/(Z-D))
001124          GVB(Z) = QF(Z)*SQRT(Z/(Z-A))
001143          GVC(Z) = QG(Z)/(SQRT(H-Z)*(G-Z)**GP)
```

```
001170          GVD(Z)  = QG(Z)/SQRT(H-Z)*(Z-E)**GP                      55VV0580
001215          GVE(Z)  = QG(Z)*((Z-E)/(Z-G))**GP                       55VV0590
001232          GTA(Z)  = QK(Z)/SQRT(G-Z)                               55VV0600
001251          GTB(Z)  = QK(Z)/SQRT(Z-B)                               55VV0610
001270          GTC(Z)  = QL(Z)*SQRT((G-Z)/(H-Z))                       55VV0620
001310          GTD(Z)  = QL(Z)*SQRT((Z-G)/(Z-F))                       55VV0630
                C                                                       55VV0640
001331          DUM = (UP - ALOW) / FLOAT(NPOINT - 1)                   55VV0650
001335          IF (DUM .LT. C1) GO TO 501                              55VV0660
001337          IF (NTYPE .EQ.  1) GT = GA(SING)                        55VV0670
001343          IF (NTYPE .EQ.  2) GT = GD(SING)                        55VV0680
001347          IF (NTYPE .EQ.  3) GT = GF(SING)                        55VV0690
001353          IF (NTYPE .EQ. 10) GT = GA2(SING)                       55VV0700
001357          IF (NTYPE .EQ. 11) GT = GA3(SING)                       55VV0710
001363          IF (NTYPE .EQ. 12) GT = GEE(SING)                       55VV0720
001367          IF (NTYPE .EQ. 13) GT = GGG(SING)                       55VV0730
001373          IF (NTYPE .EQ. 14) GT = GII(SING)                       55VV0740
001377          IF (NTYPE .EQ. 25) GT = GHA(SING)                       55VV0750
001403          IF (NTYPE .EQ. 26) GT = GHD(SING)                       55VV0760
001407          IF (NTYPE .EQ. 27) GT = GHF(SING)                       55VV0770
001413          IF (NTYPE .EQ. 34) GT = GZA(SING)                       55VV0780
001417          IF (NTYPE .EQ. 35) GT = GZB(SING)                       55VV0790
001423          IF (NTYPE .EQ. 36) GT = GZF(SING)                       55VV0800
001427          IF (NTYPE .EQ. 38) GT = GZH(SING)                       55VV0810
001433          IF (NTYPE .EQ. 39) GT = GZI(SING)                       55VV0820
001437          IF (NTYPE .EQ. 50) GT = GXA(SING)                       55VV0830
001443          IF (NTYPE .EQ. 51) GT = GXB(SING)                       55VV0840
001447          IF (NTYPE .EQ. 52) GT = GXC(SING)                       55VV0850
001453          IF (NTYPE .EQ. 53) GT = GXD(SING)                       55VV0860
001457          IF (NTYPE .EQ. 54) GT = GXE(SING)                       55VV0870
001463          IF (NTYPE .EQ. 63) GT = GVA(SING)                       55VV0880
001467          IF (NTYPE .EQ. 64) GT = GVB(SING)                       55VV0890
001473          IF (NTYPE .EQ. 65) GT = GVC(SING)                       55VV0900
001477          IF (NTYPE .EQ. 66) GT = GVD(SING)                       55VV0910
001503          IF (NTYPE .EQ. 67) GT = GVD(SING)                       55VV0920
001507          IF (NTYPE .EQ. 68) GT = GVE(SING)                       55VV0930
001513          IF (NTYPE .EQ. 79) GT = GTA(SING)                       55VV0940
001517          IF (NTYPE .EQ. 80) GT = GTB(SING)                       55VV0950
001523          IF (NTYPE .EQ. 81) GT = GTC(SING)                       55VV0960
001527          IF (NTYPE .EQ. 82) GT = GTD(SING)                       55VV0970
001533          FF2 = 0.005                                             55VV0980
001535          FF  = FF2 + FF2                                         55VV0990
001536          IF ((SING - ALOW) .GT. (UP - SING)) GO TO 401           55VV1000
001543          ABSC(1) = ALOW                                          55VV1010
001543          DO 400 I=2,NPOINT                                       55VV1020
001545          ABSC(I) = ABSC(I-1) + FF * (UP - ABSC(I-1)) / FLOAT(NPOINT-I+1)  55VV1030
001556          FF = FF + FF                                            55VV1040
001557          IF (FF .GT. 1.0) FF = 1.0                               55VV1050
001562      400 CONTINUE                                                55VV1060
001565          GO TO 403                                               55VV1070
001565      401 ABSC(NPOINT) = UP                                       55VV1080
001570          DO 402 I=2,NPOINT                                       55VV1090
001571          J = NPOINT - I + 1                                      55VV1100
001573          K = J + 1                                               55VV1110
001575          ABSC(J) = ABSC(K) - FF * (ABSC(K) - ALOW) / FLOAT(J)    55VV1120
001605          FF = FF + FF                                            55VV1130
001606          IF (FF .GT. 1.0) FF = 1.0                               55VV1140
001611      402 CONTINUE                                                55VV1150
```

```
001614      403 DO 500 I=1,NPOINT                                                  55VV1160
001616          AB = ABSC(I)                                                       55VV1170
001620          IF (ABS(AB - SING) .LT. FF2 * DUM) GO TO 499                       55VV1180
001625          GO TO (1,2,3,4,4,4,4,4,4,10,11,12,13,14,4,4,4,4,4,4,4,4,4,4,25,26 55VV1190
               A  ,27,4,4,4,4,4,4,34,35,36,36,38,39,4,4,4,4,4,4,4,4,4,50,51,52, 55VV1200
               B   53,54,4,4,4,4,4,4,4,4,63,64,65,66,67,68,4,4,4,4,4,4,4,4,4,4   55VV1210
               C  ,79,80,81,82,4,4,4,4,4,4,4,4,4                                  55VV1220
               A   ) , NTYPE                                                      55VV1230
001764        4 WRITE (6,5) NTYPE                                                 55VV1240
001772        5 FORMAT (1H0, 35H SUBROUTINE NUM CALLED FOR NTYPE =  ,I7//)        55VV1250
001772          GO TO 501                                                         55VV1260
001776        1 ORD(I) = (GA(AB) - GT) / SQRT(B - AB)                             55VV1270
002011          GO TO 500                                                         55VV1280
002015        2 ORD(I) = (GD(AB) - GT) / SQRT (AB - D)                            55VV1290
002030          GO TO 500                                                         55VV1300
002034        3 ORD(I) = (GF(AB) - GT) / SQRT (F - AB)                            55VV1310
002047          GO TO 500                                                         55VV1320
002053       10 ORD(I) = (GA2(AB) - GT) / SQRT (AB - A)                           55VV1330
002066          GO TO 500                                                         55VV1340
002072       11 ORD(I) = (GA3(AB) - GT) / SQRT (B - AB)                           55VV1350
002105          GO TO 500                                                         55VV1360
002111       12 ORD(I) = (GEE(AB) - GT) / SQRT (E - AB)                           55VV1370
002124          GO TO 500                                                         55VV1380
002130       13 ORD(I) = (GGG(AB) - GT) / SQRT (AB - 1.0)                         55VV1390
002143          GO TO 500                                                         55VV1400
002147       14 ORD(I) = (GII(AB) - GT) / SQRT (AI - AB)                          55VV1410
002162          GO TO 500                                                         55VV1420
002166       25 ORD(I) = (GHA(AB) - GT) / SQRT (AB - A)                           55VV1430
002201          GO TO 500                                                         55VV1440
002205       26 ORD(I) = (GHD(AB) - GT) / SQRT (AB - D)                           55VV1450
002220          GO TO 500                                                         55VV1460
002224       27 ORD(I) = (GHF(AB) - GT) / SQRT (F - AB)                           55VV1470
002237          GO TO 500                                                         55VV1480
002243       34 ORD(I) = (GZA(AB) - GT) / SQRT (AB - A)                           55VV1490
002256          GO TO 500                                                         55VV1500
002262       35 ORD(I) = (GZB(AB) - GT) / SQRT (B - AB)                           55VV1510
002275          GO TO 500                                                         55VV1520
002301       36 ORD(I) = (GZF(AB) - GT) / SQRT (F - AB)                           55VV1530
002314          GO TO 500                                                         55VV1540
002320       38 ORD(I) = (GZH(AB) - GT) / SQRT (AB - H)                           55VV1550
002333          GO TO 500                                                         55VV1560
002337       39 ORD(I) = (GZI(AB) - GT) / SQRT (AI - AB)                          55VV1570
002352          GO TO 500                                                         55VV1580
002356       50 ORD(I) = (GXA(AB)- GT) / SQRT (AB - A)                            55VV1590
002371          GO TO 500                                                         55VV1600
002375       51 ORD(I) = (GXB(AB)- GT) *(AB-D) ** GPM2                            55VV1610
002411          GO TO 500                                                         55VV1620
002411       52 ORD(I) = (GXC(AB)- GT) * (F-AB)** (-GP)                           55VV1630
002426          GO TO 500                                                         55VV1640
002426       53 ORD(I) = (GXD(AB)- GT) *(AB-F) ** (-GP)                           55VV1650
002443          GO TO 500                                                         55VV1660
002443       54 ORD(I) = (GXE(AB)- GT) / SQRT (G - AB)                            55VV1670
002456          GO TO 500                                                         55VV1680
002462       63 ORD(I) = (GVA(AB) - GT) / SQRT (AB - A)                           55VV1690
002475          GO TO 500                                                         55VV1700
002501       64 IF (ABS(AB-E) .LT. C1) GO TO 499                                  55VV1710
002506          ORD(I) = (GVB(AB) - GT) / SQRT (AB - D)                           55VV1720
002520          GO TO 500                                                         55VV1730
```

```
002524      65 ORD(I) = (GVC(AB) - GT) * (AB - E) ** GP                    55VV1740
002540         GO TO 500                                                   55VV1750
002540      66 ORD(I) = (GVD(AB) - GT) / (G - AB) ** GP                    55VV1760
002554         GO TO 500                                                   55VV1770
002554      67 ORD(I) = (GVD(AB) - GT) / (AB - G) ** GP                    55VV1780
002570         GO TO 500                                                   55VV1790
002570      68 ORD(I) = (GVE(AB) - GT) / SQRT (H - AB)                     55VV1800
002603         GO TO 500                                                   55VV1810
002607      79 ORD(I) = (GTA(AB) - GT) / SQRT (B - AB)                     55VV1820
002622         GO TO 500                                                   55VV1830
002626      80 ORD(I) = (GTB(AB) - GT) / SQRT(AB - C)                      55VV1840
002641         GO TO 500                                                   55VV1850
002645      81 ORD(I) = (GTC(AB) - GT) / SQRT(F - AB)                      55VV1860
002660         GO TO 500                                                   55VV1870
002664      82 ORD(I) = (GTD(AB) - GT) / SQRT(H - AB)                      55VV1880
002677         GO TO 500                                                   55VV1890
002703     499 ORD(I) = 0.0                                                55VV1900
002705     500 CONTINUE                                                    55VV1910
002710         DUMMY = ABSC(1)                                             55VV1920
002711         DO 502 I=1,NPOINT                                           55VV1930
002713     502 ABSC(I) = ABSC(I) - DUMMY                                   55VV1940
002720         CALL ITS1 (ABSC, ORD, TABLE, NPOINT)                        55VV1950
002722         ANSWER = TABLE(NPOINT)                                      55VV1960
002730         RETURN                                                      55VV1970
002731     501 ANSWER = 0.0                                                55VV1980
002732         RETURN                                                      55VV1990
002732         END                                                         55VV2000
```

SUBPROGRAM LENGTH
004377

FUNCTION ASSIGNMENTS

GA	-	000202	GA2	-	000271	GA3	-	000325	GD	-	000224
GEE	-	000357	GF	-	000246	GGG	-	000411	GHA	-	000475
GHD	-	000517	GHF	-	000541	GII	-	000443	GTA	-	001233
GTB	-	001252	GTC	-	001271	GTD	-	001311	GVA	-	001106
GVB	-	001125	GVC	-	001144	GVD	-	001171	GVE	-	001216
GXA	-	000766	GXB	-	001024	GXC	-	001040	GXD	-	001053
GXE	-	001072	GZA	-	000563	GZB	-	000615	GZF	-	000647
GZH	-	000701	GZI	-	000733	QA	-	000012	QB	-	000034
QF	-	000063	QG	-	000113	QK	-	000135	QL	-	000160

STATEMENT ASSIGNMENTS

1	-	001776	2	-	002015	3	-	002034	4	-	001764
5	-	002740	10	-	002053	11	-	002072	12	-	002111
13	-	002130	14	-	002147	25	-	002166	26	-	002205
27	-	002224	34	-	002243	35	-	002262	36	-	002301
38	-	002320	39	-	002337	50	-	002356	51	-	002375
52	-	002411	53	-	002426	54	-	002443	63	-	002462
64	-	002501	65	-	002524	66	-	002540	67	-	002554
68	-	002570	79	-	002607	80	-	002626	81	-	002645
82	-	002664	400	-	001562	401	-	001565	402	-	001611
403	-	001614	499	-	002703	500	-	002705	501	-	002731
502	-	002714									

BLOCK NAMES AND LENGTHS
- 002420

VARIABLE ASSIGNMENTS

A	-	000000C01	AA	-	002217C01	AB	-	004375	ABSC	-	000021C01
AI	-	000010C01	ALOAD	-	000247C01	B	-	000001C01	C	-	000002C01
C1	-	002415C01	D	-	000003C01	DATA	-	000000C01	DUM	-	004367
DUMMY	-	004376	E	-	000004C01	F	-	000005C01	FF	-	004371
FF2	-	004370	G	-	000006C01	GP	-	002404C01	GPM2	-	002405C01
GT	-	000000	H	-	000007C01	I	-	004372	J	-	004373
K	-	004374	ORD	-	000103C01	TAB	-	002237C01	TABLE	-	000165C01

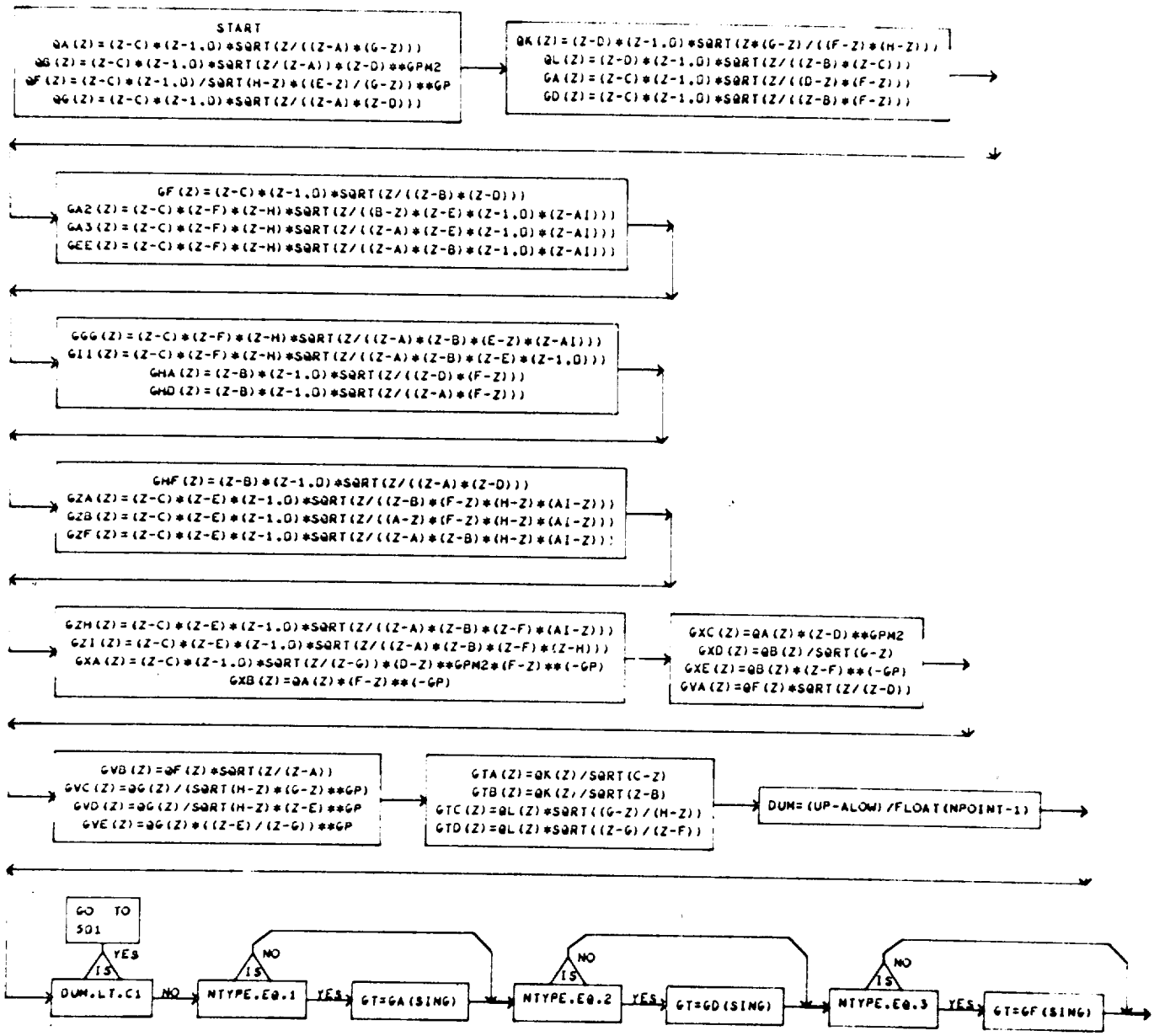
START OF CONSTANTS
002734

START OF TEMPORARIES
002747

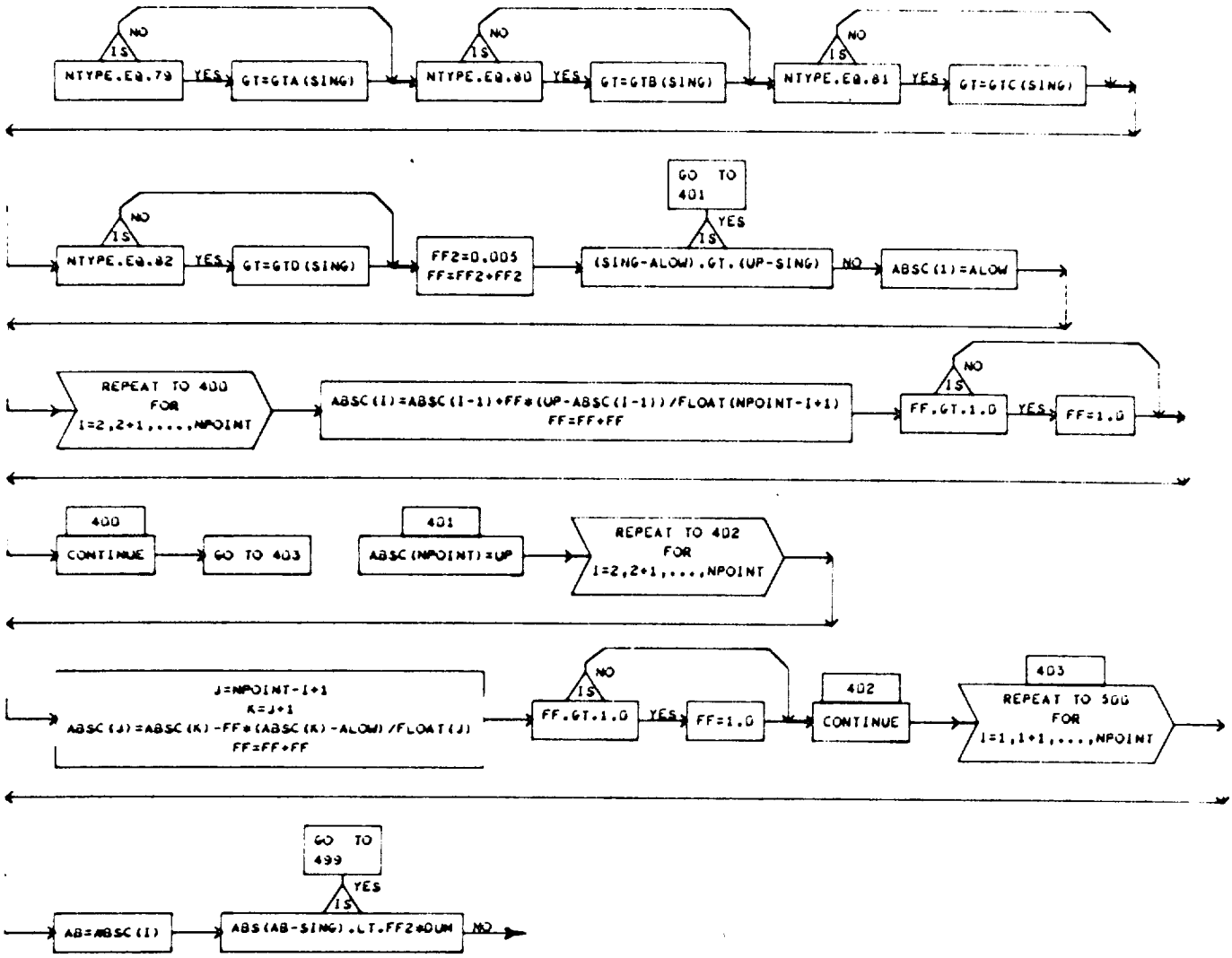
START OF INDIRECTS
004326

UNUSED COMPILER SPACE
033300

```
SUBROUTINE NUM     (UP,ALOW,SING,NPOINT,NTYPE,ANSWER,GT)
```

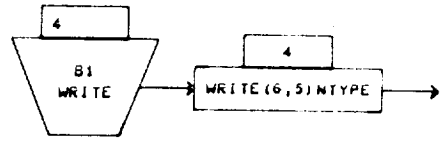


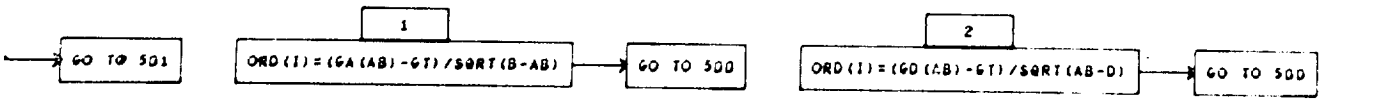
NTYPE.EQ.10
YES
GT=GA2(SING)
NTYPE.EQ.11
YES
GT=GA3(SING)
NTYPE.EQ.12
YES
GT=GEE(SING)
NTYPE.EQ.13
YES
GT=GGG(SING)
NTYPE.EQ.14
YES
GT=GII(SING)
NTYPE.EQ.25
YES
GT=GHA(SING)
NTYPE.EQ.26
YES
GT=GHD(SING)
NTYPE.EQ.27
YES
GT=GHF(SING)
NTYPE.EQ.34
YES
GT=GZA(SING)
NTYPE.EQ.35
YES
GT=GZB(SING)
NTYPE.EQ.36
YES
GT=GZF(SING)
NTYPE.EQ.38
YES
GT=GZH(SING)
NTYPE.EQ.39
YES
GT=GZI(SING)
NTYPE.EQ.50
YES
GT=GXA(SING)
NTYPE.EQ.51
YES
GT=GXB(SING)
NTYPE.EQ.52
YES
GT=GXC(SING)
NTYPE.EQ.53
YES
GT=GXD(SING)
NTYPE.EQ.54
YES
GT=GXE(SING)
NTYPE.EQ.63
YES
GT=GVA(SING)
NTYPE.EQ.64
YES
GT=GVB(SING)
NTYPE.EQ.65
YES
GT=GVC(SING)
NTYPE.EQ.66
YES
GT=GVD(SING)
NTYPE.EQ.67
YES
GT=GVD(SING)
NTYPE.EQ.68
YES
GT=GVE(SING)
NO
IS


NTYPE.EQ.79
IS
NO
YES
GT=GTA(SING)
NTYPE.EQ.80
GT=GTB(SING)
NTYPE.EQ.81
GT=GTC(SING)
NTYPE.EQ.82
GT=GTD(SING)
FF2=0.005
FF=FF2+FF2
(SING-ALOW).GT.(UP-SING)
GO TO 401
ABSC(1)=ALOW
REPEAT TO 400 FOR I=2,2+1,...,NPOINT
ABSC(I)=ABSC(I-1)+FF*(UP-ABSC(I-1))/FLOAT(NPOINT-I+1)
FF=FF+FF
FF.GT.1.0
FF=1.0
400
CONTINUE
GO TO 403
401
ABSC(NPOINT)=UP
REPEAT TO 402 FOR I=2,2+1,...,NPOINT
J=NPOINT-I+1
K=J+1
ABSC(J)=ABSC(K)-FF*(ABSC(K)-ALOW)/FLOAT(J)
FF=FF+FF
402
403
REPEAT TO 500 FOR I=1,1+1,...,NPOINT
AB=ABSC(I)
ABS(AB-SING).LT.FF2*DUM
GO TO 499

COMPUTED GO TO

IF THE VALUE OF NTYPE IS	TRANSFER TO STATEMENT
1	1
2	2
3	3
4	4
5	4
6	4
7	4
8	4
9	4
10	10
11	11
12	12
13	13
14	14
15	4
16	4
17	4
18	4
19	4
20	4
21	4
22	4
23	4
24	4
25	25
26	26
27	27
28	4
29	4
30	4
31	4
32	4
33	4
34	34
35	35
36	36
37	36
38	38
39	39
40	4
41	4
42	4
43	4
44	4
45	4
46	4
47	4
48	4
49	4
50	50
51	51
52	52
53	53
54	54
55	4
56	4
57	4
58	4
59	4
60	4
61	4
62	4
63	63
64	64
65	65
66	66
67	67
68	68
69	4
70	4
71	4
72	4
73	4
74	4
75	4
76	4
77	4
78	4
79	79
80	80
81	81
82	82
83	4
84	4
85	4
86	4
87	4
88	4
89	4
90	4
91	4





3

```
ORD(I)=(GF(AB)-GT)/SQRT(F-AB)
```

→ GO TO 500

10

```
ORD(I)=(GA2(AB)-GT)/SQRT(AB-A)
```

→ GO TO 500

11

```
ORD(I)=(GA3(AB)-GT)/SQRT(B-AB)
```

→ GO TO 500

12

```
ORD(I)=(GEE(AB)-GT)/SQRT(E-AB)
```

→ GO TO 500

13

```
ORD(I)=(GGG(AB)-GT)/SQRT(AB-1.0)
```

→ GO TO 500

14

```
ORD(I)=(GII(AB)-GT)/SQRT(AI-AB)
```

→ GO TO 500

25

```
ORD(I)=(GHA(AB)-GT)/SQRT(AB-A)
```

→ GO TO 500

26

```
ORD(I)=(GHD(AB)-GT)/SQRT(AD-D)
```

→ GO TO 500

27

```
ORD(I)=(GHF(AB)-GT)/SQRT(F-AB)
```

→ GO TO 500

34

```
ORD(I)=(GZA(AB)-GT)/SQRT(AB-A)
```

→ GO TO 500

35

```
ORD(I)=(GZB(AB)-GT)/SQRT(B-AB)
```

→ GO TO 500

36

```
ORD(I)=(GZF(AB)-GT)/SQRT(F-AB)
```

→ GO TO 500

38

```
ORD(I)=(GZH(AB)-GT)/SQRT(AB-H)
```

→ GO TO 500

39

```
ORD(I)=(GZI(AB)-GT)/SQRT(AI-AB)
```

→ GO TO 500

50

```
ORD(I)=(GXA(AB)-GT)/SQRT(AB-A)
```

→ GO TO 500

51

```
ORD(I)=(GXB(AB)-GT)*(AB-D)**GPM2
```

→ GO TO 500

52

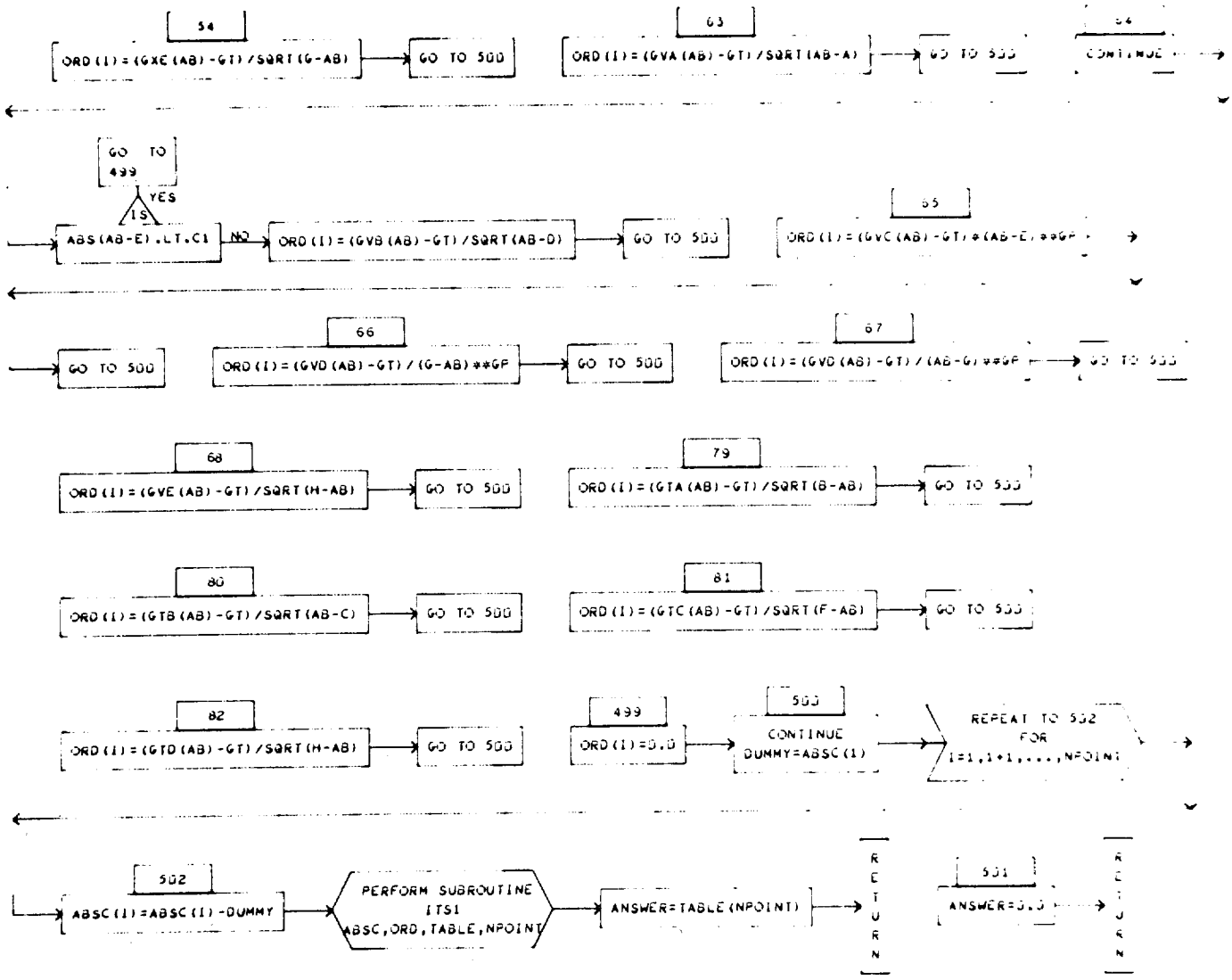
```
ORD(I)=(GXC(AB)-GT)*(F-AB)**(-GP)
```

→ GO TO 500

53

```
ORD(I)=(GXD(AB)-GT)*(AB-F)**(-GP)
```

→ GO TO 500


54
ORD(I)=(GXE(AB)-GT)/SQRT(G-AB)
GO TO 500
63
ORD(I)=(GVA(AB)-GT)/SQRT(AB-A)
GO TO 500
64
CONTINUE
GO TO 499
IS
YES
ABS(AB-E).LT.C1
NO
ORD(I)=(GVB(AB)-GT)/SQRT(AB-D)
GO TO 500
65
ORD(I)=(GVC(AB)-GT)*(AB-E)**GP
GO TO 500
66
ORD(I)=(GVD(AB)-GT)/(G-AB)**GP
GO TO 500
67
ORD(I)=(GVD(AB)-GT)/(AB-G)**GP
GO TO 500
68
ORD(I)=(GVE(AB)-GT)/SQRT(H-AB)
GO TO 500
79
ORD(I)=(GTA(AB)-GT)/SQRT(B-AB)
GO TO 500
80
ORD(I)=(GTB(AB)-GT)/SQRT(AB-C)
GO TO 500
81
ORD(I)=(GTC(AB)-GT)/SQRT(F-AB)
GO TO 500
82
ORD(I)=(GTD(AB)-GT)/SQRT(H-AB)
GO TO 500
499
ORD(I)=0.0
500
CONTINUE
DUMMY=ABSC(1)
REPEAT TO 502
FOR
I=1,1+1,...,NPOINT
502
ABSC(I)=ABSC(I)-DUMMY
PERFORM SUBROUTINE
ITS1
ABSC,ORD,TABLE,NPOINT
ANSWER=TABLE(NPOINT)
RETURN
501
ANSWER=0.0
RETURN

SYMBOLS USED IN SUBROUTINE NUM (DECK IDENTIFICATION 55\ /)

A	R	C	POTENTIAL AT A (MAPPING CONSTANT)	NUM
AA	R	C	ARRAY FOR RESULTS OF GEOMETRY AND DRAG INTEGRALS	NUM
AB	R	U	CURRENT VALUE OF ABSCISSA	NUM
ABSC	R	C	ABSCISSA ARRAY FOR NUMERICAL INTEGRALS	NUM
AI	R	C	POTENTIAL AT I (MAPPING CONSTANT)	NUM
ALOAL	R	C	STORAGE ARRAY FOR POTENTIAL IN THE REAL (PHYSICAL) PLANE	NUM
ALOW	R	U	POTENTIAL AT THE LOWER INTEGRATION LIMIT	NUM
ANSWER	R	U	RESULT OF NUMERICAL INTEGRAL	NUM
B	R	C	POTENTIAL AT B (MAPPING CONSTANT)	NUM
C	R	C	POTENTIAL AT C (MAPPING CONSTANT)	NUM
CG	R	C	COSINE OF DIHEDRAL ANGLE	NUM
CONI	R	C	FACTOR FOR IMAJINARY PART OF COMPLEX DISTANCE	NUM
CONR	R	C	FACTOR FOR REAL PART OF COMPLEX DISTANCE	NUM
C1	R	C	CONSTANT THAT IS COMPUTER HARDWARE DEPENDENT	NUM
C2	R	C	CONSTANT THAT IS COMPUTER HARDWARE DEPENDENT	NUM
C3	R	C	CONSTANT THAT IS COMPUTER HARDWARE DEPENDENT	NUM
D	R	C	POTENTIAL AT D (MAPPING CONSTANT)	NUM
DATA	R	C	ARRAY FOR INPUT DATA	NUM
DUM	R	U	CALCULATION VARIABLE	NUM
DUMMY	R	U	DUMMY VARIABLE	NUM
E	R	C	POTENTIAL AT E (MAPPING CONSTANT)	NUM
ETAIBQ	R	C	INBOARD SPANWISE LOCATION (NONDIMENSIONAL)	NUM
ETAOBQ	R	C	OUTBOARD SPANWISE LOCATION (NONDIMENSIONAL)	NUM
F	R	C	POTENTIAL AT F (MAPPING CONSTANT)	NUM
FF	R	U	CALCULATION VARIABLE	NUM
FF2	R	U	CALCULATION VARIABLE	NUM
G	R	C	POTENTIAL AT G (MAPPING CONSTANT)	NUM
GAMMA	R	C	DIHEDRAL ANGLE IN DEGREES	NUM
GP	R	C	DIHEDRAL ANGLE DIVIDED BY PI	NUM
GPM2	R	C	GP - 0.5	NUM
GPP2	R	C	GP + 0.5	NUM
GT	R	U	CONSTANT OF REMOVABLE SINGULARITY	NUM
H	R	C	POTENTIAL AT H (MAPPING CONSTANT)	NUM
I	I	U	DO LOOP INDEX	NUM
ILAST	I	C	FLAG FOR LAST ITERATION	NUM
ININT	I	C	NUMBER OF ORDINATES IN NUMERICAL INTEGRALS	NUM
J	I	U	CALCULATION VARIABLE	NUM
K	I	U	CALCULATION VARIABLE	NUM

SYMBOLS USED IN SUBROUTINE NUM (DECK IDENTIFICATION 55VV)

MODE	I	C	PROGRAM MODE OF OPERATION	NUM
NDUMMY	I	U	UNUSED DUMMY VARIABLE	NUM
NPOINT	R	U	NUMBER OF ORDINATES IN NUMERICAL INTEGRAL	NUM
NTYPE	R	U	NUMERICAL INTEGRAL TYPE (INTEGRAL NUMBER)	NUM
OMGP	R	C	1.0 - GP	NUM
ORD	R	C	ORDINATE ARRAY FOR NUMERICAL INTEGRALS	NUM
PIREQ	R	C	INBOARD LENGTH OR DISTANCE (NONDIMENSIONAL)	NUM
POREQ	R	C	OUTBOARD LENGTH OR DISTANCE (NONDIMENSIONAL)	NUM
R1	R	C	SQUARE ROOT OF SUM OF SQUARES OF ERRORS IN REQD CONDITIONS	NUM
SING	R	U	POTENTIAL AT THE SINGULARITY	NUM
TAB	R	C	INTERPOLATION ARRAY FOR INDEPENDENT VARIABLE AND FUNCTION	NUM
TABLE	R	C	INTEGRATED VALUES OF ORDINATES AS A FUNCTION OF ABSCISSA	NUM
UP	R	U	POTENTIAL AT THE UPPER INTEGRATION LIMIT	NUM

Subroutine LOAD (Deck 55VW). This subroutine computes the loading on each configuration to produce minimum induced drag. The data in the ALOAD array (see description of subroutine STORE, Deck 55VS) are shifted so that pylon and span coordinates increase with pylon length or outboard spanwise location. Subroutine LØAD uses subroutine INTEST, TABLOD, and INTERP to actually compute the loading, and then writes out the results.

```
                SUBROUTINE LOAD                                                 55VW0010
          C                                                                     55VW0020
          C     THIS SUBROUTINE INTERPOLATES IN THE ALOAD ARRAY TO DETERMINE THE 55VW0030
          C     LOAD DISTRIBUTION TO PRODUCE MINIMUM INDUCED DRAG.              55VW0040
          C                                                                     55VW0050
000002          COMMON DATA(17), ABSC(50), ORD(50), TABLE(50), ALOAD(1000), AA(16)55VW0060
               A , TAB(101), GP, GPM2, GPP2, CG, OMGP, CONR, CONI, ILAST, C2, C3 55VW0070
               B , C1, R1                                                       55VW0080
          C                                                                     55VW0090
000002          EQUIVALENCE              (DATA( 1),      A) , (DATA( 2),      B) 55VW0100
               A , (DATA( 3),      C) , (DATA( 4),      D) , (DATA( 5),      E) 55VW0110
               B , (DATA( 6),      F) , (DATA( 7),      G) , (DATA( 8),      H) 55VW0120
               C , (DATA( 9),     AI) , (DATA(10),   MODE) , (DATA(11),  NTYPE) 55VW0130
               D , (DATA(12),  ININT) , (DATA(13), ETAIRQ) , (DATA(14), ETAORQ) 55VW0140
               E , (DATA(15),  PIREQ) , (DATA(16),  POREQ) , (DATA(17),  GAMMA) 55VW0150
000002          EQUIVALENCE (TAB,NN)                                            55VW0160
          C                                                                     55VW0170
000002          LOGICAL LESS                                                    55VW0180
          C                                                                     55VW0190
000002          NN = ININT                                                      55VW0200
          C                                                                     55VW0210
          C     TRANSFER TO CODING FOR SPECIFIC CONFIGURATION                   55VW0220
          C                                                                     55VW0230
000004          GO TO ( 100, 200, 300, 400, 500, 600, 700 ) , NTYPE             55VW0240
000017      100 CONTINUE                                                        55VW0250
          C                                                                     55VW0260
          C     WING WITH ONE PYLON                                             55VW0270
          C                                                                     55VW0280
          C     CORRECT ALGEBRAIC SIGNS IN TREFFTZ PLANE SO THAT COORDINATES ALONG55VW0290
          C     PYLON AND SPAN INCREASE WITH PYLON LENGTH OR OUTBOARD SPAN      55VW0300
          C     LOCATION.                                                       55VW0310
          C                                                                     55VW0320
000017          DO 101 I=1,ININT                                                55VW0330
000021          ALOAD(I+150) = ALOAD(I+150) + ALOAD(ININT+50)                   55VW0340
000024          ALOAD(I+350) = (-ALOAD(I+350))                                  55VW0350
000025      101 ALOAD(I+450) = (-ALOAD(I+450))                                  55VW0360
000031          DO 102 I=1,ININT                                                55VW0370
000032          ALOAD(I+350) = ALOAD(ININT+350) - ALOAD(I+350)                  55VW0380
000035          ALOAD(I+550) = ALOAD(ININT+550) - ALOAD(I+550)                  55VW0390
000037      102 ALOAD(I+450) = ALOAD(I+450) + ALOAD(ININT+150)                  55VW0400
000042          SS = ALOAD(551) - ALOAD(ININT+550)                              55VW0410
000044          PL = ALOAD(ININT+250) - ALOAD(251)                              55VW0420
          C                                                                     55VW0430
          C     SET UP TAB ARRAY FOR INTERPOLATION TO DETERMINE LOADING ON WING 55VW0440
          C                                                                     55VW0450
000046          CALL INTEST (450, 551)                                          55VW0460
000050          CALL TABLOD (551)                                               55VW0470
000052          CALL INTERP (   0, SS)                                          55VW0480
000054          ALIFT = TABLE(ININT)                                            55VW0490
000057          CALL INTERP ( 100, SS)                                          55VW0500
000061          ALIFT = TABLE(ININT) + ALIFT                                    55VW0510
000064          CALL INTERP ( 400, SS)                                          55VW0520
000066          ALIFT = TABLE(ININT) + ALIFT                                    55VW0530
          C                                                                     55VW0540
          C     DETERMINE LOADING ON PYLON                                      55VW0550
          C                                                                     55VW0560
000071          CALL INTEST (250, 351)                                          55VW0570
```

```
000073          CALL TABLOD (351)                                                55VW0580
000075          CALL INTERP ( 200, PL)                                           55VW0590
000077          BLIFT = TABLE(ININT)                                             55VW0600
000102          DUM1  = BLIFT / ALIFT * PIREQ                                    55VW0610
000104          WRITE (6,103) DUM1                                               55VW0620
000112      103 FORMAT (1H1, 26X, 21H WING WITH ONE PYLON  // 8X, 45H RATIO OF LOA55VW0630
               ADING ON PYLON TO LOAD ON WING =  ,F14.8// 8X, 4HWING , 21X,      55VW0640
               B 10HELLIPTICAL / 6X, 8HSPANWISE , 2(8X, 6HWING  ), 7X, 5HPYLON ,  55VW0650
               C 10X, 5HPYLON / 5X, 10HCOORDINATE , 6X, 7HLOADING , 7X, 7HLOADING 55VW0660
               D , 5X, 10HCOORDINATE , 6X, 7HLOADING //)                         55VW0670
000112          DO 104 I=1,ININT                                                 55VW0680
000114          ALOAD(I     ) = ALOAD(I     ) / ALIFT                            55VW0690
000117          ALOAD(I+200) = ALOAD(I+200) / BLIFT                              55VW0700
000120          DUM = FLOAD(ALOAD(I+50))                                         55VW0710
000123      104 WRITE (6,105) ALOAD(I+50), ALOAD(I), DUM, ALOAD(I+250),          55VW0720
               A                ALOAD(I+200)                                    55VW0730
000144      105 FORMAT (1H , 5F14.8)                                             55VW0740
000144          DO 106 I=1,ININT                                                 55VW0750
000145          ALOAD(I+100) = ALOAD(I+100) / ALIFT                              55VW0760
000147          DUM = FLOAD(ALOAD(I+150))                                        55VW0770
000152      106 WRITE (6,107) ALOAD(I+150), ALOAD(I+100), DUM                    55VW0780
000166      107 FORMAT (1H , 3F14.8)                                             55VW0790
000166          DO 108 I=1,ININT                                                 55VW0800
000167          ALOAD(I+400) = ALOAD(I+400) / ALIFT                              55VW0810
000171          DUM = FLOAD(ALOAD(I+450))                                        55VW0820
000174      108 WRITE (6,107) ALOAD(I+450), ALOAD(I+400), DUM                    55VW0830
000210          RETURN                                                           55VW0840
000210      200 CONTINUE                                                         55VW0850
        C                                                                        55VW0860
        C       WING WITH TWO PYLONS BELOW                                       55VW0870
        C                                                                        55VW0880
000210          DO 201 I=1,ININT                                                 55VW0890
000212          ALOAD(I+ 50) = (-ALOAD(I+ 50))                                   55VW0900
000214          ALOAD(I+150) = (-ALOAD(I+150))                                   55VW0910
000215          ALOAD(I+250) = (-ALOAD(I+250))                                   55VW0920
000216          ALOAD(I+750) = (-ALOAD(I+750))                                   55VW0930
000220      201 ALOAD(I+850) = (-ALOAD(I+850))                                   55VW0940
000223          DO 202 I=1,ININT                                                 55VW0950
000224          ALOAD(I+150) = ALOAD(I+150) + ALOAD(ININT+50)                    55VW0960
000227          ALOAD(I+350) = ALOAD(ININT+350) - ALOAD(I+350)                   55VW0970
000231          ALOAD(I+750) = ALOAD(ININT+750) - ALOAD(I+750)                   55VW0980
000233      202 ALOAD(I+950) = ALOAD(ININT+950) - ALOAD(I+950)                   55VW0990
000236          DO 203 I=1,ININT                                                 55VW1000
000240      203 ALOAD(I+450) = ALOAD(I+450) + ALOAD(ININT+150)                   55VW1010
000245          DO 204 I=1,ININT                                                 55VW1020
000246      204 ALOAD(I+550) = ALOAD(I+550) + ALOAD(ININT+450)                   55VW1030
000253          DO 205 I=1,ININT                                                 55VW1040
000254      205 ALOAD(I+850) = ALOAD(I+850) + ALOAD(ININT+550)                   55VW1050
000261          SS = ALOAD(951) - ALOAD(ININT+950)                               55VW1060
000263          PI = ALOAD(ININT+250) - ALOAD(251)                               55VW1070
000265          PO = ALOAD(ININT+650) - ALOAD(651)                               55VW1080
        C                                                                        55VW1090
        C       SET UP TAB ARRAY FOR INTERPOLATION TO DETERMINE LOAD ON WING     55VW1100
        C                                                                        55VW1110
000267          CALL INTEST (850, 951)                                           55VW1120
000271          CALL TABLOD (951)                                                55VW1130
000273          CALL INTERP (   0, SS)                                           55VW1140
000275          ALIFT = TABLE(ININT)                                             55VW1150
```

```
000300          CALL INTERP ( 100, SS)                                          55VW1160
000302          ALIFT = TABLE(ININT) + ALIFT                                    55VW1170
000305          CALL INTERP ( 400, SS)                                          55VW1180
000307          ALIFT = TABLE(ININT) + ALIFT                                    55VW1190
000312          CALL INTERP ( 500, SS)                                          55VW1200
000314          ALIFT = TABLE(ININT) + ALIFT                                    55VW1210
000317          CALL INTERP ( 800, SS)                                          55VW1220
000321          ALIFT = TABLE(ININT) + ALIFT                                    55VW1230
                C                                                               55VW1240
                C     DETERMINE LOAD ON INBOARD PYLON                           55VW1250
                C                                                               55VW1260
000324          CALL INTEST (250, 351)                                          55VW1270
000326          CALL TABLOD (351)                                               55VW1280
000330          CALL INTERP (200, PI)                                           55VW1290
000332          BLIFT = TABLE(ININT)                                            55VW1300
                C                                                               55VW1310
                C     DETERMINE LOAD ON OUTBOARD PYLON                          55VW1320
                C                                                               55VW1330
000335          CALL INTEST (650, 751)                                          55VW1340
000337          CALL TABLOD (751)                                               55VW1350
000341          CALL INTERP (600, PO)                                           55VW1360
000343          CLIFT = TABLE(ININT)                                            55VW1370
000346          DUM1 = BLIFT / ALIFT * PIREQ                                    55VW1380
000350          DUM2 = CLIFT / ALIFT * POREQ                                    55VW1390
000352          WRITE (6,206) DUM1, DUM2                                        55VW1400
000362      206 FORMAT (1H1, 26X, 28H WING WITH TWO PYLONS BELOW   // 12X,      55VW1410
              A   44H RATIO OF INBOARD PYLON LOAD TO WING LOAD =  ,F14.8 /11X,  55VW1420
              B  45H RATIO OF OUTBOARD PYLON LOAD TO WING LOAD =  ,F14.8 // 6X, 55VW1430
              C  4HWING , 15X, 10HELLIPTICAL , 2(3X, 7HINBOARD) , 1X, 2( 3X,    55VW1440
              D  8HOUTBOARD ) / 4X, 8HSPANWISE , 5X, 4HWING , 7X, 4HWING ,      55VW1450
              E  2( 7X, 5HPYLON , 5X, 5HPYLON) / 3X, 10HCOORDINATE , 2X,        55VW1460
              F  7HLOADING , 4X, 7HLOADING , 2( 3X, 10HCOORDINATE , 2X, 7HLOADING55VW1470
              G  ) //)                                                          55VW1480
                                                                                55VW1490
                                                                                55VW1500
000362          DO 207 I=1,ININT                                                55VW1510
000364          ALOAD(I     ) = ALOAD(I     ) / ALIFT                           55VW1520
000367          ALOAD(I+200) = ALOAD(I+200) / BLIFT                             55VW1530
000370          ALOAD(I+600) = ALOAD(I+600) / CLIFT                             55VW1540
000372          DUM = ELOAD(ALOAD(I+50))                                        55VW1550
000375      207 WRITE (6,208) ALOAD(I+50), ALOAD(I), DUM, ALOAD(I+250),         55VW1560
              A  ALOAD(I+200), ALOAD(I+650), ALOAD(I+600)                       55VW1570
000422      208 FORMAT (1H , 7F11.6)                                            55VW1580
000422          DO 209 I=1,ININT                                                55VW1590
000423          ALOAD(I+100) = ALOAD(I+100) / ALIFT                             55VW1600
000425          DUM = ELOAD(ALOAD(I+150))                                       55VW1610
000430      209 WRITE (6,210) ALOAD(I+150), ALOAD(I+100), DUM                   55VW1620
000444      210 FORMAT (1H , 3F11.6)                                            55VW1630
000444          DO 211 I=1,ININT                                                55VW1640
000445          ALOAD(I+400) = ALOAD(I+400) / ALIFT                             55VW1650
000447          DUM = ELOAD(ALOAD(I+450))                                       55VW1660
000452      211 WRITE (6,210) ALOAD(I+450), ALOAD(I+400), DUM                   55VW1670
000466          DO 212 I=1,ININT                                                55VW1680
000467          ALOAD(I+500) = ALOAD(I+500) / ALIFT                             55VW1690
000471          DUM = ELOAD(ALOAD(I+550))                                       55VW1700
000474      212 WRITE (6,210) ALOAD(I+550), ALOAD(I+500), DUM                   55VW1710
000510          DO 213 I=1,ININT                                                55VW1720
000511          ALOAD(I+800) = ALOAD(I+800) / ALIFT                             55VW1730
000513          DUM = ELOAD(ALOAD(I+850))
000516      213 WRITE (6,210) ALOAD(I+850), ALOAD(I+800), DUM
```

```
000532          RETURN                                                          55VW1740
000532      300 CONTINUE                                                        55VW1750
          C                                                                     55VW1760
          C     WING WITH HORIZON .  TAIL (BIPLANE)                             55VW1770
          C                                                                     55VW1780
000532          DO 301 I=1,ININT                                                55VW1790
000534          ALOAD(I+150) = ALOAD(I+150) - ALOAD(ININT+150)                  55VW1800
000537          ALOAD(I+350) = (-ALOAD(I+350)) - ALOAD(ININT+250)               55VW1810
000542          ALOAD(I+250) = (-ALOAD(I+250))                                  55VW1820
000543          ALOAD(I+450) = (-ALOAD(I+450))                                  55VW1830
000544      301 ALOAD(I+550) = ALOAD(ININT+550) - ALOAD(I+550)                  55VW1840
000550          SS = ALOAD(ININT+50)                                            55VW1850
000552          PI = ALOAD(551)                                                 55VW1860
000553          CALL INTEST ( 50, 151)                                          55VW1870
000555          CALL TABLOD (151)                                               55VW1880
000557          CALL INTERP (  0, SS)                                           55VW1890
000561          ALIFT = TABLE(ININT)                                            55VW1900
000564          CALL INTEST (450, 551)                                          55VW1910
000566          CALL TABLOD (551)                                               55VW1920
000570          CALL INTERP (400, PI)                                           55VW1930
000572          BLIFT = TABLE(ININT)                                            55VW1940
000575          DUM1  = BLIFT / ALIFT * PIREQ                                   55VW1950
000577          WRITE (6,302) DUM1                                              55VW1960
000605      302 FORMAT (1H1, 18X, 37H WING WITH HORIZONTAL TAIL (BIPLANE) //7X, 55VW1970
             A  46H RATIO OF HORIZONTAL TAIL LOAD TO WING LOAD =  , F14.8 //    55VW1980
             B  2(8X,6HWING  ) , 5X, 10HELLIPTICAL , 2(4X, 10HHORIZONTAL) / 8X, 55VW1990
             C  4HSPAN , 9X, 5HLOCAL , 10X, 4HWING , 7X, 10HTAIL  SPAN , 7X,    55VW2000
             D  4HTAIL / 5X, 10HCOORDINATE , 5X, 7HLOADING , 7X, 7HLOADING ,    55VW2010
             E  6X, 10HCOORDINATE , 5X, 7HLOADING //)                           55VW2020
                                                                                55VW2030
000605          DO 303 I=1,ININT                                                55VW2040
000607          ALOAD(I     ) = ALOAD(I     ) / ALIFT                           55VW2050
000612          ALOAD(I+400) = ALOAD(I+400) / BLIFT                             55VW2060
000613          DUM = ELOAD(ALOAD(I+50))                                        55VW2070
000616      303 WRITE (6,304) ALOAD(I+50), ALOAD(I), DUM, ALOAD(I+450),         55VW2080
             A                ALOAD(I+400)                                     55VW2090
000637      304 FORMAT (1H , 5F14.8)                                            55VW2100
000637          RETURN                                                          55VW2110
000637      400 CONTINUE                                                        55VW2120
          C                                                                     55VW2130
          C     WING WITH PYLON ON TOP AND ON BOTTOM.                           55VW2140
          C                                                                     55VW2150
000637          DO 401 I=1,ININT                                                55VW2160
000641          ALOAD(I+150) = (-ALOAD(I+150)) - ALOAD(ININT+ 50)               55VW2170
000644          ALOAD(I+ 50) = (-ALOAD(I+ 50))                                  55VW2180
000646          ALOAD(I+250) = (-ALOAD(I+250))                                  55VW2190
000647          ALOAD(I+350) = ALOAD(ININT+350) - ALOAD(I+350)                  55VW2200
000651          ALOAD(I+650) = (-ALOAD(I+650))                                  55VW2210
000653          ALOAD(I+750) = ALOAD(ININT+750) - ALOAD(I+750)                  55VW2220
000655      401 ALOAD(I+950) = ALOAD(ININT+950) - ALOAD(I+950)                  55VW2230
000661          DO 402 I=1,ININT                                                55VW2240
000662          ALOAD(I+450) = ALOAD(I+450) + ALOAD(ININT+150)                  55VW2250
000665      402 ALOAD(I+850) = ALOAD(951) - ALOAD(I+850) + ALOAD(ININT+850)     55VW2260
000671          DO 403 I=1,ININT                                                55VW2270
000673      403 ALOAD(I+550) = ALOAD(I+550) - ALOAD(ININT+550) + ALOAD(851)     55VW2280
000701          SS = ALOAD(ININT+450)                                           55VW2290
000702          PI = ALOAD(ININT+250)                                           55VW2300
000704          PO = ALOAD(ININT+650)                                           55VW2310
          C
```

```
C     PERFORM INTERPOLATION FOR AND INTEGRATION OF LOADING ON WING
C
      LESS = .TRUE.
      CALL TABLOD (951)
      CALL INTERP (  0, SS)
      IF (ILAST .EQ. 1) GO TO 404
      LESS = .FALSE.
      CALL TABLOD (851)
      CALL INTERP (  0, SS)
  404 ALIFT = TABLE(ININT)
      CALL INTERP (100, SS)
      IF (ILAST .EQ. 1) GO TO 405
      LESS = .FALSE.
      CALL TABLOD (851)
      CALL INTERP (100, SS)
  405 ALIFT = ALIFT + TABLE(ININT)
      IF (LESS) CALL INTERP (400, SS)
      IF (LESS) CALL TABLOD (851)
      CALL INTERP (400, SS)
      CALL INTEST (450, 551)
      CALL TABLOD (551)
      CALL INTERP (400, SS)
      ALIFT = ALIFT + TABLE(ININT)
C
C     PERFORM INTERPOLATION FOR INBOARD PYLON
C
      CALL INTEST (250, 351)
      CALL TABLOD (351)
      CALL INTERP (200, PI)
      BLIFT = TABLE(ININT)
C
C     PERFORM INTERPOLATION FOR OUTBOARD PYLON
C
      CALL INTEST (650, 751)
      CALL TABLOD (751)
      CALL INTERP (600, PO)
      CLIFT = TABLE(ININT)
      DUM1 = BLIFT / ALIFT * PIREQ
      DUM2 = CLIFT / ALIFT * POREQ
      WRITE (6,406) DUM1, DUM2
  406 FORMAT (1H1, 21X, 39H WING WITH PYLON ABOVE AND PYLON BELOW  //
     A  12X, 44H RATIO OF INBOARD PYLON LOAD TO WING LOAD =  , F14.8 /
     B 11X, 45H RATIO OF OUTBOARD PYLON LOAD TO WING LOAD =  , F14.8 //
     C  28X, 4HWING , 2X, 2( 3X, 8HINBOARD ) , 2( 3X, 8HOUTBOARD) / 3X,
     D  10HWING  SPAN , 4X, 4HWING , 4X, 10HELLIPTICAL , 4(3X , 5HPYLON
     E   , 3X) / 3X, 10HCOORDINATE , 2( 3X, 7HLOADING) , 2( 3X,
     F   10HCOORDINATE , 2X, 7HLOADING ) //)
      DO 407 I=1,ININT
      ALOAD(I     ) = ALOAD(I     ) / ALIFT
      ALOAD(I+200) = ALOAD(I+200) / BLIFT
      ALOAD(I+600) = ALOAD(I+600) / CLIFT
      DUM = FLOAD(ALOAD(I+50))
  407 WRITE (6,408) ALOAD(I+50), ALOAD(I), DUM, ALOAD(I+250),
     A   ALOAD(I+200), ALOAD(I+650), ALOAD(I+600)
  408 FORMAT (1H , 7F11.6)
      DO 409 I=1,ININT
      ALOAD(I+100) = ALOAD(I+100) / ALIFT
      DUM = FLOAD(ALOAD(I+150))
```

```
001063     409 WRITE (6,410) ALOAD(I+150), ALOAD(I+100) , DUM                  55VW2900
001077     410 FORMAT (1H , 3F11.6)                                            55VW2910
001077         DO 411 I=1,ININT                                                55VW2920
001100         ALOAD(I+400) = ALOAD(I+400) / ALIFT                             55VW2930
001102         DUM = ELOAD(ALOAD(I+450))                                       55VW2940
001105     411 WRITE (6,410) ALOAD(I+450), ALOAD(I+400), DUM                   55VW2950
001121         RETURN                                                          55VW2960
001121     500 CONTINUE                                                        55VW2970
          C                                                                    55VW2980
          C    WING WITH DIHEDRAL AND PYLON BELOW AT DIHEDRAL BREAK            55VW2990
          C                                                                    55VW3000
001121         DO 501 I=1,ININT                                                55VW3010
001123         ALOAD(I+150) = ALOAD(I+150) + ALOAD(ININT+ 50)                  55VW3020
001126         ALOAD(I+350) = ALOAD(I+350) - ALOAD(ININT+350)                  55VW3030
001130         ALOAD(I+450) = ALOAD(I+450) * CG                                55VW3040
001132         ALOAD(I+550) = ALOAD(I+550) * CG                                55VW3050
001133         ALOAD(I+650) = ALOAD(ININT+650) + ALOAD(ININT+750) - ALOAD(I+650) 55VW3060
001136     501 ALOAD(I+750) = ALOAD(ININT+750) - ALOAD(I+750)                  55VW3070
001142         DO 502 I=1,ININT                                                55VW3080
001143         ALOAD(I+450) = ALOAD(ININT+150) - ALOAD(I+450)                  55VW3090
001146     502 ALOAD(I+550) = ALOAD(ININT+550) - ALOAD(I+550) + ALOAD(651)     55VW3100
001153         SS = ALOAD(ININT+450)                                           55VW3110
001154         PI = ALOAD(ININT+250)                                           55VW3120
          C                                                                    55VW3130
          C    DETERMINE LOADING ON WING                                       55VW3140
          C                                                                    55VW3150
001156         CALL TABLOD (751)                                               55VW3160
001157         CALL INTERP (  0, SS)                                           55VW3170
001161         IF (ILAST .EQ. 1) GO TO 521                                     55VW3180
001163         CALL TABLOD (651)                                               55VW3190
001165         CALL INTERP (  0, SS)                                           55VW3200
001167     521 ALIFT = TABLE(ININT)                                            55VW3210
001172         CALL INTERP (100,SS)                                            55VW3220
001174         IF (ILAST .EQ. 1) GO TO 522                                     55VW3230
001176         CALL INTEST (150, 651)                                          55VW3240
001200         CALL TABLOD (651)                                               55VW3250
001202         CALL INTERP (100,SS)                                            55VW3260
001204     522 ALIFT = ALIFT + TABLE(ININT)                                    55VW3270
001207         CALL INTEST (450, 551)                                          55VW3280
001211         CALL TABLOD (551)                                               55VW3290
001213         CALL INTERP (400,SS)                                            55VW3300
001215         ALIFT = ALIFT + TABLE(ININT)                                    55VW3310
001220         CLIFT = TABLE(ININT) / CG                                       55VW3320
          C                                                                    55VW3330
          C    DETERMINE LOADING ON PYLON                                      55VW3340
          C                                                                    55VW3350
001223         CALL INTEST (250, 351)                                          55VW3360
001225         CALL TABLOD (351)                                               55VW3370
001227         CALL INTERP (200,PI)                                            55VW3380
001231         BLIFT = TABLE(ININT)                                            55VW3390
001234         DUM1 = BLIFT / ALIFT * PIREQ                                    55VW3400
001236         DUM2 = CLIFT / ALIFT                                            55VW3410
001240         WRITE (6,503) DUM1, DUM2                                        55VW3420
001247     503 FORMAT (1H1, 13X, 48H WING WITH PYLON AND DIHEDRAL OUTBOARD OF PYL55VW3430
           AON  // 10X, 41H RATIO OF PYLON LOAD TO WING LIFT LOAD =  ,F14.7 //55VW3440
           B       62H RATIO OF OUTBOARD WING PANEL NORMAL LOAD TO WING LIFT L55VW3450
           COAD =  , F14.7 // 28X, 8HOUTBOARD / 18X, 2(5H WING , 6X) ,        55VW3460
           D  10HELLIPTICAL / 4X, 10HWING  SPAN , 5X, 4HLIFT , 6X, 6HNORMAL , 55VW3470
```

```
              E  8X, 4HWING , 2( 7X, 5HPYLON) / 4X, 10HCOORDINATE , 3( 3X,        55VW3480
              F   7HLOADING , 2X) , 2X, 10HCOORDINATE , 3X, 7HLOADING //)          55VW3490
001247          DO 504 I=1,ININT                                                  55VW3500
001251          ALOAD(I     ) = ALOAD(I     ) / ALIFT                             55VW3510
001254          ALOAD(I+200) = ALOAD(I+200) / BLIFT                               55VW3520
001255          DUM = ELOAD(ALOAD(I+50))                                          55VW3530
001260      504 WRITE (6,505) ALOAD(I+50), ALOAD(I), DUM, ALOAD(I+250),           55VW3540
              A   ALOAD(I+200)                                                    55VW3550
001301      505 FORMAT (1H , 2F12.7, 12X, 3F12.7)                                 55VW3560
001301          DO 506 I=1,ININT                                                  55VW3570
001302          ALOAD(I+100) = ALOAD(I+100) / ALIFT                               55VW3580
001304          DUM = ELOAD(ALOAD(I+150))                                         55VW3590
001307      506 WRITE (6,507) ALOAD(I+150), ALOAD(I+100), DUM                     55VW3600
001323      507 FORMAT (1H , 2F12.7, 12X, F12.7)                                  55VW3610
001323          DO 508 I=1,ININT                                                  55VW3620
001324          ALOAD(I+400) = ALOAD(I+400) / ALIFT                               55VW3630
001326          DUM1 = ALOAD(I+400) / CG                                          55VW3640
001330          DUM  = ELOAD(ALOAD(I+450))                                        55VW3650
001333      508 WRITE (6,509) ALOAD(I+450), ALOAD(I+400), DUM1, DUM               55VW3660
001351      509 FORMAT (1H , 4F12.7)                                              55VW3670
001351          RETURN                                                            55VW3680
001351      600 CONTINUE                                                          55VW3690
        C                                                                         55VW3700
        C       WING WITH INBOARD PYLON AND OUTBOARD DIHEDRAL                     55VW3710
        C                                                                         55VW3720
001351          DO 601 I=1,ININT                                                  55VW3730
001353          ALOAD(I+150) = ALOAD(I+150) + ALOAD(ININT+ 50)                    55VW3740
001356          ALOAD(I+350) = ALOAD(I+350) - ALOAD(ININT+350)                    55VW3750
001360          ALOAD(I+650) = ALOAD(I+650) * CG                                  55VW3760
001362          ALOAD(I+750) = ALOAD(I+750) * CG                                  55VW3770
001363          ALOAD(I+850) = ALOAD(ININT+950) - ALOAD(I+850) + ALOAD(ININT+850) 55VW3780
001366      601 ALOAD(I+950) = ALOAD(ININT+950) - ALOAD(I+950)                    55VW3790
001372          DO 602 I=1,ININT                                                  55VW3800
001373          ALOAD(I+550) = ALOAD(ININT+150) - ALOAD(ININT+450) - ALOAD(I+550) 55VW3810
001377          ALOAD(I+450) = ALOAD(ININT+150) - ALOAD(I+450)                    55VW3820
001401      602 ALOAD(I+750) = ALOAD(851) + ALOAD(ININT+750) - ALOAD(I+750)       55VW3830
001405          DO 603 I=1,ININT                                                  55VW3840
001407      603 ALOAD(I+650) = ALOAD(ININT+550) - ALOAD(I+650)                    55VW3850
001414          SS = ALOAD(ININT+650)                                             55VW3860
001415          PI = ALOAD(ININT+250)                                             55VW3870
        C                                                                         55VW3880
        C       DETERMINE LOADING ON WING                                         55VW3890
        C                                                                         55VW3900
001417          LESS = .FALSE.                                                    55VW3910
001420          CALL TABLOD (951)                                                 55VW3920
001421          CALL INTERP (  0, SS)                                             55VW3930
001423          CALL INTEST (550, 851)                                            55VW3940
001425          IF (ILAST .EQ. 1) GO TO 621                                       55VW3950
001427          LESS = .TRUE.                                                     55VW3960
001430          CALL TABLOD (851)                                                 55VW3970
001431          CALL INTERP (  0, SS)                                             55VW3980
001433      621 ALIFT = TABLE(ININT)                                              55VW3990
001436          CALL INTERP (100, SS)                                             55VW4000
001440          IF (ILAST .EQ. 1) GO TO 622                                       55VW4010
001442          LESS = .TRUE.                                                     55VW4020
001443          CALL TABLOD (851)                                                 55VW4030
001444          CALL INTERP (100, SS)                                             55VW4040
001446      622 ALIFT = ALIFT + TABLE(ININT)                                      55VW4050
```

```
001451          CALL INTERP (400, SS)
001453          IF (ILAST .EQ. 1) GO TO 623                                     55VW4060
001455          LESS = .TRUE.                                                   55VW4070
001456          CALL TABLOD (851)                                               55VW4080
001457          CALL INTERP (400, SS)                                           55VW4090
001461      623 ALIFT = ALIFT + TABLE(ININT)                                    55VW4100
001464          IF (.NOT. LESS) CALL TABLOD (851)                               55VW4110
001467          CALL INTERP (500, SS)                                           55VW4120
001471          ALIFT = ALIFT + TABLE(ININT)                                    55VW4130
001474          CALL INTEST (650, 751)                                          55VW4140
001476          CALL TABLOD (751)                                               55VW4150
001500          CALL INTERP (600, SS)                                           55VW4160
001502          ALIFT = ALIFT + TABLE(ININT)                                    55VW4170
001505          CLIFT = TABLE(ININT) / CG                                       55VW4180
          C                                                                     55VW4190
          C     DETERMINE LOADING ON PYLON                                      55VW4200
          C                                                                     55VW4210
001510          CALL INTEST (250, 351)                                          55VW4220
001512          CALL TABLOD (351)                                               55VW4230
001514          CALL INTERP (200, PI)                                           55VW4240
001516          BLIFT = TABLE(ININT)                                            55VW4250
001521          DUM1 = BLIFT / ALIFT * PIREQ                                    55VW4260
001523          DUM2 = CLIFT / ALIFT                                            55VW4270
001525          WRITE (6,604) DUM1, DUM2                                        55VW4280
001534      604 FORMAT (1H1,       80H WING WITH ONE PYLON INBOARD AND OUTBOARD DIH55VW4300
             AEDRAL WITH BREAK OUTBOARD OF PYLON  // 10X, 41H RATIO OF PYLON LOA55VW4310
             BD TO WING LIFT LOAD =  ,F15.8 //  3X, 56H RATIO OF OUTBOARD WING N55VW4320
             CURMAL LOAD TO WING LIFT LOAD =  ,F14.8// 29X, 8HOUTBOARD / 11X,   55VW4330
             D  2( 8X, 4HWING) , 5X, 10HELLIPTICAL / 4X, 10HWING  SPAN , 5X,    55VW4340
             E  4HLIFT , 7X, 6HNORMAL , 7X, 4HWING , 2( 7X, 5HPYLON) / 4X,      55VW4350
             F  10HCOORDINATE , 3( 3X, 7HLOADING , 2X) , 2X, 10HCOORDINATE ,    55VW4360
             G   3X, 7HLOADING //)                                              55VW4370
001534          DO 605 I=1,ININT                                                55VW4380
001536          ALOAD(I     ) = ALOAD(I     ) / ALIFT                           55VW4390
001541          ALOAD(I+200) = ALOAD(I+200) / BLIFT                             55VW4400
001542          DUM = ELOAD(ALOAD(I+50))                                        55VW4410
001545      605 WRITE (6,606) ALOAD(I+50), ALOAD(I), DUM, ALOAD(I+250),         55VW4420
             A    ALOAD(I+200)                                                  55VW4430
001566      606 FORMAT (1H , 2F12.7, 12X, 3F12.7)                               55VW4440
001566          DO 607 I=1,ININT                                                55VW4450
001567          ALOAD(I+100) = ALOAD(I+100) / ALIFT                             55VW4460
001571          DUM = ELOAD(ALOAD(I+150))                                       55VW4470
001574      607 WRITE (6,608) ALOAD(I+150), ALOAD(I+100), DUM                   55VW4480
001610      608 FORMAT (1H , 2F12.7, 12X, F12.7)                                55VW4490
001610          DO 609 I=1,ININT                                                55VW4500
001611          ALOAD(I+400) = ALOAD(I+400) / ALIFT                             55VW4510
001613          DUM = ELOAD(ALOAD(I+450))                                       55VW4520
001616      609 WRITE (6,608) ALOAD(I+450), ALOAD(I+400), DUM                   55VW4530
001632          DO 610 I=1,ININT                                                55VW4540
001633          ALOAD(I+500) = ALOAD(I+500) / ALIFT                             55VW4550
001635          DUM = ELOAD(ALOAD(I+550))                                       55VW4560
001640      610 WRITE (6,608) ALOAD(I+550), ALOAD(I+500), DUM                   55VW4570
001654          DO 611 I=1,ININT                                                55VW4580
001655          ALOAD(I+600) = ALOAD(I+600) / ALIFT                             55VW4590
001657          DUM1 = ALOAD(I+600) / CG                                        55VW4600
001661          DUM = ELOAD(ALOAD(I+650))                                       55VW4610
001664      611 WRITE (6,612) ALOAD(I+650), ALOAD(I+600), DUM1, DUM             55VW4620
001702      612 FORMAT (1H , 4F12.7)                                            55VW4630
```

```
001702          RETURN
001702      700 CONTINUE                                                        55VW4640
          C                                                                     55VW4650
          C     WING WITH END-PLATED VERTICAL FENCE AT THE TIP                  55VW4660
          C                                                                     55VW4670
001702          DO 701 I=1,ININT                                                55VW4680
001704          ALOAD(I+150) = ALOAD(I+150) + ALOAD(ININT+50)                   55VW4690
001707          ALOAD(I+350) = ALOAD(I+350) + ALOAD(ININT+250)                  55VW4700
001711          ALOAD(I+650) = ALOAD(ININT+650) - ALOAD(I+650) + ALOAD(ININT+450) 55VW4710
001714          ALOAD(I+450) = ALOAD(ININT+450) - ALOAD(I+450)                  55VW4720
001716          ALOAD(I+550) = (-ALOAD(I+550))                                  55VW4730
001717          ALOAD(I+750) = ALOAD(ININT+750) - ALOAD(I+750)                  55VW4740
001722      701 ALOAD(I+850) = ALOAD(ININT+850) - ALOAD(I+850)                  55VW4750
001725          SS = ALOAD(ININT+150)                                           55VW4760
          C                                                                     55VW4770
          C     PI WILL REPRESENT TOTAL LENGTH OF VERTICAL FENCE                55VW4780
          C     PO WILL REPRESENT TOTAL LENGTH OF END PLATE ON VERTICAL FENCE   55VW4790
          C                                                                     55VW4800
001727          PI = ALOAD(ININT+350)                                           55VW4810
001730          PO = ALOAD(ININT+550)                                           55VW4820
          C                                                                     55VW4830
          C     INTERPOLATE FOR WING LOADING                                    55VW4840
          C                                                                     55VW4850
001732          CALL INTEST (150, 851)                                          55VW4860
001734          CALL TABLOD (851)                                               55VW4870
001736          CALL INTERP (  0, SS)                                           55VW4880
001740          ALIFT = TABLE(ININT)                                            55VW4890
001743          CALL INTERP (100, SS)                                           55VW4900
001745          ALIFT = ALIFT + TABLE(ININT)                                    55VW4910
          C                                                                     55VW4920
          C     INTERPOLATE FOR LOADING ON VERTICAL FENCE                       55VW4930
          C                                                                     55VW4940
001750          CALL INTEST (350, 751)                                          55VW4950
001752          CALL TABLOD (751)                                               55VW4960
001754          CALL INTERP (200, PI)                                           55VW4970
001756          BLIFT = TABLE(ININT)                                            55VW4980
001761          CALL INTERP (300, PI)                                           55VW4990
001763          BLIFT = BLIFT + TABLE(ININT)                                    55VW5000
          C                                                                     55VW5010
          C     DETERMINE LOADING ON END PLATES                                 55VW5020
          C                                                                     55VW5030
001766          T = (ALOAD(651) - ALOAD(ININT+550)) / ALOAD(ININT+550)          55VW5040
001771          IF (T .GT. 0.0 .AND. T .LE. C1) ALOAD(651) = ALOAD(ININT+550)   55VW5050
               A                                          - C2 * ABS(ALOAD(ININT+550)) 55VW5060
002005          DO 721 I=1,ININT                                                55VW5070
002007          J = 2 * I                                                       55VW5080
002010          K = J + 1                                                       55VW5090
002012          TAB(J) = ALOAD(I+550)                                           55VW5100
002014      721 TAB(K) = ALOAD(I+500)                                           55VW5110
002022          CALL INTERP (400, PO)                                           55VW5120
002024          CLIFT = TABLE(ININT)                                            55VW5130
002027          DLIFT = CLIFT                                                   55VW5140
002030          CALL INTERP (600, PO)                                           55VW5150
002032          DO 702 I=1,ININT                                                55VW5160
002034          J = I + 600                                                     55VW5170
002036          K = I + 400                                                     55VW5180
002037          ALOAD(K) = (-ALOAD(K))                                          55VW5190
002041      702 ALOAD(J) = (-ALOAD(J))                                          55VW5200
                                                                                55VW5210
```

```
002046          CLIFT = CLIFT + TABLE(ININT)                                    55VW5220
002051          ELIFT = TABLE(ININT)                                            55VW5230
002053          DUM1 = BLIFT / ALIFT * PIREQ                                    55VW5240
002055          DUM2 = DLIFT / ALIFT * ETAIRQ                                   55VW5250
002057          DUM3 = ELIFT / ALIFT * ETAORQ                                   55VW5260
002061          WRITE (6,703) DUM1, DUM2, DUM3                                  55VW5270
002073      703 FORMAT (1H1, 16X, 48H WING WITH END-PLATED VERTICAL FENCE AT THE T55VW5280
            A1P // 12X, 45H RATIO OF VERTICAL FENCE LOAD TO WING LOAD =  ,F14.755VW5290
            B /  9X, 48H RATIO OF INBOARD END PLATE LOAD TO WING LOAD =  ,F14.755VW5300
            C / 8X, 49H RATIO OF OUTBOARD END PLATE LOAD TO WING LOAD =  ,F14.755VW5310
            D // 48X, 9HEND PLATE / 25X, 10HELLIPTICAL , 2X, 9HEND PLATE , 2X  55VW5320
            E , 8HLOADING,,2( 3X, 8HVERTICAL ) / 3X, 10HWING  SPAN , 2( 4X,    55VW5330
            F  4HWING , 3X) , 4X, 4HSPAN , 5X, 8HSTARTING , 2(4X, 5HFENCE, 2X) 55VW5340
            G / 3X, 10HCOORDINATE , 2(3X, 8HLOADING ) , 11H COORDINATE , 2X,   55VW5350
            H 8HOUTBOARD , 2X, 10HCOORDINATE , 2X, 7HLOADING  //)              55VW5360
002073          DO 704 I=1,ININT                                                55VW5370
002075          ALOAD(I     ) = ALOAD(I     ) / ALIFT                           55VW5380
002100          ALOAD(I+200) = ALOAD(I+200) / BLIFT                             55VW5390
002101          ALOAD(I+600) = ALOAD(I+600) / CLIFT                             55VW5400
002103          DUM = ELOAD(ALOAD(I+50))                                        55VW5410
002106      704 WRITE (6,705) ALOAD(I+50), ALOAD(I), DUM, ALOAD(I+650),         55VW5420
            A ALOAD(I+600), ALOAD(I+250), ALOAD(I+200)                          55VW5430
002133      705 FORMAT (1H , 7F11.6)                                            55VW5440
002133          DO 706 I=1,ININT                                                55VW5450
002134          ALOAD(I+100) = ALOAD(I+100) / ALIFT                             55VW5460
002136          ALOAD(I+300) = ALOAD(I+300) / BLIFT                             55VW5470
002140          ALOAD(I+400) = ALOAD(I+400) / CLIFT                             55VW5480
002142          DUM = ELOAD(ALOAD(I+150))                                       55VW5490
002145      706 WRITE (6,705) ALOAD(I+150), ALOAD(I+100), DUM, ALOAD(I+450),    55VW5500
            A   ALOAD(I+400), ALOAD(I+350), ALOAD(I+300)                        55VW5510
002171          RETURN                                                          55VW5520
002171          END                                                             55VW5530
```

SUBPROGRAM LENGTH
003065

FUNCTION ASSIGNMENTS

STATEMENT ASSIGNMENTS

100	-	000017	103	-	002205	105	-	002244	107	-	002247
200	-	000210	203	-	000240	204	-	000246	205	-	000254
206	-	002262	208	-	002342	210	-	002345	300	-	000532
302	-	002353	304	-	002421	400	-	000637	403	-	000673
404	-	000717	405	-	000732	406	-	002427	408	-	002503
410	-	002506	500	-	001121	503	-	002515	505	-	002573
507	-	002577	509	-	002603	521	-	001167	522	-	001204
600	-	001351	603	-	001407	604	-	002611	606	-	002672
608	-	002676	612	-	002702	621	-	001433	622	-	001446
623	-	001461	700	-	001702	703	-	002710	705	-	003003

BLOCK NAMES AND LENGTHS
- 002420

VARIABLE ASSIGNMENTS

AA	-	002217C01	ABSC	-	000021C01	ALIFT	-	003047	ALOAD	-	000247C01
BLIFT	-	003050	CG	-	002407C01	CLIFT	-	003055	C1	-	002416C01
C2	-	002414C01	DATA	-	000000C01	DLIFT	-	003062	DUM	-	003052
DUM1	-	003051	DUM2	-	003056	DUM3	-	003064	FLIFT	-	003063
ETAIMQ	-	000014C01	ETAORQ	-	000015C01	I	-	003044	ILAST	-	002413C01
ININT	-	000013C01	J	-	003060	K	-	003061	LESS	-	003043
NN	-	002237C01	NTYPE	-	000012C01	ORD	-	000103C01	PI	-	003053
PIREQ	-	000016C01	PL	-	003046	PO	-	003054	POREQ	-	000017C01
SS	-	003045	T	-	003057	TAB	-	002237C01	TABLE	-	000165C01

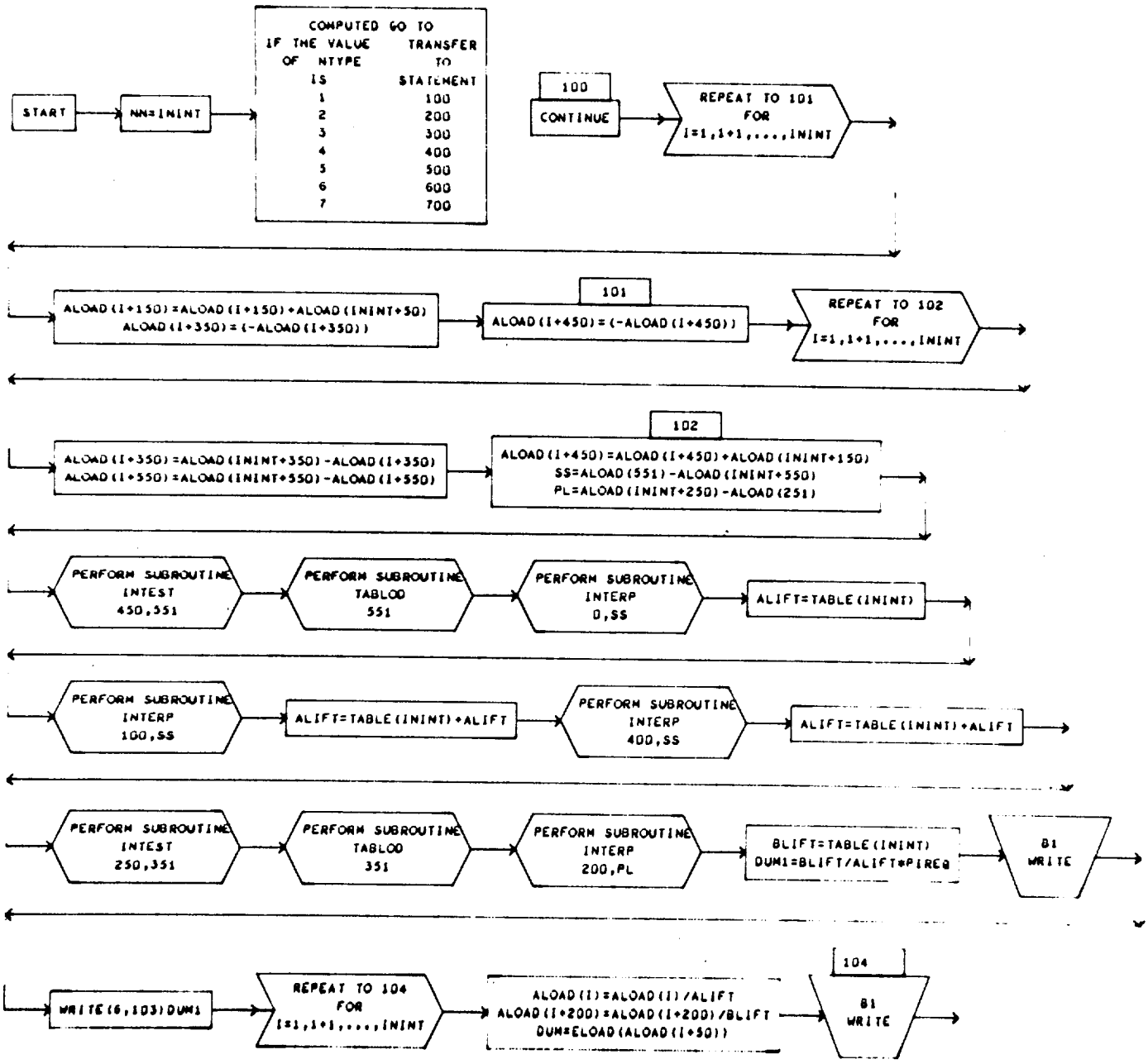
START OF CONSTANTS
002173

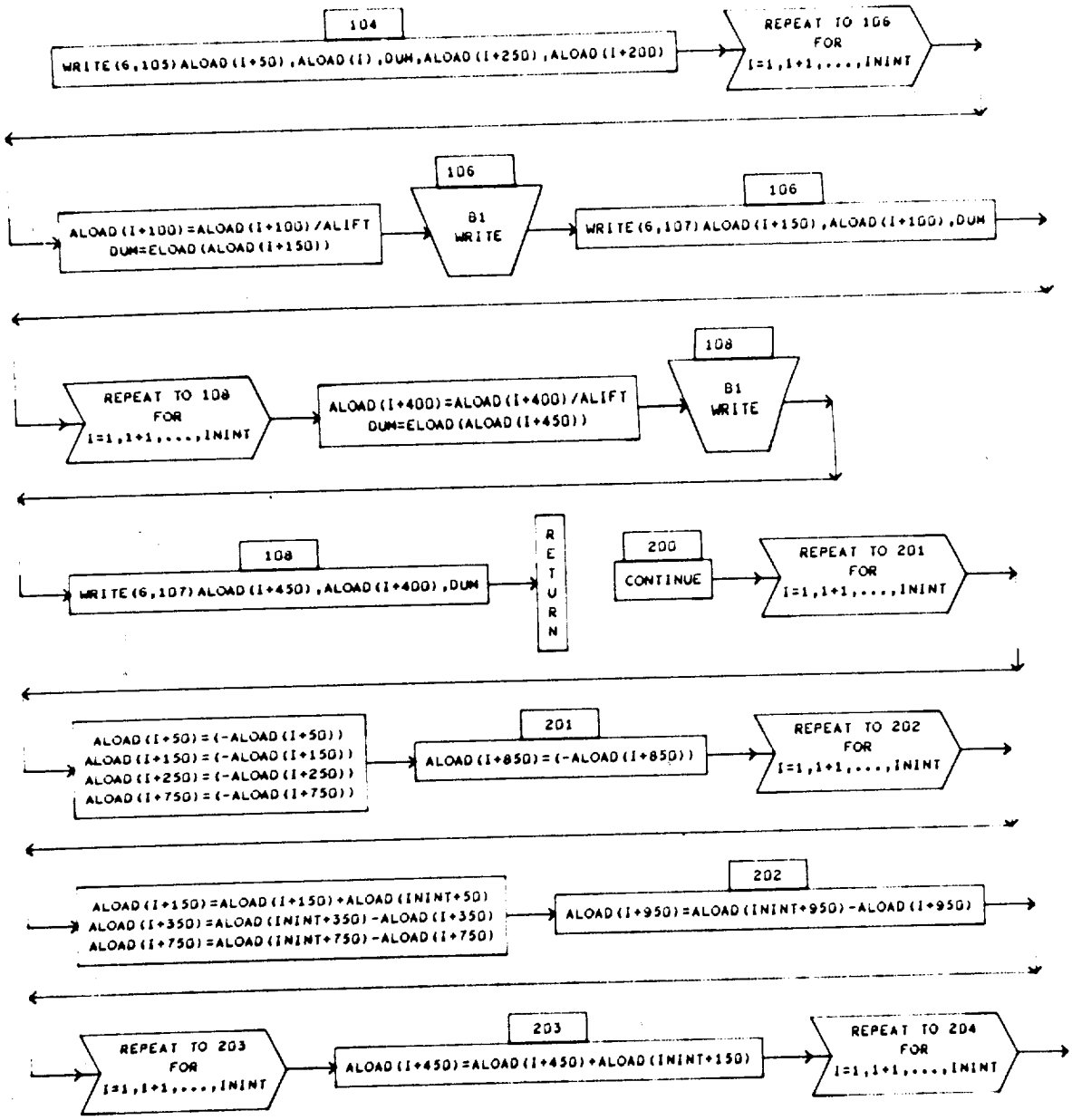
START OF TEMPORARIES
003006

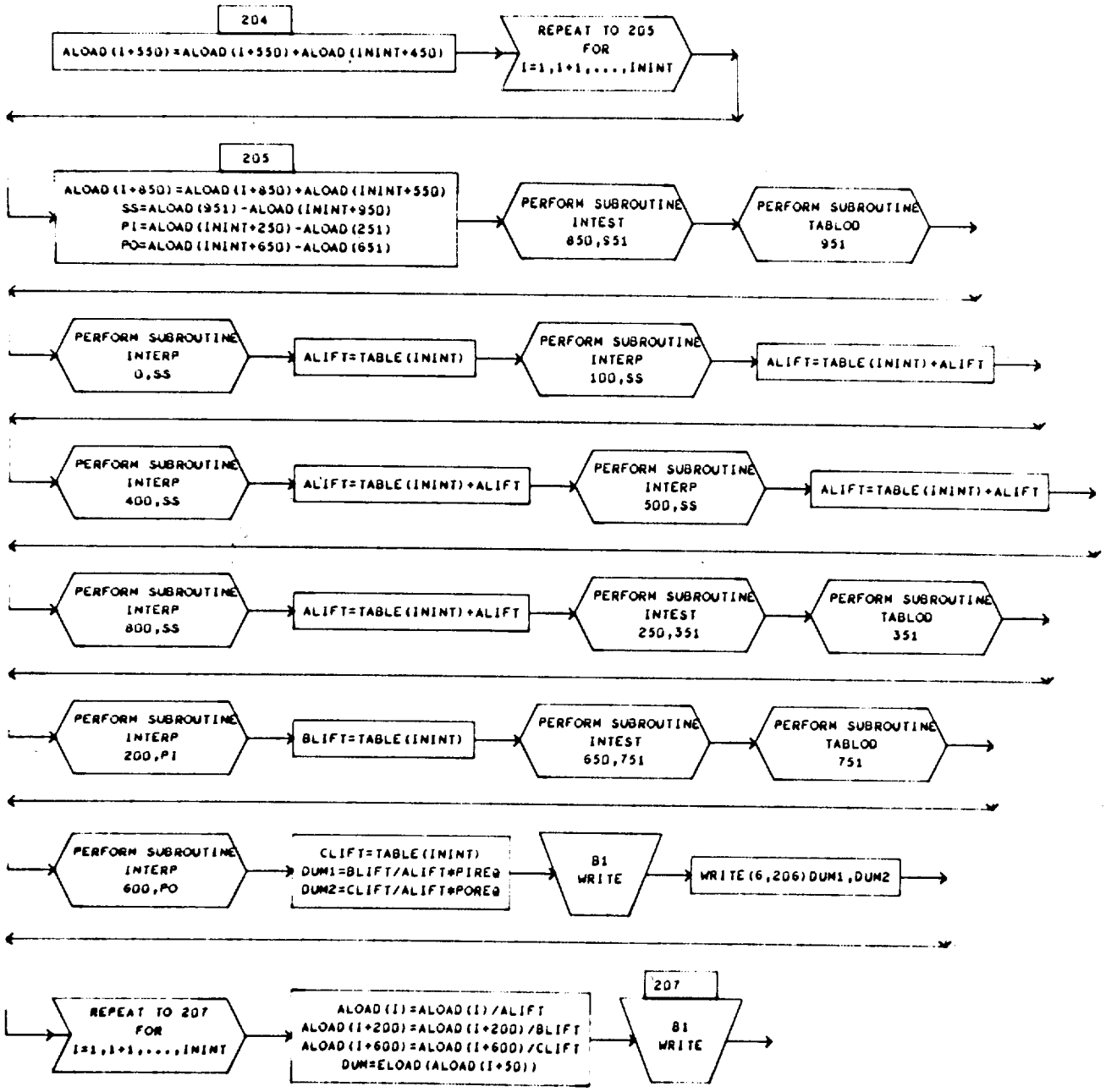
START OF INDIRECTS
003022

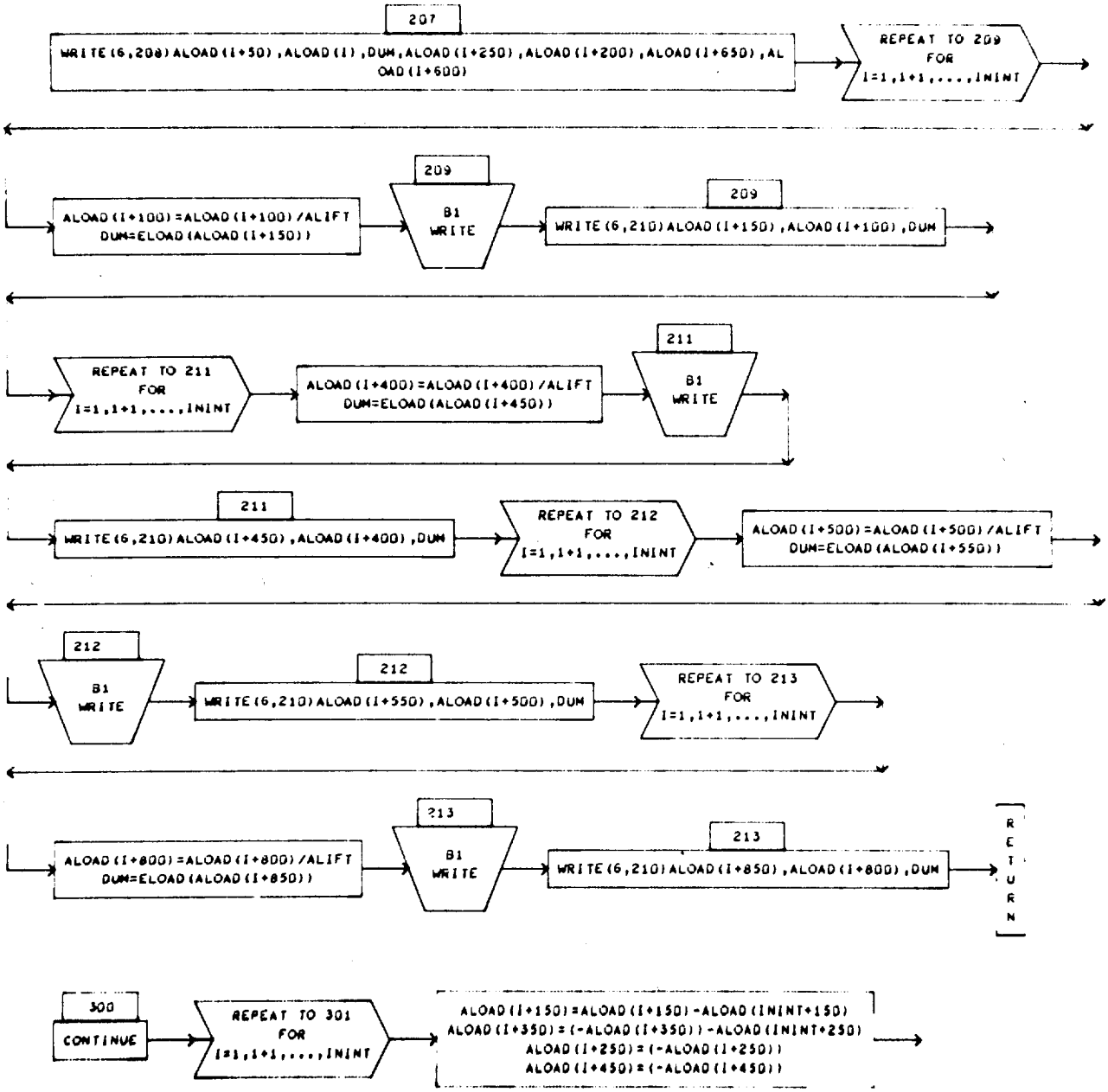
UNUSED COMPILER SPACE
033700

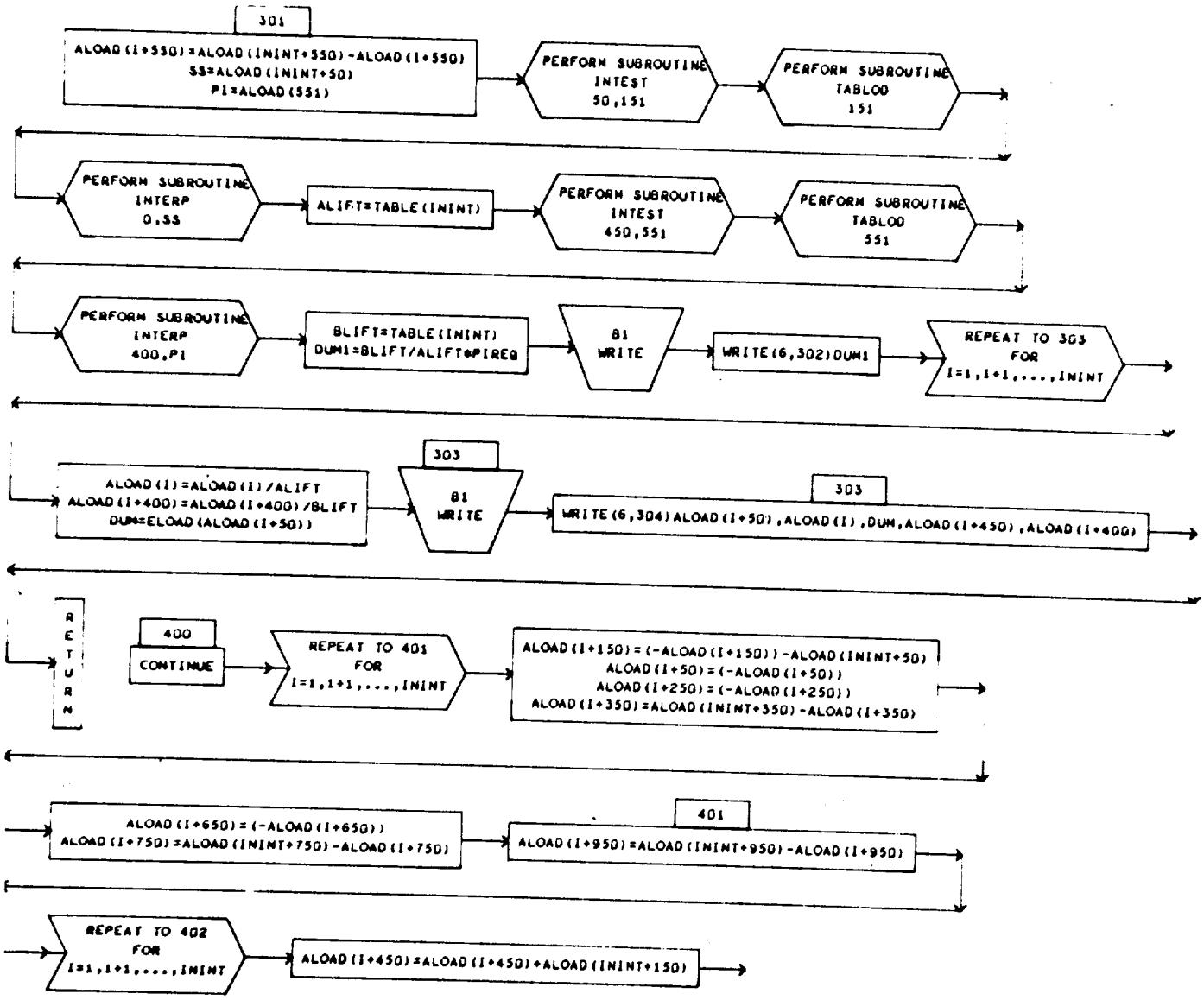
SUBROUTINE LOAD

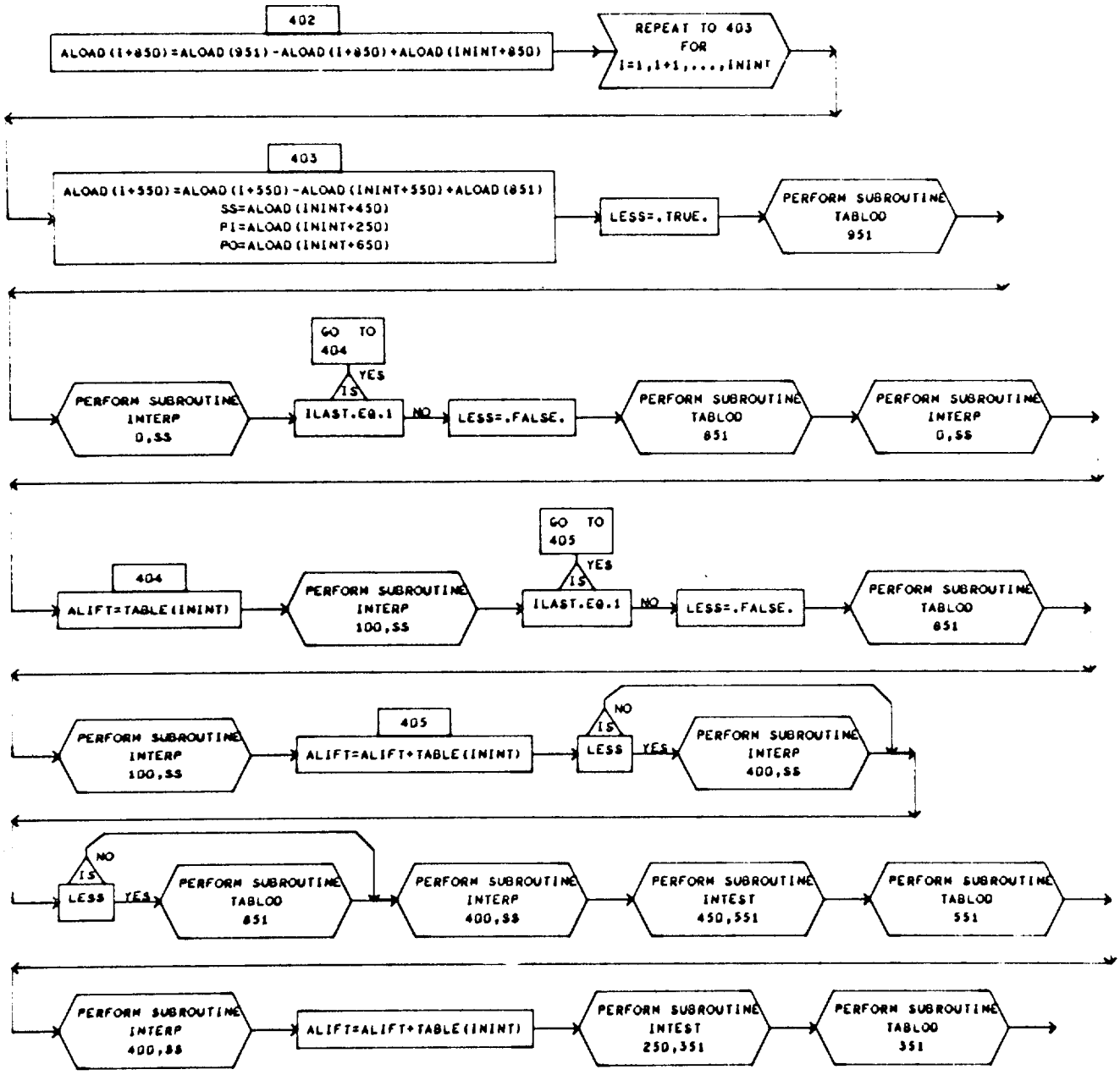


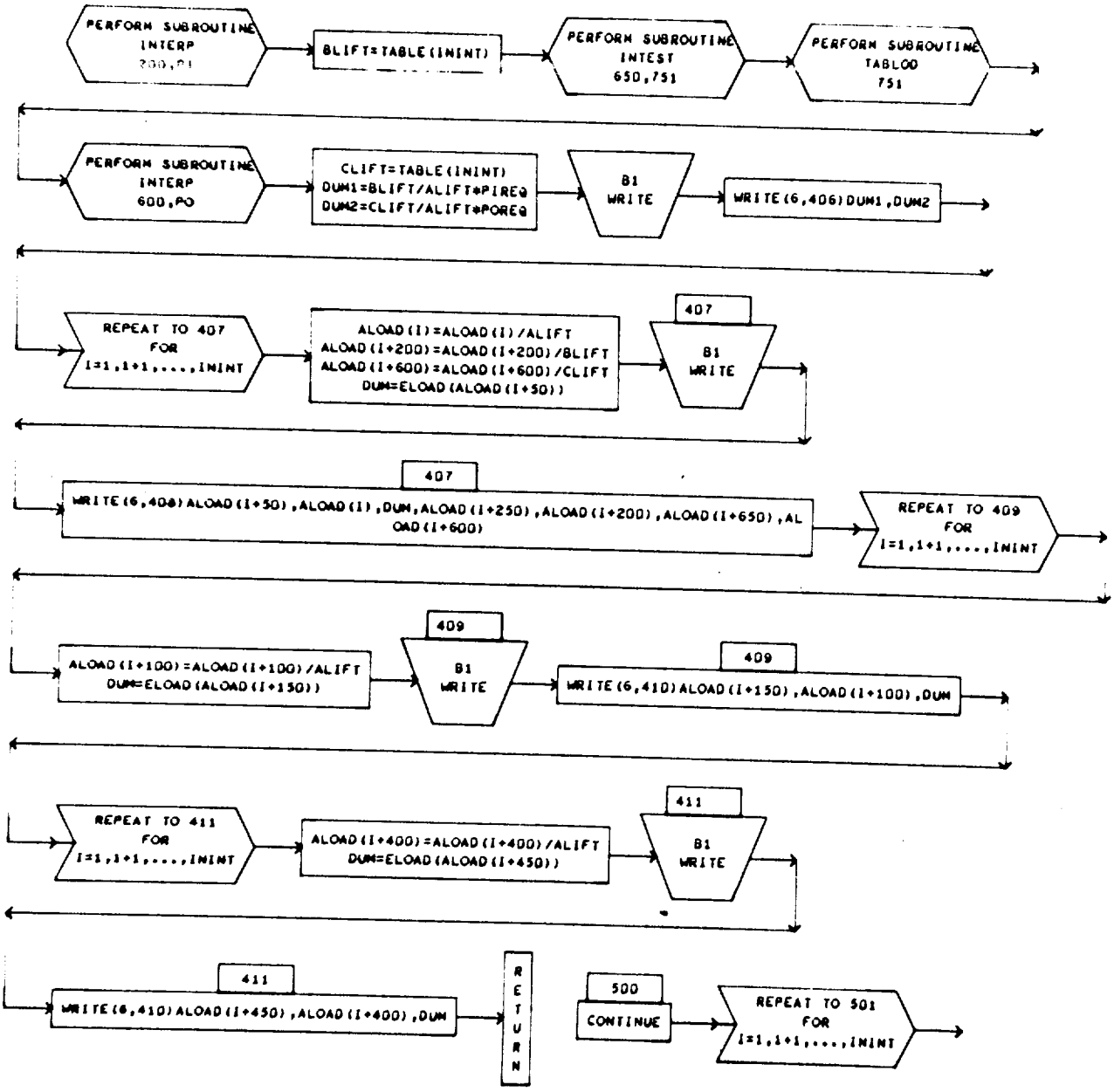

104
WRITE(6,105)ALOAD(I+50),ALOAD(I),DUM,ALOAD(I+250),ALOAD(I+200)
REPEAT TO 106
FOR
I=1,I+1,...,ININT
106
ALOAD(I+100)=ALOAD(I+100)/ALIFT
DUM=ELOAD(ALOAD(I+150))
B1
WRITE
106
WRITE(6,107)ALOAD(I+150),ALOAD(I+100),DUM
REPEAT TO 108
FOR
I=1,I+1,...,ININT
ALOAD(I+400)=ALOAD(I+400)/ALIFT
DUM=ELOAD(ALOAD(I+450))
108
B1
WRITE
108
WRITE(6,107)ALOAD(I+450),ALOAD(I+400),DUM
RETURN
200
CONTINUE
REPEAT TO 201
FOR
I=1,I+1,...,ININT
ALOAD(I+50)=(-ALOAD(I+50))
ALOAD(I+150)=(-ALOAD(I+150))
ALOAD(I+250)=(-ALOAD(I+250))
ALOAD(I+750)=(-ALOAD(I+750))
201
ALOAD(I+850)=(-ALOAD(I+850))
REPEAT TO 202
FOR
I=1,I+1,...,ININT
ALOAD(I+150)=ALOAD(I+150)+ALOAD(ININT+50)
ALOAD(I+350)=ALOAD(ININT+350)-ALOAD(I+350)
ALOAD(I+750)=ALOAD(ININT+750)-ALOAD(I+750)
202
ALOAD(I+950)=ALOAD(ININT+950)-ALOAD(I+950)
REPEAT TO 203
FOR
I=1,I+1,...,ININT
203
ALOAD(I+450)=ALOAD(I+450)+ALOAD(ININT+150)
REPEAT TO 204
FOR
I=1,I+1,...,ININT

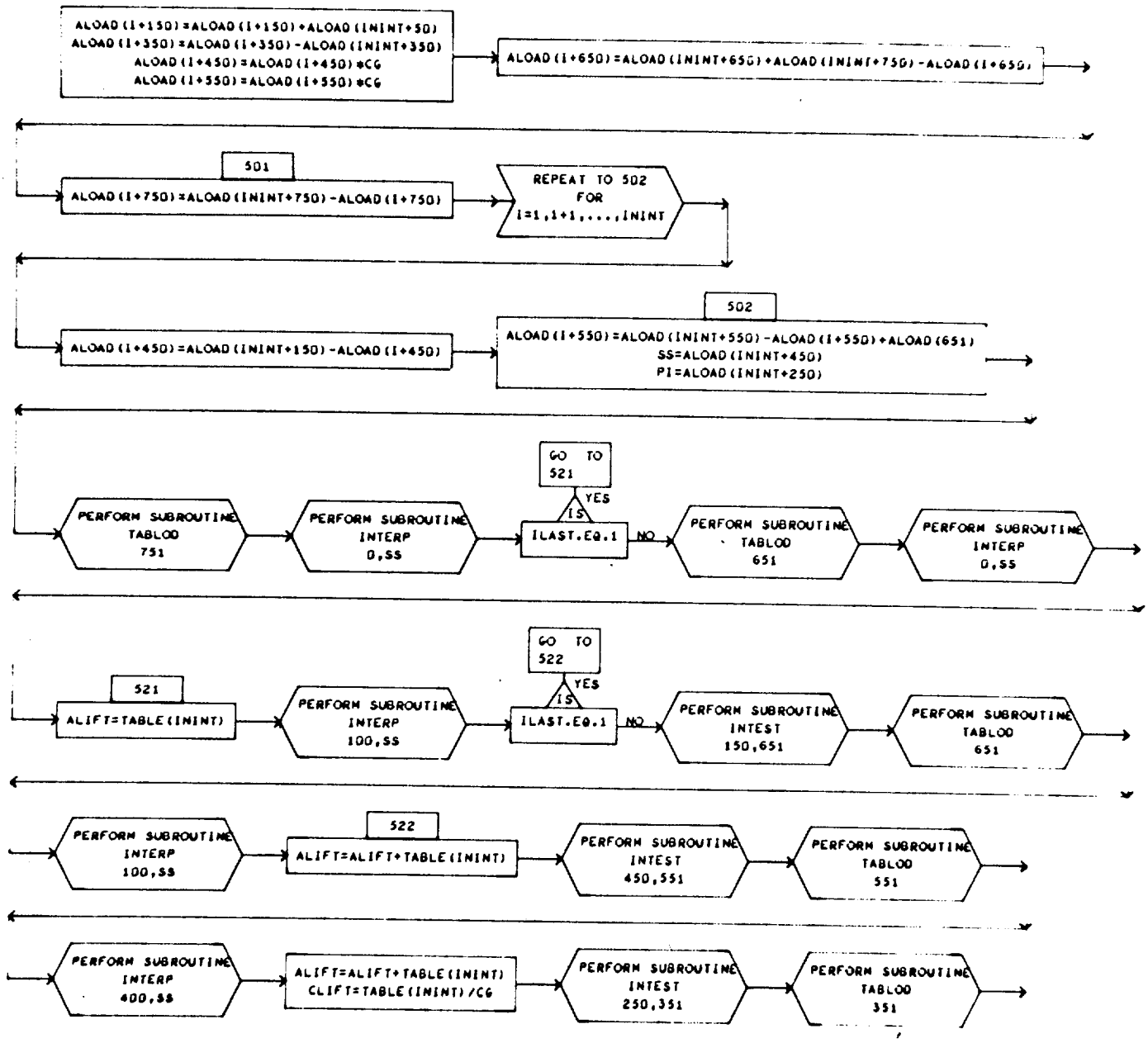

204
ALOAD(I+550)=ALOAD(I+550)+ALOAD(ININT+450)
REPEAT TO 205
FOR
I=1,I+1,...,ININT
205
ALOAD(I+850)=ALOAD(I+850)+ALOAD(ININT+550)
SS=ALOAD(951)-ALOAD(ININT+950)
PI=ALOAD(ININT+250)-ALOAD(251)
PO=ALOAD(ININT+650)-ALOAD(651)
PERFORM SUBROUTINE
INTEST
850,951
PERFORM SUBROUTINE
TABLOO
951
PERFORM SUBROUTINE
INTERP
0,SS
ALIFT=TABLE(ININT)
PERFORM SUBROUTINE
INTERP
100,SS
ALIFT=TABLE(ININT)+ALIFT
PERFORM SUBROUTINE
INTERP
400,SS
ALIFT=TABLE(ININT)+ALIFT
PERFORM SUBROUTINE
INTERP
500,SS
ALIFT=TABLE(ININT)+ALIFT
PERFORM SUBROUTINE
INTERP
800,SS
ALIFT=TABLE(ININT)+ALIFT
PERFORM SUBROUTINE
INTEST
250,351
PERFORM SUBROUTINE
TABLOO
351
PERFORM SUBROUTINE
INTERP
200,PI
BLIFT=TABLE(ININT)
PERFORM SUBROUTINE
INTEST
650,751
PERFORM SUBROUTINE
TABLOO
751
PERFORM SUBROUTINE
INTERP
600,PO
CLIFT=TABLE(ININT)
DUM1=BLIFT/ALIFT*PIREQ
DUM2=CLIFT/ALIFT*PORE0
B1
WRITE
WRITE(6,206)DUM1,DUM2
REPEAT TO 207
FOR
I=1,I+1,...,ININT
ALOAD(I)=ALOAD(I)/ALIFT
ALOAD(I+200)=ALOAD(I+200)/BLIFT
ALOAD(I+600)=ALOAD(I+600)/CLIFT
DUM=ELOAD(ALOAD(I+50))
207
B1
WRITE

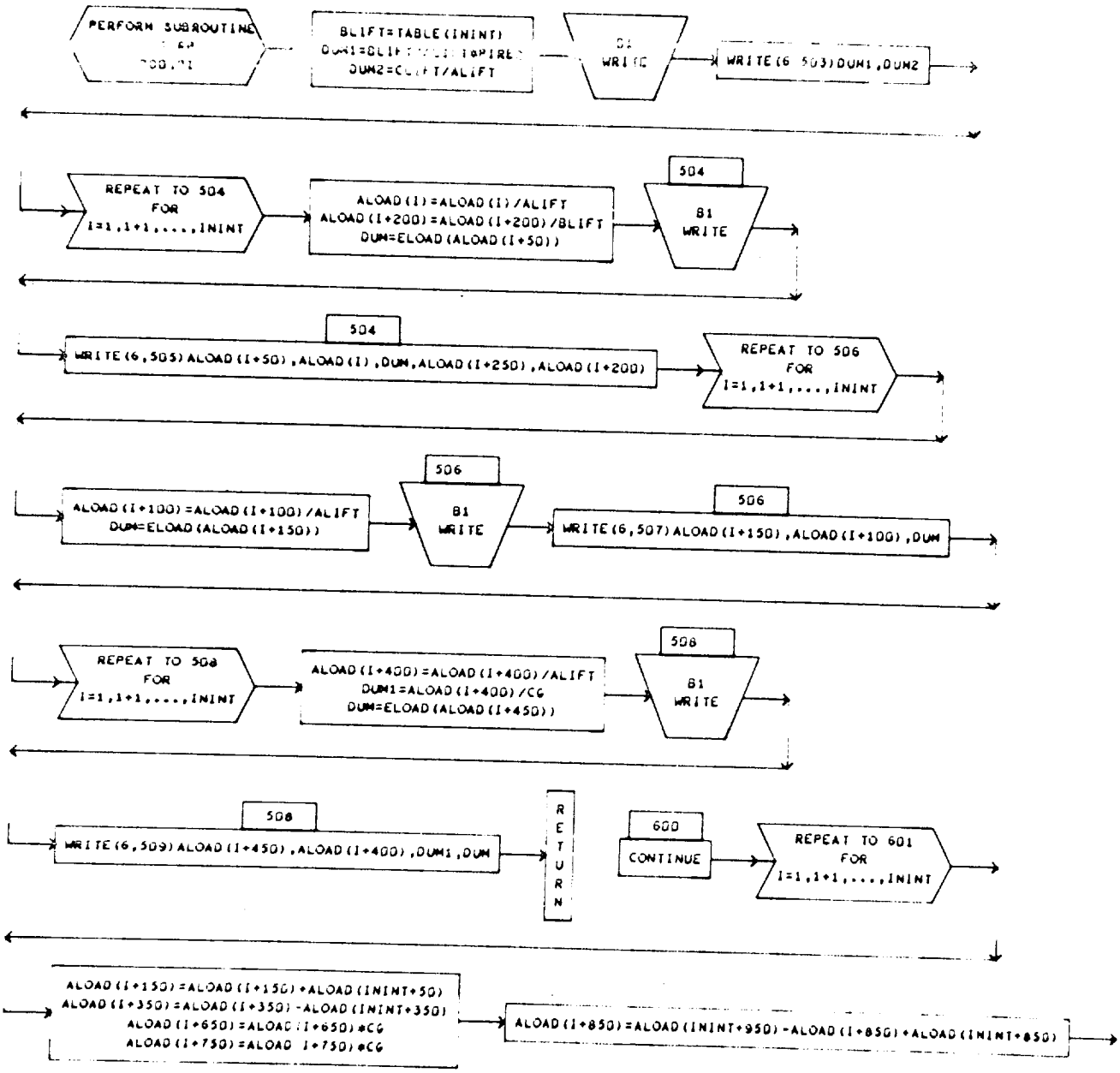

207
WRITE(6,208)ALOAD(I+50),ALOAD(I),DUM,ALOAD(I+250),ALOAD(I+200),ALOAD(I+650),ALOAD(I+600)
REPEAT TO 209 FOR I=1,1+1,...,ININT
209
B1 WRITE
ALOAD(I+100)=ALOAD(I+100)/ALIFT
DUM=ELOAD(ALOAD(I+150))
209
WRITE(6,210)ALOAD(I+150),ALOAD(I+100),DUM
REPEAT TO 211 FOR I=1,1+1,...,ININT
ALOAD(I+400)=ALOAD(I+400)/ALIFT
DUM=ELOAD(ALOAD(I+450))
211
B1 WRITE
211
WRITE(6,210)ALOAD(I+450),ALOAD(I+400),DUM
REPEAT TO 212 FOR I=1,1+1,...,ININT
ALOAD(I+500)=ALOAD(I+500)/ALIFT
DUM=ELOAD(ALOAD(I+550))
212
B1 WRITE
212
WRITE(6,210)ALOAD(I+550),ALOAD(I+500),DUM
REPEAT TO 213 FOR I=1,1+1,...,ININT
213
B1 WRITE
ALOAD(I+800)=ALOAD(I+800)/ALIFT
DUM=ELOAD(ALOAD(I+850))
213
WRITE(6,210)ALOAD(I+850),ALOAD(I+800),DUM
RETURN
300
CONTINUE
REPEAT TO 301 FOR I=1,1+1,...,ININT
ALOAD(I+150)=ALOAD(I+150)-ALOAD(ININT+150)
ALOAD(I+350)=(-ALOAD(I+350))-ALOAD(ININT+250)
ALOAD(I+250)=(-ALOAD(I+250))
ALOAD(I+450)=(-ALOAD(I+450))

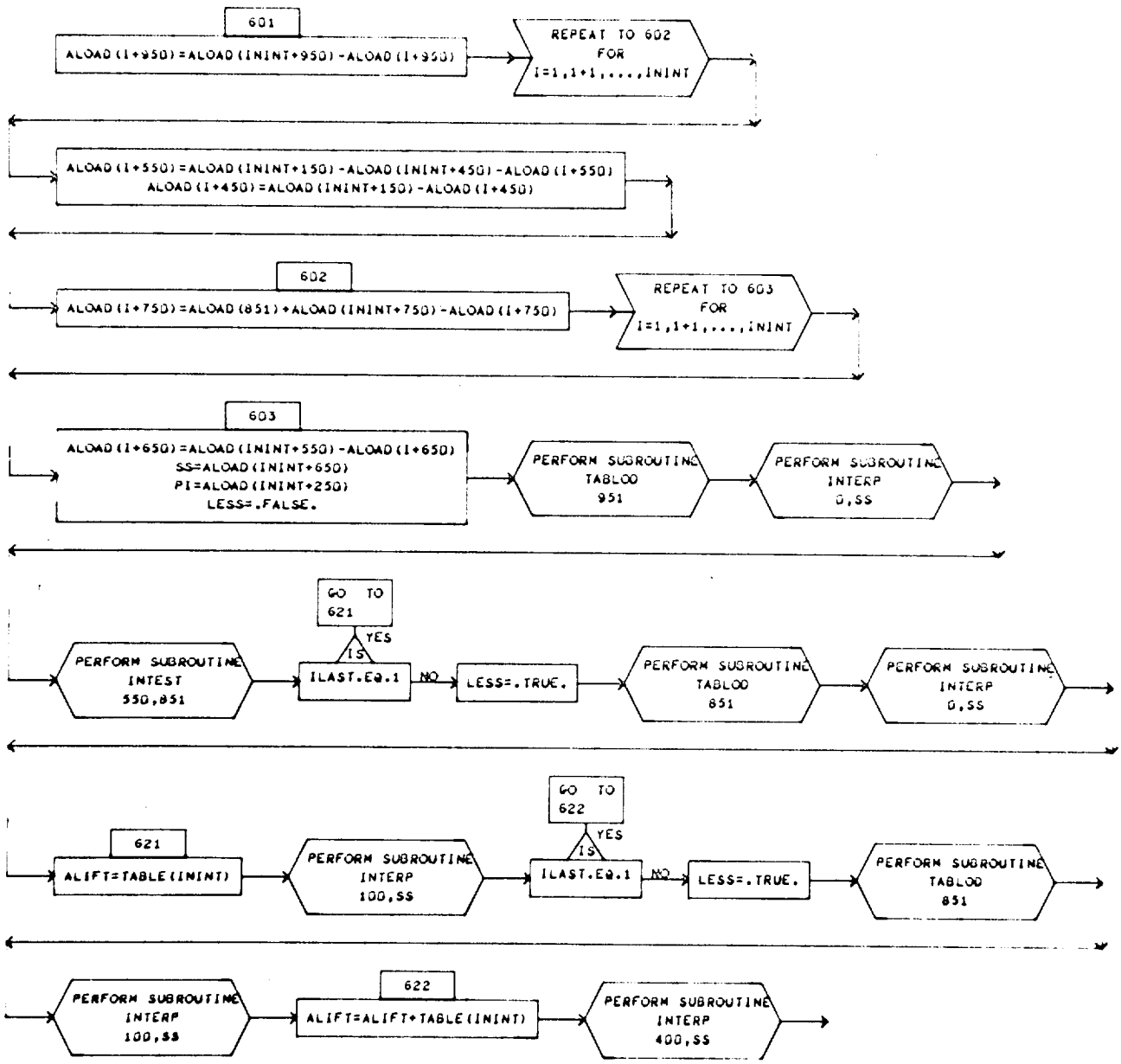

301
ALOAD(I+550)=ALOAD(ININT+550)-ALOAD(I+550)
SS=ALOAD(ININT+50)
PI=ALOAD(551)
PERFORM SUBROUTINE INTEST 50,151
PERFORM SUBROUTINE TABLOO 151
PERFORM SUBROUTINE INTERP 0,SS
ALIFT=TABLE(ININT)
PERFORM SUBROUTINE INTEST 450,551
PERFORM SUBROUTINE TABLOO 551
PERFORM SUBROUTINE INTERP 400,PI
BLIFT=TABLE(ININT)
DUM1=BLIFT/ALIFT*PIREQ
B1 WRITE
WRITE(6,302)DUM1
REPEAT TO 303 FOR I=1,I+1,...,ININT
303
ALOAD(I)=ALOAD(I)/ALIFT
ALOAD(I+400)=ALOAD(I+400)/BLIFT
DUM=ELOAD(ALOAD(I+50))
B1 WRITE
303
WRITE(6,304)ALOAD(I+50),ALOAD(I),DUM,ALOAD(I+450),ALOAD(I+400)
RETURN
400
CONTINUE
REPEAT TO 401 FOR I=1,I+1,...,ININT
ALOAD(I+150)=(-ALOAD(I+150))-ALOAD(ININT+50)
ALOAD(I+50)=(-ALOAD(I+50))
ALOAD(I+250)=(-ALOAD(I+250))
ALOAD(I+350)=ALOAD(ININT+350)-ALOAD(I+350)
ALOAD(I+650)=(-ALOAD(I+650))
ALOAD(I+750)=ALOAD(ININT+750)-ALOAD(I+750)
401
ALOAD(I+950)=ALOAD(ININT+950)-ALOAD(I+950)
REPEAT TO 402 FOR I=1,I+1,...,ININT
ALOAD(I+450)=ALOAD(I+450)+ALOAD(ININT+150)

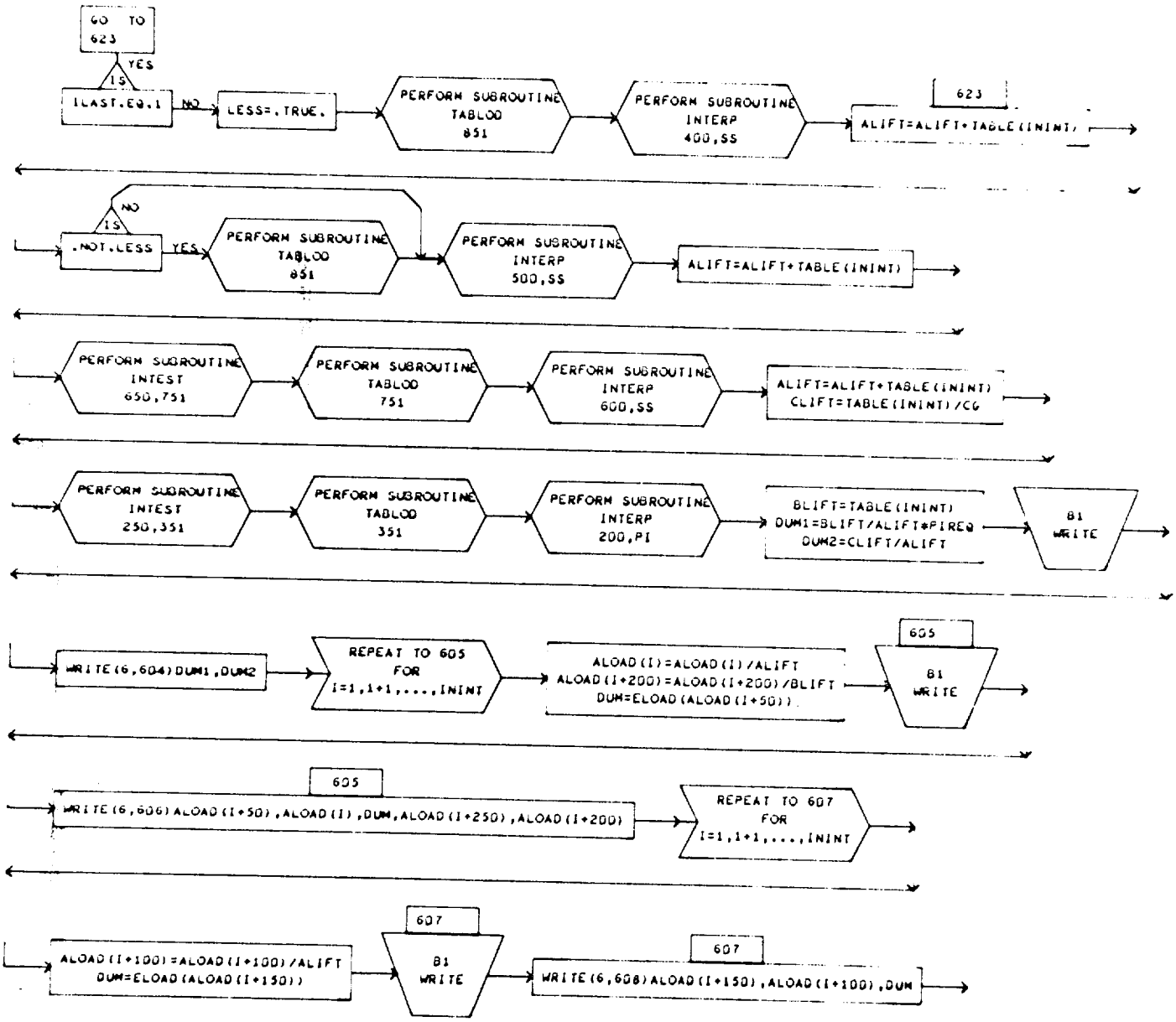

402
ALOAD(I+850)=ALOAD(951)-ALOAD(I+850)+ALOAD(ININT+850)
REPEAT TO 403
FOR
I=1,I+1,...,ININT
403
ALOAD(I+550)=ALOAD(I+550)-ALOAD(ININT+550)+ALOAD(851)
SS=ALOAD(ININT+450)
PI=ALOAD(ININT+250)
PO=ALOAD(ININT+650)
LESS=.TRUE.
PERFORM SUBROUTINE
TABLOO
951
PERFORM SUBROUTINE
INTERP
0,SS
GO TO
404
IS
YES
ILAST.EQ.1
NO
LESS=.FALSE.
PERFORM SUBROUTINE
TABLOO
851
PERFORM SUBROUTINE
INTERP
0,SS
404
ALIFT=TABLE(ININT)
PERFORM SUBROUTINE
INTERP
100,SS
GO TO
405
IS
YES
ILAST.EQ.1
NO
LESS=.FALSE.
PERFORM SUBROUTINE
TABLOO
851
PERFORM SUBROUTINE
INTERP
100,SS
405
ALIFT=ALIFT+TABLE(ININT)
IS
NO
LESS
YES
PERFORM SUBROUTINE
INTERP
400,SS
IS
NO
LESS
YES
PERFORM SUBROUTINE
TABLOO
851
PERFORM SUBROUTINE
INTERP
400,SS
PERFORM SUBROUTINE
INTEST
450,551
PERFORM SUBROUTINE
TABLOO
551
PERFORM SUBROUTINE
INTERP
400,SS
ALIFT=ALIFT+TABLE(ININT)
PERFORM SUBROUTINE
INTEST
250,351
PERFORM SUBROUTINE
TABLOO
351

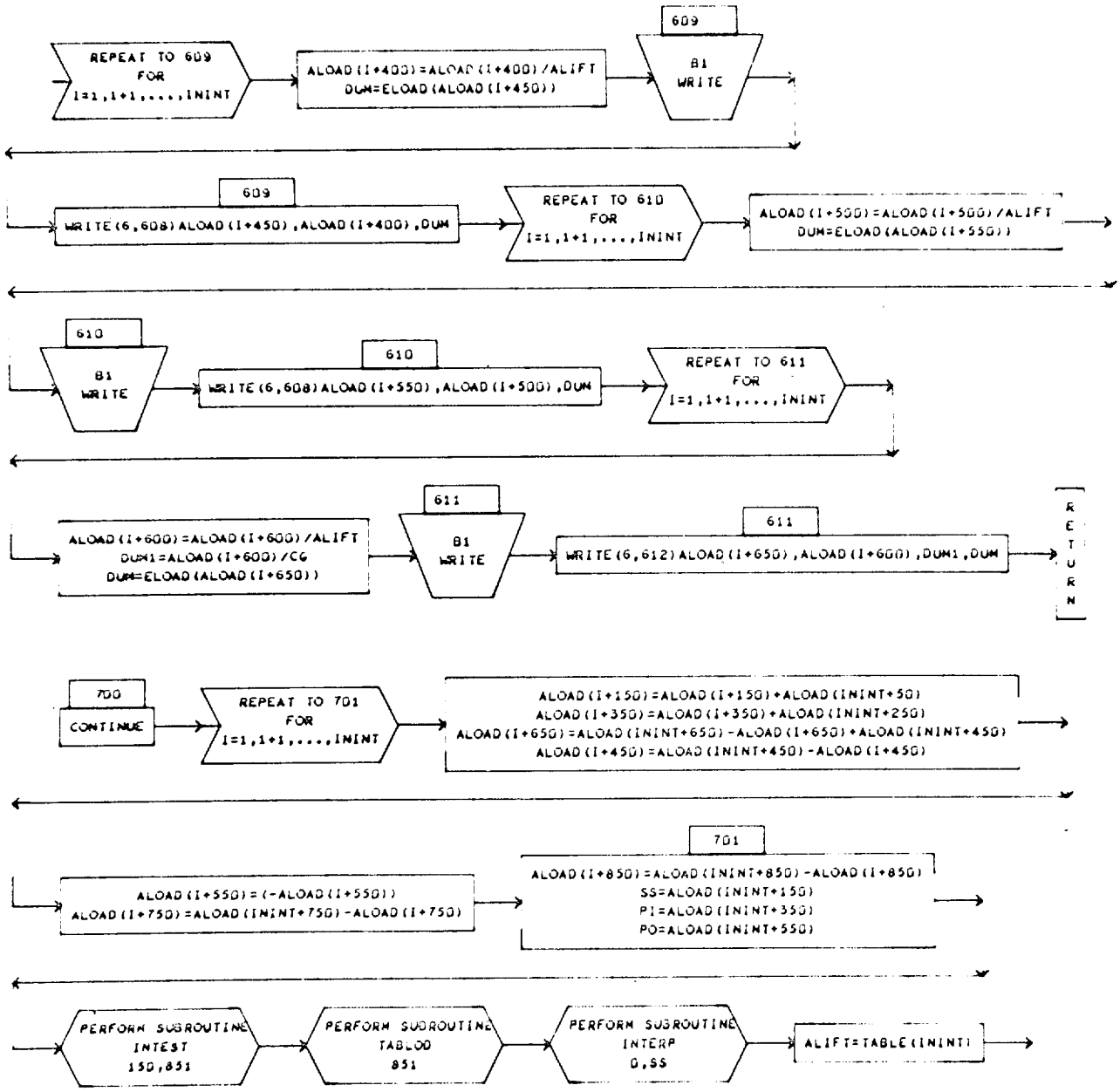

PERFORM SUBROUTINE INTERP 200,P1
BLIFT=TABLE(ININT)
PERFORM SUBROUTINE INTEST 650,751
PERFORM SUBROUTINE TABLOO 751
PERFORM SUBROUTINE INTERP 600,PO
CLIFT=TABLE(ININT)
DUM1=BLIFT/ALIFT*PIREQ
DUM2=CLIFT/ALIFT*PORE@
B1 WRITE
WRITE(6,406)DUM1,DUM2
REPEAT TO 407 FOR I=1,I+1,...,ININT
ALOAD(I)=ALOAD(I)/ALIFT
ALOAD(I+200)=ALOAD(I+200)/BLIFT
ALOAD(I+600)=ALOAD(I+600)/CLIFT
DUM=ELOAD(ALOAD(I+50))
407
B1 WRITE
407
WRITE(6,408)ALOAD(I+50),ALOAD(I),DUM,ALOAD(I+250),ALOAD(I+200),ALOAD(I+650),ALOAD(I+600)
REPEAT TO 409 FOR I=1,I+1,...,ININT
ALOAD(I+100)=ALOAD(I+100)/ALIFT
DUM=ELOAD(ALOAD(I+150))
409
B1 WRITE
409
WRITE(6,410)ALOAD(I+150),ALOAD(I+100),DUM
REPEAT TO 411 FOR I=1,I+1,...,ININT
ALOAD(I+400)=ALOAD(I+400)/ALIFT
DUM=ELOAD(ALOAD(I+450))
411
B1 WRITE
411
WRITE(6,410)ALOAD(I+450),ALOAD(I+400),DUM
RETURN
500
CONTINUE
REPEAT TO 501 FOR I=1,I+1,...,ININT

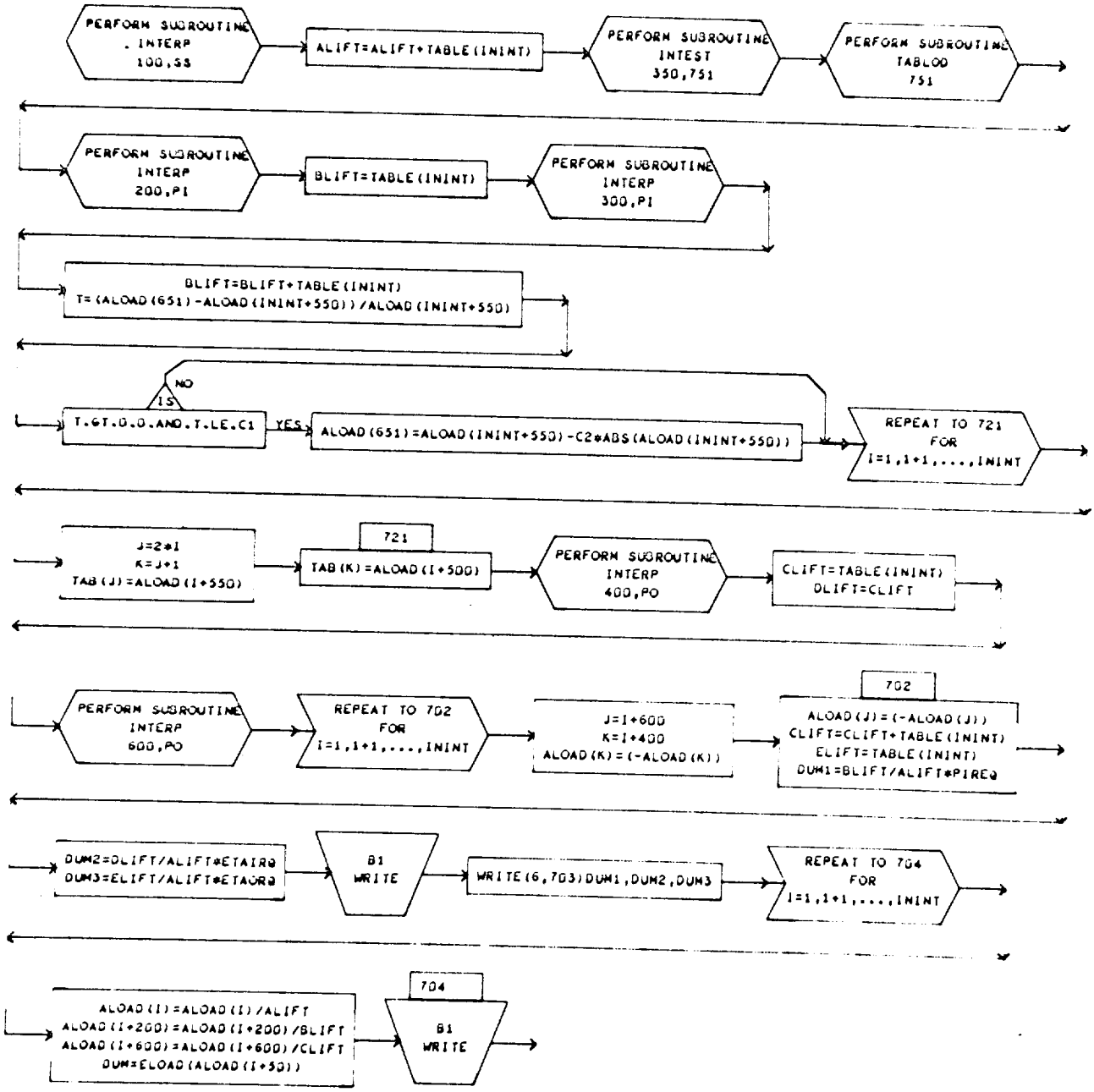

ALOAD(I+150)=ALOAD(I+150)+ALOAD(ININT+50)
ALOAD(I+350)=ALOAD(I+350)-ALOAD(ININT+350)
ALOAD(I+450)=ALOAD(I+450)*CG
ALOAD(I+550)=ALOAD(I+550)*CG
ALOAD(I+650)=ALOAD(ININT+650)+ALOAD(ININT+750)-ALOAD(I+650)
501
ALOAD(I+750)=ALOAD(ININT+750)-ALOAD(I+750)
REPEAT TO 502
FOR
I=1,I+1,...,ININT
ALOAD(I+450)=ALOAD(ININT+150)-ALOAD(I+450)
502
ALOAD(I+550)=ALOAD(ININT+550)-ALOAD(I+550)+ALOAD(651)
SS=ALOAD(ININT+450)
PI=ALOAD(ININT+250)
PERFORM SUBROUTINE
TABLOO
751
PERFORM SUBROUTINE
INTERP
0,SS
GO TO
521
YES
IS
ILAST.EQ.1
NO
PERFORM SUBROUTINE
TABLOO
651
PERFORM SUBROUTINE
INTERP
0,SS
521
ALIFT=TABLE(ININT)
PERFORM SUBROUTINE
INTERP
100,SS
GO TO
522
YES
IS
ILAST.EQ.1
NO
PERFORM SUBROUTINE
INTEST
150,651
PERFORM SUBROUTINE
TABLOO
651
PERFORM SUBROUTINE
INTERP
100,SS
522
ALIFT=ALIFT+TABLE(ININT)
PERFORM SUBROUTINE
INTEST
450,551
PERFORM SUBROUTINE
TABLOO
551
PERFORM SUBROUTINE
INTERP
400,SS
ALIFT=ALIFT+TABLE(ININT)
CLIFT=TABLE(ININT)/CG
PERFORM SUBROUTINE
INTEST
250,351
PERFORM SUBROUTINE
TABLOO
351

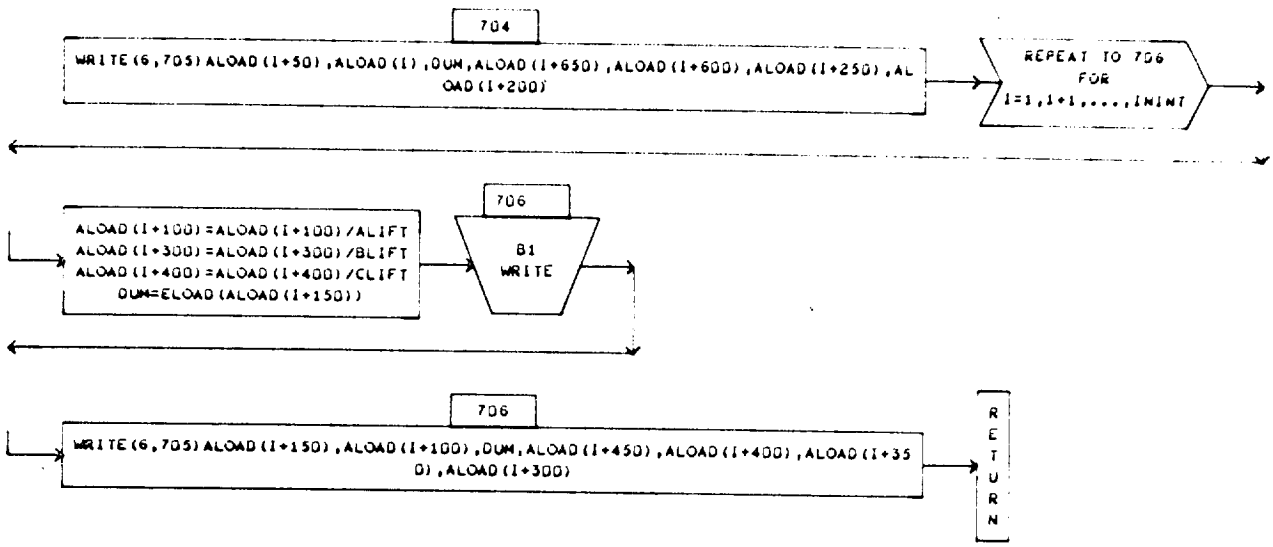

PERFORM SUBROUTINE
BLIFT=TABLE(ININT)
DUM2=CLIFT/ALIFT
WRITE
WRITE(6 503)DUM1,DUM2
REPEAT TO 504
FOR
I=1,I+1,...,ININT
ALOAD(I)=ALOAD(I)/ALIFT
ALOAD(I+200)=ALOAD(I+200)/BLIFT
DUM=ELOAD(ALOAD(I+50))
504
B1
WRITE
504
WRITE(6,505)ALOAD(I+50),ALOAD(I),DUM,ALOAD(I+250),ALOAD(I+200)
REPEAT TO 506
FOR
I=1,I+1,...,ININT
ALOAD(I+100)=ALOAD(I+100)/ALIFT
DUM=ELOAD(ALOAD(I+150))
506
B1
WRITE
506
WRITE(6,507)ALOAD(I+150),ALOAD(I+100),DUM
REPEAT TO 508
FOR
I=1,I+1,...,ININT
ALOAD(I+400)=ALOAD(I+400)/ALIFT
DUM1=ALOAD(I+400)/CG
DUM=ELOAD(ALOAD(I+450))
508
B1
WRITE
508
WRITE(6,509)ALOAD(I+450),ALOAD(I+400),DUM1,DUM
RETURN
600
CONTINUE
REPEAT TO 601
FOR
I=1,I+1,...,ININT
ALOAD(I+150)=ALOAD(I+150)+ALOAD(ININT+50)
ALOAD(I+350)=ALOAD(I+350)-ALOAD(ININT+350)
ALOAD(I+650)=ALOAD(I+650)*CG
ALOAD(I+750)=ALOAD(I+750)*CG
ALOAD(I+850)=ALOAD(ININT+950)-ALOAD(I+850)+ALOAD(ININT+850)


601
ALOAD(I+950)=ALOAD(ININT+950)-ALOAD(I+950)
REPEAT TO 602
FOR
I=1,1+1,...,ININT
ALOAD(I+550)=ALOAD(ININT+150)-ALOAD(ININT+450)-ALOAD(I+550)
ALOAD(I+450)=ALOAD(ININT+150)-ALOAD(I+450)
602
ALOAD(I+750)=ALOAD(851)+ALOAD(ININT+750)-ALOAD(I+750)
REPEAT TO 603
FOR
I=1,1+1,...,ININT
603
ALOAD(I+650)=ALOAD(ININT+550)-ALOAD(I+650)
SS=ALOAD(ININT+650)
PI=ALOAD(ININT+250)
LESS=.FALSE.
PERFORM SUBROUTINE
TABLOO
951
PERFORM SUBROUTINE
INTERP
0,SS
PERFORM SUBROUTINE
INTEST
550,851
GO TO
621
YES
IS
ILAST.EQ.1
NO
LESS=.TRUE.
PERFORM SUBROUTINE
TABLOO
851
PERFORM SUBROUTINE
INTERP
0,SS
621
ALIFT=TABLE(ININT)
PERFORM SUBROUTINE
INTERP
100,SS
GO TO
622
YES
IS
ILAST.EQ.1
NO
LESS=.TRUE.
PERFORM SUBROUTINE
TABLOO
851
PERFORM SUBROUTINE
INTERP
100,SS
622
ALIFT=ALIFT+TABLE(ININT)
PERFORM SUBROUTINE
INTERP
400,SS


GO TO 623
YES
IS
ILAST.EQ.1
NO
LESS=.TRUE.
PERFORM SUBROUTINE TABLOO 851
PERFORM SUBROUTINE INTERP 400,SS
623
ALIFT=ALIFT+TABLE(ININT)
NO
IS
.NOT.LESS
YES
PERFORM SUBROUTINE TABLOO 851
PERFORM SUBROUTINE INTERP 500,SS
ALIFT=ALIFT+TABLE(ININT)
PERFORM SUBROUTINE INTEST 650,751
PERFORM SUBROUTINE TABLOO 751
PERFORM SUBROUTINE INTERP 600,SS
ALIFT=ALIFT+TABLE(ININT)
CLIFT=TABLE(ININT)/CG
PERFORM SUBROUTINE INTEST 250,351
PERFORM SUBROUTINE TABLOO 351
PERFORM SUBROUTINE INTERP 200,PI
BLIFT=TABLE(ININT)
DUM1=BLIFT/ALIFT*FIREQ
DUM2=CLIFT/ALIFT
B1 WRITE
WRITE(6,604)DUM1,DUM2
REPEAT TO 605 FOR I=1,I+1,...,ININT
ALOAD(I)=ALOAD(I)/ALIFT
ALOAD(I+200)=ALOAD(I+200)/BLIFT
DUM=ELOAD(ALOAD(I+50))
605
B1 WRITE
605
WRITE(6,606)ALOAD(I+50),ALOAD(I),DUM,ALOAD(I+250),ALOAD(I+200)
REPEAT TO 607 FOR I=1,I+1,...,ININT
ALOAD(I+100)=ALOAD(I+100)/ALIFT
DUM=ELOAD(ALOAD(I+150))
607
B1 WRITE
607
WRITE(6,608)ALOAD(I+150),ALOAD(I+100),DUM


REPEAT TO 609
FOR
I=1,1+1,...,ININT
ALOAD(I+400)=ALOAD(I+400)/ALIFT
DUM=ELOAD(ALOAD(I+450))
609
B1
WRITE
609
WRITE(6,608)ALOAD(I+450),ALOAD(I+400),DUM
REPEAT TO 610
FOR
I=1,1+1,...,ININT
ALOAD(I+500)=ALOAD(I+500)/ALIFT
DUM=ELOAD(ALOAD(I+550))
610
B1
WRITE
610
WRITE(6,608)ALOAD(I+550),ALOAD(I+500),DUM
REPEAT TO 611
FOR
I=1,1+1,...,ININT
ALOAD(I+600)=ALOAD(I+600)/ALIFT
DUM1=ALOAD(I+600)/CG
DUM=ELOAD(ALOAD(I+650))
611
B1
WRITE
611
WRITE(6,612)ALOAD(I+650),ALOAD(I+600),DUM1,DUM
RETURN
700
CONTINUE
REPEAT TO 701
FOR
I=1,1+1,...,ININT
ALOAD(I+150)=ALOAD(I+150)+ALOAD(ININT+50)
ALOAD(I+350)=ALOAD(I+350)+ALOAD(ININT+250)
ALOAD(I+650)=ALOAD(ININT+650)-ALOAD(I+650)+ALOAD(ININT+450)
ALOAD(I+450)=ALOAD(ININT+450)-ALOAD(I+450)
ALOAD(I+550)=(-ALOAD(I+550))
ALOAD(I+750)=ALOAD(ININT+750)-ALOAD(I+750)
701
ALOAD(I+850)=ALOAD(ININT+850)-ALOAD(I+850)
SS=ALOAD(ININT+150)
PI=ALOAD(ININT+350)
PO=ALOAD(ININT+550)
PERFORM SUBROUTINE
INTEST
150,851
PERFORM SUBROUTINE
TABLOO
851
PERFORM SUBROUTINE
INTERP
0,SS
ALIFT=TABLE(ININT)


PERFORM SUBROUTINE
. INTERP
100,SS
ALIFT=ALIFT+TABLE(ININT)
PERFORM SUBROUTINE
INTEST
350,751
PERFORM SUBROUTINE
TABLOD
751
PERFORM SUBROUTINE
INTERP
200,PI
BLIFT=TABLE(ININT)
PERFORM SUBROUTINE
INTERP
300,PI
BLIFT=BLIFT+TABLE(ININT)
T=(ALOAD(651)-ALOAD(ININT+550))/ALOAD(ININT+550)
NO
IS
T.GT.0.0.AND.T.LE.C1
YES
ALOAD(651)=ALOAD(ININT+550)-C2*ABS(ALOAD(ININT+550))
REPEAT TO 721
FOR
I=1,1+1,...,ININT
J=2*I
K=J+1
TAB(J)=ALOAD(I+550)
721
TAB(K)=ALOAD(I+500)
PERFORM SUBROUTINE
INTERP
400,PO
CLIFT=TABLE(ININT)
DLIFT=CLIFT
PERFORM SUBROUTINE
INTERP
600,PO
REPEAT TO 702
FOR
I=1,1+1,...,ININT
J=I+600
K=I+400
ALOAD(K)=(-ALOAD(K))
702
ALOAD(J)=(-ALOAD(J))
CLIFT=CLIFT+TABLE(ININT)
ELIFT=TABLE(ININT)
DUM1=BLIFT/ALIFT*PIREQ
DUM2=DLIFT/ALIFT*ETAIRQ
DUM3=ELIFT/ALIFT*ETAORQ
B1
WRITE
WRITE(6,703)DUM1,DUM2,DUM3
REPEAT TO 704
FOR
I=1,1+1,...,ININT
ALOAD(I)=ALOAD(I)/ALIFT
ALOAD(I+200)=ALOAD(I+200)/BLIFT
ALOAD(I+600)=ALOAD(I+600)/CLIFT
DUM=ELOAD(ALOAD(I+50))
704
B1
WRITE


704
WRITE(6,705)ALOAD(I+50),ALOAD(I),DUM,ALOAD(I+650),ALOAD(I+600),ALOAD(I+250),ALOAD(I+200)
REPEAT TO 706
FOR
I=1,I+1,...,ININT
ALOAD(I+100)=ALOAD(I+100)/ALIFT
ALOAD(I+300)=ALOAD(I+300)/BLIFT
ALOAD(I+400)=ALOAD(I+400)/CLIFT
DUM=ELOAD(ALOAD(I+150))
706
B1
WRITE
706
WRITE(6,705)ALOAD(I+150),ALOAD(I+100),DUM,ALOAD(I+450),ALOAD(I+400),ALOAD(I+350),ALOAD(I+300)
RETURN

SYMBOLS USED IN SUBROUTINE LOAD (DECK IDENTIFICATION 55VW)

A	R	C	POTENTIAL AT A (MAPPING CONSTANT)	LOAD
AA	R	C	ARRAY FOR RESULTS OF GEOMETRY AND DRAG INTEGRALS	LOAD
ABSC	R	C	ABSCISSA ARRAY FOR NUMERICAL INTEGRALS	LOAD
AI	R	C	POTENTIAL AT I (MAPPING CONSTANT)	LOAD
ALIFT	R	U	VALUE OF LIFT INTEGRAL	LOAD
ALOAD	R	C	STORAGE ARRAY FOR POTENTIAL IN THE REAL (PHYSICAL) PLANE	LOAD
B	R	C	POTENTIAL AT B (MAPPING CONSTANT)	LOAD
BLIFT	R	U	VALUE OF LIFT INTEGRAL	LOAD
C	R	C	POTENTIAL AT C (MAPPING CONSTANT)	LOAD
CG	R	C	COSINE OF DIHEDRAL ANGLE	LOAD
CLIFT	R	U	VALUE OF LIFT INTEGRAL	LOAD
CONI	R	C	FACTOR FOR IMAJINARY PART OF COMPLEX DISTANCE	LOAD
CONR	R	C	FACTOR FOR REAL PART OF COMPLEX DISTANCE	LOAD
C1	R	C	CONSTANT THAT IS COMPUTER HARDWARE DEPENDENT	LOAD
C2	R	C	CONSTANT THAT IS COMPUTER HARDWARE DEPENDENT	LOAD
C3	R	C	CONSTANT THAT IS COMPUTER HARDWARE DEPENDENT	LOAD
D	R	C	POTENTIAL AT D (MAPPING CONSTANT)	LOAD
DATA	R	C	ARRAY FOR INPUT DATA	LOAD
DLIFT	R	U	VALUE OF LIFT INTEGRAL	LOAD
DUM	R	U	CALCULATION VARIABLE	LOAD
DUM1	R	U	CALCULATION VARIABLE	LOAD
DUM2	R	U	CALCULATION VARIABLE	LOAD
DUM3	R	U	CALCULATION VARIABLE	LOAD
E	R	C	POTENTIAL AT E (MAPPING CONSTANT)	LOAD
ELIFT	R	U	VALUE OF LIFT INTEGRAL	LOAD
ETAIRG	R	C	INBOARD SPANWISE LOCATION (NONDIMENSIONAL)	LOAD
ETAORG	R	C	OUTBOARD SPANWISE LOCATION (NONDIMENSIONAL)	LOAD
F	R	C	POTENTIAL AT F (MAPPING CONSTANT)	LOAD
G	R	C	POTENTIAL AT G (MAPPING CONSTANT)	LOAD
GAMMA	R	C	DIHEDRAL ANGLE IN DEGREES	LOAD
GP	R	C	DIHEDRAL ANGLE DIVIDED BY PI	LOAD
GPM2	R	C	GP - 0.5	LOAD
GPP2	R	C	GP + 0.5	LOAD
H	R	C	POTENTIAL AT H (MAPPING CONSTANT)	LOAD
I	I	U	DO LOOP INDEX	LOAD
IDUM	I	U	CALCULATION VARIABLE	LOAD
ILAST	I	C	FLAG FOR LAST ITERATION	LOAD
ININT	I	C	NUMBER OF ORDINATES IN NUMERICAL INTEGRALS	LOAD

SYMBOLS USED IN SUBROUTINE LOAD (DECK IDENTIFICATION 55VW)

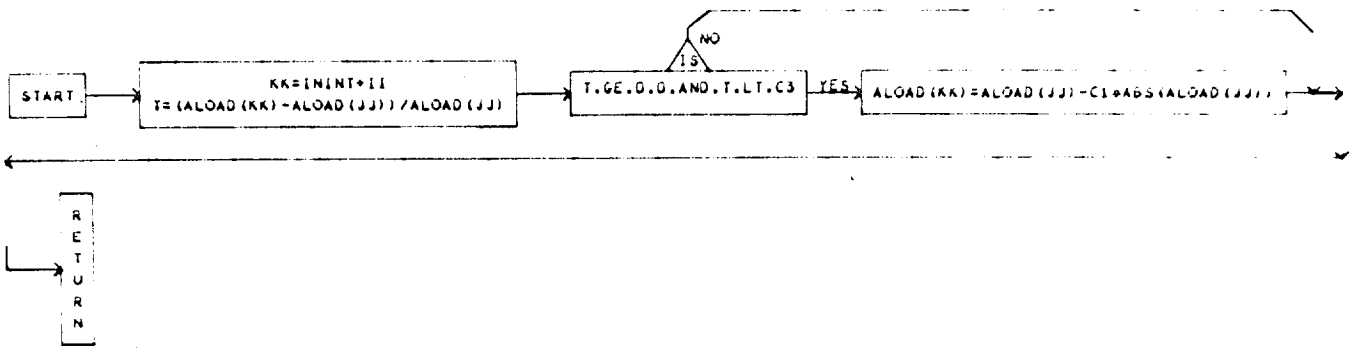
J	I	U	CALCULATION VARIABLE	LOAD
K	I	U	CALCULATION VARIABLE	LOAD
LESS	L	U	INTERPOLATION POSITION FLAG	LOAD
MODE	I	C	PROGRAM MODE OF OPERATION	LOAD
NN	I	C	NUMBER OF NUMERICAL INTEGRAL ORDINATES	LOAD
NTYPE	I	C	CONFIGURATION TYPE	LOAD
OMGP	R	C	1.0 - GP	LOAD
ORD	R	C	ORDINATE ARRAY FOR NUMERICAL INTEGRALS	LOAD
PI	R	U	INBOARD PYLON OR FENCE LENGTH	LOAD
PIREQ	R	C	INBOARD LENGTH OR DISTANCE (NONDIMENSIONAL)	LOAD
PL	R	U	PYLON OR FENCE LENGTH	LOAD
PO	R	U	OUTBOARD PYLON OR FENCE LENGTH	LOAD
POREQ	R	C	OUTBOARD LENGTH OR DISTANCE (NONDIMENSIONAL)	LOAD
R1	R	C	SQUARE ROOT OF SUM OF SQUARES OF ERRORS IN REQD CONDITIONS	LOAD
SS	R	U	SEMISPAN	LOAD
TAB	R	C	INTERPOLATION ARRAY FOR INDEPENDENT VARIABLE AND FUNCTION	LOAD
TABLE	R	C	INTEGRATED VALUES OF ORDINATES AS A FUNCTION OF ABSCISSA	LOAD

Subroutine INTEST (II, JJ) (Deck 55VX). Subroutine INTEST tests the data in the ALOAD array and modifies it, if necessary, to improve the interpolation for loading. The last, and hence, largest element of the dependent-interpolation array is tested. If it lies only a small amount above the range of the independent interpolation array, it is set to a value just inside the latter array.

```
                SUBROUTINE INTEST (II, JJ)                                      55VX0010
                                                                                55VX0020
          C
          C     THIS SUBROUTINE TESTS THE LAST (AND HENCE, LARGEST) ELEMENT OF  55VX0030
          C     THE DEPENDENT INTERPOLATION ARRAY TO SEE IF IT LIES ONLY SLIGHTLY 55VX0040
          C     ABOVE THE RANGE OF THE INDEPENDENT INTERPOLATION ARRAY.  IF SO, 55VX0050
          C     IT IS SET TO A VALUE JUST INSIDE THE RANGE OF THE LATTER ARRAY. 55VX0060
          C                                                                     55VX0070
000004          COMMON DATA(17), ABSC(50), ORD(50), TABLE(50), ALOAD(1000), AA(16)55VX0080
               A , TAB(101), GP, GPM2, GPP2, CG, OMGP, CONR, CONI, ILAST, C1, C2, 55VX0090
               B   C3, R1                                                       55VX0100
                                                                                55VX0110
          C
000004          EQUIVALENCE                (DATA( 1),      A) , (DATA( 2),      B)  55VX0120
               A , (DATA( 3),      C) , (DATA( 4),      D) , (DATA( 5),      E)  55VX0130
               B , (DATA( 6),      F) , (DATA( 7),      G) , (DATA( 8),      H)  55VX0140
               C , (DATA( 9),     AI) , (DATA(10),   MODE) , (DATA(11),  NTYPE)  55VX0150
               D , (DATA(12),  ININT) , (DATA(13), ETAIRQ) , (DATA(14), ETAORQ)  55VX0160
               E , (DATA(15),  PIREQ) , (DATA(16),  POREQ) , (DATA(17),  GAMMA)  55VX0170
                                                                                55VX0180
          C
000004          KK = ININT + II                                                 55VX0190
000005          T = (ALOAD(KK) - ALOAD(JJ)) / ALOAD(JJ)                         55VX0200
000012          IF (T .GE. 0.0 .AND. T .LT. C3) ALOAD(KK) = ALOAD(JJ)           55VX0210
               A                                        - C1 * ABS(ALOAD(JJ))   55VX0220
                                                                                55VX0230
000027          RETURN                                                          55VX0240
000030          END
```

```
SUBPROGRAM LENGTH
000051

FUNCTION ASSIGNMENTS

STATEMENT ASSIGNMENTS

BLOCK NAMES AND LENGTHS
        -  002420

VARIABLE ASSIGNMENTS
AA      -  002217C01 ABSC    -  000021C01 ALOAD   -  000247C01 C1      -  002414C01
C3      -  002416C01 DATA    -  000000C01 ININT   -  000013C01 KK      -  000047
ORD     -  000103C01 T       -  000050    TAB     -  002237C01 TABLE   -  000165C01

START OF CONSTANTS
000032

START OF TEMPORARIES
000033

START OF INDIRECTS
000043

UNUSED COMPILER SPACE
043300
```

SUBROUTINE INTEST (II, JJ)



SYMBOLS USED IN SUBROUTINE INTEST (DECK IDENTIFICATION 55VX)

A	R	C	POTENTIAL AT A (MAPPING CONSTANT)	INTEST
AA	R	C	ARRAY FOR RESULTS OF GEOMETRY AND DRAG INTEGRALS	INTEST
ABSC	R	C	ABSCISSA ARRAY FOR NUMERICAL INTEGRALS	INTEST
AI	R	C	POTENTIAL AT I (MAPPING CONSTANT)	INTEST
ALOAD	R	C	STORAGE ARRAY FOR POTENTIAL IN THE REAL (PHYSICAL) PLANE	INTEST
B	R	C	POTENTIAL AT B (MAPPING CONSTANT)	INTEST
C	R	C	POTENTIAL AT C (MAPPING CONSTANT)	INTEST
CG	R	C	COSINE OF DIHEDRAL ANGLE	INTEST
CONI	R	C	FACTOR FOR IMAJINARY PART OF COMPLEX DISTANCE	INTEST
CONR	R	C	FACTOR FOR REAL PART OF COMPLEX DISTANCE	INTEST
C1	R	C	CONSTANT THAT IS COMPUTER HARDWARE DEPENDENT	INTEST
C2	R	C	CONSTANT THAT IS COMPUTER HARDWARE DEPENDENT	INTEST
C3	R	C	CONSTANT THAT IS COMPUTER HARDWARE DEPENDENT	INTEST
D	R	C	POTENTIAL AT D (MAPPING CONSTANT)	INTEST
DATA	R	C	ARRAY FOR INPUT DATA	INTEST
E	R	C	POTENTIAL AT E (MAPPING CONSTANT)	INTEST
ETAIRG	R	C	INBOARD SPANWISE LOCATION (NONDIMENSIONAL)	INTEST
ETAORG	R	C	OUTBOARD SPANWISE LOCATION (NONDIMENSIONAL)	INTEST
F	R	C	POTENTIAL AT F (MAPPING CONSTANT)	INTEST
G	R	C	POTENTIAL AT G (MAPPING CONSTANT)	INTEST
GAMMA	R	C	DIHEDRAL ANGLE IN DEGREES	INTEST
GP	R	C	DIHEDRAL ANGLE DIVIDED BY PI	INTEST
GPM2	R	C	GP - 0.5	INTEST
GPP2	R	C	GP + 0.5	INTEST
H	R	C	POTENTIAL AT H (MAPPING CONSTANT)	INTEST
II	I	U	LOCATION IN ALOAD ARRAY	INTEST
ILAST	I	C	FLAG FOR LAST ITERATION	INTEST
ININT	I	C	NUMBER OF ORDINATES IN NUMERICAL INTEGRALS	INTEST
JJ	I	U	LOCATION IN ALOAD ARRAY	INTEST
KK	I	U	CALCULATION VARIABLE	INTEST
MODE	I	C	PROGRAM MODE OF OPERATION	INTEST
NTYPE	I	C	CONFIGURATION TYPE	INTEST
OMGP	R	C	1.0 - GP	INTEST
ORD	R	C	ORDINATE ARRAY FOR NUMERICAL INTEGRALS	INTEST
PIREG	R	C	INBOARD LENGTH OR DISTANCE (NONDIMENSIONAL)	INTEST
POREG	R	C	OUTBOARD LENGTH OR DISTANCE (NONDIMENSIONAL)	INTEST
R1	R	C	SQUARE ROOT OF SUM OF SQUARES OF ERRORS IN REQD CONDITIONS	INTEST
T	R	U	CALCULATION VARIABLE	INTEST

SYMBOLS USED IN SUBROUTINE INTEST (DECK IDENTIFICATION 55VX)

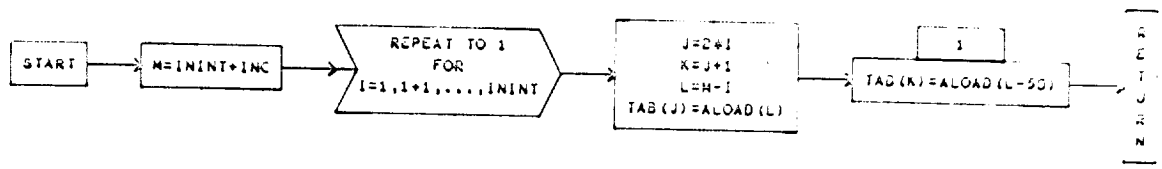
TAB	R	C	INTERPOLATION ARRAY FOR INDEPENDENT VARIABLE AND FUNCTION	INTEST
TABLE	R	C	INTEGRATED VALUES OF ORDINATES AS A FUNCTION OF ABSCISSA	INTEST

Subroutine TABLOD (INC) (Deck 55VY). The independent interpolation data are transferred from the ALOAD array to the TAB array by this subroutine.

```
                SUBROUTINE TABLOD (INC)                                         55VY0010
          C                                                                     55VY0020
          C     THIS SUBROUTINE SETS UP THE INTERPOLATION ARRAY TAB IN ORDER OF 55VY0030
          C     INCREASING ABSCISSA ELEMENTS.  THE SET-UP CONFORMS TO THE REQUIRE-55VY0040
          C     MENTS OF THE STANDARD DOUGLAS INTERPOLATION ROUTINE INS1.       55VY0050
          C                                                                     55VY0060
000000          COMMON DATA(17), ABSC(50), ORD(50), TABLE(50), ALOAD(1000), AA(16)55VY0070
                A , TAB(101), GP, GPM2, GPP2, CG, OMGP, CONR, CONI, ILAST, C1, C2 55VY0080
                B , C3, R1                                                      55VY0090
          C                                                                     55VY0100
000003          EQUIVALENCE              (DATA( 1),      A) , (DATA( 2),      B) 55VY0110
                A , (DATA( 3),      C) , (DATA( 4),      D) , (DATA( 5),      E) 55VY0120
                B , (DATA( 6),      F) , (DATA( 7),      G) , (DATA( 8),      H) 55VY0130
                C , (DATA( 9),     AI) , (DATA(10),   MODE) , (DATA(11),  NTYPE) 55VY0140
                D , (DATA(12),  ININT) , (DATA(13), ETAIRQ) , (DATA(14), ETAORQ) 55VY0150
                E , (DATA(15),  PIREQ) , (DATA(16),  POREQ) , (DATA(17),  GAMMA) 55VY0160
          C                                                                     55VY0170
000003          M = ININT + INC                                                 55VY0180
000005          DO 1 I=1,ININT                                                  55VY0190
000006          J = 2 * I                                                       55VY0200
000007          K = J + 1                                                       55VY0210
000011          L = M - I                                                       55VY0220
000013          TAB(J) = ALOAD(L)                                               55VY0230
000016        1 TAB(K) = ALOAD(L-50)                                            55VY0240
000023          RETURN                                                          55VY0250
000023          END                                                             55VY0260
```

```
SUBPROGRAM LENGTH
000041

FUNCTION ASSIGNMENTS

STATEMENT ASSIGNMENTS

BLOCK NAMES AND LENGTHS
       - 002420

VARIABLE ASSIGNMENTS
AA      - 002217C01 ABSC    - 000021C01 ALOAD   - 000247C01 DATA    - 000000C01
I       - 000035    ININT   - 000013C01 J       - 000036    K       - 000037
L       - 000040    M       - 000034    ORD     - 000103C01 TAB     - 002237C01
TABLE   - 000165C01

START OF CONSTANTS
000025

START OF TEMPORARIES
000026

START OF INDIRECTS
000030

UNUSED COMPILER SPACE
043400
```

SUBROUTINE TABLOO (INC)



SYMBOLS USED IN SUBROUTINE TABLOD (DECK IDENTIFICATION 55YY)

AA	R	C	ARRAY FOR RESULTS OF GEOMETRY AND DRAG INTEGRALS	TABLOD
ABSC	R	C	ABSCISSA ARRAY FOR NUMERICAL INTEGRALS	TABLOD
ALOAD	R	C	STORAGE ARRAY FOR POTENTIAL IN THE REAL (PHYSICAL) PLANE	TABLOD
CG	R	C	COSINE OF DIHEDRAL ANGLE	TABLOD
CONI	R	C	FACTOR FOR IMAJINARY PART OF COMPLEX DISTANCE	TABLOD
CONR	R	C	FACTOR FOR REAL PART OF COMPLEX DISTANCE	TABLOD
C1	R	C	CONSTANT THAT IS COMPUTER HARDWARE DEPENDENT	TABLOD
C2	R	C	CONSTANT THAT IS COMPUTER HARDWARE DEPENDENT	TABLOD
C3	R	C	CONSTANT THAT IS COMPUTER HARDWARE DEPENDENT	TABLOD
DATA	R	C	ARRAY FOR INPUT DATA	TABLOD
GP	R	C	DIHEDRAL ANGLE DIVIDED BY PI	TABLOD
GPM2	R	C	GP - 0.5	TABLOD
GPP2	R	C	GP + 0.5	TABLOD
I	I	U	DO LOOP INDEX	TABLOD
ILAST	I	C	FLAG FOR LAST ITERATION	TABLOD
INC	I	U	LOCATION IN ALOAD ARRAY	TABLOD
GPM2	R	C	GP - 0.5	TABLOD
GPP2	R	C	GP + 0.5	TABLOD
I	I	U	DO LOOP INDEX	TABLOD
ILAST	I	C	FLAG FOR LAST ITERATION	TABLOD
INC	I	U	LOCATION IN ALOAD ARRAY	TABLOD
J	I	U	CALCULATION VARIABLE	TABLOD
K	I	U	CALCULATION VARIABLE	TABLOD
L	I	U	CALCULATION VARIABLE	TABLOD
M	I	U	CALCULATION VARIABLE	TABLOD
OMGP	R	C	1.0 - GP	TABLOD
ORD	R	C	ORDINATE ARRAY FOR NUMERICAL INTEGRALS	TABLOD
R1	R	C	SQUARE ROOT OF SUM OF SQUARES OF ERRORS IN REQD CONDITIONS	TABLOD
TAB	R	C	INTERPOLATION ARRAY FOR INDEPENDENT VARIABLE AND FUNCTION	TABLOD
TABLE	R	C	INTEGRATED VALUES OF ORDINATES AS A FUNCTION OF ABSCISSA	TABLOD

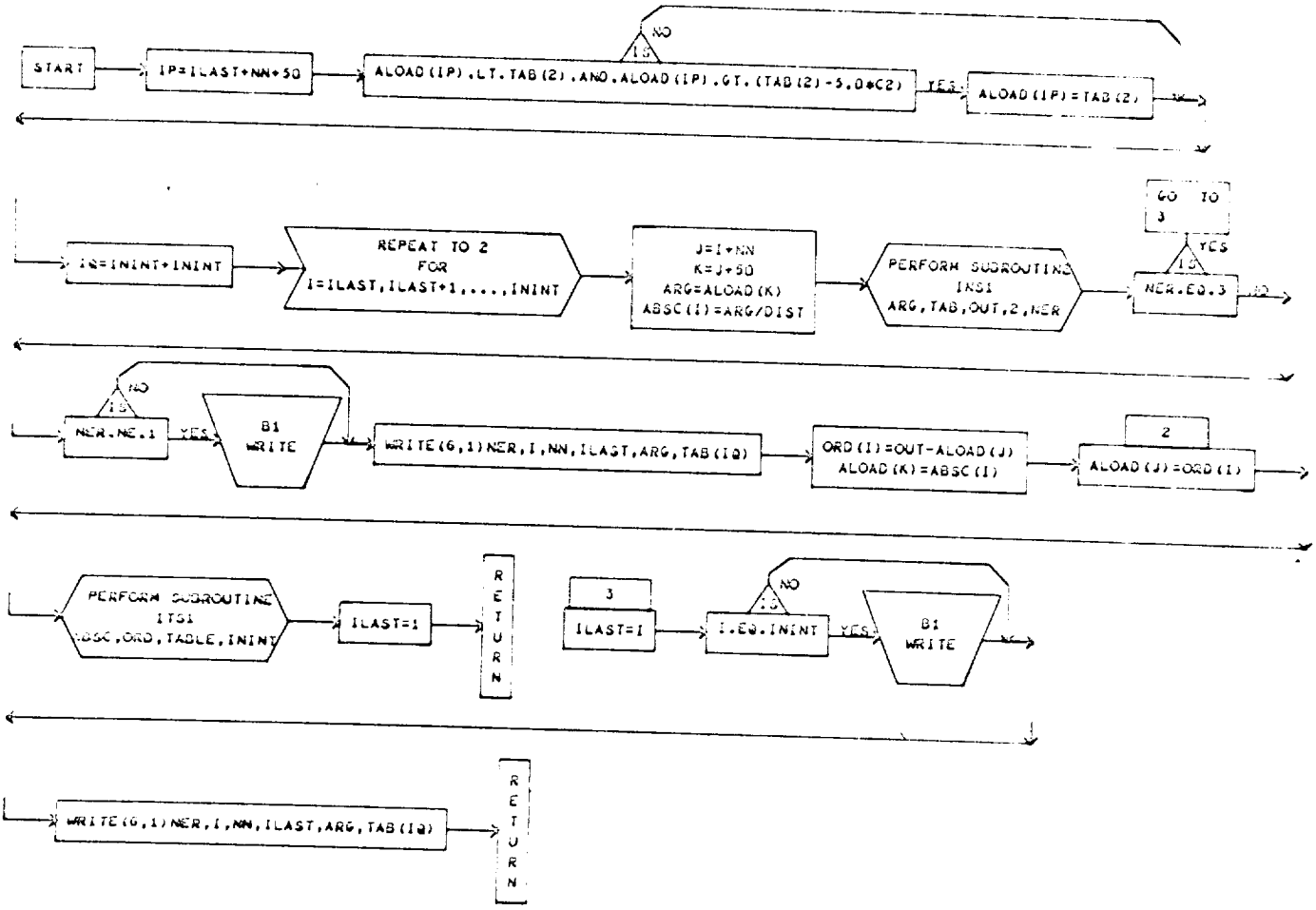
Subroutine INTERP (NN, DIST) (Deck 55VZ). The interpolation for loading is set up by this subroutine which calls subroutine INS1 to actually perform the interpolation. The interpolation loop is sometimes halted before all of the abscissae in the dependent interpolation array have been used, since the range of the independent interpolation abscissa array is not always adequate. This is normal operation for subroutine INTERP, which stores as ILAST the abscissa element number for which the range is first exceeded; the interpolation may then be continued with this element when new independent interpolation data have been loaded into the TAB array. The results of the interpolation are stored in both the ALØAD array and the numerical integration arrays ABSC and ORD. The abscissae are nondimensionalized with respect to DIST.

If interpolation is completed satisfactorily for the last element of the dependent abscissa array, the loading is numerically integrated by subroutine ITS1 so that loading, too, may be nondimensionalized in subroutine ALOAD.

```
          SUBROUTINE INTERP (NN , DIST)
C                                                                       55VZ0010
C         THIS SUBROUTINE PERFORMS THE ACTUAL INTERPOLATION (USING THE  55VZ0020
C         STANDARD DOUGLAS INTERPOLATION ROUTINE) TO DETERMINE THE DESIRED 55VZ0030
C         LOAD DISTRIBUTION                                             55VZ0040
C                                                                       55VZ0050
000004    COMMON DATA(17), ABSC(50), ORD(50), TABLE(50), ALOAD(1000), AA(16)55VZ0060
        A , TAB(101), GP, GPM2, GPP2, CG, OMGP, CONR, CONI, ILAST, C1, C2 55VZ0070
        B , C3, R1                                                      55VZ0080
C                                                                       55VZ0090
000004    EQUIVALENCE                (DATA( 1),      A) , (DATA( 2),      B) 55VZ0100
        A , (DATA( 3),      C) , (DATA( 4),      D) , (DATA( 5),      E) 55VZ0110
        B , (DATA( 6),      F) , (DATA( 7),      G) , (DATA( 8),      H) 55VZ0120
        C , (DATA( 9),     AI) , (DATA(10),   MODE) , (DATA(11),  NTYPE) 55VZ0130
        D , (DATA(12),  ININT) , (DATA(13), ETAIRQ) , (DATA(14), ETAORQ) 55VZ0140
        E , (DATA(15),  PIREQ) , (DATA(16),  POREQ) , (DATA(17),  GAMMA) 55VZ0150
C                                                                       55VZ0160
000004    IP = ILAST + NN + 50                                          55VZ0170
C                                                                       55VZ0180
C         TEST FOR INITIAL INTERPOLATION VARIABLE TO SEE IF IT IS SLIGHTLY 55VZ0190
C         LESS THAN THE SMALLEST INDEPENDENT VARIABLE IN THE INTERPOLATION 55VZ0200
C         ARRAY.  IF SO, SET THEM EQUAL.                                55VZ0210
C                                                                       55VZ0220
000006    IF (ALOAD(IP) .LT. TAB(2) .AND. ALOAD(IP) .GT.(TAB(2) - 5.0*C2)) 55VZ0230
        A     ALOAD(IP) = TAB(2)                                        55VZ0240
000025    IQ = ININT + ININT                                            55VZ0250
000025    DO 2 I=ILAST,ININT                                            55VZ0260
000027    J = I + NN                                                    55VZ0270
000030    K = J + 50                                                    55VZ0280
000032    ARG = ALOAD(K)                                                55VZ0290
000034    ABSC(I) = ARG / DIST                                          55VZ0300
000037    CALL INS1 (ARG, TAB, OUT, 2, NER)                             55VZ0310
000042    IF (NER .EQ. 3) GO TO 3                                       55VZ0320
000045    IF (NER .NE. 1) WRITE (6,1) NER,I,NN,ILAST,ARG,TAB(IQ)        55VZ0330
000072  1 FORMAT (1H , 69H**SUBROUTINE INTERP   NER .NE. 1   NER, I, NN, ILA55VZ0340
       AST, ARG, TAB(IQ) =     , 4I5, 2E18.9)                           55VZ0350
000072    ORD(I) = OUT - ALOAD(J)                                       55VZ0360
000077    ALOAD(K) = ABSC(I)                                            55VZ0370
000102  2 ALOAD(J) = ORD(I)                                             55VZ0380
000107    CALL ITS1 (ABSC, ORD, TABLE, ININT)                           55VZ0390
000112    ILAST = 1                                                     55VZ0400
000113    RETURN                                                        55VZ0410
000114  3 ILAST = 1                                                     55VZ0420
000116    IF (I .EQ. ININT) WRITE( 6,1) NER,I,NN,ILAST,ARG,TAB(IQ)      55VZ0430
000144    RETURN                                                        55VZ0440
000145    END                                                           55VZ0450
                                                                        55VZ0460
```

```
SUBPROGRAM LENGTH
000212

FUNCTION ASSIGNMENTS

STATEMENT ASSIGNMENTS
1       -  000153    3       -  000114

BLOCK NAMES AND LENGTHS
        -  002420

VARIABLE ASSIGNMENTS
AA      -  002217C01 ABSC    -  000021C01 ALOAD   -  000247C01 ARG     -  000207
C2      -  002415C01 DATA    -  000000C01 I       -  000204    ILAST   -  002413C01
ININT   -  000013C01 IP      -  000202    IQ      -  000203    J       -  000209
K       -  000206    NER     -  000211    ORD     -  000103C01 OUT     -  000210
TAB     -  002237C01 TABLE   -  000165C01

START OF CONSTANTS
000147

START OF TEMPORARIES
000166

START OF INDIRECTS
000174

UNUSED COMPILER SPACE
043000
```

SUBROUTINE INTERP (NN , DIST)



SYMBOLS USED IN SUBROUTINE INTERP (DECK IDENTIFICATION 55VZ)

A	R	C	POTENTIAL AT A (MAPPING CONSTANT)	INTERP
AA	R	C	ARRAY FOR RESULTS OF GEOMETRY AND DRAG INTEGRALS	INTERP
ABSC	R	C	ABSCISSA ARRAY FOR NUMERICAL INTEGRALS	INTERP
AI	R	C	POTENTIAL AT I (MAPPING CONSTANT)	INTERP
ALOAD	R	C	STORAGE ARRAY FOR POTENTIAL IN THE REAL (PHYSICAL) PLANE	INTERP
ARG	R	U	ARGUMENT IN INTERPOLATION ROUTINE (INDEPENDENT VARIABLE)	INTERP
B	R	C	POTENTIAL AT B (MAPPING CONSTANT)	INTERP
C	R	C	POTENTIAL AT C (MAPPING CONSTANT)	INTERP
CG	R	C	COSINE OF DIHEDRAL ANGLE	INTERP
CONI	R	C	FACTOR FOR IMAJINARY PART OF COMPLEX DISTANCE	INTERP
CONR	R	C	FACTOR FOR REAL PART OF COMPLEX DISTANCE	INTERP
C1	R	C	CONSTANT THAT IS COMPUTER HARDWARE DEPENDENT	INTERP
C2	R	C	CONSTANT THAT IS COMPUTER HARDWARE DEPENDENT	INTERP
C3	R	C	CONSTANT THAT IS COMPUTER HARDWARE DEPENDENT	INTERP
D	R	C	POTENTIAL AT D (MAPPING CONSTANT)	INTERP
DATA	R	C	ARRAY FOR INPUT DATA	INTERP
DIST	R	U	NONDIMENSION DISTANCE IN PHYSICAL OR REAL PLANE	INTERP
E	R	C	POTENTIAL AT E (MAPPING CONSTANT)	INTERP
ETAIRQ	R	C	INBOARD SPANWISE LOCATION (NONDIMENSIONAL)	INTERP
ETAORQ	R	C	OUTBOARD SPANWISE LOCATION (NONDIMENSIONAL)	INTERP
F	R	C	POTENTIAL AT F (MAPPING CONSTANT)	INTERP
G	R	C	POTENTIAL AT G (MAPPING CONSTANT)	INTERP
GAMMA	R	C	DIHEDRAL ANGLE IN DEGREES	INTERP
GP	R	C	DIHEDRAL ANGLE DIVIDED BY PI	INTERP
GPM2	R	C	GP - 0.5	INTERP
GPP2	R	C	GP + 0.5	INTERP
H	R	C	POTENTIAL AT H (MAPPING CONSTANT)	INTERP
I	I	U	DO LOOP INDEX	INTERP
ILAST	I	C	FLAG FOR LAST ITERATION	INTERP
ININT	I	C	NUMBER OF ORDINATES IN NUMERICAL INTEGRALS	INTERP
IP	I	U	CALCULATION VARIABLE	INTERP
IQ	I	U	CALCULATION VARIABLE	INTERP
J	I	U	CALCULATION VARIABLE	INTERP
K	I	U	CALCULATION VARIABLE	INTERP
MODE	I	C	PROGRAM MODE OF OPERATION	INTERP
NER	I	U	ERROR FLAG FOR INTERPOLATION ROUTINE	INTERP
NN	I	U	LOCATION IN ALOAD ARRAY	INTERP
NTYPE	I	C	CONFIGURATION TYPE	INTERP

SYMBOLS USED IN SUBROUTINE INTERP (DECK IDENTIFICATION 55V2)

CMGP	R	C	1.0 - GP	
CRD	R	C	ORDINATE ARRAY FOR NUMERICAL INTEGRALS	INTERP
CUT	R	U	OUTPUT OF INTERPOLATION ROUTINE	INTERP
PIREQ	R	C	INBOARD LENGTH OR DISTANCE (NONDIMENSIONAL)	INTERP
POREQ	R	C	OUTBOARD LENGTH OR DISTANCE (NONDIMENSIONAL)	INTERP
R1	R	C	SQUARE ROOT OF SUM OF SQUARES OF ERRORS IN REQD CONDITIONS	INTERP
TAB	R	C	INTERPOLATION ARRAY FOR INDEPENDENT VARIABLE AND FUNCTION	INTERP
TABLE	R	C	INTEGRATED VALUES OF ORDINATES AS A FUNCTION OF ABSCISSA	INTERP

Subroutine INS1 (ARG, TABLE, OUTPUT, NLQ, NER) (Deck 55V1). G. E. Short of the Scientific Computer Programming Group completed this subroutine on August 13, 1963 for the Douglas Aircraft Company's subroutine library. Subroutine INS1 interpolates for f(x) in a table of x versus f(x) having nonuniform spacing in x with either linear or quadratic interplation.

```
      SUBROUTINE INS1(ARG,TABLE,OUTPUT,NLQ,NER     )                    55V10010
C                                                                       55V10020
C     THIS SUBROUTINE WAS WRITTEN BY G. E. SHORT ON 8/13/63 FOR THE     55V10030
C     DOUGLAS SET OF STANDARD SUBROUTINES.                              55V10040
C                                                                       55V10050
CINS1  INS1 IS AN INTERPOLATION ROUTINE  - ONE FUNCTION OF A SINGLE     55V10060
C     VARIABLE.I.E. Y = F(X) - LINEAR  OR QUADRATIC                     55V10070
C     ARGUMENTS OF THE SUBROUTINE ARE AS FOLLOWS                        55V10080
C     ARG     =  INPUT INTERPOLATION ARGUMENT (X)                       55V10090
C     TABLE   =  THE  SET  OF X VALUES FOLLOWED BY THE Y VALUES         55V10100
C                WHERE THE FIRST VALUE IS THE SIZE OF THE ARRAY (INTEGER)55V10110
C                HENCE, TABLE(1) = THE NUMBER OF TABLE ORDINATES        55V10120
C                       TABLE(2) = X(1)                                 55V10130
C                       TABLE(3) = FUNCTION(X(1))                       55V10140
C                       TABLE(4) = X(2)                                 55V10150
C                       TABLE(5) = FUNCTION(X(2)) , ETC.                55V10160
C     OUTPUT =  INTERPOLATED VALUE OF  Y = F(X)                         55V10170
C     NLQ    =  1   LINEAR INTERPOLATION                                55V10180
C            =  2   QUADRATIC INTERPOLATION                             55V10190
C     NER    =  ERROR CODE                                              55V10200
C               1 = INTERPOLATION SUCCESS FUL                           55V10210
C               2 = OFF CHART LOW END, MIN VALUE SUBSTITUTED            55V10220
C               3 = OFF CHART HIGH END, MAX VALUE SUBSTITUTED           55V10230
C               4 = NOT ENOUGH ENTRIES  TO INTERPOLATE                  55V10240
C               5 = X ENTRIES  NOT IN  ASCENDING ORDER                  55V10250
C     METHOD USED IS  LAGRANGE INTERPOLATION                            55V10260
C
000006      DIMENSION TABLE(1001)                                       55V10270
000006      EQUIVALENCE  ( NOENTR, A)                                   55V10280
000006      A  = TABLE  ( 1 )                                           55V10290
000007      J = 3                                                       55V10300
000010      IF(NLQ .NE. 1 ) GO TO 2                                     55V10310
000012      IF( NOENTR . GE. 2) GO TO 4                                 55V10320
000014    1 NER = 4                                                     55V10330
000015      GO TO 75                                                    55V10340
000016    2 IF( NOENTR .LT. 3) GO TO 1                                  55V10350
000021    4 IF( ARG - TABLE(2) ) 5,7,8                                  55V10360
000024    5 NER = 2                                                     55V10370
000025      GO TO 70                                                    55V10380
000026    7 NER = 1                                                     55V10390
000027      GO TO 70                                                    55V10400
000030    8 NOS  = 2* NOENTR                                            55V10410
000032      DO 20   I  = 4,NOS,2                                        55V10420
000033      J  =  I + 1                                                 55V10430
000035      IF(TABLE(I) -TABLE(I-2))15,15,17                            55V10440
000040   15 NER =  5                                                    55V10450
000041      GO TO  75                                                   55V10460
000042   17 IF(TABLE(I) - ARG)20,7 ,34                                  55V10470
000045   20 CONTINUE                                                    55V10480
000050      NER  = 3                                                    55V10490
000051      GO TO 70                                                    55V10500
000051   34 IF(NLQ-1)35,35,40                                           55V10510
C                                                                       55V10520
C     USE LINEAR INTERPOLATION                                          55V10530
C                                                                       55V10540
000053   35 OUTPUT =  TABLE(I+1)*(ARG - TABLE(I-2))/(TABLE(I) - TABLE(I-2)) + 55V10550
         1 TABLE(I-1) *(ARG - TABLE(I)) /(TABLE (I-2) - TABLE(I))       55V10560
                                                                        55V10570
```

```
000071          NER = 1                                                 55V10580
000072          GO TO 75                                                55V10590
          C                                                             55V10600
          C     USE  QUADRATIC INTERPOLATION                            55V10610
          C                                                             55V10620
000073       40 I = I-2                                                 55V10630
000075          IF(I-4)41,42,42                                         55V10640
000077       41 I = I + 2                                               55V10650
000101       42 XA1 = ARG - TABLE(I)                                    55V10660
000104          XA0 = ARG - TABLE(I-2)                                  55V10670
000106          XA2 = ARG - TABLE(I+2)                                  55V10680
000110          X01 = TABLE(I-2) - TABLE(I)                             55V10690
000113          X02 = TABLE(I-2) - TABLE(I+2)                           55V10700
000116          X12 = TABLE(I)   - TABLE(I+2)                           55V10710
000121          OUTPUT = TABLE(I-1)*XA1/X01* XA2/X02 - TABLE(I+1)*XA0/X01*XA2/X12 55V10720
               1+ TABLE(I+3)*XA0/X02 * XA1 /X12                         55V10730
000135          NER = 1                                                 55V10740
000135          GO TO 75                                                55V10750
000136       70 OUTPUT  = TABLE(J)                                      55V10760
000140       75 RETURN                                                  55V10770
000141          END                                                     55V10780
```

SUBPROGRAM LENGTH
000204

FUNCTION ASSIGNMENTS

STATEMENT ASSIGNMENTS

1	-	000014	2	-	000016	4	-	000021	5	-	000024
7	-	000026	8	-	000030	15	-	000040	17	-	000042
20	-	000045	34	-	000051	35	-	000053	40	-	000073
4	-	000077	42	-	000101	70	-	000136	75	-	000140

BLOCK NAMES AND LENGTHS

VARIABLE ASSIGNMENTS

A	-	000172	I	-	000175	J	-	000173	NOFNTR	-	000172
NOS	-	000174	XA0	-	000177	XA1	-	000176	XA2	-	000200
X01	-	000201	X02	-	000202	X12	-	000203			

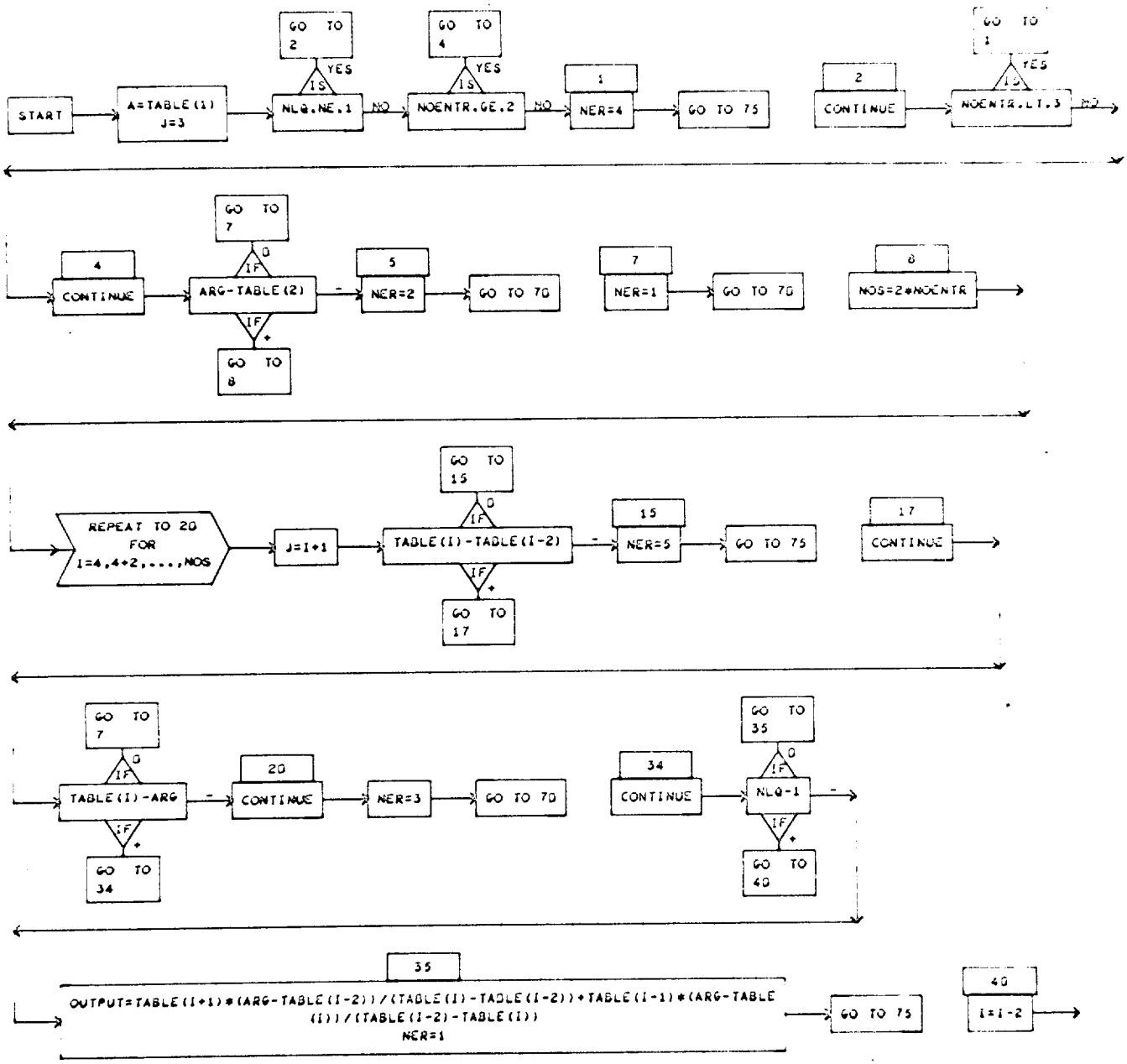
START OF CONSTANTS
000143

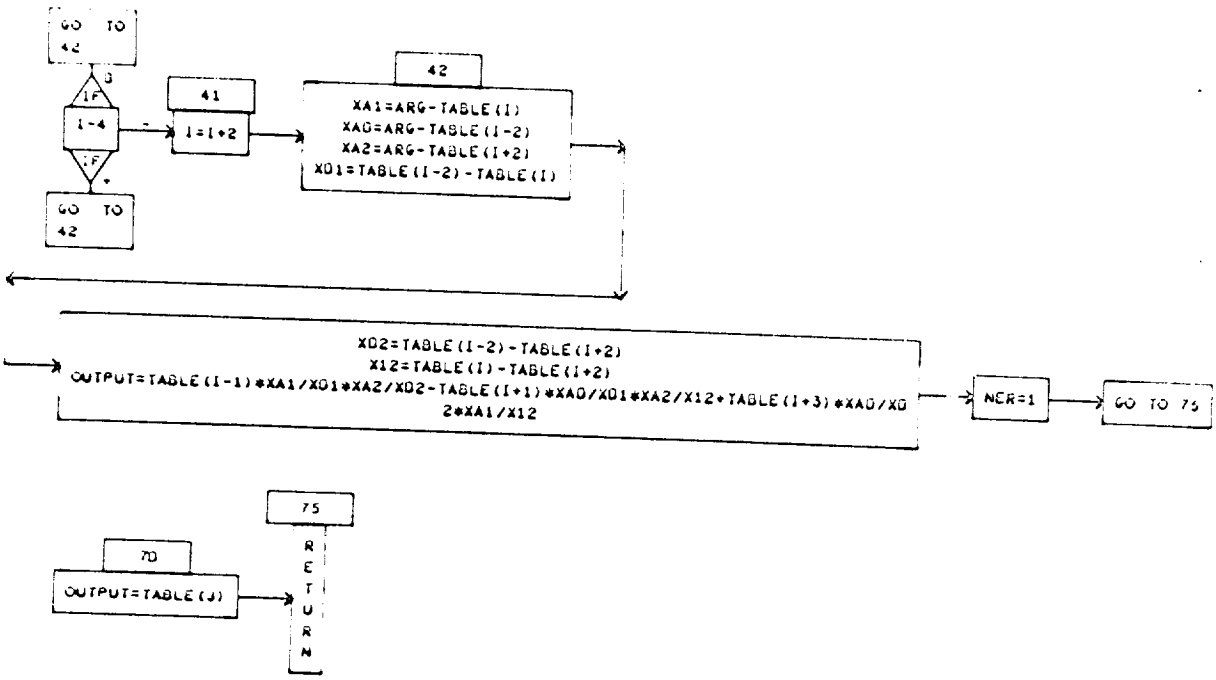
START OF TEMPORARIES
000144

START OF INDIRECTS
000156

UNUSED COMPILER SPACE
043000

SUBROUTINE INS1(ARG,TABLE,OUTPUT,NLQ,NER




GO TO 42
IF
I-4
IF
GO TO 42
41
I=I+2
42
XA1=ARG-TABLE(I)
XA0=ARG-TABLE(I-2)
XA2=ARG-TABLE(I+2)
XD1=TABLE(I-2)-TABLE(I)
XD2=TABLE(I-2)-TABLE(I+2)
X12=TABLE(I)-TABLE(I+2)
OUTPUT=TABLE(I-1)*XA1/XD1*XA2/XD2-TABLE(I+1)*XA0/XD1*XA2/X12+TABLE(I+3)*XA0/XD
2*XA1/X12
NER=1
GO TO 75

70
OUTPUT=TABLE(J)
75
RETURN

SYMBOLS USED IN SUBROUTINE INS1 (DECK IDENTIFICATION 55V11)

A	R	U	REAL VALUE OF THE INTEGER NCENTR	INS1
ARG	R	U	INTERPOLATION ARGUMENT	INS1
I	I	U	DO LOOP INDEX	INS1
J	I	U	CALCULATION VARIABLE	INS1
NER	I	U	ERROR FLAG	INS1
NLQ	I	U	FLAG FOR LINEAR OR QUADRATIC INTERPOLATION	INS1
NCENTR	I	U	NUMBER OF ABSCISSAE IN TABLE	INS1
NOS	I	U	CALCULATION VARIABLE	INS1
OUTPUT	R	U	RESULT OF INTERPOLATION	INS1
TABLE	R	U	TABLE OF ABSCISSAE AND ORDINATES FOR INTERPOLATION	INS1
NLQ	I	U	FLAG FOR LINEAR OR QUADRATIC INTERPOLATION	INS1
NCENTR	I	U	NUMBER OF ABSCISSAE IN TABLE	INS1
NOS	I	U	CALCULATION VARIABLE	INS1
OUTPUT	R	U	RESULT OF INTERPOLATION	INS1
TABLE	R	U	TABLE OF ABSCISSAE AND ORDINATES FOR INTERPOLATION	INS1
XA0	R	U	CALCULATION VARIABLE	INS1
XA1	R	U	CALCULATION VARIABLE	INS1
XA2	R	U	CALCULATION VARIABLE	INS1
X01	R	U	CALCULATION VARIABLE	INS1
X02	R	U	CALCULATION VARIABLE	INS1
X12	R	U	CALCULATION VARIABLE	INS1

Function ILOAD (X) (Deck 55V2). This function routine computes the ratio of vorticity at nondimensional span station X to the average vorticity for an elliptical span loading. If X does not lie in the interval

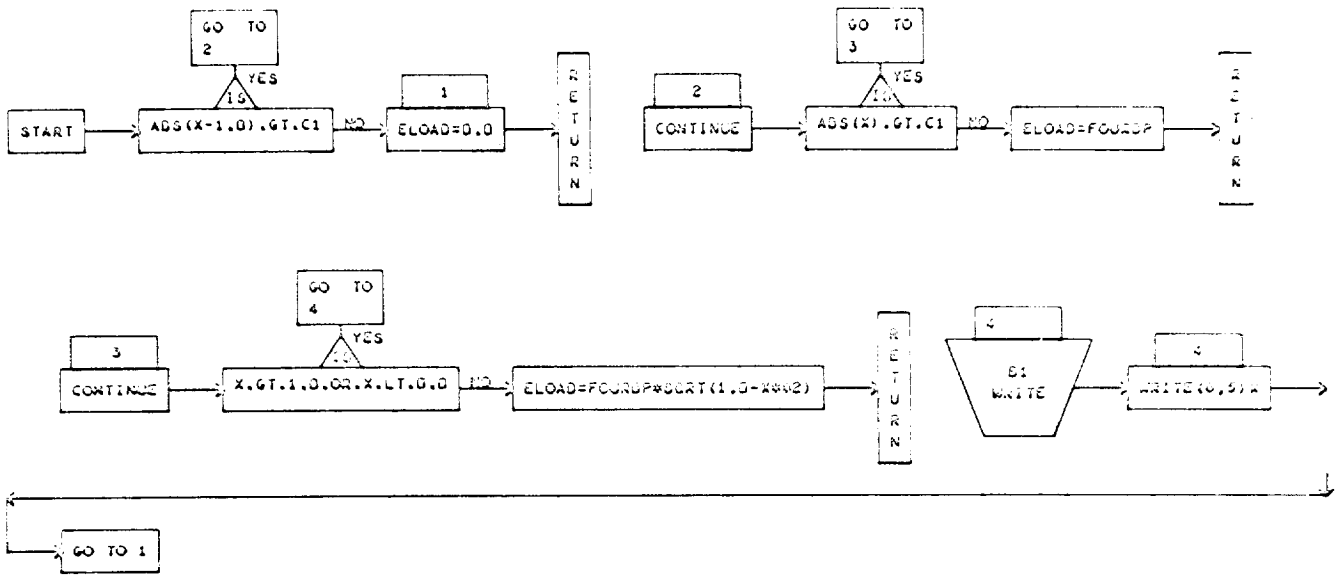
$$- C1 \leq X \leq 1 + C1 ,$$

an error message is printed. The parameter C1 is computing machine hardware dependent, and is roughly equal or somewhat larger than the smallest number that has significance relative to 1.

```
                FUNCTION ELOAD(X)                                       55V20010
          C                                                             55V20020
          C     THIS FUNCTION COMPUTES THE RATIO OF VORTICITY AT STATION X TO THE 55V20030
          C     AVERAGE VORTICITY FOR AN ELLIPTICAL SPAN LOADING.  THE SPANWISE   55V20040
          C     COORDINATE X MUST BE NONDIMENSIONAL RELATIVE TO SEMISPAN.         55V20050
          C                                                             55V20060
000003          COMMON DATA(17), ABSC(50), ORD(50), TABLE(50), ALOAD(1000), AA(16)55V20070
               A , TAB(101), GP, GPM2, GPP2, CG, OMGP, CONR, CONI, ILAST, C2, C1  55V20080
               B , C3, R1                                               55V20090
          C                                                             55V20100
000003          DATA  FOURBP / 1.2732395 /                              55V20110
          C                                                             55V20120
000003          IF (ABS(X - 1.0) .GT. C1) GO TO 2                       55V20130
000010        1 ELOAD = 0.0                                             55V20140
000011          RETURN                                                  55V20150
000012        2 IF (ABS(X) .GT. C1) GO TO 3                             55V20160
000016          ELOAD = FOURBP                                          55V20170
000017          RETURN                                                  55V20180
000017        3 IF (X .GT. 1.0 .OR. X .LT. 0.0) GO TO 4                 55V20190
000026          ELOAD = FOURBP * SQRT(1.0 - X**2)                       55V20200
000034          RETURN                                                  55V20210
000034        4 WRITE (6,5) X                                           55V20220
000042        5 FORMAT (1H , 26H SUBROUTINE ELOAD(X), X =   , E18.8)    55V20230
000042          GO TO 1                                                 55V20240
000044          END                                                     55V20250
```

```
SUBPROGRAM LENGTH
000071

FUNCTION ASSIGNMENTS

STATEMENT ASSIGNMENTS
1       -  000010    2       -  000012    3       -  000017    4       -  000034
5       -  000051

BLOCK NAMES AND LENGTHS
        -  002420

VARIABLE ASSIGNMENTS
AA      -  002217C01 ARSC    -  000021C01 ALOAD   -  000247C01 C1      -  002415C01
DATA    -  000000C01 ELOAD   -  000067    FOURBP  -  000070    ORD     -  000103C01
TAB     -  002237C01 TABLE   -  000165C01

START OF CONSTANTS
000046

START OF TEMPORARIES
000057

START OF INDIRECTS
000067

UNUSED COMPILER SPACE
043300
```

FUNCTION ELOAD(X)



SYMBOLS USED IN SUBROUTINE ELOAD (DECK IDENTIFICATION 55V2)

A	R	C	POTENTIAL AT A (MAPPING CONSTANT)	ELOAD
AA	R	C	ARRAY FOR RESULTS OF GEOMETRY AND DRAG INTEGRALS	ELOAD
ABSC	R	C	ABSCISSA ARRAY FOR NUMERICAL INTEGRALS	ELOAD
AI	R	C	POTENTIAL AT I (MAPPING CONSTANT)	ELOAD
ALOAD	R	C	STORAGE ARRAY FOR POTENTIAL IN THE REAL (PHYSICAL) PLANE	ELOAD
B	R	C	POTENTIAL AT B (MAPPING CONSTANT)	ELOAD
C	R	C	POTENTIAL AT C (MAPPING CONSTANT)	ELOAD
CG	R	C	COSINE OF DIHEDRAL ANGLE	ELOAD
CONI	R	C	FACTOR FOR IMAJINARY PART OF COMPLEX DISTANCE	ELOAD
CONR	R	C	FACTOR FOR REAL PART OF COMPLEX DISTANCE	ELOAD
C1	R	C	CONSTANT THAT IS COMPUTER HARDWARE DEPENDENT	ELOAD
C2	R	C	CONSTANT THAT IS COMPUTER HARDWARE DEPENDENT	ELOAD
C3	R	C	CONSTANT THAT IS COMPUTER HARDWARE DEPENDENT	ELOAD
D	R	C	POTENTIAL AT D (MAPPING CONSTANT)	ELOAD
DATA	R	C	ARRAY FO INPUT DATA	ELOAD
E	R	C	POTENTIAL AT E (MAPPING CONSTANT)	ELOAD
ETAIRD	R	C	INBOARD SPANWISE LOCATION (NONDIMENSIONAL)	ELOAD
ETAORD	R	C	OUTBOARD SPANWISE LOCATION (NONDIMENSIONAL)	ELOAD
F	R	C	POTENTIAL AT F (MAPPING CONSTANT)	ELOAD
FOURBP	R	U	4.0 DIVIDED BY PI	ELOAD
G	R	C	POTENTIAL AT G (MAPPING CONSTANT)	ELOAD
GAMMA	R	C	DIHEDRAL ANGLE IN DEGREES	ELOAD
GP	R	C	DIHEDRAL ANGLE DIVIDED BY PI	ELOAD
GPM2	R	C	GP - 0.5	ELOAD
GPP2	R	C	GP + 0.5	ELOAD
H	R	C	POTENTIAL AT H (MAPPING CONSTANT)	ELOAD
ILAST	I	C	FLAG FOR LAST ITERATION	ELOAD
ININT	I	C	NUMBER OF ORDINATES IN NUMERICAL INTEGRALS	ELOAD
MODE	I	C	PROGRAM MODE OF OPERATION	ELOAD
NTYPE	I	C	CONFIGURATION TYPE	ELOAD
OMGP	R	C	1.0 - GP	ELOAD
ORD	R	C	ORDINATE ARRAY FOR NUMERICAL INTEGRALS	ELOAD
PIREQ	R	C	INBOARD LENGTH OR DISTANCE (NONDIMENSIONAL)	ELOAD
POREQ	R	C	OUTBOARD LENGTH OR DISTANCE (NONDIMENSIONAL)	ELOAD
R1	R	C	SQUARE ROOT OF SUM OF SQUARES OF ERRORS IN REQD CONDITIONS	ELOAD
TAB	R	C	INTERPOLATION ARRAY FOR INDEPENDENT VARIABLE AND FUNCTION	ELOAD
TABLE	R	C	INTEGRATED VALUES OF ORDINATES AS A FUNCTION OF ABSCISSA	ELOAD
X	R	U	NONDIMENSIONAL SPAN STATION	ELOAD

```
CORE MAP  13.56.14.  NORMAL                    CONTROL
        ---TIME---LOAD MODE --L1--L2-------TYPE-------------------USER---++---CALL-------------------FWA LOAD--LWA LOAD --BLNK COMM--LENGTH--
        FWA LOADER  074507  FWA TABLES  067256
        -PROGRAM----ADDRESS-                                  --LABELED---COMMON--
        MAIN        000100
        TEST        005212
        INPUT       005311
        SERIES      005466
        TRANS       005520
        ITS1        005643
        DATACK      006374
        HEADER      007236
        INNER       010134
        GEOM        013633
        RSSERR      020062
        NUMINT      020123
        INVERT      020640
        MIS1        026755
        COMPAT      027226
        STORE       027301
        POTOUT      027640
        DRAGI       027776
        NUM         033363
        LOAD        037762
        INTEST      043047
        TABLOD      043120
        INTERP      043161
        INS1        043373
        ELOAD       043577
        ARC1        043670
        ATM1        044305
        ATM2        044443
        ATM3        044604
        ATM4        045067
        COR1        045247
        COR2        045420
        COR3        045647
        DRK1        046007
        ELP1        046177
        INS1        046505
        ITS2        046711
        ITS3        046761
        ITS4        047074
        LSP1        047200
        LSP2        047352
        LSP3        047550
        MIS1        047730
        MIS2        050201
        MIS3        050623
        ONEVAR      051173
        SRT1        051560
        ZZL1        051662
        SYSTEM      051715
        OUTPTC      053033                                  SCOPE2      051715
        ACCOER      054365
        SQRT        054377
        OVERFL      054442
        DVCHK       054447
        INPUTC      054455
        DBAIEX      055554
        SINCOS      055632
        RBAREX      055731
```

DABS	056010
DSQRT	056025
EXP	056105
ALNLOG	056164
DBADEX	056253
DLNLOG	056362
LOCF	056560
SIGS	056563
GETBA	057412
DEXP	057431

--ENTRY------ADDRESS-

MAIN	000101
TEST	005213
INPUT	005312
SERIES	005467
TRANS	005521
ITS1	005644
DATACK	006375
HEADER	007237
INNER	010135
GEOM	013634
RSSCAR	020063
NUMINT	020125
INVERT	026641

REFERENCES

MAIN	000117	000130					
SERIES	005472	005476	005502	005505			
MAIN	000131						
MAIN	000140	000211	000313	000353	000455	000541	000642
	000750						
SERIES	005470						
DATACK	006651						
INNER	010330	010474	010701	010735	011100	011304	011426
	011652	012024	012241	012267	012417	012631	012664
	013020	013407					
NUMINT	024404						
NUM	036304						
INTERP	043272						
MAIN	000133						
SERIES	005473						
SERIES	005474						
INNER	010155	010310	010353	010363	010373	010477	010665
	010672	010703	010720	010730	010737	010756	011103
	011205	011275	011306	011321	011431	011617	011624
	011633	011643	011654	011671	011676	012052	012234
	012243	012270	012271	012306	012440	012624	012633
	012650	012655	012666	012701	013023	013224	013233
	013243	013263	013263	013270			
INNER	010157	010561	011105	011433	012654	012442	013025
GEOM	013730	013753	013761	013764	014014	014017	014046
	014051	014100	014103	014132	014135	014273	014276
	[illegible]	[illegible]	[illegible]	014342	014410	014413	014441
	014444	014456	014461	014517	014522	014551	014554
	014603	014606	014635	014640	014777	015002	015030
	015033	015061	015064	015076	015101	015137	015142
	015171	015174	015330	015333	015345	015350	015414
	015417	015445	015450	015474	015477	015522	015525
	015553	015556	015605	015610	015642	015645	015657
	015662	016060	016063	016075	016100	016143	016146
	016172	016175	016224	016227	016260	016263	016317
	016322	016334	016337	016511	016514	016542	016545
	016573	016576	016622	016625	016657	016662	016710
	016713	016743	016746	016775	017000	017032	017035
	017063	017066	017224	017227	017255	017260	017313
	017316	017344	017347	017400	017403	017432	017435
	017450	017453	017502	017505	017533	017536	
INNER	010462	011060	011414	012004	012403	013002	013371

MIS1	026757	INVERT	026707	026712					
		LSP1	047315	047320					
		LSP3	047672	047675					
COMPAT	027227	INNER	010473	011077	011425	012023	012416	013017	013406
STORE	027302	GEOM	013740	013771	014024	014056	014110	014142	014303
			014320	014367	014420	014451	014466	014527	014561
			014613	014645	015007	015040	015071	015106	015147
			015201	015340	015355	015424	015455	015504	015532
			015563	015615	015652	015667	016070	016105	016153
			016202	016234	016270	016327	016344	016521	016552
			016603	016632	016667	016720	016753	017005	017042
			017073	017234	017265	017323	017354	017410	017442
			017460	017512	017543				
POTOUT	027641	SERIES	005477						
DRAGI	027777	SERIES	005500						
NUM	033365	DRAGI	030066	030071	030111	030114	030134	030137	030257
			030262	030266	030271	030327	030332	030347	030352
			030372	030375	030474	030477	030515	030520	030540
			030543	030663	030666	030672	030675	030733	030736
			030756	030761	030765	030770	031052	031055	031073
			031076	031117	031122	031146	031151	031155	031160
			031311	031314	031332	031335	031355	031360	031401
			031404	031430	031433	031453	031456	031574	031577
			031617	031622	031642	031645	031665	031670	
LOAD	037763	SERIES	005503						
INTEST	043050	LOAD	040031	040054	040252	040307	040320	040536	040547
			040731	040742	040753	041161	041172	041206	041406
			041457	041473	041715	041733			
TABLOD	043121	LOAD	040033	040056	040254	040311	040322	040540	040551
			040670	040676	040711	040725	040733	040744	040755
			041140	041146	041163	041174	041210	041402	041412
			041425	041440	041450	041461	041675	041717	041735
INTERP	043162	LOAD	040035	040042	040047	040050	040256	040263	040270
			040275	040302	040313	040324	040542	040553	040672
			040700	040705	040713	040722	040727	040735	040746
			040757	041142	041150	041155	041165	041176	041212
			041404	041414	041421	041427	041434	041442	041452
			041463	041477	041721	041726	041737	04174[illegible]	042005
			042013						
INS1	043374	INTERP	043222						
ELOAD	043600	LOAD	040103	040132	040154	040355	040610	040432	040454
			040476	040576	041010	041043	041065	041240	041267
			041313	041525	041554	041576	041620	041644	042066
			042125						
APC1	043671								
ATM1	044306								
ATM2	044445								
ATM3	044606								
ATM4	045072								
COR1	045250								

COR2	045421
COR3	045650
DRK1	046012
ELP1	046200
INS1	046506
ITS2	046712
ITS3	046763
ITS4	047075
LSP1	047202
LSP2	047354
LSP3	047553
MIS1	047732
MIS2	050203
MIS3	050625
ONEVAR	051174
SRT1	051561
ZZL1	051663
QBNTRY	051716
SYSTEM	052257
SYSTEMC	052224
SYSTEMP	052252
END	052150

INTERP	043222	
INVERT	026707	026712
LSP1	047315	047320
LSP3	047672	047675
LSP2	047512	047515
COR1	045255	
COR2	045426	
MAIN	000102	
OUTPTC	053047	054176
ACGOER	054370	
SQRT	054324	
INPUTC	054514	055365
DBAIEX	055613	
SINCOS	055675	055700
RBAREX	055770	
DABS	056017	
DSQRT	056066	
EXP	056140	
ALNLOG	056224	056226
DBADEX	056335	
DLNLOG	056457	056461

MAIN	000757
TEST	005260
INPUT	005401
SERIES	005510
TRANS	005625
ITS1	006252
DATACK	007016
HEADER	007317
INNER	013412
GEOM	017624
RSSERR	020110
NUMINT	024426
INVERT	026725
COMPAT	027267
STORE	027571
POTOUT	027736
DRAGI	031732
NUM	036316
LOAD	042154
TABLOD	043144
INTERP	043327
INS1	043535
ELOAD	043644
ARC1	044200

		ATM1	044405						
		ATM2	044547						
		ATM3	045012						
		ATM4	045203						
		COR1	045361						
		COR2	045574						
		COR3	045750						
		DRK1	046151						
		ELP1	046364						
		INS1	046647						
		ITS2	046741						
		ITS3	047054						
		ITS4	047155						
		LSP1	047323						
		LSP2	047520						
		LSP3	047700						
		MIS1	050147						
		MIS2	050552						
		MIS3	051137						
		ONEVAR	051431						
		SRT1	051644						
		ZZL1	051705						
STOP	052177	INPUT	005377						
EXIT	052171								
ABNORML	052207	OUTPTC	053050	054177					
		ACGOER	054371						
		INPUTC	054515	055366					
OUTPTC	053035	MAIN	000123	000125	000126				
		TEST	005235	005236	005244	005245	005251	005254	005255
		INPUT	005317	005320	005374	005375			
		DETACK	006401	006403	006405	006407	006410	006424	006425
			006430	006431	006455	006457	006460	006474	006475
			006505	006507	006510	006546	006550	006552	006553
			006563	006564	006567	006571	006573	006575	006577
			006601	006603	006604	006617	006621	006622	006627
			006631	006632	006645	006647	006650	006654	006655
			006660	006662	006664	006666	006670	006672	006674
			006675	006740	006742	006743	006756	006756	006757
			006775	006777	007000	007011	007012		
		HEADER	007255	007256	007262	007263	007267	007270	007274
			007275	007301	007302	007306	007307	007313	007314
		INNER	010162	010166	010174	010200	010201	010504	010510
			010516	010522	010523	011110	011114	011122	011126
			011127	011436	011442	011450	011454	011455	012057
			012063	012071	012075	012076	012445	012451	012457
			012463	012464	013030	013034	013042	013046	013047
		GEOM	013642	013646	013651				
		NUMINT	022643	022645	022646				
		INVERT	026650	026651	026656	026667	026673	026701	026702
			026717	026720					
		STORE	027561	027563	027565	027566			
		POTOUT	027644	027645	027663	027667	027672	027676	027702
			027705	027711	027715	027720	027724	027730	027733
		DRAGI	030004	030010	030013	030164	030170	030174	030175
			030423	030427	030433	030434	030570	030574	030600
			030601	031034	031040	031044	031045	031225	031231
			031235	031237	031240	031510	031514	031520	031522
			031523	031716	031722	031726	031727		
		NUM	035351	035353	035354				

		LOAD	040070	040072	040073	040106	040110	040113	040115
			040117	040121	040122	040135	040137	040141	040143
			040144	040157	040161	040163	040165	040166	040336
			040340	040342	040343	040360	040362	040365	040367
			040371	040373	040375	040377	040400	040413	040415
			040417	040421	040422	040435	040437	040441	040443
			040444	040457	040461	040463	040465	040466	040501
			040503	040505	040507	040510	040563	040565	040566
			040601	040603	040606	040610	040612	040614	040615
			040771	040773	040775	040776	041013	041015	041020
			041022	041024	041026	041030	041032	041033	041046
			041050	041052	041054	041055	041070	041072	041074
			041076	041077	041223	041225	041227	041230	041243
			041245	041250	041252	041254	041256	041257	041272
			041274	041276	041300	041301	041316	041320	041322
			041324	041326	041327	041510	041512	041514	041515
			041530	041532	041535	041537	041541	041543	041544
			041557	041561	041563	041565	041566	041601	041603
			041605	041607	041610	041623	041625	041627	041631
			041632	041647	041651	041653	041655	041657	041660
			042045	042047	042051	042053	042054	042071	042073
			042076	042100	042102	042104	042106	042110	042111
			042130	042132	042134	042136	042140	042142	042144
			042146	042147					
		INTERP	043231	043233	043235	043237	043241	043243	043247
			043250	043302	043305	043307	043311	043313	043315
			043321	043322					
		ELOAD	043635	043637	043640				
		COR1	045260	045262	045264	045266	045267	045320	045322
			045326	045331	045337	045341	045345	045350	
		COR2	045431	045433	045435	045437	045440	045472	045474
			045500	045503	045511	045513	045517	045522	045532
			045534	045541	045544	045552	045554	045560	045563
		COR3	045654	045656	045657	045727	045731	045736	045742
		ONEVAR	051367	051371	051372	051403	051405	051406	051415
			051416	051425	051426				
KODER	053171								
ACGOER	054366								
		MAIN	000146						
		TRANS	005526	005550	005570				
		HEADER	007243						
		INNER	010144						
		GEOM	013660						
		NUMINT	022505						
		COMPAT	027234						
		POTOUT	027651						
		DRAGI	030021						
		NUM	035213						
		LOAD	037771						
		DRK1	046025						
		ITS3	047025						
		ITS4	047110	047135					
SQRT	054400								
		MAIN	000167	000230	000250	000335	000372	000413	000474
			000560	000575					
		GEOM	013751	014002	014035	014067	014121	014153	014331
			014345	014400	014427	014474	014505	014540	014572
			014624	014656	015020	015047	015114	015125	015160
			015212	015366	015402	015435	015464	015512	015543
			015574	015626	015700	015714	016116	016131	016161
			016371	016532	016561	016611	016643	016700	017104
			017245	017276	017334	017365	017421	017471	017523
			017554						
		RSSERR	020105						

		NUMINT	020144	020173	020215	020247	020300	020331	020361
			020403	020[illegible]	020453	020475	020517	020546	020600
			020634	020666	020720	020752	020777	021023	021045
			021070	021117	021154	021211	021246	021300	021332
			021364	021410	021435	021460	021526	021560	021577
			021644	021663	021716	021735	021754	021774	022015
			022654	022673	022712	022731	022750	022767	023006
			023025	023044	023062	023077	023116	023135	023154
			023173	023212	023231	023247	023264	023303	023322
			023341	023360	023377	023416	023434	023451	023470
			023507	023526	023545	023564	023603	023621	023636
			023737	023756	023774	024011	024030	024047	024146
			024163	024202	024221	024240	024257	024276	024314
			024333	024352					
		STORE	027370	027504	027530				
		DRAGI	030101	030124	030147	030301	030315	030337	030362
			030405	030505	030530	030553	030705	030721	030743
			031000	031014	031063	031204	031322	031345	031466
			031607	031632	031655	031700			
		NUM	033403	033425	033455	033504	033527	033551	033573
			033615	033637	033670	033723	033755	034007	034041
			034066	034110	034132	034161	034213	034245	034277
			034331	034363	034442	034475	034514	034535	034562
			034622	034641	034661	034702	035364	035403	035422
			035441	035460	035477	035516	035535	035554	035573
			035612	035631	035650	035667	035706	035725	035744
			036031	036050	036073	036156	036175	036214	036233
			036252						
		ELOAD	043627						
		ATM1	044353						
		ATM2	044504						
		ATM3	045003						
		ATM4	045163						
OVERFL	054443	TEST	005216						
DVCHK	054450	TEST	005220						
INPUTC	054457	INPUT	005323	005325	005331	005334	005343	005347	005353
			005360	005364	005370				
KRAKER	054561								
DBAIEX	055555	ITS1	005774	006000	006025	006031	006074	006100	
SIN	055633	INNER	012042	012435					
COS	055636	INNER	012032	012047	012430				
RBAREX	055732	GEOM	016213	016245	016301	016355	016731	016764	017016
			017053						
		NUMINT	020157	020230	020262	020313	020344	021422	021447
			021503	021516	021551	021621	021634	021705	023603
			023677	023714	023731	024074	024110	024124	024140
		STORE	027447	027476	027524				
		DRAGI	031106	031132	031170	031370	031414	031443	
		NUM	033440	033470	034376	034401	034417	034432	034465
			034540	034574	034612	035771	036006	036023	036120
			036134	036150					
		ATM1	044344	044350	044370	044377			
		ATM2	044501						
		ATM4	045133	045137	045151	045157			
DABS	056011	ARC1	043701						
DSQRT	056026	ARC1	044035						

EXP	056106	ATM2	044534			
		ATM3	044723	044756		
		RBAREX	055753			
ALOG	056170	ATM3	044702	044713	044735	044746
		RBAREX	055750			
ALOG10	056165					
DBADEX	056254	ELP1	046221			
DBAREX	056261					
RBADEX	056266					
DLOG	056363	ELP1	046246			
		DBADEX	056313			
DLOG10	056366					
LOCF	056561	ZZL1	051670	051675		
XLOCF	056561					
BKSPRU.	057061					
FIZBAK.	057072					
POSFIL.	057130	OUTPTC	053065			
RDPRU.	057140					
DAT.	057162	OUTPTC	053107	053123	053070	
		INPUTC	054533	054501	054524	
CIO1.	057026					
OPEN.	056565	SYSTEM	052530			
		OUTPTC	053055			
		INPUTC	054477			
SIO.	056700	OUTPTC	053122			
		INPUTC	054527			
GETBA	057412	OUTPTC	053043			
		INPUTC	054465			
DEXP	057432	DBADEX	056320			

----UNSATISFIED EXTERNALS----- REFERENCES

```
02/26/68  L A D C SCOPE 3.107 12/1/67
13.54.40.$ 3755 CARDS READ -$$READ -PP  187 SEC
13.54.40.$SEQUENCE,17020
13.54.40.$JOB,91386.
13.54.40.STAPE,SCR=00,OLD=00,NEW=00.
13.54.40.17020DN .PR=2600 TL=00500  FL=077700.
13.54.40.COMPILE,17,500,77700.
13.54.41.ATTACH.
13.54.41.NOW REMOTE JOB
13.54.41.RUN(S)
13.54.43.FILE OPENED ON DISK---- OUTPUT
13.54.43.FILE OPENED ON DISK---- LGO
13.55.25.MODE1.
13.55.26.COMMON(LIB)
13.55.26.REWIND(LIB)
13.55.28.COPYBF(LIB,X)
13.55.28.FILE OPENED ON DISK---- X
13.55.33.RETURN(LIB)
13.55.34.FILE RETURNED --  LIB
13.55.34.SET(0)
13.55.54. MEMORY IS ERROR FREE
13.55.55.LOAD(LGO)
13.56.03.LOAD(X)
13.56.06.EXECUTE./RFL/
13.56.27.$$CP  024.813 SEC. (077700=FL)
13.56.27.$$PP  028.008 SEC.
13.56.27.$$XX  133.380 SEC.
13.56.28.*-- FL USED DURING EXECUTION -(062300)
13.56.28.*-- FL REQUIRED FOR LOADING  -(072700)
13.59.34.FATAL ERROR 0084
13.59.35.$$CP  204.711 SEC. (062300=FL)
13.59.35.$$PP  040.000 SEC.
13.59.35.$$XX  146.799 SEC.
13.59.35. CONTROL POINT = 5
13.59.35.DISK FILE OUTPUT  - 002255/ 002255 PRUS
13.59.35.      DC = 0000, LC/S =000025, FETA =062300
13.59.35.DISK FILE INPUT   - 001102/ 001102 PRUS
13.59.35.      DC = 0000, LC/S =000021, FETA =001146
13.59.35.DISK FILE LGO     - 000536/ 000536 PRUS
13.59.35.      DC = 0000, LC/S =740031, FETA =000021
13.59.35.DISK FILE X       - 000173/ 000175 PRUS
13.59.35.      DC = 0000, LC/S =000021, FETA =000326
```

```
02/26/68  L A D C SCOPE 3.107 12/1/67
13.54.40.S 3755 CARDS READ -$$READ -pp  187 SEC
13.54.40.$SEQUENCE,17020
13.54.40.$JOB,91386.
13.54.40.$TAPE,SCR=00,OLD=00,NEW=00.
13.54.40.17020DN .PR=2600 TL=00500  FL=077700.
13.54.40.COMPILE,17,500,77700.
13.54.41.ATTACH.
13.54.41.NOW REMOTE JOB
13.54.41.RUN(S)
13.54.43.FILE OPENED ON DISK---- OUTPUT
13.54.43.FILE OPENED ON DISK---- LGO
13.55.25.MODE1.
13.55.26.COMMON(LIB)
13.55.26.REWIND(LIB)
13.55.28.COPYBF(LIB,X)
13.55.28.FILE OPENED ON DISK---- X
13.55.33.RETURN(LIB)
13.55.34.FILE RETURNED --  LIB
13.55.34.SET(0)
13.55.54. MEMORY IS ERROR FREE
13.55.55.LOAD(LGO)
13.56.03.LOAD(X)
13.56.06.EXECUTE./RFL/
13.56.27.$$CP  024.813 SEC. (077700=FL)
13.56.27.$$PP  028.008 SEC.
13.56.27.$$XX  133.380 SEC.
13.56.28.*-- FL USED DURING EXECUTION -(062300)
13.56.28.*-- FL REQUIRED FOR LOADING  -(072700)
13.59.34.FATAL ERROR 0084
13.59.35.$$CP  204.711 SEC. (062300=FL)
13.59.35.$$PP  040.000 SEC.
13.59.35.$$XX  146.799 SEC.
13.59.35. CONTROL POINT = 5
13.59.35.DISK FILE OUTPUT  - 002255/ 002255 PRUS
13.59.35.      DC = 0000, LC/S =000025, FETA =062300
13.59.35.DISK FILE INPUT   - 001102/ 001102 PRUS
13.59.35.      DC = 0000, LC/S =000021, FETA =001146
13.59.35.DISK FILE LGO     - 000536/ 000536 PRUS
13.59.35.      DC = 0000, LC/S =740031, FETA =000021
13.59.35.DISK FILE X       - 000173/ 000175 PRUS
13.59.35.      DC = 0000, LC/S =000021, FETA =00032/
```

Test Cases

To test all major computing modules of Computer Program 55VD, fourteen test cases are presented. The input data for these test cases are given in Table II. Reference 10 contains a detailed description of the data loading procedure for Computer Program 55VD. No special operating instructions are required for the test cases. Table III is an index for the answer listings of the fourteen test cases, and appears at the end of the text for this section.

Program Operation Mode 1. The first seven test cases are examples of the program operation in the mode that determines the mapping constants by electrostatic analog with iterative correction for Configurations 1-7. For Configurations 1-5, the initial values of the mapping constants have been measured with an analog field plotter. For Configurations 6 and 7, the initial values of the mapping constants are taken from solutions for Configurations 5 and 1, respectively, that are similar to the desired configurations. The two latter cases are examples of the idea that any method which produces values of the mapping constants with small errors is suitable for estimating initial values of the mapping constants for program operation in mode 1.

The answer listing for Configuration 1 is typical of a successfully completed case for Configurations 1-7. The first page lists the configuration number, the program operation mode, the number of ordinates used in each of the numerical integrals for geometry or minimum induced drag, and the unscaled (input) and scaled initial values of the Schwarz-Christoffel mapping constants.

The second page of answer listing for Configuration 1 first provides a summary for the iteration that determines the mapping constants accurately. The first line of numerical data gives the initial values of the mapping constants, the errors in the required geometry conditions, and the square root of the sum of the squares of the errors in the required geometry conditions. Below the first line of numerical output, the correction matrix $\partial C_i/\partial p_j$ is printed without a header. Below the correction matrix are six lines (the number varies from case to case) of numerical data identical in form with the first line on the page. Each of the first five lines summarizes the numerical results of one cycle of the iteration. By definition, convergence of the iteration occurs if $\bar{e} \leq 3\times10^{-8}$. In this example, five cycles of the iteration reduce $\bar{e}$ from approximately 0.037 to 1×10^{-8}. The sixth and last line of this iteration summary answer listing repeats the results of the line above that has the smallest value of $\bar{e}$; subsequent calculations requiring the mapping constants use the values in the last line. In this example, the last line is identical to the line immediately above it. Below the last line of the iteration summary answer listing, the final values of the mapping constants are printed in exponential format. The balance of the data on the second page of answer listing describes the geometry specified by the final values of the mapping constants, and also includes the results for the minimum induced drag integral and the minimum induced drag parameter k.

Table II

Input Data for Test Cases - Computer Program 55VD (Page 1)

```
0 1 1 15
173.9      196.5      216.8      242.5      497.9
823.0
0.5        0.0        0.125
0 1 2 15
260.5      268.5      277.0      287.8      301.4
319.9      343.9      527.7      285.1
0.361      0.625      0.1        0.1
0 1 3 15
243.5      503.2      762.3      782.8      793.2
805.9
0.479      0.0        0.1
0 1 4 15
236.7      243.1      251.0      263.1      499.9
687.9      715.0      733.0      767.9
0.361      0.625      0.1        0.1
0 1 5 15
315.0      336.2      342.1      370.2      576.4
709.1      723.0
0.5        0.0        0.1        0.0        30.0
0 1 6 15
```

Table II

Input Data for Test Cases - Computer Program 55VD (Page 2)

```
-0.076      0.0         0.05        0.1         0.11
1.0         1.52978     1.5805
0.49        0.51        0.1         0.0         30.0
0 1 7 15
-1.269      0.0         0.12812     0.23355     1.0
1.26393     2.3709      4.72868
0.02        0.02        0.15
0 2 1 15
0.7         0.0         0.08
0 2 2 15
0.4         0.7         0.08        0.08
0 2 3 15
0.3         0.0         0.1
0 2 4 15
0.4         0.7         0.08        0.08
0 2 5 15
0.7         0.0         0.08        0.0         10.0
0 2 6 15
0.65        0.75        0.08        0.0         10.0
0 2 7 15
0.04        0.04        0.08
```

The third page of answer listing for Configuration 1 specifies the required loading for the minimum induced drag. Given first is the ratio of the gross load on the pylon or fence to the gross load on the wing semispan. Then, tabular data specify the required loading on both wing and pylon in nondimensional form. The elliptical span loading is given for the wing. The answer listing for each of the seven configurations follows this basic form in program operation mode 1.

The answer listings for the test cases of Configurations 2-7 are normal in every respect. Configuration 2 requires two pages of answer listing to specify the loading on the wing. Relative to the iteration for the mapping constants of Configuration 1, the iterations for Configurations 4, 6, and 7 are significantly slower.

Program Operation Mode 2. The seven test cases forming the second group are examples of program operation in the mode that determines the mapping constants by the successive solution scheme. The first page of the answer listings does not give unscaled and scaled initial values of the mapping constants for each case, as the unscaled values are not part of the input data when Computer Program 55VD operates in the second mode. The subsequent pages of answer listings described the solution for each member of the series of configurations that are analyzed to arrive at the final configuration, and are identical in form with their counterparts when the program operation mode is 1. The iteration summary data, the geometry data, and the required loading are given for each configuration in the series.

For Configuration 1, the input data of Table II specify a test case geometry with pylon or vertical fence length $\ell = 0.08$ and fence location parameter $\eta = 0.7$ (see Figure 2). The initial values of the mapping constants are computed for the degenerate monoplane equivalent configuration ($\eta = 0.7$, $\ell = 0$) for which the potentials at B, C, and D are equal. For numerical reasons associated with the calculation of the influence matrix $\partial C_i/\partial p_j$, the initial values of the mapping constants at B and C are reduced by small and arbitrary amounts to separate them from the potential at D and from each other. Each member of the series of configurations has $\eta = 0.7$. The first member of the series has $\ell = 0.04$, and subsequent members of the series for Configuration 1 have ℓ incremented by 0.04 relative to their predecessors. The increment in ℓ between the final two configurations will be less than 0.04 unless the desired value of ℓ is an integer multiple of 0.04. For the test case, two configurations are analyzed to arrive at the desired geometry with $\eta = 0.7$, and $\ell = 0.08$. The answer listing for Configurations 2-7 generally follows the form of the listing for Configuration 1.

For the Configuration 2 test case, three configurations are analyzed to obtain the desired values $\ell_i = \ell_o = 0.08$ (see Figure 2), as the pylon length is incremented by 0.03 semispans between members of the series of configurations. For each of the three configurations, $\eta_i = 0.4$, and $\eta_o = 0.7$.

For the mode 2 test case of Configuration 3, the location of the second lifting surface above the main wing ℓ_H (see Figure 2) is 0.3 for each of the three configurations analyzed. The span of this surface is incremented by 0.04 semispans of the main wing between members of the series of configurations, and is 0.04 for the first configuration. The desired value of s_H is 0.1 s.

The test case for Configuration 4 has $\ell_i = \ell_o = 0.08$ with $\eta_i = 0.4$ and $\eta_o = 0.7$ (Figure 2). The lengths ℓ_i and ℓ_o are incremented nominally by 0.02 between members of the series of configurations. The iteration for the mapping constants of the first configuration is an example of a weakly converging iteration. The sixth cycle of the iteration reduces $\bar{e}$ from roughly 9×10^{-5} to 7×10^{-5} with the initial influence matrix. The ratio of these values of $\bar{e}$ is greater than 0.5, the critical value for Configuration 4. The answer listing shows that a new influence matrix was calculated and printed after the sixth cycle, and that the iteration converged rapidly with the new influence matrix.

Four geometries are analyzed for the test case of Configuration 5. The desired geometry is specified by $\eta = 0.7$, $\ell = 0.08$, and $\Gamma = 10$ degrees (Figure 2). The first iteration for the mapping constants is another example of weak convergence. The sixth cycle of the iteration reduces $\bar{e}$ from roughly 4.3×10^{-5} to 2.6×10^{-5}. The ratio is slightly greater than 0.5, the critical value for Configuration 5. A second influence matrix is therefore calculated and printed, and the iteration converges with this new matrix in one more cycle. The change of geometry between the second configuration and the first is maximum, as ℓ is increased by 0.03 and Γ is increased by 3 degrees. Ten cycles are required to achieve convergence in the mapping constant iteration. However, the change of geometry between the second and third configurations is less than maximum; ℓ changes by 0.02, rather than 0.03, while the change of Γ remains at 3 degrees. The third iteration converges somewhat faster than the second as it requires only 8 cycles. The change of geometry between the third and fourth configurations is the smallest. The parameter ℓ does not change, and Γ change by only 1 degree. The fourth iteration converges in only 4 cycles, the smallest number of cycles required for convergence in the four iterations.

The test case for Configuration 6 is quite similar to the test case for Configuration 5. For this test case, $\eta_i = 0.65$, $\eta_o = 0.75$, $\ell_i = 0.08$, and $\ell_o = 0.08$. None of the four mapping constant iterations requires the calculation and use of a second influence matrix.

The test case for Configuration 7 is specified by $\ell = 0.08$, $\ell_i = 0.04$, and $\ell_o = 0.04$ (Figure 2). The starting configuration for the successive solution scheme is not the monoplane, but the configuration specified by $\ell = 0.04$, $\ell_i = 0.01$, and $\ell_o = 0.01$ since, for Configuration 7, the convergence of the mapping constant iteration becomes weaker to the point of failure as the geometric parameters become small. Consequently, the increments in geometry between the neighboring members of the series of configurations are also smaller than those of Configurations 1-6, according

to the relationships

$$\Delta\ell = 0.01 \text{ for } \ell < 0.06; \text{ otherwise, } \Delta\ell = 0.02$$

$$\Delta\ell_i = 0.005 \text{ for } \ell_i < 0.02; \text{ otherwise, } \Delta\ell_i = 0.01 \qquad (14)$$

$$\Delta\ell_o = 0.005 \text{ for } \ell_o < 0.02; \text{ otherwise, } \Delta\ell_o = 0.01$$

The first iteration requires 16 cycles to reduce $\bar{e}$ from 1.2×10^{-2} to 1×10^{-8}. The ratio of successive values of $\bar{e}$ is less than 0.5 for each cycle so that a converged solution for the mapping constants is obtained with the initial influence matrix. The second iteration converges somewhat faster than the first as it requires only 12 cycles to reduce $\bar{e}$ by a directly comparable ratio. The geometry differences between the second and third configurations are twice as large as the geometry differences between the first and second configurations, as required by (14). The third iteration requires 18 cycles to converge. The answer listing for the fourth iteration shows for the first time a characteristic of the program coding logic common to Configurations 1-7 in either mode of program operation. Nominally, if $\bar{e}$ exceeds a definite fraction (either 0.5 or 0.7, depending on which Configuration of Figure 2 is being analyzed) of the value of $\bar{e}$ from the preceding cycle, the convergence of the iteration is defined to be too slow; a new influence matrix is calculated, printed in the answer listings, and then, used for the next cycle of the iteration. However, this convergence rate requirement is not applied to the first three cycles. The fourth iteration for the test case of Configuration 7 is an example of relaxation of the convergence rate requirement for the first three cycles. The critical value of the convergence rate is 0.5 for Configuration 7, and each of the first four cycles of the fourth iteration fails this requirement. The second influence matrix is calculated and printed after the fourth cycle, and the iteration converges rapidly with its subsequent use.

A fifth configuration is analyzed for the test case of Configuration 7. The analysis is unnecessary, and occurs only because the desired geometry is effectively an integer multiple of the geometry increment between members of the series of configurations. As a result, one or more of ℓ, ℓ_i, or ℓ_o for the fourth configuration is less than the desired value by a small round-off error. Extraneous output in this form can occur for each of the seven configurations of Figure 2.

Table III

Index for Answer Listing of Test Case

Configuration Number	Program Mode of Operation	Page Number
1	1	290
2	1	293
3	1	297
4	1	300
5	1	303
6	1	306
7	1	310
1	2	313
2	2	318
3	2	328
4	2	335
5	2	344
6	2	353
7	2	366

```
* * * * * * * * * * * * * * * * *   NEW CASE DATA LOADED BY SUBROUTINE INPUT   * * * * * * * * * * * * * * * *

                    CONFIGURATION   1

          PROGRAM OPERATION MODE NUMBER =   1

         NUMBER OF INTEGRATION STEPS =      15


INPUT POTENTIALS (CONSTANTS OF THE SCHWARZ-CHRISTOFFEL TRANSFORMATION)

                    A =   173.90000000
                    B =   196.50000000
                    C =   216.80000000
                    D =   242.50000000
                    E =   497.90000000
                    F =   823.00000000

TRANSFORMED POTENTIALS (LINEARLY SCALED)

                    A =     0.00000000
                    B =      .06975309
                    C =      .13240741
                    D =      .21172840
                    E =     1.00000000
                    F =     2.00339506
```

WING WITH ONE PYLON (CONFIGURATION 1)

TRANSFORMATION CONSTANTS (INDEPENDENT VARIABLES)						ERRORS IN REQUIRED CONDITIONS				
A	B	C	D	E	F	PYLON CLOSURE	WING CLOSURE	PYLON LOCATION	PYLON LENGTH	SQUARE ROOT OF SUM OF SQR. ERRORS
0.0	.06975309	.13240741	.21172840	1.0	2.00339506	-.02015710	-.03104806	.00019072	.00097906	.03703088
-.547	-1.593	-1.735	.632							
4.148	-1.593	-1.946	-1.540							
1.937	-1.611	-1.091	.567							
.014	-1.547	-.377	-.097							
0.0	.07313918	.13229431	.21466930	1.0	1.97653733	.00016081	.00099988	-.00019200	.00030633	.00107532
0.0	.07335083	.13250471	.21473233	1.0	1.97734254	-.00005205	-.00005756	.00001157	.00000015	.00007847
0.0	.07335602	.13250124	.21473835	1.0	1.97728644	.00000093	.00000330	-.00000085	.00000129	.00000377
0.0	.07335680	.13250198	.21473831	1.0	1.97728937	-.00000021	-.00000021	.00000005	-.00000000	.00000030
0.0	.07335682	.13250196	.21473834	1.0	1.97728916	.00000000	.00000001	-.00000000	.00000001	.00000001
0.0	.07335682	.13250196	.21473834	1.0	1.97728916	.00000000	.00000001	-.00000000	.00000001	.00000001

TRANSFORMATION CONSTANTS IN EXPONENTIAL FORMAT

0.	7.33568156E-02	1.32501964E-01	2.14738335E-01	1.00000000E+00	1.97728916E+00

CLOSURE INTEGRAL (PYLON) =	.00000000
CLOSURE INTEGRAL (WING) =	.00000001
PYLON SEMISPAN LOCATION =	.50000000
PYLON LENGTH (SEMISPANS) =	.12500001
INDUCED DRAG INTEGRAL =	1.51209870
DISTANCE FROM A TO B =	.48938697
DISTANCE FROM B TO C =	-.12234675
DISTANCE FROM C TO D =	.12234675
DISTANCE FROM D TO E =	.48938698
DISTANCE FROM E TO F =	-.97877393
LIFTING EFFICIENCY K =	1.00483666

WING WITH ONE PYLON

RATIO OF LOADING ON PYLON TO LOAD ON WING = .00895395

WING SPANWISE COORDINATE	WING LOADING	ELLIPTICAL WING LOADING	PYLON COORDINATE	PYLON LOADING
0.00000000	1.27940364	1.27323950	0.00000000	1.27760022
.00765242	1.27986731	1.27320025	.04143026	1.27647338
.01394202	1.27978886	1.27311575	.07350650	1.27404979
.02194368	1.27961900	1.27293291	.11552909	1.26881331
.03325770	1.27924947	1.27253516	.17455574	1.25745750
.04970680	1.27844199	1.27168559	.25913859	1.23280221
.07387983	1.27667454	1.26975993	.37952790	1.17956031
.10935263	1.27282890	1.26560391	.54406365	1.06715413
.15064506	1.26647687	1.25870918	.71099796	.89153206
.18307467	1.26007457	1.25172049	.81913313	.72540991
.21062139	1.25361959	1.24462294	.89321873	.56721437
.23599976	1.24711194	1.23739802	.94373439	.41565277
.25832752	1.24050403	1.23002243	.97630810	.26908105
.27957698	1.23380785	1.22246614	.99433712	.12087662
.29975913	1.22700881	1.21468948	1.00000000	.00000034
.29975913	1.22700881	1.21468948		
.31918636	1.22008536	1.20663885		
.33814249	1.21300455	1.19823923		
.35691559	1.20567562	1.16937967		
.37584869	1.19797229	1.17988691		
.39543151	1.18980620	1.16946471		
.41652873	1.18089475	1.15753074		
.44121916	1.17051566	1.14260434		
.46101552	1.16240954	1.12986264		
.47399412	1.15727323	1.12112304		
.48266793	1.15393919	1.11510754		
.48858463	1.15171483	1.11092248		
.49275318	1.15017305	1.10793370		
.49592016	1.14901599	1.10564059		
.50000000	1.14754378	1.10265775		
.50000000	1.05602721	1.10265775		
.51352064	1.05062253	1.09253783		
.52406171	1.04568891	1.08439299		
.53804072	1.03831702	1.07323757		
.55818930	1.02610114	1.05642416		
.58857375	1.00441065	1.02934223		
.63600800	.96364262	.98253659		
.71090109	.88226347	.89545916		
.80067622	.75399244	.76279437		
.86727338	.62749658	.63385960		
.91755183	.50077470	.50626066		
.95459890	.37420249	.37929063		
.98011191	.24714295	.25266873		
.99507128	.11585665	.12625723		
.99999997	.00000036	0.00000000		

```
* * * * * * * * * * * * * * * * * *   NEW CASE DATA LOADED BY SUBROUTINE INPUT   * * * * * * * * * * * * * * * * * *

                    CONFIGURATION   2

          PROGRAM OPERATION MODE NUMBER =   1

         NUMBER OF INTEGRATION STEPS =      15


INPUT POTENTIALS (CONSTANTS OF THE SCHWARZ-CHRISTOFFEL TRANSFORMATION)

                    A =   260.50000000
                    B =   268.50000000
                    C =   277.00000000
                    D =   287.80000000
                    E =   301.40000000
                    F =   319.90000000
                    G =   343.90000000
                    H =   527.70000000
                    I =   785.10000000

TRANSFORMED POTENTIALS (LINEARLY SCALED)

                    A =     -.48663102
                    B =     -.34402852
                    C =     -.19251337
                    D =     0.00000000
                    E =      .24242424
                    F =      .57219251
                    G =     1.00000000
                    H =     4.27629234
                    I =     8.86452763
```

WING WITH TWO PYLONS BELOW (CONFIGURATION 2)

TRANSFORMATION CONSTANTS (INDEPENDENT VARIABLES)									ERRORS IN REQUIRED CONDITIONS							
A	B	C	D	E	F	G	H	I	INBOARD PYLON CLOSURE	OUTBOARD PYLON CLOSURE	WING CLOSURE	INBOARD PYLON LOCATION	OUTBOARD PYLON LOCATION	INBOARD PYLON LENGTH	OUTBOARD PYLON LENGTH	SQUARE ROOT OF SUM OF SQR. ERRORS
-.487	-.344	-.193	0.0	.242	.572	1.0	4.276	8.865	-.00652	-.00866	.01259	.00697	-.00292	.01284	.00845	.02385840
-.227	-.035	-.339	-.549	-.131	.251	.067										
1.494	-.046	-.339	-.625	-.137	-.549	.076										
.852	-.132	-.339	-.404	-.159	.250	.136										
-.089	.644	-.339	-.238	-.185	-.136	-.229										
.049	-.908	.677	.296	.498	.132	.033										
-.009	.336	-.340	-.093	-.046	-.037	.097										
.000	.002	.680	.078	-.200	.025	.029										
-.476	-.320	-.181	0.0	.282	.591	1.0	4.266	8.868	-.00037	-.00062	.00001	.00004	.00001	.00075	.00027	.00107669
-.475	-.318	-.180	0.0	.283	.591	1.0	4.263	8.859	-.00006	-.00006	.00000	.00001	-.00000	.00009	.00003	.00012381
-.475	-.318	-.180	0.0	.283	.591	1.0	4.262	8.858	-.00001	-.00001	.00000	.00000	-.00000	.00001	.00000	.00001493
-.475	-.318	-.180	0.0	.283	.591	1.0	4.262	8.858	-.00000	-.00000	.00000	.00000	-.00000	.00000	.00000	.00000182
-.475	-.318	-.180	0.0	.283	.591	1.0	4.262	8.858	-.00000	-.00000	.00000	.00000	-.00000	.00000	.00000	.00000022
-.475	-.318	-.180	0.0	.283	.591	1.0	4.262	8.858	-.00000	-.00000	.00000	.00000	-.00000	.00000	.00000	.00000003
-.475	-.318	-.180	0.0	.283	.591	1.0	4.262	8.858	-.00000	-.00000	.00000	.00000	-.00000	.00000	.00000	.00000003

TRANSFORMATION CONSTANTS IN EXPONENTIAL FORMAT

-4.75036445E-01 -3.17978368E-01 -1.79663494E-01 0. 2.83353322E-01 5.91088351E-01 1.0F+00 4.24215188E+00 8.85021759E+00

CLOSURE INTEGRAL(INBOARD PYLON) = -.00000002
CLOSURE INTEGRAL(OUTBOARD PYLON) = -.00000001
CLOSURE INTEGRAL(WING) = .00000000

INBOARD PYLON SEMISPAN LOCATION = .36100000
OUTBOARD PYLON SEMISPAN LOCATION = .62500000
INBOARD PYLON LENGTH (SEMISPANS) = .10000002
OUTBOARD PYLON LENGTH (SEMISPANS) = .10000000

INDUCED DRAG INTEGRAL = 33.58626893

DISTANCE FROM A TO B = 1.66328649
DISTANCE FROM B TO C = -.46074430
DISTANCE FROM C TO D = .46074422
DISTANCE FROM D TO E = 1.21636463
DISTANCE FROM E TO F = -.46074422
DISTANCE FROM F TO G = .46074418
DISTANCE FROM G TO H = 1.72779067
DISTANCE FROM H TO I = -4.60744179

LIFTING EFFICIENCY K = 1.00721496

WING WITH TWO PYLONS BELOW

RATIO OF INBOARD PYLON LOAD TO WING LOAD = .00343286
RATIO OF OUTBOARD PYLON LOAD TO WING LOAD = .00770205

WING SPANWISE COORDINATE	WING LOADING	ELLIPTICAL WING LOADING	INBOARD PYLON COORDINATE	INBOARD PYLON LOADING	OUTBOARD PYLON COORDINATE	OUTBOARD PYLON LOADING
0.000000	1.280506	1.273240	0.000000	1.270836	0.000000	1.276577
.005495	1.280489	1.273220	.041220	1.269774	.040934	1.275485
.009757	1.280452	1.273179	.073134	1.267489	.072629	1.273136
.015357	1.280371	1.273089	.114945	1.262551	.114158	1.268059
.023276	1.280197	1.272895	.173679	1.251830	.172509	1.257045
.034793	1.279815	1.272469	.257854	1.228508	.256181	1.233110
.051730	1.278980	1.271535	.377707	1.177930	.375455	1.181327
.076623	1.277162	1.269496	.541664	1.070235	.539008	1.071541
.105683	1.274156	1.266109	.708382	.899620	.705965	.898707
.128591	1.271122	1.262669	.816773	.735909	.814955	.733846
.148266	1.268059	1.259167	.891368	.578154	.890188	.575678
.165907	1.264961	1.255594	.942500	.425574	.941857	.423178
.182156	1.261817	1.251938	.975671	.277032	.975405	.275005
.197416	1.258626	1.248182	.994155	.127362	.994095	.125474
.211973	1.255377	1.244306	1.000000	.000000	1.000000	.000000
.211973	1.255377	1.244306				
.226054	1.252058	1.240281				
.239864	1.248646	1.236069				
.253615	1.245097	1.231611				
.267563	1.241382	1.226818				
.282079	1.237425	1.221535				
.297822	1.233080	1.215462				
.316381	1.227973	1.207836				
.331355	1.223940	1.201309				
.341208	1.221361	1.196830				
.347804	1.219675	1.193748				
.352307	1.218530	1.191605				
.355481	1.217726	1.190076				
.357893	1.217121	1.188903				
.361000	1.216348	1.187380				
.361000	1.172722	1.187380				
.364210	1.171909	1.185789				
.366701	1.171254	1.184544				
.369978	1.170359	1.182891				
.374621	1.169028	1.180520				
.381405	1.166951	1.176993				
.391480	1.163583	1.171618				
.406594	1.157931	1.163243				
.424857	1.150227	1.152613				
.439831	1.143283	1.143472				
.453149	1.136694	1.135010				
.465470	1.130294	1.126898				
.477147	1.123993	1.118953				
.488404	1.117730	1.111052				
.499404	1.111459	1.103095				
.499404	1.111459	1.103095				
.510287	1.105070	1.094991				
.521189	1.098429	1.086635				
.532264	1.091595	1.077897				
.543717	1.084479	1.068590				
.555862	1.076932	1.058412				
.569277	1.068671	1.046788				

.585387	1.056990	1.032284
.598577	1.051369	1.019948
.607328	1.046511	1.011527
.613208	1.043347	1.005759
.617229	1.041232	1.001764
.620066	1.039765	.998920
.622222	1.038664	.996744
.625000	1.037262	.993922
.625000	.938939	.993922
.634972	.933574	.983622
.642742	.928884	.975406
.653036	.922000	.964258
.667847	.910810	.947670
.690120	.890581	.921437
.724823	.852737	.877180
.779847	.779919	.797010
.846785	.664943	.677282
.897292	.552512	.562055
.935896	.441532	.448531
.964586	.330345	.335841
.984455	.216894	.223626
.996144	.097488	.111705
1.000000	.000000	0.000000

```
* * * * * * * * * * * * * * * * *   NEW CASE DATA LOADED BY SUBROUTINE INPUT   * * * * * * * * * * * * * * * * * * *

                    CONFIGURATION   3

          PROGRAM OPERATION MODE NUMBER =   1

         NUMBER OF INTEGRATION STEPS =      15

     INPUT POTENTIALS (CONSTANTS OF THE SCHWARZ-CHRISTOFFEL TRANSFORMATION)

                    A =   243.50000000
                    B =   503.20000000
                    C =   762.30000000
                    D =   782.80000000
                    E =   793.20000000
                    F =   805.90000000

          TRANSFORMED POTENTIALS (LINEARLY SCALED)

                    A =   -16.78964401
                    B =    -8.38511327
                    C =     0.00000000
                    D =      .66343042
                    E =     1.00000000
                    F =     1.41100324
```

WING WITH HORIZONTAL TAIL (BIPLANE CASE, CONFIGURATION 3)

TRANSFORMATION CONSTANTS (INDEPENDENT VARIABLES)						ERRORS IN REQUIRED CONDITIONS				
A	B	C	D	E	F	WING CLOSURE	HORIZONTAL TAIL (2ND WING) CLOSURE	HORIZONTAL TAIL (2ND WING) HEIGHT	HORIZONTAL TAIL (2ND WING) LENGTH	SQUARE ROOT OF SUM OF SQR. ERRORS
-16.78964401	-8.38511327	0.0	.66343042	1.0	1.41100324	-.00370086	-.00643251	.01410305	-.00061853	.01594842
.374	.000	-.056	-.010							
.315	-.505	-.197	-.154							
-.460	.838	.627	-.011							
.175	-.364	-.167	.122							
-17.81350393	-8.87478813	0.0	.64310631	1.0	1.42215306	-.00009611	.00003922	.00008229	.00017020	.00021568
-17.81863969	-8.87724610	0.0	.64361361	1.0	1.42156004	-.00000156	-.00000225	.00000568	.00000756	.00000984
-17.81911625	-8.87748026	0.0	.64362935	1.0	1.42153231	-.00000006	-.00000020	.00000037	.00000042	.00000060
-17.81914841	-8.87749605	0.0	.64363009	1.0	1.42153074	-.00000001	-.00000001	.00000002	.00000003	.00000004
-17.81915047	-8.87749705	0.0	.64363013	1.0	1.42153065	-.00000000	-.00000000	.00000000	.00000000	.00000000
-17.81915047	-8.87749705	0.0	.64363013	1.0	1.42153065	-.00000000	-.00000000	.00000000	.00000000	.00000000

TRANSFORMATION CONSTANTS IN EXPONENTIAL FORMAT

-1.78191506E+01	-8.87749705E+00	0.	6.43630134E-01	1.00000000E+00	1.42153065E+00

CLOSURE INTEGRAL (WING) =	-.00000000
CLOSURE INTEGRAL (HORIZONTAL) =	-.00000000
HORIZONTAL LOCATION ABOVE WING =	.47900000
HORIZONTAL SEMISPAN / WING B/2 =	.10000000
INDUCED DRAG INTEGRAL =	126.38356838
DISTANCE FROM A TO B =	8.94355683
DISTANCE FROM B TO C =	-8.94355684
DISTANCE FROM C TO D =	4.28396374
DISTANCE FROM D TO E =	.89435569
DISTANCE FROM E TO F =	-.89435570
LIFTING EFFICIENCY K =	1.00588931

WING WITH HORIZONTAL TAIL (BIPLANE)

RATIO OF HORIZONTAL TAIL LOAD TO WING LOAD = .00436010

WING SPAN COORDINATE	WING LOCAL LOADING	ELLIPTICAL WING LOADING	HORIZONTAL TAIL SPAN COORDINATE	HORIZONTAL TAIL LOADING
0.00000000	1.27039329	1.27323950	0.00000000	1.27197432
.03779348	1.26951646	1.27232986	.03980895	1.27097292
.06707278	1.26762746	1.27037227	.07063883	1.26881823
.10547516	1.26353496	1.26613729	.11104949	1.26415919
.15955382	1.25460897	1.25692832	.16787696	1.25403877
.23747440	1.23500999	1.23681695	.24951077	1.23198689
.34970875	1.19180929	1.19284521	.36632591	1.18400436
.50702072	1.09722937	1.09744825	.52779230	1.08104687
.67387072	.94095593	.94073209	.69488548	.91556043
.78795069	.78434531	.78398238	.80575076	.75420773
.87024855	.62757841	.62721419	.88346377	.59648396
.92923990	.47071715	.47043265	.93764056	.44200451
.96918619	.31379773	.31363618	.97334322	.29013747
.99238654	.15684374	.15681514	.99353297	.13760248
1.00000000	.00000001	0.00000000	.99999999	.00000010

```
* * * * * * * * * * * * * * * * * *   NEW CASE DATA LOADED BY SUBROUTINE INPUT   * * * * * * * * * * * * * * * * * * * *

                    CONFIGURATION   4

          PROGRAM OPERATION MODE NUMBER =   1

          NUMBER OF INTEGRATION STEPS =      15


INPUT POTENTIALS (CONSTANTS OF THE SCHWARZ-CHRISTOFFEL TRANSFORMATION)

                A =   236.70000000
                B =   243.10000000
                C =   251.00000000
                D =   263.10000000
                E =   499.90000000
                F =   687.90000000
                G =   715.00000000
                H =   733.00000000
                I =   767.90000000

TRANSFORMED POTENTIALS (LINEARLY SCALED)

                A =     -.05842000
                B =     -.04425758
                C =     -.02677584
                D =     0.00000000
                E =      .52400974
                F =      .94003098
                G =     1.00000000
                H =     1.03983182
                I =     1.11706130
```

WING WITH ONE PYLON ABOVE AND ONE PYLON BELOW (CONFIGURATION 4)

TRANSFORMATION CONSTANTS (INDEPENDENT VARIABLES)									ERRORS IN REQUIRED CONDITIONS							
A	B	C	D	E	F	G	H	I	INBOARD PYLON CLOSURE	OUTBOARD PYLON CLOSURE	WING CLOSURE	INBOARD PYLON LOCATION	OUTBOARD PYLON LOCATION	INBOARD PYLON LENGTH	OUTBOARD PYLON LENGTH	SQUARE ROOT OF SUM OF SQR. ERRORS
-.058	-.044	-.027	0.0	.524	.940	1.0	1.040	1.117	.00807	.02110	.00120	-.04417	-.03036	.01141	.01278	.06064577
-2.069	.033	-2.735	-4.658	.782	2.265	.153										
12.609	.033	-2.739	-5.142	.782	-4.632	.153										
-16.256	-.066	5.506	12.467	-1.571	.623	-.308										
.028	-.086	5.466	.552	-1.056	.205	-.243										
.025	7.323	5.477	.818	-6.792	.291	.009										
-.012	-4.833	-2.729	-.410	2.896	-.146	1.580										
-.012	.200	-2.728	-.413	2.509	-.146	-.587										
-.067	-.046	-.027	0.0	.528	.946	1.0	1.041	1.129	-.00095	.00066	.00006	-.00268	-.00076	.00218	.00106	.00386816
-.067	-.046	-.027	0.0	.528	.947	1.0	1.041	1.129	-.00056	-.00003	.00001	-.00026	.00002	.00055	.00013	.00083983
-.067	-.046	-.027	0.0	.528	[illegible]	1.0	1.041	1.129	-.00019	-.00002	.00000	-.00002	.00001	.00013	.00001	.00023783
-.067	-.046	-.027	0.0	.528	.947	1.0	1.041	1.129	-.00006	-.00000	.00000	-.00000	.00000	.00003	.00000	.00006506
-.067	-.046	-.027	0.0	.528	.947	1.0	1.041	1.129	-.00002	-.00000	-.00000	-.00000	.00000	.00001	.00000	.00001717
-.067	-.046	-.027	0.0	.528	.947	1.0	1.041	1.129	-.00000	-.00000	-.00000	-.00000	.00000	.00000	.00000	.00000447
-.067	-.046	-.027	0.0	.528	.947	1.0	1.041	1.129	-.00000	-.00000	-.00000	-.00000	-.00000	.00000	.00000	.00000116
-.067	-.046	-.027	0.0	.528	.947	1.0	1.041	1.129	-.00000	.00000	-.00000	-.00000	-.00000	.00000	.00000	.00000030
-.067	-.046	-.027	0.0	.528	.947	1.0	1.041	1.129	-.00000	.00000	-.00000	-.00000	-.00000	.00000	.00000	.00000008
-.067	-.046	-.027	0.0	.528	.947	1.0	1.041	1.129	-.00000	.00000	-.00000	-.00000	-.00000	.00000	.00000	.00000002
-.067	-.046	-.027	0.0	.528	.947	1.0	1.041	1.129	-.00000	.00000	-.00000	-.00000	-.00000	.00000	.00000	.00000002

TRANSFORMATION CONSTANTS IN EXPONENTIAL FORMAT

-6.74549421E-02 -4.59884446E-02 -2.65863535E-02 0. 5.28106499E-01 9.46858707E-01 1.0E+00 1.04097812E+00 1.12887315E+00

CLOSURE INTEGRAL (LOWER PYLON) = -.00000002
CLOSURE INTEGRAL (UPPER PYLON) = .00000000
CLOSURE INTEGRAL (WING) = -.00000000

INBOARD PYLON SEMISPAN LOCATION = .36100000
OUTBOARD PYLON SEMISPAN LOCATION = .62500000
INBOARD PYLON LENGTH (SEMISPANS) = .10000001
OUTBOARD PYLON LENGTH (SEMISPANS) = .10000000

INDUCED DRAG INTEGRAL = .55246038

DISTANCE FROM A TO B = .21313834
DISTANCE FROM B TO C = -.05904110
DISTANCE FROM C TO D = .05904109
DISTANCE FROM D TO E = .37727257
DISTANCE FROM E TO F = -.22140409
DISTANCE FROM F TO G = .05904109
DISTANCE FROM G TO H = -.05904109
DISTANCE FROM H TO I = -.36900681

LIFTING EFFICIENCY K = 1.00895668

WING WITH PYLON ABOVE AND PYLON BELOW

RATIO OF INBOARD PYLON LOAD TO WING LOAD = .00387777
RATIO OF OUTBOARD PYLON LOAD TO WING LOAD = .00792519

WING SPAN COORDINATE	WING LOADING	WING ELLIPTICAL LOADING	INBOARD PYLON COORDINATE	INBOARD PYLON LOADING	OUTBOARD PYLON COORDINATE	OUTBOARD PYLON LOADING
0.000000	1.286537	1.273240	0.000000	1.275376	0.000000	1.277148
.005522	1.286519	1.273220	.041518	1.274261	.035369	1.276328
.009804	1.286482	1.273178	.073662	1.271865	.062780	1.274563
.015432	1.286401	1.273088	.115770	1.266688	.098756	1.270741
.023390	1.286226	1.272891	.174910	1.255458	.149490	1.262423
.034962	1.285843	1.272461	.259633	1.231069	.222823	1.244214
.051978	1.285004	1.271518	.380159	1.178369	.329182	1.204191
.076979	1.283180	1.269461	.544722	1.066959	.480533	1.116338
.106148	1.280168	1.266046	.711471	.892494	.645649	.969071
.129126	1.277135	1.262580	.819392	.727027	.762785	.818634
.148847	1.274079	1.259056	.893319	.569075	.850577	.664289
.166515	1.270993	1.255464	.943746	.417436	.916067	.505379
.182778	1.267873	1.251791	.976290	.270566	.962314	.341320
.198039	1.264714	1.248022	.994327	.122156	.990387	.171429
.212587	1.261504	1.244136	1.000000	.000000	1.000000	.000000
.212587	1.261504	1.244136				
.226649	1.258232	1.240105				
.240430	1.254881	1.235891				
.254143	1.251426	1.231435				
.268042	1.247827	1.226648				
.282499	1.244016	1.221377				
.298169	1.239864	1.215324				
.316632	1.235037	1.207729				
.331524	1.231279	1.201234				
.341321	1.228906	1.196777				
.347880	1.227371	1.193712				
.352357	1.226351	1.191581				
.355513	1.225646	1.190060				
.357911	1.225117	1.188895				
.361000	1.224447	1.187380				
.361000	1.174990	1.187380				
.376752	1.171124	1.179420				
.389114	1.167444	1.172895				
.405688	1.161665	1.163755				
.430080	1.151619	1.149469				
.468091	1.132968	1.125137				
.529709	1.097646	1.079935				
.629057	.939776	.989765				
.746952	.829022	.846549				
.832730	.694937	.704966				
.896599	.557428	.563846				
.943232	.418208	.422889				
.975169	.278146	.281976				
.993846	.137484	.141034				
1.000000	.000000	0.000000				

```
* * * * * * * * * * * * * * * * * *   NEW CASE DATA LOADED BY SUBROUTINE INPUT   * * * * * * * * * * * * * * * * * * * *

                    CONFIGURATION   5

          PROGRAM OPERATION MODE NUMBER =   1

         NUMBER OF INTEGRATION STEPS =      15

     INPUT POTENTIALS (CONSTANTS OF THE SCHWARZ-CHRISTOFFEL TRANSFORMATION)

                A =   315.00000000
                B =   336.20000000
                C =   347.10000000
                D =   370.20000000
                E =   576.40000000
                F =   709.10000000
                G =   723.00000000

          TRANSFORMED POTENTIALS (LINEARLY SCALED)

                A =    -.08825978
                B =     0.00000000
                C =      .04537805
                D =      .14154871
                E =     1.00000000
                F =     1.55245629
                G =     1.61032473
```

WING WITH BREAK STATION, PYLON AT THE BREAK, AND DIHEDRAL OUTBOARD OF THE BREAK (CONFIGURATION 5)

TRANSFORMATION CONSTANTS (INDEPENDENT VARIABLES)							ERRORS IN REQUIRED CONDITIONS					
A	B	C	D	E	F	G	INBOARD WING CLOSURE	PYLON CLOSURE	OUTBOARD WING CLOSURE	PYLON LENGTH	PYLON SPANWISE LOCATION	SQUARE ROOT OF SUM OF SQR. ERRORS
-.08825978	0.0	.04537885	.14154871	1.0	1.55245629	1.61032473	-.05946285	.02263960	.00844316	-.00134989	-.05787545	.08643529
-1.910	-.100	-.391	.217	-2.288								
-2.122	2.538	-.560	.425	-2.474								
-.742	1.146	.357	1.175	-.885								
1.126	.010	-2.381	.339	1.439								
-2.959	-.009	-1.582	-.188	-.829								
-.08285841	0.0	.05445546	.15077523	1.0	1.53067524	1.58244054	-.00109260	-.00033943	.00008243	.00025600	.00191954	.00225006
-.08227413	0.0	.05386006	.14924467	1.0	1.53045550	1.58101953	.00018761	-.00005543	.00018610	-.00004135	.00009541	.00028933
-.08225677	0.0	.05384448	.14924617	1.0	1.53039776	1.58100618	-.00000704	-.00000010	-.00000656	.00000055	-.00000609	.00001140
-.08225972	0.0	.05384488	.14924723	1.0	1.53039993	1.58100760	.00000013	.00000035	.00000008	.00000003	.00000027	.00000047
-.08225932	0.0	.05384498	.14924726	1.0	1.53039983	1.58100744	-.00000000	-.00000006	.00000001	.00000000	-.00000001	.00000006
-.08225937	0.0	.05384495	.14924724	1.0	1.53039984	1.58100746	.00000000	.00000001	.00000000	-.00000000	.00000000	.00000001
-.08225937	0.0	.05384495	.14924724	1.0	1.53039984	1.58100746	.00000000	.00000001	.00000000	-.00000000	.00000000	.00000001

TRANSFORMATION CONSTANTS IN EXPONENTIAL FORMAT

-8.22593748E-02	0.	5.38449527E-02	1.49247237E-01	1.00000000E+00	1.53039984E+00	1.58100746E+00

CLOSURE INTEGRAL (INBOARD WING) =	.00000000
CLOSURE INTEGRAL (PYLON) =	.00000001
CLOSURE INTEGRAL (OUTBOARD WING) =	.00000000
PYLON LENGTH (SEMISPANS) =	.10000000
PYLON LOCATION (SEMISPANS) =	.50000000
INDUCED DRAG INTEGRAL =	1.01657316
DISTANCE FROM A TO B =	.39061157
DISTANCE FROM B TO C =	-.07812230
DISTANCE FROM C TO D =	.07812231
DISTANCE FROM D TO E =	-.45103939
REAL PART OF D TO E =	.22551969
IMAJ. PART OF D TO E =	.39061157
DISTANCE FROM E TO F =	.45103939
REAL PART OF E TO F =	-.22551969
IMAJ. PART OF E TO F =	-.39061157
DISTANCE FROM F TO G =	-.39061157
OUTBOARD DIHEDRAL (DEGREES) =	30.00000000
LIFTING EFFICIENCY K =	1.06039716

WING WITH PYLON AND DIHEDRAL OUTBOARD OF PYLON

RATIO OF PYLON LOAD TO WING LIFT LOAD = .0085335

RATIO OF OUTBOARD WING PANEL NORMAL LOAD TO WING LIFT LOAD = .4400147

WING SPAN COORDINATE	WING LIFT LOADING	OUTBOARD WING NORMAL LOADING	ELLIPTICAL WING LOADING	PYLON COORDINATE	PYLON LOADING
0.0000000	1.2776048		1.2732395	0.0000000	1.3434227
.0083103	1.2775695		1.2731955	.0416206	1.3386105
.0147547	1.2774936		1.2731009	.0738443	1.3317490
.0232215	1.2773294		1.2728962	.1160606	1.3198347
.0351905	1.2769724		1.2724509	.1753602	1.2982312
.0525829	1.2761929		1.2714781	.2603362	1.2582630
.0781129	1.2744896		1.2693491	.3812827	1.1838987
.1154828	1.2707988		1.2647209	.5465226	1.0472112
.1587766	1.2647507		1.2570879	.7138989	.8548036
.1925638	1.2587199		1.2494101	.8219143	.6846580
.2212802	1.2527053		1.2416763	.8955086	.5291626
.2467440	1.2467049		1.2338718	.9453247	.3854983
.2699256	1.2407161		1.2259783	.9771578	.2469841
.2914283	1.2347355		1.2179716	.9945878	.1018079
.3116721	1.2287581		1.2098191	1.0000000	.0000010
.3116721	1.2287581		1.2098191		
.3309809	1.2227776		1.2014765		
.3496335	1.2167836		1.1928810		
.3679056	1.2107640		1.1839385		
.3861146	1.2046932		1.1745008		
.4047020	1.1985335		1.1643121		
.4244353	1.1922091		1.1528652		
.4471512	1.1855309		1.1388598		
.4650997	1.1810287		1.1271458		
.4767652	1.1786155		1.1192166		
.4845291	1.1772887		1.1137983		
.4898147	1.1765375		1.1100440		
.4935354	1.1760899		1.1073692		
.4963609	1.1758058		1.1053199		
.5000000	1.1755457		1.1026578		
.5000000	1.0609045	1.2250269	1.1026578		
.5041290	1.0601100	1.2241095	1.0996057		
.5088979	1.0586057	1.2223725	1.0960388		
.5163663	1.0555253	1.2188156	1.0903613		
.5287675	1.0490691	1.2113606	1.0806833		
.5501020	1.0352796	1.1954379	1.0632791		
.5879093	1.0052785	1.1607957	1.0299576		
.6556433	.9388810	1.0841264	.9613859		
.7473512	.8246520	.9522201	.8459775		
.8226456	.7044339	.8134102	.7239060		
.8842543	.5774195	.6667466	.5946100		
.9330823	.4441983	.5129160	.4579369		
.9691869	.3035006	.3504523	.3136327		
.9919584	.1535064	.1772540	.1611669		
1.0000000	.0000005	.0000006	0.0000000		

```
* * * * * * * * * * * * * * * * *   NEW CASE DATA LOADED BY SUBROUTINE INPUT   * * * * * * * * * * * * * * * * * * * *

                    CONFIGURATION   6

          PROGRAM OPERATION MODE NUMBER =   1

         NUMBER OF INTEGRATION STEPS =      15

     INPUT POTENTIALS (CONSTANTS OF THE SCHWARZ-CHRISTOFFEL TRANSFORMATION)

                    A =     -.07600000
                    B =     0.00000000
                    C =      .05000000
                    D =      .10000000
                    E =      .11000000
                    F =     1.00000000
                    G =     1.52978000
                    H =     1.58050000

          TRANSFORMED POTENTIALS (LINEARLY SCALED)

                    A =     -.07600000
                    B =     0.00000000
                    C =      .05000000
                    D =      .10000000
                    E =      .11000000
                    F =     1.00000000
                    G =     1.52978000
                    H =     1.58050000
```

WING WITH TWO BREAK STATIONS, PYLON AT THE INBOARD STATION, AND DIHEDRAL OUTBOARD OF THE OUTBOARD STATION (CONFIGURATI[illegible])

TRANSFORMATION CONSTANTS (INDEPENDENT VARIABLES)								ERRORS IN REQUIRED CONDITIONS						
A	B	C	D	E	F	G	H	INBOARD WING CLOSURE	PYLON CLOSURE	OUTBOARD WING CLOSURE	PYLON LOC.	DIHEDRAL BREAK LOC.	PYLON LENGTH	SQU[illegible] OF [illegible] SQR. [illegible]
-.07600	0.0	.05000	.10000	.11000	1.0	1.52978	1.58050	.04183	-.00422	-.00840	.04021	-.00445	.00619	.[illegible]9
-1.918	-.260	.397	-2.841	-.237	.243									
-2.018	3.212	.549	-3.191	-.330	.468									
-2.944	2.355	1.448	-2.957	-.840	2.231									
2.096	-.144	-1.701	1.673	.976	.217									
1.057	.034	2.340	.373	-1.027	.125									
2.176	-.009	-3.091	1.499	-1.373	.287									
-.07445	0.0	.04721	.09526	.10004	1.0	1.53877	1.59375	-.00160	.00002	.00093	.00037	-.00028	.00004	.[illegible]94
-.07397	0.0	.04687	.09460	.10070	1.0	1.53834	1.59318	.00040	-.00000	-.00026	.00008	.00014	.00001	.[illegible]24
-.07403	0.0	.04692	.09469	.10052	1.0	1.53841	1.59326	-.00007	.00000	.00005	-.00001	-.00002	-.00000	.[illegible]62
-.07402	0.0	.04691	.09467	.10055	1.0	1.53840	1.59325	.00001	-.00000	-.00001	.00000	.00000	.00000	.[illegible]742
-.07402	0.0	.04691	.09468	.10055	1.0	1.53840	1.59325	-.00000	.00000	.00000	-.00000	-.00000	-.00000	.[illegible]32
-.07402	0.0	.04691	.09468	.10055	1.0	1.53840	1.59325	.00000	-.00000	-.00000	.00000	.00000	.00000	.[illegible]63
-.07402	0.0	.04691	.09468	.10055	1.0	1.53840	1.59325	-.00000	.00000	.00000	-.00000	-.00000	-.00000	.[illegible]12
-.07402	0.0	.04691	.09468	.10055	1.0	1.53840	1.59325	.00000	-.00000	-.00000	.00000	.00000	.00000	.[illegible]02
-.07402	0.0	.04691	.09468	.10055	1.0	1.53840	1.59325	.00000	-.00000	-.00000	.00000	.00000	.00000	.[illegible]02

TRANSFORMATION CONSTANTS IN EXPONENTIAL FORMAT

-7.40183232E-02 0. 4.69141205E-02 9.46776651E-02 1.00546220E-01 1.00000000E+00 1.53839835E+00 1.593247[illegible]00

CLOSURE INTEGRAL (INBOARD WING) = .00000002
CLOSURE INTEGRAL (PYLON) = -.00000000
CLOSURE INTEGRAL (OUTBOARD WING) = -.00000001

PYLON LOCATION (SEMISPAN) = .49000000
DIHEDRAL BREAK LOCATION (SEMISPAN) = .51000001
PYLON LENGTH (SEMISPANS) = .10000000

INDUCED DRAG INTEGRAL = 1.04663021

DISTANCE FROM A TO B = .38957366
DISTANCE FROM B TO C = -.07950483
DISTANCE FROM C TO D = .07950483
DISTANCE FROM D TO E = .01590099
DISTANCE FROM E TO F = .44984090
DISTANCE FROM F TO G = -.44984092
DISTANCE FROM G TO H = -.40547463

DIHEDRAL ANGLE (DEGREES) = 30.00000000

LIFTING EFFICIENCY K = 1.05411106

WING WITH ONE PYLON INBOARD AND OUTBOARD DIHEDRAL WITH BREAK OUTBOARD OF PYLON

RATIO OF PYLON LOAD TO WING LIFT LOAD = .00567646

RATIO OF OUTBOARD WING NORMAL LOAD TO WING LIFT LOAD = .43430377

WING SPAN COORDINATE	WING LIFT LOADING	OUTBOARD WING NORMAL LOADING	ELLIPTICAL WING LOADING	PYLON COORDINATE	PYLON LOADING
0.0000000	1.2656047		1.2732395	0.0000000	1.2660866
.0080378	1.2655724		1.2731984	.0410245	1.2650567
.0142709	1.2655030		1.2731098	.0727894	1.2628410
.0224605	1.2653528		1.2729183	.1144107	1.2580511
.0340383	1.2650263		1.2725017	.1728941	1.2476513
.0500667	1.2643134		1.2715914	.2567612	1.2250143
.0755721	1.2627551		1.2695985	.3763207	1.1758665
.1117639	1.2593770		1.2652624	.5402451	1.0709119
.1537497	1.2538366		1.2581005	.7074142	.9036529
.1865739	1.2483046		1.2508826	.8162990	.7420181
.2145227	1.2427805		1.2435972	.8912414	.5851823
.2393528	1.2372620		1.2362298	.9425447	.4324820
.2620021	1.2317471		1.2287618	.9757464	.2829744
.2830537	1.2262334		1.2211692	.9941881	.1325214
.3029147	1.2207064		1.2134197	1.0000000	.0000003
.3029147	1.2207064		1.2134197		
.3219001	1.2151676		1.2054696		
.3402818	1.2096027		1.1972570		
.3583309	1.2039970		1.1886896		
.3763612	1.1983249		1.1796221		
.3948117	1.1925402		1.1698042		
.4144493	1.1865567		1.1587402		
.4371127	1.1801542		1.1451602		
.4550500	1.1757328		1.1337725		
.4667307	1.1732840		1.1260524		
.4745044	1.1718799		1.1207730		
.4797980	1.1710316		1.1171136		
.4835246	1.1704953		1.1145057		
.4863548	1.1701254		1.1125076		
.4900000	1.1696914		1.1099114		
.4900000	1.0978226		1.1099114		
.4904066	1.0977753		1.1096203		
.4907220	1.0977365		1.1093942		
.4911362	1.0976827		1.1090970		
.4917217	1.0976012		1.1086764		
.4925721	1.0974714		1.1080642		
.4938194	1.0972564		1.1071639		
.4956411	1.0968907		1.1058435		
.4977415	1.0963986		1.1043130		
.4993683	1.0959591		1.1031217		
.5007376	1.0955482		1.1021150		
.5019371	1.0951561		1.1012301		
.5030123	1.0947797		1.1004345		
.5039905	1.0944164		1.0997087		
.5048892	1.0940622		1.0990402		
.5048892	1.0940622		1.0990402		
.5057202	1.0937157		1.0984206		
.5064915	1.0933768		1.0978443		
.5072086	1.0930458		1.0973075		
.5078746	1.0927206		1.0968080		

.5084905	1.0924022		1.0963453
.5090541	1.0920905		1.0959212
.5095575	1.0917867		1.0955419
.5098240	1.0916067		1.0953408
.5099304	1.0915254		1.0952605
.5099728	1.0914885		1.0952285
.5099897	1.0914718		1.0952158
.5099964	1.0914641		1.0952107
.5099991	1.0914606		1.0952087
.5100000	1.0914590		1.0952080
.5100000	1.0914590	1.2603082	1.0952080
.5106287	1.0909258	1.2596926	1.0947330
.5122927	1.0897439	1.2583278	1.0934720
.5160882	1.0871076	1.2552837	1.0905748
.5241767	1.0811863	1.248[illegible]441	1.0843029
.5406934	1.0679876	1.2332060	1.0710742
.5738595	1.0384296	1.1990753	1.0427242
.6390577	.9717425	1.1220716	.9793233
.7327277	.8554813	.9878247	.8664675
.8116765	.7312656	.8443927	.7437048
.8768276	.6001515	.6929952	.6121798
.9286626	.4620928	.5335787	.4722771
.9670921	.3167171	.3657134	.3239460
.9913865	.1[illegible]4244	.1852422	.1667356
1.0000000	.0[illegible]0005	.0000006	0.0000000

```
* * * * * * * * * * * * * * * * * *   NEW CASE DATA LOADED BY SUBROUTINE INPUT   * * * * * * * * * * * * * * * * * * * *

                    CONFIGURATION   7

          PROGRAM OPERATION MODE NUMBER =   1

         NUMBER OF INTEGRATION STEPS =      15

INPUT POTENTIALS (CONSTANTS OF THE SCHWARZ-CHRISTOFFEL TRANSFORMATION)

                A =    -1.26900000
                B =     0.00000000
                C =      .12812000
                D =      .23355000
                E =     1.00000000
                F =     1.26393000
                G =     2.37090000
                H =     4.72868000

TRANSFORMED POTENTIALS (LINEARLY SCALED)

                A =     0.00000000
                B =      .55927721
                C =      .61574262
                D =      .66220802
                E =     1.00000000
                F =     1.11631996
                G =     1.60418687
                H =     2.64331424
```

WING WITH AN END-PLATED VERTICAL FENCE AT THE TIP (CONFIGURATION 7)

TRANSFORMATION CONSTANTS (INDEPENDENT VARIABLES)								ERRORS IN REQUIRED CONDITIONS						
A	B	C	D	E	F	G	H	WING CLOSURE	VERTICAL FENCE CLOSURE	END PLATE CLOSURE	VERTICAL FENCE LENGTH	INBOARD F. P. LENGTH	OUTBOARD E. P. LENGTH	SQUARE ROOT OF SUM OF SQR. ERRORS
0.0	.55928	.61574	.66221	1.0	1.11632	1.60419	2.64331	.00175	-.00052	-.00754	-.05016	.00001	.00001	.05075971
-1.171	.915	-.097	-.358	.153	.025									
-2.196	.815	.925	-.252	-.836	.037									
3.480	-1.478	-.944	.462	.594	-.070									
-.984	.206	-.284	.018	-.019	.249									
1.281	.270	-.011	.212	.016	.025									
-1.265	-.013	.006	-.073	-.013	-.015									
0.0	.46588	.56283	.61192	1.0	1.11335	1.82156	2.90721	-.00455	-.00523	.00190	.00249	-.00100	-.00046	.00768312
0.0	.47603	.56673	.61634	1.0	1.11450	1.81746	2.89453	-.00016	.00160	.00012	-.00037	-.00020	.00000	.00166767
0.0	.47400	.56532	.61524	1.0	1.11471	1.81648	2.89457	-.00005	-.00032	.00004	.00009	-.00004	.00001	.00034337
0.0	.47449	.56552	.61546	1.0	1.11463	1.81665	2.89455	-.00000	.00007	.00000	-.00002	-.00001	-.00000	.00007288
0.0	.47439	.56546	.61540	1.0	1.11465	1.81664	2.89459	-.00000	-.00001	.00000	.00001	-.00000	.00000	.00001610
0.0	.47441	.56547	.61541	1.0	1.11465	1.81665	2.89458	-.00000	.00000	.00000	-.00000	-.00000	-.00000	.00000358
0.0	.47440	.56547	.61541	1.0	1.11465	1.81665	2.89458	-.00000	-.00000	.00000	.00000	-.00000	.00000	.00000079
0.0	.47441	.56547	.61541	1.0	1.11465	1.81665	2.89458	.00000	.00000	.00000	-.00000	-.00000	-.00000	.00000017
0.0	.47441	.56547	.61541	1.0	1.11465	1.81665	2.89458	-.00000	-.00000	.00000	.00000	-.00000	.00000	.00000004
0.0	.47441	.56547	.61541	1.0	1.11465	1.81665	2.89458	.00000	.00000	-.00000	-.00000	-.00000	-.00000	.00000001
0.0	.47441	.56547	.61541	1.0	1.11465	1.81665	2.89458	.00000	.00000	-.00000	-.00000	-.00000	-.00000	.00000001

TRANSFORMATION CONSTANTS IN EXPONENTIAL FORMAT

0.	4.74405320E-01	5.65466529E-01	6.15410315E-01	1.00000000E+00	1.11464627E+00	1.81664562E+00	2.89450291E+00

WING CLOSURE INTEGRAL =	.00000000
VERTICAL FENCE CLOSURE INTEGRAL =	.00000001
END-PLATE CLOSURE INTEGRAL =	-.00000000
VERTICAL FENCE LENGTH =	.15000000
INBOARD END-PLATE LENGTH =	.02000000
OUTBOARD END-PLATE LENGTH =	.02000000
INDUCED DRAG INTEGRAL =	3.22223516
DISTANCE FROM A TO B =	1.31163887
DISTANCE FROM B TO C =	-.19674582
DISTANCE FROM C TO D =	-.02623278
DISTANCE FROM D TO E =	.05246555
DISTANCE FROM E TO F =	-.02623278
DISTANCE FROM F TO G =	.19674583
DISTANCE FROM G TO H =	-1.31163887
LIFTING EFFICIENCY K =	1.19236384

WING WITH END-PLATED VERTICAL FENCE AT THE TIP

RATIO OF VERTICAL FENCE LOAD TO WING LOAD = .0544725
RATIO OF INBOARD END PLATE LOAD TO WING LOAD = -.0005514
RATIO OF OUTBOARD END PLATE LOAD TO WING LOAD = .0009505

WING SPAN COORDINATE	WING LOADING	ELLIPTICAL WING LOADING	END PLATE SPAN COORDINATE	END PLATE LOADING, STARTING OUTBOARD	VERTICAL FENCE COORDINATE	VERTICAL FENCE LOADING
0.000000	1.179732	1.273240	1.000000	.000161	0.000000	1.506406
.017696	1.179584	1.273040	.996916	.572854	.019238	1.461099
.031417	1.179265	1.272611	.987104	1.176109	.034153	1.429990
.049441	1.178574	1.271682	.969474	1.826434	.053742	1.394464
.074912	1.177070	1.269662	.942418	2.493327	.081409	1.354597
.111897	1.173786	1.265243	.903346	3.230134	.121538	1.318228
.166097	1.166592	1.255554	.847471	4.037389	.180202	1.270440
.245144	1.150931	1.234389	.763411	4.975926	.265307	1.195522
.336103	1.125059	1.199169	.682528	5.683498	.362282	1.112829
.406463	1.098871	1.163317	.624214	6.125887	.436366	1.050370
.465720	1.072612	1.126731	.583539	6.400973	.497980	.998536
.517772	1.045806	1.089280	.555249	6.577074	.551417	.952998
.564697	1.018669	1.050802	.535141	6.697371	.598971	.912111
.607785	.991344	1.011681	.519803	6.786438	.642064	.874766
.647926	.963574	.969831	.500000	6.898010	.681670	.840146
.647926	.963574	.969831	.500000	-4.320025	.681670	.840146
.685796	.934703	.926655	.478469	-4.200382	.718518	.807697
.721968	.905233	.880988	.461803	-4.107176	.753208	.776957
.756989	.874851	.831970	.439977	-3.984315	.786289	.747880
.791471	.842249	.778212	.409344	-3.816875	.818340	.720401
.826242	.807604	.717236	.365527	-3.574341	.850101	.693610
.862713	.768331	.643870	.303411	-3.207800	.882773	.667991
.904232	.720490	.543728	.219298	-2.748122	.919129	.643102
.936811	.678265	.445424	.135533	-2.187987	.947035	.628447
.957938	.648245	.365386	.082703	-1.610790	.964873	.621800
.971993	.625100	.299222	.047575	-1.108882	.976651	.618783
.981562	.608404	.243375	.024401	-.701791	.984641	.617409
.988297	.592122	.194223	.009989	-.387797	.990255	.616777
.993412	.573837	.145910	.002318	-.157749	.994515	.616486
1.000000	.547051	0.000000	0.000000	0.000000	1.000000	.616348

```
* * * * * * * * * * * * * * * * *   NEW CASE DATA LOADED BY SUBROUTINE INPUT   * * * * * * * * * * * * * * * * * * * *

                    CONFIGURATION    1

          PROGRAM OPERATION MODE NUMBER =    2

          NUMBER OF INTEGRATION STEPS =      15
```

WING WITH ONE PYLON (CONFIGURATION 1)

TRANSFORMATION CONSTANTS (INDEPENDENT VARIABLES)						ERRORS IN REQUIRED CONDITIONS				
A	B	C	D	E	F	PYLON CLOSURE	WING CLOSURE	PYLON LOCATION	PYLON LENGTH	SQUARE ROO OF SUM OF SQR. ERROR
0.0	.26585716	.27585716	.28585716	1.0	2.00000000	-.00047871	.00102770	-.00987646	-.02950086	.0311308
-.002	-1.565	-1.277	.024							
1.714	-1.568	-2.245	-.614							
1.564	-1.595	-1.937	.593							
.000	-1.555	-.520	-.008							
0.0	.24879049	.28191880	.31755992	1.0	1.99773634	-.00143791	.00030132	-.00195255	-.00449570	.0051168
0.0	.24831805	.28452914	.32446070	1.0	1.99665773	-.00068663	.00005457	.00043021	-.00100274	.0012930
0.0	.24781911	.28461334	.32562266	1.0	1.99617640	-.00018063	.00002459	.00026037	-.00017324	.0003619
0.0	.24763661	.28451449	.32573128	1.0	1.99606313	-.00003370	.00000643	.00007168	-.00001639	.0000811
0.0	.24759588	.28447705	.32571913	1.0	1.99604441	-.00000366	.00000116	.00001349	.00000178	.0000141
0.0	.24758946	.28446824	.32571008	1.0	1.99604293	.00000019	.00000012	.00000150	.00000134	.0000020
0.0	.24758907	.28446688	.32570753	1.0	1.99604323	.00000023	-.00000001	-.00000007	.00000038	.0000004
0.0	.24758921	.28446678	.32570704	1.0	1.99604340	.00000007	-.00000001	-.00000009	.00000007	.0000001
0.0	.24758928	.28446681	.32570698	1.0	1.99604345	.00000001	-.00000000	-.00000003	.00000001	.0000000
0.0	.24758929	.28446683	.32570698	1.0	1.99604345	.00000000	-.00000000	-.00000001	-.00000000	.0000000
0.0	.24758929	.28446683	.32570698	1.0	1.99604345	.00000000	-.00000000	-.00000001	-.00000000	.0000000

TRANSFORMATION CONSTANTS IN EXPONENTIAL FORMAT

0.	2.47589294E-01	2.84466827E-01	3.25706984E-01	1.00000000E+00	1.99604345E+00

CLOSURE INTEGRAL (PYLON) =	.00000000
CLOSURE INTEGRAL (WING) =	-.00000000
PYLON SEMISPAN LOCATION =	.69999999
PYLON LENGTH (SEMISPANS) =	.04000000
INDUCED DRAG INTEGRAL =	1.56230150
DISTANCE FROM A TO B =	.69763597
DISTANCE FROM B TO C =	-.03986491
DISTANCE FROM C TO D =	.03986491
DISTANCE FROM D TO E =	.29898685
DISTANCE FROM E TO F =	-.99662282
LIFTING EFFICIENCY K =	1.00134404

WING WITH ONE PYLON

RATIO OF LOADING ON PYLON TO LOAD ON WING = .00156538

WING SPANWISE COORDINATE	WING LOADING	ELLIPTICAL WING LOADING	PYLON COORDINATE	PYLON LOADING
0.00000000	1.27372555	1.27323950	0.00000000	1.27378517
.01337968	1.27361242	1.27312553	.03884465	1.27281823
.02375409	1.27336850	1.27288023	.06893332	1.27073743
.03738216	1.27284070	1.27234956	.10838557	1.26623721
.05663976	1.27169308	1.27119554	.16390611	1.25645727
.08459983	1.26918663	1.26867495	.24378947	1.23512607
.12556602	1.26370480	1.26316215	.35849058	1.18861360
.18528220	1.25180380	1.25119381	.51816818	1.08833937
.25391182	1.23224220	1.23151209	.68542497	.92575040
.30688467	1.21264281	1.21180151	.79796473	.76569598
.35136427	1.19309533	1.19205626	.87784802	.60793230
.39027667	1.17336458	1.17226858	.93417346	.45228287
.42516711	1.15364339	1.15242800	.97167755	.29838824
.45697539	1.13396102	1.13252023	.99308718	.14409253
.48633001	1.11433232	1.11252512	1.00000000	.00000051
.48633001	1.11433232	1.11252512		
.51368378	1.09463490	1.09241350		
.53938588	1.07453543	1.07214241		
.56373240	1.05435817	1.05164171		
.58700542	1.03407336	1.03079306		
.60953280	1.01359770	1.00937443		
.63182523	.99272177	.98690179		
.65509270	.97081457	.96199421		
.67157332	.95593046	.94339166		
.68147445	.94733648	.93181003		
.68778972	.94220727	.92425726		
.69199957	.93901430	.91914893		
.69493333	.93690841	.91555364		
.69715105	.93538484	.91281630		
.69999999	.93351606	.90927488		
.69999999	.88366719	.90927488		
.70595244	.87953274	.90178364		
.71062889	.87582615	.89580935		
.71691253	.87025029	.88765562		
.72621315	.86089163	.87531346		
.74092993	.84393776	.85508516		
.76568982	.81123233	.81896037		
.80862168	.74418852	.74908369		
.86459094	.63684821	.63977276		
.90840119	.52939720	.53234093		
.94250034	.42293710	.42552325		
.96811068	.31779687	.31897563		
.98596477	.20826287	.21257177		
.99651201	.09203075	.10625115		
1.00000000	.00000001	0.00000000		

WING WITH ONE PYLON (CONFIGURATION 1)

TRANSFORMATION CONSTANTS (INDEPENDENT VARIABLES)

ERRORS IN REQUIRED CONDITIONS

A	B	C	D	E	F	PYLON CLOSURE	WING CLOSURE	PYLON LOCATION	PYLON LENGTH	SQUARE RO OF SUM O SQR. ERRO
0.0	.24758929	.28446683	.32570698	1.0	1.99604345	.00000000	-.00000000	-.00000001	-.04000000	.040000
-.015	-1.572	-1.250	.090							
1.816	-1.572	-1.713	-.610							
1.408	-1.592	-1.327	.468							
-.000	-1.560	-.530	-.031							
0.0	.20989447	.27921217	.36233206	1.0	1.98594281	-.00486526	.00082565	.00193244	-.00034972	.005311
0.0	.21215838	.28042197	.36562457	1.0	1.98329417	.00008339	.00004102	.00088355	-.00036515	.000960
0.0	.21093882	.27948010	.36499125	1.0	1.98327776	-.00012293	.00001785	.00003441	.00017002	.000213
0.0	.21117701	.27954721	.36492426	1.0	1.98325423	.00005490	-.00000316	-.00000611	-.00001413	.000057
0.0	.21112173	.27951794	.36488986	1.0	1.98328340	-.00000639	.00000001	-.00000743	.00000656	.000011
0.0	.21114020	.27952977	.36489755	1.0	1.98328105	.00000230	-.00000021	.00000017	-.00000226	.000003
0.0	.21113589	.27952789	.36489717	1.0	1.98328181	-.00000077	.00000005	-.00000005	.00000038	.000000
0.0	.21113696	.27952845	.36489762	1.0	1.98328145	.00000015	-.00000001	.00000007	-.00000012	.000000
0.0	.21113666	.27952827	.36489751	1.0	1.98328151	-.00000004	.00000000	-.00000001	.00000003	.000000
0.0	.21113673	.27952831	.36489752	1.0	1.98328149	.00000001	-.00000000	.00000000	-.00000001	.000000
0.0	.21113673	.27952831	.36489752	1.0	1.98328149	.00000001	-.00000000	.00000000	-.00000001	.000000

TRANSFORMATION CONSTANTS IN EXPONENTIAL FORMAT

0.	2.11136732E-01	2.79528311E-01	3.64897524E-01	1.00000000E+00	1.98328149E+00

CLOSURE INTEGRAL (PYLON) = .00000001
CLOSURE INTEGRAL (WING) = -.00000000

PYLON SEMISPAN LOCATION = .70000000
PYLON LENGTH (SEMISPANS) = .07999999

INDUCED DRAG INTEGRAL = 1.53493483

DISTANCE FROM A TO B = .68991469
DISTANCE FROM B TO C = -.07884738
DISTANCE FROM C TO D = .07884739
DISTANCE FROM D TO E = .29567772
DISTANCE FROM E TO F = -.98559241

LIFTING EFFICIENCY K = 1.00594760

WING WITH ONE PYLON

RATIO OF LOADING ON PYLON TO LOAD ON WING = .00618720

WING SPANWISE COORDINATE	WING LOADING	ELLIPTICAL WING LOADING	PYLON COORDINATE	PYLON LOADING
0.00000000	1.27385073	1.27323950	0.00000000	1.27695478
.01260075	1.27375229	1.27313841	.03986990	1.27591508
.02237153	1.27354001	1.27292084	.07074750	1.27367823
.03520741	1.27308063	1.27245013	.11122192	1.26884248
.05334806	1.27208168	1.27142638	.16814267	1.25834272
.07969422	1.26989981	1.2[illegible]018978	.24992094	1.23548550
.11832118	1.26512673	1.2[illegible]29547	.36696969	1.18584894
.17471072	1.25475876	1.2[illegible]65683	.52880285	1.07979664
.23970420	1.23770229	1.23611944	.69623545	.91065505
.29007210	1.22056242	1.21849631	.80715663	.74719227
.33255493	1.20342140	1.20077166	.88469248	.58868852
.36991061	1.18617460	1.18292525	.93853336	.43454992
.40360225	1.16878621	1.16493080	.97383752	.28386609
.43452835	1.15134936	1.14675354	.99368303	.13225738
.46330096	1.13386589	1.12834612	1.00000000	.00000165
.46330096	1.13386589	1.12834612		
.49037540	1.11632345	1.10964263		
.51612261	1.09851554	1.09054836		
.54088455	1.08017922	1.07091773		
.56502566	1.06149166	1.05051595		
.58902164	1.04227760	1.02892682		
.61367183	1.02215621	1.00530050		
.64088359	1.00009226	.97738712		
.66150235	.98411564	.95485717		
.67454286	.97461716	.93995084		
.68310277	.96839174	.92987515		
.68889200	.96435162	.92292547		
.69295447	.96160447	.91798193		
.69603524	.95957107	.91419566		
.70000000	.95701850	.90927487		
.70000000	.85825880	.90927487		
.70778212	.85298216	.89945559		
.71384575	.84840627	.89165340		
.72187946	.84173102	.88110569		
.73343932	.83095418	.86549082		
.75083532	.81234670	.84096127		
.77800347	.77833165	.79992543		
.82138096	.71075061	.72623032		
.87482555	.60674968	.61680570		
.91567281	.50365866	.51174616		
.94717998	.40149539	.40833108		
.97074567	.30143964	.30571761		
.98713750	.19772447	.20355732		
.99680644	.08584111	.10167528		
1.00000000	.00000001	0.00000000		

```
* * * * * * * * * * * * * * * * *   NEW CASE DATA LOADED BY SUBROUTINE INPUT   * * * * * * * * * * * * * * * * * * * *

                    CONFIGURATION   2

          PROGRAM OPERATION MODE NUMBER =   2

          NUMBER OF INTEGRATION STEPS =     15
```

WING WITH TWO PYLONS BELOW (CONFIGURATION 2)

TRANSFORMATION CONSTANTS (INDEPENDENT VARIABLES)									ERRORS IN REQUIRED CONDITIONS							
A	B	C	D	E	F	G	H	I	INBOARD PYLON CLOSURE	OUTBOARD PYLON CLOSURE	WING CLOSURE	INBOARD PYLON LOCATION	OUTBOARD PYLON LOCATION	INBOARD PYLON LENGTH	OUTBOARD PYLON LENGTH	SQUARE ROOT OF SUM OF SQR. ERRORS
-.413	-.049	-.025	0.0	.901	.951	1.0	4.529	9.470	-.00067	-.00048	.00120	-.01074	-.01040	-.01811	-.01950	.03055910
	-.003	-.000	-.318	-.502	-.101		.017	.005								
	.790	-.001	-.316	-.911	-.115		-.275	.006								
	.730	-.001	-.321	-.892	-.119		.224	.007								
	.000	.344	-.318	-.130	-.318		-.005	-.111								
	-.000	-.667	.639	.247	.824		.010	.006								
	.000	.324	-.321	-.117	-.293		-.005	.102								
	-.000	-.000	.636	.000	-.165		.003	.003								
-.441	-.114	-.059	0.0	.738	.866	1.0	4.123	8.680	-.00108	-.00162	.00007	.00326	.00149	-.00107	-.00061	.00426703
-.437	-.118	-.061	0.0	.735	.862	1.0	4.090	8.603	-.00013	-.00002	.00001	.00045	.00060	.00009	-.00001	.00077006
-.437	-.117	-.061	0.0	.734	.861	1.0	4.098	8.619	.00002	-.00000	-.00000	-.00000	-.00003	.00002	.00004	.00005769
-.437	-.117	-.061	0.0	.734	.862	1.0	4.098	8.620	.00000	.00001	.00000	-.00001	-.00001	-.00000	-.00001	.00001837
-.437	-.117	-.061	0.0	.734	.862	1.0	4.098	8.620	.00000	-.00000	-.00000	.00000	-.00000	-.00000	.00000	.00000275
-.437	-.117	-.061	0.0	.734	.862	1.0	4.098	8.620	-.00000	.00000	.00000	-.00000	.00000	.00000	-.00000	.00000070
-.437	-.117	-.061	0.0	.734	.862	1.0	4.098	8.620	.00000	-.00000	-.00000	.00000	-.00000	.00000	.00000	.00000016
-.437	-.117	-.061	0.0	.734	.862	1.0	4.098	8.620	-.00000	.00000	.00000	-.00000	.00000	.00000	-.00000	.00000004
-.437	-.117	-.061	0.0	.734	.862	1.0	4.098	8.620	.00000	-.00000	-.00000	.00000	-.00000	-.00000	.00000	.00000001
-.437	-.117	-.061	0.0	.734	.862	1.0	4.098	8.620	.00000	-.00000	-.00000	.00000	-.00000	-.00000	.00000	.00000001

TRANSFORMATION CONSTANTS IN EXPONENTIAL FORMAT

-4.37030770E-01 -1.17366192E-01 -6.12599948E-02 0. 7.34436890E-01 8.61607152E-01 1.0E+00 4.09834184E+00 8.61966495E+00

CLOSURE INTEGRAL (INBOARD PYLON) = .00000000
CLOSURE INTEGRAL (OUTBOARD PYLON) = -.00000001
CLOSURE INTEGRAL (WING) = -.00000000

INBOARD PYLON SEMISPAN LOCATION = .40000000
OUTBOARD PYLON SEMISPAN LOCATION = .69999999
INBOARD PYLON LENGTH (SEMISPANS) = .03000000
OUTBOARD PYLON LENGTH (SEMISPANS) = .03000000

INDUCED DRAG INTEGRAL = 32.15528106

DISTANCE FROM A TO B = 1.80924148
DISTANCE FROM B TO C = -.13569311
DISTANCE FROM C TO D = .13569311
DISTANCE FROM D TO E = 1.35693105
DISTANCE FROM E TO F = -.13569314
DISTANCE FROM F TO G = .13569312
DISTANCE FROM G TO H = 1.35693112
DISTANCE FROM H TO I = -4.52310365

LIFTING EFFICIENCY K = 1.00059735

WING WITH TWO PYLONS BELOW

RATIO OF INBOARD PYLON LOAD TO WING LOAD = .00036916
RATIO OF OUTBOARD PYLON LOAD TO WING LOAD = .00088030

WING SPANWISE COORDINATE	WING LOADING	ELLIPTICAL WING LOADING	INBOARD PYLON COORDINATE	INBOARD PYLON LOADING	OUTBOARD PYLON COORDINATE	OUTBOARD PYLON LOADING
0.000000	1.274169	1.273240	0.000000	1.272888	0.000000	1.273249
.007165	1.274137	1.273207	.038675	1.271935	.038592	1.272297
.012721	1.274067	1.273136	.068632	1.269885	.068487	1.270250
.020020	1.273916	1.272984	.107914	1.265449	.107688	1.265821
.030339	1.273588	1.272653	.163201	1.255808	.162863	1.256195
.045332	1.272872	1.271931	.242764	1.234772	.242277	1.235192
.067338	1.271306	1.270350	.357060	1.188872	.356391	1.189358
.099539	1.267905	1.266916	.516339	1.089760	.515514	1.090367
.136809	1.262312	1.261268	.683508	.928558	.682692	.929295
.165843	1.256716	1.255608	.796285	.769298	.795612	.770084
.190453	1.251107	1.249934	.876560	.611802	.876074	.612566
.212195	1.245494	1.244244	.933329	.455950	.933030	.456628
.231888	1.239878	1.238534	.971248	.301475	.971104	.302016
.250032	1.234236	1.232798	.992965	.146693	.992927	.147109
.266962	1.228571	1.227030	1.000000	.000000	1.000000	.000000
.266962	1.228571	1.227030				
.282921	1.222887	1.221219				
.298102	1.217179	1.215350				
.312674	1.211437	1.209400				
.326806	1.205642	1.203328				
.340710	1.199759	1.197060				
.354735	1.193683	1.190437				
.369734	1.187158	1.183015				
.380660	1.182629	1.177394				
.387316	1.180065	1.173859				
.391619	1.178539	1.171543				
.394502	1.177584	1.169974				
.396516	1.176951	1.168869				
.398040	1.176492	1.168029				
.400000	1.175928	1.166943				
.400000	1.159416	1.166943				
.402721	1.158613	1.165425				
.404841	1.157933	1.164234				
.407653	1.156958	1.162642				
.411705	1.155417	1.160325				
.417835	1.152809	1.156767				
.427556	1.148122	1.150994				
.443803	1.139232	1.140981				
.466040	1.125661	1.126516				
.485930	1.112517	1.112809				
.504433	1.099563	1.099380				
.521914	1.086428	1.086071				
.538569	1.073277	1.072808				
.554524	1.060145	1.059548				
.569879	1.047020	1.046258				
.569879	1.047020	1.046258				
.584715	1.033895	1.032901				
.599111	1.020753	1.019439				
.613154	1.007574	1.005813				
.626950	.994317	.991930				
.640664	.980907	.977621				
.654609	.967181	.962528				

.669611	.952673	.945651
.680552	.942412	.932902
.687257	.936511	.924899
.691580	.933005	.919661
.694476	.930812	.916116
.696500	.929362	.913622
.698031	.928310	.911725
.700000	.927019	.909275
.700000	.889657	.909275
.705266	.885961	.902654
.709427	.882585	.897352
.715069	.877424	.890063
.723568	.868589	.878857
.737397	.852218	.860021
.761486	.819934	.825290
.804519	.753002	.756210
.861464	.644536	.646576
.906241	.536046	.538278
.941135	.428446	.430395
.967351	.321688	.322688
.985630	.210951	.215074
.996428	.093639	.107514
1.000000	.000000	0.000000

ERRORS IN REQUIRED CONDITIONS

TRANSFORMATION CONSTANTS (INDEPENDENT VARIABLES) A	B	C	D	E	F	G	H	I	INBOARD PYLON CLOSURE	OUTBOARD PYLON CLOSURE	WING CLOSURE	INBOARD PYLON LOCATION	OUTBOARD PYLON LOCATION	INBOARD PYLON LENGTH	OUTBOARD PYLON LENGTH	SQUARE ROOT OF SUM OF SQR. ERRORS
-.437	-.117	-.061	0.0	.734	.862	1.0	4.098	8.620	.00000	-.00000	-.00000	.00000	-.00000	-.03000	-.03000	.04242640
-.010	-.002	-.347	-.528	-.104	.044	.015										
.887	-.003	-.347	-.791	-.117	-.299	.019										
.739	-.004	-.348	-.690	-.123	.227	.021										
-.002	.386	-.347	-.152	-.236	-.016	-.124										
-.003	-.711	.696	.263	.635	.027	.006										
-.001	.326	-.350	-.113	-.173	-.011	.103										
.000	.001	.695	.090	-.178	.007	.009										
-.455	-.212	-.113	0.0	.515	.742	1.0	3.726	7.862	-.00333	-.00437	.00015	.00552	.00261	.00113	.00066	.00832162
-.446	-.205	-.111	0.0	.526	.742	1.0	3.693	7.754	.00054	.00073	-.00000	-.00019	.00067	-.00032	-.00033	.00123263
-.447	-.207	-.112	0.0	.522	.741	1.0	3.703	7.796	-.00015	-.00020	-.00000	.00005	-.00019	.00017	.00021	.00042115
-.447	-.206	-.112	0.0	.524	.741	1.0	3.699	7.788	.00006	.00009	.00000	-.00005	.00003	-.00007	-.00008	.00016331
-.447	-.207	-.112	0.0	.523	.741	1.0	3.700	7.791	-.00002	-.00004	.00000	.00002	-.00001	.00002	.00003	.00006080
-.447	-.206	-.112	0.0	.523	.741	1.0	3.700	7.789	.00001	.00001	.00000	-.00001	.00000	-.00001	-.00001	.00002188
-.447	-.206	-.112	0.0	.523	.741	1.0	3.700	7.790	-.00000	-.00000	-.00000	.00000	-.00000	.00000	.00000	.00000795
-.447	-.206	-.112	0.0	.523	.741	1.0	3.700	7.790	.00000	.00000	.00000	-.00000	.00000	-.00000	-.00000	.00000291
-.447	-.206	-.112	0.0	.523	.741	1.0	3.700	7.790	-.00000	-.00000	-.00000	.00000	-.00000	.00000	.00000	.00000106
-.447	-.206	-.112	0.0	.523	.741	1.0	3.700	7.790	.00000	.00000	.00000	-.00000	.00000	-.00000	-.00000	.00000039
-.447	-.206	-.112	0.0	.523	.741	1.0	3.700	7.790	-.00000	-.00000	-.00000	.00000	-.00000	.00000	.00000	.00000014
-.447	-.206	-.112	0.0	.523	.741	1.0	3.700	7.790	.00000	.00000	.00000	-.00000	.00000	-.00000	-.00000	.00000005
-.447	-.206	-.112	0.0	.523	.741	1.0	3.700	7.790	-.00000	-.00000	-.00000	.00000	-.00000	.00000	.00000	.00000002
-.447	-.206	-.112	0.0	.523	.741	1.0	3.700	7.790	-.00000	-.00000	-.00000	.00000	-.00000	.00000	.00000	.00000002

TRANSFORMATION CONSTANTS IN EXPONENTIAL FORMAT

-4.46882043E-01 -2.06419827E-01 -1.11856626E-01 0. 5.23128383E-01 7.41170475E-01 1.0E+00 3.69973055E+00 7.789[illegible]47E+00

CLOSURE INTEGRAL(INBOARD PYLON) = -.00000001
CLOSURE INTEGRAL(OUTBOARD PYLON) = -.00000001
CLOSURE INTEGRAL(WING) = -.00000000

INBOARD PYLON SEMISPAN LOCATION = .40000001
OUTBOARD PYLON SEMISPAN LOCATION = .70000000
INBOARD PYLON LENGTH (SEMISPANS) = .06000001
OUTBOARD PYLON LENGTH (SEMISPANS) = .06000001

INDUCED DRAG INTEGRAL = 26.45405077

DISTANCE FROM A TO B = 1.63843130
DISTANCE FROM B TO C = -.24576473
DISTANCE FROM C TO D = .24576471
DISTANCE FROM D TO E = 1.22882341
DISTANCE FROM E TO F = -.24576475
DISTANCE FROM F TO G = .24576470
DISTANCE FROM G TO H = 1.22882347
DISTANCE FROM H TO I = -4.09607818

LIFTING EFFICIENCY K = 1.00377374

WING WITH TWO PYLONS BELOW

RATIO OF INBOARD PYLON LOAD TO WING LOAD = .00150627
RATIO OF OUTBOARD PYLON LOAD TO WING LOAD = .00347494

WING SPANWISE COORDINATE	WING LOADING	ELLIPTICAL WING LOADING	INBOARD PYLON COORDINATE	INBOARD PYLON LOADING	OUTBOARD PYLON COORDINATE	OUTBOARD PYLON LOADING
0.000000	1.275513	1.273240	0.000000	1.273332	0.000000	1.275103
.006701	1.275486	1.273211	.039601	1.272331	.039429	1.274099
.011897	1.275427	1.273149	.070271	1.270178	.069968	1.271938
.018724	1.275299	1.273016	.110476	1.265523	.110003	1.267265
.028377	1.275020	1.272727	.167025	1.255411	.166322	1.257114
.042406	1.274411	1.272094	.248293	1.233374	.247286	1.234995
.063010	1.273078	1.270709	.364687	1.185413	.363326	1.186866
.093201	1.270183	1.267698	.525858	1.082452	.524234	1.083580
.128242	1.265415	1.262726	.693107	.916841	.691593	.917537
.155647	1.260632	1.257722	.804386	.755282	.803212	.755657
.178980	1.255826	1.252680	.882549	.597319	.881758	.597469
.199700	1.250997	1.247593	.937119	.442599	.936666	.442602
.218581	1.246146	1.242451	.973113	.290507	.972914	.290419
.236101	1.241268	1.237243	.993477	.137769	.993429	.137568
.252587	1.236330	1.231954	1.000000	.000000	1.000000	.000000
.252587	1.236330	1.231954				
.268291	1.231339	1.226560				
.283420	1.226283	1.221031				
.298178	1.221141	1.215320				
.312792	1.215876	1.209351				
.327573	1.210426	1.202990				
.343063	1.204659	1.195970				
.360571	1.198175	1.187591				
.374137	1.193340	1.180768				
.382839	1.190414	1.176238				
.388591	1.188577	1.173176				
.392494	1.187381	1.171068				
.395238	1.186566	1.169571				
.397320	1.185962	1.168427				
.400000	1.185202	1.166943				
.400000	1.153236	1.166943				
.403340	1.152269	1.165078				
.405934	1.151472	1.163616				
.409354	1.150361	1.161673				
.414223	1.148662	1.158871				
.421406	1.145914	1.154665				
.432283	1.141243	1.148128				
.449196	1.132935	1.137554				
.470645	1.120934	1.123408				
.488936	1.109678	1.110672				
.505581	1.098755	1.098525				
.521162	1.087731	1.086656				
.535983	1.076750	1.074905				
.550232	1.065788	1.063170				
.564044	1.054812	1.051370				
.564044	1.054812	1.051370				
.577537	1.043785	1.039427				
.590822	1.032670	1.027252				
.604027	1.021414	1.014726				
.617321	1.009935	1.001672				
.630967	.998095	.987791				
.645462	.985622	.972490				

.662057	.971800	.954236
.675044	.961595	.939368
.683420	.955134	.929497
.688971	.951080	.922830
.692743	.948440	.918241
.695395	.946642	.914986
.697408	.945310	.912498
.700000	.943637	.909275
.700000	.869709	.909275
.707015	.864989	.900433
.712496	.860776	.893402
.719792	.854562	.883870
.730391	.844385	.869660
.746628	.826488	.847012
.772712	.793057	.808197
.815811	.725764	.736339
.870300	.620581	.627099
.912436	.515275	.521034
.945089	.411189	.416112
.969565	.309149	.311731
.986611	.202489	.207652
.996674	.088638	.103756
1.000000	.000000	0.000000

WING WITH TWO PYLONS BELOW (CONFIGURATION 2)

TRANSFORMATION CONSTANTS (INDEPENDENT VARIABLES)									ERRORS IN REQUIRED CONDITIONS							
A	B	C	D	E	F	G	H	I	INBOARD PYLON CLOSURE	OUTBOARD PYLON CLOSURE	WING CLOSURE	INBOARD PYLON LOCATION	OUTBOARD PYLON LOCATION	INBOARD PYLON LENGTH	OUTBOARD PYLON LENGTH	SQUARE ROOT OF SUM OF SQR. ERRORS
-.447	-.206	-.112	0.0	.523	.741	1.0	3.700	7.790	-.00000	-.00000	-.00000	.00000	-.00000	-.02000	-.02000	.02828426
	-.046	-.008	-.384	-.574	-.111	.102	.033									
	1.106	-.013	-.383	-.740	-.121	-.380	.040									
	.784	-.021	-.384	-.562	-.133	.235	.050									
	-.012	.470	-.384	-.185	-.196	-.042	-.154									
	.014	-.795	.767	.281	.537	.058	.011									
	-.004	.333	-.385	-.106	-.092	-.021	.103									
	.001	.003	.767	.098	-.199	.016	.020									
-.450	-.253	-.139	0.0	.407	.672	1.0	3.465	7.290	-.00105	-.00210	.00005	.00190	.00098	.00095	.00047	.00334806
-.445	-.249	-.138	0.0	.413	.672	1.0	3.452	7.256	.00014	.00032	.00000	-.00018	.00008	-.00017	-.00013	.00045487
-.446	-.250	-.138	0.0	.412	.672	1.0	3.457	7.266	-.00002	-.00006	-.00000	.00003	-.00002	.00004	.00004	.00009924
-.446	-.249	-.138	0.0	.412	.672	1.0	3.456	7.265	.00000	.00002	.00000	-.00001	.00000	-.00001	-.00001	.00002503
-.446	-.249	-.138	0.0	.412	.672	1.0	3.456	7.265	-.00000	-.00000	-.00000	.00000	-.00000	.00000	.00000	.00000626
-.446	-.249	-.138	0.0	.412	.672	1.0	3.456	7.265	.00000	.00000	.00000	-.00000	.00000	-.00000	-.00000	.00000153
-.446	-.249	-.138	0.0	.412	.672	1.0	3.456	7.265	-.00000	-.00000	-.00000	.00000	-.00000	.00000	.00000	.00000037
-.446	-.249	-.138	0.0	.412	.672	1.0	3.456	7.265	.00000	.00000	.00000	-.00000	.00000	-.00000	-.00000	.00000009
-.446	-.249	-.138	0.0	.412	.672	1.0	3.456	7.265	-.00000	-.00000	-.00000	.00000	-.00000	.00000	.00000	.00000002
-.446	-.249	-.138	0.0	.412	.672	1.0	3.456	7.265	-.00000	-.00000	-.00000	.00000	-.00000	.00000	.00000	.00000002

TRANSFORMATION CONSTANTS IN EXPONENTIAL FORMAT

-4.45996077E-01 -2.49407813E-01 -1.38018929E-01 0. 4.11875848E-01 6.72073760E-01 1.0E+00 3.45607769E+00 7.26484158E+00

CLOSURE INTEGRAL(INBOARD PYLON) = -.00000000
CLOSURE INTEGRAL(OUTBOARD PYLON) = -.00000002
CLOSURE INTEGRAL(WING) = -.00000000

INBOARD PYLON SEMISPAN LOCATION = .40000001
OUTBOARD PYLON SEMISPAN LOCATION = .70000000
INBOARD PYLON LENGTH (SEMISPANS) = .08000001
OUTBOARD PYLON LENGTH (SEMISPANS) = .08000001

INDUCED DRAG INTEGRAL = 23.06519210

DISTANCE FROM A TO B = 1.52753698
DISTANCE FROM B TO C = -.30550742
DISTANCE FROM C TO D = .30550742
DISTANCE FROM D TO E = 1.14565266
DISTANCE FROM E TO F = -.30550746
DISTANCE FROM F TO G = .30550740
DISTANCE FROM G TO H = 1.14565272
DISTANCE FROM H TO I = -3.81884236

LIFTING EFFICIENCY K = 1.00687077

WING WITH TWO PYLONS BELOW

RATIO OF INBOARD PYLON LOAD TO WING LOAD = .00259514
RATIO OF OUTBOARD PYLON LOAD TO WING LOAD = .00610235

WING SPANWISE COORDINATE	WING LOADING	ELLIPTICAL WING LOADING	INBOARD PYLON COORDINATE	INBOARD PYLON LOADING	OUTBOARD PYLON COORDINATE	OUTBOARD PYLON LOADING
0.000000	1.276869	1.273240	0.000000	1.273164	0.000000	1.276753
.006439	1.276845	1.273213	.040246	1.272133	.040050	1.275705
.011432	1.276792	1.273156	.071413	1.269914	.071066	1.273452
.017993	1.276677	1.273033	.112259	1.265118	.111719	1.268580
.027269	1.276426	1.272766	.169683	1.254702	.168882	1.258004
.040755	1.275879	1.272182	.252123	1.232023	.250982	1.234987
.060571	1.274682	1.270902	.369934	1.182753	.368411	1.185030
.089639	1.272079	1.268114	.532305	1.077403	.530542	1.078420
.123449	1.267786	1.263500	.699454	.909189	.697899	.908727
.149970	1.263470	1.258840	.809620	.746406	.808492	.745062
.172627	1.259125	1.254125	.886335	.588373	.885633	.586620
.192822	1.254746	1.249346	.939464	.434538	.939102	.432744
.211303	1.250332	1.244491	.974243	.284039	.974105	.282459
.228534	1.245879	1.239544	.993780	.132662	.993753	.131181
.244841	1.241364	1.234486	1.000000	.000000	1.000000	.000000
.244841	1.241364	1.234486				
.260474	1.236768	1.229208				
.275652	1.232084	1.223911				
.290591	1.227286	1.218296				
.305547	1.222328	1.212349				
.320877	1.217135	1.205912				
.337211	1.211553	1.198665				
.356065	1.205150	1.189793				
.370982	1.200207	1.182381				
.380681	1.197129	1.177373				
.387137	1.195157	1.173955				
.391532	1.193852	1.171590				
.394625	1.192954	1.169906				
.396974	1.192282	1.168617				
.400000	1.191432	1.166943				
.400000	1.150131	1.166943				
.403530	1.149119	1.164971				
.406271	1.148293	1.163426				
.409879	1.147152	1.161373				
.415002	1.145429	1.158420				
.422516	1.142686	1.154007				
.433769	1.138125	1.147219				
.450917	1.130221	1.136450				
.472101	1.119094	1.122417				
.489799	1.108835	1.110056				
.505718	1.098974	1.098422				
.520534	1.089056	1.087144				
.534599	1.079205	1.076023				
.548132	1.069377	1.064927				
.561296	1.059523	1.053752				
.561296	1.059523	1.053752				
.574226	1.049599	1.042397				
.587052	1.039553	1.030750				
.599924	1.029317	1.018664				
.613042	1.018792	1.005924				
.626711	1.007816	.992175				
.641504	.996067	.976727				

.658842	.982737	.957834
.672727	.972647	.942058
.681816	.966058	.931405
.687884	.961815	.924144
.692021	.959009	.919123
.694935	.957078	.915552
.697149	.955636	.912819
.700000	.953811	.909275
.700000	.856421	.909275
.707798	.851141	.899435
.713875	.846563	.891616
.721925	.839887	.881045
.733509	.829112	.865395
.750939	.810513	.840812
.778153	.776527	.799690
.821575	.709016	.725875
.875020	.605182	.616359
.915830	.502322	.511290
.947291	.400386	.407915
.970812	.300566	.305376
.987168	.197164	.203314
.996815	.085544	.101547
1.000000	.000000	0.000000

```
* * * * * * * * * * * * * * * * *   NEW CASE DATA LOADED BY SUBROUTINE INPUT   * * * * * * * * * * * * * * * * * * * *

                    CONFIGURATION   3

          PROGRAM OPERATION MODE NUMBER =   2

          NUMBER OF INTEGRATION STEPS =     15
```

WING WITH HORIZONTAL TAIL (BIPLANE CASE, CONFIGURATION 3)

TRANSFORMATION CONSTANTS (INDEPENDENT VARIABLES) A	B	C	D	E	F	ERRORS IN REQUIRED CONDITIONS: WING CLOSURE	HORIZONTAL TAIL (2ND WING) CLOSURE	HORIZONTAL TAIL (2ND WING) HEIGHT	HORIZONTAL TAIL (2ND WING) LENGTH	SQUARE ROOT OF SUM OF SQR. ERRORS
-37.01602640	-18.50801320	0.0	.81491987	1.0	1.18508013	-.00426657	.00430986	.03548662	-.00816708	.03691584
.170	-.000	-.018	-.002							
.212	-.298	-.201	-.101							
-.366	.538	.560	.010							
.162	-.248	-.184	.077							
-44.86763921	-22.41774142	0.0	.75135729	1.0	1.28132584	-.00009020	.00007432	.00027981	.00095038	.00099759
-44.94564638	-22.45896293	0.0	.75610625	1.0	1.27595596	.00007623	-.00008078	-.00003596	.00011948	.00016705
-44.95882434	-22.46373475	0.0	.75656421	1.0	1.27508421	.00001767	-.00001865	-.00001840	.00001493	.00003495
-44.95805521	-22.46339270	0.0	.75661447	1.0	1.27494856	.00000299	-.00000317	-.00000458	.00000159	.00000652
-44.95754605	-22.46313911	0.0	.75661945	1.0	1.27492963	.00000042	-.00000045	-.00000090	.00000011	.00000110
-44.95740766	-22.46307072	0.0	.75661976	1.0	1.27492739	.00000005	-.00000005	-.00000015	-.00000001	.00000017
-44.95737946	-22.46305670	0.0	.75661973	1.0	1.27492720	.00000000	-.00000000	-.00000002	-.00000000	.00000002
-44.95737946	-22.46305670	0.0	.75661973	1.0	1.27492720	.00000000	-.00000000	-.00000002	-.00000000	.00000002

TRANSFORMATION CONSTANTS IN EXPONENTIAL FORMAT

-4.49573795E+01	-2.24630567E+01	0.	7.56619729E-01	1.00000000E+00	1.27492720E+00

CLOSURE INTEGRAL (WING) = .00000000
CLOSURE INTEGRAL (HORIZONTAL) = -.00000000

HORIZONTAL LOCATION ABOVE WING = .29999998
HORIZONTAL SEMISPAN / WING B/2 = .04000000
INDUCED DRAG INTEGRAL = 797.75766497

DISTANCE FROM A TO B = 22.49135857
DISTANCE FROM B TO C = -22.49135848
DISTANCE FROM C TO D = 6.74740705
DISTANCE FROM D TO E = .89965418
DISTANCE FROM E TO F = -.89965428

LIFTING EFFICIENCY K = 1.00396767

WING WITH HORIZONTAL TAIL (BIPLANE)

RATIO OF HORIZONTAL TAIL LOAD TO WING LOAD = .00046086

WING SPAN COORDINATE	WING LOCAL LOADING	ELLIPTICAL WING LOADING	HORIZONTAL TAIL SPAN COORDINATE	HORIZONTAL TAIL LOADING
0.00000000	1.27182895	1.27323950	0.00000000	1.27263867
.03779669	1.27093485	1.27232971	.03911469	1.27166640
.06707845	1.26900864	1.27037179	.06941091	1.26957417
.10548406	1.26483542	1.26613609	.10913148	1.26504939
.15956718	1.25573659	1.25692557	.16501774	1.25521681
.23749395	1.23579053	1.23681087	.24539040	1.23377435
.34973647	1.19200763	1.19283203	.36068059	1.18703566
.50705745	1.09676262	1.09742073	.52085018	1.08635444
.67391171	.94014315	.94068448	.68804441	.92335511
.78798911	.78347636	.78391979	.80010259	.76316748
.87028023	.62679004	.62714293	.87937425	.60550362
.92926238	.47009264	.47036065	.93510436	.45015373
.96919869	.31338901	.31357360	.97211851	.29671264
.99239042	.15668262	.15677531	.99320326	.14282546
.99999997	.00000036	0.00000000	.99999988	.00000133

WING WITH HORIZONTAL TAIL (BIPLANE CASE, CONFIGURATION 3)

TRANSFORMATION CONSTANTS (INDEPENDENT VARIABLES) A	B	C	D	E	F	ERRORS IN REQUIRED CONDITIONS WING CLOSURE	HORIZONTAL TAIL (2ND WING) CLOSURE	HORIZONTAL TAIL (2ND WING) HEIGHT	HORIZONTAL TAIL (2ND WING) LENGTH	SQUARE ROOT OF SUM OF SQR. ERRORS
-44.95737946	-22.4630 570	0.0	.75661973	1.0	1.27492720	.00000000	-.00000000	-.00000002	-.04000000	.04000000
.140	-.000	-.013	-.002							
.207	-.279	-.166	-.091							
-.343	.485	.460	.003							
.145	-.216	-.140	.069							
-43.98037347	-21.94343457	0.0	.54183711	1.0	1.55162993	-.00720148	.00727693	.00174112	-.00007280	.01038517
-43.17281056	-21.52609771	0.0	.56012486	1.0	1.56098338	.00073222	-.00071660	.00127787	-.00076170	.00180632
-43.63226374	-21.75614373	0.0	.55120706	1.0	1.56954290	-.00053480	.00053023	-.00002301	.00030058	.00081119
-43.53908123	-21.70879069	0.0	.55449665	1.0	1.56767288	.00018977	-.00018861	.00006188	-.00008607	.00028780
-43.58745639	-21.73324025	0.0	.55332860	1.0	1.56834781	-.00006543	.00006484	-.00002225	.00003689	.00010169
-43.57073739	-21.72479538	0.0	.55377121	1.0	1.56806265	.00002541	-.00002521	.00000694	-.00001310	.00003874
-43.57680097	-21.72786179	0.0	.55360922	1.0	1.56816010	-.00000921	.00000914	-.00000286	.00000482	.00001413
-43.57450995	-21.72670392	0.0	.55366900	1.0	1.56812331	.00000340	-.00000338	.00000102	-.00000180	.00000522
-43.57534597	-21.72712650	0.0	.55364691	1.0	1.56813690	-.00000126	.00000125	-.00000038	.00000066	.00000193
-43.57503672	-21.72697018	0.0	.55365505	1.0	1.56813192	.00000046	-.00000046	.00000014	-.00000024	.00000071
-43.57515091	-21.72702790	0.0	.55365205	1.0	1.56813376	-.00000017	.00000017	-.00000005	.00000009	.00000026
-43.57510881	-21.72700662	0.0	.55365316	1.0	1.56813308	.00000006	-.00000006	.00000002	-.00000003	.00000010
-43.57512435	-21.72701447	0.0	.55365275	1.0	1.56813333	-.00000002	.00000002	-.00000001	.00000001	.00000004
-43.57511862	-21.72701157	0.0	.55365290	1.0	1.56813324	.00000001	-.00000001	.00000000	-.00000000	.00000001
-43.57511862	-21.72701157	0.0	.55365290	1.0	1.56813324	.00000001	-.00000001	.00000000	-.00000000	.00000001

TRANSFORMATION CONSTANTS IN EXPONENTIAL FORMAT

-4.35751186E+01	-2.17270116E+01	0.	5.53652897E-01	1.00000000E+00	1.56813324E+00

CLOSURE INTEGRAL (WING) =	.00000001
CLOSURE INTEGRAL (HORIZONTAL) =	-.00000001
HORIZONTAL LOCATION ABOVE WING =	.30000000
HORIZONTAL SEMISPAN / WING B/2 =	.08000000
INDUCED DRAG INTEGRAL =	753.03873108
DISTANCE FROM A TO B =	21.84717625
DISTANCE FROM B TO C =	-21.84717607
DISTANCE FROM C TO D =	6.55415290
DISTANCE FROM D TO E =	1.74777390
DISTANCE FROM E TO F =	-1.74777409
LIFTING EFFICIENCY K =	1.00440025

WING WITH HORIZONTAL TAIL (BIPLANE)

RATIO OF HORIZONTAL TAIL LOAD TO WING LOAD = .00185959

WING SPAN COORDINATE	WING LOCAL LOADING	ELLIPTICAL WING LOADING	HORIZONTAL TAIL SPAN COORDINATE	HORIZONTAL TAIL LOADING
0.00000000	1.27039035	1.27323950	0.00000000	1.27237149
.03779584	1.26953662	1.27232975	.04080541	1.27131748
.06707695	1.26769214	1.27037192	.07240093	1.26905001
.10548170	1.26368106	1.26613641	.11380090	1.26414872
.15956364	1.25487426	1.25692630	.17197531	1.25350973
.23748876	1.23536910	1.23681248	.25540683	1.23036373
.34972911	1.19207750	1.19283553	.37437551	1.18016501
.50704768	1.09719005	1.09742805	.53762175	1.07321474
.67390078	.94069926	.94069717	.70447828	.90347097
.78797884	.78401385	.78393652	.81360970	.74017873
.87027174	.62724796	.62716204	.88911875	.58239101
.92925633	.47043709	.47038003	.94113014	.42936042
.96919531	.31360086	.31359051	.97502146	.28003389
.99238937	.15675318	.15678612	.99398312	.12963601
.99999996	.00000036	0.00000000	.99999989	.00000119

WING WITH HORIZONTAL TAIL (BIPLANE CASE, CONFIGURATION 3)

TRANSFORMATION CONSTANTS (INDEPENDENT VARIABLES)						ERRORS IN REQUIRED CONDITIONS				
A	B	C	D	E	F	WING CLOSURE	HORIZONTAL TAIL (2ND WING) CLOSURE	HORIZONTAL TAIL (2ND WING) HEIGHT	HORIZONTAL TAIL (2ND WING) LENGTH	SQUARE ROOT OF SUM OF SQR. ERRORS
-43.57511862	-21.72701157	0.0	.55365290	1.0	1.56813324	.00000001	-.00000001	.00000000	-.02000000	.02000000
.144	-.000	-.013	-.003							
.261	-.334	-.121	-.110							
-.352	.498	.363	.004							
.129	-.202	-.092	.064							
-42.69294335	-21.25721990	0.0	.46593626	1.0	1.71294355	-.00170722	.00172719	.00032091	-.00005937	.00245036
-42.57581893	-21.19548280	0.0	.46891577	1.0	1.71650560	.00008476	-.00008163	.00009463	-.00006957	.00016626
-42.60738802	-21.21126915	0.0	.46826387	1.0	1.71719823	-.00002484	.00002425	-.00000181	.00001277	.00003703
-42.60464412	-21.20987023	0.0	.46837658	1.0	1.71713028	.00000417	-.00000409	.00000086	-.00000175	.00000615
-42.60527224	-21.21018889	0.0	.46835838	1.0	1.71714051	-.00000066	.00000065	-.00000016	.00000032	.00000099
-42.60516928	-21.21013674	0.0	.46836151	1.0	1.71713849	.00000012	-.00000011	.00000002	-.00000005	.00000017
-42.60518622	-21.21014533	0.0	.46836098	1.0	1.71713881	-.00000002	.00000002	-.00000000	.00000001	.00000003
-42.60518622	-21.21014533	0.0	.46836098	1.0	1.71713881	-.00000002	.00000002	-.00000000	.00000001	.00000003

TRANSFORMATION CONSTANTS IN EXPONENTIAL FORMAT

-4.26051862E+01	-2.12101453E+01	0.	4.68360980E-01	1.00000000E+00	1.71713881E+00

CLOSURE INTEGRAL (WING) =	-.00000002
CLOSURE INTEGRAL (HORIZONTAL) =	.00000002
HORIZONTAL LOCATION ABOVE WING =	.30000000
HORIZONTAL SEMISPAN / WING B/2 =	.10000001
INDUCED DRAG INTEGRAL =	722.46536693
DISTANCE FROM A TO B =	21.39569924
DISTANCE FROM B TO C =	-21.39569965
DISTANCE FROM C TO D =	6.41870975
DISTANCE FROM D TO E =	2.13957034
DISTANCE FROM E TO F =	-2.13956994
LIFTING EFFICIENCY K =	1.00471801

WING WITH HORIZONTAL TAIL (BIPLANE)

RATIO OF HORIZONTAL TAIL LOAD TO WING LOAD = .00292483

WING SPAN COORDINATE	WING LOCAL LOADING	ELLIPTICAL WING LOADING	HORIZONTAL TAIL SPAN COORDINATE	HORIZONTAL TAIL LOADING
0.00000000	1.26933600	1.27323950	0.00000000	1.27203392
.03779516	1.26850933	1.27232978	.04181606	1.27093019
.06707575	1.26671944	1.27037202	.07418705	1.26855622
.10547983	1.26281873	1.26613666	.11658657	1.26342657
.15956082	1.25421601	1.25692688	.17611435	1.25230062
.23748464	1.23503145	1.23681377	.26132949	1.22813572
.34972326	1.19211455	1.19283831	.38236931	1.17591067
.50703993	1.09751149	1.09743386	.54715576	1.06545880
.67389211	.94112447	.94070724	.71346112	.89230322
.78797070	.78442670	.78394977	.82074126	.72775503
.87026501	.62760027	.62717716	.89410571	.57032380
.92925154	.47070276	.47039535	.94412744	.41885056
.96919264	.31376513	.31360388	.97642799	.27187551
.99238853	.15680914	.15679466	.99435198	.12341934
1.00000000	.00000023	0.00000000	.99999997	.00000033

```
* * * * * * * * * * * * * * * * *   NEW CASE DATA LOADED BY SUBROUTINE INPUT   * * * * * * * * * * * * * * * * * * * *

                    CONFIGURATION   4

          PROGRAM OPERATION MODE NUMBER =   2

         NUMBER OF INTEGRATION STEPS =      15
```

WING WITH ONE PYLON ABOVE AND ONE PYLON BELOW (CONFIGURATION 4)

TRANSFORMATION CONSTANTS (INDEPENDENT VARIABLES)									ERRORS IN REQUIRED CONDITIONS							
A	B	C	D	E	F	G	H	I	INBOARD PYLON CLOSURE	OUTBOARD PYLON CLOSURE	WING CLOSURE	INBOARD PYLON LOCATION	OUTBOARD PYLON LOCATION	INBOARD PYLON LENGTH	OUTBOARD PYLON LENGTH	SQUARE ROOT OF SUM OF SQR. ERRORS
-.051	-.001	-.001	0.0	.562	.999	1.0	1.001	1.175	-.00002	-.00000	.00004	-.00244	-.00135	-.01769	-.01898	.02609700
-.001	-.000	-2.576	-3.948	.873	.025	.001										
5.972	-.000	-2.686	-9.655	.909	-2.189	.001										
-11.817	.000	4.505	24.383	-1.515	-.015	-.002										
-.000	.000	5.124	.673	-1.251	.004	-.002										
-.000	5.240	4.924	.918	-12.944	.005	.013										
.000	-2.653	-2.611	-.489	4.505	-.003	1.176										
.000	.000	-2.553	-.487	2.052	-.003	-.004										
-.059	-.009	-.005	0.0	.566	.991	1.0	1.009	1.191	-.00079	-.00031	-.00071	.00671	.01966	-.00326	-.00669	.02209481
-.058	-.011	-.005	0.0	.562	.988	1.0	1.011	1.182	-.00043	-.00051	.00025	.00241	.00855	-.00029	-.00169	.00907448
-.057	-.011	-.006	0.0	.560	.988	1.0	1.012	1.178	-.00006	-.00019	.00019	.00056	.00256	.00008	-.00018	.00264728
-.057	-.011	-.006	0.0	.559	.987	1.0	1.012	1.177	.00000	-.00003	.00005	.00003	.00040	.00006	.00008	.00042009
-.057	-.011	-.006	0.0	.559	.988	1.0	1.012	1.176	.00001	.00001	.00000	-.00003	-.00006	.00002	.00005	.00009101
-.057	-.011	-.006	0.0	.559	.988	1.0	1.012	1.176	.00000	.00001	-.00000	-.00001	-.00007	.00000	.00002	.00007191
-.032	.000	-2.548	-3.867	.865	.212	.026										
6.212	.000	-2.586	-6.391	.876	-2.063	.026										
-11.651	-.000	5.163	15.952	-1.748	.042	-.052										
.001	-.002	5.098	.673	-1.244	.036	-.045										
.000	5.197	5.112	.962	-7.862	.049	-.001										
-.000	-2.771	-2.540	-.479	3.177	-.024	.947										
-.000	.006	-2.538	-.487	2.030	-.025	-.072										
-.057	-.011	-.006	0.0	.559	.988	1.0	1.012	1.176	-.00000	-.00000	-.00000	-.00000	-.00000	.00000	.00000	.00000163
-.057	-.011	-.006	0.0	.559	.988	1.0	1.012	1.176	-.00000	-.00000	.00000	-.00000	-.00000	.00000	.00000	.00000012
-.057	-.011	-.006	0.0	.559	.988	1.0	1.012	1.176	-.00000	-.00000	.00000	-.00000	-.00000	.00000	.00000	.00000001
-.057	-.011	-.006	0.0	.559	.988	1.0	1.012	1.176	-.00000	-.00000	.00000	-.00000	-.00000	.00000	.00000	.00000001

TRANSFORMATION CONSTANTS IN EXPONENTIAL FORMAT

-5.71962492E-02 -1.07952905E-02 -5.55824724E-03 0. 5.59272707E-01 9.87553152E-01 1.0E+00 1.01176851E+00 1.17662001E+00

CLOSURE INTEGRAL (LOWER PYLON) =	-.00000000
CLOSURE INTEGRAL (UPPER PYLON) =	-.00000000
CLOSURE INTEGRAL (WING) =	.00000000
INBOARD PYLON SEMISPAN LOCATION =	.40000000
OUTBOARD PYLON SEMISPAN LOCATION =	.69999999
INBOARD PYLON LENGTH (SEMISPANS) =	.02000000
OUTBOARD PYLON LENGTH (SEMISPANS) =	.02000001
INDUCED DRAG INTEGRAL =	.59870684
DISTANCE FROM A TO B =	.24684102
DISTANCE FROM B TO C =	-.01234205
DISTANCE FROM C TO D =	.01234205
DISTANCE FROM D TO E =	.37026153
DISTANCE FROM E TO F =	-.18513077
DISTANCE FROM F TO G =	.01234205
DISTANCE FROM G TO H =	-.01234205
DISTANCE FROM H TO I =	-.43197178
LIFTING EFFICIENCY K =	1.00087458

WING WITH PYLON ABOVE AND PYLON BELOW

RATIO OF INBOARD PYLON LOAD TO WING LOAD = .00017509
RATIO OF OUTBOARD PYLON LOAD TO WING LOAD = .00039271

WING SPAN COORDINATE	WING LOADING	WING ELLIPTICAL LOADING	INBOARD PYLON COORDINATE	INBOARD PYLON LOADING	OUTBOARD PYLON COORDINATE	OUTBOARD PYLON LOADING
0.000000	1.273180	1.273240	0.000000	1.272688	0.000000	1.272791
.007343	1.273146	1.273205	.038379	1.271751	.037264	1.271906
.013037	1.273072	1.273131	.068110	1.269732	.066135	1.270001
.020518	1.272913	1.272971	.107097	1.265366	.104007	1.265879
.031092	1.272566	1.272624	.161980	1.255876	.157357	1.256914
.046457	1.271809	1.271865	.240995	1.235161	.234283	1.237320
.069003	1.270153	1.270205	.354607	1.189926	.345256	1.194409
.101982	1.266559	1.266601	.513245	1.092079	.501327	1.100988
.140125	1.260652	1.260678	.680333	.932381	.667900	.946641
.169807	1.254742	1.254749	.793561	.773959	.782714	.791381
.194936	1.248833	1.248814	.874514	.616703	.866227	.635121
.217105	1.242912	1.242871	.932014	.460546	.926598	.477767
.237150	1.236991	1.236918	.970590	.305320	.967836	.319283
.255581	1.231070	1.230952	.992782	.149892	.992002	.159669
.272734	1.225137	1.224970	1.000000	.000000	1.000000	.000001
.272734	1.225137	1.224970				
.288853	1.219189	1.218966				
.304122	1.213231	1.212930				
.318696	1.207254	1.206849				
.332720	1.201234	1.200698				
.346360	1.195158	1.194428				
.359875	1.188976	1.187933				
.373894	1.182516	1.180893				
.383660	1.178184	1.175804				
.389421	1.175825	1.172730				
.393051	1.174468	1.170765				
.395454	1.173642	1.169452				
.397123	1.173105	1.168535				
.398383	1.172723	1.167840				
.400000	1.172260	1.166943				
.400000	1.161118	1.166943				
.407573	1.158585	1.162687				
.413944	1.155876	1.159033				
.423372	1.151199	1.153499				
.439485	1.142143	1.143688				
.469315	1.123385	1.124310				
.525198	1.083189	1.083502				
.622969	.996993	.995987				
.742175	.850260	.853333				
.829525	.709925	.711065				
.894650	.568085	.568842				
.942191	.426185	.426633				
.974729	.283209	.284431				
.993740	.139724	.142239				
1.000000	.000001	0.000000				

WING WITH ONE PYLON ABOVE AND ONE PYLON BELOW (CONFIGURATION 4)

TRANSFORMATION CONSTANTS (INDEPENDENT VARIABLES)									ERRORS IN REQUIRED CONDITIONS							
A	B	C	D	E	F	G	H	I	INBOARD PYLON CLOSURE	OUTBOARD PYLON CLOSURE	WING CLOSURE	INBOARD PYLON LOCATION	OUTBOARD PYLON LOCATION	INBOARD PYLON LENGTH	OUTBOARD PYLON LENGTH	SQUARE ROOT OF SUM OF SQR. ERRORS
-.057	-.011	-.006	0.0	.559	.988	1.0	1.012	1.176	-.00000	-.00000	.00000	-.00000	-.00000	-.02000	-.02000	.02828427
-.032	.000	-2.548	-3.867	.865	.212	.026										
6.211	.000	-2.586	-6.390	.876	-2.063	.026										
-11.650	-.000	5.163	15.951	-1.748	.042	-.052										
.001	-.002	5.097	.673	-1.244	.036	-.045										
.000	5.195	5.112	.962	-7.862	.049	-.001										
-.000	-2.771	-2.540	-.479	3.177	-.024	.947										
-.000	.006	-2.538	-.487	2.030	-.025	-.072										
-.063	-.021	-.011	0.0	.557	.976	1.0	1.022	1.179	-.00144	-.00086	-.00045	.00186	.00361	-.00104	-.00208	.00499172
-.063	-.022	-.011	0.0	.556	.974	1.0	1.023	1.177	-.00018	-.00027	-.00004	.00048	.00069	-.00004	-.00005	.00090106
-.063	-.022	-.011	0.0	.556	.974	1.0	1.023	1.177	-.00001	-.00001	.00000	.00005	.00007	.00002	.00002	.00008990
-.063	-.022	-.011	0.0	.556	.974	1.0	1.023	1.177	.00000	.00000	.00000	-.00000	-.00001	.00000	.00001	.00001073
-.063	-.022	-.011	0.0	.556	.974	1.0	1.023	1.177	.00000	.00000	.00000	-.00000	-.00000	.00000	.00000	.00000310
-.063	-.022	-.011	0.0	.556	.974	1.0	1.023	1.177	.00000	.00000	.00000	-.00000	-.00000	-.00000	-.00000	.00000044
-.063	-.022	-.011	0.0	.556	.974	1.0	1.023	1.177	-.00000	-.00000	-.00000	-.00000	-.00000	-.00000	-.00000	.00000003
-.063	-.022	-.011	0.0	.556	.974	1.0	1.023	1.177	-.00000	-.00000	-.00000	.00000	.00000	-.00000	-.00000	.00000001
-.063	-.022	-.011	0.0	.556	.974	1.0	1.023	1.177	-.00000	-.00000	-.00000	.00000	.00000	-.00000	-.00000	.00000001

TRANSFORMATION CONSTANTS IN EXPONENTIAL FORMAT

-6.33826400E-02 -2.16616954E-02 -1.14740213E-02 0. 5.55738668E-01 9.74392144E-01 1.0E+00 1.02289909E+00 1.17664330E+00

CLOSURE INTEGRAL (LOWER PYLON) = -.00000000
CLOSURE INTEGRAL (UPPER PYLON) = -.00000000
CLOSURE INTEGRAL (WING) = -.00000000

INBOARD PYLON SEMISPAN LOCATION = .40000000
OUTBOARD PYLON SEMISPAN LOCATION = .70000001
INBOARD PYLON LENGTH (SEMISPANS) = .04000000
OUTBOARD PYLON LENGTH (SEMISPANS) = .04000000

INDUCED DRAG INTEGRAL = .60340868

DISTANCE FROM A TO B = .24761459
DISTANCE FROM B TO C = -.02476146
DISTANCE FROM C TO D = .02476146
DISTANCE FROM D TO E = .37142189
DISTANCE FROM E TO F = -.18571094
DISTANCE FROM F TO G = .02476146
DISTANCE FROM G TO H = -.02476146
DISTANCE FROM H TO I = -.43332554

LIFTING EFFICIENCY K = 1.00244185

WING WITH PYLON ABOVE AND PYLON BELOW

RATIO OF INBOARD PYLON LOAD TO WING LOAD = .00069924
RATIO OF OUTBOARD PYLON LOAD TO WING LOAD = .00156518

WING SPAN COORDINATE	WING LOADING	WING ELLIPTICAL LOADING	INBOARD PYLON COORDINATE	INBOARD PYLON LOADING	OUTBOARD PYLON COORDINATE	OUTBOARD PYLON LOADING
0.000000	1.274085	1.273240	0.000000	1.273199	0.000000	1.273700
.007010	1.274054	1.273208	.039003	1.272228	.036725	1.272836
.012446	1.273988	1.273141	.069213	1.270140	.065181	1.270975
.019588	1.273844	1.272995	.108822	1.265623	.102515	1.266950
.029684	1.273533	1.272678	.164558	1.255806	.155123	1.258192
.044355	1.272851	1.271986	.244729	1.234397	.231033	1.239039
.065892	1.271362	1.270472	.359778	1.187726	.340712	1.197044
.097415	1.268127	1.267184	.519754	1.087161	.495491	1.105358
.133926	1.262809	1.261769	.686987	.924262	.661730	.953140
.162395	1.257481	1.256338	.799251	.764082	.777259	.799236
.186555	1.252149	1.250887	.878774	.606353	.862001	.643426
.207927	1.246798	1.245412	.934743	.450877	.923796	.485488
.227316	1.241437	1.239907	.971950	.297267	.966389	.325297
.245216	1.236063	1.234365	.993160	.143226	.991586	.162774
.261960	1.230666	1.228776	1.000000	.000000	1.000000	.000001
.261960	1.230666	1.228776				
.277793	1.225235	1.223126				
.292916	1.219768	1.217393				
.307509	1.214250	1.211545				
.321765	1.208642	1.205528				
.335934	1.202908	1.199246				
.350440	1.196950	1.192497				
.366314	1.190492	1.184738				
.378189	1.185906	1.178674				
.385620	1.183246	1.174764				
.390469	1.181633	1.172165				
.393739	1.180606	1.170390				
.396030	1.179919	1.169136				
.397766	1.179417	1.168180				
.400000	1.178795	1.166943				
.400000	1.156538	1.166943				
.410289	1.153283	1.161138				
.418574	1.150053	1.156334				
.430133	1.144778	1.149437				
.448355	1.135105	1.138093				
.479541	1.115957	1.117292				
.535116	1.076127	1.075606				
.630705	.993267	.988063				
.747207	.836331	.846184				
.832707	.701007	.705011				
.896547	.561370	.563981				
.943202	.421465	.422994				
.975161	.280154	.282018				
.993846	.138172	.141036				
1.000000	.000000	0.000000				

WING WITH ONE PYLON ABOVE AND ONE PYLON BELOW (CONFIGURATION 4)

TRANSFORMATION CONSTANTS (INDEPENDENT VARIABLES)									ERRORS IN REQUIRED CONDITIONS							
A	B	C	D	E	F	G	H	I	INBOARD PYLON CLOSURE	OUTBOARD PYLON CLOSURE	WING CLOSURE	INBOARD PYLON LOCATION	OUTBOARD PYLON LOCATION	INBOARD PYLON LENGTH	OUTBOARD PYLON LENGTH	SQUARE ROOT OF SUM OF SQR. ERRORS
-.063	-.022	-.011	0.0	.556	.974	1.0	1.023	1.177	-.00000	-.00000	-.00000	.00000	.00000	-.02000	-.02000	.02828428
-.130	.001	-2.539	-3.800	.862	.428	.051										
6.601	.001	-2.554	-5.349	.865	-2.217	.052										
-11.634	-.001	5.138	13.368	-1.738	.121	-.104										
.002	-.008	5.081	.668	-1.234	.071	-.090										
.001	5.201	5.089	.957	-6.725	.097	-.019										
-.000	-2.925	-2.532	-.478	2.759	-.048	.982										
-.000	.024	-2.531	-.486	2.014	-.049	-.145										
-.070	-.032	-.017	0.0	.552	.962	1.0	1.033	1.178	-.00148	-.00090	-.00030	.00124	.00233	-.00076	-.00149	.00358824
-.070	-.033	-.018	0.0	.552	.961	1.0	1.033	1.177	-.00012	-.00019	-.00002	.00027	.00036	-.00004	-.00001	.00050275
-.070	-.033	-.018	0.0	.552	.961	1.0	1.033	1.176	-.00001	.00000	.00000	.00002	.00002	.00001	.00001	.00003773
-.070	-.033	-.018	0.0	.552	.961	1.0	1.033	1.176	.00000	.00000	.00000	.00000	-.00000	.00000	.00000	.00000494
-.070	-.033	-.018	0.0	.552	.961	1.0	1.033	1.176	.00000	.00000	.00000	-.00000	-.00000	.00000	.00000	.00000093
-.070	-.033	-.018	0.0	.552	.961	1.0	1.033	1.176	.00000	.00000	-.00000	-.00000	-.00000	-.00000	-.00000	.00000009
-.070	-.033	-.018	0.0	.552	.961	1.0	1.033	1.176	-.00000	-.00000	-.00000	-.00000	.00000	-.00000	-.00000	.00000001
-.070	-.033	-.018	0.0	.552	.961	1.0	1.033	1.176	-.00000	-.00000	-.00000	-.00000	.00000	-.00000	-.00000	.00000001

TRANSFORMATION CONSTANTS IN EXPONENTIAL FORMAT

-6.99011203E-02	-3.25570382E-02	-1.77239684E-02	0.	5.51532658E-01	9.60600609E-01	1.0E+00	1.03333679E+00	1.17647284E+00

CLOSURE INTEGRAL (LOWER PYLON) =	-.00000000
CLOSURE INTEGRAL (UPPER PYLON) =	-.00000000
CLOSURE INTEGRAL (WING) =	-.00000000
INBOARD PYLON SEMISPAN LOCATION =	.40000000
OUTBOARD PYLON SEMISPAN LOCATION =	.70000000
INBOARD PYLON LENGTH (SEMISPANS) =	.06000000
OUTBOARD PYLON LENGTH (SEMISPANS) =	.05999999
INDUCED DRAG INTEGRAL =	.60705792
DISTANCE FROM A TO B =	.24806681
DISTANCE FROM B TO C =	-.03721002
DISTANCE FROM C TO D =	.03721002
DISTANCE FROM D TO E =	.37210022
DISTANCE FROM E TO F =	-.18605011
DISTANCE FROM F TO G =	.03721002
DISTANCE FROM G TO H =	-.03721002
DISTANCE FROM H TO I =	-.43411692
LIFTING EFFICIENCY K =	1.00483072

WING WITH PYLON ABOVE AND PYLON BELOW

RATIO OF INBOARD PYLON LOAD TO WING LOAD = .00156889
RATIO OF OUTBOARD PYLON LOAD TO WING LOAD = .00350172

WING SPAN COORDINATE	WING LOADING	WING ELLIPTICAL LOADING	INBOARD PYLON COORDINATE	INBOARD PYLON LOADING	OUTBOARD PYLON COORDINATE	OUTBOARD PYLON LOADING
0.000000	1.275112	1.273240	0.000000	1.273840	0.000000	1.275029
.006718	1.275085	1.273211	.039672	1.272833	.036182	1.274183
.011928	1.275025	1.273149	.070396	1.270665	.064220	1.272361
.018772	1.274896	1.273015	.110672	1.265977	.101010	1.268420
.028450	1.274616	1.272724	.167318	1.255795	.152869	1.259842
.042514	1.274004	1.272088	.248718	1.233610	.227751	1.241076
.063168	1.272665	1.270697	.365277	1.185346	.336116	1.199885
.093428	1.269757	1.267670	.526603	1.081826	.489563	1.109743
.128538	1.264971	1.262677	.693874	.915581	.655420	.959468
.155985	1.260170	1.257654	.805047	.753686	.771639	.806861
.179346	1.255356	1.252595	.883048	.595638	.857615	.651533
.200081	1.250518	1.247494	.937441	.441049	.920864	.493111
.218968	1.245659	1.242341	.973274	.269224	.964863	.331334
.236485	1.240775	1.237124	.993522	.136712	.991143	.165965
.252962	1.235854	1.231829	1.000000	.000000	1.000000	.000001
.252962	1.235854	1.231829				
.268649	1.230882	1.226433				
.283756	1.225852	1.220905				
.298485	1.220741	1.215198				
.313065	1.215504	1.209236				
.327806	1.210085	1.202887				
.343249	1.204358	1.195883				
.360700	1.197982	1.187527				
.374222	1.193283	1.180725				
.382895	1.190457	1.176208				
.388628	1.188692	1.173156				
.392519	1.187548	1.171054				
.395253	1.186770	1.169562				
.397328	1.186195	1.168422				
.400000	1.185474	1.166943				
.400000	1.152166	1.166943				
.412179	1.148472	1.160052				
.421839	1.144916	1.154409				
.435012	1.139258	1.146456				
.455010	1.129188	1.133802				
.487660	1.109845	1.111581				
.543503	1.070726	1.068767				
.637551	.992762	.980914				
.751736	.820885	.839656				
.835590	.690774	.699458				
.898269	.553842	.559521				
.944119	.416064	.419664				
.975551	.276776	.279822				
.993941	.136486	.139952				
1.000000	.000000	0.000000				

WING WITH ONE PYLON ABOVE AND ONE PYLON BELOW (CONFIGURATION 4)

TRANSFORMATION CONSTANTS (INDEPENDENT VARIABLES)								ERRORS IN REQUIRED CONDITIONS							
A	B	C	E	F	G	H	I	INBOARD PYLON CLOSURE	OUTBOARD PYLON CLOSURE	WING CLOSURE	INBOARD PYLON LOCATION	OUTBOARD PYLON LOCATION	INBOARD PYLON LENGTH	OUTBOARD PYLON LENGTH	SQUARE ROOT OF SUM OF SQR. ERRORS
-.070	-.033	-.018	0.0 .552	.961	1.0	1.033	1.176	-.00000	-.00000	-.00000	-.00000	.00000	-.02000	-.02000	.02828428
-.294	.002	-2.534	-3.725	.859		.655	.077								
7.048	.002	-2.542	-4.778	.860		-2.411	.077								
-11.646	-.003	5.114	11.867	-1.729		.200	-.155								
.005	-.017	5.071	.663	-1.222		.106	-.134								
.001	5.224	5.078	.955	-6.079		.145	-.037								
-.001	-3.094	-2.528	-.478	2.529		-.073	1.039								
-.001	.054	-2.527	-.485	1.997		-.073	-.219								
-.076	-.043	-.024	0.0 .547	.947	1.0	1.043	1.177	-.00151	-.00089	-.00028	.00098	.00181	-.00063	-.00127	.00306562
-.077	-.043	-.024	0.0 .547	.946	1.0	1.043	1.176	-.00009	-.00016	-.00001	.00019	.00024	-.00004	.00000	.00035547
-.077	-.043	-.024	0.0 .547	.946	1.0	1.043	1.176	-.00001	.00000	.00000	.00001	.00001	.00001	.00001	.00002374
-.077	-.043	-.024	0.0 .547	.946	1.0	1.043	1.176	.00000	.00000	.00000	.00000	-.00000	.00000	.00000	.00000306
-.077	-.043	-.024	0.0 .547	.946	1.0	1.043	1.176	.00000	.00000	.00000	-.00000	-.00000	.00000	.00000	.00000043
-.077	-.043	-.024	0.0 .547	.946	1.0	1.043	1.176	.00000	.00000	-.00000	-.00000	-.00000	-.00000	-.00000	.00000004
-.077	-.043	-.024	0.0 .547	.946	1.0	1.043	1.176	-.00000	-.00000	-.00000	-.00000	.00000	-.00000	-.00000	.00000000
-.077	-.043	-.024	0.0 .547	.946	1.0	1.043	1.176	-.00000	-.00000	-.00000	-.00000	.00000	-.00000	-.00000	.00000000

TRANSFORMATION CONSTANTS IN EXPONENTIAL FORMAT

-7.67374115E-02 -4.34520852E-02 -2.42853111E-02 0. 5.46667853E-01 9.46273180E-01 1.0E+00 1.04304325E+00 1.17593097E+00

CLOSURE INTEGRAL (LOWER PYLON) = -.00000000
CLOSURE INTEGRAL (UPPER PYLON) = -.00000000
CLOSURE INTEGRAL (WING) = -.00000000

INBOARD PYLON SEMISPAN LOCATION = .40000000
OUTBOARD PYLON SEMISPAN LOCATION = .70000000
INBOARD PYLON LENGTH (SEMISPANS) = .08000000
OUTBOARD PYLON LENGTH (SEMISPANS) = .08000000

INDUCED DRAG INTEGRAL = .60974337

DISTANCE FROM A TO B = .24821602
DISTANCE FROM B TO C = -.04964320
DISTANCE FROM C TO D = .04964320
DISTANCE FROM D TO E = .37232403
DISTANCE FROM E TO F = -.18616201
DISTANCE FROM F TO G = .04964320
DISTANCE FROM G TO H = -.04964320
DISTANCE FROM H TO I = -.43437803

LIFTING EFFICIENCY K = 1.00806279

WING WITH PYLON ABOVE AND PYLON BELOW

RATIO OF INBOARD PYLON LOAD TO WING LOAD = .00277856
RATIO OF OUTBOARD PYLON LOAD TO WING LOAD = .00617758

WING SPAN COORDINATE	WING LOADING	WING ELLIPTICAL LOADING	INBOARD PYLON COORDINATE	INBOARD PYLON LOADING	OUTBOARD PYLON COORDINATE	OUTBOARD PYLON LOADING
0.000000	1.276238	1.273240	0.000000	1.274606	0.000000	1.276756
.006462	1.276213	1.273213	.040390	1.273557	.035636	1.275926
.011474	1.276159	1.273156	.071668	1.271300	.063253	1.274138
.018059	1.276044	1.273032	.112658	1.266420	.099496	1.270269
.027370	1.275792	1.272763	.170279	1.255827	.150601	1.261848
.040904	1.275243	1.272174	.252985	1.232775	.224447	1.243417
.060789	1.274041	1.270885	.371127	1.182748	.331479	1.202928
.089952	1.271428	1.268078	.533805	1.076031	.483562	1.114150
.123858	1.267123	1.263435	.700986	.906321	.648993	.965635
.150439	1.262798	1.258749	.810932	.742790	.765877	.814253
.173134	1.258453	1.254011	.887319	.584608	.853085	.659421
.193351	1.254079	1.249213	.940096	.431094	.917812	.500604
.211840	1.249673	1.244342	.974559	.281247	.963260	.337363
.229070	1.245232	1.239384	.993869	.130387	.990674	.169227
.245365	1.240743	1.234317	1.000000	.000000	1.000000	.000000
.245365	1.240743	1.234317				
.260978	1.236190	1.229115				
.276127	1.231559	1.223737				
.291031	1.226825	1.218126				
.305942	1.221938	1.212188				
.321218	1.216830	1.205764				
.337488	1.211354	1.198539				
.356261	1.205133	1.189698				
.371113	1.200423	1.182315				
.380768	1.197520	1.177327				
.387194	1.195675	1.173924				
.391570	1.194462	1.171569				
.394649	1.193630	1.169893				
.396988	1.193011	1.168610				
.400000	1.192229	1.166943				
.400000	1.147960	1.166943				
.413623	1.144005	1.159219				
.424347	1.140259	1.152918				
.438799	1.134388	1.144114				
.460277	1.124129	1.130351				
.494293	1.104852	1.106821				
.550657	1.066839	1.062813				
.643633	.994367	.974454				
.755840	.804719	.833662				
.838222	.679474	.694334				
.899844	.545574	.555402				
.944957	.410086	.416597				
.975906	.273083	.277809				
.994026	.134668	.138967				
1.000000	.000000	0.000000				

```
* * * * * * * * * * * * * * * * *   NEW CASE DATA LOADED BY SUBROUTINE INPUT   * * * * * * * * * * * * * * * * * * *

                    CONFIGURATION   5

          PROGRAM OPERATION MODE NUMBER =   2

          NUMBER OF INTEGRATION STEPS =      15
```

WING WITH BREAK STATION, PYLON AT THE BREAK, AND DIHEDRAL OUTBOARD OF THE BREAK (CONFIGURATION 5)

ERRORS IN REQUIRED CONDITIONS

TRANSFORMATION CONSTANTS (INDEPENDENT VARIABLES)							INBOARD WING CLOSURE	PYLON CLOSURE	OUTBOARD WING CLOSURE	PYLON LENGTH	PYLON SPANWISE LOCATION	SQUARE ROOT OF SUM OF SQR. ERRORS
A	B	C	D	E	F	G						
-.36213274	0.0	.01362133	.02724265	1.0	1.97275735	2.36213274	-.08466068	-.00125752	.01602610	-.02127045	-.05900912	.10658498
-1.792	-.003	-.222	.009	-1.268								
-2.806	1.573	-1.140	.163	-1.935								
-2.580	1.435	-1.047	.418	-1.827								
1.609	.000	-1.606	.019	1.256								
-1.808	.000	-.237	-.010	-.708								
-.35302257	0.0	.03993627	.08331506	1.0	1.91961737	2.29104550	.00173555	-.00223493	.00520091	-.00122110	.00470123	.00765827
-.34538884	0.0	.03953817	.08404900	1.0	1.91574465	2.27992660	.00092384	-.00013223	.00102121	.00042374	.00091346	.00170836
-.34473405	0.0	.03882769	.08265264	1.0	1.91549840	2.27912316	-.00009360	.00011295	-.00012686	.00009289	-.00010558	.00023907
-.34483420	0.0	.03879402	.08250329	1.0	1.91559285	2.27931210	-.00005439	.00001355	-.00005554	-.00000427	-.00004157	.00009170
-.34487374	0.0	.03881157	.08253124	1.0	1.91562650	2.27936158	-.00000122	-.00000207	-.00000144	-.00000328	.00000037	.00000433
-.34487263	0.0	.03881441	.08253843	1.0	1.91562632	2.27935921	.00000151	-.00000061	.00000167	-.00000013	.00000124	.00000264
-1.778	-.008	-.208	.029	-1.369								
-2.298	1.673	-.733	.196	-1.777								
-1.962	1.358	-.435	.830	-1.515								
1.534	.003	-1.583	.070	1.442								
-1.804	-.000	-.233	-.036	-.823								
-.34487145	0.0	.03881396	.08253809	1.0	1.91562529	2.27935758	-.00000000	-.00000000	-.00000002	.00000000	-.00000001	.00000002
-.34487145	0.0	.03881396	.08253809	1.0	1.91562529	2.27935758	-.00000000	-.00000000	-.00000002	.00000000	-.00000001	.00000002

TRANSFORMATION CONSTANTS IN EXPONENTIAL FORMAT

-3.44871448E-01	0.	3.88139590E-02	8.25380874E-02	1.00000000E+00	1.91562529E+00	2.27935758E+00

CLOSURE INTEGRAL (INBOARD WING) =	-.00000000
CLOSURE INTEGRAL (PYLON) =	-.00000000
CLOSURE INTEGRAL (OUTBOARD WING) =	-.00000002
PYLON LENGTH (SEMISPANS) =	.03000000
PYLON LOCATION (SEMISPANS) =	.69999999
INDUCED DRAG INTEGRAL =	2.70162598
DISTANCE FROM A TO B =	.91757011
DISTANCE FROM B TO C =	-.03932444
DISTANCE FROM C TO D =	.03932444
DISTANCE FROM D TO E =	-.39378405
REAL PART OF D TO E =	.02060906
IMAJ. PART OF D TO E =	.39324438
DISTANCE FROM E TO F =	.39378403
REAL PART OF E TO F =	-.02060906
IMAJ. PART OF E TO F =	-.39324436
DISTANCE FROM F TO G =	-.91757011
OUTBOARD DIHEDRAL (DEGREES) =	3.00000000

LIFTING EFFICIENCY K = 1.00097418

WING WITH PYLON AND DIHEDRAL OUTBOARD OF PYLON

RATIO OF PYLON LOAD TO WING LIFT LOAD = .0009397

RATIO OF OUTBOARD WING PANEL NORMAL LOAD TO WING LIFT LOAD = .1875096

WING SPAN COORDINATE	WING LIFT LOADING	OUTBOARD WING NORMAL LOADING	ELLIPTICAL WING LOADING	PYLON COORDINATE	PYLON LOADING
0.0000000	1.2729814		1.2732395	0.0000000	1.2782603
.0136640	1.2728631		1.2731206	.0387159	1.2771734
.0242588	1.2726084		1.2728648	.0687056	1.2749063
.0381759	1.2720573		1.2723114	.1080297	1.2700943
.0578409	1.2708591		1.2711079	.1633750	1.2597981
.0863891	1.2682422		1.2684795	.2430223	1.2376554
.1282057	1.2625202		1.2627322	.3574342	1.1899879
.1891252	1.2501049		1.2502613	.5168565	1.0883780
.2590571	1.2297144		1.2297737	.6841166	.9250020
.3129498	1.2093355		1.2092842	.7968768	.7648333
.3581232	1.1889664		1.1887911	.8770589	.6071952
.3975660	1.1686052		1.1682909	.9336861	.4517339
.4328549	1.1482507		1.1477790	.9714447	.2979875
.4649464	1.1279010		1.1272484	.9930256	.1436096
.4944766	1.1075528		1.1066882	.9999999	.0000003
.4944766	1.1075528		1.1066882		
.5218997	1.0872048		1.0860821		
.5475595	1.0668335		1.0654045		
.5717389	1.0464344		1.0446111		
.5946943	1.0259698		1.0236233		
.6167057	1.0053721		1.0022862		
.6381859	.9844817		.9802445		
.6600995	.9628120		.9564298		
.6751441	.9484082		.9392508		
.6839415	.9406477		.9288751		
.6894612	.9362304		.9222361		
.6931089	.9335713		.9177929		
.6956401	.9318642		.9146832		
.6975496	.9306622		.9123225		
.7000000	.9292464		.9092749		
.7000000	.8892082	.8904285	.9092749		
.7047783	.8861881	.8874043	.9032713		
.7087145	.8832017	.8844138	.8982646		
.7141312	.87[illegible]833	.8796889	.8912828		
.7223741	.8701968	.8713910	.8804483		
.7358827	.8545383	.8557111	.8621220		
.7595442	.8230326	.8241621	.8281858		
.8021135	.7571164	.7581554	.7603425		
.8589872	.6498874	.6507793	.6518949		
.9041108	.5426035	.5433481	.5440313		
.9395474	.4349253	.4355222	.4359812		
.9663365	.3268130	.3272615	.3275808		
.9851254	.2182708	.2185703	.2187897		
.9962888	.1093027	.1094527	.1095920		
1.0000000	.0000005	.0000005	0.0000000		

WING WITH BREAK STATION, PYLON AT THE BREAK, AND DIHEDRAL OUTBOARD OF THE BREAK (CONFIGURATION 5)

TRANSFORMATION CONSTANTS (INDEPENDENT VARIABLES)							ERRORS IN REQUIRED CONDITIONS					
A	B	C	D	E	F	G	INBOARD WING CLOSURE	PYLON CLOSURE	OUTBOARD WING CLOSURE	PYLON LENGTH	PYLON SPANWISE LOCATION	SQUARE ROOT OF SUM OF SQR. ERRORS
-.34487145	0.0	.03881396	.08253809	1.0	1.91562529	2.27935758	-.06637762	-.00176958	.02979239	-.03390611	-.04742572	.09324983
-1.760	-.010	-.191	.025	-1.296								
-2.308	1.578	-.651	.171	-1.660								
-1.841	1.244	-.368	.733	-1.359								
1.606	.002	-1.612	.060	1.339								
-1.821	.000	-.252	-.031	-.749								
-.29426775	0.0	.07533103	.16695540	1.0	1.83483594	2.17018893	.00227471	-.00570762	.00724299	-.00056203	.00594797	.01122080
-.29114129	0.0	.07080717	.16134128	1.0	1.83272976	2.16014592	.00063564	.00085290	.00044949	.00105704	.00033449	.00160086
-.29096486	0.0	.07006131	.15897189	1.0	1.83303645	2.16076226	-.00041503	.00023767	-.00046707	.00002699	-.00040307	.00078108
-.29123685	0.0	.07025553	.15921809	1.0	1.83325065	2.16121180	-.00001701	-.00003490	-.00001384	-.00000896	.00001625	.00006603
-.29118909	0.0	.07028911	.15932241	1.0	1.83322859	2.16113470	.00001711	-.00001172	.00001980	.00000271	.00001535	.00003263
-.29118998	0.0	.07027534	.15930069	1.0	1.83322597	2.16112643	.00000009	.00000321	-.00000062	.00000180	-.00000141	.00000399
-.29118975	0.0	.07027557	.15929864	1.0	1.83322607	2.16112904	-.00000086	.00000007	-.00000084	-.00000021	-.00000065	.00000138
-.29119027	0.0	.07027598	.15929950	1.0	1.83322641	2.16112954	.00000008	-.00000009	.00000009	-.00000007	.00000013	.00000021
-.29119011	0.0	.07027598	.15929958	1.0	1.83322632	2.16112931	.00000003	-.00000001	.00000003	.00000002	.00000001	.00000004
-.29119013	0.0	.07027596	.15929952	1.0	1.83322633	2.16112933	-.00000000	.00000001	-.00000001	.00000000	-.00000001	.00000001
-.29119013	0.0	.07027596	.15929952	1.0	1.83322633	2.16112933	-.00000000	.00000001	-.00000001	.00000000	-.00000001	.00000001

TRANSFORMATION CONSTANTS IN EXPONENTIAL FORMAT

-2.91190132E-01	0.	7.02759564E-02	1.59299524E-01	1.00000000E+00	1.83322633E+00	2.16112933E+00

CLOSURE INTEGRAL (INBOARD WING) =	-.00000000
CLOSURE INTEGRAL (PYLON) =	.00000001
CLOSURE INTEGRAL (OUTBOARD WING) =	-.00000001
PYLON LENGTH (SEMISPANS) =	.06000000
PYLON LOCATION (SEMISPANS) =	.69999999
INDUCED DRAG INTEGRAL =	2.35040969
DISTANCE FROM A TO B =	.85396740
DISTANCE FROM B TO C =	-.07319720
DISTANCE FROM C TO D =	.07319721
DISTANCE FROM D TO E =	-.36800199
REAL PART OF D TO E =	.03846668
IMAJ. PART OF D TO E =	.36598604
DISTANCE FROM E TO F =	.36800199
REAL PART OF E TO F =	-.03846668
IMAJ. PART OF E TO F =	-.36598603
DISTANCE FROM F TO G =	-.85396741
OUTBOARD DIHEDRAL (DEGREES) =	6.00000000
LIFTING EFFICIENCY K =	1.00539605

WING WITH PYLON AND DIHEDRAL OUTBOARD OF PYLON

RATIO OF PYLON LOAD TO WING LIFT LOAD = .0038553

RATIO OF OUTBOARD WING PANEL NORMAL LOAD TO WING LIFT LOAD = .1859928

WING SPAN COORDINATE	WING LIFT LOADING	OUTBOARD WING NORMAL LOADING	ELLIPTICAL WING LOADING	PYLON COORDINATE	PYLON LOADING
0.0000000	1.2725624		1.2732395	0.0000000	1.2864847
.0130640	1.2724559		1.2731308	.0396002	1.2851462
.0231937	1.2722266		1.2728970	.0702705	1.2824593
.0365004	1.2717305		1.2723911	.1104769	1.2768869
.0553045	1.2706517		1.2712909	.1670321	1.2651936
.0826077	1.2682957		1.2688877	.2483211	1.2405075
.1226168	1.2631438		1.2636317	.3647796	1.1883608
.1809557	1.2519636		1.2522199	.5261176	1.0795726
.2480436	1.2335956		1.2334493	.6936104	.9090091
.2998734	1.2152317		1.2146437	.8050156	.7454625
.3434405	1.1968655		1.1957940	.8831682	.5872982
.3816048	1.1784906		1.1768878	.9376086	.4335640
.4158803	1.1601014		1.1579087	.9734033	.2831618
.4471919	1.1416893		1.1378338	.9935701	.1313511
.4761639	1.1232406		1.1196315	1.0000000	.0000011
.4761639	1.1232406		1.1196315		
.5032532	1.1047389		1.1002559		
.5288224	1.0861611		1.0806398		
.5531933	1.0674526		1.0606759		
.5766930	1.0485483		1.0401870		
.5997296	1.0293078		1.0188497		
.6229706	1.0094403		.9959855		
.6480032	.9881518		.9697471		
.6664717	.9732637		.9492387		
.6779371	.9647792		.9359837		
.6853895	.9597354		.9271432		
.6904056	.9565838		.9210901		
.6939176	.9545206		.9168018		
.6965781	.9530428		.9135251		
.7000000	.9513014		.9092749		
.7000000	.8686374	.8734221	.9092749		
.7056836	.8654293	.8701963	.9021246		
.7105160	.8621431	.8668921	.8959545		
.7171577	.8569824	.8617029	.8873345		
.7270567	.8480648	.8527362	.8741774		
.7425481	.8316832	.8362643	.8528066		
.7678769	.7998233	.8042290	.8156340		
.8104103	.7350636	.7391125	.7459437		
.8651300	.6312571	.6347342	.6385902		
.9081451	.5276725	.5305790	.5330487		
.9419319	.4235619	.4258950	.4275587		
.9675666	.3187691	.3205250	.3216408		
.9856236	.2132607	.2144354	.2151211		
.9964018	.1070162	.1076057	.1079133		
1.0000000	.0000005	.0000005	0.0000000		

WING WITH BREAK STATION, PYLON AT THE BREAK, AND DIHEDRAL OUTBOARD OF THE BREAK (CONFIGURATION 5)

TRANSFORMATION CONSTANTS (INDEPENDENT VARIABLES)							ERRORS IN REQUIRED CONDITIONS					
A	B	C	D	E	F	G	INBOARD WING CLOSURE	PYLON CLOSURE	OUTBOARD WING CLOSURE	PYLON LENGTH	PYLON SPANWISE LOCATION	SQUARE ROOT OF SUM OF SQR. ERRORS
-.29119013	0.0	.07027596	.15929952	1.0	1.83322633	2.16112933	-.05928537	-.00313121	.02872641	-.02700526	-.04494364	.08425545
-1.741	-.036	-.172	.049	-1.378								
-2.057	1.716	-.408	.181	-1.589								
-1.401	1.107	.017	.686	-1.113								
1.597	.010	-1.626	.133	1.438								
-1.832	-.000	-.264	-.067	-.807								
-.25862677	0.0	.09012935	.21321804	1.0	1.76834535	2.07251939	-.00014781	-.00252078	.00330059	.00051913	.00336900	.00537492
-.25676297	0.0	.08679517	.20714248	1.0	1.76843206	2.06716049	-.00015288	.00053458	-.00035253	.00053443	-.00019405	.00086988
-.25717184	0.0	.08666792	.20632656	1.0	1.76905666	2.06809322	-.00007073	.00004730	-.00010193	-.00005655	-.00007537	.00016281
-.25712733	0.0	.08677339	.20655140	1.0	1.76902531	2.06808382	.00000843	-.00001993	.00001584	-.00000513	.00001515	.00003123
-.25712524	0.0	.08676200	.20654070	1.0	1.76902049	2.06806415	.00000076	.00000164	.00000027	.00000204	-.00000009	.00000274
-.25712639	0.0	.08676107	.20653685	1.0	1.76902188	2.06806703	-.00000031	.00000024	-.00000041	-.00000015	-.00000037	.00000069
-.25712632	0.0	.08676146	.20653762	1.0	1.76902181	2.06806714	.00000002	-.00000007	.00000005	-.00000003	.00000005	.00000011
-.25712630	0.0	.08676144	.20653761	1.0	1.76902178	2.06806705	.00000000	.00000000	.00000000	.00000001	.00000000	.00000001
-.25712630	0.0	.08676144	.20653761	1.0	1.76902178	2.06806705	.00000000	.00000000	.00000000	.00000001	.00000000	.00000001

TRANSFORMATION CONSTANTS IN EXPONENTIAL FORMAT

-2.57126298E-01	0.	8.67614360E-02	2.06537614E-01	1.00000000E+00	1.76902178E+00	2.06806705E+00

CLOSURE INTEGRAL (INBOARD WING) = .00000000
CLOSURE INTEGRAL (PYLON) = .00000000
CLOSURE INTEGRAL (OUTBOARD WING) = .00000000

PYLON LENGTH (SEMISPANS) = .08000001
PYLON LOCATION (SEMISPANS) = .70000000

INDUCED DRAG INTEGRAL = 2.10394395

DISTANCE FROM A TO B = .80574015
DISTANCE FROM B TO C = -.09208459
DISTANCE FROM C TO D = .09208460
DISTANCE FROM D TO E = -.34962163
REAL PART OF D TO E = .05469287
IMAJ. PART OF D TO E = .34531720
DISTANCE FROM E TO F = .34962163
REAL PART OF E TO F = -.05469287
IMAJ. PART OF E TO F = -.34531721
DISTANCE FROM F TO G = -.80574015

OUTBOARD DIHEDRAL (DEGREES) = 9.00000000

LIFTING EFFICIENCY K = 1.01092831

WING WITH PYLON AND DIHEDRAL OUTBOARD OF PYLON

RATIO OF PYLON LOAD TO WING LIFT LOAD = .0069780

RATIO OF OUTBOARD WING PANEL NORMAL LOAD TO WING LIFT LOAD = .1852938

WING SPAN COORDINATE	WING LIFT LOADING	OUTBOARD WING NORMAL LOADING	ELLIPTICAL WING LOADING	PYLON COORDINATE	PYLON LOADING
0.0000000	1.2718021		1.2732395	0.0000000	1.2951420
.0126996	1.2717032		1.2731368	.0401966	1.2935027
.0225469	1.2714904		1.2729158	.0713257	1.2903508
.0354830	1.2710300		1.2724377	.1121265	1.2839728
.0537645	1.2700288		1.2713979	.1694954	1.2708625
.0803127	1.2678422		1.2691266	.2518838	1.2437162
.1192275	1.2630605		1.2641574	.3696998	1.1874733
.1760104	1.2526823		1.2533621	.5322706	1.0725716
.2413989	1.2356273		1.2355847	.6998362	.8964506
.2920133	1.2185696		1.2177444	.8102856	.7306716
.3346514	1.2015005		1.1998271	.8870749	.5725444
.3720952	1.1844114		1.1818140	.9400862	.4204742
.4058207	1.1672936		1.1636807	.9746251	.2727775
.4367340	1.1501330		1.1453944	.9939056	.1229119
.4654523	1.1329167		1.1269099	1.0000000	.0000003
.4654523	1.1329167		1.1269099		
.4924342	1.1156199		1.1081636		
.5180528	1.0982091		1.0890637		
.5426514	1.0806205		1.0694677		
.5665949	1.0627693		1.0491449		
.5903576	1.0444889		1.0276848		
.6147344	1.0254416		1.0042475		
.6416161	1.0047356		.9766077		
.6619768	.9899689		.9543240		
.6748549	.9814019		.9395877		
.6833092	.9762491		.9296292		
.6890275	.9730031		.9227615		
.6930403	.9708790		.9178768		
.6960836	.9693577		.9141360		
.7000000	.9675950		.9092749		
.7000000	.8546260	.8652790	.9092749		
.7056297	.8517502	.8623674	.9021931		
.7106053	.8486207	.8591988	.8958397		
.7175285	.8435995	.8541151	.8868484		
.7279006	.8348187	.8452248	.8730383		
.7440711	.8186200	.8288242	.8506519		
.7701146	.7871749	.7969872	.8122066		
.8128576	.7236087	.7326286	.7416142		
.8669221	.6220929	.6298473	.6346381		
.9092167	.5207495	.5272407	.5300790		
.9424708	.4186732	.4238920	.4256293		
.9677781	.3156565	.3195912	.3206076		
.9856745	.2114013	.2140364	.2147429		
.9964033	.1060459	.1073678	.1078915		
1.0000000	.0000005	.0000005	0.0000000		

WING WITH BREAK STATION, PYLON AT THE BREAK, AND DIHEDRAL OUTBOARD OF THE BREAK (CONFIGURATION 5)

TRANSFORMATION CONSTANTS (INDEPENDENT VARIABLES)							ERRORS IN REQUIRED CONDITIONS					
A	B	C	D	E	F	G	INBOARD WING CLOSURE	PYLON CLOSURE	OUTBOARD WING CLOSURE	PYLON LENGTH	PYLON SPANWISE LOCATION	SQUARE ROOT OF SUM OF SQR. ERRORS
-.25712630	0.0	.08676144	.20653761	1.0	1.76902178	2.06806705	-.01817546	-.00131253	.00923715	-.00305933	-.01483778	.02543451
-1.737	-.062	-.169	.070	-1.500								
-1.942	1.861	-.353	.201	-1.668								
-1.236	1.059	.166	.701	-1.075								
1.545	.018	-1.625	.206	1.593								
-1.832	-.002	-.265	-.105	-.908								
-.25586407	0.0	.08671160	.20768231	1.0	1.75667055	2.04971814	-.00004422	-.00000019	-.00001009	.00003437	.00011068	.00012445
-.25577120	0.0	.08664731	.20751335	1.0	1.75675680	2.04961464	-.00000094	.00000025	-.00000271	.00000042	-.00000038	.00000293
-.25577290	0.0	.08664760	.20751381	1.0	1.75675921	2.04961733	.00000002	-.00000000	.00000002	-.00000002	.00000002	.00000004
-.25577286	0.0	.08664761	.20751384	1.0	1.75675918	2.04961729	-.00000000	.00000000	.00000000	.00000000	-.00000000	.00000000
-.25577286	0.0	.08664761	.20751384	1.0	1.75675918	2.04961729	-.00000000	.00000000	.00000000	.00000000	-.00000000	.00000000

TRANSFORMATION CONSTANTS IN EXPONENTIAL FORMAT

-2.55772855E-01	0.	8.66476072E-02	2.07513837E-01	1.00000000E+00	1.75675918E+00	2.04961729E+00

CLOSURE INTEGRAL (INBOARD WING) =	-.00000000
CLOSURE INTEGRAL (PYLON) =	.00000000
CLOSURE INTEGRAL (OUTBOARD WING) =	.00000000
PYLON LENGTH (SEMISPANS) =	.08000000
PYLON LOCATION (SEMISPANS) =	.70000000
INDUCED DRAG INTEGRAL =	2.06720954
DISTANCE FROM A TO B =	.79828939
DISTANCE FROM B TO C =	-.09123307
DISTANCE FROM C TO D =	.09123307
DISTANCE FROM D TO E =	-.34740184
REAL PART OF D TO E =	.06032569
IMAJ. PART OF D TO E =	.34212402
DISTANCE FROM E TO F =	.34740184
REAL PART OF E TO F =	-.06032569
IMAJ. PART OF E TO F =	-.34212402
DISTANCE FROM F TO G =	-.79828939
OUTBOARD DIHEDRAL (DEGREES) =	10.00000000
LIFTING EFFICIENCY K =	1.01190558

WING WITH PYLON AND DIHEDRAL OUTBOARD OF PYLON

RATIO OF PYLON LOAD TO WING LIFT LOAD = .0070583

RATIO OF OUTBOARD WING PANEL NORMAL LOAD TO WING LIFT LOAD = .1858045

WING SPAN COORDINATE	WING LIFT LOADING	OUTBOARD WING NORMAL LOADING	ELLIPTICAL WING LOADING	PYLON COORDINATE	PYLON LOADING
0.0000000	1.2715059		1.2732395	0.0000000	1.2972177
.0127092	1.2714071		1.2731367	.0402287	1.2954910
.0225639	1.2711945		1.2729153	.0713826	1.2922179
.0355098	1.2707345		1.2724365	.1122154	1.2856450
.0538051	1.2697341		1.2713952	.1696283	1.2722189
.0803729	1.2675494		1.2691204	.2520767	1.2445756
.1193157	1.2627719		1.2641439	.3699678	1.1876000
.1761368	1.2524037		1.2533333	.5326105	1.0[illegible]17609
.2415636	1.2353682		1.2355325	.7001881	.8949428
.2922018	1.2183335		1.2176711	.8105904	.7289592
.3348548	1.2012914		1.1997351	.8873062	.5708999
.3723070	1.1842340		1.1817059	.9402364	.4[illegible]90611
.4060355	1.1671532		1.1635592	.9747010	.2716779
.4369474	1.1500356		1.1452624	.9939269	.1[illegible]19856
.4656602	1.1328681		1.1267706	1.0000000	.0000000
.4656602	1.1328681		1.1267706		
.4926329	1.1156273		1.1080204		
.5182390	1.0982807		1.0889201		
.5428217	1.0807673		1.0693276		
.5667459	1.0630040		1.0490127		
.5904860	1.0448291		1.0275652		
.6148364	1.0259143		1.0041463		
.6416864	1.0053923		.9765329		
.6620223	.9908020		.9542728		
.6748848	.9823667		.9395528		
.6833290	.9773166		.9296056		
.6890405	.9741464		.9227458		
.6930486	.9720802		.9178667		
.6960882	.9706072		.9141303		
.7000000	.9689161		.9092749		
.7000000	.8544648	.8676463	.9092749		
.7054264	.8517734	.8649134	.9024508		
.7102875	.8487729	.8618665	.8962482		
.7170885	.8439007	.8569192	.8874252		
.7273241	.8353102	.8481962	.8738169		
.7433468	.8193625	.8320025	.8516778		
.7692471	.7882349	.8003947	.8135382		
.8118977	.7250725	.7362579	.7433168		
.8660158	.6239006	.6335252	.6366409		
.9084672	.5226766	.5307398	.5321582		
.9419182	.4205526	.4270403	.4276078		
.9674263	.3171309	.3220231	.3223244		
.9854992	.2124758	.2157536	.2160431		
.9963545	.1065677	.1082117	.1086200		
1.0000000	.0000000	.0000000	0.0000000		

```
* * * * * * * * * * * * * * * * *   NEW CASE DATA LOADED BY SUBROUTINE INPUT   * * * * * * * * * * * * * * * * * * * *

                    CONFIGURATION   6

          PROGRAM OPERATION MODE NUMBER =   2

          NUMBER OF INTEGRATION STEPS =      15
```

WING WITH TWO BREAK STATIONS, PYLON AT THE INBOARD STATION, AND DIHEDRAL OUTBOARD OF THE OUTBOARD STATION (CONFIGURATION 6)

TRANSFORMATION CONSTANTS (INDEPENDENT VARIABLES)								ERRORS IN REQUIRED CONDITIONS						
A	B	C	D	E	F	G	H	INBOARD WING CLOSURE	PYLON CLOSURE	OUTBOARD WING CLOSURE	PYLON LOC.	DIHEDRAL BREAK LOC.	PYLON LENGTH	SQUARE ROOT OF SUM OF SQR. ERRORS
-.28216	0.0	.01282	.02564	.15193	1.0	1.84807	2.28216	-.06657	-.00042	-.02212	-.04098	.01285	-.01867	.08434307
-1.718	-.002	.147	-1.509	-.101	.015									
-1.978	1.890	.326	-2.423	-.224	.225									
-1.974	1.790	.400	-2.359	-.254	1.116									
.906	-.002	-.854	.436	.499	.008									
1.749	.001	1.456	.595	-.574	.013									
.853	-.000	-.907	.396	-.153	.007									
-.27320	0.0	.03106	.06335	.16031	1.0	1.80793	2.21874	-.00273	-.00139	.00013	.00155	-.00060	-.00127	.00371000
-.27041	0.0	.03130	.06464	.16458	1.0	1.80401	2.21387	-.00009	-.00015	-.00006	.00048	.00002	.00009	.00052152
-.27012	0.0	.03109	.06430	.16482	1.0	1.80370	2.21342	.00002	.00002	-.00002	.00001	.00001	.00003	.00005217
-.27011	0.0	.03107	.06424	.16479	1.0	1.80370	2.21343	.00000	.00000	-.00000	-.00001	.00000	.00000	.00001093
-.27012	0.0	.03107	.06424	.16478	1.0	1.80371	2.21344	.00000	.00000	.00000	-.00000	-.00000	-.00000	.00000171
-.27012	0.0	.03107	.06424	.16478	1.0	1.80371	2.21344	-.00000	-.00000	.00000	.00000	-.00000	-.00000	.00000016
-.27012	0.0	.03107	.06424	.16478	1.0	1.80371	2.21344	-.00000	-.00000	.00000	.00000	-.00000	-.00000	.00000002
-.27012	0.0	.03107	.06424	.16478	1.0	1.80371	2.21344	-.00000	-.00000	.00000	.00000	-.00000	-.00000	.00000003

TRANSFORMATION CONSTANTS IN EXPONENTIAL FORMAT

-2.70123016E-01 0. 3.10735388E-02 6.42391229E-02 1.64776171E-01 1.00000000E+00 1.80370764E+00 2.21344129E+00

CLOSURE INTEGRAL (INBOARD WING) =	-.00000001
CLOSURE INTEGRAL (PYLON) =	-.00000001
CLOSURE INTEGRAL (OUTBOARD WING) =	.00000000
PYLON LOCATION (SEMISPAN) =	.65000003
DIHEDRAL BREAK LOCATION (SEMISPAN) =	.75000000
PYLON LENGTH (SEMISPANS) =	.03000000
INDUCED DRAG INTEGRAL =	2.42083626
DISTANCE FROM A TO B =	.80651157
DISTANCE FROM B TO C =	-.03722362
DISTANCE FROM C TO D =	.03722361
DISTANCE FROM D TO E =	.12407866
DISTANCE FROM E TO F =	.31062244
DISTANCE FROM F TO G =	-.31062244
DISTANCE FROM G TO H =	-.93059023
DIHEDRAL ANGLE (DEGREES) =	3.00000000
LIFTING EFFICIENCY K =	1.00103920

WING WITH ONE PYLON INBOARD AND OUTBOARD DIHEDRAL WITH BREAK OUTBOARD OF PYLON

RATIO OF PYLON LOAD TO WING LIFT LOAD = .00077602

RATIO OF OUTBOARD WING NORMAL LOAD TO WING LIFT LOAD = .14438686

WING SPAN COORDINATE	WING LIFT LOADING	OUTBOARD WING NORMAL LOADING	ELLIPTICAL WING LOADING	PYLON COORDINATE	PYLON LOADING
0.0000000	1.2729037		1.2732395	0.0000000	1.2728297
.0124592	1.2728053		1.2731407	.0385591	1.2718823
.0221200	1.2725937		1.2729280	.0684279	1.2698434
.0348109	1.2721358		1.2724678	.1075950	1.2654333
.0527449	1.2711400		1.2714672	.1627240	1.2558473
.0787857	1.2689653		1.2692817	.2420748	1.2349296
.1169476	1.2642095		1.2645026	.3561078	1.1892744
.1726005	1.2538877		1.2541306	.5151480	1.0906297
.2366113	1.2369318		1.2370851	.6823013	.9299892
.2860681	1.2199842		1.2200299	.7952624	.7710488
.3276347	1.2030249		1.2029624	.8758004	.6136537
.3640311	1.1860756		1.1858785	.9328457	.4576990
.3966916	1.1691149		1.1687727	.9710076	.3029506
.4264868	1.1521427		1.1516372	.9928988	.1479289
.4539963	1.1351632		1.1344609	.9999997	.0000003
.4539963	1.1351632		1.1344609		
.4796351	1.1181575		1.1172270		
.5037189	1.1011186		1.0999104		
.5265098	1.0840312		1.0824689		
.5482492	1.0668521		1.0648293		
.5692060	1.0495265		1.0468511		
.5897855	1.0318795		1.0282172		
.6109444	1.0134777		1.0079900		
.6255897	1.0011341		.9933206		
.6342034	.9944136		.9844258		
.6396250	.9905472		.9787228		
.6432135	.9881882		.9749028		
.6457055	.9866577		.9722285		
.6475862	.9855647		.9701985		
.6500000	.9842424		.9675782		
.6500000	.9513178		.9675782		
.6513307	.9505945		.9661265		
.6523639	.9500060		.9649957		
.6537247	.9491953		.9635017		
.6556582	.9479748		.9613694		
.6584993	.9460394		.9582163		
.6627685	.9428326		.9534327		
.6693203	.9372931		.9459838		
.6775063	.9295298		.9364892		
.6844239	.9224066		.9282990		
.6906982	.9155889		.9207344		
.6965673	.9089550		.9135384		
.7021485	.9024451		.9065854		
.7075081	.8960255		.8998050		
.7126872	.8896742		.8931545		
.7126872	.8896742		.8931545		
.7177129	.8833797		.8866064		
.7226036	.8771336		.8801429		
.7273720	.8709307		.8737522		
.7320268	.8647739		.8674273		

.7365733	.8586585		.8611654
.7410125	.8525891		.8549691
.7453390	.8465826		.8488505
.7479070	.8429735		.8451809
.7490664	.8413323		.8435147
.7495879	.8405912		.8427633
.7498231	.8402562		.8424240
.7499297	.8401042		.8422703
.7499781	.8400349		.8422004
.7500000	.8400032		.8421688
.7500000	.8400032	.8411559	.8421688
.7503801	.8394588	.8406108	.8416196
.7512194	.8382601	.8394105	.8404050
.7530555	.8356253	.8367721	.8377368
.7570604	.8298265	.8309653	.8318645
.7657179	.8170266	.8181479	.8189179
.7839750	.7887805	.7898630	.7904173
.8202638	.7269817	.7279793	.7282744
.8709295	.6246504	.6255076	.6256800
.9119371	.5216665	.5223824	.5225333
.9443304	.4181568	.4187307	.4188939
.9689531	.3141574	.3145885	.3148018
.9862676	.2096964	.2099842	.2102823
.9965710	.1048076	.1049514	.1053509
1.0000000	.0000001	.0000001	0.0000000

WING WITH TWO BREAK STATIONS, PYLON AT THE INBOARD STATION, AND DIHEDRAL OUTBOARD OF THE OUTBOARD STATION (CONFIGURATION 6)

TRANSFORMATION CONSTANTS (INDEPENDENT VARIABLES)								ERRORS IN REQUIRED CONDITIONS						
A	B	C	D	E	F	G	H	INBOARD WING CLOSURE	PYLON CLOSURE	OUTBOARD WING CLOSURE	PYLON LOC.	DIHEDRAL BREAK LOC.	PYLON LENGTH	SQUARE ROOT OF SUM OF SQR. ERRORS
3.27012	0.0	.03107	.06424	.16478	1.0	1.80371	2.21344	-.06413	.00079	-.02180	-.02992	.01347	-.03106	.08142102
-1.711	-.009	.141	-1.530	-.100	.035									
-1.945	1.896	.294	-2.109	-.210	.234									
-2.014	1.655	.443	-1.955	-.287	1.046									
.955	-.008	-.853	.494	.515	.020									
1.749	.004	1.460	.628	-.593	.037									
.822	-.001	-.932	.403	-.152	.018									
-.22882	0.0	.05899	.12380	.19858	1.0	1.74591	2.12231	-.00448	-.00430	.00057	.00225	-.00122	-.00088	.00679406
-.22907	0.0	.05667	.12145	.20332	1.0	1.74211	2.11835	.00055	.00041	-.00054	.00071	.00041	.00042	.00127116
-.22841	0.0	.05645	.12074	.20269	1.0	1.74196	2.11794	.00001	.00006	-.00001	-.00018	-.00002	.00004	.00019675
-.22861	0.0	.05646	.12074	.20257	1.0	1.74206	2.11816	-.00000	.00001	.00001	-.00000	-.00000	-.00002	.00001997
-.22858	0.0	.05648	.12077	.20259	1.0	1.74205	2.11813	-.00000	-.00001	.00000	.00000	-.00000	.00000	.00000716
-.22858	0.0	.05648	.12077	.20259	1.0	1.74205	2.11814	.00000	.00000	-.00000	.00000	.00000	.00000	.00000155
-.22858	0.0	.05648	.12077	.20259	1.0	1.74205	2.11814	-.00000	-.00000	.00000	-.00000	-.00000	.00000	.00000026
-.22858	0.0	.05648	.12077	.20259	1.0	1.74205	2.11814	.00000	.00000	.00000	.00000	.00000	-.00000	.00000005
-.22858	0.0	.05648	.12077	.20259	1.0	1.74205	2.11814	-.00000	-.00000	.00000	-.00000	-.00000	.00000	.00000001
-.22858	0.0	.05648	.12077	.20259	1.0	1.74205	2.11814	-.00000	-.00000	.00000	-.00000	-.00000	.00000	.00000001

TRANSFORMATION CONSTANTS IN EXPONENTIAL FORMAT

-2.28581047E-01 0. 5.64802133E-02 1.20769658E-01 2.02592789E-01 1.00000000E+00 1.74204706E+00 2.11813574E+00

CLOSURE INTEGRAL (INBOARD WING) =	-.00000001
CLOSURE INTEGRAL (PYLON) =	-.00000001
CLOSURE INTEGRAL (OUTBOARD WING) =	.00000000
PYLON LOCATION (SEMISPAN) =	.65000000
DIHEDRAL BREAK LOCATION (SEMISPAN) =	.75000000
PYLON LENGTH (SEMISPANS) =	.06000000
INDUCED DRAG INTEGRAL =	2.15719016
DISTANCE FROM A TO B =	.76031183
DISTANCE FROM B TO C =	-.07018264
DISTANCE FROM C TO D =	.07018263
DISTANCE FROM D TO E =	.11697105
DISTANCE FROM E TO F =	.29403840
DISTANCE FROM F TO G =	-.29403840
DISTANCE FROM G TO H =	-.87728288
DIHEDRAL ANGLE (DEGREES) =	6.00000000
LIFTING EFFICIENCY K =	1.00371813

WING WITH ONE PYLON INBOARD AND OUTBOARD DIHEDRAL WITH BREAK OUTBOARD OF PYLON

RATIO OF PYLON LOAD TO WING LIFT LOAD = .00308473

RATIO OF OUTBOARD WING NORMAL LOAD TO WING LIFT LOAD = .14434083

WING SPAN COORDINATE	WING LIFT LOADING	OUTBOARD WING NORMAL LOADING	ELLIPTICAL WING LOADING	PYLON COORDINATE	PYLON LOADING
0.0000000	1.2723846		1.2732395	0.0000000	1.2736497
.0118809	1.2722965		1.2731496	.0393439	1.2726586
.0210935	1.2721068		1.2729562	.0698168	1.2705259
.0331961	1.2716964		1.2725378	.1097668	1.2659140
.0503002	1.2708038		1.2716278	.1659692	1.2558945
.0751408	1.2688544		1.2696400	.2467753	1.2340546
.1115593	1.2645909		1.2652917	.3626195	1.1864979
.1647205	1.2553353		1.2558474	.5233457	1.0842860
.2259806	1.2401232		1.2403031	.6906841	.9195107
.2734370	1.2248951		1.2247162	.8024308	.7583617
.3134419	1.2096553		1.2090776	.8811674	.6004437
.3485916	1.1944001		1.1933752	.9362827	.4454484
.3802604	1.1791011		1.1775931	.9727192	.2928283
.4092890	1.1637737		1.1617098	.9933734	.1396268
.4362461	1.1483886		1.1456958	.9999998	.0000003
.4362461	1.1483886		1.1456958		
.4615499	1.1329321		1.1295089		
.4855347	1.1173804		1.1130881		
.5085010	1.1016777		1.0963374		
.5307591	1.0857612		1.0790994		
.5527033	1.0694983		1.0610899		
.5749869	1.0526041		1.0417169		
.5991732	1.0343254		1.0193803		
.6171474	1.0213583		1.0018454		
.6283572	1.0138423		.9904848		
.6356602	1.0092855		.9829014		
.6405811	1.0063889		.9777085		
.6440281	1.0044509		.9740305		
.6466401	1.0030340		.9712210		
.6500000	1.0012780		.9675783		
.6500000	.9357970		.9675783		
.6516293	.9349463		.9658000		
.6528934	.9342624		.9644150		
.6545557	.9333313		.9625866		
.6569100	.9319510		.9599831		
.6603452	.9298089		.9561547		
.6654324	.9263611		.9504200		
.6730170	.9206423		.9417223		
.6820803	.9129727		.9310911		
.6894027	.9061780		.9223071		
.6958132	.8998355		.9144696		
.7016408	.8937778		.9072224		
.7070535	.8879213		.9003841		
.7121500	.8822193		.8938489		
.7169929	.8766375		.8875504		
.7169929	.8766375		.8875504		
.7216246	.8711550		.8814441		
.7260743	.8657573		.8755001		
.7303630	.8604499		.8696980		
.7345047	.8552176		.8640250		

.7385079	.8500553		.8584752
.7423746	.8449856		.8530515
.7460953	.8400161		.8477728
.7482713	.8370670		.8446580
.7492394	.8357419		.8432656
.7496688	.8351502		.8426467
.7498598	.8348852		.8423711
.7499451	.8347668		.8422480
.7499832	.8347131		.8421930
.7500000	.8346887		.8421688
.7500000	.8346887	.8392864	.8421688
.7503582	.8341852	.8387801	.8416514
.7511704	.8330594	.8376481	.8404760
.7529730	.8305476	.8351225	.8378571
.7569448	.8249646	.8295088	.8320350
.7655806	.8125393	.8170150	.8191259
.7838031	.7849590	.7892828	.7906937
.8199147	.7243752	.7283653	.7289114
.8702903	.6236340	.6270691	.6271203
.9111737	.5218025	.5246767	.5246032
.9436738	.4190287	.4213369	.4212858
.[illegible]84776	.3153407	.3170777	.3171647
.98600[illegible]1	.2107650	.2119259	.2122455
.9964936	.1053409	.1059212	.1065305
1.0000000	.0000001	.0000001	0.0000000

WING WITH TWO BREAK STATIONS, PYLON AT THE INBOARD STATION, AND DIHEDRAL OUTBOARD OF THE OUTBOARD STATION (CONFIGURATION 6)

TRANSFORMATION CONSTANTS (INDEPENDENT VARIABLES)								ERRORS IN REQUIRED CONDITIONS						
A	B	C	D	E	F	G	H	INBOARD WING CLOSURE	PYLON CLOSURE	OUTBOARD WING CLOSURE	PYLON LOC.	DIHEDRAL BREAK LOC.	PYLON LENGTH	SQUARE ROOT OF SUM OF SQR. ERRORS
-.22858	0.0	.05648	.12077	.20259	1.0	1.74205	2.11814	-.06009	.00139	-.02076	-.02904	.01398	-.02198	.07459871
-1.703	-.038	.133	-1.602	-.100	.067									
-1.883	2.056	.238	-1.964	-.184	.241									
-2.047	1.555	.478	-1.722	-.325	1.004									
1.035	-.018	-.886	.570	.564	.044									
1.747	.014	1.469	.665	-.629	.082									
.791	-.002	-.960	.412	-.154	.037									
-.20266	0.0	.07152	.15519	.22571	1.0	1.69637	2.04519	-.00246	-.00162	-.00033	.00114	-.00048	.00016	.00321086
-.20249	0.0	.06998	.15271	.22721	1.0	1.69407	2.04302	.00021	.00023	-.00020	.00005	.00015	.00018	.00044029
-.20247	0.0	.06997	.15253	.22682	1.0	1.69417	2.04311	-.00001	.00000	.00001	-.00004	-.00002	-.00001	.00004468
-.20248	0.0	.06998	.15257	.22685	1.0	1.69417	2.04312	-.00000	-.00000	.00000	.00000	.00000	-.00000	.00000488
-.20248	0.0	.06998	.15257	.22685	1.0	1.69417	2.04312	.00000	.00000	-.00000	-.00000	.00000	.00000	.00000035
-.20248	0.0	.06998	.15257	.22685	1.0	1.69417	2.04312	.00000	.00000	.00000	-.00000	.00000	-.00000	.00000006
-.20248	0.0	.06998	.15257	.22685	1.0	1.69417	2.04312	-.00000	-.00000	.00000	.00000	-.00000	-.00000	.00000001
-.20248	0.0	.06998	.15257	.22685	1.0	1.69417	2.04312	-.00000	-.00000	.00000	.00000	-.00000	-.00000	.00000001

TRANSFORMATION CONSTANTS IN EXPONENTIAL FORMAT

-2.02482699E-01 0. 6.99818674E-02 1.52569840E-01 2.26847730E-01 1.00000000E+00 1.69416681E+00 2.04312056E+00

CLOSURE INTEGRAL (INBOARD WING) = -.00000000
CLOSURE INTEGRAL (PYLON) = -.00000001
CLOSURE INTEGRAL (OUTBOARD WING) = .00000000

PYLON LOCATION (SEMISPAN) = .65000000
DIHEDRAL BREAK LOCATION (SEMISPAN) = .75000000
PYLON LENGTH (SEMISPANS) = .08000000

INDUCED DRAG INTEGRAL = 1.97188280

DISTANCE FROM A TO B = .72575321
DISTANCE FROM B TO C = -.08932348
DISTANCE FROM C TO D = .08932347
DISTANCE FROM D TO E = .11165433
DISTANCE FROM E TO F = .28261532
DISTANCE FROM F TO G = -.28261531
DISTANCE FROM G TO H = -.83740755

DIHEDRAL ANGLE (DEGREES) = 9.00000000

LIFTING EFFICIENCY K = 1.00695486

WING WITH ONE PYLON INBOARD AND OUTBOARD DIHEDRAL WITH BREAK OUTBOARD OF PYLON

RATIO OF PYLON LOAD TO WING LIFT LOAD = .00542369

RATIO OF OUTBOARD WING NORMAL LOAD TO WING LIFT LOAD = .14481547

WING SPAN COORDINATE	WING LIFT LOADING	OUTBOARD WING NORMAL LOADING	ELLIPTICAL WING LOADING	PYLON COORDINATE	PYLON LOADING
0.0000000	1.2714035		1.2732395	0.0000000	1.2741320
.0115315	1.2713218		1.2731548	.0398710	1.2731106
.0204734	1.2711460		1.2729726	.0707493	1.2709130
.0322206	1.2707656		1.2725784	.1112245	1.2661618
.0488238	1.2699385		1.2717210	.1681454	1.2558433
.0729403	1.2681319		1.2698480	.2499217	1.2333705
.1083091	1.2641802		1.2657494	.3669608	1.1845215
.1599770	1.2555996		1.2568411	.5287644	1.0799366
.2196037	1.2414898		1.2421586	.6961500	.9125271
.2658900	1.2273508		1.2274072	.8070433	.7500807
.3049990	1.2131937		1.2125731	.8845766	.5919896
.3394521	1.1989983		1.1976388	.9384384	.4377594
.3705876	1.1847539		1.1825817	.9737793	.2866039
.3992283	1.1704573		1.1673716	.9936637	.1346425
.4259363	1.1560732		1.1519675	.9999999	.0000003
.4259363	1.1560732		1.1519675		
.4511315	1.1415930		1.1363122		
.4751586	1.1269720		1.1203237		
.4983389	1.1121529		1.1038761		
.5210194	1.0970502		1.0867670		
.5436585	1.0814961		1.0686382		
.5670330	1.0651663		1.0487611		
.5930004	1.0471771		1.0252152		
.6128010	1.0340942		1.0061613		
.6253761	1.0263230		.9935387		
.6336479	1.0215191		.9850056		
.6392479	1.0184228		.9791221		
.6431795	1.0163297		.9749391		
.6461617	1.0147876		.9717371		
.6500000	1.0128615		.9675782		
.6500000	.9264803		.9675782		
.6517339	.9256087		.9656856		
.6530788	.9249096		.9642115		
.6548468	.9239604		.9622656		
.6573491	.9225581		.9594957		
.6609946	.9203924		.9554270		
.6663756	.9169338		.9493480		
.6743438	.9112551		.9401823		
.6837574	.9037337		.9290948		
.6912657	.8971470		.9200437		
.6977652	.8910508		.9120552		
.7036144	.8852688		.9047410		
.7089980	.8797221		.8979018		
.7140251	.8743488		.8914206		
.7187653	.8691143		.8852233		
.7187653	.8691143		.8852233		
.7232661	.8639956		.8792603		
.7275601	.8589895		.8734982		
.7316705	.8540893		.8679145		
.7356126	.8492775		.8624956		

.7393950	.8445578		.8572363
.7430182	.8399553		.8521426
.7464672	.8354789		.8472419
.7484574	.8328478		.8443907
.7493304	.8316770		.8431345
.7497124	.8311585		.8425839
.7498799	.8309283		.8423421
.7499537	.8308260		.8422356
.7499861	.8307797		.8421888
.7500000	.8307588		.8421688
.7500000	.8307588	.8411143	.8421688
.7503314	.8302905	.8406402	.8416900
.7511034	.8292342	.8395707	.8405731
.7528428	.8268471	.8371539	.8380466
.7567206	.8214934	.8317334	.8323655
.7652248	.8094772	.8195675	.8196647
.7832615	.7826114	.7923668	.7915633
.8190922	.7232368	.7322520	.7304089
.8692289	.6239401	.6317176	.6295022
.9101224	.5230789	.5295991	.5275535
.9428018	.4208473	.4260932	.4244394
.9678740	.3172665	.3212213	.3201377
.9856881	.2123548	.2150019	.2146418
.9964000	.1061434	.1074665	.1079406
1.0000000	.0000000	.0000000	0.0000000

WING WITH TWO BREAK STATIONS, PYLON AT THE INBOARD STATION, AND DIHEDRAL OUTBOARD OF THE OUTBOARD STATION (CONFIGURATION 6)

TRANSFORMATION CONSTANTS (INDEPENDENT VARIABLES)								ERRORS IN REQUIRED CONDITIONS						
A	B	C	D	E	F	G	H	INBOARD WING CLOSURE	PYLON CLOSURE	OUTBOARD WING CLOSURE	PYLON LOC.	DIHEDRAL BREAK LOC.	PYLON LENGTH	SQUARE ROOT OF SUM OF SQR. ERRORS
-.20248	0.0	.06998	.15257	.22685	1.0	1.69417	2.02312	-.01903	.00058	-.00667	-.00956	.00484	-.00084	.02285962
-1.704	-.073	.135	-1.705	-.102	.089									
-1.821	2.217	.231	-2.004	-.175	.252									
-2.076	1.529	.508	-1.714	-.356	1.014									
1.091	-.023	-.927	.648	.616	.067									
1.703	.022	1.464	.697	-.674	.116									
.777	-.003	-.966	.444	-.166	.052									
-.20134	0.0	.06963	.15137	.22648	1.0	1.68440	2.02724	-.00004	.00000	.00001	.00007	-.00002	.00000	.00008439
-.20125	0.0	.06961	.15131	.22654	1.0	1.68432	2.02716	-.00000	-.00000	.00000	-.00000	-.00000	-.00000	.00000146
-.20125	0.0	.06961	.15131	.22654	1.0	1.68432	2.02716	.00000	.00000	-.00000	.00000	.00000	.00000	.00000003
-.20125	0.0	.06961	.15131	.22654	1.0	1.68432	2.02716	.00000	.00000	-.00000	.00000	.00000	.00000	.00000003

TRANSFORMATION CONSTANTS IN EXPONENTIAL FORMAT

-2.01253632E-01 0. 6.96074103E-02 1.51313696E-01 2.26544581E-01 1.00000000E+00 1.68432348E+00 2.02716079E+00

CLOSURE INTEGRAL (INBOARD WING) = .00000001
CLOSURE INTEGRAL (PYLON) = .00000001
CLOSURE INTEGRAL (OUTBOARD WING) = -.00000001

PYLON LOCATION (SEMISPAN) = .65000002
DIHEDRAL BREAK LOCATION (SEMISPAN) = .75000000
PYLON LENGTH (SEMISPANS) = .08000001

INDUCED DRAG INTEGRAL = 1.94195300

DISTANCE FROM A TO B = .72001131
DISTANCE FROM B TO C = -.08861678
DISTANCE FROM C TO D = .08861678
DISTANCE FROM D TO E = .11077095
DISTANCE FROM E TO F = .28119947
DISTANCE FROM F TO G = -.28119948
DISTANCE FROM G TO H = -.83078226

DIHEDRAL ANGLE (DEGREES) = 10.00000000

LIFTING EFFICIENCY K = 1.00755076

WING WITH ONE PYLON INBOARD AND OUTBOARD DIHEDRAL WITH BREAK OUTBOARD OF PYLON

RATIO OF PYLON LOAD TO WING LIFT LOAD = .00541965

RATIO OF OUTBOARD WING NORMAL LOAD TO WING LIFT LOAD = .14530123

WING SPAN COORDINATE	WING LIFT LOADING	OUTBOARD WING NORMAL LOADING	ELLIPTICAL WING LOADING	PYLON COORDINATE	PYLON LOADING
0.0000000	1.2709830		1.2732395	0.0000000	1.2739171
.0115344	1.2709015		1.2731548	.0398739	1.2728969
.0204784	1.2707259		1.2729725	.0707545	1.2707019
.0322285	1.2703461		1.2725781	.1112326	1.2659561
.0488357	1.2695201		1.2717203	.1681575	1.2556494
.0729580	1.2677159		1.2698463	.2499396	1.2332015
.1083349	1.2637698		1.2657458	.3669844	1.1844040
.1600138	1.2552016		1.2568335	.5287929	1.0799165
.2196514	1.2411135		1.2421450	.6961769	.9126257
.2659442	1.2269985		1.2273882	.8070642	.7502532
.3050569	1.2128663		1.2125494	.8845904	.5921969
.3395118	1.1986988		1.1976114	.9384459	.4379679
.3706476	1.1844835		1.1825512	.9737823	.2867869
.3992873	1.1702184		1.1673389	.9936643	.1348078
.4259931	1.1558697		1.1519334	1.0000000	.0000008
.4259931	1.1558697		1.1519334		
.4511851	1.1414269		1.1362776		
.4752082	1.1268479		1.1202895		
.4983836	1.1120743		1.1038434		
.5210585	1.0970228		1.0867366		
.5436912	1.0815282		1.0686112		
.5670585	1.0652681		1.0487388		
.5930176	1.0473719		1.0251991		
.6128120	1.0343717		1.0061504		
.6253833	1.0266602		.9935314		
.6336526	1.0218983		.9850006		
.6392510	1.0188326		.9791188		
.6431815	1.0167618		.9749370		
.6461628	1.0152370		.9717359		
.6500000	1.0133338		.9675782		
.6500000	.9270315		.9675782		
.6517352	.9261691		.9656841		
.6530812	.9254770		.9642088		
.6548507	.9245363		.9622613		
.6573550	.9231453		.9594891		
.6610037	.9209945		.9554168		
.6663899	.9175539		.9493318		
.6743672	.9118933		.9401551		
.6837942	.9043819		.9290509		
.6913150	.8977959		.9199836		
.6978262	.8916954		.9119796		
.7036860	.8859078		.9046507		
.7090790	.8803550		.8977981		
.7141140	.8749752		.8913052		
.7188603	.8697343		.8850982		
.7188603	.8697343		.8850982		
.7233651	.8646095		.8791282		
.7276607	.8596031		.8733624		
.7317697	.8547027		.8677788		
.7357071	.8498925		.8623649		

.7394805	.8451806		.8571168
.7430891	.8405891		.8520424
.7465153	.8361318		.8471732
.7484849	.8335174		.8443510
.7493453	.8323565		.8431131
.7497200	.8318434		.8425728
.7498837	.8316160		.8423367
.7499554	.8315150		.8422332
.7499867	.8314693		.8421879
.7500000	.8314487		.8421688
.7500000	.8314487	.8442751	.8421688
.7503178	.8309915	.8438109	.8417097
.7510648	.8299591	.8427626	.8406289
.7527575	.8276186	.8403860	.8381708
.7565503	.8223550	.8350412	.8326165
.7649100	.8105047	.8230080	.8201408
.7827344	.7839174	.7960106	.7924081
.8183402	.7249386	.7361220	.7317742
.8684143	.6259624	.6356189	.6313221
.9094106	.5251854	.5332872	.5295396
.9422620	.4228515	.4293746	.4263780
.9675247	.3189912	.3239121	.3218449
.9855123	.2136240	.2169195	.2159467
.9963507	.1067774	.1084246	.1086762
1.0000000	.0000000	.0000000	0.0000000

```
* * * * * * * * * * * * * * * * *   NE= CASE DATA LOADED BY SUBROUTINE INPUT   * * * * * * * * * * * * * * * * * * * *

                    CONFIGURATION   7

          PROGRAM OPERATION MODE NUMBER =   2

          NUMBER OF INTEGRATION STEPS =      15
```

WING WITH AN END-PLATED VERTICAL FENCE AT THE TIP (CONFIGURATION 7)

TRANSFORMATION CONSTANTS (INDEPENDENT VARIABLES)								ERRORS IN REQUIRED CONDITIONS						
A	B	C	D	E	F	G	H	WING CLOSURE	VERTICAL FENCE CLOSURE	END PLATE CLOSURE	VERTICAL FENCE LENGTH	INBOARD E. P. LENGTH	OUTBOARD E. P. LENGTH	SQUARE ROOT OF SUM OF SQR. ERRORS
0.0	.70139	.72838	.75639	1.0	1.07727	1.34024	2.34719	.00000	.00000	.00000	-.01000	-.00500	-.00500	.01224745
-1.296	.616	-.082	-.254	.140	.017									
-2.176	.570	.800	-.210	-.716	.022									
3.568	-1.053	-.816	.387	.502	-.043									
-1.158	.115	-.218	.005	-.014	.184									
1.387	.197	-.005	.140	.011	.020									
-1.372	-.005	.000	-.033	-.008	-.008									
0.0	.63922	.66687	.69596	1.0	1.10013	1.37673	2.36393	-.00389	.00037	.00349	.00019	-.00228	.00050	.00573868
0.0	.64862	.67124	.70412	1.0	1.09810	1.37776	2.36928	.00160	-.00019	-.00160	.00004	.00118	-.00011	.00256085
0.0	.64563	.67070	.70183	1.0	1.09846	1.37711	2.36719	-.00072	.00008	.00073	-.00002	-.00060	.00003	.00118935
0.0	.64668	.67047	.70242	1.0	1.09839	1.37729	2.36782	.00029	-.00003	-.00030	.00000	.00026	-.00001	.00049389
0.0	.64630	.67065	.70226	1.0	1.09841	1.37724	2.36763	-.00012	.00001	.00013	-.00000	-.00011	.00000	.00021102
0.0	.64644	.67055	.70230	1.0	1.09840	1.37725	2.36769	.00005	-.00000	-.00005	.00000	.00005	-.00000	.00008743
0.0	.64639	.67060	.70229	1.0	1.09841	1.37725	2.36767	-.00002	.00000	.00002	.00000	-.00002	.00000	.00003642
0.0	.64641	.67058	.70229	1.0	1.09840	1.37725	2.36768	.00001	-.00000	-.00001	-.00000	.00001	-.00000	.00001505
0.0	.64640	.67059	.70229	1.0	1.09841	1.37725	2.36767	-.00000	.00000	.00000	.00000	-.00000	.00000	.00000622
0.0	.64641	.67058	.70229	1.0	1.09840	1.37725	2.36767	.00000	-.00000	-.00000	-.00000	.00000	-.00000	.00000257
0.0	.64640	.67058	.70229	1.0	1.09840	1.37725	2.36767	-.00000	.00000	.00000	.00000	-.00000	.00000	.00000106
0.0	.64640	.67058	.70229	1.0	1.09840	1.37725	2.36767	.00000	-.00000	-.00000	-.00000	.00000	-.00000	.00000044
0.0	.64640	.67058	.70229	1.0	1.09840	1.37725	2.36767	-.00000	.00000	.00000	.00000	-.00000	.00000	.00000018
0.0	.64640	.67058	.70229	1.0	1.09840	1.37725	2.36767	.00000	-.00000	-.00000	-.00000	.00000	-.00000	.00000007
0.0	.64640	.67058	.70229	1.0	1.09840	1.37725	2.36767	-.00000	.00000	.00000	.00000	-.00000	.00000	.00000003
0.0	.64640	.67058	.70229	1.0	1.09840	1.37725	2.36767	.00000	-.00000	-.00000	-.00000	.00000	-.00000	.00000001
0.0	.64640	.67058	.70229	1.0	1.09840	1.37725	2.36767	.00000	-.00000	-.00000	-.00000	.00000	-.00000	.00000001

TRANSFORMATION CONSTANTS IN EXPONENTIAL FORMAT

0.	6.46404566E-01	6.70582756E-01	7.02294162E-01	1.00000000E+00	1.09840496E+00	1.37725122E+00	2.36767429E+00

WING CLOSURE INTEGRAL = .00000001
VERTICAL FENCE CLOSURE INTEGRAL = -.00000000
END-PLATE CLOSURE INTEGRAL = -.00000001

VERTICAL FENCE LENGTH = .05000000
INBOARD END-PLATE LENGTH = .01500001
OUTBOARD END-PLATE LENGTH = .01500000

INDUCED DRAG INTEGRAL = 2.19445678

DISTANCE FROM A TO B = 1.13813547
DISTANCE FROM B TO C = -.05690677
DISTANCE FROM C TO D = -.01707204
DISTANCE FROM D TO E = .03414406
DISTANCE FROM E TO F = -.01707203
DISTANCE FROM F TO G = .05690677
DISTANCE FROM G TO H = -1.13813546

LIFTING EFFICIENCY K = 1.07849773

WING WITH END-PLATED VERTICAL FENCE AT THE TIP

RATIO OF VERTICAL FENCE LOAD TO WING LOAD = .0140352
RATIO OF INBOARD END PLATE LOAD TO WING LOAD = -.0003688
RATIO OF OUTBOARD END PLATE LOAD TO WING LOAD = .0007515

WING SPAN COORDINATE	WING LOADING	ELLIPTICAL WING LOADING	END PLATE SPAN COORDINATE	END PLATE LOADING, STARTING OUTBOARD	VERTICAL FENCE COORDINATE	VERTICAL FENCE LOADING
0.000000	1.229808	1.273240	1.000000	.000144	0.000000	1.352363
.020725	1.229564	1.272966	.996750	.487915	.015823	1.321425
.036793	1.229041	1.272377	.986480	.997169	.028093	1.300340
.057895	1.227907	1.271104	.968176	1.560209	.044215	1.276498
.087697	1.225443	1.268334	.940328	2.104756	.067007	1.250272
.130914	1.220059	1.262282	.900474	2.725921	.100132	1.227777
.194066	1.208278	1.249033	.844026	3.378154	.148772	1.195679
.285582	1.182680	1.220215	.759962	4.137227	.220030	1.147817
.389594	1.140498	1.172637	.679752	4.702258	.302736	1.095801
.468703	1.098354	1.124724	.622198	5.049326	.367467	1.056896
.534109	1.055773	1.076417	.582143	5.259620	.422681	1.024892
.590405	1.013162	1.027640	.554314	5.396488	.471856	.996921
.640028	.970890	.978296	.534543	5.489228	.516861	.971881
.684462	.927667	.928253	.519465	5.557467	.558872	.949037
.724707	.884402	.877335	.500000	5.642394	.598720	.927973
.724707	.884402	.877335	.500000	-3.066821	.598720	.927973
.761484	.841231	.825293	.477444	-2.970478	.637066	.908230
.795352	.796710	.771772	.459993	-2.895945	.674498	.889488
.826787	.751948	.716216	.437157	-2.802896	.711619	.871706
.856245	.706111	.657717	.405164	-2.671230	.749150	.854991
.884251	.658183	.594619	.359579	-2.478493	.788118	.838706
.911607	.607929	.523381	.295483	-2.196373	.830327	.823038
.940170	.549759	.433801	.210100	-1.894559	.880089	.807920
.960954	.504957	.352314	.127258	-1.436935	.920285	.799009
.974010	.474174	.288392	.076440	-1.029168	.946763	.794965
.982660	.453518	.236080	.043407	-.697014	.964500	.793128
.988566	.435680	.191987	.022023	-.437460	.976613	.792291
.992737	.416826	.153181	.008932	-.241346	.985152	.791905
.995909	.401255	.115049	.002057	-.098656	.991640	.791728
1.000000	.379614	0.000000	0.000000	0.000000	1.000000	.791645

WING WITH AN END-PLATED VERTICAL FENCE AT THE TIP (CONFIGURATION 7)

TRANSFORMATION CONSTANTS (INDEPENDENT VARIABLES)								ERRORS IN REQUIRED CONDITIONS						
A	B	C	D	E	F	G	H	WING CLOSURE	VERTICAL FENCE CLOSURE	END PLATE CLOSURE	VERTICAL FENCE LENGTH	INBOARD E. P. LENGTH	OUTBOARD E. P. LENGTH	SQUARE ROOT OF SUM OF SQR. ERRORS
0.0	.64640	.67058	.70229	1.0	1.09840	1.37725	2.36767	.00000	-.00000	-.00000	-.01000	-.00500	-.00500	.01224745
-1.233	.715	-.147	-.298	.227	.021									
-2.476	.670	1.099	-.242	-.987	.032									
3.811	-1.235	-1.058	.455	.655	-.058									
-1.124	.103	-.253	-.003	-.017	.212									
1.391	.226	-.008	.160	.014	.029									
-1.373	-.007	.000	-.041	-.011	-.013									
0.0	.59405	.62061	.65281	1.0	1.11762	1.41253	2.38746	-.00449	.00034	.00396	.00024	-.00295	.00037	.00670101
0.0	.60077	.62255	.65792	1.0	1.11642	1.41257	2.38987	.00122	-.00012	-.00120	-.00002	.00093	-.00007	.00195172
0.0	.59917	.62239	.65680	1.0	1.11660	1.41244	2.38910	-.00041	.00004	.00041	.00000	-.00035	.00002	.00067237
0.0	.59960	.62229	.65703	1.0	1.11657	1.41246	2.38928	.00012	-.00001	-.00012	-.00000	.00011	-.00000	.00020291
0.0	.59949	.62234	.65698	1.0	1.11658	1.41246	2.38923	-.00004	.00000	.00004	.00000	-.00003	.00000	.00006173
0.0	.59952	.62232	.65699	1.0	1.11657	1.41246	2.38924	.00001	-.00000	-.00001	-.00000	.00001	-.00000	.00001829
0.0	.59951	.62233	.65698	1.0	1.11657	1.41246	2.38924	-.00000	.00000	.00000	.00000	-.00000	.00000	.00000539
0.0	.59951	.62232	.65699	1.0	1.11657	1.41246	2.38924	.00000	-.00000	-.00000	-.00000	.00000	-.00000	.00000158
0.0	.59951	.62233	.65699	1.0	1.11657	1.41246	2.38924	-.00000	.00000	.00000	.00000	-.00000	.00000	.00000046
0.0	.59951	.62233	.65699	1.0	1.11657	1.41246	2.38924	.00000	-.00000	-.00000	-.00000	.00000	-.00000	.00000013
0.0	.59951	.62233	.65699	1.0	1.11657	1.41246	2.38924	-.00000	.00000	.00000	.00000	-.00000	.00000	.00000004
0.0	.59951	.62233	.65699	1.0	1.11657	1.41246	2.38924	.00000	-.00000	-.00000	-.00000	.00000	-.00000	.00000001
0.0	.59951	.62233	.65699	1.0	1.11657	1.41246	2.38924	.00000	-.00000	-.00000	-.00000	.00000	-.00000	.00000001

TRANSFORMATION CONSTANTS IN EXPONENTIAL FORMAT

0.	5.99510962E-01	6.22325048E-01	6.56985316E-01	1.00000000E+00	1.11657425E+00	1.41246008E+00	2.38924212E+00

WING CLOSURE INTEGRAL =	.00000001
VERTICAL FENCE CLOSURE INTEGRAL =	-.00000000
END-PLATE CLOSURE INTEGRAL =	-.00000001
VERTICAL FENCE LENGTH =	.06000000
INBOARD END-PLATE LENGTH =	.02000001
OUTBOARD END-PLATE LENGTH =	.02000000
INDUCED DRAG INTEGRAL =	2.23101044
DISTANCE FROM A TO B =	1.13818848
DISTANCE FROM B TO C =	-.06829131
DISTANCE FROM C TO D =	-.02276378
DISTANCE FROM D TO E =	.04552754
DISTANCE FROM E TO F =	-.02276377
DISTANCE FROM F TO G =	.06829131
DISTANCE FROM G TO H =	-1.13818847
LIFTING EFFICIENCY K =	1.09636043

WING WITH END-PLATED VERTICAL FENCE AT THE TIP

RATIO OF VERTICAL FENCE LOAD TO WING LOAD = .0187038
RATIO OF INBOARD END PLATE LOAD TO WING LOAD = -.0005421
RATIO OF OUTBOARD END PLATE LOAD TO WING LOAD = .0011570

WING SPAN COORDINATE	WING LOADING	ELLIPTICAL WING LOADING	END PLATE SPAN COORDINATE	END PLATE LOADING, STARTING OUTBOARD	VERTICAL FENCE COORDINATE	VERTICAL FENCE LOADING
0.000000	1.221158	1.273240	1.000000	.000141	0.000000	1.332899
.020123	1.220935	1.272982	.996707	.471726	.015421	1.303582
.035724	1.220454	1.272427	.986323	.962823	.027380	1.283631
.056213	1.219414	1.271226	.967852	1.507314	.043094	1.261114
.085154	1.217151	1.268615	.939809	2.029598	.065311	1.236442
.127132	1.212209	1.262908	.899766	2.628084	.097609	1.215546
.188503	1.201394	1.250414	.843181	3.249680	.145061	1.185087
.277541	1.177888	1.223219	.759121	3.972277	.214662	1.140199
.378958	1.139145	1.178273	.679077	4.509931	.295630	1.091600
.456328	1.100347	1.132944	.621709	4.836622	.359190	1.055322
.520509	1.061260	1.087164	.581805	5.035388	.413575	1.025577
.575956	1.021940	1.040849	.554087	5.164359	.462171	.999610
.625031	.982824	.993891	.534397	5.251549	.506800	.976410
.669180	.943174	.946145	.519383	5.315588	.548610	.955279
.709376	.903029	.897418	.500000	5.395139	.588420	.935812
.709376	.903029	.897418	.500000	-2.820501	.588420	.935812
.746330	.862936	.847438	.477167	-2.728956	.626885	.917586
.780599	.821837	.795816	.459504	-2.659557	.664595	.900304
.812668	.779920	.741950	.436396	-2.571575	.702162	.883929
.843018	.737402	.684858	.404040	-2.446458	.740329	.868535
.872223	.692107	.622751	.357989	-2.264332	.780162	.853560
.901204	.644412	.551815	.293389	-1.999962	.823551	.839169
.932132	.588417	.461063	.207728	-1.724762	.875012	.825294
.955224	.544791	.376730	.125193	-1.293093	.916791	.817118
.970024	.513393	.309408	.074917	-.920182	.944393	.813408
.979943	.491724	.253727	.042415	-.620889	.962909	.811724
.986757	.475623	.206525	.021466	-.388980	.975561	.810956
.991582	.455758	.164860	.008689	-.214550	.984483	.810603
.995257	.439058	.123857	.001997	-.087810	.991263	.810440
1.000000	.415504	0.000000	0.000000	0.000000	1.000000	.810364

WING WITH AN END-PLATED VERTICAL FENCE AT THE TIP (CONFIGURATION 7)

TRANSFORMATION CONSTANTS (INDEPENDENT VARIABLES)								ERRORS IN REQUIRED CONDITIONS						
A	B	C	D	E	F	G	H	WING CLOSURE	VERTICAL FENCE CLOSURE	END PLATE CLOSURE	VERTICAL FENCE LENGTH	INBOARD E. P. LENGTH	OUTBOARD E. P. LENGTH	SQUARE ROOT OF SUM OF SQR. ERRORS
0.0	.59951	.62233	.65699	1.0	1.11657	1.41246	2.38924	.00000	-.00000	-.00000	-.02000	-.01000	-.01000	.02449490
-1.165	.808	-.217	-.340	.316	.025									
-2.755	.762	1.377	-.267	-1.239	.042									
4.024	-1.400	-1.272	.512	.789	-.072									
-1.095	.095	-.282	-.010	-.020	.236									
1.393	.252	-.011	.179	.018	.037									
-1.373	-.009	.001	-.049	-.015	-.017									
0.0	.50797	.53640	.57147	1.0	1.15036	1.47927	2.42955	-.01085	.00016	.00956	.00152	-.00657	.00122	.01600683
0.0	.52605	.54492	.58596	1.0	1.14675	1.47688	2.43213	.00476	-.00013	-.00461	-.00040	.00342	-.00036	.00747984
0.0	.51940	.54286	.58102	1.0	1.14768	1.47712	2.43043	-.00247	.00009	.00243	.00015	-.00202	.00013	.00401832
0.0	.52204	.54290	.58259	1.0	1.14739	1.47702	2.43103	.00109	-.00005	-.00108	-.00006	.00092	-.00005	.00178807
0.0	.52098	.54301	.58203	1.0	1.14751	1.47709	2.43083	-.00051	.00002	.00051	.00003	-.00045	.00002	.00085247
0.0	.52144	.54289	.58223	1.0	1.14746	1.47705	2.43090	.00023	-.00001	-.00023	-.00001	.00021	-.00001	.00038291
0.0	.52124	.54296	.58215	1.0	1.14748	1.47707	2.43087	-.00010	.00000	.00010	.00001	-.00009	.00000	.00017461
0.0	.52133	.54293	.58219	1.0	1.14747	1.47706	2.43088	.00005	-.00000	-.00005	-.00000	.00004	-.00000	.00007836
0.0	.52129	.54294	.58217	1.0	1.14747	1.47707	2.43088	-.00002	.00000	.00002	.00000	-.00002	.00000	.00003524
0.0	.52130	.54293	.58218	1.0	1.14747	1.47707	2.43088	.00001	-.00000	-.00001	-.00000	.00001	-.00000	.00001578
0.0	.52130	.54294	.58218	1.0	1.14747	1.47707	2.43088	-.00000	.00000	.00000	.00000	-.00000	.00000	.00000706
0.0	.52130	.54294	.58218	1.0	1.14747	1.47707	2.43088	.00000	-.00000	-.00000	-.00000	.00000	-.00000	.00000316
0.0	.52130	.54294	.58218	1.0	1.14747	1.47707	2.43088	-.00000	.00000	.00000	.00000	-.00000	.00000	.00000141
0.0	.52130	.54294	.58218	1.0	1.14747	1.47707	2.43088	.00000	-.00000	-.00000	-.00000	.00000	-.00000	.00000063
0.0	.52130	.54294	.58218	1.0	1.14747	1.47707	2.43088	-.00000	.00000	.00000	.00000	-.00000	.00000	.00000028
0.0	.52130	.54294	.58218	1.0	1.14747	1.47707	2.43088	.00000	-.00000	-.00000	-.00000	.00000	-.00000	.00000013
0.0	.52130	.54294	.58218	1.0	1.14747	1.47707	2.43088	-.00000	.00000	.00000	.00000	-.00000	.00000	.00000006
0.0	.52130	.54294	.58218	1.0	1.14747	1.47707	2.43088	.00000	-.00000	-.00000	-.00000	.00000	-.00000	.00000002
0.0	.52130	.54294	.58218	1.0	1.14747	1.47707	2.43088	.00000	-.00000	-.00000	-.00000	.00000	-.00000	.00000002

TRANSFORMATION CONSTANTS IN EXPONENTIAL FORMAT

0.	5.21299113E-01	5.42937168E-01	5.82176521E-01	1.00000000E+00	1.14747264E+00	1.47706628E+00	2.43087956E+00

WING CLOSURE INTEGRAL =	.00000001
VERTICAL FENCE CLOSURE INTEGRAL =	-.00000000
END-PLATE CLOSURE INTEGRAL =	-.00000001
VERTICAL FENCE LENGTH =	.08000000
INBOARD END-PLATE LENGTH =	.03000001
OUTBOARD END-PLATE LENGTH =	.03000000
INDUCED DRAG INTEGRAL =	2.30051779
DISTANCE FROM A TO B =	1.13744041
DISTANCE FROM B TO C =	-.09099523
DISTANCE FROM C TO D =	-.03412323
DISTANCE FROM D TO E =	.06824642
DISTANCE FROM E TO F =	-.03412321
DISTANCE FROM F TO G =	.09099523

```
DISTANCE FROM G TO H =      -1.13744039
LIFTING EFFICIENCY K =       1.13200517
```

WING WITH END-PLATED VERTICAL FENCE AT THE TIP

RATIO OF VERTICAL FENCE LOAD TO WING LOAD = .0289002
RATIO OF INBOARD END PLATE LOAD TO WING LOAD = -.0009239
RATIO OF OUTBOARD END PLATE LOAD TO WING LOAD = .0021035

WING SPAN COORDINATE	WING LOADING	ELLIPTICAL WING LOADING	END PLATE SPAN COORDINATE	END PLATE LOADING, STARTING OUTBOARD	VERTICAL FENCE COORDINATE	VERTICAL FENCE LOADING
0.000000	1.205262	1.273240	1.000000	.000138	0.000000	1.311773
.019105	1.205072	1.273007	.996652	.452561	.014989	1.284198
.033917	1.204661	1.272507	.986117	.922035	.026613	1.265463
.053373	1.203771	1.271425	.967428	1.438081	.041888	1.244363
.080858	1.201837	1.269070	.939133	1.939981	.063487	1.221347
.120740	1.197611	1.263925	.898847	2.511153	.094895	1.202137
.179100	1.188363	1.252652	.842089	3.095952	.141068	1.173422
.263934	1.168250	1.228091	.758036	3.774601	.208884	1.131680
.360933	1.135086	1.187412	.678207	4.279272	.287973	1.086693
.435318	1.101744	1.146268	.621079	4.581430	.350265	1.053188
.497376	1.068298	1.104580	.581369	4.766252	.403749	1.025827
.551327	1.034366	1.062250	.553795	4.885668	.451711	1.001975
.599410	1.000442	1.019154	.534211	4.966137	.495924	.980719
.643004	.966505	.975127	.519277	5.025090	.537508	.961399
.683043	.931527	.929947	.500000	5.098127	.577269	.943618
.683043	.931527	.929947	.500000	-2.524678	.577269	.943618
.720219	.896333	.883306	.476784	-2.439097	.615854	.926997
.755089	.860865	.834764	.458827	-2.376391	.653855	.911262
.788159	.823763	.783643	.435345	-2.293729	.691897	.896381
.819955	.785730	.728838	.402488	-2.176781	.730746	.882388
.851154	.745714	.668344	.355801	-2.007845	.771513	.868802
.882891	.701756	.597885	.290526	-1.765431	.816177	.855769
.917928	.650077	.505154	.204522	-1.520166	.869481	.843214
.945104	.607209	.416055	.122444	-1.122577	.912984	.835820
.963004	.576291	.343120	.072912	-.792157	.941810	.832466
.975172	.553281	.281960	.041120	-.531948	.961174	.830944
.983584	.536758	.229755	.020745	-.332495	.974415	.830249
.989559	.518468	.183514	.008375	-.183342	.983754	.829930
.994116	.500234	.137923	.001921	-.075154	.990853	.829783
1.000000	.473882	0.000000	0.000000	0.000000	1.000000	.829714

WING WITH AN END-PLATED VERTICAL FENCE AT THE TIP (CONFIGURATION 7)

TRANSFORMATION CONSTANTS (INDEPENDENT VARIABLES)								ERRORS IN REQUIRED CONDITIONS						
A	B	C	D	E	F	G	H	WING CLOSURE	VERTICAL FENCE CLOSURE	END PLATE CLOSURE	VERTICAL FENCE LENGTH	INBOARD F. P. LENGTH	OUTBOARD E. P. LENGTH	SQUARE ROOT OF SUM OF SQR. ERRORS
0.0	.52130	.54294	.58218	1.0	1.14747	1.47707	2.43088	.00000	-.00000	-.00000	-.00000	-.01000	-.01000	.01414213
-1.026	.983	-.361	-.420	.493	.031									
-3.274	.932	1.894	-.303	-1.703	.066									
4.405	-1.702	-1.651	.606	1.024	-.103									
-1.047	.086	-.330	-.021	-.025	.274									
1.399	.296	-.016	.215	.025	.051									
-1.374	-.013	.001	-.065	-.022	-.025									
0.0	.47505	.48904	.52443	1.0	1.17816	1.45634	2.36421	-.00789	.00079	.00735	-.00118	-.00380	.00060	.01153781
0.0	.48342	.49494	.53399	1.0	1.17705	1.46319	2.37662	.00481	-.00047	-.00470	.00031	.00340	-.00015	.00755147
0.0	.48003	.49421	.53105	1.0	1.17707	1.46129	2.37304	-.00306	.00020	.00304	-.00003	-.00262	.00009	.00505302
0.0	.48168	.49366	.53191	1.0	1.17702	1.46184	2.37422	.00201	-.00008	-.00201	-.00002	.00185	-.00006	.00338821
-.565	1.078	-.875	-.489	1.048	.029									
-4.563	1.041	3.142	-.311	-2.869	.110									
5.057	-1.899	-2.215	.689	1.415	-.142									
-1.067	.022	-.352	-.048	-.032	.289									
1.455	.329	-.030	.232	.036	.076									
-1.418	-.015	.004	-.067	-.031	-.035									
0.0	.48110	.49399	.53169	1.0	1.17705	1.46169	2.37387	-.00005	.00001	.00005	-.00000	-.00004	.00000	.00008534
0.0	.48111	.49398	.53170	1.0	1.17705	1.46169	2.37387	-.00000	-.00000	.00000	.00000	-.00000	.00000	.00000230
0.0	.48111	.49398	.53170	1.0	1.17705	1.46169	2.37387	-.00000	.00000	.00000	.00000	-.00000	.00000	.00000017
0.0	.48111	.49398	.53170	1.0	1.17705	1.46169	2.37387	.00000	-.00000	-.00000	.00000	.00000	-.00000	.00000000
0.0	.48111	.49398	.53170	1.0	1.17705	1.46169	2.37387	.00000	-.00000	-.00000	.00000	.00000	-.00000	.00000000

TRANSFORMATION CONSTANTS IN EXPONENTIAL FORMAT

0.	4.81112684E-01	4.93977693E-01	5.31698355E-01	1.00000000E+00	1.17705067E+00	1.46168791E+00	2.37387251E+00

WING CLOSURE INTEGRAL =	.00000000
VERTICAL FENCE CLOSURE INTEGRAL =	-.00000000
END-PLATE CLOSURE INTEGRAL =	-.00000000
VERTICAL FENCE LENGTH =	.08000000
INBOARD END-PLATE LENGTH =	.04000000
OUTBOARD END-PLATE LENGTH =	.04000000
INDUCED DRAG INTEGRAL =	2.19156002
DISTANCE FROM A TO B =	1.10268060
DISTANCE FROM B TO C =	-.08821445
DISTANCE FROM C TO D =	-.04410722
DISTANCE FROM D TO E =	.08821445
DISTANCE FROM E TO F =	-.04410722
DISTANCE FROM F TO G =	.08821445
DISTANCE FROM G TO H =	-1.10268060

LIFTING EFFICIENCY K = 1.147450

WING WITH END-PLATED VERTICAL FENCE AT THE TIP

RATIO OF VERTICAL FENCE LOAD TO WING LOAD = .0315956
RATIO OF INBOARD END PLATE LOAD TO WING LOAD = -.0013157
RATIO OF OUTBOARD END PLATE LOAD TO WING LOAD = .0033445

WING SPAN COORDINATE	WING LOADING	ELLIPTICAL WING LOADING	END PLATE SPAN COORDINATE	END PLATE LOADING, STARTING OUTBOARD	VERTICAL FENCE COORDINATE	VERTICAL FENCE LOADING
0.000000	1.199605	1.273240	1.000000	.000134	0.000000	1.254657
.018754	1.199425	1.273016	.996513	.429512	.013990	1.231701
.033296	1.199037	1.272534	.985596	.871859	.024840	1.216199
.052395	1.198197	1.271491	.966347	1.350202	.039099	1.198883
.079379	1.196371	1.269222	.937389	1.826500	.059269	1.180318
.118539	1.192380	1.264262	.896445	2.352061	.088619	1.165716
.175856	1.183646	1.253397	.839197	2.895147	.131833	1.141442
.259223	1.164652	1.229717	.755127	3.512252	.195518	1.107921
.354651	1.133329	1.190477	.675858	3.969901	.270259	1.072217
.427941	1.101810	1.150762	.619370	4.237236	.329611	1.045762
.489184	1.070236	1.110495	.580185	4.401879	.381001	1.024356
.542517	1.038136	1.069578	.553002	4.507440	.427490	1.005768
.590140	1.006004	1.027888	.533703	4.578159	.470733	.989293
.633403	.973915	.985261	.518990	4.629733	.511787	.974389
.673226	.940849	.941480	.500000	4.693323	.551423	.960682
.673226	.940849	.941480	.500000	-2.112319	.551423	.960682
.710291	.907448	.896243	.475551	-2.036029	.590276	.947915
.745151	.873804	.849118	.456654	-1.977446	.628945	.935882
.778310	.838748	.799442	.431973	-1.900745	.668081	.924549
.810303	.802545	.746133	.397537	-1.793334	.708504	.913863
.841828	.764737	.687225	.348887	-1.640654	.751432	.903545
.874088	.722783	.618499	.281658	-1.460474	.799052	.893681
.910074	.673658	.527688	.194957	-1.218152	.856632	.884194
.938556	.631889	.439429	.114625	-.872612	.904137	.878608
.957901	.601831	.365544	.067413	-.605203	.935807	.876076
.971446	.578170	.302089	.037669	-.402408	.957142	.874927
.981010	.560652	.246955	.018869	-.250341	.971750	.874403
.987885	.543806	.197593	.007574	-.137954	.982059	.874162
.993161	.524320	.148651	.001729	-.056718	.989898	.874051
1.000000	.495521	0.000000	0.000000	-.000000	1.000000	.873999

WING WITH AN END-PLATED VERTICAL FENCE AT THE TIP (CONFIGURATION 7)

TRANSFORMATION CONSTANTS (INDEPENDENT VARIABLES)								ERRORS IN REQUIRED CONDITIONS						
A	B	C	D	E	F	G	H	WING CLOSURE	VERTICAL FENCE CLOSURE	END PLATE CLOSURE	VERTICAL FENCE LENGTH	INBOARD E. P. LENGTH	OUTBOARD E. P. LENGTH	SQUARE ROOT OF SUM OF SQR. ERRORS
0.0	.48111	.49398	.53170	1.0	1.17705	1.46169	2.37387	.00000	-.00000	-.00000	.00000	.00000	-.00000	.00000000
0.0	.48111	.49398	.53170	1.0	1.17705	1.46169	2.37387	.00000	-.00000	-.00000	.00000	.00000	-.00000	.00000000

TRANSFORMATION CONSTANTS IN EXPONENTIAL FORMAT

0.	4.81112684E-01	4.93977693E-01	5.31698356E-01	1.00000000E+00	1.17705067E+00	1.46168791E+00	2.37387251E+00

WING CLOSURE INTEGRAL =	.00000000
VERTICAL FENCE CLOSURE INTEGRAL =	-.00000000
END-PLATE CLOSURE INTEGRAL =	-.00000000
VERTICAL FENCE LENGTH =	.08000000
INBOARD END-PLATE LENGTH =	.04000000
OUTBOARD END-PLATE LENGTH =	.04000000
INDUCED DRAG INTEGRAL =	2.19156002
DISTANCE FROM A TO B =	1.10268060
DISTANCE FROM B TO C =	-.08821445
DISTANCE FROM C TO D =	-.04410722
DISTANCE FROM D TO E =	.08821445
DISTANCE FROM E TO F =	-.04410722
DISTANCE FROM F TO G =	.08821445
DISTANCE FROM G TO H =	-1.10268060
LIFTING EFFICIENCY K =	1.14745067

WING WITH END-PLATED VERTICAL FENCE AT THE TIP

RATIO OF VERTICAL FENCE LOAD TO WING LOAD = .0315956
RATIO OF INBOARD END PLATE LOAD TO WING LOAD = -.0013157
RATIO OF OUTBOARD END PLATE LOAD TO WING LOAD = .0033445

WING SPAN COORDINATE	WING LOADING	ELLIPTICAL WING LOADING	END PLATE SPAN COORDINATE	END PLATE LOADING, STARTING OUTBOARD	VERTICAL FENCE COORDINATE	VERTICAL FENCE LOADING
0.000000	1.199605	1.273240	1.000000	.000134	0.000000	1.254657
.018754	1.199425	1.273016	.996513	.429512	.013990	1.231701
.033296	1.199037	1.272534	.985596	.871859	.024840	1.216199
.052395	1.198197	1.271491	.966347	1.350202	.039099	1.198883
.079379	1.196371	1.269222	.937389	1.826500	.059269	1.180318
.118539	1.192380	1.264262	.896445	2.352061	.088619	1.165716
.175856	1.183646	1.253397	.839197	2.895147	.131833	1.141442
.259223	1.164652	1.229717	.755127	3.512252	.195518	1.107921
.354651	1.133329	1.190477	.675858	3.969901	.270259	1.072217
.427941	1.101810	1.150762	.619370	4.237236	.329611	1.045762
.489184	1.070236	1.110495	.580185	4.401879	.381001	1.024356
.542517	1.038136	1.069578	.553002	4.507440	.427490	1.005768
.590140	1.006004	1.027888	.533703	4.578159	.470733	.989293
.633403	.973915	.985261	.518990	4.629733	.511787	.974389
.673226	.940849	.941480	.500000	4.693323	.551423	.960682
.673226	.940849	.941480	.500000	-2.112319	.551423	.960682
.710291	.907448	.896243	.475551	-2.036029	.590276	.947915
.745151	.873804	.849118	.456654	-1.977446	.628945	.935882
.778310	.838748	.799442	.431973	-1.900745	.668081	.924549
.810303	.802545	.746133	.397537	-1.793334	.708504	.913863
.841828	.764737	.687225	.348887	-1.640654	.751432	.903545
.874088	.722783	.618499	.281658	-1.460474	.799052	.893681
.910074	.673658	.527688	.194957	-1.218152	.856632	.884194
.938556	.631889	.439429	.114625	-.872612	.904137	.878608
.957901	.601831	.365544	.067413	-.605203	.935807	.876076
.971446	.578170	.302089	.037669	-.402408	.957142	.874927
.981010	.560652	.246955	.018869	-.250341	.971750	.874403
.987885	.543806	.197593	.007574	-.137954	.982059	.874152
.993161	.524320	.148651	.001729	-.056718	.989898	.874051
1.000000	.495521	0.000000	0.000000	-.000000	1.000000	.873999

Program OVERLAY Structure

Computer Program 55VD does not require the use of OVERLAY on a CDC 6000 series computer or on a IBM 7094. Nevertheless, Program 55VD has been written to exploit the FORTRAN OVERLAY feature if it should become either necessary or desirable to do so. Figure 5 presents an OVERLAY STRUCTURE FOR Program 55VD that saves considerable core storage, yet does not require excessive transfer of large subroutines into core storage.

REFERENCES

1. Munk, Max M.: The Minimum Induced Drag of Aerofoils, NACA Report 121, 1921.

2. Mangler, W.: The Lift Distribution of Wings with End Plates, NACA TM No. 856, 1938 (Translated by J. Vanier from: Die Auftriebsverteilung am Tragflugel mit Endscheiben, Luftfahrtforschung, V. 14, No. 11, Nov. 20, 1937, pp 564-569).

3. Mangler, W.: Lift Distribution on Wings with Lateral Plates, Douglas Aircraft Company Report No. LB-33497, December, 1966 (Translated by G. deMontalvo from Luftfahrtforschung, V. 16, 1939, pp 219-228).

4. Falkner, V. M., and Darwin, C.: The Design of Minimum Drag Tip Fins, A.R.C. R&M 2279, 1945.

5. Küchemann, D., and Kettle, D. J.: The Effect of End Plates on Swept Wings, A.R.C. CP104, June, 1951.

6. Weber, J.: Theoretical Load Distribution on a Wing with Vertical Plates, A.R.C. R&M 2960, March, 1954.

7. Cone, C. D., Jr.: The Theory of Induced Lift and Minimum Induced Drag of Nonplanar Lifting Systems, NASA Technical Report R-139, 1962.

8. Lundry, J. L., and Lissaman, P.B.S.: A Numerical Solution for the Minimum Induced Drag of Nonplanar Wings, AIAA J. of Aircraft, V. 5, No. 1, January-February, 1968.

9. Douglas Aircraft Company, Inc., Technical Proposal 3388T (Unsolicited), Load Distribution to Produce Minimum Induced Drag for Nonplanar Wings, March, 1967.

10. Lundry, J. L.: A Numerical Solution for the Minimum Induced Drag, and the Corresponding Loading, of Nonplanar Wings - User's Manual for Computer Program 55VD, McDonnell-Douglas Corporation, Report No. DAC-66898, February 29, 1968.

11. Lundry, J. L.: A Numerical Solution for the Minimum Induced Drag, and the Corresponding Loading, of Nonplanar Wings - Final Report, McDonnell-Douglas Corporation, Report DAC-66900, March 29, 1968.

12. von Karman, Th. and Burgers, J. M.: General Aerodynamic Theory - Perfect Fluids, Division E, Vol. II, p. 138 of Aerodynamics Theory, W. F. Durand, Editor-in-Chief, Julius Springer 1934-36 and Dover Publications.

13. Nance, C. C.: Announcement of Computer Program ZROG, FLOTRAN, Automatic Flowcharter for Fortran IV, Douglas Aircraft Company Memorandum A2-260-AKE1-ZR06-1, July 29, 1966.

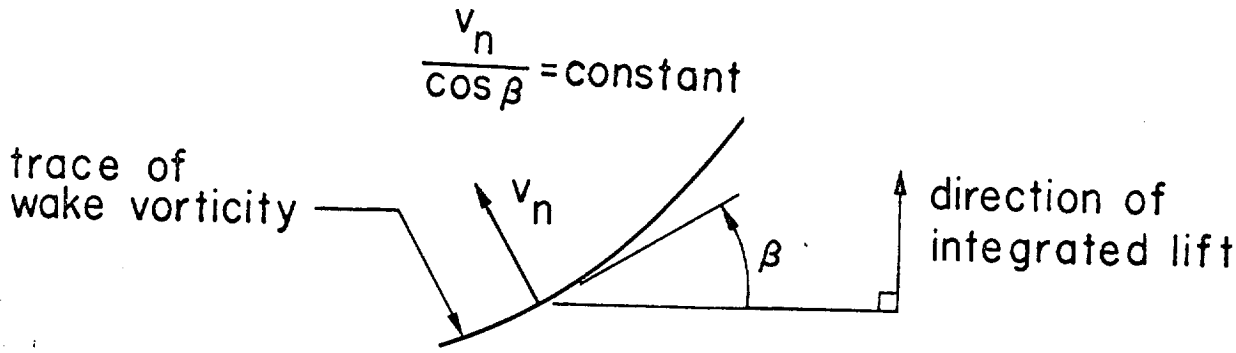


Figure 1. Munk's Minimum Induced Drag Criterion.

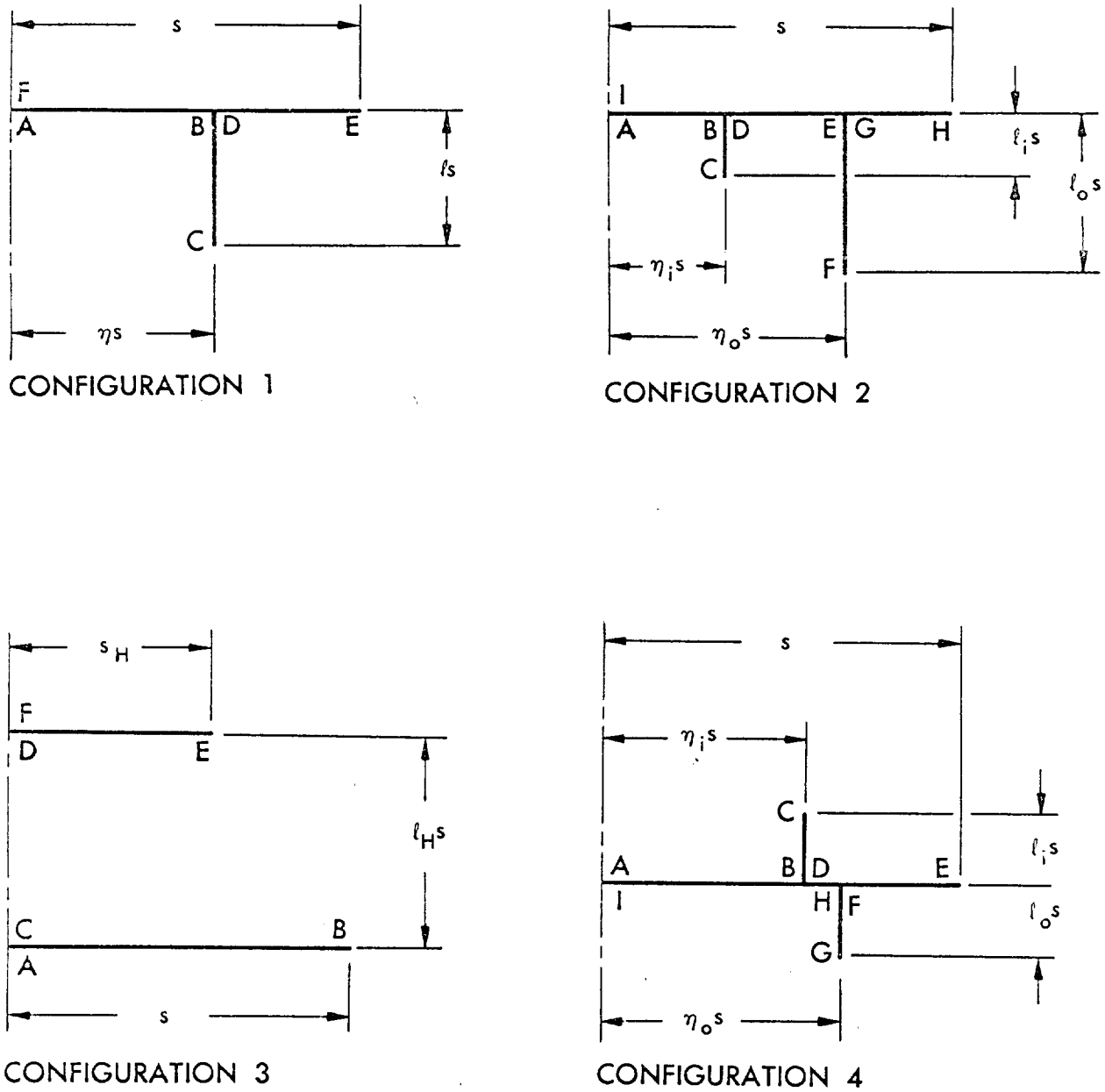
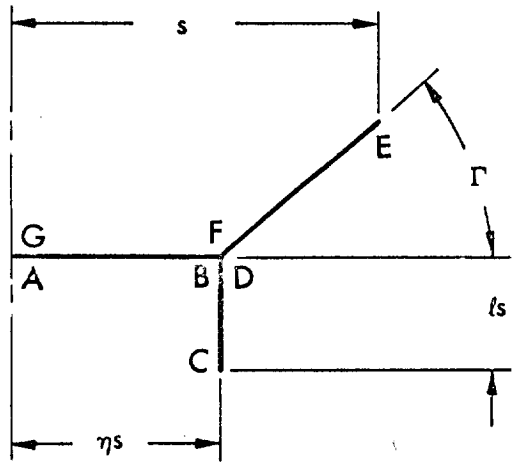
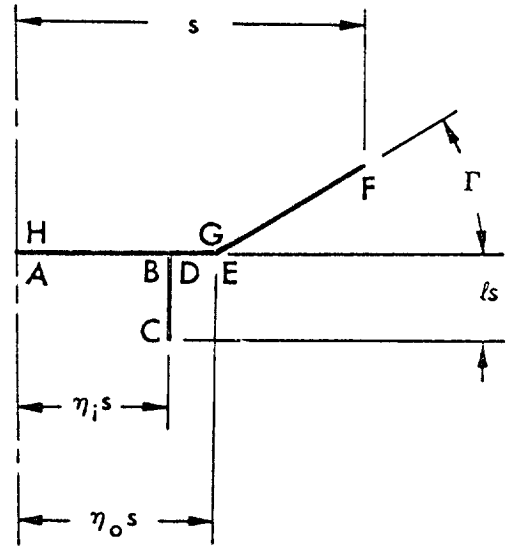


Figure 2a. Configurations 1-4

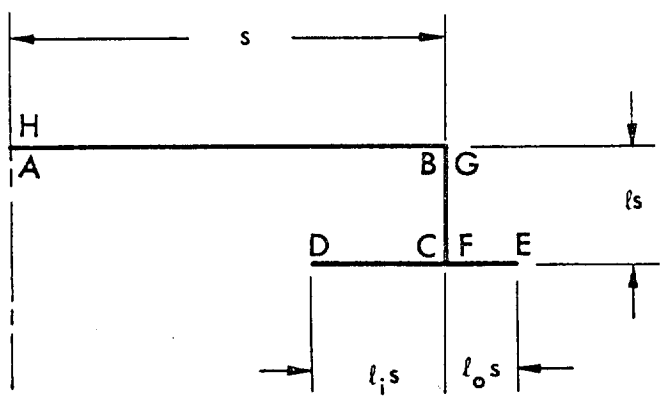
Figure 2. Configurations Analyzed by Program 55VD



CONFIGURATION 5



CONFIGURATION 6



CONFIGURATION 7

Figure 2b. Configurations 5-7

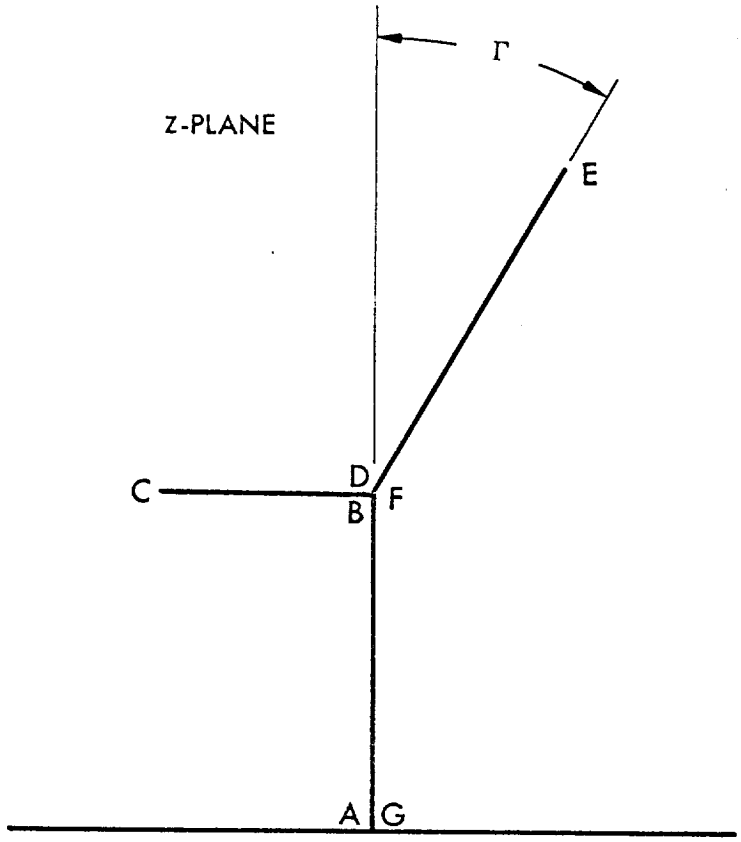


Figure 3a. Real Plane

$$\frac{dz}{d\zeta} = \frac{(\zeta-c)(\zeta-e)}{[(\zeta-a)(\zeta-b)(\zeta-g)]^{1/2}} (\zeta-d)^{\frac{\Gamma}{\pi}-\frac{1}{2}} (\zeta-f)^{-\frac{\Gamma}{\pi}}$$

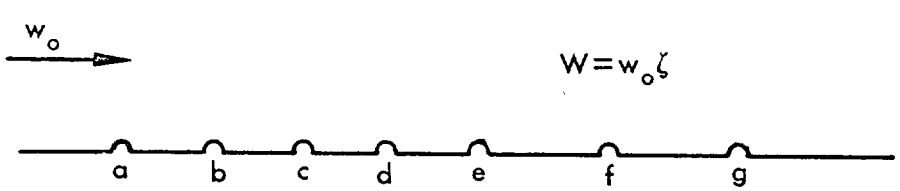


Figure 3b. Auxiliary Plane

Figure 3. Mapping Planes for Configuration 5.

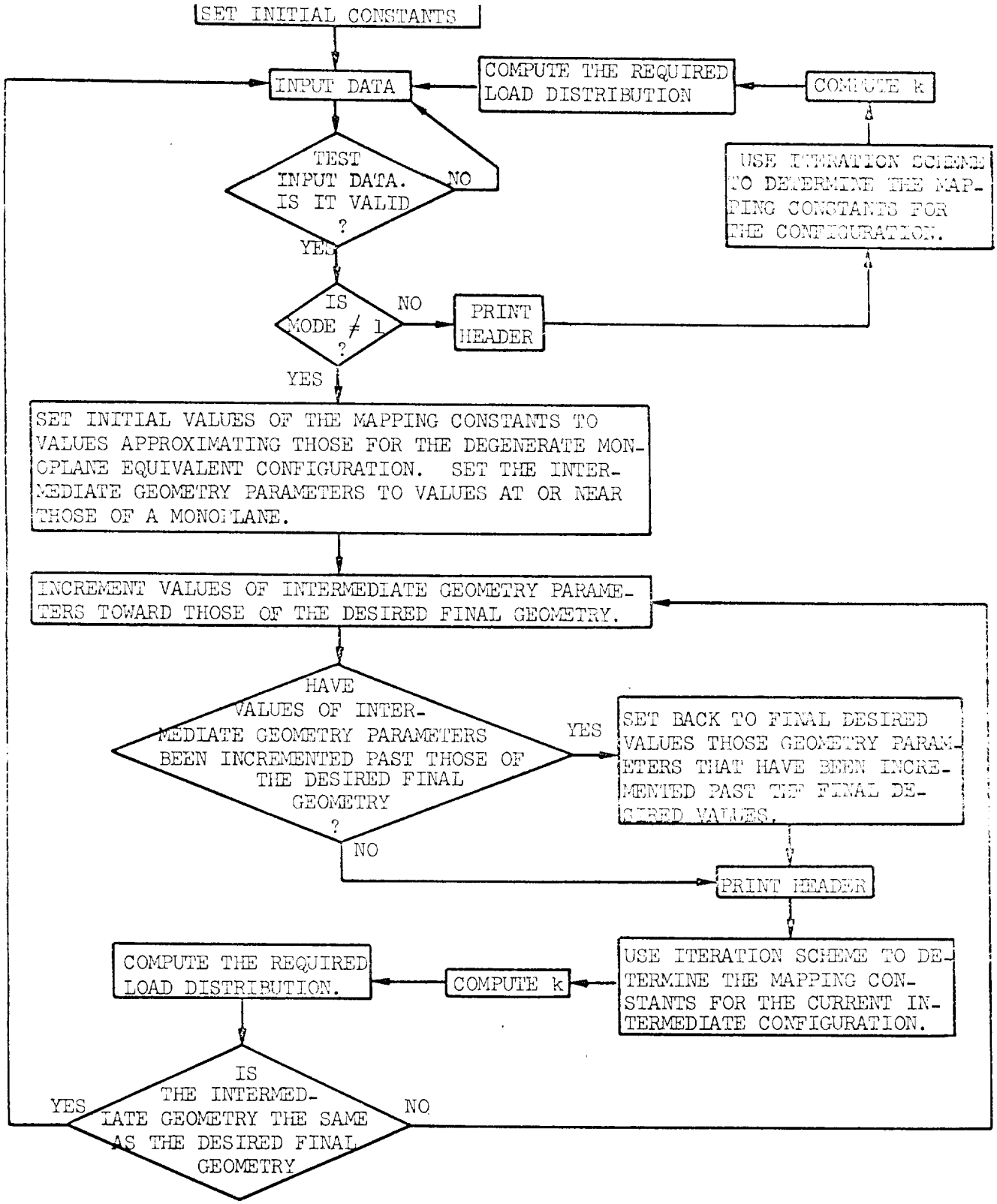


Figure 4. Overall Logical Flow of Computer Program 55VD

Deck name is followed by deck identification in parenthesis.

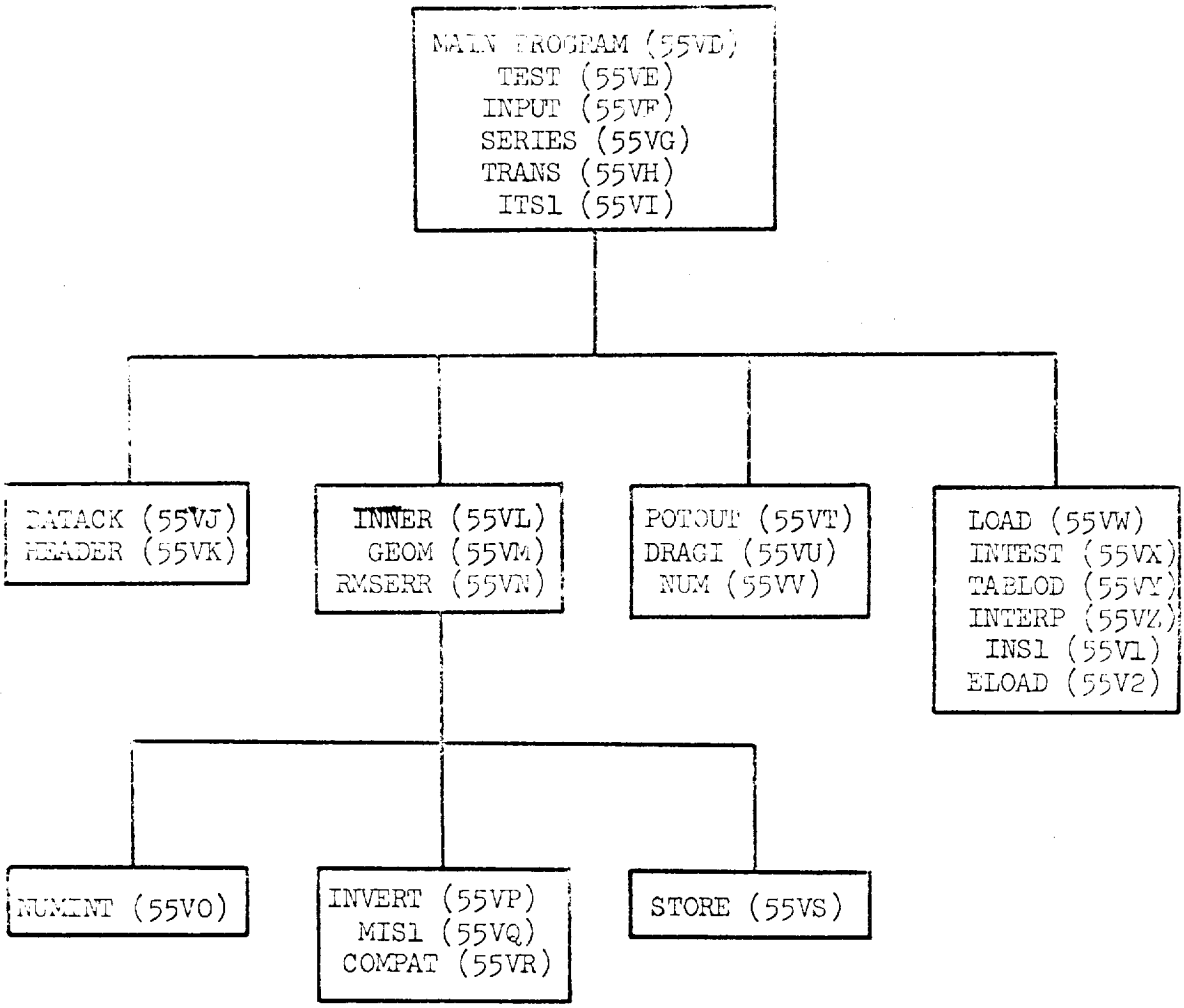


Figure 5. OVERLAY Structure for Computer Program 55VD